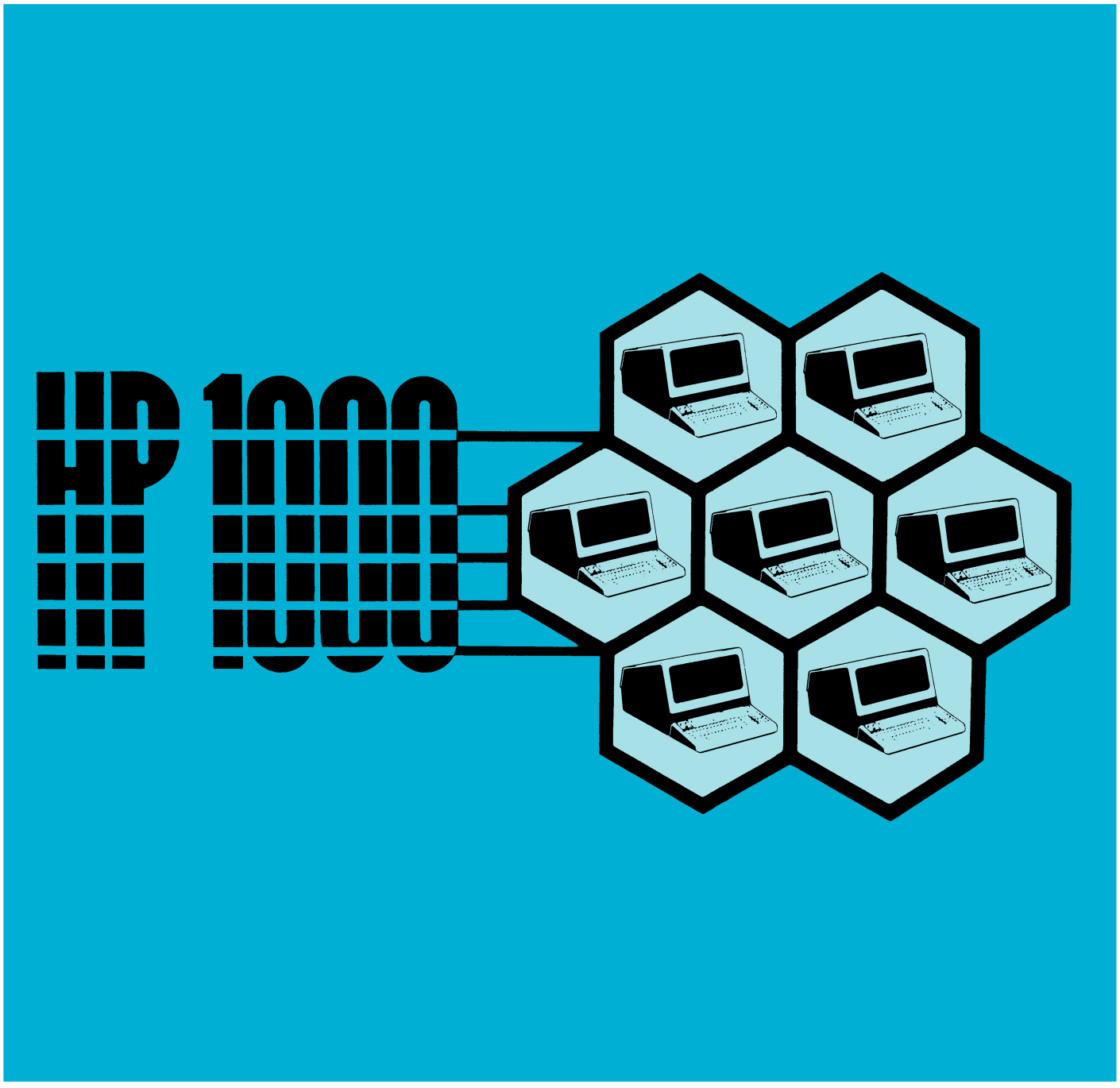
HEWLETT PACKARD

HP 92068A
RTE-IVB Terminal User's

Reference Manual



RTE-IVB Terminal User's

Reference Manual

HEWLETT-PACKARD COMPANY
Data Systems Division
11000 Wolfe Road
Cupertino, California 95014

MANUAL PART NO. 92068-90002
Printed in U.S.A. January 1983
U1283

PRINTING HISTORY

The Printing History below identifies the Edition of this Manual and any Updates that are included. Periodically, Update packages are distributed which contain replacement pages to be merged into the manual, including an updated copy of this Printing History page. Also, the update may contain write-in instructions.

Each reprinting of this manual will incorporate all past Updates, however, no new information will be added. Thus, the reprinted copy will be identical in content to prior printings of the same edition with its user-inserted update information. New editions of this manual will contain new information, as well as all Updates.

To determine what manual edition and update is compatible with your current software revision code, refer to the appropriate Software Numbering Catalog, Software Product Catalog, or Diagnostic Configurator Manual.

Third Edition	Feb 1980	
Update 1	Apr 1980	
Update 2	Jul 1980	
Update 3	Oct 1980	
Update 4	Jan 1981	
Update 5	Jul 1981	
Reprinted	Jul 1981	Updates 1 thru 5 incorporated
Update 6	Oct 1981	
Reprinted	Oct 1981	Update 6 incorporated
Update 7	Jan 1983	READT/WRITT enhancements, add LUPRN
Reprinted	Jan 1983	Update 7 incorporated
Update 8	Dec 1983	Manual errors corrected

Preface

This manual describes the scope, format, and use of the interactive operator commands and utility programs available with the RTE-IVB operating system. It is intended to be the primary reference source for operators who will be using the computer to perform data entry, file manipulation, program development, and other interactive tasks.

The manual is organized into six separate sections to facilitate the referencing of information.

Chapter 1 gives a general descripton of the RTE-IVB operating system features.

Chapter 2 describes the operating environment when under session control and not under session control. It describes how to log on and off the system, restricted access to operator commands based upon command capability level, and cartridge ownership when operating under session control.

Chapter 3 describes the file management commands that allow the operator to interactively manipulate data files and disc cartridges. File creation, global parameters, and procedure files are also described.

Chapter 4 describes the RTE and break-mode commands. Program states and I/O processing are also briefly discussed.

Chapter 5 describes the interactive editor and its commands for creating and editing files.

Chapter 6 describes the utility programs available in RTE-IVB for program development, and status checking.

For additional information on the RTE-IVB operating system or any of its subsystems, refer to the appropriate document shown on the Documentation Map in this manual. Some available manuals offering other levels of information that may be directly relevant to system operation and user applications are briefly summarized below:

* RTE-IVB Programmer's Reference Manual:

  Describes the features provided by the RTE-IVB Operating System which allow the user to programmatically use system services and/or control system resources. Detailed descriptions of the calling sequences which invoke system action, as well as, FORTRAN-IV and Assembly Language examples are provided.

* RTE-IVB Batch and Spooling Reference Manual:

  Describes the uses and requirements of the Batch Spool Monitor Subsystem for those who wish to use the batch processing and spooling features.

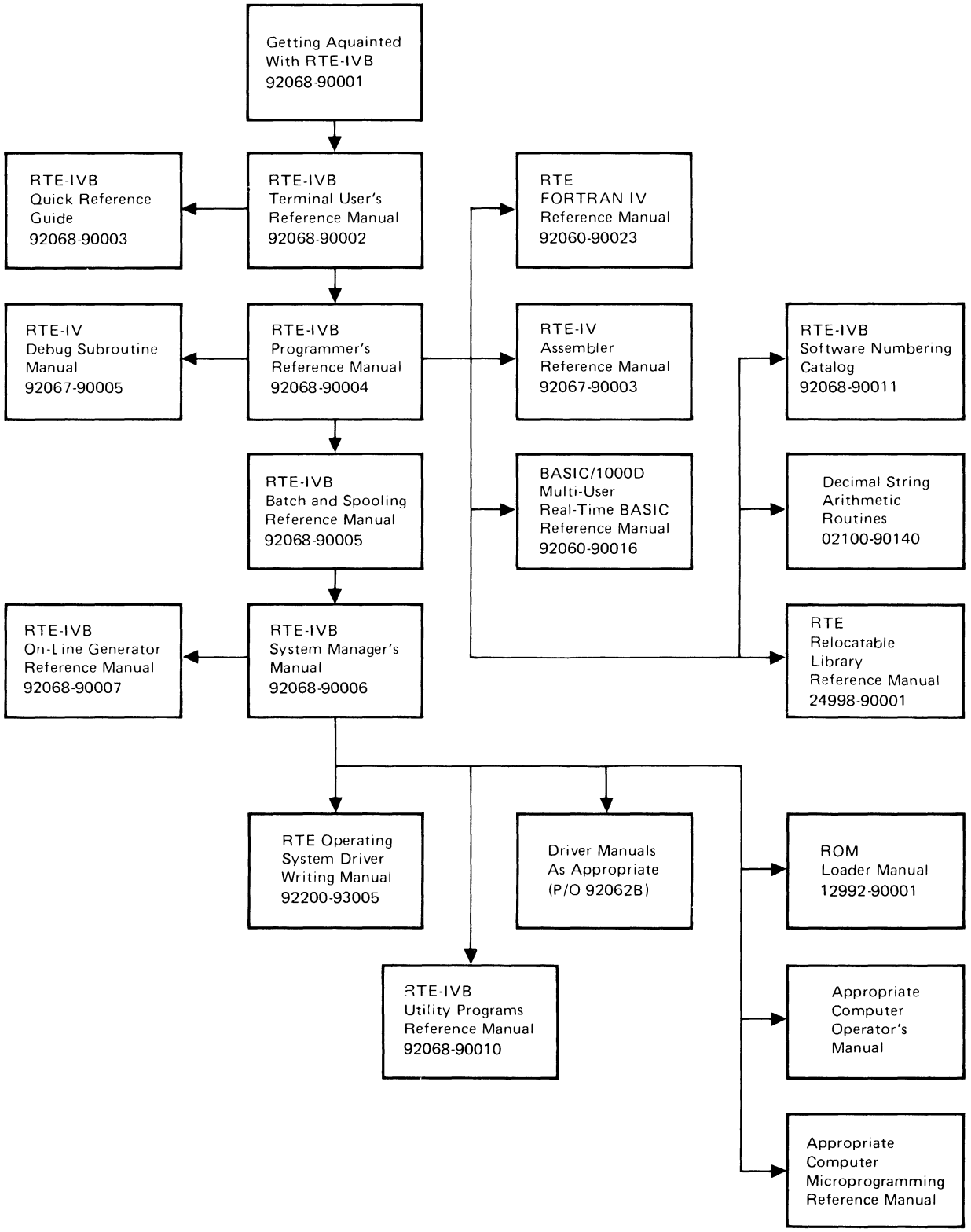
* RTE-IVB System Manager's Manual:

  Describes planning, generating, and maintaining a system. In conjunction with the RTE-IVB On-Line Generator Reference Manual, all system generation information is available.

* RTE-IVB On-Line Generator Reference Manual:

  Describes detailed procedures for generating a new RTE-IVB operating system without shutting down the existing RTE-IVB system. Complete examples of each phase and heavily annotated worksheets are provided. The manual is primarily intended for System Managers and system programmers who are involved in the design and maintenance of total system configurations.

RTE-IVB Documentation Map



7700-444

Table of Contents

Chapter 2 Operating Environment

Chapter 3 File Management Commands

Chapter 4 System and Break Mode Commands

Chapter 5 Interactive Editor Commands

Chapter 6 Interactive Utilities

List of Illustrations

List of Tables

Chapter 1
General Description

Introduction

RTE-IVB is a disc based operating system that provides the supervisory functions necessary to coordinate requests for, and allocation of, system services and resources. RTE-IVB processes all decision and scheduling tasks internally unless overridden by user intervention. User requests for system action can be made by a "call" from within a program or interactively via an operator command.

Additional software packages available with RTE-IVB complement the the basic operating system. Program development capabilities are enhanced by an interactive editor, language compilers and assembler, and the interactive loader. File management and batch processing are provided through the file management system, spooling of input and output is provided by the spool monitor; and the multi-user environment is enhanced by the session monitor.

Operating System Features

As the major control element within the operating environment, RTE-IVB provides the user with various services and automatically handles the machine-related functions associated with each service. The major features of RTE-IVB are briefly summarized below:

* Real-Time Programming
* Multiprogramming with Timeslicing
* Program Partitions
* Extended Memory Area
* Program Segmentation
* Resource Management

Following is a brief discussion of these features. The descriptions that follow are intended to only be an introduction to the system. For a more detailed description of the system and its features refer to the RTE-IVB Programmer's Reference Manual and the RTE-IVB System Manager's Manual.

Real-Time Programming

The Real-Time Executive allows event and time scheduling of programs, in addition to program, operator and batch scheduling. In RTE-IVB, external events are serviced after they cause interrupts to the operating system. Time scheduling is accommodated through a Time List which is checked every 10 milliseconds by the operating system.

Multiprogramming with Timeslicing

The Multiprogramming feature of RTE-IVB allows several programs to be active concurrently; each program executes during the unused central processor time of the others. Programs can be memory or disc resident.

Disc-resident programs are swapped in and out of partitioned memory in accordance with availability of system resources, program priority, and time scheduling criteria. Programs may be scheduled by pre-determined time intervals, an external event, operator command, or another program.

At a given program priority level, program scheduling may be conducted on a linear (ie. FIFO), or circular with timeslicing basis. Linear scheduling prevails above a specified priority level (i.e. at a higher priority), and circular scheduling prevails below it.

Linear scheduling means that at a given priority level, the first program entering the queue has priority over another program subsequently entering. Circular scheduling means that all programs in the queue (at the same priority level) take turns for a specified quantum or slice of time regardless of the order in which they entered the queue. Programs of higher priority levels always have precedence over lower priority programs.

Program Partitions

Partitions are blocks of physical memory reserved for disc resident programs. Partitions are defined during system generation but may be redefined during the reconfiguration process at system boot-up (see RTE-IVB System Manager's Manual for system reconfiguration details).

Partitions can be defined as real-time or background. In addition, they may also be specified as mother partitions (refer to discussion below). Real-time programs run in real-time partitions unless no real-time partition is available, in which case they will run in any available background partition large enough to accommodate them. Background programs run in background partitions.

Mother partitions are larger than the logical address space of 32K words. Mother partitions are used for running Extended Memory Area (EMA) programs and may be composed of subpartitions. When the mother partition is not in use, other programs can run in the subpartitions.

Extended Memory Area (EMA)

An Extended Memory Area is an area for program data that extends beyond the logical address space of a program up to the available physical memory. This allows an EMA program to process large amounts of data from main memory rather than swapping in blocks of data from disc. Without EMA, the program and its data would be limited to approximately 27K words.

Program Segmentation

The program segmentation feature allows large programs to be separated into a main program and related segments, thereby allowing them to execute in partitions smaller than their total size. While a segmented program is executing, its main is kept in the program's partition and its segments are swapped back and forth between disc and memory when needed.

Resource Management

The RTE-IVB operating system allows cooperating programs to manage common system resources. A resource is defined to be any element within the RTE-IVB environment that can be accessed by a user's program, e.g., an I/O device, a file, a program, or a subroutine.

This feature allows the user to manage a specific resource shared by a particular set of programs, so that no two of these programs can access the shared resource at the same time.

Session Monitor

The Session Monitor provides an enhanced multi-user environment by controlling access to the system, separating users, and controlling system resources.

User access to the system is controlled by the LOGON program. To gain access to the system the user must provide LOGON with the appropriate user identification information as stored in the Account File on disc.

Separation of users is aided by a private, group and system cartridge structure. A user is prohibited from accessing other users' or groups' disc cartridges.

Control of system resources is aided by confining users to only those resources defined in their respective Session Switch Tables. The Session Switch Table (SST) provides a mapping between session logical unit numbers which the user addresses to system logical unit numbers where the request is processed.

File Management

File management can be accomplished either programmatically via FMP calls or interactively via the program FMGR. This manual describes the interactive FMGR commands. For a description of FMP calls see the RTE-IVB Programmer's Reference Manual.

Interactive File Manager (FMGR) commands can be used to create and manipulate disc or non-disc files; develop programs and procedure files; save programs and procedure files; initialize disc cartridges; and manipulate disc cartridges.

Batch Processing

Batch processing, the entry of one or more jobs for processing in a single job stream, is controlled by FMGR commands. The jobs themselves may be stored on disc or on a peripheral input device. In either case, batch job operation is controlled through FMGR commands that delimit the job and that may be included with the job. See the RTE-IVB Batch and Spooling Reference Manual for batch processing details.

Spool Monitor

The Spool Monitor can increase the throughput of a job stream that is limited by slow peripheral devices by temporarily spooling the data onto a disc file. This allows input and output to be performed independently of each other and of job processing. See the RTE-IVB Batch and Spooling Reference Manual for spooling details.

Interactive Editor

The Interactive Editor (EDITR) can be used to create, edit and merge ASCII files. The Interactive Editor commands are covered in detail in Chapter 5 of this manual.

System Utility Programs

Standard system utilities are on-line programs that run under the RTE operating system and are called by the user to perform various program preparation, system status and housekeeping processes. The presence of any utility program in the system is optional, depending upon site-specific requirements. The programs available are:

* Compile Utility (COMPL)
* Relocating Loader (LOADR)
* Compile and Load Utility (CLOAD)
* On-Line Generator (RT4GN)
* Disc Backup
* Disc Update
* System Status Program (WHZAT)
* Merge Utility (MERGE)
* Disc Cartridge Save/Restore Utilities (WRITT, READT)
* Interactive Debugging (DBUGR)
* Soft Key Programs (KEYS and KYDMP)
* Track Assignment Table Log Program (LGTAT)

For a more detailed description of the On-Line Generator, see the On-Line Generator Manual; for the Disc Backup and Disc Update utilities, see the RTE-IVB Utility Programs Reference Manual. All other interactive utility programs will be discussed in more detail in Chapter 6 of this manual. A short description of each of the utilities follows.

Compile Utility (COMPL)

The compile program enables the user to invoke any HP supported compiler or assembler. The utility will select the appropriate compiler or assembler by reading the control statement (first statement required in the program). Optionally, the programmer can specify a new control statement when running COMPL.

Relocating Loader (LOADR)

The Relocating Loader program accepts relocatable programs and outputs absolute load modules in conformance with loader control commands specified by the user. Other command parameters cause the loader to list system status information; i.e., currently available programs; or purge unwanted, permanently loaded programs from the system.

Compile and Load Utility (CLOAD)

CLOAD performs a composite function. It invokes the appropriate source translator (as COMPL does) and inputs the relocatable results to the loader (LOADR) for the creation of an executable memory-image program.

On-Line Generator (RT4GN)

The On-Line Generator permits use of an existing RTE-IVB System to configure a new RTE-IVB System according to user specifications. Generation can be directed from an answer file, logical unit or operator console.

Disc Backup Utility

The Disc Backup programs can be used either on-line or off-line to transfer data from disc to magnetic tape or vice-versa, copy data from disc to disc, verify successful transfer or copy operation, and to initialize a disc cartridge.

Disc Update Utility

The Disc Update process can be used to replace disc cartridge files with files stored on an HP minicartridge tape. The primary purpose is to update master software discs with either HP software distributed on minicartridges or user-written program modifications.

System Status Program (WHZAT)

The WHZAT program provides status information regarding the current system environment. A list of active programs associated with the user's session can be displayed on the user terminal. This is the default mode without any parameters. A list of all active programs and the current status of each program can be displayed. A list of programs having "father-son" relationship can be displayed. And finally, a list of all partitions with their sizes and current status (occupied or non-occupied).

Merge Utility (MERGE)

The Merge utility provides a quick and simple way for files to be merged. Taking its input from a command file or interactively from a terminal, MERGE can concatenate files within an existing file or a file created by MERGE.

Disc Cartridge Save/Restore Utilities (WRITT,READT)

The disc cartridge Save/Restore utilities allow the user to save and restore peripheral FMP disc cartridges through the use of magnetic tape. Saving user files or cartridges on mag tape is accomplished through the WRITT program and restoring them to the system is accomplished through the READT program.

Interactive Debugger (DBUGR)

The DBUGR Subroutine can be appended to a user program by specifying an option in the Relocating Loader. It can then aid the user in checking for logical errors in a program through interactive control commands. Debugging is performed at the Assembly Language level. See the subset of DBUGR control commands described in Chapter 6 of this manual or the DBUGR Reference Manual for a complete description of all DBUGR functions.

Soft Key Programs (KEYS and KYDMP)

The KEYS and KYDMP programs are used to create user-defined command sets for programming the soft keys on the HP 2645A/48A Display Station. Soft Keys provide the capability to enter entire sequences of commands with a single key stroke. The advantages are speed of entry and a significant reduction in operator errors during terminal entry sessions.

Track Assignment Table Log Program (LGTAT)

The LGTAT program logs and displays the status of the system and auxiliary disc tracks (LU 2 and 3).

System Configuration Display Utility (LUPRN)

LUPRN supplies the current system device configuration and identifies each device by its true name. The list can be sorted by system LU number, session LU number, or by device type.

Chapter 2
Operating Environment

Introduction

Provided with the RTE-IVB Operating System are two software packages which handle multi-terminal operation: the Session Monitor and the Multi-Terminal Monitor. The features provided by each of these multi-terminal handlers will be described in this section along with a description of their respective operating environments.

Operating with the Session Monitor

Significant features provided by the Session Monitor include:

* Controlled access to the system.
* Separation of users.
* Control of system resource usage.

The LOGON program controls access to the system by checking the identification provided by a user when attempting to log-on with that stored in the Account File on disc. The System Manager initializes and maintains the Account File which includes entries for all authorized users as well as other session related information such as the maximum number of users authorized to access the system at one time.

Separation of users is accomplished through the private, group and system cartridge structure; the automatic program renaming feature which allows each user to have his own copy of a program instead of waiting for another user to finish with the program first; and the capability to interactively set up spool files for I/O operations.

Control of system resource usage is accomplished through the Session Switch Table (SST) which is defined for each user. The SST defines which I/O devices are accessible to a session user. It also provides a mapping function from session logical unit numbers that the user references to system logical unit numbers where the requests are processed. Additional Session Monitor features which provide control over system resource usage include command capability levels which restrict users to a prespecified subset of the operator commands; a spare cartridge pool where users may request temporary disc area and return it when they are finished; and the RTE-IVB timeslicing of programs feature.

In addition to the above features, the Session Monitor also provides the following:

* A HELP utility which provides the user a simple means (through the HE command) of receiving error explanations and guidance in possible corrective action from the system.

* User HELLO files which the system may automatically transfer to when the user logs on.

* Station configuration independence which allows the user to reference certain I/O devices by the same logical unit number regardless of the terminal on which he has logged on. For example, a session terminal's CRT can always be referenced as LU 1, its left cartridge tape unit as LU 4, and right cartridge tape unit as LU 5.

* User message files which allow session users to send and receive message files and/or strings from other session users via the File Manager SM and ME commands.

* Access to 254 system logical unit numbers (63 at one time) through the session to system logical unit number mapping feature of the Session Switch Table.

Terminals which are not controlled by the Session Monitor terminal handlers (PRMPT and RPN) will not have access to most of these features. Examples of non-session terminals might be the system console, which can be set to operate in or out of the session environment, or dedicated application terminals. Operation under both of these types of terminals are further discussed later in this section.

Log-On

The Log-on sequence is initiated by pressing any key on a currently enabled (but not currently Logged-on) session terminal.

```
+-------------------------------------------------------+
|                          NOTE                         |
|                                                       |
|  If the terminal is a  Multipoint terminal, the ENTER |
|  key must be pressed.                                 |
+-------------------------------------------------------+
```

The PRMPT program then issues a prompt string to the terminal requesting the user's identification. The prompt string may be defined by the System Manager or the System Manager may use the "PLEASE LOG-ON:" default prompt.

The LOGON program then picks up the user name input and checks the Account File for a match. The format for the Log-on is as follows:

```
+-----------------------------------------------------+
|                                                     |
|        "PLEASE LOG-ON:"  XXXX.ZZZZ                  |
|                                                     |
|  where:                                             |
|                                                     |
|  XXXX is the user account name defined in the Account |
|       File (up to 10 characters) upper or lower case. |
|                                                     |
|  .    is the separator for user and group information.|
|                                                     |
|  ZZZZ is the group name the user is associated with  |
|       (up to 10 characters) upper or lower case. The |
|       default group name is GENERAL.                |
|                                                     |
+-----------------------------------------------------+
```

An example of a valid Log-on might be:

```
PLEASE LOG-ON:  BROWN.MANUF
```

In addition to the user identification, a password may be required for additional security. If the Account File indicates that a password is required by the user, the user is prompted with "PASSWORD?". The format of the password is as follows:

```
+-----------------------------------------------------+
|                                                     |
|    PASSWORD?  YYYY                                  |
|                                                     |
|  where:                                             |
|                                                     |
|  YYYY is the user password defined in the Account    |
|       File (up to 10 characters) upper or lower case.|
|       Echo suppression is used to maintain privacy  |
|       of the password.                              |
|                                                     |
+-----------------------------------------------------+
```

The password can also be provided with the user identification. If the password was COST, the user would type in:

```
PLEASE LOG-ON:  BROWN.MANUF/COST
```

In this case, the user would not be prompted for the password again. The password will be echoed back to the terminal, however.

If the identification information provided by the user is not recognized as being valid by LOGON, the following error message will be returned to the user's terminal:

```
SESSION     9  LGON 04 NO SUCH USER
               UNABLE TO COMPLETE LOG-ON
```

The user can try to Log-on again by repeating the process. After a second unsuccessful attempt the user should contact his System Manager for assistance.

After a successful Log-on, the system will send a message to the session terminal containing the session identification number, the time of day that the session started, and the user's cumulative connect time to date. An example would be:

```
SESSION    9 ON  9:54 AM  THU., 15  FEB., 1980
PREVIOUS TOTAL SESSION TIME: 00 HRS., 40 MIN., 19 SEC.
```

A line containing the session identification number, the time of day that the session started, and the user's name is also sent to the system console:

```
SESSION    9 ON  9:54 AM  THU., 15 BROWN.MANUF
```

The following messages may then be issued to the session terminal:

a. A system message file defined by the System Manager. An example being:

```
THE SYSTEM WILL BE DOWN FOR MAINTENANCE AT 10:00 PM
TONIGHT.  PLEASE LOG-OFF BEFORE THEN.
```

b. A "MESSAGES WAITING" if there is pending mail in the user's Message File. For more information on the Message File, see the report and clear messages command (ME) and the send message command (SM) in Chapter 3 of this manual.

The user's copy of FMGR is then scheduled and the following services performed by LOGON:

a. The User Hello File is transferred to if it has been defined. This file can be unique to a single user or can be shared among several users. It is a procedure file which is transferred to when the user logs on. It allows the System Manager to tailor the Session Monitor to individual users and their applications.

b. The default log and list devices are made to be the user's session terminal.

c. The severity code is set to 0 (see the SV command in Chapter 3 for a description of severity codes).

Log-Off

The LGOFF program is responsible for severing the user's connection with the system and returning all temporary resources allocated to the session. This includes:

a. Terminating any active session programs,

b. Releasing main program ID segments assigned to temporary session programs,

c. Optionally dismounting private and/or group disc cartridges,

d. Releasing all System Available Memory (SAM) or disc tracks allocated to any temporary session programs,

e. Updating all accounting information stored in the session Account File such as total CPU and connect time.

The Log-off sequence is initiated by entering the following command:

```
:EX ,SP [,RG [,KI ]]
    ,RP

where:

SP = save private cartridges (see note below).

RP = release private cartridges (see note below).

RG = Release group cartridges; default is to not
     release group cartridges.

KI = Abort any active session programs.


                *** note ***

        The specification of either SP or
        RP is only required if the user
        has a private cartridge currently
        mounted to him.
```

Before a user can be logged off, all active session programs must be terminated. The following diagnostic is issued if any session programs are active:

```
PROG1
PROG2
ABOVE SESSION PROGRAMS ACTIVE
OK TO ABORT ? (Y OR N)
```

If a "N" is entered, the Log-off sequence is terminated. If a "Y" is entered, the active programs (PROG1 and PROG2) will be aborted and the Log-off sequence continued.

If the kill option (KI) is specified, the above message will not be issued and all active programs will be aborted.

The LGOFF program then updates the session's user and group accounting information by storing the log-off time, cumulative connect time, and CPU usage in the Account File.

An example Log-off message to the session terminal is:

```
SESSION    9  OFF 10:09 AM  THU., 15  FEB., 1980
CONNECT TIME:              00 HRS.,  04 MIN.,  20 SEC.
CPU USAGE:                 00 HRS.,  00 MIN.,  19 SEC.,   40 MS.
CUMULATIVE CONNECT TIME:   06 HRS.,  04 MIN.,  03 SEC.
END OF SESSION
```

The first line of the Log-off message is also sent to the system console.

Command Capability Levels

Command capability levels define which set of interactive operator commands can be executed by the user. A user has access to all commands at or below his capability level. For example, a user with capability level 30 has access to all commands available at that level, as well as those of the lower capability levels of 10 and 20.

A user's command capability level is defined by the System Manager and stored in the Account File. As a user's needs change, his command capability level can be altered. Table 2-1 shows the default command capability level assignments provided with the Session Monitor.

Table 2-1. Command Capability Level Assignments

FMGR	SYSTEM and BREAK-MODE	CAPABILITY LEVEL						
		--	--	--	--	--	--	--
EX SY TR HE	HE OP		1					
		--						
AC MC TE CL ME WH DC *SL ?? DL SM ** LI	$BL RS *TO +BR *SL UP *EQ ST WH FL TE $QU TI			10				
			--					
AN DP RN CN DU ST CO LL SV CR PK CT PU					20			
				--				
+OF RT SL RP RU SP	+GO RT +SS +OF RU SZ					30		
					--			
CA PA SE IF							40	
						--		
LO SL	AS ON UR IT PR							50
							--	
IN OF	BL GO SS BR LU TM DN IF TO EQ QU							60
								--

* Single parameter only

\+ Program must be under session's control

$ No parameters permitted

Cartridge Ownership

There are four categories of cartridges in the session environment:

1. System cartridges,
2. Private cartridges,
3. Group cartridges, and
4. Non-session cartridges.

System cartridges can be further broken down into two categories: those available to all users with unrestricted access (global cartridges), and those available to all users but with restricted access (LU 2 and 3). Cartridges with system logical units 2 and 3 are primarily read-only, while system global cartridges have both read/write access. System LU 2 is where the disc resident operating system resides. The remaining tracks on LU 2 and LU 3 are where stored programs reside, the user message files, system manager defined procedure files and other system manager files reside.

Private cartridges can be accessed solely by the user who has it mounted to his session and the System Manager (who has access to all cartridges in the system).

Group cartridges are accessible to only members of a group who have it mounted to their sessions and the System Manager.

Non-session cartridges are only accessible to non-session users and the System Manager. Since the system console has the capability of operating in both the session and non-session environment by switching modes, non-session cartridges can be accessed through the system console.

Session Related Data Structures

There are four session related tables which the user should have some familiarity with. These are:

1. Account File
2. Session Control Block (SCB)
3. Session Switch Table (SST)
4. Configuration Table

The Account File is a disc resident file created and maintained through the account set-up program (ACCTS) by the System Manager. It contains the user, group and password names for all users under session control, their respective command capability levels, group and user SSTs, and their cumulative connect and CPU times. Also contained are group cumulative connect and CPU times, and the LUs of cartridges in the spare disc cartridge pool.

The Session Control Block (SCB) is created by the LOGON program and stored in System Available Memory (SAM) after the user has successfully logged-on. It contains the user's command capability level, user and group name, CPU usage, SST, and a list of cartridges mounted to the user.

The Session Switch Table (SST) located in the SCB is built up from the user and group SSTs associated with the user, and the Configuration Table - all of which are taken from the Account File. Each entry in the SST describes a session LU which the user addresses and an associated system LU where the I/O request is actually directed. Only those devices entered into the user's SST are accessible by him. The group SST defines those devices available to all members of a specific group, and the user's SST defines those additional devices made accessible to the user.

The Configuration Table is similar to the group and user SSTs in that each entry describes a Session LU which the user addresses and an associated System LU where the I/O request is actually directed. However, the Configuration Table relates session and system LUs only for those devices associated with a specific terminal (station). Examples would be the right and left cartridge tape units on a HP 2645A/48A terminal or a terminal auxiliary printer.

System Console Operation

The system console can operate in either a session or non-session mode when the Session Monitor software modules are generated into the system.

Immediately following boot-up, the system console will be in a non-session mode. The user must strike a key to get the system's attention. The system will prompt the user by issuing an asterisk (*) to the terminal CRT.

EN (Enable the System Console to be a Session Terminal)

In order for the system console to operate under session, the terminal must be enabled as a session terminal.

```
+-----------------------------------------------+
|                                               |
|   *EN,security code [,option]                 |
|                                               |
|  where:                                       |
|                                               |
|  security code = the FMP master security code |
|     specified in the File Manager IN command  |
|     (see Chapter 3).                          |
|                                               |
|  option = 0 or 1.                             |
|                                               |
|                                               |
|     0 = master security code not required in  |
|         "OP" commands (default).              |
|                                               |
|     1 = master security code is required in   |
|         "OP" commands.                        |
|                                               |
+-----------------------------------------------+
```

Once this command has been entered, the system console is enabled as a session terminal. To reset the system console to operate non-session, the user can re-boot the system or reset it through the ACCTS program (see the RTE-IVB System Manager's Manual for more details).

Once logged on, all break mode requests are checked for session capability restrictions. Any attempt to execute a command requiring a capability greater than the caller possesses will result in a "CAPABILITY ERROR".

OP (Allow System Level Operator Command)

The "OP" command provides a method of entering a single system level command from a low capability session. This may be required to correct a system level problem while LU 1 is being used as a session terminal. Unauthorized use of this command can be prevented by requiring knowledge of the master security code when session mode is enabled.

```
S=01 COMMAND ?OP [,security code [,command ]]

where:

 S=01 COMMAND ? is the break-mode prompt.

 security code  is the FMP master security
    code defined by the File Manager IN
    command (see Chapter 3).  If specified
    in the EN command, the security code is
    required.

 command  is the system command to be
    executed.
```

AB (Abort the Currently Executing Batch Job)

The AB command which aborts the currently executing batch job is only valid when input at the system console. See the RTE-IVB Batch and Spooling Reference Manual for details on batch job processing.

```
      --  --
     |    |
  AB | ,0 |
     | ,1 |
     |    |
     --  --

where:

0   is the default case.  It terminates the
    current batch job immediately if it is
    executing, scheduled or operator sus-
    pended.  If the job is I/O, memory or
    disc suspended, the batch job is termin-
    ated the next time it is scheduled.  Disc
    tracks are not released.

1   immediately terminates the batch job and
    releases all disc tracks.  If suspended
    for I/O, a system generated CLEAR request
    is issued to the driver.
```

The AB command is only valid when input from the following modes:

```
+-----------------------------------------------+
|                                               |
| System Console in non-session mode:           |
|                                               |
|    *AB                                        |
|                                               |
| System Console in session mode:               |
|                                               |
|    S=01 COMMAND ?OP,master security code,AB   |
|                                               |
+-----------------------------------------------+
```

Non-Session Terminals in a Session Environment

When the Session Monitor is generated into the system, any non-session terminal (other than the system console) will be intended for running dedicated application programs. These programs will normally be scheduled from the system console to run at the non-session dedicated terminal or can be set up to automatically run by the operator causing an asynchronous interrupt on the terminal.

An example of a dedicated application would be data entry functions such as processing of sales orders or material purchases.

Because of the dedicated application nature of these terminals, the user will not be able to input any interactive operator commands.

Operating without the Session Monitor in the System

If the user does not desire the features provided by Session Monitor in his system, he may include the Multi-Terminal Monitor (MTM) instead. The following section assumes that MTM is included in the user's operating system and describes interactive operation under the MTM environment.

Multi-Terminal Monitor (MTM)

When operating under MTM, the user can input any interactive operator command which is not solely related to the Session Monitor. Chapters 3 and 4 of this manual describe operator commands available to the user under MTM.

MTM will perform several services for the user in conjunction with a terminal's copy of FMGR. A terminal with Logical Unit number xx has its own copy of FMGR if a program exists named FMGxx. For example, the copy of FMGR for Logical Unit 09 would be FMG09.

If a copy of FMGR named FMGxx does not exist for a terminal, the standard MTM prompt (xx>) will be issued and the user will be conversing with the RTE operating system. The remainder of this manual section assumes that the terminal has its own copy of FMGR named FMGxx.

Available MTM Services

In an MTM environment, a user terminal with its own copy of FMGR has access to three services:

1. Automatic scheduling of FMGxx when the user terminal interrupts the operating system.

2. Automatic renaming of user programs scheduled from FMGxx.

3. Automatic execution of transfer file named HI.

If a copy of FMGR called FMGxx exists for the terminal, striking a key on the terminal causes FMGxx to be scheduled for execution.

If the terminal's copy of FMGR is available for execution (not busy or suspended), two events will occur:

First, the prompt

```
xx>FMGxx
```

will be issued to the terminal.

Second, control is transferred to a file named HI, which must exist on LU 2, the system disc.

The HI file is a procedure file usually written by the System Manager and placed on Logical Unit 2. Although the file may be empty, it must nevertheless exist or an FMGR -006 error will result. When the end of the HI file is reached, control is transferred to the interrupting terminal.

AB (Abort Currently Executing Batch Job)

The format of the AB command for aborting the currently executing batch job is the same as with session (see the AB command description in the OPERATING WITH THE SESSION MONITOR section in this manual). The command must be input from the system console to be valid. For details on batch job processing, see the RTE-IVB Batch and Spooling Reference Manual.

Interactive Operator Commands

Interactive operator commands are the means by which the user communicates with the computer. The commands provided with the RTE-IVB system allow the user to create and edit files; in conjunction with the compiler and loader utilities, develop programs and schedule programs to be run.

File Manager (FMGR) Commands

Some functions of the FMGR commands include:

* Creating files
* Listing file contents
* Sending messages
* Requesting error messages
* Mounting cartridges
* Dismounting cartridges
* Listing mounted cartridges
* Listing file directories of mounted cartridges
* Running programs
* Terminating programs

The FMGR prompt is a colon (:). Chapter 3 describes all the FMGR commands.

RTE System Commands

Some of the functions of the RTE system commands include:

* Schedule programs to run at a specific time
* Schedule programs to run immediately
* Abort a program
* Suspend a program
* Reschedule a suspended program
* Change a program's priority
* Assign a program to a partition
* Check the status of program
* Check the status of a device
* Down a device
* Up a device
* Set the real time clock
* Read the time
* Set the timeslice quantum and fence

All interactive RTE system commands can be input by one of two ways:

1. On the system console (in the non-session mode) or an MTM terminal in the operating mode. Operator mode prompt is the asterisk (*) on the system console, (lu>) on an MTM terminal.

2. On any terminal in the FMGR mode. The command must then be preceeded by SY. The FMGR prompt is a colon (:). An example would be:

```
:SYTI
```

   where TI is the RTE system command to print the time.

Most of the RTE system commands can also be input during break-mode operation. Break-mode is described in the next subsection in this manual.

Chapter 4 of this manual describes all the RTE system commands.

RTE Break Mode Commands

Break mode is reached by interrupting the system (by striking any key) while the system does not have a read request pending from the terminal (i.e., no prompt is issued to the CRT). An example of break mode operation would be to interrupt the system during a list process to the terminal CRT. After the interrupt is recognized by RTE, the break mode prompt

```
S=xx COMMAND ?        (for a Session Terminal)

        or

lu>                   (for an MTM Terminal)
```

where xx is the session identification number and lu is the logical unit number of the MTM terminal, is issued to the terminal CRT. At this point, any break mode command can be issued. Also, many of the RTE system commands can be issued (see Chapter 4 for more details).

For instance, if the session identification number is 15 and the operator wishes to abort a listing on his CRT, the user could issue the BR command in response to the break mode prompt. For example:

```
S=15 COMMAND ?BR
```

Or if on an MTM terminal which has been defined as logical unit 15, the operator can abort a listing on his CRT by issuing the command:

```
15>BR
```

Interactive Editor Commands

The interactive editor commands are used for editing files. Before the user can edit files he must be in the edit mode by either running the program EDITR (or a copy of EDITR) or transferring to a procedure file which runs EDITR (or a copy of EDITR). Some of the features of the RTE Editor include:

* File creation capability.
* User defined tab stops.
* Merging of files.
* Selectively displaying file contents by one or more lines at a time.
* Editing of files using line edits, character edits or pattern edits.
* Finding and editing of text patterns.
* Capability of moving blocks of text.
* Automatic line sequencing in columns 73 through 80.
* User defined windows to facilitate pattern searching and editing.

The EDITR prompt is a slash (/). Chapter 5 describes the interactive editor commands available under RTE.

Chapter 3
File Management Commands

Introduction

File Management is performed through program calls to the File Management Package (FMP) library and by interactive operator commands to the program FMGR. The FMP calls mainly control input to and output from disc files or peripheral devices treated as files. The file management capability is increased by using FMGR for interactive program development, disc cartridge manipulation, and batch job control.

This Chapter describes the FMGR operator commands available under RTE-IVB. For more details on FMP program calls, the reader is referred to the RTE-IVB Programmer's Reference Manual. For more details on batch job control, refer to the RTE-IVB Batch and Spooling Reference Manual.

Certain terms used in discussing file organization for file management are defined in Table 3-1. The File Manager Commands are summarized in Table 3-2. Note that the table categorizes the commands by function, whereas, their descriptions in this section are organized alphabetically.

Table 3-1. File Management Terms

DISC	A rotating random access storage device on which files may be stored and from which they may be retrieved. Discs may be physically permanent or they may be removable.
CARTRIDGE	A cartridge is a set of contiguous cylinders on a disc unit. Cartridges contain disc files with a directory of the files stored on each cartridge. All files on the same FMP cartridge must have unique names. The system disc on logical unit 2 contains the RTE operating system, and may contain FMP files.
CYLINDER	A cylinder is the area that passes under all heads during one revolution of the disc surfaces.
TRACK	A subdivision of the disc. It is the area that passes under one head during one revolution of the disc surfaces.
SECTOR	A further subdivision of the disc; specifically, a logical sector is a portion of a disc track consisting of 64 words. Two sectors make up one FMP block, the unit of information that may be physically transferred between the disc and the memory.
FILE	A disc file is a collection of records terminated by an end-of-file mark. Non-disc devices are treated by the File Management System as if the device were itself a file. The device, like a file, is a collection of records terminated by an end-of-file mark that depends on the device. Any file can have zero or more records and is designated by a name of six characters or less.

Table 3-1. File Management Terms (continued)

NON-DISC DEVICE	A peripheral device, not a disc, that may be treated as a file and controlled by the File Management Package. Non-disc devices include the line printer for output, magnetic tape and minicartridges for input or output, and terminals through which an operator can interact with the system.
LOGICAL UNIT	An integer assigned to each input-output device or disc cartridge at system generation by which the cartridge or device can be referenced.
BLOCK	A subdivision of a disc file containing 128 words (two logical disc sectors) that is the smallest unit that can be physically transferred between disc and memory during file access.
RECORD	A logical collection of 16-bit words in a file or device. A record may have zero or more words. A record is the smallest unit that may be accessed by an FMP call from a program.
EXTENT	An extension to a file automatically provided by FMP as needed. Each extent is the same length and type as the file and is identified by a positive integer called the extent number.
DATA CONTROL BLOCK	FMP uses an array called the Data Control Block as an interface between FMP and the user's program. It contains control information for the file and also serves as a buffer for the transfer of data between a file and the user's program. Space must be allocated for the Data Control Block in any program making an FMP file access call.
USER BUFFER	An area in the calling program to hold data during file access.
ID SEGMENT	A block of words, associated with each resident program, that is used by the system to keep track of the program's name, software priority field, current scheduling status and other characteristics. Every program must have its own ID segment.

Table 3-2. Summary of File Manager Commands

INFORMATIONAL COMMANDS			
COMMAND	CAPABILITY LEVEL	DESCRIPTION	PAGE NO.
??	10	Request explanation of FMGR error code.	3-30
HE	1	Request error explanation and possible corrective action.	3-64
WH	10	Display current system status.	3-125

FILE AND DEVICE MANIPULATION			
COMMAND	CAPABILITY LEVEL	DESCRIPTION	PAGE NO.
CN	20	Control non-disc device.	3-40
CO	20	Copy files from one cartridge to another.	3-42
CR	20	Create disc file.	3-44
CR	20	Create non-disc file.	3-46
CT	20	Control terminal.	3-49
DU	20	Transfer data to existing file.	3-58
LI	10	List file contents.	3-71
LL	20	Change list device.	3-74
LO	50	Change log device.	3-75
PU	20	Purge file and its extents from system.	3-85
RN	20	Rename file.	3-87
SP	30	Save program as disc file.	3-109
ST	20	Transfer data and create file.	3-111

Table 3-2. Summary of File Manager Commands (continued)

FMP CARTRIDGE MANIPULATION			
COMMAND	CAPABILITY LEVEL	DESCRIPTION	PAGE NO.
AC	10	Allocate cartridge from spare cartridge pool.	3-32
CL	10	List mounted cartridges.	3-38
CO	20	Copy files from one cartridge to another.	3-42
DC	10	Dismount cartridge.	3-51
DL	10	List contents of file directory.	3-54
IN	60	Initialize FMP cartridge.	3-67
MC	10	Mount cartridge.	3-76
PK	20	Pack one or all mounted cartridges.	3-83

PROCEDURE FILE RELATED COMMANDS			
COMMAND	CAPABILITY LEVEL	DESCRIPTION	PAGE NO.
**	10	Declare a line of comments.	3-31
CA	40	Calculate globals.	3-36
DP	20	Display parameters.	3-56
IF	40	Conditional skip.	3-65
PA	40	Pause and send message.	3-81
SE	40	Clear or set globals.	3-96
SV	20	Change severity code.	3-118
TR	1	Transfer control to a file or logical unit.	3-122

Table 3-2. Summary of File Manager Commands (continued)

PROGRAM AND RESOURCE CONTROL

COMMAND	CAPABILITY LEVEL	DESCRIPTION	PAGE NO.
OF	30/60	Terminate Program.	3-80
RP	30	Restore program and assign ID segment or release ID segment.	3-88
RT	30	Release disc tracks assigned to a program.	3-91
RU	30	Run program.	3-92
SP	30	Save program as disc file.	3-109

MESSAGE RELATED COMMANDS

COMMAND	CAPABILITY LEVEL	DESCRIPTION	PAGE NO.
AN	20	Send message to list device.	3-35
ME	10	Display contents of message file.	3-79
SM	10	Send message to session user.	3-107
TE	10	Send message to system console.	3-121

MISCELLANEOUS COMMANDS

COMMAND	CAPABILITY LEVEL	DESCRIPTION	PAGE NO.
EX	1	Terminate session or FMGR.	3-62
SL	10	Examine session switch table contents.	3-97
SL	30/50	Modify session switch table.	3-105
SL	30	Set-up Spool File for I/O Device.	3-99
SY	1	Execute system command from FMGR.	3-120

Files

A file is a collection of information logically organized into records. The information in files may be programs or the data used by the programs. Data may be in binary or ASCII code. Programs may be in the form of ASCII source code, or binary code in either relocatable or absolute form. Programs may also be in memory-image form, a form used by RTE for programs ready to be executed.

Files can be stored on disc or they may reference non-disc peripheral devices by name. FMGR can be used to control and access files whether they are disc files or non-disc files.

Types of Files

Eight file types are defined by the file system. Additional types may be defined by the user. Only the first four types differ in in format; all subsequent types differ only in the type of data the file system expects the file to contain. The file types may be divided into three categories as shown in Table 3-3. The first category contains one type, type zero. This type includes all non-disc devices defined as files and accessible by name. The second category contains two file types, types 1 and 2. These fixed-length record files are used for quick random access. The remaining file types all belong to the third category of files with variable-length records designed for sequential access. All files may be extended automatically as needed. Disc file record formats are illustrated in Appendix C.

Table 3-3. Categories of File Types

CATEGORY	TYPE	DESCRIPTION
Control	0	Non-disc files
Fixed-length, random access	1	Fixed-length 128 word record files
	2	Fixed user-defined record length files
Variable length, sequential access,	3	Variable-length records; any data type
	4	Source program file; ASCII
	5	Object program file; relocatable binary
	6	Executable program file; memory-image code
	7	Absolute binary
	8-32767	User defined data format

Type 0 Files

Type 0 files are used to reference non-disc devices by name. They afford a measure of device independence in that the standard file commands can be used to control the device. A directory entry is made for the device as if it were a file. A type 0 file is created with a FMGR command, not with an FMP call. The File Directory entry for a file of this type contains special entries that specify logical unit number and the operations allowed on the particular device.

Type 1 Files

Type 1 files have fixed length records of 128 words. Because the File Management Package transfers data to and from disc in 128-word blocks, this file type allows direct access between disc and the user's buffer area in his program, thereby eliminating the need to go through a packing buffer (the Data Control Block.). As a result, type 1 files have the fastest transfer rate. Any other file type, except type 0, may be opened and accessed as a type 1 file in order to take advantage of the faster transfer rate. However, if the files being transferred have less than 128-word logical records, the user must be able to recognize where his records begin and end within the 128-words, or if his records are longer, be able to work with part of a record at a time. The end-of-file is the last word of the last block.

Type 2 Files

The record lengths of type 2 files are fixed, but the length is defined by the user at file creation. Like type 1 files, the end-of-file is the last word of the last block. Only one logical record is transferred at a time, but unlike type 1 files, the transfer must go through a packing buffer (the Data Control Block). For this reason, files of type 2 and above have a slower transfer rate than type 1 files.

Type 3 Files

These files have variable length records, are extendable, and may contain data, source code, relocatable or absolute binary code. Only one logical record is transferred at a time and the transfer must be made through the packing buffer (Data Control Block). The first and last words of each record as written on disc always contain the number of words in the record (minus the two length words). A zero-length record, consisting of two zero words, can be used to separate groups of records into sub-files. The end-of-file is marked by a -1 as the first length word in the next record. Words following the end-of-file are undefined. However, FMP can write records beyond the end-of-file by replacing the end-of-file with a new record followed by an end-of-file mark.

Type 4 Files

This file type is the same as type 3, except the file system expects type 4 files to contain ASCII code. Typically, source program files are type 4.

Type 5 Files

This file type is the same as type 3, except the file system assumes type 5 files contain relocatable binary code. Typically object program files are type 5.

Type 6 Files

This type file is the same as a type 3 file, except the system assumes it contains a program in memory-image format that is ready to run. Type 6 files are created by the Save.Program (SP) command, which can copy a temporary program, created by the LOADR and stored on the system scratch tracks, or a permanently loaded program into a type 6 file on the FMP tracks of LU 2 or LU 3. These files are always accessed by the File Management Package as type 1 files.

The first two sectors of a type 6 file are used to record ID segment information for the program. As a result, this file type can be used for programs that do not have a permanent ID segment.

Type 7 Files

This file type is the same as type 3, except the system expects type 7 files to contain absolute binary code.

Type 8 and Greater Files

This file type is the same as type 3, but the content is user-defined. FMP does no special processing based on file type for types greater than 7. For instance, any checksums must be specifically requested. Content is also user-defined; it can be source, relocatable binary, memory-image format, and so forth.

File Access

Type 1 and type 2 files contain fixed length records, which makes it possible to calculate the position of a desired record. On the other hand, type 3 files and above contain variable length records, so the system must access the disc at least once, and in some cases several times, in order to position to the desired record location. For this reason, access takes longer for file types greater than type 2.

File Extents

All files can be automatically extended whenever a write request points to a location beyond the range of the currently defined file. The extent is created by FMP with the same name and size as the main file, and access continues. FMP numbers each extent starting with 1. The extent number and location is kept in the file directory entry for the file extent. When a file with extents is referenced by its file name, any extents are provided automatically. A file may have a maximum of 255 extents. Refer to the RTE-IVB Programmer's Reference Manual for a description of file extents.

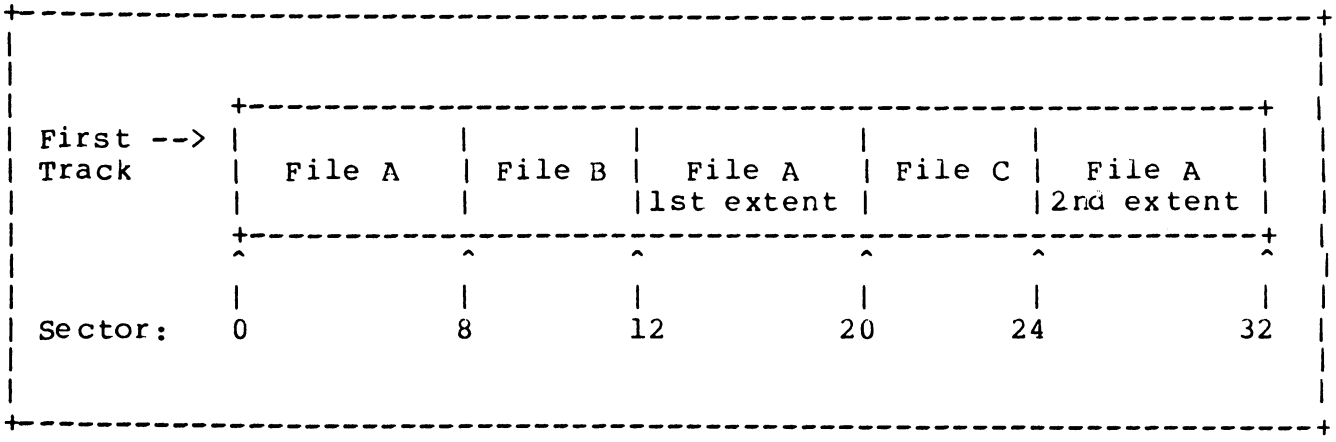
Cartridges

Files managed by the File Management Package, whether they are program files, data files, or spool files, are kept on FMP disc cartridges. An FMP disc cartridge is a logical entity that may correspond directly to a disc platter or may be a subdivision of the disc platter. On some cases, a cartridge may even cross platter boundaries. Care must be taken if the cartridge is partly on a removable platter and partly on a fixed platter.

Each cartridge is defined as a contiguous block of tracks. Each is assigned a logical unit number and a cartridge reference number (either of which may be used to reference the cartridge). Files on the same cartridge must have unique names. Duplicate names may be used as long as the duplicates are on separate cartridges so the file can be uniquely identified by its name and a cartridge identifier.

Locations of Files on Disc Cartridges

Files are stored in a consecutive fashion on disc cartridges. Generally, files are located contiguously, but if the file has extents, the extents may not be contiguous. For example:



Note that File A and its extents are not contiguous on the disc. As files are stored on disc, file directory entries are created in the file directory of the cartridge. The file directory starts on the last track of the cartridge.

Removable cartridges containing FMP files are interchangeable between drives of the same type within a system, or between drives on different systems provided that logical track 0 refers to the same physical track on every disc unit. (Refer to Figure 3-1 for an illustration of disc organization using one cartridge on the system disc starting at the first FMP track, and one on a peripheral disc starting at track 0.)

At cartridge initialization, the number of directory tracks for that cartridge is specified. The first track must be assigned at initialization; the number of sectors per track may be specified at this time, but is supplied by FMP as a default if not.

Files may cross track boundaries, but in a multi-cartridge environment, no one file may cross cartridge boundaries. Files are subject to being moved whenever a cartridge is packed. This causes files to be relocated within a cartridge and no absolute file addresses should be kept in any file or program.

Files always start on even sector boundaries and all accesses are multiples of 128 words addressed to even sectors.

Disc errors are passed back to the user for corrective action. Error codes are printed on the system log device when using the FMGR operator commands, or passed to the user program when calling a File Management Package library routine. You may report bad tracks to the system through the FMGR Initialization command. Bad tracks discovered by the system result in an error returned to the calling program.

Directories

Two types of directories are maintained by FMP: the FMP cartridge directory on the system disc, and the file directory on each cartridge.

Cartridge Directory

The cartridge directory is a master index to all currently mounted FMP cartridges. It is maintained in the system area of LU 2 (refer to Figure 3-1). Its length is two blocks and it has an entry for each currently mounted cartridge. The directory has room to describe up to 63 cartridges using four words for each. Appendix C shows the format for the cartridge directory.

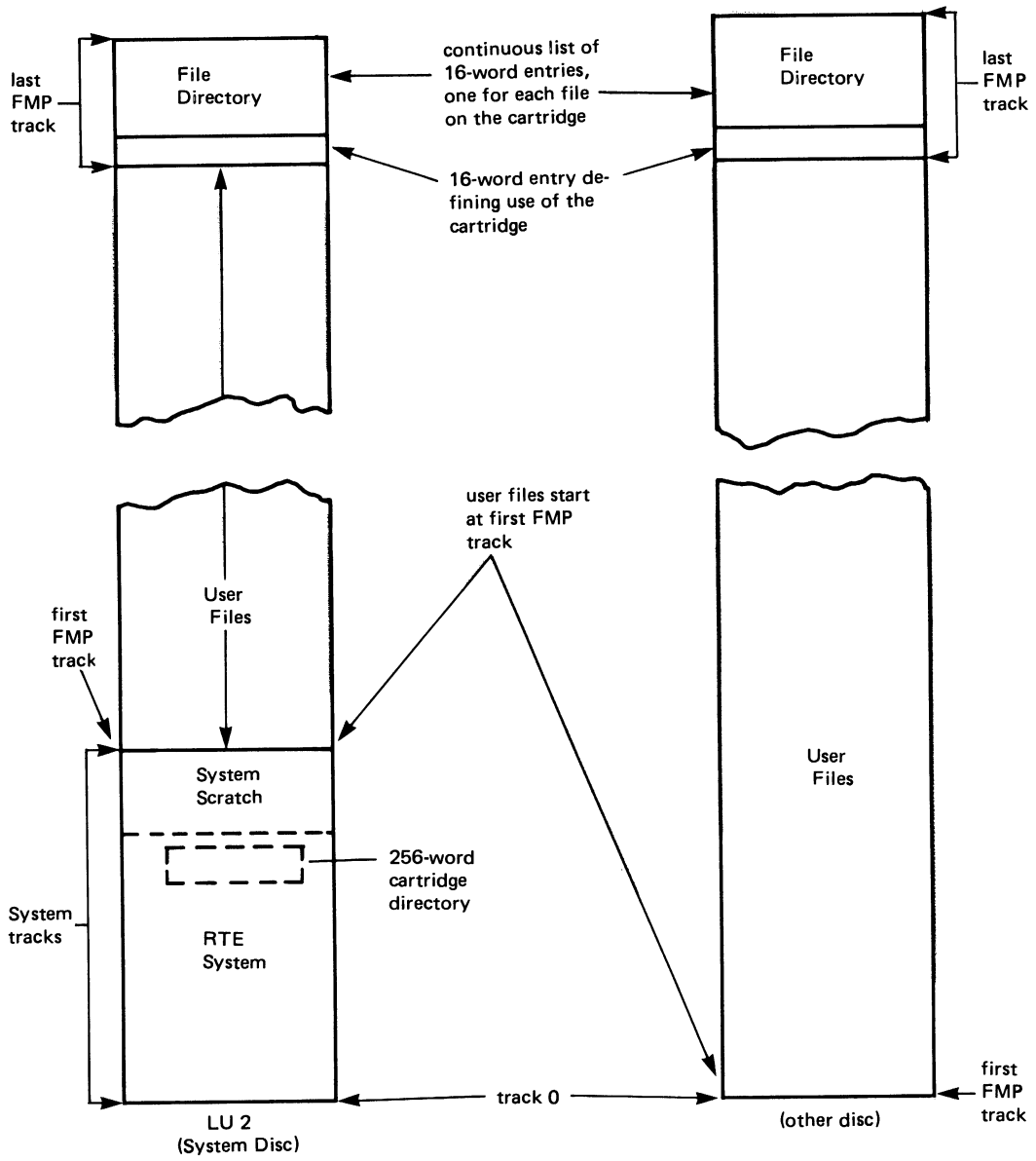


Figure 3-1. Disc Cartridge Organization

File Directory

A file directory, maintained on each cartridge, contains information on each file on that particular cartridge. Each directory starts in sector 0 of the last track available to FMP. The first 16-word entry in this directory contains label and track information for the cartridge itself. Each subsequent 16-word entry has information on a user file. The last entry is followed by a zero word. When a file is purged, the first word in the directory entry for the file is set to -1. When the cartridge is packed, the directory entry for any purged file is cleared and the cartridge area where the file was located is overwritten by non-purged files wherever possible.

Security

Besides the security features provided through the Session Monitor, the File Management Package provides two additional levels: system security and file security.

System Security

During FMP initialization, a master security code is assigned to the file system. If the code is zero, no security is provided. If non-zero, the master code must be known in order to get directory listings that include the specific file security codes and also in order to re-initialize an FMP cartridge.

File Security

Each file has a security code. This one word code may be zero, positive, or negative. A zero code allows the file to be opened to any caller (who can access the disc cartridge containing that file) with no restrictions; in effect this code provides zero security. A positive code restricts writing on the file but not reading; that is, a user who does not know the code may open the file for read only, but may not write on the file. A negative code restricts all access to the file; this code must be specified in order to open a file protected by it. An attempt to open a file so protected without the correct security code results in an error message.

Types of Cartridges

There are four categories of cartridges in the session environment:

1. System and global cartridges.

2. Private cartridges.

3. Group cartridges.

4. Non-session cartridges.

When a disc cartridge is mounted to the system, an ID is put on the cartridge list entry which identifies the category of the cartridge and to whom the cartridge was mounted (refer to Appendix C).

System Cartridges

System cartridges are those cartridges that only the System Manager can mount or dismount. The cartridge where the system tracks reside must be defined during system generation as LU 2. An auxiliary system cartridge, defined as LU 3 at system generation, can also be defined.

LU 2 and LU 3 are generally read-only cartridges for session users. Only the System Manager and non-session users can write on these cartridges. LU 2 contains:

1. Operating System

2. System Library

3. System Scratch Tracks

4. Logical Source Area

5. Cartridge Directory

6. FMP Tracks

LU 3 can be defined if more room is required for type 6 files than what is available on LU 2; if more scratch tracks are desired; or if more tracks are desired for system transfer files, hello files, etc.

In addition to LU 2 and LU 3, system global cartridges can be defined by the System Manager which provide both read and write access to all users on the system.

Private Cartridges

These cartridges are accessible only to the user who mounts them to his session and the System Manager (who has access to all cartridges in the system).

Group Cartridges

Group cartridges are accessible only to members of a group who have them mounted to their sessions and to the System Manager.

Non-Session Cartridges

Non-session cartridges are only accessible to non-session users and the System Manager. Non-session cartridges can be accessed through the system console or by programs running in a non-session environment.

Cartridge Allocation and Access in the Session Environment

A session user may interactively request disc cartridge allocation from the system through either the File Manager MC (Mount Cartridge) command or AC (Allocate Cartridge) command. Three basic differences between these commands are:

1. MC requests that a specific cartridge, identified by an unique logical unit number, be mounted. AC requests that the first available cartridge in the spare cartridge pool meeting the user's specified size requirement be mounted. When using the AC command, it is unnecessary for the session user to know the logical unit number associated with a specific cartridge.

2. MC requires that a cartridge entry already exists in the user's Session Switch Table (SST) prior to the request, whereas AC does not. AC will make the appropriate SST entry after a free cartridge has been allocated from the spare cartridge pool.

3. MC allows the user to mount any cartridge defined at system generation subject only to availability and Session Switch Table constraints, whereas AC only allows users to mount cartridges from the spare cartridge pool.

The spare cartridge pool may be thought of as a source of temporary disc work space to the session user. When the user mounts a cartridge from that area with the AC command, the file directory on the cartridge is automatically cleared for the user. In conjunction with the disc cartridge save and restore utilities, WRITT and READT, the user may easily free up disc space for other users by properly utilizing the spare cartridge pool.

Other cartridges defined at system generation and not contained in the spare cartridge pool, can be thought of as being reserved for longer duration usage. Some of the cartridges may be considered as dedicated cartridges such as system cartridges or cartridges assigned to specific users or groups by the System Manager.

The session monitor provides the session user with protection for his mounted cartridges by restricting cartridge access to only those cartridges mounted in the user's Session Control Block (SCB). Whenever a cartridge is specified in a call, FMP checks that the cartridge is mounted not only to the system, but is also mounted to the user's SCB.

There are some exceptions to the above check:

1. The System Manager is allowed access to all cartridges mounted in the system.

2. Some internal subsystems (for example, spooling) need to have and are given access to all cartridges mounted to the system.

3. With a few exceptions, read only access will be allowed on LU 2 and LU 3, but read/write access will be allowed on all other system global cartridges when operating under session.

4. User message files residing on LU 2 or LU 3 are available for session users to open, read, and write into via the ME and SM commands described later in this Chapter.

5. Type 6 files (programs) may be created and purged on LU 2 and LU 3 via the SP and PU commands described later in this Chapter.

6. Procedure files set up by the System Manager on LU 2 or 3 may be run by any session user. The commands in this file are not subject to normal cartridge or command capability level restrictions. Through these commands the user may be given read and write access to LU 2 and 3 in addition to his own private and group cartridges.

When accessing a file, if a particular cartridge is not specified, the user's private discs are searched in the order that they are mounted in the system cartridge directory. Then the user's group cartridges are searched in the order that they appear in the system cartridge directory. Finally, system cartridges are searched in the order that they appear in the directory.

Cartridge Allocation and Access without Session Monitor

Without session monitor, no cartridge protection is provided to prevent users from accessing all cartridges mounted in the system. In addition, all users will have read/write access to all system cartridges.

Logical Unit Numbers

Logical unit (LU) numbers are integers assigned to each input/output device or disc cartridge by which the cartridge or device can be referenced. In the session environment, devices have two different types of LU numbers associated with them: system LUs and session LUs. The session switch table maps the session LU which the user addresses, to the system LU, where the system processes the call. In the non-session environment, the user addresses the system LU. In either environment the user can only address LUs from 1 to 63.

System Logical Unit Number

Each I/O device and disc cartridge is assigned an unique system logical unit number at system generation. These LU numbers allow the system to identify each I/O device and disc cartridge for data transfers and other operations. They can be any integers between 1 and 254, but each must be unique.

Session Logical Unit Number

Session logical unit (LU) numbers need be unique only within a user's session. In conjunction with the session switch table, they allow each session user to reference his resources by LUs different than the system LUs defined at system generation (except for cartridges). Thus, each user can reference his session terminal as LU 1 regardless of what the system LU for that terminal is.

Session logical unit numbers can be any integers between 1 and 63. They can be assigned:

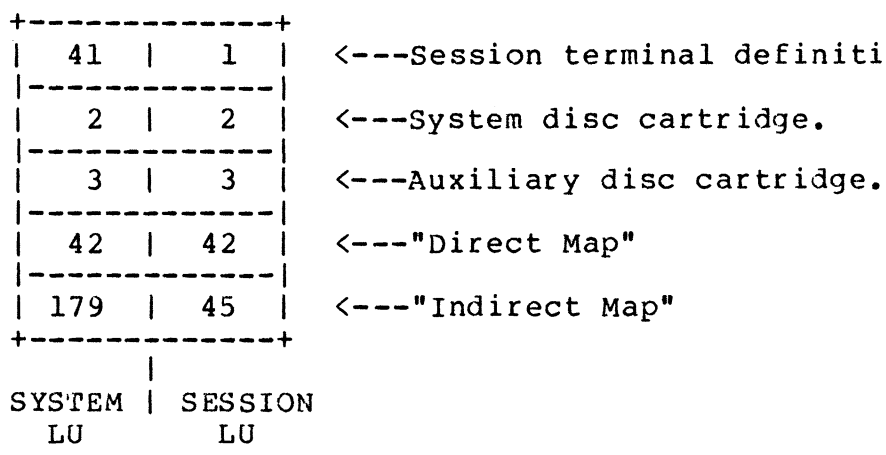
1. At Account File setup
2. At log on (via configuration table entries)
3. By the operator (via the SL command)

Session LUs can only differ from system LUs for non-disc devices. They must be identical for disc cartridges.

Session Switch Table

The session switch table (SST) provides a mapping from the session LU, which the user addresses, to the system LU, where the system processes the call. Through this mapping scheme, the SST allows session users access to system LUs greater than 63. It also limits the session user from accessing any devices or disc cartridges which are not listed in the user's SST.

The table, placed in memory by the LOGON program, consists of system LUs and session LUs. For every session LU, there is a system LU which defines the actual device which may be referenced. This may be a direct map (session LU 25 and system LU 25), or an indirect map (session LU 1 and system LU 41). An example of a user SST might be:



In the above example, the user has access to system LU 41, 2, 3, 42, and 179. The user would address requests to those LUs by specifying session LU 1, 2, 3, 42 and 45 respectively. Note that the session LUs could refer to different system LUs in another user's SST.

Global Parameters

Global parameters are variables that may be set, examined, and manipulated by FMGR commands. Global parameters may replace any FMGR command parameter. When used in procedure files, they are similar to formal parameters in a procedure or subroutine to which actual values are passed through the TR command. There are three ways to access the global parameters: as G globals, as P globals, or S globals. Each global is identified by a variable name that combines an integer with the letter G, P, or S (refer to Table 3-4 for the relationship between G, S, and P globals).

G Globals

The 11 G-type globals are named 0G through 10G. Globals 1G through 9G can be set or altered with the TR, CA, or SE commands and may be assigned any values: they may be null (no value assigned), have a numeric value, or contain up to six ASCII characters.

Globals 0G and 10G have particular values: 0G is set to the logical unit number of the input device with which FMGR is scheduled or, for batch job processing, to the job input logical unit. 10G is set by a PRTN call in a program executed by the FMGR RU command. Notably, it is set to the name of the loaded program following the command :RU,LOADR (10G is always ASCII).

P Globals

There are nine P globals, 1P through 9P. The first five P globals represent integer values returned by a program executed with the RU command. Global 6P is the current error code and 7P is the current severity code. Global 8P is the session identifier, or the terminal logical unit number if not in session. Global 9p is the session user's command capability level, or 0 if not in session. The contents of 8P and 9P can be displayed with the DP command and tested with the IF command, but cannot be modified with the CA command. The first three P globals correspond to global 10G; that is, 1P is the integer equivalent of the first two characters in 10G, 2P of the second two characters, and 3P of the last two characters. Whenever it is necessary to reference any of these three values separately, the P globals can be used instead of 10G.

The RU command results in the first five P-type globals set only when the program passes back parameter values in a call to PRTN. These parameter values may be used as parameters in subsequent commands. To illustrate:

```
:RU,PROG1  <----------------PROG1 must call PRTN to set up 1P
                            through 5P and return values to
                            FMGR.
:RU,PROG2,1P,2P,3P,4P,5P <--PROG2 can retrieve values from 1P
                            through 5P (10G, 4P, and 5P).
```

The P-type globals are particularly useful when one-word parameters are expected. To illustrate:

```
:RU,XYZ,-27P,-26P,-25P  <---Passes global 3G to program XYZ.
:RU,ABC,1P,2P,3P        <----Passes global 10G to program ABC.
```

S Globals

S-type globals are set by the FMGR commands JO or LU when running under control of the Spool Monitor. They may be referenced by FMGR commands within spooled jobs.

Two S globals have a particular meaning: 1S is null or is the ASCII file name of the last created spool file; 0S is null or is the spool logical unit number of the last spool file set up. 0S and 1S are always null when spooling is not being used.

See the RTE-IVB Batch and Spooling Reference Manual for more details on batch job processing and using the Spooling System.

Table 3-4. Global Equivalence.

S	G	P	
0	-2	-48	Type
		-47	1
		-46	2
		-45	3
1	-1	-44	Type
		-43	1
		-42	2
		-41	3
2	0	-40	Type
		-39	1
		-38	2
		-37	3
3	1	-36	Type
		-35	1
		-34	2
		-33	3
4	2	-32	Type
		-31	1
		-30	2
		-29	3
5	3	-28	Type
		-27	1
		-26	2
		-25	3
6	4	-24	Type
		-23	1
		-22	2
		-21	3
7	5	-20	Type
		-19	1
		-18	2
		-17	3
8	6	-16	Type
		-15	1
		-14	2
		-13	3
9	7	-12	Type
		-11	1
		-10	2
		- 9	3
10	8	- 8	Type
		- 7	1
		- 6	2
		- 5	3
11	9	- 4	Type
		- 3	1
		- 2	2
		- 1	3
12	10	0	Type
		1	1
		2	2
		3	3
13	11	4	4
		5	5
		6	6 Last FMGR error
		7	7 Severity code
		8	8 Session identifier
		9	9 User's capability level

The standard values are shown within dark lines.

Global Format

All globals are kept in an array in memory. G and S globals use four-word accesses; P globals use one-word accesses. Table 3-5 illustrates the format of G and S globals.

Table 3-5. G and S Global Format

word 0	global type=0 (null)	1 (numeric)	3 (ASCII)
word 1	0	integer	characters 1,2
word 2	0	0	characters 3,4
word 3	0	0	characters 5,6

Word zero defines the global type. If the type is null, the remaining words are also 0. If the type is numeric, then word one contains the integer value and the last two words are zero. An ASCII global (type 3) can contain up to six ASCII characters. This format applies only to G and S globals. Each P global corresponds to one word of the S and G globals (refer to Table 3-4).

Since the system checks a global access by its position in the array (Table 3-4), a P global can be used to reference one word of an S or G global, and S global can be used to reference a G global, or vice versa.

File Manager Commands

Program FMGR is run from the system console or from a terminal in a multi-terminal environment. It responds to a set of more than 40 commands. Any command may be entered directly from the terminal to perform a particular function, or one or more commands may be stored in a file as a procedure. Commands can also be included in jobs to be entered from peripheral devices in order to provide batch job control. In addition, FMGR commands which are entered from a terminal are placed into a command stack (resident in the FMGR program). Commands in the command stack can be displayed, modified and executed later. (Refer to the section on Command Stacking later in this Chapter.) Program FMGR allows an operator to perform the following basic functions:

1. Control FMGR by sending messages to the terminal or list device, requesting error explanations, changing the log or list devices or error severity.

2. Create and maintain files, both disc and non-disc, including maintenance of the file directory.

3. Keep track of the disc cartridge on which files are placed, including maintenance of the cartridge directory.

4. Transfer data or programs between files, creating new files as needed.

5. Establish and transfer to procedure files, and manipulate the global parameters used in these files to receive and return data from other commands or procedures.

6. Control execution of jobs in the batch stream, including assigning a job time limit, switching logical units, and aborting the job.

7. Schedule programs for execution.

8. Interactively set up spooling to/from an I/O device (available only with Session Monitor).

Since the files controlled by FMGR include data files, program files, non-disc devices, procedure files, and batch jobs, this program can provide full control over all input to and output from FMP. It uses many of the same FMP program calls for the actual input and output as does the user.

Command Structure

Each command is specified by a mnemonic code consisting of at least two letters to indicate the operation to be performed. Depending on the command, parameters may be entered to further specify the command operation. For example, STORE can be specified, but ST is always sufficient.

The following syntax conventions are used in this manual to specify format.

UPPER-CASE BLOCK LETTERS	Literals that must be specified exactly as shown; if underlined, the letters may be omitted.
lower-case letters	Type of information to be supplied by the user; most parameters are in this form.
[,parameter]	Optional parameters are enclosed in brackets; FMGR supplies a default value if omitted.
parameter 1 parameter 2 parameter 3	One and only one of the stacked parameters may be specified.
+- -+ | parameter 1 | | parameter 2 | | parameter 3 | +- -+	All bracketed parameters are optional; if all are omitted, FMGR supplies default value, or only one may be specified.
[,param1[,param2]]	Series of optional parameters; the last parameter may be omitted with no indication; embedded parameters must be indicated by a comma wnen omitted.
...	Ellipsis indicates that the previous parameter or series of bracketed parameters can be repeated

Parameter Syntax Rules

A parsing routine checks every parameter specified in a FMGR command according to the following syntax rules:

1. The first parameter is separated from the command code by a comma (,) or a colon (:). Subsequent parameters are separated from each other by commas.

2. Subparameters are legal in the first two parameters. Also, they are legal anywhere within a privileged command (see below). Subparameters are separated from each other by a colon (:). The first two subparameters may be ASCII or numeric, the rest must be numeric.

3. Blanks on either side of a delimiter or the command code are deleted from the command entry; they are not transmitted or echoed back.

4. Parameters are first assumed to be numeric, but if the parameter fails to convert, it is treated as ASCII unless it is a number immediately followed by B, G, P, S, or is preceded by a plus (+) or minus (-) sign.

5. Numeric parameters observe the following rules:

   a. A leading plus sign (+) is ignored; a number is assumed to be positive unless preceded by a minus sign (-).

   b. A number followed by the letter B is octal.

   c. A number followed by G, P, or S is a global reference.

   d. Integers should be greater than or equal to - 32768 and less than or equal to +32767.

6. ASCII parameters are parsed to a maximum of six characters; only the first six characters are interpreted. If fewer than six, the parameter is padded with trailing blanks (octal code 40, the ASCII space). Blanks within a number are ignored. In a message command (AN, TE, SM, CT, HE, or PA), a parameter may contain more than six ASCII characters since the message parameter is not interpreted.

7. The total number of characters in any parameter must be less than: 128 - (8 times the parameter number) i.e., parameter 10 must be less than 48 characters: 128 - (8*10) = 48.

8. The maximum number of parameters in one command is 14.

9. Comments may be entered following the last parameter as long as they do not replace an omitted optional parameter. They are subject to the length, number, and subparameter restriction on all parameters, but like messages are not interpreted.

10. For privileged commands (**, AN, DP, PA, RU, SY, HE, SM, CT and TE), minimum syntax checking occurs before the command is processed. The only syntax requirements for privileged commands are:

    a. If issued from a non-interactive device, the first character must be a colon (:).

    b. Global values specified in the command string must be within the legal range (refer to section GLOBAL PARAMETERS)

    c. The "constructed" command line length (after globals are replaced and blanks on either side of the delimiters are removed) must be less than 80 characters.

NAMR PARAMETER

A special parameter, NAMR, is used to identify a file or device in a FMGR command. It uses subparameters and can appear only as the first or second parameter.

The format for NAMR when identifying a file is:

```
file name[:security[:cartridge[:file type[:file size[:record size]]]]]
                                \-----------------v----------------/
                                      required for file creation
```

The format for NAMR when identifying a device is:

```
     logical unit number
```

Unless specifically noted, each subparameter has a default value of 0.

This value is selected so that, as closely as can be predicted, it provides the most general case. This means that in many cases, all subparameters can be omitted and the file be completely specified by name alone.

NAMR Subparameters

filename — 6-character ASCII file name; restricted as follows:

* only printable characters, (A-Z),(0-9),!,",#,$,%, &,',(,),=,^,\,@,[,_,;,*,],<,>,.,/,?
* plus (+), minus (-), colon (:), or comma (,) not allowed
* first character must not be blank (space) or a number
* embedded blanks not allowed
* must be unique to FMP cartridge

logical unit — positive integer specifying logical unit number of non-disc device.

security — positive or negative integer or two printing ASCII characters (which should pass file namr test); range is from -32767 through 32767; security can be:

- zero — file is unprotected (default)
- +integer — write protected; can be read with any security or none; can be written only with correct code or negative (twos complement) of correct code.
- -integer — file is fully protected; can be referenced only with correct negative code.

cartridge — positive or negative integer or two printing ASCII characters; range is from -63 through 32767; used to identify FMP disc cartridge; it can be:

- zero — first available cartridge that satisfies the request is used (default).
- +integer — cartridge reference number (CR) by which the cartridge is identified.
- -integer — logical unit number (LU) associated with the cartridge.

file type — positive integer in range 0 through 32767; default depends on command.

(see "Types of Files" above.)

file size — decimal number of blocks in range 1 through 16383, or 128-block multiples in range 2 through 32767; a block is 128 words (two 64-word sectors); indicates space allocated to file:

- +integer — allocate specified number of blocks to file; minimum is 1.
- -integer — allocate specified number of 128 - block multiples to file, integer > 1.
- -1 — allocate remainder of available space on the cartridge (up to the maximum allowed file size) to file when creating or storing a file.

record size — decimal number of words in range 1 through 32767; applies only to type 2 files; type 1 files use 128-word records, other types use variable length records.

NAMR Examples:

10	logical unit 10
20B	logical unit 16 (octal 20)
$XYZ:AA	file name $XYZ is write protected by ASCII code AA (040501 octal or 16705 decimal)
ABS:-10:-3:2:40:64	file named ABS fully protected by security code -10, is located on LU 3, is a type 2 file 40 blocks long, each record has 64 words.

?? (Request Error Explanation)

```
+-------+
| LEVEL |
|  10   |
+-------+
```

Provides a brief explanation of a FMGR error code.

```
+-----------------------------------------------------------------+
|                                                                 |
| ??[,error#]                                                     |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
|                                                                 |
| error#                                                          |
| FMGR error code whose explanation is requested; if omitted, the |
| explanation of the last error issued is given; if two error codes |
| were given the explanation usually refers to the first number only.|
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

```
:LI,PROGA       <-------------List contents of file PROGA.
FMGR-006        <-------------FMGR error message.
:??,-006        <-------------Request for error explanation.
FMGR -06 FILE NOT FOUND. <---Explanation of error.
```

COMMENTS:

After FMGR assumes control of terminal communication, if a command cannot be understood (input error) or has caused a recognized problem, an error message is printed in the form:

```
FMGR nnn
```

where nnn is a three-digit number. (Refer to Appendix B for a list of all FMGR error codes, their meaning and corrective action.)

In some cases, when an error code is issued, it is followed automatically by additional information. This may consist of the line in which the error occurred, up to the point where the error was detected, or it may be a second FMGR error code.

Any error code explanation can be requested by entering the code number as the error# parameter. A list of all FMGR error codes and their explanations is printed at the list device if error# 99 is entered. Be sure to include the comma separator or the current error will be explained.

For additional error information, the HELP command may be used. See the file manager or break-mode HE command for a description of this command.

** (Comments)

LEVEL
10

Allows user to include lines of comments within FMGR command entry list to explain command flow. This command is especially useful in procedure files.

```
** comment line
      or,
**comment line
     or,
*,comment line
     or,
* comment line
```

comment line
This is any string of alphanumeric characters which must be separated from the first asterisk (*) by another asterisk, a comma, or a blank.

*** NOTE ***

This is a privileged command and is subject to the rules described in the PARAMETER SYNTAX RULES section.

Only legal file name characters may be used in the comment line. Refer to the NAMR PARAMETER section for a complete list of allowable characters.

EXAMPLE:

```
:**CHANGE LIST DEVICE TO LU 6, THE LINE PRINTER
:LL,6
:* TRANSFER CONTROL TO PROCEDURE FILE NAMED TEST
:TR,TEST
```

COMMENTS:

The lines denoted as comments to not affect command execution. They are most commonly used in procedure files (refer to the PROCEDURE FILES section in this chapter). In order to include a comment line interactively, use the first form given above (two asterisks followed by a blank).

AC (Allocate Cartridge)

```
+-------+
| LEVEL |
|  10   |
+-------+
```

Allocates a cartridge to the session user from the spare cartridge pool. Before the cartridge is mounted to the system and the caller's session, any files residing on the allocated cartridge will be purged. The AC command is only valid when operating under session control.

```
AC,crn [,P/G [,size [,id [,#dir tracks ]]]]
                \-------------v-------------/
                  (Useful for specifying
                 size of cartridge required
                 from spare cartridge pool)
```

crn
Cartridge reference number to be assigned to the allocated cartridge (1 to 32767 or 2 ASCII characters). Must be a unique CRN to the session, but need not be unique to the system.

P/G
Private (P) or group (G) cartridge designation (default is private cartridge).

size
Size refers to the number of tracks needed on the allocated disc cartridge. If defaulted, the first available cartridge in the spare cartridge pool will be allocated.

id
ASCII identifier assigned to the cartridge; up to 6 ASCII characters. If defaulted, the id will be DC00XX, where XX is the system logical unit number of the terminal from which the AC command is input.

dir tracks
Number of directory tracks used by the file directory on the cartridge (if omitted, 1 track is assumed).

EXAMPLES:

:AC,1000,G <--- The first available cartridge in the spare cartridge pool is cleared off, assigned 1000 as its cartridge reference number and assigned to the caller as a group cartridge.

:AC,DD,P <----- The first available cartridge in the spare cartridge pool is cleared off, assigned DD as its cartridge reference number and assigned to the caller as a private cartridge.

:AC,50,,75 <--- Search the spare cartridge pool for a free cartridge that has at least 75 tracks. The first one found is initialized by clearing it off, defining it to be 75 tracks (starting with first track as 0), and assigning 50 as the cartridge reference number. Cartridge is defaulted to being private and is mounted to the system and the caller's session.

COMMENTS:

The AC command is a convenient way to mount cartridges to one's session, because unlike the MC command, it is not necessary to know the logical unit number assignment for any disc cartridge.

The AC command will search the spare cartridge pool for a free cartridge (one that is not mounted to any other user or group) that is large enough to accommodate the "size" requirement if specified. If a cartridge large enough is found, it is given the specified cartridge reference number, cleared off (all files on the cartridge are purged), mounted to the system cartridge list, and mounted to the user's session by creating an entry in the user's session control block (SCB). The first track on the cartridge is track 0 and the first file directory track is equal to "size -1".

If "size" is not specified in the AC command, the first available cartridge in the spare cartridge pool will be cleared off and mounted. The file directory in this case will start on the last track of the cartridge (defined at system generation).

If the cartridge reference number specified is already mounted in the system cartridge list with the caller's private or group ID, but is not currently active (see the DC command for an explanation of the inactive bit), the specified cartridge will simply be activated. A new cartridge will not be allocated and cleared off.

In the session environment, there are certain restrictions that the user should be aware of before mounting a cartridge. First, each user is restricted to having mounted at one time to his session control block (SCB) only a specified number of private and group cartridges. The number is assigned by the System Manager and put into the user's account file entry. If the user tries to mount more than the specified number of cartridges, an FMGR 063 error results.

Second, there must be room in the user's session switch table (SST) to post the LU number of the cartridge being mounted. Also, there cannot be another session LU currently in the SST that has the same session LU as the cartridge being posted. These two conditions can cause FMGR 066 and FMGR 065 errors respectively.

Third, if a user has a particular cartridge reference number (CRN) on a cartridge he has mounted to his session as a private cartridge, he cannot mount a group cartridge to his session that also has that CRN assigned to it or vice versa. An attempt to do this will cause an FMGR 012 error. Also, a user cannot mount to his session a private or group cartridge with the same CRN as a system cartridge.

NOTE

If an account in one group is linked to an account in another group, all users in both groups must be careful not to use duplicate cartridge reference numbers when allocating (AC) or mounting (MC) cartridges.

AN (Send Message to List Device)

```
+-------+
| LEVEL |
|  20   |
+-------+
```

Sends a message to be printed out on the list device.

```
+-----------------------------------------------------------------+
|                                                                 |
| ANNOTATE,message                                                |
|   ------                                                        |
|-----------------------------------------------------------------|
|                                                                 |
|                                                                 |
| message                                                         |
| Message to be sent to list device; maximum length of 72 characters |
| following FMGR colon (:) prompt.                                |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

```
   :LL,6      <----------------------Change list device to line printer,
                                     LU 6. (Default list device is
                                     user's terminal)
   :AN,MESSAGE TO LINE PRINTER <---Message to be sent to list device.

    MESSAGE TO LINE PRINTER  <------Message printed on line printer.
```

COMMENTS:

AN differs from the TE command in that it is sent to the list device, not the system console. Because it is printed on the list device, it is useful within batch job command files to annotate the job.

One other command (PA) may be used to send messages. PA suspends current operation and transfers control to a specified device. Optionally, it sends a message. This command is particularly useful to request operator intervention during non-interactive operation.

AN is a privileged command. (Refer to the PARAMETER SYNTAX RULES section in this Chapter).

CA (Calculate Globals)

LEVEL
40

Individual G-type and P-type global parameters can be assigned values or nulled with the CA command. The values assigned can be the result of arithmetic or logical calculations.

```
CALCULATE,global#[,p1[,op1,p2[,op2,p3[...[op(n),p(n+1)]]]]]
  -------
```

global#
G-type or P-type global which is to be set to the result of the calculation. Integers 1 through 9 identify the globals 1G through 9G (0G and 10G cannot be modified). Globals -36P through -1P and 1P through 6P also may be set to the result of the calculation using the following entry form for global#:

```
n:P
```

where n is the P type global to be set in the range -36 through +6 (excluding 0). For example, global 6P would be specified as 6:P.

p1-p(n+1)
Values used in calculations; if omitted, global# is nulled.

op1-op(n)
Operations performed on operands; may be:

+	add two operands
-	subtract one operand from another
/	divide one operand by another
*	multiply one operand by another
O[R]	inclusive OR two operands
X[OR]	exclusive OR two operands
A[ND]	AND two operands

EXAMPLES:

```
1. :CA,6,FTN4        <----------Set global 6G to ASCII value "FTN4".
   :DP,-16P          <----------Display global 6G type.
    3                <----------Global 6G is ASCII (type 3).
   :CA,6             <----------Clear global 6G to null value.
   :DP,-16P          <----------Display global 6G type.
    0                <----------Global 6G is null (type 0).
```

```
2. :CA,2,15          <----------Set global 2G to integer value 15.
   :DP,-32P          <----------Display global 2G type.
    1                <----------Global 2G is numeric (type 1).
   :CA,7,2G          <----------Set global 7G to current value of 2G.
   :DP,-12P          <----------Display global 7G type.
    1                <----------Global 7G is numeric (type 1).
   :DP,2G,7G         <----------Display contents of globals 2G and 7G.
    15,15            <----------Contents of 2G and 7G.
   :CA,1,2G,*,14,+,1   <------Set 1G to product of 2G and 14 plus 1.
   :DP,1G            <----------Display contents of 1G.
    211              <----------Contents of 1G (15x14+1).
   :CA,7,7G,-,1      <----------Decrement 7G by 1.
   :DP,7G            <----------Display contents of 7G.
    14               <----------Contents of 7G (15-1).

3. :CA,1,7,OR,15     <----------Inclusive OR 7 and 15 (octal 17),
                                assign to 1G.
   :CA,2,7,XOR,15    <----------Exclusive OR same values, assign to 2G.
   :CA,3,7,AND,15    <----------AND these values, assign to 3G.
   :DP,1G,2G,3G      <----------Display globals 1G, 2G, and 3G.
    15,8,7           <----------Contents of globals 1G, 2G, and 3G.

4.  :CA,1:P,8P       <----------Set global 1P to value stored in
                                8P (8P is the system logical unit
                                number of the terminal).
```

COMMENTS:

Evaluation proceeds from left to right until a null operation code is detected. Any other precedence is effected by multiple CA statements.

The type of the result depends on the type of the operands. If operand types differ in any one CA statement, the highest type value is used, where type 0=null, type 1=numeric, and type 3=ASCII in ascending order from 0 to 3.

Except for divide and multiply, calculations are performed separately on each word of three-word ASCII globals. For divide and multiply, all three words of the first operand are divided or multiplied by word 1 of the second operand.

In its simplest form, CA is used to null an individual global parameter or to set an individual global to the value of p1.

Globals and global types are discussed in the GLOBAL PARAMETERS section. The CA command is generally used within procedure files. For a discussion of procedure files, refer to the PROCEDURE FILES section in this Chapter.

CL (List Mounted Cartridges)

```
+-------+
| LEVEL |
|  10   |
+-------+
```

Displays list of mounted cartridges.

```
+-----------------------------------------------------------------+
|                                                                 |
| CL      <-------------Displays list of all cartridges accessible |
|                       by user.                                  |
|                                                                 |
| CLALL   <-------------Displays list of all cartridges in system |
|    -                  cartridge list.                           |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
|                                                                 |
| No parameters required.                                         |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

```
:CL
  LU   LAST TRACK    CR    LOCK  P/G/S

  35      00202    00033          P
  31      00202    01000          G
  02      00202    00002          S
  10      00202    00050          S


:CLAL
  LU   LAST TRACK    CR    LOCK  P/G/S  USER/GROUP

  02      00202    00002          S    MANAGER.SYS
  10      00202    00050          S    MANAGER.SYS
  32      00202    01000          G    ACCTG
  39      00202    02000
  31      00202    01000          G    MANUF
  35      00202    00033          P    BROWN.MANUF
  33      00202    01000          G    QA
  38      00202    00016          P    SMITH.QA
                                       JOHNSON.QA
  40      00101    05000          P    WILSON.MANUF
```

COMMENTS:

The cartridge list is issued to the list device (default is the user's terminal). The list contains the following categories:

LU	<---Logical unit number of the cartridge.
LAST TRACK	<---Last track assigned to the FMP on that cartridge.
CR	<---Cartridge reference number.
LOCK	<---Name of program locking the cartridge; blank if not locked.
P/G/S	<---Indicates whether the cartridge is mounted as private, group or system. If blank, indicates non-session cartridge.
USER/GROUP	<---Name of the user or group to whom the cartridge is mounted. If blank, indicates non-session cartridge.

The CL command lists only those cartridges mounted to the user, the user's group or the system. With the CLAL command, all cartridges mounted to all users of the system are included in the list including non-session users. Note that LU 39 in the previous example is a non-session cartridge.

For non-session users, only non-session and system cartridges are included in the list with the CL command. The CLAL command still lists all cartridges mounted to the system, however. If the session monitor software modules are not included in the system, the CL and CLAL commands will produce the same lists.

Some cartridges may be shared between session users by linking the users' accounts together. Note that LU 38 in the previous example is shared (and except for the System Manager is only accessible) by the two users SMITH.QA and JOHNSON.QA. When either of these two users log on, LU 38 will be mounted to his session as a private cartridge. For more details on account linking refer to the RTE-IVB System Manager's Manual.

CN (Control Non-Disc Device)

LEVEL
20

Controls non-disc devices such as mag tape, minicartridges and type 0 files.

```
CN [,namr [,function [,subfunction ]]]
```

namr
Logical unit number of the device to be controlled or its type 0 file name previously defined in a CR command; default is 8 (recommended logical unit for magnetic tape). Range of logical unit numbers is from 1 to 63.

function
Control function to be performed on non-disc device. Can be either two character mnemonic or octal function code. The mnemonic codes are:

RW	<--------	Rewind (default for mag tape, terminal cartridge tape unit, and mass storage devices).
EO	<--------	End-of-file.
TO	<--------	Top-of-form (default for line printer and terminal CRT).
FF	<--------	Forward Space File
BF	<--------	Backspace File.
FR	<--------	Forward Space Record.
BR	<--------	Backspace Record.
LE	<--------	Leader (default for paper tape punch).

subfunction
Carriage control characters for line printer or terminal; use if "function" is "TO" (top-of-form); it may be:

0	<--------	To suppress spacing on next print operation only.
+n	<--------	To space n lines before next print operation.
-n	<--------	To page eject on line printer or space n lines on terminal.

EXAMPLES:

1. To rewind magnetic tape:

```
:CN       <------------Defaults to LU 8 (mag tape) which has
 or                    default function of RW (rewind).
:CN,8
 or
:CN,8,RW
```

2. To eject to top of new page on the line printer:

```
:CN,6     <------------Default function for line printer (LU 6) is
 or                    TO (top-of-form).  For line printer, TO
:CN,6,TO               causes page eject.
```

3. To space two spaces on your terminal and return the carriage:

```
:CN,1     <------------Default function for terminal is
 or                    TO (top-of-form).  For terminal, TO causes
:CN,1,TO               spacing over of two spaces and carriage
                       return.
```

4. Skip 5 spaces on the line printer (no page eject):

```
:CN,6,TO,5
```

5. To space forward one record on magnetic tape:

```
:CN,8,FR
```

6. To backspace one file on type 0 file MT assigned to LU 8 at creation.

```
:CN,MT,BF
```

7. To write end-of-file mark on magnetic tape:

```
:CN,,EO
```

COMMENTS:

The CN command is similar to the FCONT subroutine call (see RTE-IVB Programmer's Reference Manual for details on this command). The function codes used in the CN command correspond to the FCONT function codes.

The function default values are determined from the driver type of the device. For the octal function codes and default values refer to the appropriate driver manual for the device to be controlled.

CO (Copy One Cartridge to Another)

```
+-------+
| LEVEL |
|  20   |
+-------+
```

Copies all files on a currently mounted cartridge to another currently mounted cartridge. When operating in a session environment, the caller must have both cartridges mounted to his session.

```
+-----------------------------------------------------------------+
|                                                                 |
| COPY,cartridge1,cartridge2                                      |
|   --                                                            |
|-----------------------------------------------------------------|
|                                                                 |
|                                                                 |
| cartridge1                                                      |
| Cartridge reference number of mounted cartridge containing files to|
| be copied; if negative, identifies cartridge logical unit number.  |
|                                                                 |
| cartridge2                                                      |
| Cartridge reference number of mounted cartridge to which files are |
| to be transferred; if negative, identifies logical unit number.    |
|                                                                 |
|                                                                 |
|                          *** NOTE ***                           |
|                                                                 |
| Files are transferred record by record; records longer than 128    |
| words are truncated.                                            |
+-----------------------------------------------------------------+
```

EXAMPLE:

Assume files A, B, C, and D on cartridge LA are to be copied to cartridge LB, and a file C already exists on cartridge LB:

```
   :CO,LA,LB     <-------Copy files contained on cartridge LA to
                         cartridge LB.
   A             <-------System prints file names as they are copied.
   B             <-------/
   C
   FMGR -002     <-------Error message indicating C is a duplicate name;
                         it is not copied.
   D             <-------System indicates that file D is being copied.
```

COMMENTS:

As each file is copied, its name is displayed on the log device. If a file on cartridge2 has the same name as a file being transferred from cartridge1, the file is not transferred and an informative message is sent to the log device.

The files being copied are not affected by the copy, except by the 128-word record length restriction,. For type 2 files containing records longer than 128 words, a warning message is displayed indicating that records are truncated to 128 words. If files already exist on the cartridge to which the files are being copied, they are not affected. Entries for the copied files are added to the file directory on cardridge 2. If there were any entries for purged files in this directory, entries for the copied files may be interspersed with entries for existing files. To know where the new file entries are placed, request a directory list with the DL command. The CO command cannot be used to copy type 0 files.

If it is desired to abort the copy before it is completed, the user may enter break-mode and specify the OF,FMGXX command. If operating under session the break-mode RS command may also be used. The break-mode BR command will not abort the copy. The BR command will only cause the copying of the current file to be aborted - the copy function will continue on the remaining files. Note that break-mode can only be entered from terminals which are controlled by the terminal handler programs PRMPT and RPN. Refer to Chapter 2 for a description of how to enter break-mode and Chapter 4 for a description of the break-mode commands.

CR (Create a Disc File)

```
+-------+
| LEVEL |
|  20   |
+-------+
```

Creates a disc file of specified type and size (no data is transferred to the file).

```
+-----------------------------------------------------------------+
|                                                                 |
| CREATE,namr                                                     |
|   ----                                                          |
|-----------------------------------------------------------------|
|                                                                 |
|                                                                 |
| namr                                                            |
| File descriptor; must not be a logical unit number; omitted     |
| subparameters default to zero; file type and file size must be  |
| specified as greater than zero;record size need be specified only |
| for type 2 files; (refer to the NAMR PARAMETER section).        |
+-----------------------------------------------------------------+
```

EXAMPLES:

1. Create a disc file named MYFILE which is a type 4 file, has a security code of -25 (read and write are restricted to users knowing the code), is allocated to a cartridge with cartridge reference number 100, and uses 10 blocks (20 sectors) of disc space.

   ```
   :CR,MYFILE:-25:100:4:10
   ```

2. Create a disc file named URFILE which is a type 2 file, uses 20 blocks, and has 72 words per record. Security code and cartridge are defaulted.

   ```
   :CR,URFILE:::2:20:72
   ```

3. Create a disc file named MYFILE which is a type 3 file with security code EJ (only write restricted) on cartridge 100 and allocate the remaining unused portion of the cartridge up to the maximum allowed file size to the file.

   ```
   :CR,MYFILE:EJ:100:3:-1
   ```

COMMENTS:

When a disc file is created, an entry is made in the file directory on the cartridge to which the file is allocated. If the subparameter "cartridge" is specified, the file is allocated to that cartridge; otherwise, user private cartridges are searched until one is found with enough room to accommodate the file, if no private cartridge is available or none can accommodate the file, group cartridges then system global cartridges are searched. If the user is not under session control, the file is sent to the first non-session or system cartridge found with enough room starting at the head of the cartridge directory on the system disc. If a file with the given name already exists on the first cartridge with enough space, a FMGR error -002 is issued.

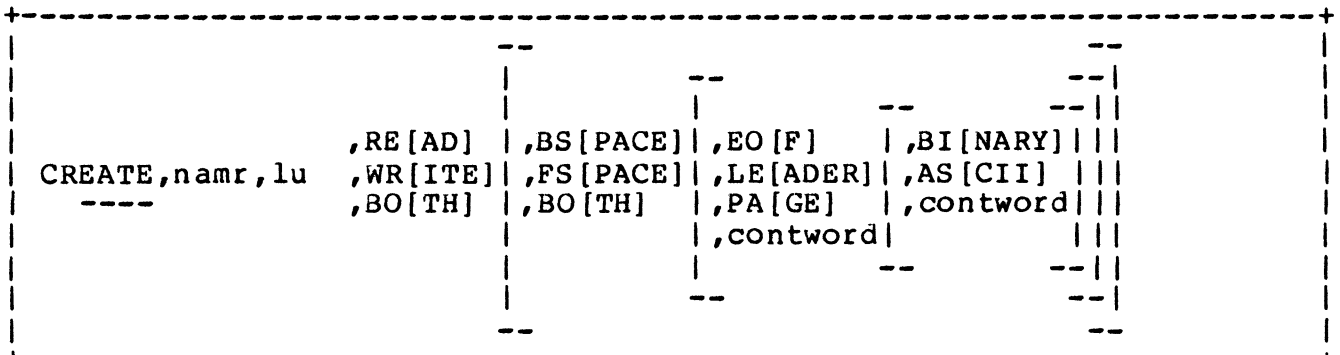
If a file is type 3 or greater, an end-of-file mark is written at the beginning of the file. As data is entered serially in the file, the mark is moved to the end of the data.

The format of a file directory entry for a created file is illustrated in Appendix C. The information in the entire file directory (all files on the cartridge) can be listed with the DL command, information in an individual file with the LI command.

CR (Create a Non-Disc File)

```
+-------+
| LEVEL |
|  20   |
+-------+
```

Creates non-disc (type 0) files by creating a file directory entry that specifies device control information.



namr
Only the file name and, optionally, the security code and the cartridge reference are specified (refer to the NAMR PARAMETER section for details); file type is default value 0 and other subparameters do not apply.

lu
Logical unit of the non-disc device; a positive integer.

READ, WRITE or BOTH specify the legal input/output mode of the device; it must be specified, there is no default.

- RE - device accepts input only; forward spacing is assumed.
- WR - device is output only; no forward spacing is supported.
- BO - device is used for input or output; backspacing and forward spacing are legal.

BSPACE, FSPACE, or BOTH specify the type of spacing the device supports; if omitted, FSPACE is assumed for READ devices and no spacing for other devices.

- BS - backspacing is supported.
- FS - forward spacing is supported.
- BO - both forward and backspacing are supported.

CR (non-disc file) . . . cont'd

EOF, LEADER, PAGE, or contword (control word) specify the particular type of end-of-file to be written on the device; if omitted, default depends on driver type.

EO - end of file mark for magnetic tape (default if device has driver type greater than 16 octal, magnetic tape, or mass storage device).
LE - leader on paper tape (default if driver type 02, paper tape punch).
PA - page eject for line printer or two line feeds on teleprinter (default if not a punch or if driver type less than 17 octal).

contword (control word) - control subfunction (equivalent to function code in FCONT, see RTE-IVB Programmer's Reference Manual); supplied if further end-of-file definition needed; specify as octal integer of which only least 5 bits are used.

BINARY, ASCII or contword (control word) specify the type of data on the device; ASCII is the default.

BI - binary data
AS - ASCII data

contword (control word) - subfunction (equivalent to bits 6-10 of IOPTN parameter in FMP OPEN call, see RTE-IVB Programmer's Reference Manual); supplied if further data definition needed; specify as decimal or octal integer of which only the lowest five bits are used.

EXAMPLES:

1. Create non-disc file named LP as output file for LU 6 (line printer); defaults are no spacing and ASCII data. The type 0 file is created on cartridge reference 20.

```
:CR,LP::20,6,WR,,PA
```

2. Create non-disc file named MT as input/output file for LU 8 (mag tape); both forward and back spacing supported; security code is 32107.

```
:CR,MT:32107,8,BO,BO
```

3. Create a read-only magnetic tape file.

```
:CR,MAG:JT,8,RE
```

4. Create non-disc file named READR as input only file on LU 5.

```
:CR,READR,5,RE
```

COMMENTS:

Programs can use non-disc (type 0) files as a means of controlling access to a device. Thus, type 0 files provide a measure of device independence in that the standard file calls can be used to control a peripheral device.

When a type 0 file is created, an entry is made in the file directory on the disc. The cartridge is locked when a type 0 file is purged.

The type 0 file entry differs from the disc file entry in that control information replaces the track, sector, and record length information. The directory entry for type 0 files is illustrated in Appendix C.

In general, a type 0 file can be specified with only the required parameters. FMGR needs a name, the logical unit, and whether the device is read only, write only, or both. The other parameters usually follow from this information.

CT (Control Terminal)

```
+-------+
| LEVEL |
|  20   |
+-------+
```

Issues a control request to an interactive terminal, optionally writing a message to the terminal.

```
CT, namr [,function [,subfunction [,message ]]]
```

namr
Type 0 file name or logical unit number of an interactive device.

function
Control function to be performed on interactive device. Default is to enable terminal (octal function code 20B).

11B	<---Space down a specified number of lines (used in conjunction with subfunction parameter).
20B	<---Enable terminal.
21B	<---Disable terminal.
22B	<---Set time-out for terminal (used in conjunction with subfunction parameter).

subfunction
Additional parameter which may be required with function. Default is no subfunction.

function	subfunction	
11B	0	<---To space down 2 lines before next print operation.
	+n	<---To space down n+1 lines before next print operation (i.e. skip n lines).
	-n	<---To space down n+1 lines before next print operation (i.e. skip n lines).
20B	Only required when enabling multipoint terminals (refer to 91730A Multipoint Terminal Interface Subsystem User's Guide; part no. 91730-90002).	
22B	Timeout value in units of 10 milliseconds.	

message
Message to be written to terminal; default is no message.

EXAMPLES:

```
:CT,20,,,TERMINAL READY   <---Enable terminal with system logical
                               unit number 20, sending "TERMINAL
                               READY" message (input from
                               non-session mode).

:CT,20,21B                <---Disable terminal with system logical
                               unit number 20 (input from
                               non-session mode).
```

COMMENTS:

The CT command is similar to the CN command. CT is a specialized command used to enable terminals. Unlike the CN command, CT allows the user to specify system logical unit numbers greater than 63 when operating out of session. It is useful for enabling terminals in the system WELCOM file (see the RTE-IVB System Manager's Manual for more details).

DC (Dismount Cartridge)

LEVEL
10

Logically removes a disc cartridge from a user's environment.

DC, cartridge	sets inactive bit in session control block entry; if operating non-session, deletes cartridge entry in system cartridge list.
DC, cartridge, RR	deletes cartridge entry in session control block and system cartridge list releasing cartridge resource back to system; in non-session environment, RR option is ignored and DC operates same as DC, cartridge.

cartridge
Cartridge identifier; positive or alphanumeric cartridge reference number assigned to cartridge or negative logical unit number associated with cartridge.

RR
Optional parameter for session users only; specification of RR deletes the cartridge's entry in the system cartridge list. If the cartridge originally came from the spare cartridge pool, it is returned.

* * * W A R N I N G * * *

If the cartridge being dismounted originally came from the spare cartridge pool, before specifying the RR option make sure that there are no files on the cartridge that need to be saved. If the RR option is specified, the cartridge will be released back to the spare cartridge pool. If a user then mounts that cartridge with the AC command or with the READT utility, all previous files on that cartridge will be lost.

If LU2 is dismounted out of session, it is taken off the cartridge list and thus not available to session users. The system manager must do a "DC,-2,RR" to remount it as a system disc.

EXAMPLE:

```
:CL
  LU   LAST TRACK      CR       LOCK    P/G/S
  35     00202        00033              P
  31     00202        01000              G
  02     00202        00002              S
  10     00202        00050              S
```

```
:DC,33 <---  set inactive bit in cartridge entry in user's session
             control block.

DISC CRN 33 LU 35 INACTIVE

:CL
  LU    LAST TRACK      CR       LOCK     P/G/S
  31       00202       01000                G
  02       00202       00002                S
  10       00202       00050                S
```

Cartridge 33 is no longer in the user's operating environment, but is still mounted to the user in the system cartridge list.

```
:CLAL
  LU    LAST TRACK      CR       LOCK     P/G/S    USER/GROUP
  02       00202       00002                S      MANAGER.SYS
  10       00202       00050                S      MANAGER.SYS
  32       00202       01000                G      ACCTG
  31       00202       01000                G      MANUF
  35       00202       00033                P      BROWN.MANUF
  33       00202       01000                G      QA
  38       00202       00016                P      SMITH.QA
  39       00202       02000
```

Cartridge 33 is still mounted to user (BROWN.MANUF), but has been set inactive by prior DC,33 command.

```
:MC,35 <--- resets inactive bit; places cartridge back into user's
            operating environment.  (Note that the MC Command
            requires that the cartridge be specified by its
            logical unit number.)

:CL
  LU    LAST TRACK      CR       LOCK     P/G/S
  35       00202       00033                P
  31       00202       01000                G
  02       00202       00002                S
  10       00202       00050                S
```

To return the cartridge back to the system, the RR option (release resource) must be specified.

```
:DC,33,RR
DISC CRN 33 LU 35 DISMOUNTED FROM SYSTEM (POOL)
```

```
:CLAL
  LU    LAST TRACK      CR       LOCK     P/G/S    USER/GROUP
  02      00202       00002                 S       MANAGER.SYS
  10      00202       00050                 S       MANAGER.SYS
  32      00202       01000                 G       ACCTG
  31      00202       01000                 G       MANUF
  33      00202       01000                 G       QA
  38      00202       00016                 P       SMITH.QA
  39      00202       02000
```

Cartridge 33 is no longer mounted to BROWN.MANUF. It has been returned to the system and is now available for any user to mount it.

COMMENTS:

When the session user does a dismount cartridge operation using the DC command, FMGR checks to see whether the cartridge specified is mounted as a private or group cartridge within the user's Session Control Block (SCB). If it is not, a FMGR 054 error (DISC NOT MOUNTED) results.

If the session user does not specify the RR option in the DC command, the cartridge will not be returned to the system. An inactive bit is set in the cartridge entry in the user's SCB to logically remove it from the user's operating environment. The cartridge would then be omitted from any file search operation which occurs when a cartridge is not specified.

If the RR option is specified, FMGR checks to see whether the cartridge specified is mounted as a private or group cartridge within the user's SCB. If it is mounted as private or group, then FMGR removes its entry from the user's SCB. FMGR then checks to see whether any other user currently logged onto the system has the cartridge mounted to his SCB. If no other user has it currently mounted, the entry for the cartridge is removed from the system cartridge directory.

When a non-session user does a dismount cartridge operation using the DC command, FMGR checks the list of non-session cartridges for the specified cartridge to be dismounted. If the cartridge specified is not a non-session cartridge, a FMGR 054 error results. If it is, the cartridge entry in the system cartridge list is removed.

LU 2 and LU 3 (if used) cannot be removed from the system cartridge directory. The command DC,2,RR will remove the entry for LU 2, but it will also put a new entry for LU 2 at the bottom of the directory. Therefore, the DC command can be used to place LU 2 or LU 3 at the bottom of the directory but not to remove it. Note that when an entry for LU 2 or LU 3 is put back into the cartridge directory, the ID of the user issuing the DC command is used in the cartridge directory entry (i.e., system manager or non-session user). This is how LU 2 or 3 can be made system type or non-session type disc.

DL (Directory List)

```
+-------+
| LEVEL |
|  10   |
+-------+
```

Provides a list of all FMP files on a specified cartridge, a list of all FMP files on all cartridges within the user's operating environment, or optionally, a list of files with common namr characteristics.

```
DL [,cartridge [,master security ]]
   or
DL,namr [,master security ]
```

cartridge
Cartridge identifier; positive for cartridge reference number, negative for logical unit number; if omitted or zero, the directories of all cartridges mounted to the user are listed.

master security
Code assigned to the system at initialization; if correctly specified, directory list includes file security code and track and sector address for each file (long list).

namr
File descriptor; must not be a logical unit number; omitted subparameters default to zero; (refer to NAMR PARAMETER section). As described below, minus signs (-) can be used as place holders in the namr to allow more flexibility.

EXAMPLES:

```
   :DL,2     <---------List all FMP files on cartridge 2.

CR=00002
 ILAB=SYSTEM NXTR= 00101 NXSEC=040 #SEC/TR=096 LAST TR=00255 #DR TR=01

NAME    TYPE    SIZE/LU     OPEN TO

+@CCT!  00001  00057 BLKS LOGON LGOFF
WELCOM  00004  00001 BLKS
"WELCO  00004  00002 BLKS
"DEV    00004  00002 BLKS
"LCHEL  00004  00003*BLKS
"HELLO  00004  00003 BLKS
```

```
:DL,A              <-----List all files whose name is A.
:DL,A-----         <-----List all files whose name starts with an A.
:DL,--A--A:-5:2    <-----List all files whose 3rd and 6th name
                         characters are A and whose security code is
                         -5 or +5 and which are on cartridge 2.
:DL,------::::1    <-----List all files of length 1.
:DL,------:::::16  <-----List all files with record length 16.
```

COMMENTS:

The DL,namr format shown above allows the user more flexibility when specifying which directory entries are to be listed. When this format is used, the following conditions must be met before a given file entry will be listed:

a. The file name must match the name portion of <namr> except that the minus sign "-" if used in <namr> "matches" any character.

b. Zero as a subparameter matches any actual subparameter, however, if a non-zero subparameter is used, it must match the file's actual parameter.

c. The standard security code match is used, i.e.,
   -n matches n and -n
   +n matches n only

The directory list is provided in two formats: a short list and a long list. Both lists have the same header information describing the cartridge itself; they differ in the file information provided. The long list includes a file security code for each file and the track and sector address for the file.

For session users, only the directories of those cartridges mounted to the user, the user's group or the system are listed. For non-session users, only the directories of non-session cartridges and system cartridges are listed.

The asterisk preceding the word BLKS in the directory listing is printed if the file size is in 128-block multiples. In the first example, note that the file size for "LCHEL is 3 128-block multiples.

DP (Display Parameters on Log Device)

```
+-------+
| LEVEL |
|  20   |
+-------+
```

Displays parameters onto the log device.

```
+-----------------------------------------------------------------+
|                                                                 |
| DP [,p1 [,p2... [,p14 ]]]                                       |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
|                                                                 |
| p1-p14                                                          |
| Parameter values or global names to be displayed; if omitted,   |
| nothing is displayed (up to 14 parameters can be displayed).    |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

1. Display non-global parameters:

```
:DP,MESSAGE TO LOG DEVICE  <---Message to be sent to log device.

  MESSAGE TO LOG DEVICE    <---Message displayed on log device.

:DP,WORD,10B,-1,-32768,50  <---Parameters to be displayed on log
                               device.
  WORD,10B,-1,-32768,50    <---Parameters displayed on log device.
```

2. Display values of globals 1G and 2G passed in a TR command:

```
:TR,TEST,PROG,1  <---Transfer to procedure file, TEST, passing
                     parameters PROG and 1 through 1G and 2G.
:DP,1G,2G        <---Display contents of globals 1G and 2G
                     (command is within procedure file, TEST).
  PROG,1         <---Parameters displayed on log device.
```

3. Assume 1G and 2G have values set in previous example and that value of 3G is null. Display global types of 1G, 2G and 3G.

```
:DP,-36P,-32P,-28P <---Display contents of P-type globals (refer
                       to Table 3-3 for global equivalence).
  3,1,0            <---Global types for 1G, 2G and 3G displayed
                       (3=ASCII, 1=numeric, 0=null).
```

COMMENTS:

This command is commonly used to display global values. It may, however, be used to display any parameter. Numeric values are printed as decimal values with no leading sign when they are positive, as decimal values with a leading minus sign when negative.

Null globals are displayed as adjacent commas (,,) unless they are P-type globals which are never null.

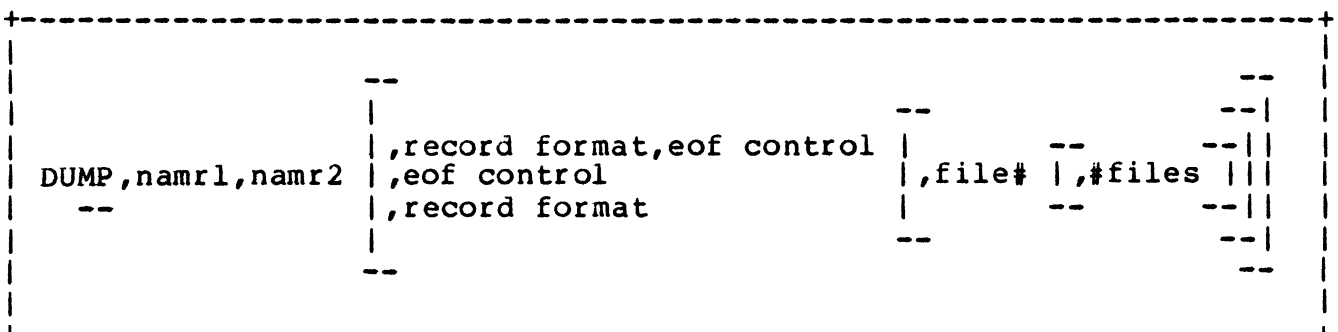
The P-type globals can be used to display the global type (refer to the GLOBAL PARAMETERS section in this Chapter).

The display is not inhibited by a severity code greater than 0.

DU (Transfer Data to Existing File)

```
+-------+
| LEVEL |
|  20   |
+-------+
```

Transfers data from an existing file or logical unit to another existing file or logical unit.



namr1
File name of existing file or non-disc logical unit number; data is transferred from namr1. (Refer to the NAMR PARAMETER section).

namr2
Name of existing file or non-disc logical unit number to which data is transferred. (Refer to the NAMR PARAMETER section).

record format
Format of data being transferred; default is derived from namr1 if namr1 is a disc file; default is ASCII if namr1 is an non-disc device.

eof control
SA to transfer end-of-file or subfile marks from namr1 to namr2; IH to inhibit end-of-file on namr2 with subfile marks not transferred. If omitted, end-of-file is written at end of data on namr 2 (refer to EOF control description below).

file#
Positive integer indicating file (or subfiles) relative to beginning of namr2 to which beginning of data is transferred; default is 1.

#files
Positive integer indicating number of non-disc files or disc subfiles to be transferred; default is 1, unless namr1 is a disc file and file# is omitted in which case default is 9999.

DU (Transfer Data to Existing File) . . . cont'd

*** NOTE ***

Only one place-holding comma is required when both record format and EOF control are omitted.

Files are transferred record by record; records longer than 128 words are truncated.

EXAMPLES:

1. Dump contents of MYFILE to left cartridge tape unit:

```
:DU,MYFILE,4
```

2. Transfer three files from magnetic tape and one file from right cartridge tape unit to disc:

```
:ST,8,A1,AS,IH,1,3   <-----Write 3 files from LU 8 as one file on
                            A1; inhibit EOF mark.
:DU,5,A1,AS,2        <-----Append 1 file from right cartridge tape
                            unit and write EOF on A1.
```

3. Transfer three files from right cartridge tape unit to disc file WXY as three subfiles and then transfer the first two subfiles to magnetic tape as two files following an existing file on the magnetic tape.

```
:ST,5,WXY,SA,1,3     <-----Transfer 3 files from right cartridge
                            tape unit to WXY as subfiles.
:DU,WXY,8,SA,2,2     <-----Transfer first two subfiles from WXY to
                            magnetic tape following existing file.
```

4. To add the contents of file B to the end of file A when both are existing disc files:

```
:DU,B,A,,2           <-----File B is transferred to follow file A.
```

COMMENTS:

If RECORD FORMAT is omitted, the file type of namrl is used to derive the format; if namrl is a non-disc device, the default record format is ASCII. The choices for record format are:

AS ASCII records are transferred.

BA Binary absolute records are transferred; checksum is performed (see Appendix C for format).

BR Binary relocatable records are transferred; checksum is performed (see Appendix C for format).

BN Binary relocatable records are transferred; without checksum (see Appendix C for format).

MT Magnetic tape ASCII records are transferred (MT record format is identical to AS record format).

MS Standard record formats are expected on namr1, magnetic tape SIO (System I/O) records are written to namr2 (see Appendix C for SIO record format).

Record formats can be combined as follows:

MSBR Magnetic tape SIO binary relocatable records.

MSBA Magnetic tape SIO binary absolute records.

If EOF CONTROL is omitted, an end-of-file mark is written on namr2 following the last data transferred; on paper tape, leader is punched at the beginning of the file as well as at the end. Any zero-length records on disc or embedded end-of-file marks on non-disc files are not transferred to namr2.

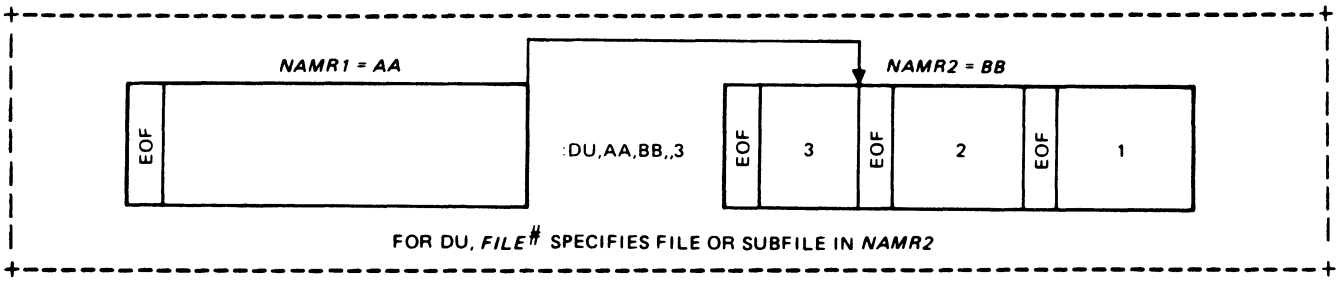
If specified, eof control is one of the following:

IH Inhibit the terminating end-of-file; useful only if namr2 is a non-disc file. On paper tape punch, inhibits initial leader.

SA Transfer embedded end-of-file marks or subfile separators from namr1 to namr2.

When SA is specified, the embedded end-of-file marks are converted to the form used by namr2.

The FILE# specifies at which file or subfile on namr2 transfer begins. This feature allows you to append one file to another or to replace one file with another. For example, specify file# as 3 to skip the first two files or subfiles on namr2 and append a third file or subfile. Transfer is always from the start of namr1.



The #FILES parameter applies to files on non-disc devices or subfiles on disc. It specifies the number of files or subfiles to transfer starting with file#. File# and #files are specified or omitted under the following circumstances:

Transfer:	*file #*	*#files*
1 particular non-disc file or the rest of a disc file	yes	no
1 non-disc file or all of a disc file	no	no
a number of non-disc files or disc subfiles from beginning of file	no	yes
a number of non-disc files or disc subfiles from particular file or subfile	yes	yes

For a description of end-of-file marks and subfile marks refer to the ST command. For file formats refer to Appendix C.

EX (Terminate Session)

```
+-------+
| LEVEL |
|   1   |
+-------+
```

Initiates log-off process when operating under control of the session monitor; terminates the program FMGR when operating in a non-session environment.

```
+-----------------------------------------------------------------------+
|                                                                       |
|       ,SP                                                             |
| EXIT ,RP [,RG [,KI ]]   <-----Session.                                |
|   --                                                                  |
|                                                                       |
| EXIT                    <-----Non-session.                            |
|   --                                                                  |
|                                                                       |
|-----------------------------------------------------------------------|
|                                                                       |
|                                                                       |
| SP                                                                    |
| Save private cartridges.                                              |
|                                                                       |
| RP                                                                    |
| Release private cartridges.                                           |
|                                                                       |
| RG                                                                    |
| Release group cartridges; default is not to dismount group            |
| cartridges.                                                           |
|                                                                       |
| KI                                                                    |
| Abort any active session programs.                                    |
|                                                                       |
+-----------------------------------------------------------------------+
```

EXAMPLES:

```
   :EX,SP        <-----Log-off saving private and group cartridges.
   :EX,RP,,KI    <-----Log-off aborting any active session programs,
                       releasing private cartridges and saving group
                       cartridges.
   :EX,RP,RG     <-----Log-off releasing private and group cartridges.
```

COMMENTS:

When logging off a session, either SP or RP must be specified if the user has a private cartridge. If neither is specified, a FMGR 071 error will result. If the user does not have a private cartridge mounted, SP or RP is not required.

Log-off will also occur if the session terminal times out five consecutive times without intervening input. A prompt is re-issued to the terminal after each time out. If a log-off occurs due to a time out, it will default to EX,SP,,KI.

Refer to Chapter 2 for a detailed description of the EX command when terminating a session.

HE (Help Function)

```
+-------+
| LEVEL |
|   1   |
+-------+
```

Provides a detailed explanation of an error and guidance in possible corrective action.

```
+-----------------------------------------------------------------+
|                                                                 |
| HELP [,keyword [,lu ]]   <---Session                            |
|   --                                                            |
|                                                                 |
| HELP,keyword [,lu ]      <---Non-session                        |
|   --                                                            |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
|                                                                 |
| keyword                                                         |
| A select group of eight character (or less) identifiers related to |
| an error code. Under session, its default is the last error posted|
| to the user's session control block (see Appendix C for description|
| of session control block); when not under session, the keyword must|
| be specified.                                                   |
|                                                                 |
| lu                                                              |
| Logical unit number of device where explanation will be output.  |
| Default is the user's terminal.                                 |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

```
:HE              <----Displays explanation of last error posted to
                      user's session control block on user's session
                      terminal.

:HE,FMGR-006,6 <----Display explanation of FMGR error code -006 on
                      logical unit number 6 (line printer).
```

COMMENTS:

All keywords and their corresponding explanations are contained in a disc resident help file. A standard list of keywords provided with RTE-IVB include the error codes of several systems and subsystems, as well as, other useful information.

One of the features of the help function is that the System Manager can modify existing entries in the help file and/or create additional keywords. This feature is described in the RTE-IVB System Manager's Manual.

IF (Conditional Skip)

LEVEL
40

Compares two values (usually globals) and skips a specified number of commands depending on the result of the comparison. This command will not be executed from an interactive device; it must be within a procedure file or a batch job.

```
IF,p1,operator,p2 [,skip ]
```

p1, p2
Values to be compared; one or both may be global parameters.

operator
Relative operator used to compare values of p1 and p2; entered as one of the following two-character operator abbreviations:

operator	operation
EQ	p1 equals p2
NE	p1 does not equal p2
LT	p1 is less than p2
GT	p1 is greater than p2
GE	p1 is greater than or equal to p2
LE	p1 is less than or equal to p2

skip
Skip count; positive or negative integer specifying the number of commands to skip when relation between p1 and p2 is true; forward skip if positive, backward if negative; if omitted, one command is skipped; if relation not true, next sequential command is executed.

EXAMPLES:

```
:IF,1G,GE,2G,2     <----If the contents of global 1G is greater
                        than or equal to the contents of global 2G,
                        then skip the next 2 commands in the
                        procedure file; otherwise, execute next
                        command.

:IF,1G,LT,5,-3     <----If the contents of global 1G is less than
                        5, then go back 2 commands in the procedure
                        file and execute that command; otherwise,
                        execute next command.
```

COMMENTS:

The specified relation between p1 and p2 is examined and if it is true then commands are skipped. One command is skipped if "skip" is not specified. If skip is specified, then that number of commands are skipped. A skip of -1 causes the IF command to be repeated. To skip back to the preceding command, specify -2.

IF will not skip past the beginning or the end of the procedure file. Such an attempt will cause a skip to the beginning or end of file mark; no error message is issued.

When a negative skip is used, the file must be on a device that recognizes a backspace. For example, it is useless to attempt a negative skip on a paper tape reader. Jobs that are spooled (refer to RTE-IVB Batch and Spooling Reference Manual) recognize backspace commands.

The following relations hold for mixed types:

null < numeric < ASCII

This corresponds to the type codes: null=0, numeric=1, ASCII=3.

If p1 and p2 are both ASCII, the comparison is based on the ASCII collating sequence (refer to Appendix A).

For a discussion on procedure files refer to the PROCEDURE FILES section in this Chapter, for a discussion on batch job command files see the RTE-IVB Batch and Spooling Reference Manual.

IN (Initialize Cartridge)

LEVEL
60

Initializes a cartridge by defining an entry for the cartridge in the file directory maintained for each cartridge. The command can also be used to change this cartridge entry or to assign a new master security code to the system.

IN,master security,cartridge,label,id [,first track [,#dir tracks [,sec/track [,bad tracks]]]]

IN,master security--new security

*** NOTE ***

The first format initializes a cartridge or changes the description of an initialized cartridge; the second format changes the master security code of the cartridge.

master security
Security code that governs access to the FMP cartridge directory and to all file security codes; must be 2 ASCII characters; if omitted, file security codes can be accessed with any code or none. If using the disc cartridge initialization, security code if ignored. See comments.

If the character "Control E" is used in the Master Security Code, the code cannot be changed until a "SWTCH" to a new system is accomplished.

cartridge
Cartridge identifier; if positive or ASCII specifies cartridge reference number, if negative, the logical unit number; must be negative the first time cartridge is initialized.

label
Cartridge reference number (CN) that identifies the cartridge; must be positive integer from 1 through 32767 or two ASCII characters.

id
ASCII identifier assigned to cartridge; up to 6 ASCII characters specified exactly like an FMP file name (refer to the NAMR PARAMETER section).

IN (Initialize Cartridge) . . . cont'd

first track
First FMP track on cartridge; a positive integer; if omitted, 0 is assumed. For the system disc (LU2), it must be at least 8 greater than the last system track; therefore, the first track of LU2 must be explicitly stated.

#dir tracks
Number of directory tracks used by file directory on cartridge; positive integer from 1 through 48; if omitted, one track is assumed.

sec/track
Number of 64-word sectors per track; if cartridge is on same channel as (that is, the same hardware select code) LU 2 or LU 3, this parameter is ignored; on any other channel, it is optional.

bad tracks
Up to six track numbers, separated by commas, specifying bad tracks on the cartridge; if omitted, all tracks are assumed to be usable.

EXAMPLES:

1. New Cartridge Initialization:

```
:IN,,-14,9600,CLASYS        <----Initialize cartridge defined at
                                 generation time as LU 14, and
                                 define cartridge reference number,
                                 9600, and cartridge identifier,
                                 CLASYS.
```

2. Re-initializing a Cartridge:

```
:IN,,9600,9700,NEWSYS       <----Re-initialize cartridge from
                                 previous example changing
                                 cartridge reference number from
                                 9600 to 9700 and cartridge
                                 identifier from CLASYS to NEWSYS.
```

3. Change the system master security code:

```
:IN,SC--RT                  <----Change master security code from
                                 SC to RT.
```

COMMENTS:

The system and system auxiliary disc cartridges (LUs 2 and 3) must be initialized the first time the RTE system is run after system generation. This process is described in the RTE-IVB System Manager's Manual.

The master security code entered for the system is the code that controls all access to FMP files and cartridges. If specified, then that code must be used in all other initializations. Once specified, it is important to remember the master security code since it is never printed or displayed by the system.

The master security code is ignored if initializing a disc cartridge in session monitor environment. A user may initialize any cartridges within his session capabilities. Thus, a user may initialize a private cartridge belonging to his session; a group manager may initialize a group cartridge belonging to his group; the system manager may initialize any cartridge. If the user does not have sufficient access privilege to the cartridge being initialized, a FMGR 046 is displayed and the IN command is aborted.

For other cartridges (other than LU 2 and 3) the MC command can also be used to initialize cartridges. One difference between the two commands is that the IN command requires that the cartridge be mounted, whereas, the MC command mounts the cartridge at the same time that it initializes it. See the MC command for more information on its format and use.

Whenever an FMP cartridge is initialized the first time, "cartridge" must be a negative number specifying the logical unit with which the cartridge is associated. "Label" is always a positive CR number.

Any cartridge, including LU 2 and 3, can be re-initialized in order to change the initialization parameters. Before attempting to re-initialize a cartridge, however, all files on the cartridge must be closed. If any files are open, the FMGR-008 error message is issued, and the IN command is not executed.

If the first FMP track is lowered on the system cartridge (LU 2), it may lower the FMP area into the RTE system area. FMGR checks and if there is no conflict, assigns the tracks to FMP. But if the tracks requested for FMP conflict with the RTE system tracks, error message FMGR 059 is issued and the highest assigned track is reported. The IN command is not exectued for this case. You may re-enter IN using the next highest track number as the first track.

Whenever the first FMP track is lowered, you must pack the disc with the PK command in order to recover the area. When the first track assigned to FMP is higher than the previous first track for the cartridge, the extra tracks are returned to RTE.

If the new parameters in a re-initialization raise the first track or lower the directory into an existing file, FMGR 060 is issued. This is a caution message that allows you to abort the initialization by entering either ?? or NO in response to the message. If you enter YES all files on the disc cartridge are purged.

Bad track information is returned during RTE generation or, if discovered by the File Management Package, is returned as a -001 error code and reported on the system console. When you know a track is bad, you must enter the track number as a parameter in the IN command. This information is kept in the cartridge directory and FMP compensates for bad tracks when it assigns tracks during file creation or packing. The first track of a new file is increased until the file contains no bad tracks. If a crested file is to use the rest of the disc, it is allocated an area above the highest numbered bad track. During packing, if a file is found to include a declared bad track, the file is purged.

Anyone knowing the master security code has access to all the file security codes on the cartridge. For this reason, it is never printed and if you know the code, you must remember it.

If the current master security code is zero (default if omitted), you must still enter some code, any code will do, in order to assign a new master security code. If the master code is other than zero, you must enter the exact code in order to change it.

The new security code can be any two ASCII characters except:

colon (:) comma (,) a leading blank.

Non-printing characters are acceptable; in fact, such characters provide greater security since they are never printed or displayed.

To remove an existing master security code, the new security code can be set to two blanks. The blanks need not be specified; they are supplied by the parameter parsing routine.

LI (List File Contents)

```
+-------+
| LEVEL |
|  10   |
+-------+
```

Lists the contents of a file, file directory information, or data stored on a logical unit on the list device.

```
LIST,namr [,format [,L1 [,L2 ]]]
  --
```

namr
File name or logical unit number; (refer to the NAMR PARAMETER section) if file is protected by a negative security code, it must be specified; if cartridge reference number is included, that cartridge is searched for the file name, otherwise, the first found with that name is listed.

format
Specifies list format:

S	ASCII source format
B	binary format
D	directory information only

If omitted, file type determines format: S if file is type 0, 3, or 4; B for all other types.

L1, L2
Starting and ending line numbers of file being listed in the specified format. If neither L1 (starting line) nor L2 (ending line) are given, the entire file is listed. If L1 is specified but not L2, one line is listed. If L1 is greater than L2, no lines are listed. If L1 is not specified, but L2 is, L1 defaults to line 1 of the file.

EXAMPLES:

```
:ST,1,AA                <----Create file AA from terminal.
FIRST RECORD FILE AA    <----File AA contents.
CNTL/D                  <----End input for file AA
                             (CNTL/D is input by simultaneously
                             depressing the CNTL and D keys
                             on the terminal).

:ST,1,BB                <----Create file BB from terminal.
FIRST RECORD FILE BB    <----File BB contents.
CNTL/D                  <----End input for file BB.

:DU,BB,AA,,2            <----Dump file BB to AA.
```

1. Source Listing:

```
   :LI,AA    <---------------S is default for type 3 or 4 files

AA     T=00003 IS ON CR00002 USING 00001 BLKS R=0000

0001  FIRST RECORD FILE AA
0002  FIRST RECORD FILE BB
```

2. Binary Listing:

```
   :LI,AA,B

AA     T=00003 IS ON CR00002 USING 00001 BLKS R=0000

REC# 00001

043111 051123 052040 051105 041517 051104 020106 044514*FIRST RECORD FIL
042440 040501                                          *E AA

REC# 00002

043111 051123 052040 051105 041517 051104 020106 044514*FIRST RECORD FIL
042440 041102                                          *E BB
```

3. Directory Listing:

```
   :LI,AA,D

AA    T=00003 IS ON CR00002 USING 00001 BLKS R=0000
```

COMMENTS:

LI lists the specified file record by record. Any binary records longer than 128 words are truncated. Source records are truncated to 72 characters. On a teleprinter, the list starts in column one, on other list devices two blanks precede the list line.

HEADINGS

If namr is a file, the listing is headed by:

```
    file name T= file type IS ON CR cartridge USING file size*
    BLKS R= record size
```

where the lower-case words are replaced by the actual values in tne file directory for the file (see examples). The asterisk following the file size is printed if the size of the file is in 128-block multiples. Note that this is the size of the main only, the extents are not shown by the LI command.

If namr is a logical unit, then a brief heading is printed with asterisks replacing the file name.

```
******T=00000 IS ON LU nn
```

where nn is the logical number.

DIRECTORY FORMAT

When D is specified one of the headings shown above is all that is listed.

SOURCE FORMAT

When S is specified, each line number (1-9999) followed by a line of text is printed. Lines may not exceed 72 characters; longer lines are truncated.

BINARY FORMAT

When B is specified, the record number is printed followed by each word of the record. Words are printed in octal followed by an ASCII equivalent if a legal ASCII character corresponds to the octal. The ASCII is separated from the octal by an asterisk. Lines are truncated after the last non-blank character (asterisks are treated as blank characters in this case). Binary format prints eight words per line, using as many lines as are needed to print the record up to the maximum of 128 words.

For zero-length records, only the record number is printed.

LL (Change List Device)

```
+-------+
| LEVEL |
|  20   |
+-------+
```

Changes current list device assignment.

```
+-----------------------------------------------------------------+
|                                                                 |
| LL, namr                                                        |
|                                                                 |
+-----------------------------------------------------------------+
|                                                                 |
| namr                                                            |
| New list device; may be either a file or a logical unit number  |
| (refer to the NAMR PARAMETER section).                          |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

1. Change list device from user's terminal to line printer and back.

```
:LL,6    <------ Change list device from default of user's terminal
                 to LU 6, the line printer.

:LL,0G   <------ Change list device back to user's terminal (global
                 0G equals LU of terminal).
```

2. Change list device from user's terminal to file named LISTF and back.

```
:LL,LISTF::1000 <--- Change list device from default of user's
                     terminal to disc file, LISTF, on cartridge 1000.

:LL,0G  <----------- Change list device back to user's terminal
                     (global 0G equals LU of terminal).
```

COMMENTS:

The namr parameter may refer to any existing device or logical unit, however, it should be a device allowing output.

Certain FMGR commands (AN, LI, CL, and DL) direct their output to the list device. By default this is the user's terminal. If a printed copy is desired, the list device can be changed to the line printer as shown in the first example above.

LO (Change Log Device)

```
+-------+
| LEVEL |
|  40   |
+-------+
```

Changes current log device assignment.

```
+-----------------------------------------------------------------+
|                                                                 |
| LOG,lu                                                          |
|   -                                                             |
|                                                                 |
+-----------------------------------------------------------------+
|                                                                 |
| lu                                                              |
| Specifies the logical unit number of the new log device; note that |
| a file name cannot be used as a log device.                     |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

If you want to send a message from your terminal (system LU 17) to another terminal (system LU 18) the following commands can be input:

```
:SL,18,18 <------ places terminal, where message is to be sent, into
                  your SST so that it may be accessed.

:LO,18 <--------- Change log device from your terminal to terminal
                  where message is to be sent.

:DP,MESSAGE <---- Display message on new log device.

:LO,0G <--------- Changes log device back to your terminal (global 0G).
```

COMMENTS:

The logical unit specified in the LO command must be a two-way device such as a teleprinter or CRT terminal since it is used both to log messages and to correct errors.

All error messages are printed or displayed on the log device. When the log device is not the input device and an error occurs that requires operator correction, control is transferred from the input to the log device and corrective action must be taken at the log device. To transfer control back to the input device, simply type a colon after the colon prompt. The second colon is interpreted as a transfer command.

If the LO command is specified within a batch job, the default device (your terminal) is re-established at the end of the job.

MC (Mount Cartridge)

LEVEL
10

Makes an unmounted cartridge available to a user.

```
MC,lu [,P/G [,size [,id [,# dir tracks [,label ]]]]]
                    \-------------v-------------/
                      (Only used if there is not a
                        valid directory on 1st
                           directory track)
```

lu
Logical unit number of cartridge to be mounted (can be positive or negative). If operating under session control, cartridge must first be in user's Session Switch Table (SST).

P/G
Private, group or non-session cartridge designation. When operating under session control, a private cartridge can be designated by typing a "P" or a group by typing "G" (default is private). When operating non-session this designation is meaningless, but its place must be provided for when specifying other optional parameters (see example).

size
The number of tracks to be used on the cartridge is specified with the size parameter. First track is always 0 and the last track is equal to size - 1.

id
ASCII identifier assigned to the cartridge; up to 6 ASCII characters. When defaulted, the id will be DC00XX, where XX is the system logical unit number of the terminal from which the MC command is input.

dir tracks
Number of directory tracks used by the file directory on the cartridge (if omitted, 1 track is assumed).

label
Cartridge reference number to be assigned to the cartridge being mounted (only applies to cartridges that do not have a valid directory at the specified track). If the cartridge already has a cartridge reference number associated with it, specification of a label will be ignored (to change the cartridge reference number of a disc LU, refer to the IN command description).

EXAMPLES:

1. Operating under session, mount cartridge with logical unit number 39 as a group cartridge.

```
:MC,-39,G
```

2. Operating non-session, mount cartridge with logical unit number 40.

```
:MC,40
```

3. Operating non-session, mount cartridge with logical unit number 41. LU 41 does not have a valid directory on track 201.

```
:MC,-41,,202,DATA,2,1250
```

(Note that even though it is unnecessary to specify the "P/G" parameter, its position must be maintained.)

COMMENTS:

The MC command will initialize the cartridge only if there is not a valid directory on the first directory track. This depends on the size parameter if it is specified. For a cartridge (LU 41) that is defined in the track map table to have 203 tracks, an MC,-41 command would require a valid directory at track 202. The command MC,-41,,100 would require a valid directory on track 99. If the cartridge has a valid directory, then the specifications of id, dir tracks, and label are ignored. Even though these parameters are ignored, they must still be valid, or a FMGR error will occur.

If the cartridge doesn't have a valid directory, the label parameter must be specified before the cartridge can be mounted. If it is not, an error is issued and MC terminates. The other parameters can be defaulted to their default values.

In the session environment, there are certain restrictions that the user should be aware of before mounting a cartridge. First, each user is restricted to having mounted at one time to his session control block (SCB) only a specified number of private and group cartridges. The number is assigned by the system manager and put into the user's account file entry. If the user tries to mount more than the specified number of cartridges, an FMGR 063 error results.

Second, there must be room in the user's session switch table (SST) to post the LU number of the cartridge being mounted. Also, there cannot be another session LU currently in the SST that has the same LU as the cartridge being posted. These two conditions can cause FMGR 066 and FMGR 065 errors respectively.

Third, if a user has a particular cartridge reference number (CRN) on a cartridge he has mounted to his session as a private cartridge, he cannot mount a group cartridge to his session that also has that CRN assigned to it or vice versa. An attempt to do this will cause an FMGR 012 error. Also, a user cannot mount to his session a private or group cartridge with the same CRN as a system cartridge.

NOTE

If an account in one group is linked to an account in another group, all users in both groups must be careful not to use duplicate cartridge reference numbers when allocating (AC) or mounting (MC) cartridges.

ME (Display Messages)

```
+-------+
| LEVEL |
|  10   |
+-------+
```

Displays contents of user's message file.

```
+-----------------------------------------------------------------+
|                                                                 |
| ME [,namr [,clear ]]                                            |
|                                                                 |
+-----------------------------------------------------------------+
|                                                                 |
|                                                                 |
| namr                                                            |
| File name or logical unit number of non-disc device where messages |
| are to be sent  (default is user's  session terminal). (Refer to |
| NAMR PARAMETER section).                                        |
|                                                                 |
| clear                                                           |
| Optional parameter if set equal to 1 clears message file after  |
| listing it; if set to 0 does not clear (default case).          |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

1) :ME <-----Displays contents of message file to user's terminal. File not cleared.

2) :ME,6,1 <-----Displays contents of message file on line printer (LU6) and clears file.

3) :ME,MYFILE,1 <--Save the contents of message file in MYFILE (file created by ME command), then clear message file.

COMMENTS:

When a user successfully logs on, a message of "MESSAGES WAITING" will be displayed if there are any entries in the user's message file.

If a file name is specified for the namr parameter, the file will be created if it does not currently exist and the message file contents will be stored in it. If the file does exist, the message file contents will be dumped into it. Note that if the list file is not extendable (not type 3 or above), the complete message may not fit in the list file.

The ME command, like the SM command which sends messages, is valid only in the session mode.

OF (Terminate Program)

```
+-------+
| LEVEL |
| 30/60 |
+-------+
```

Immediately terminates a specified program. If the user has a capability level of 30 or higher, any program within the caller's current session can be terminated. If the user has a capability level of 60 or higher, any program in the system can be terminated. If the program is a temporary program loaded on-line, its ID segment is cleared and returned to the system.

```
+-----------------------------------------------------------------+
|                                                                 |
|  OFF, program                                                   |
|    -                                                            |
|                                                                 |
|-----------------------------------------------------------------+
|                                                                 |
|  program                                                        |
|  Name of the program to be terminated.                          |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

1. :OF,APROG <---------- remove APROG and its ID segment from system

2. :OF,MAIN
   :OF,SEG1 <-------------- remove program MAIN and its two segments
   :OF,SEG2

COMMENTS:

The OF command clears the program's ID segment, if it was a temporary loaded program, and returns any disc tracks used by it to the system. The ID segment and tracks become available for use by another program. If executing, the program is also terminated. If the program is segmented, all segments must be removed with separate OF commands; an OF naming the main program will not remove its segments automatically.

The File Manager OF command is the same as the RTE command OF,program,8 (See Chapter 4). The message PROG ABORTED generated by OF,program,8 is also generated by OF,program.

Removal of a program with OF causes an abort message to be displayed at the system console unless the program is a background disc-resident segment.

Note that permanent programs created at generation or by the loader maintain their ID segments on the disc and can be removed only with the loader (See Chapter 4).

PA (Pause and Send Message)

LEVEL
40

Suspends execution of the current job or procedure file, transfers control to the log device or some other specified device, and optionally displays a message. PA is used to send messages only during non-interactive processing.

```
PAUSE [,lu [,message ]]
  ---
```

lu
Logical unit number of device to which control transfers and where message is displayed; default is log device.

message
Message to be displayed on lu; must conform to parameter syntax rules for privileged commands (Refer to the PARAMETER SYNTAX RULES section).

EXAMPLE:

Use PA to issue a request to the session terminal to load a source program into the left cartridge tape unit. PA is part of the procedure file PROC that compiles, loads and runs the program.

```
  :TR,PROC     <---------- transfer to procedure file, PROC


        :PA,,PUT MINICARTRIDGE CONTAINING SOURCE PROGRAM INTO LEFT CTU
        :ST,4,PROG
file    :RU,FTN4,PROG,0G,%PROG
"PROC"  :RU,LOADR,,%PROG,0G
        :RU,10G
        :TR
```

When PA is executed, the following message is printed at the console:

```
  :PA,,PUT MINICARTRIDGE CONTAINING SOURCE PROGRAM INTO LEFT CTU
```

The user then loads the minicartridge and enters TR to return to PROC at the command following PA.

COMMENTS:

The PA command causes a transfer to the log device or specified logical unit where the entire commmand line is printed. The message, if any, must conform to the syntax rules for any parameter (refer to PARAMETER SYNTAX RULES section).

When job processing is suspended with PA, it may be continued by entering a TR or : command on the device to which control transferred. Control returns to the command following PA.

PK (Pack Cartridge)

```
+-------+
| LEVEL |
|  20   |
+-------+
```

Recovers the tracks assigned to purged files and closes any gaps between files.

```
+-----------------------------------------------------------------------+
|                                                                       |
| PK [,cartridge]                                                       |
|                                                                       |
|-----------------------------------------------------------------------+
|                                                                       |
| cartridge                                                             |
| Cartridge identifier; a positive or ASCII cartridge reference         |
| number or, if negative, the logical unit number of the FMP            |
| cartridge to be packed; if omitted, all private, group and global     |
| cartridges mounted to a user are packed.                              |
|                                                                       |
+-----------------------------------------------------------------------+
```

EXAMPLES:

```
:PK,1000      <-------- Pack cartridge mounted to user with
                        cartridge reference number 1000.

:PK           <-------- Pack all cartridges mounted to user
                        as a private, group or global cartridge.
```

COMMENTS:

When operating under the session monitor, only the system manager can pack LU 2 or 3. Any cartridges which the user specifies to be packed must be mounted as either a private or group cartridge to the user, or as a global cartridge (i.e. a system cartridge other than LU 2 or 3).

When PK executes, it moves files into empty spaces left from purging, if possible, and updates the file addresses in the file directory. When all files are packed, it then packs the directory removing any entries for purged files.

PK will purge all files that contain bad tracks reported by the IN command. If you do not want the file with bad tracks purged, you must save it on another cartridge using the ST command.

Since the PK command locks the cartridge(s) being packed, you must be sure that there are no open files on the cartridge(s) before requesting PK. If files are open, a FMP -008 error message is issued followed by the cartridge reference number if cartridge was specified or the logical unit if not. If the cartridge being packed contains any programs restored by RP, a FMGR 011 error is issued followed by a list of all the programs. Programs must be terminated with an OF, program,8 command before packing can take place.

*** NOTE ***

To pack a cartridge containing spool files, you must first shut down the spool system (see RTE-IVB Batch and Spooling Reference Manual).

If the system fails during execution of PK, it is possible to lose, at most, one file. In order to determine which file, if any, has been lost, a copy of the file directory listing before PK was entered is needed. This list should be the long list requested with the master security code so that is shows length, first track and sector addresses for each file. When power is returned, get another such list and compare the two lists. Look for the first file with an old disc address preceded by a file with a new address. Since the directory is updated after each file is successfully moved, this unchanged entry may indicate that the file has been lost. To illustrate, suppose the directory list before packing shows:

```
NAME   TYPE    #BLKS/LU   SCODE   TRACK   SEC   OPEN TO

 A     00003     00001             0100    000
 B     00004     00003             0100    002
                                           <--2 sector gap follows B
 C     00004     00002             0100    010
 D     00003     00003             0100    014
```

If the directory after packing shows C in sector 8 and D in sector 12, no files have been lost; PK has completed its operation. If, however, either file C or D is listed in the same sector (10 or 14 respectively), there is a good chance that it was being moved when the system failed and is now lost. If the second directory shows duplicate entries this simply means the directory was being re-written when the failure occurred and no files were lost. This can be corrected by storing the affected files in new files, purging the affected files, and changing the names of the new files back to their original names.

PU (Purge A File)

```
+-------+
| LEVEL |
|  20   |
+-------+
```

Removes a file and its extents from the system. Type 0 files can be purged only with this command; other file types can also be purged with the programmatic PURGE call (See RTE-IVB Programmer's Reference Manual for details on the PURGE call).

```
+-----------------------------------------------------------------+
|                                                                 |
| PURGE, namr                                                     |
|   ---                                                           |
|                                                                 |
|-----------------------------------------------------------------+
|                                                                 |
| namr                                                            |
| File descriptor; enter file name and, optionally, security code |
| and cartridge reference; (refer to NAMR PARAMETER section).     |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

1. :PU,AA <------- purge the first file found named AA

2. :PU,CC:55:100 <------- purge file CC protected by security code 55 on cartridge 100

3. Assume files A, B, and C are the last files in the directory:

```
:PU,A
:PU,B
:PU,C
```

When C, the last file is purged, FMGR releases its disc space and then checks the next to last file, B. Since B has also been purged, its disc space is released and A is checked. It too was purged and FMGR releases its disc space. This procedure continues until FMGR finds a file that has not been purged.

COMMENTS:

When operating under the session monitor, any programs saved on LU 2 or 3 by the SP command can only be directly purged by either the person who saved the program, the System Manager, or a non-session user.

If a file is protected by a security code it cannot be purged unless the correct code is entered. If a label is specified, that cartridge is searched for the file to be purged; otherwise, cartridges are searched and the first file found with the correct file name is purged.

When a file is purged, the first word in its file directory entry is set to -1. Tracks assigned to the file are returned to tne system only if the file was the last on the cartridge. When a type 6 program is purged, its tracks are released to the system in the same manner as any other file. All purged files except the last can be returned to the system only by packing the cartridge with tne PK command, or by creating a file or a file extent that is the same size as a purged file.

A purged file cannot be accessed and its name will not appear on the file directory list requested with the DL command. A new file with the same name can now be created on the cartridge.

A type 6 file cannot be purged if an ID segment pointing to the disc space occupied by the type 6 file exists (the file has been RP'ed).

RN (Rename a File)

LEVEL
20

Renames an existing, but closed, disc file; none of the file characteristics except the name are changed.

```
RN,namr,newname
```

namr
File name and, optionally, security code and cartridge reference number of existing file (Refer to the NAMR PARAMETER section).

newname
New file name to replace existing file name.

*** NOTE ***

Subparameters of namr may not be changed.

EXAMPLES:

1. :RN,MYFILE:-25:100,MF <----search for MYFILE on cartridge number 100 and, if the security code is correct, change its name to MF.

2. :RN,URFILE,FILEA <----search cartridges for first file named URFILE and change its name to FILEA; the file is not protected by a security code.

COMMENTS:

The new name must be unique to the FMGR cartridge to which the existing file is allocated. If the file was created with a non-zero security code, then the namr in this command must include that code. If a cartridge reference number is included in the namr, then only that FMGR cartridge is searched. If the cartridge reference number is omitted, all mounted cartridges are searched and the first file found with the corresponding namr is renamed. If operating with the session monitor, the search starts with the user's private cartridges, then group, then system. In a non-session environment, the search starts with the first non-session or system cartridge in the system cartridge directory.

RP (Restore Program)

```
+-------+
| LEVEL |
|  30   |
+-------+
```

Assigns an ID segment to a type 6 file, or releases an ID segment previously assigned.

```
+-----------------------------------------------------------------+
| 1. Restore program file "namr":  RP,namr[,,pname]               |
|                                                                 |
| 2. Restore program file "namr" using the ID segment of "program":|
|                                                                 |
|    RP,namr,program[,pname]                                      |
|                                                                 |
| 3. Release disc tracks and ID segment assigned by previous RP:  |
|                                                                 |
|    RP,,program                                                  |
|-----------------------------------------------------------------|
| namr                                                            |
| Identifies type 6 file on that was saved with SP; security code,|
| and cartridge references may be specified; (Refer to the NAMR   |
| PARAMETER section); namr must not be a logical unit number.  Namr|
| is only omitted in format 3 to release the ID segment of the named|
| program.                                                        |
|                                                                 |
| program                                                         |
| 1-5 character name of program whose ID segment is assigned to   |
| namr (format 2) or released (format 3).  Its ID segment must have|
| been created by a previous RP command.                          |
|                                                                 |
| pname                                                           |
| optional 1-5 character NAME which the program restored from file|
| "namr" is to have. If not specified, the program defaults to the|
| first 5 characters of "namr".                                   |
+-----------------------------------------------------------------+
```

EXAMPLES:

1. Restore program file APROG1 as program APROG:

```
:RP,APROG1      <--- file APROG1 is restored as program APROG;
                     its ID segment is stored in memory
```

2. Restore program file TEST01 a program TEST0 and assign it APROG's ID segment:

```
:RP,TEST01,APROG <--APROG must be inactive
```

3. Release ID segment and tracks previously assigned to TEST0:

```
:RP,,TEST0
```

4. Restore program file APROG1 as program F00:

```
:RP,APROG1,,F00
```

COMMENTS:

Format 1

A program file, "namr", is restored as a program that can be accessed by the RTE system commands. It is restored with the same time parameters, priority, and partition assignment it had when saved. The name of the restored program will be "pname", if it was specified, otherwise the first five characters of "namr" define the program name. If the file "namr" has been renamed since it was saved, then the new name is used.

If a program with the same name is already in the system, error message FMGR 023 is issued. To avoid this error, it is a good idea to delete any program saved with SP using the File Manager OF, program command or to rename the type 6 file before it is restored.

All programs restored with RP are temporary; they are not recorded in the system area of disc and will not be restored automatically when the system is restarted with the bootstrap loader or when the user logs on in a later session.

If a blank ID segment cannot be found for namr, FMGR 014 is issued. An ID segment may be freed by deleting a program with OF, program or by using the "program" option of RP (formats 2 or 3).

Format 2

The restored program can be assigned the ID segment of a previously restored program by using the "program" option. "program" must identify a currently inactive program that was restored with RP.

RP first releases the ID segment assigned to "program" and then allocates a blank ID segment to "namr". RP looks for a blank ID segment from a temporary program, but if none are available, it uses a blank ID segment from a permanent program. Such ID segments occur when a permanent program is purged by the interactive loader (See Chapter 6). If "namr" is a program segment, short ID segments are used where possible.

Format 3

The third format releases the ID segment and any assigned tracks of "program". The ID segment is returned to the system as a blank ID segment that can be used by another program.

If "program" does not exist and thus does not have an ID segment, FMGR 009 is issued but does not cause transfer to the log device. If program is currently active, FMGR 018 results; enter "OF, program" to immediately terminate the program and try RP again to release the ID segment.

The restored program file cannot be purged while an ID segment points to it and while it is still running. The cartridge where the type 6 file resides cannot be packed with the FMGR PK command until the ID segment pointing to the type 6 file is released. Failure to return the ID segment before packing results in a FMGR 011 message.

Because the program ID segment table can directly address disc LU's 2 and 3, if the program file to be restored, "namr", resides on LU2 or LU3 then an ID segment is created whose program disc address word refers directly to the program file. However, if "namr" resides on a cartridge other than LU2 or LU3, then the program's memory image must be copied from the program file into the system track pool area and an ID segment created whose program disc address word refers to the system track pool copy.

This difference can affect system performance. SP'd programs which are shared among many users should reside on LU2 or LU3 to eliminate the need to copy the program image each time the program is restored. This is especially true with the RUN command automatic renaming feature in RTE-IVB which creates a duplicate copy of the program for each user every time it is run. Thus system programs which many users access should be placed on LU2 or LU3 for increased system performance. An individual user's private programs may be saved on any cartridge.

RT(Release Tracks)

```
+-------+
| LEVEL |
|  30   |
+-------+
```

Releases disc tracks currently assigned to a dormant program back to the system.

```
+-----------------------------------------------------------------------+
|                                                                       |
| RT,program                                                            |
|                                                                       |
|-----------------------------------------------------------------------+
|                                                                       |
| program                                                               |
| Name of program whose assigned tracks are to be released; must        |
| be a dormant program.                                                 |
|                                                                       |
+-----------------------------------------------------------------------+
```

EXAMPLE:

```
:RT,EDITR      <----- all LS tracks assigned to EDITR are released.
```

COMMENTS:

If the named program is dormant, all tracks assigned to that program are released. Any released tracks become available to the system and all programs suspended and waiting for disc track allocation are rescheduled. If the named program is not dormant, the request is illegal. Error message ILLEGAL STATUS is issued.

This command can be used to release any LS tracks assigned to the EDITR program. These tracks may accumulate through use of MS commands and also may be left when EDITR terminates leaving an LS area assigned. See Appendix D for a discussion of LS tracks.

Tracks can be assigned to a program programmatically through an EXEC 4 command (See RTE-IVB Programmer's Reference Manual for details on EXEC calls). The RT command is equivalent to the EXEC 5 command except that it is interactive rather than programmatic.

RU (Run Program)

```
+-------+
| LEVEL |
|  30   |
+-------+
```

Searches for and executes a named program.

```
+-----------------------------------------------------------------+
|                                                                 |
|      ,program                                                   |
| RUN  ,namr     [,parameters]  <-----  passes command string     |
|  -                                                              |
|           or                                                    |
|                                                                 |
|      ,program                                                   |
| RUIH ,namr     [,parameters]  <-----  inhibits passing of       |
|                                      command string             |
|           or                                                    |
|                                                                 |
|      ,program:IH                                                |
| RUN  ,namr:IH     [,parameters]  <--  inhibits automatic        |
|  -                                   renaming feature of        |
|                                      RTE-IVB                    |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| program                                                         |
| Program name; 5-character name of program to be executed.       |
|                                                                 |
| namr                                                            |
| File descriptor; identifies type 6 file containing program to   |
| be executed; should not be logical unit number (see the NAMR    |
| PARAMETER section).                                             |
|                                                                 |
| parameters                                                      |
| Parameters or string to be passed to program; omitted if no     |
| parameters or string is to be passed.                           |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

1. Run program PROGA, inhibiting the automatic renaming feature of RTE-IVB and passing the values 10, 20, 30, 40, 50 to PROGA.

   ```
   :RU,PROGA:IH,10,20,30,40,50
   ```

   The above command allows the user to run the program PROGA, (not a copy of PROGA), and pass parameters to the program, which can be picked up by calling the subroutine RMPAR. If no parameters are specified, and the program executes a RMPAR call, the program will receive the scheduling terminal LU as the first parameter, and zeroes for the remaining four parameters.

2. Run program PROGB, passing a string to PROGB and not inhibiting the automatic renaming feature of RTE-IVB.

   ```
   :RU,PROGB, I AM SENDING THIS MESSAGE TO PROGB
   ```

   The above command will run a copy of PROGB, and pass the command string, which is everything to the right of the FMGR colon prompt, to a buffer in System Available Memory (SAM). The string, I AM SENDING THIS MESSAGE TO PROGB, can then be picked up by PROGB by calling the RTE library subroutine GETST (See RTE-IVB Programmer's Reference Manual for details on GETST).

3. Run program PROGB, inhibiting the passing of the command string and inhibiting the RTE-IVB automatic renaming feature.

   ```
   :RUIH,PROGB:IH, I AM SENDING THIS MESSAGE TO PROGB
   ```

   The above command allows the user to run the actual program PROGB (not a copy of PROGB) and not pass the command string so that no buffer in SAM need be created. No parameters are passed in this example.

COMMENTS:

When RU is executed, a search is made of the system ID segments for the named program. If found, the program is executed. If not found, a search is made for a type 6 file in which the named program is stored. If such a file is found, it is restored, its ID segment is built in memory, and the program is executed. It is unnecessary to specify RP; RU will insure that RP is performed. Following execution of a program restored from a type 6 file, the program's ID segment is released and any tracks used by the program also are released.

When parameters are specified for a program to be executed, the particular parameters depend on the program. For instance, the assembler (ASMB) and the FORTRAN compiler (FTN4) have a standard set of parameters that are used when these programs are executed with RU. A program may pass back up to five one-word parameters with the routine PRTN. The five program parameters are passed back to FMGR in global parameters 1P through 5P. The first three parameters are also returned to FMGR as the global parameter 10G, a three-word ASCII parameter that usually contains a file name (refer to section GLOBAL PARAMETERS).

Global parameters may be used within any of the RU command parameters. The global parameters are interpreted and correct values are passed through to the scheduled program.

When a program is scheduled using the RU command form, a section of System Available Memory is allocated for storage of a command string. A command string consists of every item following the prompt character in a scheduling command entry.

When a program scheduled by FMGR passes a command string back to FMGR via an EXEC 14, FMGR wull execute the string as the next command following termination of the program. See the RTE-IVB Programmer's Reference Manual for a description of EXEC calls.

Passing of the command string to a buffer in System Available Memory allows the passing of strings to programs via the RU command. The RU command is a privileged command and conforms to the syntax rules for privileged commands (Refer to section FILE MANAGER COMMANDS). The string passage capability is described in detail in the RTE-IVB Programmer's Reference Manual.

The following descriptions define system action when a program that was restored (implicit RP) via the RU command terminates:

1. If the running program terminates, is aborted, or terminates serially reusable (See RTE-IVB Programmer's Reference Manual for description of serially reusable):

   a. The system releases program owned tracks, e.g., EDITR, LS, system scratch and compiler scratch tracks.

   b. The system releases the program's ID segment.

   c. The system releases program owned resource numbers, logical unit numbers, and program owned memory.

2. If the running program calls EXEC to place itself into the time list and then terminates saving resources:

   a. The system does not release any resources.

   b. The program's ID may not be released by RP (the OF command can be used). An attempt to release the program's ID (RU or RP) will result in an error (FMGR 018).

   c. The program remains within the system.

3. If the running program calls EXEC to place itself into the time list and then terminates, or terminates serially reusable:

   a. The system releases program owned resource numbers, logical unit numbers, and program owned memory.

   b. The program's ID may not be released by RP (the OF command can be used). An attempt to release the program's ID (RU or RP) will result in an error (FMGR 018).

   c. The program remains in the system.

In the RTE-IVB environment, ID segments are managed so that each user can have his own copy of a program. If the user wishes to run a program with FMGXX as the father (i.e., :RU,PROGX but not :SYRU,PROGX), then in certain circumstances a copy of the program will be created belonging to the particular terminal and run for the user at the terminal. RTE-IVB will perform this action whenever the program to be run is a permanently loaded or SP'd program and is a son of FMGXX. A copy of the program will be created with the last two characters being XX and scheduled for execution from the terminal with system logical unit number XX. Temporary loaded programs are not automatically renamed. Programs that are saved then restored with the RP command and run from peripheral cartridges will not be renamed.

For example, if the EDITR is loaded on-line as a temporary load, and saved as a type 6 file, the RU,EDITR command will create a program named EDIXX, and schedule it to terminal XX. When EDIXX is finished, the ID segment will automatically be returned to the system.

The advantage of processing the ID segments in this way is that all terminals can run the same program so that each user gets his own copy of the program. Therefore, the user does not have to wait for other users to finish with a program before he can use it himself.

Even if the program to be run has been previously restored using the RP command, the above procedure will still work properly. In fact, the program will be created more quickly since there would be no disc search time before the program could be run.

The program renaming feature may be temporarily inhibited when the user runs a program. The following form of the RU command can be used:

```
:RU,PROGX:IH
```

In this case, the actual program named PROGX will be run, and not a copy. This ability is especially useful when loading permanent programs. The program named LOADR (See Chapter 6 for description of LOADR) is the only program that can load programs permanently into the system; a copy of the LOADR cannot do this. Therefore, if the user is operating from FMGXX at terminal XX, the following command can be used to load a permanent program:

```
:RU,LOADR:IH, .......
```

For a type 6 file with a positive security code, the security code and cartridge should be specified when using the IH inhibit renaming feature:

```
RU,PROGX:1:2:IH
```

If the type 6 file has a negative security code, IH cannot be used.

SE (Set Global Parameters)

```
+-------+
| LEVEL |
|  40   |
+-------+
```

Assigns values to the global parameters 1G through 9G.

```
+-----------------------------------------------------------------------+
|                                                                       |
|  SET [,p1 [,p2 [... [,p9 ]]]]                                         |
|    -                                                                  |
|                                                                       |
|-----------------------------------------------------------------------|
|                                                                       |
|  p1 through p9                                                        |
|  Values assigned to globals 1G through 9G; if all parameters are      |
|  omitted, globals are nulled; if any one parameter is omitted, the    |
|  corresponding global is unchanged.                                   |
|                                                                       |
+-----------------------------------------------------------------------+
```

EXAMPLES:

```
1. :SE,,,,,FIVE              <-----global 5G assigned ASCII FIVE
   :SE,256,NEWFIL,AA,0       <-----globals 1G through 4G are assigned
                                     values; 5G through 9G are unchanged
   :DP,1G,2G,3G,4G,5G        <-----display globals 1G through 5G
     256,NEWFIL,AA,0,FIVE

2. :SE,4G                    <-----global 1G is set to the value
                                     of 4G; in this case, 0
   :DP,1G                    <-----display global 1G
     0
```

COMMENTS:

The value in p1 is assigned to 1G, that in p2 to 2G, and so forth. By using SE alone with no parameters, all the globals 1G through 9G can be cleared (nulled). Individual globals can be nulled or set with the CA COMMAND. Any integer value between -32767 and 32767 or an ASCII value up to six characters can be assigned to p1 through p9. If, nowever, you enter a global name (0G - 10G, 1P - 5P, 0S or 1S) as a parameter, the value of the specified global is assigned to the global corresponding to the parameter position.

SL (Display Session LU Information)

```
+-------+
| LEVEL |
|  10   |
+-------+
```

Displays the corresponding system logical unit number, equipment table (EQT) entry number and subchannel number for either a specified session logical unit number or all session logical units in a user's session switch table (SST).

```
SL [,lu]

lu
Session logical unit for which linkage information is desired;
default is to list information for all session LUs in user's
session switch table.
```

EXAMPLES:

```
:SL,14  <-----  Display linkage information for session logical unit 14.

 SLU  14=LU  # 14 = E 1 S 5 D
  ^     ^          ^   ^   ^
  |     |          |   |   |
  |     |          |   |   --device status
  |     |          |   --subchannel #
  |     |          |
  |     |          --EQT #
  |     |
  |     --System LU
  |
  --Session LU

:SL  <-----  Display linkage information for all session LUs in
             user's SST.

 SLU   1=LU  #  7 = E 7
 SLU   2=LU  #  2 = E 1
 SLU   3=LU  #  3 = E 1 S 6
 SLU   4=LU  # 29 = E 7 S 1
 SLU   5=LU  # 30 = E 7 S 2
 SLU   6=LU  #  6 = E 6
 SLU   7=LU  # 10 = E10
 SLU   8=LU  #  8 = E 8
 SLU  14=LU  # 14 = E 1 S 5 D
```

COMMENTS:

The session LU information is always displayed on the user's terminal. Use of the File Manager LL Command will not switch the SL output to another device such as the line printer.

When operating in a non-session environment, the SL Command will provide EQT, Subchannel and device status information for any specified LU in the system. If the LU parameter is defaulted, information for all logical units defined in the system will be provided on the user's terminal.

SL (Set up Spool File for I/O Device)

LEVEL
30/50

Allows a user to interactively assign or be allocated a spool file for spooling to an I/O device. Users with command capability levels of 50 or greater may also automatically schedule a specified program (PROG) when the spool file is closed with the 16-word string consisting of the spool set up buffer passed to it. For more details on Spooling see the RTE-IVB Batch and Spooling Reference Manual.

SL,lu ,[namr [,attribute [,outlu [,priority [,prog]]]]]

lu
The logical unit number which the program doing I/O references. If the LU is not currently a session LU in the user's Session Switch Table (SST), this command will make an entry for it providing that "outlu" is currently a session LU. Disc LUs are not permitted when specifying this parameter.

namr
Name of existing file to be used as a spool file. If omitted, system assigns a file from the spool pool defined at spool set-up.

attribute
Defines characteristics of the spool access for the current session or job. Defaults are as follows:

	SPOOL POOL FILE	USER FILE
OUTLU	WR HO SH SP	WR HO SH SA
NO OUTLU	BO HO ST SP	RE HO ST SA

WR = WRITE ONLY
RE = READ ONLY
BO = BOTH READ AND WRITE
ST = STANDARD FILE
SH = OUTSPOOL HEADERS
SP = SPOOL POOL FILE
HO = HOLD TILL CLOSE
SA = SAVE (DON'T PURGE)

SL (Spooling) . . . cont'd

Note that in all cases no buffering and hold (until close) are defaulted. Any three of the following attributes codes can be combined in any order, without delimiters in order to override defaults:

NO NOW; queue the file for outspooling now. If not given, the file will be queued (providing that an outlu is present) when the CS,lu command is given or an SL,lu,[...] command is given or in any case at the end of the job or session.

RE READ; file is read only.

WR WRITE; file is write only. An EOF will be written at the start of the file as part of set up (except see BO).

BO BOTH; same as RE WR except that EOF is not written. Note that if both RE and WR are coded, it is the same as BO and the EOF is not written.

WN WRITE NOW; same as WR plus NO. Writes EOF at start of file, except if RE also coded (see BO).

BU BUFFERED; file is buffered.

PU PURGE; file is to be purged on completion of outspooling (if file is not to be outspooled, it is purged when it is closed). This parameter is ignored if namr is not specified.

SH WRITE SPOOL HEADERS; allows faithful reproduction when file is outspooled to an outspool LU.

ST STANDARD FILE FORMAT; specifies that headers are not to be written. This is required if the file is to be positioned and reread.

SL (Spooling) . . . cont'd

outlu
Session logical unit number which maps into the system LU to which the spool file will be outspooled. The system LU associated with outlu must have been specified as an outspool destination LU during GASP initialization. If omitted, the outspool file will not be output to a device when the spool file is closed.

priority
This is the outspool priority of the program's output. The default is 99 if the program is scheduled interactively from a session terminal.

prog
Program prog will be immediately scheduled by the spool system when the spool LU is closed. The 16-word Spool Setup buffer will be passed to prog when it is scheduled. Note that the spool file will not be outspooled, and it is prog's responsibility to properly dispose of the file.

EXAMPLES:

1. While in session, spool all output to the line-printer:

```
:SL,6,,,6
```

When the user logs-off, the spool file will be closed and the output will the spooled to the line-printer (LU 6).

2. In a session, dump file ASC to the line printer via spooling:

```
:SL,20,ASC,,6       <----Set up spool equivalence.
:CS,20              <-----Close spool LU (see RTE-IVB Batch and
                          Spooling Reference Manual for details).
```

3. Use spooling to perform formatted writes to a disc file:

```
   PROGRAM WRSPL
   WRITE (45,10)       <----Program does formatted write.
10 FORMAT ("HI THERE")
   END

   :SL,45,"TEXT,WR     <----Equivalence session LU 45 to existing
                            file "TEXT.
   :RU,WRSPL           <----Run program to write message.
```

The file "TEXT now contains the ASCII string "HI THERE" as its first record.

4. Use spooling to perform a formatted read from a disc file:

```
   PROGRAM RESPL
   DIMENSION IST(39)
   READ (45,10) IST    <----FORTRAN program to read a 78 character
                            message into array IST.
10 FORMAT(39A2)
   END

   :SL,45,"TEXT        <----Equivalence session LU 45 to existing
                            file "TEXT.
   :RU,RESPL           <----Run program.
```

Array IST in program RESPL now contains the ASCII string "HI THERE".

5. Set up a spool file in a session. When the spool LU is closed, have the spool system schedule program GETIT to process the file with subroutine PROC:

```
   PROGRAM GETIT
   INTEGER SETUP(16)
   CALL EXEC(14,1,SETUP,16)  <----Retrieve 16-word buffer.
   CALL PROC(SETUP)
   END

   :SL,20,TEXT,,6,,GETIT     <----Set up spool equivalence of
                                  existing file TEXT and session
                                  LU 20. Schedule program GETIT
                                  when the spool file is closed.
   :LL,20          <----------------Change list device to LU 20.
   :DL             <----------------Do a directory listing.
   :CS,20          <----------------Close spool LU. Program GETIT
                                    is automatically scheduled and
                                    processes the file.
```

COMMENTS:

Logical Unit Assignment

The system associates a spool logical unit with the specified logical unit. The spool logical unit associated with lu can be retrieved by displaying the global parameter OS with the FMGR DP command.

The logical unit (lu) specified should correspond to the device type of the actual device for which it is intended (outlu). This is used to set up the device type to ensure that control functions and control requests are issued to the device as expected. For example, if output is to be sent to the line printer, logical unit 6 associated with the line printer should be specified. If a logical unit specified for lu has not been associated with a particular device, magnetic tape is assumed.

The lu must not be any of the following:

* logical unit 2 (system disc), or 3 (if assigned as auxiliary disc)

* any logical unit associated with a disc driver (DVR 30, 31, 32 or 33)

* if in a job, system logical unit 5 (standard spool input device). If you want to use the true system logical unit 5, use an SL switch command (see version of SL command describing modification of a user's Session Switch Table).

* a spool logical unit

Spool File Assignment

If namr is not specified, an available file from the spool pool files (SPOL01-SPOL80) is associated with the logical unit. If a particular file is specified, then that file is associated with lu and will be used as a spool file during spooling. The specified file must be an existing user file; SL does not create the file. The name of the most recently assigned spool file can be retrieved by displaying the global parameter 1S with the FMGR DP command.

Attributes

If lu is to be used for output only or for output as well as input, then its attributes must be specified.

No buffering saves System Available Memory space. Generally, there is no need to request buffering with BU. Spool file headings are used rather than standard format (ST) if you want to read back the file either within the JOB or after the job. If the file is to be outspooled only, it is better to use the spool headers format. If ST is specified, no control is passed through to the outspool routine (SPOUT). In this case a default EOF action based on the actual device to which the file is being outspooled is supplied.

If you do not want to save the user-defined spool file namr, you must specify PU. Spool pool files are never saved. The normal case is to hold the spool file for outspooling until the spool is closed. If you specify NO, the file will be placed in the outspool queue immediately.

The SL hold attribute is not the same as the GASP hold established with the GASP CS command. The SL hold prevents the file from being placed in the outspool queue (an entry is not made for it in SPLCON) until a CS command removes the hold or the session or job terminates. The GASP hold places a hold on a file already entered into the outspool queue (already listed in SPLCON), and may only be removed with a subsequent GASP command.

Outspool Logical Unit Assignment

The device to which output from spooling is to be directed must be specified with the outlu parameter, or the file will not be outspooled. The specified logical unit is not associated with a spool file or spool logical unit. Any outspool logical unit specified at GASP initialization may be used. An octal function code that specifies an I/O function code in bits 6-10 and a logical unit in bits 0-5 can be used when a file with standard format is output; files with spool header format use the function code in the second header word. For example, the code 104B indicates punch binary on logical unit 4. The function code is equivalent to the CONWD parameter in the standard I/O EXEC calls (refer to the RTE-IVB Programmer's Reference Manual).

For spool header file format, if the file is to be outspooled, be careful that the proper attribute is used to indicate the file type SPOUT will actually see.

If outspool format is specified for a spool file, the validity test performed by SPOUT will usually find the error and change it to standard format, however, one or more records may be transmitted before the test fails.

SL (Modify Session Switch Table)

```
+-------+
| LEVEL |
| 30/50 |
+-------+
```

Defines an LU mapping which is to be entered or deleted from a user's Session Switch Table (SST). With a command capability level of 30 or greater, a user may map a new session LU to a system LU currently within the user's SST. With a command capability level of 50 or greater, a user may add a system LU to his SST with this command. A user may only delete LU mappings which have been created during his session.

```
+-----------------------------------------------------------------------+
|                                                                       |
| SL,session lu,system lu  <-----  defines LU mapping in SST            |
|                                                                       |
| SL,session lu,-          <-----  deletes LU mapping in SST            |
|                                                                       |
|-----------------------------------------------------------------------+
|                                                                       |
| session lu                                                            |
| Logical unit number by which session users address a system           |
| logical unit number. Can be the same as the system LU, but is         |
| only required to be so for disc cartridges. Session logical           |
| units must be unique within any user's session switch table.          |
|                                                                       |
| system lu                                                             |
| Logical unit number by which the system addresses a physical          |
| or logical device. System logical unit numbers are defined at         |
| system generation time.                                               |
|                                                                       |
+-----------------------------------------------------------------------+
```

EXAMPLES:

```
:SL,7,10  <-----  Adds system logical unit 10 to the user's
                  SST and assigns it session logical unit
                  number 7.

:SL,7,-   <-----  Deletes entry for session logical unit 7
                  for the user's SST.
```

COMMENTS:

The System Manager defines each user's SST configuration at account set-up. The SST configuration resides in the Account File on disc and can only be changed by running the account program.

At account set up time the System Manager also defines the number of additions (new LU mappings) a user can make to his SST with the SL command. An attempt to exceed this limit will result in an FMGR 066 error.

See Chapter 2 for a description and purpose of the Session Switch Table (SST).

SM (Send Message to Session User)

LEVEL
10

Sends a file and/or message to a specified session user's message file

```
SM,user,namr,message
```

user
Log on identification of user to whom message is to be sent.

namr
Name of existing file or logical unit number of a non-disc device (a positive integer) where data to be sent is stored.

message
Message string entered from your terminal and directed to and stored in the specific user's message file.

EXAMPLES:

1.
```
:SM,SMITH.ACCTG,INVENT::1000  <---passes inventory data to another
                                  user in another group.
```

2.
```
:SM,SMITH.ACCTG,,WHERE ARE YOUR COST ESTIMATES   <--passes literal
                                                    message to
                                                    another user.
```

3.
```
:SM,SMITH.ACCTG,8,HERE IS THE DATA THAT YOU REQUESTED
```

(Sends contents (first file) on LU 8, mag tape, to another user and precedes it by a message.)

COMMENTS:

When the user who is to receive the message logs on, he will be informed of any messages pending in his message file ("MESSAGES WAITING").

When an LU is the source of a message file (e.g.,mag tape or paper tape) the first file will be read, terminating at an EOF. When a file is entered from a terminal, EOF is acknowledged by control D. A message string is terminated by a carriage return.

Stored with the file is a header, indicating the sender of the message, and the time it was sent.

There is no limit to the size of the message, other than the available room on the cartridge containing the message file.

If both a file and a message string are sent to a user, the message string will precede the file in the user's message file.

SP (Save Program as Disc File)

LEVEL
30

Saves a disc-resident program as a type 6 FMP file.

```
          /,PR
SP,namr[     [,cap]]
          \,GR
```

namr
File name of the type 6 file which is created by the SP command (refer to the NAMR PARAMETER section); "namr" cannot be a logical unit number; first five characters of the file name must be identical to the name of the saved program. "namr" subparameters default to:

security code	default to 0
cartridge	default to first cartridge in cartridge list
file type	forced to type 6
file size	forced to size of program
record size	forced to 128

PR
If specified, only the user who issued the SP command (or users linked to this account) will be allowed to run or RP the program from the type 6 file.

GR
If specified, only users belonging to the same group as the user who issued the SP command will be allowed to run or RP the program from the type 6 file.

cap
An integer representing the minimum capability level required to run or RP the program from the type 6 file. Note that the user must be able to invoke the RU or RP command (30), regardless of the capability specified in the SP command.

EXAMPLES:

1. Save a temporary disc resident program APROG as a type 6 file named APROG1:

```
:SP,APROG1 <-----save APROG and its ID segment as a file named APROG1

:OF,APROG  <-----delete the original program fron disc
```

2. Save a permanent program PROG as a type 6 file named PROG, then purge the permanent program.

```
:SP,PROG   <-----save PROG and its ID segment as a file named PROG
```

```
:RU,LOADR:IH  <-----run interactive loader (See Chapter 6); renaming
                    feature is inhibited because only the original
                    version of LOADR can create or purge permanent
                    programs.

/LOADR: OP,PU <-----  purge option specified.

  /LOADR: PNAME ?PROG  <-----  purge PROG.
```

COMMENTS:

Any cartridge may be specified.

A program name is a maximum of five characters; a file name can be six characters. This means a type 6 file for more than one version of the same program can be created by adding one character to a five character program name. If, however, the program name is less than five characters, the file name must be identical.

The SP command is usually used in conjunction with the RP command. A program that has been edited, compiled or assembled, has been loaded and then run successfully can be saved in its loaded form for future use with the SP command. To run this program through FMGR, issue either the RU command alone or RP and RU. In either case, the file is restored and then executed.

Segmented programs may be SP'ed to any disc cartridge, but may only be run directly (restored by the RUN command) if saved on LU 2 or LU 3. When a segmented program is saved on a peripheral disc cartridge, the main and all segments must be RP'ed prior to RUnning.

Which cartridges type 6 files are placed on may affect system performance. See the section on the RP command for a discussion of this.

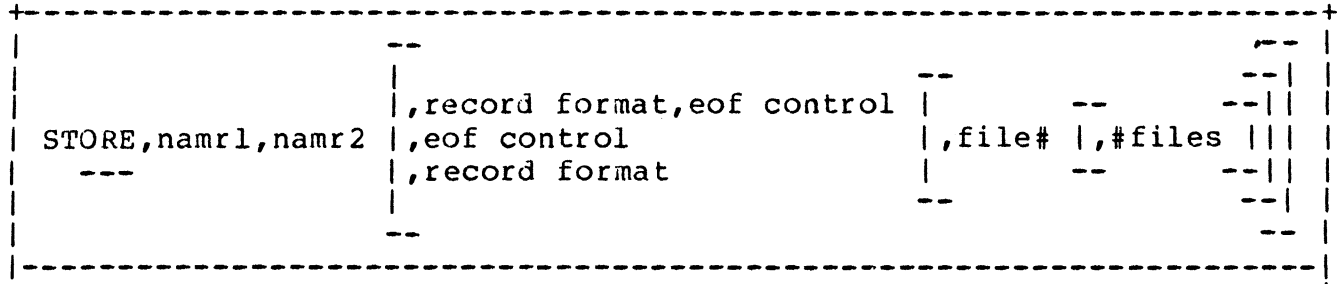
One reason to save a program with SP is to release its ID segment for use by another program. The two examples above demonstrate the procedure used to SP either a temporary or permanent program and release its ID segment by using either the FMGR OF command or by purging it with the interactive loader.

Temporary programs created during a session are automatically purged by the Session Monitor when the user logs off and the program's ID segment is released back to the System (if the program has segments appended to it, these will not be purged and their ID segments will not be released at log off). If the user wishes to save temporary programs when logging off, an SP, program command must be issued first.

ST (Transfer Data and Create File)

LEVEL
20

Transfers data from a file or logical unit to a disc file or logical unit; the receiving file is created by the command unless it is a logical unit. To transfer data to an existing file use the DU command.



namr1
File name of existing file or non-disc logical unit number; data is transferred from namr1. (Refer to the NAMR PARAMETER section).

namr2
File name or logical unit to which data is transferred from namr1; if a file name, the file is created using the last three namr subparameters if supplied.

record format
Format of data in namr1; default is derived from file type of namr1 or is ASCII; refer to record format description below.

eof control
SA to transfer embedded subfile marks in namr1 to namr2 and write end-of-file mark at end of data on namr2. IH to inhibit end-of-file on namr2; subfile marks are not transferred. If omitted, end-of-file marks are saved, subfile marks purged.

file#
Positive integer indicating file (or subfile) relative to beginning of namr1 at which to start transfer; default is 1.

#files
Positive integer indicating number of non-disc files or disc subfiles to be transferred; default is 1 unless namr1 is a disc file and file# is omitted, in which case default is 9999.

ST (Transfer Data and Create File) . . . cont'd.

*** NOTE ***

Files are transferred record by record; records longer than 128 words are truncated.

Only one place-holding comma is required when both record format and eof control are omitted.

EXAMPLES:

1. Create file AA and copy contents of disc file XYZ into it.

```
:ST,XYZ,AA
```

2. Copy a file from cartridge 2 to cartridge 17.

```
:ST,XYZ::2,XYZ::17
```

3. Create and store ASCII data in a file, FILEX, from a terminal.

```
:ST,1,FILEX

    .  <-----  enter lines of ASCII data; terminate each
    .          line with a carriage return.
    .

CNTL/D <-----  press CNTL and D keys simultaneously to
               terminate input (Control D).
```

4. Copy first five subfiles on disc file DFILE to magnetic tape as five separate files.

```
:ST,DFILE,8,MT,SA,,5
```

5. Transfer a binary relocatable file from minicartridge to disc. Data is input from cartridge tape unit LU4 and stored in a file created by the ST command called BINRE.

```
:ST,4,BINRE,BR
```

COMMENTS:

namr1 can be a created file (disc or non-disc) or a logical unit number defined for the system.

namr2 can be a logical unit number or a disc file name; it cannot name a type 0 file. The file type subparameter for namr2 defaults to the file type of namr1 if namr1 is a disc file.

If namr1 is not a disc file, the file type of namr2 is based on the record format of namr1:

type 3 record format - MT, MS, AS (ASCII, System I/O) or omitted
type 5 record format - MSBR or BR (binary relocatable)
type 7 record format - MSBA or BA (binary absolute).

Type 6 files are normally created with the SP command; type 4 and 5 files with the EDITR and FORTRAN Compiler or the Assembler. These files can be stored to new files with the ST command.

The file size subparameter for namr2 defaults to 24 blocks (4096 bytes) when namr1 is a device, otherwise file size defaults to size of namr1 without extents. The size of namr2 is truncated to the actual number of records used in namr1, when namr1 is a device or a disc file.

Any records greater than 128 words are truncated in namr2.

Record Format

If record format is omitted, the file type of namr1 is used to derive the format:

type 0 (non-disc) 3 or 4 (disc files) record format - AS (ASCII).
type 5 record format - BR (binary relocatable)
type 7 record format - BA (binary absolute)

The choices for record format are:

AS ASCII records are transferred

BA binary absolute records are transferred; checksum is performed (see Appendix C for format)

BR binary relocatable records are transferred; checksum is performed (see Appendix C for format)

BN binary relocatable records are transferred; without checksum (see Appendix C for format)

MT magnetic tape ASCII records are transferred (MT record format is identical to AS record format)

Record formats can be combined as follows:

MSBR	magnetic tape SIO binary relocatable records
MSBA	magnetic tape SIO binary absolute records.

Subfiles

Before discussing the remaining parameters, it is necessary to understand subfiles. On non-disc devices, files are separated by end-of-file marks that depend on the device:

magnetic tape	EOF mark
paper tape	leader
line printer	top-of-form indicator (page eject)
terminal	two spaces (two carriage return/line feeds) on output; CTRL/D on input.
minicartridge tape	carriage return/line feed.

If namrl is a non-disc device, it may contain many such subfiles: the physical end-of-file for the particular logical unit (device) terminates the collection of files or information. A disc file is terminated by a special end-of-file, -1, in the first length word of the last record. Disc files may be divided into subfiles by zero length records, two length words set to zero. These are logical divisions that usually result from transferring non-disc files to disc. (Refer to Appendix C for file formats.)

Subfiles are useful when you want to save more than one tape file or relocatable program file (such as a library) on a single disc file and retain the files as separate entities. Subfiles can subsequently be stored (ST command) or dumped (DU command) to another file separately or as a single file (see Figure 3-2).

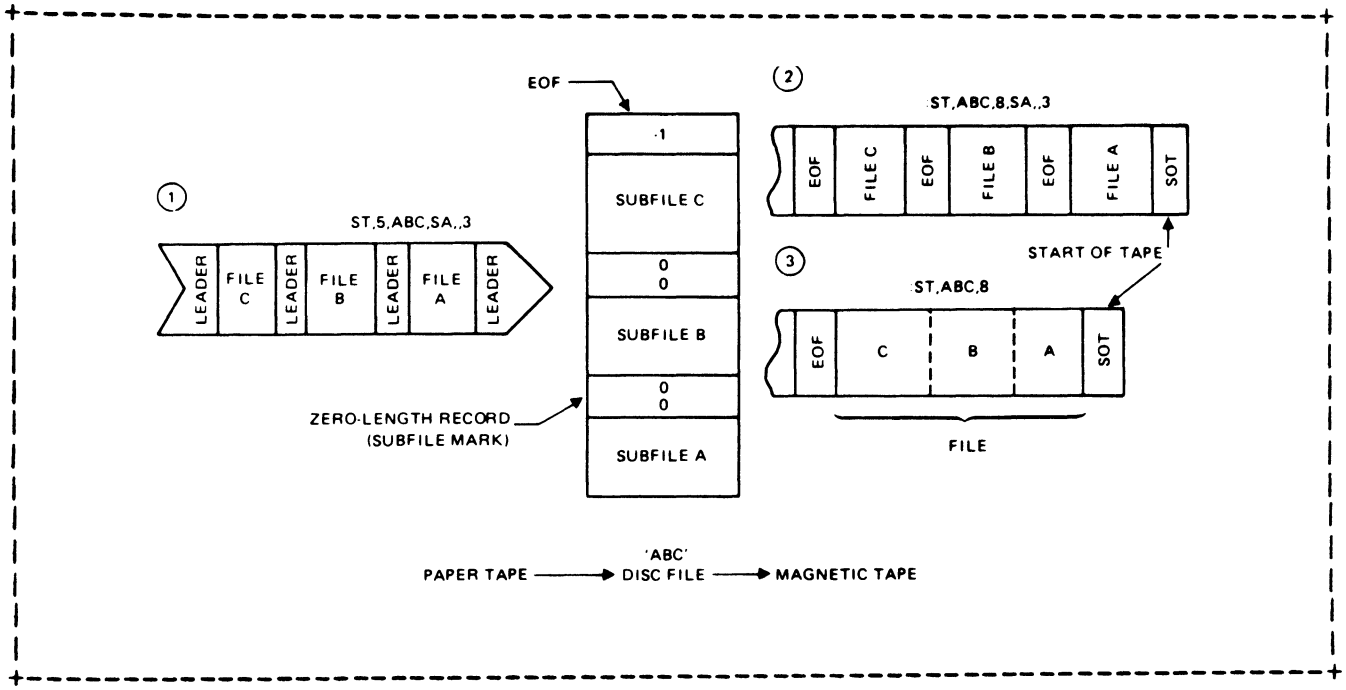


Figure 3-2. Relation of Files to Subfiles

EOF Control

If EOF control is omitted, an end-of-file mark is written on namr2 following the last data transferred; on paper tape, leader is punched at the beginning of the file as well as at the end. Any zero-length records on disc or embedded end-of-file marks on non-disc files are not transferred to namr2 (see (3) in Figure 3-2).

If specified, EOF control is one of the following:

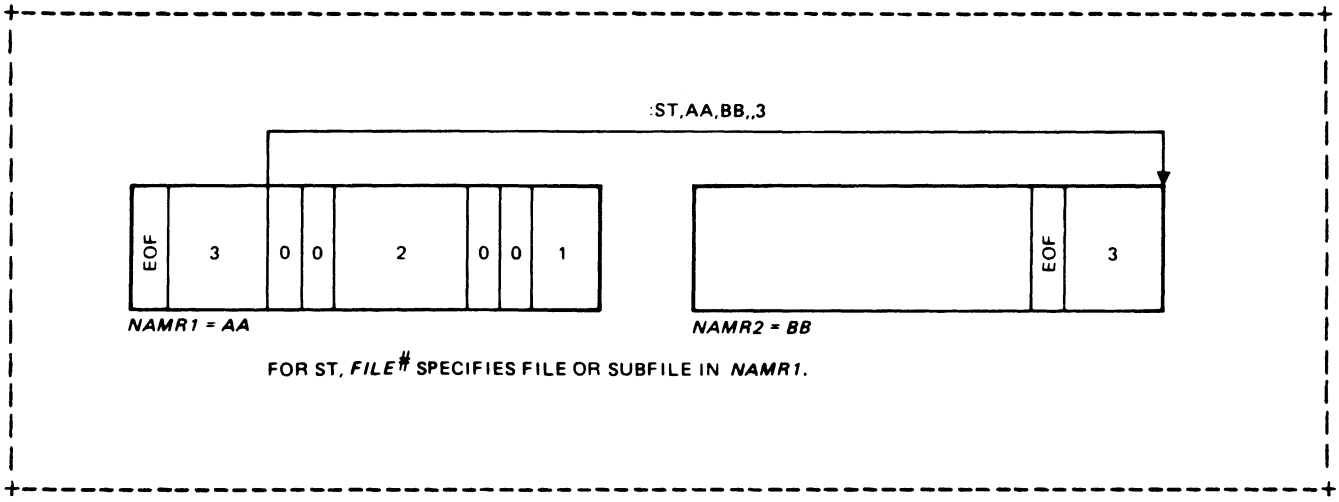
IH inhibit the terminating end-of-file; useful only if namr2 is a non-disc file. On paper tape punch, inhibits initial leader.

SA transfer embedded end-of-file marks or subfile separators from namr1 to namr2

When SA is specified, the embedded end-of-file marks are converted to the form used by namr2. For instance, if a paper tape file is being stored on disc, the leader is converted to zero-length records (see (1) in Figure 3-2) and zero-length records to EOF marks on magnetic tape (see (2) in Figure 3-2).

File Number

The file# parameter applies to files on non-disc devices or subfiles on disc. It specifies which file or subfile relative to the first with which to start the transfer. For instance, if file# = 3, transfer starts with the third file or subfile. If omitted, transfer starts at the beginning of namrl. Transfer is always to the beginning of namr2.



Number of Files

The #files parameter applies to files on non-disc devices or subfiles on disc. It specifies the number of files or subfiles to transfer starting with file#. File# and #files are specified or omitted under the following circumstances:

Transfer:	file#	#files
1 particular non-disc file or the rest of a disc file	yes	no
1 non-disc file or all of a disc file	no	no
a number of non-disc files or disc subfiles from beginning of file	no	yes
a number of non-disc files or disc subfiles from particular file or subfile	yes	yes

SV (Change Severity Code)

```
+-------+
| LEVEL |
|  20   |
+-------+
```

Allows the user to change the severity code currently being used by FMGR.

```
+-----------------------------------------------------------------+
|                                                                 |
| SV,severity [,global# [,IH ]]                                   |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| severity                                                        |
| New severity code; it may be:                                   |
|                                                                 |
|  0  Display error codes and echo commands on log device (default).
|                                                                 |
|  1  Display error codes on log device, inhibit command echo.    |
|                                                                 |
|  2  Error code displayed only if error requires transfer of con-
|     trol to log device for correction, in this case, any active |
|     batch job terminates; no command echo.                      |
|                                                                 |
|  3  Same as 2 except active job not terminated when an error    |
|     causes transfer to log device. You can transfer back to the |
|     job. No command echo.                                       |
|                                                                 |
|  4  If a FMGR command error is encountered, job continues       |
|     automatically; no command echo, no transfer to log device, and
|     no job abort occurs.                                        |
|                                                                 |
| global #                                                        |
| Optional G global number in the range 1-9, in which the original
| severity code is placed.                                        |
|                                                                 |
| IH                                                              |
| Optional parameter to inhibit echo of command entry.            |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

Create a transfer file which displays a message to the Log device; changes the severity code to a 2 storing the original severity code in global 9G; inhibits the echo of command entries; and resets the severity code to the original severity code at the end of the transfer file. (See Chapter 5 for a description of the interactive editor.)

```
:RU,EDITR  <------------  run EDITR.
SOURCE FILE?
/0
EOF
/ :SV,2,9,IH
/ :DP,MESSAGE FROM TRANSFER FILE
/ :SV,9G,,IH
/ :TR
/EC/EXMPL <-------------  end edit; create transfer file /EXMPL
END OF EDIT
```

The original severity code is 0 (default case). Transferring to the transfer file:

```
:TR,/EXMPL <------------  transfer control to transfer file /EXMPL
 MESSAGE FROM TRANSFER FILE <------  output to terminal
 :TR  <---------------------------  command echo
```

In the above example, echo of the two :SV commands is inhibited by the "IH" option. The :DP command echo has been inhibited by the severity code of 2. The :TR command was echoed back to the terminal because the severity code was changed back to a 0, the original severity code.

COMMENTS:

The normal default mode is to echo command as it is entered on the input device and to log all errors on the same device. During interactive operation, the severity code should be this default value, zero. When there is no advantage to echoing commands at the console, for instance when commands are entered in a batch job, from a peripheral device, or through a file, the severity code can be set to 1. If, in addition, it is desired to suppress messages unless they require action, the code can be set to 2. This code terminates any currently executing job when an error occurs so that the job will not continue with errors. A severity code of 3 allows jobs to continue in spite of errors. This code would be useful when more than one batch job must be processed so that the job containing errors does not hinder subsequent job processing. If SV is specified within a batch job, the severity is reset to zero when the job is terminated with the EO command.

Whenever command echo is suppressed, any command causing an error is displayed preceding the error code unless the error display is also inhibited.

SY (Execute System Command from FMGR)

LEVEL
1

Allows execution of system commands from FMGR; the following commands: FL, HE, RS, SL, and TE are not allowed; BR and OF require that a program name be supplied. See Chapter 4 for a description of system and break mode commands and their command capability level requirements.

SYcommand
command The system command mnemonic code. No delimiter is permitted between SY and the command.

EXAMPLES:

```
:SYUP,6 <-------------- Make logical unit 6 available to the system.

:SYTI <---------------- Request the time and date from the system.
```

COMMENTS:

FMGR strips off the SY prefix and passes the remaining characters to the system for execution. Any messages resulting from execution of the system command are passed back through FMGR to the log device, but do not force a transfer to the log device.

If you do not specify a logical unit number in an SY command request for the system command RU or ON, the SY command will supply the log device as the default logical unit.

The SY command is a privileged command and subject to privileged command syntax rules (refer to the PARAMETER SYNTAX RULES section).

TE (Send Message to System Console)

LEVEL
10

Allows user to send a message to the system console.

```
TELL,message
  --
```

message
Message to be sent to system console; must conform to parameter syntax rules for privileged commands (refer the PARAMETER SYNTAX RULES section).

EXAMPLE:

```
:TE,MESSAGE TO CONSOLE <------ command typed in at user's terminal.

 TE,MESSAGE TO CONSOLE <------ message displayed on system console
                               CRT.
```

COMMENTS:

The message may consist of any printable upper-case ASCII characters. It is limited by the length and number restrictions placed on any FMGR command parameters (refer to the PARAMETER SYNTAX RULES section). Note that any commas divide the message into separate parameters. A one-parameter message could be as long as 60 characters.

TR (Transfer Control to a File or LU)

```
+-------+
| LEVEL |
|   1   |
+-------+
```

Allows transfer of control to a file or logical unit, optionally passing values to the globals 1G through 9G.

```
+-----------------------------------------------------------------+
|            --                    --                             |
|            | ,namr                |                            |
| TRANSFER   | ,-integer [,parameters] |                           |
|   ------   |                      |                            |
|            --                    --                             |
|-----------------------------------------------------------------|
|                                                                 |
|                                                                 |
| namr                                                            |
| Identifies file or logical unit to which TR transfers; if omitted, |
| TR returns control to the namr of a previous transfer; up to 10 |
| transfers can be nested - a stack of return pointers is saved.  |
|                                                                 |
| -integer                                                        |
| Indicates a transfer back the specified number of files in the  |
| nested stack. The transfer stack is flushed if the integer exceeds |
| the current level of nesting.                                   |
|                                                                 |
| parameters                                                      |
| The values to be set into the globals 1G through 9G; position   |
| determines to which global the value is passed; omitted globals |
| are unchanged.                                                  |
|                                                                 |
|                          **NOTE**                               |
|                                                                 |
|       A colon (:) or a comma (,) may replace "TR," as           |
|       the transfer command code.                                |
+-----------------------------------------------------------------+
```

EXAMPLES:

1. Set up procedure file RP to restore a main program and two segments or three main programs or two main programs and a segment:

```
          :RP,1G
file      :RP,2G
"RP"      :RP,3G
          ::
```

   Transfer to RP to restore program MAIN and two segments SEG1 and SEG2: `::RP,MAIN,SEG1,SEG2`

2. Assume three procedure files, FILEA, FILEB, and FILEC stacked so that FILEA transfers to FILEB to FILEC:

```
:TR,FILEA
:

(FILEA)   :TR,FILEB

(FILEB)   :TR,FILEC

(FILEC)   :TR,-3        <------ returns to command after
                                original transfer.
```

3. Assume a control file, CNTFL, containing a command to store file name ABC into file name DEF on cartridge 3. When the command is executed, cartridge 3 is full resulting in an error. A solution is to pack cartridge 3 and then backspace 1 line while transferring back to file CNTFL, as follows:

```
:TR,CNTFL <-------------------- transfer to control file
                                file CNTFL.
      :ST,ABC,DEF::3 <-------- command line in control file.
FMGR-033 <--------------------- error message.
:PK,3 <------------------------ pack cartridge 3.
:TR,:-1 <---------------------- transfer back one line in
                                control file.
```

COMMENTS:

Whenever a TR command is executed, the "namr" associated with TR is saved in a stack that can contain up to ten file names or logical unit numbers of these ten, the first file name or LU stacked is the input device; the last entry is reserved in case an error necessitates a transfer to the log device. This allows the system to transfer back when TR or "TR, -integer" is specified. If the "namr" identifies a disc file, the current record number is also saved. The transfer returns to the point immediately following the specified TR command.

When transferring back to a control file on disc, it is possible to backspace and re-execute one or more commands within that file. To accomplish this, specify a negative integer (-integer) as the security code of a null namr. For example:

```
    :TR,:-2
or,
    :TR::-2
or,
    :::-2
```

will result in a backspace of two commands while transferring back to the previous control file.

*** NOTE ***

For non-disc devices on which a backspace is legal, e.g., LU5 in spooled jobs, the backspace record function of the CN command can be used (:CN,5,BR).

If an error requiring operator intervention occurs and the severity code is less than 4, the system transfers control to the log device. The TR command (TR or :) is used, in this situation, to return control to the procedure file or input device where the error occurred.

When parameters are specified, the parameter values are passed to the global parameters 1G through 9G. Position determines which globals receive values. For example:

```
:TR,PROG2A,,,,ABC,3270,,XYZ  <---sets 4G to "ABC", 5G to 3270,
          ^ ^ ^           ^          7G to "XYZ".
          | | |           |
          | | |           |
          | | |           +--6G
          | | +--3G
          | +--2G
          +--1G
```

WH (Run WHZAT Program)

LEVEL
10

Schedules the WHZAT program to display the current system status. Both program and partition status may be selected.

```
WH [,lu [,option ]]
```

lu
The session logical unit number of the device on which the information to be displayed. Defaults to the lu of the terminal from which command is input.

option
Indicates type of information to be displayed. Default is to display only those program associated with the user's session.

AL	Status of all scheduled and suspended programs.
SM	Same as AL option except that all state-3 programs not having "father-son" relationship will not be displayed.
PA	Status of all partitions being used.

EXAMPLES:

```
:WH  <---------- Displays  status  of  scheduled  and  suspended
                 programs  associated  with  the  user's session on
                 input terminal.

:WH,,PA <------- Displays  status,  on  input  terminal,  of  all
                 partitions being used.
```

COMMENTS:

The WHZAT program can also be run outside the session environment:

```
:RU,WHZAT[,lu[,option]]
```

For details, refer to WHZAT description in Chapter 6.

Procedure Files

A procedure file is a set of FMGR commands stored in a disc file. Instead of entering each FMGR command interactively at the terminal, a group of consecutive commands can be saved as a procedure file to which control can be transferred using the transfer command (TR or :). This feature is especially useful when the same set of commands is to be used over and over.

A procedure file can be generalized with global parameters (refer to the GLOBAL PARAMETERS section). The TR command allows the user to pass values to global parameters 1G through 9G when control is transferred to the procedure file (refer to the TR COMMAND in this Chapter).

A set of commands that manipulate parameters may be useful within a procedure file. These commands are:

Command	Purpose
SE	Assigns values to the global parameters 1G through 9G.
CA	Can assign values to most P and G-type globals which result from arithmetic or logical calculations, and can null most P and G-type globals. (Globals -36P through -1P, 1P through 6P, and 1G through 9G.)
IF	Compares two values (usually globals) and skips a specified number of commands depending on the result of the calculation.

Other FMGR commands often used in procedure files include:

Command	Purpose
**	Allows inclusion of comment lines within command entry list to explain the command flow.
DP	Displays parameter values or a message string to the log device.
PA	Suspends execution of the procedure file and transfers control to the log device or some other specified device; also, can optionally send a message to the device where control is being transferred.
SV	Changes severity code; allows inhibiting of echo commands on log device.

Each procedure file is terminated with TR or : in order to return control to the point of origin. Up to ten procedure files can be nested. The return is normally to the command following the TR in the preceding procedure file. If files are nested, the return may skip to any of the preceding files.

FMGR commands in procedure files located on disc cartridges LU 2 and LU 3, are not restricted by FMGR command capability level checking. This allows the System Manager to create procedure files on LU 2 and 3 which can circumvent certain Session Monitor constraints for special applications.

A typical example might be for a low capability level session user to have access indirectly to a higher capability level command by transferring to a procedure file which executes the command for him. Only the System Manager or a non-session user can create a procedure file on LU 2 or LU 3.

General Example Using Procedure File with Globals

This example illustrates all the commands discussed with procedure files. A procedure file MOVE is used to copy a number of files from one FMP cartridge to another without copying them all as is the case with the CO command. Four message files are used to request input specifying the cartridge to be copied, the cartridge to which files are copied, a security code for the copied files, and the names of the files themselves. If desired, the copied file can be given a new name.

When you transfer to MOVE with the TR command, you should enter the session logical unit of the terminal at which you want to enter input as the first parameter to be passed to MOVE. If omitted, a message is displayed at the log device and you must re-enter the command with the input parameter.

The messages asking for input are displayed at the terminal you specify. The procedure then pauses so that you can return the information as parameters in the : version of the TR command.

The message files are:

File #MESS1

```
PRECEDE ANSWERS TO THE FOLLOWING QUESTIONS WITH A COLON AND A
COMMA:
:,ANSWER1,ANSWER2,...
```

File #MESS2

```
ENTER THE 'CR' OF THE CARTRIDGE WITH THE FILES TO BE MOVED:
```

File #MESS3

```
  ENTER DESTINATION CARTRIDGE'S CR, FILE'S NEW SC:
```

File #MESS4

```
  NOW ENTER SOURCE AND DESTINATION FILE NAMES. IF NEW FILE IS TO
  HAVE THE SAME NAME AS OLD, ONLY ONE NAME IS REQUIRED. END LIST
  WITH /E.
```

The file MOVE:

```
  :SV,1
  :CA,4,1G <----------------save terminal LU in 4G
  :IF,-36P,EQ,1,2 <---------if not specified, display error message
  :DP,YOUR LU WAS NOT GIVEN
  :: <----------------------and return
  :DU,#MESS1,4G <-----------send first message to terminal
  :DP
  :DU,#MESS2,4G
  :PA,4G <------------------at pause, operator enters TR followed
                           by information
  :CA,6,1G <----------------CR of cartridge containing files is saved
                           in 6G
  :DU,#MESS3,4G
  :PA,4G <------------------enter cartridge and security code at pause
  :CA,8,2G <----------------security (SC) in 8G
  :CA,7,1G <----------------and cartridge (CR) in 7G
  :DU,#MESS4,4G
  :DP <---------------------note use of DP to generate blank line
  :SE,0,0,0 <---------------clear 1G, 2G, and 3G prior to entry of
                           files
  :PA,4G,FILES: <-----------enter files
  :IF,1G,EQ,/E,5
  :IF,1G,EQ,,-3
  :IF,-32P,GT,1 <-----------test for new file name
  :CA,2,1G
  :ST,1G::6G,2G:8G:7G::-1<--transfer file
  :IF,A,EQ,A,-8             return for next file
  :SV,0
  ::
```

To use this procedure:

```
  :TR,MOVE
  :SV,1
    YOUR LU WAS NOT GIVEN
  :TR,MOVE,1
  PRECEDE ANSWERS TO THE FOLLOWING QUESTIONS WITH A COLON AND A COMMA:
      :,ANSWER1,ANSWER2,...
```

```
ENTER THE 'CR' OF THE CARTRIDGE WITH THE FILES TO BE MOVED:

:PA,4G
::,2
ENTER DESTINATION CARTRIDGE'S CR, FILE'S NEW SC:

:PA,4G
::,13,JG
NOW ENTER SOURCE AND DESTINATION FILE NAMES. IF NEW FILE IS TO HAVE
THE SAME NAME AS OLD, ONLY ONE NAME IS REQUIRED.
END LIST WITH /E.

:PA,4G,FILES:
::,OLDFIL,NEWFIL <-----enter first file to be moved and rename it
:PA,4G,FILES:
::,FILEA <-------------enter next file and use same name for copy
:PA,4G,FILES:
::,/E <----------------no more files
::
:
```

Command Stacking

For FMGR commands which are input from a terminal, several typing aids are provided by the FMGR program. All FMGR commands entered from a terminal, whether they are correct and executed, or are incorrect and produce an error, are placed in a command stack, resident within the FMGR program. These commands may be displayed, modified and executed later. The modify commands are a subset of the edit commands provided by the EDITR program. The following stack manipulation commands are provided:

:Ln	"n" is the number of lines to list (default is to list the entire command stack).
:P	Display or edit the pending line in the command stack. Edit options are CNTL/R, CNTL/I, CNTL/S, CNTL/T and CNTL/C. See the Chapter on the Interactive Editor. Also see the example below.
:n	Position pending line to the "n"th line in the command stack.
:^n or Rn	Position "n" lines preceding pending line.
:/n	Position "n" lines past pending line.
:-n	Delete "n" lines from command stack from the pending line.

Once a line has been displayed as the pending line, it may be executed by typing a carriage return.

An aging feature eliminates duplicate entries from the command stack. Before a new command is placed on the stack, all other entries are checked for duplication and deleted if necessary.

To be assured of a command stack position, use "TR" for the transfer command. When a colon (:) or a comma (,) is used, the command is not placed in the command stack.

Implied RUN: All input from a terminal that is not either a legal FMGR command or command stack command will have a :RU, inserted in front of it, and it will be interpreted as a run command. This feature makes programs and FMGR commands appear to be more alike, and reduces the amount of typing required. If the entry is neither a legal FMGR or command stack command nor a program name, FMGR will return a FMGR 067 (program not found) error.

The implied RUN feature cannot be used to run programs named the same as a valid FMGR command; these programs must be run with the RU command. For example, a user program named ST, STO, STOR, or STORE and executed with the implied RUN, will cause the FGMR ST command to be executed instead of the user program. These programs will run with the RU command.

The implied run feature cannot be used in a transfer file. Attempting to do so results in a FMGR 056 (bad parameter) error.

With the implied RUN command installed, the pending line edit feature is slightly limited. As noted below, the first character of the pending line may not be edited since FMGR will recognize the input as an implicit RUN command.

Example	Comments
:L	
RU,FTN4,$TEST1,-,-	this is the listing of the command stack
:PRU,FTN4,&TEST2,-,-	user is attempting to edit the pending line
FMGR 067	FMGR issues an error, recognizes "PRU" as an implicit RUN command of the program PRU.
:P/////////////2	One correct method-replacing lst character with "/".

Chapter 4
System and Break Mode Commands

Introduction

System and break-mode commands allow the user to interactively communicate with the RTE-IVB Operating System to obtain system status, modify system parameters and control the scheduling of programs. Some of the functions performed by these commands include:

* Scheduling and terminating programs
* Suspending and restarting programs
* Scheduling programs to execute at specific times
* Changing the priority of programs
* Examining the status of any partition, program or I/O device
* Dynamically altering the I/O structure and buffering designations
* Examining and dynamically altering an I/O device's time-out parameter
* Declaring I/O devices up or down
* Initializing the real-time clock and printing the time
* Setting device buffering limits
* Releasing disc tracks assigned to dormant programs

The commands and their functions are summarized in Table 4-1; complete descriptions are given later in this Chapter.

Chapter 2 of this manual introduced the user to the operational differences between system and break-mode commands. In this Chapter, the PROCESSING OF SYSTEM AND BREAK-MODE COMMANDS section describes the fundamental differences of how the system processes the two modes.

Table 4-1. System and Break-Mode Command Summary

COMMAND	CAPABILITY LEVEL	DESCRIPTION	PAGE NO.
AB	--	Abort current executing batch job	2-11
AS	50	Assigns program to partition	4-21
BL	10	Examine current buffer limits	4-23
BL	60	Change current buffer limits	4-23
BR	10	Set break flag in program's ID segment (program within session)	4-25
BR	60	Set break flag in program's ID segment (any program in system)	4-25
DN	60	Down an I/O controller	4-27
EN	--	Enable system console to be session terminal.	2-9
EQ	10	EQT description and status	4-29
EQ	60	Change automatic buffering designation for an EQT	4-30
FL	10	Flushes terminal buffer	4-31
GO	30	Reschedules operator suspended program (program within session)	4-32
GO	60	Reschedules operator suspended program (any program in system)	4-32
HE	1	Calls HELP program	4-34
IT	50	Put program in time list	4-36
LU	60	Prints EQT assignment for LU	4-38
LU	60	Reassigns EQT assignment for LU	4-38
OF	30	Terminates a program (program within session)	4-40
OF	60	Terminates a program (any program in system)	4-40
ON	50	Schedules a program	4-42
OP	--	Allow system level operator command	2-10
PR	50	Changes program priority	4-45
QU	10	Examine current system timeslice parameters	4-46
QU	60	Modify system timeslice quantum and/or fence	4-46
RS	10	Aborts and reschedules a session's copy of FMGR	4-48
RT	30	Releases disc tracks assigned to a program	4-49
RU	30	Run a program	4-50
SL	10	Display session LU information	4-52
SS	30	Suspends a program (program within session)	4-53
SS	60	Suspends a program (any program in system)	4-53

Table 4-1. System and Break-Mode Command Summary (continued)

COMMAND	CAPABILITY LEVEL	DESCRIPTION	PAGE NO.
ST	10	Displays program or partition status	4-54
SZ	30	Prints current program size information	4-56
SZ	30	Changes program size requirement	4-56
TE	10	Send message to system console	4-59
TI	10	Print time	4-60
TM	60	Set real time clock	4-61
TO	10	Prints time-out value for EQT	4-62
TO	60	Sets time-out value for EQT	4-62
UP	10	Ups an I/O controller	4-64
UR	50	Releases a reserved partition	4-65
WH	10	Calls the program WHZAT	4-66

Program States

Within the RTE-IVB operating environment, programs can exist in 7 states. The state of a program can be changed by the user (RTE-IVB System command), by another program (EXEC call), or by the system to reflect an environmental condition, i.e., program requested memory or disc space that was not available, requested I/O on a down device, etc. At any given time, the state of a program indicates its relationship to the RTE-IVB operating environment.

State 0 - DORMANT. This state indicates that a program is not scheduled to execute. A program can be in this state if it has never been scheduled or if it has been placed in this state from a previous state by an operator command or by an EXEC call.

State 1 - SCHEDULED. This state indicates that a program is scheduled to execute; it has been placed in the scheduled list. The program that is currently executing is generally at the head of the scheduled list and is therefore in State 1 also.

State 2 - I/O SUSPENDED. This state indicates that a program has requested I/O servicing and the system is currently performing the I/O operation. This condition occurs with any input operation, or with an output operation to an unbuffered device. If the class I/O technique discussed in the RTE-IVB Programmer's Reference Manual is used, or if the output device is buffered, the program will be allowed to continue executing while the I/O operation is being performed.

State 3 - GENERAL WAIT. A program is placed in this state if it has requested system resources that are temporarily unavailable or services that temporarily cannot be performed. For example, a program would be placed in this state if it were waiting for a data buffer to be supplied by another program, or if it requested I/O on a device that was allocated to, and presently being used by, another program.

State 4 - MEMORY SUSPENDED. A program is placed in this state if it has requested an operation that requires the use of System Available Memory (SAM) and an insufficient amount of SAM is available. This is a temporary state and when enough SAM becomes available, the program will by placed back into the scheduled list.

State 5 - DISC SUSPENDED. A program is placed in this state if it has requested disc space that is unavailable. This is a temporary condition and when disc space becomes available the program will be placed back into the scheduled list.

State 6 - OPERATOR SUSPENDED. A program is placed into this state by an operator command or by an EXEC call from within a program. Another operator command or EXEC call is necessary to remove the program from this state.

Program State Lists

RTE uses the LINKED LIST technique for connecting programs within a given state list. Using this technique, RTE moves programs from state to state by linking and unlinking the program's ID segment between the appropriate lists.

Each state list originates with a LIST HEADER pointing to the ID segment of the first program in the list. The first word of each program's ID segment is then used as a LINKAGE WORD pointing to the ID segment of the next program in the list. The ordering of the programs within the list is on a priority basis with the highest priority program first.

The list headers for program states 1, 3, 4, 5 and 6 are located within the system base page (see the Memory Management section of the RTE-IVB Programmer's Reference Manual for a description of the system base page). The list header for program state 2 (I/O Suspend) is the first word of the Equipment Table (EQT) entry for the device to which I/O is being directed or requested. Note that state 0 (dormant) has no list associated with it.

Figure 4-1 illustrates examples for a scheduled and an operator suspended state list. Both lists originate with the list header containing the address of the first program's linkage word. Since there is only one program in the operator suspended list, the program's linkage word contains a zero to indicate that it is the last program in the list. In the scheduled list, each program's linkage word contains the address of the next program's linkage word except the last which contains a zero.

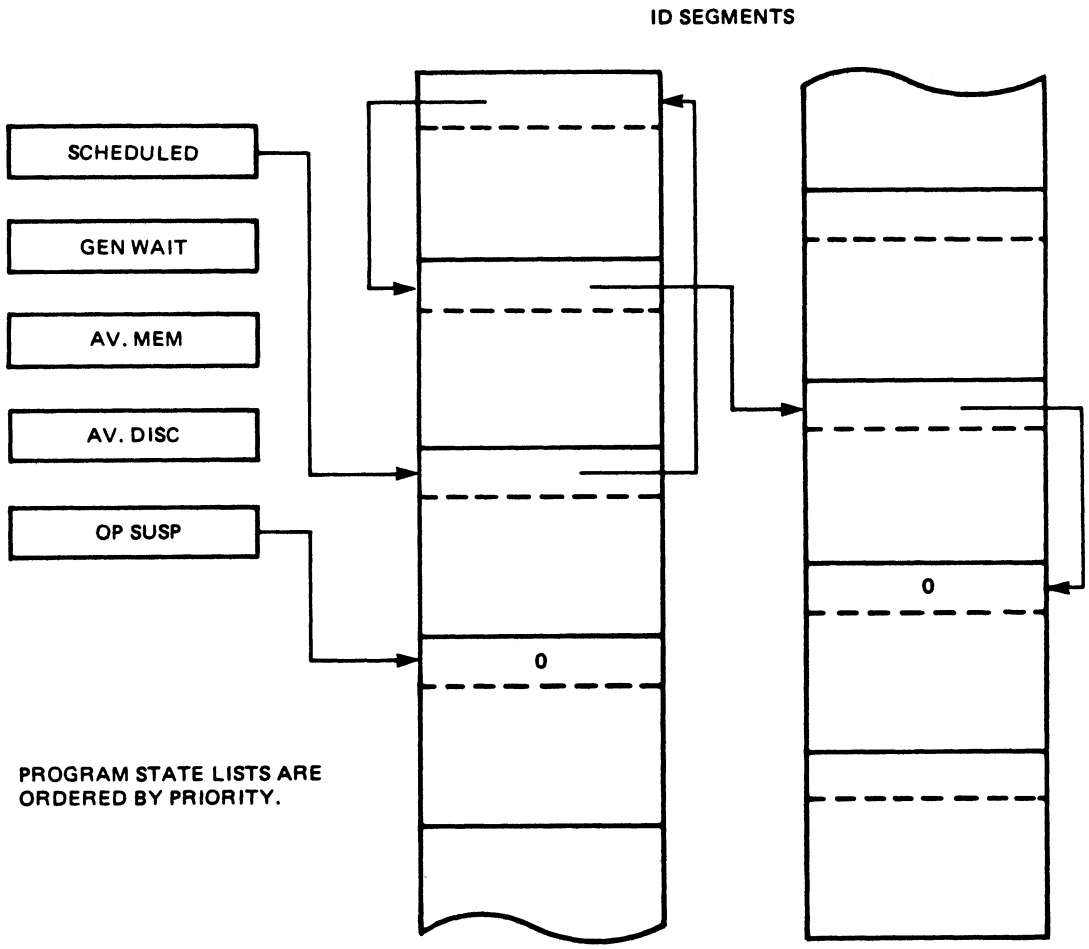


Figure 4-1. Program State List Examples

Buffered and Unbuffered Input/Output

The four ways in which input/output (I/O) operations may be processed in RTE are:

1. Unbuffered I/O
2. Re-entrant I/O
3. Automatic Output Buffering
4. Class I/O

Unbuffered and re-entrant I/O require that the I/O process be completed before the program can continue executing (i.e., the program will be I/O suspended). For disc resident programs, the main difference between the two I/O processes is that re-entrant I/O allows program swapping and unbuffered I/O does not.

Both unbuffered and re-entrant I/O link the program ID segment to the Equipment Table (EQT) entry of the device to which I/O is directed or requested. However, with re-entrant I/O, a temporary data block (TDB) is created in System Available Memory (SAM) to act as a buffer to the device. Since I/O is read from or to the buffer instead of the program, the program can then be swapped out if a higher priority program needs its partition. The TDB is linked to the program and EQT through the program's ID segment.

Automatic output buffering and class I/O allow programs to continue executing while I/O is being processed. Both processes create buffers in System Available Memory (SAM) to which data is temporarily stored. The SAM buffer is then linked to the device's EQT.

Figure 4-2 illustrates the basic difference between buffered and unbuffered output to a device (for a read operation, the arrows would merely point in the opposite direction). Figure 4-3 illustrates how I/O requests are linked to the EQT entry for buffered and unbuffered I/O.

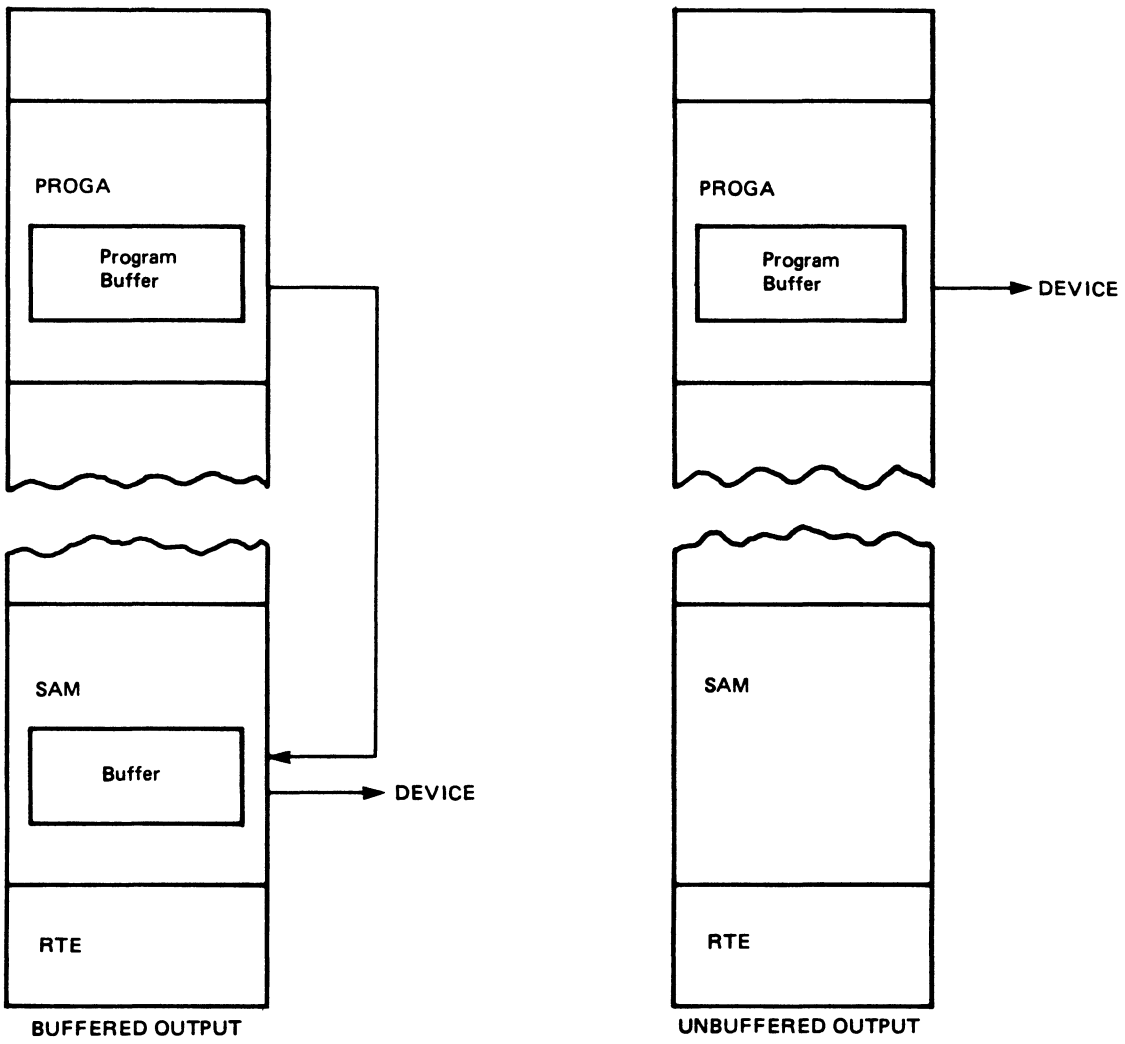


Figure 4-2. Buffered and Unbuffered Output to a Device

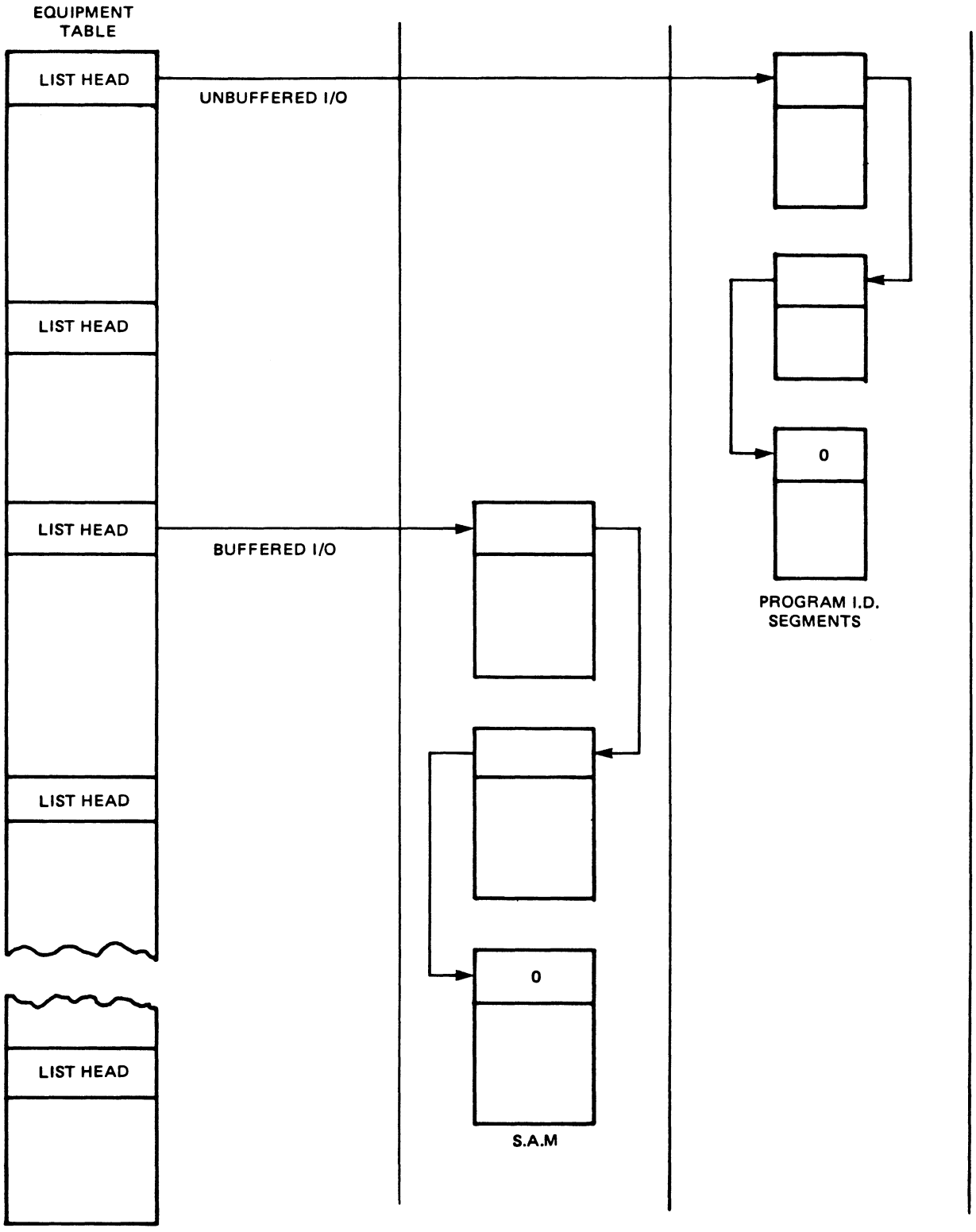


Figure 4-3. Buffered and Unbuffered I/O Linkage

A device can be designated as using automatic output buffering or being unbuffered by setting the appropriate bit in its associated Equipment Table (EQT) entry. This bit can be set at either system generation (see RTE-IVB On-Line Generator Manual) or interactively through the break-mode EQ command described later in this section.

There are two situations where an EQT can have both program ID segments and SAM buffers directly linked to it:

1. The EQT is unbuffered and Class I/O requests are linked to it.

2. The EQT uses automatic output buffering and unbuffered and/or re-entrant I/O read requests are linked to it.

Figure 4-4 illustrates an example where both SAM buffers and program ID segments are linked to the same EQT.

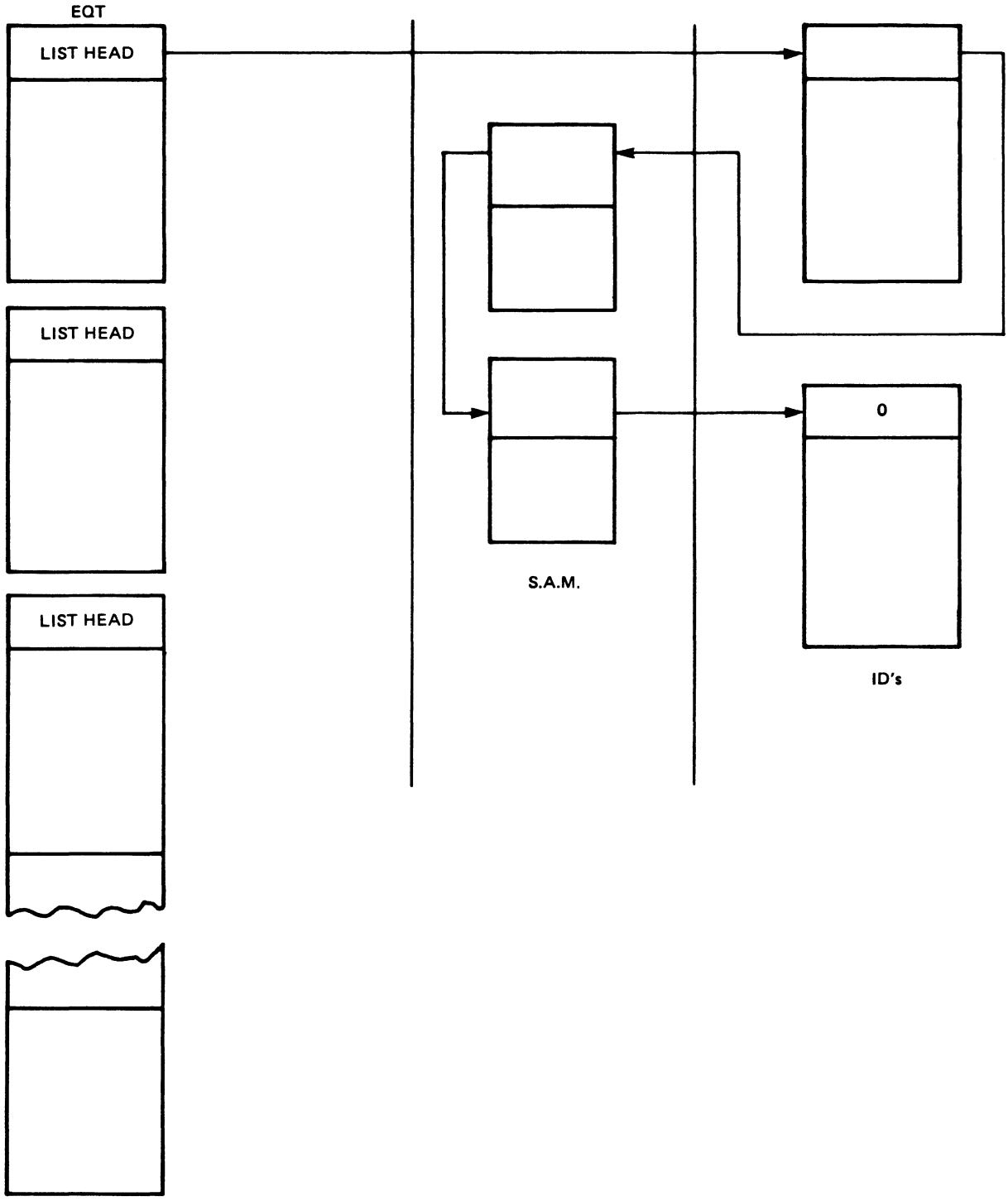


Figure 4-4. I/O Request Linkage

Time List

RTE allows users to schedule programs to execute at specific times by placing them in the TIME LIST. When the specified time for the program to execute arrives, it is placed in the scheduled list by RTE.

Programs in the time list are linked together through word 17 of their respective ID segments. The ordering of the programs is based upon a first-in-first-out (FIFO) basis. Also stored within the ID segment is the next execution time and execution time interval for the program.

When the next execution time for the program arrives, the program is placed in the scheduled list by RTE (see PROGRAM STATE LISTS in this Chapter). Figure 4-5 illustrates an example of a time list and a scheduled list. Note that programs PROGA and PROGC are in both the time list and scheduled list.

If a session program is in the time list, the following access restrictions are enforced:

1. EXEC schedule or program time value requests referencing a program in the time list may only be issued by another program of the same session. An attempt to reference a session's time scheduled program by another session, or a non-session program will result in an SC11 error.

2. The session operator commands IT, RU and ON have the same access restrictions as described above. Attempts to reference a time scheduled program belonging to another session will result in an "ILLEGAL STATUS" error.

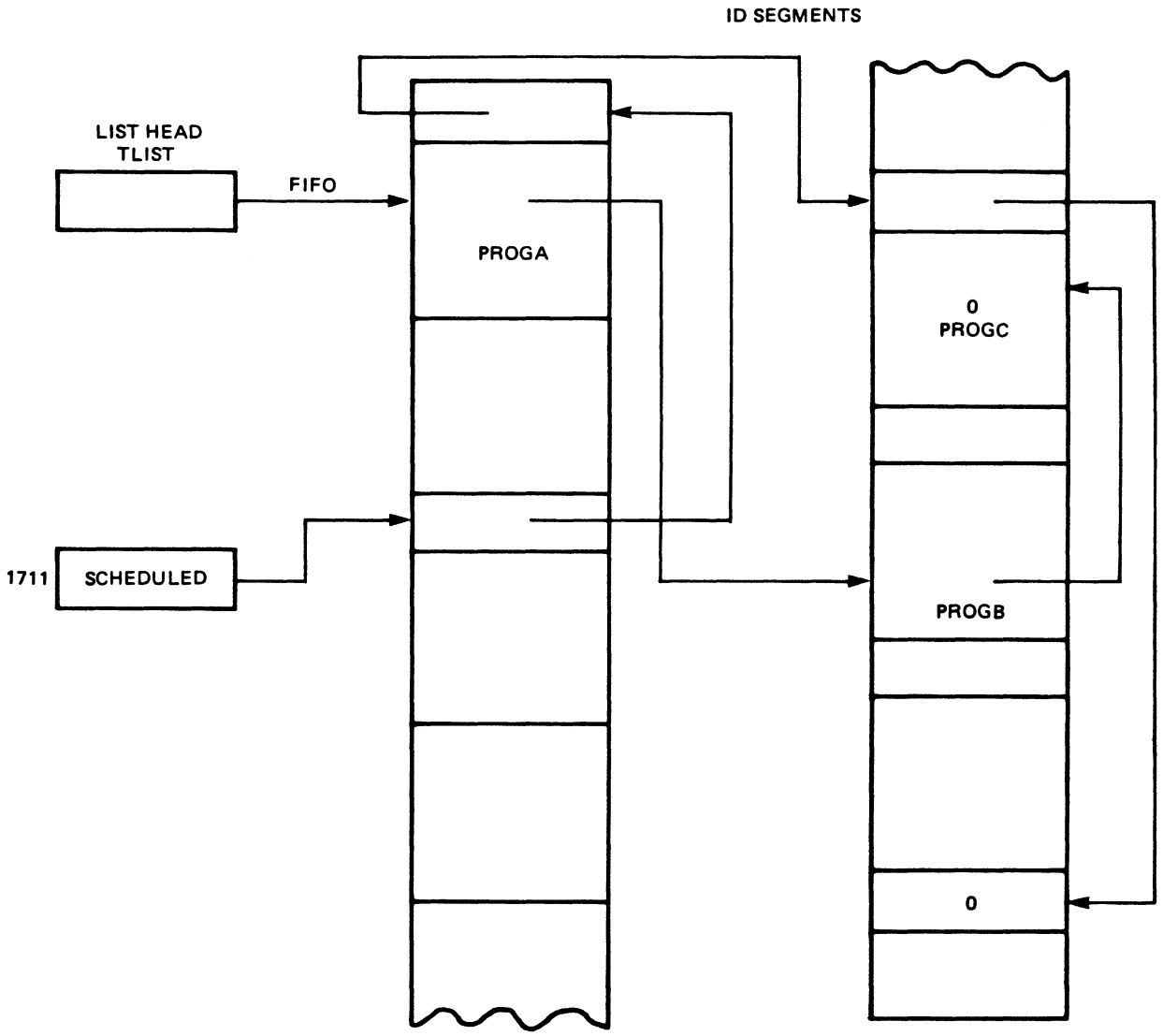


Figure 4-5. Time List Example

Processing of System and Break Mode Commands

Processing of Break-Mode commands is done through the PRMPT and RPN software modules. Both the Session Monitor (SM) and the Multi-Terminal Monitor (MTM) have their own versions of these two modules. System commands are processed by the operating system.

Break-Mode can only be entered from terminals which schedule the prompt processor (PRMPT) when the user makes an unsolicited interrupt at the terminal. When operating with the Session Monitor, Break-Mode can be entered from any session terminal, including the system console if it has been enabled to operate in session. When operating with the Multi-Terminal Monitor, Break-Mode can be entered from any auxiliary terminal (i.e., any terminal other than the system console).

An unsolicited interrupt occurs when the user strikes a key while the system does not have a read request pending from the terminal (i.e., no prompt is issued to the CRT). After the interrupt is recognized by the system, the Break-Mode prompt:

```
S=xx COMMAND?          (for a session terminal)

     or

lu>                    (for an MTM terminal)
```

Where xx is the session identification number and lu is the logical unit number of the MTM terminal, is issued to the terminal CRT.

The response processor (RPN) is scheduled once by PRMPT and then suspends itself on a class "GET" request (see the RTE-IVB Programmer's Reference Manual for a description of Class I/O). All operator requests input in Break-Mode are thus passed to the RPN module through the class "GET" request.

The RPN module passes most of the commands entered through Break-Mode directly to the operating system for processing, but some commands require special handling which the RPN module processes. The following commands are session related and receive special handling by RPN:

FL - make a control request to flush all requests to this terminal's driver until all buffered requests have been cleared or a read request is found.

WH - schedule the WHZAT program for this session.

HE - schedule the HELP program for this session.

SL - examine contents of Session Switch Table (SST).

TE - send a message to the system console.

RS - abort and then restart this session's copy of FMGR.

OF - if no parameters, abort the current program running under the session's copy of FMGR (FMGxx). If FMGxx does not have a son running, the break flag is set in FMGxx's ID segment.

BR - if no parameters, set the break flag of the current session program.

If Break-Mode is entered while operating under the Multi-Terminal Monitor, the following commands receive special handling by RPN (the MTM Version):

FL - Make a control request to flush all requests to this terminal's driver until all buffered requests have been cleared or a read request is found.

BR - if no parameters, set the break flag of the last son of FMGxx. If FMGxx does not have a son running, the break flag is set in FMGxx's ID segment.

AB - if no parameters, abort the last son of FMGxx (same as OF,program,1). If FMGxx does not have a son running, the break flag is set in FMGxx's ID segment.

Note that several commands available under Session Monitor are not available under MTM (i.e., WH, HE, SL, TE, and RS).

All other commands entered during Break-Mode are passed to the system for processing. See Table 4-1 for a complete listing of the Break-Mode commands.

Three commands which are only valid when entered from the system console are:

AB - abort the currently executing batch job.

EN - enable system console to be session terminal.

OP - allow system level operator command.

For more information on these commands, see the System Console description in Chapter 2.

Commands which the operating system processes can also be entered from File Manager (FMGR) via the SY command (see Chapter 3 on FMGR commands for more details). File Manager strips off the SY prefix and passes the remaining command string directly to the operating system.

Figures 4-6 and 4-7 summarize System and Break-Mode command processing under Session Monitor and MTM respectively. The prompts shown are:

1) * <--- System Prompt (System Console)

2) : <--- FMGR Prompt

3) S=xx COMMAND? <--- Session Break-Mode Prompt

4) lu> <--- MTM Break-Mode Prompt

By following the arrowed paths starting with the appropriate prompt, the user can quickly see which commands are valid for each prompt.

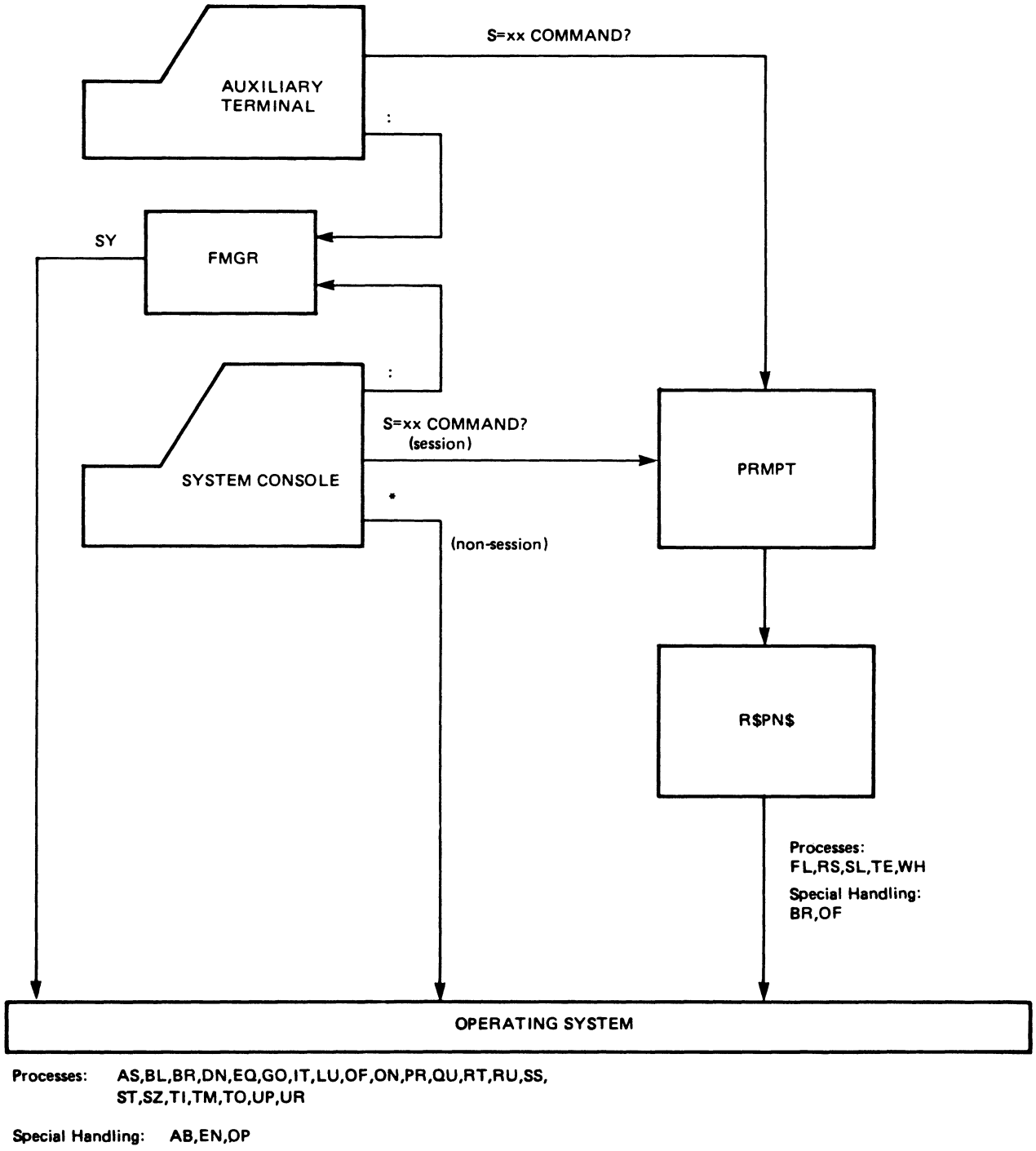


Figure 4-6. Break-Mode Command Processing Under Session Monitor

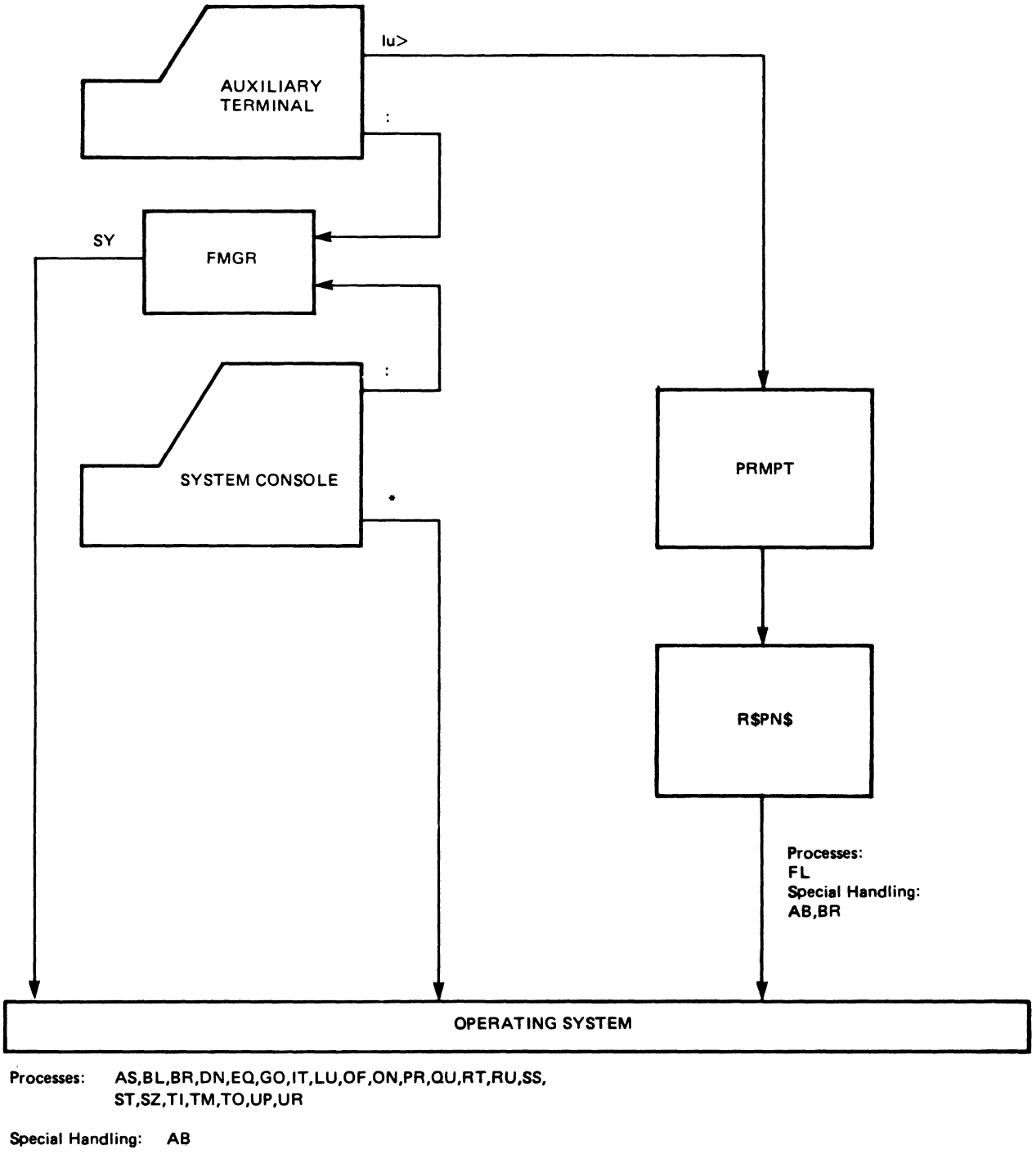


Figure 4-7. Break-Mode Command Processing Under MTM

System and Break Mode Commands

The following are detailed explanations of the interactive system and break-mode commands available with RTE-IVB. System Commands are handled by the operating system; break-mode commands are handled by the program RPN (see Figures 4-6 and 4-7). The capability level required for each command is shown in a box in the upper right hand corner of the page.

Command Structure

Each command is specified by a mnemonic code of two letters to indicate the operation to be performed. Depending on the command, parameters may be entered to further specify the command operation.

Command Conventions

The command conventions used in the following Break-Mode command descriptions are summarized in Table 4-2. Additionally, the following comments are worthwhile to note:

* When a command is entered, the items outside the brackets are required symbols. Items inside the brackets are optional.

* Two commas in sequence defaults a parameter to its default value.

* Each command entered must be completed with an end-of-record terminator (RETURN or ENTER key on a CRT or TTY system input device).

* An error made while entering a command parameter can be corrected by using the BACK SPACE key on a CRT system input device (the CONTROL and A keys struck simulataneously will delete the last character entered on TTY input devices). To delete an entire line, use the DEL key (RUBOUT key on TTY devices). Corrections to a command must be made before the RETURN key is pressed or the system will issue an error return. Note that line feed is supplied by the system.

* Whenever the operating system is rebooted, parameters changed by user command will be restored to their original values established during system generation.

The required command capability levels for each of the Break-Mode commands are shown in the upper right hand corner of the command descriptions. Note that several commands have two levels associated with them, the higher level giving additional capability.

Table 4-2. Command Syntax Conventions

Item	Meaning
UPPER CASE LETTERS	These words are literals and must be specified as shown.
lower case letters	Symbolic representations indicating what type of information is to be supplied.
[,item]	Items with brackets are optional. However, if item is not supplied, its position must be accounted for with a comma; this causes item to automatically default.
,item 1 ,item 2 ,item 3	Indicates that exactly one item may be specified.
,item 1 ,item 2 ,item 3	Indicates that there is a choice of entries for the parameter, but one parameter must be specified.

AS (Assign Partition to a Program)

```
+-------+
| LEVEL |
|  50   |
+-------+
```

Assigns a program to always be dispatched to same partition when scheduled for execution.

```
+-----------------------------------------------------------------------+
|                                                                       |
| AS,program,partition#                                                 |
|                                                                       |
|-----------------------------------------------------------------------|
|                                                                       |
| program                                                               |
| Name of program to be assigned to specified partition.                |
|                                                                       |
| partition#                                                            |
| Partition number which program is to be assigned to (since a          |
| maximum of 64 partitions can be defined at system generation,         |
| this parameter can be any integer from 1 to 64 depending upon the     |
| the number of memory partitions defined at generation).               |
|                                                                       |
+-----------------------------------------------------------------------+
```

EXAMPLE:

```
S=07 COMMAND ?AS,PROGA,12
\-----v------/     ^    ^
  Break-Mode       |    |
    Prompt         |    --partition number
                   |
                   --program name
```

COMMENTS:

Unless a program is assigned to a specific partition, when it is scheduled for execution it is dispatched to any partition of the proper type large enough to run the program.

The AS command is only valid if the program specified is dormant and is not currently residing in a partition (a program may be dormant and residing in a partition if it ended saving resources, ended serially reusable or was operator suspended); otherwise, the error 'ILLEGAL STATUS' will be returned and the command input ignored by the system.

If the partition specified is not large enough to run the program, an 'ILLEGAL PART'N' error will be returned. Trying to assign a program to an undefined partition will also generate the 'ILLEGAL PART'N' message.

If the program specified cannot be found (i.e., its ID segment cannot be located), a 'NO SUCH PROG' error message will be issued.

If partition# is specified as 0, the program will be unassigned and can be dispatched to any partition of the proper type large enough to run the program.

BL (Examine or Modify Buffer Limits)

```
+-------+
| LEVEL |
| 10/60 |
+-------+
```

Allows a low capability level user (10) to examine the current buffer limits, and a high capability level user (60) to change the current buffer limits.

```
+-----------------------------------------------------------------+
|                                                                 |
| BL [,lower limit [,upper limit ]]                               |
|    \------------v---------------/                               |
|     Requires level 60 to specify                                |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| lower limit                                                     |
| Lower buffer limit specified in number of words.  If upper limit|
| is changed and lower limit is not specified it defaults to 0.   |
|                                                                 |
| upper limit                                                     |
| Upper buffer limit specified in number of words.  If lower limit|
| is changed and upper limit is not specified it remains the same |
| as the existing upper buffer limit.                             |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

```
S=07 COMMAND ?BL     <----Examine current buffer limits.
  100  400

   |    |
   |    --current upper limit
   --current lower limit

S=07 COMMAND ?BL,200,500  <---Change buffer limits.

S=07 COMMAND ?BL     <----Examine current buffer limits.
  200  500

   |    |
   |    --new upper limit
   --new lower limit
```

COMMENTS:

Each time a standard I/O request is made to a device with automatic output buffering enabled, the system adds up all the words in the I/O requests currently queued to the equipment table (EQT) entry pointing to the device where I/O is directed. If the sum is less than the upper limit, the new request is added to the queue. If the sum is larger than the upper limit, the requesting program is suspended and planned in the general wait (STATUS=3) list.

When a buffered I/O request completes, the system adds up the remaining words in the I/O requests queued to the EQT entry and compares the number to the lower limit. When the sum is less than the lower limit, any programs suspended for exceeding the buffer limits on the EQT are rescheduled.

Any program with a priority of 1 through 40 will not be suspended for buffer limit, so that alarm messages, etc., are not inhibited.

Setting the upper and lower buffer limits with this command can prevent an inoperative or slow I/O device from monopolizing System Available Memory (SAM).

For instance, by increasing the buffer limit, a program can be allowed to store its output in a buffer and continue executing rather than possibly be I/O suspended because the buffer is already full.

BR (Set Break Flag)

```
+-------+
| LEVEL |
| 10/60 |
+-------+
```

Sets a break flag in a program's ID segment. With a capability level of 10 or greater, the user can set the break flag in a program currently running under his session. With a level of 60, the user can set the break flag in any program in the system.

```
+-----------------------------------------------------------------+
|                                                                 |
| BR [,program]                                                   |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| program                                                         |
| Name of program whose break flag is to be set; default is current |
| session program.                                                |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

1.
```
S=07 COMMAND ?BR,PROGC    <----Set break flag in PROGC's ID segment.
```

2.
```
S=07 COMMAND ?BR          <----Set break flag in current session
                               program's ID segment (the "current"
                               session program is found by searching
                               for the last son of the session
                               progenitor, FMGxx)
```

3.
```
S=07 COMMAND ?BR,PROGC    <----Session user with command capability
                               level of 60 sets break flag in program
                               currently running under another user's
                               session.
```

COMMENTS:

The BR command allows the user to break execution of a program if the program requests this via the IFBRK system subroutine. When BR is executed, a break flag in the named user program's ID segment is set. The user's program can call the HP-supplied subfunction that will test the break flag and then act accordingly. The calling sequence is:

```
I=IFBRK(IDUMY)
```

Where IDUMY is a dummy parameter to make the call appear as a function (IDUMY need not be supplied in Assembly Language). The returned value will be negative if the break flag is set; the break flag will be cleared by IFBRK.

DN (Down an I/O Controller or Device)

LEVEL
60

Declares an I/O controller or device down (i.e., unavailable for use by the RTE system).

```
+-------------------------------------------------------------------+
|                                                                   |
| DN,eqt    <---------Downs an EQT                                  |
|                                                                   |
|   or                                                              |
|                                                                   |
| DN,,lu    <---------Downs an LU                                   |
|                                                                   |
|-------------------------------------------------------------------|
|                                                                   |
| eqt                                                               |
| Specifies the equipment table (EQT) entry number of the I/O       |
| controller to be set down.                                        |
|                                                                   |
| lu                                                                |
| Specifies the system logical unit (LU) number of the I/O device   |
| to be set down.                                                   |
|                                                                   |
+-------------------------------------------------------------------+
```

EXAMPLE:

```
  S=07 COMMAND ?LU,9      <----Find out what EQT system logical unit
                               9 points to.
  LU  #  9 = E 9
               ^
               |--EQT 9

  S=07 COMMAND ?LU,32     <----Find out what EQT system logical unit
                               32 points to.
  LU  # 32 = E 9 S 1
               ^   ^
               |   |--subchannel 1
               |--EQT 9
```

Both system logical units 9 and 32 point to EQT 9. If it is desired to selectively down one of these devices and not the other the second format for the DN command can be used:

```
  S=07 COMMAND ?DN,,32
```

System LU 9 will still be up (available for use) in this case. If all devices pointing to EQT 9 are desired to be downed the first format for the DN command can be used:

```
  S=07 COMMAND ?DN,9
```

COMMENTS:

Setting an I/O controller (EQT entry) down effectively sets all devices connected to the I/O channel down by blocking any I/O operations on the select code. The state of the devices (LUs) associated with the select code are unchanged.

Setting the I/O device (LU) down will make only the specific device unavailable. However, all other LUs pointing to the device will also be set down. Other devices using the device's I/O select code are unaffected.

The I/O controller or I/O device remains unavailable until the I/O controller is set up by the UP command. The operator might set a device down because of equipment problems, tape change, etc.

EQ (Status of EQT Entry)

```
+-------+
| LEVEL |
|  10   |
+-------+
```

Prints the description and status of an I/O controller, as recorded in the equipment table (EQT) entry.

```
+-----------------------------------------------------------------+
|                                                                 |
| EQ,eqt                                                          |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| eqt                                                             |
| Equipment table entry number of an I/O controller.              |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

```
  S=07 COMMAND ?EQ,5
    21 DV.05 0 B U 1 2

     |   |   | |   | |--status (see comments below).
     |   |   | |   |--last subchannel addressed.
     |   |   | |--automatic output buffering used.
     |   |   |--DMA not required.
     |   |--driver routine.
     |--I/O select code.
```

COMMENTS:

The status information is printed as:

```
   select code  DV.nn  D  B  Unn  status
```

where:

```
   select code is the I/O select code number.
   DV.nn       is the driver routine.
   D           is D if DMA required; 0 if not.
   B           is B if automatic output buffering is used; 0 if not.
   Unn         is the last subchannel addressed.
   status      is the logical status:

                   0 = available
                   1 = I/O controller unavailable (down)
                   2 = I/O controller unavailable (busy)
                   3 = waiting for DMA assignment
```

Note that if EQT is 0, it is a bit bucket, as is any logical unit associated with it.

EQ (Buffering)

LEVEL
60

Changes the automatic buffering designation for a particular I/O controller.

```
EQ,eqt [,UN|,BU]
```

eqt
Equipment table (EQT) entry number of the I/O controller.

UN
Turns off buffering (UNbuffer).

BU
Turns on buffering (BUffer).

EXAMPLE:

```
S=07 COMMAND ?EQ,5      <---Check status of EQT 5.
  21 DV.05 0 B U 1 2
             ^
             |--automatic output buffering used.

S=07 COMMAND ?EQ,5,UN   <---Turn off buffering.

S=07 COMMAND ?EQ,5      <---Check new status of EQT 5.
  21 DV.05 0 0 U 1 2
             ^
             |--automatic output buffering turned off.
```

COMMENTS:

When the system is rebooted from disc, all buffering designations are reset to the values originally specified during generation.

FL (Flush Terminal Buffer)

LEVEL
10

Eliminates buffered output to an auxiliary terminal.

FL
No parameters required.

EXAMPLE:

```
S=07 COMMAND ?FL
```

COMMENTS:

The FL command is only valid when in break-mode. It is illegal if entered from the system console unless it has been enabled to run under session control.

Other methods for clearing the buffer are using the EXEC call (see RTE-IVB Programmer's Reference Manual for descriptions of the EXEC calls):

```
CALL EXEC(3,23B,lu)
```

or the FMGR command:

```
:CN,lu,23B
```

where lu is the logical unit number of the terminal to be flushed (normally LU 1 when under session control).

GO (Reschedule Program)

```
+-------+
| LEVEL |
| 30/60 |
+-------+
```

Reschedules a program previously suspended by the SS command or a Suspend EXEC call. Users with capability levels of 30 or higher can reschedule any program suspended within their session, and users with capability levels of 60 or higher can reschedule any suspended program in the system.

```
+-----------------------------------------------------------------+
|                                                                 |
| GO,program [,p1 [,... [,p5 ]]]]]    <---passes command string.  |
|                                                                 |
|    or                                                           |
|                                                                 |
| GOIH,program [,p1 [,... [,p5 ]]]]]  <---inhibits passing of     |
|                                         command string.         |
|    or                                                           |
|                                                                 |
| GO                                  <---reschedule the current  |
|                                         session program.        |
|-----------------------------------------------------------------|
|                                                                 |
| program                                                         |
| Name of suspended program to be rescheduled for execution; default |
| is current session program.                                     |
|                                                                 |
| p1 ... p5                                                       |
| Parameters to be passed to program; only passed if program has  |
| suspended itself through EXEC call, ignored if program was      |
| operator suspended with the SS command.                         |
+-----------------------------------------------------------------+
```

EXAMPLES:

```
1. S=07 COMMAND ?SS,PROGA      <---User suspends program PROGA which
                                   was scheduled within his session.

   S=07 COMMAND ?GO,PROGA      <---User reschedules PROGA to execute.

2. S=07 COMMAND ?GO,PROGB,1,3 <---User reschedules PROGB which had
                                   suspended itself via an EXEC call
                                   and passes the constants 1 and 3
                                   to the program.

3. S=07 COMMAND ?RU,PROGA      <---User A schedules PROGA to execute.

   S=07 COMMAND ?SS,PROGA      <---User B, who has a command capability
                                   level of 60, suspends PROGA.

   S=07 COMMAND ?GO,PROGA      <---User B reschedules PROGA to execute.
```

4. S=07 COMMAND ?SS <---Suspend current session program.

   S=07 COMMAND ?GO <---Reschedule current session program.

COMMENTS:

The GO command is illegal if the program has not been suspended previously by the operator or has not suspended itself.

Parameters p1 through p5 can be entered in ASCII or numeric form. Octal numbers are designated by the "B" suffix and negative numbers by a leading minus sign. For example:

```
GO,program,FI,LE,3106lB
```

Note that only two ASCII characters per parameter will be returned by a RMPAR subroutine call: if one is given, the second character is passed as a blank (blank = 40B). If the first parameter is ASCII "NO" it must then be repeated (the system interprets it as "NOW" in the GO command). For example:

```
GO,program,NO,NO,FI,3,4,5
```

is interpreted as shown below. NO (NOW) is not used except to push out the parameters.

```
NO
FI
3
4
5
```

After a program has suspended itself and is restarted with the GO command, the parameters passed by GO are stored in the TEMP area of the program's ID segment. An immediate call to library subroutine RMPAR (refer to DOS/RTE Relocatable Library Reference Manual for details on RMPAR) retrieves the parameters. If the program has not suspended itself (via SS command), the parameters are ignored.

The program may also recover the ASCII command string (up to 80 characters typed after the prompt) that scheduled it by using the String Passage EXEC call (see RTE-IVB Programmer's Reference Manual for a description of this EXEC call). If the program was rescheduled with a GOIH (inhibit string passage) or if the program has not suspended itself, the command string is not passed.

The current session program is the last son of the session progenitor (FMGxx) or FMGxx itself.

HE (Help Function)

```
+-------+
| LEVEL |
|   1   |
+-------+
```

Provides a detailed explanation of an error and guidance in possible corrective action.

```
HELP [,keyword [,lu ]]
  --
```

keyword
A select group of eight character (or less) identifiers related to the system or any of the subsystems. The keyword may be an error code itself. Its default is the last error posted to the user's session control block (SCB).

lu
Logical unit number of device where explanation will be output. Default is session user's terminal.

EXAMPLES:

```
S=07 COMMAND ?HE                <---Displays explanation of last error
                                    posted to user's session control
                                    block.

S=07 COMMAND ?HE,FMGR-006,6     <---Displays explanation of FMGR error
                                    code -006 on line printer (LU 6).
```

COMMENTS:

All keywords and their corresponding explanations are contained in a disc resident help file. A standard list of keywords provided with RTE-IVB include the error codes of several systems, subsystems and utility programs.

One of the features of the HELP function is that the System Manager can modify existing entries in the help file and can create additional keywords and add them to the help file. This feature is further discussed in the RTE-IVB System Manager's Manual.

HELP may also be invoked by the File Manager HE command or by running the program HELP. For example:

```
:RU,HELP[,keyword[,lu]]
  or
S=xx COMMAND ?RU,HELP[,keyword[,lu]]
  or
S=xx COMMAND ?ON,HELP[,keyword[,lu]]
```

where xx is the session identification number which the system uses to identify each session.

The Break-Mode HE Command is only available in the session environment.

IT (Interval Timer)

```
+-------+
| LEVEL |
|  50   |
+-------+
```

Sets time values for a program so that it automatically executes at selected times when scheduled with the ON command.

```
+-----------------------------------------------------------------+
|                                                                 |
| 1. Put program into Time List:                                  |
|                                                                 |
|    IT,program,res,mpt [,hr [,min [,sec [,ms ]]]]                |
|               \--v--/ \-----------v------------/                |
|                 set      set initial start time                 |
|               interval                                          |
|                timer                                            |
|                                                                 |
| 2. Take program out of Time List (program must be dormant):     |
|                                                                 |
|    IT,program                                                   |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| program                                                         |
| Name of the program to be placed in Time List.                  |
|                                                                 |
| res                                                             |
| Designates resolution code:                                     |
|    1 - tens of milliseconds                                     |
|    2 - seconds                                                  |
|    3 - minutes                                                  |
|    4 - hours                                                    |
|                                                                 |
| mpt                                                             |
| Multiplier used in conjunction with resolution code to set time |
| interval; number can be from 0 to 4095.  If 0 is specified, the |
| program runs only once.                                         |
|                                                                 |
| hr,min,sec,ms                                                   |
| Parameters set initial start time in terms of hour, minute,     |
| second, and tens of milliseconds.  Default for any parameter not|
| specified is zero (0).                                          |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

1. Schedule program PROGA to execute for the first time at 1:00 p.m. and run every hour after that.

```
S=07 COMMAND ?IT,PROGA,4,1,13
S=07 COMMAND ?ON,PROGA
```

2. Schedule program PROGB to execute now and run every 15 minutes.

```
S=07 COMMAND ?IT,PROGB,3,15
S=07 COMMAND ?ON,PROGB,NOW
```

3. Take program PROGB out of the Time List (i.e., it will no longer run every 15 minutes). PROGB is dormant.

```
S=07 COMMAND ?IT,PROGB
```

COMMENTS:

The resolution code (res) is the units in time to be multiplied by the execution interval value (mpt) to get the total time interval. Thus, if res=2 and mpt=100, the program specified would be scheduled every 100 seconds. If hr,min,sec and ms are present, the first execution occurs at the initial start time specified by these parameters (the program must be initialized with the ON command).

When the system is rebooted from the disc, time values set by the IT command are lost, and the original time values set at original load time are reinstated.

Format 1 of the IT command puts the program into the Time List. This is a list of all programs which are to be executed at specific times in the future. When the time for the program to execute arrives, the program is also linked to the scheduled list.

When the session user logs off, any programs placed in the Time List during the session will be removed.

The IT command is similar to the Execution Time EXEC Call (see the RTE-IVB Programmer's Reference Manual for a description of this EXEC call).

Refer to the Time List description in this Chapter for access restrictions to time scheduled session programs.

LU (Assignment or Reassignment)

```
+-------+
| LEVEL |
|  60   |
+-------+
```

Can print the EQT entry number, device subchannel number, and I/O device status currently associated with a system logical unit number, or can change the EQT and subchannel associated with a system logical unit number.

```
+-----------------------------------------------------------------+
|                                                                 |
| LU,lu                              <---Prints current LU assignment
|                                        and status.              |
|                                                                 |
| LU,lu,eqt [,subchannel ]           <---Changes LU assignment.   |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| lu                                                              |
| System logical unit number for which assignment information is  |
| desired or for which reassignment is desired.                   |
|                                                                 |
| eqt                                                             |
| Equipment table (EQT) entry number to assign LU.  If zero (0) is|
| specified, lu becomes the bit bucket.                           |
|                                                                 |
| subchannel                                                      |
| Subchannel number (0 to 31) to assign lu.                       |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

1.

```
  S=07 COMMAND ?LU,7       <---Print current LU assignment and status
                                for system logical unit 7.
  LU # 7 = E12 S 1 D
       ^     ^   ^ ^
       |     |   | |--I/O device status (down in this case)
       |     |   |--Subchannel number
       |     |--EQT number
       --System logical unit number
```

If the LU's device is unavailable (down), a D is printed as the status; otherwise the position is left blank.

2.

```
  S=07 COMMAND ?LU,8,0     <---Make LU 8 the bit bucket.
```

COMMENTS:

The SL command also gives EQT and device status for a logical unit number. Information on down devices can also be obtained with the WH command.

The SL command is only available under session control. It uses session logical units rather than system logical units to specify a device for which information is desired. See the SL command for more details.

Restrictions for changing system logical unit assignments are:

a. LU 1 (system console) must be an interactive console device.

b. LU 2 (system disc) and LU 3 (auxiliary disc) cannot be changed to another EQT entry number.

c. An LU cannot be changed to point at the same device as LU 2 or LU 3.

When an irrecoverable problem occurs on an I/O device, the operator can bypass the downed device for future requests by reassigning the logical unit number to an operable device on another select code or essentially flushing the command by reassigning it to the bit bucket.

When the system is rebooted from the disc, all LU assignments are reset to those originally established during generation.

OF (Terminate Program)

LEVEL
30/60

Terminates a program. If user has capability level of 30 or higher, can terminate any program within caller's current session. If user has capability level of 60 or higher, can terminate any program in the system.

OF,program,0 or	<---If I/O suspended, waits for completion of I/O before terminating; program's disc tracks are not released.
OF,program,1 or	<---Immediately terminates program; releases program's disc tracks.
OF,program,8 or	<---Immediately terminates program; releases program's disc tracks; if program is temporary program loaded on-line, it is removed from the system (see Relocating Loader in Chapter 6).
OF	<---Performs an OF, program, 1 on the current Session program.

program Name of the program to be terminated.

EXAMPLES:

1. Purge a temporary program (i.e., one that was loaded on-line with the LOADR) from the system.

```
S=07 COMMAND ?OF,PROG,8
```

If an attempt is later made to schedule PROG, a FMGR-067 error (program not found) will result. To run the program, it will first have to be re-loaded into the system.

2. Immediately terminate a temporary program, but do not purge it from the system.

```
S=07 COMMAND ?OF,PROG,1
```

PROG can later be run without having to re-load it on to the system.

3. Terminate a program after it finishes I/O processing.

```
S=07 COMMAND ?OF,PROG
    or
S=07 COMMAND ?OF,PROG,0
```

COMMENTS:

For options 1 and 8 of the OF command, the message "program ABORTED" will be displayed for programs (but not segments) after the command is executed. If the program is I/O suspended, a system-generated request to clear the device is issued to the driver and the program is immediately terminated.

For programs with segments, the OF,program,8 command must be used on the segments as well as the main to remove them from the system.

OF,program,8 will not remove permanently loaded programs, since their ID segments on the disc are not altered by this request. A permanently loaded program is defined as a program loaded during generation, or on-line with the LOADR as permanent. For temporary programs loaded on-line, the ID segment is blanked to make it available for use by another program loaded with the LOADR. A permanently loaded disc resident program may only be removed permanently with the LOADR as described in Chapter 6.

If the program is I/O suspended, a system generated clear request is issued to the driver. The OF,program,8 command must then be entered a second time to permanently remove the program from the system.

The current session program is determined by finding the last son of the session progenitor (FMGxx). If a son can't be found, the break flag is set in FMGxx's ID segment.

ON (Schedule a Program)

```
+-------+
| LEVEL |
|  50   |
+-------+
```

Schedules a program for execution. Up to five parameters and the command string may be passed to the program.

```
+-------------------------------------------------------------------+
|                                                                   |
| ON,program [,NOW] [,p1 [,... [,p5 ]]]]]     <---Passes command    |
|               -                                 string.          |
|       or                                                          |
|                                                                   |
| ONIH,program [,NOW] [,p1 [,... [,p5 ]]]]]   <---Inhibits passing  |
|                -                                of command string.|
|                                                                   |
|-------------------------------------------------------------------|
|                                                                   |
| program                                                           |
| Name of program to be scheduled.                                  |
|                                                                   |
| NOW                                                               |
| Schedules a program immediately that is normally scheduled by the |
| system clock (see IT command). If program is placed in the Time   |
| List (in conjunction with the IT command), but not scheduled for  |
| immediate execution, this parameter and its preceding comma are   |
| omitted.                                                          |
|                                                                   |
| p1 ... p5                                                         |
| Parameters passed to the program when it is scheduled.            |
|                                                                   |
+-------------------------------------------------------------------+
```

EXAMPLES:

1. Schedule program PROG to run every five minutes starting from now.

```
S=07 COMMAND ?IT,PROG,3,5
S=07 COMMAND ?ON,PROG,NOW
```

2. Schedule program PROG to run every hour starting at midnight. Pass the ASCII characters AA, BB, CC, DD and EE to the program.

```
S=07 COMMAND ?IT,PROG,4,1
S=07 COMMAND ?ON,PROG,AA,BB,CC,DD,EE
```

Program PROG can pick-up the parameters by making a call to the subroutine RMPAR. See the DOS/RTE Relocatable Library Reference Manual for a description of RMPAR.

3. Schedule program PROG to run every hour starting at midnight. Pass the ASCII string MESSAGESTRING to the program.

```
S=07 COMMAND ?IT,PROG,4,1
S=07 COMMAND ?ON,PROG,MESSAGESTRING
```

Program PROG can pick-up the ASCII string by making a call to the subroutine GETST. See the RTE-IVB Programmer's Reference Manual for a description of GETST.

COMMENTS:

Parameters p1 through p5, which are retrieved by an immediate call to RMPAR, are stored in XTEMP words 1 through 5 in the program's ID segment (see Appendix C for the format of the ID segment). Note that when using Session Monitor, any parameters not entered as part of the ON command will be returned as zeroes by a call to RMPAR. See the DOS/RTE Relocatable Library Reference Manual for a description of RMPAR. Under MTM if no optional parameters are passed with the ON command, the terminal LU will be passed to the first parameter by RMPAR. If a blank is entered before the ON command the terminal LU is not passed by RMPAR.

Parameters p1 through p5 can be entered in ASCII or numeric form. Octal numbers are designated by the "B" suffix and negative numbers by a leading minus sign. For example:

```
ON,program,FI,LE,31061B
```

Note that only two ASCII characters per parameter will be returned by a RMPAR subroutine call; if only one is given, the second character is passed as a blank (blank=40B). If the first parameter is ASCII "NO" then it must be repeated (the system interprets it as "NOW" in the ON command). For example:

```
ON,program,NO,NO,FI,3,4,5
```

is interpreted as

```
NO
FI
3
4
5
```

The program can recover the ASCII command string (up to 80 characters typed after the prompt) by using the String Passage EXEC call (see the RTE-IVB Programmer's Reference Manual for a description of this EXEC call). The ONIH command inhibits the passage of the command string. The string, if passed, is stored in a buffer in System Available Memory (SAM).

If the resolution code in the ID segment of the program is not zero, RTE places the program in the time list for execution at specified times (unless NOW appears; in which case, the program is scheduled immediately.). The resolution may be non-zero as a result of:

a. Generation

   1. With a resolution code in the NAM record
   2. Entry of a resolution code during parameter input phase.

b. The IT command.

c. Scheduling the program with absolute start time or offset by some program in the system (see EXEC calls in the RTE-IVB Programmer's Reference Manual).

Note that if there is no partition large enough to run the program, or if the program is assigned to a partition that is too small or does not exist, the error message 'SIZE ERROR' will be reported. A condition under which the error message could be output when attempting to run might be:

```
:SP,program
```

Reboot and reconfigure memory to remove partitions large enough for this program.

```
:RU,program
```

Note that a session user may not run a program while it is time scheduled in another session. Only that session which has it time scheduled may run the program. See the Time List description in this Chapter for more details.

PR (Change Program Priority)

```
+-------+
| LEVEL |
|  50   |
+-------+
```

Changes the priority of a program.

```
+-----------------------------------------------------------------+
|                                                                 |
| PR,program,new priority                                         |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| program                                                         |
| Name of program whose priority is to be changed.                |
|                                                                 |
| new priority                                                    |
| New priority number to be assigned to program; 1 is the highest |
| priority and 32767 is the lowest.                               |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

```
  S=07 COMMAND ?PR,PROGA,40
```

COMMENTS:

When the system is restarted from the disc, the priority of "program" resets to the value set by the generator or LOADR.

QU (Timeslice Quantum)

LEVEL
10/60

Allows a user with a command capability level of 10 or greater to examine the current system timeslice quantum and fence. A user with the capability level of 60 may change the timeslice parameters with this command.

```
QU [,quantum [,limit ]]
```

quantum
New system slice quantum; value must lie between 0 and 32767 milliseconds.

limit
Priority level at which timeslicing begins; default is 50; all programs of equal or lower priority (higher priority number) will be timesliced.

EXAMPLES:

1. Examine current system timeslice quantum and limit.

```
S=07 COMMAND ?QU
Q=1500 P=50
```

```
     |     |--Program priority level at which timeslicing starts.
     |--System timeslice quantum (1.5 seconds).
```

2. Change system timeslice quantum to 2.0 seconds and timeslice limit to a priority level of 100.

```
S=07 COMMAND ?QU,2000,100
```

COMMENTS:

A program's slice quantum is a function of the program's priority and the system slice quantum. The system slice quantum is the minimum slice allowed. A multiplier is formed from the program priority and is included in the following equation:

```
Program Quantum = System Quantum * ( 1 + Priority Multiplier)
```

The lower the priority (higher number), the larger the priority multiplier. This provides larger quantums for processes which execute infrequently.

The QU command allows high level users to increase or decrease the system slice quantum and/or the priority at which programs are timesliced. This command also allows the user to turn off timeslicing (QU,0 or QU,,32767).

For more details in selecting an efficient system timeslice quantum and limit, see the RTE-IVB System Manager's Manual.

RS (Restart Session Copy of FMGR)

```
+-------+
| LEVEL |
|  10   |
+-------+
```

Aborts and reschedules a session's copy of FMGR.

```
+-----------------------------------------------------------------+
|                                                                 |
| RS                                                              |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| No parameters required.                                         |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

1. While a user is listing an ASCII file on the line printer, the printer goes down causing the user's copy of FMGR to be suspended.

```
:LL,6            <----File Manager command to change list device to
                       line printer (LU 6).

:LI,FILEA        <----File Manager command to list file FILEA on the
                       list device.

IONR L*  6 E 6 S 0 ***   <---System error message indicating that
                             LU 6 is down (I/O not ready).

S=07 COMMAND ?RS         <---Abort and reschedule session's copy of
                             FMGR.
```

2. A user offs his copy of FMGR and then wants it back (for instance, so that he can log-off).

```
:OF,FMG07        <----File Manager command to off session 7's copy of
                       FMGR.

S=07 COMMAND ?RS         <---Reschedules FMG07.
```

COMMENTS:

This command is especially useful for situations such as in the first example above.

RT (Release Disc Tracks)

```
+-------+
| LEVEL |
|  30   |
+-------+
```

Releases all disc tracks assigned to a program.

```
+-----------------------------------------------------------------+
|                                                                 |
| RT,program                                                      |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| program                                                         |
| Name of the program whose assigned disc tracks are to be released. |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

```
  S=07 COMMAND ?RT,PROGA
```

COMMENTS:

A program can request disc tracks through either an EXEC 4 or EXEC 15 call. For more details on EXEC calls see the RTE-IVB Programmer's Reference Manual.

The RT command is illegal if the named program is not dormant. If the program is dormant, all tracks assigned to the program are released.

Any tracks released as a result of this command cause all programs in disc track allocation suspension to be rescheduled. More information on disc tracks may be obtained from the utility program LGTAT, described in Chapter 6 of this manual.

RU (Run a Program)

```
+-------+
| LEVEL |
|  30   |
+-------+
```

Immediately schedules a program for execution. If the program is currently in the Time List, its entry is not affected. Up to five parameters and the command string may be passed to the program.

```
+-----------------------------------------------------------------+
|                                                                 |
| RU,program [,p1 [,... [,p5 ]]]]]        <---Passes command string. |
|                                                                 |
|     or                                                          |
|                                                                 |
| RUIH,program [,p1 [,... [,p5 ]]]]]      <---Inhibits passing of |
|                                             command string.     |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| program                                                         |
| Name of program to be run.                                      |
|                                                                 |
| p1 ... p5                                                       |
| Parameters passed to the program when it is scheduled.          |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

1. Immediately schedule program PROG passing it the ASCII characters AA, BB, CC, DD, and EE.

```
S=07 COMMAND ?RU,PROG,AA,BB,CC,DD,EE
```

Program PROG can pick-up the parameters by making a call to the subroutine RMPAR. See the RTE-IVB DOS/RTE Relocatable Library Reference Manual for a description of RMPAR.

2. Immediately schedule program PROG passing it the ASCII string MESSAGESTRING.

```
S=07 COMMAND ?RU,PROG,MESSAGESTRING
```

Program PROG can pick-up the ASCII string by making a call to the subroutine GETST. See the RTE-IVB Programmer's Reference Manual for a description of GETST.

COMMENTS:

The parameters p1 through p5, which are retrieved by am immediate call to RMPAR, are stored in XTEMP words 1 through 5 in the program's ID segment. (see Appendix C for the format of the ID segment). If no optional parameters are passed with the RU command, the terminal LU will be passed to the program as the first parameter to be picked up by RMPAR. If a blank is entered before the RU command, the terminal LU is not passed (zero is passed instead). The remaining four parameters retrieved by RMPAR are zero.

Parameters p1 through p5 can be entered in ASCII or numeric form. Octal numbers are designated by the "B" suffix and negative numbers by a leading minus sign. For example:

```
RU,program,FI,LE,31061B
```

Note that only two ASCII characters per parameter will be returned by a RMPAR subroutine call; if only one is given, the second character is passed as a blank (blank=40B). If the first parameter is ASCII "NO" then it must be repeated (this is because the RU command is related to the ON,program,NOW command; see ON command for details).

The program can recover the ASCII command string (up to 80 characters typed after the prompt) by using the String Passage EXEC call (see the RTE-IVB Programmer's Reference Manual for a description of this call). The RUIH command inhibits the passage of the command string. If there are no characters past "program", the command string is not transmitted. The string, if passed, is stored in a buffer in System Available Memory (SAM).

The RU command is used when the operator desires to run a program without affecting its entry in the Time List. It is similar to the File Manager RU command except that it does not do automatic program renaming (i.e. the break-mode RU command actually runs "program" not a copy of "program").

Note that a session user may not run a program while it is time scheduled in another session. Only that session which has the program time scheduled may run the program. See the Time List description in this Chapter for more details.

SL (Display Session LU Information)

```
+-------+
| LEVEL |
|  10   |
+-------+
```

Displays the corresponding system logical unit number, equipment table (EQT) entry number and subchannel number for either a specified session logical unit number or all session logical units in a user's session switch table (SST).

```
+-----------------------------------------------------------------+
|                                                                 |
| SL [,lu]                                                        |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| lu                                                              |
| Session logical unit for which linkage information is desired;  |
| default is to list information for all session logical units in |
| caller's session switch table.                                  |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

```
  S=07 COMMAND ?SL,14      <---Display linkage information for session
                                logical unit 14.
   SLU  14=LU  # 14 = E 1 S 5 D
    ^     ^          ^   ^   ^--device status.
    |     |          |   |--subchannel number.
    |     |          |--EQT number.
    |     |--System logical unit number.
    |--Session logical unit number.

  S=07 COMMAND ?SL         <---Display linkage information for all
                                session LUs in user's SST.
   SLU   1=LU  #  1 = E 7
   SLU   2=LU  #  2 = E 1
   SLU   3=LU  #  3 = E 1 S 6
   SLU   4=LU  # 29 = E 7 S 1
   SLU   5=LU  # 30 = E 7 S 2
   SLU   6=LU  #  6 = E 6
   SLU   7=LU  # 10 = E10
   SLU   8=LU  #  8 = E 8
   SLU  14=LU  # 14 = E 1 S 5 D
```

COMMENTS:

The session LU information is always displayed on the user's terminal. Use of the File Manager LL command will not switch the output to another device such as the line printer.

SS (Operator Suspend a Program)

LEVEL
30/60

Allows user to suspend a non-dormant program. Users with a command capability level of 30 or higher can suspend any program which was scheduled within their session; users with a capability level of 60 can suspend any program in the system.

```
SS[,program]
```

program
Name of the program to be suspended; default is the current session program.

EXAMPLES:

1. User A schedules a program, PROG, to execute. While the program is executing, user A enters break-mode and suspends the program.

```
S=07 COMMAND ?RU,PROG
S=07 COMMAND ?SS,PROG
```

To reschedule the program, user A would use the GO command. To abort the program, user A would use the OF command.

2. User B schedules a program, PROGB, to execute. While it is executing, user A who has a command capability level of 60 enters break-mode and suspends PROGB.

```
S=09 COMMAND ?RU,PROGB  <---User B runs PROGB from his session.
S=07 COMMAND ?SS,PROGB  <---User A suspends PROGB from his session.
```

COMMENTS:

The SS command places the program in the operator suspended list immediately if the program is executing or scheduled. The request is illegal if the program is dormant. If the program is suspended for I/O, memory, disc track allocation, or is in the time list, RTE waits until the current state is ended and then operator-suspends the program.

The SS command is similar to the Program Suspend EXEC call (see the RTE-IVB Programmer's Reference Manual for a description).

The current session program is the last son of the session progenitor (FMGxx) or FMGxx itself.

ST (Status)

```
+-------+
| LEVEL |
|  10   |
+-------+
```

Depending upon format, can request the status (priority, current list, time values) of a named program, determine the name and partition number of the program currently executing, or print the name of the program occupying a specified partition.

ST,program	<---Program status.
ST,0	<---Name of currently executing program and its partition number; if no program is currently executing, 0 is printed.
ST,partition#	<---Name of program in specified partition; if empty, 0 is printed.

program
Name of the program whose status is to be printed.

partition#
Partition number defined at system generation; can be from 1 to 64 depending upon how may partitions are defined at system generation. If an undefined partition number is specified, the error NO SUCH PROG is printed. If partition is empty, 0 is printed.

EXAMPLES:

1. Check program status of WHZAT which was placed in the time list and scheduled to run every hour with first execution scheduled for 1:00 p.m. (See IT command for placing program into time list.)

```
S=07 COMMAND ?ST,WHZAT
    41 0 4  1  1 0 0 0 T
     ^ ^ \-v-/ \--v--/ ^
     | |   |      |    |--Program is in time list; blank if not.
     | |   |      |--Next scheduled execution time (hr,min,sec,ms).
     | |   |--Time interval.
     | |--Current state of the program (0=dormant).
     |--Program priority.
```

2. Find name and partition number of currently executing program.

```
S=07 COMMAND ?ST,0
R$PN$  2
  |     |--Partition number in which program is currently residing.
  |--Currently executing program.
```

3. Find name of program in partition number 3.

```
S=07 COMMAND ?ST,3
LGOFF
  |--Name of program currently residing in partition 3.
```

COMMENTS:

The status of a program is printed on one line in a fixed format:

```
   pr s res mpt hr min sec ms T
```

where:

pr is the priority (a decimal value from 1 to 32767).

s is the current state of the program:

```
0 = dormant
1 = scheduled
2 = I/O suspended
3 = general wait
4 = unavailable memory suspend
5 = disc allocation suspend
6 = operator suspend or programmed suspend
9 = background segment
```

res, mpt, hr, min, sec, and ms are all zero (0) unless the program is scheduled by the clock (see the IT command in this Chapter for the meaning of these terms).

The letter "T" appears when the program is currently in the time list as a result of the IT and ON commands.

The ST,0 command is only useful from the system console since from break-mode the currently executing program will always be RPN.

SZ (Program Size)

```
+-------+
| LEVEL |
|  30   |
+-------+
```

Depending on format, either prints current program size information or allows changing of program size requirements.

1. Print program size information:

   SZ,program

2. Change program size requirements:

   SZ,program,P1 <---For non-EMA programs.

   SZ,program,P1,P2 <---For EMA programs.

program
Name of program for which size information is desired or to be changed.

P1
New required program size in pages for non-EMA programs. For EMA programs, P1 is the new EMA size.

P2
New MSEG size for the EMA program specified.

EXAMPLES:

1. Print program size information for EMA program PROGE.

```
S=07 COMMAND ?SZ,PROGE
65211  200  180   2
  ^    ^    ^    ^
  |    |    |    |--Program's MSEG size (pages).
  |    |    |--Program's EMA size (pages).
  |    |--Minimum required partition size for program (pages).
  |--Address of last word plus 1 of the user's program.
```

2. Change required program size for program PROGA, which is a non-EMA program, to 12 pages.

```
S=07 COMMAND ?SZ,PROGA,12
```

3. Change required size for EMA program PROGE to 190 pages and MSEG size to 1 page.

```
S=07 COMMAND ?SZ,PROGE,190,1
```

COMMENTS:

The output for the program size information as printed out by SZ,program will be formatted as:

```
AAAAA BB CCCC DD
```

where:

AAAAA = the address of the last word plus 1 of the user's program. If the program is segmented, AAAAA is the address of the last word plus 1 of the largest segment.

BB = minimum required partition size of the program. If the program is of EMA type, BB equals the program code size plus its EMA size.

CCCC = the program's EMA size. Printed for EMA programs only.

DD = the program's MSEG size. This will only be printed if the program is of EMA type.

The SZ,program,P1 and SZ,program,P1,p2 forms allow the user to increase the page requirements of a program. Certain programs such as compilers, assemblers, loader and generator use memory after the end of the program for symbol table or data space. The SZ command modifies the size of the additional memory used by the program.

Before program size requirements can be changed, the program must be dormant and not currently resident in a partition (i.e., it must not have terminated with save-resources or in a serially reusable condition), and there must be at least one partition large enough to run the program at its new size.

The following conditions will be flagged as errors and the error 'SIZE ERROR' reported:

FOR NON-EMA PROGRAMS

1. An attempt to make P1 larger than 32K word program address space.
2. An attempt to make P1 larger than any currently existing partitions.
3. If the program is assigned, an attempt to make P1 larger than the partition size.
4. An attempt to make P1 smaller than the actual code of the program.

FOR EMA PROGRAMS

1. An attempt to set P1 such that the program size plus the EMA size is larger than the largest partition in the system.
2. If the program is assigned to a partition, an attempt to set P1 such that the program size plus the EMA size is larger than that partition.
3. An attempt to set P1 less than 1.
4. An attempt to set P2 such that the program size plus P2 exceeds maximum program address space.
5. An attempt to set P2 less than 1.

EMA size changes are only allowed for those programs where no EMA size was specified within the program itself; that is, the default was taken. An attempt to increase or decrease the EMA size in a program where the EMA size was specified within the program causes a 'SIZE ERROR' message to be issued. MSEG changes may be made for any EMA type program. All FTN4 programs have specified EMA sizes.

TE (Send Message to System Console)

LEVEL
10

Allows user to send a message to the system console.

```
TE,message
```

message
Message to be sent to system console; must conform to parameter syntax rules for privileged commands (refer to Chapter 3).

EXAMPLE:

```
S=07 COMMAND ?TE,MESSAGE TO CONSOLE
```

COMMENTS:

The message may consist of any printable ASCII characters. It is limited by the length and number restrictions placed on any FMGR command parameters (refer to Parameter Syntax rules, Chapter 3). Note that any commas divide the message into separate parameters. A one-parameter string could be as long as 60 characters.

TI (Print Time)

```
+-------+
| LEVEL |
|  10   |
+-------+
```

Prints the current year, day and time, as recorded in the real-time clock.

```
+-----------------------------------------------------------------+
|                                                                 |
| TI                                                              |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| No parameters required.                                         |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

```
  S=07 COMMAND ?TI
  1978 345  13  17  43
    ^   ^    ^   ^   ^
    |   |    |   |   |--Seconds.
    |   |    |   |--Minutes.
    |   |    |--Hour of day.
    |   |--Day of year.
    |--Year.
```

COMMENTS:

The TI command is similar to the Time Request EXEC call (see RTE-IVB Programmer's Reference Manual for a description of this EXEC call).

TM (Set Real-Time Clock)

```
+-------+
| LEVEL |
|  60   |
+-------+
```

Allows user to set or reset the real-time clock.

```
+-----------------------------------------------------------------+
|                                                                 |
| TM,year,day [,hr [,min [,sec ]]]                                |
|                                                                 |
|-----------------------------------------------------------------|
| year                                                            |
| Four-digit year.                                                |
|                                                                 |
| day                                                             |
| Three-digit day of the year.                                    |
|                                                                 |
| hr                                                              |
| Two-digit hour of the day; default is 0.                        |
|                                                                 |
| min                                                             |
| Two-digit minute of the hour; default is 0.                     |
|                                                                 |
| sec                                                             |
| Two-digit second of the minute; default is 0.                   |
+-----------------------------------------------------------------+
```

EXAMPLE:

```
  S=07 COMMAND ?TM,1978,150,14,50,36   <---Set time.
  S=07 COMMAND ?TI                     <---Print time.
  1978  150  14  50  36
    ^    ^   ^   ^   ^
    |    |   |   |   |--Seconds.
    |    |   |   |--Minutes.
    |    |   |--Hour.
    |    |--Day of year.
    |--Year.
```

COMMENTS:

The TM command is entered in response to the message:

```
   SET TIME
```

which is displayed when the RTE system is booted from disc.

Enter a time value ahead of real-time. When real-time equals the entered value, press the RETURN key. The system will then be synchronized with the time of day.

When resetting the time, do not enter a time value less than the Log-on Time of the user.

TO (Device Time-Out)

LEVEL
10/60

Allows users with a command capability level of 10 or more to examine the current time-out value of an I/O controller. Users with a capability level of 60 can change the time-out value.

TO,eqt	<---Print time-out value for EQT (level 10 command).
TO,eqt,numb	<---Change time-out value for EQT (level 60 command).

eqt
Equipment table (EQT) entry number of the I/O controller.

numb
Number of 10 millisecond intervals to be used as the time-out value; cannot be less than 500 (5 sec) for terminals driven by drivers DVR00 or DVR05.

EXAMPLES:

1. Examine the current time-out value for a terminal with EQT#7.

```
S=07 COMMAND ?TO,7
TO#  7=12000
     ^   ^
     |   |--Time-out value (120 sec)
     |--EQT number
```

2. Change the time-out value for the above terminal to 60 seconds.

```
S=07 COMMAND ?TO,7,6000
```

To check to see whether the time-out value has changed, the TO,eqt command must be entered to print-out the current value.

COMMENT:

The time-out value is calculated using numb time-base generator interrupts (the time-base generator interrupts once every 10 ms). For example, numb = 100 sets a time-out value of one second (100*10 ms =1 second). When the system is rebooted from the disc, time-out values set by TO are reset to the values originally set during generation.

A time-out value of zero is equivalent to not using the time-out feature for a particular controller. If a time-out parameter is not entered, its value remains zero and time-out is disabled for the controller.

When a controller times-out, a driver has the option of performing its own time-out processing or of letting the system handle it entirely.

If the time-out value is set to approximately 20 seconds or above (TO,eqt,2000), for a Session Terminal, the Session will be terminated after five time-outs. If the time-out value is set below approximately 20 seconds, the Session will not be terminated.

For more details on time-out processing, see the RTE Operating System Driver Writing Manual.

UP (Make Available)

```
+-------+
| LEVEL |
|  10   |
+-------+
```

Declares an I/O controller and all associated devices as up (i.e., available for use by the RTE system).

```
+-----------------------------------------------------------------+
|                                                                 |
| UP[,eqt]                                                        |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| eqt                                                             |
| Equipment table (EQT) entry number of the I/O controller to be  |
| re-enabled. If not specified, re-enable the device (EQT number) |
| that FMGxx or one of its sons is waiting for.                   |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

```
S=07 COMMAND ?UP,8
```

COMMENTS:

When the operator has previously set an I/O controller or device down for some reason, the condition should be corrected before using the UP command to declare the item available again. If the problem is irrecoverable, the LU command can be used to switch the Logical Unit number assignment to another device for further requests (see the LU command in this Chapter). Previous requests made to this device are switched to the new device. To prevent indefinite I/O suspension on a downed device, time-out is used. Refer to the TO command in this Chapter.

The UP command places all downed devices (LUs) and the I/O controller (EQT entry) in the available state. Any I/O operations associated with downed devices are queued on the EQT entry for processing. If a device's problem has not been corrected, it will be reset down and an error message will be printed:

```
IONR L #lu E #eqt S #sub xxx
        ^     ^      ^   ^
        |     |      |   |--Device Status
        |     |      |--Subchannel
        |     |--EQT number
        |--LU number
```

UR (Release Reserved Partition)

LEVEL
50

Releases a partition previously reserved during generation or reconfiguration.

```
+-----------------------------------------------------------------+
|                                                                 |
| UR,partition                                                    |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| partition                                                       |
| The number of the partition to be released (1 to 64, depending  |
| upon how many memory partitions were defined in system generation).|
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

```
S=07 COMMAND ?UR,5
```

COMMENTS:

Once the command is entered, any program that fits into the partition may run in it. Note that although partitions may be released on-line, they may not be reserved on-line, since such action could prevent a currently swapped-out program from regaining use of its system-assigned partition when it was again scheduled.

WH (Run WHZAT Program)

LEVEL
10

Schedules the WHZAT program to display the current system status (see Chapter 6 for a detailed description of the WHZAT program).

WH,lu [,option]

lu
The logical unit number of the device on which the information is to be displayed (default value is the lu of the terminal from which command is input).

option
Indicates type of status to be displayed. Default is to display only those programs associated with the user's session.

option value	description
AL	Status of all scheduled and suspended programs.
SM	Same as AL except that all state-3 programs not having "father-son" relationship will not be displayed.
PA	Status of all partitions being used.

EXAMPLES:

```
S=07 COMMAND ?WH          <---displays status of scheduled and
                              suspended programs on input terminal.

S=07 COMMAND ?WH,,PA      <---displays status, on input terminal, of
                              all partitions being used.
```

COMMENTS:

For details on the system status program (WHZAT), see Chapter 6.

Chapter 5

INTERACTIVE EDITOR COMMANDS

INTRODUCTION

The RTE Interactive Editor (EDITR) is commonly used for:

* Creation of new programs or data files in ASCII code.
* Modification of new or existing ASCII files.
* Merging several ASCII files to each other to form a single file.

EDITR operates in either interactive or in batch mode. When used interactively, EDITR accepts operator commands from a keyboard device. When used with batch jobs, EDITR commands are included as part of the job command file. Refer to the RTE-IVB Batch and Spooling Reference Manual for details on batch job processing. The EDITR in Batch Environment section describes the use of EDITR commands in a batch environment.

EDITR WORK AREAS

EDITR references a source file and an output file and uses two temporary work areas on disc. The source file is read into a work area on disc called the source work area. Edited text is passed to another work area, also on disc, called the destination work area. An editing pass is completed when EDITR has read and passed all of the text lines in the source work area. Before another editing pass can be made, the source work area is replaced by the old destination work area, and a new destination work area is designated.

If the user backs up to edit a line preceding the current line, the editing pass is completed. The remaining information in the source work area is passed to the destination work area and the "destination work area" now becomes the "source work area".

When editing is complete and EDITR is terminated with one of the terminate commands, the remaining data in the source work area is passed to the destination work area, and the destination work area is written to an output file. The relationship between these files and work areas is shown in Figure 5-1.

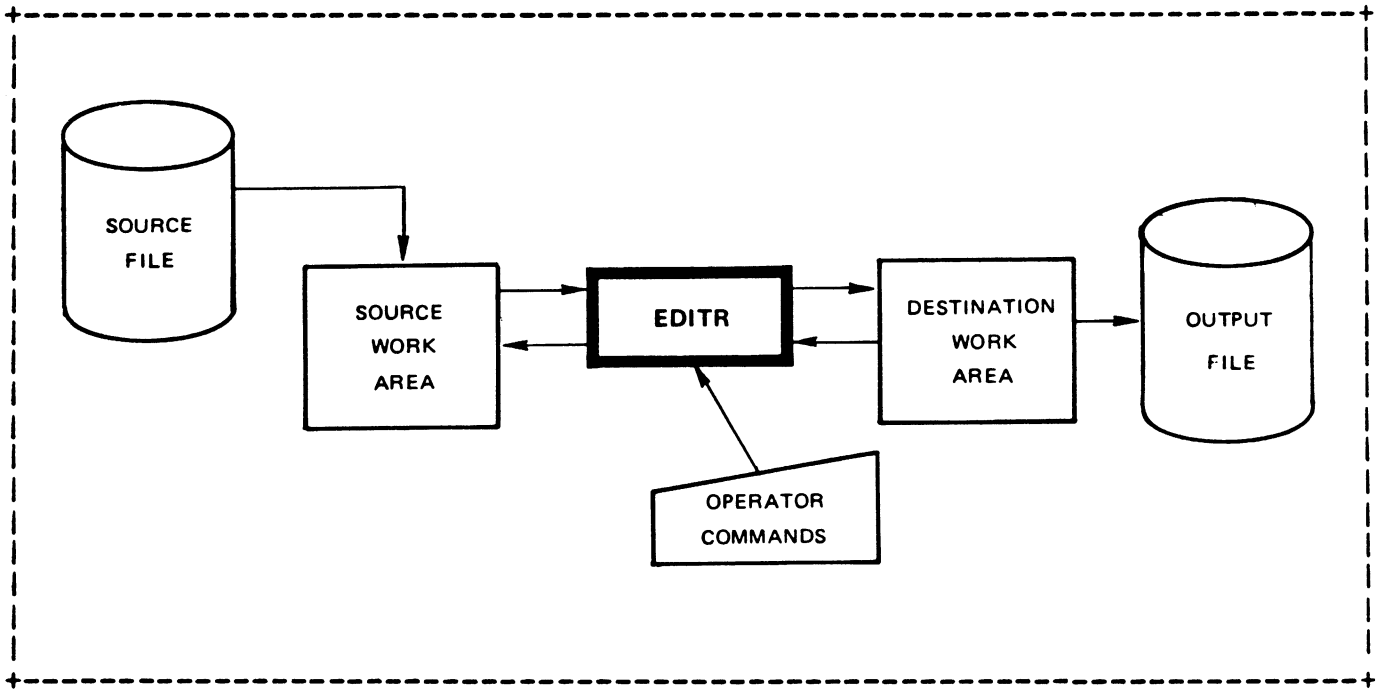


Figure 5-1. File/Work Area Relationship

PENDING LINE

When EDITR is run, and the source file the user names at the beginning of the editing session is read into the source work area, the first line is displayed. This displayed line is the current line available for editing and is called the "pending line". This line remains the pending line until the user requests a new pending line with an EDITR command. In this way the user can continue to re-edit the same line until he is satisfied.

When the user requests a particular line of text, EDITR searches through the source work area until the requested line is encountered. That line then becomes the new "pending line" and is displayed on the user's terminal. EDITR maintains a "pending line" pointer into the source work area.

When the new pending line is displayed, the old pending line, and all other lines passed over by the pointer in the search for the new pending line, are usually written to the destination work area (see the Jump command for an exception), and are no longer accessible until the "destination work area" becomes the "source work area". In other words, the user cannot go from line 0028 back to line 0024 without the source work area being replaced by the destination work area. Generally the user does not have to be concerned with this exchange because it is done automatically. However, the user does need to remember that once the "source work area" is replaced, the previous "source work area" is totally gone along with any lines deleted or changed in the last pass of EDITR. The user should remember that if he made any insertions or deletions, these lines are no longer accessible by their original line numbers. For example, if line 3 is deleted in one editing pass, the original line 4 will be the new line 3. EDITR permits lines of text to be accessed without using line numbers.

KEYBOARD CONVENTIONS

Keyboard commands are used to direct EDITR to do replacements, insertions, deletions, searches and exchanges of text. These functions can be performed on characters within a line; they can be used to manipulate one entire line; or they can be used on groups of lines.

To use EDITR efficiently, the user must know the command syntax and keyboard conventions it expects. The command syntax used in this manual is summarized in Table 5-1, for keyboard conventions check the appropriate manual for the terminal being used. If a Multipoint terminal is being used, also read the Multipoint User's Guide.

Conventions Used in Examples:

{ }	Space entered at the keyboard.
CNTL/X	Indicates a nonprinting control character (in this case, X) entered at the keyboard.

Table 5-1. Command Syntax

CONVENTION	MEANING	COMMAND	EXAMPLE	COMMENTS
UPPER CASE LETTERS	Literals that must be specified as shown	F	F	No Parameters
lower case letters	Variables to be replaced by values as defined in text	Wa,b	W7,9	Constants 7 and 9 replace variables a and b
CNTL/	Nonprinting control characters entered by pressing the CNTL key and another key simultaneously	CNTL/key	CNTL/C	
[ ]	Bracketed parameters are optional; if omitted default values are supplied	filename[:sc[:crn]]	example: MYFILE:AA:2	

DISPLAY FORMATS

The pending line displayed and listings output by EDITR are always preceded by two blanks. This convention allows room for the EDITR prompt and a single character command, and results in the new text being automatically aligned with that displayed by EDITR. Error messages, in contrast, always begin in column 1. This is so that the difference between an EDITR message and a line of text can be easily seen. For example:

```
   EOF     EDITR text
EOF        EDITR message
```

CALLING EDITR

The user can call EDITR with either the File Manager or System RU commands. The File Manager (FMGR) RU command is described in Chapter 3 and the System RU command is described in Chapter 4.

The two optional parameters passed to EDITR when it is scheduled are:

lu - the logical unit number of the device to be used for command input. The default for this parameter is the user's terminal.

line length - the maximum output line length, in characters; where the default and maximum size is 150 characters. Any line longer than 150 characters, or the specified length, will be truncated.

For example, to schedule EDITR from the File Manager with commands input from the user's terminal and lines limited to 72 characters, the command would be:

```
:RU,EDITR,,72
```

LOGICAL UNIT

The logical unit may specify any device on which the user can enter EDITR commands. Usually it is an interactive keyboard device. If the user is editing in batch mode as described in the EDITR IN BATCH ENVIRONMENT section, this device could be a read-only device such as a card reader. If the user is using batch with spooling, this device could be LU 5 (command input), or an LU corresponding to a spool file containing the commands.

LINE LENGTH

The user can specify a line length which is compatible with the device to which his output file is going to. Otherwise, long lines may be truncated. Table 5-2 shows the output commonly used with EDITR and the number of characters per line each supports. The user should remember that because of the two space convention used in displays, printed lines may be 74 characters.

Table 5-2. Maximum Line Lengths for Common Devices

DEVICE	LINE LENGTHS	COMMENTS
Punched Cards	80 Characters	
CRT	72 - 80 Characters	
Magnetic Tape	150 Characters	Default maximum length supported by EDITR.
TTY Device	72 Characters	Longer lines will encounter printing problems.
Paper Tape	150 Characters	Default maximum length supported by EDITR.
Disc	150 Characters	Default maximum length supported by EDITR.
Line Printer	80 - 132 Characters	Number of characters varies with printer model. Consult appropriate manual for your printer.

EDITR PROMPT CHARACTER

When EDITR is used in the interactive mode, it prompts for input with a slash (/). The X command (refer to X (Change EDITR Prompt Character) section) can be used to change this prompt character to any other character. Thereafter, the specified character is output as the EDITR prompt.

FILE DEFINITION

The NAMR used by the EDITR is a simplified version of the File Manager NAMR (refer to Chapter 3 for description). The EDITR version of NAMR is defined to be a file name followed by two subparameters. The subparameters may be omitted from the end of the list. If an embedded subparameter is omitted, its position must be indicated by the colon (:). The NAMR format is:

```
filename [:security code [:cartridge reference number ]]
```

See NAMR description in Chapter 3 for explanation of parameters.

EDIT EXISTING FILE

When EDITR begins execution, it requests information about the file to be edited and prompts the user:

```
SOURCE FILE?
/
```

At this point, there are four legal responses the user can make, each of which must be followed by a carriage return:

1. 0 (zero)
2. : (colon)
3. filename
4. { } (blank)

If "0" is specified, the user will automatically begin working with an empty file. The EDITR commands can then be used to enter and edit lines of text. When creating a file, this is the recommended response. See the CREATE FILE WITH EDITR section for an illustrative example.

If ":" is specified, the EDITR is immediately aborted and control returns to the program from which the EDITR was scheduled.

If a file name is specified, the contents of the file will be copied into the EDITR's source work area. The number of characters per line in the named file cannot be greater than the current maximum line length.

The file may be a type 0 file to read source information directly from a peripheral device. The file name may be specified with or without a security code and a cartridge reference number. For example, to specify a file without a security code, but with a cartridge reference number:

```
/FILE1::27
```

Refer to the discussion on type 0 files and the NAMR parameter in Chapter 3 for more details.

If a blank is specified, the current Logical Source (LS) area of the disc is copied into the EDITR's source work area. The EDITR commands may now be used to edit the source text. See Appendix D for a discussion of the Logical Source area.

CREATE FILE WITH EDITR

To create a file with the EDITR, enter a 0 (zero) in response to the EDITR prompt "SOURCE FILE?". EOF is printed and the following lines can be entered as illustrated:

```
:RU,EDITR
SOURCE FILE?
/0  <------------Enter 0 to put empty file into EDITR's
                 source work area.
EOF
/ NEW FILE  <----Enter space to add line of text following
                 pending line, then enter line of text to
                 be added.
/ LINE TWO
/ LINE THREE
/ECFILE1 <-------End EDITR and create FMP file named FILE1.
```

The above command sequence will enter the three lines of text shown into the newly created file named FILE1.

EDITR TERMINATION

The EDITR terminate commands (EC and ER) usually assign the final version of text in the destination work area to a File Manager type 4 file. The destination work area can also be output directly to a device through a type 0 file (see Chapter 3 for discussion of type 0 files), or to the RTE system Logical Source (LS) tracks (see Appendix D for a discussion of Logical Source tracks).

LOADING EDITR ON-LINE

The EDITR program can be loaded on-line using the RTE Relocating Loader (LOADR). See the RTE System Manager's Manual for details on the recommended size and type of program to be specified. Chapter 6 describes the use of the Relocating Loader.

EDITR COMMANDS

All of the EDITR commands are summarized in Table 5-3. The commands are broken down into functional groups to facilitate referencing.

The CONTROL COMMANDS provide additional EDITR features beyond the normal text manipulation functions.

The DISPLAY COMMANDS cause the contents of the source work area or information about the contents to be displayed. The display normally occurs on the user's terminal, but some commands allow displays on the line printer and other logical units.

The LINE EDIT COMMANDS allow the user to manipulate one entire line of text at a time. In addition to these EDITR commands, certain keys on the keyboard such as DEL can also be used.

The DEL key is used to delete a line if pressed before the RETURN key enters the line into the destination work area. It prints a back slash (\), and causes a line feed and a carriage return to the start of the next line (prompt is not repeated) where the correct line can be entered.

The CHARACTER EDIT COMMANDS provide a means to change the contents and modify the structure of the current line of text. Four commands allow the replacement, insertion, and deletion of characters. Each of these commands uses nonprinting control characters so that alignment is maintained in the text. Each must be used with an initial "P", "C", or "O" command.

The current delimiter is used as a "place holder" to preserve existing text in the pending line. The P, C and O commands are not themselves character edit functions. They are used to determine the line disposition during the edit: P leaves the edited line as the pending line, C advances the pending line to the next line after the edit, O sends the pending line to the destination work area and then edits a copy of the line and leaves it pending.

Special considerations apply to these commands in a multipoint environment. Refer to the EDITR IN A MULTIPOINT ENVIRONMENT section for a complete explanation on how to use the EDITR in that environment.

The PATTERN EDITS involve multiple lines of text from the source work area. There are two kinds of pattern edits: searches and exchanges.

A search matches a character string called a "find field" with a corresponding string in the source work area. There are four search commands (B, D, F, and J) which differ by where they begin the search and the length of the data block which they search.

Exchanges consist of two sets of character strings entered at the keyboard. Whenever the first string is encountered in the source work area, it is replaced by the second string. The commands used to perform exchanges are: G, Y, X, Z, V, and U.

The TERMINATE COMMANDS are used to end EDITR operations. With the exception of the Abort command, they perform any enabled exchanges, cause data remaining in the source work area to go to the destination work area, and cause the edited destination work area to be written to the output file.

All of the normal EDITR termination commands begin with E. Once E is entered EDITR begins its termination process. Only the second letter of a normal terminate command or DEL can be entered at this time.

Output to a device of a type 0 file can only be done using the ER terminator since EDITR will not create a type 0 file using the EC. The ELR terminator (see Appendix D) can also be used to output to a device of a type 0 file, but it requires use of the Logical Source tracks.

Table 5-3. EDITR Command Summary

FUNCTION	COMMAND	DESCRIPTION	Page No.
CONTROL	X	Change EDITR Prompt Character	5-13
	CNTL/G	Bell Control	5-14
	T	Set Tab Stops	5-15
	W	Set Window	5-17
	#	Sequence Numbers	5-18
	=	Set Line Length	5-20
	K	Kill Trailing Blanks	5-21
	Mnamr	Merge Source File Following Pending Line	5-23
DISPLAY	Ln	Display a Number of Lines	5-25
	n	Display Specified Line	5-27
	/ or +	Space Down a Number of Lines and Display	5-28
	N	Display Pending Line Number	5-30
	ND	Display the Number of Lines in the Destination Work Area	5-31
	H	Display the Number of Characters in the Pending Line	5-33
	HL	Display Header	5-35
	^	Back up in Destination Work Area	5-36
	S	Display Approximate Number of Words in the Destination Work Area	5-37
	P	Display Pending Line	5-38
LINE EDITS	P	Edit and Display Pending Line	5-38
	C	Edit Pending Line and Advance Line	5-39
	O	Duplicate Pending Line and Edit	5-40
	R	Replace Pending Line with Text	5-42
	I	Insert Text before Pending Line	5-43
	{ }	Insert Text after Pending Line	5-44
	-	Delete a Number of Lines	5-45

Table 5-3. EDITR Command Summary (Continued)

FUNCTION	COMMAND	DESCRIPTION	Page No.
CHARACTER EDITS	CNTL/R	Replace Characters	5-46
	CNTL/I	Insert Characters	5-47
	CNTL/S	Insert Characters	5-47
	CNTL/C	Cancel Characters	5-49
	CNTL/T	Truncate Remainder of Pending Line	5-50
PATTERN EDITS	Search Commands		
	;	Find Tab Field	5-51
	esc	Find Field of Indefinite Length	5-51
	/	Find Field within Window	5-51
	CNTL@	Find a Zero Length Line within this Field	5-51
	B	Find a Line with Find Field SOF to EOF	5-51
	F	Find a Line with Find Field Pending Line to EOF	5-54
	D	Delete Lines to Find Field or EOF	5-57
	J	Jump to Find Field and Make It New Pending Line	5-62
	Exchange Commands		
	G	Character Replace on Pending Line	5-65
	Y	Exchange on Pending Line, Display Next Occurrence of Pattern	5-67
	X	Enable Exchange Pattern All Lines (list)	5-70
	Z	Enable Exchange Pattern for Next Edit (no list)	5-72
	V	Unconditional Exchange (list)	5-75
	U	Unconditional Exchange (no list)	5-78
TERMINATE	A	Abort EDITR	5-81
	ECnamr	Create File Manager File	5-82
	ER	Replace Old File with New File Do Not Change Name	5-83
	ERnamr	Replace Old File with New File Change Name	5-83

X (CHANGE EDITR PROMPT CHARACTER)

Changes the EDITR prompt character (default is slash) to a user defined prompt character.

```
+-----------------------------------------------------------------+
|                                                                 |
| Xx                                                              |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| x                                                               |
| New prompt character (default is slash).                        |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

```
  /X$     <-----Changes the prompt from a slash (/) to a dollar sign
                ($).
  $       <-----New EDITR prompt.
```

COMMENTS:

If the prompt character is changed, then the new prompt is required as the delimiter between the fields of an exchange command (i.e., for X, Y etc. commands) and for each character to be preserved when doing a character edit (i.e., in P, C, and O commands).

The space down command (/) is not affected by changing the prompt character.

The default for the EDITR prompt is always a slash at the start of an edit session regardless of what it might have been changed to in a prior edit session.

CNTL/G (BELL CONTROL)

Turns terminal bell on or off. When EDITR is scheduled, on a terminal with a bell, the bell is rung automatically every time a prompt is displayed.

CNTL/G
A control G is input by striking the "G" key while depressing the "CNTL" key on the terminal. It is a nonprinting character.

COMMENTS:

The bell control only pertains to the period within an edit session. The bell is always ON when the edit session starts, even if it had been turned OFF in a prior session and not turned back on.

T (SET TAB STOPS)

Changes the EDITR tab character (;) and the default tab stops (7th and 21st columns) to user defined values.

```
+-----------------------------------------------------------------+
|                                                                 |
| Tx                  <---Change tab control character, leave stops.
|                                                                 |
| Ttsl,s2,...,sl0     <---Change tab stops, leave control character.
|                                                                 |
| Txsl,s2,...,sl0     <---Change tab stops and control character.
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| x                                                               |
| New tab control character (replaces the original semicolon or   |
| current tab control character).                                 |
|                                                                 |
| t                                                               |
| Current tab control character.                                  |
|                                                                 |
| sl,s2,...,.sl0                                                  |
| New column numbers of the tab stops (replacing default values of|
| 7 and 21). The maximum number of tab stops that can be defined at|
| one time is 10.                                                 |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLES:

1.

```
/T%4,9          <---Changes the tab character to a percent sign (%)
                    and changes the tab stops to columns 4 and 9.
/ %A%B          <---Command to add a line with an "A" in column 4 and
                    a "B" in column 9.
/P              <---Command to display pending line.
     A    B     <---Displayed pending line showing "A" in column 4 and
                    "B" in column 9.
```

2.

```
/T;11,22        <---Changes tab stops to columns 11 and 22 without
                    changing tab control character from semicolon (;).
```

COMMENTS:

Tabs used beyond the highest defined stop are replaced with blanks. For instance, using the above example:

```
/ %A%B%C%D      <---Command to add line with letters A, B, C, and
                    D at tab stop locations.
/P              <---Command to display pending line.
    A   B C D   <---Displayed pending line showing "A" in column
                    4 and "B" in column 9 (the defined tab stops).
                    Since no tab stops have been defined beyond
                    column 9, "C" and "D" follow with one blank
                    space preceding each of them.
```

When EDITR is called, the tab character and tab stops are defaulted to a semicolon (;) and columns 7 and 21 even though they may have been redefined in a prior edit session and not changed back.

In a Multipoint environment, special considerations are necessary for setting tab stops. Refer to the EDITR IN A MULTIPOINT ENVIRONMENT section for details.

W (SET WINDOW)

Allows user to limit the area of each record which is searched for a character string to be found, and/or exchanged.

```
+-----------------------------------------------------------------+
|                                                                 |
| Wa,b                                                            |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| a                                                               |
| Initial column number of window (default is 1).                 |
|                                                                 |
| b                                                               |
| Final column number of window (default is 150).                 |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

```
  /W7,9     <----The window is set to include only columns 7,8 and 9.
```

COMMENTS:

The set window command is especially useful when used in conjunction with the search commands (F, B, D, and J) and the exchange commands (G, Y, X, Z, V, and U). See the X command (for exchanging string patterns) for an example of using the set window command.

When EDITR is called, the display field or "window" consists of columns 1 through 150 even if the set window command had been invoked to define another window in a prior edit session.

(SEQUENCE NUMBERS)

Allows user to place sequence numbers (in columns 76 through 80) on all lines in a file. Also, a three column identifier (in columns 73 through 75) can be specified by the command.

```
# [xxx [nl [,n2 ]]]
```

xxx
Three character identifier. It occupies columns 73-75 and must be included when specifying nl and n2 (it may contain blanks).

nl
Starting sequence number. If omitted, numbers start with 00000.

n2
Incrementing value for sequence numbers. nl is incremented by n2 for each line. If omitted, nl is incremented by 10.

EXAMPLES:

Given a file containing the following three lines:

```
THIS IS LINE ONE OF SEQUENCING EXAMPLE
THIS IS LINE TWO OF SEQUENCING EXAMPLE
THIS IS LINE THREE OF SEQUENCING EXAMPLE
```

any one of the following four commands can be given (in all the examples it is assumed that line 0001 is the pending line). The resulting edited file is also shown.

1. /# <---Sequence lines with no three character identifier and default values for nl and n2.

```
THIS IS LINE ONE OF SEQUENCING EXAMPLE                    00000
THIS IS LINE TWO OF SEQUENCING EXAMPLE                    00010
THIS IS LINE THREE OF SEQUENCING EXAMPLE                  00020
```

2. /#ABC <---Sequence lines with ABC as the three character identifier with default values for nl and n2.

```
THIS IS LINE ONE OF SEQUENCING EXAMPLE                 ABC00000
THIS IS LINE TWO OF SEQUENCING EXAMPLE                 ABC00010
THIS IS LINE THREE OF SEQUENCING EXAMPLE               ABC00020
```

3. /#ABC,1 <---Sequence lines with ABC as the three character identifier, the first line starting with 00001 and n2 defaulted to 10.

```
THIS IS LINE ONE OF SEQUENCING EXAMPLE                    ABC00001
THIS IS LINE TWO OF SEQUENCING EXAMPLE                    ABC00011
THIS IS LINE THREE OF SEQUENCING EXAMPLE                  ABC00021
```

4. /#ABC,1,1 <---Sequence lines with ABC as the three character identifier, first line starting with 00001 and subsequent lines incremented by 1.

```
THIS IS LINE ONE OF SEQUENCING EXAMPLE                    ABC00001
THIS IS LINE TWO OF SEQUENCING EXAMPLE                    ABC00002
THIS IS LINE THREE OF SEQUENCING EXAMPLE                  ABC00003
```

COMMENTS:

When listing a file on the terminal or the line printer, sequence numbers may be lost due to truncation. To delete the blanks in columns 60-70 so that the entire sequence number field can be output, these commands are entered at the terminal:

```
/W60,70       <-------Set window to delete blanks.
/Z          / <-------Exchange 10 blank spaces for no blank
                      spaces (see Z exchange command).
/3            <-------Number of lines to be changed (required
                      for Z command).
```

= (SET LINE LENGTH)

Allows user to reset the line length to another value and to truncate any characters in the line beyond the new limit.

```
+-----------------------------------------------------------------+
|                                                                 |
| =n                                                              |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| n                                                               |
| Number of columns in line length.                               |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

Given a file containing the following two lines:

```
  THIS IS LINE ONE OF THE LINE LENGTH EXAMPLE
  THIS IS LINE TWO OF THE LINE LENGTH EXAMPLE
```

The set line length command can be issued to truncate the lines to eliminate everything beyond column 8.

```
  /1   <----Makes line one pending line.
  /=8  <----Changes line length to 8 columns.
```

The resulting file will look like:

```
    THIS IS
    THIS IS
```

If a third line was input at this point,

```
  / THIS IS LINE THREE OF THE LINE LENGTH EXAMPLE
```

The line would be truncated to include only the first 8 columns

```
  /P          <---Display pending line.
    THIS IS   <---Truncated third line.
```

COMMENTS:

The line length initially defaults to 150 characters when the EDITR is turned on unless a different line length is passed through the parameter string in the RU,EDITR command (see the CALLING EDITR section).

The line length command is useful for such things as eliminating line sequence numbers.

K (KILL TRAILING BLANKS)

Deletes trailing blanks from all lines in the work area.

```
+-----------------------------------------------------------------+
|                                                                 |
| K                                                               |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| No parameters required.                                         |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

Given a file with the following three lines of text:

```
  LINE ONE OF KILL EXAMPLE                                  00000
  LINE TWO OF KILL EXAMPLE                                  00010
  LINE THREE OF KILL EXAMPLE                                00020
```

If the sequence numbers are eliminated via the set line length command,

```
  /=72    <---Columns 73 through 80 will be truncated.
```

the blanks up to column 72 will remain thus requiring each record in in the file to be 72 characters.

```
  /H      <---Display number of characters in pending line.
    72    <---Number of characters in pending line.
```

To eliminate these unnecessary blanks from the file, the kill trailing blanks command can be used.

```
  /1      <---Make line one the pending line.
  /K      <---Kill trailing blanks in the file.
```

The resulting lines of text will then be stored more efficiently by making each record (or line of text) only as long as necessary.

```
  /1      <---Make line one the pending line.
  /P      <---Display pending line.

    LINE ONE OF KILL EXAMPLE

  /H      <---Display number of characters in pending line.
    24    <---Number of characters in pending line (always even).
```

COMMENTS:

When a line of text is input through the EDITR, only the characters input prior to the carriage return are stored in a record. If the line

```
/ LINE ONE OF KILL EXAMPLE
```

had been input and immediately followed by a carriage return, there would only be 24 characters on that line. The kill trailing blanks command would not be necessary in this case.

The kill trailing blanks command always causes the source work area to be replaced by the destination work area, and always ends with an EOF message.

M (MERGE SOURCE FILE)

Allows user to merge a specified file after the pending line.

```
+-----------------------------------------------------------------+
|                                                                 |
| Mnamr                                                           |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| namr                                                            |
| filename [:security code [:cartridge reference number ]]        |
|                                                                 |
| (Refer to Chapter 3 for the description of NAMR parameter.)     |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

This example will merge two files, a source file called FILE1 and a file to be merged called FILE2.

Following is a listing of the source file, FILE1:

```
  :LI,FILE1     <-----FMGR command to list FILE1.

FILE1  T=00004 IS ON CR01000 USING 00001 BLKS R=0002

0001  LINE ONE OF SOURCE FILE.
0002  LINE TWO OF SOURCE FILE.
```

Following is a listing of the file to be merged, FILE2:

```
  :LI,FILE2     <-----FMGR command to list FILE2.

FILE2  T=00004 IS ON CR01000 USING 00001 BLKS R=0002

0001 LINE ONE OF FILE TO BE MERGED.
0002 LINE TWO OF FILE TO BE MERGED.
```

To merge FILE2 at the end of FILE1, the following commands should be input at the terminal.

```
  :RU,EDITR                       <---FMGR command to run EDITR.
  SOURCE FILE?                    <---EDITR response requesting file to
                                      be edited.
  /FILE1                          <---Name of file to be edited.
    LINE ONE OF SOURCE FILE.      <---First line of FILE1.
  //                              <---EDITR command to advance pending
                                      line by one line.
    LINE TWO OF SOURCE FILE.      <---New pending line.
  /MFILE2                         <---EDITR command to merge FILE2 after
                                      pending line.
  EOF
  /ER                             <---EDITR command to end edit session,
                                      replacing original version of FILE1
                                      with edited version.
  END OF EDIT
```

The edited version of FILE1 can now be listed to show the effect of the merge command.

```
  :LI,FILE1                       <---FMGR command to list FILE1.

FILE1  T=00004 IS ON CR01000 USING 00001 BLKS R=0002

0001 LINE ONE OF SOURCE FILE.
0002 LINE TWO OF SOURCE FILE.
0003 LINE ONE OF FILE TO BE MERGED.
0004 LINE TWO OF FILE TO BE MERGED.
```

COMMENTS:

The file to be merged can be merged in after any desired point in the source file. The merge takes place after the pending line and before the next line. The next line becomes the new pending line.

L (DISPLAY A NUMBER OF LINES)

Displays a specified number of lines, starting with the pending line.

```
+-------------------------------------------------------------------+
|                                                                   |
| Ln [,lu]                                                          |
|                                                                   |
|-------------------------------------------------------------------|
|                                                                   |
| n                                                                 |
| Number of lines to be printed (starting with pending line).       |
|                                                                   |
| lu                                                                |
| Logical unit number of list device; default is user's terminal.   |
|                                                                   |
+-------------------------------------------------------------------+
```

EXAMPLE:

FILEA contains the following three lines of text:

```
  LINE ONE.
  LINE TWO.
  LINE THREE.
```

While on line one as the pending line, the first two lines can be displayed by the following command:

```
  /L2              <---EDITR command to display two lines of text.
    LINE ONE.      <---First line of text displayed.
    LINE TWO.      <---Second line of text displayed.
```

COMMENTS:

When a file is listed on the line printer using the File Manager LI command (see Chapter 3 for an explanation of this command), line numbers will precede each line of text. Using FILEA above as an example:

```
  :LL,6            <---Change list device to line printer (LU 6).
  :LI,FILEA        <---FMGR command to list file.
```

The display on the line printer will look like:

```
FILEA  T=00004 IS ON CR01000 USING 00001 BLKS R=0002


0001 LINE ONE.
0002 LINE TWO.
0003 LINE THREE.
```

If it desired that the file heading and line numbers not be listed, the EDITR "L" command can be used to list the file. In addition, the optional parameter, lu, can be specified to list the file on some devices other than the default list device which is the user's terminal.

If only the pending line is desired to be displayed on the user's terminal, the EDITR P command may also be used. Using the example above to demonstrate:

```
/1              <---Make line one pending line and display it.
  LINE ONE.
/P              <---Display pending line.
  LINE ONE.
```

n (DISPLAY A SPECIFIED LINE AND MAKE IT PENDING LINE)

Displays the requested line and makes it the pending line.

n
n The line number of the line to be displayed and made pending line.

EXAMPLE:

Given FILEA containing the following three lines of text:

```
LINE ONE.
LINE TWO.
LINE THREE.
```

Assuming the pending line is the first line in the file and the third line is desired to be displayed and made the pending line, the following command can be issued:

```
/3              <---EDITR command to display third line of file
                    and make it pending line.
  LINE THREE.   <---Displayed third line of file.
/P              <---Display pending line.
  LINE THREE.   <---Displayed pending line.
```

COMMENTS:

If the line number requested is less than or equal to the current pending line number, the destination work area replaces the source work area, changing text line numbers, if any insertions or deletions were made to the text.

/ OR + (SPACE DOWN A NUMBER OF LINES)

Advances the pending line the specified number and displays the new pending line.

+[n[,lu]]

or

/[n[,lu]]

n
The number of lines to be skipped (default is one).

lu
Logical unit number of device on which the display occurs (default is the user's terminal). EDITR only recognizes this parameter when command is used in conjunction with an exchange command (G,Y, X,Z,V, and U).

EXAMPLE:

Given FILEA containing the following three lines,

```
LINE ONE.
LINE TWO.
LINE THREE.
```

If you are in the edit mode and line one is the pending line, you can make line three the pending line and display it by using the space down a number of lines command.

```
/P              <--------------Display pending line.
  LINE ONE.     <--------------Pending line displayed.
/+2             <--------------Skip two lines and display pending
                               line.
  LINE THREE.<--------------Pending line displayed.
```

The same thing can be done by using the slash (/) instead of the plus (+) sign.

```
/P              <--------------Display pending line.
  LINE ONE.     <--------------Pending line displayed.
//2             <--------------Skip two lines and display pending
                               line.
  LINE THREE.<--------------Pending line displayed.
```

COMMENTS:

The optional lu parameter can be used in conjunction with an exchange command. The following commands would be an example of this:

```
/1              <--------------Makes line one pending line.
  LINE ONE.     <--------------Pending line displayed.
/XLINE/LINE NUMBER<---------Exchange each occurence of the word
                              LINE with LINE NUMBER.
/+3,6           <--------------Make the above requested exchange of
                              words for three lines starting from
                              the pending line and print the
                              changed lines on the printer which
                              is lu 6.
```

The changed lines will then be printed on the line printer. If the lu parameter had been omitted, the changed lines would have been displayed on the user's terminal.

N (DISPLAY A PENDING LINE NUMBER)

Displays the line number of the pending line in the source work area.

N
No parameters required

EXAMPLE:

FILEA contains the following three lines of text,

```
LINE ONE.
LINE TWO.
LINE THREE.
```

If you are in the edit mode and line one is the pending line then you could give the following commands at your terminal

```
/P            <--------------Display pending line.
  LINE ONE.   <--------------Pending line displayed.
/N            <--------------Display pending line number.
  1           <--------------Pending line number.
/+2           <--------------Space down two lines and display
                             pending line.
  LINE THREE.<--------------Pending line displayed.
/N            <--------------Display pending line number.
  3           <--------------Pending line number.
```

COMMENTS:

For an explanation of what the source work area is see the EDITR WORK AREAS section in this Chapter.

ND (DISPLAY LINE NUMBER IN DESTINATION WORK AREA)

Displays the line number of the last line written to the destination work area. The source work area pending line and EOF's are not included in the count.

ND
No parameters required

EXAMPLE:

FILEA contains the following three lines of text

```
LINE ONE.
LINE TWO.
LINE THREE.
```

If you are in the edit mode and line one is the pending line then you could input the following commands:

```
/P            <--------------Display pending line.
  LINE ONE.   <--------------Pending line displayed.
/ND           <--------------Display line number in destination
                             work area.
              <--------------Since no lines have been sent to
                             the destination work area yet, no
                             line number is displayed.
//            <--------------Space down one line and display
                             pending line.
  LINE TWO.   <--------------Displayed pending line.
/ND           <--------------Display line number in destination
                             work area.
  1           <--------------Line number of last line written to
                             the destination work area.
/+3           <--------------Space down three lines and display
                             pending line.
EOF           <--------------End of file.
/ND           <--------------Display line number in destination
                             work area.
  3           <--------------Line number of last line written to
                             the destination work area.
                             (Remember that the file contains
                             only three lines of text, all
                             three lines have been transferred
                             to the destination work area, and
                             the EOF is not included in the count)
```

COMMENTS:

For a description of the EDITR source and destination work areas see the EDITR WORK AREAS section in this Chapter.

H (DISPLAY NUMBER OF CHARACTERS IN PENDING LINE)

Displays the number of characters in the pending line. The count includes a padded blank if the number of characters is uneven.

H
No parameters required

EXAMPLE:

FILEA contains the following two lines of text

```
THIS LINE HAS FORTY-ONE ASCII CHARACTERS.
THIS LINE CONTAINS FORTY-SIX ASCII CHARACTERS.
```

If you are in the edit mode then you could input the following commands at your terminal

```
/P              <--------------Display pending line.
  THIS LINE HAS FORTY-ONE ASCII CHARACTERS.
/H              <--------------Display number of characters in
                               pending line.
  42            <--------------Number of characters in pending line
                               (includes padded blank).
//              <--------------Space down one line and display
                               pending line.
  THIS LINE CONTAINS FORTY-SIX ASCII CHARACTERS.
/H              <--------------Display number of characters in
                               pending line.
  46            <--------------Number of characters in pending line.
```

COMMENTS:

The count includes a padded blank if the number of characters is uneven, because EDITR generates word-length records of two characters per word. This padding may be lost during character edits but replaced as the line is added to the destination work area as shown in the following example.

```
:RU,EDITR     <--------------FMGR Command to run EDITR.
SOURCE FILE?  <--------------EDITR requesting name of file to be
                               edited.
/0            <--------------Put empty file into EDITR's source
                               work area.
```

The following command enters a line of text into the EDITR's source work area

```
/ THIS LINE HAS FORTY-ONE ASCII CHARACTERS.
```

The line has not been transferred to the EDITR's destination work area yet; therefore, if you request the number of characters in the pending line the padded blank will not be included.

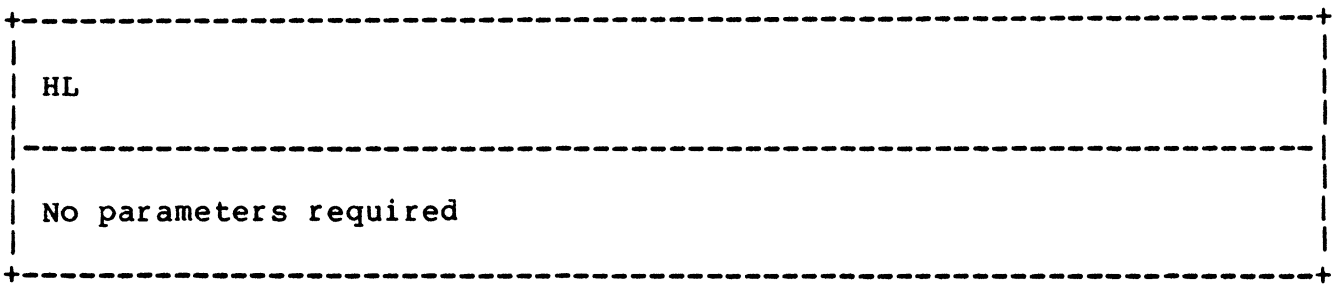
```
/H              <--------------Display number of characters in
                               pending line.
  41            <--------------Displayed number of characters.
```

Now by moving the line to the EDITR's destination work area, the padded blank will be included when the number of characters is requested.

```
/l              <--------------Display first line of text and
                               make it the pending line.
  THIS LINE HAS FORTY-ONE ASCII CHARACTERS.
/H              <--------------Display number of characters in
                               pending line.
  42            <--------------Displayed number of characters.
```

HL (DISPLAY HEADER)

Displays a header which facilitates column location.



EXAMPLE:

```
/HL              <--------------Display column header.
  '''/'''1'''/'''2'''/'''3'''/'''4'''/'''5'''/'''6'''/'''7'''/'
'8
```

^ (BACK UP IN DESTINATION WORK AREA)

Allows user to back up a specified number of lines in the destination work area and display the new pending line.

```
+-----------------------------------------------------------------+
|                                                                 |
| ^[n]                                                            |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| n                                                               |
| Number of lines to be backed up in destination work area.  Default |
| is to back up one line.                                         |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

FILEA contains the following three lines of text

```
LINE ONE.
LINE TWO.
LINE THREE.
```

If you are in the edit mode, the following commands can be input

```
/P            <--------------Display pending line.
  LINE ONE.   <--------------Pending line displayed.
/+2           <--------------Advance pending line two lines and
                             display new pending line.
  LINE THREE.<--------------Pending line displayed.
/^2           <--------------Back up pending line two lines and
                             display new pending line.
  LINE ONE.   <--------------Pending line displayed.
```

COMMENTS:

Following input of this command, EDITR copies the remainder of the source work area (pending line to end of file) into the destination work area. The destination work area then becomes the new source work area with the pending line moved back n lines. All lines prior to the new pending line are then copied into a new destination work area (lines 1 through n-1).

If n is so large that it causes the pointer to back up past the beginning of the destination work area, the error message ?? is displayed and a smaller value should be specified.

S (DISPLAY APPROXIMATE NUMBER OF WORDS IN DESTINATION WORK AREA)

Prints the approximate number of words in the destination work area.

S
No parameters required

EXAMPLE:

FILEA contains the following three lines of text

```
LINE ONE.
LINE TWO.
LINE THREE.
```

If you are in the edit mode you can input the following commands

```
/P              <--------------Display pending line.
  LINE ONE.     <--------------Pending line displayed.
/L3             <--------------Display three lines of text.  Command
  LINE ONE.                    also moves the three lines to the
  LINE TWO.                    destination work area by moving the
  LINE THREE.                  pending line down three lines.
EOF             <--------------End of file.
/S              <--------------Display number of words in destination
                               work area.
  19            <--------------Number of words in destination work
                               area.
```

COMMENTS:

This command can be used to determine the need to break up files into smaller segments (pieces) which would result in large paper tapes. For example, a paper tape containing more than 14,000 words exceeds the standard box size. If the word count is determined with the S command, the file can be broken with a series of zero-length records in the appropriate spots.

P (EDIT AND DISPLAY THE PENDING LINE)

Edits and displays the pending line.

```
+-------------------------------------------------------------------+
|                                                                   |
| P[editstring]                                                     |
|                                                                   |
|-------------------------------------------------------------------|
|                                                                   |
| editstring                                                        |
| The editstring can be composed of delimiters used as place holders |
| to preserve existing text, new text to be inserted or to replace  |
| existing text, and a non-printing control command to replace,     |
| insert or delete characters.  If no editstring is specified, then |
| the existing pending line is displayed. (Delimiters must be the   |
| current prompt character-default prompt character is a slash).    |
|                                                                   |
+-------------------------------------------------------------------+
```

EXAMPLE:

FILEA contains the following two lines of text:

```
LINE ONE.
LINE TWO.
```

The following edit session demonstrates use of the "P" command.

```
:RU,EDITR       <----------FMGR Command to run EDITR.
SOURCE FILE?
/FILEA          <----------File to be edited.
  LINE ONE.     <----------EDITR responds with firs line of file.
/P              <----------Display pending line.
  LINE ONE.     <----------Pending line displayed.
/P/////FIRST.   <----------Edit pending line and display new
                           pending line.
  LINE FIRST.   <----------EDITR responds with new pending line.
/ER             <----------End edit session replacing old
                           version of FILEA with edited version.
END OF EDIT     <----------EDITR response indicating edit session
                           is over.
```

COMMENTS:

The O and C commands are similar to the P command when used for character edits (see the following two sections for a description of these commands).

The P command is also used frequently to display the unedited pending line. Many of the examples for other commands make use of this feature.

C (EDIT PENDING LINE AND ADVANCE LINE)

Allows editing of the pending line and advances the pending line following the edit.

```
+-------------------------------------------------------------------+
|                                                                   |
| C[editstring]                                                     |
|                                                                   |
|-------------------------------------------------------------------|
|                                                                   |
| editstring                                                        |
| The editstring can be composed of delimiters used as place holders |
| to preserve existing text, new text to be inserted or to replace  |
| existing text, and a non-printing control command to replace,     |
| insert or delete characters. If no editstring is specified, then  |
| the pending line will be displayed, no change will be made to it, |
| and the pending line will be advanced one line and displayed.     |
| (Delimiters must be the current prompt character-default prompts  |
| character is a slash).                                            |
|                                                                   |
+-------------------------------------------------------------------+
```

EXAMPLE:

FILEA contains the following two lines of text:

```
LINE ONE
LINE TWO
```

To change the above two lines, the following set of commands can be input while in the edit mode

```
/P              <----------Display pending line.
  LINE ONE      <----------Pending line displayed.
/C/////FIRST    <----------Edit pending line changing ONE to FIRST.
                           Display edited line, advance pending
                           line, and display new pending line.
  LINE FIRST    <----------Edited line displayed.
  LINE TWO      <----------New pending line.
/C/////SECOND   <----------Edit pending line changing TWO to
                           SECOND. Display edited line, advance
                           pending line, and display new pending
                           line.
  LINE SECOND   <----------Edited line displayed.
EOF             <----------End of file.
```

O (DUPLICATE PENDING LINE AND EDIT)

Duplicates the pending line and allows character edits on the duplicate.

```
+-----------------------------------------------------------------+
|                                                                 |
| O[editstring]                                                   |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| editstring                                                      |
| The editstring is composed of delimiters used as place holders  |
| to preserve existing text, new text to be inserted or to replace|
| existing text, and a non-printing control command to replace,   |
| insert or delete characters.  If no editstring is specified, then|
| the existing pending line is duplicated and displayed. (Delimiters|
| must be the current prompt character-default prompt character is|
| a slash).                                                       |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

FILEA contains the single line of text:

```
   LINE ONE
```

If you are in the edit mode the following EDITR commands can be input at your terminal to create two additional lines:

```
   /O              <-------------Duplicate pending line and display
                                 duplicate.
     LINE ONE      <-------------Duplicate line displayed.
   /P/////TWO      <-------------Change ONE to TWO and display edited
                                 line.
     LINE TWO      <-------------Edited line displayed.
   /O              <-------------Duplicate pending line and display
                                 duplicate.
     LINE TWO      <-------------Duplicate line displayed.
   /P/////THREE    <-------------Change TWO to THREE and display edited
     LINE THREE                  line.
   /1              <-------------Make line one pending line and display
                                 it.
     LINE ONE      <-------------Pending line displayed.
   /L3             <-------------Display three lines of text.
     LINE ONE
     LINE TWO
     LINE THREE
   EOF             <-------------End of file.
```

COMMENTS:

A more efficient method of accomplishing the above example would be to use the editstring parameter in the O command. For example:

```
/P              <-------------Display pending line.
  LINE ONE      <-------------Pending line displayed.
/O////TWO       <-------------Duplicates pending line and
                              edits duplicate.
  LINE TWO            <-------Edited line displayed and made new
                              pending line.
/O////THREE     <------------Duplicates pending line and
                              edits duplicate.
  LINE THREE    <------------Edited line displayed and made
                              new pending line.
```

R (REPLACE PENDING LINE WITH TEXT)

Replaces the pending line with new text entered at the terminal.

Rtext
text New line of text to replace pending line. If no new text is specified, the pending line becomes zero length.

EXAMPLE:

The following example demonstrates use of this command to replace the pending line with a new line of text.

```
/P            <--------------Display pending line.
  LINE ONE.   <--------------Pending line displayed.
/RFIRST LINE.<--------------Replace pending line text with new
                            text.
/P            <--------------Display pending line.
  FIRST LINE.<--------------Pending line displayed.
```

I (INSERT TEXT BEFORE PENDING LINE)

Inserts a new line of text before the pending line.

```
+-------------------------------------------------------------------+
|                                                                   |
| Itext                                                             |
|                                                                   |
|-------------------------------------------------------------------|
|                                                                   |
| text                                                              |
| Line of text to be inserted before pending line in destination    |
| work area.  If command is entered with no new text, the inserted  |
| line becomes zero length.                                         |
|                                                                   |
+-------------------------------------------------------------------+
```

EXAMPLE:

FILEA contains the following three lines of text

```
LINE ONE.
LINE TWO.
LINE THREE.
```

If you are in the edit mode, you can input the following commands:

```
/P            <--------------Display pending line.
  LINE ONE.   <--------------Pending line displayed.
//            <--------------Advance pending line one line and
                             display new pending line.
  LINE TWO.   <--------------Pending line displayed.
/INEW LINE.   <--------------Insert new line of text before
                             pending line.
/1            <--------------Make line one pending line and
                             display pending line.
  LINE ONE.   <--------------Pending line displayed.
/L4           <--------------Display four lines of text.
  LINE ONE.
  NEW LINE.
  LINE TWO.
  LINE THREE.
EOF
```

{ } (INSERT TEXT AFTER PENDING LINE)

Used to insert new text immediately after the pending line. The new line then becomes the pending line. Note that {} represents a space.

{ }text
text New line of text to be inserted following pending line. If command is entered without new text, the new line has zero length.

EXAMPLE:

FILEA contains the following three lines of text

```
LINE ONE.
LINE TWO.
LINE THREE.
```

If you are in the edit mode, then you could input the following commands

```
/P            <--------------Display pending line.
  LINE ONE.   <--------------Pending line displayed.
/ NEW LINE.   <--------------Insert new line of text following
                             pending line and make it new pending
                             line.
/P            <--------------Display pending line.
  NEW LINE.   <--------------Pending line displayed.
/1            <--------------Make line one pending line and
                             display pending line.
  LINE ONE.   <--------------Pending line displayed.
/L4           <--------------Display four lines of text starting
                             from pending line.
  LINE ONE.
  NEW LINE.
  LINE TWO.
  LINE THREE.
EOF           <--------------End of file.
```

- (DELETE A NUMBER OF LINES)

Deletes a specified number of lines in the text.

```
+-----------------------------------------------------------------+
|                                                                 |
| -n                                                              |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| n                                                               |
| Number of lines of text to be deleted (starting with pending line).|
| If no value is specified, one line is deleted.                  |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

FILEA contains the following three lines of text

```
LINE ONE.
LINE TWO.
LINE THREE.
```

If you are in the edit mode, then you could give the following set of commands:

```
/P            <--------------Display pending line.
  LINE ONE.   <--------------Pending line displayed.
/-2           <--------------Delete two lines of text and display
                             new pending line.
  LINE THREE.<--------------Pending line displayed.
/1            <--------------Make line one pending line and display
                             pending line.
  LINE THREE.<--------------Pending line displayed.
/L3           <--------------Display three lines of text.
  LINE THREE.
EOF           <--------------End of file.
```

CNTL/R (REPLACE CHARACTERS)

Used in character edits to replace characters on a line.

CNTL/R
A control R is input by striking the "R" key while depressing the "CNTL" key on the terminal. It is a non-printing character. (Refer to the EDITR IN A MULTIPOINT ENVIRONMENT section for Multipoint operation).

COMMENTS:

Character replacement is the EDITR default mode so that the control character CNTL/R can be omitted when EDITR is initially turned on, or following a P,C, or O command. It is only required following one of the control characters CNTL/I, CNTL/S, or CNTL/C.

Any embedded character you wish to preserve in the original text is skipped by entering the current delimiter (/). Using the tab character will also cause characters to be skipped. Skipped characters in the new line appear exactly as they did in the old line. A carriage return preserves the rest of the line unless CNTL/T was specified to truncate.

An example of a character edit combination where the CNTL/R is required is shown below.

```
/P                <-------------Display pending line.
  OLD EDITR TEST FILE
/CNEW//////     /////FOR TESTS
 ^         ^     ^    ^
 |         |     |    |
 |         |     |  CNTL/S <----Insert new text.
 |         |     |
 |         |   CNTL/R  <--------Return to replace with control
 |         |                    character to preserve text.
 |         |
 |       CNTL/C  <-------------Cancel with space for place holder.
 |
 |
 |----------------------------Replace with no control character.

  NEW EDITR FILE FOR TESTS  <--Edited line of text.
```

CNTL/I OR CNTL/S (INSERT CHARACTERS)

Used to insert characters in the pending line.

CNTL/I or CNTL/S
A control I or S is input by striking the "I" or "S" key while depressing the "CNTL" key on the terminal. They are non-printing characters. (Refer to the EDITR IN A MULTIPOINT ENVIRONMENT section for Multipoint operation).

EXAMPLES:

1) CNTL/S example.

```
/P              <---------------Display pending line.
  LINE ONE.     <---------------Pending line displayed.
/P////////^OF TEXT <-----------Edit pending line and display it.
          |
          -- CNTL/S

  LINE ONE OF TEXT.   <--------Edited version of pending line.
```

2) CNTL/I example.

```
/P              <---------------Display pending line.
  LINE ONE.     <---------------Pending line displayed.
/P////////^         <-----------Edit pending line and display it.
          |
          -- CNTL/I
 OF TEXT            <-----------CNTL/I also acts as a carriage return.

 LINE ONE OF TEXT. <-----------Edited version of pending line.
```

COMMENTS:

The CNTL/I and CNTL/S commands are used with the P,C, and O commands (for a description of these commands, see the explanations given earlier in this Chapter).

Once CNTL/I or CNTL/S is entered, entering the current delimiter (/) to do character skipping inserts blanks. The tab character also causes the tabbed number of blanks to be inserted.

CNTL/C (CANCEL CHARACTERS)

Used to delete characters from the pending line.

CNTL/C
A control C is input by striking the "C" key while depressing the "CNTL" key on the terminal. It is a non-printing character (Refer to the EDITR IN A MULTIPOINT ENVIRONMENT section for Multipoint operation).

EXAMPLE:

```
/P                      <---------Display pending line.
  LINE ONE OF TEXT.     <---------Pending line displayed.
/P////////XXXXXXXX       <---------Edit pending line and display.
          ^
          |
        CNTL/C    <-------------Delete characters.

  LINE ONE.             <---------Edited version of pending line.
```

COMMENTS:

Each character or place holder entered after CNTL/C will delete one character from the pending line. Character skipping, i.e., entering the current delimiter deletes characters. The tab character deletes everything up to the tab stop. When carriage return is entered, the pending line will be left justified.

The CNTL/C command is used in conjunction with the P,C, or O commands. For a description of these commands, see the explanations given earlier in this Chapter.

CNTL/T (TRUNCATE CHARACTERS)

Used to delete characters from the end of the pending line. When the control character is entered, the remainder of the line is eliminated.

CNTL/T
A control T is input by striking the "T" key while depressing the "CNTL" key on the terminal. It is a non-printing character.

EXAMPLE:

```
/P                <------------Display pending line.
  LINE ONE OF TEXT  <----------Pending line displayed.
/P////////          <------------Edit pending line and display.
          ^
          |
       CNTL/T       <------------Eliminate characters following CNTL/T.

  LINE ONE          <------------Pending line displayed.
```

COMMENTS:

When operating under multipoint, the CNTL/T command can be replaced by the Q command. For details on the Q command see the section on EDITR IN A MULTIPOINT ENVIRONMENT.

B (FIND A LINE WITH A FIND FIELD -- SOF to EOF)

Searches the source work area, from the start-of-file to the end-of-file, for the first line of text which matches the find field. When a line containing the correct character string is found, it becomes the new pending line. The search ends at EOF if no match is found.

```
+-------------------------------------------------------------------+
|                                                                   |
| B                <------------Search for zero length record.      |
|  ^                                                                |
|  |---CNTL/@                                                       |
|                                                                   |
| Btext            <------------Search for left justified text.     |
|                                                                   |
| Btext            <------------Search for text anywhere in line.  |
|  ^                                                                |
|  |---ESC key                                                      |
|                                                                   |
| B/text           <------------Search for text in window.         |
|                                                                   |
| B;text           <------------Search for field at tab stop.      |
|                                                                   |
| B                <------------Successive search for same field.  The|
|                               same line will always be found by this|
|                               command.                            |
|                                                                   |
|-------------------------------------------------------------------|
|                                                                   |
| CNTL/@                                                            |
| A control @ is input by striking the "@" key while depressing     |
| the "CNTL" key on the terminal.  It is a non-printing character.  |
|                                                                   |
| text                                                              |
| This is the portion of text that is to be found by the B command. |
|                                                                   |
| ESC                                                               |
| The ESC key is located on the terminal.  It is a non-printing     |
| character.                                                        |
|                                                                   |
+-------------------------------------------------------------------+
```

EXAMPLES:

FILEA contains the following four lines of text

```
   LINE ONE OF EXAMPLE.
   LINE TWO OF EXAMPLE.
                                  <---(zero length record)
          LINE FOUR OF EXAMPLE.
```

1) B CNTL/@ example.

```
/B              <---------------Search for zero length record and
  ^                               make it pending line.
  |
  ---CNTL/@
                <---------------Pending line (zero length record).
/^              <---------------Move pending line back one line and
                                display new pending line.
  LINE TWO OF EXAMPLE.  <------Pending line displayed.
```

2) Btext example.

```
/BLINE          <---------------Search for first occurence of LINE
                                (left justified), making line
                                pending line, and display it.
  LINE ONE OF EXAMPLE.  <------Pending line displayed.
```

3) Btext example.

```
  ^
  |--ESC

/BTWO           <---------------Search for first occurrence of TWO,
  ^                             making line containing TWO pending
  |--ESC                        line, and display it.
  LINE TWO OF EXAMPLE.  <------Pending line displayed.
```

4) B/text example.

```
/W6,11          <---------------Make window between columns 6
                                through 11.
/BLINE          <---------------Find first occurence of LINE within
                                window constraints, making line
  ^                             containing text pending line,
  |--ESC                        and display pending line.
         LINE FOUR OF EXAMPLE.
```

5) B;text example.

```
/B;LINE         <---------------Find first occurrence of LINE at tab
                                stop, making it pending line and
                                display pending line.
         LINE FOUR OF EXAMPLE.
```

6) B example.

```
/B;LINE          <---------------Find first occurence of LINE at tab
                                 stop, making it pending line and
                                 display pending line.
          LINE FOUR OF EXAMPLE.
/1               <---------------Make line one pending line and
                                 display pending line.
  LINE ONE OF EXAMPLE.
/B               <---------------Search for same field sought
                                 previously.
          LINE FOUR OF EXAMPLE.
```

COMMENTS:

The B command always starts at the first line of the file to find the text. Once the text is found, the line containing it becomes the pending line. The B command cannot be used for successive searches for the same text, because it will always find the same line unless the text being searched for is changed on that line. All lines passed over in the search are written to the destination work area; they are not deleted. The B command always forces the destination work area to replace the source work area before the search.

The slash in the command format B/text, is representative of the initial delimiter. If the delimiter has been changed (X - CHANGE EDITR PROMPT CHARACTER), then the appropriate delimiter character must be used in place of the /. If the initial tab character is changed (T - SET TAB STOPS), the new tab character must be used in place of the ;.

Note that if the window is set at its default values (1 to 150), the B/text command does the same thing as the B^text.

```
                                  |--ESC
```

F (FIND A LINE WITH FIND FIELD FROM PENDING LINE TO EOF)

Causes a search of the source work area, beginning at the line following the pending line and continuing to the end-of-file for a line containing a specified character string. When a line containing this field is found, it becomes the new pending line. If the line is not found, the search ends at the end-of-file.

Ftext	<----------Search for left justified field.
Ftext ^ |--ESC	<----------Search for field embedded anywhere in line.
F/text	<----------Search for field within a window.
F;text	<----------Search for field beginning at tab stop.
F	<----------Successive searches for same field.

text This is the portion of text that is to be found by the F command.
ESC The ESC key is located on the terminal. It is a non-printing character.

EXAMPLE:

FILEA contains the following four lines of text

```
   THIS IS AN EXAMPLE
   FOR DEMONSTRATING
   THE F COMMAND
          THE F COMMAND
```

1) Ftext example.

```
/P                      <--------Display pending line.
  THIS IS AN EXAMPLE    <--------Pending line displayed.
/FTH                    <--------Search for left justified field
                                 with text TH, making line containing
                                 field pending line, and display it.
  THIS IS AN EXAMPLE    <--------Pending line displayed.
/+                      <--------Advance pending line one line and
                                 display new pending line.
  FOR DEMONSTRATING     <--------New pending line displayed.
/FTH                    <--------Search for left justified field
                                 with text TH, making line containing
                                 field pending line, and display it.
  THE F COMMAND         <--------Pending line displayed.
```

2) Ftext example.

```
 |--ESC

/P                      <--------Display pending line.
  THIS IS AN EXAMPLE    <--------Pending line displayed.
/FDEMO                  <--------Search for text embedded anywhere
  ^                              in line, making line pending line,
  |--ESC                         and display it.

  FOR DEMONSTRATING     <--------Pending line containing text.
/+                      <--------Advance pending line by one line
                                 and display new pending line.
  THE F COMMAND         <--------Pending line displayed.
/FDEMO                  <--------Search for text embedded anywhere
  ^                              in line, making line pending line,
  |--ESC                         and display it.

EOF                     <--------End of file.  Indicates remainder of
                                 of file was searched and text was
                                 not found.
```

3) F/text example.

```
/P                      <--------Display pending line.
  THIS IS AN EXAMPLE    <--------Pending line displayed.
/W6,10                  <--------Set windown between columns 6 and 10.
/F/COM                  <--------Search for text within window, making
                                 line containing text pending line,
                                 and display pending line.
  THE F COMMAND         <--------Pending line displayed.
```

4) F;text example.

```
/P                      <--------Display pending line.
  THIS IS AN EXAMPLE    <--------Pending line displayed.
/F;THE                  <--------Search for text beginning at tab
                                 stop, making line containing text
                                 pending line, and display pending
                                 line.
          THE F COMMAND   <-----Pending line displayed.
```

5) F example.

```
/P                      <--------Display pending line.
  THIS IS AN EXAMPLE    <--------Pending line displayed.
/FTHE                   <--------Search for text embedded anywhere
  ^                              in line, making line containing
  |--ESC                         text new pending line, and display
                                 new pending line.

  THE F COMMAND         <--------Pending line displayed.
/F                      <--------Successive search for same text,
                                 making line containing text new
                                 pending line, and display line.
          THE F COMMAND   <-----Pending Line displayed.
```

COMMENTS:

The slash in the command format F/text, is representative of the initial delimiter. If the delimiter has been changed (see the CHANGE EDITR PROMPT CHARACTER section), then the appropriate delimiter character must be used in place of the /. If the initial tab character is changed (see the SET TAB STOPS section). the new tab character must be used in place of the ;.

Note that if the window is set at its default values (1 to 150), then the F/text command does the same thing as the F^text.

|--ESC

D (DELETE LINES TO FIND FIELD OR EOF)

Deletes a block of text from the pending line to the line containing a specified field. After the deletion, the line of text containing the specified field becomes the new pending line. If the specified field is not encountered, the remainder of the file is deleted. If this command is used without specifying a field, the last field entered is used.

D ^ |--CNTL/@	<----------Delete lines until a zero length field is encountered.
Dtext	<----------Delete lines until specified text is found in left justified field.
Dtext ^ |--ESC	<----------Delete lines until specified text is found in field located anywhere in line.
D/text	<----------Delete lines until specified text is found in field located within window.
D;text	<----------Delete lines until specified text is found in field beginning at tab stop.
D	<----------Successive deletes to specified text in prior delete command.

PARAMETERS:

CNTL/@
A control @ is input by striking the "@" key while depressing the "CNTL" key on the terminal. It is a non-printing character.

text
This is the portion of text that is to be found by the D command.

ESC
The ESC key is located on the terminal. It is a non-printing character.

EXAMPLE:

FILEA contains the following 7 lines of text

```
   THIS IS AN EXAMPLE
   DEMONSTRATING THE
                          <--------Zero length record.
   DELETE LINES COMMAND.
                          <--------Zero length record.
   THIS NEXT LINE BEGINS
         IN COLUMN 7.

1) D   example.
    ^
    |--CNTL/@

   /1                  <------------Make line one pending line and display
                                    it.
     THIS IS AN EXAMPLE  <--------Pending line displayed.
   /D                  <------------Delete all lines until zero length
     ^                              record is found.  Make zero length
     |--CNTL/@                      record pending line and display it.

                       <------------Zero length record displayed.
   /1                  <------------Make line one pending line and display
                                    it.
                       <------------Pending line displayed (zero length
                                    record).
   /L6                 <------------Display six lines of text.

     DELETE LINES COMMAND.

     THIS NEXT LINE BEGINS
           IN COLUMN 7.
   EOF                 <------------End of file.

2) Dtext example.

   /1                  <------------Make line one pending line and display
                                    it.
     THIS IS AN EXAMPLE  <--------Pending line displayed.
   /DTHIS              <------------Delete all lines until THIS is
                                    encountered in a left justified
                                    field, making line containing
                                    text pending line, and displaying it.
     THIS NEXT LINE BEGINS  <-----Pending line displayed.
   /1                  <------------Make line one pending line and display
                                    it.
     THIS NEXT LINE BEGINS  <-----Pending line displayed.
   /L6                 <------------Display six lines of text.
     THIS NEXT LINE BEGINS
           IN COLUMN 7.
   EOF                 <------------End of file.
```

```
3) Dtext example.
   ^
    |--ESC

   /1                 <------------Make line one pending line and display
                                   it.
     THIS IS AN EXAMPLE     <-----Pending line displayed.
   /DBEGINS             <------------Delete all lines until BEGINS is
     ^                             encountered anywhere in file, making
     |--ESC                        line containing BEGINS pending line,
                                   and displaying it.
     THIS NEXT LINE BEGINS  <-----Pending line displayed.
   /1                 <------------Make line one pending line and display
                                   it.
     THIS NEXT LINE BEGINS  <-----Pending line displayed.
   /L6
     THIS NEXT LINE BEGINS
           IN COLUMN 7.
   EOF                <------------End of file.

4) D/text example.

   /1                 <------------Make line one pending line and display
                                   it.
     THIS IS AN EXAMPLE     <-----Pending line displayed.
   /W10,15            <------------Set window between columns 10 and 15.
   /D/LINE            <------------Delete all lines until LINE is located
                                   in a field within the specified
                                   window.  Make line containing LINE
                                   pending line and display it.
     THIS NEXT LINE BEGINS  <-----Pending line displayed.
   /1                 <------------Make line one pending line and display
                                   it.
     THIS NEXT LINE BEGINS  <-----Pending line displayed.
   /L6                <------------Display six lines of text.
     THIS NEXT LINE BEGINS
           IN COLUMN 7.
   EOF                <------------End of file.
```

5) D;text example.

```
/1                  <------------Make line one pending line and display
                                 it.
  THIS IS AN EXAMPLE      <-----Pending line displayed.
/D;IN               <------------Delete all lines until IN is
                                 encountered at the first tab stop
                                 on some line.  Make line containing
                                 IN pending line and display it.
        IN COLUMN 7.      <-----Pending line displayed.
/1                  <------------Make line one pending line and display
                                 it.
        IN COLUMN 7.      <-----Pending line displayed.
/L6                 <------------Display six line of text.
        IN COLUMN 7.
EOF                 <------------End of file.
```

6) D example.

```
/1                  <------------Make line one pending line and display
                                 it.
  THIS IS AN EXAMPLE      <-----Pending line displayed.
/D                  <------------Delete all lines until a zero length
 ^                               record is encountered, making zero
  |--CNTL/@                      length record pending line and display
                                 it.
                    <------------Zero length record displayed as
                                 pending line.
/L6                 <------------Display six lines of text.

  DELETE LINES COMMAND

  THIS NEXT LINE BEGINS
        IN COLUMN 7.
EOF                 <------------End of file.
/1                  <------------Make line one pending line and display
                                 it.
                           <-----Pending line displayed (zero length
                                 record).
/D                  <------------Delete all lines until text field
                                 specified in last delete command is
                                 encountered.  Make line containing
                                 text new pending line and display it.
                           <-----Pending line displayed (zero length
                                 record).
 /L6                 <------------Display six lines of text.

  THIS NEXT LINE BEGINS
        IN COLUMN 7.
EOF                 <------------End of file.
```

COMMENTS:

The slash in the command for D/text, is representative of the initial delimiter. If the delimiter has been changed (see the CHANGE EDITR PROMPT CHARACTER section), then the appropriate delimiter character must be used in place of the /. If the initial tab character is changed (see the SET TAB STOPS section), the new tab character must be used in place of the ;.

Note that if the window is set at its default values (1 to 150), then the

```
D/text
```

command does the same thing as the

```
Dtext.
 ^
 |--ESC
```

J (JUMP TO FIND FIELD LINE AND MAKE IT NOW PENDING LINE)

Used to either delete or move text. It causes the pending line pointer to jump to a line containing a specified text string in the source work area without moving the jumped over lines to the destination work area.

Jtext	<----------Jump to line containing text in left justified field.
J^text |--ESC	<----------Jump to line containing text embedded anywhere in line.
J/text	<----------Jump to line containing text embedded anywhere within window.
J	<----------Jump to same line that was jumped to with prior jump command.

text
This is the portion of text that is to be found by the J command.

ESC
The ESC key is located on the terminal. It is a non-printing character.

EXAMPLE:

```
Source Work Area      Pending Line      Destination Work Area
----------------      ------------      ---------------------
FIRST LINE            <----PL
      SECOND LINE
THIRD LINE
      FOURTH LINE

/JTHIRD     <-------------------Jump to line with text in left
                                justified field.
```

```
Source Work Area      Pending Line      Destination Work Area
----------------      ------------      ---------------------
FIRST LINE                              FIRST LINE
      SECOND LINE
THIRD LINE            <----PL
      FOURTH LINE

/JCOND      <--------------------Jump to line with text embedded
  ^                              in line.
  |--ESC

Source Work Area      Pending Line      Destination Work Area
----------------      ------------      ---------------------
FIRST LINE                              FIRST LINE
      SECOND LINE     <----PL           THIRD LINE
THIRD LINE
      FOURTH LINE

/W10,20     <--------------------Set window between columns 10 to 20.
/J/LINE     <--------------------Jump to line containing first
                                 occurrence of text which is embedded
                                 within window.

Source Work Area      Pending Line      Destination Work Area
----------------      ------------      ---------------------
FIRST LINE                              FIRST LINE
      SECOND LINE     <----PL           THIRD LINE
THIRD LINE                                    SECOND LINE
      FOURTH LINE

/J          <--------------------Jump to same line as previous jump
                                 command.

Source Work Area      Pending Line      Destination Work Area
----------------      ------------      ---------------------
FIRST LINE                              FIRST LINE
      SECOND LINE     <----PL           THIRD LINE
THIRD LINE                                    SECOND LINE
      FOURTH LINE                             SECOND LINE

/1          <--------------------Make line one pending line.  Also,
                                 copies remainder of source work area
                                 from pending line on down into
                                 destination work area and then makes
                                 destination work area new source work
                                 area.
```

Source Work Area	Pending Line	Destination Work Area
FIRST LINE	<----PL	
THIRD LINE		
SECOND LINE		
SECOND LINE		
SECOND LINE		
THIRD LINE		
FOURTH LINE		

COMMENTS:

Before using the J command for deleting and moving blocks of text, you should be familiar with the concept of source and destination work areas (refer to the EDITR WORK AREAS). Remember that whenever the pending line pointer goes back to a prior line in the source work area, the destination work area will become the new source work area.

Whenever text is sought with a J command, the first line containing the text in the source work area will become the pending line regardless of where the pending line pointer was prior to the command. The only line moved by the J command to the destination work area is the pending line prior to the jump.

The slash in the command format J/text, is representative of the initial delimiter. If the delimiter has been changed (see the CHANGE EDITR PROMPT CHARACTER section), then the appropriate delimiter character must be used in place of the /. If the initial tab character is changed (see the SET TAB STOPS section), the new tab character must be used in place of the ;.

Note that if the window default values (1 to 150) are used, the

```
J/text
```

command does the same thing as the

```
Jtext.
 ^
 |--ESC
```

G (CHARACTER REPLACE ON PENDING LINE)

Exchanges an old character string with a new character string on the pending line. The new pending line following the exchange is then displayed.

```
+-----------------------------------------------------------------+
|                                                                 |
| Goldstring/newstring                                            |
|                                                                 |
|-----------------------------------------------------------------|
|                                                                 |
| oldstring                                                       |
| Old character string to be exchanged with new. Can be located   |
| anywhere in pending line. If several identical character strings|
| are located on the line, all of them will be exchanged for the  |
| new string. Can be any length except zero.                      |
|                                                                 |
| newstring                                                       |
| This is the new character string which is to replace the old    |
| string. Can be any length including zero.                       |
|                                                                 |
+-----------------------------------------------------------------+
```

EXAMPLE:

```
/P              <---------------Display pending line.
  OLD STRING            <-------Pending line displayed.
/GOLD/NEW       <---------------Exchange OLD text for NEW text and
                                display new pending line.
  NEW STRING            <-------Pending line displayed.
```

COMMENTS:

If the pending line contains several identical strings of text, but not all of them are desired to be changed, then the set window command can be used to selectively change character strings. For example, if the pending line is

```
OLD OLD OLD OLD
```

and only the third occurence of OLD is desired to be exchanged for NEW, the following commands can be input to accomplish this.

```
/W8,12          <---------------Set window between columns 8 and 12.
/GOLD/NEW       <---------------Exchange OLD text with NEW text and
                                display new pending line.
  OLD OLD NEW OLD       <-------Pending line displayed.
```

Once the Goldstring/newstring Command has been issued, subsequent exchanges may be made by specifying G without any parameters, i.e., G parameters remain in effect until changed.

The exchange fields used in the G command are not tabbed, so that the tab character can be used like any other character in the exchange string.

The only time the current delimiter is recognized as a delimiter is the first time it occurs separating the old string and the new string.

Y (EXCHANGE ON PENDING LINE, DISPLAY NEXT OCCURRENCE OF PATTERN)

Performs an exchange of an old string with a new string on the pending line.

Yoldstring/newstring	<------Exchange all occurrences of old string on pending line with new string. After exchange, the next occurrence of old string becomes the new pending line. If same exchange on new pending line is desired, only the command Y need be input.

oldstring
Old string of text to be exchanged with new text string. It can be any length except zero.

newstring
This is the new text string which is to replace the old. It can be any length including zero.

EXAMPLE:

FILEA contains the following five lines of text

```
THIS IS AN OLD STRING.
THE OLD STRING CAN BE
REPLACED BY A NEW
STRING WITH THE Y
COMMAND.
```

To change all occurrences of the text, STRING, with the text, TEXT, the following commands can be input while in the edit mode.

```
/YSTRING/TEXT      <-----------Exchange STRING with TEXT on pending
                               line, display edited line, find next
                               occurrence of STRING, making it the
                               pending line and display it.
  THIS IS AN OLD TEXT.   <-----Edited line displayed.
  THE OLD STRING CAN BE  <-----New pending line displayed.
/Y           <-----------------Exchange STRING with TEXT on pending
                               line, display edited line, find next
                               occurrence of STRING, making it
                               pending line and display it.
  THE OLD TEXT CAN BE    <-----Edited line displayed.
  STRING WITH THE Y      <-----New pending line displayed.
/Y           <-----------------Exchange STRING with TEXT on pending
                               line, display edited line, find next
                               occurrence of STRING, making it
                               pending line and display it.
  TEXT WITH THE Y        <-----Edited line displayed.
EOF          <-----------------End of file.
```

COMMENTS:

A combination of Y and F commands can be used to do selective exchanges. The F command is entered (without parameters) to skip an exchange on this occurrence and find the next occurrence. Using FILEA as an example, the first and last occurrence of STRING can be changed to TEXT leaving the second occurrence unchanged by the following commands

```
/YSTRING/TEXT    <-------------Exchange STRING with TEXT on pending
                               line, display edited line, find next
                               occurrence of STRING, making it the
                               pending line and display it.
  THIS IS AN OLD TEXT.   <-----Edited line displayed.
  THE OLD STRING CAN BE  <-----New pending line displayed.
/F               <-------------Find next occurrence of STRING and
                               make it pending line leaving present
                               pending line unchanged.
  STRING WITH THE Y      <-----New pending line displayed.
/Y           <-----------------Exchange STRING with TEXT on pending
                               line, display edited line, find next
                               occurrence of STRING, making it
                               pending line and display it.
  TEXT WITH THE Y        <-----Edited line displayed.
EOF          <-----------------End of file.
```

If the string to be changed occurs more than once on the pending line, all occurrences of the string will be replaced by the new string. To selectively change identical strings on the pending line, the set window command can be used. For example, if the pending line were

```
OLD OLD OLD OLD
```

and only the third occurrence of old was desired to be changed, the following commands can be input.

```
/W8,12       <------------------Set window between columns 8 and 12.
/YOLD/NEW    <------------------Exchange OLD with NEW on pending
                                line, display edited line, find next
                                occurrence of OLD, making it pending
                                line and display it.
  OLD OLD NEW OLD          <-----Edited line displayed.
EOF          <------------------End of file.
```

The exchange fields used in the Y command are not tabbed, so that the tab character can be used like any other character in the exchange string.

Only the first occurrence of the current delimiter (/) is recognized. It is assumed to separate the old string and the new string.

X (ENABLE EXCHANGE PATTERN OVER RANGE OF LINES, WITH LIST)

Used to make string exchanges over a user specified range of lines and then list the lines which have been changed. It is always used in combination with a second command which defines the range of lines over which the string exchange is to take place.

Xoldstring/newstring	<------Exchange all occurrences of old string with new string over a specified range of lines within a column window. All changed lines are listed.
range command	<------Range of lines over which exchanges are to take place.

oldstring
Old string of text which is to be exchanged with new string. It can be any length except zero.

newstring
This is the new string of text which is to be exchanged for the old. It can be any length including zero.

range command
This is any command which will cause the pending line pointer to advance through the source file.

EXAMPLE:

FILEA contains the following five lines of text:

```
THIS IS AN OLD STRING.
THE OLD STRING CAN BE
REPLACED BY A NEW
STRING WITH THE X
COMMAND.
```

If it is desired to replace all occurrences of the text, STRING, with the text, TEXT, the following commands can be input

```
/XSTRING/TEXT      <------------Exchange all occurrences of STRING
                                with TEXT over the range specified
                                in the next command.
/FEOF              <------------Search all lines for STRING and
                                change to TEXT until line with
                                EOF is found in left justified
                                field.
  THIS IS AN OLD TEXT    <------Changed line of text.
  THE OLD TEXT CAN BE    <------Changed line of text.
  TEXT WITH THE X        <------Changed line of text.
EOF                <------------End of file.
```

COMMENTS:

Other examples of for range commands would be the advance line commands (+ or /), or specifying a specific line number for the pending line pointer to move to (n).

The default print device is the terminal that you input the commands at, but by using the delete lines command (D), the advance line commands (+ or /), or specifying a specific line number for the pending line pointer to move to (n), the optional LU command can be invoked to send the changed lines to the printer or a disk file.

The exchange takes place over all occurrences of the old string text within each line unless the window command is used to limit the range of columns over which the exchange is to take place. Using FILEA as an example to demonstrate this the following commands can be input to change only the STRING which starts in column 9 of line two.

```
/W8,15             <------------Set window between columns 8 and 15.
/XSTRING/TEXT      <------------Exchange command.
/+10               <------------Range command. Advance pending
                                line down 10 lines.
  THE OLD TEXT CAN BE    <------Changed line.
EOF                <------------End of file.
```

If additional exchanges are to be made beyond the line positioned to with the range command, the current parameters of the X command may be re-enabled by specifying X with no parameters.

The exchange fields used in the X command are not tabbed, so that the tab character can be used like any other character in the exchange string.

Only the first occurrence of the current delimiter is recognized. It is assumed to separate the old string and the new string.

Z (ENABLE EXCHANGE PATTERN OVER RANGE OF LINES, WITHOUT LIST)

Used to exchange a string of text with a new string of text over a specified number of lines. It is always used in combination with a second command which specifies the range of lines over which the exchange is to take place. It is the same as the X command except that there is no listing of the changed lines.

Zoldstring/newstring	<------Exchange all occurrences of old string with new string over a specified range of lines within a column window.
range command	<------Range of lines over which exchanges are to take place.

oldstring
Old string of text which is to be exchanged with new string. It can be any length except zero.

newstring
This is the new string of text which is to be exchanged for the old. It can be any length including zero.

range command
This is any command which will cause the pending line pointer to advance through the source file.

EXAMPLE:

FILEA contains the following five lines of text

```
THIS IS AN OLD STRING.
THE OLD STRING CAN BE
REPLACED BY A NEW
STRING WITH THE Z
COMMAND.
```

If it is desired to replace all occurrences of the text, STRING, with the text, TEXT, the following commands can be input

```
/ZSTRING/TEXT        <----------Exchange all occurrences of STRING
                                with TEXT over the range specified
                                in the next command.
/FEOF                <----------Search all lines for STRING and
                                change to TEXT until line with
                                EOF is found in left justified
                                field.
EOF                  <----------End of file.
/1                   <----------Make line one pending line.
/L5                  <----------Display five lines of text.
  THIS IS AN OLD TEXT.
  THE OLD TEXT CAN BE
  REPLACED BY A NEW
  TEXT WITH THE Z
  COMMAND.
EOF                  <----------End of file.
```

COMMENTS:

Other examples for range commands would be the advance line commands (+ or /), or specifying a specific line number for the pending line pointer to move to (n).

The exchange takes place over all occurrences of the old string text within each line unless the window command is used to limit the range of columns over which the exchange is to take place. Using FILEA as an example to demonstrate this, the following commands can be input to change only the STRING which starts in column 9 of line two.

```
/W8,15               <----------Set window between columns 8 and 15.
/ZSTRING/TEXT        <----------Exchange command.
/+10                 <----------Range command. Advance pending
                                line down 10 lines.
EOF                  <----------End of file.
/1                   <----------Make line one pending line.
/L5                  <----------Display five lines of text.
  THIS IS AN OLD STRING.
  THE OLD TEXT CAN BE
  REPLACED BY A NEW
  STRING WITH THE Z
  COMMAND.
EOF                  <----------End of file.
```

If additional exchanges are to be made beyond the line positioned to with the range command, the current parameters of the Z command may be re-enabled by specifying Z with no parameters.

The exchange fields used in the Z command are not tabbed, so that the tab character can be used like any other character in the exchange string.

Only the first occurrence of the current delimiter is recognized. It is assumed to separate the old string and the new string.

V (UNCONDITIONAL CHARACTER REPLACE, WITH LIST)

Used to insert, delete or replace characters on one or more lines starting from the first column of the edit window (see the SET WINDOW section for details on the W command). It is always used in combination with a second command which defines the range of lines over which the string exchange is to take place. The lines which are changed by this command are displayed to your terminal.

V/newstring	<----------Insert new string starting at first column of window.
Vxxx/	<----------Delete number of characters specified by number of characters of xxx starting at first column of window.
Vxxx/newstring	<----------Replace number of characters specified by number of characters of xxx with new string starting at first column of window.
range command	<----------Range of lines over which exchanges are to take place.

xxx
Can be any alphanumeric characters since only the number of characters in xxx matters. xxx is not used as a pattern for a search, but rather, to specify the number of characters to be replaced or deleted at the start of the window by the new character string. xxx can be a maximum of 148 alphanumeric characters.

newstring
This is the newstring which is to either replace the number of characters specified by xxx or to be inserted at the first column of the window.

range command
This is any command which will cause the pending line pointer to advance through the source file.

EXAMPLES:

FILEA contains the following three lines of text

```
AAAAAAAAAAAAAAA
BBBBBBBBBBBBBBB
CCCCCCCCCCCCCCC
```

1) V/newstring example (insert new string).

```
/W5,150            <------------Set window between columns 5 to 150.
/P                 <------------Display pending line.
  AAAAAAAAAAAAAAA    <---------Pending line displayed.
/V/XXX             <------------Insert XXX starting in the first
                                column of the window over the range
                                specified in the next command.
/+2                <------------Advance pending line down two lines.
  AAAAXXXAAAAAAAAAAA <---------Changed line displayed.
  BBBBXXXBBBBBBBBBBB <---------Changed line displayed.
  CCCCCCCCCCCCCCC    <---------New pending line.
```

2) Vxxx/ example (delete characters).

```
/W1,3              <------------Set window between columns 1 to 3.
/P                 <------------Display pending line.
  AAAAAAAAAAAAAAA    <---------Pending line displayed.
/V12345/           <------------Delete first five characters starting
                                in the first column of the window
                                over the range specified in the next
                                command.
/FEOF              <------------Find EOF.
  AAAAAAAAAA       <------------Changed line displayed (first 5
                                characters deleted).
  BBBBBBBBBB       <------------Changed line displayed (first 5
                                characters deleted).
  CCCCCCCCCC       <------------Changed line displayed (first 5
                                characters deleted).
EOF                <------------End of file.
```

3) Vxxx/newstring example (replace characters).

```
/W1,3              <------------Set window between columns 1 to 3.
/P                 <------------Display pending line.
  AAAAAAAAAAAAAAA
/VAA/1111          <------------Replace first two characters starting
                                in the first column of the window
                                over the range specified in the next
                                command with the string 1111.
/+                 <------------Advance pending line one line.
  1111AAAAAAAAAAAAA  <---------Changed line displayed.
  BBBBBBBBBBBBBB     <---------New pending line.
/W5,150            <------------Set window between columns 5 to 150.
/VBB/2222          <------------Replace first two characters starting
                                in the first column of the window
                                over the range specified in the next
                                command with the string 2222.
/+                 <------------Advance pending line one line.
  BBBB2222BBBBBBBBB  <---------Changed line displayed.
  CCCCCCCCCCCCCCC    <---------New pending line.
/1                 <------------Make line one pending line and display
                                it.
  1111AAAAAAAAAAAAA  <---------Pending line displayed.
/L5                <------------Display five lines of text.
  1111AAAAAAAAAAAAA
  BBBB2222BBBBBBBBB
  CCCCCCCCCCCCCCC
EOF                <------------End of file.
```

COMMENTS:

The exchange field starts at the first column of the window, but is not limited by the width of the window (note example 2 above).

The default print device is the terminal that you input the commands at, but by using the delete lines command (D), the advance line commands (+ or /), or specifying a specific line number for the pending line pointer to move to (n), the optional LU parameter in these commands can be invoked to send the changed lines to the printer or a disk file.

The exchange fields used in the V command are not tabbed, so that the tab character can be used like any other character in the exchange string.

Only the first occurrence of the current delimiter is recognized. It is assumed to separate the xxx parameter with the new string.

U (UNCONDITIONAL CHARACTER REPLACEMENT, WITHOUT LIST)

Used to insert, delete or replace characters on one or more lines starting from the first column of the edit window (see the SET WINDOW section for details on the W command). It is always used in combination with a second command which defines the range of lines over which the string exchange is to take place. It is the same as the V command except that there is no listing of the changed lines.

U/newstring	<----------Insert new string starting at first column of window.
Uxxx/	<----------Delete number of characters specified by number of characters of xxx starting at first column of window.
Uxxx/newstring	<----------Replace number of characters specified by number of characters of xxx with new string starting at first column of window.
range command	<----------Range of lines over which exchanges are to take place.

xxx
Can be any alphanumeric characters since only the number of characters in xxx matters. xxx is not used as a pattern for a search, but rather, to specify the number of characters to be replaced or deleted at the start of the window by the new character string. xxx can be a maximum of 148 alphanumeric characters.

newstring
This is the newstring which is to either replace the number of characters specified by xxx or to be inserted at the first column of the window.

range command
This is any command which will cause the pending line pointer to advance through the source file.

EXAMPLES:

FILEA contains the following three lines of text

```
AAAAAAAAAAAAAAA
BBBBBBBBBBBBBBB
CCCCCCCCCCCCCCC
```

1) U/newstring example (insert new string).

```
/W5,150            <-------------Set window between columns 5 to 150.
/P                 <-------------Display pending line.
  AAAAAAAAAAAAAAA       <-------Pending line displayed.
/U/XXX             <-------------Insert XXX starting in the first
                                 column of the window over the range
                                 specified in the next command.
/+2                <-------------Advance pending line down two lines.
  CCCCCCCCCCCCCCC       <-------New pending line.
/1                 <-------------Make line one pending line and display
                                 it.
  AAAAXXXAAAAAAAAAAA    <-------Pending line displayed.
/L5                <-------------Display five lines of text.
  AAAAXXXAAAAAAAAAAA
  BBBBXXXBBBBBBBBBBB
  CCCCCCCCCCCCCCC
EOF                <-------------End of file.
```

2) Uxxx/ example (delete characters).

```
/W1,3              <-------------Set window between columns 1 to 3.
/P                 <-------------Display pending line.
  AAAAAAAAAAAAAAA       <-------Pending line displayed.
/U12345/           <-------------Delete first five characters starting
                                 in the first column of the window
                                 over the range specified in the next
                                 command.
/FEOF              <-------------Find EOF.
EOF                <-------------End of file.
/1                 <-------------Make line one pending line and display
                                 it.
  AAAAAAAAAA            <-------Pending line displayed.
/L5                <-------------Display five lines of text.
  AAAAAAAAAA            <-------Changed line displayed (first 5
                                 characters deleted).
  BBBBBBBBBB            <-------Changed line displayed (first 5
                                 characters deleted).
  CCCCCCCCCC            <-------Changed line displayed (first 5
                                 characters deleted).
EOF                <-------------End of file.
```

3) Uxxx/newstring example (replace characters).

```
/W1,3              <-------------Set window between columns 1 to 3.
/P                 <-------------Display pending line.
  AAAAAAAAAAAAAAA      <-------Pending line displayed.
/UAA/1111          <-------------Replace first two characters starting
                                 in the first column of the window
                                 over the range specified in the next
                                 command with the string 1111.
/+                 <-------------Advance pending line one line and
                                 display it.
  BBBBBBBBBBBBBB       <-------New pending line (changed line not
                                 displayed).
/W5,150            <-------------Set window between columns 5 to 150.
/UBB/2222          <-------------Replace first two characters starting
                                 in the first column of the window
                                 over the range specified in the next
                                 command with the string 2222.
/+                 <-------------Advance pending line one line and
                                 display it.
  CCCCCCCCCCCCCC       <-------New pending line (changed line not
                                 displayed).
/1                 <-------------Make line one pending line and display
                                 it.
  1111AAAAAAAAAAAAA    <-------Pending line displayed.
/L5                <-------------Display five lines of text.
  1111AAAAAAAAAAAAA    <-------Changed line displayed (AA replaced
                                 with 1111 starting in column 1).
  BBBB2222BBBBBBBBB    <-------Changed line displayed (BB replaced
                                 with 2222 starting in column 5).
  CCCCCCCCCCCCCC       <-------Unchanged line displayed.
EOF                <-------------End of file.
```

COMMENTS:

The exchange field starts at the first column of the window, but is not limited by the width of the window (note example 2 above).

The exchange fields used in the U command are not tabbed, so that the tab character can be used like any other character in the exchange string.

Only the first occurrence of the current delimiter is recognized. It is assumed to separate the xxx parameter with the new string.

A (ABORT EDIT SESSION)

Used to abort the edit session and leave the original source file unchanged.

A
No parameters required

EXAMPLE:

FILEA contains the following two lines of text:

```
   THIS EXAMPLE DEMONSTRATES
   THE ABORT COMMAND

   :RU,EDITR            <----------FMGR command to call EDITR.
   SOURCE FILE?         <----------EDITR requesting name of file to
                                   be edited.
   /FILEA               <----------File to be edited.
     THIS EXAMPLE DEMONSTRATES  <-First line of file.
   /-2                  <----------Delete two lines of text.
   EOF                  <----------End of file mark indicates all text
                                   has been deleted.
   /A                   <----------Abort edit session leaving original
                                   file unchanged.
   EDITR ABORTED        <----------EDITR response verifying edit session
                                   has been aborted.
   :LI,FILEA            <----------FMGR command to display contents of
                                   FILEA.

FILEA  T=00004 IS ON CR01000 USING 00001 BLKS R=0002

0001  THIS EXAMPLE DEMONSTRATES
0002  THE ABORT COMMAND
```

EC (END EDIT AND CREATE A FILE MANAGER FILE)

Ends the edit session and creates a named File Manager file for storage of the contents of the destination work area.

```
+-------------------------------------------------------------------+
|                                                                   |
| ECnamr                                                            |
|                                                                   |
|-------------------------------------------------------------------|
|                                                                   |
| namr                                                              |
| filename[:security code[:cartridge reference]]                    |
|                                                                   |
| (Refer to Chapter 3 for description of NAMR parameter).           |
+-------------------------------------------------------------------+
```

EXAMPLE:

```
:RU,EDITR              <----------FMGR command to run EDITR.
SOURCE FILE?           <----------EDITR request for name of file to be
                                  edited.
/0                     <----------Put empty file into EDITR's source
                                  work area.
EOF                    <----------End of file.
/ LINE ONE             <----------Insert first line of text.
/ LINE TWO             <----------Insert second line of text.
/EC&FILEA:AA:1000      <----------End edit session and store destination
                                  work area into created file named
                                  &FILEA with security code AA on
                                  cartridge with reference number 1000.
END OF EDIT            <----------EDITR response indicating edit session
                                  is terminated.

:LI,&FILEA:AA:1000     <----------FMGR command to list the contents of
                                  file named &FILEA.


FILEA  T=00004 IS ON CR01000 USING 00001 BLKS R=0002
0001  LINE ONE
0002  LINE TWO
```

ER (REPLACE OLD FILE WITH NEW FILE)

Ends an edit session and replaces the edited FMGR file with the edited version stored in the destination work area.

ER	<----------End edit session and replace disc resident file with edited version.
ERnamr	<----------End edit session and replace disc resident file with edited version.

namr
filename[:security code[:cartridge reference]]

(Refer to Chapter 3 for a description of NAMR parameter).

EXAMPLE:

FILEA contains the following two lines of text:

```
LINE ONE
LINE TWO
```

1) ER example.

```
:RU,EDITR          <----------FMGR command to run EDITR.
SOURCE FILE?       <----------EDITR request for name of file to be
                              edited.
/FILEA             <----------Name of file to be edited.
  LINE ONE         <----------First line of file.
/+                 <----------Advance pending line one line and
                              display it.
  LINE TWO         <----------Pending line displayed.
/ LINE THREE       <----------Insert new line of text.
/ER                <----------End edit session and replace contents
                              of disc resident file with contents of
                              destination work area.
END OF EDIT        <----------EDITR response indicating edit session
                              has terminated.

:LI,FILEA          <----------FMGR command to list the contents of
                              FILEA to terminal CRT.
```

```
FILEA  T=00004 IS ON CR01000 USING 00001 BLKS R=0002

0001  LINE ONE
0002  LINE TWO
0003  LINE THREE
```

2) ERnamr example (Editing a file with a security code).

```
:RU,EDITR            <----------FMGR command to run EDITR.
SOURCE FILE?         <----------EDITR request for name of file to be
                                edited.
/FILEA               <----------Name of file to be edited.
  LINE ONE           <----------First line of file.
/+                   <----------Advance pending line one line and
                                display it.
  LINE TWO           <----------Pending line displayed.
/ LINE THREE         <----------Insert new line of text.
/ERFILEA:SM          <----------End edit session and replace disc file
                                named FILEA with edited version
                                (note that security code, SM, is
                                required).
END OF EDIT          <----------EDITR response indicating edit session
                                has terminated.
```

EDITR IN BATCH ENVIRONMENT

In batch mode, commands to EDITR are supplied as part of the job command file; that is, all EDITR commands must be supplied before the job is begun. Once the job is under way, you can not interact with the computer to change the course of the processing.

All EDITR commands function the same in batch as they do in the interactive mode. You need not supply the EDITR prompt (/) on the input commands.

It is important that you know beforehand what is going to result from the edit commands in the job deck. Any condition which causes an error message causes EDITR to abort since there is no possibility of user intervention to fix the error. You must also insure that EDITR goes through the normal terminate sequence initiated by EL, EC, ER, ELR, or ELC. If an edit deflects this sequence, EDITR aborts.

If EDITR is run in batch mode without spooling, File Manager commands and EDITR commands must be read from an external device. Under spooling, however, EDITR commands can be read from a file if the file appears to be an external device. This is done by setting up an input spool to reference a file name as shown in the examples which follow. The default attributes should always be specified for the input spool.

1) EDITR WITHOUT SPOOLING

```
:RU,EDITR,5          <---------Card Reader defined as LU 5.
(Job deck in card reader)
```

2) EDITR WITH SPOOLED JOB

```
:JO,SMITH
:RU,EDITR,5
    ^
    |                <---------The EDITR commands appear in the job
    |                           stream which may be a file or a card
EDITR commands                  deck, or tape, etc.
    |
    |
    v
ECnamr
:EOJ

:JO,SMITH
:LU,50,COMND         <---------Set input spool to reference file
                               COMND containing the EDITR
                               commands.
:RU,EDITR,50         <---------In this example, EDITR acts as if it
                               were reading from LU 50, but it is
                               actually reading records from COMND
                               through the spool monitor driver.
:EOJ
```

For more information on batch job processing and spooling, refer to the RTE-IVB Batch and Spooling Reference Manual.

EDITR IN A MULTI-TERMINAL ENVIRONMENT

In RTE-IVB, both the Session Monitor and the Multi-Terminal Monitor (MTM) allow any terminal user to operate independently of other users. Several users then can run the same program at the same time if each terminal has a separate copy of the program.

When the user types in the command to run EDITR,

```
:RU,EDITR
```

a program named EDIxx is created and scheduled to run at terminal xx, where xx is the LU of the terminal when operating under the Multi-Terminal Monitor or the session identification number when operating under the Session Monitor. When EDIxx is finished, the ID segment is automatically returned to the system.

EDITR IN A MULTIPOINT ENVIRONMENT

In a multipoint environment, several special considerations apply to EDITR operations. A terminal is defined to be in a multipoint environment if its I/O is being processed through driver DVR07. To determine if a terminal is operating under multipoint, observe the transmit break light on the terminal. If it is blinking intermittently, the terminal is probably a multipoint terminal.

When the EDITR is invoked from a multipoint terminal, several actions are performed for the user.

1. The intrinsic tab function of the terminal is enabled which allows the user to use the TAB key on the terminal and set and clear tab stops with the SET TAB and CLEAR TAB keys on the terminal (the EDITR's tab character (;) remains on).

2. The INSERT CHAR, DELETE CHAR, and CLEAR DSPLY keys on the terminal are enabled for use as editing functions.

3. The Q and O commands, which will be described later in this section, are enabled for character edits.

Functions which do not work in a multipoint environment include:

1. The CNTL and ESC keys do not work in a multipoint environment, therefore, the Q and O commands must be used for character edits.

2. The INSERT LINE and DELETE LINE keys do not work, therefore, the EDITR commands to insert and delete lines must be used.

3. The RETURN key (carriage return) is not used for data transmital in a multipoint environment. The ENTER key is used to transmit text to the multipoint controller.

Furthermore, the characters that the EDITR will process are usually delimited by the left margin on the left and the current cursor position on the right.

Except for the differences cited above, all of the other EDITR commands may be used in a multipoint environment.

CHARACTER EDITS WITH THE Q AND O COMMANDS

The Q and O commands are used for character edits in a multipoint environment. Both commands are used to edit the pending line. The only difference between them is that the Q command edits the pending line, whereas, the O command duplicates the pending line then edits the duplicate. Although only the Q command is discussed throughout the rest of this section, the discussion applies to the O command as well.

When the Q or O command is entered, the pending line is displayed, along with a delimiter (GS) to the left of the line. The delimiter is not part of the text string, but it must be preserved to assure proper operation. The delimiter is represented as a pound sign (#) throughout the rest of this section. The EDITR will position the cursor underneath the first character of the line.

To retain the pending line as it is, immediately hit the ENTER key. For example:

```
/Q
 #ABCDE        <--------------Cursor displayed under first character in
  -                           line. Press the ENTER key and line is
                              retained as is.
/P             <--------------Display pending line.
  ABCDE        <--------------Pending line displayed.
```

DELETE CHARACTERS FROM THE END OF A LINE

To truncate characters from the end of a line, position the cursor immediately after the last character to be retained. Strike the ENTER key to enter all the characters between the left margin and the current cursor position. The intrinsic terminal key CLEAR DSPLY can be used to delete characters at the end of the line from the screen. After using the CLEAR DSPLY key, the ENTER key is used to enter the edited line. For example:

```
/Q             <--------------Enter Q to make character edit.
 #ABCDE        <--------------Pending line is displayed. Cursor is
    -                          moved under the D using the right arrow
                               key (->) to delete D and E from line.
                               CLEAR DSPLY key can then be struck to
                               clear D and E from screen.
  ABC          <--------------Edited line is displayed.
```

INSERT CHARACTERS WITHIN A LINE

To add characters in the middle of a line, the INSERT CHAR key may be used. Press the INSERT CHAR key. The red light above the key should come on. Move the cursor to the position where the characters are to be added, and type in the new characters. Finally, position the cursor at the end of the line and hit the ENTER key. The insert light will go off and the edited line will remain as the pending line. For example:

```
/Q             <--------------Enter Q to make character edit.
 #ABCDE        <--------------Pending line is displayed. Cursor is
   -                           moved under the C using the right
                               arrow key (->). INSERT CHAR key is
                               depressed and XXX is typed in.
ABCXXXDE     <--------------Edited line is displayed.
```

DELETE CHARACTERS WITHIN A LINE

To delete one or more characters, position the cursor under the character to be deleted and press the DELETE CHAR key. The character will be deleted from the display and the rest of the line will be shifted left to fill in the gap. After all of the desired deletions have been made, move the cursor to the end of the line and press the ENTER key. Do not delete the delimiter at the beginning of the line. For example:

```
/Q              <--------------Enter Q to make character edit.
 #ABCDE         <--------------Pending line is displayed. Cursor is
   -                            moved under the B using the right
                                arrow key (->). DELETE CHAR key is
                                depressed three times to delete the B,
                                C, and D. Move the cursor to the end
                                of the line and press ENTER.
 AE             <--------------Edited line is displayed.
```

TAB CONTROL IN MULTIPOINT ENVIRONMENT

When the EDITR is invoked in a multipoint environment, the TAB key on the terminal is enabled (the EDITR tab character remains on). The tab stops are initially set to columns 7 and 21. These may be changed using the SET TAB and CLEAR TAB keys on the terminal. The TAB key may be used to position the cursor at any time. It is equivalent to moving the cursor using the terminal keys with arrows on them.

Chapter 6
Interactive Utilities

The following interactive utility programs, currently available with RTE-IVB, are described in this Chapter:

* Compile Utility (COMPL)
* Relocating Loader (LOADR)
* Compile and Load Utility (CLOAD)
* Interactive Debugging Utility (DBUGR)
* System Status Program (WHZAT)
* Merge Utility (MERGE)
* Disc Cartridge Save/Restore Utilities (WRITT,READT)
* Soft Key Programs (KEYS and KYDMP)
* Track Assignment Table Log Program (LGTAT)
* System Configuration Display Utility (LUPRN)

Compile Utility (COMPL)

This utility allows the user to automatically invoke the appropriate HP Supported Compiler or Assembler for a specified source file. It can be run interactively by first calling the utility:

```
:RU,COMPL
```

then providing the parameters requested by the utility's prompt:

```
NAMR(S),NAMR(L),NAMR(R),<C.S.>
```

or it can be run by specifying all the required information on one line:

```
:RU,COMPL,namr(s)[,namr(l)[,namr(r)[,<c.s.>]]]
                                    [,options file namr]]]
```

Note that only NAMR(S) is required. The default values for the other parameters will depend upon either the format or the file contents of NAMR(S).

The parameters used by COMPL are described below:

NAMR(S) Name of the source file to be compiled or assembled.

NAMR(L) Disc file or logical unit (LU) number to which listing will be directed.

If a spoolable LU (i.e., one set up at GASP initialization) is specified, and Spool exists in the system, and the LU is not an interactive device, then the output will automatically be spooled to that LU. Also, the message:

```
SPOOL FILE = COXXYY
```

will be issued to the user's terminal, where

XX = Session identification number (or terminal LU if operating under Multi-Terminal Monitor instead of Session Monitor).

YY = The number of the COMPL or CLOAD spool file. Any one session is limited to a maximum of 80 COMPL or CLOAD spool files. See the COMPILE AND LOAD UTILITY section in this chapter for a description of the CLOAD utility.

Note that COMPL assumes that SPLCON and JOBFIL reside on the SPOOL DISK. If it is not desired that output be automatically spooled to a spoolable LU, then the LU should be specified as "lu:NS" (for example, 6:NS instead of 6).

DEFAULTS: There are two types of defaults for this parameter, a minus symbol (-) or a null (omitted). If the minus symbol is specified, and the source file name begins with an ampersand (&), the ampersand is replaced with an apostrophe and the remaining source file characters are used for the list file name. The list file is created and stored on the same cartridge as the source file. If NAMR (L) is omitted or the source file name does not begin with an ampersand, the listing goes to the user's terminal.

NAMR(R) Name of file in which relocatable code is to be placed.

DEFAULTS: There are two types of defaults for this parameter, a minus symbol (-) or a null (omitted). Both types of defaults give the same result. If the source file name begins with an ampersand, the ampersand is replaced with a percent sign and the remaining file characters are used for the relocatable file name. The file is created and stored on the same cartridge as the source file. If the source file name does not begin with an ampersand, a file name must be specified, else the relocatable code will not be saved.

<C.S.> Optional control statement which allows user to override control statement in the source file.

DEFAULT: The default for this parameter is to use the control statement contained in the source file. See the appropriate Compiler or Assembler manual for the format of the control statement.

options file namr Name of file containing a control string option list. Usable with PASCAL compiler only.

For a description of the file specification format for a NAMR, see the NAMR PARAMETER section in Chapter 3 of this manual.

EXAMPLES:

1. :RU,COMPL,&PROG <---Note that file name starts with ampersand (&).

Since the source file name starts with an ampersand (&), the listing goes to the user's terminal and the relocatable code is placed in a file named %PROG which is created by the utility.

2. :RU,COMPL,&PROG,-,- <---Note that file name starts with ampersand (&).

Since the source file name starts with an ampersand (&), the listing goes to a disc file named 'PROG and relocatable code is placed into a disc file named %PROG, both disc files are created by the utility.

3. :RU,COMPL,PROG <---Note that file name does not start with ampersand (&).

   Compiles the source file, PROG, listing going to the user's terminal, but no relocatable code file is created.

4. :RU,COMPL,PROG,-,%PRG <---Note that file name does not start with ampersand (&).

   Compiles the source file, PROG. The listing goes to the user's terminal and relocatable code is placed into disc file named %PRG created by utility.

5. :RU,COMPL,PROG,,%PRG

   Same as example 4.

6. :RU,COMPL,PROG,-,%PRG,FTN4,L,A

   Same as example 4, except that the control statement in program PROG is overridden by the control statement specified:

   FTN4,L,A <---Specified control statement overrides control statement in program PROG.

7. :RU,COMPL,&PROG,6,-

   The source file &PROG is compiled. The listing is spooled to LU6 (the line printer) and the binary relocatable code is placed in the disc file %PROG created by the utility.

8. :RU,COMPL,&PROG,6:NS,-

   The source file &PROG is compiled. The listing goes to LU6 (the line printer), but is not spooled. The binary relocatable code is placed in the disc file %PROG created by the utility.

9. :RU,COMPL

   By running COMPL and not specifying a source program, the user enters interactive mode and COMPL prompts to the user's terminal:

   NAMR(S),NAMR(L),NAMR(R),<C.S.>

   The user can then enter the source program to be compiled and whatever other optional parameters desired.

Relocating Loader (LOADR)

The Relocating Loader (LOADR) reads relocatable code from any input device or FMP file, and produces an absolute load module that is ready for execution. The loader automatically sets up the linkage between the program and any required library files. That is, the user does not have to specify library searches during the load process. The program may be relocated as a background, real-time, or large background program and optionally have a debug routine appended.

In addition to its linking functions, the LOADR's command parameter options may also be used to list program names and blank ID segments, purge permanent programs from the system and add or replace permanent programs.

The Relocating Loader has the following features:

* Can be operated under control of the File Manager in batch mode.

* Is swappable and can be operated in either real-time or background disc-resident areas.

* Allows programs declaring COMMON to reference either a system COMMON area (shared with other programs) or a local COMMON area (not shared with other programs).

* Can relocate programs from relocatable files (Type 5 files).

* Can scan and relocate from user library files.

* Allows a program to be permanently added or deleted from the system. Only the loader can be used to purge a permanent program. (The OF, name, 8 command will not remove a permanent program from the system.)

* Can read LOADR commands from a command file or interactively from the user's terminal to control the load process.

* Allows temporary loads into either the real-time or background area for execution with an optional debug routine.

* Allows linking on base page and on the current page.

* Allows a program to reference absolute and code replacement type ENT macros.

* Uses system area disc tracks left vacant by deleted programs.

* Uses a short ID segment when loading a background program segment (when available; see "On-Line Modification" below).

Loader Requirements

The loader is approximately 13 or 14 pages long (including the RTE-IV loader main (~ 15k octal words), the loader library (~ 4k octal words), and necessary system routines). The size of the loader should be increased to 16 to 20 pages to increase the memory available for symbol table use and allow larger programs to be processed.

RU, LOADR Command Options

Parameter options are available in the RU,LOADR statement that permit user specification of the following items:

1. Command file name.
2. File or the logical unit number of the input device for relocatable code.
3. File or the Logical Unit number of the list destination.
4. An operation code that allows Subsystem Global Area (SSGA) flag access together with COMMON type and program type.
5. A program format code that includes temporary loads with DBUGR features.
6. Listing characteristics.
7. Current page linking characteristics.

A detailed description of the RU,LOADR statement is given under Loader operation in this section.

At load time, the user need not know the actual address of the partition in which the program will run because each partition appears to be within the first 32K words of memory. The location at which a program area appears to begin is a logical address, and the program is relocated with respect to this logical address. It is not necessary to declare the partition number that a program will execute in, since a program will run in any partition of the same type large enough to accommodate it.

Program Relocation

During loading, programs are relocated to start at the beginning of the disc-resident program area of logical memory. If COMMON is declared, the program will be preceded by the COMMON area. The logical address of the program location always begins at a page boundary. The first two words of the program location are allocated for saving the contents of the X and Y registers whenever the program is suspended. The next 32 words are used for saving the map if a swap occurs. Once relocated, the program is linked to external references such as EXEC or the Relocatable Library.

Any program segments will overlay the memory area immediately following the main program and its subroutines.

The loader stores the absolute version of the program, its subroutines and linkages on a disc track or a group of contiguous disc tracks and then assigns the disc tracks to the system.

The program, together with its subroutines and its largest segment, may be as large as the largest partition of the same type. If a program is assigned to a partition, it must not be larger than the partition or an L-RQ PGS error results (see Loader Error Messages in Appendix B). COMMON may be allocated in one of several areas according to the needs of the programmer (see the optional parameter list for the RU,LOADR request).

On-Line Modification

The operator can use the loader to permanently modify the set of disc resident programs previously loaded during generation. The loader adds new disc-resident real-time or background programs, and also replaces disc-resident programs with updated versions having the same name. A program to be replaced must have all the following conditions present:

* Must be dormant
* Not currently occupying a partition
* Not in the time list
* Have a zero point of suspension.

The OF,xxxxx,8 operator command deletes disc-resident programs or segments that were loaded temporarily into the system by the loader. The OF command cannot delete programs or segments that were permanently added on-line using the loader, or stored during generation using the On-Line Generator (RT4GN).

The On-Line Generator stores disc-resident programs on disc in an absolute, packed format. Each main program is identified and located by a 33-word ID segment. The ID segments are stored in the ID segment area of the system disc area and brought into main memory when the system is started up. For disc-resident programs, the program's disc location as well as its main memory and base page addresses are kept

in the ID segment. When a main program and segments are loaded, the segments are identified and located by a nine-word short ID segment. When a main program declares an External Memory Area, a three-word ID extension is allocated. See Appendix C for the ID segment and extension format.

RT4GN can create a number of blank 33-word and 9-word ID segments so that the loader can later add new programs and segments to the permanent system. It can also create blank ID extensions. The addition or replacement of a program involves the conversion of relocatable programs into an absolute unit, finding space on the disc to store it, and recording information in the ID segment.

The loader always attempts to use the short ID segment for identifying a program segment. However, a standard 33-word ID segment is used if a short ID segment is not available.

A program declaring an EMA cannot be loaded if an ID extension does not exist for the program.

When replacing a program, the new program may overlay the old program's disc space only if the length of the new program (plus base page linkages) does not exceed the disc space formerly occupied by the previous program. A track or group of tracks is allocated for program storage when adding a program or if space requirements of a replacement program exceed those of the old. These newly allocated tracks are software-protected but not hardware-protected. Memory resident programs can neither be added nor replaced in the system.

When performing an on-line modification, the disc hardware protect must be physically disabled prior to the loading (and then enabled afterwards) unless the protection is always kept disabled. RTE provides additional software protection for any tracks containing system programs or user programs.

When the session user does a temporary load and has set the program priority, the loader checks the user's "PR" capability level. If this level is less than 50, then a warning message is issued, and the priority defaults to 99.

Segmented Programs

Segmented modules can be added and replaced in any order provided that the main program is always entered first. Permanent replacement of a permanent program or main segment programs will not necessarily result in the main and segments being stored on contiguous tracks.

When replacing segmented program modules, the operator must either replace every segment with a new segment having the same name, or else remove the original segments permanently from the system.

Note that a main and all its segments must be relocated at the same time (see "Loading Segmented Programs" later in this section).

Adding New Programs

A new program to be added to the system is stored on a complete disc track or several contiguous tracks. A blank ID segment is allocated to record the program's memory and disc boundaries, name, type, priority, assigned partition, and time values. The loader attempts to use available disc space in the system before allocating new full tracks. If new tracks must be allocated, they are assigned to the system and are software-protected.

Program Replacement

When replacing one program with another, the following sequence of events take place as appropriate to the current conditions:

1. The new program is first relocated onto scratch disc tracks.

2. The new program will use the same ID segment as the old program but will only use the same disc space if the length of the code and base page does not exceed the old program size. Also the disc space must have been set aside originally for programs at generation time.

3. If the new program cannot be fitted into the disc area of the replaced program, the loader then looks for another area of appropriate size if one was previously freed by the user through deleting a program incorporated during generation. In this case, the deleted program's ID segment had its name blanked but its disc space was retained. That disc space is given to the new program.

4. If neither condition exists (items 2 and 3), the scratch tracks on which the new program was generated become system protected and the old ID segment is retained.

Addition or Replacement Limitations

Several limitations may prohibit the final addition or replacement of disc-resident programs:

1. System or reverse COMMON is requested but the program's COMMON length exceeds that of the COMMON area.

2. Local COMMON is requested and COMMON is not declared by the first relocatable module encountered by the loader, even though the module is a dummy module that contains no executable code.

3. The base page linkages exceed the corresponding linkage area for disc-resident programs established by the system during generation.

4. The length of the absolute program unit (including current page linkages) exceeds the area available.

5. Disc space is not available to store the program.

6. A blank ID segment is not available for adding a program (program previously loaded can be deleted to create a blank ID segment) or its segments.

7. An ID extension is not available for adding a program with an EMA.

Program Deletion

A temporary program is deleted from the system with the OF,name,8 command. A permanent program (i.e., a program loaded during generation, or on-line with the loader as a permanent addition or replacement load) is deleted with the loader. When using the loader to delete a permanent program, the opcode parameter is set to PU, which blanks the program's ID segment and makes it available for loading another program.

The tracks containing the program are released unless they are system tracks. If the program had been saved through the File Manager on FMP tracks, the tracks are not released to the system but remain as FMP tracks.

Any time a temporary or permanent program is deleted from the system, all its segments must also be deleted. This is required since segments may have occupied tracks that were released by deleting the main program.

NOTE
Only the LOADR may perform permanent loads or deletes. Copies of LOADR may perform temporary loads but will be aborted with an L-PE LDR error return if the attempt is made to perform permanent loads or purges.

Common Allocations

Three options can be specified when allocating a COMMON area for a program:

SYSTEM COMMON. This implies a background program with COMMON in the background system COMMON area, or a real-time program with COMMON in the real-time COMMON area. System COMMON is established when the system is generated.

LOCAL COMMON. The local COMMON area for a program is established at the beginning of the background program's area. The COMMON area will be swapped together with the program. It is necessary for the first COMMON allocation to be the largest declared. RTE FORTRAN IV named COMMON is handled the same as local COMMON.

REVERSE COMMON. This implies a background program with its COMMON in the real-time COMMON area. Conversely, a real-time program can reference and use the background system COMMON area. Reverse COMMON is established when the system is generated.

Link Allocation

Three options are available for allocating linkages for a program.

Default. The default option forces all linkages to occur on base page.

Current Page. The current page option allocates linkages on the current page whenever possible. A base page link is allocated when a current page link cannot be used. External references go direct (i.e. do not use a link at all) when possible.

Mixed Page. The mixed page option allocates linkages on the current page whenever possible except for external references. EXT linkages are forced to base page. Other linkages are on base page only if current page links were not possible.

Program Types

When a program is assembled or compiled, it may be assigned to a program type that is kept in the NAM record. The type information is used by the On-Line System Generator and, in some cases, by the Relocating Loader. (Refer to the RTE-IVB On-Line Generator Reference Manual for information on program types handled by the generator.)

The Relocating Loader handles Type 6, 7, 8 and 14 modules as though they were normal subroutines (Type 7) to be appended to the program making reference to them. The loader SE command (see below) will relocate these types of modules if an entry point in a module satisfies a previous external reference.

The loader opcodes corresponding to module's NAM types are as follows:

NAM Type	LOADR Opcode
2	RT
3	BG
4	LB
0	BG (default)

where NAM Types:

2,3,4 and 0	are main programs (NAM Type 5 is a program segment)
Type 2 programs	are real-time programs that are relocated with access to Table Area II.
Type 3 programs	are background programs that are relocated with access to Table Area II.
Type 4 programs	are background programs that require a larger logical address space for the program. A larger address space can be acquired, since Table Area II and the System Driver area are not included in the program's address space.

For additional information on program types, see the RTE-IVB Programmer's Reference Manual.

Loader Operation

The loader is scheduled for execution with the RU or ON operator command in the format.

```
RU,LOADR[,command[,input[,list[,opcode[,format[,partition[,size]]]]]]]
```

where:

command — The command file structure can be used for loads when more than one relocatable file is required. The <command> parameter specifies:

1. A command file <namr>.

2. An interactive input device from which commands may be entered. When commands are entered interactively on such a device, a /LOADR: prompt is displayed when the loader is ready for a new command.

3. A non-interactive input device, such as a tape cassette, from which commands may be entered. No prompt is issued by the loader to solicit new commands.

If this and all other parameters are omitted, command entry defaults to the Logical Unit number of the user's terminal. If running under Batch Mode, the default is logical unit 5.

input — The file name of the relocatable main program or the Logical Unit number of the relocatable input. There is no default case.

list — List output device. The default setting is the User's Terminal. In batch mode, the default is Logical Unit 6. Refer to the <opcode> parameter below for list options. The list device is locked for the duration of the load if the LU is not interactive and is not a file.

Alternately, a list file <namr> may be specified. The listing will then go to a file. The file named must not already exist. The loader must create the file. The one exception to this is if the specified file name has an apostrophe as its first character; for example:

```
'name
```

In this case, the loader will create the file if it does not exist, or simply open the file if it does exist.

opcode Mnemonic operation code. The parameter defines the program type, COMMON type, and whether or not the program requires the Subsystem Global Area (SSGA). To determine the operation code mnemonic, select one or more (or none) from each of the following columns:

Program Type	COMMON Type	Load Type
BG	SC	PE
RT	RC	TE
LB	NC	RP
	SS	

where:

BG = Background program
RT = Real-time program
LB = Large background program
SC = System COMMON
RC = Reverse COMMON
NC = No COMMON (or local COMMON)
SS = Use Subsystem Global (SSGA). SS can also be used with other elements in its same column.
PE = Permanent program.
TE = Temporary program.
RP = Replace permanent program (do not also specify PE).

The default setting is BGNCTE.

Note that the program type does not default to the type that is kept in the NAM record, but to type 3 (BG).

The elements of the selected mnemonic code can be specified in any order with no intervening commas or blanks. For example, PEBGSS is interpreted the same as SSBGPE, which specifies a background program using Subsystem Global to be made a permanent program. One, two, or all three parameters can be specified.

Note that the Session user must have a capability level of at least 60 to use the PE, TE, and RP op codes.

format — Mnemonic format code. This actually is an extension of the opcode that was filled. The parameter defines the format for the program load operation. To determine the format code, select one or none from each of the following columns:

DEBUG Append	List Options	Don't Copy	Current Page Linking Options
DB	LE	DC	MP
	NL		CP
			BP

where:

DB = append DBUGR subroutine to the program
LE = list entry points and base page linkages
NL = no listing desired
DC = don't copy. Copies of this program will not be made. This option should be specified when multiple copies of the program are not desired.
MP = Use current page links where possible, except for external references. EXT linkages are forced to base page (mixed page links).
CP = Use current page links where possible, including external references. External references go direct if possible.
BP = Use base page links only. No current page linking done. Default is BP (External forced to use base page link).

Format and opcode parameters may be intermixed and intermingled in any order. For instance, DBBGRT will relocate a real-time program and append the DBUGR to it. Note that a later specification will override an earlier specification.

partition — The specific partition number in which program is to be executed. If not specified, the program will execute in any available partition of sufficient size. This is the same as using the AS operator command.

size — Allows a logical address space larger than the program size. Permits use of a dynamic buffer at the end of the program for use as a data array, symbol table space, etc., when the program requires such space. If the program is an EMA program, the EMA area immediately follows the dynamic buffer area.

The <opcode> and <format> parameter mnemonics can be intermingled in any order. That is, <opcode> mnemonics can be mixed with <format> mnemonics, and vice versa. A comma must be included as a parameter position marker if:

1. The character count within the parameter exceeds six, or

2. Subsequent parameters such as <partition> are to be specified.

The following examples show typical usage of the <opcode> and <format> parameters:

```
*RU,LOADR,PROG1, , ,RTDBSS,NL
          ----- - - ------ --
            ^   ^ ^   ^     ^
            |   | |   |     |------------ <format/opcode> parameter
            |   | |   |------------------ <opcode/format> parameter
            |   | |-------------------- <list output> parameter position
            |   |------------------- <input> parameter position
            |------------------- <command> parameter

*RU,LOADR, , , ,RP, ,7
          - - - -- - -
          ^ ^ ^  ^ ^ ^
          | | |  | | |---------------- <partition> parameter
          | | |  | |------------------ <format/opcode> parameter position
          | | |  |------------------- <opcode/format> parameter
          | | |------------------- <list output> parameter position
          | |------------------- <input> parameter position
          |------------------- <command> parameter position
```

If a track allocation cannot be made for a relocation, the loader displays the message WAITING FOR DISC SPACE. The loader repeats the disc request and is suspended until space becomes available.

Following the relocation of a program that has its external references satisfied, the loader terminates with one of the following messages:

```
ww PAGES RELOCATED  xx PAGES REQ'D  NO PAGES EMA  NO PAGES MSEG
      or
ww PAGES RELOCATED  xx PAGES REQ'D  DEFAULT  EMA  zz PAGES MSEG
      or
ww PAGES RELOCATED  xx PAGES REQ'D  yy PAGES EMA  zz PAGES MSEG
LINKS:rr  PROGRAM:ss  LOAD:tt  COMMON:uuuu vv qq
/LOADR:name READY at HR:MIN AM/PM day, date, year

/LOADR:$END
```

where:

ww = the number of pages occupied by the relocated code (includes base page).

xx = size in pages of the partition required by the program

yy = the EMA size in pages (for EMA programs only)

zz = the MSEG size in pages (for EMA programs only)

rr = CP, MP, or BP for current page, mixed page or base page depending on which link option if set.

ss = program type, RT, BG, or LB

tt = TE or PE for temporary or permanent program respectively. RP for replacing a permanent program.

uuuu = SC, RC, or NC for system common, reverse system common or no system common, followed by SS for subsystem global area (SSGA) or blanks if SSGA not available.

vv = DB if DEBUG appended, blank if not.

qq = FO if force load, blank if not.

name = name of main program. The loader terminates and the program is ready to run.

If a new program is loaded bearing the same name as a main program or a program segment with an ID segment, the following message is displayed:

```
DUPLICATE PROG NAME -<nnnnn>
W-DU PGM
```

where <nnnnn> is the duplicated program name. The loader automatically attempts to create a unique program name by replacing the first two characters of the new program's name with period characters (..). If successful, the loading process continues and when completed, the following messages are displayed:

```
/LOADR: <..nnn> READY
/LOADR: $END
```

where <..nnn> is the modified program name.

If unsuccessful; that is, a program named <..nnn> already exists, the loader is aborted and the appropriate error message is displayed.

Whenever the loader completes a successful or unsuccessful load, it returns five words of information about the load to the program that scheduled it, via the PRTN system subroutine. The returned information can be accessed via RMPAR. See the DOS/RTE Relocatable Library Reference Manual for a description of RMPAR. For example, when the loader is run from the File Manager, FMGR picks up the information in parameters 1P, 2P, 3P (this is also the FMGR 10G), 4P and 5P. A successful load gives the following:

```
    1P,2P,3P = program name
       4P,5P = 4 spaces
```

If an unsuccessful load occurred, the following information would be returned:

```
    1P,2P,3P = 6 character mnemonic error code
          4P = L-
          5P = 0
```

If the load was aborted, the following information is returned:

```
    1P,2P,3P = 77 XXX
          4P = L-
          5P = 0
```

See Chapter 3 for a description of G- and P-type globals.

Additional Opcode Parameters

The loader's <opcode> parameter has two other uses. Entering LI or PU causes the loader to, respectively, list all currently active programs in the system, or purge a permanent program. Opcodes LI and PU may be used in the interactive mode. They are not used in batch mode, entered from a command file, or used during program relocation.

The syntax for the list option is as follows:

```
   RU,LOADR,,,lu,LI
```

In this case, a list of all active programs in the system is transmitted to the specified Logical Unit. The list will include:

1. program name
2. program type
3. priority
4. main program addresses (low and high+1)
5. Base Page addresses (low and high)
6. program size in pages
7. EMA size if EMA is declared
8. MSEG size if EMA is declared
9. partition number if the program is assigned to a partition
10. T if temporary or P is permanent
11. system common type (SC,RC,NC)
12. SSGA access (SS or blank)
13. If a program is loaded under session monitor, the session identifier is also shown.

It is printed as a table in the form:

```
NAME TY PRIOR LMAIN HMAIN LO BP HI BP SZ EMA MSEG PTN TM COM S-ID
```

An alternate form of the request is:

```
RU,LOADR,,PROG,lu#,LI
```

This will list all of the above information only for the program named PROG.

If the opcode is PU, the message

```
/LOADR: PNAME?
```

is output on the assigned Logical Unit device. Entering a program name following the prompt causes the loader to permanently purge the referenced program from the system. Entering a /A will prevent any purge operation and terminate the loader.

An alternate form of the request is:

```
RU,LOADR,,PROG,,PU
```

This will cause the loader to permanently purge the program named PROG from the system.

Loading the Binary Code

The RTE-IV loader will accept binary relocatable code from any FMP file on any disc cartridge accessible to the user. The file <namr> of the main may be included in the RUN statement. If all segments and all subroutines are in the input file <namr>, then no further information is needed. However, segments and subroutines will frequently be in several files throughout the system, and in this case, additional commands to the LOADR are required. The additional commands may be specified through a command file, an interactive or non-interactive Logical Unit. The file <namr> or LU is specified in the first loader RUN parameter.

Loader Command File

The loader will load all relocatable input found in the file specified by the RUN statement. However, subroutines or segments will often be located in other files. In order to facilitate loading of a program broken up in this manner, the loader will take input from a command file. The command file syntax and meaning are described below. Note that only the first two characters of any command are required unless otherwise specified.

SEARCH	Searches the system disc library for undefined externals.

SEARCH,<namr>	Searches the file <namr> for undefined externals. Only the first two characters of this command need be specified for a single-pass search of the named file. If more than two characters are used in the command; that is, SExxxx,NAMR instead of SE,NAMR, the file is searched multiple times to ensure that backward references are satisfied. The SE,NAMR form is faster but will not satisfy backward references.
MSEARCH,<namr>	Searches the file <namr> for undefined externals. The file is searched multiple times to ensure that backward references are satisfied.
RELOCATE,<namr>	Loads file <namr> as part of the program. The <namr> specified may be a program, subroutine or segment.
LOCATE,ADDRESS (LO,XXXXXB)	Changes the load address of the next module to be relocated to the specified address. If the address specified is in octal, a 'B' must be placed after the last digit (i.e., 46000B). This command is useful when it is desirable to align modules to page boundaries to conserve base page links. Moves the relocation base to location XXXXX. Decimal input is assumed; use a 'B' suffix for octal. This command is useful for base page linking problems.
LIBRARY,XXXX	This sets up file XXXX as a library file. Up to 10 files may be specified. These files will be searched automatically at the end of every load and between segments in a segment load.
SL (SEARCH LIB)	Search all files specified in the library command.
TR, <namr> or TR	Go to <namr> for succeeding LOADR commands. The current command file is not saved. If <NAMR> is not specified, return to command file suspended when undefined external was encountered.
FORCE	Force loads a program and/or program segment. Undefined externals will be ignored.

DISPLAY	Causes a list of undefined externals to be printed on the list device, or in the interactive mode, on the interactive command device. Note that the undefined externals listed are those referenced by the module being loaded; that is, undefined externals in the main of a segmented program will not be listed if the current module being relocated is a segment.
ECHO (see footnote)	Causes the input commands from a file to be echoed on the list device as they are encountered. This is useful for debugging loader command files. The command is ignored if the commands are not from a file.
END EXIT /E	End of command input. Signals the loader to exit the command mode and finish up the load. If undefined externals exist at this time, an automatic scan of the system library is performed.
/A AB	Aborts the loader immediately. A clean termination of the load operation is performed.
*	Denotes a comment line when entered as the first character of an entry line. The loader ignores the entire line. Comments may also follow a command and be in the same entry line as the command, providing two commas appear in the line. For example: SE,,SEARCH THE LIBRARY RE,XTABS,LOAD PROGRAM NAMED XTABS DI,,DISPLAY UNSATISFIED EXT REFS
AS,xx (see footnote)	Assigns the relocated program to partition xx.
SZ,<yy> (see footnote)	This command allows the user to request more memory for the program than the actual program code requires. The extra space is called dynamic buffer area. YY is the number of the pages of memory for the program and dynamic buffer area. For EMA programs, the EMA area will immediately follow the dynamic buffer area. Note that this dynamic buffer area may be changed on-line for non-EMA programs with the SZ operator command.
LL,<namr> (see footnote)	Specifies the list Logical Unit number or file name if the listing is to go to a file. If a file name is specified, the file must not already exist unless its name begins with an apostrophe (’).

OP,<opcode> or <opcode> (see footnote)	Specifies an <opcode> parameter, where <opcode> is as defined previously. Note that opcodes LI or PU are illegal in a file, but are legal in the interactive mode. (The OP, <opcode> form is used to avoid confusion between command file commands and operation codes.) When the user in session has a capability level less than 60 and uses the opcodes PE, TE, and RP, an error message is issued and the program is aborted.
FM,<format> or <format> (see footnote)	Specifies a <format> parameter, where <format> is as defined previously.

At the end of every segment load, main load, and at the end of a command file, the system library is searched for undefined externals. If undefined externals still exist and the loader was not initiated from an interactive device, then the undefined externals are listed and the loader aborts.

The loader prints the message:

```
UNDEFINED EXTS
```

The external references are listed, one per line, followed by the error message:

```
L-UN EXT
```

FOOTNOTE:
Specification of these commands must precede specification of any RELOCATE or SEARCH command. Otherwise, the control command is ignored if entered from an interactive device, or causes errors if entered from a file. These commands can be entered either within the RU command or from a command file. Note that RU command parameters are overridden by any commands subsequently entered from a command file. Two exceptions exist. An LI or an PU in the runstring will cause the command file to be ignored.

Note that during the load process, undefined externals are allowed in the main of a segmented program because they might be satisfied in a segment. When the user specified the end of the loading process, the main is then checked for undefined externals. If undefined externals exist, the following warning or error message is issued:

```
MAINS
UNDEFINED EXT
L-UN EXT or W-UN EXT
```

and the loader will suspend and issue a prompt if initiated from an interactive device. If not initiated from an interactive device or if the interactive device times out five times after prompts are issued, the loader will abort unless the FORCE option is in effect. The suspend is to allow the user to do a possible search or scan to satisfy the undefined externals. From this point on, the interactive device will be prompted at the completion of each LOADR command until a TR is entered. A TR will transfer control back to the command file or LU which was being processed when the undefined external occurred. Note that a TR, <namr> is not a legal command from the interactive device in this mode, nor is a RE, <namr>.

The loader will not allow undefined externals in a segment because one segment's entry points may not satisfy another segment's externals. This is because only one segment may be in memory at a given time. The DISPLAY command will list undefined externals. Note that the list refers only to the main or current segment being loaded.

The abort may be prevented by the FORCE command. The FORCE command will force load a program and/or program segment.

Loading From a Logical Unit

Relocatable code from a Logical Unit can be accepted by the RU,LOADR,,<lu> command or interactively with the RELOCATE,<lu> command. If more than one tape is to be mounted for the load, the interactive mode must be used and the RELOCATE,<lu> command reentered for each tape. If mini-cartridge is used, a file cannot be in two cartridges. The loader treats EOT as EOF. In this case, a disc file should be created and then relocated.

Loading Segmented Programs

The loading of segmented programs requires special loader processing. The loading speed of such programs can be increased if the load process is understood and the suggestions given below are followed. Generally, all the relocatable code will be in one file or several files scattered throughout the system.

Assume the following program:

A program has three segments and seven subroutines located in one file, as illustrated in Figure 6-1.

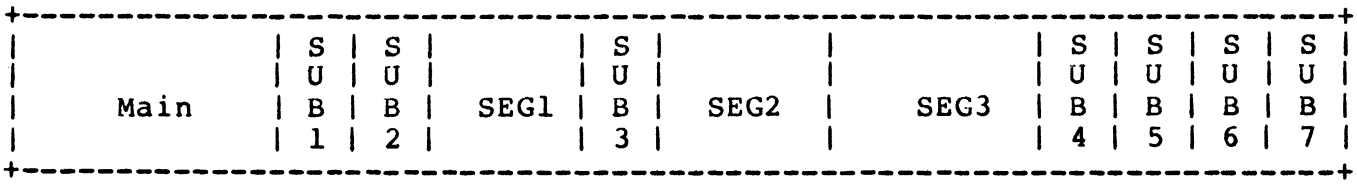


Figure 6-1. Segmented Program Example

The loader would relocate this program as follows:

1. Load MAIN program.

2. Load SUB1 and then SUB2.

3. Search entire file for subroutines required by the MAIN if there are undefined external references.

4. If any subroutines are loaded in Step 3, repeat Step 3 to satisfy backward external references (i.e., assume SUB6 is loaded and it references SUB3).

5. If there any undefined external references, search the system library and relocatable library.

6. If there are still undefined externals, continue loading (they may be satisfied by a segment).

7. Load SEG1.

8. Scan any subroutines following this segment and before next segment for undefined externals (i.e., SUB3) and load them if necessary.

9. If there are undefined externals, search the entire file for referenced subroutines.

10. If any subroutines are loaded in Step 9, repeat Step 9 to satisfy backward external references.

11. If there are undefined external references, search the system and relocatable libraries.

12. If there are still undefined externals, issue a warning message if initiated from an interactive LU, or else abort the load. If a command is entered, process that command and repeat this step. If no command is entered, abort the load. When a TR is entered, continue.

13. Continue Steps 7 through 12 for each segment.

The loading sequence described above has several implications for the user when preparing a segment load:

a. A subroutine called by many segments need only appear once in the file.

b. Subroutines referenced in the MAIN are loaded with the MAIN and are thus sharable by all segments. Subroutines loaded with the MAIN are not loaded with segments.

c. Any subroutines located before the first segment are relocated with the MAIN.

d. Any subroutines located after the first segment in a file are loaded only with those segments that reference them.

What the above basically implies is that subroutines may appear anywhere in the file, even as a library concatenated at the end of a file. This provides optimal loading in terms of program size, but does not necessarily provide optimal loading speed. To optimize loading speed, subroutines that are referenced by a segment should be located directly behind that segment.

When a relocatable program is contained in several files, a command file should be used to load the program. Typically, the MAIN program would be in one file, each segment in a separate file, and perhaps a file of subroutines that are referenced by some of the segments. The command file for loading such a segmented program might consist of the following:

File Entry	Command	Resulting Action
a.	RE,MAIN	Relocates program named MAIN
b.	SE,LIBRY	Searches library named LIBRY
c.	RE,SEG1	Relocates segment named SEG1
d.	SE,LIBRY	Searches library named LIBRY
e.	RE,SEG2	Relocates segment named SEG2
f.	SE,LIBRY	Searches library named LIBRY
.	.	.
.	.	.
.	.	.

When the loader encounters the command in file entry c, it recognizes the program as segmented. Before SEG1 is loaded, LOADR searches the system and relocatable libraries for undefined external references. Undefined externals are still permitted at this point, since they might be satisfied in a segment.

However, at file entry e, undefined externals remaining after the system and relocatable libraries are searched will cause LOADR execution to be suspended or aborted. This is because a segment may not satisfy an undefined external reference through another segment. (The FORCE option may be specified to force load the code and prevent an abort or suspend condition.) Upon completion of the loading process, any remaining undefined external references in the MAIN program would result in the loader being suspended or aborted and display of the following messages:

```
/LOADR: MAINS
/LOADR: UNDEFINED EXTERNALS
/LOADR: <list of MAIN program's undefined externals>
          .
          .
          .
/LOADR: L-UN EXT or W-UN EXT
/LOADR:
```

At this time a command could be entered, otherwise the loader will reprompt up to five times and abort when the terminal times out the fifth time. Note that in the non-interactive case the loader would immediately abort with an L-UN EXT error instead of issuing the prompt.

Reducing Segmented Program Load Time

There are several ways to increase segmented program loading speed. Those described below are suggestive only, and are not to be considered as required procedures:

1. Place any referenced subroutine with the segment that calls it. This eliminates unnecessary file scans in search of a subroutine that will be relocated with a segment.

2. Place subroutines into files in the sequence in which they are called. That is, if SUB1 calls SUB2, place SUB1 in the file before SUB2, etc. For example, assume these subroutines are in a library file to be searched by the loader and that the loader is looking for SUB1. Ideally, the loader would pick up SUB1 and create SUB2 as an undefined external reference. The loader would then continue the file search; if SUB2 was then encountered, it would be picked up on the same pass. However, if SUB2 was located in front of SUB1, an additional file search would then be necessary.

3. If all the relocatable code is within the same file, place the subroutines in the sequence suggested in Item 2.

4. If several segments reference the same subroutine, place that subroutine immediately following the MAIN program. Segments may share subroutines that are loaded together with the MAIN program.

Optimizing the Use of Current Page Linking

Current page links can only be allocated directly in front and directly behind of a module (where a module begins with a NAM record and ends with an END record). An instruction word requiring a link can use a current page link only if the instruction word and link are on the same page.

Large modules, covering several pages, will not be able to use very many current page links. First, links will only be allocated before and after the module. Second, only code on the first and last pages will be able to use those links. All code on the middle pages will have to use base page links.

Smaller modules would have links allocated before and after each module. In the case that a module is less than a page in length, potentially all the code in the module could use links on the current page.

If two modules are small enough to fit on the same page, and they access common externals, they can share the same current page links to the externals.

Relocating two such modules together would result in the following: the first module would be relocated and current page links would be created for the external references. The second module would be relocated and the links created for the first module would be used since they are on the same page and access the required externals.

If modules which reference each other are loaded on the same page, no links at all will be required for those references. The references will be direct with the CP option set.

If current page linking is desired but backwards compatibility is also desired, the MP option provides a mixed linking scheme. Externals are forced to use base page links as in previous RTE systems. However, all other links will use the current page if possible. The MP option may also be useful in the case that both the base page limit and program size limit are critical. The CP option might cause an L-OV MEM error and the BP option might cause an L-OV BSE error where the MP option might load with no errors.

If the added overhead of current page links, namely a larger program size, is to be avoided, the BP option will force all linking to occur on base page. Note that this is not possible if a program needs more links than is available on base page.

DBUGR Library Subroutine

DBUGR is a utility subroutine distributed with the RTE-IVB operating systems. It is appended to the end of a user's program by the loader when the opcode parameter in the RU,LOADR command is DB. DBUGR allows the user to debug a program by means of Trace, Break Point and other features. A summary of DBUGR commands is given in Chapter 6 of this manual. For a detailed description of DBUGR commands, see the RTE-IV Debug Subroutine Reference Manual.

LOADR Error Reporting

All loader errors are reported to the list device. If the loader was initiated from an interactive device which is different than the list device, all errors are also reported to the interactive device. The list device may be specifically declared in the ON or RU scheduling command, or defaulted. The default list device is specified under "LIST" earlier in this section.

The error codes are displayed on the list device in the following form:

```
/LOADR:<error code>
```

For some non-recoverable error conditions, LOADR aborts execution and displays the error report as follows:

```
/LOADR:<error code>
/LOADR:LOADR ABORTED
```

At times, the user may wish to abort a load while the load is going on. Entering a BR,LOADR command will cause the loader to abort a load and perform a clean and orderly termination. This is greatly preferable to using an OF,LOADR command during a load process, which may leave files open.

For some error codes, the name of the program module and the entry point name of the subroutine being relocated are displayed prior to the error code display line, as follows:

```
/LOADR:<module name>
/LOADR:<entry point name>
/LOADR:<error code>
```

Loader Error Codes

All error codes are prefixed by L-characters.

When possible (and appropriate) the module name is printed BEFORE the diagnostic and the entry point name is printed AFTER the module name.

Compile and Load Utility (CLOAD)

This utility is an extension of COMPL (see the COMPILE UTILITY section). It performs all of COMPL's functions, and in addition, schedules LOADR (see the Relocating Loader) to relocate the compiled code.

Only the default loader options are available. See the RELOCATING LOADER description in this section for a discussion of all the options available with LOADR.

The Compile and Load Utility (CLOAD) can be run interactively by first calling the utility:

```
:RU,CLOAD
```

then providing the parameters requested by the utility's prompt:

```
NAMR(S),NAMR(L),NAMR(R),<C.S.>
                        ,options file namr
```

or it can be run by specifying all the required information on one line:

```
:RU,CLOAD,namr(s)[,namr(l)[,namr(r.)[,<c.s.>]]]
                                    [,options file namr]]]
```

The parameters and their default values are the same as those for COMPL except that the list device for CLOAD must be an LU. See the COMPILE UTILITY section for a description of the parameters.

EXAMPLES:

1. `:RU,CLOAD,&PROG` <---Note that file name starts with ampersand (&).

   Since the source file name starts with an ampersand (&), the listings go to the user's terminal, the relocatable code is placed in a file named %PROG, and the LOADR relocates %PROG creating a temporary program.

2. `:RU,CLOAD,&PROG,-,-` <---Note that file name starts with ampersand (&).

   Since the source file name starts with an ampersand (&), the listings go to the user's terminal, relocatable code is placed into a disc file named %PROG, and the LOADR relocates %PROG creating a temporary program.

4. :RU,CLOAD,PROG,-,%PRG <---Note that file name does not start with ampersand (&).

   The listings go to the user's terminal, relocatable code is placed into a disc file named %PRG, and the LOADR relocates %PRG creating a temporary program.

5. :RU,CLOAD,PROG,,%PRG

   Same as example 4.

6. :RU,CLOAD,PROG,-,%PRG,FTN4,L,A

   Same as example 4, except that the control statement in program PROG is overridden by the control statement specified:

   FTN4,L,A <---Specified control statement overrides control statement in program PROG.

7. :RU,CLOAD,&PROG,6,-

   The listings are spooled to LU 6 (the line printer), the binary relocatable code is placed in the disc file %PROG, and the LOADR relocates %PROG creating a temporary program.

8. :RU,CLOAD,&PROG,6:NS,-

   The listings go to LU 6 (the line printer), but are not spooled. The binary relocatable code is placed in the disc file %PROG, and the LOADR relocates %PROG creating a temporary program.

9. :RU,CLOAD

   By running CLOAD and not specifying a source program, the user enters interactive mode and CLOAD prompts to the user's terminal:

   NAMR(S),NAMR(L),NAMR(R),<C.S.>

   The user can then enter the source program and specify whatever optional parameters are desired.

Interactive Debugging (DBUGR)

DBUGR is a Hewlett-Packard utility subroutine used to interactively check programs for logical errors during execution. Using DBUGR, the user may examine and modify memory, examine and modify registers, set breakpoints and trace instruction execution. Multipoint terminals using DVR07 will work using DBUGR with use of the \\U command. In the following discussion, only the most frequently used DBUGR functions are described; refer to the RTE-IV Debug Subroutine Reference Manual for the complete range of DBUGR capabilities.

Calling DBUGR

DBUGR can be automatically appended to a program at load time by calling the LOADR with the following command parameters:

```
:RU,LOADR,,filename,,DB
```

where DB instructs the LOADR to append DBUGR onto the relocatable code in file filename. Refer to the LOADR section in this manual for more information on the LOADR parameters. This command will also handle segmented programs, though there are some special procedures involving breakpoints in segmented programs. These are explained in the section on breakpoints.

When a program with appended DBUGR is subsequently run with the command:

```
:RU,program
```

DBUGR will be entered and the user will be able to give any legal DBUGR command. DBUGR calls the system subroutine LOGLU to obtain the logical unit from which the program was scheduled. It then uses this logical unit for all I/O.

DBUGR is also callable from Assembly Language and FORTRAN programs. The Assembly Language calling sequence is:

```
     NAM prog
     EXT DBUGR
      .
      .
      .
     JSB DBUGR              call to DBUGR
     DEF RTN                address of return point
     DEF LU                 optional pointer to LU number

RTN  -return point-
      .
      .
LU   BSS 1                  interactive LU DBUGR will use for I/O
```

The FORTRAN calling sequence is:

```
CALL DBUGR(LU)

    or

CALL DBUGR
```

according to whether the optional LU is passed in as a parameter.

In either Assembly Language or FORTRAN, if the optional LU is not passed in, DBUGR calls the system library subroutine LOGLU to determine the interactive LU to use for I/O. LOGLU returns to DBUGR the LU number of the user's interactive log device. If none exists, LU number 1 is returned specifying that the system console is to be used, when operating in a non-session environment, and the session terminal is used when operating in a session environment.

Entering DBUGR

When DBUGR is entered, it prints the following message on the appropriate LU:

```
START DBUGR
```

The user is now conversing with DBUGR and any legal command may be entered.

All DBUGR operations are conducted at the assembly language level. A load map and a mixed Assembly Language listing of the program is essential. A mixed Assembly Language listing of the program is also necessary if debugging a program written in a high level language.

DBUGR Commands

The following paragraphs give a concise explanation of the main features of DBUGR. Throughout these paragraphs, the conventions described in Table 6-1 apply. DBUGR supports the RUBOUT key but not the backspace key for deleting a typing mistake.

Table 6-1. DBUGR Command Conventions

SYMBOL	MEANING
\	Escape key (altmode key)
_	current position of the cursor
[CR]	carriage return
[LF]	line feed (control-J on some terminals) automatically advances to next memory location
italics	words and numbers to be supplied by the user

DBUGR Modes

DBUGR operates in one of four modes - symbolic, constant,ASCII, or address. DBUGR uses symbolic mode when it is first entered.

In symbolic mode, the contents of memory are inverse-assembled and displayed as an opcode and a memory reference (if it is a memory reference instruction). The user types "escape S" to enter symbolic mode as follows:

```
\S   -
```

In constant mode, the contents of memory are displayed as octal constants. The user types "escape C" to enter constant mode as follows:

```
\C   -
```

In ASCII mode, the contents of memory are displayed as two ASCII characters. The user types "escape H" to enter ASCII mode as follows:

```
\H   -
```

In address mode, the contents of memory are displayed as an offset to a previously defined label. DBUGR will use any label that precedes the contents by less than octal 11, or any single character label otherwise. The user types "escape A" to enter address mode as follows:

```
\A   -
```

When DBUGR is in a particular mode, the mode can be temporarily switched when examining a memory location. The contents of the memory location will then be immediately displayed again in the temporary mode. With the cursor still on the displayed line of the memory location being examined, type one of the following symbols to temporarily enter the particular mode desired:

!	exclamation point - temporary symbolic mode
=	equals sign - temporary constant mode
'	single quote - temporary ASCII mode
-	underscore - temporary address mode

Expressions and Terms

Expressions are used to specify memory locations to be examined. An expression consists of one or more terms combined with operators as in the following example:

```
AA+10
```

A term may be a previously defined symbol, a number, or certain symbols preceded by an escape key (denoted in the text by a reverse slash (\)). The following examples are all terms:

```
ABC
SYMBOL
-32768
1005
\Q
```

Legal operators are the following:

+	plus operator
blank	alternate plus operator
-	subtract operator
,	comma - inclusive OR

Examine Memory

To examine the contents of a memory location, simply type in an expression that evaluates to the memory location to be examined followed by a delimiting slash (/). For example, one way to examine memory location 50234 is:

```
50232+2/
```

DBUGR will print out on the same line the contents of the specified memory location in either octal or symbolic form. The example above might display:

```
50232+2/  LDA 50277       -
```

informing the user that location 50234 contains an LDA instruction referencing memory location 50277.

To examine the next sequential memory location, press the line feed (LF) key or control J. Continuing the above example, an LF is used to display the contents of memory location 50235:

```
50232+2/  LDA 50277       [LF]
50235/    ADA 50400       -
```

Modify Memory

To modify the contents of a memory location, the user must first open the memory location by examining it. After DBUGR displays the contents of the memory location, it is ready to insert new contents into the memory location examined. If an assembly language instruction is now typed in, DBUGR will assemble it and insert it into the memory location. If an octal constant is entered, DBUGR will insert it directly into the memory location. For example, to modify the contents of location 50234:

```
50234/  LDA 50277       CCA[CR]  Display location 50234, change to
                                 CCA instruction

50234/  CCA             [LF]     Display new contents of 50234, use
                                 line feed to examine 50235

50235/  ADA 50400       100[CR]  Change contents of 50235 to 100 octal

50235/  100             _        Display new contents of location 50235
```

Examine Registers

The A and B registers are addressed as memory locations 0 and 1, respectively. The overflow register, the extend register, and the X and Y registers require special procedures for examination.

Memory location EOREG may be thought of as containing the overflow register and the extend register, each of which is one bit in length. These bits may be examined by typing "EOREG/" as follows:

```
EOREG/
```

DBUGR will respond on the same line with an octal digit between 0 and 3 that is the status word. This octal digit may be broken down into two binary bits (EO) which are interpreted as follows:

```
E (bit 1 of EOREG) = 0 extend register is clear
                     1 extend register is set

O (bit 0 of EOREG) = 0 overflow register is clear
                     1 overflow register is set
```

The user may modify these bits immediately after examining them by typing in the new octal digit to replace the status word.

Memory locations XREG and YREG may be thought of as containing the X and Y registers. The X-register may be examined by typing "XREG/" as follows:

```
XREG/
```

The Y-register may be examined by typing "YREG/" as follows:

```
YREG/
```

DBUGR prints out the contents of the X or Y register on the same line. They may then be modified if desired. Note that the X and Y registers are a full 16 bits wide. For example:

```
0/          10          [CR]  user types 0/ to examine A-register

1/          10          [LF]  user types 1/ to examine B-register
                              the LF causes the next memory location
                              to be displayed automatically.

EOREG/       3          2[LF] user clears the overflow register

XREG/      677          0[LF] examine and clear the X-register

YREG/       50         -1[CR] change the Y-register from octal 50
                              to 177777 (two's complement of -1)
```

Setting a Label

DBUGR can reference memory locations relative to a label. A label consists of one to six alphanumeric characters, the first of which must be alphabetic. To equate a label to a particular memory location, the user must first examine the memory location. After DBUGR has displayed the contents of the memory location, the label is entered followed by a colon (:). DBUGR then equates the label with the examined address. For example, the label S is equated with memory location 50234 as follows:

```
50234/  LDA 50277          S: [CR]
```

Location 50237 may now be referenced by typing:

```
S+3/
```

Execute Program

To proceed with execution of the user program when DBUGR has control, the user types "escape P":

```
\P
```

Upon initial entry to DBUGR, execution proceeds at the transfer address of the program. When a breakpoint is encountered, execution resumes at the instruction where the breakpoint was set.

When proceeding from a breakpoint, the user has the option of typing:

```
n\P
```

DBUGR will then execute breakpoints octal n times before it will break.

If the proceed instruction is given and there is no breakpoint in the program, DBUGR displays the following message before control returns to the executing program:

```
END DBUGR
```

The user may instruct DBUGR where to resume execution of the program by typing the address of the instruction to be executed, followed by "escape G". For example, to resume program execution at location 50234, type:

```
50234\G
```

Breakpoints

When an instruction with a breakpoint is encountered, control is transferred to DBUGR immediately prior to the execution of the instruction with the breakpoint. DBUGR displays information about the state of the machine, and the user may then enter any legal DBUGR command.

A breakpoint is set at an address by entering the octal address followed by "escape B". For example, to set a breakpoint at 50234, type:

```
50234\B
```

Up to ten breakpoints are normally allowed.

A breakpoint that has been set is cleared by typing "escape B" at the beginning of a line:

```
\B
0              50234
ENTER INDEX OF BP TO DELETE, A TO END 0[CR]
ENTER INDEX OF BP TO DELETE, A TO END A
```

0 is the index of the breakpoint at address 50234.

If the executing program reaches a breakpoint, control returns to DBUGR. DBUGR then displays the following information about the state of the machine:

```
ADDRESS(INSTRUCTION) A-REG B-REG X-REG Y-REG EOREG -
```

where:

ADDRESS is the address of the breakpoint

INSTRUCTION is the contents of the ADDRESS

A-REG,B-REG,X-REG,Y-REG are the contents of the registers

EOREG is the status of the extend and overflow bits as explained in the section on examining registers

For example:

```
50234\B                                   set breakpoint at 50234

\P                                        proceed with execution

50234(LDA 50277)  77  11  177776  3  3       \P
       breakpoint information displayed, user types \P to proceed

50234(LDA 50277)  77   0  177776  3  3       [CR]
       breakpoint encountered again; B-REG has changed to 0

1/      0               11[CR]            change B-REG to octal 11

\P                                        proceed
```

When a segmented program has been loaded with the command:

```
:RU,LOADR,,filename,,DB
```

use the following commands to control the setting of breakpoints within segments:

```
["A]\B      break at entry to all segments

["N]\B      break at entry to no segments

[seg]\B     break at entry to segment
```

To set a breakpoint within a segment, enter the following command:

```
addr[seg]\B
```

where:

addr is the address within the segment at which the breakpoint is set.

seg is the name of the segment in which the breakpoint is set.

The breakpoint will be set when the segment is loaded into memory.

When a segment entry load break occurs, DBUGR will break at the start of the new segment and print the following message:

```
SEGMENT  seg   BREAK
  --BREAKPOINT INFORMATION--
```

where:

seg is the name of the new segment

BREAKPOINT INFORMATION is the normal breakpoint information

DBUGR does not check the validity of the segment name. The segment name may not begin with the two characters quote A ("A) or quote N ("N). This is to avoid confusion in setting the breaks in segment entry points as explained above.

DBUGR will not allow breakpoints below the memory protect fence or outside the user's partition. An attempt to set such a breakpoint will cause a memory protect ("MP?") or a dynamic mapping ("DM?") error message to be printed.

There are certain legal instructions that DBUGR cannot execute without causing memory protect (MP) or dynamic mapping (DM) errors. The instruction "JSB $LIBR" is one example. When such a situation arises, DBUGR will not allow execution of the instruction, and prints out a message of "DM?" or "MP?" depending on the error that execution of the instruction would cause. To execute the instruction, simply move the breakpoint and proceed. An exception is "JSB EXEC" which will trace like a multiword instruction.

Tracing

When DBUGR has control, the instructions of a program can be traced (single-stepped) by typing "escape T". After each instruction is executed, the same information about the state of the machine will be displayed as after a breakpoint. For example:

```
50234\B                          set a breakpoint at 50234

\P                               proceed

50234(LDA 50277)  77  11  177776  3  7       \T
       breakpoint information displayed, start trace

50235(ADA 50101) 100  11  177776  3  7       \T
       breakpoint information displayed, continue trace

50236(LDB 50282) 107  11  177776  3  7       -
                                 .
                                 .
                                 .
```

A specified number of instructions can also be traced by specifying an octal number before the trace command. Type:

n\T

to trace octal n instructions and halt.

When DBUGR attempts to trace an instruction that will cause a memory protect or dynamic mapping violation, an "MP?" or "DM?" error will be printed. If the instruction is legal, put a breakpoint on the instruction to which control will return and then proceed.

Note: Privileged routines (see the RTE-IVB Programmer's Reference Manual) cannot be traced.

DBUGR Example

The following example demonstrates a typical session with DBUGR.

```
:RU,PROG     (Execute program loaded with DBUGR.)

START DBUGR

["A]/B                              break on all segment entries.

16002/  CCA      M:                 examine location 16002 in the main
                                    program; equate 16002 to M.
23456/  NOP      S:                 examine location 23456 in the
                                    segment; equate 23456 to S.
S+5\B                               use escape B to set a breakpoint

[SEG1]\B                            set a segment entry breakpoint

\P                                  proceed.


SEGMENT SEG1 BREAK                  a break is executed upon entry
S   (0)   17542 5608 17702 22 6     to the segment

S+5[SEG2]\B                         set a breakpoint within SEG2

\P                                  proceed

SEGMENT SEG2 BREAK                  break at S+5 in SEG2.
S+5  (0) 17542 5606 45 22  4

M+50\B                              set a breakpoint within the main

S+10[SEG4]\B                        set a breakpoint in SEG4

\P                                  proceed

M+50 (LDA M+700)  0 2234 54 72 5 break in main

M+700/  ALF,ALF  =  1727   1777[LF]  examine location M+700, temporary
                                     octal display,change contents
                                     to 1777
M+701/  0     [CR]                  this location automatically displayed

M+700/ ALF,CLE,SLA,ALF [CR]         re-examine location M+700
```

```
2\T                                  trace two instructions

M+50 (LDA M+700) 0 2234 54 72 5      breakpoint instruction is executed

M+51 (STA M+701) 1777 2234 54 72 5   \P   next instruction is
                                     executed; proceed with execution

[SEG4]\B                             set a segment entry breakpoint
                                     for SEG4.

SEGMENT SEG4 BREAK                   break upon entry to the segment
S (0) 17445 5562 7422 3322 5

\P

S+10 (JSB 112,I) 24 0 177777 55 6 Break at S+10 in SEG4.

["N]\B                               clear segment breakpoint except for
                                     those explicitly requested

\P                                   proceed

END DBUGR
```

System Status Program (WHZAT)

WHZAT is a Hewlett-Packard utility program that provides current system environment information. It can be used to display:

* All scheduled and suspended programs and their status.
* The status of all partitions in numeric sequence.
* Only those programs associated with the user session (default mode).

WHZAT must be either a Type 1 (with or without Table Area II), 2, 3, or 4 program. It can be run outside the session environment:

```
:RU,WHZAT[,lu[,option]]
```

The default mode and parameters are the same as in the WH command description in Chapter 3.

The information displayed is preceded by a heading which includes the current system time and column headings. Figure 6-2 contains a sample printout of the program status mode with the AL option and Figure 6-3 shows the partition status mode (PA option) printout.

WHZAT Output

When you run WHZAT, your program may be listed in one of the following states:

0 dormant
1 scheduled
2 I/O suspended
3 suspended in general wait list
4 suspended waiting for system available memory
5 suspended waiting for disc tracks
6 either operator or program suspended

Refer to Appendix G of the RTE-IVB Programmer's Reference Manual for a detailed description of program states.

In the case of the AL or SM option, the WHZAT output consists of three distinct blocks. The first block shows the user session programs. The second block shows state 3 programs having father-son relationship. The calling (father) programs are identified with two stars. The third block consists of all other individual programs.

All locked LU's and EQT's are displayed at the end as follows:

```
LOCKED LU'S  (PROG NAME)    xx(program name),--,xx(program name)
LOCKED EQT'S (PROG NAME)    xx(program name),--,xx(program name)
```

If no LU or EQT is locked, the corresponding message is suppressed. Also displayed at the end is the maximum contiguous free tracks available at that instant of time. The corresponding LU number is also displayed. For example:

```
LOCKED LU'S (PROG NAME)     6(SPOUT),
MAX CONT. FREE TRKS :       6,  LU   3
```

The following information is displayed for state-4 programs (memory suspended).

```
MAX CONT. SAM AVAIL:          xxx
TOTAL SAM SVAILABLE:          xxx
MAX CONT. SAM EVER AVAIL:     xxx
```

The program has to go privileged to compute the above information. Therefore, this information is displayed ONLY if a program is memory suspended.

The program counter value is displayed for every program irrespective of the state. The message P:xxxxx is displayed under the PRG CNTR column. If a program is swapped out, SWP will appear at the end of the octal number (i.e., P:xxxxxSWP).

If a program is in state 2 (I/O suspended) and the EQT is down, the following message appears:

```
2,EQ:xxx,DN,AV:x,ST:xxx
```

```
wnere       xxx is the EQT number in decimal
          AV:x is the availability code (driver independent)
             0 available
             1 disabled
             2 busy
             3 waiting for DMA
        ST:xxx is the status (xxx in octal) of the EQT in operation
```

However, if a LU is down, the message is:

```
2,LU:xxx DN,EQ:xx,AV:x,ST:xxx
```

In a real time situation, the following condition might occur. PRGX invokes PRGY and suspends itself until PRGY terminates. PRGY at the mean time invokes PRGX, thus a deadly embrace situation occurs(DEADLOCK) condition is created. In this case, the following message would appear:

```
PRGX            *   *        3,PRGY
PRGY            .   .        3,PRGX'S QUEUE
**  BLOCK  **
**           *  DEADLOCK  **
**           *  SEE ABOVE FOR REPORT ON PRGX  **
```

As WHZAT executes, the state of the system changes constantly. A case might occur in which the status of a program is reported and in the meantime that program was invoked by another program. The status of that might be reported again as the son of the calling program. For example:

```
PRGA          *                  3,PRGX
PRGX          .  1
 |
 |
PRGB          **                 3,PRGX
**          * SEE ABOVE FOR REPORT ON PRGX  **
```

When a program is in state 3 (general wait state), WHZAT will display a message giving the reason for the wait. These messages and the reasons are listed in Table 6-2. (All examples refer to programs displayed in Figure 6-2.)

Alternate WHZAT Output

The SM program status mode is similar to the AL mode output. The partition status mode output is shown in Figure 6-3; it provides a dynamic map of the activity in each partition and the partition size.

Table 6-2. General Wait State Messages

MESSAGE	REASON FOR WAIT
LULK lu,LKPRG=progx	The listed program attempted to put a lock on logical unit lu. Program progx already has a lock on lu. The listed program will be rescheduled when progx removes its lock.
RN xx,LKPRG=progx	The listed program attempted to set resource number xx. Program progx already has a lock on the resource number. The listed program will be rescheduled when progx removes the lock.
RESOURCE	The listed program attempted to allocate a resource number. The system has no more resource numbers available. The operating system will reschedule the listed program when a resource number is available.
CLASS #	The listed program requested a class number but the system has no more available. The operating system will reschedule the listed program when a class number becomes available.
CL xx	The listed program is waiting on completion of a class GET to class number xx.
progx	The listed program scheduled progx with wait. The listed program will be rescheduled when progx completes.
progx'S QUEUE	The listed program scheduled progx on the queue with wait. progx is not dormant so the listed program must wait. The listed program will be rescheduled after the scheduling of progx completes.
BL,EQT xx	Buffer limit exceeded on the controller in EQT entry xx.
EQLK xxx,LKPRG = PRGA	Program suspended for a locked EQT
EQLK TABLE FULL	Program attempts to lock an EQT and the EQT table is full.

```
+-----------------------------------------------------------------------+
| 15:18:34:130                                                          |
|-----------------------------------------------------------------------|
|  PRGRM T PRIOR PT SZ DO.SC.IO.WT.ME.DS.OP.      .PRG CNTR.  .NEXT TIME.|
|-----------------------------------------------------------------------|
|**FMG76 3 00090 33 10 * * * *  3,..ZAT  * * * * * P:46274              |
|  ..ZAT 3 00001 22  4 .  1, . . . . . . . . . . . P:40306              |
|**FATHR 3 00099 36  2 * * * *  3,SON1   * * * * * P:40073SWP           |
|  SON1  3 00099 36  2 . . . .  3,SON2   . . . . . P:40073SWP           |
|  SON2  3 00099 36  2 . . . .  3,SON3   . . . . . P:90073SWP           |
|  SON3  3 00099 36  . . . . .  3,FATHR'S QUEUE  . P:40067SWP           |
|** BLOCK **                                                            |
|********** DEAD LOCK **                                                |
|    *** SEE ABOVE FOR REPORT ON FATHER                                 |
|.                                                                      |
|  LOCKB 3 00099 21  4 . . . .  3,RN  018,LKPRG=LOCKA                   |
|                                                    P:40075SWP         |
|** BLOCK **                                                            |
|  LOCKA 3 00099 19  4 . . . .  3,RN  017,LKPRG=LOCKB                   |
|                                                    P:40075SWP         |
|** BLOCK **                                                            |
|********** DEAD LOCK **                                                |
|    *** SEE ABOVE FOR REPORT ON LOCKB                                  |
|  PLOCK 3 00099  1A 2 . . . . . . . . . 6,  . . . P:40074              |
|  DUMMY 3 00099  1A 2 .  1, . . . . . . . . . . . P:00000              |
|.                                                                      |
|**FMG88 3 00090 26 10 * * * *  3,EDI88  * * * * * P:46274              |
|  EDI88 3 00050 21 17 . . . 2,EQ:  5,AV:2,ST:002  P:55226              |
|.                                                                      |
|  SPOUT 1 00011  0  . .  1, . . . . . . . . . . . P:44652              |
|  UPLIN 1 00003  0  . 0,  . . . . . . . . . . . . P:00000   15:18:38:380|
|  GRPM  1 00004  0  . . . . .  3,CL  060  . . . . P:47607              |
|  RTRY  1 00020  0  . . . . .  3,CL  059  . . . . P:50376              |
|  R$PN$ 2 00001  5  4 . . . .  3,CL  061  . . . . P:40062              |
|  LOGON 3 00050 37 11 . . . .  3,CL  062  . . . . P:42137              |
|  LGOFF 3 00090 27  9 . . . .  3,CL  063  . . . . P:41133              |
|  RFAM  3 00030 22  8 . . . .  3,CL  053  . . . . P:53723SWP           |
|  EXECM 3 00030 40  3 . . . .  3,CL  054  . . . . P:40054              |
|  OPERM 3 00030 29  2 . . . .  3,CL  052  . . . . P:41132              |
|  RTOPM 3 00030 39  2 . . . .  3,CL  055  . . . . P:50052              |
|  EXECW 3 00030 16  2 . . . .  3,CL  056  . . . . P:41460SWP           |
|  DLIST 3 00030 15  3 . . . .  3,CL  057  . . . . P:43457SWP           |
|  PROGL 3 00030 35  5 . . . .  3,CL  051  . . . . P:40065              |
|  QCLM  3 00028 28  2 . . . .  3,CL  058  . . . . P:40045              |
|  FMG87 3 00090 34 10 . . . 2,EQ: 10,AV:2,ST:002  P:56206              |
|  LPTUP 3 00010 38  6 0,  . . . . . . . . . . . . P:40216   15:18:42:250|
|-----------------------------------------------------------------------|
|ALL LU'S OK                                                            |
|ALL EQT'S OK                                                           |
|LOCKED LU'S (PROG NAME)   6(SPOUT),                                    |
|MAX CONT. FREE TRKS :     6,  LU   3                                   |
|-----------------------------------------------------------------------|
|15:18:34:660                                                           |
+-----------------------------------------------------------------------+
```

Figure 6-2. Program Status Mode (AL) Output

```
6:40:24:530

PTN#  SIZE    PAGES    BG/RT PRGRM
 1     17     47-  63   RT   PLOCK
 2     10     64-  73   RT   D.RTR
 3      6     74-  79   RT   SMP
 4      6     80-  85   RT   <NONE>
 5      4     86-  89   RT   R$PN$
 6      4     90   93   RT   <NONE>
 7      2     94-  95   RT   PRMPT
 8      4     96-  99   RT   <NONE>
 9      2    100- 101   RT   <NONE>
10M   120    102- 221 BG     <NONE>
11S    28    102- 129 BG     LOGON
12S    28    130- 157 BG     <NONE>
13S    28    158- 185 BG     LOCKA
14S    28    186- 213 BG     LOCKB
15S     4    214- 217 BG     DLIST
16S     4    218- 221 BG     EXECW
17M   120    222- 341 BG     <NONE>
18S    28    222- 249 BG     <NONE>
19S    28    250- 277 BG     <NONE>
20S    28    278- 305 BG     <NONE>
21S    28    306- 333 BG     <NONE>
22S     8    334- 341 BG     RFAM
23M   120    342- 461 BG     <NONE>
24S    28    342- 369 BG     <NONE>
25S    28    370- 397 BG     <NONE>
26S    28    398- 425 BG     <NONE>
27S    28    426- 453 BG     <NONE>
28S     2    454- 455 BG     QCLM
29S     6    456- 461 BG     OPERM
30M   120    462- 581 BG     <NONE>
31S    28    462- 489 BG     <NONE>
32S    28    490- 517 BG     ..ZAT
33S    28    518- 545 BG     FATHR
34S    28    546- 573 BG     SON1
35S     8    574- 581 BG     PROGL
36     28    582- 609 BG     <NONE>
37     11    610- 620 BG     FMG89
38     11    621- 631 BG     LPTUP
39      4    632- 635 BG     PTOPM
40      4    636- 639 BG     EXECM

6:40:24:930
```

Figure 6-3. Partition Status Mode Output.

WHZAT display abbreviations are shown below:

Program mode:

PRGRM	program name
T	program type
PRIOR	program priority
PT	partition number; 0 means memory-resident
SZ	page size of program; ** means memory-resident
DO	dormant (state 0)
SC	scheduled (state 1)
IO	I/O suspended (state 2)
WT	general wait state (state 3)
ME	memory suspended (state 4)
DS	disc suspended (state 5)
OP	operator suspended (state 6)
PRG CNTR	point of suspension; number shown is octal
NEXT TIME	time program is listed on time list
A	after the partition number means the program was assigned to the partition
E	after the program's type means it is an EMA program
B	after the program's priority means the program is running under batch

Partition mode:

M	mother partition
C	subpartition in chain mode
S	subpartition available
R	reserved partition
PTN#	partition number
SIZE	page size of program
PAGES	physical pages where program resides
BG	background program
RT	real-time program
PRGRM	program name

Merge Utility (MERGE)

The MERGE program allows the user to concatenate files by either storing them in an existing file by overlaying the file or storing them in a file created by MERGE; or transfer files to a logical unit (LU). The names of the files to be concatenated or transferred can be provided interactively from the user's terminal, from a command file on disc, or from a logical unit such as a minicartridge or mag tape.

The format for running MERGE is:

```
RU,MERGE [,namr(s) [,namr(d) ]]
```

where:

namr(s) is the command file or logical unit from which the names of the files to be concatenated or stored will be supplied. If only namr(d) is specified, namr(s) will be defaulted to the the user's terminal. If both namr(d) and namr(s) are defaulted, MERGE will prompt the user for these parameters. (See Chapter 3 for a description of the NAMR parameter.)

namr(d) is the destination file or logical unit to which the files to be concatenated or stored will be sent. If an existing file is specified, it will be overlayed by the concatenated files; if the file specified does not exist, MERGE will create it and store the concatenated files within it. If an LU is specified, MERGE will transfer the files specified to that LU.

If both namr(s) and namr(d) are not specified, MERGE will prompt the user for these parameters:

ENTER DESTINATION NAMR	<---MERGE prompt requesting name of destination file or logical unit to which the files to be concatenated or stored are to be sent.
ENTER COMMAND NAMR	<---MERGE prompt requesting name of command file or LU which contains the names of the files to be concatenated or stored.

If the command namr is specified to be the user's terminal, MERGE will prompt the user for the names of the files to be concatenated or transferred:

```
ENTER NAMR                    <---MERGE prompt requesting name of
                                  file to be concatenated or
                                  transferred.
```

After each entry to the ENTER NAMR prompt, MERGE returns ready to receive more input with another display of the ENTER NAMR prompt. When the user has completed entering the names of the files, a "/E" is entered to tell MERGE that input has completed:

```
ENTER NAMR /E
```

If the command namr is specified to be a command file, MERGE will search for and execute the command file. An example of a command file might be:

```
FILEA:AA:1000,MAIN DATA BANK
FILEB::1000,PURCHASING DATA
FILEC::1000,SHIPPING DATA
```

Note that it is not necessary to specify "/E" in the command file. The end-of-file mark (EOF) indicates to MERGE that input is completed. Also, comments may be added by inserting a comma after the file name.

After each file is written into the destination namr, a zero length record is written as a file delimiter. Subfile marks within a file are saved on concatenation, and an EOF will be written on completion.

If the user is concatenating files by using a command file, the process may be halted before its natural completion by entering break-mode (i.e., striking any key on the terminal while there is no read pending on the terminal) and entering the BR command. MERGE will stop just before reading another file name.

Also, when the file names are provided through a command file, if the file is not found the error is reported (FMGR-006) but the program continues on to read the next file name and continues file concatenation.

An example of appending files to an existing file (without destroying the existing file) might be:

```
RU,MERGE
ENTER DESTINATION NAMR  FILEA:AA:1000
                        -------------
ENTER COMMAND NAMR 1
                   -
ENTER NAMR  FILEA:AA:1000
            -------------
ENTER NAMR  FILEB::1000
            ------------
ENTER NAMR  FILEC::1000
            ------------
ENTER NAMR  /E
            --
```

Note that the destination NAMR and the first NAMR to be concatenated are the same in this case.

Disc Cartridge Save/Restore Utilities (WRITT,READT)

The Disc Cartridge Save and Restore Utilities provide means for the user to backup disc cartridges on magnetic tape. Utility WRITT saves FMP cartridges on magnetic tape. Utility READT restores disc cartridges from magnetic tape. WRITT and READT error messages appear in Appendix B of this manual.

Save Disc Cartridge (WRITT)

The format for running WRITT is:

```
[RU,]WRITT[,sdisc[,MT:lu[,IH[,DC[,VE[,"..."]]]]]]
```

where:

sdisc is the disc cartridge to be saved. This parameter can be either a negative LU number or a positive CRN number. The defaults are defined below.

NOTE The following optional parameters may be specified in any sequence. The order is not important.

MT:lu is the LU number of the magnetic tape unit on which the cartridge is to be saved. The default is LU 8. This parameter also may be given as the positive or negative LU number only, without the MT: prefix, but it must then be given as the second parameter in the run string.

IH inhibits tape rewind. The default is to rewind the tape before and after each cartridge restoration.

DC disables overlay checking. Unless it is disabled by this command, WRITT checks to see if a previously-stored cartridge file will be overlaid by the current operation.

VE enables mag tape verification by comparing the data on the tape with the data on the disc.

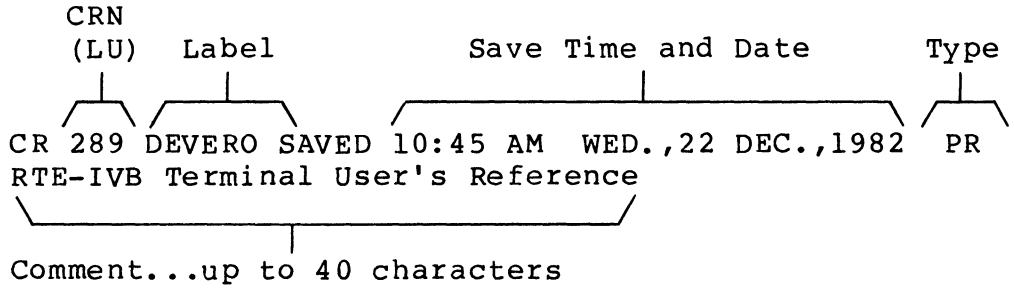
"..." is a comment to be added at the tape header, and displayed by WRITT during the overlay check. The comment may be up to 40 characters long.

If the source disc parameter is defaulted, and you are running under the Session Monitor, WRITT saves the first private or group cartridge mounted to your session. If the source disc parameter is defaulted in a non-session environment, WRITT saves the first cartridge in the cartridge list. In either environment, if there are only system cartridges mounted, Writt issues the message:

```
ONLY THE SYS MNGR MAY SAVE SYSTEM DISCS
/WRITT:STOP
```

and exits.

When a cartridge is saved on tape, WRITT creates a file header to identify the cartridge, as in:



WRITT automatically rewinds the tape before and after a cartridge is saved, unless the IH parameter is specified to inhibit the rewind. If the end of the tape is reached before the entire cartridge is saved, WRITT rewinds the tape and issued the message:

```
Please mount another tape
After mounting enter "GO"
```

and waits for you to mount the tape. WRITT increments the tape number and writes the number to the continuation tape, then completes the save operation.

CAUTION

> Do not attempt to replace a previously-saved cartridge file on a tape, as this can corupt the remaining files on the tape.

Multiple saves to a single tape should be performed in a series of WRITT operations, to save the cartridges sequentially on the tape. If you want to add a cartridge save to an existing tape, use the FMGR CN command to position the tape past the last file. Use the IH parameter to inhibit tape rewind before and after each save. With the rewind inhibited, you must use a CN command (CN,lu,RW) to rewind the tape after the last save operation.

..2 Example

In the following example, group cartridge 32754 (LU 39) is saved to tape. The magnetic tape unit is LU 9, and the comment "RTE-6/VM Relocatables" is added to the header.

```
:RU,WRITT,32754,DC,VE,MT:9,"RTE-6/VM Relocatables"
CR 32754  HPDISK SAVED   1:19 PM  WED., 22 DEC., 1982  GR
RTE-6/VM Relocatables
From LU     39
/WRITT: Verifying tape

/WRITT: STOP
:
```

In the following example, WRITT is invoked to save private cartridge GU to tape. The DC parameter is not specified, so WRITT scans the to see if an existing cartridge save file will be overlaid. If so, it issues a message identifying the save file that will be overlaid and the file that will overlay it, and asks if it is OK to continue. If the response is yes, it will overlay the old file. If it is no, as in this example, WRITT exits without saving the cartridge.

```
:RU,WRITT,GU,VE,MT:9,"Group Utilities CRN, Latest sort."

****** CAUTION *********
Do you want to
 overlay
CR 32754  HPDISK SAVED   1:19 PM  WED., 22 DEC., 1982  GR
RTE-6/VM Relocatables
   with
CR    GU   UTILS SAVED   1:24 PM  WED., 22 DEC., 1982  PR
Group Utilities CRN, Latest sort.
(Yes or No)?
no

*** DISC NOT SAVED ***


/WRITT: STOP
:
```

Restore Disc Cartridge (READT)

The format for running READT is:

RU,READT [,DISC[,MT[,type[,size[,IH]]]]] where:

DISC is the disc cartridge where the saved FMP cartridge is to be restored. This parameter can be either a negative LU number or a positive CRN number.

Default Conditions:

1. Session Environment.

   READT will restore the saved disc cartridge to the specified CRN if it is mounted to that user. If it is not mounted, READT will mount a disc from the spare cartridge pool and assign the specified CRN before restoring the saved disc cartridge.

   If the negative LU number is specified, READT will restore the saved disc cartridge to that LU only if it is mounted to the user. If not, READT stops and a message is displayed.

2. Non-Session environment:

   This parameter must be a negative LU number. The CRN number is not allowed.

MT is the logical unit number of the magnetic tape unit. You can specify either a "+" or "-" LU number. Default is LU 8.

type is an optional session environment parameter specifying the type of cartridge to be restored. Enter P for a private cartridge or G for a group cartridge. Default is to use the type specified in the tape header. If the size parameter is used, the position of this parameter must be maintained.

size is the desired size of the disc to which the magnetic tape contents are to be restored. The size is specified in number of tracks. Default is the size of the disc saved on tape.

IH is used to inhibit tape rewind before and after restoring a cartridge. Default is to rewind tape before and after the disc cartridge restoration. This parameter can be used with the CN command to advance tape to the desired position, e.g., CN,8,FF to advance tape one file. Tape rewind must be done separately with the "CN,8,RW" command.

If it is necessary to overlay a currently mounted cartridge, consider the following items:

1. The last track of the cartridge to be overlaid cannot be moved.

2. The cartridge type (P or G) of the cartridge to be overwritten cannot be changed.

After the cartridge has been restored from tape to disc, the header created by WRITT and the LU to which the cartridge has been restored is displayed on the terminal.

If there is no disc cartridge large enough for the cartridge to be restored and if by moving the file directory the saved cartridge can be restored to an available cartridge, the following message will be displayed:

```
CRN ###### WAS SAVED FROM A XXXXX TRACK DISC
LAST TRACK USED IS          YYYYY
WOULD LIKE TO RESTORE TO A  ZZZZZ TRACK DISC
IS IT OKAY TO MOVE DIRECTORY TRACKS (YES OR NO)?
```

A YES answer will allow READT to restore the saved cartridge to the available disc. A NO answer will terminate READT.

The size parameter specifies the desired number of tracks required to restore the cartridge to. For example, a request is made for a 203 track LU, READT will obtain a 203 track disc LU and place the first directory track at track 202.

In the event that a restore is made to a disc LU that has a sector/track value different from the sector/track value on magnetic tape, READT issues a message indicating that reformatting of the directory is necessary to maintain the integrity of the file structure:

```
TRACKS REFORMATTED FROM XXX SEC/TRK TO YYY SEC/TRK
```

READT will then proceed to restore the cartridge.

To restore FMP tracks on the system and auxilary disc (LU's 2 and 3), the user must be the system manager. The desired LU must be free of any activity, files may not be opened nor may there be ID segments pointing to files in the FMP track area. The first activity will cause a READ 009 error. The second case will cause READT to issue a message informing the user to "OF" the ID segments pointing to the FMP tracks. READT will not allow changing the starting location of FMP tracks or restoring a cartridge of a different sector/track value. Doing so will produce an error. READT errors are explained in Appendix B.

EXAMPLES:

1. A user attempts to restore a cartridge with the same CRN as one already mounted to his session.

```
:RU,READT
```

```
DUPLICATE CRN LABEL OR LU ALREADY MOUNTED
DO YOU WANT TO OVERLAY CRN 289 ON LU 53
WITH CRN 289 (YES OR NO)?
```

A YES response will restore CRN 289 from magnetic tape to LU 53. A NO response will terminate READT.

2. A user attempts to restore a cartridge to LU 53 which is already mounted in his session.

```
:RU,READT,-53

DISC ALREADY MOUNTED
DO YOU WANT TO OVERLAY LU 53
WITH CRN 289 (YES OR NO)?
```

A YES response will restore CRN 289 from magnetic tape to LU 53. A NO response will terminate READT and LU 53 is unaffected.

3. Restore a disc cartridge and change the CRN by specifying another CRN:

```
:RU,READT,1000

CRN 289 MANUF SAVED 10:45 AM FRI.,19 JAN.,1980 PR
CRN 289 HAS BEEN CHANGED TO CRN 1000

/READT:  STOP
:
```

4. When restoring LU 2 or 3, if there are ID segments pointing to FMP tracks on the desired disc LU, READT will issue the following:

```
CR  2  SYS  SAVED 4:00 PM FRI., 4 FEB 1980

THE FOLLOWING PROGRAMS HAVE ID SEGMENTS
POINTING TO FMP TRACKS  YOU'RE REPLACING.
THESE PROGRAMS MUST BE REMOVED BEFORE READT WILL
REPLACE THESE TRACKS.

FTN4
RT4GN
MERGE

/READT:  STOP
:
```

Soft Key Programs (KEYS and KYDMP)

There are two utility programs that are used in conjunction with the 2645A/2648A Display Station. They are:

1. KEYS - a program that provides a simple operator interface for generating command sets in a standard format that will program the 2645A/2648A soft keys.

2. KYDMP - a program that provides the capability of outputting a soft key command set, created by the KEYS program, from a disc file or a 2645A/2648A mini-cartridge file or LU to an 2645A/2648A Display Station to program its soft keys.

Keys — Soft Key Programming Utility

The 2645A/2648A Data Station has eight programmable soft keys, f1 through f8. The utility KEYS provides a simple operator interface for generation of soft key command sets on-line in a standard format. These soft key command sets contain all the information necessary for programming the eight 2645A/2648A soft keys when output to the 2645A/2648A.

When run, KEYS can perform the following functions:

1. Create a new soft key command set,

2. Modify a command set that exists in a file or a logical unit number,

3. List the eight soft key labels, types and command strings of the current command set or one that exists in a file or a logical unit number,

4. Output the current soft key command set or one that exists in a file or a logical unit number to a 2645A/2648A to program its soft keys,

5. Output a soft key command set to 1) a disc or mini-cartridge, or 2) a 2645A/2648A CTU logical unit number to save it.

A soft key label can be specified by the operator for each of the eight soft keys. These soft key labels are a standard part of each soft key command set created by the KEYS utility. When a soft key command set is output to a 2645A/2648A, in addition to programming the soft keys, it displays these labels in the top five lines of the screen according to the format in Figure 6-4. This label field is protected from being written over and is provided as a quick reference to the status of the soft keys.

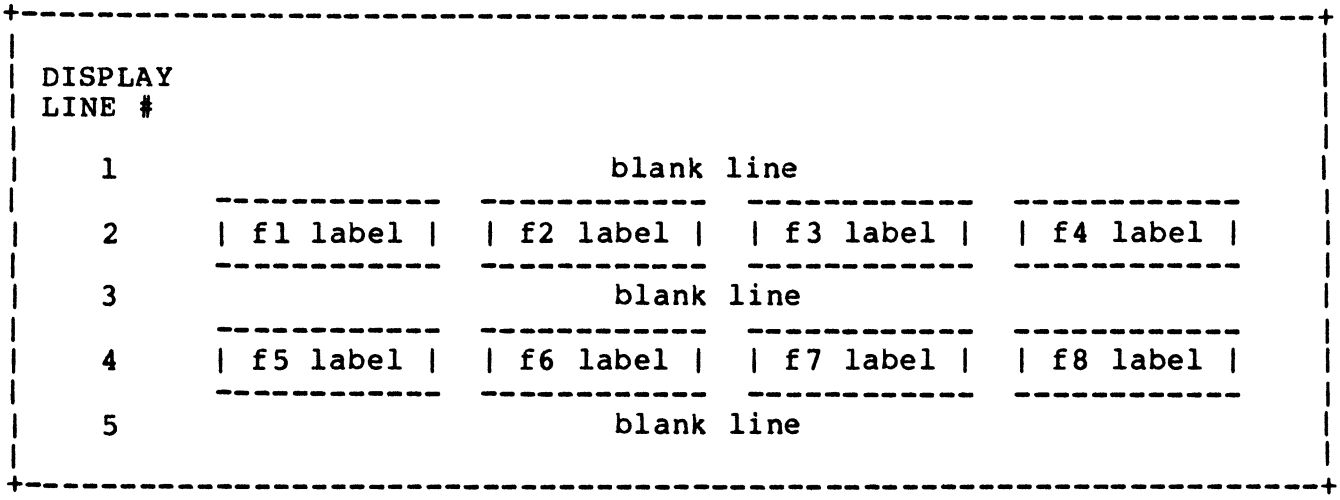


Figure 6-4. Soft Key Label Display Format

There are eight label fields, each sixteen characters long, in which the soft key labels will be written in inverse video (black letters on white background). These labels will be approximately centered in their fields. If no labels are specified when creating a command set, the label fields will be blank. Each time the soft keys are reprogrammed the label field will be updated and protected.

Initializing Keys

KEYS can be run with the following command:

```
RU,KEYS,console,list
```

Where:

console
is the logical unit number of the 2645A/2648A Display Station from which operator responses are to be made. Default value is LU1. The default value will occur if no console parameter or an invalid console parameter (i.e. an lu greater than 63 or less than 1) is passed.

list
is the logical unit number of the device to which soft key command set listings are to be output. Default value is the 'console' specified previously. Default will occur if no list parameter or an invalid list parameter (i.e. an lu greater than 63 or less than 1) is passed.

When KEYS is run with valid parameters, it will print the following request:

```
ENTER ONE OF THESE FUNCTIONS:[CREATE,MODIFY,OUTPUT,LIST]
OR PRESS [RETURN] TO TERMINATE KEYS
```

KEYS can be aborted by typing in A in response to any program request. If an invalid response is made to a program request, the request will be repeated.

Provided below are operating instructions for:

1. creating a soft key command set,
2. modifying an existing soft key command set,
3. outputting a soft key command set, or
4. listing a soft key command set.

Creating a Soft Key Command Set

If the operator wishes to create a new soft key command set, enter "C" in response to the program request shown in the section INITIALIZING KEYS.

KEYS will request the function key whose assignment is to be made:

```
ENTER [SOFT KEY NUMBER (1-8)] TO BE PROGRAMMED OR
PRESS [RETURN] IF LAST ASSIGNMENT HAS BEEN MADE:
```

Enter any soft key number between 1 and 8 or press RETURN if the last soft key has been programmed.

KEYS will print the following heading if a valid soft key number was entered:

```
SOFT KEY ASSIGNMENT FOR FUNCTION KEY X
```

where `X' is the soft key number entered above.

KEYS will request the soft key label to be assigned to this soft key:

```
ENTER UP TO [16 CHARACTERS] FOR SOFT KEY LABEL OR
PRESS [RETURN] IF NO LABEL IS TO BE ASSIGNED:
```

Enter a descriptive label of up to sixteen characters or press RETURN if no label is to be assigned to this soft key.

KEYS will request the soft key command string type:

```
ENTER: [0] FOR NORMAL OR [2] FOR TRANSMIT ONLY
COMMAND STRING TYPE:
```

Enter a 0 to create a `normal' soft key command string. When the soft key containing a type 0 command string is pressed, the command string is output but no carriage return (CR)/line feed (LF) is generated. This is useful when optional parameters are to be provided by the operator. For example a soft key could contain: RU,FTN4. When depressed this would be transmitted and the operator could add optional parameters such as ,2,1,3,99 CR.

Enter a 2 or press RETURN to create a type 2 command string. When the soft key containing a type 2 command string is pressed, a CR is automatically generated after the last character of the string is output.

KEYS will request a soft key command string to be assigned to this soft key:

```
ENTER [UP TO 80 CHARACTERS] FOR SOFT KEY COMMAND
STRING TO BE ASSIGNED TO THIS KEY OR PRESS [RETURN]
TO DEFAULT TO STANDARD COMMAND STRING:
```

Enter a command string of up to eighty characters or press RETURN to assign the appropriate default command string from the following list to the soft key:

Soft Key Number	Default Command String
1	ESCp
2	ESCq
3	ESCr
4	ESCs
5	ESCt
6	ESCu
7	ESCv
8	ESCw

When the last soft key assignment has been made and RETURN is pressed, KEYS will ask the operator what he wants to do next. Typically, after a soft key command set is created it would be checked by LISTING it and then OUTPUT to a file to save it. Also it might be OUTPUT to a 2645A/2648A to program its soft keys. Refer to the appropriate paragraph listed at the end of the section on INITIALIZING KEYS.

Modifying a Soft Key Command Set

If the operator wishes to modify a soft key command set that resides in standard format in a disc file or on mini-cartridge, enter an `M' in response to the program request shown in this section under INITIALIZING KEYS.

KEYS will request the file name or 2645A/2648A CTU logical unit where the command set to be modified resides:

```
ENTER [FILE NAME,SECURITY CODE,CARTRIDGE] OR [2645A LU]
WHERE SOFT KEY COMMAND SET TO BE MODIFIED IS STORED OR
PRESS [RETURN] TO CONTINUE MODIFYING A COMMAND SET IN THIS
PROGRAM:
```

If the old soft key command set resides in a disc file, enter the file name and optionally the security code and cartridge separated by commas.

If the command set resides on a mini-cartridge, enter the logical unit number of the 2645A/2648A CTU where the cartridge is loaded (be sure to position the cartridge at the appropriate file prior to entering the LU number).

Press RETURN to continue modifying a soft key command set that has already been created or loaded previously by KEYS. This feature is convenient in correcting errors in command sets.

KEYS will read the old file (if file name or LU specified) and proceed to request the changes desired as described in this section under CREATING A SOFT KEY COMMAND SET.

Outputting a Soft Key Command Set

If the operator wishes to output a soft key command set to a file or LU, enter the letter `O´ in response to the program request shown in this section under INITIALIZING KEYS.

KEYS will request where the soft key command set to be output is stored:

```
ENTER [FILE NAME,SECURITY CODE,CARTRIDGE], OR [2645A LU]
WHERE SOFT KEY COMMAND SET TO BE OUTPUT IS STORED OR
PRESS [RETURN] TO OUTPUT DIRECTLY FROM THIS PROGRAM
```

If the command set to be output is stored in a disc file, enter the file name and optionally the security code and cartridge separated by commas.

If the command set to be output is stored on a mini-cartridge, enter the logical unit number of the 2645A/2648A CTU where the cartridge is loaded (be sure to position the cartridge at the appropriate file prior to entering the LU number).

A command set can be output directly from the KEYS program by pressing RETURN.

KEYS will read the file if a file name or LU number is specified. It will then request where this soft key command set is to be output:

```
ENTER [FILE NAME,SECURITY CODE,CARTRIDGE] OR [2645A LU]
TO WHICH COMMAND SET IS TO BE OUTPUT OR [RETURN] TO
REPLACE ORIGINAL FILE OR LU:
```

If the command set is to be output to a new disc file, enter the file name and optionally the security code and cartridge separated by commas.

If the command set is to be output to a mini-cartridge, enter the LU number of the 2645A/2648A CTU where the cartridge is loaded (be sure to position the cartridge at the appropriate file prior to entering the LU number).

Pressing RETURN will only replace the last command set that was previously loaded by KEYS from a file or LU.

KEYS will output the command set as specified and proceed to request the next function to be performed as described in this section under INITIALIZING KEYS.

Listing a Soft Key Command Set

If the operator wishes to list the soft key labels, types and command strings of a command set, enter `L' in response to the program request in this section under INITIALIZING KEYS.

KEYS will request where the soft key command set to be listed resides:

```
ENTER [FILE NAME,SECURITY CODE,CARTRIDGE] OR [2645A LU]
WHERE SOFT KEY COMMAND TO BE LISTED IS STORED OR
PRESS [RETURN] TO LIST DIRECTLY FROM THIS PROGRAM:
```

If the command set to be listed is stored in a disc file, enter the file name and optionally the security code and cartridge separated by commas.

If the command set to be listed is stored on a mini-cartridge, enter the logical unit number of the 2645A/2648A CTU where the cartridge is loaded (be sure to position the cartridge at the appropriate file prior to entering the LU number).

If a command set has been previously loaded by KEYS, it can be listed directly from KEYS by pressing RETURN. This feature is useful for checking for errors or recalling the contents of a command set.

KEYS will read the file, if a file name or LU number is specified, and then list the soft key label types and command strings on the list device specified at turn-on.

These will be listed in the format described in Figure 6-5.

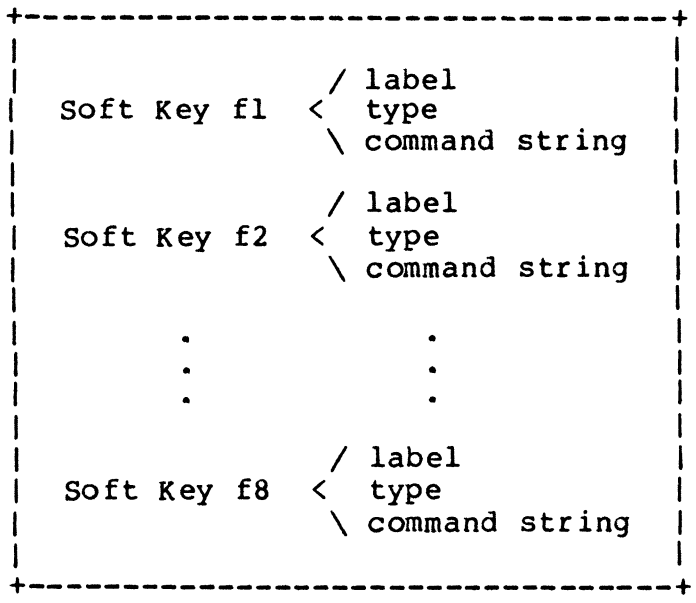


Figure 6-5. Soft Key Command Set Listing Format

Key Program Messages

The following messages can occur when running the KEYS utility. It is indicated in the meaning of each message whether it is an information only message or an error message.

	MESSAGE	MEANING
1.	KEYS HAS BEEN ABORTED!	Information only: Keys has been aborted due to operator typing in A in response to program request.
2.	FILE MANAGER ERROR -XX HAS OCCURRED	Error message: an error has occurred when trying to create, open, read, write or close a file.
3.	ERROR IN READING COMMAND SET FROM LU!	Error message: the number of words read from a command set does not conform to standard format or device error has occurred.

4.	NO ORIGINAL FILE OR LU EXISTS	Error message: an attempt has been made to output a command set to an original file or LU that was never read in by KEYS.
5.	END KEYS	Information only: normal program completion.

Guidelines

The following are some guidelines for using the KEYS utility and the soft key command sets.

1. When you are storing a soft key command set on a mini-cartridge, store it after the last file; otherwise you may overwrite the succeeding file.

2. To examine the contents of a soft key command set file, it is recommended that you use the listing capability of the KEYS utility rather than the LI command of FMGR.

3. Once the 2645A/2648A soft keys have been programmed, use only the standard keyboard keys to get the RTE prompt. Do not use the soft keys as it will hang up the terminal.

4. Pressing RESET once (soft reset) on the 2645A/2648A will not affect the soft keys. However, pressing RESET twice (full reset) will reset the soft keys to their default values (ESCp through ESCw).

Soft Key Command Set Examples

A soft keys command set example is provided below on worksheet forms.

SAMPLE SOFT KEY COMMAND SET WORKSHEET

FILE NAME	SECURITY CODE	CARTRIDGE
SFTKY1	-2	

SOFT KEY	LABEL (MAX. OF 16 CHARACTERS)	TYPE (0/2)	COMMAND STRING (MAX. OF 80 CHARACTERS)	SUBSET PAGE
f1	RUN FMGR	2	RU,FMGR	
f2	RUN EDITR	2	RU,EDITR	
f3	RUN COMPL	0	RU,COMPL,	
f4	RUN LOADR	0	RU,LOADR,	
f5	RUN WHZAT (PROG.)	2	WH	
f6	RUN WHZAT (PART.)	2	WH,,1	
f7	RUN KEYS	2	RU,KEYS	
f8				

Soft Key Command Set Worksheet

If you set up a heirarchy of soft key files, plan the arrangement using worksheets similar to the above sample before you create the soft key files.

1. Enter the file name where the command set is to be stored. If it is a disc file enter the security code and cartridge if desired.

2. Enter up to sixteen (16) characters for each soft key label. These labels will be displayed at the top of the 2645A/2648A screen when the command set is output to the terminal.

3. Enter the soft key command string type. Enter 0 for Normal or 2 for Transmit Only.

4. Enter up to eighty (80) characters for each soft key command string.

5. Next to each soft key command string that programs another soft key command set, enter the page number that describes this soft key command subset.

KYDMP — Soft Key Command Set Dump

To load the terminal soft keys from a file created by KEYS, use the FMGR command DU:

```
:DU,MYKEYS,1
```

where MYKEYS is the name of the file created by KEYS, and 1 is the session LU of your terminal. You can also use the KYDMP program to load the softkey definitions programmatically:

```
:RU,KYDMP,,MY,KE,YS
```

where the douple comma defaults to LU 1. A file security code can be included in the run string, as described in the following subsection.

Running KYDMP

Once KYDMP has been loaded into the system, it can be run from the RTE operating system or from FMGR. Use one of the following command sequences to run KYDMP from RTE or FMGR:

1. To output a soft key command stored in a disc or mini-cartridge file to a 2645A/2648A Display Station use:

   ```
   RU,KYDMP,[console],fi,[ln],[am][,security code]
   ```

2. To output a soft key command set stored in a mini-cartridge file to a 2645A/2648A Display Station use (be sure to position the mini-cartridge at the appropriate file first):

   ```
   RU,KYDMP,[console],ctu
   ```

The parameters in the above command sequences are defined as follows:

console - the logical unit number of the 2645A/2648A to which the soft key command set is to be output. Default value is LU1. Default value occurs if no parameter or an invalid LU number (i.e. an LU greater than 63 or less than 1) is passed.

fi,[ln],[am] - three pairs of ASCII characters defining the file name where the command set to be output is stored. The commas must be included even if the second and/or third ASCII pair is not specified.

Whenever the file name is separated into character pairs in the run string, each pair must contain at least one non-numeric character, or a non-numeric character must be appended to the pair. The following

examples show some typical file names and their run string divisions:

file name	runstring
ABCDEF	:RU,KYDMP,,AB,CD,EF
ABC45F	:RU,KYDMP,,AB,C4,5F
AB34E6	:RU,KYDMP,,AB,34Z,E6
A23456	:RU,KYDMP,,A2,34X,56Y

If the file name is shorter than five characters, and a security code follows in the runstring, include a place-holding comma for each missing character pair.

security code - optional security code of file.

ctu - logical unit number of the 2645A/2648A Cartridge Tape Unit (CTU) where command set to be output is stored.

When KYDMP is run with valid parameters, it will read the soft key command set from the specified source and output it to the specified 2645A/2648A Display Station.

KYDM Program Error Messages

The following error messages can occur when running the KYDMP utility. If an error occurs KYDMP aborts and must be rerun.

NO SECOND PARAMETER SPECIFIED OR NEGATIVE

File name omitted, or a negative number entered where the name belongs

FMGR ERROR -XX WHEN READING FROM FILE

For error -6 (file not found), check the character pairs in the runstring, make sure that if a pair of numeric characters appears, that it is accompanied by a non-numeric character, and and that if there are fewer than five characters in the filename, and the file has a security code, that you have inserted place- holding commas for the missing characters.

ERROR WHEN READING FROM 2645A CTU

Usually, the mini-cartridge is not inserted, it is positioned at the wrong file, or the CTU is down.

Track Assignment Table Log Program (LGTAT)

LGTAT is a Hewlett-Packard utility program designed to display information about the system and auxiliary disc subchannel tracks. The user specifies the logical unit (lu) to which the output will be directed, and whether an abbreviated or complete form of the output should be displayed.

The abbreviated LGTAT output displays the total number of available tracks on the system and auxiliary disc subchannels, and the number of tracks in the largest contiguous track block.

The complete LGTAT output displays the Track Assignment Table for the system and auxiliary disc subchannels, the location of the start of the logical source tracks, plus the abbreviated information described above.

Running LGTAT

LGTAT is run using a command of the following format:

```
RU,LGTAT[,lu[,form]]
```

where:

lu is the logical unit to which LGTAT will direct its output

form = 0 to specify the abbreviated form
= 1 to specify the complete form

If the output lu is unspecified, it defaults to the lu from which LGTAT was scheduled.

If the form is unspecified, it defaults to 0, specifying the abbreviated form.

Abbreviated LGTAT Output

If the abbreviated form is specified, LGTAT will output the following information:

```
TOTAL AVAILABLE TRACKS= xxx
LARGEST CONTIGUOUS TRACK BLOCK= yyy
```

where xxx and yyy will be filled in with appropriate numbers.

Complete LGTAT Output

If the complete form is specified, LGTAT will output a complete listing of the track assignment table for the system disc subchannel and, if present, the auxiliary disc subchannel; the information output in the abbreviated form; plus the location of the start of the Logical Source (LS) tracks. If the LS tracks are currently undefined then LGTAT's output will so inform the user.

The Track Assignment Table will be listed as an nx10 array, where n will be automatically adjusted by LGTAT to print out all the tracks on the associated disc subchannel (see Figure 6-6). Each entry in the array corresponds to one track. The meanings of the various entries are explained in Table 6-3.

LGTAT dynamically reports swapped tracks. Large EMA programs that are swapped may not always appear to be contiguous as is normally expected. This is caused by the slow I/O processing relative to the speed of track allocation and deallocation. While LGTAT is reporting the contents of a particular track, the contents of other tracks may have changed.

A track is identified by the first program contained in that track, and LGTAT will label the track as belonging to that program even if other programs also reside on it. The first program identifies the track by one of the ways specified earlier (e.g., PROGX).

LGTAT will report a system track following the system entry points tracks. Currently, this track is reserved for system use.

Table 6-3. Explanation of Entries in Track Assignment Table

ENTRY	
progx	track belongs to program progx
progx&	track holds memory-image form of program progx
progx^	track holds program progx that has been swapped from memory to the disc
--	free track
ENTS	track holds system entry points
LS	track holds logical source tracks
SYSTEM	track is a system track
GLOBAL	track is a global track
FMP	track belongs to the file manager (FMGR)
LIBRARY	track is a library track

```
TRACK ASSIGNMENT TABLE        & =PROG ^ =SWAP

TRACK     0       1       2       3       4       5       6       7       8       9
   0    SYSTEM  SYSTEM  SYSTEM  SYSTEM  $$CMD&  AUTOR&  FMGR0&  FMGR1&  FMGR4&  FMGR5&
  10    FMGR7&  FMGR9&  XXADR&  LIBRY   LIBRY   LIBRY   LIBRY   LIBRY   LIBRY   LIBRY
  20    LIBRY   LIBRY   LIBRY   LIBRY   LIBRY   LIBRY   LIBRY   ENTS    EDI24   EDI24
  30    FMGR &  GASP &  SYSTEM  EDI24   EDI24   EDI24     --      --      --    COMPL&
  40    COMPL&    --      --      --      --    EDI24     --      --      --      --
  50     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
  60     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
  70     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
  80     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
  90     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 100     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 110     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 120     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 130     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 140     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 150     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 160     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 170     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 180     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 190     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 200     FMP     FMP   D.RTR

AUXILIARY DISC

   0    LGTAT&    --      --      --    DBSPA&  DBSPA&    --      --      --      --
  10    SPOUT&    --      --      --      --      --      --      --      --      --
  20      --      --      --      --      --      --      --      --      --      --
  30    PRMPT&    --    R$PN$     --      --      --      --      --      --      --
  40      --      --      --      --      --      --      --      --      --      --
  50      --      --      --      --      --      --      --      --      --      --
  60      --      --      --      --      --      --      --      --      --      --
  70      --      --      --      --      --      --      --      --      --      --
  80      --      --      --      --      --      --      --      --      --      --
  90      --      --      --      --      --      --      --      --      --      --
 100      --      --      --      --      --      --      --      --      --      --
 110      --      --      --      --      --      --      --      --    FMG29^  FMG29^
 120    FMG18^  FMG18^  FMG26^  FMG24^  FMG24^  FMG24^    --    COMPL^  COMPL^  SMP  ^
 130      LG      LG      LG      LG      LG      --    PRMPT^  CNSLM^  PPCNV^  RQCNV^
 140     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 150     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 160     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 170     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 180     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 190     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP     FMP
 200     FMP     FMP   D.RTR ^

THE LS TRACK(S) START AT TRACK   29   OF LU     2

TOTAL AVAILABLE TRACKS =   125
LARGEST CONTIGUOUS TRACK BLOCK =    85
```

Figure 6-6. Complete LGTAT Output

System Configuration Utility LUPRN

The utility LUPRN supplies the current system device configuration and identifies each device by its true driver name. The configuration tables can be directed to your terminal or to a list device and can be ordered by system LU number, session LU number, (session mode only) or by device select code. LUPRN will execute in both session and non-session environments.

When loading LUPRN, specify a search (SE) of the library $RSLIB or %DECAR for the modules used by LUPRN, rather than relocating the selected library. Either library can be specified; however, $RSLIB will produce the fastest results.

Running LUPRN

Invoke LUPRN with the runstring

```
[RU,]LUPRN[,lst lu[,AL[,LU[,SC[,TY[,DV[,??]]]]]]]
```

where

lst lu — is the LU of the list output device. The default is to your terminal. While the options are not order-dependent, this parameter must be the first named option if you want to specify a list device other than your terminal.

AL — list all device drivers in the system, sorted by system LU number. If the device is included in your Session Switch Table (or available to your MTM station), the device SLU number is listed. If the device is not available for your use, the SLU column is blank for that device.

LU[:n:n] — list all device drivers for the system LUs, sorted by system LU number. When the optional range is specified, the list includes only those device drivers with LU numbers within the decimal range given by :n:n. Without the range parameters, this form of the command is equivalent to the AL option. LU references are system LUs; session LUs are listed for those LUs contained in your SST.

If only one :n parameter is given, or if the second :n is smaller that the first, only the first specified LU is listed. If no LUs exist within the given range, LUPRN issues the message

..no LUs found in specified range..

as the body of the table.

SC[:n:n] list all LUs assigned to specific select codes, sorted by select code number. When the optional range is specified, the list only includes those device drivers within the decimal range given by :n:n. If no drivers exist within the specified range, LUPRN returns the message

..no LUs found in specified range..

as the body of the table.

TY[:nB:nB] list all device drivers for the System LUs, sorted by driver type. When the optional range is specified, the list only includes those drivers that fall within the range given (use the DV option, if necessary, to get the full list of drivers in the system). Note that this range should be given in octal, using the B suffix. If the suffix is omitted, the table will list the driver that is equivalent to the decimal number; that is, if you enter the decimal values :12:15, the list will show the equivalent octal drivers, 14B through 17B. As with the LU option, the selected driver list will be taken from the entire system.

DV list the system driver table, followed by the LU table. This option is useful when you want to see the total number of drivers and their descriptions.

?? list the descriptive summary of LUPRN and the available optional runstring parameters.

If LUPRN is called with no options, the list will contain your session LUs only, sorted by session LU number. In a non-session environment, calling LUPRN with no options is equivalent to the call LUPRN,AL. Note that if you specify the SC option and either the LU:n:n or TY:nB:nB options, the table will be sorted by select code rather than LU or driver type.

If you specify both TY and LU with search ranges, an output may not result unless the specified combination matches an existing condition of the system. For example, the runstring

```
RU,LUPRN,LU:6,TY:12B
```

will result in an output only if LU 6 is assigned driver type code 12B.

If you specify an illegal list LU, the message

```
..Illegal LU (nn) for output (down or not defined)
```

is issued, and LUPRN exits. The erroneous LU number is given in the message.

If you invoke LUPRN with an illegal option code, the message

```
..Unknown command: xx ...use ?? for help.
```

is issued, where xx is the erroneous command code you entered.

Output Table Format

In the output table headings, the RTE operating system under which LUPRN is running is listed, together with the system's data code. The "sorted by" message will indicate Session LU, System LU, or Select Code, depending on the option selected. The Time Base Generator select code is included, or the entry <none> is shown. If the Privileged Fence card is included in the system, the select code is given, or the entry <none> is shown. The number of memory partitions is given, or the entry <none> is shown. The configuration table contains the following column headings, as applicable for the format:

SLU - User session LU numbers (this column is included at both the left and right sides of the table): Of KV f¢bips Zith a D suffix indicates that the LU is down. This column is omitted in non-session environments.

LU - System LU numbers (this column is included at both the left and right sides of the table). An LU number with a D suffix indicates that the LU is down.

EQT - Equipment table entry number.

sc - Subchannel number.

SCD - Select code.

Flags - D = DCPC, Dual Channel Port Controller
B = Buffered
P = Driver handles power fail
S = Driver handles timeout
T = Device has timed out.

Av - Device availability: 1 = Device down
2 = Device busy
3 = Device waiting DCPC.

If there is no entry in this column, the device is available.

Stats - Device status octal code. Note that the status is for the last driver activity, and may not reflect current status. (Refer to the related Device Driver Reference Manual for a description of the status codes.)

Driver- Driver name.

DP - First physical page number of the driver partition. If a dollars sign ($) follows the DP entry, this identifies the partition page as the System Driver Area (SDA).

Device Name - True name of device. Where possible, subchannel information is used to further identify the device.

Examples

In the following example, LUPRN is invoked with the DV option. The listing is sorted by driver number, and is followed by the full configuration listing, sorted by system LU number.

```
:RU,LUPRN,DV
 LUPRN's Driver List

 NUM  Name   = Description
 ---  -----    -----------
   1  DVR00    CRT/TTY terminal
   2  DVC00    CRT/TTY terminal
   3  DVD00    Logical Driver
   4  DVT00    306C Cent.printr
   5  DVM00    12792A 8-CH MUX
   6  DVX00    CIS BMUX Port
   7  DVS00    12790 MUX-DVR00
   8  DVV00    DS-Remotemapping
   9  DVR01    Papertape Reader
  10  DVC01    Papertape Reader
  11  DVR02    Papertape Punch
  12  DVC02    Papertape Punch
  13  DVR05    264x/2x Terminal
  14  DVS05    12920 MUX-DVR05
  15  DVT05    Tektronix CRT
  16  DVA05    Terminal w/modem
  17  DVQ05    BACI card GP DVR
  18  DVM05    8-CH MUX (DDV05)
    .    .           .
    .    .           .
    .    .           .
  73  DVS56    Preston A/D
  74  DVR57    12564 10 bit A/D
  75  DVR60    6129 Volt.source
  76  DVG61    12967 Sync Card
  77  DVR61    Obsolete Driver
  78  DVA62    2313 (RTE3)
  79  DVR62    2440 A/D (2313)
  80  DVR63    3000 UI Card
  81  DVS63    Parallel FDPLX
:
```

In the following example, LUPRN is invoked to list all system LUs within the range of LU 12 through LU 23. If no LUs are found within the range, the ..no LUs found in specified range.. message is issued as the body of the table. Note that the table contains entries for unidentifiable drivers, flagged with an asterisk. The configuration list also includes a note, Note 1., related to the unidentified driver entry. LUPRN provides seven notes related to specific conditions of the system. These notes are defined following the examples.

```
:RU,LUPRN,LU:12:23
                    RTE-6 System Device Configuration
               RTE-6 System rev = 2226  LUPRN's rev = 2240
            1:52 PM TUE., 9 NOV., 1982...Sorted by System LU
Time Base (14B)  Priv. Fence SC (none)  Partitions (24)  Memory size (576K)

SLU LU EQT,sc SCD Flags AV T.out Stats  Driver DP Device Name      LU  SLU
--- --------- --- ----- -- ----- -----  ------ -- ---------------- --- ---
     12 38,3  31B  B              130B  DVB12  45 2608A read-back   12
     13 39    71B  B                    DVZ12   4 2608A Graphics    13
     15 40    04B                       DVP43   9$Power Fail        15
 16  16  1,1  15B D                120B  DV?32*  4 .Unknown Driver.  16  16
     17  1,2  15B D                120B  DVR32   4 7905/6/20/25 DSK  17
 18  18  1,3  15B D                120B  DVR32   4 7905/6/20/25 DSK  18  18
     19  3,4  16B D                100B  DV?32* 41 .Unknown Driver.  19
     20  3,1  16B D                100B  DVR32  41 7905/6/20/25 DSK  20
     21  3,10 16B D                100B  DVR32  41 7905/6/20/25 DSK  21
 22  22  3    16B D                100B  DVR32  41 7905/6/20/25 DSK  22  22
     23  3,11 16B D                100B  DVR32  41 7905/6/20/25 DSK  23
--- --- ----- --- ----- -- ----- -----  ------ -- ---------------- --- ---

   Notel: DV?XX* indicates that the true driver name is not determinable
          since there are other drivers with the same INIT/CONT addresses.

     DP=Driver Partition page ($=SDA), SLU=Session LU
     (T.out is in seconds)                EQT Flags:
     LU   with a      EQT availability:     D=DCPC, B=Buffered, T=Timed-out
     D means the       1=down, 2=busy,      P=Driver handles Powerfail
     LU is down.       3=waiting DCPC       S=Driver handles Timeout

:
```

In the following example, LUPRN is invoked to list all driver types within the range of 5B through 12B, sorted by system LU number. If no drivers are found within the range specified, the ..no LUs found.. message is issued as the body of the table. In the listing, those devices designated as spool printers are identified as such in the Device Name column. Note the letter "D" following LU entry 86. This signifies that the LU is down.

```
:RU,LUPRN,TY:5B:12B
                    RTE-6 System Device Configuration
                 RTE-6 System rev = 2226  LUPRN's rev = 2240
              1:53 PM TUE., 9 NOV., 1982...Sorted by System LU
Time Base (14B)  Priv. Fence SC (none)  Partitions (24)  Memory size (576K)

SLU LU EQT,sc SCD Flags AV T.out Stats Driver DP Device Name      LU  SLU
--- -- ------ --- ----- -- ----- ----- ------ -- --------------- --- ---
  6   6  6     26B  B        200.00       DVR12  45 2767 80col Prntr   6   6
 11  11 38     31B  B                130B DVB12  45 2608A Printer     11  11
     12 38,3   31B  B                130B DVB12  45 2608A read-back   12
     13 39     71B  B                     DVZ12   4 2608A Graphics    13
 56  56 13     32B  B S                2B DVA05  47 Terminal w/modem  56  56
     57 14     33B  B S   2             2B DVA05  47 Terminal w/modem  57
     58 15     34B  B S   2             2B DVA05  47 Terminal w/modem  58
     59 16     35B    S      200.00     2B DVA05  47 Terminal w/modem  59
     .  .      .    .        .         .  .     .        .
     .  .      .    .        .         .  .     .        .
     .  .      .    .        .         .  .     .        .
     84 20,1   41B  B S      200.00     2B DVA05  47 Left CTU@ LU 63   84
     85 20,2   41B  B S      200.00     2B DVA05  47 Right CTU@ LU 63  85
     86D21,1   42B    S      200.00     2B DVA05  47 Left CTU@ LU 64   86
     87 21,2   42B  B S   2             2B DVA05  47 Right CTU@ LU 64  87
     88 22,1   43B  B S      200.00     2B DVA05  47 Left CTU@ LU 65   88
     89 22,2   43B  B S      200.00     2B DVA05  47 Right CTU@ LU 65  89
     94 37     44B  B S      100.00     2B DVA05  47 Terminal w/modem  94
    101 34     75B                        DVS12   9$Spool....type=12 101
    102 35     76B                        DVS12   9$Spool....type=12 102
    103 36     77B                        DVS12   9$Spool....type=12 103
    117 37     44B  B S      100.00     2B DVA05  47 Terminal w/modem 117
    118 37,1   44B  B S      100.00     2B DVA05  47 Left CTU@ LU117  118
    119 37,2   44B  B S      100.00     2B DVA05  47 Right CTU@ LU117 119
--- --- ----- --- ----- -- ------ ---  ----- -- --------------- --- ---

  DP=Driver Partition page ($=SDA), SLU=Session LU
  LU   with a      EQT availability:     D=DCPC, B=Buffered, T=Timed-out
  D means the       1=down, 2=busy,      P=Driver handles Powerfail
  LU is down.       3=waiting DCPC       S=Driver handles Timeout

:
```

The following example invokes LUPRN with the SC option to list all devices within the range of 10B through 23B, sorted by select code. The listing contains the entry "<Empty Select Code>" to indicate that no Equipment Table entry was found for this device select code. The configuration table also contains the entries ".Unknown Driver." for those drivers whose identity could not be determined, and includes an explanatory note. (Refer to the section LUPRN Notes for the text of all notes that can be listed.)

```
:RU,LUPRN,SC:10B:23B
                    RTE-6 System Device Configuration
              RTE-6 System rev = 2226   LUPRN's rev = 2240
            1:54 PM TUE., 9 NOV., 1982...Sorted by Select Code
Time Base (14B)  Priv. Fence SC (none)  Partitions (24)  Memory size (576K)

SLU LU EQT,sc SCD Flags AV T.out  Stats Driver DP Device Name      LU SLU
--- -- ------ --- ----- -- ------ ----- ------ -- ---------------- -- ---
               10B  <Empty Select Code>
    104 41    11B   PS               26B DVA66  53 HDLC/BiSync card 104
    106 43    12B   PS                4B DVA66  53 HDLC/BiSync card 106
  7   7 10    13B   PS        .03     4B DVA65  51 DS1000 to 1000     7  7
              14B                       <RTE>     Timebase Generator
  2   2  1    15B D                   120B DV?32*  4 .Unknown Driver.   2  2
  3   3  3    16B D                   100B DV?32* 41 .Unknown Driver.   3  3
      1  2    17B    S   2 327.67      2B DVR00  45 CRT/TTY terminal   1
 55   5  5    20B                      40B DVR01  45 Papertape Reader  55  5
 44   4  4    21B  B        200.00        DVR02  45 Papertape Punch   44  4
 54  54 12    22B  B S      200.00        DVA37  49 59310 HPIB Card   54 54
  9   9  9    23B   PS         .03        DVA65  51 DS1000 to 1000     9  9
--- --- ----- --- ----- -- ------ ----- ------ -- ---------------- --- --

  Note1: DV?XX* indicates that the true driver name is not determinable
         since there are other drivers with the same INIT/CONT addresses.

  DP=Driver Partition page ($=SDA), SLU=Session LU
  (T.out is in seconds)                EQT Flags:
  LU   with a      EQT availability:     D=DCPC, B=Buffered, T=Timed-out
  D means the       1=down, 2=busy,      P=Driver handles Powerfail
  LU is down.       3=waiting DCPC       S=Driver handles Timeout

:
```

The following example invokes LUPRN with no options, to list the configuration of all devices available to the session, sorted by session LU number. Note the unknown driver entries, and the explanatory note following the table.

```
:RU,LUPRN
                    RTE-6 System Device Configuration
                RTE-6 System rev = 2226  LUPRN's rev = 2240
             1:55 PM TUE., 9 NOV., 1982...Sorted by Session LU
Time Base (14B)  Priv. Fence SC (none)  Partitions (24)  Memory size (576K)

SLU LU EQT,sc SCD Flags AV T.out Stats  Driver DP Device Name      LU SLU
--- -- ------ --- ----- -- ----- -----  ------ -- ---------------- -- ---
  1  67 48     70B    ST             2B  DVV05  51 Logical DVR =  5  67  1
  2   2 1      15B D                120B  DV?32*  4 .Unknown Driver.   2  2
  3   3 3,16   16B D                100B  DV?32* 41 .Unknown Driver.   3  3
  4  92 25     45B  B        200.00  40B  DVM00  55 12792A 8-CH MUX   92  4
  5  93 26     45B  B          5.00  40B  DVM00  55 12792A 8-CH MUX   93  5
  6   6  6     26B  B        200.00       DVR12  45 2767 80col Prntr   6  6
  7   7 10,1   13B   PS          .03       DVA65  51 DS1000 to 1000     7  7
  8   8  8     24B  B S        5.00   4B  DVR23  41 9TK Mag Tape  0    8  8
  9   9  9,1   23B   PS          .03       DVA65  51 DS1000 to 1000     9  9
 10  10  7     27B    S          5.00   1B  DVR23  41 9TK Mag Tape  0   10 10
 11  11 38     31B  B                130B  DVB12  45 2608A Printer     11 11
 16  16  1,1   15B D                120B  DVR32   4 7905/6/20/25 DSK  16 16
 18  18  1,3   15B D                120B  DVR32   4 7905/6/20/25 DSK  18 18
 22  22  3     16B D                100B  DVR32  41 7905/6/20/25 DSK  22 22
 37  67 48     70B    ST             2B  DVV05* 51 .Unknown Driver.  67 37
 38  38  3,5   16B D                100B  DVR32  41 7905/6/20/25 DSK  38 38
 39  69 50     70B                  40B  DVV00  51 DS-Remotemapping  69 39
 41  41  3,8   16B D                100B  DVR32  41 7905/6/20/25 DSK  41 41
 44   4  4     21B  B        200.00       DVR02  45 Papertape Punch    4 44
 54  54 12,2   22B  B S      200.00       DVA37  49 HPIB address   2  54 54
 55   5  5     20B                  40B  DVR01  45 Papertape Reader   5 55
 56  56 13     32B  B S              2B  DVA05  47 Terminal w/modem  56 56
--- --- ------ --- ----- -- ------ ----  ------ -- ---------------- --- --

 Note1: DV?XX* indicates that the true drivername is not determinable
        since there are other drivers with the same INIT/CONT addresses.

 DP=Driver Partition page ($=SDA), SLU=Session LU
 (T.out is in seconds)                  EQT Flags:
 LU   with a      EQT availability:       D=DCPC, B=Buffered, T=Timed-out
 D means the       1=down, 2=busy,        P=Driver handles Powerfail
 LU is down.       3=waiting DCPC         S=Driver handles Timeout

:
```

The last example invokes LUPRN with the AL option to list the configuration of all LUs in the system, sorted by system LU. Where a spooling printer is unassigned, the entry <idle> is shown in the Device Name column. This listing also identifies all LUs that are unassigned in the system.

```
:RU,LUPRN,AL
                    RTE-6 System Device Configuration
                RTE-6 System rev = 2226  LUPRN's rev = 2240
             1:58 PM TUE., 9 NOV., 1982...Sorted by System LU
Time Base (14B) Priv. Fence SC (none)  Partitions (24)  Memory size (576K)

SLU LU EQT,sc SCD Flags AV T.out  Stats  Driver DP Device Name      LU SLU
--- -- ------ --- ----- -- ------ -----  ------ -- ---------------- -- ---
      1  2    17B    S     327.67   2B   DVR00  45 CRT/TTY terminal  1
  3   3  3,16 16B D      2          101B DV?32* 42 .Unknown Driver.  3   3
 44   4  4    21B  B       200.00        DVR02  45 Papertape Punch  44   4
 55   5  5    20B                 40B   DVR01  45 Papertape Reader 55   5
  6   6  6    26B  B       200.00        DVR12  45 2767 80col Prntr  6   6
  7   7 10,1  13B   PS         .03        DVA65  51 DS1000 to 1000    7   7
  8   8  8    24B  B S       5.00   1B   DVR23  41 9TK Mag Tape  0   8   8
  9   9  9,1  23B   PS         .03        DVA65  51 DS1000 to 1000    9   9
 10  10  7    27B    S       5.00   1B   DVR23  41 9TK Mag Tape  0  10  10
 11  11 38    31B  B                 130B  DVB12  45 2608A Printer    11  11
     12 38,3  31B  B                 130B  DVB12  45 2608A read-back  12
     13 39    71B  B                       DVZ12   4 2608A Graphics   13
     14 11    61B  B       200.00   40B   DVR00  45 CRT/TTY terminal 14
     15 40    04B                        DVP43   9$Power Fail       15
 16  16  1,1  15B D                 120B  DVR32   4 7905/6/20/25 DSK 16  16
     17  1,2  15B D                 120B  DVR32   4 7905/6/20/25 DSK 17
 18  18  1,3  15B D                 120B  DVR32   4 7905/6/20/25 DSK 18  18
     19  3,4  16B D      2          100B  DVR32  41 7905/6/20/25 DSK 19
      .   .    .        .           .     .     .         .          .
      .   .    .        .           .     .     .         .          .
      .   .    .        .           .     .     .         .          .
```

```
     .   .     .        .          .     .     .        .          .
     .   .     .        .          .     .     .        .          .
     .   .     .        .          .     .     .        .          .

 5   93 26     45B   B          5.00   40B   DVM00   55 12792A 8-CH MUX     5 93
     94 37     44B   B S      100.00    2B   DVA05   47 Terminal w/modem   94
     95 28     45B   B        200.00   40B   DVM00   55 12792A 8-CH MUX    95
     96 29     45B   B        200.00   40B   DVM00   55 12792A 8-CH MUX    96
     97 30     45B   B        200.00   40B   DVM00   55 12792A 8-CH MUX    97
     98 31     72B                             DVS43    9$Spooling (idle)   98
     99 32     73B                             DVS43    9$Spooling (idle)   99
    100 33     74B                             DVS43    9$Spooling (idle)  100
    101 34     75B                             DVS12    9$Spool....type=12 101
    102 35     76B                             DVS12    9$Spool....type=12 102
    103 36     77B                             DVS12    9$Spool....type=12 103
    104 41     11B    PS                26B   DVA66   53 HDLC/BiSync card 104
    105 42     11B    PS                      DVA66   53 HDLC/BiSync card 105
    106 43     12B    PS                 4B   DVA66   53 HDLC/BiSync card 106
    107 44     12B    PS                      DVA66   53 HDLC/BiSync card 107
    108 45     62B                            DVA66   53 HDLC/BiSync card 108
    109 46     62B                            DVA66   53 HDLC/BiSync card 109
    110 12     22B   B S      200.00          DVA37   49 59310 HPIB Card  110
    111 12,1   22B   B S      200.00          DVA37   49 HPIB address   1 111
    112 12,2   22B   B S      200.00          DVA37   49 HPIB address   2 112
    113 12,3   22B   B S      200.00          DVA37   49 HPIB address   3 113
    114 12,4   22B   B S      200.00          DVA37   49 HPIB address   4 114
    115 51     67B                            DV?77+  57 .Unknown Driver. 115
    116 52     66B                            DV?77+  59 .Unknown Driver. 116
    117 37     44B   B S      100.00    2B   DVA05   47 Terminal w/modem 117
    118 37,1   44B   B S      100.00    2B   DVA05   47 Left CTU@ LU117  118
    119 37,2   44B   B S      100.00    2B   DVA05   47 Right CTU@ LU117 119

        ************  System LU's 120 thru 122 Unassigned  *************
--- --- ------ --- ----- -- ------ -----  ------ -- ---------------- -- ---

 Note1: DV?XX* indicates that the true drivername is not determinable
        since there are other drivers with the same INIT/CONT addresses.

 Note2: DV?XX+ indicates that no entry point in the system matches either
        INIT/CONT addresses in this EQT.  Possibly a sysgen error or
        incomplete patch has been made, or this is a dummy driver.

 DP=Driver Partition page ($=SDA), SLU=Session LU
 (T.out is in seconds)                  EQT Flags:
 LU   with a       EQT availability:      D=DCPC, B=Buffered, T=Timed-out
 D means the        1=down, 2=busy,       P=Driver handles Powerfail
 LU is down.        3=waiting DCPC        S=Driver handles Timeout
```

LUPRN Notes

The following explanatory notes can be displayed by LUPRN as specific related conditions are found in the system configuration:

Note1: DV?XX* indicates that the true driver name is not determinable since there are other drivers with the same INIT/CONT addresses.

Note2: DV?XX+ indicates that no entry point in the system matches either INIT/CONT addresses in this EQT. Possibly a system error or incomplete patch has been made, or this is a dummy driver.

The above two notes are related to the symbols * and + as they appear next to the driver number in the Driver column.

Note3: 'No EQT' indicates that no EQT claims use of this select code. Possibly a sysgen error was made in the EQT for an interrupt address.

The above note is related to the entry or entries identified by the message <Unknown Interrupt Table Entry - no EQT> following the SCD (select code) column.

Note4: The program name listed will be scheduled upon an interrupt from this Select Code. There is no EQT or driver associated with this S.C

The above note is related to the entry or entries identified by the message <Interrupt schedules program: (prog name) ...see Note 4> following the SCD column. The program name is given in the message.

Note5: The value listed is not an ID segment address and no EQT points to this Select Code.

The above note is related to the entry or entries identified by the message <Interrupt table entry unknown (value)...See Note 5> following the SCD column. The value is given in the message. This condition may be due to an incorrect patch or a corrupt system.

Note 6: A subchannel for CS-80 disc is not part of the track map table. This LU cannot be accessed.

The above note is related to the entry or entries identified by the message <CS80 subchan bad, (nn)> following the SCD column. The subchannel is given as the number nn in the message. This condition may occur because of a system error, or as a result of assigning an LU to a non-existant disc subchannel.

Note 7: Unable to find track map table for CS-80 disc driver(s). The TMT is not in the system entry point list.

The above note is related to the fact that LUPRN only searches the system entry point list for the track map tables. This allows LUPRN to continue without asking the driver for the TMT entry. The CS-80 driver requires an I/O card to return the TMT entry.

If the device is a CS-80 cartridge tape drive (CTD), and an assigned cache is not found in memory, LUPRN issues the following message:

```
Warning: The CS-80 Cartridge Tape Unit
is not set up with a disc cache. This
will cause excessive wear on the drive.
```

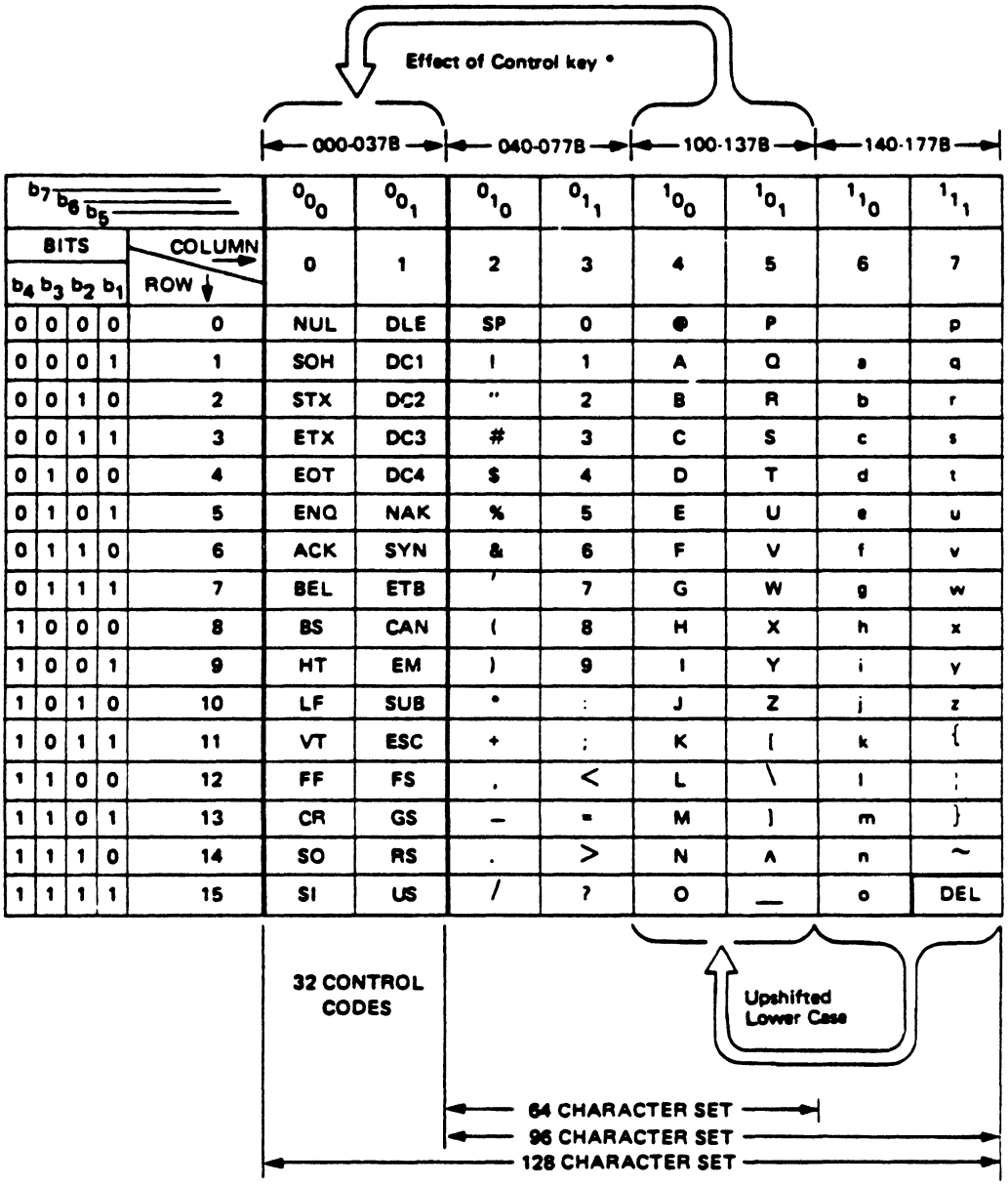
LUPRN Errors

On an error condition, LUPRN issues the message

```
...LUPRN: entry error code = nn
```

where nn identifies the I/O call error code. Refer to the RTE-IVB Quick Reference Guide for a description of the I/O error codes.

Appendix A
HP Character Set



b_7 b_6 b_5					0_0	0_1	$0_1{}_0$	$0_1{}_1$	1_0	$1_0{}_1$	$1_1{}_0$	$1_1{}_1$
BITS				COLUMN → / ROW ↓	0	1	2	3	4	5	6	7
b_4	b_3	b_2	b_1									
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	|
1	1	0	1	13	CR	GS	-	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

EXAMPLE: The representation for the character "K" (column 4, row 11) is.

	b_7	b_6	b_5	b_4	b_3	b_2	b_1
BINARY	1	0	0	1	0	1	1
OCTAL	1		1		3		

* Depressing the Control key while typing an upper case letter produces the corresponding control code on most terminals. For example, Control-H is a backspace.

9206-1A

HEWLETT-PACKARD CHARACTER SET FOR COMPUTER SYSTEMS

This table shows HP's implementation of ANS X3.4-1968 (USASCII) and ANS X3.32-1973. Some devices may substitute alternate characters from those shown in this chart (for example, Line Drawing Set or Scandanavian font). Consult the manual for your device.

The left and right byte columns show the octal patterns in a 16 bit word when the character occupies bits 8 to 14 (left byte) or 0 to 6 (right byte) and the rest of the bits are zero. To find the pattern of two characters in the same word, add the two values. For example, AB produces the octal pattern 040502. (The parity bits are zero in this chart.)

The octal values 0 through 37 and 177 are control codes. The octal values 40 through 176 are character codes.

Decimal Value	Octal Values		Mnemonic	Graphic[1]	Meaning
	Left Byte	Right Byte			
0	000000	000000	NUL	^U	Null
1	000400	000001	SOH	^A	Start of Heading
2	001000	000002	STX	^B	Start of Text
3	001400	000003	ETX	^C	End of Text
4	002000	000004	EOT	^D	End of Transmission
5	002400	000005	ENQ	^E	Enquiry
6	003000	000006	ACK	^F	Acknowledge
7	003400	000007	BEL	^G	Bell, Attention Signal
8	004000	000010	BS	^H	Backspace
9	004400	000011	HT	^I	Horizontal Tabulation
10	005000	000012	LF	^J	Line Feed
11	005400	000013	VT	^K	Vertical Tabulation
12	006000	000014	FF	^L	Form Feed
13	006400	000015	CR	^M	Carriage Return
14	007000	000016	SO	^N	Shift Out (Alternate Character Set)
15	007400	000017	SI	^O	Shift In (Alternate Character Set)
16	010000	000020	DLE	^P	Data Link Escape
17	010400	000021	DC1	^Q	Device Control 1 (X-ON)
18	011000	000022	DC2	^R	Device Control 2 (TAPE)
19	011400	000023	DC3	^S	Device Control 3 (X-OFF)
20	012000	000024	DC4	^T	Device Control 4 (TAPE)
21	012400	000025	NAK	^U	Negative Acknowledge
22	013000	000026	SYN	^V	Synchronous Idle
23	013400	000027	ETB	^W	End of Transmission Block
24	014000	000030	CAN	^X	Cancel
25	014400	000031	EM	^Y	End of Medium
26	015000	000032	SUB	^Z	Substitute
27	015400	000033	ESC	^[	Escape[2]
28	016000	000034	FS	^\	File Separator
29	016400	000035	GS	^]	Group Separator
30	017000	000036	RS	^^	Record Separator
31	017400	000037	US	^_	Unit Separator
127	077400	000177	DEL	■	Delete (Rubout)[3]

Decimal Value	Octal Values		Character	Meaning
	Left Byte	Right Byte		
32	020000	000040		Space, Blank
33	020400	000041	!	Exclamation Point
34	021000	000042	"	Quotation Mark
35	021400	000043	#	Number Sign, Pound Sign
36	022000	000044	$	Dollar Sign
37	022400	000045	%	Percent
38	023000	000046	&	Ampersand, And Sign
39	023400	000047	'	Apostrophe, Acute Accent
40	024000	000050	(	Left (opening) Parenthesis
41	024400	000051	)	Right (closing) Parenthesis
42	025000	000052	*	Asterisk, Star
43	025400	000053	+	Plus
44	026000	000054	,	Comma, Cedilla
45	026400	000055	-	Hyphen, Minus, Dash
46	027000	000056	.	Period, Decimal Point
47	027400	000057	/	Slash, Slant
48	030000	000060	0	Digits, Numbers
49	030400	000061	1	
50	031000	000062	2	
51	031400	000063	3	
52	032000	000064	4	
53	032400	000065	5	
54	033000	000066	6	
55	033400	000067	7	
56	034000	000070	8	
57	034400	000071	9	
58	035000	000072	:	Colon
59	035400	000073	;	Semicolon
60	036000	000074	<	Less Than
61	036400	000075	=	Equals
62	037000	000076	>	Greater Than
63	037400	000077	?	Question Mark

Decimal Value	Octal Values		Character	Meaning
	Left Byte	Right Byte		
64	040000	000100	@	Commercial At
65	040400	000101	A	Upper Case Alphabet Capital Letters
66	041000	000102	B	
67	041400	000103	C	
68	042000	000104	D	
69	042400	000105	E	
70	043000	000106	F	
71	043400	000107	G	
72	044000	000110	H	
73	044400	000111	I	
74	045000	000112	J	
75	045400	000113	K	
76	046000	000114	L	
77	046400	000115	M	
78	047000	000116	N	
79	047400	000117	O	
80	050000	000120	P	
81	050400	000121	Q	
82	051000	000122	R	
83	051400	000123	S	
84	052000	000124	T	
85	052400	000125	U	
86	053000	000126	V	
87	053400	000127	W	
88	054000	000130	X	
89	054400	000131	Y	
90	055000	000132	Z	
91	055400	000133	[	Left (opening) Bracket
92	056000	000134	\	Backslash, Reverse Slant
93	056400	000135	]	Right (closing) Bracket
94	057000	000136	^ ↑	Caret, Circumflex, Up Arrow[4]
95	057400	000137	_ ←	Underline, Back Arrow[4]

Decimal Value	Octal Values		Character	Meaning
	Left Byte	Right Byte		
96	060000	000140	`	Grave Accent[5]
97	060400	000141	a	Lower Case Letters[5]
98	061000	000142	b	
99	061400	000143	c	
100	062000	000144	d	
101	062400	000145	e	
102	063000	000146	f	
103	063400	000147	g	
104	064000	000150	h	
105	064400	000151	i	
106	065000	000152	j	
107	065400	000153	k	
108	066000	000154	l	
109	066400	000155	m	
110	067000	000156	n	
111	067400	000157	o	
112	070000	000160	p	
113	070400	000161	q	
114	071000	000162	r	
115	071400	000163	s	
116	072000	000164	t	
117	072400	000165	u	
118	073000	000166	v	
119	073400	000167	w	
120	074000	000170	x	
121	074400	000171	y	
122	075000	000172	z	
123	075400	000173	{	Left (opening) Brace[5]
124	076000	000174	|	Vertical Line[5]
125	076400	000175	}	Right (closing) Brace[5]
126	077000	000176	~	Tilde, Overline[5]

9208-1C

Notes

[1]This is the standard display representation. The software and hardware in your system determine if the control code is displayed, executed, or ignored. Some devices display all control codes as ||, @, or space.

[2]Escape is the first character of a special control sequence. For example, ESC followed by J clears the display on a 2640 terminal.

[3]Delete may be displayed as __, @, or space.

[4]Normally, the caret and underline are displayed. Some devices substitute the up arrow and back arrow.

[5]Some devices upshift lower case letters and symbols (` through ~) to the corresponding upper case character (@ through ^). For example, the left brace would be converted to a left bracket.

RTE SPECIAL CHARACTERS

Mnemonic	Octal Value	Use
SOH (Control A)	1	Backspace (TTY)
EM (Control Y)	31	Backspace (2600)
BS (Control H)	10	Backspace (TTY, 2615, 2640, 2644, 2645)
EOT (Control D)	4	End-of-file (TTY 2615, 2640, 2644, 2645)

9206-1D

Appendix B
Error Messages Index

FMGR Error Messages

""
FMGR-105
D.RTR DIRECTORY TRACK BUFFER TOO SMALL
THE LENGTH OF THE DIRECTORY BUFFER IN THE ROUTINE D.BUF WAS DEFINED TO BE LESS THAN 512 WORDS WHEN LOADED WITH D.RTR.
ALTER THE SIZE OF THE DIRECTORY BUFFER IN D.BUF BY ALTERING THE CONSTANT DEFINED BY D.LEN IN THE SOURCE, REASSEMBLE THE ROUTINE D.BUF AND REGENERATE THE SYSTEM WITH THE RELOCATABLE.
""
FMGR-102
ILLEGAL D.RTR CALL SEQUENCE
A LOCK WAS NOT REQUESTED FIRST OR THE FILE WAS NOT OPENED EXCLUSIVELY. POSSIBLY AN OPERATOR ERROR, SUCH AS REMOVING A CARTRIDGE WITHOUT DISMOUNTING IT FIRST.
""
FMGR-101
ILLEGAL PARAMETER IN D.RTR CALL
POSSIBLY AN OPERATOR ERROR. RECHECK THE PREVIOUS ENTRIES FOR ILLEGAL OR MISPLACED PARAMETERS.
THIS ERROR CAN ALSO HAPPEN WHEN A REQUEST IS MADE TO CREATE A SCRATCH FILE AND THAT SCRATCH FILE ALREADY EXISTS. IF D.RTR IS UNABLE TO PURGE THE EXISTING SCRATCH FILE, THIS ERROR IS RETURNED. THIS CAN ONLY HAPPEN IF SOME OTHER PROGRAM HAS THE SCRATCH FILE OPEN. SEE THE SYSTEM MANAGER.
""
FMGR-099
DIRECTORY MANAGER EXEC REQUEST WAS ABORTED
AN EXEC REQUEST MADE BY D.RTR WAS ABORTED. MAKE SURE THAT ALL DISCS BEING ACCESSED ARE UP. NOTIFY SYSTEM MANAGER.
""
FMGR-052
SPOOL SHUT DOWN. SPOOL FILE SETUP FAILED
SPOOL SHUT DOWN IS IN PROGRESS. A WRITE ONLY (WR) OR WRITE/READ (BO) SPOOL FILE CANNOT BE SET UP AT THIS TIME. START UP SPOOLING USING GASP 'SU' COMMAND AND THEN TRY THE SPOOL FILE SETUP.
""
FMGR-048
SPOOL NOT INITIALIZED OR SMP CANNOT BE SCHEDULED
IF SPOOLING NOT INITIALIZED RUN GASP TO DO SO. OTHERWISE, SMP PROGRAM IS NOT FOUND OR THERE IS NOT A BIG ENOUGH PARTITION TO RUN SMP. THE DEFAULT FOR SMP IS TYPE 2 (REALTIME) AND 6 PAGES IN SIZE.
""
FMGR-047
NO SESSION LU AVAILABLE FOR SPOOL FILE
IF THE SESSION LU TO BE USED FOR THE SPOOL FILE IS NOT SPECIFIED DURING SET UP, SMP ALLOCATES A SESSION LU LESS THAN 64 THAT IS NOT ALREADY USED

IN THE SESSION SWITCH TABLE. USE :SL,LU,- COMMAND TO RELEASE A SESSION LU IN THE SPARE PART OF THE SESSION SWITCH TABLE.

""

FMGR-046
GREATER THAN 255 EXTENTS
ATTEMPT TO CREATE EXTENT 256. MAKE FILE SIZE OF MAIN LARGER.
IF GENERATED DURING A SM COMMAND, THE MESSAGE IS NOT PUT IN THE MESSAGE FILE. IT IS TRUNCATED AT THE LAST VALID MESSAGE.

""

FMGR-041
NO ROOM IN SST
THERE ARE NO SPARE ENTRIES LEFT IN THE SESSION SWITCH TABLE. SPARE ENTRIES CAN BE RECOVERED BY USING THE :SL,LU,- COMMAND, WHERE LU IS A SESSION LOGICAL UNIT NUMBER THAT IS NOT NEEDED.

""

FMGR-040
LU NOT FOUND IN SST
TRYING TO ACCESS AN LU THAT IS NOT IN YOUR SESSION SWITCH TABLE.
USE THE SL COMMAND TO ADD THE LU TO THE SST.

""

FMGR-039
SPOOL LU NOT MAPPED TO THE SPOOL DRIVER
SPOOL LU MUST POINT TO A SPOOL EQT. SWITCH ALL SPOOL LU'S TO POINT TO SPOOL EQT'S AND TRY THE SPOOL FILE SET UP AGAIN.

""

FMGR-038
ILLEGAL SCRATCH FILE NUMBER
ATTEMPT TO CREATE A SCRATCH FILE WILL AN ILLEGAL SCRATCH FILE NUMBER. THE RANGE FOR SCRATCH FILE NUMBERS IS 0 THROUGH 99. ISSUE CREATE AGAIN WITH A NUMBER IN THE CORRECT RANGE.

""

FMGR-037
ATTEMPT TO PURGE AN ACTIVE TYPE 6 FILE
AN ATTEMPT WAS MADE TO PURGE A TYPE 6 FILE WHICH HAS BEEN RP'D INTO THE SYSTEM. OFF THE RP'D PROGRAM AND TRY AGAIN.

""

FMGR-036
LOCK ERROR ON DEVICE
A CALL TO OPENF CAUSED AN ATTEMPTED LOCK ON A DEVICE AND THAT LOCK WAS UNSUCCESSFUL. THIS COULD HAPPEN IF THE DEVICE IS ALREADY LOCKED OR IF THERE ARE NO RESOURCE NUMBERS AVAILABLE.

""

FMGR-035
ALREADY 63 DISCS MOUNTED TO SYSTEM
AN ATTEMPT WAS MADE TO MOUNT A DISC WHEN THERE ARE ALREADY 63 DISCS

""
FMGR-025

NO SPLCON ROOM
THE SPLCON IS FULL. THIS ERROR MAY OCCUR WHEN THE SPOOL SYSTEM IS COMPETING WITH PROGRAMS USING THEIR OWN SPOOLING FILE AND RUNNING OUTSIDE OF BATCH.

""
FMGR-024

NO MORE BATCH SWITCHES
THE LU SWITCH TABLE IS FULL. THE SIZE OF THE SWITCH TABLE SPECIFIED AT SYSTEM GENERATION IS INADEQUATE. NOTIFY THE SYSTEM MANAGER OF THIS CONDITION.

""
FMGR-023

NO AVAILABLE SPOOL FILES
ALL SPOOL FILES ARE CURRENTLY BEING USED. RE-RUN THE JOB AFTER A SPOOL FILE BECOMES AVAILABLE.

""
FMGR-022

NO AVAILABLE SPOOL LU'S
ALL SPOOL LOGICAL UNITS ARE CURRENTLY UNAVAILABLE. RE-RUN THE JOB AFTER A SPOOL LU BECOMES AVAILABLE.

""
FMGR-021

ILLEGAL DESTINATION LU
THE LU SPECIFIED WAS NOT ALLOCATED BY GASP. TRY AGAIN USING A LU ALLOCATED BY GASP.

""
FMGR-020

ILLEGAL ACCESS LU
1. THE LOGICAL UNIT NUMBER SPECIFIED IN THE LU OR CS COMMAND WAS NOT A POSITIVE LOGICAL UNIT NUMBER. RE-ENTER THE CORRECTED COMMAND. OR
2. THERE IS AN LU ENTRY IN THE CARTRIDGE LIST THAT DOES NOT POINT TO A DISC DEVICE. THIS HAPPENED BECAUSE AFTER THE DISC WAS MOUNTED THE LU COMMAND WAS USED TO DO A LOGICAL UNIT SWITCH ON THE DEVICE. SWITCH THE LU BACK TO ITS DISC DEFINITION. IF DESIRED, DISMOUNT THE DISC. THE LU CAN THEN BE SWITCHED TO A NON-DISC DEVICE.

""
FMGR-019

ILLEGAL ACCESS ON A SYSTEM DISC
AN ATTEMPT WAS MADE TO WRITE ON A SYSTEM DISC. THE SYSTEM MANAGER IS THE ONLY USER THAT HAS THIS CAPABILITY.

""
FMGR-018

ILLEGAL LU; LU NOT ASSIGNED TO SYSTEM
ATTEMPT TO ACCESS AN LU THAT IS NOT ASSIGNED TO THE SYSTEM.

""
FMGR-017

ILLEGAL READ/WRITE ON TYPE 0 FILE
AN ATTEMPT WAS MADE TO READ, WRITE, OR POSITION A TYPE 0 FILE THAT DOES NOT SUPPORT THE OPERATION. CHECK THE FILE PARAMETERS OR THE NAMR.

""
FMGR-016

ILLEGAL TYPE 0 OP SIZE=0
ONE OF THE FOLLOWING OCCURED:
1) THE WRONG FILE TYPE WAS SPECIFIED,
2) AN ATTEMPT WAS MADE TO CREATE OR PURGE A TYPE 0 FILE, OR
3) THE SIZE SPECIFIED WAS ZERO.
CHECK THE SIZE AND TYPE PARAMETERS.

""
FMGR-015

ILLEGAL NAME
THE FILE NAME DOES NOT CONFORM TO THE SYNTAX RULES. CORRECT THE NAME AND RE-ENTER THE COMMAND.

""
FMGR-014

DIRECTORY FULL
THERE IS NO MORE ROOM IN THE FILE DIRECTORY. PURGE ANY UNUSED FILES AND PACK THE DISC IF POSSIBLE. OTHERWISE, TRY ANOTHER CARTRIDGE.

""
FMGR-013

DISC LOCKED
THE CARTRIDGE SPECIFIED IS LOCKED. INITIALIZE THE CARTRIDGE IF IT WAS NOT INITIALIZED, OTHERWISE KEEP TRYING.

""
FMGR-012

EOF OR SOF ERROR
AN ATTEMPT WAS MADE TO READ, WRITE, OR POSITION A FILE BEYOND THE FILE BOUNDARIES. CHECK THE RECORD POSITION PARAMETERS. THE RESULTS DEPEND ON THE FILE TYPE AND THE CALL.

""
FMGR-011

DCB NOT OPEN
AN ATTEMPT WAS MADE TO ACCESS AN UNOPENED DCB. USE THE CREATE OR OPEN CALL TO OPEN THE DCB AND CHECK FOR ERRORS.

""
FMGR-010

NOT ENOUGH PARAMETERS
ONE OR MORE OF THE REQUIRED PARAMETERS WERE OMITTED FROM THE CALL. ENTER THE REQUIRED PARAMETERS.

""
FMGR-009

ATTEMPT TO USE APOSN OR FORCE TO 1 A TYPE 0 FILE
A TYPE 0 FILE CANNOT BE POSITIONED WITH APOSN OR BE FORCED TO A TYPE 1 FILE. CHECK THE FILE TYPE.

""
FMGR-008

FILE OPEN OR LOCK REJECTED
AN ATTEMPT WAS MADE TO OPEN A FILE THAT WAS ALREADY OPENED EXCLUSIVELY OR WAS ALREADY OPENED TO EIGHT PROGRAMS, OR THE CARTRIDGE CONTAINING THE FILE IS LOCKED. USE THE CL OR DL COMMAND TO LOCATE THE LOCK. IF THE FILE IS BEING PACKED, CHECK TO SEE IF SPOOLING IS SHUT DOWN.

""
FMGR-007

ILLEGAL SECURITY CODE OR ILLEGAL WRITE ON LU2 OR 3
1. AN ATTEMPT WAS MADE TO ACCESS A FILE WITHOUT SPECIFYING THE SECURITY CODE OR WITH THE WRONG SECURITY CODE. FIND OUT THE CORRECT CODE AND USE IT OR DO NOT ACCESS THE FILE. OR
2. AN ATTEMPT WAS MADE BY A SESSION USER (NOT THE SYSTEM MANAGER) TO WRITE ON LU 2 OR 3. SESSION USERS DO NOT HAVE WRITE ACCESS TO LU 2 OR 3.

""
FMGR-006

FILE NOT FOUND
AN ATTEMPT WAS MADE TO ACCESS A FILE THAT CANNOT BE FOUND. CHECK THE FILE NAME.

""
FMGR-005

RECORD LENGTH ILLEGAL
AN ATTEMPT WAS MADE TO READ OR POSITION A FILE TO A RECORD THAT HAS NOT BEEN WRITTEN, OR TO WRITE AN ILLEGAL RECORD LENGTH ON AN UPDATE. CHECK THE FILE POSITION OR SIZE PARAMETER.

""
FMGR-004

MORE THAN 32767 RECORDS IN A TYPE 2 FILE
AN ATTEMPT WAS MADE TO CREATE A TYPE 2 FILE WITH TOO MANY RECORDS OR WITH A RECORD SIZE THAT IS TOO LARGE. CHECK THE SIZE PARAMETER.

""
FMGR-003

BACKSPACE ILLEGAL
AN ATTEMPT WAS MADE TO BACKSPACE A DEVICE (OR TYPE 0 FILE) THAT CANNOT BE BACKSPACED. CHECK THE DEVICE TYPE.

""
FMGR-002

DUPLICATE FILE NAME
A FILE ALREADY EXISTS WITH THE NAME SPECIFIED. REPEAT THE COMMAND WITH A NEW NAME OR PURGE THE EXISTING FILE.

""
FMGR-001

DISC ERROR
THE DISC IS DOWN. TRY AGAIN AND THEN REPORT THE PROBLEM TO THE SYSTEM MANAGER.

""
FMGR 000

BREAK
THIS IS AN INFORMATIVE MESSAGE ONLY. NO ERROR HAS OCCURRED.

""
FMGR 001

DISC ERROR - LU REPORTED
THE DISC ASSOCIATED WITH THE LU REPORTED IS DOWN. REPORT THE PROBLEM TO THE SYSTEM MANAGER.

""
FMGR 002

INITIALIZE LU 2!
THIS ERROR INDICATES A REQUEST FOR THE COMMAND TO INITIALIZE THE SYSTEM DISC (LU 2). ENTER THE INITIALIZE COMMAND.

""
FMGR 003

INITIALIZE LU 3!
THIS ERROR INDICATES A REQUEST FOR THE COMMAND TO INITIALIZE THE AUXILIARY DISC (LU 3). ENTER THE INITIALIZE COMMAND.

""
FMGR 004

ILLEGAL RESPONSE TO FMGR 002 OR FMGR 003
A COMMAND OTHER THAN AN INITIALIZE COMMAND WAS ENTERED IN RESPONSE TO EITHER A FMGR 002 OR FMGR 003 ERROR. ENTER THE APPROPRIATE INITIALIZE COMMAND.

""
FMGR 005

REQUIRED TRACK NOT AVAILABLE - RELATIVE TAT POSITION REPORTED
THE FIRST TRACK SPECIFIED IN THE INITIALIZE COMMAND IS NOT AVAILABLE. RE-ENTER THE INITIALIZE COMMAND WITH THE FIRST AVAILABLE TRACK REPORTED IN THIS MESSAGE.

""
FMGR 006

FMGR SUSPENDED
THE FILE MANAGER SUSPENDED ITSELF. READY THE DOWN DEVICE AND ENTER 'GO,FMGR'.

""
FMGR 007

CHECKSUM ERROR
A CHECKSUM ERROR OCCURRED WHEN READING A PAPER TAPE OR THE FILE BEING READ IS NOT BINARY (TYPE 5 OR 7). CHECK THE FILE TYPE.

""
FMGR 008

D.RTR NOT LOADED
THE PROGRAM D.RTR WAS NOT FOUND IN THE SYSTEM. LOAD D.RTR AS A PERMANENT PROGRAM.

""
FMGR 009

ID SEGMENT NOT FOUND
AN RP COMMAND WAS USED TO DEALLOCATE OR REASSIGN THE ID SEGMENT TO THE PROGRAM BEING RESTORED. THE SYSTEM LOOKS FOR A BLANK ID SEGMENT.

""
FMGR 010

INPUT ERROR
A SYNTAX ERROR IN THE STATEMENT OCCURRED. LOOK FOR A MISSING COLON (BATCH INPUT) OR EXTRA COLON (INTERACTIVE INPUT), AN UNDEFINED COMMAND, AN ERROR IN THE NAMR SUBPARAMETERS, A COMMAND THAT IS TOO LONG, ETC. RE-ENTER THE COMMAND. IF RECEIVED AFTER ENTERING AN ABORT COMMAND, THERE WERE NO ACTIVE JOBS.

""
FMGR 011

DO 'OF,XXXXX,8' ON NAMED PROGRAMS
AN ATTEMPT WAS MADE TO PACK A DISC TO WHICH THE NAMED PROGRAMS ARE STILL ALLOCATED. ENTER EITHER 'RP,NAMR,PROGRAM' OR 'OF,PROGRAM,8' TO REMOVE THE NAMED PROGRAMS.

""
FMGR 012

DUPLICATE DISC LABEL OR LU
AN ATTEMPT WAS MADE TO MOUNT A CARTRIDGE THE SAME LABEL OR LOGICAL UNIT NUMBER OF A CARTRIDGE THAT IS ALREADY MOUNTED. RE-ENTER THE THE COMMAND WITH ANOTHER LABEL OR LU, OR DISMOUNT THE DUPLICATE CARTRIDGE.

""
FMGR 013

TR STACK OVERFLOW
MORE THAN 10 NESTED TR COMMANDS HAVE BEEN USED. CORRECT THE CODING.

""
FMGR 014

REQUIRED ID SEGMENT NOT FOUND
AN ID SEGMENT CANNOT BE FOUND FOR THE SPECIFIED PROGRAM. CHECK THE PROGRAM NAME OR LOAD THE PROGRAM. A BLANK ID SEGMENT CANNOT BE FOUND FOR A PROGRAM BEING RESTORED. ENTER AN 'OF' COMMAND TO RELEASE AN ID SEGMENT.

""
FMGR 015

LS TRACK REPORT
THIS IS AN INFORMATIVE MESSAGE TO REPORT THE LOGICAL UNIT NUMBER AND TRACK OF THE CURRENT LS AREA.

""
FMGR 016

FILE MUST BE AND IS NOT ON LU 2 OR LU 3
AN ATTEMPT WAS MADE TO RESTORE A PROGRAM FILE THAT IS NOT ON THE SYSTEM OR AUXILIARY DISC. MOVE THE FILE TO LU 2 OR LU 3 AND RE-ENTER THE COMMAND.

""
FMGR 017

ID SEGMENT NOT SET UP BY RP
IN ORDER FOR AN ID SEGMENT TO BE RELEASED BY A 'RP' COMMAND, IT MUST HAVE BEEN SET UP BY A 'RP' COMMAND. TRY USING 'OF,PROGRAM' TO RELEASE THE SPECIFIED PROGRAM.

""
FMGR 018

PROGRAM NOT DORMANT
AN 'RP,NAMR,PROGRAM' COMMAND WAS ATTEMPTED WHEN THE PROGRAM IS ACTIVE. ENTER 'OF,PROGRAM' AND THEN REPEAT THE 'RP' COMMAND.

""
FMGR 019

FILE NOT SET UP BY SP ON CURRENT SYSTEM
THE PROGRAM FILE BEING RESTORED HAD A PARITY ERROR, WAS NOT SET UP CORRECTLY, OR WAS NOT SET UP BY A 'SP' COMMAND IN THE CURRENT SYSTEM. RELOAD THE PROGRAM AND TRY AGAIN.

""
FMGR 020

ILLEGAL TYPE 0 FILE
AN ATTEMPT WAS MADE TO CREATE A TYPE 0 FILE ON A LOGICAL UNIT THAT IS NOT ASSIGNED IN THE SYSTEM. RE-ENTER THE COMMAND USING ANOTHER LOGICAL UNIT.

""
FMGR 021

ILLEGAL DISC SPECIFIED
AN ATTEMPT WAS MADE TO COPY FILES TO OR FROM THE SAME DISC OR A DISC THAT IS NOT MOUNTED. MOUNT ANOTHER DISC OR USE ANOTHER ALREADY MOUNTED.

""
FMGR 022

COPY TERMINATED
COPY HAS BEEN TERMINATED AS A RESULT OF COPY ERROR. CHECK THE PARAMETERS AND THE SPECIFIED DISCS.

""
FMGR 023

DUPLICATE PROGRAM NAME
THE PROGRAM BEING RESTORED IS ALREADY DEFINED IN THE SYSTEM. CHANGE THE NAME OF THE PROGRAM, ENTER 'OF,PROGRAM', OR RELEASE THE ID SEGMENT.

""
FMGR 041

PROGRAM CANNOT BE A SEGMENT

""
FMGR 042

LU CANNOT BE SWITCHED

""
FMGR 043

LU NOT FOUND IN SST

""
FMGR 044

NO MESSAGES WAITING
CALLER ISSUED A ME COMMAND BUT THERE WERE NO MESSAGES WAITING TO BE BE READ.

""
FMGR 045

SESSION COMMAND ONLY
THE SPECIFIED COMMAND OPERATES ONLY IN THE SESSION ENVIRONMENT.

""
FMGR 046

INSUFFICIENT CAPABILITY
AN ATTEMPT WAS MADE TO EXECUTE A COMMAND THAT REQUIRES A HIGHER CAPABILITY LEVEL THAT THE CAPABILITY LEVEL DEFINED FOR THIS SESSION USER.

""
FMGR 047

SPOOL SET UP FAILED
THERE ARE NO AVAILABLE SPOOL FILES OR LOGICAL UNITS, OR THE LOGICAL UNIT TABLE IS FULL. YOU CAN TRY RUNNING THE JOB AGAIN, BUT IF THE ERROR IS FROM A LACK OF SPOOL LOGICAL UNITS OR THE LOGICAL UNIT TABLE BEING FULL YOU MUST RECONFIGURE.

""
FMGR 048

GLOBAL SET OUT OF RANGE
A GLOBAL WAS SPECIFIED OUT OF THE RANGE OF THE GLOBALS. CHECK THE PARAMETERS AND RE-ENTER THE COMMAND CORRECTLY.

""
FMGR 049

CAN'T RUN RP'ED PROGRAM
THE PROGRAM RESTORED FROM THE FILE DOES NOT EXECUTE. USUALLY THIS IS CAUSE BY ATTEMPTING TO RUN A SEGMENT OF THE SPECIFIED PROGRAM. CHECK THE PROGRAM.

""
FMGR 050

NOT ENOUGH PARAMETERS
LESS THAN THE REQUIRED NUMBER OF PARAMETERS WERE SPECIFIED. RE-ENTER COMMAND CORRECTLY.

""
FMGR 051

ILLEGAL MASTER SECURITY CODE
AN ATTEMPT WAS MADE TO RE-INITIALIZE A CARTRIDGE OR LIST FILES WITH AN INCORRECT MASTER SECURITY CODE. RE-ENTER THE COMMAND WITH THE CORRECT CODE.

""
FMGR 052

ILLEGAL LU IN RESPONSE TO 002 OR 003
AN ATTEMPT WAS MADE TO INITIALIZE THE FILE MANAGER USING A LOGICAL UNIT OTHER THAN LU 2 OR LU 3, OR AN ATTEMPT WAS MADE TO MOUNT A LOGICAL UNIT WHICH IS NOT A DISC CARTRIDGE. CHECK THE LU AND RE-ENTER THE COMMAND CORRECTLY.

""
FMGR 053

ILLEGAL LABEL OR ILABEL
THE SPECIFIED CARTRIDGE REFERENCE NUMBER OR CARTRIDGE ID IS ILLEGAL. THE CARTRIDGE REFERENCE NUMBER MUST BE A POSITIVE NON-ZERO INTEGER AND THE CARTRIDGE ID MUST BE A LEGAL FILE NAME.

""
FMGR 054

DISC NOT MOUNTED
AN ATTEMPT WAS MADE TO REFERENCE AN UNMOUNTED DISC CARTRIDGE. MOUNT THE DISC CARTRIDGE USING THE 'MC' COMMAND. IF UNDER SESSION CONTROL THE 'AC' COMMAND COULD BE USED INSTEAD TO ALLOCATE DISC SPACE WITH THE SPECIFIED CRN.

""
FMGR 055

MISSING PARAMETER
A REQUIRED PARAMETER HAS BEEN OMITTED. CHECK THE COMMAND AND RE-ENTER IT WITH THE MISSING PARAMETER.

""
FMGR 056

BAD PARAMETER
A PARAMETER WAS SPECIFIED INCORRECTLY OR A TRACK PARAMETER SPECIFIES A TRACK THAT IS OUTSIDE THE RANGE OF THE FMGR TRACKS. CHECK THE COMMAND AND RE-ENTER IT CORRECTLY.

""
FMGR 057

BAD TRACK NOT IN FILE AREA
THE SPECIFIED TRACK IS IN THE SYSTEM AREA OR IS A DIRECTORY TRACK. CORRECT THE COMMAND AND RE-ENTER IT.

""
FMGR 058

LG AREA EMPTY
AN ATTEMPT WAS MADE TO SAVE THE CONTENTS OF THE LG AREA WHICH IS EMPTY. RECOMPILE THE PROGRAM OR USE THE 'MR' COMMAND.

""
FMGR 059

REPORTED TRACK UNAVAILABLE
A RE-INITIALIZATION ATTEMPT LOWERED THE FIRST TRACK INTO THE SYSTEM AREA. THE LAST TRACK IS REPORTED. RE-ENTER THE COMMAND WITH THE FIRST TRACK SPECIFIED AS THE LAST TRACK + 8 (THE MINIMUM).

""
FMGR 060
DO YOU REALLY WANT TO PURGE THIS DISC?
A RE-INITIALIZATION ATTEMPT WILL RAISE THE FIRST TRACK OR LOWER THE DIRECTORY TRACKS INTO THE FILE AREA AND DESTROY A FILE. ENTER '??' OR 'NO' TO STOP THE REINITIALIZATION. ENTER 'YES' TO CONTINUE.

""
FMGR 061
DO A "DC" AND A "MC" ON THIS CR
AN ATTEMPT WAS MADE TO REPLACE A MOUNTED CARTRIDGE WITH AN CARTRIDGE THAT HAS NOT BEEN PREVIOUSLY INITIALIZED WITHOUT ENTERING A 'DC' AND A 'MC' COMMAND. ENTER A 'DC' AND 'MC' COMMAND FOR THIS CARTRIDGE.
NOTE: BE SURE TO DO A DC SPECIFING THE RELEASE RESOURCES "RR" OPTION.

""
FMGR 062
MORE THAN 63 DISCS
AN ATTEMPT WAS MADE TO MOUNT THE 64TH CARTRIDGE (THE LIMIT IS 63 CARTRIDGES). DISMOUNT A CARTRIDGE TO MAKE ROOM, IF POSSIBLE.

""
FMGR 063
EXCEEDING SESSION DISC LIMIT
AN ATTEMPT IS BEING MADE TO MOUNT MORE DISCS TO A SESSION THAN IS ALLOWED IN THE USER'S ACCOUNT. DISMOUNT AN UNUSED DISC AND RE-ENTER THE COMMAND. TO INCREASE YOUR ACCOUNT'S DISC LIMIT, CONSULT THE SYSTEM MANAGER.

""

FMGR 064
NO DISC AVAILABLE FROM DISC POOL
ALL DISCS IN DISC POOL ARE ALLOCATED OR THERE ARE NO DISCS AVAILABLE THAT ARE BIG ENOUGH.
THIS ERROR CAN ALSO OCCUR IF # DIRECTORY TRACKS SPECIFIED IS TOO LARGE. #DIRECTORY TRACKS SPECIFIED MUST BE A REASONABLE NUMBER IN RELATIONSHIP TO THE TOTAL NUMBER OF TRACKS ON THE DISC. IF DISC SPACE IS BEING ALLOCATED FROM THE DISC POOL AND SIZE WAS NOT SPECIFIED (I.E. FIRST FREE DISC IS ALLOCATED), THE MOUNT ROUTINE WILL CONTINUE TO SEARCH THE DISC POOL UNTIL A DISC IS FOUND THAT WIL PASS THE "REASONABLE" TEST. IN THIS CASE, IT IS POSSIBLE THAT EVEN THOUGH THERE ARE FREE DISCS IN THE POOL, NONE WILL BE ALLOCATED BECAUSE # DIRECTORY TRACKS WAS SO LARGE.

""

FMGR 065
CONFLICT IN SST DEFINITION
THE SPECIFIED LU NUMBER IS ALREADY DEFINED AS A SESSION LU IN THE USER'S SESSION SWITCH TABLE (SST). THIS WILL OCCUR IF THE USER HAS SPECIFIED A DISC LU NUMBER IN THE MOUNT COMMAND, BUT THIS NUMBER IS ALREADY DEFINED IN THE SST. IF IT IS NECESSARY TO MOUNT THIS DISC LU, CHANGE THE CONFLICTING ENTRY IN THE SST. THIS CAN BE DONE BY USING THE SL COMMAND TO REMOVE THE SST ENTRY WITH THE CONFLICTING SESSION LU AND, IF DESIRED, RE-ENTERING IT IN THE SWITCH TABLE WITH A DIFFERENT SESSION LU NUMBER.

""

FMGR 066
NO ROOM IN SST
THERE ARE NO SPARE ENTRIES LEFT IN THE SESSION SWITCH TABLE. SPARE ENTRIES CAN BE RECOVERED BY USING THE :SL,LU,- COMMAND, WHERE LU IS A SESSION LOGICAL UNIT NUMBER THAT IS NOT NEEDED.

""

FMGR 067
PROGRAM NOT FOUND OR ILLEGAL FMGR COMMAND
THE PROGRAM TO BE EXECUTED WAS NOT FOUND AMONG THE SYSTEM ID SEGMENTS, NOR WAS IT FOUND AS A TYPE 6 FILE ON A SYSTEM DISC. CHECK THE PROGRAM NAME SPECIFIED FOR CORRECTNESS OR RELOAD THE PROGRAM. ON A HE (HELP) COMMAND, THE FMGR 067 ERROR INDICATES THE PROGRAM HELP COULD NOT BE FOUND. ON A WH (WHZAT) COMMAND, THE ERROR INDICATES THE PROGRAM WHZAT COULD NOT BE FOUND.

""

FMGR 068
LU NOT IN VARIABLE PART OF SST
ONLY LU'S IN THE VARIABLE PART OF THE SESSION SWITCH TABLE (SST) MAY BE DELETED.

""

FMGR 069
JOB LOGON FAILED
THE JOB ACCOUNT COULD NOT BE LOGGED ON. THE REASON FOR THE FAILURE IS PRINTED ON THE SYSTEM CONSOLE.

""

FMGR 070
SECTORS/TRACK VALUE TOO LARGE
THE SECTORS PER TRACK VALUE SPECIFIED IN THE INITIALIZE COMMAND IS LARGER THAN THE ACTUAL SECTORS PER TRACK VALUE FOR THE DISC. LET THE SECTORS PER TRACK PARAMETER DEFAULT TO THE ACTUAL SECTORS PER TRACK VALUE FOR THE DISC, OR SPECIFY A SMALLER VALUE.

""

FMGR 071
DO "EX,SP" TO SAVE OR "EX,RP" TO RELEASE PRIVATE CARTRIDGES
AN ATTEMPT WAS MADE TO LOG-OFF WITH A PRIVATE DISC(S) STILL MOUNTED TO THE USER'S SESSION. SPECIFYING "EX,RP" WILL RELEASE THE USER'S PRIVATE DISC(S); IF THE DISC WAS ALLOCATED FROM THE DISC POOL, IT IS RETURNED TO THE POOL FOR POSSIBLE RE-ALLOCATION TO ANOTHER USER. IF "EX,SP" IS SPECIFIED, THE USER'S PRIVATE DISC(S) WILL REMAIN MOUNTED TO THIS USER; ON THE NEXT LOG-ON BY THIS USER, THE DISC(S) WILL BE MOUNTED TO THE NEW SESSION. NOTE THAT GROUP DISCS ARE, BY DEFAULT, LEFT MOUNTED AT LOG-OFF. TO RELEASE GROUP DISCS AT LOG-OFF, SPECIFY "EX,,RG".

""

FMGR 072
LU NOT INTERACTIVE
THE LOGICAL UNIT SPECIFIED IN A CT COMMAND MUST REFER TO AN INTERACTIVE DEVICE.

""

FMGR 073
ACCOUNT NOT FOUND
AN ATTEMPT WAS MADE TO SEND A MESSAGE TO A USER FOR WHOM AN ACCOUNT DOES NOT EXIST. CHECK THE USER.GROUP NAME OR THE ORDER OF THE PARAMETERS IN THE SM COMMAND FOR CORRECTNESS.

""

FMGR 074
JO COMMAND EXPECTED
THE FIRST COMMAND IN A JOB MUST BE, AND WAS NOT, A JO COMMAND.

""

FMGR 075
CAN'T RESTORE TYPE 6 PGM (USER PROTECTED)
THE SPECIFIED PROGRAM IS SAVED AS A TYPE 6 FILE WITH USER PROTECTION ("SP,PROG,PR"). IT CAN ONLY BE RUN OR RP'ED FROM THE TYPE 6 FILE BY THE USER WHO ISSUED THE SP COMMAND, OR BY USERS WHO ARE LINKED TO THE ACCOUNT OF THE USER WHO ISSUED THE SP COMMAND.

""

FMGR 076
CAN'T RESTORE TYPE 6 PGM (GROUP PROTECTED)
THE SPECIFIED PROGRAM IS SAVED AS A TYPE 6 FILE WITH GROUP PROTECTION ("SP,PROG,GR"). IT CAN ONLY BE RUN OR RP'ED FROM THE TYPE 6 FILE BY USERS BELONGING TO THE SAME GROUP AS THE USER WHO ISSUED THE SP COMMAND.

""

FMGR 077
CAN'T RESTORE TYPE 6 PGM (INSUFFICIENT CAPABILITY)
THE SPECIFIED PROGRAM IS SAVED AS A TYPE 6 FILE WITH CAPABILITY LEVEL PROTECTION ("SP,PROG,,CAP", WHERE CAP IS THE MINIMUM CAPABILITY LEVEL REQUIRED TO RUN OR RP THE PROGRAM). THE PROGRAM CAN ONLY BE RUN OR RP'ED FROM THE TYPE 6 FILE BY USERS POSSESSING A CAPABILITY LEVEL GREATER THAN OR EQUAL TO THE LEVEL SPECIFIED WHEN THE PROGRAM WAS SP'ED. FOR EXAMPLE, THE COMMAND "SP,PROG,,50" WILL SAVE PROGRAM "PROG" AND ONLY USERS WITH A CAPABILITY LEVEL OF 50 OR GREATER WILL BE ALLOWED TO RUN OR RP THE PROGRAM FROM THE TYPE 6 FILE. NOTE THAT COMMAND CAPABILITY CHECKING IS STILL IN EFFECT. (THE USER STILL MUST HAVE SUFFICIENT CAPABILITY TO INVOKE THE RU OR RP COMMAND, REGARDLESS OF THE CAPABILITY LEVEL SPECIFIED IN THE SP COMMAND.)

""

FMGR 078
CAN'T RESTORE TYPE 6 PGM (INTERNAL ERROR)
INTERNAL CONSISTENCY CHECKS HAVE FAILED WHILE ATTEMPTING TO RESTORE A PROGRAM FILE.

""

FMGR 079
WARNING - RECORDS TRUNCATED TO 128 WORDS
IN A TYPE 2 FILE, RECORDS WHICH ARE LONGER THAN 128 WORDS HAVE BEEN TRUNCATED TO 128 WORDS.

""

System and Break-Mode Command Error Messages

""
OP CODE ERROR

ILLEGAL OPERATOR REQUEST CODE. ENTER CORRECT OPCODE.

""
NO SUCH PROG

THE NAME ENTERED IS NOT A MAIN PROGRAM IN THE SYSTEM. ENTER CORRECT PROGRAM NAME OR LOAD PROGRAM.

""
INPUT ERROR

A PARAMETER IS ILLEGAL. ENTER COMMAND WITH CORRECT PARAMETER.

""
ILLEGAL STATUS

PROGRAM IS ALREADY SCHEDULED. CHECK STATUS WITH "ST" COMMAND. EITHER WAIT UNTIL PROGRAM TERMINATES ITSELF OR OFF IT WITH "OF" COMMAND AND RE-ENTER "RU" COMMAND.

""
CMD IGNORED-NO MEM

NOT ENOUGH SYSTEM AVAILABLE MEMORY EXISTS FOR STORING THE PROGRAM'S COMMAND STRING. RE-ENTER THE COMMAND (RU,ON,GO) OR ENTER THE INHIBIT FORM (IH) OF THE COMMAND.

""
ILLEGAL PART'N

PARTITION DOES NOT MATCH COMMAND REQUEST. RE-ENTER COMMAND WITH CORRECT PARAMETER NUMBER.

""
SIZE ERROR

ILLEGAL PROGRAM SIZE SPECIFIED OR SIZE OF PROGRAM SPECIFIED LARGER THAN ITS ASSIGNED PARTITION OR ANY PARTITION. RE-ENTER COMMAND WITH CORRECT SIZE OR ADJUST PROGRAM SIZE WITH "SZ" COMMAND.

""

EDITR Error Messages

""

EDITR ABORTED

IF THIS MESSAGE IS DISPLAYED AFTER YOU PRESS "A", TO ABORT EDITR, THERE IS NO ERROR. OTHERWISE, THE RTE SYSTEM OR THE FILE MANAGEMENT SYSTEM HAS ABORTED EDITR. THE SOURCE FILE REMAINS UNCHANGED, THE DESTINATION FILE IS DELETED.

""

??

SYNTAX ERROR IN THE COMMAND JUST GIVEN TO EDITR, OR THE INPUT DEVICE HAS TIMED OUT WAITING FOR A COMMAND. IF SYNTAX ERROR, TRY TYPING IN THE COMMAND AGAIN.

""

EOF

A COMMAND HAS CAUSED AN ATTEMPT TO READ BEYOND THE CURRENT END OF THE SOURCE FILE. YOU CAN RETURN TO THE BEGINNING OF THE FILE BY TYPING IN A 1 TO SPECIFY THAT THE FIRST LINE BECOME THE NEW PENDING LINE AND RETRY THE COMMAND.

""

CORRUPT FILE

FILE HAS A RECORD OF MORE THAN 150 CHARACTERS. USUALLY CAUSED BY LS POINTING AT SOMETHING BESIDES A FILE. CHECK LS POINTER. RERUN EDITR USING BACK UP COPY OF SOURCE.

""

COMPL and CLOAD Error Messages

""
CL- 01

THE INPUT TO THE COMPL & CLOAD PROGRAMS MUST BE A SOURCE FILE. THESE PROGRAMS DO NOT ACCEPT INPUT FROM AN LU. THUS THE ANSWER TO THE PROMPT

NAMR(S),NAMR(L),NAMR(R),<C.S.>

MUST NOT CONTAIN AN LU FOR THE 1ST PARAMETER IE THE SOURCE NAMR.

""
CL- 02

NO CONTROL STATEMENT WAS SPECIFIED SO COMPL OR CLOAD OPENED THE SOURCE FILE TO FIND OUT WHICH LANGUAGE TO INVOKE (IE FTN4, ASMB). AN FMP ERROR WAS DETECTED ON THE OPEN REQUEST. THIS FMP ERROR WAS LISTED ALONG WITH THE CL- 02 ERROR MESSAGE.

""
CL- 03

NO CONTROL STATEMENT WAS SPECIFIED SO COMPL OR CLOAD OPENED THE SOURCE FILE TO FIND OUT WHICH LANGUAGE TO INVOKE (IE FTN4,ASMB). WHILE SCANNING THE FILE FOR THE CONTROL STATEMENT AN FMP READ ERROR OCCURED. THIS ERROR WAS LISTED ALLONG WITH THE CL- 03 ERROR MESSAGE.

""
CL- 04

NO CONTROL STATEMENT WAS SPECIFIED SO COMPL OR CLOAD OPENED THE SOURCE FILE TO FIND OUT WHICH LANGUAGE TO INVOKE (IE FTN4,ASMB). THAT CONTROL STATEMENT MAY OR MAY NOT HAVE BEEN FOUND. HOWEVER, AN FMP ERROR WAS DETECTED DURING THE CLOSE OF THE FILE. THAT ERROR WAS LISTED ALONG WITH THE CL- 04 MESSAGE.

""
CL- 05

COMPL & CLOAD RECOGNIZE THE EXISTENCE OF ALL H-P SUPPLIED LANGUAGES AND SOME NOT SUPPLIED BY H-P. THE LANGUAGES IT RECOGNIZES ARE FTN4, PASCL, ASMB, COBOL, RPG, MICRO, SPL, ALGOL, HPAL, AND SNOBL. THE CONTROL STATEMENT MUST BE SPELLED EXACTLY AS SHOWN.
IF NO CONTROL STATEMENT WAS SPECIFIED AND THE CONTROL STATEMENT OF THE PROGRAM WAS NOT IN THE FIRST 10 LINES OF THE PROGRAM, THEN A CL- 05 ERROR WILL RESULT.

""
CL- 06

THE LANGUAGE REQUESTED WAS FOUND AND INVOKED BY COMPL OR CLOAD, HOWEVER, THE EXEC 23 REQUEST MADE BY CLOAD OR COMPL WAS REJECTED BY THE OPERATING SYSTEM. THIS ERROR COULD ONLY HAPPEN IF THE LANGUAGE WAS PURGED FROM THE SYSTEM BETWEEN THE 'RP' AND THE EXEC REQUEST. IF YOU GET THIS ERROR, TRY AGAIN. IF IT HAPPENS AGAIN REPORT IT TO THE SYSTEM MANAGER.

""
CL- 07

THIS ERROR MAY OCCUR WHEN THE LANGUAGE REQUESTED IN THE OPTIONAL CONTROL STATEMENT OR THE SOURCE FILE CONTROL STATEMENT WAS RECOGNIZED BUT THE LANGUAGE WAS NOT FOUND. COMPL & CLOAD BOTH TRY TO SCHEDULE THE REQUESTED LANGUAGE, FAILING THAT, THEY BOTH TRY TO ' RP ' THE LANGUAGE. IF THAT FAILS THEN THE LANGUAGE DOES NOT EXIST ON THE SYSTEM. IF THIS ERROR OCCURS FOR A LANGUAGE THAT WAS PREVIOUSLY ON THE SYSTEM, CONTACT THE SYSTEM MANAGER AS THE LANGUAGE HAS BEEN REMOVED FROM THE SYSTEM.

""
CL- 08

THE LANGUAGE REQUESTED EXISTS ON THE SYSTEM AND COMPL OR CLOAD WAS IN THE PROCESS OF ' RP 'ING IT. WHEN THE FILE WAS CLOSED AN FMP ERROR OCCURED. THAT ERROR WAS LISTED WITH THE CL- 08 ERROR MESSAGE.

""
CL- 09

THE LANGUAGE REQUESTED EXISTS ON THE SYSTEM AND COMPL OR CLOAD WAS IN THE PROCESS OF ' RP ' ING IT. HOWEVER, THAT ' RP ' FAILED BECAUSE THE CHECKSUM CALCULATED WHEN THE LANGUAGE WAS 'SP' ED DID NOT MATCH THE SYSTEM CHECKSUM. GENERALLY THIS ERROR MEANS THAT THE PROGRAM WAS NOT LOADED ON THIS SYSTEM BUT THAT THE ABSOLUTE MEMORY IMAGE OF THE PROGRAM (TYPE 6 FILE) WAS BROUGHT OVER TO THIS SYSTEM VIA A FMGR 'ST' OR 'DU' COMMAND. PROGRAMS TO BE RUN ON THIS SYSTEM MUST BE LOADED ON THIS SYSTEM WITH THE LOADR PROGRAM OR THE GENERATOR. NO OTHER METHOD OF CREATING ABSOLUTE PROGRAMS IS ALLOWED. THE FILE CONTAINING THE LANGUAGE AND ALL ITS SEGMENT FILES SHOULD BE PURGED AND THE PROGRAM LOADED WITH THE LOADR.

""
CL- 10

THE LANGUAGE REQUESTED EXISTS ON THE SYSTEM AND COMPL OR CLOAD WAS IN THE PROCESS OF 'RP'ING THE LANGUAGE. HOWEVER, DURING THE OPEN REQUEST AN FMP ERROR OCCURED. THIS ERROR WAS REPORTED WITH THE CL- 10 ERROR MESSAGE.

""
CL- 11

THIS SESSION HAS MORE THAN 80 SPOOL FILES CURRENTLY RESIDING ON THE SPOOL DISC. CLOAD AND COMPL USE FILE NAMES CONSTRUCTED AS FOLLOWS:

CHAR 1 & 1	= CO
CHAR 3 & 4	= SESSION # (01 - 99) THIS IS THE NUMBER LISTED IN THE BREAK POINT MODE S = XX COMMAND ? THE XX IS THE USERS SESSION #
CHAR 5 & 6	(01 - 80) THIS IS JUST A COUNTER THE FILES WOULD BE CREATED AS COXX01 THEN COXX02 AND SO ON.

THESE FILES CONTAIN THE OUT SPOOLED LISTING. THE CL- 11 ERROR MEANS THAT 80 OF THESE FILES ALREADY EXIST AND NO MORE WILL BE CREATED FOR THIS SESSION.
NOTE THAT RU,COMPL,SOURCE,6:NS WILL INHIBIT SPOOLING TO LU 6. THAT IS, A ´ 6:NS ´ IN THE LIST NAMR POSITION WILL INHIBIT SPOOLING AND BYPASS THIS ERROR CONDITION.

""
CL- 12

THE COMPILER WAS ABORTED AND THUS THE COMPLILATION WAS NOT SUCCESSFULLY COMPLETED. THE ABNORMAL END WAS PROBABLY DUE TO AN ´ OF ´ COMMAND. IF THE ABNORMAL END WAS DUE TO OTHER TYPE COMPILER ERRORS THE ERROR WILL BE ON THE LISTING OR REPORTED TO YOUR TERMINAL. TRY THE COMPILATION AGAIN. IF IT FAILS AGAIN CONSULT YOUR SYSTEM MANAGER.

""
CL- 13

THE COMPILATION WAS NOT SUCCESSFUL. ERRORS OR WARNINGS WERE FOUND. YOUR BEST BET IS TO GO GET THE LISTING, CORRECT THE ERROR, AND TRY AGAIN. GOOD LUCK !

""
CL- 14

THIS ERROR RESULTS WHEN THE SYSTEM IS OUT OF ID SEGMENTS AND IT IS IMPOSSIBLE TO ´ RP ´ THE COMPILER OR LOADR. GO GET THE SYSTEM MANAGER AS HE IS THE ONLY ONE WHO WILL KNOW WHICH ID SEGMENTS CAN BE DONE AWAY WITH. AFTER SOME ID SEGMENTS ARE FREE TRY AGAIN AND THE COMPILATION SHOULD WORK.

""
CL- 30

CLOAD WAS TRYING TO ´ RP ´ THE LOADR BUT ENCOUNTERED AN FMP ERROR ON THE CLOSE OF THE FILE THAT CONTAINED THE LOADR. THE FMP ERROR WAS LISTED WITH THE CL- 30 ERROR. YOU SHOULD REPORT THIS TO THE SYSTEM MANAGER.

""
CL- 31

CLOAD WAS TRYING TO ' RP ' THE LOADR AND A CHECKSUM ERROR RESULTED. THIS COULD ONLY OCCUR IF THE LOADR WAS NOT LOADED ON THIS SYSTEM BUT WAS BROUGHT OVER TO THIS SYSTEM VIA A FMGR 'ST' OR 'DU' COMMAND. THIS ERROR IS A SERIOUS ONE AND THE SYSTEM MANAGER SHOULD BE CONSULTED.

""
CL- 32

CLOAD WAS TRYING TO ' RP ' THE LOADR BUT ENCOUNTERED AN FMP ERROR ON THE FMP OPEN REQUEST. YOU SHOULD REPORT THIS TO THE SYSTEM MANAGER.

""
CL- 33

THIS SHOULD BE AN IMPOSSIBLE ERROR ! THE ONLY WAY THIS COULD HAPPEN IS IF THE LOADR WAS NOT LOADED AT GENERATION TIME OR IF AN ILLEGAL NON SUPPORTED MEMORY OR DISC MODIFICATION HAS BEEN MADE. REPORT THIS TO THE SYSTEM MANAGER IMMEDIATELY !

""
CL- 34

THE LOADR WAS LOADING YOUR PROGRAM BUT WAS ABORTED ABNORMALLY. THIS WAS PROBABLY THE RESULT OF AN ' OF ' COMMAND. ANY OTHER ABNORMAL ENDING ERROR WILL BE REPORTED TO YOUR CONSOLE. TRY THE LOAD AGAIN. IF THE ERROR OCCURS AGAIN REPORT IT TO THE SYSTEM MANAGER.

""
CL- 35

THE LOAD WAS NOT SUCCESSFUL. MORE OFTEN THAN NOT LOAD ERRORS ARE A RESULT OF UNDEFINED EXTERNALS. CHECK THE LOADR LISTING FOR THE TYPE OF ERROR. IF IT IS AN UNDEFINED EXTERNAL, THEN YOU ARE PROBABLY MISSING A SUBROUTINE SOMEWHERE. IF THIS IS THE CASE CLOAD IS NOT THE PROGRAM YOU SHOULD BE USING. RATHER, YOU SHOULD BE USING THE PROGRAMS COMPL TO COMPILE YOUR CODE AND THE LOADR TO LOAD THE SEPARATE MODULES THAT THE PROGRAM REQUIRES.

""

LOADR Error Messages

""
L-CK SUM

THIS IS A CHECKSUM ERROR. MOST LIKELY YOU SPECIFIED A FILE TO THE LOADR THAT DID NOT CONTAIN RELOCATABLE FORMAT CODE. A TYPICAL MISTAKE IS SPECIFYING THE SOURCE FILE NAME INSTEAD OF THE BINARY FILE NAME. IF THE FILE YOU SPECIFIED WAS THE CORRECT ONE THEN THAT FILE HAS BEEN OVERLAYED OR CORRUPTED. PURGE THAT FILE AND RECOMPILE THE ORIGINAL SOURCE AND TRY AGAIN.

""
L-IL REC

THE LOADR FOUND A RECORD THAT WAS NOT A NAM, ENT, EXT, DBL, EMA, GEN, LOD, OR END RECORD. THE CHECKSUM WAS OK BUT THE RECORD WAS UNIDENTIFIED. WAS THE FILE SPECIFIED A RELOCATABLE FILE ? TRY RECOMPILING AND LOADING.

""
L-OV MEM

THE SIZE OF THE CODE LOADED SO FAR EXCEEDS THE MAX SIZE THAT YOU SPECIFIED OR EXCEEDS THE LARGEST POSSIBLE SIZE FOR A PROGRAM. MAX SIZE FOR LARGE BACKGROUND (LB) NON EMA PROGRAMS IS 28K WORDS (INCLUDING BASE PAGE) AND 26K FOR LB EMA PROGRAMS. CONSULT THE GENERATION MAP FOR THE MAX SIZE OF REAL TIME AND BACKGROUND PROGRAMS. IF YOUR PROGRAM IS JUST TOO LARGE THE FOLLOWING SOLUTIONS MIGHT BE TRIED:

1. IF THE PROGRAM IS NOT TYPE 4 (LARGE BACKGROUND [LB]) MAKE IT A TYPE 4 BY SPECIFYING THE ' OP,LB ' COMMAND TO THE LOADR.
2. IF YOU SPECIFIED A SIZE, THEN DON'T SPECIFY A SIZE THE LOADR WILL DO ALL IT CAN TO MAKE YOUR PROGRAM FIT.
3. SEGMENT THE PROGRAM
4. TRY WRITING SOME OF THE PROGRAM IN ASSEMBLY
5. SEE IF THERE ARE ANY DATA DECLARATIONS THAT CAN BE REMOVED OR ANY DATA DECLARATIONS THAT CAN BE MOVED TO EMA.

""
L-OV BSE

BASE PAGE OVERFLOW. THIS PROGRAM HAS USED TOO MANY BASE PAGE LINKS. IF THE CP OPTION WAS NOT USED, TRY USING IT TO PUT LINKS ON THE CURRENT PAGE INSTEAD OF ALL ON THE BASE PAGE. IF THE CP OPTION WAS USED, RELOAD THE PROGRAM BUT THIS TIME SPECIFY THE 'OP, LE' OPTION. THIS WILL LIST ALL ENTRY POINTS AND THE BASE PAGE LINKAGES. THIS LOAD WILL ALSO FAIL, HOWEVER, NOW YOU KNOW WHICH MODULES ARE USING UP ALL THE LINKS. BY USING THE LO,XXXXX COMMAND AND ALIGNING THOSE MODULES TO PAGE BOUNDARIES THE LINKAGE NEEDS CAN BE REDUCED. ALTERNATELY YOU MAY WISH TO REARRANGE THE LOADING ORDER OF YOUR SUBROUTINES. THIS MAY IMPROVE (OR MAKE WORSE) THE LINKAGE NEEDS OF YOUR PROGRAM.

""
L-OV SYM

THIS IS A SYMBOL TABLE OVERFLOW. THE LOADR NEEDS MORE ROOM FOR ITS INTERNAL SYMBOL TABLE AND FIX UP TABLE. SINCE THE LOADR IS A TYPE 4 PROGRAM IT CAN BE MADE AS LARGE AS THE LARGEST NORMAL BACKGROUND PARTITION. TO GIVE THE LOADR MORE ROOM USE THE ' SZ ' OPERATOR COMMAND. THAT IS,

```
*SZ,LOADR,XX      XX = # OF PAGES
```

OR FROM FMGR,

```
:SYSZ,LOADR,XX
```

BY INCREASING THE SPACE FOR THE LOADR THE L-OV SYM PROBLEM SHOULD BE SOLVED. CONSULT THE RTE-IVB TERMINAL USER'S REFERENCE MANUAL FOR MORE INFORMATION ON THE 'SZ ' COMMAND.
IF THE SZ COMMAND DOES NOT SOLVE THE PROBLEM, THEN TRY USING THE LOADR ' SE ' COMMAND AFTER EVERY LOADR 'RE' COMMAND. THIS WILL REDUCE SPACE NEEDED FOR FIXUPS. IN ADDITION TO USING THE 'SE' COMMAND AFTER EVERY 'RE' COMMAND, TRY LOADING A NUMBER OF YOUR SUBROUTINES (STILL DOING 'SE') BEFORE THE MAIN OF THE PROGRAM.

""
L-CM BLK

THIS IS A COMMON BLOCK ERROR. THIS ERROR ONLY OCCURS IF THE LARGEST COMMON DECLARATION OF A PROGRAM DOES NOT APPEAR IN THE FIRST MODULE OF THE PROGRAM LOADED. PROGRAMS THAT USE COMMON MUST DECLARE THAT COMMON IN THE FIRST ROUTINE LOADED AND THAT COMMON DECLARATION MUST BE THE LARGEST ENCOUNTERED IN THE LOAD.

""

L-DU ENT

DUPLICATE ENTRY POINT. GENERALLY THIS OCCURS WHEN THE SAME SUBROUTINE WAS LOADED TWICE. ALTERNATELY YOU NAMED A SUBROUTINE WITH THE SAME NAME (ENT IN ASMB) THAT WAS ALREADY BEING USED SOMEWHERE ELSE WITHIN THE PROGRAM THAT YOU WERE TRYING TO LOAD. CONFUSION SOMETIMES OCCURS WITH SEGMENTED PROGRAMS. A SUBROUTINE LOADED WITH THE MAIN MUST NOT BE AGAIN LOADED WITH A SEGMENT. LOOK AT THE LOAD MAP FOR THE LOAD. DID YOU TRY TO LOAD THE SUBROUTINE WITH A SEGMENT WHERE THAT SUBROUTINE WAS ALREADY LOADED WITH THE MAIN ? THE LOAD MAP WILL LIST ALL SUBROUTINES LOADED WITH THE MAIN.

""

L-TR ADD

NO TRANSFER ADDRESS. ONLY SUBROUTINES WERE LOADED. THE LOADR COULD NOT TELL WHICH MODULE OF THE PROGRAM WAS THE MAIN AND WHICH ONES WERE SUBROUTINES. IF THE PROGRAM WAS WRITTEN IN FORTRAN NO MODULES WERE FOUND THAT CONTAINED THE 'PROGRAM XXXXX' STATEMENT. IF THE PROGRAM WAS WRITTEN IN ASMB YOU PROBABLY FORGOT TO PUT A LABEL ON THE END STATEMENT. IN ASMB THE MAIN OF A SEGMENT OR OF A PROGRAM IS DIFFERENTIATED FROM SUBROUTINES BY PLACING THE LABEL OF WHERE THE PROGRAM OR SEGMENT IS TO START EXECUTION AS THE OPERAND OF THE END STATEMENT. IF MULTIPLE ROUTINES HAVE LABELS ON THE END STATEMENT THE FIRST ONE ENCOUNTERED IS USED AS THE MAIN OF THE PROGRAM.

""

L-RE SEQ

RECORD OUT OF SEQUENCE. THE LOADR WAS RELOCATING AND ENCOUNTERED RECORDS IN THE WRONG ORDER. RELOCATABLE RECORDS ARE IN THE ORDER OF GEN/LOD, NAM, ENT, EXT, DBL, AND END. GENERALLY THIS ERROR OCCURS WHEN RELOCATING FROM AN LU, SAY A MAG TAPE, AND THE TAPE IS INCORRECTLY POSITIONED. IF THE RELOCATION WAS FROM A FILE, RECOMPILE THE SOURCE AND TRY AGAIN, AS THE FILE IS CORRUPT.

""

L-IL PRM

THE RUN STRING SUBMITTED TO THE LOADER WAS IN ERROR. TRY AGAIN.

""

L-CO RES

ATTEMPT TO REPLACE A MEMORY RESIDENT PROGRAM. YOU TRIED TO REPLACE A MEMORY RESIDENT PROGRAM. THIS IS ILLEGAL.

""
L-OV FIX

THIS IS A FIXUP TABLE OVERFLOW. THE LOADR NEEDS MORE ROOM FOR ITS INTERNAL SYMBOL TABLE AND FIX UP TABLE. SINCE THE LOADR IS A TYPE 4 PROGRAM IT CAN BE MADE AS LARGE AS THE LARGEST NORMAL BACKGROUND PARTITION. TO GIVE THE LOADR MORE ROOM USE THE ' SZ ' OPERATOR COMMAND. THAT IS,

```
*SZ,LOADR,XX      XX = # OF PAGES
```

OR FROM FMGR,

```
:SYSZ,LOADR,XX
```

BY INCREASING THE SPACE FOR THE LOADR THE L-OV SYM PROBLEM SHOULD BE SOLVED. CONSULT THE RTE-IVB TERMINAL USER'S REFERENCE MANUAL FOR MORE INFORMATION ON THE ' SZ ' COMMAND. IF THE SZ COMMAND DOES NOT SOLVE THE PROBLEM, THEN TRY USING THE LOADR ' SE ' COMMAND AFTER EVERY LOADR 'RE' COMMAND. THIS WILL REDUCE SPACE NEEDED FOR FIXUPS. IN ADDITION TO USING THE 'SE' COMMAND AFTER EVERY 'RE' COMMAND, TRY LOADING A NUMBER OF YOUR SUBROUTINES (STILL DOING 'SE') BEFORE THE MAIN OF THE PROGRAM.

""
L-LM LIB

THE LIMIT ON THE NUMBER OF LIBRARIES SPECIFIED BY THE 'LI' COMMAND HAS BEEN EXCEEDED. YOU MAY SPECIFY 10 LIBRARIES. INSTEAD OF SPECIFYING ANOTHER LIBRARY YOU CAN SPECIFICALLY DO A 'SE' OF THE FILE.

""
L-IL REL

THE COMPILER PRODUCED AN ILLEGAL RECORD. ONE OF THE FOLLOWING OCCURRED: THE NUMBERS OF ENTRIES SPECIFIED IN AN ENT OR EXT RECORD WAS ZERO. THE NUMBER OF INSTRUCTION WORDS SPECIFIED IN A DBL RECORD WAS ZERO. A RELOCATABLE INDICATOR IN A DBL RECORD WAS SEVEN. A DBL RECORD WAS PRODUCED THAT REFERENCED AN EXTERNAL BUT THAT EXTERNAL WAS NOT IN ANY OF THE EXT RECORDS. ALL OF THE ABOVE ARE IMPOSSIBLE CONDITIONS. RECOMPILE AND TRY AGAIN. THIS COULD ALSO BE A COMPILER BUG.

""
L-IL PTN

YOU SPECIFIED A PARTITION IN THE LOAD OF THE PROGRAM, HOWEVER, THAT PARTITION DOES NOT EXIST OR HAS BEEN DOWNED DUE TO A PARITY ERROR. TRY AGAIN, THIS TIME SPECIFY A PARTITION THAT EXISTS OR DON'T SPECIFY ANY PARTITION AT ALL.

""
L-RQ PGS

THE NUMBER OF PAGES THAT YOU SPECIFIED IN THE LOAD OF THE PROGRAM EXCEEDS THAT NUMBER OF PAGES IN THE PARTITION YOU SPECIFIED. EITHER SPECIFY A DIFFERENT PARTITION OR NO PARTITION AT ALL.

""

L-OV PTN

THE SPECIFIED PROGRAM SIZE IS TOO LARGE FOR THE PARTITION. EITHER SPECIFY A SMALLER SIZE OR NO SIZE AT ALL. SEE ALSO L-OV MEM ERROR FOR OTHER ALTERNATIVES.

""

L-ML EMA

ILLEGAL EMA DECLARATION. TWO DIFFERENT EMA LABELS WERE USED, OR THE EMA DECLARATION WAS NOT MADE IN THE MAIN OF A PROGRAM AND THAT MAIN LOADED FIRST, OR AN EMA LABEL WAS ALSO DECLARED AS AN ENTRY POINT IN ANOTHER MODULE. THE EMA DECLARATION MUST BE IN THE MAIN OF THE PROGRAM AND THAT MAIN MUST BE THE FIRST MODULE LOADED. THE EMA STATEMENT MUST BE IN ANY SEGMENT OR SUBROUTINE REFERENCING ANY ELEMENT IN EMA.

""

L-ID EXT

NO ID EXTENSIONS AVAILABLE FOR THE EMA PROGRAM. YOU MUST FREE UP SOME ID EXTENSIONS BEFORE THE EMA PROGRAM CAN BE SUCCESSFULLY LOADED.

""

L-SZ EMA

THE PROGRAMS DECLARED EMA SIZE IS TOO LARGE FOR THIS SYSTEMS PARTITIONS DEFINITION, IE THERE IS NO EXISTING PARTITION LARGE ENOUGH TO RUN THIS PROGRAM. EITHER REBOOT AND RECONFIGURE SYSTEM TO ALLOW MORE EMA SPACE OR DECLARE LESS EMA SPACE IN THE PROGRAM.

""

L-SS ENT

YOU ATTEMPTED TO ACCESS AN SSGA ENTRY POINT BUT YOU DID NOT ASK FOR SSGA AT THE BEGINNING OF THE LOAD. RELOAD THE PROGRAM BUT THIS TIME DO A 'OP,SS' AT THE BEGINNING OF THE LOAD.

""

L-IL CMD

ATTEMPT TO PURGE A PROGRAM UNDER BATCH OR ATTEMPT TO USE THE 'LI' OR 'PU' COMMANDS WITHIN A LOADR COMMAND FILE. LI AND PU COMMANDS ARE NOT ALLOWED WITHIN A LOADR COMMAND FILE UNLESS THAT COMMAND FILE IS AN INTERACTIVE DEVICE (IE A TTY OR CRT).

""

L-ID SEG

NOT ENOUGH LONG AND SHORT ID SEGMENTS TO FINISH THE LOAD. THIS IS AN EXTREMELY RARE ERROR. THE LOADR WAS CREATING ID SEGMENTS AND THERE WERE ENOUGH ID SEGMENTS AT THE BEGINNING TO FINISH THE LOAD, HOWEVER, BETWEEN CREATING ONE ID SEGMENT AND CREATING THE NEXT ALL OTHER ID SEGMENTS WERE USED UP (MAYBE ANOTHER LOADR OR FILE MANAGER GOT THEM) AT ANY RATE THERE AREN'T ENOUGH TO FINISH THE LOAD. THE PROPER RESPONSE TO THIS ERROR IS TO ' OF ' OR PURGE ALL SEGMENTS AND THE MAIN OF THE LOAD THAT WAS JUST UNSUCCESSFUL, FREE UP SOME ADDITIONAL ID SEGMENTS AND TRY THE LOAD AGAIN. IF ENOUGH ID SEGMENTS ARE FREED UP THE LOAD WILL SUCCEED. THIS ERROR COULD ONLY OCCUR IN SEGMENTED LOADS.

""

L-RF EMA

ATTEMPT TO ACCESS AN EMA EXTERNAL WITH OFFSET OR INDIRECT. IF THIS IS A FORTRAN PROGRAM YOU MORE THAN LIKELY FORGOT TO PUT THE $EMA STATEMENT IN A SUBROUTINE THAT ACCESSED AN EMA ELEMENT. IF THE PROGRAM WAS WRITTEN IN ASMB USE THE H-P SUPPLIED ROUTINES .EMAP AND .EMIO TO MAP IN THE ARRAYS AND THEN INDEX INTO THE ARRAY VIA THE ADDRESS RETURNED, NOT VIA A REFERENCE TO THE EMA LABEL.

""

L-UN EXT

UNDEFINED EXTERNALS EXIST WHICH PROHIBITS THE LOAD FROM COMPLETING. AN UNDEFINED EXTERNAL IS A REFERENCE MADE BY THE ROUTINE YOU ARE LOADING TO ANOTHER ROUTINE. FOR EXAMPLE IF YOUR FORTRAN PROGRAM HAD THE FOLLOWING CODE :

```
CALL XYZ(I,J,K)
```

THEN THE SUBROUTINE XYZ WOULD BE AN EXTERNAL. THE PROBLEM YOU HAVE IS THAT YOU LOADED THE ROUTINE THAT CONTAINED THE CALL TO XYZ BUT YOU DIDN'T LOAD THE XYZ SUBROUTINE ITSELF. XYZ IS THE UNDEFINED EXTERNAL. THE PROPER COURSE HERE IS TO RELOAD YOUR PROGRAM BUT THIS TIME DON'T FORGET TO LOAD THE ROUTINES LISTED WHEN THE LOADR ABORTED THE LAST TIME YOU TRIED TO LOAD THE PROGRAM.
ONE LAST POINT. IT IS POSSIBLE TO FORCE LOAD A PROGRAM OR SEGMENTS THAT HAVE UNDEFINED EXTERNALS. THIS IS DONE WITH THE LOADR 'FORCE' COMMAND. HOWEVER, IF YOU FORCE LOAD THE PROGRAM IT IS YOUR RESPONSIBILITY TO MAKE SURE THAT THE LINE OF CODE THAT REFERENCES THE EXTERNAL IS NEVER EXECUTED. THAT IS, MAKE SURE THAT THE CALL TO XYZ IS NOT EXECUTED OR YOUR PROGRAM WILL PROBABLY BE ABORTED WITH A DM OR MP ERROR.

""
L-EX CPY

ATTEMPT TO REPLACE OR PURGE A PROGRAM WHERE COPIES OF THAT PROGRAM EXIST. IT IS NOT POSSIBLE TO REPLACE OR PURGE A PROGRAM FROM THE SYSTEM IF COPIES OF THAT PROGRAM EXIST. THE PROBLEM HERE IS THAT OTHER COPIES OF THE SAME PROGRAM EXIST AND MAY BE IN USE. THE PROPER COURSE HERE IS TO DO AN ' OF,PROG,8 ' ON ALL THE PROGRAMS LISTED AS COPIES. THIS WILL GET RID OF THOSE PROGRAMS SO THAT YOU CAN PERFORM THE PROGRAM PURGE OR REPLACE. NOTE THAT THIS PROCESS SHOULD ONLY BE DONE BY THE SYSTEM MANAGER.

""
L-RP CPY

ATTEMPT TO REPLACE A COPIED PROGRAM. YOU TRIED TO DO A PROGRAM REPLACE ON A PROGRAM THAT WAS A COPY OF ANOTHER PROGRAM. REPLACEMENT OPERATIONS MAY ONLY BE DONE ON THE ORIGINAL PROGRAM NOT THE COPIED PROGRAM. THE PROPER THING TO DO NOW IS EDIT THE SOURCE OF YOUR PROGRAM AND MAKE SURE THE NAME IS THE ORIGINAL PROGRAM NAME.

""
L-PE LDR

TRYING TO DO A PURGE OR PERMANENT LOAD WITH A COPY OF THE LOADR. RE-RUN THE LOADR USING THE REAL PROGRAM: RU,LOADR:IH.

""
L-DU PGM

THIS PROBLEM RESULTS WHEN YOU TRY TO LOAD THE SAME PROGRAM SEVERAL TIMES BUT DO NOT GET RID OF THE EARLIER LOADS. FOR EXAMPLE, YOU LOADED A PROGRAM CALLED XXXXX AND FOR SOME REASON LOADED THE SAME PROGRAM AGAIN. IN THIS CASE THE LOADR WARNED YOU WITH A W-DU PGM WARNING MESSAGE AND THEN RENAMED YOUR PROGRAM TO ..XXX. THAT IS THE LOADR FORGIVES YOU THE FIRST TIME. HOWEVER, YOU HAVE NOW LOADED A PROGRAM WITH THE SAME NAME A THIRD TIME. THE LOADR WILL NOT FORGIVE THIS AGAIN. THE SOLUTION IS TO DO A

```
:OF,XXXXX,8
:OF,..XXX,8
```

AND NOW START THE LOAD OVER AGAIN.

""
L-NO IDS

NOT ENOUGH ID SEGMENTS TO FINISH THE LOAD. YOUR SYSTEM HAS RUN OUT OF ID SEGMENTS. CALL THE SYSTEM MANAGER TO FREE UP SOME ID SEGMENTS. HE WILL PROBABLY USE THE OFF COMMAND TO PURGE SOME PROGRAMS FROM THE SYSTEM.

""

L-RP PGM

YOU TRIED TO REPLACE A PERMANENT PROGRAM. HOWEVER, THAT PROGRAM TERMINATED SERIALLY REUSABLE, SAVING RESOURCES, OR WAS OPERATOR SUSPENDED. THAT IS, THE PROGRAM STILL OWNED A SYSTEM PARTITION. OFF THE PROGRAM AND REPEAT THE LOAD.

""

L-IN CAP

YOU ATTEMPTED TO LOAD, PURGE, OR REPLACE A PERMANENTLY LOADED PROGRAM WITHOUT HAVING A HIGH ENOUGH SESSION CAPABILITY. A CAPABILITY LEVEL OF 60 OR HIGHER IS REQUIRED.

LOADR WARNINGS (THE RELOCATION IS NOT ABORTED)

""

W-RQ PGS

THE NUMBER OF PAGES REQUIRED EXCEEDS THE PARTITION SIZE. RECONFIGURE OR GENERATE A NEW SYSTEM CONTAINING A LARGE ENOUGH PARTITION OR REVISE THE PROGRAM.

""

W-UN EXT

UNDEFINED EXTERNALS EXIST BUT THE LOADER WAS INITIATED FROM AN INTERACTIVE DEVICE. A PROMPT IS ISSUED, DO AN SE OF THE LIBRARY CONTAINING THE REQUIRED SUBROUTINE.

""

W-DU PGM

DUPLICATE PROGRAM NAME. NO ACTION IS REQUIRED. PROGRAM XXXXX IS RENAMED ..XXX AUTOMATICALLY.

""

W-IL CMD

ATTEMPTED TO RELOCATE A MODULE OR TRANSFER TO A COMMAND FILE WHILE DOING SPECIAL PROCESSING WHEN UNDEFINED EXTERNALS EXIST. AT THIS TIME RE,XXXXX OR TR,YYYYY ARE ILLEGAL COMMANDS.

DBUGR Error Messages

""
X

THE USER PRESSED THE "RUBOUT" KEY TO ERASE A TYPING MISTAKE - DBUGR IGNORES ANY PRIOR PARTIAL EXPRESSION.

""
?

THE USER ENTERED AN UNASSIGNED CONTROL. ANY PRIOR EXPRESSION IS IGNORED.

""
U

THE SYMBOL LAST USED IS UNDEFINED, AND A DEFINITION IS REQUIRED. THE ENTIRE PRECEDING EXPRESSION IS IGNORED.

""
P?

PAGE ERROR. A MEMORY REFERENCE INSTRUCTION REFERENCED AN ADDRESS NOT IN THE CURRENT PAGE OR THE BASE PAGE. THE EXPRESSION IS IGNORED. DBUGR'S CONCEPTION OF THE "CURRENT PAGE" CAN BE CHANGED BY EXAMINING ANY LOCATION IN THE DESIRED PAGE.

""
MP?

THERE IS A BREAKPOINT OR TRACE SET FOR AN INSTRUCTION THAT IF EXECUTED BY DBUGR WOULD CAUSE A MEMORY-PROTECT VIOLATION TO OCCUR. MOVE THE BREAKPOINT AND PROCEED.

""
IN?

THERE IS A BREAKPOINT OR TRACE SET FOR AN INSTRUCTION FROM WHICH DBUGR CANNOT PROCEED. MOVE THE BREAKPOINT AND PROCEED.

""
DM?

DBUGR IS ATTEMPTING TO ACCESS A MEMORY LOCATION THAT IS NOT WITHIN THE USER'S PARTITION.

"
TP?

DBUGR IS ATTEMPTING TO OVERLOAD, TRACE, OR SET A BREAKPOINT WITHIN DBUGR.

"

READT/WRITT Error Messages

READ 001

THE REQUESTED MAG TAPE UNIT IS DOWN. USE THE "UP" COMMAND (SPECIFYING THE APPROPRIATE EQT) TO ENABLE THE DEVICE.

READ 002

THE MAG TAPE READT IS TRYING TO RESTORE CONTAINS INFORMATION IN A FORMAT NOT RESTORABLE BY READT. THE TAPE MAY HAVE BEEN SAVED WITH ANOTHER UTILITY, OR IT MAY HAVE BEEN CONSTRUCTED THROUGH THE FMGR'S "DU" OR "ST" COMMANDS. IN ANY CASE READT CANNOT RESTORE THE DATA.

THIS ERROR WILL ALSO RESULT WHEN THE NEXT TAPE OF A TWO OR MORE TAPE CARTRIDGE IS NOT THE CORRECT ONE. MOUNT THE CORRECT TAPE AND DO AS THE UTILITY SUGGESTS.

READ 003

THE MAG TAPE UNIT YOU WISH TO USE IS LOCKED TO SOME PROCESS. FIND OUT WHO CURRENTLY HAS THE MAG TAPE LOCKED (E.G., RU,WHZAT) AND WAIT UNTIL IT'S RELEASED OR HAVE THE USER RELEASE IT FOR YOU.

READ 004

THE PARAMETER DESCRIBING THE DESIRED MAG TAPE UNIT DOES NOT SATISFY READT'S REQUIREMENTS FOR A LEGAL MAG TAPE LU. THE POSSIBLE CAUSES FOR THIS ERROR INCLUDE:

1. THE SPECIFIED MAG TAPE LU IS NOT BETWEEN -63 AND +63.
2. THE DRIVER OF THE SPECIFIED LU IS NOT A MAG TAPE DRIVER.

READ 005

THE DESIRED MAG TAPE UNIT IS OFF-LINE.! THE ON-LINE BUTTON MUST BE DEPRESSED TO ENABLE THE ON-LINE SWITCH.!

READ 006

READT REJECTED THE USE OF THE SPECIFIED DISC LU. THERE ARE A VARIETY OF REASONS FOR THIS, THEY INCLUDE:

1. THE DISC LU NUMBER MUST BE A NEGATIVE NUMBER BUT NO SMALLER THAN -63.
2. THE DESIRED DISC LU IS NOT IN YOUR SST.
3. THE DRIVER TYPE OF THE REQUESTED DISC LU IS NOT A DISC DRIVER.

READ 007

THE DRIVER DETECTED A PARITY ERROR WHEN READING FROM THE MAG TAPE. IF THIS HAPPENS AGAIN THE TAPE MAY BE IRRECOVERABLE. CALL THE SYSTEM MANAGER.

READ 008

THE END OF TAPE WAS REACHED. MOUNT THE FOLLOWING TAPE TO READ THE REMAINING PORTIONS OF THE CARTRIDGE. TO CONTINUE THE PROGRAM ENTER "GO". TO HALT THE PROCESS ENTER "AB". NOTE HOWEVER THAT A REPLY OF AN "AB" WHEN RUNNING READT WILL CAUSE AN INCOMPLETE CARTRIDGE TO BE PRESENT ON THE SYSTEM.

READ 009

THE DESIRED CARTRIDGE HAS A FILE OPEN OR THE CARTRIDGE IS LOCKED TO ANOTHER PROGRAM. TRY DOING CL OR A DL ON THAT CARTRIDGE TO FIND OUT WHAT PROGRAM HAS THE CARTRIDGE LOCK OR WHAT FILES ARE OPEN.

READ 010

YOU ARE OPERATING IN A NONSESSION ENVIRONMENT. AN LU MUST BE SPECIFIED (NEGATIVE LU) SINCE THERE ISN'T A FREE DISC POOL.

READ 011

READT REJECTED THE SIZE (NUMBER OF TRACKS) YOU SPECIFIED BECAUSE IT'S OF A BAD FORMAT (E.G., NEGATIVE VALUE), OR THE SIZE REQUIRED IS NOT LARGE ENOUGH TO RESTORE THE CARTRIDGE ON MAG TAPE.

READ 012

THE ROUTINE READT USED TO MOUNT A CARTRIDGE DETECTED AN ERROR. THIS ERROR IS RETURNED IN THE FMGR FORMAT. THE FOLLOWING ARE POSSIBLE ERROR CONDITIONS. FIND THE ONE THAT APPLIES TO YOU AND DO AS SUGGESTED.

FMGR 012 DUPLICATE LABEL OR CRN ALREADY MOUNTED.
HAVE THAT DISC OR CRN RESTORED THEN RUN READT AGAIN.

FMGR 056 THE SIZE REQUESTED IS TOO LARGE FOR THE DISC LU SPECIFIED.
RUN READT AGAIN WITH A SMALLER SIZE PARAMETER.

FMGR 063 YOU CURRENTLY HAVE MOUNTED THE MAXIMUM NUMBER OF DISC CARTRIDGES IN YOUR SESSION. REMOVE ONE AND RUN READT AGAIN.

FMGR 064 THERE ARE PRESENTLY NO MORE FREE DISC LU'S IN THE DISC POOL.
HAVE SOMEONE RELEASE A CARTRIDGE THAT THEY ARE NOT CURRENTLY USING.

FMGR 065 THERE IS A CONFLICT IN SST DEFINITION. YOU ARE TRYING TO MOUNT A DISC LU THAT HAS A SESSION LU NUMBER ASSIGNED TO SOME OTHER DEVICE. CHECK YOUR SST AND FIND OUT TO WHAT LU THAT NUMBER IS ASSIGNED, THEN CHANGE IT OR CHOOSE ANOTHER DISC LU.

FMGR 066 THERE IS NO MORE ROOM IN YOUR SST TO PLACE AN ENTRY. REMOVE AN ENTRY FROM YOUR SST IF POSSIBLE. IF THAT'S NOT DESIRABLE OR POSSIBLE THEN CALL YOUR SYSTEM MANAGER.

READ 013

THE DESIRED DISC LU OR THE AVAILABLE FREE LU'S IN THE DISC POOL ARE NOT LARGE ENOUGH TO RESTORE THE CARTRIDGE THAT'S ON MAG TAPE.

READ 014

THE SYSTEM MANAGER IS THE ONLY USER ALLOWED TO RESTORE SYSTEM CARTRIDGES.

READ 015

BAD TRANSMISSION -- MEMORY TO DISC TRK XXX SEC YYY READT TRIED TO TRANSFER DATA FROM MEMORY TO A DISC LU. DURING THIS PROCESS A CHECK OF THE TRANSMISSION LOG SHOWED AN UNEXPECTED VALUE. RUN READT AGAIN, IF IT HAPPENS ONCE MORE CALL YOUR SYSTEM MANAGER.

READ 016

BAD TRANSMISSION -- MAG TAPE TO MEMORY REC XXX READT DETECTED AN ERROR IN TRANSMISSION OF DATA FROM THE MAG TAPE UNIT INTO MEMORY. TRY READING THE TAPE AGAIN. IF IT HAPPENS ONCE MORE CALL YOUR SYSTEM MANAGER.

READ 017

READT WILL NOT MOVE THE STARTING LOCATION OF FMP TRACKS ON LU 2 OR LU 3, NOR WILL IT RESTORE A CARTRIDGE WITH A SEC/TRK VALUE THAT'S DIFFERENT FROM WHAT'S FOUND ON THE DISC CARTRIDGE.

READ 018

ABORTED BY USER--THIS MESSAGE IS PRODUCED WHEN YOU RESPOND NO TO ANY PROMPT, OR WHEN READT IS HALTED BY THE BREAK COMMAND.

READ 019

DISC ERROR ON LU xx TRACK xxxx--READT ENCOUNTERED AN ERROR WHEN READING THE LISTED TRACK OF THE LISTED LU.

READ 020

VERIFY ERROR ON TRACK xxxx. A COMPARE ERROR WAS ENCOUNTERED WHEN VERIFYING THE LISTED TRACK.

WRIT 001

THE REQUESTED MAG TAPE UNIT IS DOWN. BY UPPING THE APPROPRIATE EQT THE DEVICE CAN BE ENABLED.

WRIT 002

ONLY THE SYSTEM MANAGER CAN SAVE SYSTEM DISCS.

WRIT 003

THE MAG TAPE YOU WISH TO USE IS LOCKED TO SOME PROCESS. FIND OUT WHO CURRENTLY HAS THE MAG TAPE LOCKED (E.G., RU,WHZAT) AND WAIT UNTIL IT'S RELEASED OR HAVE THE USER RELEASE IT FOR YOU.

WRIT 004

THE PARAMETER DESCRIBING THE DESIRED MAG TAPE UNIT DOES NOT SATISFY READT'S REQUIREMENTS FOR A LEGAL MAG TAPE UNIT. THE POSSIBLE CAUSES FOR THIS ERROR INCLUDE:

1. THE SPECIFIED MAG TAPE LU IS NOT BETWEEN -63 AND +63.
2. THE DRIVER OF THE SPECIFIED LU IS NOT A MAG TAPE DRIVER.

WRIT 005

THE DESIRED MAG TAPE UNIT IS OFF-LINE. THE ON-LINE BUTTON MUST BE DEPRESSED TO ENABLE THE ON-LINE SWITCH.

WRIT 006

A WRITE RING IS REQUIRED TO WRITE INFORMATION ON A MAG TAPE. PLACE A WRITE RING ON THE TAPE SPOOL AND RUN WRITT AGAIN.

WRIT 007

THE DRIVER DETECTED A PARITY ERROR WHEN WRITING TO THE MAG TAPE. TRY AGAIN, IF IT OCCURS AGAIN THEN THE TAPE MAY BE IRRECOVERABLE. CALL SYSTEM MANAGER.

WRIT 008

THE END OF THE TAPE WAS REACHED. MOUNT THE FOLLOWING TAPE TO WRITE THE REMAINING PORTIONS OF THE CARTRIDGE. TO CONTINUE THE PROGRAM ENTER "GO". TO HALT THE PROCESS ENTER "AB". NOTE HOWEVER THAT A RESPONSE OF AN "AB" WILL PLACE A PARTIALLY COMPLETED CARTRIDGE ON YOUR TAPE.

WRIT 009

THE DESIRED CARTRIDGE HAS A FILE OPEN OR THE CARTRIDGE IS LOCKED TO ANOTHER PROGRAM. TRY DOING A CL OR DL ON THAT CARTRIDGE TO FIND OUT WHAT PROGRAM HAS YOUR CARTRIDGE LOCK OR WHAT FILES ARE OPEN.

WRIT 010

THE DESIRED CARTRIDGE OR DISC LU COULD NOT BE FOUND. DO A CL (CARTRIDGE LIST) TO MAKE SURE THAT WHAT YOU'RE SEEKING IS REALLY THERE.

WRIT 011

WRITT REJECTED THE USE OF THE SPECIFIED DISC LU. THERE ARE A VARIETY OF REASONS FOR THIS, THEY INCLUDE:

1. THE DISC LU NUMBER MUST BE A NEGATIVE NUMBER BUT NO SMALLER THAN -63.
2. THE DESIRED DISC LU IS NOT IN YOUR SST.
3. THE DRIVER TYPE OF THE REQUESTED DISC LU IS NOT A DISC DRIVER.

WRIT 012

ONLY THE SYSTEM MANAGER CAN SAVE FMP TRACKS OFF LU 2 OR LU 3 (IF 3 EXISTS) WITH WRITT.

WRIT 013

WRITT TRIED TO READ DATA FROM A DISC LU INTO MEMORY AND FOUND THE TRANSMISSION IRREGULAR. RUN WRITT AGAIN, IF THE SITUATION OCCURS ONCE MORE THERE MAY BE A BAD TRACK ON THAT DISC LU. SAVE AS MUCH DATA AS YOU CAN AND NOTIFY YOUR SYSTEM MANAGER.

WRIT 014

THE TRANSMISSION OF DATA FROM MEMORY TO MAG TAPE MAY BE FAULTY. RUN WRITT AGAIN, IF IT HAPPENS ONCE MORE CALL YOUR SYSTEM MANAGER.

WRIT 016

AN ERROR WAS DETECTED IN TRANSMISSION OF DATA FROM THE MAGNETIC TAPE TO MEMORY. IF THIS ERROR RECURS, THE TAPE MAY BE FAULTY; SEE THE SYSTEM MANAGER.

WRIT 020

A COMPARE ERROR WAS ENCOUNTERED WHEN VERIFYING THE LISTED TRACK.

Appendix C
Tables, Directories, and Record Formats

Program ID Segment

Each user program has a 33-word ID segment located in memory that contains static and dynamic information defining the properties of the program. The static information is set during generation time or when the program is loaded on-line. The dynamic information is maintained by the operating system Executive.

The number of ID segments contained in a system is established during system generation, and is directly related to the number of programs that can be in main memory at any given time. If all the ID segments are in use, no more programs can be added on-line unless some other existing program is first "offed" (removed from the system) to recover an ID segment.

The format of the ID segment is illustrated in Figure C-1. Each ID segment's address is located in the Keyword Table (see location 01657).

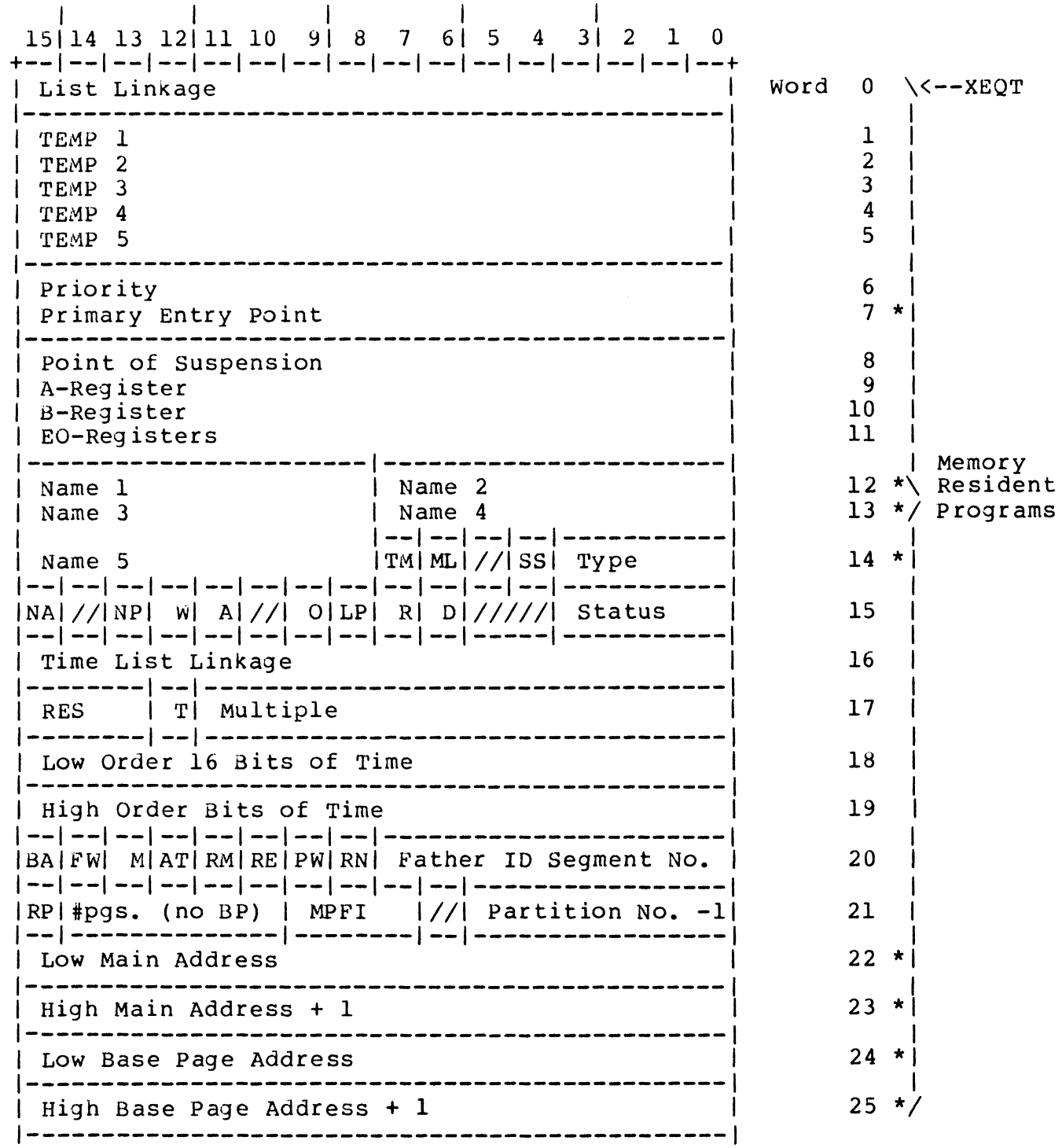
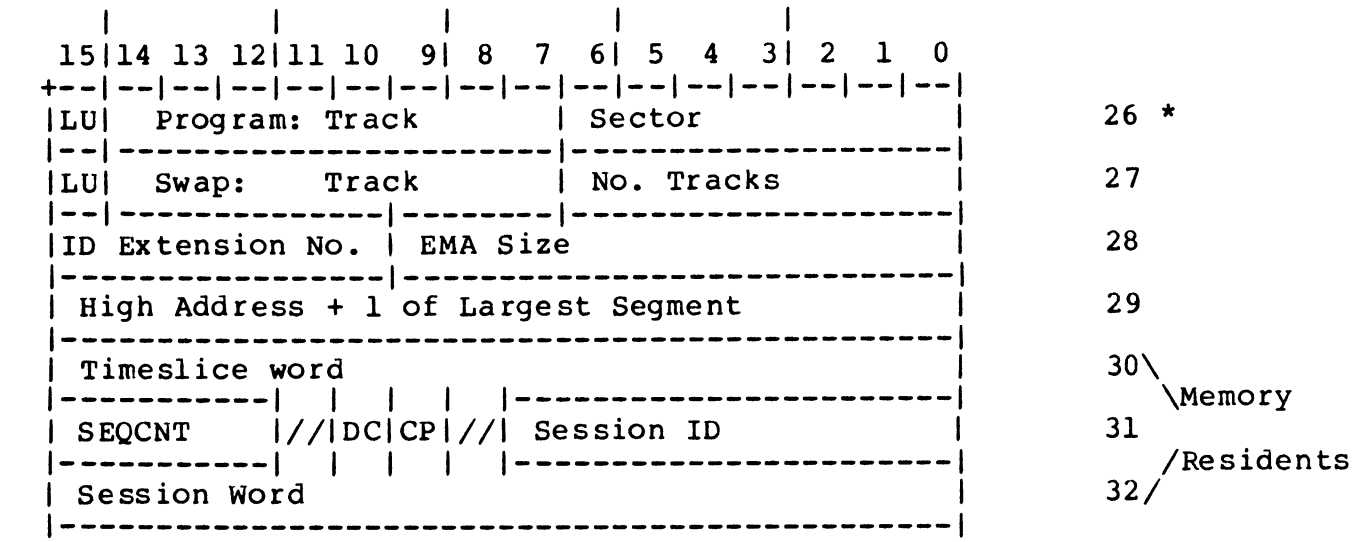


Figure C-1. ID Segment Format

Figure C-1. ID Segment Format (Continued)



where:

* = words used in short ID segments for program segments

TM = temporary load (copy of ID segment is not on the disc)

ML = memory lock (program may not be swapped)

SS = short segment (indicates a nine-word segment)

Type = specified program type (1-5)

NA = no abort (instead, pass abort errors to program)

NP = no parameters allowed on reschedule

W = wait bit (waiting for program whose ID segment address is in word 2)

A = abort on next list entry for this program

O = operator suspend on next schedule attempt

LP = load in progress; program is being dispatched from disc.

R = resource save (save resources when setting dormant)

D = dormant bit (set dormant on next schedule attempt)

Status = current program status

T = time list entry bit (program is in the time list)

BA = batch (program is running under batch)

FW = father is waiting (father scheduled with wait)

M = Multi-Terminal Monitor bit

AT = attention bit (operator has requested attention)

RM = reentrant memory must be moved before dispatching program

RE = reentrant routine now has control

PW = program wait (some other program wants to schedule this one)

RN = Resource Number either owned or locked by this program

RP = reserved partition (only for programs that request it)

MPFI = memory protect fence index

TIMESLICE WORD (30):

The timeslice word defines the timeslicing status of a program. This word is defined as follows:

1 = This program has just been rescheduled or is not timesliced.

0 = This program has used a full timeslice or program is not scheduled.

<0 = This program was running (under timeslice control) and was "bumped" from execution by a higher priority program. This word represents the remaining timeslice for this program.

OPEN FLAG WORD(31):

SEQCNT = sequence counter. Each time a program is aborted or terminates (unless saving resources) the counter is incremented. The counter value is used to build FMP open Flags.

DC = don't copy flag. Set by the generator (if 128 is added to program type) or the loader (using Don't Copy op-code).

CP = copy flag. Indicates that the program is a copy.

Session ID = System LU of terminal that program was loaded from.

SESSION WORD(32):

The session word identifies the owner of a program.

A negative value represents the logical unit number of the terminal from which the program was invoked (not under session).

A positive value represents the address of the SST length word or the session control block for the session currently using this program (under session).

Programs scheduled by interrupt will have a zero in this word.

ID Segment Extensions

Each EMA program requires a 3-word ID segment extension in addition to its 33-word ID segment. The number of ID extensions contained in the system is also set at generation time, and if all are in use, no more EMA programs can be added on-line. The format of the ID segment is illustrated in Figure C-2.

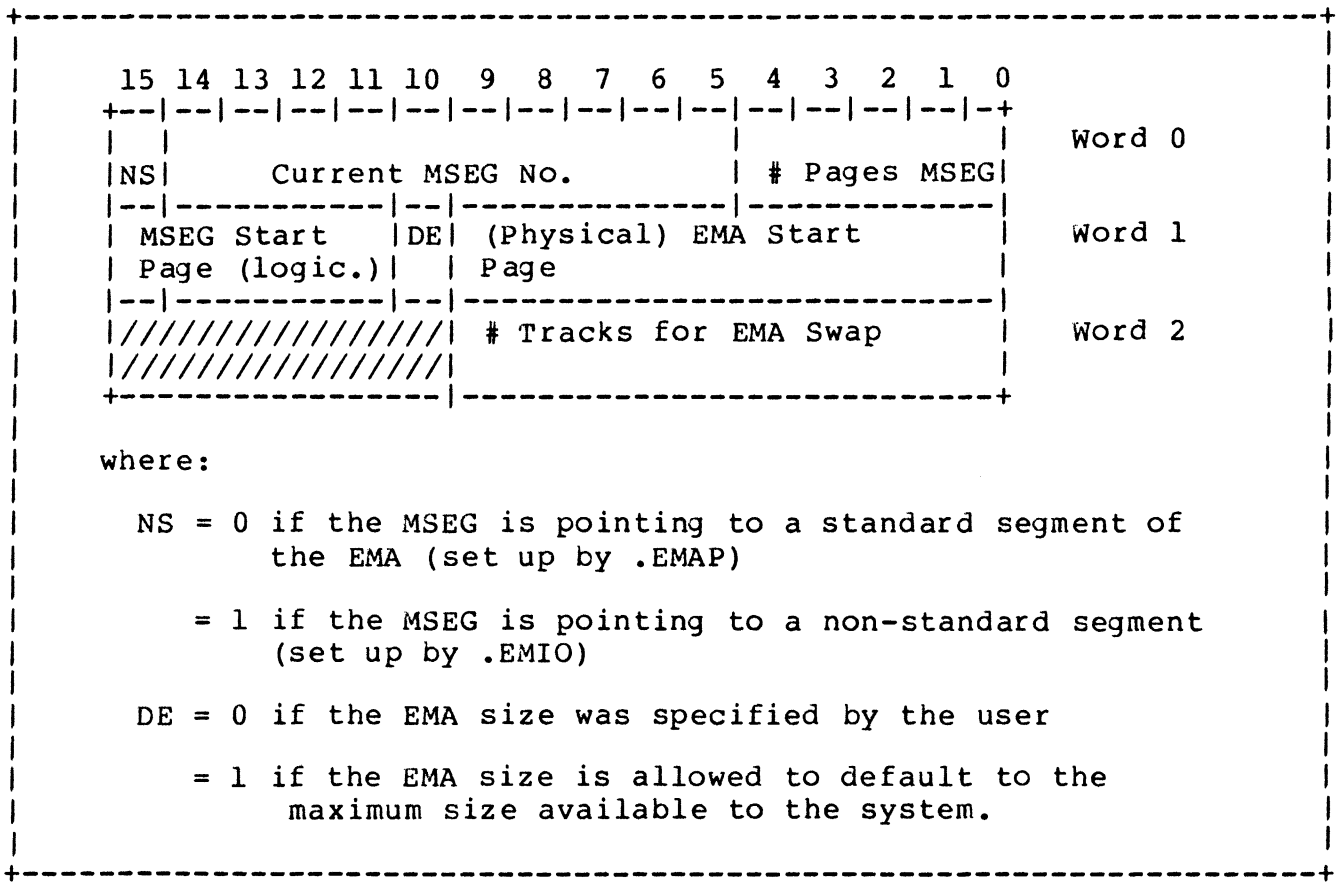


Figure C-2. ID Segment Extension

Short ID Segments

Short ID segments requiring nine words are used only for program segments. A short ID segment is required for each segment of a segmented program. If no empty short ID segments are available during an on-line load, a standard 33-word ID segment will be used. The information contained in a short ID segment is illustrated in Figure C-1.

RTE-IVB System Disc Layout

Figure C-3 illustrates how disc space is allocated when a RTE-IVB system is generated.

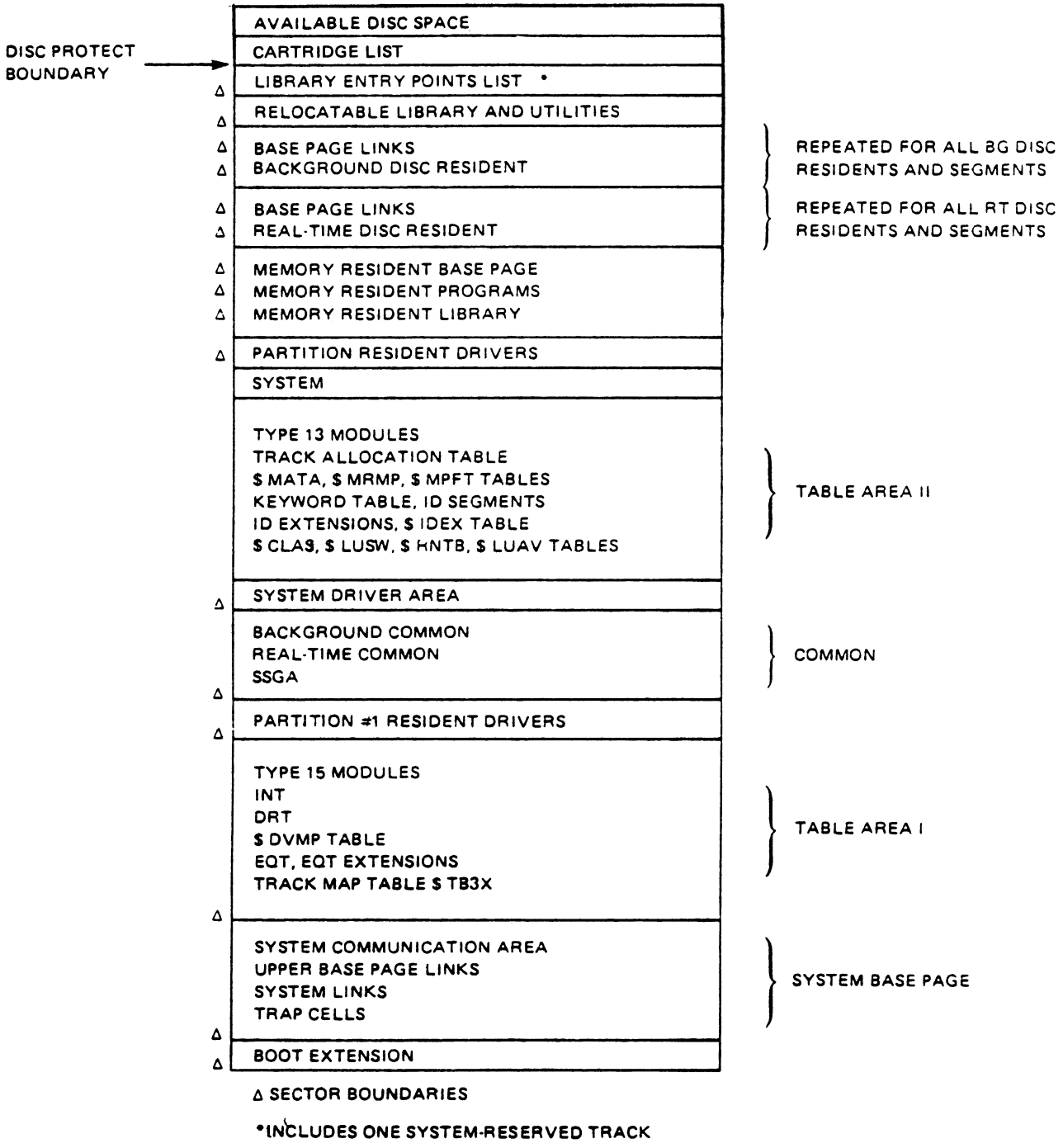


Figure C-3. RTE-IVB System Disc Layout

Source Record Formats

The source format used for the disc records by the system program EDITR and FMGR is given in Figure C-4. All records are packed ignoring sector boundaries. Binary records are packed directly onto the disc. After an END record, a zero word is written and the rest of the sector is skipped. If this zero word is the first word of the sector, it is not written. Binary files are always contiguous so a code word is not required.

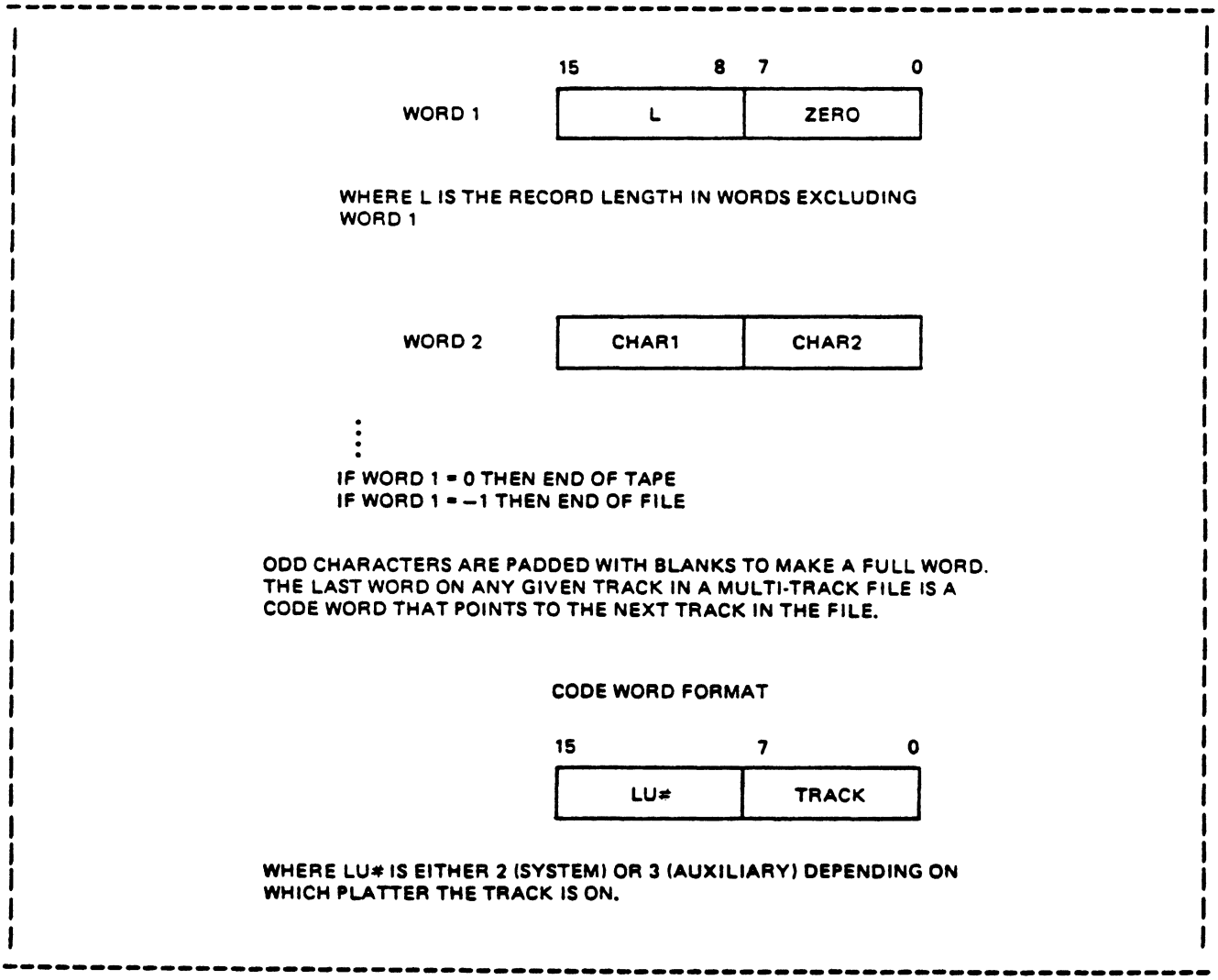


Figure C-4. Source Record Formats

Relocatable and Absolute Record Formats

The following describes the formats of relocatable and absolute records produced as object code for a given source program. The relocatable records are generated by compilers or by the assembler for a relocatable assembly. These records are stored in a relocatable file. The generator or the loader processes these relocatable records to produce an absolute module which has all program links resolved and the program is relocated and ready to run.

The absolute records are produced by the assembler for an absolute assembly. The module of records thus produced requires no processing by the generator or loader. Absolute programs must be loaded into memory and run off-line.

NAM Record

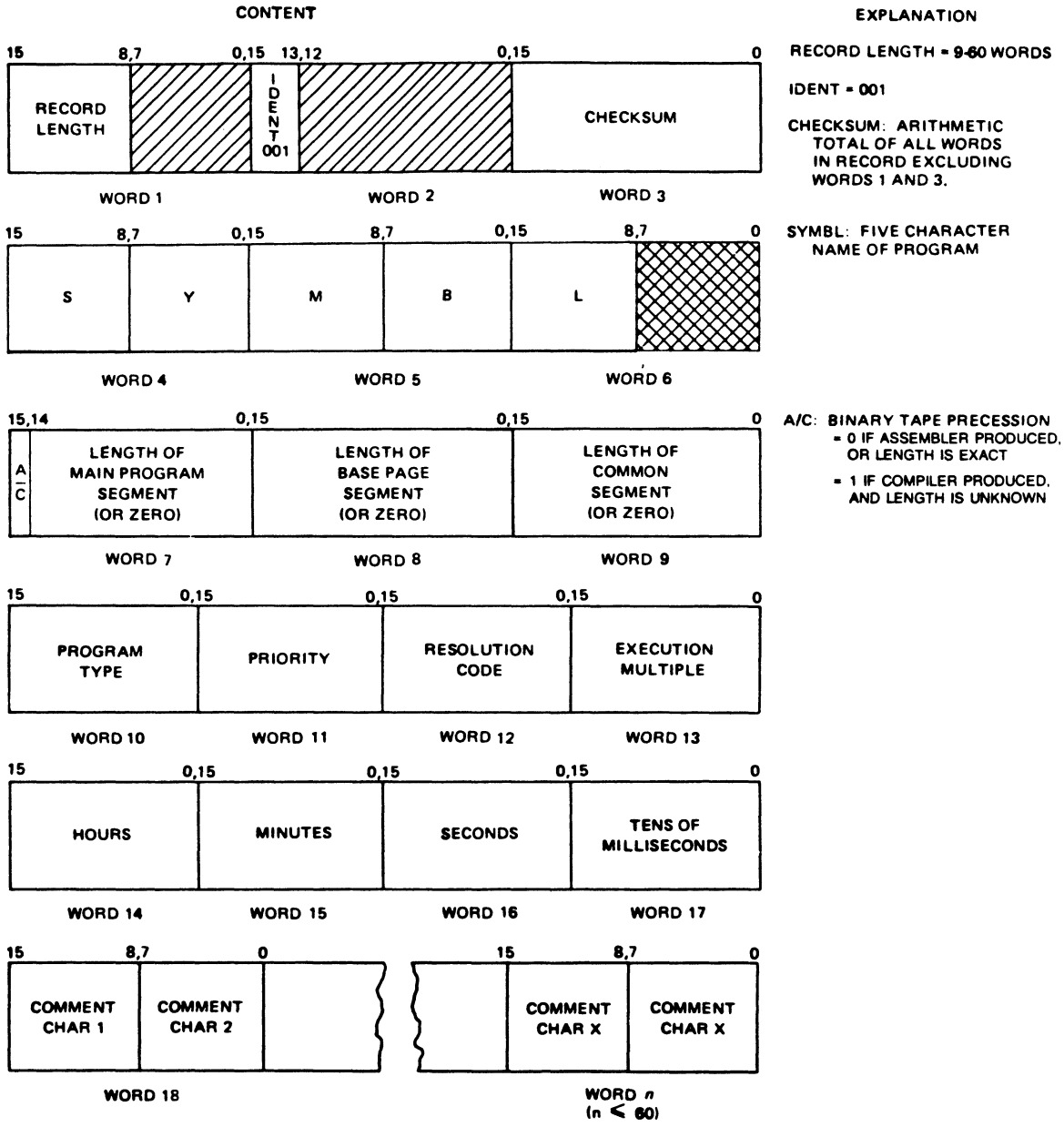


Figure C-5. Record Formats

ENT Record

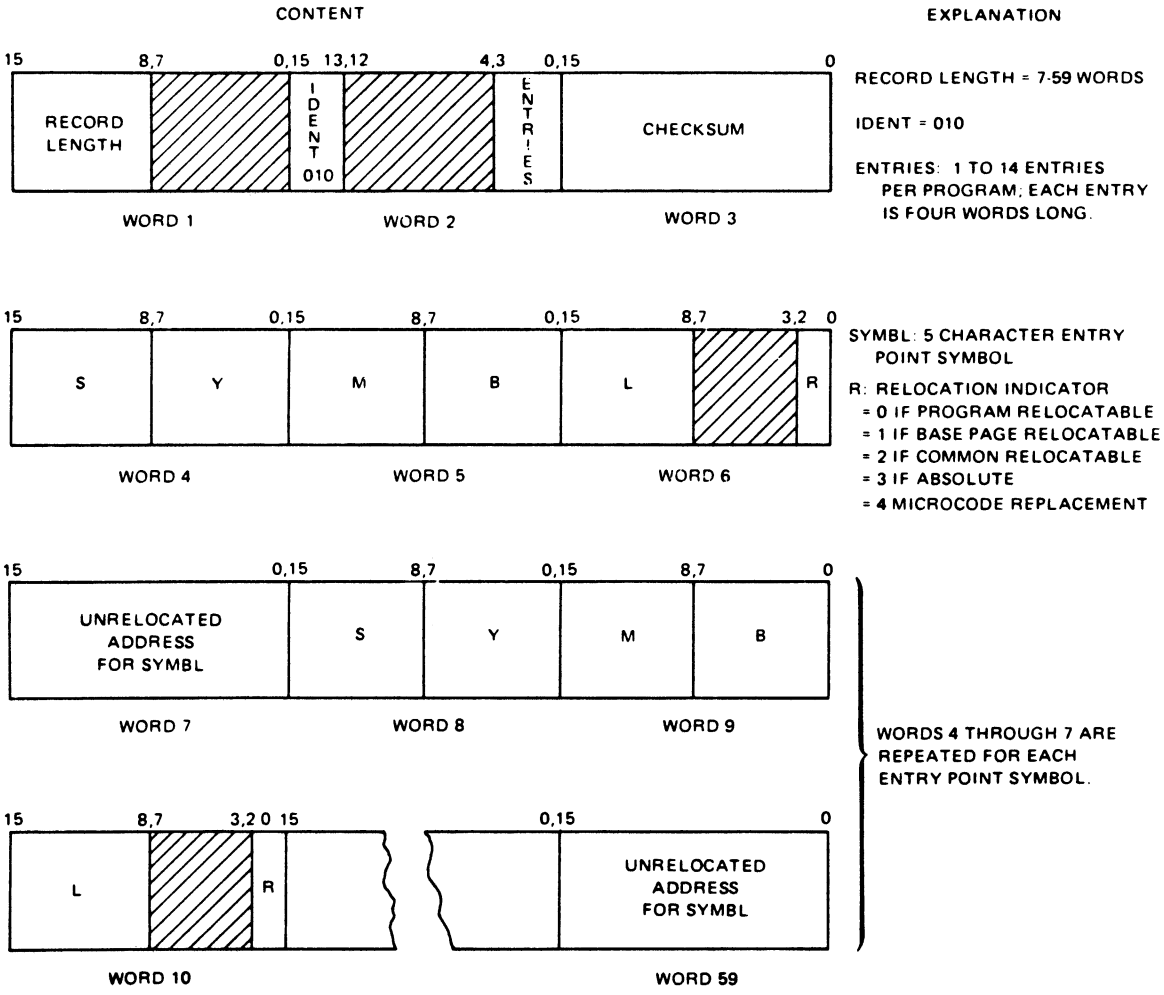


Figure C-5. Record Formats (continued)

EXT Record

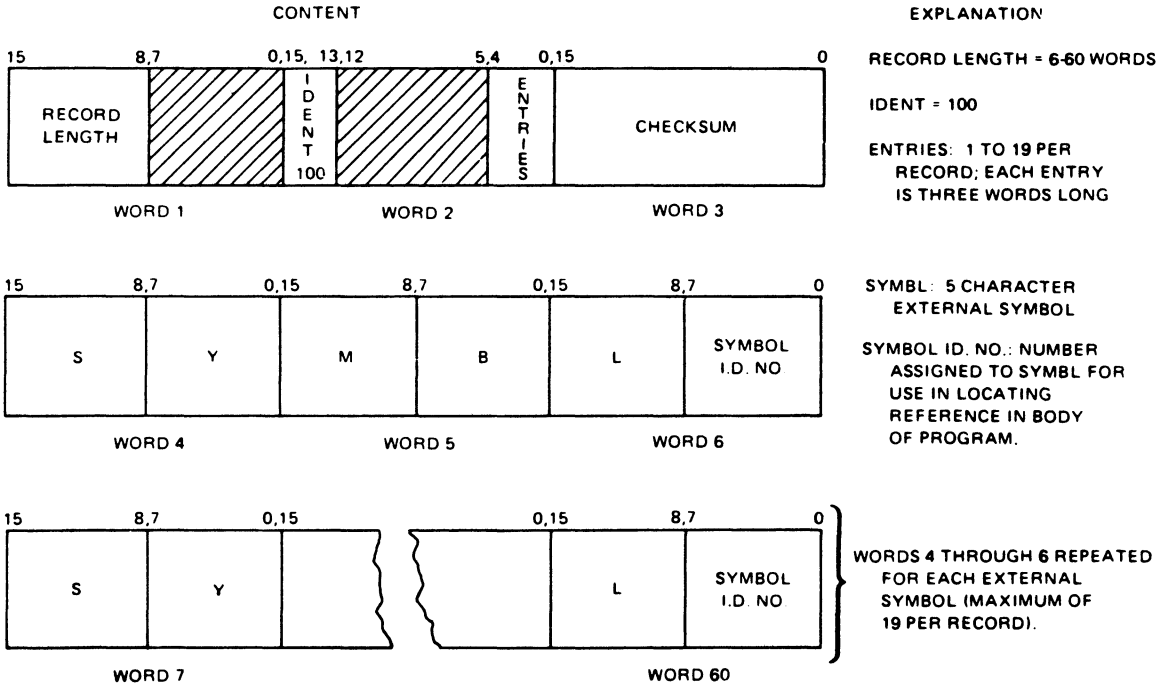


Figure C-5. Record Formats (continued)

DBL Record

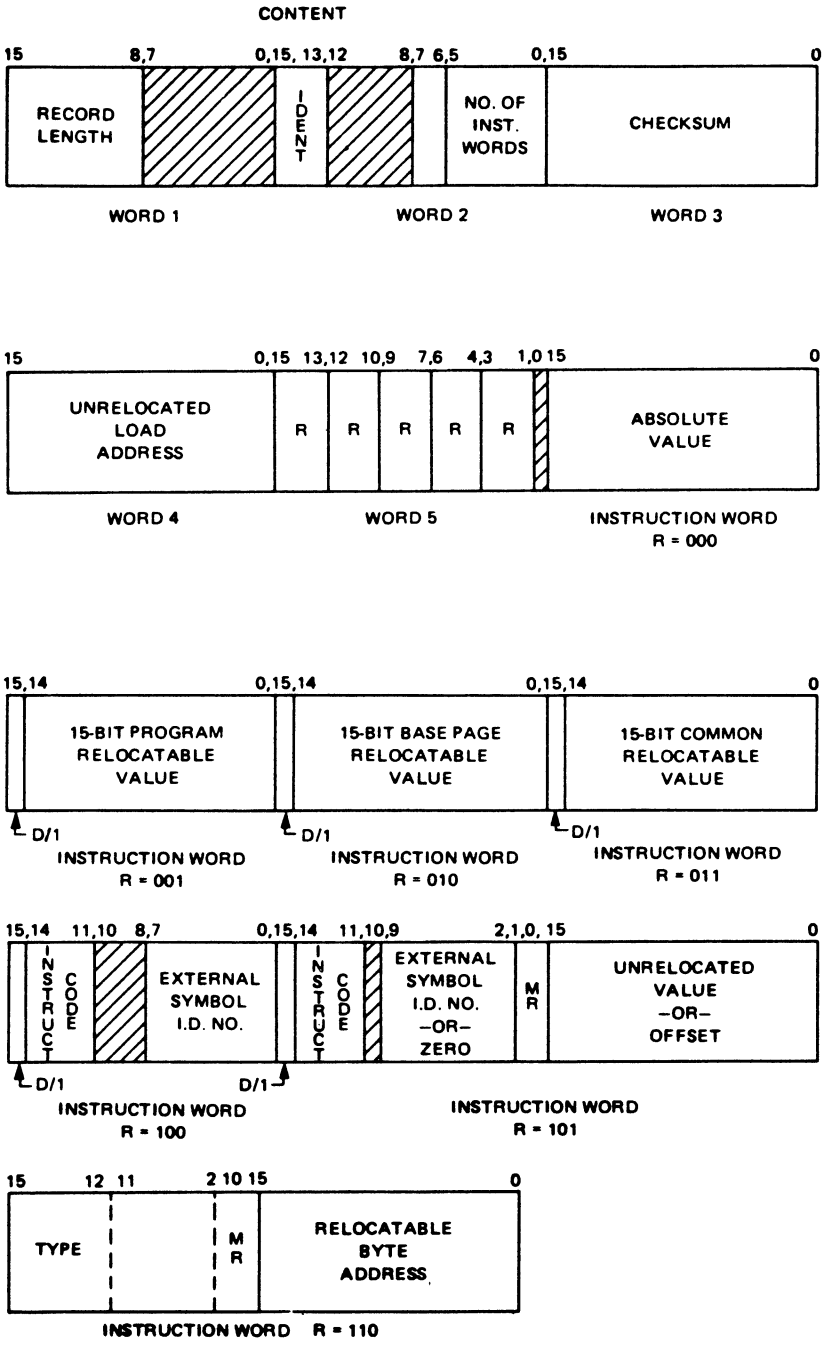
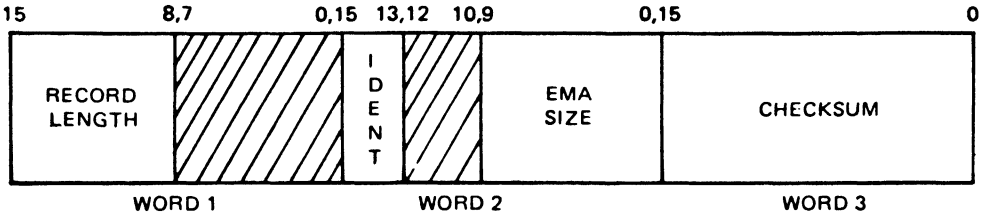


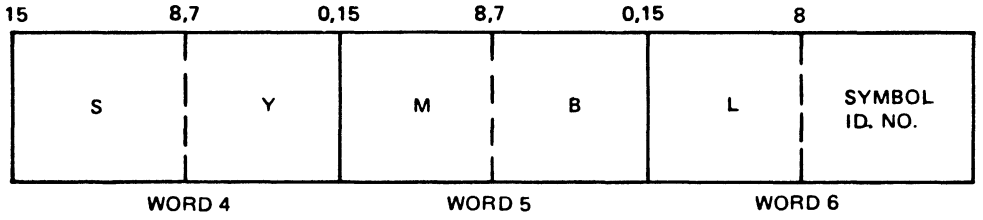
Figure C-5. Record Formats (continued)

EMA RECORD

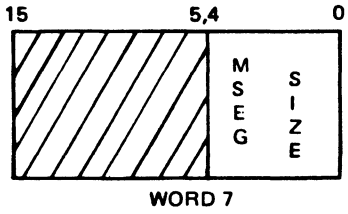


EXPLANATION

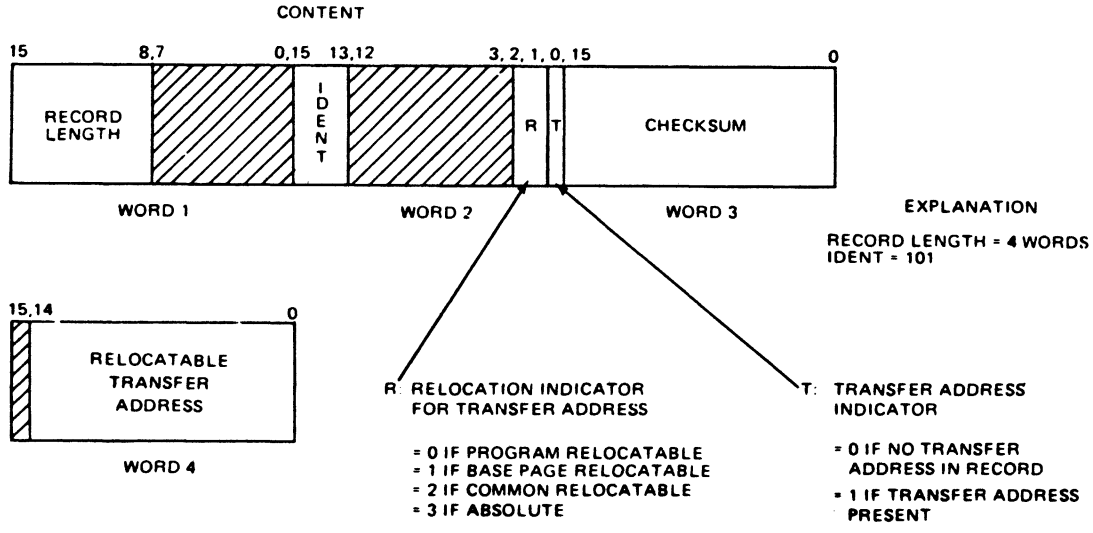
RECORD LENGTH = 7 WORD
IDENT = 110



SYMBOL ID. NO.: NUMBER ASSIGNED TO SYMBL FOR USE IN LOCATING REFERENCE IN BODY OF PROGRAM.



END Record



EXPLANATION

RECORD LENGTH = 4 WORDS
IDENT = 101

R: RELOCATION INDICATOR FOR TRANSFER ADDRESS

= 0 IF PROGRAM RELOCATABLE
= 1 IF BASE PAGE RELOCATABLE
= 2 IF COMMON RELOCATABLE
= 3 IF ABSOLUTE

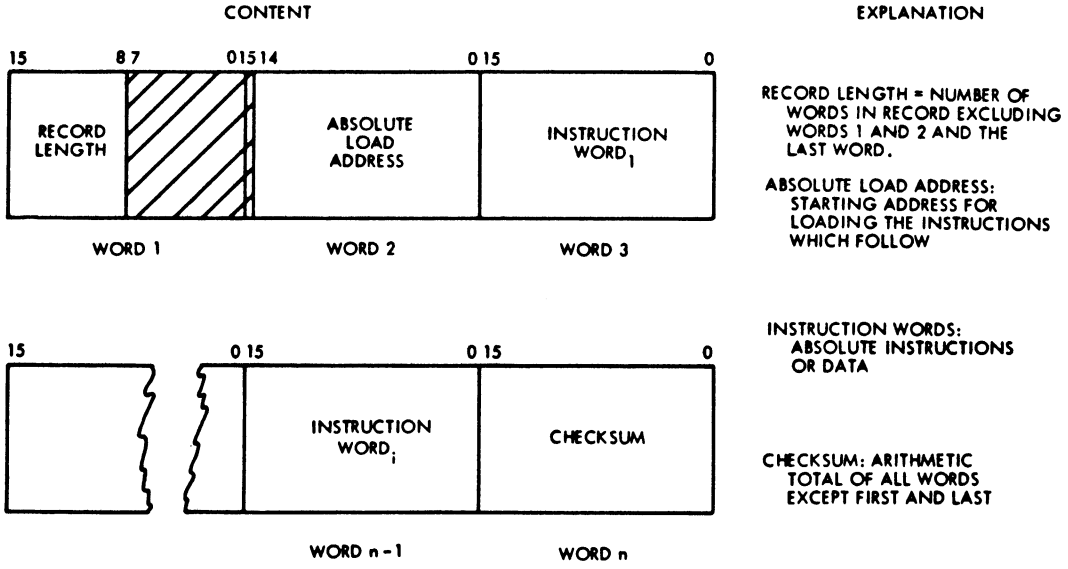
T: TRANSFER ADDRESS INDICATOR

= 0 IF NO TRANSFER ADDRESS IN RECORD
= 1 IF TRANSFER ADDRESS PRESENT

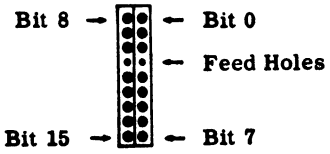
Figure C-5. Record Formats (continued)

Absolute Tape Format

Absolute binary code is written to paper tape in the following format:

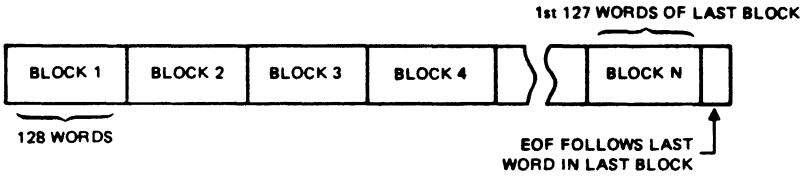


Each word represents two frames arranged as follows:



Disc File Record Formats

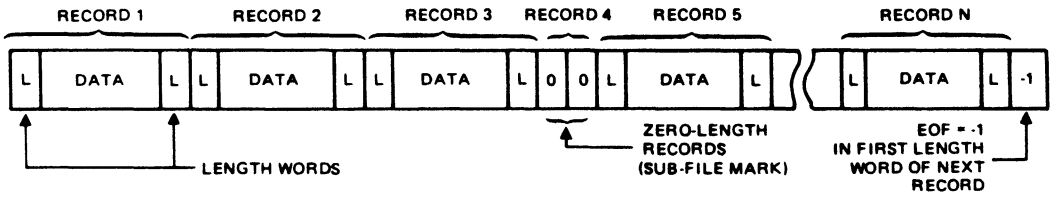
Fixed Length Formats (Types 1 and 2)



Type 1 Record length = Block length = 128 words

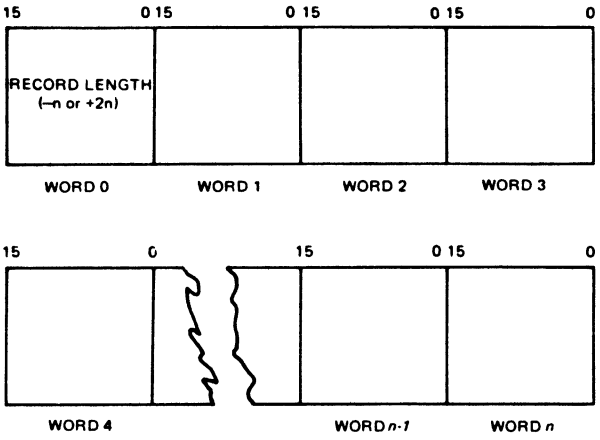
Type 2 Record length is user defined; may cross block boundaries but not past EOF

Variable Length Formats (Types 3 and Above)



SIO Record Format

Magnetic tape SIO binary records have the following format:



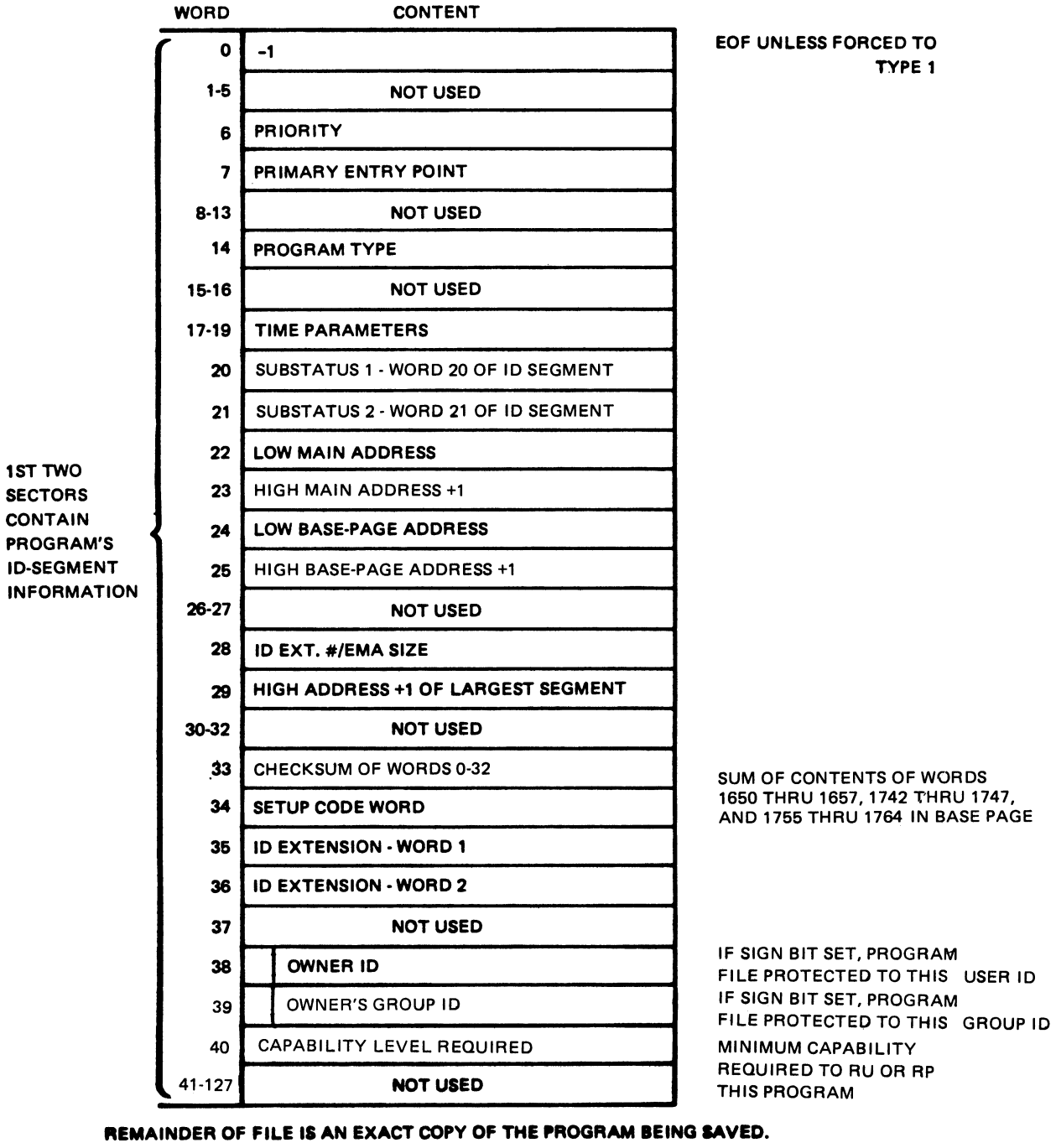
Record length = number of words or characters in record, excluding word 0; negative value denotes words, positive value denotes characters.

NOTE

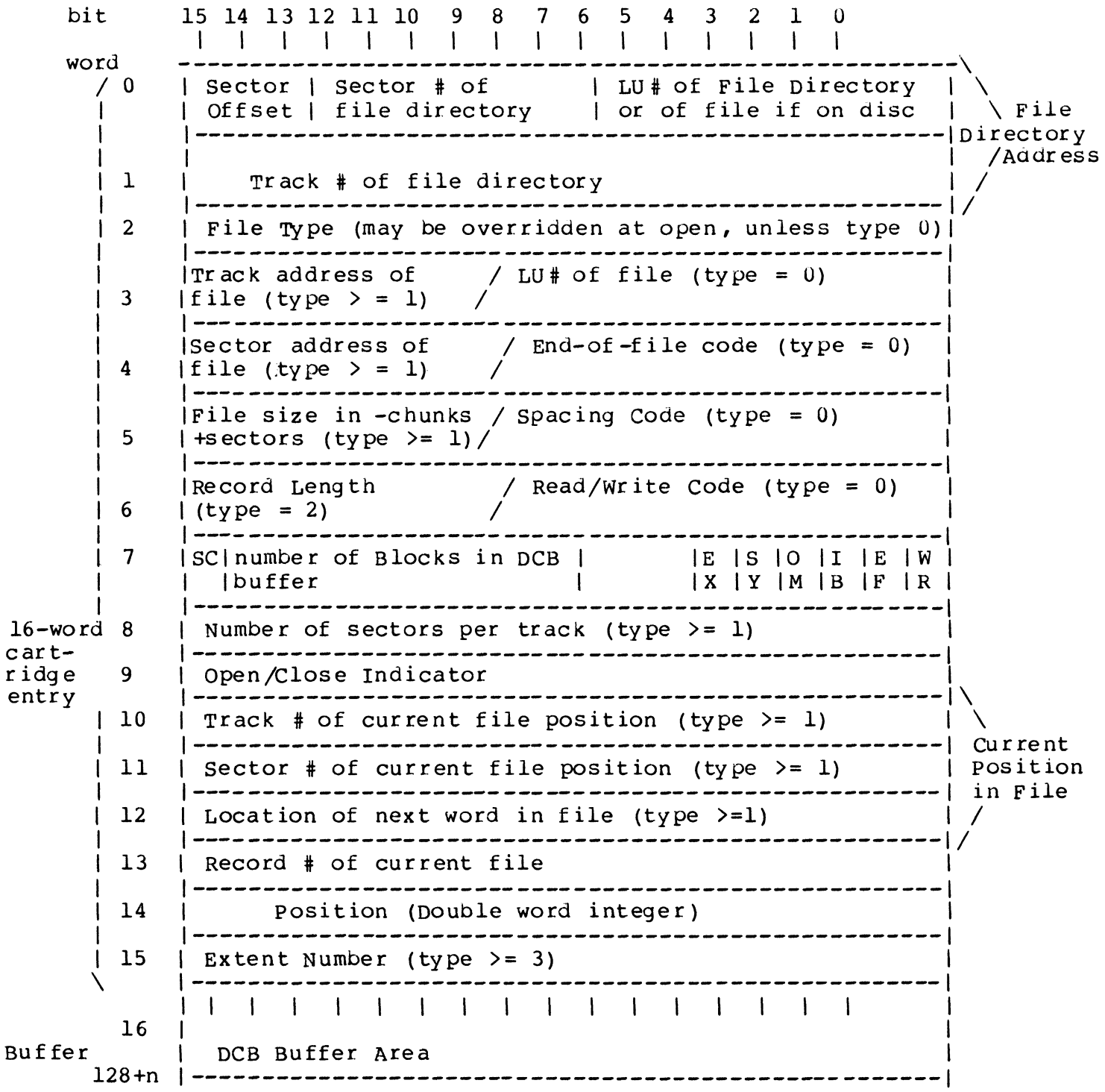
The length (word 0) is not considered part of the data record. When written with the MS option of the DU command, the length is supplied by FMGR. When read with the MS option of the ST command, the length is removed (in this case, the length word is used instead of the length supplied by the driver).

Memory-Image Program File Formats (Type 6)

Files created by the SP command as memory-image program files are always accessed as type 1 files (fixed length, 128-words per record).



Data Control Block Format

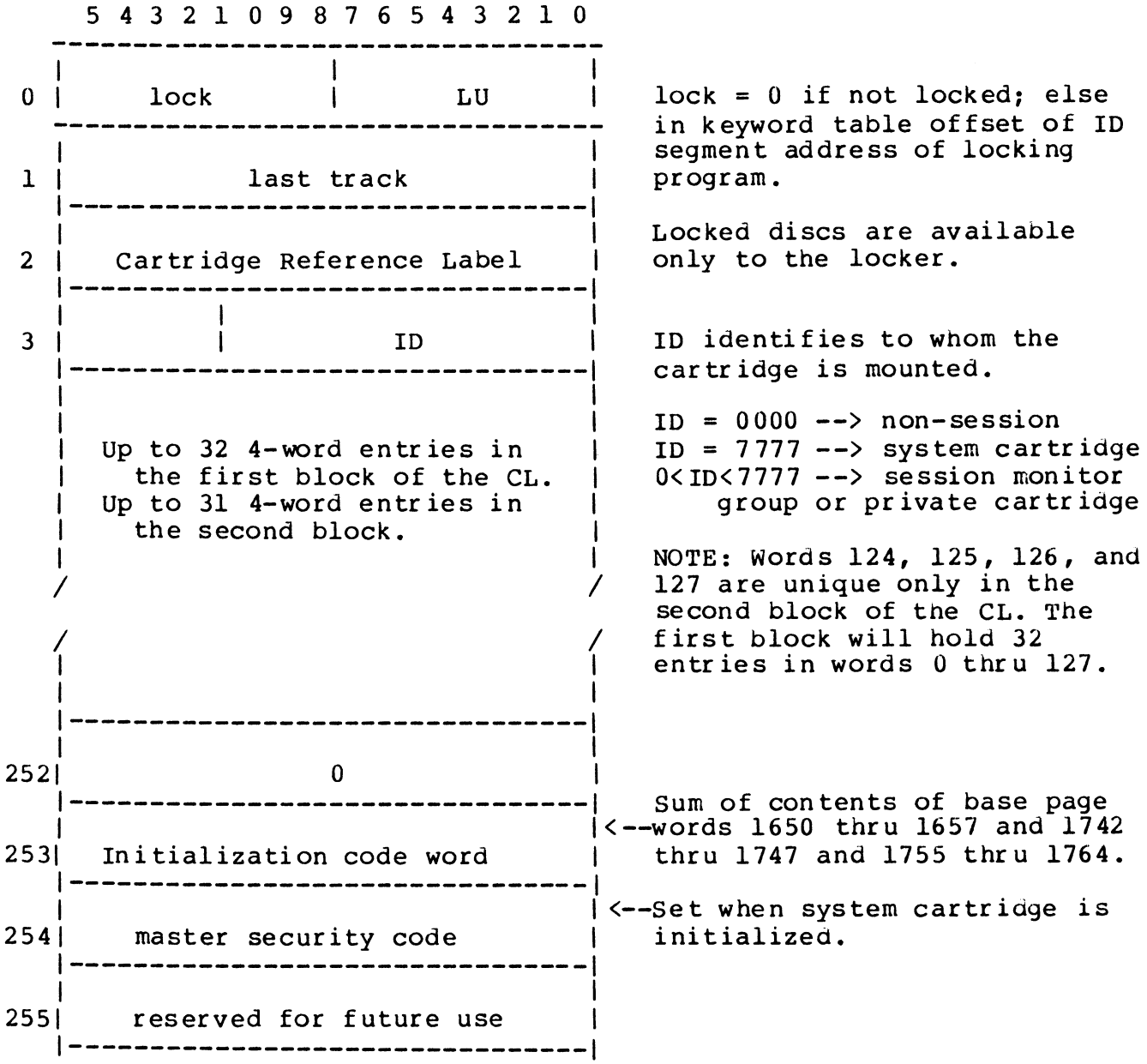


Legend for Data Control Block

Word		Content
4	End-of-File Code, type 0 file:	01 lu = EOF on Magnetic Tape 10 lu = EOF on Paper Tape 11 lu = EOF on Line Printer
5	Spacing Code, type 0 file:	bit 15 = 1 - backspace legal bit 0 = 1 - forward space legal
6	Read/Write Code, type 0 file:	bit 15 = 1 - input legal bit 0 = 1 - output legal
7	Security Code Check/Open Mode/Buffer Size/In Buffer/To Be Written/ EOF Read Flag, all file types	
	(SC) Security Code Check	bit 15 = 1 - security codes agree = 0 - security codes do not agree
	DCB Buffer:	bits 14-7 = Number of blocks in DCB buffer
	(EX) Extendible:	bit 5 = 1 file is not extendible = 0 file is extendible
	(SY) System Disc:	bit 4 = 1 file is on a system disc = 0 not on a system disc
	(OM) Open Mode:	bit 3 = 1 - update open 0 - standard open
	(IB) In Buffer Flag:	bit 2 = 1 - data in DCB buffer = 0 - data not in DCB buffer
	(EF) EOF Read Flag:	bit 0 = 1 - EOF has been read = 0 - EOF has not been read
	(WR) To Be Written:	bit 0 = 1 - data in DCB buffer to be written = 0 - data in DCB buffer not to be written
9	Open/Close Indicator: if open, contains ID segment location of program performing open. If closed, set to zero.	

Cartridge Directory Format

The cartridge directory is located in the system area on LU 2. Its length is two blocks.

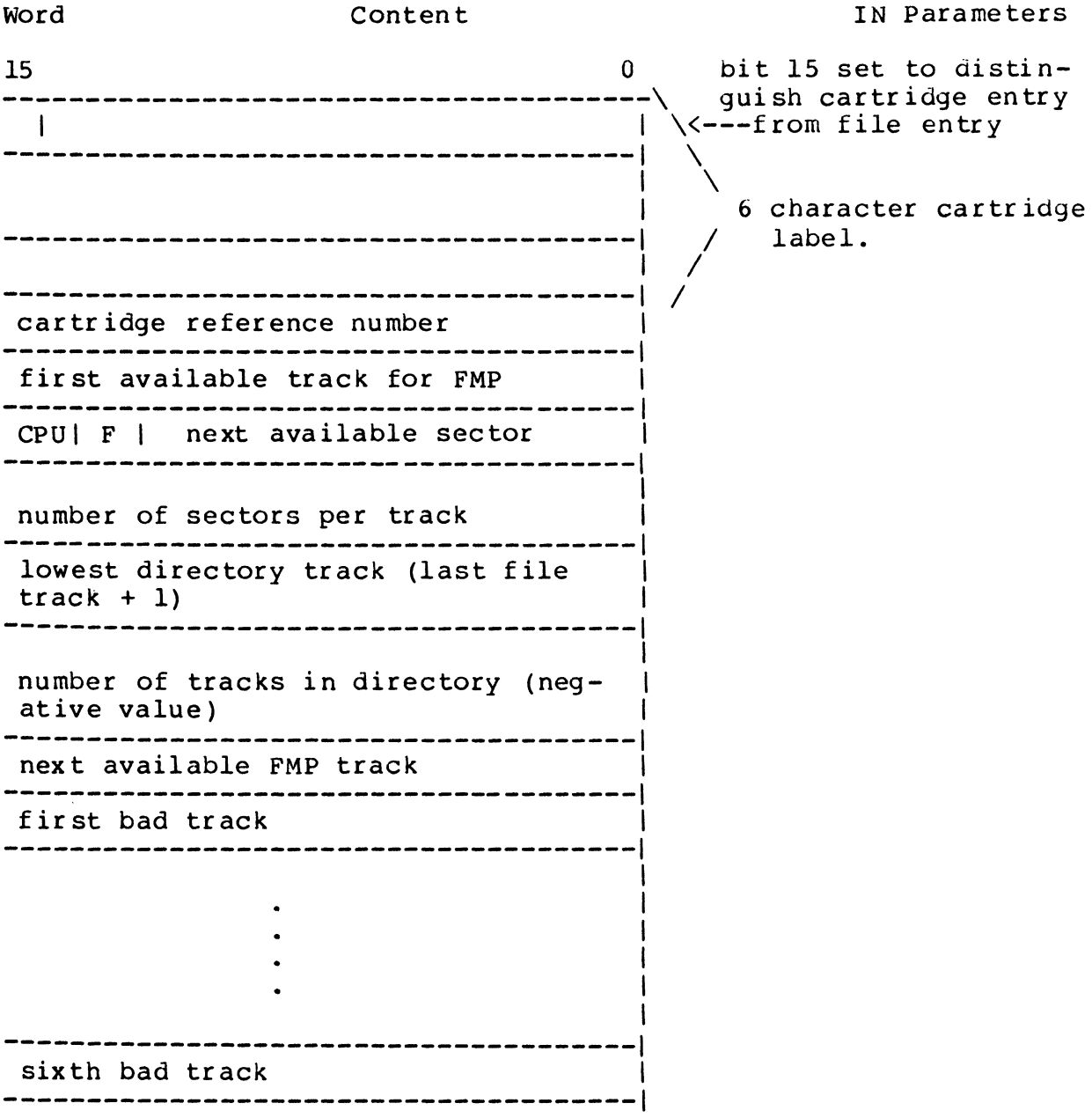


File Directory

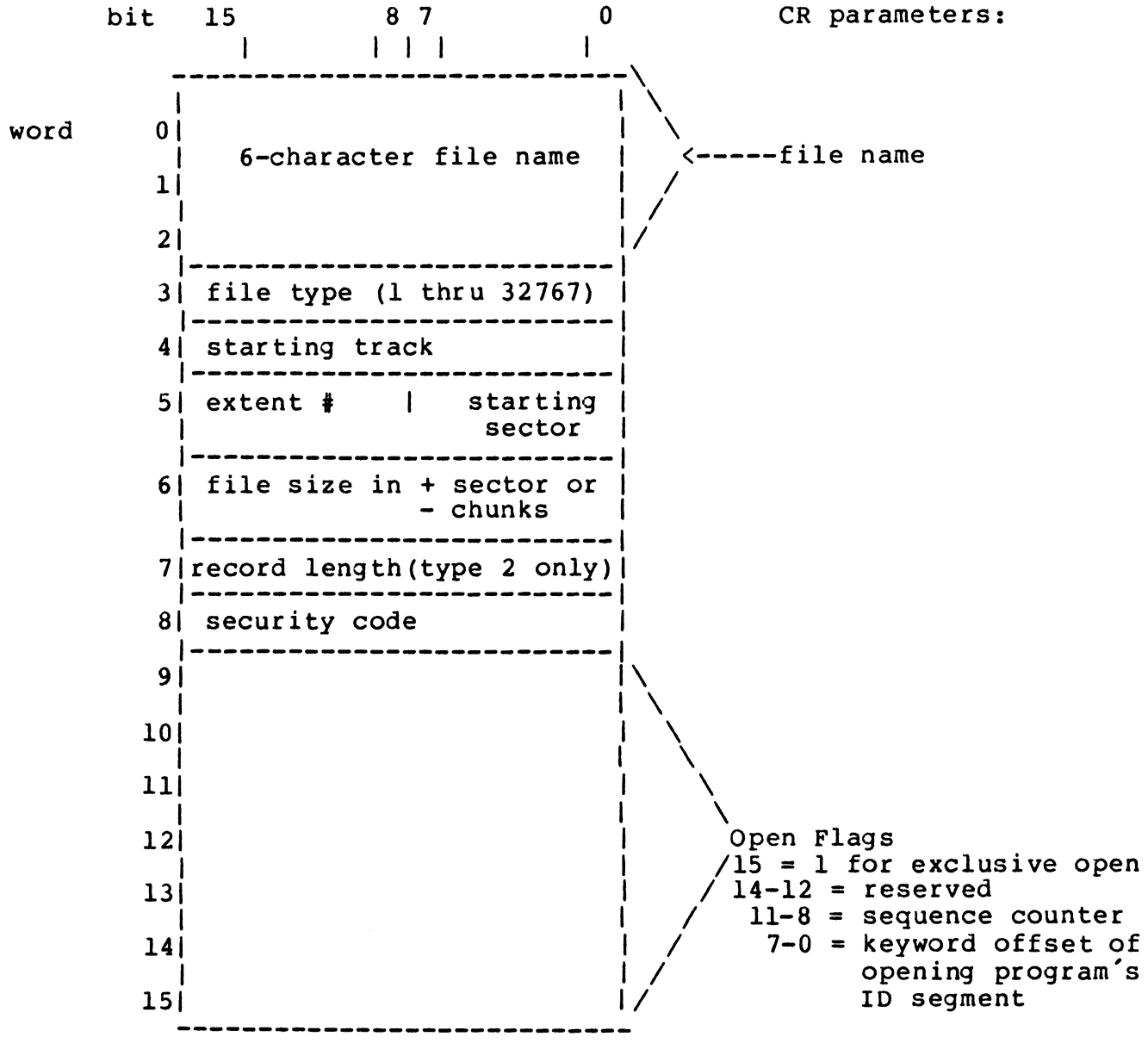
The first entry in each File Directory is the specification entry for the cartridge itself. The directory starts on the last FMP track of each cartridge in sector zero on all discs. The directory blocks are written using sector skipping. The directory sector address can be obtained from the block address by the following formula:

sector address = (block *14) modulo S/T

where S/T is the number of sectors per track. Directory blocks are 128 words long. Each Directory entry is 16 words long.



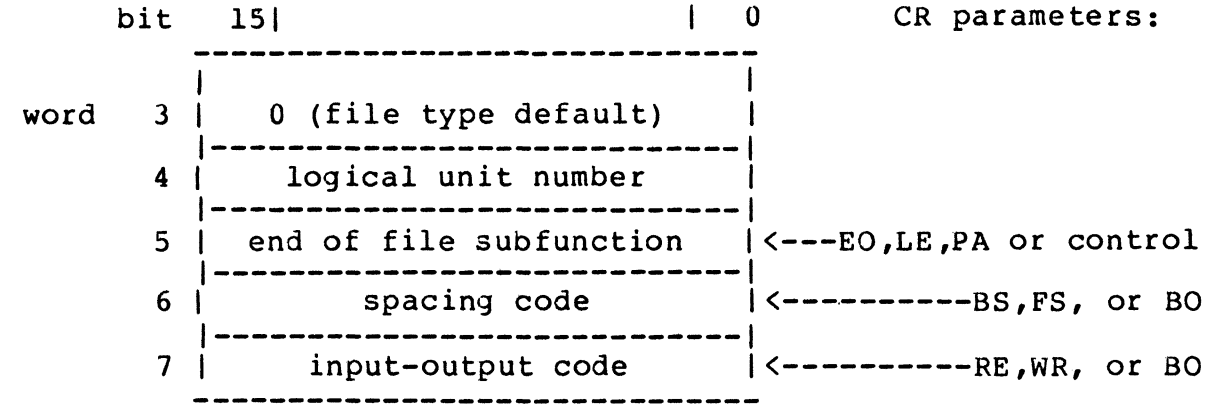
Disc File Directory



word 0 = 0 if the last entry in directory; = -1 if file is purged

Type 0 File Directory Entry

The entries for non-disc (type 0) files differ from those for disc files in words 3 through 7:



Words 5-7 are octal codes:

```
end-of-file subfunction = 01LU for MT(EO)
                          10LU for paper tape (LE)
                          11LU for line printer (PA)
                          or subfunction code

spacing code  =  bit 15 = 1 backspace legal (BS)
                 bit  0 = 1 forward space legal (FS)

input/output code  = bit 15 = 1 input legal (RE)
                     bit  0 = 1 output legal (WR)
```

Session Control Block (SCB)

A Session Control Block (SCB) is established for each user who has successfully "logged-on" to the system. The SCB contains the information necessary to identify the user to the system and describe his capabilities in terms of command processing and I/O addressing space.

The format of the SCB is shown in Figure C-6.

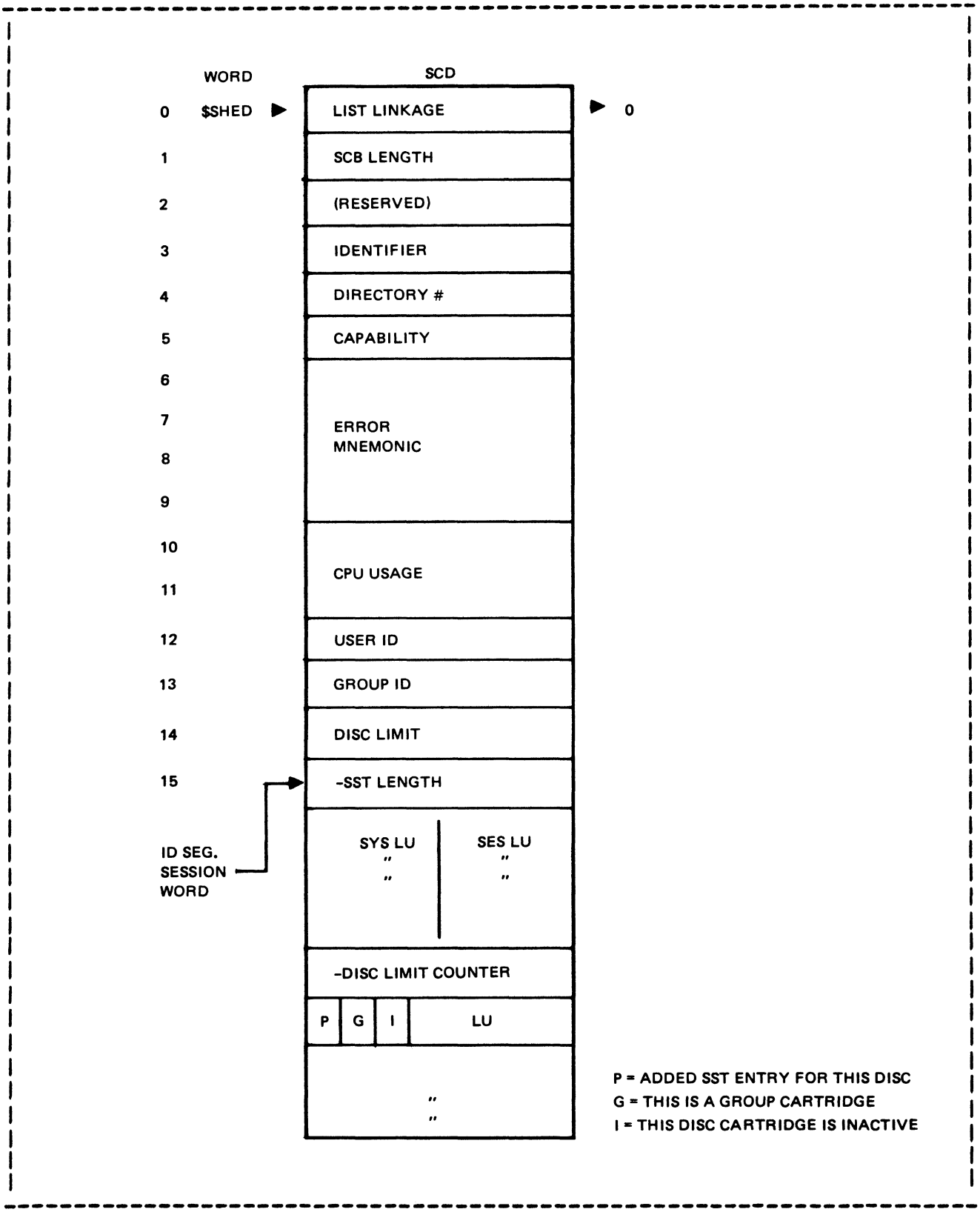


Figure C-6. Session Control Block (SCB)

Session Switch Table (SST) and Configuration Table

When operating in the session environment every I/O request is routed to the appropriate I/O device via the Session Switch Table (SST). Each SST entry describes a session LU, which the user addresses, and associated system LU where the I/O request will actually be directed. The SST describes the session user's I/O addressing capabilities by defining the system LUs the user has access to and the associated session LUs by which the user accesses them.

When the user makes an I/O request the SST is searched for the specified session LU. If the requested LU is found, it is switched to the associated system LU as specified in the SST entry and the I/O request is processed. If the requested LU is not found, an error is returned (IO12-LU not defined for this session).

The Session Switch Table is maintained in memory as part of the Session Control Block (SCB). The format of the SST is shown in Figure C-7.

System LUs can be integer numbers between 1 and 255. Session LUs can be integer numbers between 1 and 63. Session LUs are assigned:

* at log-on, via user and group account file entries, or
* on-line using SL command (see Chapter 3), or
* at log-on, via Configuration Table entries.

The Configuration Table describes the default logical units to be used for specific device logical units. Each station (terminal) logical unit defined in the Configuration Table has associated with it a set of device logical units which are assigned default logical units to be used when a user logs on at this station (terminal). The default logical unit associated with the station itself is always 1.

At log-on, these default values are written from the Configuration Table in the account file into the user's Session Control Block (SCB), unless overridden by entries in this particular user's SST. The format of the Configuration Table is shown in Figure C-8.

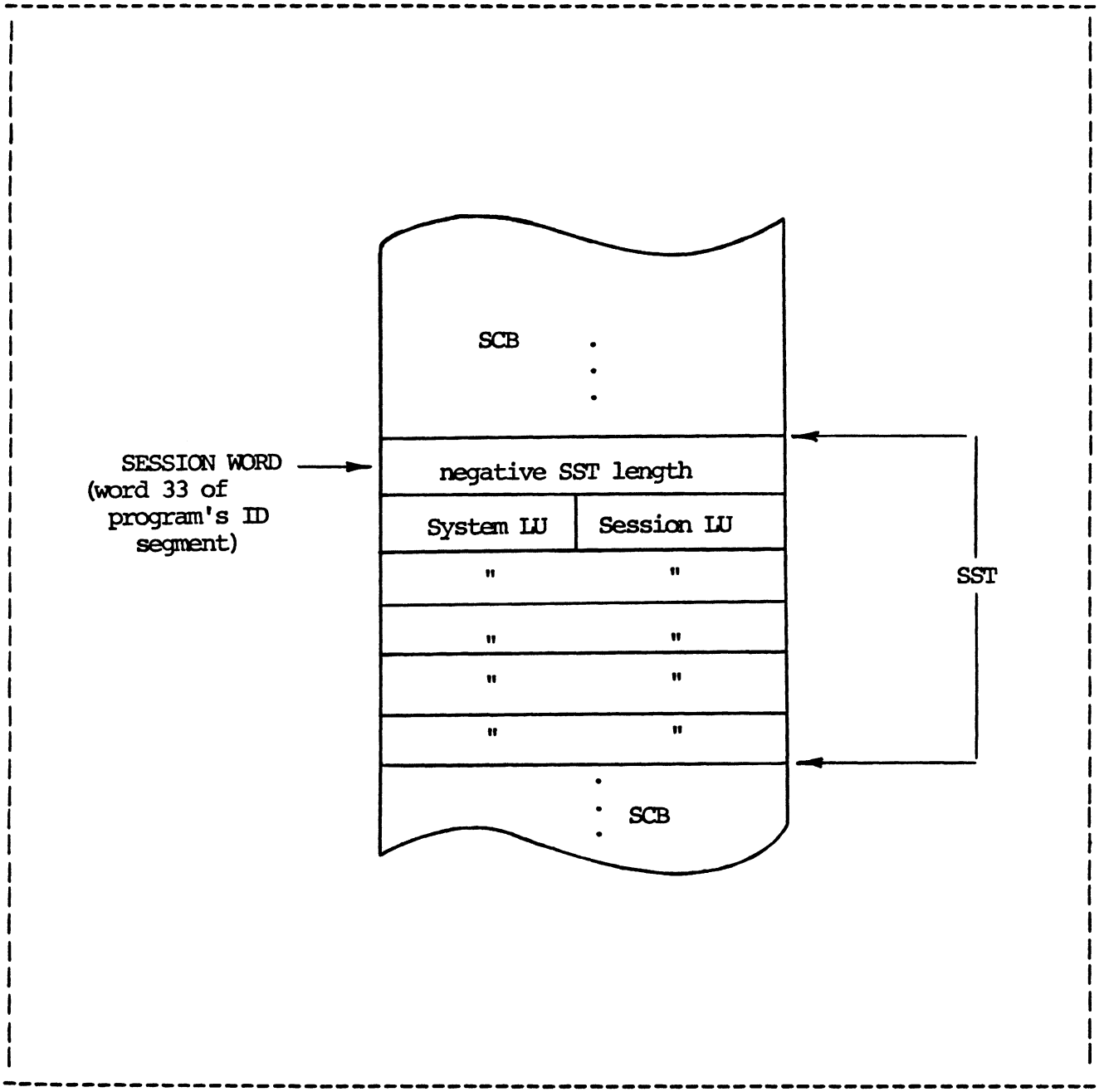


Figure C-7. Session Switch Table (SST) Format

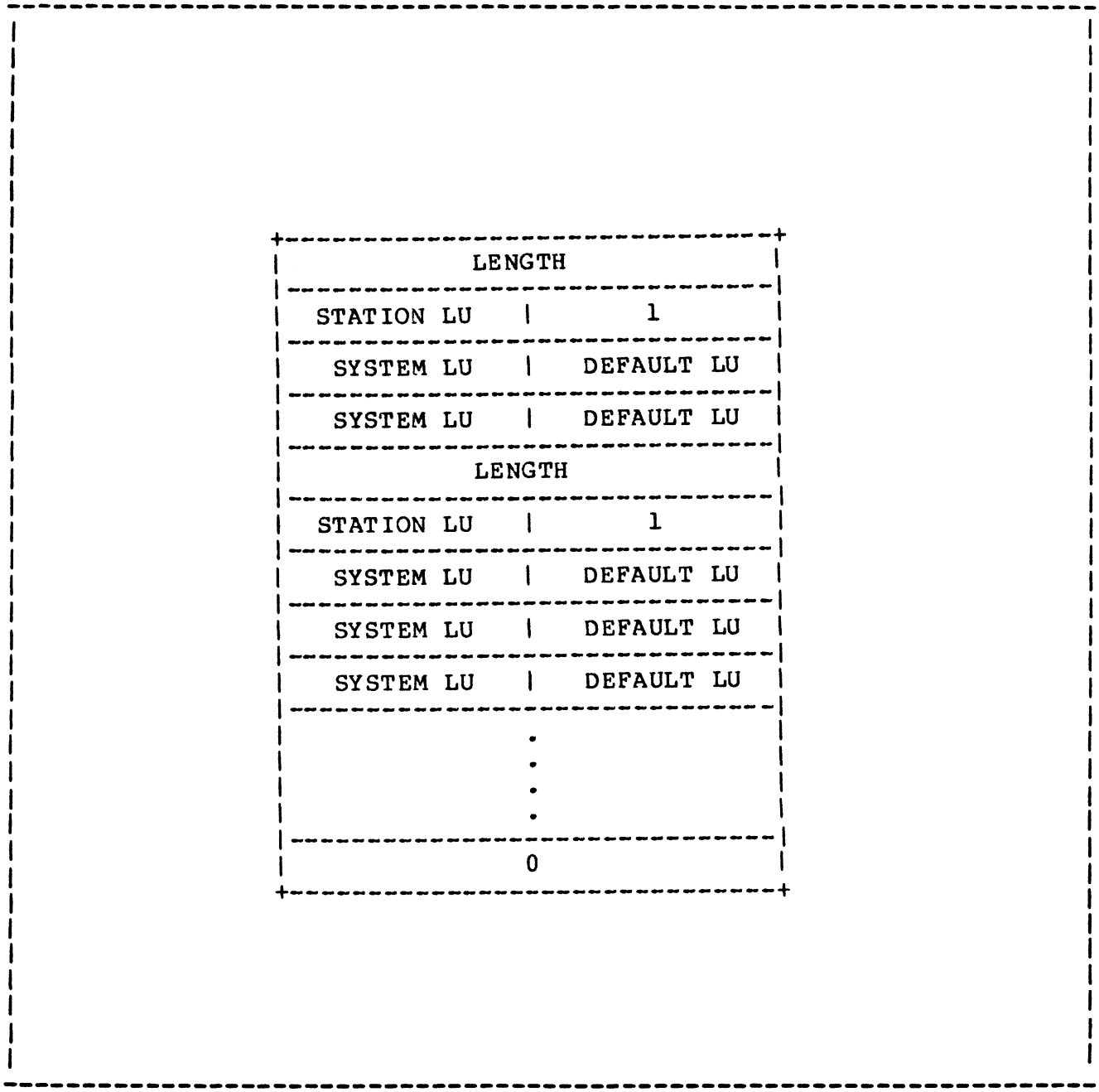


Figure C-8. Configuration Table

Appendix D
Logical Source and Load-and-Go Areas

The Logical Source (LS) and Load-and-Go (LG) areas are no longer required by HP supported subsystems. Earlier RTE Compiler and Loader versions used these areas for temporary program storage. Source programs could be stored on the LS tracks where they could be edited if necessary then compiled or assembled. The resulting binary relocatable code could then be moved to the LG area before invoking the loader.

The commands described in this Appendix are not recognized by the Session Monitor. They can only be input from a non-session environment.

The Logical Source (LS) areas can be allocated on the system (LU 2) or auxiliary (LU 3) disc. Each LS disc area is allocated in units of whole tracks. The system maintains a pointer to the last source program moved to an LS area. The user may reset or clear this pointer with the File Manager or System LS command. A program can be moved to an LS area with the File Manager MS command, and a program can be saved from the LS area as a type 4 file (source program) using the File Manager SA command.

The Load-and-Go (LG) area was used by earlier RTE compiler and assembler versions for storing the binary relocatable code of programs and merging files. The number of tracks in this area must have been specifically declared with the LG command before running an assembler or compiler. The relocatable binary program resulting from assembly or compilation may be placed in the LG area. The earlier Loader versions expected a binary relocatable program to be in the LG tracks by an LG command. Any program segments must have also been loaded into the LG area with the main program. If a main program or segment used subroutines not in a resident library, these also must have been loaded into the LG area.

The LG area must have had enough tracks to hold the main program plus any segments or subroutines. The LG area was cleared automatically when the Loader program terminated normally or at the end of a job by the FMGR EOJ command. The LG area could also be cleared by the LG command. A file in the LG area could be saved with the File Manager SA command as a type 5 file. A type 5 file could be moved to the LG area with the File Manager MR command.

Following are the LS and LG area related commands. By inspecting the prompt character, the user can determine whether the command is a File Manager, System, or Editor command.

MS (Move Source File)

The MS command moves a program on a FMGR file to an LS (logical source) area where it can be edited, compiled or assembled.

:MS,namr[,program[,IH]]

Parameters

namr	File name or logical unit number of program to be transferred; (see NAMR PARAMETER, Chapter 3).
program	Name of program to which LS tracks are assigned; if omitted, tracks are assigned to EDITR.
IH	Inhibit the pointer to the LS track from being set, which requires use of the LS command to set pointer, if omitted, pointer is set to the LS track containing namr.

NOTE

A program file to be moved with MS must not have records longer than 128 words.

When MS is entered, the logical unit number and first track of the LS area are reported on the log device as:

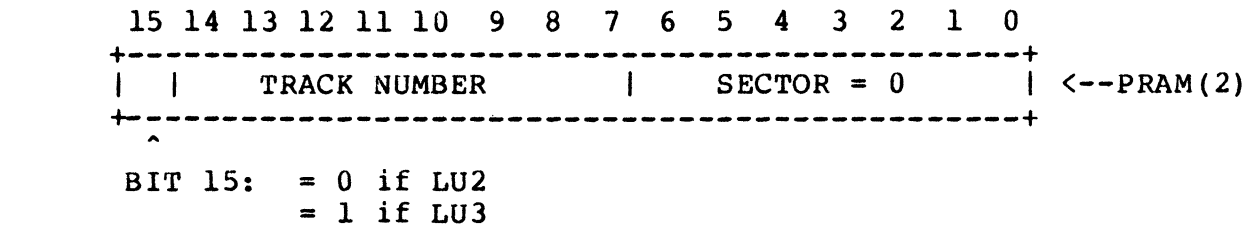
```
FMGR 015
LS LU n TRACK nnn
```

If you use the LS command to set or reset the pointer to this LS track, you will need to know these numbers.

Ordinarily, the LS tracks are assigned to the program EDITR when MS is executed. If, however, you have a reason to assign these tracks to another program, it may be specified as program. Care should be taken that the program specified does not release the LS tracks. FMGR, for instance, should never be specified as program since all tracks assigned to FMGR are released when the MS command executes and also when FMGR is terminated.

If FMGR was scheduled from a program using the RTE Program Schedule EXEC call with wait, the scheduling program can retrieve the logical unit and track number in the second parameter returned by a call to RMPAR:

```
DIMENSION PRAM (5)
CALL RMPAR (PRAM)
```



Logical source tracks are RTE system tracks so the logical unit number can be only 2 or 3. RMPAR returns the absolute track number (from system track 0, not FMP track 0).

The track number and logical unit of the LS area are important in case the LS pointer is moved and no longer points to the particular area containing your program. This may occur if another user resets the pointer. In this case, use the LS command (SET LOGICAL SOURCE POINTER) to reset the pointer to the area you want.

LS tracks assigned to the EDITR are not released automatically; they may be released with the RT command (see Chapter 3 for a description of the RT command).

Examples

1. Create a source file from the terminal and then move it to a logical source area for compilation.

```
:ST,1,SPFILE
enter program statements
CNTL/D<----------------------end file with CNTL/D

:MS,SPFILE
FMGR 015
 LS LU 2 TRACK 010<---------SPFILE is now on system track 10
```

2. Assume a program on file SOURC1 is to be moved to an LS area to be edited by EDITR:

```
:MS,SOURC1
FMGR 015
 LS LU 2 TRACK 011<---------unless a file is specified, EDITR edits
                             source file in track 11
```

3. If IH is specified, the editor, compilers or assembler will not be able to access the particular logical source until :LS is used to set the pointer:

```
:MS,SOURC2,,IH
FMGR 015
 LS LU 2 TRACK 012<---------use this LU and track number in LS command
```

LS (Set Logical Source Pointer)

A pointer to a particular logical source track can be set with the LS command; this command will also clear the current pointer. It is identical to the RTE *LS command.

```
+-----------------------------------------------------------------+
|                                                                 |
| :LS[,lu,track]                                                  |
|                                                                 |
| Parameters                                                      |
|                                                                 |
| lu          Logical unit number of LS area; must be 2 or 3 to set |
|             pointer, 0 to clear a current pointer.              |
|                                                                 |
| track       Track number to which pointer is set.               |
|                                                                 |
| If both track and lu are omitted, the current pointer is cleared; |
|                                                                 |
+-----------------------------------------------------------------+
```

Unless the pointer is set to a particular LS area, the source program in that area cannot be referenced. Since the pointer is set whenever MS is executed, LS need be used only if the pointer has been reset to a different LS area. The LS pointer is also set by the EDITR when it is terminated with an /EL name command.

Example

```
:MS,AFILE,,IH<-------------no pointer set
FMGR 015
 LS LU 2 TRACK 013
:MS,BFILE
FMGR 015
 LS LU 2 TRACK 014<--------pointer set to track 14, file BFILE
  .
  .
  .
:LS,2,13<------------------pointer set to track 13, AFILE
:RU,EDITR
/{}<-----------------------space, file in LS area can be edited
  .
  .
  .
:LS<-----------------------pointer cleared (same as :LS,0)
```

LG (Assign LG Tracks)

A set of tracks is allocated to the LG area with the LG command; this area must contain binary relocatable programs that are to be loaded with the LOADR.

```
+-----------------------------------------------------------------+
|                                                                 |
| :LG[,#tracks]                                                   |
|                                                                 |
| Parameters                                                      |
|                                                                 |
| # tracks     Positive integer indicating the number of tracks to |
|              reserve on the system or auxiliary disc for the LG |
|              area; if omitted or zero the current tracks are    |
|              returned to the system.                            |
|                                                                 |
+-----------------------------------------------------------------+
```

Whenever LG is specified, any binary relocatable program in the current LG area is cleared. The LG area is also cleared upon successful completion of LOADR and at the beginning or end of a batch job by the JO and EO commands.

If # tracks is not specified, the LG tracks are returned to the system and no tracks are available for a compiled or assembled program. If # tracks is specified, then the current LG area is cleared and a new area is allocated for the LG tracks.

The FMGR LG command is the same as the RTE LG command and the same error messages may be generated.

Example

```
:LG,3<-----------------allocate 3 tracks to LG area
:RU,ASMB,2,99<---------assemble program source in LS area;
                        relocatable binary to LG
:SA,LG,BFILE<----------save binary relocatable program as file
:LG<-------------------return LG tracks to system
```

MR (Move Relocatable Program)

A program saved in relocatable binary form (type 5 file) can be moved to the LG area with the :MR command.

```
+-------------------------------------------------------------------+
|                                                                   |
| :MR,namr                                                          |
|                                                                   |
| Parameters                                                        |
|                                                                   |
| namr        File name or logical unit number of file to be        |
|             transferred; may include security code and cartridge  |
|             reference number (refer to NAMR PARAMETER,           |
|             Chapter 3).                                           |
+-------------------------------------------------------------------+
```

If a file namr is not terminated by an END record, FMGR prints the message FMGR 006 when the end of file is reached. FMGR suspends and you should then load the next relocatable module and enter *GO,FMGR. This situation may occur when more than one relocatable binary module is entered through the paper tape reader. If it occurs within a batch job, FMGR is terminated and you must re-schedule it with the RU,FMGR command.

If no LG tracks are allocated when this command is given, the file size is checked and an LG allocation large enough to hold the file is made automatically.

Unlike the MS command, repeated MR commands add each specified file to the same LG area until it is cleared, whereas MS creates a new LS area for each specified file.

Examples

1. A relocatable binary program requiring 2 tracks has been punched on paper tape; to move it to the LG area:

```
:LG,2<-----------------clear and assign two LG tracks
:MR,5<-----------------read program from paper tape to LG tracks
```

2. Move type 5 file XX to LG area with 1 track:

```
:LG,1
:MR,XX
```

3. Merge a program into the LG area:

```
:LG,4                    a program is divided into 3 parts, each
:MR,PROGA                stored as type 5 file, are moved to an LG
:MR,SEGA1                area one following the other.
:MR,SEGA2
:SA,LG,%MAIN             save the merged program into %MAIN.
```

SA (Save Program as File)

To save the contents of a logical source (LS) area or the load-and-go (LG) area as a file, use the SA command. A file is created by SA.

```
      ,LS
:SAVE   ,namr
   --,LG
```

Parameters

LS	Source program in LS area is saved.
LG	Relocatable program in LG area is saved.
namr	File name or logical unit number of file in which LS or LG is saved; file is created and full namr can be used; (see NAMR PARAMETER, Chapter 3).

NOTE

The SA command truncates records longer than 128 words; no HP compiler or assembler generates records longer than 128 words.

If there is not enough cartridge space available when namr is a disc file, error message FMGR -033 is issued. Any portion of the file that was saved is purged.

Save LS

For SA, LS namr the file type when omitted defaults to type 4.

The file size defaults to one half the number of blocks on a system disc track. After the final end-of-file is written on the new file, its position is checked. If there are no extents, any remaining unused disc space is returned to the system. If extents were created, the file size is not shortened.

Save LG

For SA,LG,namr, the file type of namr when omitted defaults to type 5. A checksum is provided.

Following each relocatable module (main program, program segment or library subroutine) including the last, a zero-length record is written if namr is a disc file or an EOF if namr is a non-disc device.

The file size, if omitted, is computed as the maximum size possible from the amount of LG area used. Extents are not created and file will be only as long as needed.

Examples

```
1.  :LS,2,36<-----------------set LS pointer to system disc, track 36
    :SA,LS,SFILE::13<---------save LS area as type 4 file, on cartridge
                                13; default file size
2.  :SA,LG,4<-----------------punch relocatable program in LG area to
                                paper tape
3.  :SA,LG,LFILE:JM:-14<------save relocatable program as type 5 file
                                on logical unit 14 protected by security
                                code JM
```

LG (LG Tracks)

Allocates or releases a group of disc tracks for the LG area. LG tracks may be used as temporary storage for relocatable code in FMGR operations.

*LG,numb
where:
numb=0 (zero) releases the allocated LG area.
numb>0 release currently allocated LG tracks and then allocate numb contiguous tracks for an LG area.

Enough LG tracks for storing relocatable code must be allocated before storing into this area. Insufficient tracks cause the program to abort and one of the following diagnostics to be displayed on the system console:

IO06 - LG area not defined.
IO09 - LG area overflow.

An LG request should not be used while anyone is using the LG tracks. Doing so may result in the message

LGO IN USE

being displayed on the system console, and no change in the current number of LG tracks. In most cases, however, the attempt to do so results in an IO06 error being issued.

LS (Source File)

Designates the disc Logical Unit number and starting track number of source code stored in the track pool prior to an EDITR operation on the code.

```
*LS,disc lu,trk numb

where:

disc lu      is the Logical Unit number of the disc containing
             the source file.
             2 or 3 = system or auxiliary disc units.
                  0 = eliminate the current source file
                      designation.

trk numb     is the starting track number (decimal) of the source
             code.  Defaults to 0 if not specified.
```

LS replaces any previous declarations with the current source code area. Only one area may be declared at a time.

ELR (End Edit, Replace File and Name, and Store in LS Tracks)

This command ends the edit, replaces an existing File Manager file under the original name or a new name if supplied, and stores the same data in the LS tracks.

```
/ELR
 or
/ELRname[:security code[:cartridge reference number]]

 (see NAMR PARAMETER, Chapter 3, for syntax definitions)
```

Examples:

```
/L10
  FTN,L
  C
  C     MAIN PROGRAM
  C
        PROGRAM PRIME
        DIMENSION I(5)
        WRITE(6,100)
100     FORMAT(12X,"PRIMES FROM ONE TO FIVE HUNDRED TEN"////
       1,19X,"1",19X,"2",19X"3")
        J=4
        M=1
/ELRPRIME
  LS FILE 2 30
END OF EDIT

/7
  SEC   BSS 1
/L5
  SEC   BSS 1
  WEN   NOP
        JSB .ENTR      /In this example the old File Manager file is
        DEF HRS       / replaced by a new file with the same name. A
        JSB EXEC     /  file is not created with this command.  The
        DEF *+3     /   old file contents are written over with the
/ELR<--------------     contents of the edited destination work area.
  LS FILE 2 30
END OF EDIT             If a security code was used with the original
                        file name, this form of the EL terminator will
                        cause an FMGR-7 error when EDITR attempts to
                        write new contents into the old file.  The ELR
                        name form must be used with the security code.
*LG,1
```

ELC (End Edit, Create File Manager File, Store in LS Tracks)

This command ends the edit, creates a File Manager file and stores the edited text both in the newly created file and in the system LS tracks.

```
/ELCname  [:security code[:cartridge reference]]

(See NAMR PARAMETER, Chapter 3, for syntax definitions).
```

Example:

```
SOURCE FILE?
/
  ASMB,R,L,T
/L9
  ASMB,R,L,T
        NAM TIME
        ENT WEN
        EXT .ENTR,EXEC
  HRS   BSS 1
  MIN   BSS 1
  SEC   BSS 1       /The security code and cartridge reference are
  WEN   NOP        / supplied in this example.  If they are omitted,
        END       /  they would default to 0.
EOF              /
/ELCCAROL2:LS:2/   /The logical source location is returned so that
  LS FILE 2 36<---/ the pointer can be set for an assembly or
END OF EDIT         compilation.
```

EL (Assign Edited File to LS Tracks)

This command ends the editing session and assigns the contents of the edited destination area to the RTE logical source (LS) tracks. The location of the LS tracks is returned to the user, and the logical source pointer is set by EL so that the system LS command is not needed.

```
+-----------------------------------------------------------------+
|                                                                 |
| /EL                                                             |
|                                                                 |
+-----------------------------------------------------------------+
```

Example:

```
    :RU,EDITR
    SOURCE FILE?
    /{}<----------------------EDITR creates the file in the Logical
                               Source (LS) tracks.
    EOF
    / ASMB,R,L,T
    /       NAM TIME
    /       ENT WEN
    /       EXT .ENTR,EXEC
    /EL
      LS FILE 2 30<-----------The edit is terminated and the file
                               stored in the LS tracks with the EL
                               command.
    END OF EDIT



    :RU,EDITR<----------------Run EDITR again
    SOURCE FILE?
    /{}<----------------------Blank-EDITR copies LS tracks into
                               its source work area
      ASMB,R,L,T
    /L10<---------------------List 10 lines
      ASMB,R,L,T
            NAM TIME           These are lines into LS tracks by EL
                               command above.
            ENT WEN
            EXT .ENTR,EXEC
    EOF
```

This command should be used with some caution if the file came from the LS area, because once the destination file is moved to the logical source area it replaces the original text in that area and there is no longer a back up copy of the source file. This source file can be saved with the ELC or ELR EDITR commands described previously in this Appendix.

Appendix E
Running Program FMGR

To request FMGR from the system console or an auxiliary terminal, the system RU command may be used (see Chapter 4 for a description of the RU command). FMGR may also be scheduled programmatically through an EXEC call (see the RTE-IVB Programmer's Reference Manual for a description of EXEC calls).

```
*RU,FMGR [,input [,list [,severity [,log ]]]]   <--commands entered
                                                    from device

*RU,FMGR,namr [,list [,severity [,log ]]]       <--commands entered
                                                    from file
```

input
Logical unit number of input device for FMGR commands; if omitted, LU 1 is assumed; in a multi-terminal environment, default input is logical unit number of device where FMGR is scheduled.

log
Logical unit number of the log device used to log and to correct any diagnosed errors; must be an interactive device; if omitted, input device is assumed unless it is non-interactive, in which case, LU 1 is assumed.

namr
File name of file containing command input to FMGR; refer to file name discussion below for restrictions.

list
Logical unit number of device used to list results of FMGR commands; if omitted, LU 1 is assumed.

RUNNING PROGRAM FMGR . . . cont'd

severity
Severity code defines action in case of error messages;

0 Display error codes and echo commands on log device (default).

1 Display error codes on log device, inhibit command echo.

2 Error code displayed only if error requires transfer of control to log device for correction, in this case, active job terminates; no command echo.

3 Same as 2 except active job not terminated when an error causes transfer to log device. You can transfer back to the job. No command echo.

4 If a FMGR command error is encountered, job continues automatically; no command echo, no transfer to log device, and no job abort occurs.

*** NOTE ***

Once FMGR is running, "log" can be changed with the File Manager LO command, "list" with the LL command, and "severity" with the SV command (see Chapter 3 for a description of these commands).

EXAMPLES:

1. *RU,FMGR <---FMGR is scheduled with default values for all parameters: input from LU 1, errors logged on LU 1, list output to LU 1.

2. *RU,FMGR,5,,,1 <---Input from right cartridge tape unit (LU 5), log errors on LU 1, list to LU 1, severity code inhibits command echo on LU 1.

3. *RU,FMGR,FILEA1,,1 <---Input from FILEA1, severity inhibits command echo on console, default list is LU 1.

COMMENTS:

The device through which commands are sent to the FMGR program can be any device to which a logical unit has been assigned and that can be used for input. Normally, when input is from disc, a file name is used to specify the input device.

If a file name is specified and begins with NO, these characters are assumed by RTE to be the scheduling parameter NOW. Specifying a second parameter as any two ASCII characters, directs FMGR to derive the file name from the 1st parameter. The second parameter is ignored by FMGR in this case. For example:

```
*RU,FMGR,NOBLE,XX [,list [,severity [,log ]]]
```

Note that XX can be any two ASCII characters. If the file name starts with NO, then XX must be specified.

NOTE: For information on the method of scheduling FMGR programmatically via an EXEC 23 or EXEC 24, refer to the RTE-IVB Programmer's Reference Manual (92068-90004).

Programmatic Request for FMGR

To request FMGR from a program (optionally, a command string may be passed to FMGR), use the following call:

```
CALL EXEC(ICODE,6HFMGR  ,INP,LIST,ISV,LOG,IDUM,IBUFR,IBUFL)
                         ---  ----  ---  ---  ----  -----  -----
CALL EXEC(ICODE,6HFMGR  ,2HXX,LIST,ISV,LOG,IDUM,IBUFR,IBUFL)
                         ----  ----  ---  ---  ----  -----  -----
```

ICODE = 23 to schedule FMGR with wait
= 24 to schedule FMGR without wait

INP = input; if 2 ASCII characters, input is entered from the file whose name appears in IBUFR; otherwise INP is the logical unit number of the input device where the FMGR commands will be entered. If omitted, LU1 is assumed; in a multi-terminal environment, default input is logical unit number of device where FMGR was scheduled. IBUFR must contain a FMGR command which is executed before control is passed to that LU (e.g., :TR,TEST where TEST holds a number of commands.

LIST,ISV,LOG=corresponds to the parameters in the RU,FMGR command.

IDUM = dummy placeholder for parameter 5.

IBUFR,IBUFL = optional buffer containing command string to be passed to FMGR or the name of a file in which FMGR commands are stored (file HELLOO in example shown below).

example:
```
DATA IBUFR/2H,,,2HHE,2HLL,2HOO/
DATA LIST/1/ISV/0/,LOG/1/,IBUFL/4
CALL EXEC (23,6HFMGR  ,2HXX,LIST,ISV,LOG,IDUM,IBUFR,IBUFL)
```

The command string to be passed to FMGR via IBUFR must begin with a colon (:). The buffer length specified in IBUFL is a positive value for words, or a negative value for characters. The command string is ignored if:

a. The input device specified is not an interactive device.

b. The command string does not start with a colon.

Glossary

ABSOLUTE PROGRAM - A program that has been relocated and is capable of being loaded into main memory for subsequent execution. An "absolute program" is synonymous with "relocated program."

ABSOLUTE SYSTEM - The binary memory image of an RTE system (stored on Logical Unit 2).

ACCOUNT FILE - A disc resident file created and maintained by the System Manager. It contains information on all authorized session users and other session related information.

ADDRESS SPACE - See LOGICAL MEMORY or PHYSICAL MEMORY.

ASYNCHRONOUS DEVICE - A device that can perform I/O operations that are independent of time considerations but operates simultaneously with program execution. Interaction with the computer is through request/response circuitry.

AUXILIARY DISC SUBCHANNEL - An optional subchannel that is treated as a logical extension of the system disc subchannel, Logical Unit 2. If used, it is assigned to Logical Unit 3. The binary memory image of RTE-IVB may not reside on the auxiliary subchannel.

BACKGROUND (BG) - An arbitrary name for one of two types of partitions in RTE; generally used for lower priority programs whose responses to interrupts are not time-critical.

BASE PAGE - A 1024-word area of memory corresponding to logical page 0. It contains the system's communication area, driver links, trap cells for interrupt processing, and system and/or user program links.

BASE PAGE FENCE - A hardware register that divides a logical base page into a portion containing the user's base page and a portion of the system's base page.

BG - See BACKGROUND.

BIT BUCKET - Logical unit number pointing to Equipment Table entry number zero, which in turn, does not point to any existing device. I/O directed to the bit bucket is lost.

BLOCK - Two logical disc sectors of 128 bytes each, totaling a 256 bytes.

BOOT EXTENSION - An absolute program that resides on the first two sectors of logical track 0 of the system subchannel. The Boot Extension itself is first loaded into memory by the Bootstrap Loader or ROM Loader.

BOOT FILE - An optional file to which the Bootstrap Loader produced by the On-Line Generator is stored. This may be a disc file or a logical unit (e.g., a mini-cartridge).

BOOTSTRAP LOADER - A loader produced by the Generator and stored in the boot file. The Bootstrap Loader loads the Boot Extension into memory and then transfers control to the Boot Extension.

BOOT-UP - The process of bringing the Bootstrap Loader or ROM Loader contents into memory. Control is then transferred to the Boot Extension to begin the initializatrion process.

BUFFER - An area of memory (main-memory, mass memory or local peripheral memory) used to temporarily store data.

CAPABILITY LEVEL - An integer from 1 to 63 which defines the FMGR, System, and Break-Mode commands which a session user may execute.

CARTRIDGE - A set of contiguous cylinders on a disc unit. Cartridges contain disc files with a directory of the files stored on each cartridge. All files on the same FMP cartridge must have unique names. The system disc on logical unit 2 contains the RTE operating system, and may contain FMP files.

CHAINING - A technique for coordinating sequential execution of independent programs in the same portion of main memory.

CLASS I/O - A means of buffering data between devices and user programs, and between programs themselves, that permits a user program to continue execution concurrently with its own I/O. The term "I/O without program wait" is a more commonly used term.

CLOSE FILE - A method of terminating a program's access to a file so that no further read/write operations may be performed on the file.

COMMON - An area of memory that can be accessed by a program and its subprograms. Usually used to pass data from a program to a subprogram. In RTE, system COMMON may be used to pass data from one program to another.

CONFIGURATION TABLE - A set of default logical units associated with the station that a session user logs on at.

CONFIGURATOR - A two-part program that allows reconfiguration of an RTE system's I/O and physical memory structures without going through a new system generation. The configurator is initiated as an option during the startup process.

CURRENT PAGE - The memory page in which the executing instruction is located. Some 21MX memory reference instructions can only directly reference locations in two pages: current page and base page.

CYLINDER - The area that passes under all heads during one revolution of the disc surfaces.

DATA CONTROL BLOCK (DCB) - A table within an executable program that contains information used by the File Management Package (FMP) in performing disc accesses. (See the RTE Batch Spool Monitor Reference Manual.)

DCPC - See DUAL CHANNEL PORT CONTROLLER.

DEVICE DOWN - Relates to the state of a peripheral device or I/O controller. When the device is down, it is no longer available for use by the system. The term also refers to the DN operator command.

DEVICE INDEPENDENCE - Refers to the ability of a program to perform I/O without knowing which physical device is being accessed (see also Logical Unit Number).

DEVICE REFERENCE TABLE (DRT) - A table created during system generation corresponding to Logical Units 1 through 63. The contents of the Device Reference Table include a pointer to the associated EQT entry, subchannel number of the device, and information as to whether or not the device is locked. The table may be modified by the user through an LU command.

DEVICE TIMEOUT - A time interval associated with a specific I/O device. If the system expects a response from such a device and this response does not occur within the timeout period, the device is assumed to be inoperative by the system. This feature is necessary to prevent a program from getting "hung up" because it is waiting for a response from a non-functioning peripheral device.

DIRECT MEMORY ACCESS - See DUAL CHANNEL PORT CONTROLLER.

DIRECTORY - A list of programs and files currently stored on a disc subchannel that can be displayed by the user.

DISC - Strictly speaking, the term means the platter(s) with the storage medium only; however the term is also loosely used to mean the entire peripheral including the drive.

DISC-BASED - Refers to an operating system using a disc storage device as an integral part of the operating system.

DISC FORMATTING - The process by which physical track and sector addresses are written in the preamble of each disc track sector. Disc formatting may be performed by the appropriate disc diagnostic. After formatting is completed, the SWTCH program and Disc Backup utility may perform subchannel initialization.

DISC-RESIDENT - A term applied to programs in executable form (absolute) that are stored on disc and brought into main memory for execution by the system in response to a program or operator request, time-of-day schedule or an I/O interrupt.

DISC ROM BOOT - A loader residing in Read-Only Memory that loads (off-line) the Boot Extension from disc storage and transfers control to the Boot Extension. (See also BOOT EXTENSION and STARTUP.)

DISPATCHER - An RTE system module that selects, from the scheduled list, the highest priority program to be executed next. The dispatcher module loads the program into memory from disc (if the program is not already in memory) and transfers control to the program.

DMA - See DUAL CHANNEL PORT CONTROLLER.

DMS - See DYNAMIC MAPPING SYSTEM.

DORMANT PROGRAM - A dormant program is one that is "sleeping" or inactive. More specifically, in RTE it is a program that is neither executing, suspended nor scheduled.

DOWN - Status of a device controller EQT that is not available for use.

DRIVER - A software module that interfaces a device and its controller to an operating system. Drivers specified by EQT definitions will go into either a driver partition or into the System Driver area of memory.

DRIVER PARTITION - A block of memory that contains one or more drivers. In RTE-IV, all drivers are in physical memory; however, only the driver partition containing the driver currently being used is included (mapped) in the logical address space.

DRT - See DEVICE REFERENCE TABLE.

DUAL CHANNEL PORT CONTROLLER (DCPC) - A hardware accessory that permits an I/O process to transfer data to or from memory directly, or access memory, thus providing a much faster transfer of data. The operating system controls access to the DCPC channels.

DYNAMIC BUFFER SPACE - Additional buffer space allocated to a program after the program code itself. The additional size is determined by the user. Typically used only by assembly language program.

DYNAMIC MAPPING SYSTEM - A hardware accessory allowing partitioned memory systems to address memory configurations larger than 32K words of physical memory.

EMA - See EXTENDED MEMORY AREA.

EQT - See EQUIPMENT TABLE.

EQT EXTENSION - A method for increasing the size of an Equipment Table entry's buffer space, during system generation, that gives the specified I/O driver more words of storage space than are available in the EQT temporary storage area.

EQUIPMENT TABLE (EQT) - A table in memory associating each physical I/O device controller with a particular software processing routine (driver). For a given device, the EQT provides status information, temporary storage and parameter passing services (see also Device Reference Table and Interrupt Table).

EXEC - One of the RTE system modules that interfaces user programs to the operating system.

EXTENDABLE FILE - An FMP file that is automatically extended in response to a write request to points beyond the range of the currently defined file. An extent is created with the same name and size as the main, and the access is continued.

EXTENDED MEMORY AREA (EMA) - An area of physical memory that may extend beyond the user's logical address space and is used for large data arrays. Its size is limited only by the amount of physical memory available. An entire array is resident in physical memory although the entire array is not currently in the logical address space.

EXTERNAL REFERENCE - A reference to a declared symbolic name not defined in the software module in which the reference occurs. An external reference is satisfied by another module that defines the reference name by an entry point definition.

FILE - A defined section of memory on a storage device used to store data or programs.

FILE EXTENTS - See EXTENDABLE FILE.

FILE MANAGEMENT - The operating system functions associated with maintaining disc files (translating file names to physical disc memory areas; maintaining a directory; checking for security codes; etc.).

FILE MANAGEMENT PACKAGE (FMP) - A collection of subprograms used to access, control and maintain files.

FILE MANAGER (FMGR) - A program that provides FMP file creation, access and manipulation services through FMGR commands entered by the user.

FMGR - See FILE MANAGER.

FMP - See FILE MANAGEMENT PACKAGE.

FOREGROUND - A purely arbitrary name for one of the two types of partitions in RTE; generally used for higher-priority programs. The "foreground" area is synonymous with the real-time area.

GLOBAL TRACKS - Global tracks are a subset of system tracks and are accounted for in the track assignment table. Any program can read/write or release a global track (i.e., programs can share global tracks).

HP-IB - The Hewlett-Packard version of the IEEE standard 488-1975 Digital Interface for Programmable Instrumentation. The HP-IB provides two-way communication between instruments and/or between computers, instruments, or peripherals.

ID SEGMENT - A block of words, associated with each resident program, that is used by the system to keep track of the program's name, software priority field, current scheduling status and other characteristics. Every program must have its own ID segment.

ID SEGMENT EXTENSION - A method for increasing the size of an ID segment to save additional information about its associated program. The extensions are used only for EMA programs (see EMA). ID segment extensions are automatically allocated by the generator or loader, but only if sufficient ID segment extensions were specified during system generation.

INTERRUPT - The process (usually initiated by an I/O device controller) that causes the computer to signal an executing program, in an orderly fashion, for the purpose of transferring information between a device and the computer.

INTERRUPT LOCATION - A single memory location whose contents (always an instruction) are executed upon interrupt by an I/O device controller (same as trap cell).

INTERRUPT TABLE (INT) - A table that associates interrupt links with the octal select codes of peripheral devices to specific EQT entries or programs.

I/O - A general term referring to any activity between a computer and its peripheral devices.

I/O CONTROLLER - A combination of interface card(s), cable, and (for some devices) controller box used to control one or more I/O devices.

I/O DEVICE - A physical unit defined by an EQT entry (I/O controller) and subchannel.

I/O WITHOUT WAIT - See CLASS I/O.

KEYWORD TABLE - A table of ID segment addresses.

LG AREA - A group of tracks used to temporarily store relocatable code that can be accessed by the File Manager.

LIBRARY - A collection of relocatable subroutines that perform commonly-used (e.g., mathematical) functions. Subroutines are appended to referencing programs or are placed in the memory resident library for access by memory resident programs.

LOADER - A program that converts the relative addresses of relocatable programs to absolute addresses compatible with the memory layout of a particular system.

LOCAL COMMON - An area of COMMON appended to the beginning of a program and accessible only by that program, its subroutines or segments. This type of COMMON can be specified only during on-line relocation by the loader (LOADR).

LOCKED DEVICE - See LOGICAL UNIT LOCK.

LOCKED FILE - A file opened exclusively to one program and therefore not currently accessible to any other program.

LOGICAL MEMORY - Logical memory is the 32K-word (maximum) address space described by the currently enabled memory map. If the System Map is enabled, it describes those areas of physical memory necessary for the operation of RTE-IV. When the User Map is enabled, it describes those areas needed by the currently executing program. DCPC maps describe the address space to/from which the transfer is taking place.

LOGICAL UNIT LOCK - A mechanism for temporarily acquiring exclusive use of an I/O device or devices by a program, to ensure its I/O completion before being preempted by a another program.

LOGICAL UNIT NUMBER (LU) - A number used by a program to refer to an I/O device. Programs do not refer directly to the physical I/O device select code number, but rather through the LU number that has a cross-reference to the device.

LOG-OFF - The process by which a session is terminated.

LOG-ON - The process by which a session is initiated. The Log-On process involves checking the session user's identification to allow access to the system, and the creation of the user's operating environment through the User HELLO File and Session Control Block.

LU - See LOGICAL UNIT NUMBER.

MAILBOX I/O - A Class I/O term applied to a protected buffer that keeps track of the "sender" and "receiver" program for each block of data in the buffer used in program-to program communication.

MAIN PROGRAM - The main body of a user program (as opposed to the whole program, which may include subroutines or segments).

MAP - Applied to 21MX or XE machines, the term applies to a set of 32 registers that point to 32 pages of physical memory defining a 32K-word logical address space.

MAPPING SEGMENT (MSEG) - The area of an EMA that is currently accessible within the user program's logical address space.

MEMORY PROTECT - A hardware accesory that allows an address (memory protection fence) to be set so that when in protected mode, the locations below that address cannot be accessed by writes or JSB/JMP instructions.

MEMORY-RESIDENT LIBRARY - A collection of reentrant or privileged library routines available only to memory resident programs (in RTE-IV). These routines are included in the disc-resident relocatable library for appending to disc-resident programs.

MEMORY-RESIDENT PROGRAM - A program that executes from a designated area in physical memory and remains in memory, as opposed to a disc-resident program that may be swapped out to the disc or loaded from the disc to another area in memory. Memory resident programs are loaded during system generation (only), and usually are high priority programs with short execution times.

MOTHER PARTITION - A partition that may be larger than the maximum logical address space and which may consist of a group of subpartitions. The subpartitions allow many smaller programs to use the memory when the mother partition is not active.

MSEG - See MAPPING SEGMENT.

MULTIPROGRAMMING - A technique whereby two or more routines or programs may be executed concurrently by an interleaving process, using the same computer. Multiprogramming is an attempt to improve equipment efficiency by building a queue of demands for resources, achieved by having available in main memory more than one task waiting for resource usage. The concurrent tasks are then multiplexed among each other's wait time intervals.

MULTI-TERMINAL MONITOR - A system software module that provides for interactive program development and editing in a multi-terminal environment controlled by a single computer.

OFF-LINE - Refers to use of the computer and/or I/O devices by resources other than the RTE operating system or subsystems.

ON-LINE - Refers to software or I/O devices recognized and controlled by the main operating system at the time they are being used.

ON-LINE GENERATOR - A program that permits use of an existing RTE operating system's services to generate a new system from relocatable software modules found in the File Manager Area. System control can then be transferred to the new operating system through use of a program called SWTCH. (See RTE-IVB On-Line Generator Reference Manual.)

ON-LINE LOADING - The relocation of programs through use of the Relocating Loader (see RELOCATION).

OPEN FILE - A method of gaining access to a specific file to perform a read/write instruction.

OPERATOR'S CONSOLE - See SYSTEM CONSOLE.

OPERATING SYSTEM - An organized collection of programs designed to optimize the usage of a computer system. It provides the means by which user programs interact with hardware and other software. (See also REAL-TIME EXECUTIVE.)

OVERLAYS - Also called segments,these are routines that share the same portion of main memory and are called into memory by the program itself (see SEGMENTED PROGRAMS).

PAGE - The largest block of memory (1024 words) that can be directly addressed by the address field of a one-word memory reference instruction.

PARTITION - A predefined block of memory with a fixed number of pages (redefinable at system boot-up) located in the disc resident program area of memory. The user may divide the disc resident program area into as many as 64 partitions that can be classified as a mixture of real-time and background, all real-time, or all background. Disc-resident programs run in partitions and at least one partition of sufficient size must be defined during system generation to run disc resident programs.

PERIPHERAL DISC SUBCHANNEL - A disc subchannel available to the user for read/write operations but for which RTE-IVB does not manage nor maintain a track assignment table. It is the user's responsibility to manage these tracks; however, the File Manager may be used to manage peripheral subchannel tracks. A peripheral subchannnel must have a logical unit number assignment greater than 6.

PHYSICAL MEMORY - Physical memory is the total amount of memory defined at generation or reconfiguration time. It refers to the actual memory in the computer; e.g., page 67 of physical memory is associated with a certain block of actual hardware, whereas the same page might be referred to as "page 5" in a particular block of logical memory.

POWER FAIL/AUTO-RESTART - The ability for a computer to save the current state of the system in permanent memory when power is lost, and to restore the system to defined conditions when power returns.

PRIORITY - A regulation of events allowing certain actions to take precedence over others in case of timing conflicts.

PRIVILEGED DRIVERS - I/O drivers whose interrupts are not processed by the RTE operating system. Such drivers offer improved response time but must perform their own internal housekeeping; i.e., saving status upon interrupt.

PRIVILEGED INTERRUPTS - Interrupts that by-pass normal interrupt processing to achieve optimum response time for interrupts having the greatest urgency. Privileged interrupts are handled by privileged I/O drivers.

PRIVILEGED SUBROUTINE - A privileged subroutine executes with the interrupt system off (and therefore by-passes the operating system). It allows high-speed processing at the cost of losing use of operating system housekeeping services and real-time response.

PROGRAM STATE - Refers to the status of an executable program at any given time. A user program is always in one of four possible states: executing, scheduled, suspended or dormant.

PROGRAM SWAPPING - See Swapping.

PURGE - Refers to the act of instructing an operating system to delete a file or program from its directory. Usually used with reference to disc files.

REAL-TIME (RT) - An arbitrary name for one of the two types of partitions in RTE; generally used for higher-priority programs. The real-time area is synonymous with the "foreground" area.

REAL-TIME EXECUTIVE - A collection of software modules comprising the total operating system; e.g., EXEC, SCHED, RTIOC, I/O drivers and various tables. For all practical purposes, Real-Time Executive, operating system and RTE are synonymous terms.

RECORD - A logical subdivision of a file that contains zero or more words, and is terminated by an end-of-record mark.

REENTRANT - Refers to a routine that can be shared by a number of programs simultaneously; i.e., one program can be interrupted in its usage of the routine to permit a higher-priority program to utilize the routine. The first program can then reenter the routine at the point where it was interrupted.

RELOCATABLE LIBRARIES - A collection of commonly-used subroutines in relocatable format. For example:

System Library - subroutines that are appended to each user program and that are unique to the operating system. This allows a user to write programs using operating system routines but which are independent of the operating system for subroutine execution.

DOS/RTE Relocatable Library - a collection of utility subroutines that are primarily accessed by FORTRAN and Assembly Language programs.

FORTRAN Formatters - format subroutines for FORTRAN I/O operations and other programming languages.

RELOCATING LOADER (LOADR) - A HP-supplied program that sets up communications links and forms an absolute load module from a relocatable program. LOADR creates the relocated program in conformance with current system constraints and loader commands entered by the user.

RESOURCE MANAGEMENT - A feature that allows the user to manage a specific resource shared by a particular set of cooperating programs.

RESPONSE TIME - The total amount of time required to bring a real-time program or routine into execution in response to an interrupt, interval timer, call from another program or operator call. Response time is usually measured in microseconds to milliseconds.

ROM BOOT - A loader residing in Read-Only Memory that on-line loads the Boot Extension from disc storage and transfers control to the Boot Extension. The Boot Extension must reside on the disc physical unit 0, track 0, sector 0. (See also Boot Extension and Startup definitions.)

RTE - See REAL-TIME EXECUTIVE.

SAM - See SYSTEM AVAILABLE MEMORY.

SCB - See SESSION CONTROL BLOCK.

SCHEDULING - Entering a program in the schedule list for execution, either at the next entry into the dispatcher, or at the appropriate time when the program's priority is high enough.

SEGMENTED PROGRAM - A technique for accommodating programs larger than the available logical memory. "Segment" refers to those slices of the program that are brought into main memory as required, and overlay the previous segment.

SELECT CODE - An octal number (10 through 77) that specifies the address of an I/O device interface card.

SESSION CONTROL BLOCK (SCB) - A variable-length table built in physical memory by the log-on processor for each session.

SESSION IDENTIFIER - A number by which the system identifies each session. Typically it is the system logical unit number of the terminal on which the session user has logged on.

SESSION SWITCH TABLE (SST) - A table which defines a session's total I/O addressing range. It provides a mapping between Session Logical Unit numbers which the user addresses and System Logical Unit numbers which is where the system processes the call.

SIMULTANEOUS PERIPHERAL OPERATIONS ON-LINE (SPOOL) - An RTE feature generally associated with batch operations. There is both in-spooling and out-spooling . In-spooling consists of a program and data being first read in from some peripheral device and placed on the disc. Program reads are translated to disc reads instead of reads from the peripheral device. Program writes are also translated to disc writes instead of peripheral device writes, so that program output is on disc. Out-spooling is the process of taking the program's output from disc to the appropriate peripheral device.

SPARE CARTRIDGE POOL - A set of cartridges defined by the System Manager as being available to session users for temporary disc space.

SST - See SESSION SWITCH TABLE.

STARTUP - The startup process is initiated by the Boot Extension. During the startup process, the tables, registers and pointers required by the system are established. Control is then transferred to the Configurator.

STATION - A terminal and its associated peripheral devices.

SUBCHANNEL - One of a group of I/O devices connected to a single I/O controller. For example, RTE driver DVR23 can operate more than one magnetic tape drive through subchannel assignments. In the case of moving head discs, contiguous groups of tracks are treated as separate subchannels. For example, a 7905 disc platter may be divided into four subchannels. Each subchannel is referenced by an LU number.

SUBCHANNEL INITIALIZATION - The process of preparing a disc subchannel for use by the RTE operating system.

SUBCHANNEL NUMBERS - Decimal numbers (0-31) associated with the LU numbers of devices with multiple functions on the same device. Each subchannel number is associated with a specific subchannel; e.g., a 2645A terminal could have four subchannels: one for the keyboard, one each for the right and left tape channels, and one for an optional line printer.

SUBPARTITIONS - Partitions that are optional subdivisions of a mother partition. Subpartitions have the same type (RT or BG) as the mother partition. Subpartitions are treated like other partitions except that they cannot be used while the mother partition contains an executing program.

SUBSYSTEM GLOBAL AREA (SSGA) - An area of memory that consists of all Type 30 modules loaded at generation time. The area is included in the system address space and in the address spaces of programs that access it (Types 17-20, and Types 25-28). The area may be used for data (i.e., COMMON).

SWAPPING - A technique whereby an executing program is suspended and transferred to mass storage (because another program needing the same portion of memory has been scheduled). When the interrupting program has terminated, becomes suspended, or becomes eligible to be swapped out, the previously swapped program may be reloaded into memory and resumes execution at the point where it was suspended.

SWTCH PROGRAM - A system utility program that transfers an RTE-IV operating system to a specific disc area from which it can be booted up.

SYNCHRONOUS DEVICE - Devices that perform I/O operations in a fixed timing sequence, regardless of the readiness of the computer.

SYSTEM AVAILABLE MEMORY (SAM) - A temporary storage area used by the system for class I/O, reentrant I/O, automatic buffering and parameter string passing.

SYSTEM COMMON - An area of memory that is sharable by programs.

SYSTEM CONSOLE - The interactive console or terminal (LU1) that controls system operation and from which all system and utility error messages are issued. In a multi-terminal environment, a system console is distinguished from "user consoles" from which users develop programs.

SYSTEM DISC SUBCHANNEL - The disc subchannel assigned to Logical Unit 2 that contains the memory image of the RTE-IVB system.

SYSTEM DRIVER AREA - An area for privileged drivers, very large drivers, drivers that do their own mapping or drivers not included in driver partitions. It is included in the system's address space, in the address space of RT and Type 3 BG programs, and optionally in the address space of memory resident programs.

SYSTEM MAP - The 32K-word address space used by the operating system during its own exectuion.

SYSTEM TRACKS - All subchannel tracks assigned to RTE-IVB for which a contiguous track assignment table is maintained. These tracks are located on Logical Unit 2 (system), and 3 (auxiliary).

TABLE AREA I - An area of memory that is included in all address spaces and which includes the EQTs, Device Reference Table, Interrupt Table, Track Map Table, all Type 15 modules, and some system entry points.

TABLE AREA II - An area of memory that contains the system tables, ID segments, all Type 13 modules, and some system table and entry points. Table Area II is included in the address space of the system, real-time programs, Type 3 background programs, and (optionally) memory resident programs.

TIME BASE GENERATOR (TBG) - A hardware module (real-time clock) that generates an interrupt in 10 millisecond intervals. It is used to trigger execution of time-scheduled user programs at pre-determined intervals and for device time-outs.

TIME-OUT - Relating to the state of a peripheral device. When the device has timed-out, it is no longer available for system use (down). Also (noun) the parameter itself; the amount of time RTE will wait for the device to respond to an I/O transfer command before making the device unavailable.

TIME SCHEDULING - The process of automatically scheduling a program for execution at pre-determined time intervals. Program scheduling is established through use of the IT command, and requires that the Time Base Generator be installed in the CPU.

TIMESLICING - A means by which compute bound programs can be prevented from monopolizing CPU time. A timesliced program is placed in a round-robin queue with other programs of the same priority. Each program in the queue gets a quantum of CPU time to execute before another program gets its turn. Higher priority programs can interrupt any timesliced program at any time.

UP - See DEVICE UP.

USER HELLO FILE - A procedure file that control is automatically transferred to when the session user first logs on to the system.

USER MAP - The 32K-word address space used by a user program during its execution.

Index

Index

F

G

H

I

J

K

L

M

S

T

Y

Z

SYMBOLIC TERMS

READER COMMENT SHEET

RTE-IVB TERMINAL USER'S
Reference Manual

92068-90002 **January 1983**

Update No. _____

(If Applicable)

We welcome your evaluation of this manual. Your comments and suggestions help us improve our publications. Please use additional pages if necessary.

FROM:

Name _____

Company _____

Address _____

_____

_____

FOLD FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 141 CUPERTINO, CA.

— POSTAGE WILL BE PAID BY —

Hewlett-Packard Company
Data Systems Division
11000 Wolfe Road
Cupertino, California 95014
ATTN: Technical Publications

FOLD FOLD

SALES & SUPPORT OFFICES

Arranged Alphabetically by Country

Product Line Sales/Support Key

Key Product Line

A	Analytical
CM	Components
C	Computer Systems Sales only
CH	Computer Systems Hardware Sales and Services
CS	Computer Systems Software Sales and Services
E	Electronic Instruments & Measurement Systems
M	Medical Products
MP	Medical Products Primary SRO
MS	Medical Products Secondary SRO
P	Personal Computation Products
*	Sales only for specific product line
**	Support only for specific product line

IMPORTANT: These symbols designate general product line capability. They do not insure sales or support availability for all products within a line, at all locations. Contact your local sales office for information regarding locations where HP support is available for specific products.

HP distributors are printed in italics.

ANGOLA

Telectra
Empresa Técnica de Equipamentos Eléctricos, S.A.R.L.
R. Barbosa Rodrigues, 41-I DT.
Caixa Postal 6487
LUANDA
Tel: 35515,35516
E,M,P

ARGENTINA

Hewlett-Packard Argentina S.A.
Avenida Santa Fe 2035
Martinez 1640 **BUENOS AIRES**
Tel: 798-5735, 792-1293
Telex: 17595 BIONAR
Cable: HEWPACKARG
A,E,CH,CS,P

Biotron S.A.C.I.M. e I.
Av Paseo Colon 221, Piso 9
*1399 **BUENOS AIRES,***
Tel: 30-4846, 30-1851
Telex: 17595 BIONAR
M

Fate S.A. I.C.I.Electronica
Venezuela 1326
*1095 **BUENOS AIRES***
Tel: 37-9020, 37-9026/9
Telex: 9234 FATEN AR
P

AUSTRALIA

Adelaide, South Australia Office

Hewlett-Packard Australia Ltd.
153 Greenhill Road
PARKSIDE, S.A. 5063
Tel: 272-5911
Telex: 82536
Cable: HEWPARD Adelaide
A*,CH,CM,,E,MS,P

Brisbane, Queensland Office

Hewlett-Packard Australia Ltd.
49 Park Road
MILTON, Queensland 4064
Tel: 229-1544
Telex: 42133
Cable: HEWPARD Brisbane
A,CH,CM,E,M,P
Effective November 1, 1982:
10 Payne Road
THE GAP, Queensland 4061
Tel: 30-4133
Telex: 42133

Canberra, Australia Capital Territory Office

Hewlett-Packard Australia Ltd.
121 Wollongong Street
FYSHWICK, A.C.T. 2609
Tel: 80 4244
Telex: 62650
Cable: HEWPARD Canberra
CH,CM,E,P

Melbourne, Victoria Office

Hewlett-Packard Australia Ltd.
31-41 Joseph Street
BLACKBURN, Victoria 3130
Tel: 877 7777
Telex: 31-024
Cable: HEWPARD Melbourne
A,CH,CM,CS,E,MS,P

Perth, Western Australia Office

Hewlett-Packard Australia Ltd.
261 Stirling Highway
CLAREMONT, W.A. 6010
Tel: 383-2188
Telex: 93859
Cable: HEWPARD Perth
A,CH,CM,,E,MS,P

Sydney, New South Wales Office

Hewlett-Packard Australia Ltd.
17-23 Talavera Road
P.O. Box 308
NORTH RYDE, N.S.W. 2113
Tel: 887-1611
Telex: 21561
Cable: HEWPARD Sydney
A,CH,CM,CS,E,MS,P

AUSTRIA

Hewlett-Packard Ges.m.b.h.
Grottenhofstrasse 94
Verkaufsburo Graz
A-8052 **GRAZ**
Tel: 291-5-66
Telex: 32375
CH,E*

Hewlett-Packard Ges.m.b.h.
Stanglhofweg 5
A-4020 **LINZ**
Tel: 0732 51585
CH

Hewlett-Packard Ges.m.b.h.
Lieblgasse 1
P.O. Box 72
A-1222 **VIENNA**
Tel: (0222) 23-65-11-0
Telex: 134425 HEPA A
A,CH,CM,CS,E,MS,P

BAHRAIN

Green Salon
P.O. Box 557
BAHRAIN
Tel: 255503-255950
Telex: 84419
P

Wael Pharmacy
P.O. Box 648
BAHRAIN
Tel: 256123
Telex: 8550 WAEL BN
M, E

BELGIUM

Hewlett-Packard Belgium S.A./N.V.
Blvd de la Woluwe, 100
Woluwedal
B-1200 **BRUSSELS**
Tel: (02) 762-32-00
Telex: 23-494 paloben bru
A,CH,CM,CS,E,MP,P

BRAZIL

Hewlett-Packard do Brasil I.e.C. Ltda.
Alameda Rio Negro, 750
Alphaville 06400 **BARUERI SP**
Tel: (11) 421-1311
Telex: 01 133872 HPBR-BR
Cable: HEWPACK Sao Paulo
A,CH,CM,CS,E,M,P

Hewlett-Packard do Brasil I.e.C. Ltda.
Avenida Epitacio Pessoa, 4664
22471 **RIO DE JANEIRO**-RJ
Tel: (21) 286-0237
Telex: 021-21905 HPBR-BR
Cable: HEWPACK Rio de Janeiro
A,CH,CM,E,MS,P*

CANADA

Alberta

Hewlett-Packard (Canada) Ltd.
210, 7220 Fisher Street S.E.
CALGARY, Alberta T2H 2H8
Tel: (403) 253-2713
A,CH,CM,E*,MS,P*

Hewlett-Packard (Canada) Ltd.
11620A-168th Street
EDMONTON, Alberta T5M 3T9
Tel: (403) 452-3670
A,CH,CM,CS,E,MS,P*

British Columbia

Hewlett-Packard (Canada) Ltd.
10691 Shellbridge Way
RICHMOND,
British Columbia V6X 2W7
Tel: (604) 270-2277
Telex: 610-922-5059
A,CH,CM,CS,E*,MS,P*

Manitoba

Hewlett-Packard (Canada) Ltd.
380-550 Century Street
WINNIPEG, Manitoba R3H 0Y1
Tel: (204) 786-6701
A,CH,CM,E,MS,P*

New Brunswick

Hewlett-Packard (Canada) Ltd.
37 Sheadiac Road
MONCTON, New Brunswick E2B 2VQ
Tel: (506) 855-2841
CH**

Nova Scotia

Hewlett-Packard (Canada) Ltd.
P.O. Box 931
900 Windmill Road
DARTMOUTH, Nova Scotia B2Y 3Z6
Tel: (902) 469-7820
CH,CM,CS,E*,MS,P*

Ontario

Hewlett-Packard (Canada) Ltd.
552 Newbold Street
LONDON, Ontario N6E 2S5
Tel: (519) 686-9181
A,CH,CM,E*,MS,P*

Hewlett-Packard (Canada) Ltd.
6877 Goreway Drive
MISSISSAUGA, Ontario L4V 1M8
Tel: (416) 678-9430
A,CH,CM,CS,E,MP,P

Hewlett-Packard (Canada) Ltd.
2670 Queensview Dr.
OTTAWA, Ontario K2B 8K1
Tel: (613) 820-6483
A,CH,CM,CS,E*,MS,P*

Hewlett-Packard (Canada) Ltd.
220 Yorkland Blvd., Unit #11
WILLOWDALE, Ontario M2J 1R5
Tel: (416) 499-9333
CH

Quebec

Hewlett-Packard (Canada) Ltd.
17500 South Service Road
Trans-Canada Highway
KIRKLAND, Quebec H9J 2M5
Tel: (514) 697-4232
A,CH,CM,CS,E,MP,P*

Hewlett-Packard (Canada) Ltd.
Les Galeries du Vallon
2323 Du Versont Nord
STE. FOY, Quebec G1N 4C2
Tel: (418) 687-4570
CH

CHILE

Jorge Calcagni y Cia. Ltda.
Arturo Burhle 065
Casilla 16475
***SANTIAGO** 9*
Tel: 222-0222
Telex: Public Booth 440001
A,CM,E,M

Olympia (Chile) Ltda.
Av. Rodrigo de Araya 1045
Casilla 256-V
***SANTIAGO** 21*
Tel: 2-25-50-44
Telex: 340-892 OLYMP CK
Cable: Olympiachile Santiagochile
CH,CS,P

CHINA, People's Republic of

China Hewlett-Packard Rep. Office
P.O. Box 418
1A Lane 2, Luchang St.
Beiwei Rd., Xuanwu District
BEIJING
Tel: 33-1947, 33-7426
Telex: 22601 CTSHP CN
Cable: 1920
A,CH,CM,CS,E,P

COLOMBIA

Instrumentación
H. A. Langebaek & Kier S.A.
Carrera 7 No. 48-75
Apartado Aereo 6287
***BOGOTA** 1, D.E.*
Tel: 287-8877
Telex: 44400 INST CO
Cable: AARIS Bogota
A,CM,E,M,PS,P

COSTA RICA

Cientifica Costarricense S.A.
Avenida 2, Calle 5
San Pedro de Montes de Oca
Apartado 10159
SAN JOSE
Tel: 24-38-20, 24-08-19
Telex: 2367 GALGUR CR
CM,E,MS,P

CYPRUS

Telerexa Ltd.
P.O. Box 4809
14C Stassinos Avenue
NICOSIA
Tel: 62698
Telex: 2894 LEVIDO CY
E,M,P

DENMARK

Hewlett-Packard A/S
Datavej 52
DK-3460 **Birkerod**
Tel: (02) 81-66-40
Telex: 37409 hpas dk
A,CH,CM,CS,E,MS,P

Hewlett-Packard A/S
Navervej 1
DK-8600 **SILKEBORG**
Tel: (06) 82-71-66
Telex: 37409 hpas dk
CH,E

ECUADOR

CYEDE Cia. Ltda.
Avenida Eloy Alfaro 1749
Casilla 6423 CCI
QUITO
Tel: 450-975, 243-052
Telex: 2548 CYEDE ED
A,CM,E,P

Hospitalar S.A.
Robles 625
Casilla 3590
QUITO
Tel: 545-250, 545-122
Telex: 2485 HOSPTL ED
Cable: HOSPITALAR-Quito
M

EGYPT

International Engineering Associates
24 Hussein Hegazi Street
Kasr-el-Aini
CAIRO
Tel: 23829, 21641
Telex: IEA UN 93830
CH,CS,E,M

Informatic For Systems
22 Talaat Harb Street
CAIRO
Tel: 759006
Telex: 93938 FRANK UN
CH,CS,P

Egyptian International Office for Foreign Trade
P.O.Box 2558
CAIRO
Tel: 650021
Telex: 93337 EGPOR
P

EL SALVADOR

IPESA de El Salvador S.A.
29 Avenida Norte 1216
SAN SALVADOR
Tel: 26-6858, 26-6868
Telex: Public Booth 20107
A,CH,CM,CS,E,P

FINLAND

Hewlett-Packard Oy
Revontulentie 7
SF-02100 **ESPOO** 10
Tel: (90) 455-0211
Telex: 121563 hewpa sf
A,CH,CM,CS,E,MS,P

Hewlett-Packard Oy
Aatoksenkatv 10-C

SALES & SUPPORT OFFICES

Arranged Alphabetically by Country

SF-40720-72 **JYVASKYLA**
Tel: (941) 216318
CH

Hewlett-Packard Oy
Kainvuntie 1-C
SF-90140-14 **OULU**
Tel: (981) 338785
CH

FRANCE

Hewlett-Packard France
Z.I. Mercure B
Rue Berthelot
F-13763 Les Milles Cedex
AIX-EN-PROVENCE
Tel: (42) 59-41-02
Telex: 410770F
A,CH,E,MS,P*

Hewlett-Packard France
Boite Postale No. 503
F-25026 **BESANCON**
28 Rue de la Republique
F-25000 **BESANCON**
Tel: (81) 83-16-22
CH,M

Hewlett-Packard France
Bureau de Vente de Lyon
Chemin des Mouilles
Boite Postale 162
F-69130 **ECULLY** Cédex
Tel: (7) 833-81-25
Telex: 310617F
A,CH,CS,E,MP

Hewlett-Packard France
Immeuble France Evry
Tour Lorraine
Boulevard de France
F-91035 **EVRY** Cédex
Tel: (6) 077-96-60
Telex: 692315F
E

Hewlett-Packard France
5th Avenue Raymond Chanas
F-38320 **EYBENS**
Tel: (76) 25-81-41
Telex: 980124 HP GRENOB EYBE
CH

Hewlett-Packard France
Centre d'Affaire Paris-Nord
Bâtiment Ampère 5 étage
Rue de la Commune de Paris
Boite Postale 300
F-93153 **LE BLANC MESNIL**
Tel: (01) 865-44-52
Telex: 211032F
CH,CS,E,MS

Hewlett-Packard France
Parc d'Activites Cadera
Quartier Jean Mermoz
Avenue du President JF Kennedy
F-33700 **MERIGNAC**
Tel: (56) 34-00-84
Telex: 550105F
CH,E,MS

Hewlett-Packard France
32 Rue Lothaire
F-57000 **METZ**
Tel: (8) 765-53-50
CH

Hewlett-Packard France
Immueble Les 3 B
Nouveau Chemin de la Garde
Z.A.C. de Bois Briand
F-44085 **NANTES** Cedex
Tel: (40) 50-32-22
CH**

Hewlett-Packard France
Zone Industrielle de Courtaboeuf
Avenue des Tropiques
F-91947 Les Ulis Cédex **ORSAY**
Tel: (6) 907-78-25
Telex: 600048F
A,CH,CM,CS,E,MP,P

Hewlett-Packard France
Paris Porte-Maillot
15, Avenue De L'Amiral Bruix
F-75782 **PARIS** 16
Tel: (1) 502-12-20
Telex: 613663F
CH,MS,P

Hewlett-Packard France
2 Allee de la Bourgonette
F-35100 **RENNES**
Tel: (99) 51-42-44
Telex: 740912F
CH,CM,E,MS,P*

Hewlett-Packard France
98 Avenue de Bretagne
F-76100 **ROUEN**
Tel: (35) 63-57-66 CH**,CS

Hewlett-Packard France
4 Rue Thomas Mann
Boite Postale 56
F-67200 **STRASBOURG**
Tel: (88) 28-56-46
Telex: 890141F
CH,E,MS,P*

Hewlett-Packard France
Pericentre de la Cépière
F-31081 **TOULOUSE** Cedex
Tel: (61) 40-11-12
Telex: 531639F
A,CH,CS,E,P*

Hewlett-Packard France
Immeuble Péricentre
F-59658 **VILLENEUVE D'ASCQ** Cedex
Tel: (20) 91-41-25
Telex: 160124F
CH,E,MS,P*

GERMAN FEDERAL REPUBLIC

Hewlett-Packard GmbH
Technisches Büro Berlin
Keithstrasse 2-4
D-1000 **BERLIN** 30
Tel: (030) 24-90-86
Telex: 018 3405 hpbln d
A,CH,E,M,P

Hewlett-Packard GmbH
Technisches Büro Böblingen
Herrenberger Strasse 110
D-7030 **BÖBLINGEN**
Tel: (07031) 667-1
Telex: bbn or
A,CH,CM,CS,E,MP,P

Hewlett-Packard GmbH
Technisches Büro Dusseldorf
Emanuel-Leutze-Strasse 1
D-4000 **DUSSELDORF**
Tel: (0211) 5971-1
Telex: 085/86 533 hpdd d
A,CH,CS,E,MS,P

Hewlett-Packard GmbH
Vertriebszentrale Frankfurt
Berner Strasse 117
Postfach 560 140
D-6000 **FRANKFURT** 56
Tel: (0611) 50-04-1
Telex: 04 13249 hpffm d
A,CH,CM,CS,E,MP,P

Hewlett-Packard GmbH
Technisches Büro Hamburg
Kapstadtring 5
D-2000 **HAMBURG** 60
Tel: (040) 63804-1
Telex: 021 63 032 hphh d
A,CH,CS,E,MS,P

Hewlett-Packard GmbH
Technisches Büro Hannover
Am Grossmarkt 6
D-3000 **HANNOVER** 91
Tel: (0511) 46-60-01
Telex: 092 3259
A,CH,CM,E,MS,P

Hewlett-Packard GmbH
Technisches Büro Mannheim
Rosslauer Weg 2-4
D-6800 **MANNHEIM**
Tel: (0621) 70050
Telex: 0462105
A,C,E

Hewlett-Packard GmbH
Technisches Büro Neu Ulm
Messerschmittstrasse 7
D-7910 **NEU ULM**
Tel: 0731-70241
Telex: 0712816 HP ULM-D
A,C,E*

Hewlett-Packard GmbH
Technisches Büro Nürnberg
Neumeyerstrasse 90
D-8500 **NÜRNBERG**
Tel: (0911) 52 20 83-87
Telex: 0623 860
CH,CM,E,MS,P

Hewlett-Packard GmbH
Technisches Büro München
Eschenstrasse 5
D-8028 **TAUFKIRCHEN**
Tel: (089) 6117-1
Telex: 0524985
A,CH,CM,E,MS,P

GREAT BRITAIN

Hewlett-Packard Ltd.
Trafalgar House
Navigation Road
ALTRINCHAM
Chesire WA14 1NU
Tel: (061) 928-6422
Telex: 668068
A,CH,CS,E,M

Hewlett-Packard Ltd.
Oakfield House, Oakfield Grove
Clifton
BRISTOL BS8 2BN, Avon
Tel: (027) 38606
Telex: 444302
CH,M,P

Hewlett-Packard Ltd.
(Pinewood)
Nine Mile Ride
EASTHAMPSTEAD
Wokingham
Berkshire, 3RG11 3LL
Tel: 3446 3100
Telex: 84-88-05
CH,CS,E

Hewlett-Packard Ltd.
Fourier House
257-263 High Street
LONDON COLNEY
Herts., AL2 1HA, St. Albans
Tel: (0727) 24400
Telex: 1-8952716
CH,CS,E

Hewlett-Packard Ltd
Tradax House, St. Mary's Walk
MAIDENHEAD
Berkshire, SL6 1ST
Tel: (0628) 39151
CH,CS,E,P

Hewlett-Packard Ltd.
Quadrangle
106-118 Station Road
REDHILL, Surrey
Tel: (0737) 68655
Telex: 947234 CH,CS,E

Hewlett-Packard Ltd.
Avon House
435 Stratford Road
SHIRLEY, Solihull
West Midlands B90 4BL
Tel: (021) 745 8800
Telex: 339105
CH

Hewlett-Packard Ltd.
West End House 41
High Street, West End
SOUTHAMPTON
Hampshire S03 3DQ
Tel: (703) 886767
Telex: 477138
CH

Hewlett-Packard Ltd.
King Street Lane
WINNERSH, Wokingham
Berkshire RG11 5AR
Tel: (0734) 784774
Telex: 847178
A,CH,E,M

GREECE

Kostas Karaynnis S.A.
8 Omirou Street
***ATHENS** 133*
Tel: 32 30 303, 32 37 371
Telex: 215962 RKAR GR
A,CH,CM,CS,E,M,P

PLAISIO S.A.
G. Gerardos
24 Stournara Street
ATHENS
Tel: 36-11-160
Telex: 221871
P

GUATEMALA

IPESA
Avenida Reforma 3-48, Zona 9
GUATEMALA CITY
Tel: 316627, 314786
Telex: 4192 TELTRO GU
A,CH,CM,CS,E,M,P

HONG KONG

Hewlett-Packard Hong Kong, Ltd.
G.P.O. Box 795
5th Floor, Sun Hung Kai Centre
30 Harbour Road
HONG KONG
Tel: 5-8323211
Telex: 66678 HEWPA HX
Cable: HEWPACK HONG KONG
E,CH,CS,P

CET Ltd.
1402 Tung Way Mansion
199-203 Hennessy Rd.
*Wanchia, **HONG KONG***
Tel: 5-729376
Telex: 85148 CET HX
CM

Schmidt & Co. (Hong Kong) Ltd.
Wing On Centre, 28th Floor
Connaught Road, C.
HONG KONG
Tel: 5-455644
Telex: 74766 SCHMX HX
A,M

ICELAND

Elding Trading Company Inc.
Hafnarnvoli-Tryggvagotu
P.O. Box 895
IS-REYKJAVIK
Tel: 1-58-20, 1-63-03
M

INDIA

Blue Star Ltd.
Sabri Complex II Floor
24 Residency Rd.
***BANGALORE** 560 025*
Tel: 55660
Telex: 0845-430
Cable: BLUESTAR
A,CH,CM,CS,E

Blue Star Ltd.
Band Box House
Prabhadevi
***BOMBAY** 400 025*
Tel: 422-3101
Telex: 011-3751
Cable: BLUESTAR
A,M

Blue Star Ltd.
Sahas
414/2 Vir Savarkar Marg
Prabhadevi
***BOMBAY** 400 025*
Tel: 422-6155
Telex: 011-4093
Cable: FROSTBLUE
A,CH,CM,CS,E,M

Blue Star Ltd.
Kalyan, 19 Vishwas Colony
*Alkapuri, **BORODA**, 390 005*
Tel: 65235
Cable: BLUE STAR
A

Blue Star Ltd.
7 Hare Street
***CALCUTTA** 700 001*
Tel: 12-01-31
Telex: 021-7655
Cable: BLUESTAR
A,M

Blue Star Ltd.
133 Kodambakkam High Road
***MADRAS** 600 034*
Tel: 82057
Telex: 041-379
Cable: BLUESTAR
A,M

Blue Star Ltd.
Bhandari House, 7th/8th Floors
91 Nehru Place
***NEW DELHI** 110 024*
Tel: 682547
Telex: 031-2463
Cable: BLUESTAR
A,CH,CM,CS,E,M

Blue Star Ltd.
15/16:C Wellesley Rd.
***PUNE** 411 011*
Tel: 22775
Cable: BLUE STAR
A

Blue Star Ltd.
2-2-47/1108 Bolarum Rd.
***SECUNDERABAD** 500 003*
Tel: 72057
Telex: 0155-459
Cable: BLUEFROST
A,E

Blue Star Ltd.
T.C. 7/603 Poornima
Maruthankuzhi
***TRIVANDRUM** 695 013*
Tel: 65799
Telex: 0884-259
Cable: BLUESTAR
E

INDONESIA

BERCA Indonesia P.T.
P.O.Box 496/JKT.
Jl. Abdul Muis 62
JAKARTA
Tel: 373009
Telex: 46748 BERSAL IA
Cable: BERSAL JAKARTA
P

BERCA Indonesia P.T.
Wisma Antara Bldg., 17th floor
JAKARTA
A,CS,E,M
BERCA Indonesia P.T.
P.O. Box 174/SBY.
Jl. Kutei No. 11
SURABAYA
Tel: 68172
Telex: 31146 BERSAL SB
Cable: BERSAL-SURABAYA
A,E,M,P*

IRAQ
Hewlett-Packard Trading S.A.
Service Operation
Al Mansoor City 9B/3/7
BAGHDAD
Tel: 551-49-73
Telex: 212-455 HEPAIRAQ IK
CH,CS

IRELAND
Hewlett-Packard Ireland Ltd.
82/83 Lower Leeson St.
DUBLIN 2
Tel: (1) 60 88 00
Telex: 30439
A,CH,CM,CS,E,M,P

Cardiac Services Ltd.
Kilmore Road
Artane
***DUBLIN** 5*
Tel: (01) 351820
Telex: 30439
M

ISRAEL
Eldan Electronic Instrument Ltd.
P.O. Box 1270
JERUSALEM 91000
16, Ohaliav St.
JERUSALEM 94467
Tel: 533 221, 553 242
Telex: 25231 AB/PAKRD IL
A

Electronics Engineering Division
Motorola Israel Ltd.
16 Kremenetski Street
P.O. Box 25016
TEL-AVIV 67899
Tel: 3-338973
Telex: 33569 Motil IL
Cable: BASTEL Tel-Aviv
CH,CM,CS,E,M,P

ITALY
Hewlett-Packard Italiana S.p.A.
Traversa 99C
Via Giulio Petroni, 19
I-70124 **BARI**
Tel: (080) 41-07-44
M

Hewlett-Packard Italiana S.p.A.
Via Martin Luther King, 38/111
I-40132 **BOLOGNA**
Tel: (051) 402394
Telex: 511630
CH,E,MS

Hewlett-Packard Italiana S.p.A.
Via Principe Nicola 43G/C
I-95126 **CATANIA**
Tel: (095) 37-10-87
Telex: 970291
C,P

Hewlett-Packard Italiana S.p.A.
Via G. Di Vittorio 9
I-20063 **CERNUSCO SUL NAVIGLIO**
Tel: (2) 903691
Telex: 334632
A,CH,CM,CS,E,MP,P

Hewlett-Packard Italiana S.p.A.
Via Nuova San Rocco a
Capodimonte, 62/A
I-80131 **NAPLES**
Tel: (081) 7413544
Telex: 710698
A,CH,E

Hewlett-Packard Italiana S.p.A.
Viale G. Modugno 33
I-16156 **GENOVA PEGLI**
Tel: (010) 68-37-07
Telex: 215238
E,C

Hewlett-Packard Italiana S.p.A.
Via Turazza 14
I-35100 **PADOVA**
Tel: (049) 664888
Telex: 430315
A,CH,E,MS

Hewlett-Packard Italiana S.p.A.
Viale C. Pavese 340
I-00144 **ROMA**
Tel: (06) 54831
Telex: 610514
A,CH,CM,CS,E,MS,P*

Hewlett-Packard Italiana S.p.A.
Corso Svizzera, 184
I-10149 **TORINO**
Tel: (011) 74 4044
Telex: 221079
CH,E

JAPAN
Yokogawa-Hewlett-Packard Ltd.
Inoue Building
1-21-8, Asahi-cho
ATSUGI, Kanagawa 243
Tel: (0462) 28-0451
CM,C*,E

Yokogawa-Hewlett-Packard Ltd.
Towa Building
2-2-3, Kaigandori, Chuo-ku
KOBE, 650, Hyogo
Tel: (078) 392-4791
C,E

Yokogawa-Hewlett-Packard Ltd.
Kumagaya Asahi Yasoji Bldg 4F
3-4 Chome Tsukuba
KUMAGAYA, Saitama 360
Tel: (0485) 24-6563
CH,CM,E

Yokogawa-Hewlett-Packard Ltd.
Asahi Shinbun Dai-ichi Seimei Bldg., 2F
4-7 Hanabata-cho
KUMAMOTO-SHI,860
Tel: (0963) 54-7311
CH,E

Yokogawa-Hewlett-Packard Ltd.
Shin Kyoto Center Bldg. 5F
614 Siokoji-cho
Nishiiruhigashi, Karasuma
Siokoji-dori, Shimogyo-ku
KYOTO 600
Tel: 075-343-0921
CH,E

Yokogawa-Hewlett-Packard Ltd.
Mito Mitsui Building
1-4-73, San-no-maru
MITO, Ibaragi 310
Tel: (0292) 25-7470
CH,CM,E

Yokogawa-Hewlett-Packard Ltd.
Sumitomo Seimei Nagoya Bldg.
2-14-19, Meieki-Minami,
Nakamura-ku
NAGOYA, 450 Aichi
Tel: (052) 571-5171
CH,CM,CS,E,MS

Yokogawa-Hewlett-Packard Ltd.
Chuo Bldg., 4th Floor
5-4-20 Nishinakajima,
Yodogawa-ku
OSAKA, 532
Tel: (06) 304-6021
Telex: YHPOSA 523-3624
A,CH,CM,CS,E,MP,P*

Yokogawa-Hewlett-Packard Ltd.
1-27-15, Yabe,
SAGAMIHARA Kanagawa, 229
Tel: 0427 59-1311

Yokogawa-Hewlett-Packard Ltd.
Shinjuku Dai-ichi Seimei 6F
2-7-1, Nishi Shinjuku
Shinjuku-ku, **TOKYO** 160
Tel: 03-348-4611-5
CH,E

Yokogawa-Hewlett-Packard Ltd.
3-29-21 Takaido-Higashi
Suginami-ku **TOKYO** 168
Tel: (03) 331-6111
Telex: 232-2024 YHPTOK
A,CH,CM,CS,E,MP,P*

Yokogawa-Hewlett-Packard Ltd.
Daiichi Asano Building 4F
5-2-8, Oodori,
UTSUNOMIYA, 320
Tochigi
Tel: (0286) 25-7155
CH, CS, E

Yokogawa-Hewlett-Packard Ltd.
Yasudaseimei Yokohama
Nishiguchi Bldg.
3-30-4 Tsuruya-cho
Kanagawa-ku
YOKOHAMA, Kanagawa, 221
Tel: (045) 312-1252
CH,CM,E

JORDAN
Mouasher Cousins Company
P.O. Box 1387
AMMAN
Tel: 24907, 39907
Telex: 21456 SABCO JO
CH,E,M,P

KENYA
ADCOM Ltd., Inc., Kenya
P.O.Box 30070
NAIROBI
Tel: 331955
Telex: 22639
E,M

KOREA
Samsung Electronics Computer Division
76-561 Yeoksam-Dong
Kangnam-Ku
C.P.O. Box 2775
SEOUL
Tel: 555-7555, 555-5447
Telex: K27364 SAMSAN
A,CH,CM,CS,E,M,P

KUWAIT
Al-Khaldiya Trading & Contracting
P.O. Box 830 Safat
KUWAIT
Tel: 42-4910, 41-1726
Telex: 22481 Areeg kt
CH,E,M

Photo & Cine Equipment
P.O. Box 270 Safat
KUWAIT
Tel: 42-2846, 42-3801
Telex: 22247 Matin-KT
P

LEBANON
G.M. Dolmadjian
Achrafieh
P.O. Box 165.167
BEIRUT
Tel: 290293
*MP***

LUXEMBOURG
Hewlett-Packard Belgium S.A./N.V.
Blvd de la Woluwe, 100
Woluwedal
B-1200 **BRUSSELS**
Tel: (02) 762-32-00
Telex: 23-494 paloben bru
A,CH,CM,CS,E,MP,P

MALAYSIA
Hewlett-Packard Sales (Malaysia)
Sdn. Bhd.
1st Floor, Bangunan British
American
Jalan Semantan, Damansara Heights
KUALA LUMPUR 23-03
Tel: 943022
Telex: MA31011
A,CH,E,M,P*

Protel Engineering
Lot 319, Satok Road
P.O.Box 1917
*Kuching, **SARAWAK***
Tel: 53544
Telex: MA 70904 PROMAL
Cable: PROTELENG
A,E,M

MALTA
Philip Toledo Ltd.
Notabile Rd.
MRIEHEL
Tel: 447 47, 455 66
Telex: 649 Media MW
P

MEXICO
Hewlett-Packard Mexicana, S.A. de C.V.
Av. Periferico Sur No. 6501
Tepepan, Xochimilco
MEXICO D.F. 16020
Tel: 676-4600
Telex: 17-74-507 HEWPACK MEX
A,CH,CS,E,MS,P

Effective November 1, 1982:
Hewlett-Packard Mexicana, S.A. de C.V.
Ejercito Nacional #570
Colonia Granada
11560 **MEXICO**, D.F.
CH**

Hewlett-Packard Mexicana, S.A. de C.V.
Rio Volga 600
Pte. Colonia del Valle
MONTERREY, N.L.
Tel: 78-42-93, 78-42-40, 78-42-41
Telex: 038-2410 HPMTY ME
CH
Effective Nov. 1, 1982
Ave. Colonia del Valle #409
Col. del Valle
Municinio de garza garcia
MONTERREY, N.V.

ECISA
Taihe 229, Piso 10
*Polanco **MEXICO D.F.** 11570*
Tel: 250-5391
Telex: 17-72755 ECE ME
M

MOROCCO
Dolbeau
81 rue Karatchi
CASABLANCA
Tel: 3041-82, 3068-38
Telex: 23051, 22822
E

Gerep
2 rue d'Agadir
Boite Postale 156
CASABLANCA
Tel: 272093, 272095
Telex: 23 739
P

NETHERLANDS
Hewlett-Packard Nederland B.V.
Van Heuven Goedhartlaan 121
NL 1181KK **AMSTELVEEN**
P.O. Box 667
NL1180 AR **AMSTELVEEN**
Tel: (20) 47-20-21
Telex: 13 216
A,CH,CM,CS,E,MP,P

Hewlett-Packard Nederland B.V.
Bongerd 2
NL 2906VK **CAPPELLE**, A/D Ijessel
P.O. Box 41
NL2900 AA **CAPELLE**, Ijssel
Tel: (10) 51-64-44
Telex: 21261 HEPAC NL
A,CH,CS

NEW ZEALAND
Hewlett-Packard (N.Z.) Ltd.
169 Manukau Road
P.O. Box 26-189
Epsom, **AUCKLAND**
Tel: 687-159
Cable: HEWPACK Auckland
CH,CM,E,P*

Hewlett-Packard (N.Z.) Ltd.
4-12 Cruickshank Street
Kilbirnie, **WELLINGTON 3**
P.O. Box 9443
Courtenay Place, **WELLINGTON 3**
Tel: 877-199
Cable: HEWPACK Wellington
CH,CM,E,P

Northrop Instruments & Systems Ltd.
369 Khyber Pass Road
P.O. Box 8602
AUCKLAND
Tel: 794-091
Telex: 60605
A,M

Northrop Instruments & Systems Ltd.
110 Mandeville St.
P.O. Box 8388
CHRISTCHURCH
Tel: 486-928
Telex: 4203
A,M

Northrop Instruments & Systems Ltd.
Sturdee House
85-87 Ghuznee Street
P.O. Box 2406
WELLINGTON
Tel: 850-091
Telex: NZ 3380
A,M

NORTHERN IRELAND
Cardiac Services Company
95A Finaghy Road South
***BELFAST** BT 10 OBY*
Tel: (0232) 625-566
Telex: 747626
M

NORWAY
Hewlett-Packard Norge A/S
Folke Bernadottes vei 50
P.O. Box 3558
N-5033 **FYLLINGSDALEN** (Bergen)
Tel: (05) 16-55-40
Telex: 16621 hpnas n
CH,CS,E,MS

Hewlett-Packard Norge A/S
Österndalen 18
P.O. Box 34
N-1345 **ÖSTERºAS**
Tel: (02) 17-11-80
Telex: 16621 hpnas n
A,CH,CM,CS,E,M,P

OMAN
Khimjil Ramdas
P.O. Box 19
MUSCAT
Tel: 722225, 745601
Telex: 3289 BROKER MB MUSCAT
P

SALES & SUPPORT OFFICES

Arranged Alphabetically by Country

Suhail & Saud Bahwan
P.O. Box 169
MUSCAT
Tel: 734 201-3
Telex: 3274 BAHWAN MB

PAKISTAN

Mushko & Company Ltd.
1-B, Street 43
Sector F-8/1
ISLAMABAD
Tel: 26875
Cable: FEMUS Rawalpindi
A,E,M

Mushko & Company Ltd.
Oosman Chambers
Abdullah Haroon Road
KARACHI 0302
Tel: 511027, 512927
Telex: 2894 MUSKO PK
Cable: COOPERATOR Karachi
*A,E,M,P**

PANAMA

Electrónico Balboa, S.A.
Calle Samuel Lewis, Ed. Alfa
Apartado 4929
PANAMA 5
Tel: 64-2700
Telex: 3483 ELECTRON PG
A,CM,E,M,P
Foto Internacional, S.A.
Colon Free Zone
Apartado 2068
COLON 3
Tel: 45-2333
Telex: 8626 IMPORT PG
P

PERU

Cía Electro Médica S.A.
Los Flamencos 145, San Isidro
Casilla 1030
LIMA 1
Tel: 41-4325, 41-3703
Telex: Pub. Booth 25306
A,CM,E,M,P

PHILIPPINES

The Online Advanced Systems Corporation
Rico House, Amorsolo Cor. Herrera Street
Legaspi Village, Makati
P.O. Box 1510
Metro ***MANILA***
Tel: 85-35-81, 85-34-91, 85-32-21
Telex: 3274 ONLINE
A,CH,CS,E,M
Electronic Specialists and Proponents Inc.
690-B Epifanio de los Santos Avenue
Cubao, ***QUEZON CITY***
P.O. Box 2649 Manila
Tel: 98-96-81, 98-96-82, 98-96-83
Telex: 40018, 42000 ITT GLOBE MACKAY BOOTH
P

PORTUGAL

Mundinter
Intercambio Mundial de Comércio S.a.r.l
P.O. Box 2761
Av. Antonio Augusto de Aguiar 138
P-LISBON
Tel: (19) 53-21-31, 53-21-37
Telex: 16691 munter p
M

Soquimica
Av. da Liberdade, 220-2
1298 ***LISBON*** *Codex*
Tel: 56 21 81/2/3
Telex: 13316 SABASA P
Telectra-Empresa Técnica de Equipmentos Eléctricos S.a.r.l.
Rua Rodrigo da Fonseca 103
P.O. Box 2531
P-LISBON *1*
Tel: (19) 68-60-72
Telex: 12598
CH,CS,E,P

PUERTO RICO

Hewlett-Packard Puerto Rico
P.O. Box 4407
CAROLINA, Puerto Rico 00628
Calle 272 Edificio 203
Urb. Country Club
RIO PIEDRAS, Puerto Rico 00924
Tel: (809) 762-7255
A,CH,CS

QATAR

Nasser Trading & Contracting
P.O. Box 1563
DOHA
Tel: 22170, 23539
Telex: 4439 NASSER DH
M
Computearbia
P.O. Box 2750
DOHA
Tel: 883555
Telex: 4806 CHPARB
P

Eastern Technical Services
P.O. Box 4747
DOHA
Tel: 329 993
Telex: 4156 EASTEC DH

SAUDI ARABIA

Modern Electronic Establishment
Hewlett-Packard Division
P.O. Box 281
Thuobah
AL-KHOBAR
Tel: 864-46 78
Telex: 671 106 HPMEEK SJ
Cable: ELECTA AL-KHOBAR
CH,CS,E,M,P
Modern Electronic Establishment
Hewlett-Packard Division
P.O. Box 1228
Redec Plaza, 6th Floor
JEDDAH
Tel: 644 38 48
Telex: 402712 FARNAS SJ
Cable: ELECTA JEDDAH
CH,CS,E,M,P
Modern Electronic Establishment
Hewlett Packard Division
P.O. Box 2728
RIYADH
Tel: 491-97 15, 491-63 87
Telex: 202049 MEERYD SJ
CH,CS,E,M,P

SCOTLAND

Hewlett-Packard Ltd.
Royal Bank Buildings
Swan Street
BRECHIN, Angus, Scotland
Tel: (03562) 3101-2
CH
Hewlett-Packard Ltd.
SOUTH QUEENSFERRY
West Lothian, EH30 9GT
GB-Scotland
Tel: (031) 3311188
Telex: 72682
A,CH,CM,CS,E,M

SINGAPORE

Hewlett-Packard Singapore (Pty.) Ltd.
P.O. Box 58 Alexandra Post Office
SINGAPORE, 9115
6th Floor, Inchcape House
450-452 Alexandra Road
SINGAPORE 0511
Tel: 631788
Telex: HPSGSO RS 34209
Cable: HEWPACK, Singapore
A,CH,CS,E,MS,P
Dynamar International Ltd.
Unit 05-11 Block 6
Kolam Ayer Industrial Estate
SINGAPORE *1334*
Tel: 747-6188
Telex: RS 26283
CM

SOUTH AFRICA

Hewlett-Packard So Africa (Pty.) Ltd.
P.O. Box 120
Howard Place
Pine Park Center, Forest Drive, Pinelands
CAPE PROVINCE 7405
Tel: 53-7954
Telex: 57-20006
A,CH,CM,E,MS,P

Hewlett-Packard So Africa (Pty.) Ltd.
P.O. Box 37099
92 Overport Drive
DURBAN 4067
Tel: 28-4178, 28-4179, 28-4110
Telex: 6-22954
CH,CM

Hewlett-Packard So Africa (Pty.) Ltd.
6 Linton Arcade
511 Cape Road
Linton Grange
PORT ELIZABETH 6001
Tel: 041-302148
CH

Hewlett-Packard So Africa (Pty.) Ltd.
P.O. Box 33345
Glenstantia 0010 **TRANSVAAL**
1st Floor East
Constantia Park Ridge Shopping Centre
Constantia Park
PRETORIA
Tel: 982043
Telex: 32163
CH,E

Hewlett-Packard So Africa (Pty.) Ltd.
Private Bag Wendywood
SANDTON 2144
Tel: 802-5111, 802-5125
Telex: 4-20877
Cable: HEWPACK Johannesburg
A,CH,CM,CS,E,MS,P

SPAIN

Hewlett-Packard Española S.A.
c/Entenza, 321
E-BARCELONA 29
Tel: (3) 322-24-51, 321-73-54
Telex: 52603 hpbee
A,CH,CS,E,MS,P

Hewlett-Packard Española S.A.
c/San Vicente S/N
Edificio Albia II,7 B
E-BILBAO 1
Tel: (4) 23-8306, (4) 23-8206
A,CH,E,MS

Hewlett-Packard Española S.A.
Calle Jerez 3
E-MADRID 16
Tel: (1) 458-2600
Telex: 23515 hpe
A,CM,E

Hewlett-Packard Española S.A.
c/o Costa Brava 13
Colonia Mirasierra
E-MADRID 34
Tel: (1) 734-8061, (1) 734-1162
CH,CS,M

Hewlett-Packard Española S.A.
Av Ramón y Cajal 1-9
Edificio Sevilla 1,
E-SEVILLA 5
Tel: 64-44-54, 64-44-58
Telex: 72933
A,CS,MS,P

Hewlett-Packard Española S.A.
C/Ramon Gordillo, 1 (Entlo.3)
E-VALENCIA 10
Tel: 361-1354, 361-1358
CH,P

SWEDEN

Hewlett-Packard Sverige AB
Sunnanvagen 14K
S-22226 **LUND**
Tel: (046) 13-69-79
Telex: (854) 17886 (via SPÅNGA office)
CH

Hewlett-Packard Sverige AB
Vastra Vintergatan 9
S-70344 **OREBRO**
Tel: (19) 10-48-80
Telex: (854) 17886 (via SPÅNGA office)
CH

Hewlett-Packard Sverige AB
Skalholtsgatan 9, Kista
Box 19
S-16393 **SPÅNGA**
Tel: (08) 750-2000
Telex: (854) 17886
A,CH,CM,CS,E,MS,P

Hewlett-Packard Sverige AB
Frötallisgatan 30
S-42132 **VÄSTRA-FRÖLUNDA**
Tel: (031) 49-09-50
Telex: (854) 17886 (via SPÅNGA office)
CH,E,P

SWITZERLAND

Hewlett-Packard (Schweiz) AG
Clarastrasse 12
CH-4058 **BASLE**
Tel: (61) 33-59-20
A

Hewlett-Packard (Schweiz) AG
Bahnhoheweg 44
CH-3018 **BERN**
Tel: (031) 56-24-22
CH

Hewlett-Packard (Schweiz) AG
47 Avenue Blanc
CH-1202 **GENEVA**
Tel: (022) 32-48-00
CH,CM,CS

Hewlett-Packard (Schweiz) AG
19 Chemin Château Bloc
CH-1219 **LE LIGNON**-Geneva
Tel: (022) 96-03-22
Telex: 27333 hpag ch
Cable: HEWPACKAG Geneva
A,E,MS,P

Hewlett-Packard (Schweiz) AG
Allmend 2
CH-8967 **WIDEN**
Tel: (57) 31 21 11
Telex: 53933 hpag ch
Cable: HPAG CH
A,CH,CM,CS,E,MS,P

SYRIA

General Electronic Inc.
Nuri Basha
P.O. Box 5781
DAMASCUS
Tel: 33-24-87
Telex: 11216 ITIKAL SY
Cable: ELECTROBOR DAMASCUS
E

Middle East Electronics
Place Azmé
Boite Postale 2308
DAMASCUS
Tel: 334592
Telex: 11304 SATACO SY
M,P

TAIWAN

Hewlett-Packard Far East Ltd.
Kaohsiung Office
2/F 68-2, Chung Cheng 3rd Road
KAOHSIUNG
Tel: 241-2318, 261-3253
CH,CS,E

Hewlett-Packard Far East Ltd.
Taiwan Branch
5th Floor
205 Tun Hwa North Road
TAIPEI
Tel:(02) 751-0404
Cable:HEWPACK Taipei
A,CH,CM,CS,E,M,P

Ing Lih Trading Co.
3rd Floor, 7 Jen-Ai Road, Sec. 2
TAIPEI 100
Tel: (02) 3948191
Cable: INGLIH TAIPEI
A

THAILAND

Unimesa
30 Patpong Ave., Suriwong
BANGKOK *5*
Tel: 234 091, 234 092
Telex: 84439 Simonco TH
Cable: UNIMESA Bangkok
A,CH,CS,E,M

Bangkok Business Equipment Ltd.
5/5-6 Dejo Road
BANGKOK
Tel: 234-8670, 234-8671
Telex: 87669-BEQUIPT TH
Cable: BUSIQUIPT Bangkok
P

TRINIDAD & TOBAGO

Caribbean Telecoms Ltd.
50/A Jerningham Avenue
P.O. Box 732
PORT-OF-SPAIN
Tel: 62-44213, 62-44214
Telex: 235,272 HUGCO WG
A,CM,E,M,P

TUNISIA

Tunisie Electronique
31 Avenue de la Liberte
TUNIS
Tel: 280-144
E,P

Corema
1 ter. Av. de Carthage
TUNIS
Tel: 253-821
Telex: 12319 CABAM TN
M

TURKEY

Teknim Company Ltd.
Iran Caddesi No. 7
Kavaklidere, ***ANKARA***
Tel: 275800
Telex: 42155 TKNM TR
E

E.M.A.
Medina Eldem Sokak No.41/6
Yuksel Caddesi
ANKARA
Tel: 175 622
M

UNITED ARAB EMIRATES

Emitac Ltd.
P.O. Box 1641
SHARJAH
Tel: 354121, 354123
Telex: 68136 Emitac Sh
CH,CS,E,M,P

UNITED KINGDOM

see: GREAT BRITAIN

NORTHERN IRELAND

SCOTLAND

UNITED STATES

Alabama

Hewlett-Packard Co.
700 Century Park South
Suite 128
BIRMINGHAM, AL 35226
Tel: (205) 822-6802
CH,MP

Hewlett-Packard Co.
P.O. Box 4207
8290 Whitesburg Drive, S.E.
HUNTSVILLE, AL 35802
Tel: (205) 881-4591
CH,CM,CS,E,M*

Alaska

Hewlett-Packard Co.
1577 "C" Street, Suite 252
ANCHORAGE, AK 99501
Tel: (907) 276-5709
CH*

Arizona

Hewlett-Packard Co.
2336 East Magnolia Street
PHOENIX, AZ 85034
Tel: (602) 273-8000
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
2424 East Aragon Road
TUCSON, AZ 85706
Tel: (602) 889-4631
CH,E,MS**

Arkansas

Hewlett-Packard Co.
P.O. Box 5646
Brady Station
LITTLE ROCK, AR 72215
111 N. Filmore
LITTLE ROCK, AR 72205
Tel: (501) 664-8773, 376-1844
MS

California

Hewlett-Packard Co.
99 South Hill Dr.
BRISBANE, CA 94005
Tel: (415) 330-2500
CH,CS

Hewlett-Packard Co.
7621 Canoga Avenue
CANOGA PARK, CA 91304
Tel: (213) 702-8300
A,CH,CS,E,P

Hewlett-Packard Co.
5060 Clinton Avenue
FRESNO, CA 93727
Tel: (209) 252-9652
MS

Hewlett-Packard Co.
P.O. Box 4230
1430 East Orangethorpe
FULLERTON, CA 92631
Tel: (714) 870-1000
CH,CM,CS,E,MP

Hewlett-Packard Co.
320 S. Kellogg, Suite B
GOLETA, CA 93117
Tel: (805) 967-3405
CH

Hewlett-Packard Co.
5400 W. Rosecrans Boulevard
LAWNDALE, CA 90260
P.O. Box 92105
LOS ANGELES, CA 90009
Tel: (213) 970-7500
Telex: 910-325-6608
CH,CM,CS,MP

Hewlett-Packard Co.
3200 Hillview Avenue
PALO ALTO, CA 94304
Tel: (415) 857-8000
CH,CS,E

Hewlett-Packard Co.
P.O. Box 15976 (95813)
4244 So. Market Court, Suite A
SACRAMENTO, CA 95834
Tel: (916) 929-7222
A*,CH,CS,E,MS

Hewlett-Packard Co.
9606 Aero Drive
P.O. Box 23333
SAN DIEGO, CA 92123
Tel: (714) 279-3200
CH,CM,CS,E,MP

Hewlett-Packard Co.
2305 Camino Ramon "C"
SAN RAMON, CA 94583
Tel: (415) 838-5900
CH,CS

Hewlett-Packard Co.
P.O. Box 4230
Fullerton, CA 92631
363 Brookhollow Drive
SANTA ANA, CA 92705
Tel: (714) 641-0977
A,CH,CM,CS,MP

Hewlett-Packard Co.
Suite A
5553 Hollister
SANTA BARBARA, CA 93111
Tel: (805) 964-3390

Hewlett-Packard Co.
3003 Scott Boulevard
SANTA CLARA, CA 95050
Tel: (408) 988-7000
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
5703 Corsa Avenue
WESTLAKE VILLAGE, CA 91362
Tel: (213) 706-6800
E*,CH*,CS*

Colorado

Hewlett-Packard Co.
24 Inverness Place, East
ENGLEWOOD, CO 80112
Tel: (303) 771-3455
Telex: 910-935-0785
A,CH,CM,CS,E,MS

Connecticut

Hewlett-Packard Co.
47 Barnes Industrial Road South
P.O. Box 5007
WALLINGFORD, CT 06492
Tel: (203) 265-7801
A,CH,CM,CS,E,MS

Florida

Hewlett-Packard Co.
P.O. Box 24210 (33307)
2901 N.W. 62nd Street
FORT LAUDERDALE, FL 33307
Tel: (305) 973-2600
CH,CS,E,MP

Hewlett-Packard Co.
4080 Woodcock Drive, #132
Brownett Building
JACKSONVILLE, FL 32207
Tel: (904) 398-0663
C*,E*,MS**

Hewlett-Packard Co.
1101 W. Hibiscus Ave., Suite E210
MELBOURNE, FL 32901
Tel: (305) 729-0704
E*

Hewlett-Packard Co.
P.O. Box 13910 (32859)
6177 Lake Ellenor Drive
ORLANDO, FL 32809
Tel: (305) 859-2900
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
6425 N. Pensacola Blvd.
Suite 4, Building 1
P.O. Box 12826
PENSACOLA, FL 32575
Tel: (904) 476-8422
A,MS

Hewlett-Packard Co.
5750B N. Hoover Blvd., Suite 123
TAMPA, FL 33614
Tel: (813) 884-3282
A*,CH,CM,CS,E*,M*

Georgia

Hewlett-Packard Co.
P.O. Box 105005
ATLANTA, GA 30348
2000 South Park Place
ATLANTA, GA 30339
Tel: (404) 955-1500
Telex: 810-766-4890
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
P.O. Box 816 (80903)
2531 Center West Parkway
Suite 110
AUGUSTA, GA 30904
Tel: (404) 736-0592
MS

Hewlett-Packard Co.
200-E Montgomery Cross Rds.
SAVANNAH, GA 31401
Tel:(912) 925-5358
CH**

Hewlett-Packard Co.
P.O. Box 2103
WARNER ROBINS, GA 31099
1172 N. Davis Drive
WARNER ROBINS, GA 31093
Tel: (912) 923-8831
E

Hawaii

Hewlett-Packard Co.
Kawaiahao Plaza, Suite 190
567 South King Street
HONOLULU, HI 96813
Tel: (808) 526-1555
A,CH,E,MS

Illinois

Hewlett-Packard Co.
211 Prospect Road, Suite C
BLOOMINGTON, IL 61701
Tel: (309) 662-9411
CH,MS**

Hewlett-Packard Co.
1100 31st Street, Suite 100
DOWNERS GROVE, IL 60515
Tel: (312) 960-5760
CH,CS

Hewlett-Packard Co.
5201 Tollview Drive
ROLLING MEADOWS, IL 60008
Tel: (312) 255-9800
A,CH,CM,CS,E,MP

Indiana

Hewlett-Packard Co.
P.O. Box 50807
7301 No. Shadeland Avenue
INDIANAPOLIS, IN 46250
Tel: (317) 842-1000
A,CH,CM,CS,E,MS

Iowa

Hewlett-Packard Co.
1776 22nd Street, Suite 1
WEST DES MOINES, IA 50265
Tel: (515) 224-1435
CH,MS**

Hewlett-Packard Co.
2415 Heinz Road
IOWA CITY, IA 52240
Tel: (319) 351-1020
CH,E*,MS

Kansas

Hewlett-Packard Co.
1644 S. Rock Road
WICHITA, KA 67207
Tel: (316) 684-8491
CH

Kentucky

Hewlett-Packard Co.
10300 Linn Station Road
Suite 100
LOUISVILLE, KY 40223
Tel: (502) 426-0100
A,CH,CS,MS

Louisiana

Hewlett-Packard Co.
8126 Calais Bldg.
BATON ROUGE, LA 70806
Tel: (504) 467-4100
A**,CH**

Hewlett-Packard Co.
P.O. Box 1449
KENNER, LA 70062
160 James Drive East
DESTAHAN, LA 70047
Tel: (504) 467-4100
A,CH,CS,E,MS

Maryland

Hewlett-Packard Co.
7121 Standard Drive
HANOVER, MD 21076
Tel: (301) 796-7700
Telex: 710-862-1943
Eff. Dec. 1, 1982
3701 Koppers St.
BALTIMORE, MD 21227
Tel: (301) 644-5800
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
2 Choke Cherry Road
ROCKVILLE, MD 20850
Tel: (301) 948-6370
A,CH,CM,CS,E,MP

Massachusetts

Hewlett-Packard Co.
32 Hartwell Avenue
LEXINGTON, MA 02173
Tel: (617) 861-8960
A,CH,CM,CS,E,MP

Michigan

Hewlett-Packard Co.
23855 Research Drive
FARMINGTON HILLS, MI 48024
Tel: (313) 476-6400
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
4326 Cascade Road S.E.
GRAND RAPIDS, MI 49506
Tel: (616) 957-1970
CH,CS,MS

Hewlett-Packard Co.
1771 W. Big Beaver Road
TROY, MI 48084
Tel: (313) 643-6474
CH,CS

Minnesota

Hewlett-Packard Co.
2025 W. Larpenteur Ave.
ST. PAUL, MN 55113
Tel: (612) 644-1100
A,CH,CM,CS,E,MP

Mississippi

Hewlett-Packard Co.
P.O. Box 5028
1675 Lakeland Drive
JACKSON, MS 39216
Tel: (601) 982-9363
MS

Missouri

Hewlett-Packard Co.
11131 Colorado Avenue
KANSAS CITY, MO 64137
Tel: (816) 763-8000
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
P.O. Box 27307
1024 Executive Parkway
ST. LOUIS, MO 63141
Tel: (314) 878-0200
A,CH,CS,E,MP
Effective September 1982:
13001 Hollenberg Drive
BRIDGETON, MO 63044

Nebraska

Hewlett-Packard
7101 Mercy Road
Suite 101, IBX Building
OMAHA, NE 68106
Tel: (402) 392-0948
CM,MS

Nevada

Hewlett-Packard Co.
Suite D-130
5030 Paradise Blvd.
LAS VEGAS, NV 89119
Tel: (702) 736-6610
MS**

New Jersey

Hewlett-Packard Co.
W120 Century Road
PARAMUS, NJ 07652
Tel: (201) 265-5000
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
60 New England Av. West
PISCATAWAY, NJ 08854
Tel: (201) 981-1199
A,CH,CM,CS,E

New Mexico

Hewlett-Packard Co.
P.O. Box 11634
ALBUQUERQUE, NM 87112
11300 Lomas Blvd.,N.E.
ALBUQUERQUE, NM 87123
Tel: (505) 292-1330
Telex: 910-989-1185
CH,CS,E,MS

New York

Hewlett-Packard Co.
5 Computer Drive South
ALBANY, NY 12205
Tel: (518) 458-1550
Telex: 710-444-4691
A,CH,E,MS

Hewlett-Packard Co.
P.O. Box 297
9600 Main Street
CLARENCE, NY 14031
Tel: (716) 759-8621
Telex: 710-523-1893
CH

Hewlett-Packard Co.
200 Cross Keys Office
FAIRPORT, NY 14450
Tel: (716) 223-9950
Telex: 510-253-0092
CH,CM,CS,E,MS

Hewlett-Packard Co.
7641 Henry Clay Blvd.
LIVERPOOL, NY 13088
Tel: (315) 451-1820
A,CH,CM,E,MS

Hewlett-Packard Co.
No. 1 Pennsylvania Plaza
55th Floor
34th Street & 8th Avenue
NEW YORK, NY 10119
Tel: (212) 971-0800
CH,CS,E*,M*

SALES & SUPPORT OFFICES

Arranged Alphabetically by Country

Hewlett-Packard Co.
250 Westchester Avenue
WHITE PLAINS, NY 10604
CM,CH,CS,E

Hewlett-Packard Co.
3 Crossways Park West
WOODBURY, NY 11797
Tel: (516) 921-0300
Telex: 510-221-2183
A,CH,CM,CS,E,MS

North Carolina

Hewlett-Packard Co.
4915 Water's Edge Drive
Suite 160
RALEIGH, NC 27606
Tel: (919) 851-3021
C,M

Hewlett-Packard Co.
P.O. Box 26500
5605 Roanne Way
GREENSBORO, NC 27450
Tel: (919) 852-1800
A,CH,CM,CS,E,MS

Ohio

Hewlett-Packard Co.
9920 Carver Road
CINCINNATI, OH 45242
Tel: (513) 891-9870
CH,CS,MS

Hewlett-Packard Co.
16500 Sprague Road
CLEVELAND, OH 44130
Tel: (216) 243-7300
Telex: 810-423-9430
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
962 Crupper Ave.
COLUMBUS, OH 43229
Tel: (614) 436-1041
CH,CM,CS,E*

Hewlett-Packard Co.
P.O. Box 280
330 Progress Rd.
DAYTON, OH 45449
Tel: (513) 859-8202
A,CH,CM,E*,MS

Oklahoma

Hewlett-Packard Co.
P.O. Box 32008
Oklahoma City, OK 73123
1503 W. Gore Blvd., Suite #2
LAWTON, OK 73505
Tel: (405) 248-4248
C

Hewlett-Packard Co.
P.O. Box 32008
OKLAHOMA CITY, OK 73123
304 N. Meridian Avenue, Suite A
OKLAHOMA CITY, OK 73107
Tel: (405) 946-9499
A*,CH,E*,MS

Hewlett-Packard Co.
Suite 121
9920 E. 42nd Street
TULSA, OK 74145
Tel: (918) 665-3300
A**,CH,CS,M*

Oregon

Hewlett-Packard Co.
1500 Valley River Drive
Suite 330
EUGENE, OR 97401
Tel: (503) 683-8075
C

Hewlett-Packard Co.
9255 S. W. Pioneer Court
WILSONVILLE, OR 97070
Tel: (503) 682-8000
A,CH,CS,E*,MS

Pennsylvania

Hewlett-Packard Co.
1021 8th Avenue
King of Prussia Industrial Park
KING OF PRUSSIA, PA 19406
Tel: (215) 265-7000
Telex: 510-660-2670
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
111 Zeta Drive
PITTSBURGH, PA 15238
Tel: (412) 782-0400
A,CH,CS,E,MP

South Carolina

Hewlett-Packard Co.
P.O. Box 21708
Brookside Park, Suite 122
1 Harbison Way
COLUMBIA, SC 29210
Tel: (803) 732-0400
CH,E,MS

Hewlett-Packard Co.
Koger Executive Center
Chesterfield Bldg., Suite 124
GREENVILLE, SC 29615
Tel: (803) 748-5601
C

Tennessee

Hewlett-Packard Co.
P.O. Box 22490
224 Peters Road
Suite 102
KNOXVILLE, TN 37922
Tel: (615) 691-2371
A*,CH,MS

Hewlett-Packard Co.
3070 Directors Row
MEMPHIS, TN 38131
Tel: (901) 346-8370
A,CH,MS

Hewlett-Packard Co.
230 Great Circle Road
Suite 216
NASHVILLE, TN 32228
Tel: (615) 255-1271
MS**

Texas

Hewlett-Packard Co.
Suite 310W
7800 Shoalcreek Blvd.
AUSTIN, TX 78757
Tel: (512) 459-3143
E

Hewlett-Packard Co.
Suite C-110
4171 North Mesa
EL PASO, TX 79902
Tel: (915) 533-3555, 533-4489
CH,E*,MS**

Hewlett-Packard Co.
5020 Mark IV Parkway
FORT WORTH, TX 76106
Tel: (817) 625-6361
CH,CS*

Hewlett-Packard Co.
P.O. Box 42816
HOUSTON, TX 77042
10535 Harwin Street
HOUSTON, TX 77036
Tel: (713) 776-6400
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
3309 67th Street
Suite 24
LUBBOCK, TX 79413
Tel: (806) 799-4472
M

Hewlett-Packard Co.
417 Nolana Gardens, Suite C
P.O. Box 2256
McALLEN, TX 78501
Tel: (512) 781-3226
CH,CS

Hewlett-Packard Co.
P.O. Box 1270
RICHARDSON, TX 75080
930 E. Campbell Rd.
RICHARDSON, TX 75081
Tel: (214) 231-6101
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
P.O. Box 32993
SAN ANTONIO, TX 78216
1020 Central Parkway South
SAN ANTONIO, TX 78232
Tel: (512) 494-9336
CH,CS,E,MS

Utah

Hewlett-Packard Co.
P.O. Box 26626
3530 W. 2100 South
SALT LAKE CITY, UT 84119
Tel: (801) 974-1700
A,CH,CS,E,MS

Virginia

Hewlett-Packard Co.
P.O. Box 9669
2914 Hungary Spring Road
RICHMOND, VA 23228
Tel: (804) 285-3431
A,CH,CS,E,MS

Hewlett-Packard Co.
3106 Peters Creek Road, N.W.
ROANOKE, VA 24019
Tel: (703) 563-2205
CH,E**

Hewlett-Packard Co.
5700 Thurston Avenue
Suite 111
VIRGINIA BEACH, VA 23455
Tel: (804) 460-2471
CH,MS

Washington

Hewlett-Packard Co.
15815 S.E. 37th Street
BELLEVUE, WA 98006
Tel: (206) 643-4000
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
Suite A
708 North Argonne Road
SPOKANE, WA 99206
Tel: (509) 922-7000
CH,CS

West Virginia

Hewlett-Packard Co.
4604 MacCorkle Ave., S.E.
CHARLESTON, WV 25304-4297
Tel: (304) 925-0492
A,MS

Wisconsin

Hewlett-Packard Co.
150 S. Sunny Slope Road
BROOKFIELD, WI 53005
Tel: (414) 784-8800
A,CH,CS,E*,MP

URUGUAY

Pablo Ferrando S.A.C. e L.
Avenida Italia 2877
Casilla de Correo 370
MONTEVIDEO
Tel: 80-2586
Telex: Public Booth 901
A,CM,E,M

Guillermo Kraft del Uruguay S.A.
Av. Lib. Brig. Gral. Lavalleja 2083
MONTEVIDEO
Tel: 234588, 234808, 208830
Telex: 22030 ACTOUR UY
P

VENEZUELA

Hewlett-Packard de Venezuela C.A.
3A Transversal Los Ruices Norte
Edificio Segre
Apartado 50933
CARACAS 1071
Tel: 239-4133
Telex: 25146 HEWPACK
A,CH,CS,E,MS,P

Colimodio S.A.
Este 2 - Sur 21 No. 148
Apartado 1053
***CARACAS** 1010*
Tel: 571-3511
Telex: 21529 COLMODIO
M

ZIMBABWE

Field Technical Sales
45 Kelvin Road, North
P.B. 3458
SALISBURY
Tel: 705 231
Telex: 4-122 RH
C,E,M,P

Headquarters offices

If there is no sales office listed for your area, contact one of these headquarters offices.

NORTH/CENTRAL AFRICA

Hewlett-Packard S.A.
7 Rue du Bois-du-Lan
CH-1217 **MEYRIN 2**, Switzerland
Tel: (022) 98-96-51
Telex: 27835 hpse
Cable: HEWPACKSA Geneve

ASIA

Hewlett-Packard Asia Ltd.
6th Floor, Sun Hung Kai Center
30 Harbor Rd.
G.P.O. Box 795
HONG KONG
Tel: 5-832 3211
Telex: 66678 HEWPA HX
Cable: HEWPACK HONG KONG

CANADA

Hewlett-Packard (Canada) Ltd.
6877 Goreway Drive
MISSISSAUGA, Ontario L4V 1M8
Tel: (416) 678-9430
Telex: 610-492-4246

EASTERN EUROPE

Hewlett-Packard Ges.m.b.h.
Lieblgasse 1
P.O.Box 72
A-1222 **VIENNA**, Austria
Tel: (222) 2365110
Telex: 1 3 4425 HEPA A

NORTHERN EUROPE

Hewlett-Packard S.A.
Uilenstede 475
NL-1183 AG **AMSTELVEEN**
The Netherlands
P.O.Box 999
NL-1180 AZ **AMSTELVEEN**
The Netherlands
Tel: 20 437771

OTHER EUROPE

Hewlett-Packard S.A.
7 Rue du Bois-du-Lan
CH-1217 **MEYRIN 2**, Switzerland
Tel: (022) 98-96-51
Telex: 27835 hpse
Cable: HEWPACKSA Geneve
(Offices in the World Trade Center)

MEDITERRANEAN AND MIDDLE EAST

Hewlett-Packard S.A.
Mediterranean and Middle East Operations
Atrina Centre
32 Kifissias Ave.
Maroussi, **ATHENS**, Greece
Tel: 682 88 11
Telex: 21-6588 HPAT GR
Cable: HEWPACKSA Athens

EASTERN USA

Hewlett-Packard Co.
4 Choke Cherry Road
Rockville, MD 20850
Tel: (301) 258-2000

MIDWESTERN USA

Hewlett-Packard Co.
5201 Tollview Drive
ROLLING MEADOWS, IL 60008
Tel: (312) 255-9800

SOUTHERN USA

Hewlett-Packard Co.
P.O. Box 105005
450 Interstate N. Parkway
ATLANTA, GA 30339
Tel: (404) 955-1500

WESTERN USA

Hewlett-Packard Co.
3939 Lankersim Blvd.
LOS ANGELES, CA 91604
Tel: (213) 877-1282

OTHER INTERNATIONAL AREAS

Hewlett-Packard Co.
Intercontinental Headquarters
3495 Deer Creek Road
PALO ALTO, CA 94304
Tel: (415) 857-1501
Telex: 034-8300
Cable: HEWPACK

15 Aug 1982 5952-6900

MANUAL PART NO. 92068-90002
Printed in U.S.A. January 1983
E0183

HEWLETT-PACKARD COMPANY
Data Systems Division
11000 Wolfe Road
Cupertino, California 95014