RSTS/E
System Manager's Guide

Order No. DEC-11-ORSMC-A-D

Order additional copies as directed on the Software Information page at the back of this document.

digital equipment corporation · maynard. massachusetts

First Printing, October 1974

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

The software described in this document is furnished to the purchaser under a license for use on a single computer system and can be copied (with inclusion of DIGITAL's copyright notice) only for use in such system, except as may otherwise be provided in writing by DIGITAL.

Digital Equipment Corporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by DIGITAL.

The HOW TO OBTAIN SOFTWARE INFORMATION page, located at the back of this document, explains the various services available to DIGITAL software users.

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

The following are trademarks of Digital Equipment Corporation:

CDP	DIGITAL	INDAC	PS/8
COMPUTER LAB	DNC	KA10	QUICKPOINT
COMSYST	EDGRIN	LAB-8	RAD-8
COMTEX	EDUSYSTEM	LAB-8/e	RSTS
DDT	FLIP CHIP	LAB-K	RSX
DEC	FOCAL	OMNIBUS	RTM
DECCOMM	GLC-8	OS/8	RT-11
DECTAPE	IDAC	PDP	SABR
DIBOL	IDACS	PHA	TYPESET 8
			UNIBUS

CONTENTS

Updated by Addendum DEC-11-ORSMC-A-DN1 dated March 1975.

PREFACE

This guide describes the detailed procedures for generating the RSTS/E system code and building a system disk and system library files. Also included are procedures for operating and managing the system, performing privileged system operations and using system programming techniques for Version VØ5C-Ø1 of RSTS/E, the Resource Sharing Time Sharing System/Extended. The contents of this guide are of concern only to the RSTS/E system manager and users whom he designates as privileged. Any other individual should not have access to this document without the consent of the system manager.

NOTE

It is strongly recommended that the system manager read this guide in its entirety before attempting to build and start up RSTS/E.

The following are titles and order numbers of documents useful in the operation and management of the RSTS/E system.

RSTS/E System Installation Notes	DEC-11-ORINB-A-D
RSTS-11 System User's Guide	DEC-11-ORSUA-C-D
RSTS/E System Reliability Test	DEC-11-ORSRB-A-D
PDP-11 ROLLIN Utility Programmer's Manual	DEC-11-OROAA-B-D
BASIC-PLUS Language Manual	DEC-11-ORBPA-C-D
RSTS/E RUNOFF User's Guide	DEC-11-URUNA-A-D
DOS/BATCH Text Editor (EDIT) Programmer's Manual	DEC-11-UEDAA-A-D
RSTS-11 Sort User's Guide	DEC-11-ORSGA-A-D
RSTS-11 Sort User's Guide Addendum	DEC-11-ORSGA-A-DN1
RSTS/E Commercial User's Guide	DEC-11-ORCUA-A-D
RSTS/E 2780 User's Guide	DEC-11-ORJEA-A-D

For a quick reference to a subject in this guide, use the following list.

If you need to know about	See Section
System generation from magtape	2.2.1
System generation from disk cartridge	2.3.1
System generation from DECtape	2.2.2
Configuration question considerations	2.7
Configuration question examples	2.6
Bootstrapping a system disk	5.2.1
Starting time sharing operations	3.6
Terminating time sharing	6.2
Changing system files	3.4.2
Adding more memory to RSTS/E	3.5.3.6
Changing the maximum number of users	3.5.2
Changing the maximum job size allowed	3.5.2
Performing system utility operations	6.3
Determining causes of hardware errors	6.11
Organization of system disk	3.4.1
Building system library files	Ch. 4
Creating user accounts	4.4, 6.4
Creating or changing ASCII text files	4.3
System programming operations	Ch. 7
Reporting system troubles	Back of guide
Creating TTYSET.SPD file	2.7.3, 2.7.4, 4.3.4, 6.7
Enabling DH11 modem service	3.4A
Copying DOS disk files	6.19
Formatting disks	2.3.3

PREFACE TO ADDENDUM

This addendum to the RSTS/E System Manager's Guide, order number DEC-11-ORSMC-A-DN1, describes RSTS/E version VØ5C-Ø1 which supports the RPØ4 disk pack drive and the TU16 magtape drive. The following are the major changes.

Chapter 2 - Hardware Bootstrap procedures are described in Appendix F. Some SYSGEN questions have been rearranged and omitted. DH11 questions have been simplified. Record I/O is no longer optional but is included on all systems. Monitor and BASIC-PLUS patch space is also included automatically on all systems.

Chapter 3 - New information has been added to describe the RPØ4 disk pack drive and the SETKEY initialization option.

Chapter 4 - The RECIO.CTL control file used by the BUILD program no longer exists. The files it created are now created by the BUILD.CTL file.

Chapter 5 - All hardware bootstrap information has been moved to Appendix F.

Chapter 6 - The ERRDIS program contains new options to print errors concerning the RPØ4 disks and the TU16 magtape. The description of the DOSPIP program has been added.

Chapter 7 - Certain errors have been corrected.

Appendix F - All bootstrap information is now in Appendix F. With the new hardware loader, the user can bootstrap a RSTS/E system disk from other than

a non-zero unit. However, the RSTS/E software is not conditioned to run from other than unit Ø.

Changes to the document are denoted by bars in the outside margin of the changed page. An asterisk on the outside margin of a changed page indicates that material was deleted. Where changed or additional information could not fit on a current page, overflow pages were created and annotated with fractional page numbers. An addition, a replacement, or an overflow page containing new information is denoted only by a date in the lower outside corner of the page. A page containing neither a change bar, an asterisk, nor a date is unchanged from the previous edition.

To update the current manual, perform the following steps.

Replace the current Contents and Preface pages with the new Contents and Preface pages.

Replace the current Chapter 2 with the new Chapter 2.

Replace current pages 3-1 through 3-4 with new pages 3-1 through 3-4.1.

Replace the current pages 3-11 through 3-16 with new pages 3-11 through 3-16.1.

Replace the current pages 3-21 and 3-22 with new pages 3-21 and 3-22.

Replace current pages 3-45 and 3-46 with new pages 3-45 and 3-46.

Replace current pages 3-59 and 3-6Ø with new pages 3-59, 3-59.1, 3-59.2, 3-59.3, 3-59.4, and 3-6Ø.

Replace the current Chapter 4 with the new Chapter 4.

Replace current pages 5-1 through 5-4 with new pages 5-1 through 5-4.

Replace current pages 6-69 and 6-7Ø with new pages 6-69 and 6-7Ø.

Replace current page 6-111 with new pages 6-111 through 6-115.

Replace current pages 7-15 through 7-18 with new pages 7-15 through 7-18.

Replace current pages 7-23 through 7-26 with new pages 7-23 through 7-26.

Replace current pages 7-79 and 7-8Ø with new pages 7-79 and 7-8Ø.

Replace current pages 7-87 and 7-88 with new pages 7-87 and 7-88.

Replace current pages 7-97 through 7-1ØØ with new pages 7-97 through 7-1ØØ.

Replace current Appendix C with new Appendix C.

Replace current Appendix D with new Appendix D.

Add new Appendix F.

Replace current HOW TO OBTAIN SOFTWARE INFORMATION page with new version of same.

Place the Reader's Comments page behind the current Reader's Comments page.

Place this Preface behind the current Preface.

CHAPTER 1
RSTS/E SYSTEM STRUCTURE AND SYSTEM MANAGEMENT

RSTS/E (Resource Sharing Time Sharing System/Extended) runs on either a PDP-11/40 or PDP-11/45 computer and allows up to 63 users simultaneous, time shared access to PDP-11 hardware resources and to RSTS/E system software components through either local or remote terminals. A RSTS/E user performs time sharing operations using the full computational and data processing power of the BASIC-PLUS language.

The current version of RSTS/E employs the same file structure and programming language of previous versions but provides the capability to handle more users running larger resident programs faster. The system manager using RSTS/E has increased control over system file structures and the conditions under which time sharing operations are conducted. This chapter introduces the system manager to the hardware and software structures of RSTS/E and provides references to further descriptions of the philosophy and uses of RSTS/E hardware and software.

1.1 SYSTEM HARDWARE

The hardware system building block is either a PDP-11/45 Central Processor Unit (CPU) with optional hardware floating point processor (FPP) and optional MOS memory or a PDP-11/40 CPU with Extended Instruction Set (EIS) and, optionally, Floating Instruction Set (FIS).

Either the BM792-YB Hardware Bootstrap Loader or the MR11-DB Bulk Storage Bootstrap Loader is required on the system. Also, either the Programmable Real Time Clock KW11-P or the Line Time Clock KW11-L is necessary.

In order to provide adequate secondary storage, RSTS/E must include at least two disks from the following types of disks: The

RF11 Controller with up to 8 RS11 disks (platters) and the RK11 Controller with up to 8 RK05 (or RK03) DECpack cartridge disk drives, or at least one disk attached to the RP11-C Controller which can handle up to 8 RP03 Disk pack drives. The optimal disk configuration includes a removable moving head disk which acts as the RSTS/E system disk, and an auxiliary fixed head disk which performs swapping operations. Additional disks are used to augment the capacity of public and private storage space and swapping storage space.

Additional devices such as the following can be used on the RSTS/E system. Terminals may be standard Teletypes[1] or LA30 DECwriters, 2741 type terminals, and VT50, VT05 or VT05B Alphanumeric Displays. The PC11 High Speed Reader/Punch or the PR11 High Speed Reader can be used to accelerate paper tape input/output operations. Up to eight LP11 and/or LS11 Line Printers can be configured in RSTS/E to make hard copy output more efficient. The CR11 punched card reader, CM11 mark-sense card reader or CD11 high speed punched card reader can be used to handle 80-column punched data cards or 40-column mark-sense cards. RSTS/E also supports magnetic tape devices: the TC11/TU56 DECtape Control and Dual DECtape transport or the TM11/TU10 Magtape Control and Transport.

Local terminals can be connected to the PDP-11 computer via the KL11, DL11A, DL11B, DL11C, DL11D, or LC11 line interface devices. Terminals on remote, dial-up lines can be connected via either the DC11 or DL11E remote line interface devices. Both local and remote lines can be connected through the DH11 multiplexer and the DM11B modem control multiplexer.

The impact of these hardware elements on system performance and configuration requirements is described in Section 2.7 of this guide.

1.2 SYSTEM SOFTWARE

RSTS/E system software exists as PDP-11 assembly language code and as BASIC-PLUS language code. The assembly language code is tailored at system generation time according to the hardware configuration on which the system runs and to the software features which the system manager chooses to include in his RSTS/E system. Once the

[1]Teletype is a registered trademark of the Teletype Corporation.

system is generated, this code is frozen and alterable only by use of patching or by generating new system code. The BASIC-PLUS language code exists in a system library of programs which are executed by the system itself or by individual users.

1.2.1 Assembly Language Code

The RSTS/E assembly language code, when generated, is stored on a disk as a core image library (CIL). A core image library, as it resides on disk or in memory, is pure code and immediately executable by the PDP-11 computer. Assembly language elements of RSTS/E comprise many distinct modules which are either permanently resident in memory, temporarily resident, or disk resident. Permanently resident elements are the following:

a) interrupt and trap vectors
b) small and large system buffers
c) system information and data tables
d) disk and I/O device drivers
e) file processor modules
f) BASIC-PLUS text editor and analyzer
g) incremental compiler
h) BASIC-PLUS Run Time System

Temporarily memory resident elements are the following:

a) system initialization code

Disk resident elements (overlay code) include the following:

a) file processor modules
b) DECtape and magtape processor
c) infrequently used utility routines

The system initialization code is bootstrapped into memory when the system starts. The code is executed and performs a series of consistency checks to ensure the integrity of the software before any operations are allowed. When checking is completed, the system manager has available many initialization options which are made possible by the initialization code as described in Chapter 3 of this guide. Once time sharing operations start, the initialization code

is overlaid by the resident RSTS/E monitor and by user programs. As time sharing operations proceed, infrequently used overlay code is called as needed from its storage area on the disk and loaded into memory.

1.2.2 BASIC-PLUS Language Code

A system library of programs is produced by the system manager from BASIC-PLUS source code supplied by DEC and is stored on disk as intermediate (compiled) code. System library programs exercise wide control over the system since they use privileged program status and SYS system function calls. The system assembly language code calls upon some of these programs to augment its own functions. The system manager can use some of them to monitor and regulate system usage. Ordinary users can run them to perform common utility functions.

1.3 DISK FILE STRUCTURE

Access to all executable code and to system and user data on the RSTS/E system is accomplished through a logical structure of files. The file structure is flexible enough so that it can control and access any type of information. The file structure design is based on the need to control and access data and code on disk.

The logical disk structure is divided into two types: public and private. The public disk structure consists of a system disk and additional public disk packs or disk cartridges. All public disks must be physically on-line and logically mounted whenever the system is running and must be accessible to all users during time sharing operations.

The system disk contains the assembly language code of the RSTS/E system and BASIC-PLUS intermediate code of the system library of programs. The system disk may also be used for storage of active user jobs which are temporatily swapped out of memory. If the system disk is a moving head device, an auxiliary fixed head disk can be used as the swapping device. In such cases, the swapping device is a logical extension of the system disk and can be configured to contain, in addition to the swapping files, other frequently used system files to improve speed of access.

Remaining space on the system disk and all space on other public disks is available for general storage of user programs and data files.

Any remaining disk drives in the RSTS/E disk structure can be devoted to private disk packs or disk cartridges. Private disks can be logically mounted and dismounted and interchanged as needed during time sharing operations. A private disk provides a means to restrict disk storage to a defined set of users. The file structure on a disk, whether it is designated public or private, is the same.

Control of and access to files in the RSTS/E system is accomplished by two structures called a Master File Directory and a User File Directory. A Master File Directory, or MFD, exists on each disk initialized for use on the RSTS/E system. The MFD is treated as an account on the disk, has a project-programmer number [1,1], and catalogs other accounts on the disk. The MFD on the system disk is a special case, since it maintains a catalog of the accounts which can be used to log into the system. MFD accounts on other disks contain entries of accounts which can create files on that disk. Any user gains access to any file on a private disk if the protection code of the file permits. However, only those users whose accounts are entered in the MFD of the private disk can create files on the disk.

The User File Directory, or UFD, exists on a disk for an account under which files are created. The UFD enables the system to access a user's file through the MFD on the device. A UFD for an account on a disk is not created until a file is created under that account on the disk.

1.4 SYSTEM OPERATION CONCEPTS

Immediately after logging into the system, a user's terminal is in edit mode (BASIC-PLUS command level) and is returned to edit mode when any program execution is completed or whenever a CTRL/C is typed at the terminal. In edit mode, the system examines each ASCII text line entered by the user and determines whether that line is a system command, an immediate mode statement, or a program statement. System commands are executed immediately after being entered as described in Chapter 2 of the RSTS-11 System User's Guide. Immediate mode statements are first translated into an intermediate code, which is placed in the user's job area, and are executed immediately by the Run Time System. (Immediate mode operations are described in

Chapter 4 of the BASIC-PLUS Language Manual.) Program statements (lines of ASCII text preceded by line numbers) are stored in their ASCII form in a temporary disk file named TEMPnn.TMP stored under the user's account. Each program statement is also compiled into its intermediate code representation, which is placed in the user's area of memory.

The user job area is initialized at log in time and set to a size of 2K words (K = 1024). The job area can grow in increments of 1K words to a maximum size set by the system manager at the start of time sharing operations. (Refer to the description of SWAP MAX in Chapter 3.) Intermediate code created in the user's job area upon entry of program statements in edit mode is not executed automatically. The related program statements being created can be changed. A copy of the intermediate code of the program can be transferred to disk storage (as a file with a BAC filename extension) or to an external storage medium.

A user changes from edit mode to run mode when he types the RUN system command or the CHAIN immediate mode statement. In run mode, the Run Time System interpretively executes the intermediate code stored in the user's job area. Following program execution, the user's terminal is returned to edit mode, signalled by printing of the READY message. The user can interrupt the Run Time System by typing CTRL/C, which also returns the user's terminal to edit mode.

The RSTS/E system allows user jobs to run (in either edit mode or run mode) one at a time. A user job runs until it either enters an I/O wait state or exhausts the time quantum which either the system or the system manager has assigned to it. At the point when the currently running job ceases to run, the scheduler finds the next job that is ready to run and begins running that job. Meanwhile, the interrupt-driven I/O device handlers are processing requested data transfers. Upon completion of a transfer, the scheduler marks the job that requested the transfer as ready to run again and starts it from the point at which execution ceased.

RSTS/E attempts to keep as many jobs in memory as possible. When more memory is required to run a job than is available, the system temporarily moves some jobs out of memory and stores them in one of many files called SWAPØ.SYS, SWAP1.SYS, SWAP2.SYS, and SWAP3.SYS.

This operation is called swapping. When it is again their turn to run, the jobs in one of the swapping files are swapped back into memory. Jobs waiting for keyboard input and jobs waiting for device I/O completion are most likely stored in the swapping files, while jobs currently running or involved in disk or magtape data transfers are necessarily in memory.

As the system processes each job, it maintains accounting information in memory concerning that job. When the job is logged off the system, this information is used to update the accounting information stored on the disk for that account.

1.5 SYSTEM GENERATION

RSTS/E software is distributed on either DECtape, 7-track or 9-track magtape or disk cartridge (DECpack). The system is generated by the system manager performing two basic steps. First, a batch stream is run during which the system manager specifies the hardware configuration and chooses optional software features. Software options can be included in the system to increase processing power or can be excluded from the system to conserve memory storage space. The system manager is advised, therefore, to read the entire guide to become familiar with the software. He can then select those features and parameters which best suit his applications requirements.

The batch stream and related operations are described in section 2.1 of this guide. After the system element is created by execution of the batch stream, the system manager must perform the second basic step, building the system disk, as instructed in Chapter 3. By following the instructions and guidelines given in Chapter 3, the system manager can create an initial RSTS/E system and be in a position to re-configure and re-create it to meet future expanding or altered needs without destroying current system program and data files. After building the system, the system manager must build the system library files and create the user accounts as described in Chapter 4.

1.6 SYSTEM MANAGEMENT

The manager of a RSTS/E system is provided with many options and facilities for controlling the RSTS/E system. These options are immediately made available when the system manager generates the system and builds the system disk. Many options are provided when the system is initialized as described in Chapter 3.

Special facilities are provided to the system manager to perform operations not available to ordinary users. Such facilities are termed privileged and are defined in Chapter 5 of this guide. The system manager can create a privileged account for himself and can, in turn, designate other users as privileged. Privileged users can make use of RSTS/E system facilities as described in Chapter 6 of this guide. It is suggested that the system manager read Chapters 6 and 7 of this guide before attempting to use privileged features or before assigning privileged accounts to other users.

The system manager can expand the system library to create his own utility programs to accomplish functions peculiar to the needs of his installation. He has available a wide selection of SYS system functions as described in Chapter 7 to facilitate such programming.

CHAPTER 2

RSTS/E SYSTEM GENERATION

This chapter describes the procedures to generate RSTS/E system code from software distributed on 7-track and 9-track magtape, DECtape, and RK disk cartridge (DECpack). Section 2.1 presents an overview of the system generation process and contains comments concerning the media on which DIGITAL distributes the software.

Section 2.2 contains the SYSLOD procedures required for magtape and DECtape software. Section 2.3 describes the copy procedures for disk cartridge software. Procedures to start the system generation batch command file are in Section 2.4. Examples and guidelines to answering configuration questions are in Sections 2.5, 2.6, and 2.7 respectively.

2.1 SYSTEM GENERATION OVERVIEW

The process for generating the RSTS/E system code consists of the following general steps.

a) If the software is on magnetic tape, bootstrap a tape to load the stand alone program SYSLOD and transfer the system generation monitor to a disk using SYSLOD. If the software is on disk cartridge, bootstrap the system generation disk to load the system generation monitor and copy both the system generation cartridge and the system library cartridge.

b) Log into the system generation monitor and initiate execution of the batch command file.

c) Answer configuration questions printed during the system generation dialogue.

d) Follow the instructions printed by the system generation monitor and mount and dismount tapes and disks as required.

After the RSTS/E system code is created, proceed to Chapter 3 to initialize the RSTS/E system disk and continue with Chapter 4 to build the system library files.

The system generation procedure is performed using a non-standard DOS/BATCH monitor as the system generation monitor. If the software supplied by DIGITAL is on magtape or DECtape, this monitor must be loaded onto an RF11, RK11, RPØ3, or RPØ4 disk with the stand alone program SYSLOD. If the DIGITAL-supplied software is on disk cartridge, bootstrap the cartridge to load the system generation monitor. (In this document, the disk on which the system generation monitor resides is referred to as the system generation disk. Such nomenclature differentiates the disk in question from the disk on which the RSTS/E public file structure resides and which is referred to as the RSTS/E system disk.)

The procedures for magtape or DECtape media involve transferring files to the system generation disk. After the system generation monitor is loaded onto a disk, one batch command file must be transferred from the tape to the disk. The system manager then initiates execution of the batch commands which transfer all required system generation programs (MACRO, LINK, CILUS, EDIT, PIP, and SYSLOD) from tape to disk without further interaction.

The procedures for disk cartridge media are similar to those for tape except that none of the file transfer operations are necessary. However, it is advisable to copy the disk cartridges containing the system generation and system library programs and to use the copies instead of the originals to generate the system. Copying of the disk cartridges is performed with the stand-alone program ROLLIN. To begin system generation, the system manager bootstraps the copy of the system generation disk cartridge, answers the initial monitor DIALOGUE questions, and types one command which initiates execution of the batch command file. Commands in the batch file delete any old RSTS/E system (RSTS.LCL file) which may exist from a prior system generation.

After the system generation monitor transfers all files from tape or deletes files from the disk cartridge, it executes a command in the batch file which runs the system generation program SYSGEN. The program prints approximately 70 hardware and software configuration questions and creates two files based on the answers typed in

response to each question. The answers must accurately reflect the hardware configuration on which RSTS/E will run and the software options desired. During the configuration dialogue, SYSGEN creates the configuration file CONFIG.MAC and a second batch command file SYSGEN.BAT, which are later used to create the RSTS/E system code tailored to local installation requirements.

After all configuration questions are answered, the system generation monitor executes the second batch command file. The monitor conditionally assembles the TBL module (system tables) and the TTY module (terminal service) using the configuration file created during the SYSGEN dialogue.

During execution of the second batch command file, the system generation monitor prints instructions to mount appropriate tapes or disks as each is required. If the 278Ø package is required on the system, the monitor also prints messages to mount the 278Ø tape or disk cartridge. Required 278Ø files are transferred to the system generation disk and assembled.

The system generation monitor next links the monitor code (RSTS), overlay code (OVR), error messages (ERR), initialization code (INIT), and octal debugging tool (ODT) and copies files from tape as needed. Subsequently, the monitor links the BASIC-PLUS monitor (Run Time System) code to include all BASIC-PLUS language and mathematical package options.

The final step in the system generation process creates a linked core image library (LICIL) of the RSTS/E system from the load modules created by the linking process. For magtape and DECtape media, the step includes writing the LICIL (RSTS.LCL file), system load maps, batch and configuration files, and SYSLOD program to a scratch tape. For disk cartridge software, the LICIL is created and remains on the system generation disk.

During the final step for magtape and DECtape media, the monitor prints a message indicating the exact command to type to write the contiguous core image library, or CIL, onto the RSTS/E system disk. The batch command file terminates by loading SYSLOD into memory. The system manager can type the exact command to SYSLOD and write the CIL to the RSTS/E system disk or, at some later time, can bootstrap the tape to load SYSLOD into memory. After writing the CIL to the RSTS/E

system disk, SYSLOD automatically bootstraps the device to load the RSTS/E initialization code into memory.

During the final step for disk cartridge media, the system generation monitor prints a message to mount and write enable the RSTS/E system disk. Typing one command continues executing the batch file which runs CILUS to write the CIL onto the RSTS/E system disk. If the RSTS/E system disk is either an RPØ3, RPØ4, or an RF11 disk, the batch command file terminates by bootstrapping that device. If the RSTS/E system disk is an RK disk, the system generation monitor prints a message to move the disk to RK unit Ø and to bootstrap it by the hardware loader to load the RSTS/E initialization code.

The RSTS/E CIL contains the system initialization code, the monitor, the BASIC-PLUS Run Time System, overlay code, error messages, and, optionally, the stand alone program ROLLIN. When the RSTS/E system disk is bootstrapped, the initialization code is loaded into memory. The initialization options described in Chapter 3 must be used to install necessary patches (PATCH option), to initialize the system disk (DSKINT option), to create the system files (REFRESH option), to set keyboard defaults (SETKEY option), to establish default start up parameters (DEFAULT option), and to begin time sharing (START option).

If interrupted, the entire system generation procedure need not be redone. If one of two checkpoints has been passed, merely return to the previous checkpoint. The checkpoints and procedures for recovery are indicated by messages printed by the system generation monitor.

2.2 MAGTAPE AND DECTAPE PROCEDURES

Magtape and DECtape procedures differ in the bootstrap procedures required and in the device designators used in several keyboard commands.

2.2.1 Magtape Bootstrap Procedure

Procedures to operate the magtape device are described in Section 5.6 of the RSTS-11 System User's Guide.

To bootstrap the magtape, perform the following steps.

Mount the SYSTEM GENERATION TAPE labelled

DEC-11-ORSPA-C-MC9 for a 9-track TU1Ø drive
DEC-11-ORSPA-C-MC7 for a 7-track TU1Ø drive
DEC-11-ORSPA-C-WC9 for a TU16 drive

on unit Ø with the write enable ring removed.

Ensure that the tape is at its load point. (The LD PT indicator comes on.) The computer does not bootstrap the device unless the tape is at its load point.

Set the ON-LINE/OFF-LINE switch on the tape unit to ON-LINE and ensure that the RDY indicator is lit.

Ensure that the console terminal is on line.

Follow the instructions in Appendix F for the type of hardware bootstrap on the system.

Proceed to Section 2.2.3 to transfer the system generation monitor from tape to disk.

2.2.2 DECtape Procedures

Procedures to operate the DECtape device are described in Section 5.5 of the RSTS-11 System User's Guide. To bootstrap the DECtape, perform the following steps.

Mount the DECtape reel labelled DEC-11-ORSBA-C-UC1 (SYSTEM GENERATION TAPE 1 of 2) on unit Ø.

Mount the DECtape reel labelled DEC-11-ORSBA-C-UC2 (SYSTEM GENERATION TAPE 2 of 2) on unit 1.

On DECtape units Ø and 1, set the REMOTE/OFF/LOCAL switch to REMOTE and the WRITE ENABLE/WRITE LOCK switch to WRITE LOCK.

Ensure that the console terminal is on line.

Bootstrap unit Ø by following the instructions in Appendix F for the type of hardware bootstrap device on the system.

Proceed to Section 2.2.3 to transfer the system generation monitor from tape to disk.

2.2.3 Loading the System Generation Monitor from Tape

When the tape is bootstrapped, the computer reads unit Ø and loads SYSLOD into memory. SYSLOD prints its identification line followed by the first in a series of queries as follows.

```
SYSLOD VØ8-Ø1A
CONSOLE FILL COUNT=
```

If SYSLOD does not print its identification and the processor halts, a parity error possibly was detected in reading the tape. Retry the entire procedure, including rewinding the tape to load point (if magtape). If the bootstrap procedures fail repeatedly, obtain a new tape reel.

2.2.3.1 Answering the SYSLOD Questions - When SYSLOD runs, perform the following steps.

> If the system generation disk is either an RKØ3 or RKØ5 cartridge or an RPØ3 or RPØ4 pack, mount it on drive unit Ø.
>
> Ensure that the system generation disk is on line, write enabled, and ready before proceeding. (After the DIALOGUE query is answered, SYSLOD does not recognize any devices previously not ready.)

Type the RETURN key in response to the CONSOLE FILL COUNT question and proceed as follows:

```
SYSLOD VØ8-Ø1A
CONSOLE FILL COUNT=          (Type RETURN key.)
DATE: 14-FEB-75              (Type in dd-mmm-yy format.)
DIALOGUE?                    (Type RETURN key.)
#
```

SYSLOD indicates that it is ready to accept a command string by printing the # character. A single command string is necessary to format the disk, to check for bad blocks, and to transfer the system

generation monitor to the disk. Use the following format for the SYSLOD command string.[1]

```
#xx:MONLIB.LCL/FO/CO:Ø/HO/BO<yy:MONLIB.LCL
```

where:

xx is DB for RPØ4 disk pack
DP for an RPØ3 disk pack
DK for an RKØ5 or RKØ3 disk cartridge
DF for an RF type disk.

yy is DTØ if DECtape medium is used or,
MTØ if either 7- or 9-track TU1Ø magtape medium is used.
MMØ if TU16 magtape software is used.

NOTE

The /FO switch is not necessary for RF-type disks.

The /FO switch in the SYSLOD command causes a removable disk to be formatted. If an RPØ3 disk pack is used, SYSLOD prints a message to remove a panel from the RP11C controller and set the FORMAT ENABLE/NORMAL switch. Leave the panel off the controller until formatting is complete. SYSLOD prints a second message to reset the FORMAT ENABLE/NORMAL switch before it continues.

The /CO:Ø switch writes one pattern on each block of the disk to ensure that no bad blocks are used. The /HO switch causes SYSLOD to place a pointer to the CIL in the bootstrap record. The /BO switch causes SYSLOD to bootstrap the device upon completing the transfer.

The entire process takes between 5 and 2Ø minutes depending upon the size and type of disk. If SYSLOD prints any error messages, consult Appendix B for the meaning and possible steps for recovery. Upon completing the transfer, SYSLOD prints the following messages.

```
SYSLOD COMPLETE

ANSWER WITH CARRET OR 'Y' CARRET: -IS YOUR LINE FREQUENCY 5Ø HERTZ?
DO YOU WANT TO DISABLE DIALOGUE FOREVER? NO
DOS/BATCH V9-2ØC
DATE:
```

[1]On an LA3Ø(S) DECwriter, an LA36 DECwriter II, and a VTØ5(B) alphanumeric display terminal, it may be necessary to use the SHIFT key while typing alphabetic characters in order to ensure that upper case characters are transmitted.

Type the RETURN key (CARRET denotes carriage return) if the line frequency of the power used to run the PDP-11 is 6Ø Hertz. Type YES and the RETURN key only if the power frequency is 5Ø Hertz.

SYSLOD then prints a question asking whether to disable the dialogue forever. The system generation monitor begins with a dialogue similar to that used by SYSLOD. It is possible to disable this dialogue at this time by typing YES followed by the RETURN key. For RSTS/E system generation purposes, type NO so that the dialogue is not disabled. Proceed to Section 2.4.1 for instructions on answering the system generation monitor dialogue.

2.3 DISK CARTRIDGE PROCEDURES

Disk cartridge procedures involve bootstrapping the device and copying the original cartridges using the stand alone program ROLLIN.

To prevent possible destruction of the system generation and system library disk cartridges, it is advisable to copy the cartridges and use the copies for generating systems. The cartridges are created on properly aligned drives. Since drive alignment drifts slightly in shipping and with age, problems sometimes occur. If the cartridges cannot be copied, a DIGITAL Field Service representative must check the drive alignment before system generation can continue. The stand alone program ROLLIN is included on the system generation disk cartridge to facilitate the copy operation.

2.3.1 Disk Cartridge Bootstrap

To bootstrap the cartridge, perform the following steps.

Physically mount the cartridge labelled DEC-11-ORSPA-C-HC (SYSTEM GENERATION DECPACK) in the RKØ3 or RKØ5 unit Ø.

Ensure that the RDY light is on.

If the cartridge has not been copied, ensure that the WR PROT light is on. (This condition write protects the disk.) If the copying is complete, ensure that the WR PROT light is off.

Ensure that the console terminal is on line.

Follow the instructions in Appendix F for the type of hardware bootstrap device on the system.

The system reads the disk and loads the system generation monitor which prints the following lines.

```
DOS/BATCH V9-2ØC
DATE:
```

If the monitor fails to identify itself, retry the entire operation, and carefully check the bootstrap procedure. After the monitor prints its identifying lines, proceed to Section 2.3.2 to copy the system generation and system library cartridges or proceed to Section 2.4 to start system generation.

2.3.2 Copying the Disk Cartridges Using ROLLIN

To copy the disk cartridges, perform the following steps.

Mount a new disk cartridge on drive unit 1.

Ensure that the RDY light comes on and that the WR PROT light for unit 1 is off.

Ensure that the WR PROT light for unit Ø is on. The original disk must be write protected to prevent inadvertent destruction.

Continue the dialogue in the following manner.

```
DOS/BATCH V9-2ØC
DATE:  11-JUL-74                (Type in dd-mmm-yy format.)
TIME:  Ø6:51                    (Type in hh:mm format.)
DIALOGUE?                       (Type the RETURN key.)
```

When the monitor prints the $ character, type the LO 1,1 command. The monitor prints the current date and time followed by the $ character.

```
$LO 1,1                         (Terminate with RETURN key.)
DATE:  11-JUL-74
TIME:  Ø6:52
$
```

The program CILUS is used to load ROLLIN. Type the RUN CILUS command as shown.

```
$RUN CILUS                      (Terminate with RETURN key.)
CILUS VØ8-Ø6A
#
```

CILUS runs and prints its header line followed by the # character. Type the command shown to run ROLLIN. Then type the command to format unit 1 and copy unit Ø to unit 1.

```
#ROLLIN.CIL/BO                    (Terminate with RETURN key.)

ROLLIN VØ7
#DK1:/FO<DKØ:/VE                  (Terminate with RETURN key.)
```

ROLLIN prints messages signalling the start and end of the format pass and the start of the verification pass. If no errors are encountered, ROLLIN prints the # character again as shown below.

```
STARTING RK FORMAT PASS

END RK FORMAT PASS

STARTING RK VERIFICATION PASS

#
```

If any errors are encountered, ROLLIN prints appropriate messages and the # character. A Field Service representative should be called to align the drive. If ROLLIN does not print any error messages, continue according to the following steps.

Move the LOAD/RUN switch to its LOAD position on both units Ø and 1.

When the LOAD light comes on, remove the cartridges from their respective drives.

Label the copied cartridge in such manner as SYSTEM GENERATION COPY. Store the original in a safe place.

Mount the disk cartridge labelled DEC-11-ORSLA-C-HA SYSTEM LIBRARY AND RELIABILITY DECPACK in unit Ø. Ensure that the WR PROT light is on.

Mount a second new cartridge in unit 1. Ensure that the RDY light comes on and that the WR PROT light for unit 1 is off. Ensure that the WR PROT light for unit Ø is on. The original disk on unit Ø must be write protected.

Since ROLLIN is still waiting, type the following command in response to the # character.

```
#DK1:/FO<DKØ:/VE
STARTING RK FORMAT PASS
END RK FORMAT PASS
STARTING RK VERIFICATION PASS
#
```

If any errors are encountered, ROLLIN prints appropriate messages and the # character. A Field Service representative should be called to align the drive. If ROLLIN does not print any error messages, continue according to the following steps.

Move the LOAD/RUN switch to its LOAD position on both units Ø and 1.

When the LOAD light comes on, remove the cartridges from their respective drives.

Label the copied cartridge in such manner as SYSTEM LIBRARY COPY. Store the original system library cartridge with the original system generation cartridge.

Mount the copy of the system generation disk in unit Ø and move the LOAD/RUN switch to its RUN position. Ensure that the RDY light comes on and that the WR PROT light is off. (The disk must be write enabled.)

Bootstrap unit Ø by typing the following command to ROLLIN.

```
#/BO:DK
DOS/BATCH V9-2ØC
DATE:
```

Proceed to Section 2.4.1 to start the system generation monitor.

2.3.3 Formatting Disks Using SYSLOD

During a system generation from disk cartridge distribution media, the CILUS program loads the RSTS/E Core Image Library onto the RSTS/E system disk. Since CILUS cannot format disks, the SYSLOD program

should be used at this point to format the disk to be used as the RSTS/E system disk. Bootstrap the copy of the System Generation disk cartridge, answer the monitor dialogue, perform the login procedure and run CILUS to load SYSLOD as follows.

```
DOS/BATCH V9-2ØC
DATE:  11-JAN-75                    Type the date in format shown
TIME:  16:15                        Type the time in format shown
DIALOGUE?                           Type the RETURN key

$LO 1,1
DATE:  11-JAN-75
TIME:  16:15

$RUN CILUS
CILUS VØ8-Ø6A

#SYSLOD.CIL/BO
```

The command shown bootstraps SYSLOD from the disk.

When SYSLOD runs, it prints a header line and the first query line. Mount the disk(s) to be formatted and answer the SYSLOD questions shown in the following sample. SYSLOD does not recognize any device which is not ready when the DIALOGUE question is answered.

```
SYSLOD VØ8-Ø1A
CONSOLE FILL COUNT=                 Type the RETURN key
DATE:  11-JAN-75                    Type the date in format shown
DIALOGUE?                           Type the RETURN key

#
```

Type the device designation with the /FO option as shown.

```
#DB1:/FO
#DP1:/FO

REMOVE THIRD PANEL BELOW RP11C INDICATORS
AND SET THE LEFT HAND SWITCH TO FORMAT ENABLE:  READY? YES
RESET SWITCH TYPE YES WHEN DONE YES
ARE YOU SURE? YES

#DK1:/FO
#
```

SYSLOD formats RPØ4 disk packs and RK disk cartridges without further interaction. For an RPØ3 disk pack, SYSLOD prints the messages shown above. The FORMAT ENABLE/NORMAL switch must be set and reset as requested. The YES response must be typed three times as indicated in the dialogue.

2.4 STARTING SYSTEM GENERATION

Once the system generation disk is bootstrapped, the system generation monitor runs. The system manager must perform the monitor dialogue and the login procedure; and initiate the batch command file. If magtape or DECtape distribution media are used, he must additionally transfer the batch command file from tape to the disk.

2.4.1 Monitor Dialogue

After the disk is bootstrapped, the system generation monitor prints its identification line followed by the first of several prompting lines. Type the current date in response to DATE: and the current time of day in response to TIME: as shown below. Terminate each response with the RETURN key.

```
DOS/BATCH V9-2ØC
DATE:  11-JAN-75          (Type in dd-mmm-yy format.)
TIME:  1Ø:12              (Type in hh:mm format.)
DIALOGUE?
```

The monitor dialogue can be omitted if the line printer used for system generation (unit Ø) is an LP11 with 8Ø columns and if the console fill count required is Ø. To omit the dialogue, type the RETURN key in response to the DIALOGUE query. The monitor prints the $ character. Continue at Section 2.4.2 to perform the login procedure.

To continue the dialogue, type YES followed by the RETURN key in response to the DIALOGUE query and proceed as shown.

```
DIALOGUE?  YES
DO YOU WANT TO RESET CONSOLE FILL COUNT? YES
FILL COUNT=
```

Type the console fill count in response to the FILL COUNT= query according to the following values for the type of console terminal on the computer.

Console Fill Values	Console Terminal Types
Ø	ASR-33 and ASR-35 Teletype; LA3ØS and LA3ØP DECwriter (11Ø and 15Ø baud); LA36 (any baud rate), VTØ5, VTØ5B, or VT5Ø display (11Ø, 15Ø, and 3ØØ baud).

Console Fill Values	Console Terminal Types
1	ASR-37 Teletype VTØ5B display at 6ØØ baud.
2	VTØ5B display at 12ØØ baud.
4	VTØ5B display at 24ØØ baud. LA3ØP DECwriter at 3ØØ baud.
12	LA3ØS DECwriter at 3ØØ baud.

For example, type 12 for an LA3ØS DECwriter at 3ØØ baud. If the line printer is an 132-column LP11, answer all remaining questions as shown below.

```
FILL COUNT=12
ARE ANY DEVICES DOWN? NO
DO YOU WANT TO CHANGE LINE PRINTER? YES
LS11? NO
HOW MANY COLUMNS? 132
LOWER CASE? NO
OVERPRINT? NO
DO YOU WANT TO CHANGE CARD READER DEFAULTS? NO
HAVE YOU GOT RKØ2 DISK? NO

$
```

If the line printer is an 8Ø-column LP11 (the default), type the RETURN key in response to the first line printer question. The remaining line printer questions do not appear. The card reader question is printed only if a card reader is present. Proceed to the login procedure when the $ character is printed.

2.4.2 Performing the Login Procedure

To log into the system generation monitor, type the LO 1,1 command in response to the $ character as shown below.

```
$LO 1,1
DATE:  21-JUL-74
TIME:  1Ø:56
$
```

If the login procedure is done properly, the monitor prints the current date and time followed by the $ character. Otherwise, the monitor prints an appropriate error message followed by the $ character. Simply try again.

At this point, procedures differ slightly for tape and disk media. Continue with Section 2.4.3 if the medium is either magtape or DECtape. Proceed to Section 2.4.4 if the medium is disk cartridge.

2.4.3 Transferring the Batch Command File from Tape

The PIP program must be run from tape and one command must be typed to transfer the batch command file to the system generation disk. Use the following format for the command to execute PIP.

```
$RUN xx:PIP
```

where:

xx is MT for either 7- or 9-track TU1Ø magtape,
MM for TU16 magtape, or
DT for DECtape.

When PIP prints the identification line and the # character as follows:

```
PIP V1Ø-Ø2
#
```

Use the following format to transfer the file.

```
#SY:<xx:SYSGEN
```

where:

xx is MT for either 7- or 9-track TU1Ø magtape,
MM for TU16 magtape, or
DT for DECtape.

PIP signals completion by printing the # character. Type the CTRL/C combination to terminate PIP and the KI command in response to the dot character printed by the monitor. For example,

```
#↑C
.KI
$
```

When the monitor prints the $ character, proceed to Section 2.4.4 to execute the batch command file.

2.4.4 Initiating the Batch Command File

To initiate execution of the first batch command file, type the following command in response to the $ character.

```
$BATCH SYSGEN
```

The monitor executes the commands in the batch command file SYSGEN. When the message BEGINNING OF RSTS/E SYSTEM GENERATION is printed, SYSGEN has entered the configuration dialogue. Section 2.5 contains an explanation of the various forms of the questions and of the procedure to restart the questions.

2.5 CONFIGURATION QUESTIONS

After the batch command file starts, the system generation program SYSGEN runs and enters the configuration dialogue. The dialogue is a series of approximately 70 hardware and software configuration questions. The questions come in both a long form and a short form. With each form, SYSGEN can calculate the answers. This latter feature is called automatic answers and is described in Section 2.5.1.

Long form questions contain explanatory information and are useful to anyone who is unfamiliar with the system. For a sample printout of the long form questions, see Section 2.6.1. To save time when the dialogue questions are familiar, select the short form of the questions. A sample printout of the short form questions appears in Sections 2.6.2 and 2.6.3. If a question is unclear, simply type the RETURN key in response to a short form query; SYSGEN prints the long form of that particular question.

During the dialogue, SYSGEN checks the answers entered. If an answer is incorrect, SYSGEN reprints the query or series of queries regarding that subject. To restart the dialogue, use the checkpoint facility described in Section 2.5.2. Implications of the configuration questions are given in Section 2.7.

After all configuration questions are answered, the monitor begins executing the second batch command file. For information on this part of the procedure, consult Section 2.8.

2.5.1 Automatic Answers

The SYSGEN automatic answer facility is enabled by appending /A to the response for the short/long form query. SYSGEN allows the user to accept the answer, to supply a different answer, or to print the question again. With automatic answers enabled, SYSGEN physically checks the hardware configuration of the computer on which it is running. SYSGEN addresses each device to determine its existence and the numbers and types of units. This facility is useful when the system generation is performed on the computer on which RSTS/E will run. Automatic answers can be misleading if the system generation is performed for a different machine.

SYSGEN denotes the meaning of the automatic answer by printing special characters as shown below.

1	The answer 1 is correct for the current hardware.
##Ø8##	The answer assumes something concerning the system.
##??##	The answer cannot be determined; an answer must be entered.

For example, the program can accurately determine whether the computer has either FIS or FPP and thus prints either **Y** or **NO** as the correct answer. However, for certain devices such as DECtape, Magtape, RK disk, and RP disk, SYSGEN can only verify the presence of the controller and must assume the existence of eight drives. The answer for such devices is either **ØØ** or ##Ø8## for the number of drives based upon the absence or presence of the controller. For an example of automatic answers, see Sections 2.6.1 and 2.6.2.

The following are valid responses to automatic answers.

LINE FEED key	Use the automatic answer as the response.
x and RETURN key	Use x as the response.
RETURN key	Reprint query or print long form of the query.

If SYSGEN prints an answer and it appears to be incorrect, the hardware is possibly configured incorrectly. For example, terminal interface wire jumpers are possibly cut improperly. In such circumstances, it is advisable to have a DIGITAL Field Service representative verify the hardware configuration.

Care must be taken if an automatic answer is overridden for a floating address device. The communications devices DJ11, DH11, DQ11, and DU11 have so-called floating addresses. This term means that the presence or absence of any of these devices alters the UNIBUS address assignments for the other devices. Before printing the automatic answer for such a device, SYSGEN recomputes the correct floating addresses based on responses to previous questions. For example, DJ11 multiplexers are assigned addresses on the UNIBUS before DH11 multiplexers. If 1 is the automatic answer for the DJ11 question but is overridden by typing 2, SYSGEN recomputes the floating address of and

looks for the presence of a DH11 at an address based on two DJ11 multiplexers. This procedure is correct only if the address jumpers for any DH11 are cut correctly for a system with two DJ11 units. Similarly, if the same automatic answer is overridden by entering Ø, SYSGEN recomputes the floating address of and looks for the presence of DH11 devices at an address based on no DJ11. Thus, in situations where the automatic answer is overridden for a floating address device, subsequent automatic answers are possibly incorrect. For more information on floating address assignments, see Appendix B of the RSTS/E System Reliability Test document.

2.5.2 Checkpoints

Prior to printing the message to begin RSTS/E system generation, SYSGEN prints a message indicating the start of a checkpoint. The message gives instructions for restarting the system generation if the process must be interrupted for any reason. Any steps performed before the checkpoint need not be redone. Two checkpoint messages appear during the system generation process. The first message immediately precedes the configuration dialogue. The second message immediately precedes the execution of the second batch command file SYSGEN.BAT. Each message describes the correct procedure to restart system generation from the checkpoint.

For example, to abort SYSGEN and terminate the batch stream, type the CTRL/C combination and TE (terminate). One command restarts the process from the related checkpoint. Assume SYSGEN prints a configuration question and the CTRL/C combination is typed.

```
AC FREQ?  ↑C
.
```

The monitor echoes the CTRL/C combination and prints the dot. The user then types TE in response to the dot to terminate the current batch command file, after which the monitor prints the current time, its identification and the $ character. For example,

```
.TE
TIME:-12:36:Ø1
DOS/BATCH V9-2ØC
$
```

To restart the dialogue from a checkpoint, type the appropriate command as described in the last checkpoint message. For example,

```
$BATCH SYSGN1
```

The monitor begins executing the checkpoint 1 batch command file SYSGN1.

2.6 SYSTEM GENERATION EXAMPLES

This section contains three samples of console printout produced during the generation of RSTS/E. The right hand margin of each sample has references to other sections which contain relevant descriptions. The samples show the system generation to the point where the software automatically bootstraps the resultant RSTS/E system disk and loads the initialization code into memory for the first time.

The first sample shows the output produced by using magtape software with an RPØ3 disk pack as the system generation disk and an RPØ3 disk pack as the resultant RSTS/E system disk. The sample shows all configuration questions in long form.

The second sample shows a system generation using disk cartridge (DECpack) software. The system generation disk and the resultant RSTS/E system disk are RKØ5 disk cartridges. The sample shows short form configuration questions and automatic answers.

The third sample shows a system generation using DECtape software and an RF disk as both the system generation disk and the resultant RSTS/E system disk. The printout shows the points in the procedure where the user must mount and dismount specified DECtapes.

2.6.1 Magtape Software Using Long Form Questions with Automatic Answers

Bootstrap the magtape with the hardware loader.

```
SYSLOD V08-01A                                                    (2.2.1)
CONSOLE FILL COUNT=0
DATE: 20-DEC-74                                                   (2.2.3.1)
DIALOGUE?

#DP:MONLIB.CIL/FO/CO:0/HO/BO<MT:MONLIB.LCL

REMOVE THIRD PANEL BELOW RP11C INDICATORS
AND SET THE LEFT HAND SWITCH TO FORMAT ENABLE: READY? YES
RESET SWITCH TYPE YES WHEN DONE YES
ARE YOU SURE? YES
SYSLOD COMPLETE

ANSWER WITH CARRET OR 'Y'CARRET:- IS YOUR LINE FREQUENCY 50 HERTZ?
DO YOU WANT TO DISABLE DIALOGUE FOREVER? NO

DOS/BATCH V9-20C                                                  (2.4.1)
DATE: 20-DEC-74
TIME: 10:00
DIALOGUE? YES
DO YOU WANT TO RESET CONSOLE FILL COUNT? YES
FILL COUNT=0
ARE ANY DEVICES DOWN? NO
DO YOU WANT TO CHANGE LINE PRINTER? YES
LS11? NO
HOW MANY COLUMNS ? 132
LOWER CASE? NO
OVERPRINT? NO
HAVE YOU GOT RK02 DISK? NO

$LO 1,1

DATE:-20-DEC-74                                                   (2.4.2)
TIME:-10:00:20

$RUN MT:PIP

PIP V10-02                                                        (2.4.3)

#SY:<MT:SYSGEN

#^C
.KI

$BATCH SYSGEN

                                                                  (2.4.4)
$JOB SYSGEN[1,1]
TIME:-10:00:50
$RUN MT:PIP
$RUN LINK
$RUN PIP
$RUN CILUS
$CH SYSGN3
```

```
SYSGEN:CHECKPOINT ONE:  DURING THE CONFIGURATION DIALOGUE      (2.5.2)
SYSGEN:WHICH FOLLOWS, YOU  MAY  ABORT  SYSGEN  BY  TYPING
SYSGEN:"CONTROL/C" AND THEN "TE".  RESTART FROM THE BEGIN-
SYSGEN:NING OF THE DIALOGUE BY TYPING "BATCH SYSGN1".
$RUN SYSGEN

**20-DEC-74**

BEGINNING OF RSTS/E  SYSTEM  GENERATION.                       (2.4.4)

QUESTIONS COME IN LONG AND SHORT  FORMS.                       (2.5.1)
IF  YOU  ARE  FAMILIAR WITH THEM, ANSWER.
"S" FOR SHORT; OTHERWISE, ANSWER "L" FOR
LONG FORM.  APPEND "/A" TO ENABLE  AUTO-
MATIC ANSWERS.  FORM ?  L/A

THE RSTS/E SYSTEM CAN BE  DISTRIBUTED ON
DECTAPE, MAGTAPE, RK CARTRIDGE DISKS, AND
RP DISK PACKS.  IN THE CASE  OF MAGTAPE,
A DISTINCTION MUST BE MADE  BETWEEN  THE
TU10 (MT) AND  TU16 (MM) MAGTAPE DRIVES.
ENTER THE TYPE OF DISTRIBUTION MEDIA AND
DRIVE TYPE  (IF MAGTAPE)  BEING USED FOR
THIS SYSTEM GENERATION (DT, MT, MM, RK, RP):

MEDIA ? MT

NOW YOU MUST SPECIFY THE  HARDWARE  CON-
FIGURATION  ON  WHICH THIS RSTS/E SYSTEM
WILL RUN.

WILL THIS SYSTEM RUN ON A PDP-11/45 COM-                       (2.7.1)
PUTER WITH FLOATING POINT PROCESSOR (YES
OR NO) :

FPP (11/45) ?                          **NO**

WILL THIS SYSTEM RUN ON A PDP-11/40 COM-
PUTER  WITH THE FLOATING INSTRUCTION SET
(YES OR NO) :

FIS (11/40) ?                          **Y **
```

```
THE RSTS/E SYSTEM  CLOCK CAN BE EITHER A                    (2.7.2)
KW11-L  LINE FREQUENCY CLOCK OR A KW11-P
PROGRAMMABLE CLOCK.  THE KW11-P CLOCK HAS
AN INTERNAL CRYSTAL OSCILLATOR WHICH CAN
BE USED AS THE SYSTEM TIME BASE IN AREAS
WHERE  THE  AC POWER  FREQUENCY  IS  NOT
ACCURATE.  IF YOU HAVE THE KW11-L CLOCK,
ANSWER "L".  FOR THE KW11-P, ANSWER  "P"
TO USE THE AC LINE FREQUENCY AS THE TIME
BASE, OR ANSWER  "C"  TO USE THE CRYSTAL
OSCILLATOR (L, P, OR C) :

CLOCK ?                                  **L **

THE AC POWER FREQUENCY  IS  NORMALLY  60    (2.7.3)
HERTZ  IN  THE  UNITED STATES, BUT ELSE-
WHERE IT CAN BE 50 HERTZ.  WHAT  IS  THE
AC  POWER FREQUENCY AT WHICH THIS SYSTEM
WILL RUN (50 OR 60) :

AC FREQ ?                                ##60##

THE KW11-P 10KHZ CRYSTAL OSCILLATOR WILL    (2.7.4)
BE USED  AS THE SYSTEM  TIME BASE.  THIS
FREQUENCY IS DIVIDED  IN THE HARDWARE TO
PROVIDE THE DESIRED INTERRUPT RATE.  FOR
RSTS/E  THE  INTERRUPT  RATE  CAN BE ANY
MULTIPLE OF 50HZ BETWEEN 50 AND 1000 HZ.
THE LARGER MULTIPLES PROVIDE BETTER TIME
RESOLUTION FOR JOB ACCOUNTING AND SCHED-
ULING  PURPOSES AT THE SMALL EXPENSE  OF
ADDITIONAL  CLOCK  INTERRUPT  OVERHEAD.
PLEASE  ENTER THE INTERRUPT RATE DESIRED
(50,100,150,...,1000) :

KW11P INTERRUPT RATE ?                   ##100##

THE NEXT FEW  QUESTIONS  DEAL  WITH  THE
NUMBERS AND TYPES OF TERMINAL INTERFACES
ON THE SYSTEM.

THE  CONSOLE  TERMINAL (KB0) MAY  BE  AN    (2.7.5)
ASR33,  KSR33, ASR35, OR KSR35 TELETYPE;
LA30 (PARALLEL), LA30S (SERIAL), OR LA36
DECWRITER; OR A VT05 (300 BAUD OR LESS),
VT05B (600 BAUD OR MORE), VT50, OR  VT51
SCOPE.  PLEASE IDENTIFY THE TYPE OF CON-
SOLE  TERMINAL  ON  THIS  RSTS/E  SYSTEM
(ASR33, LA36, VT05B, VT51, ETC.) :

CONSOLE TYPE ?                  ##LA30S.## LA36
```

SERIAL ASCII TERMINALS CONNECTED DIRECT- (2.7.5.1)
LY TO THE COMPUTER AND THOSE CONNECTED
THROUGH LEASED PRIVATE TELEPHONE LINES
(NOT DIAL-UP) MAY USE EITHER OF TWO
CLASSES OF SINGLE LINE INTERFACES (OR
THE DH11 MULTIPLEXER, DISCUSSED LATER).
THE FIRST CLASS INCLUDES THE KL11, LC11,
DL11A, AND DL11B INTERFACES. IF THERE
ARE NONE OF THESE, ANSWER 0; OTHERWISE
WHAT IS THE TOTAL NUMBER OF THESE SINGLE
LINE INTERFACES (1 TO 16 -DO NOT INCLUDE
THE CONSOLE TERMINAL!):

KL11,LC11,DL11A,DL11B'S ? **04**

THE SECOND CLASS OF SINGLE LINE INTER- (2.7.5.1)
FACES USED TO CONNECT TERMINALS LOCALLY
OR THROUGH LEASED TELEPHONE LINES (NOT
DIAL-UP) INCLUDES THE DL11C AND DL11D
INTERFACES. HOW MANY DL11C'S AND DL11D'S
ARE ON THIS SYSTEM (0 TO 31 -- DO NOT
INCLUDE THE CONSOLE TERMINAL) :

DL11C, DL11D'S ? **02**

THERE ARE TWO KINDS OF INDIVIDUAL INTER-
FACES FOR AUTOMATIC ANSWER DATASETS USED
ON THE DIAL-UP TELEPHONE NETWORK. THE
FIRST OF THESE IS THE DC11. HOW MANY
DC11-DA LINES DO YOU HAVE (0 TO 32) :

DC11'S ? **02**

THE SECOND KIND OF INDIVIDUAL LINE
DIAL-UP INTERFACE IS THE DL11E. HOW MANY
DL11E'S ARE ON THIS SYSTEM (0 TO 31) :

DL11E'S ? **01**

THE DJ11 MULTIPLEXER IS NOT SUPPORTED
UNDER RSTS/E. HOWEVER, TO DETERMINE THE
CORRECT FLOATING DEVICE ADDRESSES FOR
OTHER SUPPORTED HARDWARE, WE NEED TO KNOW
THE NUMBER OF DJ11 MULTIPLEXERS CON-
NECTED TO THIS SYSTEM (0 TO 16) :

DJ11'S ? **00**

EACH DH11 MULTIPLEXER CAN CONNECT UP TO
16 TERMINALS TO THE SYSTEM. PLEASE ENTER
THE TOTAL NUMBER OF DH11'S ATTACHED TO
THIS SYSTEM INCLUDING ANY WHICH WILL NOT
BE USED BY THE RSTS/E SOFTWARE (0 TO 16)

DH11'S ? **03**

```
DH11 UNIT 00 TYPE ?                ##AA##                    (2.7.5.2)

THERE ARE FIVE STANDARD DH11 VARIATIONS:

DH11-AA  WILL SUPPORT ANY COMBINATION OF
         LOCAL OR REMOTE LINES. IF AUTO-
         ANSWER DATASETS ARE TO BE SUP-
         PORTED, A DM11-BB MODEM CONTROL
         MULTIPLEXOR IS ALSO REUUIRED.

DH11-AB  WIRED FOR TELEGRAPH LINES AND
         TREATED AS A DH11-AA WITHOUT
         MODEM CONTROL BY RSTS/E.

DH11-AC  240 V, 50 HZ VERSION OF DH11-AA.

DH11-AD  CONNECTS 16 EIA/CCITT (RS232-C)
         COMPATIBLE LINES INCLUDING FULL
         MODEM CONTROL.

DH11-AE  SAME AS DH11-AD WITHOUT MODEM
         CONTROL. USED FOR CONNECTION OF
         LOCAL EIA TERMINALS OR PRIVATE
         LINE MODEMS.

PLEASE ENTER THE TYPE OF EACH DH11 UNIT
(AA, AB, AC, AD, OR AE) :

DH11 UNIT 00 TYPE ?                ##AA## AA

DOES DH11 00 INCLUDE A DM11-BB? ##??## 

THERE ARE 02 DM11-BB COMPATIBLE MODEM
CONTROL MULTIPLEXERS ON THIS SYSTEM. YOU
HAVE ACCOUNTED FOR 00 SO FAR. NOTE: ALL
DH11-AD'S INCLUDE DM11-BB MODEM CONTROL.

IF ANY DIAL-UP TELEPHONE LINES ARE TO BE
CONNECTED THROUGH AN AUTOMATIC ANSWER
DATASET TO THIS DH11 MULIIPLEXER, THE
DM11-BB MODEM CONTROL IS ALSO REQUIRED.
DOES THIS DH11 UNIT INCLUDE A DM11-BB
(YES OR NO) :

DOES DH11 00 INCLUDE A DM11-BB? ##??## YES

DH11 UNIT 00 LINES ENABLED ?     ##16##

FOR THIS DH11 UNIT, ENTER THE NUMBER OF
LINES WHICH WILL BE USED NOW OR IN THE
FORSEEABLE FUTURE. A RESPONSE OF 16 WILL
PERMIT ALL LINES TO BE USED. IF THE NUM-
BER OF LINES ENABLED (N) IS LESS THAN 16
RSTS/E WILL NOT BE CONFIGURED FOR AND
WILL NOT RECOGNIZE LINES N THRU 15 ON
THIS DH11 UNIT. A 0 RESPONSE IMPLIES THE
WHOLE DH11 UNIT IS TO BE IGNORED. ENTER
THE NUMBER OF LINES ENABLED (0 TO 16) :

DH11 UNIT 00 LINES ENABLED ?     ##16## 16

DH11 UNIT 01 TYPE ?              ##AA## AE

DH11 UNIT 01 LINES ENABLED ?     ##16## 8

DH11 UNIT 02 TYPE ?              ##AA## AD

DH11 UNIT 02 LINES ENABLED ?     ##16## 0
```

```
THE NEXT  SEVERAL  QUESTIONS  DEAL  WITH
TERMINAL RELATED SOFTWARE FEATURES WHICH
MAY BE INCLUDED IN THE RSTS/E SYSTEM  AT
THE OPTION OF THE SYSTEM MANAGER.

PSEUDO KEYBOARDS PERMIT INTERACTIVE JOBS                    (2.7.6)
TO BE RUN WITHOUT TYING UP A REAL TERMI-
NAL.  THEY ARE PRIMARILY INTENDED FOR USE
BY A BATCH CONTROL PROGRAM  WHICH  FEEDS
COMMANDS TO ONE OR MORE PSEUDO KEYBOARDS
DEDICATED TO RUNNING  BACKGROUND  TASKS.
HOW MANY PSEUDO KEYBOARDS WOULD YOU LIKE
TO HAVE (0 TO 63) :

PSEUDO KEYBOARDS ?                ##04##

RSTS/E IS CAPABLE OF SUPPORTING IBM 2741                    (2.7.7)
COMPATIBLE  TERMINALS  ON DL11D,  DL11E,
AND  DC11 SINGLE LINE INTERFACES, OR THE
DH11 MULTIPLEXER.  IF YOU DO NOT WANT ANY
2741 SUPPORT THEN ANSWER "NO".  OTHERWISE
ANSWER "SL" TO INCLUDE 2741 SUPPORT  FOR
SINGLE LINE INTERFACES ONLY; ANSWER "DH"
FOR 2741 SUPPORT ONLY ON DH11 LINES;  OR
ANSWER "BOTH" TO PROVIDE 2741 SUPPORT ON
BOTH SINGLE LINE INTERFACES AND THE DH11
MULTIPLEXER(S) :

2741'S ?                          ##NO## BOTH

2741 TERMINALS ARE AVAILABLE  WITH  MANY                    (2.7.8)
CODE/KEYBOARD  ARRANGEMENTS.  FOUR OF THE
MORE COMMON CODES  ARE  SUPPORTED  UNDER
RSTS/E.  THESE INCLUDE IBM CORRESPONDENCE
CODE (STANDARD SELECTRIC TYPEWRITER KEY-
BOARD),  EXTENDED  BINARY CODED DECIMAL,
STANDARD BINARY CODED DECIMAL,  AND CALL
360 BASIC CODE   ANY COMBINATION OF FROM
ONE TO FOUR OF THESE CODE TABLES CAN  BE
INCLUDED IN THE RSTS/E TERMINAL SERVICE.
PLEASE ENTER THE ABBREVIATED CODE  NAMES
(CORR,EBCD,SBCD,C360)  SEPARATED BY COM-
MAS FOR 2741 CODES YOU WISH TO SUPPORT :

2741 CODE(S) ?                    ##CORR.## EBCD,CORR

AN OPTIONAL FEATURE OF THE RSTS/E TERMI-                    (2.7.9)
NAL SERVICE  ALLOWS  ONE JOB TO INTERACT
WITH SEVERAL TERMINALS  THROUGH  SPECIAL
FORMS  OF THE THE RECORD I/O GET AND PUT
STATEMENTS.  THIS FEATURE IS  USEFUL  IN
APPLICATIONS  WHERE THE SAME BASIC FUNC-
TION IS PERFORMED ON  SEVERAL  TERMINALS
AND A SEPARATE JOB FOR EACH IS UNDESIRE-
ABLE OR AT LEAST INEFFICIENT.  WOULD YOU
LIKE TO INCLUDE THIS FEATURE (YES OR NO)

MULTI-TTY SERVICE ?               ##Y ##
```

```
THE NEXT QUESTIONS DEAL WITH THE NUMBERS
AND KINDS OF DISK UNITS ON THIS SYSTEM.

THE RJS04/RJS03 DISK SYSTEM CONSISTS OF                    (2.7.10)
AN RH11 DEVICE CONTROLLER AND FROM ONE
TO EIGHT RS03 (256K) OR RS04 (512K WORD)
FIXED-HEAD DISK DRIVES. RSTS/E SUPPORTS
ANY COMBINATION OF RS03'S AND RS04'S UP
TO A TOTAL OF FOUR DRIVES FOR SWAPPING
AND NON-RESIDENT CODE (ADDITIONAL UNITS
MAY BE CONNECTED TO THE RH11, HOWEVER).
PLEASE ENTER THE NUMBER OF EACH TYPE OF
DRIVE (0 TO 4) :

RS03'S ?                                   **00**

RS04'S ?                                   **00**

THE RF11 CONTROLLER IS USED TO CONTROL
UP TO 8 OF THE RS11 256K WORD FIXED-HEAD
DISKS. IF THIS SYSTEM HAS NONE OF THESE
DISKS, ANSWER 0; OTHERWISE, HOW MANY
256K WORD RS11 DISK PLATTERS ARE THERE
(1 TO 8) :

RF/RS11'S ?                                **00**

IT IS POSSIBLE TO USE THE RC11 DISK CON-
TROLLER WITH UP TO FOUR RS64 64K WORD
FIXED-HEAD DISKS FOR SWAPPING AND
NON-RESIDENT CODE. IF THERE ARE NONE OF
THESE DISK UNITS, ANSWER 0; OTHERWISE,
HOW MANY 64K WORD RS64 DISK PLATTERS ARE
THERE (1 TO 4) :

RC11/RS64'S ?                              **04**

THE RJP04 DISK SYSTEM CONSISTS OF AN
RH11 CONTROLLER AND UP TO EIGHT OF THE
RP04 40 MILLION WORD MOVING-HEAD DISK
PACK DRIVES. HOW MANY RP04 DRIVES ARE
INCLUDED IN THIS CONFIGURATION (0 TO 8)
NOTE: IF YOU HAVE TWO OR MORE RP04
DRIVES BUT PREFER TO CONSERVE MEMORY BY
USING THE NON OVERLAPPED SEEK DRIVER,
APPEND "/NO" TO THE NUMBER OF DRIVES :

RP04'S ?                                   **00**

THE RP11-C CONTROLLER IS USED TO CONTROL
ANY COMBINATION OF UP TO EIGHT RP02 (10
MILLION WORD) OR RP03 (20 MILLION WORD)
MOVING-HEAD DISK PACK DRIVES. ENTER THE
TOTAL NUMBER OF RP02 AND RP03 DRIVES ON
THIS SYSTEM (0 TO 8). NOTE: IF YOU HAVE
TWO OR MORE DRIVES BUT WANT TO CONSERVE
MEMORY BY USING THE NON OVERLAPPED SEEK
DRIVER, APPEND "/NO" TO THE NUMBER OF RP
DRIVES :

RP02/RP03'S ?                              ##08## 2
```

```
THE RK11 CONTROLLER IS USED TO CONTROL
UP TO 8 OF THE RK03 OR RK05 1.2 MILLION
WORD MOVING-HEAD CARTRIDGE DISK DRIVES.
HOW MANY RK03 OR RK05 DRIVES ARE THERE
(0 TO 8). NOTE: IF YOU HAVE TWO OR MORE
RK DRIVES BUT WANT TO CONSERVE MEMORY BY
USING THE NON OVERLAPPED SEEK DRIVER,
APPEND "/NO" TO THE NUMBER OF DRIVES :

RK03/RK05'S ?                          ##08## 2/NO

THE "SYSTEM DISK" CONTAINS THE SYSTEM
CORE IMAGE AND THE FIRST PART OF THE
PUBLIC DISK STRUCTURE. THE SYSTEM DISK
CAN BE EITHER THE RF11 (RF) FIXED-HEAD
DISK, AN RK03 OR RK05 (RK) CARTRIDGE, OR
AN RP02 (RP),RP03 (RP),OR RP04 (RB) DISK
PACK. ON THIS SYSTEM WHICH SHALL IT BE
(RF, RK, RP, OR RB) :

SYSTEM DISK ?                          **RP**

THE NEXT FEW QUESTIONS DEAL WITH THE                    (2.7.11)
PERIPHERAL DEVICES ATTACHED TO THIS
RSTS/E SYSTEM.

THE TJU16 MAGTAPE SYSTEM CONSISTS OF AN
RH11 MASSBUS INTERFACE, TM02 CONTROLLER,
AND FROM ONE TO EIGHT TU16 TAPE DRIVES.
IF THIS SYSTEM HAS NO TU16 MAGTAPE UNITS
THEN ANSWER 0; OTHERWISE, HOW MANY TU16
DRIVES ARE ON THIS SYSTEM (1 TO 8) :

TU16'S ?                               **00**

THE TM11 MAGTAPE CONTROLLER CAN SUPPORT
UP TO EIGHT TU10 MAGTAPE DRIVES. PLEASE
ENTER THE NUMBER OF TU10 DRIVES ON THIS
SYSTEM (0 TO 8) :

TU10'S ?                               *##08## 2

THE TC11 DECTAPE CONTROLLER CAN CONTROL
UP TO 8 SINGLE DRIVES. IF THIS SYSTEM
HAS NO DECTAPE AT ALL, ANSWER 0; OTHER-
WISE, HOW MANY SINGLE DECTAPE DRIVES,
NOT TU56'S, ARE THERE (1 TO 8 - NOTE
THAT EACH TU56 DUAL DRIVE HAS 2) :

DECTAPES ?                             ##08## 4
```

```
THE RSTS/E SYSTEM CAN HAVE  UP TO  EIGHT
LINE  PRINTERS OF THE LP11 OR LS11 TYPE.
THESE PRINTERS ARE REFERRED  TO  BY  THE
DEVICE NAMES LP0: THROUGH LP7:.  IF THERE
IS NO LINE PRINTER, ANSWER 0; OTHERWISE,
ENTER  THE  NUMBER OF PRINTERS OF EITHER
TYPE (1 TO 8) :

PRINTERS ?                          **03**

CERTAIN FEATURES OF  EACH  OF  THE  LINE
PRINTERS  ON THE SYSTEM MUST BE DEFINED.
THE NEXT FEW QUESTIONS DEAL  WITH  THOSE
FEATURES.

LP0: TYPE ?                         **LP**

FOR THE LINE PRINTER DESIGNATED  "LP0:",
PLEASE  DESIGNATE  THE  TYPE OF PRINTER:
"LP11" (DEC STANDARD DRUM PRINTERS),  OR
"LS11" (CENTRONICS SERIAL PRINTER) :

LP0: TYPE ?                         **LP**

LP0: WIDTH ?                        **132**

FOR THE LP11 PRINTER DESIGNATED  "LP0:",
ENTER LINE WIDTH - 80 OR 132 CHARACTERS:

LP0: WIDTH ?                        **132**

LP0: LOWER CASE ?                   ##NO##

SOME LP11 PRINTERS HAVE A  96  CHARACTER
SET  INCLUDING  LOWER  CASE  CHARACTERS.
DOES THIS  PRINTER  INCLUDE  LOWER  CASE
(YES OR NO).

LP0: LOWER CASE ?                   ##NO##

LP1: TYPE ?                         **LP**

LP1: WIDTH ?                        **80**

LP1: LOWER CASE ?                   ##NO## YES

LP2: TYPE ?                         **LS**

IF THERE IS A LINE PRINTER AVAILABLE FOR                (2.7.12)
USE DURING SYSGEN (MUST BE LP0), IT WILL
BE USED TO PRINT THE  SYSTEM  LOAD  MAPS
AND, OPTIONALLY, TO PRINT ASSEMBLY LIST-
INGS OF THE SYSTEM TABLES  AND  TERMINAL
SERVICE  MODULES.  IF YOU HAVE A PRINTER
AND WANT THE  ASSEMBLY  LISTINGS  AND/OR
LOAD MAPS  PRINTED DURING SYSGEN, ANSWER
"YES", OTHERWISE ANSWER "NO".

LP0 FOR SYSGEN ?                    **Y **
```

```
THE SYSTEM GENERATION PROCESS INCLUDES                    (2.7.13)
THE ASSEMBLY OF THE SYSTEM TABLES AND
THE TERMINAL SERVICE SOFTWARE.  IT IS
POSSIBLE TO HAVE LISTINGS OF THESE TWO
TAILORED MODULES PRINTED ON THE LINE
PRINTER.  WOULD YOU LIKE TO PRINT THESE
LISTINGS (YES OR NO).

LISTINGS ?                          ##NO##

THREE CARD READERS ARE AVAILABLE ON
RSTS/E SYSTEMS, THE CR11 PUNCHED CARD
READER, THE CM11 MARKED CARD READER, AND
THE CD11 HIGH-SPEED PUNCHED CARD READER.
IF THIS SYSTEM DOES NOT HAVE A CARD
READER, ANSWER 'NONE'; OTHERWISE, DOES
IT HAVE A 'CR11', A 'CM11' OR A 'CD11'.

CARD READER ?                       **CR** CD11

THE CARDS READ BY RSTS/E WILL NORMALLY    (2.7.14)
BE INTERPRETED AS BEING IN DEC 029 CODE.
IF THIS INSTALLATION USES CARDS IN DEC
026 CODE OR FROM A 1401, YOU MAY SELECT
THE APPROPRIATE SPECIAL DECODING TABLE.
IF YOU DO NOT WANT THE SPECIAL DECODING,
ANSWER "029"; OTHERWISE, SHOULD IT BE
"026" OR "1401" CODE :

CARD DECODE ?                       ##029## 1401

IS THERE A HIGH-SPEED PAPER TAPE READER
(YES OR NO).

P.T. READER ?                       **Y **

IS THERE A HIGH-SPEED PAPER TAPE PUNCH
(YES OR NO).

P.T. PUNCH ?                        **Y **

THE DQ11 NPR SYNCHRONOUS LINE INTERFACE
IS NOT SUPPORTED UNDER RSTS/E.  HOWEVER,
TO DETERMINE THE CORRECT FLOATING DEVICE
ADDRESSES FOR OTHER SUPPORTED HARDWARE,
WE NEED TO KNOW HOW MANY DQ11'S ARE CON-
NECTED TO THIS SYSTEM (0 TO 16) :

DQ11'S ?                            **00**
```

```
ONE DP11 OR DU11 SYNCHRONOUS LINE INTER-
FACE  IS REQUIRED FOR  OPERATION  OF THE
RSTS/E 2780 PACKAGE (IF INCLUDED BELOW).
ADDITIONAL DP11  AND DU11 INTERFACES MAY
BE  CONNECTED  TO THE  UNIBUS  FOR OTHER
PURPOSES.  PLEASE  SPECIFY THE NUMBER OF
THESE DEVICES  ON THIS SYSTEM (INCLUDING
ANY TO BE USED BY THE 2780 PACKAGE) :

DP11'S ?                          **02**

DU11'S ?                          **01**

THE  RSTS/E 2780  PACKAGE  EMULATES  THE                    (2.7.15)
OPERATION  OF  THE IBM 2780 MODEL 1 DATA
TRANSMISSION TERMINAL.  THE PACKAGE PER-
MITS  THE RSTS/E  SYSTEM  TO COMMUNICATE
WITH ANY IBM SYSTEM  WHICH SUPPORTS  THE
DEVICE,OR WITH ANOTHER RSTS 2780 SYSTEM.
THE  2780 PACKAGE  IS SOLD AND SUPPORTED
BY THE DEC  COMMUNICATIONS GROUP.  IT IS
NOT INCLUDED IN THE STANDARD RSTS/E KIT.
DO YOU HAVE THE 2780 PACKAGE AND WISH TO
INCLUDE IT IN THIS SYSTEM (YES OR NO) :

2780 ?                         ##NO## YES

THE 2780 PACKAGE REQUIRES EITHER A  DP11
OR  DU11  SYNCHRONOUS LINE INTERFACE AND
THE  KG11-A  (COMMUNICATIONS  ARITHMETIC
OPTION).  WHICH TYPE OF SYNCHRONOUS LINE
INTERFACE WILL  YOUR  2780  PACKAGE  USE
(DP11 OR DU11) :

2780 INTERFACE ?               ##DU## DP11

THE FOLLOWING DEVICES ARE NOT  SUPPORTED                    (2.7.16)
BY  THE  RSTS/E SYSTEM SOFTWARE ALTHOUGH
THEY MAY BE ATTACHED TO THE  UNIBUS  FOR
OTHER PURPOSES.

DM11-A  SINGLE-SPEED MULTIPLEXER
DN11-DA AUTOMATIC CALLING UNIT INTERFACE
DR11A,C GENERAL DEVICE INTERFACES
PA611R  TYPESET READER
PA611P  TYPESET PUNCH
DT03-FP PROGRAMMABLE BUS SWITCH
DX11    IBM 360/370 INTERFACE
GT40    GRAPHIC SYSTEM
LPS11   LABORATORY PERIPHERAL SYSTEM
KW11W   WATCHDOG TIMER

ARE ANY OF THESE  NON-SUPPORTED  DEVICES
CONNECTED TO THIS SYSTEM (YES OR NO).

NON-SUPPORTED DEVICES ?        ##Y ## YES
```

```
ENTER THE NUMBER OF DM11-A  SINGLE SPEED
MULTIPLEXERS ON THIS SYSTEM (0 TO 16) :

DM11-A'S ?                              **00**

HOW MANY DN11-DA  AUTOMATIC CALLING UNIT
LINE  INTERFACES ARE  CONNECTED  TO THIS
SYSTEM (0 TO 64) :

DN11-DA'S ?                             **02**

ENTER  THE  TOTAL  NUMBER  OF DR11-A AND
DR11-C GENERAL DEVICE INTERFACES ON THIS
SYSTEM (0 TO 32) :

DR11-A,C'S ?                            **00**

HOW  MANY  PA611R  TYPESET  READERS  ARE
INCLUDED IN THIS CONFIGURATION (0 TO 16)

PA611R'S ?                              **00**

HOW MANY PA611P TYPESET PUNCHES ARE CON-
NECTED TO THIS SYSTEM (0 TO 16) :

PA611P'S ?                              **00**

ENTER THE NUMBER OF DT03-FP PROGRAMMABLE
BUS SWITCHES ON YOUR PDP-11 (0 TO 8) :

DT03-FP'S ?                             **00**

HOW MANY DX11 IBM 360/370 INTERFACES ARE
ATTACHED TO THIS SYSTEM (0 TO 4) :

DX11'S ?                                **00**

ANSWER  "0"  IF YOU HAVE NO GT40 GRAPHIC
SYSTEM HARDWARE.  ALSO ANSWER "0" IF YOUR
GT40 HAS ITS OWN PDP-11 CPU AND THE CON-
NECTION TO  THE HOST (RSTS/E) MACHINE IS
THROUGH A STANDARD  TERMINAL  INTERFACE.
ENTER "1" ONLY IF A GT40 GRAPHIC DISPLAY
PROCESSOR IS  CONNECTED DIRECTLY  TO THE
UNIBUS  OF  THE  RSTS/E MACHINE  AND  ITS
INTERRUPT VECTORS ARE ASSIGNED ACCORDING
TO DEC STANDARDS FOR FLOATING VECTORS :

GT40 (1 ONLY) ?                         **00**

TYPE "1" IF AN LPS LAB PERIPHERAL SYSTEM
IS ATTACHED TO  THIS  SYSTEM.  OTHERWISE
ENTER "0".

LPS (1 ONLY) ?                          **00**

ENTER "1" IF A  KW11W  WATCHDOG TIMER IS
INCLUDED  IN YOUR CONFIGURATION.  OTHER-
ENTER "0".

KW11W (1 ONLY) ?                        **00**
```

```
THE REMAINING QUESTIONS DEAL WITH THE
NAME, CAPACITY, AND FEATURES OF THIS
RSTS/E SYSTEM PROVIDED AT THE SYSTEM
MANAGER'S OPTION.

RSTS/E SYSTEMS IDENTIFY THEMSELVES WITH
A 14 CHARACTER INSTALLATION NAME.  ENTER
UP TO 14 CHARACTERS:    --------------

THE INSTALLATION NAME :          ##SYSTEM TEST## SYSTEM #880

WITH SUFFICIENT HARDWARE RSTS/E CAN HAN-                    (2.7.17)
DLE UP TO 63 SIMULTANEOUS JOBS.  WHAT IS
THE MAXIMUM NUMBER OF JOBS (JOBMAX) TO
BE USED AT THIS INSTALLATION (1 TO 63) :

MAXIMUM JOBS ?                   ##20## 32

SMALL BUFFERS ARE 16 WORD BLOCKS IN MON-                    (2.7.18)
ITOR STORAGE THAT ARE DYNAMICALLY ALLO-
CATED AS INPUT AND OUTPUT BUFFERS, FILE
PARAMETER BLOCKS, ETC.  STORAGE MAY BE
ALLOCATED FOR 30 TO 999 SMALL BUFFERS.
THE RECOMMENDED NUMBER IS AT LEAST 10
FOR EACH POSSIBLE JOB.  HOW MANY SMALL
BUFFERS WOULD THIS INSTALLATION LIKE TO
HAVE (30 TO 999) :

SMALL BUFFERS ?                  ##200## 320

BIG BUFFERS ARE 256 WORD BLOCKS IN MONI-                    (2.7.19)
TOR STORAGE THAT ARE DYNAMICALLY ALLO-
CATED FOR INPUT AND OUTPUT TO DECTAPE.
STORAGE MAY BE ALLOCATED FOR FROM 1 TO 8
BIG BUFFERS.  THE RECOMMENDED NUMBER IS
ONE FOR EACH DECTAPE DRIVE THAT WILL BE
SIMULTANEOUSLY IN USE.  HOW MANY BIG
BUFFERS WOULD THIS INSTALLATION LIKE TO
HAVE (1 TO 8).

BIG BUFFERS ?                    ##01##

INTER-JOB COMMUNICATION IS ACCOMPLISHED                     (2.7.20)
THROUGH THE USE OF SEND AND RECEIVE SYS-
TEM FUNCTION CALLS.  SEND/RECEIVE CODE IS
NORMALLY NON-RESIDENT BUT A SMALL TABLE
OF ACTIVE RECEIVERS IS RESIDENT.  THIS
TABLE CONSISTS OF FIVE WORDS FOR EACH
RECEIVER ALLOWED TO BE ACTIVE SIMULTA-
NEOUSLY (4 TO 8 RECEIVERS ARE USUALLY
SUFFICIENT).  PLEASE ENTER THE NUMBER OF
ACTIVE RECEIVERS ALLOWED AT YOUR INSTAL-
LATION (0 TO JOBMAX) :

RECEIVERS ?                      ##08## 6

RSTS/E SYSTEMS CAN ATTEMPT TO AUTOMATI-                     (2.7.21)
CALLY RECOVER FROM A POWER FAILURE.  DO
YOU WANT THIS CAPABILITY IMPLEMENTED IN
THIS SYSTEM (YES OR NO) :

POWER FAIL ?                     ##Y ##
```

AN OPTIONAL FEATURE OF THE RSTS/E SYSTEM ALLOWS THE FILE PROCESSOR (FIP) TO USE FREE SMALL OR BIG BUFFERS TO STORE DISK DIRECTORY INFORMATION. THIS IMPROVES THE SPEED OF DIRECTORY PROCESSING BY AVOIDING THE NECESSITY OF REREADING COMMONLY USED DIRECTORY INFORMATION. DO YOU WISH TO INCLUDE THIS OPTION (YES OR NO) : (2.7.22)

FIP BUFFERING ? ##Y ##

IF THERE IS SUFFICIENT MEMORY AVAILABLE ON THIS SYSTEM, THE MOST FREQUENTLY USED NON-RESIDENT DISK HANDLING CODE CAN BE MADE RESIDENT IN ORDER TO PROVIDE BETTER SYSTEM PERFORMANCE. SHOULD THIS DISK PROCESSING BE DONE BY RESIDENT CODE (YES OR NO). (2.7.23)

RESIDENT DISK HANDLING ? **NO** YES

THE DISPATCH CODE AND TABLES FOR THE FIP SYSTEM FUNCTION CALLS (SYS CODE 6) ARE NORMALLY NON-RESIDENT. OVERALL SYSTEM PERFORMANCE CAN BE IMPROVED BY MAKING THIS CODE RESIDENT. DO YOU WANT THE SYS CALL DISPATCH CODE RESIDENT (YES OR NO)

RESIDENT SYS CALL DISPATCH ? ##NO##

THE ROUTINES WHICH IMPLEMENT THE SEND AND RECEIVE SYSTEM FUNCTION CALLS ARE ALSO NON-RESIDENT. IF YOUR APPLICATIONS REQUIRE INTER-JOB COMMUNICATION, YOU MAY WANT SEND/RECEIVE RESIDENT. PLEASE ANSWER YES OR NO :

RESIDENT SEND/RECEIVE ? ##NO##

NON-RESIDENT CODE IS USED TO LIST DISK DIRECTORIES (CATALOG COMMAND). IF YOU WANT THE DIRECTORY LISTER RESIDENT, TYPE "YES", OTHERWISE ANSWER "NO".

RESIDENT DIRECTORY LISTER ? ##NO##

```
THE CONCISE COMMAND LANGUAGE OPTION  EN-                    (2.7.24)
ABLES  EACH INSTALLATION TO DEFINE A SET
OF KEYBOARD COMMANDS WHICH CALL AND PASS
ONE COMMAND LINE TO COMMONLY USED SYSTEM
PROGRAMS.  DO YOU WISH TO INCLUDE THE CCL
OPTION IN THIS SYSTEM (YES OR NO) :

CCL ?                              ##Y ## YES

RSTS/E KITS INCLUDE ONE  STANDARD  TABLE
OF CCL COMMANDS.  THESE ARE LISTED IN THE
SYSTEM MANAGER'S GUIDE.  IF THIS SET  IS
ADEQUATE FOR YOUR NEEDS,  ANSWER  'YES';
OTHERWISE, ANSWER 'NEW' TO  CREATE  YOUR
OWN UNIQUE CCL COMMANDS, OR 'ADD' TO ADD
YOUR  OWN  UNIQUE  CCL  COMMANDS  TO THE
STANDARD TABLE :

STANDARD CCL TABLE ?.              ##Y ## ADD

TO CREATE A CCL ENTRY,  TYPE THE PROGRAM
NAME AND THE CCL COMMAND (SEPARATED BY A
COMMA).  THE PROGRAM NAME IS  1-6  ALPHA-
NUMERIC CHARACTERS;  THE CCL COMMAND MAY
BE UP TO 14 ALPHANUMERIC CHARACTERS.  UP
TO  20  CCL ENTRIES MAY BE DEFINED.  TYPE
"/E" TO END YOUR CCL TABLE.  TYPE "/R" TO
RESTART THE SERIES OF CCL QUESTIONS :

<PROGRAM>,<COMMAND> ?              ##??## INVENT,ONHAND

<PROGRAM>,<COMMAND> ?              ##??## INVENT,ONORDER

<PROGRAM>,<COMMAND> ?              ##??## INVENT,BACKLOG

<PROGRAM>,<COMMAND> ?              ##??## ORDER,ORDER

<PROGRAM>,<COMMAND> ?              ##??## ORDER,CANCEL

<PROGRAM>,<COMMAND> ?              ##??## ORDER,INQUIRE

<PROGRAM>,<COMMAND> ?              ##??## /END

CCL LISTING ON LP0: ?              ##NO##
```

```
FLOATING POINT NUMBERS ARE REPRESENTED                    (2.7.25)
INTERNALLY AS TWO 16-BIT WORDS, GIVING
SEVEN SIGNIFICANT DIGITS. IT IS POSSIBLE
TO MAINTAIN 17 SIGNIFICANT DIGITS BY US-
ING 4 WORDS PER NUMBER. THE FOUR WORD
MATH PACKAGES ALSO INCLUDE THE SCALED
ARITHMETIC FEATURE. WOULD THIS INSTAL-
LATION PREFER TO USE 2 OR 4 WORD MATH :

MATH PRECISION ?                    ##2 ## 4

IT IS POSSIBLE TO SAVE SPACE IN THE BA-                   (2.7.26)
SIC-PLUS SYSTEM BY OMITTING THE EXTENDED
FUNCTIONS SIN, COS, SQR, LOG, ETC., IF
THEY ARE NOT NEEDED. DOES THIS INSTAL-
LATION NEED TO COMPUTE THESE FUNCTIONS
(YES OR NO).

FUNCTIONS ?                         ##Y ##

THE RSTS/E SYSTEM NORMALLY REPORTS THE
TIME OF DAY AS "AM" OR "PM". IT IS
POSSIBLE TO HAVE 24-HOUR TIME REPORTED
INSTEAD (COMMON FOR EUROPEAN AND MILI-
TARY INSTALLATIONS). DO YOU WANT
"AM/PM" OR "24-HOUR" TIME REPORTED.

TIME FORMAT ?                       ##24## AM

SPECIAL OUTPUT FORMATTING CAN BE DONE                     (2.7.27)
USING THE "PRINT USING" STATEMENT.
WOULD THIS INSTALLATION LIKE TO HAVE
THIS OPTIONAL FEATURE (YES OR NO).

PRINT USING ?                       ##Y ##

BASIC-PLUS PERMITS THE USER TO OPERATE                    (2.7.28)
ON AN ENTIRE MATRIX USING JUST A SINGLE
STATEMENT. THESE STATEMENTS ARE THE
"MAT" STATEMENTS. WOULD THIS INSTALLA-
TION LIKE TO HAVE THIS OPTIONAL FEATURE.

MATRICES ?                          ##NO## YES

THE STAND-ALONE PROGRAM USED TO COPY                      (2.7.29)
DISKS/TAPES, ROLLIN, CAN BE INCLUDED IN
THE RSTS/E SYSTEM IMAGE. IT CAN THEN BE
LOADED FROM THE SYSTEM DISK USING THE
'LOAD' OPTION. WOULD THIS INSTALLATION
LIKE TO INCLUDE ROLLIN (YES OR NO).

ROLLIN ?                            ##Y ##
```

```
SYSGEN:IF YOU HAVE ANY SPECIAL REQUIREMENTS WHICH REQUIRE
SYSGEN:EDITING EITHER THE CONFIGURATION FILE (CONFIG.MAC)
SYSGEN:OR THE BATCH  GENERATION  FILE (SYSGEN.BAT), ABORT
SYSGEN:NOW BY TYPING "CONTROL/C" AND THEN "TE".  RESUME AT
SYSGEN:THIS  POINT BY TYPING  "BATCH SYSGN2".  OTHERWISE,
SYSGEN:TYPE "CO" TO CONTINUE WITH SYSTEM GENERATION :

A050 000000
$CO

SYSGEN:CHECKPOINT TWO:  IF ANY PROBLEMS DEVELOP DURING THE     (2.5.2)
SYSGEN:BATCH GENERATION PROCESS WHICH FOLLOWS, SYSGEN MAY
SYSGEN:BE ABORTED BY TYPING "CONTROL/C" AND "TE".  RESTART
SYSGEN:FROM THIS POINT BY TYPING "BATCH SYSGN2".

$CH SYSGEN.BAT                                                  (2.8)
$RUN PIP
$RUN MACRO
$RUN MACRO
$RUN PIP

SYSGEN:REMOVE DEC-11-ORSPA-C-XXX FROM MAGTAPE UNIT 0 TEMPORARILY
SYSGEN:MOUNT  DEC-11-ORCOA-C-MC9 OR MC7 (2780 PACKAGE) ON UNIT 0
SYSGEN:WITH NO "WRITE RING" AND SET TO "ONLINE".
SYSGEN:TYPE "CO" WHEN READY

A050 000000
$CO

$RUN PIP

SYSGEN:REMOVE  DEC-11-ORCOA-C-MC9 OR MC7 FROM MAGTAPE DRIVE 0 AND
SYSGEN:REMOUNT DEC-11-ORSPA-C-XXX ON MAGTAPE DRIVE 0
SYSGEN:WITH NO "WRITE RING" AND SET TO "ONLINE".
SYSGEN:TYPE "CO" WHEN READY

A050 000000
$CO

$RUN PIP
$RUN MACRO
$RUN EDIT
$RUN MACRO
$RUN PIP
$RUN PIP
$RUN LINK

$RUN PIP
$RUN LINK

$RUN PIP
$RUN LINK

$RUN PIP
$RUN CILUS

SYSGEN:REMOVE DEC-11-ORSPA-C-XXX AND MOUNT A NEW MAGTAPE
SYSGEN:WITH "WRITE RING" ON UNIT 0.  SET TO "ON LINE" AND
SYSGEN:TYPE "CO" WHEN READY

A050 000000
$CO
```

```
$RUN PIP

SYSGEN:WHEN SYSLOD IDENTIFIES ITSELF,
SYSGEN:MOUNT A NEW DISK ON UNIT RP UNIT 0,
SYSGEN:READY AND WRITE ENABLE THE DRIVE.
SYSGEN:RESPOND AS FOLLOWS TO THE PROMPTING:
SYSGEN:CONSOLE FILL COUNT = <RETURN>
SYSGEN:DATE:                <DD-MMM-YY>
SYSGEN:DIALOGUE?            <RETURN>
SYSGEN:#DP:/NS:512:25/FO/TO:4/HO/BO<MT:RSTS.LCL
$RUN CILUS

SYSLOD V08-01A
CONSOLE FILE COUNT=
DATE: 20-DEC-74
DIALOGUE?

#DP:/NS:512:25/TO:4/HO/BO<MT:RSTS.LCL

SYSLOD COMPLETE



RSTS V05C-01 SYSTEM #880                                    (Chapter 3)

OPTION:
```

2.6.2 Disk Cartridge Software Using Short Form Questions with Automatic Answers

Bootstrap the system generation disk cartridge using the hardware loader.

```
DOS/BATCH V9-20C                                                    (2.3.1)
DATE: 20-DEC-74                                                     (2.3.2)
TIME: 10:00
DIALOGUE?

$LO 1,1
DATE:-20-DEC-74
TIME:-10:00:20

$RUN CILUS
CILUS V08-06A

#ROLLIN.CIL/BO

ROLLIN V07

DK1:/FO<DK0:/VE

STARTING RK FORMAT PASS
END RK FORMAT PASS
STARTING VERIFICATION PASS

#DK1:/FO<DK0:/VE

STARTING RK FORMAT PASS
END RK FORMAT PASS
STARTING VERIFICATION PASS

#/BOOT:DK:

DOS/BATCH V9-20C
DATE:
```

```
DOS/BATCH V9-20C                                                    (2.3.3)
DATE: 20-DEC-74
TIME: 10:00
DIALOGUE?

$LO 1,1
DATE:-20-DEC-74
TIME:-10:00:20

$RUN CILUS
CILUS V08-06A

#SYSLOD.CIL/BO

SYSLOD V08-01A
CONSOLE FILL COUNT=4
DATE: 20-DEC-74
DIALOGUE?

#DB0:/FO
#DB1:/FO


#DP1:/FO

REMOVE THIRD PANEL BELOW RP11C INDICATORS
AND SET THE LEFT HAND SWITCH TO FORMAT ENABLE: READY? YES
RESET SWITCH TYPE YES WHEN DONE YES
ARE YOU SURE? YES

#DK0:/FO
#DK1:/FO
```

```
DOS/BATCH V9-20C                                                    (2.4.1)
DATE: 20-DEC-74
TIME: 10:00
DIALOGUE? YES
DO YOU WANT TO RESET CONSOLE FILL COUNT? YES
FILL COUNT=4
ARE ANY DEVICES DOWN? NO
DO YOU WANT TO CHANGE LINE PRINTER? YES
LS11? NO
HOW MANY COLUMNS ? 132
LOWER CASE? NO
OVERPRINT? NO
DO YOU WANT TO CHANGE CARD READER DEFAULTS? NO
HAVE YOU GOT RK02 DISK? NO

$LO 1,1                                                             (2.4.2)

DATE:-20-DEC-74
TIME:-10:00:20

$BATCH SYSGEN                                                       (2.4.4)


$JOB SYSGEN[1,1]
TIME:-10:00:30
$CH SYSGN3

SYSGEN:CHECKPOINT ONE:  DURING THE CONFIGURATION DIALOGUE           (2.5.2)
SYSGEN:WHICH FOLLOWS, YOU MAY ABORT SYSGEN BY TYPING
SYSGEN:"CONTROL/C" AND THEN "TE". RESTART FROM THE BEGIN-
SYSGEN:NING OF THE DIALOGUE BY TYPING "BATCH SYSGN1".
$RUN SYSGEN
```

```
**20-DEC-74**

BEGINNING OF RSTS/E  SYSTEM  GENERATION.

QUESTIONS COME IN LONG AND SHORT  FORMS.                    (2.5.1)
IF  YOU  ARE  FAMILIAR WITH THEM, ANSWER
"S" FOR SHORT; OTHERWISE, ANSWER "L" FOR
LONG FORM.  APPEND "/A" TO ENABLE  AUTO-
MATIC ANSWERS. FORM ?  S/A

MEDIA ? RK

FPP (11/45) ?                  **NO**                       (2.7.1)

FIS (11/40) ?                  **NO**

CLOCK ?                        **P *** C                    (2.7.2)

KW11P INTERRUPT RATE ?         ##100##                      (2.7.4)

CONSOLE TYPE ?                 ##LA30S. ## VT05B            (2.7.5)

KL11,LC11,DL11A,DL11B'S ?      **02**                       (2.7.5.1)

DL11C, DL11D'S ?               **00**

DC11'S ?                       **00**

DL11E'S ?                      **01**

DJ11'S ?                       **00**

DH11'S ?                       **01**                       (2.7.5.2)

DH11 UNIT 00 TYPE ?            ##AA##

DOES DH11 00 INCLUDE A DM11-BB? ##??## YES

DH11 UNIT 00 LINES ENABLED ?   ##16##

PSEUDO KEYBOARDS ?             ##04##                       (2.7.6)

2741'S ?                       ##NO## DH                    (2.7.7)

2741 CODE(S) ?                 ##CORR. ## C360              (2.7.8)

MULTI-TTY SERVICE ?            ##Y ##                       (2.7.9)
```

```
RS03'S ?                          **00**                    (2.7.10)

RS04'S ?                          **00**

RF/RS11'S ?                       **01**

RP04'S ?                          **00**

THE  RJP04  DISK  SYSTEM  CONSISTS OF AN
RH11 CONTROLLER AND UP TO EIGHT  OF  THE
RP04  40 MILLION WORD  MOVING-HEAD  DISK
PACK DRIVES.  HOW  MANY  RP04 DRIVES ARE
INCLUDED  IN THIS CONFIGURATION (0 TO 8)
NOTE:  IF YOU  HAVE  TWO  OR  MORE  RP04
DRIVES BUT PREFER  TO CONSERVE MEMORY BY
USING THE NON  OVERLAPPED  SEEK  DRIVER,
APPEND "/NO" TO THE NUMBER OF DRIVES :

RP04'S ?                          **00**

RP02/RP03'S ?                     **01**

RK03/RK05'S ?                     ##08## 2

SYSTEM DISK ?                     **RP**

TU16'S ?                          **00**                    (2.7.11)

TU10'S ?                          ##08## 2                  (2.7.19)

DECTAPES ?                        ##08## 4

PRINTERS ?                        **01**

LP0: TYPE ?                       **LP**

LP0: WIDTH ?                      **132**

LP0: LOWER CASE ?                 ##NO## YES

LP0 FOR SYSGEN ?                  **Y **                    (2.7.12)

LISTINGS ?                        ##NO##                    (2.7.13)

CARD READER ?                     **CR**

CARD DECODE ?                     ##029## 026               (2.7.14)

P. T. READER ?                    **Y **

P. T. PUNCH ?                     **Y **
```

```
DQ11'S ?                                **00**

THE DQ11  NPR SYNCHRONOUS LINE INTERFACE
IS NOT SUPPORTED UNDER RSTS/E.  HOWEVER,
TO DETERMINE THE CORRECT FLOATING DEVICE
ADDRESSES  FOR OTHER SUPPORTED HARDWARE,
WE NEED TO KNOW HOW MANY DQ11'S ARE CON-
NECTED TO THIS SYSTEM (0 TO 16) :

DQ11'S ?                                **00**

DP11'S ?                                **02**

DU11'S ?                                **01**

2780 ?                                  ##NO##

THE  RSTS/E 2780  PACKAGE  EMULATES  THE                    (2.7.15)
OPERATION  OF  THE IBM 2780 MODEL 1 DATA
TRANSMISSION TERMINAL.  THE PACKAGE PER-
MITS  THE RSTS/E  SYSTEM  TO COMMUNICATE
WITH ANY IBM SYSTEM  WHICH SUPPORTS  THE
DEVICE, OR WITH ANOTHER RSTS 2780 SYSTEM.
THE  2780 PACKAGE  IS SOLD AND SUPPORTED
BY THE DEC  COMMUNICATIONS GROUP.  IT IS
NOT INCLUDED IN THE STANDARD RSTS/E KIT.
DO YOU HAVE THE 2780 PACKAGE AND WISH TO
INCLUDE IT IN THIS SYSTEM (YES OR NO) :

2780 ?                                  ##NO## YES

2780 INTERFACE ?                        ##DU## DP11

NON-SUPPORTED DEVICES ?                 ##Y ## YES          (2.7.16)

DM11-A'S ?                              **00**

DN11-DA'S ?                             **00**

DR11-A, C'S ?                           **00**

PA611R'S ?                              **00**

PA611P'S ?                              **00**

DT03-FP'S ?                             **00**

DX11'S ?                                **00**

GT40 (1 ONLY) ?                         **00**

LPS (1 ONLY) ?                          **01**

KW11W (1 ONLY) ?                        **00**
```

```
THE INSTALLATION NAME :          ##SYSTEM TEST## SYSTEM #1136

MAXIMUM JOBS ?                   ##20## 24                          (2.7.17)

SMALL BUFFERS ?                  ##200## 240                        (2.7.18)

BIG BUFFERS ?                    ##01##                             (2.7.19)

RECEIVERS ?                      ##08## 6                           (2.7.20)

POWER FAIL ?                     ##Y ##                             (2.7.21)

FIP BUFFERING ?                  ##Y ##                             (2.7.22)

AN OPTIONAL FEATURE OF THE RSTS/E SYSTEM
ALLOWS  THE FILE  PROCESSOR (FIP) TO USE
FREE SMALL OR BIG BUFFERS  TO STORE DISK
DIRECTORY INFORMATION.  THIS IMPROVES THE
SPEED OF DIRECTORY  PROCESSING BY AVOID-
ING THE NECESSITY OF  REREADING COMMONLY
USED DIRECTORY INFORMATION.  DO YOU WISH
TO INCLUDE THIS OPTION (YES OR NO) :

FIP BUFFERING ?                  ##Y ##

RESIDENT DISK HANDLING ?         **NO** YES                         (2.7.23)

RESIDENT SYS CALL DISPATCH ?     ##NO##

RESIDENT SEND/RECEIVE ?          ##NO##

RESIDENT DIRECTORY LISTER ?      ##NO##

CCL ?                            ##Y ## YES                         (2.7.24)

STANDARD CCL TABLE ?             ##Y ## YES

CCL LISTING ON LP0: ?            ##NO##

MATH PRECISION ?                 ##2 ## 4                           (2.7.25)

FUNCTIONS ?                      ##Y ##                             (2.7.26)

TIME FORMAT ?                    ##24## AM

PRINT USING ?                    ##Y ##                             (2.7.27)

MATRICES ?                       ##NO## YES                         (2.7.28)

ROLLIN ?                         ##Y ##                             (2.7.29)
```

```
SYSGEN:IF YOU HAVE ANY SPECIAL REQUIREMENTS WHICH REQUIRE
SYSGEN:EDITING EITHER THE CONFIGURATION FILE (CONFIG.MAC)
SYSGEN:OR THE BATCH  GENERATION  FILE (SYSGEN.BAT), ABORT
SYSGEN:NOW BY TYPING "CONTROL/C" AND THEN "TE".  RESUME AT
SYSGEN:THIS  POINT BY TYPING  "BATCH SYSGN2".  OTHERWISE,
SYSGEN:TYPE "CO" TO CONTINUE WITH SYSTEM GENERATION :
A050 000000

$CO

SYSGEN:CHECKPOINT TWO: IF ANY PROBLEMS DEVELOP DURING THE          (2.5.2)
SYSGEN:BATCH GENERATION PROCESS WHICH FOLLOWS, SYSGEN MAY
SYSGEN:BE ABORTED BY TYPING "CONTROL/C" AND "TE".  RESTART
SYSGEN:FROM THIS POINT BY TYPING "BATCH SYSGN2".
$CH SYSGEN.BAT

S257 000002

SYSGEN:MOUNT DEC-11-ORCOA-C-HC DISK (2780 PACKAGE) ON RK DRIVE 1. (2.8)
SYSGEN:READY AND WRITE PROTECT THE DRIVE.
SYSGEN:TYPE "CO" WHEN READY
A050 000000

$CO

$RUN MACRO
$RUN MACRO
$RUN PIP

SYSGEN:REMOVE DEC-11-ORCOA-C-HC FROM RK DRIVE 1

$RUN MACRO
$RUN LINK
$RUN LINK
$RUN LINK
$RUN PIP
$RUN CILUS

SYSGEN:MOUNT A FORMATTED RK CARTRIDGE ON DRIVE 1
SYSGEN:SET TO "RUN" AND WRITE ENABLE
SYSGEN:TYPE 'CO' WHEN READY TO LOAD THE NEW SYSTEM
A050 000000

$CO

$RUN CILUS

SYSGEN:REMOVE   DEC-11-ORSPA-B-HC  FROM  RK DRIVE 0
SYSGEN:MOVE THE RSTS/E SYSTEM DISK FROM  RK DRIVE 1
SYSGEN:TO DRIVE 0.  BOOT NORMALLY TO BRING UP RSTS/E
$FI
TIME:-11:04:20

DOS/BATCH V09-20C
$


RSTS V05C-01 SYSTEM #1136                                  (Chapter 3)

OPTION:
```

2.6.3 DECtape Software Using Short Form Questions

Bootstrap the DECtape using the hardware loader.

```
SYSLOD V08-01A                                                    (2.2.2)
CONSOLE FILL COUNT=0
DATE: 20-DEC-74                                                   (2.2.3)
DIALOGUE?                                                         (2.2.3.1)

#DF:MONLIB.CIL/CO:0/HO/BO<DT0:MONLIB.LCL

SYSLOD COMPLETE

ANSWER WITH CARRET OR 'Y'CARRET:- IS YOUR LINE FREQUENCY 50 HERTZ?
DO YOU WANT TO DISABLE DIALOGUE FOREVER? NO

DOS/BATCH V9-20C
DATE: 20-DEC-74                                                   (2.4.1)
TIME: 12:30
DIALOGUE? YES
DO YOU WANT TO RESET CONSOLE FILL COUNT? YES
FILL COUNT=0
ARE ANY DEVICES DOWN? NO
DO YOU WANT TO CHANGE LINE PRINTER? YES
LS11? YES
OVERPRINT? NO
HAVE YOU GOT RK02 DISK? NO

$LO 1,1                                                           (2.4.2)

DATE:-20-DEC-74
TIME:-12:30:25

$RUN DT0:PIP                                                      (2.4.3)

PIP V10-02

#SY:<DT1:SYSGEN

#^C
.KI

$BATCH SYSGEN                                                     (2.4.4)

$JOB SYSGEN[1,1]
TIME:-12:31:46
$RUN DT0:PIP
$RUN LINK

$RUN PIP
$CH SYSGN3
```

```
SYSGEN:CHECKPOINT ONE:  DURING THE CONFIGURATION DIALOGUE          (2.5.2)
SYSGEN:WHICH FOLLOWS, YOU MAY ABORT SYSGEN BY TYPING
SYSGEN:"CONTROL/C" AND THEN "TE". RESTART FROM THE BEGIN-
SYSGEN:NING OF THE DIALOGUE BY TYPING "BATCH SYSGN1".
$RUN SYSGEN

**20-DEC-74**

BEGINNING OF RSTS/E SYSTEM GENERATION.

QUESTIONS COME IN LONG AND SHORT FORMS.                            (2.5.1)
IF YOU ARE FAMILIAR WITH THEM, ANSWER
"S" FOR SHORT; OTHERWISE, ANSWER "L" FOR
LONG FORM.  APPEND "/A" TO ENABLE AUTO-
MATIC ANSWERS.  FORM ?  S

MEDIA ? DT

FPP (11/45) ? YES                                                 (2.7.1)

CLOCK ? L                                                         (2.7.2)

AC FREQ ? 60                                                      (2.7.3)

CONSOLE TYPE ? LA30                                               (2.7.5)

KL11,LC11,DL11A,DL11B'S ? 4                                       (2.7.5.1)

DL11C, DL11D'S ? 0

DC11'S ? 0

DL11E'S ? 4

DJ11'S ? 0

DH11'S ? 0                                                        (2.7.5.2)

PSEUDO KEYBOARDS ? 4                                              (2.7.16)

2741'S ? NO                                                       (2.7.7)

MULTI-TTY SERVICE ? YES                                           (2.7.9)
```

```
RS03'S ?

THE RJS04/RJS03  DISK SYSTEM CONSISTS OF                    (2.7.10)
AN RH11 DEVICE CONTROLLER  AND  FROM ONE
TO EIGHT RS03 (256K) OR RS04 (512K WORD)
FIXED-HEAD DISK DRIVES.  RSTS/E SUPPORTS
ANY  COMBINATION OF RS03'S AND RS04'S UP
TO A TOTAL OF FOUR DRIVES  FOR  SWAPPING
AND  NON-RESIDENT CODE (ADDITIONAL UNITS
MAY BE CONNECTED TO THE  RH11, HOWEVER).
PLEASE ENTER THE NUMBER OF EACH  TYPE OF
DRIVE (0 TO 4) :

RS03'S ? 0

RS04'S ? 0

RF/RS11'S ? 2

RP04'S ? 0

RP02/RP03'S ? 0

RK03/RK05'S ? 2

SYSTEM DISK ? RF

TU16'S ? 0                                                  (2.7.11)

TU10'S ? 0

DECTAPES ? 2

PRINTERS ? 1

LP0: TYPE ? LS

LP0 FOR SYSGEN ?                                            (2.7.12)

IF THERE IS A LINE PRINTER AVAILABLE FOR
USE DURING SYSGEN (MUST BE LP0), IT WILL
BE USED TO PRINT THE  SYSTEM  LOAD  MAPS
AND, OPTIONALLY, TO PRINT ASSEMBLY LIST-
INGS OF THE SYSTEM TABLES  AND  TERMINAL
SERVICE  MODULES.  IF YOU HAVE A PRINTER
AND WANT THE  ASSEMBLY  LISTINGS  AND/OR
LOAD MAPS  PRINTED DURING SYSGEN, ANSWER
"YES", OTHERWISE ANSWER "NO".

LP0 FOR SYSGEN ? YES

LISTINGS ? NO                                               (2.7.13)

CARD READER ? NO

P.T.READER ? NO
```

```
DQ11'S ? 0

DP11'S ?

ONE DP11 OR DU11 SYNCHRONOUS LINE INTER-
FACE  IS REQUIRED FOR  OPERATION  OF THE
RSTS/E 2780 PACKAGE (IF INCLUDED BELOW).
ADDITIONAL DP11  AND DU11 INTERFACES MAY
BE  CONNECTED  TO THE  UNIBUS  FOR OTHER
PURPOSES.  PLEASE  SPECIFY THE NUMBER OF
THESE DEVICES  ON THIS SYSTEM (INCLUDING
ANY TO BE USED BY THE 2780 PACKAGE) :

DP11'S ? 1

DU11'S ? 2

2780 ? YES                                          (2.7.15)

2780 INTERFACE ?

THE 2780 PACKAGE REQUIRES EITHER A  DP11
OR  DU11  SYNCHRONOUS LINE INTERFACE AND
THE  KG11-A  (COMMUNICATIONS  ARITHMETIC
OPTION).  WHICH TYPE OF SYNCHRONOUS LINE
INTERFACE WILL  YOUR  2780  PACKAGE  USE
(DP11 OR DU11) :

2780 INTERFACE ? DU11

NON-SUPPORTED DEVICES ? YES                         (2.7.16)

DM11-A'S ? 0

DN11-DA'S ? 0

DR11-A,C'S ? 0

PA611R'S ? 0

PA611P'S ? 0

DT03-FP'S ? 0

DX11'S ? 0

GT40 (1 ONLY) ? 0

LPS (1 ONLY) ? 1

KW11W (1 ONLY) ? 0
```

```
THE INSTALLATION NAME : SYSTEM #372
MAXIMUM JOBS ? 24                                   (2.7.17)
SMALL BUFFERS ? 240                                 (2.7.18)
BIG BUFFERS ? 1                                     (2.7.19)
RECEIVERS ? 6                                       (2.7.20)
POWER FAIL ? YES                                    (2.7.21)
FIP BUFFERING ? YES                                 (2.7.22)
RESIDENT DISK HANDLING ? YES                        (2.7.23)
RESIDENT SYS CALL DISPATCH ? NO
RESIDENT SEND/RECEIVE ? NO
RESIDENT DIRECTORY LISTER ? NO
CCL ? YES                                           (2.7.24)
STANDARD CCL TABLE ? NEW
<PROGRAM>,<COMMAND> ? PIP,PIP
<PROGRAM>,<COMMAND> ? QUE,QUE
<PROGRAM>,<COMMAND> ? PIP,HELP
<PROGRAM>,<COMMAND> ? TTYSET,SET
<PROGRAM>,<COMMAND> ? /E
CCL LISTING ON LP0: ? NO
MATH PRECISION ? 4                                  (2.7.25)
FUNCTIONS ? Y                                       (2.7.26)
TIME FORMAT ? AM
PRINT USING ? YES                                   (2.7.27)
MATRICES ? NO                                       (2.7.28)
ROLLIN ? YES                                        (2.7.29)
```

```
SYSGEN:IF YOU HAVE ANY SPECIAL REQUIREMENTS WHICH REQUIRE
SYSGEN:EDITING EITHER THE CONFIGURATION FILE (CONFIG.MAC)
SYSGEN:OR THE BATCH  GENERATION  FILE (SYSGEN.BAT), ABORT
SYSGEN:NOW BY TYPING "CONTROL/C" AND THEN "TE".  RESUME AT
SYSGEN:THIS  POINT BY TYPING  "BATCH SYSGN2".  OTHERWISE,
SYSGEN:TYPE "CO" TO CONTINUE WITH SYSTEM GENERATION :

A050 000000
$CO

SYSGEN:CHECKPOINT TWO:  IF ANY PROBLEMS DEVELOP DURING THE      (2.5.2)
SYSGEN:BATCH GENERATION PROCESS WHICH FOLLOWS, SYSGEN MAY
SYSGEN:BE ABORTED BY TYPING "CONTROL/C" AND "TE".  RESTART
SYSGEN:FROM THIS POINT BY TYPING "BATCH SYSGN2".
$CH SYSGEN.BAT

SYSGEN:REMOVE DEC-11-ORSBA-C-UC1 FROM DECTAPE DRIVE 0          (2.8)
SYSGEN:REMOVE DEC-11-ORSBA-C-UC2 FROM DECTAPE DRIVE 1
SYSGEN:MOUNT  DEC-11-ORSEA-C-UC1  ON  DECTAPE DRIVE 0
SYSGEN:MOUNT  DEC-11-ORCOA-C-UC   ON  DECTAPE DRIVE 1 (2780 PACKAGE)
SYSGEN:SET TO "REMOTE" AND "WRITE LOCKED"
SYSGEN:TYPE "CO" WHEN READY

A050 000000
$CO

$RUN PIP
$RUN MACRO
$RUN MACRO
$RUN PIP
$RUN PIP
$RUN MACRO
$RUN EDIT
$RUN MACRO
$RUN PIP

SYSGEN:REMOVE DEC-11-ORSEA-C-UC1 FROM DECTAPE DRIVE 0
SYSGEN:REMOVE DEC-11-ORCOA-C-UC  FROM DECTAPE DRIVE 1
SYSGEN:MOUNT  DEC-11-ORSEA-C-UC2  ON  DECTAPE DRIVE 0
SYSGEN:MOUNT  DEC-11-ORSEA-C-UC3  ON  DECTAPE DRIVE 1
SYSGEN:SET BOTH TO "REMOTE" AND "WRITE LOCKED"
SYSGEN:TYPE "CO" WHEN READY

A050 000000
$CO

$RUN PIP
$RUN LINK

$RUN PIP
$RUN PIP
$RUN LINK

$RUN PIP
$RUN PIP
$RUN LINK

$RUN PIP
$RUN PIP
```

```
SYSGEN:REMOVE DEC-11-ORSEA-C-UC2 FROM DECTAPE DRIVE 0
SYSGEN:REMOVE DEC-11-ORSEA-C-UC3 FROM DECTAPE DRIVE 1
SYSGEN:MOUNT A NEW FORMATTED DECTAPE ON UNIT 0
SYSGEN:SET TO "REMOTE" AND "WRITE ENABLED"
SYSGEN:TYPE "CO" WHEN READY

A050 000000
$CO

$RUN PIP
$RUN CILUS
$RUN PIP

SYSGEN:WHEN SYSLOD IDENTIFIES ITSELF,
SYSGEN:RESPOND AS FOLLOWS TO THE PROMPTING:
SYSGEN:CONSOLE FILL COUNT = <RETURN>
SYSGEN:DATE:                <DD-MMM-YY>
SYSGEN:DIALOGUE?            <RETURN>
SYSGEN:#DF:/NS:256:49/TO:4/HO/BO<DT:RSTS.LCL

$RUN CILUS

SYSLOD V08-01A
CONSOLE FILE COUNT=
DATE: 20-DEC-74
DIALOGUE?

#DF:/NS:256:49/TO:4/HO/BO<DT:RSTS.LCL

SYSLOD COMPLETE



RSTS V05C-01 SYSTEM #372                                    (Chapter 3)

OPTION:
```

2.7 CONFIGURATION QUESTION CONSIDERATIONS

The questions printed during the SYSGEN dialogue concern the hardware configuration parameters and software options. Those parameters and options requiring more explanation than available in the long form of the question are explained in this section. The explanations appear in the order in which the program prints the related questions. Appendix D of this manual lists the memory requirements for most hardware and software available.

2.7.1 Central Processor Unit and Options

The RSTS/E system runs on either a PDP-11/45 or a PDP-11/40 central processor. Either processor can optionally include floating point hardware to reduce the execution time for floating point calculations. The PDP-11/45 Floating Point Processor (FPP) provides both 2-word and 4-word floating point operations in the hardware. The PDP-11/40 Floating Instruction Set (FIS) provides 2-word floating point operations in the hardware.

The answers to the FPP, FIS, MATH PRECISION, and FUNCTION questions determine which mathematical package is included in the BASIC-PLUS Run Time System. See Section 2.7.25 for a description of the different mathematical packages.

2.7.2 Clocks

The RSTS/E system can operate with one of two types of system clocks. The KW11-L Line Time Clock divides time into intervals based on the line frequency of the power source - either 50 Hz or 60 Hz. (The designation Hertz (Hz) is the international standard of measurement for cycles per second.) The KW11-P Programmable Real Time Clock can also employ the line frequency of the power source. However, the KW11-P has a crystal-controlled oscillator which can be used as the system time base independent of the power source frequency. The crystal operation KW11-P is beneficial in areas where the line frequency of the power source is not constant.

The answer to the CLOCK question determines which type of clock RSTS/E will use and, for the KW11-P, which mode of operation (line frequency or crystal) is desired. If line frequency is used for either clock, SYSGEN prints the AC FREQ question and omits the KW11-P

INTERRUPT RATE question. If the crystal mode of operation is indicated, SYSGEN omits the AC FREQ question but prints the KW11-P INTERRUPT RATE question.

2.7.3 AC Power Frequency

The PDP-11 computer requires alternating current (AC) power input and is able to run on 60 Hz frequency, as is standard in the United States, or on 50 Hz frequency, as is standard in many other countries. If the computer operates with the KW11-P clock at line frequency or with the KW11-L clock, SYSGEN prints the AC FREQ question. The answer to the question must be either 50 or 60, depending on the standard frequency of the power input.

2.7.4 KW11-P Interrupt Rate

The KW11-P 10 KHz crystal-controlled oscillator allows time to be divided to a desired interrupt rate given as times per second. 100 is the recommended interrupt rate for all RSTS/E systems.

2.7.5 Terminals and Terminal Interfaces

2.7.5.1 Keyboard Numbers and Interfaces - The RSTS/E system is designed to handle a maximum of 64 terminals. Each terminal is assigned a keyboard number ranging from 0 to 63. The console terminal is given the keyboard number 0 on all RSTS/E systems and is referenced by the device designator KB0:. SYSGEN requires the user to enter the type of console terminal so that the RSTS/E terminal service and the terminal drivers in the initialization code and in ROLLIN can add the correct number of fill characters.

The assignment of a keyboard number, other than that of the console terminal, is determined by the type of line interface to which the terminal is attached. The local installation can have any combination of local and remote line interfaces as long as the total number of terminal lines and pseudo keyboards does not exceed 63, not including the console terminal. The assignment is important because the system manager must specify the terminal speeds allowed on each of the variable speed lines in the system based on the keyboard number. (See the description of the TTYSET.SPD file in Chapter 6.)

The order in which RSTS/E assigns keyboard numbers is as follows: the system console terminal; all KL11, LC11, DL11A, and DL11B lines; DL11C and DL11D lines; PK (pseudo keyboards); DC11 (remote dial); DL11E (remote dial); and DH11 lines in increasing order of unit number and increasing order of configured (enabled) lines within each unit.

2.7.5.2 Interface Considerations - The answers to the configuration questions concerning the number of each type of terminal interface must accurately reflect the hardware configuration. For example, DL11C and DL11D interfaces are similar in construction and operation to the DL11A and DL11B interfaces. However, the interfaces have different ranges of UNIBUS addresses. Because of this similarity, improper configuration of the terminal interfaces at both the hardware and software level is a common problem for new installations. It is recommended that the system manager rely on the automatic answer facility to verify the terminal interface configuration.

If the number of DH11 multiplexers specified is non-zero, SYSGEN prints several questions for each unit. The first question requires that the type of DH11 be specified. The second question is printed only when the type of DH11 is capable of employing a separate DM11-BB modem control. The final question requests the number of lines to enable on the DH11 unit.

The DH11-AA and DH11-AC multiplexers are identical except for power requirements. The DH11-AA and DH11-AC can be wired for mixtures of automatic answer, local EIA, and local 20-milliampere lines. Both types require the DM11-BB modem control multiplexer if automatic answer data sets are to be connected to either unit. Because the DM11-BB modem control multiplexer is optional on the DH11-AA and DH11-AC, SYSGEN inquires about its presence.

The DH11-AB multiplexer is identical to the DH11-AA but is wired for telegraph and telex lines. Modem control is meaningless for telegraph service and SYSGEN does not print the question concerning the DM11-BB. RSTS/E does not provide specific support for telegraph or telex line disciplines. The DH11-AB is treated as a DH11-AA without modem control.

The DH11-AD and DH11-AE multiplexers are functionally identical to a DH11-AA unit. The DH11-AD multiplexer includes modem control on all 16 lines and therefore implements 16 EIA/CCITT (RS232-C) compatible lines with full data set control. This modem control is functionally identical to the DM11-BB. The DH11-AE multiplexer is wired for 16 EIA local lines without modem control. Neither unit allows the mixture of line types as the DH11-AA and DH11-AC units do. For DH11-AD and DH11-AE multiplexers, SYSGEN does not print the question concerning the DM11-BB.

The answers concerning the type of DH11 and the presence of a DM11-BB must accurately reflect the hardware ordering of the devices. The DH11 and the DM11-BB both have floating vectors; the DH11 is also a floating address device. The presence or absence of floating address and vector devices can affect the correct address and vector assignments of other hardware. The SYSGEN unit numbering assumes increasing UNIBUS addresses. For example, if DH11 unit ØØ has base address xxxxxx depending on other floating address devices present, unit Ø1 has the address assignment xxxxxx+2Ø.[1]

DM11-BB units, like DH11 units, are numbered according to increasing UNIBUS addresses. DM11-BB addressing begins at 77Ø5ØØ for unit Ø, is followed by unit 1 at 77Ø51Ø, and continues in increments of 1Ø (octal). DM11-BB interrupt vectors are assigned in the floating vector space in the order of increasing unit numbers.

For RSTS/E to operate properly, the lowest numbered DM11-BB unit must be connected to the lowest numbered DH11 unit which is to provide modem control. The next higher DM11-BB unit must be connected to the next higher DH11 unit which provides modem control. The sequence continues for any remaining DM11-BB units. For example, DM11-BB unit Ø can be connected to DH11 unit 2 provided that DH11 units Ø and 1 do not include modem control. By the same reasoning, connecting DM11-BB unit 2 to DH11 unit Ø is always illegal.

[1]For more information on floating address and vector assignment, see Appendix B of the RSTS/E System Reliability Test manual.

If automatic answers are enabled, the SYSGEN program counts the number of DM11-BB modem control multiplexers found on the system. SYSGEN can not determine which DM11-BB is attached to which DH11. Therefore, the indeterminate answer (**??**) is given for the DM11-BB question. The system manager must determine from the hardware configuration which DH11 units include a DM11-BB and he must report the presence or absence of the unit in response to the DM11-BB question. If automatic answers are enabled, typing either the RETURN or LINE FEED key in response to the DM11-BB question causes SYSGEN to print the number found. SYSGEN verifies that the number found equals the number reported present. If the two numbers are not equal, SYSGEN prints a question allowing the system manager to reconsider his answers. If he does reconsider, SYSGEN restarts the terminal-related questions. Otherwise, SYSGEN uses the number of DM11-BB units reported present (not the number found) and continues the dialogue.

The last question for each DH11 unit is the number of lines to enable.[1] SYSGEN permits a total of 63 single line interfaces, enabled DH11 lines, and pseudo keyboards. (The console keyboard is always enabled.) If more than 63 lines are configured, SYSGEN repeats all the terminal related questions.

Each line of a DH11 need not be enabled. Therefore, RSTS/E can be configured for less than the maximum number of 16 lines physically present on each DH11. Less than 16 lines are enabled when memory must be conserved or when the limit of 63 lines must not be exceeded. For example, if four DH11 units are present, at most 63 of the 64 possible lines can be enabled. If N lines (where N is less than 16) are enabled for a DH11, SYSGEN does not enable lines numbered N through 15 on the DH11. Lines not enabled will not be available for use and will be treated as nonexistent. Thus, when 10 lines are enabled, lines Ø through 9 will be operational and lines 1Ø through 15 on that DH11 cannot be used without regenerating the system.

If possible, all DH11 lines should be enabled during the system generation so that unused lines can be enabled in the future without regenerating the system. The SYSGEN questions do not specify which enabled DH11 lines are actually to be used or which lines are to have modem control enabled. The SYSGEN questions are supplemented by the SETKEY initialization option which allows any keyboard line to be temporarily disabled and which must be used to enable modem control.

[1] The SETKEY initialization option also enables DH11 lines but only those lines configured (enabled) at SYSGEN time. Refer to Section 3.4A for operational details.

2.7.6 Pseudo Keyboards

Any number of pseudo keyboards (up to 63) can be configured into the system. Each copy of the BATCH system program requires one pseudo keyboard to run user jobs. If the installation plans to run several copies of BATCH simultaneously, at least that number of pseudo keyboards must be specified. One additional pseudo keyboard is necessary for each application to be implemented using the pseudo keyboard feature described in Section 12.7 of the BASIC-PLUS Language Manual.

2.7.7 2741 Terminals

RSTS/E supports 2741 compatible terminals connected to DL11D, DL11E, and DC11 single line interfaces or a DH11 multiplexer. 2741 terminals normally use the RS232 EIA standard connection; hence, they can be connected locally to the computer through null modems to any of the above named interfaces or can be connected through data sets or acoustic couplers for operation over telephone lines.

2741 support can be included for single line interfaces only (DL11D, DL11E or DC11); for DH11 multiplexer lines only; or for both single line interfaces and DH11 multiplexer lines.

2.7.8 2741 Code and Keyboard Arrangements

RSTS/E supports four code and keyboard arrangements as shown in Section 5.8 of the RSTS-11 System User's Guide. The system manager must define the code or codes he wants his system to support so that they are assembled into the terminal service module. He can include any combination of one to four of the four supported codes.

If more than one code is configured, a small routine is included in the terminal service to enable a user at a 2741 terminal to change codes. The first code named in response to the configuration question is assigned as the system default. For example, if both Correspondence and EBCD 2741 terminals are to be supported but EBCD is the preferred default, type EBCD, CORR in response to the 2741 CODE question.

2.7.9 Multiple Terminal Service

The multiple terminal service option allows one BASIC-PLUS program to simultaneously interact with several terminals on one I/O channel.

With the multiple terminal service on the system, a BASIC-PLUS program can control several keyboards by establishing a master keyboard on a single channel and by reserving various other keyboards as slave terminals. To perform input or output, the program executes standard Record I/O statements on that single channel and utilizes special software options to determine the unique keyboard being serviced. Additionally, in absence of keyboard input, the system can automatically stall program execution and make the program eligible to run when keyboard input is pending from a particular keyboard or any keyboard controlled by the program. Thus, a single program can interactively service low volume keyboard input and output associated with several stations. This procedure eliminates the need to run separate copies of the same program at each terminal when several terminals must perform a similar function. The option is explained in Section 11.11.1 of the BASIC-PLUS Language Manual.

2.7.10 Disk Devices

Disks in the RSTS/E system operate in either the public or private structure. The disk which contains the system accounts and executable code of RSTS/E is called the system disk and is the first disk of the public structure. All other disks in the system are referred to collectively as non-system disks.

The most practical use of the system disk on the RSTS/E system is as a removable disk as opposed to a fixed head disk. If the system disk is either an RKØ5 or RKØ3 disk cartridge or an RPØ3 or RPØ4 disk pack, it can be removed from the computer area when the system is not operating and kept in a safe place, thereby reducing the chances of inadvertent or malicious destruction. To preserve the contents of a fixed head disk, a copy must be transferred to a secondary medium each time the system is shut down.

Optimum performance is obtained if the system is configured with a moving head disk (removable) and an auxiliary swapping disk (fixed head). With such a configuration, the swapping of user jobs into and out of memory is faster and more efficient. Disk accessing operations on the moving head system device can then be confined to manipulating user files and directories while the faster fixed head device takes on the burden of moving user jobs into and out of memory. In such a case, the auxiliary swapping disk acts as a logical extension of the system disk while the system is operating but contains no valuable system data

when the system is not operating. At the start of time sharing operations, the initialization code creates the necessary files on the auxiliary swapping device.

If a large amount of public storage space is required on the system, a single RPØ3 or RPØ4 disk pack drive is preferable to multiple RKØ5 or RKØ3 DECpack disk drives. The use of multiple devices in the public structure increases the overhead required in file operations, since, each time a file is created within the public structure, the directories of each public device must be searched to ensure that a file of the same name does not already exist. An alternative method of structuring the disks to reduce system overhead in configurations using multiple drives is to use private disks rather than public disks. In systems having several types of disk devices, the decision is left to the system manager as to which device is to be used as the system disk and which devices are used in the public and private disk structure.

The RK11, RP11-C, and RH11 controllers allow several drives to perform seek operations simultaneously. Since the controller is not busy during seek operations, data transfers can be overlapped with seeks in progress on other drives. On systems with two or more drives on the same controller, SYSGEN normally configures an overlapped seek driver to accelerate processing on the related disk subsystem.

Since the overlapped seek drivers require more memory than the non-overlapped drivers, SYSGEN allows the system manager to include the non-overlapped version in place of the standard overlapped driver. The advantages in throughput afforded by the overlapped driver normally outweigh the disadvantages of the additional memory required. Therefore, the overlapped drivers are recommended for all systems with multiple drives on the same controller. To have SYSGEN substitute a non-overlapped seek driver, append /NO to the number specified in response to the questions concerning the number of drives.

2.7.11 Peripheral Devices

The use of peripheral devices in the RSTS/E system reduces the burden of storage requirements on the disk devices and provides a convenient medium for file archives. Program and data files that are not frequently used can be stored on magnetic tape (DECtape or magtape), paper tape (high speed, fan-folded or Teletype), and cards (marked or

punched) and accessed readily when required. DECtape and magtape provide a large capacity storage medium for critical file information.

Production of hard copy output on the RSTS/E system can be improved by the use of multiple line printers, each having distinct operating characteristics which must be specified during system generation. A hardware modification can be made to a line printer to inhibit the form advance (automatic page ejection) every 66 lines and thus allow the user flexible control over vertical formatting. This feature is made available to users by the MODE option in the OPEN statement as described in Section 12.5 of the BASIC-PLUS Language Manual.

2.7.12 LPØ: for SYSGEN

During system generation, four files (load maps) containing memory allocation information are created. For magtape and DECtape distribution media, the files are written to the resultant LICIL tape which contains the RSTS/E system. For disk cartridge distribution media, the files are retained on the system generation disk. The load maps are standard ASCII text files with filenames RSTS.MAP (monitor), OVR.MAP (overlay code), INIT.MAP (initialization code), and BASIC.MAP (Run Time System).

Since the load maps contain information essential to problem diagnosis, they should be printed at some time. If line printer unit Ø is available for use, SYSGEN can print the load maps during system generation whether or not a line printer is configured into the RSTS/E system. If line printer unit Ø is not available or if the user prefers to perform the printing at a later time using the PIP program, he can omit the system generation printing. The load maps are retained on the LICIL tape and on the system generation disk.

To print the load maps during system generation when the line printer is available, type YES in response to the LPØ: FOR SYSGEN question. As a result, the four load maps are automatically printed later in the system generation process. Printing takes about 1Ø minutes on a 3ØØ line per minute printer. For magtape and DECtape distribution media, a directory of the LICIL tape is also printed at unit Ø rather than at the console terminal. When the line printer is available, SYSGEN prints the question concerning listings described in Section 2.7.13.

To omit printing the load maps during system generation, type NO in response to the LPØ: FOR SYSGEN question. As a result, the load maps are not printed but the files are available for later printing as described above. For magtape and DECtape distribution media, the directory of the LICIL tape is printed at the console terminal rather than at the line printer. In addition, SYSGEN does not print the question concerning listings described in Section 2.7.13.

2.7.13 Listings

During the system generation process, the system tables and terminal service modules are assembled. An additional assembly is performed if concise command language entries are created as described in Section 2.7.24.

Since each of the assemblies is unique to the installation, the assembly listings provide information valuable for documentation and maintenance purposes. If the LPØ: FOR SYSGEN question is answered with YES, SYSGEN prints the LISTINGS question to allow the listings to be printed during the system generation process. If the answer to the LPØ: FOR SYSGEN question is NO, SYSGEN does not print the LISTINGS query. In certain circumstances, the listings can be printed at a later time.

The ability to print the listings at a later time depends on whether or not the listing files TBL.LST (system tables) and TTY.LST (terminal service modules) are retained. If the system generation disk is an RF type, the files are not retained because they can not physically fit on the disk. Therefore, the listings must be printed during system generation or they can never be printed. If the system generation disk is either an RK cartridge or RP pack, the files are retained on that disk. If magtape media is used, the files are retained on the LICIL tape created during system generation. The files can be printed from the LICIL magtape by using the PIP program under either DOS or RSTS/E. The files do not fit on one DECtape and are therefore not retained on the LICIL DECtape.

To print the listings at system generation time, simply type YES in response to the LISTING and CCL LISTING questions. As a result, the listings are automatically printed later in the system generation process. The listings are quite lengthy and take approximately 3Ø minutes to print on a 3ØØ line per minute printer. To omit printing the listings, type NO to the LISTING question.

2.7.14 Card Codes

If the RSTS/E system has a card reader, the user must configure one of three card codes. These card codes are presented for reference in Appendix D.3 of the BASIC-PLUS Language Manual. The standard card code is DECØ29 code.

2.7.15 278Ø Package

The 278Ø capability on RSTS/E allows a user to communicate with IBM 36Ø and 37Ø and other DIGITAL PDP-11 computers having the 278Ø Data Terminal capability described in the RSTS/E 278Ø User's Guide. The RSTS/278Ø software is not included in the standard RSTS/E distribution media but is sold and supported as a separate product. If the 278Ø capability is to be included in the system, the 278Ø software must be available on the same medium as the RSTS/E distribution media. Messages printed during the standard system generation tell when to mount the tape or disk containing the RSTS/278Ø software. The RSTS/E computer must have the KG11A communications arithmetic unit and either a DP11 or a DU11 synchronous line interface unit. If answers to the SYSGEN questions indicate that neither a DP11 nor DU11 interface is present, the question concerning the 278Ø package is omitted.

2.7.16 Non-Supported Devices

Devices not supported by RSTS/E may be attached to the computer on which time sharing operations are conducted. Although the system cannot address these devices, their existence must be known so that proper floating address and floating vector assignments can be made. If a positive answer is given to the non-supported device question, SYSGEN prints each device name and requires the user to type the number present on the system.

2.7.17 Maximum Number of Jobs

With sufficient hardware, RSTS/E can handle up to 63 simultaneous jobs. The maximum number of jobs must be specified at system generation time since this parameter determines the size of several monitor tables. The number can be adjusted downward when the JOB OR SWAP MAX CHANGES query is answered as described in Chapter 3. The maximum number of jobs can not be increased above the configured maximum unless the system code is regenerated.

The maximum number of jobs which can be run efficiently depends on the memory space available and the number and types of disks on the system. Memory space requirements are discussed in Chapter 1. To calculate the effect of disk devices, consult the material in Chapter 3 concerning swapping.

Jobs on the system are numbered sequentially from one to the maximum number the system can handle. Jobs include both those attached and detached.

2.7.18 Small Buffers

The RSTS/E system handles transfer requests and file processing requests by means of intermediate memory storage, called small buffers. These buffers are considered a system resource, and a sufficient number must be configured at system generation time. If an insufficient number is configured, jobs running on the system can become stalled waiting until enough buffers are freed by jobs currently claiming their use.

Small buffers are 16-word blocks residing in the Monitor part of memory. The number needed by a system at any one time depends upon the dynamic requirements of the jobs on the system. For efficient system operation, it is recommended that at least 1Ø small buffers be allocated for each possible job. Thus, on a 32-user system, 32Ø small buffers should be available. (An indicator of good system performance is that the number of free small buffers, as reported by the SYSTAT system program, never drops below ten. Refer to the description of SYSTAT in Section 6.6 of this guide.)

On systems configured for 16 or more jobs, 1Ø small buffers for each job is usually an adequate number. However, on systems configured for fewer than 16 jobs, it may be advantageous to include more than 1Ø small buffers per job. For example, each active terminal requires 4 or 5 small buffers for performing input and output operations. A system having 8 terminals therefore needs between 32 and 4Ø small buffers if all terminals are to be simultaneously active. Each active job requires three small buffers. Thus, if the same system required that ten jobs be able to run simultaneously, 3Ø more small buffers would be needed. For each job that will run detached from all terminals, subtract the four small buffers required for terminal I/O. A running total on the 8-terminal, 1Ø-job system is 62 small buffers for these two simple processing requirements.

Next, in the sample system, consider what kind of processing is necessary. One small buffer must be added for each open file on the system. If each program running on the system opens two disk files, 2Ø more small buffers must be added. If all the active programs open the maximum number of files simultaneously, 12Ø small buffers must be available. (BASIC-PLUS allows 12 open channels per user program.) In an average system, the two file situation is much more likely, so the sample system requires 2Ø more small buffers for a total of 82.

The system requires small buffers for certain transient operations. One small buffer is used for each disk transfer queued by the monitor. Programs such as QUEMAN, QUE, ERRCPY, RJ278Ø, BATCH, and SPOOL communicate through the SEND/RECEIVE system function call which claims one small buffer for each message queued for a receiving job. Normally the buffers used for these operations are employed for very short time periods (fractions of a second) but a reasonable number of buffers should be available. On the 1Ø job, 8 terminal system, a reasonable number is approximately 2Ø more small buffers.

Line printers on the system exhaust as many as one third of the available small buffers. A lower number of available small buffers places a larger burden on the system. For example, if 3Ø small buffers are available for use by the printer driver, the system can have 9ØØ characters buffered for output to the line printer. Assume that a line printer is running at 3ØØ lines per minute (5 lines per second) and that an average line is 9Ø characters. Such a line printer empties the buffers in two seconds. A spooling program for that line printer would have to be swapped into memory every 2 seconds to keep the line printer running at full speed. (For a line printer running at 12ØØ lines per minute, a swap operation would be necessary every one half second.)

The total requirement on the 1Ø job, 8 terminal system is nearly 14Ø small buffers for an average system load if the transient requirement for 2Ø is added and one third of the subtotal is added to handle the line printer running full speed (3ØØ lines per minute). Thus, the guideline of 1Ø small buffers per job is too low on such a small system. Moreover, if small buffers were subtracted from 14Ø to account for idle terminals and detached jobs and to allow for some slow down in line printer operations, the guideline is still inadequate. On such a system, between 12 and 14 small buffers per job is a better approximation. For larger systems having 16 or more jobs, ten small buffers per job is

usually a good approximation. Except for occasions of heavy keyboard and line printer activity, enough free small buffers will be available to maintain good system throughput.

2.7.19 Big Buffers

Big buffers are 256-word blocks of monitor memory used for DECtape and 278Ø operations. Since the SYSGEN program automatically configures four big buffers for 278Ø software, big buffers need be configured only if DECtape is on the system. Consequently, SYSGEN prints the BIG BUFFER question only if DECtape is on the system.

On systems without 278Ø software but with DECtape devices, one big buffer is normally provided for each DECtape drive. However, one big buffer per drive is not a definite requirement since one big buffer can accommodate any number of DECtape drives for non-simultaneous operations. Experience indicates that, unless DECtape usage is heavy, two big buffers are sufficient even for four drives. Three big buffers are recommended for six drives and four big buffers are recommended for eight DECtape drives.

On systems with both RSTS/278Ø software and DECtape drives, one big buffer is sufficient for eight DECtape units. The four big buffers automatically included for RSTS/278Ø software are available for DECtape operations when the RSTS/278Ø software is not operating. In any case, the SYSGEN program does not allow less than one big buffer when DECtape is present on the system. This fact applies whether or not the 278Ø software is present.

2.7.20 Receivers

A receiver is a job capable of performing interjob communication using the SEND/RECEIVE system function call described in Section 7.2.12. The QUEMAN job and each SPOOL and BATCH job running on the system performs as a receiver. Enough receivers must be configured for such jobs and for additional programs which declare themselves receivers. For example, on a system running one SPOOL program and one BATCH program, configure at least three receivers (QUEMAN, SPOOL, and BATCH).

2.7.21 Power Fail Recovery Code

RSTS/E systems can attempt to recover from a momentary power failure by performing an automatic restart procedure. A momentary power failure is defined in Section 2.7.1 of both the PDP-11/40 Processor Handbook and PDP-11/45 Processor Handbook.

2.7.22 File Processor (FIP) Buffering

The FIP buffering module accelerates file processing on the RSTS/E system. The module reduces the number of accesses to the disk by maintaining more than one disk directory block in memory.

Normally, RSTS/E employs only one permanently allocated 256-word buffer to store disk directory information. File processing is often delayed because the system must write the buffer to the disk to update the directory before it can read new directory information into the buffer.

The FIP buffering module claims, from the normal buffer pool, free small or big buffers and stores additional directory blocks in the buffers. Frequently used blocks are thus kept in memory to avoid accesses to the disk. If any of the buffers are required for their standard uses, the system releases them from the FIP buffering module. The number of buffers configured for other system operations need not be increased when FIP buffering is included on the system.

The FIP buffering module increases the size of the monitor and is therefore an option on all RSTS/E systems.

2.7.23 Resident Code

Certain portions of the overlay code can be configured resident in memory during time sharing rather than stored on the disk. Performance is thereby improved on systems where that code has heavy usage.

The most important code in terms of system performance is the disk handling code. This block of code includes: the system routines for disk file creation; OPEN, CLOSE, and RENAME routines; some routines needed to process the RUN command; general routines for opening other devices, deassigning devices, fetching error messages, looking up files

by name; and for logout. It is recommended that systems with more than 56K words of memory have disk handling code resident.

The SYS call dispatch code is used whenever a program executes a SYS call to the file processor (FIP). The code is called to dispatch to other non-resident or resident portions of monitor code. It is recommended that the SYS call dispatch code be made resident on any system having more than 64K words of memory and using FIP calls for many programs.

The SEND/RECEIVE code provides inter-job communications for such programs as QUEMAN, BATCH, SPOOL, and RJ278Ø. SEND/RECEIVE code is entered through the SYS call dispatch code. It is not recommended that this code be configured resident unless system operations include frequent usage of programs executing the SEND/RECEIVE SYS call and the related SYS call dispatch code is also made resident.

The directory lister code is executed for CATALOG command operations and gathers information concerning disk directories. It is small but is less important than the disk handling, SYS call dispatch, and SEND/RECEIVE code. Since the system program DIRECT is more efficient than the CATALOG command, it is recommended that DIRECT be used wherever possible. Hence, little reason exists to make this code resident.

2.7.24 Concise Command Language (CCL)

The SYSGEN program allows the user to include or omit the CCL capability described in the introductory material of Chapter 4 in the RSTS-11 System User's Guide. This capability provides a set of keyboard commands which run programs stored in the system library. If the CCL capability is included, one of three options is possible: accept a standard set of commands, define a new set of commands, or add new commands to the standard set.

The standard set of CCL commands involves the following programs and consists of the related commands.

DIRECT	DIR
EDIT	EDIT
EDIT	CREATE
PIP	HELP

PIP	PIP
QUE	QUE
SYSTAT	SYS
TTYSET	SET
UMOUNT	MOUNT
UMOUNT	DISMOUNT

The programs listed are automatically included in the system library and are coded so that the standard CCL commands function properly.

SYSGEN allows a unique set of commands to be defined by either adding to the standard set or creating a completely new set. The programs defined to run by CCL commands must conform to standards described in Section 6.18 of this guide. The system manager can add 1Ø new commands or can create a new set containing as many as 2Ø commands. To define the commands, simply type the program name and the command separated by a comma. To change the standard set of CCL commands, a new set must be entered which includes any desired commands from the standard set and any additional commands required. To restrict usage of CCL commands on the system, adjust the protection code of the related program.

The precedence of CCL commands is above that of RSTS/E commands and BASIC-PLUS immediate mode statements. As a result, the system manager can control the use of a command or an immediate mode statement on his system. For example, he could specify a command BYE which performs certain operations before allowing a user to log off the system. As another example, he could specify a command PRINT which performs operations different from those of the BASIC-PLUS immediate mode PRINT statement. Such a CCL command has no effect on a BASIC-PLUS statement preceded by a line number since a line numbered statement has a higher precedence than the CCL command.

The same program can be run by more than one CCL command. A single inventory control program might be designed to handle single line requests concerning items on hand, items on order, and orders backlogged. The sample printout of Section 2.6.1 shows this type of facility.

If new commands are added to the standard set or a new set is created, the resulting CCL table must be assembled later in the system generation process. As a result, SYSGEN prints the CCL LISTING

ON LPØ: question if a line printer is available for use during SYSGEN as discussed in Section 2.7.10. If the standard set of CCL commands is accepted, no assembly is necessary, no listing is produced, and SYSGEN does not print the question.

Since the CCL listing provides useful system documentation, it is important that the listing file be printed. The CCL listing file USRCCL.LST is preserved on disk and tape in the same manner as the system tables and terminal service listing files described in Section 2.7.13. Similarly, the file can be printed at a later time unless the system generation disk is an RF11 device. In the case of an RF11, the CCL listing must be printed during the system generation process or it cannot be printed at all.

If YES is the response to the CCL LISTING ON LPØ: query, the listing file is printed during the system generation process. This listing is fairly short and takes about two minutes to print on a 3ØØ line per minute printer.

2.7.25 Floating Point Precision and Scaled Arithmetic

The system manager can select either single precision (2-word) or double precision (4-word) floating point format for the type of numeric format to be used on his system. These floating point formats are described in Appendix F.1 of the BASIC-PLUS Language Manual.

Answers supplied to the FIS(11/4Ø), FPP(11/45), MATH PRECISION, and FUNCTIONS questions determine which mathematical software package is selected from the ten standard packages included in the RSTS/E software. The following list describes the 2-word math packages.

MA2	Without FIS or FPP; with extended functions
MA2X	Without FIS or FPP; without extended functions
MA2I	11/4Ø FIS; with extended functions
MA2IX	11/4Ø FIS: without extended functions
MA2F	11/45 FPP; with extended functions
MA2FX	11/45 FPP; without extended functions

The following list describes the 4-word math packages.

MA4	Without FIS or FPP; with extended functions
MA4X	Without FIS or FPP; without extended functions
MA4F	11/45 FPP; with extended functions
MA4FX	11/45 FPP; without extended functions

The most critical difference among packages is the implementation of floating point operations. The PDP-11/45 Floating Point Processor (FPP) provides both 2-word and 4-word floating point instructions in hardware. The PDP-11/40 Floating Instruction Set (FIS) does not provide 4-word floating point instructions in hardware. Therefore, on PDP-11/40 computers with or without FIS, RSTS/E must use the slower software packages (MA4 or MA4X) to perform 4-word floating point operations.

The scaled arithmetic feature is standard only on systems with 4-word floating point format. The feature is described in Section 6.8 of the BASIC-PLUS Language Manual and the SCALE command is described in Section 2.8 of the RSTS-11 System User's Guide. Scaled arithmetic avoids problems such as loss of precision normally associated with floating point calculations. The feature is very useful for calculating sums (such as money) which cannot be manipulated easily as integer quantities.

2.7.26 Mathematical Functions

The size of the BASIC-PLUS Run Time System (RTS) can be reduced by omitting certain mathematical functions including SIN, COS, TAN, ATN, SQR, EXP, LOG, and LOG10. These functions are described in Section 3.7 of the BASIC-PLUS Language Manual and summarized in Table 3-1 of that section.

2.7.27 PRINT USING Option

The PRINT USING optional feature allows BASIC-PLUS programs to perform special formatting of output as described in Section 10.4.1 of the BASIC-PLUS Language Manual. The size of the BASIC-PLUS Run Time System can be reduced by omitting this option.

2.7.28 Matrix Manipulation

BASIC-PLUS can operate on an entire matrix using single statements called MAT statements as described in Chapter 7 of the BASIC-PLUS Language Manual. The system manager must configure this optional feature if he wants to include the matrix manipulation capability.

2.7.29 Stand Alone Programs

The stand alone program ROLLIN can be included in the RSTS/E CIL and can be loaded into memory using the LOAD option as described in Chapter 3. ROLLIN is described in the library document entitled PDP-11 ROLLIN Utility Program which is included in the RSTS/E distribution software.

2.8 LOADING THE CIL ONTO THE RSTS/E SYSTEM DISK

After the configuration question concerning ROLLIN is answered, the SYSGEN program completes building the configuration file and the second batch command file. Subsequently, SYSGEN prints the second checkpoint message. Next, the system generation monitor executes commands in the second batch command file and generates the RSTS/E linked core image library (LICIL) and any listings necessary.

During the generation, messages are printed indicating which devices to mount and how to proceed. The entire process takes between one and three hours depending upon the devices used and the types of listings requested. If either magtape or DECtape distribution media is employed, the user must mount a new, formatted tape to which the RSTS/E LICIL is written. Instructions are printed to mount the RSTS/E system disk on unit Ø and to type the exact command to transfer the LICIL to the disk. If disk cartridge distribution media is used, instructions are printed to mount a new cartridge or pack to be used as the RSTS/E system disk.

After the CIL is written on the RSTS/E system disk and the RSTS/E code is loaded into memory for the first time, the PATCH, DSKINT, REFRESH, SETKEY, DEFAULT, and START options must be run as described in Chapter 3 and the system library must be built according to procedures described in Chapter 4.

2.9 SYSLOD AND CILUS COMMAND STRINGS

This section describes the procedures and command strings employed in loading the RSTS/E Linked Core Image Library (LICIL) onto the RSTS/E system disk. All necessary instructions to perform the load operation are printed during the system generation procedure. If the standard procedure is followed, this section need not be referenced. However, the command strings documented here are useful if the load operation is not performed as part of the system generation procedure or if the newly created LICIL must replace an old RSTS/E Core Image Library (CIL) on an existing system disk.

Loading the CIL is the final step of the system generation process. For magtape and DECtape distribution media, the newly created Linked Core Image Library (LICIL) is written to tape along with a copy of the SYSLOD program, the batch and configuration files, system load maps, and assembly listing files (magtape only). The SYSGEN program prints the commands necessary to load the CIL onto the RSTS/E system disk with SYSLOD. For disk cartridge distribution media, the LICIL resides on the system generation disk along with the batch and configuration files, system load maps, and the listing files. The command to load the CIL is never printed because the batch stream performs the load operation with the CILUS program. However, for all distribution media, the CIL need not be loaded immediately since the standard procedure can be halted just before the load operation. If the LICIL tape or system generation disk is preserved, the device can be bootstrapped and either SYSLOD or CILUS can be used to load the CIL onto the RSTS/E system disk.

Because the load operation can be performed using either a blank or an existing system disk, two general operations are described. Section 2.9.1 presents the procedures to load a new RSTS/E system onto a disk which contains no user files to be preserved. Section 2.9.2 provides guidelines for replacing an existing RSTS/E system on a system disk containing system and user files which must be preserved.

2.9.1 Loading the RSTS/E CIL onto a Blank System Disk

The procedure and command strings detailed in this section apply only to loading a CIL onto a blank system disk. It is possible to overwrite an existing CIL on a system disk which contains system and user files without destroying the file structure. This latter procedure is described in Section 2.9.2. The procedures below destroy any existing file structure on the disk being initialized as the RSTS/E system disk.

2.9.1.1 DECtape and Magtape Procedures Using SYSLOD - The tape created during system generation contains a copy of the stand-alone program SYSLOD. SYSLOD is loaded from DECtape using one of the hardware bootstrap loaders described in Appendix F. When SYSLOD is bootstrapped into memory, it will identify itself by printing the following lines.

```
SYSLOD VØ8-Ø1A
CONSOLE FILL COUNT=                 Type the RETURN key
DATE:  11-JAN-75                    Type the date in format shown
DIALOGUE?                           Type the RETURN key

#
```

Before the DIALOGUE query is answered, the disk to be used as the RSTS/E system disk should be mounted and write-enabled. In the case of an RK, RPØ3, or RPØ4 system disk, an RK cartridge or an RP pack must be mounted on drive unit Ø. No special action is required for an RF system disk. SYSLOD does not recognize any device which is not mounted and ready when the DIALOGUE query is answered. The RETURN key is sufficient response to the DIALOGUE query. SYSLOD responds by printing the pound sign (#) when it is ready to accept a command. A single command string is sufficient to create and load the CIL onto the system disk and bootstrap the RSTS/E Initialization code into memory. The exact command which must be entered depends on the type of system disk. If any error messages are printed by SYSLOD, consult Appendix B for the proper procedure to follow. The following command strings are used for the several types of system disks.

```
#DF:/NS:256:49/TO:4/HO/BO<DTØ:RSTS.LCL   (RF11 System Disk < DECtape)
#DF:/NS:256:49/TO:4/HO/BO<MTØ:RSTS.LCL   (RF11 System Disk < TU1Ø Magtape)
#DF:/NS:256:49/TO:4/HO/BO<MMØ:RSTS.LCL   (RF System Disk   < TU16 Magtape)
```

```
#DKØ:/NS:256:49/FO/TO:4/HO/BO<DTØ:RSTS.LCL  (RK11 System Disk < DECtape)
#DKØ:/NS:256:49/FO/TO:4/HO/BO<MTØ:RSTS.LCL  (RK11 System Disk < TU1Ø Magtape)
#DKØ:/NS:256:49/FO/TO:4/HO/BO<MMØ:RSTS.LCL  (RK11 System Disk < TU16 Magtape)

#DPØ:/NS:512:25/FO/TO:4/HO/BO<DTØ:RSTS.LCL  (RPØ3 System Disk < DECtape)
#DPØ:/NS:512:25/FO/TO:4/HO/BO<MTØ:RSTS.LCL  (RPØ3 System Disk < TU1Ø Magtape)
#DPØ:/NS:512:25/FO/TO:4/HO/BO<MMØ:RSTS.LCL  (RPØ3 System Disk < TU16 Magtape)

#DBØ:/NS:1Ø24:13/FO/TO:4/HO/BO<DTØ:RSTS.LCL (RPØ4 System Disk < DECtape)
#DBØ:/NS:1Ø24:13/FO/TO:4/HO/BO<MTØ:RSTS.LCL (RPØ4 System Disk < TU1Ø Magtape)
#DBØ:/NS:1Ø24:13/FO/TO:4/HO/BO<MMØ:RSTS.LCL (RPØ4 System Disk < TU16 Magtape)
```

For an RPØ3 system disk, messages are printed at the console terminal which tell the system manager to set and reset the FORMAT ENABLE/NORMAL switch which is located behind the third panel below the indicator lights on the RP11-C controller.

Upon completion of the load operation, the RSTS/E Initialization code is bootstrapped into memory signalled by the printing of the OPTION query. The system manager should proceed to initialize the system disk according to the procedures described in Chapter 3.

2.9.1.2 Disk Cartridge Procedures Using CILUS - When the disk cartridge distribution medium is used, the DOS program CILUS is used to load the RSTS/E CIL onto the system disk. CILUS does not format an RK cartridge, RPØ3, or RPØ4 pack. Refer to Section 2.3.3 for procedures to load SYSLOD and format disks.

Mount the copy of the System Generation DECpack used for the system generation on RK unit Ø. Write enable the drive and bootstrap the cartridge to load the DOS/BATCH monitor. Refer to Appendix F for DECpack bootstrap procedures. When the DOS monitor identifies itself, proceed as shown below to run CILUS.

```
DOS/BATCH V9-2ØC
DATE:  11-JAN-75                Type the date in format shown
TIME:  16:15                    Type the time in format shown
DIALOGUE?                       Type the RETURN key
$LO 1,1
DATE:  11-JAN-75
TIME:  16:15
$RUN CILUS
CILUS VØ8-Ø6A
#
```

CILUS prints the pound sign (#) when it is ready to accept a command. The CILUS command string used to create and load the RSTS/E CIL depends on the type of system disk. If the RSTS/E system is configured for an RPØ3 or RPØ4 system disk, mount a newly formatted pack on unit Ø and write-enable the drive. If the system is configured for an RK system disk, mount a newly formatted RK cartridge on RK unit 1. No special action is required for an RF system disk. One of the CILUS commands shown below is then used to load the CIL.

```
#DF:/NS:256:49/TO:4/HO/BO<DKØ:RSTS.LCL/LO  (RF11 System Disk < RK cartridge)
#DK1:/NS:256:49/TO:4/HO<DKØ:RSTS.LCL/LO    (RK11 System Disk < RK cartridge)
#DPØ:/NS:512:25/TO:4/HO/BO<DKØ:RSTS.LCL/LO (RPØ3 System Disk < RK cartridge)
#DBØ:/NS:1Ø24:13/TO:4/HO/BO<DKØ:RSTS.LCL/LO (RPØ4 System Disk < RK cartridge)
```

If the system disk is an RF11, RPØ3, or RPØ4, the CILUS command loads the CIL and then bootstraps the RSTS/E initialization code into memory. The INIT code prints the system name followed by the OPTION query. The system manager should proceed to initialize the system disk as described in Chapter 3.

The command to load the CIL onto the RK cartridge mounted on unit 1 does not bootstrap the RSTS/E Initialization code. When the load operation is complete, CILUS reprints the pound sign (#) and waits for another command. The system manager exits from CILUS and terminates the DOS monitor as shown below.

```
#^C                          CONTROL/C exit from CILUS
.KI                          "KILL" required by DOS
$FI                          FInish for an orderly exit
TIME:  16:16:16
DOS/BATCH V9-2ØC

$
```

The processor is halted by moving the HALT/ENABLE switch to the HALT position. Dismount both cartridges and move the RSTS/E system disk to RK unit Ø. When the disk is ready, write-enable the drive and bootstrap the disk cartridge (see Appendix F) to load the RSTS/E initialization code into memory. Proceed to Chapter 3 to initialize the new system disk.

2.9.2 Replacing the RSTS/E System Code

It is possible to replace the RSTS/E CIL on the system disk without destroying the file structure. This capability is important when a new system is generated to add or change hardware support or software features. The SYSLOD or CILUS command strings used for this purpose are similar to those in Section 2.9.1 but several precautions should be taken to ensure a successful replacement. Careful adherence to these procedures is critical to avoid destroying the existing file structures (system and user files) on the system disk.

It is impossible to determine the exact size of the new CIL until it is loaded onto a disk. The first step in replacing the CIL, therefore, is to load the new monitor onto a scratch disk using the standard system generation procedures or the SYSLOD or CILUS commands described in Sections 2.9.1.1 and 2.9.1.2. The scratch disk must be initialized using the DSKINT initialization option (see Section 3.3) to create the RSTS.CIL file. When a disk is initialized, pattern checking for bad blocks is normally done. However, requesting zero patterns causes DSKINT to bypass the pattern checks, saves time, and is a reasonable shortcut for the scratch disk.

The REFRESH initialization option (Section 3.4) is used after DSKINT to check the required size of the new RSTS.CIL file. REFRESH can be stopped by typing CONTROL/C after the file status table is printed. Since the scratch disk is used only to determine the size of the new RSTS.CIL file, it is of no further use after these operations are performed. The loading of the new system onto a scratch disk and the DSKINT and REFRESH operations take a total of about 1Ø minutes.

The next step is to determine the size of the CIL to be replaced. Simply obtain a directory of the system files account [Ø,1] under time sharing or print the file status table using REFRESH (see Section 3.4.3.2). The important item is the current size of the RSTS.CIL file on the system disk. The REFRESH procedures of Section 3.4 recommend that this file be made larger than the required size when the system disk is initially built. If the system manager planned for a future replacement of the system code, the current size of the old RSTS.CIL file will probably be larger than the required size of the new RSTS.CIL file.

If the old CIL is larger, the replacement can proceed as described in subsequent paragraphs. Otherwise, the system manager can attempt to increase the size of the old RSTS.CIL file using REFRESH before attempting the replacement. If the REFRESH is successful, he can proceed. If REFRESH cannot increase the size of the old RSTS.CIL file to at least the required size of the new RSTS.CIL file, the replacement cannot be performed. All library and user files must be transferred to another disk or external medium and the system disk must be initialized (destroying the existing file structure).

Assuming the RSTS.CIL file on the system disk is large enough to accommodate the new CIL, the next step is to transfer all library and user files from the system disk to a secondary storage medium. This is a time consuming but important precaution since a typographical error or a hardware malfunction while replacing the old CIL could be disastrous. The transfer is performed under time sharing using the old system. The next two sections present the SYSLOD and CILUS command strings which replace the old CIL.

2.9.2.1 DECtape and Magtape Procedures Using SYSLOD - The tape created during system generation contains a copy of the stand-alone program SYSLOD. SYSLOD is loaded from DECtape or magtape using one of the hardware bootstrap loaders described in Appendix F. When SYSLOD is bootstrapped into memory, it identifies itself by printing the following lines.

```
SYSLOD VØ8-Ø1A
CONSOLE FILL COUNT=                Type the RETURN key
DATE:  11-JAN-75                   Type the date in format shown
DIALOGUE?
#
```

Before the DIALOGUE query is answered, the old RSTS/E system disk should be mounted and write enabled. In the case of an RK, RPØ3, or RPØ4 system disk, the cartridge or pack must be mounted on unit Ø. SYSLOD does not recognize any device which is not mounted and ready when the DIALOGUE query is answered. The RETURN key is sufficient response to the DIALOGUE query. SYSLOD responds by printing the pound sign (#) when it is ready to accept a command. The following SYSLOD commands replace an old CIL on the various types of system disks.

```
#DF:/NS:256:49/TO:4/HO/BO<DTØ:RSTS.LCL    (RF11 System Disk < DECtape)
#DF:/NS:256:49/TO:4/HO/BO<MTØ:RSTS.LCL    (RF11 System Disk < TU1Ø Magtape)
#DF:/NS:256:49/TO:4/HO/BO<MMØ:RSTS.LCL    (RF11 System Disk < TU16 Magtape)

#DKØ:/NS:256:49/TO:4/HO/BO<DTØ:RSTS.LCL   (RK11 System Disk < DECtape)
#DKØ:/NS:256:49/TO:4/HO/BO<MTØ:RSTS.LCL   (RK11 System Disk < TU1Ø Magtape)
#DKØ:/NS:256:49/TO:4/HO/BO<MMØ:RSTS.LCL   (RK11 System Disk < TU16 Magtape)

#DPØ:/NS:512:25/TO:4/HO/BO<DTØ:RSTS.LCL   (RPØ3 System Disk < DECtape)
#DPØ:/NS:512:25/TO:4/HO/BO<MTØ:RSTS.LCL   (RPØ3 System Disk < TU1Ø Magtape)
#DPØ:/NS:512:25/TO:4/HO/BO<MMØ:RSTS.LCL   (RPØ3 System Disk < TU16 Magtape)

#DBØ:/NS:1Ø24:13/TO:4/HO/BO<DTØ:RSTS.LCL (RPØ4 System Disk < DECtape)
#DBØ:/NS:1Ø24:13/TO:4/HO/BO<MTØ:RSTS.LCL (RPØ4 System Disk < TU1Ø Magtape)
#DBØ:/NS:1Ø24:13/TO:4/HO/BO<MMØ:RSTS.LCL (RPØ4 System Disk < TU16 Magtape)
```

The only difference between these SYSLOD commands for replacing a CIL and those used for a blank system disk is the absence of the format switch (/FO) for RK and RPØ3 or RPØ4 system disks. Formatting destroys everything on a disk and cannot be done if the file structure is to be preserved.

Upon completion of the load operation, the RSTS/E Initialization code is bootstrapped into memory signalled by the printing of the OPTION query. The system manager must reinstall all published patches using the PATCH option (see Section 3.2), set keyboard defaults with the SETKEY option (Section 3.4A) and establish default start up conditions with the DEFAULT option (Section 3.5) before time sharing operations can resume with the new system. The DSKINT initialization option is not used in this case since initializing a disk destroys any existing file structures. The REFRESH option can be used to verify or change any of the system files but is not specifically required after replacing the CIL.

2.9.2.2 DECpack Procedures Using CILUS - When disk cartridge distribution media is used, the DOS program CILUS loads the new CIL onto the old system disk. Since CILUS does not format disks, the CILUS commands are the same as for loading a CIL onto a blank system disk. The CILUS procedures are repeated below for continuity.

Mount the copy of the system generation disk cartridge used for the system generation on RK unit Ø. Write-enable the drive and bootstrap the cartridge to load the DOS/BATCH Monitor. Refer to Appendix F

for bootstrap procedures. When the DOS Monitor identifies itself, proceed as shown below to run CILUS.

```
DOS/BATCH V9-2ØC
DATE:  11-JAN-75                Type the date in format shown
TIME:  12:3Ø                    Type the time in format shown
DIALOGUE?                       Type the RETURN key
$LO 1,1
DATE:  11-JAN-75
TIME:  12:3Ø
$RUN CILUS
CILUS VØ8-Ø6A
#
```

CILUS prints the pound sign (#) when it is ready to accept a command. If the RSTS/E system is configured for an RPØ3 or RPØ4 system disk, mount the old system disk on unit Ø and write-enable the drive. If the system is configured for an RK system disk, mount the old RK system disk cartridge on RK unit 1. No special action is required for an RF system disk. One of the CILUS commands shown below is then used to replace the old CIL.

```
#DFØ:/NS:256:49/TO:4/HO/BO<DKØ:RSTS.LCL/LO  (RF11 System Disk < RK Cartridge)
#DK1:/NS:256:49/TO:4/HO<DKØ:RSTS.LCL/LO     (RK11 System Disk < RK Cartridge)
#DPØ:/NS:512:25/TO:4/HO/BO<DKØ:RSTS.LCL/LO  (RPØ3 System Disk < RK Cartridge)
#DBØ:/NS:1Ø24:13/TO:4/HO/BO<DKØ:RSTS.LCL/LO (RPØ4 System Disk < RK Cartridge)
```

If the system disk is an RF11, RPØ3, or RPØ4, the CILUS command loads the CIL and then bootstraps the RSTS/E Initialization code into memory. The INIT code prints the system name followed by the OPTION query.

The command to replace the CIL on an RK system disk mounted on unit 1 does not bootstrap the RSTS/E Initialization code. When the load operation is complete, CILUS reprints the pound sign (#) and waits

for another command. The system manager should exit from CILUS and terminate the DOS Monitor as shown below.

```
#^C                                  CONTROL/C exit from CILUS
.KI                                  "KILL" required by DOS

$FI                                  FInish for an orderly exit
TIME:  12:35:22

DOS/BATCH V9-2ØC

$
```

The processor should be halted by moving the HALT/ENABLE switch to the HALT position. Dismount both cartridges and move the RSTS/E system disk to RK unit Ø. When the disk is ready, write enable the drive and bootstrap the disk cartridge (see Section 5.2.1) to load the RSTS/E Initialization code into memory.

When the Initialization code prints the OPTION query, the system manager must reinstall all published patches using the PATCH option of the Initialization code (see Section 3.2), set keyboard defaults with the SETKEY option (Section 3.4A) and establish default start up conditions using the DEFAULT option (Section 3.5). The DSKINT option is not used since initializing a disk destroys any existing file structure. The REFRESH option (Section 3.4) may be used to verify or change the system files but is not specifically required after replacement of a CIL.

CHAPTER 3

INITIALIZATION OPTIONS

The RSTS/E Initialization (INIT) code is a collection of routines used to create the file structures, system files, and start up conditions required for normal operation of the RSTS/E system. The INIT code is essentially one large stand-alone program with many functions. Immediately after a system generation, several options must be used before the RSTS/E system can be brought up for time sharing. Thereafter, the initialization code provides the mechanism for altering critical system files and parameters as installation requirements change. INIT includes routines which ensure the integrity of the system disk file structure and provide many checks on the hardware configuration. Options are provided which enable the system to function even when certain hardware elements are inoperative. Finally, the initialization code is responsible for loading the RSTS/E Monitor and BASIC-PLUS Run Time System into memory for normal time sharing.

This chapter contains detailed information on all the initialization options and is essential to the proper and efficient use of system features. The system manager should fully understand the material presented here before any attempt is made to operate the RSTS/E system.

3.1 OVERVIEW OF INITIALIZATION OPTIONS

Whenever the RSTS/E system disk is bootstrapped, the initialization code is loaded into memory. The bootstrap operation is performed by a hardware ROM bootstrap or by software. In the system generation example of Sections 2.6.1 and 2.6.3, the SYSLOD program performed the bootstrap operation after the new system was loaded onto the RSTS/E system disk. In normal operations, the hardware bootstrap is used. When the INIT code is loaded, it prints a header line with the installation name followed by the OPTION query. The system manager must then select one of the valid initialization options listed in Table 3-1.

Table 3-1
Initialization (Start Up) Options

Long Form of Operator Response	Short Form of Operator Response	Meaning
PATCH	PA	Alter the RSTS/E System Code to correct problems.
DSKINT	DS	Initialize and optionally format a disk.
REFRESH	RE	Create or rebuild the system files in account [Ø,1] on the system disk.
SETKEY	SE	Set keyboard defaults for disabling lines and for enabling DH11 lines as local or with modem control.
DEFAULT	DE	Establish or change start up conditions.
START	ST or LINE FEED key	Start time sharing operations.
UNISYS	UN	Diagnostic aid used in conjunction with the START option to bypass the enabling of all terminal interfaces except the console interface.
BOOT	BO	Bootstrap a device.
LOAD	LO	Load a stand-alone program from the RSTS/E CIL.
ASR33 VTØ5B LA3ØS (FILL OPTIONS)	AS VT LA	Set the fill factor of the console terminal to that of the device specified.

After a system generation is performed and the Core Image Library (CIL) is loaded onto the RSTS/E system disk, the system manager must execute the PATCH, DSKINT,[1] REFRESH,[1] DEFAULT, and START options to prepare for time sharing.

The PATCH option provides a mechanism by which the RSTS/E system code can be easily altered as problems are discovered and corrected. Patches are published in the RSTS/E Installation Notes and in monthly software publications.

The DSKINT option creates the minimal RSTS/E file structure on all RSTS/E system disks. In addition, DSKINT formats a disk and performs pattern checks to detect bad disk blocks.

REFRESH creates the system files for normal RSTS/E system operation. REFRESH also provides the capability for altering system files as installation requirements change.

The SETKEY option establishes defaults for disabling keyboard lines and enabling DH11 lines as either local or with modem control. The option supplements the SYSGEN program question concerning the number of DH11 lines to enable. SETKEY is necessary for enabling modem control on DH11 lines.

The DEFAULT option establishes start up parameters such as the maximum number of jobs which can be run and the maximum size for each job. DEFAULT allows the system manager to determine the most efficient use of memory on his system. Suboptions of DEFAULT permit locating the BASIC-PLUS Run Time System in high-speed semiconductor memory and making certain failing sections of memory unavailable.

START brings the RSTS/E up to a full running state. It also permits temporary changes to the default start-up conditions mentioned above. START is responsible for loading the monitor tables, enabling terminal interfaces, loading the RSTS/E Monitor and Run Time System into memory, and starting time sharing.

---

[1]DSKINT is not used if a new RSTS/E CIL replaces an old system on a system disk where the file structure is to be preserved (see Section 2.9). Furthermore, since system and user file structures are not disturbed by the replacement, REFRESH is not required.

UNISYS is a diagnostic aid used in conjunction with the START option. Problems are often encountered with new systems due to errors in the hardware or software terminal interface configuration. The UNISYS option helps to isolate the problem to the terminal interface configuration by allowing the system to be brought up for time sharing without enabling any terminal interface except for the console device.

The BOOT option emulates the operation of the hardware bootstrap loaders. BOOT bootstraps a program or system from disk or tape devices.

LOAD permits loading and executing stand-alone programs from the RSTS/E Core Image Library (CIL) created during system generation. SYSGEN provides a way to include the stand-alone program ROLLIN (a tape and disk copy utility) in the CIL.

Finally, the fill options VTØ5, ASR33 and LA3ØS condition the keyboard driver in the INIT code to insert the proper number of fill characters when printing messages on the console terminal. These options are used only when the actual console device is something other than the console terminal specified at system generation time.

If only the RETURN key is typed in response to the OPTION query, the following message is printed.

```
OPTION:
PLEASE ENTER ONE OF THE VALID RSTS SYSTEM INITIALIZATION
OPTIONS OR TYPE 'HELP' FOR A HELP MESSAGE:
```

If the user types other than a valid response or types HELP, the system prints the following information and reprints the OPTION query.

```
OPTION: HELP
THE VALID RSTS V05 INITIALIZATION OPTIONS ARE:
        BOOT    BOOTSTRAP A DEVICE
        LOAD    LOAD AND EXECUTE A PROGRAM FROM THIS CIL
        DEFAULT ESTABLISH OR CHANGE STARTUP DEFAULTS
        START   START TIMESHARING
        SETKEY  SET KEYBOARD DEFAULTS
        DSKINT  INITIALIZE DISK TO RSTS FILE STRUCTURE
        REFRESH REFRESH YOUR SYSTEM DISK
ONLY THE FIRST TWO CHARACTERS NEED BE TYPED.


OPTION:
```

When a valid option name is entered, the system proceeds according to the option specified. Typing the CONTROL key and C key combination (CTRL/C) when any of the initialization options is asking for a response immediately interrupts execution of the option and returns the user to the OPTION query.

The following sections explain each of the initialization options in detail.

3.2 PATCHING THE RSTS/E SYSTEM - PATCH OPTION

The RSTS/E Initialization code PATCH option provides a convenient means for altering the RSTS/E system code as errors are found and corrections are published. When a RSTS/E system generation is performed, all patches are installed immediately after the Core Image File (CIL) is loaded onto the system disk. This is necessary since patches may affect the initialization code used to build required file structures, create the system files, and set up tables used during normal time-sharing. Patches are published in the RSTS/E Installation Notes if problems are uncovered after a "code freeze," but before a new release is available from Digital's Software Distribution Center. Thereafter, patches are published in the monthly Digital Software News or Software Dispatch.

The PATCH option makes permanent changes to the RSTS/E CIL on the system disk. The CIL is made up of several modules including INIT (the initialization code) RSTS (the resident monitor and device drivers), OVR (Overlay Code), ERR (error messages), and BASIC (the BASIC-PLUS compiler and Run Time System). Any of these modules may be altered using the PATCH option.

Patches take many different forms. Some are in-place patches to one or more words in one or more modules. Others require patch space in the affected modules. "MONITOR PATCH" space and "BASIC-PLUS PATCH" space are SYSGEN options recommended for all systems. If a patch requires either monitor or BASIC-PLUS patch space, but the system is not configured for the option, a new SYSGEN is needed to add the required space. Patch space is always included in the Initialization code. The overlay code can be patched using free space in overlay segments or monitor patch space. In some cases, patches affect fixed addresses and are straightforward; however in most cases it is necessary to refer to the system load maps to find the addresses of affected sections. Published patches describe the procedures required to make the alteration correctly.

The PATCH option is called by typing PATCH or simply PA in response to the initialization code OPTION query. PATCH replies by asking for a MODULE NAME (one of the five listed above), a BASE ADDRESS, and an OFFSET ADDRESS. The module name determines the CIL module to be changed. The response "BASIC" indicates that the patch

applies to the BASIC-PLUS compiler or Run Time System. The base address further determines the actual locations to be patched. For example, the base address for the Print Using section of BASIC-PLUS is found in the BASIC load map and might be entered as the response to the BASE ADDRESS query. Finally, the offset address is the first location to be changed relative to the specified base. For example, a PRINT-USING patch may begin at an offset of 100 octal bytes from the beginning of Print Using. After these items are entered, PATCH prints the old contents of the specified location and opens the word for change. PATCH opens and changes successive locations depending on the user responses.

Details for the use of the patch option are included in the two examples presented below. The first demonstrates some of the complexities of patching. The second, a simple INIT patch, emphasizes the need for using the BOOT option (Section 3.8) after any patch is made to the Initialization code.

3.2.1 Patching the RSTS/E Monitor - EXAMPLE

The following example describes the use of the PATCH option to alter the RSTS/E monitor. (Note: when the patch is made, the monitor resides on disk and not in memory.) The Initialization Code is in memory and the PATCH option allows changes to the monitor disk image before it is loaded into memory for normal timesharing.

```
OPTION:  PATCH                                  EXAMPLE ONLY:
MODULE NAME ? RSTS                              NOT A REAL PATCH
BASE ADDRESS ? [NAME]
OFFSET ADDRESS ? 120
MODULE   BASE  OFFSET  OLD     NEW?
RSTS    [NAME] 000120 104760 ? 004737
RSTS    [NAME] 000122 103364 ? [PATCH]+20    OCTAL ADDITION.
RSTS    [NAME] 000124 005062 ? 102637
RSTS    [NAME] 000126 012762 ? [JBSTAT]-2    OCTAL SUBTRACTION.
RSTS    [NAME] 000130 004737 ? <LF>          NO CHANGE.
RSTS    [NAME] 000132 104726 ? 104730
RSTS    [NAME] 000134 010423 ? ↑C            CONTROL/C EXIT.

OPTION:  PATCH
MODULE NAME ? RSTS
BASE ADDRESS ? [PATCH]                          FROM RSTS LOAD MAP
OFFSET ADDRESS ? 20
MODULE   BASE  OFFSET  OLD     NEW?
RSTS   [PATCH] 000020 000000 ? 010203
RSTS   [PATCH] 000022 000000 ? 011104
RSTS   [PATCH] 000024 000000 ? 000207
RSTS   [PATCH] 000026 000000 ? ↑C            CONTROL/C EXIT

OPTION:
```

All numbers printed by the PATCH option and all numeric responses are octal. In the example, the notation [NAME] indicates an address that must be found in a load map or a quantity that must be computed. PATCH does not perform any arithmetic; hence, expression of the form [NAME] + 20 must be manually calculated using 2's complement arithmetic. (If unfamiliar with the octal representation of binary numbers or with 2's complement arithmetic, consult a Software Support Representative.) As PATCH opens successive locations, it prints the current or old location contents and waits for new data to be entered as an octal word. A carriage return (CR) is used to enter the new data. PATCH then sequences to the next location. A line feed <LF> with no new data causes PATCH to sequence to the next location without altering the current location. PATCH continues to open successive locations until the CONTROL/C combination is typed. CTRL/C returns to the initialization code OPTION query.

Note that changes are made immediately upon typing the carriage return key. If an error is made it becomes necessary to reenter the PATCH option to correct the mistake. Printing the old contents of a location provides a check for proper placement of a patch. If the old contents of any location shown in a published patch are not identical to those printed by the PATCH option, all locations should be restored to their old contents. This may indicate an error in the use of load maps or an error in the published patch itself. Finally, a complete patch may be double checked by reentering the PATCH option and using the line feed key to examine successive locations.

3.2.2 Patching the Initialization Code - EXAMPLE

Patching the Initialization Code is usually simpler since INIT is the same for all systems. There is usually no need to refer to a load map unless the value of a global parameter is needed for the patch. Patches are made to the CIL on disk and not in core. This is an important distinction when patching INIT since the in-core copy (which is running) is not changed by the PATCH option. It is necessary to use the BOOT option (described fully in Section 3.8) to load the altered INIT code into memory. The example below illustrates the procedure for making an INIT patch.

```
OPTION:  PATCH                                  EXAMPLE ONLY -
MODULE NAME ? INIT                              NOT A REAL PATCH
BASE ADDRESS ? 67472
OFFSET ADDRESS ? 4724
MODULE   BASE  OFFSET  OLD    NEW?
INIT    067472 004724 100200 ? 104200 SINGLE WORD PATCH.
INIT    067472 004726 XXXXXX ? <LF>   OLD CONTENTS VARIABLE.
INIT    067472 004730 005766 ? <LF>   PRINTED FOR VERIFICATION ONLY.
INIT    067472 004732 001000 ? ↑C     CONTROL/C EXIT.

OPTION:  BOOT                   BOOT REQUIRED TO LOAD ALTERED
                                INIT CODE INTO MEMORY

BOOT DEVICE ? <LF>              LINE FEED BOOTS THE SYSTEM DISK

RSTS V05B-24 TEST SYSTEM

OPTION:
```

3.3 DISK INITIALIZATION - DSKINT OPTION

The DSKINT option initializes all disks (system, public, and private) to be used on the RSTS/E system. DSKINT writes the minimal RSTS/E file structure on the disk. Hence, all disks, except auxiliary swapping disks, must be initialized prior to use. DSKINT destroys any existing file structure on a disk and should not be used on any RSTS/E disk if the file structure is to be preserved.

The minimal file structure written on all disks consists of the following elements:

(1) The MFD account [1,1] for the disk,

(2) The UFD for the system account [0,1] on the disk,

(3) Two files for the system account [0,1] of the disk: BADB.SYS and SATT.SYS, and

(4) A public or private disk designation for the device.

The initialization of a system disk differs slightly from the initialization of non-system disks. When the RSTS/E system disk is initialized, the Core Image Library (CIL) is preserved through the DSKINT process. Several additional elements are also included in the minimal file structure. These include:

(1) The system file RSTS.CIL under the system files account [0,1] which maps the Core Image Library (CIL).

(2) The entry in the MFD for the system library account [1,2]

The REFRESH option described in Section 3.4 creates the other required system files on the system disk. The BUILD process described in Chapter 4 installs all the system programs under the library account [1,2] on the system disk. Finally the REACT system program enters user accounts into the MFD on all disks.

In addition to initializing the disk with a minimal file structure, the DSKINT routines allow the user to format the disk and to specify the number of patterns used to check for bad blocks. The system manager must format any non-system disk before it is used on the RSTS/E system. Formatting the disk causes all the necessary timing and sense marks to be written on the disk and erases any extraneous information from the disk. (Formatting applies only to RK and RP type removable disks.) The system disk which contains the RSTS/E CIL should not be formatted after the CIL is loaded by SYSLOD or CILUS since the formatting operation would destroy the CIL.

The DSKINT option allows the system manager to choose from one to eight possible patterns to check for bad blocks on a disk. A bad block is a portion of the disk to which information is written but from which that same information cannot be read. It is recommended that the system manager specify a minimum of two patterns; he must specify at least one pattern.

The specification of a larger number of patterns increases the probability that all bad blocks are detected during the initialization of the disk and decreases the possibility of loss of valuable system information which is later written on the disk. The possibility of loss is less since clusters containing detected bad blocks are allocated to a file called BADB.SYS stored under account [0,1] on the disk. These clusters are never allocated for any other purpose.

The minimal file structure written by the DSKINT routines enables the disk to be used under either the public or the private structure of disks in the RSTS/E system. The system manager must, however, designate to the DSKINT routines whether the disk is to be public or private. Public and private disks are discussed in Chapter 12 of the BASIC-PLUS Language Manual. Structurally, the only difference between a public and a private disk is that a bit (termed the private bit) is set in the MFD of a private disk. Operationally, however, an important distinction is made by the RSTS/E system between a private and a public disk.

During time sharing operations all disks within the public structure must be readied, write enabled, and logically mounted. The system treats the entire public structure as one general, default reference. A user's file could reside on any one of the physical devices within the public structure and, therefore, all devices in the public structure must be available to ensure successful file access.

In addition, if a user creates a file, the directories of each public disk are searched to ensure that a file of the same name does not already exist. The system manager makes a public disk available by implementing the guidelines in Section 6.1 for mounting disks automatically at start up time.

Disks within the private structure, however, need not always be present during time sharing operations. Access to files on a private disk requires that the user refer specifically to a distinct physical device. To create or access a file on a private disk, only the directory of that device need be searched. As a result, private packs can be interchanged during time sharing operations if the proper disk management procedures are followed. (Refer to Section 6.3.2 for the discussion of disk management procedures.)

The system manager must decide how much of his disk resources to commit to the public structure or to the private structure. Disks are designated public or private only when they are initialized. To change the designation, any useful contents of the disk must be preserved elsewhere and the disk must be reinitialized. If the disk is initialized as a private disk, the system manager must enter user account information in the MFD of the disk with the REACT system program, so that files can be created on the disk. (REACT is described in Chapter 6.)

3.3.1 Using DSKINT

When the RSTS/E initialization routines are executed, they print the system installation name followed by the OPTION query. The system manager types DSKINT or simply DS to request the disk initialization option and then types the current date and time in response to two successive queries. The following sample dialog illustrates the procedure.

```
OPTION: DSKINT
DD-MMM-YY? 25/JAN/75
ILLEGAL DATE ENTERED; PLEASE TRY AGAIN
DD-MMM-YY? 25-JAN-75
HH:MM? 12:15

DISK?
```

Subsequent DSKINT queries request the user to enter the disk type and unit, pack identification and pack cluster size, and the passwords and cluster sizes for the MFD and library accounts. The DSKINT queries are presented for reference in Table 3-2.

If the routines determine that a response is incorrect or unacceptable, the query is reprinted. The queries are printed in a short form to save time. If the system manager types the RETURN key in response to any of the queries, DSKINT prints a longer, explanatory form of the query or, in some cases, the acceptable responses.

During the formatting of an RP03 type disk, messages are printed informing the user to set and disable a format enable switch. This switch is the FORMAT ENABLE/NORMAL switch and is located behind the third panel below the bank of indicator lights on the RP11-C controller. It is the switch farthest to the left. When instructed to set the FORMAT ENABLE/NORMAL switch, the user sets it to the FORMAT ENABLE position; when instructed to disable the switch, he sets it to its NORMAL position.

If bad blocks are uncovered during a pattern test, a table is printed indicating the sector and cluster number (in decimal) in which a bad block exists and the contents of the disk hardware register (in octal) at the time of the error. If an excessive number of errors are encountered, a fatal error message is printed. In such a case, it is advised that another disk be used.

The completion of disk initialization is signalled by printing the OPTION query. The system manager then builds the system files on the initialized system disk using REFRESH as described in Section 3.4. The examples contained in the two following sections illustrate the use of DSKINT for system and non-system disks.

Table 3-2

DSKINT Queries

Query	Description of Response
DD-MMM-YY?	Type the current date in the exact format shown. For example, 3-FEB-75.
HH:MM?	Type the current, 24-hour time, which is used to start the Monitor clock, in the exact format shown. For example, 13:52.
DISK?	Type two characters which indicate the type of the disk being initialized. Acceptable entries are RF, RS, RK, RP (RPØ3), or RB (RPØ4).
UNIT?	For RS, RK, RP and RB type disks. Type the physical unit number on which the disk, DECpack, or disk pack resides. Acceptable entries are Ø through 7, inclusive.
PLATTERS?	For RF type disks. Type the number of RS11 disks (platters) connected to the RF11 controller. Acceptable entries are 1 through 8, inclusive.
PACK ID?	Type up to six alphanumeric characters which are used when logically mounting or dismounting the device. (See the description of the MOUNT and DISMOUNT commands in Section 6.3.2.1.)
PACK CLUSTER SIZE?	Type the decimal number of 256-word blocks which each cluster allocated on the disk will contain. Clusters are described in Section 5.4.3. Acceptable pack cluster sizes are 1, 2, 4, 8, or 16 for RF, RS, and RK disks. For RPØ3 disks, acceptable values are 2, 4, 8, or 16. For RPØ4 disks, acceptable values are 4, 8, or 16.
MFD PASSWORD?	Type up to six alphanumeric characters which become the password of account [1,1] on the device.
MFD CLUSTER SIZE?	Type the decimal number of 256-word blocks which a cluster allocated to the MFD will contain. Must be equal to or greater than the pack cluster size. The number of user accounts which can be created is approximately 108* MFD cluster size. The MFD account is described in Chapter 5.

(continued on next page)

Table 3-2 (Cont.)
DSKINT Queries

Query	Description of Response
PUB, PRI, OR SYS?	Type PUB to designate the disk as public or type PRI to designate the disk as private. Type SYS to initialize a system disk. CAUTION: Initializing a system disk as PUB or PRI destroys the CIL.
LIBRARY PASSWORD?	Query is printed only when DSKINTing a system disk. Enter up to 6 alphanumeric characters which become the password of the library account [1,2].
LIBRARY UFD CLUSTER SIZE?	Printed only when DSKINTing a system disk. Type the decimal number of 256-word blocks allocated for each of the seven possible UFD clusters for the library account [1,2]. The number of files which can be stored under any account is approximately 72 * UFD cluster size. Library UFD cluster size must be greater or equal to pack cluster size.
FORMAT?	Type Y to write hardware timing and sensing information on the disk or N to omit same. If Y is typed, messages are printed informing the user of the start and end of the format pass. (If the disk is an RP03 type, the user is told to set and disable the format switch located on the RP11-C controller.) All new DECpack cartridges and disk packs must be formatted. System disks should never be formatted after the CIL has been loaded.
PATTERNS?	Type a decimal number between 1 and 8 to choose the number of patterns used to check for bad blocks. Type the RETURN key only to print the time required to execute each pattern according to device type.
PROCEED (Y OR N)?	Type Y to proceed with the execution of the initialization. Type N to abort initialization and return to the OPTION query. This query is printed to allow the system manager to double check the responses to the queries.

3.3.2 Initializing the System Disk

After the stand-alone program SYSLOD writes the CIL onto the RSTS/E system disk and bootstraps the initialization code into memory, the DSKINT option is used to initialize the newly formatted and otherwise blank system disk. Initialization of a newly created system disk is a unique process and is described solely in this section.

CAUTION

> DSKINT destroys any existing file structure on a disk and must not be used on any RSTS/E disk if the file structure is to be preserved. If SYSLOD or CILUS is used to write a new RSTS/E CIL onto an old system disk and the existing file structure is to be preserved, the DSKINT option is not used.

When the SYSLOD or CILUS run terminates, a new system disk contains only a contiguous CIL which is loaded beginning at sector 49 (for RF or RK type disks) or sector 5Ø for RPØ3 type disks and sector 52 for RPØ4 type disks. The CIL extends for as many blocks as needed to contain the RSTS/E code. Sector Ø of the system disk contains a bootstrap record and sectors 1 through 48 or 49 are reserved for directories and other system files. The pure code of the CIL must be incorporated into a structure which enables the RSTS/E system to access and manipulate its contents. The DSKINT initialization option creates the minimal RSTS/E file structure including a file which maps the CIL on the system disk (RSTS.CIL).

Under normal circumstances, initialization destroys all data on a disk. However, the CIL is preserved if it was bootstrapped into memory. The system disk may be bootstrapped from SYSLOD or CILUS (/BO switch) or by using the hardware bootstrap. The bootstrapping operation provides pointers to the initialization code which are used to determine the size and location of the CIL. With this information the CIL (but not user files) is preserved through the DSKINT process.

The following is a sample dialog, each line of which is denoted by a letter and explained in the subsequent text.

```
OPTION: DSKINT
DISK ? RP                                   (line a)
UNIT ? Ø                                    (line b)
PACK ID ? SYSPAK                            (line c)
PACK CLUSTER SIZE ? 2                       (line d)
MFD PASSWORD ? SYSMFD                       (line e)
MFD CLUSTER SIZE ? 4                        (line f)
PUB, PRI, OR SYS ? SYS                      (line g)
LIBRARY PASSWORD ? SYSLIB                   (line h)
LIBRARY UFD CLUSTER SIZE ? 4                (line i)
PATTERNS ? 2                                (line j)
PROCEED Y OR N ? Y                          (line k)
PATTERN # 2
PATTERN # 1

OPTION:
```

At line a, the system manager types a two-letter designator of the disk that was bootstrapped; valid designators are RF, RK, RP, or RB. The query at line b for an RK, RP, or RB system disk requires the unit number of the device on which the system disk is mounted. Since only unit Ø of a device can be bootstrapped, Ø is the only valid response. For an RF system disk, the query UNIT is replaced by the query PLATTERS, to which the system manager responds by typing the number of RS11 disks (platters) connected to the RF11 controller.

The system manager types from one to six alphanumeric characters in response to the query at line c. The characters typed are called the pack identification or pack label and are used internally by RSTS/E as the password of system account [Ø,1].

The system manager must specify, at line d, the pack cluster size for the system device. Pack cluster sizes for an RF11 disk or an RKØ5 or RKØ3 cartridge can be 1, 2, 4, 8, or 16, or, for an RPØ3 disk pack, 2, 4, 8, or 16. For an RPØ4 disk, pack cluster sizes are 4, 8, or 16. In general, larger pack cluster sizes permit faster access to data stored on the disk at the expense of possible wasted disk space. Smaller pack cluster sizes permit more efficient

allocation of disk storage space at the expense of more frequent access to retrieval information (pointers to file data) stored in user file directories. The latter situation is improved by clustering individual files and user file directories (UFDs) at some cluster size greater than the pack cluster size. Therefore, the minimal cluster size is recommended for almost all installations. (A more detailed discussion of clustering is presented in Section 5.4.3.)

The system manager types from one to six alphanumeric characters in response to the query printed at line e. The characters are used as the password of the MFD account [1,1] on the system device. The MFD is described in Section 5.4.1. The MFD password should be kept secret, since irresponsible access to the Master File Directory or for that matter, to any privileged account, can destroy system software.

At line f, the DSKINT routines request the cluster size of the MFD account [1,1]. Cluster sizes are described in Section 5.4.3. With the restriction that no cluster size be less than the pack cluster size, the MFD cluster size can be 1, 2, 4, 8, or 16. The DSKINT routines prevent the use of any unacceptable values. The cluster size of the MFD on the system disk determines the maximum number of user accounts that the system manager can create. It is recommended that the MFD cluster size be the same as the pack cluster size unless the system requires a larger number of user accounts that that value allows. The maximum number of user accounts allowed is approximated by multiplying the MFD cluster size by 108.

The system manager must type SYS in response to the query shown at line g to specify initialization of a system disk. The queries at lines h and i concern the password and UFD cluster size of the system library account [1,2]. The password can be up to six alphanumeric characters.

It is recommended that the system manager set the UFD cluster size of the account [1,2] to a minimum value of 4. A larger cluster size may be set if the system manager intends to increase the system library files above those supplied by DIGITAL and those required to build the system library. (The total number of files created under account [1,2] as a result of following the procedures described in

Chapter 4 is approximately 6Ø.) The maximum number of files allowed by any account is approximated by multiplying the UFD cluster size by 72.

The query at line j requests the number of patterns (between 1 and 8) to be used when the disk is tested for bad blocks.

The query at line k is printed as a safety measure to allow the system manager to make final checks and to ensure that he desires to proceed. If N is typed, the DSKINT routines are not executed. The initialization routines return to the OPTION query. If Y is typed, DSKINT checks for bad blocks as shown in the example.

3.3.3 Initializing a Non-System Disk

DSKINT is used to initialize, format, and pattern check non-system disks on the RSTS/E system. System disk initialization is described in Section 3.3.2. The examples below apply only to non-system disks.

The following example initializes an RK cartridge as a private disk:

```
OPTION: DSKINT
DD-MMM-YY? 22-JUL-74
HH:MM? 11:30

DISK ? RK
UNIT ? 1
PACK ID ? MYPACK
PACK CLUSTER SIZE ? 1
MFD PASSWORD ? MYMFD
MFD CLUSTER SIZE ? 1
PUB, PRI, SYS ? PRI
FORMAT ? YES
PATTERNS ? 4
PROCEED (Y OR N) ? Y

STARTING RK FORMAT PASS
END RK FORMAT PASS

PATTERN # 4
PATTERN # 3
PATTERN # 2
PATTERN # 1


OPTION:
```

3.3.4 Using DSKINT to Pattern Check a Swapping Disk

The DSKINT option may be used to perform a basic reliability test on RF11, RS03, and RS04 disks configured as swapping disks on the RSTS/E system. Swapping disks do not require the minimal file structure created by DSKINT. Using DSKINT for this purpose writes the minimal file structure on the swapping device but is useless for, and is overwritten, under normal time sharing operations.

During initial hardware installation or if a hardware malfunction is suspected, DSKINT can supplement the tests performed by standard diagnostic methods. DSKINT does not duplicate all the tests performed by diagnostic programs and should be used only in addition to standard hardware maintenance procedures.

The example below depicts pattern tests on RS03 drive unit 2. Note that DSKINT does not ask whether the drive is an RS03 or RS04. The initialization code reads the drive type registers for RS03 and RS04 units to determine the type and size of the device (RS03 = 1024 256-Word segments, RS04 = 2048 256-Word segments).

```
OPTION: DSKINT
DD-MMM-YY? 22-JUL-74
HH:MM? 12:00

DISK ? RS
UNIT ? 2
PACK ID ? TEST
PACK CLUSTER SIZE ? 1
MFD PASSWORD ? TEST
MFD CLUSTER SIZE ? 1
PUB, PRI, OR SYS ? PUB
PATTERNS ? 8
PROCEED (Y OR N) ? Y

PATTERN # 8
PATTERN # 7
PATTERN # 6
PATTERN # 5
PATTERN # 4
PATTERN # 3
PATTERN # 2
PATTERN # 1


OPTION:
```

If bad blocks are detected on a swapping device (denoted by a table of bad blocks being printed), the swapping disk should normally not be used. Call DEC field service to locate the hardware problem. As a temporary measure, the swap files can be moved to the system disk or several swap files can be used to avoid the malfunctioning portion of the swapping device. Swap files are discussed in Section 3.4.2.4.

3.4 REFRESH OPTION - CREATING THE SYSTEM FILES

Normal RSTS/E system operation requires certain system files under the system files account [0,1]. The DSKINT option creates the system files account and two system files (BADB.SYS and SATT.SYS) required on all disks. When DSKINT is used to initialize a system disk, a third system file (RSTS.CIL) is also created to map the RSTS/E Core Image Library. The remaining system files are created by the REFRESH option of the Initialization code.

Table 3-3 lists the system files with a brief description of each. An overview of the REFRESH option is included below to emphasize the importance of using the option to full advantage when the system disk is first built. Detailed discussions of each system file is then presented to provide background information for proper use of the option. Finally, the details of using REFRESH are presented with several examples.

Table 3-3

Contents of System Account [0,1]

File Name	Location	Status	Description
BADB.SYS	Each disk device in system except swapping disks.	Required	Created by the initialization option DSKINT; all clusters containing bad disk blocks detected during disk initialization are allocated to this file to prevent possible loss of data.
RSTS.CIL	System disk only	Required	RSTS/E Core Image Library containing resident and non-resident (overlay) code, basic + run time system, error messages, initialization code, and standalone programs such as ROLLIN.
SATT.SYS	Each disk device in system except swapping disks	Required	Storage allocation table. This file is a bit map used to control allocation of disk space.
SWAPØ.SYS	System or swapping disk only	Required	Files used to store user job images not in core.
SWAP1.SYS SWAP2.SYS SWAP3.SYS	System or swapping disk only	Optional	Auxiliary swap files used to store additional user job images not in core.
OVR.SYS	System or swapping disk only	Optional	Copy of non-resident (overlay) code from CIL (RSTS.CIL).
ERR.SYS	System or swapping disk only	Optional	Copy of error messages from CIL (RSTS.CIL) which can be accessed and modified by system manager.
BUFF.SYS	System or swapping disk only	Required for DECtape	File to retain DECtape directories during processing.
CRASH.SYS	System disk only	Optional	File used to store exact image of valuable contents of core following a system crash.

3.4.1 Refresh Option Overview

The REFRESH option allows the system manager to create, modify, reposition, and, in some cases, delete system files stored under the system files account [Ø,1]. The system files reside on the system disk or on auxiliary swapping disks. (In either case, these files are catalogued by the directory for account [Ø,1] on the system disk.) The REFRESH routines assume that the system disk contains at least the minimum RSTS/E file structure as written during the initialization of the system disk using the DSKINT option. The REFRESH operation verifies that the entire disk structure is intact, but only the files stored under account [Ø,1] are manipulated. Thus, it is possible to refresh a system disk which contains many files and accounts without destroying the existing files.

The REFRESH option must be used to initially create the required system files. This is done immediately after the system disk is initialized with DSKINT and all patches have been installed. Proper use of REFRESH when the system disk is created ensures that changes can be made to the system files at a later date.

Before REFRESH is used, the system manager must understand the principles in the construction of the system disk. The logical structure of the system disk is such that a linear string of unallocated space exists on the disk prior to loading the CIL. When the RSTS/E CIL is written on the system disk, it occupies the lower physical portion of the disk. Initializing the system disk writes a minimum RSTS/E file structure on the disk and assimilates the pure code of the CIL into the minimum file structure.

If the hardware includes one or more swapping disks, these devices are considered logical extensions of the system disk. Figure 3-1 shows how disk sectors are numbered for several possible system and swapping disk configurations. Swapping disks do not require a file structure or directory information to be on the disk. All such information is included in the directories on the system disk. To allocate space for system files on the swapping devices, the initialization code temporarily extends the system disk storage allocation table (SATT.SYS) to include the space on the configured swapping devices. Only the REFRESH option manipulates swapping disk space. When REFRESH completes manipulation of the system files, the allocation table is updated on the system disk. Once written back

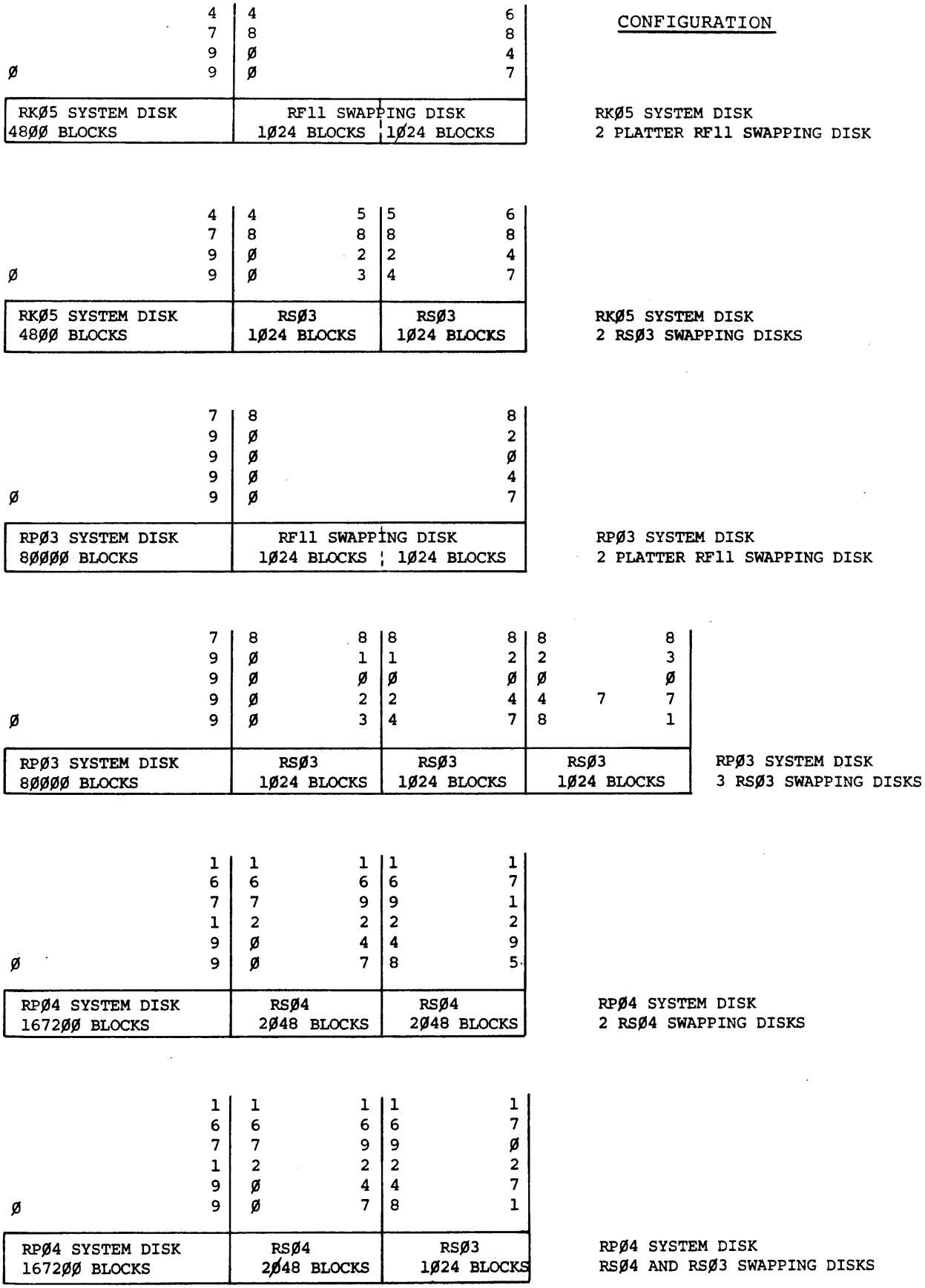


Figure 3-1
Disk Sector Numbers

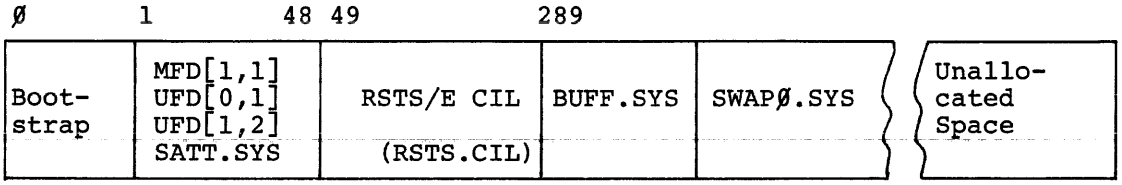
out to the system disk, the allocation table does not map any space on the swapping disks regardless of whether the space is free or allocated. REFRESH allocates space on swapping devices based on calculated sizes of configured swapping disks. The physical devices are never accessed during REFRESH. Hence the swapping disks need not be physically present for normal REFRESH operation.

The size of RC11 and RF11 swapping disks vary depending on the number of platters. Each RC platter is 64K words or 256 segments; Each RF platter is 256K words or 1024 segments. RF11 and RC11 disks are considered one unit regardless of the number of platters. Hardware permits data transfers to continue across these boundaries. The total space available is the sum of the platters sizes. RSTS/E supports the hardware maximum or eight RF platters or four RC platters.

RS03 and RS04 fixed-head disks are treated as separate units when used for swapping. The system files must fit on a unit and may not cross unit boundaries. Each RS03 is 256K words or 1024 segments; each RS04 is 512K words or 2048 segments. If a system contains both RS03s and RS04s, all RS04s logically follow the system disk and all RS03s follow the RS04s. Physical unit numbers need not be known to use REFRESH. Physical-to-logical unit mapping is done at Start up time. These distinctions are important when trying to locate system files on particular RS units. RSTS/E supports only four RS disk units for swapping but additional units may be attached to the RH11 controller for use by other operating systems.

If the REFRESH operation proceeds without the system manager's requirements, only required system files are created on the system disk. These files are allocated the minimum required storage space in disk sectors immediately following the space occupied by the CIL. A typical representation appears as follows:

ORIGIN
Sector



Such distribution of the required system files is seldom optimal. For example, the default size of SWAPØ.SYS can handle only the configured number of jobs at the default size of 8K words. Most RSTS/E systems will run BASIC-PLUS programs larger than 8K. Furthermore, REFRESH locates all the system files on the system disk, any swapping disks are not used.

At the time the system manager builds the system disk, he has available all the contiguous unallocated storage space. If he does not use this file space, he may be limited by the existing structures at a later time, since time sharing operations cause storage space on the system disk to be allocated randomly on an as-needed basis. The free contiguous space at the front portion of the system disk can be allocated to user and system program and data files, thus limiting and restricting later structural changes to the system files in account [0,1] on the system disk.

A few examples can show the significance of fully using REFRESH. The RSTS.CIL file is created at the minimum required size during the DSKINT process. (The minimum required size is actually rounded up to span an integral number of disk clusters, as determined by the pack cluster size.) The contiguous storage space at the lower portion of the system disk immediately following the RSTS.CIL file is allocated to whatever other system files are created by REFRESH. During time sharing operations, subsequent contiguous storage space is allocated to user and system program and data files. Future expansion of storage space occupied by CIL becomes restricted.

If the system manager wishes to augment the system at some future date with either new hardware or software features, the size of the CIL will necessarily increase by an undetermined amount. The probability of successfully incorporating a newly generated CIL into an existing file structure is greatly improved if the system manager increases the size spanned by the CIL when the system files are initially built on the system disk. The REFRESH option provides this capability. Thus, to allow for future changes, the system manager specifies from 20 to 40 extra blocks for the RSTS.CIL file. The extra contiguous space is then allocated to the CIL file at the lower portion of the disk, allowing for possible future, successful incorporation of a new CIL without destroying system and user files on the system disk.

For certain system files, the position of those files on the disk influences the efficiency of the system operation. For systems having a moving head system disk without a swapping disk, the system manager can position certain system files in the center portion of the disk. This appreciably reduces the average amount of SEEK time since the moving heads are not operating back and forth from opposite ends of the disk. Again, allocation of oversize files when the disk is built ensures that future growth can be accommodated. An attempt to position the files on the disk later at an optimal location becomes increasingly difficult as more of the contiguous file space is allocated during time sharing operations.

The system manager has the capability to structure the system files by responses to the many queries printed by REFRESH. The response to each file-related query determines how and where each system file is accessed and positioned. REFRESH queries are described in Section 3.4.3.3. The next section provides detailed information on each of the system files.

3.4.2 System Files

The system manager must know the purpose of each system file (Table 3-3) to use REFRESH properly. This section contains more detailed information on each of the system files noting the operations performed by REFRESH.

3.4.2.1 BADB.SYS - Bad Block File

The bad block file permanently allocates bad disk segments and makes them unavailable for use by system and user files. The bad block file is required on all disks (except swapping disks) used under RSTS/E. The file length can be zero if there are no known bad blocks. BADB.SYS is created during the pattern checks performed by the DSKINT option. A bad block is defined as a disk segment to which data can be written but the same data cannot be retrieved. Disks are usually not guaranteed to be error free. Hence, it is important to perform the pattern tests when the disk is initialized. If bad blocks are found, the clusters (equal to the pack cluster size) containing the bad blocks are allocated to the bad block file. REFRESH only prints a list of bad blocks: the capability for adding blocks to the bad block file has not yet been implemented.

3.4.2.2 RSTS.CIL - Core Image Library

The RSTS.CIL file contains the initialization code, monitor code, BASIC-PLUS compiler and Run Time System, overlay code, error messages, and usually the stand-alone program ROLLIN. This file exists only on the system disk. The CIL is loaded during system generation by either the DOS/BATCH program CILUS or the stand-alone program SYSLOD. When the system disk is initialized using DSKINT the CIL is preserved and mapped into the RSTS.CIL file.

The RSTS.CIL file cannot be moved by REFRESH and will always be located on the lower physical portion of the system disk. DSKINT creates the RSTS.CIL file at its minimum required size in the range of 250-350 disk segments of 256 words each. The size of the CIL depends on the monitor and BASIC-PLUS options included. As stated earlier, the size of the RSTS.CIL file can be increased by REFRESH to allow replacement of the CIL at a later time without disturbing the existing file structure on the system disk. The system manager performs new system generations as the installation requirements change. Hence, the CIL file should be made large enough to cover any foreseeable additions to the system. At the time of the initial system generation, increasing the size of RSTS.CIL by 20 to 40 segments over the required size allows for replacement by a substantially larger monitor. If the contiguous file space immediately following the RSTS.CIL file is allocated to user files under normal time sharing, then it may not be possible to increase the size of the CIL at a later time.

3.4.2.3 SATT.SYS - Storage Allocation Table

The storage allocation table is a bit map used to keep track of disk space. Each bit in the map represents one disk cluster of n 256-word segments where n is the pack cluster size. A one bit represents an allocated (in use) cluster and a zero bit represents a free cluster. The SATT.SYS file is required on all disks (except swapping disks) and is created by DSKINT. The disk space mapped by the SATT.SYS file on each disk includes only the actual space on that disk. The size of the file varies depending on disk size and the pack cluster size. The maximum size is 10 segments for all disk devices currently supported under RSTS/E.

REFRESH will permit repositioning of the SATT.SYS file on a moving head system disk. This capability is primarily used when no swapping disk exists and the swap files and other system files must also be located on the moving head device. Grouping all system files near the center of the disk tends to reduce average seek time for the moving heads.

3.4.2.4 SWAPn.SYS - SWAP Files

During normal timesharing operations, user job images are frequently moved (swapped) to disk in order that other jobs can be loaded into memory and executed. The REFRESH option is used to create the required swap areas or swap files according to installation requirements. Under RSTS/E there are four possible swap files called SWAPØ.SYS, SWAP1.SYS, SWAP2.SYS, and SWAP3.SYS. These system files may be located on the system disk, on auxiliary swapping disks, or on combinations of the system and swapping disks. Guidelines for creating an efficient swapping structure and the RSTS/E swap algorithm are described below. The material presented here should be fully understood before REFRESH is used to create the swap files.

SWAPØ.SYS is the only required swap file. However, other swap files may be needed depending on the hardware configuration and the installation requirements. The sizes of existing swap files limit the values of JOBMAX and SWPMAX which can be specified when using the DEFAULT and START initialization options. JOBMAX is the maximum number of jobs which can be run simultaneously. JOBMAX must always be less than, or equal to the maximum number of jobs configured during system generation. SWPMAX may be set to any number between 8 and 16(K) words. For the purpose of calculating the size of swap files, no provision is made for limiting certain jobs or even certain numbers of jobs to less than the system wide SWPMAX. Thus if any job is allowed to expand to 16K, the amount of swap space reserved for all jobs must be 16K each.

Each of the swap files holds an integral number of user job images determined by the size of the file and the desired SWPMAX. The following equations describe the relations between size of the swap files, SWPMAX, and JOBMAX.

$$\text{JOBS for SWAPn.SYS} = \frac{\text{Size of SWAPn.SYS in segments}}{\text{(Desired SWPMAX(K) * 4)}} \quad \text{(truncated)}$$

If more than one swap file is created, the total number of jobs which can be run simultaneously is equal to the sum of jobs accommodated by each swap file.

$$\text{JOBMAX} = \sum_{n=0}^{3} \text{JOBS for SWAPn.SYS} \leq \text{Configured number of jobs.}$$

The RSTS/E monitor uses four swap bit maps (one for each of the four possible swap files) to control allocation of swap space. Each bit map contains 4 words or 64 bits and each bit represents a swap "slot" of size SWPMAX. In the bit maps, a zero bit represents an allocated slot and a one bit represents a free slot. Bit Ø (out of 64) in each map is always Ø (allocated). Thus each swap file and its corresponding map can handle the theoretical maximum of 63 jobs, or up to 63 jobs could be spread over 1 to 4 swap files (see Figure 3-2).

Swap slots are dynamically allocated when a job is swapped out to disk. The bit map is scanned forward for actively running jobs. Hence, if SWAPØ.SYS is located on the fastest swapping device, active jobs will be swapped to the fastest medium. If a job is inactive, the bit map scan is reversed. Hence, inactive jobs will be swapped to the higher numbered swap files.

This mechanism is used for several reasons. First, to handle multi-unit swap structures such as multiple RS03 and/or RS04 fixed head disks. RS03s and RS04s are addressed by unit in the same way that RP03 or RK05 drives are referenced by unit. A data transfer may not continue across a physical unit boundary. The RF11 and RC11 fixed head disks are addressed as one unit regardless of the number of platters. For these devices, a data transfer may continue across platter boundaries. Secondly, this method allows systems which only occasionally run many jobs, to operate with less fast swapping space. Jobs which would overflow swapping space on the swapping disk(s) are simply swapped into a file on the system disk.

To demonstrate proper and efficient use of the swap files, several systems are considered. The first system requires only a single swap file to satisfy system requirements. The second case is when two swap files are used. The third is a large system that must use three swap files.

For the first example, consider a system that runs 15 jobs with the maximum size of 16K words. The required swap space in disk segments is calculated as

SWAP SIZE = JOBMAX * SWPMAX (in K) * 4 (SEGMENTS PER K)

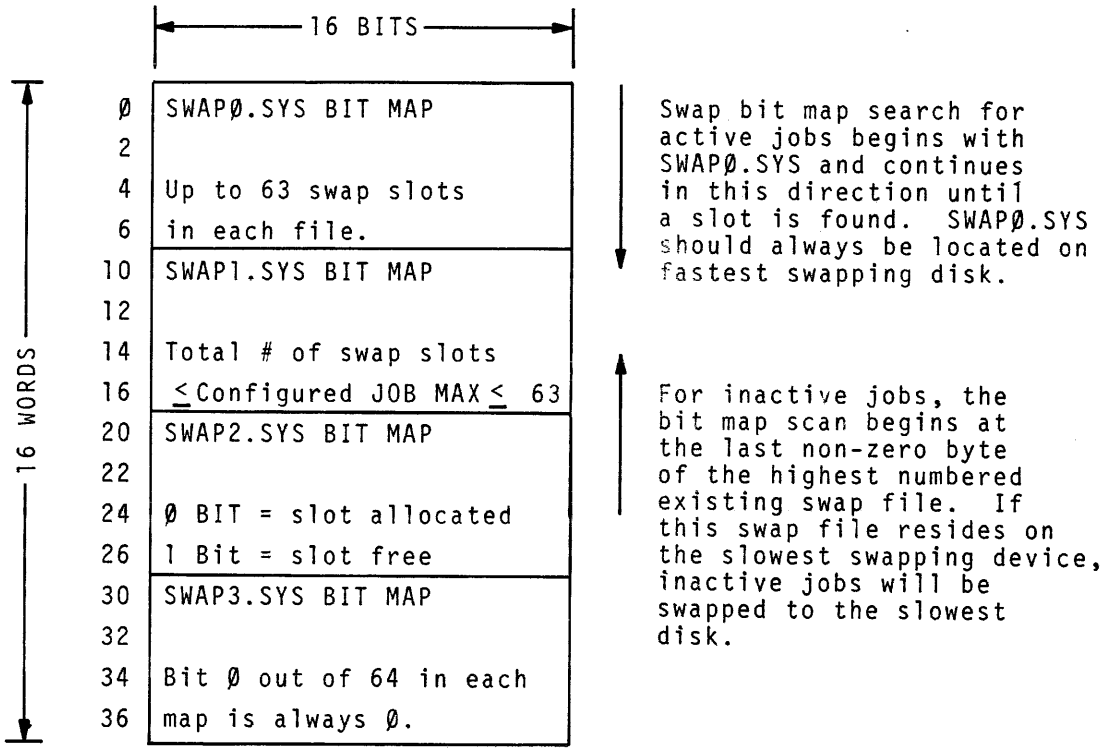


Figure 3-2. SWAP Bit Maps

In this case 15 jobs at 16K requires 960 disk segments. Also assume that the hardware configuration includes a single platter RF11 disk (1024 segments) as the swapping device. Note that the same numbers apply for a single RS03 unit which is also 1024 segments. Since the required swap space is available on the swapping device, a single swap file is sufficient to meet installation requirements. The REFRESH option is used to create SWAPØ.SYS on the swapping disk at size equal to 960 segments.

It is also advantageous to locate other system files including OVR.SYS (overlay code), ERR.SYS (error messages), and BUFF.SYS (DECtape buffers) on the swapping device to speed up access to these files. In the example above, 64 disk segments remained on the swapping disk. This is more than enough space to hold the overlay code, error messages, and any required DECtape buffers. These system files would be created and located on the swapping disk using the REFRESH option.

If the installation requirements for the system described above are increased to 20 jobs at 16K, the amount of swap space required would increase to 1280 disk segments. With the physical limit of 1024 segments on the RF11 disk, a second swap file must be used to accommodate the overflow. The system files OVR.SYS, ERR.SYS, and BUFF.SYS could be moved to the system disk to free up space for one additional job on the swapping disk. Fast access to these files is more important than one extra job on the swapping disk, however. The previous structure will be left intact and a second swap file added to handle the five additional jobs. SWAP1.SYS would be created on the system disk at size equal to 320 segments. Note that this arrangement will have little effect on system performance since inactive jobs are swapped on the slower system disk and active jobs are swapped on the faster device.

For an example of the third system, consider a very large system used to service several elementary schools. During the day, the system must service up to 48 student jobs. At non-peak hours the system is also used for large administrative programs. The installation requires a JOBMAX of 48 and SWPMAX of 16K due to the administrative processing performed.

The hardware configuration includes two RS03 fixed-head disks (1024 segments each) for swapping. Three swap files are required to handle the peak load of 48 jobs at 16K. SWAPØ.SYS is created on the first RS03 to handle 16 jobs (1024 segments). SWAPØ.SYS completely fills the first RS03 unit. SWAP1.SYS resides on the second RS03, but handles only 15 jobs (960 segments) to leave room for the other system files. Finally SWAP2.SYS is located on the system disk and handles the remaining 17 jobs (1088 segments).

Note that if the swapping device is two-platter RF11 disk rather than the two RS03 units, only one swap file is needed to cover both RF11 platters. As mentioned above, data transfers may cross platter boundaries on the RF11 but may not cross RS03 or RS04 unit boundaries. Hence, it was necessary to create two distinct swap files to use both RS03s. The BASE query described in Section 3.4.3.3 provides the mechanism by which the system files are located on specific RS03 or RS04 units.

If the hardware configuration does not include a swapping disk, a single swap file (SWAPØ.SYS) on the system disk is normally used for all jobs. Several swap files are used only if the required contiguous file space is not available to handle all jobs in one swap file.

The REFRESH option blindly creates the swap files ensuring only that they fit on the configured system disk and/or swapping disk(s). The DEFAULT and START options described in Sections 3.5 and 3.6 ensure proper correlation between JOBMAX, SWPMAX, and the sizes of the swap files created by REFRESH. The START option, furthermore, loads the monitor swap bit maps for use during normal timesharing.

3.4.2.5 OVR.SYS - Overlay Code

The non-resident (overlay) code in the RSTS/E system may be accessed directly from the Core Image Library file RSTS.CIL or it may be accessed as a separate file OVR.SYS. If the system disk is a fixed-head disk, there is no need to create a separate copy of the non-resident code. However, if the system disk is a moving-head device and an auxiliary swapping disk is available on the system, performance is increased by having the non-resident code as a separate file on the fast swapping disk.

REFRESH allows the OVR.SYS file to be created and positioned wherever desired. If a swapping disk is not included in the hardware configuration but the system files are being positioned near the center of a moving head disk (to reduce average seek time), OVR.SYS could be created and positioned with the other system files.

The size of the overlay code is the same for all systems. During system generation some of the overlay code may be made resident to increase system performance. The size of OVR.SYS is not reduced by making parts of the overlay code resident. If OVR.SYS is created, REFRESH allows its size to be greater than the required size. This provides a means to reserve contiguous file space on the system disk.

If the OVR.SYS file is created, the system will access the overlay code from this file. During REFRESH, the file contains no useful information and is not accessed in any way. The overlay code is transferred from the CIL to the OVR.SYS file when the system is brought up for normal time sharing with the START option.

3.4.2.6 ERR.SYS - Error Messages

Like the non-resident code, the RSTS/E system error messages may exist as a separate file, ERR.SYS, or they may be accessed directly from the RSTS.CIL file. If the system manager wishes to modify the error messages or have access to them on line, the ERR.SYS file must be created. Otherwise, the choice is based on fast access to the error messages and reducing the accesses to the system disk.

If the system disk is a fixed-head disk, there is normally no need to create the separate file except to allow modifications. If the system disk is a moving-head device and an auxiliary swapping disk is available on the system, the error messages can be retrieved faster if they exist as a separate file on the fast swapping device. The separate file on the swapping disk also reduces the burden on the system disk and helps in a small way to speed user file processing on the system disk. As in the case of the overlay code, REFRESH permits the ERR.SYS file to be created and positioned wherever desired. If a swapping disk is not included in the hardware configuration but the system files are being positioned near the center of a moving head disk, ERR.SYS could be created and positioned with the other system files.

The size of ERR.SYS is 2K words (8 blocks) for all systems. REFRESH allows the size of ERR.SYS to be greater than this required size to reserve contiguous file space. If the ERR.SYS file is created, the system reads the error messages from the file rather than the CIL. During REFRESH, the ERR.SYS file contains no useful information and is not accessed in any way. The error messages are transferred from the RSTS.CIL file to the ERR.SYS file when the system is brought up for normal timesharing with the START option.

3.4.2.7 BUFF.SYS - DECtape Buffers

DECtape processing in the RSTS/E system requires three 256-Word blocks of buffer storage for each drive unit in the system configuration. Thus, one TU56 Dual DECtape transport requires six blocks of buffer storage space. The file BUFF.SYS provides the required buffer space for DECtape operations. BUFF.SYS is required only on systems configured for DECtape.

The required size of BUFF.SYS is fixed by the number of configured DECtape units. REFRESH allows the size of BUFF.SYS to be increased to any number of blocks greater than the required size of three blocks per unit. This facility should be used if additional DECtape transports are added in the future. BUFF.SYS may be located on the swapping disk if one exists or may be positioned anywhere desired on a moving head system disk. On systems without DECtape, BUFF.SYS may be created to reserve contiguous file space on the system disk.

3.4.2.8 CRASH.SYS - Crash Dump File

The crash dump file CRASH.SYS is always optional but, if present, must reside on the system disk. The CRASH.SYS file must exist if the crash dump facility is to be enabled at the start of time sharing operations when using the START option or when establishing default start-up conditions using the DEFAULT option.

If the RSTS/E error handling code detects an error from which it cannot recover, the system is reloaded. In this controlled-crash sequence, the contents of the read/write area of monitor memory is dumped to the CRASH.SYS file. For example, if a memory parity error occurs in the area of memory occupied by the monitor or the BASIC-PLUS Run Time System, reloading the system is the safest recovery. This is done by the parity error handling code in the monitor. Since the dump preserves the state of the system at the time of the crash, it contains useful diagnostic information. Parity errors, for example, are logged by the error logging routines in the monitor. There is, however, a finite delay between the time an error is logged and the time this information can be picked up by the ERRCPY program. If the error causes a crash or if a crash occurs for any reason before ERRCPY can save the error information, this data (and probably the cause of the crash) will be lost if the dump was not performed. Hence, it is important that CRASH.SYS exist and the crash dump facility is enabled.

The size of CRASH.SYS depends on the size of the read/write portion of the monitor. The RSTS/E Monitor includes many tables and data areas which change during normal operation. This information resides in the lowest portion of memory (lowest physical addresses). The rest of the Monitor is system code (instructions) which is never changed under time sharing. Similarly, the BASIC-PLUS compiler and Run Time System is pure code which never changes. This "read only" code need not be dumped at the time of a crash. The size of the read/write area depends on the hardware and software configuration at each site. Hence, the size of CRASH.SYS is fixed at system generation time. The size is not the same on all systems but is normally 8 to 16K words (32 to 64 blocks).

REFRESH provides only the option to create or delete the CRASH.SYS file. If created, the size of CRASH.SYS will be the required size as discussed above. Furthermore, it will reside on the system disk wherever REFRESH can find sufficient contiguous file space.

3.4.3 Using REFRESH

The REFRESH option is invoked by typing REFRESH or simply RE in response to the initialization OPTION query. REFRESH responds by asking for the date and time followed by a series of questions. The complete REFRESH example on the following pages is included without comment to show the general flow of the REFRESH option. This example is repeated with detailed comments in Section 3.4.4.1. The sections below explain each of the REFRESH queries and the File Status Table.

```
OPTION: REFRESH
DD-MMM-YY? 22-JUL-74
HH:MM? 12:30

OLD ? NO

FILE                                          CURRENT   REQUIRED   START    START
NAME          REQUIRED? EXIST STATUS           SIZE      SIZE     CLUSTER  SECTOR

BADB   .SYS      YES     SYS     OK              0         0
RSTS   .CIL      YES     SYS     D/C           274       274         24       50
SATT   .SYS      YES     SYS     D/C            10        10          2        6
SWAP0  .SYS      YES      NO     CRE             0      1280
SWAP1  .SYS       NO      NO     OK              0         0
SWAP2  .SYS       NO      NO     OK              0         0
SWAP3  .SYS       NO      NO     OK              0         0
OVR    .SYS       NO      NO     OK              0        28
ERR    .SYS       NO      NO     OK              0         8
BUFF   .SYS      YES      NO     CRE             0        12
CRASH  .SYS       NO      NO     OK              0        37

BADS ? LIST

THERE ARE NO BAD BLOCKS

BADS ?  <LF>                                  <LF> denotes
                                              typing the LINE
RSTS.CIL CHANGES ? YES                        FEED key

SIZE ? 300

SATT.SYS CHANGES ? YES

BASE ? 40000

SWAP0.SYS CHANGES ? YES

SIZE ? 1024

DISK ? SWP

BASE ? <LF>

SWAP1.SYS CHANGES ? YES

SIZE ? 960
```

```
DISK ? SWP

BASE ? 81024

SWAP2.SYS CHANGES ? YES

SIZE ? 1088

DISK ? SYS

BASE ? 40000

SWAP3.SYS CHANGES ? NO

OVR.SYS CHANGES ? YES

CIL ? NO

SIZE ? <LF>

DISK ? SWP

BASE ? <LF>

ERR.SYS CHANGES ? YES

CIL ? NO

SIZE ? <LF>

DISK ? SWP

BASE ? <LF>

BUFF.SYS CHANGES ? YES

SIZE ? <LF>

DISK ? SWP

BASE ? <LF>

CRASH ? YES
```

<LF> denotes typing the LINE FEED key

FILE NAME	REQUIRED?	EXIST	STATUS	CURRENT SIZE	REQUIRED SIZE	START CLUSTER	START SECTOR
BADB .SYS	YES	SYS	OK	0	0		
RSTS .CIL	YES	SYS	OK	300	274	24	50
SATT .SYS	YES	SYS	OK	10	10	19999	40000
SWAP0 .SYS	YES	SWP	OK	1024	1280	39999	80000
SWAP1 .SYS	NO	SWP	OK	960	0	40511	81024
SWAP2 .SYS	NO	SYS	OK	1088	0	20004	40010
SWAP3 .SYS	NO	NO	OK	0	0		
OVR .SYS	NO	SWP	OK	28	28	40991	81984
ERR .SYS	NO	SWP	OK	8	8	41005	82012
BUFF .SYS	YES	SWP	OK	12	12	41009	82020
CRASH .SYS	NO	SYS	OK	38	37	174	350

```
OPTION:
```

3.4.3.1 Selecting the Short- or Long-Form of REFRESH

Immediately after the date and time are entered, REFRESH prints the OLD query. The response determines the type of REFRESH operation performed. When a new system disk is built, there are two basic alternatives. The system manager may let REFRESH do all the work and accept the default structure of the system files (short form). Alternatively, he may enter specifications for each system file (long form). One of these two options is selected by the response to the OLD query. Valid responses are listed in Table 3-4.

If the short form is selected, REFRESH creates the system files without further interaction. The answers to all file related queries are extracted from the existing file structure and configuration parameters. All required files are created at their required sizes and reside on the system disk. Optional files are not created at all. This procedure is not recommended since it seldom provides an optimal structure for the system files. The short form of REFRESH is primarily used on test systems where optimal structures are not required. The ability of REFRESH to extract answers to file related queries is helpful when REFRESH is used to alter the system files as installation requirements change. The long form of REFRESH is used but file related queries can be answered with the LINE FEED key to leave a current file specification unchanged.

Table 3-4

Valid Responses to OLD Query of REFRESH

Response	Meaning
YES or OLD	The short form of REFRESH. File characteristics are extracted from existing file structure and configuration parameters. Only nonexistent required files are created. Not recommended.
Y, O, or LINE FEED key	Same as YES but the first File Status Table is not printed.
NO	The long form of REFRESH. Individual file characteristics will be entered. REFRESH subsequently prints queries for each system file. Recommended.
N	Same as NO except the first File Status Table is not printed.
Anything Else	Repeats the OLD query.

The answer to the OLD query also determines whether the File Status Table is printed before the REFRESH operation is performed. The table is always printed before the option terminates to show the current structure of the system files. Normally, both tables are useful for reference. If several passes through REFRESH are needed due to some oversight, the table printed at the end of the previous pass is sufficient reference. Time can be saved by not printing the table on a subsequent try.

3.4.3.2 File Status Table

A File Status Table is printed twice during REFRESH. It is printed before any REFRESH action so that the system manager can see the current status of each system file and any REFRESH action required. The table is also printed at the end of the REFRESH operation to reflect any changes made. The status information given in the printout is summarized in Table 3-5.

The entry in the REQUIRED column indicates whether the file in that row of the table is necessary for operation of the RSTS/E system. The files BADB.SYS (bad block file) and SATT.SYS (storage allocation table) are required on all RSTS/E disks. The RSTS.CIL file is required on the system disk since it contains the permanent copy of the system code. One swap file (SWAPØ.SYS) is required while the others are optional. Separate copies of the overlay code and error messages may exist apart from the CIL as files OVR.SYS and ERR.SYS. These two files are not considered required since the information they contain may be accessed directly from the RSTS.CIL file. BUFF.SYS (DECtape buffers) is only required on systems configured for DECtape. Finally, the crash dump file (CRASH.SYS) is recommended for all systems but is not required.

The EXIST column merely indicates whether the file exists on the system disk (SYS), on the swapping disk or disks (SWP), or does not exist at all (NO). The entries in the STATUS column indicate to the system manager, the action which REFRESH must take independent of any changes he may specify. The STATUS column indicates that no REFRESH action is required (OK), the file must be created (CRE), or it must be deleted and then re-created (D/C). If any REFRESH action is required or requested on any of the system files, the other existing system files are also deleted and recreated. This is done to free up as much contiguous file space as possible before any attempt is made to restructure the system files.

Entries in the CURRENT SIZE and REQUIRED SIZE columns of the file status table display the decimal number of 256-word blocks occupied by and necessary for each system file. The values shown in the REQUIRED SIZE column depends on either the fixed system requirement for the particular file or the configuration of the system. For example, ERR.SYS is always 8 blocks (2K words) for all systems whereas the required size of the RSTS.CIL file depends on the hardware and

Table 3-5

System Files Status Table Entries

Status Table Entry Heading	Possible Entries	Meaning
FILE NAME	All system files in account [0,1]	See Section 3.4.2 for descriptions of system files in account [0,1].
REQUIRED?	YES	The file is currently necessary on the local installation system.
	NO	The file is not needed to operate the system.
EXIST	NO	The file does not currently exist.
	SYS	The file currently exists on the system disk.
	SWP	The file currently exists on the swapping disk.
STATUS	OK	No REFRESH action is required.
	CRE	File is marked for creation.
	DEL	File is marked for deletion.
	D/C	File is marked for deletion and re-creation.
CURRENT SIZE	n	The decimal number of 256-word blocks the file occupies on the disk.
REQUIRED SIZE	n	The decimal number of 256-word blocks the file minimally requires.
START CLUSTER	n	The decimal number of the first logical cluster occupied by the file.
START SECTOR	n	The decimal number of the physical sector occupied by the first 256-word block of the system file. Disk sectors are numbered from Ø to (disk size -1)

software configuration at each site. REFRESH normally does not permit the system manager to create any required file at a size smaller than the REQUIRED SIZE. The one exception is the SWAPØ.SYS file. The REQUIRED SIZE shown for SWAPØ.SYS handles the configured number of jobs at a maximum size (SWPMAX) of 8K. Since the actual number of jobs may be anything less than or equal to the configured maximum, the size of SWAPØ.SYS may be less than, equal to, or greater than the stated REQUIRED SIZE. The absolute minimum size allowed for SWAPØ.SYS is 32 blocks (1 job at 8K). For files such as OVR.SYS and ERR.SYS which are not explicitly required for correct system operation, the REQUIRED SIZE column indicates the number of blocks needed if the file is created.

The values in the CURRENT SIZE and REQUIRED SIZE columns may differ for several reasons. The most obvious is a system file which does not yet exist. The CURRENT SIZE will be zero while the REQUIRED SIZE will reflect the fixed system or configuration requirement. Secondly, the system manager may change the size of certain system files to meet installation needs. One such possibility with SWAPØ.SYS was mentioned above. The CURRENT SIZE of SWAPØ.SYS may be less than, equal to, or greater than the required size. Furthermore, if the system manager chooses to plan for future expansion, the CURRENT SIZE of system files such as RSTS.CIL and BUFF.SYS may be greater than the REQUIRED SIZE. Finally, system files occupy space to the nearest cluster boundary. Thus, if the REQUIRED SIZE is not an integral number of clusters at the system disk pack cluster size, REFRESH automatically rounds up to an integral number of clusters when the file is created. This round up is reflected in the CURRENT SIZE of the file. The sizes of the system files can be changed by response to the SIZE query printed by REFRESH. In all cases, REFRESH will prevent changes to the system file which are explicitly illegal.

The START CLUSTER and START SECTOR columns indicate, respectively, the logical and physical beginning address of the system file in question. The START CLUSTER column is a diagnostic aid to the designers. It has little practical value to the system manager. The START SECTOR column indicates the actual physical location of the file. Each of the system files occupies a contiguous area on the system disk or swapping disk(s). Disk sectors (256-word blocks) are numbered from Ø to the size of the disk minus one. An RK05 DECpack contains 4800 sectors or blocks numbered 0 to 4799. An RP03 pack contains 80000

sectors numbered Ø to 79999. RPØ4 sectors are numbered Ø to 167199. Since swapping disks are considered logical extensions of the system disk, swapping disk sectors are numbered starting at the end of the system disk (e.g., 48ØØ and up for an RKØ5 system disk, 8ØØØØ and up for an RPØ3 system disk and 167ØØØ and up for an RPØ4 system disk). Thus, by looking at the START SECTOR column, it is possible to know exactly where the system files reside. The BASE query of REFRESH permits locating certain system files at any desired start sector subject to availability of contiguous file space.

3.4.3.3 REFRESH Queries - All of the file related queries are listed for reference in Table 3-6. Additional comments are included below.

The <filename> CHANGES query allows an existing system file to be left unchanged by use of the NO, OLD, or LINE FEED response. When necessary to accommodate other system files, REFRESH repositions the system files even if no change is requested. The size of a file and the disk used are not changed, however. The situation is different if a required file does not yet exist. This situation arises when a new system disk is built. REFRESH creates a required file at the required size on the system disk if the CHANGES query is answered with the NO, OLD, or LINE FEED response.

Table 3-6
REFRESH File-Related Queries

File-Related Query	Applicable Files	Possible Responses	Meaning
BADS?	BADB.SYS	LIST	Print the start sector numbers of clusters which contain bad blocks.
		ADD	ADD option not yet implemented.
<filename> CHANGES?	RSTS.CIL SATT.SYS SWAPØ.SYS SWAP1.SYS SWAP2.SYS SWAP3.SYS OVR.SYS ERR.SYS BUFF.SYS	YES	Causes more queries to be printed each of which is directly related to the possible changes that can be made to the specific system file. (See individual file descriptions in Section 3.4.2.)
		NO, OLD or LINE FEED key	Causes the answers to the related filename changes queries to be extracted from the existing file structure or from system configuration parameters.

(continued on next page)

Table 3-6 (Cont.)
REFRESH File-Related Queries

File-Related Query	Applicable Files	Possible Responses	Meaning
SIZE?	RSTS.CIL SWAPØ.SYS SWAP1.SYS SWAP2.SYS SWAP3.SYS OVR.SYS ERR.SYS BUFF.SYS	OLD or LINE FEED key	If the file exists, retain the current size. If the file is being created, create at the required size.
		n	An integer decimal number which specifies the size in 256-word blocks. The value n must be greater than or equal to the required size except for SWAPØ.SYS which may be any size greater than 32 blocks.
DISK?	SWAPØ.SYS SWAP1.SYS SWAP2.SYS SWAP3.SYS OVR.SYS ERR.SYS BUFF.SYS	OLD or LINE FEED key	The file is written to the disk on which it previously resided. If file is newly created, place it on the system disk.
		SYS	The file is created on the system disk.
		SWP	The file is created on the swapping disk.
BASE?	SATT.SYS SWAPØ.SYS SWAP1.SYS SWAP2.SYS SWAP3.SYS OVR.SYS ERR.SYS BUFF.SYS	OLD or LINE FEED key	If the file existed on the disk, causes the file to be placed at the base physical sector number used previously. If the file is new, the base cluster value is the default of 0 (anywhere on the disk).
		n	An integer decimal number which specifies the base sector at which the file begins on the disk device. Sectors are numbered from 0 to the capacity of the disk (in 256-word blocks) minus 1. See Figure 3-1.
		0	If the number 0 is specified, the file is placed anywhere on the disk.

(continued on next page)

Table 3-6 (Cont.)
REFRESH File-Related Queries

File-Related Query	Applicable Files	Possible Responses	Meaning
CIL?	OVR.SYS ERR.SYS	YES	Indicates that the system accesses directly the relevant code in the CIL file (RSTS.CIL) on the system disk. Any existing copy of the file is deleted.
		NO	Indicates that the system manager desires a separate copy of the relevant code to be created. The file is created and the contents of the CIL code is copied to the file at start up time.
		OLD or LINE FEED key	Retain the file if it previously existed; otherwise, the code in the CIL is accessed directly.
DELETE?	BUFF.SYS SWAP1.SYS SWAP2.SYS SWAP3.SYS	YES	Causes file to be marked for deletion.
		NO or LINE FEED key	Causes more queries to be printed: SIZE?, DISK?, and BASE?.
CRASH?	CRASH.SYS	YES	Causes the file CRASH.SYS to be created.
		NO	Causes the file CRASH.SYS to be deleted.

The SIZE query is printed for several fixed-length system files. This provides a way to reserve contiguous file space on the system disk to allow for future expansion of the system files and provide for a swapping disk failure. For example, on a system with a single-platter RF11 (or RS03) swapping disk, a swap file normally resides on this swapping device. In the event of a swapping disk hardware failure, the system can continue to operate normally if this swap file is moved to the system disk. REFRESH moves the swap file if there is sufficient contiguous space available on the system disk. This space will probably not be available unless corrective action for such failures are considered when the system disk is initially built. By locating one of the system files such as BUFF.SYS (whether required or not) on the system disk and making its size equal

to 1024 blocks (size of the RF11) plus the required size of BUFF.SYS, the system can operate normally even if the swapping disk is down. REFRESH is used to move all files from the swapping disk to the system disk and BUFF.SYS is reduced to the required size. The files which previously resided on the swapping disk merely fill in the previously unused area of the BUFF.SYS file. The reverse procedure is performed when the swapping disk is again operational. The system files are moved back to the swapping disk and the size of BUFF.SYS is increased to 1024 blocks plus the required size. The RSTS.CIL file (which must reside on the system disk) can be enlarged for the same reasons. This has the added advantage of allowing a larger system (CIL) to replace the current CIL when a new system generation is performed.

The DISK query is printed only if a swapping disk(s) is included in the hardware configuration. If there is no swapping disk, all the system files must reside on the system disk.

The BASE query allows the system files to be positioned anywhere desired on the system or swapping disks. If a sector number is entered, it is considered a preferred location for the file rather than an absolute requirement. The disk allocation routines use this number as a place to start looking for contiguous file space. Any location between the specified BASE and the physical end of the disk is considered a legal start sector for the file. REFRESH works in two passes with regard to these preferred locations. If on the first pass, sufficient space is not available between the specified BASE and the end of the disk for any of the system files, a second try is automatically performed. Before the second try, all the preferred locations for all the system files are discarded. Thus, this second pass attempts to find sufficient space anywhere on the system disk or swapping disk(s). The particular disk used for a particular file is determined by the DISK query. REFRESH never moves a file from the swapping disk to the system disk (or vice-versa) unless told to do so by the appropriate response to the DISK query.

The DELETE query is printed only for optional files. For instance, if the BUFF.SYS file is created but is not required, the system manager has the option to delete the file using the REFRESH option.

3.4.3.4 REFRESH Action Message

After all file-related questions are answered, the REFRESH routines attempt to build the system file structure, as specified. If the REFRESH operation is successful, the File Status Table is reprinted indicating the new structure of the system files. The initialization routines then return to the OPTION query. The system manager should proceed to Section 3.5 and follow the procedures to establish the default start up conditions.

If the REFRESH routines are not able to make the requested changes, the following message is printed.

```
UNABLE TO CREATE REQUIRED FILES - REBUILDING DISK
```

There is a short pause while REFRESH reconstructs the original system files. The message

```
DISK REBUILT
```

appears after REFRESH recreates the original files and the File Status Table is printed. The table reflects the old file structure.

If the REFRESH operation is unsuccessful, the system manager must consider what caused the failure. The most likely reason for an unsuccessful REFRESH is an attempt to allocate more blocks to a file (or files) than there are blocks on the disk. (An RK05 DECpack has 4800 blocks; an RR03 has 80000 blocks; each RS03 unit and RF11 platter has 1024 blocks; and an RS04 has 2048 blocks). The system manager should repeat the REFRESH procedure.

3.4.4 REFRESH Examples

Two REFRESH examples are included in this section. The examples are intended to show some of the possible philosophies which can be used with REFRESH. In the first example, the system files are structured to allow for future expansion and even a hardware failure in the swapping disks. The second example is a small system where disk space is of primary importance. A reasonable structure of system files is created but no provision is made for future expansion.

3.4.4.1 RPØ3 System Disk with two RSØ3 Swapping Disks

The first example is a typical REFRESH for a large system. The hardware configuration includes an RP03 system disk and two RSØ3 fixed-head swapping disks. The only other relevant hardware included is four DECtape units. The system is configured for a maximum of 40 jobs and the desired swap maximum is 16K. The REFRESH proceeds as shown on the following pages.

After the date and time are entered, the OLD query is printed at line a. The NO response is used to request the long form of REFRESH. The system manager enters his requirements for each of the system files. The File Status Table is then printed indicating that only the BADB.SYS, RSTS.CIL, and SATT.SYS files currently exist. These three files are created by the DSKINT option. A request to LIST bad blocks at line b results in the message, THERE ARE NO BAD BLOCKS. If DSKINT had found any bad blocks on the RPØ3 pack, REFRESH prints the start sector numbers of the clusters containing bad blocks. When the BADS query is repeated at line c, the LINE FEED response is used to continue.

Changes are made to the RSTS.CIL file beginning at line d. The only operation permitted by REFRESH is to increase the size of the CIL file. The required size in this case is 274 blocks. The size is increased to 300 blocks to allow for future replacement of the CIL.

The only change which can be made to the SATT.SYS file (line e) is to position the file on the moving head RPØ3 disk. The Storage Allocation Table is referenced frequently under time sharing. It is therefore located near the center of the disk. The SWAP2.SYS file is also located near the center of the disk. The effect is to create a

```
OPTION: REFRESH
DD-MMM-YY? 22-JUL-74
HH:MM? 12:30

OLD ? NO                                                        (line a)

FILE                                          CURRENT  REQUIRED   START    START
NAME        REQUIRED? EXIST STATUS             SIZE     SIZE    CLUSTER  SECTOR

BADB  .SYS      YES    SYS     OK                 0        0
RSTS  .CIL      YES    SYS     D/C              274      274        24       50
SATT  .SYS      YES    SYS     D/C               10       10         2        6
SWAP0 .SYS      YES     NO     CRE                0     1280
SWAP1 .SYS       NO     NO     OK                 0        0
SWAP2 .SYS       NO     NO     OK                 0        0
SWAP3 .SYS       NO     NO     OK                 0        0
OVR   .SYS       NO     NO     OK                 0       28
ERR   .SYS       NO     NO     OK                 0        8
BUFF  .SYS      YES     NO     CRE                0       12
CRASH .SYS       NO     NO     OK                 0       37


BADS ? LIST                                                     (line b)

THERE ARE NO BAD BLOCKS                  <LF> denotes
                                         typing the LINE
BADS ?<LF>                               FEED key               (line c)

RSTS.CIL CHANGES ? YES                                          (line d)

SIZE ? 300

SATT.SYS CHANGES ? YES                                          (line e)

BASE ? 40000

SWAP0.SYS CHANGES ? YES                                         (line f)

SIZE ? 1024

DISK ? SWP

BASE ? <LF>

SWAP1.SYS CHANGES ? YES                                         (line g)

SIZE ? 960

DISK ? SWP

BASE ? 81024

SWAP2.SYS CHANGES ? YES                                         (line h)

SIZE ? 1088

DISK ? SYS

BASE ? 40000
```

```
SWAP3.SYS CHANGES ? <LF>                    <LF> denotes typing the LINE FEED key   (line i)

OVR.SYS CHANGES ? YES                                                              (Line j)

CIL ? NO

SIZE ? <LF>

DISK ? SWP

BASE ? <LF>

ERR.SYS CHANGES ? YES                                                              (line k)

CIL ? NO

SIZE ? <LF>

DISK ? SWP

BASE ? <LF>

BUFF.SYS CHANGES ? YES                                                             (line l)

SIZE ? <LF>

DISK ? SWP

BASE ? <LF>

CRASH ? YES                                                                        (line m)
```

FILE NAME	REQUIRED?	EXIST	STATUS	CURRENT SIZE	REQUIRED SIZE	START CLUSTER	START SECTOR
BADB .SYS	YES	SYS	OK	0	0		
NSTS .CIL	YES	SYS	OK	300	274	24	50
SATT .SYS	YES	SYS	OK	10	10	19999	40000
SWAP0 .SYS	YES	SWP	OK	1024	1280	39999	80000
SWAP1 .SYS	NO	SWP	OK	960	0	40511	81024
SWAP2 .SYS	NO	SYS	OK	1088	0	20004	40010
SWAP3 .SYS	NO	NO	OK	0	0		
OVR .SYS	NO	SWP	OK	28	28	40991	81984
ERR .SYS	NO	SWP	OK	8	8	41005	82012
BUFF .SYS	YES	SWP	OK	12	12	41009	82020
CRASH .SYS	NO	SYS	OK	38	37	174	350

```
OPTION:                                                                            (line n)
```

quiescent or "home" position for the heads near the center of the disk. The seek distance to access user file data on either side of center is thereby reduced. This will have more effect as user files and directories are scattered over the full disk surface.

The questions for SWAPØ.SYS begin at line f. Both RS03 drives are available for swap files and other system files. SWAPØ.SYS is located on the first RS03 unit. This choice is an arbitrary one in this case since both swapping disks are equally fast and are equal in size. The size of SWAPØ.SYS is set to 1024 blocks which is equal to the capacity of the drive. SWAPØ.SYS handles 16 jobs at 16K words each.

The second swap file, SWAP1.SYS (line g), is located on the second RS03. The response to the BASE query shown in the example forces REFRESH to try the allocation beginning at sector 81024 which is the first sector of the second RS03 (see Figure 3-1). It is not necessary to specify a BASE since all the space on the first RS03 has already been assigned. The second RS03 is used for SWAP1.SYS even if BASE were answered with the LINE FEED key. The size of SWAP1.SYS is set at 960 blocks to support 15 jobs at 16K. This leaves room on the swapping disk for other system files as discussed below.

Beginning at line h SWAP2.SYS is created on the system disk. SWAP2.SYS is purposely larger than necessary to allow for future expansion to 48 jobs. Expansion to 48 jobs requires a new SYSGEN but a new CIL can replace the current CIL without affecting the file structures on this system disk. With the size of SWAP2.SYS set to 1088 blocks, this file handles 17 jobs at 16K words. The size of SWAP2.SYS also provides for recovery from a total failure of the swapping disk subsystem (controller and/or both drives). In the event of a failure, SWAPØ.SYS moves to the system disk to replace SWAP2.SYS. SWAP1.SYS and SWAP2.SYS can be deleted if there is insufficient contiguous space to accommodate these files. Even if no other contiguous space is available, the space reserved in SWAP2.SYS is enough to handle 16 jobs at 16K with 64 blocks left over for the other system files. SWAP2.SYS is located near the center of the disk, physically adjacent to the SATT.SYS file.

The SWAP3.SYS CHANGES query is answered with the LINE FEED response (line i) since a fourth swap file is not needed. Since SWAP3.SYS does not exist, REFRESH does not create it.

The overlay code (OVR.SYS), error messages (ERR.SYS), and the DECtape buffers (BUFF.SYS) are all located on the second RS03. The CIL query for both OVR.SYS and ERR.SYS is answered NO to cause these files to be created. The SIZE queries for OVR.SYS, ERR.SYS, and BUFF.SYS are all answered with the LINE FEED key to allow REFRESH to create these files at their required sizes. There is no need to allocate extra space to these files when they reside on a swapping disk since unallocated space is never used for user files. The DISK queries are answered with the SWP response to locate all three files on the swapping disk. Finally, the BASE queries are also answered with LINE FEED to allow REFRESH to position these files wherever space is available on the swapping disk.

The CRASH.SYS file is created by the YES response to the CRASH query at line m. If the crash dump file is not created at this time, the system manager can not enable the crash dump facility using the DEFAULT or START option and valuable diagnostic information is lost in the event of a system crash.

The CRASH query ends the REFRESH interaction. There is a pause while REFRESH builds the system file structure as specified. The File Status Table is then reprinted to reflect the new structure. Note in the example that REFRESH positions all the files, as requested. The only other notable thing in the table is the expansion of the CRASH.SYS file to 38 blocks from the required size of 37 blocks. The pack cluster size of the RP03 system disk is the minimum pack cluster size of 2 blocks. The size of CRASH.SYS is rounded up to an integral number of clusters (38 blocks is 19 clusters with a pack cluster size of 2).

The printing of the OPTION query at line n signals the end of the REFRESH operation. The system manager then proceeds to the DEFAULT option (described in Section 3.5) to establish start up conditions for his system.

3.4.4.2 RK05 System Disk Without an Auxiliary Swapping Disk

The second example is a REFRESH for a smaller system which includes an RK system disk but no swapping disk. The system is configured for 16 jobs. Two DECtape drives are also included in the hardware configuration. The REFRESH operation is shown and explained below.

```
OPTION: REFRESH
DD-MMM-YY? 22-JUL-74
HH:MM? 18:30

OLD ? NO                                                          (line a)

FILE                                          CURRENT  REQUIRED   START    START
NAME        REQUIRED? EXIST STATUS              SIZE     SIZE    CLUSTER  SECTOR

BADB  .SYS     YES     SYS     OK                 0        0
RSTS  .CIL     YES     SYS     D/C              264      264        48       49
SATT  .SYS     YES     SYS     D/C                2        2         4        5
SWAP0 .SYS     YES      NO     CRE                0      512
SWAP1 .SYS      NO      NO     OK                 0        0
SWAP2 .SYS      NO      NO     OK                 0        0
SWAP3 .SYS      NO      NO     OK                 0        0
OVR   .SYS      NO      NO     OK                 0       28
ERR   .SYS      NO      NO     OK                 0        8
BUFF  .SYS     YES      NO     CRE                0        6
CRASH .SYS      NO      NO     OK                 0       28

BADS ? <LF>                                <LF> denotes          (line b)
                                           typing the LINE
RSTS.CIL CHANGES ? NO                      FEED key              (line c)

SATT.SYS CHANGES ? YES                                           (line d)

BASE ? 1600

SWAP0.SYS CHANGES ? YES                                          (line e)

SIZE ? 1024

BASE ? 1600

SWAP1.SYS CHANGES ? <LF>

SWAP2.SYS CHANGES ? <LF>

SWAP3.SYS CHANGES ? <LF>
```

```
OVR.SYS CHANGES ? YES                                          (line f)

CIL ? NO

SIZE ? <LF>

BASE ? 1600

ERR.SYS CHANGES ? YES                                          (line g)

CIL ? NO

SIZE ? <LF>

BASE ? 1600

BUFF.SYS CHANGES ? YES                                         (line h)

SIZE ? <LF>

BASE ? 1600

CRASH ? YES                                                    (line i)
```

<LF> denotes typing the LINE FEED key

FILE NAME	REQUIRED?	EXIST	STATUS	CURRENT SIZE	REQUIRED SIZE	START CLUSTER	START SECTOR
BADB .SYS	YES	SYS	OK	0	0		
RSTS .CIL	YES	SYS	OK	264	264	48	49
SATT .SYS	YES	SYS	OK	2	2	1599	1600
SWAP0 .SYS	YES	SYS	OK	1024	512	1601	1602
SWAP1 .SYS	NO	NO	OK	0	0		
SWAP2 .SYS	NO	NO	OK	0	0		
SWAP3 .SYS	NO	NO	OK	0	0		
OVR .SYS	NO	SYS	OK	28	28	2625	2626
ERR .SYS	NO	SYS	OK	8	8	2653	2654
BUFF .SYS	YES	SYS	OK	6	6	2661	2662
CRASH .SYS	NO	SYS	OK	28	28	12	13

```
OPTION:
```

This REFRESH is simpler than the previous example since there are fewer possibilities when no swapping disk exists. The REFRESH operation shown also has slightly different goals. On systems with RK disks, disk space for user files is usually in short supply. In this example, no disk space is reserved for future expansion. All of the system files (except SWAPØ.SYS) are created at their required sizes. Since the RK system disk is small in comparison to an RPØ3 (4800 blocks vs. 80000 blocks), it is not difficult to backup and delete all library and user files to gain space for future expansion. On systems with many RK drives, a completely different philosophy can be used. The system disk may be totally dedicated to system and library files. All unused space can be reserved by creating a dummy file under time sharing which completely fills all space not required for system or library files. The library can be easily saved and the dummy file deleted during time sharing to allow restructuring or expansion of the system files at a later date.

As shown in the example, the OLD query is answered NO at line a to request the long form of REFRESH. As in the previous example, the File Status Table indicates that only the BADB.SYS, RSTS.CIL, and SATT.SYS files exist before the initial use of REFRESH. The BADS query at line b is answered with the LINE FEED key since the status table indicates there are no bad blocks (size of BADB.SYS is zero). At line c, no change in the RSTS.CIL file is requested. The choice is to conserve disk space rather than plan for future expansion. Hence, the size of the CIL file is not increased to allow for replacement of the system at a later time.

The movable system files are all located near the center of the disk to reduce average seek time for the moving heads. At line d a change to SATT.SYS is requested and the file is positioned at sector 1600. The center of an RK cartridge disk is actually sector 2400. The system files span a range of sectors on each side of center.

Beginning at line e, a single swap file, SWAPØ.SYS is created to support 16 jobs at 16K and is positioned adjacent to the SATT.SYS file. Note that the DISK query is not printed in this case since the system is not configured for a swapping disk. The auxiliary swap files are not needed so the CHANGE queries for the other swap files are answered with the LINE FEED key.

Beginning at line f, the overlay code (OVR.SYS) and error message (ERR.SYS) files are created so that they can be positioned with the other system files. If disk space is extremely critical, this data can be accessed from the RSTS.CIL file. The DECtape buffer file is required and is created at the required size by the responses shown at line h. Finally, the CRASH.SYS file is created despite the space constraint since the crash dump facility cannot be enabled and valuable error logging information could be lost if the CRASH.SYS file did not exist.

The File Status Table is printed to reflect the new file structure and REFRESH terminates by returning to the OPTION query. The system manager proceeds to the DEFAULT option to establish start up conditions.

3.4A SETTING KEYBOARD DEFAULTS - SETKEY

The initialization option SETKEY sets default conditions for keyboard lines on the RSTS/E system. The option allows the user to enable and disable individual lines and to specify whether a DH11 multiplexor line is to be enabled with or without modem control.

The SETKEY option is required only if modem control must be enabled on a DH11-AA or DH11-AC multiplexor line (with DM11-BB modem control hardware) or on a DH11-AD line. SETKEY can optionally be used to disable any terminal interface or DH11 line which should not be used. At start up time, the initialization code enables all terminal interfaces (if the hardware is present) except those lines specifically disabled with SETKEY. DH11 lines wired for modem control are enabled as local lines unless modem control is specifically enabled with the SETKEY option. Modem control is always enabled on DL11E and DC11 single line interfaces unless the interface is disabled with SETKEY.

The default conditions established by SETKEY remain in effect until altered by SETKEY. Each time the system starts up, the initialization code enables terminals according to the SETKEY default values and the presence or absence of hardware.

To execute the SETKEY option, type SETKEY or SE in response to the OPTION: message printed by the initialization code. In response to SE, the query COMMAND is printed and one of the responses shown in Table 3-14 can be typed. The following dialog demonstrates the sequence.

```
OPTION: SETKEY
COMMAND?
```

To terminate SETKEY, type the EXIT command or the LINE FEED key and the OPTION: message is printed again.

Table 3-14
Responses to the SETKEY COMMAND Query

Response	Short Form	Meaning
DISABLE	DI	Disable a keyboard.
ENABLE	EN	Enable a keyboard on a single line interface or enable a keyboard as local on a DH11 multiplexer interface.
LIST	LI	Print a list of current default conditions for all keyboards configured on the system.
LOCAL	LO	Enable a keyboard as local on a DH11 multiplexer interface or enable a keyboard on a single line interface.
MODEM	MO	Enable a keyboard with modem control on a DH11-AA or DH11-AC multiplexer interface with DM11-BB modem hardware or on a DH11-AD multiplexer interface.
EXIT	EX	Exit to the OPTION: message.
Type the LINE FEED key		Exit to the OPTION: message.
Type the RETURN key		Print a list of legal commands and an accompanying description of action.

If either DISABLE, ENABLE, LOCAL, or MODEM is entered in response to the COMMAND query, the prompting message KB NUMBER is printed. Typing the unit number of the keyboard causes the related default condition to be set for the specified keyboard. The system continues printing the prompting message until the LINE FEED key is typed. The following sample dialog shows the procedure.

```
COMMAND?  DISABLE
KB NUMBER? 1
KB NUMBER? 3
KB NUMBER? <LF>          <LF> denotes typing the LINE FEED key

COMMAND?  EXIT

OPTION:
```

Typing DISABLE causes the system to print the prompting message. The disable condition is set when the keyboard unit number is entered. Typing the LINE FEED key returns to the COMMAND query. The EXIT command returns to the OPTION: message.

The LIST command causes the system to print a list containing keyboard unit designators, addresses, and current default settings. Each entry has the following general format.

```
                address
KBn:              or              IS     default setting
          address/address
```

The designation n gives the unit number of the keyboard. The address is either a single line interface address or the DH11 multiplexer address. (Addresses less than 160000 signify pseudo keyboards, which do not fall within the peripheral address space.) If two addresses are present and separated by a slant character, the first address is that of the DH11 multiplexer and the second is that of the DM11-BB modem control. The default setting can be either DISABLED, ENABLED, or ENABLED WITH MODEM.

The following sample dialog demonstrates the usage of the SETKEY commands.

```
OPTION:  SETKEY
COMMAND? DISABLE
KB NUMBER?
ENTER KEYBOARD NUMBER OR <LF> TO EXIT?  5
KB NUMBER? 13
KB NUMBER? 1
KB NUMBER?

COMMAND? MODEM
KB NUMBER? 20
KB NUMBER?

COMMAND?  LIST
KB0: (177560) IS ENABLED
KB1: (175610) IS DISABLED
KB2: (175620) IS ENABLED
KB3: (032424) IS ENABLED
KB4: (032430) IS ENABLED
KB5: (032434) IS DISABLED
KB6: (032440) IS ENABLED
KB7: (160020/170500) IS ENABLED
KB8: (160020/170500) IS ENABLED
KB9: (160020/170500) IS ENABLED
```

```
KB1Ø: (16ØØ2Ø/17Ø5ØØ) IS ENABLED
KB11: (16ØØ2Ø/17Ø5ØØ) IS ENABLED
KB12: (16ØØ2Ø/17Ø5ØØ) IS ENABLED
KB13: (16ØØ2Ø/17Ø5ØØ) IS DISABLED
KB14: (16ØØ2Ø/17Ø5ØØ) IS ENABLED
KB15: (16ØØ2Ø/17Ø5ØØ) IS ENABLED
KB16: (16ØØ2Ø/17Ø5ØØ) IS ENABLED
KB17: (16ØØ2Ø/17Ø5ØØ) IS ENABLED
KB18: (16ØØ2Ø/17Ø5ØØ) IS ENABLED
KB19: (16ØØ2Ø/17Ø5ØØ) IS ENABLED
KB20: (16ØØ2Ø/17Ø5ØØ) IS ENABLED WITH MODEM
KB21: (16ØØ2Ø/17Ø5ØØ) IS ENABLED
KB22: (16ØØ2Ø/17Ø5ØØ) IS ENABLED

COMMAND?
```

In the example above, the DISABLE command disabled keyboards 1, 5, and 13. The MODEM command enabled modem control on one DH11 multiplexor line (keyboard 2Ø). The LIST command prints a listing of conditions for all keyboard units on the system. The listing shows the following conditions. Keyboard units Ø through 3 are single line interfaces with discrete addresses in the I/O address space. Keyboard unit 1 is shown disabled as a result of the DISABLE command. Keyboard units 3 through 6 are pseudo keyboard lines, unit 5 of which is disabled as a result of the DISABLE command. Keyboard units 7 through 22 are DH11 multiplexor lines. The two addresses shown with each line are the UNIBUS addresses of the DH11 multiplexor and the DM11-BB modem, respectively. The listing shows that all DH11 lines are enabled and that keyboard unit 2Ø is enabled with modem control as a result of the MODEM command.

3.5 DEFAULT OPTION

The DEFAULT initialization option is used to establish certain default start up conditions for the RSTS/E system. These conditions include:

a) The maximum number of jobs the system allows during time sharing operations.

b) The maximum storage space that a job is allowed to occupy in the swapping files (and, therefore, the size to which a job is allowed to expand during time sharing operations).

c) The allocation of the BASIC-PLUS Compiler and Run Time System to an area of memory.

d) The assignment of certain portions of memory as usable or not usable, either by the system or user jobs.

e) The enabling or disabling of the crash dump facility.

The system manager is required to establish the initial default start-up conditions. The DEFAULT option is used immediately after the REFRESH operation is completed. During the first use of DEFAULT, the system manager may accept the system-assigned defaults or change them to meet the installation needs. The DEFAULT option may also be used at any later time to change the start up conditions as installation requirements change.

This section describes the start up conditions and the use of the DEFAULT option in detail. The options available with DEFAULT are also available when the START option is used to begin time sharing. It is possible to set job and swap maximums, to alter the memory allocation, and to enable or disable the crash dump facility, using either the DEFAULT or START options. There is an important distinction, however; the start up conditions established with DEFAULT are permanent. Parameters are saved in the RSTS.CIL file on the system disk so that they can be used each time the system is brought up. The DEFAULT start up conditions may be overridden for one time sharing session by making changes with the START option. Thus, changes to the start up conditions made with the START option are temporary. The DEFAULT parameters are used if the system is restarted. The permanent DEFAULT start up conditions can only be changed by a subsequent use of the DEFAULT option.

3.5.1 Using the DEFAULT Option

The system manager requests the DEFAULT option by typing DEFAULT (or DE) in response to the initialization code OPTION query. The DEFAULT routines print a series of three queries to determine which start up conditions are to be changed. A brief description of the DEFAULT queries is provided in Table 3-7. Valid responses and additional related options are described in subsequent sections.

Table 3-7
DEFAULT and START Option Queries

Query	Meaning	References
JOB OR SWAP MAX CHANGES?	Allows system manager to change (or establish) either the number of jobs allowed to run during time sharing (JOB MAX) or the length of the job swap area in the swapping files (SWAP MAX).	See Section 3.5.2 for the operational description and Table 3-8 for the possible responses.
ANY MEMORY ALLOCATION CHANGES?	Allows system manager to locate the Run Time System in an area of memory and to prohibit or allow use of certain portions of memory by the system.	See Section 3.5.3 for the operational description and Table 3-10 for the possible responses.
CRASH DUMP?	Allows system manager to either enable or disable the crash dump facility.	See Section 3.5.4 for the operational description and Table 3-13 for the possible responses.

The following sample dialogue illustrates the use of the DEFAULT option. In this particular example, old DEFAULT start up conditions are being printed for reference. When DEFAULT is used for the first time on a new system, the message "NO DEFAULTS ARE CURRENTLY SET" precedes the dialogue and the Memory Allocation Table is printed automatically.

```
OPTION: DEFAULT

YOU CURRENTLY HAVE: JOB MAX = 32, SWAP MAX = 16K.

JOB MAX OR SWAP MAX CHANGES ? OLD


ANY MEMORY ALLOCATION CHANGES ? YES

TABLE OPTION ? LIST

 ADDR   +00000  +04000  +10000  +14000  +20000  +24000  +30000  +34000
000000    MON     MON     MON     MON     MON     MON     MON     MON
040000    MON     MON     MON     MON     MON     MON     MON     MON
100000    MON     MON     MON     MON   BASIC   BASIC   BASIC   BASIC
140000  BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC
200000  BASIC   BASIC       U       U       U       U       U       U
240000      U       U       U       U       U       U       U       U
300000      U       U       U       U       U       U       U       U
340000      U       U       U       U       U       U       U       U
400000      U       U       U       U       U       U       U       U
440000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
500000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
540000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
600000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
640000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
700000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
740000    NXM     NXM     NXM     NXM     I/O     I/O     I/O     I/O

TABLE OPTION ? EXIT

YOU CURRENTLY HAVE: CRASH DUMP ENABLED.

CRASH DUMP ? OLD


RSTS V05B-24 TEST SYSTEM

OPTION:
```

The DEFAULT code is the most critical of the RSTS/E initialization routines. Prior to using this option, the KT11 memory management unit, physical memory above 28K, and the memory parity registers (if any) have not been used. DEFAULT enables memory mapping and scans the 124K of physical address space to determine the size and location of all available memory. DEFAULT also enables memory parity checking to determine the association between parity registers and the parity memory banks. If any of this hardware is not operating properly, it is very likely that DEFAULT will fail. If a crash occurs during execution of the DEFAULT option, the components mentioned should be carefully checked with diagnostics.

3.5.2 JOB and SWAP Maximums

The DEFAULT and START options allow the system manager to establish limits on the number of jobs which can be run during time sharing and the maximum size to which each job can expand. The JOB MAX and SWAP MAX established with the DEFAULT option are in effect each time the system is brought up unless specifically changed at start up time. If JOB MAX or SWAP MAX are changed using the START option, the change is in effect only for that one time sharing session. Normally JOB MAX and SWAP MAX are set with the DEFAULT option and are left unchanged when the system is brought up.

With either option, the system prints the current values of JOB MAX and SWAP MAX and then asks if any changes are desired. The sequence appears as shown below.

```
OPTION: DEFAULT
YOU CURRENTLY HAVE: JOB MAX = 32, SWAP MAX = 16K.
JOB MAX OR SWAP MAX CHANGES ?
```

When the DEFAULT option is used for the first time, the value of JOB MAX is the maximum number of jobs configured at system generation time. The value of SWAP MAX is 8K. These system defaults will probably be changed since few installations are satisfied with the 8K SWAP MAX. The system manager can request a change by the appropriate response to the JOB MAX or SWAP MAX CHANGES query. The possible responses are shown in Table 3-8.

Table 3-8
Possible Responses to JOB OR SWAP MAX CHANGES Query

Response(s)	Meaning
NO N OLD O LINE FEED key	Causes the currently established default values to remain unchanged and the second in the series of three queries, ANY MEMORY ALLOCATION CHANGES?, to be printed.
RETURN key	Causes an explanatory form of the query to be printed, after which the system manager types his response.
CONTROL key and C combination	Causes interrupt of execution and printing of the OPTION query.
YES	Indicates system manager wants to change either JOB MAX or SWAP MAX and causes additional related queries to be presented.

If a change is requested, additional queries ask the system manager to specify a new value for JOB MAX and/or SWAP MAX. These two variables are dependent on each other and on the sizes of the swap files created by the REFRESH option. Recall from the discussion of the swap files (see Section 3.4.2.4) that each file holds an integral number of jobs defined by

$$\text{JOBS for Swap File n} = \frac{\text{Size of SWAPn.SYS in segments}}{\text{SWPMAX}*4} \text{ (truncated)}$$

The sum of the jobs held by each of the existing swap files must be greater than or equal to the desired JOB MAX specified. The desired JOB MAX and SWAP MAX are entered in response to the queries shown in Table 3-9.

The initialization routines accept any value for JOB MAX between 1 and the configured maximum number of jobs. Any value of SWAP MAX between 6(K) and 28(K) is accepted. The BASIC-PLUS Run Time System does not allow any job to exceed 16K, however. The large range of values accepted for SWAP MAX is intended to provide for future additions to the RSTS/E system. After a new JOB MAX and/or SWAP MAX is entered, the DEFAULT routines print the new values and check them against the sizes of the swap files. If the existing swap files do

Table 3-9
Possible Responses to NEW JOB MAX and NEW SWAP MAX Queries

Query	Possible Responses	Meaning
NEW JOB MAX?	1 to maximum number of jobs configured at system generation time	Set the maximum number of jobs which can be run during normal time sharing operations.
NEW SWAP MAX?	6 to 28	The amount of storage space in K (1024) words to be allocated for each job in the swap file. Determines the maximum size to which each job is allowed to expand. (See text below.)
EITHER OF ABOVE QUERIES	RETURN key	Causes an explanatory message to be printed after which the query is reprinted.
	LINE FEED key, NO, or OLD	Do not change current maximum.

not accommodate the SWAP MAX and JOB MAX specified, an error message is printed and the system manager must enter new values. If the new JOB MAX and SWAP MAX are consistent with the sizes of the existing swap files, the JOB MAX OR SWAP MAX CHANGES query is reprinted. The system manager normally requests no further changes and proceeds to set up the memory allocation table described in Section 3.5.3.

The example below demonstrates the procedure for changing JOB MAX and SWAP MAX. The example assumes two swap files: SWAPØ.SYS is 1024 segments long; SWAP1.SYS has 960 segments.

```
OPTION: DEFAULT

YOU CURRENTLY HAVE: JOB MAX = 32, SWAP MAX = 8K.

JOB MAX OR SWAP MAX CHANGES ? YES

NEW JOB  MAX ?

NEW SWAP MAX ? 16

YOU CURRENTLY HAVE: JOB MAX = 32, SWAP MAX = 16K.

YOUR CURRENT JOB AND/OR SWAP MAXIMUMS ARE INCONSISTENT
WITH THE SIZE(S) OF THE EXISTING SWAP FILE(S).  YOU MUST,
THEREFORE, SPECIFY A NEW JOB MAX OR SWAP MAX.

NEW JOB  MAX ? 31

NEW SWAP MAX ?

YOU CURRENTLY HAVE: JOB MAX = 31, SWAP MAX = 16K.

JOB MAX OR SWAP MAX CHANGES ? NO

ANY MEMORY ALLOCATION CHANGES ?
```

Note in the example that the attempt to set JOB MAX to 32 and SWAP MAX to 16K failed because the swap files are not large enough. SWAPØ.SYS (1024 segments) handles 16 jobs at 16K. SWAP1.SYS (960 segments) holds only 15 jobs at 16K. The second attempt which reduced JOB MAX to 31 while leaving SWAP MAX at 16K was successful. Note also that with the same swap files, other JOB MAX and SWAP MAX combinations can be accommodated. For example, 33 jobs can be run with a SWAP MAX of 15K.

If there is any problems arriving at the desired JOB MAX and SWAP MAX, it may be necessary to abort the DEFAULT (or START) option by typing CONTROL/C and to enter REFRESH to check the sizes of the existing swap files. After REFRESH prints the File Status Table (see Section 3.4.3.1), the system manager can determine from the sizes of the swap files what values of JOB MAX and SWAP MAX are allowed. If no changes to the swap files are desired, REFRESH can be aborted with CONTROL/C after the File Status Table is printed. The system manager can then reenter DEFAULT (or START) and enter new values for JOB MAX and/or SWAP MAX. Alternatively, a complete REFRESH can be performed to increase the size of the swap files so that the desired JOB MAX and SWAP MAX are accepted.

3.5.3 Memory Allocation

The DEFAULT option allows the system manager to allocate a specific area of memory to the BASIC-PLUS Run Time System, to make certain portions of memory either unavailable or available for use, and to list the current usage and types of memory on the system.

When DEFAULT is used to establish the start up conditions for a new system, a memory allocation table is printed as shown in the example of Section 3.5.1. The system manager may accept the system default memory allocation or make changes by appropriate responses to the TABLE OPTION query. The memory allocation table is not printed automatically after the initial use of the DEFAULT option. Instead, the system prints the query, ANY MEMORY ALLOCATION CHANGES? The system manager may leave the defaults as they stand or request a change in the memory allocation by one of the responses shown in Table 3-10.

Table 3-10
Possible Responses to ANY MEMORY ALLOCATION CHANGES Query

Response(s)	Meaning
NO OLD O or LINE FEED key	Causes currently established memory allocation assignments to remain unchanged and next query, CRASH DUMP?, to be printed.
RETURN key	Causes an explanatory form of the query to be printed, after which the user types his response.
YES	Causes printing of the query TABLE OPTION? See Table 3-11 for additional related queries.
CONTROL key and C combination	Causes interrupt of execution and the printing of the OPTION query.

If the system manager indicates that he wishes to alter the current memory allocation, the TABLE OPTION query is printed. The possible responses to the TABLE OPTION query are summarized in Table 3-11. Subsequent sections describe each of the table options in more detail.

Table 3-11

Additional Related Queries and Responses to
ANY MEMORY ALLOCATION CHANGES Query

Additional Related Query	Response(s)	Result
TABLE OPTION?	RETURN key	Causes a list of the options to be printed after which the TABLE OPTION query is reprinted and user is allowed to type his response.
	LIST	Causes the current memory allocation table to be printed.
	PARITY	Used as a diagnostic tool to identify various types of parity memory on the system.
	LOCATE	Indicates the user desires to load Run Time System (BASIC-PLUS Monitor) beginning at a certain available portion of memory. Causes the query NEW RUN-TIME SYSTEM ADDRESS IS? to be printed. (See description of this query below.)
	LCK	Indicates user desires to prevent the system from using some portion of memory and prints LOCKOUT ADDRESS IS? (See query described below.)
	UNLOCK	Indicates the system manager desires to allow a currently locked portion of memory to be available for a user job and causes an additional query (UNLOCK ADDRESS IS?) to be printed. (See additional query described below.)
	RESET	Allows the initialization code to set up the memory allocation table. Unlocks all locked memory, locates RTS immediately after the monitor, and makes available for use any new memory added to the system.
	EXIT	Causes exit from routine which changes memory allocation and prints next DEFAULT option query, CRASH DUMP?

(continued on next page)

Table 3-11 (Cont.)

Additional Related Queries and Responses to ANY MEMORY ALLOCATION CHANGES Query

Additional Related Query	Response(s)	Result
NEW RUN-TIME SYSTEM ADDRESS IS?	RETURN key	Causes an explanatory message to be printed and the query to be reprinted, after which the user types the six digit octal number, xxxxxx.
	xxxxxx	The six-digit octal number representing the 18-bit address of the portion of memory at which the Run-Time System is loaded. (Loading is done from low memory to high memory.) The octal number can be gained by referring to the printout of the current memory allocation table.
LOCKOUT ADDRESS IS?	RETURN key	Causes explanatory message to be printed after which query ADDRESS OF 1K MEMORY SECTION IS? is printed.
	xxxxxx or	The 6-digit octal number which corresponds to the 18-bit address of the 1K portion of memory to be locked out of use. The octal number can be determined by referring to the printout of the current memory allocation table.
	xxxxxx-xxxxxx	Two numbers, separated by a dash, specifies a range of successive 1K portions of memory to be locked.
UNLOCK ADDRESS IS?	RETURN key	Causes explanatory message to be printed, after which ADDRESS OF 1K MEMORY SECTION IS? query is printed, allowing user to enter the six digit octal number, xxxxxx.
	xxxxxx or	The six digit octal number which corresponds to the 18-bit address of the 1K portion of memory to be unlocked for use. The octal number can be gained by referring to the printout of the current memory allocation table.
	xxxxxx-xxxxxx	Two numbers, separated by a dash, specifies a range of successive 1K portions of memory to be unlocked.

3.5.3.1 LIST Table Option

The LIST table option prints the current memory allocation table on the console terminal. The table provides useful information on the size of the RSTS/E monitor, the size of the BASIC-PLUS Run Time System (compiler and interpreter), the amount of memory available for user jobs, which portions of memory (if any) are disabled, and the total size of the memory on the machine. The table covers the full 128K of physical addresses showing the allocation or state of each 1K word section with appropriate symbols. The example below demonstrates the use of the LIST table option.

```
TABLE OPTION ? LIST

 ADDR   +00000  +04000  +10000  +14000  +20000  +24000  +30000  +34000
000000    MON     MON     MON     MON     MON     MON     MON     MON
040000    MON     MON     MON     MON     MON     MON     MON     MON
100000    MON     MON     MON     MON     MON     MON     U       U
140000    U       U       U       U       U       U       U       U
200000    U       U       U       U       U       U       U       U
240000    LCK     LCK     U       U       U       U       U       U
300000    U       U       U       U       U       U       U       U
340000    U       U       U       U       U       U       U       U
400000    U       U       U       U       U       U       U       U
440000    U       U       U       U       U       U       U       U
500000  BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC
540000  BASIC   BASIC   BASIC   BASIC   BASIC   BASIC     U       U
600000    U       U       U       U       U       U       U       U
640000    U       U       U       U       U       U       U       U
700000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
740000    NXM     NXM     NXM     NXM     I/O     I/O     I/O     I/O

TABLE OPTION ?
```

Each row of the memory allocation table represents 8K of the 128K physical address space. Each octal number in the first column of the table is a base address to which an octal number from the top row must be added to generate the full 18-bit (6 octal digit) start address for a 1K section. For example, the address of the eleventh 1K portion of memory is found by adding +10000 to 040000, giving 050000 as the 18-bit address. The symbols used in the memory allocation table are listed and described in Table 3-12.

Table 3-12
Memory Allocation Table Symbols

Symbol	Meaning
MON	Occupied by RSTS/E Monitor.
BASIC	Occupied by BASIC-PLUS Run Time System
U	Available for user space
LCK	Memory has been made unavailable for use (locked) by the system manager.
NXM	Memory does not exist (nonexistent) on the PDP-11.
I/O	Peripheral Addresses.

In the example above, the total memory on the machine is 112K words. The Monitor (MON) is large (22K) and resides at the lowest physical addresses. The BASIC-PLUS Run Time System (BASIC) begins at address 500000 (80K) and is 14K words long. Presumably high-speed MOS memory begins at address 500000. This would be the reason for locating the Run Time System at that address (see LOCATE table option in Section 3.5.3.3). There is no memory (NXM) for physical addresses 700000 to 760000. The I/O page always begins at address 760000. The rest of memory in the example is user (U) memory. However, the 2K section beginning at address 240000 has been locked out (LCK) and is not available for use (see LOCK table option in Section 3.5.3.4). Memory is locked if it is found to be defective but has not been removed from the system.

The LIST table option can be used as often as desired in both the DEFAULT and START options. If changes are made to the memory allocation table using the LOCATE, LCK (LOCK), UNLOCK, or RESET table options described in the following sections, a copy of the final DEFAULT memory allocation table should be kept with other system documentation for future reference.

THIS PAGE IS PURPOSELY LEFT BLANK

3.5.3.2 PARITY Table Option

The PARITY table option is used primarily as a diagnostic tool to identify and locate the various types of parity memory on the system. The option will be useful to DEC Field Service personnel to verify the memory configuration when the system is installed. It is also useful to identify the physical address of MOS memory. The types of memory (MOS or core) are determined by looking at the hardware parity registers. It is impossible for the software to determine the type for non-parity memories.

The listing of the parity configuration is printed if PARITY is entered as the response to the TABLE OPTION query. The example below shows all the possibilities.

```
TABLE OPTION ? PARITY

 ADDR   +00000  +04000  +10000  +14000  +20000  +24000  +30000  +34000
000000  00(C)   00(C)   00(C)   00(C)   00(C)   00(C)   00(C)   00(C)
040000  00(C)   00(C)   00(C)   00(C)   00(C)   00(C)   00(C)   00(C)
100000  14(C)   14(C)   14(C)   14(C)   14(C)   14(C)   14(C)   14(C)
140000  14(C)   14(C)   14(C)   14(C)   14(C)   14(C)   14(C)   14(C)
200000  00/14   00/14   00/14   00/14   00/14   00/14   00/14   00/14
240000  00/14   00/14   00/14   00/14   00/14   00/14   00/14   00/14
300000  02(C)   02(C)   02(C)   02(C)   02(C)   02(C)   02(C)   02(C)
340000  02(C)   02(C)   02(C)   02(C)   02(C)   02(C)   02(C)   02(C)
400000  20(M)   20(M)   20(M)   20(M)   20(M)   20(M)   20(M)   20(M)
440000  20(M)   20(M)   20(M)   20(M)   20(M)   20(M)   20(M)   20(M)
500000  24(M)   24(M)   24(M)   24(M)   24(M)   24(M)   24(M)   24(M)
540000  24(M)   24(M)   24(M)   24(M)   24(M)   24(M)   24(M)   24(M)
600000  02(C)   02(C)   02(C)   02(C)   02(C)   02(C)   02(C)   02(C)
640000    NO      NO      NO      NO      NO      NO      NO      NO
700000   NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
740000   NXM     NXM     NXM     NXM     I/O     I/O     I/O     I/O
```

Each 1K segment of the 128K physical address space is listed with one of the codes shown below:

- NO - Non parity memory
- NN(C) - Parity Core (non-interleaved)
- NN(M) - Parity MOS
- NN/MM - Parity Core (interleaved)
- NXM - Nonexistent memory
- I/O - I/O Page (peripheral addresses)

NN and MM are the last 2 octal digits of the address of the parity register responsible for that 1K section of memory. Up to 16 parity registers are in the UNIBUS address range 7721ØØ to 772136. When a parity error is detected, the parity register responsible for that section of memory contains information on the location of the last error detected. One parity register controls up to 24K (MM11-LP) or 32K (MM11-UP) of parity core memory. If core memory is interleaved, two parity registers are used for up to 48K (MM11-LP) or 64K (MM11-UP). For MOS memory one parity register handles one 16K bank. MOS memory cannot be interleaved.

Interleaving means that one bank of memory responds to "even" memory addresses while another bank responds to "odd" addresses as shown below:

ADDRESS		PARITY REGISTER
000000	EVEN	NN
000002	ODD	MM
000004	EVEN	NN
000006	ODD	MM
000010	EVEN	NN
000012	ODD	MM

Interleaving is used because core memories are destructive read out devices. After each read from a core memory, the original data must be restored. However, once the processor receives the data requested, it can go on to other things (presumably another memory reference) during the restore cycle in the memory bank just referenced. The bank remains busy until the restore cycle completes. When memory is interleaved, the probability is low that the same bank will be referenced on the next memory cycle. Hence, interleaving allows some overlap of memory operations with a resultant speed up in program execution.

All of the possibilities are included in the example above. The list below identifies the types of memory and the associated parity register shown in the example.

START ADDRESS OF 8K SECTION	MEMORY TYPE	PARITY REGISTER ADDRESS
000000	Non Interleaved Core	772100(00)
040000	Non Interleaved Core	772100(00)
100000	Non Interleaved Core	772114(14)
140000	Non Interleaved Core	772114(14)
200000	Interleaved Core	772100(00)EVEN; 772114(14)ODD
240000	Interleaved Core	772100(00)EVEN; 772114(14)ODD
300000	Non Interleaved Core	772102(02)
340000	Non Interleaved Core	772102(02)
400000	MOS	772120(20)
440000	MOS	772120(20)
500000	MOS	772124(24)
540000	MOS	772124(24)
600000	Non Interleaved Core	772102(02)
640000	Non Parity Memory	No parity register
700000	Non Existent Memory (NXM)	
740000	NXM and I/O Page	

3.5.3.3 LOCATE Table Option

The LOCATE table option allows the system manager to position the BASIC-PLUS Run Time System (compiler and interpreter) anywhere in memory. For example, if the hardware configuration includes 16K words of MOS memory, it is advantageous to locate the Run Time System in that memory. The bulk of processing done by the RSTS/E system involves executing BASIC-PLUS programs. Timing studies indicate that 70% to 80% of the CPU time is spent executing code in the Run Time System. Hence, optimal performance is realized if this code resides in the fastest memory available on the machine. If the machine has a mixture of fast and slow core memory, there may be some small advantage in locating the RTS in fast core. On most systems without semiconductor memory, the only reason to locate BASIC would be to avoid a defective section of memory. The Run Time System would normally follow immediately after the Monitor in the lowest physical memory addresses.

When the LOCATE table option is requested, the system prints one additional query, NEW RUN-TIME SYSTEM ADDRESS IS?. The system manager enters the 6-digit octal address of a 1K section of memory that is to become the lowest address of the Run Time System. The initialization code checks the address to ensure that sufficient contiguous memory is available to load the RTS starting at the specified address. If any of the memory in the required address range is locked out or is allocated to the Monitor or I/O page, an error message is printed and the TABLE OPTION query is reprinted. The system manager is free to try again. If the new Run Time System address is accepted, the TABLE OPTION query is reprinted and another option can be entered. The LIST table option can be used to verify the location of the Run Time System. The procedure is shown in the example below.

```
ANY MEMORY ALLOCATION CHANGES ? YES

TABLE OPTION ? LIST

 ADDR   +00000  +04000  +10000  +14000  +20000  +24000  +30000  +34000
000000    MON     MON     MON     MON     MON     MON     MON     MON
040000    MON     MON     MON     MON     MON     MON     MON     MON
100000    MON     MON     MON     MON   BASIC   BASIC   BASIC   BASIC
140000  BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC
200000  BASIC   BASIC       U       U       U       U       U       U
240000      U       U       U       U       U       U       U       U
300000      U       U       U       U       U       U       U       U
340000      U       U       U       U       U       U       U       U
400000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
440000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
500000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
540000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
600000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
640000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
700000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
740000    NXM     NXM     NXM     NXM     I/O     I/O     I/O     I/O

TABLE OPTION ? LOCATE
NEW RUN-TIME SYSTEM ADDRESS IS? 310000

TABLE OPTION ? LIST

 ADDR   +00000  +04000  +10000  +14000  +20000  +24000  +30000  +34000
000000    MON     MON     MON     MON     MON     MON     MON     MON
040000    MON     MON     MON     MON     MON     MON     MON     MON
100000    MON     MON     MON     MON       U       U       U       U
140000      U       U       U       U       U       U       U       U
200000      U       U       U       U       U       U       U       U
240000      U       U       U       U       U       U       U       U
300000      U       U   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC
340000  BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC
400000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
440000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
500000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
540000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
600000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
640000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
700000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
740000    NXM     NXM     NXM     NXM     I/O     I/O     I/O     I/O

TABLE OPTION ?
```

There are cases where the Run Time System should not be located in high-speed memory. Consider the configuration below where the memory allocated to the Monitor is shown but, for the purpose of this illustration, the location of the Run Time System is not shown.

ADDR	+00000	+04000	+10000	+14000	+20000	+24000	+30000	+34000
000000	MON	MON	MON	MON	MON	MON	MON	MON
040000	MON	MON	MON	MON	MON	MON	MON	MON
100000	MON	MON	MON	MON	U	U	U	U
140000	U	U	U	U	U	U	U	U
200000	MOS	MOS	MOS	MOS	MOS	MOS	MOS	MOS
240000	MOS	MOS	MOS	MOS	MOS	MOS	MOS	MOS
300000	U	U	U	U	U	U	U	U
340000	NXM	NXM	NXM	NXM	NXM	NXM	NXM	NXM
. .	.	.	.	.	.	.	.	.
. .	.	.	.	.	.	.	.	.
. .	.	.	.	.	.	.	.	.
740000	NXM	NXM	NXM	NXM	I/O	I/O	I/O	I/O

Assume that the size of the Run Time System is 14K words. If the RTS is located starting at address 200000, the remaining user memory is broken up into one 12K section below the RTS and one 10K section above. User job images are loaded into contiguous physical memory. Hence, no program larger than 12K could be run with this memory configuration. In this case it would be better to locate the Run Time System at the high end of available memory starting at address 250000. This would leave 22K of contiguous memory for user jobs between the Monitor and RTS. This configuration allows 16K programs to be run at the expense of not using the high speed MOS memory to full advantage. Only 6K of MOS is used for the RTS while the other 10K of MOS is part of the total 22K of user memory. Similar situations will arise and each memory configuration must be considered individually.

3.5.3.4 LOCK Table Option

The LOCK table option is used to lock out certain portions of memory. Locking memory prevents the RSTS/E system from using that memory during time sharing operations. This capability is valuable, for example, if a certain section of memory is found to be defective and disrupting RSTS/E system operation. Providing there is sufficient user memory remaining after the malfunctioning sections are locked, the system may be brought up for normal time sharing.

The system responds to LCK by printing a related query LOCKOUT ADDRESS? The system manager can then type a 6-digit octal number to specify the 18-bit address of the 1K section of memory to be locked. If two numbers separated by a dash are entered, a range of successive 1K sections is locked. The 1K section corresponding to the first address and successive 1K sections between and including the second address are locked. After either a valid address or a range of addresses is entered, the TABLE OPTION query is reprinted and another option may be entered. The LIST option can be used to verify that the desired sections were locked out. The example below illustrates the procedure.

Certain restrictions apply to locking out memory. Memory in use by the Monitor cannot be locked. There is no provision for relocating the Monitor to another section of memory. Similarly, memory currently in use by the Run Time System cannot be locked until the RTS is relocated to a usable area of memory. If an attempt is made to lock out a portion of memory already in use, an error message is printed. Finally, to remove a full bank of memory from use, the four, eight, or sixteen contiguous 1K sections which comprise the hardware bank must all be locked. Certain types of memory failures affect only one word. Others will affect larger sections or even the full hardware bank. Any memory failure must be carefully analyzed to determine which 1K section must be locked. The resulting memory configuration must also be considered to ensure that sufficient contiguous user memory is available to run the installation's programs. (See related discussion in Section 3.5.3.3.)

```
ANY MEMORY ALLOCATION CHANGES ? YES

TABLE OPTION ? LIST

 ADDR   +00000  +04000  +10000  +14000  +20000  +24000  +30000  +34000
000000    MON     MON     MON     MON     MON     MON     MON     MON
040000    MON     MON     MON     MON     MON     MON     MON     MON
100000    MON     MON     MON     MON     U       U       U       U
140000    U       U       U       U       U       U       U       U
200000    U       U       U       U       U       U       U       U
240000    U       U       U       U       U       U       U       U
300000    U       U       U       U       U       U       U       U
340000    U       U       U       U       U       U       U       U
400000    U       U       U       U       U       U       U       U
440000    U       U       U       U       U       U       U       U
500000    U       U     BASIC   BASIC   BASIC   BASIC   BASIC   BASIC
540000  BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC
600000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
640000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
700000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
740000    NXM     NXM     NXM     NXM     I/O     I/O     I/O     I/O

TABLE OPTION ? LCK
LOCKOUT ADDRESS IS? 240000-254000

TABLE OPTION ? LIST

 ADDR   +00000  +04000  +10000  +14000  +20000  +24000  +30000  +34000
000000    MON     MON     MON     MON     MON     MON     MON     MON
040000    MON     MON     MON     MON     MON     MON     MON     MON
100000    MON     MON     MON     MON     U       U       U       U
140000    U       U       U       U       U       U       U       U
200000    U       U       U       U       U       U       U       U
240000    LCK     LCK     LCK     LCK     U       U       U       U
300000    U       U       U       U       U       U       U       U
340000    U       U       U       U       U       U       U       U
400000    U       U       U       U       U       U       U       U
440000    U       U       U       U       U       U       U       U
500000    U       U     BASIC   BASIC   BASIC   BASIC   BASIC   BASIC
540000  BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC
600000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
640000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
700000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
740000    NXM     NXM     NXM     NXM     I/O     I/O     I/O     I/O

TABLE OPTION ?
```

RSTS/E can continue running with certain types of memory failures. On systems with parity memory, parity errors are logged by the error handling routines in the monitor. The recovery procedures depend on the usage of the offending section of memory at the time of the error. Three cases need to be considered.

The first case is a parity error in a section of memory allocated to the Monitor or Run Time System. Since continued system operation would be risky, the error is logged, a crash dump is taken (if the crash dump facility is enabled), and the system is reloaded. When the system comes back up, the ERRCRS program is used to extract the error log information from the CRASH.SYS file and ERRDIS is used to print the location of the error (see Section 6.11.3). ERRDIS provides sufficient information to identify a 1K (Core) or 4K (MOS) section which should be locked out. If the error occurred in Run Time System memory, the RTS can be relocated and the section locked. If the error was in Monitor memory, the hardware must be repaired. In either case, a hard failure may prevent the system from running at all and, hence, neither ERRCRS nor ERRDIS can be run. The only recourse in such cases is to run memory diagnostics to locate the problem. If the failure was transient the system manager has the option of taking the system down to lock out memory or to continue time sharing. At some point memory diagnostics should be run to reproduce the failure, if possible.

The situation is slightly different if a parity error occurs in user memory. The parity error handling routines will determine if a single user is affected (resides in the malfunctioning section) or more than one user is affected. If more than one user is affected the system is reloaded as described above. If a single user is affected, that user is aborted with the error message MEMORY PARITY FAILURE - PROGRAM LOST SORRY. The error is logged, the section is automatically locked out from future use, and the system continues running. The bad section of memory should be permanently locked out (until repaired) using the LCK table option the next time the system is brought up.

The third case involves multiple parity errors which occur in rapid succession. RSTS/E will halt at location 56 if a second parity error occurs while the first parity error is being processed. Memory diagnostics must be used to locate the malfunctioning memory.

On systems without parity memory, there is no way for the software to detect or locate a memory failure. Programs may get wrong results, memory management errors may occur, or any number of random problems may be observed. At worst the system crashes with perhaps misleading clues to the cause.

3.5.3.5 TABLE OPTION: UNLOCK

The UNLOCK table option is used to free a previously locked portion of memory for use by the Run Time System or by user jobs. Typing UNLOCK in response to the TABLE OPTION query causes the system to print the related query UNLOCK ADDRESS IS. The system manager can then enter one address, or two addresses separated by a dash, to unlock either a single 1K section or a range of successive 1K sections of memory. The procedure is the same as that described for LOCK. If the memory specified is not locked, an error message is printed followed by the TABLE OPTION query.

3.5.3.6 TABLE OPTION: RESET

The RESET option allows the initialization code to set up the memory allocation table. All memory which was locked is unlocked. The Run Time System is moved from wherever it was located to follow immediately after the Monitor in low physical memory. Any new memory which has been added to the system is made available for use. In fact, the RESET table option must be used whenever additional new memory is added. The initialization code recognizes and does not attempt to use memory which has been removed. However, the initialization code does not alter the memory allocation table to include previously non-existent memory unless told do so.

RESET can be used with either the DEFAULT or START option. It usually does not make sense to use RESET at start-up time, however. The types of memory allocation changes described above are most often permanent changes. RESET should only be used, therefore, with the DEFAULT option. The effect of RESET is demonstrated in the example below.

```
ANY MEMORY ALLOCATION CHANGES ? YES

TABLE OPTION ? LIST

 ADDR   +00000  +04000  +10000  +14000  +20000  +24000  +30000  +34000
000000    MON     MON     MON     MON     MON     MON     MON     MON
040000    MON     MON     MON     MON     MON     MON     MON     MON
100000    MON     MON     MON     MON     U       U       U       U
140000    U       U       U       U       U       U       U       U
200000    U       U       U       U       U       U       U       U
240000    LCK     LCK     LCK     LCK     U       U       U       U
300000    U       U       U       U       U       U       U       U
340000    U       U       U       U       U       U       U       U
400000    U       U     BASIC   BASIC   BASIC   BASIC   BASIC   BASIC
440000  BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC
500000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
540000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
600000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
640000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
700000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
740000    NXM     NXM     NXM     NXM     I/O     I/O     I/O     I/O

TABLE OPTION ? RESET

TABLE OPTION ? LIST

 ADDR   +00000  +04000  +10000  +14000  +20000  +24000  +30000  +34000
000000    MON     MON     MON     MON     MON     MON     MON     MON
040000    MON     MON     MON     MON     MON     MON     MON     MON
100000    MON     MON     MON     MON   BASIC   BASIC   BASIC   BASIC
140000  BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC   BASIC
200000  BASIC   BASIC     U       U       U       U       U       U
240000    U       U       U       U       U       U       U       U
300000    U       U       U       U       U       U       U       U
340000    U       U       U       U       U       U       U       U
400000    U       U       U       U       U       U       U       U
440000    U       U       U       U       U       U       U       U
500000    U       U       U       U       U       U       U       U
540000    U       U       U       U       U       U       U       U
600000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
640000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
700000    NXM     NXM     NXM     NXM     NXM     NXM     NXM     NXM
740000    NXM     NXM     NXM     NXM     I/O     I/O     I/O     I/O

TABLE OPTION ?

READY
```

3.5.3.7 TABLE OPTION: EXIT

When all desired changes to the memory allocation table have been made, the EXIT command causes the DEFAULT (or START) routines to continue to the next query, CRASH DUMP? The crash dump facility can be enabled or disabled as described in Section 3.5.4.

3.5.4 CRASH DUMP Facility

After the memory allocation has been determined, the DEFAULT and START routines print the current status of the crash dump facility (see Section 6.9 for further discussion). A subsequent query allows the system manager to enable or disable crash dumps. The CRASH.SYS file created by the REFRESH option (see Section 3.4.2.8) must exist if crash dumps are to be enabled.

When the DEFAULT option is used for the first time, crash dumps are disabled even if CRASH.SYS exists. Whenever crash dumps are disabled but the CRASH.SYS file exists, the following message is printed, followed by the CRASH DUMP query.

```
YOU CURRENTLY HAVE CRASH DUMP DISABLED.

CRASH DUMP?
```

If the crash dump facility is enabled and CRASH.SYS exists, the message reflects the current condition.

```
YOU CURRENTLY HAVE CRASH DUMP ENABLED.

CRASH DUMP?
```

If the CRASH.SYS file does not exist, the following message is printed regardless of the current state of the crash dump facility.

```
BUT, SINCE THE CRASH DUMP FILE DOESN'T EXIST, CRASH
DUMP MUST BE DISABLED.
```

The initialization routines automatically disable the crash dump facility if the CRASH.SYS file does not exist. If the system manager wishes to enable the crash dump facility, he must use the REFRESH option to create the CRASH.SYS file. Once the file is created, the crash dump facility can be enabled permanently, using the DEFAULT option, or temporarily, using the START option. If the crash dump facility is disabled automatically, the DEFAULT routine saves the specified default parameters in the RSTS.CIL file on the system disk and returns to the OPTION query.

When the CRASH DUMP query is printed by either the DEFAULT or START option, the responses shown in Table 3-13 are valid.

Table 3-13

Possible Responses to CRASH DUMP Query

<table>
<tr><th>Response(s)</th><th colspan="3">Meaning</th></tr>
<tr><td>LINE FEED key
OLD
O</td><td colspan="3">The default response. If typed, the system checks for the existence of the CRASH.SYS file and establishes the result shown below based on the current condition of the crash dump facility and the existence of CRASH.SYS.

<table><tr><th>Current Condition</th><th>CRASH.SYS Exists?</th><th>Result</th></tr><tr><td>Enabled</td><td>Yes</td><td>Remains enabled</td></tr><tr><td>Enabled</td><td>No</td><td>Automatically disabled</td></tr><tr><td>Disabled</td><td>N/A</td><td>Remains disabled</td></tr></table></td></tr>
<tr><td>YES
Y</td><td colspan="3">Enables the crash dump facility.</td></tr>
<tr><td>NO
N</td><td colspan="3">Disables the crash dump facility.</td></tr>
<tr><td>RETURN key</td><td colspan="3">Causes an explanatory form of the query to be printed, after which the user types his response.</td></tr>
<tr><td>CONTROL key
and C combination</td><td colspan="3">Returns user to OPTION query.</td></tr>
</table>

After enabling or disabling the crash dump facility, the DEFAULT routines save the default parameters and return to the OPTION query. START routines continue by asking for the current date and time prior to loading the system for normal timesharing.

3.6 START OPTION - STARTING TIME SHARING OPERATIONS

The START initialization option is used to bring the RSTS/E system up for normal time sharing. The START option is similar to the DEFAULT option in that job and swap maximums, the current memory allocation, and the condition of the crash dump facility may all be set during the START sequence. As emphasized in the previous section, any changes made to the start-up parameters override the DEFAULT parameters for only one time sharing session. The DEFAULT start-up conditions always apply unless specifically changed each time the START option is used.

The START option is requested by typing START, or ST, or simply the LINE FEED key in response to the initialization code OPTION query. The following sample dialogue shows the normal procedure for bringing up a new RSTS/E system for the first time.

```
OPTION: START

YOU CURRENTLY HAVE: JOB MAX = 31, SWAP MAX = 16K.

JOB MAX OR SWAP MAX CHANGES ?

ANY MEMORY ALLOCATION CHANGES ?

CRASH DUMP ?

DD-MMM-YY? 22-JUL-74
HH:MM? 12:18


CAN'T FIND FILE OR ACCOUNT
PROGRAM LOST - SORRY

READY
```

The first three queries are the same as those printed when the DEFAULT option is used. Job and swap maximums are described in Section 3.5.2. Memory allocation options are discussed in Section 3.5.3. The enabling or disabling of the crash dump facility is the subject of Section 3.5.4. Normally the DEFAULT start-up conditions are not changed. In the example, these queries are answered with the LINE FEED key which implies no change to the related DEFAULT start-up condition.

START continues by asking for the date and time which must be entered in the exact format shown. After date and time are accepted, there is a pause while the START code is executed. During these several seconds the start-up routines enable all configured terminal interfaces, dump the overlay code and error messages from the RSTS.CIL file to the OVR.SYS and ERR.SYS files (if they exist), set up monitor tables, turn on memory management, load the Monitor and Run Time System, and start the system clock. Finally the system attempts to execute the INIT system program (described in Section 6.1). When a new system is brought up for the first time, INIT does not yet exist in the system library. As a result, the error messages shown in the example are printed. The system initialization routines terminate as signaled by the printing of the READY message.

When the READY message is printed, job 1 is logged into the system at the console terminal under account [1,2] and is at the BASIC-PLUS command level. The situation is the same as if the system manager had logged into the system by use of the HELLO command described in Section 2.1.2 of the RSTS-11 System User's Guide. At this point, the system manager begins to build the files in the system library on the system disk.

The system library consists of two types of files: compiled system program files and formatted ASCII text and message files. The system manager follows the procedures described in Section 4.2 to build the compiled files. After the compiled files are created, the system manager must follow the procedures described in Section 4.3 to create the ASCII text and message files.

3.6.1 Start Up Errors and Messages

The START sequence is a critical part of the system initialization. START references all configured terminal interfaces and disables any that do not respond. The KT11 memory management unit is activated to determine the size of available memory (as in DEFAULT), to load the monitor and Run Time System, and to prepare for normal time sharing. START, furthermore, loads the stack limit register, enables parity traps for all parity memory, and activates the system clock to begin time sharing. Terminal interfaces, other than that for the console terminal, and the stack limit register have not been used up to this time. Even the clock was not critical to the operation of the System Generation Monitor or the other initialization routines. If the system does not come up as shown in the preceding example, these components are suspect and should be exercised with the standard diagnostics.

The message "SWAP MAX OF XX BEING LOWERED TO YY" is printed if the available contiguous memory is not large enough to support the requested value of SWAP MAX (see Section 3.5.2). This condition can occur if the BASIC-PLUS Run Time System is located such that it subdivides the contiguous user memory. To correct this situation the DEFAULT option can be used to relocate the BASIC-PLUS Run Time System (see LOCATE and LIST memory allocation options in Sections 3.5.3.3 and 3.5.3.1). A similar situation can occur on systems with a minimum amount of memory but many peripherals or software options. The amount of user memory left over may not be sufficient to support the desired SWAP MAX. The only way to correct this situation (other than adding memory) is to perform a new system generation requesting few jobs and fewer options. In either case, the RSTS/E system operates normally with the lower SWAP MAX but there may be limitations on which system programs can be run.

The START code attempts to enable all configured terminal interfaces, DH11 terminal multiplexors, and DM11BB modem control multiplexors. If any of these configured devices are not found, a message is printed indicating that the keyboards associated with these interfaces are being disabled. The hardware or software may not be configured properly or certain interfaces may be missing or inoperative. DEC Field Service should check the hardware configuration if unexplained messages of this type are printed. In most cases the

system will operate normally after the affected keyboards are disabled. However, it is impossible for the software to run with certain types of hardware configuration errors. These messages may precede a crash or signal hardware problems.

Finally, if the processor halts during the START sequence, the possibility exists that the wrong type of system clock was configured.

START also performs certain software checks to verify that the required system files exist and the DEFAULT start-up conditions have been established. Any attempt to start time sharing before the DEFAULT option is run results in the following message.

```
OPTION: START
DEFAULTS MUST BE SET PRIOR TO START UP
OPTION:
```

The system manager must execute the DEFAULT option (see Section 3.5) before he can begin timesharing.

Similarly, if the required system files do not exist, the START sequence aborts with the message below.

```
OPTION: START
CANNOT START WITH THIS SYSTEM DISK - PLEASE REFRESH
OPTION:
```

The REFRESH option must also be used to create the system files before timesharing can begin.

3.7 UNISYS OPTION

The UNISYS option provides a way to bring up the RSTS/E system for time sharing without enabling any terminal interfaces except the console (KBØ) interface. UNISYS is used as a diagnostic tool in cases where the system will not come up. If an otherwise nonworking system does work with UNISYS, the hardware or the software terminal configuration is probably faulty. Responses given in the SYSGEN questions should be checked against the hardware configuration. Communications devices and terminal interfaces should be checked by DEC Field Service for proper UNIBUS addresses and interrupt vectors.

The UNISYS option merely sets a flag which is checked at start-up time. The only way to clear the flag is to reboot the RSTS/E system disk to reload the initialization code (see BOOT option in Section 3.8). UNISYS is therefore used immediately before the START option. When the flag is set, START bypasses the code which enables terminal interfaces. The option is used as shown below.

```
OPTION: UNISYS

                                    (Return to OPTION
                                     is immediate)

OPTION: START

(normal START messages)
```

3.8 BOOT OPTION - BOOTSTRAPPING A DEVICE

The RSTS/E initialization code includes bootstrap routines for common disk and tape devices. The BOOT option is provided as a convenient means to simulate the hardware ROM bootstraps. BOOT is used to load another operating system into memory from its system disk and to reload an altered copy of the RSTS/E initialization code into memory after an INIT patch has been installed.

The BOOT option is invoked by typing BOOT or BO in response to the OPTION query. BOOT replies by asking for the device to be bootstrapped. The system verifies that the device exists before attempting the bootstrap operation. The following example demonstrates the use of the option.

```
OPTION: BOOT
BOOT DEVICE:
BOOTABLE DEVICES ARE:
        DT      DECTAPE DRIVE 0
        MT      MAGTAPE DRIVE 0
        DC      RC11 DISK
        DF      RF11 DISK
        DK      RK11 DISK DRIVE 0
        DP      RP11 DISK DRIVE 0
PLEASE SELECT ONE OF THESE DEVICES: DT
```

The program or secondary bootstrap for the monitor to be loaded must reside on unit Ø of the device requested. The first 64 words from block Ø on the device are read and loaded into the first 64 words of memory. Control then transfers to address Ø. Any monitor or program which can be loaded using the MR11-DB or BM792-YB hardware bootstrap can also be loaded using the BOOT option. The new monitor overlays the RSTS/E initialization code.

The RSTS/E system disk can be dismounted if the drive is needed for the system to be loaded. The initialization code continues running when the system disk is removed from the drive. Hence, if a different operating system is to be loaded, its system disk can replace the RSTS/E system disk on RK or RP unit Ø.

The default device (LINE FEED response to the BOOT DEVICE query) is the device on which the RSTS/E system disk is (or was) mounted. This is useful when the initialization code is patched using the PATCH option (see the example in Section 3.2.2). The PATCH option modifies the RSTS.CIL file. To load the altered INIT code into memory, the LINE FEED response is used to reboot the RSTS/E system disk.

3.9 LOAD OPTION - LOADING STAND-ALONE PROGRAMS

The LOAD option provides a mechanism for loading and executing stand-alone programs from the RSTS.CIL file. The DEC supplied program which currently can be included in the CIL is ROLLIN. The option is invoked by typing LOAD or LO in response to the OPTION query. The option then asks for the name of the program to load. A table of loadable programs will be printed if the RETURN key is typed in response to the LOAD PROGRAM query as shown in the example below.

```
OPTION: LOAD
LOAD PROGRAM:
  TABLE OF LOADABLE PROGRAMS:
 NAME   IDENT   LOAD    SIZE    TRANSFER
ROLLIN  V07     001000  020672  001332
PLEASE ENTER NAME OF PROGRAM TO LOAD: ROLLIN


ROLLIN V07

#
```

If the program to be loaded has an odd transfer address[1], the following message is printed.

```
THE TRANSFER ADDRESS OF THE SPECIFIED PROGRAM IS ODD.
PLEASE ENTER A NEW (EVEN) TRANSFER ADDRESS OR TYPE
CONTROL/C TO ABORT PROGRAM LOAD, NEW TRANSFER ADDRESS?
```

[1] Programs assembled or linked without specifying a transfer address are given a default transfer address of 000001. This is an odd address.

The PDP-11 computer traps to an error vector if an odd (byte) address is specified. Therefore, the user types an even (word) octal address at which the program is to be started or types the CONTROL key and C key combination simultaneously (CTRL/C) to abort the LOAD operation.

If the new transfer address typed is odd, the following message is printed and INIT returns to the OPTION query.

```
ILLEGAL NEW TRANSFER ADDRESS SPECIFIED
OPTION:
```

If the transfer address entered is even, the system searches for the specified program, loads it into memory, and starts execution. When execution of the stand-alone program has completed, RSTS/E can be restored by following the applicable bootstrap proce- Section 5.2.3).

CHAPTER 4

SYSTEM GENERATION: BUILDING SYSTEM LIBRARY FILES AND CREATING USER ACCOUNTS

The system generation procedures continue with the system manager building the system library files and creating user accounts. The DSKINT initialization option creates the system library account [1,2] on the system disk. The BUILD program is used under time sharing to create the system library files. Tailored text and message files are created with PIP or EDIT and user accounts are created with REACT.

When time sharing operations start, the monitor sets up job 1 under account [1,2] and attempts to run the INIT system program. Since no programs initially exist in account [1,2], the monitor generates the CAN'T FIND FILE OR ACCOUNT error and prints the READY message. The console terminal (KBØ:) is at BASIC-PLUS command level and job 1 is active as if a user had logged into the system.

With the console terminal at command level, a single command is typed to run BUILD from the medium on which the RSTS/E system library files are stored. Once started, the BUILD program runs detached, reads commands from a control file, and executes those commands at the console terminal. Upon completion, the BUILD program reattaches job 1 to the console keyboard.

The RSTS/E software contains the three control files listed and described below.

BUILD.CTL	Builds a general system library usable on all systems.
SPLER.CTL	Builds files executable only on a system with a SWAP MAX value larger than 8K words. *

RJ278Ø.CTL	Builds a file executable only on systems having the RSTS/278Ø optional software.

The BUILD program must be run once for each set of files required. Section 4.1 describes the files created by each control file and Section 4.2 describes the procedures to run BUILD from the three types of media. The system programs built by these procedures are documented in Chapter 4 of the RSTS-11 System User's Guide and in Chapter 6 of this guide. The system manager can remove from the RSTS-11 System User's Guide descriptions of programs he does not wish to be run on the system. The RJ278Ø program is described in the RSTS/E RJ278Ø User's Guide. A complete list of the documents and order numbers is in the Preface of this document.

After the appropriate files are built, several ASCII text and message files must be created. The procedures to do this are given in Section 4.3. Finally, the user accounts must be created on the system disk. The procedures for this activity are in Section 4.4.

4.1 CONTROL FILES FOR THE BUILD PROGRAM

This section describes the programs and files created by BUILD when it executes commands in each control file.

4.1.1 BUILD.CTL File

Commands in the BUILD.CTL file create a general system library for all systems. The programs and files are listed and described in Table 4-1. Procedures to change the sample ASCII text and messages are given in Section 4.3. References provided in Table 4-1 are section numbers in either the RSTS-11 System User's Guide (SUG) or this manual (SMG).

Table 4-1
BUILD.CTL Programs and Files

Program or File Name	Protection (if other than <124>)	Description	Section Reference[1]
LOGIN.BAC		Logs users into system	SUG - 2.1, 4.1
LOGOUT.BAC		Logs user off system	SUG - 2.1, 4.2
BUILD.BAC		Builds system library files	SMG - Ch 4
UTILTY.BAC		System utility program	SMG - 6.3
INIT.BAC		Time sharing initialization program	SMG - 6.1
SYSCAT.BAC		Creates a directory listing of accounts on a file structured device.	SMG - 6.15
PRIOR.BAC		Changes priority, run burst, and SWAP MAX value.	SMG - 6.14
TALK.BAC	<232>	Terminal communications	SMG - 6.16
ANALYS.BAC		Crash analysis	SMG - 6.9
SYSTAT.BAC	<232>	Status reports	SMG - 6.6 SUG - 4.3
UMOUNT.BAC	<232>	Private disk mounter and dismounter	SUG - 4.14
QUOLST.BAC	<232>	Lists disk quota and usage data for current user	SUG - 4.6
ERRCPY.BAC		Copies hardware error data to a disk file	SMG - 6.11
ERRDIS.BAC		Formats data saved by ERRCPY.BAC	SMG - 6.11
ERRDI1.BAC		Chained to by ERRDIS.BAC	SMG - 6.11
SHUTUP.BAC		Performs standard system shut down	SMG - 6.2
PIP.BAC	<1Ø4>	Peripheral interchange program to transfer files	SUG - 4.4

(Continued on next page)

[1]The abbreviation SUG refers to the RSTS-11 System User's Guide and SMG refers to the RSTS/E System Manager's Guide.

Table 4-1 (Cont.)
BUILD.CTL Programs and Files

Program or File Name	Protection (if other than <124>)	Description	Section Reference[1]
FILCOM.BAC	<1Ø4>	Compares ASCII files	SUG - 4.15
MONEY.BAC	<1Ø4>	System accounting program	SUG - 4.7 SMG - 6.5
GRIPE.BAC	<232>	User comments program	SUG - 4.8 SMG - 6.8
REACT.BAC		Account creator	SMG - 6.4
TTYSET.BAC	<232>	Terminal characteristics setter	SUG - 4.5 SMG - 6.7
PLEASE.BAC	<232>	Console terminal request printer	SMG - 6.16
INUSE.BAC	<232>	Terminal in use warning	SUG - 4.16
RESEQ.BAC	<1Ø4>	BASIC-PLUS line resequencer	RESEQ.DOC
CREF.BAC	<1Ø4>	BASIC-PLUS cross reference program	CREF.DOC
START.CTL	<6Ø>	Sample system start up control file	SMG - 6.1
CRASH.CTL	<6Ø>	Sample system crash recovery control file	SMG - 6.1
ACCT.SYS	<6Ø>	Sample system accounts file	SMG - 6.4.3
TTYSET.SPD	<6Ø>	Sample terminal speed control file	SMG - 6.7.1
PIP.TXT	<4Ø>	PIP help message text file	SUG - 4.4.2 SMG - 5.1.2
ERRDIS.HLP	<6Ø>	Error display program help file	SMG - 6.11.3
HELP.TXT	<4Ø>	Sample system help message text file	SUG - 2.1.4 SMG - 5.1.2

(Continued on next page)

[1]The abbreviation SUG refers to the RSTS-11 System User's Guide and SMG refers to the RSTS/E System Manager's Guide.

Table 4-1 (Cont.)
BUILD.CTL Programs and Files

Program or File Name	Protection (if other than <124>)	Description	Section Reference[1]
NOTICE.TXT	<4Ø>	Sample system notices text file	SUG - 2.1.2, 4.1.1 SMG - 5.1.2
RESEQ.DOC	<4Ø>	RESEQ program document file	SMG - 4.1
CREF.DOC	<4Ø>	CREF program document file	SMG - 4.1
EDIT.BAC	<1Ø4>	DOS/BATCH-11 compatible text editor	SUG - 4.9
EDITCH.BAC	<1Ø4>	Chained to by EDIT.BAC	SUG - 4.9
BACKUP.BAC	<232>	System back up (fail safe)	SUG - 4.1Ø
BACKDK.BAC	<232>	Chained to by BACKUP.BAC	SUG - 4.1Ø
ODT.BAC		Octal debugging tool	SMG - 6.12
DSKINT.BAC		Initializes disks	SMG - 6.17
DOSPIP.BAC		Transfers files from a DOS formatted disk to a RSTS/E disk	SMG - 6.19
DIRECT.BAC	<232>	Disk directory lister	SUG - 4.13
COPY.BAC	<1Ø4>	Copies entire tapes and disk cartridges	SUG - 4.4.7

[1]The abbreviation SUG refers to the RSTS-11 System User's Guide and SMG refers to the RSTS/E System Manager's Guide.

4.1.2 SPLER.CTL Files

Commands in the SPLER.CTL file create programs which require a job area larger than 8K words. VT5DPY additionally requires PRINT USING software. Table 4-2 describes these programs and gives an approximate job size for each. References in Table 4-2 are section numbers in either the RSTS-11 System User's Guide (SUG) or this guide (SMG).

To build these files, the system SWAP MAX value must have been set large enough when the DEFAULT initialization option was run. If the current job size is too small when BUILD creates a given program, the system generates the MAXIMUM CORE EXCEEDED error. After the BUILD program terminates, all of the SPLER.CTL files exist in the system library. Those program which generated the MAXIMUM CORE EXCEEDED error are not runnable and must be deleted to prevent further errors. If the queue management and batch facilities are not required, delete the programs QUE, SPOOL, BATCH, QUEMAN, and CHARS from the library and replace the special version of SHUTUP with the version supplied in the BUILD.CTL file set.

4.1.3 RJ278Ø.CTL File

Commands in the RJ278Ø.CTL file create the RJ278Ø.BAC program with a protection code of <232>. This file is for use by customers who have purchased the RSTS/E 278Ø software package and have included the appropriate software in the RSTS/E system. The program is described in the RSTS/E 278Ø User's Guide.

Table 4-2
SPLER.CTL Programs and Files

Program or File Name	Protection (if other than <124>)	Job Size (in words)	Description	Section Reference[1]
CHARS.BAC		8K	Creates the character generation file	SMG - App. E
QUE.BAC	<232>	1ØK	Creates requests for spooling programs	SUG - 4.11
QUEMAN.BAC		14K	Queue manager	SMG - 6.1Ø
SPOOL.BAC		1ØK	Line printer spooling program	SMG - 6.1Ø
BATCH.BAC		13K	Batch control spooling program	SUG - 4.12 SMG - 6.1Ø.3
RUNOFF.BAC	<232>	14K	Document formatting program	RUNOFF.DOC and RUNOFF USER'S GUIDE
VT5DPY.BAC	<232>	15K	System status display on VTØ5	SMG - 6.13
SHUTUP.BAC		8K	Performs standard system shut down and terminates spooling operations automatically	SMG - 6.2
ERRDIS.BAC		12K	Special version of ERRDIS not requiring chain module ERRDI1.BAC	SMG - 6.11
START.CTL	<6Ø>		Special version of START.CTL for systems with spooling	SMG - 6.1
CRASH.CTL	<6Ø>		Special version of CRASH.CTL for systems with spooling	SMG - 6.2
RUNOFF.DOC	<4Ø>		Explains use of sample file RUNOFF.RNO	See RUNOFF.BAC
RUNOFF.RNO	<4Ø>		Sample RUNOFF text file	See RUNOFF.DOC

[1]The abbreviation SUG refers to the RSTS-11 System User's Guide and SMG refers to the RSTS/E System Manager's Guide.

4.2 STARTING TIME SHARING TO BUILD THE LIBRARY

After the DEFAULT option described in Section 3.5 is executed, type START in response to the OPTION query to begin time sharing. The following sample dialog shows the procedures.

```
OPTION:  START
JOB OR SWAP MAX CHANGES?  <LF>
ANY MEMORY ALLOCATION CHANGES?  <LF>
CRASH DUMP?  <LF>

DD-MMM-YY?  2Ø-AUG-74
HH:MM?  8:55

CAN'T FIND FILE OR ACCOUNT
PROGRAM LOST - SORRY

READY
```

<LF> denotes typing the LINE FEED key.

The remaining steps depend upon the medium on which the RSTS/E software is stored. For the magtape procedure, go to Section 4.2.1; for disk cartridge, go to Section 4.2.2; and for DECtape, go to Section 4.2.3.

4.2.1 Building the System Library from Magtape

Perform the following steps:

Ensure that the write enable ring is removed from the reel labelled DEC-11-ORSLA-C-MA7 (7-track) or DEC-11-ORSLA-C-MA9 (9-track), SYSTEM LIBRARY & RELIABILITY TAPE.

Mount this tape on unit Ø. Ensure that no other drive is selected to Ø.

Ensure that the FILE PROT indicator is on.

Position the tape at its load point. (The LD PT indicator comes on.)

Set the ON-LINE/OFF-LINE switch to its ON-LINE position. Ensure that the READY indicator comes on.

4.2.1.1 Building the General System Library, BUILD.CTL - Users on all systems must execute the commands from the BUILD.CTL file. Type the following command to execute the BUILD program from either TU1Ø or TU16 magtape.

```
RUN MTØ:BUILD$
```

BUILD runs and prints the following lines:

```
SYSTEM BUILDER
SYSTEM BUILD?  YES
SYSTEM BUILD DEVICE?  MTØ
```

Type YES in response to the SYSTEM BUILD question and type MTØ (omit the colon) in response to the subsequent SYSTEM BUILD DEVICE question. The BUILD program begins executing commands in the BUILD.CTL file.

NOTE

If any errors occur in reading the magtape, retry the procedure on another drive. If no other drive is available, either request Field Service to align the heads on the drive or obtain a new magtape.

The console printout of the entire procedure is shown below.

```
RUN MT0:BUILD$
SYSTEM BUILDER
SYSTEM BUILD? YES
SYSTEM BUILD DEVICE? MT0
OLD MT0:$LOGIN
COMPILE SY0:LOGIN
CHAIN 'MT0:$BUILD' 9200

READY

READY

READY

^C
HELLO

RSTS V05C-01 SYSTEM TEST  JOB 2  KB0   25-JAN-75  01:58 PM
# 1 / 2
PASSWORD:
JOB(S) 1 ARE DETACHED UNDER THIS ACCOUNT
JOB NUMBER TO ATTACH TO?
1 OTHER USER(S) ARE LOGGED IN UNDER THIS ACCOUNT


READY

!********** BUILD.CTL - STANDARD LIBRARY PROGRAMS

RUN MT0:$PRIOR

'PRIOR' PRIORITY, BURST, MAXIMUM CHANGER

ENTER ANOTHER JOB NUMBER?

CURRENT STATISTICS ARE:
    -8 PRIORITY
     6 RUN BURST
   16K SIZE MAXIMUM
```

```
ANY CHANGES? YES
CHANGE PRIORITY? NO
CHANGE RUN BURST? NO
CHANGE SIZE MAXIMUM? YES
        CHANGE IT TO? 8

CURRENT STATISTICS ARE:
    -8 PRIORITY
     6 RUN BURST
    8K SIZE MAXIMUM

ANY CHANGES? NO

READY
OLD MT0:$BUILD
READY
COMPILE SY0:BUILD
READY
OLD MT0:$LOGOUT
READY
COMPILE SY0:LOGOUT
READY
OLD MT0:$UTILTY
READY
COMPILE SY0:UTILTY
READY
OLD MT0:$INIT
READY
COMPILE SY0:INIT
READY
OLD MT0:$SYSCAT
READY
COMPILE SY0:SYSCAT
READY
OLD MT0:$PRIOR
READY
COMPILE SY0:PRIOR
READY
OLD MT0:$TALK
READY
COMPILE SY0:TALK
READY
OLD MT0:$ANALYS
READY
COMPILE SY0:ANALYS
READY
OLD MT0:$SYSTAT
READY
COMPILE SY0:SYSTAT<40>
READY
OLD MT0:$UMOUNT
READY
COMPILE SY0:UMOUNT
READY
OLD MT0:$QUOLST
READY
COMPILE SY0:QUOLST
READY
OLD MT0:$ERRCPY
READY
COMPILE SY0:ERRCPY
READY
```

```
OLD MT0:$ERRDIS
READY
COMPILE SY0:ERRDIS
READY
OLD MT0:$ERRDI1
READY
COMPILE SY0:ERRDI1
READY
OLD MT0:$ERRCRS
READY
COMPILE SY0:ERRCRS
READY
OLD MT0:$SHUTUP
READY
COMPILE SY0:SHUTUP
READY
OLD MT0:$EDIT
READY
COMPILE SY0:EDIT<40>
READY
OLD MT0:$EDITCH
READY
COMPILE SY0:EDITCH<40>
READY
OLD MT0:$BACKUP
READY
COMPILE SY0:BACKUP
READY
OLD MT0:$BACKDK
READY
COMPILE SY0:BACKDK
READY
OLD MT0:$ODT
READY
COMPILE SY0:ODT
READY
OLD MT0:$PIP
READY
COMPILE SY0:PIP<40>
READY
OLD MT0:$DSKINT
READY
COMPILE SY0:DSKINT
READY
OLD MT0:$DOSPIP
READY
COMPILE SY0:DOSPIP<40>
READY
OLD MT0:$DIRECT
READY
COMPILE SY0:DIRECT
READY
OLD MT0:$COPY
READY
COMPILE SY0:COPY<40>
READY
OLD MT0:$FILCOM
READY
COMPILE SY0:FILCOM<40>
READY
```

```
OLD MT0:$MONEY
READY
COMPILE SY0:MONEY<40>
READY
OLD MT0:$GRIPE
READY
COMPILE SY0:GRIPE
READY
OLD MT0:$REACT
READY
COMPILE SY0:REACT
READY
OLD MT0:$TTYSET
READY
COMPILE SY0:TTYSET
READY
OLD MT0:$PLEASE
READY
COMPILE SY0:PLEASE
READY
OLD MT0:$INUSE
READY
COMPILE SY0:INUSE
READY
OLD MT0:$RESEQ
READY
COMPILE SY0:RESEQ<40>
READY
OLD MT0:$CREF
READY
COMPILE SY0:CREF<40>
READY

RUN PIP
PIP - RSTS V05C-01 SYSTEM TEST
#SY0:NOTICE.TXT<40><MT0:NOTICE.TXT$/FA
#SY0:HELP  .TXT<40><MT0:HELP  .TXT$/FA
#SY0:START .CTL    <MT0:START .CTL$/FA
#SY0:CRASH .CTL    <MT0:CRASH .CTL$/FA
#SY0:ACCT  .SYS    <MT0:ACCT  .SYS$/FA
#SY0:TTYSET.SPD    <MT0:TTYSET.SPD$/FA
#SY0:PIP   .TXT<40><MT0:PIP   .TXT$/FA
#SY0:ERRDIS.HLP    <MT0:ERRDIS.HLP$/FA
#SY0:RESEQ .DOC<40><MT0:RESEQ .DOC$/FA
#SY0:CREF  .DOC<40><MT0:CREF  .DOC$/FA
#SY0:BACKUP.TXT<40><MT0:$BACKUP.TXT/FA
#=BACKUP.BAC<232>/RE
#=BACKDK.BAC<232>/RE
#=DIRECT.BAC<232>/RE
#=TALK  .BAC<232>/RE
#=SYSTAT.BAC<232>/RE
#=UMOUNT.BAC<232>/RE
#=QUOLST.BAC<232>/RE
#=GRIPE .BAC<232>/RE
#=TTYSET.BAC<232>/RE
#=PLEASE.BAC<232>/RE
#=INUSE .BAC<232>/RE
#^C

READY

RUN UTILTY
'UTILTY' SYSTEM UTILITY PROGRAM
```

```
? LOGINS
? EXIT

READY

^C

READY

HELLO

RSTS V05C-01 SYSTEM TEST  JOB 2  [1,2]  KB0  25-JAN-75  02:25
JOB(S) 1 ARE DETACHED UNDER THIS ACCOUNT
JOB NUMBER TO ATTACH TO?  1
ATTACHING TO JOB 1

BUILD COMPLETE

READY
```

Execution of the commands in the BUILD.CTL file takes between 25 and 35 minutes. The tape rewinds several times at the beginning of the procedure but does not rewind after the statement OLD MTØ:LOGOUT$. Multiple copies of some programs are on the tape to reduce the number of rewinds required.

The end of execution is signalled by the messages BUILD COMPLETE and READY. Do not rewind or move the tape in any way. If the user is not including files from the SPLER.CTL and RJ278Ø.CTL file sets on his system, he need not build any more files and can proceed to Section 4.3 to create the ASCII text and message files. Proceed to Section 4.2.1.2 to build the spooling programs and files or skip to Section 4.2.1.3 to build the RJ278Ø program.

4.2.1.2 Building the Large Programs, SPLER.CTL - Building the programs from the SPLER.CTL file requires a user job area larger than 8K words as described in Section 4.1.2. To continue building the library, leave the magtape at its current position and type the following command.

```
RUN $BUILD
```

If the magtape is moved, the system rewinds the magtape before searching for the first file (SPLER.CTL). The above command runs the BUILD program from the system disk and leaves the tape at its current position.

The program prints the following two lines.

```
SYSTEM BUILDER
SYSTEM BUILD?  NO
```

Type NO in answer to the SYSTEM BUILD question. BUILD continues with the following questions.

```
AUXILIARY BUILD DEVICE?  MTØ
CONTROL FILE IS?  SPLER.CTL$
```

Type MTØ (omit the colon) as the device and type SPLER.CTL$ to specify the correct control file. BUILD begins executing the commands in the file SPLER.CTL. The complete console printout of the entire procedure is shown below.

```
RUN $BUILD
SYSTEM BUILDER
SYSTEM BUILD? NO
AUXILIARY BUILD DEVICE? MT0
CONTROL FILE IS? SPLER.CTL$
^C
HELLO

RSTS V05C-01 SYSTEM TEST   JOB 2  KB0  25-JAN-75  02:26
# 1 / 2
PASSWORD:
JOB(S) 1 ARE DETACHED UNDER THIS ACCOUNT
JOB NUMBER TO ATTACH TO?
1 OTHER USER(S) ARE LOGGED IN UNDER THIS ACCOUNT

READY

!********** SPLER.CTL - LARGE PROGRAM BUILD
OLD   MT0:$CHARS
```

```
READY
COMPILE SY0:CHARS
READY
RUN CHARS
READY
OLD MT0:$QUE
READY
COMPILE SY0:QUE
READY
OLD   MT0:$QUEMAN
READY
COMPILE SY0:QUEMAN
READY
OLD   MT0:$SPOOL
READY
COMPILE SY0:SPOOL
READY
OLD   MT0:$BATCH
READY
COMPILE SY0:BATCH
READY
OLD   MT0:$RUNOFF
READY
COMPILE SY0:RUNOFF
READY
OLD   MT0:$VT5DPY
READY
COMPILE SY0:VT5DPY
READY
OLD MT0:$SHUTUP.SPL
READY
COMPILE SY0:SHUTUP
READY
OLD   MT0:$ERRDIS.SPL
READY
COMPILE SY0:ERRDIS
READY
KILL "ERRDI1.BAC"
READY

RUN $PIP
PIP - RSTS V05C-01 SYSTEM TEST
#SY0:START .CTL     <  MT0:$START1.CTL
#SY0:CRASH .CTL     <  MT0:$CRASH1.CTL
#SY0:RUNOFF.DOC<40><  MT0:$RUNOFF.DOC
#SY0:RUNOFF.RNO<40><  MT0:$RUNOFF.RNO
#=QUE    .BAC<232>/RE
#=RUNOFF.BAC<232>/RE
#=VT5DPY.BAC<232>/RE
#^C

READY

HELLO

RSTS V05C-01 SYSTEM TEST  JOB 2  [1,2]  KB0  25-JAN-75  02:44
JOB(S) 1 ARE DETACHED UNDER THIS ACCOUNT
JOB NUMBER TO ATTACH TO?  1
ATTACHING TO JOB 1

BUILD COMPLETE

READY
```

Completion of the procedure is signalled by the BUILD COMPLETE message. To build the RJ278Ø program, proceed to Section 4.2.1.3; otherwise, proceed to Section 4.3 to create the ASCII text and message files.

4.2.1.3 Building the RJ278Ø Program, RJ278Ø.CTL - To build the RJ278Ø program, run BUILD from disk again and specify the RJ278Ø.CTL file as shown in the following sample printout. This procedure can be performed only if the system includes the RSTS/278Ø software and the job area is at least 1ØK words.

```
RUN BUILD
SYSTEM BUILDER
SYSTEM BUILD? NO
AUXILIARY BUILD DEVICE? MT0
CONTROL FILE IS? RJ2780.CTL
^C
HELLO

RSTS V05C-0 SYSTEM TEST  JOB 2  KB0   25-JAN-75  02:46
# 1 / 2
PASSWORD:
JOB(S) 1 ARE DETACHED UNDER THIS ACCOUNT



READY

!********** RJ2780.CTL - RJ2780 PROGRAM

OLD MT0:$RJ2780
READY
COMPILE SY0:RJ2780
READY
RUN $PIP
PIP -RSTS V05C-01 SYSTEM TEST
#=RJ2780.BAC<232>/RE
#^C

READY

HELLO

RSTS V05C-01  SYSTEM TEST   JOB 1  [1,2]  KB0  25-JAN-75  02:48
JOB(S) 1 ARE DETECHED UNDER THIS ACCOUNT
JOB NUMBER TO ATTACH TO?  1
ATTACHING TO JOB 1

BUILD COMPLETE

READY
```

The SPLER build procedure takes approximately 15 minutes. The tape does not rewind.

The RJ278Ø procedure takes approximately two minutes.

Completion of the procedure is signalled by the BUILD COMPLETE message. Proceed to Section 4.3 to create the ASCII text and message files.

4.2.2 Building the System Library from Disk Cartridge

Perform the following steps.

Mount the copy of the system library disk cartridge labelled DEC-11-ORSLA-C-HA SYSTEM LIBRARY AND RELIABILITY DECPACK on a free unit.

Place the LOAD/RUN switch on the drive to its RUN position and ensure that the READY indicator comes on.

Depress the WR PROT switch so that the WR PROT indicator goes off.

At this point, the user must type in the following BASIC-PLUS program to logically mount the disk cartridge on the system. First, type the NEW MOUNT command to clear memory, after which the system prints READY. After entering the program, type the RUNNH command to execute it.

NOTE

When entering the statement at line number 5Ø, replace <UNIT NUMBER> with the number of the drive being used. For example, if drive 1 is used, type the following.

```
50 M%(25%) = 1
```

The entire procedure is as follows.

```
NEW MOUNT

READY

10      DIM M%(30%)
20      CHANGE SYS(CHR$(6%)+CHR$(-10%)+"SYSLIB") TO M%
        ! GET THE PACK ID IN RADIX-50
30      M%(0%)=26% : M%(1%)=6%    : M%(2%)=3%
40      M%(3%)=0%  : M%(23%)=68% : M%(24%)=75%
50      M%(25%)=<UNIT NUMBER>
        ! REPLACE <UNIT NUMBER> WITH ACTUAL UNIT ON
        ! WHICH DISK IS RUNNING.
60      M%(26%)=255%
        ! SET UP FOR THE MOUNT SYS FUNCTION
70      CHANGE M% TO M$
80      M$=SYS(M$)
        ! MOUNT THE DISK
90 END

RUNNH

READY
```

If any errors occur, perform the following recovery procedures.

a) Ensure that the drive unit is READY and is write enabled.

b) Type the LISTNH command and check each line of the program to ensure each is exactly correct. For more information on inputting and editing BASIC-PLUS programs, see Sections 2.2.2 through 2.2.5 of the RSTS-11 System User's Guide.

If no errors are found by performing steps a and b above, retry the entire procedure several times. If possible, try different drive units and replace line 50 with the proper unit number. If the error persists, the drive heads may require alignment.

4.2.2.1 Building the General System Library, BUILD.CTL - The BUILD.CTL file must be executed on all systems. For the purposes of example, drive unit 1 is used in all commands. Wherever DK1 appears in the examples, substitute the device designator of the drive unit on which the disk cartridge is mounted. Type the following command to run the BUILD program from the proper disk cartridge.

```
RUN DK1: BUILD$
```

BUILD prints the following lines:

```
SYSTEM BUILDER
SYSTEM BUILD? YES
SYSTEM BUILD DEVICE? DK1
```

Type YES in answer to the SYSTEM BUILD question and type the proper device designator (omit the colon) to specify the drive unit in response to the SYSTEM BUILD DEVICE question. The unit number must be included since DK alone refers to the system device and DKØ refers to unit Ø of the RK drives. The BUILD program begins executing commands in the BUILD.CTL file. The printout for the program and the guidelines for completion time are the same as that given in Section 4.2.1.1 except that MTØ is replaced by the disk drive designator. Upon completion of the procedure, proceed to Section 4.2.2.2 to build spooling library programs, to Section 4.2.2.3 to build the RJ278Ø program, or to Section 4.3 to create the ASCII text and message files.

4.2.2.2 Building the Large Programs, SPLER.CTL - Building the programs from the SPLER.CTL file requires a user job area larger than 8K words as described in Section 4.1.2. To continue building the library, type the following command.

```
RUN $BUILD
```

The program prints the following lines.

```
SYSTEM BUILDER
SYSTEM BUILD?  NO
```

Type NO in answer to the SYSTEM BUILD question, after which BUILD prints, in turn, the following questions.

```
AUXILIARY BUILD DEVICE?  DK1
CONTROL FILE IS?  SPLER.CTL$
```

Type the proper device designator (omit the colon) and type SPLER.CTL$ to specify the correct control file. BUILD begins executing commands in the file and prints messages similar to those given in Section 4.2.1.2.

When the procedure is completed, proceed to Section 4.2.2.3 to build the RJ278Ø program or to Section 4.3 to create the ASCII text and message files.

4.2.2.3 Building the RJ278Ø Program, RJ278Ø.CTL - To build the RJ278Ø program, run BUILD again, specify the proper auxiliary device, and specify RJ278Ø.CTL as the control file. This procedure must be performed if the system includes the RSTS/278Ø software. The job area must be at least 1ØK words. The entire procedure is similar to that shown in the sample printout of Section 4.2.1.3 except that the proper device designator replaces MTØ. Upon completion, proceed to Section 4.3 to create the ASCII text and message files.

4.2.3 Building the System Library from DECtape

Perform the following steps:

Mount the DECtape reel labelled DEC-11-ORSCA-C-UA1, SYSTEM LIBRARY TAPE, on drive unit Ø.

Mount DECtape reel labelled DEC-11-ORSCA-C-UA2, SYSTEM LIBRARY TAPE, on drive unit 1.

Set the WRITE ENABLE/WRITE LOCK switch on drive Ø and drive 1 to its WRITE LOCK position.

Set the REMOTE/OFF/LOCAL switch on drive Ø and drive 1 to its REMOTE position.

The tape mounted on unit Ø contains BUILD.CTL. The tape mounted on unit 1 contains the SPLER.CTL and RJ278Ø.CTL files. Ensure that the correct tapes are mounted before proceeding.

4.2.3.1 Building the General System Library, BUILD.CTL - The BUILD.CTL file must be executed on all systems. Type the following command to execute the BUILD program from DECtape.

```
RUN DTØ:BUILD
```

BUILD runs from the DECtape and prints the following lines:

```
SYSTEM BUILDER
SYSTEM BUILD?  YES
SYSTEM BUILD DEVICE?  DTØ
```

Type YES in response to the SYSTEM BUILD question and type DTØ (omit the colon) in response to the subsequent SYSTEM BUILD DEVICE question. The BUILD program begins executing commands from the BUILD.CTL file. The entire procedure takes approximately 45 minutes. The printout for the program is similar to that shown for magtape in Section 4.2.1.1

Upon completion of the procedure, BUILD prints the BUILD COMPLETE and READY messages. Go to Section 4.2.3.2 to build spooling library programs, to Section 4.2.3.3 to build the RJ278Ø programs, or to Section 4.3 to create ASCII text and message files.

4.2.3.2 Building the Large Programs, SPLER.CTL - Building the programs from the SPLER.CTL file requires a user job area larger than 8K words as described in Section 4.1.2. To continue the build procedure, use the system library DECtape mounted on unit 1.

Type the following command to start the procedure.

```
RUN $BUILD
```

BUILD prints the following lines:

```
SYSTEM BUILDER
SYSTEM BUILD?  NO
```

Type NO in response to the SYSTEM BUILD question, after which BUILD prints, in turn, the following questions.

```
AUXILIARY BUILD DEVICE?  DT1
CONTROL FILE IS?  SPLER.CTL
```

Type DT1 (omit the colon) as the device designator and type SPLER.CTL to specify the correct control file. BUILD runs and prints messages similar to those shown in Section 4.2.1.2 for magtape. The entire procedure takes approximately 25 minutes.

When the BUILD COMPLETE and READY messages appear, proceed to Section 4.2.3.3 to build the RJ278Ø program or to Section 4.3 to create the ASCII text and message files.

4.2.3.3 Building the RJ278Ø Program, RJ278Ø.CTL - Note that the RJ278Ø program must be installed only if the system includes the RSTS/278Ø software. The job area must be at least 1ØK words.

Execute the BUILD program again as described in Section 4.2.3.2 but type RJ278Ø.CTL to specify the control file. The complete procedure is summarized in the following sample printout.

```
RUN $BUILD
SYSTEM BUILD
SYSTEM BUILDER?  NO
AUXILIARY BUILD DEVICE?  DT1
CONTROL FILE IS?  RJ278Ø.CTL
```

BUILD runs and prints messages similar to those shown for magtape in Section 4.2.1.3. The procedure takes only a few minutes. When the BUILD COMPLETE and READY messages appear, proceed to Section 4.3 to create the ASCII text and message files.

4.3 CREATING THE ASCII TEXT AND MESSAGE FILES

After the BUILD program terminates, remove the tape or disk cartridge containing the system library files and ensure that it is stored in a safe place with the remainder of the RSTS/E kit.

NOTE

For disk cartridge distribution, run the UTILTY program and execute the DISMOUNT command to dismount the system library cartridge.

At this point in the process of building the system library, run the PIP system program and create the required ASCII text files. (For information concerning the use of PIP, refer to Section 4.4 of the RSTS-11 System User's Guide.) The subsequent sections describe the procedures to follow when building the ASCII text files and contain references to further descriptions of the use and contents of the files. Editing conventions for the use of the RUBOUT key and CTRL/U as described in Section 2.2.3 of the RSTS-11 System User's Guide apply when using the PIP system program.

4.3.1 System Message File NOTICE.TXT

The file NOTICE.TXT is printed when a user is successfully logged into the system as described in Section 2.1.2 of the RSTS-11 System User's Guide. In this manner, the system manager can relay to the user information concerning operation of the installation or changes or additions to the system. Prior to creating his own NOTICE.TXT file, the system manager can print the sample file in the library on the console printer.

```
RUN PIP
PIP - RSTS  VØ5C-Ø1  SYSTEM #88Ø
#KB:/FA<NOTICE.TXT
WELCOME TO RSTS/E TIME SHARING
#
```

A tailored NOTICE.TXT file can be created and can replace the sample file by imitating the following sample procedure.

```
#SYØ:NOTICE.TXT<4Ø><KB:/FA
WELCOME TO RSTS/E SYSTEM #88Ø
HOURS OF OPERATION ARE:

     MON-FRI  Ø8ØØ TO 17ØØ HOURS
     SAT-SUN  ON REQUEST

TRANSMIT USER COMMENTS VIA THE GRIPE
SYSTEM PROGRAM
↑Z
#
```

The ↑Z at the end of the text file is necessary. It is the result of typing the CTRL/Z[1] combination and signals the end of the ASCII text, closes the file properly and returns control to PIP as indicated by the # character being printed again. The system manager can update the file as needed.

4.3.2 System Help File HELP.TXT

The file HELP.TXT is printed when a user types HELP at a logged-out terminal as described in Section 2.1.4 of the RSTS-11 System User's Guide. The user can print the sample file stored in the system library by typing the following command string. (The example assumes that PIP is still running from the previous procedures.)

```
#KB:/FA<HELP.TXT
TO GET ON-LINE AND USE RSTS/E, FOLLOW THE
INSTRUCTIONS FOUND IN CHAPTER 2 OF THE
RSTS-11 SYSTEM USER'S GUIDE.
#
```

If the text is satisfactory, but the system manager wishes to add some local information, he can use the append feature of PIP, delete the old file and rename the new file.

```
#SYØ:NEW.TXT<HELP.TXT,KB:/FA
CONTACT MR. JONES FOR A PROJECT-PROGRAMMER
NUMBER AND PASSWORD.
↑Z
#HELP.TXT/DE
#SYØ:HELP.TXT<4Ø>=NEW.TXT/RE
#
```

[1]See the description of control characters in Chapter 3 in the RSTS-11 System User's Guide and of CTRL/Z in Section 9.5 of the BASIC-PLUS Language Manual.

The sample HELP.TXT file can be replaced by using a command string similar to the one shown in Section 4.3.1.

4.3.3 Control Files START.CTL and CRASH.CTL

Two control files are required by the INIT system program whenever the RSTS/E system is initialized. The use and content of both of these files are explained in Section 6.1. The procedures described here show how to print the sample files and how to replace the sample files with versions created to suit the needs of the local installation.

It is essential that these files contain the information required to properly initialize the system for time sharing operations. Before the system manager attempts to modify or replace the contents of the example files, it is highly recommended that he thoroughly familiarize himself with the concepts presented in Section 6.1.

```
#KB:/FA<START.CTL, CRASH.CTL
     (Both files are printed.)
#SYØ:START.CTL<KB:/FA
     (Type new version.)
↑Z
#SYØ:CRASH.CTL<KB:/FA/UP
     (Type new version.)
↑Z
#
```

As a result of following the above procedures, the sample files in the system library are replaced by versions the user enters.

4.3.4 Terminal Speed Characteristics File TTYSET.SPD

The system manager must create the terminal speed characteristics file TTYSET.SPD. Its use and contents are explained in Section 6.7.1. An entry in TTYSET.SPD must be created for each local or remote line which allows a variable baud rate. The following sample dialogue shows how to print the sample file in the library and how to replace the sample file with the new version.

```
#KB:/FA<TTYSET.SPD
     (The sample file is printed.)
#SYØ:TTYSET.SPD<KB:/FA
     (Type new version)
↑Z
#
```

As a result of following the above procedure, the sample file in the system library is replaced by the new version.

4.3.5 Standard Account File ACCT.SYS

As a convenience, the REACT system program is designed to create a large number of user accounts automatically. To accomplish this, the system manager must create the ACCT.SYS file in the system library. The contents of the ACCT.SYS file are explained in Section 6.4.3. A sample of ACCT.SYS is created by the BUILD program. This sample file can be printed and replaced by performing the following procedures. (It is suggested that, at this time, the system manager make entries in ACCT.SYS to create a privileged and a non-privileged account for himself.)

```
#KB:/FA<ACCT.SYS
     (The sample file is printed.)
#SYØ:ACCT.SYS<KB:/FA
     (Type new version.)
↑Z
#↑Z

READY
```

The new version of ACCT.SYS replaces the sample ACCT.SYS in the system library and the PIP system program run is terminated. Proceed to Section 4.4 to create the user accounts.

4.4 CREATING USER ACCOUNTS

The system manager can run REACT and use the STANDARD function to create the user accounts in the MFD of the system disk. The explanation of REACT is given in Section 6.4. The following procedures describe the method used to create the user accounts from information in the ACCT.SYS file.

```
RUN REACT
'REACT' SYSTEM ACCOUNT MANAGER
FUNCTION?  STANDARD
ALL ACCOUNTS IN $ACCT.SYS ARE NOW ENTERED
FUNCTION?  ↑Z
READY
```

REACT prints an identification message and a request for a function. Type STANDARD or S and the REACT program creates user accounts from information in the ACCT.SYS file. When the standard function is completed, an advisory message is printed and the FUNCTION request reprinted. Type CTRL/Z (↑Z) to terminate REACT.

At this point, the system disk is built and the RSTS/E system is fully operational. It is suggested that the system manager shut down the system as described in Section 6.2 and restart it to test the new START.CTL file.

CHAPTER 5

SYSTEM OPERATION, SYSTEM ACCOUNTS, AND PRIVILEGE

This chapter describes halt, bootstrap and restart procedures, system accounts, and privilege program and account status on RSTS/E.

5.1 HALTING THE RSTS/E SYSTEM

A halt in the RSTS/E system is caused by an orderly occurrence of events or by randomly setting the CPU console HALT/ENABLE switch to its HALT position. The PDP-11/45 is considered in a halt state when both the RUN and PAUSE indicators on the CPU console panel are not lit. Otherwise, the CPU is running. For the PDP-11/40, consult the following chart to determine the state of the CPU based on the condition of the Console Status Lights.

Condition		Meaning
RUN light	CONSOLE light	
OFF	OFF	CPU is powered down or bus is hung.
OFF	ON	Impossible or bus is hung
ON	OFF	CPU is running
ON	ON	CPU is halted

The SHUTUP program described in Section 6.2 shuts down the RSTS/E system in an orderly fashion. SHUTUP eventually halts the CPU at address 54. The program ensures that all files are properly closed and that system accounting information is accurately updated. The halt leaves the program counter loaded in such a fashion that depressing the CONT switch on the CPU console panel causes the RSTS/E initialization code to be bootstrapped into memory from the RSTS.CIL file stored on the system disk.

Halting RSTS/E by moving the HALT/ENABLE switch on the CPU console panel to its HALT position is dangerous. Clean-up operations may not be completed; disk storage allocation tables and file directories may be left in obsolete states; file data can consequently become corrupted; and accounting information may be lost. The only way to recover from such a disorderly halt and to salvage possible vital file information is to raise the HALT/ENABLE switch back to its ENABLE position before any other action is taken, to depress the CONT switch, and thereby to return the RSTS/E system to the state in which it was before the random halt occurred.

5.2 STARTING UP RSTS/E: BOOTSTRAPPING RSTS/E INTO MEMORY

Starting up RSTS/E is a two step operation. The first step requires loading the RSTS/E initialization code into memory and the second step involves using one of the RSTS/E initialization options. This section describes the ways that the RSTS/E disk can be bootstrapped into memory. Chapter 3 describes and explains the RSTS/E initialization options.

The initialization code in RSTS/E exists in the system file RSTS.CIL on the system disk. The code is once-only code, which is loaded into the lower 28K words of memory and overlaid after its execution by system and user programs. The following sections describe the ways that initialization code can be bootstrapped into memory.

5.2.1 Bootstrapping RSTS/E via a Hardware Bootstrap Loader

The procedures to follow when bootstrapping RSTS/E depend on the type of hardware bootstrap loader and the type of disk used as the system device. The following steps explain the procedure.

a) Make sure that the system disk is physically mounted on disk unit Ø. (This does not apply if the system disk is an RF type disk.)

b) Make sure that the system disk is READY and is set to WRITE ENABLE.

c) Ensure that the console terminal is on line.

d) Refer to Appendix F for the proper bootstrap operation.

When the initialization code is loaded, it prints the system header and installation name followed by the OPTION: message. If no messages appear after performing the bootstrap procedure, ensure that the console terminal is on line and retry the bootstrap procedure. If the initialization code prints the message FATAL ERROR OCCURRED DURING CILUS PHASE, chances are the system disk is WRITE PROTECTED. To recover, write enable the disk and retry the bootstrap procedure. *

If the automatic restart facility is to be enabled, set the CPU Switch Register to 777777; that is, all switches are in the up position. The automatic restart facility remains enabled as long as the CPU Switch Register remains set to 777777. If any switch between positions 15 and Ø is set to its Ø position (that is, to its down position), the automatic restart facility is disabled for the duration of this condition.

This method of bootstrapping the initialization code is independent of any previous contents of memory and requires only that the system image be the first Core Image Library (CIL) module in the CIL file and that a special bootstrap routine reside in the first 256 words of the system disk. The RSTS.CIL file can never be moved from its place on the system disk. These conditions apply on all RSTS/E systems.

5.2.2 Bootstrapping RSTS/E After a System Halt

If the RSTS/E system halts as a result of the SHUTUP program being run or as a result of a catastrophic error or system crash, the halt address is always 54. The address lights indicate 56 in such a case, since the program counter contains the address of the next sequential instruction. (See Section 5.3.1 for a description of catastrophic errors and system crashes.) The manner in which the user recovers in such a situation depends upon his configuration and the results he wants. The alternatives are described in the following two sections.

5.2.2.1 Requesting a Memory Dump and Automatic Restart - If the RSTS/E system halts at address 54 and the crash dump facility was enabled at the start of time sharing operations, the user can request a memory dump and automatic restart. The following procedures prescribe the steps to accomplish a memory dump and automatic restart.

a) Ensure that the CPU is in a halt state as described in Section 5.1.

b) Ensure that the CPU HALT/ENABLE switch is set to its ENABLE position.

c) Set the CPU Switch Register to ØØØØ52.

d) Depress the CPU LOAD ADRS switch.

e) Set the CPU Switch Register to 777777.

f) Depress the CPU START switch.

After RSTS/E starts from address 52, it checks to determine if the CPU Switch Register is set to 777777. If the Switch Register is set to 777777, the system writes the contents of all critical memory into the CRASH.SYS file and proceeds immediately to a loader routine which causes the RSTS/E initialization code to be bootstrapped into memory from the system disk in the special automatic restart mode described in Section 5.3.2.

If the crash dump facility was enabled at the start of time sharing operations but, after starting from address 52, the system finds the Switch Register set to something other than 777777, a halt immediately occurs again at address 54. If the crash dump facility was not enabled at the start of time sharing operations, the system halts immediately at address 54, and the user, to recover, can follow the procedures described in Section 5.2.2.2.

5.2.2.2 Requesting an Initialization Option - If the RSTS/E system halts at address 54, the user can bootstrap the initialization code in the normal start up mode. The procedures to request an initialization option are as shown below.

a) Ensure that the CPU is in a halt state as described in Section 5.1.
b) Ensure that the CPU HALT/ENABLE switch is set to its ENABLE position.
c) Set the CPU Switch Register to 000050.
d) Depress the CPU LOAD ADRS switch.
e) Depress the CPU START switch; it returns automatically.

Control is passed to the initialization code, which prints the system installation name followed, on the next line, by the OPTION query. See Section 3.1 for the valid responses to the OPTION query. If the system halts at address 54 as a result of the SHUTUP system program being run or as a result of the Switch Register being set to something other than 777777 when the crash dump facility was enabled and a catastrophic error or system crash occurred, the same results as described above can be produced by simply depressing the CONT switch on the CPU console panel.

5.2.3 Bootstrapping RSTS/E from ROLLIN

If the stand-alone program ROLLIN is in memory and if the RSTS/E system disk is physically mounted, in the READY condition and in the WRITE ENABLE condition, the RSTS/E system can be bootstrapped into memory. The user can refer to Chapter 4 of the PDP-11 ROLLIN Utility Program document, DEC-11-OROAA-B-D, for the correct command and procedure. After RSTS/E is bootstrapped into memory, it prints the system installation name on the console keyboard printer followed, on a second line, by the OPTION query. See Section 3.1 for the valid responses to the OPTION query.

5.3 AUTOMATIC RECOVERY AND RESTART FACILITIES

5.3.1 Nature and Causes of Catastrophic Errors and System Crashes

A catastrophic error or a system crash is an error-trap to vector 4 or vector 10. (See Section 2.7 on Processor Traps in either the PDP11/40 Processor Handbook or the PDP11/45 Processor Handbook. See

Section 5.3.2 below for the distinction between catastrophic error and system crash.) Such traps can be caused, for example, by referring to a nonexistent (or non-responding) UNIBUS address (bus time-out trap), by referring to an odd address with an instruction that requires a word address, or by attempting to execute a reserved or nonexistent instruction.

Catastrophic errors and system crashes, therefore, can be due to any of four types of problems: (a) configuration errors, (b) privileged account programming errors, (c) hardware malfunctions, and (d) system software malfunctions. Each of these is discussed individually below.

5.3.1.1 Configuration Errors - If the software configuration and the hardware configuration do not correspond exactly, bus time out traps occur whenever the software attempts to address a peripheral interface which, for the software, logically exists but which for the hardware does not physically exist. Thus, RSTS/E system software must be generated to reflect the actual hardware on which the system runs.

5.3.1.2 Privileged-Account Programming Errors - The RSTS/E system software is designed to protect itself against programming errors perpetrated under non-privileged accounts. The system itself, upon detecting such an error, aborts execution of the potential error and reports a corresponding error message to the guilty user.

The RSTS/E software is vulnerable, however, to certain types of errors perpetrated under privileged accounts. By intent and design, the system manager (and those to whom he assigns any [1,*] account) have been given extensive powers which are not available under non-privileged accounts. Refer to Section 5.5 for a discussion of privilege. These powers allow privileged users to modify System Library programs or to create utility programs in such a fashion that they can access parts of memory. Extreme care must be used when programming with privileged SYS system functions. A mistake can cause the system to take an error trap.

5.3.1.3 Hardware Malfunctions - Hardware malfunctions can be responsible for crashing the system. If unexplainable and random-type system crashes or catastrophic errors occur (particularly on systems which

hitherto have been functioning well), it is likely that a hardware problem has arisen. In such cases a DEC Field Service Representative should be contacted.

5.3.1.4 System Software Malfunctions - Although every attempt has been made to detect and eliminate system software errors, the paths and routes which control can take through the RSTS/E software are incalculably numerous. It is possible that, given certain conditions and certain sequences, the RSTS/E software can trap to vector 4 or vector 10. If a problem of this type is discovered (it should be reproducible in a defined environment and under defined conditions), a DEC Software Specialist should be contacted. If new problems of this type become known, DEC reports them as described on the next to last page of this guide.

5.3.2 Automatic Recovery from Catastrophic Errors and Crash Dump

Whenever a trap occurs to vector 4 or vector 10, the system distinguishes the trap as one of two categories: it is either (a) a catastrophic error which affects only one particular user or (b) a system crash for which some software or hardware problem is possibly responsible. The handling of system crashes is treated below.

The handling of catastrophic errors is as follows. The system determines which user was responsible for the error-trap. It flags that user's job with a special code which causes the system to reinitialize that user's job area completely when it is next his turn to run. The system prints on that user's terminal the message

```
CATASTROPHIC ERROR
PROGRAM LOST-SORRY
```

The reinitialized user is in the same state as he would be if he had just logged into the system. The system resumes normal time sharing operations.

When the system detects a condition from which it cannot recover, it performs an automatic restart only if both of two conditions are fulfilled:

a) The crash-dump facility must have been enabled at system start up time (possible only when the CRASH.SYS file exists), and

b) The CPU's Switch Register must currently be set to 777777.

If either condition is not fulfilled, the system does not take the automatic restart path but simply halts at address 54.

If the system halts at address 54, the operator may choose one of two procedures.

a) He depresses the CPU Console CONTinue switch, which causes the system to be bootstrapped into normal system start up mode.

b) The operator starts the CPU at address 52 with CPU Switch Register set to 777777 (see Section 5.2.2.2). This causes the system first to write the contents of memory onto the CRASH.SYS file (provided the crash-dump facility had been enabled) and then to be bootstrapped from disk in the special automatic restart mode described below.

If the system takes the automatic restart path, no halt occurs. Instead, the system first writes the critical contents of memory into the CRASH.SYS file and then bootstraps itself into memory from the system disk. After the system has been bootstrapped into memory, control jumps to the initialization routines. At this point the system recognizes the fact that it was not activated through a normal system start up but rather through an automatic restart and consequently initializes itself in automatic restart mode. If two system crashes occur within the same minute (more accurately stated, two error-traps within the same minute), the system halts at address 54. This protects the system against an infinite loop of error-traps caused by some repeating hardware malfunction.

5.3.3 Automatic Restart Mode Initialization

When the system is initialized in automatic restart mode, control by-passes all parts of the initialization code which call for operator intervention and initializes the system using information already stored in memory. The system logs Job 1 into the system on KB0: under account [1,2] and causes it to run the System Library program INIT beginning at line 100. Since, in automatic restart mode, INIT begins at line 100 (rather than at its lowest line number), it takes directions from the CRASH.CTL System Library file (rather than from the START.CTL file). The CRASH.CTL file must contain INIT commands which perform all operations the system manager considers necessary in the case of an automatic restart. See Section 6.1 concerning the control of system start up.

The printout which appears on the console terminal is similar in format to the following. Section 6.1 explains why most of these lines appear.

```
SYSTEM RELOADING        !!!
READY
SYSTEM INITIALIZATION PROGRAM

RUN $ERRCPY
READY
DETACHING
```

As stated at the end of Section 5.3.2, two error-traps within the same minute halt the system at address 54.

5.4 SYSTEM ACCOUNTS

RSTS/E systems have three system accounts which are integral to the operation of the system and have auxiliary accounts for more efficient operation of the system. The MFD account [1,1] is used on the system device and other disk devices in the system to control access. It is described in Section 5.4.1. The system library account [1,2] is used by the RSTS/E system to manage a library of generally available and restricted use system programs and message and control files. It is described in Section 5.4.2.

system account [0,1] contains RSTS/E Monitor files and routines which are critical to the operation of the system. System account [0,1] is described in Section 5.4.3. The auxiliary accounts are explained in Section 9.1.1 of the BASIC-PLUS Language Manual.

5.4.1 MFD Account [1,1] on the System Device

Access to the RSTS/E system is controlled by the use of project-programmer numbers and passwords. The system manager, operating under his privileged account, creates a new account by the system program REACT. The project-programmer number and password of the new account are given, along with other information, to allow a user the means to access system facilities.

When a new account is created, the new account information is stored on the system device under the MFD account [1,1]. The password is stored in packed Radix-50 format. When the new user first creates a file, an area is created on the system device which is related directly to the user's account (project-programmer number). This area is called the User File Directory or UFD. The UFD contains information concerning the files created under that account number.

The account [1,1] contains a catalog of information of all User File Directories on the system and is called the Master File Directory or MFD. When a user attempts to gain access to the RSTS/E system by giving his account and password, the system program LOGIN runs automatically. LOGIN checks the MFD on the system device to determine whether the account number and password given compare with one stored in the MFD. If so, the user is allowed access to the system.

Other account information is stored in the MFD for each account in the system. This information is summarized in Table 5-1.

The account information in the MFD is accessed by various system programs. The LOGIN system program has already been mentioned. The MONEY system accounting program references the accumulated usage information. The system manager uses the MONEY system program to read and reset usage information. The disk storage information is referenced by the LOGOUT system program. The system manager can change the quota

Table 5-1
Account Information Stored in the MFD on the System Device

Type	Description	Explanation
Identification	Project-programmer number (account)	Refer to Chapter 9 of BASIC-PLUS Language Manual.
	Password	6 characters stored within 2 words in Radix-50 format
Accumulated Usage	Central Processor Unit (CPU) time (Run Time)	Processor time the account has used to date in tenths of a second.
	Connect time (login time)	Number of minutes the user has been connected to the system via a terminal or remote line.
	Kilo-core-ticks (kct's	Core usage factor. One kct is the usage of 1K words of memory for one tenth of a second.
	Device time	Number of hours and minutes of peripheral device time the account has used.
Disk Storage	Quota	Number of 256-word blocks the user is allowed to retain at log out time.

by use of the UTILTY system program. Refer to the descriptions of the individual system programs in Chapter 6 for details.

A facility is provided whereby the system manager can write programs which access the information in the MFD. See the description of the system function SYS in Chapter 7.

5.4.1.1 MFD on Other Devices - The system disk exists in what is called the public structure. Additional disk devices can be added to the public structure and to what is called the private structure. Each device that is added to the system also contains its own MFD.

The MFD of each additional device is created when the system manager uses the initialization option DSKINT. The MFD of any device contains account and storage information for that device only.

The MFD on non-system disks is treated differently from the MFD on the system disk. The RSTS/E system allocates space for a user's file on the disk in the public structure that has the most free space. If the user's account is not in the MFD of that disk with the most free space, his account information is added dynamically into the MFD of that disk and a UFD is created for the user on that disk. The MFD on a disk within the private structure is different. If a user desires to create a file on a disk within the private structure, he cannot gain access if his account information does not already exist in the MFD of that device. The system manager or a privileged user grants access to a disk in the private structure by entering the account information on the desired disk with the REACT system program.

The MFD on devices within the public and private structure aside from the system device contains its own functional information. The information is pack label and account information.

The pack label information consists of pack cluster size, pack status (private or public), and pack identification (id). The pack id is the 6-character name in Radix-50 format given at the time the disk was initialized via the initialization option DSKINT. The pack id is utilized whenever the disk is logically mounted via the system program UTILTY or via INIT. (A distinction must be made between physical mounting and logical mounting. The disk is mounted physically by making the hardware ready to use the disk. The disk must be mounted logically by the system program UTILTY or from commands in the START.CTL or CRASH.CTL files by the INIT system program. The disk must also be logically dismounted before it is physically dismounted if corrupted disk structures are to be avoided.)

5.4.2 System Library Account [1,2]

The system library account [1,2] is created on the system device during the DSKINT operation of building the system disk. During subsequent operations of building the system disk, the system manager creates the contents of the system library account [1,2]. This section describes briefly the contents of account [1,2]. The entire content of the account is tabulated in Chapter 4.

The system library catalogs, as files, system programs which are available to general users and to privileged users. It also contains text and control files used by system programs.

For operational purposes, the system library is accessed automatically during normal system start-up. As a result of specifying the START option when RSTS/E is bootstrapped into memory, the console keyboard is logged in automatically under account [1,2]. At this time, the commands in the file START.CTL, stored under account [1,2], are executed by the INIT system program, which is also stored under account [1,2]. Unless the system manager desires to add to or modify the contents of the system library after the system disk has been built, he need not log into the system under account [1,2].

For automatic restart purposes, the system library file CRASH.CTL is used by the INIT system program when recovering from system crashes. When automatic restart mode is entered, the RSTS/E system performs actions similar to those described above for normal start-up, except that the commands of the system library file CRASH.CTL are executed.

Two files, HELP.TXT and NOTICE.TXT, are provided to supply information to the user. The HELP.TXT file is printed automatically by the system program LOGIN if a user types HELP at a logged-out terminal. The NOTICE.TXT file is printed automatically by the system program LOGIN after a user has typed a valid account number and password during the log-in procedure. The NOTICE.TXT file provides a means by which the system manager can communicate up-to-date installation information to users. One other system library information file, PIP.TXT, is printed out when the user types /HELP while running the related system program, PIP.

The file GRIPE.TXT is created dynamically when a user runs the GRIPE system program. The system manager has read and delete access to the GRIPE.TXT file by means of the GRIPE system program.

The file QUEUE.SYS is the system queue file and contains user requests created by the QUE program. The system queue manager program QUEMAN creates the file and initializes it. All spooling programs running on the system receive their requests from the QUEUE.SYS file through the QUEMAN program.

The ERRLOG.FIL file contains a history of hardware errors logged on the RSTS/E system. The ERRCPY program transfers the error data from monitor storage to the file ERRLOG.FIL. The system manager gains information on its contents and deletes its contents by running the ERRDIS system program.

5.4.3 System Account [Ø,1] on the System Device

The system account [Ø,1], like the system library account, is created on the system device during the DSKINT operation of building the system disk. Account [Ø,1] has a password identical to the pack id, and it contains name and retrieval information for system files. Account [Ø,1] is used solely by the RSTS/E Monitor to execute various operational and control actions. In this respect, it is a direct access account for use by the RSTS/E Monitor, and no user need be concerned with referencing its contents. This section describes the contents of the account in order to familiarize the system manager with the internal operations of the RSTS/E system.

The system manager creates and organizes the files stored under account [Ø,1] when he builds the system disk and uses the REFRESH option. The possible contents of account [Ø,1] are presented for reference in Chapter 3 and described in the following paragraphs.

The file SATT.SYS is the mechanism by which RSTS/E controls the allocation and deallocation of storage space for the disk. The file maps the entire space on the disk in a bit map table called

a SAT (storage allocation table). Each bit in a SAT represents either allocated or unallocated space. The system sets a bit in the SAT to 1 when that space is allocated for any purpose.

The system allocates storage space in terms of clusters. Each bit in the SAT represents one cluster of disk space. A cluster is a fixed number of contiguous 256-word blocks of storage on the disk. The cluster size, or cluster factor, defines how many contiguous 256-word blocks are contained in the cluster.

Cluster sizes in RSTS/E are defined for disks, directories, and files. Table 5-2 presents the types of clusters and related information.

Table 5-2
Valid Cluster Size Ranges

Cluster Size	Minimum Size	Maximum Size	When Defined
Pack (for non-system disk, public and private)	1 2, for RP03 only	16_{10}	At initialization time via DSKINT option (stored in MFD)
Directory (both for MFD and UFD)	Pack Cluster Size	16_{10}	At creation of the directory via either the DSKINT initialization option, REACT, or SYS system function.
File	Pack Cluster Size	256_{10}	At creation of the file via either an OPEN or OPEN FOR OUTPUT.

The system manager specifies the cluster size of the disk when he initializes it and gives a value in response to the PACK CLUSTER SIZE question. The pack cluster size defines the number of contiguous 256-word blocks a cluster comprises and, therefore, the extent of contiguous space represented by each bit in the SAT. A pack cluster size of 1 means that one 256-word block of storage is allocated for each bit set to 1. A pack cluster size of 2 means that two contiguous 256-word blocks are allocated for each bit set to 1. Allowable pack cluster sizes for an RF or an RK type disk are 1, 2, 4, 8, or 16; for an RP type disk, 2, 4, 8, or 16.

The pack cluster size affects the efficiency of storage space allocation. A large size decreases the efficiency. For example, if the size is 16, a one block file on the disk has allocated one cluster of sixteen contiguous blocks. Fifteen blocks are wasted. A fifteen block file also requires one cluster but only one block is wasted. Thus, the system manager must choose the pack cluster size which best fits the type of processing and the access requirements of the local installation.

The directory cluster size is defined for both a master file directory (MFD) and user file directory (UFD) and its minimum value is the pack cluster size. The system manager specifies the MFD cluster size when he initializes the disk; he specifies the UFD cluster size when he creates an account. A directory cluster size is equal to the pack cluster size or a power of 2 times the pack cluster size, to a maximum of 16. Thus, for a pack cluster size of 2, the directory cluster size on that device can be 2, 4, 8, or 16. For a pack cluster size of 8, a directory cluster size on that device can be 8 or 16.

The directory cluster size limits the size to which a directory can expand. A directory, whether an MFD or a UFD, expands to catalog accounts or files but can occupy only seven clusters on a disk. For an MFD on the system disk, the cluster size determines how many accounts the system can handle. The following formula gives the number of user accounts, A, for each allowable MFD cluster size, MC.

$$\frac{(217 \times MC) - 1}{2} = A$$

The minimum A is 108 accounts for an MFD cluster size of 1 and the maximum is 1735 accounts for an MFD cluster size of 16.

The UFD cluster size determines how many files a user can create under one account. The following formula gives the number of user files, UF, for each allowable UFD cluster size, UC. (The formula assumes that all files are a minimum size (seven or fewer clusters long.)

$$\frac{(217 \times UC) - 1}{3} = UF$$

The minimum UF is 72 files for a UFD cluster size of 1 and the maximum UF is 1157 for a UFD cluster size of 16.

A large directory cluster size enables the directory to catalog more files than is possible if the directory and pack cluster sizes are equal. For example, assume an MFD has a cluster size of 2 on a disk with a pack cluster size of 1. When the directory requires more storage space to catalog more files, the system allocates it a directory cluster. Since the directory cluster size is 2, the directory cluster contains two contiguous pack clusters, each containing a 256-word block. Since a directory can expand to a maximum of seven clusters, the larger directory cluster size allows the MFD to span more storage space and therefore catalog more accounts.

The file cluster size is based on the same principle as the directory cluster size and is discussed extensively under the heading CLUSTERSIZE Option in Section 9.2.2 of the BASIC-PLUS Language Manual.

The files OVR.SYS and BUFF.SYS are crucial to RSTS/E system response time. Optimum efficiency is gained when these files reside on a fast access, fixed head disk. If the system disk is not a fixed head disk, the system manager can use the REFRESH option and cause a copy of the non-resident (overlay) code to be created, placed in the file OVR.SYS, and stored on the fast access disk, if available. On systems with only a moving head disk, it is more efficient to create the file OVR.SYS and optimally position it.

DECtape processing is expedited by the use of BUFF.SYS. When a file on DECtape is opened, the directory of the DECtape is written to the file BUFF.SYS. The BUFF.SYS file requires three 256-word blocks for each DECtape drive on the system. Any updates to the DECtape directory which arise during processing cause the system to manipulate the copy in BUFF.SYS. This technique eliminates the need for continuous winding and rewinding of DECtape. The copy of the DECtape directory in BUFF.SYS is read back to the DECtape when the last file open on the DECtape unit is closed or any output file is closed. By use of the REFRESH option, the BUFF.SYS file can be placed on a fast access type disk or can be optimally positioned on the moving head system device.

It is recommended that the copy of the BASIC-PLUS error messages from the CIL (RSTS.CIL) be kept in the file ERR.SYS, which can be accessed and modified.

The RSTS/E Core Image Library resides in the file RSTS.CIL on the system disk. This file contains all executable code for system operations, resident and non-resident, and optionally contains code for the stand-alone program ROLLIN. The CRASH.SYS file can be created to retain a copy of valuable contents of memory at the time of a system crash. (See Chapters 5 and 6 for a discussion of system crashes and analysis of system crashes, respectively.)

5.4.3.1 System Account [Ø,1] on Non-System Disks - The system account [Ø,1] on non-system disks contains only a BADB.SYS file and a SATT.SYS file. The system account on non-system disks is created when the user initializes the disk.

5.5 PRIVILEGE

Certain accounts in the RSTS/E system have special capabilities outlined in this section. The accounts having special capabilities are called privileged accounts and are recognized as privileged if its project number of the project-programmer number group is a 1. The system library account [1,2], discussed in a preceding section of this chapter, is an example of a privileged account.

The system manager, operating under his own privileged account, designates accounts as privileged by assigning a project number 1 to the account when he creates it using the REACT system program. The available privileged account numbers are [1,3] through [1,254]. The assigned privileged accounts have the same special capabilities that the library privileged account [1,2] has. The user of an assigned privileged account also can create new accounts, as the system manager can. Privileged accounts, the system manager's as well as those he assigns, have the following special capabilities.

a) Unlimited file access. See Section 5.5.1.

b) Creation and modification of privileged programs. See Section 5.5.2.

c) Use of privileged aspects of system programs. See Section 5.5.3.

d) Use of privileged SYS system functions and the PEEK function. See Chapter 7.

The above listed capabilities are described in the remaining sections of this chapter and in Chapters 6 and 7. It must be emphasized that there is no fail safe and that no error messages are generated if the use of such special capabilities is to result in destruction of the system. For this reason, it is suggested that the system manager assign privileged accounts sparingly. It is recommended that the system manager create additional non-privileged accounts for himself and perform most of his functions under them. The system manager should use a privileged account only when necessary.

5.5.1 Unlimited File Access

No file in the RSTS/E system can be protected against the user of a privileged account. A privileged user can create and delete files under any account number and can access files on LOCKED disks.

Under such circumstances, no protection violation occurs. (Protection violation is a normal user recoverable error, number 10, as described in Appendix C of the BASIC-PLUS Language Manual.)

5.5.2 Creation and Modification of Privileged Programs

A program is privileged when it has a BAC filename extension and a protection code of <192> or greater. Protection code <192> means that the privileged protection <128> and the compiled file protection <64> are set. Both protection code values must be set for a program to have privileged status. The system manager or privileged user designates a program as privileged by assigning to its compiled form a protection code of <192> or greater with the NAME-AS command. For example:

```
NAME "FILE.BAC" AS "FILE.BAC <232>"
```

designates the compiled program FILE as privileged with write protection against the owner's group and all others not in the owner's group. Refer to Chapter 9 of the BASIC-PLUS Language Manual for the definition of protection codes.

If a program is designated privileged, any user can run the program in privileged program status (provided he has READ access to the file). The privileged program status exists for the duration of the program run. Privileged program status means that system operations normally reserved to a user of a privileged account can be executed while running under a non-privileged account.

If the user running a privileged program interrupts execution of the program by typing CTRL/C, the program loses its privileged status.

The ability to designate a program as privileged allows the system manager to extend use of privileged functions to non-privileged users. For example, the program TTYSET allows a user to change characteristics of his terminal. Such an action is a privileged system function executable only by owners of privileged accounts. With the privileged program status, execution of the function by the owner of a non-privileged account does not cause the normal program trap.

The same TTYSET program additionally allows a privileged user to change characteristics of a terminal other than his own. A check is built into the program to ensure that a user attempting to change the characteristics of a terminal other than his own is indeed a privileged user. As a result, the execution of privileged functions is made available to the non-privileged user but privileged features are restricted. The system manager likewise can control the use of privileged operations.

5.5.3 Use of Privileged Features of System Programs

The owner of a privileged account can execute the privileged features of system programs. Since the list of privileged features is lengthy, an entire chapter is devoted to explaining them. Certain programs, such as TTYSET, SYSTAT, and MONEY, are privileged but contain features helpful to the general user. These programs are therefore described in the RSTS-11 System User's Guide for the non-privileged user.

Two of the programs, TTYSET and MONEY, have privileged features about which the non-privileged user is not informed in the User's Guide. In Chapter 6 of this guide, only privileged features of these programs are presented. The other program, SYSTAT, is discussed in Chapter 6 of this guide in greater depth than is done in the User's Guide although it contains no explicit privileged features. Thereby, the system manager can appreciate more fully the information returned by SYSTAT and be in a position to make sound judgments concerning the system.

CHAPTER 6

PRIVILEGED FEATURES OF SYSTEM PROGRAMS

This chapter describes those features of system programs which the system manager or privileged user must understand to operate and manage the system. Many of the system programs are documented in Chapter 4 of the RSTS-11 System User's Guide for use by general users. The format of that chapter is such that the system manager, as he deems necessary, can remove a description of a system program from that guide without affecting the descriptions of other system programs. The descriptions in this chapter assume a knowledge of the programs found in Chapter 4.

The presentation in this chapter is mainly expository. Reference aids have been included by appending the relevant system program file name to the section heading and by tabulating of all relevant commands and options.

6.1 CONTROLLING SYSTEM START UP - INIT

The user can control system start up by means of the system initialization program INIT. System start up occurs when the user executes the START option or when the monitor attempts an automatic restart after a system crash. At start up time, the monitor automatically runs INIT which, in turn, executes special commands on the system. By the ability to specify these special commands, the user controls the actions which occur at system start up.

To control system start up efficiently, the system manager must understand the conditions in effect at start up time. The following conditions pertain.

a) Login attempts are prohibited (the monitor disables the login capability).

b) The monitor logically mounts only the system disk.

c) No output is made to any terminal.

d) The monitor logs the console terminal (KBØ:) onto the system under the system library account [1,2].

e) At the console terminal, the monitor executes the command equivalent to CHAIN "INIT" or CHAIN "INIT" 100.

As a consequence of condition (e), INIT runs and reads one of two control files on the system disk (SYØ:) under account [1,2]: either START.CTL as a result of a normal start up and the CHAIN "INIT" action or CRASH.CTL as a result of a system crash and the CHAIN "INIT" 100 action.

The two control files contain special commands for the initialization of the system for time sharing. The following sections describe the INIT commands and their usage.

6.1.1 INIT Program Commands

The RSTS/E system is not fully initialized until INIT runs and executes commands in the control file. By specifying the INIT commands described in Table 6-1, the system manager can control system start up according to requirements at the local installation. For example, the system possibly uses other disk devices in addition to the system disk. By means of the MOUNT command, the system manager can make

Table 6-1

Control File Contents

Command Name and Format[1]	Use
LOGINS	Allows users at both local and remote terminals to type requests to the LOGIN system program.
SEND␣xxx	Transmits the optional text xxx to all keyboards currently online except the console keyboard (KBØ:).
LOGIN␣KBn:␣[n,m]	Logs the terminal specified by KBn: onto the system using the account indicated by [n,m]. INIT automatically looks up the password. NOTE Cannot be used on KBØ: because INIT must run on that terminal.
FORCE␣KBn:␣xxx	Causes the text xxx to be placed in the input buffer of the terminal specified by KBn: and executed as if typed at that terminal. The text can be any BASIC-PLUS command or TTYSET system program SET commands. The LOGINS command must appear before the FORCE commands can be executed. NOTE If the ↑ character is the first character of the text xxx, a CTRL/C is placed in the terminal buffer ahead of the specified text xxx. However, INIT generates an error if an attempt is made to force a ↑C to KBØ.
MOUNT␣dev:id	Causes the disk unit specified by the device designator dev: and by the pack identification (id) to be logically recognized by the RSTS/E system. Additionally, the MOUNT command as used in the control file causes a clean operation (if necessary) and an unlock operation. Refer to Section 6.3.2 for information concerning mount, clean and unlock operations.
BYE or END	Causes execution of the INIT system program to be terminated. BYE causes the job running under account [1,2] to be logged out, thus freeing the console keyboard for other use and preventing unauthorized use of account [1,2]. END must be used in place of BYE when running ERRCPY since END does not log-out the job running ERRCPY under account [1,2]. See Section 6.11 for a description of ERRCPY.

[1]The notation ␣ indicates that a space character is required.

such devices in the public and in the private structure immediately available to users before they can log into the system. The local start up procedures must include making the specified devices physically ready on the proper drive units.

To execute other actions on the system, the system manager can cause INIT to execute BASIC-PLUS commands and programs. Executing the LOGINS command enables further logins on the system. The LOGIN command automatically logs a specified job onto the system at a designated terminal to allow execution of commands and programs. For example, the FORCE command can run the TTYSET system program at the terminal and can subsequently execute commands to establish terminal characteristics of certain keyboards. The SEND command prints text on all on-line terminals.

The following sample START.CTL file and accompanying explanation shows the usage of INIT commands.

```
MOUNT     DK1:PACK1                    (line a)
MOUNT     DK2:PRIV1                    (line b)
LOGINS                                 (line c)
LOGIN     KB1: [1,5]                   (line d)
FORCE     KB1: RUN $TTYSET             (line e)
FORCE     KB1: KB1:                    (line f)
FORCE     KB1: VTØ5B                   (line g)
FORCE     KB1: EXIT                    (line h)
FORCE     KB1: BYE F                   (line i)
FORCE     KBØ: RUN $ERRCPY             (line j)
SEND      RSTS/E IS NOW ON THE AIR...(line k)
END                                    (line l)
```

The user makes other disks available on the system as shown at lines a and b. Line a causes the system to recognize the additional disk cartridge PACK1 of the public structure on RK11 drive unit 1. The user must mount all public disks (except the system disk) in this manner if all user files are to be available immediately. The system also cleans (if necessary) and unlocks PACK1 so that users can create files on it.

Line b causes the same results as line a but for the disk cartridge PRIV1 in the private structure.

Logins must be enabled before attempting to log a terminal onto the system. The LOGINS command at line c is required to enable logins and to allow users access to the system.

The command at line d logs in a job at keyboard number 1 under account [1,5]. INIT automatically looks up the password of the account. In this manner, secrecy is maintained by not requiring passwords in control files.

The commands at lines e through h run the TTYSET system program to set terminal characteristics. For more information on TTYSET, see Section 6.7.

At line i, keyboard number 1 is logged off the system to prevent unauthorized use of the account.

The command at line j causes the console terminal (KBØ:) to run the ERRCPY system program under account [1,2]. This action enables the RSTS/E system to take advantage of hardware error logging on the system.

The SEND command notifies users that time sharing operations have begun. The message is printed on all terminals on line to RSTS/E.

The END command at line l terminates the INIT program running at the console terminal. However, the command leaves KBØ: logged into the system so that the system can execute the command forced into the keyboard buffer by the command at line j. Since ERRCPY detaches from the terminal, KBØ: does not remain logged into the system after initialization.

6.1.2 Creation and Usage of Control Files

The INIT system program control files START.CTL and CRASH.CTL must contain commands to properly initialize the RSTS/E system and must be stored on the system disk (unit Ø of the public structure).

The files included in the RSTS/E system generation kit are only samples and, without modification, may not execute properly on a given system. The system manager must ensure that the files include the necessary commands to initialize the local installation. For example, to replace the sample START.CTL file supplied with the RSTS/E kit, run PIP and proceed as follows.

```
RUN $PIP
PIP RSTS V5B-24  SYSTEM #880
#SYØ:START.CTL$<KB:/FA
       (User types his new INIT commands.)
↑Z
#
```

The procedure shown ensures that the new file is created on unit Ø (SYØ:). This is important since, when INIT runs, only the system disk in the public structure is mounted.

It is important that both control files perform the following functions:

a) Mount all non-system public disks,
b) enable logins,
c) set keyboard characteristics for non-ASR33 terminals, and
d) run any system programs which must execute during time sharing.

Since, at start up time, the system sets the characteristics of all terminals to those of an ASR33-type terminal, the user must run the TTYSET system program to determine the correct characteristics. For example, if keyboards 1 and 3 are LA30S-type terminals to run at 300 baud, if keyboard 2 is an ASR33-type terminal, and if keyboard 4 is a VT05B to run at 2400 baud; the system manager can specify the following sequence of commands to properly set the characteristics.

```
LOGIN KB4: [1,5]
FORCE KB4: RUN $TTYSET
FORCE KB4: KB4:
FORCE KB4: VT05B
FORCE KB4: KB1:
FORCE KB4: LA30S
```

```
FORCE KB4: KB3:
FORCE KB4: LA30S
FORCE KB4: EXIT
```

The sequence of INIT commands shows the optimum use of a keyboard on which to execute commands. Before forcing the next command to a terminal, INIT waits until the terminal output buffers are empty and the job is in the keyboard input wait state. Therefore, it is advantageous to force commands to a terminal which generates output at the highest speed since that terminal's output buffers empty faster. Note, in the example, that keyboard 4 is established as a VTØ5B whose default output speed is 2400 baud. (Refer to the table of TTYSET macro commands in Section 4.5 of the RSTS-11 System User's Guide for the default characteristics of different terminal devices.) The interface speed is the important factor since it is not necessary that a terminal actually be connected to the interface unless a visual record of the start up procedure is desired.

In the case of INIT running as a result of a system crash, it is important that commands be executed to obtain crash-dump information. To discover the cause of a system crash, the system manager can specify that INIT run the crash analysis program ANALYS. The following commands inserted in the CRASH.CTL file ensure that the analysis occurs.

```
LOGIN KB4: [1,5]
FORCE KB4: RUN $ANALYS
FORCE KB4: [0,1] CRASH.DMP
FORCE KB4:
```

For more information on ANALYS, see Section 6.9.

The system manager can specify commands to be executed on KBØ: even though INIT runs at the console terminal. The system assigns job number 1 to INIT since it is the first job to run at start up time. Because INIT runs at the console terminal, commands forced to KBØ: are not executed until INIT terminates. To prevent premature termination, INIT does not allow a CTRL/C combination to be forced to the console terminal. Therefore, the user must not specify the ↑ character in a FORCE command on the console terminal.

If the system manager wishes to run a certain job as job number 1, he can specify the proper commands to the console terminal. For example, it is often desired to run the ERRCPY system program as job 1. The following command in the control file ensures this action.

```
FORCE KBØ: RUN$ERRCPY
```

ERRCPY runs and detaches immediately after INIT terminates as the result of the END command.

6.1.2.1 START.CTL File Example - The following is an example of a START.CTL file which sets keyboards 1 and 2, starts the spooling programs SPOOL and BATCH, and runs the ERRCPY program.

```
MOUNT DK1:PRIV1
MOUNT DK2:PRIV2
LOGINS
LOGIN KB1: [1,2]
FORCE KB1: RUN $TTYSET
FORCE KB1: KB1:
FORCE KB1: VTØ5B
FORCE KB1: KB2:
FORCE KB1: LA3ØS
FORCE KB1: EXIT
FORCE KB1: RUN $QUEMAN
FORCE KB1: \DE
LOGIN KB1: [1,2]
FORCE KB1: RUN $SPOOL
FORCE KB1: Ø
LOGIN KB1: [1,2]
FORCE KB1: RUN $BATCH
FORCE KB1:
FORCE KBØ: RUN $ERRCPY
SEND RSTS/E IS NOW ON THE AIR ...
END
```

6.1.2.2 CRASH.CTL File Example - The following is an example of a CRASH.CTL file which runs the appropriate programs to recover crash information and initializes the system for time sharing.

```
MOUNT DK1:PRIV1
MOUNT DK2:PRIV2
SEND RSTS/E RECOVERING FROM A CRASH...
LOGINS
LOGIN KB1: [1,2]
FORCE KB1: RUN $TTYSET
FORCE KB1: KB1:
FORCE KB1: VTØ5B
FORCE KB1: KB2:
FORCE KB1: LA3ØS
FORCE KB1: EXIT
FORCE KB1: RUN $ANALYS
FORCE KB1:
FORCE KB1: ANALYS.TMP
FORCE KB1: RUN $ERRCRS
FORCE KB1: ERRCRS.TMP
FORCE KB1:
FORCE KB1: RUN $ERRDIS
FORCE KB1: ERRCRS.TMP
FORCE KB1: ERRDIS.TMP
FORCE KB1: ALL/S
FORCE KB1: ALL
FORCE KB1: /K
FORCE KB1: RUN $QUEMAN
FORCE KB1: \DE
LOGIN KB1: [1,2]
FORCE KB1: RUN $SPOOL
FORCE KB1: Ø
LOGIN KB1: [1,2]
FORCE KB1: RUN $BATCH
FORCE KB1:
FORCE KBØ: RUN $ERRCPY
SEND RSTS/E IS NOW ON THE AIR..
END
```

6.1.2.3 Simplified CRASH.CTL File Example - The following is an example of a CRASH.CTL file which demonstrates how INIT can run itself and therefore initialize the system for time sharing.

```
SEND RSTS/E RECOVERING FROM A CRASH...
LOGINS
FORCE KB1: SET VTØ5B
LOGIN KB1: [1,2]
FORCE KB1: RUN $ANALYS
FORCE KB1:
FORCE KB1: ANALYS.TMP
FORCE KB1: RUN $ERRCRS
FORCE KB1: ERRCRS.TMP
FORCE KB1:
FORCE KB1: RUN $ERRDIS
FORCE KB1: ERRCRS.TMP
FORCE KB1: ERRDIS.TMP
FORCE KB1: ALL/S
FORCE KB1: ALL
FORCE KB1: /K
FORCE KBØ: RUN $INIT
END
```

6.2 PERFORMING SYSTEM SHUT DOWN - SHUTUP

The shut down procedures for the RSTS/E system are critically important. If system shut down is not conducted in an orderly and careful fashion, much valuable user data can be irretrievably lost. To fully understand shut down procedures, knowledge of other RSTS/E system procedures is necessary. The system manager should familiarize himself with the concepts presented elsewhere in this chapter under the titles "On-Line System Control", "Disk Management", "Monitoring System Status" and "Spooling Operations".

To perform orderly shut down procedures, the system manager must run the SYSTAT and SHUTUP system programs. SYSTAT is described in Section 6.6. By running SYSTAT, the system manager can determine the active jobs on the system and the disk and assignable devices in use and perhaps attach any active, detached jobs to a terminal prior to shutdown. SHUTUP is run to perform the orderly shut down procedures.

The SHUTUP system program can be run only from the console terminal. The program should be stored in its compiled form in the system library with protection code <232>. The following sample dialog shows the use of SHUTUP.

```
RUN $SHUTUP
AUTOMATIC SYSTEM SHUTDOWN PROGRAM

HOW MANY MINUTES UNTIL SYSTEM SHUTDOWN? 5
HOW MANY MINUTES BETWEEN WARNING MESSAGES? 1
            (SHUTUP action messages are printed.)

PLEASE WAIT FOR THE COMPUTER TO ACTUALLY 'HALT'
WHEN IT DOES, PRESSING 'CONT' WILL BOOT BACK RSTS/E
```

When SHUTUP runs, it prints its header line, followed, on a second line, by the first of two queries. The first query asks how long a time is necessary before the system can be shut down, and the second query asks the intervals between warning messages. After the two queries are answered, the SHUTUP program proceeds with its actions. Further logins are disabled to prevent more users from entering the system. Messages are sent to all on-line terminals at the interval specified by the system manager. Each message tells how many minutes are left until the system shutdown. When no time is left, any terminal

still logged into the system is automatically logged out. Jobs still active are terminated by the KILL action. All non-system disks are then dismounted. When SHUTUP terminates, it actually halts the machine at address 54.

On systems running QUEMAN and spooling programs, the SHUTUP program includes procedures for communicating with the QUEMAN program. SHUTUP notifies QUEMAN that time sharing operations are stopping. QUEMAN, in turn, notifies the spooling programs which kill themselves when they are finished. When all spooling programs terminate, QUEMAN sends a message to SHUTUP and immediately terminates itself. Unless the user specifies otherwise, SHUTUP does not proceed with the system shut down until it receives this message from QUEMAN.

It is necessary to establish administrative procedures governing RSTS/E system operational hours. All users can be kept informed by means of the NOTICE.TXT file which prints out when a user successfully logs into the system. System shut down times are to be considered fixed so that users can plan a work load and properly complete their processing within the allotted hours of scheduled operational time.

6.3 SYSTEM UTILITY OPERATIONS - UTILTY

While running the UTILTY system program, the user has on-line system control and can perform such operations as:

a) Enable and disable logins.
b) Broadcast messages and force strings to any or all keyboards on the system.
c) Terminate execution of a job (kill) or cause a remote line to hang up.
d) Set and reset system date and time.

The system manager also can perform various disk management operations such as:

a) Cause private disk packs to be used or to be prohibited from use on specified disk drive units (mount and dismount operations).
b) Prohibit or allow creation and accessing of files on specified disk drive units (lock and unlock operations).
c) Rebuild the storage allocation table on a corrupted disk (clean operation).
d) Change the number of blocks of disk storage an account can retain at LOGOUT time (quota) and change an account password.
e) Remove all files from a user's account on a specified disk drive unit (zero operation).

The commands of the UTILTY system program are presented in Table 6-2 for reference. The following three sections explain the commands according to their functional uses: program control, on-line system control, and disk management.

6.3.1 Invoking and Terminating UTILTY

The system manager or a privileged user executes the UTILTY system program by typing the following command while logged in at any terminal.

```
RUN $UTILTY
```

The program responds by printing one header and one query line as follows:

```
'UTILTY' SYSTEM UTILITY PROGRAM
?
```

after which any valid command can be specified. A list of all valid commands is printed if the HELP command is typed. The query line need not be present in order to to type in any subsequent commands. When no commands are pending, the program prints the ? character.

Termination of the UTILTY system program can be properly accomplished by typing either the CTRL/Z combination or the EXIT command. (CTRL/Z is echoed by ↑Z being printed on the keyboard printer.) If CTRL/Z or EXIT is typed at any time, the operations currently pending are properly completed, and control is returned to the BASIC-PLUS editor. The completion of the UTILTY run is signalled by READY being printed at the keyboard.

If the CTRL/C combination is typed in order to terminate the program run, any operation in progress is interrupted immediately. Control is returned to the BASIC-PLUS editor. (CTRL/C is echoed at the keyboard printer by ↑C being printed). Typing CTRL/C to terminate a UTILTY program run is not considered proper termination since the effect of uncompleted internal system operations is unpredictable. The system manager is advised to use CTRL/Z or the EXIT command to terminate UTILTY.

6.3.2 Principles of Disk Management

Certain commands of the UTILTY system program are used to properly accomplish the management of disks on the RSTS/E system. Such commands are presented for reference in Table 6-2 under the category of disk management. To more easily convey the proper use of individual disk management commands, the following description of each command is presented with a discussion of its function and value.

6.3.2.1 Preparing a Disk for Use on a Drive - A disk pack or cartridge, to be used on the RSTS/E system, must be made known to the system. The disk is made known to the system by associating its logical name, the pack label (or identification), with the physical device unit number on which the disk resides. Such a process is called logically mounting a disk.

Logically mounting a disk associates the specific device designator and the exact pack label of the disk. A new disk must first be formatted and initialized for use on the RSTS/E system by the initialization option DSKINT. An already formatted disk can be initialized by the DSKINT system program. A disk pack or cartridge, when initialized, is assigned a unique 6-character alphanumeric name called the pack label, or id, which is required to logically mount the disk.

When the RSTS/E system is initialized at start up time, only the system disk is logically mounted. As is described in the discussion of controlling system start up in Section 6.1, the system manager must ensure that other packs in the public structure (and perhaps the private structure) are mounted by commands from the START.CTL or CRASH.CTL file. The same principle applies to private disks added to the system after system start up. The system manager can logically mount a disk on the system by means of the UTILTY system program MOUNT command or can use the CCL MOUNT command described in Section 4.14 of the RSTS-11 System User's Guide.

Table 6-2

UTILTY System Program Commands

Category	Command and Format[1]	Use
Operational Control	LOGINS	Enables users to login to the RSTS/E system.
	NO␣LOGINS	Prevents further login attempts.
	SEND␣[KBn: / ALL]␣␣xxx	Causes the text string xxx to be printed on the keyboard unit n or all keyboards.
	FORCE␣␣[KBn: / ALL]␣␣xxx	Causes the text string xxx to be forced into the input buffers of keyboard unit n or all keyboards as if it had been typed in. If the ↑ character is the first character of the string, it is replaced by a ↑C.
	KILL␣␣n	Immediately terminates user job specified by n.
	HANGUP␣␣KBn:	Disconnects the remote line specified by KBn.
	DATE␣␣dd-mon-yy	Sets the RSTS/E system date to the value of day, month and year (for example, 13-NOV-72).
	TIME␣␣hh:mm	Sets the RSTS/E 24 hour clock to the value of hours and minutes (for example, 21:52 means 9:52 p.m.).
Disk Management	MOUNT␣␣dev:id	Logically associates the disk residing on the disk drive unit specified by the device designator dev: with the pack identification label (id) so that data on the disk can be properly accessed by the system. For example, MOUNT DK1:PRIV1 associates the pack PRIV1 with the physical device designator DK1:.
	DISMOUNT␣␣dev:	Disassociates a disk pack from its physical drive specified by dev:. Must be used prior to removing the cartridge from the disk drive unit.

[1]The notation ␣␣ indicates that a space character is required.

Table 6-2 (Cont.)
UTILTY System Program Commands

Category	Command and Format[1]	Use
Disk Management (Cont.)	LOCK␣dev:	Places the disk drive unit dev: in a state which prevents files from being OPENed.
	UNLOCK␣dev:	Allows users to OPEN files on disk drive unit dev:.
	CLEAN␣dev:	Rebuilds the SAT (Storage Allocation Table) of the pack mounted and locked on disk drive unit dev:. To be used only when message DEVICE NEEDS CLEANING is printed and the device is locked.
	QUOTA␣[n,m]␣q	Sets the quantity of 256-word blocks the user account [n,m] is allowed to retain at logout time to the decimal number q. A value for q of zero means unlimited quota.
	CHANGE␣[n,m] passwd	Alters the password of user account [n,m] to the 6-character alphanumeric passwd.
	ZERO␣dev: [n,m]	Deletes all files from user account [n,m]on the disk drive unit specified by dev:.
Program Control	HELP	Prints a list of valid UTILTY commands at the keyboard printer.
	CTRL/C	Terminates execution of the current operation and the UTILTY run.
	CTRL/Z	Allows completion of pending operations before termination of the UTILTY run
	EXIT	Allows completion of pending operations before termination of the UTILTY run.

[1]The notation ␣ indicates that a space character is required.

The MOUNT command of UTILTY does not perform two other operations which the MOUNT command in the control files performs and which are necessary before the disk can be safely used. First, the SAT (Storage Allocation Table) in the SATT.SYS file on the disk must be rebuilt if necessary. (The message DEVICE NEEDS CLEANING is printed if the SAT requires rebuilding.) The operation of rebuilding the SAT is called cleaning the disk. The disk must be cleaned if it was not properly dismounted.

For instance, the SATT.SYS file can be corrupted if a user physically removes an operational disk from a drive unit without performing the proper logical dismounting of the disk. When corruption occurs, the amount of storage actually occupied on the disk as a result of file processing is not reflected in the SAT. Storage occupied is not always reflected because storage allocation tables are manipulated in memory individually and are not written back to the disk immediately.

The system manager must rebuild the corrupted SATT.SYS file by using the CLEAN command of the UTILTY system program. Before the clean operation on a disk can proceed, the disk must be in a state which ensures unavailability for normal usage (opening of files). Such a state is termed the locked state. When the disk device is in the locked state, no user except a privileged user can open files on that disk device.

It is therefore necessary, before a clean operation is to be performed, to make the disk device unavailable to users. The system manager makes the device unavailable by use of the LOCK command of the UTILTY system program. After the action of the LOCK command is completed, the system manager then initiates the clean operation by use of the CLEAN command.

Once the clean operation is completed, the system manager can safely make the disk available to users of the RSTS/E system. He accomplishes this by use of the UNLOCK command. Once a disk device is placed in the unlock state by the UNLOCK command, users are allowed to open files on the disk.

Before a disk pack or cartridge can be properly removed from a drive unit, actions similar to those of preparing the disk for use must be employed. For example, if the system manager desires to replace a private pack with another private pack, he must follow a careful procedure. (Under no circumstances should a public pack be removed from the system during normal system operation.) First, he must ensure that no files are open on the drive unit. The system manager can do this by requesting a disk status report through the SYSTAT system program. His next action would be to lock the device unit by use of the LOCK command. This action ensures that non-privileged user programs cannot open any more files on the disk.

When the disk pack to be removed from the drive unit has no OPEN files and has been LOCKed, the system manager next must disassociate that disk pack from the drive unit. He does so by logically dismounting the device with the DISMOUNT command. After the dismounting action is completed, the disk pack can safely be removed from the drive unit. Any pack which is to replace the pack removed must undergo the procedures previously described for proper use of the pack. These procedures are summarized in Table 6-3.

6.3.2.2 Removing Files from an Account - Before an account can be deleted from the RSTS/E system or deleted from a private disk, the account must contain no files. The ZERO command of UTILTY enables the system manager to remove all files from an account on a device.

6.3.2.3 Changing Quota and/or Password of an Account - Each user account in the RSTS/E system has associated with it a quota of disk storage that the account can retain at logout time and a password which allows access to the system. The quota and password are specified by the system manager when the account is created.

The system manager can change the quota by use of the QUOTA command of UTILTY. The system manager specifies, in the QUOTA command, the account number and the decimal number of 256-word blocks of disk storage he wishes the account to be able to retain a logout time. If he specifies zero for the quota, he allows the account an unlimited quota of blocks.

The system manager can change the password of an account by using the CHANGE command.

Table 6-3

Procedures for Using Disk Packs and Cartridges

Enter a Pack or Cartridge to the System[1]	Remove a Pack or Cartridge from the System
1. Use the SYSTAT system program to ensure that the drive unit is free.	1. Invoke UTILTY and use LOCK command on the drive unit containing the pack to be removed.
2. Place the pack in the drive. When the drive is READY, write enable it.	2. Use SYSTAT disk status report to determine the number of OPEN files. If zero, proceed. If non-zero, wait until all files are closed before proceeding.
3. Invoke the UTILTY system program and use MOUNT command to notify system of new pack.	3. With no files open on the device unit, use the DISMOUNT command to notify system that the pack is being removed.
4. Use CLEAN command if necessary.	4. Remove pack from disk drive unit.
5. Use UNLOCK command to free device for use.	

[1]The disk pack is assumed to have been initialized and formatted using the DSKINT initialization option. See Chapter 3 for a description of DSKINT initialization option.

6.3.3 Operational Control of the System

Certain commands of the UTILTY system program control the operation of the system. Such commands are listed for reference in Table 6-2 under the category of operational control. These commands and examples of their possible usage are described in this section.

When the system manager is on-line in the RSTS/E system, he can monitor and control system operations. By use of the SYSTAT system program, he observes the number of free small buffers. If, for example, the number of free small buffers drops below 10, system efficiency declines. He can remedy the possible degradation of system efficiency by preventing more users from logging into the system. The system manager prevents further logins by using the NO LOGINS command. The system manager also can use the NO LOGINS command in preparing to shut down time sharing operations of the RSTS/E System.

The system manager can communicate with a user at his terminal or with all users by the SEND command. The SEND command causes a specified text string to be placed in the output buffer of a terminal or all terminals and, as a result, be printed on the terminal keyboard printer. If a user assigns a peripheral device for an inordinately long time, for example, the system manager can transmit a message requesting the user to deassign the device. By specifying ALL in place of the device designator of a single keyboard, the system manager can transmit the message to each on-line terminal in the RSTS/E system.

If it becomes necessary, during the course of system operations, to handle troublesome users, the system manager has two capabilities. He can cause a user's terminal (or all users' terminals) to execute a text string by the FORCE command. He can also terminate a user's job by the KILL command. The FORCE command places a text string in the input buffer of a specified terminal as if it had been typed by the user. If the first character of the text is the up-arrow character (↑), it is replaced by a CTRL/C (↑C). The following sequence of two FORCE commands causes a user's terminal to receive two command strings, which, when executed, log the user's job out.

```
? FORCE KB4: ↑BYE
? FORCE KB4: YES
```

The KILL command terminates the user's job. The user is immediately logged off the system.

If the system manager determines that a dataset line is in use but no keyboard activity is taking place (by SYSTAT job status report), he can disconnect the dataset. The HANGUP command causes the remote line specified by KBn: to be disconnected. The hangup capability prevents a user from monopolizing the line without being charged for connect time and frees the line for other remote line users.

The DATE and TIME commands allow the system manager to set the system date and the value of the 24-hour clock, respectively.

6.4 CREATING AND DELETING USER ACCOUNTS - REACT

The system manager or a privileged user creates and deletes accounts by use of the REACT system program. The REACT system program enters user accounts on and deletes user accounts from either the system device in the public structure or individual disk devices in the private structure.

REACT is called by using the RUN command as is shown below.

```
RUN $REACT
```

REACT responds by printing the following message which requests that the user specify a function.

```
'REACT' SYSTEM ACCOUNT MANAGER
FUNCTION?
```

The three valid functions are described in Table 6-4, and explained in the following sections.

6.4.1 Creating Individual Accounts - ENTER Function

The ENTER function creates individual user accounts. When the system manager runs REACT, he invokes the ENTER function by typing E in response to the request for a function. Upon recognition of the E response, REACT prints a series of questions. A response to each question must be typed by the user before the appearance of the next question. The questions are explained in Table 6-5.

Table 6-4
REACT System Program Functions

Function	Abbreviation	Purpose
ENTER	E	To enter individual accounts on system disk or a disk in the private structure.
DELETE	D	To delete individual accounts from the system disk in the public structure or from a disk in the private structure.
STANDARD	S	To create standard user accounts on the system disk from the ACCT.SYS file at system generation time.

The following is a sample dialogue for the ENTER function.

```
RUN $REACT
'REACT' SYSTEM ACCOUNT MANAGER
FUNCTION?  E
PROJ,PROG?   100,100
DISK:PASSWORD?  DEMO
QUOTA?   500
CLUSTER SIZE?   Ø
PROJ,PROG?    ↑Z

READY
```

If the system manager enters an account on a private disk, he permits the owner of the account to create files on that disk. Prior to using REACT, the pack must be logically mounted and placed in the unlock state by means of the UTILTY system program. Refer to the discussion under the title "Disk Management" in this chapter.

If the system manager enters an account on the system disk, he permits the owner of that account access to the RSTS/E system and use of storage space within the public structure. When a new account is created, REACT also places an entry for the new account in the ACCT.SYS file.

Table 6-5

Responses to ENTER Function Queries

Question	Response Format	Meaning
PROJ,PROG?	n,m	The user account number to be entered in the MFD, where 1<=n<=254 and Ø<=m<=254.
	CTRL/Z	The user terminates the dialogue and REACT by typing the CONTROL key and Z combination simultaneously.
DISK:PASSWORD?	passwd	To enter a password for an account on the system disk MFD, where password is from 1 to 6 alphanumerics. No value for DISK need be specified since the system disk is assumed.
	dev: passwd	To enter a password to an account on a device in the private structure, where dev is the device designator and passwd is from 1 to 6 alphanumerics. For example: DK1:PASS DK1 must be logically mounted and in the unlock state by means of the UTILTY program prior to invoking REACT.
QUOTA?	n	The number of 256-word blocks of disk storage the user account is allowed to retain at LOGOUT time where Ø<=n<=65,535 and Ø means no quota is imposed upon the user's account. Therefore, a value of Ø allows the user account unlimited disk storage retention.
CLUSTER SIZE?	n	The account UFD cluster size where n is Ø,1,2,4,8, or 16. If Ø is specified, the pack cluster size is used. If non-zero the value for n must be at least the pack cluster size. Cluster sizes of 1,2, or 4 are recommended for most accounts. The maximum number of files a user is allowed to create is approximated by multiplying the UFD cluster size by 72.

6.4.2 Deleting Accounts - DELETE FUNCTION

The DELETE function removes individual user accounts from the system disk or from a disk in the private structure. As in the case of the ENTER function, a disk in the private structure must be logically mounted and in the unlock state prior to deletion of an account. In addition, before an account is deleted, the UFD of that account must contain no files. To clear the UFD of an account of files, the ZERO command of the UTILTY system program or the /ZE switch of the PIP system program must be used. See the description of the UTILTY system program in this chapter and of PIP in Chapter 4 of the RSTS-11 System User's Guide.

The DELETE function is invoked by typing D in response to the request for a function. The REACT program prints a series of questions. A response to the question must be typed by the user before the next question appears. The questions are explained in Table 6-6.

Table 6-6
Responses to DELETE Function Queries

Question	Response Format	Meaning
PROJ,PROG?	n,m	The user account number to be deleted from the MFD.
	CTRL/Z	The user terminates the dialog and REACT by typing the CONTROL key and Z key combination simultaneously.
DISK?	null	By typing the RETURN key, the system device is specified.
	dev:	The device designator of a disk in the private structure. For example: DK1: The disk must be logically mounted and in the unlock state by means of the UTILTY system program prior to invoking REACT.

The following is a sample dialog for the DELETE function.

```
RUN $REACT
'REACT' SYSTEM ACCOUNT MANAGER
FUNCTION?   D
PROJ,PROG?   100,100
DISK?   DK1:
PROJ,PROG?   ↑Z

READY
```

6.4.3 Automatic Creation of User Accounts - STANDARD Function

The STANDARD function in the REACT system program is provided to facilitate automatic creation of a large number of user accounts when the system disk is built. Explanation of the STANDARD function is presented in detail in Section 4.4 of this manual. A few ancillary remarks are made here.

The user creates the ACCT.SYS file as shown in Section 4.3.5 in the sample dialog for building the system library. The file ACCT.SYS is stored in the system library account [1,2]. ACCT.SYS is an ASCII text file, each line of which is formatted with the following: the items which would be specified by the user in response to the questions of the ENTER function and a name item. Each line of the file represents a single account to be created. The general format is as follows.

```
proj,prog,passwd,quota,cluster,name
```

The items proj, prog, passwd, quota, and cluster are described under the ENTER function. The name can contain any additional information about the account such as the owner's name. The name item is not used by REACT but is used by GRIPE. The item, name, must contain no commas, single quotes, or double quotes. The accounts [1,1] and [1,2] can appear in the ACCT.SYS file although they have been created previously during the REFRESH action of the system generation procedure. These account entries in ACCT.SYS are only used by the GRIPE system program.

6.5 PERFORMING SYSTEM ACCOUNTING OPERATIONS - MONEY

The MONEY system program enables the system manager to extract system accounting information for all accounts in the system or for selected accounts. MONEY can be run by a non-privileged user to obtain his own account information (excluding password) as described in Chapter 4 of the RSTS-11 System User's Guide.

MONEY is called by typing the following command while logged into the RSTS/E system.

```
RUN $MONEY
```

If the caller is a privileged user, a sequence of option queries is printed at the keyboard. Typing an answer to one query causes the next one to be printed. The queries and the explanation for each are given in Table 6-7. The account data given as output for each account is described in Table 6-8.

The MONEY system program can be run during normal time sharing. No conflicts arise if the system attempts to update a user's accounting information while the MONEY program is accessing it. When the RESET option is used, MONEY reads and resets to zero the user's accounting information before any system action can update the values being read and reset. Thus, no user accounting information is lost.

Some of the items output can be used to weight billing or evaluate usage. The item KCT, in effect, reflects system usage more accurately than the CPU-TIME item. For example, two users may each exhaust one minute of CPU-TIME in an accounting period. However, one user may tie up 2K words of memory while the other may occupy 6K words each time he runs. The first user's KCT value is incremented by 2 for each tenth of a second his 2K job is running. With the 6K user, each tenth of a second he runs, his KCT value is incremented by 6. The 6K user is tying up more system resources and this is reflected in his higher KCT value. Thus, a user's average job size can be gained by dividing the number of KCT's reflected for the accounting period by the number of tenths of seconds derived from the value of CPU-TIME. Referring to the example of values in Table 6-8,

Table 6-7

MONEY System Accounting Program Options

Option Query	Reply	Explanation
OUTPUT DEVICE?	dev:filename.ext	A file structured or non-file structured device can be specified. Indicating a disk file reduces processing time. For example: MONEY.DMP MONEY writes the data to the file which can later by queued for printing.
	RETURN key	Output is printed at the logged-in keyboard printer.
PRINT PASSWORDS?	NO	Typing anything except NO causes passwords to be printed.
RESET?	NO	Typing anything except YES causes the accumulated accounting data to be preserved.
	YES	Typing YES causes the following items to be reset to zero. CPU-TIME KCT's CONNECT TIME DEVICE TIME
DISK?	DKn: DPn:	Type the disk device designator with unit number n to select the accounting data from a private pack.[1]
	RETURN key	The accounting data selected is for all public disks.
SELECTIVE?	NO	Typing anything except YES causes accounting data for all accounts (on the private pack or on the system, whichever the reply to the DISK? query indicates) to be dumped.
	YES	Typing YES causes an additional query ACCOUNT?

(continued on next page)

[1] Meaningful accounting data on a private disk is account number, number of blocks occupied, disk quota, and UFD cluster size. The system updates only the system disk MFD with CPU time, kct's, connect time, and device time.

Table 6-7 (Cont.)

MONEY System Accounting Program Options

Option Query	Reply	Explanation
ACCOUNT?	n,m	The account query appears if the reply to the SELECTIVE? query is YES. Accounting data is dumped for the account specified by the project-programmer number [n,m], following which, the query is repeated.
	CTRL/C CTRL/Z	Typing CTRL/C or CTRL/Z terminates the program run.

Table 6-8

MONEY System Accounting Program Output

Header Description	Meaning	Example
ACCT	Project-programmer number (account)	100,100
PASSWORD	Account password given at login time	DEMO
CPU-TIME	Number of hours:minutes: seconds.tenths of a second of processor time the account has used since the last reset.	1:23.2 (one minute, 23.2 seconds)
KCT's	Core usage factor (kilo-core-ticks). One KCT is the usage of 1K of core for one tenth of a second.	3000
CONNECT	Number of hours and minutes (hh:mm) of terminal connect time.	2:34 (2 hours, 34 minutes)
DEVICE	Number of hours and minutes (hh:mm) of device usage time, excluding public disks.	20 (20 minutes)
DISK	Number of 256-word blocks of disk storage allocated.	100
QUOTA	Number of 256-word blocks the account is allowed to retain at logout time.	500
UFD	UFD cluster size	2

user [100,100] average job size of 3.6K is computed by dividing the number of KCT's, 3000, by the number of tenths of seconds derived from CPU-TIME, 832.

The value under the header description DISK reflects the actual number of blocks tied up in file allocation on disk. It is not the same value reported by the CATALOG system command. A file may occupy 1 block on a disk as reflected by the CATALOG command, but ties up 3 additional blocks of disk storage if the file cluster size is 4 blocks. In essence, the user is depriving the system from claiming those three contiguous blocks in the cluster although the file is not currently occupying all the space.

The information given under the header UFD is the user file directory cluster size. No other system program returns this value which the system manager specifies when the account is created by the REACT system program. It is provided in the MONEY output for information purposes only and has no accounting value.

It is advised that the system manager periodically execute the RESET option to prevent overflowing the accounting values stored on disk. The following list shows the maximum times that can be stored for each statistic without an overflow.

	On Disk	In Memory
Device Time	1092 hours	1092 hours
Connect Time	1092 hours	68 hours
CPU Time	116 hours	29 hours
KCT's	116 hours at 16K words	29 hours at 16K words

At logout time, the system updates the values on the disk with accumulated values from memory. Thus to prevent loss of accounting data, the user must log a job off the system before any of the values in memory overflow. In like manner, the system manager must execute the RESET option before the values on disk overflow. The sizes of the accounting data fields on disk allow approximately one week of continuous system operation without overflow. Therefore, MONEY must be run with the RESET option at least once per week. The MONEY program indicates reset is in effect by adding the text WITH DATA BEING RESET to the accounting printout header line.

6.6 MONITORING SYSTEM STATUS - SYSTAT

During normal time sharing operations, the system manager should monitor the status of the RSTS/E system. He gains information concerning system status either by use of the system program SYSTAT or, on systems with sufficient memory storage, by running the VT5DPY program. The options and output supplied by the SYSTAT system program are described in Chapter 4 of the RSTS-11 System User's Guide. The VT5DPY program description appears in Section 6.13. The discussion here gives the system manager guidelines on how and when to use system status information.

Several uses of SYSTAT are described elsewhere in this chapter in conjunction with other system manager operations. Those instances are listed here.

a) During preparation for system shut down, to determine active jobs and disk devices and assignable devices in use.
b) In conjunction with the UTILTY system program command HANGUP, for determining misuse of a remote line.
c) In conjunction with the UTILTY command NO LOGINS, when the number of free small buffers is less than 10.

Refer to the discussion relevant to the individual system program or system operation for more information on the above uses of SYSTAT.

Further uses of SYSTAT are listed below and discussed in the ensuing paragraphs.

d) Uncovering malfunctioning keyboards by the HUNG TTY count.
e) Guarding against a disk device filling up by watching the FREE block count.
f) Following the progress of user jobs or detached jobs by the STATE and RUN-TIME items of job status.

Item (d) refers to the HUNG TTY count reported in the SYSTAT buffer status report. A HUNG TTY count of zero is good. A HUNG TTY count of non-zero indicates the presence of a malfunctioning terminal or terminals. The ERRDIS program can identify the device or devices causing the error count. If the HUNG TTY count increases rapidly, a field hardware specialist should be consulted.

The FREE block count mentioned in item (e) reflects the apparent number of free blocks on each disk and is given in the disk status report of SYSTAT. For practical purposes, however, such as for allocating a file on the device, all of the free blocks reported by SYSTAT may not be usable. A NO ROOM FOR USER ON DEVICE message may be generated although SYSTAT reports that enough FREE blocks exist. The file cluster size or the number of clusters required can prevent a file from fitting on the device desired. For example, a file whose cluster size is 16 and whose length is 10 blocks possibly does not fit on a device which SYSTAT reports to have 50 free blocks of file space remaining. The cluster size of 16 demands that 16 contiguous blocks of free space must exist on the device before the file can be allocated to the device. In some cases, 16 contiguous blocks simply do not exist on a device. (It must be pointed out that RSTS/E does not allow a file to extend to another physical device.)

A further condition exists for showing NO ROOM FOR USER on a device. The UFD is perhaps full and cannot accommodate the creation of another file. The UFD cluster size was not made large enough when the account was created with REACT.

The occurrence of jobs being stalled in a resource sharing system is detectable by the means presented in item (f). If the system manager notices that a RUN-TIME value of a job is not increasing (the value is printed out in a job status report), it indicates that the job is stalled, waiting for an I/O device. One user job in the system can ASSIGN a device or keep an assignable device locked by having one file open on it. The system manager can determine the selfish user job by examining the device status report which associates the busy device with the job number of the user controlling that device. The system manager can request that the user free the device or, if that is not viable, can use UTILTY commands to force the job off the system.

The status of detached jobs is of interest also. If a detached job is reported by SYSTAT to be in the HB state (hibernate), it is never eligible for run time. The HB state indicates that the detached job generated an error or has completed execution. The problem of a detached job in the HB state is handled by logging into the system at a free terminal, by using the attach capability of LOGIN and attaching the job to a terminal. Once the detached job is attached to a terminal, messages can be printed.

6.7 DETERMINING TERMINAL AND REMOTE LINE CHARACTERISTICS - TTYSET

The RSTS/E system operates with a variety of terminals. During system generation, the system manager specifies the numbers and types of terminal interfaces as part of the hardware configuration. The types of terminals are intentionally not specified. Since many terminals can operate in Teletype mode at a speed of 110 baud, the system automatically sets the default characteristics of all line interfaces as those of the ASR-33 device. The system manager or privileged user must set the characteristics of other types of terminals.

The TTYSET system program sets terminal characteristics. As described in Section 4.5 of the RSTS-11 System User's Guide, a user can set the characteristics of his own terminal by TTYSET commands. The system manager or privileged user can set the characteristics of terminals other than his own by the privileged TTYSET KBN: command. The standard method of setting characteristics of local lines is by the TTYSET commands in the START.CTL file. This procedure sets up all local lines automatically at the start of each time sharing session.

Two methods exist to set characteristics of remote lines. First, all remote lines can have the default characteristics of the ASR-33 device. Consequently a remote user must log into the system at a speed of 110 baud and then run TTYSET to set the characteristics of his terminal. If the remote line is connected to a DC11 single line interface or to a DH11 multiplexor line, the user can type the TTYSET SPEED command to change the baud rate. The terminal characteristics revert to those of the ASR-33 device when the user hangs up the line.

The second method for setting characteristics of remote lines involves setting the so called ring characteristics. The system manager or privileged user can run TTYSET to set the ring characteristics which cause the system to automatically use the ring characteristics every time it answers a call on a particular remote line. By setting ring characteristics for a remote line, the system manager can establish certain lines for alphanumeric display terminals running at 300 baud, for 2741 communications terminals, or for other types of terminals. The ring characteristics remain in effect for the current time sharing session unless changed again by TTYSET. TTYSET commands in the START.CTL file can also establish ring characteristics.

6.7.1 Establishing the Terminal Speed Characteristics File - TTYSET.SPD

The TTYSET system program has a command which sets the baud rate of a variable speed line so that a variable speed terminal device can be enabled to operate at any of its legal speeds. The command is the SPEED command. The proper functioning of the SPEED command requires that the system manager create a file called TTYSET.SPD and store it under the system library account [1,2]. The TTYSET.SPD file contains entries in ASCII format which specify the values for the speeds (baud rates) that a given terminal line can handle. Currently, the DC11 remote line interface and the DH11 multiplexer interface are the only RSTS/E supported devices which support programmable baud rates.

The system manager creates the TTYSET.SPD file under the system library account [1,2] by running PIP and specifying values as the following sample format demonstrates.

```
RUN $PIP
PIP - RSTS VØ5B-24 SYSTEM #219
#SYØ:$TTYSET.SPD<KB:/FA
11,  0,50, 75,110,-1,150,200,300,600,1200,1800,2400,4800,9600,-1,-1
12,11Ø,-1,15Ø,3ØØ,-1, -1, -1, -1, -1,  -1,  -1,  -1,  -1,-1,-1
13,  0,-1, -1,110,-1,150, -1,300,600,1200,  -1,2400,  -1,  -1,-1,-1
↑Z
#↑Z

READY
```

Each line of the file TTYSET.SPD contains 17 entries separated by commas. The first entry is the keyboard number of the line interface which allows programmable baud rates. Each of the remaining 16 entries in the line represents the baud rate corresponding to a valid speed setting of Ø through 15. Speed settings that are not used must have a value of -1.

The conventional baud rates are shown in the sample format above at the line beginning with keyboard line number 11. The sixth, sixteenth, and seventeenth entries in the line are not used on RSTS/E and must contain -1. The entries are, on the DH11 multiplexer, 134.5 baud and two externally controlled baud rates.[1] If a baud rate is

---

[1] Although the 134.5 entry must not appear in the TTYSET.SPD file, the speed 134.5 is used for 2741 terminals. When the user types the macro command 2741, the program sets speed 4 (134.5) on the DH11 line or speed 2 on the DC11 line but does not check the entry in the TTYSET.SPD file.

allowed on a keyboard line, the value for that speed must appear for the speed setting established to represent that baud rate on the related line interface device.

Take, for demonstration purposes, the line beginning with keyboard line number 12 in the same format above. The entry for KB12 represents values that would be entered for a typical DC11 remote line interface which is established to handle the valid, variable speeds of a VTØ5A display terminal or LA3ØS serial DECwriter. The DC11 interface supports four different speeds. The four entries in the line after the entry for the keyboard line number are used to define the speeds allowed on that DC11 line. The VTØ5A terminal and LA3ØS DECwriter can handle speeds of 110, 150, and 300 baud. The entries for those speeds are made in the relative positions of the line for KB12. The entry for 134.5 (between 110 and 150) is not usable on RSTS/E[1] and must contain a -1. All other entries in a line for a DC11 must contain a -1.

The line beginning with keyboard line number 13 in the sample format above represents values that would be entered for a DH11 multiplexer interface which is established to handle the valid speeds of a VTØ5B display terminal. The DH11 interface supports up to 16 different speeds including Ø baud. (A speed of zero effectively turns the line off.) The VTØ5B terminal can handle speeds of 110, 150, 300, 600, 1200, and 2400 baud; Ø baud and split speeds are also allowable. The entries for those speeds are made in the proper positions of the line for KB13. For those speeds not supported by or not allowed for the VTØ5B terminal on the KB13 line, the user must indicate by making a -1 entry.

---

1 Although the 134.5 entry must not appear in the TTYSET.SPD file,the speed 134.5 is used for 2741 terminals. When the user types the macro command 2741, the program sets speed 4 (134.5) on the DH11 line or speed 2 on the DC11 line but does not check the entry in the TTYSET.SPD file

The system manager ensures that the file TTYSET.SPD is stored in the system library account [1,2] on the system disk when he types the SYØ: device specification and the dollar sign character ($) preceding the filename in the PIP command string. The CTRL/Z combination (↑Z) typed on the line following the text for KB13 terminates the entry of text to the ASCII file TTYSET.SPD and returns control to PIP, signalled by the # character being printed. The CTRL/Z combination typed on the line with the # character terminates the PIP run and returns control to the RSTS/E Monitor. It is recommended that, after creating the TTYSET.SPD file, the system manager run the TTYSET program and execute the SPEED command for each line.

6.7.2 TTYSET Privileged Feature - KBn: Command

The system manager sets the characteristics of other terminals in the RSTS/E system by use of the KBn: command. The following example demonstrates the use of the KBn: command. The system manager, while logged into the system under his privileged account, calls the TTYSET system program as follows.

```
RUN $TTYSET
'TTYSET' TERMINAL CHARACTERISTICS PROGRAM
?
```

The program responds with a header line and a question mark character (?), which indicates that the program is ready to accept commands. If the system manager wishes to set the characteristics of a VTØ5A terminal at keyboard 3, he types the following commands.

```
?KB3:
FOR KB3:? VTØ5
FOR KB3:?
```

In response to the system manager typing the KB3: command, TTYSET prints the FOR KB3: prompting message. This message indicates that commands typed apply to keyboard number 3. The VT05 command immediately sets the characteristics of the line to those of a VT05A alphanumeric display terminal. TTYSET prints the prompting message again. The user can type another KBn: command to set characteristics of another keyboard.

The system manager can also change the individual characteristics of a local terminal. If he desires to limit the line length of the terminal at keyboard unit 4, he types the following commands.

```
FOR KB3:? KB4:
FOR KB4:? WIDTH 60
FOR KB4:? EXIT

READY
```

As a result of the execution of the above commands, whenever 60 characters are printed on a line of the terminal at keyboard unit 4, a carriage return and line-feed operation is performed. The EXIT command terminates TTYSET.

6.7.3 Automatic Setting of Local Terminal Characteristics

The setting of terminal characteristics in the RSTS/E system applies only to the current time sharing session. This condition allows for replacement of an ASR-33 type terminal with any one of the authorized terminals without having to change the system configuration and, thus, rebuild the system. Since it is quite bothersome for the system manager or a user to set characteristics of local terminals each time the system is initialized, the INIT system program can automatically set local terminal characteristics by commands in the START.CTL and CRASH.CTL files. Refer to the sample START.CTL and CRASH.CTL files in Section 6.1.

6.7.4 Setting Terminal Characteristics of Remote Lines - /RING

A remote line user whose terminal is other than an ASR-33 type terminal must set the characteristics of his terminal each time he logs into the system if he wants to ensure recognition of the characteristics of that terminal by the RSTS/E system. The characteristics remain set until either the telephone is hung up, the line is disconnected or the system shuts down or restarts.

The /RING option relieves the user from the necessity of setting the characteristics each time he logs into the system on a certain remote line. For example, to set the characteristics of the remote line on keyboard 14 for the current time sharing session, the system manager can run TTYSET under a privileged account and type commands as follows.

```
?KB14:/RING
FOR KB14:(RING)? VTØ5B
FOR KB14:(RING)?
```

When the system manager indicates the /RING option with the KBn: command, TTYSET prints the prompting message FOR KBn:(RING) after which any TTYSET command can be typed. The command takes effect immediately and TTYSET prints the prompting message again. If the user is not logged into the system under a privileged account or the characteristics of the line conflict with the command, TTYSET prints an error message and the prompting message. If TTYSET is successful, the characteristics for the remote line are set for the duration of the current time sharing session. Each time a user dials the particular line, the system uses the characteristics for that terminal.

One caution is in order. The DL11E-type interface and the individual local interfaces (KL11, DL11-A through DL11-E, and LC11) do not have programmable baud rates. Therefore, the user must not execute commands to change baud rates on a keyboard line having any of those interfaces.

6.8 PROCESSING USER COMMENTS - GRIPE

The RSTS/E system includes a program to allow users of the system to communicate comments to the system manager. Comments are entered, under the control of the GRIPE system program, to a common file named GRIPE.TXT. The file, GRIPE.TXT, which retains the user comments for inspection by the system manager, is created, expanded, and deleted on an as-needed basis under the system library account [1,2]. As an aid in identifying the user who entered the comment, the GRIPE system program uses a name item supplied in the individual user's account information in the ACCT.SYS file, also stored in the system library account [1,2]. However, the name item and entry for the account in the ACCT.SYS file are not required for GRIPE to run. See Section 6.4.3 for the ACCT.SYS file format.

The system manager or a privileged user invokes the GRIPE system program in the same manner as the general user. (GRIPE is described in Chapter 4 of the RSTS-11 System User's Guide.) Once GRIPE prints its query line, the system manager can then examine the contents of GRIPE.TXT or can clear the contents of GRIPE.TXT.

The *LIST command is used in the following manner to examine the contents of the GRIPE.TXT file.

```
RUN $GRIPE
YES?  (END WITH ESCAPE)
*LIST$ OUTPUT? LP:

READY
```

The system manager types *LIST and the ESCAPE or ALT MODE key immediately after the query line. (Typing the ESCAPE or ALT MODE key on a separate line causes *LIST to be entered as text into the GRIPE.TXT file.) If the GRIPE.TXT file is empty, the message NO GRIPES FOUND is printed, followed by the READY message. Otherwise, the GRIPE program requests an output device on which to list the contents of the GRIPE.TXT file by printing the OUTPUT query. The system manager can type the RETURN key to have the comments listed at the keyboard printer or can type a device designator, such as LP: shown in the example above. The output for each user comment in the GRIPE.TXT file consists of an identification line (including the account entering the comment, the date and time it was entered, and an account name taken

from the ACCT.SYS file) and the text of the comment. The program run is automatically terminated upon completion of the output. Control is returned to BASIC-PLUS. This action is signalled by printing of the READY message.

The system manager clears the contents of the GRIPE.TXT file by using the *RESET command after invoking GRIPE. The following example demonstrates the use of *RESET.

```
RUN $GRIPE
YES?  (END WITH ESCAPE)
*RESET$

READY
```

The system manager must type *RESET immediately followed by the ESCAPE or ALT MODE key. The clearing of the GRIPE.TXT file and termination of the GRIPE run is signalled by the READY message.

6.9 ANALYZING SYSTEM CRASHES - ANALYS

When a system crash occurs in RSTS/E, time sharing operations are halted. If the required conditions described in Chapter 5 are met, the critical contents of memory are written into the CRASH.SYS file in the system account [Ø,1] and the system disk is bootstrapped in automatic restart mode.

The occurrence of a system crash subsequent to an initial one causes the CRASH.SYS file to be overwritten. Therefore, the system manager is provided with a means of retaining information in the CRASH.SYS file. This means is the ANALYS system program. The use of the ANALYS system program to document system crashes requires that the CRASH.SYS file be created at REFRESH time and that the crash dump feature be enabled at system start up time.

The ANALYS system program is invoked by the following command:

```
RUN $ANALYS
```

In response, a header line and two successive query lines are printed as follows:

```
'ANALYS' CRASH DUMP ANALYZER

INPUT?
OUTPUT?   CRASH.DMP
```

The first query line requests a filename of the file to be analyzed, which by default is CRASH.SYS in account [Ø,1]. The user need only type the RETURN key, after which the second query line is printed. The second query line requests a disk file or a device designator for the output medium, which, for example purposes, is CRASH.DMP on the system disk in the sample dialogue. Normally, ANALYS takes about 5 minutes to run. Upon completion of the output, program execution is automatically terminated and READY is printed at the keyboard printer.

It is useful for the system manager to cause the ANALYS system program to run automatically. This automatic invocation of ANALYS is accomplished by placing the proper commands in the CRASH.CTL file as FORCE commands. As a consequence of a system crash, the INIT

system program, executing the FORCE commands in the CRASH.CTL file, causes a job to run ANALYS. The method of accomplishing the automatic invocation of ANALYS is described in Section 6.1.

The output of an ANALYS system program run supplies valuable hardware and software information which can be used by a software specialist to determine possible causes of system crashes.

6.10 SPOOLING OPERATIONS - QUEMAN, SPOOL, AND BATCH

Spooling operations on RSTS/E depend upon interjob communication between a spooling program and a queue management program QUEMAN. Possible spooling programs are SPOOL, BATCH and RJ2780. SPOOL executes requests for a line printer unit and facilitates more efficient use of the device. BATCH executes requests on a pseudo keyboard device and provides non-attended job operations. RJ2780 is optional on all systems and executes requests for communications between two computer systems. QUEMAN manages all requests for spooling programs and ensures that requests are handled properly.

Interjob communication is attained by QUEMAN sending and receiving messages with the send/receive SYS system function call to FIP. A user job must be privileged to run QUEMAN since it declares itself a receiving job on the system. Spooling programs on the system also employ the send/receive message facility to communicate with the common receiving job QUEMAN.

QUEMAN creates a common file QUEUE.SYS on account [1,2] on the system disk. The file accommodates up to 254 queued requests for spooling programs. Each request handles a maximum of 14 files. Each job request queued with the /AFTER option occupies two request slots in QUEUE.SYS.

The QUEMAN program must have write access to the QUEUE.SYS file since it creates requests for spooling programs and updates control information in the file. A spooling program receives requests by notifying QUEMAN that it is ready to process. QUEMAN, in turn, accesses the queue file to determine if any requests exist for that spooling job. If no queued requests exists, the spooling program performs an indefinite sleep operation until awakened by QUEMAN with a request to process. In this manner, one program alone is responsible for maintaining the queue file.

A user creates a request for a spooling program by running the QUE program. When QUE executes a command to queue a request, it sends messages to the QUEMAN program rather than directly writing the request to QUEUE.SYS. However, when a user requests to list current requests, QUE reads the information directly from QUEUE.SYS. This action possibly confuses a user if he tries to list a queue soon after he types

a request. Often QUEMAN is in the process of parsing the new request when QUE accesses QUEUE.SYS to list pending requests. For more information on the QUE program, see Section 4.11 of the RSTS-11 System User's Guide.

QUEMAN enters requests in the QUEUE.SYS file based on a priority number between 1 and 255 specified in the QUE command. QUEMAN inserts a request with a given priority number ahead of those with lower priority numbers. Routine requests are assigned a priority of 128. A privileged user can assign a priority higher than 128 and thus can force QUEMAN to process the request before others of lower priority. All users can assign a priority of less than 128 and can thus force QUEMAN to process the request only after all others of higher priority are processed. In this manner, requests of a non-critical nature do not slow up the processing of routine requests.

Whenever a spooling program encounters an error, it interrupts processing and informs QUEMAN of the error. QUEMAN generates a request for user action and broadcasts the appropriate identifying information on the system console terminal. A user on the system handles the request by removing the error condition and typing a reply to QUEMAN. QUEMAN, in turn, clears the request from its internal tables and sends the reply to the spooling program. The spooling program, upon receipt of the reply, continues processing based on the user's request.

For a spooling program servicing a line printer, certain errors are possibly self-correcting. This situation arises because the device handler for the line printer tests the ready status of an off line unit every 10 seconds. If the user removes the cause of the error (for example, fixes a paper jam) and puts the line printer on line again, the line printer software detects the ready status and automatically continues output. As a result, the spooling program detects that the error condition is cleared and informs QUEMAN. Finally, QUEMAN removes the request for user action since the spooling program has continued processing the queued request.

Because of the highly interdependent nature of the queue management and spooling mechanism, it is recommended that the user not terminate any of the programs involved. Such termination possibly results in loss of data or destruction of the queue file. The user can leave terminating of spooling operations to the system shutdown procedures.

The SHUTUP system program executes shutdown procedures which preserve file integrity. SHUTUP sends two messages to QUEMAN. The first message initiates QUEMAN's off-line procedure which entails informing each spooling program to complete processing the correct file. Thereafter, a spooling program closes all files, tells QUEMAN it is off-line, and kills itself. After sending the second message, SHUTUP waits for a reply from QUEMAN. After receiving the reply, SHUTUP waits 30 seconds before killing any jobs and thus gives QUEMAN time to kill itself. If SHUTUP does not receive a reply from QUEMAN within 30 seconds, it prints a message at the console terminal and does not proceed unless the user types an indicated response.

6.10.1 Operating the Queue Manager - QUEMAN

To run QUEMAN, the job must be privileged and the user must type the following command while at BASIC-PLUS command level.

```
READY

RUN $QUEMAN
QUEMAN VØ5B-Ø5 - RSTS VØ5B-24
STARTED AT 11:31 ON 29-APRIL-74
*
```

QUEMAN runs and prints its header line containing program and system names and version numbers and, subsequently, an accounting message giving the time of day and current system date.

QUEMAN opens the system file QUEUE.SYS. If the file does not exist, QUEMAN creates and initializes it. QUEMAN notifies the user of this action by printing the following message.

```
NO QUEUE FILE FOUND -- WILL INITIALIZE
```

If the file exists and QUEMAN has write access to the file, it prints the * character indicating its readiness to accept commands. Another program having write access to the file causes QUEMAN to print the message QUEUE FILE OPENED BY ANOTHER PROGRAM ALREADY... CAN'T RUN STOP AT LINE 3Ø. At this point, the user must determine which job has the file $QUEUE.SYS open and must terminate that job. By typing the CONT system command in response to the STOP AT LINE 3Ø message, the user causes QUEMAN to retry the open operation.

To cause QUEMAN to DETACH, type \DE in response to the * character. For example,

```
* \DE
DETACHING...
```

As a result, QUEMAN prints the DETACHING message and detaches itself from the keyboard.

NOTE

> On ASR33 type terminals, the backslash character is generated by typing the SHIFT key and the L key simultaneously. The system echoes the SHIFT/L combination by printing the \ character.

In general, QUEMAN should be run in the DETACHed state, except when some actual interaction with it is required.

To execute QUEMAN commands, the user must attach the QUEMAN program to the terminal and type the particular command. For example, to terminate the QUEMAN program, perform the following actions as shown in the sample dialog.

```
ATTACH 5
PASSWORD:
ATTACHING TO JOB 5
'QUEMAN' ATTACHED
TYPE \DE TO DETACH
* \EX
READY
```

The ATTACH command described in Section 4.1 of the RSTS-11 System User's Guide attaches the job to the terminal. The QUEMAN program prints the 'QUEMAN' ATTACHED message to indicate that it is attached to the terminal.

NOTE

At no time should the user type the CTRL/C combination. Such action possibly destroys the QUEUE.SYS file and causes the program to print the QUEUE FILE ENDANGERED message and to terminate execution.

The asterisk character (*) indicates that QUEMAN is ready to execute a command. To terminate QUEMAN, type \EX. Subsequently, the program closes its files and exits to the system monitor as indicated by the READY message.

Other commands recognized by QUEMAN are listed and described in Table 6-9. A command is preceded by a backslash character to distinguish it from other possible responses.

Whenever QUEMAN requires help from the user, it performs a standard procedure. For example, if a spooler encounters a hung line printer, it sends a message to the QUEMAN program. QUEMAN broadcasts the standard text on the system console keyboard in the following format:

```
****QUEMAN (j) m:n REQUEST:  text ?
```

where:

- j — is the job number under which QUEMAN runs.
- m — is the message identification number between 0 and 255 which QUEMAN relates to a request for user action.
- n — is the logical name of the requesting program. Examples of names are LPnSPL, BATCH, or RJ2780. The character n in LPnSPL denotes the line printer unit to which the SPOOL program directs output.
- text — indicates QUEMAN must process a request from a spooling program. The text between the colon and the ? characters is the action message to which the user must respond.

QUEMAN performs no further processing for the designated spooling job unless the user attaches QUEMAN to the terminal and responds to the request or unless QUEMAN deletes the message for the spooling job.

Table 6-9

QUEMAN Commands

Command	Meaning
\IN	Initialize the QUEUE.SYS file. All pending jobs for all devices are removed and the protection is set to <40>.
\EX	Immediately close all files, remove QUEMAN from system receiver table and reset the priority. It is recommended that the user terminate all spooling programs before typing the \EX command. See Section 6.10.4 for terminating instructions.
\RE:n	Remove the spooling job indicated by the job number n from QUEMAN internal tables. The user can determine the proper n for a spooling program by using the \ST command.
\ME	Print at the terminal any pending messages from spooling jobs.[1]
\ST	Print at the terminal the status of spooled jobs currently running on the system.
\DE	Detach QUEMAN from the terminal.

[1]The QUEMAN program does not ordinarily print messages when it is attached to a terminal. When attached to a terminal, QUEMAN prints messages under the following circumstances.

a. If ten messages have been received since the job was attached,

b. If the user attempts to detach, or

c. If the user types the \ME command.

To respond to a request from QUEMAN, the user must type a line containing the identification number of the request being serviced and the command to the indicated spooling job. For example:

```
****QUEMAN  (05)  1:LPOSPL REQUEST:LPO HUNG?
ATTACH 5
PASSWORD:
ATTACHING TO JOB 5
'QUEMAN' ATTACHED
TYPE \DE TO DETACH
*1 CO

*
```

The request LP0 HUNG is an error message generated by the spooling program SPOOL running for line printer unit 0 (LP0:). The request identification 1 tells the user that this is the first active request for user action. The user can determine the cause of the error condition and remedy it. He then attaches the QUEMAN job to the terminal and types a response to the asterisk character printed after QUEMAN attaches. The user types the identification number of the request to which he is responding (1) and the proper command (CO) to the spooling program. QUEMAN prints the asterisk again. The user can wait to determine if the response to the request is sufficient to correct the error.

If a line printer error condition is removed before the user types a response to the request, the system automatically resumes printing according to the option the requester specified when he queued the file. The spooling program SPOOL notifies QUEMAN, which deletes the request. QUEMAN, in turn, notifies the user by printing a message. The following sample printout shows the action.

```
****QUEMAN (Ø5) 2:LPØSPL REQUEST: LPØ:HUNG?
REQUEST #2 DELETED by LPØSPL
```

(For more information regarding SPOOL, see Section 6.10.2.)

QUEMAN handles a response to an action request by passing the text typed to the spooling program. The spooling program runs and processes the text. In the example shown, SPOOL executes the CO command of the SPOOL program. If the user's action properly remedied the hung condition of line printer unit 0, the CO command causes SPOOL to perform the defined operation. When the command or the user's action is not sufficient to continue processing, the spooling program generates another request message for the QUEMAN program.

If the user types an invalid message number, QUEMAN prints text similar to the following sample.

```
MESSAGE #n NOT OUTSTANDING
*
```

The user is allowed to type a command or another response to the action request.

QUEMAN prints error messages when the user types an invalid command or response to the * character. These messages are listed and described in Table 6-10.

The QUEMAN program monitors the activity of spooling programs and broadcasts information messages to the system console terminal. For example, if the user runs SPOOL, QUEMAN broadcasts a message similar to the following.

```
QUEMAN MESSAGE:  LP1SPL (4) PUT ONLINE AT Ø4:23 PM
```

This message informs the user that a spooling program started execution on line printer unit 1 with job number 4.

When the user types the CTRL/C combination to a SPOOL program, QUEMAN typically prints the following text.

```
LP1SPL MESSAGE:  ↑C TO SPOOLER
QUEMAN MESSAGE:  LP1SPL (4) REQUESTED OFF-LINE -- TAKEN
                 OFF-LINE AT Ø4:22 PM
```

Table 6-10

QUEMAN Error Messages

Message	Meaning
INVALID RESPONSE -- x	The response x typed by user was not one of the following valid commands: \IN, \EX, \RE:n, or \ST.
JOB n NOT ON-LINE	An attempt was made to release a job by the \RE:n command and that job is not running under QUEMAN control.
MESSAGE #n NOT OUTSTANDING	The user typed an identification number indicated by n. QUEMAN determines the number requires no user action response, and continues operation.
QUEUE FILE ENDANGERED	QUEMAN detects a CTRL/C and attempts to exit without destroying the QUEUE.SYS file.
SPOOLERS ON-LINE -- CAN'T INIT	User types the \IN command, not executable by QUEMAN since spooling programs are currently running.
BAD JOB NUMBER	User typed the \RE:n command and n is not a valid number. For example, \RE:A.

The first line records that the spooling program for LP1: detected a CTRL/C combination. The second line informs the user that QUEMAN removed the job from the QUEUE.SYS file at a specific time.

The BATCH program sends messages to QUEMAN which, in turn, broadcasts them to the system console terminal. In such a manner, BATCH maintains a log of the jobs it executes. For example, QUEMAN prints such a sequence of messages from BATCH.

```
BATCH MESSAGE:  COMPAR: STARTED AT 03:02:22 AM
      .
      .
      .
BATCH MESSAGE:  COMPAR: COMPLETED AT 03:08:09 PM
```

The text BATCH MESSAGE identifies the information as coming from the BATCH program. The text COMPAR identifies the BATCH job name currently being executed. The remaining portion of each message tells the time in hours, minutes, and seconds for the start and completion of the job. For more information on BATCH operation, see Section Section 6.10.3.

If any messages occur while QUEMAN is attached to a terminal, the program saves the QUEMAN job messages until the user types either the \ME command or the \DE command. In this manner, the user can type responses to requests without interruption from QUEMAN. Additionally, if new messages are pending and the user types the \DE command, QUEMAN prints the messages and gives the user a chance to respond to an action request before it detaches itself. For example,

```
* \DE
ADDITIONAL MESSAGES:
QUEMAN MESSAGE: LP1SPL (4) RELEASED -- TAKEN OFF-LINE AT 05:17 PM
TYPE \DE TO DETACH
*
```

The user types the \DE command to QUEMAN when new messages are pending. QUEMAN prints any additional messages but not all pending messages and prints TYPE \DE TO DETACH followed by the * character. The user can then type a response to the action request or can type a QUEMAN command.

6.10.2 Line Printer Spooling Program - SPOOL

The line printer spooling program runs without user intervention and transfers files from disk, DECtape or magtape to a line printer. To run SPOOL, the user job is logged into the system under a privileged account. To start SPOOL, type the following command.

```
RUN $SPOOL
```

SPOOL runs and checks that the project number of the account is 1. If the account is not privileged, the program prints the PROTECTION VIOLATION message and terminates. Otherwise, the program prints two lines. The first line tells the program and system names and version numbers and the second line requests the unit number of the printer. For example,

```
SPOOL VØ5B-9 - RSTS VØ5B-24 SYSTEM #880
LP UNIT#?
```

SPOOL checks that the unit number entered is between Ø and 7. If the number is invalid, SPOOL prints the query again.

With the line printer unit number, the user can specify the following options.

Option	Meaning
/AS	Reserve the line printer unit to the job as if the user had typed the ASSIGN LPn: command.
/LN:nnn	Set the default form length to nnn which can be between 1 and 127. This value is used in the line printer mode option.
/WI:nn	Set the line printer width to nn to properly adjust the width of the job header pages. This is useful when 80 column paper is used in a 132-column unit. The option does not prevent the program from sending lines longer than nn to the printer. A line longer than 80 characters prints off the page if the printer is configured for 132 columns but 80 column paper is used.

For example, to set the line printer width to 80 columns and assign line printer unit 1 when SPOOL starts, type the following.

```
LP UNIT #?  1/WI:80/AS
```

After entering a valid response to the UNIT # query, SPOOL opens the character generation file CHARS.QUE on the system library account. If the file does not exist, the program prints the message CHARS.BAS HAS NOT BEEN RUN -- CAN'T RUN and terminates. The user must run the CHARS.BAS program described in Appendix E.

The program next determines the width of the printer by inspecting a parameter in the device data block.

SPOOL next communicates with the QUEMAN program which initializes entries in the QUEUE.SYS file for the related device. If the queue manager is not running, SPOOL prints the following message:

```
QUEMAN NOT RUNNING -- CAN'T RUN
```

SPOOL terminates. The user must run QUEMAN as described in Section 6.10.1 and run SPOOL again. If a spooling program is currently running for that device, the system generates an error (ERR=18) and the program prints the following message.

```
ILLEGAL SYS ( ) USAGE AT LINE 21010 -- SPOOLER WILL HALT
```

The program attempted to declare the job as an eligible receiver and determined that SPOOL is currently active for that device. As a result, it terminates. Otherwise, SPOOL subsequently prints the following message:

```
DETACHING...
```

The program then enables CTRL/C trapping and detaches itself from the terminal.

To properly terminate SPOOL, the user attaches QUEMAN to the terminal and types the \RE command with the job number of SPOOL. If the user does not know the proper job number, he can type the \ST command to gain a status report. A typical procedure is shown in the following dialog.

```
ATTACH 2
PASSWORD:
ATTACHING TO JOB 2
'QUEMAN' ATTACHED
TYPE \DE TO DETACH
* \ST
 2 SPOOLERS ON-LINE
 3              LPØSPL  LPØ  2  ØØØØ
 4              LP1SPL  LP1  2  ØØØØ
* \RE:4
* \DE
ADDITIONAL MESSAGES:
QUEMAN MESSAGE:  LP1SPL (4) RELEASED   AT Ø5:17 PM
TYPE \DE TO DETACH
* \DE
DETACHING ...
```

Before detaching, QUEMAN prints an information message concerning the LP1SPL job. For more information, see the description of terminating spooling programs in Section 6.10.4.

The user can terminate SPOOL by attaching it to a terminal and typing the CTRL/C combination. This method is not recommended since the SPOOL program possibly is printing a job or QUEMAN is currently sending a print request to the SPOOL program. If the user does terminate SPOOL by typing the CTRL/C combination, SPOOL kills the job under which it is running. This action leaves the terminal logged off the system. QUEMAN subsequently prints the following messages at the system console terminal.

```
LP1SPL MESSAGE:  ↑C TO SPOOLER
QUEMAN MESSAGE:  LP1SPL (4) REQUESTED OFF-LINE--TAKEN
                 OFF-LINE AT 04:22 PM
```

The first message informs the user that the LP1SPL job detected the CTRL/C combination. The second message reports that QUEMAN has removed the LP1SPL job from its internal tables.

Normally, the SPOOL program starts when the INIT system program executes commands in the START.CTL and CRASH.CTL files. For example, a portion of a typical START.CTL file appears as follows.

```
FORCE KB9:  RUN $QUEMAN
FORCE KB9:  \DE
      .
      .
      .
FORCE KB12: RUN $SPOOL
FORCE KB12: Ø
```

These commands assume both keyboards are logged into the system under privileged accounts. For more information on the INIT system program, see Section 6.1 of this document.

6.10.2.1 Recovery from Line Printer Errors - Control of error handling in the spooling process is by interaction between the spooling program and the QUEMAN program. For example, if the supply of paper is exhausted or if the paper jams, SPOOL generates the DEVICE HUNG OR WRITE LOCKED error (ERR=14), sends a message to QUEMAN, and discontinues processing. QUEMAN prints a message at the system console terminal (KBØ:) similar to the following sample printout.

```
****QUEMAN (Ø2)  3:LPØSPL REQUEST: LPØ: HUNG?
```

The message tells the user that the spooling job for line printer unit Ø (LPØSPL) has generated request number 3. Before SPOOL can resume, the user must correct the error. If SPOOL determines that the error is gone, it continues printing based on the options specified when the requester queued the file for printing. In such a case, SPOOL notifies QUEMAN which, in turn, deletes request number 3 and prints a message to that effect.

If the user cannot correct the error, or wishes to perform some other operation, he must leave the line printer off-line, attach the QUEMAN job to a terminal and type the request number followed by a SPOOL command. Table 6-11 describes the valid commands. If the user types an invalid command, SPOOL causes QUEMAN to print the ILLEGAL RESPONSE message.

Table 6-11
SPOOL Commands

Format	Operation	Meaning
CO	Continue	Continues by printing the current job according to options the user specified to QUE.
DE	Defer	Places current job at the end of the queue to be rerun at a more convenient time.
RE	Restart	Continues by printing the entire job again from the job header page onward.
KI	Kill	Terminates the current job and removes it from the queue file.

To respond to a pending action request, leave the line printer off-line, note the job number reported in the QUEMAN message, and type the ATTACH command with the number as follows.

```
ATTACH 2
PASSWORD:
ATTACHING TO JOB 2
'QUEMAN' ATTACHED
TYPE \DE TO DETACH
*
```

To restart the job from the beginning, type the request number with the RE command.

```
*3   RE
*   \DE
```

QUEMAN passes the string RE to the spooling program related to request number 3 and prints the asterisk character. The user can then place the line printer on line. SPOOL runs and begins printing the job from the file header. The user then types \DE to detach QUEMAN. For more detailed information on QUEMAN, see Section 6.10.1.

6.10.2.2 SPOOL Line Printer Output - SPOOL generates job header and file header pages to identify print jobs and files within a print job. Both types of header page contain identification and general accounting information. The identification information consists of large, easily

readable, aggregate letters created from the character generation file CHARS.QUE. The accounting information is in standard letters and placed on the page according to the type of header.

The job header identification consists of the account number of the user requesting the job and the job name the user gave in the QUE command. If no job name appeared in the QUE command, SPOOL prints the filename of the first file in the request as the job name.

General accounting information for the job header is centered on the page and is offset from the identification information by two rows of special characters. The accounting information contains three lines of data. The first line consists of the job name, current date, current time, and requester's account in the following general format.

JOB name PRINTED ON date AT time FOR USER [n,m]

The second line comprises the date and time when the user created the request and the device for which he created the request. The third line gives the QUE options the user specified in the job identification part of the QUE command. If the user gave no options, SPOOL prints only /MODE=Ø.

The file header identification shows, on separate lines, the filename and extension of the file SPOOL printed. If SPOOL does not print the file because of an error, the identification and accounting information are replaced by an error message framed above and below by five rows of special characters. The error message has the following format:

FILE filename specification -- RSTS error message

The file specification includes the device, filename, extension and project-programmer field. The error message is the text generated by the system upon encountering such an error. These errors are summarized in Appendix C of either the BASIC-PLUS Language Manual or the RSTS-11 System User's Guide.

The accounting information for the file header appears below the identification and is framed, above and below, by two rows of special characters. The first of two lines of data gives the job name used

when SPOOL printed the file and shows the current date, time, and account as the job header accounting information does. The second line gives the complete file specification and the QUE options the user specified in the file identification part of the QUE command. The QUE options are preceded by the text SWITCHES=. If the user gave no options in the QUE command, SPOOL prints the assigned default values. For more information on the QUE options, see Section 4.11 of the RSTS-11 System User's Guide.

6.10.2.3 Job Error Messages - Errors SPOOL encounters during printing are reported in the line printer output. SPOOL differentiates the error message from requested output by framing the text with five rows of special characters.

The text of the error message is split in two parts separated by - characters. The first part of the text varies according to the type of error. If the error is other than DEVICE HUNG OR WRITE LOCKED, the message begins with the RSTS error message text. Otherwise, the text begins with the line printer device designator followed by the words HUNG ERROR. The remainder of the first part qualifies the nature of the error. If the error is file related, SPOOL includes the text FILE followed by the filename specification.

The second part of the text varies according to the operator or system action taken to recover from the error. The following list shows the texts and related meanings.

JOB RESUMED	Operator used CO command or system automatically continued the job.
JOB DEFERRED	Operator used the DE command to defer the job.
JOB RESTARTED	Operator used the RE command to restart the job or the system automatically restarts the job.
JOB ABORTED	Operator used the KI command to terminate the print job.

If operator action was involved in the error, the program prints the text BY OPERATOR following the error text. If, for any reason, SPOOL cannot continue or restart, it includes the text FAILURE TO RESTART.

6.10.3 Batch Processor Program - BATCH

The batch system program runs without user intervention and executes files of standardized commands queued on the related batch device. The number of batch devices possible depends upon the number of pseudo-keyboards available for use by BATCH. To run BATCH, the user job is logged into the system under a privileged account. To start BATCH, type the following command.

```
RUN $BATCH
```

BATCH runs and checks that the project number of the account is 1. If the account is not privileged, the program prints the PROTECTION VIOLATION message and terminates. Otherwise, the program prints two lines. The first line gives the program name and version number and the second line requests the unit number of the batch device. For example:

```
BATCH VØ1-27
BATCH UNIT BA?
```

The user can type the RETURN key to indicate any batch device (BA:) or can type a decimal number to indicate a distinct batch device (BA1:, BA2:, and onward).

BATCH runs and processes messages through the QUEMAN program, which must be running on the system. For more information on sending messages through QUEMAN, see Section 4.12.3.8 of the RSTS-11 System User's Guide.

6.10.4 Terminating an Individual Spooling Program

It is possible to terminate an individual spooling program. However, it is not recommended since QUEMAN automatically terminates spooling programs when SHUTUP runs. To terminate a particular spooling job, the user must attach QUEMAN to a terminal, release the desired spooling job, and run the UTILTY system program to kill the job. It is important that the user release the spooling job by QUEMAN because requests in the QUEUE.SYS file may otherwise be lost.

The following sample dialog shows the proper way to terminate the BATCH controller program.

```
ATTACH 2
PASSWORD:
ATTACHING TO JOB 2
'QUEMAN' ATTACHED
TYPE \DE TO DETACH
*\ST
 3 SPOOLERS ON-LINE
 4            LP1SPL LP1  2  Ø  Ø  Ø
 3            LPØSPL LPØ  2  Ø  Ø  Ø
 5            BATCH5 BA*  3  Ø  Ø  Ø
*\RE:5
*\DE
ADDITIONAL MESSAGES:
QUEMAN MESSAGE: BATCH5 ( 5 ) RELEASED -- TAKEN OFF-LINE AT 12:04 PM
TYPE \DE TO DETACH
*\DE
DETACHING...



HELLO 1/1ØØ
PASSWORD:
JOB(S) 2 3 4 5 ARE DETACHED UNDER THIS ACCOUNT
JOB NUMBER TO ATTACH TO?
4 OTHER USER(S) ARE LOGGED IN UNDER THIS ACCOUNT



READY

RUN UTILTY$
'UTILTY' SYSTEM UTILITY PROGRAM
? KILL 5
? EXIT

READY

BYEF
```

If the \RE:n command is typed and the spooling program is currently processing a request, QUEMAN places that request at the beginning of the queue. When the spooling program starts again, that request is immediately restarted from the beginning. However, when SHUTUP terminates time sharing, the status of jobs terminated depends on the individual spooling program: SPOOL and RJ2780 jobs are placed at the beginning of the queue and BATCH jobs are deleted from the queue. For jobs terminated individually using the \RE:n command, QUEMAN always places a currently processing request at the beginning of the queue.

6.11 MANAGING RSTS/E ERROR LOGGING - ERRCPY, ERRCRS AND ERRDIS

Logging of hardware errors is an automatic function of the RSTS/E monitor. To gain the advantages of error logging, the system manager must properly employ the ERRCPY, ERRCRS, and ERRDIS system programs.

The ERRCPY program retrieves error-related data logged by the RSTS/E monitor. Upon occurrence of a hardware error, special routines save the contents of the device registers in small buffers and effectively send a message to the ERRCPY program. The system awakens ERRCPY which transfers the saved data to disk. Since the number of messages which can be queued at any given time is limited, ERRCPY must be running to prevent loss of valuable diagnostic information.

The ERRCRS program retrieves error-related data saved following a system crash. When a system crash occurs and certain conditions are in effect, the monitor preserves the contents of certain critical parts of the system. The system file CRASH.SYS holds this information along with other error-related device data. The ERRCRS program transfers the information from the CRASH.SYS file to another disk file which has the same format as the one created by the ERRCPY program.

The ERRDIS system program produces summaries of error-related data and formats it for output to a hard copy device. This program provides the record of errors logged on the RSTS/E system.

6.11.1 Operation and Use of the Error Copy Program - ERRCPY

The error copy system program ERRCPY reads error-related information stored in the monitor part of memory and writes it to a special disk file. The system manager must ensure that the proper commands are created in the START.CTL and CRASH.CTL files as described in Section 6.1 so that ERRCPY is started and active during time sharing operations. The following discussion outlines the entire process of activating the job which runs ERRCPY.

When the RSTS/E system starts up, commands in either the START.CTL or CRASH.CTL control file are executed by the INIT system program. If the command FORCE KB0: RUN $ERRCPY appears in the control file, the command RUN $ERRCPY is placed in the input buffer of the console

terminal (KBØ:) as if it had been typed at the terminal. Meanwhile, the accompanying END command in the control file causes termination of the INIT system program and causes the console terminal to be placed at BASIC-PLUS command level (edit mode), as signalled by the READY message being printed. The console terminal remains logged into the system under account [1,2].

When the system executes the command RUN $ERRCPY from the input buffer of KBØ:, the ERRCPY program runs and detaches itself from the console terminal as indicated by the message DETACHING printed at the console terminal. The console terminal is not thereafter logged into the system, but ERRCPY continues running as a detached job under account [1,2].

When ERRCPY is activated, it exists in the SL (sleep) state and neither occupies memory storage nor uses CPU time until awakened by the RSTS/E Monitor error logging routines. When error logging detects a hardware error, it causes ERRCPY to run and write the error-related information to a special file ERRLOG.FIL. The file is stored under the system library account [1,2] on the system disk. If ERRCPY is not running, the diagnostic area can overflow and the history of subsequent errors can be lost. Therefore, the system manager must properly start the ERRCPY job.

The ERRCPY program automatically kills itself whenever logins are disabled. Therefore, if the system manager disables logins, he should restart ERRCPY.

The system manager gains information concerning the hardware errors detected and placed in the ERRLOG.FIL by running the ERRDIS system program as described in Section 6.11.3. If a system crash occurs, the system manager can retain error data by following the instructions in Section 6.11.2.

6.11.2 Use of the Error Crash Program - ERRCRS

The ERRCRS system program saves error information retrieved at the time of a system crash. When system crash occurs, critical contents of memory are written to the system file CRASH.SYS if the user enabled the CRASH DUMP facility at start up time. The ERRCRS system program transfers certain error information from the file CRASH.SYS to a user designated file. The following sample dialog shows the use of ERRCRS.

```
RUN $ERRCRS
ERRCRS VØ5-Ø6
OUTPUT FILE NAME? FILE.CRS
CRASH DUMP FILE NAME?

READY
```

ERRCRS is executed by typing the RUN $ERRCRS command from a terminal logged into the system under a privileged account. Two queries are printed. The response to the first query designates the name of a file to which error information will be written. The response to the second query is simply the RETURN key, designating the file CRASH.SYS stored under the system account [0,1]. The ERRCRS program writes the error information to the file named FILE.CRS (in this sample) and terminates automatically, as signalled by the READY message being printed.

The system manager can later print a report on the error information saved if he uses the ERRDIS system program as described in Section 6.11.3 and designates the filename specified as output of the ERRCRS program run as the input filename for ERRDIS. It is highly recommended that users place the proper commands in the CRASH.CTL file so that ERRCRS runs automatically upon initialization of the system after a system crash.

6.11.3 Operation and Use of the Error Display Program - ERRDIS

The error display program ERRDIS allows the system manager to gain full or partial history or a full or partial summary of the error-related information preserved by the ERRCPY or ERRCRS system programs. ERRDIS prints, in an organized and formatted fashion, the error-related information read from a disk file according to options and

switches specified by the system manager. The file is created by either the ERRCPY or ERRCRS system program and exists under the system library account [1,2] with protection code <60>. The disk file can maintain a history of a maximum of 880 errors and can record a maximum of 100 of any one type of error. If either of these limits is reached, ERRDIS prints in the output history a message telling how many errors were missed due to no room or to the limit of 100. The following two sections describe how to run and terminate ERRDIS and how to optimally use ERRDIS features.

6.11.3.1 Running and Terminating ERRDIS - The system manager or privileged user runs the ERRDIS program by typing the following command while logged into the RSTS/E system.

```
RUN $ERRDIS
```

The program responds by printing a program header line, followed, in turn, on subsequent lines, by three queries as shown below.

```
ERRDIS VØ5-1Ø
INPUT FILE NAME (<CR> FOR DEFAULT)?
OUTPUT TO?
OPTIONS?
```

The user types the RETURN key in response to the query concerning the input file name and ERRDIS prints the second query concerning the output device or file. The default input file name is $ERRLOG.FIL. The user can specify as input the name of the file created by the ERRCRS program. If the user types the RETURN key in response to the second query, the error-related information subsequently requested is printed at his terminal keyboard printer. To indicate a different output device, or file, type the proper specification followed by the RETURN key.

After the user designates the output, ERRDIS prints the OPTIONS query. An option from those given in Table 6-12 can be typed. An option can be modified by any of several switches as described in Table 6-13. After output of the option or options specified is completed, the OPTIONS query is printed again. To terminate the ERRDIS program, type the EXIT command in response to the OPTIONS query.

Table 6-12
ERRDIS Program Options

Option Type	Option Format	Meaning
General	ALL	Error-related information for all errors is printed in the order in which they were detected and recorded, from the earliest to the most recent.
	EX	Terminate ERRDIS and exit to the monitor.
	HE	Print the help file ERRDIS.HLP.
	MS	Missed errors.
Peripheral Errors	DH	Prints error-related information concerning the DH11 multiplexer.
	DT	TC11/TU56 DECtape.
	RF	RF11/RS11 fixed head disk.
	RC	RC11/RS64 fixed head disk.
	RK	RK11/RKØ5 or RKØ3 DECpack cartridge drive.
	RP	RP11-C/RPØ3 disk pack drive.
	RS	RH11/RSØ3 and RH11/RSØ4 fixed head disk.
	MA	RH11/TMØ2/TU16 magtape.
	RB	RH11/RPØ4 disk pack drive.
	RJ	DP11 or DU11 interface for RSTS/278Ø software.
	CD	Card reader.
	MT	TM11/TU1Ø Magtape.
	KB	Hung Teletype errors by job number and keyboard line number.
Processor Errors	T4	Traps through vector location 4.
	TØ	Traps through location ØØØØØØ.
	JØ	JMP instructions executed to location ØØØØØØ.
	RI	Reserved instruction traps.
	PF	Occurrences of power failure.
	CK	Checksum errors.
	MP	11/45 or 11/4Ø memory parity.
	MM	Memory management
	??	Illegal code.

Table 6-13
ERRDIS Program Option Switches

Switch Format	Meaning
/S	Print only a summary of information of the error type indicated in the option. (Used alone, /S is meaningless.)
/K	Delete (kill) information in the error logging file. If the file is $ERRLOG.FIL, the /K option simply zeroes the file but keeps it in the directory. If the file is anything other than $ERRLOG.FIL, the /K option deletes the directory entry. In either case, ERRDIS terminates and returns control to READY.
/H	Used alone; causes a help file to be printed.
/dd-mmm-yy	Prints information concerning the error type indicated in the option if it was detected on or after the date designated by /dd-mmm-yy. For example, 19-MAY-73.
/hh:mm	Prints information concerning the error type indicated in the option if it was detected at or after the time of day designated by /hh:mm. For example, 8:5Ø or 2Ø:5Ø. If a date switch appears with a time switch, ERRDIS prints errors detected at or after the date and time of day. If a time switch appears without a date switch, ERRDIS uses the current system date.

```
OPTIONS?   EXIT
READY
```

Control is returned to BASIC-PLUS command level, as indicated by the READY message being printed.

6.11.3.2 Recommended Usage of ERRDIS - The recommended procedure for using the ERRDIS program is to daily request at least two specific options: ALL/S and ALL. The procedure entails running ERRDIS and answering the OPTIONS query in the following manner.

```
OPTIONS? ALL/S
OPTIONS? ALL
```

The ERRDIS program first creates a summary (/S) of all error-related information. The output starts with 4 lines of accounting data. On the first line, ERRDIS indicates the option requested, followed, on a second line, by the file name from which the information is taken (usually $ERRLOG.FIL). On the third line appears the output specification used and, on the fourth line, the time of day and current date. Following the accounting data is the summary of the total number of errors-recorded and errors missed by type and a tally of certain disk input and output information.

For the second option requested (ALL), ERRDIS prints the accounting data and the entire history of the errors logged. Information for each occurrence of a logged error is printed in chronological order, beginning with the earliest error and continuing to the most recent occurrence. For each error logged, a header line is printed which describes the type of error and the time of day and date of the occurrence. Following the header line for each error, ERRDIS prints such data as job number, keyboard number (if a hung Teletype error), processor status word (PSW) contents, and the contents (in octal) of the device registers at the time of the error. Consult the PDP11 Peripherals Handbook for the meaning of the device register abbreviations and the types of errors encountered. (A job number of Ø indicates the null job.) A comment line is appended to some error-related information, such as that of a hung Teletype.

At the conclusion of the error history, ERRDIS prints the number of missed errors (if any) and the total number of errors listed of those logged since the beginning of the error history. It is recommended that the user specify a disk file to contain the output of the options. The output can be gained by queuing the file on an 128 column line printer. The printouts of the complete summary and the complete history should be inspected and stored in a central location reserved for them. They provide the basis for planning preventive maintenance and the means to more readily isolate potentially dangerous hardware problems. The printouts should be available to the DEC Field Service or Software Support representative. Periodically, the system manager can delete the contents of the file $ERRLOG.FIL by specifying the following option in response to the ERRDIS program OPTION query.

OPTION? /K

The system manager should be alert for certain conditions reported by ERRDIS. Several hung Teletypes are not serious, but a steadily increasing number of hung Teletypes on a certain keyboard line indicates a possibly dangerous condition which should be remedied. Any occurrence of a TØ error is serious and indicates that an interrupting device has presented an incorrect vector location to the bus. An increasing number of disk errors (particularly on the system disk) indicates a need for immediate maintenance.

6.12 OCTAL DEBUGGING TOOL - ODT

The system program ODT opens a file, a peripheral device, or memory as an address space and allows a user to examine and change word or byte locations within the address space. As auxiliary operations, the user can list the contents of certain conventional table locations in the operating system.

The program immediately interprets and executes each character as the user types it. This action is termed ODT submode or ODT character mode and differs from the procedure used by other system programs which interpret input only after the user enters an entire line of characters. Since ODT performs processing based upon single characters typed at the terminal, its language is highly interpretive and interactive. It thus provides a quick and efficient means of finding errors in program and data files and changing data in those files for testing purposes. Because of the quickness and efficiency of ODT, it is advised that only experienced user's employ it to perform testing and error correction on a system's data base.

The program accesses and manipulates data in word and byte locations based on octal values. The word is the 16-bit PDP-11 word and can have a value between 0 (octal) and 177777 (octal), the limit imposed by 16 bits. A word has a high order (odd address) and low order (even address) byte. A byte can have a value between 0 (octal) and 377 (octal) - the limit that can be represented by 8 bits. For the purposes of clarity in this chapter, the following symbols express the octal values used by ODT.

Symbol	Meaning
n	Represents an octal integer between Ø and 7. The use of 8 or 9 generates an error.
k	Represents an octal value up to 6 digits in length. If more than 6 digits are specified or a value greater than 177777 (octal) is specified, ODT truncates the value to the low order (rightmost) 16 bits. If the octal value is preceded by a minus sign, ODT uses the 2's complement value of the number.

For example, ODT interprets the following values as shown.

1	ØØØØØ1
-1	177777 (2's complement)
400	ØØØ4ØØ
-177730	ØØØØ5Ø (2's complement)
1234567	Ø34567 (truncated to low-order 16-bits)

The user can represent a location within the permissable address space by typing an octal value or an expression which reduces to an octal value. The following are the correct forms and the interpretation by ODT.

k — The 6 digit octal value of k. If the address space is partitioned in 512-byte blocks (for example, a disk file), the leftmost 3 digits specify a block number and the rightmost 3 digits specify a byte location within the block. For example,

```
000017
```

The value designates byte 17 (octal) in block Ø.

```
3412
```

The value designates byte 412 (octal) in block 3 (octal).

n,k — The resultant address is the value of k added to the contents of the relocation register specified by n. Relocation registers are numbered from Ø to 7. See Section 6.12.4 for more information concerning relocation registers.

The special characters and symbols in Table 6-14 are recognized by RSTS/E ODT and explained in the remainder of the section.

Table 6-14
ODT Characters and Symbols

Character(s) or Symbols	Meaning
/ k/	Open the previously open location as a word or open the location designated by k as a word.
\ k\	Open the previously open location as a byte or open the location designated by k as a byte.
" k"	Give the ASCII representation of the currently open or last previously open location or of the location specified by k.
% k%	Give the ASCII representation of the Radix-50 value in the currently open or last previously open location or in the location specified by k.
RETURN key k followed by RETURN key	Close the currently open location or modify the contents of the currently open location with the value k and close it.
LINE FEED key k followed by LINE FEED key	Close the currently open location and open the next sequential location or modify the contents of the currently open location with the value k before closing it and opening the next sequential location.
^ or ↑	Close the currently open location and open the preceding sequential location. (On some terminals, the ↑ or ^ character is typed by depressing the SHIFT and N keys simultaneously.)

(continued on next page)

Table 6-14 (Cont.)
ODT Characters and Symbols

Character(s) or Symbols	Meaning
↑ or —	Take the contents of the currently open location as a PC relative offset and calculate the next location to be opened; close the currently open location and open the location thus evaluated.
@ k@	Take the contents of the currently open location as an absolute address, close the currently open location, and open and print the contents of the location thus evaluated. If @ is preceded by k, the value k replaces the contents of the currently open location before it is closed.
> k>	Take the low order byte of the currently open location as a relative branch offset and calculate the address of the next location to be opened; close the currently open location and open and print the contents of the relative branch location thus evaluated. If >is preceded by k, the value k replaces the contents of the currently open location before it is closed.
<	Close the currently open location and open the last location explicitly open. Returns ODT to the origin of a sequence of relative locations determined by ↑, @, and > character operations.

(continued on next page)

Table 6-14 (Cont.)
ODT Characters and Symbols

Character(s) or Symbols	Meaning
,	Separates a relocation register number from an octal value. ODT adds the contents of the specified relocation register to the octal value following the comma and forms a relocatable address.
;	Separates multiple values in a list request using the L character and in a register operation using the R character.
.	Specifies the last explicitly open location similar to that used by the < character operation.
+ space bar	Add the preceding value and following value and use the result.
−	Subtract the following value from the preceding value and use the result.
R	Reset all relocation registers to -1 (177777).
nR	Reset relocation register n to -1 (177777).
k;nR	Set relocation register n to the value k.

(continued on next page)

Table 6-14 (Cont.)
ODT Characters and Symbols

Character(s) or Symbols	Meaning
F	Set relocation calculation for list requests using L character.
1F	Disable relocation calculation set by F character.
C	Print out monitor table symbolic names and memory addresses.
$S	Print out the processor status word.
Q	Use the last quantity printed by ODT.
k1;k2L	Print contents of locations k1 through k2 at the terminal.
1;k1;k2L	Print contents of location k1 through k2 on line printer unit (Ø).
2;k1;k2L	Print contents of location k1 through k2 on another device. ODT prints DEVICE question, to which user types the device designator.

6.12.1 Running and Terminating ODT

The user runs ODT by typing the following command.

```
RUN $ODT
FILE?
```

ODT runs and prints the question FILE. The user response to this query determines how ODT runs and what address space ODT accesses. The possible responses are listed and described in Table 6-15.

Table 6-15
ODT FILE Question Responses

Response	Meaning
Type the RETURN key only.	Allows read access to memory only if user is privileged.
Type the ALT MODE key only.	Allows read access to the file CRASH.SYS in account [0,1].
Type the LINE FEED key only.	Same as the RETURN key.
Type a file specification followed by the RETURN key or the ALTMODE key.	Allows read access to the file on the device specified. If no device is specified, the system disk is used.
Type the file specification followed by the LINE FEED key.	Allows read and write access to the file specified.

ODT determines the address space by the response to the FILE question and indicates its readiness to accept commands by printing the * character. For example,

```
FILE?  ABC.DAT      (Terminate with LF)
*
```

ODT opens for read and write access the file ABC.DAT on the system disk under the current account. To terminate ODT, type the CTRL/Z combination in response to the * character. For example,

```
* ^Z
READY
```

ODT closes any file currently open and returns control to BASIC-PLUS command level.

6.12.2 Opening and Closing Locations in the Address Space (/and\)

ODT accesses the address space as either a word or a byte. The user indicates the type of access by specifying the / or \ character. For example,

```
*1ØØØ/
```

The user types the address 1000 (octal) followed by the / character. ODT opens the location as a word, generates a space, prints the 6-digit octal contents of the word, generates another space, and leaves the location open for change. The following demonstrates the results.

```
*1ØØØ/ ØØ41ØØ
```

ODT prints the contents of location 1ØØØ as ØØ41ØØ. To close the location, type the RETURN key. ODT closes that location prints an * character on the next line, and does not open a new location.

The user specifies the \ character to open a location as a byte. For example,

```
*1ØØØ\ ØØØ
*
```

ODT opens location 1000 as a byte, generates a space, prints the 3-digit octal contents of the byte, generates another space, and leaves the location open for change. To close the location, type the RETURN key. ODT closes the location, prints an asterisk character on the next line, and does not open a new location.

To change location 10ØØ, type a new 6-digit octal value followed by the RETURN key. For example, if location 1ØØØ is open as a word,

```
*1ØØØ/  ØØ41ØØ  ØØ4ØØØ      (Type RETURN key)
*
```

ODT replaces the current contents 004100 with the specified contents ØØ4ØØØ, closes the location, and prints the * character on the next line.

To determine the contents of the current word location, type the / character. For example, if the current location is 1000, the following occurs.

```
*/ 004000
```

ODT opens the current location, generates a space, prints the 6-digit contents and generates another space.

If the user types the LINE FEED key while a word location is open, ODT closes the current location and opens the next sequential location. For example, if location 1000 is open as follows,

```
*1000/  004000
```

and the user types the LINE FEED key, the following occurs

```
*1000/  004000
001002/  012345
```

ODT generates a carriage return and line feed operation and prints the address of the next location, followed by the / and space characters and the contents of the word. The new location 1002 is open. Repetitive use of the LINE FEED key causes ODT to open and display the contents of sequential locations.

If the user types the LINE FEED key while a byte location is open, ODT performs the same actions as described for a word location except that the next location is treated as a byte.

6.12.2.1 Opening the Preceding Location (↑ or ^) - Typing the up arrow (↑) character or the circumflex (^) character when a location is open causes ODT to close the currently open location and to open the immediately preceding location.

NOTE

On ASR-33 type terminals, type the ↑ character by depressing the SHIFT key and the N keys simultaneously.

For example, if two sequential locations are successively opened, typing the ↑ character opens the immediately preceding location.

```
*1000/  002345              (Type LF key)
001002/  012740↑
001000/  002345
```

Typing the ↑ character closes location 1002 and opens location 1000 and prints its contents. If a byte location is currently open, typing the ↑ character opens the immediately preceding byte location. If the user types a value followed by the ↑ character, ODT replaces the current contents with the specified value before closing the location.

If the user types the ↑ character and a location is not currently open, ODT opens and prints the contents of the last currently open word or byte location. For example,

```
*1000/  002345          (Type RETURN key)
*↑
001000/ 002345
```

ODT prints the address and the contents of the word location on the next line.

6.12.2.2 Opening a PC Relative Location (← or _) - Typing the ← character or _ character when a location is currently open causes ODT to add 2 to the address of the current location, to add the resultant sum to the contents of the current location, and to open the location specified by the final sum. For example,

```
*1000/  000040←
001042/  012345
```

ODT closes location 1000, adds 2 to the address (1000), and adds the resultant 1002 to the contents (40) of the current location. As a result, ODT opens location 1042 and prints its contents. This method of calculating the next location to open is similar to that used in relative addressing by the program counter in the PDP-11 computer. Such a method of address calculation is for position independent code.

If the contents of the current location is an odd value, ODT opens and prints the contents of the low-order byte of the PC relative location. If the user types a value followed by the ← character, ODT modifies the current contents and uses the new value to calculate the PC relative address.

6.12.2.3 Opening an Absolute Location (@) - Typing the @ character when a location is currently open causes ODT to take the contents of the current location as the address of the next location to open. As a result, ODT closes the current location, opens the calculated location and prints its contents. For example,

```
*1006/  001024  @
001024/  ØØØ5ØØ
```

ODT uses the contents of the current location (1024) as the next location to open.

NOTE

On ASR-33 type terminals, type the @ character by depressing the SHIFT key and the P key simultaneously.

If the user types a value followed by the @ character, ODT changes the contents of the current location to the value and uses the new value to determine the next location to open. The method is equivalent to absolute addressing on the PDP-11 computer where the contents of the location following an instruction are taken as the address of the operand. The address is absolute since it remains constant regardless of where in memory the assembled instruction is executed.

6.12.2.4 Opening a Relative Branch Offset Location (>) - Typing the > character when a location is currently open causes ODT to use the signed value of the low-order byte of the current location to determine an offset from the current location. ODT uses the final sum to open the next location and print its contents. For example,

```
*1032/  000407>
001052/  001456
```

ODT takes the contents of the low order byte (ØØ7) and multiplies by 2 to give 16 (octal). Next, ODT adds 2 to the address of the current location (1032) to give 1034. Finally, ODT adds these two quantities (1034 + 16) to give the address of the word (1052) to open. ODT closes the currently open location, opens the calculated location, and prints the address and the contents on the next line.

If the user specifies a value followed by the > character, ODT modifies the contents of the currently open location and uses the low order byte of the new value to calculate the relative branch offset location. For example,

```
*1Ø32/   ØØØ4Ø7  3Ø1>
ØØØ636/  ØØØØ1Ø
```

ODT interprets the byte value 301 as a negative value since the high order bit is 1. The absolute value of 301 is 77 (octal) which is multiplied by 2 to give 176 (octal). ODT subtracts the relative branch offset (176) from the address plus 2 of the current location to give 636 as the address of the next location to open. ODT opens the new location and prints its contents.

6.12.2.5 Returning to an Interrupted Sequence (<) - Typing the < character causes ODT to close the currently open location and open the last explicitly open location. This command is useful, when a user has typed the ^, @, and > characters, or any sequence thereof, and wishes to open the locations from which ODT calculated subsequent relative locations. For example,

```
*1032/   000301 >
000636/  ØØØØ1Ø @
000010/  123456 <
001032/  000301
```

After typing the > and @ characters, ODT opens location 1Ø. The user returns to the last explicitly open location by typing the < character. ODT opens and prints the contents of location 1032, the last location explicitly opened.

6.12.3 Printing the Contents of Locations

The user can type the L character in three ways to print the contents of locations in the address space open by ODT. For example, to print the contents of a certain range of addresses, specify the start and end addresses with the L command as follows.

```
*0;776L
```

ODT prints at the terminal, the octal contents of each word between address ØØØØØØ and ØØØ776. Beginning each line of the printout, ODT prints the address of the first word on the line. The user can turn the printing on and off by typing the CTRL/O combination.

To print a listing on line printer unit Ø, type the L command as follows.

```
*1;Ø;776L
```

ODT prints, on line printer unit Ø, the octal contents of each word between addresses 000000 and 000776. To specify another unit, type 2 preceding the command.

```
*2;Ø;776L
DEVICE? LP1:
```

ODT prints the DEVICE question, to which the user types the device designator of the line printer unit to be used.

6.12.3.1 Printing ASCII Format (") - Typing the " character when a location is currently open causes ODT to print the ASCII representation of the word or byte. For example,

```
*1000  \  101  "  A  (Type LF key)
001001   103  "  C  (Type CR key)
*1000/ 41501
```

If the currently open location is open as a word, typing the " character causes ODT to print 2 characters representation of the word. For example,

```
*1032/ 034567  "  W9
```

The low order byte contains 167 (octal) which is W and the high order byte contains 071 (octal) which is 9.

If the user types the " character and a location is not currently open, ODT prints the ASCII representation of the previously open location.

6.12.3.2 Printing Radix-50 Format (%) - Typing the % character when a location is currently open causes ODT to print the 3-character ASCII representation of the Radix-50 word. For example,

```
*1000/ 034567 % IGl
```

If the user types a value preceding the % character, ODT interprets it as the address whose contents are to be interpreted and printed. For example,

```
*1000 % IGl
```

ODT interprets 1000 as the address to use. If the user types the % character and a location is not currently open, ODT prints the 3-character ASCII representation of the previously open location.

6.12.4 Relocation Registers - ODT has available eight relocation registers numbered Ø through 7 which the user can employ to specify relative addresses. ODT initially sets the relocation registers to -1 (177777, the highest possible address) to prevent inadvertent errors in address calculation. The user sets a relocation register by typing the relative address, followed by a semicolon and the specification of one of the eight relocation registers. For example, to set relocation register Ø to 1000, type the following,

```
*1000;ØR
*
```

The user can subsequently use the value in relocation register Ø as an offset or a base address in specifying a location. For example, to open location 1032, as a word, the user types the following.

```
*Ø,32/ ØØØØ1Ø
```

ODT adds the offset 32 to the contents of relocation register Ø to open the location. Since relocation register Ø contains 1000, ODT opens location 1032.

To reset the contents of all relocation registers to -1, the user need type only the R character. For example,

```
*R
*
```

ODT generates the carriage return and line feed operation and prints the * character again. To reset the contents of any one register to -1, simply specify the register number followed by the R character. For example,

```
*1R
*
```

The above command resets relocation register 1 to 177777 (octal).

To print the contents of locations based on a fixed offset from the relocation registers, type the F character and subsequently use the L character. For example,

```
*F
*1;Ø;3000L
*
```

The F character conditions ODT to calculate relocated addresses for a printout. The next command tells ODT to add the offsets from Ø to 3000 to the value of relocation register Ø and print the contents of those resultant locations on line printer unit Ø. The procedure is repeated for the values of relocation registers 1 through 7. The resultant printout contains a listing of the contents of addresses n,Ø through n,3ØØØ where n is between Ø and 7 (the relocation registers).

To turn off calculation of relocation addresses for a printout, type 1F. For example,

```
*1F
*
```

Subsequent listing requests generate printout for actual addresses rather than relocated addresses.

6.12.5 Interpretive Address Quantities (Q and .)

ODT uses the variable Q to store the last value which it printed at the terminal. The user can type Q to designate the value so stored.

ODT performs any valid operation requested and automatically extracts the value from Q. For example, if the user desired to increase the value in an open location by a certain increment, he could proceed as shown in the following sample.

```
*1342/  173214 Q+10   (Type RETURN key)
*/ 173224
```

The user types 1342/ to open that location as a word. ODT prints the contents of the location and stores that value in Q. The user subsequently types Q+10 followed by the RETURN key. ODT adds 10 to the value in Q and modifies the current location with the sum. The user thus does not have to retype the number or calculate the sum. To verify that ODT has changed the location properly, the user types the / character. ODT opens the last previously open location (1342), prints the contents, and, additionally, updates the value Q with the most recently printed quantity.

The . character indicates the currently open or last explicitly open location and can be typed to indicate the current address for ODT operations. This is the same address used by the < character described in Section 6.12.1.5. In most cases, the . character value is the address used by ODT when the user types the /, \, ", %, and LINE FEED characters. For example, to open as a word a location 16 bytes from the last explicitly open location, the user types the following,

```
*.+16/  012345
```

ODT adds 16 (octal) to the address given by the . character, opens the resultant address, and prints its contents.

6.12.6 Error Procedures

If the user types an invalid or unrecognized character, ODT prints a ? character, generates a carriage return and a line feed operation, and prints the * character. For example,

```
* 1008?
*
```

ODT indicates that the character 8 is an error. The user must retype the number correctly.

If ODT encounters an error while performing output to a device, it prints the message I/O ERROR? followed by the * character. The user must correct the device error and type the command again.

6.13 VT05 DISPLAY OF SYSTEM STATUS - VT5DPY

The VT5DPY system program displays the system status on a VT05 alphanumeric display terminal and updates the status at given intervals. The program is stored in the system library with protection code <232> and requires a 14K job area to run. Since the program runs in such a large job area, it is suggested that it be used only on systems with ample memory.

The operation of the program is automatic or can be varied at user option. When the user starts the program, he specifies the interval at which VT5DPY updates the information on the screen. The information displayed is similar to that given by the SYSTAT system program. However, the user can type commands on the VT05 to modify the items displayed. For more information in SYSTAT, refer to Section 4.3 in the RSTS-11 System User's Guide and Section 6.6 of the RSTS/E System Manager's Guide.

6.13.1 Running and Terminating VT5DPY

A non-privileged job can run VT5DPY by typing the following command at a terminal.

```
RUN $VT5DPY
INTERVAL?
```

VT5DPY runs and prints the INTERVAL question to which the user can type the number of seconds between updates and any combination of the following modifiers.

Modifier	Description
/DET	Have the program run detached from this terminal or from the terminal specified.
/KBn:	Run the program at keyboard unit n if it is available. If /DET is specified, run the program detached.
/NO FILL	Set the fill factor to 0. Improves program performance when the VT05 runs at 300 baud or less.
/PRIORITY	Run the program at special priority rather than at -8 priority.

If the user types only the RETURN key or types no number with a modifier in response to the INTERVAL question, VT5DPY updates the status every 15 seconds.

Running the program detached allows the user to temporarily interrupt the program and use the screen to run other programs. To interrupt VT5DPY, type the CTRL/C combination and the program prints a message telling the user that the screen is available. When the user releases the screen, VT5DPY automatically displays the status information on the screen again.

When VT5DPY displays the system status, it prints a header line on the top line of the screen, skips a line, and fills the leftmost half of the screen with job status information and the rightmost half of the screen with, in turn, disk structure, busy device, free buffer status, and message receiver information. Upon filling the screen the program moves the blinking cursor to the first character position on the second line of the screen. This indicates that the program is idle.

At the interval specified, VT5DPY interrogates system tables and updates the status information on the screen with any changed data. While executing routines to extract update information, VT5DPY prints the message WORKING ... and leaves the cursor positioned to the right of the message. After completing the update, VT5DPY returns the cursor to its idle position.

While the cursor is at its idle position, the user can type commands to modify the contents and arrangement of items on the screen. Any command typed should be terminated by the ALTMODE key. Although any line terminator works, ALTMODE leaves the cursor positioned on the blank line. The program takes no action on invalid commands.

Some commands which add items to the screen can be preceded by a minus sign which indicates taking away the effect of the command. Table 6-16 lists and describes the commands.

6.13.2 Screen Layout

VT5DPY partitions the screen into three parts: the header line, the lefthand half and the righthand half. The components of these parts are defined in the following subsections.

Table 6-16

VT5DPY Commands

Command Type	Format	Description
General	C	Clear the screen and display new status.
	S	Display memory status in lieu of job status
	J	Display job status in standard manner.
	Jn	Display job status starting with n+1 active job. Use to overcome physical limitation of the screen.
	Xn	Change the interval to n seconds.
	XØ	Update the display with an interval of Ø seconds (that is, run continuously) but lower the priority so that other jobs are not stalled.
Job Status	O	Display the account number of operator jobs as [OPR]. An operator job has a project number 1 and a programmer number less than 200.
	-O	Replaces OPR in operator account designations with the actual project and programmer numbers.
	T	Display total CPU time each job has expended. The time is displayed as number of hours, minutes, seconds, and tenths of seconds under the RUN-TIME column.
	+	Display the increment of CPU time each job has expended since VT5DPY last updated the screen. User can return to total CPU time by typing T.
	%	Display the amount of CPU time each job has expended as a percent of the total CPU time expended. User can return to total or increment of CPU time by typing, respectively, T or +.
	JO	Display only operator jobs.
	J-O	Display all but operator jobs.
	J+O	Display normal and operator jobs.
	J-DET	Display all but detached jobs.

Table 6-16 (Cont)

VT5DPY Commands

Command Type	Format	Description
Job Status (Cont)	J+DET	Display all jobs including those detached.
	P	Indicate priority of jobs more exactly than + and - characters.
	W	Indicate under STATE column, the last WAIT state rather than actual state.
	-W	Remove last WAIT state and indicate actual state of each job.
	K	Display, under SIZE column, the amount of memory occupied by each job.
	-K	Display, under SIZE column, the amount of memory remaining to each job.
Disk Structure	D Dn -D	Display disk structure statistics. If n is 1, place item first on the screen. Preceding minus sign removes same from the screen.
	L -L	Display, under COMMENTS column, the logical name of each device. Preceding minus sign replaces logical names with standard PUB, PRI, NFS, or LCK notations.
Busy Devices	B Bn -B	Display busy device statistics. If n is 1, place item first on the screen. Preceding minus sign removes same from the screen.
Free Buffer	F Fn -F	Display free buffer statistics. If n is 1, place item first on the screen. Preceding minus sign removes same from the screen.
Message	M Mn -M	Display message receiver statistics. If n is 1, place item first on the screen. Preceding minus sign removes same from screen.

6.13.2.1 Header Line - The header line looks like the following sample.

```
RSTS V5B-20 SYSTEM #880 STATUS ON 31-JUL-74 11:15 UP 2:22:22
```

The header line contains the system identification, the current date and time of day and the number of hours, minutes and seconds since the start of time sharing operations. The latter item is termed up time.

6.13.2.2 Job Status - The job status consists of job statistics information similar to that of SYSTAT with the addition of a few states and a PR column for running priority.

The additional states which can appear in the STATE column are listed and described below.

Xnn	Job is swapped out and occupies slot number nn in swapping file X. The swapping file is denoted by letters A, B, C, or D to represent respectively files SWAPØ.SYS through SWAP3.SYS.
↑C	Job is in CTRL/C state, awaiting input to the monitor.
LCK	Job is locked in memory for the current operation.
PK	Job is accessing a pseudo keyboard.
RS	Job is runnable and waiting for memory to be allocated so that the system can swap it in.
SWI	Job is currently being swapped into memory.
SWO	Job is currently being swapped out of memory.

The PR column can display the following abbreviations.

+	Positive or zero
^	CTRL/C
K	Keyboard (programmed) input wait
S	Special priority

-1	Negative
Ø	Zero
+1	Positive

6.13.2.3 Memory Status - By typing the S command, the user causes VT5DPY to print a table indicating the usage of each 1K word portion of memory. This memory status report replaces the job status report on the lefthand half of the screen. To display the job information again, use the J command.

The memory status report uses the following abbreviations.

MON	Occupied by RSTS/E monitor
RTS	Occupied by BASIC-PLUS compiler and Run Time System
n	Occupied by job number n
nLK	Job number n is locked in this 1K portion
nSO	Job number n is being swapped out of memory
nSI	Job number n is being swapped into memory
NXM	Memory space is locked or nonexistent
I/O	Memory Space is reserved for I/O vectors

6.13.2.4 Disk Structure - The disk structure report is the same as that printed by SYSTAT. The COMMENT column can have the following items.

PRI	Cartridge or pack is private.
PUB	Cartridge or pack is public.
NFS	Disk is open as a non-file structured device.
LK	Disk is in locked state.
xxxxxx	Logical name of disk (the identification label or the name associated to the disk by the ASSIGN command). Appears only if user types the L command.

6.13.2.5 Busy Device Statistics - The busy device statistics are the same as the ones generated by SYSTAT.

6.13.2.6 Message Receiver Statistics - The message receiver statistics reflect information on jobs using the send/receive SYS system function call to FIP. VT5DPY displays the following information.

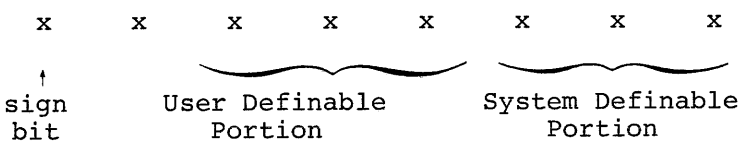
xxxxxx (PRIV)	The six character logical identification of the receiving job. The designation (PRIV) indicates the job receives messages from privileged sending jobs.
x	Number of the job.
x/x	Lefthand number (decimal) shows number of messages queued for the job and rightmost number shows the declared maximum number of messages the job can have queued.

6.13.2.7 Free Buffer Status - The free buffer statistics are the same as those generated by the SYSTAT program.

6.14 SETTING JOB PRIORITY, RUN BURST AND MAXIMUM SIZE

The PRIOR system program reports the priority, run burst and maximum size assigned to an existing job. The system manager can change any of the current values to increase or decrease the chance of gaining run time in relation to other running jobs, to determine how much CPU time the job can have when it is compute bound and to increase the area a job can occupy.

The system runs jobs on the basis of priority. The higher a job's priority, the better are its chances of obtaining run time in relation to other running jobs. Priority is determined by an 8-bit priority byte, as shown below.



By running PRIOR, a privileged user can set the user definable portion of the priority byte for his own job or for another job on the system. Since the three system definable bits are normally zero, standard priorities are multiples of 8 between -120 (lowest priority) and +120 (highest priority). Zero is a legal priority. When PRIOR changes a priority, it truncates any value not a multiple of eight to the next lowest legal priority. For example, if the system manager specifies a priority of +10, PRIOR sets the value to +8.

All of the system definable bits are normally off (zero). The least significant bit is set when a keyboard delimiter is typed during an I/O stall. Keyboard delimiters are the CR, LF and ALT keys. The next significant bit is set whenever the CTRL/C combination is typed and can be set when the system is either I/O bound or compute bound. Finally, the most significant bit of the system definable portion is set by a SYS system function call. All system definable bits are cleared when another program is chained or when the system prints READY.

The system definable portion of the priority byte is always less significant than the user definable portion. Therefore, the system definable bits affect priority only within the user definable priority range. If two jobs are running under priority -8, for example, the user who types the CTRL/C combination has a higher priority (that is, priority -6 in this case) than the user who does not type that combination. On the other hand, a third user with priority Ø supersedes two other users whose priorities are -8 and -6.

When a user first logs in on a system, LOGIN is run with priority Ø. LOGIN automatically sets the user's job to priority -8. This is the default priority with which most or all of the jobs are run. Only in unusual circumstances should priorities other than -8 be assigned.

On occasion, the user may want to run a non-urgent program that requires a great deal of computation. If time is not a factor in obtaining results, the privileged user can decrease the job priority to improve efficiency for the other users on the system. Conversely, infrequently used detached programs often have higher priorities (typically priority Ø) since they can be run quickly once, and ignored thereafter.

Run burst is the maximum time a job can run compute bound before another job obtains access to the CPU. Each unit of run burst time is equal to 1/60th of 1/50th of a second, depending on the system's power line frequency. If the system is operating off a 60 Hz power line, one run burst unit equals 1/60th of a second. In that case, six units equal 1/10th of a second, which is the run burst default value.

If a specific job is assigned a run burst of 6 units but does not require that much compute bound time, the system automatically transfers control to the next user before the six units have been used. One tenth of a second is generally considered the best run burst time period to insure efficient overall system operation. If a job is guaranteed to become I/O bound (that is, I/O stalled) after a certain amount of computations, PRIOR can be used to specify a run burst period larger than 6. In many cases, a run burst of 8 units has a significant effect on long computational programs.

The maximum size assigned to a job refers to the available memory space. By specifying the maximum size of a job up to 16K words of memory, a system manager can allow certain jobs to run programs larger than 8K words. This assigned limit does not affect privileged, compiled programs. Thus, a user with a small amount of space can still run system programs that would normally exceed the memory limit.

6.14.1 Running PRIOR

PRIOR is called as follows.

```
RUN $PRIOR
```

The first query line printed is:

```
ENTER ANOTHER JOB NUMBER?
```

If the current job is to be checked, type the CR key alone. If, however, the job to be checked is not the one under which PRIOR is running, type the job number to be considered. A job number less than 1 or greater than the maximum number of assignable jobs returns the error message ILLEGAL JOB NUMBER ENTERED. Only active, running jobs can be referenced; unassigned job numbers return the above error message.

PRIOR now prints the current priority, run burst and maximum size assigned to the specified job. For example:

```
CURRENT STATISTICS ARE
     -8 PRIORITY
      6 RUN BURST
    16K SIZE MAXIMUM
```

The next query line is:

```
ANY CHANGES?
```

If any or all of this information is to be changed, type Y in response to this query. Typing N or the CR key alone automatically ends the program.

If the user indicates that changes are to be made, PRIOR prints a query line for each parameter in turn. If the value assigned to any parameter is not to be changed, type N or the CR key alone to skip to the next query line. When the typed response to the query line is Y, the message CHANGE IT TO? is printed. Type the new specification as shown below in the sample dialog.

```
RUN $PRIOR

'PRIOR' PRIORITY, BURST, MAXIMUM CHANGER

ENTER ANOTHER JOB NUMBER?

CURRENT STATISTICS ARE:
     -8 PRIORITY
      6 RUN BURST
    16K SIZE MAXIMUM

ANY CHANGES?  Y
CHANGE PRIORITY?  Y
        CHANGE IT TO?  -16
CHANGE RUN BURST?  N
CHANGE SIZE MAXIMUM?  Y
        CHANGE IT TO?  8

CURRENT STATISTICS ARE:
     -16 PRIORITY
      6 RUN BURST
      8K SIZE MAXIMUM

ANY CHANGES?  N

READY
```

Once the last query line has been answered, PRIOR prints the new statistics for verification and prints the message:

```
ANY CHANGES?
```

Typing Y allows the user to make additional changes to this information. Typing N or the CR key alone ends the program.

6.15 DISK SYSTEM CATALOG - SYSCAT

The SYSCAT (system catalog) system program prints a current directory listing of any disk. The system manager or a privileged user must specify the file or device on which the information is to be printed. While a standard CAT command prints a listing of the user's files on a device, SYSCAT prints a file listing of all files on a given disk. SYSCAT is called as follows:

```
RUN $SYSCAT
```

The first query line printed is:

```
OUTPUT CATALOG TO?
```

Type the device or file on which the catalog is to be printed. If the CR key is typed alone, SYSCAT prints the information on the user keyboard. If the device specified does not exist, an error message NOT A VALID DEVICE is given. Similarly, if the device specified is not ready for use, an error message DEVICE HUNG OR WRITE LOCKED is given.

The next query line printed is:

```
CATALOG OF?
```

Type the specification of the disk to be examined. If the CR key is typed alone, SYSCAT prints a catalog of the system disk(s). If the specified disk is not on line, an error message DISK PACK IS NOT MOUNTED AT LINE 1ØØ is given. Similarly, specifying a device other than a disk gives the error message: ILLEGAL SYS() USAGE AT LINE 1ØØ. Shown below is a sample dialog.

```
RUN $SYSCAT
OUTPUT CATALOG TO?
CATALOG OF? DKØ:
SYSTEM CATALOG OF DKØ: ON 18-JUL-74 Ø2:14 PM
```

SYSCAT prints the name of the disk, the current date, and time of day and continues with the catalog.

SYSCAT prints each account in the order of its creation date and lists the quota for each account. The number following the word DISK is the effective number of blocks used, as a function of cluster size. For example, if one block is used for a specific file, but the file cluster size is 8, the number of effective blocks used is 8.

The number following the word UFD indicates the user file directory cluster size. The first six columns of information in the catalog listing correspond to the standard CAT printout. The last column indicates the cluster size (in blocks) of each file. After each account listing, SYSCAT prints the number of files and the number of blocks used in the account. For example:

```
2 FILES CLAIMING 7 BLOCKS IN ACCOUNT
```

The following sample of a partial listing of a system disk demonstrates the format SYSCAT employs.

```
RUN $SYSCAT
OUTPUT CATALOG TO?
CATALOG OF?
SYSTEM CATALOG OF SYSTEM DISK(S) ON 25-SEP-74 1Ø:58 AM

ACCOUNT [1,1]              QUOTA Ø            DISK Ø          UFD 16

ACCOUNT [Ø,1]              QUOTA Ø            DISK 4292       UFD 4

BADB   .SYS       Ø          63       29-JUL-74 Ø4-MAR-74 1Ø:ØØ AM 2
RSTS   .CIL       3ØØ        63       19-AUG-74 3Ø-JUL-74 12:1Ø AM 2
SATT   .SYS       1Ø         63       Ø6-AUG-74 3Ø-JUL-74 12:1Ø AM 2
SWAPØ  .SYS       2Ø48       63       28-AUG-74 3Ø-JUL-74 12:1Ø AM 2
SWAP1  .SYS       672        63       3Ø-JUL-74 3Ø-JUL-74 12:1Ø AM 2
SWAP2  .SYS       976        63       2Ø-AUG-74 3Ø-JUL-74 12:1Ø AM 2
OVR    .SYS       28         63       3Ø-JUL-74 3Ø-JUL-74 12:1Ø AM 2
ERR    .SYS       8          63       22-AUG-74 3Ø-JUL-74 12:1Ø AM 2
BUFF   .SYS       12         63       1Ø-SEP-74 3Ø-JUL-74 12:1Ø AM 2
CRASH  .SYS       52         63       23-SEP-74 3Ø-JUL-74 12:1Ø AM 2
 1Ø FILES CLAIMING 41Ø6 BLOCKS IN ACCOUNT

ACCOUNT [1,2]              QUOTA Ø            DISK 1858       UFD 16

LOGIN .BAC        23         124      25-SEP-74 Ø1-AUG-74 Ø9:19 AM 4
LOGOUT.BAC        15         124      25-'C

READY
```

6.16 COMMUNICATING WITH OTHER TERMINALS - PLEASE AND TALK

A user can communicate with the system console terminal or with another user's terminal by using the system programs, PLEASE and TALK, respectively. These programs are discussed individually in the sections that follow.

Both programs have a protection code of <232>, permitting all users to run them. The system manager can change the protection code to <124> so only privileged users can run these programs.

6.16.1 PLEASE Program

PLEASE enables the user to send a message to the system console terminal (KBØ:). This message, along with the user's job number and keyboard number, is broadcast directly to the system console terminal.

PLEASE is called as follows:

```
RUN $PLEASE
```

PLEASE prints the query line shown below:

```
YES!! MAY I HELP YOU?  (END WITH CONTROL/Z)
```

Characters are not transmitted until the CR key is typed, at which point the entire line is broadcast. So type the CR key after each line.

Since each line of the message is broadcast immediately, no prompting characters are printed on the sending terminal between the lines of the message. One line of the message can be typed into the sending terminal while simultaneously another line is being printed out on the system console terminal. If the system console terminal is in use, each line of the message is stored in one of that terminal's buffers and printed at a later time.

Type the CTRL/C or CTRL/Z combination to terminate the program after the last line of the message has been broadcast.

NOTE

Do not type the CTRL/C or CTRL/Z combination to broadcast the last line. Using either of these combinations before typing the CR key terminates the program and the last line is not broadcast.

PLEASE prints THANK YOU. to indicate all lines have been broadcast.

Here is an example of a user broadcasting a message to the console terminal.

```
RUN $PLEASE
YES!! MAY I HELP YOU? (END WITH CONTROL/Z)

I WILL BE OUT OF TOWN FOR THE NEXT TWO WEEKS.
WILL LET YOU KNOW WHEN I RETURN.
                    -- JOHN SMITH

↑Z
THANK YOU.

READY
```

The message printed on the system console terminal automatically includes the sender's job and keyboard numbers as shown below.

```
JOB 9 KB 15: I WILL BE OUT OF TOWN FOR THE NEXT TWO WEEKS.
JOB 9 KB 15: WILL LET YOU KNOW WHEN I RETURN.
JOB 9 KB 15:                    -- JOHN SMITH
```

6.16.2 TALK Program

TALK enables users to broadcast messages, line by line, to each other's terminals. Both the sending and the receiving terminal must be on-line, but no user need be logged in on the receiving terminal.

TALK is called as follows:

```
RUN $TALK
```

The first query line printed is:

TO WHICH KEYBOARD (KB)?

Type the number of the keyboard to which the message is to be sent; then type the CR key. If the receiving terminal is in use, each line of the message is stored in one of that terminal's buffers and printed at a later time.

NOTE

As a general rule, don't broadcast to a terminal in use before consulting with that terminal's operator. An unscheduled message can disrupt an elaborate printout, sometimes destroying hours of work.

Typing the number of a non-existent or off-line terminal does not return an error message. In the case of a non-existent terminal, the program terminates when the CR key is typed. Messages sent to an off-line terminal, however, are broadcast normally, but never received. In this case, typing the CR key does not terminate the program.

TALK prints the following instructions:

YOU MAY PROCEED - CARRIAGE RETURN SENDS THE LINE
'ALTMODE' ('ESCAPE') SENDS AND TERMINATES THE PROGRAM

To send a message, type the CR key after typing each line. Characters are not transmitted until the CR key is typed, at which point the entire line is broadcast. Since each line of the message is broadcast immediately, no prompting characters are printed on the sending terminal between the lines of the message.

The ALT or ESCAPE key, used to terminate the program, can be typed after the last CR key or instead of it. This key prints an ESC character ($) on the sending terminal. When typed instead of the CR key, the ALT or ESCAPE key also prints an ESC character on the receiving terminal. When this happens, the system does not perform a carriage return operation on the receiving terminal.

Shown below is a typical sending terminal dialog between keyboard 12 (sender) and keyboard 9 (receiver).

```
RUN $TALK
TO WHICH KEYBOARD (KB)?  9
YOU MAY PROCEED - CARRIAGE RETURN SENDS THE LINE
'ALTMODE' ('ESCAPE') SENDS AND TERMINATES THE PROGRAM
GEORGE

    PLEASE MOUNT MY DECTAPE WHEN YOU GET A CHANCE.

                                        THANKS...

                                         HERMAN
$
READY
```

The receiving terminal's printout automatically identifies the sending device by enclosing it in asterisks. No ESC character is printed. The actual message is printed next to the identification, as follows:

```
   KB12 ** GEORGE
** KB12 **
** KB12 **     PLEASE MOUNT MY DECTAPE WHEN YOU GET A CHANCE.
** KB12 **
** KB12 **                                    THANKS...
** KB12 **
** KB12 **                                     HERMAN
```

If at this point, keyboard 9 returns a message, it is done as follows:

```
RUN $TALK
TO WHICH KEYBOARD (KB)?  12
YOU MAY PROCEED - CARRIAGE RETURN SENDS THE LINE
'ALTMODE' ('ESCAPE') SENDS AND TERMINATES THE PROGRAM
OKAY, HERMAN...
YOUR DECTAPE IS MOUNTED ON UNIT #2
SORRY FOR THE DELAY!
                                        GEORGE$
READY
```

Notice that the ESC key was typed instead of the final CR key in the above example. This prints the ESC character on both the sending and receiving terminals (see below).

```
** KB9 ** OKAY, HERMAN...
** KB9 **
** KB9 ** YOUR DECTAPE IS MOUNTED ON UNIT #2.
** KB9 ** SORRY FOR THE DELAY!
** KB9 **
** KB9 **                                GEORGE$
```

6.17 INITIALIZING A DISK - DSKINT

The system manager or a privileged user can initialize a disk with the DSKINT system program rather than the DSKINT initialization option. DSKINT writes a minimal file structure on a previously formatted disk. All user files on the disk are destroyed. All disks must be initialized before they can be used on a RSTS/E system.

System disk initialization is a unique process performed by the system and is described in Chapter 3. The procedures described in this section apply only to non-system disks. Each user can initialize his disk without affecting other users on the system.

DSKINT is called by using the RUN command as shown below.

```
RUN $DSKINT
```

DSKINT responds by printing questions requesting specific information. Refer to Chapter 3 for answers to these questions. Shown below is a sample dialog.

```
RUN $DSKINT
DISK ? RK
UNIT ? 1
PACK ID ? MARK
PACK CLUSTER SIZE ? 16
MFD PASSWORD ? LEVY
MFD CLUSTER SIZE ? 16
PUB OR PRI ? PRI

READY
```

Because the DSKINT system program does not perform any pattern checks for bad blocks and cannot format a disk, it is not as versatile as the DSKINT initialization option. Furthermore, the program does not create the storage allocation table. After the disk is initialized by the DSKINT system program, the user must build the SAT by executing the CLEAN command of the UTILTY system program. See Section 6.3.2 for a description of the CLEAN command.

6.18 SYSTEM PROGRAMMING CONVENTIONS

Many RSTS/E system programs are designed to run by methods other than by the RUN command. The following are some of the alternative methods.

a) At a logged out terminal by means of LOGIN. For example, SYS typed at a logged out terminal causes the SYSTAT system program to run.

b) At a terminal by means of a CCL command. For example, the standard CCL command QUE runs the QUE system program.

It is useful for the system manager to be able to duplicate some of these actions in his own utility programs on his system. Also, the system manager can alter system programs to tailor them to local installation needs. The guidelines in the following sections describe how to perform these actions.

6.18.1 Running a Program From a Logged Out Terminal

A program runs from a logged out terminal by means of the LOGIN system program. When the user types characters at a terminal not logged in, the monitor runs LOGIN which compares the characters typed with those it is designed to recognize. LOGIN is designed to accept a command line from a logged out terminal and chain to a system program.

The following discussion employs the SYSTAT system program as an example of coding both LOGIN and a user program to run at a logged out terminal. The monitor runs LOGIN at line 32000 if a line is typed at a terminal not logged into the system. LOGIN extracts the characters typed and compares the leftmost characters typed with commands in a set of DATA statements between lines 32200 and 32299. If a command matches the characters and the third item in the DATA statement is 4, LOGIN puts the command line extracted in the core common area and chains to line 32000 of the program and account specified in the second element of the DATA statement. Thus, for SYS typed at a logged out terminal, LOGIN chains to the SYSTAT program in the system library account. The following statement in LOGIN ensures that action.

```
32280 DATA SYS, $SYSTAT, 4
```

To chain to a certain program, the user can supply a DATA statement in the same format between lines 32101 and 32199 in LOGIN. For example, to run a program named HELP under account [2,2] in response to a command named WHAT, insert into LOGIN.BAS a DATA statement similar to the following. The DATA statement must include an account number since there is no default account when running at a terminal not logged into the system.

```
32190 DATA WHAT, "[2,2]" HELP, 4
```

After LOGIN is compiled and stored in the system library account, the program HELP under account [2,2] starts running at line 32000 whenever WHAT is typed at a logged out terminal.

The program chained to at line 32000 must contain statements which process the information passed to it in core common as a command line. Provision must also be made for resetting variables used as flags and for initiating error handling. Because a job not logged into the system has no project-programmer number, the program chained to cannot assume a default project-programmer number when opening a file. The CHAIN command in the LOGIN system program does not drop the special login privileges afforded by JFHIBY being set. The program to which LOGIN chains can therefore read or write any file on the system because it retains the full privileges. To implement protection, the program itself must perform the protection check. When a logged out job terminates, it can print the BYE message to inform the user that the terminal is still logged out. When the logged out job executes an END statement or a STOP statement, the system immediately removes the program from memory and leaves the user terminal in a logged out state.

6.18.2 Designing a Program to Run by a CCL Command

By means of the CCL command option at system generation time, the system manager can specify unique commands which, when typed on a logged in terminal, load and run BASIC-PLUS programs from the system library account. The monitor maintains a table of such commands and programs and, by convention, chains to a program at line 3ØØØØ.

To pass information to the program, the monitor writes to the core common area the entire line typed at the terminal including the command typed. The user program must extract the data from core

common and parse the entire line typed. This action allows the user to design a program to run by several CCL commands. As a convention, programs supplied by DIGITAL and invoked by standard CCL commands reserve lines 3ØØØØ to 3Ø999 for routines which parse the command line, check for errors, and dispatch to routines within the program.

6.18.3 Changing LOGIN to Set a Different Swap Maximum

The LOGIN system program sets the swap maximum to 8K words for all users except those whose project numbers are one. This action means that privileged users run with a swap maximum of 16K words. Since, on many systems, programs must run under non-privileged accounts in job areas larger than 8K words, it is necessary to modify LOGIN to set a swap maximum larger than 8K words.

To modify the LOGIN.BAS program, the user must alter the J% = 8% statement in the first physical line of the multiple statement line at line number 15Ø1Ø and compile the program on the system library account. The following statement sets the priority, run burst, and swap maximum factors.

```
15Ø1Ø  J% = 8%
       : J% = 16% IF (A% AND -256%) = 256%
       : I$ = SYS(CHR$(6%)+CHR$(-13%)+CHR$(-1%)+
                  CHR$(-1%)+CHR$(-2%)+
                  CHR$( Ø%)+CHR$( 6%)+
                  CHR$(-1%)+CHR$( J%))
       : RETURN
```

Change the value 8% in the statement J% = 8% to any value less than or equal to the current default swap maximum used at system start-up time. Compile the program on the system library account with protection code <6Ø> as follows.

```
COMPILE SYØ:LOGIN$
READY
```

It is recommended that the system manager not replace the original source file LOGIN.BAS with the modified version.

6.19 DOS/BATCH DISK FILE UTILITY PROGRAM - DOSPIP

The DOSPIP program copies files from a DOS/BATCH disk cartridge or prints a directory listing of files stored under a specified user identification code (UIC) on a disk cartridge.

NOTE

DOSPIP can not write files on a DOS disk cartridge.

Since the program is stored in the system library with a protection code of 124, only privileged users can run it. To run DOSPIP, mount a DOS/BATCH disk cartridge in an RK11 drive unit, write protect the drive, and type the following command.

```
RUN $DOSPIP
DOSPIP  V5B-3

DOS DISK IS ON DRIVE?
```

The program runs and prints its header and a question requesting the unit number of the drive on which the cartridge is mounted. After the unit number is typed, the program requests the type of operation by printing the following question.

```
FUNCTION?
```

To request a function, type one of the responses described in Table 6-17. If an invalid response is typed, DOSPIP prints the following text.

```
FUNCTIONS ARE LIST AND COPY

FUNCTION?
```

To terminate DOSPIP, type EXIT in response to the FUNCTION query. The following sections describe the different functions available under DOSPIP.

Table 6-17
DOSPIP Functions

Function	Responses	Meaning
Copy	C Typing RETURN key only.	Copy a DOS/BATCH file to a RSTS medium.
List	L	Print a DOS/BATCH directory listing.
Exit	E	Terminate execution and exit to the RSTS/E monitor.

6.19.1 Copying a DOS/BATCH Disk File - COPY

To copy a DOS/BATCH disk file, type COPY or anything beginning with C in response to the FUNCTION question. DOSPIP prints the following additional questions to determine the DOS user identification code, the input and output file names, and the output file cluster size.

```
FUNCTION? COPY
DOS UIC?  1,1
FILE NAME? ABC.PAL
OUTPUT FILE TO? ABC.PAL
CLUSTER SIZE?
DETACH? NO

FUNCTION?
```

In response to the DOS UIC question, type the UIC under which the file to be copied is stored. If the user-group number and the user's number within the group are not octal numbers between 1 and 376, DOSPIP prints the error message ILLEGAL UIC and reprints the FUNCTION question. If the UIC specified is not on the disk, the program prints the error message UIC NOT FOUND and reprints the FUNCTION question.

In response to the question FILE NAME, type the name and extension of the file to be copied. If the file does not exist under the specified UIC, the program prints the error message FILE NOT FOUND and reprints the FUNCTION question. If DOSPIP finds the file, it prints the OUTPUT FILE TO question.

To specify the destination of the file to be copied, type the file specification in response to the OUTPUT FILE TO question. DOSPIP then prints the CLUSTER SIZE question to determine the cluster size for the output file. If either Ø or nothing is given for the cluster size, DOSPIP uses the pack cluster size of the output disk. After the user types the response to the CLUSTER SIZE question, the program creates a file with the name, extension, and cluster size specified. This action destroys any existing file with the same name and extension.

The response to the DETACH question allows the user to execute the copy operation as a detached job. Typing Y (or YES) causes DOSPIP to detach from the terminal, execute the copy operation, and enter the hibernate (HB) state upon completion of the copy operation. Any other response not beginning with Y leaves DOSPIP attached to the terminal and starts the copy operation. DOSPIP reprints the FUNCTION question at the terminal when the copy operation terminates.

6.19.2 Creating a DOS/BATCH Directory Listing - LIST

To create a directory listing, type LIST or anything beginning with L in response to the FUNCTION question. DOSPIP prints the following questions to determine the DOS user identification code and the output medium.

```
FUNCTION? LIST
DOS UIC? 1,1
OUTPUT TO? DOS.LST

FUNCTION?
```

In response to the DOS UIC question, type the UIC of the directory. If the group number and the user number are not octal numbers between 1 and 376, DOSPIP prints the error message ILLEGAL UIC and reprints the FUNCTION question. If the UIC specified is not on the disk, the program prints the error message UIC NOT FOUND and reprints the FUNCTION question.

In response to the OUTPUT TO question, type the specification of the medium on which DOSPIP is to create the listing. If the specification designates a disk file, the program creates a file with the name and extension given and destroys any existing file with the same name and extension. The disk file can be later queued for output to

a line printer. To print the listing at the current keyboard, type KB: in response to the OUTPUT TO question. After creating the listing, DOSPIP reprints the FUNCTION question. The listing created is in standard DOS/BATCH format.

CHAPTER 7
SYS SYSTEM FUNCTION CALLS AND THE PEEK FUNCTION

7.1 GENERAL SYS SYSTEM FUNCTION CALLS

SYS system function calls allow a user program to perform special I/O functions, to establish special characteristics for a job, to set terminal characteristics, and to cause the monitor to execute special operations.

The specified SYS format is employed for two reasons. One, the calls are unique to the BASIC-PLUS implementation of the BASIC language. As such, the calls are system dependent and have calling format different from any BASIC language call. Second, the SYS format allows the usage of a variable number of parameters.

SYS calls are separated into two classes - privileged and non-privileged. The privileged calls can be used only by a privileged user as defined in Section 5.5 or by a privileged program as defined in Section 5.5.2. The non-privileged calls can be used by anyone and are completely safe in the sense that their misuse can do no damage to other programs.

7.1.1 SYS System Function Formats and Codes

The general format of the SYS call is as follows:

```
V$ = SYS(CHR$(F) + O$)
```

where:

V$ is the target string returned by the call

F is the SYS system function code

O$ is the optional (by function code) parameter string passed by the call

Function codes denoted by F in the general format are from zero through nine, inclusive. SYS calls which specify a code outside of these numbers or which pass a zero length string generate the ILLEGAL SYS() USAGE error (ERR = 18). The following list summarizes the codes and their usages. The subsequent pages describe the usage, calling format, and purpose of the calls.

Function Code(F)	Usage
Ø	Cancel ↑O effect on console terminal
1	Enter tape mode on console terminal
2	Enable echoing on console terminal
3	Disable echoing on console terminal
4	Enable single character input mode (ODT submode) on console terminal
5	Exit to editor with no READY message
6	SYS call to the file processor
7	Get core common string
8	Put core common string
9	Exit to editor and set up NONAME program

7.1.2 General SYS System Function Calls

7.1.2.1 Cancel ↑O Effect on Console Terminal — Not Privileged (F=Ø)

Data Passed:

Byte(s)	Meaning
1	CHR$(Ø%), the cancel ↑O effect code
2	Optional. If specified, CHR$ (N%) where N% is the number of the channel on which the system executes the call.

Data Returned: The target string is equivalent to the passed string.

Discussion:

This call cancels the effect of the user typing a CTRL/O combination at the program's console terminal. If the call is in the form SYS(CHR$(Ø%) + CHR$(N%)), the system performs the action on the terminal open on channel N%. See Section 3.7 of the RSTS-11 System User's Guide for a description of the CTRL/O combination.

7.1.2.2 Enter Tape Mode on Console Terminal Not Privileged (F=1)

Data Passed:

Byte(s)	Meaning
1	CHR$(1%), the enter tape mode code
2	Optional. If specified, CHR$(N%) where N% is the number of the channel on which the system executes the call.

Data Returned: The target string is equivalent to the passed string.

Discussion:

The action of this call is the same as that of the TAPE command described in the RSTS-11 System User's Guide. If the call is in the form SYS(CHR$(1%) + CHR$(N%)), the system performs the action on the terminal open on channel N%.

7.1.2.3 Enable Echoing on Console Terminal Not Privileged (F=2)

Data Passed:

Byte(s)	Meaning
1	CHR$(2%), the enable echoing code
2	Optional. If specified, CHR$(N%) where N% is the number of the channel on which the system executes the call.

Data Returned: The target string is equivalent to the returned string.

Discussion:

This code cancels the effect of SYS calls with codes 1 and 3. If the form of the call is SYS(CHR$(2%) + CHR$(N%)), the action is performed on the terminal open on channel N%.

7.1.2.4 Disable Echoing on Console Terminal — Not Privileged (F=3)

Data Passed:

Byte(s)	Meaning
1	CHR$(3%), the disable echoing code
2	Optional. If Specified, CHR$(N%) where N% is the number of the channel on which the system executes the call.

Data Returned: The target string is equivalent to the passed string.

Discussion:

This call prevents the system from echoing information typed at the console terminal. As a result, information such as a password is kept secret but accepted as valid input by the system. If the form of the call is SYS(CHR$(3%) + CHR$(N%)), the action is performed at the terminal open on channel N%.

7.1.2.5 Enable Single Character Input Mode (ODT Submode) on Console Terminal — Not Privileged (F=4)

Data Passed:

Byte(s)	Meaning
1	CHR$(4%), the enable single character input mode code
2	Optional. If specified, CHR$(N%) where N% is the number of the channel on which the system executes the call.

Data Returned: The target string is equivalent to the passed string.

Discussion:

Allows a single character to be accepted as input from the terminal. Normally, the system waits until a line terminated by a RETURN, LINE FEED, FORM FEED, or ESCAPE character has been typed before accepting input. In single character mode, a character typed at the terminal is passed immediately to the program by the next keyboard input request statement without waiting for the delimiting character.

This function must be enabled prior to every input request statement that passes a single character to the program. A GET Statement is used as the input request statement. (INPUT or INPUT LINE statements cause repeated generation of the input request until a line terminator is detected and, therefore, must not be used.)

If a program performs other lengthy operations before it executes either another SYS call and GET statement or other input/output operation at the terminal, it allows time for the user to type more than one character. To provide for such a possibility, the program should examine the system variable RECOUNT after executing each GET statement. This procedure determines how many characters the user typed between keyboard input operations and enables the program to process all the characters without losing any.

Since this function is used in the system program ODT.BAS, it is sometimes referred to as "ODT submode". If the form of the call is SYS(CHR$(4%) + CHR$(N%)), the action is performed on the terminal open on channel N%.

7.1.2.6 Exit to Editor with no READY Message

Not Privileged (F=5)

Data Passed:

Byte(s)	Meaning
1	CHR$(5%), the exit with no READY code

Data Returned: No data is returned.

Discussion:

This call causes the same effect as the END statement, except that it can appear anywhere in the program and does not cause a READY message to be printed.

7.1.2.7 FIP Function Call

Both Privileged and (F=6) Not privileged

See Section 7.2 for a description of SYS calls to the file processor.

7.1.2.8 Get Core Common String — Not Privileged (F=7)

Data Passed:

Byte(s)	Meaning
1	CHR$(7%), the get core common string code

Data Returned: The target string is the contents of the core common area.

Discussion:

Allows a program to extract a single string from a data area loaded by another program previously run by the same job. The data area is called the core common and is from Ø to 127 8-bit bytes long. Refer to SYS function code 8.

7.1.2.9 Put Core Common String — Not Privileged (F=8)

Data Passed:

Byte(s)	Meaning
1	CHR$(8%), the put core common string code
2-128	The string to put in the core common area.

Data Returned: The target string is the passed string.

Discussion:

Allows a program to load a single string in a common data area called core common. This string can be extracted later by another program, running under the same job and called via the CHAIN statement. The string is from Ø to 127 8-bit bytes long. If the string to be put into the core common area is longer than 127 bytes, the system sets the length of the core common string to Ø. Refer to SYS function code 7.

This function provides a means for passing a limited amount of information when a CHAIN statement is executed. If a larger amount of information is to be passed, it must be written to a disk file and read back by the later program.

7.1.2.10 Exit to Editor and Set Up NONAME Program — Not Privileged (F=9)

Data Passed:

Byte(s)	Meaning
1	CHR$(9%), the exit and set up NONAME code

Data Returned: The target string is equivalent to the passed string.

Discussion:

This function causes the same actions as the END statement placed in the code, and, in addition, clears the program out of memory. This is the proper method of stopping a program that is not to be rerun. Also, the same action is performed by the command NEW NONAME.

7.2 SYS SYSTEM FUNCTION CALLS TO FIP (FUNCTION CODE 6)

The SYS function call whose code is 6 is a more specialized case of the general system function call. It is specialized by an additional code called the FIP code. The FIP code causes a dispatch call to be made to special resident or non-resident code that performs File Processing.

The format of the call is:

```
V$ = SYS(CHR$(6%) + CHR$(FØ) + O$)
```

where:

V$ is the data (target) string returned by the call
FØ is the FIP function code
O$ is the optional (by function) parameter string.

The general format of the target variable (V$) is:

Byte(s)	Meaning
1	Job number times 2
2	Value of Internal Function called (meaningless to general user)
3-30	Data returned

NOTE

Thirty bytes are always passed back.
Unused bytes are either internal data or Ø.

The proper use of the FIP system function call requires that the user program build a parameter string to pass and that the program later extract the data from the returned string, called the target string. Each call returns a string of 30 bytes, each byte (or character) of which can or can not contain useful information. The descriptions of the FIP codes specify the contents of each useful byte in the string, from which the user determines whether the information contained is of interest.

7.2.1 Building a Parameter String

Some FIP calls require no parameters except the function and subfunction codes; other FIP calls require either variable length parameter strings or very simple parameter strings. For such FIP calls, it is usually easiest to set up and execute the function call in a single statement. The following sample statements show the procedure.

```
A$ = SYS(CHR$(6%) + CHR$(-7%))
         !ENABLE CTRL/C TRAP
         !(NO PARAMETER STRING)

A$ = SYS(CHR$(6%) + CHR$(-1Ø%) +
          "DKØ:FILE.EST")
         !FILE NAME STRING SCAN
         !(VARIABLE LENGTH
         !  PARAMETER STRING)

A$ = SYS(CHR$(6%) + CHR$(-8%) + CHR$(1%))
         !FCB/DDB INFORMATION
         ! FOR FILE OPEN ON
         ! CHANNEL 1
         ! (SIMPLE PARAMETER
         ! STRING)
```

Many FIP calls require more complex data formats. For example, the kill a job FIP call, FØ = 8, requires byte 3 to be the job number to kill, byte 27 to be Ø, and byte 28 to be 255. A recommended method of building the complex parameter string to pass to a function is to dimension a 3Ø byte list (an integer array) and set the items in the list to values which map into those required in the parameter string format. The list can later be set to a character string by the CHANGE statement before it is passed as the parameter string of the FIP system function call. The resulting character string is in proper format and contains the correct byte values so that it can be placed as the parameter string of the FIP system function call. For example,

```
1Ø  DIM A%(3Ø%)              !the string is 3Ø bytes
2Ø  J% = 4%                  !kill job number 4
3Ø  A%(Ø%) = 3Ø%             !Øth element is length of list
4Ø  A%(1%) = 6%              !SYS call code 6 (FIP call)
5Ø  A%(2%) = 8%              !FIP code 8 (kill job)
6Ø  A%(3%) = J%              !job number to kill
7Ø  A%(27%) = Ø%             !this byte must be Ø
8Ø  A%(28%) = 255%           !this byte must be 255
```

Following the code which builds the list is the CHANGE statement and the call itself, as shown below.

```
1ØØ  CHANGE A% TO A$         !generates character
          .                  !string from the
          .                  !integer list

2ØØ  B$ = SYS(A$)            !invoke system
                             !function call
```

7.2.2 Unpacking the Returned Data

In the case of the FIP call using code 8, above, the action performed (kill a job) rather than the data returned is of importance. However, many FIP calls return a data string which is the primary interest of the user. In such a case, the data in the string must be unpacked.

As in the case of building the parameter string, there are two recommended methods of unpacking the returned string. If the user needs only a few pieces of the data, it may be easiest to operate directly on the returned string. For example, if the user wants only the 4-byte Radix-50 representation of a 6-byte string, the filename string scan FIP call (FIP code -1Ø) can be used as follows:

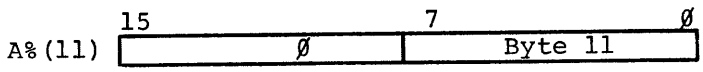
```
A$ = MID(SYS(CHR$(6%) + CHR$(-1Ø%) + S$), 7%, 4%)
```

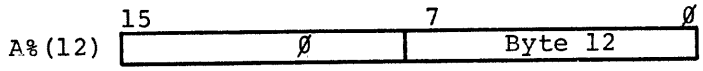
to extract bytes 7 through 1Ø of the returned string. To extract numeric data, ASCII or CVT$% functions can be used.

In some cases, many pieces of the returned data are needed. In other cases, the string returned by the FIP call is needed later to set up another FIP call. In such cases, the user program can transform the returned string to a 3Ø-byte list using a CHANGE statement,

```
CHANGE A$ TO A%
       or
CHANGE SYS(...) TO A%
```

When the returned string has been converted in this manner, it is necessary to do further conversions in order to get numeric data into a usable form. Take, for example, the data returned by a FIP code of 15 (directory lookup on index). The layout of the data returned specifies that bytes 11 and 12 are the filename extension encoded in Radix-50 format. In order to convert those bytes into an ASCII string, to OPEN the file for example, the RAD$ BASIC-PLUS function must be used on the two bytes converted to a single integer. The integer representation of each byte, however, occupies a full word, 16 bits in length. Thus, list items number 11 and 12 appear as the following:

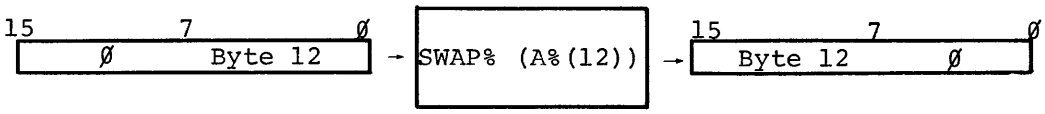




A%(11) contains the low byte portion of the Radix-50 word; A%(12) contains the high byte portion of the Radix-50 word. The two bytes must be combined into a single word and converted to the proper character string representation. This is accomplished by the following:

```
S$ = RAD$(A%(11) + SWAP%(A%(12)))
```

The SWAP% function reverses the bytes (the low byte takes the high byte position and vice-versa) in an integer word. Graphically, the operation appears as follows:



Thus, byte 12 takes the high byte position in the word. The two words are then combined by the + operator to form one word. The RAD$ function performs the conversion on that one integer word to produce the 3-character string representation of the file name extension. The character string is assigned to the character variable S$ and is in ASCII format.

To convert a longer string from Radix-50 to ASCII format, the above procedure must be applied to each two bytes in the string. For example, the filename from FIP call 15 is returned in bytes 7 through 1Ø. In order to convert these bytes to ASCII format, use the following routine.

```
A$ = RAD$(A%(7%) + SWAP%(A%(8%))
B$ = RAD$(A%(9%) + SWAP%(A%(1Ø%))
F$ = A$ + B$
```

or, in a single statement,

```
F$ = RAD$(A%(7%) + SWAP%(A%(8%))) +
     RAD$(A%(9%) + SWAP%(A%(1Ø)))
```

7.2.3 Notation and References Used in FIP Call Descriptions

Many FIP calls require that a project-programmer number (PPN) be specified in the calling string, and several return a PPN. Where such is the case, the PPN field is in the following general form.

```
Bytes X-(X+1)        PPN
```

The value X is odd. The intended meaning of this notation is that byte X in the string holds the programmer number, and byte (X+1) holds the project number. For example, to set up a FIP call to zero an account on a disk (FIP code 13), the calling format shows

```
Bytes 5-6            Project-programmer number
```

If the call is to be set up in a 3Ø-entry list, A%, then the format requires that

```
A%(5%) = programmer number
A%(6%) = project number
```

Many of the FIP calls described in this chapter return or require integer data in two (consecutive) bytes of the returned data string. When this is the case, the field in the returned string is described in the format:

```
Bytes X-(X+1)        integer value
```

If the return string is to be processed directly (that is, without changing it to an integer array), then the integer value of the two bytes can be obtained using the following statement.

```
I% = SWAP%(CVT$%(MID(A$,X,2%)))
```

where A$ holds the returned string. If the returned data string is first transferred to an integer array, A%, using the CHANGE statement, then the integer value can be obtained using the statement below.

```
I% = A%(X) + SWAP%(A%(X%+1%))
```

For example, the get monitor tables (part I) FIP call (FIP code-3) returns the address of monitor's job table in bytes 11 and 12. If A$ holds the returned string, then either of the following two routines put the address of the job table into the integer variable I%.

```
I% = SWAP%(CVT$%(MID(A$,11%,2%)))
```

or

```
CHANGE A$ TO A%
I% = A%(11%) + SWAP%(A%(12%))
```

In some integer fields in the FIP calls, the value is a full 16-bit unsigned integer between Ø and 65535. The sign bit indicates an extra power of two rather than positive or negative. Since an integer value in BASIC-PLUS is between -32768 and +32767, any value greater than 32767 must be stored as a floating point value. Assume that in some SYS call, an unsigned integer is returned in bytes 5 and 6, and that the returned string has been changed to an array, A%. As always, the high byte of the integer is in byte 6, the low byte in byte 5. The statement

```
Q = 256.*A%(6%) + A%(5%)
```

gets the full 16 bit value into the floating point variable, Q. Q is always positive. Note that replacing the 256.* with SWAP%() causes the expression to be first evaluated as a normal integer expression, and then changed to a floating point value. The resulting value is between -32768 and +32767. The 256.* forces the expression to be evaluated as a floating point number.

To convert an unsigned integer to two bytes to pass to a SYS call also requires special processing. Assuming that Q holds the unsigned value, and that the value is to be placed in A%(5%) (low order) and A%(6%) (high order), then the most direct method of transformation is:

```
A%(6%) = Q/256.
A%(5%) = Q-A%(6%)*256.
```

On PDP-11 computers without FIS or FPP (floating-point hardware), division operations are relatively slow. On these machines, a faster method is:

```
10 IF Q<32768.THEN Q% = Q
   ELSE IF Q = 32768.  THEN Q% = 32767% + 1%
   ELSE    Q = Q-65536.
20 A%(5%) = Q% AND 255%
30 A%(6%) = SWAP% (Q%) AND 255%
```

The disadvantage of this second method is that it requires more code.

The filename string scan SYS function (FIP code-1Ø) is useful as a "front-end" for many FIP functions. Most of the FIP calls which require device or filename information in their parameter strings expect information in the format which the FIP-1Ø call returns it. For example, FIP code 17, lookup a file by name, expects its calling string to be passed in exactly the same format as that returned by the FIP-1Ø call, with a change of only four data bytes. The following routine sets up and executes the lookup call on the file DKØ:INVENT.DAT[1Ø,2Ø] using the filename string scan FIP call.

```
10 DIM A%(30%)                                  ! SET UP LIST ARRAY.
20000 A$="DK0:INVENT.DAT[10,20]"                ! SET UP VARIABLE.
20010 CHANGE SYS(CHR$(6%)+CHR$(-10%)+A$) TO A%
                                                ! DO THE FNS CALL TO
                                                ! SET UP ARRAY PROPERLY.
20020 A%(0%)=30%                                ! SET UP FOR 30 LONG.
20030 A%(1%)=6%                                 ! THIS IS A FIP CALL.
20040 A%(2%)=17%                                ! DISK LOOKUP BY NAME.
20050 A%(3%),A%(4%)=0%                          ! SET INDEX TO ZERO.
20060 CHANGE A% TO A$                           ! SET UP AS A STRING
20070 CHANGE SYS(A$) TO A%                      ! AND DO THE CALL.
32767 END
```

In order to avoid redundancy in the descriptions in Section 7.2, any field for any of the calls which are either passed to or returned from the function in the same format as that returned by FIP code-1Ø are identified by a + superscript after the field specification. For a detailed explanation of fields so identified, see Section 7.2.4.1.

Table 7-1 is a quick reference index of the FIP functions in order of FIP code (FØ). For detailed information on each of the functions, refer to the Section shown beside the name in the table.

Table 7-1
FIP SYS Calls (By Sub-function Code)

FUNCTION CODE (FØ)	PRIVILEGED STATUS	FUNCTION NAME	SECTION
-23	No	Terminating file name string scan	7.2.4.1
-22	Yes	Set special run priority	7.2.6.2
-21	Yes	Drop (temporary) privileges	7.2.6.4
-2Ø	Yes	Lock/unlock job in core	7.2.6.3
-19	Yes	Set number of logins	7.2.14
-18	*** Reserved ***		
-17	*** Reserved ***		
-16	Yes	System shutdown	7.2.5.1
-15	Yes	Accounting dump	7.2.15.2
-14	Yes	Change system date/time	7.2.5.2
-13	Yes	Change priority/run burst/job size	7.2.6.1
-12	No	Get monitor tables - Part II	7.2.17.2
-11	Yes	Change file backup statistics	7.2.9
-1Ø	No	Filename string scan	7.2.4.1
-9	Yes	Hangup a dataset	7.2.5.3
-8	No	FCB/DDB Information	7.2.17.3
-7	No	CTRL/C Trap enable	7.2.4.7
-6	Yes(1)	Poke core	7.2.13
-5	Yes	Broadcast to terminal	7.2.5.4
-4	Yes	Force input to terminal	7.2.5.5
-3	No	Get monitor tables - Part I	7.2.17.1
-2	Yes	Disable logins	7.2.5.6
-1	Yes	Enable logins	7.2.5.7
Ø	Yes	Create user account	7.2.7.1
1	Yes	Delete user account	7.2.7.2
2	Yes	Clean up a disk pack	7.2.5.9
3	Yes	Disk packs	7.2.5.8
	Yes	TTYSET	7.2.8
4	Yes(2)	Login	7.2.1Ø.1
5	Yes(2)	Logout	7.2.1Ø.2
6	Yes(2)	Attach	7.2.11.2
	Yes	Reattach	7.2.11.3
7	Yes	Detach	7.2.11.1
8	Yes	Change password/quota	7.2.5.1Ø
	Yes	Kill job	7.2.5.11
	Yes	Disable TTY	7.2.5.12
9	No	Return error messages	7.2.4.2
1Ø	No	Assign/reassign device	7.2.4.3
11	No	Deassign device	7.2.4.4
12	No	Deassign all devices	7.2.4.5
13	Both	Zero a device	7.2.4.6
14	Both	Read or Read and Reset Accounting Data	7.2.15.1
15	No	Directory lookup on index	7.2.16.1
	No	Special magtape directory lookup	7.2.16.2
16	Yes	Set terminal characteristics	7.2.8
17	No	Disk directory lookup on filename	7.2.16.3
	No	Disk wildcard directory lookup	7.2.16.4
18	No	Send a message	7.2.12.2
	Yes	Declaring a receiver and receiving a message	7.2.12.1
	Yes	Remove from receive table	7.2.12.3
--	Yes	PEEK function	7.3

(1) Poke core can be executed only from account [1,1].
(2) These functions require login privileges (JFHIBY).

The privileged status column indicates whether the SYS call can be used only by a privileged user or by any user. A non-privileged user who attempts to call a privileged SYS function always receives the ILLEGAL SYS() USAGE error (ERR = 18). The notation BOTH in the privileged status column indicates that some facilities of the specified function are available to a non-privileged user, while the privileged user has a more powerful set.

7.2.4 General Utility SYS Calls to FIP

The SYS calls to the file processor described in this section are available to both privileged and non-privileged users.

7.2.4.1 File Name String Scan

Not Privileged (FØ=-1Ø) (FØ=-23)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-1Ø), the filename string scan code. CHR$(-23) is the same as CHR$(-1Ø) except that the scan terminates on certain characters. See discussion.
3-?	Character string to scan; can be any length.

Data Returned: Sets the STATUS variable and returns the following.

Byte(s)	Meaning
1-4	Internal coding
5-6	Project-programmer number (Ø means the current account)
7-1Ø	File name in Radix-50 format
11-12	File name extension in Radix-5Ø format
13-20	Not used
21	If no protection code is found, this byte is Ø. Otherwise, this byte is 255 and byte 22 contains the protection
22	Protection code when byte 21 is 255
23-24	Device name in ASCII format or Ø if none is found. For example, DK and DT
25	Device unit number if byte 26 is 255
26	If this byte is Ø, no explicit unit number was found for the device. If this byte is 255, the value in byte 25 is the explicitly specified device unit number. The 255 value here indicates that a zero in byte 25 is explicitly unit Ø of the device.
27-28	First flag word. See discussion.
29-30	Second flag word. See discussion.

Possible Errors:

Meaning	ERR Value
ILLEGAL FILE NAME The character string scanned contains unacceptable characters. See Section 9.1 of the BASIC-PLUS Language Manual for a description of a file specification.	2

Discussion:

The file name string scan function determines specific file syntax information (for example, whether a given file name is valid or whether a certain disk is mounted) and returns information in the format required for all other file and device related SYS calls. In addition, it provides a means of packing a string in Radix-50 format.

The STATUS variable is set for the device type found in the string scanned.

The following example demonstrates how a string can be converted to Radix-5Ø format by a user defined function and the file name string scan SYS call.

```
10  DEF FNPØ$(A$) = MID (SYS(CHR$(6%)+
      CHR$(-1Ø%)+A$),7%,4%)
      ! PACK 6 CHARACTERS TO RADIX-5Ø
```

The function FNPØ$ returns a 4 character string which is the Radix-5Ø representation of the first 6 characters of A$. (Note that no error handling is included and that errors can occur.) The file name string scan call is the only function in BASIC-PLUS which packs a string in Radix-5Ø format. To pack strings longer than six characters, the user must make multiple calls to the SYS function.

The two words in bytes 27 and 28 and in bytes 29 and 3Ø hold easily accessible flags indicating exactly what fields in the source string were found and what kind of information they contained. For the purposes of the discussion, it is assumed that the returned string was converted by a CHANGE statement to an integer array, M%(3Ø%). The flag words are then created by doing the proper arithmetic operations on the bytes, as shown:

```
flag word 1:  SØ% = M%(27%)+SWAP%(M%(28%))
flag word 2:  S1% = M%(29%)+SWAP%(M%(3Ø%))
```

Once these two words are created, the data coded into them is accessible by means of an AND operation between the word and the bit relating to a particular piece of information (see Section 12.3.7.7 of the BASIC-PLUS Language Manual for information about the AND operation). Each bit of the PDP-11 word can be used to hold a YES or NO answer. In the case of SØ%, only the high-order 8 bits are used. In the case of S1%, all 16 bits are used.

Flag word 1 is redundant; that is, all information returned in flag word 1 is also in flag word 2. Flag word 2 holds much more information than flag word 1.

In the following discussion, it is assumed that bytes 27 and 28 have been put into SØ% and bytes 29 and 3Ø have been put into S1% as described above.

Flag word 1: where SØ% = M%(27%)+SWAP%(M%(28%))

Bit	Comparison	Meaning
8	SØ% AND 256%<>Ø%	Filename was found in the source string (and is returned in Radix-50 format in bytes 7 through 1Ø)
9	SØ% AND 256% = Ø%	No filename found
	SØ% AND 512%<>Ø%	A dot was found in source string
	SØ% AND 512% = Ø%	No dot was found in source string implying that no extension could have been specified either
10	SØ% AND 1Ø24%<>Ø%	A project-programmer number was found in source string
	SØ% AND 1Ø24% = Ø%	No project-programmer number was found
11	SØ% AND 2Ø48%<>Ø%	A left angle bracket (<) was found in source string implying that a protection code was found.
	SØ% AND 2Ø48% = Ø%	No left angle bracket (<) was found (no protection was specified)
12	SØ% AND 4Ø96%<>Ø%	A colon (but not necessarily a device name) was found.
	SØ% AND 4Ø96% = Ø%	No colon was found implying that no device could have been specified
13	SØ% AND 8192%<>Ø%	Device name was specified and specified device name was a logical device name.
	SØ% AND 8192% = Ø%	Device name (if specified) was an absolute (non-logical) device name (if device name was not specified, this will be Ø)
15	SØ%<Ø%	Source string contained wild card characters (either? or * or both) in filename, extension or project-programmer number fields. In addition, the device name specified, though a valid logical device name, possibly does not correspond to any of the logical device assignments currently in effect. The user program must extract the device name and attempt to access the unit. (See Section 7.2.16.4 for a description of wild card file specifications.)

Flag Word 2: where S1% = M%(29%)+SWAP%(M%(3Ø%))

Bit	Comparison	Meaning
Ø	S1% AND 1%<>Ø%	File name was found in the source string
	S1% AND 1% = Ø%	No file name was found (and the following 2 comparisons return Ø)
1	S1% AND 2%<>Ø%	File name was an * character and is returned in bytes 7 through 1Ø as the Radix-50 representation of the string "??????".
	S1% AND 2% = Ø%	File name was not an * character
2	S1% AND 4%<>Ø%	Filename contained at least one ? character
	S1% AND 4% = Ø%	Filename did not contain any ? characters
3	S1% AND 8%<>Ø%	A dot (.) was found
	S1% AND 8% = Ø%	No dot was found implying that no extension was specified (and the following three comparisons return Ø)
4	S1% AND 16%<>Ø%	An extension was found (that is, the field after the dot was not null)
	S1% AND 16% = Ø%	No extension was found (the field after the dot was null - the following two comparisons return Ø)
5	S1% AND 32%<>Ø%	Extension was an * character and is returned in bytes 11 and 12 as the Radix-50 representation of the string "???"
	S1% AND 32% = Ø%	Extension was not an * character
6	S1% AND 64%<>Ø%	Extension contained at least one ? character
	S1% and 64% = Ø%	Extension did not contain any ? characters
7	S1% AND 128%<>Ø%	A project-programmer number was found
	S1% AND 128% = Ø%	No project-programmer number was found (the following two comparisons return Ø)
8	S1% AND 256%<>Ø%	Project number was an * character (that is, the project-programmer number was of the form [*, PROG]) and is returned in byte 6 as 255
	S1% AND 256% = Ø%	Project number was not an * character

Bit	Comparison	Meaning
9	S1% AND 512%<>Ø%[1]	Programmer number was an * character (that is, the project-programmer number was of the form [PROJ,*]) and is returned in byte 5 as 255.
	S1% AND 512% = Ø%	Programmer number was not an * character
1Ø	S1% AND 1Ø24%<>Ø%	A protection code was found
	S1% AND 1Ø24% = Ø%	No protection code was found
11	S1% AND 2Ø48%<>Ø%	The protection code currently set as default by the current job was used.
	S1% AND 2Ø48% = Ø%	The assignable protection code was not used (protection code given is either the system default, 6Ø, or that found in the source string)
12	S1% AND 4Ø96%<>Ø%	A colon (:), but not necessarily a device name, was found in the source string.
	S1% AND 4Ø96% = Ø%	No colon was found (no device could have been specified); the following 3 comparisons return Ø.
13	S1% AND 8192%<>Ø%	A device name was found
	S1% AND 8192% = Ø%	No device name was found; the following 2 comparisons return Ø.
14	S1% AND 16384%<>Ø%	Device name specified was a logical device name.
	S1% AND 16384% = Ø%	Device name specified was an actual device name.
15	S1%<Ø%	The device name specified was logical and is not assigned to some actual device; the logical name is returned in bytes 23 through 26 as a Radix-50 string.
	S1%>=Ø%	The device name specified (if any was either an actual device name, or a logical device name to which a value has been assigned; the actual device name is returned in bytes 23 through 26 as described above.

[1]Note that if the project-programmer number was of the form [*,*], then both bit 8 and bit 9 of the data byte returned are non-zero values.

Since flag word 2 contains all the information returned in flag word 1 plus more information, it is the more useful of the two words. The following program uses this word. It prints out a list of all the bits returned in the word.

```
5 DIM M%(30%)                                   ! SET UP AN ARRAY TO RETURN TO
10 PRINT "STRING TO SCAN";
20 INPUT LINE S$
30 S$=CVT$$(S$,-1%)                             ! GET RID OF GARBAGE BYTES
40 CHANGE SYS(CHR$(6%)+CHR$(-10%)+S$) TO M%
50 S1%=M%(29%)+SWAP%(M%(30%))
100 IF S1% AND 1%         THEN    PRINT "FILENAME FOUND"
110 IF S1% AND 2%         THEN    PRINT "FILENAME WAS AN *"
120 IF S1% AND 4%         THEN    PRINT "FILENAME HAD '?'S"
130 IF S1% AND 8%         THEN    PRINT "DOT (.) FOUND"
140 IF S1% AND 16%        THEN    PRINT "NON-NULL EXTENSION FOUND"
150 IF S1% AND 32%        THEN    PRINT "EXTENSION WAS '*'"
160 IF S1% AND 64%        THEN    PRINT "EXTENSION HAD '?'S"
170 IF S1% AND 128%       THEN    PRINT "PPN FOUND"
180 IF S1% AND 256%       THEN    PRINT "PROJECT NUMBER WAS '*'"
190 IF S1% AND 512%       THEN    PRINT "PROGRAMMER NUMBER WAS '*'"
200 IF S1% AND 1024%      THEN    PRINT "PROTECTION CODE FOUND"
210 IF S1% AND 2048%      THEN    PRINT "ASSIGN'D PROTECTION USED"
220 IF S1% AND 4096%      THEN    PRINT "COLON (:) FOUND"
230 IF S1% AND 8192%      THEN    PRINT "DEVICE NAME FOUND"
240 IF S1% AND 16384%     THEN    PRINT "DEVICE NAME WAS LOGICAL"
250 IF S1%<0%             THEN    PRINT "DEVICE NAME WAS NOT ASSIGN'D"
260 IF S1% AND 4096%      THEN
        IF S1%>0%         THEN    PRINT "'STATUS' HAS BEEN SET"
490 PRINT FOR I%=1% TO 2%
500 GOTO 10
32767 END
```

The following examples show some of the above messages:

```
STRING TO SCAN? ABCDEF.EXT
FILENAME FOUND
DOT (.) FOUND
NON-NULL EXTENSION FOUND


STRING TO SCAN? SY:FILENM.DEX
FILENAME FOUND
DOT (.) FOUND
NON-NULL EXTENSION FOUND
COLON (:) FOUND
DEVICE NAME FOUND
'STATUS' HAS BEEN SET
```

```
STRING TO SCAN? SY:FILENM.EXT[1,203]
FILENAME FOUND
DOT (.) FOUND
NON-NULL EXTENSION FOUND
PPN FOUND
COLON (:) FOUND
DEVICE NAME FOUND
'STATUS' HAS BEEN SET


STRING TO SCAN? SY:FILENM.EXT[2,103]<52>
FILENAME FOUND
DOT (.) FOUND
NON-NULL EXTENSION FOUND
PPN FOUND
PROTECTION CODE FOUND
COLON (:) FOUND
DEVICE NAME FOUND
'STATUS' HAS BEEN SET


STRING TO SCAN? SY:FILENNM.EXT[*,201]
FILENAME FOUND
DOT (.) FOUND
NON-NULL EXTENSION FOUND
PPN FOUND
PROJECT NUMBER WAS '*'
COLON (:) FOUND
DEVICE NAME FOUND
'STATUS' HAS BEEN SET


STRING TO SCAN? SY:A.*
FILENAME FOUND
DOT (.) FOUND
NON-NULL EXTENSION FOUND
EXTENSION WAS '*'
COLON (:) FOUND
DEVICE NAME FOUND
'STATUS' HAS BEEN SET


STRING TO SCAN?

STRING TO SCAN? SY:FILE??.EXT
FILENAME FOUND
FILENAME HAD '?'S
DOT (.) FOUND
NON-NULL EXTENSION FOUND
COLON (:) FOUND
DEVICE NAME FOUND
'STATUS' HAS BEEN SET


STRING TO SCAN? :A
FILENAME FOUND
COLON (:) FOUND
'STATUS' HAS BEEN SET
```

The STATUS variable is set or not set depending on the presence or absence of a device in the string scanned. The following three conditions pertain.

a. When no device name is found in the string (no colon is found), the STATUS is random. This condition pertains when bit 12 of flag word 2 tests as equal to Ø.

b. When the device name is logical and untranslatable (an actual device is not assigned), STATUS is random. This condition pertains when bits 12, 13 and 14 of flag word 2 test as not equal to Ø and bit 15 tests as on (S1%<Ø%).

c. When the device name is either an actual device name or is logical and translatable (an actual device is assigned), STATUS is set for the device. This condition pertains when bit 12 tests as not equal to Ø and bit 15 tests as equal to Ø (S1%>=Ø%).

Line 26Ø of the sample program shows the test to determine when STATUS is set by the call.

The file name string scan using CHR$(-23%) in place of CHR$(-1Ø%) terminates without error on the following characters.

= (equality sign)
/ (slant)
; (semi-colon)
, (comma)
end of string

The number of unscanned characters is returned in the BASIC-PLUS variable RECOUNT. For example,

```
S$=SYS(CHR$(6%) + CHR$(-23%) + "SY:[1,4]ABC<4Ø>")
```

returns the data as described above for CHR$(-1Ø%) and RECOUNT equals Ø. The following call

```
S$ = SYS(CHR$(6%) + CHR$(-23%) + "SY:[1,4]ABC<4Ø>,DT:DEF")
```

returns the data described above for the string "SY:[1,4]ABC<4Ø>" with RECOUNT equals 7. (The scan terminates on the comma between file names.)

Any other characters, including the angle bracket character (<), generate an error and none of the data is returned. Thus, the call with CHR$(-23%) simplifies the task of decoding command strings with multiple file names.

7.2.4.2 Return Error Message — Not Privileged (FØ=9)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(9), the return error message code
3	CHR$(E%), where E% is the RSTS ERR variable number and is between Ø and 127
4-3Ø	Not used

Data Returned:

Byte(s)	Meaning
1	The current job number times 2
2	The current keyboard number times 2
3-3Ø	Error message padded to length 28 with CHR$(Ø%) characters.

Possible Errors: No errors are possible.

Discussion:

This SYS system function call extracts error message text from the file ERR.SYS in the core image library. The text is associated with the value of the ERR variable passed as byte 3 of the call. The number in byte 2 of the returned string is two times the number of the keyboard involved in generating the error. This is an exception to the conventional contents of byte 2 which usually contains the job number times 2. A sample usage of the call is to print the system header line containing the system name and the local installation name. To do this, the character representation of the ERR value of Ø% is used in the call.

```
10      INPUT "ERROR NUMBER";E%
20      S$=SYS(CHR$(6%)+CHR$(9%)+CHR$(E%))
30      S1$=MID(S$,3%,INSTR(3%,S$,CHR$(0%))-3%)
40      PRINT S1$
50      PRINT FOR I%=1% TO 2%
60 GOTO 10
32767 END
```

To extract the message text from the data returned by the SYS call, the program executes an INSTR function based on the NUL byte (FILL character) indicating the end of the text. The MID substring of the returned data string, starting at byte number 3, extracts the number of bytes according to the value returned by the INSTR function.

Error numbers used in the call can include those associated with recoverable and non-recoverable errors.

```
RUNNH
ERROR NUMBER? 0
RSTS V05B-24 SYSTEM #880
```

7.2.4.3 Assign/Reassign Device — Not Privileged (FØ = 10)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(10%, the assign/reassign device code
3-6	Not used
7	Must be Ø for assign; for reassign, must be the job number to reassign the device to.
8	Must be Ø
9-22	Not used
23-24+	Device name (DT, PR, PP, MT, CR, LP, KB)
25+	Unit number
26+	Unit number flag
27-30	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
For the assign call	
NOT A VALID DEVICE The device name specified in bytes 23 and 24 is a logical device name which is currently not assigned.	6
DEVICE NOT AVAILABLE The device specified is currently assigned or in use by another job.	8
For the reassign call	
ACCOUNT OR DEVICE IN USE The device specified is currently open or has an open file.	3
NOT A VALID DEVICE The device name specified in bytes 23 and 24 is a logical device name which is currently not assigned.	6
DEVICE NOT AVAILABLE The device specified is currently assigned to another job or is in use by another job.	8

Example:

```
10 A$ = SYS(CHR$(6%)+CHR$(10%)+SPACE$(4%)+
        CHR$(0%)+CHR$(0%)+SPACE$(14%)+
        "LP" + CHR$(1%)+CHR$(255%))
        ! ASSIGN LP1: TO CURRENT JOB.
20 X%=4%
30 A$=SYS(CHR$(6%)+CHR$(10%)+SPACE$(4%)+
      CHR$(X%)+CHR$(0%)+SPACE$(14%)+
      "LP"+CHR$(1%)+CHR$(255%))
      ! REASSIGN LP1: TO JOB # X%.
```

7.2.4.4 Deassign a Device — Not Privileged (FØ = 11)

Data Passed:

Byte(s)	Meaning
1	CHR(6%), the SYS call to FIP
2	CHR$(11%), the deassign a device code
3-6	Not used
7-8	Must be Ø
9-22	Not used
23-24+	Device name
25+	Unit number
26+	Unit number flag
27-30	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
NOT A VALID DEVICE The device name specified in bytes 23 and 24 is not a valid device name.	6

Discussion:

This call performs the same action as the DEASSIGN system command described in the RSTS-11 System User's Guide.

Example:

The following statement deassigns line printer unit 1 which is assigned to the current job.

```
10  A$ = SYS(CHR$(6%) + CHR$(11%) + SPACE$(4%) +
       CHR$(Ø%) + CHR$(Ø%) + SPACE$(14%) +
       "LP" + CHR$(1%) + CHR$(255%))
       ! DEASSIGN LP1:
```

7.2.4.5 Deassign All Devices — Not Privileged (FØ = 12)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(12%), the deassign all devices code
3-30	Not used

Data Returned: No meaningful data is returned.

Possible Errors: No errors are returned.

Example:

The following statement deassigns all devices currently assigned to the job.

```
10 A$ = SYS(CHR$(6%) + CHR$(12%))
```

7.2.4.6 Zero a Device — Both Privileged and (FØ=13) Not Privileged

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(13%), the zero a device code
3-4	Not used
5-6	Project-programmer number (see note 1)
7-22	Not used
23-24+	Device designator (Disk, magtape, or DECtape)
25+	Unit number
26+	Unit number flag
27-30	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
CAN'T FIND FILE OR ACCOUNT The device specified is disk and the account specified in bytes 5 and 6 does not exist on the device	5
NOT A VALID DEVICE The device name specified is a logical name which is not currently assigned	6
DEVICE NOT AVAILABLE The device is currently assigned or in use by another job or has a file open.	8
DEVICE NOT FILE STRUCTURED The device specified does not allow access by file name.	30

Note 1:

Only privileged users can specify any account other than their own account to be zeroed. Any values a non-privileged user specifies in bytes 5 and 6 are forced to the caller's own project-programmer number. Zeroes in bytes 5 and 6 indicate the project-programmer number of the calling program.

Note 2:

When the zero a device SYS call is specified on magtape or DECtape, the entire medium is zeroed without regard to any project-programmer number. On DECtape, the directory is cleared. On magtape, the tape is rewound to LOAD POINT, three end of file marks are written, and the tape is rewound.

Example:

```
10  A$=SYS(CHR$(6%)+CHR$(13%)+SPACE$(2%)+
       CVT%$(0%)+SPACE$(16%)+"SY"+CVT%$(0%))
       ! ZERO MY OWN ACCOUNT ON THE SYSTEM.
20  P0%=10% : P2%=20%   !WANT TO ZERO [10,20]
30  A$=SYS(CHR$(6%)+CHR$(13%)+SPACE$(2%)+
       CHR$(P1%)+CHR$(P0%)+SPACE$(16%)+"SY"+
       CVT%$(0%))
       ! ZERO [10,20] ON THE SYSTEM.
       ! IF PROGRAM IS NON-PRIVILEGED, ZEROES
       ! CURRENT ACCOUNT
40  A$=SYS(CHR$(6%)+CHR$(13%)+SPACE$(20%)+
       "MT"+CVT%$(0%))
       ! ZERO MT:
```

7.2.4.7 CTRL/C Trap Enable — Not Privileged (FØ=-7)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-7%), the CTRL/C trap enable code
3-30	Not used.

Data Returned: No meaningful data is returned.

Possible Errors: No errors are possible.

Discussion:

After this FIP function is executed in the user program, the Run Time System treats the first CTRL/C subsequently typed on any terminal belonging to the job as a trappable error (ERR=28). Upon execution of the trap, control is immediately passed to the numbered program statement which has been designated as the error-handling routine by the last execution of an ON ERROR GOTO statement. After the trap, CTRL/C trapping is disabled. If it is desired that CTRL/C trapping remain in effect, the SYS call must be executed again.

Such trapping of CTRL/C, however, guarantees only that a defined set of statements is executed when CTRL/C is typed. It is not possible to resume execution at the exact point where the CTRL/C occurred.

If certain critical sections of BASIC-PLUS code are to be completely immune to possible CTRL/C aborts, three actions must occur.

a. The job must detach itself from its terminal. See the description of FIP code +7.

b. The program must have CLOSED all channels on which other terminals in the job had been OPENed.

c. The job must have DEASSIGNed any terminal which had been previously ASSIGNed to it. See the description of FIP code +11.

If the three actions occur, program execution under the job proceeds immune to any CTRL/C.

After the job has completed its critical processing in the detached state, one of three actions can occur.

a. The job can kill itself by means of FIP code +8.

b. The job can find a free terminal (presumably the one from which it detached itself) and "force" into that terminal input buffer the character strings needed for logging into the system and attaching the job to the terminal. (See the descriptions of FIP codes -4, +4, and +6.)

c. The job can find a free terminal and use the REATTACH SYS call to attach itself to the terminal (See the description of FIP code +6.)

The following sample program shows the procedure.

```
10 ON ERROR GOTO 100                              ! GET SET TO TRAP
20 A$=SYS(CHR$(6%)+CHR$(-7%))                     ! ENABLE ^C TRAPPING
30 PRINT "HI ";
40 SLEEP 10%                                      ! WAIT AWHILE
50 GOTO 30

100 IF ERR<>28% THEN ON ERROR GOTO 0
        ELSE RESUME 110                           ! LOOK FOR A CTRL/C
110 PRINT "^C TRAPPED"
120 SLEEP 10%
130 GOTO 20                                       ! GO BACK TO LOOP.
32767 END
```

The program prints "HI" at the keyboard every 1Ø seconds until a CTRL/C is typed. Then it prints the "C TRAPPED" message and a sleep operation for 1Ø seconds before reenabling the CTRL/C trap and printing "HI". The SLEEP statement before reenabling the trap is included to allow the user to type a second CTRL/C and actually stop the program.

Ordinarily, two CTRL/C characters typed very quickly at a terminal stop a program even if CTRL/C trapping is enabled. However, on a lightly loaded system, it is sometimes possible for the program to react quickly enough to the first CTRL/C that the second one can also be trapped. In this situation, the only means of stopping the job is through the kill job SYS call (or the KILL command in the UTILTY program). Thus, after the original trap, the user can stop the program by typing CTRL/C within 10 seconds. It is recommended that programs which trap CTRL/C characters be designed to include a certain amount of time after a trap in which a second CTRL/C actually stops the program.

7.2.5 Privileged Utility SYS Calls

The FIP calls described in this section are privileged calls; that is, they can be called only by a privileged user or by a privileged program. (See Section 5.5 for a discussion of privilege.) Any attempts to execute these calls by non-privileged users or programs result in the error ILLEGAL SYS( ) USAGE (ERR=18). Other errors are specified in the individual descriptions. The functions described in Sections 7.2.5.2 through 7.2.5.11 are used by the UTILTY system program. Examples of their usage can be found in the source code of that program.

7.2.5.1 Special Shutup Logout — Privileged (FØ=-16)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-16%), the special shutup logout code
3-30	Not used

Data Returned: No meaningful data is returned.

Errors Returned: Refer to the discussion.

Discussion:

This system function logs the current job off the system (as does the FIP system function call code 5) but, in addition, brings the CPU to an orderly halt at address 54 after the job is logged off the system.

Before this FIP call executes properly, several system conditions must be true. First, one and only one job can be running on the system when the SYS call is invoked. Next, the number of logins allowed on the system must be 1 (that is, LOGINS DISABLED. See Section 7.2.5.6). Next, no disks except the system disk can be mounted. Finally, no files can be open on the system disk.

If all of these conditions are fulfilled, the system shuts down. If any are not true, any attempt to invoke this SYS call results in the error ILLEGAL SYS( ) USAGE (ERR=18).

This SYS call is used by the SHUTUP program.

7.2.5.2 Date and Time Changer

Privileged (FØ=-14)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-14%), the date and time changer code
3	CHR$(D%) where D% is in the required format to generate the date by the function DATE$(D%). See Section 8.8 of the BASIC-PLUS Language Manual for a description of the DATE$ function.
4	CHR$(SWAP%(D%)) where D% is the same value used in byte 3. This generates the high byte of the value used by the DATE$(Ø%) function.
5	CHR$(T%) where T% is in the required format to generate the time by the function TIME$(T%). See Section 8.8 of the BASIC-PLUS Language Manual for a description of the TIME$ function.
6	CHR$(SWAP%(T%)) where T% is the same value used in byte 5. This generates the high byte of the value used by the TIME$(Ø%) function.
7-30	Not used.

Data Returned: No meaningful data is returned.

Discussion:

This function changes the monitor date and time of day values which are returned by the DATE$(Ø%) and TIME$(Ø%) functions in BASIC-PLUS.

7.2.5.3 Hang Up a Dataset

Privileged (FØ=-9)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-9%), the hang up a dataset code
3	CHR$(N%) where N% is the keyboard number of the line to hang up.
4	CHR$(S%) where S% is the number of seconds to wait before hanging up the line.
5-30	Not used.

Data Returned: No meaningful data is returned.

7.2.5.4 Broadcast to a Terminal — Privileged (FØ=-5)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-5%), the broadcast to a terminal code
3	CHR$(N%) where N% is the keyboard number of the terminal to receive the message.
4-?	M$ is the message to broadcast; LEN(M$) can be greater than 27. The string must not be null.

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE Generated if LEN(M$) is Ø.	18

Discussion:

The data broadcast is printed on the destination keyboard. The received message affects any output formatting being performed on the destination keyboard. System programs which use this function are TALK, PLEASE, BACKUP, and QUEMAN and UTILTY.

7.2.5.5 Force Input to a Terminal — Privileged (FØ=-4)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-4%), the force input to a terminal code
3	CHR$(N%) where N% is the keyboard number of the terminal to receive the forced input.
4-?	I$ is the input string to force to the terminal. The string must not be null. LEN(I$) can be greater than 27.

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE Generated if LEN(I$) is Ø.	18

Discussion:

The data forced is seen as input by the system. Other system programs besides UTILTY which use this function are BUILD, INIT, and CONTRL.

7.2.5.6 Disable Further Logins — Privileged (FØ=-2)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-2%), the disable further logins code
3-30	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE A non-privileged user attempts to execute this call.	18

Discussion:

This call sets the number of logins allowed on the system to 1. If no jobs are active on the system, one user can successfully log into the system. However, once one user is logged in, any delimiter typed at a logged out terminal returns the NO LOGINS message. This call is used by the UTILTY and SHUTUP programs.

7.2.5.7 Enable Further Logins — Privileged (FØ=-1)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-1%), the enable further logins code
3-30	Not used.

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE A non-privileged user attempts to execute this call.	18

Discussion:

This call sets the number of logins allowed to the number specified at start-up time - JOBMAX. The enable logins call is used by the UTILTY and INIT programs.

7.2.5.8 Disk Pack and Terminal Status — Privileged (FØ=3)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(3%), the disk pack and terminal status code
3	CHR$(N%); the following values of N% determine the resultant action.

Value	Action
Any Odd	Set terminal status. See FIP call 16.
Ø	Mount a disk pack or cartridge
2	Dismount a disk pack or cartridge
4	Lock out a disk pack or cartridge
6	Unlock a disk pack or cartridge

For Mount, Dismount, Lock, and Unlock

23-24+	Device name
25+	Unit number
26+	Must be 255.

For Mount

7-10+	Pack identification label in Radix-50 format
?	All bytes not specified are ignored.

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ACCOUNT OR DEVICE IN USE An attempt to dismount a disk which has an open file	3
NOT A VALID DEVICE Device name specified is not valid	6
DEVICE HUNG OR WRITE LOCKED An attempt to mount a disk which is not write enabled	14

Meaning	ERR Value
ILLEGAL SYS() USAGE An attempt by a non-privileged user to execute this call; or an attempt to mount a disk which is already mounted or which resides in a non-dismounted drive; or disk specified is the system disk.	18
PACK IDS DON'T MATCH An attempt to mount a disk with an incorrect pack label.	20
DISK PACK IS NOT MOUNTED An attempt is made to lock, unlock or dismount a disk which is not mounted.	21
DISK PACK NEEDS 'CLEANING' The storage allocation table on the disk needs to be restructured since the disk was not properly dismounted when it was last used. Disk is logically mounted but cannot be accessed until cleaned by the CLEAN command of UTILTY system program or by the FIP call with code 2.	25
FATAL DISK PACK MOUNT ERROR The disk is beyond recovery. For example, the cluster size is larger than 16 or the storage allocation table is unreadable.	26
DEVICE NOT FILE STRUCTURED An attempt to lock, unlock, or dismount a disk currently opened in non-file structured mode.	30

Discussion:

Note that if byte 3 contains any odd value, the call is interpreted as a set terminal characteristics call and is exactly equivalent to FIP call 16 discussed in Section 7.2.8. This call is used by the SHUTUP, UMOUNT, and BATCH system programs. (The terminal characteristics form of the call is used by the TTYSET system program). For a discussion of disk management on RSTS/E, see Section 6.3.2.

7.2.5.9 Clean Up a Disk Pack — Privileged (FØ=2)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(2%), the clean up a disk pack code
3-22	Not used
23-24+	Device name A zero in both bytes means the system disk.
25+	Unit number
26+	Unit number flag
27-30	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE An attempt to use this call by a non-privileged user; the device specified is not a disk; the disk is not locked; a file is open on the disk.	18
DISK PACK IS NOT MOUNTED The disk is not yet mounted.	21
CORRUPTED FILE STRUCTURE The link words in the directories are destroyed or completely meaningless.	29

Discussion:

A clean operation on an RK disk cartridge takes up to 3Ø seconds and on an RPØ3 disk pack takes up to 5 minutes. See Section 6.3.2 for a discussion of disk management and the clean operation.

7.2.5.10 Change Password/Quota — Privileged (FØ=8)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(8%), the change password/ quota, kill job, and disable terminal code
3-6	Not used
7-8	Project programmer number. Zero for both values means the current account. See Section 7.2.3 for an explanation of the value of each byte.
9-12	New password in Radix-50 format. All zeroes mean no change. See Section 7.2.4.1 for a description of converting strings to Radix-50 format.
13-14	CHR$(N%)+CHR$(SWAP%(N%)), where N% is the number of blocks for the quota. Zero in this word means unlimited quota if byte 21 is 255. Otherwise, zero means no change.
15-20	Not used
21	CHR$(255%) if the quota of Ø specified in bytes 13 and 14 is valid rather than no change.
23-24+	Device name
25+	Unit number
26+	Unit number flag
27	Not used
28	Must be CHR$(Ø%)
29-30	Not used.

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
CAN'T FIND FILE OR ACCOUNT The account is not present on the disk specified.	5
NOT A VALID DEVICE Device specified is not valid.	6

Meaning	ERR Value
ILLEGAL SYS() USAGE An attempt by a non-privileged user to execute this call or the device specified is not a disk.	18

Discussion:

Either the password or the quota can be changed individually as well both can be changed in the same call.

7.2.5.11 Kill Job — Privileged (FØ=8)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(8%), the change password/ quota, kill job, and disable terminal code
3	CHR$(N%) where N% is the number of the job to kill
4-26	Not used
27	Must be CHR$(Ø%); this byte differentiates the kill job call from the disable terminal call
28	Must be CHR$(255%)
29-30	Not used.

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE The job number specified is Ø or is greater than the system JOB MAXIMUM value. An attempt by a non-privileged user to execute this call.	18

Discussion:

There is only one proper way for a job to terminate itself under programmed control. The job must execute the kill FIP call on its own job number. The kill does all of the clean-up that the logout FIP call (FØ=5) does, but this function can be executed under program control by any (privileged) program, whereas the logout call requires certain special conditions. Examples of this SYS call can be found in the ERRCPY, BATCH, SPOOL, and QUEMAN programs.

7.2.5.12 Disable Terminal — Privileged (FØ=8)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(8%), the change password/ quota, kill job, and disable terminal code
3	CHR$(N%), where N% is the keyboard number of the terminal to disable
4-26	Not used
27	Must be CHR$(255%) to differentiate this call from the kill job call
28	Must be CHR$(255%)
29-30	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE Keyboard number is greater than the number of terminals on the system (NULINE); Keyboard number corresponds to a line used by a pseudo keyboard; Keyboard number relates to a terminal on a DH11 multiplexor line; The terminal is currently opened or assigned by a job; or The keyboard is the system console terminal (KBØ:)	18

Discussion:

This FIP call disables a keyboard line. After this function has been executed, no input from the disabled keyboard is processed or echoed by the system, and output generated for the terminal is ignored. There is no complementary function in RSTS/E. Once a keyboard is disabled, it remains disabled until the next time the system starts time sharing.

Note that this function only disables keyboards connected to the PDP-11 with single line interfaces (KL11, DL11, etc).

No system program (including UTILTY) currently uses this call. It is included at this point because of its relationship to the change password and kill job FIP calls.

7.2.6 Job Scheduling SYS Calls to FIP

7.2.6.1 Priority, Run Burst and Size Maximum Changer — Privileged (FØ=-13)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-13), the priority, run burst, and size maximum changer
3	CHR$(J%) where J% is the job number affected or is -1 to denote the current running job
4	CHR$(A%) where A% is Ø% to indicate no change to the parameter in byte 5 or is non-zero to indicate a change to the parameter as specified in byte 5.
5	CHR$(P%) where P% is the value of the running priority and ranges from -128 to +128 in steps of 8.
6	CHR$(A%) where A% is Ø% to indicate no change to the parameter in byte 7 or is non-zero to indicate a change to the parameter as specified in byte 7.
7	CHR$(R%) where R% is the run burst
8	CHR$(A%) where A% is Ø% to indicate no change to the parameter in byte 9 or is non-zero to indicate a change to the parameter as specified in byte 9.
9	CHR$(S%) where S% is the maximum size, in 1024-word units, to which a job can expand and is between 2 and the current value for SWAP MAX on the system.

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE An attempt by a non-privileged user to execute this call	18

Data Returned: No meaningful data is returned.

Discussion:

This system function allows a privileged user to give a running job an increased or decreased chance of gaining run time in relation to other running jobs, and to determine how much CPU time the job can have if it is compute bound. The CPU time is termed the job's run burst and is measured by the number of clock interrupts during which the job can run if it is compute bound.

The size of a job at log in time is set at 2K and can grow during processing to a size limited by the value of SWAP MAX. SWAP MAX is determined at the start of time sharing operations by the system manager. (Refer to the description of SWAP MAX given in the START and DEFAULT option discussion in Chapter 3.) The maximum size to which a job can grow can never be greater than the currently assigned value of SWAP MAX, which must be between 8K and 16K words. Therefore, the privileged user has the option of limiting the size to which a job can grow by specifying a value for S% between 2 and the maximum of SWAP MAX.

Values for each of the variables in the parameter string must be specified. The value for A preceding the related parameter variable determines whether that parameter changes or remains unchanged.

The PRIOR system program provides a direct example of the use of this FIP call. The call is also used by the LOGIN, SPOOL, and RJ2780 system programs.

No error-checking is done by the system on the data passed by the user. Values are used as passed even if they generate illogical results. For instance, if a priority is specified which is not a multiple of 8, its value is truncated to the next lowest multiple of 8. A priority greater than 128 is considered negative. Setting a job's run burst to Ø prevents the job from obtaining any run-time. Setting a (compute bound) job's run-burst to some high number tends to lock out other jobs. However, setting S% to 255, or any value greater than the system SWAP MAX does not override the system maximum.

7.2.6.2 Set Special Run Priority

Privileged (FØ=-22)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-22%), the set special run priority call
3-30	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE An attempt by a non-privileged user to execute this call.	18

Discussion:

This SYS function sets the special run priority bit in the job priority word. This action raises the priority of the job slightly above that of other jobs in its priority class. The priority bit is cleared whenever the job returns to the READY state. Thus, a privileged job can raise its priority without protecting against a user typing CTRL/C and retaining the higher priority. This FIP call is used by the QUE system program.

7.2.6.3 Lock/Unlock Job in Memory

Privileged (FØ=-20)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-20%), the lock/unlock job in memory code
3	CHR$(N%) where N% is Ø for lock and is 255 for unlock
4-30	Not used

Data Returned: No meaningful data is returned

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE An attempt by a non-privileged user to execute this call	18

Discussion:

This call prevents unnecessary swapping by forcing the job executing the call to remain in memory. This action is performed without affecting the job priority or run burst. The call merely eliminates the swapping time between run bursts.

A program having certain time sensitive routines can lock itself in memory. The duration of the locked time must be very short to prevent degradation of system performance. Depending on the memory configuration, a locked job can prohibit the system from swapping any other job into memory. If the job expands its size in memory, the system can swap it out of memory regardless of its locked status.

The following sample code demonstrates the lock and unlock procedure.

```
10      A$ = SYS(CHR$(6%) + CHR$(-2Ø%) + CHR$(Ø%))
        ! LOCK JOB IN MEMORY
100     A$ = SYS(CHR$(6%) + CHR$(-2Ø%) + CHR$(255%))
        ! UNLOCK JOB FROM MEMORY
```

7.2.6.4 Drop Temporary Privileges — Privileged (FØ=-21)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-21%), the drop temporary privileges code
3-30	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE An attempt by a non-privileged user to execute this call	18

Discussion:

This call allows a compiled program which has a privileged protection code to drop temporary privileges and privileges afforded by JFHIBY being set. A program normally executes this call after it has used the special privileges to set itself up. The program can take advantage of built in monitor protections (for example, file protection code arbitration) which are otherwise overridden by a program's temporary privileges. The call does not affect the permanent privileges of an account. Both the QUE and LOGIN system programs use this call.

The following sample code shows how a program might drop its temporary privileges.

```
1Ø      A$ = SYS(CHR$(6%) + CHR$(-22%)
        ! SET SPECIAL RUN PRIORITY
2Ø      OPEN "SYSTEM.FIL$" AS FILE 1%
        ! OPEN REFERENCE FILE, REGARDLESS OF PROTECTION
3Ø      A$ = SYS(CHR$(6%) + CHR$(-21%)
        ! DROP TEMPORARY PRIVILEGES
4Ø      OPEN "ACCT.FIL" AS FILE 2%
        ! THIS FAILS IF FILE IS PROTECTED AGAINST THE
        ! CURRENT ACCOUNT
```

7.2.7 Account Creation and Deletion SYS Functions

7.2.7.1 Create User Account — Privileged (FØ=Ø)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(Ø%), the create user account code
3-6	Not used
7-8	Project-programmer number; see Section 7.2.3. The project number can be between 1 and 254; the programmer number can be between Ø and 254.
9-12	Password in Radix-50 format; see Section 7.2.4.1 for a description of converting a string to Radix-50 format.
13-14	Disk Quota. See Section 7.2.3 for a description of unsigned numbers. Zero means unlimited quota.
15-22	Not used
23-24+	Device name
25+	Unit number
26+	Unit number flag
27-28	User file directory (ufd) cluster size; Ø means use the pack cluster size.
29-30	Not used.

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ILLEGAL FILE NAME Password is missing in the call	2
NO ROOM FOR USER ON DEVICE The directory currently has the maximum number of accounts.	4
PROTECTION VIOLATION The project-programmer number is [Ø,Ø] or either the project or programmer number is 255.	10
NAME OR ACCOUNT NOW EXISTS The account specified in the call currently exists on the device specified.	16

Meaning	ERR Value
ILLEGAL CLUSTER SIZE The cluster size specified in the call is either greater than 16 or is non-zero and less than the pack cluster size. See Section 5.4.3 for a discussion of valid cluster size values.	23
DEVICE NOT FILE STRUCTURED The device specified is not a disk or the disk is open in non-file structured mode.	30

Discussion:

This call is used by the REACT system program to create accounts.

7.2.7.2 Delete User Account — Privileged (FØ=1)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(1%), the delete user account code
3-6	Not used
7-8	Project-programmer number. This call generates an error if account [Ø,Ø], [Ø,1], or [1,1] is specified. See Section 7.2.3 for an explanation of the value of each byte.
9-22	Not used
23-24+	Device name; must be a disk
25+	Unit number
26+	Unit number flag
27-30	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ACCOUNT OR DEVICE IN USE The account contains files (it has not been zeroed) or a user is currently logged into the system under the account.	3
CAN'T FIND FILE OR ACCOUNT The account specified does not exist.	5
PROTECTION VIOLATION Account specified is either [Ø,Ø], [Ø,1], or [1,1].	10
DEVICE NOT FILE STRUCTURED Device specified is not a disk or is a disk open in non-file structured mode.	30

Discussion:

The REACT system program uses this call to delete user accounts. To prevent error number 3, the user must first zero the account using either the /ZE option in the PIP system program or the ZERO command of the UTILTY system program. The FIP call (FØ=13) can also zero an account.

7.2.8 Set Terminal Characteristics

Privileged (FØ=16)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(16%), the set terminal characteristics code; if CHR$(3%), byte 3 must be odd.
3	If byte 2 is CHR$(3%), this byte must be CHR$(N%) where N% is an odd value.
4	CHR$(N%) where N% is 255% for the current keyboard or is the keyboard number of the terminal to alter.
5	CHR$(N%) where N% is Ø% for no change or is the terminal width plus 1. The call sets the number of characters per line to N-1 where N% can be between 2 and 255. The WIDTH n command sets this byte.
6	CHR$(N%) where N% is: Ø for no change 128 to enable hardware horizontal tab feature. The TAB command sets this characteristic. (Device must have the requisite hardware.) 255 to enable software horital tab positions which are set every 8 character positions, beginning at position 1. The NO TAB command sets this characteristic.
7	CHR$(N%) where N% is: Ø for no change 128 to enable the software to perform form feed and vertical tab operations by executing 4 line feed operations. The NO FORM command sets this characteristic. 255 to enable hardware form feed and vertical tab. The FORM command sets this characteristic.
8	CHR$(N%) where N% is: Ø for no change 128 to allow terminal to receive and print lower case alphabetic characters. The LC OUTPUT command sets this characteristic. 255 to have the system translate lower case alphabetic characters to upper case before transmitting to a terminal. The NO LC OUTPUT sets this characteristic.

Byte(s)	Meaning
9	CHR$(N%) where N% is: Ø for no change 128 to have terminal not respond to XON CHR$(17) and XOFF CHR$(19) characters because it lacks the requisite hardware. The NO XON command sets this characteristic. 255 Terminal has requisite hardware to respond to XON and XOFF characters. Terminal stops sending characters when it receives a CHR$(19) character (XOFF) and resumes sending characters when it receives a CHR$(17) character (XON). The XON command sets this characteristic.
10	CHR$(N%) where N% is: Ø for no change 128 to have characters typed at the terminal sent to the computer only. The computer echoes (transmits back to the terminal) what it receives and performs any necessary translation. The FULL DUPLEX command sets this characteristic. 255 to have the terminal (or its acoustic coupler) locally echo the characters typed. The computer does not echo the characters received. The LOCAL ECHO command sets this characteristic.
11	CHR$(N%) where N% is: Ø for no change 128 Terminal does not have features of a video display terminal. The NO SCOPE command sets this characteristic. 255 Terminal is a video display, or cathode ray tube (CRT) type, and uses the following features: a. Responds to synchronization standard described by byte 17. b. The system executes a DEL character (RUBOUT) by sending a backspace, a space, and a backspace to the terminal.

Byte(s)	Meaning
	c. Any location on the screen can be addressed by direct cursor placement. The SCOPE command sets this characteristic.
12	CHR$(N%) where N% is: Ø for no change 128 System treats certain characters received as follows: a. Translate CHR$(125) and CHR$(126) into the ESC character CHR$(27). b. Translate lower case characters to upper case characters. Set by NO LC INPUT command. 255 Terminal transmits the full ASCII character set and system treats special characters as follows: a. Treat only CHR$(27) as an escape character (echoed as the $ character and handled as a line terminating character. b. Treat CHR$(125) and CHR$(126) as printed characters { and ~. c. Do not translate lower case characters to upper case format. Set by LC INPUT command.
13	CHR$(N%) where N% is: Ø for no change n Set fill factor of terminal to N%-1. The command FILL n determines this value. 255 Set fill factor for an LA30S (serial) DECwriter. The command FILL LA30S sets this characteristic.
14	CHR$(N%) where N% is: Ø for no change n The internal speed value to determine the baud rate at which the terminal receives characters. If byte 16 is Ø, this value also determines the transmit (output) baud rate. If byte 16 is 255, this byte must be 255. For a DH11 line, n is between 1 and 16. For a DC11 line, n is between 1 and 4. See the PDP-11 Peripherals Handbook for the related baud rates.

Byte(s)	Meaning
14 (Cont.)	255 2741-type terminal. Byte 16 must also be 255%.
15	CHR$(N%) where N% is: Ø for no change 1 Do not set the output parity bit. This value is set by the NO PARITY command. 2 Set the output parity bit for even parity format. The EVEN PARITY command sets this value. 3 Generate an output parity bit for odd parity format. The ODD PARITY command sets this value.
16	CHR$(N%) where N% is: Ø Both the receive (input) and transmit (output) speeds are determined by n in byte 14. n The internal speed value to determine the baud rate at which the terminal transmits characters when a split speed setting is used. For a DH11 line, n is between 1 and 16; for a DC11 line, n is between 1 and 4. 255 2741-type terminal. (See description of byte 20). Byte 14 must also be 255%.
17	CHR$(N%) where N% is: Ø for no change 128 Terminal ignores synchronization standards described for 255% value. The NO STALL command sets this characteristic. 255 Terminal obeys the synchronization standard as follows. The computer stops sending characters if the terminal transmits a CHR$(19) character (XOFF, or the CTRL/S combination). Computer resumes sending characters when the terminal transmits a CHR$(17) character (XON, or the CTRL/Q combination). The STALL command sets this characteristic.

Byte(s)	Meaning
18	CHR$(N%) where N% is: Ø for no change 128 System echo prints a control character received as the up arrow (↑or^) character followed by the equivalent printable character. For example, the CTRL/D combination is printed as ↑D, CHR$(94) followed by CHR$(68). The UPARROW command sets this characteristic. 255 System treats control characters as such. The NO UPARROW command sets this characteristic.
19	CHR$(N%) where N% is a value between 1 and 30 and determines the maximum number of characters in a burst for a DH11 line. For low speed lines, this is set to 30; for VT5Ø lines, it is set to 12. The DH BURST n command sets this value
20	CHR$(N%) where N% depends on the values of two other bytes. If bytes 14 and 16 are both 255, the value of this byte applies to the 2741-type terminal as follows: n = 8 + DATA+STOP+PARITY where: DATA is Ø for 5 bits per character 1 for 6 bits per character 2 for 7 bits per character 3 for 8 bits per character STOP is Ø for 1 stop bit per character 4 for 2 stop bits per character or 1.5 bits if DATA=Ø. PARITY is Ø for no parity bit. 16 for even parity format 48 for odd parity format If either byte 14 or 16 is other than 255, this byte is not used. The 2741 command determines the value of this byte.

Byte(s)	Meaning
21	CHR$(N%) where N% is: Ø for no change 255 Set the ring list entry for a terminal attached to a DC11, DL11E or DH11 line interface to default to permanent characteristics when modem is answered. The /RING option with a TTYSET KBn: command determines this value.
22	CHR$(N%) where N% is: Ø for no change 128 The system software treats an incoming ESC CHR$(27%) character as a line terminating character and echoes it as the $ character. The NO ESC SEQ command sets this value. 255 The software treats an incoming ESC CHR$(27%) character and the next incoming character as a special escape sequence. The software does not echo either character but transposes the characters and changes the ESC character to a CHR$(155%) character. That is, if the incoming data is: ESC X 27 XXX then the user receives: X ESCAPE XXX 155 The ESC SEQ command sets this value.

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR VAlue
ILLEGAL SYS() USAGE The keyboard number specified in byte 4 of the call is out of range of the valid keyboard numbers.	18

Discussion:

If the terminal specified by the keyboard number in byte 4 of the call either is disabled (as a result of the system initialization procedure or of executing the disable terminal SYS call) or is a pseudo keyboard, the call is not executed by the system. The TTYSET system program employs this call to set terminal characteristics. Refer to the discussion of TTYSET in Section 4.5 of the RSTS-11 System User's Guide or in Section 6.7 of this guide.

7.2.9 Change File Statistics — Privileged (FØ=-11)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-11%), change file statistics code
3	CHR$(N%) where N% is the internal channel on which the file is open. Must be between 1 and 12, inclusive.
4-5	Desired date of last access
6-7	Desired date of creation
8-9	Desired time of creation
10-30	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE The file open on the channel specified is not a disk file or is a user file directory. An attempt by a non-privileged user to execute this call.	18

Discussion:

The data passed by this call replaces the related data in the accounting block of the file open on the channel specified in byte 3. No error checking is done on the date and time values passed. Since the call does not supply default values, the user program must supply all three date and time values each time the call executes. The call is used by the BACKDK system program to maintain original date and time statistics for a file.

The following is a partial directory listing of a privileged account showing the file whose statistic information is to be changed.

```
CAT
CTPBLD.BAS        2        60      04-SEP-74 03-MAY-74 09:00 PM
```

The following program changes the date of creation to 22-July-74, and the date and time of last access to 12:00 noon, 22-July-74, as shown on the partial directory listing following the program.

```
LISTNH
10      OPEN "CTPBLD.BAS" AS FILE 1%     ! OPEN FILE TO CHANGE.
20      DIM M%(30%)                      ! USE THIS ARRAY TO SET UP CALL
30      M%(0%)=30%
40      M%(1%)=6%
50      M%(2%)=-11%                      ! SET UP FOR CHANGE STATS CALL
60      M%(3%)=1%                        ! ON CHANNEL 1.
70      M%(4%)=4203% AND 255%
        : M%(5%)=SWAP%(4203%) AND 255%   ! SET UP DATE OF CREATION
                                         !  FOR 22-JUL-74 (DATE$(4203%)=
                                         !  22-JUL-74).
80      M%(6%)=4203% AND 255%
        : M%(7%)=SWAP%(4203%) AND 255%   ! SET UP LAST ACCESS TO 22-JUL-74.
90      M%(8%)=720% AND 255%
        : M%(9%)=SWAP%(720%) AND 255%    ! SET UP TIME OF LAST ACCESS TO 12 NOON
                                         !  (TIME$(720%)=12:00 PM)
100     CHANGE M% TO M$                  ! SET IT UP AS A STRING.
110     M$=SYS(M$)                       ! AND DO THE WORK
32767 END

READY

RUNNH

READY

CAT
CTPBLD BAS       2       60     22-JUL-74 22-JUL-74 12:00 PM


READY
```

7.2.10 LOGIN and LOGOUT SYS Calls

7.2.10.1 LOGIN — Privileged (FØ=4)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(4%), the LOGIN code
3-6	Not used
7-8	Project-programmer number. Must not be account [Ø,1]. See Section 7.2.3 for an explanation of the value of each byte.
9-12	Password in Radix-50 format. See Section 7.2.4.1 for a description of converting a string to Radix-50 format.
13-30	Not used

Data Returned:

Byte(s)	Meaning
1-2	Internal data
3	Total number of jobs logged into the system under this account
4-?	Job numbers of each job running detached under this account. A byte of CHR$(Ø%) signifies the end of the list. Only the first 26 job numbers are returned.

Possible Errors:

Meaning	ERR Value
CAN'T FIND FILE OR ACCOUNT The project-programmer number specified in the call is [Ø,1], does not exist, or its password does not match the password of the account on the system.	5

Discussion:

If the calling job is already logged into the system, this call does not change the job's account. The data returned in bytes 3 through 30 refers to the same account under which the job is running.

7.2.10.2 LOGOUT Privileged (FØ=5)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(5%), the LOGOUT code
3-30	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR VAlue
ILLEGAL SYS() USAGE The JFHIBY value is not set when the call is executed.	18

Discussion:

This call closes all open channels, deassigns all devices, updates statistics on the disk, clears the job from the monitor message table, and disassociates the project-programmer number from the job number. However, none of these actions are performed unless the monitor sets a special flag called JFHIBY.

The monitor sets JFHIBY only when a user types the HELLO, BYE, or ATTACH commands at a terminal logged into the system or types anything at a terminal not logged into the system. No way exists for a BASIC-PLUS program to set JFHIBY. Thus, an already running program can never log itself off the system.

When the monitor sets the JFHIBY flag, the running program has full privileges. The drop temporary privileges call resets the JFHIBY flag to disable the privileges afforded by the flag. The LOGOUT and LOGIN system programs use the logout call.

7.2.11 Detach, Attach, and Reattach SYS Calls

7.2.11.1 Detach

Privileged (FØ=7)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(7%), the detach code
3-30	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE The current job is already detached or its console keyboard is open on a channel other than channel Ø.	18

Discussion:

This call disassociates the job and its console keyboard. The following sample program segment prints a message and detaches from the keyboard.

```
100  PRINT "DETACHING..."
     ! NOTIFY THE USER
110  A$ = SYS(CHR$(6%) + CHR$(7%))
     ! DO THE DETACH
```

When data is entered at a RSTS/E terminal, the system activates a job to handle the input and gives the job the next available job number. If the data is recognized by the system, certain actions are executed under that job number, one of which can be logging a user into the system. (See Sections 2.1 and 4.1 of the RSTS-11 System User's Guide for the operational details.) When a user is logged into the system, the activated job is associated by the system with both the terminal at which he is typing and the account number which he used to identify himself to the system. The job is then considered active on the system and in attached mode, or, simply, attached to the terminal.

A privileged job can become detached from the terminal by executing this call. Once a job is placed in the detached state, it runs as any other job logged into the system but it does not employ a terminal device on channel Ø. The detached state is advantageous for non-interactive jobs. The job running detached frees a terminal for other usage and makes the job immune from interruption by someone typing a CTRL/C combination.

When the user desires to attach a detached job to a terminal, he can log into the system at any free terminal using the account number under which the detached job was made active and attach that job to the terminal. (This procedure is described in Section 4.1 of the RSTS-11 System User's Guide.) Since the system associated the job number of the detached job with the account number under which that job was made active, it reports the detached job under the same account number.

This attachment facility is valuable in another manner. A job can be placed in a detached state by the system when the carrier is dropped on a remote line. This means that, if the telephone connection is lost while a job is running from a terminal at a remote location, the content of the job is not lost. The user simply logs into the system again with the same account number and reattaches to the job he was previously running.

The ERRCPY, QUEMAN, SPOOL, and BACKUP system programs use this call.

7.2.11.2 Attach — Privileged (FØ=6)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(6%), the attach and reattach code. The code to attach and reattach is the same but the format of the data passed is quite different. See Section 7.2.11.3 for the reattach format.
3	The number of the job to attach to the terminal.
4	Must be Ø
5-6	Project-programmer number of the job to attach to the terminal. See Section 7.2.3 for a description of the exact contents of each byte.
7-1Ø	Password of the account specified in bytes 5 and 6 in Radix-5Ø format. See Section 7.2.4.1 for information on converting a string to Radix-5Ø format.
11-3Ø	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	Err Value
ILLEGAL SYS() USAGE Each of the following conditions generates this error: 1. The job executing the call has an open channel. 2. The job executing the call is a source (BAS) program rather than a compiled (BAC) program. 3. The job number specified in byte 3 is not a detached job. 4. The account or password in the call is not valid. 5. The job executing the call is detached.	18

Discussion:

The LOGIN system program is the only program than can execute this call. See the LOGIN.BAS listing for an example of its usage. Note that, if byte 3 is the number of the job executing the call, the system performs the reattach action. See Section 7.2.11.3 for a description of the reattach process.

7.2.11.3 Reattach — Privileged (FØ=6)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(6%), the attach and reattach code. The code to attach and reattach is the same but the format of the data passed is quite different. See Section 7.2.11.2 for the attach format.
3	CHR$(J%) where J% is the number of the job executing the call
4	CHR$(K%) where K% is the keyboard number of the terminal to which the calling job is to be attached
5-30	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	Err Value
ILLEGAL SYS() USAGE Each of the following conditions generates this error: 1. The job number specified in byte 3 is less than 1 or greater than the JOB MAX value on the system. 2. The job executing the call is not detached. 3. The keyboard number in byte 4 is out of range. 4. The terminal specified by the keyboard number in byte 4 is currently assigned, opened, or the console keyboard of some job. 5. The job executing the call has an open file. 6. The job executing the call is a source (BAS) program rather than a compiled (BAC) program.	18

Discussion:

A privileged job can execute this form of the attach and reattach call. The call establishes the terminal specified in byte 4 as the console keyboard of the detached job executing the call. In this manner, a job can reattach to a terminal without having to force the proper data to the desired terminal.

7.2.12 Send and Receive Messages

Both Privileged (FØ=18) and not Privileged

This function call allows a user's job to do the following:

a. declare itself a receiving job,
b. send a message to a receiving job,
c. receive a message from pending messages,
d. stall until a message is pending, and
e. eliminate itself or another job as a receiving job.

The monitor controls eligibility to receive messages by maintaining a table of receiving jobs. The response to the RECEIVERS query at system generation time determines the maximum number of jobs eligible to receive messages at any given time. To be eligible to receive messages, a job must declare itself and have its identification entered in the table of receiving jobs. A job sending a message succeeds only if the job to which the message is sent has an entry in the table.

Sending and receiving messages on the system uses small buffers. Each message occupies one small buffer. A job defines, in its first receive call, the number of pending messages (messages sent to the job but not yet received by the job) which monitor allows to be queued for the job at any given time. This maximum can be as high as 127. To prevent occupying a large number of small buffers, and thereby degrading system performance, the pending maximum for each receiver should be small (1Ø to 15) and each receiving job should extract its messages quickly.

The system controls message operations on a first-in, first-out (FIFO) basis. It maintains pending messages as a linked chain of small buffers. When a job sends a message to an eligible receiving job, the system appends the related small buffer to the last small buffer in the chain of messages pending for the job. When a receiving job asks for a pending message, the system makes available the first message in the chain and removes the related small buffer from the chain.

The system continues message operations for a receiving job until either the maximum number of messages are pending or the supply of small buffers is exhausted. Since such conditions affect system operations, a receiving job must process its pending messages frequently to maintain adequate system performance. Since poorly designed use of the receive mechanism can drastically degrade overall system performance, the receive operation can be executed only by a privileged user.

A receiving job must remove itself from the table of receiving jobs or have another job remove it. To keep non-active jobs from occupying entries in the table of receivers, both the logout SYS call and the kill job SYS call remove the job from the table of eligible receiving jobs.

7.2.12.1 Declaring a Receiver and Receiving a Message — Privileged (FØ=18)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(18%), the send and receive a message code
3	CHR$(N%) where N% is one of the following values: 1 Attempt to receive a message or declare this job as a receiver and attempt to receive a message. Return an error condition if no messages are pending. 2 Receive with sleep. Similar to 1 except that the job executes a sleep operation if no messages are pending. This action occurs in place of generating an error condition.
4	CHR$(P%+L%) where P% can be either Ø or 128. If P is Ø, messages can be received from any sending job. If P is 128, messages can be received from and queued only by sending jobs which are privileged. L% is the number of messages (between 1 and 127) which can be simultaneously pending for this receiving job.
5-8	Receiving job logical name in Radix-5Ø format. See Section 7.2.4.1 for a description of converting a string to Radix-5Ø format.
9-3Ø	Not used

Data Returned:

Byte(s)	Meaning
1-4	Not used
5	CHR$(J%) where J% is the job number times 2 of job sending the message
6	Must be CHR$(Ø%)
7-8	Project-programmer number of the pending job. See Section 7.2.3 for a description of each byte.
9-28	The message string. The system pads any unused bytes with NUL characters to a length of 20 bytes.
29-3Ø	Not used.

Possible Errors:

Meaning	ERR Value
NO ROOM FOR USER ON DEVICE When the job attempts to declare itself as a receiving job, the monitor table containing data of eligible receiving jobs is full or is zero length.	4
CAN'T FIND FILE OR ACCOUNT For receive only, error indicates no messages are pending. For receive with sleep, error indicates no messages were pending when the receive was executed by monitor. Error is returned to the program when monitor awakens job from sleep.	5
ILLEGAL SYS() USAGE When the job attempts to declare itself as a receiving job, either the logical name is missing from the call (bytes 5 through 8 are not given) or another job has already declared itself a receiver with the logical name given.	18

Discussion:

A receive call checks the eligibility of a job to receive messages and performs both or one of two actions based on the result of the check. First, if the job is not eligible to receive messages, the call declares the job as an eligible receiver and attempts to receive a message. Second, if the job is already eligible, the call attempts to receive a message. Since the same call performs two actions, it is important that the user program handle the declaration and receiving procedure properly.

To check the eligibility of a job to receive messages, the system determines if the calling job's job number appears in a table in the monitor part of memory. If the job number is not in the so called receiver table, the system ensures that the logical name specified in bytes 5 through 8 of the data passed is not currently being used by another job. If the logical name is unique and an empty slot is available, the system declares the job as a receiver by entering in the table its job number, its logical name, and other data. Subsequent to declaring the job as a receiver, the receive call never refers to the logical name. The logical name exists so that a sending job can easily refer to a receiver without supplying its job number.

A receive call can not change a logical name in the table of eligible receivers because the system refers to the job number in the receive table rather than to the logical name. If the job number of the current job is in the table, the system considers the job eligible and has no need to refer to the logical name. This condition is important if a previous receiver with the same job number as the current receiver failed to remove itself from the table before terminating processing. Thus, a logical name already appears in the table for the current receiver when it attempts to declare itself a receiver.

To eliminate the possibility of a spurious logical name appearing in the table for the current job, it is recommended that a program execute the call to remove itself as a receiver before it executes the first receive call to declare itself a receiver. In this manner, the program ensures that the logical name in the table for the current job is, in fact, the name declared in bytes 5 through 8 of the call.

When the receive call declares a job as a receiver, it also attempts to receive a message. Because the system does not queue messages for a job which is not an eligible receiver, the first attempt to receive a message always fails.

When the receive call determines that a job is eligible, it attempts to receive a message. If a message is pending for the job, the call returns the information in bytes 9 through 28 of the target string. If no message is pending for the job, the call executes according to the value of N% in byte 3 of the data passed. If the value of N% is 1, the call immediately generates a recoverable error (ERR = 5). If the value of N% is 2, the call puts the job in a SLEEP state (called a receiver sleep).

The system awakens a job in a receiver sleep if a message becomes queued for it or if a line terminating character is typed on one of its keyboards. Since the system presets the recoverable error condition (ERR = 5) before putting the job to sleep, the receiving job, upon awakening, detects the error condition. The system does not pass the message to the job. To obtain the message queued, the job must execute the receive call again.

7.2.12.2 Send a Message Both Privileged and Not Privileged (FØ=18)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(18%), the send and receive a message code
3	CHR$(-1%); this value indicates that the call is a request to send a message
4	CHR$(J%) where J% is the job number times 2 of the job to receive the message. If J% is Ø, the call uses the logical name in bytes 5 through 8 to determine the receiving job
5-8	Receiving job name in Radix-5Ø format. See Section 7.2.4.1 for a description of converting a string to Radix-5Ø format
9-28	Message text to send. Can be a maximum of 2Ø bytes and the system pads the message with NUL characters to the length of 2Ø bytes
29-3Ø	Not used.

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
CAN'T FIND FILE OR ACCOUNT The receiving job specified is not in the monitor table of eligible receiving jobs.	5
ILLEGAL SYS() USAGE The receiving job specified is capable of receiving messages only from privileged jobs and the sending job is not privileged.	18
NO ROOM AVAILABLE FOR FCB One of two conditions is possible. 1) The number of messages pending for the receiving job is at its declared maximum. The sending job must try again. If this condition occurs frequently, it indicates that the declared maximum is too low or that the receiving job is not processing its messages quickly enough. 2) Also, sending a message requires a small buffer and one is not available.	32

Discussion:

The SEND operation sends a message to a declared receiving job in one of two ways: by either a job number or by a job identification. When byte 4 of the data passed is non-zero, the call ignores bytes 5 through 8 and attempts to send the message to the job designated by the value in byte 4. If byte 4 is zero, the call attempts to send the message to the receiver whose identification matches that given in bytes 5 through 8.

The sending job can be either privileged or non-privileged. The sending job must be privileged if the receiver is capable of receiving messages only from privileged sending jobs. (The receiver determines this capability by specifying a proper value in byte 4 of the Declare a Receiver SYS call.) If a non-privileged sender attempts to send a message to a receiver which is accepting messages only from privileged sending jobs, the monitor does not queue the message and returns the ILLEGAL SYS() USAGE error to the non-privileged sender. The sending job can be either privileged or non-privileged if the receiver is capable of receiving messages from any sending job.

7.2.12.3 Removing a Receiver — Privileged (FØ=18)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(18%), the send and receive a message code
3	CHR$(Ø%) to remove a receiving job from the monitor table of receiving jobs.
4	CHR$(N%) where N% is the number of the job to remove or is Ø to remove the job executing the call.
5-3Ø	Not used.

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE An attempt by a non-privileged job to execute this call.	18

Discussion:

This function removes the job number and logical name from the receive table. All pending messages are lost.

7.2.13 Poke Core

Privileged (FØ=-6)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-6%), the poke core code
3-4	CHR$(A%)+CHR$(SWAP%(A%)) where A% is the address to change
5-6	CHR$(V%)+CHR$(SWAP%(V%)) where V% is the value to insert at the address specified by bytes 3 and 4.
7-3Ø	Not Used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
PROTECTION VIOLATION The job executing the call is not operating under account [1,1] or the address specified in the call is an odd value.	10

Discussion:

This call changes a word in the monitor part of memory to the value the user specifies. Obviously, this is a very dangerous capability, and it is, therefore, heavily protected. It can only be called from a job running on account [1,1].

The poke call allows only full word changes. If the user desires a byte change, he must read the word (using the PEEK function), change the desired byte, and rewrite (using the POKE call) the entire word.

7.2.14 Set Logins

Privileged (FØ=-19)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-19%), the set logins code
3	CHR$(N%) where N% is the number of logged in jobs to allow
4-3Ø	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

Meaning	ERR Value
ILLEGAL SYS() USAGE An attempt by a non-privileged job to execute this call	18

Discussion:

This function sets the number of allowable logins to the number specified in byte 3. If N is Ø, the number set is 1. If N is greater than the system JOBMAX set at start up time, then the number set is the value of JOBMAX.

7.2.15 Accounting Information

7.2.15.1 Read or Read and Reset Accounting Data

Both Privileged (FØ=14) and Not Privileged

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(14%), the read or read and reset accounting data code
3-4	CHR$(N%)+CHR$(SWAP%(N%)) where N% is the index number of the account to read. If N% is Ø, read the account specified in bytes 7 and 8.
5-6	CHR$(N%) where N% is Ø to indicate read only and is non-zero to indicate read and reset. If the job executing this call is not privileged, the system does not access this word and performs only a read operation.
7-8	Project-programmer number. Used only if bytes 3 and 4 are Ø. See Section 7.2.3 for a description of each byte.
9-22	Not used
23-24+	Device name; must be a disk. A zero in both bytes indicates the system disk.
25+	Unit number
26+	Unit number flag
27-3Ø	Not used

Date Returned:

Byte(s)	Meaning
1-4	Internal coding
5-6	Number of blocks owned by the account read
7-8	Project-programmer number of the account read
9-12	Password of the account read; in Radix-5Ø format
13-14	Low order word (16 bits) of the CPU time (in tenths of seconds) used by the account
15-16	Connect time (in minutes) used by the account
17-18	Low order word (16 bits) of kilo-core ticks used by the account
19-2Ø	Device time (in minutes) used by the account

Byte(s)	Meaning
21-22	High order bits for CPU time and kilo-core ticks. See the discussion for an explanation of how the values are stored.
23-26	Device and unit information unchanged
27-28	Disk quota in number of blocks; Ø means unlimited quota
29-3Ø	User file directory cluster size

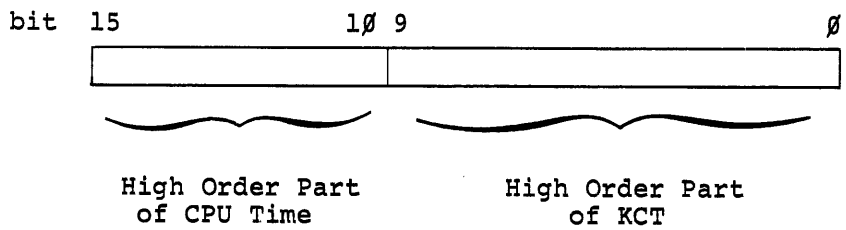
Possible Errors:

Meaning	ERR Value
CAN'T FIND FILE OR ACCOUNT The project-programmer number specified does not exist on the disk or the index specified is greater than the number of accounts on the disk.	5
ILLEGAL SYS() USAGE Device specified is not a disk.	18

Discussion:

This FIP call is the only one provided in RSTS/E to lookup accounts on a disk. By starting the index (bytes 3 and 4) at 1 and incrementing it for each call, the user program can retrieve the project-programmer number of every account on the disk. See the description of the MONEY system program (Section 6.5) for a discussion of the accounting information.

The word returned in bytes 21 and 22 holds the high order bits of CPU time and kilo-core ticks. The bottom 1Ø bits of this word apply to kilo-core ticks, and the top 6 bits apply to CPU time. Graphically, the word looks like the following:



If a non-privileged program executes this call, the system forces the following bytes in the data passed to the values shown.

3 and 4	Ø	Look up the account specified in bytes 7 and 8
5 and 6	Ø	Read only
7 and 8	current PPN	Look up data for current project-programmer number

If a privileged program executes this call and bytes 5 and 6 of the data passed are non-zero, the following account information is read and reset to zero.

CPU time
kilo-core ticks
connect time
device time

7.2.15.2 Accounting Dump — Privileged (FØ=-15)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-15%), the accounting dump code
3-4	Project-programmer number of the account to which the system dumps the accumulated usage data. See Section 7.2.3 for a description of each byte. If both bytes are zero, the data is dumped to the current account.
5-3Ø	Not used

Data Returned: No meaningful data is returned.

Possible Errors:

	Meaning
CAN'T FIND FILE OR ACCOUNT	The account specified in bytes 3 and 4 does not exist.

Discussion:

This function allows a program to dump accumulated accounting data to the account specified in bytes 3 and 4. This capability enables user callable utility programs to run on an account different from the account which called them and still charge the calling account for the time accumulated by the utility. For example, the SPOOL program must run on a privileged account, but is callable by non-privileged users through the QUE command. It is desirable that the calling user be charged for the time the SPOOL program accumulates processing the job. The SPOOL program could take advantage of this SYS call to effect this process.[1]

[1]The version of SPOOL released with RSTS/E does not perform this function.

7.2.16 Directory Look Up

The SYS function calls described in this section look up file names under programmed control. Although only two codes are available, four different types of operation are possible. As a result, four descriptions appear in this section.

The four types of operation return the data in the same format as described below.

Data Returned:

Byte(s)	Meaning
1-4	Not used.
5-6	Same as data passed. (Project-programmer number)
7-10	File name in Radix-50 format. See Section 7.2.2 for a description of converting a string in Radix-50 format.
11-12	Extension in Radix-50 format.
13-14	Length in blocks or sectors. (Not used for special magtape look up described in Section 7.2.16.2).
15	Protection code of the file
16	Ø
17-18	For disk, the date of last access; for tape, the date of creation.
19-20	For disk, the date of creation; for tape, not used.
21-22	For disk, the time of creation; for magtape, the project-programmer number of the file.
23-26	Same as data passed. (Device name, unit number, and flag byte.)
27-28	For disk, the file cluster size; for tape, not used.
29	Number of entries returned: for disk, 8; for tape, 6. (Not returned if FØ is 17.)
30	Not used.

7.2.16.1 Directory Look Up on Index — Not Privileged (FØ=15)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(15%), the directory look up on index code
3-4	CHR$(N%) +CHR$(SWAP%(N%)) where N% is the index of the file to read. If N% is Ø, return the data for the first file in the directory. If N% is x, return the data for the x+1 file in the directory. On magtape, N% must be Ø to rewind the tape before reading the first file. See Section 7.2.16.2 for a description of magtape operations. On DECtape, N% must be Ø to read the directory blocks from the tape before reading the first file. Subsequent calls where N% is not zero read the directory from the BUFF.SYS file.
5-6	Project-programmer number of the directory to look up. If both bytes are Ø and the device specified in bytes 23 and 24 is magtape, the call returns information for each file read. If the device specified in bytes 23 and 24 is DECtape, the call does not use these bytes but returns information for each file read. See Section 7.2.3 for a description of these bytes.
7-22	Not used
23-24+	Device name for look up
25+	Unit number
26+	Unit number flag
27-30	Not used

Data Returned: See introductory material for Section 7.2.16.

Possible Errors:

Meaning	ERR Value
CAN'T FIND FILE OR ACCOUNT The account specified does not exist on the device specified or no more files exist on the account (the index value is greater than the number of files on the account).	5

Meaning	ERR Value
DEVICE NOT FILE STRUCTURED The device specified in the call is not a file structured device.	30
VARIOUS DEVICE DEPENDENT ERRORS The call also returns device dependent errors such as DEVICE HUNG and DISK PACK NOT MOUNTED.	

Discussion:

The CATALOG system command employs the same routines as this call to print a directory listing. The ordering of the files in the listing is by index value from the lowest to the highest. The user can therefore determine the index value for a certain file by counting its position in a CATALOG listing and subtracting one.

If the device specified is magtape, the monitor, after reading a file label, skips to the end of the file on the tape to determine the number of blocks in the file.

7.2.16.2 Special Magtape Directory Look Up Not Privileged (FØ=15)

Data Passed: Same as that described in Section 7.2.16.1 except for the following data.

Byte(s)	Meaning
5-6	Both bytes are CHR$(255%) to execute the special magtape directory look up
23-24	Must be MT

Data Returned: See introductory material for Section 7.2.16.

Possible Errors:

Meaning	ERR Value
CAN'T FIND FILE OR ACCOUNT No more files exist on the tape.	5
DEVICE NOT FILE STRUCTURED The device specified in bytes 23 and 24 is not file structured.	30

Discussion:

The standard directory look up call (described in Section 7.2.16.1) executed on a magtape unit results in the following actions by the monitor:

a. Reads one record from the tape (a label record).
b. Spaces the tape forward to the next end of file record and calculates the number of records in the file.
c. Returns the directory information if the account number of the file matches the one specified in the call or if both bytes in the account specification in the call are zero.

When the monitor executes the action described in statement a, the tape must be positioned immediately before a label record. Otherwise, an error is generated or garbage information is returned.

In an application program which must search a tape for a specific file and read each specific file found, the OPEN FOR INPUT statement necessitates a rewind operation. The OPEN FOR INPUT statement executed on a file structured magtape generally causes the following actions.

a. Reads one record from the tape which must be a label record.

b. If the read operation is successful, then opens the file and returns control to the user program.

c. If the read operation is unsuccessful and this is the first label read, then rewinds the tape and executes the action described at a.

d. If the logical end of tape is detected, return an error.

e. If the label read does not match, skips to the end of this file and executes the action at a.

The required rewind operations consume time and are clearly unwanted.

To avoid the rewind operations, the application program can execute the special magtape directory look up call and perform certain actions. By specifying bytes 5 and 6 both as CHR$(255%) in the call, the program causes the following actions by the monitor.

a. Reads from the tape a record which must be a label record.

b. Backspaces one record which leaves the tape in a position to read the label record again.

c. Returns the directory information (except for file length) to the program.

To take advantage of this special action, the program can perform the following actions.

a. Determine from the information returned whether the file is the one required.

b. If the file is required, execute the OPEN FOR INPUT statement using the file name and requesting no rewind. The action executes without a rewind because the tape is positioned properly. If the file is not required, space the tape forward to the next end of file record (see Section 12.3.7 of the BASIC-PLUS Language Manual) before executing another call.

c. After processing the required file, execute a CLOSE statement to position the tape at the end of file record and to be ready to execute another call.

The special look up call returns directory information on each file read regardless of its account number. However, the OPEN FOR INPUT statement must specify the correct account number if the account number of the file does not correspond to the current account number.

7.2.16.3 Disk Directory Look Up by File Name — Not Privileged (FØ=17)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(17%), the disk directory look up by file name and disk wildcard directory look up code. See Section 7.2.16.4 for a description of the latter call.
3-4	Both bytes must be CHR$(255%)
5-6	Project-programmer number of the file to look up. See Section 7.2.3 for a description of each byte.
7-10	File name in Radix-50 format. See Section 7.2.4.1 for a description of converting a string to Radix-50 format.
11-12	Extension in Radix-50 format
13-22	Not used
23-23+	Device name; must be disk
25+	Unit number
26+	Unit number flag
27-30	Not used

Data Returned: See introductory material for Section 7.2.16.

Possible Errors:

Meaning	ERR Value
ILLEGAL FILE NAME File name in bytes 7 through 10 is missing.	2
CAN'T FIND FILE OR ACCOUNT The device specified in bytes 23 and 24 is not a disk or the file specified does not exist on the specified disk.	5

Discussion:

This call works only on disk files and returns information for the specified file.

7.2.16.4 Disk Wild Card Directory Look Up Not Privileged (FØ=17)

Data Passed: Same as that described in Section 7.2.16.3 except for the following data.

Byte(s)	Meaning
3-4	CHR$(I%) + CHR$(SWAP%(I%)). If I% is Ø, return the data for the first file which matches the wild card specification. If I% is x, return the data for the x + 1 file which matches the wild card specification.
7-10	Radix-50 representation of a wild card file name specification where an * character can replace the file name or a ? character can replace any character in the file name. Used with the extension in bytes 11 and 12 to create a wild card file specification. See Section 7.2.4.1 for a description of converting a string to Radix-50 format.
11-12	Radix-50 representation of a wild card extension specification where an * character can replace the extension or a ? character can replace any character in the extension. Used with the file name in bytes 7 through 10 to create a wild card file specification. See Section 7.2.4.1 for a description of converting a string to Radix-50 format.

Data Returned: See Introductory material for Section 7.2.16.

Errors Possible:

Meaning	ERR Value
ILLEGAL FILE NAME No file name appears in bytes 7 through 10.	2
CAN'T FIND FILE OR ACCOUNT The device specified in bytes 23 and 24 is not a disk or no match exists for the index value given in bytes 3 and 4.	5
DISK PACK IS LOCKED OUT The disk is in the locked state and the account under which the call is executed is not privileged.	22

Discussion:

This call allows a program to supply a wild card specification and to increment an index value to gain directory information for all occurrences of files matching the wild card specification. The following are typical wild card specifications and their meanings.

FILE??.*	All files with FILE as the first four characters in the name and with any extension (including no extension)
*.BAS	All files with BAS extensions
*.BA?	All files with BA as the first two characters in the extension

The program supplies an index of Ø and executes the call. The system returns directory information for the first file which matches the wild card specification. The program can increment the index by 1 and execute the call again to gain directory information for second and subsequent matching occurrences of files. The system returns error number 5 to indicate no more matching occurrences exist in the account. The entire procedure releives the program of the overhead required to translate each file name in the directory and to compare for a match.

7.2.16.5 General Guidelines for Usage of Directory Look Up Calls

The following conditions apply to executing one of the directory look up calls described in Section 7.2.16.

If a program specifies either DECtape or magtape, the monitor assigns the related unit to the calling job while the call executes. The unit remains assigned after the call completes.

When a program repeatedly executes one of the calls on disk and increments the index for each repetition, the execution time increases for each successive call. The increase occurs because the monitor must read the file name blocks for indices numbered Ø through N-1 before it reads the file name block for index number N. The process is the only one possible since the index value has no other relationship to the actual disk address of the file name block.

When a program repeatedly executes one of the calls on a system disk structure having multiple public disks, the increase in execution time related to the index value is more critical. Since the monitor has no means of determining how many files exist on each unit of a multiple public disk structure, it must read the file name blocks of each unit beginning at unit Ø until the Nth file is read. Therefore, on such a system, execution time can be decreased if the program executes the call repeatedly on each specific unit of the public structure (for example, DKØ:, DK1:, and upward) rather than on the entire public structure (SY:).

7.2.17 Monitor Tables and FCB or DDB Information

The two monitor table SYS system function calls to FIP return to the user program either an address or a data value. They are commonly employed with the PEEK function to read various system parameters and tables which give configuration and run time information. Because it is beyond the scope of this manual to describe the monitor, this section only briefly describes the information returned by the monitor table functions. Section 7.3 describes the use of the PEEK function for certain convenient programming operations.

In this section, each item of information described is denoted by a name in all upper case letters. This name is the same one used to identify the information in the RSTS/E assembly listings. If the name is enclosed by parentheses, the information returned is an address of the data described. If the name is not enclosed by parentheses, the information returned is the actual data value. For example, the get monitor table (part I) call returns NULINE in byte 3. The value returned is the number of terminal lines configured on the system. However, in bytes 11 and 12 is (JOBTBL), the address of the table of jobs. The user program can inspect the address by using the PEEK function.

7.2.17.1 Get Monitor Tables - Part I Not Privileged (FØ=-3)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-3%), the get monitor tables (part I) code
3-30	Not used

Data Returned:

Byte(s)	Meaning
1-2	Not used
3	NULINE - the number of keyboards configured on the system
4	MAXCNT - the maximum job number allowed during the current time sharing session
5-6	(DEVCNT) - the address of the table of maximum unit numbers for all devices configured on the system
7-8	(DEVPTR) - the address of the table of pointers to device DDBs.
9-10	(CORTBL) - the address of the memory allocation table
11-12	(JOBTBL) - the address of the job table
13-14	(JBSTAT) - the address of the job status table
15-16	(JBWAIT) - the address of the table of job wait flags
17-18	(UNTCLU) - the address of the table of unit cluster sizes for mounted disks
19-2Ø	(UNTCNT) - the address of the status table of all disk units on the system
21-22	(SATCTL) - the address of the table of free block counts for each disk (other than swapping disks) on the system. The table SATCTL contains the least significant word (16 bits) of the double precision unsigned integer (32 bits) count of free blocks. Each word applies to a separate disk unit.
23-24	(TBLNAM) - the address of the program name table
25-26	(SATCTM) - the address of the table of free block counts for each disk (other than swapping disks) on the system. The table SATCTM contains the most significant word (16 bits) of the double precision unsigned integer (32 bit) count of free blocks. Each word applies to a separate disk unit.
27-28	Current date in internal format
29-3Ø	Not used

Possible Errors: No errors are possible.

7.2.17.2 Get Monitor Tables - Part II

Not Privileged (FØ=-12)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-12%), the get monitor tables (part II) code
3-30	Not used

Data Returned:

Byte(s)	Meaning
1-2	Not used
3-4	(FREES) - the address of the table of free (small and large) buffer information
5-6	(DEVNAM) - the address of the device name table
7-8	(TTILST) - the address of the keyboard input CSR table
9-10	(TTSLST) - the address of the modem default list
11-12	(TTYHCT) - the address of the number of hung terminal errors since system start up
13-14	(JOBCNT) - the address of: current number of jobs, current job number limit, configured job number limit
15-16	Reserved
17-18	(ERRCTL) - address of error logging control data
19-20	(TBLMES) - the address of the table of eligible message receiving jobs
21-22	(LOGNAM) - the address of the table of logical names for mounted disks
23-24	(JOBRES) - the address of the table of job residency pointers
25-3Ø	Not used

Possible Errors: No errors are possible.

7.2.17.3 Get Open Channel Statistics — Not Privileged (FØ=-8)

Data Passed:

Byte(s)	Meaning
1	CHR$(6%), the SYS call to FIP
2	CHR$(-8%), the get open channel statistics code
3	CHR$(N%) where N% is the channel number (between Ø and 15) of either the FCB or DDB
4-30	Not used

Data Returned:

Byte(s)	Meaning
1-2	Not used
3-4	Word 1 of either the FCB or DDB
5-6	Word 2 of either the FCB or DDB
.	.
.	.
.	.
27-28	Word 13 of either the FCB or DDB
29-30	Word 14 of either the FCB or DDB

Possible Errors:

Meaning	ERR Value
I/O CHANNEL NOT OPEN The channel specified in byte 3 of the call is not open.	9

Discussion:

The layout of an FCB and DDB for each device configured on the system is in the listing of the TBL.LST file created during system generation.

The use of this call is rendered obsolete by the STATUS variable described in Section 12.8.1 of the BASIC-PLUS Language Manual.

7.3 THE PEEK FUNCTION

The PEEK function allows a privileged user to examine any word location in the monitor part of memory. The user program can examine words in the BASIC-PLUS run time system, in small or large buffers, in the resident portion of the file processor, and in the low core and tables section of memory. The function does not allow a user program to examine the contents of another user's program.

A privileged program executes the PEEK function in the following manner.

```
I% = PEEK(J%)
```

The function takes an (even) integer argument (J%) and returns an integer value (I%). The value returned is the contents of the address in memory specified by the argument. Since, on the PDP-11 computer, addresses of word locations are always even, and odd addresses indicate byte locations, the user must always be careful to specify an even integer address as the argument to PEEK. To examine an odd address, the program must specify the next lower integer as the argument to PEEK. The contents of the odd address is the high order byte of the value returned by PEEK.

The PEEK function is normally used to examine either addresses returned by get monitor tables calls or addresses of fixed monitor locations.

The following are possible errors generated by incorrect usage of the PEEK function.

Meaning	ERR Value
PROTECTION VIOLATION An attempt by a non-privileged user to execute this call	10
UNIBUS TIMEOUT FATAL TRAP The address specified as an argument to PEEK is either odd or out of range of the allowed addresses.	33

7.3.1 Fixed Locations in Monitor

The following information is stored in fixed locations in the monitor part of memory and is obtained by executing a PEEK(X%) where X% is the address shown.

Address (decimal)	Name	Meaning
36(word)	IDATE	The date when the system was last started by START
38(word)	ITIME	The time of day when the system was last started by START
512(word)	DATE	Current system date
514(word)	TIME	Current time of day
518(byte)	JOB	Job number times 2 of the job currently running (always is the user's own job number)
520(word)	JOBDA	Address of the job data block (JDB) of the currently running job (always the user's own job data block)

7.3.2 Useful Routines

7.3.2.1 Finding the Current Project-Programmer Number

Two methods exist for a program to determine the project-programmer number under which it is running. The first method, the only one available to non-privileged users, is to execute the read or read and reset accounting data FIP function (FØ=14). If the index and the project-programmer number passed in the call are both Ø, the project-programmer number returned in bytes 5 and 6 is that of the program executing the call. This first method is slow because it requires FIP handling and possibly requires one or more disk accesses.

The second method, available only to privileged users, is faster and involves executing the PEEK function to examine two bytes in the job data block (JDB) of the job. The contents of the JDB bytes 26 and 27 is the project-programmer number of the current job. The high byte returned by PEEK is the project number; the low byte is the

programmer number. The address of the JDB of the currently running job is in the fixed monitor location JOBDA (address 52Ø). The following statement

```
A% = PEEK(PEEK(52Ø%) + 26%)
```

puts the project-programmer word into the variable A%. The following statements put the project number in B% and the programmer number in C%.

```
B% = SWAP%(A%) AND 255%
C% = A% AND 255%
```

7.3.2.2 Determining an Attached or Detached Condition

Only one method exists for a program to determine whether or not it is attached to a terminal. It is beyond the scope of this manual to describe the mechanics of the method. It is sufficient to say that the method determines whether or not a console keyboard exists for the job. The following statements show the procedure.

```
10 IF ((PEEK(PEEK(PEEK(PEEK(520%)))+2%) AND 255%)=(PEEK(518%) AND 255%)
        AND
        (PEEK(PEEK(PEEK(PEEK(520%)))+30%) AND 8192%)=8192%)
        THEN GOTO 20
        ELSE GOTO 30
20 REM: THIS LINE IS REACHED ONLY IF THE JOB IS ATTACHED TO A TERMINAL
25 PRINT "ATTACHED"
        : STOP
30 REM: THIS LINE IS REACHED ONLY IF THE JOB IS DETACHED
35 STOP
```

Line 10 determines the attached or detached condition. The parentheses are important.

Once a program determines that it is attached to a terminal, it normally is not necessary to find the keyboard number. The program has normal access to the terminal by executing either an OPEN "KB:" statement on a free channel or a PRINT or INPUT statement without a channel specified.

APPENDIX A
RSTS/E CONSISTENCY ERROR MESSAGES

During the RSTS/E system build procedures and during each execution of RSTS/E initialization routines, many checks are made to determine the consistency of system structures. The existing structures are compared to their definitions and references as they appear in other parts of the system. The checks performed must always be successful. If they are not successful, a consistency error has been detected and the system is corrupted. These errors can never occur on a sound system. If an error does occur, contact a software support specialist.

If the error is considered fatal, the initialization code prints the following message.

```
FATAL RSTS VØ5 SYSTEM INITIALIZATION ERROR!
```

If the initialization code finds any problem with the RSTS/E Core Image Library (CIL), the system prints the following additional message.

```
FATAL ERROR OCCURRED DURING THE CILUS PHASE OF SYSTEM
INITIALIZATION; THERE IS NO RECOVERY.
```

The following list gives the messages generated when consistency errors are detected. The list is in alphabetical order for ease of reference.

```
ATTEMPT TO ASK FOR OPTION WHEN CILUS PHASE NOT DONE.
BAD DIRECTORY DETECTED DURING CLEAN.
CHECKSUM ERROR IN CIL INDEX.
DEVICE BOOTED DOES NOT MATCH SYSTEM DEVICE.
DEVICE BOOTED IS NOT A LEGAL RSTS VØ5 SYSTEM DEVICE.
DEVICE ERROR WHEN READING CIL INDEX.
DOUBLE OCCURRENCE OF SOME SYSTEM IMAGE.
```

EXISTING SYSTEM FILE EMPTY OR NON-CONTIGUOUS.
FILE [0,1].SYS MISSING FROM SYSTEM DISK.
FORMAT ERROR IN CIL INDEX.
ILLEGAL BLOCK SIZE IN CILUS BOOTSTRAP PARAMETERS.
ILLEGAL SWAPPING DISK DETECTED.
INIT BUG - ATTEMP TO DELETE NON-EXISTENT FILE.
INIT BUG - FAILED TO CREATE RSTS.CIL ON 2ND TRY.
INIT BUG - FILE EXISTED WHEN TRYING TO CREATE.
INIT BUG - INSUFFICIENT DIRECTORY SPACE FOR CREATE.
INIT BUG - INSUFFICIENT DISK SPACE FOR CREATE.
INIT BUG - SATT.SYS NON-EXISTENT AT TIME OF WOMP.
INIT BUG - UNABLE TO REBUILD DISK.
INIT WAS INCORRECTLY ASSEMBLED OR LINKED.
INSUFFICIENT DIRECTORY SPACE FOR SYSTEM FILES.
INSUFFICIENT DISK SPACE FOR [0,1] DIRECTORY.
ONE OR MORE CRITICAL MODULES MISSING FROM CIL.
PACK CLUSTER SIZE IS NOT 1, 2, 4, 8 or 16.
REQUIRED FILE BADB.SYS DOES NOT EXIST.
REQUIRED FILE SATT.SYS FILE DOES NOT EXIST.
REQUIRED LIBRARY ACCOUNT [1,2] DOESN'T EXIST.
RSTS CIL IS NOT ON A CLUSTER BOUNDARY.
RUN-TIME SYSTEM MISSING FROM CIL.
SYSTEM DISK SAT SIZE NOT EQUAL TO COMPUTED SIZE.
SYSTEM FILE CONTAINS BAD BLOCKS - CANNOT REFRESH.
SYSTEM MODULE IS MISSING FROM CIL.
TOO MANY NON-SYSTEM IMAGES IN CIL.

APPENDIX B
SYSLOD ERROR MESSAGES

The system loader program SYSLOD transfers a linked core image library from one device to a disk as a contiguous core image library. The user runs SYSLOD from magtape or DECtape during the system generation procedure. The user must consult this section for error messages and possible recovery procedure if errors are generated during SYSLOD execution.

Error messages issued by the SYSLOD program can be of two types: recoverable errors and non-recoverable errors.

B.1 RECOVERABLE ERRORS

The following errors are diagnosed and printed by SYSLOD. Once the error message is printed, SYSLOD restarts by identifying itself again, and printing the # (input request) character at the keyboard. The user should retry the most recent command, making the indicated corrections. Error messages for recoverable errors are preceded by one of the following:

```
CIL dev
LICIL dev        (dev represents the device mnemonic)
```

depending upon whether an error has been detected in the CIL (output) or LICIL (input) side of the most recent command string.

SYNTAX ERROR

This message is printed if the command input line contains a syntax error.

TOO MANY SWITCHES

SYSLOD does not accept switches on the input side of the command string. If too many switches are specified on the output side of the command string for SYSLOD to handle, it issues this message.

UNKNOWN SWITCH

If SYSLOD does not recognize the switch as a valid switch, it prints this message.

SWITCH ERROR

If a switch is used incorrectly, SYSLOD prints this message. Incorrect use of a switch implies specification of an argument when no argument is valid or the lack of an argument when one is required.

SWITCH CONTEXT ERROR

This message is issued when switches are specified incorrectly for their definitions.

ERROR IN SWITCH ARGUMENT

This message is issued when decimal argument for any switch is too large to be contained in 16 bits.

NONEXISTENT DISK OR DISK NOT READY

Either (1) the disk specified in a command string does not exist in the configuration, or (2) the disk exists, but is not ready.

UNKNOWN DISK NAME SPECIFIED

This message is issued when a disk name other than those listed below is specified in a command string.

DF (DFØ through DF7)
DK (DKØ through DK7)
DP (DPØ through DP7)

ERROR WHILE FORMATTING RK DISK

This message is issued whenever an error is detected while formatting an RK disk unit.

A xxx
READY

This message is issued when a problem exists with a peripheral device not in the "ready" state. Check the device and the operating instructions to determine the cause of the error and correct the error. When the problem has been rectified, type YES and the CARRIAGE RETURN key to continue.

B.2 NON-RECOVERABLE ERRORS

The SYSLOD program prints an error message at the keyboard device when a non-recoverable error is encountered during processing. These messages are listed below, along with the action to be taken (if any).

INPUT IS NOT A LICIL

The first line of the input file is not in correct format for a LICIL. The most probable cause of this error is an attempt to transfer a load module (filnam.LDA) instead of the LICIL (filnam.LCL).

END OF FILE BEFORE CIL LINE READ

This error message is issued when SYSLOD reaches end-of-file before detecting the CIL line. If the CIL is being loaded from DECtape, magtape, or disk, it is likely that part of the file has been destroyed and must be rebuilt.

BOOTSTRAP NOT IN BLOCK Ø

This message occurs in Replace mode only (neither the NS nor ZE switch has been specified). In Replace mode, SYSLOD searches

bootstrap parameters to find the CIL to be replaced. If the first block number of the CIL hooked to the bootstrap (location 176 of BOOT) is Ø, then block Ø is not a hooked bootstrap, and this message is issued.

BOOTSTRAP NOT HOOKED TO CIL; CANNOT REPLACE

This message occurs in Replace mode only; (neither the NS nor ZE switch has been specified). If the first block indicator of the bootstrap (location 176 of BOOT) is non-zero, it must be pointing to a CIL. If the first formatted binary line of the "hooked" file is not COMD section #3, then the file is not a CIL.

BLOCK SIZE DISCREPANCY BETWEEN CILUS AND SYSLOD

This error message is issued when the NS switch is used with SYSLOD. Exactly the same parameters must be specified for SYSLOD as were specified for CILUS.

LICIL TOO BIG, NOT ENOUGH RESERVED BLOCKS

This message occurs for either of two conditions:

a) In Replace mode, the new LICIL is larger that the old.
b) In any other mode, the number of reserved blocks (BL:nnnn) is not large enough for the CIL.

1ST LINE NOT COMD SECTION #4 OR 1

After reading the CIL line, SYSLOD begins to load the LICIL. The first formatted binary line after the CIL line must be COMD section #4 or COMD section #1. After each core image is loaded, SYSLOD is set to load a new core image. If the beginning of the new core image is not COMD section #4 or #1, this error message is issued.

COMD SECTION #4 SEQUENCE ERROR

This error message occurs if the LICIL is being loaded from paper tapes, and one or more tapes are out of order.

INPUT ERROR

After a READ, the status in the buffer header has indicated that one of the following errors occurred:

a) invalid line error
b) checksum error
c) character parity error or illegal binary format
d) device parity error

LOGICAL BLOCK SIZE ERROR

This error message is issued when the logical block size specified for the NS switch is not an integral multiple of the physical block size for the disk.

END OF DISK BEFORE CIL COMPLETE

The last block number of the output disk has been written, but the CIL is not complete.

ILLEGAL EMT CALL

An EMT call was made that was not recognizable as a valid DOS/BATCH EMT. Notify your Software Support representative.

FATAL ERROR RETRY

A fatal error has aborted the current operation. It must be retried.

MT DISASTER NXM OR ILC - IRRECOVERABLE

An irrecoverable I/O error has occurred while reading or writing magtape.

IRRECOVERABLE MT PROBLEM

A persistent error has occurred while reading magtape.

LICIL FILE NOT FOUND

The specified LICIL could not be found under UIC [1,1] or UIC [200,200].

NO SPACE FOR CIL

Not enough contiguous disk space is available to create the CIL. Either (1) too many files already exist on the disk, or (2) there are too many bad blocks on the disk.

BLOCK 0 OR BLOCK 1 BAD

Either block 0 or block 1 on the disk is bad; the system cannot be generated, as both these blocks are essential to the disk file structure.

PACK IS TOO BAD

The current disk pack cannot be used, as there are too many bad blocks (BADB.SYS is full).

ESSENTIAL DISK BLOCK HAD I/O ERROR

A MFD, UFD, or bit map block could not be written without encountering I/O errors. Note that these blocks are written after the verification phase.

B.3 NOTES

If the default block size is not used, the block size must be an integral multiple of the default size for the disk.

The COMD (COMmunication Directory) contains a code number that identifies the kind of information that follows. SYSLOD expects the first formatted binary line it receives on input to be identified as code #4 (indicating that it is a LICIL).

On occasion, certain hardware problems such as magtape and RK disk head alignment, UNIBUS time out traps, and disk controller errors cause SYSLOD to halt. The system manager should immediately contact a DIGITAL field service representative.

APPENDIX C

TABULATION OF SHORT FORM CONFIGURATION QUESTIONS

The user initiates the short form of the dialogue by typing S in response to the first question printed by the SYSGEN program. The following list shows the questions numbered in hierarchical order, the possible answers, and comments on SYSGEN actions.

Question Number	Question	Possible Answers	Comment
1	MEDIA?	MT,DT,RK,MM,RP	
2.1	FPP (11/45)?	Y,N	If Y, SYSGEN skips to question 3
2.2	FIS (11/40)?	Y,N	
3	CLOCK?	L,P,C	SYSGEN prints 4.1 if answer is L or P or prints 4.2 if answer is C.
4.1	AC FREQ?	5Ø,6Ø	
4.2	KW11P INTERRUPT RATE?	5Ø,1ØØ,...1ØØØ	
5	CONSOLE TYPE?	ASR33,KSR33, ASR35,KSR35, LA3ØP,LA3ØS, VTØ5,VTØ5B VT5Ø,VT51, LA36	
6	KL11,LC11,DL11A,DL11B'S?	Ø to 15	Do not include the console terminal.
7	DL11C,DL11D'S?	Ø to 31	Do not include the console terminal.
8	DC11'S?	Ø to 32	
9	DL11E'S?	Ø to 31	If the sum of the answers to this question and question number 7 is

Question Number	Question	Possible Answers	Comment
			less than or equal to 31, SYSGEN proceeds to 10. Otherwise, SYSGEN repeats 6 through 9.
10	DJ11'S?	Ø to 16	
11	DH11'S?	Ø to 16	If answer is greater than Ø, SYSGEN proceeds to 11.1. Otherwise, it goes to question 12.
11.1	DH11 UNIT xx TYPE?	AA,AB,AC,AD,AE	If answer is AA or AC, SYSGEN prints 11.1.1 to determine presence of DM11-BB on this DH11 unit. If answer is AB,AD, or AE, SYSGEN prints 11.2. SYSGEN repeats 11.1 for each unit.
11.1.1	DOES DH11 xx INCLUDE A DM11-BB?	Y,N	
11.2	DH11 UNIT xx LINES ENABLED?	Ø to 16	SYSGEN repeats 11.2 for each unit.
12	PSEUDO KEYBOARDS?	Ø to 63	If the sum of answers to 6, 7, 8, 9, and 11.2 is less than or equal to 63, then SYSGEN prints question 13. Otherwise, it repeats from 6.
13	2741'S?	NO,SL,DH or BO	No support; support single line only; support only a DH11 line; support both a single and a DH11 line.
13.1	2741 CODE(S)?	CORR,EBCD,SBCD C360	Any combination of 1 to 4 separated by commas.
14	MULTI-TTY SERVICE?	Y,N	
15	RSØ3'S?	Ø to 4	
16	RSØ4'S?	Ø to 4	SYSGEN prints 17 only if answers to 15 and 16 are Ø. If the sum of the answers to 15 and 16 is greater than 4, SYSGEN repeats 15.

Question Number	Question	Possible Answers	Comment
17	RF/RS11'S?	Ø to 8	If Ø, SYSGEN prints 17.1. Otherwise, SYSGEN prints 18.
17.1	RC11/RS64'S?	Ø to 4	
18	RPØ4'S?	Ø to 8 n/NO	If answer is 2 or greater, appending /NO substitutes the non-overlapped seek driver. SYSGEN prints 19 only if answer is Ø.
19	RPØ2/RPØ3'S?	Ø to 8 n/NO	If answer is 2 or greater, appending /NO substitutes the non-overlapped seek driver.
2Ø	RKØ3/RKØ5'S?	Ø to 8 n/NO	If answer is 2 or greater, appending /NO substitutes the non-overlapped seek driver.
21	SYSTEM DISK?	RF,RK,RP,RB	
22	TU16'S?	Ø to 8	If answer is Ø, question 23 is printed. Otherwise, question 24 is printed.
23	TU1Ø'S?	Ø to 8	
24	DECTAPES?	Ø to 8	
25	PRINTERS?	Ø to 8	
25.1.1	LPn: TYPE?	LP,LS	Each line printer unit must be defined; SYSGEN repeats these questions for each line printer unit.
25.1.2	LPn: WIDTH?	8Ø,132	
25.1.3	LPn: LOWER CASE?	Y,N	
26	LPØ FOR SYSGEN?	Y,N	If Y, SYSGEN prints 26.1. Otherwise, SYSGEN proceeds to 27.
26.1	LISTINGS?	Y,N	
27	CARD READER?	N,CR,CD,CM	If N, SYSGEN omits 27.1.
27.1	CARD DECODE?	Ø29,Ø26,14Ø1	
28	P.T. READER?	Y,N	

Question Number	Question	Possible Answers	Comment
29	P.T. PUNCH?	Y,N	
3Ø	DQ11'S?	Ø to 16	
31	DP11'S?	Ø to 32	
32	DU11'S?	Ø to 32	SYSGEN omits 32.1 if answers to 31 and 32 are Ø.
32.1	278Ø?	Y,N	If N, SYSGEN proceeds to question 33. If Y and the sum of the answers to 31 and 32 is 1, SYSGEN proceeds to question 33.
32.1.1	278Ø INTERFACE?	DP,DU	Printed only if answer to 32.1 is Y and answers to both 31 and 32 are non-zero.
33	NON-SUPPORTED DEVICES?	Y,N	If Y, then SYSGEN prints 33.1 through 33.10. Otherwise, it prints 34.
33.1	DM11-A'S?	Ø to 16	Single speed multiplexer.
33.2	DN11-DA'S?	Ø to 64	Automatic calling unit interface.
33.3	DR11-A,C'S?	Ø to 32	General device interface.
33.4	PA611R'S?	Ø to 16	Typeset reader.
33.5	PA611P'S?	Ø to 16	Typeset punch.
33.6	DTØ3-FP'S?	Ø to 8	Programmable UNIBUS switch.
33.7	DX11'S?	Ø to 4	IBM 360/370 interface.
33.8	GT4Ø (1 ONLY)?	Ø,1	Answer 1 only if GT4Ø does not have its own CPU and is not connected through a standard terminal interface.
33.9	LPS (1 ONLY)?	Ø,1	LAB peripheral system.
33.1Ø	KW11W (1 ONLY)?	Ø,1	Watchdog timer.
34	THE INSTALLATION NAME:		14 characters or less.

Question Number	Question	Possible Answers	Comment
35	MAXIMUM JOBS?	1 to 63	
36	SMALL BUFFERS?	3Ø to 999	1Ø times the maximum number of jobs recommended.
37	BIG BUFFERS?	1 to 8	SYSGEN prints this question only if the answer to question 24 is greater than Ø.
38	RECEIVERS?	Ø to 63	From 4 to 8 is usually sufficient.
39	POWER FAIL?	Y,N	
40	FIP BUFFERING?	Y,N	
41	RESIDENT DISK HANDLING?	Y,N	
42	RESIDENT SYS CALL DISPATCH?	Y,N	
43	RESIDENT SEND/RECEIVE?	Y,N	
44	RESIDENT DIRECTORY LISTER?	Y,N	
45	CCL?	Y,N	If N, SYSGEN proceeds at 46.
45.1	STANDARD CCL TABLE?	Y,NEW,ADD	If NEW or ADD, SYSGEN prints 45.1. Otherwise, it prints 46.
45.1.1	<PROGRAM>,<COMMAND>?		User types program name and command separated by a comma. SYSGEN prints question up to 20 times or until user types /E to end or types /R to restart.
45.2	CCL LISTING ON LPØ:?	Y,N	Printed only if reply to question 26 was Y and to question 45.1 was NEW or ADD.
46	MATH PRECISION?	2,4	
47	FUNCTIONS?	Y,N	
48	TIME FORMAT?	24,AM	
49	PRINT USING?	Y,N	
50	MATRICES?	Y,N	
51	ROLLIN?	Y,N	

APPENDIX D

SYSTEM MODULE SIZES

The following tabulation gives the approximate memory sizes of software modules on RSTS/E. By summing the values given, the total size of the system can be estimated.

Module[1]	Decimal Words	Comments
Monitor Code (Scheduler, FIP, ERRLOG, and other modules)	6ØØØ	All systems
Tables	-	Varies greatly
Monitor 11/45 FPP Support	37	
Monitor Options		
FIP Buffering	4Ø8	Optional
Resident Disk Handling	1388	Optional
Resident SEND/RECEIVE	2Ø7	Optional
Resident SYS Call Dispatch	226	Optional
Resident Directory Lister	242	Optional
Small Buffers	16 per buffer	Ten buffers per job recommended
Big Buffers	256 per buffer	For DECtape support
Receivers	5 per receiver	Optional
Power Fail	32	Optional
278Ø Package	56ØØ	RSTS/278Ø systems only
Terminal Service		
Common	134Ø	All systems
KL11, LC11, DL11A, DL11B	Ø	16 words per keyboard unit
DL11C, DL11D	2Ø	16 words per keyboard unit
DL11E	2Ø	16 words per keyboard unit
DH11	3ØØ	16 words per line enabled at system generation time
Common Modem Control	15Ø	First DM11-BB, DC11, DL11E
DL11E, DC11 Modem Control	2Ø	
DH11 Modem Control	15Ø	

[1]Certain entries share code with or require other modules. Therefore, the user must add those values to attain the total size.

Module	Decimal Words	Comments
Pseudo Keyboards	118	Optional. Add 32 words per PK unit configured.
2741 Support		Optional
Common	22Ø	Any 2741 support
Single line	1ØØ	Single Line 2741 support
DH11	1ØØ	DH11 2741 support
Code table	128 per table	Add 35 words if more than one table; four tables maximum
Multiple Terminal Service	1ØØ	Optional
Disks		
RS Disk (RSØ3/RSØ4)	112	
RF Disk	71	
RC Disk	62	
RB Disk (non-overlapped)	2Ø2	RPØ4
RB Disk (overlapped)	312	
RP Disk (non-overlapped)	111	RPØ2 or RPØ3
RP Disk (overlapped)	213	
RK Disk (non-overlapped)	82	
RK Disk (overlapped)	278	
Disk Common	14	All systems
Disk Optimization	43	If any RK, RP, or RB disk
Tape		
Magtape - TU16	625	Add 16 words per drive
Magtape - TU1Ø	64Ø	Add 16 words per drive
DECtape	4Ø5	Add 16 words per drive
Other Peripherals		
Line Printer	397	Add 16 words per unit of any printer type
Card Reader - CD11	265	Add 16 words for DDB
Card Reader - CR11	238	Add 16 words for DDB
Card Reader - CM11	238	Add 16 words for DDB
Card Decode Table	47	Ø26, Ø29, or 14Ø1 codes
Card Reader Buffer	82	Any card reader
Paper Tape Reader	81	Add 16 words for DDB
Paper Tape Punch	54	Add 16 words for DDB
BASIC-PLUS		
Compiler and RTS	11ØØØ	All systems
CCL Common	46	Optional
Standard CCL Table	69	Optional. Add 1Ø words per additional CCL command

Module	Decimal Words	Comments
BASIC-PLUS (cont.)		
Math Package		
MA2	246Ø	See Section 2.7.25 for the description of math packages.
MA2X	1989	
MA2I	2126	
MA2IX	1655	
MA2F	193Ø	
MA2FX	1586	
MA4	359Ø	
MA4X	2844	
MA4F	246Ø	
MA4FX	199Ø	
Print Using	727	Optional
Matrices	754	Optional
Record I/O	297	All Systems

APPENDIX E
CHARACTER GENERATION FILE

The line printer spooling program SPOOL requires the character generation file CHARS.QUE. The file is a virtual core array and is stored on the system disk during the system generation procedure when the user executes the BUILD program with the SPLER.CTL file. If for any reason the file is damaged or destroyed, the user can recreate it by running the CHARS program.

To create the CHARS.QUE file, first ensure that the old copy is deleted from the system library. The CHARS program terminates with an error if a file named CHARS.QUE exists in the system library directory. Next, run the CHARS program by typing the following command.

```
RUN $CHARS
READY
```

After terminating, CHARS returns control to the monitor as indicated by the READY message.

APPENDIX F

HARDWARE BOOTSTRAP PROCEDURES

Bootstrapping a device involves using the central processor unit (CPU) console switches to access and initiate a hardware loader. The hardware loader contains machine instructions for reading a special record from the device. The record, called a bootstrap record, is transferred into memory and executes a specially designed software program. For the bootstrap operation to succeed, the device accessed must be on line and ready; the medium accessed must contain a proper bootstrap record; and the console terminal must be on line.

The console switches and their usage are described in Chapter 8 of the PDP-11/45 Processor Handbook and the PDP-11/40 Processor Handbook. The bootstrap procedure to use depends upon the type of hardware bootstrap device on the system. Table F-1 summarizes the addresses needed to bootstrap a device. The detailed procedures to bootstrap a device are presented according to the types of hardware bootstrap devices available.

Table F-1

Summary of Hardware Bootstrap Addresses

Device to Bootstrap	Bootstrap Type			
	BM873-YA	BM873-YB	MR11-DB	BM792-YB[1]
RF11 disk	773ØØØ	773136	7731ØØ	777462
RK11 disk cartridge	773Ø1Ø	773Ø3Ø	77311Ø	7774Ø6
RPØ3 disk pack	7731ØØ	77335Ø	773154	776716
RPØ4 disk pack	-	77332Ø	-	-
TM11/TU1Ø magtape	773Ø5Ø	77311Ø	773136	[2]
TMØ2/TU16 magtape	-	77315Ø	-	-
TC11/TU56 DECtape	773Ø3Ø	773Ø7Ø	77312Ø	777344

[1] For the BM792-YB loader, set the address 7731ØØ in the Switch Register, depress the LOAD ADRS switch, set the value from the table in the Switch Register, and press the START switch.

[2] To bootstrap a magtape, use the loading routine described in Section F.4.1.

F.1 BM873-YA PROCEDURE

If the BM873-YA Restart/Loader is on the system, perform the following steps.

Move the CPU Console ENABLE/HALT switch to its HALT position and back to its ENABLE position.

Set the CPU Switch Register to one of the following values.

773ØØØ	for RF11 disk
773Ø1Ø	for RK11 disk cartridge
7731ØØ	for RPØ3 disk pack
773Ø5Ø	for TM11/TU1Ø magtape
773Ø3Ø	for TC11/TU56 DECtape

Depress the CPU LOAD ADRS switch.

Depress the CPU START switch.

F.2 BM873-YB PROCEDURE

If the BM873-YB Restart/Loader is on the system, perform the following steps.

Move the CPU Console ENABLE/HALT switch to its HALT position and back to its ENABLE position.

Set the CPU Switch Register to one of the following values.

773Ø3Ø	for RK11 disk cartridge
773136	for RF11 disk
77332Ø	for RPØ4 disk pack
77335Ø	for RPØ3 disk pack
7731lØ	for TM11/TU1Ø magtape
77315Ø	for TMØ2/TU16 magtape
773Ø7Ø	for TC11/TU56 DECtape

Depress the CPU LOAD ADRS switch.

Depress the CPU START switch.

F.3 MR11-DB PROCEDURE

If the MR11-DB Bulk Storage Loader is on the system, perform the following steps.

Move the CPU Console ENABLE/HALT switch to its HALT position and back to its ENABLE position.

Set the CPU Switch Register to one of the following values.

773100 for RF11 disk
773110 for RK11 disk cartridge
773154 for RP03 disk pack
773136 for TM11/TU10 magtape
773120 for TC11/TU56 DECtape

Depress the CPU LOAD ADRS switch.

Depress the CPU START switch.

F.4 BM792-YB PROCEDURE

If the BM792-YB Hardware Loader is on the system, perform the following steps.

Move the CPU Console ENABLE/HALT switch to its HALT position and back to its ENABLE position.

Set the CPU Switch Register to 7731ØØ.

Depress the CPU LOAD ADRS switch.

Set the CPU Switch Register to one of the following values.

777462	for RF11 disk
7774Ø6	for RK11 disk cartridge
776716	for RPØ3 disk pack
777344	for TC11/TU56 DECtape

Depress the CPU START switch.

To bootstrap a TM11/TU1Ø magtape when the system has neither the BM873 nor the MR11-DB loader, the user must manually enter a load routine into memory using the CPU console Switch Register and the DEP switch.

To load the routine, perform the following steps.

Move the CPU Console ENABLE/HALT switch to its HALT position and back to its ENABLE position.

Set the CPU Switch Register to Ø1ØØØØ.

Depress the CPU LOAD ADRS switch.

Load the following contents into memory using the Switch Register and DEP switch.

Address	Contents
Ø1ØØØØ	Ø127ØØ
Ø1ØØØ2	172524
Ø1ØØØ4	ØØ531Ø
Ø1ØØØ6	Ø1274Ø
Ø1ØØ1Ø	Ø6ØØ11
Ø1ØØ12	1Ø571Ø
Ø1ØØ14	1ØØ376
Ø1ØØ16	ØØ571Ø
Ø1ØØ2Ø	1ØØ767
Ø1ØØ22	Ø1271Ø
Ø1ØØ24	Ø6ØØØ3

Address	Contents
Ø1ØØ26	1Ø571Ø
Ø1ØØ3Ø	1ØØ376
Ø1ØØ32	ØØ571Ø
Ø1ØØ34	1ØØ777
Ø1ØØ36	ØØ5ØØ7

Set the Console Switch Register to Ø1ØØØØ.

Depress the CPU LOAD ADRS switch.

Depress the CPU START switch.

If the system reads the tape but halts at address Ø1ØØ34, the device generated a magtape error. The user can try another drive. If the system appears to take no action and halts, verify the accuracy of the routine by using the CPU Console EXAM switch. Use the Switch Register and the DEP switch to correct any erroneous contents. Rewind the tape to its load point before executing the routine again. If no recovery is successful, it will be necessary to have a DIGITAL Field Service representative check the device. If the hardware is working properly, it will be necessary to use a new magtape reel.

HOW TO OBTAIN SOFTWARE INFORMATION

SOFTWARE NEWSLETTERS, MAILING LIST

The Software Communications Group, located at corporate headquarters in Maynard, publishes software newsletters for the various DIGITAL products. Newsletters are published monthly, and keep the user informed about customer software problems and solutions, new software products, documentation corrections, as well as programming notes and techniques.

There are two similar levels of service:

- The Software Dispatch
- The Digital Software News

The Software Dispatch is part of the Software Maintenance Service. This service applies to the following software products:

PDP-9/15
RSX-11D
DOS/BATCH
RSTS/E
DECsystem-10

A Digital Software News for the PDP-11 and a Digital Software News for the PDP-8/12 are available to any customer who has purchased PDP-11 or PDP-8/12 software.

A collection of existing problems and solutions for a given software system is published periodically. A customer receives this publication with his initial software kit with the delivery of his system. This collection would be either a Software Dispatch Review or Software Performance Summary depending on the system ordered.

A mailing list of users who receive software newsletters is also maintained by Software Communications. Users must sign-up for the newsletter they desire. This can be done by either completing the form supplied with the Review or Summary or by writing to:

Software Communications
P.O. Box F
Maynard, Massachusetts 01754

SOFTWARE PROBLEMS

Questions or problems relating to DIGITAL's software should be reported as follows:

North and South American Submitters:

Upon completion of Software Performance Report (SPR) form remove last copy and send remainder to:

Software Communications
P.O. Box F
Maynard, Massachusetts 01754

The acknowledgement copy will be returned along with a blank SPR form upon receipt. The acknowledgement will contain a DIGITAL assigned SPR number. The SPR number or the preprinted number should be referenced in any future correspondence. Additional SPR forms may be obtained from the above address.

All International Submitters:

Upon completion of the SPR form, reserve the last copy and send the remainder to the SPR Center in the nearest DIGITAL office. SPR forms are also available from our SPR Centers.

PROGRAMS AND MANUALS

Software and manuals should be ordered by title and order number. In the United States, send orders to the nearest distribution center.

Digital Equipment Corporation
Software Distribution Center
146 Main Street
Maynard, Massachusetts 01754

Digital Equipment Corporation
Software Distribution Center
1400 Terra Bella
Mountain View, California 94043

Outside of the United States, orders should be directed to the nearest Digital Field Sales Office or representative.

USERS SOCIETY

DECUS, Digital Equipment Computers Users Society, maintains a user exchange center for user-written programs and technical application information. The Library contains approximately 1,900 programs for all DIGITAL computer lines. Executive routines, editors, debuggers, special functions, games, maintenance and various other classes of programs are available.

DECUS Program Library Catalogs are routinely updated and contain lists and abstracts of all programs according to computer line:

- PDP-8, FOCAL-8, BASIC-8, PDP-12
- PDP-7/9, 9, 15
- PDP-11, RSTS-11
- PDP-6/10, 10

Forms and information on acquiring and submitting programs to the DECUS Library may be obtained from the DECUS office.

In addition to the catalogs, DECUS also publishes the following:

DECUSCOPE	-The Society's technical newsletter, published bi-monthly, aimed at facilitating the interchange of technical information among users of DIGITAL computers and at disseminating news items concerning the Society. Circulation reached 19,000 in May, 1974.
PROCEEDINGS OF THE DIGITAL EQUIPMENT USERS SOCIETY	-Contains technical papers presented at DECUS Symposia held twice a year in the United States, once a year in Europe, Australia, and Canada.
MINUTES OF THE DECsystem-10 SESSIONS	-A report of the DECsystem-10 sessions held at the two United States DECUS Symposia.
COPY-N-Mail	-A monthly mailed communique among DECsystem-10 users.
LUG/SIG	-Mailing of Local User Group (LUG) and Special Interest Group (SIG) communique, aimed at providing closer communication among users of a specific product or application.

Further information on the DECUS Library, publications, and other DECUS activities is available from the DECUS offices listed below:

DECUS
Digital Equipment Corporation
146 Main Street
Maynard, Massachusetts 01754

DECUS EUROPE
Digital Equipment Corp. International
(Europe)
P.O. Box 340
1211 Geneva 26
Switzerland

DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASSACHUSETTS 01754