HEWLETT-PACKARD
SYSTEMS PROGRAMMING COURSE

STUDENTS MANUAL

LEVEL 1

REAL-TIME MULTIPROGRAMMING SYSTEM

(HP STOCK NO. 5951-2134)

-NOTICE-

The information contained in this manual is for training purposes only. Consult the Hewlett-Packard documentation supplied with the system for current information concerning the specific computer operating system furnished.

AUTOMATIC MEASUREMENT DIVISION 395 Page Mill Road, Palo Alto, California 94306 Telephone (415) 326-1755

Printed in U.S.A. Jan 72

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

LECTURE – Day One

LAB – Day One

LECTURE – Day Two

LAB – Day Two

LECTURE – Day Three

LAB – Day Three

LECTURE – Day Four

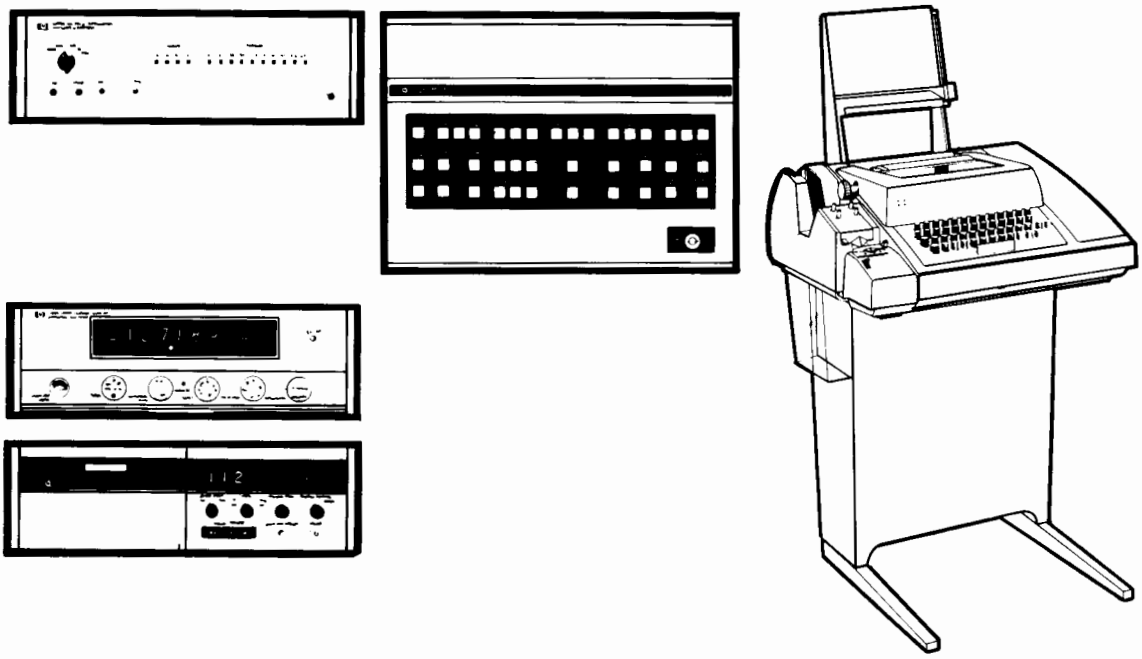
LAB – Day Four

LECTURE – Day Five

LAB – Day Five

INTRODUCTION 1

TBG
+



FUNDAMENTAL COMPUTER INSTRUMENT SYSTEM (A)

FUNDAMENTAL COMPUTER INSTRUMENT SYSTEM (B)

AVAILABLE SOFTWARE

I/O CONTROL
RELOCATING LOADER
DEBUG ROUTINE
PREPARE CONTROL SYSTEM
SYMBOLIC EDITOR
RELOCATABLE LIBRARY
FORTRAN IV LIBRARY
BASIC
PREPARE BASIC SYSTEM
FORTRAN II
ASSEMBLER
ALGOL
CROSS-REFERENCE GENERATOR
+
I/O DRIVERS
+
DIAGNOSTICS

FEATURES

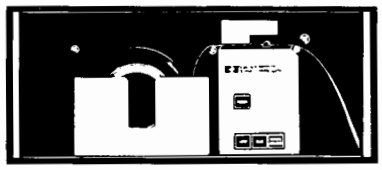
"BCS" OR "BASIC" OPERATING SYSTEMS
DATA ACQUISITION, REDUCTION AND CONTROL

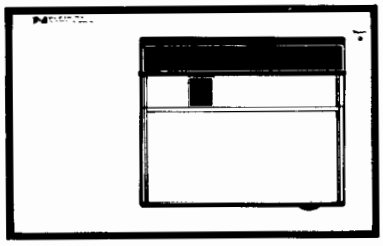
ADVANTAGES

COMBINATIONS OF ASSEMBLY, FORTRAN & ALGOL PROGRAMMING
"BASIC" INSTRUMENT DRIVERS AVAILABLE FROM DATA CENTERS

BENEFITS

DACE OPERATING SYSTEM
REAL-TIME, ONLINE, REAL DATA ANALYSIS AND RESULTS
EASE OF EXPANDING SYSTEM WITH USER CAPABILITY





AVAILABLE SOFTWARE

I/O DRIVERS
+
DIAGNOSTICS

HIGH-SPEED PAPER TAPE (A)

FEATURES

READ RATE ≈ 500 CHARACTERS/SEC.
CORE LOAD TIME ≈ 65 SEC.
PUNCH RATE ≈ 75 CHARACTERS/SEC.
CORE DUMP TIME ≈ 7.25 MIN.
RACK HEIGHT ≈ 21 INCHES.

ADVANTAGES

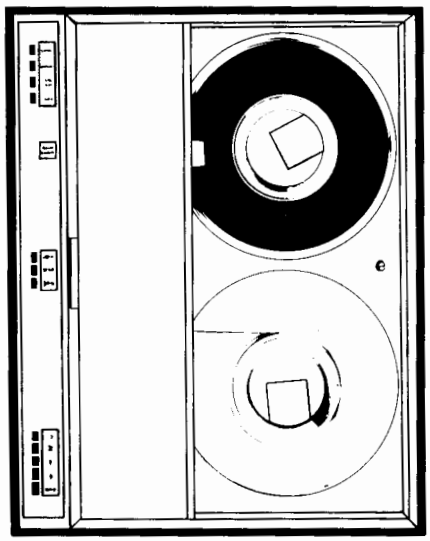
INCREASES CORE LOAD RATE BY 50
LOWEST RELATIVE COST
ACCESS TIME TO CORE LOADS IS A FUNCTION OF OPERATOR'S SEARCH AND LOAD CAPABILITY

BENEFITS

A NECESSARY STEPPING STONE TO MORE SOPHISITCATED OPERATING SYSTEMS

HIGH-SPEED PAPER TAPE (B)

DMA
+



AVAILABLE SOFTWARE

BOOTSTRAP
.IPL.
UTILITY
PREPARE TAPE SYSTEM
+
I/O DRIVERS
+
DIAGNOSTICS

MAGNETIC TAPE (A)

MAGNETIC TAPE (B)

FEATURES

READ/WRITE RATE ≈ 18,000 (16-BIT) WORDS/SEC.
CORE LOAD/DUMP TIME ≈ 1 SEC.
RACK HEIGHT ≈ 24 INCHES.

ADVANTAGES

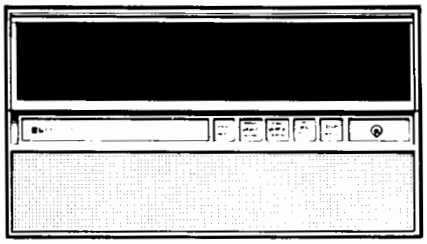
MAGNETIC TAPE OPERATING SYSTEM
- W/BATCH OPERATION
- W/CHAINING

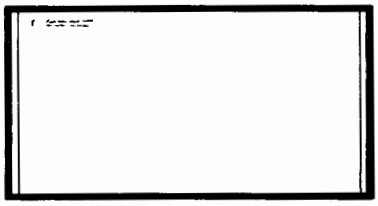
INFINITE OFFLINE STORAGE
ACCESS TIME TO CORE LOADS IS A FUNCTION OF POSITION OF CORE LOAD ON TAPE

BENEFITS

MINIMUM SOPHISTICATED OPERATING SYSTEM (MTOS)
HIGH INFORMATION STORAGE EFFICIENCY
A USEFUL TIMESAVING BACKUP TO MORE SOPHISTICATED OPERATING SYSTEMS

EAU
+
DMA
+





+
MP

AVAILABLE SOFTWARE

ALGOL COMPILER
DOS-M SYSTEM GENERATOR
DOS-M CORE-RESIDENT SYSTEM EXEC MODULES
JOB PROCESSOR
BOOT STRAP LOADER
RELOCATING LOADER
ASSEMBLER
FORTRAN II COMPILER
RTE/DOS RELOCATABLE LIBRARY (EAU AND NON-EAU)
RTE/DOS FORTRAN IV LIBRARY
RTE/DOS FORTRAN FORMATTER
RTE/DOS FORTRAN IV COMPILER
RTE/DOS FORTRAN IV (10K COMPILER AREA)
DOS CROSS REFERENCE ROUTINE
DOS-M EXTENDED FILE MGR.
+
RTE MH SYSTEM GENERATOR
RTE EXECUTIVE
RTE SCHEDULER
RTE RTIOC
+
I/O DRIVERS
+
DIAGNOSTICS

MOVING HEAD DISC (A)

FEATURES

READ/WRITE RATE ≈ 156K WORDS/SEC.
CORE DUMP/LOAD TIME ≈ 103.8 MSEC.
RACK HEIGHT 17½ INCHES

ADVANTAGES

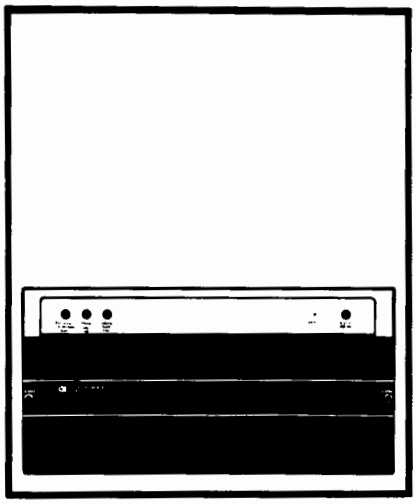
INFINITE OFFLINE STORAGE
(1.2 M 16-BIT WORDS 1 PLATTER)

BENEFITS

DOS-M OPERATING SYSTEM
RTE-M OPERATING SYSTEM

MOVING HEAD DISC (B)

AVAILABLE SOFTWARE



DOS SYSTEM GENERATOR
DOS CORE-RESIDENT EXEC MODULES
JOB PROC/FILE MGR.
ASSEMBLER
FORTRAN
RELOCATING LOADER
SYSTEM DUMP
PREPARE TAPE SYSTEM
RTE/DOS ALGOL MAIN CONTROL
RTE/DOS RELOCATABLE LIBRARY (NON-EAU OR EAU)
RTE/DOS FORTRAN IV LIBRARY
RTE/DOS FORTRAN FORMATTER
RTE/DOS FORTRAN IV COMPILER
RTE/DOS FORTRAN IV (10K COMPILER AREA)
DOS CROSS REFERENCE ROUTINE
RTE FH GENERATOR
RTE EXEC
RTE SCHED
RTE RTIOC
+
I/O DRIVERS
+
DIAGNOSTICS

+
DMA
+
MP
+
EAU

FIXED HEAD DISC (A)

FIXED HEAD DISC (B)

FEATURES

READ/WRITE RATE 118K WORDS/SEC.
CORE LOAD DUMP TIME 82.5 MSEC.
RACK HEIGHT 28 TO 49 INCHES

ADVANTAGES

GREATER RELIABILITY THAN MOVING HEAD DISC
FASTER ACCESS TIME
BETTER ENVIRONMENTAL TOLERANCE

BENEFITS

DOS OPERATING SYSTEM
RTE OPERATING SYSTEM

REAL TIME MULTIPROGRAMMING SYSTEM OBJECTIVES

REQUIREMENTS

1 – THAT PROGRAMS BE SCHEDULABLE
   - A) BY OPERATOR REQUEST
   - B) ON THE BASIS OF REAL TIME OF DAY
   - C) BY OTHER PROGRAMS
   - D) BY EXTERNAL EVENT
   - E) IN ORDER OF USER ASSIGNED PRIORITY

2 – THAT THE USER BE PERMITTED TO COMMUNICATE THRU THE EXEC
   - A) TO HIS PROGRAMS
   - B) TO I/O DEVICES
   - C) TO THE SCHEDULER

3 – THAT I/O OPERATIONS
   - A) DO NOT DESTROY THE SYSTEM
   - B) BE ALLOCATED DMA CHANNELS ACCORDING TO SOME PRIORITY
   - C) CAN BE BUFFERED
   - D) FUNCTION ACCORDING TO CALLING PROGRAMS PRIORITY

4 – THAT PROGRAMS MAY BE
   - A) PERMANENTLY ADDED TO THE SYSTEM ON LINE
   - B) REPLACED ON LINE WITHOUT SYSTEM RE-GENERATION
   - C) EDITED AND COMPILED ON LINE WITHOUT THE USE OF PAPER TAPE

FOREGROUND CORE RESIDENT **CODE TYPE 1**

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 MAY BE LOGICALLY SUB-ORDINATE TO A SET OF DISC RESIDENT CONTROL PROGRAMS. RECOMMENDED PRIORITY, 1 - 49.

FOREGROUND DISC RESIDENT **CODE TYPE 2**

THIS TYPE RESIDES IN ABSOLUTE FORMAT ON THE DISC AND MUST BE TRANSFERRED INTO A RESERVED PART OF CORE BEFORE EXECUTING. THE LARGEST USER PROGRAM OF THIS TYPE LOADED AT GENERATION TIME DETERMINES THE MINIMUM SIZE OF THIS CORE AREA. THUS, DISC RESIDENT PROGRAMS PROVIDE SLOWER RESPONSE TO REAL TIME EVENTS THAN CORE RESIDENT SINCE ALL FOREGROUND 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 IF IN I/O SUSPENSION [EXCEPT IF I/O BUFFER IS IN COMMON]) IF A HIGHER PRIORITY PROGRAM NEEDS THE AREA. RECOMMENDED PRIORITY, 50 - 74.

USER AREA-FOREGROUND

BACKGROUND CORE RESIDENT **CODE TYPE 4**

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. RECOMMENDED PRIORITY, 1 - 49.

BACKGROUND DISC RESIDENT **CODE TYPE 3,5**

UNLIKE THE REAL-TIME DISC RESIDENT TYPE, BACKGROUND DISC RESIDENT PROGRAMS ARE NEVER SWAPPED, BUT MAY BE SEGMENTED. THEY OCCUPY CORE UNTIL COMPLETED OR ABORTED. BACKGROUND DISC RESIDENT SOFTWARE USUALLY INCLUDES THE FOLLOWING:

- ASSEMBLER (4K)
- FORTRAN II (4K)
- FORTRAN IV (4K OR 10K)
- ALGOL (6.5K)
- EDITOR (3K)
- RELOCATING LOADER (3K)

RECOMMENDED PRIORITY, 75 - 99.

USER AREA-BACKROUND

REAL TIME LIBRARY

LIBRARY ROUTINES ARE OF THREE TYPES

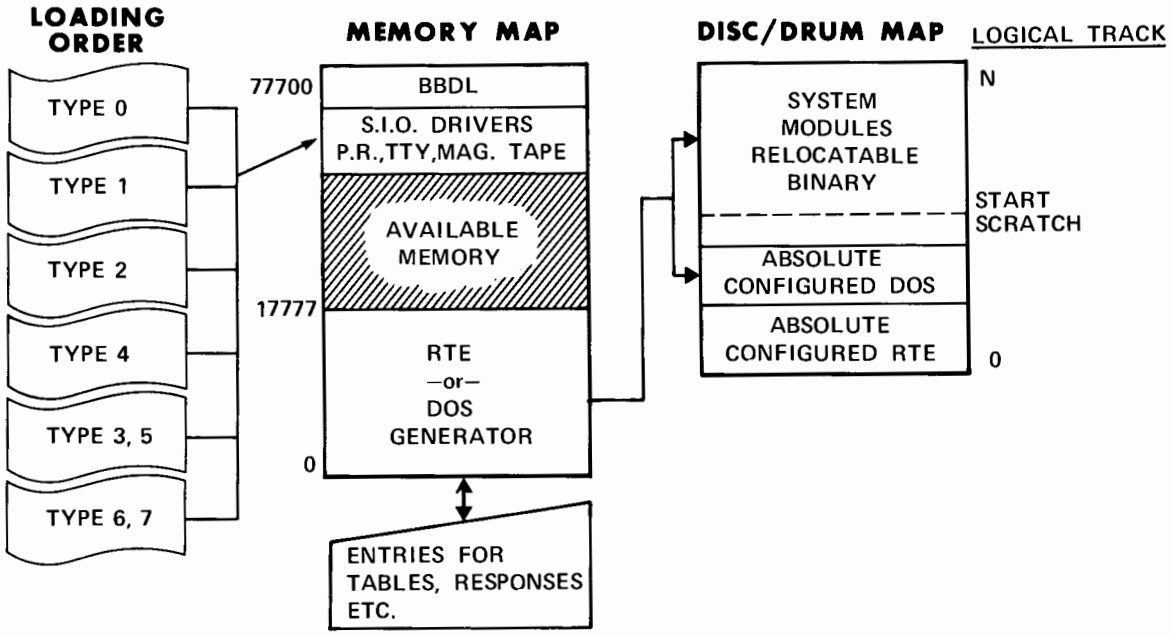
TYPE 6	RE-ENTRANT (INTERRUPT ENABLED)	ROUTINES WHOSE PROCESSING MAY BE INTERRUPTED WHEN A HIGHER PRIORITY PROGRAM IS CALLED BEFORE COMPLETION, AND THEREFORE STORES ITS VARIABLES AND RETURN POINTS IN BUFFER MEMORY WHEN ENTERED. EXEC CALLS ARE ILLEGAL.
TYPE 6	PRIVILEGED (INTERRUPT DISABLED)	ROUTINES WITH SHORT EXECUTION TIME ($<$ 1 ms). THE ROUTINE IS ALLOWED TO COMPLETE ITS CURRENT OPERATION BEFORE BEING ENTERED BY A DIFFERENT PROGRAM CALL. EXEC CALLS ARE ILLEGAL.
TYPE 7	UTILITY	ROUTINES THAT ARE NEITHER RE-ENTRANT NOR PRIVILEGED AND THEREFORE MUST BE ATTACHED TO EVERY PROGRAM BY WHICH THEY ARE CALLED.

AT SYSTEM GENERATION TIME:

* ALL TYPE 6 LIBRARY ROUTINES REQUIRED BY CORE RESIDENT PROGRAMS ARE STORED IN THE CORE AREA TERMED "RESIDENT LIBRARY". ALL TYPE 7 LIBRARY ROUTINES ARE STORED IN THE CORE MEMORY AREA TERMED "CORE RESIDENT" IMMEDIATELY FOLLOWING THE CALLING PROGRAM.

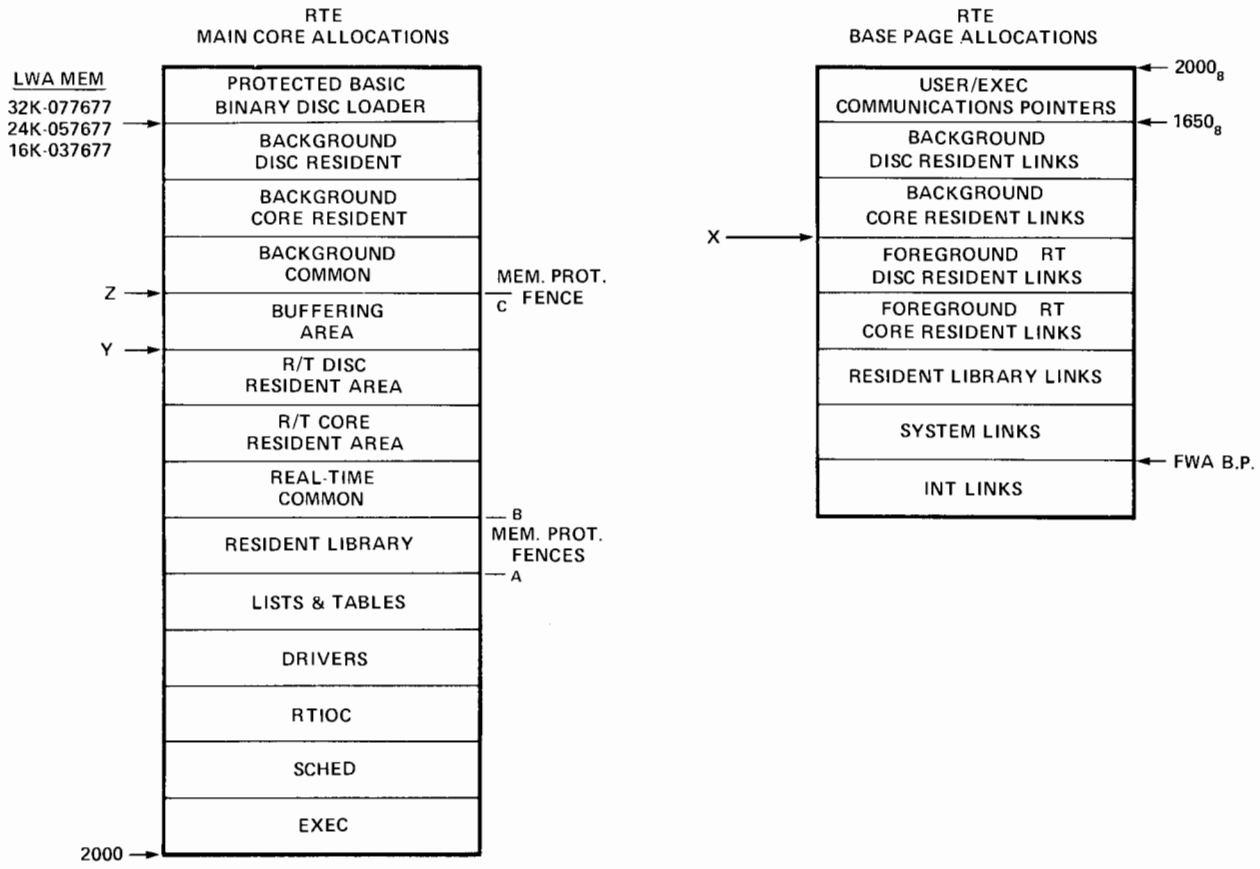
* ALL REMAINING LIBRARY ROUTINES ARE STORED ON THE DISC IN RELOCATABLE FORMAT FOR USE BY THE RELOCATING LOADER.

SYSTEM GENERATION (TAPE)

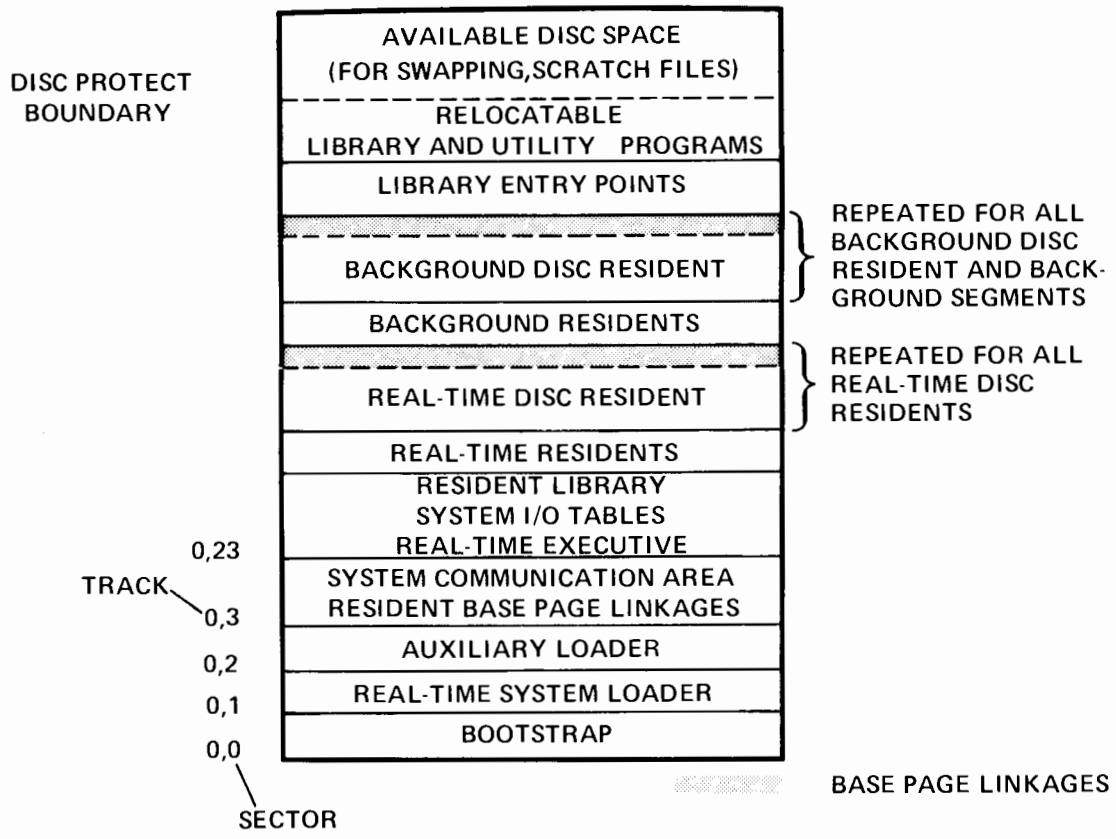


- THE SYSTEM GENERATOR IS LOADED INTO MEMORY USING THE BBDL.
- INITIALIZATION PHASE – ESTABLISHES DISC SIZE, TYPE, SYSTEM HARDWARE INFO.
- PROGRAM INPUT PHASE – SYSTEM AND USER PROGRAMS ARE COPIED ON THE DISC/DRUM.
- PARAMETER INPUT PHASE – PROGRAM PRIORITIES AND TYPE CODES MAY BE CHANGED.
- DISC LOADING PHASE – ALL TABLES ARE CONSTRUCTED AND THE ABSOLUTE SYSTEM IS CREATED ON THE SYSTEM DISC/DRUM.

CONFIGURED SYSTEM MEMORY MAP



CONFIGURED SYSTEM DISC MAP

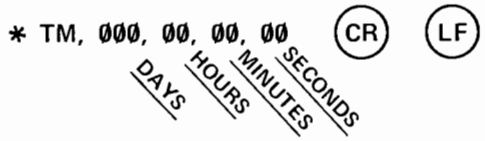


OPERATOR REQUESTS

THE OPERATOR GAINS THE ATTENTION OF THE EXECUTIVE BY DEPRESSING ANY KEY ON THE SYSTEM TELEPRINTER KEYBOARD. THE COMPUTER RESPONDS BY TYPING A SINGLE ASTERISK (*). AT THAT POINT THE OPERATOR MAY REQUEST ANY ONE OF 17 OPERATIONS. ALL OPERATOR REQUESTS HAVE A TWO CHARACTER FORMAT AND UP TO SEVEN ADDITIONAL PARAMETER FIELDS, SEPARATED BY COMMAS.

FOR EXAMPLE

OPERATOR REQUEST TO RESET THE R-T CLOCK



IF A MISTAKE IS MADE ON INPUT, A "RUBOUT" WILL DELETE THE LINE. TO DELETE A SINGLE INCORRECT CHARACTER, DEPRESS THE "CONTROL" KEY AND THE "A" KEY SIMULTANEOUSLY.

ST

PURPOSE

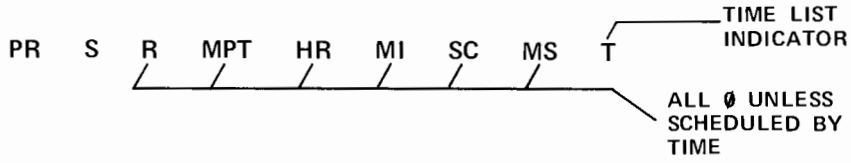
TO REQUEST THAT THE STATUS (PRIORITY, CURRENT LIST, TIME VALUES) OF A PROGRAM BE PRINTED ON THE CONSOLE.

FORMAT

ST, name

WHERE name IS THE NAME OF THE PROGRAM WHOSE STATUS IS TO BE PRINTED.

THE STATUS IS PRINTED ON ONE LINE IN A FIXED FORMAT:



WHERE PR IS THE PRIORITY, A VALUE FROM 1 TO 99_{10},

S IS THE CURRENT LIST IN WHICH THE PROGRAM IS LOCATED:

- Ø – DORMANT
- 1 – SCHEDULED
- 2 – I/O SUSPEND
- 3 – NOT USED
- 4 – UNAVAILABLE MEMORY SUSPEND
- 5 – DISC ALLOCATION SUSPEND
- 6 – OPERATOR SUSPEND

ON

PURPOSE

TO SCHEDULE A PROGRAM FOR EXECUTION. UP TO FIVE PARAMETERS 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.

P1 THROUGH P5 IS A LIST OF PARAMETERS PASSED TO name WHEN IT IS SCHEDULED (MUST BE POSITIVE INTEGERS $\leq$ 32767).

NOTE:

PARAMETERS ARE PICKED UP BY A

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

SINCE THE ADDRESS OF THE PARAMETERS IS IN THE 'B' REGISTER WHEN THE PROGRAM NAME IS SCHEDULED.

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 p IS AN INTEGER, AND
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 OWN/GLOBAL 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. IF I/O SUSPENDED, OF, name, 8 MUST BE ENTERED AGAIN TO PERMANENTLY REMOVE name FORM THE CORE 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 LOADED ON-LINE) ARE RELEASED. THE BLANK ID SEGMENT IS THEN AVAILABLE FOR LOADING ANOTHER PROGRAM WITH LOADER.

SS

PURPOSE

TO SUSPEND A PROGRAM FROM EXECUTION.

FORMAT

SS, name

WHERE name IS THE NAME OF THE PROGRAM TO BE SUSPENDED.

THIS REQUEST PLACES THE PROGRAM IN THE OPERATOR SUSPEND STATUS IF THE PROGRAM IS EXECUTING, SCHEDULED, OR ALREADY IN OPERATOR SUSPENSION, IF THE PROGRAM IS DORMANT THE REQUEST IS ILLEGAL. IF THE PROGRAM IS I/O, MEMORY, OR DISC SUSPENDED, THE EXEC WILL SUSPEND THE PROGRAM AGAIN AT THE CONCLUSION OF THE CURRENT SUSPEND PERIOD.

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

IF THE PROGRAM HAS NOT BEEN SUSPENDED PREVIOUSLY BY THE OPERATOR OR HAS NOT SUSPENDED ITSELF, THE REQUEST IS ILLEGAL.
THE ADDRESS OF THE PARAMETERS P1 THRU P5 IS PLACED IN THE 'B' REGISTER PRIOR TO EXECUTION OF PROGRAM name

IT

PURPOSE

TO SET TIME VALUES FOR A PROGRAM, SO THAT THE PROGRAM EXECUTES AUTOMATICALLY AT SELECTED TIMES WHEN TURNED ON.

FORMAT

IT, name, R, MPT [, HR, MI [,SC [,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 Ø TO 999 WHICH IS USED WITH R TO GIVE THE ACTUAL TIME INTERVAL FOR SCHEDULING,

HR – HOURS
MI – MINUTES
SC – SECONDS
MS – TENS OF MS.

} SETS AN INITIAL START TIME.

NOTE: WHEN THE SYSTEM IS RELOADED FROM THE DISC, "IT" VALUES WILL BE OVERLAYED WITH THE IT VALUE ESTABLISHED AT GENERATION TIME.

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.

THE PRIORITY OF name RESETS (TO THAT SET BY RTGEN OR LOADR) WHENEVER THE RTE SYSTEM RESTARTS FROM DISC. ONE IS THE HIGHEST PRIORITY, AND 99 IS THE LOWEST.

ONLY dormant PROGRAMS CAN HAVE THEIR PRIORITY CHANGED.

EQ

PURPOSE

TO PRINT THE DESCRIPTION AND STATUS OF AN I/O DEVICE AS RECORDED IN EQT ENTRY, AND, OPTIONALLY, TO CHANGE THE AUTOMATIC 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
IF P = Ø THEN DELETE BUFFERING
OR
IF P = 1 THEN SPECIFY BUFFERING
OR
IF P IS ABSENT

THE INFORMATION IS PRINTED AS:

SC DVRnn D B Un LS

WHERE SC IS THE SELECT CODE (I/O CHANNEL)
DVRnn IS THE DRIVER ROUTINE,
D IS D IF DMA REQUIRED, Ø IF NOT,
B IS B IF AUTOMATIC OUTPUT BUFFERING USED, Ø IF NOT,
Un IS THE UNIT NUMBER FOR SUBCHANNEL ADDRESSING

LS IS THE LOGICAL STATUS:

Ø – AVAILABLE
1 – UNAVAILABLE (DOWN)
2 – UNAVAILABLE (BUSY)
3 – WAITING FOR DMA ASSIGNMENT

WHEN THE SYSTEM IS RESTARTED FROM THE DISC, BUFFERING DESIGNATIONS MADE BY EQ, n, p ARE RESET TO THE VALUES ORIGINALLY MADE BY RTGEN.

THE STANDARD LINE PRINTER (CONTROLLED BY I/O DRIVER DVR12) MAY NOT BE BUFFERED WHEN USED BY THE FTN II, FTN IV, ALGOL COMPILERS, ASMB, EDIT, OR LOADER.

LU

PURPOSE

TO PRINT OR CHANGE A LOGICAL UNIT NUMBER ASSIGNMENT.

FORMAT

LU, n [, x [, s]]

WHERE n IS A LOGICAL UNIT NUMBER FROM 1 TO 63_{10},
x IS AN EQT ENTRY NUMBER TO ASSIGN TO n.
(IF x = Ø, THE CURRENT ASSIGNMENT OF LOGICAL UNIT n IS RELEASED; IF x IS ABSENT, THE ASSIGNMENT OF LOGICAL UNIT n IS PRINTED.)
s IS A SUBCHANNEL NUMBER FROM 0 to 7_{10}

LOGICAL UNIT ASSIGNMENTS ARE PRINTED ON ONE LINE:

LU #n = #x, #s

WHERE n IS THE LOGICAL UNIT NUMBER, x IS THE EQT ENTRY NUMBER ASSIGNED TO n AND s IS THE SUBCHANNEL NUMBER IF OTHER THAN ZERO.

WHEN AN IRRECOVERABLE PROBLEM OCCURS ON AN I/O DEVICE, THE OPERATOR CAN BYPASS THE DOWNED DEVICE 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.

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 EQUIPMENT TABLE (EQT) ENTRY NUMBER OF THE I/O DEVICE TO BE SET DOWN.

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.

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.

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.

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.

IF m IS ABSENT THE TIME-OUT VALUE OF EQT n IS PRINTED.

THE INFORMATION IS PRINTED AS

TO # 03 = 100

WHICH MEANS EQT ENTRY NUMBER 3 HAS A TIME-OUT VALUE OF ONE SECOND.

NOTE: I/O DEVICES USING DVR ØØ AND DVR Ø5 MAY NOT BE REASSIGNED A TIME OUT LESS THAN 500 (5 SEC).

LS

PURPOSE

TO DESIGNATE THE DISC LOGICAL UNIT NUMBER AND STARTING TRACK NUMBER OF AN EXISTING SOURCE FILE, SUBSEQUENT TO OPERATING ON THAT SOURCE FILE WITH EDIT, FTN, 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 = Ø ELIMINATES THE CURRENT SOURCE FILE DESIGNATION.

P2 IS THE STARTING TRACK NUMBER OF THE SOURCE FILE (IN DECIMAL).

LS REPLACES ANY PREVIOUS FILE DECLARATIONS WITH THE CURRENT FILE. ONLY ONE FILE MAY BE DECLARED AT A TIME.

—

LG

PURPOSE

TO ALLOCATE OR RELEASE A GROUP OF CONTIGUOUS DISC TRACKS ON A SINGLE LU FOR LOAD-AND-GO OPERATIONS.

FORMAT

LG, n

WHERE n IS:

Ø (ZERO) – RELEASE THE ALLOCATED LOAD-AND-GO AREA.

n (> Ø) – ALLOCATE n CONTIGUOUS TRACKS FOR A LOAD-AND-GO AREA; SET FLAGS FOR LOAD-AND-GO OPERATION

LG MUST ALLOCATE ENOUGH TRACKS FOR STORING BINARY OBJECT CODE BEFORE A LOAD-AND-GO COMPILATION OR ASSEMBLY. IF NOT, THE COMPILER OR ASSEMBLER ABORTS AND A DIAGNOSTIC APPEARS:

IOØ6 – LOAD-AND-GO AREA NOT DEFINED
IOØ9 – OVERFLOW OF LOAD-AND-GO AREA

RT

PURPOSE

TO RELEASE ALL OWN DISC TRACKS ASSIGNED TO A PROGRAM

FORMAT

RT, name

WHERE

name IS THE NAME OF THE PROGRAM THAT IS TO HAVE ITS TRACKS RELEASED.

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 PROGRAMS IN DISC TRACK ALLOCATION SUSPENSION ARE RESCHEDULED.

TI

PURPOSE

TO PRINT THE CURRENT TIME OF DAY AND DAY OF THE YEAR, AS RECORDED IN THE REAL-TIME CLOCK.

FORMAT

```
TI
```

THE COMPUTER PRINTS OUT THE DAY AND TIME:

```
DAY   HR   MI   SC
```

WHERE DAY IS THE THREE-DIGIT DAY OF THE YEAR, AND HR, MI, SC IS THE TIME ON A 24-HOUR CLOCK.

OPERATOR REQUEST ERRORS

MESSAGE	MEANING
OP CODE ERROR	ILLEGAL OPERATOR REQUEST WORD.
NO SUCH PROG	THE name GIVEN IS NOT A MAIN PROGRAM IN THE SYSTEM.
ILLEGAL STATUS	A PROGRAM IS NOT IN THE APPROPRIATE STATE.
INPUT ERROR	A PARAMETER IS ILLEGAL.

ASSEMBLER, COMPILER OPERATIONS

INITIATION

* ON, { ASMB / FTN / FTN4 / ALGOL }, P1, P2, P3, P4, P5

WHERE P1 = INPUT UNIT (5 IF NOT SPECIFIED)
P2 = LIST UNIT (6 IF NOT SPECIFIED)
P3 = PUNCH UNIT (4 IF NOT SPECIFIED)
P4 = NUMBER OF LINES PER PAGE (56 IF NOT SPECIFIED)
P5 = LOAD-AND-GO OPTION, SPECIFIED BY TYPING 99.
IF PRESENT, THIS IS THE TERMINATING PARAMETER.

FOR EXAMPLE

* ON, { ASMB / FTN / FTN4 / ALGOL } < IS EQUIVALENT TO, * ON, { ASMB / FTN / FTN4 / ALGOL }, 5, 6, 4, 56, >

COMPLETION

$END, { ASMB / FTN / FTN4 / ALGOL }

REAL TIME ASSEMBLER

[redacted]

THE HP REAL TIME ASSEMBLER OPERATES AS A SEGMENTED BACKGROUND DISC RESIDENT PROGRAM AND REQUIRES AT LEAST 4K OF MEMORY.

ADDITIONAL ASSEMBLER FEATURES

1. "LOAD AND GO" CAPABILITY
2. MEMORY REFERENCE INSTRUCTIONS PERMIT NUMERIC OPERANDS IN THE RANGE ($0-1777_8$)

CHANGES

1. THE "ORB" (ORIGIN BASE PAGE) STATEMENT HAS BEEN DELETED.
2. THE "NAM" STATEMENT ALLOWS FOR ADDITIONAL PARAMETERS NEEDED TO SPECIFY PROGRAM TYPE, PRIORITY, AND EXECUTION TIME.
3. THE SOURCE FILE CAN BE ON THE SYSTEM DISC.
4. NON-EAU OBJECT CODE CAN BE SPECIFIED (X OPTION)
5. ABSOLUTE ASSEMBLIES ARE PERMITTED, BUT EXECUTION MUST BE OFF-LINE OR ON ANOTHER COMPUTER.

[redacted]

THE NAM STATEMENT

GENERAL FORM:

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

WHERE P1 SPECIFIES PROGRAM TYPE

- 0 = SYSTEM
- 1 = RT CORE RESIDENT
- 2 = RT DISC RESIDENT
- 3 = BG DISC RESIDENT
- 4 = BG CORE RESIDENT
- 5 = BG SEGMENT
- 6 = LIBRARY
- 7 = UTILITY

P2 SPECIFIES PRIORITY (Ø – 99)

P3 THROUGH P8 SPECIFIES THE EXECUTION TIME

- P3 = RESOLUTION CODE (Ø – 4)
- P4 = EXECUTION MULTIPLE (Ø – 999)
- P5 = HOURS (Ø – 23)
- P6 = MINUTES (Ø – 59)
- P7 = SECONDS (Ø – 59)
- P8 = 1Ø's OF MILLISECONDS (Ø – 99)

NOTE:

IF NO PARAMETERS ARE SPECIFIED, PRIORITY (P2) IS SET TO 99 (LOWEST) AND THE TYPE (P1) IS SET TO 3, (BACKGROUND DISC). ALL OTHER PARAMETERS ARE SET TO ZERO.

REAL TIME FORTRAN II

HP R-T FORTRAN II OPERATES AS A SEGMENTED BACKGROUND PROGRAM AND REQUIRES AT LEAST 4K OF MEMORY, PLUS 4 * (NUMBER OF VARIABLES, CONSTANTS, LABELS, FUNCTIONS) IN THE LARGEST PROGRAM TO BE COMPILED.

ADDITIONAL FEATURES OF R-T FORTRAN

1. THE DATA STATEMENT
2. THE EXTERNAL STATEMENT
3. A BINARY DISC PROGRAM AND A SIMPLE "ERROR PRINT" ROUTINE HAS BEEN ADDED TO THE FORTRAN LIBRARY.
4. "LOAD AND GO" CAPABILITY.

CHANGES

1. CONTROL STATEMENT OPTION "T" ALLOWS PRINTING OF THE SYMBOL TABLE, AND A PARAMETER (N) ALLOWS THE USER TO SUPPLY HIS OWN ERROR PRINT ROUTINE.
2. THE PAUSE AND STOP STATEMENTS WERE MODIFIED TO PREVENT THE HALT HP NORMALLY ASSOCIATED WITH THESE FUNCTIONS.
3. THE PROGRAM STATEMENT NOW ALLOWS FOR AN ADDITIONAL PARAMETER STRING TO SPECIFY PROGRAM TYPE, PRIORITY, AND EXECUTION TIME.

FORTRAN II CONTROL STATEMENTS

FTN, A, B, L, n

WHERE: A = ASSEMBLY LISTING (INCLUDING SYMBOL TABLE)

B = BINARY OBJECT TAPE

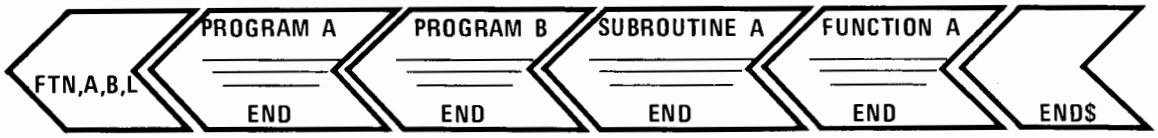
L = SOURCE LISTING

n = 1 - 9 TO SPECIFY AN ERROR ROUTINE OF THE FORM ERRn

NOTE: IF n IS NOT SPECIFIED, A LIBRARY ROUTINE CALLED ERRØ WILL BE USED IN CASE OF ERRORS DURING EXECUTION OF: ALOG, SIN, COS, SQRT, .RTOR., RTOI, EXP, .ITOI, TAN.

FTN, T, B, L

WHERE: T = LIST SYMBOL TABLE –



NOTE: A MAXIMUM OF FIVE PROGRAMS AND OR SUBPROGRAMS MAY BE GROUPED UNDER ONE CONTROL STATEMENT.

THE PROGRAM STATEMENT

GENERAL FORM

PROGRAM *name* (P1, P2, P3, P4, P5, P6, P7, P8)

WHERE: P1 SPECIFIES PROGRAM TYPE

- 0 = SYSTEM
- 1 = RT CORE RESIDENT
- 2 = RT DISC RESIDENT
- 3 = BG DISC RESIDENT
- 4 = BG CORE RESIDENT
- 5 = BG SEGMENT
- 6 = LIBRARY
- 7 = UTILITY

P2 SPECIFIES PRIORITY (Ø – 99)

P3 THROUGH P8 SPECIFIES THE EXECUTION TIME

- P3 = RESOLUTION CODE (Ø – 4)
- P4 = EXECUTION MULTIPLE (Ø – 999)
- P5 = HOURS (Ø – 23)
- P6 = MINUTES (Ø – 59)
- P7 = SECONDS (Ø – 59(
- P8 = 10's OF MILLISECONDS (Ø – 99)

NOTE:

IF NO PARAMETERS ARE SPECIFIED, PRIORITY (P2) IS SET TO 99 (LOWEST) AND THE TYPE (P1) IS SET TO 3, (BACKGROUND DISC). ALL OTHER PARAMETERS ARE SET TO ZERO.

ERRn EXAMPLE

A USER MAY SUPPLY A SUBROUTINE TO PROVIDE SPECIAL HANDLING IN THE EVENT OF ERRORS DURING LIBRARY PROGRAM EXECUTION. THE LINKAGE TO THE ERROR PROCESSOR IS SET UP WITH THE FORTRAN CONTROL STATEMENT.

FOR EXAMPLE

```
FTN, B, L, 9

      PROGRAM TNT

      A = -1.

      X = ALOG(A)

      DLD   A
      JSB   ALOG
      JSB   ERR9
      -NORMAL RETURN
```

SINCE THE ARGUMENT IS NEGATIVE THE ALOG SUBROUTINE REJECTS THE CALL BY PLACING THE ERROR CODE "02 UN" IN REGISTERS A & B AND RETURNING TO P + 1 OF THE CALLING PROGRAM.

```
ASMB, R, B, L, T
      NAM ERR9
      ENT ERR9

ERR9  NOP           -ENTRY-
      STA   BUF     SAVE ERROR
      STB   BUF+1   CODES.
```

AT THIS POINT THE USER MAY INSERT LOGIC OF HIS OWN CHOOSING. THE PROGRAM COULD BE SUSPENDED TO ALLOW OPERATOR INTERVENTION OR DATA CHECKING.

```
      JMP   ERR9,I
```

THE DATA STATEMENT

THE DATA STATEMENT IS USED TO DEFINE INITIAL VALUES OF VARIABLES AND ARRAY ELEMENTS, AND IS ALLOWED TO APPEAR ONLY IMMEDIATELY BEFORE THE FIRST EXECUTABLE STATEMENT.

FOR EXAMPLE

```
1    5    10   15   20   25   30   35   40
     PROGRAM USE(3,99)
100  FORMAT(3I6/F10.3/I6)
     DIMENSION J(3),K(2)
     COMMON I(2)
     EQUIVALENCE (I(1),REG,IAREG)
X                (I(2),IBREG)
     DATA X/2.5/,Y /4.0/
     DATA J(1),J(2),J(3)/7,32767,-13/
     DATA K(1),K(2)/2*1/
     Z=X+Y
     I=K(1)+K(2)
     WRITE(6,100)J,Z,I
     END
```

WOULD PRODUCE THE FOLLOWING OUTPUT.

```
^^^^^7^ 32767^^^-13
6.500
2
```

NOTE: A DATA STATEMENT MAY NOT HAVE THE FOLLOWING FORM,

DATA B/10*0.0/ A DIAGNOSTIC E-0004 WILL RESULT

THE EXTERNAL STATEMENT

THE EXTERNAL STATEMENT ALLOWS THE DEFINITION OF EXTERNAL PROCEDURE NAMES IN FORTRAN PROGRAMS. THIS ALLOWS USER OR SYSTEM FUNCTIONS OR SUBROUTINE NAMES TO BE USED AS PARAMETERS. ONLY FIVE EXTERNALS PER PROGRAM ALLOWED.

FOR EXAMPLE:

```
      FUNCTION TRIG(FUN,X)
      TRIG=FUN(X)
      END
      PROGRAM TEST
      EXTERNAL SIN,COS
      WRITE(6,101)
      DO 10 I=1,90
      A=FLOAT(I)*.01745
      X=TRIG(SIN,A)
      Y=TRIG(COS,A)
   10 WRITE(6,102) I,X,Y
  101 FORMAT(1H1,5X"ANGLE"5X"SIN"7X"COS"//)
  102 FORMAT(3X,I3,6X,F9.6,1X,F9.6)
      PAUSE
      END
```

PRODUCES THIS OUTPUT

ANGLE	SIN	COS
1	.017449	.999848
2	.034893	.999391
3	.052326	.998630
4	.069743	.997565
5	.087139	.996196
⋮	⋮	⋮

REAL TIME HP ALGOL

* HP R-T ALGOL OPERATES AS A TWO PART SEGMENTED BACKGROUND PROGRAM AND REQUIRES AT LEAST 6.5K, PLUS 4 * (NUMBER OF VARIABLES + CONSTANTS, LABELS, INTERNAL PROCEDURES, CODE PROCEDURES) IN THE LARGEST PROGRAM TO BE COMPILED

ADDITIONAL FEATURES OF HP R-T ALGOL

* A COMPLETE DESCRIPTION OF THE HP ALGOL LANGUAGE, INCLUDING ERROR MESSAGES, MAY BE FOUND IN THE HP ALGOL PROGRAMMERS REFERENCE MANUAL.

DIFFERENCES IN OPERATION

* THE PAUSE AND STOP STATEMENTS WERE MODIFIED TO PREVENT THE HALT HP NORMALLY ASSOCIATED WITH THESE FUNCTIONS.
* THE HP ALGOL CONTROL STATEMENT NOW ALLOWS FOR AN ADDITIONAL PARAMETER STRING TO SPECIFY PROCEDURE TYPE, PRIORITY, AND EXECUTION TIME.
* IN THE HP ALGOL CONTROL STATEMENT. THE 'S'(:= SWITCH REGISTER CONTROL) IS NOW IGNORED

HP REAL TIME 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, IN QUOTES, WITHOUT BLANKS
p1 = n: A DIGIT FROM 1 THROUGH 9 SPECIFYING THE ERROR-ROUTINE NAME. A LIBRARY ROUTINE, ERRn, 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:

IF (P_n IS ABSENT) THEN P_n : = Ø;

IF (P_3 IS ABSENT) THEN P_3 : = 99;

IF (S_4 IS ABSENT) AND (P_2 IS ABSENT) THEN P2 : = 3;

IF (S_4 IS PRESENT) AND (P_2 IS ABSENT) THEN P2 : = 7;

REAL TIME EDITOR

THE HP REAL TIME EDITOR OPERATES AS A BACKGROUND DISC RESIDENT PROGRAM.

ADDITIONAL EDITOR FEATURES

1 – THE EDITOR HAS THE CAPABILITY TO CREATE SOURCE FILES ON THE DISC. THE SOURCE FILE CAN BE EDITED OR MAY BE USED AS THE INPUT FILE FOR THE ASSEMBLER OR FORTRAN COMPILER.

CHANGES

1 – THE EDIT STATEMENT (/L) (LIST) HAS BEEN ELIMINATED. THE LIST REQUEST IS NOW A PARAMETER IN THE "ON, EDIT" STATEMENT.

2 – A NEW STATEMENT (/A) (ABORT) WILL ABORT THE EDIT OPERATION IMMEDIATELY. THIS IS USEFUL ONLY WHEN THE KEYBOARD IS THE SELECTED EDIT FILE INPUT UNIT.

3 – MINOR CHANGES WERE MADE IN CERTAIN ERROR MESSAGE FORMATS.

EDITOR OPERATIONS

EDITOR INITIATION

* ON, EDIT, P1, P2, P3, P4

WHERE P1 = INPUT UNIT FOR THE EDIT FILE (5 IF NOT SPECIFIED)
P2 = INPUT UNIT FOR THE SYMBOLIC FILE (5 IF NOT SPECIFIED)
P3 = OUTPUT UNIT FOR THE UPDATED (OR CURRENT) FILE (4 IF NOT SPECIFIED)
P4 = Ø, A "NORMAL" EDIT OPERATION (Ø IF NOT SPECIFIED)
1, LIST OPERATION ONLY
2, CREATE A DISC SOURCE FILE ONLY

EDITOR CONTINUATION

* GO, EDIT, P1

WHERE P1 = Ø, READ NEXT TAPE.
≠ Ø, TERMINATE EDIT PROCESS.

REAL TIME LOADER

THE REAL TIME LOADER CONVERTS RELOCATABLE BINARY OBJECT PROGRAMS INTO ABSOLUTE BINARY AND WRITES THE ABSOLUTE CODE ON AN AVAILABLE DISC TRACK. LIBRARY ROUTINES ARE "LOADED" OR "LINKED" DEPENDING UPON TYPE. AFTER CONVERSION,THE PROGRAM CAN BE LOADED INTO THE BACKGROUND OR REAL-TIME DISC RESIDENT AREA, AND EXECUTED.

SYSTEM PROGRAM EDIT FEATURES

THE R-T LOADER PROVIDES A METHOD TO PERMANENTLY MODIFY THE SET OF DISC-RESIDENT USER PROGRAMS IN A CONFIGURED SYSTEM. THE USER MAY:

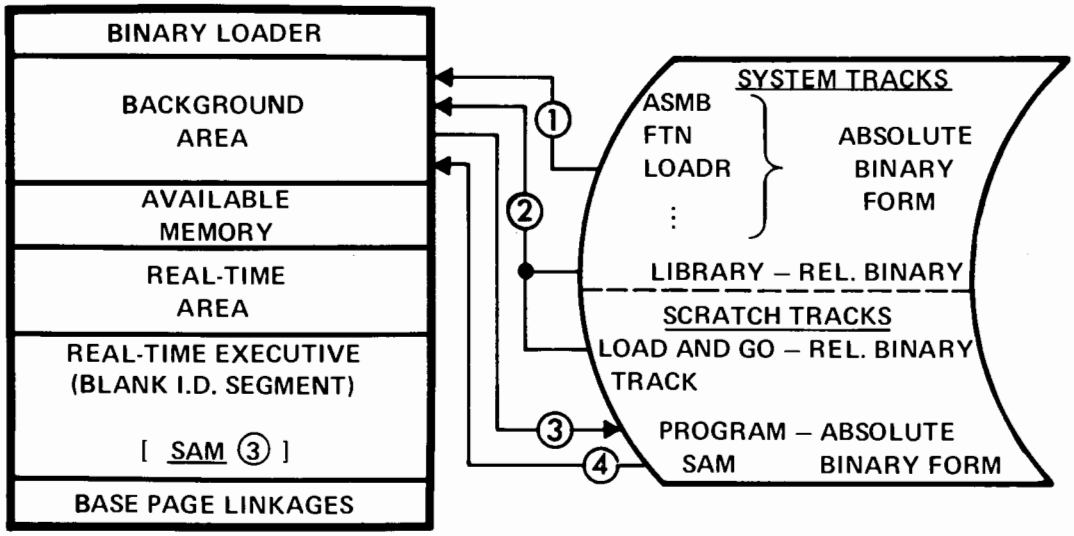
1 – ADD A NEW DISC RESIDENT REAL-TIME, OR BACKGROUND PROGRAM.

2 – REPLACE A USER PROGRAM WITH AN UPDATED VERSION.

LOAD AND GO FEATURE

ALLOWS LOADING OF RELOCATABLE BINARY OBJECT PROGRAMS DEPOSITED IN THE "LOAD AND GO" AREA OF DISC BY FORTRAN OR ASSEMBLER.

LOADER OPERATION DIAGRAM



1 – ON, LOADR PLACES THE LOADER INTO THE BACKGROUND FOR EXECUTION.
2 – THE LOADER PROCESSES THE LOAD & GO DATA AND REFERENCED LIBRARY PROGRAMS.
3 – THE ABSOLUTE BINARY PRODUCED BY THE LOADER IS WRITTEN ON A SCRATCH TRACK, AND THE BLANK I.D. SEGMENT IS FILLED OUT.
4 – ON, SAM LOADS THE PROGRAM INTO THE BACKGROUND FOR EXECUTION.

REAL TIME LOADER OPERATIONS

LOADER INITIATION * ON, LOADR, P1, P2, P3, P4, P5

WHERE

P1, IS THE LOGICAL UNIT # OF THE INPUT DEVICE.
(IF 99, A "LOAD AND GO" OPERATION IS SPECIFIED)

P2, IS THE LOGICAL UNIT # OF THE LIST OUTPUT DEVICE.

P3, SPECIFIES THE "TYPE" OF LOADER OPERATION

= Ø, TEMPORARY SYSTEM ADDITION
= 1, SAME AS Ø, BUT WITH DEBUG OPTION
= 2, PERMANENT ADDITION OR REPLACEMENT. ℗
= 3, LIST ALL SYSTEM PROGRAMS

P4, IS THE STRUCTURE PARAMETER

= Ø, A SINGLE MAIN PROGRAM (W/WO SUBROUTINES)
= 1, A BACKGROUND MAIN/SEGMENT LOAD OPERATION

P5, MEMORY MAP AND BOUNDS LIST PARAMETER

= 1, THE MAP AND BOUNDS LIST ARE SUPPRESSED.

NOTE: IF NOT SPECIFIED, THE LOADER ASSUMES P1 THRU P5 AS FOLLOWS:
* ON, LOADR, 5, 6, Ø, Ø, Ø

℗ DO NOT REPLACE TYPE 80 PROGRAMS OR PROGRAMS NAMED IN THE INTERRUPT TABLE. SUCH PROGRAMS CAN ONLY BE CHANGED AT RTGEN TIME.

REAL TIME LOADER OPERATIONS (CONT.)

IF THE SYSTEM PROGRAM LIBRARY EDITING (P3 = 2) WAS SELECTED, THEN THE LOADER PRINTS:

/LOADR: "GO" WITH EDIT PARAMETERS

THEN YOU MUST

* GO, LOADR, P1, P2, P3

WHERE P1 = 1, ADDITION OF A NEW PROGRAM
2, REPLACEMENT OF A DORMANT PROGRAM

P2 = PROGRAM TYPE
2 = FOREGROUND DISC RESIDENT
3 = BACKGROUND DISC RESIDENT

P3 = PRIORITY, Ø TO 99

NOTE: A GO, LOADR WOULD MEAN

* GO, LOADR, 1, 3, 99

REAL TIME LOADER OPERATIONS (CONT.)

AFTER EACH END-OF-TAPE INDICATION THE LOADER IS SUSPENDED. TO CONTINUE THE LOADING OPERATION THE FOLLOWING COMMAND IS USED:

* GO, LOADR, P1, P2

WHERE:

P1 = n, n IS A CODE DESIGNATING THE NEXT OPERATION
IF n = Ø, LOAD FROM BINARY INPUT UNIT
IF n = 1, LOAD REFERENCED LIBRARY PROGRAMS
IF n = 2, LOAD FROM LOAD-AND-GO AREA
IF n = 3, LIBRARY LOAD FOR THE LAST SEGMENT IN A MAIN/SEGMENT LOAD.

P2 = 1, OMIT THE LIST OF ENTRY POINTS AT THE END OF LOADING

NOTE IF SUSPENSION WAS CAUSED BY A CHECKSUM ERROR OR ILLEGAL RECORD (LØ1, LØ2) THEN * GO, LOADR IS THE APPROPRIATE COMMAND.

LABORATORY EXERCISE GUIDE

LESSON: INTRODUCTION TO HP REAL TIME MULTIPROGRAMMING SYSTEM

Exercise: 1-1

A - OBJECTIVE

1. To provide a practical exercise designed to familiarize the student with the operation and use of a configured Real Time Executive system.

B - PROBLEM

1. On pages I.L.2 thru I.L 4 of this exercise you will find listings of a FORTRAN main program and an Assembly Language Subroutine. The FORTRAN program is error-free, however the Assembly Language Subroutine has three obvious errors. The problem solution involves using the HP Real Time Executive system capabilities to: Compile, Edit, Assemble, Load, and Execute this FORTRAN Main program and Subroutine.
2. Each student group will be judged on their accuracy in using the system and the time expended in completing the job.

C - PROCEDURES

1. The procedure to be followed in this exercise is as follows:

   No tapes are to be punched during this exercise. All editing operations and all binary object files will use the system mass storage device.

   a) Load the system from the mass storage device using the BBDL.

   b) Initialize the Real Time Clock with TM,ØØØ,ØØ,ØØ.

   c) Edit the Assembly Language Subroutine using mass storage as the destination device.

   d) Reserve a Load-and-Go Track.

   e) Assemble the Subroutine using the Logical Source File option for input and the Load-and-Go output option.

   f) If Assembly errors still exist you must repeat steps d and e.

   g) Compile the FORTRAN Main program using paper tape input and the Load-and-Go output option. (Note: Do not re-initialize the Load-and-Go Track).

   h) Use the Loader Load-and-Go option to load the programs.

   i) Execute the programs.

2. Remember, no paper tape.

D - RESULTS

1. Your results should agree with the answers shown on page I.L.5. Retain the system Teleprinter listing and program output listings for discussing the exercise with your instructor.

```
 PAGE 0001


0001                    ASMB,R,L,T

 # 001
OP  0002  NAM TIME

 # 001
NO  0003         ENT WEN

 # 001
SY  0023  TBUFFB BSS 5
WEN    R 000003
.ENTR  X 000001
EXEC   X 000002
HRS    R 000000
MIN    R 000001
SEC    R 000002
TCODE  R 000021
TBUFF  R 000022
**0003 ERRORS*
```

```
 PAGE 0002 #01


0001                    ASMB,R,L,T

PG 000
OP  0002  NAM TIME
0002NAM TIME

PG 002
NO  0003          ENT WEN
0003                      ENT WEN
0004                      EXT .ENTR,EXEC
0005  00000 000000  HRS   BSS 1
0006  00001 000000  MIN   BSS 1
0007  00002 000000  SEC   BSS 1
0008  00003 000000  WEN   NOP
0009  00004 016001X       JSB .ENTR
0010  00005 000000R       DEF HRS
0011  00006 016002X       JSB EXEC
0012  00007 000012R       DEF *+3
0013  00010 000021R       DEF TCODE
0014  00011 000022R       DEF TBUFF
0015  00012 062025R       LDA TBUFF+3
0016  00013 172000R       STA HRS,I
0017  00014 062024R       LDA TBUFF+2
0018  00015 172001R       STA MIN,I
0019  00016 062023R       LDA TBUFF+1

PG 002
M   0020          STA SEC,IN
0020  00017 072002R       STA SEC,IN
0021  00020 126003R       JMP WEN,I
0022  00021 000013  TCODE DEC 11
0023  00022 000000  TBUFFB BSS 5
0024                      END

PG 002
**0003 ERRORS*
```

```
PAGE  0001


FTN,L
      PROGRAM PRIME
      DIMENSION I(5)
      WRITE(6,100)
100    FORMAT(12X,"PRIMES FROM ONE TO FIVE HUNDRED TEN"////
     19X,"1",19X,"2",19X"3")
      J=4
      M=1
15     C=0.0
      DO 65 L=2,510
 40     IF (J-J/L*L)65,60,65
60     C=C+1.
65    CONTINUE
70     IF (C-1.0) 200,77,200
77     I(M)=J
       M=M+1
      IF (M-6) 200,150
150    M=1
      WRITE (6,400) I
400   FORMAT(5(I10))
200    J=J+1
      IF (J-510) 15,15,500
500   CALL WEN(IHR,MIN,ISEC)
      WRITE(6,101) IHR,MIN,ISEC
101   FORMAT(///"  ELAPSED TIME:"I3" HOURS"I3" MINUTES"I3 " SECONDS")
      END
```

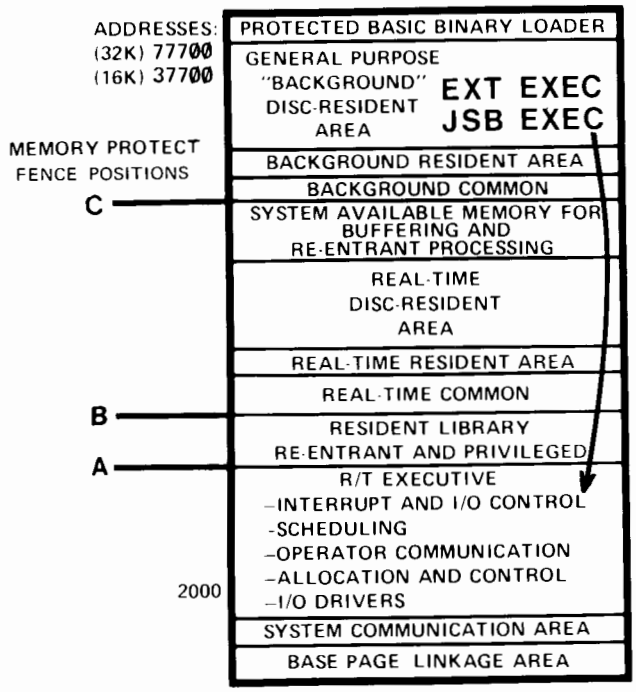
PRIMES FROM ONE TO FIVE HUNDRED TEN

1		2		3
5	7	11	13	17
19	23	29	31	37
41	43	47	53	59
61	67	71	73	79
83	89	97	101	103
107	109	113	127	131
137	139	149	151	157
163	167	173	179	181
191	193	197	199	211
223	227	229	233	239
241	251	257	263	269
271	277	281	283	293
307	311	313	317	331
337	347	349	353	359
367	373	379	383	389
397	401	409	419	421
431	433	439	443	449
457	461	463	467	479
487	491	499	503	509

ELAPSED TIME: 0 HOURS 8 MINUTES 51 SECONDS

USER PROGRAM/EXEC COMMUNICATIONS II

USER/EXECUTIVE COMMUNICATIONS



A USER PROGRAM COMMUNICATES WITH THE R-T EXEC BY FORCING A MEMORY PROTECT VIOLATION TO OCCUR.

FOR EXAMPLE –

IF A BACKGROUND PROGRAM WERE IN EXECUTION THE MEMORY PROTECT FENCE WOULD BE SET AT POINT C ON THE MEMORY MAP. A JSB EXEC INSTRUCTION (AS SHOWN) WOULD FORCE THE MEMORY PROTECT HARDWARE TO INTERRUPT. THE INTERRUPT PROCESSOR WOULD THEN TRANSFER CONTROL TO THE EXECUTIVE WHERE THE INSTRUCTION CAUSING THE MEMORY PROTECT VIOLATION IS ANALYZED.

READ/WRITE

ASSEMBLY LANGUAGE

```
                EXT   EXEC
                 .
                 .
                 .
                JSB   EXEC          (TRANSFER CONTROL TO RTE)
                DEF   * + 5 (OR 7)  (POINT OF RETURN FROM RTE;
                                     7 IS FOR DISC REQUEST)
                DEF   RCODE         (REQUEST CODE)
                DEF   CONWD         (CONTROL INFORMATION)
                DEF   BUFFR         (BUFFER LOCATION)
                DEF   BUFFL         (BUFFER LENGTH)
                DEF   DTRAK         (TRACK NUMBER-DISC TRANSFER ONLY)
                DEF   DSECT         (SECTOR NUMBER-DISC TRANSFER ONLY)
                return point        (CONTINUE EXECUTION)
                 .
                 .
RCODE           DEC   1 (OR 2)      (1 = READ, 2 = WRITE)
CONWD           OCT   conwd         ( conwd IS DESCRIBED BELOW)
BUFFR           BSS   n             (BUFFER OF n WORDS)
BUFFL           DEC   n (OR -2n)    (SAME n; WORDS (+) OR CHARACTERS (-)
DTRAK           DEC   f             (TRACK NUMBER, DECIMAL)
DSECT           DEC   q             (SECTOR NUMBER, DECIMAL)
```

CONWD

A = PUNCH ASCII (TTY)
K = PRINT ASCII (TTY)
M = BINARY
V = VARIABLE LENGTH RECORDS
X = CYCLIC CHECKING
(M.H. DISC ONLY)

0	0	0	0	0	X	A	K	V	M	LOGICAL UNIT #					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

NOTES:

'A' REG = EQT word #5
'B' REG = # of words or characters actually transmitted

IF "PROGRAM" TRIES TO WRITE ON TRACKS NOT ASSIGNED AS "OWN" OR "GLOBAL" THEN "PROGRAM" IS ABORTED. (SEE ERROR MESSAGES.)

I/O CONTROL

PURPOSE

TO PERFORM AUXILIARY OPERATIONS SUCH AS, BACKSPACE, WRITE E.O.F., GENERATE PAPER TAPE LEADER, TRAILER, PAGE EJECT ON LINE PRINTER, ETC.

ASSEMBLY LANGUAGE

```
            EXT     EXEC
             .
             .
             .
            JSB     EXEC        (TRANSFER CONTROL TO RTE)
            DEF     * + 4 (OR 3) (POINT OF RETURN FROM RTE)
            DEF     RCODE       (REQUEST CODE)
            DEF     CONWD       (CONTROL INFORMATION)
            DEF     PARAM       (OPTIONAL PARAMETER)
            RETURN POINT        (CONTINUE EXECUTION)
             .
             .
RCODE       DEC     3           (REQUEST CODE = 3)
CONWD       OCT     conwd       (SEE COMMENTS)
PARAM       DEC     n           (REQUIRED FOR SOME CONTROL FUNCTIONS)
```

CONWD

BITS

0	0	0	0	0	FUNCTION CODE					LOGICAL UNIT #					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

CODE

001 – WRITE E.O.F.
002 – BACKSPACE (RECORD)
003 – FOR. SPACE (RECORD)
004 – REWIND
005 – REWIND (STBY)
006 – DYNAMIC STATUS
007 – E.O.T. (PAPER)
010 – GENERATE LEADER (PAPER)
011 – LINE SPACING (LIST)
000 – NOT USED

FOR DYNAMIC STATUS (006)

'A' REG = DYNAMIC EQT WORD 5 OF SUBCHANNEL

I/O STATUS

PURPOSE

TO REQUEST THE STATUS CONDITION AND TYPE OF EQUIPMENT ASSIGNED TO A GIVEN LOGICAL UNIT NUMBER.

ASSEMBLY LANGUAGE

```
          EXT   EXEC
           :
          JSB   EXEC          (TRANSFER CONTROL TO RTE)
          DEF   * + 4 (OR 5)  (POINT OF RETURN FROM RTE)
          DEF   RCODE         (REQUEST CODE)
          DEF   CONWD         (CONTROL INFORMATION)
          DEF   STAT1         (STATUS WORD 1)
          DEF   STAT2         (STATUS WORD 2 - OPTIONAL)
          RETURN POINT
           :
RCODE     DEC   13            (REQUEST CODE = 13)
CONWD     DEC   n             (LOGICAL UNIT NUMBER)
STAT1     NOP                 (WORD 5 OF EQT ENTRY RETURNED HERE)
STAT2     NOP                 (WORD 4 RETURNED HERE, OPTIONAL)
```

D = DMA INDICATOR
B = BUFFERED OUTPUT INDICATOR
T = TIMED OUT FLAG
AV = AVAILABILITY CODE
0, AVAILABLE 1, DOWN
2, BUSY 3, WAITING FOR A DMA CHANNEL

EQUIPMENT TABLE

WORD									
4	D	B	0	0	T	0	0	UNIT #	CHANNEL #
5	AV		EQUIP CODE					STATUS	

OWN, GLOBAL
DISC ALLOCATION

PURPOSE

TO REQUEST THAT *n* CONTIGUOUS DISC TRACKS BE ASSIGNED

ASSEMBLY LANGUAGE

```
          EXT   EXEC
          :
          JSB   EXEC     (TRANSFER CONTROL TO RTE)
          DEF   * + 6    (POINT OF RETURN FROM RTE)
          DEF   RCODE    (REQUEST CODE)
          DEF   # TRAK   (NUMBER OF contiguous TRACKS REQUIRED)
          DEF   STRAK    (START TRACK NUMBER)
          DEF   DISC     (LOGICAL UNIT NUMBER)
          DEF   SECT #   (NUMBER OF SECTORS/TRACK)
          RETURN POINT   (CONTINUE EXECUTION)
          :
RCODE     DEC   4 OR 15  (4 = OWN, 15 = GLOBAL)
# TRAK    OCT   n        (n = NUMBER OF CONTIGUOUS TRACKS WITHIN
                          THE SAME DISC UNIT REQUESTED. IF BIT 15
                          OF #TRAK = 1, THE PROGRAM IS NOT SUS-
                          PENDED IF TRACKS ARE NOT AVAILABLE; IF
                          BIT 15 = 0, THE PROGRAM IS SUSPENDED
                          UNTIL THE TRACKS ARE AVAILABLE.)
STRAK     NOP          (RTE STORES STARTING TRACK NUMBER
                          HERE, OR -1 IF THE TRACKS ARE NOT AVAIL-
                          ABLE.)
DISC      NOP          (RTE STORES DISC LOGICAL UNIT NUMBER
                          HERE.)
SECT #    NOP          (RTE STORES NUMBER OF SECTORS/TRACK
                          HERE.)
```

'A' AND 'B' REGISTERS ARE NOT AFFECTED.

OWN DISC RELEASE

PURPOSE

TO RELEASE ANY, OR ALL DISC TRACKS CURRENTLY ASSIGNED TO THIS PROGRAM.

ASSEMBLY 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             (5 = RELEASE PROGRAM'S TRACKS)
ITRAK       DEC     n             (IF n = -1, RELEASE ALL TRACKS ASSIGNED
                                   TO PROGRAM, ISTRK AND DISC ARE UN-
                                   NECESSARY, SO THE RETURN POINT IS *+3.
                                   OTHERWISE, n IS THE NUMBER OF CONTIG-
                                   UOUS TRACKS TO BE RELEASED STARTING
                                   AT ISTRK.)
ISTRK       NOP                   (STARTING TRACK NUMBER)
IDISC       NOP                   (MASS STORAGE LOGICAL UNIT)
```

NOTE:

IF THE TRACKS ARE NOT "OWN"(ED), THEN THE "PROGRAM" IS ABORTED. (SEE ERROR MESSAGES.)

'A' AND 'B' REGISTERS ARE NOT AFFECTED.

GLOBAL DISC RELEASE

PURPOSE: TO RELEASE SOME CONTIGUOUS MASS STORAGE TRACKS, PREVIOUSLY ASSIGNED GLOBALLY.

ASSEMBLY LANGUAGE

```
        EXT EXEC
            .
            .
        JSB EXEC        (TRANSFER CONTROL TO RTE)
        DEF *+5         (POINT OF RETURN FROM RTE)
        DEF ICODE       (REQUEST CODE POINTER)
        DEF ITRAK       (NUMBER OF CONTIGUOUS TRACKS POINTER)
        DEF ISTRK       (STARTING TRACK NUMBER POINTER)
        DEF IDISC       (DISC LOGICAL UNIT POINTER)
        (RETURN POINT)
            .
            .
ICODE   DEC   16        GLOBAL TRACK RELEASE
ITRAK   DEC   n         NUMBER OF CONTIGUOUS TRACKS
ISTRK   DEC   m         STARTING TRACK NUMBER
IDISC   DEC   p         DISC LOGICAL UNIT
```

AND, UPON RETURN

'A' REG = 0	TRACKS RELEASED
'A' REG = -1	NO TRACKS RELEASED, IN USE
'A' REG = -2	NO TRACKS RELEASED, NOT GLOBAL

PROGRAM COMPLETION

PURPOSE

TO NOTIFY THE EXEC THAT THE CALLING PROGRAM IS FINISHED AND WISHES TO TERMINATE. THE EXECUTIVE SETS THE PROGRAM DORMANT.

ASSEMBLY LANGUAGE

```
        EXT   EXEC
          :
        JSB   EXEC      (TRANSFER CONTROL TO RTE)
        DEF   * + 2     (RETURN POINT FROM RTE)
        DEF   RCODE     (REQUEST CODE)
        RETURN POINT
          :
RCODE   DEC   6         (REQUEST CODE = 6)
```

PROGRAM SUSPEND

PURPOSE

TO SUSPEND THE CALLING PROGRAM UNTIL RESCHEDULED BY THE OPERATOR.

ASSEMBLY LANGUAGE

```
          EXT   EXEC
            :
          JSB   EXEC      (TRANSFER CONTROL TO RTE)
          DEF   * + 2     (POINT OF RETURN FROM RTE)
          DEF   RCODE     (REQUEST CODE)
          RETURN POINT    (CONTINUE EXECUTION)
            :
            :
RCODE     DEC   7         (REQUEST CODE = 7)
```

'A' & 'B' REG — NOT CHANGED

OR

'B' REG = ADDRESS OF PARAMETERS GIVEN IN GO, name,P_1,P_2,P_3,P_4,P_5 COMMAND OR ZERO, IF NOT SPECIFIED

PROGRAM SEGMENT LOAD

PURPOSE

TO LOAD A BACKGROUND SEGMENT OF THE CALLING PROGRAM FROM THE DISC INTO THE BACKGROUND OVERLAY AREA AND TRANSFER EXECUTION TO THE SEGMENT'S ENTRY POINT.

ASSEMBLY LANGUAGE

```
        EXT   EXEC
          :
        JSB   EXEC        (TRANSFER CONTROL TO RTE)
        DEF   * + 3       (POINT OF RETURN FROM RTE)
        DEF   RCODE       (REQUEST CODE)
        DEF   SNAME       (SEGMENT NAME)
        RETURN POINT      (CONTINUE EXECUTION)
          :
RCODE   DEC   8           (REQUEST CODE = 8)
SNAME   ASC   3,XXXXX     (XXXXX IS THE SEGMENT NAME)
```

NOTE:

ALL "SEGMENTS" MUST REFER TO AT LEAST ONE DECLARED ENTRY POINT IN THE "MAIN" AS EXTERNAL.

PROGRAM SCHEDULE

PURPOSE

TO SCHEDULE A DORMANT PROGRAM FOR EXECUTION, AND OPTIONALLY TO TRANSFER UP TO FIVE PARAMETERS TO THAT PROGRAM.

ASSEMBLY LANGUAGE

```
            EXT   EXEC
             :
            JSB   EXEC        (TRANSFER CONTROL TO RTE)
            DEF   * + 3 + n   (RETURN POINT, n = NUMBER OF PARAMETERS)
            DEF   RCODE       (REQUEST CODE)
            DEF   PNAME       (NAME OF PROGRAM TO SCHEDULE)
            DEF   p1          (UP TO FIVE OPTIONAL PARAMETERS
             :                 
            DEF   p5          )
            RETURN POINT      (CONTINUE EXECUTION)
             :
RCODE       DEC   9 (OR 10)   (9 = SCHEDULE WITH WAIT,
                               10 = NO WAIT)
PNAME       ASC   3,XXXXX     (XXXXX IS THE NAME OF THE PROGRAM TO
                               SCHEDULE.)
p1          
                              (UP TO FIVE OPTIONAL PARAMETERS)
p5
```

NOTE: "WAIT" (CODE 9) INSURES THAT THE SCHEDULED PROGRAM RUNS TO COMPLETION BEFORE THE CALLING PROGRAM RESUMES EXECUTION.

CALLING PROGRAM, UPON RE-ENTRY:

'A' REG = STATUS OF PROGRAM BEING SCHEDULED WHEN SCHEDULED

CALLED PROGRAM, UPON ENTRY:

'B' REG = ADDRESS OF P1 THRU P5 OR ZERO, IF NOT SPECIFIED

TIME REQUEST

PURPOSE

TO REQUEST THE CURRENT TIME RECORDED IN THE REAL TIME CLOCK.

ASSEMBLY LANGUAGE

```
        EXT     EXEC
         :
        JSB     EXEC        (TRANSFER CONTROL TO RTE)
        DEF     * + 3       (POINT OF RETURN FROM RTE)
        DEF     RCODE       (REQUEST CODE)
        DEF     ARRAY       (TIME VALUE ARRAY)
        RETURN POINT        (CONTINUE EXECUTION)
RCODE   DEC     11          (REQUEST CODE = 11)
ARRAY   BSS     5           (TIME VALUE ARRAY)
```

ARRAY CONTENTS

(1) TENS OF MILLISECONDS
(2) SECONDS
(3) MINUTES
(4) HOURS
(5) DAY OF THE YEAR

EXECUTION TIME (INITIAL OFFSET VERSION)

PURPOSE

TO SCHEDULE A PROGRAM FOR EXECUTION AT SPECIFIED TIME INTERVALS, STARTING AFTER SOME INITIAL OFFSET TIME. THE EXECUTIVE PLACES THE SPECIFIED PROGRAM IN THE TIME LIST AND RETURNS.

ASSEMBLY LANGUAGE

```
         EXT   EXEC
          :
         JSB   EXEC       (TRANSFER CONTROL TO RTE)
         DEF   * + 6      (POINT OF RETURN FROM RTE)
         DEF   RCODE      (REQUEST CODE)
         DEF   PROG       (PROGRAM TO BE PUT IN TIME LIST)
         DEF   RESL       (RESOLUTION CODE)
         DEF   MTPLE      (EXECUTION MULTIPLE)
         DEF   OFFSET     (INITIAL TIME OFFSET)
         RETURN POINT     (CONTINUE EXECUTION)
                :
                :
RCODE    DEC   12         (REQUEST CODE = 12)
         NOP              (PUT CALLING PROGRAM IN TIME LIST)
PROG     OR
         ASC   3,XXXXX    (XXXXX IS THE PROGRAM TO BE PUT IN THE
                           TIME LIST.)
RESL     DEC   X          (X IS THE RESOLUTION CODE.)
MTPLE    DEC   Y          (Y IS THE EXECUTION MULTIPLE.)
OFFSET   DEC   -Z         (X = UNITS) GIVES THE INITIAL OFFSET; THE
                           NEGATIVE SIGN SIGNALS THIS CALL VERSION
                           TO RTE.)
```

EXECUTION TIME (ABSOLUTE START TIME)

PURPOSE

TO SCHEDULE A PROGRAM FOR EXECUTION AT A PARTICULAR ABSOLUTE TIME. THE EXECUTIVE PLACES THE SPECIFIED PROGRAM IN THE TIME LIST AND RETURNS.

ASSEMBLY LANGUAGE

```
          EXT   EXEC
           :
          JSB   EXEC
          DEF   * + 9
          DEF   RCODE
          DEF   PROG
          DEF   RESL
          DEF   MTPLE
          DEF   HOURS
          DEF   MINS
          DEF   SECS
          DEF   MSECS
          (RETURN)
           :
RCODE     DEC 12
PROG      NOP OR ASC 3,XXXXX
RESL      DEC X
MTPLE     DEC Y
HOURS     DEC A  )
MINS      DEC B  )        ABSOLUTE STARTING TIME IN
SECS      DEC C  )        HOURS, MINUTES, SECS AND
MSECS     DEC D  )        10'S OF MILLISECONDS ON A
                          24 HOUR CLOCK.
```

EXECUTIVE ERROR CODES (PART 1)

WHEN THE EXECUTIVE DISCOVERS AN ERROR IN AN EXEC CALL, IT MAY TERMINATE THE PROGRAM; RELEASE ANY DISC TRACKS ASSIGNED TO THE PROGRAM; PRINT AN ERROR MESSAGE ON THE OPERATOR CONSOLE, (name ABORTED); AND PROCEED TO EXECUTE THE NEXT PROGRAM IN THE SCHEDULED LIST.

MEMORY PROTECT ERRORS

MEMORY PROTECT VIOLATIONS THAT ARE NOT CALLS TO THE EXEC CAUSE THE FOLLOWING MESSAGE:

MP name address (address IS VIOLATING INSTRUCTION ADDRESS)

REQUEST CODE ERRORS

RQ name address (address IS THE ADDRESS OF THE ILLEGAL REQUEST

THE GENERAL FORM OF OTHER ERROR CODES

type name address

WHERE: type = A 4 CHARACTER ERROR CODE
name = THE PROGRAM NAME
address = THE ADDRESS OF THE CALL

EXECUTIVE ERROR CODES (PART 2)

ERROR CODES FOR SCHEDULE CALLS (SCHED)

SCØ 1 = MISSING PARAMETER
SCØ 2 = ILLEGAL PARAMETER
SCØ 3 = 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 I/O CALLS (RTIOC)

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.

IOØ8 = DISC TRANSFER LONGER THAN TRACK BOUNDARY
IOØ9 = OVERFLOW OF LOAD-AND-GO AREA

ERROR CODES FOR DISC ALLOCATION CALLS (EXEC)

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

LABORATORY EXERCISE GUIDE

LESSON: REAL-TIME SYSTEM REQUESTS

Exercise: 2-1

A - OBJECTIVE

1. To provide a simple but meaningful "first" program using the system requests.

B - PROBLEM

1. A message coded in ASCII has been written somewhere on the disc. You are to write a program that will find this message and print the message correctly on the list output device.
2. The complete message is stored on one track. The correct track contains the ASCII code for "HP" in the word 0 of sector 0 and also in the word 0 of the last sector. Each of the 10 lines are stored on a sector within that track. The word 0 of each correct sector contains the ASCII number of that line. The total message contains 62 characters, the first 2 of which are line sequence numbers that should be stripped off prior to printing.

C - PROCEDURE

1. Use Assembly language or FORTRAN to create a program that will provide a solution to the problem outlined in B.
2. Use the Load-and-Go option for binary output.
3. Remember any program can Read from an 'own' Disc track but only the "holder" of the track may Write on it.
4. When the program prints the correct message it should make a completion call.
5. The program should be made Background Disc resident.
6. The line number is right justified on the Disc. b1 (Text), b2 (Text), . . . 10 (Text).
   Note that b = ASCII code for space.

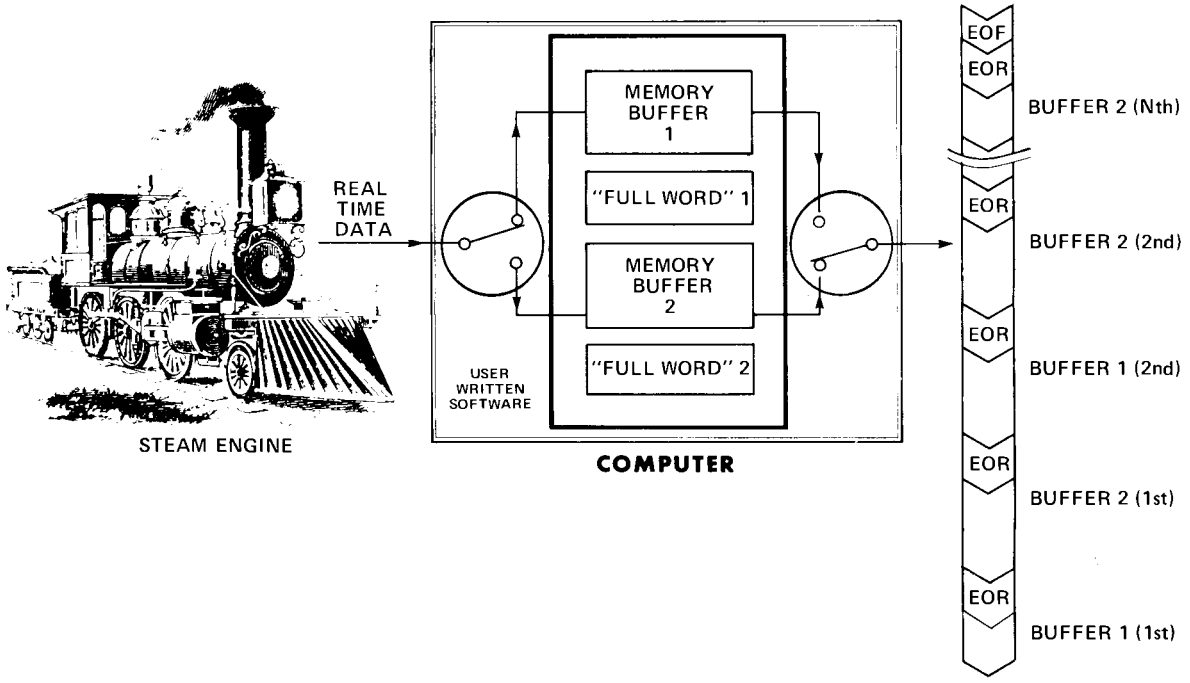
D - RESULTS

1. The results should compare exactly with the results on page II.L.2.

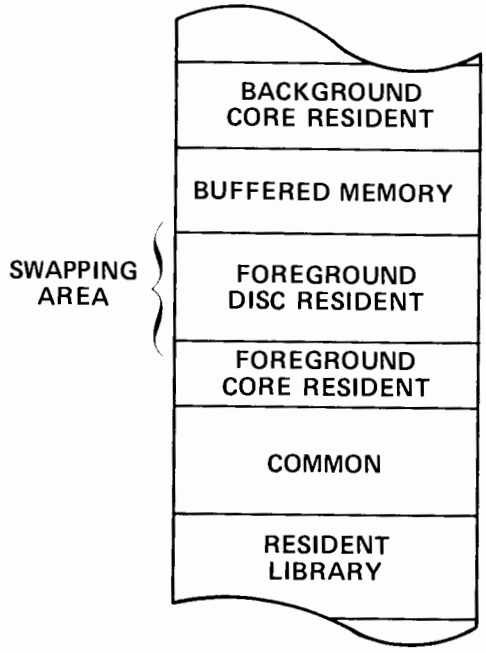
```
1  THE HP 2005B REAL-TIME EXECUTIVE CUSTOM TAILORS AN HP
2  2116B COMPUTER AND DISC STORAGE INTO A TRUE REAL-TIME
3  MULTI-PROGRAMMING SYSTEM.
4
5  PROGRAMS FOR THE 2005B CAN BE WRITTEN USING THE HP RT
6  ASSEMBLY LANGUAGE OR HP REAL-TIME FORTRAN. THE NUMBER OF
7  PROGRAMS IN THE SYSTEM IS LIMITED ONLY BY DISC CAPACITY
8  AND THE DYNAMICS OF THE APPLICATION. CORE MEMORY IS MINI-
9  MIZED BY CONFIGURING A TAILORED SYSTEM FROM STANDARD
10 MODULES.
```

PROGRAMMING TECHNIQUES III

THE THRUPUT PROBLEM



WHERE TO PUT THE BUFFERS



CONSIDER A FOREGROUND DISC RESIDENT PROGRAM 'A' MAKING AN INPUT/OUTPUT REQUEST FROM A NON-COMMON MEMORY AREA TO A NON-BUFFERED DEVICE.

1 — THE PROGRAM 'A' IS PLACED IN I/O SUSPEND STATUS (2)

2 — THE FOREGROUND DISC RESIDENT AREA IS "FROZEN" BY PROGRAM 'A' UNTIL THE NON-COMMON MEMORY AREA IS EMPTY (I.E., COMPETITION OF I/O REQUEST)

NOW CONSIDER A FOREGROUND DISC RESIDENT PROGRAM 'A' MAKING AN INPUT/OUTPUT REQUEST USING A COMMON MEMORY AREA TO A NON-BUFFERED DEVICE.

1 — THE PROGRAM 'A' IS PLACED IN I/O SUSPEND STATUS (2)

2 — THE FOREGROUND DISC RESIDENT AREA IS AVAILABLE FOR SWAPPING SINCE THE I/O BUFFER IS IN A CORE RESIDENT AREA, COMMON MEMORY

OUTPUT TO A BUFFERED DEVICE WILL REQUIRE MORE OVERHEAD FOR SERVICING THE OUTPUT REQUEST AND:

IF (THE BUFFER HAS ENOUGH SPACE
FOR [DATA BLOCK + ID SEGMENT])
THEN (IMMEDIATE COMPLETION OF
OUTPUT REQUEST)
ELSE (MEMORY SUSPEND (4));

WHERE TO PUT THE PROGRAM(S)

BACKGROUND DISC RESIDENT - SEE SLIDE I.1.2.3

1 – BACKGROUND PROGRAMS CANNOT DIRECTLY ACCESS FOREGROUND COMMON

2 – BACKGROUND DISC RESIDENT PROGRAMS CAN ONLY BE SEGMENTED, NOT SWAPPED

3 – BACKGROUND DISC RESIDENT PROGRAMS USUALLY HAVE A LARGE PROGRAM AREA SUITABLE FOR ANALYSIS, NOT REAL TIME

4 – ALL PROGRAMS AND SEGMENTS MUST RUN TO COMPLETION OR BE ABORTED

FOREGROUND DISC RESIDENT - SEE SLIDE I.1.2.2

1 – IF SWAPPING IS TO BE USED, THEN WHEN PROGRAM "A" SUSPENDS ITSELF, IT SHOULD ALLOW ENOUGH TIME TO SWAP PROGRAM 'A', LOAD PROGRAM 'B', SWAP PROGRAM 'B', LOAD PROGRAM 'A' SO AS NOT TO TIE UP THIS PROGRAM AREA

2 – PROGRAMS IN I/O SUSPEND MAY NOT BE SWAPPED UNLESS THE DATA BLOCK IS IN COMMON MEMORY

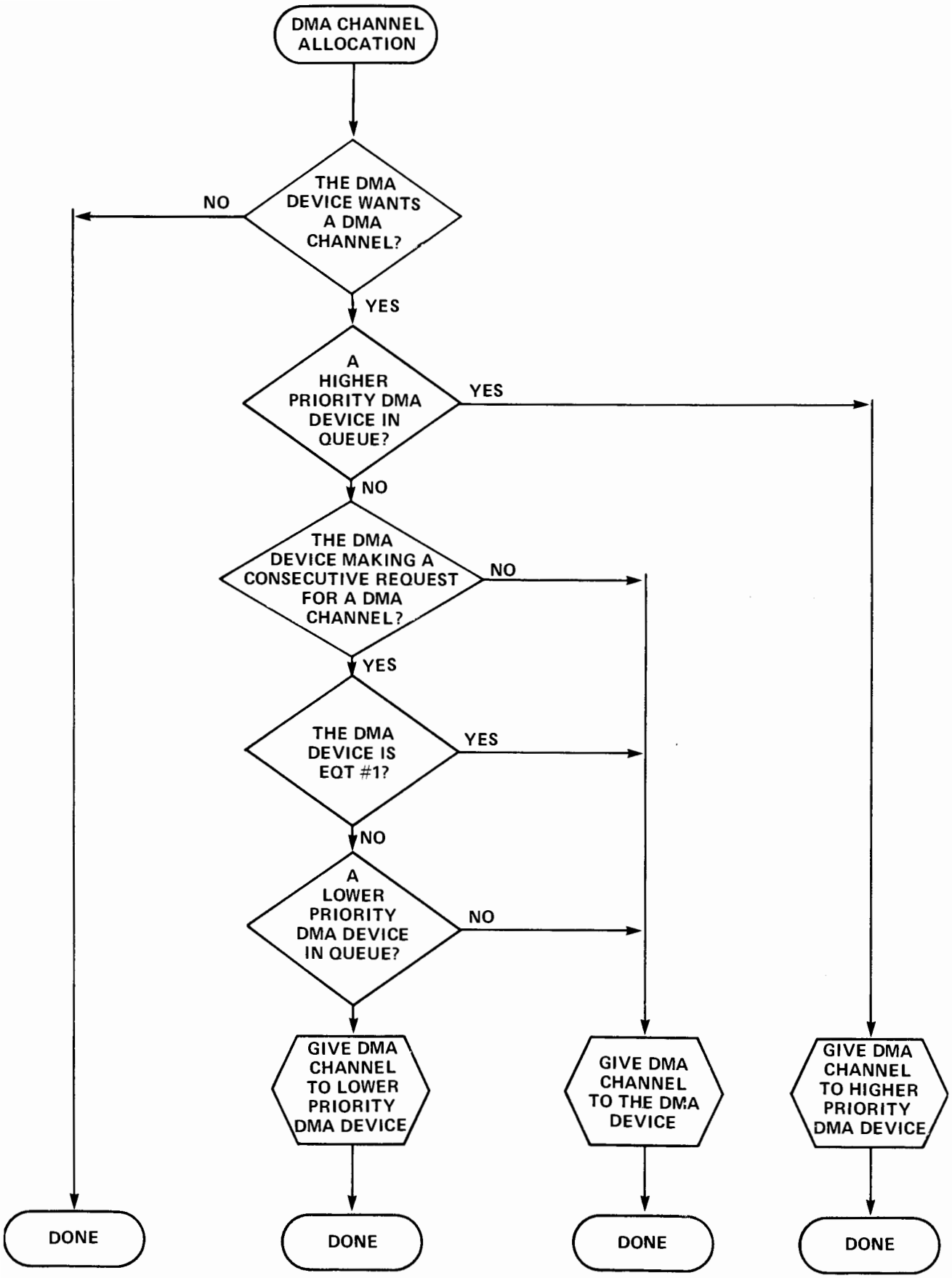
CORE RESIDENT - SEE SLIDES I.1.2.2 AND I.1.2.3

1 – REQUIRES NON-OVER-LAY-ABLE CORE ALLOCATION AT RTGEN TIME

2 – PROGRAMS MAY NOT BE REPLACED OR DELETED AT RUN TIME WITHOUT DESTROYING THE SYSTEM

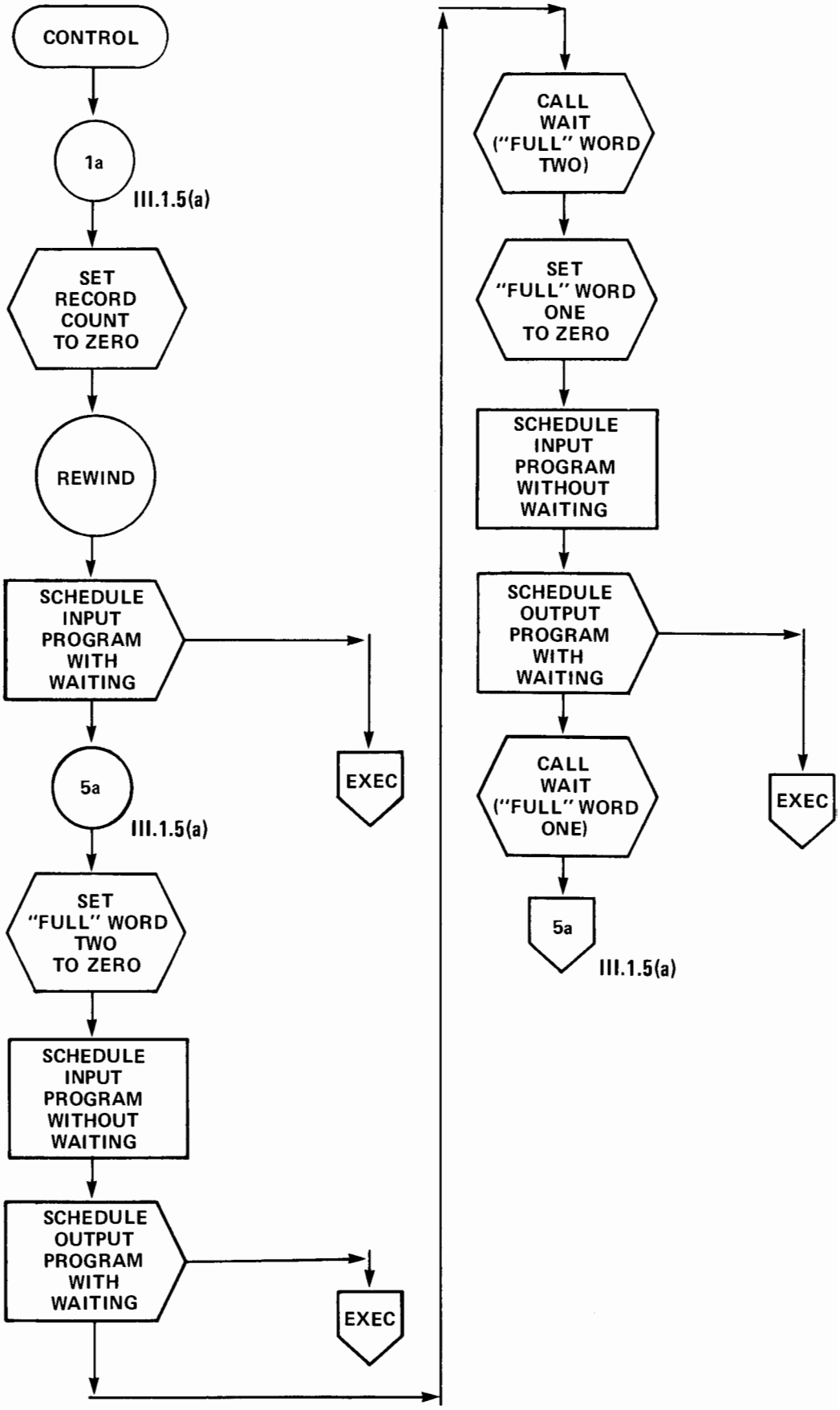
```
IF (THE DMA DEVICE REQUESTS A DMA CHANNEL)
THEN IF (A HIGHER PRIORITY DMA DEVICE HAS REQUESTED A DMA CHANNEL)
        THEN (GIVE THE HIGHER PRIORITY DMA DEVICE A DMA CHANNEL)
        ELSE IF (THE DMA DEVICE IS MAKING A CONSECUTIVE REQUEST FOR
                A DMA CHANNEL)
            THEN IF (THE DMA DEVICE IS EQT #1)
                THEN (GIVE THE DMA DEVICE A DMA CHANNEL)
                ELSE IF (A LOWER PRIORITY DMA DEVICE HAS REQUESTED
                        A DMA CHANNEL)
                    THEN (GIVE THE LOWER PRIORITY DMA DEVICE
                            A DMA CHANNEL)
                    ELSE (GIVE THE DMA DEVICE A DMA CHANNEL)
            ELSE (GIVE THE DMA DEVICE A DMA CHANNEL)
ELSE (FORGET IT);
```

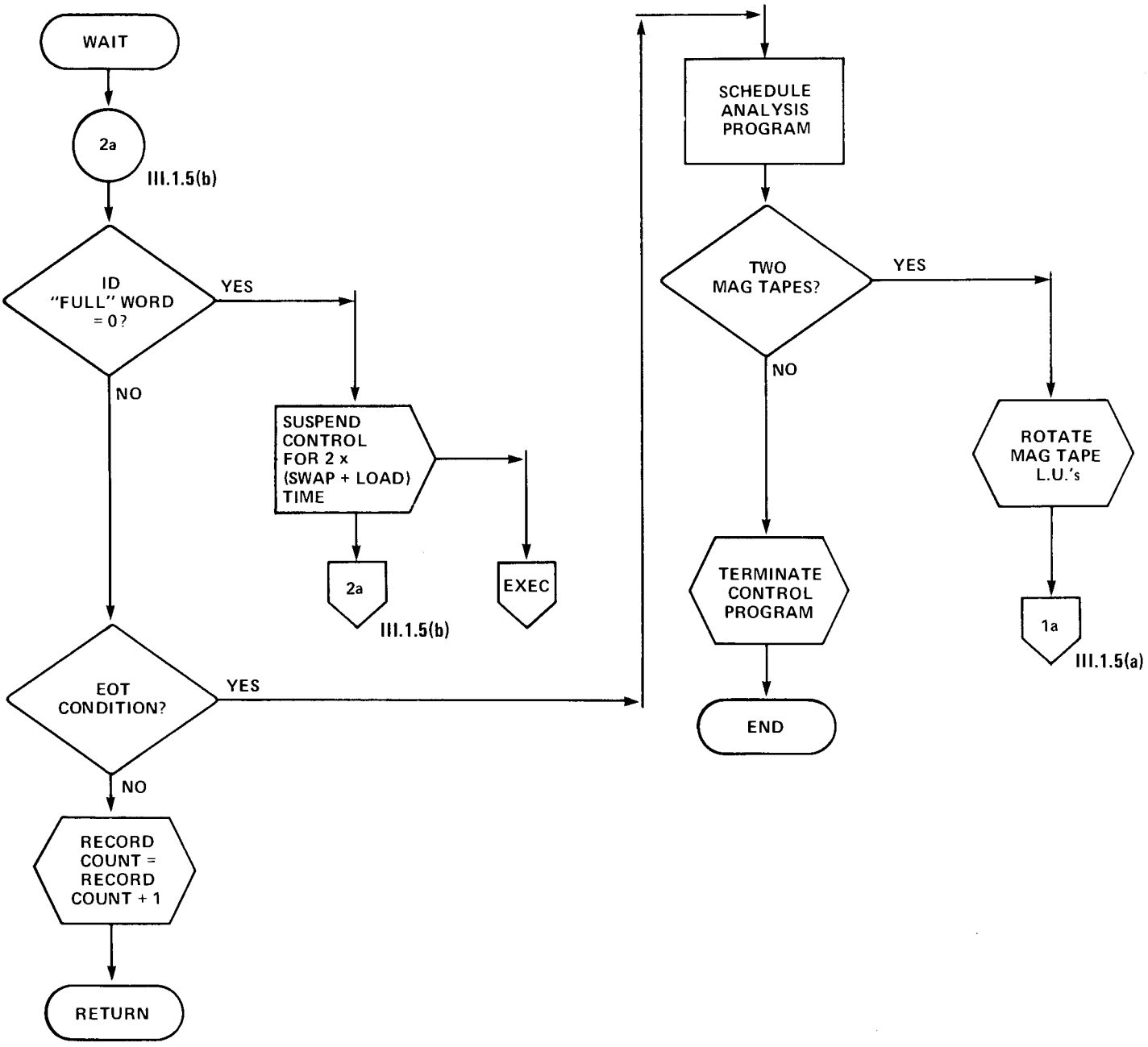
DMA CHANNEL ALLOCATION ALGORITHM (A)



DMA CHANNEL ALLOCATION ALGORITHM (B)

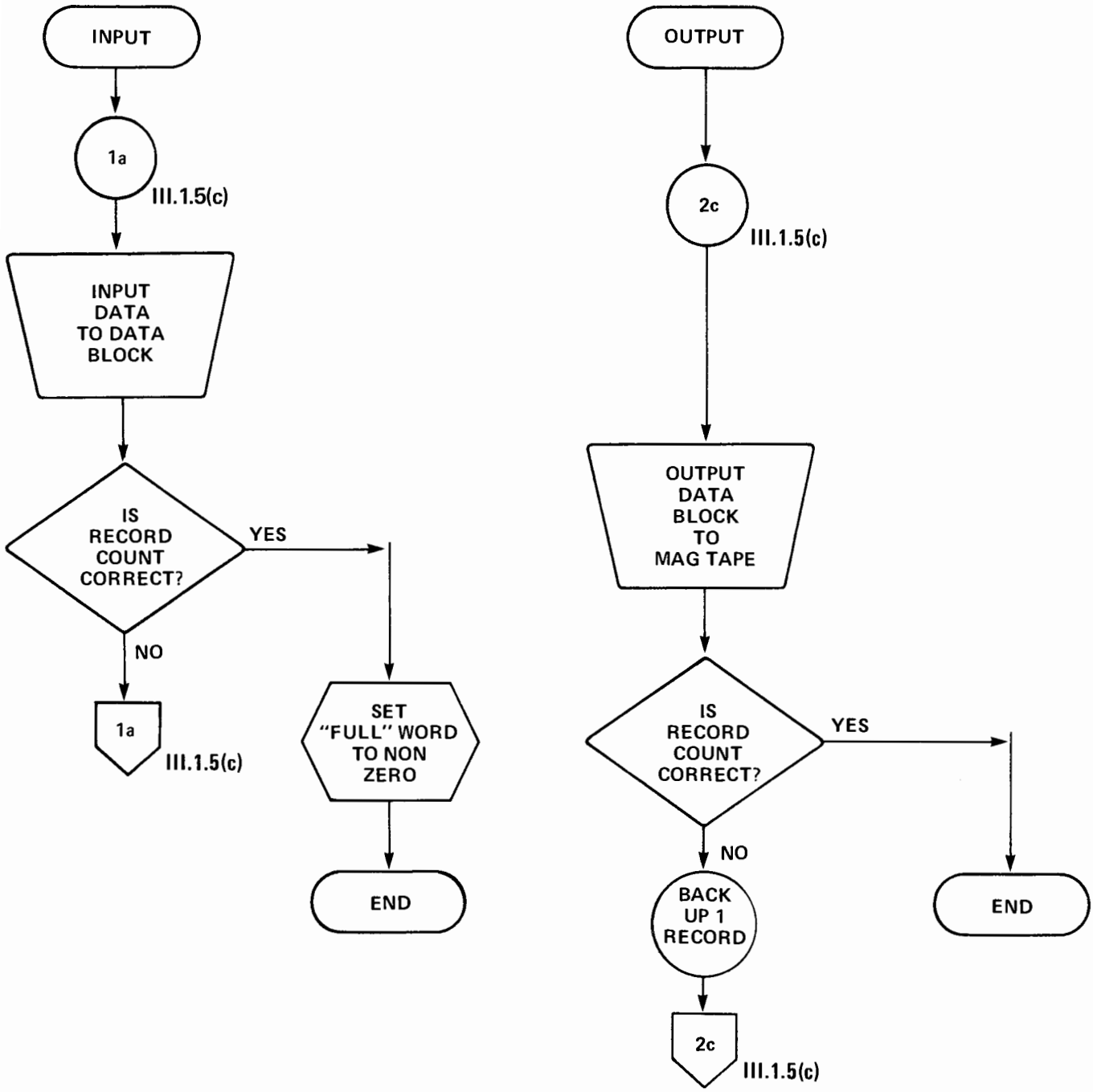
THRUPUT PROGRAMS (A)





THRUPUT PROGRAMS (B)

THRUPUT PROGRAMS (C)



STATE / EVENT	DORMT	SCHED	IO SUSP	DISC SUSP	MEMRY SUSP	OPER SUSP	TIME LIST	SWAP-PING	LOAD-ING	EXE-CUTING	WAIT LIST	TERMI-NATE	DMA 1	DMA 2	INTER-RUPT
0	CIO														
1	IO	C													
2	IO			C					C				DISC		
3	IO	C								C					DISC
4	IO		C										MT		
5	IO	C								C					MT
6	O	CI								I	C_I				
7	O	C	I								C_I		ADC		
8	O	CI								I	C_I				ADC
9	IO	C								C		I			

PROGRAM	TYPE	PRIORITY	CODE
CONTROL	2	1	C
INPUT	1	2	I
OUTPUT	1	3	O

DEVICE	EQT #
DISC	1
MT	8
ADC	10

STATE / EVENT	DORMT	SCHED	IO SUSP	DISC SUSP	MEMRY SUSP	OPER SUSP	TIME LIST	SWAP-PING	LOAD-ING	EXE-CUTING	WAIT LIST	TERMI-NATE	DMA 1	DMA 2	INTER-RUPT
10	O	CI								C					
11		CIO								I	C_O				
12		CO	I							O	C_O		ADC		
13		C	IO								C_O		ADC	MT	
14		CI	O							I	C_O			MT	ADC
15	I	C	O								C_O	I		MT	
16	I	CO								O	C_O				MT
17	IO	C								C		O			
18	IO	C								C			MT*		
19	G	O		T	O		S	T	E	P		#	1	O	

PROGRAM	TYPE	PRIORITY	CODE
CONTROL	2	1	C
INPUT	1	2	I
OUTPUT	1	3	O

DEVICE	EQT #
DISC	1
MT	8
ADC	10

THRUPUT STATUS CHART (B)

DVR56 REAL-TIME EXECUTIVE DRIVER
WITH
2310A/B SUBSYSTEM (A)

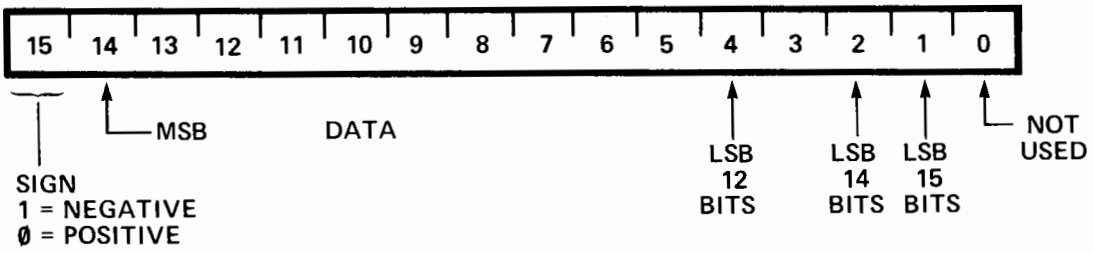
```
EXT EXEC
    .
    .
    .
JSB EXEC
DEF*+7
DEF ONE
DEF IDRT
DEF IBUFF
DEF N
DEF ICHAN
DEG ICODE
```

DEFINITIONS OF PARAMETERS

ONE	A binary one. This is the request code to both the driver and the RTE software. The latter interprets a request code of one as an I/O - READ request and passes the call list parameters to the driver in the proper format for DVR56.
IDRT	The device reference number assigned to the subsystem.
IBUFF	The address of the data storage buffer.
N	The number of conversions required on this call. (N $\leq$ length of IBUFF)
ICHAN	The multiverter channel number for a single channel measurement (digitize mode), or the starting channel number for a sequential scan. The parameter can either be decimal (0-63) or octal (0-77).
ICODE	The multiverter mode of operation.
6 =	Digitize Mode (monitoring single channel)
7 =	Sequential Mode

DVR56 REAL-TIME EXECUTIVE DRIVER WITH 2310A/B SUBSYSTEM (B)

INPUT DATA WORD BIT STRUCTURE



DVR56 REAL-TIME EXECUTIVE DRIVER
WITH
2310C SUBSYSTEM (A)

```
      EXT EXEC
       .
       .
       .
      JSB EXEC
      DEF *+7
      DEF ONE
      DEF IDRT
      DEF IBUFF
      DEF N
      DEF ICHAN
      DEF ICODE
```

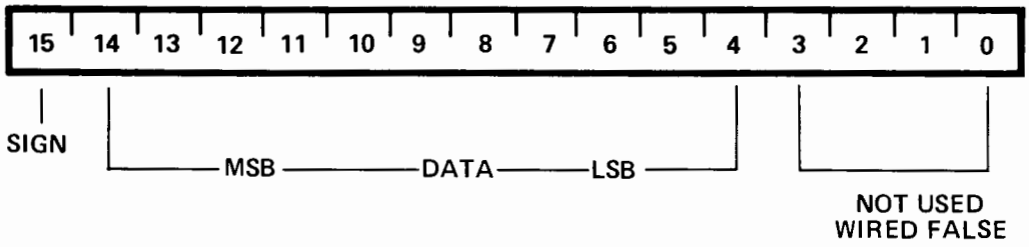
DEFINITIONS OF PARAMETERS

ONE	A binary one. This is the request code to both the driver and the RTE software. The latter interprets a request code of one as an I/O - READ request and passes the call list parameters to the driver in the proper format for DVR56.
IDRT	The device reference number assigned to the subsystem.
IBUFF	The address of the data storage buffer.
N	The number of conversions required on this call. (N $\leq$ length of IBUFF)
ICHAN	The converter channel number for a single channel measurement (digitize mode), or the starting channel number for a 2310 sequential scan. (The 2311 always starts at channel zero in sequential mode.) The parameter can be either decimal (0-63) or octal (0-77).

ICODE		The converter mode of operation.
0	=	2311 Digitize DMA encode
1	=	2311 Digitize with pacer
2	=	2311 Sequential DMA encode
3	=	2311 Sequential with pacer
4	=	2311 Digitize free-run
5	=	2311 Sequential free-run
6	=	2310 Digitize
7	=	2310 Sequential

DVR56 REAL-TIME EXECUTIVE DRIVER WITH 2310C SUBSYSTEM (B)

INPUT DATA WORD FORMAT



DVR56 REAL-TIME EXECUTIVE DRIVER WITH 2311A SUBSYSTEM(A)

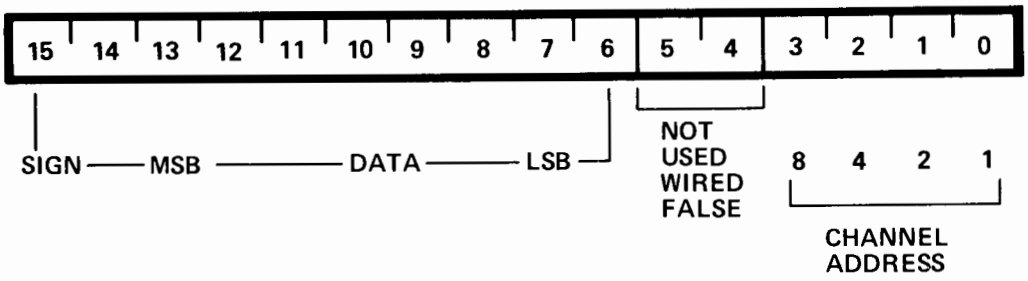
```
EXT EXEC
  .
  .
  .
JSB EXEC
DEF*+7
DEF ONE
DEF IDRT
DEF IBUFF
DEF N
DEF ICHAN
DEF ICODE
```

DEFINITIONS OF PARAMETERS

ONE	A binary one. (This is the request code to both the driver and the RTE software. The latter interprets a request code of one as an I/O - READ request and passes the call list parameters to the driver in the proper format for DVR56.)
IDRT	The device reference number assigned to the subsystem.
IBUFF	The address of the data storage buffer.
N	The number of conversions required on this call. (N $\leq$ length of IBUFF)
ICHAN	The channel number for a single channel measurement. The parameter can be either decimal (0-15) or octal (0-17). (Used only in random mode but must be filled with an integer value in sequential mode.)
ICODE	The 5610A Converter mode of operation. 0 = Single channel measurement (random mode) using DMA issued encodes. 1 = Single channel measurement (random mode) using external pacer-issued encodes. 2 = Sequential mode using DMA issued encodes. 3 = Sequential mode using external pacer-issued encodes. 4 = Single channel measurement (random mode) using 5610A's free run-issued encodes. 5 = Sequential mode using 5610A's free run-issued encodes.

DVR56 REAL-TIME EXECUTIVE DRIVER WITH 2311A SUBSYSTEM (B)

INPUT DATA WORD BIT FORMAT



DVR55 REAL-TIME EXECUTIVE DRIVER (A)

```
        JSB     EXEC
        DEF     *+6
        DEF     RCODE
        DEF     CONWD
        DEF     BUFFR
        DEF     LENTH
        DEF     PWPTR
        .
        .
        .
PWPTR   DEF     PWBUF
```

DEFINITIONS OF PARAMETERS

RCODE	is the request code (=1 for read).
CONWD	is the control word (device reference number).
BUFFR	is the data storage buffer.
LENTH	is the length of the buffer.
PWPTR	is a pointer to the program word buffer.
PWBUF	is the program word buffer.

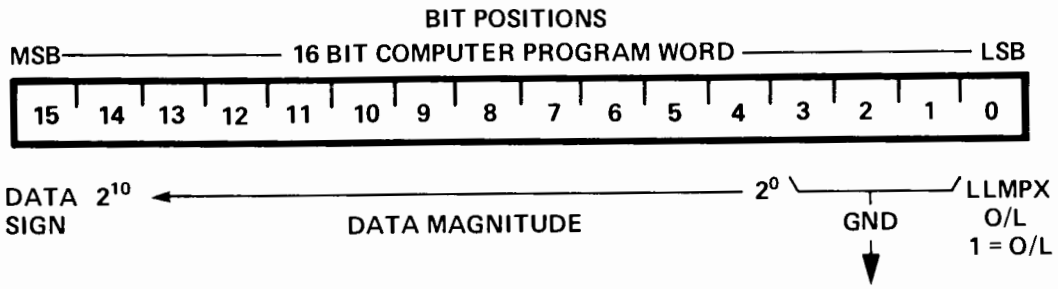
CONVERSION

$$V_{in} = \frac{X}{16}\ \frac{5}{G} \text{ millivolts}$$

where:
- V_{in} = Voltage at HP 2930A input
- G = HP 2930A measurement gain
- X = Decimal translation of the 16-bit data word.
- 5 = Miniverter bit value of 5 millivolts
- 16 = First 4 bits of data word. Bits 0 -3 carry no voltage data.

DVR55 REAL-TIME EXECUTIVE DRIVER (B)

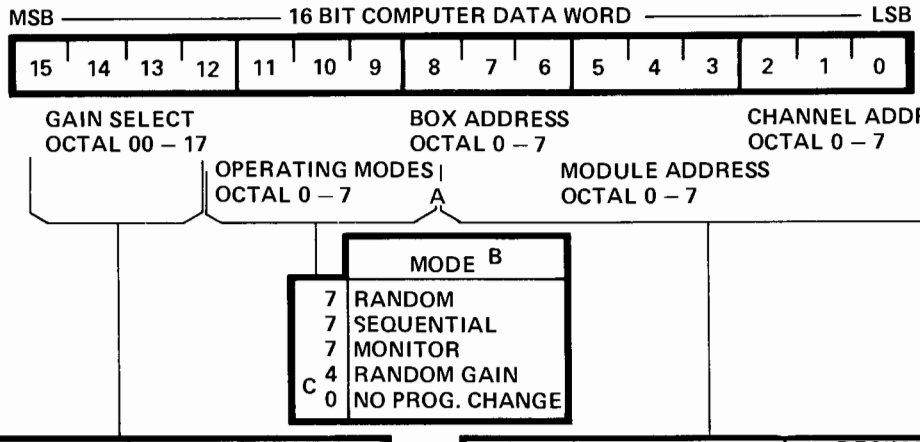
DATA WORD



DVR55 REAL-TIME EXECUTIVE DRIVER (C)

PROGRAM WORD

BIT POSITIONS



OCTAL GAIN CODE	LLMPX GAIN	FULL-SCALE INPUT
00	1	10.24V
01	2	5.12V
02	4	2.56V
03	8	1.28V
04	16	640 mV
05	32	320 mV
06	64	160 mV
07	128	80 mV
10	256	40 mV
11	512	20 mV
12-17	1024	10 mV

2930A BOX NO.	OCTAL ADDRESS RANGE	DECIMAL ADDRESS RANGE
0	000 – 077	000 – 063
1	100 – 177	064 – 127
2	200 – 277	128 – 191
3	300 – 377	192 – 255
4	400 – 477	256 – 319
5	500 – 577	320 – 383
6	600 – 677	384 – 447
7	700 – 777	448 – 511

DVR76 REAL-TIME EXECUTIVE DRIVER WITH HP 2320A AND HP 2322A (A)

```
EXT EXEC
  .
  .
JSB EXEC
DEF *+7
DEF ONE
DEF IDRT
DEF IDATA
DEF L
DEF ICHAN
DEF IPROG
JSB   CONV
DEF   * + 2      Return Pointer
DEF   DATA       Data Pointer
```

DEFINITIONS OF PARAMETERS

ONE — A binary one. (This is the request code to both the driver and the RTE software. The latter interprets a request code of one as an I/O - READ request and passes the call list parameters to the driver in the proper format for DVR76.)

IDRT — The device reference number assigned to the subsystem.

IDATA — The name of the data buffer into which the DSI data is to be stored. (The buffer must be at least two words long — the length is specified by the next parameter, L.)

L — The length of the data buffer.

If L < 2, the READ request is rejected by the driver.

If L = 2, the 32 bits of bcd data from the DSI are stored as:

IDATA	D4	D3	D2	D1
IDATA+1	R	F	D6	D5

If L > 2, the 32 bits are stored as in the L = 2 case, and additionally the function bits are stored as the third buffer word to facilitate checking for DVM overload:

IDATA	D4	D3	D2	D1
IDATA+1	R	F	D6	D5
IDATA+2				F

(Dn is a bcd digit, R is the DVM Range, and F is the DVM Function.)

ICHAN — The scanner channel number.

IPROG — The instrument program code word — a binary number coded as defined.

DVR76 REAL-TIME EXECUTIVE DRIVER
WITH
HP 2320A AND HP 2322A (B)

2320A SUBSYSTEM PROGRAM WORD

NOT USED				DELAY (MS)				MODE		FUNCTION			RANGE		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				USE NOT RECOMMENDED	000 001 010		NOT USED	MEAS. +CAL – –CAL	000 001 010 011	AC VOLTS – FREQUENCY –	000 001 010 011		AUTORANGE – .1V 1V/1KΩ	000 001 010 011	
				27 42 62 145 500	011 100 101 110 111				100 101 110 111	DC VOLTS RESISTANCE – –	100 101 110 111		10V/10KΩ 100V/100KΩ 1KV/1MΩ 10MΩ	100 101 110 111	

2322A SUBSYSTEM PROGRAM WORD (IPGM)

	NOT USED				DELAY (MS)				SAMPLE PER. CODE		FUNCTION			RANGE		
OCTAL	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0 1 2 3 4 5 6 7					SHORTER DELAYS NOT RECOMMENDED 27 42 62 145 500			NOT USED	0 1 2	1.00S 0.10S 0.01S	0 1 2 3 4 5	AC NORMAL AC FAST FREQUENCY PERIOD DC VOLTS OHMS		0 AUTORANGE 1 +10 GAIN, 2411A 2 .1V 3 1V 4 10V 5 100V 6 1000V 7 10 MEGOHM		

DVR74 REAL-TIME EXECUTIVE DRIVER FOR HP2321A DATA ACQUISITION SUBSYSTEM (A)

```
EXT EXEC
     .
     .
     .
JSB EXEC
DEF *+7
DEF ONE
DEF IDRT
DEF IDATA
DEF L
DEF ISCAN
DEF IPROG

JSB    C2321
DEF *+2        Return Pointer
DEF DATA       Data Pointer
```

DEFINITIONS OF PARAMETETERS

ONE	is the request code to the RTE for a read operation.
IDRT	is the device reference number assigned to the 2321A DSI card, at system generation.
IDATA	is a two or three word buffer that will receive the reading in the order. LO-HALF;HI-HALF;FUNCTION The buffer must be at least two words long, the length is specified by the next parameter, L.
L	is a 3 for the three-word buffer as above, or 2 for a two-word buffer, in which case the function is not returned.
ISCAN	is the scanner program word with bits 0 thru 11 the decimal channel number and bits 12 thru 15 the scanner delay code.
IPROG	is the 3450A program word as defined in the tables.

DVR74 REAL-TIME EXECUTIVE DRIVER
FOR
HP2321A DATA ACQUISITION SUBSYSTEM(B)

3450A PROGRAM WORD (IPROG)

15	14 13 12	11 10 9	8 7 6	5 4	3 2 1	0
	MISC PGM	RATIO RANGE*	NON-RATIO RANGE*	FUNCTION*		
NOT USED	RC 1000 1/60S GATE 0100 10MEG INPUT Z 0010 100MS DELAY 0001	X1 000 X10 001 X100 010 X1000 011 AUTO 1--	10MΩ 0001 1MΩ, 1KV 0010 100KΩ, 100V 0011 10KΩ, 10V 0100 1KΩ, 1V 0101 100Ω, 100MV 0110 AUTO 1---	DC 000 AC 001 OHMS 010 DC RATIO 100 AC RATIO 101 OHMS RATIO 110		NOT USED

16-BIT SCANNER PROGRAM WORD (ISCAN)

	NOT USED										FUNCTION			DELAY (MS)		
OCTAL	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0 1 2 3 4 5 6 7											NOT USED	AC/DC 00 FREQ 01 RESIST 10 –		15 000 17.5 001 22 010 27 011 42 100 62 101 145 110 500 111		

DVR77 REAL-TIME EXECUTIVE DRIVER
FOR
HP 2323A SUBSYSTEM (A)

```
EXT EXEC
    .
    .
    .
JSB EXEC
DEF *+7
DEF ONE
DEF IDRT
DEF IDATA
DEF L
DEF ICHAN
DEF IPROG

JSB     CONV
DEF     *+2         Return Pointer
DEF     DATA        Data Pointer
```

DEFINITIONS OF PARAMETERS

ONE A binary one. (This is the request code to both the driver and the RTE software. The latter interprets a request code of one as an I/O - READ request and passes the call list parameters to the driver in the proper format for DVR77.)

IDRT The device reference number assigned to the subsystem. [

IDATA The name of the data buffer into which the DSI data is to be stored. (The buffer must be either two or three words long. The length is specified by the next parameter, L.)

L The length of the data buffer (2 or 3 words).

If L < 2, the READ request is rejected by the driver.

If L = 2, the 32 bits of BCD data from the DSI are stored as:

IDATA	D4	D3	D2	D1
IDATA+1	R	F	D6	D5

If L = 3, the 32 bits of BCD data are stored in the first two buffer words (see above), and additionally the function bits are stored in the third buffer word to facilitate checking for DVM overload.

	IDATA	D4	D3	D2	D1
	IDATA+1	R	F	D6	D5
*	IDATA+2				F

(Dn is a BCD digit, R is the DVM Range, and F is the DVM Function.)

ICHAN The scanner channel number.

IPROG The instrument program code word; a binary number coded as defined in the table.

DVR77 REAL-TIME EXECUTIVE DRIVER FOR HP 2323A SUBSYSTEM (B)

OCTAL	NOT USED	SCANNER DELAY	MODE	FUNCTION	RANGE
	15 14 13 12	11 10 9	8 7 6	5 4 3	2 1 0
0		1.5ms 0 0 0	Measure 0 0 0	AC Volts 0 0 0	Autorange 0 0 0
1		5ms 0 0 1	CAL+ 0 0 1	–	–
2		20ms 0 1 0	–	Frequency 0 1 0	0.1 volt (dc only) 0 1 0
3		100ms 0 1 1	CAL- 0 1 1	–	1.0 volt 0 1 1
4				DC volts 1 0 0	10 volt 1 0 0
5			–		100 volt 1 0 1
6			–		1000 volt 1 1 0

LABORATORY EXERCISE GUIDE

LESSON: PROGRAMMING TECHNIQUES

Exercise: 3-1

A – OBJECTIVE

1. To provide a deeper insight to the possible states a program may achieve.
2. To give the student a vehicle for solving complicated programming situations in the real-time environment.

B – PROBLEM DEFINITION

1. On pages III.1.5(a) thru III.1.5(b) are flow charts of the thruput problem.
2. From these flow charts, determine the status achievable by the "Control", "Input"; and "Output" programs, when all three are foreground disc resident (Type 2).
3. Chart the status using the charts supplied with this exercise.
4. Determine the allocation of DMA channels 1 and 2 during each state, using the DMA channel allocation flow chart and algorithm, page III.1.4(b)
5. Determine when devices interrupt or DMA interrupts.

C – PROCEDURE

1. Refer to pages III.1.6(a) thru III.1.6(b) of the lecture material for a guideline.
2. Using this example, re-construct the blank status charts to solve the lab problem.
3. Use the blank status charts provided or prepare a general purpose status diagram.
4. Determine the priorities the programs should have, and the EQT numbers the devices should have, prior to filling out the status charts.

STATE / EVENT	DORMT	SCHED	IO SUSP	DISC SUSP	MEMRY SUSP	OPER SUSP	TIME LIST	SWAP-PING	LOAD-ING	EXE-CUTING	WAIT LIST	TERMI-NATE	DMA 1	DMA 2	INTER-RUPT
0															
1															
2															
3															
4															
5															
6															
7															
8															
9															

PROGRAM	TYPE	PRIORITY	CODE

DEVICE	EQT #

STATUS CHART

EVENT / STATE	0	1	2	3	4	5	6	7	8	9
DORMT										
SCHED										
IO SUSP										
DISC SUSP										
MEMRY SUSP										
OPER SUSP										
TIME LIST										
SWAP-PING										
LOAD-ING										
EXE-CUTING										
WAIT LIST										
TERMI-NATE										
DMA 1										
DMA 2										
INTER-RUPT										

PROGRAM	TYPE	PRIORITY	CODE

DEVICE	EQT #

STATE / EVENT	DORMT	SCHED	IO SUSP	DISC SUSP	MEMRY SUSP	OPER SUSP	TIME LIST	SWAP-PING	LOAD-ING	EXE-CUTING	WAIT LIST	TERMI-NATE	DMA 1	DMA 2	INTER-RUPT
0															
1															
2															
3															
4															
5															
6															
7															
8															
9															

PROGRAM	TYPE	PRIORITY	CODE

DEVICE	EQT #

STATUS CHART

STATE / EVENT	DORMT	SCHED	IO SUSP	DISC SUSP	MEMRY SUSP	OPER SUSP	TIME LIST	SWAP-PING	LOAD-ING	EXE-CUTING	WAIT LIST	TERMI-NATE	DMA 1	DMA 2	INTER-RUPT
0															
1															
2															
3															
4															
5															
6															
7															
8															
9															

PROGRAM	TYPE	PRIORITY	CODE

DEVICE	EQT #

STATE / EVENT	DORMT	SCHED	IO SUSP	DISC SUSP	MEMRY SUSP	OPER SUSP	TIME LIST	SWAP-PING	LOAD-ING	EXE-CUTING	WAIT LIST	TERMI-NATE	DMA 1	DMA 2	INTER-RUPT
0															
1															
2															
3															
4															
5															
6															
7															
8															
9															

PROGRAM	TYPE	PRIORITY	CODE

DEVICE	EQT #

STATUS CHART

LIST and TABLES IV

SCHEDULER IDLE MODE

THE FOLLOWING INSTRUCTIONS ARE A PART OF THE SCHEDULER MODULE:

IDLE LOOP

```
CCA  }
CCB  }   SET A AND B TO ALL 1's
STF  Ø   ENABLE INTERRUPTS
STC  5   ENABLE MEMORY PROTECT
JMP  *
```

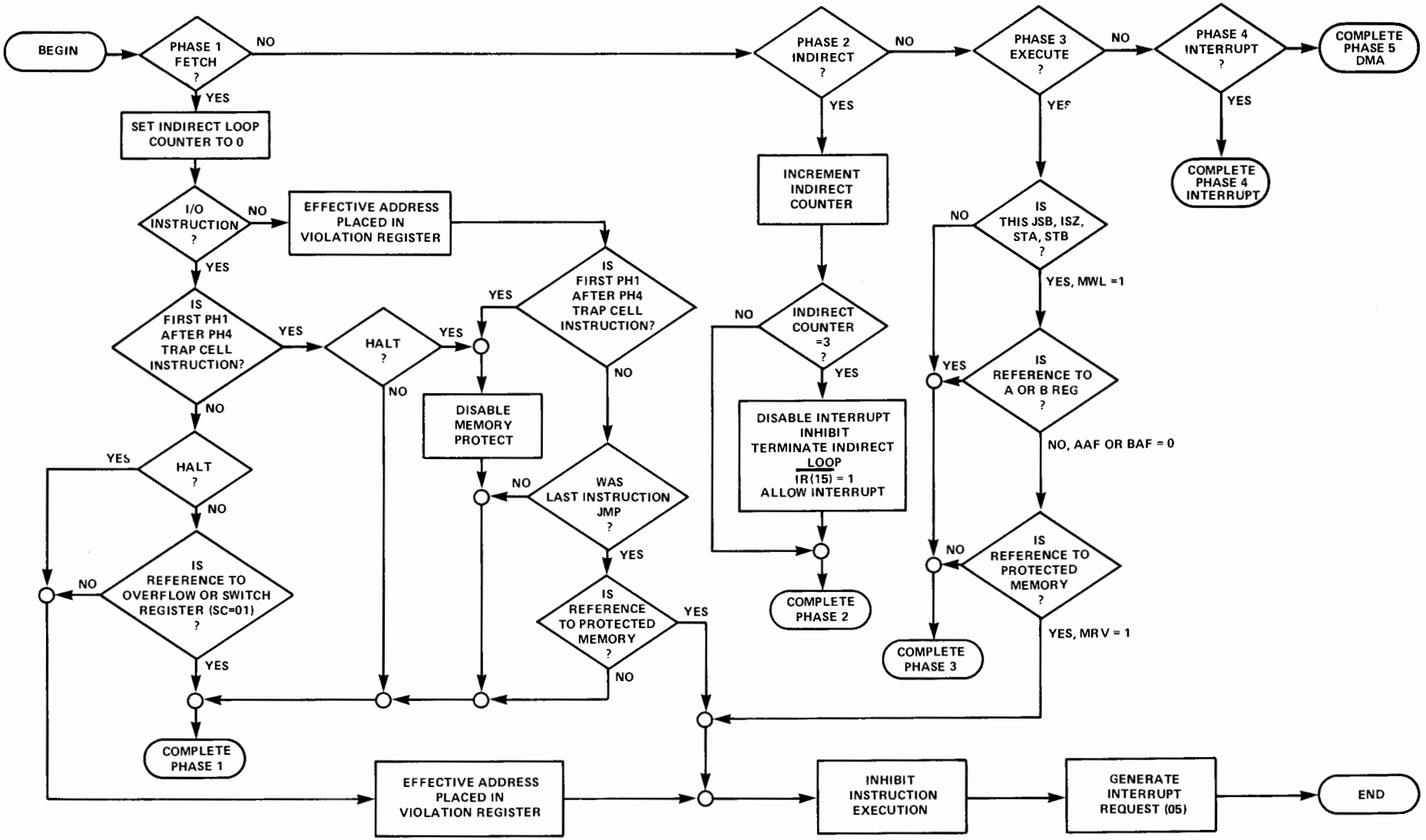
MODIFICATION (2116 SERIES)

```
STF  Ø   ENABLE INTERRUPTS
STC  5   ENABLE MEMORY PROTECT
LIA  1   GET SWITCH REGISTER
LDB  A,I GET MEMORY CONTENTS
JMP  *-2 LOOP
```

THIS CODING SEQUENCE IS THE POINT OF EQUILIBRIUM IN A REAL TIME SYSTEM. ONLY INTERRUPTS CAN DRIVE THE COMPUTER OUT OF THIS IDLE LOOP.

INTERRUPT SOURCES

1 – TIME BASE GENERATOR
2 – POWER FAIL
3 – MEMORY PROTECT VIOLATION
4 – OPERATOR CONSOLE
5 – I/O DEVICES
6 – PARITY ERROR



HP2116 SERIES
MEMORY PROTECT
HP12581B

BASE PAGE POINTERS (1)

SYSTEM TABLES

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 COMMUNICATIONS

01660	EQT1	
01661	EQT2	ADDRESSES
01662	EQT3	OF
01663	EQT4	
01664	EQT5	
01665	EQT6	FIRST
01666	EQT7	11-WORDS
01667	EQT8	OF
01670	EQT9	CURRENT
01671	EQT10	EQT
01672	EQT11	ENTRY
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 COMMUNICATIONS

01676	RQCNT	NO. OF REQUEST PARAMETERS -1
01677	RQRTN	RETURN POINT ADDRESS
01700	RQP1	
01701	RQP2	ADDRESSES OF REQUEST
01702	RQP3	
01703	RQP4	PARAMETERS
01704	RQP5	
01705	RQP6	(SET FOR MAXIMUM OF 8 PARAMETERS)
01706	RQP7	
01707	RQP8	

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

EXECUTING PROGRAM ID SEGMENT

01717	XEQT	ID SEGMENT ADDR. OF CURRENT PROG.
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

BASE PAGE POINTERS(2)

SYSTEM MODULE FLAGS

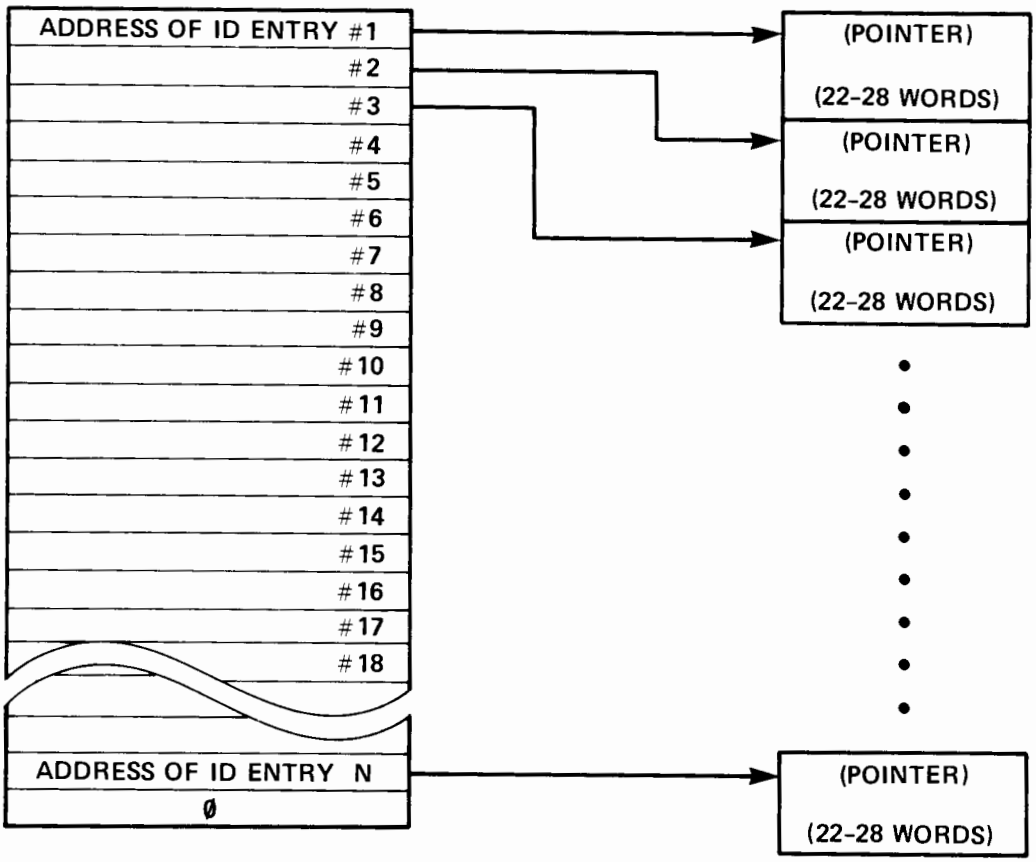
01734	OPATN	OPERATOR/KEYBOARD ATTENTION FLAG
01735	OPFLG	OPERATOR COMMUNICATION FLAG
01736	SWAP	RT DISC RESIDENT SWAPPING FLAG
01737	DUMMY	I/O ADDRESS OF DUMMY INT. CARD
01740	IDSDA	DISC ADDR. OF FIRST ID SEGMENT
01741	IDSDP	-POSITION WITHIN SECTOR

MEMORY ALLOCATION BASES

01742	BPA1	FWA R/T DISC RES. BP LINK AREA
01743	BPA2	LWA R/T 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 REAL-TIME AREA
01747	RTCOM	LENGTH OF REAL-TIME COMMON AREA
01750	RTDRA	FWA OF R/T 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
01755	TATLG	LENGTH OF TRACK ASSIGNMENT TABLE
01756	TATSD	# OF TRACKS ON SYSTEM DISC

UTILITY PARAMETERS

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 PROGS
01764	DSCUN	# OF RELOC UTILITY PROGS
01765	LGOTK	LOAD-N-GO: LU, STG TRACK, # OF TRKS
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	ADDRESSES 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



KEYWORD TABLE

PROGRAM ID ENTRY

TEMP WORD #	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
* 1	(LIST LINKAGE)											X	L	I	N	K
* 2												X	T	E	M	P
* 3	(A		5		W	O	R	D								
* 4		B	L	O	C	K		U	S	E	D					
* 5			F	O	R		P	A	S	S	I	N	G			
* 6					P	A	R	A	M	E	T	E	R	S	)	
7	(PRIORITY)											X	P	R	I	O
8	(ENTRY POINT)											X	P	E	N	T
* 9	(REGISTERS											X	S	U	S	P
* 10	AT TIME											X	A			
* 11	OF											X	B			
* 12	SUSPENSION)											X	E	O		
13	N A M E (1)											N	A	M	E	(2)
14	N A M E (3)											N	A	M	E	(4)
15	N A M E (5)								a					T	Y	P E
* 16				b	c			d			e			S	T	A T
* 17	(TIME LIST LINKAGE)											T	L	I	N	K
* 18	RESL	R					M	U	L	T	I	P	L	E		
* 19	(NEXT											T	M	S	E	C
* 20	EXECUTION											T	S	E	C	
* 21	TIME)											T	M	I	N	
* 22												T	H	O	U	R
23	(FOR									M	E	M	(	1	)	
24	DISC									M	E	M	(	2	)	
25	RESIDENTS									M	E	M	(	3	)	
26	ONLY)									M	E	M	(	4	)	
27														D	M	A N
* 28														S	M	A N

a : = IF Set
THEN BG ONLINE
ELSE PERMANENT

b : = IF Set
THEN WAITING FOR
SCHEDULED
PROGRAM
COMPLETION

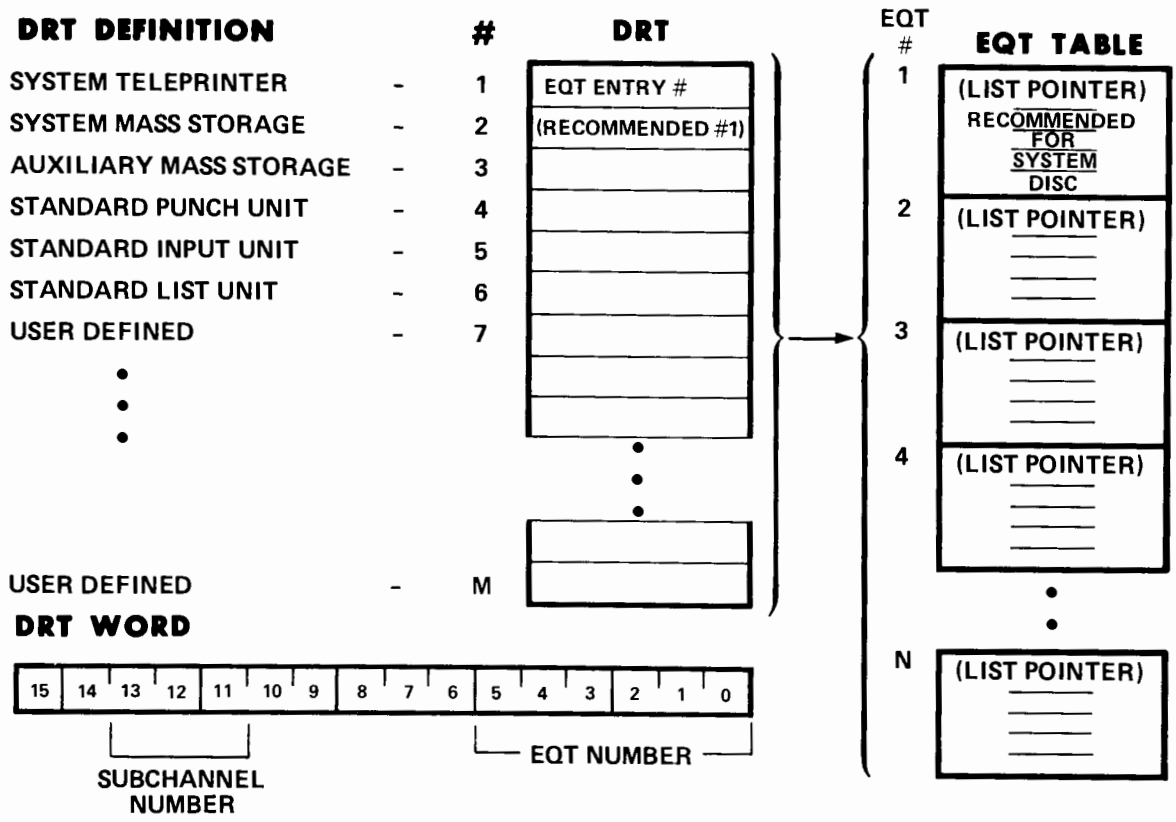
c : = IF Set
THEN PROGRAM
TO BE ABORTED
AFTER CURRENT
SUSPENSION

d : = IF Set
THEN PROGRAM TO BE
OP SUSP AFTER
CURRENT SUSPENSION

e : = IF Set
THEN PROGRAM TO BE
MADE DORMANT
AFTER CURRENT
SUSPENSION

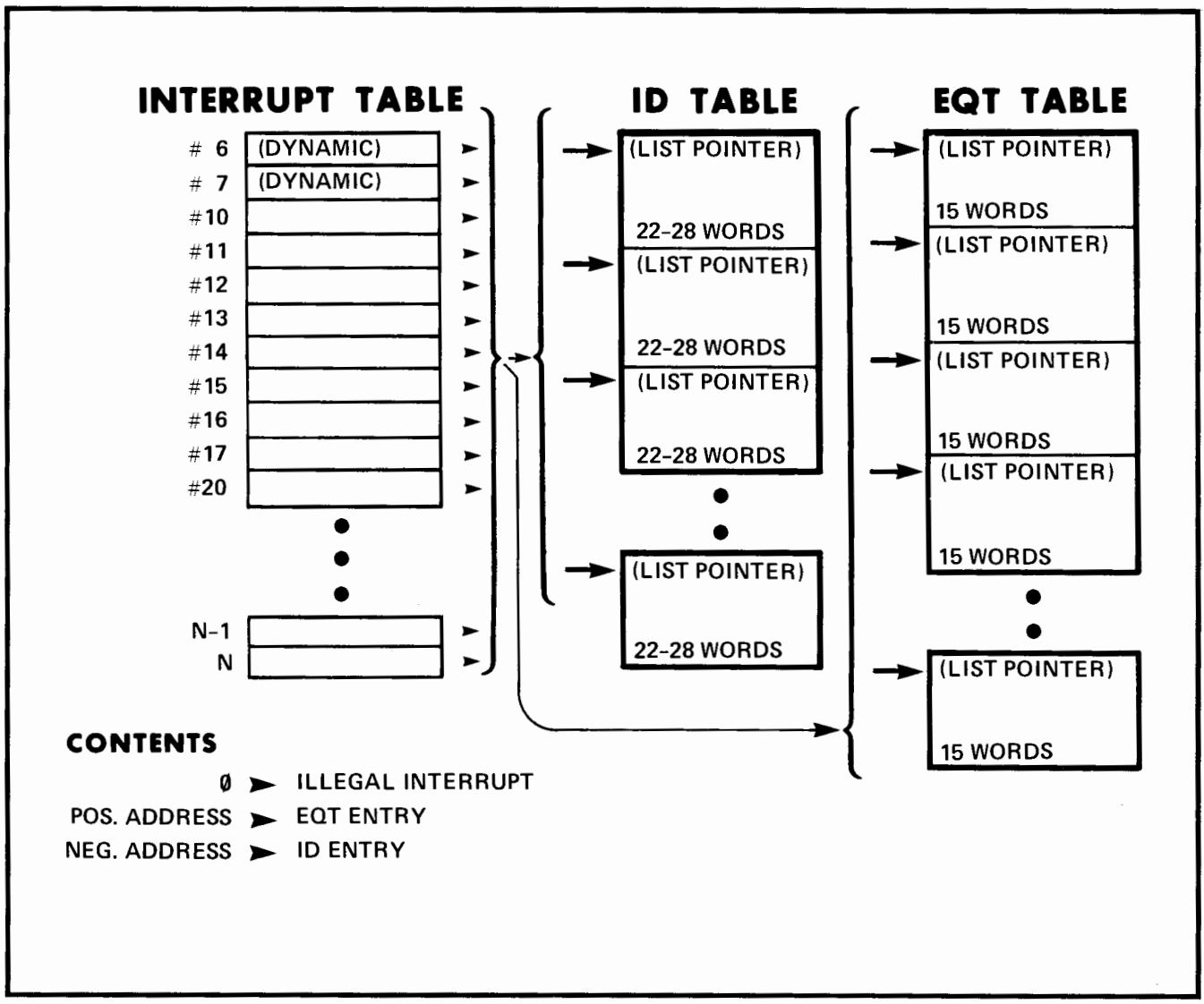
f : = IF Set
THEN PROGRAM IS
IN THE
TIME LIST

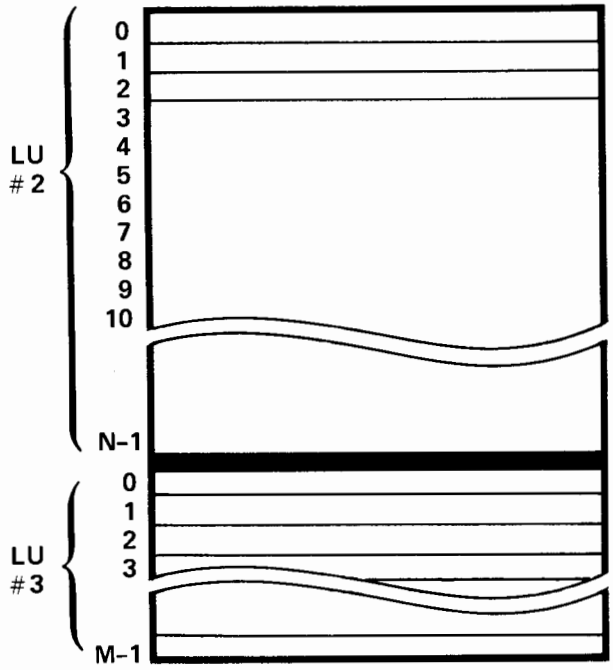
DEVICE REFERENCE TABLE



EQT ENTRY

WORD #	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		SUBCHANNEL #		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 CONTROL WORD															
10	TEMPORARY, DISC SECTOR #, OR CONTROL WORD															
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															


INTERRUPT TABLE
ID TABLE
EQT TABLE
6 (DYNAMIC)
7 (DYNAMIC)
#10
#11
#12
#13
#14
#15
#16
#17
#20
N-1
N
(LIST POINTER)
22-28 WORDS
(LIST POINTER)
22-28 WORDS
(LIST POINTER)
22-28 WORDS
(LIST POINTER)
22-28 WORDS
(LIST POINTER)
15 WORDS
(LIST POINTER)
15 WORDS
(LIST POINTER)
15 WORDS
(LIST POINTER)
15 WORDS
(LIST POINTER)
15 WORDS
CONTENTS
Ø ➤ ILLEGAL INTERRUPT
POS. ADDRESS ➤ EQT ENTRY
NEG. ADDRESS ➤ ID ENTRY



TAT ENTRY

ONE WORD PER TRACK, THE CONTENTS OF WHICH MIGHT BE:

1) | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

: = SYSTEM TRACK
: = PROTECTED TRACK
: = SWAP TRACK
: = PROGRAMS LOADED ON LINE
: = LOAD AND GO TRACKS

2) | ID SEGMENT'S ADDRESS |

: = OWN TRACK

3) | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

: = GLOBAL TRACK

★ LOAD AND GO TRACKS MUST BE CONTIGUOUS ON THE SAME LOGICAL UNIT.

★ EDIT TRACKS NEED NOT BE CONTIGUOUS SINCE THEY ARE ALLOCATED INDIVIDUALLY

$0 < N+M \leq 512$

TRACK ASSIGNMENT TABLE

PROGRAM STATES

WORD 16 OF THE PROGRAM I.D. SEGMENT INDICATES THE CURRENT STATUS OF THAT PROGRAM. TO SAY IT ANOTHER WAY, THE STATUS WORD INDICATES WHAT "LIST", OR "QUEUE" THE PROGRAM IS CURRENTLY IN.

THERE ARE 6 MAIN "LISTS" OR PROGRAM STATES

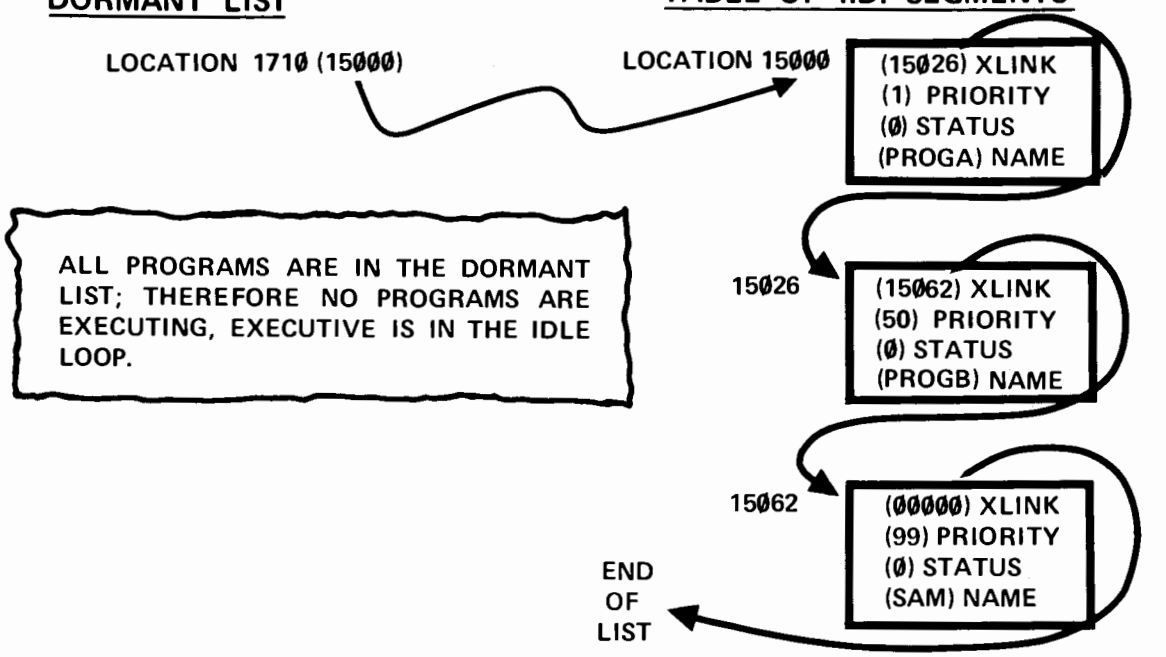
STATUS	MEANING
Ø	DORMANT
1	SCHEDULED
2	I/O SUSPEND (ONE LIST FOR EACH EQT ENTRY)
3	(NOT USED)
4	UNAVAILABLE MEMORY SUSPEND
5	DISC ALLOCATION SUSPEND
6	OPERATOR SUSPEND

ALL PROGRAMS SCHEDULED BY THE R-T CLOCK ARE LINKED IN A SECONDARY LIST CALLED THE "TIME" LIST.

THE DORMANT LIST

DORMANT STATUS IS CODE Ø

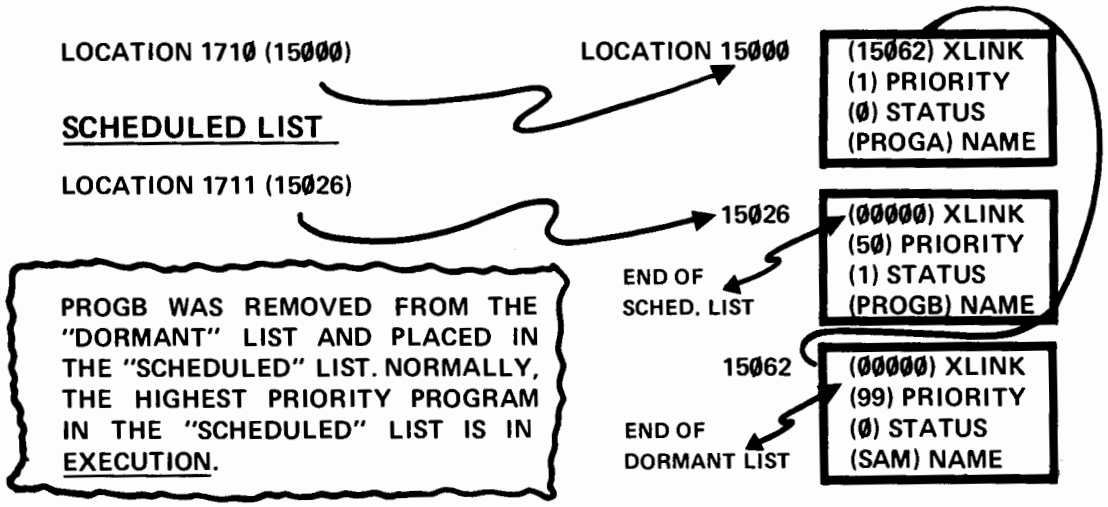
THE "HEAD" OF THE DORMANT LIST IS THE ADDRESS STORED IN LOCATION 1710_8. IF WE ASSUME ALL 3 PROGRAMS IN THE SIMPLIFIED SYSTEM ARE "DORMANT", THE LIST WOULD LOOK LIKE THIS:



THE SCHEDULED LIST

SCHEDULED STATUS IS CODE 1

THE HEAD OF THE SCHEDULED LIST IS THE ADDRESS STORED IN LOCATION 1711_8. ASSUME THE OPERATOR HAS TYPED "*ON, PROGB", THIS ACTION BY THE OPERATOR WILL PLACE PROGB IN THE SCHEDULED LIST; FOR EXAMPLE:



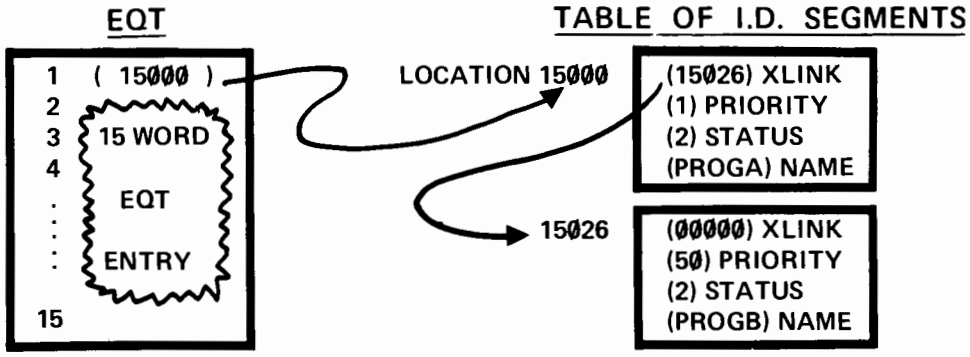
I/O SUSPENSION

I/O SUSPENSION STATUS IS CODE 2

EACH I/O DEVICE HAS ITS OWN SUSPENSION LIST. AN INPUT, OR NON-BUFFERED OUTPUT REQUEST CAUSES THE REQUESTING PROGRAM TO SUSPEND. WORD 1 OF THE "EQT" ENTRY FOR THE DEVICE POINTS TO THE "XLINK" WORD OF THE PROGRAM ID SEGMENT. IF XLINK = Ø, ONLY "THIS" PROGRAM IS IN THE QUEUE. IF XLINK ≠ Ø, IT POINTS TO THE NEXT PROGRAM IN THE LIST.

EXAMPLE

PROGA AND PROGB ARE IN I/O SUSPENSION. PROGRAMS ARE "LINKED" INTO A LIST ON THE BASIS OF THEIR PRIORITY.



AVAILABLE MEMORY, DISC, OPERATOR SUSPEND LISTS

AVAILABLE MEMORY STATUS IS CODE 4	AVAILABLE DISC STATUS IS CODE 5	OPERATOR SUSPEND STATUS IS CODE 6
AVAILABLE MEMORY	AVAILABLE DISC	OPERATOR
SUSPEND LIST	SUSPEND LIST	SUSPEND LIST
LOCATION 1714_8	LOCATION 1715_8	LOCATION 1716_8

ALL OF THE ABOVE LISTS USE A SIMILAR TECHNIQUE –

1 – IF THE LIST LOCATION EQUALS Ø, IT INDICATES A NULL LIST.

2 – IF NON-ZERO, IT IS THE ID SEGMENT ADDRESS OF THE HIGHEST PRIORITY PROGRAM IN THAT PARTICULAR LIST.

NOTE – PROGRAMS OF EQUAL PRIORITY ARE PLACED IN A LIST ON A "FIRST-IN", "FIRST-OUT" BASIS.

3 – IF MORE THAN ONE PROGRAM IS IN A LIST, THE "XLINK" WORD OF THE FIRST POINTS TO THE "XLINK" WORD OF THE SECOND, ETC.

4 – THE END-OF-LIST IS ALWAYS INDICATED BY AN "XLINK" VALUE OF ZERO.

THE TIME LIST

NORMALLY A PROGRAM IS IN ONE AND ONLY ONE LIST. THIS CONVENTION IS NOT TRUE FOR PROGRAMS SCHEDULED BY THE REAL TIME CLOCK. A PROGRAM MAY BE IN THE "SCHEDULED" OR "DORMANT" LIST, AND ALSO BE IN THE "TIME" LIST WAITING FOR THE CLOCK. THE "TIME" LIST HAS NO BASE PAGE ADDRESS. HOWEVER, IF A PROGRAM IS IN THE "TIME" LIST THE LETTER "T" IS PRINTED IN ADDITION TO THE NORMAL STATUS INFORMATION.

FOR EXAMPLE

```
*IT, SAM,  2, 300, 01, 10, 00
```

MULTIPLE RESOLUTION (SECONDS)
HOURS
MINUTES
SECONDS

```
*ON, SAM
*ST, SAM
XX 0 2 300 01 10 00 T
```

EXPLANATION

SET THE INITIAL TIME PARAMETERS { RUN PROGRAM EVERY 300 SECONDS (5 MINUTES) STARTING AT TIME 01:10:00 }

STATUS SHOWS THAT "SAM" IS IN THE DORMANT LIST (0), AND THE "T" INDICATES INCLUSION IN THE "TIME" LIST.

LABORATORY EXERCISE GUIDE

LESSON: LISTS & TABLES

Exercise: 4-1

A - OBJECTIVE

1. To involve the student in the operating system's lists and tables.

B - PROBLEM

1. On pages IV.L.2 and IV.L.3 of this exercise is a listing of the FORTRAN program "CDUMP", which permits the operator to obtain an octal dump of memory, including the operating system.
2. On pages IV.L.4 and IV.L.5 of this exercise is a sample dump using "CDUMP".
3. Using "CDUMP", obtain a dump of:
   a) The user/exec communications area (1650_8 - 1777_8).
   b) The system lists and tables area of the operating system. These addresses will be determined from the "SYS GEN" information in the "RTE FH SYSTEM BINDER".
4. Identify each list, table, and segment in the dump, showing links through each list "DORMT", "IO SUSPEND", "SKEDD", "MEMRY SUSPEND", "DISC SUSPEND", "OPER SUSPEND". Naturally, it would be advantageous, but not necessary, to have a program in each list while dumping memory.

C - PROCEDURE

1. To be determined by each student to the best of his ability.

```
PAGE  0001


FTN,L,B
      PROGRAM COUMP
      DIMENSION J(8)
      WRITE(1,100)
100   FORMAT("ENTER #FIRST,#LAST ←")
      READ(1,*)IFST,ILST
      IFST = IAND (IFST,77770B)
150   M = IFST
      IF (IAND(IFST,778)) 170,160,170
160   WRITE(6,161)
161   FORMAT(/)
170   DO 200 I = 1,8
         J(I)= IGET(M)
         M=M+1
200   CONTINUE
      WRITE (6,300)IFST,J
300   FORMAT(2X,K6,2X,8(2X,K6))
      IFST =IFST+8
      IF (IFST-ILST) 150,150,999
999   CONTINUE
      END
```

```
 PAGE 0002 #01


0001                          ASMB,R,L,B
0002   00000                         NAM IGET,7
0003                                 ENT IGET
0004   00000 000000   IGET    NOP
0005   00001 104200          DLD IGET,I
       00002 100000R
0006   00003 101100          SWP
0007   00004 160000          LDA 0,I
0008   00005 160000          LDA 0,I
0009   00006 124001          JMP 1,I
0010                          END
** NO ERRORS*
```

```
*ON,CDUMP
ENTER @FIRST,@LAST @1650,@1777
```

```
001650    021122  000013  021367  000015  021404  000022  023615  021426
001660    021160  021161  021162  021163  021164  021165  021166  021167
001670    021151  021152  021153  000006  000010  021177  000003  026470

001700    026665  026666  026672  027006  000000  000000  000000  000000
001710    021643  021607  000000  000000  000000  000000  000000  022231
001720    022231  022232  022233  022234  022235  022236  022237  022240
001730    022241  022242  022243  022244  000000  000000  000001  000000
001740    000265  000077  000471  000777  001000  023725  023725  000202
001750    024334  025400  026000  000202  027056  177670  000040  000132
001760    000060  002322  000003  002323  000241  000000  000000  004200
001770    000000  021173  021174  021175  021176  026000  000000  037677
```

LABORATORY EXERCISE GUIDE

LESSON: LISTS & TABLES

Exercise: 4-2

A – OBJECTIVE

1. To dynamically involve the student in the operating systems lists and tables.

B – PROBLEM

1. Exercise 4-1 provides us with a static analysis of the operating system and user programs. It would be very beneficial to the system manager to dynamically determine the status of all non-dormant programs.
2. Page IV.L.7 is a printout of the dynamic status. Pages IV.L.8 thru IV.L.11 is an ALGOL listing of the "STATS" program that generated the dynamic status, page IV.L.7.
3. This program "STATS" requires about 1K of memory in ALGOL which makes it unreasonable for a core-resident program, which it should be so as to give dynamic status at any time the system manager desires.
4. Code this program in Assembly language, background core resident, priority 1 in the most efficient manner possible.

C – PROCEDURE

1. To be determined by each student according to his ability.
2. Keep in mind that:
   a) minimum core requirements
   b) minimum CPU time for execution

   are the programming guidelines in order of priority.

```
*LU,7,4
TIME  00:08:36:670
*****************************************************************
*NAMES,T,PR*DORMT*SCHED* I/O ,EQT #*MEMRY*DISC *OPER * NEXT TIME   *
*****************************************************************
DHSAD,1,48 ************  02 ,  09 *
AHSAD,2,50 *******  01 *
LAB2 ,2,74 *  00 ***********************************00:09:09:360**
##ATS,3,75 *******  01 ***********************************00:09:18:620**
TIME  00:09:24:320

*
```

```
HPAL,L,B,"STATS",0,4,25
    BEGIN
  INTEGER LU := 7 ;
  INTEGER OUT := 2 ;
  INTEGER TIMECALL := 11 ;
  INTEGER ASTERISKS := "**" ;
  INTEGER COLON := ":" ;
  INTEGER ZERO := "00" ;
  INTEGER SPACES := "  " ;
  INTEGER COMMA := ",," ;
  INTEGER EQTA := @1650 ;
  INTEGER NUMBEROFEQTS := @1651 ;
  INTEGER KEYWD := @1657 ;
  INTEGER EQTADDRESS ;
  INTEGER I ;
  INTEGER CHARACTERS ;
  INTEGER STATUS ;
  INTEGER INDEX ;
  INTEGER TLIST ;
    LABEL START1,START2,
          IOSUSPEND,
          TIMELIST,
          FINIS ;
  INTEGER
    ARRAY TIMEARRAY[1:5] ;
  INTEGER
    ARRAY TIME[18:21] := 100,60,60,24 ;
  INTEGER
    ARRAY DATA[-1:35] := @16436,@17036,@17036 ;
  INTEGER
    ARRAY HEADER[0:35] := @17036,@17036,"*N","AM","ES",",T",",P","R*",
                          "DO","RM","T*","SC","HE","D*"," I","/O","  ,",
                          "EQ","T ","**","ME","MR","Y*","DI","SC"," *",
                          "OP","ER"," *"," N","EX","T ","TI","ME","  ",
                          " *" ;
PROCEDURE EXEC2(NO1,NO2) ;
  INTEGER NO1,NO2 ;
    CODE ;
PROCEDURE EXEC4(NO1,NO2,NO3,NO4) ;
  INTEGER NO1,NO2,NO3,NO4 ;
    CODE ;
  INTEGER
PROCEDURE IGET(ADDRESS) ;
  INTEGER ADDRESS ;
    CODE ;
  INTEGER
PROCEDURE CONVERT(NUMBER) ;
    VALUE NUMBER ;
  INTEGER NUMBER ;
    BEGIN
  CONVERT := ZERO ;
  CONVERT := CONVERT * ROTATE(NUMBER\10)
                     * NUMBER MOD 10 ;
      END CONVERT ;
  INTEGER
PROCEDURE KEYWORD(NUMBER) ;
    VALUE NUMBER ;
  INTEGER NUMBER ;
    BEGIN
  INTEGER ADDRESS ;
```

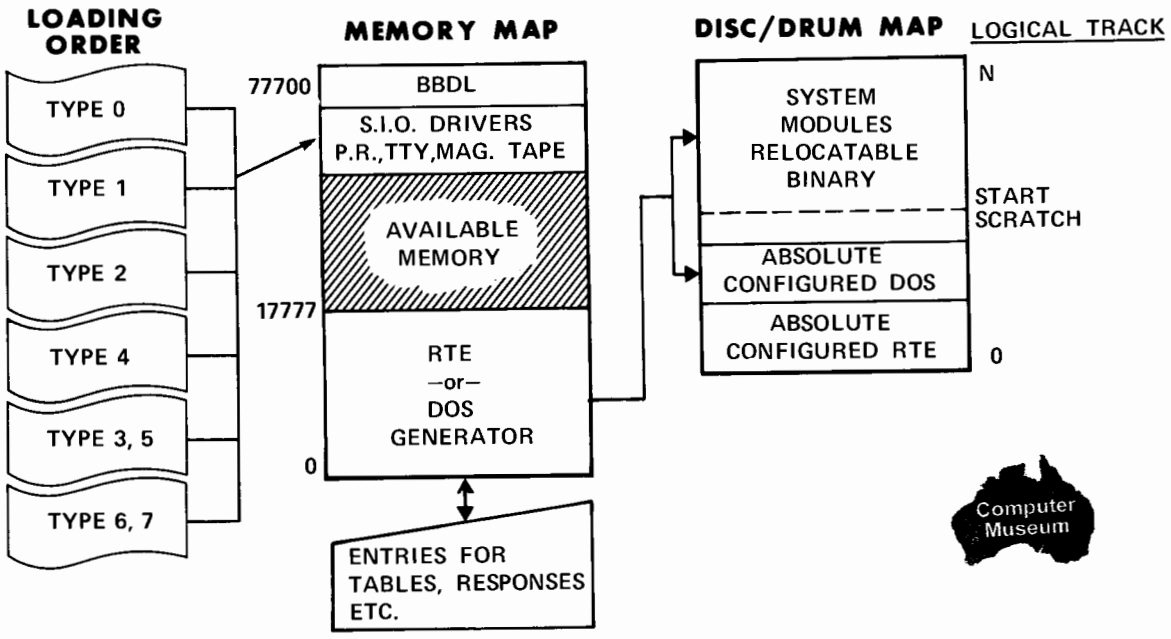
```
          ADDRESS := IGET(KEYWD)+NUMBER ;
  KEYWORD := IGET(ADDRESS) ;
      END KEYWORD ;
  INTEGER
PROCEDURE ID(ELEMENT) ;
    VALUE ELEMENT ;
  INTEGER ELEMENT ;
    BEGIN
  INTEGER ADDRESS ;
          ADDRESS := KEYWORD(INDEX)+ELEMENT ;
       ID := IGET(ADDRESS) ;
      END ID ;
  INTEGER
PROCEDURE ASTERISK ;
    BEGIN FOR I := 2 STEP 1 UNTIL 35
          DO DATA[I] := ASTERISKS ;
      END ASTERISK ;
PROCEDURE DISPLAY(INFORMATION) ;
  INTEGER INFORMATION ;
    BEGIN EXEC4(OUT,LU,INFORMATION,CHARACTERS) ;
      END DISPLAY ;
  BOOLEAN
PROCEDURE LISTSEARCH(LISTADDRESS,IDSEGMENT) ;
    VALUE LISTADDRESS,IDSEGMENT ;
  INTEGER LISTADDRESS,IDSEGMENT ;
    BEGIN
    LABEL S0 ;
S0:         IF LISTADDRESS=IDSEGMENT
            THEN LISTSEARCH := TRUE
            ELSE IF LISTADDRESS#0
                 THEN BEGIN
                      LISTADDRESS := IGET (LISTADDRESS) ;
                      GO TO S0 ;
                      END
                 ELSE LISTSEARCH := FALSE ;
      END LISTSEARCH ;
PROCEDURE CONVERTTIME(WHERE) ;
    VALUE WHERE ;
  INTEGER WHERE ;
    BEGIN FOR I := 0 STEP 1 UNTIL 1
          DO BEGIN
             DATA[WHERE+3*I] := CONVERT(TIMEARRAY[4-2*I]) ;
             DATA[WHERE+2+3*I] := CONVERT(TIMEARRAY[3-2*I]) ;
             DATA[WHERE+1+3*I] := COLON AND #177400
                                  OR ROTATE(DATA[WHERE+2+3*I]
                                  AND #177400 ) ;
             DATA[WHERE+2+3*I] := ROTATE(DATA[WHERE+2+3*I]
                                  AND #377) OR COLON AND #377 ;
             END ;
          DATA[WHERE+5] := DATA[WHERE+5] AND #177400
                           OR ZERO AND #377 ;
      END CONVERTTIME ;
PROCEDURE TIMEOFDAY ;
    BEGIN EXEC2(TIMECALL,TIMEARRAY[1]) ;
          DATA[2] := HEADER[32] ;
          DATA[3] := HEADER[33] ;
          DATA[4] := SPACES ;
          CONVERTTIME(5) ;
      END TIMEOFDAY ;
COMMENT   DISPLAY
```

```
2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
*************************************************************************
*NAMES,T,PR*DORMT*SCHED* I/O ,EQT #*MEMRY*DISC *OPER * NEXT TIME  **
*************************************************************************
NAME ,T,PR**  00 *  01 *  02 *  EQ *  04 *  05 *  06 *HH:MM:SS:MM0**
          HEADER ;
START1:   TIMEOFDAY ;
          CHARACTERS := -24 ;
          DISPLAY(DATA[-1]) ;
          ASTERISK ;
          CHARACTERS := -72 ;
          DISPLAY(DATA[0]) ;
          DISPLAY(HEADER[0]) ;
          DISPLAY(DATA[0]) ;
          INDEX := 0 ;
          WHILE KEYWORD(INDEX) # 0
          DO BEGIN
START2:      STATUS := ID(15) AND @7 ;
             TLIST := ID(17) AND @10000 ;
             IF (STATUS OR TLIST) # 0
             THEN BEGIN
                  ASTERISK ;
                  DATA[2] := ID(12) ;
                  DATA[3] := ID(13) ;
                  DATA[4] := ID(14) AND @177400
                             OR COMMA AND @377 ;
                  DATA[5] := ROTATE(CONVERT(ID(14) AND @7) AND @377)
                             OR COMMA AND @377 ;
                  DATA[6] := CONVERT(ID(6) AND @177) ;
                  DATA[7] := SPACES AND @177400
                             OR ASTERISKS AND @377 ;
                  I := 3*STATUS ;
                  DATA[I+8] := SPACES ;
                  DATA[I+9] := CONVERT(STATUS) ;
                  DATA[I+10] :=SPACES AND @177400
                             OR ASTERISKS AND @377 ;
                  CHARACTERS := -2*(I+11) ;
IOSUSPEND:        IF STATUS=2
                  THEN BEGIN
                       FOR EQTADDRESS := IGET(EQTA) STEP 15 UNTIL
                                         IGET(EQTA)+15*
                                        (IGET(NUMBEROFEQTS)-1)
                       DO IF LISTSEARCH(EQTADDRESS,
                                        KEYWORD(INDEX))=TRUE
                          THEN BEGIN
                               DATA[16] := SPACES AND @177400
                                           OR COMMA AND @377 ;
                               DATA[17]:= SPACES ;
                               DATA[18] := CONVERT((EQTADDRESS-
                                           IGET(EQTA))\15+1);
                               DATA[19] := SPACES AND @177400
                                           OR ASTERISKS AND @377 ;
                               CHARACTERS := -40 ;
                               GO TO TIMELIST ;
                               END ;
                       GO TO START2 ;
                       END ;
TIMELIST:         IF TLIST#0
                  THEN BEGIN
                       FOR I := 1 STEP 1 UNTIL 4
```

```
                        DO TIMEARRAY[I] := TIME[17+I]+ID(17+I) ;
                        CONVERTTIME(29) ;
                        CHARACTERS := -72
                        END ;
                    DISPLAY(DATA[0]) ;
                    END ;
                INDEX := INDEX + 1 ;
                END ;
FINIS:      TIMEOFDAY ;
            CHARACTERS := -22 ;
            DISPLAY(DATA[0]) ;
        END ;
```

SYSTEM GENERATION V

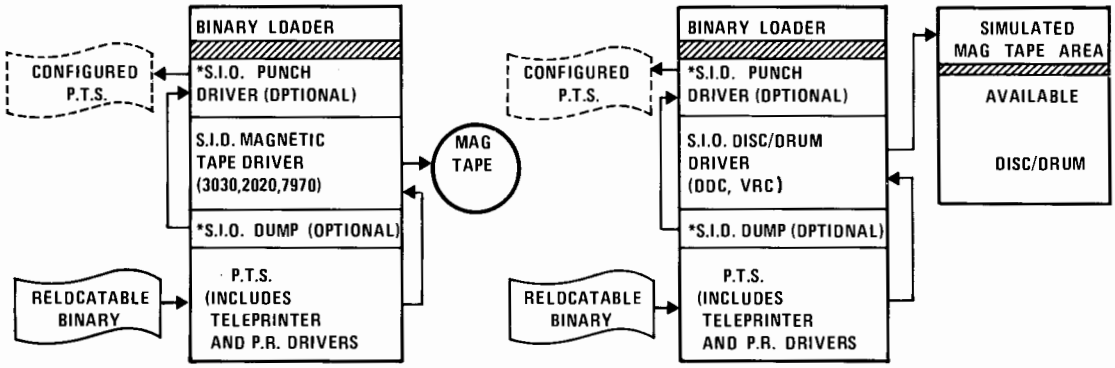
SYSTEM GENERATION (TAPE)



- THE SYSTEM GENERATOR IS LOADED INTO MEMORY USING THE BBDL.
- INITIALIZATION PHASE – ESTABLISHES DISC SIZE, TYPE, SYSTEM HARDWARE INFO.
- PROGRAM INPUT PHASE – SYSTEM AND USER PROGRAMS ARE COPIED ON THE DISC/DRUM.
- PARAMETER INPUT PHASE – PROGRAM PRIORITIES AND TYPE CODES MAY BE CHANGED.
- DISC LOADING PHASE – ALL TABLES ARE CONSTRUCTED AND THE ABSOLUTE SYSTEM IS CREATED ON THE SYSTEM DISC/DRUM.

PREPARE TAPE SYSTEM

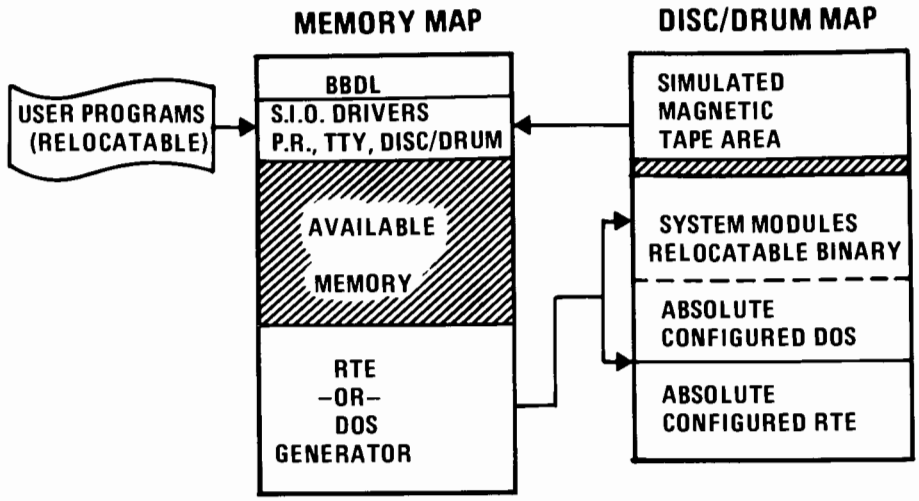
THE PTS PROGRAM CAN BE USED TO CREATE A MAGNETIC TAPE OR DISC FILE CONTAINING ALL THE RELOCATABLE MODULES OF A REAL-TIME OR DISC OPERATING SYSTEM.



*A CONFIGURED P.T.S. TAPE CAN BE CREATED USING THE S.I.O. DUMP. DETAILED OPERATING INSTRUCTIONS ARE OUTLINED IN THE P.T.S. MANUAL. HP P/N 2116-91751.

*THE S.I.O. DISC DRIVER ALLOWS A SPECIFIED NUMBER OF DISC TRACKS TO SIMULATE A CONTINUOUS MAGNETIC TAPE.

SYSTEM GENERATION (DISC/DRUM)



TO AVOID CONFLICTS WITH THE SYSTEM GENERATOR THE NUMBER OF DISC TRACKS WITHIN THE SIMULATED MAG. TAPE AREA MUST BE SUBTRACTED FROM THE TOTAL NUMBER OF TRACKS AVAILABLE. THE MAG. TAPE AREA CONSISTS OF A CONTIGUOUS GROUP OF HIGH ORDER DISC TRACKS.

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

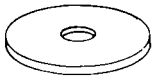
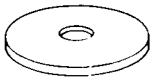
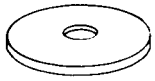
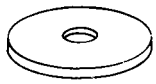
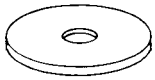
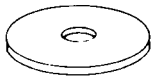
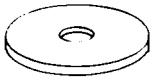
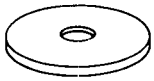
1. FIXED HEAD AUXILIARY DISC
2. FIXED HEAD PERIPHERAL DISCS
3. MOVING HEAD PERIPHERAL DISCS
4. MOVING HEAD AUXILIARY DISC

MOVING HEAD SYSTEM DISC

1. MOVING HEAD AUXILIARY DISC
2. MOVING HEAD PERIPHERAL DISCS
3. FIXED HEAD PERIPHERAL DISCS

POSSIBLE DISC COMBINATIONS RTE SYSTEMS

MOVING HEAD DISC

MOVING HEAD DRIVE	MOVING HEAD DRIVE	MOVING HEAD DRIVE	MOVING HEAD DRIVE
SUBCHANNEL 0	SUBCHANNEL 2	SUBCHANNEL 4	SUBCHANNEL 6
FIXED 			
NO. OF TRACKS AVAILABLE, NO0 _____	NO. OF TRACKS AVAILABLE, NO2 _____	NO. OF TRACKS AVAILABLE, NO4 _____	NO. OF TRACKS AVAILABLE, NO6 _____
FIRST TRACK, FT0 _____	FIRST TRACK, FT2 _____	FIRST TRACK, FT4 _____	FIRST TRACK, FT6 _____
SUBCHANNEL 1	SUBCHANNEL 3	SUBCHANNEL 5	SUBCHANNEL 7
REMOVABLE 			
NO. OF TRACKS AVAILABLE, NO1 _____	NO. OF TRACKS AVAILABLE, NO3 _____	NO. OF TRACKS AVAILABLE, NO5 _____	NO. OF TRACKS AVAILABLE, NO7 _____
FIRST TRACK, FT1 _____	FIRST TRACK, FT3 _____	FIRST TRACK, FT5 _____	FIRST TRACK, FT7 _____

SYSTEM SUBCHANNEL NUMBER _____ SCRATCH SUBCHANNEL NUMBER _____

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

MOVING HEAD AUXILIARY DISC TRACK ALLOCATION TABLE

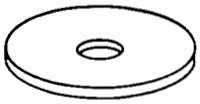
(USER PREPARED)

```
ASMB, R, B, L
        NAM     $TB31,0
        ENT     $TB31
$TB31   DEC     -X
        DEC     FT0,FT1,FT2,FT3,FT4,FT5,FT6,FT7
        DEC     NO0,NO1,NO2,NO3,NO4,NO5,NO6,NO7
        END
```

FIXED HEAD SYSTEM DISC

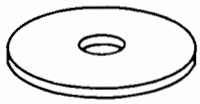
DRUM	DISC	NO. OF LOGICAL TRACKS	SECTORS PER TRACK
–	2766	32	128
2773	–	48	
2773-003	2766-002	64	
2774	2766-003	96	
2774-003	2766-004	128	
–	2770	32	90
–	2770-01	64	
–	2771	64	
–	2771-01	128	

SYSTEM (LU2)



NO. OF TRACKS AVAILABLE _____
START SCRATCH _____
NO. OF PROTECTED TRACKS _____
SECTORS/TRACK _____

AUXILIARY (LU3)



NO. OF TRACKS AVAILABLE _____
SECTORS/TRACK _____

A. ASSIGN ALL DEVICES THAT REQUIRE PRIVILEGED INTERRUPT IN ORDER OF DECREASING SPEED.

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 ALL DEVICES THAT *DO NOT* USE DMA IN ORDER OF DECREASING SPEED.

E. ASSIGN ALL DEVICES THAT *DO* USE DMA IN ORDER OF DECREASING SPEED.

F. 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.

SELECT CODE ASSIGNMENTS

INPUT/OUTPUT CONFIGURATION WORKSHEET

RTGEN NUMBER _____ DATE _____ PREPARED BY _____

STD. LOGICAL UNIT NOS.	I/O INTERFACE CARD NAME
	SC / SUB (10)
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)

INITILIZATION PHASE — MH

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
(0?)
(1?)
(2?)
(3?)
(4?)
(5?)
(6?)
(7?)

PUNCH BOOT?
YES
.
.
.
NO

INITIALIZATION PHASE - FH

FH. DISC CHNL?	TBG CHNL?
SYS DISC SIZE?	PRIV. INT. CARD ADDR?
START SCRATCH?	SWAPPING?
NO. PROTECTED?	LWA MEM?
# SECTORS/TRACK?	PRGM INPT?
AUX DISC SIZE?	LIBR INPT?
# SECTORS/TRACK?	PRAM INPT?

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	T	T	T	T	0	N	N

2	1	0
0	N	N

= 000, PROGRAM INPUT DEVICE

001, { LIST UNDEFINED EXTERNALS (1st "RUN") OR TERMINATE (2nd "RUN")

010, LIBRARY INPUT DEVICE

IF (| 2 | 1 | 0 | | 0 | N | N | = 000) AND (NO TYPE CODE IN NAM RECORD)

THEN | 6 | 5 | 4 | 3 | | T | T | T | T | INDICATES PROGRAM TYPE

SWITCH REGISTER CONTROL DURING PROGRAM INPUT PHASE

NOTE: LOAD PROGRAMS IN THE FOLLOWING ORDER

TYPE 0 – EXEC
SCHED
RTIOC
DRIVERS
USER WRITTEN

TYPE 1 – USER WRITTEN

TYPE 2 – USER WRITTEN

TYPE 4 – USER WRITTEN

TYPE 3,5 – USER WRITTEN
ASMB
FTN
FTN4
ALGOL
LOADR
EDIT

TYPE 6,7 – USER WRITTEN
PLOTTER LIBRARY
FTN FORMATTER OR FTN4 LIBRARY
EAU LIBRARY

PROGRAM INPUT PHASE

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

_____,_____,_____,_____,_____,_____,_____,_____,_____

_____,_____,_____,_____,_____,_____,_____,_____,_____

_____,_____,_____,_____,_____,_____,_____,_____,_____

_____,_____,_____,_____,_____,_____,_____,_____,_____

_____,_____,_____,_____,_____,_____,_____,_____,_____

_____,_____,_____,_____,_____,_____,_____,_____,_____

_____,_____,_____,_____,_____,_____,_____,_____,_____

_____,_____,_____,_____,_____,_____,_____,_____,_____

_____,_____,_____,_____,_____,_____,_____,_____,_____

/E

OF BLANK ID SEGMENTS?

_____

FWA BP LINKAGE

_____

PARAMETER INPUT PHASE

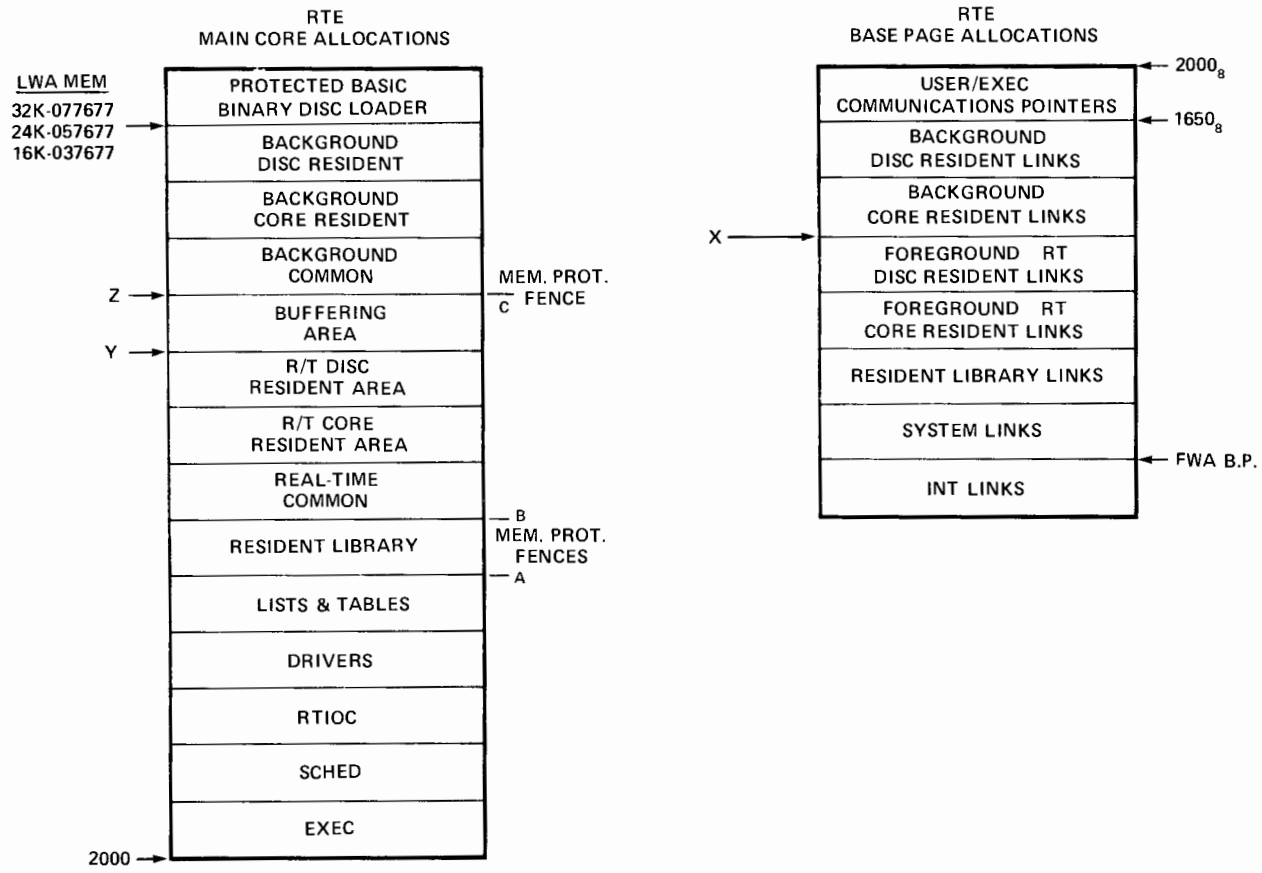
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

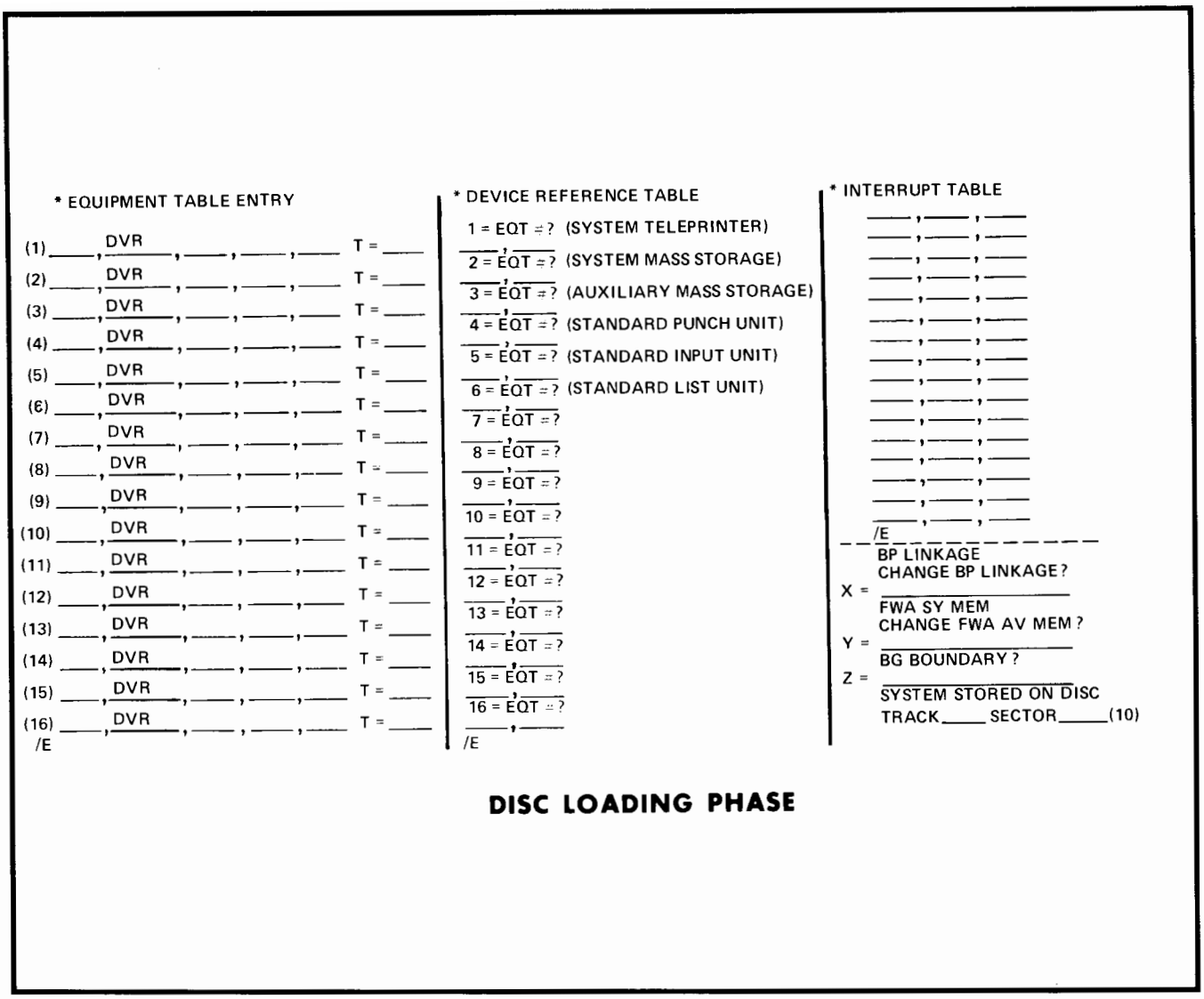
IF (P = 1)

THEN (LIST ALL ENTRY POINTS TO MODULE)

SWITCH REGISTER CONTROL DURING DISC LOADING PHASE

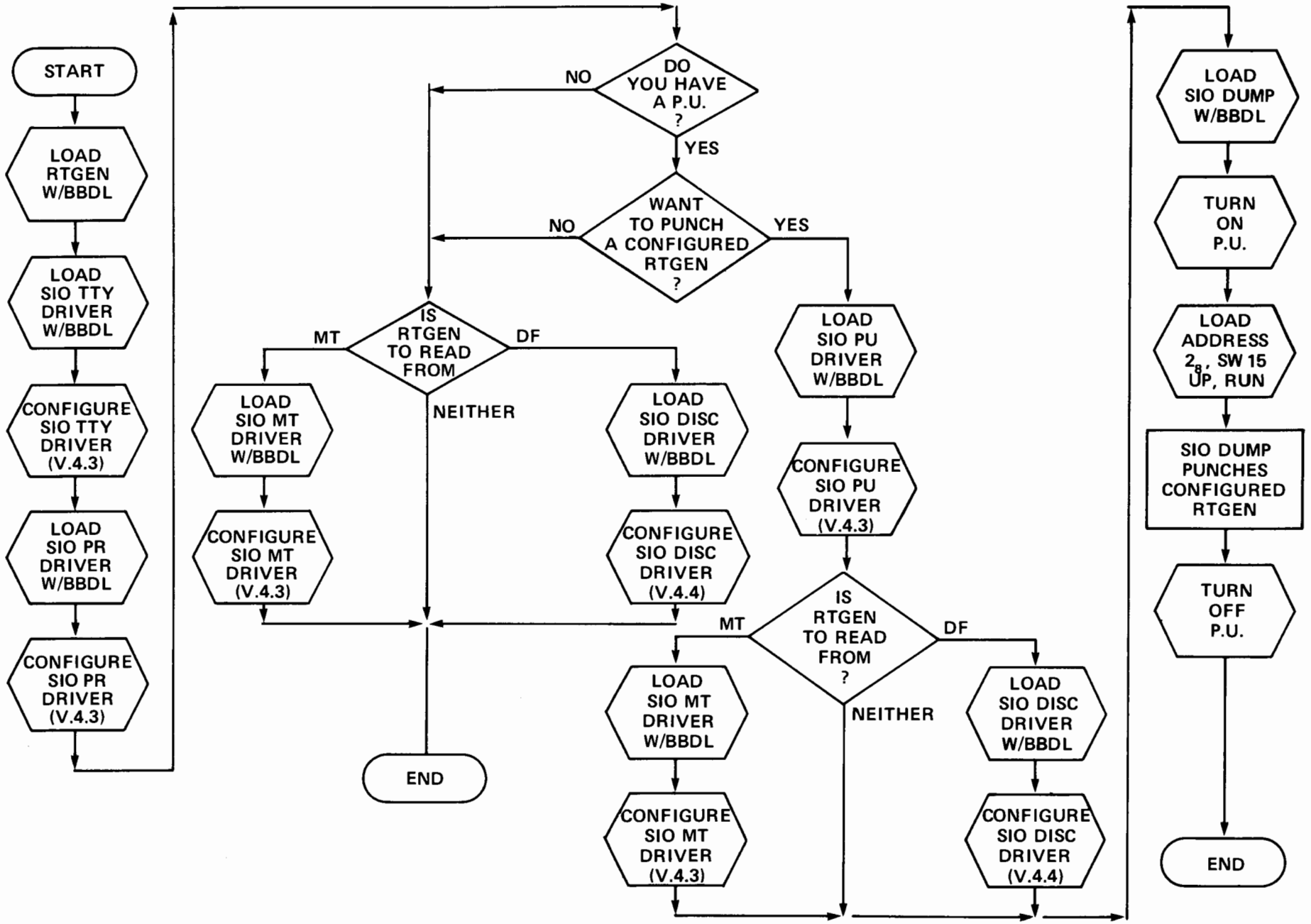
CONFIGURED SYSTEM MEMORY MAP



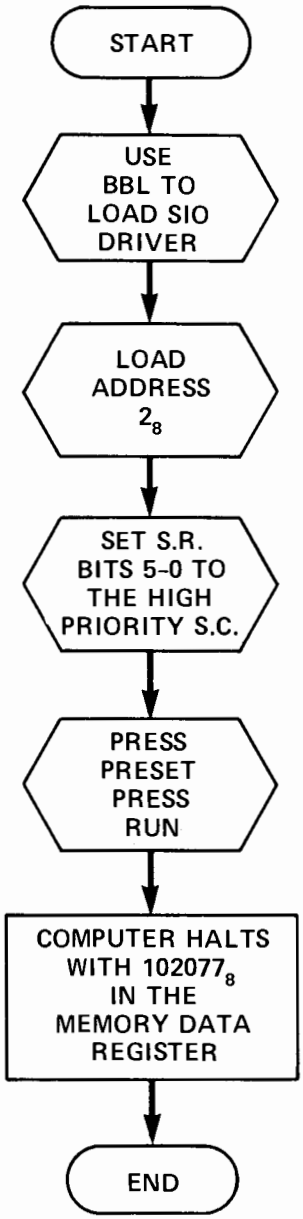


DISC LOADING PHASE

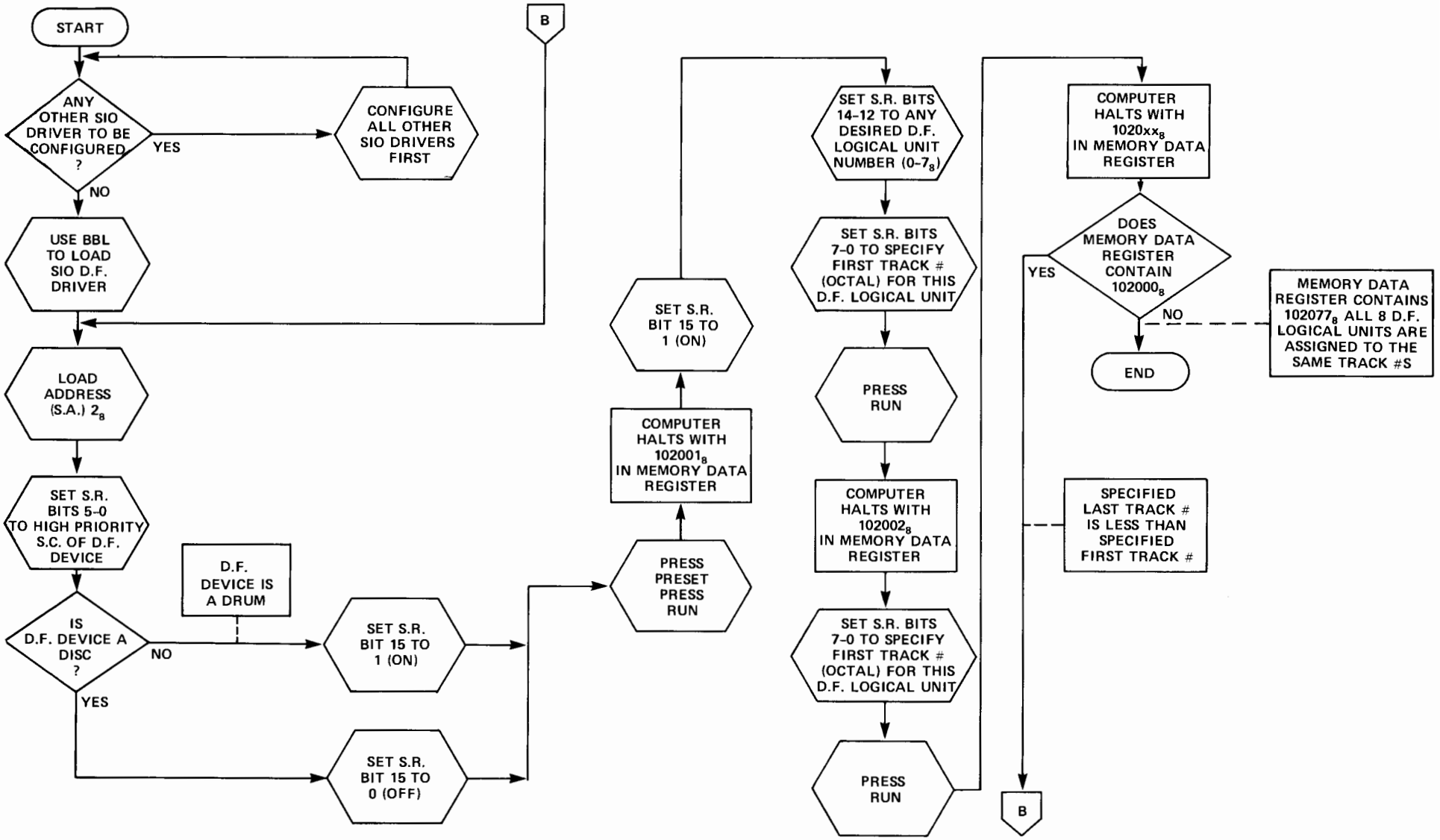
RTGEN CONFIGURATION



PTS CONFIGURATION



SIO DRIVER CONFIGURATION



SIO DISC DRIVER CONFIGURATION DIAGRAM

LABORATORY EXERCISE GUIDE

LESSON: THE REAL-TIME SYSTEM GENERATOR

Exercise: 5-1

A - OBJECTIVE

1. To provide a practical exercise that will enable the student to configure a Real-Time System.

B - PROBLEM

1. To configure a Real-Time System.

C - PROCEDURES

1. Using the configuration diagram and the information obtained during the lecture, prepare a parameter tape. The parameter tape will contain the responses normally typed in at generation time. In order to save time in the lab each student must prepare a parameter tape using an off-line Teleprinter.
2. Have the parameter tape listing checked by the instructor. If the parameter tape listing checks out, the instructor will assign a system to your group.
3. Use the procedures in Section 7 of the Manual to configure a system. All the Real-Time Software will be on a mass storage device (Disc or Tape) for this exercise. The parameter tape is read from the Teleprinter reader.

D - RESULTS

1. Your efforts during this exercise should yield a working Real-Time system. The system may be given some simple test like a FORTRAN compilation or Edit to verify its operation.

World-Wide Hewlett-Packard Sales & Service

Call your HP System Sales Engineer at any of these convenient locations:

UNITED STATES

ALABAMA
Huntsville
Tel: (205) 881-4591

ARIZONA
Phoenix
Tel: (602) 252-5061
Tucson
Tel: (602) 298-2313

CALIFORNIA
Fullerton
Tel: (714) 870-1000
North Hollywood
Tel: (213) 877-1282
Palo Alto
Tel: (415) 327-6500
Sacramento
Tel: (916) 482-1463
San Diego
Tel: (714) 279-3200

COLORADO
Englewood
Tel: (303) 771-3455

CONNECTICUT
East Hartford
Tel: (203) 289-9394
Norwalk
Tel: (203) 853-1251

FLORIDA
Ft. Lauderdale
Tel: (305) 731-2020
Orlando
Tel: (305) 841-3970

GEORGIA
Atlanta
Tel: (404) 436-6181

ILLINOIS
Skokie
Tel: (312) 677-0400

INDIANA
Indianapolis
Tel: (317) 546-4891

LOUISIANA
Kenner
Tel: (504) 721-6201

MARYLAND
Baltimore
Tel: (301) 944-5400
Rockville
Tel: (301) 948-6370

MASSACHUSETTS
Lexington
Tel: (617) 861-8960

MICHIGAN
Southfield
Tel: (313) 353-9100

MINNESOTA
St. Paul
Tel: (612) 645-9461

MISSOURI
Kansas City
Tel: (816) 763-8000
St. Louis
Tel: (314) 962-5000

NEW JERSEY
Paramus
Tel: (201) 265-5000
Cherry Hill
Tel: (609) 667-4000

NEW MEXICO
Albuquerque
Tel: (505) 265-3713
Las Cruces
Tel: (505) 526-2485

NEW YORK
Albany
Tel: (518) 869-8462
Endicott
Tel: (607) 754-0050
Poughkeepsie
Tel: (914) 454-7330
Rochester
Tel: (716) 473-9500
Syracuse
Tel: (315) 454-2486
Woodbury
Tel: (516) 921-0300

NORTH CAROLINA
High Point
Tel: (919) 885-8101

OHIO
Cleveland
Tel: (216) 835-0300
Columbus
Tel: (614) 846-1300
Dayton
Tel: (513) 298-0351

OKLAHOMA
Oklahoma City
Tel: (405) 848-2801

OREGON
Portland
Tel: (503) 292-9171

PENNSYLVANIA
Monroeville
Tel: (412) 271-0724
King of Prussia
Tel: (215) 265-7000

RHODE ISLAND
East Providence
Tel: (401) 434-5535

TEXAS
Richardson
Tel: (214) 231-6101
Houston
Tel: (713) 781-6000
San Antonio
Tel: (512) 434-4171

UTAH
Salt Lake City
Tel: (801) 487-0715

VERMONT
South Burlington
Tel: (802) 658-4455

VIRGINIA
Richmond
Tel: (703) 285-3431

WASHINGTON
Bellevue
Tel: (206) 454-3971

WEST VIRGINIA
Charleston
Tel: (304) 768-1232

CANADA

ALBERTA
Edmonton
Tel: (403) 482-5561

BRITISH COLUMBIA
North Burnaby
Tel: (604) 433-8213

MANITOBA
Winnipeg
Tel: (204) 786-7581

NOVA SCOTIA
Halifax
Tel: (902) 455-0511

ONTARIO
Ottawa
Tel: (613) 722-4223
Rexdale
Tel: (416) 677-9611

QUEBEC
Pointe Claire
Tel: (514) 697-4232

EUROPE

AUSTRIA
Vienna

BELGIUM
Brussels

DENMARK
Birkeroed

FINLAND
Helsinki

FRANCE
Orsay
Lyon
Blagnac

GERMANY
Berlin W
Böblingen
Dusseldorf
Frankfurt
Hamburg
Munchen

ITALY
Milan
Rome

NETHERLANDS
Amsterdam

NORWAY
Haslum

SWEDEN
Mölndal
Solna

SWITZERLAND
Zurich
Meyrin-Geneva

UNITED KINGDOM
Altrincham, Cheshire
Slough, Bucks

FOR EUROPEAN AREAS NOT LISTED, CONTACT:

Hewlett-Packard S.A.
Rue du Bois-du-Lan 7,
1217 Meyrin-Geneva
Tel: (022) 41 54 00

CENTRAL AND SOUTH AMERICA

ARGENTINA
Buenos Aires

BRAZIL
Sao Paulo
Rio de Janeiro

MEXICO
Mexico City

VENEZUELA
Caracas

AFRICA, ASIA, AUSTRALIA

AUSTRALIA
Melbourne
Sydney
Adelaide
Perth, W.A.

JAPAN
Osaka
Nagoya
Tokyo

NEW ZEALAND
Wellington

SOUTH AFRICA
Cape Town
Johannesburg

FOR OTHER AREAS NOT LISTED CONTACT:

Hewlett-Packard Export Marketing
3200 Hillview Ave.
Palo Alto, California 94304, U.S.A.
Telex: 034-8461
Cable: HEWPACK Palo Alto

Printed in U.S.A. 1/72