REAL-TIME EXECUTIVE SOFTWARE SYSTEM

HEWLETT-PACKARD COMPANY

Palo Alto, California

The paragraphs or illustrations that have been revised are designated by a vertical bar in the margin.

First Edition
October 1971
Revised March 1972

Printed in U.S.A.

PROGRAMMING AND OPERATING MANUAL

Part No. 02005-90001

REAL-TIME EXECUTIVE SOFTWARE SYSTEM

- IMPORTANT NOTICE -

This manual contains information on Hewlett-Packard Real-Time Executive Software. The reader is assumed to be a programmer familiar with one of the Hewlett-Packard programming languages, ALGOL, Assembler or FORTRAN.

Microfiche No. 02005-90002

Printed OCT 1971

HEWLETT PACKARD

REAL-TIME EXECUTIVE SOFTWARE SYSTEM

MANUAL UPDATING SUPPLEMENT 1 MARCH 1972

MANUAL IDENTIFICATION

Manual Printed: Oct. 1971

Manual Part Number: 02005-90001

SUPPLEMENT DESCRIPTION

This supplement is provided to correct manual errors, to describe differences between the software furnished and that described in the manual, and to provide additional software instructions as required.

When new pages are provided for insertion into the RTE Manual, the corrected or new material will be designated by a bar (|) in the margin.

ITEM	DESCRIPTION

1 Changes have been incorporated into the Real-Time Executive software and Real-Time Executive Relocating Loader. These changes affect operation of the RTE System as described in the following paragraphs.

a. The OF, name, 8 operator request has been changed with regard to global tracks. When this command is used on a program which has been stored on global tracks through the RTE File Manager, those tracks are not released to the system but remain as global tracks. In all other respects the command is unchanged.

b. The RTE Relocating Loader has been modified to check for both End of File (EOF) or End of Tape (EOT) to accommodate inputs from paper tape as well as magnetic tape devices.

c. The RTE Relocating Loader will not permit the on-line replacement of a program if that program is not dormant, is in the time list, or has a non-zero point of suspension. If any of these conditions are true, the loader suspends with the message:

SET PRGM INACTIVE

The program to be replaced must first be set inactive using the operator request:

OF, name, p

Where

name is the name of the program, and $p > 0$, $\neq 8$

The operator reschedules the loader with GO, LOADR

NOTE

Do not enter parameters

d. The program name FMGR has been incorporated into the RTE System, therefore, no user program other than one supplied by Hewlett-Packard should have this name.

e. Changes to the RTE Relocating Loader have necessitated a part number change for the loader.

HP 20792-60001 has been replaced by the new RTE Relocating Loader HP 29022-60001. HP 20792-60001 is obsolete.

The new loader HP 29022-60001 will operate correctly with previous revisions of HP 29016-60001, 2, and 3.

Printed in U.S.A.

ITEM	DESCRIPTION
2	A change has been incorporated into the Real-Time Generator (RTGEN) programs, for both fixed head and moving head systems, to allow the user more control over establishing base page linkages during the Disc Loading Phase of RTGEN. When RTGEN requests the first word available for base page linkages (FWA BP LINKAGE?) set switch 14 = 1 before responding on the teleprinter. Then, as each module is loaded, RTGEN examines switch 14 and if set, creates at each end of a module which crosses a page boundary, a current page-link buffer. The current page-link buffer is used for DBL records that reference addresses across the current page boundary. If this current page-link buffer overflows, then RTGEN reverts to using base page links for the module. The end result is that a base page word is saved for each link established in the current page buffer. NOTE ENT and EXT records and Assembler Language type 3 or 5 programs still use exclusively base page for establishing links.

TABLE OF CONTENTS

Section		Page
Glossary of terms used in this manual		vii
I	GENERAL DESCRIPTION	1-1
	Introduction	1-1
	Software	1-1
	Hardware	1-1
	System Description	1-1
	Multiprogramming	1-1
	Real-Time Core Resident	1-2
	Real-Time Disc Resident	1-2
	Background Core Resident	1-2
	Background Disc Resident	1-2
	Subroutines	1-3
	Program Scheduling (SCHED)	1-3
	Priority Level	1-4
	Program Initiation and Swapping	1-4
	Input/Output Control (RTIOC)	1-4
	Interrupt Processing	1-4
	Input/Output Processing	1-4
	EXEC Communication (EXEC)	1-4
	Operator Requests	1-5
	System Configuration	1-5
	System/Auxiliary Discs	1-5
	Peripheral Discs	1-5
	RTE System Summary	1-5
II	OPERATOR REQUESTS	2-1
	DN (down)	2-1
	EQ,n (EQT status)	2-1
	EQ,n,p (EQT buffering)	2-2
	GO (reschedule)	2-2
	IT (set time)	2-2
	LG (load-and-go)	2-3
	LS (source file)	2-3
	LU (logical unit)	2-3
	OF (terminate)	2-4
	ON (schedule)	2-4
	PR (priority)	2-4
	RT (release tracks)	2-5
	SS (suspend)	2-5
	ST (status)	2-5
	TI (time)	2-6
	TM (set clock)	2-6
	TO (time-out)	2-6
	UP (up)	2-8
	Error Messages	2-8
III	EXEC CALLS	3-1
	Assembler Language Format	3-1
	FORTRAN/FORTRAN IV Format	3-1
	ALGOL Format	3-2
	Read/Write	3-3
	Comments	3-3
	Control Word	3-3
	Binry	3-4
	I/O Control	3-5
	Control Word	3-5
	I/O Status	3-6
	Comments	3-6
	Disc Allocation	3-7
	Comments	3-7
	Disc Release-Program Tracks	3-8
	Disc Release-Global Tracks	3-9
	Comments	3-9
	Program Completion	3-10
	Program Suspend	3-10
	Comments	3-10
	Program Segment Load	3-11
	Comments	3-11
	Program Schedule	3-12
	Comments	3-12
	Waiting and No Waiting	3-12
	Parameters	3-13
	Time Request	3-13
	Comments	3-13
	Execution Time	3-14
	Comments	3-15
	Error Messages	3-15
	Error Codes for Schedule Calls	3-15
	Error Codes for Disc Allocation Calls	3-16
	Error Codes for I/O Calls	3-16
IV	REAL-TIME PROGRAMMING	4-1
	Load-And-Go	4-1
Part 1.	RTE Editor	4-1
	Editor Input/Output Files	4-3
	Editing Process	4-3
	Operating Procedures	4-3
	Messages to Operator	4-3
	Edit Commands	4-4
	Edit Command Formats	4-4
	Edit File Editing	4-5
	Editor Error Messages	4-5

TABLE OF CONTENTS (Continued)

Section		Page
Part 2.	RTE FORTRAN	4-7
	FORTRAN Reference	4-7
	Compiler Operation	4-7
	Messages to Operator	4-7
	RTE FORTRAN Language	4-8
	FORTRAN Control Statement	4-8
	Program Statement	4-9
	Data Statement	4-9
	External Statement	4-10
	Pause and Stop Statements	4-10
	ERRØ Library Routine	4-10
Part 3.	RTE ALGOL	4-11
	ALGOL Reference	4-11
	Compiler Operation	4-11
	Message to Operator	4-11
	RTE ALGOL Language	4-12
	ALGOL Control Statement	4-12
Part 4.	RTE Assembler	4-13
	Assembler Reference	4-13
	Assembler Operation	4-13
	Messages to Operator	4-13
	RTE Assembler Language	4-14
	Assembler Control Statement	4-14
	NAM Statement	4-15
	ORB Statement	4-15
Part 5.	RTE Loader	4-17
	Load-And-Go	4-17
	Background Loading	4-17
	On-Line Modification	4-17
	Limitations	4-18
	Segmented Background Programs	4-18
	New Program Addition	4-18
	Program Replacement	4-18
	Loader Operation (Background)	4-18
	Comments	4-19
	Entering the Relocatable Object Code	4-19
	Matching Entries and Externals	4-19
	End of Loading	4-20
	Relocating Loader Messages	4-20
	Error Messages	4-20
	Additional Messages	4-20
	Listing Programs	4-21
	Loader Operation (On-Line)	4-21
	Comments	4-21
	RTE Debug Library Subroutine	4-22
	Debug Operations	4-22
Part 6.	RTE Relocatable Libraries	4-23
	Re-Entrant Subroutine Structure	4-23
	Format of Re-Entrant Routine	4-23
	Privileged Subroutine Structure	4-24
	Format of Privileged Routine	4-24
	Utility Subroutine Structure	4-24
Part 7.	Segmented Programs	4-25
	RTE ALGOL Segmentation	4-25
	RTE FORTRAN Segmentation	4-25
	RTE Assembler Segmentation	4-25
V	REAL-TIME INPUT/OUTPUT	5-1
	Software I/O Structure	5-1
	The Equipment Table	5-1
	Logical Unit Numbers	5-2
	The Interrupt Table	5-2
	Input/Output Drivers	5-2
	Programmed I/O	5-3
	Planning I/O Drivers	5-3
	Initiation Section	5-3
	Functions of Initiation Section	5-3
	DMA Initialization	5-3
	Completion Section	5-4
	I/O Device Time-Out	5-5
	Sample I/O Driver	5-6
	Privileged Interrupt Processing	5-11
	Privileged Interrupts	5-11
	Special Processing by CIC	5-11
	Privileged Interrupt Routines	5-11
	Sample Privileged Driver	5-11
	Initiation Section, I.xx	5-12
	Privileged Section, P.xx	5-12
	True Completion Section, T.xx	5-13
	Logical Completion Section, C.xx	5-13
VI	RTE SYSTEM INSTALLATION	6-1
	General Information	6-1
Part 1.	Instructions for Planning RTE	6-5
	Input/Output Planning	6-5
	System Disc Planning	6-8
	Fixed Head System Disc	6-8
	Fixed Head Scratch Area	6-8
	Moving Head System Disc	6-9
	Moving Head Scratch Area	6-9
	RTE Configuration Worksheet	6-10
	Initialization Phase	6-10
	Moving Head Initialization	6-10
	Fixed Head Initialization	6-11
	Program Input Phase	6-11

TABLE OF CONTENTS (Continued)

Section		Page
	Parameter Input Phase	6-11
	Disc Loading Phase	6-12
	Equipment Table Entry	6-12
	Device Reference Table	6-12
	Interrupt Table	6-12
	Prepare Tape System	6-13
Part 2.	Fixed Head RTE System Generation	6-15
	Fixed Head Disc RTGEN	6-15
	Operating Procedures	6-15
	Initialization Phase	6-15
	Program Input Phase	6-17
	Parameter Input Phase	6-17
	Disc Loading Phase	6-18
	Initiating RTE From the Disc	6-20
	Loading the RTE System	6-20
	SDUMP	6-20
	SDUMP Error Messages	6-21
	FH RTGEN Error Messages	6-21
Part 3.	Moving Head RTE System Generation	6-25
	Moving Head Disc RTGEN	6-25
	Operating Procedures	6-25
	Initialization Phase	6-26
	Program Input Phase	6-28
	Parameter Input Phase	6-28
	Disc Loading Phase	6-29
	Initiating RTE From the Disc	6-31
	Operating Instructions	6-31
	Error Halts	6-31
	MH RTGEN Error Messages	6-31
Appendix A	TABLES	A-1
	Equipment Table	A-1
	Base Page Communication Area	A-1
	Disc Layout of RTE System	A-2
	Program ID Segment	A-3
	BBDL Listing	A-3
Appendix B	REAL-TIME DISC USAGE	B-1
	Relation to Other Software	B-1
	System Organization	B-1
	System Versus Peripheral Discs	B-1
	System	B-1
	Peripheral	B-1
	RTE Moving Head Driver	B-2
	Read Data	B-2
	Write Data	B-2
	Track Access	B-2
	Record Formats	B-2
Appendix C	SAMPLE RTGEN Fixed Head Disc	C-1
	Fixed Head RTGEN Listing	C-1
	Fixed Head Worksheets	C-4
	Moving Head RTGEN Listing	C-6
	Moving Head Worksheets	C-9
Appendix D	SUMMARY OF EXEC CALLS	D-1
Appendix E	SUMMARY OF ERROR MESSAGES	E-1
	Operator Request Error Messages	E-1
	EXEC Call Error Messages	E-1
	Input/Output Error Messages	E-1
	FORTRAN Compiler Error Messages	E-2
	ALGOL Compiler Error Messages	E-2
	Assembler Error Messages	E-3
	Editor Error Messages	E-3
	Relocating Loader Messages	E-4
	Additional Messages	E-4
Appendix F	LINE PRINTER FORMATTING	F-1
	Carriage Control Channels	F-1
	Automatic Page Eject	F-1
Appendix G	ASCII/OCTAL TABLE	G-1
Appendix H	SUMMARY OF OPERATOR REQUESTS	H-1
Appendix H	SUMMARY OF OPERATOR REQUESTS (Removable)	H-1a
Index		Index-1

LIST OF ILLUSTRATIONS

Figure	Title	Page
1-1.	Core Allocations in the HP RTE System	1-2
4-1.	RTE Library Configuration Diagram	4-23
4-2.	Segmented Programs	4-25
4-3.	Main Calling Segment	4-26
4-4.	Segment Calling Segment	4-26
4-5.	Main-To-Segment Jumps	4-26
5-1.	I/O Driver Initiation Section	5-4
5-2.	I/O Driver Completion Section	5-5
—	Input/Output Configuration Worksheet (Removable)	6-3
—	System Disc Worksheet (Removable)	6-3
—	RTE Configuration Worksheet (Removable)	6-4
—	Input/Output Configuration Worksheet	6-6
—	System Disc Worksheet	6-6
—	RTE Configuration Worksheet	6-7
A-1.	Disc Allocations in RTE System	A-3
A-2.	BBDL Listing	A-4

LIST OF TABLES

Table	Title	Page
1-1.	Real-Time Software	1-1
1-2.	Minimum RTE Hardware	1-2
2-1.	Day of Year	2-7
3-1.	Summary of Request Codes (ICODE)	3-1
5-1.	Equipment Table Entry Diagram	5-1
6-1.	Approximate Number of 64-Word Sectors Required to Store RTE in Relocatable Format	6-10
6-2.	Octal/Decimal Conversion	6-20
A-1.	Equipment Table Entry Diagram	A-1
A-2.	ID Segments	A-3
B-1.	Source Format	B-3

GLOSSARY OF TERMS USED IN THIS MANUAL

ABSOLUTE SYSTEM – The absolute binary code of the Real-Time Executive (stored on logical unit 2).

AUXILIARY DISC – The disc is optional and when used is assigned to logical unit 3. (The absolute binary code of RTE does not reside on the auxiliary disc.) When the auxiliary disc is part of a moving head disc drive, it is contained on one subchannel of that drive. The auxiliary disc has the same status in the RTE as does the system disc in that it is treated as a logical extension of the system disc.

CONTROLLER – Two computer interface cards plugged into adjacent I/O slots, and connected to the disc drive with a cable.

DEVICE DOWN – (adj.) Relates to the state of a peripheral device. When the device is down, it is no longer operable. Also (noun), refers to the operator command DN, which sets the device down.

DEVICE UP – (adj.) Relates to the state of a peripheral device. When the device is up, it is operable. Also (noun), refers to the operator command UP, which sets the device UP after it has been set down.

DISC DRIVE – Consists of a mechanism to rotate the disc, and electronic circuitry to write data on and read data off the disc through disc heads.

FIXED HEAD DISC DRIVE – Consists of the mechanism to rotate one disc with a read/write head per track.

GLOBAL TRACKS – Global tracks are a subset of RTE System tracks and are accounted for in the track assignment table. Any program can read/write or release a global track.

MOVING HEAD DISC DRIVE – Consists of a mechanism to rotate one or two discs, one permanently mounted and the other removable. There is one head per recording surface that is attached to a movable arm. The head is moved to the addressed track by means of an actuator driving the arm and head.

PERIPHERAL DISC – A peripheral disc is a disc that is available to the user for read/write operations but for which the Real-Time Executive does not manage the disc nor maintain a track assignment table. A peripheral disc must have a logical unit number assignment greater than 6.

PROGRAM SWAPPING – Where program A is removed from computer memory and stored on the disc in its current state of execution, and program B is placed in the computer memory area (for execution) formerly occupied by program A. Program A is eventually returned to memory and continued.

REAL-TIME EXECUTIVE – The total operating system comprised of three program modules, EXEC, SCHED, and RTIOC, plus I/O drivers, and various tables. Abbreviated RTE.

RTE SYSTEM TRACKS – All those disc tracks assigned to the Real-Time Executive for which the RTE maintains a contiguous track assignment table. These disc tracks are located on logical unit 2 (system), and 3 (auxiliary).

SCRATCH AREA – A number of disc tracks used during system generation for storage of the relocatable binary code of RTE.

SUBCHANNEL – One of a group of peripherals connected to a single controller. For example, one moving head disc drive has two discs. The permanently mounted disc is subchannel 0 and the removable disc is subchannel 1.

SYSTEM DISC – The disc assigned to logical unit 2. When the system disc is part of a moving head disc drive, it is contained on one subchannel of that drive. The absolute binary code of the Real-Time Executive resides on the system disc.

TIME-OUT – (adj.) Relating to the state of a peripheral device. When the device has timed-out, it is no longer operable. Also (noun), the parameter itself. Amount of time the RTE will wait for the device to respond to an I/O transfer command before RTE makes the device inoperable.

SECTION I
GENERAL DESCRIPTION

INTRODUCTION

The Real-Time Software Manual is a programmer's guide to using the Hewlett-Packard 2005 Real-Time Executive (RTE) Systems. The reader should be familiar with operating procedures of the HP 2100 Series Computer and other system related hardware, and in addition, to software programming languages as presented in the FORTRAN (02116-9015), FORTRAN IV (5951-1321), ALGOL (02116-9072), and Assembler (02116-9014) Programmer's Reference Manuals.

SOFTWARE

Table 1-1 is a list of the software that makes up the Real-Time Executive System. This list is comprised of the RTE software only and does not include System Input/Output (SIO) Drivers for the various peripheral equipment, or other hardware related drivers.

HARDWARE

The HP 2005 Real Time Executive System is divided into three models as follows:

a) HP 2005A for the HP 2116 Computer and Fixed Head Disc Drive

b) HP 2005B for the HP 2100 Computer and Fixed Head Disc Drive

c) HP 2005C for the HP 2100 Computer and Moving Head Disc Drive.

The hardware furnished under each of the above HP 2005 Model numbers is listed in Table 1-2. If the RTE System has been purchased to operate an existing computer system, use Table 1-2 as a guide to the minimum hardware requirements. It should be noted that the equipment configuration shown in Table 1-2 is not restricting (i.e., a moving head disc drive can be used with an HP 2116 Computer).

Table 1-1. Real-Time Software

Binary Tape Number	Description
29014-60001	RTE Generator (moving head)
29015-60001	RTE Generator (fixed head)
29016-60001	RTE Executive
29016-60002	RTE Scheduler
29016-60003	RTE I/O Control
20874-60001	RTE Assembler – Main Control
20874-60002	RTE Assembler – Segment D
20874-60003	RTE Assembler – Segment 1
20874-60004	RTE Assembler – Segment 2
20874-60005	RTE Assembler – Segment 3
20874-60006	RTE Assembler – Segment 4
20874-60007	RTE Assembler – Segment 5
20875-60001	RTE FORTRAN II – Main Control
20875-60002	RTE FORTRAN II – Pass 1
20875-60003	RTE FORTRAN II – Pass 2
20875-60004	RTE FORTRAN II – Pass 3
20875-60005	RTE FORTRAN II – Pass 4
20792-60001	RTE Relocating Loader
20802-60001	System Dump
20805-60001	RTE Editor
24016-60001	Prepare Tape System
24129-60001	RTE/DOS Algol – Main Control
24129-60002	RTE/DOS Algol – Segment 1
24151-60001	RTE/DOS Relocatable Library (EAU)
24170-60001, 60002, 60003	RTE/DOS FORTRAN IV Compiler (three tapes)
24177-60001, 60002	RTE/DOS FORTRAN IV Compiler (fast compiler on two tapes – available with 24K or 32K system as alternative to 24170-60001, 60002 and 60003)
24152-60001	RTE/DOS FORTRAN IV Library
24153-60001	RTE/DOS FORTRAN Formatter

The hardware furnished under each of the above HP 2005 Model numbers is listed in Table 1-2. If the RTE System has been purchased to operate an existing computer system, use Table 1-2 as a guide to the minimum hardware requirements. It should be noted that the equipment configuration shown in Table 1-2 is not restricting (i.e., a moving head disc drive can be used with an HP 2116 Computer).

SYSTEM DESCRIPTION

MULTIPROGRAMMING

The Real-Time Executive (RTE) is a multiprogramming system that allows several programs to operate concurrently, each program executing during the unused central processor time of the others. All input/output and interrupt processing is controlled by RTE, except for special privileged interrupts which circumvent RTE for quicker response. When a program requests a non-buffered I/O transfer, RTE places the program in an I/O suspend state, initiates the I/O operation, and starts executing the next highest priority scheduled program. When the I/O transfer is complete, RTE reschedules the suspended program for execution. Operating programs can be written in Real-Time Assembler, ALGOL, or FORTRAN Languages. Programs are scheduled by time intervals, an external device, an

operator request, or by another program. RTE has a scheduling module which decides when to execute the competing programs.

The RTE system has up to four user defined program areas for execution of his programs.

- Real-time core resident
- Real-time disc resident
- Background core resident
- Background disc resident

Table 1-2. Minimum RTE Hardware

HP 2005A RTE SYSTEM
HP 2116 Computer with 16, 24, or 32K memory HP 12578 Direct Memory Access HP 12579 Extended Arithmetic Unit HP 12581 Memory Protect HP 12539 Time Base Generator HP 2160 Auxiliary Power Supply HP 2766 Disc Memory (fixed head) HP 2772 Disc Power Supply HP 12610 Interface Kit HP 2752 Teleprinter HP 12531 Interface Kit HP 2748 Punched Tape Reader HP 12597-002 Interface Kit
HP 2005B RTE SYSTEM
HP 2100 Computer complete with operator's panel, extended arithmetic (EAU), parity, power fail, memory protect, and 14 I/O locations. HP 12885 8K Memory Module (two or more) HP 12895 Direct Memory Access HP 12539 Time Base Generator HP 2766 Disc Memory (fixed head) HP 2772 Disc Power Supply HP 12610 Interface Kit HP 2752 Teleprinter HP 12531 Interface Kit HP 2748 Punched Tape Reader HP 12597-002 Interface Kit
HP 2005C RTE SYSTEM
HP 2100 Computer complete with operator's panel, extended arithmetic (EAU), parity, power fail, memory protect, and 14 I/O locations. HP 12885 8K Memory Module (two or more) HP 12895 Direct Memory Access HP 12539 Time Base Generator HP 7900 Disc Memory (moving head) HP 13215 Power Supply HP 13210 Interface Kit HP 2752 Teleprinter HP 12531 Interface Kit HP 2748 Punched Tape Reader HP 12597-002 Interface Kit

Figure 1-1 shows the core layout for the core-resident and disc-resident programs.

REAL-TIME CORE RESIDENT

Real-time core resident programs are always resident in core and are intended for high priority tasks requiring quick response to real-time conditions. Programs of this type can control a logically subordinate set of disc resident programs.

REAL-TIME DISC RESIDENT

This type resides in absolute format on the disc and must be transferred into a reserved part of core before executing. Thus, disc resident programs provide slower response than core resident. Since all real-time disc resident programs have the same point of origin in core, only one program may be in core at a time. With the optional swapping feature, programs can be suspended and swapped out of core (except I/O suspended) if a higher priority program needs the area.

BACKGROUND CORE RESIDENT

These programs are identical to the real-time core resident type, except that the program area is located at the start of the background region and shares a common area with the background disc resident programs.

BACKGROUND DISC RESIDENT

Unlike the real-time disc resident type, background disc resident programs are never swapped, they occupy core until completion or abortion. The memory requirements

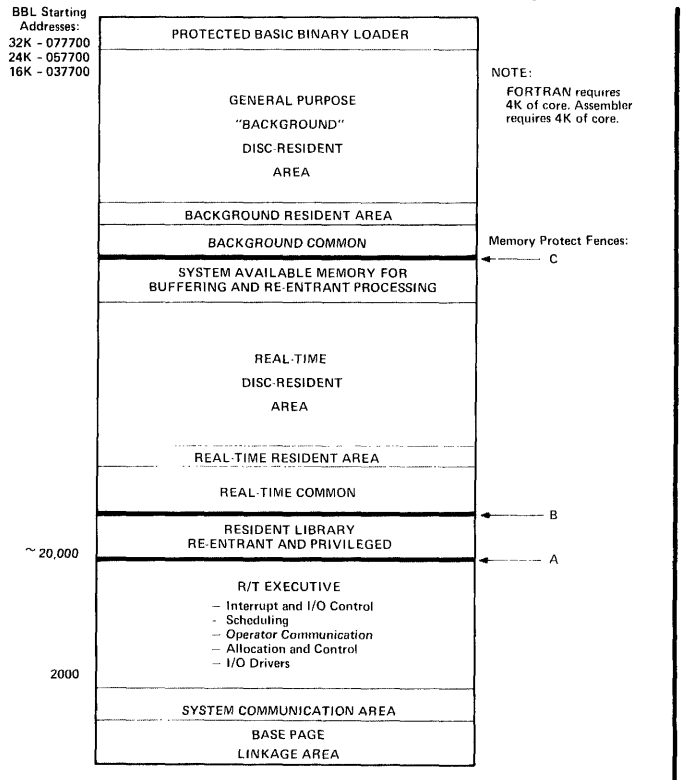


Figure 1-1. Core Allocations in the HP RTE System

stated apply to the program plus a resonable area for symbol tables. Background disc resident software usually includes the following:

REAL-TIME ASSEMBLER – Accepts source programs in HP Assembler Language and outputs a relocatable binary program to a load-and-go disc track and/or punches the program on paper tape. Usually only one pass of the source tape is required.

REAL-TIME FORTRAN – Compiles FORTRAN programs and provides additional statements for real-time control. Same output options as Assembler.

REAL-TIME FORTRAN IV – Compiles source programs written in FORTRAN IV with extended precision arithmetic and bit manipulation via logical expressions. The three tape version of FORTRAN IV compiler requires 6K of background core, and the two tape version 12K.

REAL-TIME ALGOL – Compiles source programs written in HP ALGOL. An 8K background area is required to use this compiler with the real-time system.

REAL-TIME SYMBOLIC EDITOR – Edits, updates, and lists source language programs from a paper tape or disc file input.

REAL-TIME RELOCATING LOADER – Provides on-line loading of user generated programs. Programs can be debugged and tested in background then brought up into real-time disc resident area if desired.

REAL-TIME DEBUG – When user program is loaded with relocating loader, DEBUG can be appended to each main program and segments to provide checkout. When the program is run, DEBUG takes control of program execution and requests instructions from the keyboard.

The background disc resident area can be used to create new programs to control future processes while the real-time area is handling external events. The programs can be compiled, tested, and placed into operation without any paper tape output, and little operator intervention. In addition, new programs can be added to the set of permanent user programs, and old ones deleted.

SUBROUTINES

Each user program (main or segment) consists of a primary routine, containing the transfer point for entry into the program from RTE, and optionally a series of subroutines. In Assembler Language, the transfer point is the location of the label appearing in the END statement. In FORTRAN, the transfer point is the first executable instruction in a routine containing a PROGRAM statement. The primary routine is linked with its subroutines (which are defined by external references within the primary routine) when it is loaded.

The Relocatable Library consists of a number of subroutines that may be linked to user programs. (See Section IV, Part 6.) Each subroutine is either re-entrant, privileged, or utility. These terms are defined as follows:

- Re-entrant – Can be interrupted
- Privileged – Cannot be interrupted
- Utility – Used by one program only

The classification of a specific routine is based on its functions, word length, and execution times. A re-entrant or privileged subroutine may be used by more than one program at a time; if they are referenced by core-resident programs, the Real-Time Generator (RTGEN) places them in the resident library. (See Figure 1-1.) If called by disc resident programs but not in the resident library, the re-entrant and privileged subroutines are appended to the absolute version of the calling program.

Subroutines which cannot be shared because of internal design or I/O considerations are utility subroutines, and a copy of the utility subroutine is appended to each primary routine, whether core or disc resident, that calls it. RTGEN stores all library programs that are not included in the resident library on the disc in relocatable format (as utility routines to be used by the Relocating Loader).

PROGRAM SCHEDULING (SCHED)

Scheduling of all programs is done by the scheduling module SCHED, and is based on priority. Programs may be scheduled for execution by an operator request, a program request, a device interrupt, or the completion of a time interval. The RTE System can be generated such that one program is automatically scheduled each time the system is loaded from the disc. Whenever programs conflict because of simultaneous demands for execution, SCHED decides in favor of the highest priority program. Priorities are assigned by the user during RTGEN or on-line loading, and may be changed by an operator request.

The RTE System handles priorities on a completely generalized basis. The highest priority program scheduled for execution executes first. Then, if that program suspends execution (possibly for an input wait), the next highest priority scheduled program executes. This process continues down the scheduled list until a higher priority program is rescheduled.

Programs that were removed from the executing state to wait for an event to occur before re-scheduling are in the suspended state. Programs which are not currently in either the scheduled, executing, or suspended state are in the dormant state. Programs may thus be in one of four states:

- Executing
- Scheduled
- Suspended
- Dormant

The status field in the ID segment (see Appendix A) records the state of the program.

Programs may be suspended for several reasons:

- Waiting for the completion of an I/O operation
- Waiting for the availability of needed memory space
- Waiting for the completion of a disc allocation

- Waiting for the completion of a program scheduled by the suspended program
- The operator has requested that a program be suspended
- The program has requested that it be suspended

PRIORITY LEVEL

Program priority determines the order of a program in the scheduled, suspended, and dormant states. The priority field of the ID segment (see Appendix A) records the priority of the program. Priorities range from 0 (the highest, reserved for system programs) to 99 (the lowest). The priority of any dormant program can be changed by an operator request, and more than one program can be at the same priority.

For each program state except executing, RTE maintains an ordered list of the programs in that state, connecting the ID segments according to the priority of the programs. There are three types of lists:

- Scheduled
- Suspended
- Dormant

The base page communication area (see Appendix A) contains the pointers to the ID segment of the first, or highest priority, program in each list. Then, the linkage field of each ID segment contains the location of the next ID segment in the list. There are one scheduled list, one dormant list, and four types of suspension lists:

- I/O suspension lists (one for each device)
- Memory availability list
- Disc allocation list
- Operator suspension list

PROGRAM INITIATION AND SWAPPING

A program is initiated immediately once it is transferred to core memory. Real-time resident programs are normally given highest priority and are always in core. If the program is on disc and is to be executed in the real-time disc-resident area, and an uncompleted program already occupies that space, the uncompleted program is transferred out to the disc and saved in its uncompleted and modified state. Then the new program is transferred into core. If the program is on disc and to be transferred into the background disc resident area, that area has to be unoccupied (i.e., previous program run to completion), before the transfer is made. During the transfer a check is made to see if work can be done in one of the other areas of core. I/O operations can continue simultaneously.

INPUT/OUTPUT CONTROL (RTIOC)

RTIOC is responsible for processing all system interrupts and input/output operations. Section V describes the I/O structure of the RTE System in detail, especially I/O drivers, and privileged interrupt processing.

INTERRUPT PROCESSING

All interrupts, except privileged interrupts, cause a transfer to the central interrupt control (CIC) in RTIOC, which is responsible for saving and restoring the various registers, analyzing the source of the interrupt, and calling the appropriate processing routine.

An interrupt table, ordered by hardware interrupt priority, contains the correct processor routine for each interrupt. Processors that respond to standard system interrupts (real-time clock routine, memory protect, standard I/O drivers) are called directly by CIC. Processors that respond to user-controlled devices or interrupt sources are scheduled just like other programs.

When an interrupt occurs, the instruction in the word corresponding to the I/O channel number is executed. For all active interrupt locations, except privileged interrupts, this instruction is a jump subroutine (indirect) to CIC.

INPUT/OUTPUT PROCESSING

RTIOC allocates DMA channels for I/O devices, provides for referencing I/O devices by logical unit number rather than directly by EQT entry number or I/O channel, stacks program I/O requests for a particular device by priority of the calling program, and provides automatic output buffering, when specified, for low- to medium-speed devices.

I/O drivers are under control of RTIOC for initiation and completion of program-requested I/O operations; they provide simultaneous multi-device control.

Program requests for I/O are made by EXEC calls which specify the type of transfer and device desired. RTIOC handles the request. It suspends the requesting program until the operation is complete unless the request is for output to a buffered device. All input/output operations occur concurrently with program execution, however, if the transfer is non-buffered, the requesting program is suspended and the next lower priority scheduled program is allocated execution time during the suspension.

EXEC COMMUNICATION (EXEC)

When an executing program makes an EXEC call, it attempts to execute a jump subroutine to EXEC in the protected area of core. This causes a memory protect violation interrupt, which CIC recognizes, and transfers to EXEC. EXEC examines the parameters associated with the jump subroutine in the calling program. If the parameters are legal, EXEC either handles the request itself or goes to the appropriate part of RTE to process it.

Using EXEC calls, which are the line of communication between an executing program and RTE, a program is able to:

- Perform input and output operations.
- Allocate and release disc space.
- Terminate or suspend itself.
- Load its segments (if background disc type).

- Schedule other programs.
- Obtain the time of day.
- Set execution time cycles.

OPERATOR REQUESTS

The operator retains ultimate control of the RTE System with requests entered through the teleprinter keyboard. (See Section II.) Operator requests, which are handled by SCHED, can interrupt RTE to:

- Turn programs on and off.
- Suspend and restart programs.
- Examine the status of any program or I/O device.
- Schedule programs to execute at specified times.
- Change the priority of dormant programs.
- Set up load-and-go operations and source files.
- Declare I/O devices up or down.
- Dynamically alter the logical I/O structure and buffering designations.
- Eliminate disc-resident programs from the system.
- Examine and dynamically alter an I/O device's time-out parameter.
- Release tracks assigned to dormant programs.
- Initialize the real-time clock and print the time.

SYSTEM CONFIGURATION

User real-time programs and background programs, system programs, library routines, and Real-Time Executive Modules are incorporated into a configured RTE System. The RTE software is modular and of a general nature, so the user can configure his particular programs and I/O device drivers into a real-time system tailored to his exact needs.

Using the Real-Time Generator (RTGEN), the relocatable software modules and user programs are converted into a configured real-time system in absolute binary format which is stored on a disc. In operation the configured system is loaded into the computer from the Real-Time Disc Resident area of the disc. The remaining disc storage is dynamically allocated by the configured system to user programs or is utilized by the scheduler for swapping operations.

An RTE System can be configured using both fixed head and moving head disc units. The discs are placed into three classifications according to their function.

SYSTEM/AUXILIARY DISCS

The RTE System disc tracks are those for which RTE controls and maintains a contiguous track usage table. These are logical units 2 (system), and 3 (auxiliary). The system disc tracks are used for swapping, and by the editor, assembler, and compilers for source, load-and-go, and scratch area. They may also be used by user programs for storage. The only differences between a system disc and an auxiliary disc are that the absolute code of RTE is stored on the system disc, and that the auxiliary disc is optional.

PERIPHERAL DISCS

Peripheral discs are not managed by the RTE System. Track allocation and usage are totally up to the user programs. Peripheral discs are distinguished from system discs by having logical unit numbers greater than 6.

An RTE System can be configured using both fixed head and moving head disc units. The possible combinations are as follows:

- RTE System stored on a fixed head disc.
  a. Fixed head auxiliary disc.
  b. Fixed head peripheral discs.
  c. Moving head peripheral discs.
  d. Moving head auxiliary disc.
- RTE System stored on a moving head disc.
  a. Moving head auxiliary disc.
  b. Moving head peripheral discs.
  c. Fixed head peripheral discs.

The fixed head disc drive is a single unit that contains one disc that is interfaced through a single controller (one controller per disc drive). The disc is accessed through a logical unit reference number. The system disc containing the absolute code of RTE is logical unit number 2, and the auxiliary disc, if specified, is logical unit number 3. Peripheral discs have logical unit numbers other than 2 or 3, and are not managed by the RTE System.

The HP 7900 Moving Head Disc Drive is a single unit that contains two discs; one permanently mounted, and the other housed in a removable cartridge. The drive is interfaced through a single controller, however, it is possible to daisy-chain up to four drives to the same controller. This combination provides up to eight discs. Each disc is a subchannel, and is accessed through a logical unit reference number that is referenced back to the equipment table (EQT) entry number of the controller. Therefore, one controller, containing eight subchannels linked to eight logical unit numbers, can control up to eight discs. When the RTE moving head disc system is generated using RTGEN, the user designates areas, or tracks, of each disc that are available to RTE. The system is limited to these tracks, and as a result, it is possible for the user to create many different configurations of RTE Systems, all coresiding on the same moving head disc channel.

Additional information on system configuration is found in Section VI and Appendix B of this manual.

RTE SYSTEM SUMMARY

The Hewlett-Packard Real-Time Executive Software System is a multiprogramming, foreground-background system with priority scheduling, interrupt handling, and program load-and-go capabilities.

With multiprogramming, a number of data acquisition systems or test stands can be operated simultaneously on a 24-hour a day basis. Data reduction and report preparation functions can be scheduled to execute in the background area during times when foreground real-time activities permit. The same computer can also be used by the programming group for ongoing development work with RTE's background compilers for FORTRAN, FORTRAN IV, and ALGOL, and with the HP Assembler, Editor, and other auxiliaries. Programs can be added to the system on-line, and on a load-and-go basis (no intervening paper tapes). For system protection, new programs can be debugged in background while memory protect maintains the integrity of the real-time foreground area.

Scheduling of all programs is based on priority. External events can interrupt to schedule programs for execution, or a program can be scheduled by an operator request, a program request, or on a real-time clock basis. Priorities are assigned by the user during RTGEN or on-line loading, and may be changed by an operator request.

The Executive controls I/O processing through a central routine that directs requests and interrupts to the appropriate device driver subroutine. For efficiency, programs awaiting I/O are suspended to let other programs use the computer. Outputs to slow devices can be buffered. For processes that cannot tolerate ordinary system overhead, a privileged interrupt option lets a device contact its driver directly without going through the Executive.

The operator retains ultimate control of the RTE System with requests entered through the teleprinter keyboard. The operator can turn programs on, make status checks, or perform other operations.

Configuration is efficient. The generation program RTGEN, in a dialog with the operator, configures the software for a particular hardware system on that system itself.

Mass storage of programs and data on a fixed-head disc for fast response or on a moving-head disc for large storage is provided. Programs can be swapped within the real-time area. Very large programs can be segmented for execution in the background area.

SECTION II
OPERATOR REQUESTS

The operator controls an executing Real-Time Executive System by operator requests entered through the teleprinter console. These operator requests can interrupt RTE to:

a. Turn programs on and off.
b. Suspend and restart programs.
c. Examine the status of any program or I/O device.
d. Schedule programs to execute at specified times.
e. Change the priority of dormant programs.
f. Set up load-and-go operations and source files.
g. Declare I/O devices up or down.
h. Dynamically alter the I/O structure and buffering designations.
i. Eliminate disc-resident programs from the system.
j. Examine and dynamically alter an I/O device's time-out parameter.
k. Release tracks assigned to dormant programs.
l. Initialize the real-time clock and print the time.

The operator gains the attention of RTE by pressing any key on the console. When RTE responds with an asterisk (*), the operator types any operator request, consisting of a two-character request word (e.g., ON, UP, etc.) and the appropriate parameters separated by commas.

NOTE

Two commas in a row mean a parameter is zero.

When the data is entered, format can include items inside and outside brackets. The items outside the brackets are required symbols, and the items inside the brackets are optional. Note that when RTE is restarted from the disc, any parameters entered are restored to their original value set during RTGEN.

If an error is made in entering the parameters, CONTROL and A struck simultaneously will delete the last character entered, or, RUBOUT followed by carriage return/line feed will delete the entire request. Each request must be completed with an end-of-record terminator (e.g., carriage return/line feed for the teleprinter).

The operator requests are presented in alphabetical order and described in detail in this section. For a handy reference of all operator requests their parameters, tear out Appendix H at the back of this manual and tape to the teletype console.

DN

Purpose

To declare an I/O device down (i.e., unavailable for use by the RTE system).

Format

DN, *n*

Where

n is the EQT entry number of the I/O device to be set down.

COMMENTS

The device set down is unavailable until set up by the UP operator request. The operator might set a device down because of equipment problems, tape change, etc.

EQ, *n*

Purpose

To print the description and status of an I/O device, as recorded in EQT entry.

Format

EQ, *n*

Where

n is the EQT entry number of the I/O device.

COMMENTS

The information is printed as:

SC DVR*nn D B* U*nn LS*

Where

SC	is the select code I/O channel).
DVR*nn*	is the driver routine,
D	is D if DMA required, 0 if not,
B	is B if automatic output buffering used, 0 if not,
U*nn*	is the last subchannel addressed
LS	is the logical status: 0 - available 1 - unavailable (down) 2 - unavailable (busy) 3 - waiting for DMA assignment

EQ, *n, p*

Purpose

To change the automatic output buffering designation for a particular I/O device.

Format

EQ, *n, p*

Where

n is the EQT entry number of the I/O device, and
p is 0 to delete buffering, or
p is 1 to specify buffering.

COMMENTS

When the system is restarted from the disc, buffering designations made by EQ, *n, p* are reset to the values originally made by RTGEN.

NOTE

The standard line printers (controlled by I/O Driver DVR12) may not be buffered. (see Appendix F.)

GO

Purpose

To reschedule a program that has been suspended by an SS operator request or a Suspend EXEC Call.

Format

GO, *name* [, *p1*, ... , *p5*]

Where

name is the name of an operator suspended program to be scheduled for execution.
p1 through *p5* is a list of parameters to be passed to name. Note, these parameters should be used only when the program has suspended itself.

COMMENTS

If the program has not been suspended previously by the operator or has not suspended itself, the request is illegal.

NOTE

Do not enter parameters if the program was suspended by an SS operator request lest the saved contents of the B-Register be destroyed.

When a program resumes execution after suspending itself, the address of the parameters passed by GO is in the B-Register. In FORTRAN, an immediate call to the library subroutine RMPAR retrieves the parameters. (See Section III, Suspend EXEC Call.) If no parameters are entered (program suspended by SS operator request), the B-Register is restored to its value before suspension

IT

Purpose

To set time values for a program, so that the program executes automatically at selected times when turned on with the ON operator request.

Format

IT,*name,r,mpt*[,*h,min*[,*s*[,*ms*]]]

Where

name is the name of the program,

r is the resolution code:
1 - tens of milliseconds
2 - seconds
3 - minutes
4 - hours
mpt is a number from 0 to 999 which is used with *r* to give the actual time interval for scheduling (see Comments),
h hours
min minutes
s seconds
ms tens of ms.
} sets an initial start time.

COMMENTS

The resolution code (*r*) is the units in time to be multiplied by the multiple execution interval value (*mpt*) to get the total time interval. Thus, if *r*=2 and *mpt*=1ØØ, *name* would be scheduled every 100 seconds. If *h, min, s,* and *ms* are present, the first execution occurs at the initial start time which these parameters specify. (Program must be initialized with ON operator request.) If *h, min, s,* and *ms* are not present, the program can be called by another program or started with the ON, NOW operator request.

When the system is restarted from the disc, time values set by IT are lost, and the original time values set during RTGEN or LOADR are reinstated.

The IT operator request is similar to the Execution Time EXEC Call. (See Section III.)

LG

Purpose

To allocate or release a group of disc tracks for load-and-go operations.

Format

LG, *n*

Where

n = 0 (zero) release the allocated load-and-go area.
n > 0 allocate *n* contiguous tracks for a load-and-go area; set flags for load-and-go operation.

COMMENTS

The user must allocate enough tracks for storing binary object code before each load-and-go compilation or assembly. If not, the compiler or assembler aborts and a diagnostic is printed on the system console.

IOØ6 - Load-and-go area not defined.
IOØ9 - Overflow of load-and-go area.

An LG request should not be used while a compiler or the Assembler is using the load-and-go tracks. If done, this results in the message

LGO IN USE

being printed on the system console, and no change in the current number of load-and-go tracks.

LS

Purpose

To designate the disc logical unit number and starting track number of an existing source file before operating on it with EDIT, FTN, FTN4, ALGOL, or ASMB.

Format

LS, *p1, p2*

Where

p1 is the logical unit number of the disc containing the source file.
p1 = 2 or 3 system or auxiliary disc units.
p1 = 0 eliminate the current source file designation.
P2 is the starting track number of the source file (in decimal).

COMMENTS

LS replaces any previous file declarations with current file. Only one file may be declared at a time.

For details on creating, updating, compiling, or assembling source files, see Section IV, Real-Time Programming.

LU

Purpose

To print or change a logical unit number assignment.

Format

LU, *n*[,*m*[,*p*]]

Where

n is a logical unit number from 1 to 63_{10}.
m, if present, is an EQT entry number to assign to *n*.
m, if zero (0), releases logical unit number.
p, if present, is a subchannel number (0-7) to assign to *n*.
If *m* and *p* are absent the assignment of logical unit *n* is printed.

COMMENTS

The information is printed as

LU #07 = #05, U02

which means logical unit number 7 is assigned to EQT entry number 5, subchannel number 2.

LU1 must be a keyboard entry device (e.g., teleprinter). If LU1 is changed from one keyboard device to another, the new device will print a double asterisk (**). LU2 (system disc) and LU3 (aux disc) cannot be changed to a new EQT entry number (i.e., if *n* = 2 or 3 no other parameter can be entered.

When an irrecoverable problem occurs on an I/O device, the operator can bypass the downed device for future requests by reassigning the logical unit number to an operable device on another channel. Any programs referencing the downed device are suspended until the device is declared UP.

When the system is restarted from the disc, any assignments made by LU are reset to those originally set by RTGEN.

Section V, Real-Time Input/Output, explains logical unit numbers, equipment table entry numbers, and subchannel numbers in detail.

OF

Purpose

To terminate a program, or to remove a background program which was loaded on-line but not permanently incorporated into the protected RTE system.

Format

OF, *name, p*

Where

name is the name of a program,

p = 0 terminates and removes from the time list any executing, scheduled, or operator suspended program; terminates programs which are I/O, memory, or disc suspended the next time they are scheduled. In neither case are disc tracks released.

p>0,≠8 terminates immediately the program named, removes it from time list, and releases all disc tracks. If suspended for I/O, the device and channel are cleared by a CLC.

p = 8 same as for *p*>0, plus, if the program is not I/O suspended, the program is completely removed from the core RTE system. If the program is I/O suspended, the OF request is treated as if *p* were greater than 0, but not equal to 8. OF, *name,* 8 must be entered again to permanently remove *name* from the RTE System. Should be used only on programs loaded on-line, but not permanently incorporated into the system. The ID segment is blanked, and the tracks containing the program (if not protected) are released. The blank ID segment is then available for loading another program with LOADR.

COMMENTS

For programs with segments, OF must be used on the segments as well as the program. OF should not be used to remove permanent programs because their ID segments on the disc are not altered by this request.

ON

Purpose

To schedule a program for execution. Up to five parammeters may be passed to the program.

Format

ON, *name* [,NOW] [*,p1, p2,. . .,p5*]

Where

name is the name of a program,
NOW schedules a program immediately that is normally scheduled by the clock time (see IT), and
p1 through *p5* are parameters passed to the program when it is scheduled (must be positive integers less than 32767).

COMMENTS

The ON operator request schedules the various background programming aids (i.e., *name* equals EDIT, FTN, FTN4, ALGOL, ASMB, or LOADR). ON passes parameters to these programs (See Section IV, Real-Time Programming.)

If the resolution code in the ID segment of the program is not zero, RTE places the program in the time list for execution at specified times (unless NOW appears, in which case, the program executes first, then goes into the time list). The resolution code may be non-zero as a result of:

a. Generation
   1. With a resolution code in the *name* record
   2. Entry of a resolution code during parameter input phase.
b. The IT command.
c. Scheduling the program with the clock by some program in the system.

PR

Purpose

To change the priority of a program.

Format

PR, *name, n*

Where

name is the name of the program,
n is the new priority.

COMMENTS

One is the highest priority, and 99 is the lowest. When the system is restarted from the disc, the priority of *name* resets to the value set by RTGEN or LOADR.

Only dormant programs can have their priority changed.

RT

Purpose

To release all disc tracks assigned to a program.

Format

RT, *name*

Where

name is the name of the program that is to have its tracks released.

COMMENTS

If the program is not dormant, the request is illegal.

If the program is dormant, all tracks assigned to that program are released.

If any tracks are released as a result of this request, all programes in disc track allocation suspension are rescheduled.

SS

Purpose

To suspend a program from execution.

Format

SS, *name*

Where

name is the name of the program to be suspended.

COMMENTS

The SS request places the program in the operator suspended list immediately if the program is executing, scheduled, or already there. If the program is dormant the request is illegal. If the program is suspended for I/O, memory or disc, RTE waits until the current suspend is over, then suspends the program with SS.

The SS operator request is similar to the Program Suspend EXEC Call. (See Section III.)

ST

Purpose

To request the status (priority, current list, time values) of a program

Format

ST, *name*

Where

name is the name of the program whose status is to be printed.

COMMENTS

The status is printed on one line in a fixed format:

PR S R MPT H MIN S MS T

Where

PR is the priority, a value from 1 to 99_{10},
S is the current list in which the program is located:
- 0 - Dormant
- 1 - Scheduled
- 2 - I/O suspend
- 3 - Not used
- 4 - Unavailable memory suspend
- 5 - Disc allocation suspend
- 6 - Operator suspend or programmed suspend (EXEC 7 Call)

R, MPT, H, MIN, S and MS are all zero (0) unless the program is scheduled by the clock (see IT, this section, for the meaning of these items).

The letter "T" appears when the program is currently in the time list (as the result of an ON operator request).

TI

Purpose

To print the current time of day and day of the year, as recorded in the real-time clock.

Format

TI

COMMENTS

The computer prints out the day and time:

DAY H MIN S

Where

DAY is the three-digit day of the year and
H, MIN, S is the time on a 24-hour clock. (See Table 2-1 for day of year conversion.)

The TI operator request is similar to the Time Request EXEC Call. (See Section III.)

TM

Purpose

To set the real-time clock

Format

TM, *day, h, min, s*

Where

day is a three-digit day of the year (see Table 2-1),
h, min, and *s* is the current time on a 24-hour clock.

COMMENTS

The operator usually gives TM in response to the message printed when the RTE system is initiated from the disc:

SET TIME

The response sets the time when the line-feed key is pressed. Enter a time value ahead of real-time and press carriage-return. When real-time equals the entered value, press line-feed. The system is now synchronized with the time of day.

TO

Purpose

To print or change the time-out parameter of an I/O device

Format

TO,*n*[,*m*]

Where

n is the EQT entry number of the I/O device, and
m is the number of 10 ms intervals to be used as the time-out value. (*m* cannot be less than 500 (5 sec) for the system input device.)

If *m* is absent the time-out value of EQT *n* is printed.

COMMENTS

The information is printed as

TO #03 = 100

which means EQT entry number 3 has a time-out value of one second.

When the system is restarted from the disc, time-out values set by TO are reset to the values originally set during RTGEN.

The time-out value is calculated using *m* time base generator interrupts (the time base generator interrupts once every 10 ms). For example, *m* = 100 sets a time-out value of one second: 100 · 10 ms = 1 second.

If a device has been initiated, and it does not interrupt within the interval set by the time-out parameter, the following events take place:

a. The calling program is rescheduled, and a zero transmission log is returned to it.

b. The device is set to the down status, and bit 11 in the fourth word of the device's EQT entry is set to one. An error message is printed; e.g.,

I/O ERROR TO EQT #3

c. The system issues a CLC to the device's I/O select code(s) through the EQT number located in the interrupt table.

The device set down is unavailable until set up by the UP operator request.

Table 2-1. Day of Year

JANUARY						
	1/2 (2)	1/3 (3)	1/4 (4)	1/5 (5)	1/6 (6)	1/7 (7)
1/8 (8)	1/9 (9)	1/10 (10)	1/11 (11)	1/12 (12)	1/13 (13)	1/14 (14)
1/15 (15)	1/16 (16)	1/17 (17)	1/18 (18)	1/19 (19)	1/20 (20)	1/21 (21)
1/22 (22)	1/23 (23)	1/24 (24)	1/25 (25)	1/26 (26)	1/27 (27)	1/28 (28)
1/29 (29)	1/30 (30)	1/31 (31)				

FEBRUARY						
2/1 (32)	2/2 (33)	2/3 (34)	2/4 (35)	2/5 (36)	2/6 (37)	2/7 (38)
2/8 (39)	2/9 (40)	2/10 (41)	2/11 (42)	2/12 (43)	2/13 (44)	2/14 (45)
2/15 (46)	2/16 (47)	2/17 (48)	2/18 (49)	2/19 (50)	2/20 (51)	2/21 (52)
2/22 (53)	2/23 (54)	2/24 (55)	2/25 (56)	2/26 (57)	2/27 (58)	2/28 (59)
2/29 (60)	LEAP YEAR ONLY					

MARCH						
3/1 (60)	3/2 (61)	3/3 (62)	3/4 (63)	3/5 (64)	3/6 (65)	3/7 (66)
3/8 (67)	3/9 (68)	3/10 (69)	3/11 (70)	3/12 (71)	3/13 (72)	3/14 (73)
3/15 (74)	3/16 (75)	3/17 (76)	3/18 (77)	3/19 (78)	3/20 (79)	3/21 (80)
3/22 (81)	3/23 (82)	3/24 (83)	3/25 (84)	3/26 (85)	3/27 (86)	3/28 (87)
3/29 (88)	3/30 (89)	3/31 (90)				

APRIL						
4/1 (91)	4/2 (92)	4/3 (93)	4/4 (94)	4/5 (95)	4/6 (96)	4/7 (97)
4/8 (98)	4/9 (99)	4/10 (100)	4/11 (101)	4/12 (102)	4/13 (103)	4/14 (104)
4/15 (105)	4/16 (106)	4/17 (107)	4/18 (108)	4/19 (109)	4/20 (110)	4/21 (111)
4/22 (112)	4/23 (113)	4/24 (114)	4/25 (115)	4/26 (116)	4/27 (117)	4/28 (118)
4/29 (119)	4/30 (120)					

MAY						
5/1 (121)	5/2 (122)	5/3 (123)	5/4 (124)	5/5 (125)	5/6 (126)	5/7 (127)
5/8 (128)	5/9 (129)	5/10 (130)	5/11 (131)	5/12 (132)	5/13 (133)	5/14 (134)
5/15 (135)	5/16 (136)	5/17 (137)	5/18 (138)	5/19 (139)	5/20 (140)	5/21 (141)
5/22 (142)	5/23 (143)	5/24 (144)	5/25 (145)	5/26 (146)	5/27 (147)	5/28 (148)
5/29 (149)	5/30 (150)	5/31 (151)				

JUNI						
6/1 (152)	6/2 (153)	6/3 (154)	6/4 (155)	6/5 (156)	6/6 (157)	6/7 (158)
6/8 (159)	6/9 (160)	6/10 (161)	6/11 (162)	6/12 (163)	6/13 (164)	6/14 (165)
6/15 (166)	6/16 (167)	6/17 (168)	6/18 (169)	6/19 (170)	6/20 (171)	6/21 (172)
6/22 (173)	6/23 (174)	6/24 (175)	6/25 (176)	6/26 (177)	6/27 (178)	6/28 (179)
6/29 (180)	6/30 (181)					

JULY						
7/1 (182)	7/2 (183)	7/3 (184)	7/4 (185)	7/5 (186)	7/6 (187)	7/7 (188)
7/8 (189)	7/9 (190)	7/10 (191)	7/11 (192)	7/12 (193)	7/13 (194)	7/14 (195)
7/15 (196)	7/16 (197)	7/17 (198)	7/18 (199)	7/19 (200)	7/20 (201)	7/21 (202)
7/22 (203)	7/23 (204)	7/24 (205)	7/25 (206)	7/26 (207)	7/27 (208)	7/28 (209)
7/29 (210)	7/30 (211)	7/31 (212)				

AUGUST						
8/1 (213)	8/2 (214)	8/3 (215)	8/4 (216)	8/5 (217)	8/6 (218)	8/7 (219)
8/8 (220)	8/9 (221)	8/10 (222)	8/11 (223)	8/12 (224)	8/13 (225)	8/14 (226)
8/15 (227)	8/16 (228)	8/17 (229)	8/18 (230)	8/19 (231)	8/20 (232)	8/21 (233)
8/22 (234)	8/23 (235)	8/24 (236)	8/25 (237)	8/26 (238)	8/27 (239)	8/28 (240)
8/29 (241)	8/30 (242)	8/31 (243)				

SEPTEMBER						
9/1 (244)	9/2 (245)	9/3 (246)	9/4 (247)	9/5 (248)	9/6 (249)	9/7 (250)
9/8 (251)	9/9 (252)	9/10 (253)	9/11 (254)	9/12 (255)	9/13 (256)	9/14 (257)
9/15 (258)	9/16 (259)	9/17 (260)	9/18 (261)	9/19 (262)	9/20 (263)	9/21 (264)
9/22 (265)	9/23 (266)	9/24 (267)	9/25 (268)	9/26 (269)	9/27 (270)	9/28 (271)
9/29 (272)	9/30 (273)					

OCTOBER						
10/1 (274)	10/2 (275)	10/3 (276)	10/4 (277)	10/5 (278)	10/6 (279)	10/7 (280)
10/8 (281)	10/9 (282)	10/10 (283)	10/11 (284)	10/12 (285)	10/13 (286)	10/14 (287)
10/15 (288)	10/16 (289)	10/17 (290)	10/18 (291)	10/19 (292)	10/20 (293)	10/21 (294)
10/22 (295)	10/23 (296)	10/24 (297)	10/25 (298)	10/26 (299)	10/27 (300)	10/28 (301)
10/29 (302)	10/30 (303)	10/31 (304)				

NOVEMBER						
11/1 (305)	11/2 (306)	11/3 (307)	11/4 (308)	11/5 (309)	11/6 (310)	11/7 (311)
11/8 (312)	11/9 (313)	11/10 (314)	11/11 (315)	11/12 (316)	11/13 (317)	11/14 (318)
11/15 (319)	11/16 (320)	11/17 (321)	11/18 (322)	11/19 (323)	11/20 (324)	11/21 (325)
11/22 (326)	11/23 (327)	11/24 (328)	11/25 (329)	11/26 (330)	11/27 (331)	11/28 (332)
11/29 (333)	11/30 (334)					

DECEMBER						
12/1 (335)	12/2 (336)	12/3 (337)	12/4 (338)	12/5 (339)	12/6 (340)	12/7 (341)
12/8 (342)	12/9 (343)	12/10 (344)	12/11 (345)	12/12 (346)	12/13 (347)	12/14 (348)
12/15 (349)	12/16 (350)	12/17 (351)	12/18 (352)	12/19 (353)	12/20 (354)	12/21 (355)
12/22 (356)	12/23 (357)	12/24 (358)	12/25 (359)	12/26 (360)	12/27 (361)	12/28 (362)
12/29 (363)	12/30 (364)	12/31 (365)				

Note: For leap year, add one to each number starting at 3/1 (60).

UP

Purpose

To declare an I/O device up (i.e., available for use by the RTE system).

Format

UP, *n*

Where

n is the EQT entry number of the device to be set up.

COMMENTS

When the operator or the RTE system has set an I/O device down for some reason, the operator should correct the situation before declaring the device available again with the UP operator request. If the problem is irrecoverable, the operator can use LU to switch the logical unit number assignment to another device for future requests (see LU, this section). However, previous requests made to this device are not switched to the new device. To prevent indefinite I/O suspension on a downed device time-out is used. Refer to I/O Device Time-Out in Section V.

ERROR MESSAGES

RTE rejects operator requests for various reasons. When a request is in error, RTE prints one of the messages below. The operator enters the request again, correctly.

Message	Meaning
OP CODE ERROR	Illegal operator request word.
NO SUCH PROG	The name given is not a main program in the system.
INPUT ERROR	A parameter is illegal.
ILLEGAL STATUS	Program is not in appropriate state.

SECTION III
EXEC CALLS

This section describes the basic formats of FORTRAN, FORTRAN IV, ALGOL, and Assembler Language EXEC calls with each EXEC call presented in detail. Table 3-1 is a summary of the EXEC call request codes listed in the order of appearance in this section. The error messages associated with the EXEC calls are listed at the end of this section. Refer to Appendix D at the rear of this manual for a summary of the EXEC calls and required parameters.

Table 3-1. Summary of Request Codes (ICODE)

ICODE DEC Value	Function
1	READ
2	WRITE
3	I/O Control
13	I/O Status
4	Disc Allocation
15	Global Disc Allocation
5	Disc Release
16	Global Disc Release
6	Program Completion
7	Program Suspend
8	Program Segment load
9	Program Schedule With Wait
10	Program Schedule Without Wait
11	Time Request
12	Execution Time

Using EXEC calls, which are the line of communication between an executing program and RTE, a program is able to:

a. Perform input and output operations.
b. Allocate and release disc space.
c. Terminate or suspend itself.
d. Load its segments (if background disc-resident).
e. Schedule other programs.
f. Obtain the time of day.
g. Set execution time cycles.

An EXEC call is a block of words consisting of a JSB EXEC instruction and a list of parameters defining the request. The execution of the instruction causes a memory protect violation interrupt and transfers control into RTE. RTE then determines the type of request (from the parameter list) and, if it is legally specified, initiates processing of the request.

In FORTRAN and FORTRAN IV, EXEC calls are coded as CALL statements. In ALGOL, EXEC calls must be declared as CODE Procedures and parameters must be declared either as NAME or VALUE. In Assembler Language, EXEC calls are coded as JSB EXEC. followed by a series of parameter definitions. For any particular call, the object code generated for the FORTRAN CALL Statement is equivalent to the corresponding Assembler Language object code.

ASSEMBLER LANGUAGE FORMAT

The following is a general model of an EXEC call in Assembler Language:

```
        EXT   EXEC      Used to link program to RTE
        :
        JSB   EXEC      Transfer control to RTE
        DEF   * + n + 1 Defines point of return from
                        RTE, n is number of parameters;
                        may not be an indirect address
        DEF   p1        Define addresses of parameters
                        which may occur anywhere in
                        program; may be multi-level
                        indirect
        DEF   pn
        return point    Continue execution of program
          .
          :
          .
          .
p1      ---   }
 .            }
 .            }       Actual parameter values
 .            }
pn      ---   }
```

FORTRAN/FORTRAN IV FORMAT

In FORTRAN and FORTRAN IV the EXEC call consists of a CALL statement and a series of assignment statements defining the variable parameters of the call:

CALL EXEC (*p1, p2 . . , pn*)

Where

p1 through *pn* are either integer values or integer variables defined elsewhere in the program.

Example

```
CALL EXEC (7)
      or
ICODE = 7
CALL EXEC (ICODE)
```

Equivalent calling sequences

Some EXEC Call functions are handled automatically by the FORTRAN compilers or special subroutines. Refer to "FORTRAN," Section IV, Real-Time Programming, and the specific EXEC calls.

ALGOL FORMAT

In ALGOL certain conventions must be followed in making EXEC references (calls). The END statement in an ALGOL main procedure automatically declares the EXEC as external with a single integer parameter calling sequence. Therefore, without any further declaration of types of EXEC references through CODE procedures, any attempt to declare the EXEC with other than one integer parameter results in an error.

To avoid errors of this nature, CODE procedures with names other than EXEC must be declared with the proper number and type of parameters. A CODE procedure must be declared for each EXEC reference with a different number or type of parameters.

For example, the following is a main procedure referencing the EXEC with one parameter (EXECA), and then with four parameters (EXECB). The two respective CODE procedures are listed after the main procedure.

For example,

(In the main program):

```
:
PROCEDURE EXECA (A) , INTEGER A; CODE;
PROCEDURE EXECB (A,B,C,D) , INTEGER A,B,
C,D; CODE;
:
EXECA (I) ,
:
EXECB (J,K,L,M) ,
:
END$
```

(External)

```
HPAL,P,B,L, "EXECA"
PROCEDURE EXECA (A) , INTEGER A;
BEGIN PROCEDURE EXEC (A) , INTEGER A;
CODE; EXEC (A) ,
END;
```

(External)

```
HPAL, P,B,L, "EXECB"
PROCEDURE EXECB (A,B,C,D) , INTEGER A,B,
C,D;
BEGIN PROCEDURE EXEC (A,B,C,D) ,
INTEGER A,B,C,D , CODE , EXEC (A,B,C,D) ,
END ,
```

READ/WRITE

Purpose

To transfer information to or from an external I/O device. For a READ request, or, if the I/O device is not buffered, the program is placed in the I/O suspend list until the operation is complete. RTE then reschedules the program.

Assembler Language

```
        EXT   EXEC
         :
        JSB   EXEC        Transfer control to RTE
        DEF   *+ 5, 6 or 7  Point of return from RTE; 6 or 7 is for additional parameters
        DEF   ICODE       Request code
        DEF   ICNWD       Control information
        DEF   IBUFR       Buffer location
        DEF   IBUFL       Buffer length
        DEF   IPRM1       Optional parameter or track number if disc transfer
        DEF   IPRM2       Optional parameter or sector number if disc transfer
return point              Continue execution
         :
ICODE   DEC   1 (or 2)    1=READ, 2=WRITE
ICNWD   OCT   conwd       conwd is described in Comments
IBUFR   BSS   n           Buffer of n words
IBUFL   DEC   n (or -2n)  Same n; words (+) or characters (-)
IPRM1   DEC   f           Optional parameter or decimal track number if disc transfer
IPRM2   DEC   q           Optional parameter or decimal sector number if disc transfer
```

FORTRAN

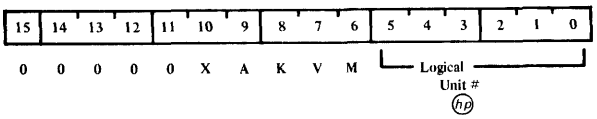
I/O transfers to most devices are programmed by standard READ and WRITE statements in FORTRAN. I/O on the disc can be done with a library subroutine BINRY (described in Comments).

COMMENTS

Parameters IPRM1 and IPRM2 are optional, except in the case of disc transfers. If the data transfer involves a disc, IPRM1 is the disc track number and IPRM2 is the disc sector number. In calls to other I/O devices these parameters may have other uses. For example, driver DVR77 (HP 2323A Subsystem) uses IPRM1 for the scanner channel number and IPRM2 for the instrument program word.

CONTROL WORD

The control word (*conwd*) required in the calling sequence contains several fields defining the nature of the data transfer:



Where

X = 1 for moving head disc write with cyclic checking.

A = 1 designates punching ASCII characters on the teleprinter (M = 0). ASCII is usually printed; but since it is sometimes desirable to punch ASCII tapes, this option is provided. (If A = 0, M determines mode of transfer.)

hp *Modified to contain request code before entry into driver.*

K = 1 causes keyboard input to be printed as received. If K = 0 input from the keyboard is not printed.

V = 1, and M = 1, causes the length of punched tape input to be determined by the word count in the first non-zero character read from the tape.

V = 0, and M = 1, the length of the punched tape input is determined by the buffer length specified in the EXEC call.
M determines the mode of data transfer (if applicable).

M = 0 for ASCII

M = 1 for binary

In an Assembler Language calling sequence, the buffer length can be a positive number for words (+) or a negative number for characters (-).

End-of-operation information is transmitted to the program in the A- and B-registers. The A-Register contains word 5 (status word) of the device EQT entry; the B-Register contains a positive number which is the number of words or characters (depending upon which the program specified) actually transmitted.

BINRY

FORTRAN programs can call BINRY, the disc read/write library subroutine, to transfer information to or from the disc. The call must specify a buffer array, the array length in words, the disc logical unit number, track number, sector number, and offset in words within the sector. (If the offset equals 0, the transfer begins on the sector boundary; if the offset equals *n*, then the transfer skips *n* words into the sector before starting.) BINRY has two entry points: BREAD for read operations and BWRIT for write operations.

For example,

```
CALL BWRIT (ARRAY,N,IDISC,ITRK,ISECT,IOFST)
CALL BREAD (ARRAY,N,IDISC,ITRK,ISECT,IOFST)
```

Where:

ARRAY = Address of the first element
N = Number of words
IDISC = Disc logical unit number
ITRK = Starting track number
ISECT = Starting sector number
IOFST = Number of words offset within a sector

Comments

There are three basic ways that data can be written on the disc in relation to sector boundaries. Care must be used in planning the WRITE statement in two of the cases to avoid losing existing data.

One form of writing data on the disc is offset=*n* (i.e., transfer begins within a sector), and less than the sector is written, or the data transfer ends on a sector boundary. The entire first sector is initially read into an internal buffer, the data is modified according to the BWRIT statement, and then the entire sector is rewritten on the disc with no data loss. No special precautions are required in this instance.

A second form of writing data on the disc is offset=0 (i.e., transfer begins on a sector boundary), and less than the sector is written. The remaining data in the sector will be lost if the following precaution is not taken. The entire existing sector on the disc can first be read into a user's buffer, modified to reflect the desired changes, and then rewritten on the disc as a full sector.

A third form of writing data on the disc is offset=0 or *n*, and a sector boundary is crossed in the data transfer. The remaining data in the final sector will be lost if the following precaution is not taken. The entire final sector (of the data transfer) on the disc should be read into a user's buffer, modified to reflect the desired changes, and then rewritten on the disc as a full sector.

NOTE

When a REAL array is transmitted, the buffer length must still be the total number of words required (i.e., 2 times REAL array length, or 3 times double precision array length).

I/O CONTROL

Purpose

To carry out various I/O control operations, such as backspace, write end-of-file, rewind, etc. If the I/O device is not buffered, the program is placed in the I/O suspend list until the control operation is complete.

Assembler Language

```
        EXT   EXEC
        :
        JSB   EXEC          Transfer control to RTE
        DEF   *+ 4 (or 3)   Point of return from RTE
        DEF   ICODE         Request code
        DEF   ICNWD         Control information
        DEF   IPRAM         Optional parameter
        return point        Continue execution
        :
ICODE   DEC   3             Request code = 3
ICNWD   OCT   conwd         See Control Word
IPRAM   DEC   n             Required for some control functions; see Control Word
```

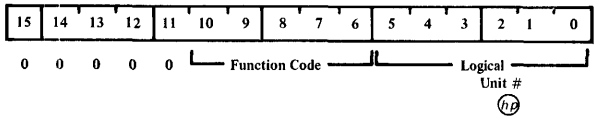
FORTRAN

Use the FORTRAN auxiliary I/O statements or an EXEC call sequence.

```
ICODE = 3           Request code
ICNWD = conwd       See Control Word
IPRAM = x           Optional; see Control Word
CALL EXEC (ICODE, ICNWD, IPRAM)
CALL EXEC (ICODE, ICNWD)
```

CONTROL WORD

The control word value (*conwd*) has two fields:



Function Code (Octal)	Action
00	Unused
01	Write end-of-file (magnetic tape)
02	Backspace one record (magnetic tape)
03	Forward space one record (magnetic tape)
04	Rewind (magnetic tape)
05	Rewind standby (magnetic tape)
06	Dynamic status (magnetic tape)
07	Set end-of-paper tape
10	Generate paper tape leader
11	List output line spacing

Function Code 11_8 (list output line spacing), requires the optional parameter IPRAM. IPRAM must designate the number of lines to be spaced on the specified logical unit. A negative parameter specifies a page eject on a line printer. For details of line printer formatting consult Appendix F.

(hp) *Modified to contain request code before entry into driver.*

I/O STATUS

Purpose

To request information (status condition and device type) about the device assigned to a logical unit number.

Assembler Language

```
        EXT   EXEC
         :
        JSB   EXEC      Transfer control to RTE
        DEF   *+ 4 (or 5)  Point of return from RTE
        DEF   ICODE     Request code
        DEF   ICNWD     Control information
        DEF   ISTA1     Status word 1
        DEF   ISTA2     Status word 2 - optional
        return point    Continue execution
         :
ICODE   DEC   13        Request code = 13
ICNWD   DEC   n         Logical unit number
ISTA1   NOP             Word 5 of EQT entry returned here
ISTA2   NOP             Word 4 returned here, optional
```

FORTRAN

```
ICODE = 13          Request code
ICNWD = nn          nn is the logical unit number
CALL EXEC (ICODE, ICNWD, ISTA1, ISTA2)
```

COMMENTS

The calling program is not suspended. The second status word is optional in the I/O Status EXEC Call. For a description of words 4 and 5 of the EQT, see Section V, Real-Time Input/Output.

DISC ALLOCATION

Purpose

To request that RTE assign a specific number of contiguous disc tracks for data storage. The tracks are either assigned to the calling program or assigned globally.

Assembler Language

```
        EXT   EXEC
         :
        JSB   EXEC      Transfer control to RTE
        DEF   *+ 6      Point of return from RTE
        DEF   ICODE     Request code
        DEF   ITRAK     Number of contiguous tracks required
        DEF   ISTRK     Start track number
        DEF   IDISC     Logical unit number
        DEF   ISECT     Number of 64 word sectors/track
        return point    Continue execution
         :
ICODE   DEC   4 or 15   4 = allocate track to program
                        15 = allocate track globally
ITRAK   DEC   n         n = number of contiguous tracks within the same disc unit requested. If
                        bit 15 of ITRAK = 1 the program is not suspended if tracks are not
                        available; if bit 15 = 0, the program is suspended until the tracks are
                        available.
ISTRK   NOP             RTE stores starting track number here, or -1 if the tracks are not
                        available.
IDISC   NOP             RTE stores disc logical unit number here.
ISECT   NOP             RTE stores number of 64 word sectors/track here.
```

FORTRAN

Example (with no suspension):

```
ICODE = 4
ITRAK = 100000B + n
CALL EXEC (ICODE, ITRAK, ISTRK, IDISC, ISECT)
```

Example (with suspension until tracks available):

```
ICODE = 4
ITRAK = n
CALL EXEC (ICODE, ITRAK, ISTRK, IDISC, ISECT)
```

COMMENTS

RTE supplies only whole tracks within one disc. When writing or reading from the tracks (see READ/WRITE EXEC Call), RTE does not provide automatic track switching; the user program is completely responsible for file management within its tracks. RTE will prevent other programs from writing on program assigned tracks, but not from reading out of them. The program retains the tracks until it or the operator releases them, or the program is aborted.

Globally assigned tracks are available to any program for READ, WRITE, or release. The user is completely responsible for their management. RTE will not prevent other programs from writing on globally assigned tracks or releasing them.

DISC RELEASE-PROGRAM TRACKS

Purpose

To release some contiguous mass storage tracks which were previously assigned to a program. (See Disc Allocation EXEC Call.)

Assembler Language

```
        EXT   EXEC
         :
        JSB   EXEC        Transfer control to RTE
        DEF   *+ 5 (or 3) Point of return from RTE
        DEF   ICODE       Request code
        DEF   ITRAK       Number of contiguous tracks, or -1
        DEF   ISTRK       Starting track number
        DEF   IDISC       Mass storage logical unit
        return point      Continue execution
         :
ICODE   DEC   5           Release program's tracks
ITRAK   DEC   n           If n = -1, release all tracks assigned to program. ISTRK and IDISC are
                          unnecessary, so the return point is *+3. Otherwise, n is the number of
                          contiguous tracks to be released starting at ISTRK.
ISTRK   DEC   m           Starting track number
IDISC   DEC   p           Mass storage logical unit
```

FORTRAN

Release of *n* contiguous tracks starting at *m* on LU *p*:

```
ICODE = 5
ITRAK = n
ISTRK = m
IDISC = p
CALL EXEC (ICODE, ITRAK, ISTRK, IDISC)
```

Release all tracks allocated to the program

```
ICODE = 5
ITRAK = -1
CALL EXEC (ICODE, ITRAK)
```

DISC RELEASE-GLOBAL TRACKS

Purpose

To release some contiguous mass storage tracks which were previously assigned globally. (See Disc Allocation EXEC Call.)

Assembler Language

```
        EXT   EXEC
         .
         .
        JSB   EXEC      Transfer control to RTE
        DEF   * + 5     Point of return from RTE
        DEF   ICODE     Request Code
        DEF   ITRAK     Number of contiguous tracks
        DEF   ISTRK     Starting track number
        DEF   IDISC     Mass storage logical unit
        return point    Continue execution
         :
ICODE   DEC   16        Release global tracks
ITRAK   DEC   n         The number of contiguous tracks to be released starting at ISTRK.
ISTRK   DEC   m         Starting track number
IDISC   DEC   p         Mass storage logical unit
```

FORTRAN

Release of *n* contiguous global tracks starting at *m* on LU *p*:

```
ICODE = 16
ITRAK = n
ISTRK = m
IDISC = p
CALL EXEC (ICODE, ITRAK, ISTRK, IDISC)
```

COMMENTS

If any one of the tracks to be released is either not assigned globally or is currently in use (i.e., some program is queued to read or write on the track at the time of the release request), none of the tracks is released.

The requesting program is rescheduled after the request with the A-Register set as follows:

- A = 0 The tracks have been released.
- A = -1 No tracks have been released – at least one of them was in use.
- A = -2 No tracks have been released – at least one of them was not assigned globally.

PROGRAM COMPLETION

Purpose

To notify RTE that the calling program is finished and wishes to terminate. RTE returns the program to the dormant list.

Assembler Language

```
        EXT   EXEC
         :
        JSB   EXEC      Transfer control to RTE
        DEF   *+ 2      Return point from RTE
        DEF   ICODE     Request code
        return point    Continue execution
         :
ICODE   DEC   6         Request code = 6
```

FORTRAN and ALGOL

The FORTRAN and ALGOL compilers generate a Program Completion EXEC Call automatically when they compile an END statement.

PROGRAM SUSPEND

Purpose

To suspend the calling program from execution until restarted by the GO operator request.

Assembler Language

```
        EXT   EXEC
         :
        JSB   EXEC      Transfer control to RTE
        DEF   *+ 2      Point of return from RTE
        DEF   ICODE     Request code
        return point    Continue execution
         :
ICODE   DEC   7         Request code = 7
```

FORTRAN and ALGOL

The FORTRAN and ALGOL library subroutine PAUSE, which is automatically called by a PAUSE statement, generates the Suspend EXEC Call.

COMMENTS

When a program is suspended, both the A- and B-Registers are saved. When the program is restarted with the GO request and no parameters, both registers are restored as they were at the point of suspension and the program continues.

When the program is restarted with a GO and parameters, the B-Register contains the address of a five-word parameter array set by the GO request. In a FORTRAN program, the library subroutine RMPAR can load these parameters using the address in the B-Register as a pointer. It must be noted, however, that when RMPAR is used, parameters

must accompany the GO request. Otherwise RMPAR uses the restored B-Register as an address to parameters which do not exist. The call to RMPAR must occur immediately following the Suspend EXEC Call, as in the following example:

```
DIMENSION I(5)      Must always specify at least 5
CALL EXEC (7)       Suspend
CALL RMPAR (I)      Return point; get parameters
```

The Program Suspend EXEC Call is similar to the SS operator request. (See Section III.)

PROGRAM SEGMENT LOAD

Purpose

To load a background segment of the calling program from the disc into the background overlay area and transfer execution control to the segment's entry point. (See Section IV, Part 7, Real-Time Programming, for information on segmented programs.)

Assembler Language

```
          EXT   EXEC
           :
          JSB   EXEC          Transfer control to RTE
          DEF   *+ 3          Point of return from RTE
          DEF   ICODE         Request code
          DEF   INAME         Segment name
          return point        Continue execution
           :
ICODE     DEC   8             Request code = 8
INAME     ASC   3,xxxxx       xxxxx is the segment name
```

FORTRAN

```
DIMENSION NAME (3)
ICODE  =  8
INAME (1) =  xxxxxB          First two characters
INAME (2) =  xxxxxB          Second two
INAME (3) =  xxxxxB          Last character in upper 8 bits
CALL EXEC (ICODE, INAME)
```

COMMENTS

See Section IV, Overlay Segments and Segmented Programs, for a description of segmented background programs.

In the FORTRAN calling sequence, the name of the segment must be converted from ASCII to octal and stored in the INAME array, two characters per word. Refer to the table in Appendix G for the ASCII to octal conversion.

PROGRAM SCHEDULE

Purpose

To schedule a dormant program for execution, and optionally, to transfer up to five parameters to that program.

Assembler Language

```
            EXT   EXEC
             :
            JSB   EXEC        Transfer control to RTE
            DEF   *+3+n       Return point, n = number of parameters
            DEF   ICODE       Request code
            DEF   INAME       Name of program to schedule
            DEF   IPRM1  ┐
             :           │    Up to five optional parameters
            DEF   IPRM5  ┘
            return point      Continue execution
             :
ICODE       DEC   9 or 10     9 = schedule with wait,
                              10 = no wait; see Comments
INAME       ASC   3,xxxxx     xxxxx is the name of the program to schedule
IPRM1  ┐
IPRM5  ┘                      Up to five optional parameters
```

FORTRAN

Can only pass value parameters, not address.

```
DIMENSION INAME (3)
ICODE      = 9 (wait) or 10 (no wait)
INAME (1) = xxxxxB  ┐
INAME (2) = xxxxxB  ├   Name of program in octal; see previous request, Comments.
INAME (3) = xxxxxB  ┘
CALL EXEC (ICODE, INAME, IPRM1 . . . IPRM5)
```

COMMENTS

If the program to be scheduled is dormant, it is scheduled and a zero is returned to the calling program in the A-Register.

If the program to be scheduled is not dormant, it is not scheduled by this call, and its status (which is some non-zero value) is returned to the calling program in the A-Register.

WAITING AND NO WAITING

A schedule with waiting (ICODE = 9) causes RTE to put the calling program in waiting status. The called program runs at its own priority, which may be greater than, less than, or equal to that of the calling program. Only when the called program terminates does RTE resume execution of the original program at the point immediately following the schedule call.

A background disc-resident program must not schedule another background disc-resident program with waiting. RTE aborts the calling program and prints the error message "SCØ3". A real-time disc-resident program, however, may schedule another real-time disc-resident program with waiting, because real-time disc-resident programs are swapped according to their priority when they conflict over use of their core area.

All other schedule combinations are legal: a disc-resident can call a core-resident, a core-resident can call a disc-resident, and a core-resident can call a core-resident.

A Schedule EXEC Call with no waiting (ICODE = 10) causes the specified program to be scheduled for execution according to its priority.

PARAMETERS

When the called program begins executing, the B-Register contains the address of a five-word list of parameters from the calling program (the parameters equal zero if none were specified). A call to the library subroutine RMPAR, the first statement of a called FORTRAN program, transfers these parameters to a specified five-word array within the called program. For example,

```
PROGRAM XQF
DIMENSION IPRAM (5)
CALL RMPAR (IPRAM)
```

The Program Schedule EXEC Call is similar to the ON operator request. (See Section II.) The Execution Time EXEC Call also schedules programs for execution, but without passing parameters.

TIME REQUEST

Purpose

To request the current time recorded in the real-time clock.

Assembler Language

```
        EXT     EXEC
         :
        JSB     EXEC        Transfer control to RTE
        DEF     *+ 3        Point of return from RTE
        DEF     ICODE       Request code
        DEF     ITIME       Time value array
        return point        Continue execution
         :
ICODE   DEC     11          Request code = 11
ITIME   BSS     5           Time value array
```

FORTRAN

```
ICODE = 11
DIMENSION ITIME (5)
CALL EXEC (ICODE, ITIME)
```

COMMENTS

When RTE returns, the time value array contains the time on a 24-hour clock:

Assembler		FORTRAN/ALGOL		
ITIME	or	ITIME(1)	=	Tens of milliseconds
ITIME+1	or	ITIME(2)	=	Seconds
ITIME+2	or	ITIME(3)	=	Minutes
ITIME+3	or	ITIME(4)	=	Hours
ITIME+4	or	ITIME(5)	=	Day of the year

The Time Request EXEC Call is similar to the TI operator request. (See Section II.)

EXECUTION TIME

Purpose

To schedule a program for execution at specified time intervals, starting after an initial offset time (Initial Time Version) or at a particular absolute time (Absolute Start-Time Version). RTE places the specified program in the time list and returns to the calling program.

Assembler Language (Initial Offset Version)

```
        EXT   EXEC
         :
        JSB   EXEC      Transfer control to RTE
        DEF   *+ 6      Point of return from RTE
        DEF   ICODE     Request code
        DEF   IPROG     Program to put in time list
        DEF   IRESL     Resolution code
        DEF   MTPLE     Execution multiple
        DEF   IOFST     Initial time offset
        return point    Continue execution
         :
ICODE   DEC   12        Request code = 12
IPROG   DEC   0         Put calling program in time list
          or
        ASC   3,xxxxx   xxxxx is the program to put in the time list
IRESL   DEC   x         x is the resolution code
MTPLE   DEC   y         y is the execution multiple
IOFST   DEC   -z        z (units set by x) gives the initial offset; the negative sign signals the
                        initial offset version to RTE
```

FORTRAN (Initial Offset Version)

```
IPROG = 0 or DIMENSION IPROG (3)
        IPROG (1)=xxxxxB  }  Define program name;
        IPROG (2)=xxxxxB  }  See Program Segment Load
        IPROG (3)=xxxxxB  }  EXEC Call
ICODE = 12
IRESL = x       see IT, Section II
MTPLE = y
IOFST = -z      z (units set by x) gives the initial offset; the negative sign signals the initial offset version to RTE.
CALL EXEC  (ICODE, IPROG, IRESL, MTPLE, IOFST)
```

Assembler Language (Absolute Start-Time Version)

The Absolute Start-Time Version is the same as the Initial Offset Version up to parameter 5 (IOFST). The sign must be positive. The positive number in parameter 5 signals RTE to expect three more parameters. The return point must be "* + 9" instead of "* + 6".

For example,

```
                DEF *+ 9
                 :
parameter five = DEF IHRS (instead of negative IOFST)
                 DEF MINS
```

Assembler Language (continued)

```
        DEF ISECS
        DEF MSECS
        return point
          :
IHRS    DEC a     Absolute starting time
MINS    DEC b     in hours, minutes, seconds,
ISECS   DEC c     and tens of milliseconds,
MSECS   DEC d     on a 24-hour clock
```

FORTRAN (Absolute Start-Time Version)

Same as Initial Offset Version with additional time parameters listed in above Assembler Language call.

CALL EXEC (ICODE, IPROG, IRESL, MTPLE, IHRS, MINS, ISECS, MSECS)

In the initial offset version only, if IPROG = 0 (current/calling program), the currently executing program is made dormant, but the point of suspension is retained. The program is then placed in the time list for rescheduling from the point of suspension after a delay.

COMMENTS

The Execution Time EXEC Call is similar to the IT operator request (see Section II, except that the EXEC call places the program in the time list whereas IT only changes the time values but does not put the program in the time list.

Notice that this call can place either itself or another program in the time list, with an execution time cycle starting at a specified real time, or starting after a fixed offset from the current time.

ERROR MESSAGES

When RTE discovers an error in an EXEC call, it terminates the program, releases any disc tracks assigned to the program, prints an error message on the operator console, and proceeds to execute the next program in the scheduled list.

When RTE aborts a program, it prints this message:

name ABORTED

When a memory protect violation occurs that is not an EXEC Call, this message is printed: (*address* is the location that caused the violation.)

MP *name address*

When an EXEC Call contains an illegal request code, this message is printed: (*address* is the location that made the illegal call.)

RQ *name address*

The general error format, for other errors, is:

type name address

Where

type is a 4-character error code.

name is the program that made the call.

address is the location of the call (equal to the exit point if the error is detected after the program suspends).

ERROR CODES FOR SCHEDULE CALLS

SCØ1 = Missing parameter

SCØ2 = Illegal parameter

SCØ3 = Program cannot be scheduled

SCØ3 INT *xx* occurs when an external interrupt attempts to schedule a program that is already scheduled. RTE ignores the interrupt and returns to the point of interruption.
xx is the interrupt location address.

SCØ5 = Program given is not defined.

SCØ6 = No resolution code in Execution Time EXEC Call.

ERROR CODES FOR DISC ALLOCATION CALLS

DRØ1 = Insufficient number of parameters

DRØ2 = Number of tracks≤zero, or>255; illegal logical unit;
or number of tracks to release is zero or negative.

DRØ3 = Attempt to release track assigned to another program

ERROR CODES FOR I/O CALLS

IOØ1 = Not enough parameters

IOØ2 = Illegal logical unit

IOØ3 = Logical unit not assigned

IOØ4 = Illegal user buffer

IOØ5 = Illegal disc track or sector

IOØ6 = Reference to a protected track; or using load-and-go before assigning load-and-go tracks (see LG, Section II)

IOØ8 = Disc transfer longer than track boundary

IOØ9 = Overflow of load-and-go area

SECTION IV
REAL-TIME PROGRAMMING

This section is divided into seven parts that describe the operating procedures and formatting conventions of background programming aids of the Real-Time Software. The memory requirements stated apply to the program plus a reasonable area for symbol tables.

PART 1. RTE EDITOR

The RTE Editor creates, lists, and edits symbolic source language tapes and disc files. The Editor requires at least 4K background disc-resident area.

PART 2. RTE FORTRAN

The FORTRAN compilers accept source programs from either an input device or a source file created by the RTE Editor, and translates the source programs into relocatable object programs. The relocatable code is punched on paper tape or stored in the load-and-go tracks of the disc or both. The basic FORTRAN compiler requires 4K, the 3-tape version of FORTRAN IV requires 5K, and the 2-tape version of FORTRAN IV requires 12K of background disc-resident area.

PART 3. RTE ALGOL

The ALGOL compiler accepts source programs from either an input device or a source file created by the RTE Editor, and translates the source programs into relocatable object programs. The relocatable code is punched on paper tape or stored in the load-and-go tracks of the disc or both. The ALGOL compiler requires 8K background disc-resident area.

PART 4. RTE ASSEMBLER

The Assembler accepts source programs from either an input device or a source file created by the RTE Editor, and translates the source programs into either absolute or relocatable object programs. Absolute code is punched in binary, suitable for execution outside of RTE. The relocatable code is punched on paper tape or stored in the load-and-go tracks of the disc or both. The Assembler requires 4K background disc-resident area.

PART 5. RTE LOADER

The loader accepts relocatable object programs from either an input device, or a file created by the Assembler, ALGOL, or FORTRAN compilers on load-and-go tracks. The program can optionally be loaded into the background and run; or the program can be loaded into the background with the DEBUG library routine linked to it; or the program can be loaded into the disc-resident user program area.

PART 6. RTE RELOCATABLE LIBRARY

This part describes the libraries used by RTE, re-entrant subroutine structure, privileged subroutine structure, and utility subroutine structure.

PART 7. SEGMENTED PROGRAMS

This part describes the procedures for writing segmented programs in Assembler, ALGOL, and FORTRAN.

LOAD-AND-GO

The RTE System provides the facility for load-and-go which is defined as compilation or assembly, loading, and executing of a user program without intervening object paper tapes. To accomplish this, the compiler or assembler stores the relocatable object code, which it generates from source statements, on the disc in a predefined group of load-and-go tracks (see LG operator request). Then separate operator requests initiate loading (ON, LOADR) and execution (ON, program). All of the operating procedures have optional parameters that specify input, output, and list device. These parameters are usually logical unit 5 for input, 4 for output, and 6 for list device. If the parameter is missing, the operating procedure defaults the parameter to these values. Other parameters, if missing, usually default to 0. Missing parameters must be separated by commas (,) unless all the parameters are missing. For example, ON, EDIT is the same as ON, EDIT,5,5,4,0 (refer to the EDIT format box). Also, if the symbolic file is to be listed (p4=1) ON, EDIT,,,,1 is the same as ON, EDIT,5,5,4,1. Each format box in all seven parts show an example. The standard system devices are referenced by logical unit numbers as follows:

Logical Unit Number	Function
1	System Teleprinter
2	System Disc
3	Auxiliary Disc
4	Standard Punch Device
5	Standard Input Device
6	Standard List Device
7	Can be assigned to any devices by the user, for the defined range of logical units.
8	
9	
10	
.	
.	
63_{10}	

PART 1
RTE EDITOR

The RTE Editor, a general-purpose background program, creates, lists, and edits symbolic source language tapes and disc files. The editor is the only program that can create and release source files.

EDITOR INPUT/OUTPUT FILES

In general, RTE Editor requires two inputs: an edit file containing edit commands, and a symbolic file containing source programs, on tape or disc. Both the edit file and the symbolic file may consist of more than one physical tape. The RTE Editor processes all properly related inputs and produces a single updated file on a paper tape, or a disc file, or a list of the source file. Both the symbolic file and the updated file consist of a series of records; each record contains 1 to 72 ASCII characters. In a disc file, records are packed within a track, and tracks are chained together.

EDITING PROCESS

The editor is turned on by an ON, EDIT operator request and reads the edit file (which it stores in core) from a specified input device other than 2, checking the file for possible format errors. The editor terminates if available memory is exceeded. After storing the edit file, the editor reads the symbolic file from a specified input device (paper tape or disc), counting each record as entered and performing the required editing before adding the record to the updated file.

The editor can also list program tapes or files and create disc files; but in these two cases, it does no editing and does not expect an edit file.

OPERATING PROCEDURES

The choice of input device for the edit file depends on the complexity of the editing. For only one or two short edit commands, using a teleprinter keyboard is faster; but for a longer edit file, it is faster to punch the file on tape and read it through the photo-reader. If the photo-reader is used, the edit file tape must be in place before scheduling the editor with an ON operator request. The format for scheduling the editor is:

ON, EDIT, *p1, p2, p3, p4*

Where

p1 = The logical unit number of the edit file input device. The standard input (logical unit 5) is used if none is specified (must not = 2). If the edit file is entered through the system teleprinter, the following message will be printed:

/EDIT:ENTER EDIT FILE:

p2 = The logical unit number of the symbolic file input device. The standard input (logical unit 5) is used if none is specified. If *p2* = 2 (disc), the symbolic file must be a disc file defined by an LS operator request preceding the ON, EDIT request or else the edit aborts (see LS, Section II). The old file is released upon completion of editing.

p3 = The logical unit number of an output device for the updated file or the listing (see *p4*). For a disc file, *p3* = 2.

p4 = The type of Editor operation:

If *p4* = 0 (or no value), a normal edit;

If *p4* = 1, the symbolic file is only listed (*p3* may not be 2).

If *p4* = 2, a disc file is generated from the symbolic file (*p3* is ignored). If *p2* = 2, the old file is not released.

Example:

ON,EDIT < is equivalent to ON,EDIT, 5,5,4,0>

MESSAGES TO OPERATOR

At the end of the edit file, RTE Editor prints the message

/EDIT. END EDIT FILE

and suspends itself.

The symbolic file must be loaded into the correct device before typing

GO, EDIT

to resume operations. If the updated file is paper tape, RTE Editor generates blank leader on the output tape.

The editor suspends itself on the end-of-tape condition. RTE outputs a message and the I/O device is set down.

The device must be declared up before continuing:

I/O ERR ET EQT#*n*

UP, *n*

Where

n is the decimal EQT entry number of the device.

Editing can continue or terminate at this point. The operator inputs:

GO, EDIT, *m*

Where

m, if 0 (or not given), means to read the next tape; if *m* is any other number, it means terminate the editing process.

If the symbolic file is on the disc, RTE Editor runs to completion without operator intervention. If the updated file is a paper tape, RTE Editor produces trailer before terminating. If the edit operation is from a disc file to a disc file, the old file is released. If the updated file is a disc file or if a disc file is created, RTE Editor prints the logical unit numbers and track number of the file in decimal:

/EDIT. TRACKS IN NEW FILE:
/EDIT: *nn, xxxx*

Editing finally terminates with the messages:

/EDIT: END OF EDIT RUN

Where

nn is the disc logical unit number and *xxxx* is the track number. This information is used with the LS operator request.

RTE Editor must be rescheduled for the next operation

EDIT COMMANDS

The edit commands, consisting of ASCII characters terminated by a RETURN and a LINE FEED, direct the editing process and have this format:

/ *ee, p1, p2, p3*

Where

The first non-blank character must be a slash (/),

ee is a one or two-character editing code, and

p1 through *p3* are either record sequence numbers or character numbers referring to the symbolic file.

Sequence numbers are from one to four digits, character numbers either one or two digits. All sequence numbers must be in ascending order, greater than zero, and unique; a particular sequence number may be used in only one edit command.

EDIT COMMAND FORMATS

Format	Function
/I, *r*	Insert new records after record *r*. The new records follow the /I command in the edit file.
/D, *r1*[,*r2*]	Delete record *r1*, or records *r1* through *r2*, inclusive.
/R, *r1*[, *r2*]	Replace record *r1*, or records *r1* through *r2*, inclusive. The replacement records follow the /R command.
/CI, *r,c*	Insert new character(s) after character *c* in record *r*. The new characters follow the /CI command.
/CD, *r,c1*,[,*c2*]	Delete character *c1*, or characters *c1* to *c2*, inclusive, in record *r*.
/CR, *r,c1*,[,*c2*]	Replace character *c1* to *c2*, inclusive, in record *r* With the character(s) following the /CR.
/E	Ends the edit file, or, without other commands, dumps a file from the disc on the output device or copies a tape.
/A	Aborts editing – useful only when entering the edit file from teleprinter.

EDIT FILE EDITING

Sometimes, a mistake is made near the end of a long edit file tape. In these cases, the editor itself can correct the mistakes on the edit tape.

For most commands, the format is the same as regular editing; but if edit commands must be replaced or inserted, the RTE Editor must be able to make a distinction. That is, it has to know whether to execute an edit command, to insert it, or to replace it. For commands to be inserted or replaced, a character "!" makes the distinction. For example,

```
/R, 6
!/D, 314, 315
```

tells RTE Editor to "replace record 6 (in the first edit file) with the record "/D, 314, 315".

EDITOR ERROR MESSAGES

RTE Editor prints error messages on the operator console in this format:

```
/EDIT. error message : illegal edit command
```

Then the RTE Editor continues with the edit file; there is no on-line correction of illegal edit commands.

Error Message	Meaning
MEM OVERFLOW	The edit file overflows available memory; RTE Editor prints the command causing the overflow. Edit terminates.
CS ERR	Illegal edit command, which is printed.
PARAM ERR	Edit command "r" or "c" is illegal: non-numeric, = 0, > 72, $r_2 \leqslant r_1, c_2 \leqslant c_1$; command printed.
SEQ ERR	"r" parameter $\leqslant$ a previous "r", or "r" greater than range of symbol file; command printed.
/I ERR	No insert source statements after /I; command printed.
/R ERR	No replacement statements after /R; command printed.
/C OVF	Character overflow in edit statement (i.e., > 72 characters)
DISK OVF	No disc space for file; edit terminates.
FILE UN	Undefined symbolic file, or LUN (edit file) = 2. Edit terminates.

PART 2
RTE FORTRAN

Regular FORTRAN and FORTRAN IV are segmented programs that execute in the background under control of RTE. The compilers consist of a main program and overlay segments, and reside in the protected area of the disc. At least 4K background disc-resident area is required to execute the regular FORTRAN compiler; 5K for the FORTRAN IV 3-tape version; and 12K for the 2-tape FORTRAN IV. Only one FORTRAN IV compiler can be used in the system at any one time.

RTE FORTRAN, a problem-oriented programming language translated by the compiler, is very similar to regular HP FORTRAN. Source programs, accepted from either an input device or a source file created by RTE Editor, are translated into relocatable object programs, and punched on paper tape and/or stored in the load-and-go tracks of the disc. The object programs can be loaded by the RTE Relocating Loader and executed by an ON operator request. When a FORTRAN program has been completely debugged, the RTE Relocating Loader can make it a permanent part of the RTE System if desired.

FORTRAN REFERENCE

For a complete description of the regular HP FORTRAN Language, read the FORTRAN Programmer's Reference Manual (02116-9015). For a complete description of the HP FORTRAN IV Language, read the FORTRAN IV Programmer's Reference Manual (5951-1321).

COMPILER OPERATION

An ON, FTN operator request schedules the regular RTE FORTRAN compiler for execution. If FORTRAN IV is used, the operator request is ON, FTN4. All other parameters are the same. Before using ON, FTN, the operator must place the source program in the input device, or, if input is from a source file, specify the file location with an LS operator request. If planning to load and go, the operator allocates load-and-go tracks with an LG operator request. The format for scheduling the FORTRAN compiler is:

ON, FTN, *p1, p2, p3, p4,* 99

or

ON, FTN4, *p1, p2, p3, p4,* 99

Where

p1 = Logical unit number of input device. Use 2 for source file input from the disc (set to 5 if not given).

p2 = Logical unit number of list device (set to 6 if not given).

p3 = Logical unit number of punch device (set to 4 if not given).

p4 = Lines/page on listing (set to 56 if not given).

99 = The load-and-go parameter. If present, the object program is stored in the load-and-go tracks for later loading. Any punching requested still occurs. The 99 may occur anywhere in the parameter list, but terminates the list.

Example:

ON,FTN < is equivalent to ON, FTN,5,6,4,56>

MESSAGES TO OPERATOR

When the end of the source tape is encountered the following message is output to the system teleprinter:

I/O ERR ET EQT #*n*

EQT #*n* is unavailable (see DN, Section II) until the operator declares it up:

UP,*n*

More than one source tape can be compiled into one program. The next source tape is loaded by placing it in the input device and declaring the device UP,*n* as shown above.

At the end of compilation, the following message is printed.

$END, FTN

and RTE executes the next scheduled background disc resident program.

Two I/O error messages may be generated by RTE when FTN attempts to write on the load-and-go tracks (RTE aborts FTN).

IOØ6

IOØ9

IOØ6 means that the load-and-go tracks were not defined by an LG operator request, and IOØ9 means that the load-and-go tracks overflowed. The operator must define more load-and-go tracks with LG and start compilation over again.

The compiler terminates abnormally if:

a. No source file is declared by LS, although logical unit 2 is given for input. Compiler error E-ØØ19 (FTN2), or ERROR Ø5 (FTN4) is printed on the list device.

b. The symbol table overflows. Compiler error E-ØØ14 (FTN2), or ERROR Ø3 (FTN4) is printed on the list device. $END,FTN does not appear after the error message using FTN2, but does appear when using FTN4.

RTE FORTRAN LANGUAGE

The RTE FORTRAN Language is similar to the regular HP FORTRAN Language. The differences are described in the next few pages. RTE FORTRAN has additional capabilities, using EXEC calls. Read Section III for complete details on the EXEC calls.

FORTRAN CONTROL STATEMENT

FTN,B,L,A

B = Binary output

L = List output

A = Assembly listing

Besides the standard options shown above, two additional compiler options, T and *n*, are available.

T

Lists the symbol table for each program in the compilation. If a "u" follows the address of a variable, that variable is undefined (the program does not assign a value to it). The A option includes this T Option.

n

n is a decimal digit (1 through 9) which specifies an error routine. The user must supply an error routine, ERR*n*. If this option does not appear, the standard library error routine, ERRØ, is used. The error routine is called when an error occurs in ALOG, SQRT, .RTOR, SIN, COS,.RTOI, EXP,. ITOI or TAN.

PROGRAM STATEMENT

The program statement, which must be the first statement in a FORTRAN source program, includes optional parameters defining the program type, priority, and time values:

PROGRAM *name* (*p1, p2, p3, p4, p5, p6, p7, p8*)

Where

name is the name of the program (and its entry point),

p1 is the program type (set to 3 for main program, or 7 for subroutines, if not given)

0 = System Program
1 = Real-Time Core-Resident
2 = Real-Time Disc-Resident
3 = Background Disc-Resident
4 = Background Core-Resident
5 = Background Segment
6 = Library (re-entrant or privileged)
7 = Utility

p2 is the priority (0-99, set to 99 if not given)

p3 is the resolution code
p4 is the execution multiple
p5 is hours
p6 is minutes
p7 is seconds
p8 is tens of milliseconds

(Time values are set to 0 if not given. See Section II, IT, for meaning)

DATA STATEMENT

The DATA statement sets initial values for variables and array elements. The format of the DATA statement is:

$$\text{DATA } k_1/d_1/, k_2/d_2/, \ldots, k_n/d_n/$$

Where

k is a list of variables and array elements separated by commas.

d is a list of constants (optionally signed) which can be immediately preceded by an integer constant (followed by an asterisk) identifying the number of times the constant is to be repeated.

/ is a separation, and is used to bind each constant list.

The elements of d_i are serially assigned to the elements of k_i, therefore, k_i and d_i must correspond one-to-one. If a list contains more than one entry, the entries must be separated by commas.

Elements of k_i may not be from COMMON.

Arrays must be defined (i.e., DIMENSION) before the DATA statements in which they appear.

Example:
```
DIMENSION A(3), I (2)
DATA A (1), A(2), A(3)/1.Ø,2.Ø,3.Ø/ ,
I (1), I (2)/ 2*1/
```

EXTERNAL STATEMENT

With the statement EXTERNAL, subroutines and functions can be passed as parameters in a subroutine or function call. For example, the routine XYZ can be passed to a subroutine if XYZ is previously declared EXTERNAL. Each program may declare up to five EXTERNAL routines.

The format of the EXTERNAL statement is

EXTERNAL $v_1, v_2, \ldots, v_5$

Where

v_1 is the entry point of a function, subroutine, or library program, which exists externally.

EXAMPLE

```
FUNCTION RMX (X,Y,A,B)
RMX=X (A) * Y (B)
END
PROGRAM ABCDE
EXTERNAL XYZ, FL1
.
Z=Q-RMX (XYZ, FL1, 3.56,4.75)
.
END
```

NOTE

If a library routine, such as SIN, is used as an EXTERNAL, the compiler changes the first letter of the entry point to "%" Special versions of the library routines exist with the first character changed to "%". See RTE Relocatable Library, Part 6, in this section.

ERROR E-ØØ18 means too many EXTERNALS.

PAUSE & STOP STATEMENTS

PAUSE causes the following message to be printed.

name: PAUSE *xxxx*

The program is then suspended (see Exec Suspend Call).

Where

name is the program name, and

xxxx is the octal number (hp) given in the PAUSE.

To restart the program, use a GO operator request. (See Section II, GO.)

STOP causes the program to be terminated after the following message.

name: STOP *xxxx*

Where

name is the program name, and

xxxx is the octal number (hp) given in STOP

(hp) *Does not require the 'B' octal designator suffix.*

ERRØ LIBRARY ROUTINE

ERRØ, the error print routine referred to under the FORTRAN control statement, prints the following message whenever an error occurs in a library routine:

name: nn xx

Where

name is the program name,

nn is the routine identifier, and

xx is the error type.

The compiler generates calls to ERRØ automatically.

If the FORTRAN control statement includes an *n* option, the call will be to ERR*n*, a routine which the user must supply.

Read the FORTRAN manual for the meaning of error codes.

PART 3
RTE ALGOL

The RTE ALGOL compiler is a segmented program requiring 8K of background disc-resident area. The compiler accepts source programs written according to regular HP ALGOL with some additions and changes.

ALGOL REFERENCE

For a complete description of the HP ALGOL Language, including error messages, read the HP ALGOL Programmer's Reference Manual (02116-9072).

COMPILER OPERATION

An ON, ALGOL operator request schedules the RTE ALGOL compiler for execution. Before using ON,ALGOL, the operator must place the source program in the input device, or, if input is from a source file, specify the file location with an LS operator request. If planning to load and go, the operator allocates load-and-go tracks with an LG operator request. The format for scheduling the ALGOL compiler is:

ON,ALGOL,*p1,p2,p3,p4*,99

Where

p1 = Logical unit number of input device. Use 2 for source file input from the disc. (Set to 5 if not given).

p2 = Logical unit number of list device (set to 6 if not given).

p3 = Logical unit number of punch device (set to 4 if not given).

p4 = Lines/page on listing (set to 56 if not given).

99 = The load-and-go parameter. If present, the object program is stored in the load-and-go tracks for later loading. Any punching requested still occurs. The 99 may occur anywhere in the parameter list, but terminates the list.

Example:

ON,ALGOL < is equivalent to ON,ALGOL,5,6,4,56>

MESSAGES TO OPERATOR

When the end of the source tape is encountered the following message is output to the system teleprinter:

I/O ERR ET EQT #*n*

EQT #*n* is unavailable (see DN, Section II) until the operator declares it up:

UP,*n*

If source input is indicated to be from the disc (by *p1* = 2 in the ON control statement), and the source pointer is not set, the diagnostic

NO SOURCE

is output to the system teleprinter and the compilation ceases.

Two I/O error messages may be generated by RTE when ALGOL attempts to write on the load-and-go tracks (RTE aborts ALGOL).

IOØ6
IOØ9

IOØ6 means that the load-and-go tracks were not defined by an LG operator request, and IOØ9 means that the load-and-go tracks overflowed. The operator must define more load-and-go tracks with LG and start compilation over again.

At the end of a program, a program-termination request is made to the Executive. No message is printed.

In case of a PAUSE statement, the following message is printed:

name: PAUSE *xxxx*

Where

name = the program name

xxxx = number which has no significance.

Execution is then suspended. To restart the program, type

GO,name[,*p1,p2,p3,p4,p5*]

RTE ALGOL LANGUAGE

The HPAL control statement for RTE ALGOL does not use the symbol option S (sense switch control). Also, after the NAM record-name, additional parameters may be specified.

ALGOL CONTROL STATEMENT

HPAL[,*s1,s2,s3,s4*],
"NAM"[,*p1,p2,p3,p4,p5,p6,p7,p8,p9*]

Where

s1 = L. Produce source program listing

s2 = A: Produce object code listing

s3 = B: Product object tape

s4 = P· A procedure only is to be compiled

"NAM" = : Program name

p1 = *n*: A digit from 1 through 9 specifying the error-routine name. A library routine, ERR*n*, with *n* = 1 –9 must be supplied by the user. If this option is not specified, the error-routine name is ERRØ. The error routine is called when an error occurs in the following routines: ALOG, SQRT, .RTOR, SIN, COS, .RTIO, EXP, .ITOI, TAN.

p2 = Program type

p3 = Priority

p4 = Resolution code (0-4)

p5 = Execution multiple (0-999)

p6 = Hours (0-23)

p7 = Minutes (0-59)

p8 = Seconds (0-59)

p9 = Tens of milliseconds (0-99)

Note that the program-name specified in "NAM" must be enclosed in quotation marks, must be a legitimate identifier, and must not contain blanks.

If no symbols are specified (*s1* through *s4*), and if load-and-go is not specified in the ON, ALGOL control statement, the program is compiled but does not produce output other than diagnostic messages.

If there is an error in the control statement, the diagnostic "HPAL????" is printed on the system teleprinter. The compiler then returns control to the system.

The parameters *p1* through *p9* must appear in the order shown above but each parameter is optional. A *p* parameter is set to 0 if not specified, except the priority parameter (*p3*) which is set to 99, the lowest priority, and the program type parameter *p2*.

Parameter *p2* is set to 3 if both *p2* and *s4* are not specified by the programmer. Parameter *p2* is set to 7 if both *p2* is not specified and *s4* is specified.

PART 4
RTE ASSEMBLER

The RTE Assembler is a segmented program requiring 4K of background disc-resident area. The Assembler consists of a main program and segments, and resides in the protected system area of the disc.

RTE Assembler Language, a machine-oriented programming language, is very similar to regular HP Extended Assembler Language. Source programs, accepted from either an input device or a source file, are translated into absolute or relocatable object programs. Absolute code is punched in binary records suitable for execution outside of RTE. ASMB can store relocatable code in the load-and-go area of the disc for on-line execution, as well as punch it on paper tape. The RTE Relocating Loader accepts Assembler Language relocatable object programs from paper tape or the load-and-go area.

The source tape passes through the input device only once, unless there is insufficient disc storage space. In this case, two passes are required. (See next page Messages To Operator.)

ASSEMBLER REFERENCE

For a complete description of the HP Assembler Language, read the Assembler Programmer's Reference Manual (02116-9014).

ASSEMBLER OPERATION

An ON operator request schedules the RTE Assembler for execution. Before using ON, ASMB, the operator must place the source program in the input device, or if input is from a source file, specify the file location with an LS operator request. If planning to load and go, the operator must allocate load-and-go tracks with an LG operator request. The format for scheduling the Assembler is:

ON, ASMB, *p1, p2, p3, p4,* 99

Where

p1 = Logical unit number of input device. Use 2 for source file input from the disc. (Set to 5 if not given).

p2 = Logical unit number of list device (set to 6 if not given).

p3 = Logical unit number of punch device (set to 4 if not given).

p4 = Lines/page on listing (set to 56 if not given).

99 = Load-and-go parameter. If present, the object program is stored on the disc for loading, and any punching requested still occurs. The 99 may occur anywhere in the parameter list, but terminates the list.

Example:

ON, ASMB <is equivalent to ON,ASMB,5,6,4,56>

MESSAGES TO OPERATOR

When the end of the source tape is encountered the following message is output to the system teleprinter:

I/O ERR ET EQT #*n*

EQT #*n* is unavailable (see Down, Section II) until the operator declares it up:

UP, *n*

More than one source tape can be assembled into one program. The next source tape is loaded by placing it in the input device and declaring the device UP, *n* as shown above.

At the end of compilation, the following message is printed:

$END ASMB

and RTE executes the next scheduled program.

If another pass of the source program is required, the following message appears at the end of pass one.

$END ASMB PASS

The operator must replace the program in the input device and type:

GO, ASMB

If an error is found in the Assembler control statement, the following message appears:

$END ASMB CS

The current assembly aborts.

If an end-of-file condition occurs before an END statement is found (LS File only), the teleprinter signals:

$END ASMB XEND

The current assembly aborts.

If source input for logical unit 2 (disc) is requested, but no file has been declared (see LS, Section II), the teleprinter signals:

$END ASMB NPRG

The current assembly aborts.

RTE generates two messages when ASMB attempts to write on the load-and-go tracks (RTE aborts ASMB).

IOØ6

IOØ9

IOØ6 means that the load-and-go tracks were not defined by an LG operator request, and IOØ9 means that the load-and-go tracks have overflowed. The operator must define more load-and-go tracks with LG and start compilation over again.

The next message is associated with each error diagnostic printed during pass 1.

nnn

nnn is the "tape" number where the error (reported on the next line of the listing) occurred. A program may consist of more than one tape. The tape counter starts with one and increments whenever an end-of-tape condition occurs (paper tape) or a blank card is encountered. When the counter increments, the numbering of source statements starts over at one.

Each error diagnostic printed during pass 2 of the assembly is associated with a different message:

PG *ppp*

ppp is the page number (in the listing) of the previous error diagnostic.

PG 000 is associated with the first error in the program.

These messages (#*nnn* and PG *ppp*) occur on a separate line, above each error diagnostic in the listing.

RTE ASSEMBLER LANGUAGE

The RTE Assembler Language is equivalent to extended Assembler Language, as defined in the Assembler Programmer's Reference Manual (02116-9014). A few language changes are required to run under RTE; programs must request certain functions, such as I/O, from the executive. These requests are made using the EXEC calls described in Section III.

ASSEMBLER CONTROL STATEMENT

The control statement has the same form as that of regular Assembler Language; and although only relocatable code can be run under RTE, the RTE Assembler accepts and assembles absolute code. Absolute code is never stored in the load-and-go area. To get absolute code, the control statement must include an "A" The "R", however, is not required for relocatable code. An "X" causes the assembler to generate non-extended arithmetic unit (EAU) code.

NAM STATEMENT

The NAM statement, which must be the first statement in an Assembler source program, includes optional parameters defining the program type, priority, and time values:

NAM *name, p1, p2, p3, p4, p5, p6, p7, p8*

Where

name is the name of the program,

p1 is the program type (set to 0 if not given):

- 0 = System program
- 1 = Real-Time core-resident
- 2 = Real-Time disc-resident
- 3 = Background disc-resident
- 4 = Background core-resident
- 5 = Background segment
- 6 = Library (re-entrant or privileged)
- 7 = Utility

p2 is the priority (0 to 99, set to 99 if not given)

p3 is the resolution code

p4 is the execution multiple

p5 is hours

p6 is minutes

p7 is seconds

p8 is tens of milliseconds

(Time values, set to 0 if not given. See Section II, IT, for meaning)

These parameters are optional; but if any one parameter is given, those preceding it must appear also.

ORB STATEMENT

RTE Assembler Language does not contain the ORB statement, since information cannot be loaded into the protected base page area by user programs. However, programs can read information from base page using absolute address operands up to 1777_8.

The memory protect feature, which protects the resident executive from alteration, interrupts the execution of a user program under these conditions:

a. Any operation that would modify the protected area or jump into it.

b. Any I/O instruction, except those referencing the switch register or overflow.

c. Any halt instruction.

Memory protect gives control to RTE when an interrupt occurs, and RTE checks whether it was an EXEC call

(JSB EXEC, JSB $LIBR, JSB $LIBX).

If not, the user program is aborted.

PART 5
RTE LOADER

The RTE Relocating Loader, a background disc-resident program, has two basic functions:

a. To relocate and load background programs (without incorporating them permanently into the system) so that they can be scheduled by an ON operator request.

b. To permanently modify – by adding or replacing – the set of disc-resident programs, both real-time and background.

The loader accepts relocatable object programs from either an input device, or a file created by the Assembler, ALGOL, or FORTRAN compilers on load-and-go tracks. The program can optionally be loaded into the background and run; or the program can be loaded into the background with the DEBUG library routine linked to it; or the program can be loaded into the disc-resident user program area.

LOAD-AND-GO

The RTE System provides the facility for load-and-go which is defined as compilation or assembly, loading, and executing of a user program without intervening object paper tapes. To accomplish this, the compiler or assembler stores the relocatable object code, which it generates from source statements, on the disc in a predefined group of load-and-go tracks (see LG operator request). Then separate operator requests initiate loading (ON, LOADR) and execution (ON, program).

RTE can store the object code of one main program, or one segment and associated subroutines, on the disc. The relocating loader locates it on the disc, relocates it into an executable absolute program unit, and initializes the load-and-go tracks for further relocatable code (i.e., once the load-and-go information is loaded, it cannot be used again).

BACKGROUND LOADING

During loading, the programs are relocated to the start of the background disc-resident area and linked to external references such as EXEC, the resident library, or the relocatable library. Any segments overlay the core area following the main program and its subroutines.

These programs, regardless of the program type recorded in the NAM record, are treated as background disc-resident programs but are not permanently added to the protected RTE System (as is done in on-line modification to the system which will be explained later). The DEBUG library subroutine can be linked to the program if desired.

The loader stores the absolute version of the program, its subroutines, and linkages on a disc track or a group of contiguous tracks; it then assigns the disc tracks to the RTE System (i.e., not available as scratch or data tracks by programs) and updates, in core only, the ID segment assigned to the program. The program and its subroutines may be as large as the background disc-resident area, except that any common region is allocated within this area, not the regular background common area.

ON-LINE MODIFICATION

Using the loader, the operator can permanently modify the set of disc-resident user programs in a configured RTE System. The loader adds new disc-resident real-time or background programs, and replaces disc-resident programs with updated versions that have the same name. The OF operator request deletes those disc-resident programs loaded into the background by the loader. Do not use the OF operator request to delete program segments that were added on-line.

When the RTE System is generated, RTGEN, the system generator, stores the programs on the disc in an absolute, packed format. Each main program is identified and located by an identification segment (ID segment) in the core resident RTE Area. For disc-resident programs, the program's disc location as well as its core memory bounds, is kept in the ID segment.

RTGEN can create a number of blank ID segments so that the loader can add new programs to the permanent system later. The addition or replacement of a program involves the conversion of relocatable programs into an absolute unit, finding space on the disc to store it, and recording information in an ID segment.

In replacing, the new program may overlay the old program's disc space only if the length of the new program plus base page linkages does not exceed the disc space formerly occupied by the old program. A track or group of tracks is allocated for program storage if adding a program, or if space requirements of a replacement program exceed those of the old. These newly allocated tracks are software-protected, but not hardware-protected.

Core-resident programs cannot be replaced because the length of the program and linkage area is not kept in the ID segment for core-resident programs, nor can they be added because this would require changing the disc-resident program area origins.

LIMITATIONS

Several limitations may prohibit the final addition or replacement of disc-resident programs:

a. A common length exceeds the original common block.

b. The base page linkages exceed the original linkage provided.

c. The length of the absolute program unit exceeds the area available.

d. Disc space is not available to store the program.

e. A blank ID segment is not available for adding a program. (A program previously added could be deleted to provide a blank ID segment.)

The disc hardware protect must be physically disabled prior to the loading (and then enabled afterwards), unless the protection is always kept disabled. RTE provides additional software protection for any tracks containing systems programs or user programs.

SEGMENTED BACKGROUND PROGRAMS

Segmented programs can be added and replaced using the loader in any order as long as the main program is always entered first.

When replacing segmented programs that were incorporated into the RTE System at generation time, there must be a one-to-one name equivalence between the old and new programs. That is, the operator must replace every segment with a new segment having the same name, or eliminate the segment with an OF,*name*,8 operator request. Additional extra segments, however, may be added in a replacement.

If enough blank ID segments exist for all the parts of the new segmented program, the original ID segments are blanked, but kept for later use of the original disc space allocated to them. If there are an insufficient number of blank ID segments for the new program, it uses the ID segments of the program being replaced and the original disc space is lost.

When replacing segmented programs that were added on-line, there must always be a complete one-to-one name replacement. In no case should the OF,*name*,8 operator request be used to delete segments. If OF,*name*,8 is used in this instance, the complete set of disc tracks allocated to the old program is released immediately, and the ID segments are blanked for further use by any program, including the new segmented program.

NEW PROGRAM ADDITION

When a new program is added, it is stored on a complete disc track or several contiguous tracks. A blank ID segment is allocated to record the program's memory and disc boundaries, name, type, priority, and time values (as done by RTGEN; see Section VI). The loader attempts to use available disc space in the system before allocating new full tracks. If new tracks must be allocated, they are assigned to the system and are software-protected.

A program added to the system must be thoroughly debugged because, once incorporated, it has all the rights of an original program. Specifically, a real-time disc-resident program has access to the real-time resident area, the block of system available memory for I/O buffers and re-entrant blocks, and the background areas.

PROGRAM REPLACEMENT

In a replacement, if the new program can fit in the disc area of the old program (both programs must have the same name), the new program uses the ID segment of the old. The new program is generated onto temporary tracks, and then, if it can fit in the old area, it is transferred. If not, it stays on the temporary tracks and a blank ID segment is assigned to it. The old ID segment is blanked but retains its disc space for later use by another program.

LOADER OPERATION (BACKGROUND)

The operator schedules the RTE Relocating Loader for execution with an ON, LOADR request. The format for scheduling the loader is:

ON,LOADR,*p1*,*p2*,*p3*,*p4*,*p5*

Where

p1 = Logical unit number of input device. If set to 99, the load-and-go tracks are used, but 99 does not terminate the parameter list. (Set to 5 if not given.)

p2 = Logical unit number of list device (set to 6 if not given).

p3 = Operation Code:

- 0 – Normal background load.
- 1 – Background load with DEBUG.
- 2 – On-Line modification (this parameter required to modify programs).
- 3 – List all programs.

p4 = Structure parameter:

- 0 – Single main program and subroutines.
- 1 – Main and segments

p5 = If 1, omit list of program names and locations during loading.

Example:

ON,LOADR < is equivalent to
ON,LOADR,5,6,0,0,0>

COMMENTS

After the loader is initiated, there is a perceptible delay while the loader allocates a disc track (or three contiguous tracks for segmented programs) and writes zeroes in every location. This means that when the program is stored on the disc, any BSS location will contain NOP instructions (i.e., zero). "WAITING FOR DISC SPACE" is printed when a track allocation cannot be made. The loader repeats the disc request and is suspended until space becomes available. This message is primarily for information, as no action is required.

Setting parameter *p3* to one (*p3* = 1) causes DEBUG to be appended to each main program and segments. The loader sets the primary entry point of each to DEBUG, rather than the user program. When the program is run, DEBUG takes control of program execution and requests instructions from the keyboard. (See "DEBUG" for legal DEBUG commands.)

If parameter *p4* equals zero (*p4* = 0), a single main program and subroutines will be merged into an absolute program unit. To load another main program, the loader must be scheduled again.

If parameter *p4* equals one (*p4* = 1), a main program and segments will be loaded. However, only one part of the segmented program may reside on the load-and-go tracks of the disc; the others must be on paper tape. If a program and segments are loaded directly from a disc the segments will be erroneously loaded as subroutines. This could lead to exceeding available core and the loss of segmentation.

Entering the Relocatable Object Code

For a main/segment load, the main program must be entered first to establish the segment area boundaries. The library must be scanned (GO,LOADR,1) after each main program and segment (except the last segment).

The loader scans the relocatable programs and subroutines as it reads them in, keeping track of any external references. If input is initially from the disc as specified by 99 in the ON statement, the loader immediately scans the library for entry points. If input is from paper tape, the loader suspends with the message:

I/O ERR ET EQT #*n*
/ LOADR: LOAD

EQT #*n* is unavailable (see DN, Section II) until the operator declares it up.

UP, *n*

The operator reschedules the loader with the GO request according to the following format:

GO,LOADR,*p1*[,*p2*]

Where

p1 = 0 – Load additional programs from the input unit.

= 1 – Load subroutines needed from the relocatable library.

= 2 – Load subroutines from the load-and-go area of the disc (the library is not automatically scanned after this operation).

= 3 – Load from the relocatable library for the last segment in a main/segment load.

p2 = 1, if present, omit the list of entry points at the end of loading.

Matching Entries and Externals

External references to resident library programs use the existing base page links to those entry points, but external references to disc-resident relocatable library subroutines cause these routines to be loaded along with the referencing program. If a segment references a library routine also referenced by the main program, the segment shares the routine loaded with the main program.

After matching all possible entry points to external references, if there are still undefined external references, the loader prints this message:

UNDEFINED EXTS

The external references are listed, one per line, and the loader suspends itself.

To load additional programs from the input unit, the operator types:

GO,LOADR

To continue, without fulfilling external references, the operator types:

GO, LOADR, 1

The loader proceeds to relocate the program or segment and subroutines into absolute format, and prints a list (on

the list device) of all entry points (unless instructed not to print the list) as each routine is loaded. The entry point listing format is:

**name address*

where

name is the entry point name, and

address is its absolute location in octal.

End of Loading

At the end of a normal load, or after loading the last segment of a main/segment load, the loader prints this message:

name READY-LOADING COMPLETE

where

name is the name of the main user program. The loader then terminates.

After loading a main or segment of a main-segment load (end-of-tape mark) the loader prints this message:

/LOADR: LOAD

and waits for the GO,LOADR entry for the next segment.

After entering the last segment and subroutines from the input device (not the disc), the operator reschedules the loader with a

GO,LOADR,3

and the loader proceeds to the end of loading, as above.

The operator can schedule the program for execution by an ON,*name* operator request (see Section II, ON). The disc tracks containing the program are assigned to the system and are software-protected. The program can be eliminated from the system by the OF,*name*,8 operator request (see Section II, OF).

RELOCATING LOADER MESSAGES

Messages are printed in this format:

/LOADR: *message*

ERROR MESSAGES

LØ1 – Checksum error

LØ2 – Illegal record

These errors are recoverable. The offending record can be reread by repositioning the tape and typing:

GO,LOADR

For irrecoverable errors, one of the following messages is printed, followed by "LOADR ABORTED" (the loader is terminated):

LØ3 – Memory overflow

LØ4 – Base page linkage area overflow

LØ5 – Symbol table area overflow

LØ6 – Common block error

a. Exceeding allocation in a replacement or addition

b. In a normal background load, first program did not declare largest common block

LØ7 – Duplicate entry points

LØ8 – No transfer address (main program) in the program unit. Another program may be entered with a GO operator request. (This also occurs when load-and-go is specified, but no program exists in the load-and-go area.)

LØ9 – Record out of sequence

L1Ø – Operator request parameter error. GO requests may be retyped; ON requests may not.

ADDITIONAL MESSAGES

"NO BLANK ID SEGMENTS" is printed when available (i.e., blank) ID segment is not found. The loader calls for program suspension. The operator may then delete a program from the system (OF operator request) or may terminate the loader.

"DUPLICATE PROG NAME – name" is printed when a program name is already defined in the system for a normal load or a program addition. The loader changes the name of the current program by replacing the first two characters with "# #".

"WAITING FOR DISC SPACE" is printed when a track allocation cannot be made. The loader repeats the disc request and is suspended until space becomes available. This message is primarily for information, as no action is required.

"UNDEFINED EXTS" is printed followed by a list of all remaining undefined external symbols after a scan of the library. Additional programs may be loaded by the GO operator request.

"LOAD" is printed and the loader is suspended whenever an end-of-tape condition is detected from the input unit.

LISTING PROGRAMS

If parameter *p3* equals three (*p3* = 3), a listing of all the programs and blank ID segments in the system is printed.

ON,LOADR,5,6,3

For each ID segment in the system, this line is printed:

name t pr

Where

name is the program name

t is the program type:

1 – Real-Time resident.
2 – Real-Time disc-resident.
3 – Background disc resident.
4 – Background resident.
5 – Background segment.

pr is the program priority, from 1 to 99.

A blank (i.e., available for use by the loader) ID segment is noted by the line:

<BLANK ID SEGMENT>

The loader terminates after the list is complete

Example

ON,LOADR, , , 3 <is equivalent to
ON,LOADR, 5,6,3>

LOADER OPERATION (ON-LINE)

The operator schedules the RTE Relocating Loader for execution with an ON, LOADR request. The format for scheduling the loader is:

ON,LOADR,*p1*,*p2*,*p3*,*p4*,*p5*

Where

p1 = Logical unit number of input device. If set to 99, the load-and-go tracks are used, but 99 does not terminate the parameter list. (Set to 5 if not given.)

p2 = Logical unit number of list device (set to 6 if not given).

p3 = Operation code:

∅ – Normal background load.
1 – Background load with DEBUG.
2 – On-line modification (this parameter required to modify programs).
3 – List all programs.

p4 = Structure parameter:

∅ – Single main program and subroutines.
1 – Main and segments.

p5 = If 1, omit list of program names and locations during loading.

Example: ON,LOADR,,,2,1,1 <is equivalent to ON,LOADR,5,6,2,1,1>

COMMENTS

The operator schedules the loader for on-line modifications to the system by setting parameter *p3* equal to two (*p3* = 2). The loader requires additional information to carry out the modifications, so it prints the message,

/LOADR: "GO" WITH EDIT PARAMETERS

and suspends itself. The operator, after checking that the hardware disc protect is disabled, reschedules the loader with a GO operator request.

GO,LOADR,*p1*,*p2*[,*p3*]

Where

p1 = 1 for an addition operation, or

= 2 for a replacement operation (program must be dormant).

p2 = 2 for a real-time disc-resident program, or

3 for a background disc-resident program.

p3 = priority (optional) from 0 to 99 (a 0 means use the priority value in the NAM record of the program or, if that priority is 0, use 99).

Any errors cause the error message "L1Ø" to be printed. The disc hardware protect must be disabled before the program is loaded, then re-enabled after loading.

When the GO request is entered, the loader proceeds to load the program, keeping track of any external references. If input is from the disc as specified by 99 in the ON statement, the loader immediately scans the library for entry points. If input is from paper tape, the loader suspends with the message.

I/O ERR ET EQT #*n*
/LOADR: LOAD

EQT #*n* is unavailable (see DN, Section II) until the operator declares it up.

UP,*n*

The operator reschedules the loader with the GO request exactly as described previously under GO, LOADR, *p1*[,*p2*] (background).

RTE DEBUG LIBRARY SUBROUTINE

DEBUG, a utility subroutine of the RTE Relocatable Library, allows programs to be debugged (i.e., checked for logical errors on-line) during execution. Programs that expect starting parameters or that call RMPAR (see Section III, Program Suspend Exec Call) cannot use DEBUG because DEBUG uses the parameters.

The first parameter of the ON request specifies a teleprinter logical unit (DEBUG uses 1, if none is given).

The operator schedules the loader for execution with the usual ON,LOADR request:

ON,LOADR,*p1*,*p2*,*p3*,*p4*,*p5*

If the *p3* parameter of the ON,LOADR operator request equals one, the loader combines DEBUG with the main program and segments being loaded. The primary entry point (the location where execution begins) is set to DEBUG so that when the program is turned on with an ON operator request, DEBUG takes control and prints a message:

BEGIN 'DEBUG' OPERATION

The programmer can enter any legal debug operation. Illegal requests are ignored and a message is printed:

ENTRY ERROR

For further details on the uses of DEBUG, refer to the Pocket Guide to Hewlett-Packard Computers (5950-8313) BCS Section, where the non-RTE DEBUG routine is described.

DEBUG OPERATIONS

B,A	Instruction breakpoint at address A. (NOTE: If A = JSB EXEC, a memory protect violation occurs.)
D,A,N1[,N2]	ASCII dump of core address N1 or from N1 to N2.
D,B,N1[,N2]	Binary dump of core address N1 or from N1 to N2.
M,A	Sets absolute base of relocatable program unit.
R,A	Execute user program starting at A. Execute starting at next location in user program (used after a breakpoint or to initiate the program at the transfer point in the user program).
S,A1,D1	Set D1 in location A1.
S,A1,D1 .D*n*	Set D1 to D*n* in successive memory locations beginning at location A1.
W,A,D1	Set A-Register to D1.
W,B,D2	Set B-Register to D2.
W,E,D3	Set E-Register (0 = off, non-zero = on).
W,O,D4	Set Overflow (0 = off, non-zero = on).
X,A	Clear breakpoint at address A.
A	Abort Debug operation.

PART 6
RTE RELOCATABLE LIBRARY

There are two libraries or collections of relocatable subroutines that can be used by RTE: the RTE/DOS Relocatable Library, EAU version (F2E.n), and the RTE/DOS FORTRAN IV Library (F4D.n). When the system is generated, it must contain the F2E.n library while the F4D.n library is optional. The F4D.n library contains a formatter, extended precision, and other routines that reference routines from the F2E.n library. If extended precision is not required but the formatter is, a separate RTE/DOS Basic FORTRAN Formatter is available.

The F2E.n library also contains mathematical and utility routines such as SIN, COS, BINRY, etc. A program signifies its need for a subroutine by means of an "external reference", External references are generated by EXT statements in Assembler Language, by CALL statements and the compiler in FORTRAN, and by CODE procedures and the compiler in ALGOL.

All of these libraries and the subroutines they contain are documented in the Relocatable Subroutine Manual (02116-91780).

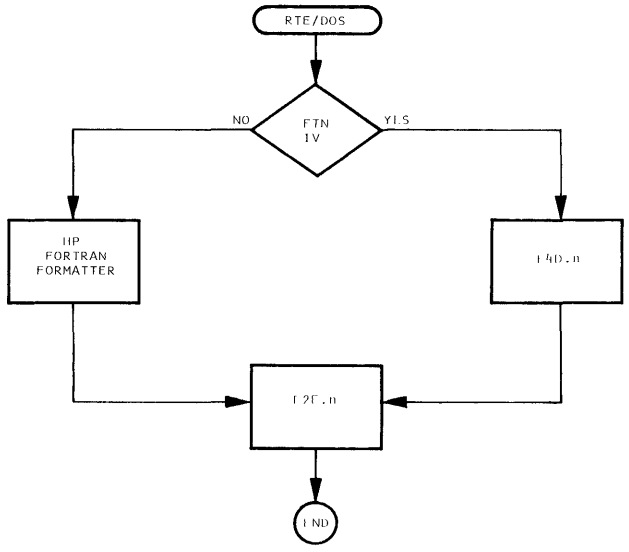


Figure 4-1. RTE Library Configuration Diagram

RE-ENTRANT SUBROUTINE STRUCTURE

Many executing programs can reference one resident library subroutine on a priority basis. If the subroutine is structured as re-entrant, it must not modify any of its own instructions; and it must save temporary results, flags, etc., if it is called again (by a higher priority program) before completing its current task.

Each time the re-entrant routine begins executing, the address and length of its temporary data block are transferred to RTE through entry point $LIBR to save the data. At the end of execution, the re-entrant routine calls RTE through entry point $LIBX to restore the temporary data, if any.

Re-entrant structure is used for programs with an execution time exceeding one millisecond. For shorter execution times, the overhead time RTE uses in saving and restoring temporary data makes re-entrant structure unreasonable. Faster subroutines can be structured as privileged.

NOTE

It is illegal for a type 6 program (library) to call any program other than type 6 or type 0 (system). If this occurs, RTGEN will issue an Error 15 when attempting to put the program in the library.

FORMAT OF RE-ENTRANT ROUTINE

```
        EXT   $LIBR,$LIBX
ENTRY   NOP                 Entry point of routine
        JSB   $LIBR         Call RTE to save temporary data
        DEF   TDB           Address of temporary data
        :                   Program instructions
EXIT    JSB   $LIBX         Call RTE to restore data
        DEF   TDB
        DEC   n             n is for routines with two return points in the calling program; 0 specifies the error-print return and 1 the normal return. For routines with only one return point, n=0.
```

```
TDB     NOP             Linkage address to previous block
        DEC   n         Total length of block
        NOP             Return address to calling program
        :               Temporary data
```

PRIVILEGED SUBROUTINE STRUCTURE

Privileged subroutines execute with the interrupt system turned off. This feature allows many programs to use a single privileged subroutine without incurring re-entrant overhead. As a result, privileged subroutines need not save temporary data blocks but must be very quick in execution to minimize the time that the interrupt system is disabled.

FORMAT OF PRIVILEGED ROUTINE

```
        EXT   $LIBR,$LIBX
ENTRY   NOP             Entry point to the routine
        JSB   $LIBR     Call RTE to disable the interrupt system
        NOP             Denotes privileged format

EXIT    JSB   $LIBX     Call RTE to return to calling program and enable interrupts
        DEF   ENTRY     Location of return address
```

UTILITY SUBROUTINE STRUCTURE

Utility subroutines are subroutines which cannot be shared by several programs because of internal design or I/O operations. A copy of the utility routine is appended to every program that calls for it. The library subroutine FRMTR, which carries out FORTRAN I/O operations and the PAUSE subroutine are examples of utility routines.

When RTGEN creates the disc-resident RTE System, all library subroutines not included in the resident library are stored on the disc in relocatable format as utility routines. They are appended onto programs by the RTE Relocating Loader in its background loading process (see RTE Relocating Loader).

PART 7
SEGMENTED PROGRAMS

Background disc-resident programs may be structured into a main program and several overlapping segments, as shown in Figure 4-2. The main program begins from the start of the background disc-resident area, and must be loaded during RTGEN or with the loader prior to its segments. The area for overlay segments starts immediately following the last location of the main program. The segments reside permanently on the disc, and are read in by an EXEC Call when needed. Only one segment may reside in core at a time.

RTE ALGOL SEGMENTATION

ALGOL programs can be segmented if certain conventions are followed. The main program must be type 3, and the segment must be type 5 in the HPAL statement. The segment must be initiated using the Program Segment Load EXEC Call from the main or another segment.

To establish the proper linkage between a main program and its segments, each segment must declare the main program a CODE procedure. For example, if MAIN is the main program, the following must be declared in each segment:

```
PROCEDURE MAIN; CODE;
```

Chaining of segments is unidirectional. Once a segment is loaded, execution transfers to it. The segment, in turn, may call another segment using an EXEC Call, but a segment written in ALGOL cannot return to the main program.

RTE FORTRAN SEGMENTATION

FORTRAN programs can be segmented if certain conventions are followed. The main program must be type 3, and the segment must be type 5 in the PROGRAM statement. The segment must be initiated using the Program Segment Load EXEC Call from the main or another segment.

Each segment must make a dummy call to the main program. They must not use data statements. In this way, the proper linkage is established between mains and segments:

```
CALL MAIN
END
```

Chaining of segments is unidirectional. Once a segment is loaded, execution transfers to it. The segment, in turn, may call another segment, but a segment written in FORTRAN cannot return to the main program. All communication between the main program and segments must be through COMMON.

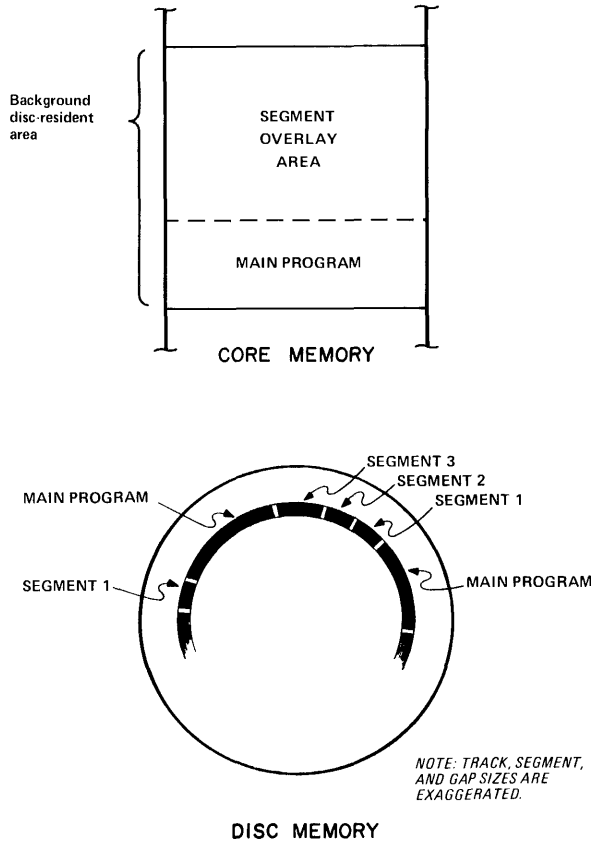


Figure 4-2. Segmented Programs

RTE ASSEMBLER SEGMENTATION

The main program must be type 3, and the segments must be type 5. One external reference from each segment to its main program is required for RTGEN to link the segments and main programs. Also, each segmented program should use unique external reference symbols. Otherwise, RTGEN or the loader may link segments and main programs incorrectly.

Figure 4-3 shows how an executing main program may call in any of its segments from the disc, via a "JSB EXEC". The main program is not suspended, but control is passed to the transfer point of the segment.

An executing segment may itself call in another of the main program's segments using the same "JSB EXEC" request. (See Figure 4-4.) However, a segment of the FORTRAN or ALGOL Compiler may not call in a segment of the Assembler.

When a main program and segment are currently residing in core, they operate as one single program. Jumps from a segment to a main program (or vice versa) can be programmed by declaring an external symbol and referencing it via a JMP instruction. (See Figure 4-5.) A matching entry symbol must be defined at the destination in the other program. RTGEN and the loader associate the main programs and segments, replacing the symbolic linkage with actual absolute addresses (i.e., a jump into a segment is executed as a jump to a specific address). The programmer should be sure that the correct segment is in core before any JMP instructions are executed.

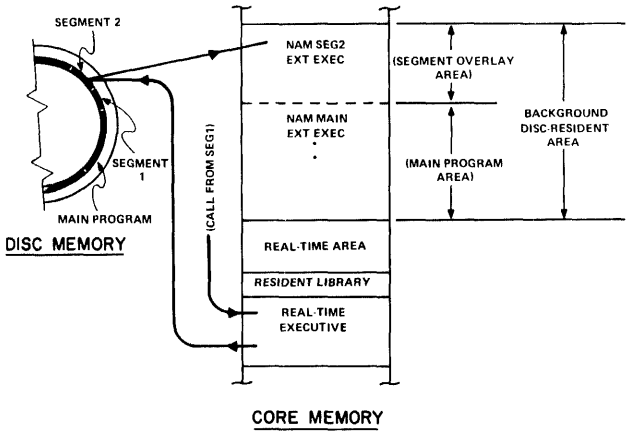


Figure 4–4. Segment Calling Segment

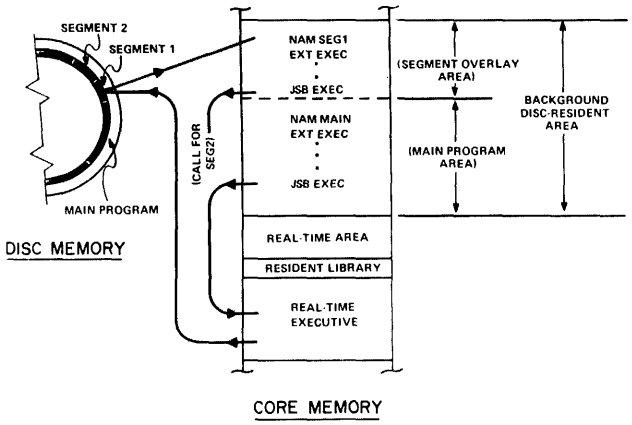


Figure 4–3. Main Calling Segment

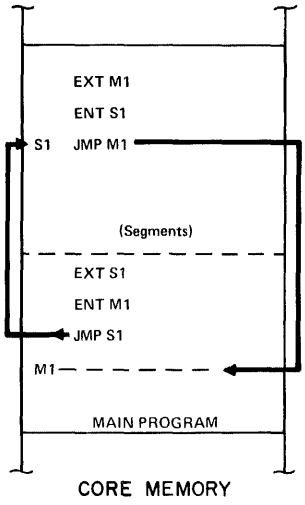


Figure 4–5. Main–To–Segment Jumps

SECTION V
REAL-TIME INPUT/OUTPUT

In the Real-Time Executive System, centralized control and logical referencing of I/O operations effect simple, device-independent programming. Each I/O device is interfaced to the computer through one or more I/O channels which are linked by hardware to corresponding core locations for interrupt processing. By means of several user-defined I/O tables, self-contained multi-device drivers, and program EXEC calls, RTE relieves the programmer of most I/O problems.

For further details on the hardware input/output organization, consult the appropriate computer manuals.

SOFTWARE I/O STRUCTURE

An equipment table records each device's I/O channels, driver, DMA, buffering and time-out specifications. A device reference table assigns one or more logical unit numbers to each entry in the equipment table, thus allowing the programmer to reference changeable logical units instead of fixed physical units.

An interrupt table directs RTE's action when an interrupt occurs on any channel; RTE can call a driver, schedule a specified program, or handle the interrupt itself.

Drivers are responsible for initiating and continuing operations on all devices of an equivalent type.

The programmer requests I/O by means of an EXEC call in which he specifies the logical unit, control information, buffer location, buffer length, and type of operation. Other devices (e.g., disc or HP 2323A Subsystem) may require additional parameters.

THE EQUIPMENT TABLE

The equipment table (EQT) has an entry for each device recognized by RTE (these entries are established by the user when the RTE System is generated). These EQT entries reside in the permanent core-resident part of the system, and have this format:

Table 5-1. Equipment Table Entry Diagram

Word	Contents															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	Device Suspended List Pointer															
2	Driver "Initiation" Section Address															
3	Driver "Completion" Section Address															
4	D	B	Not Used			T	Not Used			Unit #			Channel #			
5	AV		EQUIP. TYPE CODE					STATUS								
6	CONWD (Current I/O Request Word)															
7	Request Buffer Address															
8	Request Buffer Length															
9	Temporary, Disc Track #, or Optional Parameter															
10	Temporary, Disc Sector #, or Optional Parameter															
11	Temporary Storage for Driver															
12	Temporary Storage for Driver															
13	Temporary Storage for Driver															
14	Device Time-Out Value															
15	Device Time-Out Clock															

Where

D = 1 if DMA required.

B = 1 if automatic output buffering used.

T = 1 if device timed out (system sets to zero before each I/O request).

Unit = Last sub-channel addressed.

Channel = I/O select code for device (lower number if a multi-board interface).

AV = availability indicator:

0 = available for use.

1 = disabled (down).

2 = busy (currently in operation).

3 = waiting for an available DMA channel.

STATUS = the actual physical status or simulated status at the end of each operation. For paper tape devices, two status conditions are simulated: Bit 5 = 1 means end-of-tape on input, or tape supply low on output.

EQUIP. TYPE CODE = type of device. When this number is linked with "DVR", it identifies the device's software driver routine:

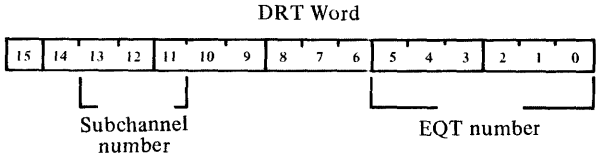
00 to 07_8 = paper tape devices (or system control devices)

00 = teleprinter (or system keyboard control device)
01 = photo-reader
02 = paper tape punch
10 to 17 = unit record devices
10 = plotter
12 = line printer
20 to 37 = magnetic tape/mass storage devices
30 = fixed head disc or drum
31 = moving head disc
40 to 77 = instruments
CONWD = user control word supplied in the I/O EXEC Call (see Section III).

When RTE initiates or continues an I/O operation, it places the addresses of the EQT entry for the device into the base page communication area (see Appendix A) before calling the driver routine.

LOGICAL UNIT NUMBERS

Logical unit numbers from 1 to 63_{10} provide logical addressing of the physical devices defined in the EQT and the subchannels within the physical devices (if applicable). These numbers are maintained in the Device Reference Table (DRT), which is created by RTGEN and can be modified by the LU operator request.



Each one-word entry in the DRT contains the EQT entry number of the device assigned to the logical unit, and the subchannel number within the EQT entry. The functions of logical units 1 through 6_{10} are predefined in the RTE System:

1 – system teleprinter
2 – system mass storage
3 – auxiliary mass storage
4 – standard punch unit
5 – standard input unit
6 – standard list unit

Logical units 7 through 63_{10} may be used for any functions desired. The operator can assign EQT numbers and subchannel numbers within the EQT entries to the logical unit numbers when the RTE System is generated (see Section VI) or after the system is running (see Section II, LU). The user determines the number of logical units when the system is generated.

Logical unit numbers are used by executing programs to specify on which device I/O transfers are to be carried out. In an I/O EXEC Call, the program simply specifies a logical unit number and does not need to know which actual device or which I/O channel and subchannel handles the transfer.

THE INTERRUPT TABLE

The interrupt table contains an entry, established at system generation time, for each I/O channel in the computer. If the entry is equal to 0, the channel is undefined in the system. If an interrupt occurs on one of these channels, RTE prints this message:

ILL INT *xx*

where *xx* is the octal I/O channel number. RTE then clears the interrupt flag on the channel and returns to the point of interruption.

If the contents of the entry are positive, the entry contains the address of the EQT entry for the device on the channel. If the contents are negative, the entry contains the negative of the address for the ID segment of a program to be scheduled whenever an interrupt occurs on the channel.

The interrupt locations in core contain a JSB $CIC; CIC is the central interrupt control routine which examines the interrupt table to decide what action to take. On a power failure interrupt RTE halts (however, the user can write his own routine to handle power failure interrupts). If privileged interrupt processing is included in the system, the privileged channels bypass $CIC and the interrupt table entirely.

INPUT/OUTPUT DRIVERS

The I/O driver routines, part of the resident RTE, handle the actual transfer of information between the computer and external devices. When a transfer is initiated, RTE places the EQT entry addresses into the base page communication area and jumps to the driver entry point. The driver configures itself for the particular channel (in this way the same driver can handle devices of the same type on many channels), initiates the transfer and returns to RTE. When an interrupt occurs on the channel, indicating continuation or completion of the transfer, RTE again transfers control to the driver.

The RTE System always includes the following standard I/O drivers. Other peripheral instrument and subsystem RTE drivers are available.

DVR00 – teleprinter driver
DVR01 – photo-reader driver
DVR30 – fixed head disc/drum driver
and/or
DVR31 – moving head disc driver

The driver name consists of the letters "DVR" added to the equipment type code (see equipment table). In addition, the programmer can write drivers for special devices, following the guidelines in this section. The driver is only responsible for updating the STATUS field in the EQT entry; RTE handles the availability field.

PROGRAMMED I/O

A user program makes an EXEC Call to initiate I/O transfers. If the device is not buffered, or in the case of all input transfers, the RTE places the calling program into I/O suspension until the transfer is complete so that other programs competing for execution time can have control.

A program in the real-time disc-resident area (type 2) cannot be swapped while it is in I/O suspension. There is, however, one exception; that is, if the I/O buffer is wholly within the system's real-time common area. (Note: the FORMATTER'S internal I/O buffer is not in real-time common.) This feature must be used with extreme caution. If the I/O buffer extends beyond the common area, incoming data can destroy the swapped-in program or, vice versa, the outgoing data might be destroyed by the swapped-in program.

PLANNING I/O DRIVERS

Before attempting to program an I/O driver, the programmer should be thoroughly familiar with Hewlett-Packard computer hardware I/O organization, interface kits, computer I/O instructions and DMA.

An I/O driver, operating under control of the input/output control (RTIOC) and central interrupt control (CIC) modules of RTE, is responsible for all data transfer between an I/O device and the computer. The device EQT entry contains the parameters of the transfer, and the base page communication area contains the number of the allocated DMA channel, if required. It should be noted that RTE operation makes it mandatory that a synchronous device driver must use a DMA or privileged interrupt channel for data transfer.

An I/O driver always has an initiation section and usually a completion section. If *nn* is the octal equipment type code of the device, I*.nn* and C*.nn* are the entry point names of the two sections respectively, and DVR*nn* is the driver name. Privileged drivers are in a special class. Refer to the end of this section for a discussion of privileged drivers.

INITIATION SECTION

The RTIOC module of RTE calls the initiation section directly when an I/O transfer is initiated. Locations EQT1 through EQT15 of the base page communication area (see Appendix A) contain the addresses of the appropriate EQT entry. CHAN in base page contains the number of the DMA channel assigned to the device, if needed. This section is entered by a jump subroutine to the entry point, I*.nn*. The A-Register contains the select code (channel number) of the device (bits 0 through 5 of EQT entry word 4). The driver returns to IOC by an indirect jump through I*.nn*.

Before transferring to I*.nn*, RTE places the request parameters from the user program's EXEC Call into words 6 through 10 of the EQT entry. The subchannel number is placed into bits 6 through 8 of word 4. Word 6, CONWD, is modified to contain the request code in bits 0 through 5 in place of the logical unit. See the EQT entry diagram in Table 5-1, and Section III, Read/Write Exec Call, for details of the parameters.

Once initiated, the driver can use words 6 through 13 of the EQT entry in any way, but words 1 through 4 must not be altered. The driver updates the status field in word 5, if appropriate, but the rest of word 5 must not be altered.

Functions of the Initiation Section

The initiation section of the driver operates with the interrupt system disabled (or as if it were disabled, in the case of privileged interrupt processing; see the discussion of special conditions under "Privileged Interrupt Processing").

The initiation section of the driver is responsible for these functions (as flow-charted in Figure 5-1).

1. Rejects the request and proceeds to step 6 if:
   a. the device is inoperable,
   b. the request code, or other of the parameters, is illegal.
2. Configures all I/O instructions in the driver to include the select code (and DMA channel) of the device.
3. Initializes DMA, if appropriate.
4. Initializes software flags and activates the device. All variable information pertinent to the transmission must be saved in the EQT entry because the driver may be called for another device before the first operation is complete.
5. Optionally sets the device time-out clock (EQT15).
6. Returns to RTIOC with an A-Register set to indicate initiation or rejection and the cause of the reject:

   If A = 0, then operation was initiated.

   If A = 1, 2, 3, then operation rejected because:
   1 – read or write illegal for device,
   2 – control request illegal or undefined,
   3 – equipment malfunction or not ready,

   If A = 4, immediate completion. (Transmission log should be returned in the B-Register in this case.)

DMA Initialization

If a driver requires DMA but does not require or use the DMA interrupt, the DMA control should be cleared after DMA initialization. Further special processing is not required in this case.

If a driver requires DMA, and the DMA interrupt, special processing must be included in the driver. After disabling the interrupt system, initiating DMA and clearing control, the driver sets a software flag to indicate that a DMA channel is active.

The software flag is either the first or second word of the interrupt table, depending on which DMA channel is used. The flag is set by making bit 15 equal to 1.

INTBL (1) – channel 1 (location 6)

INTBL (2) – channel 2 (location 7)

The address of INTBL is contained in the word INTBA in the base page communication area. When bit 15 is set, the rest of the word must not be altered. The operation can be performed only if DUMMY is non-zero (meaning the system includes privileged interrupt processing.)

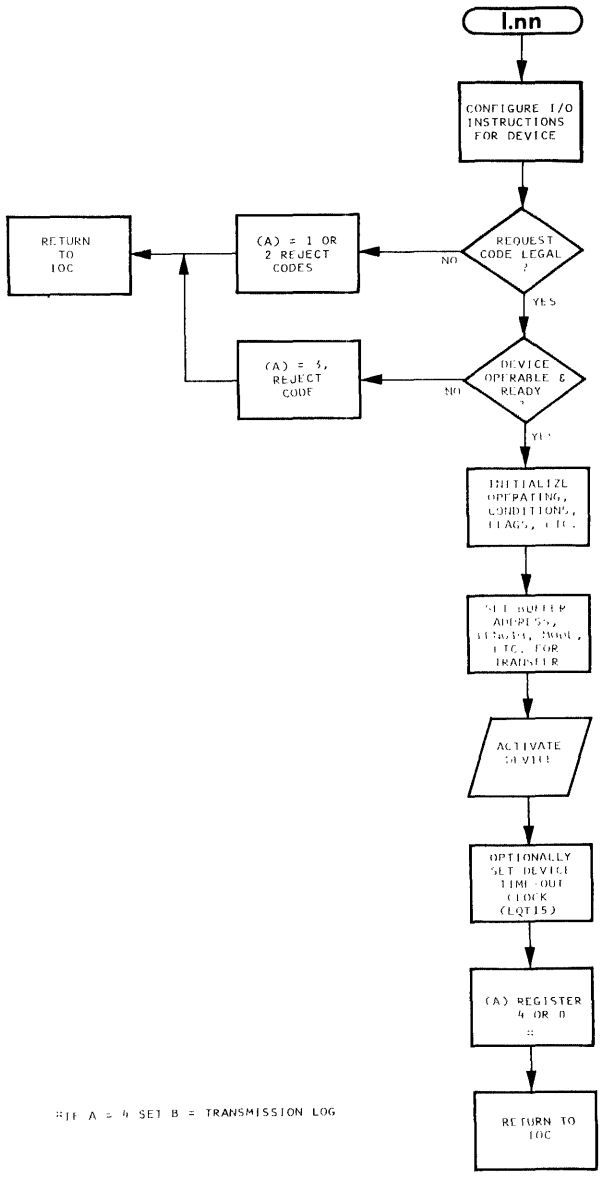


Figure 5-1. I/O Driver Initiation Section

The following code demonstrates these principles:

```
CLF 0                 Disable interrupts.
STC DMA, C            Initiate DMA
CLA
CPA DUMMY             Bypass this section if
JMP X                 DUMMY = 0 and special
                      processing is not needed.
CLC DMA
LDB INTBA             Clear DMA control. Set B =
LDA CHAN              address of the appropriate
CPA = D7              entry in the interrupt table.
INB
LDA B,I
IOR = B100000         Set bit 15 of the entry equal
STA B,I               to 1 and return to the inter-
STF 0                 rupt table. Enable interrupt
                      system.
X  continue
```

COMPLETION SECTION

RTE calls the completion section of the driver whenever an interrupt is recognized on a device associated with the driver. Before calling the driver, CIC sets the EQT entry addresses in base page, sets the interrupt source code (select code) in the A-Register, and clears the I/O interface or DMA flag. The interrupt system is disabled (or appears to be disabled if privileged interrupt processing is present). The calling sequence for the completion section is:

Location	Action
	Set A-register equal to interrupt source code
(P)	JSB C.*nn*
(P+1)	completion return from C.*nn*
(P+2)	continuation or error retry return from C.*nn*

The return points from C.*nn* to CIC indicate whether the transfer is continuing or has been completed (in which case, end-of-operation status is returned also).

The completion section of the driver is responsible for these functions (as flow-charted in Figure 5-2):

1. Checks whether word 1 (device suspended list pointer) of the EQT entry equals zero. If it does, a spurious interrupt has occurred (i.e., no I/O operation was in process on the device). The driver ignores the interrupt, sets EQT 15 (time-out clock) to zero to prevent time-out, and makes a continuation return. If not zero, the driver configures all I/O instructions in the completion section to reference the interrupting device, and then proceeds to step 2.

2. If both DMA and the device completion interrupts are expected and the device interrupt is significant, the DMA interrupt is ignored by returning to CIC in a continuation return.

3. Performs the input or output of the next data item if the device is driven under program control. If the transfer is not completed, the driver proceeds to step 6.
4. If the driver detects a transmission error, it can re-initiate the transfer and attempt a retransmission. A counter for the number of retry attempts can be kept in the EQT. The return to CIC must be (P+2) as in step 6.
5. At the end of a successful transfer or after completing the retry procedure, the following information must be set before returning to CIC at (P+1):
   a. Set the actual or simulated device status, into bits 0 through 7 of EQT word 5.
   b. Set the number of words or characters (depending on which the user requested) transmitted into the B-Register.
   c. Set the A-Register to indicate successful or unsuccessful completion and the reason:

      A equals 0 for successful operation,

      A does not equal 0 for unsuccessful:

      1 – device malfunction or not ready,

      2 – end-of-tape (information),

      3 – transmission parity error.
6. Clears the device and DMA control, if end-of-operation, or sets the device and DMA for the next transfer or retry. If not end-of-operation (i.e., a continuation exit is to be made), the driver can again optionally set the device time-out clock. Returns to CIC at:

   (P+1) – completion, with the A and B-Registers set as in step 5.

   (P+2) – continuation; the registers are not significant.

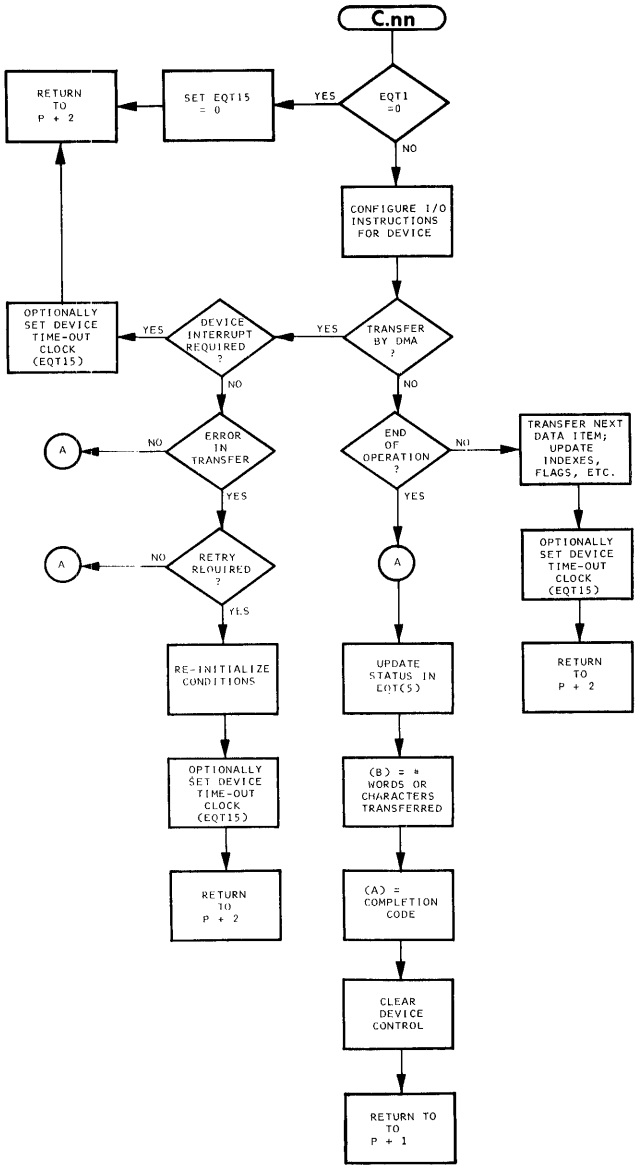


Figure 5-2. I/O Driver Completion Section

I/O DEVICE TIME-OUT

Each I/O device can have a time-out clock that will prevent indefinite I/O suspension. Indefinite I/O suspension can occur when a program initiates I/O, and the device fails to return a flag (possible hardware malfunction or improper program encoding). Without the device time-out, the program which made the I/O call would remain in I/O suspension indefinitely awaiting the operation-done indication from the device. With respect to privileged drivers, the time-out parameter must be long enough to cover the period from I/O initiation to transfer completion.

Two words, EQT14 and EQT15, of the EQT entry for each I/O device function as a device time-out clock. EQT15 is the actual working clock, and before each I/O transfer is initiated, is set to a value *m*, where *m* is a number of 10 ms time intervals. If the device does not interrupt within the required time interval, it is to be considered as having "timed out". The EQT15 clock word for each device can be individually set by two methods.

1. The system inserts the contents of EQT14 into EQT15 before a driver (initiation or completion section) is entered. EQT14 can be preset to *m* by entering (T=) *m* during the EQT entry phase of RTGEN execution (see RTGEN, Section VI), or it can be set or modified on-line with the TO operator request (see Section II).

2. When the driver initiates I/O, and it expects to be entered due to a subsequent interrupt, the driver can set the value *m* into EQT15 just before it exits.

This value *m* can be coded permanently into the driver or else passed to the driver as an I/O call parameter.

NOTE

The system always inserts the contents of EQT14 into EQT15 before entering a driver. However, a time-out value inserted directly into EQT15 by the driver over-rides any value previously set by the system (from EQT14).

Should a device time out, the following events take place:

a. The calling program is rescheduled, and a zero transmission log is returned to it.

b. The device is set to the down status, and bit 11 in the fourth word of the device's EQT entry is set to one. An error message is printed; e.g.,

I/O ERROR TO EQT #3

c. The system issues a CLC to the device's select code(s) through the EQT number located in the interrupt table.

Due to event "c" above, each device interface card requires an entry in the interrupt table during RTGEN. If, for example, the fixed head disc controller I/O cards, which do not interrupt, did not have an entry in the interrupt table, and the disc had timed out, the system would not be able to issue a CLC to the controller I/O cards. Without the CLC, the operator might not be able to reactuate the disc, depending on the particular disc controller, with the UP operator request.

A time-out value of zero is equivalent to not using the time-out feature for that particular device. If a time-out parameter is not entered, its value remains zero and time-out is disabled for the device.

SAMPLE I/O DRIVER

The sample driver on the following pages demonstrates the principles involved in programming an I/O driver for the RTE System. Note that this driver is for tutorial purposes only and not one of the drivers supplied with the RTE System.

```
PAGE 0002 #01  ** RT EXEC DRIVER <70> G.P.R. (OUTPUT) **


0001                    ASMB,R,L
0003  00000                                   NAM DVR70
0004*
0005*
0006                                          ENT I.70,C.70
0007*
0008*
0009*   DRIVER 70 OPERATES UNDER THE CONTROL OF THE
0010* I/O CONTROL MODULE OF THE REAL-TIME EXECUTIVE.
0011* THIS DRIVER IS RESPONSIBLE FOR CONTROLLING
0012* OUTPUT TRANSMISSION TO A 16 BIT EXTERNAL
0013* DEVICE. <70> IS THE EQUIPMENT TYPE CODE ASSIGNED
0014* GENERALLY TO THESE DEVICES.  I.70 IS THE
0015* ENTRY POINT FOR THE *INITIATION* SECTION AND C.70
0016* IS THE *COMPLETION* SECTION ENTRY.
0017*
0018*
0019* - THE INITIATION SECTION IS CALLED FROM I/O
0020*   CONTROL TO INITIALIZE A DEVICE AND INITIATE
0021*   AN OUTPUT OPERATION.
0022*
0023*   I/O CONTROL SETS THE ADDRESS OF EACH WORD OF THE
0024*   15 WORD EQT ENTRY (FOR THE DEVICE) IN THE SYSTEM
0025*   COMMUNICATIONS AREA FOR BOTH INITIATOR AND CONTINUATOR
0026*   SECTIONS.  THE DRIVER REFERENCES TO THE EQT ARE:
0027*   -EQT1 THRU EQT15-
0028*
```

```
0029*   CALLING SEQUENCE:
0030*
0031*        (A) = SELECT CODE OF THE I/O DEVICE.
0032*        P    JSB I.70
0033*        P+1 -RETURN-
0034*
0035*
0036*           (A) = 0, OPERATION INITIATED
0037*           (A) = REJECT CODE
0038*
0039*               1. ILLEGAL REQUEST
0040*               2. ILLEGAL MODE
0041*
0042* -THE COMPLETION SECTION IS CALLED BY CENTRAL
0043*   INTERRUPT CONTROL TO CONTINUE OR COMPLETE
0044*   AN OPERATION.
0045*
0046*     CALLING SEQUENCE:
0047*
0048*        (A) = SELECT CODE OF THE I/O DEVICE.
0049*
0050*        P    JSB C.70
0051*        P+1 -COMPLETION RETURN-
0052*        P+2   CONTINUATION RETURN-
0053*
0054*          (A) = 0, SUCCESSFUL COMPLETION WITH
0055*                (B) = # OF WORDS TRANSMITTED.
0056*          (A) = 2, TRANSMISSION ERROR DETECTED
0057*                (B), SAME AS FOR (A)=0
0058*
0059*          - CONTINUATION RETURN: REGISTERS
0060*             MEANINGLESS
0061*
0062* -RECORD FORMAT-
0063*
0064*    THIS DRIVER PROVIDES A 16 BIT BINARY
0065*     WORD TRANSFER ONLY.

0067*      *** INITIATION SECTION ***
0068*
0069  00000 000000  I.70  NOP            ENTRY FORM IOC
0070*
0071  00001 016071R       JSB SETIO      SET I/O INSTRUCTIONS FOR DEVICE
0072*
0073  00002 161665        LDA EQT6,I     GET CONTROL WORD OF REQUEST,
0074  00003 012105R       AND =B3         ISOLATE.
0075*
0076  00004 052106R       CPA =B1        IF REQUEST IS FOR INPUT
0077  00005 126000R       JMP I.70,I      THEN REJECT.
0078  00006 052107R       CPA =B2        PROCESS FOR WRITE REQUEST
0079  00007 026012R       JMP D.X1        GO TO WRITE REQUEST
0080*
0081* REQUEST ERROR- CAUSE REJECT RETURN TO I/O CONTROL.
0082*
0083  00010 062107R       LDA =B2        SET A=2 FOR ILLEGAL CONTRL REQ.
0084  00011 126000R       JMP I.70,I     -EXIT-
0085*
0086* WRITE REQUEST PROCESSING
0087*
```

```
0088  00012 161666  D.X1  LDA EQT7,I    GET REQUEST BUFFER ADDRESS
0089  00013 171670        STA EQT9,I     AND SET AS CURRENT ADDRESS
0090  00014 161667        LDA EQT8,I     GET BUFFER LENGTH
0091  00015 003004        CMA,INA        SET NEGATIVE AND SAVE
0092  00016 171671        STA EQT10,I    AS CURRENT BUFFER LENGTH.
0093  00017 002002        SZA            CHECK LENGTH
0094  00020 026024R       JMP D.X3       NON-ZERO
0095  00021 062110R       LDA =B4        IMMEDIATE COMPLETION
0096  00022 006400        CLB            SET TLOG IN B-REG
0097  00023 126000R       JMP I.70,I     IF ZERO
0098*
0099* CALL COMPLETION SECTION TO WRITE FIRST WORD.
0100*
0101  00024 062104R D.X3  LDA P2         ADJUST RETURN
0102  00025 072031R       STA C.70       TO INITIATOR SECTION.
0103  00026 026036R       JMP D.X2       GO TO COMPLETION SECTION
0104*
0105  00027 002400  IEXIT CLA            RETURN TO I/O CONTROL WITH
0106  00030 126000R       JMP I.70,I     OPERATION INITIATED.
0107*
0109*
0110*     *** COMPLETION SECTION ***
0111*
0112  00031 000000  C.70  NOP            ENTRY
0113  00032 165660        LDB EQT1,I     SPURIOUS
0114  00033 006003        SZB,RSS        INTERRUPT?
0115  00034 026051R       JMP SPURI      YES - IGNORE
0116  00035 016071R       JSB SETIO      SET I/O INSTRUCTIONS FOR DEVICE
0117*
0118  00036 002400  D.X2  CLA            IF CURRENT BUFFER LENGTH = 0,
0119  00037 151671        CPA EQT10,I    THEN,GO TO
0120  00040 026054R       JMP I.3        STATUS SECTION.
0121*
0122  00041 165670        LDB EQT9,I     GET CURRENT BUFFER ADDRESS
0123  00042 135670        ISZ EQT9,I     ADD 1 FOR NEXT WORD
0124  00043 160001        LDA B,I        GET WORD
0125  00044 135671        ISZ EQT10,I    AND INDEX WORD COUNT
0126  00045 000000        NOP            IGNORE P+1 IF LAST WORD.
0127*
0128*
0129  00046 102600  I.1   OTA 0          OUTPUT WORD TO INTERFACE
0130  00047 103700  I.2   STC 0,C       TURN DEVICE ON
0131  00050 002001        RSS
0132  00051 175774  SPURI STB EQT15,I    ZERO TIME-OUT CLOCK WORD
0133*
0134  00052 036031R       ISZ C.70       ADJUST RETURN TO P+2
0135  00053 126031R       JMP C.70,I     -EXIT-
0136*
```

```
0138*
0139* STATUS AND COMPLETION SECTION.
0140*
0141  00054 102500  I.3   LIA 0          GET STATUS WORD
0142  00055 012111R       AND =B77       STRIP OFF BITS
0143  00056 070001        STA B           AND SAVE IN B
0144  00057 161664        LDA EQT5,I     REMOVE PREVIOUS
0145  00060 012112R       AND =B177400    STATUS BITS
0146  00061 030001        IOR B          SET NEW
0147  00062 171664        STA EQT5,I      STATUS BITS
0148*
0149  00063 002400        CLA            SET NORMAL RETURN COND
0150  00064 056110R       CPB =B4        ERROR STATUS BIT ON?
0151  00065 062107R       LDA =B2        YES, SET ERROR RETURN
0152*
0153  00066 165667        LDB EQT8,I     SET (B) = TRANSMISSION LOG
0154*
0155  00067 106700  I.4   CLC 0          CLEAR DEVICE
0156*
0157  00070 126031R       JMP C.70,I     -EXIT FOR COMPLETION
0158*
0159*
0160* SUBROUTINE <SETIO> CONFIGURES I/O INSTRUCTIONS.
0161*
0162  00071 000000  SETIO NOP
0163  00072 032103R       IOR LIA        COMBINE LIA WITH I/O
0164  00073 072054R       STA I.3        SELECT CODE AND SET.
0165*
0166  00074 042113R       ADA =B100      CONSTRUCT OTA INSTRUCTION
0167  00075 072046R       STA I.1
0168*
0169  00076 042114R       ADA =B1100     CONSTRUCT STC,C INSTRUCTION
0170  00077 072047R       STA I.2
0171*
0172  00100 032115R       IOR =B4000     CONSTRUCT CLC INSTRUCTION
0173  00101 072067R       STA I.4
0174*
0175  00102 126071R       JMP SETIO,I    -RETURN-
0176*
```

```
0178*
0179* CONSTANT AND STORAGE AREA
0180*
0181  00000          A      EQU 0          A-REGISTER
0182  00001          B      EQU 1          B-REGISTER
0183*
0184  00103 102500  LIA    LIA 0
0185  00104 000026R P2     DEF IEXIT-1
0186*
0187*** SYSTEM AND BASE PAGE COMMUNICATIONS AREA ***
0188*
0189  01650          .      EQU 1650B
0190*
0191*   I/O MODULE/DRIVER COMMUNICATION
0192*
0193  01660          EQT1   EQU .+8
0194  01661          EQT2   EQU .+9
0195  01662          EQT3   EQU .+10
0196  01663          EQT4   EQU .+11
0197  01664          EQT5   EQU .+12
0198  01665          EQT6   EQU .+13
0199  01666          EQT7   EQU .+14
0200  01667          EQT8   EQU .+15
0201  01670          EQT9   EQU .+16
0202  01671          EQT10  EQU .+17
0203  01672          EQT11  EQU .+18
0204  01771          EQT12  EQU .+81
0205  01772          EQT13  EQU .+82
0206  01773          EQT14  EQU .+83
0207  01774          EQT15  EQU .+84
0208*
0209*
      00105 000003
      00106 000001
      00107 000002
      00110 000004
      00111 000077
      00112 177400
      00113 000100
      00114 001100
      00115 004000
0210                         END
**  NO ERRORS*
```

PRIVILEGED INTERRUPT PROCESSING

When a special I/O interface card, HP 12620, is included in the system, RTE allows a class of privileged interrupts to be processed independently of regular RTE operation, with a minimal delay in responding to interrupts. The presence and location of the special I/O card is selected at system generation time by RTGEN. Its actual hardware location is stored in the word DUMMY in base page (or if not available, zero). See Section VI for the exact specification procedure.

The special I/O card separates the privileged (high priority, low-channel numbers) interrupts from the regular system-controlled interrupts. When this card is present, RTE does not operate with the interrupt system disabled, but rather, sets control on the special I/O card to hold off lower-priority interrupts. The privileged interrupts are enabled when RTE is running and they can interrupt any RTE operation.

PRIVILEGED INTERRUPTS

The privileged interrupts can be recognized within 100 microseconds and processed in two ways:

1. Through a privileged driver which has in general the structure of a standard I/O driver plus a special privileged interrupt processor routine;
2. Through a special routine embedded in the system area.

Note that the routines which handle privileged interrupts must be completely independent of RTE.

If the first method is used, the calling program can make a standard I/O call to the privileged device. The calling program will be suspended for the time it takes to do the transfer, after which it will be rescheduled. To the calling program, there is no difference between a privileged type device I/O call and a non-privileged (standard) type device I/O call. If the privileged driver is assigned a time-out parameter, the parameter must be long enough to cover the period from I/O initiation to transfer completion.

If the second method is used, a "JSB --, I" in the interrupt location (set by using "ENT, *name*" when configuring the interrupt table) channels the special interrupt directly to the entry point of its associated special routine. CIC and the rest of RTE are not aware of these interrupts. CIC sets a software flag indicating the status of the memory protect facility, MPTFL, in base page.

If MPTFL equals zero, the memory protect is "ON" Any special interrupt routine must issue a "STC 5" instruction immediately before returning to the point of interruption by a "JMP ---, I".

If MPTFL equals one, RTE itself was executing when the privileged interrupt occurred, and memory protect is "OFF". The special routine must not issue the "STC 5" in this case.

SPECIAL PROCESSING BY CIC

During interrupt processing, CIC saves the registers, issues a CLF instruction to the interrupt location, sets the memory protect flag, MPTFL, equal to one, and checks the DUMMY flag. If the DUMMY flag is zero, the hardware interrupt system is left disabled and normal processing continues. If non-zero, the value is used to issue a STC to the I/O location (this assumes that the flag on the special I/O card is set); the STC holds off lower priority interrupts until the control is cleared on the special card.

CIC clears the control flip-flop on each DMA channel to defer DMA completion interrupts, and enables the interrupt system (a zero is in the interrupt location for the special card so that interrupts from the card are ignored).

$IRT, a special subroutine within CIC, resets the flags and DMA channels upon completion of normal system processing. Other modules of RTE use this routine also. $IRT sets MPTFL = 0 to indicate that memory protect is "ON", and also sets the control flip-flop on the active DMA channels if bit 15 of the DMA interrupt entries equals one.

PRIVILEGED INTERRUPT ROUTINES

A privileged interrupt routine, whether embedded directly within the system or within a privileged driver, must save and restore all registers, and restore memory protect to its original state (word MPTFL contains this status). This is because any interrupt automatically disables memory protect. The privileged interrupt routine must not use any features or requests of RTE, nor use either DMA channel. It can communicate with normal user programs by use of the appropriate COMMON region. Flags, parameters, control words, etc., can be set and monitored by either routine in pre-defined locations in COMMON. The starting address of either COMMON region is available in base page. (See Appendix A.) A normal user program can be scheduled to run at periodic time intervals to scan and set indicators in COMMON.

SAMPLE PRIVILEGED DRIVER

The following discussion describes an example privileged driver, generalized to DVRXX, which is controlling a device operating in the privileged mode.

The device transfers one word of data each time it interrupts, and the data is stored into the buffer passed to the driver via the call parameters. Also passed to the driver is the number of data words to be input from the privileged device, this being the length of the data buffer.

The concepts behind such a driver are as follows:

a. It is called by a standard EXEC I/O call.

b. The calling program is placed into I/O suspension.

c. The device's trap cell is changed from "JSB CIC" to "JSB P.XX" where P.XX is the entry point to the privileged routine within the driver.

d. Therefore, each time the device interrupts the RTIOC overhead is circumvented because the privileged routine P.XX is entered directly.

e. After each interrupt, if another data point is still required to satisfy the buffer length, the device is again encoded to subsequently interrupt, and the privileged routine is exited.

f. When the entire data buffer has been filled, the driver needs a way to communicate to the Executive that the transfer is complete. The driver must also wait until it knows that the Executive is not running. This is accomplished by using the time base generator interrupt trap cell (since the TBG is not a privileged device, its interrupt is enabled only when the Executive is not running). The TBG trap cell is changed from "JSB CIC" to "JSB T.XX" where T.XX is the entry point of another routine within the driver. Then, the next time TBG interrupts, the driver is entered with the knowledge that the Executive is not running.

g. When the TBG interrupts, the driver is entered at T.XX, which then means two things have taken place; the I/O transfer has completed (all data has been taken), and the Executive has just stopped running.

h. When T.XX is entered, it sets up to simulate a non-privileged device interrupt entry to RTIOC from the privileged device by doing some of RTIOC's initial overhead. T.XX sets the TBG to interrupt (when the interrupt system is re-enabled for non-privileged devices), and restores the TBG trap cell to "JSB CIC".

i. T.XX then enters RTIOC at a predetermined point with the A-Register set such that, to RTIOC, the privileged device interrupted in a standard non-privileged mode.

j. This causes the Executive to call the completion section (C.XX) of the privileged driver.

k. C.XX returns the transmission log, via the B-Register, and a successful completion indication, via the A-Register, to RTIOC.

l. RTIOC then reschedules the program which called the driver through its normal I/O completion machinery.

m. When the Executive has scheduled a user program and turned the interrupt system back on, the TBG will interrupt immediately. Thus, one TBG interrupt may have been delayed slightly, but none will have been missed entirely.

A standard RTE driver uses the EQT for all its temporary storage so that the same driver can be driving more than one device simultaneously. A privileged driver, however, cannot do this because it can never know the state of pointers to the EQT while it is running since it is running independently of the Executive. The privileged driver keeps its temporary storage internally, and therefore, can control only one device. For each device the driver will control, the driver must be reassembled with all names DVRXX and $JPXX (for this example) changed to another number. Then one driver per device must be loaded into the system at generation time.

Initiation Section, I.XX

Refer to the partial listing of the sample privileged driver following this discussion. A standard I/O call to input from the device causes the calling program to be I/O suspended and the driver to be entered at I.XX. The request code is checked for validity.

Because this driver can control just one device (unlike standard drivers), there is no need to configure it more than once. Therefore, the first time the driver is entered, it is configured and the switch at "FIRST" is cleared so that on all subsequent entries, the configuration code is not executed.

The modification of the device trap cell is performed just once, after the configuring routine, and is not modified again on all later entries into the initiator. The trap cell is altered so that the device interrupts will be channeled to the P.XX subroutine instead of to RTIOC. The "JSB P.XX" instruction and its associated base page link are established via the small program "$JPXX" (see listing). A "JSB T.XX" instruction and its associated link are also established by "$JPXX" for the TBG trap cell modification, which is done later by the P.XX privileged routine on its last execution. Also at this point, the address within RTIOC at which T.XX is to enter is computed.

A counter, which is incremented in routine P.XX, is established for the number of readings to be taken; the buffer address for the storage of the data is saved, and the device is set up to initiate a reading and is encoded. The initiator then exits.

Privileged Section. P.XX

When the device interrupts, P.XX is entered as a result of the device's trap cell modification.

Because entry is made directly into P.XX the routine must do the housekeeping which RTIOC does when entered from an interrupt. Before P.XX can turn the interrupt system back on for higher priority interrupts, it must ensure that the DMA channels cannot interrupt, save the old memory protect status, and set its new status.

P.XX then loads and stores the data in the next unfilled buffer word. If there is yet another data point to be taken, P.XX sets up the device for the next reading, disables the interrupt system, encodes the device, restores memory protect status and its flag, turns the interrupt system back on, and exits. This basically resets the system to its state before P.XX was entered.

When the last reading is taken, P.XX disables the interrupt system, turns off the device, and modifies the TBG trap cell so that T.XX is entered on the next TBG interrupt. Before P.XX exits, it restores memory protect status and its flags, and turns the interrupt system back on.

True Completion Section, T.XX

The status of the device and driver is now unchanged until the TBG interrupts. Thus, the TBG interrupt entry to T.XX is used as a flag to indicate that the privileged I/O transfer has completed. T.XX, like P.XX, does the housekeeping which RTIOC would have done had RTIOC been entered directly. T.XX saves the registers in the same words as RTIOC would have stored them, etc. This is done because T.XX enters RTIOC at the point "$XCIC+1" with an indication that the device apparently has interrupted in non-privileged mode. When this occurs, RTIOC thinks it has been entered directly from an interrupt from the device. Before entering RTIOC, T.XX sets the TBG flag so that it will again interrupt, and resets the TBG trap cell to "JSB CIC".

Logical Completion Section, C.XX

Thus, RTIOC passes control to C.XX which returns a transmission log and a normal completion indication to RTIOC.

RTIOC then goes to its I/O completion section which reschedules the calling program and processes the device queue as if it were a standard (non-privileged) device.

```
ASMB,R,L,T,B
*
*
*  DRIVER WITH PRIVILEGED INTERRUPT
*
*
       NAM DVRXX
       ENT I.XX,P.XX,T.XX,C.XX
       EXT $CIC,$XCIC,$JPXX,$JTXX
*
*
*
*  CALLING SEQUENCE:
*      JSB EXEC       CALL EXEC
*      DEF *+5        RETURN POINT
*      DEF RCODE      REQUEST CODE
*      DEF CONWD      CONTROL WORD
*      DEF BUFFR      ADDRESS OF BUFFER
*      DEF LENTH      LENGTH OF BUFFER
*
*
*
*
*  CAUTION:  THIS DRIVER WILL NOT WORK WITH MORE THAN
*      ONE SUBSYSTEM.  IF MORE THAN ONE SUBSYSTEM
*      EXISTS IN A SYSTEM, BOTH DVRXX AND $JPXX MUST
*      BE RE-ASSEMBLED WITH ALL THE NAMES CONTAINING
*      'XX' CHANGED TO SOME OTHER NUMBER.  THEN ONE
*      DRIVER PER SUBSYSTEM MUST BE PUT INTO THE SYSTEM
*      AT GENERATION TIME.
*
*  INITIATION SECTION
*
I.XX   NOP
       STA SCODE      SAVE SELECT CODE
       LDA EQT6,I     REQUEST CODE TO A
       AND M77
       CPA =B1        READ REQUEST?
       JMP *+3         YES
REJCT  CLA,INA         NO - ERROR
       JMP I.XX,I     REJECT RETURN
FIRST  RSS            CONFIGURE FIRST
       JMP INIT        TIME ONLY
```

```
       LDA SCODE
       IOR LIA          CONFIGURE
       STA IOØ           IO INSTRUCTIONS
       •
       •
       •
       LDA $JPXX        SET TRAP CELL TO
       STA SCODE,I       JSB P.XX
       LDB AXCIC        COMPUTE
       RBL,CLE,SLB,ERB
       LDB B,I          ADDRESS
       SSB
       JMP *-3          XCIC+I
       INB
       STB AXCIC        IN RTIOC
       CLA              SET SO AS NOT TO
       STA FIRST         CONFIGURE AGAIN
*
INIT   LDA EQT8,I       NUMBER OF CONVERSIONS TO A
       CMA,INA,         NEGATE FOR
       STA CVCTR         CONVERSION COUNTER
       SSA,RSS          REJECT IF
       JMP REJCT         NUMBER <=Ø
       LDA EQT7,I       SAVE DATA BUFFER
       STA DAPTR         ADDRESS FOR P.XX
       JSB READ         START A READING
IO1    STC IO,C         ENCODE DEVICE
       JMP I,XX,I       RETURN
*
READ   NOP              ROUTINE CONTAINING
       •                 CONFIGURED IO
       •                 INSTRUCTIONS TO
       •                 SET UP THE DEVICE
       JMP READ,I        TO INITIATE ONE READING
*
*  PRIVILEGED INTERRUPT ROUTINE
*
P.XX   NOP
       CLF Ø            TURN OFF INTERRUPTS
       CLC 6            TURN OFF
       CLC 7             DMA INTERRUPTS
       STA ASV          S
       STB BSV           A
       ERA,ALS            V
       SOC                 E
       INA
       STA EOSV          REGISTERS
       LDA MPTFL        SAVE MEMORY
       STA MPFSV         PROTECT FLAG
       CLA,INA          TURN OFF MEMORY
       STA MPTFL         PROTECT FLAG
       STF Ø            TURN ON INTERRUPTS
*
       •                LOAD IN DATA
       •                 FROM DEVICE
       •                 VIA IO INSTRUCTIONS
```

```
*
        STA DAPTR,I    STORE IN DATA BUFFER
        ISZ CVCTR      LAST CONVERSION
        RSS             NO
        JMP DONE        YES
        ISZ DAPTR      SET UP FOR
        JSB READ        NEXT CONVERSION
        CLF Ø          TURN OFF INTERRUPTS
I04     STC IO,C       ENCODE DEVICE
*
EXIT    LDA MPFSV      WAS MEMORY
        SZA             PROTECT ON?
        JMP EXIT1       NO, FORGET DMA'S
        LDB INTBA      TURN
        LDA B,I         DMA'S
        SSA              BACK
        STC 6             ON
        INB                IF
        LDA B,I             THEY
        SSA                  WERE
        STC 7                 ON
EXIT1   LDA EOSV       RESTORE
        CLO             E AND
        SLA,ELA          O
        STF 1             FLAGS
        LDB BSV        RESTORE B
        LDA MPFSV      RESTORE MEMORY PROTECT
        STA MPTFL       FLAG IN SYSTEM
        SZA            MEMORY PROTECT ON?
        JMP *+5         NO
        LDA ASV         YES, RESTORE A
        STF Ø          TURN ON INTERRUPTS
        STC 5          SET MEMORY PROTECT
        JMP P.XX,I     RETURN
        LDA ASV        RESTORE A
        STF Ø          TURN ON INTERRUPTS
        JMP P.XX,I     RETURN
*
DONE    CLF Ø          TURN OFF INTERRUPTS
I07     CLC IO         TURN OFF DEVICE
        LDA TBG,I      SAVE TIME BASE
        STA TBSAV       TRAP CELL
        LDA $JTXX      CONNECT TBG TRAP CELL
        STA TBG,I       TO T.XX
        JMP EXIT       GO TO EXIT
*
*
*  TIMER INTERRUPT ROUTINE
*
*
T.XX    NOP
        CLF Ø          TURN OFF INTERRUPTS
        STA XA,I       SAVE
        STB XB,I        REGISTERS
        ERA,ALS           IN
        SOC                SYSTEM
        INA                 AREA
        STA XEO,I
        LDA T.XX       STORE RETURN
        STA $CIC        ADDRESS IN CIC
```

```
        LDA TBG         CAUSE
        IOR STF          TIME BASE TO
        STA *+1           INTERRUPT
        STF Ø              AGAIN
        LDA TBSAV       RESTORE TIME BASE
        STA TBG,I        TRAP CELL
        LDA SCODE       SELECT CODE TO A
        JMP AXCIC,I
*
*
*  COMPLETION SECTION
*
*
C.XX  NOP
        CLA,CCE         SET UP FOR NORMAL RETURN
        LDB EQT8,I      TRANSMISSION LOG
        ELB,RBR          TO B
        JMP C.XX,I      RETURN
*
*
*  CONSTANTS AND TEMPORARIES
*
*
SCODE OCT Ø
CVCTR OCT Ø
        .
        .
        .
LIA   LIA Ø
M77   OCT 77
DAPTR DEF Ø
ASV   OCT Ø
BSV   OCT Ø
EOSV  OCT Ø
MPFSV OCT Ø
TBSAV OCT Ø
AXCIC DEF $XCIC
*
*
*  SYSTEM COMMUNICATION AREA
*
.     EQU 1650B
INTBA EQU .+4
EQT6  EQU .+13
EQT7  EQU .+14
EQT8  EQU .+15
TBG   EQU .+20
XA    EQU .+49
XB    EQU .+50
XEO   EQU .+51
MPTFL EQU .+80
        END


ASMB,R,L,B
        NAM $JPXX
        ENT $JPXX,$JTXX
        EXT P.XX,T.XX
$JPXX JSB P.XX
$JTXX JSB T.XX

        END
```

SECTION VI
RTE SYSTEM INSTALLATION

GENERAL INFORMATION

This section is divided into three parts. Part 1 contains directions for planning or laying out an RTE System. General directions are provided for completing the RTE System Configuration Worksheet, and then transferring the plans into instructions to RTGEN. If the user is quite familiar with the RTGEN process, a punched tape containing all the parameter inputs can be punched from the configuration worksheet, and then placed in the tape reader on the system teleprinter. Each time RTGEN requires an input from the operator the punched tape will provide the normal operator response. This method speeds up the system configuration considerably. If the user is not familiar with the RTGEN process, it is recommended that this section be read several times to completely familiarize him with all the steps required in a successful system configuration. It is also recommended that the user read Appendix B and Section I again before attempting to configure a system.

Part 2 provides the instructions required to configure an RTE System based on a fixed head disc drive, and Part 3 provides the instructions required to configure an RTE System based on a moving head disc drive.

NOTE

Part 1 must be read first before going to Part 2 or 3.

In this section the following terms are used:

- Track – A software addressable (logical) track.
- RTE System Tracks – All the disc tracks for which RTE maintains a contiguous track assignment table. These tracks are located on logical unit 2 (system), and 3 (auxiliary).
- Platter – The recording surface where the data is stored (both sides of a physical disc).
- System Disc – The disc assigned to logical unit 2. Also the platter where the absolute binary code of RTE resides.
- Auxiliary Disc – The disc assigned to logical unit 3. (Has same status as LU2.)
- Scratch Area – A number of disc tracks used during system generation for storage of the relocatable binary code of RTE.
- Peripheral Disc – A peripheral disc is available to the RTE user for read/write operations, however, RTE does not manage the tracks nor maintain a track assignment table for them. A peripheral disc must have a logical unit number assignment from seven up.

This page blank. Worksheets on pages 6–3/6–4 are removable – tear along perforation.

INPUT/OUTPUT CONFIGURATION WORKSHEET

RTGEN NUMBER _____ **DATE** _____ **PREPARED BY** _____

SC / SUB	10																															
I/O INTERFACE CARD NAME																																
STD. LOGICAL UNIT NOS.																																
1 SYS. TTY																																
2 SYS. MASS STORAGE																																
3 AUX MASS STORAGE																																
4 PUNCH OUTPUT																																
5 INPUT																																
6 LIST OUTPUT																																
7_{10} TO 63_{10}																																
DVR IDENT. (DVRxx)																																
DMA REQUIRED (D)																																
EQT ENTRY NO.																																
BUFFERED OUTPUT (B)																																
TIME-OUT (T)																																
OCTAL SELECT CODE / SUBCHANNEL	10																															

SYSTEM DISC WORKSHEET

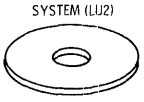
FIXED HEAD SYSTEM DISC

DRUM/DISC MEMORY CHARACTERISTICS

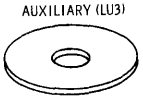
DRUM	DISC	# LOGICAL TRACKS	SECTORS PER TRACK
–	2766	32	128
2773	–	48	
2773-003	2766-002	64	
2774	2766-003	96	
2774-003	2766-004	128	
–	2770	32	90
–	2770-01	64	
–	2771	64	
–	2771-01	128	

DIODES 12606 I/O	# LOGICAL PROT. TRKS	DIODES 12610 I/O
NONE	1	CR1, 2
CR1	2	CR1-3
CR1, 2	4	CR1-4
CR1-3	8	CR1-5
CR1-4	16	CR1-6
CR1-5	32	CR1-7
CR1-6	64	CR1-8
CR1-7	128	CR1-9
–	192	CR1-10

SYSTEM (LU2)



AUXILIARY (LU3)



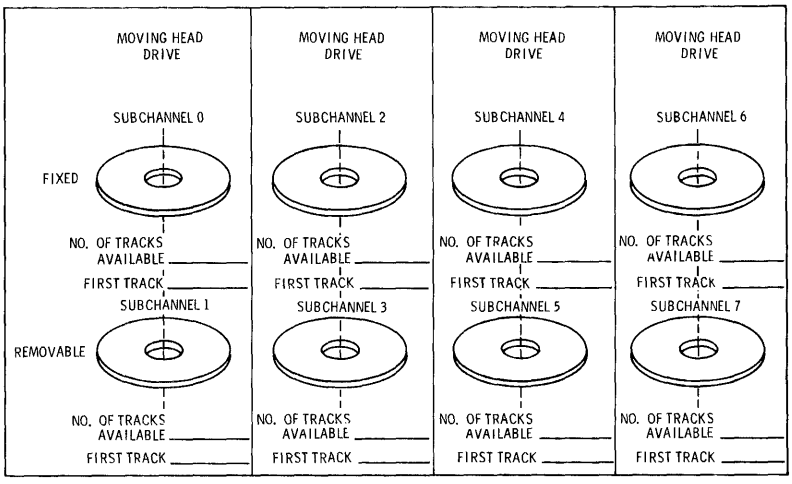
NO. OF TRACKS AVAILABLE _____ NO. OF TRACKS AVAILABLE _____

START SCRATCH _____ SECTORS/TRACK _____

NO. OF PROTECTED TRACKS _____

SECTORS/TRACK _____

MOVING HEAD SYSTEM DISC



SYSTEM SUBCHANNEL NUMBER _____ SCRATCH SUBCHANNEL NUMBER _____

AUXILIARY SUBCHANNEL NUMBER _____ START SCRATCH (I.E., 1ST TRACK - 0) _____

NOTE:

IF A MOVING HEAD DISC DRIVE IS USED AS A PERIPHERAL, THIS PORTION CAN BE USED FOR TRACK ASSIGNMENT. IN THIS CASE, NO SUBCHANNEL ASSIGNMENTS OF THE PERIPHERAL WILL BE ASSIGNED TO LU2 OR LU3.

RTE CONFIGURATION WORKSHEET

INITIALIZATION PHASE

MOVING HEAD INITIALIZATION

MH. DISC CHNL?

TRKS, FIRST TRK ON SUBCHNL:
0?
1?
2?
3?
4?
5?
6?
7?

SYSTEM SUBCHNL?

AUX DISC (YES OR NO)?

AUX DISC SUBCHNL?

SCRATCH SUBCHNL?

START SCRATCH?

128 WORD SECTORS/TRACK?

TBG CHNL?

PRIV. INT. CARD ADDR.

SWAPPING?

LWA MEM?

PRGM INPT?

LIBR INPT?

PRAM INPT?

INITIALIZE SUBCHNL:n

PUNCH BOOT?
YES ,

FIXED HEAD INITIALIZATION

FH. DISC CHNL?

SYS DISC SIZE?

START SCRATCH?

NO. PROTECTED?

SECTORS/TRACK?

AUX DISC SIZE?

SECTORS/TRACK?

TBG CHNL?

PRIV. INT. CARD ADDR?

SWAPPING?

LWA MEM?

PRGM INPT?

LIBR INPT?

PRAM INPT?

PROGRAM INPUT PHASE

EXEC CONTROL (EXEC)
SCHEDULER (SCHED)
I/O CONTROL (RT10C)
I/O DRIVERS
USER'S SYSTEM PROGRAMS
FOREGROUND CORE RESIDENT PROGRAMS
FOREGROUND DISC RESIDENT PROGRAMS
BACKGROUND CORE RESIDENT PROGRAMS
BACKGROUND DISC RESIDENT PROGRAMS
AND THEIR RESPECTIVE SEGMENTS
LIBRARY PROGRAMS
UTILITY PROGRAMS
SUBROUTINES

NO UNDEF EXTS

PARAMETER INPUT PHASE

NAME, TYPE [, PRIORITY] [, EXECUTION INTERVAL]

/E
OF BLANK ID SEGMENTS?

DISC LOADING PHASE

FWA BP LINKAGE

* EQUIPMENT TABLE ENTRY
(1) ,DVR , , , T =
(2) ,DVR , , , T =
(3) ,DVR , , , T =
(4) ,DVR , , , T =
(5) ,DVR , , , T =
(6) ,DVR , , , T =
(7) ,DVR , , , T =
(8) ,DVR , , , T =
(9) ,DVR , , , T =
/E

* DEVICE REFERENCE TABLE
1 = EQT #? (SYSTEM TELEPRINTER)
2 = EQT #? (SYSTEM MASS STORAGE)
3 = EQT #? (AUXILIARY MASS STORAGE)
4 = EQT #? (STANDARD PUNCH UNIT)
5 = EQT #? (STANDARD INPUT UNIT)
6 = EQT #? (STANDARD LIST UNIT)
7 = EQT #?
8 = EQT #?
9 = EQT #?
10 = EQT #?
11 = EQT #?
12 = EQT #?
13 = EQT #?
14 = EQT #?
15 = EQT #?
16 = EQT #?
17 = EQT #?
/E

* INTERRUPT TABLE
/E

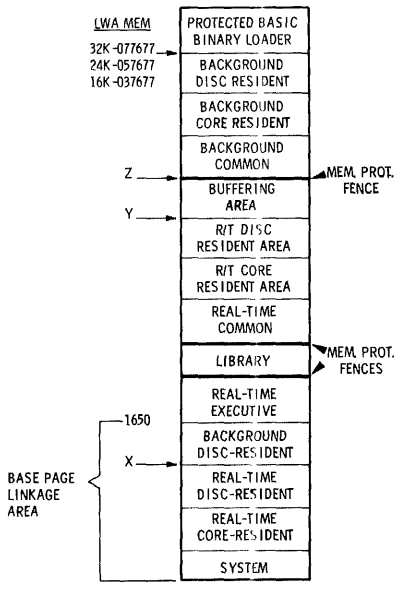
BP LINKAGE
X CHANGE BP LINKAGE?

FWA SY MEM
Y CHANGE FWA AV MEM?

Z BG BOUNDARY?

* SYSTEM STORED ON DISC
TRACK SECTOR (10)

RTE
CORE ALLOCATIONS



PART 1

INSTRUCTIONS FOR PLANNING RTE

INPUT/OUTPUT PLANNING

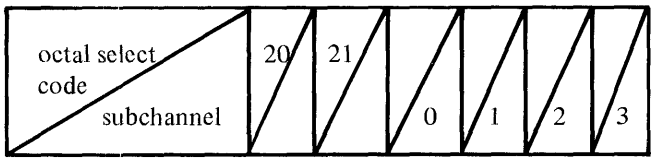
Input/output locations in all HP 2100 series computers have the same sequence of priority addresses: the highest priority address is the lowest numbered select code (I/O location). The octal select codes start at 10_8 and continue up to 67_8, with the 2150 series extender.

Interface cards are assigned to priority addresses according to the speed of interrupt response required by the I/O device. Interface cards for high-speed devices are assigned higher priority addresses than low-speed devices. Devices requiring privileged interrupt are always assigned to the highest priority addresses, while DMA devices are assigned the lowest. The one exception to the DMA rule is in regard to the moving head system disc controller. For the fastest interrupt response, assign moving head disc controller to the next available I/O slots after the Time Base Generator (TBG).

1. Considering the factors given in the preceding paragraphs and the instructions given below, select the priority addresses for each I/O card, and fill in the top portion of the Input/Output Configuration Worksheet table with the I/O card name, and the appropriate select code (I/O slot).

   NOTE

   The top portion of the table is used for either the select code or the subchannel number. For example, if two moving head disc drives (four subchannels) are connected to a controller in select codes 20 and 21, the top portion of the table would be completed thus.



   This method of noting subchannel numbers will facilitate assigning logical unit numbers later in the table.

The following detailed steps show how to assign select codes to devices starting at the highest priority address, select code 10_8. In addition to these steps, make certain that any peripheral devices or subsystems that use multiple I/O slots have their I/O cards together and in the relative order required by that device or subsystem.

a. Assign all devices that require privileged interrupt in order of decreasing interrupt rate (i.e., the transfer speed in interrupts per second).

b. After the privileged devices, assign the privileged interrupt I/O card HP 12620.

c. Assign the TBG I/O card HP 12539.

d. Assign the moving head disc controller I/O cards if a moving head disc is to be configured into the system.

e. Assign all devices that *do not* use DMA in order of decreasing speed.

   NOTE

   There will be occasions when a device uses DMA for data transfer and still generates an interrupt for end-of-record (EOR) processing. In these cases the hardware priority of the device should be treated as a non-DMA device, with the interrupt rate of the EOR condition determining its priority location. Some consideration should be given to the priority of a data transfer vs the priority of a record termination. Data transfers would normally be given priority over EOR interrupts of equivalent or even slightly slower interrupt rates.

f. Assign all devices that *do* use DMA in order of decreasing speed.

g. If an I/O extender is required and the extender does not have DMA capability, the order of steps "e" and "f" can be reversed so that all DMA devices are in the computer mainframe. If this step is necessary, maintain the same relative order of speed assignment among the DMA and non-DMA devices.

2. Make the standard logical unit number (LU) assignments (1 thru 6) to I/O devices by placing an X at the intersection of the standard logical unit number, and the I/O card select code. If a moving head disc drive is used for the system and auxiliary mass storage devices, place an X under one of the subchannels for LU2, and LU3 if applicable. Any remaining subchannels of the moving head disc controller can be assigned logical unit numbers from seven up (i.e., they become peripheral if desired).

INPUT/OUTPUT CONFIGURATION WORKSHEET

RTGEN NUMBER __________ DATE __________ PREPARED BY __________

SC / SUB	10																													
I/O INTERFACE CARD NAME																														
STD. LOGICAL UNIT NOS.																														
1 SYS. TTY																														
2 SYS. MASS STORAGE																														
3 AUX MASS STORAGE																														
4 PUNCH OUTPUT																														
5 INPUT																														
6 LIST OUTPUT																														
7_{10} TO 63_{10}																														
DVR IDENT. (DVRxx)																														
DMA REQUIRED (D)																														
EQT ENTRY NO.																														
BUFFERED OUTPUT (B)																														
TIME-OUT (T)																														
OCTAL SELECT CODE / SUBCHANNEL	10																													

SYSTEM DISC WORKSHEET

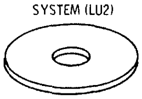
FIXED HEAD SYSTEM DISC

DRUM/DISC MEMORY CHARACTERISTICS

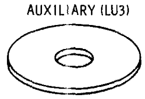
DRUM	DISC	# LOGICAL TRACKS	SECTORS PER TRACK
–	2766	32	128
2773	–	48	
2773-003	2766-002	64	
2774	2766-003	96	
2774-003	2766-004	128	
–	2770	32	90
–	2770-01	64	
–	2771	64	
–	2771-01	128	

DIODES 12606 I/O	# LOGICAL PROT. TRKS	DIODES 12610 I/O
NONE	1	CR1, 2
CR1	2	CR1-3
CR1, 2	4	CR1-4
CR1-3	8	CR1-5
CR1-4	16	CR1-6
CR1-5	32	CR1-7
CR1-6	64	CR1-8
CR1-7	128	CR1-9
–	192	CR1-10

SYSTEM (LU2)



AUXILIARY (LU3)



NO. OF TRACKS AVAILABLE ________

START SCRATCH ________

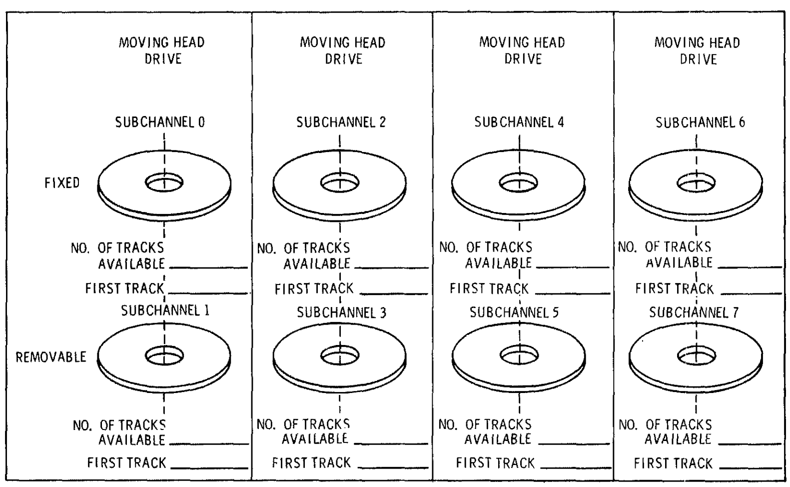
NO. OF PROTECTED TRACKS ________

SECTORS/TRACK ________

NO. OF TRACKS AVAILABLE ________

SECTORS/TRACK ________

MOVING HEAD SYSTEM DISC



SYSTEM SUBCHANNEL NUMBER ________

AUXILIARY SUBCHANNEL NUMBER ________

SCRATCH SUBCHANNEL NUMBER ________

START SCRATCH (I. E., 1ST TRACK = 0) ________

NOTE:

IF A MOVING HEAD DISC DRIVE IS USED AS A PERIPHERAL, THIS PORTION CAN BE USED FOR TRACK ASSIGNMENT. IN THIS CASE, NO SUBCHANNEL ASSIGNMENTS OF THE PERIPHERAL WILL BE ASSIGNED TO LU2 OR LU3.

RTE CONFIGURATION WORKSHEET

INITIALIZATION PHASE

MOVING HEAD INITIALIZATION

MH. DISC CHNL?

TRKS, FIRST TRK ON SUBCHNL:
0?
1?
2?
3?
4?
5?
6?
7?

SYSTEM SUBCHNL?

AUX DISC (YES OR NO)?

AUX DISC SUBCHNL?

SCRATCH SUBCHNL?

START SCRATCH?

128 WORD SECTORS/TRACK?

TBG CHNL?

PRIV. INT. CARD ADDR.

SWAPPING?

LWA MEM?

PRGM INPT?

LIBR INPT?

PRAM INPT?

INITIALIZE SUBCHNL:n

PUNCH BOOT?
YES

FIXED HEAD INITIALIZATION

FH. DISC CHNL?

SYS DISC SIZE?

START SCRATCH?

NO. PROTECTED?

SECTORS/TRACK?

AUX DISC SIZE?

SECTORS/TRACK?

TBG CHNL?

PRIV. INT. CARD ADDR?

SWAPPING?

LWA MEM?

PRGM INPT?

LIBR INPT?

PRAM INPT?

PROGRAM INPUT PHASE

EXEC CONTROL (EXEC)
SCHEDULER (SCHED)
I/O CONTROL (RT10C)
I/O DRIVERS
USER'S SYSTEM PROGRAMS
FOREGROUND CORE RESIDENT PROGRAMS
FOREGROUND DISC RESIDENT PROGRAMS
BACKGROUND CORE RESIDENT PROGRAMS
BACKGROUND DISC RESIDENT PROGRAMS
AND THEIR RESPECTIVE SEGMENTS
LIBRARY PROGRAMS
UTILITY PROGRAMS
SUBROUTINES

NO UNDEF EXTS

PARAMETER INPUT PHASE

NAME, TYPE [, PRIORITY] [, EXECUTION INTERVAL]

/E
OF BLANK ID SEGMENTS?

DISC LOADING PHASE

FWA BP LINKAGE

* EQUIPMENT TABLE ENTRY
(1) ,DVR , , , T=
(2) ,DVR , , , T=
(3) ,DVR , , , T=
(4) ,DVR , , , T=
(5) ,DVR , , , T=
(6) ,DVR , , , T=
(7) ,DVR , , , T=
(8) ,DVR , , , T=
(9) ,DVR , , , T=
/E

* DEVICE REFERENCE TABLE
1 = EQT #? (SYSTEM TELEPRINTER)
2 = EQT #? (SYSTEM MASS STORAGE)
3 = EQT #? (AUXILIARY MASS STORAGE)
4 = EQT #? (STANDARD PUNCH UNIT)
5 = EQT #? (STANDARD INPUT UNIT)
6 = EQT #? (STANDARD LIST UNIT)
7 = EQT #?
8 = EQT #?
9 = EQT #?
10 = EQT #?
11 = EQT #?
12 = EQT #?
13 = EQT #?
14 = EQT #?
15 = EQT #?
16 = EQT #?
17 = EQT #?
/E

* INTERRUPT TABLE
, ,
, ,
, ,
, ,
, ,
, ,
, ,
, ,
, ,
, ,
, ,
, ,
, ,
, ,
, ,
, ,
, ,
, ,
, ,
, ,
/E

BP LINKAGE
X CHANGE BP LINKAGE?

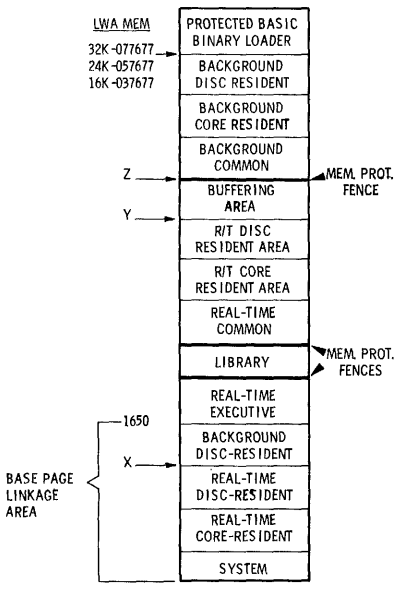
FWA SY MEM
Y CHANGE FWA AV MEM?

Z BG BOUNDARY?

* SYSTEM STORED ON DISC

TRACK SECTOR (10)

RTE
CORE ALLOCATIONS



3. Starting with decimal 7, write in the logical unit numbers sequentially for each device or subchannel number as applicable. These numbers can be arbitrarily assigned to I/O devices, and do not have to be written in a left to right order on this table. Order is attained when RTGEN prints * DEVICE REFERENCE TABLE.

NOTE

If a device has two I/O cards use only the highest priority I/O card for steps 2 and 3.

4. Write in the driver identification number, obtained from the software box, for each device; e.g., teleprinter driver is DVR00. If the moving head disc drive is used, in addition to placing DVR31 under the high-priority card, place a large "1" under the low-priority card. For other devices or subsystems that have more than one I/O card, refer to the I/O card or subsystem documentation covering that device and driver. Place an "1" under the select code number of all I/O cards (i.e., every I/O card must have an entry in the interrupt tables). Place a dash under subchannel numbers.
5. Write in a large "D" for DMA required under each device that will use DMA.
6. Starting with decimal 1, write in the Equipment Table Entry (EQT) numbers sequentially for each device. The system disc should be number 1 to permit special priority assignment to an available DMA channel. Other DMA devices should then be assigned EQT numbers in order of their DMA priority. A device that has subchannels is assigned the same EQT number for each subchannel.
7. Write in a large "B" for devices that will use output buffering (line printer cannot be buffered).
8. Write in a large "T" for devices that will use the time-out parameter. Values will be assigned later on the configuration worksheet, under disc loading phase.

SYSTEM DISC PLANNING

The next step in planning the RTE System is to decide how many tracks of each platter are to be assigned to RTE, and where and how large the scratch area will be. The decisions concerning track assignments for the moving head disc drive as opposed to the fixed head disc drive are somewhat different. However, the disposition of the tracks is the same. The tracks in a fixed head and moving head system can fall into two categories: (1) those assigned to the system (LU2) and auxiliary (LU3) for which RTE maintains a track assignment table, and (2) all other tracks designated under control of RTE, but for which a track assignment table is not maintained by RTE. The tracks in category 1 fall under the definition of RTE System tracks, and the tracks in category 2 fall under the definition of peripheral tracks.

FIXED HEAD SYSTEM DISC

Unless otherwise stated, the following discussion deals with logical disc tracks. The number of tracks available on the system and auxiliary discs (LU2 and LU3) depends on two factors, model number of the disc drive, and if any of the disc tracks are already being used. Determine the number of available tracks by referring to the table on the Fixed Head portion of the System Disc Worksheets, and then subtracting any tracks allocated to another software system.

The number of hardware protected tracks is selected by removing diodes on the HP 12606 or 12610 Disc/Drum Controller cards. Refer to the individual I/O card manuals to determine if the diodes apply to logical or physical tracks. (Four physical tracks = one logical track.) The number of software protected tracks is selected by the user and entered through the teleprinter during RTGEN. The average RTE system requires approximately six to eight tracks for the absolute code (see the sample RTGEN in Appendix C). Compare the RTGEN listing in Appendix C to the list of software intended for this generation to estimate the approximate number of tracks to protect. For example, if the system will use more than eight logical tracks, but less than 16, the user has the option of hardware protecting 16 tracks, or hardware protecting only eight, but software protecting a greater number. Always remember, tracks for the absolute code of other software operating systems should also be protected and contiguous with the RTE tracks within the hardware protected area. Enter the number on the worksheet.

CAUTION

Never hardware protect more tracks than will be declared as software protected.

Determine the number of 64 word sectors per track from the table on the worksheet and fill in the blanks on the worksheet.

Fixed Head Scratch Area

The scratch area is the tracks on the system platter (LU2) into which the relocatable software modules will first be loaded by RTGEN during the program input phase. Two factors must be considered in selecting the size of the area. (1) The area must be large enough to accommodate all the relocatable modules, otherwise an ERR17 will occur, and (2) the area cannot be so large that the system area will prematurely overflow into it (ERR38) during the disc loading phase. The ERR38 can occur when the absolute system is started on track 0 (by RTGEN) and built upwards, toward, and into the relocatable modules overlaying the latter before they have been converted into absolute code. Therefore, the scratch area must be at a point high enough to prevent overlaying the relocatable modules before they are used. The recommended entry for start scratch is zero (0). This entry causes RTGEN to start the scratch area at the midpoint of the available disc area. If either of the two error codes (ERR17 or ERR38) are experienced during RTGEN, use the data in Table 6-1 as a guide in adjusting the scratch area. A rule-of-thumb formula

for determining the approximate number of 64 word sectors a user written program will occupy is as follows:

$$\#\text{ of 64 word sectors} = \frac{x}{33}$$

where

x = number of words of core the program occupies

NOTE

Table 6-1 and the formula are only approximate guides to be used as an aid if starting scratch at midpoint does not work or if difficulty is experienced in estimating some other starting point for scratch.

If a moving head disc drive is used as a peripheral or auxiliary disc in a fixed head RTE System, use the moving head portion of the System Disc Worksheet to allocate disc tracks. To initialize the moving head disc tracks, use the HP diagnostic program tape (HP 13210) prior to generation. To allocate the tracks (generate track map), the number of tracks to be assigned is entered using the $TB31 program (see Appendix B). Also, if a moving head disc is used as an auxiliary disc in a fixed head system, the number of sectors per track question relates to the number of 64-word sectors per track.

MOVING HEAD SYSTEM DISC

The number of tracks available on each platter that can be assigned to RTE ranges from zero to the maximum available on the disc. If zero tracks on a platter is designated, RTE and RTGEN will not read or write data on that platter. The only limit to designating tracks to RTE is that the tracks be contiguous.

NOTE

More than one RTE System can be located on a platter, and tracks can be shared by more than one RTE system. In designating tracks, those that are shared would be included and declared during each RTE generation. The restriction is that any tracks of an RTE System on a platter that are assigned to LU2 or LU3 must be unique to that system. Remaining tracks on the platter can be assigned to other RTE's.

Determine the number of tracks available and starting track number for each subchannel, and fill in the blanks on the worksheet.

Determine which subchannel will be the system disc and which subchannel the auxiliary disc. Note that the moving head system generator (RTGEN) will not allow a fixed head auxiliary disc in a moving head system. Also, the moving head auxiliary disc assignment is optional. Fill in the blanks on the worksheet.

Moving Head Scratch Area

Scratch is the area required for the relocatable modules used to build the system and can be any one of the subchannels assigned to the system. Two factors must be considered in selecting the size of the area. (1) The area must be large enough to accommodate all the relocatable modules, otherwise, an ERR17 will occur, and (2) the area cannot be so large that the system area will prematurely overflow into it (ERR38) during the disc loading phase. The ERR38 can occur when the scratch area is located on the system disc, and the absolute system is built upwards, toward, and into the relocatable modules overlaying the latter before they have been converted into absolute code. Due to this possibility, it is recommended that the scratch area not be located on the system subchannel to prevent overlaying the relocatable modules before they are used.

Determine which subchannel will be the scratch platter. If the scratch area is located on the system subchannel, it is recommended that the entry for start scratch is zero (0). This entry causes RTGEN to start the scratch area at the midpoint of the available disc area. (Note that this default occurs only when scratch is located on the system subchannel.) If either of the two error codes (ERR17 or ERR38) are experienced during RTGEN, use the data in Table 6-1 as a guide in adjusting the scratch area. A rule of thumb formula for determining the approximate number of 64 word sectors a user written program will occupy is as follows:

$$\#\text{ of 64 word sectors} = \frac{x}{33}$$

where

x = number of words of core the program occupies

NOTE

Table 6-1 and the formula are only approximate guides to be used as an aid if starting scratch at midpoint does not work or if difficulty is experienced in estimating some other starting point for scratch.

If the scratch area is assigned to a subchannel other than the system subchannel, that subchannel should not have tracks shared with another system, or any data on it that must be retained. This is because the scratch subchannel tracks assigned to the system being generated are initialized by RTGEN. As a result, any data on them is destroyed.

The first logical track number of the scratch area is always zero (0) regardless of the actual track address. For example, if the scratch is located on a platter with tracks 100 to 200 available, the starting track for scratch would be track zero (0). To start the scratch on a track inside the available area count the number of tracks into the area and use that number as the starting track, (e.g., to skip the first 10 tracks, start scratch on track number 10).

Table 6-1. Approximate Number of 64-Word Sectors Required to Store RTE in Relocatable Format

Name	64-Word Sectors
Executive Software	158
RTE ASMB	174
RTE FORTRAN	348
RTE/DOS FORTRAN IV Compiler	464
RTE/DOS FORTRAN IV 10K Compiler	354
RTE ALGOL	176
RTE Editor	38
RTE Loader	94
RTE/DOS Relocatable Library	240
RTE/DOS FORTRAN IV Library	290
RTE/DOS HP FORTRAN Formatter	42
RTE/DOS Plotter Library	78
Drivers	Allow 11 sectors per driver

RTE CONFIGURATION WORKSHEET

Once the system is planned, the actual configuration process takes place. This process is divided into four phases and must be executed in the order presented on the RTE Configuration Worksheet. The worksheet should be completed in advance from the information entered on the I/O and Disc Worksheets so your replies are known in advance. The following general instructions will provide aid in completing the configuration worksheet. Actual operating instructions for RTGEN (sample configuration) are found in Part 2 or 3 of this section.

INITIALIZATION PHASE

RTGEN requires specifications of all disc tracks belonging to RTE, memory, time base generator channel, swapping option, and program input devices.

MOVING HEAD INITIALIZATION

1. Write in the lower numbered/highest priority select code for the disc controller.
2. Fill in the track assignments for each subchannel. A zero (0) entered for number of tracks causes RTGEN to ignore that subchannel. A"/E" terminates these entries.
3. Fill in all blanks down to the TBG channel address. If only one subchannel is assigned to RTE, place a dash (i.e., does not apply) in system subchannel, auxiliary disc, auxiliary disc subchannel, and scratch subchannel blanks. The number of 128 word sectors per track is 48 for the HP 7900A. This is the number of logical sectors per track and is two times the number of sectors on one side of the platter.
4. Fill in the select code of the TBG card. Fill in the select code of the privileged interrupt card. If there is none, enter a zero (0).
5. The swapping option allows real-time disc-resident programs to swap in and out of core according to priority. If swapping is not allowed, the programs will run to completion. Answer YES or NO.
6. Last word of available memory depends on computer memory size.

   16K - 37677

   24K - 57677

   32K - 77677
7. The use of input units is interchangeable. Maximum versatility can be achieved by designating one device for paper tape and one for magnetic tape if magnetic tape is present on the system. If programs are prepared in advance of system generation, to be stored on a mass storage device (magnetic tape or fixed head disc file), they must contain a specified program type parameter. The program input and library input devices can be:

   PT - paper tape (photo reader)

   TY - teleprinter

   MT - magnetic tape

   DF - fixed head disc file

   The parameter input device is either PT or TY.
8. All disc tracks belonging to RTE may or may not be initialized. RTGEN automatically initializes tracks on the system, auxiliary, and scratch subchannels. If other subchannels are available RTGEN asks if they are to be initialized by subchannel number in ascending order. If the answer is YES, RTGEN initializes only the RTE tracks on that subchannel and reports any defective tracks. If the answer is NO, RTGEN does not initialize that subchannel.

NOTE

Any tracks on a subchannel that are shared with other systems should not be initialized because that data will be destroyed.

9. The punch boot question requires a minimum of one YES answer. Write in YES as many times as bootstrap tapes you desire. Write in NO for termination.

Proceed to the Program Input phase.

FIXED HEAD INITIALIZATION

1. Fill in all blanks down to the privileged interrupt card address. If an auxiliary disc is assigned to RTE, enter the disc size and the number of 64-word sectors per track. (Note that if the auxiliary disc is an HP 7900 the answer must still be the number of 64 word sectors per track.) If there is no auxiliary disc, write a zero (0) in the auxiliary disc size blank, and a dash (i.e., does not apply) in the # sectors/track blank.
2. Fill in the select code of the privileged interrupt card. If there is none, enter a zero (0).
3. The swapping option allows real-time disc-resident programs to swap in and out of core according to priority. If swapping is not allowed, the programs will run to completion. Answer YES or NO.
4. Last word of available memory depends on computer memory size.

   16K - 37677

   24K - 57677

   32K - 77677
5. The use of input units is interchangeable. Maximum versatility can be achieved by designating one device for paper tape and one for magnetic tape if magnetic tape is present on the system. If programs are prepared in advance of system generation, to be stored on a mass storage device (magnetic tape or fixed head disc file), they must contain a specified program type parameter. The program input and library input devices can be:

   PT - paper tape (photo reader)

   TY - teleprinter

   MT - magnetic tape

   DF - fixed head disc file

   The parameter input device is either PT or TY.

PROGRAM INPUT PHASE

Due to the large number of tapes to be loaded during this phase, it is recommended that they be placed on a table in the recommended order of loading.

PARAMETER INPUT PHASE

During the parameter input phase, the operator can modify the type, priority, or execution intervals of any of the programs entered during the program input phase (except that the program type code of background main programs and their segments cannot be changed without losing their relationship to each other).

RTGEN has an additional feature that applies only to type 1, 2, 3, or 4 programs. During the Parameter Input Phase one program of this type can be scheduled to execute automatically whenever the RTE System is loaded from the system disc. This is accomplished by adding 80 to the program's type code. For example, if PROG is originally a type 2 program (real-time disc-resident), it can be changed to:

PROG, 82 [,priority] [,execution interval]

This will cause PROG to be scheduled automatically each time the system is loaded into core from the disc. Only one program can be assigned a type 80 code for automatic scheduling. If more than one program is assigned an 81, 82, 83, or 84 type code, only the last one entered in this phase is automatically scheduled.

Each parameter record is of this general form:

name, type [,*priority*] [,*execution interval*]

Where

name is the name of the program,

type is the program type code:

1 - real-time core-resident

2 - real-time disc-resident

3 - background disc-resident

4 - background core-resident

5 - background segment

6 - re-entrant or privileged library

7 - utility

8 - if program is a main, it is deleted from the system

-or-

8 - if program is a subroutine, then it is used to satisfy any external references during generation. However, it is not loaded in the relocatable library area of the disc.

Where

priority is the program priority from 1 to 99, with 1 the highest.

execution interval is a list of six parameters specifying the times the program should be scheduled for execution, once it is turned on. The first two values specify the execution interval, and the last four specify an initial absolute starting time:

resolution code (0 to 4):

0 – no execution interval

1 – tens of milliseconds

2 – seconds

3 – minutes

4 – hours

execution multiple (0 to 999); the resolution code gives the units for the execution multiple.

initial absolute starting time (four values):

hours (0 - 23)

minutes (0 - 59)

seconds (0 - 59)

tens of milliseconds (0 - 99)

Fill in the blanks for any programs that are to be modified.

The next blank concerns blank ID segments. One blank ID segment is required for each program that will be loaded permanently into the system on-line by the RTE relocating loader. If five segments are allocated, then only five additional programs can be loaded into the system on-line. If a program is deleted from the system by an OF, *name*, 8 operator command, the program's ID segment is returned to the system to use for another on-line load. Each ID segment occupies 29 words in the real-time core-resident area. Fill in the number of blank ID segments required. (Note: 0 is changed to 1 to allow loading at least one program.)

DISC LOADING PHASE

The Disc Loading Phase is the final phase that generates all the required tables, and converts the relocatable programs into absolute binary code. The first word available for base page linkages is established from the I/O Configuration Worksheet as the last used select code plus one (e.g., if the last I/O card in the priority string is located in select code 26, the FWA BP Linkage would be 27). Determine the number from the worksheet and fill in the blank.

EQUIPMENT TABLE ENTRY

The first table to be completed is the Equipment Table. Each entry is located on the I/O worksheet and is to be transferred into this table. The numbers in parentheses are the EQT entry numbers. EQT number one should be the system disc. The DVR number is either DVR30 for the fixed head or DVR31 for the moving head. The remaining blanks are for D (DMA required), B (buffered output), and T (time-out). If T is specified, a value for T must be entered in the T = blank. The value must be a positive decimal number of up to four digits. This is then the number of time base generator interrupts (10 msec intervals) between the time I/O is initiated on the device and the time after which the device should have interrupted. (Note that for privileged drivers T must be long enough to cover the period from I/O initiation to transfer completion.) If the device has not interrupted by this time, it is considered to have timed-out and is set-down, except in the case of the system teletype. For a device controlled by driver DVR00 (e.g., teleprinter), or DVR05 (DVR05 reserved for future system control device), T should not be less than 500.

Example: (1) 15, DVR30, D, __, ___ T = ___
(2) 17, DVR02, B, T, ___ T = 40

This tells RTGEN that EQT entry number one relates to select code 15 and uses DVR30 with DMA required. EQT entry number two relates select code 17 and uses DVR02 (tape punch) with output buffering and time-out after 0.4 second.

Refer to the I/O worksheet and write in the octal select code number, DVR number, and D B and T options if applicable, for each EQT number in sequential order. Note that the driver's identifying suffix letter is not included.

DEVICE REFERENCE TABLE

The Device Reference Table, which contains the logical unit (LU) numbers, is cross referenced to the EQT entries here. Refer to the I/O worksheet to obtain the EQT entry number, LU number, and subchannel number if applicable. The EQT entry number is the number in parenthesis located in Equipment Table Entry above. This should be the same number from the I/O worksheet.

INTERRUPT TABLE

The interrupt links are tied to the select codes for input/output processing. Each I/O card, (select code) in ascending order, is given an interrupt option. The format is:

select code, option, entry

Select code is entered from the I/O worksheet in ascending order.

Option directs RTE in handling the interrupt; there are four options:

EQT,*n2* - relates channel to EQT entry n2.

PRG,*name* - causes program name to be scheduled upon interrupt.

ENT,*entry* - causes control to transfer to the entry point of a user-written system program upon interrupt.

ABS,*xxxxxx* - places an absolute octal value (instruction code) in the interrupt location (may be NOP, CLC, etc.)

The moving head disc controller I/O cards both require an interrupt link to their EQT entry number. Reference the

select code numbers to the DVR31 EQT entry number as shown below.

13, EQT, 1
14, EQT, 1

For other devices or subsystems that have more than one I/O card, refer to the I/O card or subsystem documentation covering the device and driver. In all cases, each I/O card must have an interrupt entry. Note that interrupt location 4 (power fail) may be changed from its present HLT 4 to an ENT entry if a power-fail routine is to be included in the system. For example:

4, ENT, POWER

where POWER is an entry point in a user-supplied power-fail routine.

This ends the planning phase of RTGEN. The final three questions (X, Y, and Z) cannot be effectively answered until RTGEN loads the system and reports the actual boundaries involved.

Boundary X concerns base page linkages. After the real-time disc-resident programs are loaded, RTGEN reports the next available base page link above the links used. The linkage area for real-time disc-resident programs is initially established by the program loaded that requires the most links. If programs requiring more links are to be loaded on-line by the RTE relocating loader, this area will have to be expanded by moving the boundary up (it cannot be moved down). However, enough links must be reserved in the background disc-resident area for the background programs yet to be loaded by RTGEN.

A recommended boundary address of 1100_8 will usually optimize the system if it is to include the usual background programs ASMB, FTN, FTN4, LOADR, EDIT, etc. These programs usually require approximately 550_8 links. The ideal boundary is one which allows RTGEN to allocate as near to 1650 links as possible. For example, if boundary X is established at 1100_8, and, after loading the background disc resident programs RTGEN reported the next base page linkage available as 1651, then the linkage area is as optimized as possible.

CAUTION

If RTGEN reports that more than 1650_8 links were used, the generation is VOID, and must be restarted at the beginning.

Boundary Y defines the real-time disc-resident area for on-line loading with the RTE relocating loader. This area is initially established by the largest program loaded into that area. If a larger program will be loaded into the system on-line, this area must be made larger. This boundary also affects the area reserved for re-entrant processing, buffered transfers, and for background; that is, the more area given to foreground, the less there is available for these other areas.

Boundary Z establishes the background area used for disc-resident, and core-resident programs, plus the common area that is used by both. A recommended procedure for determining the boundaries of Y and Z is as follows:

1. Calculate the area needed for the largest background disc-resident program that will be used.
2. Add to this area needed for the background core-resident programs, and the background common area.
3. Subtract this area from the last word of available memory (LWA MEM)

   Example: $37677 - (11677)_8 = 26000$

   37677 = LWAM in 16K
   11677 = A + B + C
   A = size of largest disc-resident
   B = total core resident size
   C = common size

   This is boundary Z (BG BOUNDARY).
4. Subtract the area required for buffering. The amount recommended for buffering is 400 words.

   Example: $26000 - 400_8 = 25400$

   This is boundary Y (response to CHANGE FWA SYS AV MEM).

Once the system is generated and the tracks used are reported, write this information on the configuration sheet for future reference. Note that RTGEN reports this information in decimal.

PREPARE TAPE SYSTEM

Using the Prepare Tape System as described below, the user can create a magnetic tape or fixed head disc file of the relocatable program modules used during RTGEN.

The Prepare Tape System (PTS) is a program for creating files of relocatable software modules on fixed head disc, drum, or magnetic tape units. RTGEN can then use these files to generate a configured RTE system. When using PTS to generate a file for use during RTGEN execution, follow the instructions given in the Prepare Tape System Manual (02116-91751). The instructions given in the manual for DSGEN apply directly to RTGEN.

In Section III of the PTS Manual under Operating Instructions, substitute the following information under step 1.

1. Gather all the relocatable system and user program tapes. The suggested order for loading onto the mass storage device is as follows:

   Exec Control (EXEC)

   Scheduler (SCHED)

   I/O Control (RTIOC)

I/O Drivers

System programs written by the user

Foreground core-resident programs

Foreground disc-resident programs

Background core-resident programs

Assembler (main and its segments)

FORTRAN (main and its segments) and/or

FORTRAN IV (main and its segments but not both FORTRAN IV versions)

Relocating loaders

Editor

Other background disc-resident programs and their respective segments, if any.

Library Programs

Utility Programs

NOTE

Some of the above relocatable modules may not be present in some configurations.

PART 2
FIXED HEAD RTE SYSTEM GENERATION

The set-up and operation of a Fixed Head RTE System involves two essential steps and one optional step: the RTE System must be configured using the Real-Time System Generator, RTGEN (29015-60001): it must be initiated from the disc by the core-resident basic binary disc loader, (BBDL); and it may be dumped onto tape using the System Dump, SDUMP, as protection against a disc failure.

This part describes the three routines, RTGEN, BBDL, and SDUMP, that are responsible for these processes.

FIXED HEAD DISC RTGEN

RTGEN operates on the same minimum configuration as that required for an RTE System, and configures the system to fit a particular user's core memory size, I/O equipment, and programming needs.

To accomplish this, RTGEN requests certain information from the user; then it accepts the relocatable program modules to be included in the system, determines where they belong in the core, relocates them into absolute format, and stores them on the disc. RTGEN also creates I/O tables by identifying each I/O device and its associated driver routine, and establishing procedures for interrupt processing on each channel. All of this information is pre-planned and is located on the Input/Output and Configuration Worksheets filled out in Part 1. The answers used in this part are from an example RTE System that has been configured on worksheets per the instructions in Part 1. The completed worksheets and RTGEN listing are located in Appendix C.

RTGEN is an absolute program, loaded into core by the BBDL from paper tape. Since RTGEN is independent of the RTE System which it generates, the I/O operations of RTGEN require SIO Drivers for the devices used in generation.

OPERATING PROCEDURES

The operation of RTGEN involves four phases:

a. INITIALIZATION PHASE. RTGEN requests specifications for the RTE System, including description of available fixed head disc space, memory, time base generator channel, swapping option, and program input devices.

b. PROGRAM INPUT PHASE. The operator loads the relocatable programs provided with the system and created by the user.

c. PARAMETER INPUT PHASE. Parameters describing or changing the type, priority, and execution interval of each program may be entered. (Although this information may already be included in the program's NAM record, it can be changed at this point.) RTGEN requests the number of blank ID segments to be reserved for subsequent program addition.

d. DISC LOADING PHASE. RTGEN requests a specification of the base page linkage, and begins loading programs onto the disc. The modules of RTE, drivers, and system programs are loaded first, after which RTGEN then requests information for the equipment table, device reference table, and interrupt table and proceeds to load the rest of the programs onto the disc.

NOTE

During any one of the phases, RTGEN can restart that phase at starting address 100_8. Note, the switch register must be cleared before pressing RUN.

To execute RTGEN and configure an RTE System, follow these steps:

a. Turn on all equipment; set the system teleprinter to LINE, and disable the disc protect switch on the interface card. (See Drum Memory Interface Kit Manual, or Disc Memory Interface Kit Manual.)

b. Load the RTGEN tape and configured SIO Drivers into core using the BBDL.

c. If a fixed head disc is to be used to store the relocatable modules, its SIO driver must be configured as described in the PTS manual on the SIO Disc/Drum Driver Configuration Flowchart.

d. Go to the starting address of the program, 100_8, clear the switch register, and push RUN. RTGEN begins the initialization phase.

INITIALIZATION PHASE

During the initialization phase, RTGEN requests information necessary to begin generating the RTE System. After each question is printed, the operator responds with the required answer, followed by carriage return/line-feed (CR/LF). Operator responses are shaded and are only examples; actual responses should be appropriate to the particular system being generated. If an error is made and discovered before LF is entered, type the RUB OUT key then CR/LF. Otherwise restart at step "d" above.

RTGEN requests the higher priority select code (octal) of the system disc or drum unit controller

. . . . . FH DISC CHNL?

Operator responds

. 23

RTGEN requests the number of tracks (decimal) on the system disc or drum. (If a relocatable file exists on the unit, subtract these tracks from the size.)

. . . . . SYS DISC SIZE?

Operator responds

. 32

RTGEN requests the track number starting the disc scratch area. This is the area into which the relocatable software modules will be first loaded by RTGEN during the program input phase. If there is a large quantity of relocatable modules the scratch area needs to be large enough to accommodate them all; however, there is also a limit to how small the number can be. The absolute system is started on track zero (0), and built upwards toward and sometimes into the relocatable modules overlaying the latter after they have been converted into absolute code. Therefore, the scratch area must be started at a point high enough so that the absolute system will not overlay the relocatable modules before they are used

. . . . . START SCRATCH?

Operator responds

. . . . . 4 (hp)

RTGEN requests the number of software protected tracks

. . . . . NO. PROTECTED?

Operator responds

. 8

RTGEN requests the number of 64-word sectors (decimal) per track on the system disc or drum

. . . . #SECTORS/TRACK?

Operator responds

. 128

RTGEN requests the number of tracks on the auxiliary disc or drum

. . . . . AUX DISC SIZE?

Operator responds with number of tracks (or zero (0), if there is no auxiliary disc)

. 0

Note that if an auxiliary disc is specified (moving head or fixed head), the number of sectors per track relates to the number of 64-word sectors per track.

(hp) *An entry of 0 here defaults the start scratch to the midpoint of the available disc space. (e.g., if disc size is 32, a 0 entry starts the scratch at track 16.)*

RTGEN requests the select code of the time base generator (octal)

. . . . . TBG CHNL?

Operator responds

. 12

RTGEN requests the address of the privileged interrupt I/O card (if present)

. . . . . PRIV. INT. CARD ADDR?

Operator responds with the octal select code of the privileged interrupt HP 12620 card (and all devices in higher priority slot become privileged) or zero if the card is not used

. 11

RTGEN asks whether the swapping option (which allows real-time disc-resident programs to swap in and out of core according to priority) is to be included in the RTE System

. . . . . SWAPPING?

Operator responds with YES or NO

. . . . . YES

RTGEN requests the last word of available core memory, in octal

. . . . . LWA MEM?

Operator responds with 37677 for 16K, 57677 for 24K, or 77677 for 32K

. 37677

RTGEN requests the type of input unit for relocatable program modules

. . . . . PRGM INPT?

Operator responds with PT (for paper tape), TY (for teleprinter), MT (for magnetic tape), or DF (for disc file)

. . . . . PT

RTGEN requests the type of input unit for relocatable library programs

. . . . . LIBR INPT?

Operator responds with PT, TY, MT, or DF

. . . . . PT

NOTE

Any type of program can be entered through the program input unit or the library input unit.

RTGEN requests the type of input unit for parameters, describing the relocatable programs

. . . . . PRAM INPT?

Operator responds with PT or TY

. . . . . TY

When RTGEN finishes the initialization phase, the hardware disc protect switch on the disc controller is checked. If the disc is protected RTGEN prints

. . . . . TURN OFF DISC PROTECT – PRESS RUN

and halts with the Memory Data Register = 102032. The operator must turn off the disc protect switch and press RUN to continue to the next phase. If the switch already off, the message is not printed and the computer halts with the Memory Data Register = 102077. If the response to PGM INPT or LIB INPT was MT, the magnetic tape unit will rewind to the load point, and then space forward to relocatable file number two, before the HALT 102077.

PROGRAM INPUT PHASE

During the program input phase, RTGEN accepts relocatable programs from the program input unit and library input unit specified during the initialization phase. The operator selects the input device by setting switch register bits 0-1.

00 = program input unit

10 = library input unit

01 = print list of undefined externals

- or -

01 = terminate input phase

Relocatable programs should be loaded in the following order:

Exec Control (EXEC)

Scheduler (SCHED)

I/O Control (RTIOC)

I/O Drivers

System programs written by the user

Foreground core-resident programs

Foreground disc-resident programs

Background core-resident programs

Assembler (main and its segments)

FORTRAN (main and its segments) and/or

FORTRAN IV (main and its segments but not both FORTRAN IV versions)

Relocating loader

Editor

Other background disc-resident programs and their respective segments, if any.

Library Programs

Utility Programs

NOTE

Some of the above relocatable modules may not be present in some configurations.

If a program is being loaded from paper tape, and was not generated with the type code in the NAM record, the operator must set switch register bits 3-6 to the type code (0 to 8). However, programs on a magnetic tape or fixed head disc file must have the type code in the NAM record.

The operator presses RUN. After a program is loaded, the message "*EOT" is printed whenever an end-of-record occurs. The computer halts.

At this point, the operator has several options:

a. Additional programs can be loaded through the same device by repeating the steps above.

b. Input can be switched to the other input device by setting the switch register bits to 00_2 or 10_2.

c. After each *EOT message, a list of all undefined externals can be printed by setting the switch register bits to 01_2 and pushing RUN. At this point the operator can reset the switches to point to the desired input device, and load additional routines needed to satisfy any undefined externals. If there are none, the message NO UNDEF EXTS is printed and the computer executes a HALT 77. To continue loading programs, reset the switch register to 00_2, or 10_2 place the program in the input device, and push RUN.

d. To restart this phase go to the starting address of the program, 100_8, clear the switch register, and push RUN.

e. To terminate the program input phase, set the switch register to 01_2 and push RUN. (RUN must be pushed again after NO UNDEF EXTS is printed.) If magnetic tape is used for the program input, it will rewind and go off-line at this point (after the second RUN).

PARAMETER INPUT PHASE

If the teletype was not specified as the PRAM INPT device during initialization, the computer executes a HALT 77 to wait for the parameter tape to be inserted in the photoreader. Push RUN to continue. If there are any errors on the parameter tape, they will be printed on the teletype.

During the parameter input phase, the operator can modify the type, priority, or execution intervals of any of the programs entered during the program input phase (except that the program type code of background main programs and their segments cannot be changed without losing their relationship to each other).

Each parameter record is of this general form:

name, type [,*priority*] [, *execution interval*]

Refer to the Configuration Worksheet and enter any modification parameters that are listed. If there are none, enter a /E on the teletype. As an example (see Appendix C) the operator can respond with

```
..... START,82,3
      /E
```

NOTE

Refer to Part 1, Parameter Input Phase for an explanation of type codes 81 thru 84.

RTGEN requests the number of blank ID segments to be allocated for on-line loading of programs by the relocating loader

. . . . . # OF BLANK ID SEGMENTS?

This phase can be restarted up to this point by going to the starting address of the program, 100_8, clearing the switch register, and pushing RUN. If restart is not initiated, the operator responds with a one or two digit decimal number (zero is changed to one, because one is required to do any on-line loading); 29 words are reserved in the resident table area for each blank ID segment

. 5

DISC LOADING PHASE

RTGEN requests the first word of available core memory in base page

. . . FWA BP LINKAGE?

Operator responds with the first available octal select code number after the last I/O card

. 30

Disc loading begins with the modules of the Real-Time Executive, including I/O drivers. As RTGEN loads these programs, it prints SYSTEM, followed by a memory map giving the starting locations and, if switch register bit 15 is set, the entry points for all main programs and subroutines (subroutines are indented two spaces).

After the last system module is loaded RTGEN gives the new first word of available core memory in base page

. . . BP LINKAGE 00420

Next, RTGEN generates the three I/O tables: Equipment table, device reference table, and the interrupt table.

RTGEN requests the equipment table entries

. . *EQUIPMENT TABLE ENTRY

Operator responds with a series of one line EQT entries, which are assigned EQT numbers sequentially from one as they are entered. The EQT entry relates the EQT number to an I/O channel and driver. Refer to the Configuration Worksheet and enter the Equipment table entries. The Appendix C example is entered as follows

. . . . . 23, DVR30, D
25, DVR31, D
10, DVR55
13, DVR76, T
T =
100
16, DVR00, B
17, DVR01, T
T =
10
20, DVR02, B, T
T =
40
21, DVR12
22, DVR56, D
/E

RTGEN requests the logical unit assignments for the device reference table

. . *DEVICE REFERENCE TABLE

For each logical unit number, RTGEN prints

. . . . . *n*=EQT# ?

where *n* is a decimal integer starting with one.

Operator responds with an EQT entry number appropriate to the standard definition of *n*, and the subchannel number if appropriate. Logical unit numbers 1 through 6 are predefined in the RTE System as:

1 – system teleprinter
2 – system mass storage
3 – auxiliary mass storage
4 – standard punch unit
5 – standard input unit
6 – standard list unit

Refer to the Configuration Worksheet and enter the Device Reference Table Entries. The Appendix C example is entered as follows . . . . . .

(system teleprinter) 1 = EQT #?
5
(system mass storage) 2 = EQT #?
1
(auxiliary mass storage) 3 = EQT #?
0
(standard punch unit) 4 = EQT #?
7
(standard input unit) 5 = EQT #?
6
(standard list unit) 6 = EQT #?
8
7 = EQT #?
2,0
8 = EQT #?
2,1
9 = EQT #?
4

```
10 = EQT #?
3
11 = EQT #?
9
12 = EQT #?
/E
```

RTGEN requests the interrupt table entries

. . *INTERRUPT TABLE

Operator responds with an entry for each I/O card, in ascending order (except I/O location 4). Refer to the Configuration Worksheet and enter the Interrupt Table Entries.

Note that the entry for location 4 (power-fail) may be entered out of order. This is the only location allowed out of order. The Appendix C example is entered as follows . . .

```
4,ENT,POWER
10,EQT,3
13,EQT,4
14,ABS,106714
15,ABS,106715
16,EQT,5
17,EQT,6
20,EQT,7
21,EQT,8
22,EQT,9
23,EQT,1
24,EQT,1
25,EQT,2
26,EQT,2
/E
```

When the interrupt table is completed RTGEN prints LIBRARY, followed by names and entry point addresses of all routines which are referenced by real-time and background programs. After this RTGEN reports the new first word of available core memory in base page

. . . . BP LINKAGExxxxx

Next RTGEN prints RT RESIDENTS, and loads the real-time core resident programs and reports the new first word of available core memory in base page

. . . . BP LINKAGExxxxx

NOTE

If there are no library or real-time core resident programs, RTGEN prints (NONE) and does not report the links because they have not changed.

RTGEN prints RT DISC RESIDENTS and loads the real-time disc-resident programs and reports the new first word of available core memory in a base page

. . . BP LINKAGE 00421

Next, RTGEN requests any change to the base page linkage area (in order to increase base page linkage area for future addition of larger real-time disc-resident programs on-line with the RTE Relocating Loader

. . . . . CHANGE BP LINKAGE?

Operator responds with zero for no change, or an octal address greater than the last base page link reported. Refer to Part 1 for guidelines in determining this entry. A recommended entry is

. 1100

RTGEN next reports the first word address of the system available memory

. . . . . FWA SY MEM 22340

Next, RTGEN requests any changes to this address (in order to increase the real-time disc-resident area for future addition of larger programs)

. . . . CHANGE FWA SY AV MEM?

Operator responds with zero for no change, or an octal address greater than the FWA reported. Refer to Part 1 for guidelines in determining this entry. The Appendix C response is

. 25400

Finally, RTGEN requests the first address of the background area. This is the area between the FWA SY MEM and the background and is used for temporary storage of output buffers, and re-entrant temporaries; it should be large enough to handle the largest anticipated buffered I/O transfer. Refer to Part 1 for guidelines in determining this entry

. . . . . BG BOUNDARY?

The Appendix C response is octal address

. 26000

RTGEN prints BG RESIDENTS, and proceeds to load the background resident programs and prints names and entry points for mains and subroutines. RTGEN then prints BG DISC RESIDENTS, and proceeds to load the background disc-resident programs and prints names and entry points for mains and subroutines. RTGEN completes the loading and gives the total base page links used

. . . . BP LINKAGE 1561

RTGEN then reports the system is loaded on the disc by printing

. . . . SYSTEM STORED ON DISC

RTGEN gives the address of the last sector and track used in storing the RTE System on the disc, and halts

LAST SYS DISC ADDR:
TRK 06 SEC 014 (10)

where both the track number and the sector number, are decimal numbers. These should be recorded if the RTE

System is going to be dumped on to tape for back-up protection using SDUMP.

The disc loading phase can be restarted at this point by going to starting address 100_8, clearing the switch register, and pushing RUN. In addition, the previous phase (parameter input) can also be restarted at this point by going to address 4000_8, clearing the switch register, and pushing RUN.

INITIATING RTE FROM THE DISC

The basic binary disc loader (BBDL), a modified version of the standard basic binary loader, resides in the highest, protected 64 words of core and loads either absolute format paper tapes or disc-based systems, such as RTE System.

LOADING THE RTE SYSTEM

a. The operator goes to the starting address of the BBDL;

   037760_8 for 16K

   057760_8 for 24K

   077760_8 for 32K

   and pushes RUN.

b. When the computer halts with 102077_8 in the memory data register, the operator clears switch register and presses RUN. When RTE is loaded, it will type:

   SET TIME

The operator either sets the clock to current day time using the TM operator request, or types any other request (the system starts at time zero).

SDUMP

SDUMP, the System Dump, is an independent absolute program that can create back-up copies of fixed head disc-based systems on punched tape or magnetic tape. The back-up copy can later be reloaded onto the fixed head disc by SDUMP. In a fixed head system with a moving head auxiliary disc, SDUMP will only dump that portion of the system located on the fixed head disc.

Because it is an independent program like RTGEN, SDUMP requires the SIO Drivers. For paper tape storage, the SIO Teleprinter Driver, SIO Paper Tape Reader Driver, and the SIO Paper Tape Punch Driver are required. For magnetic tape storage, the SIO Teleprinter Driver and SIO Magnetic Tape Unit Driver are required. The operator loads the SDUMP tape and SIO Driver tapes as described for RTGEN. The magnetic tape driver must be loaded after SDUMP.

After loading SDUMP, go to starting address 100_8, and press RUN. SDUMP prints out a request guide on teleprinter:

DUMP = D,T[-S] [,T[-S]] ([ ] = OPTIONAL)

VERIFY = V

LOAD = L

TERMINATE = T

Then, SDUMP requests the lower-number select code of the disc in octal

. . . . . .DISC CHNL?

Operator responds

. 23

After SDUMP types

. . . . . COMMAND

Operator responds with *V, L, T, or D* (*D* requires parameters)

. . . . . D,0,10-7

An explanation of the parameters that SDUMP uses is as follows.

D for dumping requires octal parameters specifying the track or tracks to be dumped. For example, the operator response above tells SDUMP to dump the absolute binary code stored on tracks 0 through 10 up to and including sector 7. If only one number is entered, only that track is dumped. These numbers are obtained from the RTGEN listing. RTGEN gives the address of the last sector and tracks used in storing the RTE system under the heading

LAST SYS DISC ADDR:
TRK nn SECmmm(10)

It should be noted that RTGEN gives the track and sector numbers in decimal. Use Table 6-2 to convert these numbers to octal for SDUMP. (First physical track of the disc is decimal 0.)

Table 6-2. Octal/Decimal Conversion

Octal	Decimal	Octal	Decimal
0	0	16	14
1	1	17	15
2	2	20	16
3	3	21	17
4	4	22	18
5	5	23	19
6	6	24	20
7	7	25	21
10	8	26	22
11	9	27	23
12	10	30	24
13	11	31	25
14	12	32	26
15	13	33	27

The sector parameter (-*S*) is optional. If output is to paper tape, trailer and leader blank tape is produced, and two tracks are dumped at a time. Always start with a new roll of tape. If output is to magnetic tape, the entire information is dumped, followed by an End-Of-File which is written over by subsequent dumps.

V for verifying, involves placing the dump in the input device, reading it in, and checking each record against the contents of the disc. Comparison errors are reported. If magnetic tape is verified, only one file is checked.

L for loading causes the dumped information to be loaded back onto the disc. The information is verified after it is output to the disc, and comparison errors are reported.

T by itself is for termination. However, *T* used in the parameter explanation stands for track.

An illegal command causes the message:

STATEMENT ERROR

An error in specifying the disc channel, causes the message:

PARAM ERROR: NON-NUMERIC OR NON-OCTAL

If the magnetic tape is used, a rewind is issued during initialization, before and after a verify or load operation, and rewind/standby after *T* for termination.

SDUMP ERROR MESSAGES

The following messages may be printed on the teleprinter by SDUMP:

STATEMENT ERROR

Re-type input statement in correct format.

EOT

The end of the input tape being read has been reached; either load the next tape or go on to the next phase.

CHANGE OUTPUT TAPE, HIT RUN

Two full tracks have been dumped onto paper tape; perform the requested action.

TURN OFF DISC PROTECT, HIT RUN

Set the disc track protect switch off, then press RUN.

DISC INPUT ERROR

Disc error diagnostic, for a parity or decode or abort status after 10 re-trys. Input sequence repeated on restart.

DISC WRITE ABORT

Disc error diagnostic, for an abort status after a write attempt. Sequence is repeated if restarted.

TRACK 27(8) SECTOR 107 (8)

Identification information for the disc and tape error diagnostics are described as follows:

TAPE/DISC VERIFY ERROR

Disc and tape records do not agree. Disc record is re-written on restart.

TAPE CHECKSUM ERROR

The checksum in the tape record does not match the sum computed by SDUMP. Current record is ignored if re-started.

MT ERROR – READ PARITY
MT ERROR – EOT RESTART

Magnetic tape errors. Error recovery procedures are completed by driver. Restart to re-try sequence.

FH RTGEN ERROR MESSAGES

The following messages may be printed on the teleprinter during execution of RTGEN:

Messages During Initialization and Input Phase

ERR 01

Meaning: Invalid response to initialization request.

Action: Message is repeated. Enter valid reply.

ERR 02

Meaning: Checksum error on program input.

Action: Computer halts; reposition tape to beginning of record and press RUN to reread. If input is from MT or DF, it is automatically backspaced when RUN is pressed.

ERR 03

Meaning: Record out of sequence.

Action: Same as ERR 02.

ERR 04

Meaning: Illegal record type.

Action: Same as ERR 02.

ERR 05 name

Meaning: Duplicate entry point.

Action: Revise program by re-labeling the entry points (the current entry point replaces the previous entry point).

ERR 06

Meaning: Invalid base page length (must be zero).

Action: Base page area is ignored, but memory protect error will occur if program is executed.

ERR 07

Meaning: Program name or entry point table overflow of available memory.

Action: Irrecoverable error. Revise or delete programs.

ERR 08 name

Meaning: Duplicate program name.

Action: The current program replaces the previous program.

Messages During the Parameter Phase

ERR 09

Meaning: Parameter name error (no such program).

Action: Enter valid parameter statement.

ERR 10

Meaning: Parameter type error.

Action: Same as ERR 09.

ERR 11

Meaning: Parameter type error.

Action: Same as ERR 09.

ERR 12

Meaning: Execution interval error.

Action: Same as ERR 09.

General Messages

ERR 13

Meaning: BG segment precedes BG main disc-resident program.

Action: Irrecoverable.

ERR 14

Meaning: Invalid background bounds or illegal response to CHANGE FWA SYS MEM or to CHANGE BP LINKAGE?

Action: Message is repeated. Enter valid reply.

ERR 15

Meaning: Type 6 program illegally calling a program that is not type 0 or 6.

Action: Revise type 6 program.

ERR 16

Meaning: Base page linkage overflow into system communication area.

Action: Diagnostic printed for each word required (communication area is used). Revise order of program loading or CHANGE BP LINKAGE answer to reduce linkage requirements.

ERR 17

Meaning: Current disc address exceeds number of available tracks.

Action: Irrecoverable error.

ERR 18

Meaning: Memory overflow (absolute code exceeds LWA memory).

Action: Diagnostic printed for each word required (absolute code is generated beyond LWA). Revise program or BG BOUNDARY answer.

ERR 19

Meaning: Program overlay (current word of absolute code has identical location to previous).

Action: Current word (the address is printed) is ignored.

ERR 2Ø

Meaning: Binary DBL record overflow of internal table.

Action: Records overlay previous DBL records (diagnostic printed for each overflow record). Revise program.

ERR 21

Meaning: Module containing entry point $CIC not loaded.

Action: Irrecoverable error.

ERR 22

Meaning: Read parity/decode disc error. A-Register bits 7-14 show track number; bits 0-6 show sector number.

Action: After ten attempts to read or write the disc sector, the computer halts. To try ten more times, press RUN.

ERR 23

Meaning: Invalid FWA BP LINKAGE.

Action: Message repeated; enter valid reply.

Messages During I/O Table Entry

ERR 24

Meaning: Invalid channel number.

Action: Enter valid EQT statement.

ERR 25

Meaning: Invalid driver name or no driver entry points.

Action: Same as ERR 24.

ERR 26

Meaning: Invalid or duplicate D,B,T operands.

Action: Same as ERR 24.

ERR 27

Meaning: Invalid logical unit number.

Action: Enter valid DRT statement.

ERR 28

Meaning: Invalid channel number.

Action: Enter valid INT statement

ERR 29

Meaning: Channel number decreasing.

Action: Same as ERR 28.

ERR 30

Meaning: Invalid mnemonic.

Action: Same as ERR 28.

ERR 31

Meaning: Invalid EQT number.

Action: Same as ERR 28.

ERR 32

Meaning: Invalid program name.

Action: Same as ERR 28.

ERR 33

Meaning: Invalid entry point.

Action: Same as ERR 28.

ERR 34

Meaning: Invalid absolute value.

Action: Same as ERR 28.

ERR 35

Meaning: Base page interrupt locations overflow into linkage area.

Action: Re-start Disc Loading Phase at FWA BP LINKAGE? request.

ERR 36

Meaning: Invalid number of characters in final operand.

Action: Same as ERR 28.

General Message

ERR 37 name

Meaning: Invalid declaration of common in system or library programs (name is the illegal program).

Action: Revise the program.

ERR 38

Meaning: System area overflows scratch area. (This error sometimes possible when restarting disc loading phase; irrecoverable).

Action: Check order of loading programs or move scratch boundary up.

ERR 39 name

Meaning: System illegally referenced a type 6 program (name is the type 6 program).

Action: Revise the program.

PART 3
MOVING HEAD RTE SYSTEM GENERATION

The set up and operation of a Moving Head RTE System involves two essential steps; the RTE System must be configured using the Real-Time System Generator, RTGEN (29014-60001), and it must be initiated from the disc by a bootstrap tape that is punched during RTGEN.

This part describes the steps necessary to configure an RTE System, with the absolute binary code of the RTE based on a moving head disc drive. All of the required information should be pre-planned and located on the Input/Output and Configuration Worksheets filled out in Part 1. The answers used in this part are from an example RTE System that has been configured on worksheets per the instructions in Part 1. The completed worksheets and RTGEN listing are located in Appendix C.

MOVING HEAD DISC RTGEN

RTGEN operates on the same minimum configuration as that required for an RTE System, and configures the system to fit a particular user's core memory size, I/O equipment, and programming needs.

To accomplish this, RTGEN requests certain information from the user; then it accepts the relocatable program modules to be included in the system, determines where they belong in core, relocates them into absolute format, and stores them on the disc. RTGEN also creates I/O tables by identifying each I/O device and its associated driver routine, and establishing procedures for interrupt processing on each channel.

During generation, RTGEN will report defective tracks and initialize them defective. Up to ten bad tracks are allowed in the system before RTGEN aborts. The system cannot operate if any bad tracks are located within the absolute code of RTE on LU2.

The RTE System is limited to the disc tracks specified during RTGEN. As a result, the user can create many different configurations of RTE Systems, all coresiding on the same moving head disc subchannels.

RTGEN is an absolute program, loaded into core by the Basic Binary Loader (BBL) from paper tape. Since RTGEN is independent of the RTE System which it generates, the I/O operations of RTGEN require SIO Drivers.

OPERATING PROCEDURES

The operation of RTGEN involves four phases:

a. INITIALIZATION PHASE. RTGEN requests specifications for the RTE System, including a track map of system disc space, memory, time base generator channel, swapping option, and program input devices.

b. PROGRAM INPUT PHASE. The operator loads the relocatable programs provided with the system, and created by the user.

c. PARAMETER INPUT PHASE. Parameters describing or changing the type, priority, and execution interval of each program may be entered. (Although this information may already be included in the program's NAM record, it can be changed at this point.) RTGEN requests the number of blank ID segments to be reserved for subsequent program addition.

d. DISC LOADING PHASE. RTGEN requests a specification of the base page linkage, and begins loading programs onto the disc. The modules of RTE, drivers, and system programs are loaded first, after which RTGEN then requests information for the equipment table, device reference table, and interrupt table, and proceeds to load the rest of the programs onto the disc.

NOTE

During any of the phases, RTGEN can restart that phase at starting address 100_8. Note, the switch register must be cleared before pressing RUN.

To execute RTGEN and configure an RTE System, follow these steps:

1. Apply power to all equipment, and set the system teleprinter to LINE. Apply power to the HP 7900 Disc Drive and proceed as follows.
   a. Insert the removable disc cartridge into the HP 7900. While the door is open, set the two data protect slide switches (one on the right side, and one on the left side) to OFF.
   b. Close the door and set the LOAD/UNLOAD switch to LOAD. In approximately 30 seconds the drive will be at operating speed and the DRIVE READY lamp will light.
   c. The disc OVERRIDE/PROTECT switch is located on the HP 7900 Disc Drive. Remove the front filter screen by pushing the bottom edge in. This cocks the screen at an angle and allows it to be pulled away from the opening. Set the switch to the OVERRIDE position.
   d. If, during operation, the DRIVE FAULT lamp should light, cycle the LOAD/UNLOAD switch allowing the drive to stop, and then come up to

speed again. If the DRIVE FAULT lamp remains lit, refer to the HP 7900 manual, Data Protect Logic checks.

2. Load the RTGEN tape and configured SIO Drivers into core using the BBL.
3. If a fixed head disc is to be used to store the relocatable modules, its SIO driver must be configured as described in the PTS Manual on the SIO Disc/Drum Driver Configuration Flowchart.
4. Go to the starting address of the program, 100_8, clear the switch register, and push RUN. RTGEN begins the initialization phase.

INITIALIZATION PHASE

During the initialization phase, RTGEN first requests information necessary to generate a disc track map that defines which tracks of each platter are assigned to RTE. Once the track map is established RTGEN goes on to request more information necessary to generate the RTE System. After each question is printed, the operator responds with the required answer followed by carriage return/line feed (CR/LF). Operator responses are shaded and are only examples (see Appendix C); actual responses should be appropriate to the particular system being generated. If an error is made and discovered before LF is entered, type the RUBOUT key then CR/LF. Otherwise restart at step 4 above.

RTGEN requests the higher priority select code (octal) of the system disc controller

. . . . . MH DISC CHNL?

Operator responds

. 13

RTGEN requests the starting track and number of tracks (decimal) of each subchannel that will be assigned to the system. Up to eight track assignments can be entered, one for each existing subchannel. Subchannel 0 is the fixed platter and subchannel 1 is the removable platter of the first drive. Thereafter the even numbered subchannels are the fixed platters and the odd numbered subchannels are the removable platters

. . . . # TRKS, FIRST TRK ON SUBCHNL: 0?

Operator responds with the decimal number of tracks and starting track number for subchannel 0. If there are to be no tracks from subchannel 0 assigned to the system, enter a 0. Refer to the Configuration Worksheet and enter the correct numbers. The Appendix C example is entered as follows:

. 0

RTGEN continues to request the track assignments for each subchannel up to seven or until /E is entered.

1?
0
2?
50,0
3?
200,0
4?
100,50
5?
50,50
6?
/E

NOTE

The following four questions concerning system subchannel, auxiliary disc, auxiliary disc subchannel, and scratch disc subchannel are not asked if only one subchannel is assigned to the RTE System.

RTGEN requests the subchannel number of the system disc (LU2). This is the disc that the absolute code will be stored upon and can be any one of the subchannels assigned to the system

. . . . SYSTEM SUBCHNL?

Operator responds with a subchannel number corresponding to one of the above entries that contains enough tracks for the absolute code

. 5

RTGEN asks if there is to be an auxiliary disc (LU3)

. . . . . AUX DISC (YES OR NO)?

Operator responds with yes or no

. . . . . YES

RTGEN requests the auxiliary disc subchannel number. If there is not an auxiliary disc (answer to above question was no), then RTGEN does not ask this question

. . . AUX DISC SUBCHNL?

Operator responds with a valid subchannel number (not the system subchannel)

. 4

RTGEN requests the subchannel number of the scratch disc. This is the area required for the relocatable modules used to build the system, and can be any one of the subchannels assigned to the system. It is recommended that the scratch area not be located on the system subchannel. Then the absolute area cannot overlay the relocatable modules before they are used.

. . . SCRATCH SUBCHNL?

Operator responds

. 4

RTGEN requests the track number starting the disc scratch area

. . . . . START SCRATCH?

Operator responds with a relative track number. For example, subchannel 4 has tracks addresses 50 to 150 available. To start the scratch area on the first available track (50) enter a zero (0). To start the scratch at track 75, enter 25. Note that if the scratch subchannel is the same as the system subchannel, entering a zero defaults the start scratch to the midpoint of the available disc space on the system subchannel

. 50

RTGEN requests the number of 128 word sectors (decimal) per logical track on the system disc

. . . . . # 128 WORD SECTORS/TRACK?

Operator responds

. 48

RTGEN requests the select code of the time base generator (octal)

. . . . . TBG CHNL?

Operator responds

. 12

RTGEN requests the address of the privileged interrupt I/O card (if present)

. . . . . PRIV. INT. CARD ADDR?

Operator responds with the octal select code of the privileged interrupt HP 12620 card (and all devices in higher priority slots become privileged), or zero if the card is not used

. 11

RTGEN asks whether the swapping option (which allows real-time disc-resident programs to swap in and out of core according to priority) is to be included in the RTE System

. . . . . SWAPPING?

Operator responds with YES or NO

. . . . . YES

RTGEN requests the last word of available core memory, in octal

. . . . . LWA MEM?

Operator responds with 37677 for 16K, 57677 for 24K, or 77677 for 32K

. 37677

RTGEN requests the type of input unit for relocatable program modules

. . . . . PRGM INPT?

Operator responds with PT (for paper tape), TY (for teleprinter), MT (for magnetic tape) or DF (for disc file)

. . . . . PT

RTGEN requests the type of input unit for relocatable library programs

. . . . . LIBR INPT?

Operator responds with PT, TY, MT, or DF

. . . . . PT

NOTE

Any type of program can be entered through the program input unit or the library input unit.

RTGEN requests the type of input unit for parameters, describing the relocatable programs

. . . . . PRAM INPT?

Operator responds with PT or TY

. . . . . TY

RTGEN asks if the operator wishes to initialize disc subchannels other than the system, auxiliary, and scratch subchannel. RTGEN asks this question only for those subchannels assigned to RTE but not declared as LU2, LU3, or scratch, these subchannels being automatically initialized by RTGEN

. . . . . INITIALIZE SUBCHNL: 2?

Operator responds with a YES or NO. If the disc is new or has any write protect flags written on it, it must be initialized. If the disc has data stored on it in the system designated area, and the user does not want to disturb it, the answer is NO

. . . . . NO

3?

. . . . . NO

Next RTGEN checks the hardware disc protect switch. If the disc is protected RTGEN prints

. . . . . TURN OFF DISC PROTECT – PRESS RUN

and halts with the Memory Data Register = 102032. The operator must turn off the disc protect switch and press RUN to continue to the next phase. If the switch is already off the message is not printed and the computer halts with Memory Data Register = 102077. If the response to PGM INPT or LIB INPT was MT, the magnetic tape unit will rewind to the load point, and then space forward to relocatable file number two, before the HALT 102077.

RTGEN asks if a paper tape bootstrap is to be punched. The first answer should be YES. Note that the bootstrap is unique to the absolute system and first track number only. RTGEN keeps repeating the question until NO is entered. In this fashion the user can punch as many bootstraps as he feels he needs

. . . . . PUNCH BOOT?

NOTE

Turn the paper tape punch ON.

Operator responds with
. . . . . YES

RTGEN repeats the above question
. . . . . PUNCH BOOT?

Operator responds with yes or no
. . . . . NO

PROGRAM INPUT PHASE

During the program input phase, RTGEN accepts relocatable programs from the program input unit and library input unit specified during the initialization phase. The operator selects the input device by setting switch register bits 0-1.

00 = program input unit

10 = library input unit

01 = print list of undefined externals

— or —

01 = terminate input phase

Relocatable programs should be loaded in the following order.

Exec Control (EXEC)

Scheduler (SCHED)

I/O Control (RTIOC)

I/O Drivers

System programs written by the user

Foreground core-resident programs

Foreground disc-resident programs

Background core-resident programs

Assembler (main and its segments)

FORTRAN (main and its segments) and/or

FORTRAN IV (main and its segments but not both FORTRAN IV versions)

Relocating loader

Editor

Other background disc-resident programs and their respective segments, if any.

Library Programs

Utility Programs

NOTE

Some of the above relocatable modules may not be present in some configurations.

If a program is being loaded from paper tape, and was not generated with the type code in the NAM record, the operator must set switch register bits 3-6 to the type code (0 to 8). However, programs on a magnetic tape or fixed head disc file must have the type code in the NAM record.

The operator presses RUN. After a program is loaded, the message "*EOT" is printed whenever an end-of-record occurs. The computer halts.

At this point, the operator has several options:

a. Additional programs can be loaded through the same device by repeating the steps above.

b. Input can be switched to the other input device by setting the switch register bits to 00_2 or 10_2.

c. After each *EOT message, a list of all undefined externals can be printed by setting the switch register bits to 01_2 and pushing RUN. At this point the operator can reset the switches to point to the desired input device, and load additional routines needed to satisfy any undefined externals. If there are none, the message NO UNDEF EXTS is printed and the computer executes a HALT 77. To continue loading programs reset the switch register to 00_2 or 10_2, place the program in the input device, and push RUN.

d. To restart this phase, go to the starting address of the program, 100_8 clear the switch register, and push RUN.

e. To terminate the program input phase, set the switch register to 01_2 and push RUN (RUN must be pushed again after NO UNDEF EXTS is printed). If magnetic tape is used for the program input, it will rewind and go off-line at this point (after the second RUN).

PARAMETER INPUT PHASE

If the teletype was not specified as the PRAM INPT device during initialization, the computer executes a HALT 77 to wait for the parameter tape to be inserted in the photoreader. Push RUN to continue. If there are any errors on the parameter tape they will be printed on the teletype.

During the parameter input phase, the operator can modify the type, priority, or execution intervals of any of the programs entered during the program input phase (except that the program type code of the background main programs and their segments cannot be changed without losing their relationship to each other).

Each parameter record is of this general form:

name, type [,*priority*] [,*execution interval*]

Refer to the Configuration Worksheet and enter any modification parameters that are listed. If there are none, enter a /E on the teletype. As an example (see Appendix C), the operator can respond with

. . . . . START,81,4
. . . . . FSWP,2,3
. . . . . /E

NOTE

Refer to Part 1, Parameter Input Phase for an explanation of type codes 81 thru 84.

RTGEN requests the number of blank ID segments to be allocated for on-line loading of programs by the relocating loader

. . . . . # OF BLANK ID SEGMENTS?

This phase can be restarted up to this point by going to the starting address of the program, 100_8, clearing the switch register and pushing RUN. If restart is not initiated, the operator responds with a one or two digit decimal number (zero is changed to one, because one is required to do any on-line loading); 29 words are reserved in the resident table area for each blank ID segment

. 4

DISC LOADING PHASE

RTGEN requests the first word of available core memory in base page

. . . FWA BP LINKAGE?

Operator responds with the first available octal select code number after the last I/O card

. 26

Disc loading begins with the modules of the Real-Time Executive, including I/O drivers. As RTGEN loads these programs, it prints SYSTEM, followed by a memory map giving the starting locations and, if switch register bit 15 is set, the entry points for all main programs and subroutines (subroutines are indented two spaces).

After the last system module is loaded RTGEN gives the new first word of available core memory in base page

. . . BP LINKAGE 00410

Next, RTGEN generates the three I/O tables: Equipment table, device reference table, and the interrupt table.

RTGEN requests the equipment table entries

. . * EQUIPMENT TABLE ENTRY

Operator responds with a series of one line EQT entries, which are assigned EQT numbers sequentially from one as they are entered. The EQT entry relates the EQT number to an I/O channel and driver. Refer to the Configuration Worksheet and enter the Equipment table entries. The Appendix C example is entered as follows

```
13,DVR31,D,T
T =
30
10,DVR55
21,DVR77,T
T =
40
24,DVR30,D,T
T =
30
15,DVR00,B,T,
T =
1000
16,DVR00,B
17,DVR01,T
T =
10
20,DVR02,B,T
T =
40
/E
```

RTGEN requests the logical unit assignments for the device reference table

*DEVICE REFERENCE TABLE

For each logical unit number, RTGEN prints

. . . . . *n*=EQT#?

where

n is a decimal integer starting with one.

Operator responds with an EQT entry number appropriate to the standard definition of *n*, and the subchannel number if appropriate. Logical unit numbers 1 through 6 are predefined in the RTE System as:

1 – system teleprinter
2 – system mass storage
3 – auxiliary mass storage
4 – standard punch unit
5 – standard input unit
6 – standard list unit

Refer to the Configuration Worksheet and enter the Device Reference Table Entries. The Appendix C example is entered as follows . . . . . .

```
(system teleprinter)                1 = EQT #?
6
(system mass storage)               2 = EQT #?
1,5
(auxiliary mass storage)            3 = EQT #?
1,4
(standard punch unit)               4 = EQT #?
8
(standard input unit)               5 = EQT #?
7
(standard list unit)                6 = EQT #?
5
```

```
    7 = EQT #?
3
    8 = EQT #?
2
    9 = EQT #?
4
   10 = EQT #?
1,2
   11 = EQT #?
1,3
   12 = EQT #?
/E
```

RTGEN requests the interrupt table entries

. . *INTERRUPT TABLE

Operator responds with an entry for each I/O card in ascending order (except I/O location 4). Refer to the Configuration Worksheet and enter the Interrupt Table Entries.

Note that the entry for location 4 (power-fail) may be entered out of order. This is the only location allowed out of order. The Appendix C example is entered as follows . .

```
10,EQT,2
13,EQT,1
14,EQT,1
15,EQT,5
16,EQT,6
17,EQT,7
20,EQT,8
21,EQT,3
22,EQT,3
23,EQT,3
24,EQT,4
25,EQT,5
/E
```

When the interrupt table is completed RTGEN prints LIBRARY, followed by names and entry point addresses of all routines which are referenced by real-time and background programs. After this RTGEN reports the new first word of available core memory in base page

. . . . BP LINKAGExxxxx

Next RTGEN prints RT RESIDENTS, and loads the real-time core-resident programs and reports the new first word of available core memory in base page

. . . BP LINKAGE 00411

NOTE

If there are no library or real-time core-resident programs, RTGEN prints (NONE) and does not report the links because they have not changed.

RTGEN prints RT DISC RESIDENTS, and loads the real-time disc-resident programs and reports the new first word of available core memory in base page

. . . BP LINKAGE 00411

Next, RTGEN requests any change to the base page linkage area (in order to increase base page linkage area for future addition of larger real-time disc-resident programs on-line with the RTE Relocating Loader

. . . CHANGE LINKAGE?

Operator responds with zero for no change, or an octal address greater than the last base page link reported. Refer to Part 1 for guidelines in determining this entry. The Appendix C response is

. 1000

RTGEN next reports the first word address of the system available memory

. . . . FWA SY MEM 21617

Next, RTGEN requests any changes to this address (in order to increase the real-time disc-resident area for future addition of larger programs)

. . . . . .CHANGE FWA SY AV MEM?

Operator responds with zero for no change, or an octal address greater than the FWA reported. Refer to Part 1 for guidelines in determining this entry. The Appendix C response is

. 25300

Finally, RTGEN requests the first address of the background area. This is the area between the FWA SY MEM and the background and is used for temporary storage of output buffers, and re-entrant temporaries; it should be large enough to handle the largest anticipated buffered I/O transfer. Refer to Part 1 for guidelines in determining this entry

. . . . . BG BOUNDARY?

The Appendix C response is octal address

. 26000

RTGEN prints BG RESIDENTS, and proceeds to load the background resident programs and prints names and entry points for mains and subroutines. RTGEN then prints BG DISC RESIDENTS and proceeds to load the background disc-resident programs and prints names and entry points for mains and subroutines. RTGEN completes the loading and gives the total base page links used by printing

. . . . . BP LINKAGE 1461

RTGEN then reports the system is loaded on the disc by printing

. . . . . SYSTEM STORED ON DISC

RTGEN reports the system size by printing the number of track and sectors used (decimal)

. . . . SYS SIZE: 08 TRKS 009 SECS(10)

where both the track number and the sector number are decimal numbers.

The disc loading phase can be restarted at this point by going to starting address 100_8, clearing the switch register, and pushing RUN. In addition, the previous phase (parameter input) can also be restarted at this point by going to address 4000_8, clearing the switch register, and pushing RUN.

INITIATING RTE FROM THE DISC

When RTE has been configured onto the moving head disc, it is loaded into core and initiated by using the small paper tape bootstrap that was punched during RTGEN. Once RTE has been loaded and initiated, it is ready to process user tasks.

OPERATING INSTRUCTIONS

a. Turn on all equipment.

b. Load the bootstrap tape into core using the basic binary loader. Starting addresses are

   037700_8 for 16K

   057700_8 for 24K

   077700_8 for 32K

c. Go to starting address 100_8 (starting address of the bootstrap).

d. Clear the switch register and push RUN.

e. When RTE has been loaded into core, it prints the following message:

   SET TIME

f. The operator either sets the clock to current day time using the TM operator request, or types any other request (the system starts at time zero).

ERROR HALTS

The following halts can occur during use of the bootstrap.

Halt Code	Cause	Recovery Action
102011	Disc error status is in the A-Register.	Check that the disc drive is ready-push RUN to retry.
102031	Same as above. Occurs during execution of disc-resident part of bootstrap.	Check that the disc drive is ready-push RUN to retry.

MH RTGEN ERROR MESSAGES

The following messages may be printed on the teleprinter during execution of RTGEN:

Messages During Initialization and Input Phase

ERR 01

Meaning: Invalid response to initialization request.

Action: Message is repeated. Enter valid reply.

ERR 02

Meaning: Checksum error on program input.

Action: Computer halts; reposition tape to beginning of record and press RUN to reread. If input is from MT or DF, it is automatically backspaced when RUN is pressed.

ERR 03

Meaning: Record out of sequence.

Action: Same as ERR 02.

ERR 04

Meaning: Illegal record type.

Action: Same as ERR 02.

ERR 05

Meaning: Duplicate entry point.

Action: Revise program by re-labeling the entry points (the current entry point replaces the previous entry point).

ERR 06

Meaning: Invalid base page length (must be zero).

Action: Base page area is ignored, but memory protect error will occur if program is executed.

ERR 07

Meaning: Program name or entry point table overflow of available memory.

Action: Irrecoverable error. Revise or delete programs.

ERR 08 name

Meaning: Duplicate program name.

Action: The current program replaces the previous program.

Messages During the Parameter Phase

ERR 09

Meaning: Parameter name error (no such program).

Action: Enter valid parameter statement.

ERR 10

Meaning: Parameter type error.

Action: Same as ERR 09.

ERR 11

Meaning: Parameter priority error.

Action: Same as ERR 09.

ERR 12

Meaning: Execution interval error.

Action: Same as ERR 09.

General Messages

ERR 13

Meaning: BG segment precedes BG main disc-resident program.

Action: Irrecoverable.

ERR 14

Meaning: Invalid background bounds or illegal response to CHANGE FWA SYS MEM? or to CHANGE BP LINKAGE?

Action: Message is repeated. Enter valid reply.

ERR 15

Meaning: Type 6 program illegally calling a program that is not type 0 or 6.

Action: Revise type 6 program.

ERR 16

Meaning: Base page linkage overflow into system communication area.

Action: Diagnostic printed for each word required (communication area is used). Revise order of program loading or CHANGE BP LINKAGE answers to reduce linkage requirements.

ERR 17

Meaning: Current disc address exceeds number of available tracks.

Action: Irrecoverable error.

ERR 18

Meaning: Memory overflow (absolute code exceeds LWA memory).

Action: Diagnostic printed for each word required (absolute code is generated beyond LWA). Revise program or BG BOUNDARY answer.

ERR 19

Meaning: Program overlay (current word of absolute code has identical location to previous).

Action: Current word (the address is printed) is ignored.

ERR 20

Meaning: Binary DBL record overflow of internal table.

Action: Records overlay previous DBL records (diagnostic printed for each overflow record). Revise program.

ERR 21

Meaning: Module containing entry point $CIC not loaded.

Action: Irrecoverable error.

ERR 22

Meaning: Read parity/decode disc error. A-Register bits 7-14 show track number; bits 0-6 show sector number.

Action: After ten attempts to read or write the disc sector, the computer halts. To try ten more times, press RUN.

ERR 23

Meaning: Invalid FWA BP LINKAGE.

Action: Message repeated; enter valid reply.

Messages During I/O Table Entry

ERR 24

Meaning: Invalid channel number.

Action: Enter valid EQT statement.

ERR 25

Meaning: Invalid driver name or no driver entry points.

Action: Same as ERR 24.

ERR 26

Meaning: Invalid or duplicate D,B,T operands.

Action: Same as ERR 24.

ERR 27

Meaning: Invalid logical unit number.

Action: Enter valid DRT statement.

ERR 28

Meaning: Invalid channel number.

Action: Enter valid INT statement.

ERR 29

Meaning: Channel number decreasing.

Action: Same as ERR 28.

ERR 30

Meaning: Invalid mnemonic.

Action: Same as ERR 28.

ERR 31

Meaning: Invalid EQT number.

Action: Same as ERR 28.

ERR 32

Meaning: Invalid program name.

Action: Same as ERR 28.

ERR 33

Meaning: Invalid entry point.

Action: Same as ERR 28.

ERR 34

Meaning: Invalid absolute value.

Action: Same as ERR 28.

ERR 35

Meaning: Base page interrupt locations overflow into linkage area.

Action: Re-start Disc Loading Phase at FWA BP LINKAGE? request.

ERR 36

Meaning: Invalid number of characters in final operand.

Action: Same as ERR 28.

General Message

ERR 37 name

Meaning: Invalid declaration of common in system or library programs (name is the illegal program).

Action: Revise the program.

ERR 38

Meaning: System area overflows scratch area. (This error sometimes possible when restarting disc loading phase; irrecoverable.)

Action: Check order of loading programs or move scratch boundary up.

ERR 39 name

Meaning: System illegally referenced a type 6 program (name is the type 6 program).

Action: Revise the program.

ERR 40

Meaning: First system track defective or first scratch track defective.

Action: Redefine the track areas.

ERR 41

Meaning: More than 10 bad tracks on system, auxiliary and scratch discs combined.

Action: Redefine track area.

ERR 42

Meaning: Absolute system area contains a bad track.

Action: Redefine track area.

ERR 43

Meaning: Disc specifications do not conform to system disc.

Action: Redefine track area or sectors per track answers.

APPENDIX A
TABLES

This Appendix contains several useful tables and diagrams.

EQUIPMENT TABLE

The Equipment Table (EQT) has an entry for each device recognized by RTE (these entries are established by the user when the RTE System is generated). These EQT entries reside in the permanent core-resident part of the system, and have this format:

Table A-1. Equipment Table Entry Diagram

Word	Contents															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	Device Suspended List Pointer															
2	Driver "Initiation" Section Address															
3	Driver "Completion" Section Address															
4	D	B	Not Used			T	Not Used			Unit #			Channel #			
5	AV		EQUIP. TYPE CODE					STATUS								
6	CONWD (Current I/O Request Word)															
7	Request Buffer Address															
8	Request Buffer Length															
9	Temporary, Disc Track #, or Optional Parameter															
10	Temporary, Disc Sector #, or Optional Parameter															
11	Temporary Storage for Driver															
12	Temporary Storage for Driver															
13	Temporary Storage for Driver															
14	Device Time-Out Value															
15	Device Time-Out Clock															

BASE PAGE COMMUNICATION AREA

A block of storage in base page, starting at 1650_8, contains the system communication area and is used by RTE to define request parameters, I/O tables, scheduling lists, operating parameters, memory bounds, etc. The Real-Time Assembler allows absolute references into this area (i.e., less than 2000_8) within relocatable programs, so that user programs can read information from this area, but cannot alter it because of the memory protect feature.

The base page communication area contains:

Octal Location	Contents	Description
SYSTEM TABLE DEFINITION		
01650	EQTA	FWA of equipment table
01651	EQT#	No. of EQT entries
01652	DRT	FWA of device reference table
01653	LUMAX	No. of logical units (in DRT)
01654	INTBA	FWA of interrupt table
01655	INTLG	No. of interrupt table entries
01656	TAT	FWA of track assignment table
01657	KEYWD	FWA of keyword block
I/O MODULE/DRIVER COMMUNICATION		
01660	EQT1	Addresses
01661	EQT2	of
01662	EQT3	first
01663	EQT4	11-words
01664	EQT5	of
01665	EQT6	current
01666	EQT7	EQT
01667	EQT8	entry
01670	EQT9	(see 01771
01671	EQT10	for last
01672	EQT11	4 words)
01673	CHAN	Current DMA channel no.
01674	TBG	I/O address of time-base card
01675	SYSTY	EQT entry address of system TTY
SYSTEM REQUEST PROCESSOR/'EXEC' COMMUNICATION		
01676	RQCNT	No. of request parameters - 1
01677	RQRTN	Return point address
01700	RQP1	Addresses of
01701	RQP2	request
01702	RQP3	parameters
01703	RQP4	(set
01704	RQP5	for
01705	RQP6	maximum
01706	RQP7	of 8
01707	RQP8	parameters)

Octal Location	Contents	Description
ADDRESSES OF SYSTEM LISTS		
01710	DORMT	Address of 'dormant' list
01711	SKEDD	'Schedule' list
01714	SUSP3	'Available memory' list
01715	SUSP4	'Disc allocation' list
01716	SUSP5	'Operator suspend' list
DEFINITION OF EXECUTING PROGRAM ID SEGMENT		
01717	XEQT	ID segment addr. of current program
01720	XLINK	'Linkage'
01721	XTEMP	'Temporary' (5-words)
01726	XPRIO	'Priority' word
01727	XPENT	'Primary entry point'
01730	XSUSP	'Point of suspension'
01731	XA	'A-Register' at suspension
01732	XB	'B-Register' at suspension
01733	XEO	'E and overflow' at suspension
SYSTEM MODULE COMMUNICATION FLAGS		
01734	OPATN	Operator/keyboard attention flag
01735	OPFLG	Operator communication flag
01736	SWAP	RT disc resident swapping flag
01737	DUMMY	I/O address of dummy int. card
01740	IDSDA	Disc addr. of first ID segment
01741	IDSDP	Position within sector
DEFINITION OF MEMORY ALLOCATION BASES		
01742	BPA1	FWA RT disc res. BP link area
01743	BPA2	LWA RT disc res. BP link area
01744	BPA3	FWA bkg disc res. BP link area
01745	LBORG	FWA of resident library area
01746	RTORG	FWA of RT area
01747	RTCOM	Length of RT common area
01750	RTDRA	FWA of RT disc resident area
01751	AVMEM	FWA of system available memory
01752	BKGRG	FWA of background area
01753	BKCOM	Length of background common area
01754	BKDRA	FWA of BKG disc resident area
UTILITY PARAMETERS		
01755	TATLG	Length of track assignment table
01756	TATSD	# of tracks on system disc
01757	SECT2	# sectors/track on LU 2 (system)
01760	SECT3	# sectors/track on LU 3 (aux.)
01761	DSCLB	Disc addr of res lib entry pts
01762	DSCLN	# of res lib entry points
01763	DSCUT	Disc addr of reloc utility programs
01764	DSCUN	# of reloc utility progs
01765	LGOTK	Load-n-go; LU, stg track, # of tracks
01766	LGOC	Current LGO track/sector address
01767	SFCUN	Source file LU and disc address
01770	MPTFL	Memory protect on/off flag (0/1)
01771	EQT12	Address of
01772	EQT13	last 4
01773	EQT14	words of
01774	EQT15	current EQT
01775	FENCE	Memory protect fence address
01777	BKLWA	LWA of memory in background

DISC LAYOUT OF RTE SYSTEM

Figure A-1 on the next page diagrams the allocation of disc space by RTGEN when it creates an RTE System. The bottom portion of the figure shows the difference between loader area in the moving head disc system and fixed head disc system.

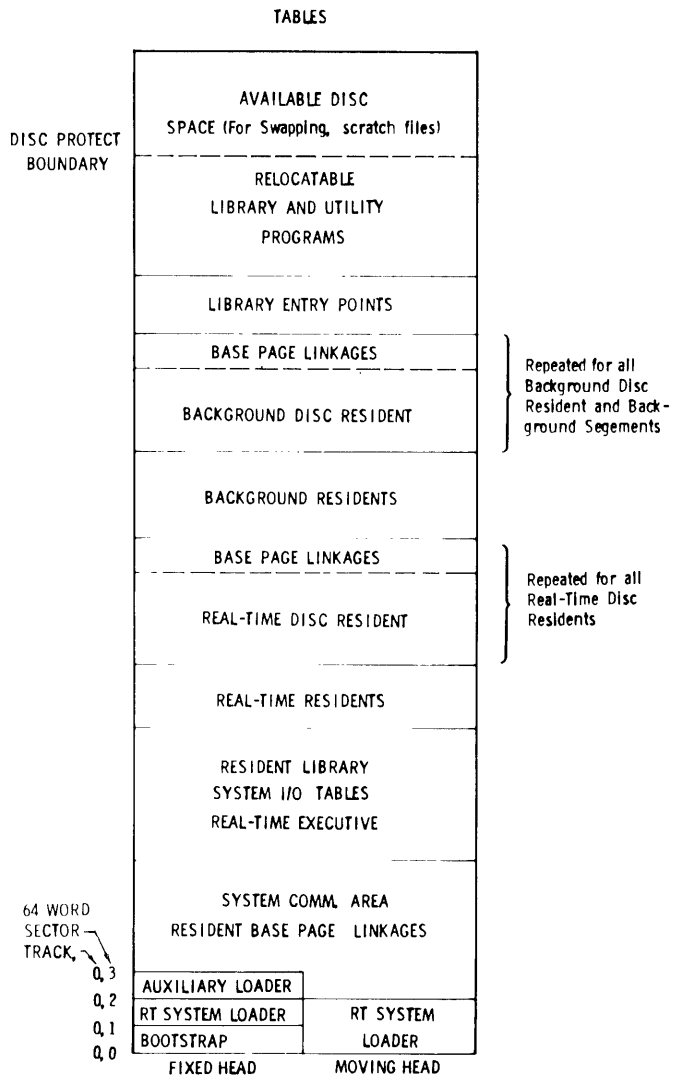


Figure A-1. Disc Allocations in RTE System

PROGRAM ID SEGMENT

Each main user program has a 22- to 28-word ID segment in the resident system area. Certain information in an ID segment is static and is set by RTGEN or the loader. Other information is variable and is maintained by the Executive. In the table below are recorded the names of information stored in the ID segment, whether it is Dynamic (d) or Static (s), and words used.

Table A-2. ID Segments

Word (hp)	Label	Size	S or D	Use
1	XLINK	1	d	List linkage
2-6	XTEMP	5	d	Temporary storage
7	XPRIO	1	s (d on-line)	Priority
8	XPENT	1	s	Primary entry point
9	XSUSP	1	d	Point of suspension
10	XA	1	d	A-Register
11	XB	1	d	B-Register
12	XEO	1	d	E- and O-Registers
13-15	NAME	3	s	Program name and type
16	STAT	1	d	Status
17	TLINK	1	d	Time linkage
18	RESL	1	d	Resolution code
19	TMSEC	1	d	Tens of milliseconds
20	TSEC	1	d	Seconds
21	TMIN	1	d	Minutes
22	THOUR	1	d	Hours
23-26	MEM	4	s	For Disc Resident Programs Only: Mem(1) = Low main; M(3) = Low base; M(2) = Hi main; M(4) = Hi base.
27	DMAN	1	s	Source disc address (L.U./tr/sec)
28	SMAN	1	d	Swap location

(hp) *Addresses of words 1-12 are loaded into base page XEQT Area during execution of a program.*

BBDL LISTING

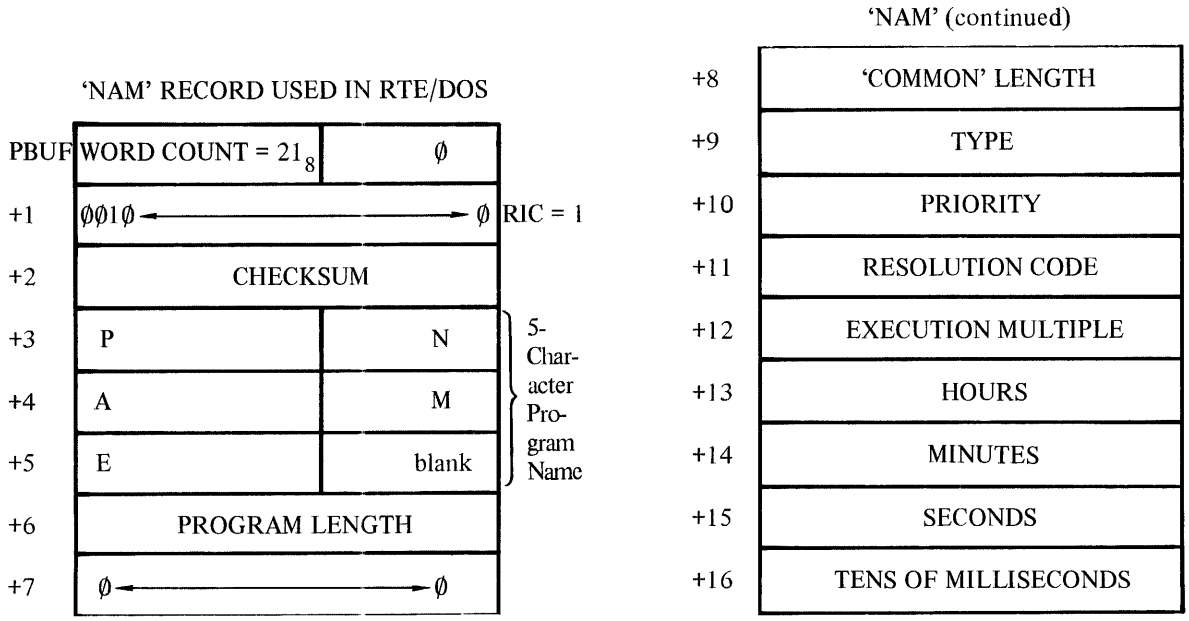
Figure A-2 is an octal listing of the Basic Binary Disc Loader that resides in the protected, highest 64 words of cord. If the loader is destroyed in core, it can be replaced through the switch register using this listing. The operator simply replaces symbolic items with the value appropriate to the configuration.

A	B 0	1	2	3	4	5	6	7
0*m*7700:	107700	002401	063726	006700	017742	007306	027713	002006
0*m*7710:	027703	102077	027700	077754	017742	017742	074000	077757
0*m*7720:	067757	047755	002040	027740	017742	040001	177757	037757
0*m*7730:	000040	037754	027720	017742	054000	027702	102011	027700
0*m*7740:	102055	027700	000000	006600	1037*cc*	1023*cc*	027745	1074*cc*
0*m*7750:	002041	127742	005767	027744	000000	1*z*0100	0200*nn*	000000
0*m*7760:	107700	063756	102606	002700	1026*qq*	001500	102602	063777
0*m*7770:	102702	102602	103706	1027*nn*	067776	074077	024077	177700

Legend: A + B = Memory Address

m = 1 for 8K, 3 for 16K, 5 for 24K, 7 for 32K memory
nn = first disc channel
qq = second disc channel
cc = photoreader or teleprinter address
z = 6 for 8K, 4 for 16K, 2 for 24K, 0 for 32K memory

Figure A-2. BBDL Listing



APPENDIX B
REAL-TIME DISC USAGE

RELATION TO OTHER SOFTWARE

The Hewlett-Packard 2100 series computers are general-purpose, and as such, can utilize other software operating systems when the Real-Time Executive System is inactive. Every computer is shipped with the standard software and documentation appropriate to the system configuration.

In addition, the disc/computer combination may include more than one disc-based software system simultaneously (although only one can execute in core at a time). With the fixed head disc drive, the RTE System, and another software system (not the Time Shared Basic System) can be located on the same platter; and with the moving head disc drive, several RTE Systems and another software system (again not the Time Shared Basic System) can be located on the same platter. When loading into core from the fixed head disc drive with the Basic Binary Disc Loader (BBDL), the operator specifies which system to load by settting switch register bit 0 equal to 0 (for RTE System) or 1 (for another platter based system) after the BBDL halts. (See Section VI.) When loading into core from the moving head disc drive, the proper bootstrap paper tape is loaded via the BBL. Set the switch register to octal 100, push RUN, and the desired system is loaded.

When systems are generated, they must be stored on different areas of the platter. In the case of the fixed head disc drive, this is accomplished by protecting enough tracks to cover both systems. First generate the RTE System onto the lowest tracks, and then generate the other system on ther remaining protected tracks. In this way, RTE does not attempt to write on the other system. Another way to use the computer and fixed head disc drive for two or more software systems is to dump one system on tape using SDUMP (See Section VI) before loading another system from tape.

In the case of the moving head disc drive, the respective domains of each RTE System are defined during RTGEN, such that the system will not overlay into some other system's tracks, or domain. The track map entered during RTGEN defines the domain.

SYSTEM ORGANIZATION

An RTE System can be configured using both fixed head and moving head disc drives. The possible combinations are as follows:

Fixed Head System Disc

1. Fixed head auxiliary disc
2. Fixed head peripheral discs
3. Moving head peripheral discs
4. Moving head auxiliary disc

Option 4 above is not recommended due to the differences in speed between the fixed head and the moving head discs.

Moving Head System Disc

1. Moving head auxiliary disc
2. Moving head peripheral discs
3. Fixed head peripheral discs.

NOTE

The moving head system generator (RTGEN) will not allow a fixed head auxiliary disc drive in a moving head system.

The HP 7900 Disc Drive has nominally 200 tracks (203 maximum) per platter available for use. The two platters are divided as follows:

1,247,232 words per platter
6144 words per track
203 tracks per platter
48 sectors per track
128 words per sector

The RTE moving head driver regards a track as having 96 sectors of 64 words each.

SYSTEM VERSUS PERIPHERAL DISCS

System

The system disc tracks are those for which RTE controls and maintains a continguous track usage table. These are logical units 2 (system), and 3 (auxiliary), and because RTE treates them as system discs the operator can never change their assignment. The system disc tracks are used for swapping, and by the editor, assembler, and compilers for source, load-and-go, and scratch area. They may also be used by user programs for storage.

Peripheral

Peripheral discs are not managed by the RTE System. Track allocation and usage are totally up to the user programs. Peripheral discs are distinguished from system discs by having logical unit numbers greater than 6. The logical unit numbers for peripheral discs must not reference the same physical discs as the system discs.

The RTE System performs the following services for all discs in the system, including peripheral discs:

a. Checks that track and sector information is provided on all calls.

b. Prints a special error message on parity error from a disc transfer (see Appendix F). For peripheral disc transfers, a parity error will cause the transmission log to be sent back to the calling program as -1.

RTE MOVING HEAD DRIVER

The moving head driver can control up to eight platter subchannels, two per drive, with all four drives daisy-chained to a single controller (EQT). These platters are accessed by subchannel numbers 0 through 7. Subchannels are numbered so that the even numbered subchannels are fixed platters and odd numbered subchannels are removable platters.

READ DATA

The driver divides each track into 64 word sectors. Whenever more than 64 words are transmitted, the READ request is fastest when begun on an even sector.

WRITE DATA

WRITE requests starting on an odd sector, or ending in an even sector, require more time; that is, the fastest transfers are WRITE requests that start on an even sector and end in an odd sector. It should be noted that the system always organizes programs and swaps them out in such a way that its transfers always start on an even sector, thus minimizing program load and swap times. The WRITE request data can be checked for recoverability by setting bit 10 in the control word which causes a cyclic check to be made on all data written.

TRACK ACCESS

Each subchannel may contain from 0 to the maximum number of tracks on the physical disc (203 for the HP disc). The first track may also be defined as any track on the platter. These two parameters for each platter are core resident in a track map table. Tracks available to the driver are numbered relative to the first track assigned to the system on each subchannel; thus, if the first available physical track on a subchannel is 10, accesses by the driver to this physical track must specify the logical track number 0.

If a program tries to access a track that is not on the system (i.e., a track number greater than the number of tracks for the given subchannel in the track map table), the driver sets bit 5 in the status word (end of disc) and exits with the transmission log containing the number of tracks on the given subchannel. Normally a program would make a request to an impossible track number once to obtain this information and thereafter stay within the bounds on the subchannel.

In a moving head system the generator generates the track map table. To use a moving head disc as a peripheral or auxiliary in a fixed head system the track map table for the moving head driver must be supplied in the form of a program. This program must have the following format:

```
ASMB, R, B, L
        NAM     $TB31,Ø
        ENT     $TB31
$TB31   DEC     -X
        DEC     FTØ,FT1,FT2,FT3,FT4,FT5,
                FT6,FT7
        DEC     NOØ,NO1,NO2,NO3,NO4,NO5,
                NO6,NO7
        END
```

Where X is the number of 64-word sectors per logical track.

FTØ through FT7 are the number of the first tracks on subchannels Ø through 7 respectively, and NOØ through NO7 are the number of tracks on subchannels Ø through 7 respectively.

Example:

HP7900 Disc with two platters. Tracks Ø – 1ØØ on subchannel Ø, and tracks 2Ø – 8Ø on subchannel 1 inclusive

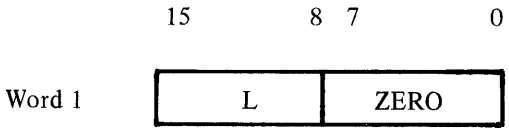
```
ASMB, R, B, L
        NAM     $TB31,Ø
        ENT     $TB31
$TB31   DEC     -96
        DEC     Ø,2Ø,Ø,Ø,Ø,Ø,Ø,Ø
        DEC     1Ø1,61,Ø,Ø,Ø,Ø,Ø,Ø
        END
```

Note that all subchannels are declared, even if they are not part of the system. The Ø entry is required for the track map.

RECORD FORMATS

The source format used for the disc records by the system programs, Edit, Assembler, ALGOL, FORTRAN and FORTRAN IV, is given in Table B-1. All records are packed ignoring sector boundaries. Binary records are packed directly onto the disc. After an END record, a zero word is written and the rest of the sector is skipped. If this zero word is the first word of a sector, it is not written. Binary files are always contiguous so a code word is not required.

Table B-1. Source Format

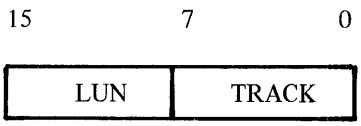


Where L is the record length in words excluding Word 1

If Word 1 = ∅ then end of TAPE
If Word 1 = -1 then end of FILE

Odd characters are padded with blanks to make a full word. The last word on any given track in a multi-track file is a code word that points to the next track in the file.

Code Word Format



Where LUN is either 2 (system) or 3 (auxiliary) depending on which platter the track is on.

APPENDIX C
SAMPLE RTGEN FIXED HEAD DISC

```
CLEAR SR  PUSH RUN


FH DISC CHNL?
23

SYS DISC SIZE?
32

START SCRATCH?
4

NO. PROTECTED?
8

# SECTORS/TRACK?
128

AUX DISC SIZE?
0

TBG CHNL?
12

PRIV. INT. CARD ADDR?
11

SWAPPING?
YES

LWA MEM?
37677

PRGM INPT?
PT

LIBR INPT?
PT

PRAM INPT?
TY

TURN OFF DISC PROTECT - PRESS RUN
```

```
*EOT
*EOT
*EOT
*EOT
*EOT

 NO UNDEF EXTS


PARAMETERS

START,82,3
/E


# OF BLANK ID SEGMENTS?
5

FWA BP LINKAGE?
30


SYSTEM

EXEC (00)   02000

SCHED(00)   03470

RTIOC(00)   10700

DVR00(00)   14076

DVR01(00)   14661

DVR02(00)   15213

DVR30(00)   15433

DVR12(00)   15775

DVR31(00)   16631
  $IB31     17747
```

```
DVR55(00)   17770
  $JP55     20247

DVR56(00)   20251

DVR76(00)   20503

POWER(00)   20675

BP LINKAGE 00420

* EQUIPMENT TABLE ENTRY

23,DVR30,D

25,DVR31,D

10,DVR55

13,DVR76,T
 T =
100

16,DVR00,B

17,DVR01,T
 T =
10

20,DVR02,B,T
 T =
40

21,DVR12

22,DVR56,D

//E

* DEVICE REFERENCE TABLE

 1 =EQT #?
5

 2 =EQT #?
1

 3 =EQT #?
0

 4 =EQT #?
7

 5 =EQT #?
6

 6 =EQT #?
8

 7 =EQT #?
2,0

 8 =EQT #?
2,1

 9 =EQT #?
4

10 =EQT #?
3

11 =EQT #?
9

12 =EQT #?
/E

* INTERRUPT TABLE

4,ENT,POWER

10,EQT,3

13,EQT,4

14,ABS,106714

15,ABS,106715

16,EQT,5

17,EQT,6

20,EQT,7

21,EQT,8

22,EQT,9

23,EQT,1

24,EQT,1

25,EQT,2

26,EQT,2

/E

LIBRARY

(NONE)

RT RESIDENTS

(NONE)
```

```
R1 DISC RESIDENTS

START(03)  22326

BP LINKAGE 00421

CHANGE BP LINKAGE?
1100

FWA SY MEM 22340

CHANGE FWA SYS AV MEM?
25400

BG BOUNDARY?
26000

BG RESIDENTS

(NONE)

BG DISC RESIDENTS

EDIT (99)  26000
  SREAD    30411
  %WRIS    31144
  .OPSY    31434

LOADR(99)  26000

ASMB (99)  26000

ASMBD(99)  32631

ASMB1(99)  32631

ASMB2(99)  32631

ASMB3(99)  32631

ASMB4(99)  32631

ASMB5(99)  32631

FTN  (99)  26000

FTN01(99)  26720
  SREAD    35032
  .OPSY    35565

FTN02(99)  26720

FTN03(99)  26720

FTN04(99)  26720
  %WRIT    33306
  FADSB    34004
  .OPSY    34162
  .FLUN    34222
  .PACK    34243
  .ZRLB    34357
  .EAU.    34420

BP LINKAGE 01561

*SYSTEM STORED ON DISC
LAST SYS DISC ADDR: TRK 06 SEC 014(10)
```

INPUT/OUTPUT CONFIGURATION WORKSHEET

RTGEN NUMBER _____ **DATE** _____ **PREPARED BY** _____

SC / SUB	10	11	12	13	14	15	16	17	20	21	22	23	24	25	26	0	1
I/O INTERFACE CARD NAME	12566 MICROCIRCUIT 2312 SUBSYSTEM	12620 PRIV. INT	TIME BASE GEN.	← 2320 → 12604 DSI CARD	12535 SCANNER CARD	12567 DVM CARD	TELEPRINTER I/O	TAPE READER I/O	TAPE PUNCH I/O	LINE PRINTER I/O	12566 MICROCIRCUIT 2311 SUBSYSTEM	F.H. DISC	CONTROLLER	M.H. DISC	CONTROLLER	SUBCHANNEL	SUBCHANNEL
STD. LOGICAL UNIT NOS.																	
1 SYS. TTY							X										
2 SYS. MASS STORAGE												X					
3 AUX MASS STORAGE																	
4 PUNCH OUTPUT									X								
5 INPUT								X									
6 LIST OUTPUT										X							
7_{10} TO 63_{10}	10	-	-	9	-	-	-	-	-	-	11	-	-	-	-	7	8
DVR IDENT. (DVRxx)	55	-	-	76	I	I	00	01	02	12	22	30	I	31	I		
DMA REQUIRED (D)											D	D		D			
EQT ENTRY NO.	3	-	-	4	-	-	5	6	7	8	9	1		2			
BUFFERED OUTPUT (B)							B		B								
TIME-OUT (T)				T				T	T								
OCTAL SELECT CODE / SUBCHANNEL	10	11	12	13	14	15	16	17	20	21	22	23	24	25	26	0	1

SYSTEM DISC WORKSHEET

FIXED HEAD SYSTEM DISC

DRUM/DISC MEMORY CHARACTERISTICS

DRUM	DISC	# LOGICAL TRACKS	SECTORS PER TRACK
-	2766	32	128
2773	-	48	
2773-003	2766-002	64	
2774	2766-003	96	
2774-003	2766-004	128	
-	2770	32	90
-	2770-01	64	
-	2771	64	
-	2771-01	128	

DIODES 12606 I/O	# LOGICAL PROT. TRKS	DIODES 12610 I/O
NONE	1	CR1, 2
CR1	2	CR1-3
CR1, 2	4	CR1-4
CR1-3	8	CR1-5
CR1-4	16	CR1-6
CR1-5	32	CR1-7
CR1-6	64	CR1-8
CR1-7	128	CR1-9
-	192	CR1-10

SYSTEM (LU2)

NO. OF TRACKS AVAILABLE 32
START SCRATCH 4
NO. OF PROTECTED TRACKS 8
SECTORS/TRACK 128

AUXILIARY (LU3)

NO. OF TRACKS AVAILABLE _____
SECTORS/TRACK _____

MOVING HEAD ~~SYSTEM~~ DISC (PERIPHERAL)

MOVING HEAD DRIVE	MOVING HEAD DRIVE	MOVING HEAD DRIVE	MOVING HEAD DRIVE
LU7 SUBCHANNEL 0	SUBCHANNEL 2	SUBCHANNEL 4	SUBCHANNEL 6
FIXED			
NO0 NO. OF TRACKS AVAILABLE 101	NO. OF TRACKS AVAILABLE _____	NO. OF TRACKS AVAILABLE _____	NO. OF TRACKS AVAILABLE _____
FT0 FIRST TRACK 0	FIRST TRACK _____	FIRST TRACK _____	FIRST TRACK _____
LU8 SUBCHANNEL 1	SUBCHANNEL 3	SUBCHANNEL 5	SUBCHANNEL 7
REMOVABLE			
NO1 NO. OF TRACKS AVAILABLE 61	NO. OF TRACKS AVAILABLE _____	NO. OF TRACKS AVAILABLE _____	NO. OF TRACKS AVAILABLE _____
FT1 FIRST TRACK 20	FIRST TRACK _____	FIRST TRACK _____	FIRST TRACK _____

SYSTEM SUBCHANNEL NUMBER _____ SCRATCH SUBCHANNEL NUMBER _____
AUXILIARY SUBCHANNEL NUMBER _____ START SCRATCH (I.E., 1ST TRACK=0) _____

NOTE:

IF A MOVING HEAD DISC DRIVE IS USED AS A PERIPHERAL, THIS PORTION CAN BE USED FOR TRACK ASSIGNMENT. IN THIS CASE, NO SUBCHANNEL ASSIGNMENTS OF THE PERIPHERAL WILL BE ASSIGNED TO LU2 OR LU3.

RTE CONFIGURATION WORKSHEET

INITIALIZATION PHASE

MOVING HEAD INITIALIZATION

MH. DISC CHNL?

TRKS, FIRST TRK ON SUBCHNL:
0?
1?
2?
3?
4?
5?
6?
7?

SYSTEM SUBCHNL?

AUX DISC (YES OR NO)?

AUX DISC SUBCHNL?

SCRATCH SUBCHNL?

START SCRATCH?

128 WORD SECTORS/TRACK?

TBG CHNL?

PRIV. INT. CARD ADDR.

SWAPPING?

LWA MEM?

PRGM INPT?

LIBR INPT?

PRAM INPT?

INITIALIZE SUBCHNL:n

PUNCH BOOT?
YES

FIXED HEAD INITIALIZATION

FH. DISC CHNL?
23
SYS DISC SIZE?
32
START SCRATCH?
4
NO. PROTECTED?
8
SECTORS/TRACK?
128
AUX DISC SIZE?
0
SECTORS/TRACK?
—
TBG CHNL?
12
PRIV. INT. CARD ADDR?
11
SWAPPING?
YES
LWA MEM?
37677
PRGM INPT?
PT
LIBR INPT?
PT
PRAM INPT?
TY

PROGRAM INPUT PHASE

EXEC CONTROL (EXEC)
SCHEDULER (SCHED)
I/O CONTROL (RT10C)
I/O DRIVERS
USER'S SYSTEM PROGRAMS
FOREGROUND CORE RESIDENT PROGRAMS
FOREGROUND DISC RESIDENT PROGRAMS
BACKGROUND CORE RESIDENT PROGRAMS
BACKGROUND DISC RESIDENT PROGRAMS
AND THEIR RESPECTIVE SEGMENTS
LIBRARY PROGRAMS
UTILITY PROGRAMS
SUBROUTINES

NO UNDEF EXTS

PARAMETER INPUT PHASE

NAME, TYPE [, PRIORITY] [, EXECUTION INTERVAL]

START, 82, 3

/E

OF BLANK ID SEGMENTS?
5

DISC LOADING PHASE

FWA BP LINKAGE
30

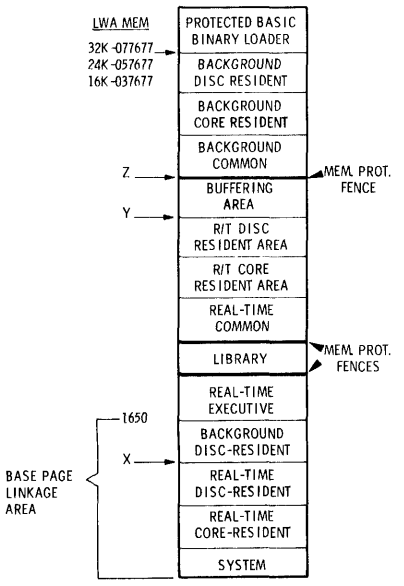
* EQUIPMENT TABLE ENTRY
(1) 23, DVR30, D T = —
(2) 25, DVR31, D T = —
(3) 10, DVR55 T = —
(4) 13, DVR76, T T = 100
(5) 16, DVR00, B T = —
(6) 17, DVR01, T T = 10
(7) 20, DVR02, B, T T = 40
(8) 21, DVR12 T = —
(9) 22, DVR56, D T = —
/E

* DEVICE REFERENCE TABLE
1 - EQT #? 5 (SYSTEM TELEPRINTER)
2 - EQT #? 1 (SYSTEM MASS STORAGE)
3 - EQT #? 0 (AUXILIARY MASS STORAGE)
4 - EQT #? 7 (STANDARD PUNCH UNIT)
5 - EQT #? 6 (STANDARD INPUT UNIT)
6 - EQT #? 8 (STANDARD LIST UNIT)
7 - EQT #? 20
8 - EQT #? 21
9 - EQT #? 4
10 - EQT #? 3
11 - EQT #? 9
12 - EQT #?
13 - EQT #?
14 - EQT #?
15 - EQT #?
16 - EQT #?
17 - EQT #?
/E

* INTERRUPT TABLE
4, ENT, POWER
10, EQT, 3
13, EQT, 4
14, ABS, 106714
15, ABS, 106715
16, EQT, 5
17, EQT, 6
20, EQT, 7
21, EQT, 8
22, EQT, 9
23, EQT, 1
24, EQT, 1
25, EQT, 2
26, EQT, 2
/E

BP LINKAGE 00412
X CHANGE BP LINKAGE?
1100
FWA SY MEM 22340
Y CHANGE FWA AV MEM?
25400
Z BG BOUNDARY?
26000
* SYSTEM STORED ON DISC
TRACK 06 SECTOR 14 (10)

RTE
CORE ALLOCATIONS



```
CLEAR SR  PUSH RUN

MH DISC CHNL?
13

# TRKS, FIRST TRK ON SUBCHNL:
  0?
0
  1?
0
  2?
50,0
  3?
200,0
  4?
100,0
  5?
50,50
  6?
/E

SYSTEM SUBCHNL?
5

AUX DISC (YES OR NO)?
YES

AUX DISC SUBCHNL?
4

SCRATCH SUBCHANL?
4

START SCRATCH?
50

# 128 WORD SECTORS/TRACK?
48

IBG CHNL?
12

PRIV. INT. CARD ADDR?
11

SWAPPING?
YES

LWA MEM?
37677

PRGM INPT?
PT

LIBR INPT?
PT

PRAM INPT?
TY

INITILIZE SUBCHNL:
  2?
NO
  3?
NO
TURN OFF DISC PROTECT - PRESS RUN

PUNCH BOOT?
YES

PUNCH BOOT?
YES

PUNCH BOOT?
NO


*EOT
*EOT
*EOT
*EOT
*EOT

 NO UNDEF EXTS
```

```
PARAMETERS

START,81,4
FSWP,2,3
/E


# OF BLANK ID SEGMENTS?
4

FWA BP LINKAGE?
26


SYSTEM

EXEC (00)  02000

SCHED(00)  03470

RTIOC(00)  10700

DVR00(00)  14076

DVR01(00)  14661

DVR02(00)  15213

DVR30(00)  15433

DVR31(00)  15775

DVR55(00)  17105
  $JP55    17364

DVR77(00)  17366
  $LK77    17705

BP LINKAGE 00410

* EQUIPMENT TABLE ENTRY

13,DVR31,D,T
 T =
30

10,DVR55

21,DVR77,T
 T =
40

24,DVR30,D,T
 T =
30

15,DVR00,B,T
 I =
1000

16,DVR00,B

17,DVR01,T
 T =
10

20,DVR02,B,T
 T =
40

/E


* DEVICE REFERENCE TABLE

 1 = EQT #?
6

 2 = EQT #?
1,5

 3 = EQT #?
1,4

 4 = EQT #?
8

 5 = EQT #?
7

 6 = EQT #?
5

 7 = EQT #?
3

 8 = EQT #?
2

 9 = EQT #?
4

10 = EQT #?
1,2

11 = EQT #?
1,3

12 = EQT #?
/E
```

```
* INTERRUPT TABLE

10,EQT,2

13,EQT,1

14,EQT,1

15,EQT,5

16,EQT,6

17,EQT,7

20,EQT,8

21,EQT,3

22,EQT,3

23,EQT,3

24,EQT,4

25,EQT,4

/E


LIBRARY


(NONE)

RT RESIDENTS

START(04)  21531

BP LINKAGE 00411

RT DISC RESIDENTS

SWP1 (99)  21612

FSWP (03)  21612

BP LINKAGE 00411

CHANGE BP LINKAGE?
1000

FWA SY MEM 21617

CHANGE FWA SYS AV MEM?
25300

BG BOUNDRY?
26000

BG RESIDENTS

(NONE)

BG DISC RESIDENTS

EDIT (99)  26000
  SREAD    30411
  %WRIS    31144
  .OPSY    31434

LOADR(99)  26000

ASMB (99)  26000

ASMBD(99)  32631

ASMB1(99)  32631

ASMB2(99)  32631

ASMB3(99)  32631

ASMB4(99)  32631

ASMB5(99)  32631

FTN  (99)  26000

FTN01(99)  26720
  SREAD    35032
  .OPSY    35563

FTN02(99)  26720

FTN03(99)  26720

FTN04(99)  26720
  %WRIT    33306
  FADSB    34004
  .OPSY    34162
  .FLUN    34222
  .PACK    34243
  .ZRLB    34357
  .EAU.    34420

BP LINKAGE 01461

SYSTEM STORED ON DISC
SYS SIZE: 08 TRKS, 009 SECS(10)
```

INPUT/OUTPUT CONFIGURATION WORKSHEET

RTGEN NUMBER _____ DATE _____ PREPARED BY _____

STD. LOGICAL UNIT NOS.	SC/SUB	10	11	12	13	14	0	1	2	3	4	5	15	16	17	20	21	22	23	24	25
	I/O INTERFACE CARD NAME	12566 MICROCIRCUIT 2312 SUBSYSTEM	12620 PRIV. INT.	TIME BASE GEN.	M.H. DISC CONTROLLER	SUBCHANNEL					SUBCHANNEL	TELEPRINTER I/O	TELEPRINTER I/O	TAPE READER I/O	TAPE PUNCH I/O	12604 DSI CARD	←2323→ 12576 SCANNER CARD	12576-01 DVM CARD	F.H. DISC CONTROLLER	
1	SYS. TTY													X							
2	SYS. MASS STORAGE											X									
3	AUX MASS STORAGE										X										
4	PUNCH OUTPUT															X					
5	INPUT														X						
6	LIST OUTPUT												X								
7_{10} TO 63_{10}		8	-	-	-	-	-	-	10	11	-	-	-	-	-	-	7	-	-	9	
DVR IDENT. (DVRxx)		55	-	-	31	I	-	-	-	-	-	-	00	00	01	02	77	I	I	30	I
DMA REQUIRED (D)																				D	
EQT ENTRY NO.		2	-	-	1		-	-	-	-	-	-	5	6	7	8	3	-	-	4	
BUFFERED OUTPUT (B)													B	B		B					
TIME-OUT (T)					T								T		T	T	T			T	
OCTAL SELECT CODE	SUBCHANNEL	10	11	12	13	14	0	1	2	3	4	5	15	16	17	20	21	22	23	24	25

SYSTEM DISC WORKSHEET

PERIPHERAL

FIXED HEAD ~~SYSTEM~~ DISC

DRUM/DISC MEMORY CHARACTERISTICS

DRUM	DISC	# LOGICAL TRACKS	SECTORS PER TRACK
-	2766	32	128
2773	-	48	
2773-003	2766-002	64	
2774	2766-003	96	
2774-003	2766-004	128	
-	2770	32	90
-	2770-01	64	
-	2771	64	
-	2771-01	128	

DIODES 12606 I/O	# LOGICAL PROT. TRKS	DIODES 12610 I/O
NONE	1	CR1, 2
CR1	2	CR1-3
CR1, 2	4	CR1-4
CR1-3	8	CR1-5
CR1-4	16	CR1-6
CR1-5	32	CR1-7
CR1-6	64	CR1-8
CR1-7	128	CR1-9
-	192	CR1-10

SYSTEM (LU2)

NO. OF TRACKS AVAILABLE 32
START SCRATCH -
NO. OF PROTECTED TRACKS -
SECTORS/TRACK 128

AUXILIARY (LU3)

NO. OF TRACKS AVAILABLE
SECTORS/TRACK

MOVING HEAD SYSTEM DISC

MOVING HEAD DRIVE	MOVING HEAD DRIVE	MOVING HEAD DRIVE	MOVING HEAD DRIVE
SUBCHANNEL 0	LU10 SUBCHANNEL 2	LU3 SUBCHANNEL 4	SUBCHANNEL 6
FIXED			
NO. OF TRACKS AVAILABLE 0	NO. OF TRACKS AVAILABLE 50	NO. OF TRACKS AVAILABLE 100	NO. OF TRACKS AVAILABLE
FIRST TRACK	FIRST TRACK 0	FIRST TRACK 0	FIRST TRACK
SUBCHANNEL 1	LU11 SUBCHANNEL 3	LU2 SUBCHANNEL 5	SUBCHANNEL 7
REMOVABLE			
NO. OF TRACKS AVAILABLE 0	NO. OF TRACKS AVAILABLE 200	NO. OF TRACKS AVAILABLE 50	NO. OF TRACKS AVAILABLE
FIRST TRACK	FIRST TRACK 0	FIRST TRACK 50	FIRST TRACK

SYSTEM SUBCHANNEL NUMBER 5
AUXILIARY SUBCHANNEL NUMBER 4
SCRATCH SUBCHANNEL NUMBER 4
START SCRATCH (I.E., 1ST TRACK = 0) 50

NOTE:

IF A MOVING HEAD DISC DRIVE IS USED AS A PERIPHERAL, THIS PORTION CAN BE USED FOR TRACK ASSIGNMENT. IN THIS CASE, NO SUBCHANNEL ASSIGNMENTS OF THE PERIPHERAL WILL BE ASSIGNED TO LU2 OR LU3.

RTE CONFIGURATION WORKSHEET

INITIALIZATION PHASE

MOVING HEAD INITIALIZATION

MH. DISC CHNL?
13
TRKS, FIRST TRK ON SUBCHNL:
0?
0,
1?
0,
2?
50,0
3?
200,0
4?
100,0
5?
50,50
6?
,
7?
,
SYSTEM SUBCHNL?
3
AUX DISC (YES OR NO)?
YES
AUX DISC SUBCHNL?
4
SCRATCH SUBCHNL?
4
START SCRATCH?
50
128 WORD SECTORS/TRACK?
48
TBG CHNL?
12
PRIV. INT. CARD ADDR.
11
SWAPPING?
YES
LWA MEM?
37677
PRGM INPT?
PT
LIBR INPT?
PT
PRAM INPT?
TY

INITIALIZE SUBCHNL:n
2-NO
3-NO

PUNCH BOOT?
YES ,YES ,NO

FIXED HEAD INITIALIZATION

FH. DISC CHNL?
SYS DISC SIZE?
START SCRATCH?
NO. PROTECTED?
SECTORS/TRACK?
AUX DISC SIZE?
SECTORS/TRACK?
TBG CHNL?
PRIV. INT. CARD ADDR?
SWAPPING?
LWA MEM?
PRGM INPT?
LIBR INPT?
PRAM INPT?

PROGRAM INPUT PHASE

EXEC CONTROL (EXEC)
SCHEDULER (SCHED)
I/O CONTROL (RT10C)
I/O DRIVERS
USER'S SYSTEM PROGRAMS
FOREGROUND CORE RESIDENT PROGRAMS
FOREGROUND DISC RESIDENT PROGRAMS
BACKGROUND CORE RESIDENT PROGRAMS
BACKGROUND DISC RESIDENT PROGRAMS
AND THEIR RESPECTIVE SEGMENTS
LIBRARY PROGRAMS
UTILITY PROGRAMS
SUBROUTINES

NO UNDEF EXTS

PARAMETER INPUT PHASE

NAME, TYPE [, PRIORITY] [, EXECUTION INTERVAL]

START, 81, 4
FSWP, 2, 3
/E
OF BLANK ID SEGMENTS?
4

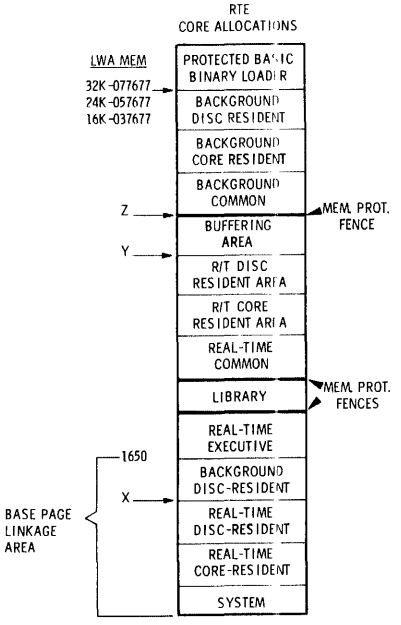
DISC LOADING PHASE

FWA BP LINKAGE
26
* EQUIPMENT TABLE ENTRY
(1) 13,DVR 31,D,T, T-30
(2) 10,DVR 55, , , T-
(3) 21,DVR 77,T, , T-40
(4) 24,DVR 30,D,T, T-30
(5) 15,DVR 00,B,T, T-1000
(6) 16,DVR 00,B, , T-
(7) 17,DVR 01,T, , T-10
(8) 20,DVR 02,B,T, T-40
(9) ,DVR , , , T-
/E

* DEVICE REFERENCE TABLE
1 - EQT #? 6 (SYSTEM TELEPRINTER)
2 - EQT #? 1,5 (SYSTEM MASS STORAGE)
3 - EQT #? 1,4 (AUXILIARY MASS STORAGE)
4 - EQT #? 8 (STANDARD PUNCH UNIT)
5 - EQT #? 7 (STANDARD INPUT UNIT)
6 - EQT #? 5 (STANDARD LIST UNIT)
7 - EQT #? 3
8 - EQT #? 2
9 - EQT #? 4
10 - EQT #? 1,2
11 - EQT #? 1,3
12 - EQT #?
13 - EQT #?
14 - EQT #?
15 - EQT #?
16 - EQT #?
17 - EQT #?
/E

* INTERRUPT TABLE
10, EQT, 2
13, EQT, 1
14, EQT, 1
15, EQT, 5
16, EQT, 6
17, EQT, 7
20, EQT, 8
21, EQT, 3
22, EQT, 3
23, EQT, 3
24, EQT, 4
25, EQT, 4
/E

BP LINKAGE 00411
X CHANGE BP LINKAGE?
1000
FWA SY MEM 21617
Y CHANGE FWA AV MEM?
25300
Z BG BOUNDARY?
26000
* SYSTEM STORED ON DISC
TRACK 08 SECTOR 009 (10)



APPENDIX D
SUMMARY OF EXEC CALLS

Consult Section III for the complete details on each EXEC call. The general format of an EXEC call in Assembler Language is:

```
        EXT   EXEC      Used to link program to RTE
        :
        JSB   EXEC      Transfer control to RTE
        DEF   *+ n + 1  Defines point of return from
                        RTE, n is number of parameters;
                        must be direct address
        DEF   p1        } Define addresses of parameters
        :               } which may occur anywhere in
        DEF   pn        } program; may be multi-level in-
                        } direct
        return point    Continue execution of program
        .
        :
        .
        :
p1      ---             }
.                       } Actual parameter values
.                       }
pn      ---             }
```

For each EXEC call, this appendix includes only the parameters (*p1* through *pn* in the format above) of the Assembler Language calling sequence.

READ/WRITE: Transfers input or output.

ICODE DEC	1 (read) or 2 (write)
ICNWD OCT	See Section III for control information.
IBUFR BSS	*n* (*n*-word buffer)
IBUFL DEC	*n* or *-2n* (buffer length, words (+), characters (-)
IPRM1 DEC	*p* optional parameter. Used for disc track in disc call.
IPRM2 DEC	*q* optional parameter. Used for disc sector in disc call.

I/O CONTROL: Carry out control operations

ICODE DEC 3	
ICNWD OCT	See Section III for control information.
IPRAM DEC	*n* (Optional parameter required by some CONWDs)

I/O STATUS: Request device status.

ICODE DEC 13	
ICNWD DEC *n*	Logical unit number
ISTA1 NOP	Word 5 of EQT entry returned here
ISTA2 NOP	Optional parameter for word 4 of EQT

DISC ALLOCATION: Request allocation of contiguous tracks.

ICODE DEC 4 or 15	4 = Allocate track to program, or 15 = allocate track globally.
ITRAK DEC *n*	Number of contiguous tracks desired. If bit 15 = 1, do not suspend until available.
ISTRK NOP	Starting track returned here, or -1, not available.
IDISC NOP	Disc logical unit returned here.
ISECT NOP	Number of 64 word sectors returned here.

DISC RELEASE: Release some disc tracks assigned to the program.

ICODE DEC 5 or 16	5 = Release program's tracks, or 16 = release global tracks.
ITRAK DEC *n*	If = -1, release all program tracks. If = *n*, the number of contiguous tracks starting at ISTRK.
ISTRK NOP	Starting track number.
IDISC NOP	Logical unit.

PROGRAM COMPLETION: Signal end of program.

```
ICODE  DEC 6
```

PROGRAM SUSPEND: Suspend calling program.

```
ICODE  DEC 7
```

PROGRAM SEGMENT LOAD: Load segment of calling program.

```
ICODE  DEC 8
INAME  ASC 3,xxxxx      xxxxx is segment name
```

PROGRAM SCHEDULE: To schedule another program.

```
ICODE  DEC 9 or 10      9 (waiting) or 10 (no waiting)
INAME  ASC 3,xxxxx      xxxxx is the program name
p1     . . .  }
p2            }
         .    }  Up to five optional parameters.
         .    }
p5       . .  }
```

TIME REQUEST: Request the 24-hour time and day.

```
ICODE  DEC 11
ITIME  BSS 5            Time values: tens of milliseconds, seconds, minutes, hours, day, returned in that order
```

EXECUTION TIME: Schedule a program by time.

Initial Offset Version

```
ICODE  DEC 12
IPROG  DEC 0            Schedule calling program, or
       ASC 3,xxxxx      Schedule xxxxx
IRESL  DEC x            Resolution code
MTPLE  DEC y            Execution multiple
IOFST  DEC-z            z (units set by x) equals the initial offset.
```

Absolute Start-Time Version

```
ICODE  DEC 12
IPROG  DEC 0            Schedule calling program, or
       ASC 3,xxxxx      Schedule xxxxx
IRESL  DEC x            Resolution code
MTPLE  DEC y            Execution multiple
IHRS   DEC a  }
MINS   DEC b  }  Defines absolute start-time
ISECS  DEC c  }
MSECS  DEC d  }
```

APPENDIX E
SUMMARY OF ERROR MESSAGES

OPERATOR REQUEST ERROR MESSAGES

When an operator request is in error, RTE rejects the request and prints one of the messages below. The operator enters the request again, correctly.

Message	Meaning
OP CODE ERROR	Illegal operator request word.
NO SUCH PROG	The *name* given is not a main program in the system.
INPUT ERROR	A parameter is illegal.

EXEC CALL ERROR MESSAGES

When RTE discovers an error in an EXEC call, it terminates the program, releases any disc tracks assigned to the program, prints an error message on the operator console, and proceeds to execute the next program in the scheduled list.

When RTE aborts a program, it prints this message:

name ABORTED

When a memory project violation occurs that is not an EXEC call, this message is printed:

MP *name address* (*address* is the location that caused the violation.)

When an EXEC call contains an illegal request code, this message is printed:

RQ *name address* (*address* is the location that made the illegal call.)

The general error format, for other errors, is:

type name address

where *type* is a 4-character error code,
name is the program that made the call, and
address is the location of the call (equal to the exit point if the error is detected after the program suspends).

ERROR CODES FOR SCHEDULE CALLS

SCØ1 = Missing parameter,

SCØ2 = Illegal parameter,

SCØ3 = Program cannot be scheduled,

SCØ3 INT *xx* occurs when an external interrupt attempts to schedule a program that is already scheduled. RTE ignores the interrupt and returns to the point of interruption.

SCØ5 = Program given is not defined.

SCØ6 = No resolution code in EXECUTION TIME EXEC call.

ERROR CODES FOR DISC ALLOCATION CALLS

DRØ1= Insufficient number of parameters,

DRØ2= Number of tracks is zero, >255, or <zero; illegal logical unit; or number of tracks to release is zero or negative.

DRØ3= Attempt to release track assigned to another program.

ERROR CODES FOR I/O CALLS

IOØ1 = Not enough parameters,

IOØ2 = Illegal logical unit,

IOØ3 = Logical unit not assigned,

IOØ4 = Illegal user buffer,

IOØ5 = Illegal disc track or sector,

IOØ6 = Reference to a protected track; or using load-and-go before assigning load-and-go tracks (see LG, Section II),

IOØ8 = Disc transfer longer than track, and

IOØ9 = Overflow of load-and-go area.

INPUT/OUTPUT ERROR MESSAGES

ILLEGAL INTERRUPTS

When an illegal interrupt occurs, RTE prints this message:

ILL INT *xx*

Where *xx* is the octal channel number.

RTE clears the interrupt flag on the channel and returns to the point of interruption.

EQUIPMENT MESSAGES

Message	Meaning
I/O ERR ET EQT #*n*	End-of-tape condition on device #*n*. Correct the condition and set device UP
I/O ERR TO EQT #*n*	Device #*n* has timed-out. Examine device. Correct problem and set device UP.
I/O ERR NR EQT #*n*	Device #*n* is not ready. Make ready and set device UP
I/O ERROR PE EQT #*n*	Parity error in data transmission from device #*n*. Examine device.
TR *nnnn* EQT *mm*, U*pp* S(or U)	Irrecoverable disc transfer parity error. If to system or auxiliary disc: a. If user request (U), then program is abnormally terminated and track is made unavailable for further operations. If the user request was an on-line modification with the RTE loader, the parity error could be the result of failing to turn off the hardware disc protect switch. The loader should be executed again with the protect switch off. b. If system request (S), the program transfer terminates. If user request to peripheral disc, a transmission log of -1 is returned to the calling program.

FORTRAN COMPILER ERROR MESSAGES

When the end of the source tape is encountered the following message is output to the system teleprinter.

I/O ERR ET EQT #*n*

EQT #*n* is unavailable (see DN, Section II) until the operator declares it up:

UP,*n*

Thus, more than one source tape can be compiled into one program. The next source tape is loaded each time the compiler detects an end-of-tape.

At the end of compilation, the following message is printed.

$END, FTN

and RTE executes the next scheduled program.

Two messages are I/O error messages generated by RTE when FTN attempts to write on the load-and-go tracks (RTE aborts FTN).

IOØ6

IOØ9

IOØ6 means that the load-and-go tracks were not defined (i.e., by LG), and IOØ9 means that the load-and-go tracks overflowed. The operator must define more load-and-go tracks with LG and start compilation over again.

The compiler terminates abnormally if

a. No source file is declared by LS, although logical unit 2 is given for input. Compiler error E-ØØ19 (FTN2), or ERROR Ø5 (FTN4) is printed on the list device.

b. The symbol table overflows. Compiler error E-ØØ14 (FTN2), or ERROR Ø3 (FTN4) is printed on the list device. $END, FTN does not appear after the error message using FTN2, but does appear when using FTN4.

ALGOL MESSAGES

When the end of the source tape is encountered the following message is output on the system teleprinter.

I/O ERR ET EQT #*n*

EQT #*n* is unavailable (see DN, Section II) until the operator declares it up:

UP,*n*

If source input is indicated to be from the disc and the source pointer is not set, the diagnostic

NO SOURCE

is printed on the system teleprinter and compilation ceases.

At the end of a program, a program-termination request is made to the Executive. No message is printed. In case of a PAUSE statement, the following message is printed.

name: PAUSE *xxxx*

where:

name = program name

xxxx = a number which has no significance.

Execution is then suspended. To restart the program type:

GO,*name*[,*p1,p2,p3,p4,p5*]

ASSEMBLER ERROR MESSAGES

When the end of the source tape is encountered the following message is output to the system teleprinter:

I/O ERR ET EQT #*n*

EQT #*n* is unavailable (see DN, Section II) until the operator declares it up:

UP, *n*

More than one source tape can be assembled into one program. The next source tape is loaded by placing it in the input device and declaring the device UP, *n* as shown above.

At the end of compilation, the following message is printed:

$END ASMB

and RTE executes the next scheduled program.

If another pass of the source program is required, the following message appears at the end of pass one.

$END ASMB PASS

The operator must replace the program in the input device and type

GO, ASMB

If an error is found in the Assembler control statement, the following message appears:

$END ASMB CS

the current assembly aborts.

If an end-of-file condition occurs before an END statement is found, the teleprinter signals:

$END ASMB XEND

The current assembly aborts.

If source input for logical unit 2 (disc) is requested, but no file has been declared (see LS, Section II), the teleprinter signals:

$END ASMB NPRG

RTE generates two messages when ASMB attempts to write on the load-and-go tracks (RTE ASMB aborts).

IOØ6

IOØ9

IOØ6 means than the load-and-go tracks were not defined by an LG operator request, and IOØ9 means that the load-and-go tracks have overflowed. The operator must define more load-and-go tracks with LG and start compilation over again.

The next message is associated with each error diagnostic printed during pass 1.

nnn

nnn is the "tape" number where the error (reported on the next line of the listing) occurred. A program may consist of more than one tape. The tape counter starts with one and increments whenever an End-of-Tape (paper tape) or a blank card is encountered. When the counter increments, the numbering of source statements starts over at one.

Each error diagnostic printed during pass 2 of the assembly is associated with a different message:

PG *ppp*

ppp is the page number (in the listing) of the previous error diagnostic. PG ØØØ is associated with the first error in the program.

The messages (#*nnn* and PG *ppp*) occur on a separate line, just above each error diagnostic in the listing.

EDITOR ERROR MESSAGES

RTE Editor prints error messages on the operator console in this format:

/EDIT. *error message: illegal edit command*

Then the RTE Editor continues with the Edit File; there is no on-line correction of illegal edit commands.

Error Message	Meaning
MEM OVERFLOW	The Edit File; overflows available memory; RTE Editor prints the command causing the overflow. Edit terminates.

CS ERR	Illegal edit command, which is printed.
PARAM ERR	Edit command "r" or "c" is illegal: non-numeric, = Ø, >72, $r_2 \leq r_1$, $c_2 \leq c_1$, command printed.
SEQ ERR	"r" parameter ≤ a previous "r",or "r" greater than range of Symbolic File; command printed.
/I ERR	No insert source statements after /I; comand printed.
/R ERR	No replacement statements after /R; command printed.
/C OVF	Character overflow in edit statement (i.e., >72 character)
DISK OVF	No disc space for file; edit terminates.
FILE UN	Undefined symbolic File, or LUN (Edit File) = 2. Edit terminates.

RELOCATING LOADER MESSAGES

Messages are printed in this format:

ERROR MESSAGES /LOADR: *message*

LØ1 – checksum error

LØ2 – illegal record

These errors are recoverable. The offending record can be reread by repositioning the tape and typing:

GO,LOADR

For irrecoverable errors, one of the following messages is printed, followed by "LOADR ABORTED" (the loader is terminated):

LØ3 – Memory overflow

LØ4 – Base page linkage area overflow

LØ5 – Symbol table area overflow

LØ6 – Common block error

a. Exceeding allocation in a replacement or addition

b. In a normal background load, first program did not declare largest common block.

LØ7 – Duplicate entry points

LØ8 – No transfer address (main program) in the program unit. Another program may be entered with a GO operator request. (This also occurs when load-and-go is specified, but no program exists in the load-and-go area.)

LØ9 – Record out of sequence

L1Ø – Operator request parameter error. GO requests may be retyped; ON requests may not.

ADDITIONAL MESSAGES

NO BLANK ID SEGMENTS is printed when an available (i.e., blank) ID Segment is not found. The loader calls for program suspension. The operator may then delete a program from the system (OF operator request) or may terminate the loader.

WAITING FOR DISC SPACE is printed when a track allocation cannot be made. The loader repeats the disc request and is suspended until space becomes available. This message is primarily for information, as no action is required.

UNDEFINED EXTS is printed followed by a list of all remaining undefined external symbols after a scan of the library. Additional programs may be loaded by the GO operator request.

LOAD is printed and the loader is suspended whenever an End-of-Tape condition is detected from the input unit.

DUPLICATE PROG NAME – *name* is printed when a program *name* is already defined in the system for a normal load or a program addition. The loader changes the name of the current program by replacing the first two characters with "##"

APPENDIX F
LINE PRINTER FORMATTING

When a user program makes a READ/WRITE EXEC call to an HP line printer, the line printer driver, DVR12, interprets the first character in the line as a carriage control character and prints it as a space. Ⓗ The control characters have the following meanings:

Character	Meaning
blank	Single space (print on every line)
∅	Double space (print on every other line)
1	Eject page
*	Suppress space (overprint current line)
others	Single space.

Each printed line is followed by a single space unless suppressed by the control character asterisk (*). Double spacing requires an additional single space prior to printing the next line. If the last line of a page is printed and the following line contains a "1", then a completely blank page occurs.

When a user program makes an EXEC call for I/O control with the function bits in ICNWD set to 011_8 (see Section III, the optional parameter IPRAM word defines the format action to be performed by the line printer.

Parameter Word (Decimal)	Meaning
<∅	Page eject
∅	Suppress space on the next print operation only
1 to 55	Space 1 to 55 lines ignoring page boundaries
56 to 63	Use carriage control channel equal to the word – 55
64	Set automatic page eject mode
65	Clear automatic page eject mode

CARRIAGE CONTROL CHANNELS

If the parameter word is between 56 to 63, the printer spaces using the carriage control channels, which have the following meanings:

Channel 1	Single space with automatic page eject
Channel 2	Skip to next even line with automatic page eject
Channel 3	Skip to next triple line with automatic page eject
Channel 4	Skip to next 1/2 page boundary
Channel 5	Skip to next 1/4 page boundary
Channel 6	Skip to next 1/6 page boundary
Channel 7	Skip to bottom of the page
Channel 8	Skip to top of next page

AUTOMATIC PAGE EJECT

During non-automatic page eject mode, if the parameter word is equal to 56, then it is interpreted as equal to 1. Automatic page eject mode applies only to single space operations.

Ⓗ *DVR12 checks for certain program names (FTN, ALGOL, ASMB, LOADR, EDIT); for these programs it prints the first character of each line and generates a single space. As a result, the user must not specify automatic buffering on the line printer, since the program names are lost when the device is buffered.*

APPENDIX G
ASCII/OCTAL TABLE

ASCII Character	First Character Octal Equivalent	Second Character Octal Equivalent
A	040400	000101
B	041000	000102
C	041400	000103
D	042000	000104
E	042400	000105
F	043000	000106
G	043400	000107
H	044000	000110
I	044400	000111
J	045000	000112
K	045400	000113
L	046000	000114
M	046400	000115
N	047000	000116
O	047400	000117
P	050000	000120
Q	050400	000121
R	051000	000122
S	051400	000123
T	052000	000124
U	052400	000125
V	053000	000126
W	053400	000127
X	054000	000130
Y	054400	000131
Z	055000	000132
0	030000	000060
1	030400	000061
2	031000	000062
3	031400	000063
4	032000	000064
5	032400	000065
6	033000	000066
7	033400	000067
8	034000	000070
9	034400	000071
space	020000	000040
!	020400	000041
"	021000	000042
#	021400	000043
$	022000	000044
%	022400	000045
&	023000	000046
'	023400	000047
(	024000	000050
)	024400	000051
*	025000	000052
+	025400	000053
,	026000	000054
–	026400	000055

ASCII Character	First Character Octal Equivalent	Second Character Octal Equivalent
.	027000	000056
/	027400	000057
	035000	000072
,	035400	000073
<	036000	000074
=	036400	000075
>	037000	000076
?	037400	000077
@	040000	000100
[	055400	000133
\	056000	000134
]	056400	000135
↑	057000	000136
←	057400	000137
ACK	036000	000174
①	036400	000175
ESC	037000	000176
DEL	037400	000177
NULL	000000	000000
SUM	000400	000001
EOA	001000	000002
EOM	001400	000003
EOT	002000	000004
WRU	002400	000005
RU	003000	000006
BELL	003400	000007
FE_0	004000	000010
HT/SK	004400	000011
LF	005000	000012
V_{TAB}	005400	000013
FF	006000	000014
CR	006400	000015
SO	007000	000016
SI	007400	000017
DC_0	010000	000020
DC_1	010400	000021
DC_2	011000	000022
DC_3	011400	000023
DC_4	012000	000024
ERR	012400	000025
SYNC	013000	000026
LEM	013400	000027
S_0	014000	000030
S_1	014400	000031
S_2	015000	000032
S_3	015400	000033
S_4	016000	000034
S_5	016400	000035
S_6	017000	000036
S_7	017400	000037

APPENDIX H
SUMMARY OF OPERATOR REQUESTS

DN,*n*	Set EQT device *n* down.
EQ,*n*	Print EQT entry *n*.
EQ,*n*,*p*	Delete (*p*=0) or specify (*p*=1) buffering.
GO,*name*[,*p1*,*p2*,...*p5*]	Re-schedule suspended *name*.
IT,*name*,R,MPT [,*h*,*min*[,*s*[,*ms*]]]	Set time values of *name*.
LG,*n*	Declare *n* L&G tracks or release L&G tracks (*n*=0).
LS,*p1*,*p2*	Declare disc number *p1*, track number *p2* as source file.
LU,*n*[,*m*[,*p*]]	Assign EQT *m* subchannel *p* to LU *n*, release *n* (*m*=0), or print *n* (m *absent*).
OF,*name*,*p*	Terminates *name* (*p*=0). Purge name (*p*=8). Abort *name* (*p*>0).
ON, *name*[,NOW] [,*p1*,*p2*,...*p5*]	Schedules *name*. NOW means ignore time values.
PR,*name*,*n*	Set priority of *name*=*n*.
RT,*name*	Release disc tracks assigned to *name*.
SS,*name*	Suspend *name*.
ST,*name*	Print status of *name*.
TI	Print time.
TM, *day*,*h*,*min*,*s*	Specify day of year, and 24 hour time.
TO,*n*[,*m*]	Assign time-out value *m* to EQT *n*, or print value of *n* (*m* absent).
UP,*n*	Set EQT device *n* up.

APPENDIX H
SUMMARY OF OPERATOR REQUESTS

DN,*n*	Set EQT device *n* down.
EQ,*n*	Print EQT entry *n*.
EQ,*n*,*p*	Delete (*p*=0) or specify (*p*=1) buffering.
GO,*name*[,*p1*,*p2*, .*p5*]	Re-schedule suspended *name*.
IT,*name*,R,MPT[,*h*,*min*[,*s*[,*ms*]]]	Set time values of *name*.
LG,*n*	Declare *n* L&G tracks or release L&G tracks (*n*=0).
LS,*p1*,*p2*	Declare disc number *p1*, track number *p2* as source file.
LU,*n* [,*m* [,*p*]]	Assign EQT *m* subchannel *p* to LU *n*, release *n* (*m*= 0), or print *n* (*m* absent).
OF,*name*,*p*	Terminates *name* (*p* =0). Purge name (*p*=8). Abort *name* (*p* >0).
ON,*name*[,NOW] [,*p1*,*p2*,...*p5*]	Schedules *name*. NOW means ignore time values.
PR,*name*,*n*	Set priority of *name*=*n*.
RT,*name*	Release disc tracks assigned to *name*.
SS,*name*	Suspend *name*.
ST,*name*	Print status of *name*.
TI	Print time.
TM,*day*,*h*,*min*,*s*	Specify day of year, and 24 hour time.
TO,*n* [,*m*]	Assign time-out value *m* to EQT *n*, or print value of *n*(*m* absent).
UP,*n*	Set EQT device *n* up.

INDEX

A

	Page
Absolute Start Time	3-14, 3-15
ALGOL Control Statement	4-12
ALGOL Format	3-2
ALOG	4-12
Assembler Language Format	3-1
Automatic Scheduling	1-3, 6-11

B

Background Core Resident	1-2
Background Disc Resident	1-2
Base Page Communication Area	1-4
Base Page Linkages	6-13
Binary Records	B-2
BINRY	3-4, 4-23
Blank ID Segments	6-12
BSS	4-19

C

Central Interrupt Control	1-4
Code Procedures	3-2, 4-25
Common (Privileged)	5-11
Control A	2-1
Control Character Asterisk (*)	F-1
Core Allocations	1-2
COS	4-12, 4-23
Cyclic Check	3-3, B-2

D

Data Statement	4-9
Defective Tracks	6-25
Device Reference Table Word	5-2
Dimension	4-9
Disc Allocation	3-7
Disc Release – Global Tracks	3-9
Disc Release – Program Tracks	3-8
DMA	5-3, 5-4, 5-6
DN	2-1
Dormant	2-5, 3-10, 3-12, 3-15
Double Asterisk (**)	2-3
Drive Fault Lamp	6-25
Drive Ready Lamp	6-25
DR*xx*	3-16
DUMMY	5-4, 5-11

D (Continued)

	Page
DVR Drivers	5-2
DVR05	6-12
DVR12	F-1
DVR77	3-3

E

END Records	B-2
END Statement	3-2, 3-10
EQ, *n*	2-1
EQ, *n*, *p*	2-2
Equipment Table	5-1, A-1
ERRØ	4-10, 4-12
Errors	
DR*xx*	3-16
ERROR Ø5	4-8
E-ØØ18	4-10
E-ØØ19	4-8
ILL INT*xx*	5-2, E-1
IO*xx*	3-16
L1Ø	4-22
MP	3-15
PG*ppp*	4-14
RQ	3-15
SC*xx*	3-15
SCØ3	3-12
EXEC Communication	1-4
Execution Time	2-2, 3-14
EXP	4-12
External Statement	4-10
E-ØØ18	4-10
E-ØØ19	4-8

F

FORTRAN Control Statement	4-8
FORTRAN/FORTRAN IV Format	3-1
FORTRAN Formatter	4-23, 5-3
FRMTR	4-24
F2E.A	4-23
F4D.A	4-23

G

Globally Assigned Tracks	3-7
GO	2-2, 3-10
GO, LOADR (Background)	4-19
GO, LOADR (On-Line)	4-22

H

	Page
Hardware Protected Tracks	6-8
HLT4	6-13
HP 2005	1-1
HP 7900A	1-5, 6-10, 6-11, B-1, B-2

I

ID Segment	1-3, 1-4, 2-4, 4-17, 4-18, 4-20, A-3
ILL INT*xx*	5-2, E-1
Initial Offset Time	3-14
Input/Output Control (RTIOC)	1-4
IO*xx*	3-16
I/O Buffer	5-3
I/O Control	3-5
I/O Status	3-6
I/O Suspension	5-5
IT	2-2, 3-14, 3-15
ITOI	4-12

J

JSB $CIC	5-2

L

L1Ø	4-22
Library Routines	
F2E.A	4-23
F4D.A	4-23
LG	2-3, 3-16
LS	2-3
LU	2-3

M

MP	3-15
MPTFL	5-11
Multiprogramming	1-1

N

NAM Record	6-17, 6-28, A-4
NAM Statement	4-15, 4-17

O

OF	2-4
OF, *name*, 8	2-4, 4-18, 4-20, 6-12
ON	2-4, 3-13
ON, ALGOL	4-11
ON, ASMB	4-13
ON, EDIT	4-3
ON, FTN	4-7

O (Continued)

	Page
ON, FTN4	4-7
ON, LOADR (Background)	4-18
ON, LOADR (Listing)	4-21
ON, LOADR (On-Line)	4-21
ORB Statement	4-15

P

Parity Error	E-2
Pause	4-24
Pause and Stop Statements	4-10
Peripheral Discs	1-5
PG*ppp*	4-14
Power Fail	6-13
PR	2-4
Priority Level	1-4
Privilieged Driver Time-Out	5-5
Privileged Subroutine Structure	4-24
Program Completion	3-9
Program Scheduling	1-3, 3-12
Program Segment Load	3-11
Program Statement	4-9
Program Suspend	2-5, 3-10

R

Real Array	3-4
Real-Time Core Resident	1-2
Real-Time Disc Resident	1-2
Read/Write	3-3
Re-entrant Subroutine Structure	4-23
RMPAR	2-2, 3-10, 3-11, 3-13, 4-22
RQ	3-15
RT	2-5
.RTIO	4-12
.RTOR	4-12
Rubout	2-1

S

SC*xx*	3-15
SCØ3	3-12, 3-15
Segmented Programs	
ALGOL	4-25
Assembler	4-25
FORTRAN	4-25
SIN	4-10, 4-12, 4-23
Software Flag	5-4
Software Protected Tracks	6-8
Spurious Interrupt	5-4
SQRT	4-12
SS	2-5
ST	2-5
Status Field	5-3

S (Continued)

	Page
Subroutines	1-3
ALOG	4-12
BINRY	3-4, 4-23
COS	4-12, 4-23
ERRØ	4-10, 4-12
FORTRAN Formatter	4-23, 5-3
FRMTR	4-24
.ITOI	4-12
RMPAR	2-2, 3-10, 3-11, 3-13, 4-22
.RTIO	4-12
.RTOR	4-12
SIN	4-10, 4-12, 4-23
SQRT	4-12
TAN	4-12
Swapping	1-2, 5-3, 6-10, 6-11
System/Auxiliary Discs	1-5

T

TAN	4-12
TI	2-6, 3-13
Time-Out	5-5, 5-6, 6-12
Time Request	2-6, 3-13

T (Continued)

	Page
Time Shared Basic	B-1
TM	2-6
TO	2-6
Transmission Parity Error	5-5
Type 80 Code	6-11

U

UP	2-8
Utility Subroutine Structure	4-24

W

Waiting and No Waiting	3-12

SPECIAL CHARACTERS

* (Control Character Asterisk)	F-1
##	4-20, E-4
** (Double Asterisk)	2-3
%	4-10
$IRT	5-11
$LIBR	4-15, 4-23
$LIBX	4-15, 4-23
$TB31	6-9, B-2

SALES & SERVICE OFFICES

UNITED STATES

ALABAMA
P.O. Box 4207
2003 Byrd Spring Road S.W.
Huntsville 35802
Tel: (205) 881-4591
TWX: 810-726-2204

ARIZONA
2336 E. Magnolia St.
Phoenix 85034
Tel: (602) 252-5061
TWX: 910-951-1330

5737 East Broadway
Tucson 85716
Tel: (602) 298-2313
TWX: 910-952-1162

CALIFORNIA
1430 East Orangethorpe Ave.
Fullerton 92631
Tel: (714) 870-1000

3939 Lankershim Boulevard
North Hollywood 91604
Tel: (213) 877-1282
TWX: 910-499-2170

1101 Embarcadero Road
Palo Alto 94303
Tel: (415) 327-6500
TWX: 910-373-1280

2220 Watt Ave.
Sacramento 95825
Tel: (916) 482-1463
TWX: 910-367-2092

9606 Aero Drive
San Diego 92123
Tel: (714) 279-3200
TWX: 910-335-2000

COLORADO
7965 East Prentice
Englewood 80110
Tel: (303) 771-3455
TWX: 910-935-0705

CONNECTICUT
508 Tolland Street
East Hartford 06108
Tel: (203) 289-9394
TWX: 710-425-3416

111 East Avenue
Norwalk 06851
Tel: (203) 853-1251
TWX: 710-468-3750

FLORIDA
P.O. Box 24210
2806 W. Oakland Park Blvd.
Ft. Lauderdale 33307
Tel: (305) 731-2020
TWX: 510-955-4099

P.O. Box 20007
Herndon Station 32814
621 Commonwealth Avenue
Orlando
Tel: (305) 841-3970
TWX: 810-850-0113

GEORGIA
P.O. Box 28234
450 Interstate North
Atlanta 30328
Tel: (404) 436-6181
TWX: 810-766-4890

ILLINOIS
5500 Howard Street
Skokie 60076
Tel: (312) 677-0400
TWX: 910-223-3613

INDIANA
3839 Meadows Drive
Indianapolis 46205
Tel: (317) 546-4891
TWX: 810-341-3263

LOUISIANA
P.O. Box 856
1942 Williams Boulevard
Kenner 70062
Tel: (504) 721-6201
TWX: 810-955-5524

MARYLAND
6707 Whitestone Road
Baltimore 21207
Tel: (301) 944-5400
TWX: 710-862-9157

P.O. Box 1648
2 Choke Cherry Road
Rockville 20850
Tel: (301) 948-6370
TWX: 710-828-9684

MASSACHUSETTS
32 Hartwell Ave.
Lexington 02173
Tel: (617) 861-8960
TWX: 710-326-6904

MICHIGAN
21840 West Nine Mile Road
Southfield 48075
Tel: (313) 353-9100
TWX: 810-224-4882

MINNESOTA
2459 University Avenue
St. Paul 55114
Tel: (612) 645-9461
TWX: 910-563-3734

MISSOURI
11131 Colorado Ave.
Kansas City 64137
Tel: (816) 763-8000
TWX: 910-771-2087

2812 South Brentwood Blvd.
St. Louis 63144
Tel: (314) 962-5000
TWX: 910-760-1670

NEW JERSEY
W. 120 Century Road
Paramus 07652
Tel: (201) 265-5000
TWX: 710-990-4951

1060 N. Kings Highway
Cherry Hill 08034
Tel: (609) 667-4000
TWX: 710-892-4945

NEW MEXICO
P.O. Box 8366
Station C
6501 Lomas Boulevard N.E.
Albuquerque 87108
Tel: (505) 265-3713
TWX: 910-989-1665

156 Wyatt Drive
Las Cruces 88001
Tel: (505) 526-2485
TWX: 910-983-0550

NEW YORK
1702 Central Avenue
Albany 12205
Tel: (518) 869-8462
TWX: 710-441-8270

1219 Campville Road
Endicott 13760
Tel: (607) 754-0050
TWX: 510-252-0890

82 Washington Street
Poughkeepsie 12601
Tel: (914) 454-7330
TWX: 510-248-0012

39 Saginaw Drive
Rochester 14623
Tel: (716) 473-9500
TWX: 510-253-5981

5858 East Molloy Road
Syracuse 13211
Tel: (315) 454-2486
TWX: 710-541-0482

1 Crossways Park West
Woodbury 11797
Tel: (516) 921-0300
TWX: 510-223-0811

NORTH CAROLINA
P.O. Box 5188
1923 North Main Street
High Point 27262
Tel: (919) 885-8101
TWX: 510-926-1516

OHIO
25575 Center Ridge Road
Cleveland 44145
Tel: (216) 835-0300
TWX: 810-427-9129

3460 South Dixie Drive
Dayton 45439
Tel: (513) 298-0351
TWX: 810-459-1925

1120 Morse Road
Columbus 43229
Tel: (614) 846-1300

OKLAHOMA
2919 United Founders Boulevard
Oklahoma City 73112
Tel: (405) 848-2801
TWX: 910-830-6862

OREGON
Westhills Mall, Suite 158
4475 S.W. Scholls Ferry Road
Portland 97225
Tel: (503) 292-9171
TWX: 910-464-6103

PENNSYLVANIA
2500 Moss Side Boulevard
Monroeville 15146
Tel: (412) 271-0724
TWX: 710-797-3650

1021 8th Avenue
King of Prussia Industrial Park
King of Prussia 19406
Tel: (215) 265-7000
TWX: 510-660-2670

RHODE ISLAND
873 Waterman Ave.
East Providence 02914
Tel: (401) 434-5535
TWX: 710-381-7573

TEXAS
P.O. Box 1270
201 E. Arapaho Rd.
Richardson 75080
Tel: (214) 231-6101
TWX: 910-867-4723

P.O. Box 22813
6300 Westpark Drive
Suite 100
Houston 77027
Tel: (713) 781-6000
TWX: 910-881-2645

231 Billy Mitchell Road
San Antonio 78226
Tel: (512) 434-4171
TWX: 910-871-1170

UTAH
2890 South Main Street
Salt Lake City 84115
Tel: (801) 487-0715
TWX: 910-925-5681

VERMONT
P.O. Box 2287
Kennedy Drive
South Burlington 05401
Tel: (802) 658-4455
TWX: 510-299-0025

VIRGINIA
P.O. Box 6514
2111 Spencer Road
Richmond 23230
Tel: (703) 285-3431
TWX: 710-956-0157

WASHINGTON
433-108th N.E.
Bellevue 98004
Tel: (206) 454-3971
TWX: 910-443-2303

***WEST VIRGINIA**
Charleston
Tel: (304) 768-1232

FOR U.S. AREAS NOT LISTED:
Contact the regional office nearest you: Atlanta, Georgia . . . North Hollywood, California . . . Paramus, New Jersey . . . Skokie, Illinois. Their complete addresses are listed above.

*Service Only

CANADA

ALBERTA
Hewlett-Packard (Canada) Ltd.
11745 Jasper Ave.
Edmonton
Tel: (403) 482-5561
TWX: 610-831-2431

BRITISH COLUMBIA
Hewlett-Packard (Canada) Ltd.
4519 Canada Way
North Burnaby 2
Tel: (604) 433-8213
TWX: 610-922-5059

MANITOBA
Hewlett-Packard (Canada) Ltd.
511 Bradford Ct.
Winnipeg
Tel: (204) 786-7581
TWX: 610-671-3531

NOVA SCOTIA
Hewlett-Packard (Canada) Ltd.
2745 Dutch Village Rd.
Suite 206
Halifax
Tel: (902) 455-0511
TWX: 610-271-4482

ONTARIO
Hewlett-Packard (Canada) Ltd.
880 Lady Ellen Place
Ottawa 3
Tel: (613) 722-4223
TWX: 610-562-1952

Hewlett-Packard (Canada) Ltd.
50 Galaxy Blvd.
Rexdale
Tel: (416) 677-9611
TWX: 610-492-4246

QUEBEC
Hewlett-Packard (Canada) Ltd.
275 Hymus Boulevard
Pointe Claire
Tel: (514) 697-4232
TWX: 610-422-3022
Telex: 01-20607

FOR CANADIAN AREAS NOT LISTED:
Contact Hewlett-Packard (Canada) Ltd. in Pointe Claire, at the complete address listed above.

CENTRAL AND SOUTH AMERICA

ARGENTINA
Hewlett-Packard Argentina S.A.C.e.I
Lavalle 1171 - 3°
Buenos Aires
Tel: 35-0436, 35-0627, 35-0431
Telex: 012-1009
Cable: HEWPACKARG

BRAZIL
Hewlett-Packard Do Brasil I.e.C Ltda.
Rua Frei Caneca 1119
Sao Paulo - 3, SP
Tel: 288-7111, 287-5858
Cable: HEWPACK Sao Paulo

Hewlett-Packard Do Brasil
Praca Dom Feliciano 78
Salas 806/808
Porto Alegre
Rio Grande do Sul (RS)-Brasil
Tel: 25-8470
Cable: HEWPACK Porto Alegre

Hewlett-Packard Do Brasil I.e.C. Ltda.
Rua da Matriz 29
Botafogo ZC-02
Rio de Janeiro, GB
Tel: 246-4417
Cable: HEWPACK Rio de Janeiro

Hewlett-Packard Do Brasil Indústria e Comercia Ltda.
Praca Dom Feliciano 78
Salas 806-8
Porto Alegre RGS

CHILE
Héctor Calcagni y Cia, Ltda.
Bustos, 1932-3er Piso
Casilla 13942
Santiago
Tel: 423 96
Cable: CALCAGNI Santiago

COLOMBIA
Instrumentacion
Henrik A. Langebaek & Kier Ltda.
Carrera 7 No. 48-59
Apartado Aereo 6287
Bogota, 1 D.E.
Tel: 45-78-06, 45-55-46
Cable: AARIS Bogota
Telex: 44400 INSTCO

COSTA RICA
Lic. Alfredo Gallegos Gurdián
Apartado 10159
San José
Tel: 21-86-13
Cable: GALGUR San José

ECUADOR
Laboratorios de Radio-Ingenieria
Calle Guayaquil 1246
Post Office Box 3199
Quito
Tel: 212-496; 219-185
Cable: HORVATH Quito

EL SALVADOR
Electrónica
Apartado Postal 1589
Blvd. Venezuela 1231
San Salvador
Tel: 217527; 214895
Cable: ELECTRONICA
San Salvador

MEXICO
Hewlett-Packard Mexicana, S.A. de C.V.
622 Adolfo Prieto
Col. del Valle
Mexico 12, D.F.
Tel: 543-4232; 523-1874
Telex: 0017-74507

NICARAGUA
Roberto Terán G.
Apartado Postal 689
Edificio Terán
Managua
Tel: 3451, 3452
Cable: ROTERAN Managua

PANAMA
Electrónico Balboa, S.A.
P.O. Box 4929
Ave. Manuel Espinosa No. 13-50
Bldg. Alina
Panama City
Tel: 230833
Telex: 3481003, Curundu, Canal Zone
Cable: ELECTRON Panama City

PERU
Compañia Electro Medica S.A.
Ave. Enrique Canaual 312
San Isidro
Casilla 1030
Lima
Tel: 22-3900
Cable: ELMED Lima

PUERTO RICO
San Juan Electronics, Inc.
P.O. Box 5167
Ponce de Leon 154
Pda. 3-PTA de Tierra
San Juan 00906
Tel: (809) 725-3342, 722-3342
Cable: SATRONICS San Juan
Telex: SATRON 3450 332

SURINAME
Surtel-Radio Holland N.V.
P.O. Box 155
Paramaribo
Tel: 72118
Cable: Treurniet Paramaribo

URUGUAY
Pablo Ferrando S.A.
Comercial e Industrial
Avenida Italia 2877
Casilla de Correo 370
Montevideo
Tel: 40-3102
Cable: RADIUM Montevideo

VENEZUELA
Hewlett-Packard De Venezuela C.A.
Apartado 50933
Caracas
Tel: 71.88.05, 71.88.69, 71.99.30
Cable: HEWPACK Caracas
Telex: 39521146

FOR AREAS NOT LISTED,

CONTACT:
Hewlett-Packard
INTERCONTINENTAL
3200 Hillview Ave.
Palo Alto, California 94304
Tel: (415) 493-1501
TWX: 910-373-1267
Cable: HEWPACK Palo Alto
Telex: 034-8461

EUROPE

AUSTRIA
Unilabor GmbH
Wissenschaftliche Instrumente
Rummelhardtgasse 6
P.O. Box 33
A-1095 **Vienna**
Tel: (222) 42 61 81, 43 13 94
Cable: LABORINSTRUMENT Vienna
Telex: 75 762

BELGIUM
Hewlett-Packard S.A. Benelux
348 Boulevard du Souverain
B-1160 **Brussels**
Tel: (02) 722240
Cable: PALOBEN Brussels
Telex 23 494

DENMARK
Hewlett-Packard A/S
Datavej 38
DK-3460 **Birkerod**
Tel: (01) 81 66 40
Cable: HEWPACK AS
Telex 66 40

Hewlett-Packard A/S
Torvet 9
DK-8600 **Silkeborg**
Tel: (06)-82-71-66

FINLAND
Hewlett-Packard Oy
Bulevardi 26
P.O. Box 12185
Helsinki 12
Tel: 13-730
Cable: HEWPACKOY-Helsinki
Telex: 12-1563

FRANCE
Hewlett-Packard France
Quartier de Courtaboeuf
Boite Postale No. 6
F-91 **Orsay**
Tel: 1-920 88 01
Cable: HEWPACK Orsay
Telex: 60048

Hewlett-Packard France
4 Quai des Etroits
F-69 **Lyon** 5ème
Tel: 78-42 63 45
Cable: HEWPACK Lyon
Telex: 31617

Hewlett-Packard France
29 rue de la Gara
F-31 **Blagnac**
Tel: (61) 85 82 29
Telex: 51957

GERMANY
Hewlett-Packard Vertriebs-GmbH
Berliner Strasse 117
Postfach 560/40
D6 Nieder-Eschbach/Ffm 56
Tel: (0611) 50 10 64
Cable: HEWPACKSA Frankfurt
Telex: 41 32 49 FRA

Hewlett-Packard Vertriebs-GmbH
Wilmersdorfer Strasse 113/114
D-1000 **Berlin W.** 12
Tel: (0311) 3137046
Telex: 18 34 05

Hewlett-Packard Vertriebs-GmbH
Herrenbergerstrasse 110
D7030 **Böblingen**, Württemberg
Tel: (07031) 66 72 86
Cable: **HEPAG Böblingen**
Telex: 72 65 739

Hewlett-Packard Vertriebs-GmbH
Vogelsanger Weg 38
D4 **Düsseldorf**
Tel: (0211 63 80 31/35
Telex: 85/86 533

Hewlett-Packard Vertriebs-GmbH
Wendenstr. 23
D2 **Hamburg** 1
Tel: (0411) 24 05 51/52
Cable: HEWPACKSA Hamburg
Telex: 21 53 32

Hewlett-Packard Vertriebs-GmbH
Reginfriedstrasse 13
D8 **München** 9
Tel: (0811) 69 59 71/75
Cable: HEWPACKSA München
Telex: 52 49 85

GREECE
Kostas Karayannis
18, Ermou Street
Athens 126
Tel: 230301,3,5
Cable: RAKAR Athens
Telex: 21 59 62 RKAR GR

IRELAND
Hewlett-Packard Ltd.
224 Bath Road
Slough, SLI 4 DS, Bucks
Tel: Slough 753-33341
Cable: HEWPIE Slough
Telex: 84413

ITALY
Hewlett-Packard Italiana S.p.A.
Via Amerigo Vespucci 2
1-20124 **Milan**
Tel: (2) 6251 (10 lines)
Cable: HEWPACKIT Milan
Telex: 32046

Hewlett-Packard Italiana S.p.A.
Palazzo Italia
Piazza Marconi 25
1-00144 **Rome** - Eur
Tel: 6-591 2544
Cable: HEWPACKIT Rome
Telex: 61514

NETHERLANDS
Hewlett-Packard Benelux, N.V.
Weerdestein 117
P.O. Box 7825
Amsterdam, Z 11
Tel: 020-42 77 77
Cable: PALOBEN Amsterdam
Telex: 13 216

NORWAY
Hewlett-Packard Norge A/S
Box 149
Nesveien 13
N-1344 **Hasium**
Tel: (02)-53 83 60
Cable: HEWPACK Oslo
Telex: 16621

PORTUGAL
Telectra
Empresa Tecnica de Equipamentos
Electricos, S.a.r.l.
Rua Rodrigo da Fonseca 103
P.O. Box 2531
Lisbon 1
Tel: 68 60 72
Cable: TELECTRA Lisbon
Telex: 1598

SPAIN
Ataio Ingenieros SA
Enrique Larreta 12
Madrid, 16
Tel: 215 35 43
Cable: TELEATAIO Madrid
Telex: 27249E

Ataio Ingenieros SA
Ganduxer 76
Barcelona 6
Tel: 211-44-66
Cable: TELEATAIO Barcelona

SWEDEN
Hewlett-Packard Sverige AB
Enighetsvägen 1-3
Fack
S-161 20 **Bromma** 20
Tel: (08) 98 12 50
Cable: MEASUREMENTS Stockholm
Telex: 10721

Hewlett-Packard Sverige AB
Hagakersgatan 9C
S-431 41 **Mölndal**
Tel: 031 - 27 68 00
Telex: 21 312 hpmlndl

SWITZERLAND
Hewlett Packard Schweiz AG
Zurcherstrasse 20
CH-8952 **Schlieren** Zurich
Tel: (051) 98 18 21/24
Cable: HPAG CH
Telex: 53933

Hewlett Packard Schweiz A.G.
Rue du Bois-du-Lan 7
1217 Meyrin 2 **Geneva**
Tel: (022) 41 54 00
Cable: HEWPACKSA Geneva
Telex: 27333 HPSA CH

TURKEY
Telekom Engineering Bureau
P.O. Box 376
Karakoy
Istanbul
Tel: 49 40 40
Cable: TELEMATION Istanbul

UNITED KINGDOM
Hewlett-Packard Ltd.
224 Bath Road
Slough, SLI 4 DS, Bucks
Tel: Slough (0753) 33341
Cable: HEWPIE Slough
Telex: 84413

Hewlett-Packard Ltd.
The Graftons
Stamford New Road
Altrincham, Cheshire
Tel: 061 928-8626
Telex: 668068

YUGOSLAVIA
Belram S.A.
83 avenue des Mimosas
Brussels 1150, Belgium
Tel: 34 33 32, 34 26 19
Cable: BELRAMEL Brussels
Telex: 21790

SOCIALIST COUNTRIES PLEASE CONTACT:
Hewlett-Packard Ges.m.b.H
Innstrasse 23/2
Postfach
A1204 **Vienna,** Austria
Tel: (222) 3366 06/09
Cable: HEWPACK Vienna
Telex: 75923

ALL OTHER EUROPEAN COUNTRIES CONTACT:
Hewlett-Packard S.A.
Rue du Bois-du-Lan 7
1217 Meyrin 2 **Geneva**
Switzerland
Tel: (022) 41 54 00
Cable: HEWPACKSA Geneva
Telex: 2.24.86

AFRICA, ASIA, AUSTRALIA

ANGOLA
Telectra Empresa Técnia de Equipamentos Eléctricos *SAR*
Rua de Barbosa Rodrigues 42-1°
Box 6487
Luanda
Cable: TELECTRA Luanda

AUSTRALIA
Hewlett-Packard Australia Pty. Ltd.
22-26 Weir Street
Glen Iris, 3146
Victoria
Tel: 20.1371 (6 lines)
Cable: HEWPARD Melbourne
Telex: 31024

Hewlett-Packard Australia Pty. Ltd.
61 Alexander Street
Crows Nest 2065
New South Wales
Tel: 43.7866
Cable. HEWPARD Sydney
Telex: 21561

Hewlett-Packard Australia Pty. Ltd.
97 Churchill Road
Prospect 5082
South Australia
Tel: 65.2366
Cable: HEWPARD Adelaide

Hewlett Packard Australia Pty. Ltd.
2nd Floor, Suite 13
Casablanca Buildings
196 Adelaide Terrace
Perth, W.A. 6000
Tel: 21-3330
Cable: HEWPARD Perth

Hewlett-Packard Australia Pty. Ltd.
10 Woolley Street
P.O. Box 191
Dickson A.C.T. 2602
Tel: 49-8194
Cable. HEWPARD Canberra ACT

Hewlett-Packard Australia Pty. Ltd.
75 Simpsons Road
Bardon
Queensland, 4068
Tel: 36-5411

CEYLON
United Electricals Ltd.
P.O. Box 681
Yahala Building
Staples Street
Colombo 2
Tel: 5496
Cable: HOTPOINT Colombo

CYPRUS
Kypronics
19 Gregorios & Xenopoulos Road
P.O. Box 1152
Nicosia
Tel: 6282-75628
Cable: HE-I-NAMI

ETHIOPIA
African Salespower & Agency Private Ltd., Co.
P. O. Box 718
58/59 Cunningham St.
Addis Ababa
Tel: 12285
Cable: ASACO Addisababa

HONG KONG
Schmidt & Co. (Hong Kong) Ltd.
P.O. Box 297
1511, Prince's Building 15th Floor
10, Chater Road
Hong Kong
Tel: 240168, 232735
Cable: SCHMIDTCO Hong Kong

INDIA
Blue Star Ltd.
Kasturi Buildings
Jamshedji Tata Rd.
Bombay 20BR, India
Tel: 29 50 21
Telex: 2156
Cable: BLUEFROST

Blue Star Ltd.
Band Box House
Prabhadevi
Bombay 25DD, India
Tel: 45 73 01
Telex: 2156
Cable: BLUESTAR

Blue Star Ltd.
14/40 Civil Lines
Kanpur, India
Tel: 6 88 82
Cable: BLUESTAR

Blue Star, Ltd.
7 Hare Street
P.O. Box 506
Calcutta 1, India
Tel: 23-0131
Telex: 655
Cable: BLUESTAR

Blue Star Ltd.
Blue Star House,
34 Ring Road
Lajpat Nagar
New Delhi 24, India
Tel: 62 32 76
Telex: 463
Cable, BLUESTAR

Blue Star Ltd.
17-C Ulsoor Road
Bangalore-8

Blue Star, Ltd.
96 Park Lane
Secunderabad 3, India
Tel: 7 63 91
Cable: BLUEFROST

Blue Star, Ltd.
23/24 Second Line Beach
Madras 1, India
Tel: 2 39 55
Telex: 379
Cable: BLUESTAR

Blue Star, Ltd.
1B Kaiser Bungalow
Dindli Road
Jamshedpur, India
Tel: 38 04
Cable: BLUESTAR

INDONESIA
Bah Bolon Trading Coy. N.V.
Djalah Merdeka 29
Bandung
Tel: 4915; 51560
Cable: ILMU
Telex: 08-809

IRAN
Telecom, Ltd.
P. O. Box 1812
240 Kh. Saba Shomali
Teheran
Tel: 43850, 48111
Cable: BASCOM Teheran

ISRAEL
Electronics & Engineering Div. of Motorola Israel Ltd.
17 Aminadav Street
Tel-Aviv
Tel: 36941 (3 lines)
Cable: BASTEL Tel-Aviv
Telex: Bastel Tv 033-569

JAPAN
Yokogawa-Hewlett-Packard Ltd.
Ohashi Building
1-59-1 Yoyogi
Shibuya-ku, **Tokyo**
Tel: 03-370-2281/7
Telex: 232-2024YHP
Cable: YHPMARKET TOK 23-724

Yokogawa-Hewlett-Packard Ltd.
Nisei Ibaragi Bldg.
2-2-8 Kasuga
Ibaragi-Shi
Osaka
Tel: (0726) 23-1641
Telex: 385-5332 YHPOSAKA

Yokogawa-Hewlett-Packard Ltd.
Ito Building
No. 59, Kotori-cho
Nakamura-ku, Nagoya City
Tel: (052) 551-0215

Yokogawa-Hewlett-Packard Ltd.
Nitto Bldg.
2300 Shinohara-cho,
Kohoku-ku
Yokohama 222
Tel: (405) 432-1504/5

KENYA
Kenya Kinetics
P.O. Box 18311
Nairobi, Kenya
Tel: 57726
Cable: PROTON

KOREA
American Trading Co.,
Korea, Ltd.
Seoul P.O. Box 1103
7th & 8th floors, DaeKyung Bldg.
107 Sejong Ro
Chongro-Ku, Seoul
Tel: 75-5841 (4 lines)
Cable: AMTRACO Seoul

LEBANON
Constantin E. Macridis
Clemenceau Street
P.O. Box 7213
Beirut
Tel: 220846
Cable: ELECTRONUCLEAR Beirut

MALAYSIA
MECOMB Malaysia Ltd.
2 Lorong 13/6A
Section 13
Petaling Jaya, **Selangor**
Cable: MECOMB Kuala Lumpur

MOZAMBIQUE
A. N. Goncalves, LDA.
4.1 Apt. 14 Av. D. Luis
P.O. Box 107
Lourenco Marques
Cable: NEGON

NEW ZEALAND
Hewlett-Packard (N.Z.) Ltd.
94-96 Dixson St.
P.O. Box 9443
Wellington, N.Z.
Tel: 56-559
Cable: HEWPACK Wellington

Hewlett Packard (N.Z.) Ltd.
Box 51092
Pukuranga
Tel: 573-733
Cable: HEWPACK, Auckland

PAKISTAN (EAST)
Mushko & Company, Ltd.
1, Jinnah Avenue
Dacca 2
Tel: 280058
Cable: NEWDEAL Dacca

PAKISTAN (WEST)
Mushko & Company, Ltd.
Oosman Chambers
Abdullah Haroon Road
Karachi 3
Tel: 511027, 512927
Cable: COOPERATOR Karachi

PHILIPPINES
Electromex Inc.
Makati Commercial Center
2129 Pasong Tamo
Makati, Rizal D 708
P.O. Box 1028
Manila
Tel: 89-85-01; 88-91-71
Cable: ELEMEX Manila

SINGAPORE
Mechanical and Combustion Engineering Company Ltd.
9, Jalan Kilang
Red Hill Industrial Estate
Singapore, 3
Tel: 642361-3; 632611
Cable: MECOMB Singapore

Hewlett-Packard Far East Area Office
P.O. Box 87
Alexandra Post Office
Singapore 3
Tel: 633022
Cable: HEWPACK SINGAPORE

SOUTH AFRICA
Hewlett Packard South Africa (Pty.), Ltd.
P.O. Box 31716
Braamfontein Transvaal
Milnerton
30 De Beer Street
Johannesburg
Tel: 725-2080, 725-2030
Telex: 0226 JH
Cable: HEWPACK Johannesburg

Hewlett Packard South Africa (Pty.), Ltd.
Breecastle House
Bree Street
Cape Town
Tel: 3-6019, 3-6545
Cable: HEWPACK Cape Town
Telex: 5-0006

Hewlett Packard South Africa (Pty.), Ltd.
30B Glenwood Centre
Corner Hunt & Moore Roads
P.O. Box 99
Overport, Natal
Tel: 347536

TAIWAN
Hewlett Packard Taiwan
39 Chung Shiao West Road
Sec. 1
Overseas Insurance
Corp. Bldg. 7th Floor
Taipei
Tel: 579-605, 579-610, 579-613
Telex: TP824 HEWPACK
Cable: HEWPACK Taipei

THAILAND
The International Engineering Co., Ltd.
P. O. Box 39
614 Sukhumvit Road
Bangkok
Tel: 910722 (7 lines)
Cable: GYSOM
TLX INTENCO BK-226 Bangkok

UGANDA
Uganda Tele-Electric Co., Ltd.
P.O. Box 4449
Kampala
Tel: 57279
Cable: COMCO Kampala

VIETNAM
Peninsular Trading Inc.
P.O. Box H-3
216 Hien-Vuong
Saigon
Tel: 20805, 93398
Cable: PENTRA, SAIGON 242

ZAMBIA
R. J. Tilbury (Zambia) Ltd.
P.O. Box 2792
Lusaka
Zambia, Central Africa
Tel: 73793
Cable: ARJAYTEE, Lusaka

MEDITERRANEAN AND MIDDLE EAST COUNTRIES NOT SHOWN PLEASE CONTACT:
Hewlett-Packard Correspondence Office
Piazza Marconi 25
I-00144 Rome-Eur, Italy
Tel: (6) 59 40 29
Cable: HEWPACKIT Rome
Telex: 61514

OTHER AREAS NOT LISTED, CONTACT:
Hewlett-Packard INTERCONTINENTAL
3200 Hillview Ave.
Palo Alto, California 94304
Tel: (415) 326-7000
(Feb. 71 493-1501)
TWX: 910-373-1267
Cable: HEWPACK Palo Alto
Telex: 034-8461

MANUAL PART NO.
02005-90001

PRINTED IN U.S.A.