In addition to the changes needed to correct the HP 2000A TSB System Version E, the following new features have been added:

1. All lines printed on the ASR-35 by the commands REPORT and DIRECTORY will be terminated by XOFF CR LF instead of CR LF.

2. The HP7970 Magnetic Tape Unit may be substituted for the HP 3030 Magnetic Tape Unit. Users of HP 7970 should type:

   MAGTAPE - select code *

   when entering MAG TAPE commands. (The * must appear if SC ≠0)

3. If a HP 7970 is being used, the status command will print an * after the Magtape select code.

Reasons for HP 2000A TSB changes were to correct the following:

1. Using = signs inside parameters to functions can cause fatal errors when the program is run. e.g. A = INT (A=0)

2. Redimensioning matrices to total size > 32767 can cause fatal errors. e.g. MATA = ZER (1000, 1000)

3. In the CAT command, random shipping within the directory can occur at the end of each line. This is non-fatal.

4. If a user hangs up his phone or exhausts his log on time simultaneously with typing CR or break, he may crash the system.

5. If a user in tape mode hangs up, the next user to come on that port is still in tape mode.

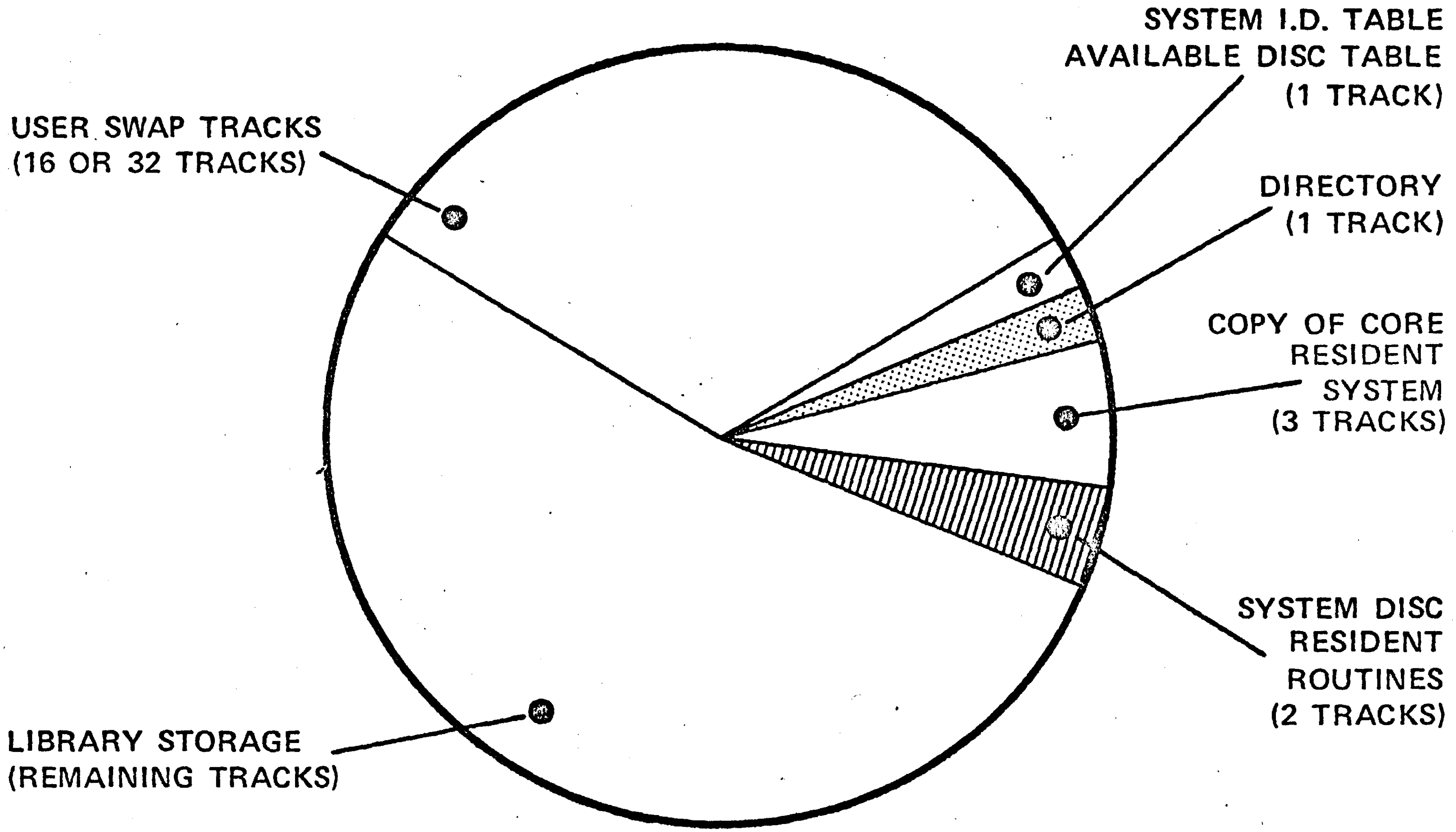
6. Disc errors occurring in the loader are printed incorrectly.

7. At the end of a magtape SLEEP, a verify procedure has been added. The following message is printed:

   VERIFY?

   A response of NO will result in the system printing DONE and halting. A response of YES will cause the tape to be reread and checked for validity. If it is valid, the system will print DONE and halt. Otherwise, it will print TAPE BAD OR TOO SHORT, and halt. The dump can be restarted by pressing RUN.

TYPICAL SYSTEM DISC USAGE



LOGICAL DISC 0

SLEEP COMMAND

▶ PURPOSE: PROVIDE FOR SYSTEMATIC SHUTDOWN OF SYSTEM.

▶ FORMAT: SLEEP – CHARACTER STRING (MESSAGE)

▶ EFFECT:

1. MESSAGE SENT TO ALL ACTIVE USERS
2. ALL USERS ARE DISCONNECTED
3. THE CURRENT SYSTEM IS DUMPED TO DISC
4. LIBRARY TRACKS ARE PACKED FOR EFFICIENCY PURPOSES.
5. IF MAGNETIC TAPE IS PRESENT, THE SYSTEM IS DUMPED TO MAG TAPE
6. COMPUTER HALTS

NOTE: SLEEPING A SYSTEM TYPICALLY REQUIRES 5 MINUTES.

```
STA

IDT Ø/21/ØØØ 116Ø  ADT Ø/21/Ø19 Ø3Ø4

DIREC
Ø/22/ØØØ 1424  Ø/23/ØØØ 2272  Ø/24/ØØØ 1784  Ø/25/ØØØ 1568

SYSTEM Ø/ØØ/ØØØ Ø/Ø1/ØØØ Ø/Ø2/ØØØ Ø/Ø3/ØØØ Ø/Ø4/ØØØ

USERS
Ø/Ø5/Ø14 Ø/Ø6/ØØ2 Ø/Ø7/ØØØ Ø/Ø8/ØØØ Ø/Ø9/ØØØ Ø/1Ø/Ø21 Ø/11/ØØ5 Ø/12/ØØ
Ø/13/ØØØ Ø/14/Ø32 Ø/15/ØØØ Ø/16/ØØ5 Ø/17/ØØØ Ø/18/ØØØ Ø/19/ØØØ Ø/2Ø/ØØ

MAG= 17  PHONES= 16  DISC= 14-128 14-128 14-128 ØØ-ØØØ

 TRACKS
Ø   ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ
1   ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ
2   ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ
3   1111111111111111111111111111111111111111111111111111111111111111
```

SAMPLE STATUS

OVERVIEW OF INTERNAL TIME SHARING OPERATION

I Core Map Elements

A. Scheduler (TSB heart) - Schedules/ initiates/ suspends/ terminates tasks.

B. BASIC interpreter - syntax checking; RUN,LIST,PUNCH

1. Re-enterable processor - i.e. one copy shared among all users

C. Swap area

1. User swap area(1240B → (1240B + 5440), each user swapped in/out between (the one) in-core swap area and that user's particular disc swap track.

2. System library program work space (1244B → (1244B + 15440)

D. System Library Program, (SLIP) Overlay Area System routine execute area (user & operator commands)

1. Programs only read in -- i.e. no need to swap out.

2. SLIP's run to completion, or self suspension (during output).

E. Drivers

1. MPXR - Multiplexes input/output from/to the 16 user teletypes.

2. Disc Driver - uses DMA 3,7 (2,6 not used)

3. Console (TTY35) - handled separately from users. Requires tty board.

4. Power Fail - System should be fully restored upon power failure.

F. TTY tables - in-core tables containing user buffer information, and scheduling status.

1. One for each user.

II. Disc Map Elements

A. Swap tracks - 1/user

B. Core-resident copy of system

1. Mainly for re-loading

2. Portions also used for SLEEP

C. System Library Programs (disc-resident)

D. DIRECTORY track (s) - 1/64 tracks (i.e. each additional 64 tracks require another DIRECTORY track).

E. IDT/ADT (I.D. table, available disc table) track.

F. Remainder available for user files/programs.

III. Interaction
- A. Swap tracks to/from swap area
- B. SLIP's to SLIP overlay area
- C. DIRECTORY, IDT, ADT to/from SLIP work-area (user swap area) when being examined/modified.

[Scheduler function (objective): To schedule, initiate, suspend, or terminate tasks]

IV. Tasks
- A. Def: A body of programming work to be executed by the CPU. e.g.:
  - 1. Syntax processing for user BASIC statement
  - 2. User commands (RUN, CAT, REN, ...)
  - 3. Operator commands (ROS, DIS, SLE,...)
- B. Each port (not user) associated with a task; plus operator console.
  - 1. Therefore, maximum of 16+1 ready tasks can request service, simultaneously.
- C. Types of tasks
  - 1. Core-resident -
    - a. Executed immediately: SCR, TAP, KEY
    - b. Scheduled: Syntax processing, RUN, LIST, PUNCH.
    - c. Use swap area for BASIC programs(swapped in/out).
  - 2. Disc-resident - all other (user and operator) commands.
    - a. Executed in SLIP overlay area
    - b. Use user swap area for:
      - i. BASIC program (e.g. REN, APP,...)
      - ii. Work space (examining/modifying DIRECTORY, IDT, ADT).
    - c. No need to swap program out (because they run to completion, or self-suspension).

V. Scheduling - deciding who runs, when, and begin executed of appropriate task.
- A. Only one task has control of CPU at any time. Therefore, when a task is ready to run, it is placed in a waiting list (the queue). Queue is list of those tasks ready to be executed.
- B. Placement in queue depends on:
  - 1. When (first in, first out)
  - 2. Priority level (PLEV) - (queue is actually a queue of four sub-queues. i.e. it is "last-in, last-out" within any given priority level).
- C. STAT - status of task determines its priority level. Possible statuses:
  - 1. idle
  - 2. I/O suspend
  - 3. Syntax processing
  - 4. RUN, LIS, PUN
  - 5. Other commands

D. Priority Philosophy: "Service the interactive user quickl by giving him high priority at the expense of long-running compute-bound programs" (i.e. Users are more concerned with slow syntaxing and slow interation, than long executi time).

<u>Priority Scheme</u>: (PLEV):

0: BASIC syntax
interactive programs
1: Short RUN programs
LIS, PUN
2: Dis-resident routines (other user commands, operator commands)
4: Compute - bound programs
(1 second time-slices).

VI. Scheduler Functions

A. Task <u>scheduling</u> = placing in queue

1.	BASIC line, CR	Syntax
2.	User command, CR	Command
3.	Operator command, CR	Command
4.	Disconnection	Bye

B. Task <u>initiation</u> = begin execution of top-of-queue

C. Task <u>suspension</u> = removing from queue temporarily

D. Task <u>re-scheduling</u> = placing back in queue

1. return from I/O wait

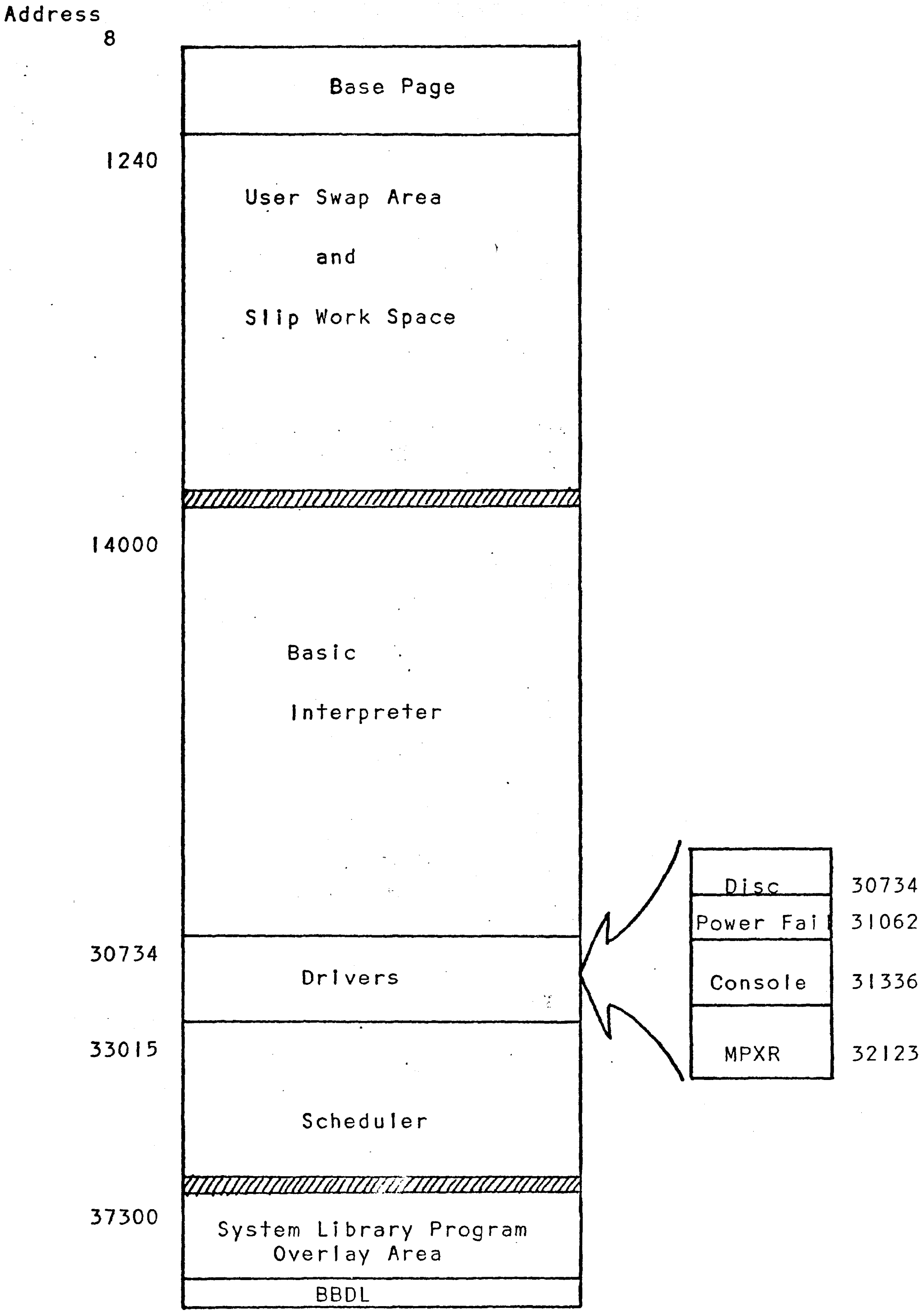
E. Task <u>termination</u> = removing from queue

1. Normal task end (incl. BYE, SLE)
2. User abort

F. Miscellaneous scheduler functions:

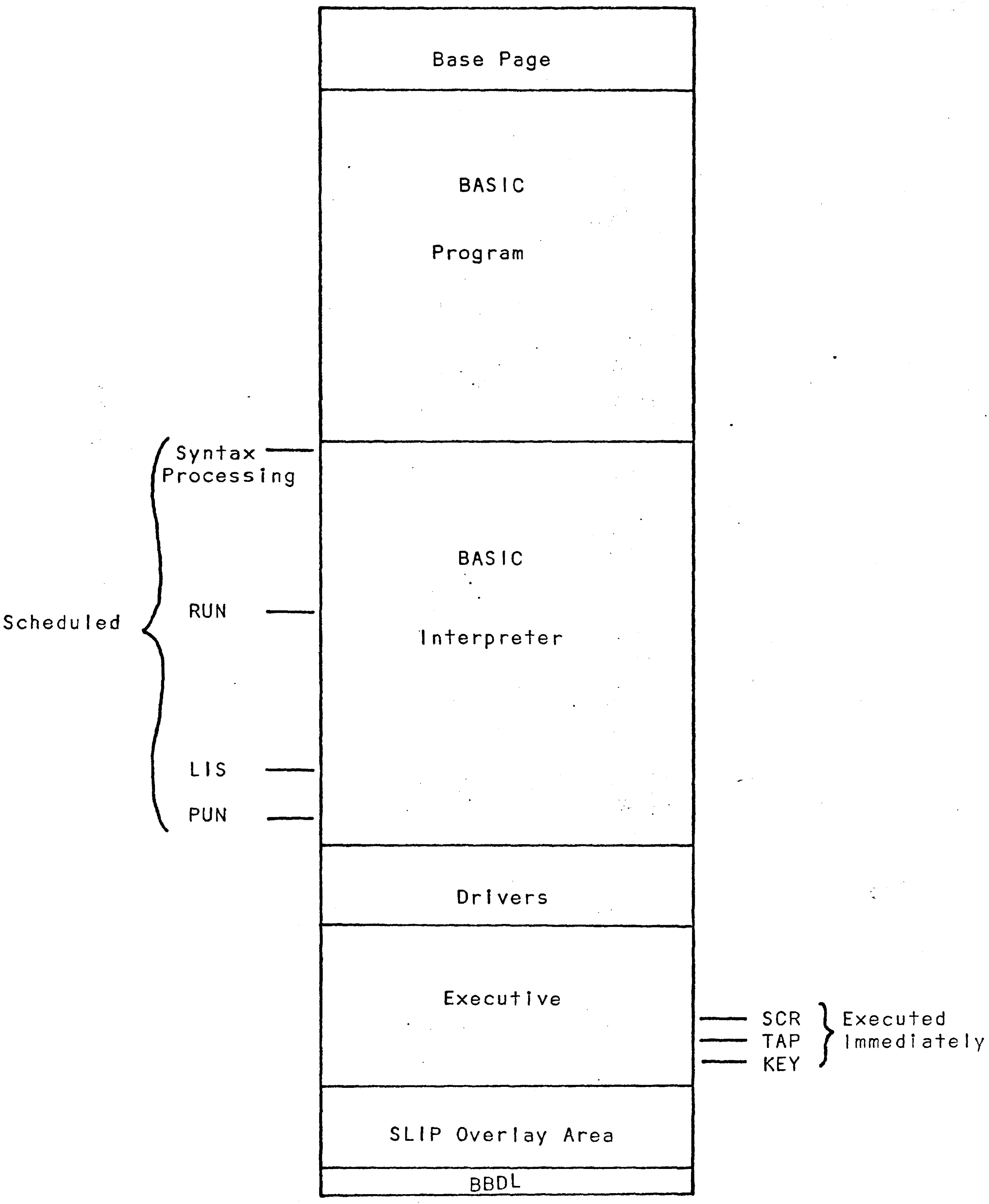
1. Bump top-of-queue, due to higher priority request
2. Time-out RUN jobs. = placing job at bottom of queue

TSB CORE MAP

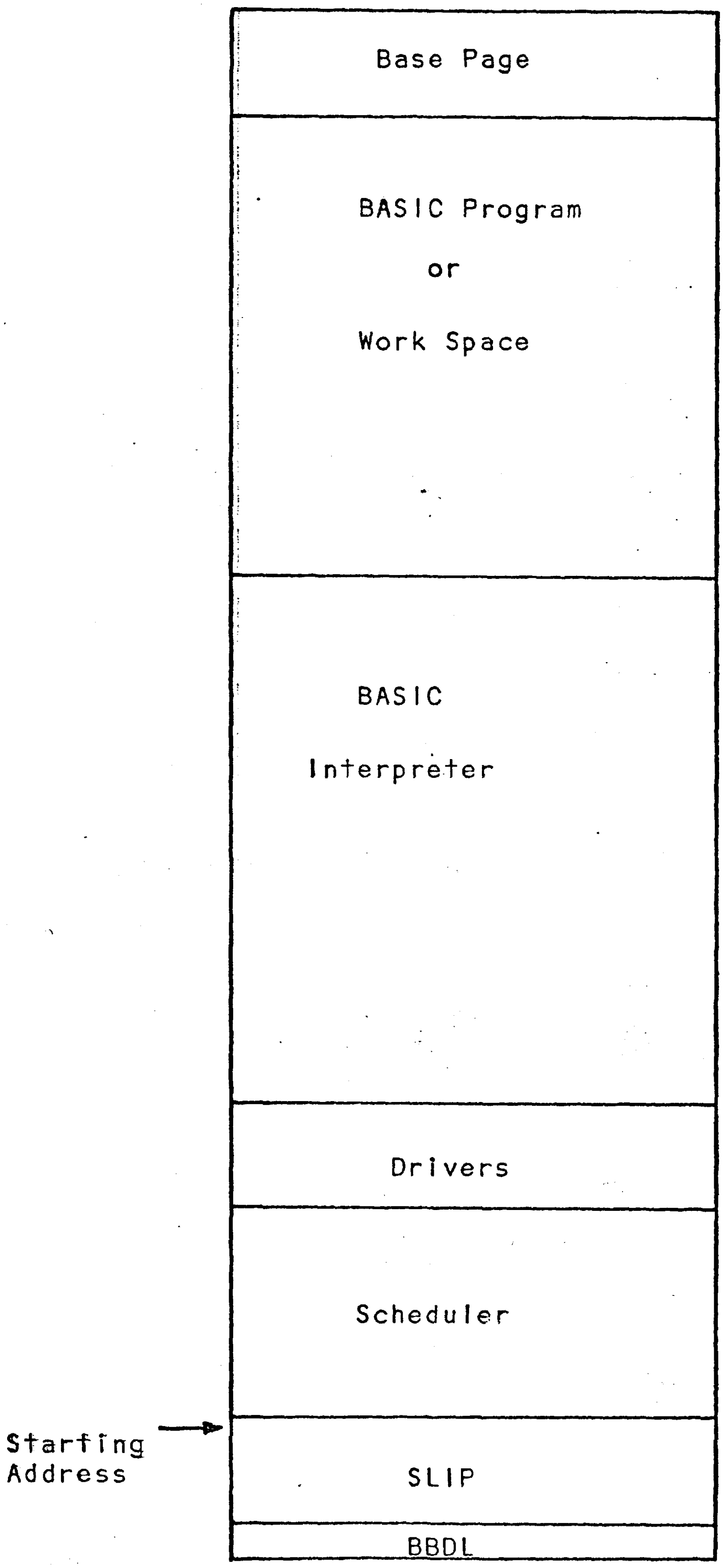


Address
8
Base Page
1240
User Swap Area
and
Slip Work Space
14000
Basic
Interpreter
30734
Drivers
33015
Scheduler
37300
System Library Program
Overlay Area
BBDL
Disc
30734
Power Fail
31062
Console
31336
MPXR
32123

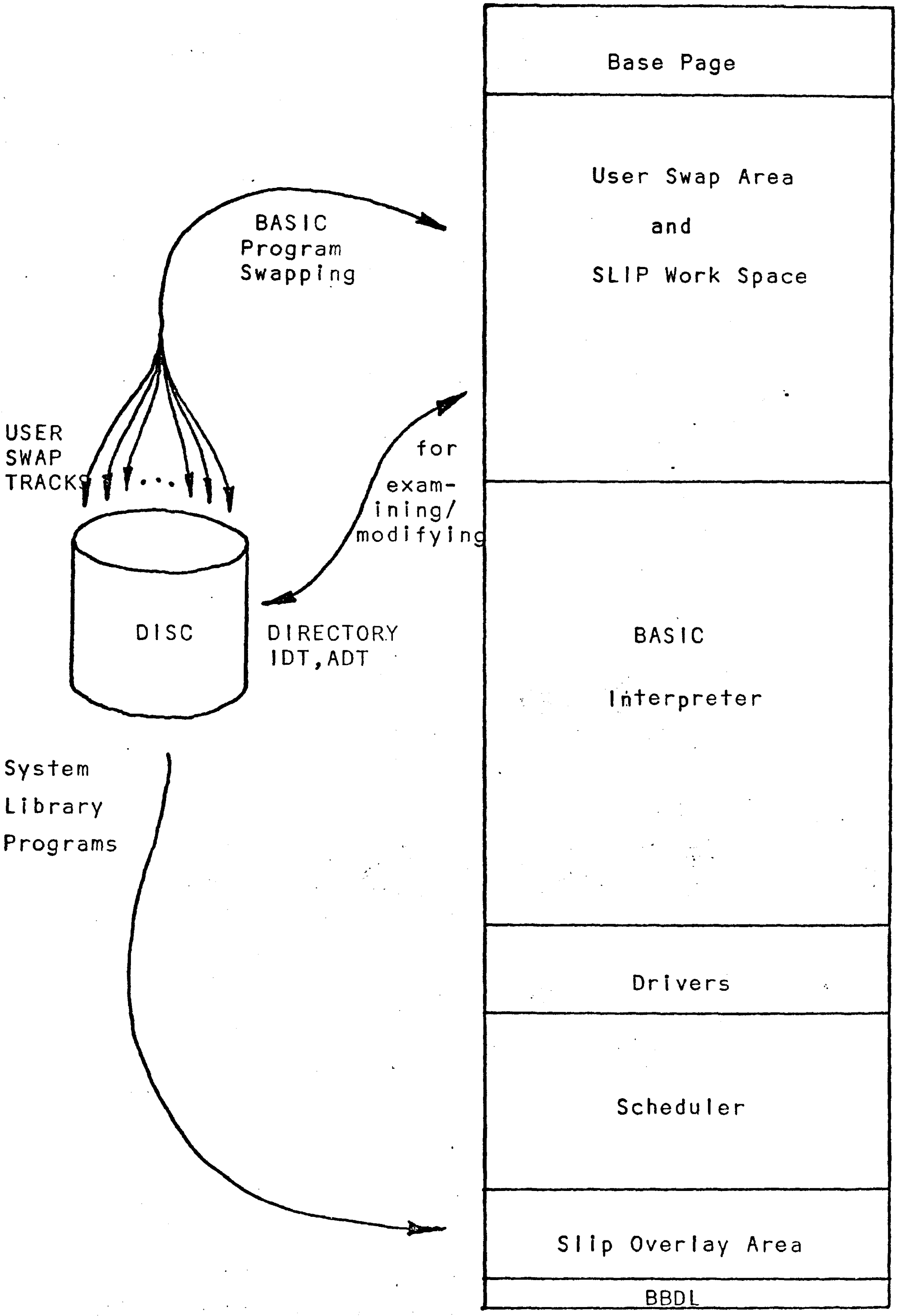
CORE RESIDENT TASKS



DISC RESIDENT TASKS

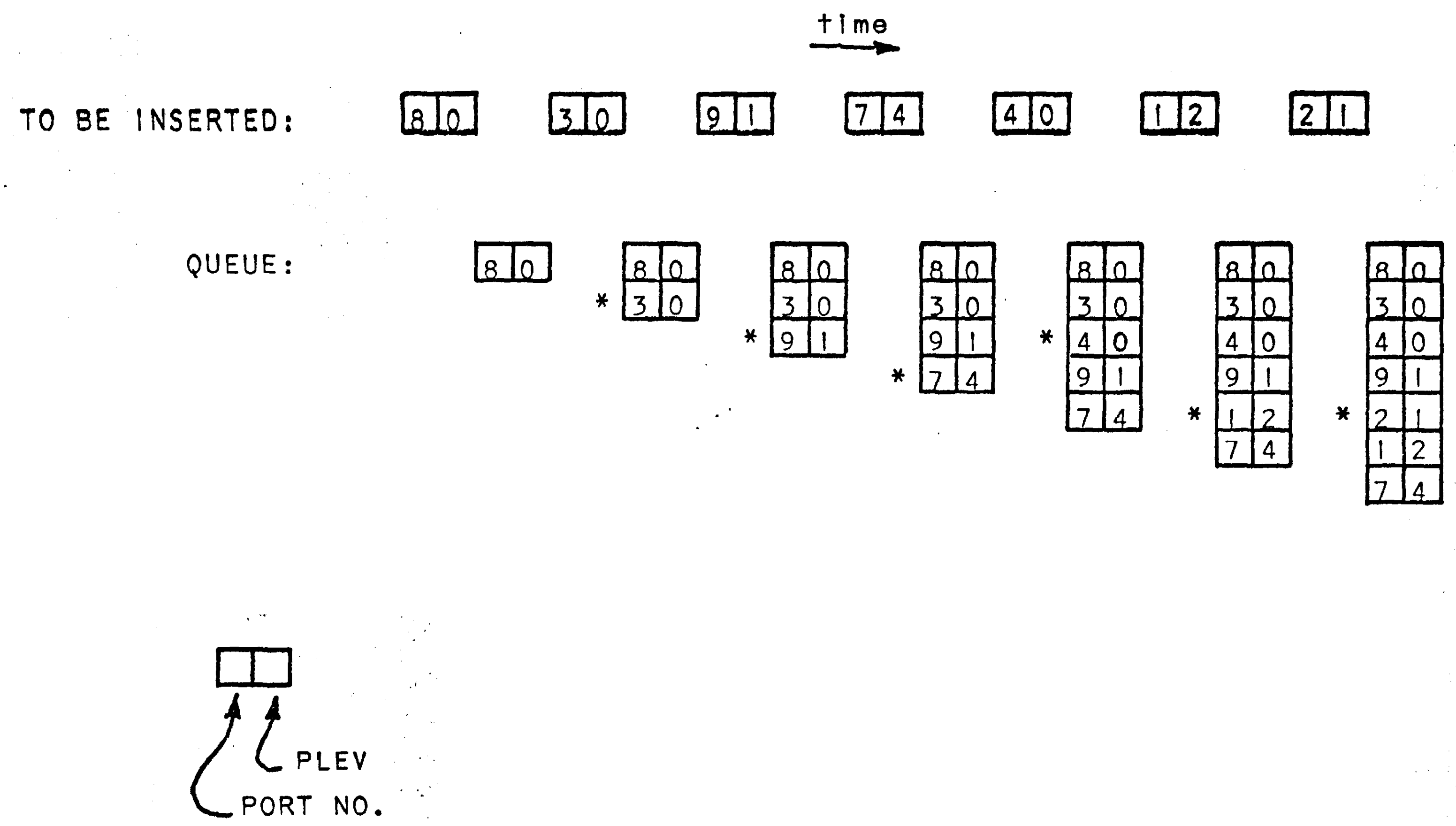


CORE/DISC INTERACTION



QUEUE INSERTION

ACCORDING TO TIME & PLEV



THE MULTIPLEXOR

HP TELEPRINTER MULTIPLEXER INTERFACE

- Permits up to 16 Teleprinters or Bell system data sets, or any combination of the two, to be connected to a 2116B.

- Provides for bit serial transfer of data between the computer and the external device.

- Permits simultaneous input and output.

- The card contains and 880 HZ oscillator which is used by software to establish the sampling rate of the input lines.

- Due to its relatively rapid, asynchronus interrupt rate, it should be assigned a relatively high priority I/O address.

OPTION 01

- Auto disconnect option - allows individual status for each data set to be controlled by computer. This provides protection against "housewife" calls.

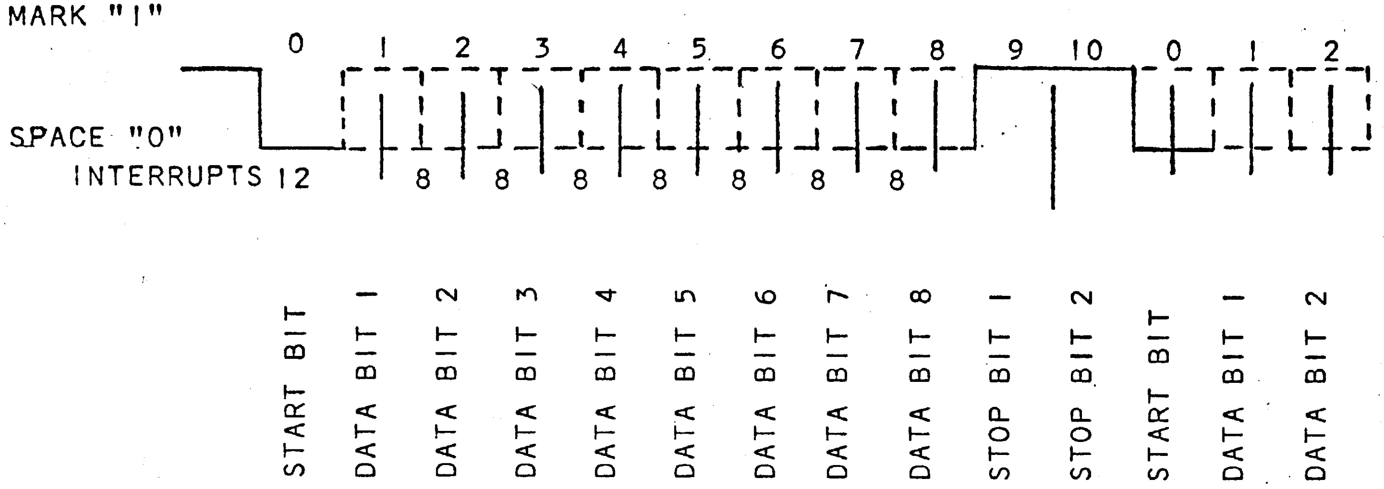
- Also, if an input port is inactive for a time established by computer software, the computer causes a disconnect. The assumption here is that the operator at the remote terminal has completed transmissions, but has not properly terminated the data set.

START BIT 8 INTERRUPTS

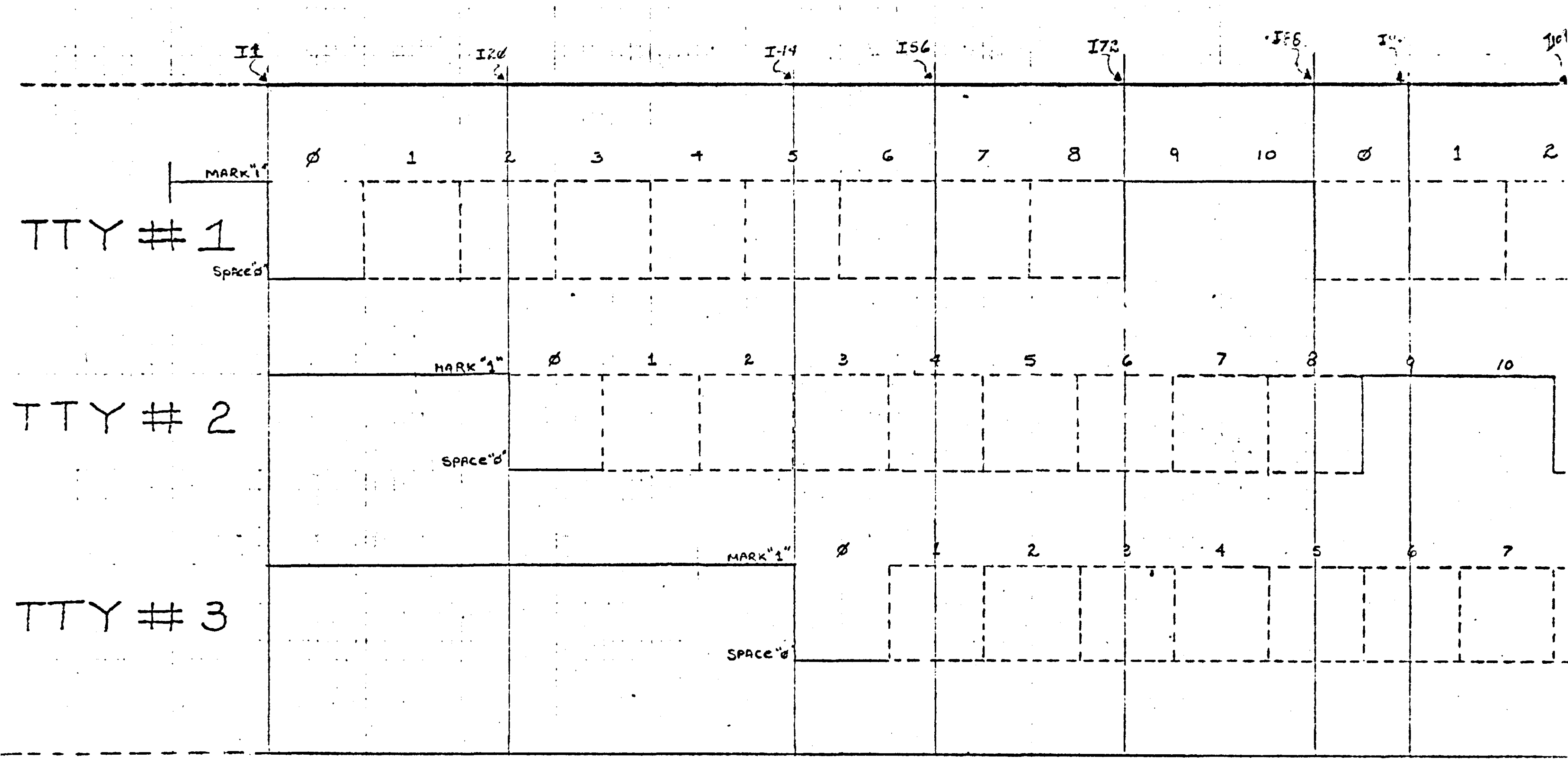
EACH DATA BIT 8 INTERRUPTS

STOP BITS 16 INTERRUPTS

TOTAL INTERRUPTS/CHARACTER = 88



TELEPRINTER MULTIPLEXOR INTERRUPT TIMING DIAGRAM



11 - MULTIPLEXOR INTERRUPT DRIVER SENSES BEGINNING OF CHARACTER TTY #1

120 - MID SENSES BEGINNING OF CHARACTER FOR TTY #2 AND ALSO THAT IT MUST SAMPLE SECOND BIT OF CHARACTER FORM TTY #1

144 - MID SENSES BEGINNING OF CHARACTER FOR TTY #3 AND ALSO TO SAMPLE 5TH DATA BIT OF TTY #1 AND TO DO NOTHING WITH TTY #2

156 - SAMPLE BIT 4 FOR TTY #2 AND SAMPLE BIT #1 FOR TTY #3 AND DO NOTHING FOR TTY #1

172 - END OF CHARACTER FOR TTY #1; SAMPLE BIT 6 OF TTY #2; SAMPLE BIT 3 OF TTY #3

188 -

196 -

1109 -

MPX DRIVER

1. DETERMINE WHEN A START CHARACTER BIT HAS BEEN SENT FROM MULTIPLEXOR

2. ONCE DETERMINED, SET UP WHEN TO SAMPLE SUCCEEDING BITS

3. HOW TO DISTINGUISH BETWEEN SUCCEEDING BITS

   a) DATA BITS
   b) PARITY BITS
   c) STOP BITS

4. ONCE DISTINGUISHED, PERFORM APPROPRIATE ACTION

   a) ⟹ PACK BIT
   b) ⟹ IGNORE
   c) ⟹ IGNORE

5. HOW TO ECHO EACH BIT - START BIT INCLUDED

6. WHEN IS CHARACTER COMPLETE

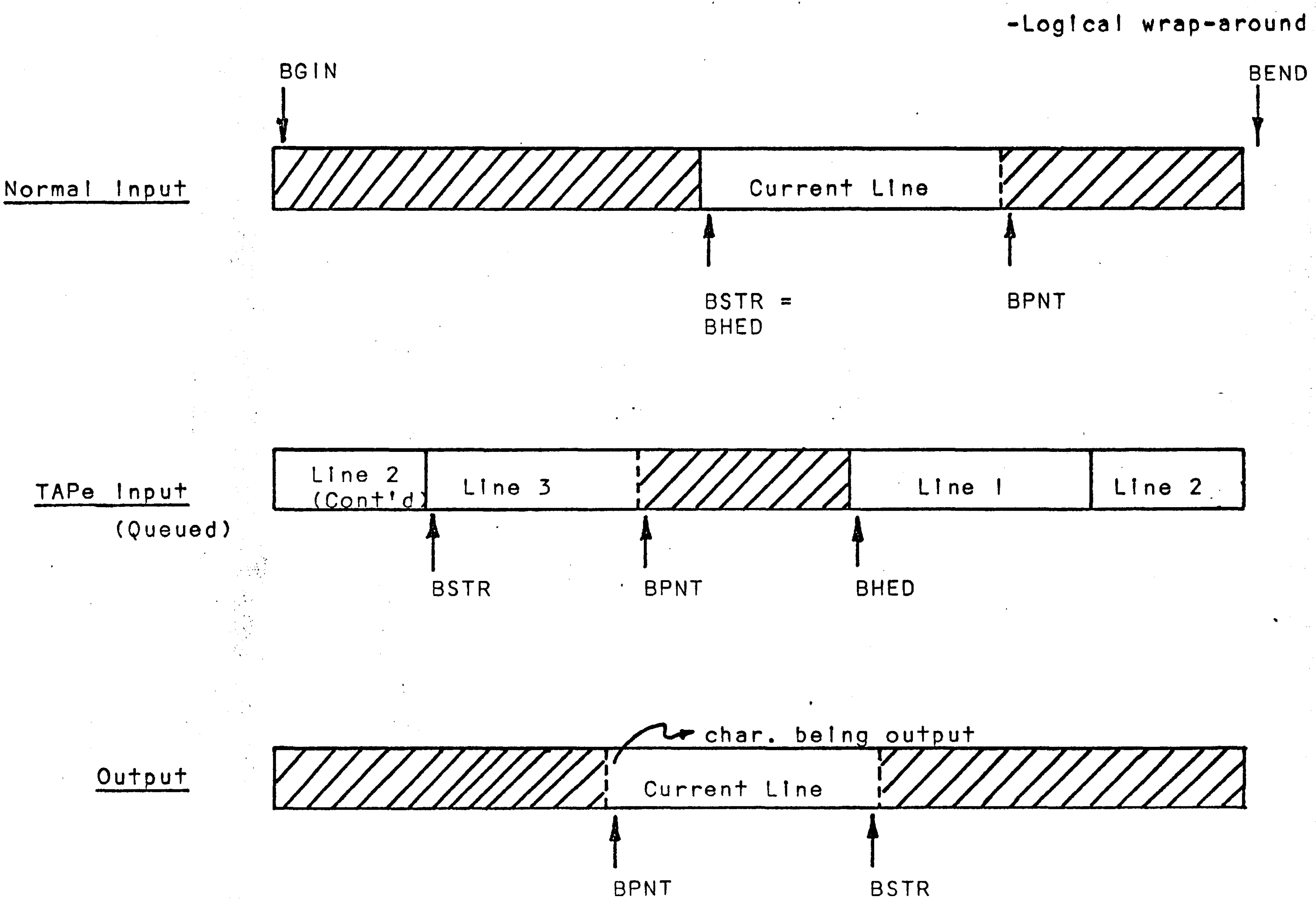
7. BUFFER A LINE

8. HANDLE CHARACTERS OF SPECIAL SIGNIFICANCE

9. OUTPUT SCHEME

10. DETECT SUCCESSFUL ABORT REQUEST AND SIGNAL EXECUTIVE

TTY BUFFERS


-Logical wrap-around
BGIN
BEND
Normal Input
Current Line
BSTR =
BHED
BPNT
TAPe Input
(Queued)
Line 2
(Cont'd)
Line 3
Line 1
Line 2
BSTR
BPNT
BHED
char. being output
Output
Current Line
BPNT
BSTR

TELETYPE TABLES (1/port)

OCTAL WORD NO.		
0	BTIM	Counter for interrupts between bit sampling (initially 4; 8 for each succeeding).
1	CHAR	Packing/unpacking of bits.
2	BCNT	Counter for number of bits withing a character (10 on input, 11 on output).
3	MASK[1]	2 ↑ (port #); e.g.: bit 4 set for port 4.
4	CCNT	**Input:** not used **Output:** negative number of characters to be transmitted.
5	BPNT	**Input:** address[2] of where next character goes. **Output:** address[2] of character currently being output.
6	BSTR	**Input:** address [2] of first character of line currently being input. **Output:** address[2] of last character in buffer.
7	BHED	**Input:** address[2] of first character of first line not yet processed (significant only during tape made input). **Output:** not used.
10	BGIN[1]	Address[2] of beginning of physical buffer.
11	BEND[1]	Address[2] of first character beyond physical buffer.
12	LADR[1]	Address of LADDR entry.

[1] A fixed parameter.

[2] Character-address: bits 15-1 are word address; bit 0 is word-half (0=left half, 1 = right half).

13 DISC Dis-address of swap area.

14 PROG Address of last core location used in swap area. Relevant only when user is not in core. (PBPTR contains last address when user is in core (updated by interpreter).

15 ID User = I.D. 0 = no user on this port.

16-20 NAME Program name.

21 PHON Phone timeing: value of DATIM+1 necessary to force disconnection.

22-23 TIME DATIM, DATIM+1 upon logon.

24 ABCN Counter for user abort request (114 msec).

25 CLOC RUN timeout counter (1 second slices).

26 RSTR Starting address when a task is first scheduled, or re-scheduled when returning from I/O suspend (PREG contains restart address when task is otherwise interrupted).

27 STAT Status

-2 System disconnect
-1 User abort request
0 Idle
1 Aborting
2 Input suspend
3 Output suspend
4 Syntax processing
5-60_8 Command processing STAT number corresponds to COMTAE E.g.: 5=RUN, 6=LIST, 7=PUN, etc.)

30 LINK Links queue: address of LINK word of next (lower) port on queue.

31 PLEV Priority level, when on queue.

0 Syntax
Returning from I/O suspend
System library routines when they reach the top-of-queue.

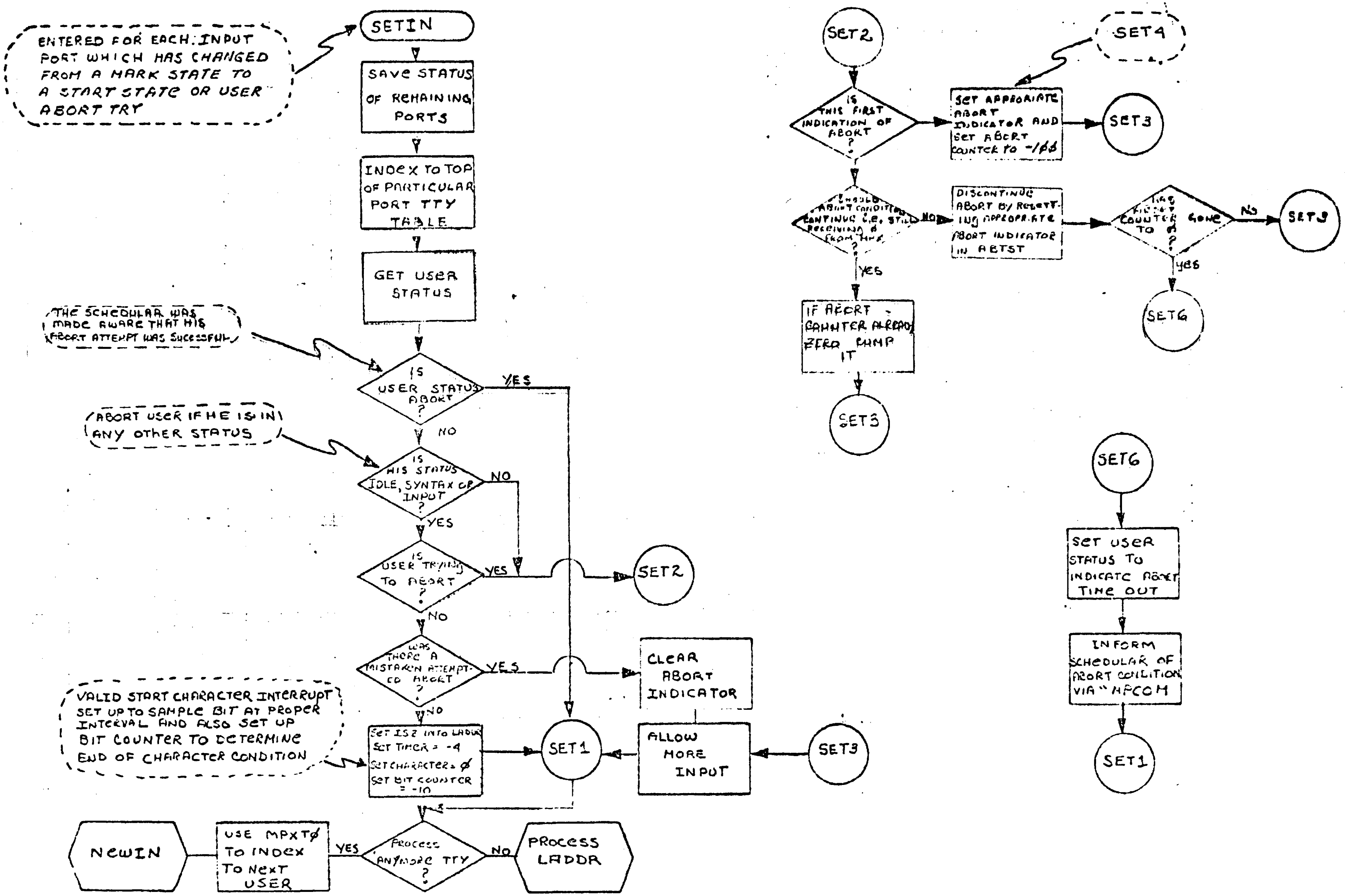
1 RUN, LIST, PUNCH

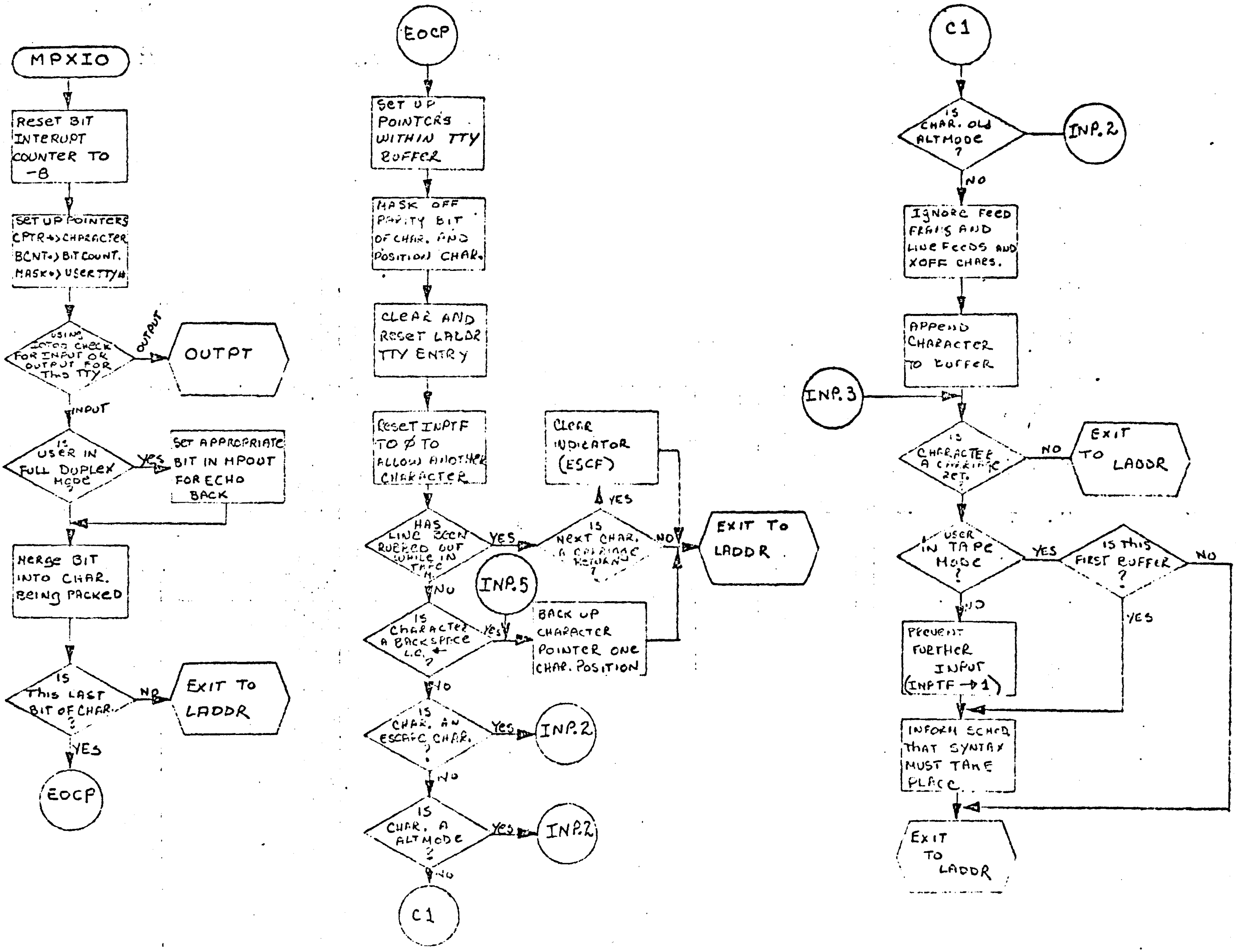
2 System library routines until they reach the top-of-queue.

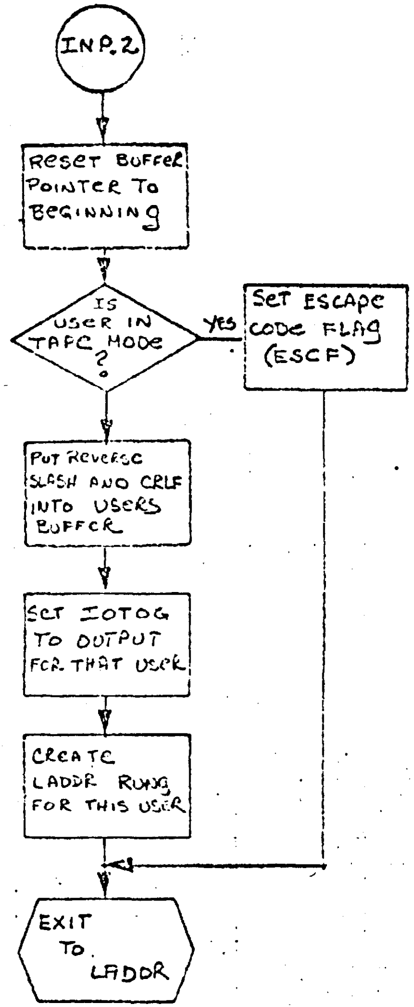
4 Compute-bound (RUN) programs.

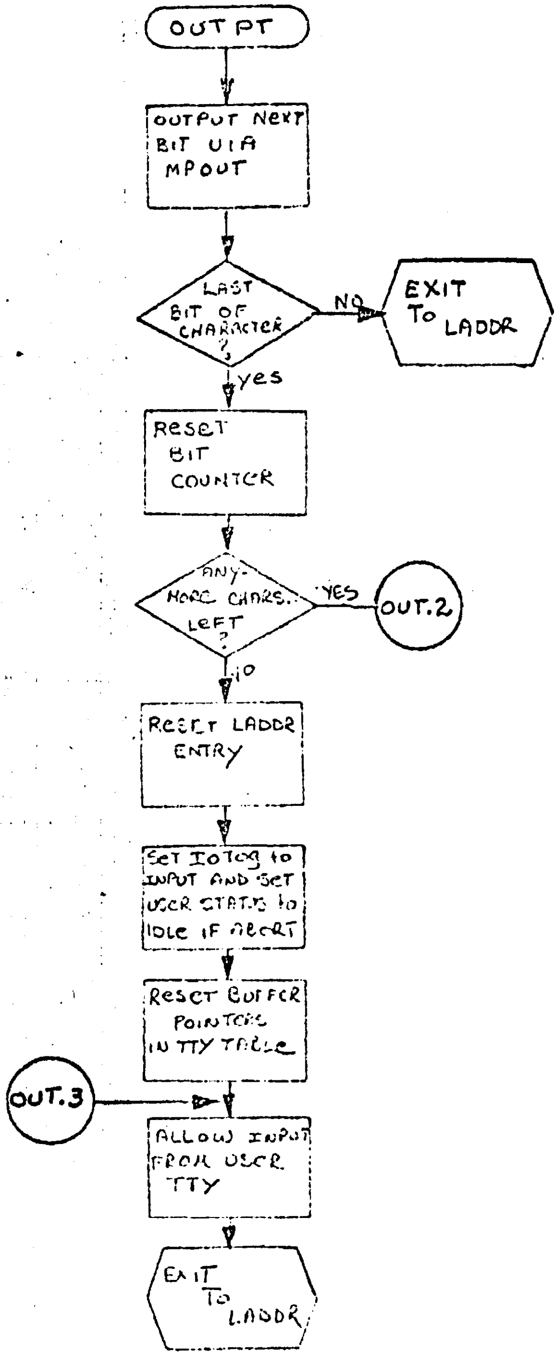
TSB VERSION E, TTY TABLES

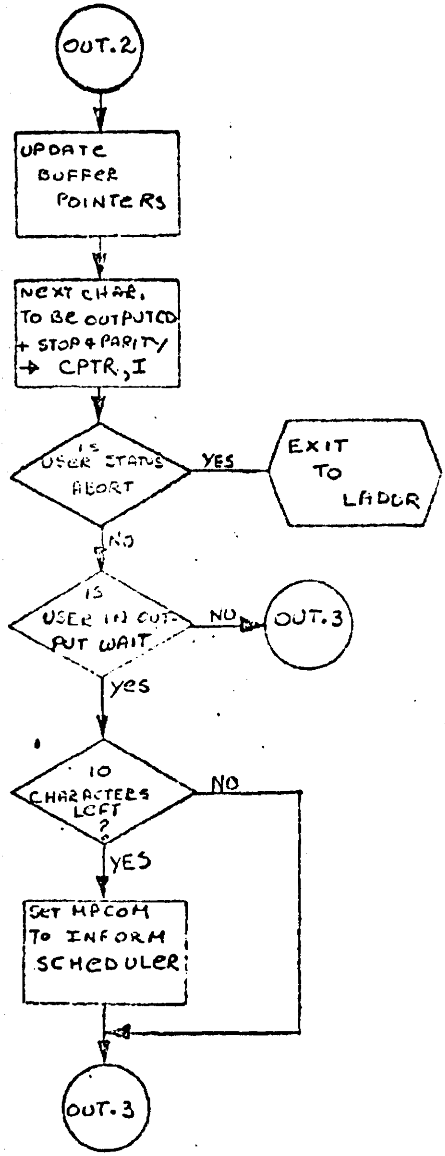
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
BTIM CHAR BCNT MASK CCNT	33015	33047	33101	33133	33165	33217	33251	33303	33335	33367	33421	33453	33505	33537	33571	33623
BPNT BSTR BHED BGIN BEND	33022	33054	33106	33140	33172	33224	33256	33310	33342	33374	33426	33460	33512	33544	33576	33630
LADR DISC PROG ID NAME1	33027	33061	33113	33145	33177	33231	33263	33315	33347	33401	33433	33465	33517	33551	33603	33635
NAME2 NAME3 PHON TIME1 TIME2	33034	33066	33120	33152	33204	33236	33270	33322	33354	33406	33440	33472	33524	33556	33610	33642
ABCN CLOC RSTR STAT LINK	33041	33073	33125	33157	33211	33243	33275	33327	33361	33413	33445	33477	33531	33563	33615	33647
PLEV	33046	33100	33132	33164	33216	33250	33302	33334	33366	33420	33452	33504	33536	33570	33622	33654


ENTERED FOR EACH INPUT PORT WHICH HAS CHANGED FROM A MARK STATE TO A START STATE OR USER ABORT TRY
SETIN
SAVE STATUS OF REMAINING PORTS
INDEX TO TOP OF PARTICULAR PORT TTY TABLE
GET USER STATUS
THE SCHEDULAR WAS MADE AWARE THAT HIS ABORT ATTEMPT WAS SUCESSFUL
IS USER STATUS ABORT ?
YES
NO
ABORT USER IF HE IS IN ANY OTHER STATUS
IS HIS STATUS IDLE, SYNTAX OR INPUT ?
NO
YES
IS USER TRYING TO ABORT ?
YES
SET2
NO
WAS THERE A MISTAKEN ATTEMPTED ABORT ?
YES
CLEAR ABORT INDICATOR
NO
VALID START CHARACTER INTERRUPT SET UP TO SAMPLE BIT AT PROPER INTERVAL AND ALSO SET UP BIT COUNTER TO DETERMINE END OF CHARACTER CONDITION
SET IS2 INTO LADDR
SET TIMER = -4
SET CHARACTER = Ø
SET BIT COUNTER = -10
SET1
ALLOW MORE INPUT
SET3
NEWIN
USE MPXTØ TO INDEX TO NEXT USER
YES
PROCESS ANYMORE TTY ?
NO
PROCESS LADDR
SET2
IS THIS FIRST INDICATION OF ABORT ?
SET4
SET APPROPRIATE ABORT INDICATOR AND SET ABORT COUNTER TO -1ØØ
SET3
SHOULD ABORT CONDITION CONTINUE I.E. STILL RECEIVING Ø FROM MPX ?
NO
DISCONTINUE ABORT BY RESETTING APPROPRIATE ABORT INDICATOR IN ABTST
HAS ABORT COUNTER GONE TO Ø ?
NO
SET3
YES
SET6
YES
IF ABORT COUNTER ALREADY ZERO BUMP IT
SET3
SET6
SET USER STATUS TO INDICATE ABORT TIME OUT
INFORM SCHEDULAR OF ABORT CONDITION VIA "APCOM
SET1


MPXIO
Reset BIT INTERUPT COUNTER TO -8
SET UP POINTERS CPTR→CHARACTER BCNT→BIT COUNT. MASK→USER TTY#
USING CHECK FOR INPUT OR OUTPUT FOR THIS TTY
OUTPUT
OUTPT
INPUT
IS USER IN FULL DUPLEX MODE ?
YES
SET APPROPRIATE BIT IN MPOUT FOR ECHO BACK
MERGE BIT INTO CHAR. BEING PACKED
IS THIS LAST BIT OF CHAR. ?
NO
EXIT TO LADDR
YES
EOCP
EOCP
SET UP POINTERS WITHIN TTY BUFFER
MASK OFF PARITY BIT OF CHAR. AND POSITION CHAR.
CLEAR AND RESET LADDR TTY ENTRY
RESET INPTF TO Ø TO ALLOW ANOTHER CHARACTER
HAS LINE BEEN RUBBED OUT WHILE IN TAPE M.
YES
IS NEXT CHAR. A CARRIAGE RETURN ?
YES
CLEAR INDICATOR (ESCF)
NO
EXIT TO LADDR
NO
INP.5
IS CHARACTER A BACKSPACE I.E. ← ?
YES
BACK UP CHARACTER POINTER ONE CHAR. POSITION
NO
IS CHAR. AN ESCAPE CHAR. ?
YES
INP.2
NO
IS CHAR. A ALTMODE ?
YES
INP.2
NO
C1
C1
IS CHAR. OLD ALTMODE ?
INP.2
NO
IGNORE FEED FRAMES AND LINE FEEDS AND XOFF CHARS.
APPEND CHARACTER TO BUFFER
INP.3
IS CHARACTER A CARRIAGE RET. ?
NO
EXIT TO LADDR
USER IN TAPE MODE ?
YES
IS THIS FIRST BUFFER ?
NO
YES
NO
PREVENT FURTHER INPUT (INPTF → 1)
INFORM SCHED THAT SYNTAX MUST TAKE PLACE.
EXIT TO LADDR


INP.2
Reset Buffer Pointer To Beginning
IS USER IN TAPE MODE ?
YES
Set Escape Code Flag (ESCF)
Put Reverse Slash And CRLF Into Users Buffer
Set IOTOG To Output For That User
Create LADDR Rung For This User
EXIT To LADDR


OUTPT
OUTPUT NEXT BIT VIA MPOUT
LAST BIT OF CHARACTER ?
NO
EXIT To LADDR
yes
Reset Bit Counter
ANY MORE CHARS. LEFT ?
YES
OUT.2
NO
Reset LADDR Entry
Set IOTog to Input And Set User Status to Idle if Abort
Reset Buffer Pointers in TTY Table
OUT.3
Allow Input From User TTY
EXIT To LADDR


OUT.2
UPDATE BUFFER POINTERS
Next Char. To Be Outputed + Stop & Parity → CPTR,I
IS USER STATUS ABORT
YES
EXIT To LADDR
NO
IS USER IN OUTPUT WAIT
NO
OUT.3
yes
NO CHARACTERS LEFT ?
NO
YES
Set MPCOM To Inform Scheduler
OUT.3

TSB TABLES

REFER TO THE MULTIPLEXOR NOTES FOR DESCRIPTIONS OF THE TELETYPE TABLES.

DISC-RESIDENT TABLES

DIRECTORY:

The DIRECTORY is a table which contains all necessary information about each program or file in the system library. It resides on the disc and may occupy from 1 to 4 disc tracks, depending upon how many discs there are on the system. A core resident table called DIREC contains information on the DIRECTORY itself.

A directory entry consists of 8 words and has the following format:

WORD	
Ø	USER I.D.
1 2 3	PROGRAM OR FILE NAME
4	UNUSED
5	DATE
6	DISC ADDRESS
7	-LENGTH IN WORDS

I.D. TABLE

The I.D. table (IDT) is a disc resident table which contains one 8-word entry for each I.D. code on the system. The entries are kept sorted according to the I.D. codes. An entry has the following format:

WORD: Ø user I.D.
1-3 password (filled with Ø's if fewer than 6 characters)
4 time allowed (in minutes)
5 time used (in minutes)
6 disc allowed (in sectors)
7 disc used (in sectors)

Words 4 - 7 are 16 bit quantities with values between Ø and 65535.

AVAILABLE DISC TABLE

The available disc table (ADT) is a disc resident table which contains one two-word entry for each area of the disc which is unallocated. An entry has the following form:

WORD	Ø disc address
	1 length in sectors

Entries are sorted according to word Ø. Each entry may refer to as much as one full track, and no two consecutive entries ever refer to two adjacent disc areas (two tracks are not considered to be adjacent).

Besides the entries for unallocated areas, there is also one ADT entry for each of the five tracks on which the system itself resides, and for each of the sixteen tracks allocated for user swapping.

FUSS TABLE:

The FUSS table is a 128 word table which resides on the disc.

FUSS is divided into 16 sections of 8 words each. The 8 words in each section are the disc addresses of the user files currently being accessed by the user corresponding to that table. Addresses with bit 7=1 indicates the user has read only access.

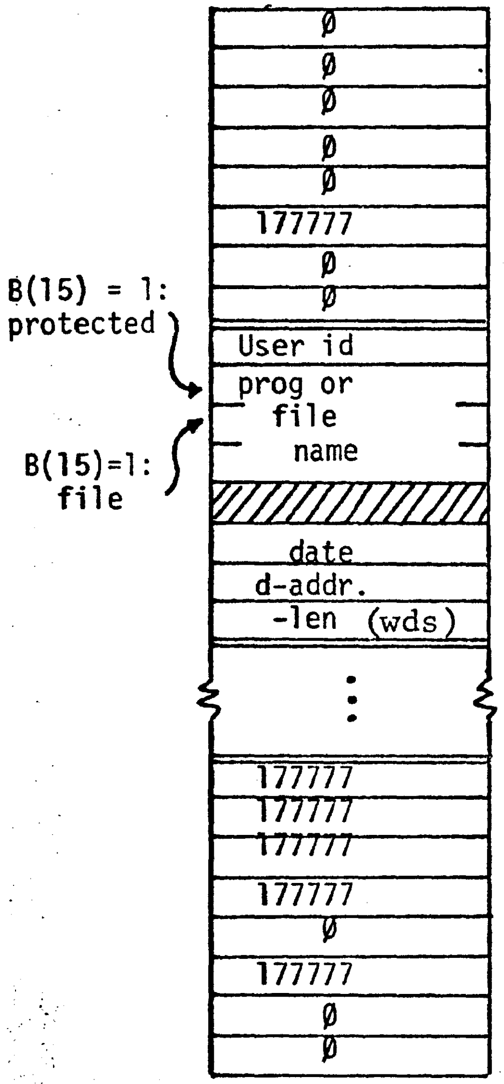
The purpose of maintaining this table is to:

1) prevent simultaneous write access by two users to one file
2) prevent KILLing a file when some user has access to it

DISC-RESIDENT TABLES

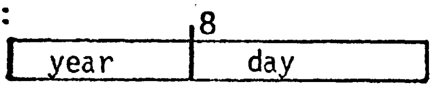
DIRECTORY

8 wd. entry/file,prog.



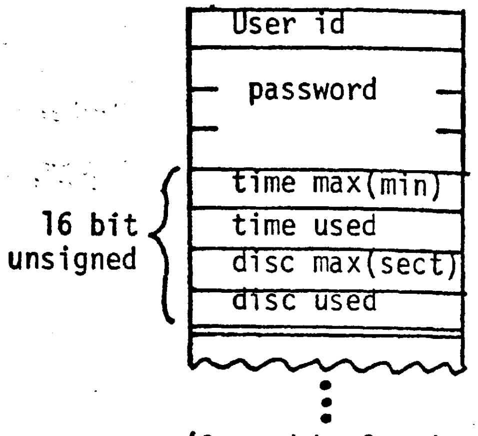
(Sorted by 1st 4 words without bit 15's)

DATE:



IDT

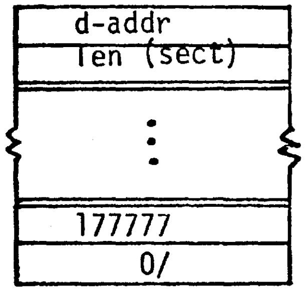
8 wd. entry/user



(Sorted by 1st 4 words)

ADT

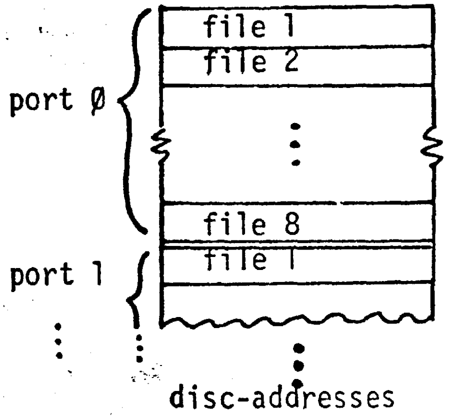
2 wd. entry/space



(Sorted by 1st words)
(System tracks are Ø-le

FUSS

8 wd. entry/port



EQUIPMENT TABLE (EQT)

(CORE RESIDENT)

DIREC

DIREC is a core resident table which contains information about the disc directory. It has the following structure:

WORD	Ø	length in words of first directory track
	1 - 4	same as first 4 words of first directory track
	5	unused
	6	disc address of first directory track
	7 -13	same as 0 - 6 but applied to 2nd directory track
	14 -20	same as 0 - 6 but applied to 3rd directory track
	21 -27	same as 0 - 6 but applied to 4th directory track

The disc address of a directory is always sector Ø of a track. Each directory track may contain as many as 5440 words = 85 sectors = 680 directory entries.

IDLOC: disc address of IDT table

IDLEN: negative length in words of IDT

ADLOC: disc address of ADT table

ADLEN: negative length in words of ADT

TRAX: this is a table of which disc tracks are physically available to the system. Locations 140 - 143 correspond to disc Ø, 144 - 147 to disc 1 etc. Track Ø of disc Ø is represented by bit Ø of 140, etc. A bit is Ø when the track is available, 1 when unavailable.

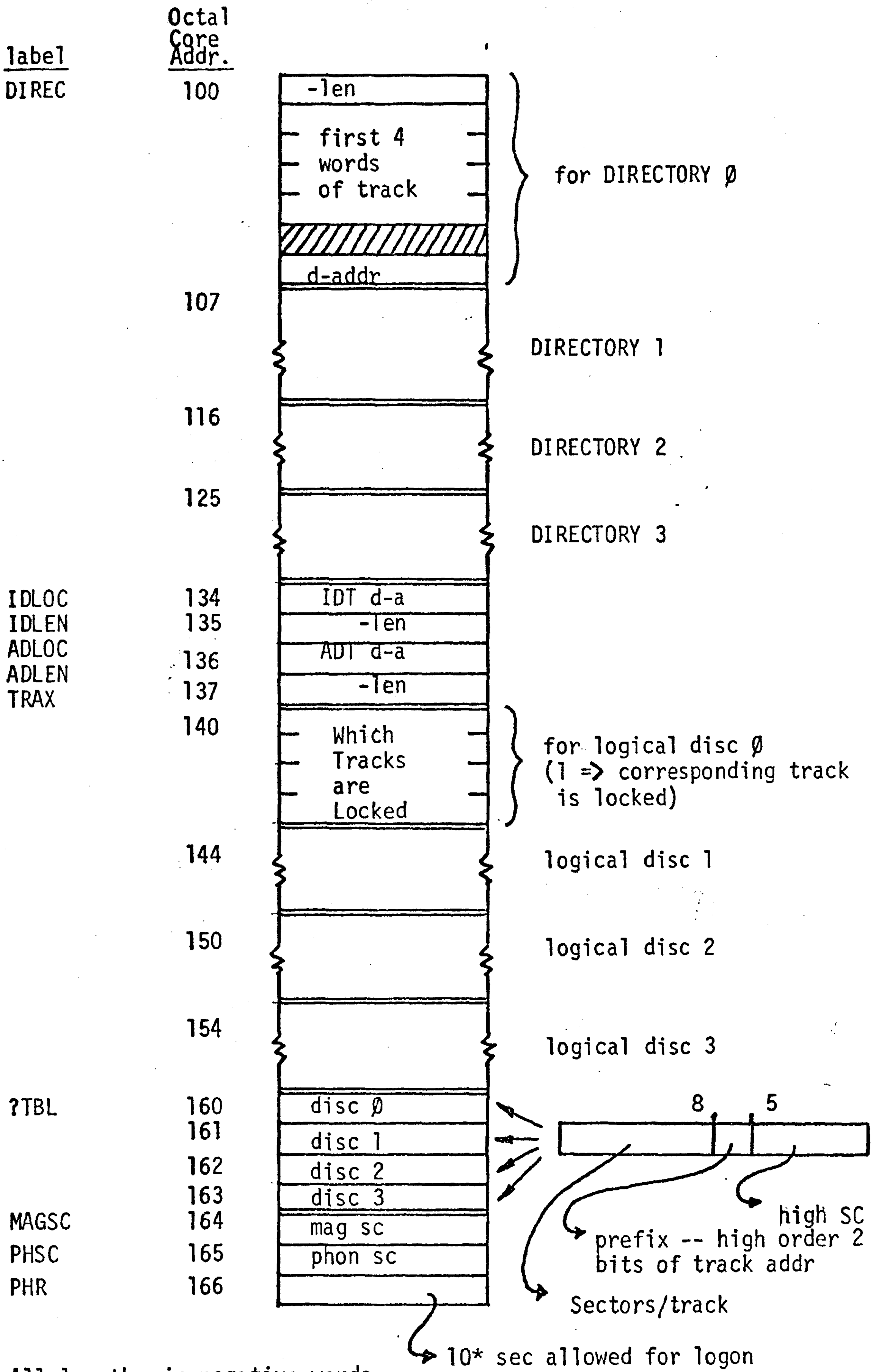
?TBL There is one word in this area for each of the four discs. When the word is zero, the particular disc does not exist. Otherwise bits 15:8 contain the number of sectors/track, bits 7:6 the disc prefix, and bits 5:0 the high priority select code. The prefix is used by the disc driver as the high order 2 bits of the 8-bit track address.

<u>MAGSC:</u> high priority select code for mag. tape; if nonexistent, MAGSC=Ø

<u>PHSC</u>: select code for auto disconnect board, if nonexistent, PHSC =Ø

<u>PHR:</u> = 10 X number of seconds allowed for user to log on; applicable only if PHSC ≠0

THE EQUIPMENT TABLE (EQT)

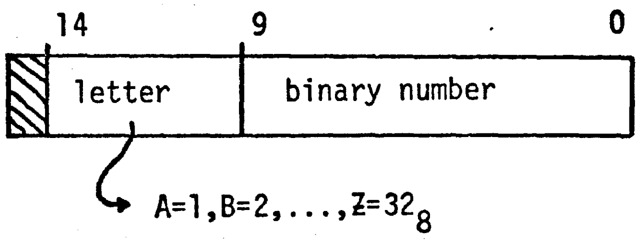


label	Octal Core Addr.		
DIREC	100	-len	for DIRECTORY Ø
		first 4 words of track	
		▨	
		d-addr	
	107		DIRECTORY 1
	116		DIRECTORY 2
	125		DIRECTORY 3
IDLOC	134	IDT d-a	
IDLEN	135	-len	
ADLOC	136	ADT d-a	
ADLEN	137	-len	
TRAX	140	Which Tracks are Locked	for logical disc Ø (1 => corresponding track is locked)
	144		logical disc 1
	150		logical disc 2
	154		logical disc 3
?TBL	160	disc Ø	
	161	disc 1	
	162	disc 2	
	163	disc 3	
MAGSC	164	mag sc	
PHSC	165	phon sc	
PHR	166		10* sec allowed for logon

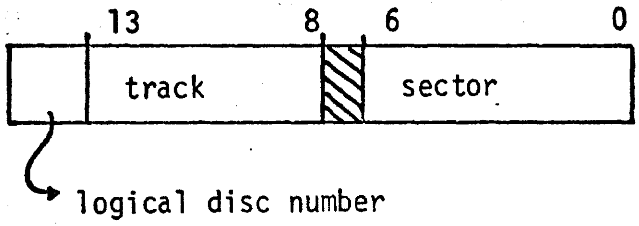
All lengths in negative words
"d-a" -- disc-address
"sc" -- select code

TSB WORD FORMATS

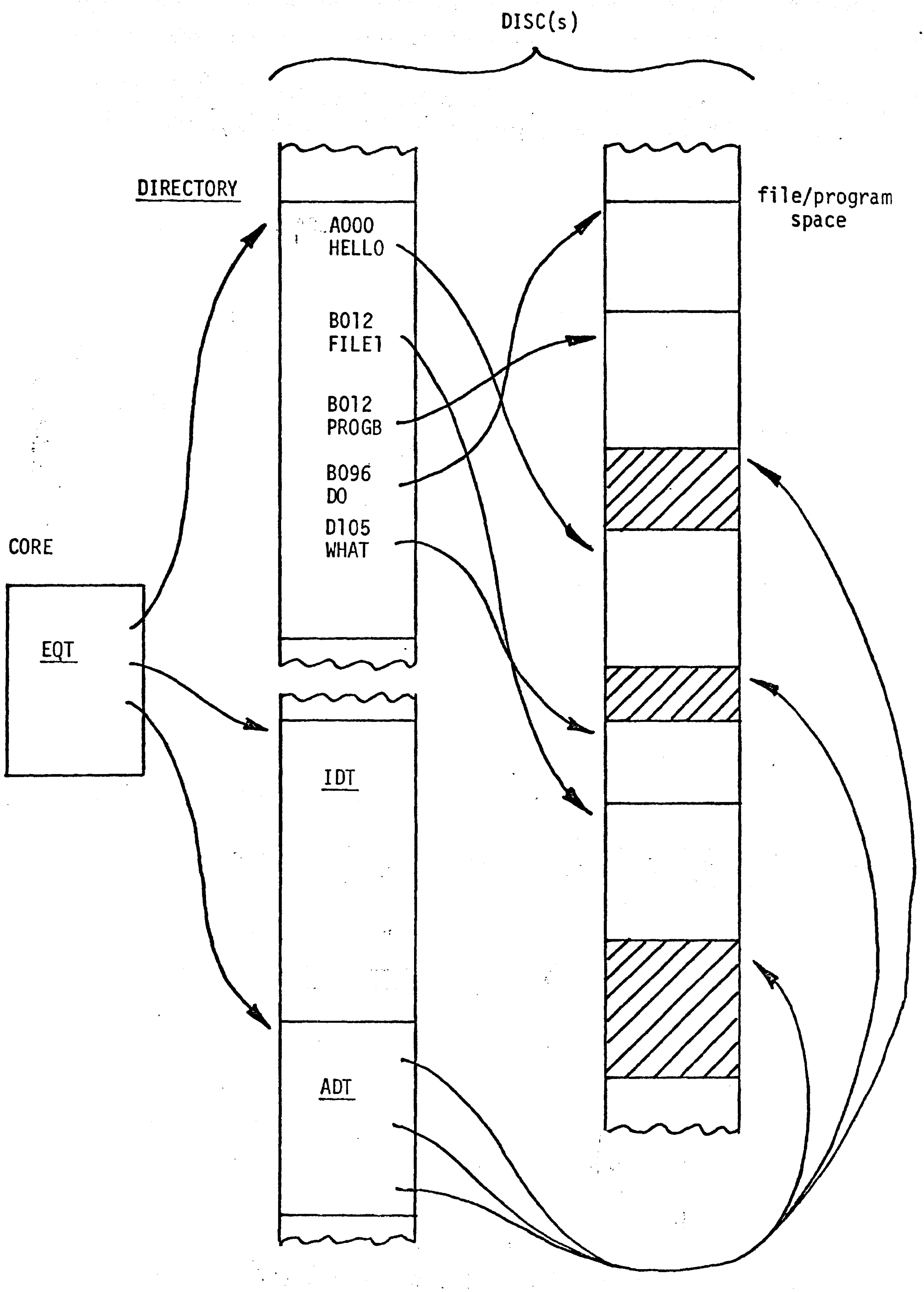
ID's



Disc-Addresses

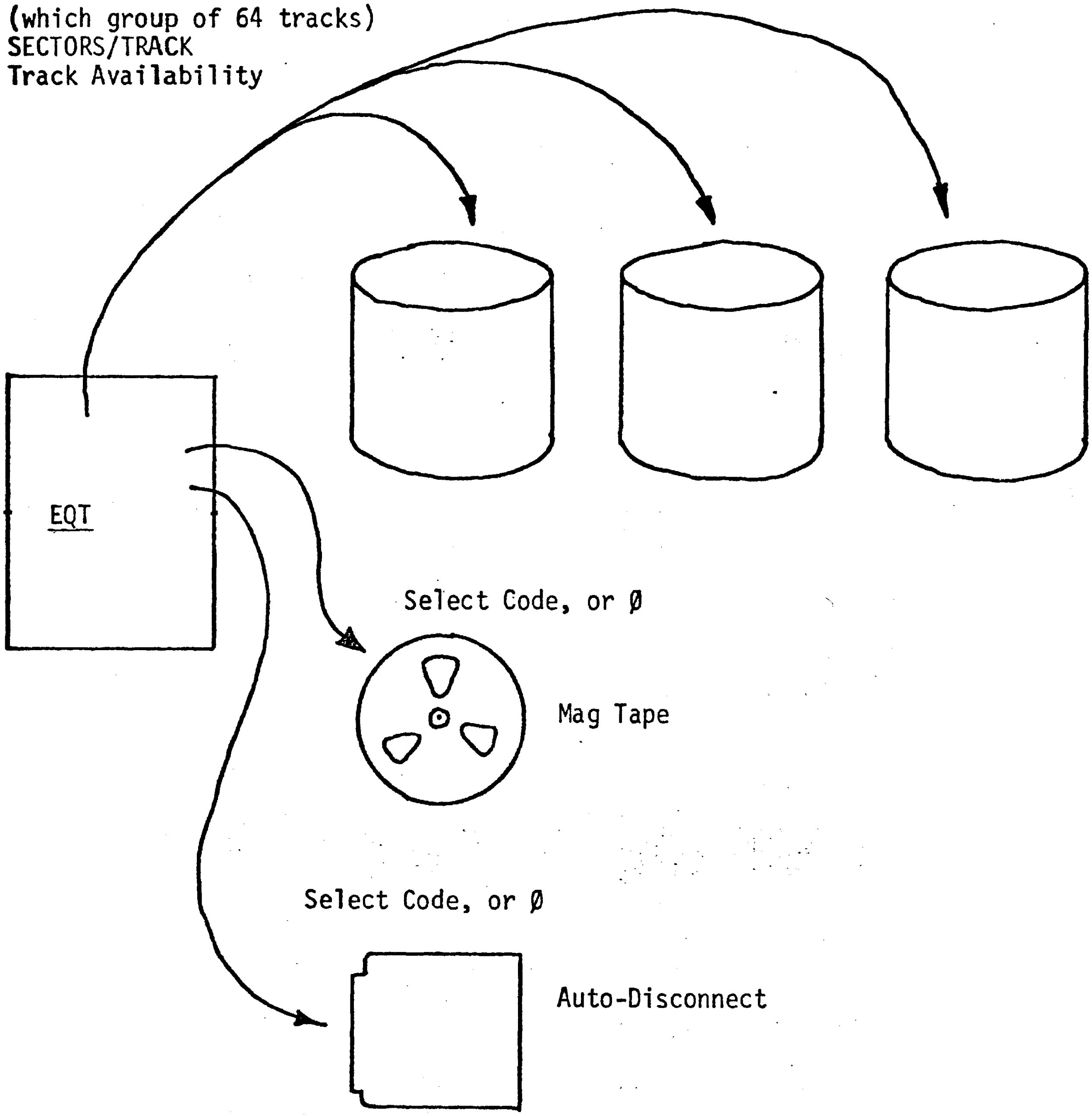


YEAR	year	year
DATIM	24 * day + hour	hour of year
DATIM+1	600*min+10*sec+1/10sec-36000	1/10 sec of hour.


DISC(s)
DIRECTORY
A000
HELLO
B012
FILE1
B012
PROGB
B096
DO
D105
WHAT
file/program
space
CORE
EQT
IDT
ADT

DESCRIPTIONS

Select code
(which group of 64 tracks)
SECTORS/TRACK
Track Availability



TSB TABLES EXERCISE

In fig. 1 are four groups of commands and their associated error diagnostics. For each group, indicate which TSB table is accessed to determine the error conditions of that group.

DIRECTORY

The DIRECTORY contains information about all the files and programs in the system.

1. Where is it maintained?

2. Why?

Each entry in the DIRECTORY describes a particular file or program.

3. Consider the GET command. What information must each entry contain in order for it to work?

4. There is no confusion between identically named programs for different users. What does this imply about the argument of the search in GET?

5. Consider the PURGE command. What information must each entry contain in order for it to work?

fig. 1

Group 1

SAVE	DUPLICATE ENTRY
GET	NO SUCH PROGRAM
	ENTRY IS A FILE
KILL	NO SUCH PROGRAM
PROTECT	NO SUCH PROGRAM
OPEN	DUPLICATE ENTRY

Group 2

SAVE	SYSTEM OVERLOAD
OPEN	SYSTEM OVERLOAD

Group 3

SAVE	FILE SPACE FULL
HELLO	ILLEGAL ACCESS
	NO TIME LEFT
OPEN	FILE SPACE FULL

Group 4

KILL	FILE IN USE

6. Somehow we must prevent GETing files. How can we do this?
7. How can we prevent LISTing protected programs, or accessing protected files?
8. For efficiency in searching, how is the DIRECTORY ordered?
9. What operator command lists (almost) all of the information in the DIRECTORY?
10. What doesn't it list?

IDT

Each entry in the IDT contains information associated with each user in the system.

For the commands listed below, indicate what information the IDT must contain in order for the command to work:

1. HELLO
2. SAVE
3. How should the IDT be structured?

ADT

Each entry in the ADT describes an available area on the discs.

1. How can such an area be defined?
2. What commands access the ADT?

SCHEDULER

I. Schedule summary
   A. Tasks compete for CPU control
   B. Ready tasks are queued
   C. Placement on queue by:
      1. Priority level - each task is queued ahead of all lower priority tasks
      2. Time-each task is queued behind all equal and higher priority tasks
   D. Priority level determined by status of task
   E. Functions of scheduler

II. Task statuses

III. Task priorities

IV. The queue

V. Macro flow chart as implementation of necessary scheduler functions

VI. Changes in task status/priority

VIII. Example

SCHEDULER FUNCTIONS

Task Scheduling = placing in queue

1.	BASIC line, CR	Syntax
2.	User command, CR	Command
3.	Operator command, CR	Command
4.	Disconnection	BYE

Task initiation = begin execution of top-of-queue

Task suspension = removing from queue, temproarily

1. I/O suspend - normal, or forced (by some SLIP's)

Task re-scheduling = placing back on queue

1. Return from I/O suspend

Task termination = removing from queue

1. Normal task end (incl. BYE, SLE)
2. User abort

Task interruption = remove from top- of-queue (but still in queue)

1. Bump due to higher priority task
2. Time out RUN jobs - place on bottom

STAT Values

-2	System disconnect (only with PHONES) - Not logged on in time - Lost carrier - BYE
-1	User abort request - Break key held > 114 msec
0	Idle
1	Aborting
2	Input wait - Task requests input
3	Output wait - Buffer is full on output - Library routine
4	Syntax Processing
≥5	Command - value corresponds to command's position in COMTABLE

STAT transitions

a →	-2	Disconnection
-1 →	1	Abort request detected by scheduler
0 →	-2, or≥4	Initiate task or disconnection
1 →	0	"STOP" message complete
2,3→	previous STAT	Task returning from I/O suspend
≥4→	0	Task completion
≥5→	2,3	Into I/O suspend

"a" is anything

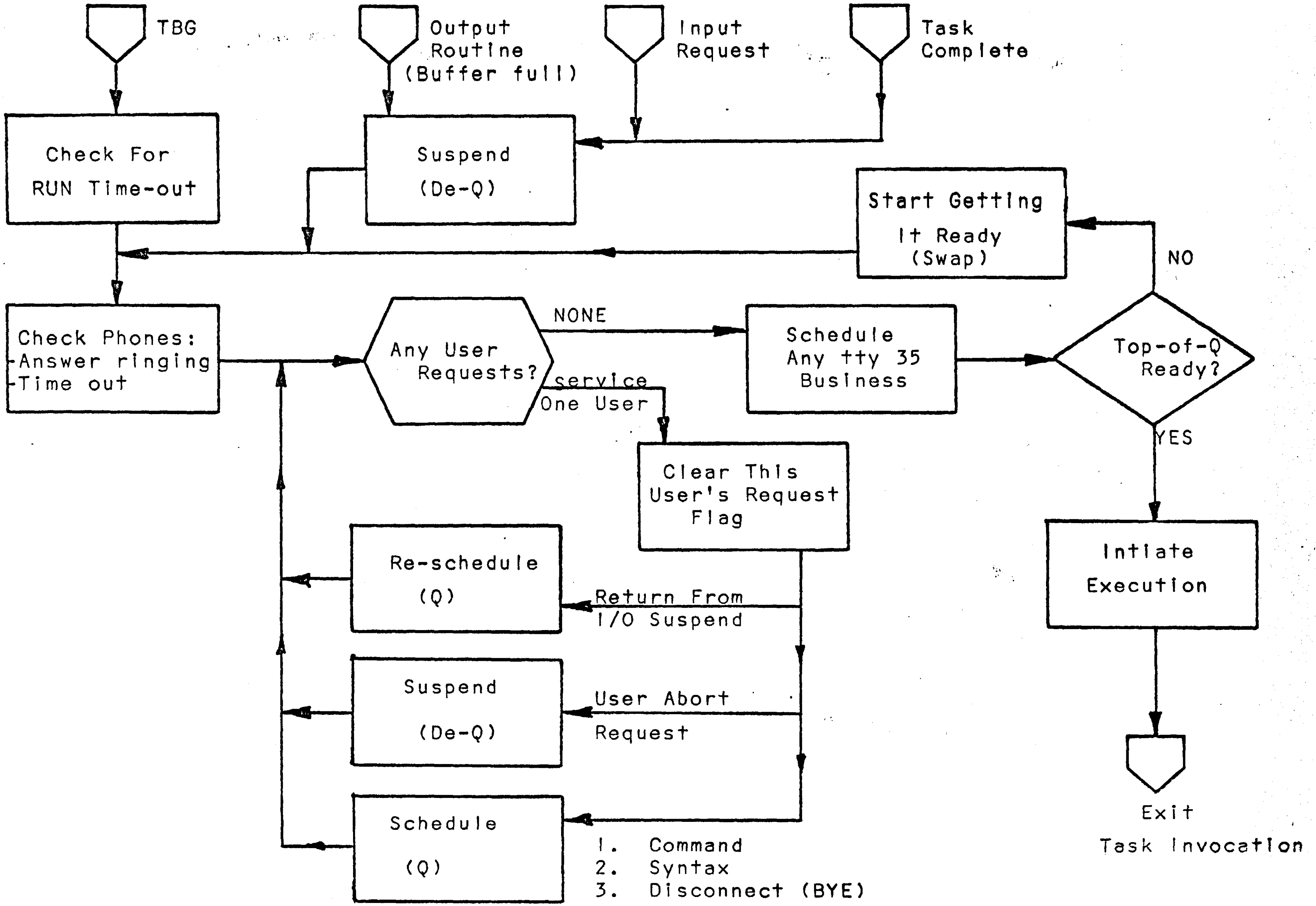
PLEV values

0	Syntax processing Return from I/O suspend Disc-resident routines when at top-of-queue
1	RUN (initially), LIST, PUNCH
2	Disc-resident routines before reaching top-of-queue
4	Compute-bound programs

PLEV transitions

1 → 4	RUN job timed-out
0 → 4	RUN job (that returned from I/O suspend) timed-out
2 → 0	SLIP reaching top-of-queue
→ 0	Return from I/O suspend
→ 2 (BYE)	Disconnection

TSB SCHEDULER



TASK STATUS TRANSITIONS

FROM					TO			
STAT	PLEV	MEANING	SET BY	WHEN	RESET TO	PLEV	BY	WHEN
-2	OFF Q	DISCONNECT [3]	PHONES	1. NOT LOGGED IN IN TIME, OR 2. CARRIER LOST FOR 1 SEC.	-2 [5]	2	SCHEDULER	DETECTION OF FLAG[1] FROM PHONES INDICATING DISCONNECTION
-1	OFF Q	USER ABORT REQUEST	MPXR ABORT TIMER	BREAK KEY > 114 MSEC[2]	1	OFF Q	SCHEDULER	DETECTION OF FLAG[1] FROM MPXR INDICATING ABORT REQUEST
0	OFF Q	IDLE	SCHEDULER MPXR OUTPUT ROUTINE	TASK TERMINATION COMPLETION OF "STOP", WHEN ABORTING (I.E. OUTPUT COMPLETE AND STAT=1)	NORMAL STATE - CAN GO TO -2,≥4			
1	OFF Q	ABORTING	SCHEDULER	WHEN TYPING "STOP"	0	OFF Q	MPXR OUTPUT ROUTINE	COMPLETION OF OUTPUT
2	PREVIOUS STAT	INPUT WAIT	SCHEDULER	TASK REQUESTS INPUT	PREVIOUS STAT	0	SCHEDULER	DETECTION OF FLAG[1] FROM MPXR INDICATING CR RECEIVED
3	PREVIOUS STAT	OUTPUT WAIT	SCHEDULER OUTPUT ROUTINE SCHEDULER OUTPUT SUSPEND	OUTPUT BUFFER IS FULL REQUESTED BY LIBRARY ROUTINE	PREVIOUS STAT	0	SCHEDULER	DETECTION OF FLAG[1] FROM MPXR INDICATING ONLY 10 CHARACTERS LEFT IN BUFFER
4	0	SYNTAX	SCHEDULER	DETECTION OF FLAG[1] FROM MPXR INDICATING CR, & A NUMBERED LINE DURING IDLE	0	OFF Q	SCHEDULER	SYNTAX COMPLETION
5,6,7 > 7	1, 4 2	RUN [4] LIST PUNCH CORRESPONDING COMMAND	SCHEDULER	1. DETECTION OF FLAG FROM MPXR INDICATING CR, & UNNUMBERED LINE DURING IDLE, OR 2. DETECTION OF FLAG[1] FROM MPXR INDICATING RETURN FROM I/O SUSPEND	0	OFF Q	SCHEDULER	TASK COMPLETION
					2, 3	OFF Q	SCHEDULER	I/O SUSPEND

NOTES:

1. FLAG IS BIT SET IN MPCOM.
2. MPXR ACCEPTS NO INPUT (INCL. BREAK) UNTIL ACKNOWLEDGING LF IS OUTPUT BY TASK.
3. ANY STATUS TASK CAN ENTER DISCONNECT.
4. RUN JOBS' PLEV CHANGE TO 4 UPON TIME RUN-OUT. STAT REMAINS SAME.
5. ALTHOUGH STAT REMAINS -2, SCHEDULER TREATS DISCONNECT AS 'BYE' COMMAND.

EXPLANATION OF THE QUEUE SEQUENCE EXAMPLE

The example is one of a TSB system with 7 ports (2,3,6,7,8,9,10) logged on. Each frame depicts the status (STAT) and priority (PLEV) of each user as well as the order of the queue at that time. The queue is the top group of entries with the top-of-queue on top. Each port entry shows the port number in the left box, that user's status in the middle box, and his PLEV in the left box. The time between frames is strictly relative.

1 - 8 is doing syntax.

2 - 10 requests syntax processing. Note: does not receive CPU control immediately due to 8 not being finished.

3 - 8 finishes. 10 receives control.

4 - 9 requests RUN.

5 - 10 finishes.

6 - 9 gets interrupted by (higher-priority) syntax job of 8.

7 - 7 types a line of syntax (bumping 9 further).

8 - 8 finishes. 7 get control.

9 - 7 finishes. 9 resumes where it was interrupted.

10- 10 requests LIST. 6 requests KILL.

11- 9's RUN job gets timed-out. i.e. total time that it was on top-of-queue (frames 5,9,10) was 1 second. 9 gets reassigned a PLEV of 4 and placed on bottom of queue. 10 receives control.

12- 3 requests syntax, interrupting 10 and seizing control.

13- 2 requests syntax, and gets queued under 3.

14- 3 finishes. 2 receives control

15- 2 finishes. 10 resumes the LIST program. This fills the buffer and 10 is placed in output suspend (off the queue).

16- Notice that 10's STAT upon output suspend (i.e. LIST) is saved in PLEV (which has no relevance when a user is off the queue). 6's KILL request has acquired control, and its PLEV is changed to 0 (so that it may run to completion or self-suspension).

17- 7 requests syntax, but gets queued under 6.

18- 6 finishes.

19- 10 returns from output wait (10 characters left in buffer) and gets queued with 0 PLEV.

20- 9's user has pressed the break-key for more than 114 msec. The multiplexor changes his STAT and signals the scheduler. In the meantime, the Phones logic has detected a lost carrier on 6, changes STAT and also signals scheduler.

21- The scheduler detects 9's -1 PLEV, dequeues him, and changes his STAT to 1 while "STOP" is being typed. The scheduler also detects 6's -2 PLEV and queues him as if he requested BYE (the only difference is that his STAT remains -2).

22- 10 goes into output wait again.

23- The MPXR output routine detects the last character has been output for port 9 and STAT was 1 (i.e. the "STOP" abort message is complete) and resets STAT to the idle state. Also, 8's RUN job has encountered an INPUT statement and goes to input wait (STAT (=5) is saved). 6 receives control and the BYE routine is run with 0 PLEV.

24- 6 finishes and 8's user has responded.

25- 2 requests syntax and 10 returns from I/O suspend; but notice that both get queued under 8's RUN job due to 8's high priority upon returning from I/O suspend.

26- But 8 gets timed-out again (frames 24 +25 > 1 second) and becomes a miserable low-STAT task again.

And so on ...

QUEUE SEQUENCE EXAMPLE

1

8	syn	0

2	0	
3	0	
6	0	
7	0	
9	0	
10	0	

2

8	syn	0
10	syn	0

2	0	
3	0	
6	0	
7	0	
9	0	

3

10	syn	0

2	0	
3	0	
6	0	
7	0	
8	0	
9	0	

4

10	syn	0
9	RUN	1

2	0	
3	0	
6	0	
7	0	
8	0	

5

9	RUN	1

2	0	
3	0	
6	0	
7	0	
8	0	
10	0	

6

8	syn	0
9	RUN	1

2	0	
3	0	
6	0	
7	0	
10	0	

time →

7

8	syn	0
7	syn	0
9	RUN	1

2	0	
3	0	
6	0	
10	0	

8

7	syn	0
9	RUN	1

2	0	
3	0	
6	0	
8	0	
10	0	

9

9	RUN	1

2	0	
3	0	
6	0	
7	0	
8	0	
10	0	

10

9	RUN	1
10	LIS	1
6	KIL	2

2	0	
3	0	
7	0	
8	0	

11

10	LIS	1
6	KIL	2
9	RUN	4

2	0	
3	0	
7	0	
8	0	

12

3	syn	0
10	LIS	1
6	KIL	2
9	RUN	4

2	0	
7	0	
8	0	

QUEUE SEQUENCE EXAMPLE (Cont'd.)

13

3	syn	0
2	syn	0
10	LIS	1
6	KIL	2
9	RUN	4

7	0	
8	0	

14

2	syn	0
10	LIS	1
6	KIL	2
9	RUN	4

3	0	
7	0	
8	0	

15

10	LIS	1
6	KIL	2
9	RUN	4

2	0	
3	0	
7	0	
8	0	

16

6	KIL	0
9	RUN	4

2	0	
3	0	
7	0	
8	0	
10	3	6

17

6	KIL	0
7	syn	0
9	RUN	4

2	0	
3	0	
8	0	
10	3	6

18

7	syn	0
9	RUN	4

2	0	
3	0	
6	0	
8	0	
10	3	6

19

7	syn	0
10	LIS	0
9	RUN	4

2	0	
3	0	
6	0	
8	0	

20

7	syn	0
10	LIS	0
9	-1	4

2	0	
3	0	
6	-2	
8	0	

21

10	LIS	0
8	RUN	1
6	-2	2

2	0	
3	0	
7	0	
9	1	

22

8	RUN	1
6	-2	2

2	0	
3	0	
7	0	
9	1	
10	3	6

23

6	-2	0

2	0	
3	0	
7	0	
8	2	5
9	0	
10	3	6

24

8	RUN	0

2	0	
3	0	
6	0	
7	0	
9	0	
10	3	6

QUEUE SEQUENCE EXAMPLE (Cont'd.)

25

8	RUN	0
2	syn	0
10	LIS	0

3	0	
6	0	
7	0	
9	0	

26

2	syn	0
10	LIS	0
8	RUN	4

3	0	
6	0	
7	0	
9	0	

Consider...   Exercise

Suppose someone dials up TSB. The phones'logic begins timing him. He types in his HELLO command, but during the start bit of the carriage return, the Phones' logic times him out. What Happens?

What? No TBG?

Exercise & Lab

Assume a TSB system has a malfunctioning time-based generator, in that it does not give any interrupts. Analyze what effect this would have on the system if the bad board doesn't break the priority string; and if it does. What symptoms would you expect?

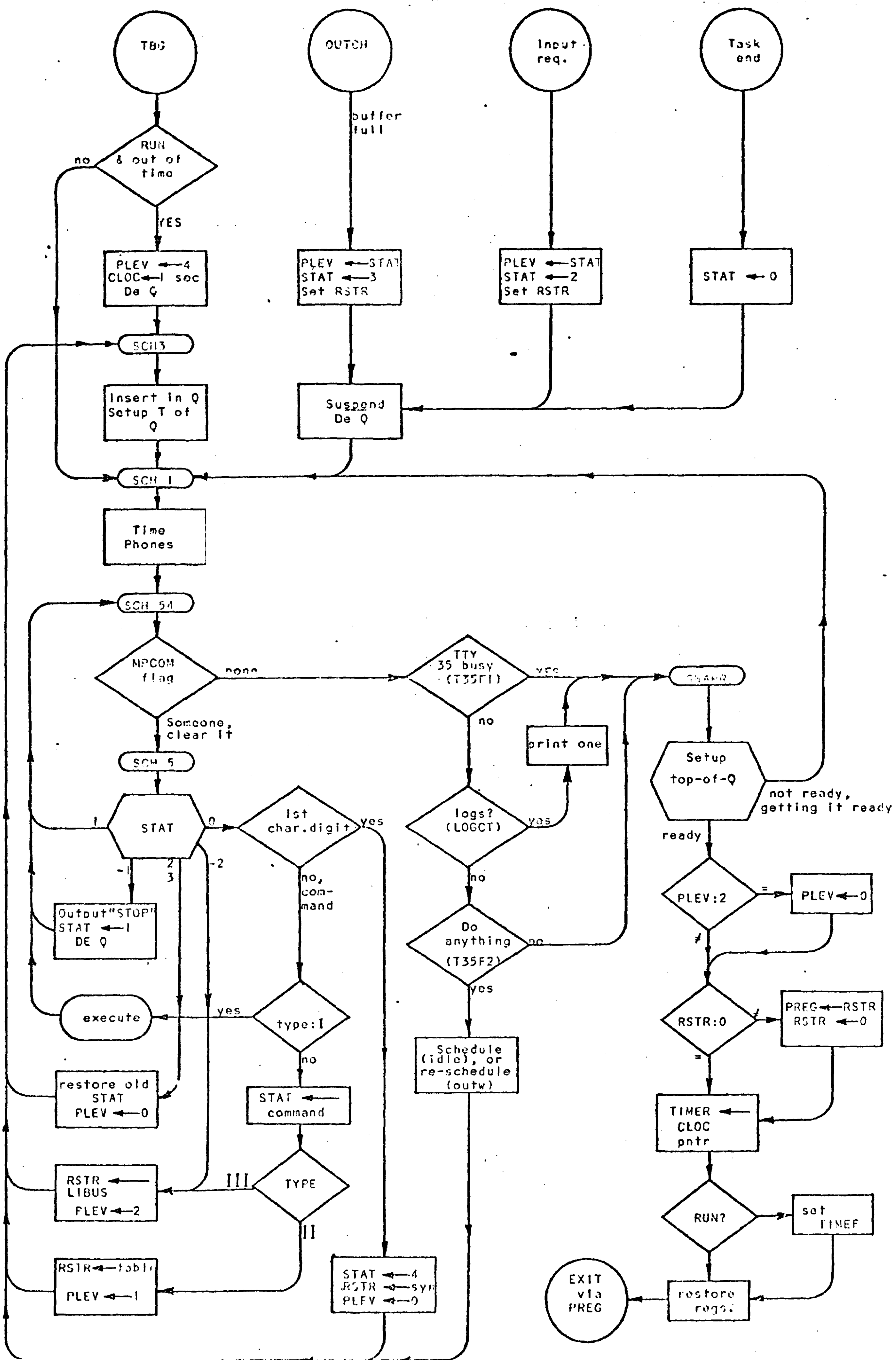
On one of the four systems, replace the TBG board with a jumper board, and prove your theories. Try different types of RUN programs, any commands you suspect would operate improperly, and (if possible) on the system with auto-disconnect (lab B), try dialing up and check the effects.

The time-slices

Exercise

The system currently provides 1 second time-slices for compute-bound RUN programs. It may be advantageous for CAI to increase this to 2 seconds. How can such a mod be made (i.e. generate the mod)?

You may wish to try it on one of the systems.


TBG
OUTCH
Input req.
Task end
RUN & out of time
no
YES
buffer full
PLEV ←4
CLOC ←1 sec
De Q
PLEV ←STAT
STAT ←3
Set RSTR
PLEV ←STAT
STAT ←2
Set RSTR
STAT ←0
SCH3
Insert in Q
Setup T of Q
Suspend
De Q
SCH 1
Time Phones
SCH 54
NPCOM flag
none
Someone, clear it
SCH 5
TTY 35 busy (T35F1)
YES
no
print one
logs? (LOGCT)
yes
no
Do anything (T35F2)
no
yes
Schedule (idle), or re-schedule (outw)
Setup top-of-Q
not ready, getting it ready
ready
STAT
1
0
-1
2
3
-2
1st char.digit
yes
no, command
Output"STOP"
STAT ←1
DE Q
execute
yes
type:I
no
restore old STAT
PLEV ←0
STAT ← command
RSTR ←
LIBUS
PLEV ←2
III
TYPE
II
RSTR ←tabl
PLEV ←1
STAT ←4
RSTR ←syn
PLEV ←0
PLEV:2
=
PLEV ←0
≠
RSTR:0
≠
PREG ←RSTR
RSTR ←0
=
TIMER ←
CLOC
pntr
RUN?
set TIMEF
EXIT via PREG
restore regs.

PHONES' LOGIC:

The Phones logic constantly monitors the input from the disconnect board to detect any "ringing" indications, i.e. status changes from 1 to Ø. If the software detects a "ringing" signal from a data set, it responds by setting the appropriate bit in PHO which when outputted to the disconnect board makes the data terminal ready and answers the data set.

Once a "ringing" has been detected, the required response time plus the present time is put into the user's TTY table ?PHON Entry and the appropriate bit in PHT corresponding to the user's TTY port number is set to indicate that this particular user is being timed. Each time the phones' logic is envoked by the scheduler, and the bit in PHT is still set, the current time (DATIM) is checked to see if it equals the user's PHON entry. If it doesn't equal, nothing is done. If it does equal, the user has run out of time and a hangup must be initiated.

If the user successfully logs on in the required time interval, the timing mechanism (PHT) in the phones logic is cleared in the HELLO Library routine and the user is allowed on the system.

The software continues to sample the status line; if a Ø is still present after the data set has been answered, the data set is transmitting data to the computer and nothing is done.

These are the ways in which the user may be disconnected:

a) carrier is lost
b) user signs off
c) user is timed out

a) When carrier is lost, the phones logic detects this because the input from the disconnect board changes from Ø to 1. When the software detects this transition, it knows it's a line dropout and initiates a 1 sec time interval via the user ?PHON TTY table Entry, sets the user's PHT bit and allows the time-out logic of phones' timing to handle the dropout disconnect.

b) When the user signs off (BYE), the BYE routine sets the user's bit in PHT and sets his ?PHON entry to a 4 second time interval and allows the time-out logic of phones' timing to handle the sign off disconnect. The reason for the 4 second interval is so as to allow the standard log off message to be outputted the user's TTY.

c) The user is timed out when the time in his?PHON TTY table entry is equal to or less than the time contained in DATIM. When this occurs, the user's status is set to a -2 (system disconnect) and the appropriate TTY bit is set in MPCOM to inform the scheduler of the user's status. When the scheduler detects the system disconnect status for that user, it schedules the user for the BYE Library routine.

When the BYE routine is called in to handle this user, the user may be in two states.

1) he has already typed BYE (signed off), 2) he has not signed off). If he has already signed off, his? I.D. entry in the TTY table had been cleared from his previous BYE processing. So that, when BYE is entered again for him, the I.D. contained in the ?I.D. TTY table entry is zero which signifies the user received his log off message and it is O.K. now to disconnect him from the system. The disconnect is accomplished by clearing his timing bit (PHT) and setting his PHO bit to one which is the hang up signal to the disconnect board.

If the user had not signed off and BYE was scheduled by the Scheduler it was because the user had timed out either for I second (line drop out) normal response time for a log on had elasped. Therefore, the user was either in core or not in core. If he wasn't in core he is disconnected then. If he was in core (i.e., the case of a line drop out) 4 seconds must elaspe before he is disconnected so as to allow the standard output message to be completed.

PHL

Each bit corresponds to 2^N TTY number.

Set at load time to TTY status of disconnect board. For purposes of simplication, assume nothing was happening with the auto disconnect board at sleep time. Therefore, when the system was awakened, the TTY's are in their quiescent state, i.e. PHL is set to all ones. PHL is updated, such that, it will reflect the most recent input from disconnect board.

PHN

Each bit corresponds to 2^N TTY number. Each time the phones' logic is envoked by the schedule, PHN is updated so as to reflect the current input from the disconnect board.

PHO

Each bit corresponds to 2^N TTY number. When the phones' logic finishes with all the TTY's that need update in relation to the disconnect board, it outputs the word PHO. This word is the output communication to inform the board of any status change. The quiescent state for PHO is all 1's.

PHONES'BOARD LEVELS

Input = Ø implies carrier or ring
Input = 1 implies neither
Output = Ø answer the phone
Output = 1 hang phone up

PHT

Each bit corresponds to 2^N TTY number set to the user's port which is currently being timed. The user can be timed for three reasons:

1. Normal response timing to log on system
2. Timing because carrier was lost
3. 4 Second time out due to time required to output standard sign off message to user's TTY

SELECTED LIBRARY ROUTINES

HELLO

1. LOGS OFF OLD USER IF HE HASN'T DONE SO.
    - a) REINITIALIZE NEW USER'S PORT FUSS TABLE ENTRY TO ZERO
    - b) OUTPUT LOG OFF MESSAGE TO SYSTEM TTY
    - c) SETUP PHONES TIMING FOR NEW USER

2. READ IN IDT TABLE AND CHECKS FOR VALID NEW ID.

3. CHECK FOR VALID PASSWORD IF ID CORRECT.

4. CHECK THAT THE TIME USED TO DATE IS LESS THAN THE ALLOWED.

5. USER HAS SUCCESSFULLY LOGGED ON, THEREFORE DISCONTINUE ANY PHONES TIMING BY SETTING THE BIT IN PHT CORRESPONDING TO THE NEW USER'S PORT NUMBER.

6. RUN ANY "HELLO" PROGRAM WHICH IS IN PUBLIC LIBRARY, OTHERWISE PRINT "READY" MESSAGE TO ALLOW USER ACCESS TO SYSTEM.

7. IF "HELLO" PROGRAM PRESENT, TRANSFER CONTROL TO BASIC.

HELLO EXERCISE:

ATTACHED TO THIS EXERCISE IS A LISTING OF THE PUBLIC LIBRARY ROUTINE $ HELLO WHICH IS RUN EACH TIME A USER SUCCESSFULLY LOGS ON. ACCORDING TO THE HELLO FUNCTION OF THE TSB SYSTEM, THE $ HELLO PROGRAM IS WRITTEN INTO THE USER'S SWAP AREA AND EXECUTED, HOWEVER THE PROGRAM NEVER RESIDES IN THE USER'S SWAP AREA AFTER IT IS EXECUTED, WHY?

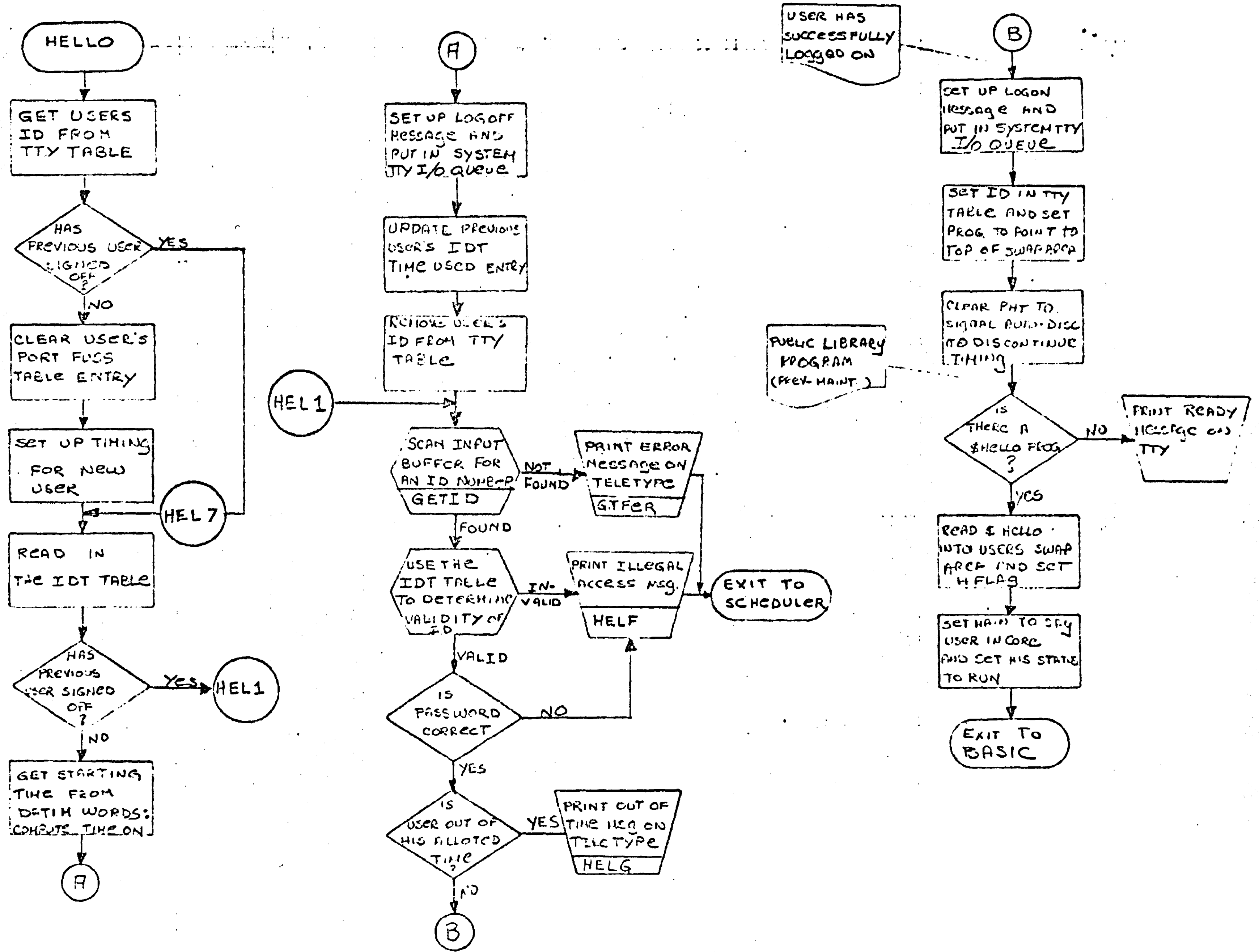
```
STOP


GET-$HELLO



LIST
HELLO

8  PRINT
10  PRINT "     HP IN-HOUSE SERVICE BUREAU"
15  PRINT "          WELCOMES YOU TO"
20  PRINT "             SYSTEM/1"
30  PRINT
31  PRINT "       PREVENTIVE MAINTENANCE"
40  PRINT "     DAILY FROM 0800 TO 0815 PDT"
50  PRINT
55  PRINT "** DENOTES SPECIAL MESSAGES"
56  PRINT
65  PRINT "    Y   D  A ER"
70  END
```


HELLO
GET USERS ID FROM TTY TABLE
HAS PREVIOUS USER SIGNED OFF ?
YES
NO
CLEAR USER'S PORT FUSS TABLE ENTRY
SET UP TIMING FOR NEW USER
HEL 7
READ IN THE IDT TABLE
HAS PREVIOUS USER SIGNED OFF ?
YES
HEL1
NO
GET STARTING TIME FROM DATIM WORDS: COMPUTE TIME ON
A
A
SET UP LOGOFF MESSAGE AND PUT IN SYSTEM TTY I/O QUEUE
UPDATE PREVIOUS USER'S IDT TIME USED ENTRY
REMOVE USER'S ID FROM TTY TABLE
HEL1
SCAN INPUT BUFFER FOR AN ID NUMBER
GETID
NOT FOUND
PRINT ERROR MESSAGE ON TELETYPE
GTFER
FOUND
USE THE IDT TABLE TO DETERMINE VALIDITY OF ID
IN-VALID
PRINT ILLEGAL ACCESS MSG.
HELF
EXIT TO SCHEDULER
VALID
IS PASSWORD CORRECT
NO
YES
IS USER OUT OF HIS ALLOTED TIME ?
YES
PRINT OUT OF TIME MSG ON TELETYPE
HELG
NO
B
USER HAS SUCCESSFULLY LOGGED ON
B
SET UP LOGON MESSAGE AND PUT IN SYSTEM TTY I/O QUEUE
SET ID IN TTY TABLE AND SET PROG. TO POINT TO TOP OF SWAP AREA
CLEAR PHT TO SIGNAL AUTO-DISC TO DISCONTINUE TIMING
PUBLIC LIBRARY PROGRAM (PREV-MAINT.)
IS THERE A $HELLO PROG ?
NO
PRINT READY MESSAGE ON TTY
YES
READ $ HELLO INTO USERS SWAP AREA AND SET H FLAG
SET MAIN TO SAY USER IN CORE AND SET HIS STATE TO RUN
EXIT TO BASIC

HELLO-BYE EXERCISE:

An interesting occurrence happens when a user logs on and types echo-off and then logs on again without typing BYE. For this exercise, go up to a terminal and try it. What happened, why, and how do you fix it?

GET

1. TRANSLATE NAME OF PROGRAM FROM USER'S INPUT. IF PRECEEDED BY $ SET UP FOR A000 SEARCH; OTHERWISE SET FOR SEARCHING ON USER'S ID.

2. PERFORM DIRECTORY SEARCH. PRINT ERROR IF NOT FOUND.

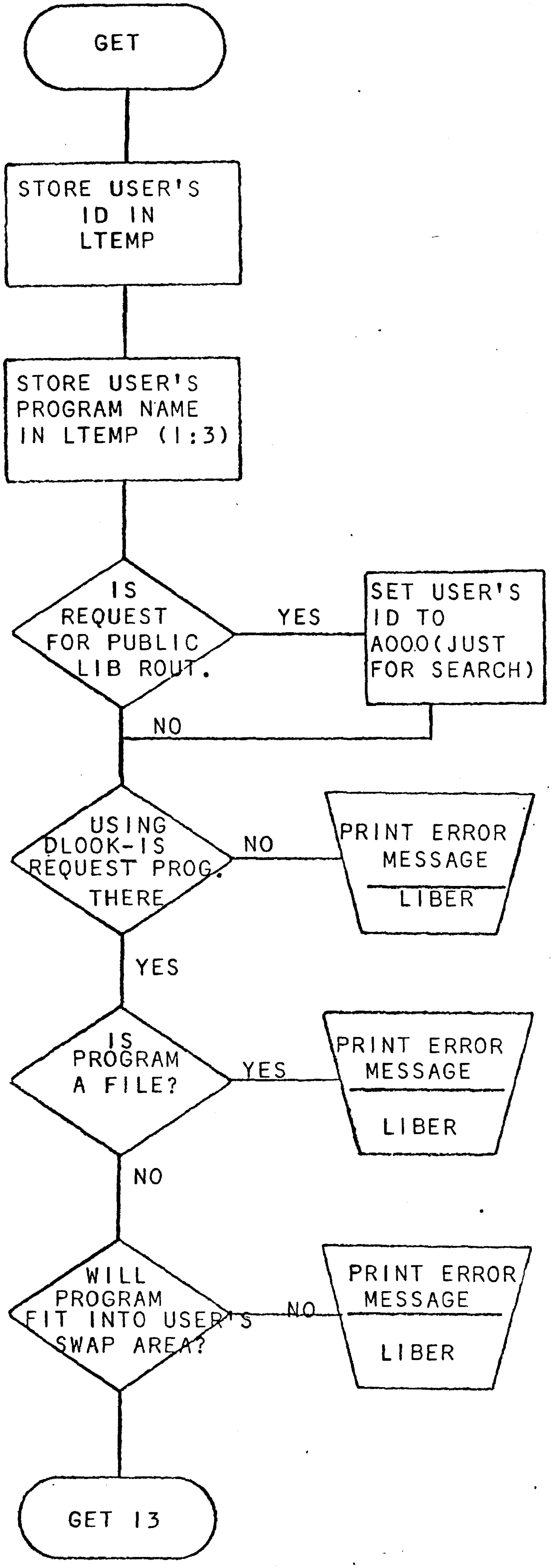
3. ERROR IF ENTRY FOUND IS FILE (BIT 15 OF WORD 2 OF ENTRY IS 1).

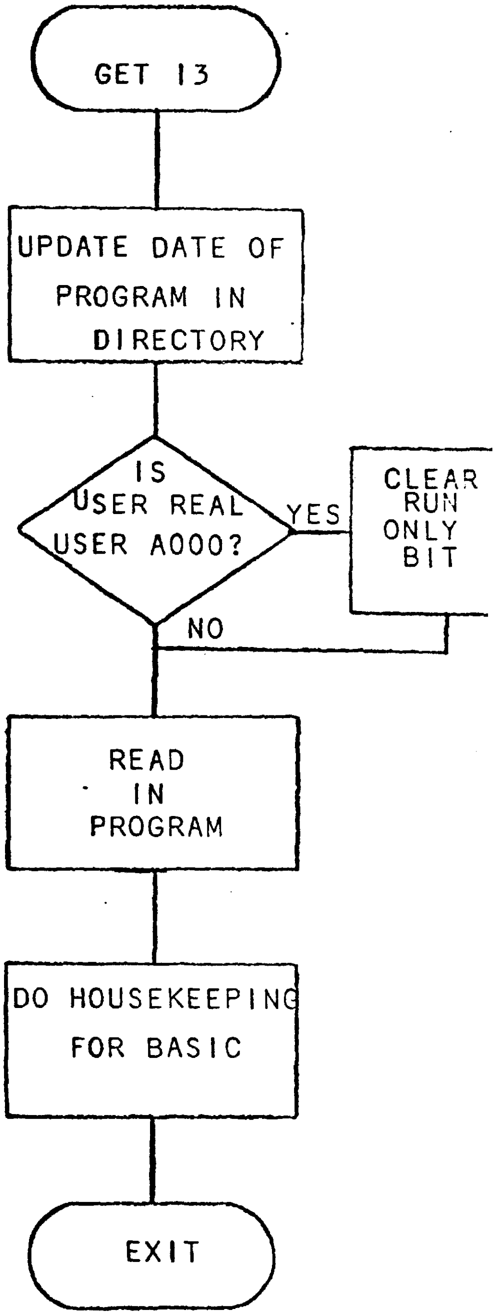
4. CHECK THAT THE PROGRAM WILL FIT INTO THE USER AREA. THIS IS NECESSARY IN CASE A PROGRAM WHICH WAS SAVED UNDER AN OLD VERSION OF THE SYSTEM CAN NO LONGER FIT WITH THE CURRENT VERSION.

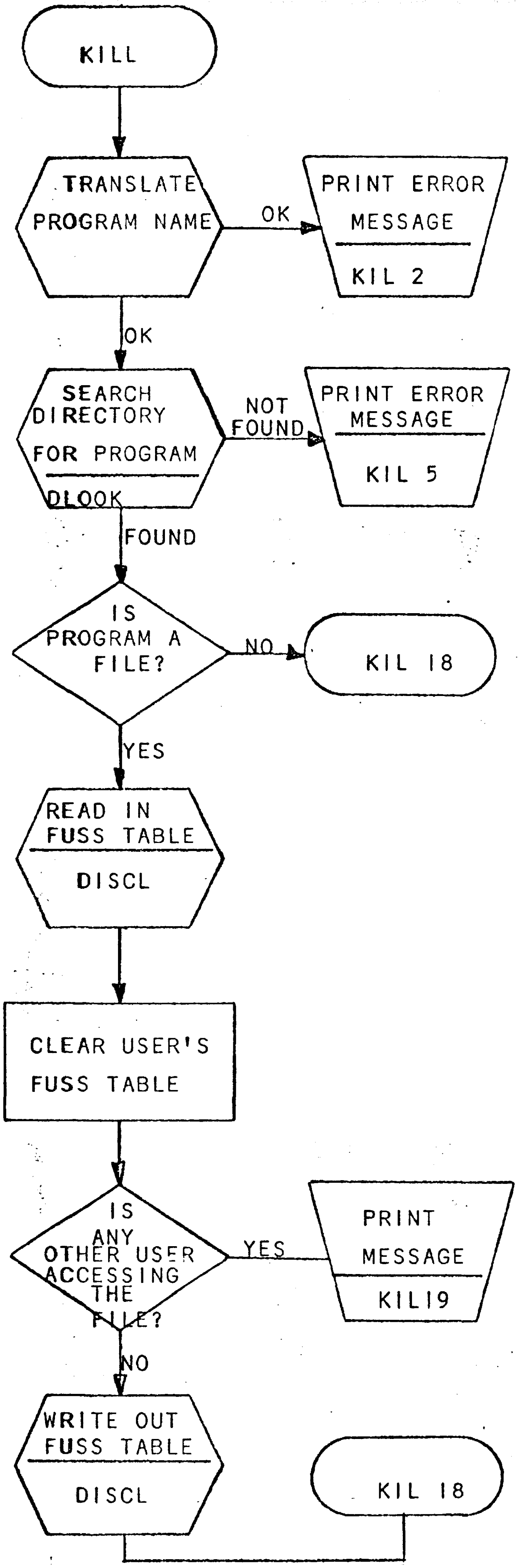
5. UPDATE THE DATE OF THE USER'S PROGRAM (PURGE).

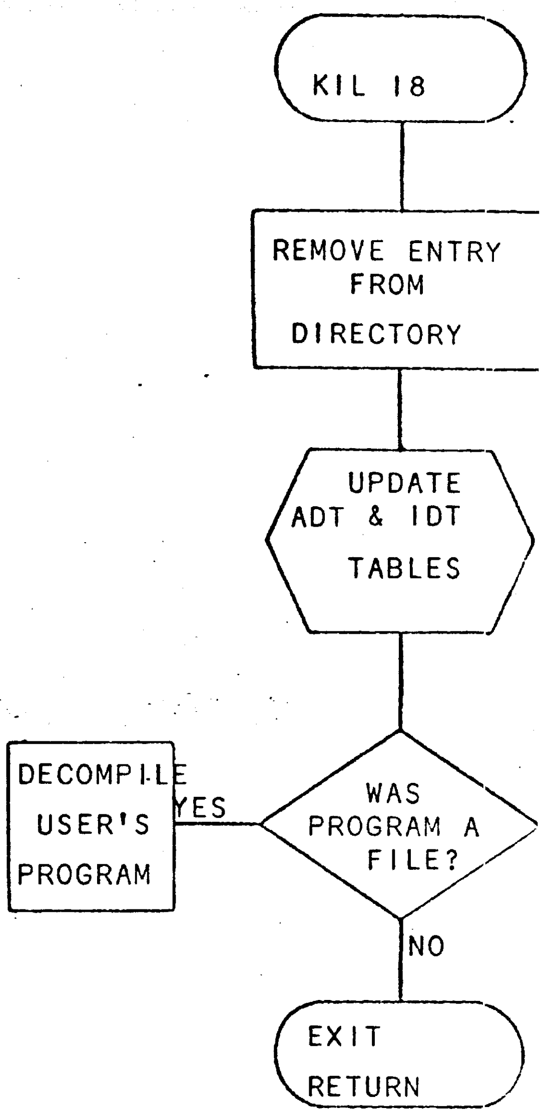
6. SET RUN-ONLY BIT IF RUN-ONLY PROGRAM UNLESS USER IS A000.

7. READ IN PROGRAM AND UP DATE CERTAIN FLAGS FOR BASIC, THEN EXIT.


GET
STORE USER'S ID IN LTEMP
STORE USER'S PROGRAM NAME IN LTEMP (1:3)
IS REQUEST FOR PUBLIC LIB ROUT.
YES
SET USER'S ID TO A000(JUST FOR SEARCH)
NO
USING DLOOK-IS REQUEST PROG. THERE
NO
PRINT ERROR MESSAGE
LIBER
YES
IS PROGRAM A FILE?
YES
PRINT ERROR MESSAGE
LIBER
NO
WILL PROGRAM FIT INTO USER'S SWAP AREA?
NO
PRINT ERROR MESSAGE
LIBER
GET 13


GET 13
UPDATE DATE OF PROGRAM IN DIRECTORY
IS USER REAL USER A000?
YES
CLEAR RUN ONLY BIT
NO
READ IN PROGRAM
DO HOUSEKEEPING FOR BASIC
EXIT


KILL
TRANSLATE PROGRAM NAME
OK
PRINT ERROR MESSAGE
KIL 2
OK
SEARCH DIRECTORY FOR PROGRAM
DLOOK
NOT FOUND
PRINT ERROR MESSAGE
KIL 5
FOUND
IS PROGRAM A FILE?
NO
KIL 18
YES
READ IN FUSS TABLE
DISCL
CLEAR USER'S FUSS TABLE
IS ANY OTHER USER ACCESSING THE FILE?
YES
PRINT MESSAGE
KIL19
NO
WRITE OUT FUSS TABLE
DISCL
KIL 18


KIL 18
REMOVE ENTRY FROM DIRECTORY
UPDATE ADT & IDT TABLES
WAS PROGRAM A FILE?
YES
DECOMPILE USER'S PROGRAM
NO
EXIT RETURN

KILL

1. TRANSLATE THE PROGRAM OR FILE NAME AND PERFORM A DIRECTORY SEARCH. FAIL TO ERROR ROUTINE IF ILLEGAL NAME OR THE SEARCH FAILS.

2. IF THE ENTRY TO BE FILLED IS A FILE, SEARCH THE FUSS TABLE TO SEE IF ANY OTHER USER HAS ACCESS TO THE FILE. IF SO, PRINT A MESSAGE AND TERMINATE. IF NOT, CLEAR THE USER'S SECTION OF FUSS.

3. DELETE THE ENTRY FROM THE DIRECTORY AND ADJUST DIREC.

4. SUBTRACT THE PROGRAM LENGTH FROM THE USER'S IDT ENTRY (SYSTEM ADMINISTRATION).

5. UPDATE ADT TO REFLECT ADDED DISC SPACE.

6. DECOMPILE USER IF FILE WAS KILLED. THIS GUARANTEES THAT ANY OLD REFERENCES TO FILE WILL DISAPPEAR.

ECHO:

The ECHO command is used to control the computer echo of teletype input. Echoing is determined by the user's bit in the word PLEX. Bit = 0 implies no echo, I implies echo. The user will want echoing if and only if his teletype is full duplex. The command format is:

```
ECHO-ON for full duplex
ECHO-OFF for half duplex
```

OPEN

1. TRANSLATE AND CHECK THE FILE NAME AND LENGTH.

2. CHECK THE IDT AND ADT TO SEE IF
   a) THE USER HAS ENOUGH DISC ALLOCATED TO HIM TO SATISFY THE COMMAND. (ADMINISTRATIVE PURPOSES)
   b) THERE IS AN AREA ON THE DISC WHICH IS LARGE ENOUGH TO ACCOMMODATE THE FILE. IF THERE IS AN AREA WAIT UNTIL ROOM IN DIRECTORY IS FOUND.

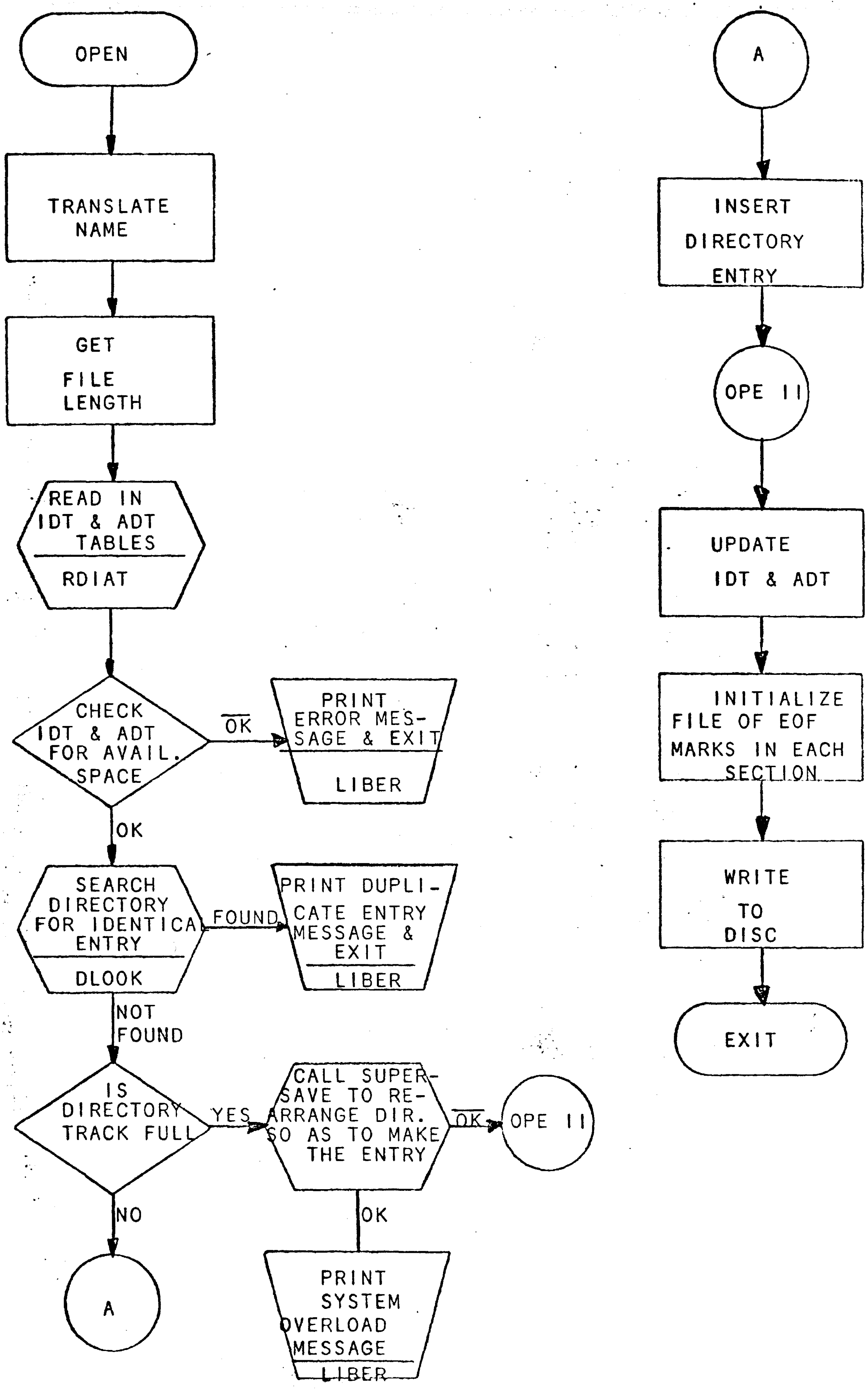
3. PERFORM A DIRECTORY SEARCH ON THE FILE NAME. IF FOUND, THIS IS A DUPLICATE ENTRY, SO TERMINATE.

4. IF DIRECTORY TRACK IS NOT FULL, INSERT NEW ENTRY.

5. IF DIRECTORY TRACK IS FULL, CALL IN SUPERSAVE TO RESTRUCTURE THE DIRECTORY AND INSERT THE ENTRY.

6. UPDATE THE IDT AND ADT APPROPRIATELY.

7. INITIALIZE THE FILE SO THAT A -1 (END-OF-FILE) IS AT THE BEGINNING OF EVERY SECTOR. WRITE THE FILE TO THE DISC AND THEN TERMINATE.


OPEN
TRANSLATE NAME
GET FILE LENGTH
READ IN IDT & ADT TABLES
RDIAT
CHECK IDT & ADT FOR AVAIL. SPACE
$\overline{OK}$
PRINT ERROR MESSAGE & EXIT
LIBER
OK
SEARCH DIRECTORY FOR IDENTICAL ENTRY
DLOOK
FOUND
PRINT DUPLICATE ENTRY MESSAGE & EXIT
LIBER
NOT FOUND
IS DIRECTORY TRACK FULL
YES
CALL SUPERSAVE TO REARRANGE DIR. SO AS TO MAKE THE ENTRY
$\overline{OK}$
OPE II
OK
PRINT SYSTEM OVERLOAD MESSAGE
LIBER
NO
A
A
INSERT DIRECTORY ENTRY
OPE II
UPDATE IDT & ADT
INITIALIZE FILE OF EOF MARKS IN EACH SECTION
WRITE TO DISC
EXIT

BYE:

1. SET THE USER'S PLEX BIT TO FULL DUPLEX.

2. IF USER ID = 0, THEN DISCONNECT PHONES TIMING AND TERMINATE BYE.

3. IF USER IN CORE, READ IN FUSS TABLE AND CLEAR USER'S ENTRY IN TABLE.

4. READ IN IDT TABLE AND COMPUTE HIS TIME USED TO UPDATE HIS IDT TABLE ENTRY.

5. CREATE A LOG-OFF MESSAGE FOR USER AND PUT IN SYSTEM TELETYPE I/O QUEUE.

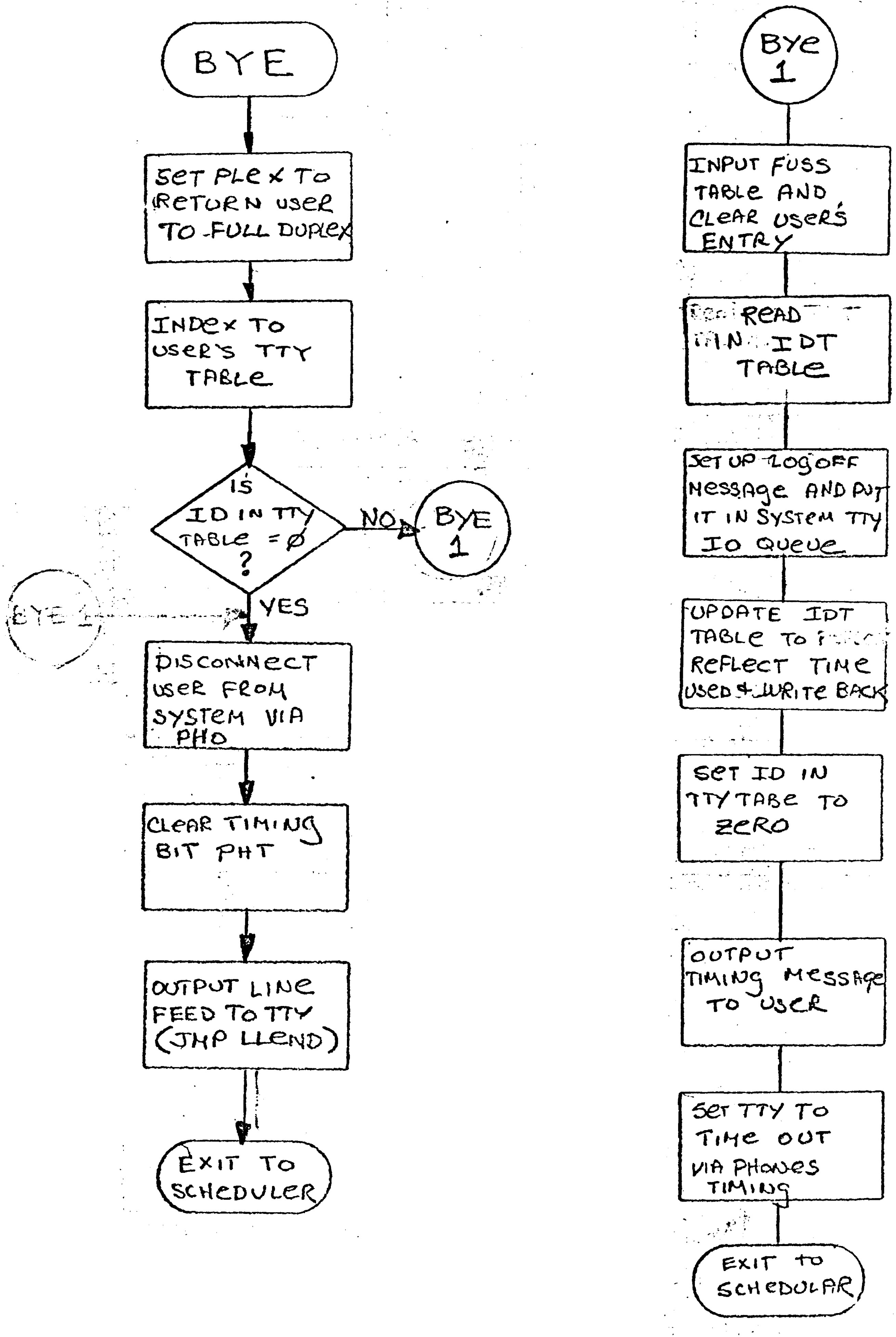
6. REMOVE USER'S ID FROM TTY TABLE.

7. CALCULATE TIME USED AND OUTPUT TO USER'S TTY.

8. ENABLE AUTO DISCONNECT VIA PHT TO DISCONNECT USER FROM SYSTEM.

BYE EXERCISE

YOU HAVE IMPLEMENTED A MOD TO MAKE A PORT OR A PARTICULAR SEQUENCE OF PORTS HALF-DUPLEX IN THE MULTIPLEXOR DRIVER. CAN A SIMILAR MOD BE IMPLEMENTED IN THE BYE ROUTINE? STATE REASON(S) WHY IT CAN OR CANNOT.


BYE
SET PLEX TO RETURN USER TO FULL DUPLEX
INDEX TO USER'S TTY TABLE
IS ID IN TTY TABLE = Ø ?
NO
BYE 1
YES
BYE 2
DISCONNECT USER FROM SYSTEM VIA PHO
CLEAR TIMING BIT PHT
OUTPUT LINE FEED TO TTY (JMP LLEND)
EXIT TO SCHEDULER
BYE 1
INPUT FUSS TABLE AND CLEAR USER'S ENTRY
READ IN IDT TABLE
SET UP LOGOFF MESSAGE AND PUT IT IN SYSTEM TTY IO QUEUE
UPDATE IDT TABLE TO REFLECT TIME USED & WRITE BACK
SET ID IN TTY TABE TO ZERO
OUTPUT TIMING MESSAGE TO USER
SET TTY TO TIME OUT VIA PHONES TIMING
EXIT TO SCHEDULAR

SAVE

1. TEST IF PROGRAM HAS NAME. FAIL IF NONE.

2. TEST IF THERE IS A PROGRAM TO SAVE. FAIL IF NONE.

3. IF THE USER'S PROGRAM IS IN COMPILED FORM (CFLAG bit=1), CALL DCMPL TO PUT IT INTO THE FORM IN WHICH WE WILL SAVE IT.

4. READ IN IDT TO SEE IF THE USER HAS SUFFICIENT DISC SPACE ALLOCATED TO SAVE THE PROGRAM. FAIL IF NOT ENOUGH ROOM.

5. READ IN ADT TO CHECK TO FIND FIRST ENTRY LARGE ENOUGH TO HOLD THE PROGRAM. FAIL IF ONE IS NOT FOUND.

6. PERFORM A DIRECTORY SEARCH ON THE PROGRAM TO BE SAVED. FAIL IS SUCH AN ENTRY NAME ALREADY EXISTS.

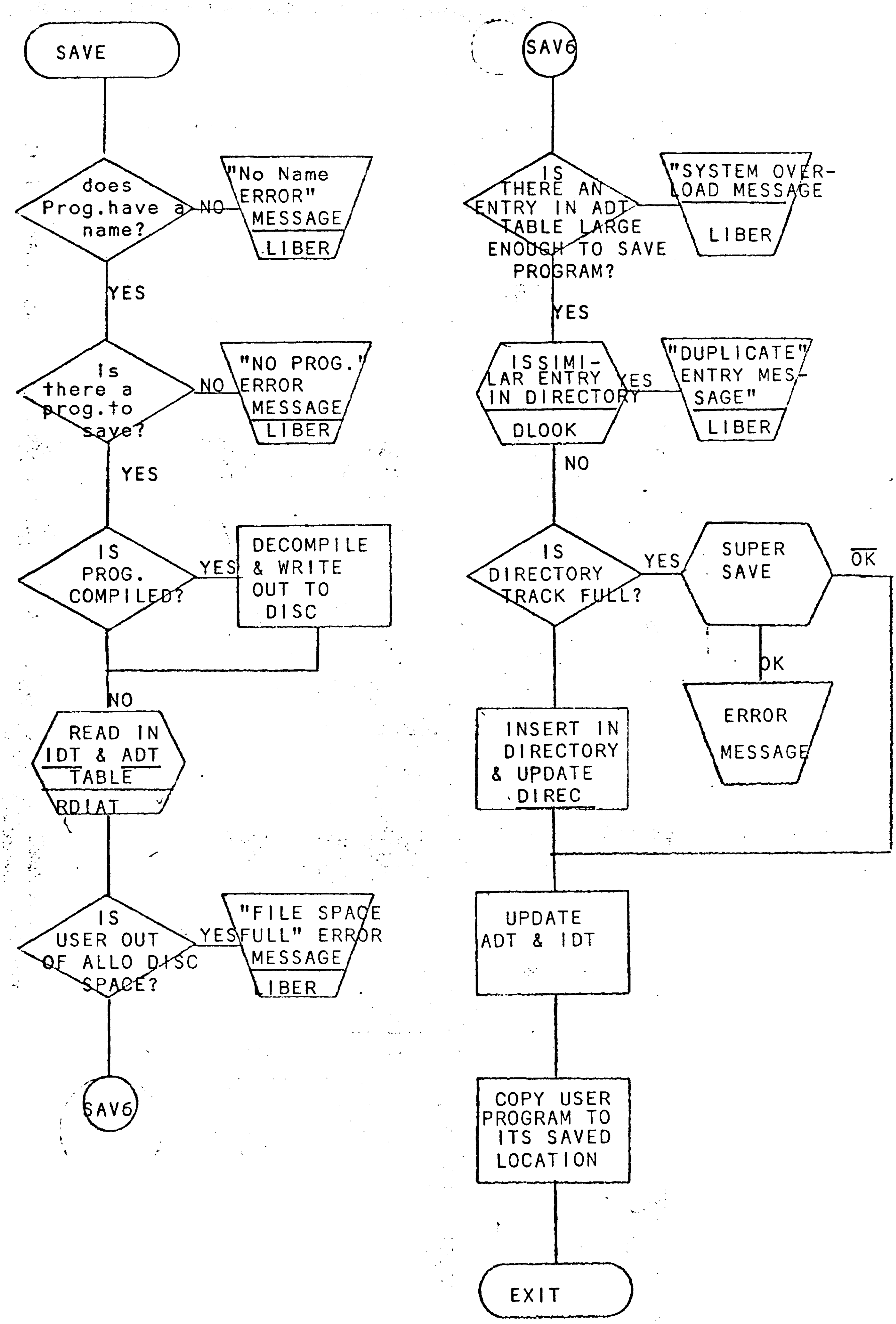
7. IF THE DIRECTORY TRACK IS FULL, THE SUPERSAVE LIBRARY ROUTINE IS CALLED TO ATTEMPT TO REALLOCATE THE DIRECTORY. IF IT'S SUCCESSFUL IN ITS REALLOCATION ALGORITHM, THE NEW DIRECTORY ENTRY IS MADE. IF SUPERSAVE FAILS IN ITS ATTEMPT, A SYSTEM OVERLOAD MESSAGE WILL APPEAR ON THE USER'S TTY.

8. INSERT A NEW DIRECTORY ENTRY INTO THE DIRECTORY IF THERE IS NO NEED FOR SUPERSAVE.

9. UPDATE THE IDT TO REFLECT THE NEW AMOUNT OF DISC USED FOR THAT USER.

10. UPDATE THE ADT TO REFLECT NEW LENGTH BECAUSE OF DISC ALLOCATION FOR USER'S PROGRAM.

11. COPY THE USER'S PROGRAM TO ITS LIBRARY AREA.


SAVE
does Prog.have a name?
NO
"No Name ERROR" MESSAGE
LIBER
YES
Is there a prog.to save?
NO
"NO PROG." ERROR MESSAGE
LIBER
YES
IS PROG. COMPILED?
YES
DECOMPILE & WRITE OUT TO DISC
NO
READ IN IDT & ADT TABLE
RDIAT
IS USER OUT OF ALLO DISC SPACE?
YES
"FILE SPACE FULL" ERROR MESSAGE
LIBER
SAV6
SAV6
IS THERE AN ENTRY IN ADT TABLE LARGE ENOUGH TO SAVE PROGRAM?
"SYSTEM OVER-LOAD MESSAGE
LIBER
YES
ISSIMI-LAR ENTRY IN DIRECTORY
DLOOK
YES
"DUPLICATE" ENTRY MES-SAGE"
LIBER
NO
IS DIRECTORY TRACK FULL?
YES
SUPER SAVE
OK
OK
ERROR MESSAGE
INSERT IN DIRECTORY & UPDATE DIREC
UPDATE ADT & IDT
COPY USER PROGRAM TO ITS SAVED LOCATION
EXIT

SUPERSAVE:

SUPERSAVE (SAVE OVERLAY ROUTINE) IS CALLED BY THE SAVE AND OPEN ROUTINES WHENEVER THEY WANT TO MAKE A DIRECTORY ENTRY ON A TRACK THAT IS ALREADY FULL.

SUPERSAVE ATTEMPTS TO REDISTRIBUTE THE DIRECTORY TRACKS SO THAT THEY WILL BE AS EQUAL IN LENGTH AS POSSIBLE. THIS WILL GENERALLY PREVENT IT FROM BEING CALLED VERY FREQUENTLY. THE OPERATION IS AS FOLLOWS:

1. SCAN THROUGH DIREC AND DETERMINE THE TOTAL LENGTH OF ALL DIRECTORY TRACKS, AND ADD 8 FOR THE NEW ENTRY. IF ALL DIRECTORY TRACKS ARE FULL, EXIT THROUGH FAILURE LOCATION.

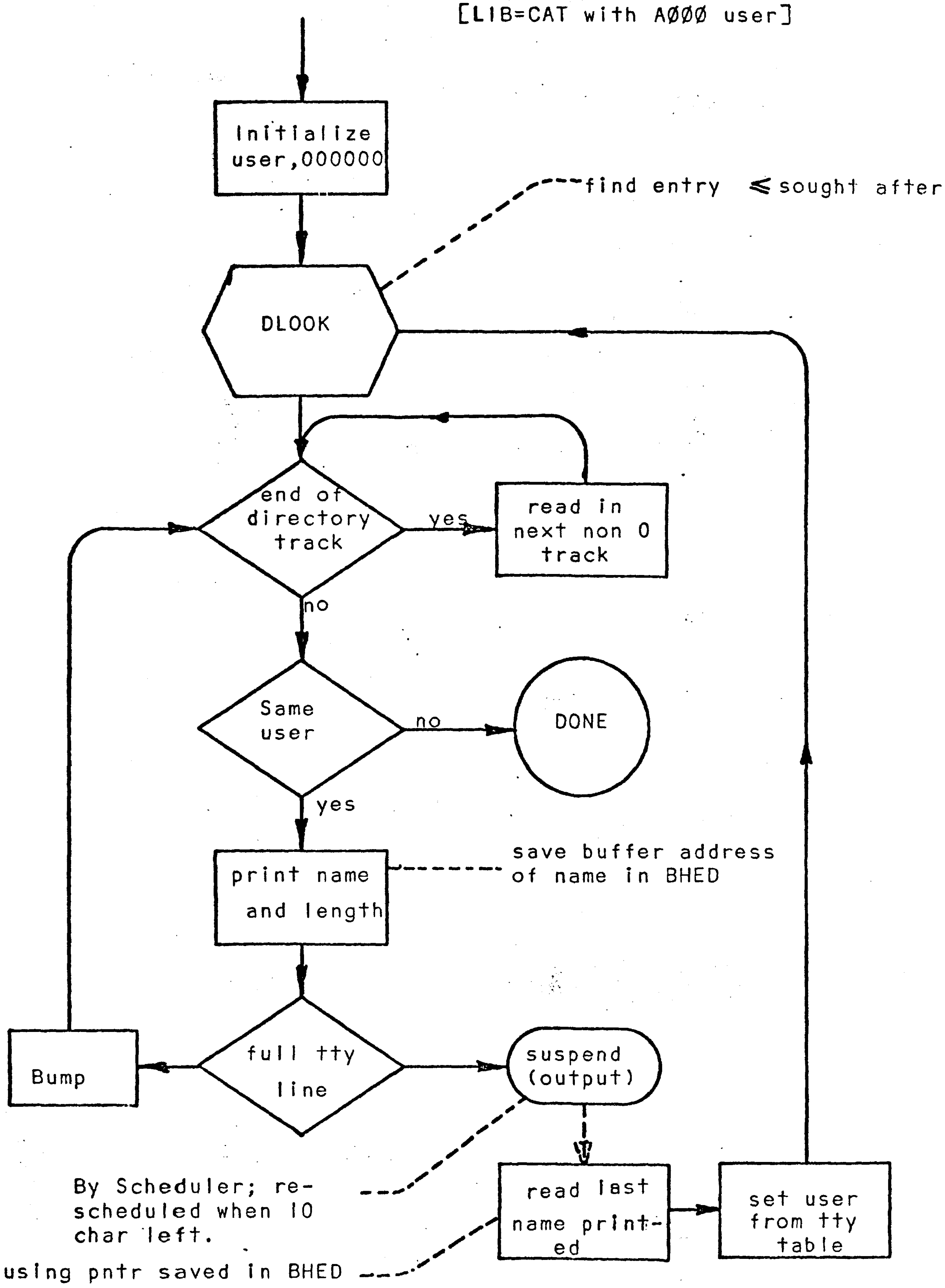
2. DIVIDE TOTAL DIRECTORY LENGTH BY NUMBER OF AVAILABLE DISC TRACKS TO DETERMINE THEIR NEW INDIVIDUAL LENGTHS.

3. NOW REDISTRIBUTE THE DIRECTORY TRACKS. THE BASIC IDEA OF THE ALGORITHM IS TO FILL THE SWAP AREA WITH AS MUCH OF THE DIRECTORY INFORMATION AS WE CAN, READING FROM THE BEGINNING AND THEN TO WRITE OUT AS MUCH AS WE CAN, ALWAYS MAKING SURE THAT WHEN WRITING, WE DON'T OVERLAY ANY PORTION THAT HASN'T BEEN READ YET.

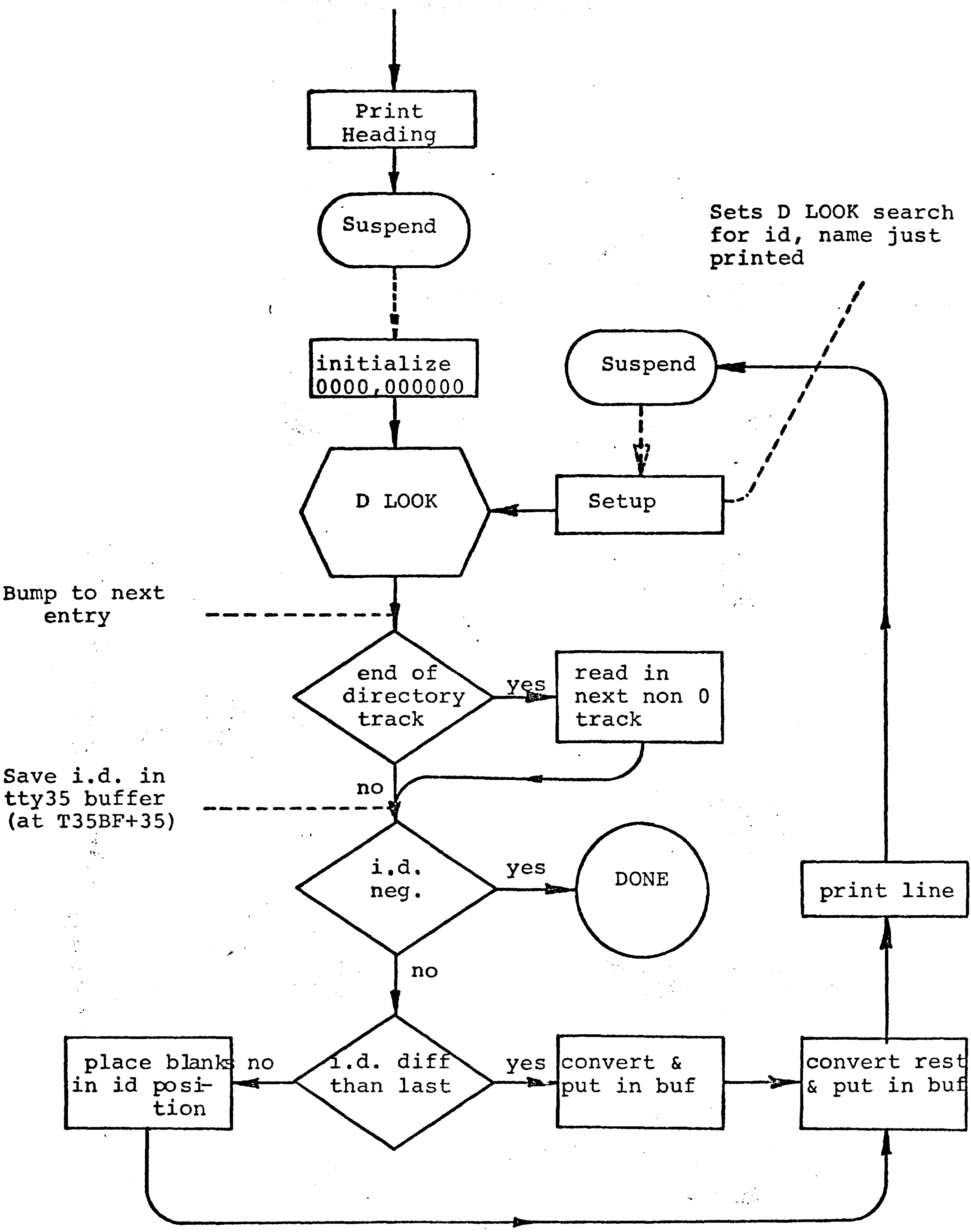
SLEEPING AND LOADING

CATALOG

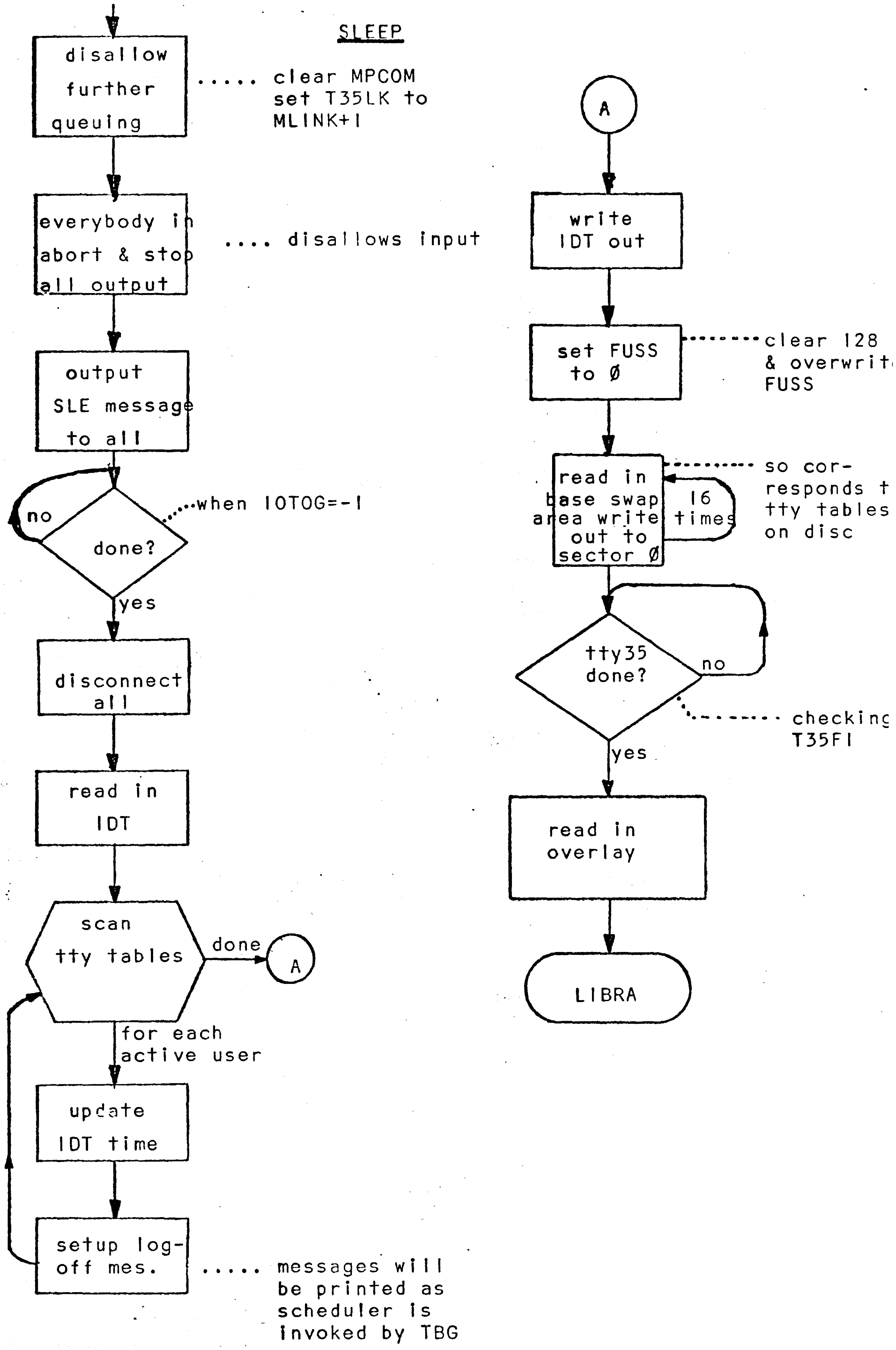
[LIB=CAT with AØØØ user]



DIRECTORY

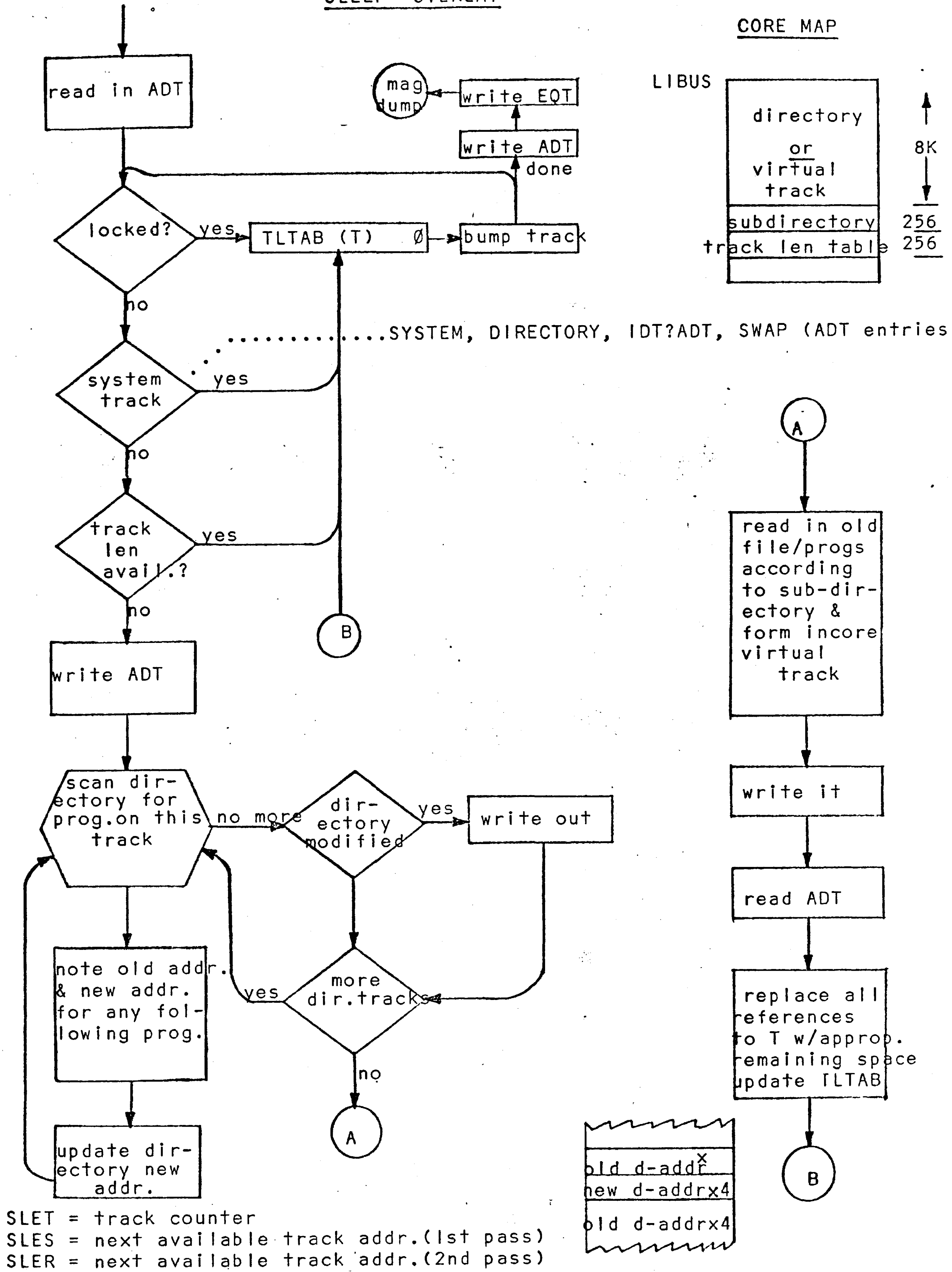

Print Heading
Suspend
initialize 0000,000000
D LOOK
Suspend
Setup
Sets D LOOK search for id, name just printed
Bump to next entry
end of directory track
yes
read in next non 0 track
no
Save i.d. in tty35 buffer (at T35BF+35)
i.d. neg.
yes
DONE
no
print line
place blanks in id position
no
i.d. diff than last
yes
convert & put in buf
convert rest & put in buf

SLEEP



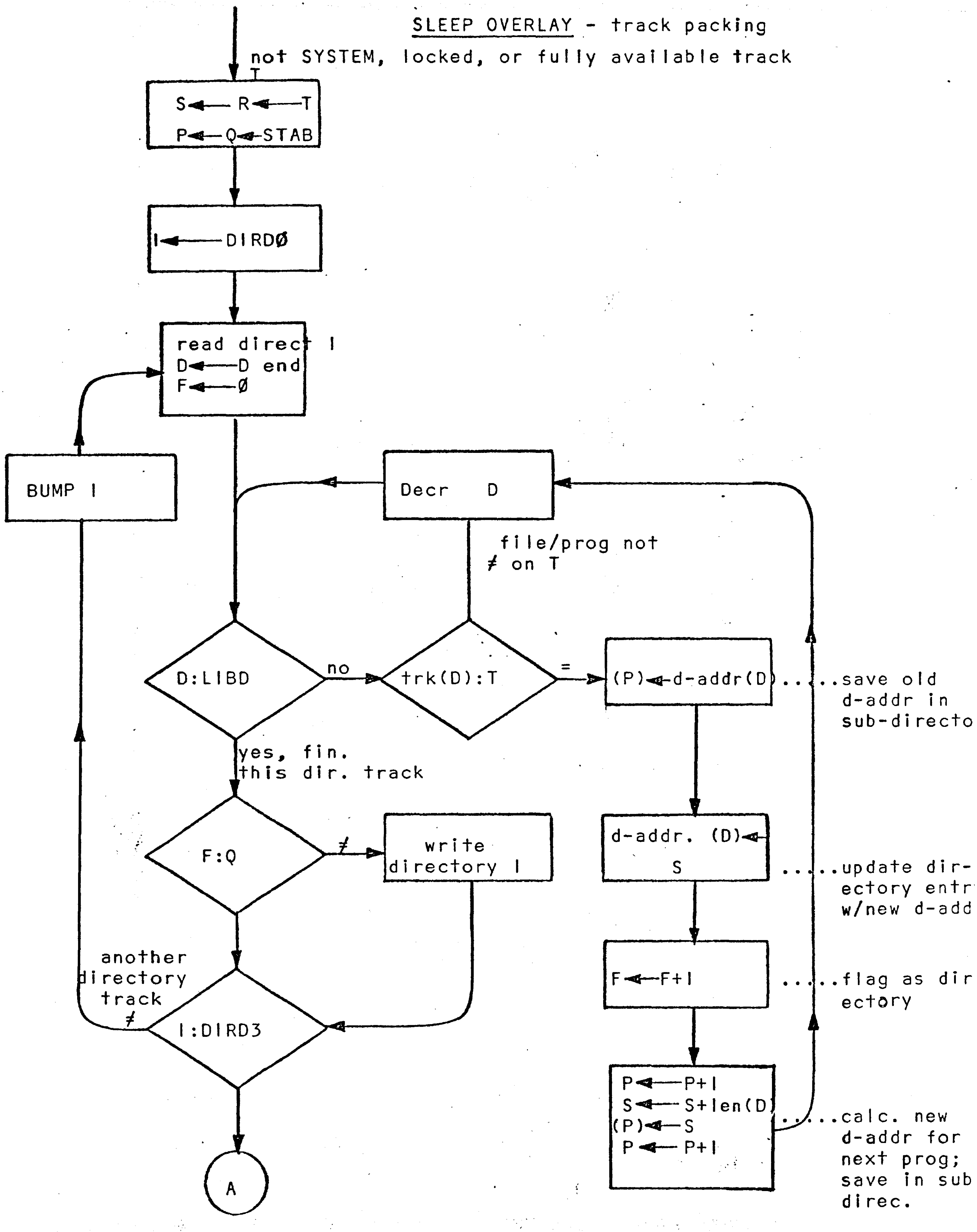
disallow further queuing
..... clear MPCOM set T35LK to MLINK+1
everybody in abort & stop all output
.... disallows input
output SLE message to all
done?
no
when IOTOG=-1
yes
disconnect all
read in IDT
scan tty tables
done
A
for each active user
update IDT time
setup log-off mes.
..... messages will be printed as scheduler is invoked by TBG
A
write IDT out
set FUSS to Ø
clear 128 & overwrit FUSS
read in base swap area write out to sector Ø
16 times
so cor-responds t tty tables on disc
tty35 done?
no
checking T35FI
yes
read in overlay
LIBRA

SLEEP OVERLAY


read in ADT
locked?
yes
TLTAB (T) Ø
bump track
done
write ADT
write EOT
mag dump
no
system track
yes
....SYSTEM, DIRECTORY, IDT?ADT, SWAP (ADT entries
no
track len avail.?
yes
no
B
write ADT
scan directory for prog.on this track
no more
directory modified
yes
write out
note old addr. & new addr. for any following prog.
update directory new addr.
more dir.tracks
yes
no
A
CORE MAP
LIBUS
directory or virtual track
8K
subdirectory
256
track len table
256
A
read in old file/progs according to sub-directory & form incore virtual track
write it
read ADT
replace all references to T w/approp. remaining space update TLTAB
B
old d-addr x
new d-addrx4
old d-addrx4

SLET = track counter
SLES = next available track addr.(1st pass)
SLER = next available track addr.(2nd pass)
SLEP = 1st pass sub-directory pointer
SLEQ = 2nd pass sub-directory pointer

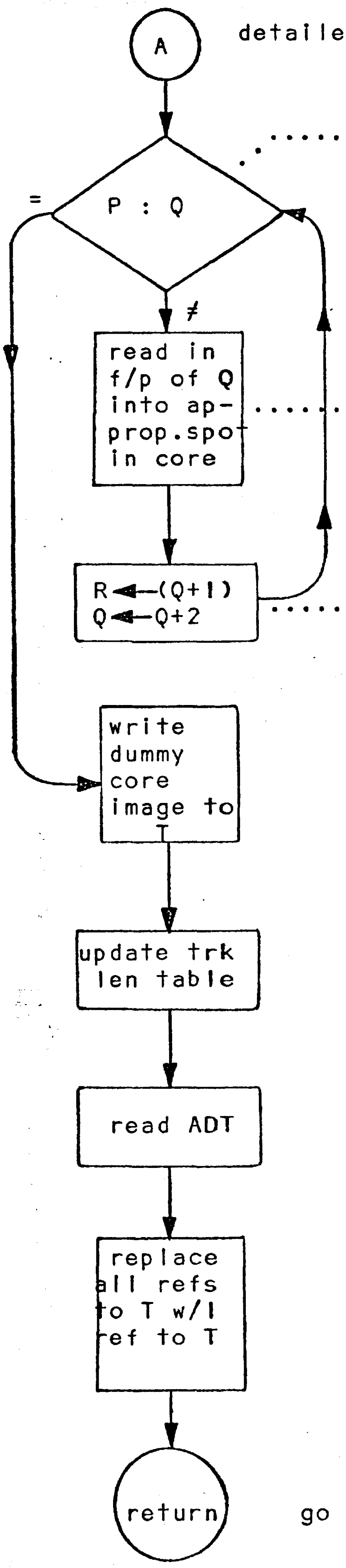
SLEEP OVERLAY - track packing



go form dummy in-core
track image.

SLEEP OVERLAY - Track packing (cont'd.)

detailed view

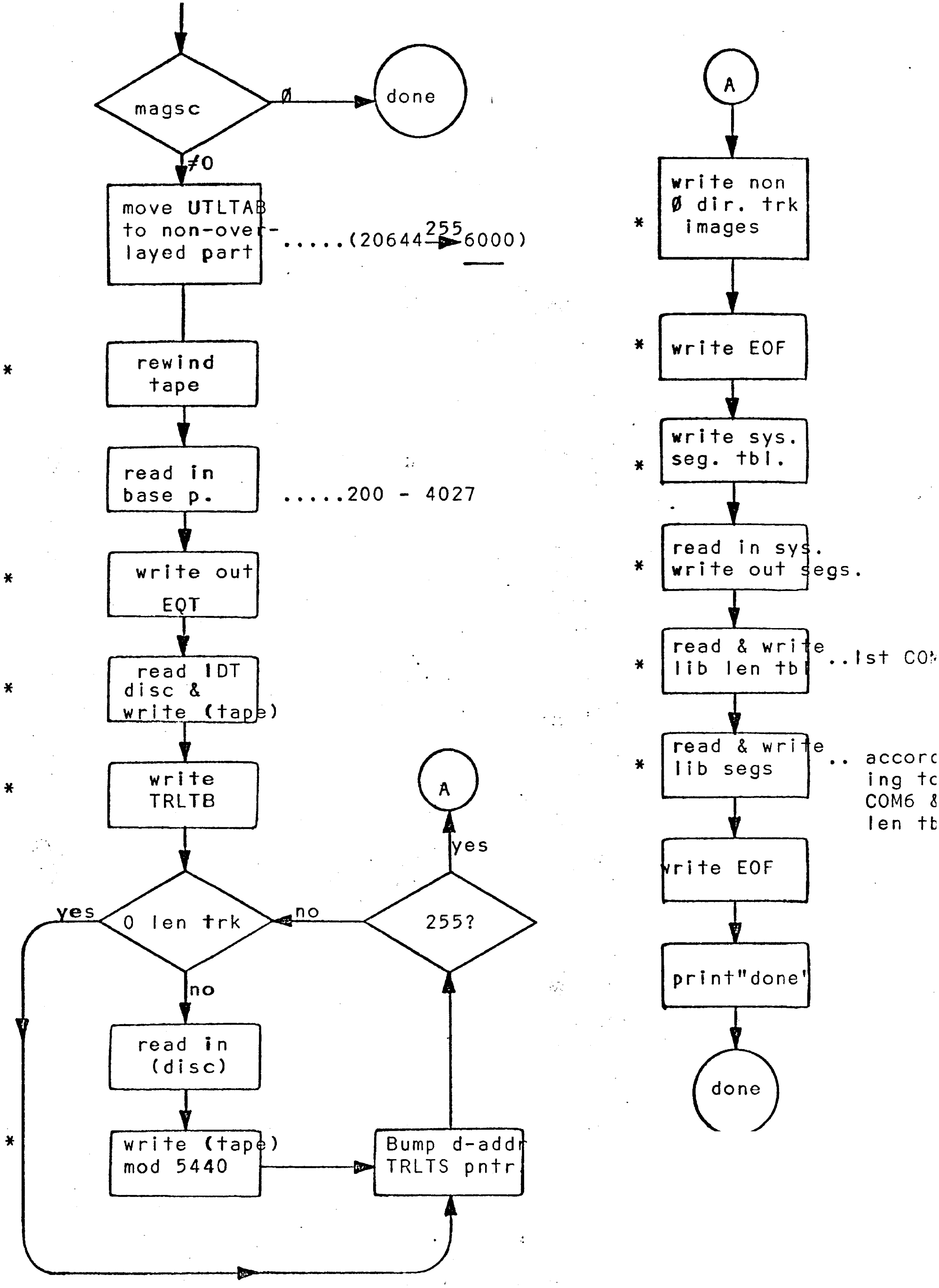

A
P : Q
=
≠
read in f/p of Q into ap- prop.spot in core
R ◄—(Q+1)
Q ◄—Q+2
write dummy core image to T
update trk len table
read ADT
replace all refs to T w/I ref to T
return

P phts to past last sub-directory
Q phts to Ist sub-directory entry

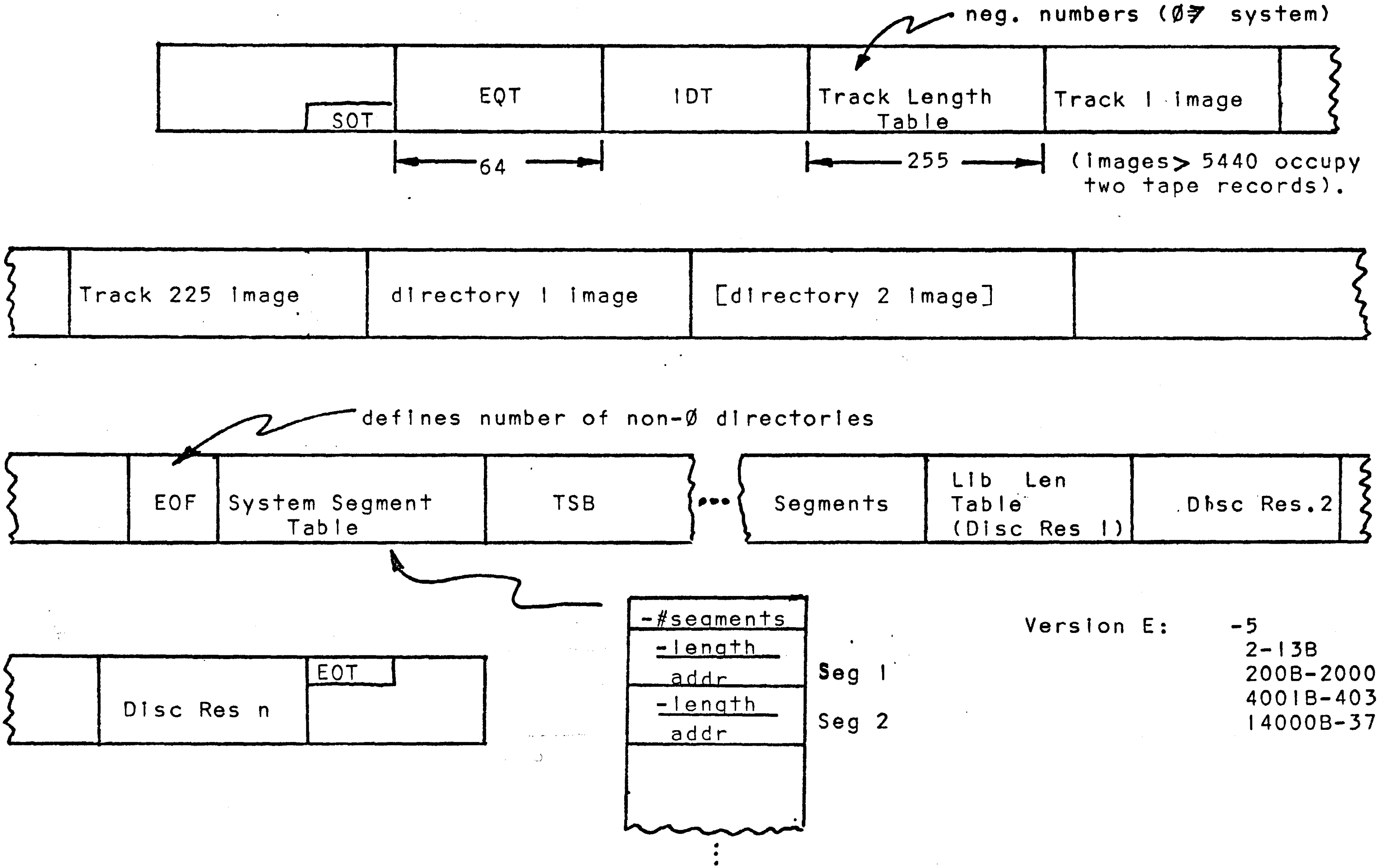
old d-addr = (Q)
len = [(Q+1)-R] *64 words
new d-addr =R
corresponding core loc = LIBD +64*R

update new d-addr pntr, R
update sub-direct pntr, Q

go process next track


magsc
Ø
done
≠0
move UTLTAB to non-overlayed part
.....(20644 —255→ 6000)
*
rewind tape
read in base p.
.....200 - 4027
*
write out EQT
*
read IDT disc & write (tape)
*
write TRLTB
yes
0 len trk
no
255?
yes
A
no
read in (disc)
*
write (tape) mod 5440
Bump d-addr TRLTS pntr.
A
*
write non Ø dir. trk images
*
write EOF
*
write sys. seg. tbl.
*
read in sys. write out segs.
*
read & write lib len tbl
..lst COM
*
read & write lib segs
.. accord ing to COM6 & len tb
write EOF
print"done"
done

SLEEP TAPE FORMAT



Version E: -5
2-13B
200B-2000B
4001B-4030B
14000B-37300B

TSB BBDL LOADING

S.A. = 37760B
HLT 77B
Protect loader, and RUN, w/switch 0 down

a) Obtained from disc:
   i) System
   ii) I.D. information
   iii) User library
   iv) Hardware configuration
b) Requires a SLEpt System on disc

TSB LOADER OPTIONS

1. Generation - S.A. = 2000B; NO LIBRARY
   Creates a system from scratch; no library, i.d's, etc.
   a) Configurations different than:
      - One 64 track disc/drum
      - NO MAG tape
      - NO PHONES
      must be indicated

2. Update - S.A. = 2000B; LIBRARY; NO MAG TAPE
   Updates the software (new versions, patches), preserving the user library
   a) Loads system from paper tape
   b) Preserves
      i) DIRECTORY and user programs/files
      ii) ID information
      iii) Latest TSB hardware configuration
   c) Requires a SLEpt system on disc
   d) Cannot LOCK any system track. (system, swap, IDT/ADT, or DIRECTORY)

TSB LOADER OPTIONS (Cont'd)

3. Mag Tape Awake - S.A. = 2000B; LIBRARY; MAG TAPE
   Restores a system from a SLEEP tape
   a) System, ID information, and user library from mag tape
   b) Preserves SLEEP tape's hardware configuration
   c) Requires a SLEpt system on mag tape. (the contents of the core or disc are irrelevant)
   d) Can LOCK any hardware disc tracks (except 0)

4. Emergency Resuscitation - S.A. = 3000B
   Attempts to restore a system after a failure
   a) Loads system from paper tape
   b) Attempts to preserve
      i) DIRECTORY and user programs/files
      ii) ID information
      iii) Hardware configuration

   The success or failure of Emergency Resuscitation depends on whether the EQT and the tables are organized correctly and consistently.A knowledge of the cause and place of failure can determine this, in most cases.

TSB BBDL LOADING

BBDL

Reads track Ø, sector Ø (disc boot) into core Ø → 77B
inserts 'JMP 77B' in 77B, and jump to 77B.
77B overlayed with last word of Ø, Ø; which is a JMP
to status section of disc boot.

DISC BOOT

Status of disc transfer okay: HLT 77B [Operator now
protects BBDL and specifies system (SWØ=Ø:TSB/SW0=1:DOS)]
DOS: Boot reads Ø,2 into "DMS" and transfers to "DMS"
TSB: Boot reads Ø,1 into "RT/TS" and transfers to "RT/TS"
("DMS" and/or "RT/TS" must have been correctly set)

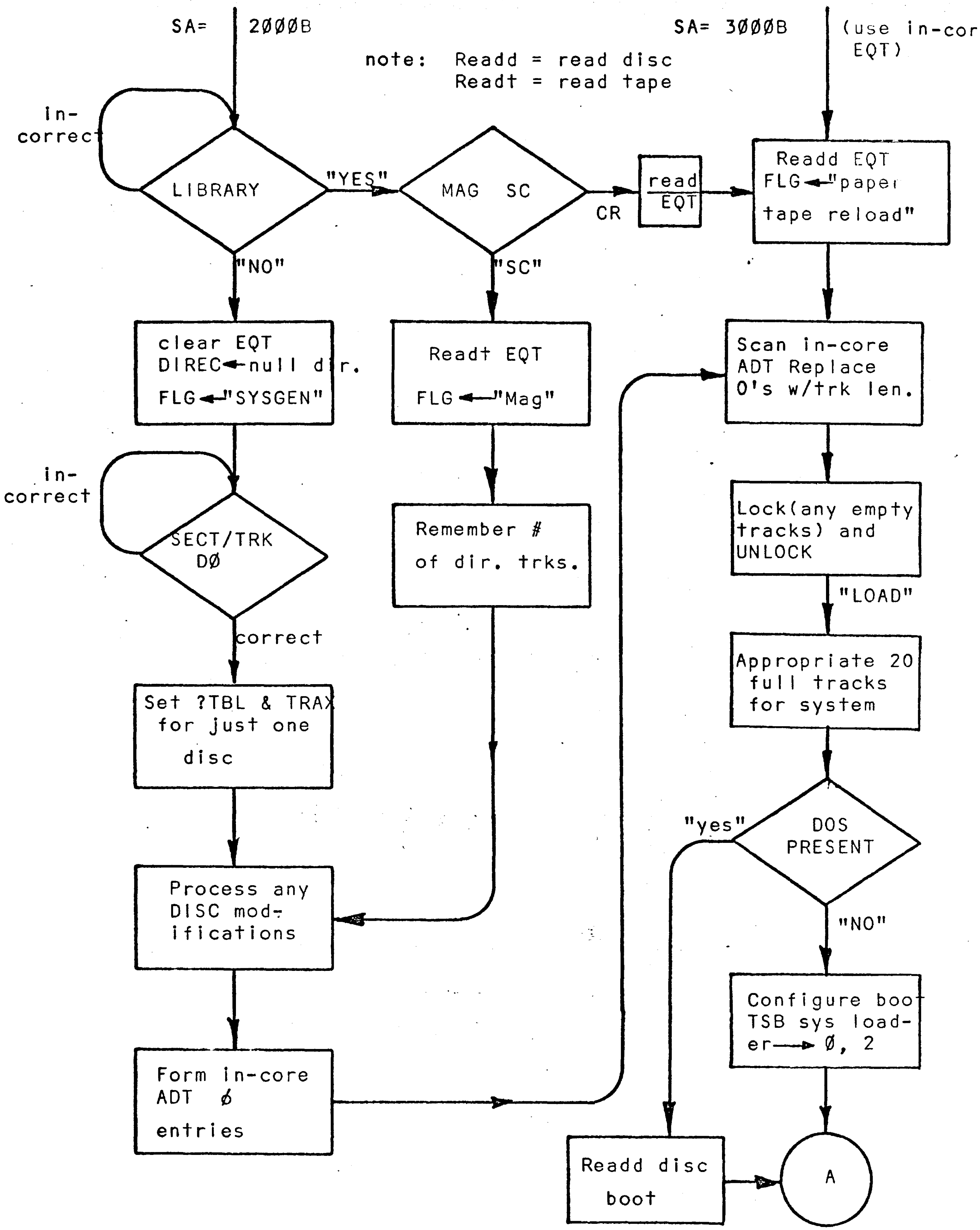
TSB
SYSTEM
LOADER

("RT/TS")

Reads: Ø, 3 into	(ØB to 12ØØØB)	TSB base & Loader
sys 2 into	(14ØØØB to 265ØØB)	Core resident$_1$
sys 3 into	(265ØØB to 373ØØB)	Core resident$_2$

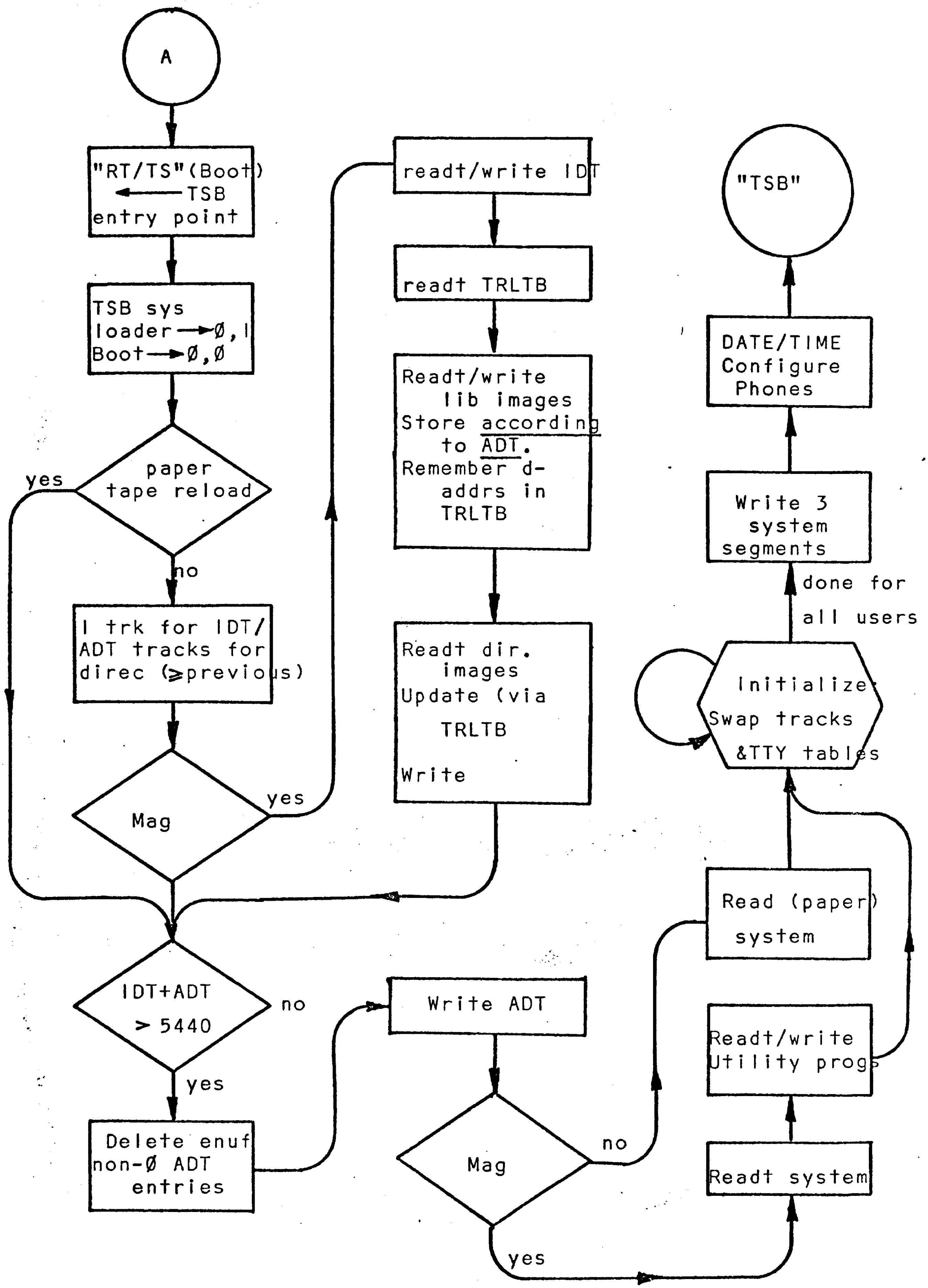
(sys 2 and sys 3 must be set)
Transfers control to date/time section of loader.

2ØØØA TSB LOADER

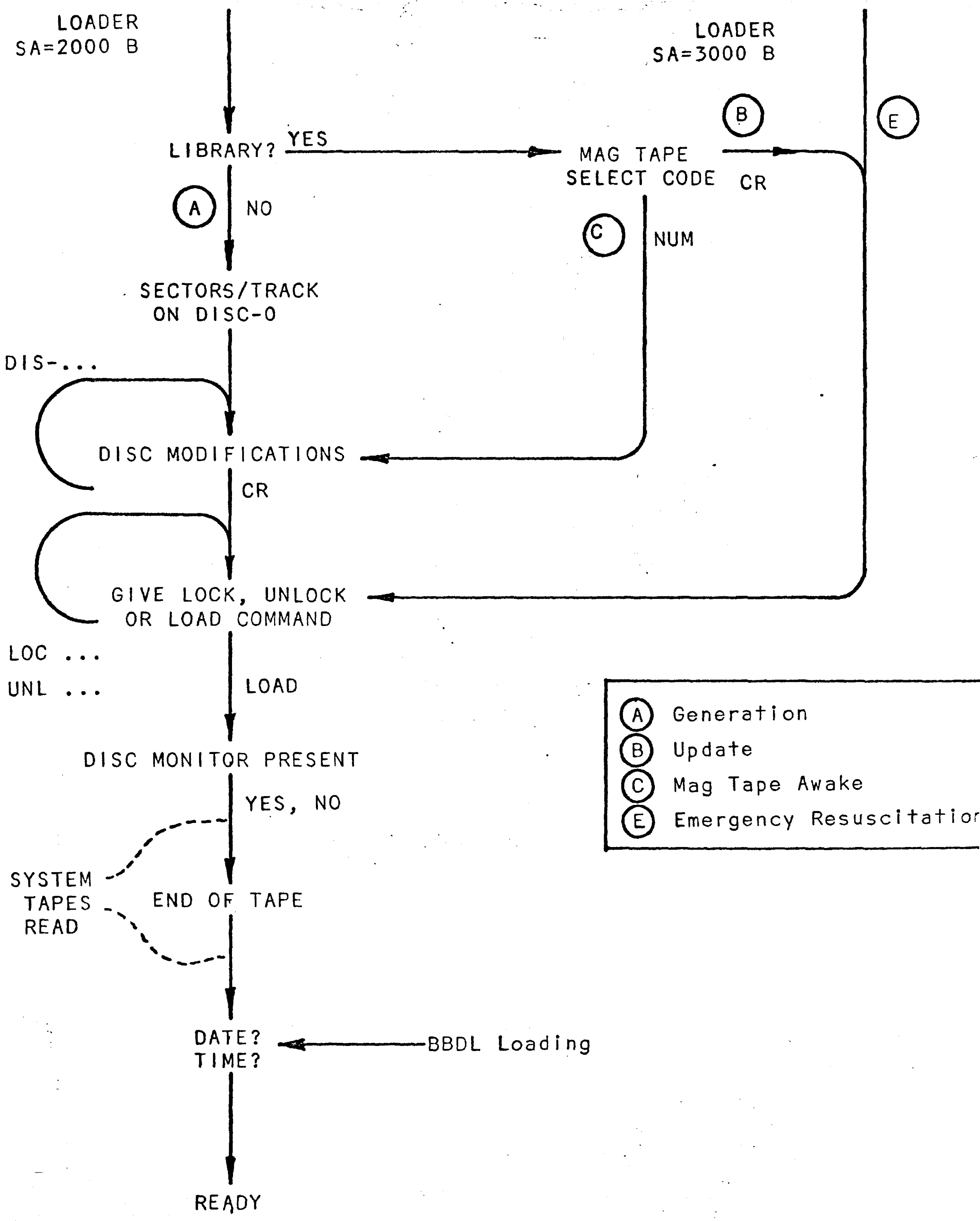


note: Readd = read disc
Readt = read tape

SA= 2ØØØB
in-correct
LIBRARY
"YES"
MAG SC
CR
read EQT
SA= 3ØØØB
(use in-cor EQT)
Readd EQT FLG ← "paper tape reload"
"NO"
"SC"
clear EQT DIREC ← null dir. FLG ← "SYSGEN"
Readt EQT FLG ← "Mag"
Scan in-core ADT Replace O's w/trk len.
in-correct
SECT/TRK DØ
Remember # of dir. trks.
Lock(any empty tracks) and UNLOCK
correct
"LOAD"
Set ?TBL & TRAX for just one disc
Appropriate 20 full tracks for system
"yes"
DOS PRESENT
Process any DISC mod-ifications
"NO"
Configure boot TSB sys load-er → Ø, 2
Form in-core ADT ø entries
Readd disc boot
A

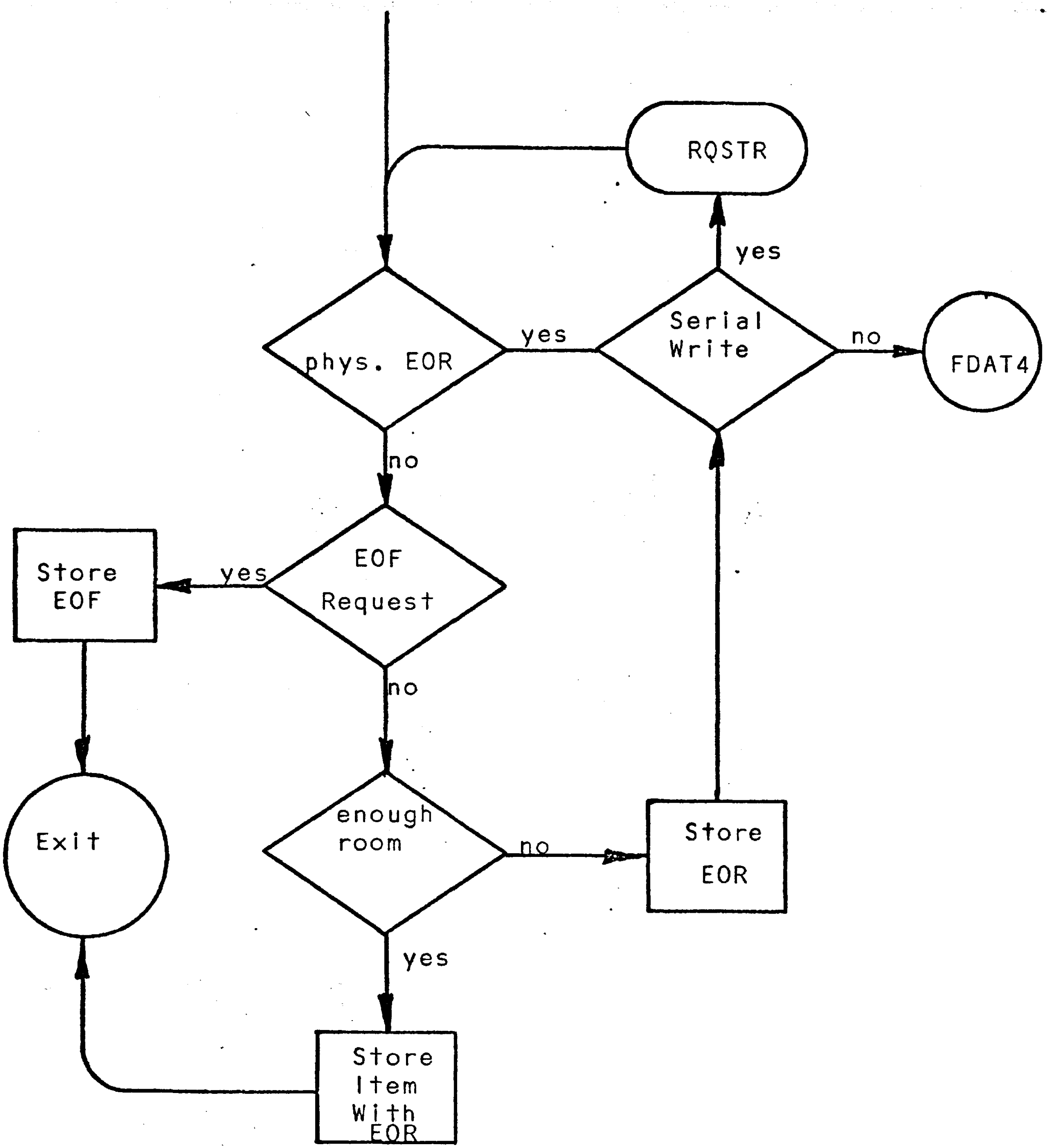

A
"RT/TS"(Boot)
←TSB
entry point
TSB sys
loader→Ø,1
Boot→Ø,Ø
paper
tape reload
yes
no
1 trk for IDT/
ADT tracks for
direc (≥previous)
Mag
yes
readt/write IDT
readt TRLTB
Readt/write
lib images
Store according
to ADT.
Remember d-
addrs in
TRLTB
Readt dir.
images
Update (via
TRLTB
Write
IDT+ADT
> 5440
no
yes
Delete enuf
non-Ø ADT
entries
Write ADT
Mag
no
yes
Readt system
Readt/write
Utility progs
Read (paper)
system
Initialize
Swap tracks
&TTY tables
done for
all users
Write 3
system
segments
DATE/TIME
Configure
Phones
"TSB"

TSB LOADER OPERATOR'S FLOW CHART

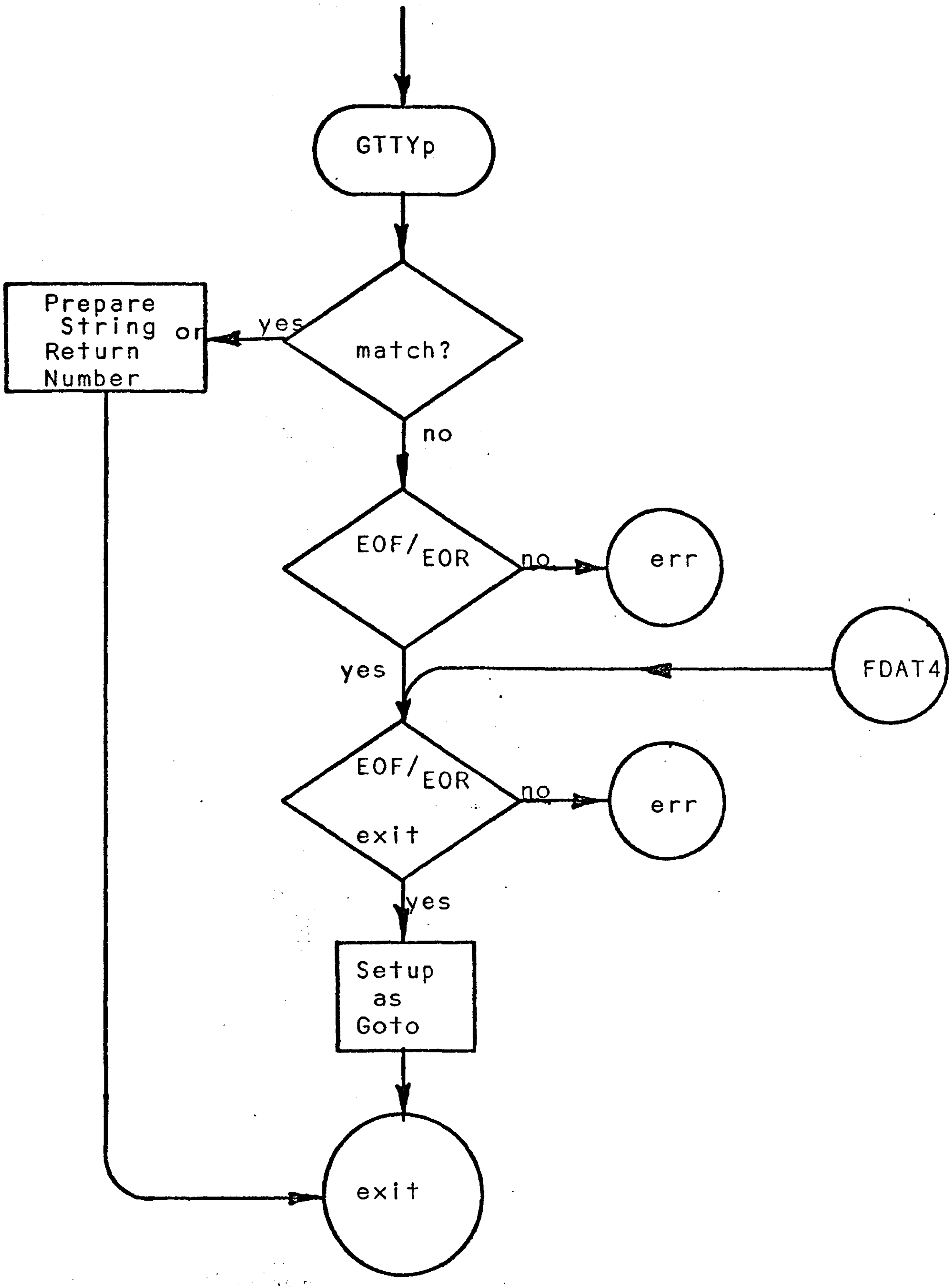


BASIC FILE MANIPULATION ROUTINES

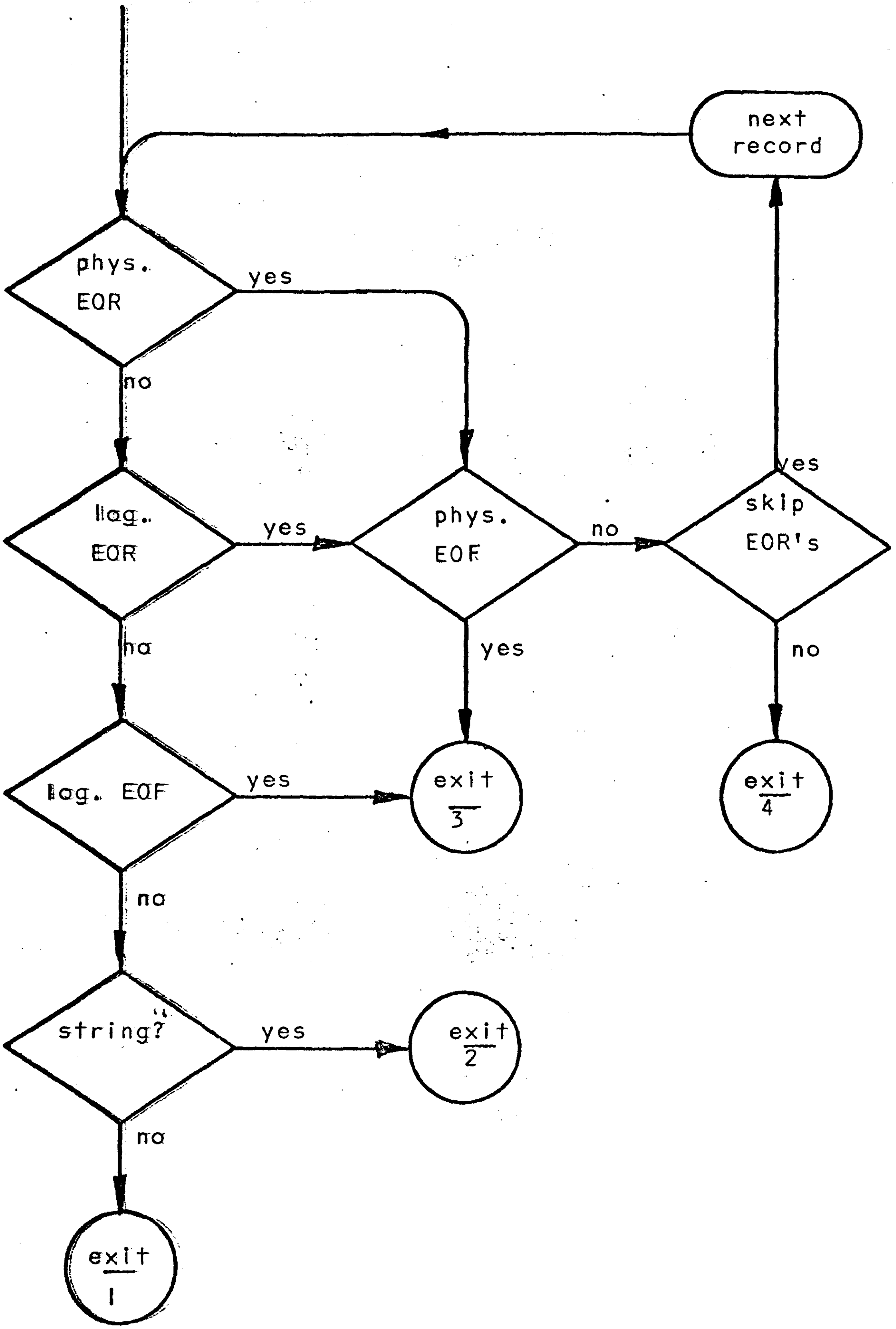
BASIC - STORE FILE DATA ITEM (FILST)



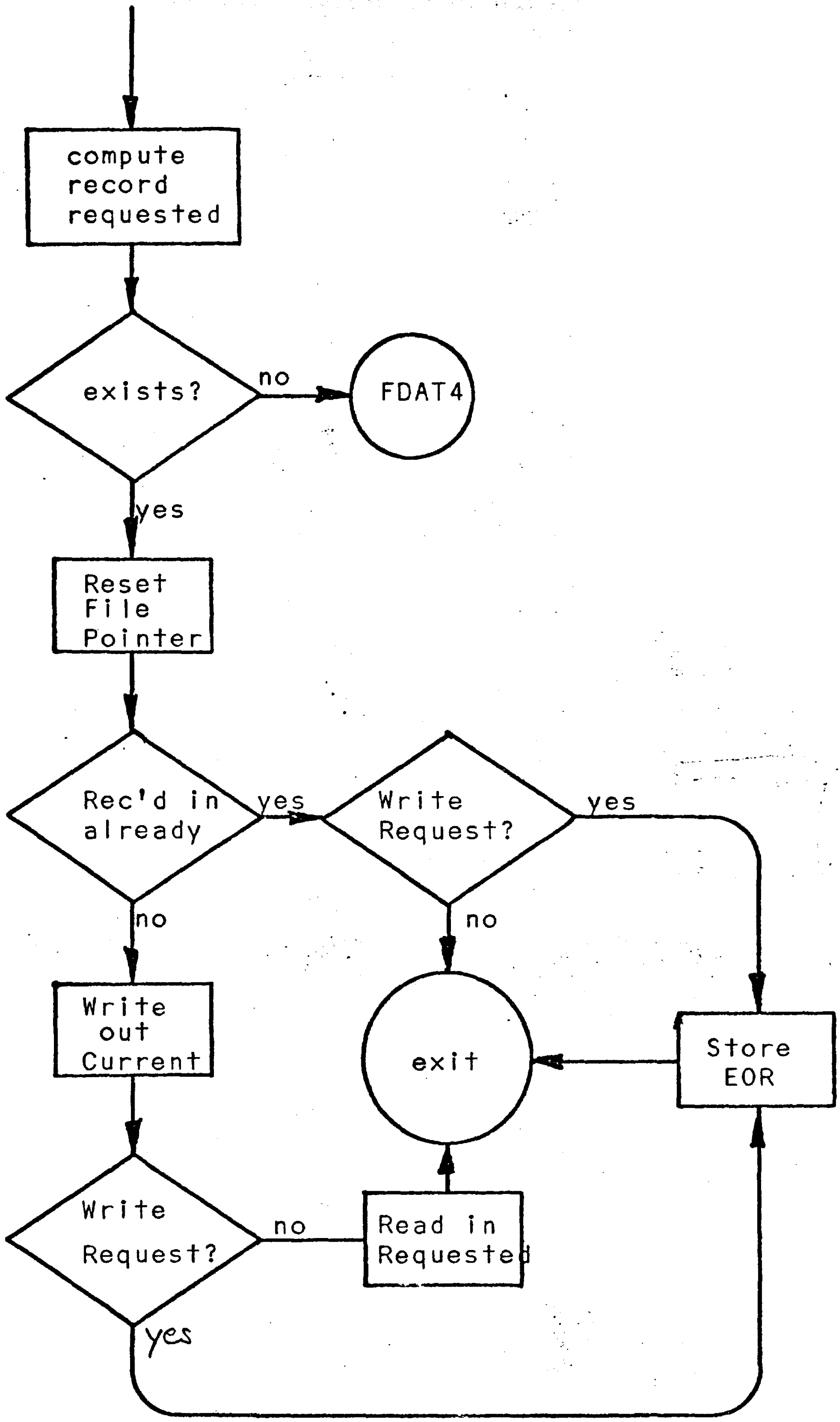
BASIC - FETCH FILE DATA ITEM (FDAT)



BASIC FILE DATA TYPE DETERMINATION (GTTYp)



BASIC FILE RECORD MANIPULATION (RQSTR)



compute record requested
exists?
no
FDAT4
yes
Reset File Pointer
Rec'd in already
yes
Write Request?
yes
no
no
Write out Current
exit
Store EOR
Write Request?
no
Read in Requested
yes