VS1 SYSTEM PROG

WEEK ONE	STRUCTURE AND LOGIC IPL/NIP TASK MANAGEMENT INTRO	TASK MANAGEMENT	TASK MANAGEMENT	JOB MANAGEMENT	EXCP CODING
WEEK TWO	IOS REVIEW DUMP QUESTIONS	PAGE MANAGEMENT SERVICE AIDS	VSAM CATALOG DATA SECURITY	SYSTEM CONSOLES ENQ/DEQ AUTO IPL	OS/VS CATALOG FINAL EXAM

YSLINTRO

SYSTEMS PROGRAMMING

Student Materials

This material was produced for Educational purposes only. Changes may obsolete part or all of this publication. No responsibility is assumed for any inaccuracies that occur.

Los Angeles Advanced Education Center
3424 WILSHIRE BOULEVARD • LOS ANGELES, CALIF. 90010

VIRTUAL STORAGE

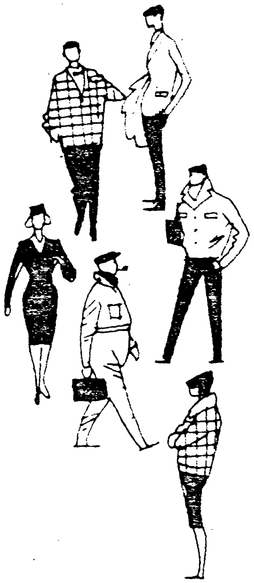
AND VS/1

REFERENCES

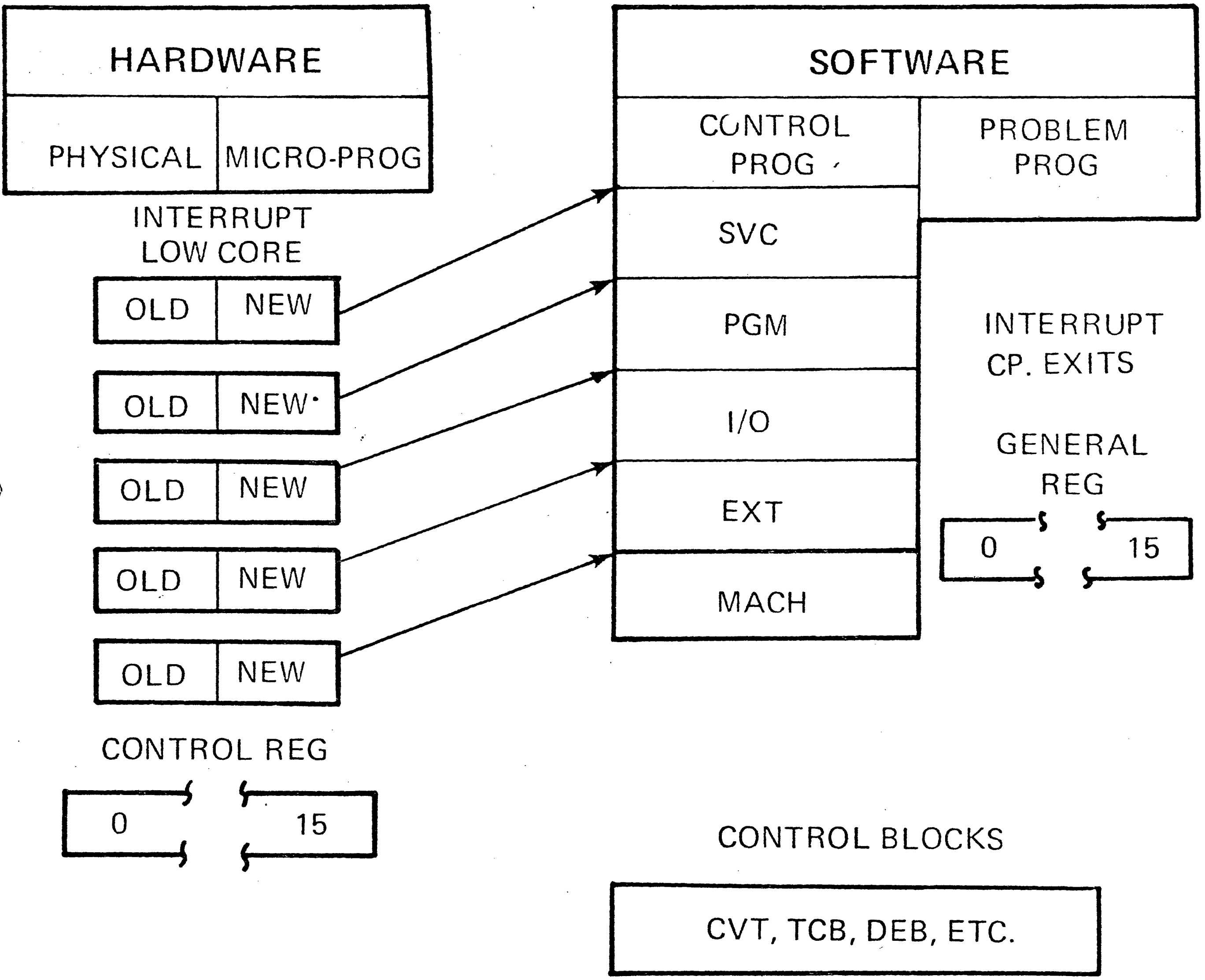
OS/VS1 Planning and Use Guide	GC24-5090
OS/VS1 Guide to Debugging	GC24-5093
OS/VS1 Sysgen Reference	GC26-3791
System/370 Principles of Operation	GA22-7000
System/370 Reference Summary	GX20-1850

PEOPLE

LIGHTS
CONSOLE
SYSOUT
RESULTS



CONSOLE
(COMMANDS)
JCL
INSTRUCTIONS
(LANGUAGES)
INFORMATION
(DATA)



HARDWARE

PHYSICAL | MICRO-PROG

INTERRUPT
LOW CORE

OLD | NEW

OLD | NEW

OLD | NEW

OLD | NEW

OLD | NEW

CONTROL REG

0 … 15

SOFTWARE

CONTROL
PROG

PROBLEM
PROG

SVC

PGM

I/O

EXT

MACH

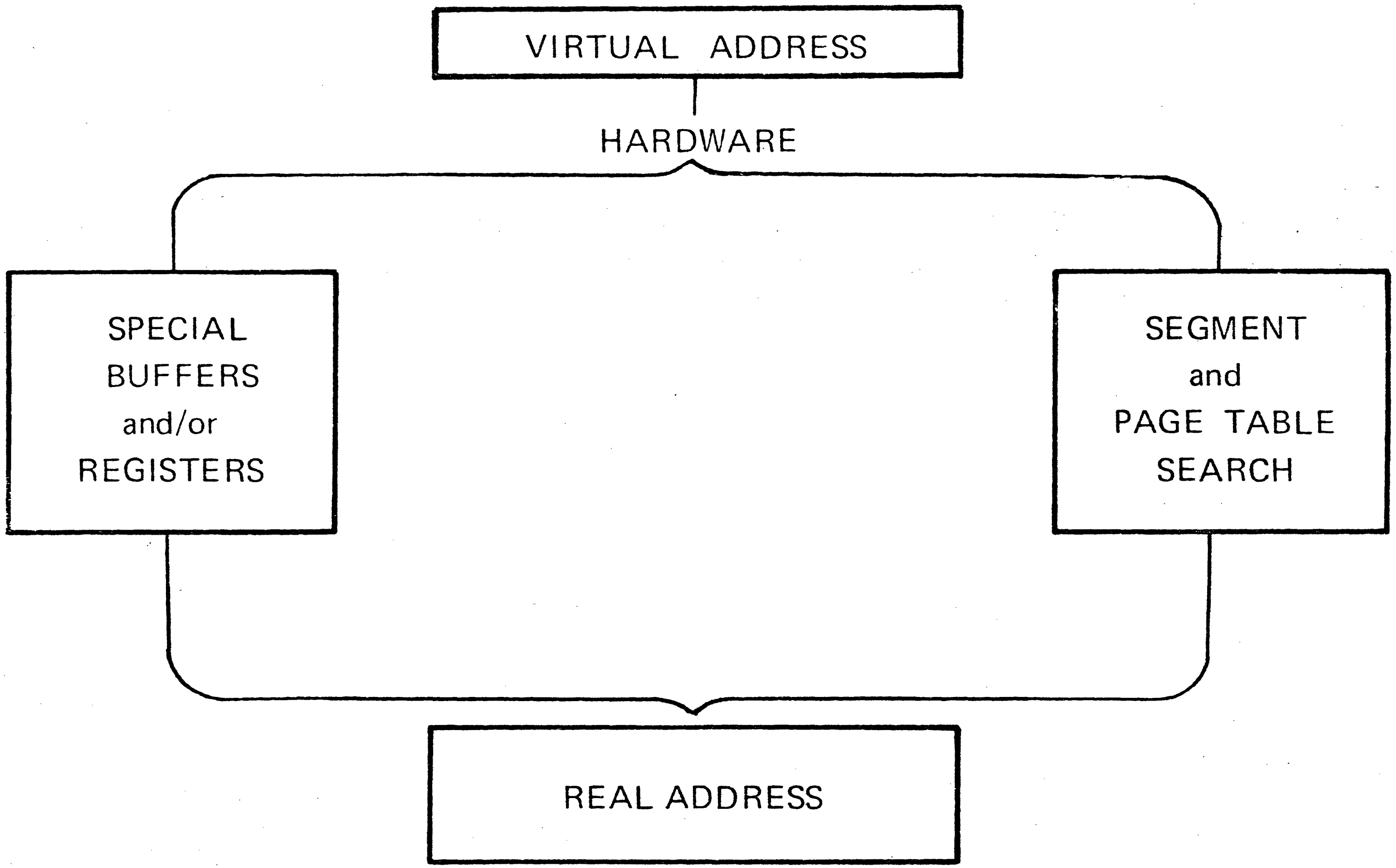
INTERRUPT
CP. EXITS

GENERAL
REG

0 … 15

CONTROL BLOCKS

CVT, TCB, DEB, ETC.


VIRTUAL ADDRESS
HARDWARE
SPECIAL BUFFERS and/or REGISTERS
SEGMENT and PAGE TABLE SEARCH
REAL ADDRESS

INFORMATION USED BY DAT FEATURE

- VIRTUAL ADDRESS
- PROGRAM STATUS WORD
- CONTROL REGISTERS
- REAL STORAGE
- SEGMENT TABLE
- PAGE TABLE

STRUCTURE OF VIRTUAL ADDRESS

SEGMENT	PAGE	BYTE DISPLACEMENT

SEG	PAGE				
64K	2K	8	15 16	20 21	31
64K	4K	8	15 16	19 20	31
1M	2K	8	11 12	20 21	31
1M	4K	8	11 12	19 20	31

PROGRAM STATUS WORD (BC Mode)

14

Channel masks	E	Protect'n key	CMWP	Interruption code
0 6	7	8 11	12 15	16 23 24 31

ILC	CC	Program mask	Instruction address
32	34	36 39	40 47 48 55 56 63

0–5 Channel 0 to 5 masks
6 Mask for channel 6 and up
7 (E) External mask
12 (C=0) Basic control mode
13 (M) Machine-check mask
14 (W=1) Wait state
15 (P=1) Problem state

32-33 (ILC) Instruction length code
34-35 (CC) Condition code
36 Fixed-point overflow mask
37 Decimal overflow mask
38 Exponent underflow mask
39 Significance mask

PROGRAM STATUS WORD (EC Mode)

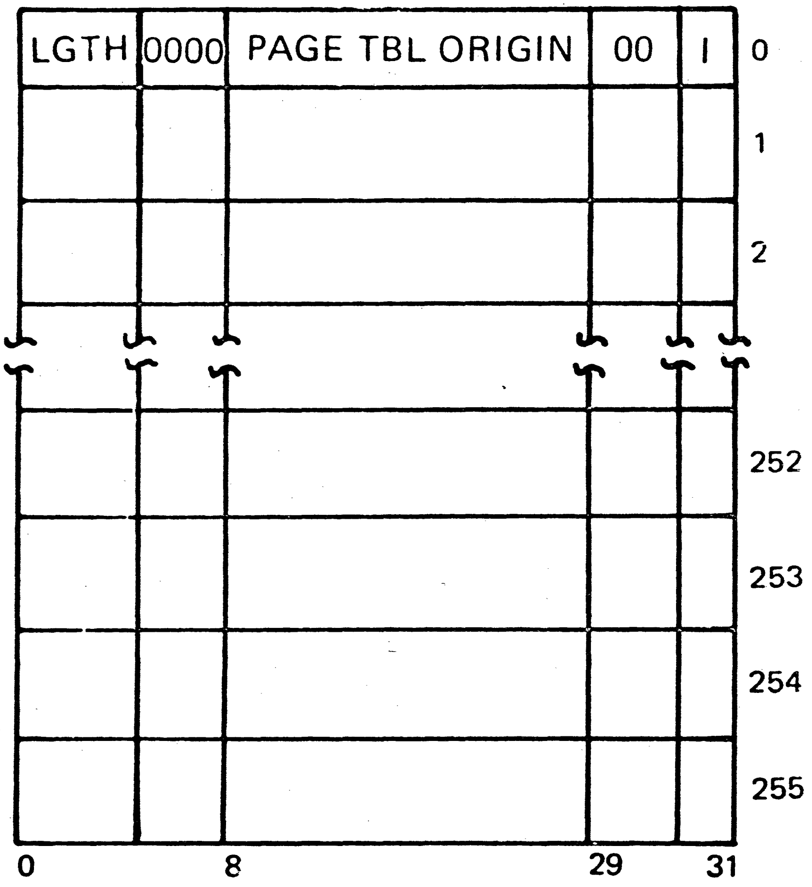
0R00 0TIE	Protect'n key	CMWP	00	CC	Program mask	0000 0000
0 7	8 11	12 15	16	18	20 23	24 31

0000 0000	Instruction address
32 39	40 47 48 55 56 63

1 (R) Program event recording mask
5 (T=1) Translation mode
6 (I) Input/output mask
7 (E) External mask
12 (C=1) Extended control mode
13 (M) Machine-check mask
14 (W=1) Wait state

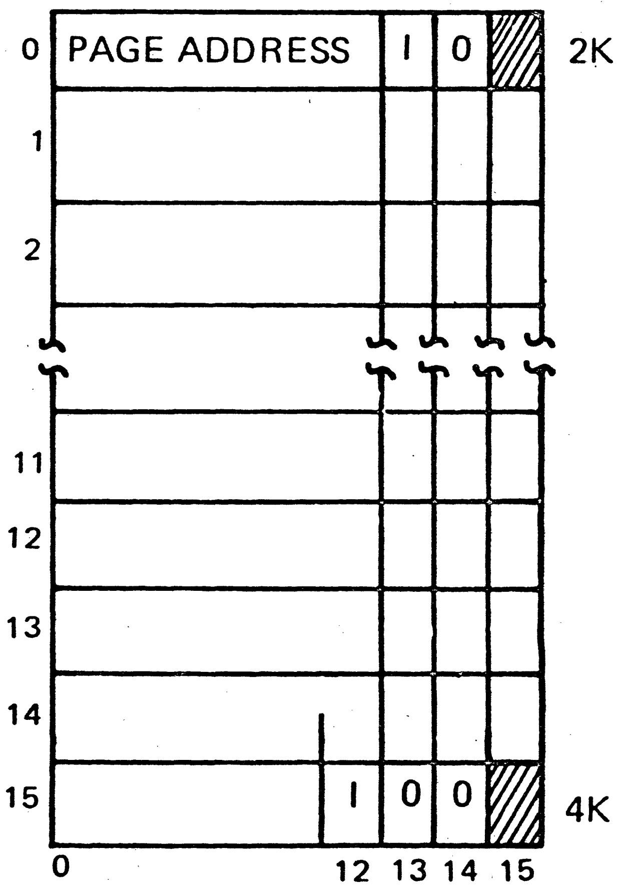
15 (P=1) Problem state
18-19 (CC) Condition code
20 Fixed-point overflow mask
21 Decimal overflow mask
22 Exponent underflow mask
23 Significance mask

SEGMENT TABLE



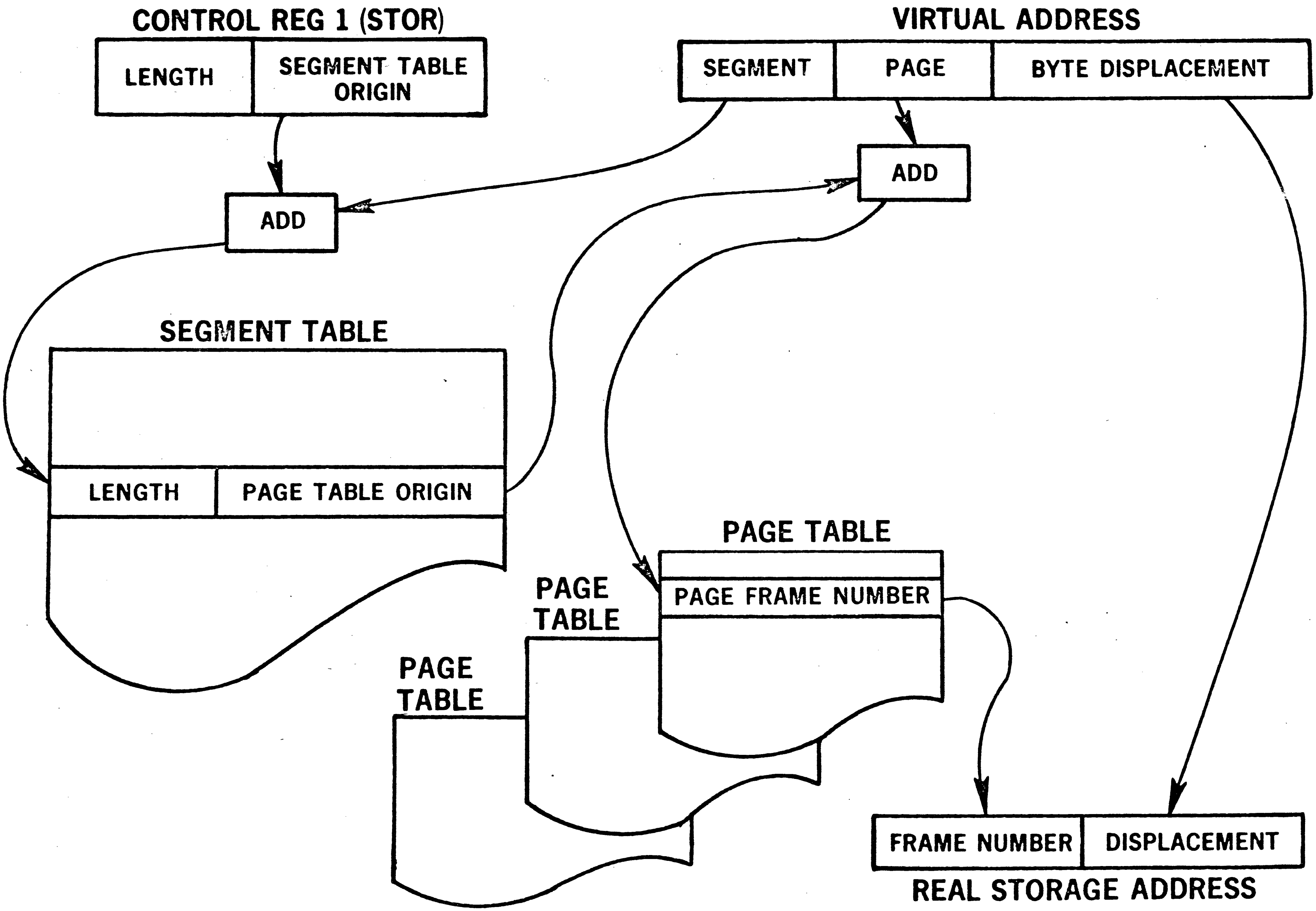
I = INVALID BIT

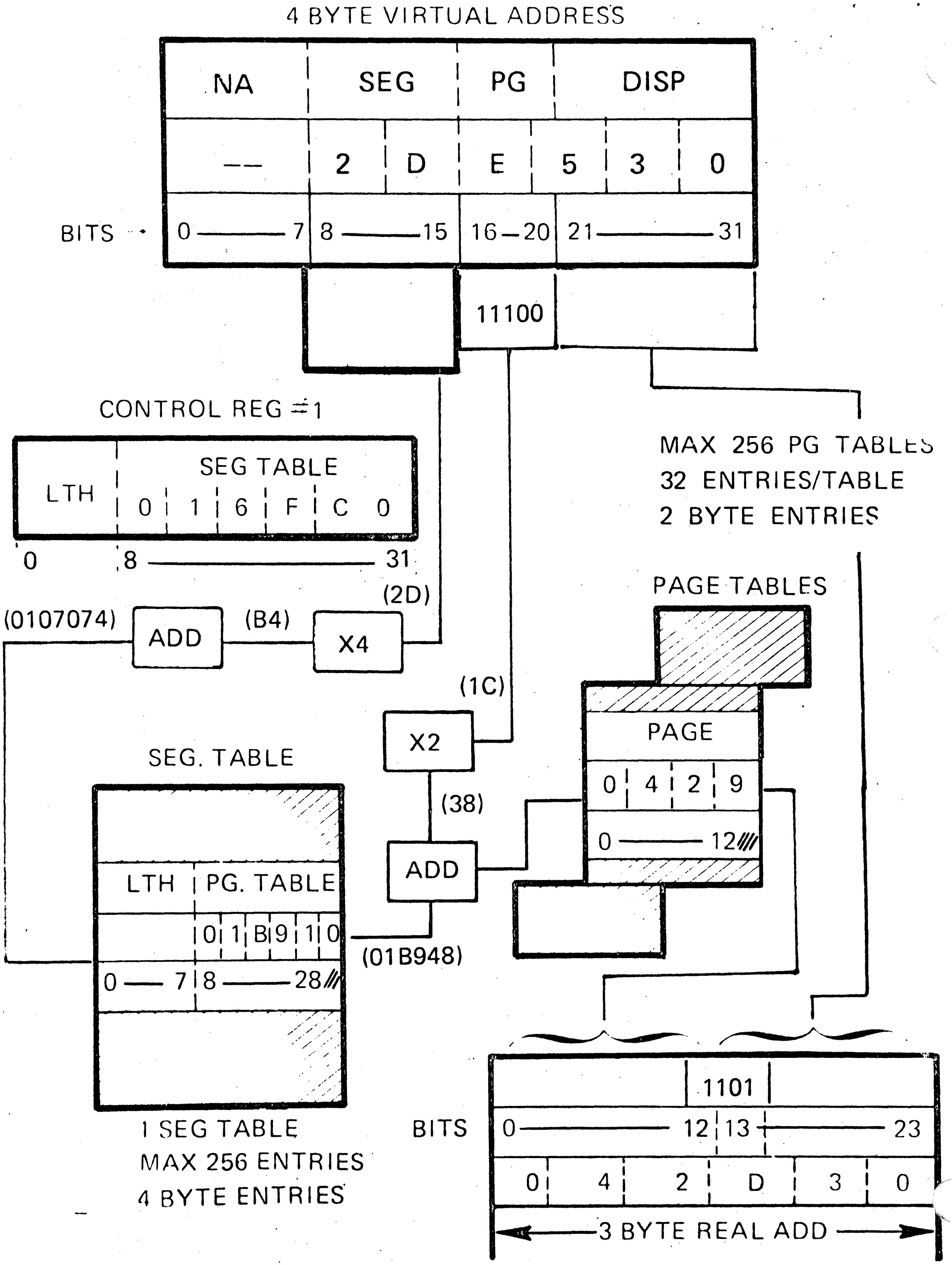
PAGE TABLE



I = INVALID BIT

RELOCATION ACTION TABLE LOOKUP



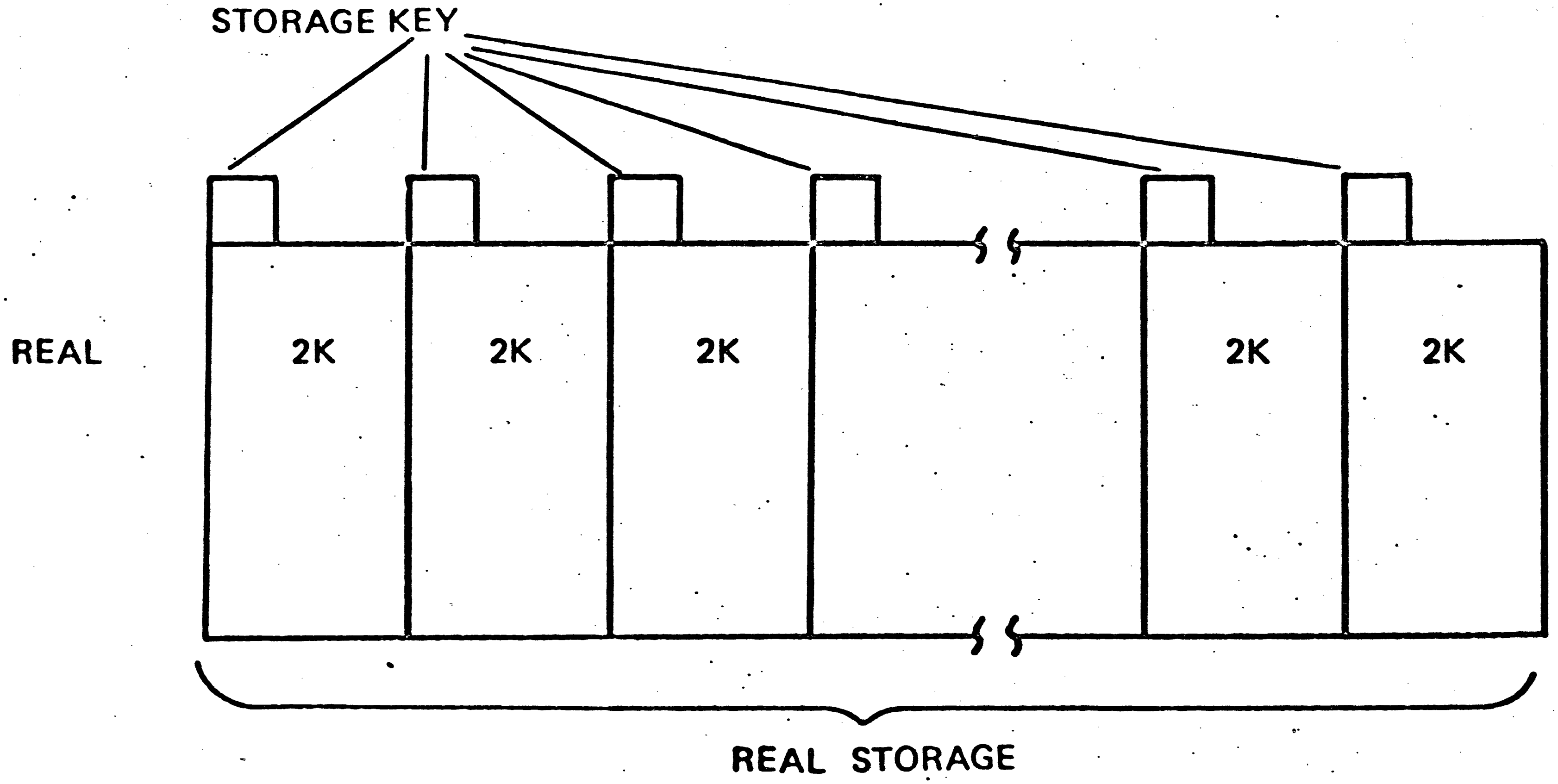

4 BYTE VIRTUAL ADDRESS
NA
SEG
PG
DISP
--
2
D
E
5
3
0
BITS
0 — 7
8 — 15
16 — 20
21 — 31
11100
CONTROL REG =1
LTH
SEG TABLE
0
1
6
F
C
0
0
8 — 31
(2D)
(0107074)
ADD
(B4)
X4
(1C)
X2
(38)
ADD
SEG. TABLE
LTH
PG. TABLE
0
1
B
9
1
0
0 — 7
8 — 28
(01B948)
1 SEG TABLE
MAX 256 ENTRIES
4 BYTE ENTRIES
MAX 256 PG TABLES
32 ENTRIES/TABLE
2 BYTE ENTRIES
PAGE TABLES
PAGE
0
4
2
9
0 — 12
1101
BITS
0 — 12
13 — 23
0
4
2
D
3
0
3 BYTE REAL ADD

TRANSLATION LOOKASIDE BUFFER   MODEL 158, 168

- USED TO BY–PASS FULL ADDRESS TRANSLATION
- KEEPS MOST RECENTLY TRANSLATED ADDRESSES
- 128 ENTRIES
    - VIRTUAL ADDRESS
    - REAL ADDRESS
    - IDENTIFICATION BITS
    - PARITY BITS

ASSOCIATIVE ARRAY   MODEL 135, 145

- SAME FUNCTION AS TLB
- 8 ENTRIES


STORAGE KEY
REAL
2K
2K
2K
2K
2K
REAL STORAGE

ACC	F	R	C	U
0 ——— 3	4	5	6	7

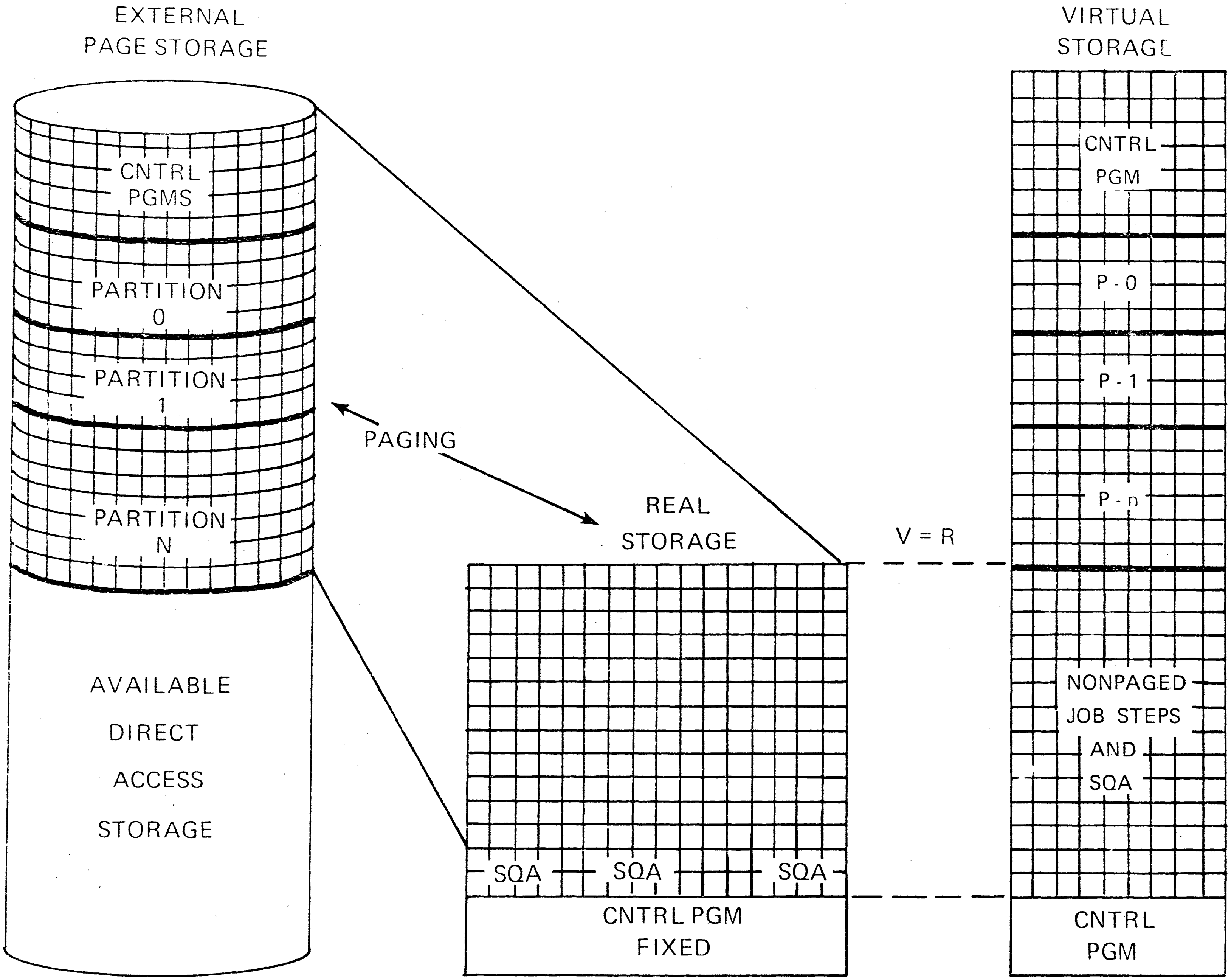
ACC	–	*4 BIT PROTECT KEY*
F	–	*FETCH PROTECT*
R	–	*STORAGE HAS BEEN REFERENCED*
C	–	*STORAGE HAS BEEN CHANGED*
U	–	*RESERVED*

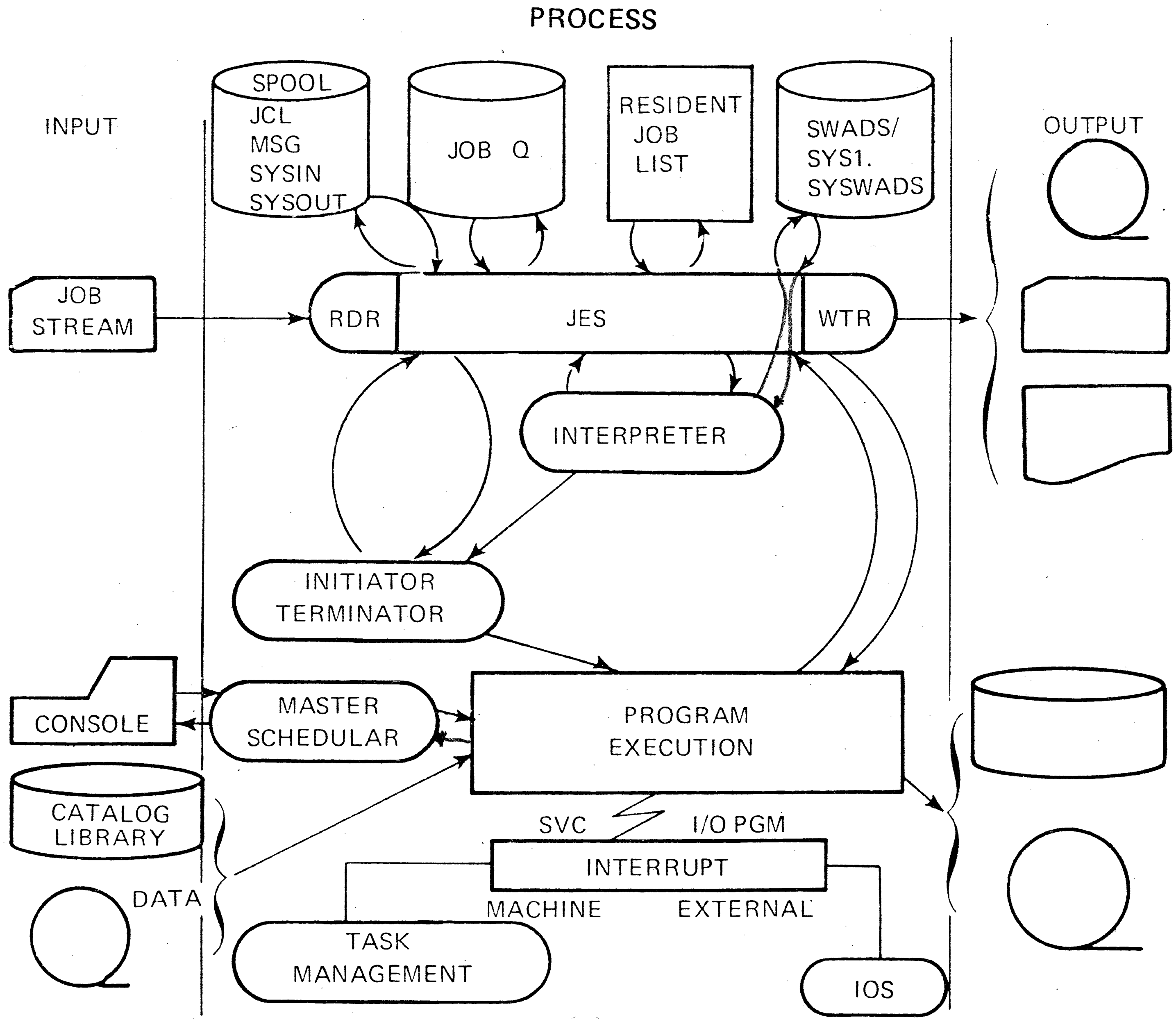
DYNAMIC STORAGE ALLOCATION

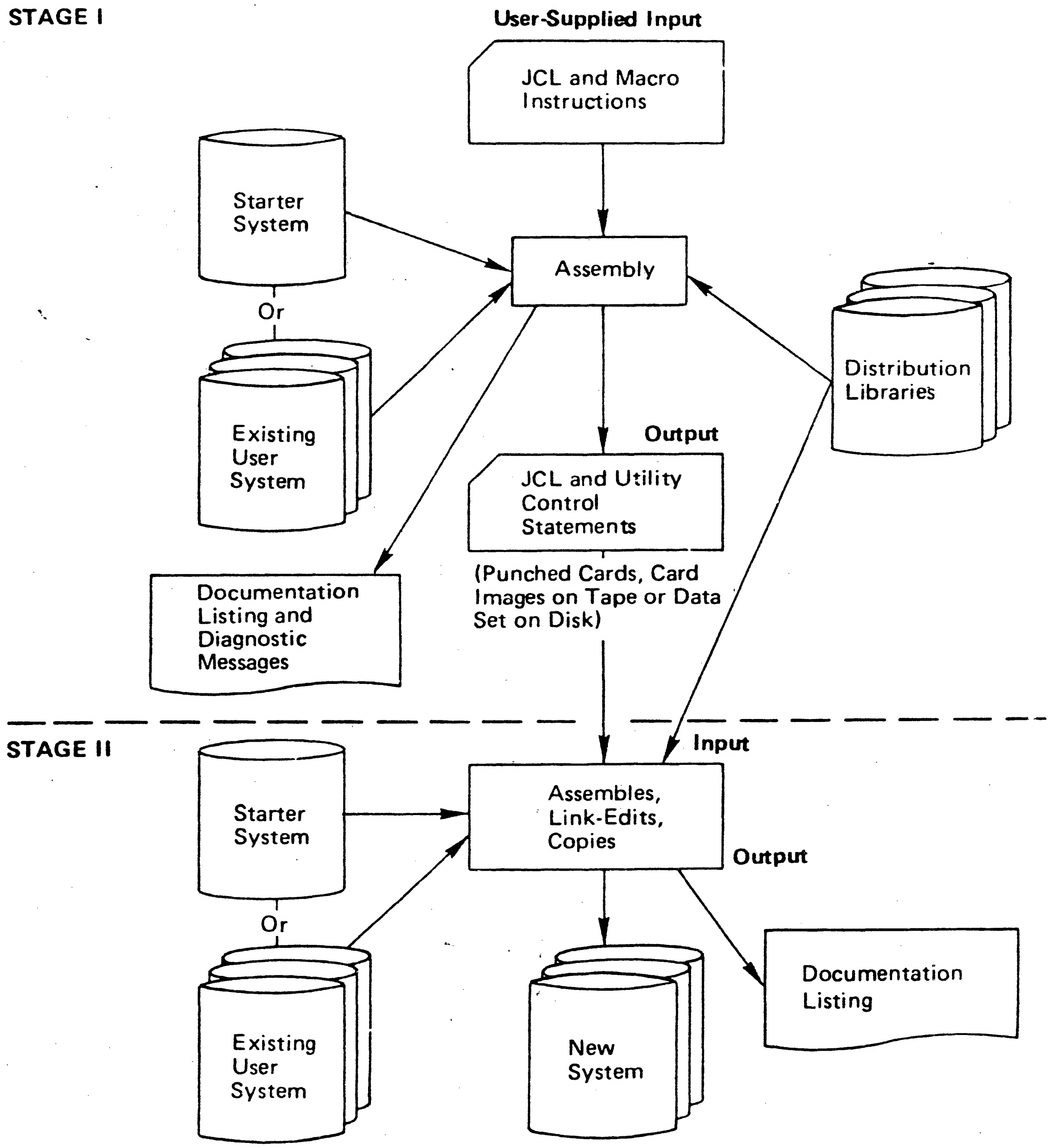
- REFERENCE BIT – BIT 5 OF STORAGE KEY
- CHANGE BIT - BIT 6 OF STORAGE KEY
- RRB INSTRUCTION – OP CODE = B213
  1) RESETS REFERENCE BIT
  2) SETS CONDITION CODE

     0 – R BIT = 0, C BIT = 0
     1 – R BIT = 0, C BIT = 1
     2 – R BIT = 1, C BIT = 0
     3 – R BIT = 1, C BIT = 1
- SSK INSTRUCTION USED TO RESET CHANGE BIT

RRB (new in 370)
para [illegible] se a
página foi [illegible] no último
ciclo.
ver anterior foil


EXTERNAL PAGE STORAGE
CNTRL PGMS
PARTITION 0
PARTITION 1
PARTITION N
AVAILABLE DIRECT ACCESS STORAGE
PAGING
REAL STORAGE
SQA
SQA
SQA
CNTRL PGM FIXED
V = R
VIRTUAL STORAGE
CNTRL PGM
P - 0
P - 1
P - n
NONPAGED JOB STEPS AND SQA
CNTRL PGM


PROCESS
INPUT
OUTPUT
SPOOL
JCL
MSG
SYSIN
SYSOUT
JOB Q
RESIDENT
JOB
LIST
SWADS/
SYS1.
SYSWADS
JOB
STREAM
RDR
JES
WTR
INTERPRETER
INITIATOR
TERMINATOR
CONSOLE
MASTER
SCHEDULAR
PROGRAM
EXECUTION
CATALOG
LIBRARY
DATA
SVC
I/O PGM
INTERRUPT
MACHINE
EXTERNAL
TASK
MANAGEMENT
IOS


STAGE I
User-Supplied Input
JCL and Macro Instructions
Starter System
Or
Existing User System
Assembly
Distribution Libraries
Output
JCL and Utility Control Statements
(Punched Cards, Card Images on Tape or Data Set on Disk)
Documentation Listing and Diagnostic Messages
STAGE II
Input
Starter System
Or
Existing User System
Assembles, Link-Edits, Copies
Output
New System
Documentation Listing

Group	Macro Instruction	Complete	Nucleus	I/O Device
Machine Configuration	CENPROCS	required	required	required
	CHANNEL	required	required	required
	IODEVICE	required	required	required
	UCS	optional	--	optional
	UNITNAME	required	--	required
Control Program	CKPTREST	optional	--	--
	CTRLPROG	required	required	required
	EDITOR	optional	--	--
	JES	optional	--	--
	LOADER	optional	--	--
	MACLIB	optional	--	--
	PAGE	required	required	required
	PARTITNS	required	required	required
	SCHEDULR	required	required	required
	SECONSLE	optional	optional	optional
Data Management	DATAMGT	optional	optional	optional
	GRAPHICS	optional	optional	optional
User-written Routines	LINKLIB	optional	optional	optional
	RESMODS	optional	optional	--
	SVCLIB	optional	optional	--
	SVCTABLE	optional	optional	optional
Generation	DATASET	optional	optional	optional
	GENERATE	required	required	required

System Data Sets	Type	System Residence	Cataloged	Use
Required System Data Sets				
SYSCTLG	seq.	required	no	Production
SYS1.DSSVM	seq.	optional	yes	Control
SYS1.LINKLIB	PDS	optional	yes	Production-Control
SYS1.LOGREC	seq.	required	no	Error Recovery
SYS1.MACLIB	PDS	optional	yes	Production
SYS1.NUCLEUS	PDS	required	yes	Control
SYS1.PARMLIB	PDS	optional	yes	Initialization
SYS1.PROCLIB	PDS	optional	yes	Production
SYS1.SAMPLIB	PDS	optional	optional	Control
SYS1.SVCLIB	PDS	required	yes	Control-Production
SYS1.SYSJOBQE	seq.	optional	yes	Production
SYS1.SYSPOOL	seq.	optional	no	Production
SYS1.SYSWADS	seq.	optional	optional	Production
Optional System Data Sets				
SYS1.ACCT	seq.	optional	no	Administration
SYS1.BRODCAST	direct	optional	yes	Control-Production
SYS1.CMDLIB	PDS	optional	yes	Production
SYS1.DUMP	seq.	optional	optional	Error Recovery
SYS1.IMAGELIB	PDS	optional	yes	Initialization
SYS1.INDMAC	PDS	optional	yes	Production
SYS1.ISPMAC	PDS	optional	yes	Control
SYS1.MANX	seq.	optional	yes	Administration
SYS1.MANY	seq.	optional	yes	Administration
SYS1.RMTMAC	PDS	optional	yes	Initialization
SYS1.TELCMLIB	PDS	optional	yes	Production
SYS1.UADS	PDS	optional	yes	Control
SYS1.VTAMLIB	PDS	optional	yes	Production

IEA760A - SPECIFY VIRTUAL STORAGE SIZE

IEA101A - SPECIFY SYSTEM and/or SET PARAMETERS

[illegible]

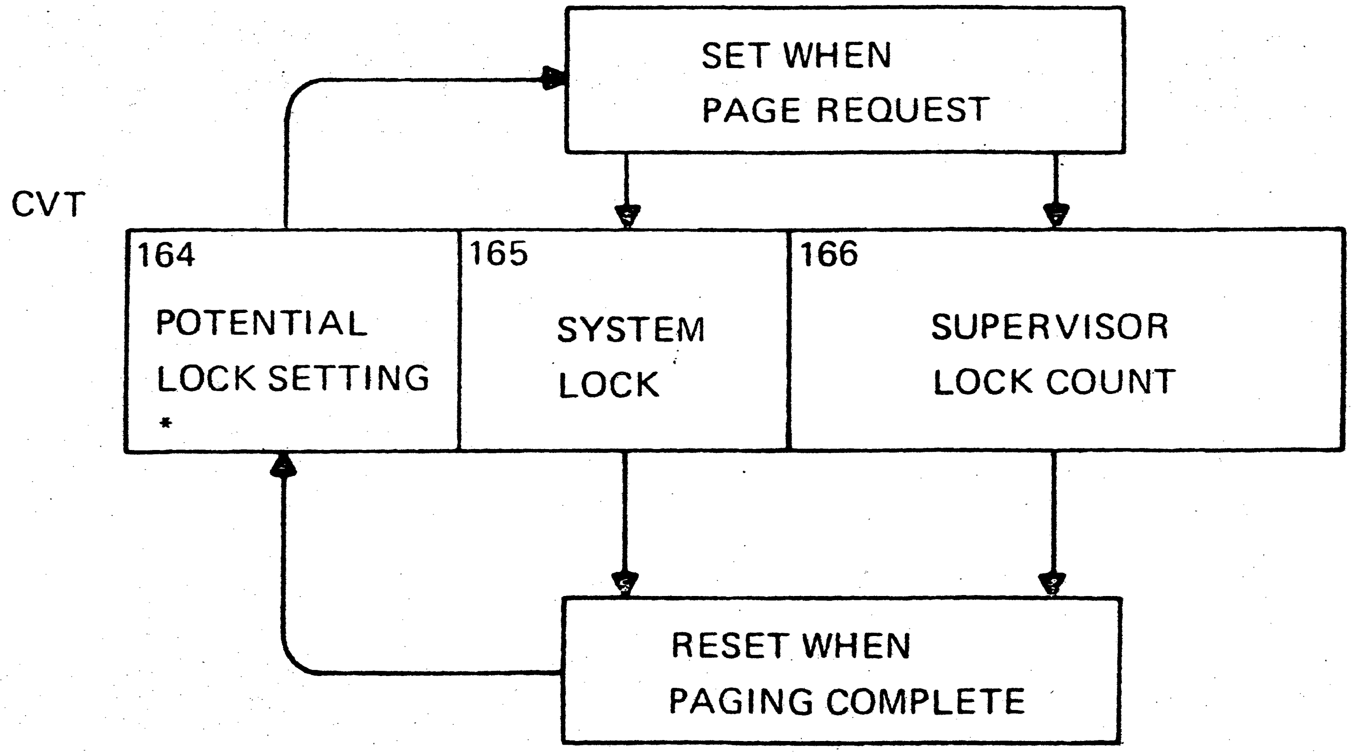
IEE801D - CHANGE PARTITIONS - REPLY YES/NO(, List)

IEE114A - SPECIFY SET PARAMETERS or u

IEE357A - REPLY WITH SMF VALUES or u

SVCs	1	2	3	4
LOCATION	RESIDENT	RESIDENT	TRANSIENT OR RESIDENT/PAGEABLE OR RESIDENT/FIXED	TRANSIENT OR RESIDENT/PAGEABLE OR RESIDENT/FIXED
DESIGN	SERIALLY REUSABLE OR REENTRANT DISABLED	REENTRANT ENABLED OR DISABLED	REENTRANT ENABLED OR DISABLED	REENTRANT ENABLED OR DISABLED
SIZE	ANY SIZE	ANY SIZE	ONE MODULE $\leq$2048	MULTIPLE MODULES EACH $\leq$ 2048
NAME	IGC nnn	IGC nnn	IGC00 nnn	IGCss nnn

LOCKS



SET WHEN PAGE REQUEST

CVT

164 POTENTIAL LOCK SETTING *

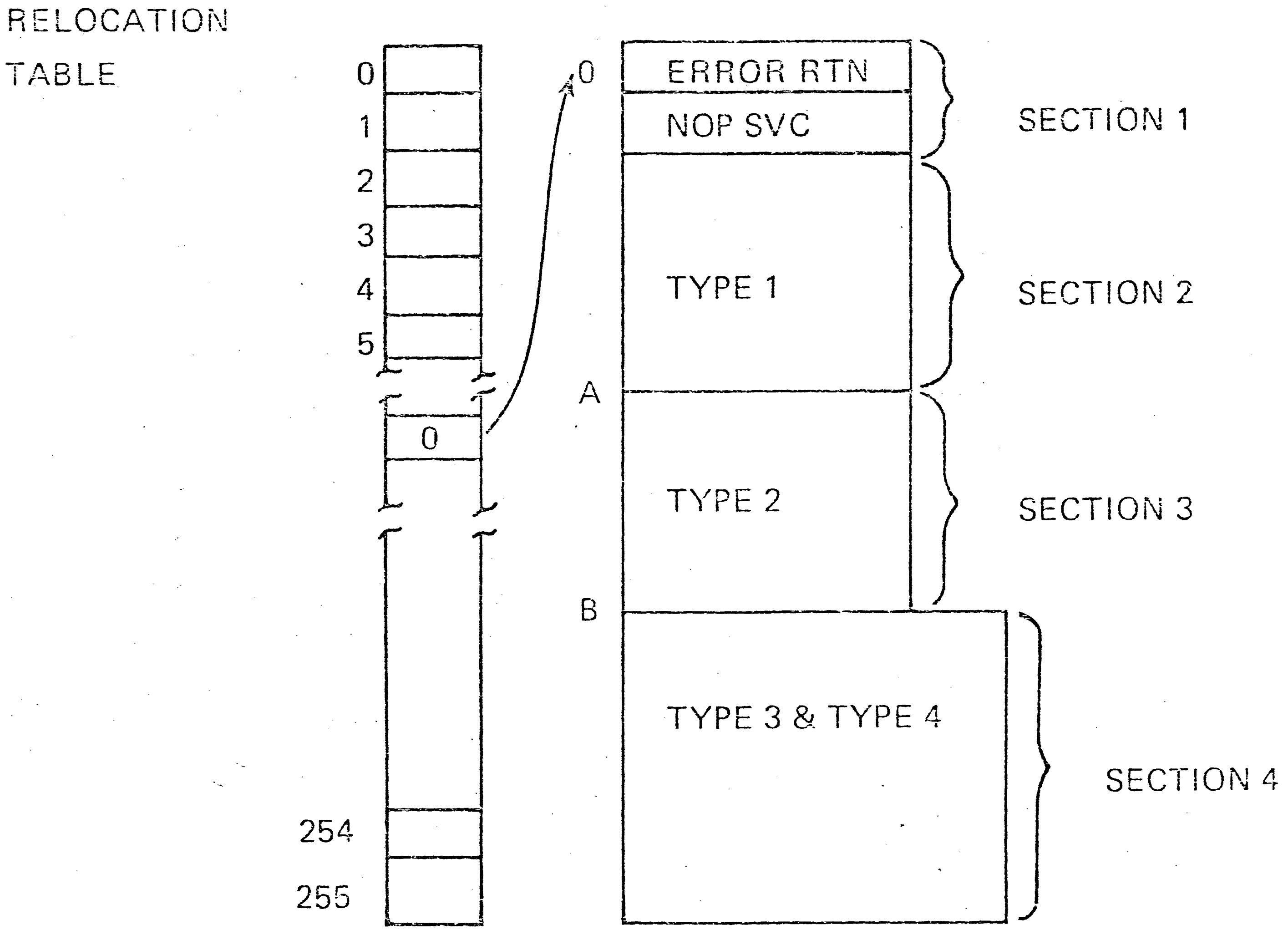
165 SYSTEM LOCK

166 SUPERVISOR LOCK COUNT

RESET WHEN PAGING COMPLETE

* X'C0' - SYSTEM LOCK
X'40' - SUPERVISOR LOCK

SVC TABLE


RELOCATION
TABLE
0
1
2
3
4
5
0
254
255
0
ERROR RTN
NOP SVC
TYPE 1
A
TYPE 2
B
TYPE 3 & TYPE 4
SECTION 1
SECTION 2
SECTION 3
SECTION 4

SVC's

TYPE 1

LOCK	0	VIRTUAL STORAGE ADDRESS OF SVC	0
0			31

TYPE 2

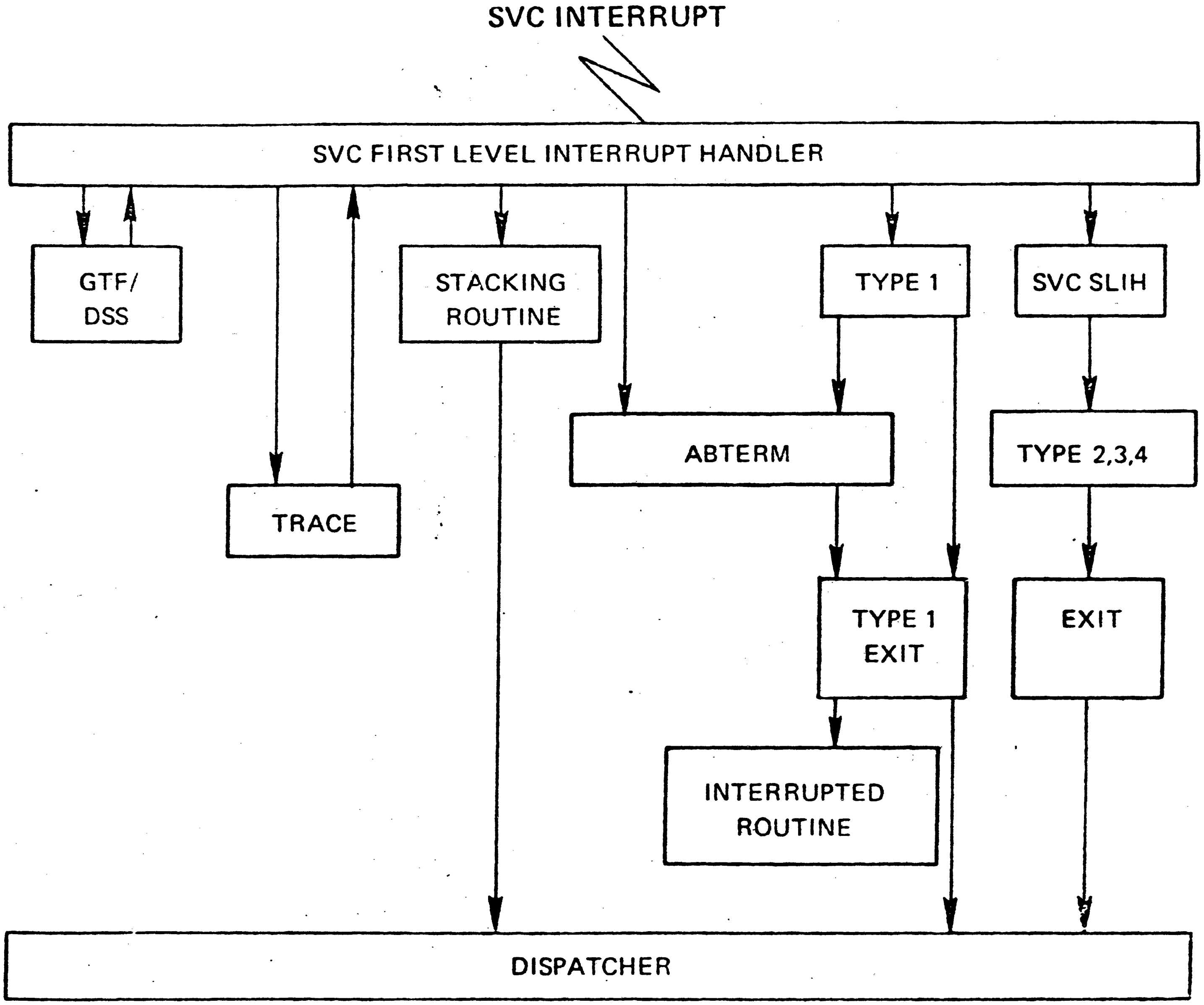
LOCK	0	ENABLE	VIRTUAL STORAGE ADDRESS OF SVC	0 0 0
0				31

TYPE 3 OR TYPE 4 – RESIDENT

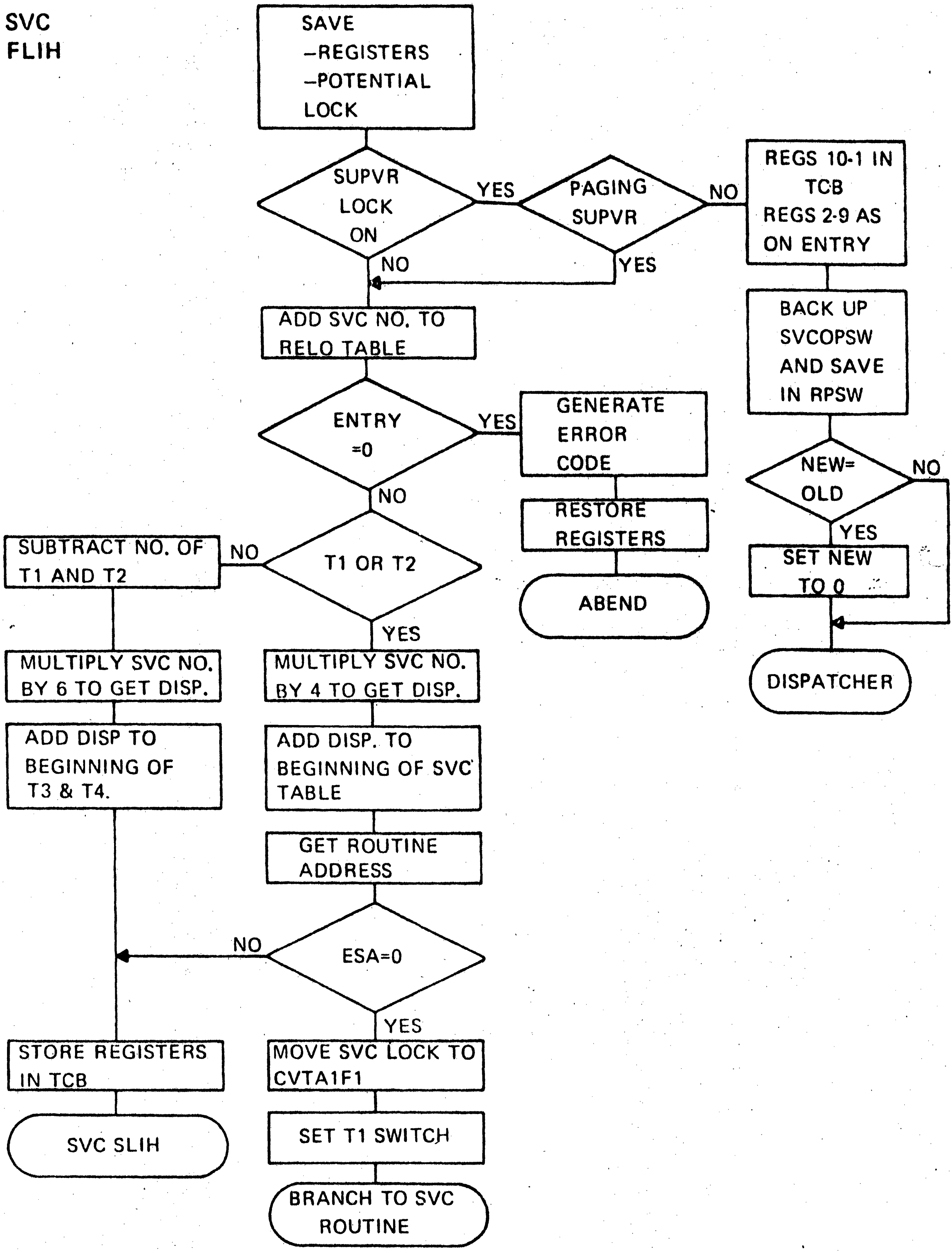
X'FF'	VIRTUAL STORAGE ADDRESS OF SVC	LOCK	0	ENABLE	0 0 0
0					47

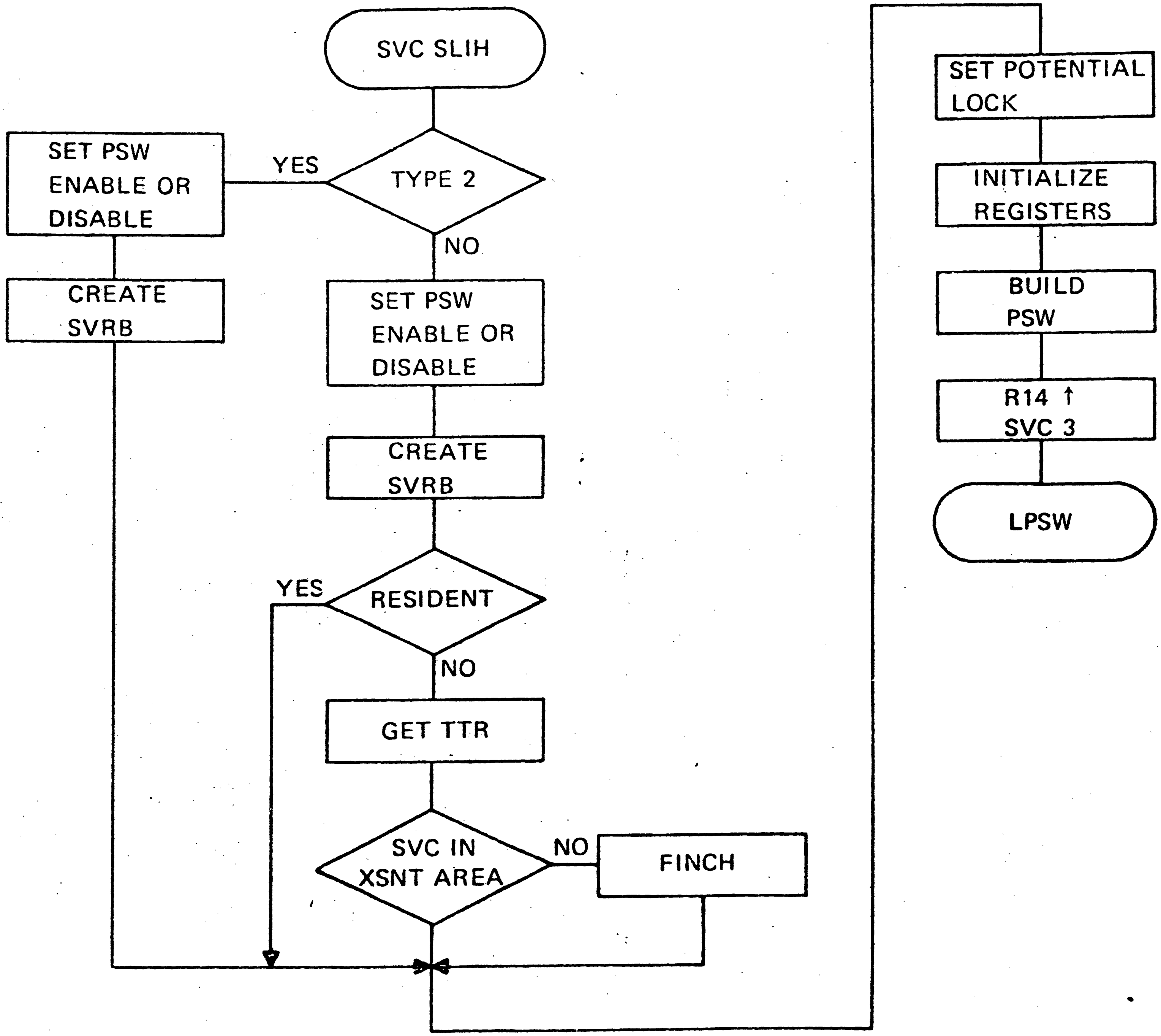
TYPE 3 OR TYPE 4 – NON-RESIDENT

TTR	LENGTH OF FIRST TEXT RECORD	LOCK	MODULE LENGTH	ENABLE	
0					47

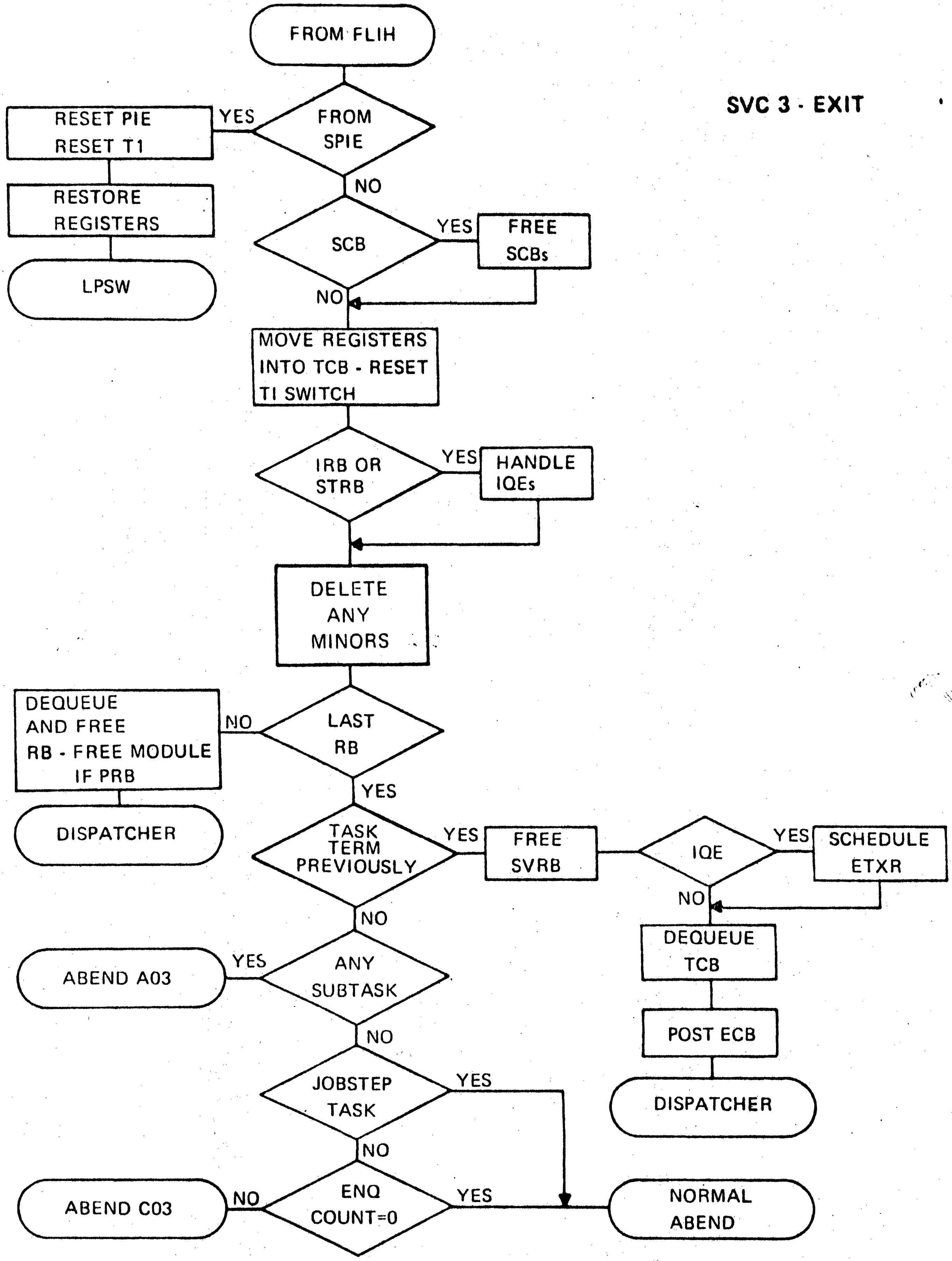

SVC INTERRUPT
SVC FIRST LEVEL INTERRUPT HANDLER
GTF/ DSS
TRACE
STACKING ROUTINE
TYPE 1
SVC SLIH
ABTERM
TYPE 2,3,4
TYPE 1 EXIT
EXIT
INTERRUPTED ROUTINE
DISPATCHER

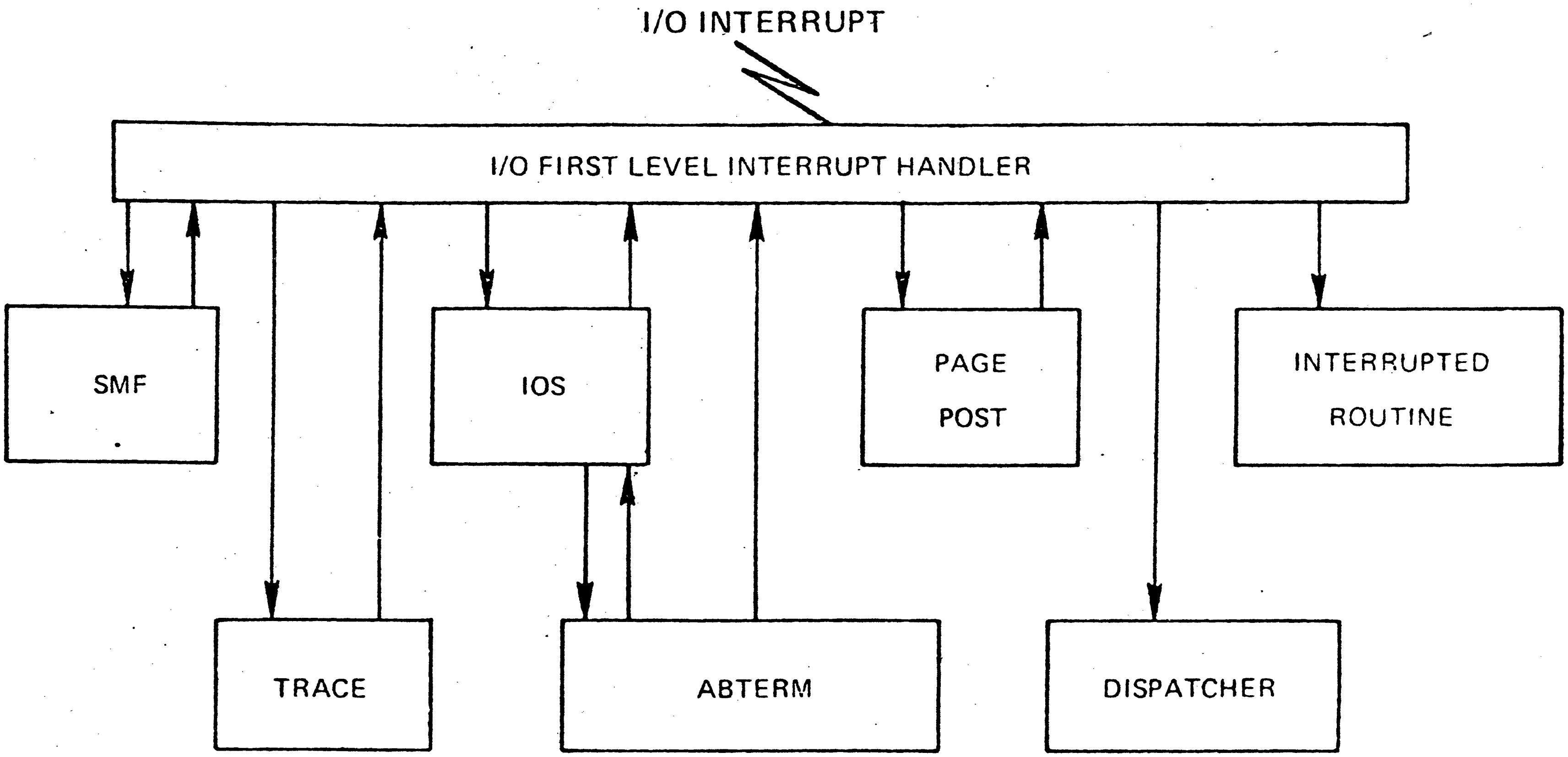
SVC
FLIH


SAVE
–REGISTERS
–POTENTIAL
LOCK
SUPVR LOCK ON
YES
PAGING SUPVR
NO
REGS 10-1 IN TCB
REGS 2-9 AS ON ENTRY
YES
NO
BACK UP SVCOPSW AND SAVE IN RPSW
NEW= OLD
NO
YES
SET NEW TO 0
DISPATCHER
ADD SVC NO. TO RELO TABLE
ENTRY =0
YES
GENERATE ERROR CODE
RESTORE REGISTERS
ABEND
NO
T1 OR T2
NO
SUBTRACT NO. OF T1 AND T2
MULTIPLY SVC NO. BY 6 TO GET DISP.
ADD DISP TO BEGINNING OF T3 & T4.
YES
MULTIPLY SVC NO. BY 4 TO GET DISP.
ADD DISP. TO BEGINNING OF SVC TABLE
GET ROUTINE ADDRESS
ESA=0
NO
YES
STORE REGISTERS IN TCB
SVC SLIH
MOVE SVC LOCK TO CVTA1F1
SET T1 SWITCH
BRANCH TO SVC ROUTINE


SVC SLIH
TYPE 2
YES
NO
SET PSW ENABLE OR DISABLE
CREATE SVRB
SET PSW ENABLE OR DISABLE
CREATE SVRB
RESIDENT
YES
NO
GET TTR
SVC IN XSNT AREA
NO
FINCH
SET POTENTIAL LOCK
INITIALIZE REGISTERS
BUILD PSW
R14 ↑ SVC 3
LPSW

SVC 3 - EXIT


FROM FLIH
FROM SPIE
YES
RESET PIE RESET T1
RESTORE REGISTERS
LPSW
NO
SCB
YES
FREE SCBs
NO
MOVE REGISTERS INTO TCB - RESET TI SWITCH
IRB OR STRB
YES
HANDLE IQEs
DELETE ANY MINORS
LAST RB
NO
DEQUEUE AND FREE RB - FREE MODULE IF PRB
DISPATCHER
YES
TASK TERM PREVIOUSLY
YES
FREE SVRB
IQE
YES
SCHEDULE ETXR
NO
DEQUEUE TCB
POST ECB
DISPATCHER
NO
ANY SUBTASK
YES
ABEND A03
NO
JOBSTEP TASK
YES
NO
ENQ COUNT=0
NO
ABEND C03
YES
NORMAL ABEND


I/O INTERRUPT
I/O FIRST LEVEL INTERRUPT HANDLER
SMF
IOS
PAGE POST
INTERRUPTED ROUTINE
TRACE
ABTERM
DISPATCHER

SYSGEN MACROS – RESIDENT T3 & T4 SVCs

CTRLPROG OPTIONS = TRSVCTBL

TABLE CONTAINING RELATIVE TRACK ADDRESS OF ALL TRANSIENT SVCs IS TO BE STORED IN RESIDENT PORTION OF CONTROL PROGRAM.

COULD BE SPECIFIED EVEN IF NO T3 OR T4 SVCs WERE TO BE MADE RESIDENT

RESIDNT = TRSVC

T3 AND T4 SVCs ARE TO BE MADE RESIDENT IN PAGEABLE SUPERVISOR AREA.

SYS1.PARMLIB

IEARSV00 - CONTAINS NAMES OF SVCs TO BE RESIDENT AND PAGEABLE

IEARSV01 - CONTAINS NAMES OF SVCs TO BE RESIDENT AND FIXED

IPL - RESIDENT T3 & T4 SVCs

IEA101A SPECIFY SYSTEM AND/OR SET PARAMETERS

RSVC=XX   FOR PAGEABLE

IEARSVXX LIST IN SYS1.PARMLIB
WILL BE USED
DEFAULT - IEARSV00

RSVCF=XX   FOR FIXED

IEARSVXX LIST IN SYS1.PARMLIB
WILL BE USED
DEFAULT - IEARSV01

* SVC ON BOTH LISTS WILL BE PAGEABLE

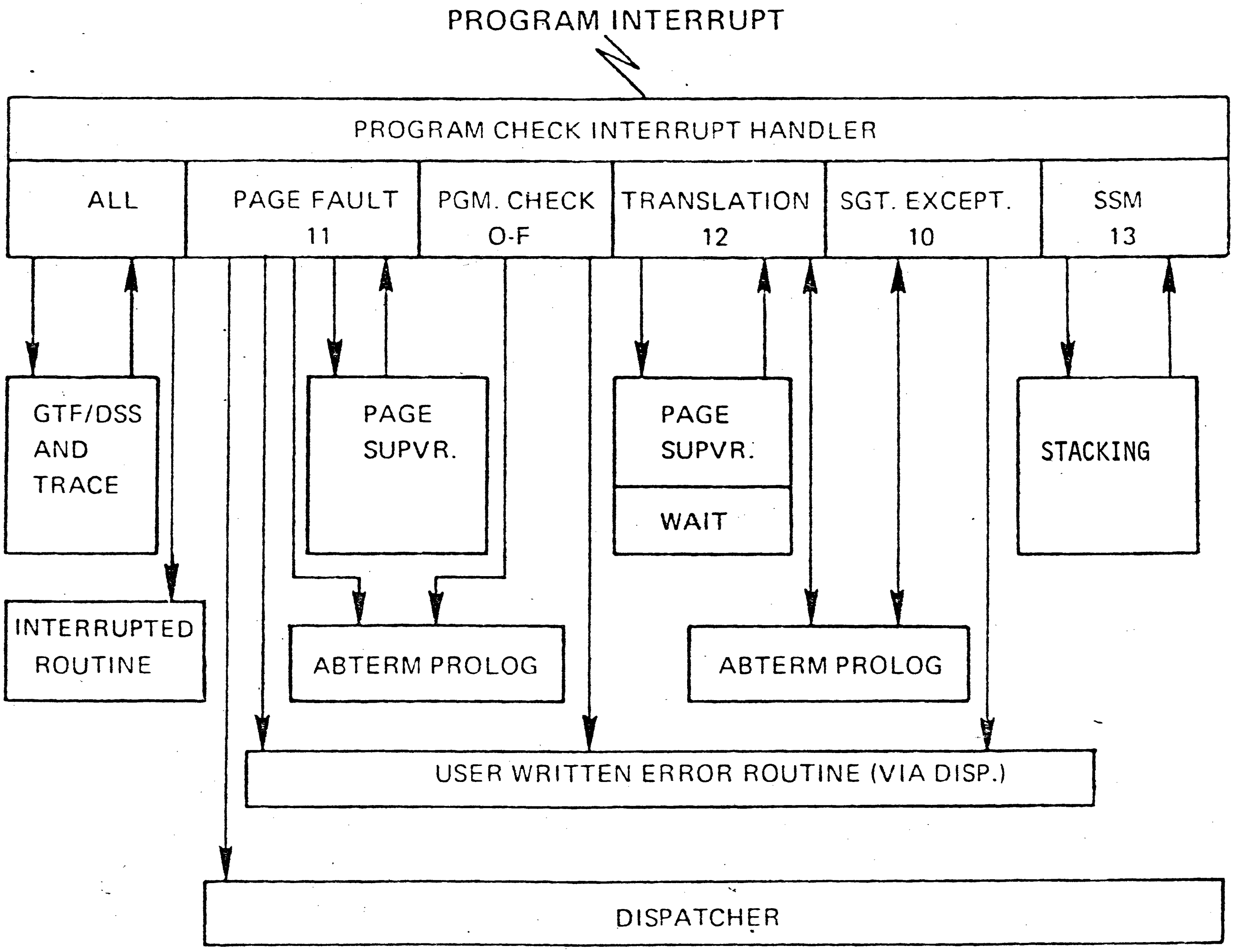
REGISTER CONTENTS UPON ENTRY TO A SVC ROUTINE

REGISTER	MEANING
3	COMMUNICATION VECTOR TABLE (CVT) POINTER
4	TASK CONTROL BLOCK (TCB) POINTER
5	SUPERVISOR REQUEST BLOCK (SVRB) POINTER FOR TYPES 2, 3 AND 4. LAST ACTIVE REQUEST BLOCK FOR TYPE 1 ROUTINES
14	CONTAINS RETURN ADDRESS
0,1,15	USED FOR PASSING INFORMATION BETWEEN ROUTINES . NOT RESTORED.

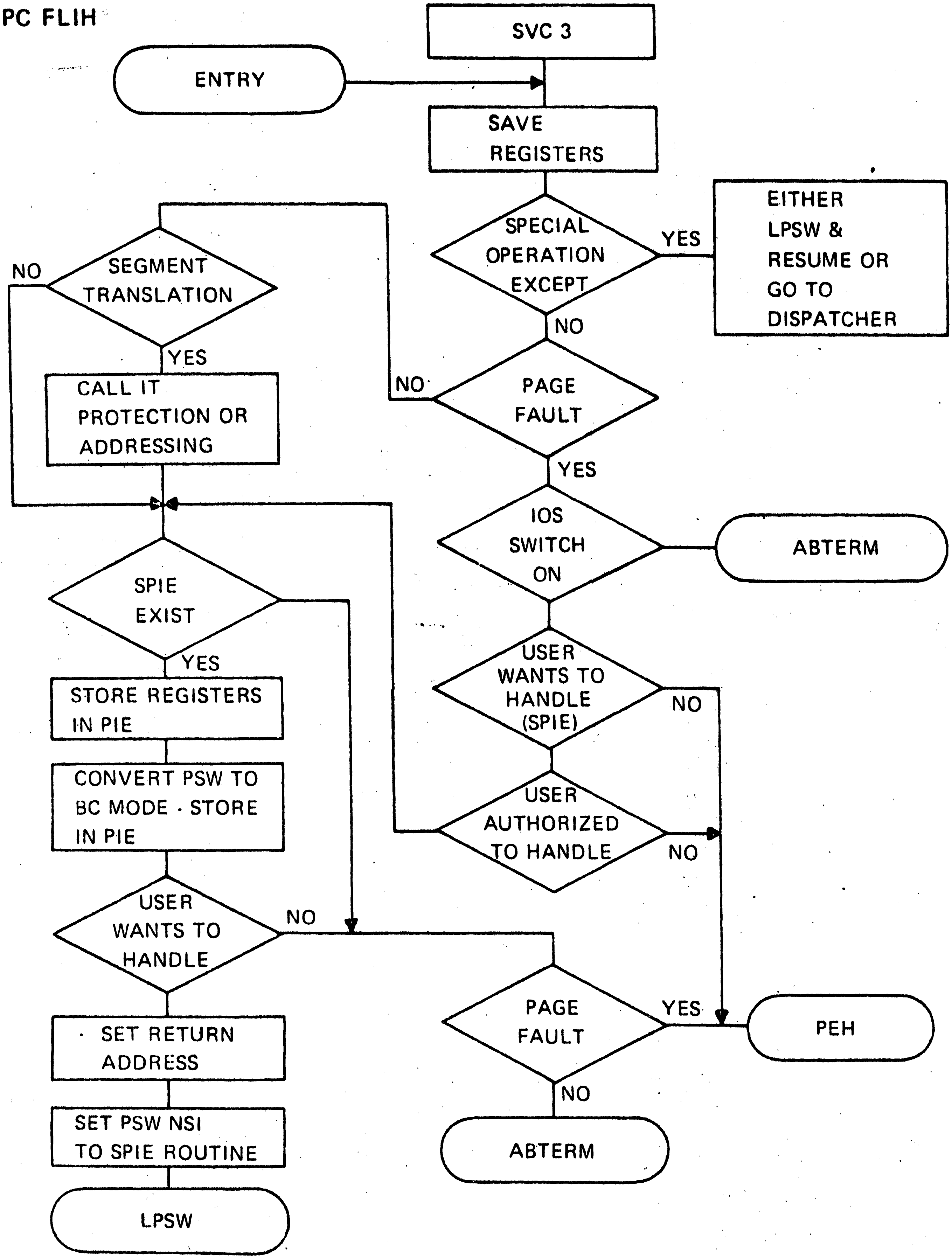
SVCs – PROGRAMMING CONVENTIONS

CONVENTIONS	TYPE 1	TYPE 2	TYPE 3	TYPE 4
PART OF RESIDENT CONTROL PROGRAM	YES	YES	NO	NO
SIZE OF ROUTINE	ANY	ANY	≤2048 BYTES	EACH LOAD MODULE ≤2048 BYTES
REENTERABLE ROUTINE	OPTIONAL, BUT MUST BE SERIALLY REUSABLE	YES	YES	YES
MAY ALLOW INTERRUPTIONS	NO	YES	YES	YES
ENTRY POINT	MUST BE THE FIRST BYTE OF THE ROUTINE OR LOAD MODULE, AND MUST BE ON A DOUBLEWORD BOUNDARY			
NUMBER OF ROUTINE	NUMBERS ASSIGNED TO YOUR SVC ROUTINES SHOULD BE IN DESCENDING ORDER FROM 255 THROUGH 200			
NAME OF ROUTINE	IGCnnn	IGCnnn	IGC00nnn	IGCssnnn
REGISTERS CONTENTS AT ENTRY TIME	REGISTERS 3, 4, 5, and 14 CONTAIN COMMUNICATION POINTERS; REGISTERS 0, 1, and 15 ARE PARAMETER REGISTERS			
MAY CONTAIN RELOCATABLE DATA	YES	YES	NO	NO
CAN SUPERVISOR REQUEST BLOCK (SVRB) BE EXTENDED	NOT APPLICABLE	YES	YES	YES
MAY ISSUE WAIT MACRO INSTRUCTION	NO	YES	YES	YES
MAY ISSUE XCTL MACRO INSTRUCTION	NO	NO	NO	YES
MAY PASS CONTROL TO WHAT OTHER TYPES OF SVC ROUTINES	NONE	ANY	ANY	ANY
TYPE OF LINKAGE WITH OTHER SVC ROUTINES	NOT APPLICABLE	ISSUE SUPERVISOR CALL (SVC) INSTRUCTION		
EXIT FROM SVC ROUTINE	BRANCH USING RETURN REGISTER 14			
METHOD OF ABNORMAL TERMINATION	USE RESIDENT ABNORMAL TERMINATION ROUTINE	USE ABEND MACRO INSTRUCTION OR RESIDENT ABNORMAL TERMINATION ROUTINE		

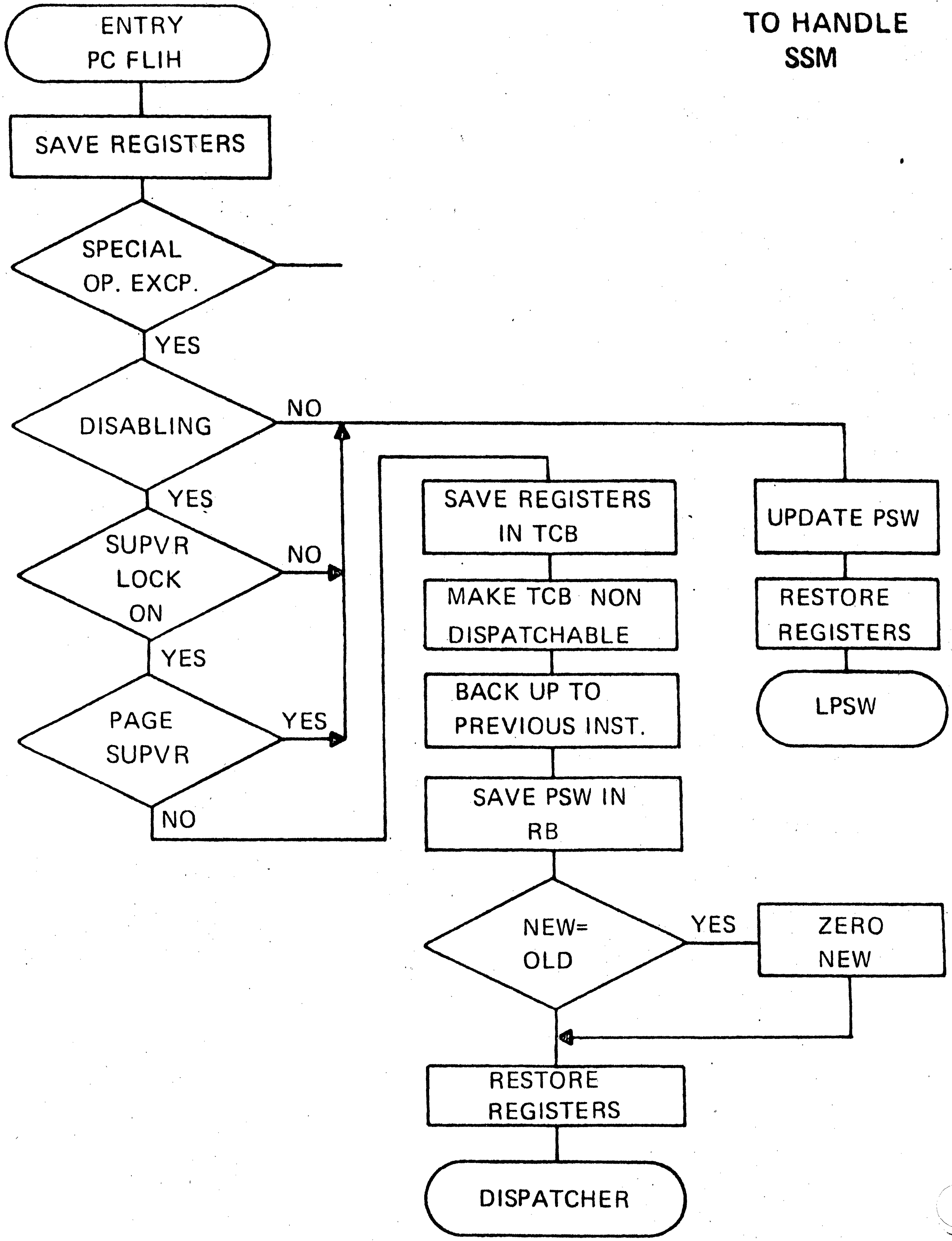
```
TYPE 1 AND TYPE 2
                                                              CC72

      RESMODS     MEMBERS=(NAME 1, NAME 2, . . .),             X
                  PDS=SYS1.NAME
                                                      RESIDENT IN NUCLEUS
                            ( Da )
      SVCTABLE    SVC-nnn- {      } - Sb
                            ( Ea )

TYPE 3 AND TYPE 4
                                                              CC72

      SVCLIB      MEMBERS=(NAME 1, NAME 2, . . .),             X
                                                        TRANSIENT
                  PDS=SYS1.NAME,                               X
                  RESIDNT=(NAMEA,. . .),                       X
                                                        RESIDENT, FIXED
                  VIRTUAL=(NAMEX, . . .)                RESIDENT, PAGEABLE

                            ( Da )
      SVCTABLE    SVC-nnn- {      } - Sb
                            ( Ea )
```

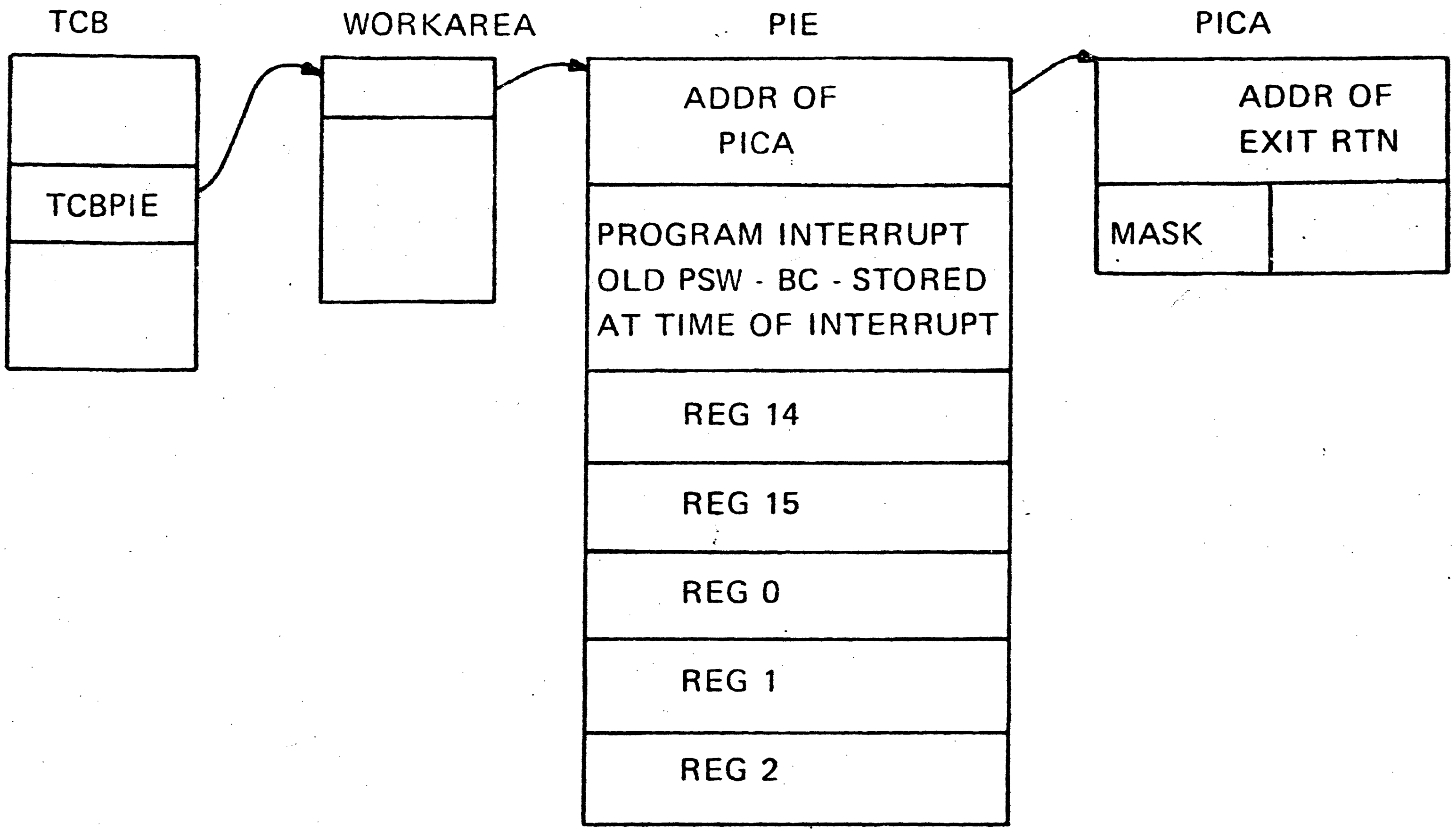

PROGRAM INTERRUPT
PROGRAM CHECK INTERRUPT HANDLER
ALL
PAGE FAULT 11
PGM. CHECK O-F
TRANSLATION 12
SGT. EXCEPT. 10
SSM 13
GTF/DSS AND TRACE
PAGE SUPVR.
PAGE SUPVR.
WAIT
STACKING
INTERRUPTED ROUTINE
ABTERM PROLOG
ABTERM PROLOG
USER WRITTEN ERROR ROUTINE (VIA DISP.)
DISPATCHER

PC FLIH


SVC 3
ENTRY
SAVE REGISTERS
SPECIAL OPERATION EXCEPT
YES
EITHER LPSW & RESUME OR GO TO DISPATCHER
NO
PAGE FAULT
NO
SEGMENT TRANSLATION
NO
YES
CALL IT PROTECTION OR ADDRESSING
YES
IOS SWITCH ON
ABTERM
SPIE EXIST
YES
STORE REGISTERS IN PIE
CONVERT PSW TO BC MODE - STORE IN PIE
USER WANTS TO HANDLE (SPIE)
NO
USER AUTHORIZED TO HANDLE
NO
USER WANTS TO HANDLE
NO
PAGE FAULT
YES
PEH
NO
ABTERM
SET RETURN ADDRESS
SET PSW NSI TO SPIE ROUTINE
LPSW

TO HANDLE SSM

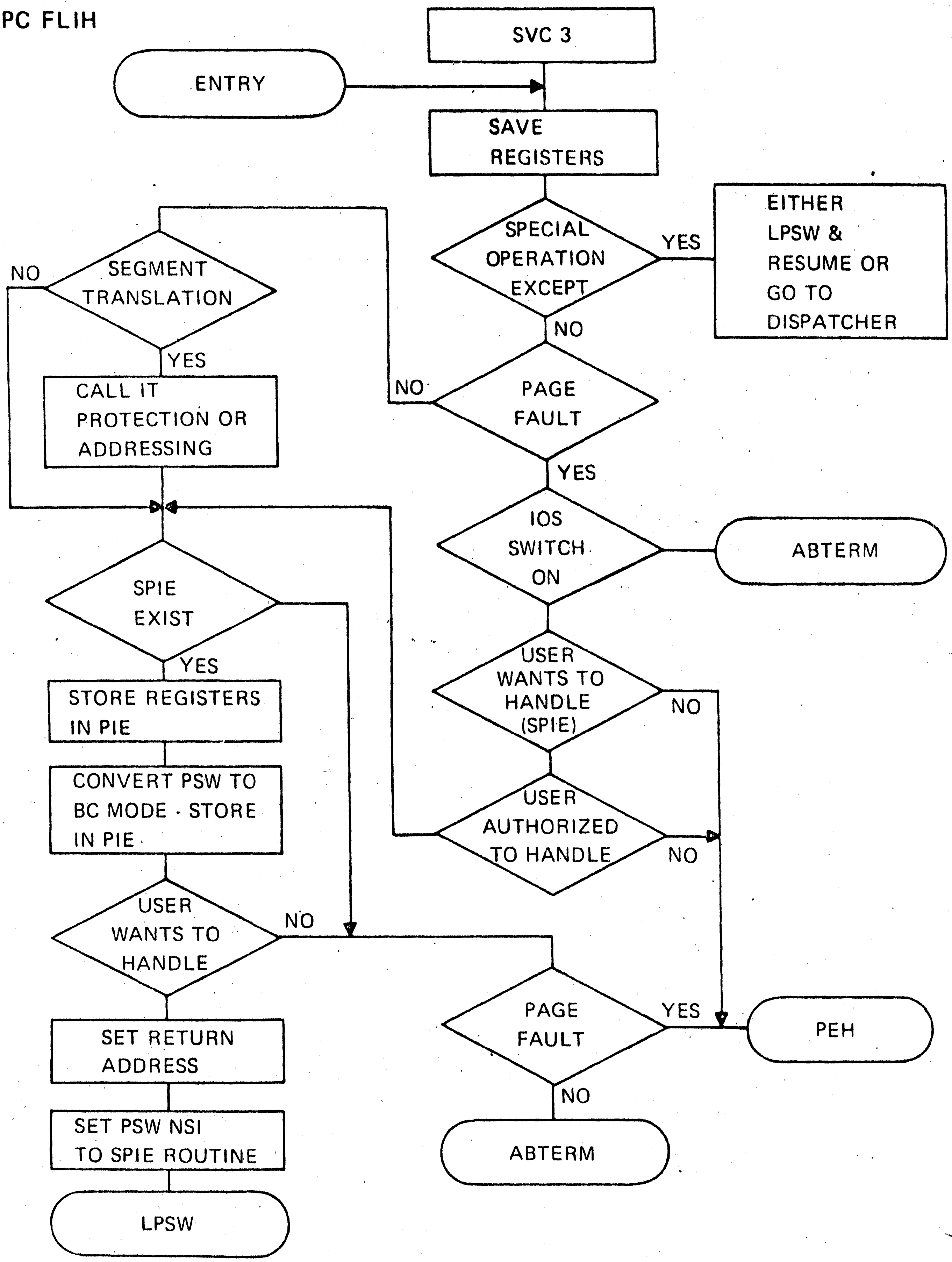


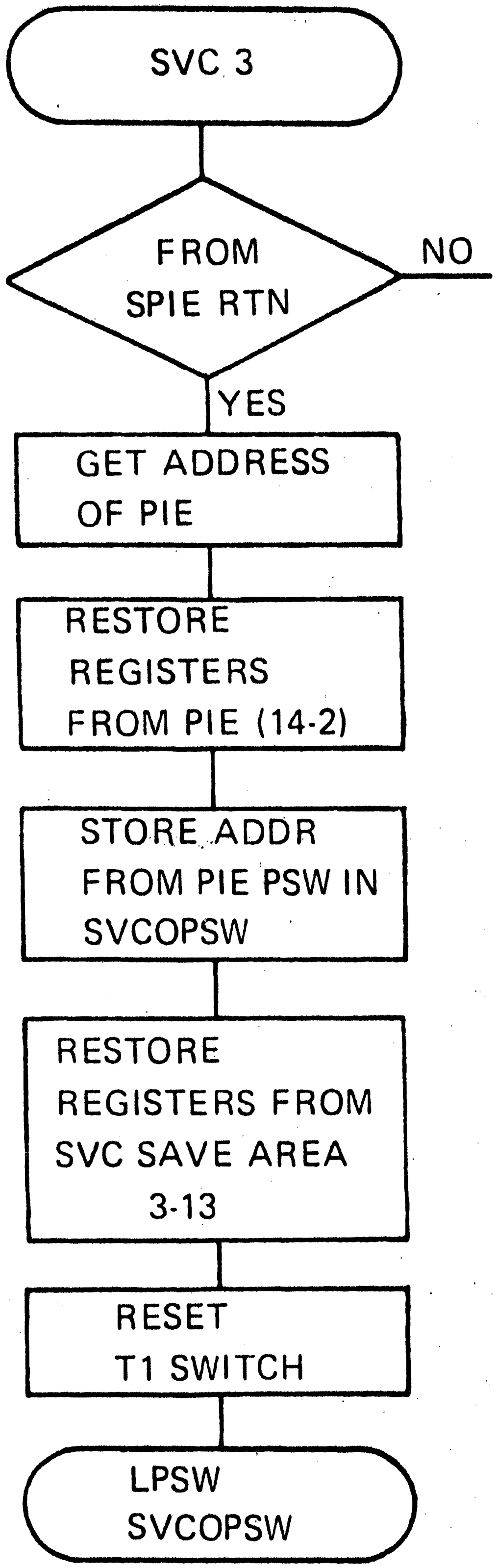


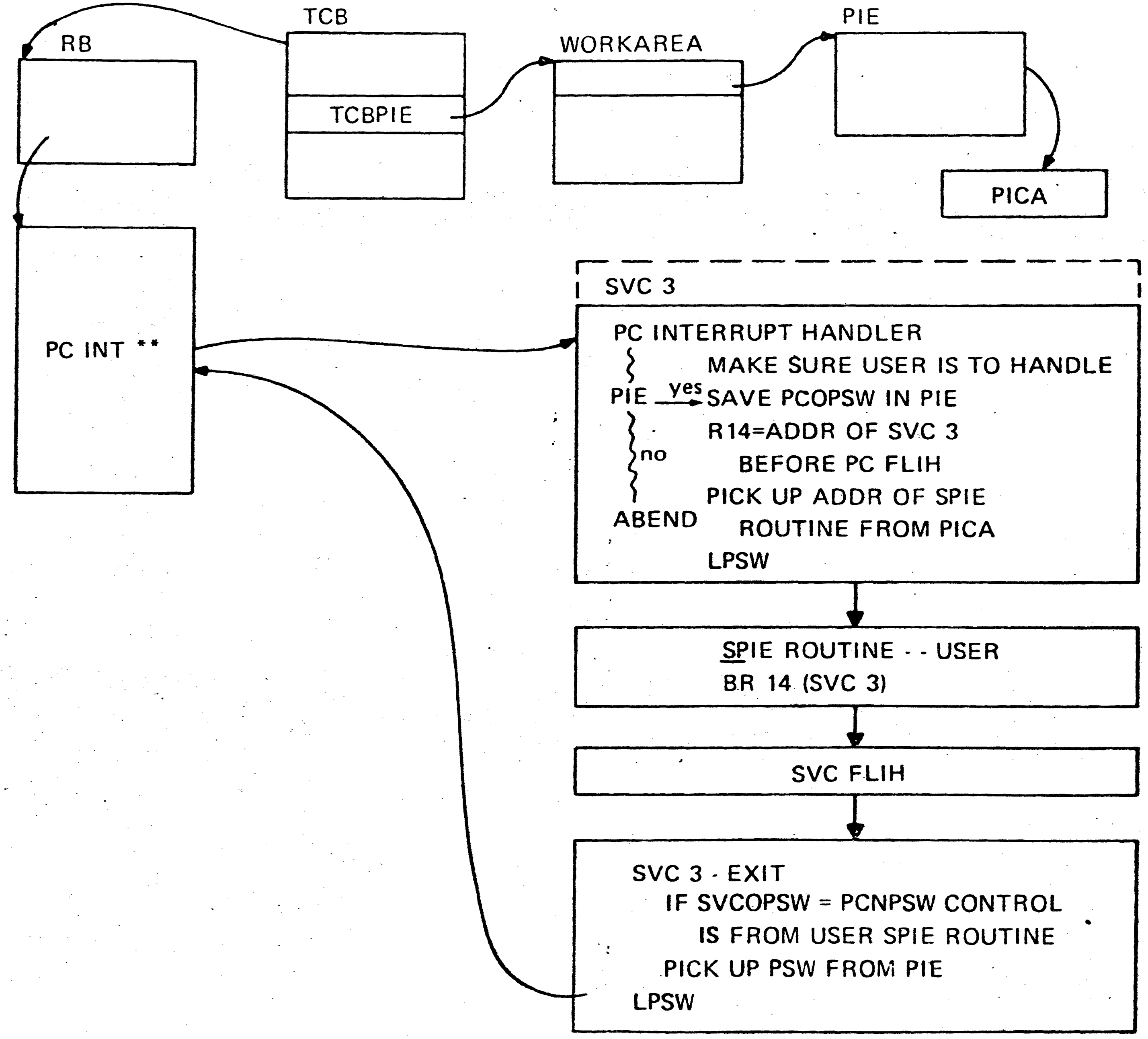
ON ENTRY TO SPIE EXIT ROUTINE

REG 0 - CONTROL PROGRAM INFORMATION
REG 1 - ADDRESS OF PIE
REG 2-12 - SAME AS WHEN INTERRUPT OCCURRED
REG 13 - ADDRESS OF MAIN PROGRAM SAVE AREA
DO NOT USE THIS SAVE AREA
REG 14 - RETURN ADDRESS
REG 15 - ENTRY POINT TO SPIE EXIT ROUTINE

PC FLIH


SVC 3
ENTRY
SAVE REGISTERS
SPECIAL OPERATION EXCEPT
YES
EITHER LPSW & RESUME OR GO TO DISPATCHER
NO
SEGMENT TRANSLATION
NO
YES
CALL IT PROTECTION OR ADDRESSING
NO
PAGE FAULT
YES
IOS SWITCH ON
ABTERM
SPIE EXIST
YES
USER WANTS TO HANDLE (SPIE)
NO
STORE REGISTERS IN PIE
CONVERT PSW TO BC MODE - STORE IN PIE
USER AUTHORIZED TO HANDLE
NO
USER WANTS TO HANDLE
NO
PAGE FAULT
YES
PEH
SET RETURN ADDRESS
NO
ABTERM
SET PSW NSI TO SPIE ROUTINE
LPSW


SVC 3
FROM SPIE RTN
NO
YES
GET ADDRESS OF PIE
RESTORE REGISTERS FROM PIE (14-2)
STORE ADDR FROM PIE PSW IN SVCOPSW
RESTORE REGISTERS FROM SVC SAVE AREA 3-13
RESET T1 SWITCH
LPSW SVCOPSW



TCB
RB
WORKAREA
PIE
TCBPIE
PICA
PC INT **
SVC 3
PC INTERRUPT HANDLER
MAKE SURE USER IS TO HANDLE
PIE
yes
SAVE PCOPSW IN PIE
R14=ADDR OF SVC 3
BEFORE PC FLIH
no
PICK UP ADDR OF SPIE
ROUTINE FROM PICA
ABEND
LPSW
SPIE ROUTINE - - USER
BR 14 (SVC 3)
SVC FLIH
SVC 3 - EXIT
IF SVCOPSW = PCNPSW CONTROL
IS FROM USER SPIE ROUTINE
PICK UP PSW FROM PIE
LPSW

LOADING USI

IBM

SYSTEMS PROGRAMMING

Student Materials

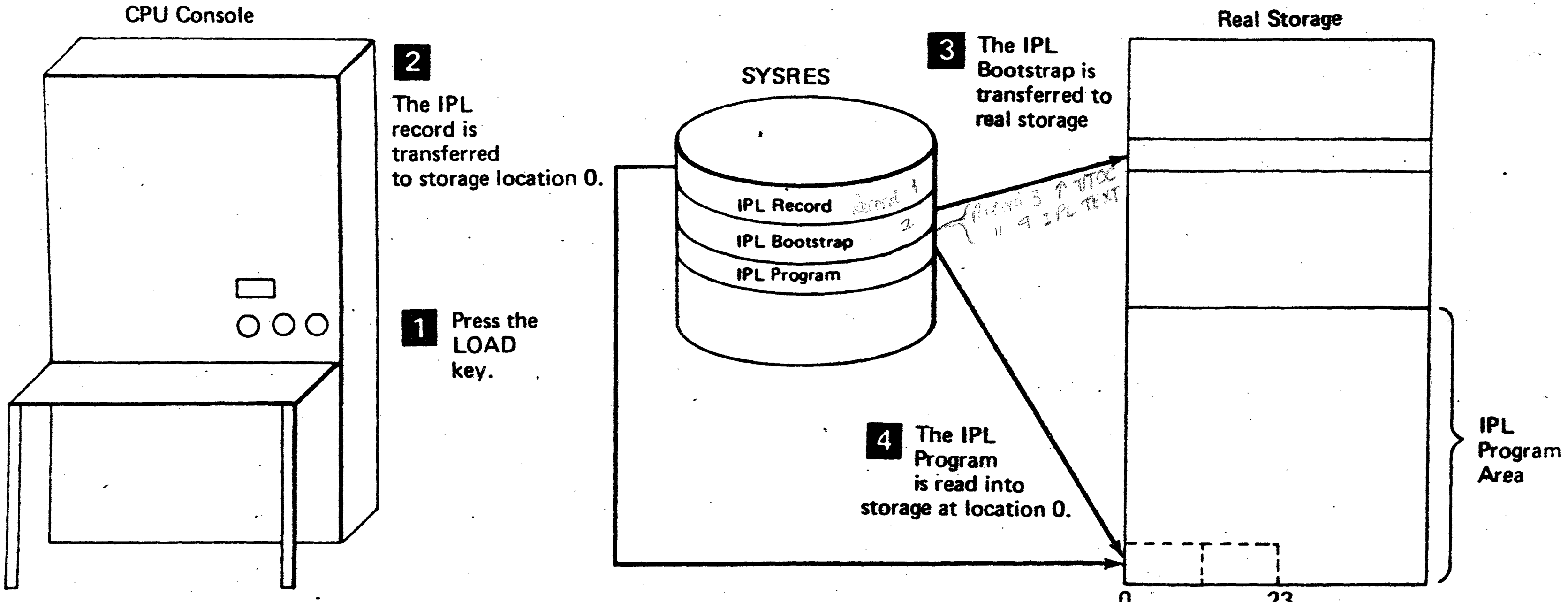
This material was produced for Educational purposes only. Changes may obsolete part or all of this publication. No responsibility is assumed for any inaccuracies that occur.

Los Angeles Advanced Education Center
3424 WILSHIRE BOULEVARD • LOS ANGELES, CALIF. 90010

LOADING THE CONTROL PROGRAM

REFERENCES:

OS/VS1 IPL and NIP Logic	SY24-5160
OS/VS1 Message Library: System Messages	GC38-1001
IBM System/370 Principles of Operation	GA22-7000
OS/VS1 Job Management Logic	SY24-5161



1 Press the LOAD key.

2 The IPL record is transferred to storage location 0.

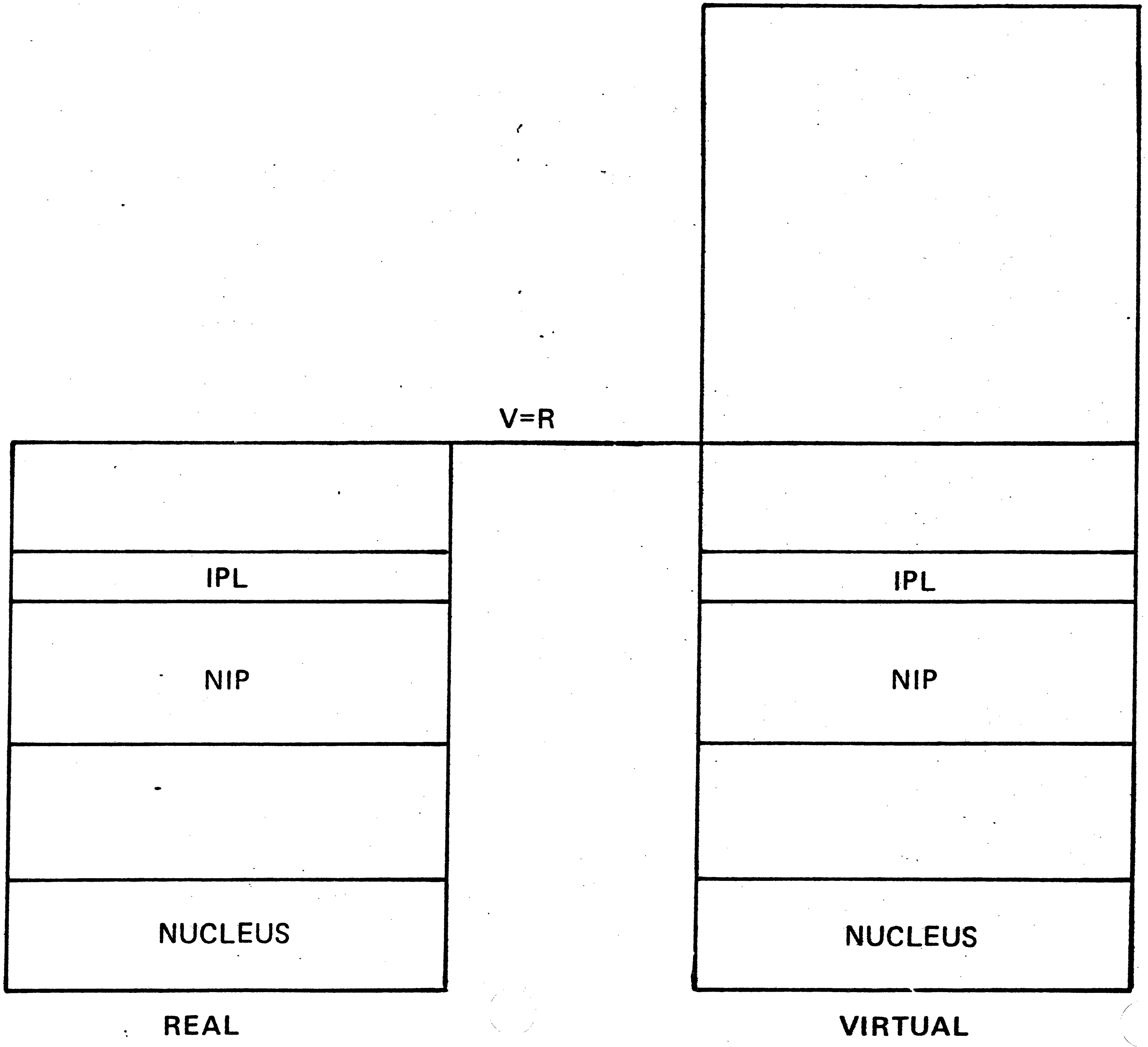
3 The IPL Bootstrap is transferred to real storage

4 The IPL Program is read into storage at location 0.

5 The operator may select an alternate nucleus or limit the size of apparent storage.

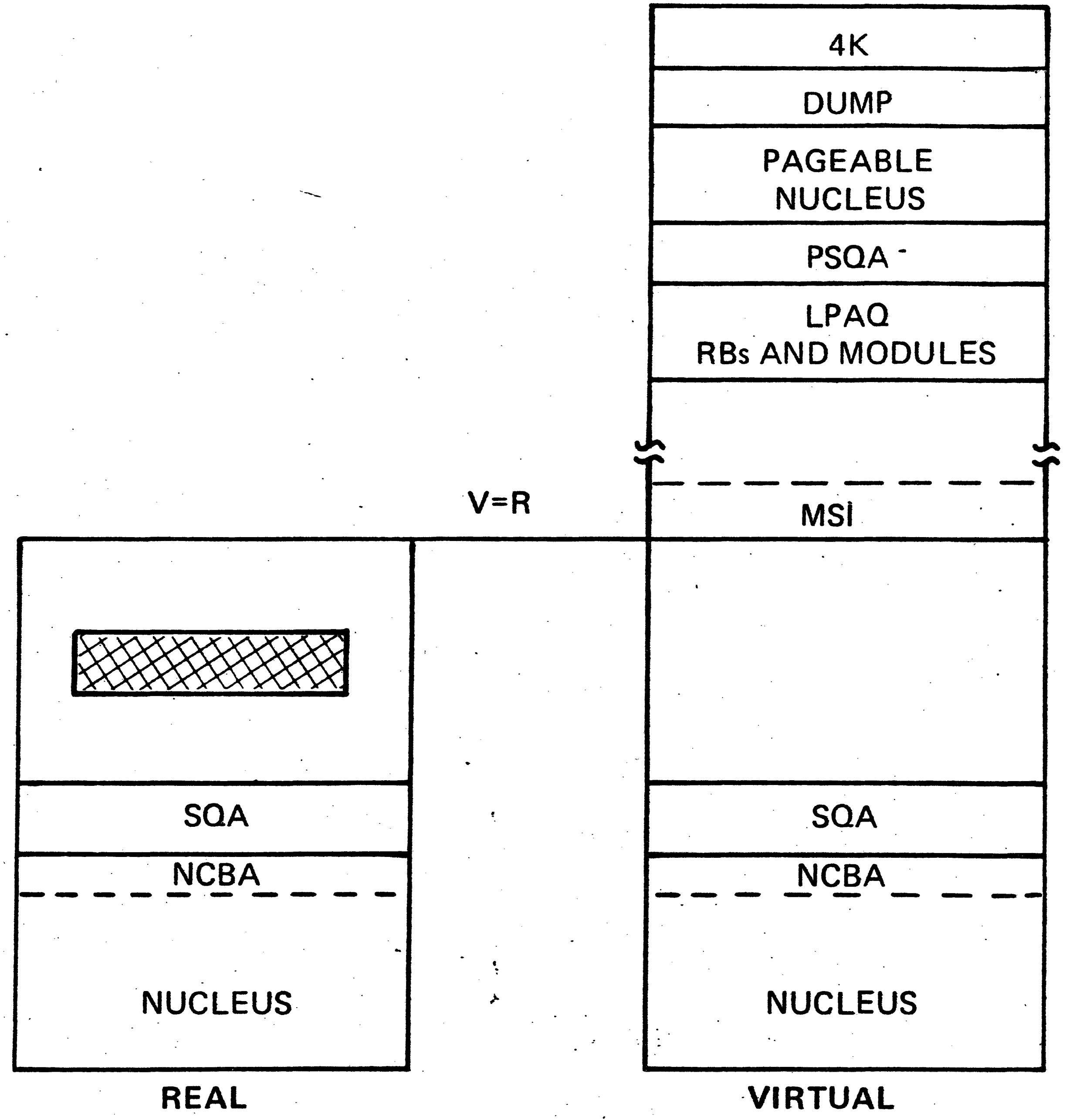
IPL FUNCTIONS

- CLEAR STORAGE AND ESTABLISH SIZE
- FIND SELECTED NUCLEUS
- RELOCATE IPL PROGRAM
- LOAD NIP
- LOAD NUCLEUS
- PASS CONTROL TO NIP


V=R
IPL
NIP
NUCLEUS
REAL
IPL
NIP
NUCLEUS
VIRTUAL

NIP FUNCTIONS

- ESTABLISH MASTER CONSOLE
- ESTABLISH VIRTUAL STORAGE
- SPLIT NUCLEUS INTO NONPAGEABLE AND PAGEABLE CSECTS
- ESTABLISH CVT
- BUILD SEGMENT TABLE
- BUILD TCB TABLE
- INITIALIZE DASD UCBs
- INITIALIZE SYSTEM DATA SETS
    - SYS1.LINKLIB
    - SYS1.SVCLIB
    - SYS1.LOGREC
    - SYS1.DUMP
- DETERMINE USER OPTIONS
- INITIALIZE PAGE MANAGEMENT
    - BUILD APCB
    - BUILD RSPT
    - INITIALIZE PAGE DATA SET
- ESTABLISH SQA AND PAGEABLE SQA
- BUILD LINK PACK AREA QUEUE
- PASS CONTROL TO MSI


4K
DUMP
PAGEABLE NUCLEUS
PSQA
LPAQ RBs AND MODULES
V=R
MSI
SQA
NCBA
NUCLEUS
SQA
NCBA
NUCLEUS
REAL
VIRTUAL

MSI FUNCTIONS

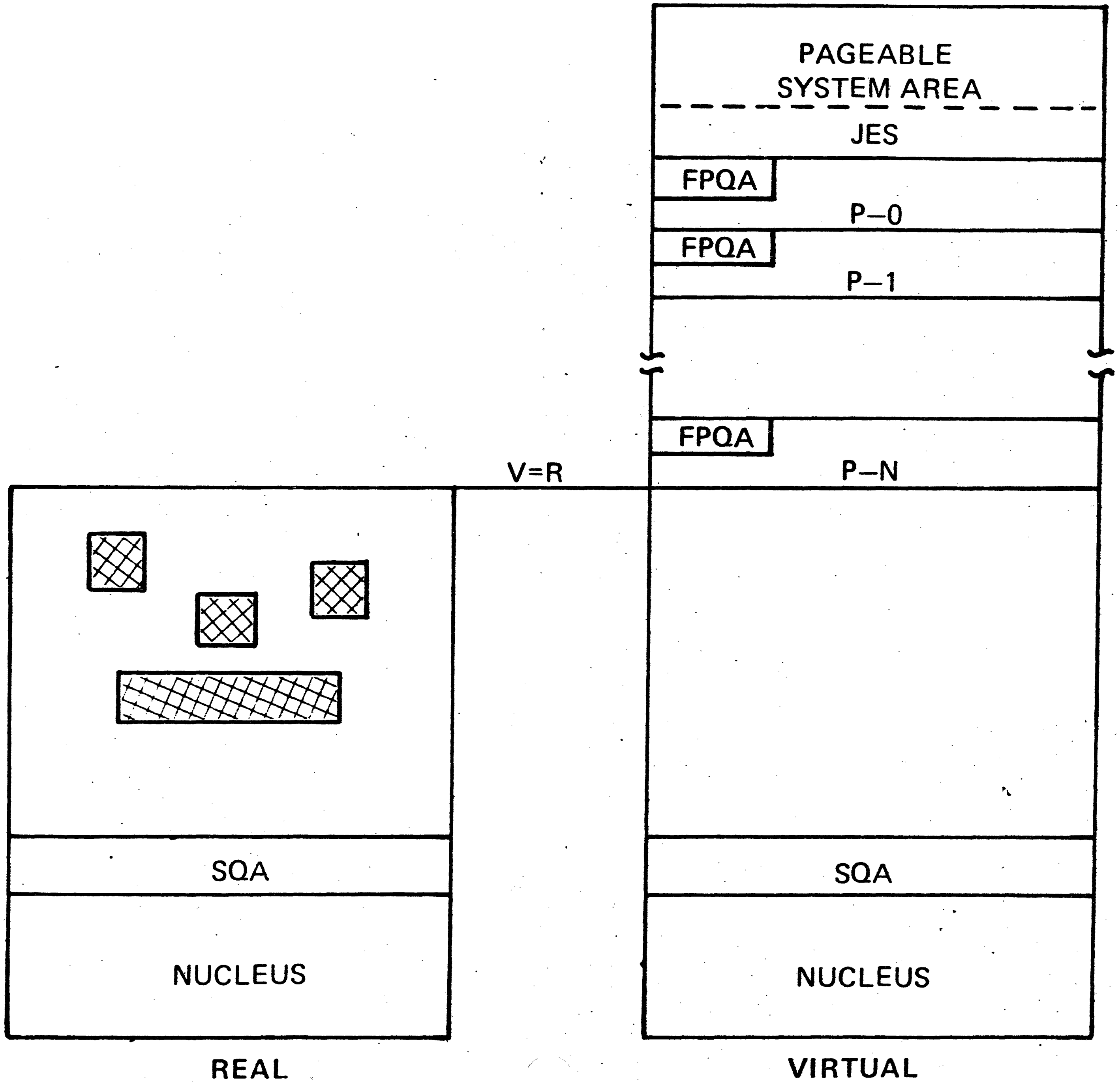
- INITIALIZES COMMUNICATIONS
- INITIALIZES JES
- DEFINES PARTITIONS

  BUILDS PAGE TABLES
  BUILDS RB FOR (IEFSD510) AND
  CONNECTS IT TO PARTITION TCB

- SET PARAMETERS

  Q
  SPOOL
  PROCLIB

- SYSTEM RESTART
- SETS VOLUME ATTRIBUTES
- IPL REASON
- INITIALIZE SMF
- PASS CONTROL TO MASTER SCHEDULAR


PAGEABLE SYSTEM AREA
JES
FPQA
P—0
FPQA
P—1
FPQA
P—N
V=R
SQA
NUCLEUS
SQA
NUCLEUS
REAL
VIRTUAL

TASK MGMT INTRO

IBM

SYSTEMS PROGRAMMING

Student Materials

This material was produced for Educational purposes only. Changes may obsolete part or all of this publication. No responsibility is assumed for any inaccuracies that occur.

Los Angeles Advanced Education Center
3424 WILSHIRE BOULEVARD • LOS ANGELES, CALIF. 90010

VS/1

INTRODUCTION TO TASK MANAGEMENT

REFERENCES:

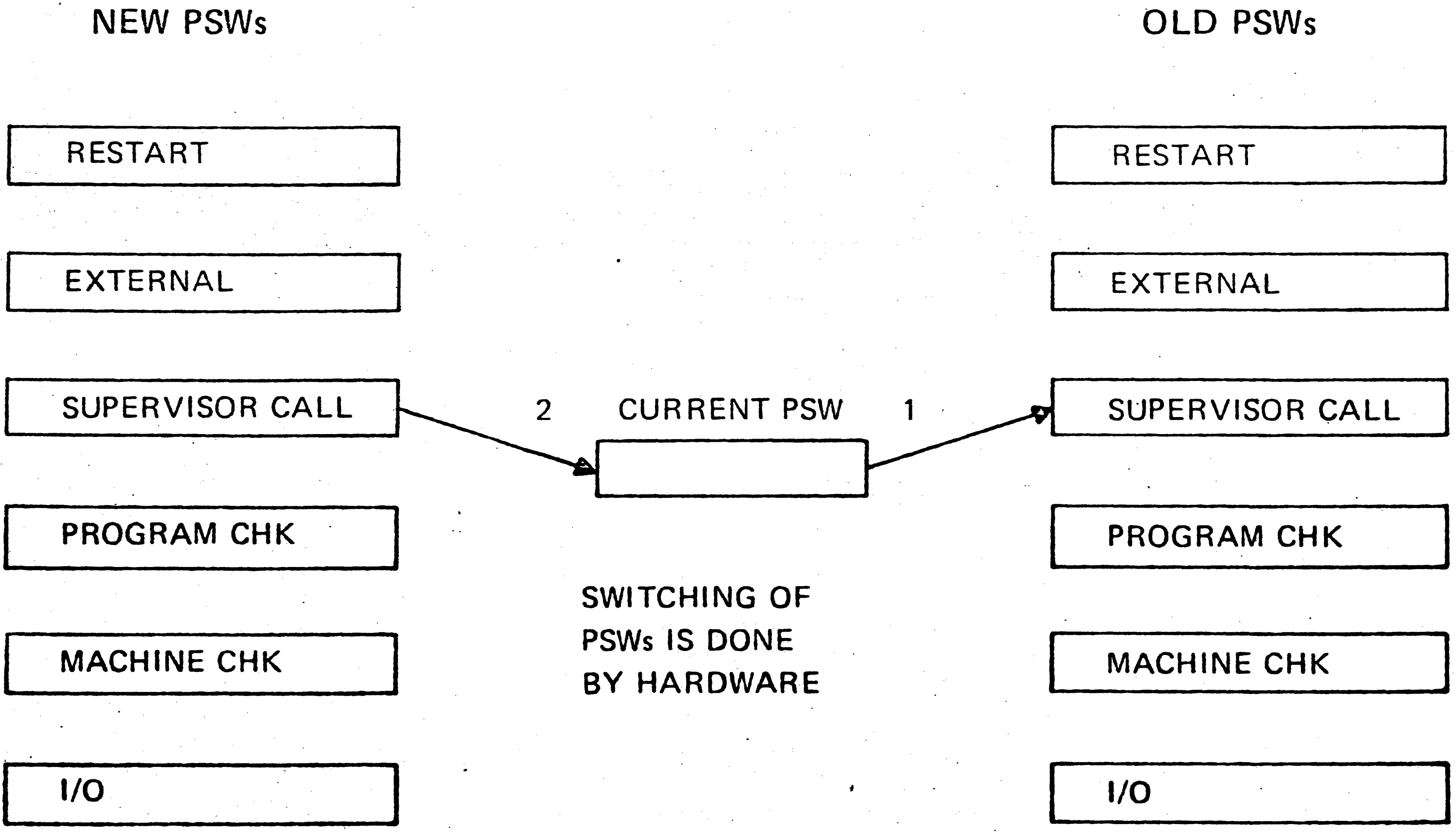
OS/VS1 Planning and Use Guide
OS/VS1 Features Supplement
OS/VS1 Debugging Guide
OS/VS1 System Data Areas

FUNCTIONS OF TASK MANAGEMENT

– HANDLE INTERRUPTS

– PERFORM SERVICES

– DISPATCH TASKS



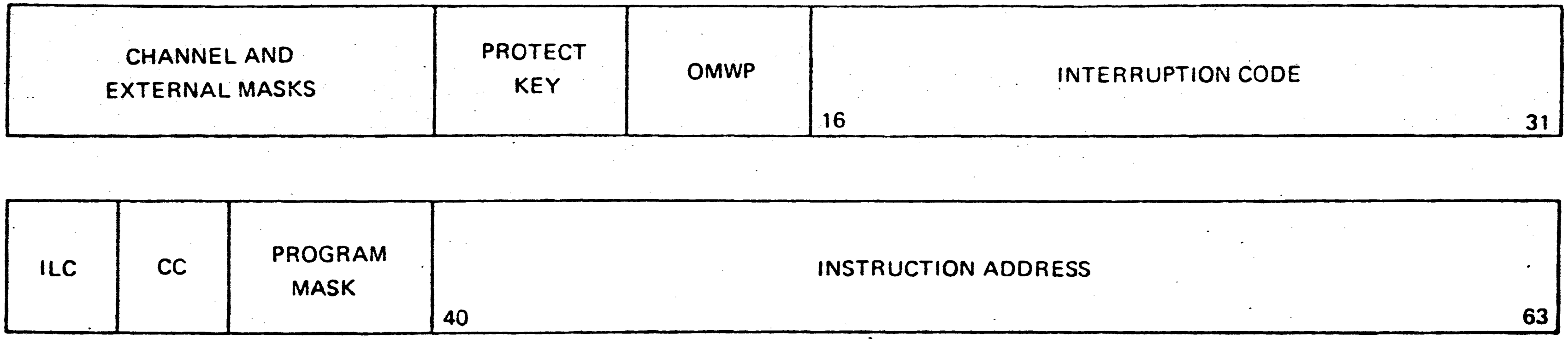
SWITCHING OF
PSWs IS DONE
BY HARDWARE

CONTAINS ADDRESS OF
ROUTINE WITHIN SUPERVISOR
TO HANDLE INTERRUPT

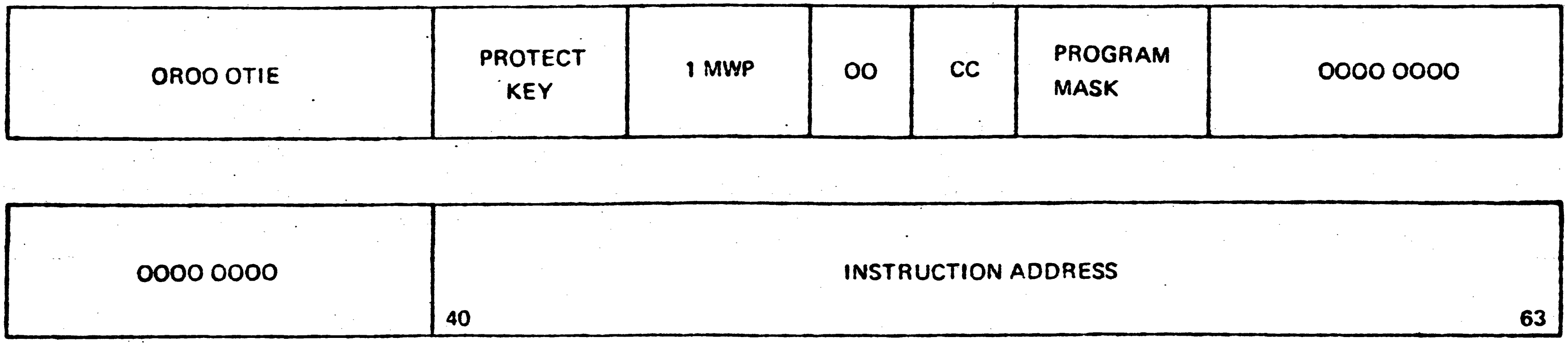
PROVIDES A SAVEAREA FOR
PSW THAT WAS CURRENT AT
TIME OF INTERRUPT

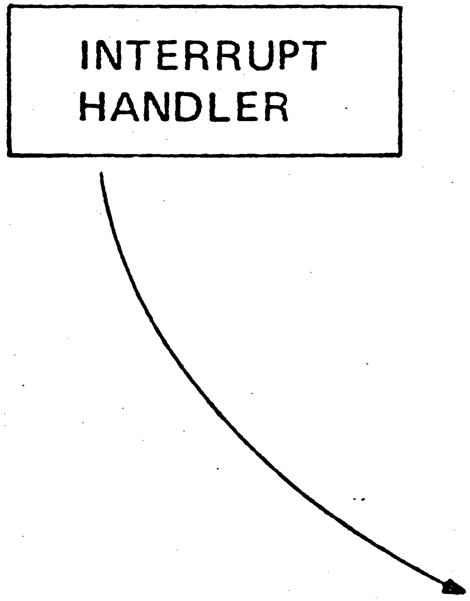
0	RESTART NEW PSW		RESTART OLD PSW			
10	CVT ADDRESS		EXTERNAL OLD PSW			
20	SVC OLD PSW		PROGRAM CHK OLD PSW			
30	MACHINE CHK OLD PSW		I/O OLD PSW			
40					4C CVT ADDRESS	
50			EXTERNAL NEW PSW			
60	SVC NEW PSW		PROGRAM CHK NEW PSW			
70	MACHINE CHK NEW PSW		I/O NEW PSW			
80		EXTERNAL INTERRUPT CODE	SVC ILC	SVC IC	PGM ILC	PGM IC
90	TRANSLATION EXCEPTION ADDRESS					

BC


CHANNEL AND EXTERNAL MASKS
PROTECT KEY
OMWP
INTERRUPTION CODE
16
31
ILC
CC
PROGRAM MASK
INSTRUCTION ADDRESS
40
63

EC

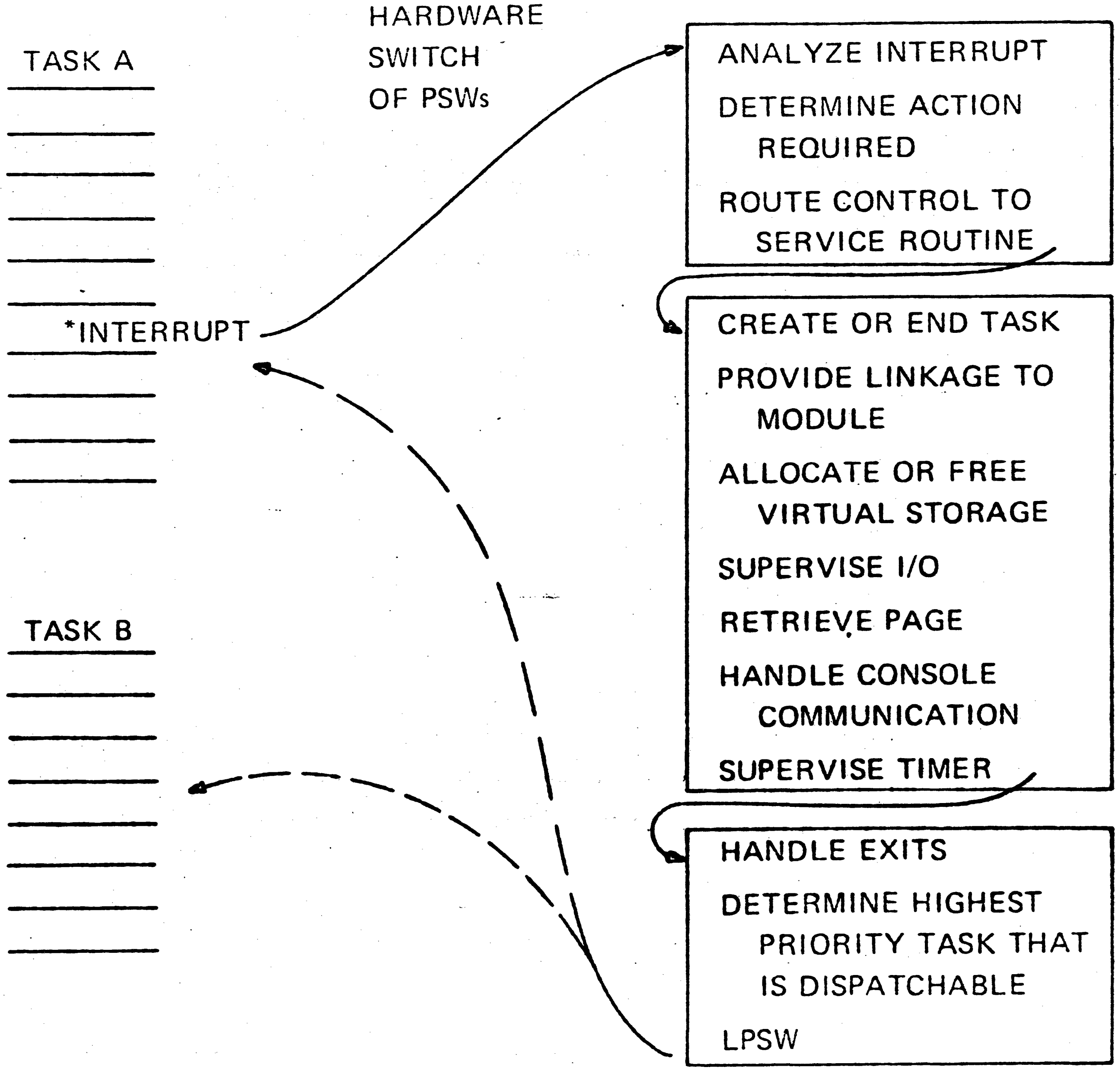

OROO OTIE
PROTECT KEY
1 MWP
OO
CC
PROGRAM MASK
0000 0000
0000 0000
INSTRUCTION ADDRESS
40
63



INTERRUPT HANDLER

1. TASK MANAGEMENT
2. TASK CREATION
3. TASK TERMINATION
4. CONTENTS MANAGEMENT
5. VIRTUAL STORAGE MANAGEMENT
6. TIMER MANAGEMENT
7. PAGE MANAGEMENT

SUPERVISOR FLOW



TASK A

HARDWARE
SWITCH
OF PSWs

*INTERRUPT

TASK B

ANALYZE INTERRUPT

DETERMINE ACTION
REQUIRED

ROUTE CONTROL TO
SERVICE ROUTINE

CREATE OR END TASK

PROVIDE LINKAGE TO
MODULE

ALLOCATE OR FREE
VIRTUAL STORAGE

SUPERVISE I/O

RETRIEVE PAGE

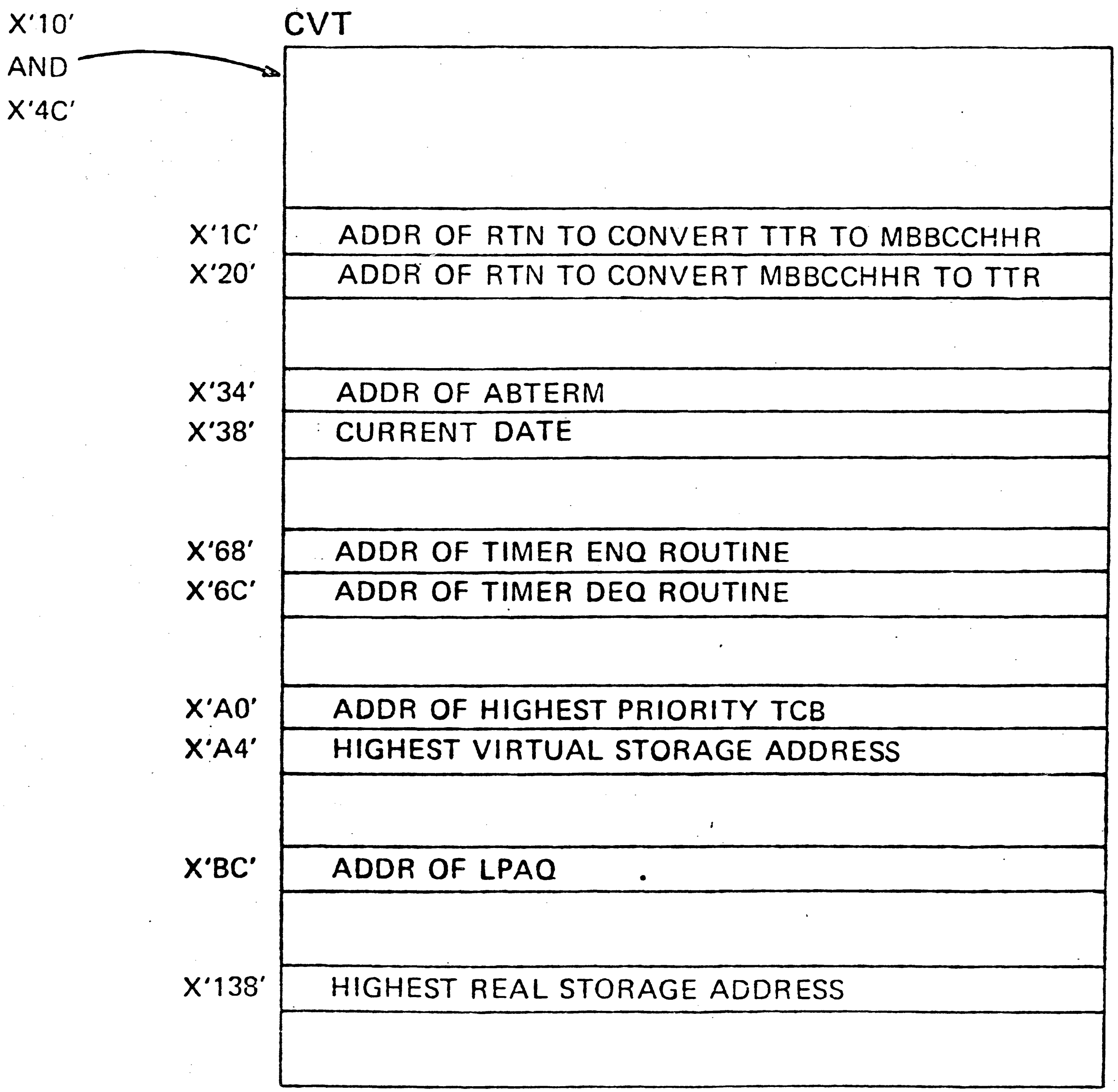
HANDLE CONSOLE
COMMUNICATION

SUPERVISE TIMER

HANDLE EXITS

DETERMINE HIGHEST
PRIORITY TASK THAT
IS DISPATCHABLE

LPSW



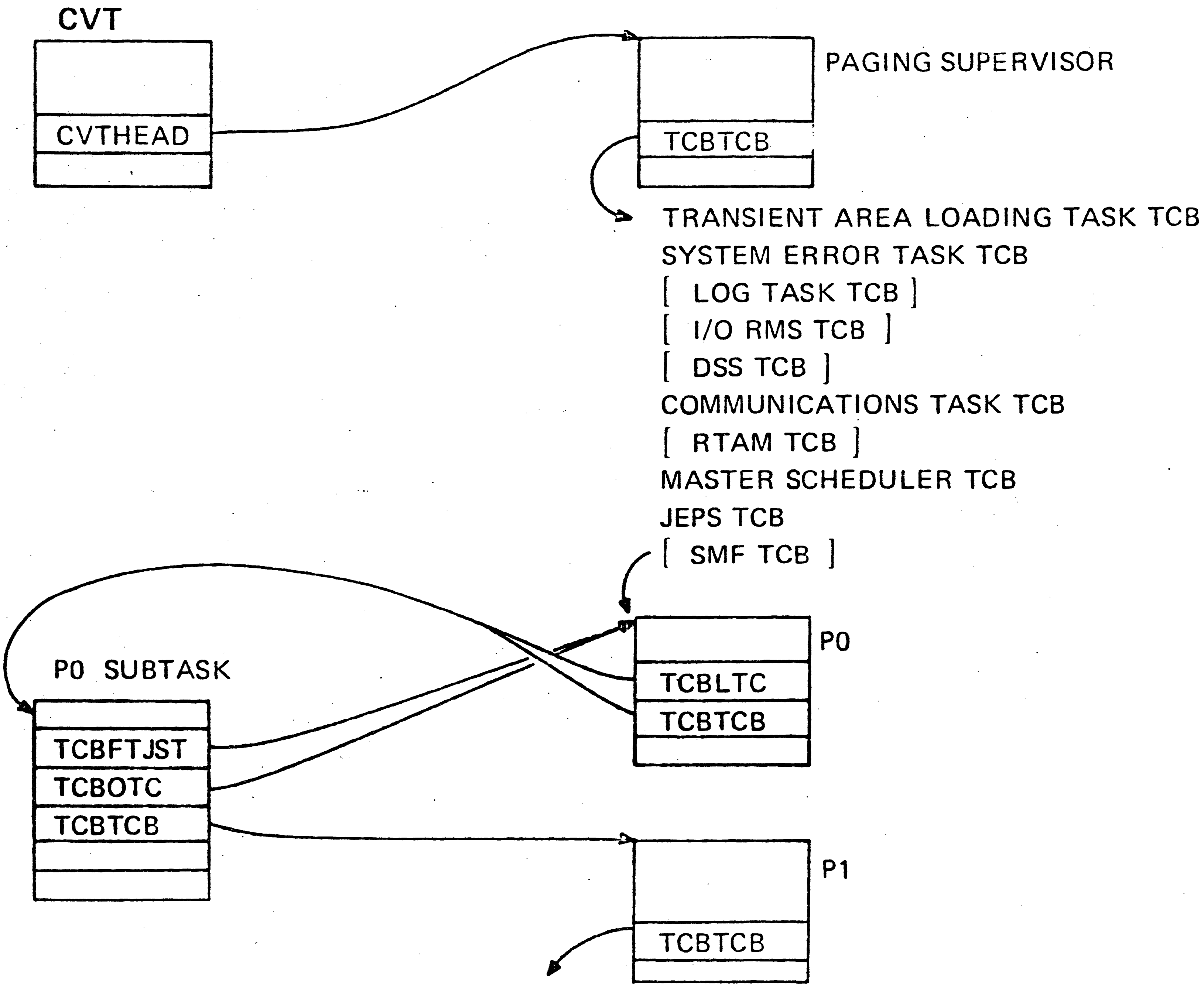
CVT

X'10'
AND
X'4C'

	CVT
X'1C'	ADDR OF RTN TO CONVERT TTR TO MBBCCHHR
X'20'	ADDR OF RTN TO CONVERT MBBCCHHR TO TTR
X'34'	ADDR OF ABTERM
X'38'	CURRENT DATE
X'68'	ADDR OF TIMER ENQ ROUTINE
X'6C'	ADDR OF TIMER DEQ ROUTINE
X'A0'	ADDR OF HIGHEST PRIORITY TCB
X'A4'	HIGHEST VIRTUAL STORAGE ADDRESS
X'BC'	ADDR OF LPAQ
X'138'	HIGHEST REAL STORAGE ADDRESS

TCB

0	ADDR OF ACTIVE RB Q					
8	ADDR OF DEB Q					
18	BOUNDARY BOX ADDR			PROTECT KEY		FLAGS
20		NO. OF ENQs	DISPATCH PRIORITY	ADDR OF LOADLIST		
28				ADDR OF JS TCB		
30	GENERAL REGISTER SAVE AREA					
70				ADDR OF NEXT TCB		
78				ADDR OF PIB		
98	LIMIT PRIORITY					


CVT
CVTHEAD
PAGING SUPERVISOR
TCBTCB
TRANSIENT AREA LOADING TASK TCB
SYSTEM ERROR TASK TCB
[LOG TASK TCB]
[I/O RMS TCB]
[DSS TCB]
COMMUNICATIONS TASK TCB
[RTAM TCB]
MASTER SCHEDULER TCB
JEPS TCB
[SMF TCB]
P0 SUBTASK
TCBFTJST
TCBOTC
TCBTCB
P0
TCBLTC
TCBTCB
P1
TCBTCB

LPRB

MAJOR RB ADDR	
LOAD LIST POINTERS	
^{0}MODULE NAME	
	↑ ENTRY POINT
^{10}RESUME PSW	
1C	↑ NEXT RB/TCB

LRB

LOAD LIST POINTERS	
^{0}MODULE NAME	
	↑ ENTRY POINT

PRB

^{0}MODULE NAME	
	↑ ENTRY POINT
^{10}RESUME PSW	
1C	↑ NEXT RB/TCB

FRB

LOAD LIST POINTERS
^{0}MODULE NAME
ADDR OF WLE
10

SVRB

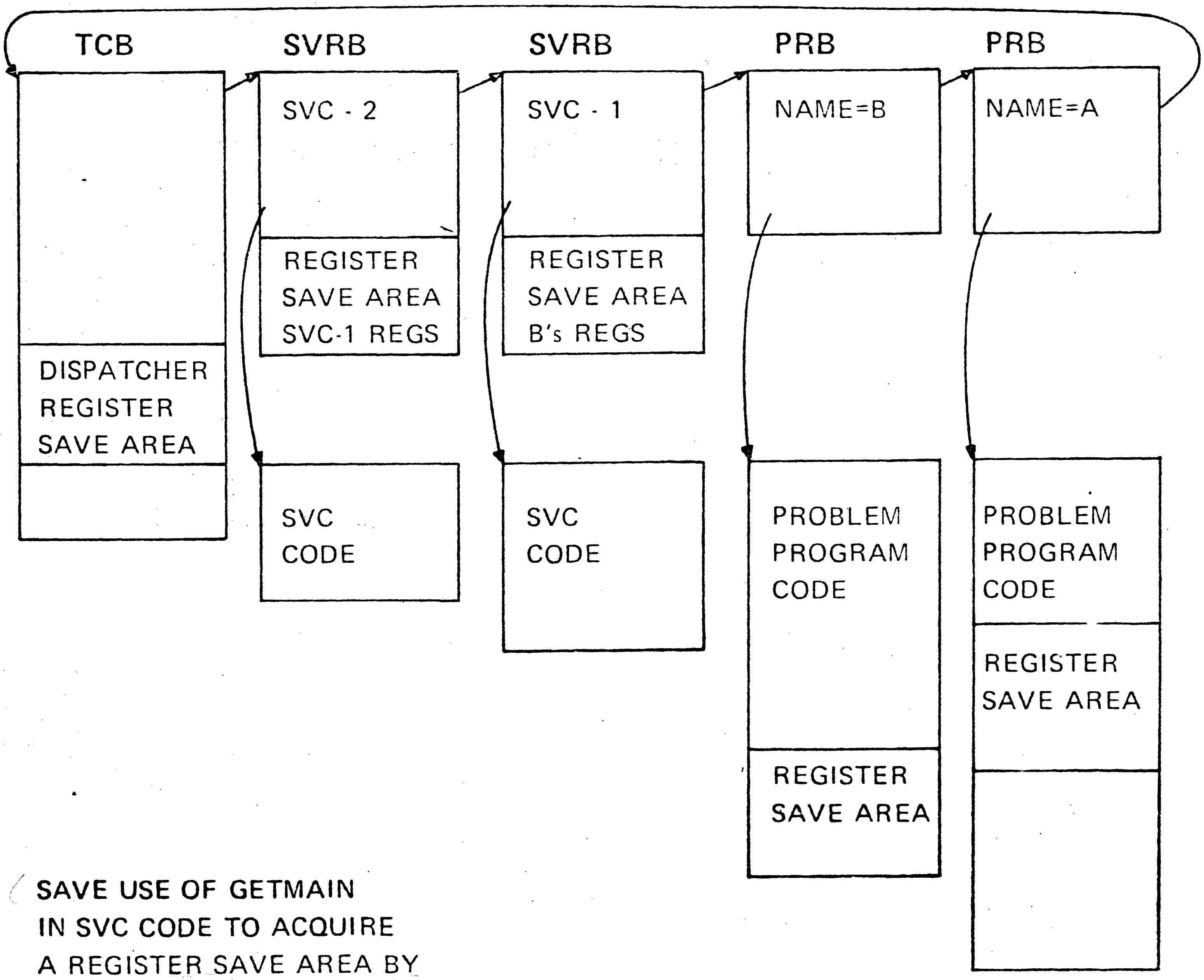
^{0}SVC NAME	
	↑ ENTRY POINT
^{10}RESUME PSW	
1C	↑ NEXT RB/TCB
REGISTER SAVE AREA	
EXTENDED SAVE AREA	

IRB

0	
	↑ ENTRY POINT
^{10}RESUME PSW	
1C	↑ NEXT RB/TCB
REGISTER SAVE AREA	

SIRB

^{0}ERROR ROUTINE NAME	
	↑ ENTRY POINT
^{10}RESUME PSW	
1C	↑ NEXT RB/TCB
REGISTER SAVE AREA	



TCB

DISPATCHER REGISTER SAVE AREA

SVRB

SVC - 2

REGISTER SAVE AREA SVC-1 REGS

SVC CODE

SVRB

SVC - 1

REGISTER SAVE AREA B's REGS

SVC CODE

PRB

NAME=B

PROBLEM PROGRAM CODE

REGISTER SAVE AREA

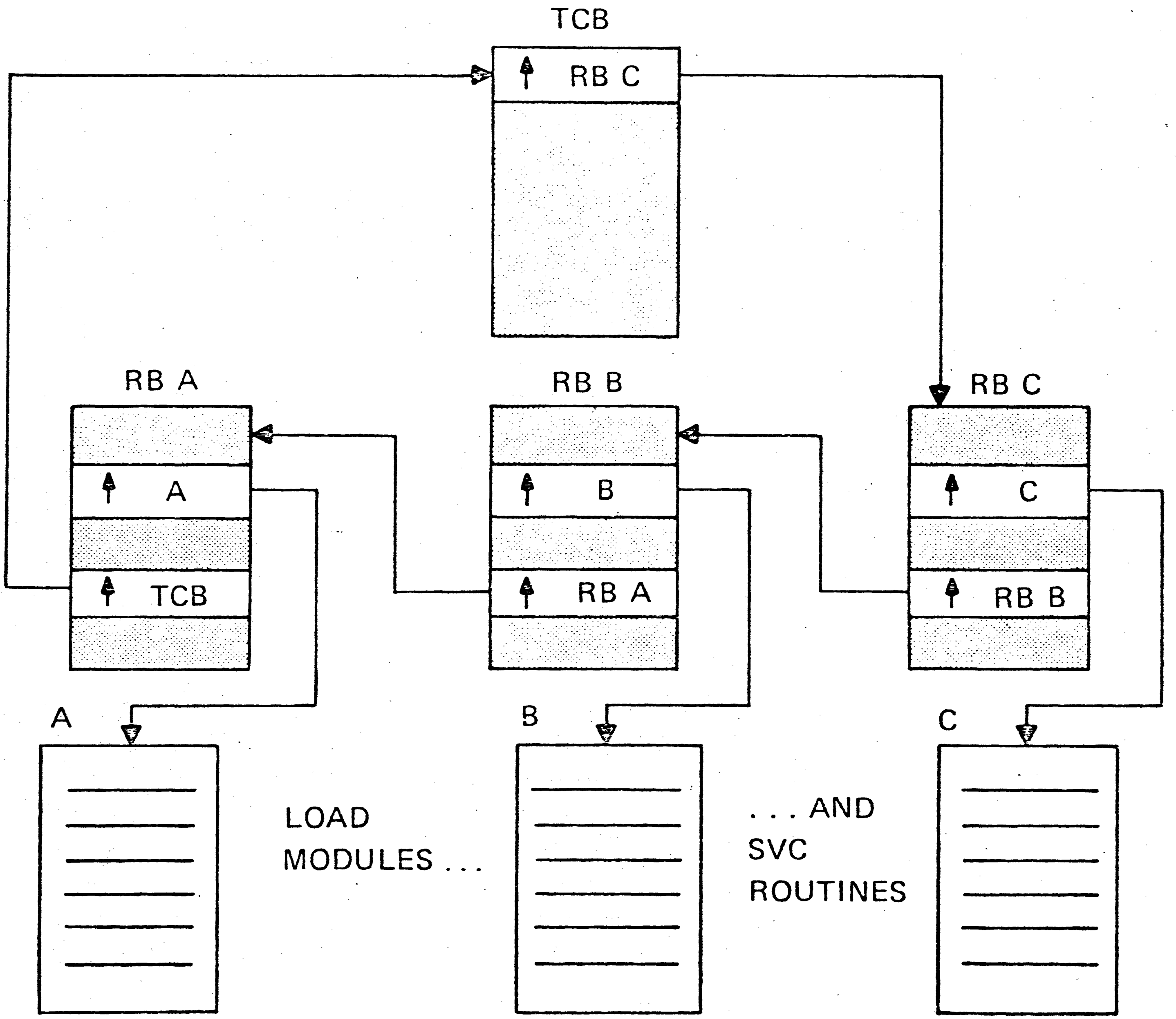
PRB

NAME=A

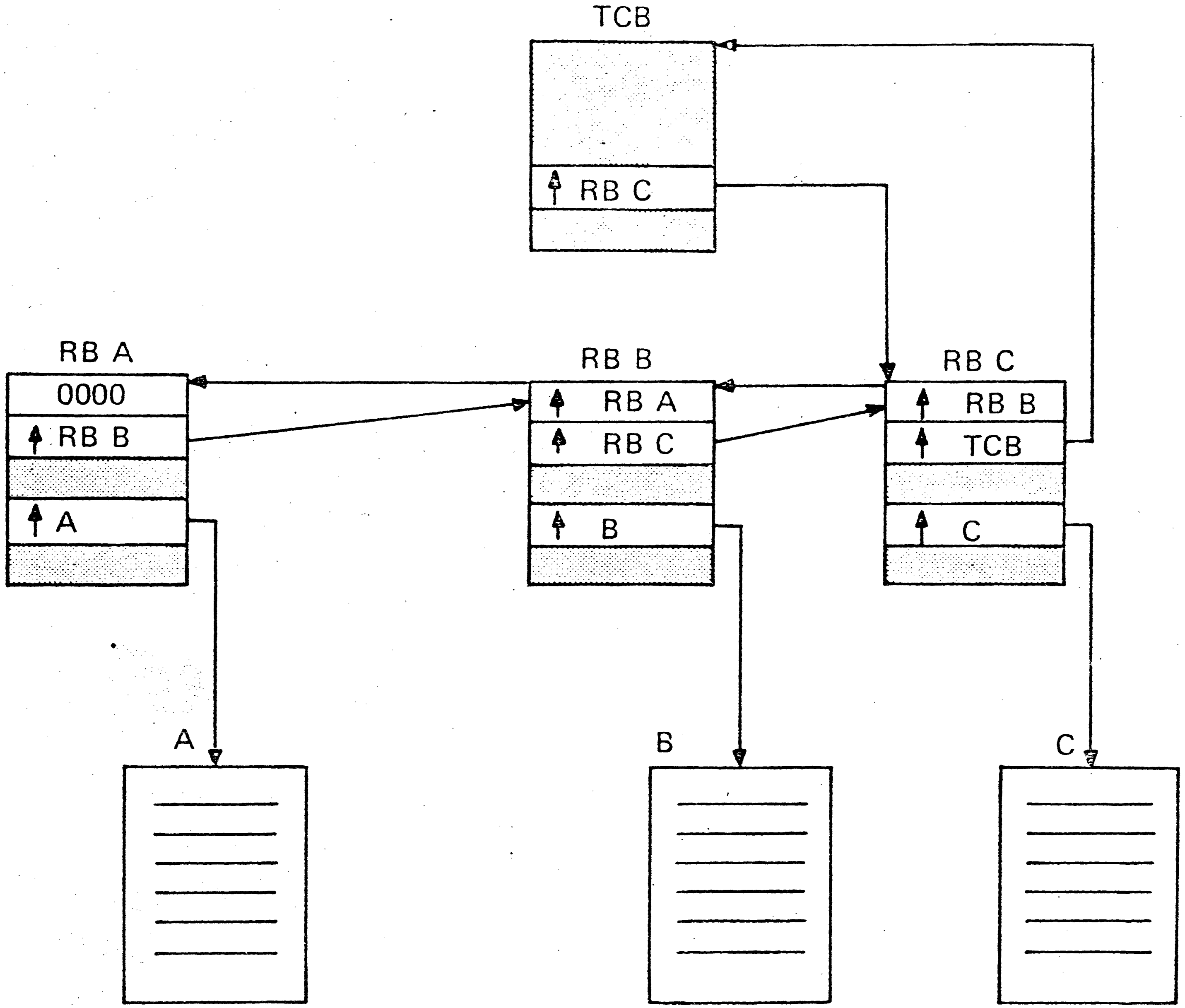
PROBLEM PROGRAM CODE

REGISTER SAVE AREA

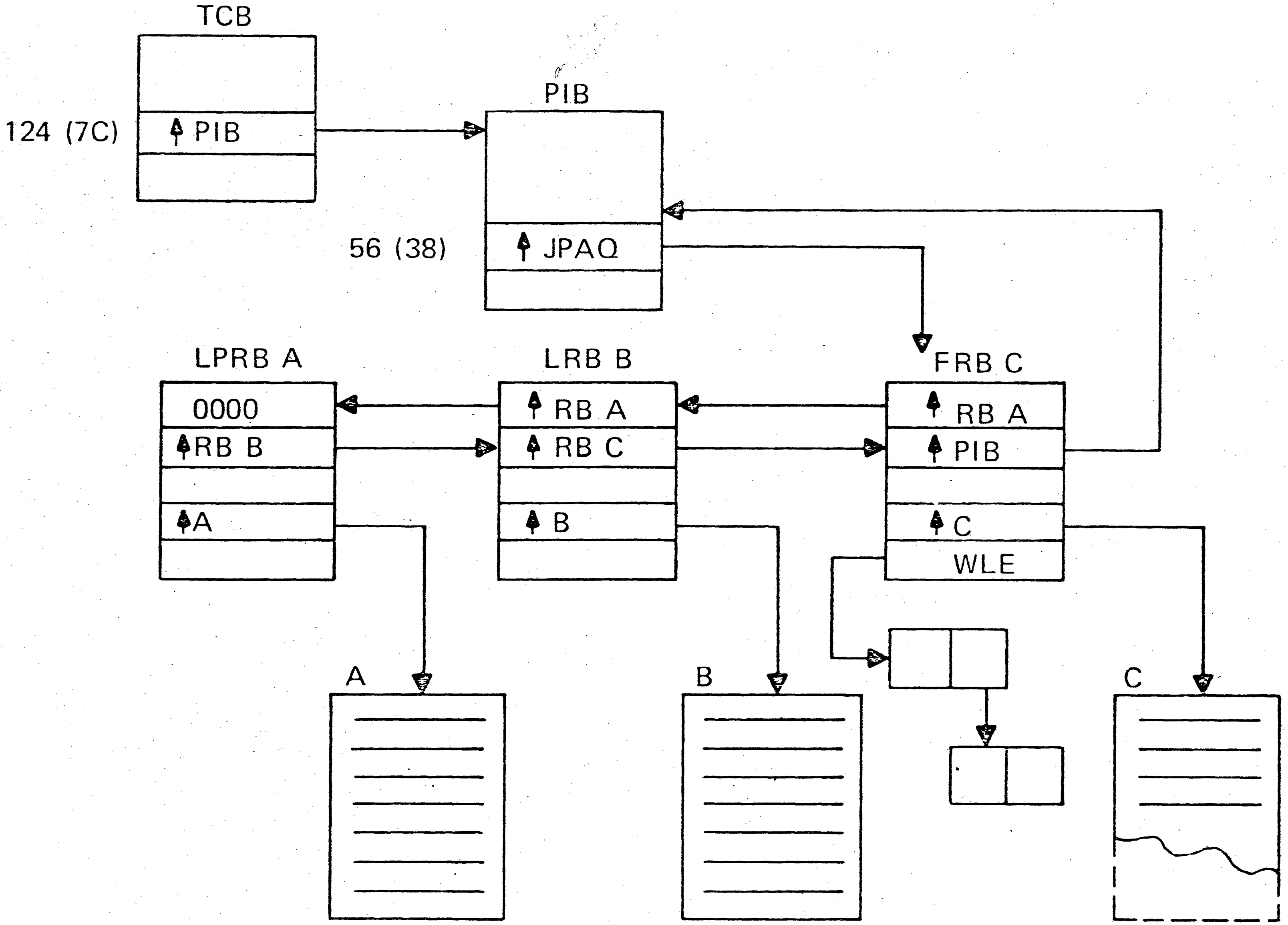
SAVE USE OF GETMAIN
IN SVC CODE TO ACQUIRE
A REGISTER SAVE AREA BY
USING SPACE IN SVRB



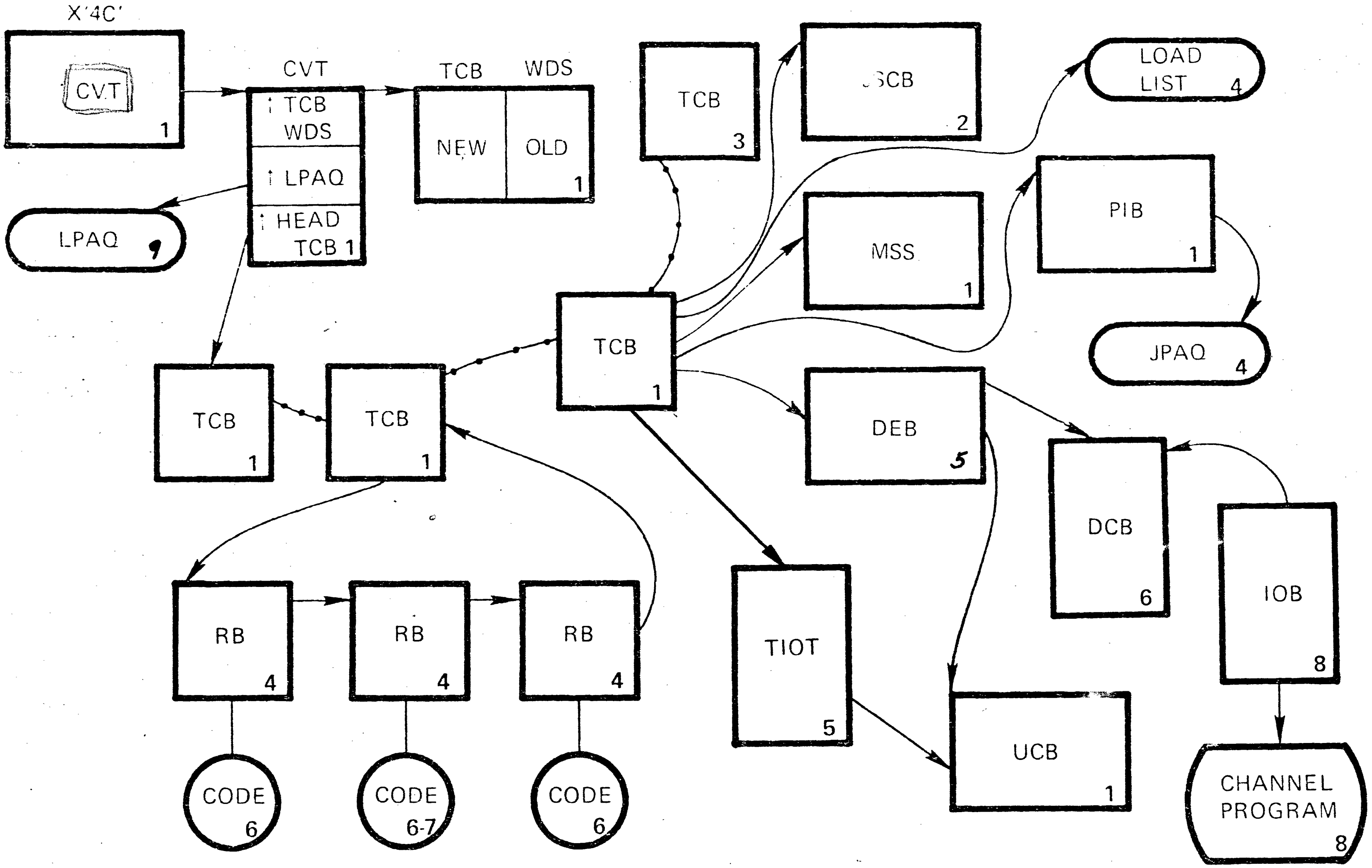
ACTIVE RB QUEUE



LOAD LIST



JOB PACK AREA QUEUE (JPAQ)



1 – NUCLEUS–SYSGEN
2 – SQA
3 – FPQA–CREATED VIA ATTACH
4 – FPQA
5 – HIGH END OF PARTITION–KEY=Ø (P-PQA)

6 – LOW END OF PARTITION–KEY= PP
7 – LOADED ROUTINES–KEY= PP or Ø
8 – NORMALLY LOADED WITH ACESS METHODS (7)
9 – PAGEABLE SUPERVISOR AREA

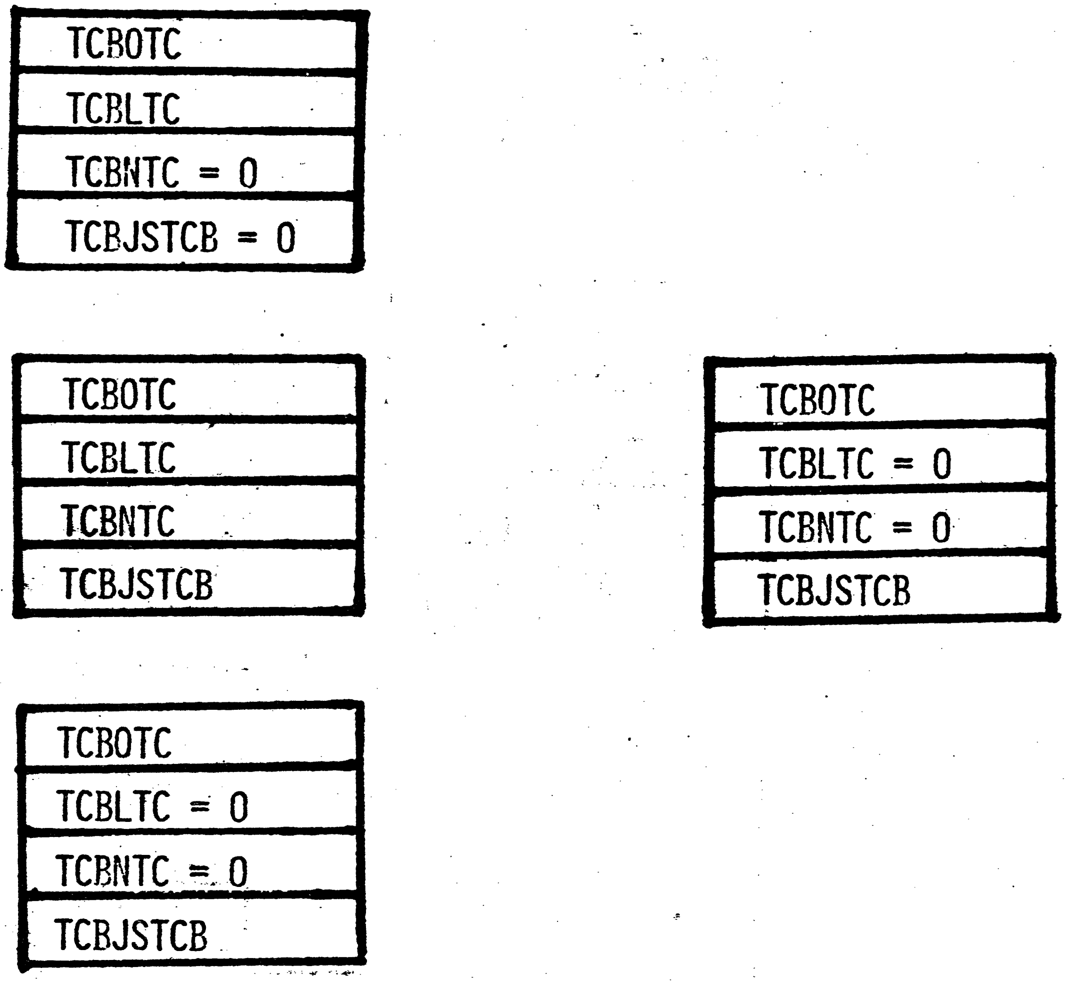
TCB POINTERS

JST ↑ job step [illegible]

OTC ↑ owning task [illegible]

LTC ↑ last task created

NTC ↑ NEXT task CREAT



TCBOTC
TCBLTC
TCBNTC = 0
TCBJSTCB = 0

TCBOTC
TCBLTC
TCBNTC
TCBJSTCB

TCBOTC
TCBLTC = 0
TCBNTC = 0
TCBJSTCB

TCBOTC
TCBLTC = 0
TCBNTC = 0
TCBJSTCB

1. ATTACH

2. [illegible]

[illegible]

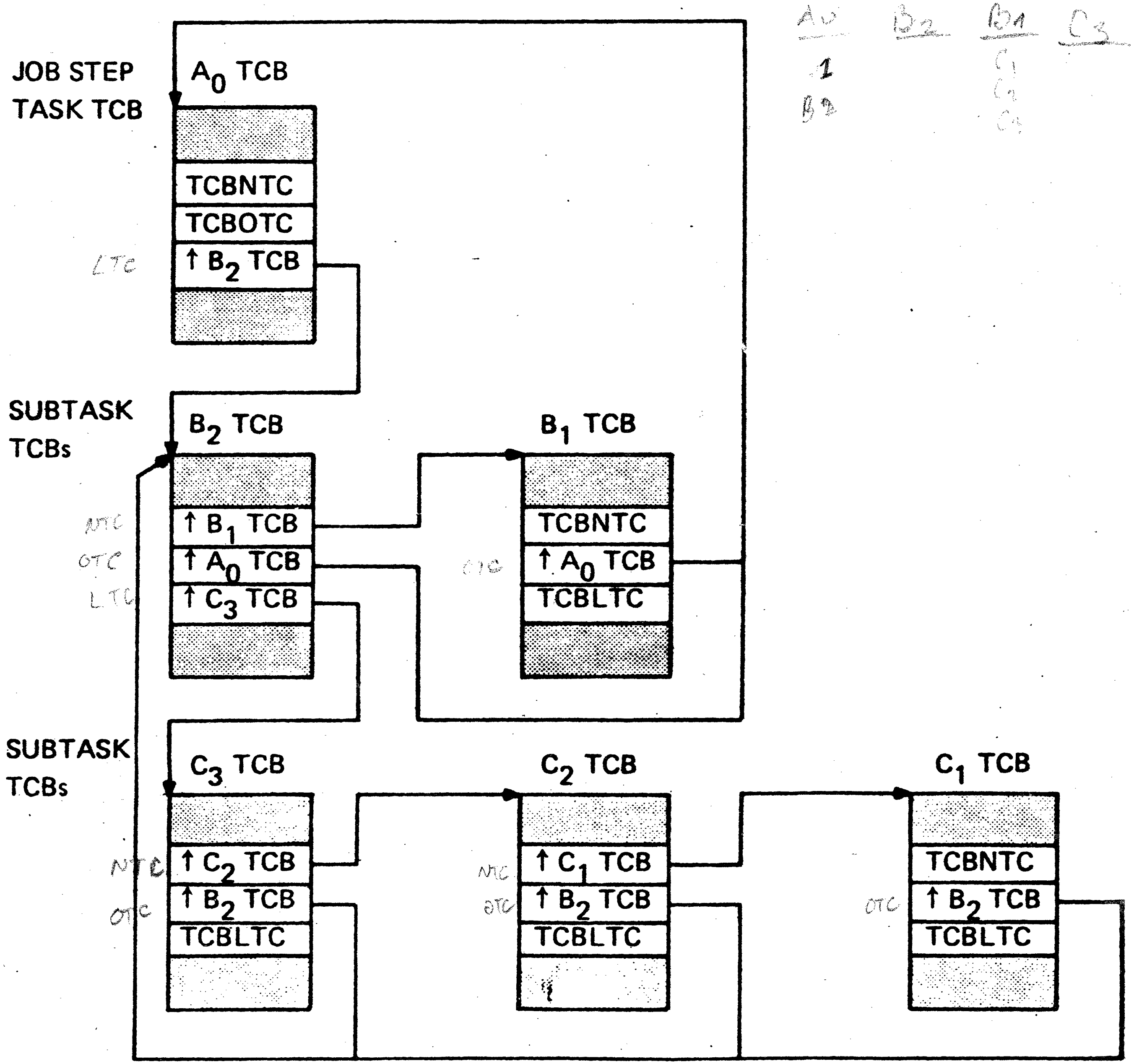
5. [illegible]

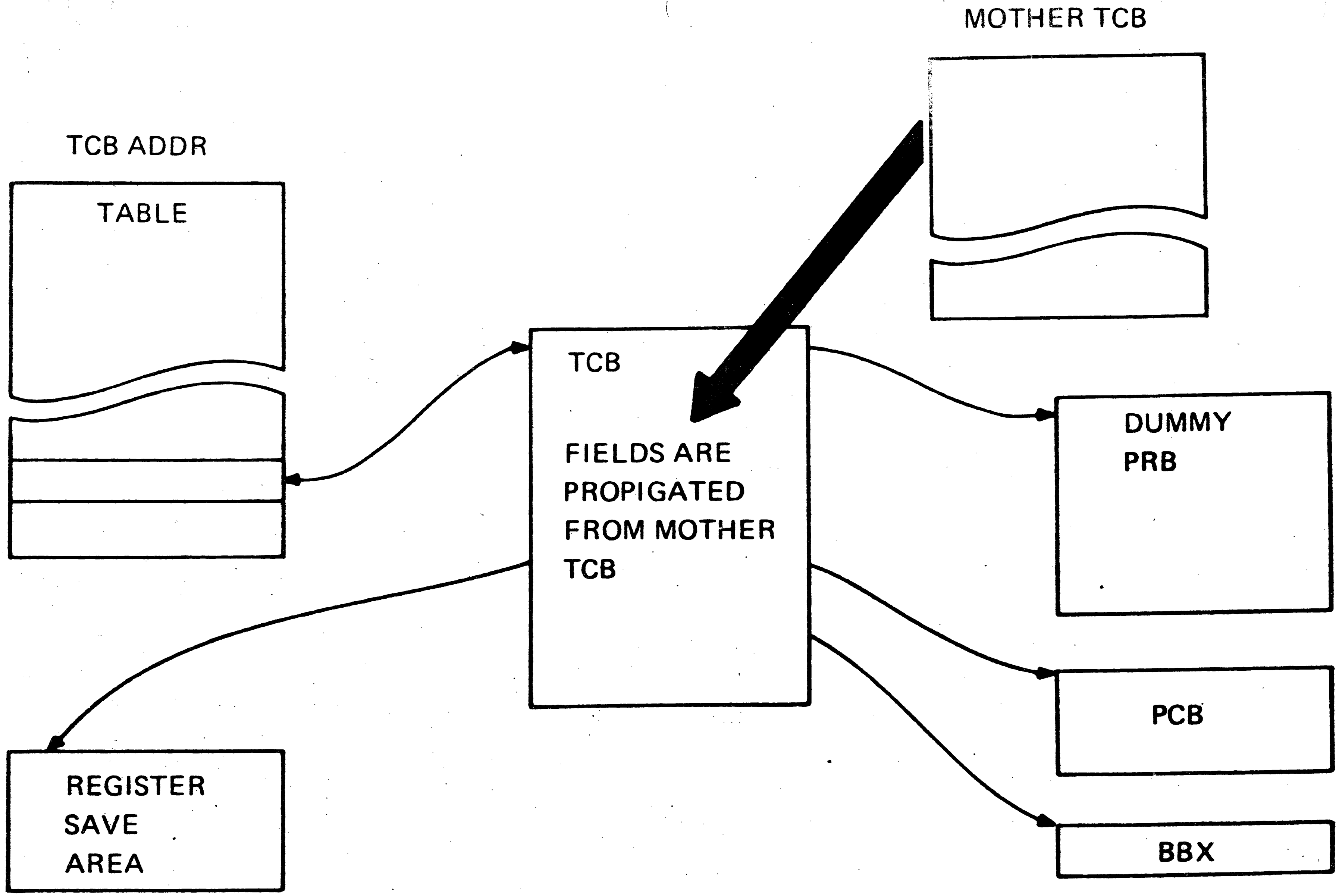
[illegible] 3

ATTach create an subtask ⇒ TCB

A0	B2	B1	C3	C2	C1
1		C1			
B2		C2			
		C3			

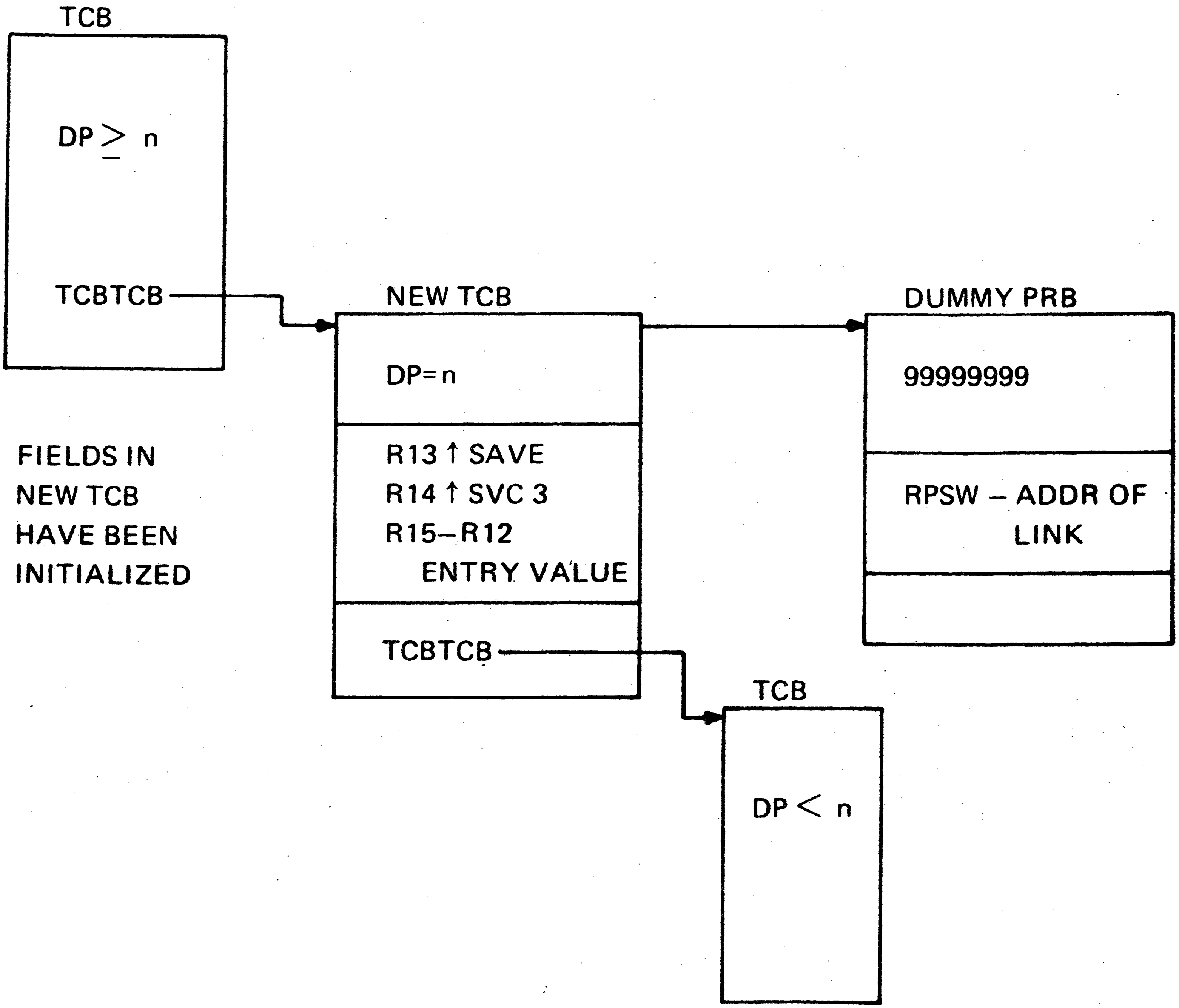
SUBTASKING





FINCH

PARAMETER LIST



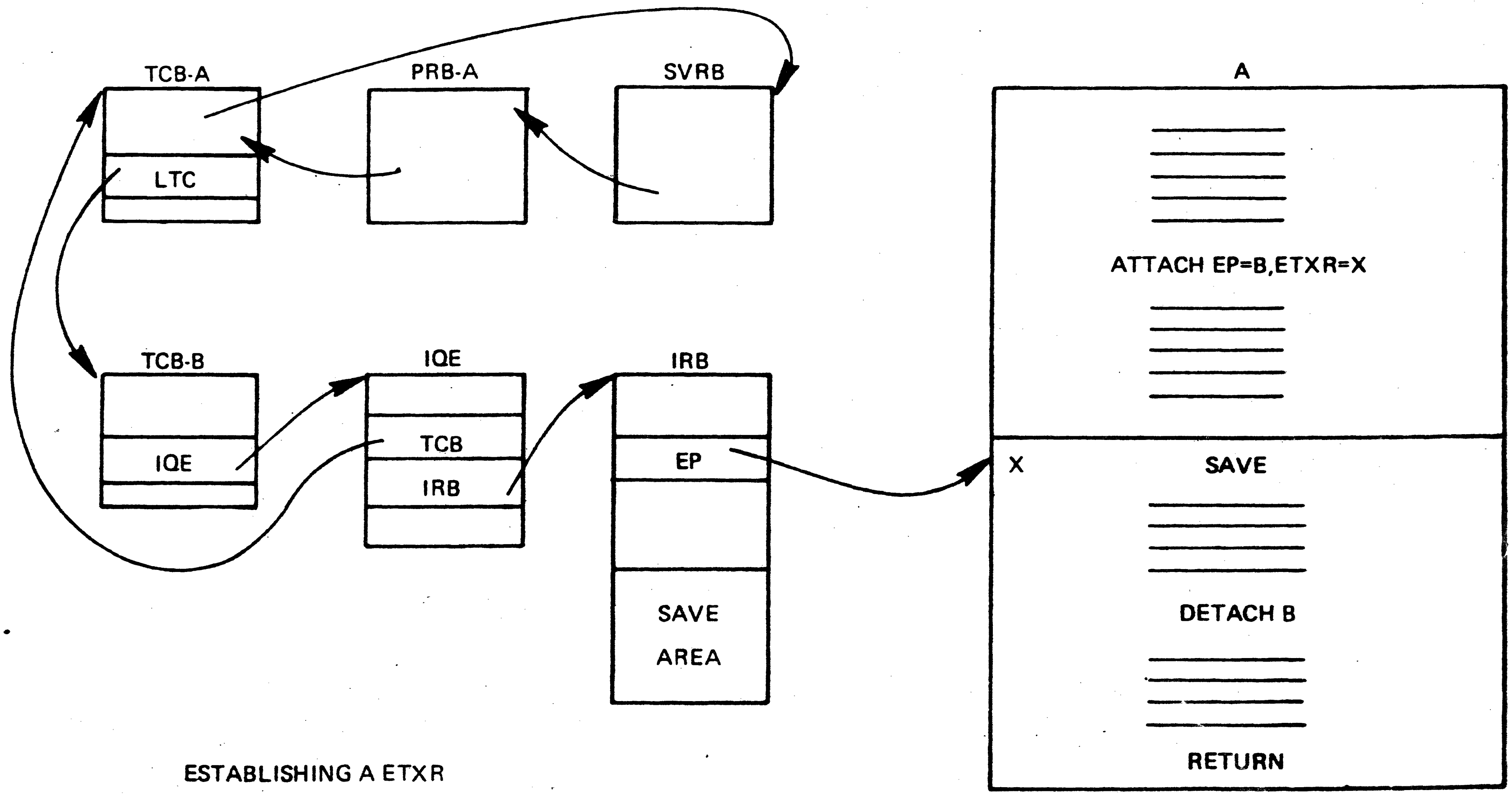
FIELDS IN
NEW TCB
HAVE BEEN
INITIALIZED

IQE

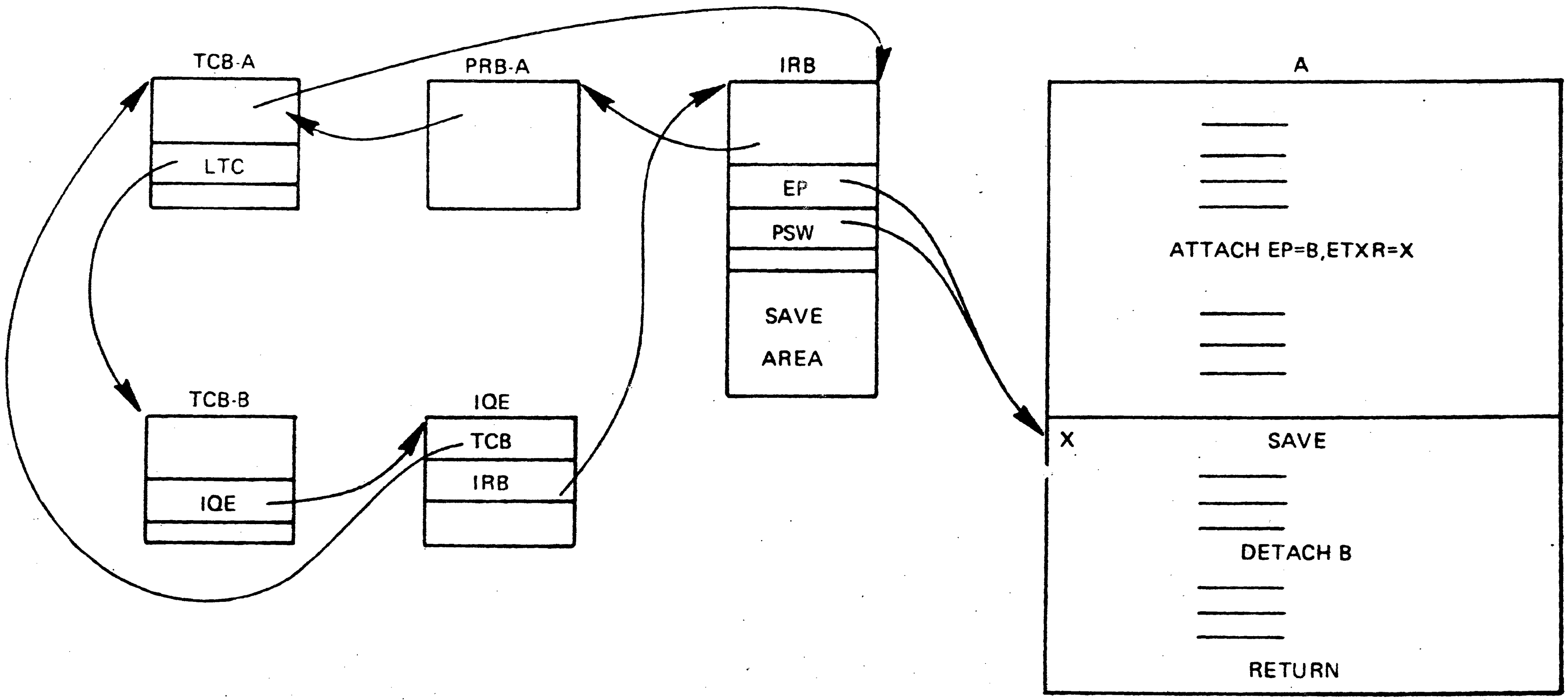
↑ NEXT IQE	↑ PARAMETER
↑ IRB	↑ TCB

IRB

0		
8		↑ ENTRY POINT
10	RESUME PSW	
18	↑ NEXT IQE	↑ NEXT RB/TCB
	REGISTER SAVE AREA	



ESTABLISHING A ETXR

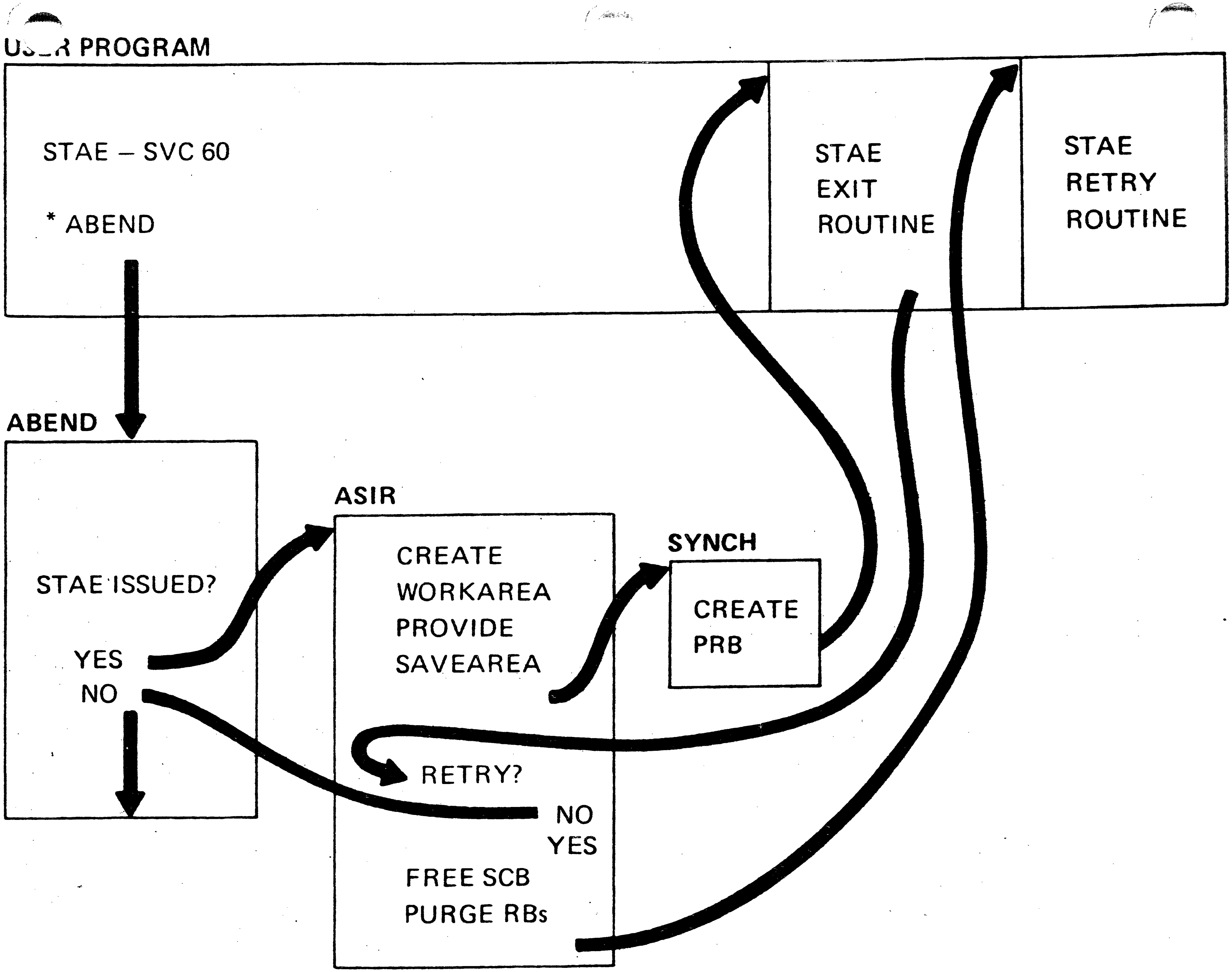


SCHEDULING A ETXR

ABEND – NORMAL END

CONDITION	ACTION
	TCBCMP
TCBTME ≠ 0 AND TQE ON QUEUE	ISSUE TTIMER CANCEL
TCBDEB ≠ 0	CLOSE DCBs AND DEQUE DEBs
	PURGE OUTSTANDING I/O REQUESTS AND WTORs
SUBTASK	
DP ≠ LP	ISSUE CHAP
	INITIALIZE BBX – FREE AREAS POINTED TO BY GQEs
	CLEAR POINTER IN PIB TO JPAQ
	CLEAR FIELDS IN TCB, LLS, JLB, FSA, PIE, TIOT, ETC
	CREATE PRB AND XCTL TO TERMINATOR

FRB EXIST FOR SUBTASK OR DAUGHTERS	FREE FRB AND WLEs DECREMENT WAIT COUNT IN SVRB
	RELEASE ENQ'D RESOURCES
	SVC 3


USER PROGRAM
STAE – SVC 60
* ABEND
STAE
EXIT
ROUTINE
STAE
RETRY
ROUTINE
ABEND
STAE ISSUED?
YES
NO
ASIR
CREATE
WORKAREA
PROVIDE
SAVEAREA
SYNCH
CREATE
PRB
RETRY?
NO
YES
FREE SCB
PURGE RBs

STAE WORKAREA

0	STAE EXIT ROUTINE PARAMETER LIST ADDRESS OR 0	ABEND COMPLETION CODE *
8	PSW AT TIME OF ABEND – BC MODE	
10	LAST PROBLEM PROGRAM PSW BEFORE ABEND OR 0 IF IT IS A SUPERVISOR ROUTINE – BC MODE	
18	≈ GENERAL REGISTERS 0-15 AT TIME OF ABEND ≈	
58	NAME OF FAILING PROGRAM OR 0	
60	ENTRY POINT ADDRESS OF FAILING PROGRAM	ZEROES

IF THE STAE USER IS A SUPERVISOR PROGRAM, THE FORMAT CHANGES TO:

58	RB ADDRESS OF FAILING PROGRAM	ZEROES
60	ZEROES	ZEROES

STAE EXIT ROUTINE

- STANDARD LINKAGE CONVENTIONS
- CANNOT ISSUE STAE OR ATTACH
- RESIDENT
- FUNCTIONS
  - PREFORM PRETERMINATION PROCESSING
  - DIAGNOSE
  - RETRY
- SCHEDULE RETRY OR CONTINUE WITH ABEND

STAE RETRY ROUTINE

- RESIDENT
- NO SAVEAREA PROVIDED
- CAN REISSUE STAE
- RESTORE PROGRAM REGISTERS FROM WORKAREA
- FREE WORKAREA
- RETURN TO PROGRAM

TASK SUPERVISION

IBM

SYSTEMS PROGRAMMING

Student Materials

This material was produced for Educational purposes only. Changes may obsolete part or all of this publication. No responsibility is assumed for any inaccuracies that occur.

Los Angeles Advanced Education Center
3424 WILSHIRE BOULEVARD • LOS ANGELES, CALIF. 90010

VS/I

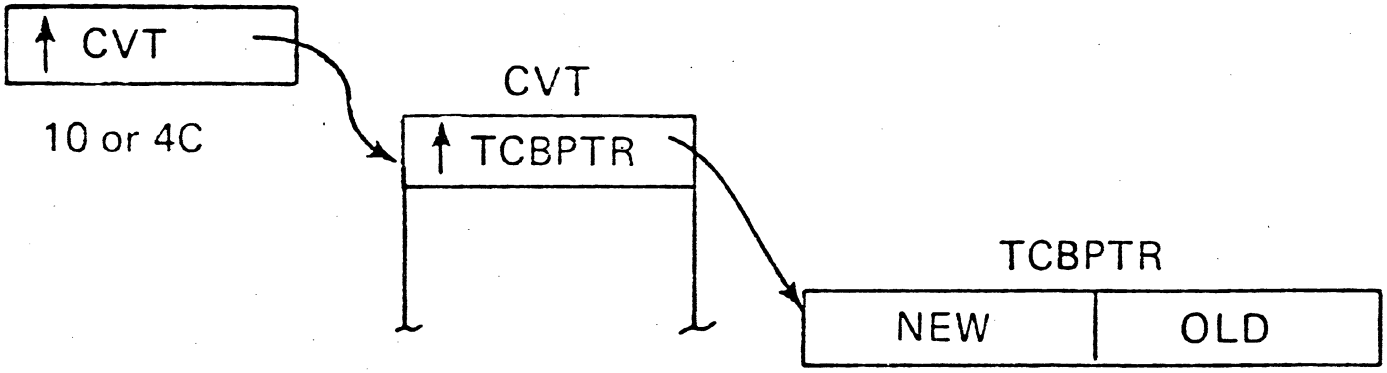
TASK SUPERVISION-TASK SWITCH LOGIC

-DYNAMIC DISPATCHING

-TIME SLICING

-AUTHORIZED PROGRAM FACILITY

TASK SWITCH LOGIC

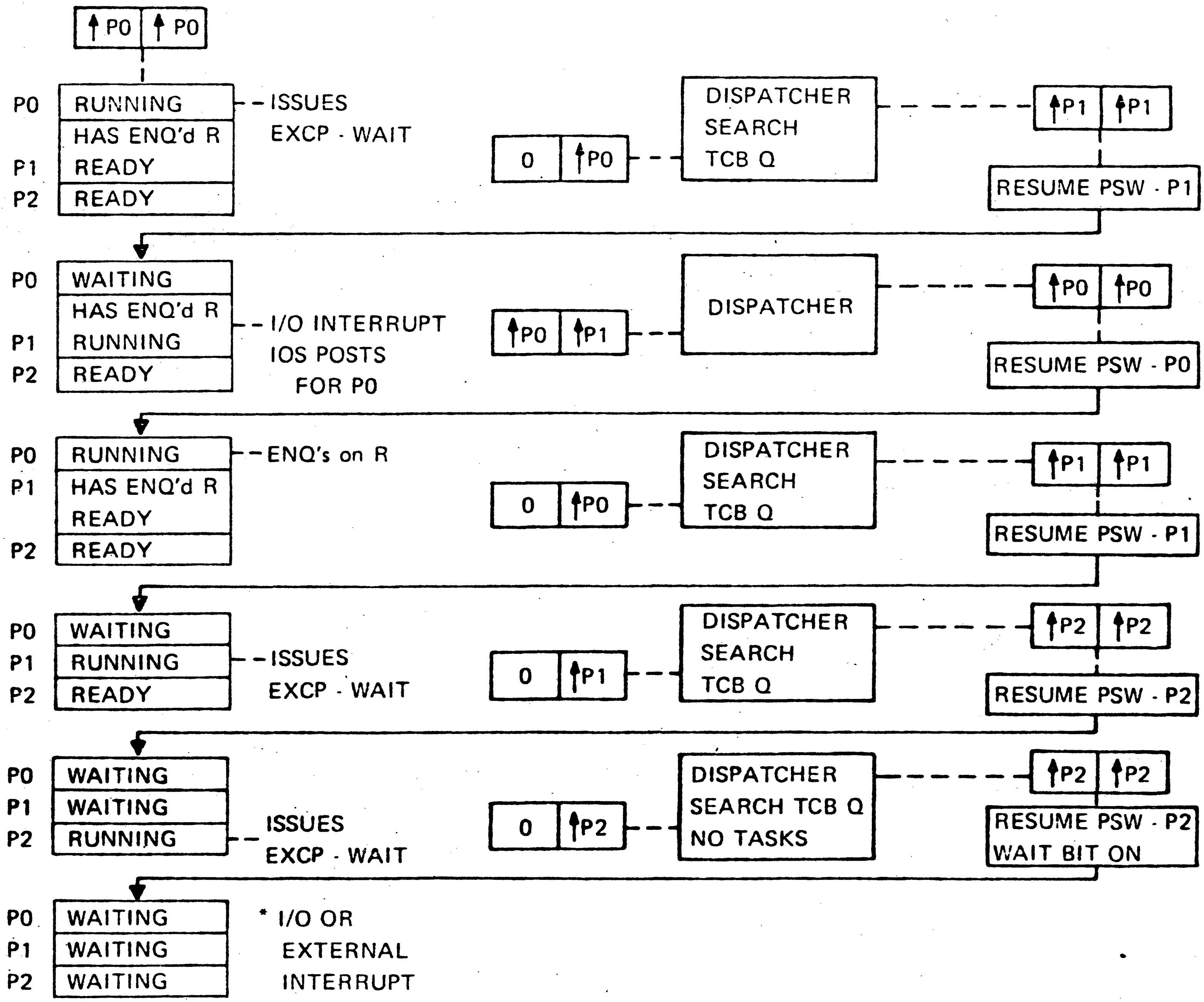


NEW=OLD – DISPATCH NEW

* NEW≠OLD – DISPATCH NEW

* NEW=ZERO – SEARCH STARTING WITH OLD

* INDICATES A TASK SWITCH


↑P0 ↑P0
P0 RUNNING
HAS ENQ'd R
P1 READY
P2 READY
ISSUES
EXCP - WAIT
0 ↑P0
DISPATCHER
SEARCH
TCB Q
↑P1 ↑P1
RESUME PSW - P1
P0 WAITING
HAS ENQ'd R
P1 RUNNING
P2 READY
I/O INTERRUPT
IOS POSTS
FOR P0
↑P0 ↑P1
DISPATCHER
↑P0 ↑P0
RESUME PSW - P0
P0 RUNNING
P1 HAS ENQ'd R
READY
P2 READY
ENQ's on R
0 ↑P0
DISPATCHER
SEARCH
TCB Q
↑P1 ↑P1
RESUME PSW - P1
P0 WAITING
P1 RUNNING
P2 READY
ISSUES
EXCP - WAIT
0 ↑P1
DISPATCHER
SEARCH
TCB Q
↑P2 ↑P2
RESUME PSW - P2
P0 WAITING
P1 WAITING
P2 RUNNING
ISSUES
EXCP - WAIT
0 ↑P2
DISPATCHER
SEARCH TCB Q
NO TASKS
↑P2 ↑P2
RESUME PSW - P2
WAIT BIT ON
P0 WAITING
P1 WAITING
P2 WAITING
* I/O OR
EXTERNAL
INTERRUPT

WAIT

- VALIDATE ECB
- ONE EVENT
  SET RB WAIT COUNT TO 1
- MULTIPLE EVENTS
  - NUMBER OF ECBs $\geq$ NUMBER OF EVENTS SPECIFIED
  - SET RB WAIT COUNT TO NUMBER OF EVENTS SPECIFIED
- CAN MEASURE WAIT TIME IF ALL TASKS IN PARTITION ARE WAITING
- INDICATE OLD TCB $\neq$ NEW TCB SO CURRENT TCB WILL NOT BE DISPATCHED

POST

- VALIDATE ECB ADDRESS

- TURN OFF WAIT BIT
  TURN ON COMPLETE BIT
  STORE COMPLETION CODE

- DECREMENT WAIT COUNT IN RB
  - IF WAIT COUNT NOT 0 – RETURN
  - IF WAIT COUNT = 0 AND
    - WAITING ON LESS THAN
      TOTAL NUMBER OF ECBs
      ZERO ALL ECBs.

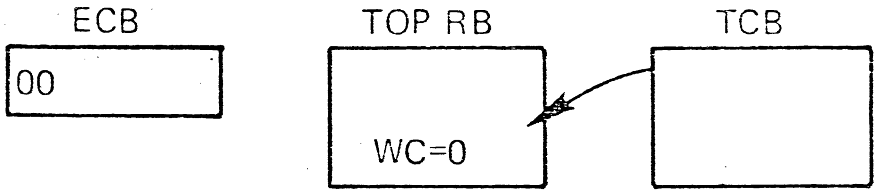
- RESTORE TASK TQE

- DETERMINE TCB TO BE DISPATCHED

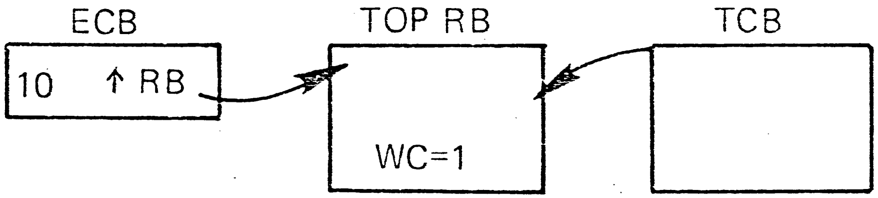
WAIT/POST

ONE EVENT (NOT YET POSTED)

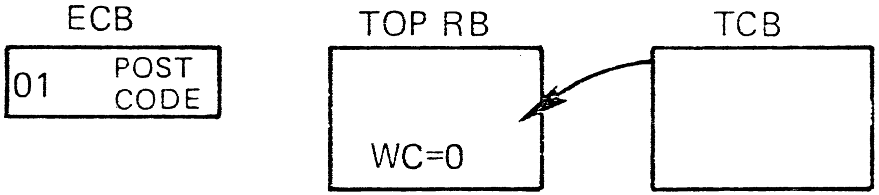
❶ INITIALLY



❷ WAIT

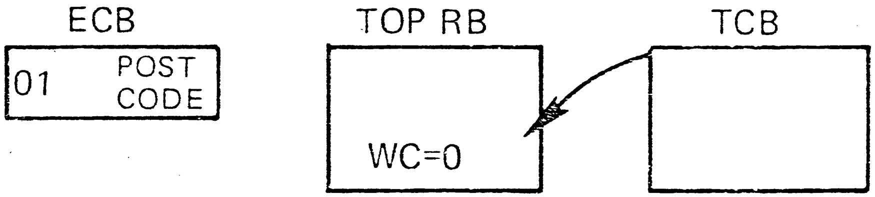


❸ POST



ONE EVENT (POSTED)

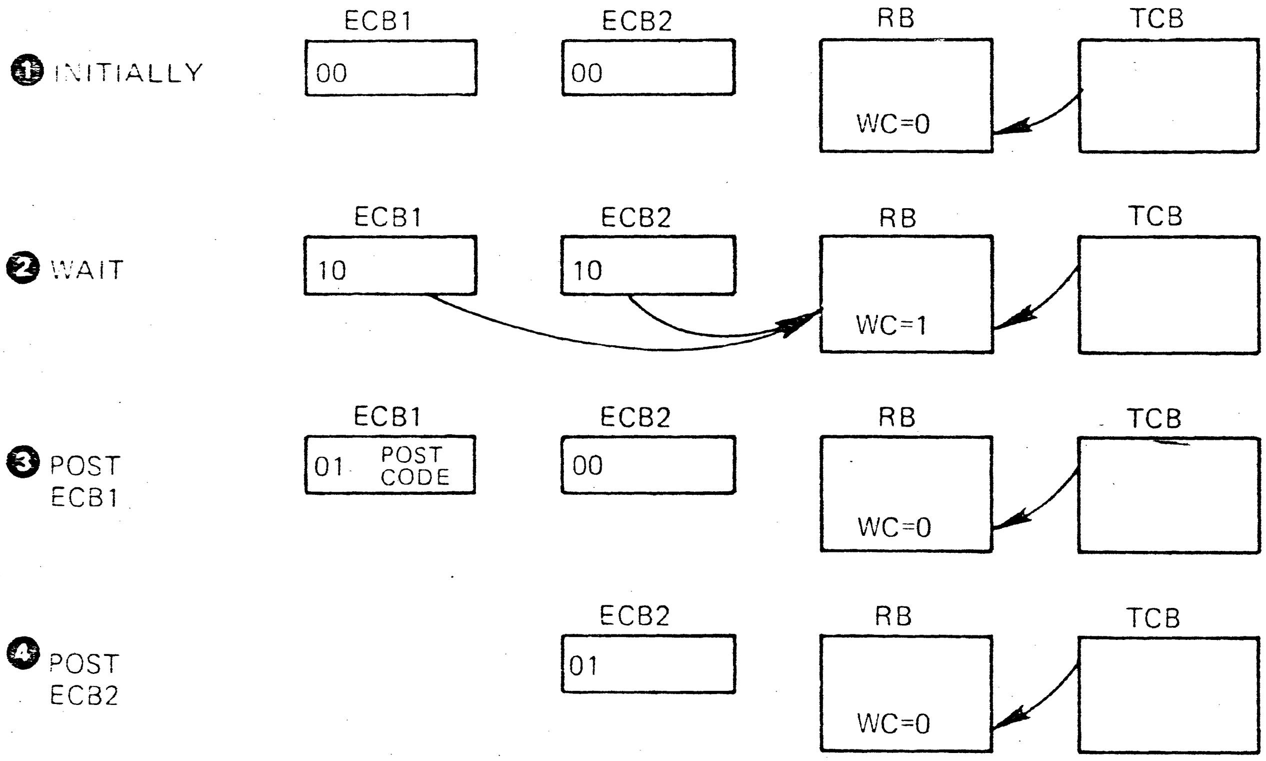
❶ INITIALLY



❷ WAIT (NO-OP)

WAIT/POST

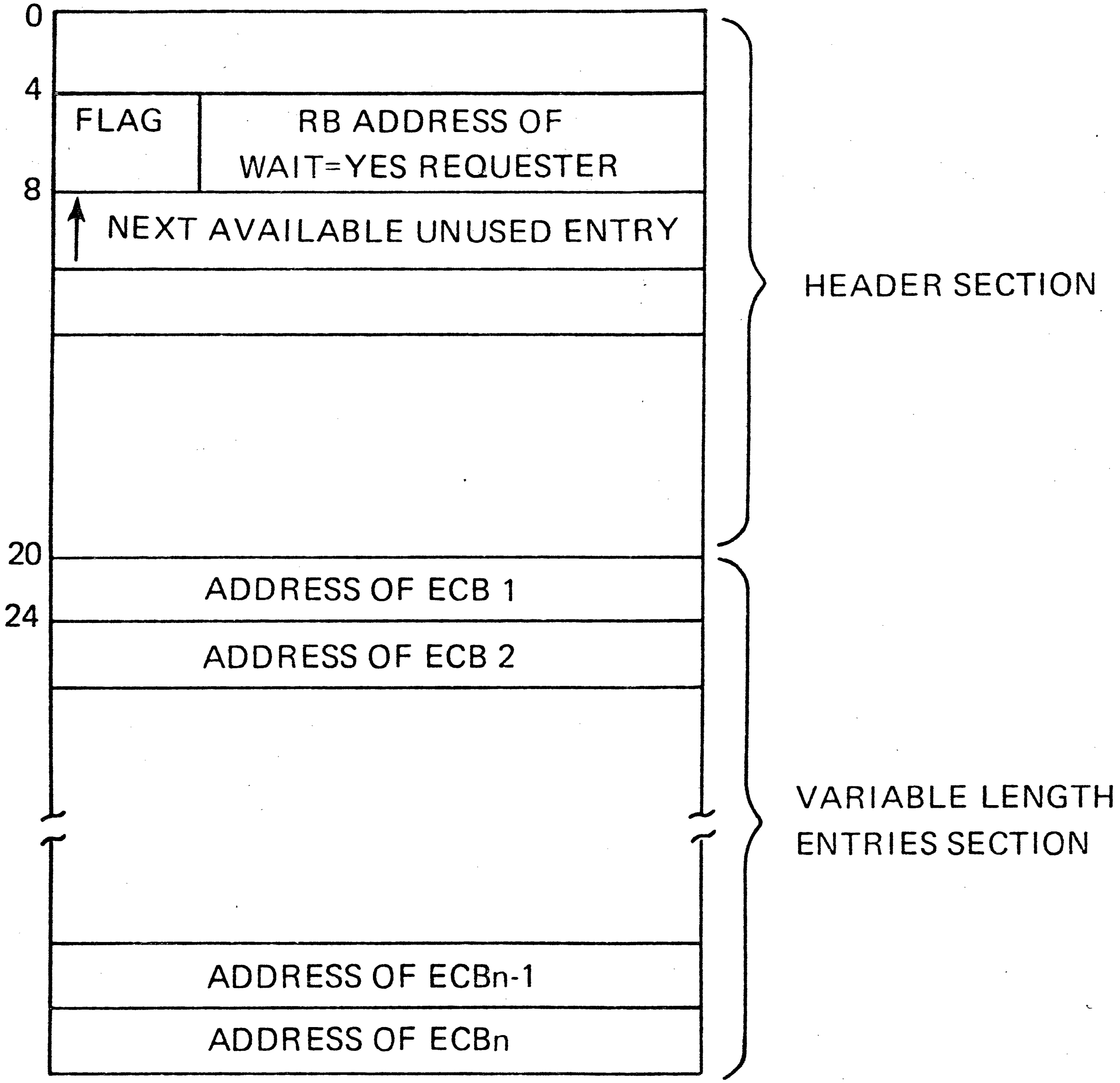
MULTIPLE EVENTS (1/2)



	ECB1	ECB2	RB	TCB
❶ INITIALLY	00	00	WC=0	
❷ WAIT	10	10	WC=1	
❸ POST ECB1	01 POST CODE	00	WC=0	
❹ POST ECB2		01	WC=0	

$$\text{EVENTS}\left\{\begin{array}{l}\text{ENTRIES} = \text{n}\\ \\ \text{TABLE} = \text{address}\left\{\begin{array}{l}\text{, WAIT} = \left\{\begin{array}{l}\text{yes}\\ \text{no}\end{array}\right\}\\ \text{, ENTRIES} = \text{DEL}\\ \text{, ECB} = \text{address}\end{array}\right\}\left[\begin{array}{l}\text{, ECB} = \text{address}\\ \text{, LAST} = \text{address}\end{array}\right]\end{array}\right\}$$

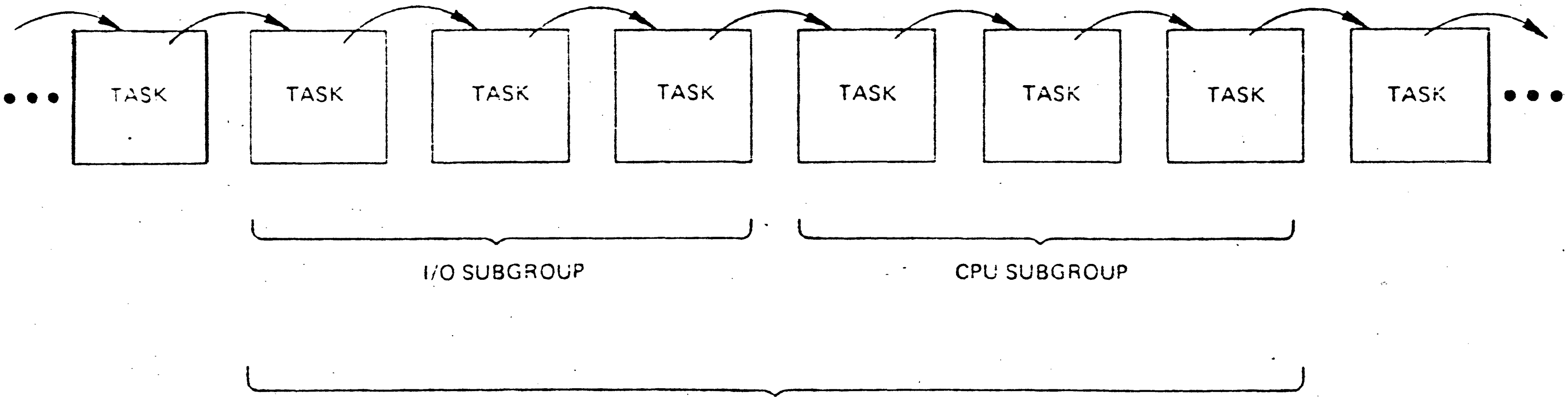
EVENTS TABLE



AUTOMATIC PRIORITY GROUP (APG)

- SINGLE PRIORITY LEVEL
  - SYSGEN
  - NIP
- NOT ALSO TIME-SLICE
- BALANCE CPU–I/O RESOURCES
- HEURISTIC
- PARAMETERS
  - TIME INTERVAL
  - INCREMENT
  - LOWER/UPPER LIMIT
  - RATIO
  - STATISTICS INTERVAL

TASK DISPATCHER
TCB QUEUE



DYNAMIC DISPATCHING

ORIGINAL TASK STATUS	REASON FOR LOSS OF CPU CONTROL	NEW TASK STATUS	ACTION TAKEN
I/O BOUND	VOLUNTARY SURRENDER	NO CHANGE	SEARCH DOWN I/O QUEUE FOR NEXT TASK TO DISPATCH.
I/O BOUND	TIME INTERVAL ENDED	CPU BOUND	MOVE TASK TO HEAD OF CPU SUBGROUP AND SEARCH DOWN I/O QUEUE FROM OLD LOCATION OF TASK.
I/O BOUND	PREEMPTION FOR ANOTHER TASK	NO CHANGE	DISPATCH PREEMPTING TASK.
CPU BOUND	VOLUNTARY SURRENDER	I/O BOUND	MOVE TASK TO BOTTOM OF I/O SUBGROUP AND SEARCH DOWN CPU QUEUE FROM OLD LOCATION OF TASK.
CPU BOUND	TIME INTERVAL ENDED	NO CHANGE	MOVE TASK TO BOTTOM OF CPU SUBGROUP AND SEARCH DOWN CPU QUEUE FROM OLD LOCATION OF TASK.
CPU BOUND	PREEMPTION FOR ANOTHER TASK	NO CHANGE	MOVE TASK TO BOTTOM OF CPU SUBGROUP AND DISPATCH PREEMPTING TASK.

SYSGEN	IPL
DYNPART = (Pn - Pm)	DDG = (Pn - Pm) OR DDG =,
DYNINTR = (A, B, C, D)	
A = DELTA VALUE TO BE ADDED TO OR SUBTRACTED FROM TIME SLICE INTERVAL AT END OF EACH STATISTICS INTERVAL	DDDEL =
B = LOWER BOUND OF TIME SLICE THAT CAN BE GIVEN TO A TASK	DDMIN =
C = RATIO OF CPU TO I/O BOUND TASKS	DDRATIO =
D = LENGTH OF STATISTICS INTERVAL	DDSTAT =

CVT + X'170' – ↑ DDCE

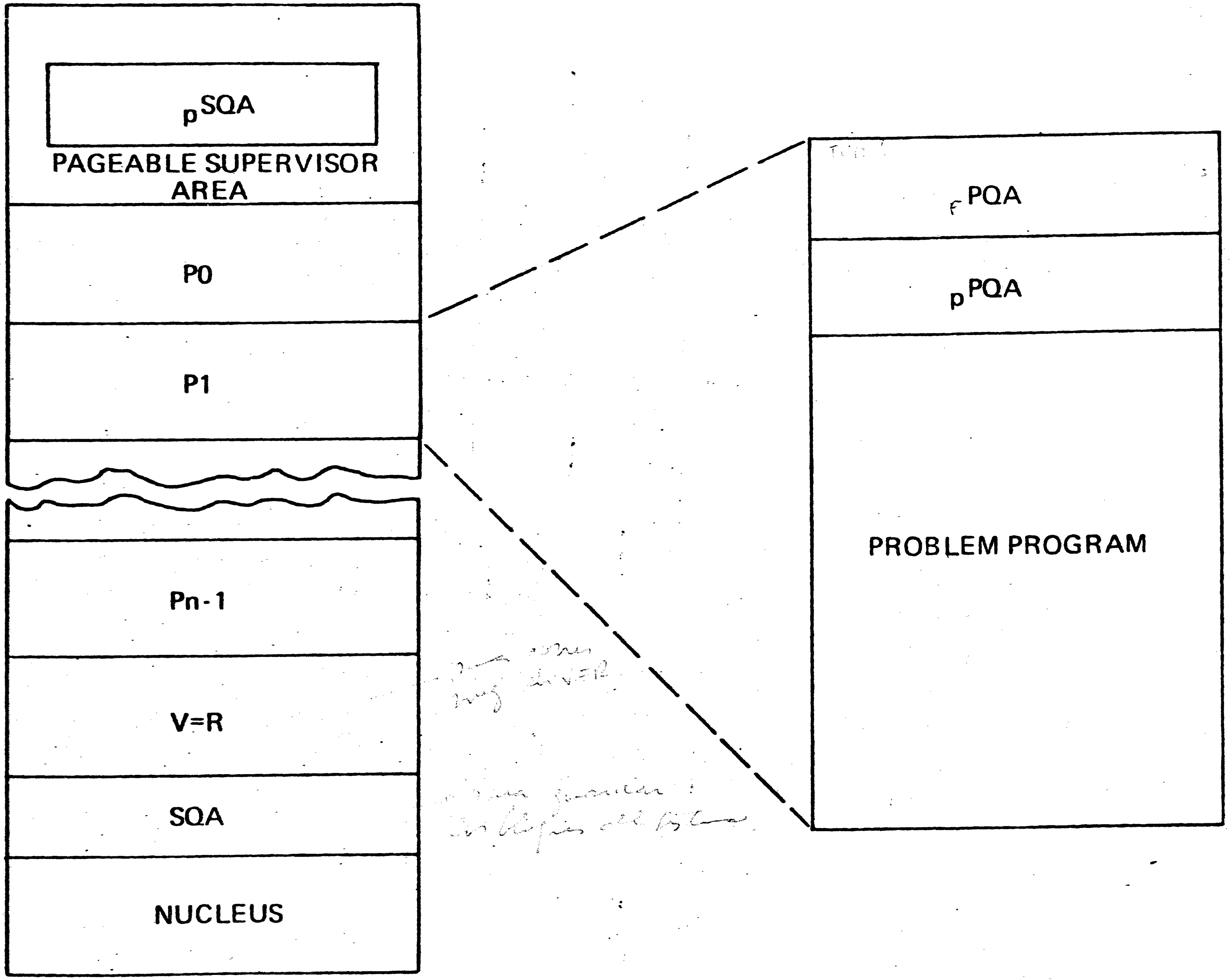
0	PRTY	ADDR OF TCB BEFORE DDG	FLAGS	ADDR OF LAST TCB IN DDG
8		ADDR OF LAST I/O BOUND TCB	LENGTH OF TIME SLICE	
10	RATIO OF CPU TO I/O BOUND TASKS		MINIMUM TIME SLICE INTERVAL ALLOWED	
18	DELTA APPLIED TO TIME SLICE		LENGTH OF STATISTICS INTERVAL	

TCB + X 'B8' – TCBDDEXC - NUMBER OF TIMES A DD TASK HAS HAD ITS TIME SLICE EXPIRE

TCB + X 'BA' – TCBDDWTC - NUMBER OF TIMES A DD TASK IS NOT INTERRUPTED BY THE END OF A TIME SLICE

TCB + X '9C' – TCBDDRTI - AMOUNT OF TIME REMAINING IN TASK'S DD TIME SLICE

TCB + X'CB' – TCBFLGS7 - BITS INDICATE MEMBER OF DDG, I/O BOUND, CPU BOUND



[illegible] in V=R

[illegible]

STORAGE MGMT

boundary box
of prog area partition

BBX 1	↑ PFQE CHAIN	↑ P_X LOW BOUNDARY	↑ P_X HIGH BOUNDARY	↑ PPQE CHAIN
		(PROBLEM PROGRAM AREA BOX)		
BBX 2	↑ FQE CHAIN	↑ ORIGINAL FIXED PQA LOW BOUND	↑ ORIGINAL FIXED PQA HIGH BOUND	↑ ALLOCATED FIXED PQA CHAIN
		(FIXED PQA BOX)		
BBX 3	COPY OF BBX 1 WHEN A JOB STEP TASK RUNS V = R			
BBX 4	↑ PFQE CHAIN			
		(PAGEABLE PQA BOX)		

[illegible] PQA

	AREA	POSITION	GQE		GQE ON	EXAMPLE OF USE
			SUB	JS	TCB	
0-127	PROBLEM PROGRAM	HIGH*	YES**		TASK	USER GETMAIN
128		HIGH	YES		TASK	LOAD – NON REENTRANT MODULE
238		LOW	YES		TASK	LINK OR XCTL – MODULE
240		HIGH				PARAMETER LIST
250		HIGH				LOAD – REENTRANT MODULE FROM PDS OTHER THAN LINKLIB OR SVCLIB
251		LOW				
252	PAGEABLE PQA		YES	YES	JS	LOAD – REENTRANT MODULE FROM LINKLIB OR SVCLIB
253			YES	YES	TASK	
254			YES	YES	JS	DEB
255						
233	PQA		YES	YES	TASK	RB
234			YES	YES	JS	RB FOR MODULE ON JPAQ
235						RB FOR DUMMY PRB USED IN ATTACH
241	PAGEABLE SQA					
242	V=R					
245	SQA					

* WILL BE LOW CORE IF REQUESTED BY SUPVR STATE

** ONLY IF REQUESTED BY KEY 0

1. INITIATOR GETS REGISTER SAVE AREA AND AREA FOR PARAMETERS.

2. INITIATOR OPENS STEPLIB – DEB IN pPQA.

3. INITIATOR XCTLs TO PROBLEM PROGRAM MODULE A.

4. A LOADS NON RE-ENTRANT MODULE B – 3K.

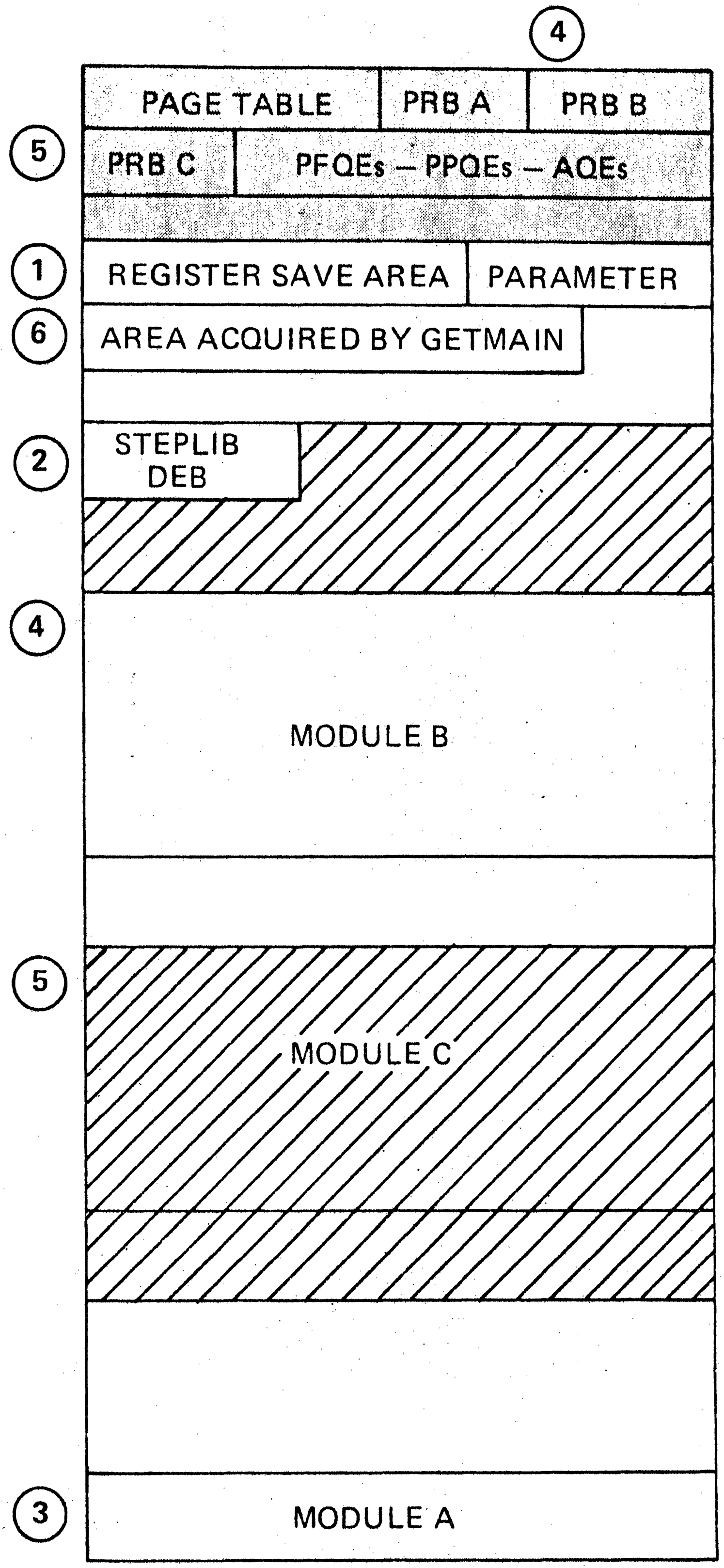
5. A LOADS C – A 3K RE-ENTRANT MODULE FROM SYS1.LINKLIB.

6. A DOES GETMAIN FOR 100 BYTES.

░ PQA

▨ pPQA

□ PROBLEM PROGRAM



PROBLEM PROGRAM SPACE IS REQUESTED

SEARCH PROBLEM PROGRAM PFQE CHAIN FOR STORAGE TO SATISFY REQUEST

- NOT AVAILABLE – RETURN
- AVAILABLE –
  - UPDATE PFQE CHAIN TO REFLECT REDUCED FREE SPACE
  - IF AREA REQUESTED = FREE AREA, FREE PFQE IN PQA.

REQUEST FOR FIXED PQA

TRY TO GET SPACE FROM EXISTING PQA

IF AVAILABLE – RETURN

IF NOT AVAILABLE – GET NEW PAGE FROM PROBLEM PROG. SPACE (ASSUME PAGE DOES NOT BORDER ON BEGINNING OR END OF FREE AREA)

– UPDATE EXISTING PFQE TO REFLECT REDUCED PROBLEM PROGRAM SPACE

– GET NEW PFQE (FROM PQA) AND INDICATE REMAINDER OF ORIGINAL AREA

– GO TO PAGE MGMT ROUTINES TO FIX PAGE FRAME

– SET STORAGE KEY TO 0

– GET FQE AND PUT NEW PAGE IN PQA FQE QUEUE

– GET AQE AND UPDATE PQA ALLOCATION QUEUE

– RETURN

CONTENTS SUPERVISION

SYSTEMS PROGRAMMING

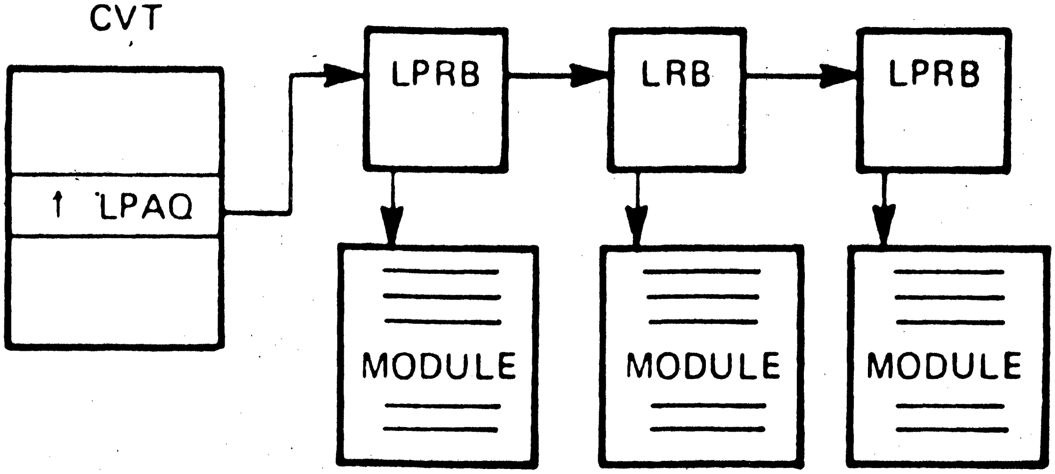
Student Materials

This material was produced for Educational purposes only. Changes may obsolete part or all of this publication. No responsibility is assumed for any inaccuracies that occur.

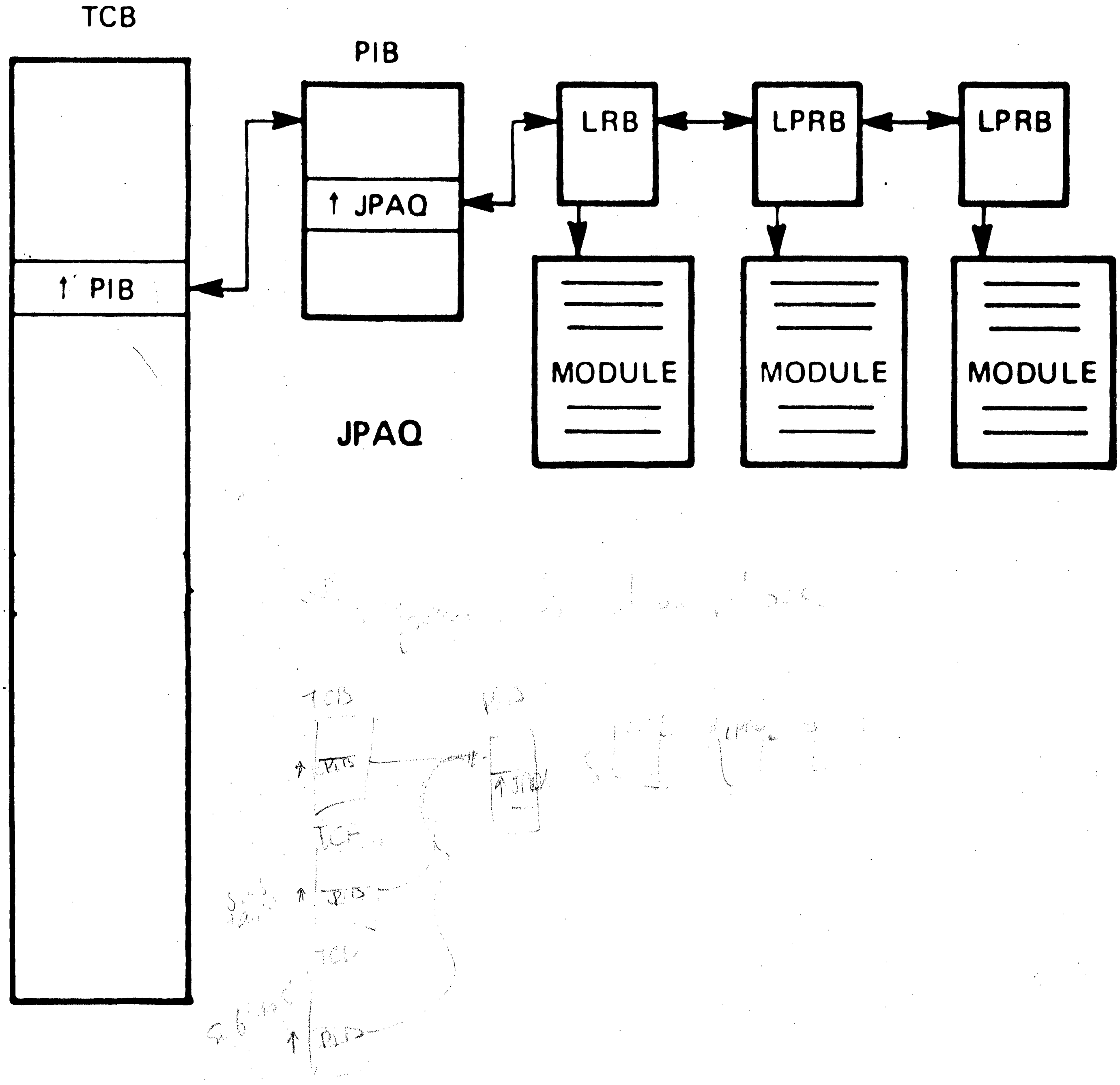
Los Angeles Advanced Education Center
3424 WILSHIRE BOULEVARD • LOS ANGELES, CALIF. 90010

VSI

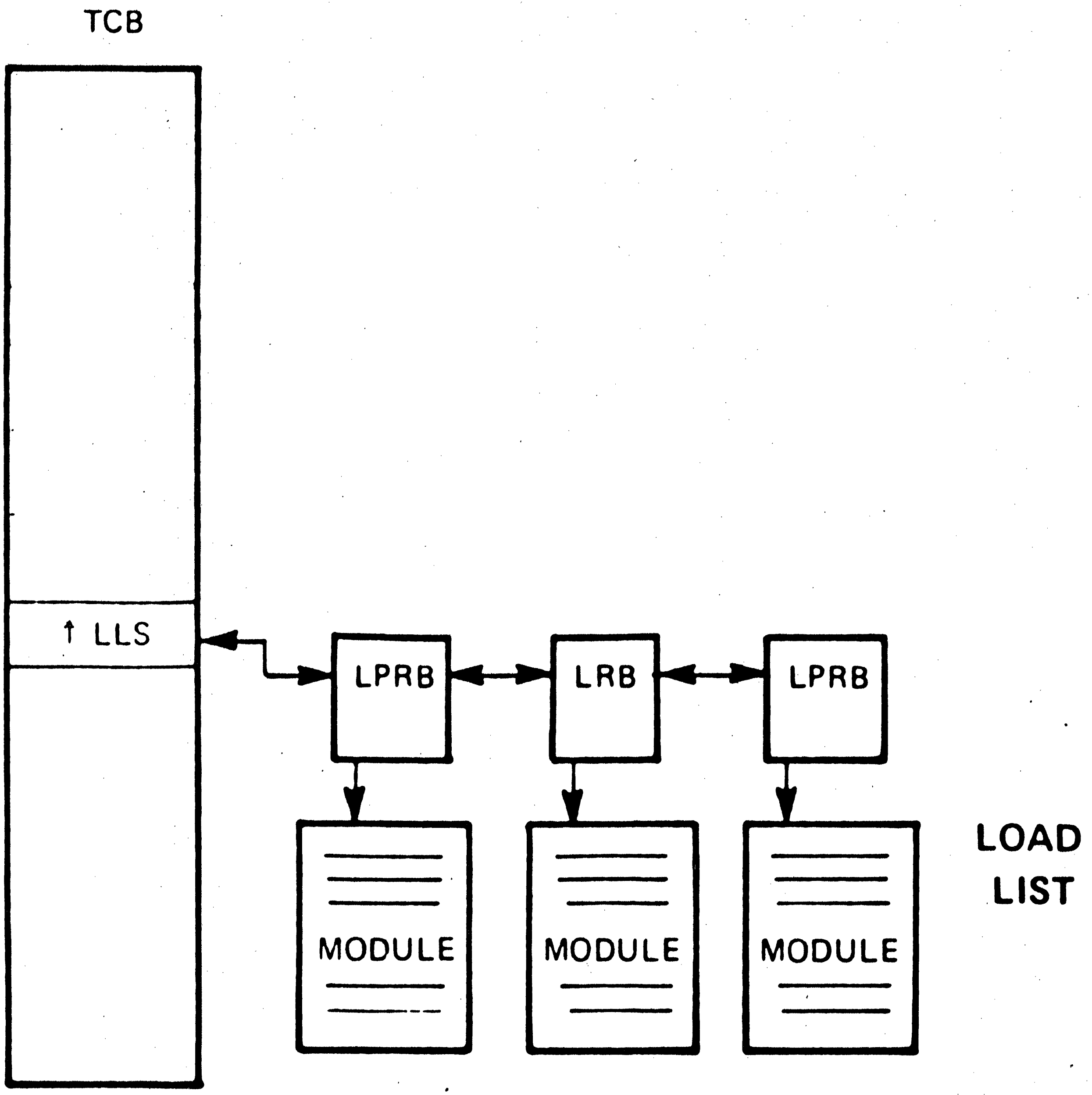
CONTENTS SUPERVISION



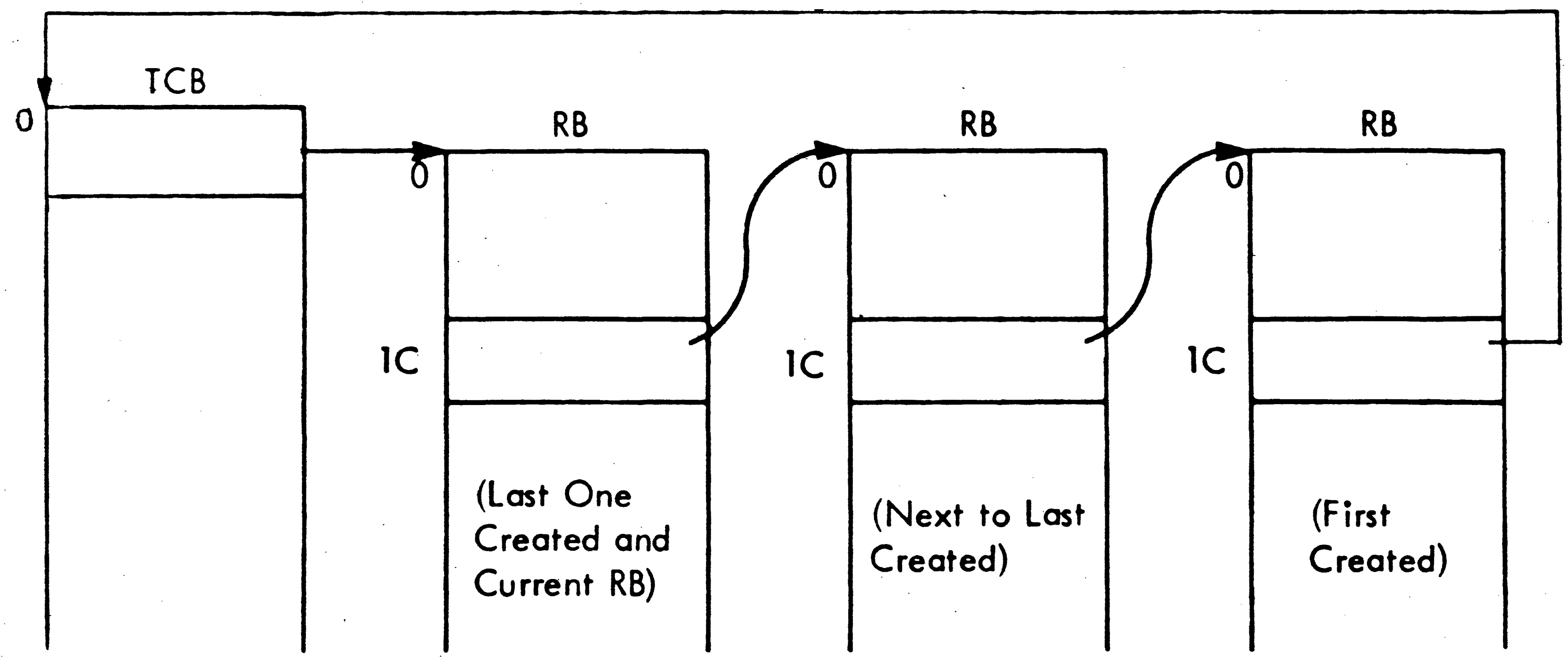
LPAQ

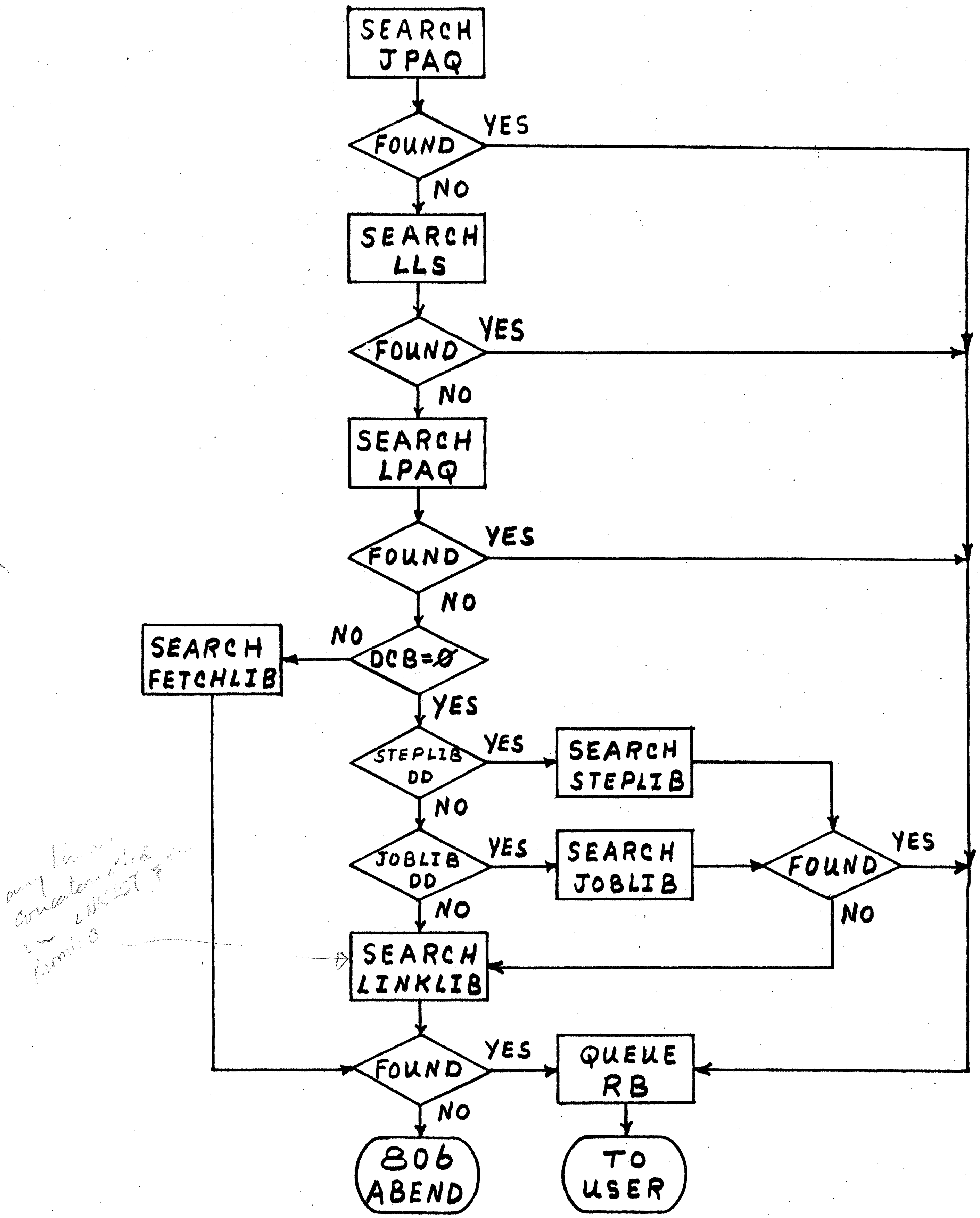

TCB
↑ PIB
PIB
↑ JPAQ
LRB
LPRB
LPRB
MODULE
MODULE
MODULE
TCB
↑ PIB
PIB
↑ JPAQ
TCB
↑ PIB
TCB
↑ PIB

JPAQ


TCB
↑ LLS
LPRB
LRB
LPRB
MODULE
MODULE
MODULE
LOAD
LIST

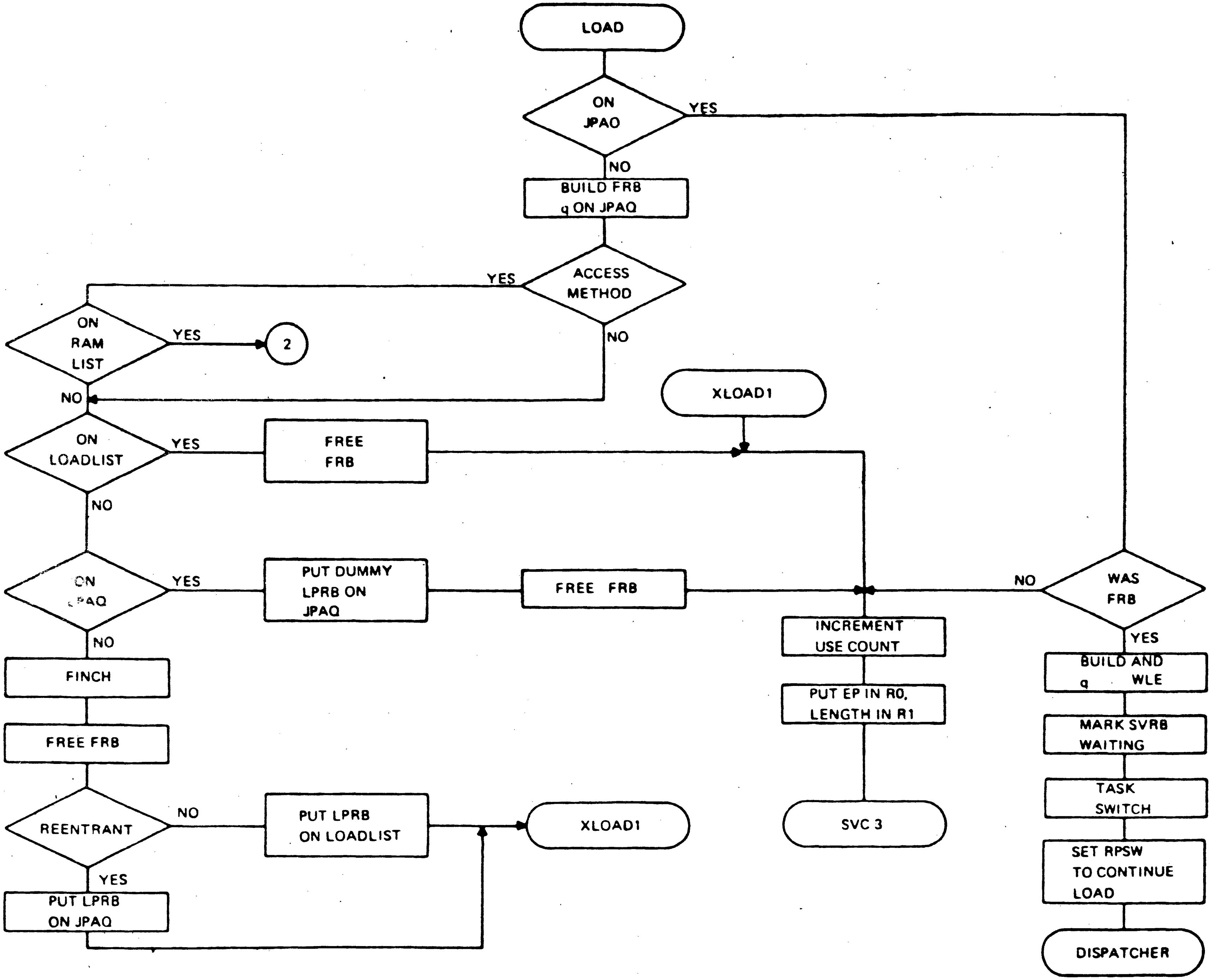
ACTIVE QUEUE

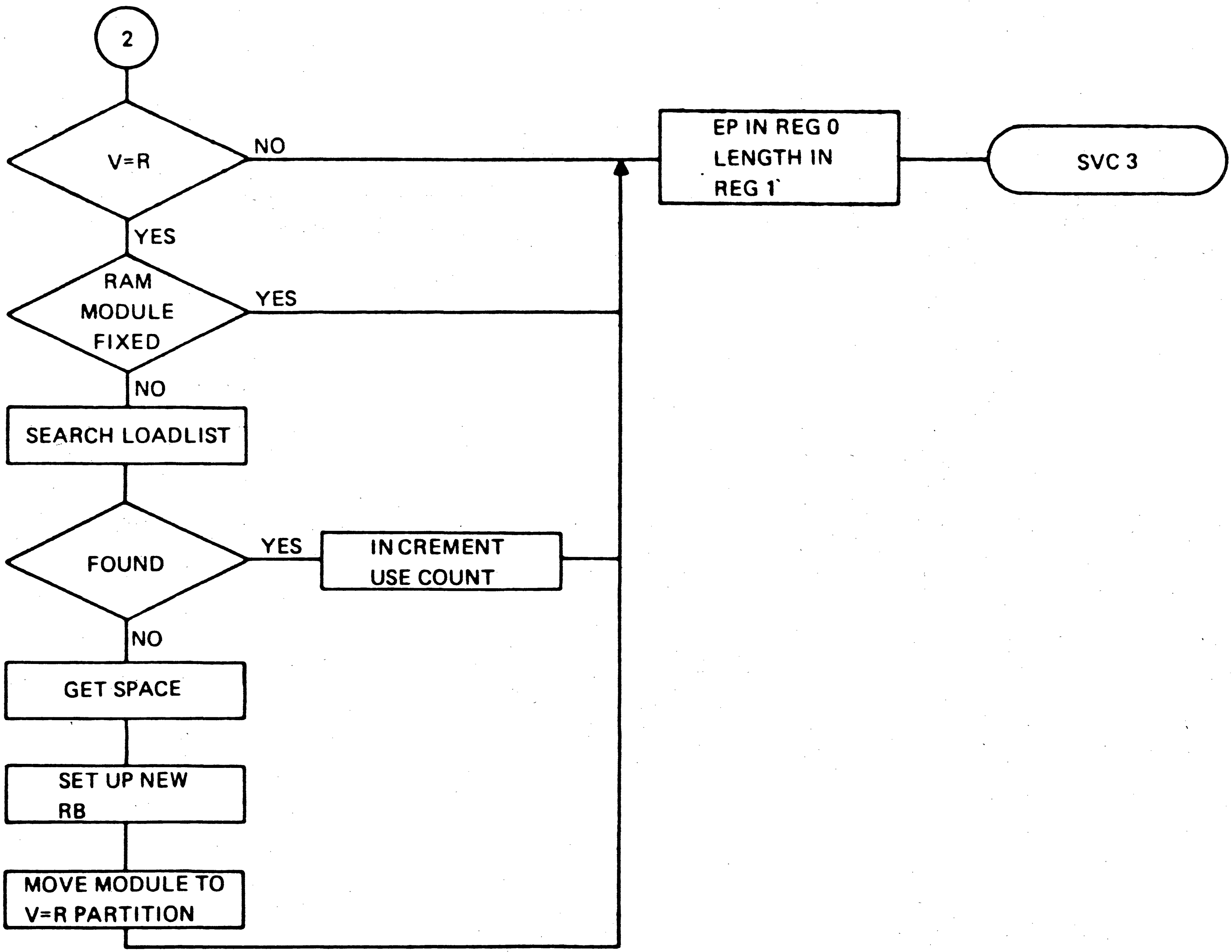


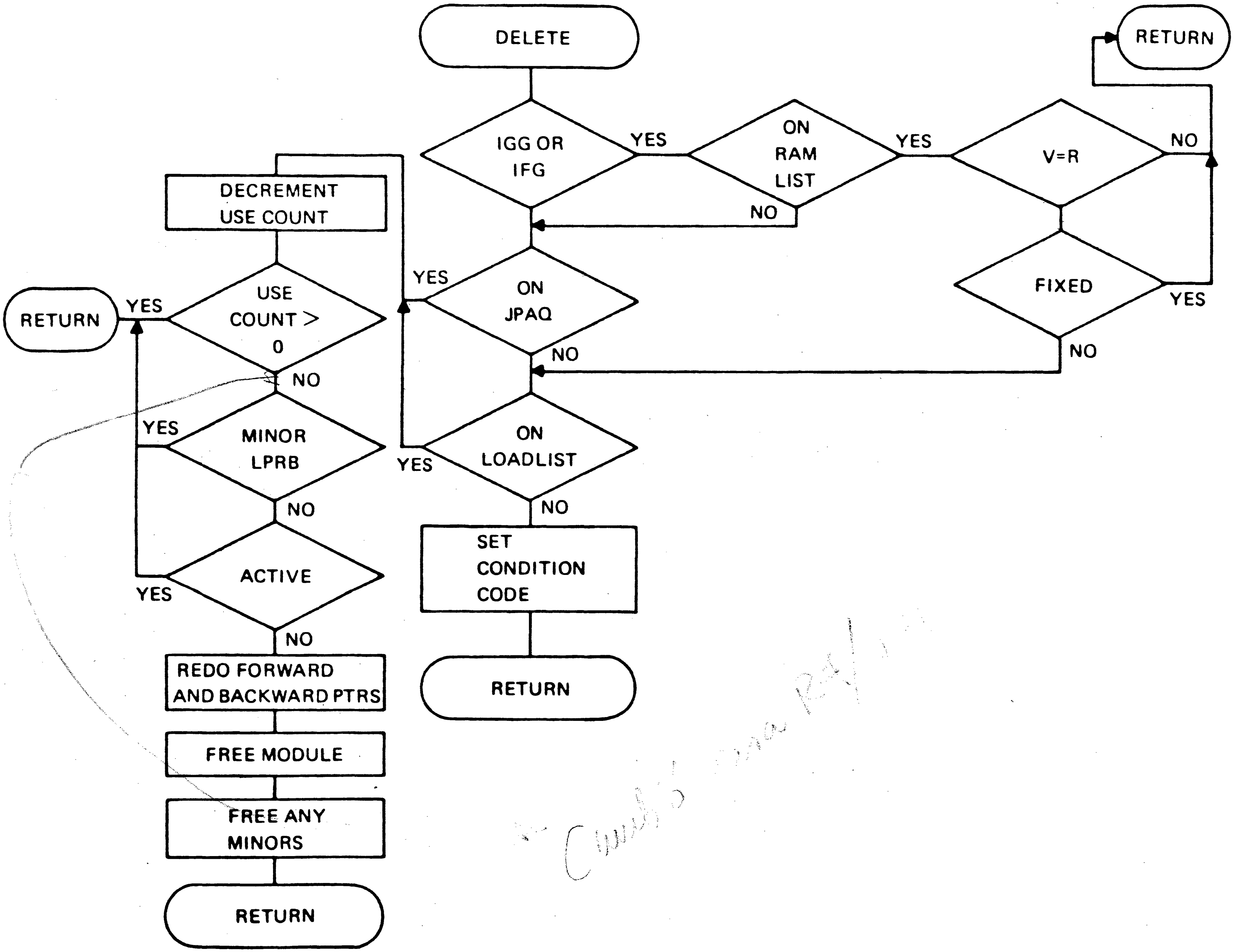


VS1 MODULE SEARCH LOGIC

no mater if the macro is XCTL, LINK, LOAD, — . The search logic is the same!

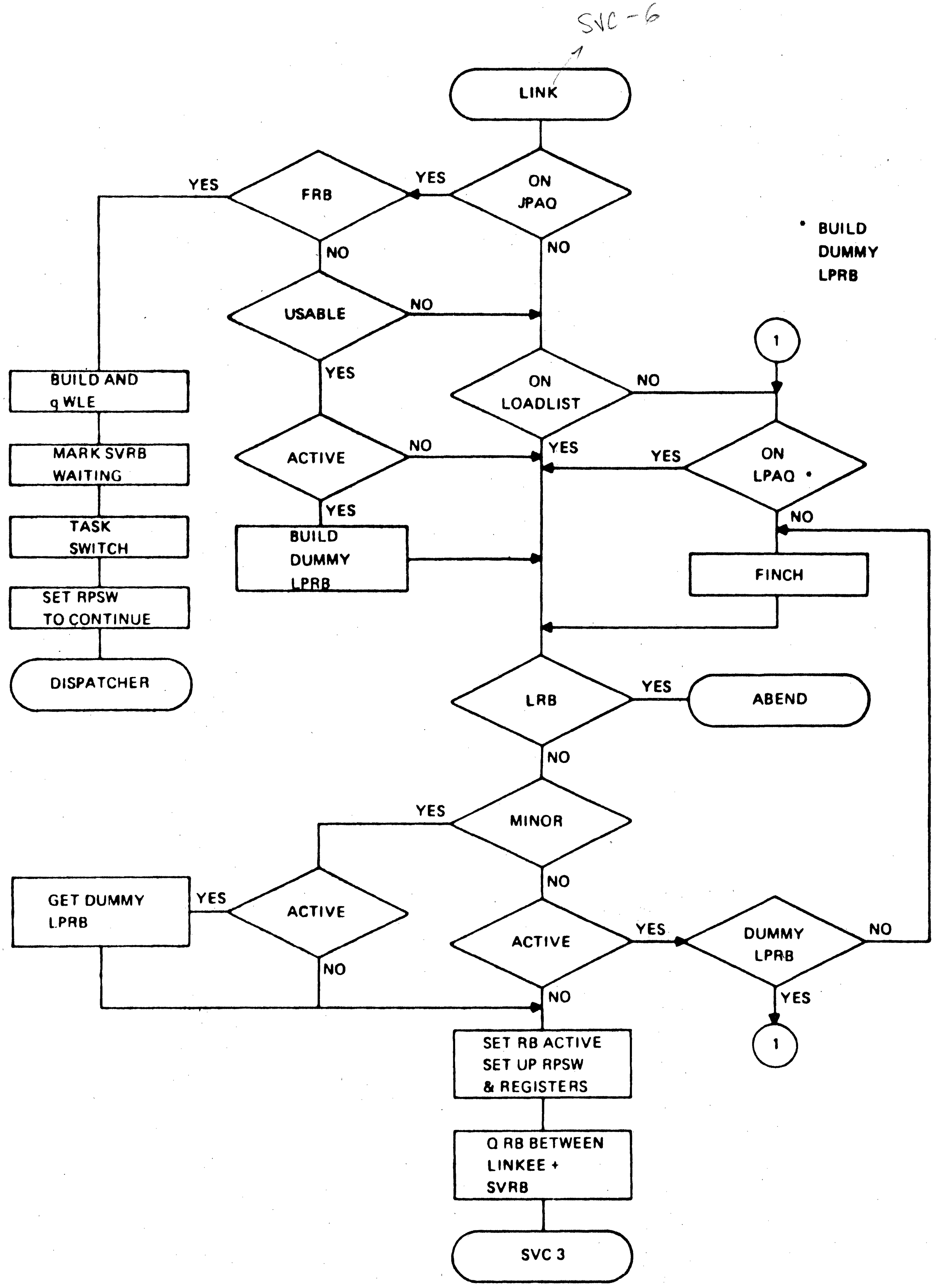

LOAD
ON
JPAO
YES
NO
BUILD FRB
q ON JPAQ
ACCESS
METHOD
YES
NO
ON
RAM
LIST
YES
2
NO
XLOAD1
ON
LOADLIST
YES
FREE
FRB
NO
ON
JPAQ
YES
PUT DUMMY
LPRB ON
JPAQ
FREE FRB
NO
WAS
FRB
YES
INCREMENT
USE COUNT
PUT EP IN R0,
LENGTH IN R1
SVC 3
BUILD AND
q WLE
MARK SVRB
WAITING
TASK
SWITCH
SET RPSW
TO CONTINUE
LOAD
DISPATCHER
NO
FINCH
FREE FRB
REENTRANT
NO
PUT LPRB
ON LOADLIST
XLOAD1
YES
PUT LPRB
ON JPAQ


2
V=R
NO
YES
RAM MODULE FIXED
YES
NO
SEARCH LOADLIST
FOUND
YES
INCREMENT USE COUNT
NO
GET SPACE
SET UP NEW RB
MOVE MODULE TO V=R PARTITION
EP IN REG 0 LENGTH IN REG 1
SVC 3

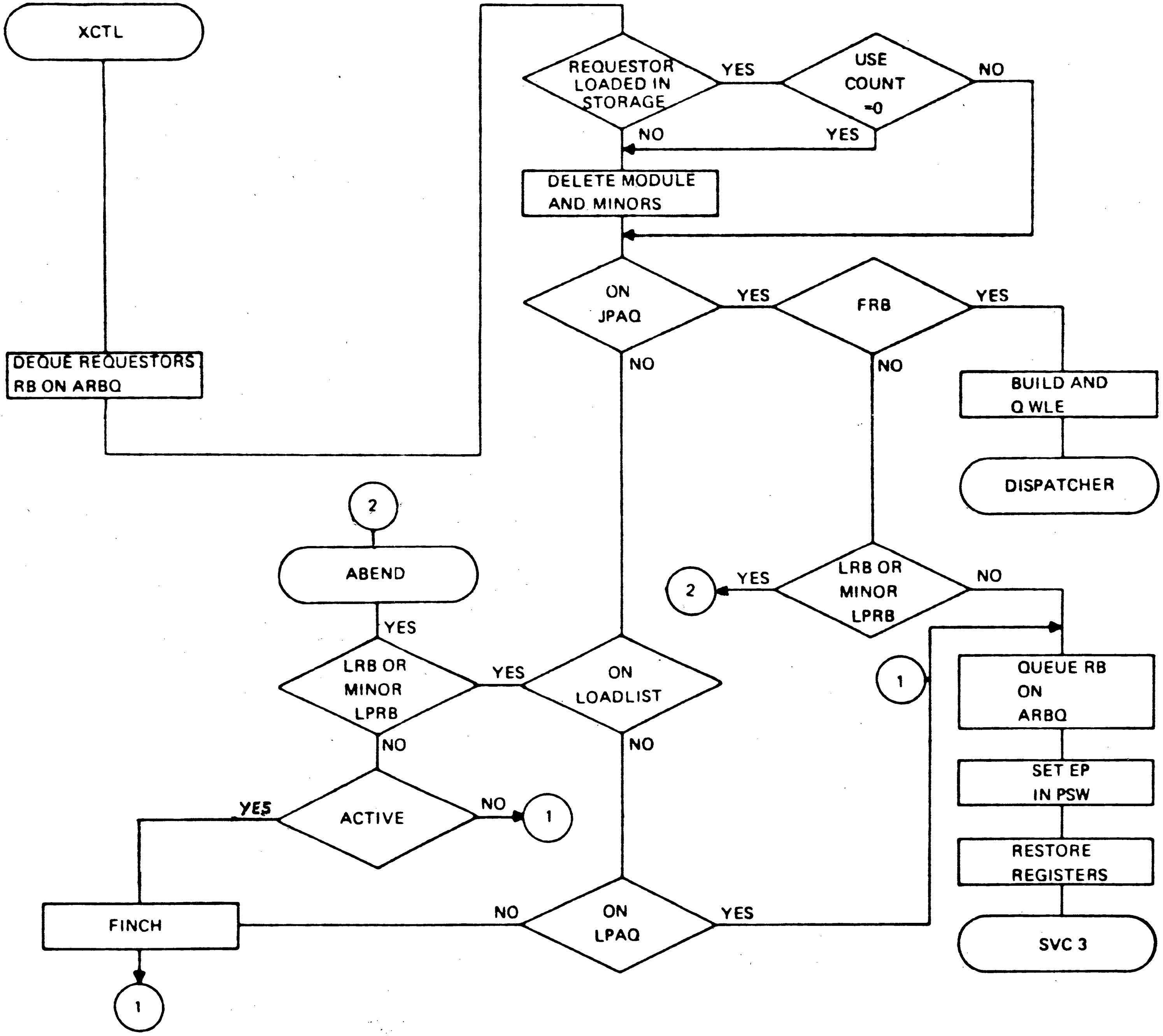

DELETE
IGG OR IFG
YES
NO
ON RAM LIST
YES
NO
V=R
NO
FIXED
YES
NO
RETURN
ON JPAQ
YES
NO
ON LOADLIST
YES
NO
SET CONDITION CODE
RETURN
DECREMENT USE COUNT
USE COUNT > 0
YES
NO
RETURN
MINOR LPRB
YES
NO
ACTIVE
YES
NO
REDO FORWARD AND BACKWARD PTRS
FREE MODULE
FREE ANY MINORS
RETURN

Chuck's [illegible]

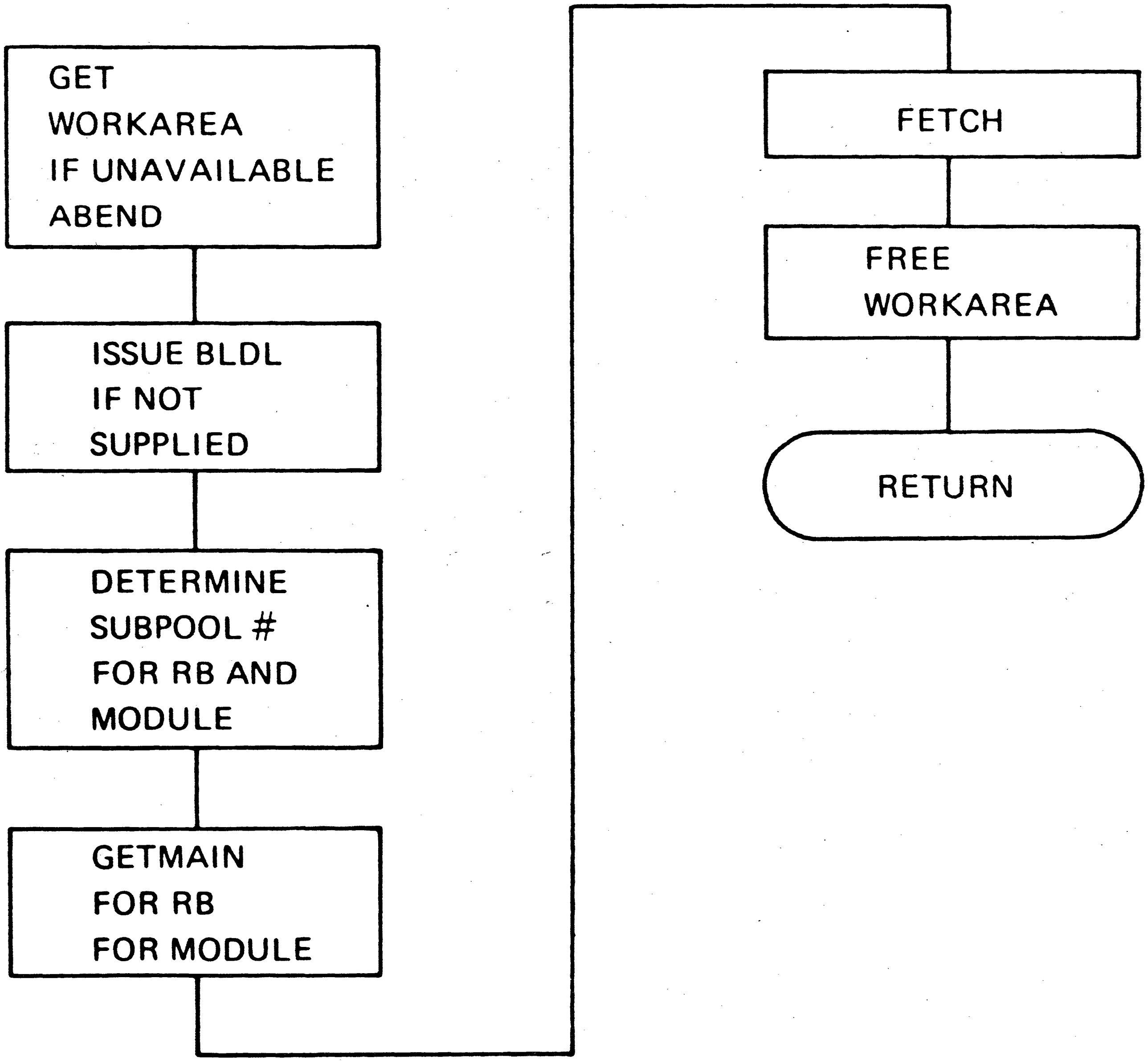
IDENTIFY Crea un nuevo Entry Point



SVC 7


XCTL
DEQUE REQUESTORS RB ON ARBQ
REQUESTOR LOADED IN STORAGE
YES
NO
USE COUNT =0
NO
YES
DELETE MODULE AND MINORS
ON JPAQ
YES
NO
FRB
YES
NO
BUILD AND Q WLE
DISPATCHER
2
ABEND
YES
LRB OR MINOR LPRB
YES
NO
ON LOADLIST
NO
LRB OR MINOR LPRB
YES
NO
2
1
QUEUE RB ON ARBQ
SET EP IN PSW
RESTORE REGISTERS
SVC 3
ACTIVE
YES
NO
1
FINCH
1
ON LPAQ
NO
YES

FINCH – LOAD, LINK, XCTL – USER MODULES



GET
WORKAREA
IF UNAVAILABLE
ABEND

ISSUE BLDL
IF NOT
SUPPLIED

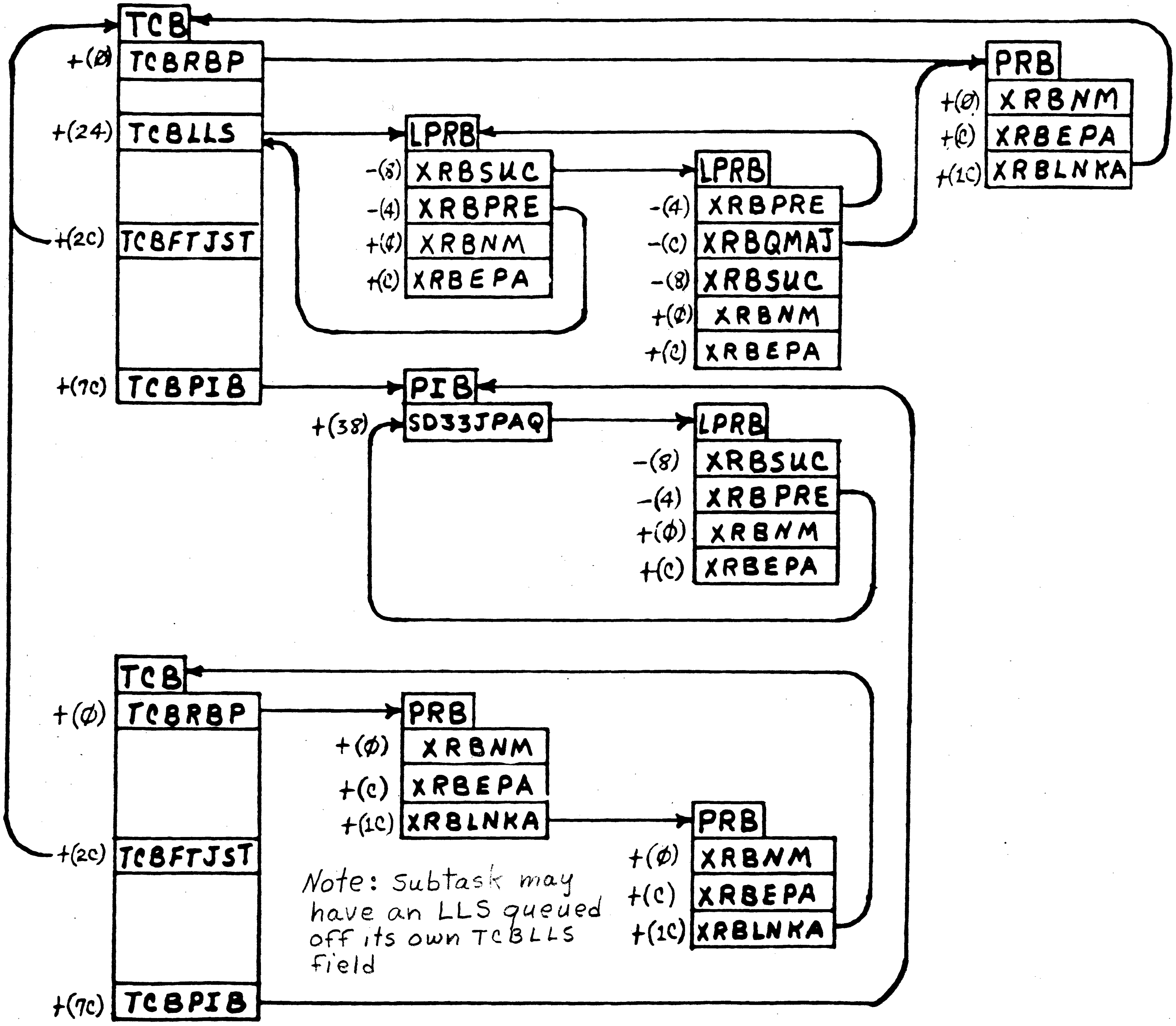
DETERMINE
SUBPOOL #
FOR RB AND
MODULE

GETMAIN
FOR RB
FOR MODULE

FETCH

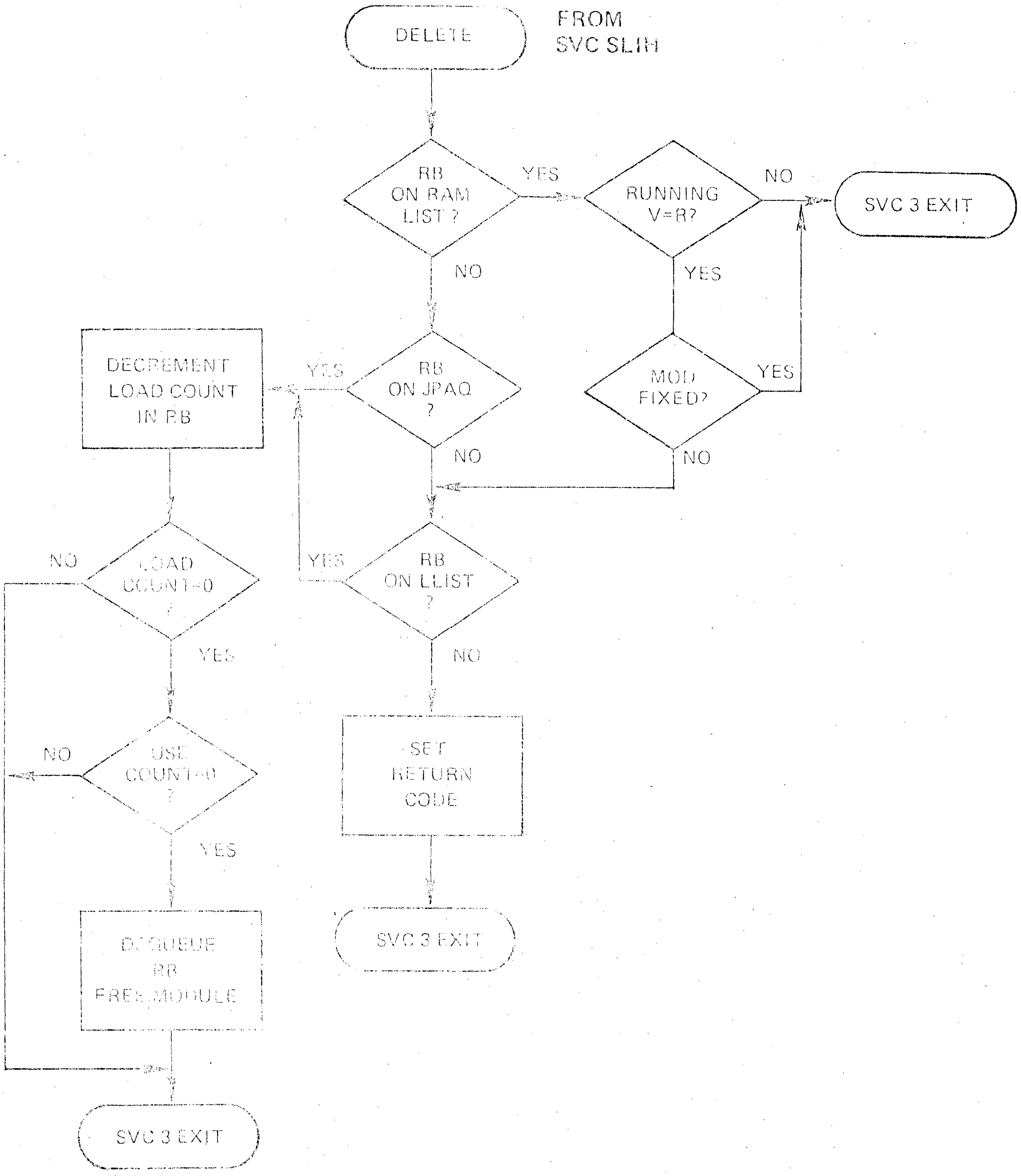
FREE
WORKAREA

RETURN



Note: Subtask may
have an LLS queued
off its own TCBLLS
field

DELETE LOGIC

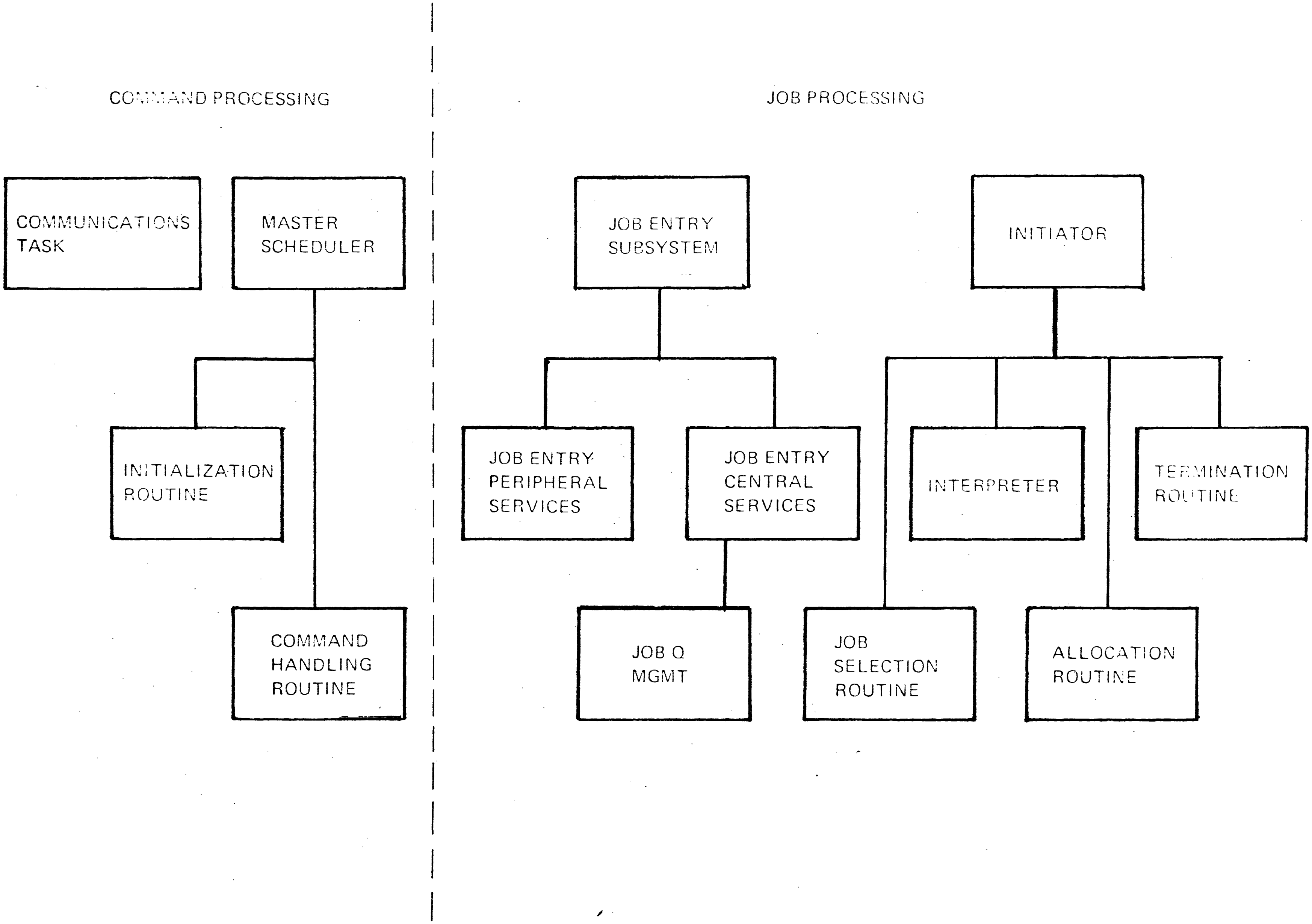


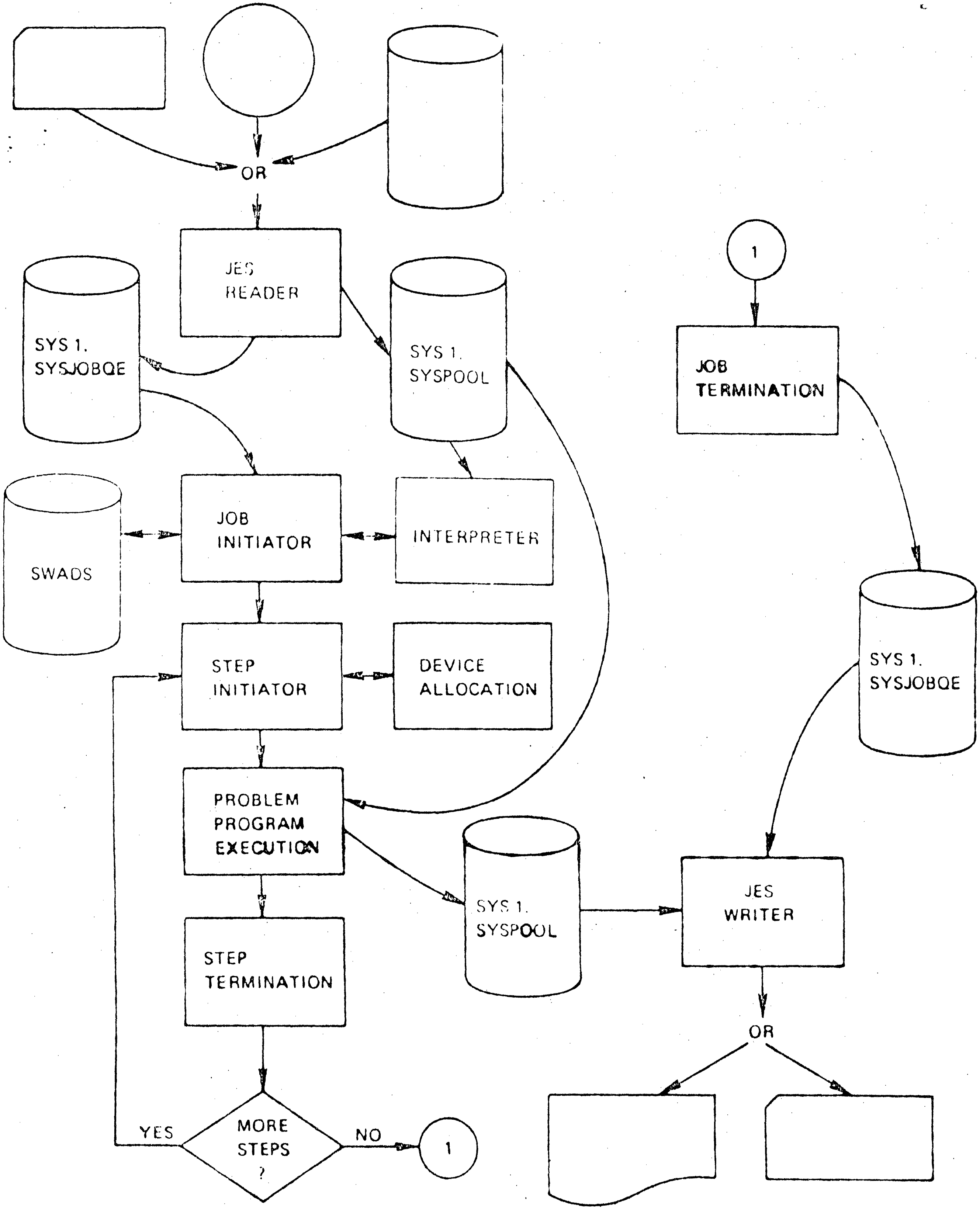
JOB MANAGEMENT

References:

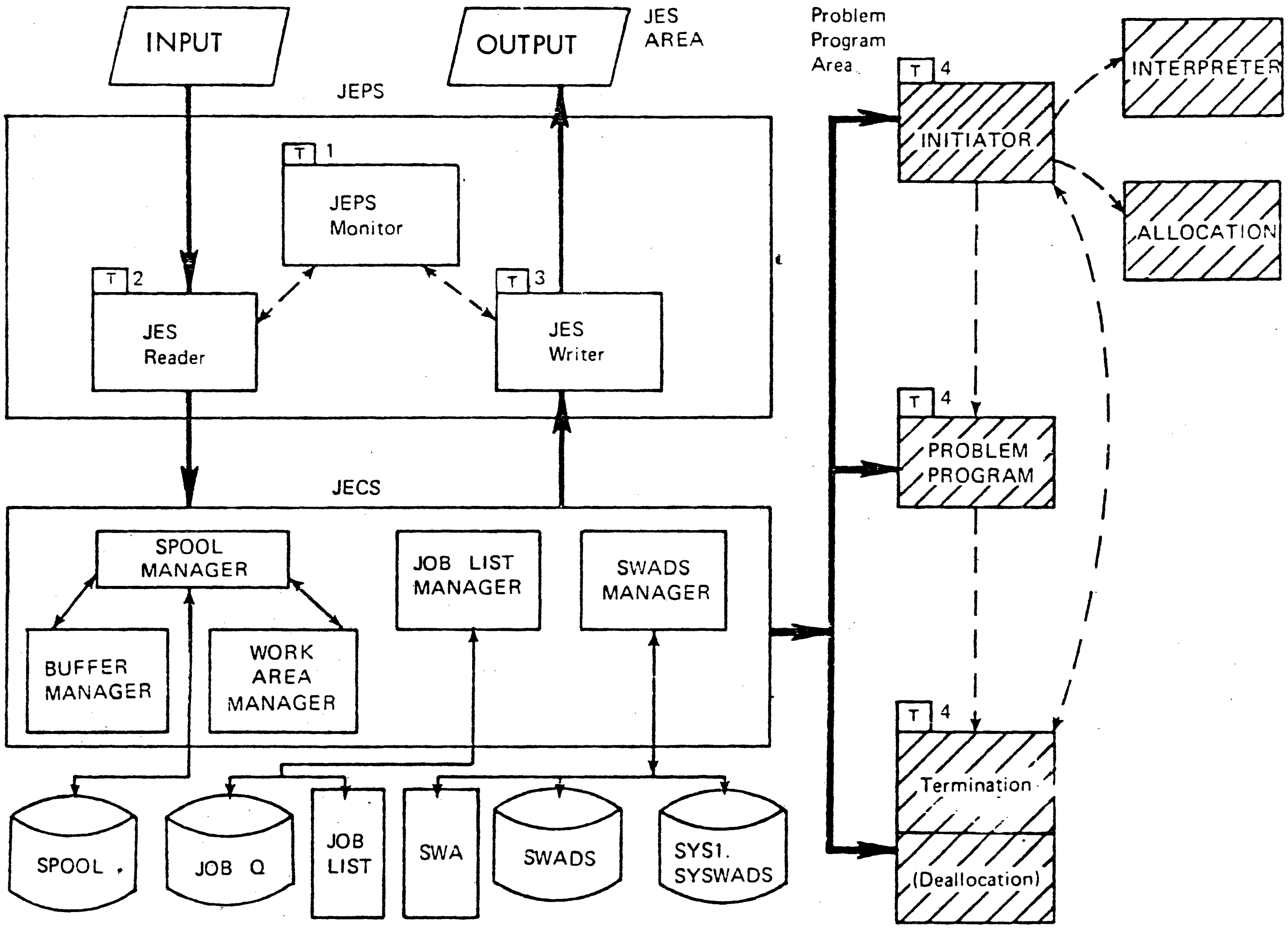
OS/VS1 Planning and Use Guide	GC24-5090
OS/VS1 Job Management Logic	SY24-5161
OS/VS1 System Generation Reference	GC26-3791
OS/VS1 Debugging Guide	GC24-5093
System Data Areas	SY28-0605
OS/VS System Management Facilities	GC35-0004

OS/VS1 JOB MANAGEMENT ORGANIZATION



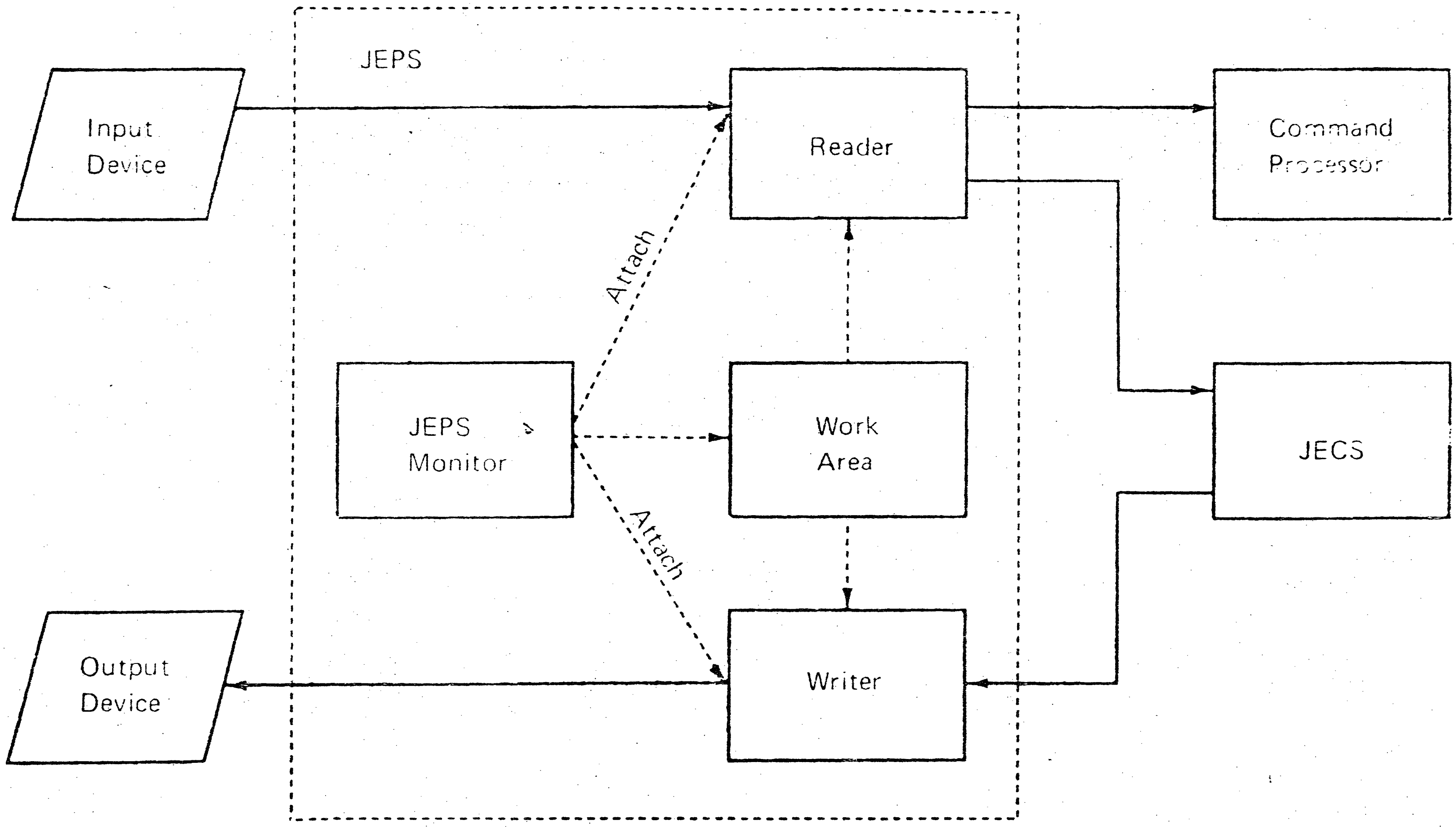

OR
JES
READER
SYS 1.
SYSJOBQE
SYS 1.
SYSPOOL
1
JOB
TERMINATION
SWADS
JOB
INITIATOR
INTERPRETER
STEP
INITIATOR
DEVICE
ALLOCATION
SYS 1.
SYSJOBQE
PROBLEM
PROGRAM
EXECUTION
SYS 1.
SYSPOOL
JES
WRITER
STEP
TERMINATION
OR
YES
MORE
STEPS
?
NO
1

OS/VS1 JES AND JOB SCHEDULER COMPONENTS



T : THE FUNCTION RUNS AS A TASK

Job Entry Peripheral Services JEPS



JEPS MONITOR FUNCTIONS

* Gains control during IPL from Master Scheduler

* Obtains a work area equal to 2K x number of RDRs and WTRs specified at SYSGEN or IPL time

* Initializes portions of JESCT — tabla. comunicaciones : bloque de control para comunicación entre programas.

* OPEN's SYS1.PROCLIB

* GOES into WAIT state

* POST'ed on "S RDR" or "S WTR" command

* Formats work area for RDR or WTR
Reads its JCL from SYS1.PROCLIB
ATTACH'es RDR or WTR
Goes back into WAIT State

JES READER OPERATION

metodos de acceso del Jes para leer en los Readers

* Reads input using JAM or QSAM

Indice Parametros

* Scans JCL to isolate

  - * JOB card - assigns ID, time-stamp, class, priority; handles ISSP parameters
  - * JCL - asks JECS to write JCL data set on SPOOL
  - * SYSIN - asks JECS to write SYSIN d.s. on SPOOL
  - * PROCS - USER or in-stream Procedures are written on SPOOL
  - * COMMANDS - commands between JOBS are passed to Command Processing Routines

* Builds DER 1SK entry record and asks JECS to enqueue JOB

* Provides statistics to SMF (# SYSIN Records)

* Supports TYPRUN=SCAN request

* If EOF is sensed on tape or disk, RDR is closed

el beginning Record
for all the data sets.

WRITER FUNCTIONS

* Supports Printers, Punches, Tapes

* Multi - Tasked

* Uses JAM for unit record devices

* Uses QSAM for tape

* Interfaces with JECS to get Spool records

* Links to user-written separator routines

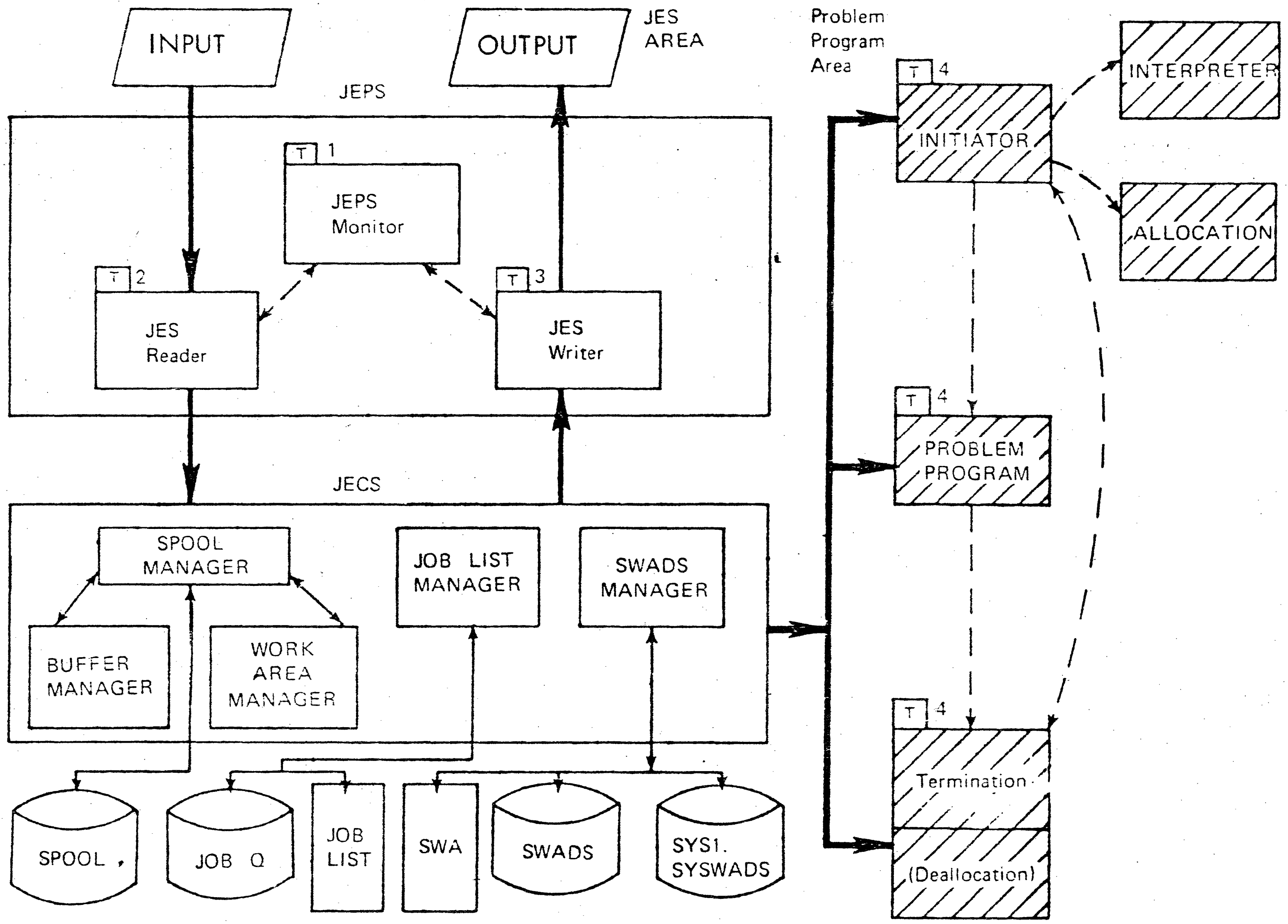
* Provides Accounting Information to SMF

* Supports "WRITER" Command

* OUTLIM PARAMETER

* COPIES PARAMETER

OS/VS1 JES AND JOB SCHEDULER COMPONENTS



T : THE FUNCTION RUNS AS A TASK

SYS1.SYSPOOL

* Data set(s) may reside on as many as 10 volumes
* Each data set must consist of only one extent
* It may be formatted at IPL time
* Logical Cylinders are used as allocation units
* Threshold value for spool capacity is monitored
* Contains INPUT data sets

  JCL / SYSIN / USER and IN-STREAM PROCS
* Contains OUTPUT Data sets

  SYSOUT / MESSAGES / LOG

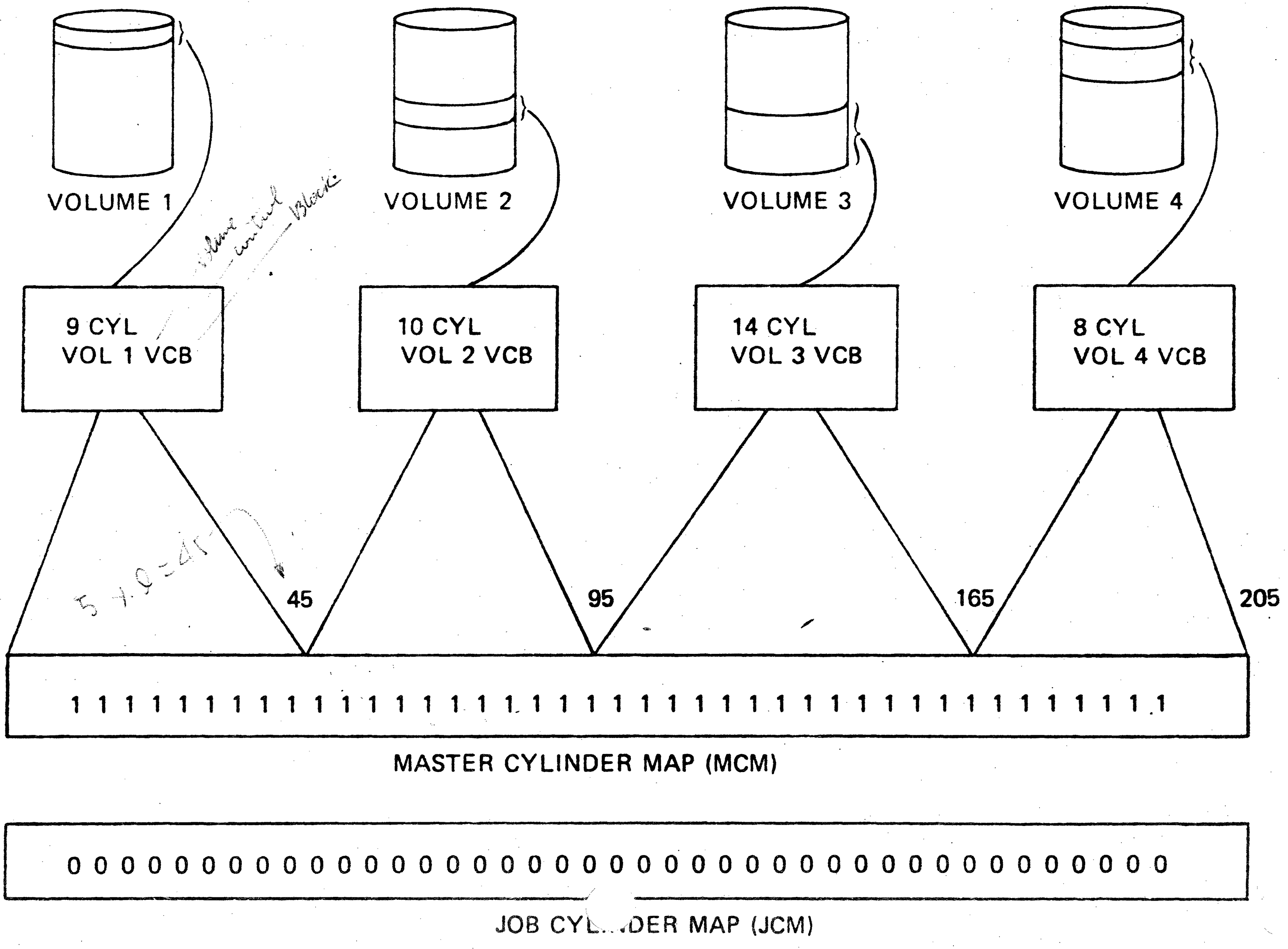
Default SPOOL Device Allocation

Type	Phys Trk/Log. Cyl	Bytes
2314	5	36,460
3330	3	39,090
2305-2	3	43,980

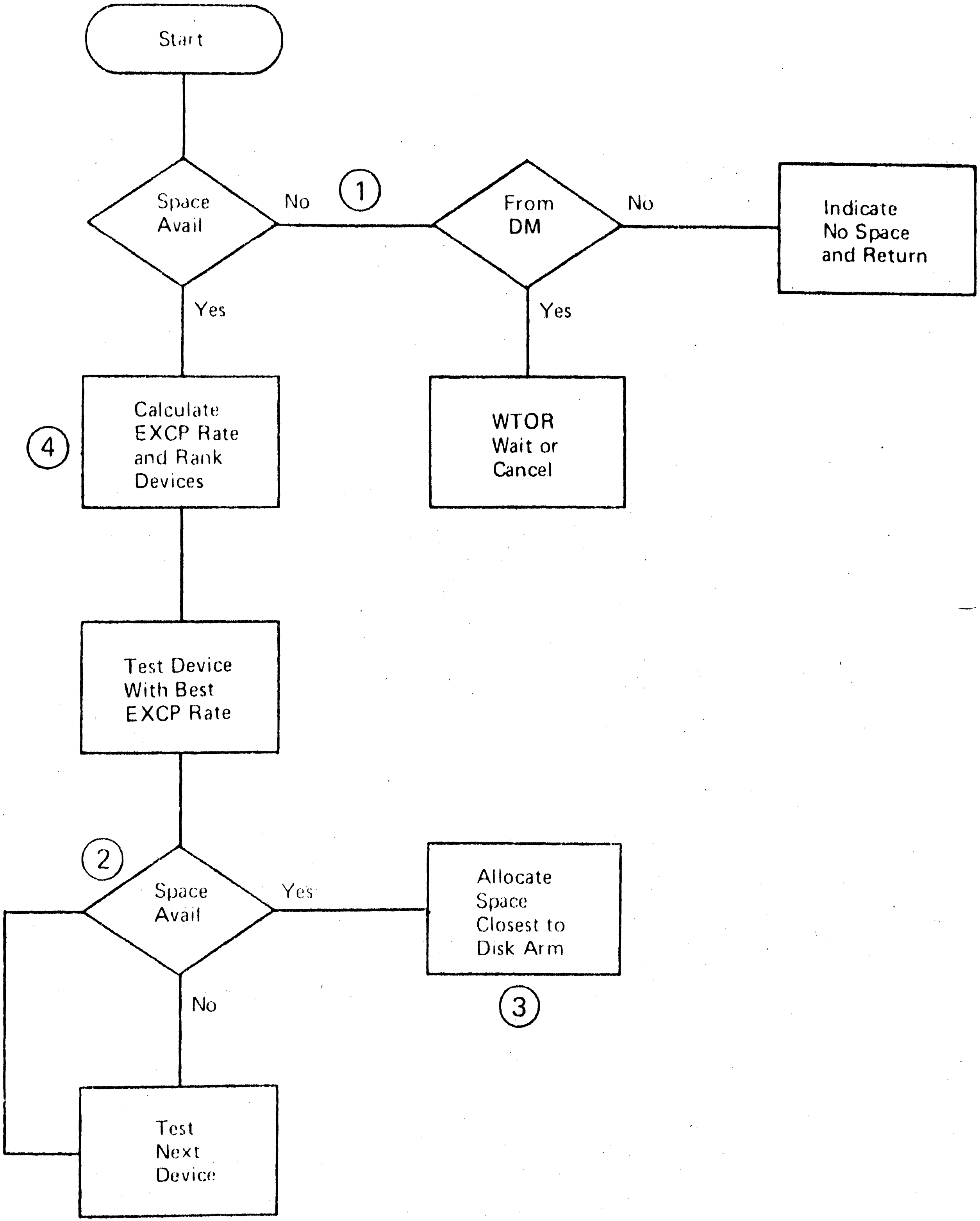
example for 2314

BIT MAP ALLOCATION

10.



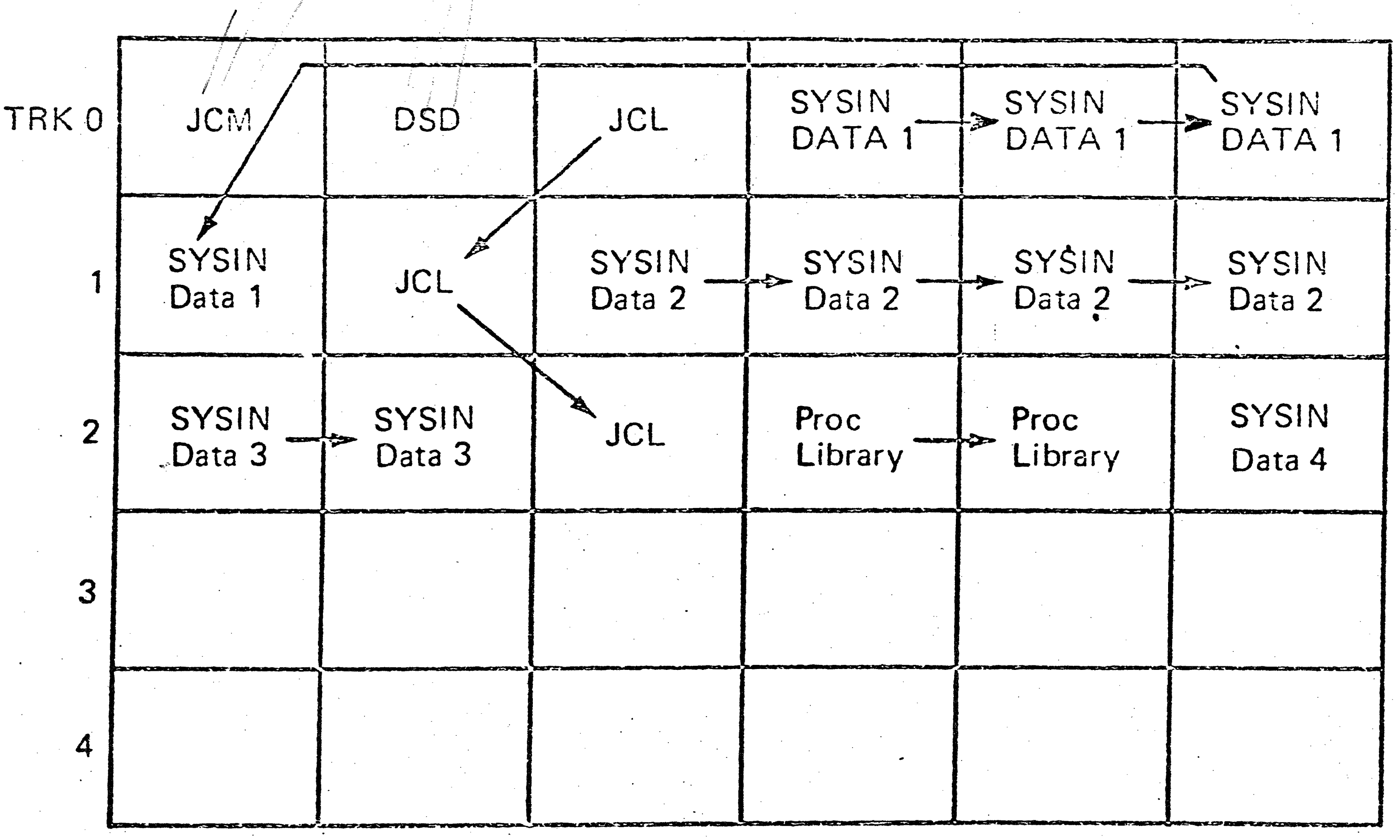
SPOOL Allocation


Start
Space Avail
No
1
From DM
No
Indicate No Space and Return
Yes
Yes
Calculate EXCP Rate and Rank Devices
4
WTOR Wait or Cancel
Test Device With Best EXCP Rate
2
Space Avail
Yes
Allocate Space Closest to Disk Arm
3
No
Test Next Device

Job ~~cyl~~ Cylinder map

Data set Directory

SPOOL Format: Input (2314)



TRK 0	JCM	DSD	JCL	SYSIN DATA 1	SYSIN DATA 1	SYSIN DATA 1
1	SYSIN Data 1	JCL	SYSIN Data 2	SYSIN Data 2	SYSIN Data 2	SYSIN Data 2
2	SYSIN Data 3	SYSIN Data 3	JCL	Proc Library	Proc Library	SYSIN Data 4
3						
4						

logical cylinder

SYS1.SYSPOOL

DATA SET NAMES

JLxxxx00 - Input JCL Statements

SMxxxxyy - SYSTEM and WTP messages

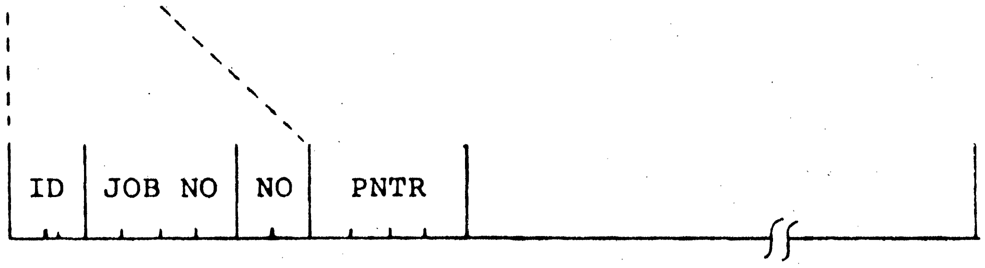
SIxxxxww - SYSIN

SOxxxxzz - SYSOUT

PLxxxx00 - Proceedures

LXxxxx00 - SYSLOGX (LOG)

LYxxxx00 - SYSLOGY (LOG)



xxxx = Four byte binary internal Job Identifier

yy = Two byte binary unique value for each Job Step

ww = Two byte binary unique value for each SYSIN request

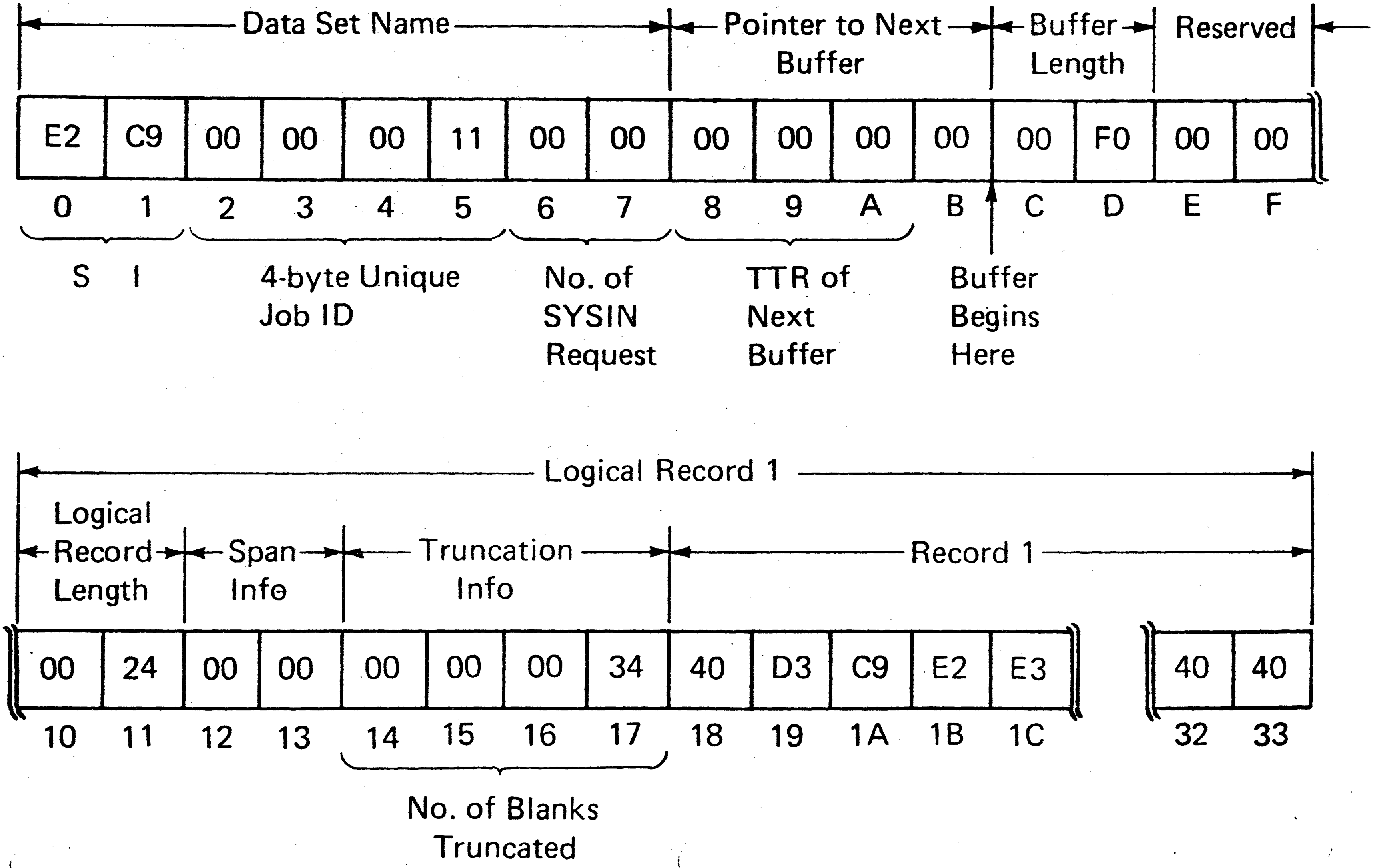
zz = Two byte binary unique value for each SYSOUT request

Truncation Example

Position on Control Card (SYSIN)

```
1 . . . . . . . . . . . . . . . . . . . . . 40 . . . . . . . . . . . . . . . . . . . . . 80

   LISTCTLG     VOL=2314=SYSRESb̸b̸ . . . . . . . . . . . . . . . . . . . . b̸
   LISTVTOC     DUMP,VOL=2314=SYSLNKb̸b̸ . . . . . . . . . . . . . . . b̸
   LISTVTOC     DUMP,VOL=2314=SYSPOLb̸b̸ . . . . . . . . . . . . . . . b̸
   LISTVTOC     DUMP,VOL=2314=SYSQUEb̸b̸ . . . . . . . . . . . . . . . b̸
   LISTVTOC     DUMP,VOL=2314=SYSRESb̸b̸ . . . . . . . . . . . . . . . b̸
   LISTVTOC     DUMP,VOL=2314=SYSPAGb̸b̸ . . . . . . . . . . . . . . . b̸
```

Logical Data Set Record



Data Set Name								Pointer to Next Buffer				Buffer Length		Reserved	
E2	C9	00	00	00	11	00	00	00	00	00	00	00	F0	00	00
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S	I	4-byte Unique Job ID				No. of SYSIN Request		TTR of Next Buffer				Buffer Begins Here			

Logical Record 1														
Logical Record Length		Span Info		Truncation Info				Record 1						
00	24	00	00	00	00	00	34	40	D3	C9	E2	E3	40	40
10	11	12	13	14	15	16	17	18	19	1A	1B	1C	32	33
				No. of Blanks Truncated										

Es parte de la tabla JLMDA

Es un checkpoint data set.

TYPES OF SYS1.SYSJOBQE RECORDS

CCR — CONFIGURATION CONTROL RECORD Describe el Job Queue

JLHRT — JOBLIST HOLD/RELEASE TABLE Describe el físical Track del Job Queue

DCA — DATA CONTROL AREA

DER — DISK ENTRY RECORD — cuando por el Reader y puesto en JQ. Contiene el JOB NAME en el ID.

DSB — DATA SET BLOCK — 1 por cada SYSOUT CARD

JACT — JOB ACCOUNTING CONTROL TABLE

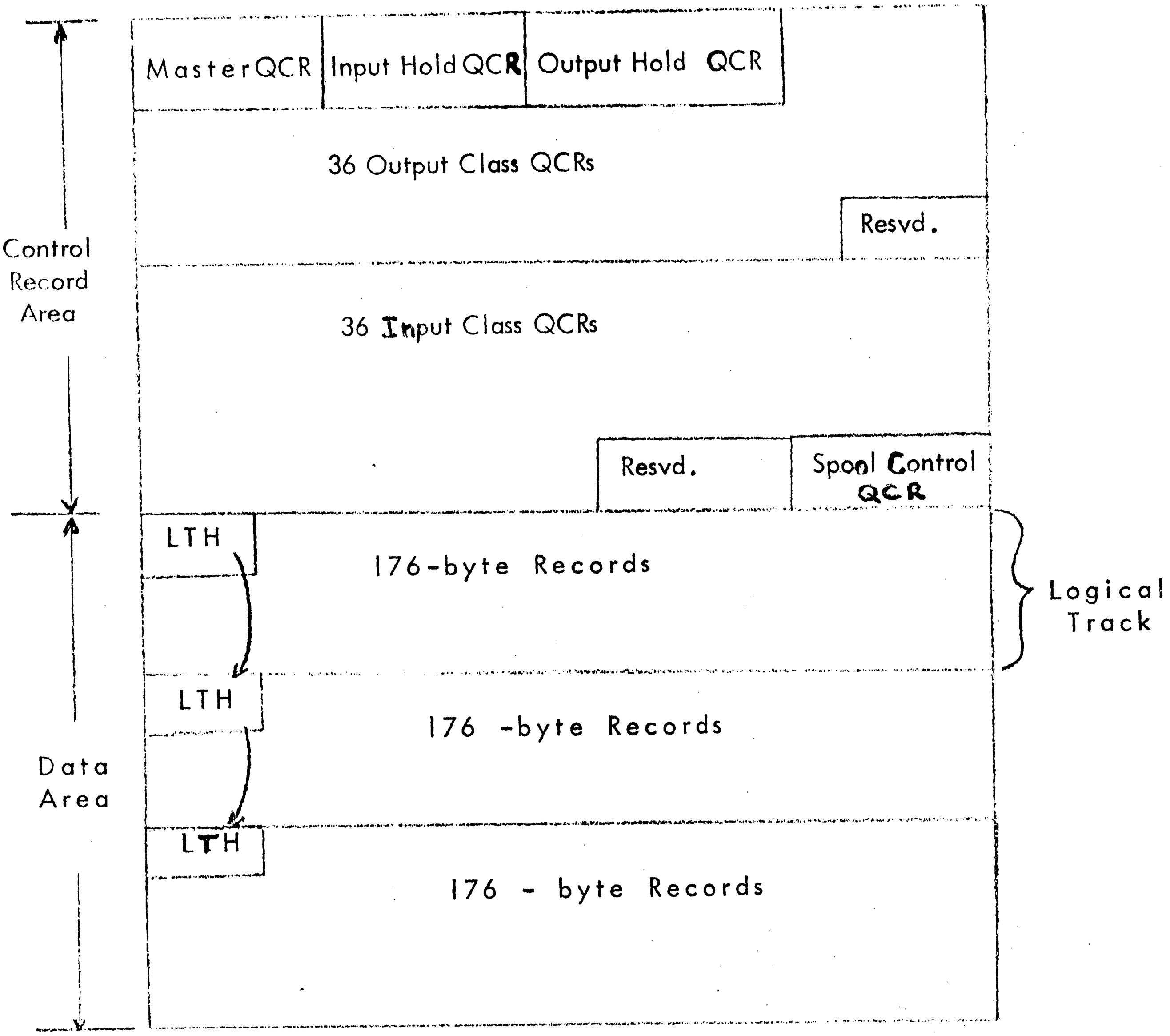
JMR — JOB MANAGEMENT RECORD → # SYSIN RECORDS
Records a [illegible]

RTBL — ROUTE TABLE

SCD — SYSOUT CLASS DIRECTORY

all these records 177 ~~records~~ bytes long

SYS1.SYSWADS FORMAT



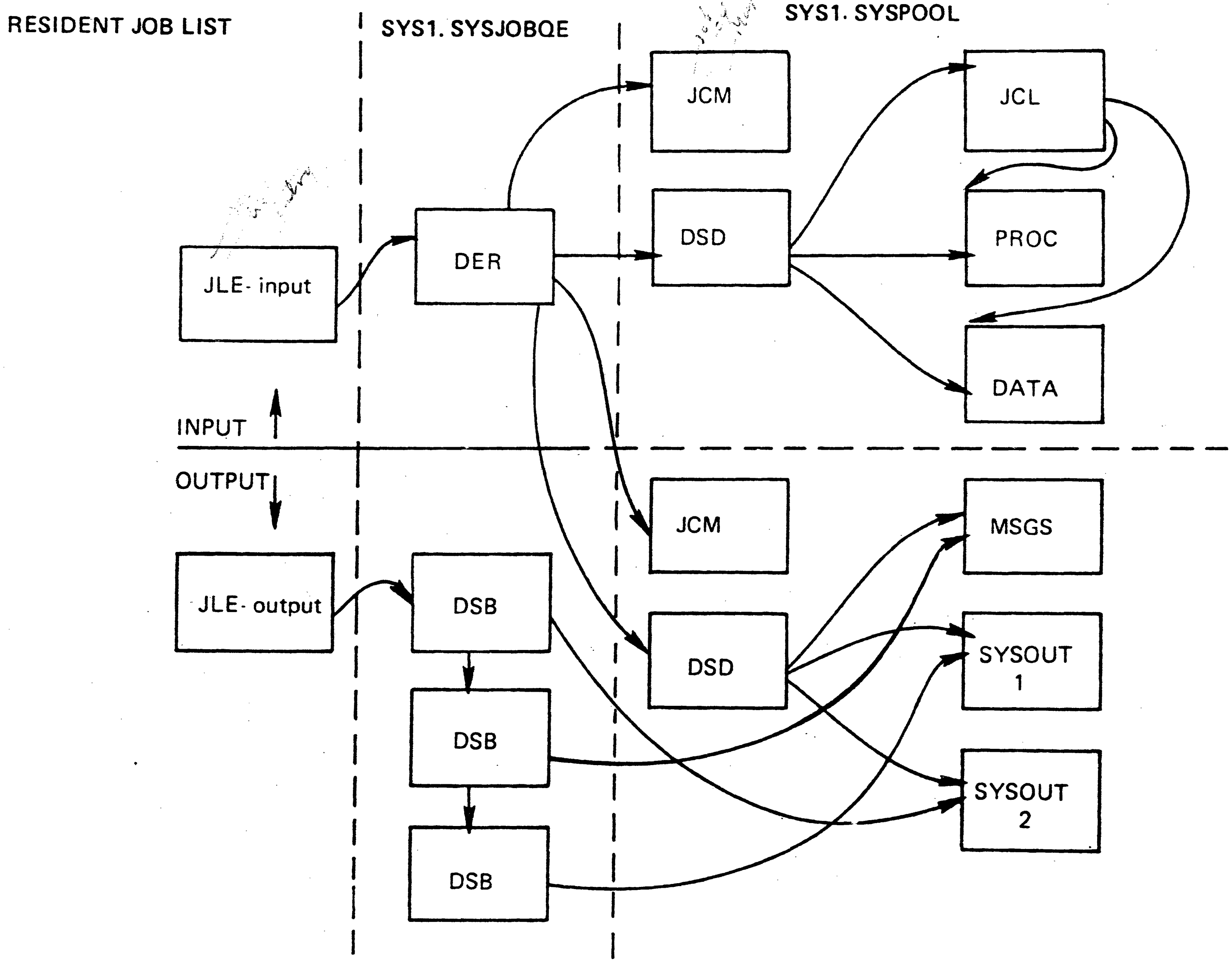
HEX	DEC		
0	0	ADDRESS OF LAST LTH OF HIGHEST 2 PRIORITY ENTRY ON QUEUE	14 2
4	4	13 2	12 2
8	8	11 2	10 2
C	12	9 2	8 2
10	16	7 2	6 2
14	20	5 2	4 2
18	24	3 2	2 2
1C	28	1 2	0 2
20	32	HOLD QUEUE HIGHEST[1] PRIORITY	ADDRESS OF ECB FOR FIRST REQUESTING WORK[3]

QCR RECORD FORMAT

JOB LIST MANAGER FUNCTIONS

- ASSIGN, READ/WRITE, DELETE
- DEQUEUE
- RESOURCE BRAKING
- ENQUEUE
- JOB LIST SCAN
- UNCHAIN

JOBLIST/SYS1.SYSJOBQE/SYS1.SYSPOOL RELATIONSHIP



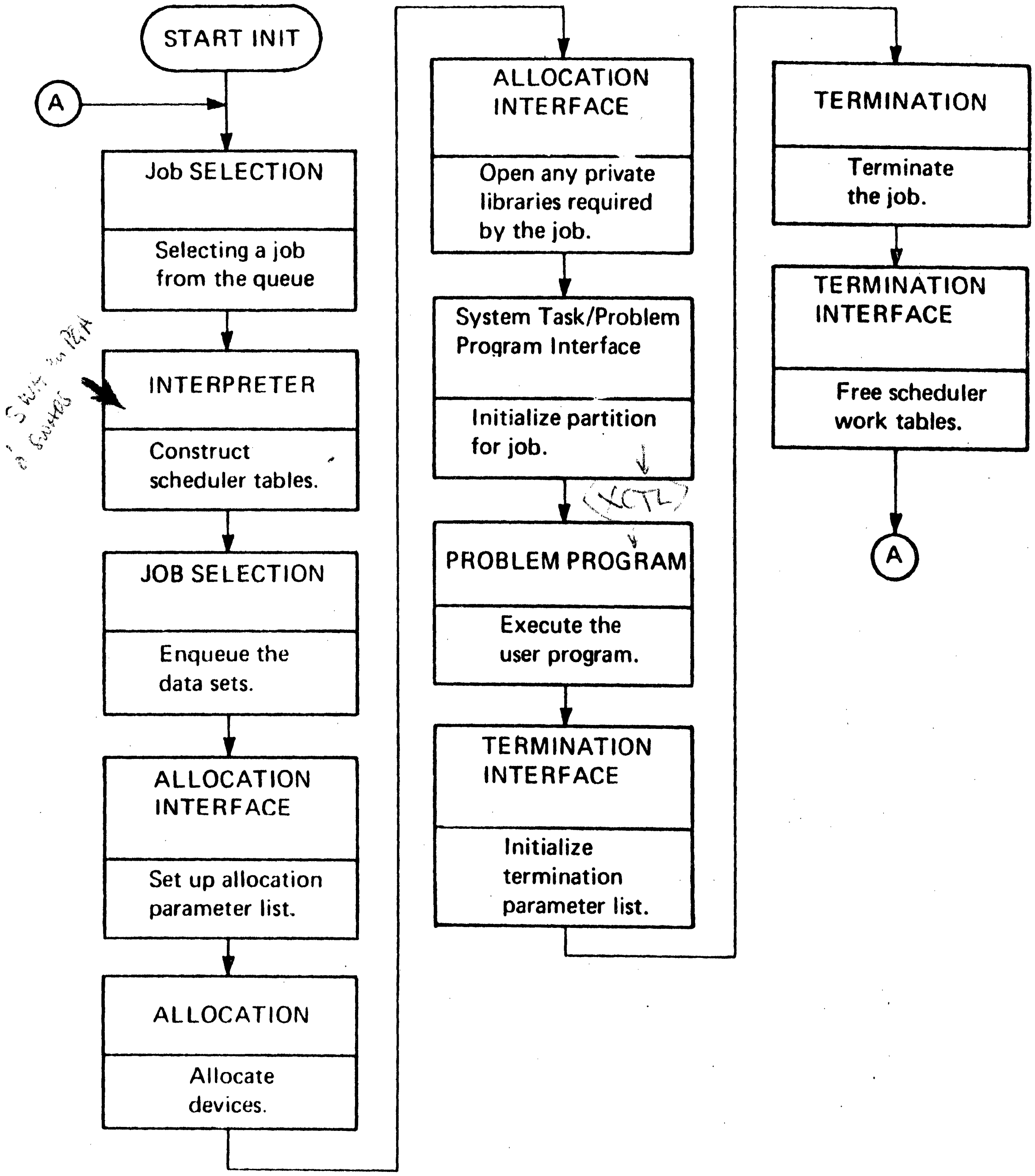
SWADS

Contiene tablas que
contienen JCL que va a usar
el Interpreter

- Created at START INIT
  - Temporary Data Set
  - Dedicated work file for one initiator
- Deleted at STOP INIT
- Contains Scheduler Tables
  (All 176-byte records)
  - JCT Job control Table
  - SCT Step C.T. creada por cada exec card
  - JFCB Job file C.B. Describe cada DD card.
  - SIOT
  - TIOT Task Input output Table. Used for Data set allocation
  - LCT Linkage Control Table

20

Initiator Control Flow

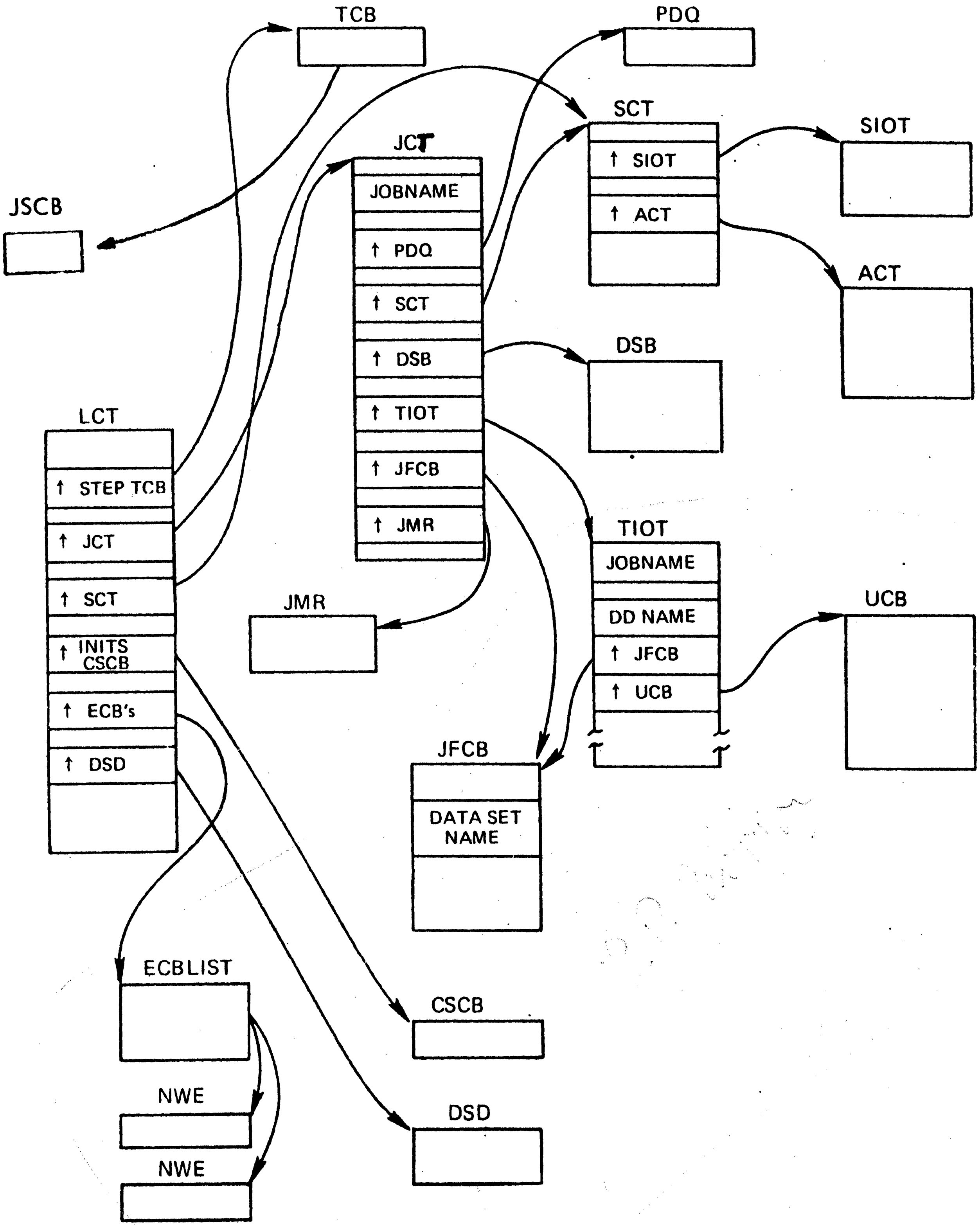

START INIT
A
Job SELECTION
Selecting a job from the queue
SWA in PCA
& SWADS
INTERPRETER
Construct scheduler tables.
JOB SELECTION
Enqueue the data sets.
ALLOCATION INTERFACE
Set up allocation parameter list.
ALLOCATION
Allocate devices.
ALLOCATION INTERFACE
Open any private libraries required by the job.
System Task/Problem Program Interface
Initialize partition for job.
(XCTL)
PROBLEM PROGRAM
Execute the user program.
TERMINATION INTERFACE
Initialize termination parameter list.
TERMINATION
Terminate the job.
TERMINATION INTERFACE
Free scheduler work tables.
A

22

Initiator's ECB List

A (ECB - SVC34)
A (ECB - Job Class A)
A (ECB - Job Class B)
⋮
A (ECB - Job Class O)

SCHEDULER TABLES



DATA SET CHARACTERISTICS

1. TEMPORARY DATA SETS

- NO DSNAME
- DSN=&&NAME
- DISP=(NEW,DELETE)
- BACKWARD REF TO A TEMP. DATA SET

2. NON–TEMP DATA SETS

- HAS NONE OF THE ABOVE ATTRIBUTES
- OLD DATA SETS ARE NON–TEMP.

MOUNT ATTRIBUTES

PERMANENTLY RESIDENT: ARE VOLUMES *NEVER* AVAILABLE TO THE SYSTEM FOR DISMOUNTING.

EX. VOLUMES CONTAINING.

- LINKLIB
- JOBQE
- PROCLIB
- IPL
- ANY PHYSICALLY NON–DEMOUNTABLE DEVICE.

RESERVED: NOT AVAILABLE TO THE SYSTEM FOR DISMOUNTING. OPERATOR CAN REMOVE WITH *UNLOAD.*

REMOVABLE: AT THE DISPOSITION OF THE OPERATING SYSTEM.

USE ATTRIBUTES

PRIVATE: CAN ONLY SATISFY SPECIFIC REQUESTS.

PUBLIC: CAN SATISFY

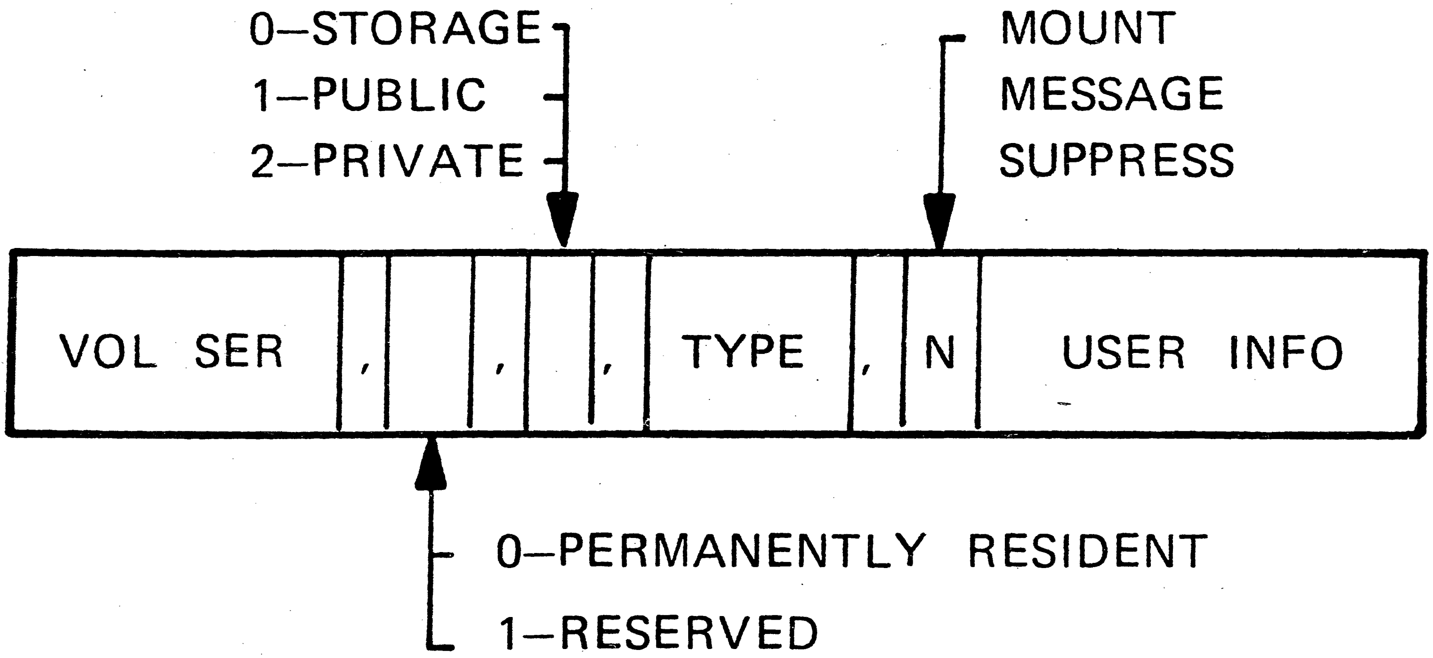
- SPECIFIC REQUESTS
- NON–SPECIFIC REQUESTS FOR TEMP. DS.

STORAGE: CAN SATISFY

- SPECIFIC REQUESTS
- NON–SPECIFIC REQUESTS FOR TEMP D.S. AND NON TEMP DATASETS.

PRESRES

- MEMBER OF PARMLIB
- RECORD 80 BYTES
- EBCDIC CHARACTERS
- MAINTAINED USING IEBUPDTE

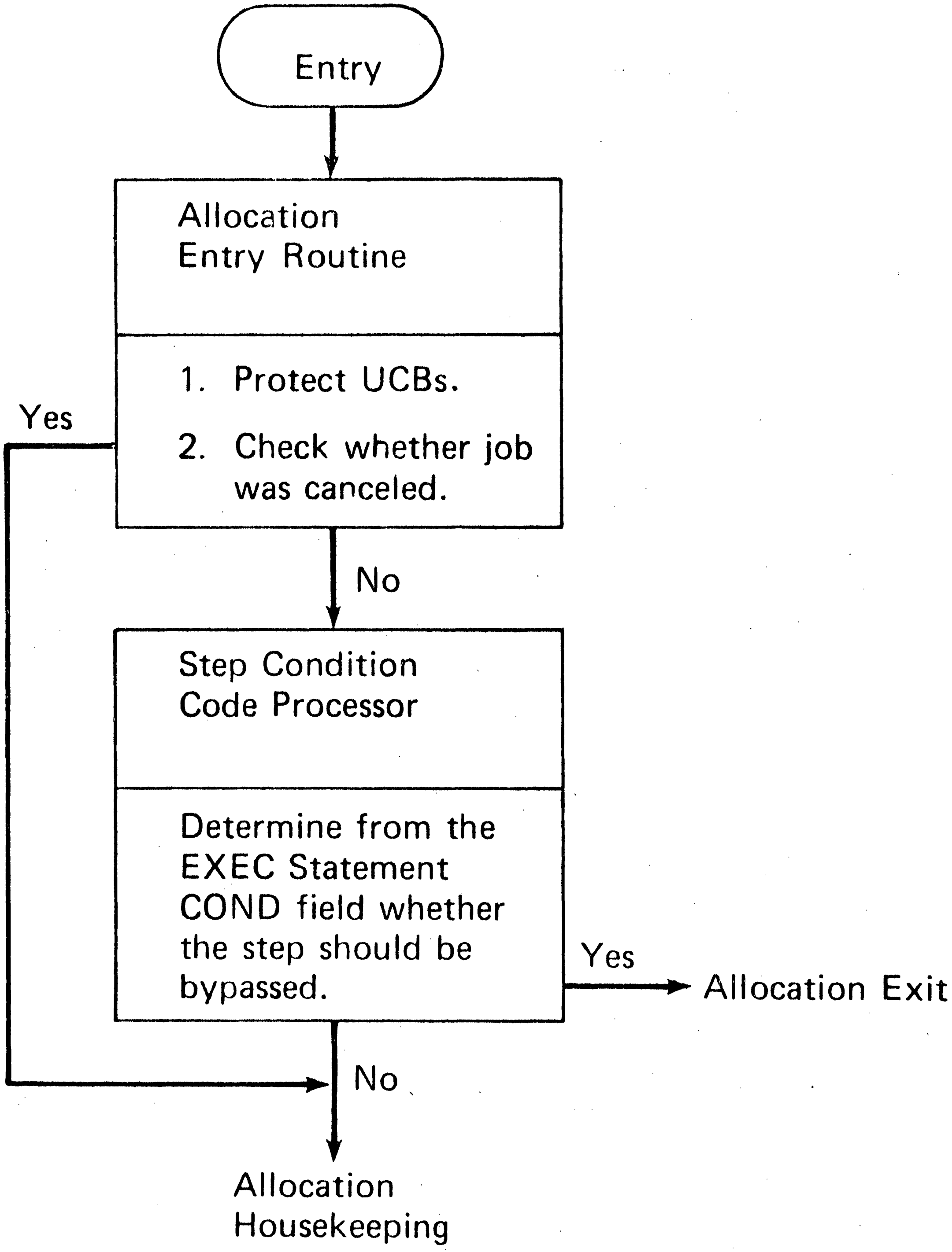


27

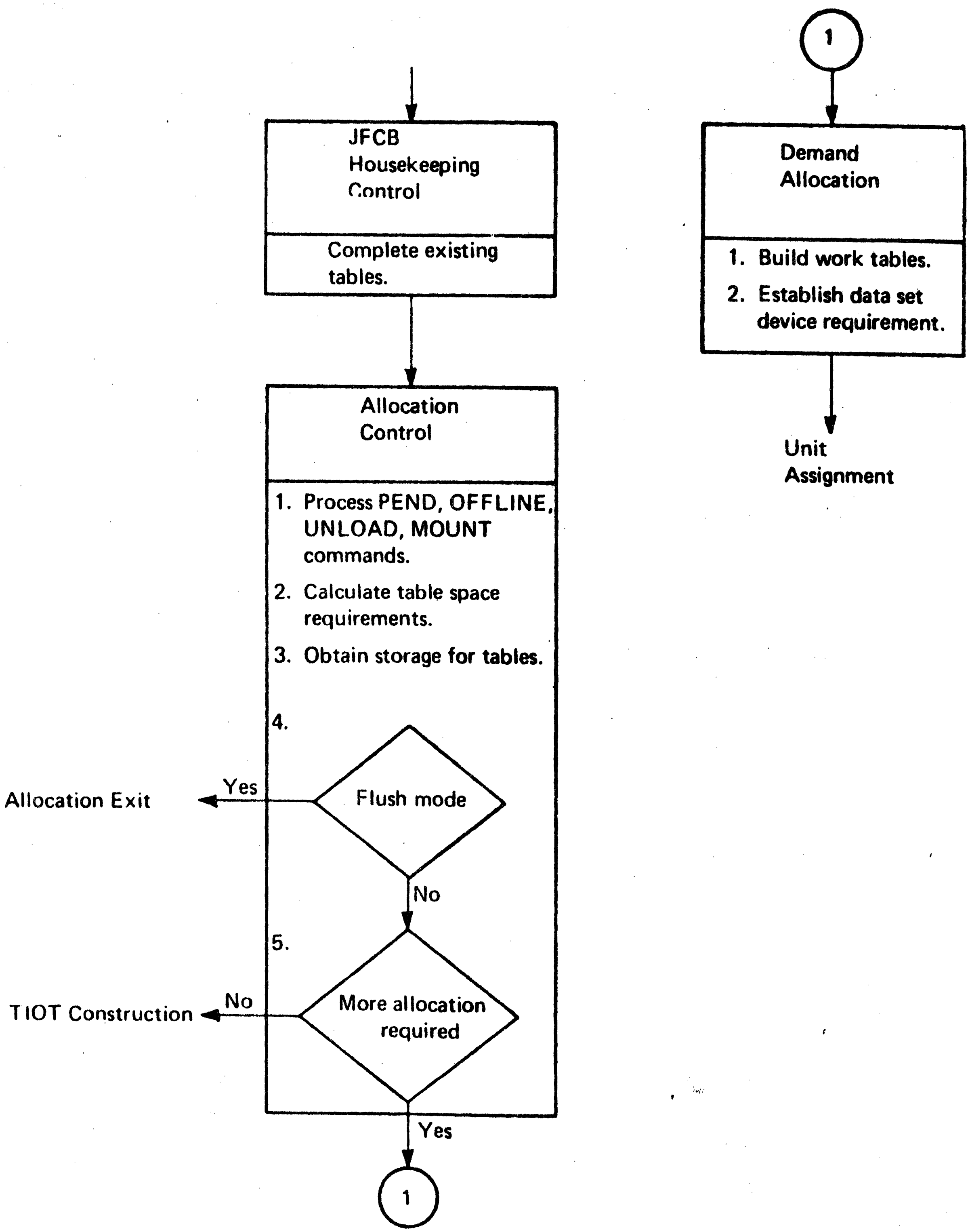
REQUEST CHARACTERISTICS

1. SPECIFIC REQUESTS
   - VOL=SER=XXXXXX
   - VOL=REF
   - RETRIEVAL THROUGH CATALOG
2. NON–SPECIFIC REQUESTS
   - MADE *ONLY* FOR *NEW* DATA SETS
   - TEMPORARY DATA SETS
   - NON–TEMP. DATA SETS

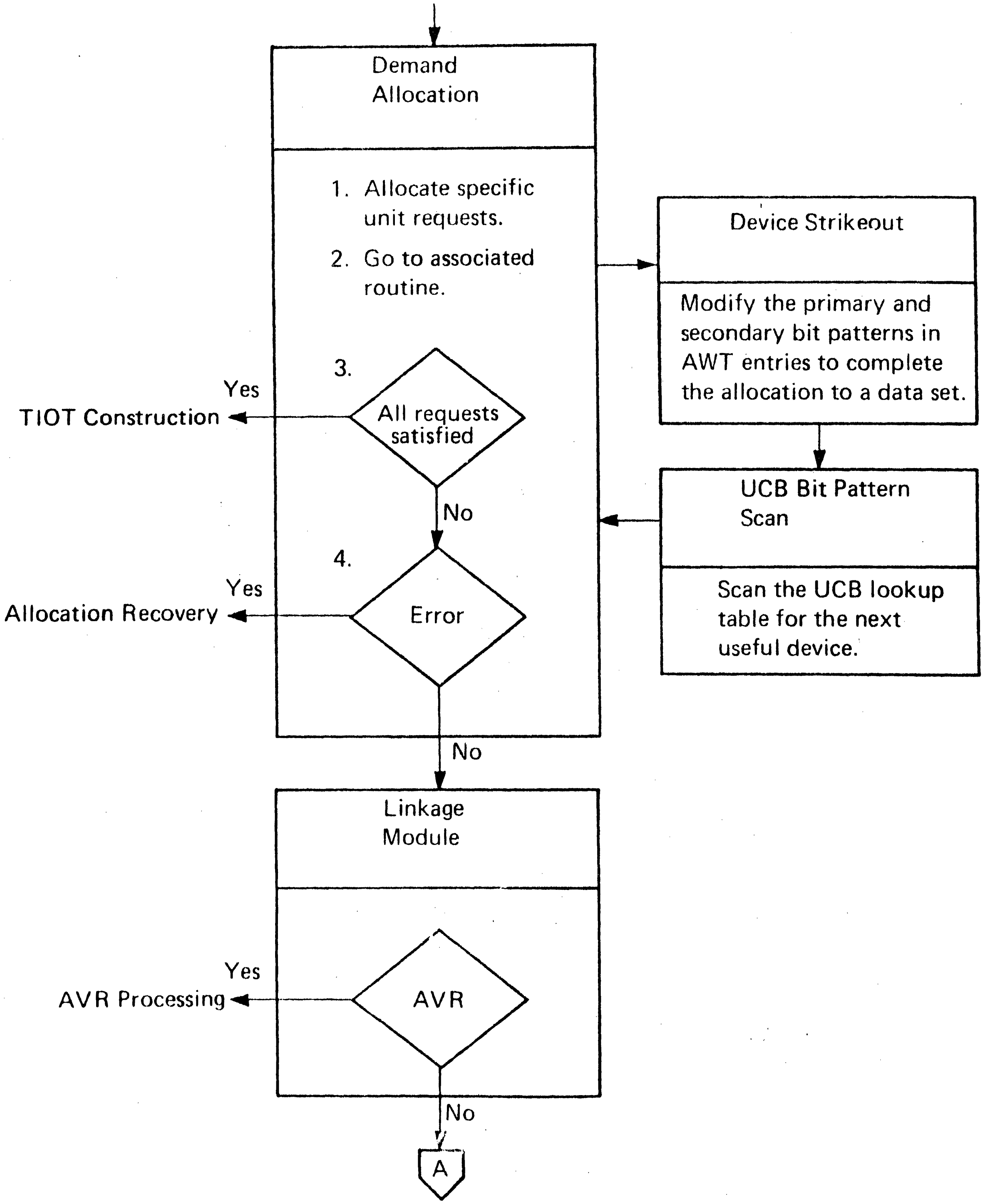
Allocation Entry


Entry
Allocation Entry Routine
1. Protect UCBs.
2. Check whether job was canceled.
Yes
No
Step Condition Code Processor
Determine from the EXEC Statement COND field whether the step should be bypassed.
Yes
Allocation Exit
No
Allocation Housekeeping

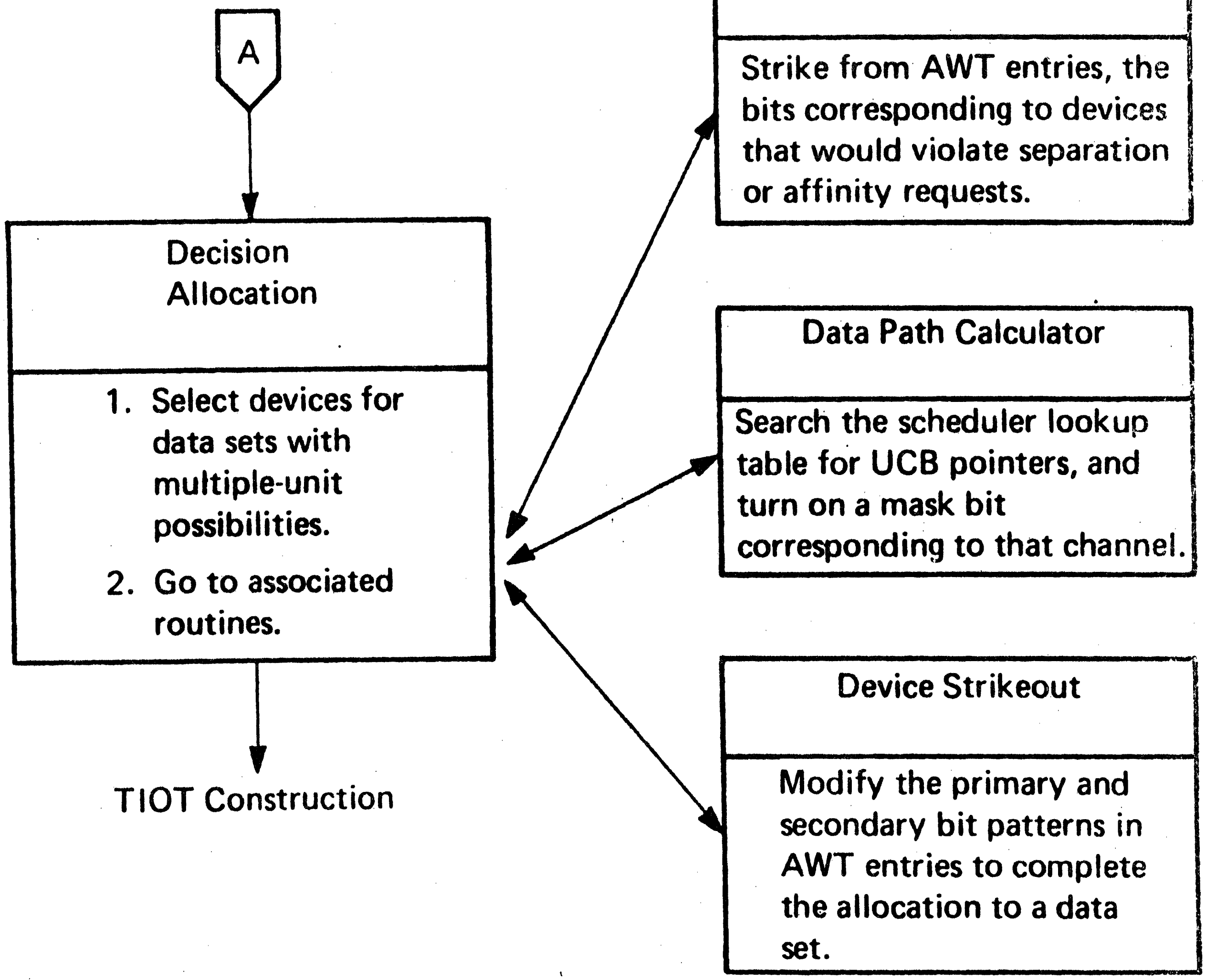
Allocation Housekeeping


JFCB Housekeeping Control
Complete existing tables.
Allocation Control
1. Process PEND, OFFLINE, UNLOAD, MOUNT commands.
2. Calculate table space requirements.
3. Obtain storage for tables.
4.
Flush mode
Yes
Allocation Exit
No
5.
More allocation required
No
TIOT Construction
Yes
1
1
Demand Allocation
1. Build work tables.
2. Establish data set device requirement.
Unit Assignment

Unit Assignment (Part 1 of 2)


Demand Allocation
1. Allocate specific unit requests.
2. Go to associated routine.
3.
All requests satisfied
Yes
TIOT Construction
No
4.
Error
Yes
Allocation Recovery
No
Device Strikeout
Modify the primary and secondary bit patterns in AWT entries to complete the allocation to a data set.
UCB Bit Pattern Scan
Scan the UCB lookup table for the next useful device.
Linkage Module
AVR
Yes
AVR Processing
No
A

Unit Assignment (Part 2 of 2)



Decision Allocation

1. Select devices for data sets with multiple-unit possibilities.
2. Go to associated routines.

TIOT Construction

Separation Strikeout

Strike from AWT entries, the bits corresponding to devices that would violate separation or affinity requests.

Data Path Calculator

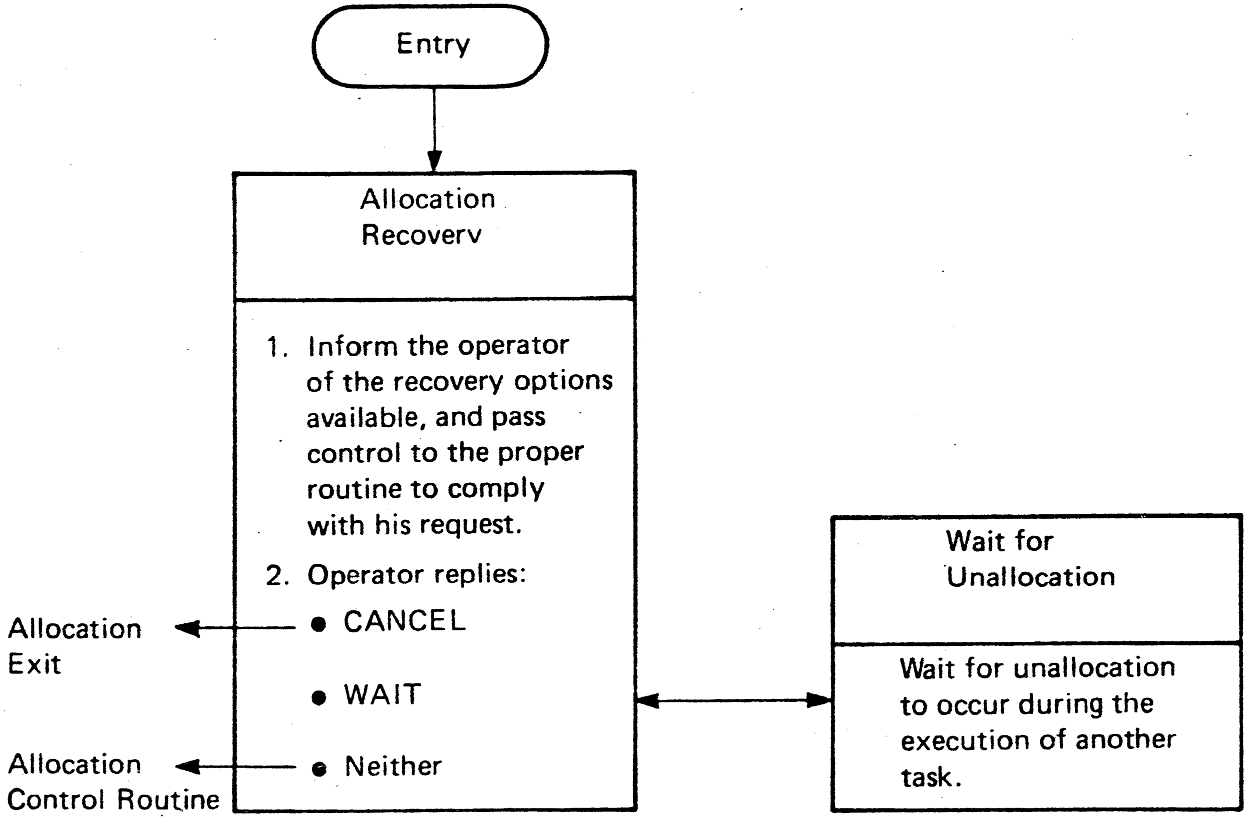
Search the scheduler lookup table for UCB pointers, and turn on a mask bit corresponding to that channel.

Device Strikeout

Modify the primary and secondary bit patterns in AWT entries to complete the allocation to a data set.

33

Allocation Recovery



Entry

Allocation Recoverv

1. Inform the operator of the recovery options available, and pass control to the proper routine to comply with his request.
2. Operator replies:
   - CANCEL
   - WAIT
   - Neither

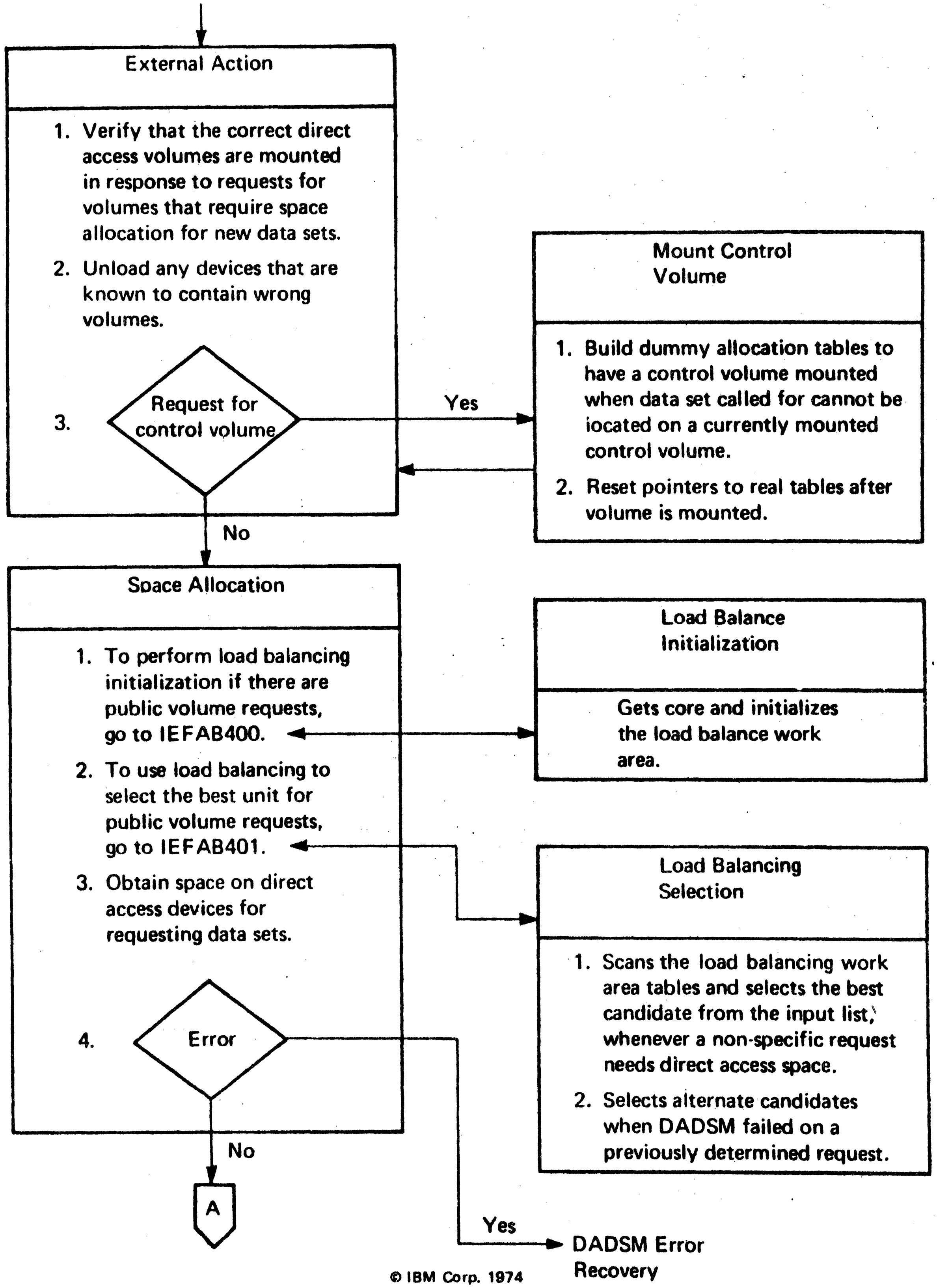
Allocation Exit

Allocation Control Routine

Wait for Unallocation

Wait for unallocation to occur during the execution of another task.

Space Assignment (Part 1 of 2)



External Action

1. Verify that the correct direct access volumes are mounted in response to requests for volumes that require space allocation for new data sets.
2. Unload any devices that are known to contain wrong volumes.
3. Request for control volume

Yes

Mount Control Volume

1. Build dummy allocation tables to have a control volume mounted when data set called for cannot be located on a currently mounted control volume.
2. Reset pointers to real tables after volume is mounted.

No

Space Allocation

1. To perform load balancing initialization if there are public volume requests, go to IEFAB400.
2. To use load balancing to select the best unit for public volume requests, go to IEFAB401.
3. Obtain space on direct access devices for requesting data sets.
4. Error

Load Balance Initialization

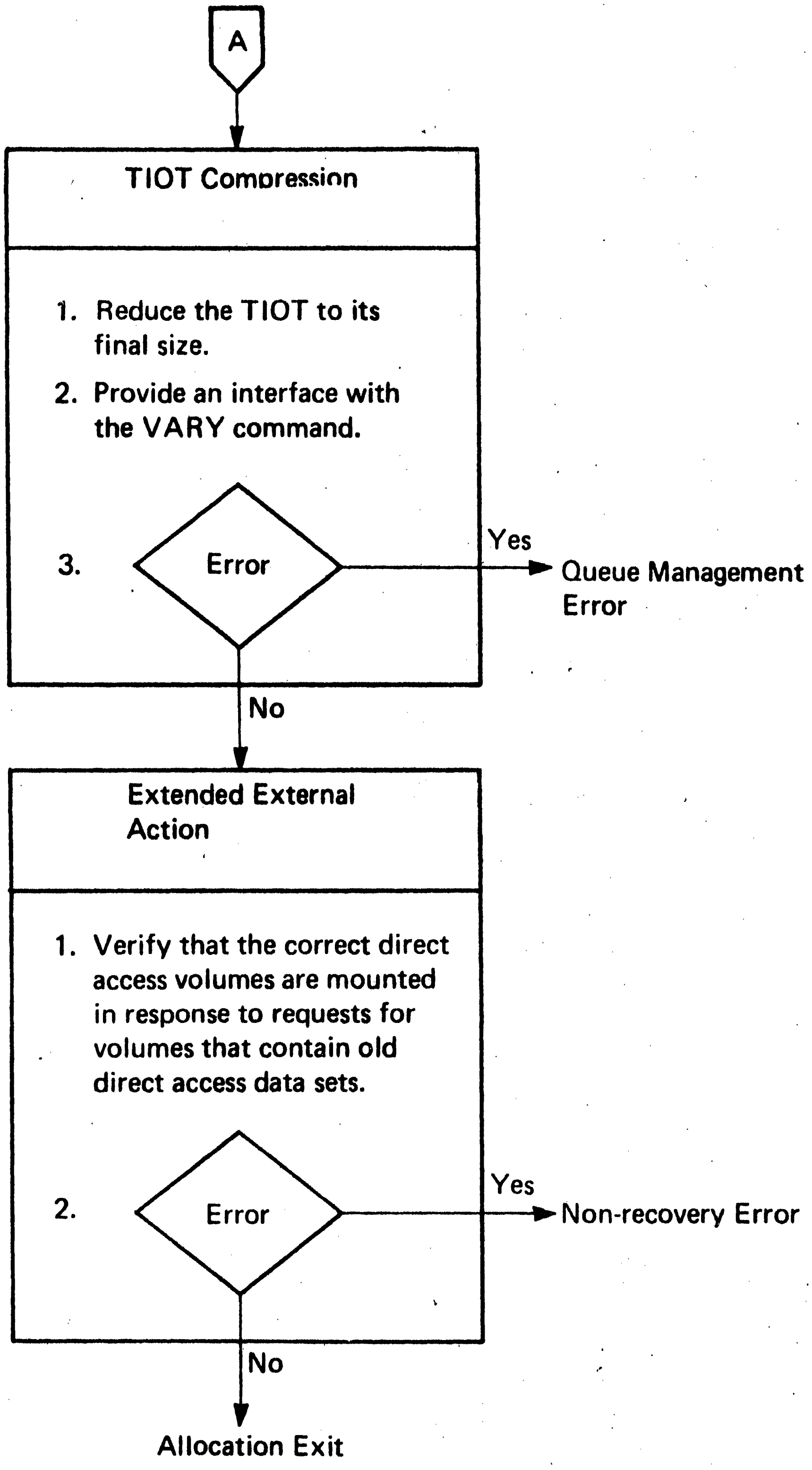
Gets core and initializes the load balance work area.

Load Balancing Selection

1. Scans the load balancing work area tables and selects the best candidate from the input list, whenever a non-specific request needs direct access space.
2. Selects alternate candidates when DADSM failed on a previously determined request.

DADSM Error Recovery

Space Assignment (Part 2 of 2)



TIOT Comoression

1. Reduce the TIOT to its final size.
2. Provide an interface with the VARY command.
3. Error

Extended External Action

1. Verify that the correct direct access volumes are mounted in response to requests for volumes that contain old direct access data sets.
2. Error

OS/VS1 Initiator Proc

```
XXIEFPROC    EXEC    PGM=IEFIIC, PARM='A, RESV=nn, SWA=n'
XXIEFRDER    DD      DSN=&&SWADS, UNIT=2314,                        X
                     SPACE=(176,(250,,CONTIG),)                     X
                     DISP=(NEW, DELETE)
XXDEDDD      DD      UNIT=2314, SPACE=(400,(200, 50),)              X
                     DISP=(NEW, DELETE)
```

JES Writer Proc

```
XXIEFPROC     EXEC     PGM=IEFOSC01, PARM='PA'
XXIEFRDER     DD       UNIT=1403, VOLUME=(,,,35),                  X
XX                     DSN=SYSOUT, DISP=(NEW, KEEP),               X
XX                     DCB=(RECFM=FM, LRECL=133)
```

JES READER PROCEDURE

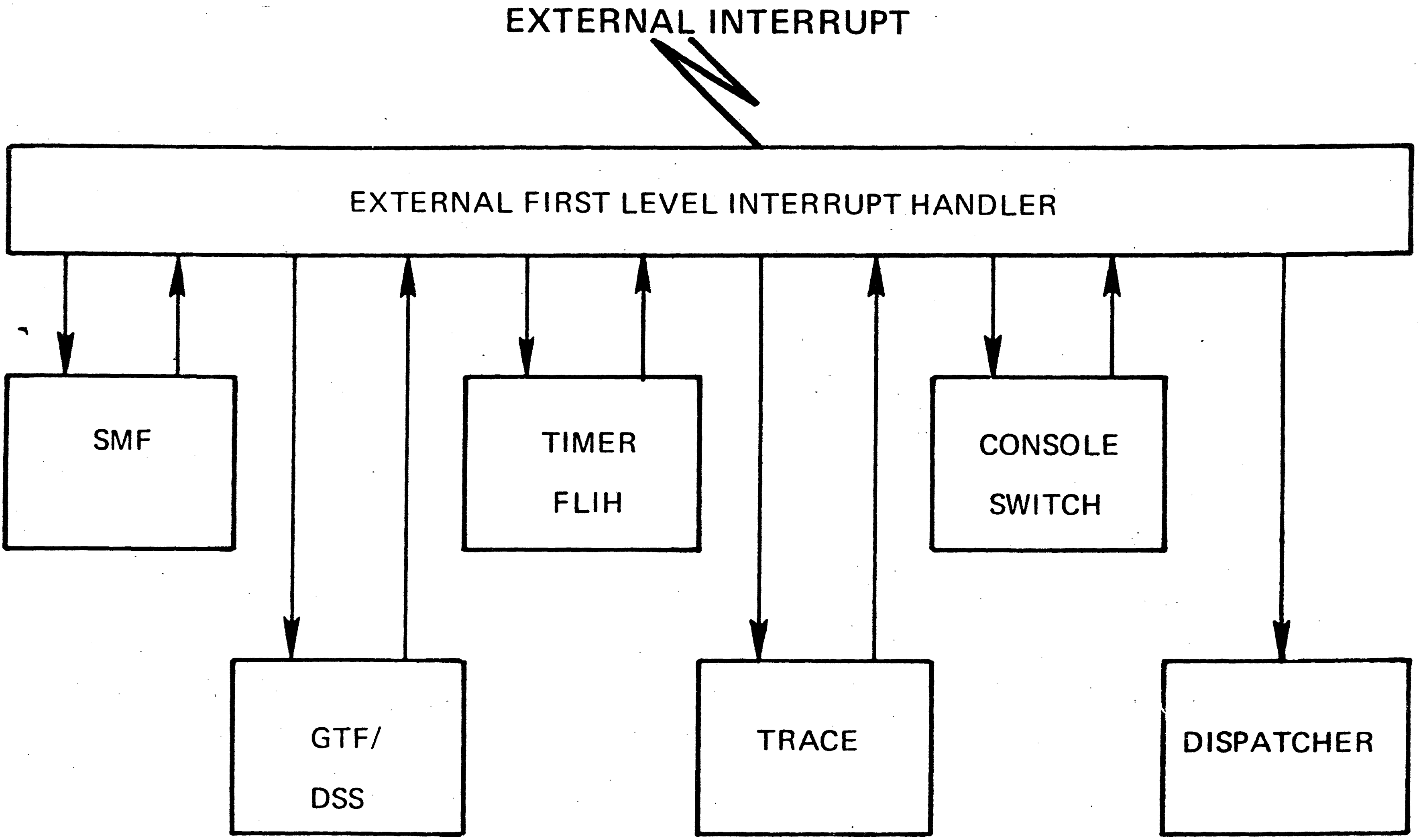
Procedure: RDR

```
// IEFPROC      EXEC      PGM=IEFVMA                                  X
//                        PARM='bppttttsscccrlaaaaefh'
// IEFRDER      DD        UNIT=2540,LABEL=(,NL),  VOLUME=SER=SYSIN,   X
//                        DCB=(LRECL=80,  RECFM=F),  DISP=OLD
```

Procedure: RDRT

```
// IEFPROC      EXEC      PGM=IEFVMA,                                 X
//                        PARM='bppttttssccclaaaaefh'
// IEFRDER      DD        UNIT=2400,LABEL=(,NL),  VOLUME=SER=SYSIN,   X
//                        DCB=(LRECL=80,  RECFM=F,  BLKSIZE=80),  DISP=OLD
```

TIMER PROCESSING



EXTERNAL INTERRUPT

EXTERNAL FIRST LEVEL INTERRUPT HANDLER

SMF

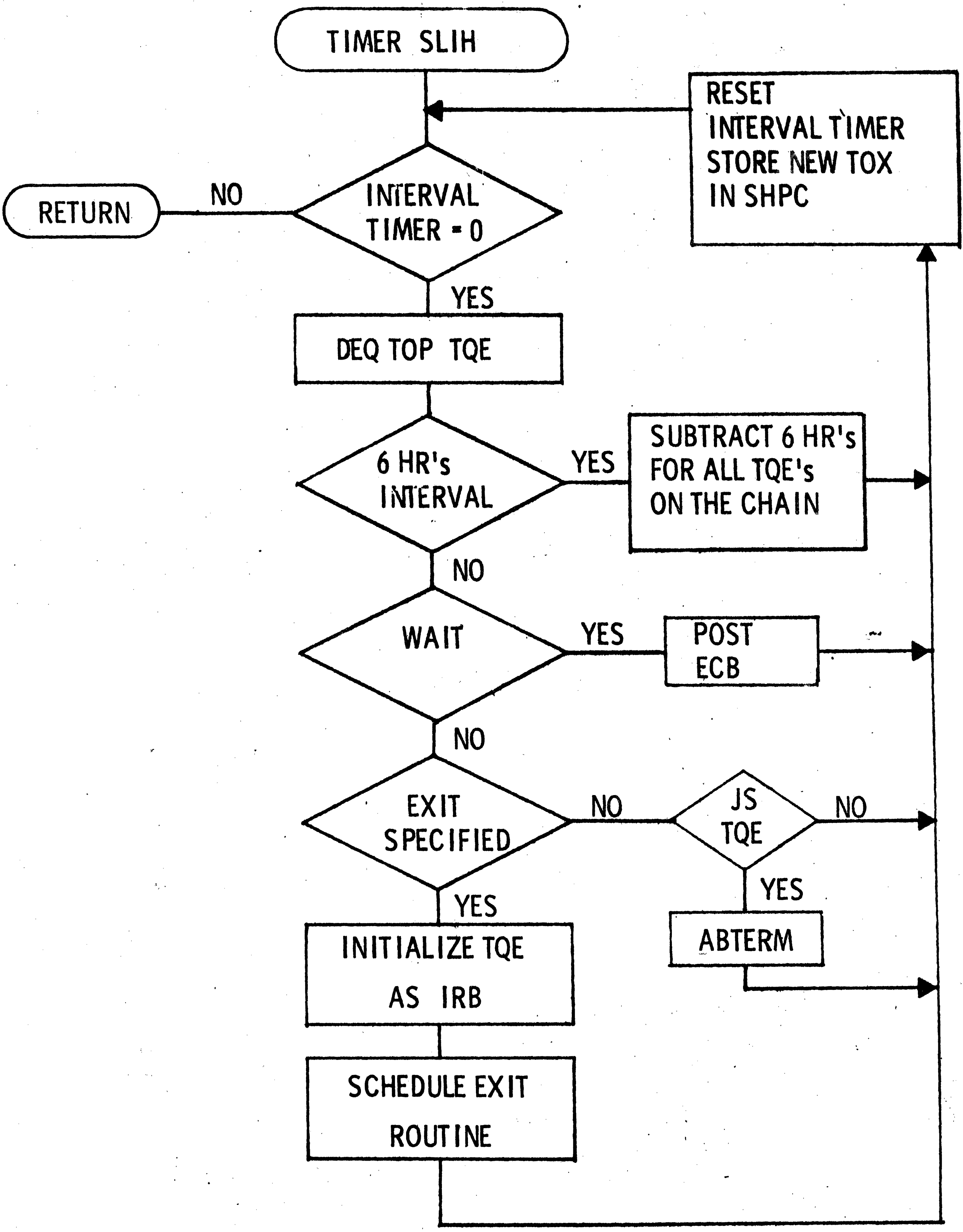
GTF/ DSS

TIMER FLIH

TRACE

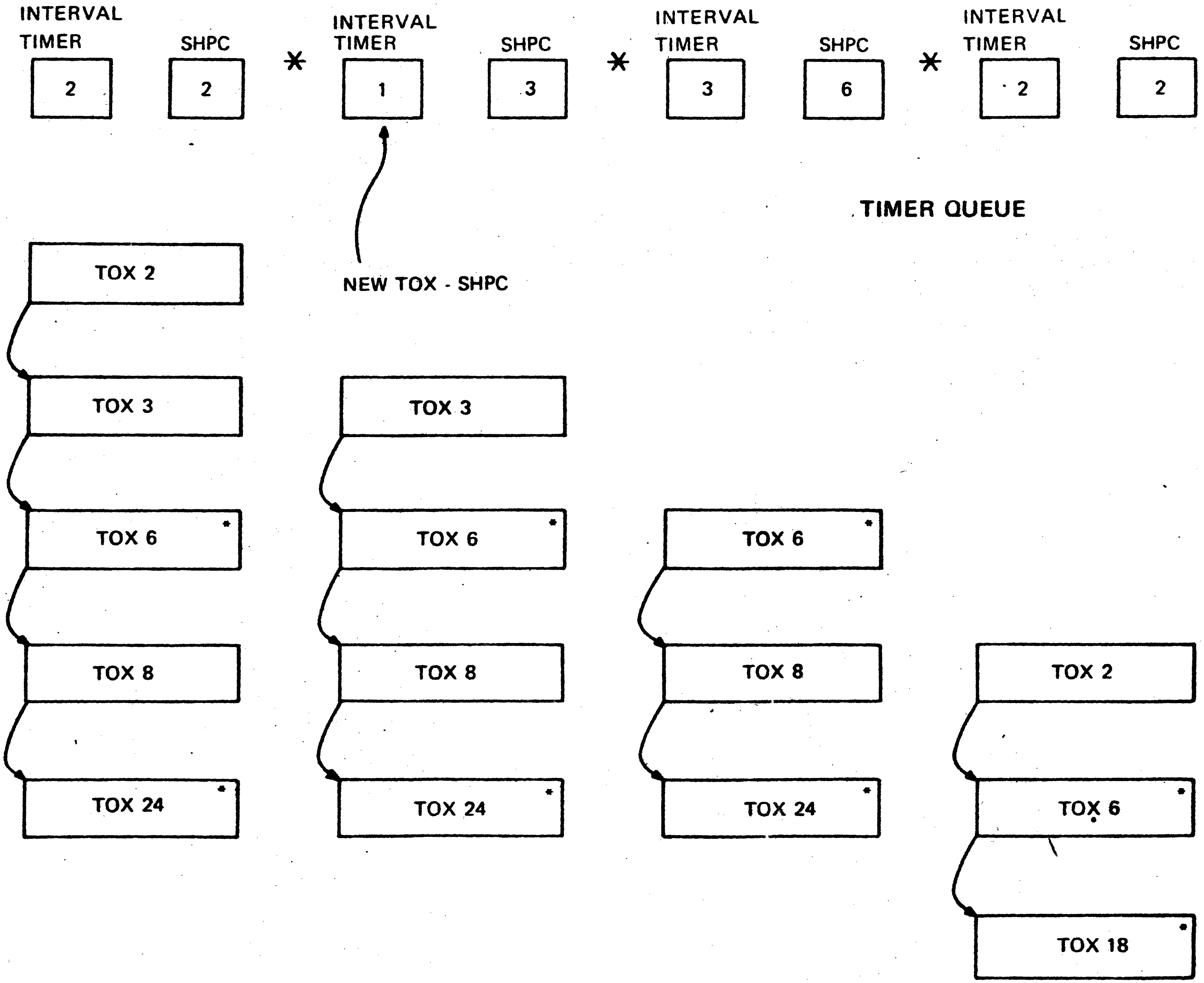
CONSOLE SWITCH

DISPATCHER


TIMER SLIH
RESET INTERVAL TIMER STORE NEW TOX IN SHPC
RETURN
NO
INTERVAL TIMER = 0
YES
DEQ TOP TQE
6 HR's INTERVAL
YES
SUBTRACT 6 HR's FOR ALL TQE's ON THE CHAIN
NO
WAIT
YES
POST ECB
NO
EXIT SPECIFIED
NO
JS TQE
NO
YES
ABTERM
YES
INITIALIZE TQE AS IRB
SCHEDULE EXIT ROUTINE

TIMER QUEUE ELEMENT

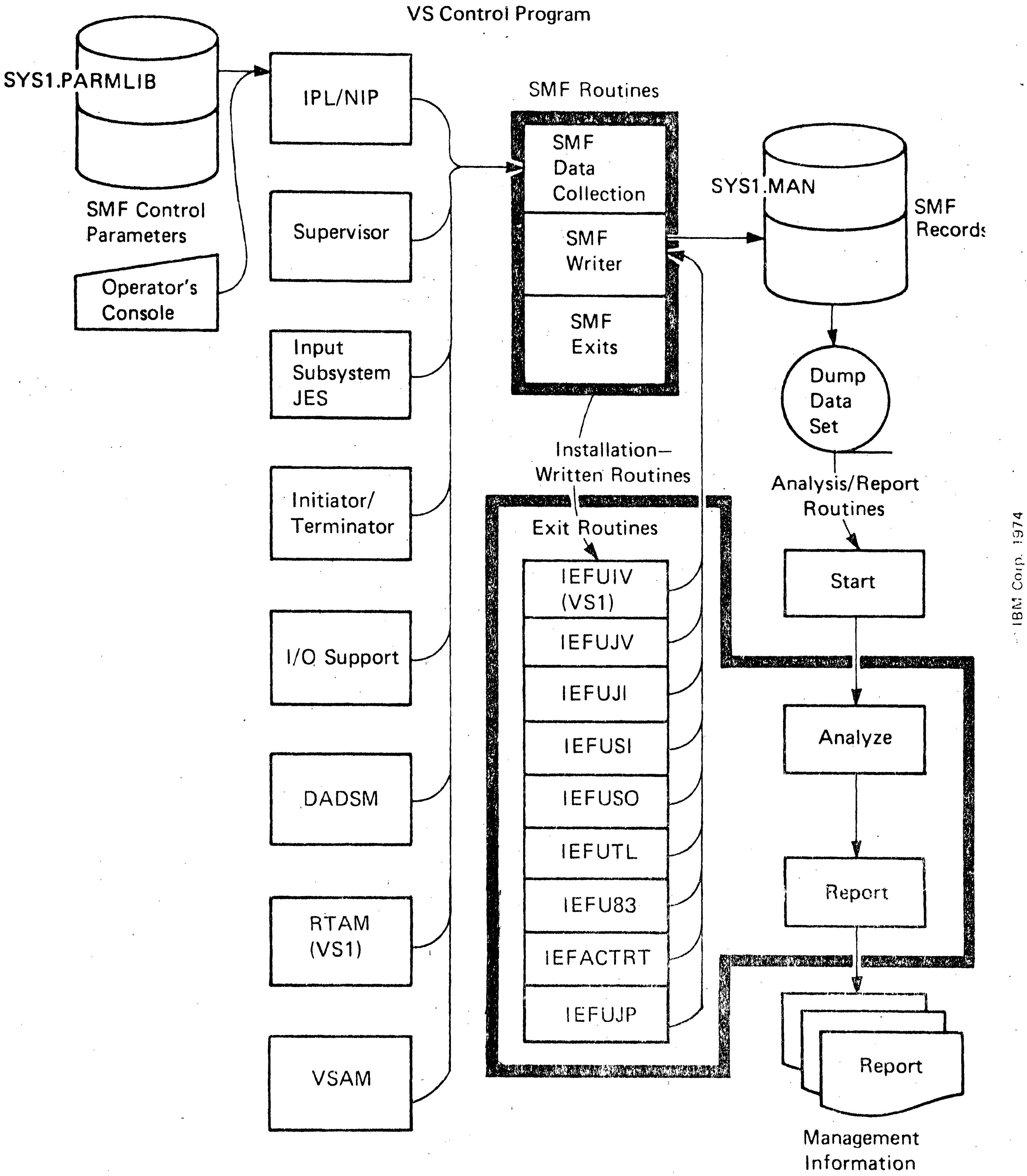
0	ADDRESS OF TCB	LINK TO NEXT TQE
8	LINK TO PREVIOUS TQE	TIME OF EXPIRATION/TIME REMAINING
10	LEFT HALF OF PSW	
18		ADDRESS OF EXIT ROUTINE
20	EXPECTED TIME OF EXPECTED RETURN	
28		
70		ECB WHEN WAIT


INTERVAL TIMER
2
SHPC
2
*
INTERVAL TIMER
1
SHPC
3
*
INTERVAL TIMER
3
SHPC
6
*
INTERVAL TIMER
2
SHPC
2
NEW TOX - SHPC
TOX 2
TOX 3
TOX 6
TOX 8
TOX 24
TOX 3
TOX 6
TOX 8
TOX 24
TOX 6
TOX 8
TOX 24
TOX 2
TOX 6
TOX 18

TIMER QUEUE

SMF

SMF in the VS1 System



© IBM Corp. 1974

© IBM Corp. 1974

ACCOUNTING RECORDS

RECORD TYPE	WHEN WRITTEN	INFORMATION CONTAINED
4	AFTER NORMAL OR ABNORMAL TERMINATION OF A JOB STEP FOR BACKGROUND JOBS.	JOB IDENTIFICATION, TIME OF DAY THAT CERTAIN EVENTS OCCUR DURING STEP PROCESSING, STEP CPU TIME, AMOUNT OF MAIN STORAGE ALLOCATED AND USED, DEVICES USED, STEP PAGING ACTIVITY, COMPLETION CODE, STEP PRIORITY, STEP ACCOUNTING DATA, TERMINATION INDICATOR.
5	AFTER NORMAL OR ABNORMAL JOB TERMINATION FOR BACKGROUND JOBS.	JOB IDENTIFICATION, TIME OF DAY THAT CERTAIN EVENTS OCCUR DURING JOB PROCESSING, JOB CPU TIME, COMPLETION CODE, JOB PRIORITY, JOB ACCOUNTING DATA, TERMINATION INDICATOR.
6	AFTER PROCESSING OF A SYSOUT CLASS OR FORM WITHIN A CLASS FOR A BACKGROUND JOB HAS FINISHED.	WRITER START AND END TIMES, NUMBER OF SYSOUT DATA SETS WITHIN THE CLASS AND FORM, NUMBER OF LOGICAL RECORDS PROCESSED.
20	EACH TIME A JOB IS INITIATED.	JOB IDENTIFICATION, PROGRAMMER'S NAME, USER IDENTIFICATION, NUMBER AND CONTENTS OF ACCOUNTING FIELDS ON JOB STATEMENT.

DATA SET ACTIVITY RECORDS

RECORD TYPE	WHEN WRITTEN	INFORMATION CONTAINED
14	A USER'S DATA SET OPENED FOR INPUT OR RDBACK IS CLOSED OR PROCESSED BY EOV.	CREATION AND EXPIRATION DATES, DEVICE TYPE, EXCP COUNT, VOLUME SERIAL NUMBERS, NUMBER OF VOLUMES, RECORD FORMAT AND LENGTH, AND PERTINENT PORTIONS OF SYSTEM CONTROL BLOCKS.
15	A USER'S DATA SET OPENED FOR OUTPUT, UPDAT, INOUT, OR OUTIN PROCESSING IS CLOSED OR PROCESSED BY EOV.	SAME TYPE AS RECORD 14.
17	A USER'S DATA SET IS SCRATCHED.	DATA SET NAME, NUMBER OF VOLUMES, VOLUME SERIAL NUMBERS.
18	A DATA SET IS RENAMED.	OLD DATA SET NAME, NEW DATA SET NAME, NUMBER OF VOLUMES, VOLUME SERIAL NUMBERS.
62	AT THE SUCCESSFUL OR UNSUCCESSFUL OPENING OF A VSAM COMPONENT.	THE NAME OF THE CATALOG IN WHICH THE COMPONENT OR CLUSTER IS DEFINED AND THE VOLUMES ON WHICH THE CATALOG AND THE COMPONENT OR CLUSTER ARE STORED.
64	WHEN A VSAM COMPONENT OR CLUSTER IS CLOSED, WHEN IT BECOMES NECESSARY TO SWITCH TO ANOTHER VOLUME TO CONTINUE PROCESSING, OR WHEN NO MORE SPACE IS AVAILABLE ON A VOLUME. ONE RECORD IS WRITTEN FOR EACH COMPONENT CLOSED. IF A CLUSTER IS CLOSED, ONE RECORD IS WRITTEN FOR EACH COMPONENT IN THE CLUSTER.	THE CONDITION THAT CAUSED THE RECORD TO BE WRITTEN, IDENTIFIES THE VOLUME ON WHICH THE COMPONENT IS STORED, EXTENTS OF THE COMPONENT ON THE VOLUME, AND STATISTICS ABOUT PROCESSING EVENTS THAT HAVE OCCURRED SINCE THE COMPONENT WAS OPENED.
68	WHEN A VSAM CLUSTER OR COMPONENT IS RENAMED.	THE NAME OF THE VSAM CATALOG IN WHICH THE COMPONENT IS DEFINED AND THE OLD AND NEW NAMES.

SYSTEM USE RECORDS

Record Type	When Written	Information Contained
0	During system initialization after IPL.	Real and virtual storage size and SMF options in effect.
1	At SMF initialization and at the first job step termination following the expiration of a ten-minute interval of elapsed system time.	CPU wait time, system paging statistics accumulated during all of the ten-minute intervals that expired since the last type 1 record was written, the expiration time of the last ten-minute interval.
2	At the beginning of a dump data set.	System identification and the time and date the record was moved to the SMF buffer. (This record is the standard record header.)
3	At the end of a dump data set.	Same type as 2.
7	After any period when there was no SMF data set available for recording. This is the first record written when an SMF data set again becomes available.	Count of SMF records generated but not written and the start and end times of the period during which no records were written.
8	During system initialization after IPL.	Descriptions of each online device at IPL. (Each entry description includes the device class, unit type, and channel/unit address.)
9	During processing of the VARY ONLINE operator command.	Identification of the device added to the configuration.
10	After a device is added to the configuration.	Identification of the device made available by device class, unit type, and device address. Job requiring the allocation is identified.
11	During processing of the VARY OFFLINE operator command.	Identification of the device removed from the configuration.
12	During processing of HALT or SWITCH operator commands.	System wait time and paging statistics since the last record type 1 and the time this record was built.
13	At IPL and after each DEFINE command is processed under VS1 only.	The amount of storage assigned to each partition.
43R	During RTAM initialization under VS1 only.	RES information including name of RTAM start procedure, maximum numbers of readers and writers, number of entries in LINE table, number of line DCTs, number of lines to activate, line names, and unit addresses.
44R	Whenever a MODIFY RTAM command is issued under VS1 only.	RES information including name of start procedure, type of MODIFY, number of lines modified, line numbers, and unit addresses.
45R	When a STOP RTAM command is issued under VS1 only.	RTAM information including name of RTAM STOP procedure, stop status, and number of lines started when STOP was received.
47R	Whenever a valid LOGON record is received by RTAM under VS1 only.	RES information including QID entry, passback area, and LOGON record.
48R	Whenever a LOGOFF record is received by RTAM under VS1 only.	RES information consisting of the QID entry.
49R	Whenever an invalid LOGON record is received by RTAM under VS1 only.	RES information including the QID entry, passback area, and LOGON record.

I/O SUPERVISION

SYSTEMS PROGRAMMING

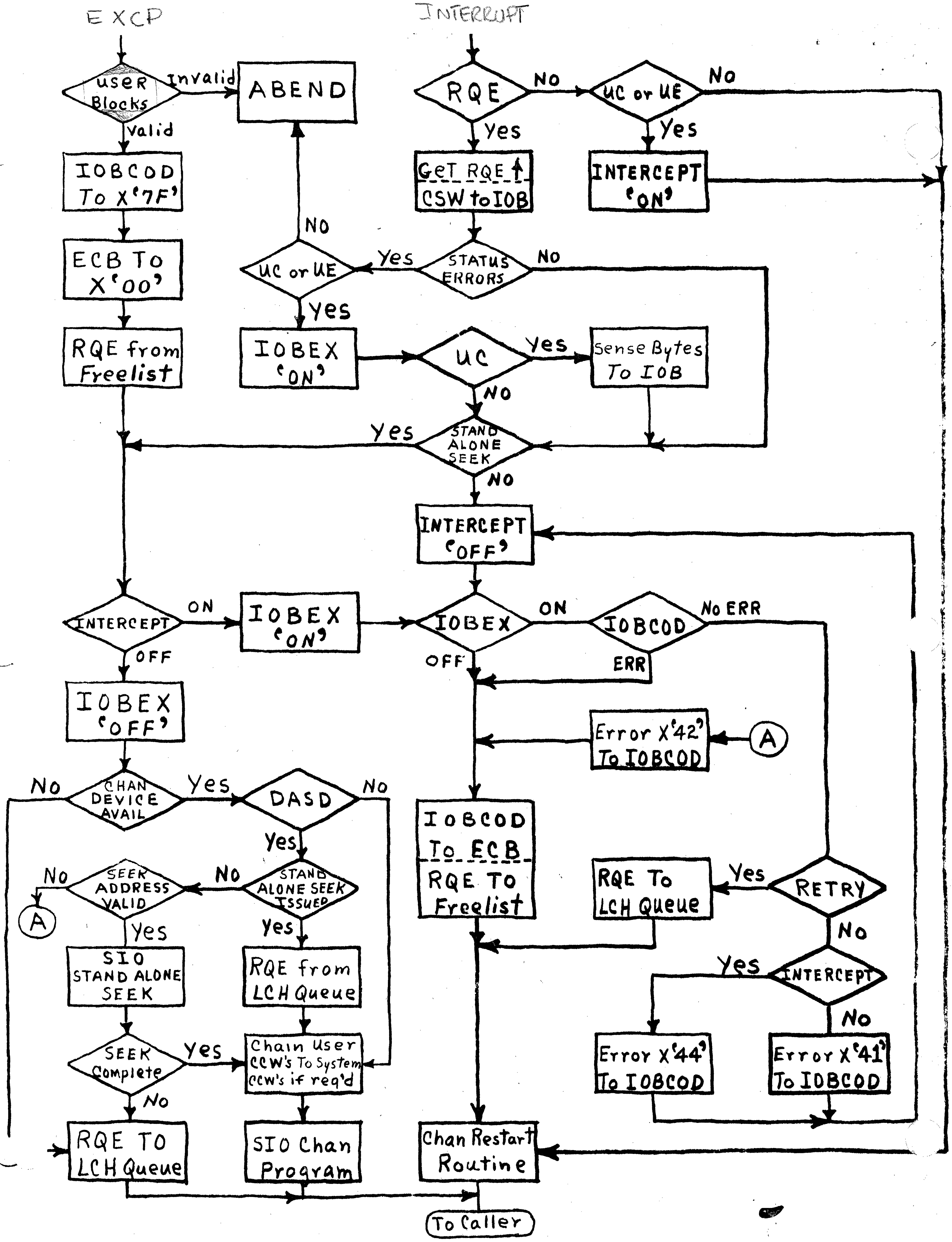
Student Materials

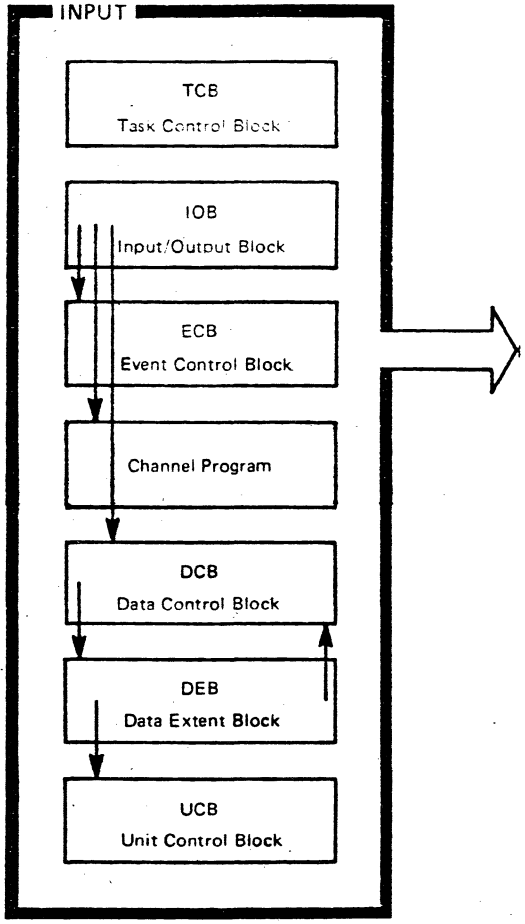
This material was produced for Educational purposes only. Changes may obsolete part or all of this publication. No responsibility is assumed for any inaccuracies that occur.

Los Angeles Advanced Education Center
3424 WILSHIRE BOULEVARD • LOS ANGELES, CALIF. 90010

INPUT / OUTPUT

SUPERVISOR


EXCP
INTERRUPT
user Blocks
Invalid
Valid
ABEND
RQE
NO
Yes
UC or UE
No
Yes
IOBCOD To X'7F'
GeT RQE ↑
CSW to IOB
INTERCEPT 'ON'
NO
ECB To X'00'
UC or UE
Yes
STATUS ERRORS
NO
Yes
RQE from Freelist
IOBEX 'ON'
UC
Yes
Sense Bytes To IOB
NO
Yes
STAND ALONE SEEK
NO
INTERCEPT 'OFF'
INTERCEPT
ON
IOBEX 'ON'
IOBEX
ON
IOBCOD
No ERR
OFF
OFF
ERR
IOBEX 'OFF'
Error X'42' To IOBCOD
A
No
CHAN DEVICE AVAIL
Yes
DASD
NO
IOBCOD To ECB
RQE To Freelist
Yes
No
SEEK ADDRESS VALID
NO
STAND ALONE SEEK ISSUED
RQE To LCH Queue
Yes
RETRY
A
Yes
yes
No
SIO STAND ALONE SEEK
RQE from LCH Queue
Yes
INTERCEPT
No
SEEK Complete
Yes
Chain User CCW's To System CCW's if req'd
Error X'44' To IOBCOD
Error X'41' To IOBCOD
No
RQE TO LCH Queue
SIO Chan Program
Chan Restart Routine
To Caller



Extended Description

1 If DEB validity check was specified at system generation, the DEB validity check module, IFGDEBVR, is entered for non key=0 I/O requests to ensure that the DEB passed to IOS is on the DEB chain. If not, the ABEND routine is entered.

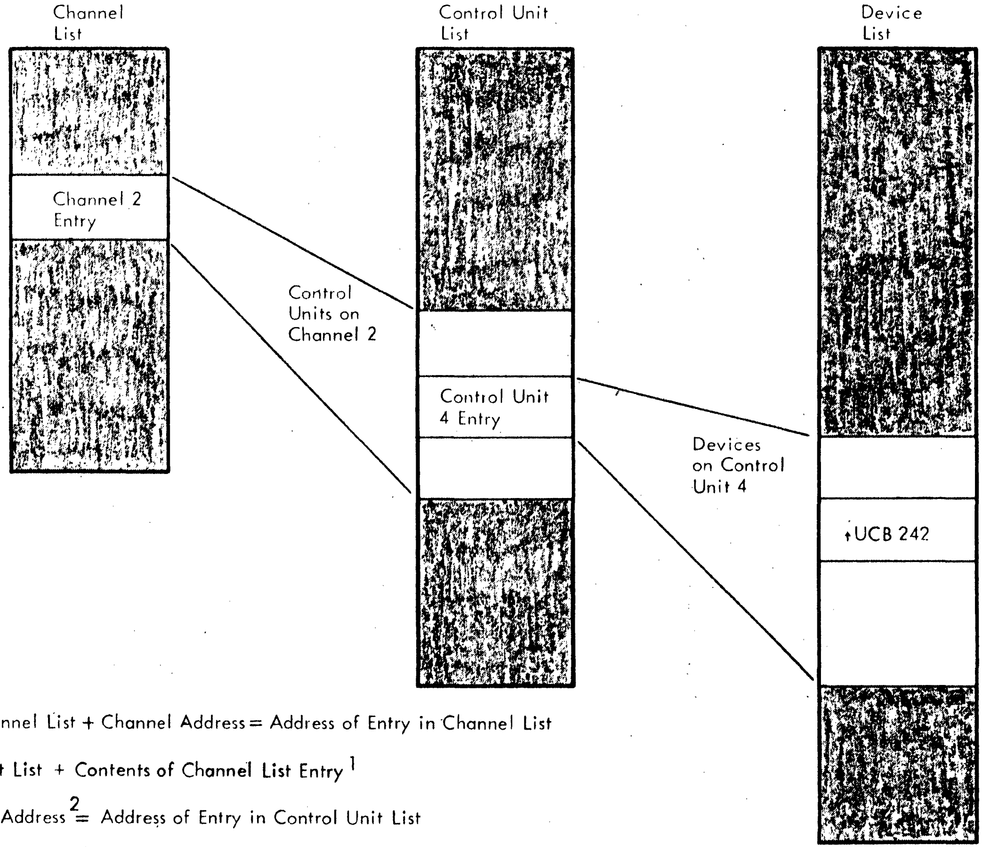
2 The IOB's pointer to the DCB must be the same as the DEB's pointer to the DCB. If not, the ABEND routine is entered.

3 If the system has the protection feature, a non-supervisor requester's key must equal the storage key in the DEB. The ABEND routine is entered if the keys do not match.

4 The UCB is invalid if the UCBID field is not all 1's.

5 If this routine is entered from the supervisor's SVC First Level Interruption Handler, the Get Request Element routine is entered to build an RQE for this request.

If this routine is entered from the Error EXCP routine, the address of the logical channel word is calculated and placed in a register. The Error EXCP routine is then reentered.



Beginning Address of Channel List + Channel Address = Address of Entry in Channel List

Beginning of Control Unit List + Contents of Channel List Entry [1]

+Control Unit Address [2] = Address of Entry in Control Unit List

Beginning Address of Device List + 2 (Contents of Control Unit List Entry)

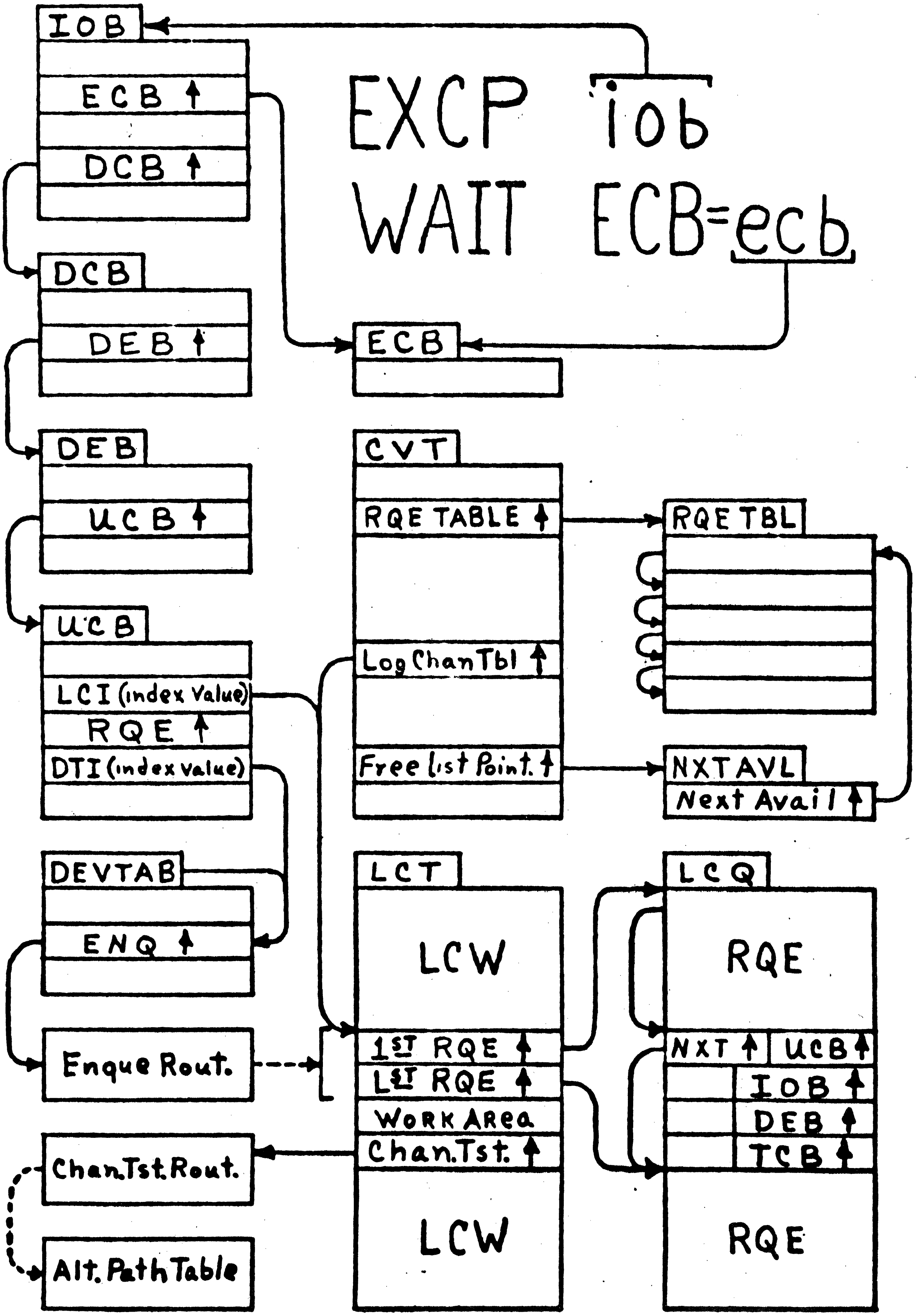
+ 2 (Device Number) = Address of Entry in Device List

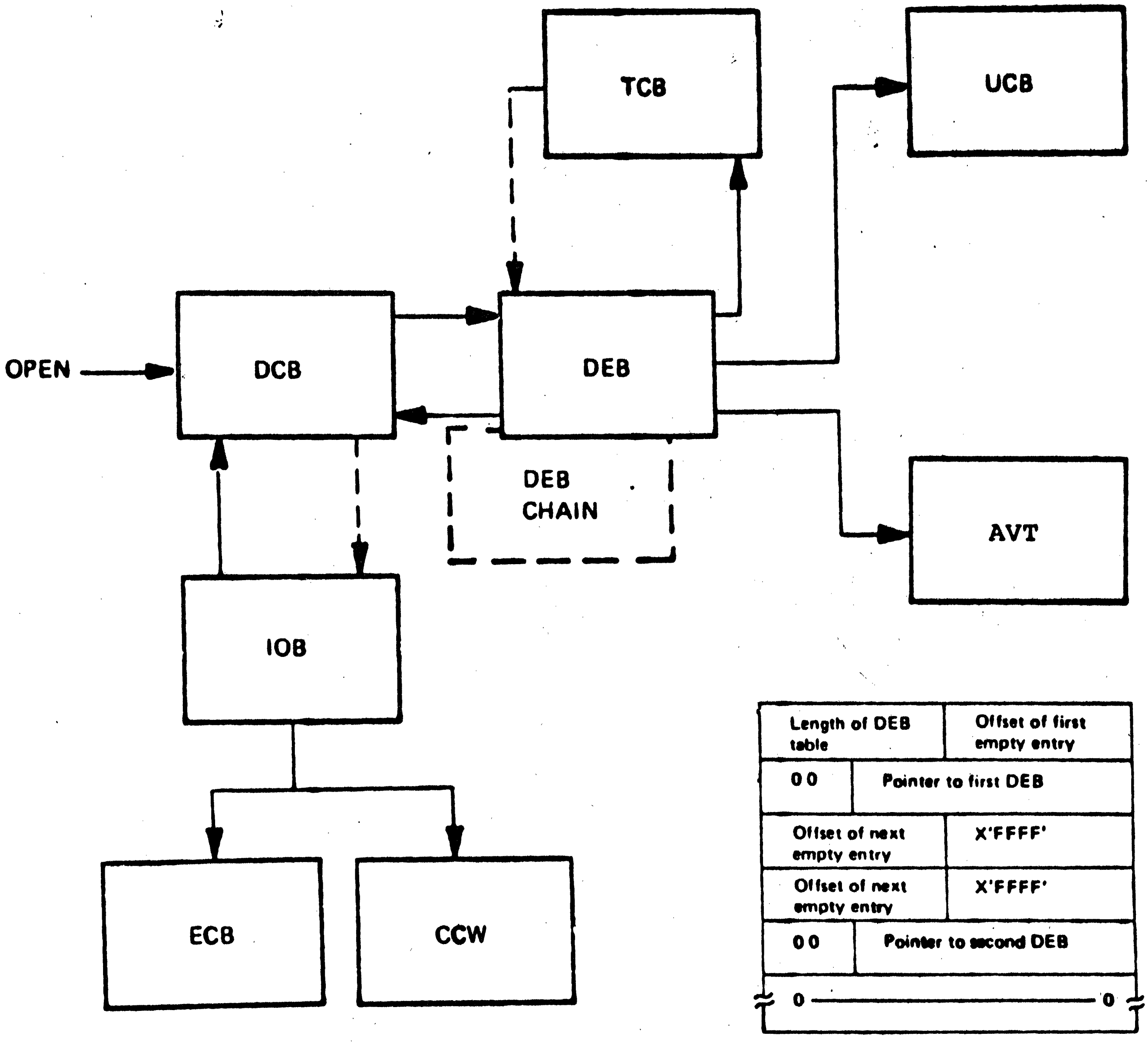
[1] If the channel number is 7 or greater, the contents of the channel list entry is doubled.

[2] If the control unit entry list contains 2-byte entries, the control unit address is multiplied by 2 before it is used.

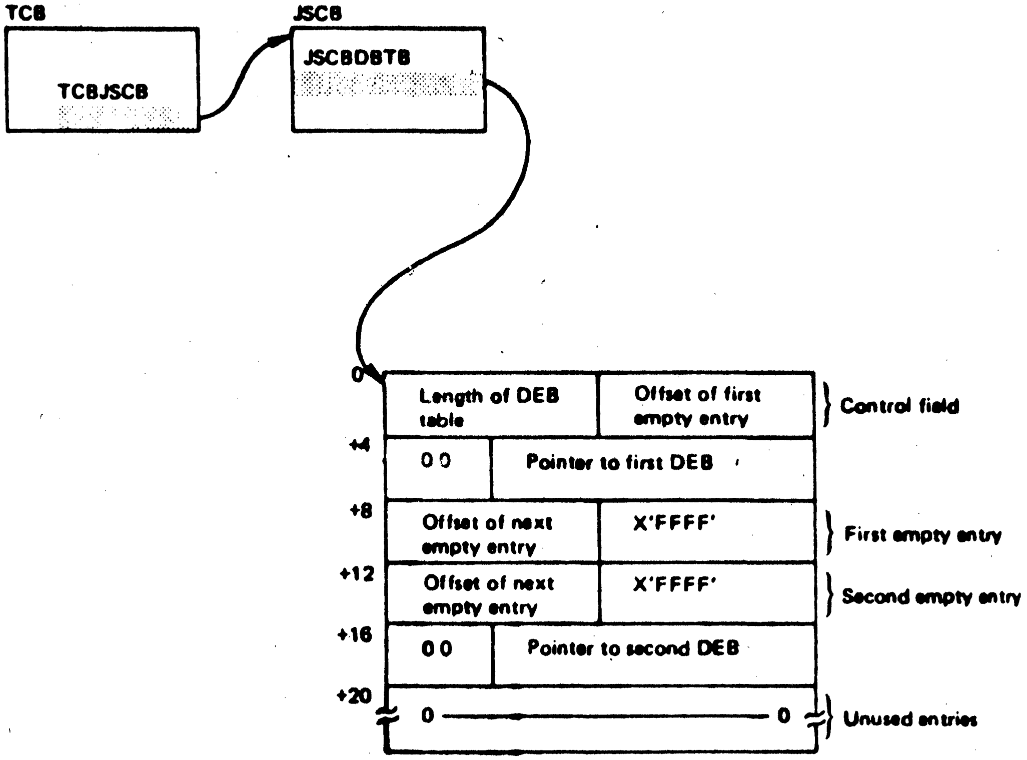
UCB Lookup Table Entry Relationships

EXCP SVC 0


EXCP iob
WAIT ECB=ecb
IOB
ECB ↑
DCB ↑
DCB
DEB ↑
ECB
DEB
UCB ↑
CVT
RQE TABLE ↑
RQE TBL
UCB
LCI (index value)
RQE ↑
DTI (index value)
Log Chan Tbl ↑
Free list Point. ↑
NXTAVL
Next Avail ↑
DEVTAB
ENQ ↑
LCT
LCW
LCQ
RQE
Enque Rout.
1ST RQE ↑
LST RQE ↑
Work Area
Chan.Tst. ↑
NXT ↑
UCB ↑
IOB ↑
DEB ↑
TCB ↑
Chan.Tst.Rout.
Alt. Path Table
LCW
RQE



Length of DEB table	Offset of first empty entry
0 0	Pointer to first DEB
Offset of next empty entry	X'FFFF'
Offset of next empty entry	X'FFFF'
0 0	Pointer to second DEB
0	0

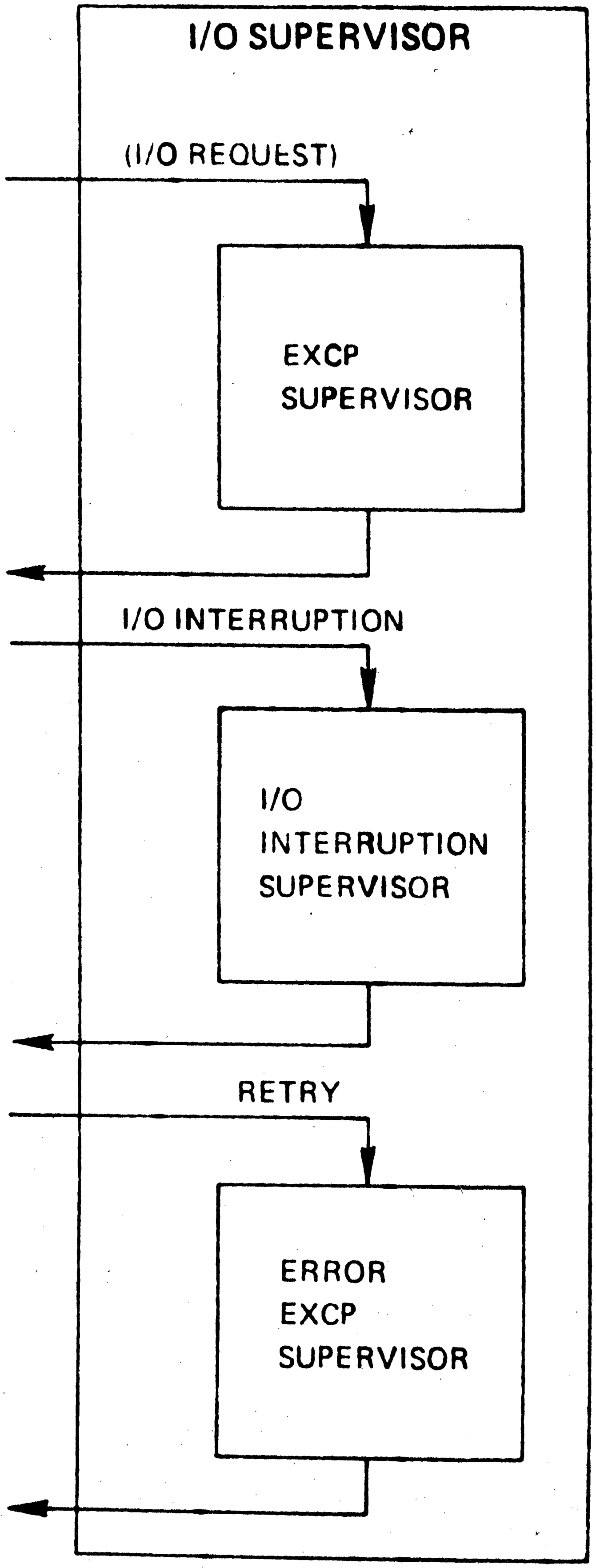

TCB
TCBJSCB
JSCB
JSCBDBTB
0
Length of DEB table
Offset of first empty entry
Control field
+4
0 0
Pointer to first DEB
+8
Offset of next empty entry
X'FFFF'
First empty entry
+12
Offset of next empty entry
X'FFFF'
Second empty entry
+16
0 0
Pointer to second DEB
+20
0
0
Unused entries

IOS VIRTUAL FUNCTIONS

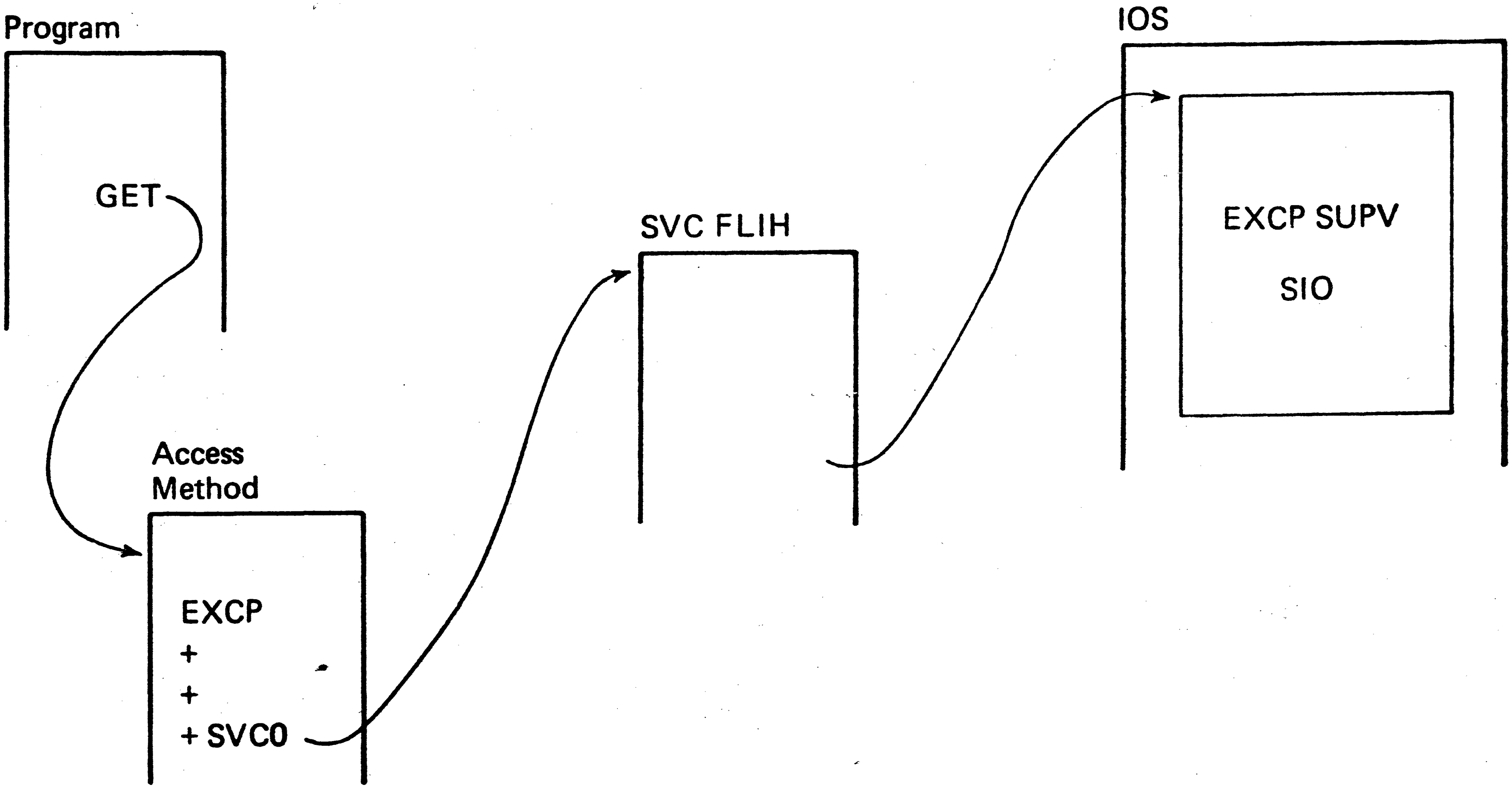
- TRANSLATE CCW DATA ADDRESSES FROM VIRTUAL TO REAL
- PROVIDE FOR I/O AREA CROSSING NON-CONTIGUOUS PAGES
- ASSURE PAGES REQUIRED TO PERFORM I/O OPERATION ARE FIXED

SHORT. TERM FIXED.

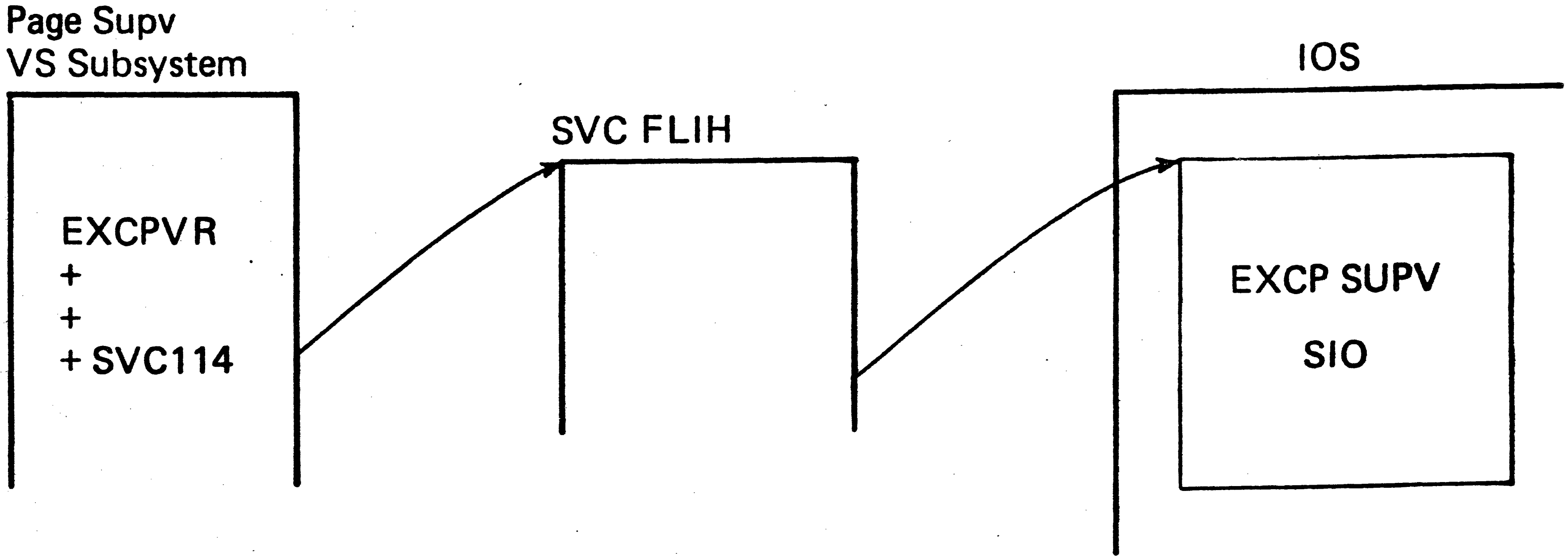
→ las pag. q' se fijan sólo para la operación de I/O


I/O SUPERVISOR
(I/O REQUEST)
EXCP
SUPERVISOR
I/O INTERRUPTION
I/O
INTERRUPTION
SUPERVISOR
RETRY
ERROR
EXCP
SUPERVISOR

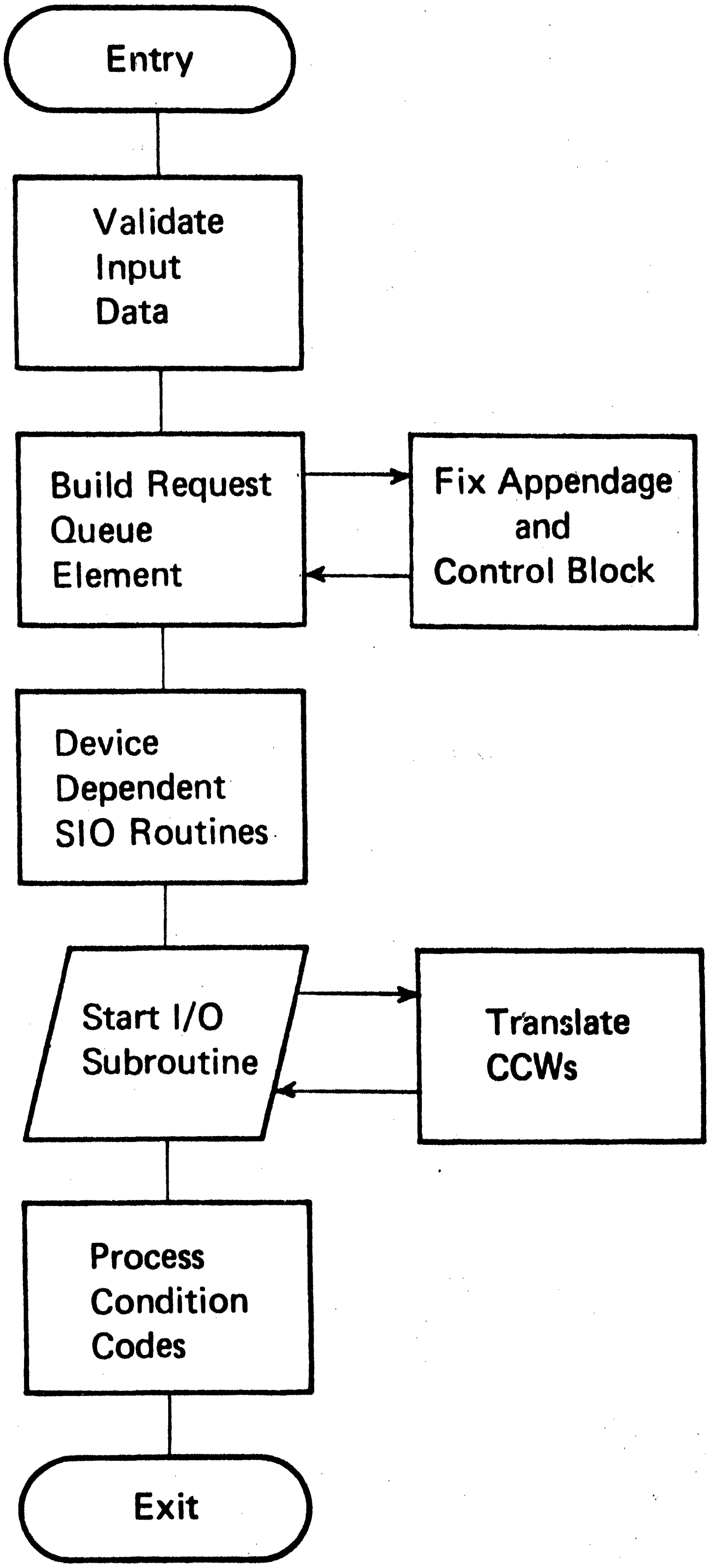
Starting I/O (V = V and V = R)



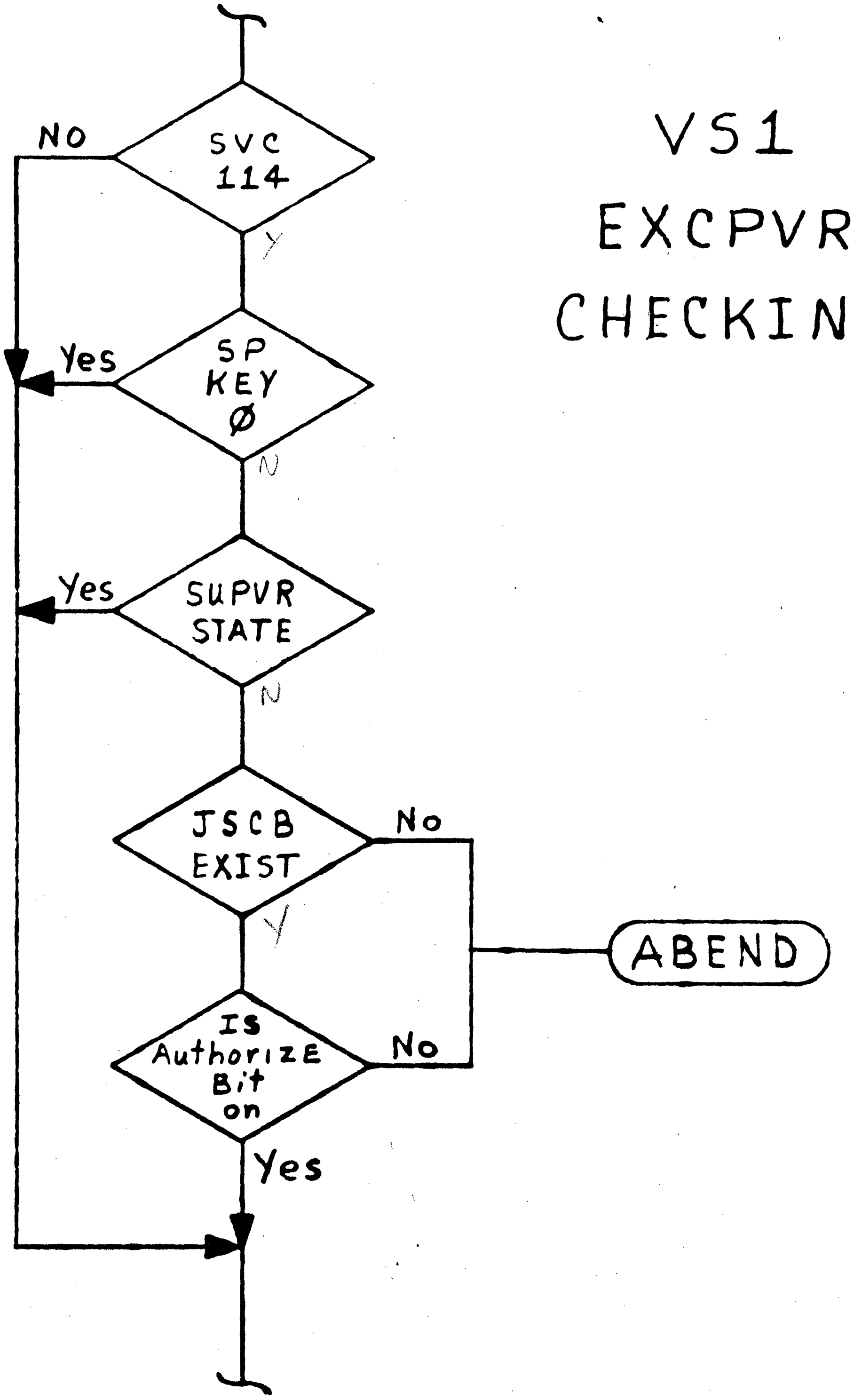
Starting I/O (Page Supervisor and VS Subsystems)

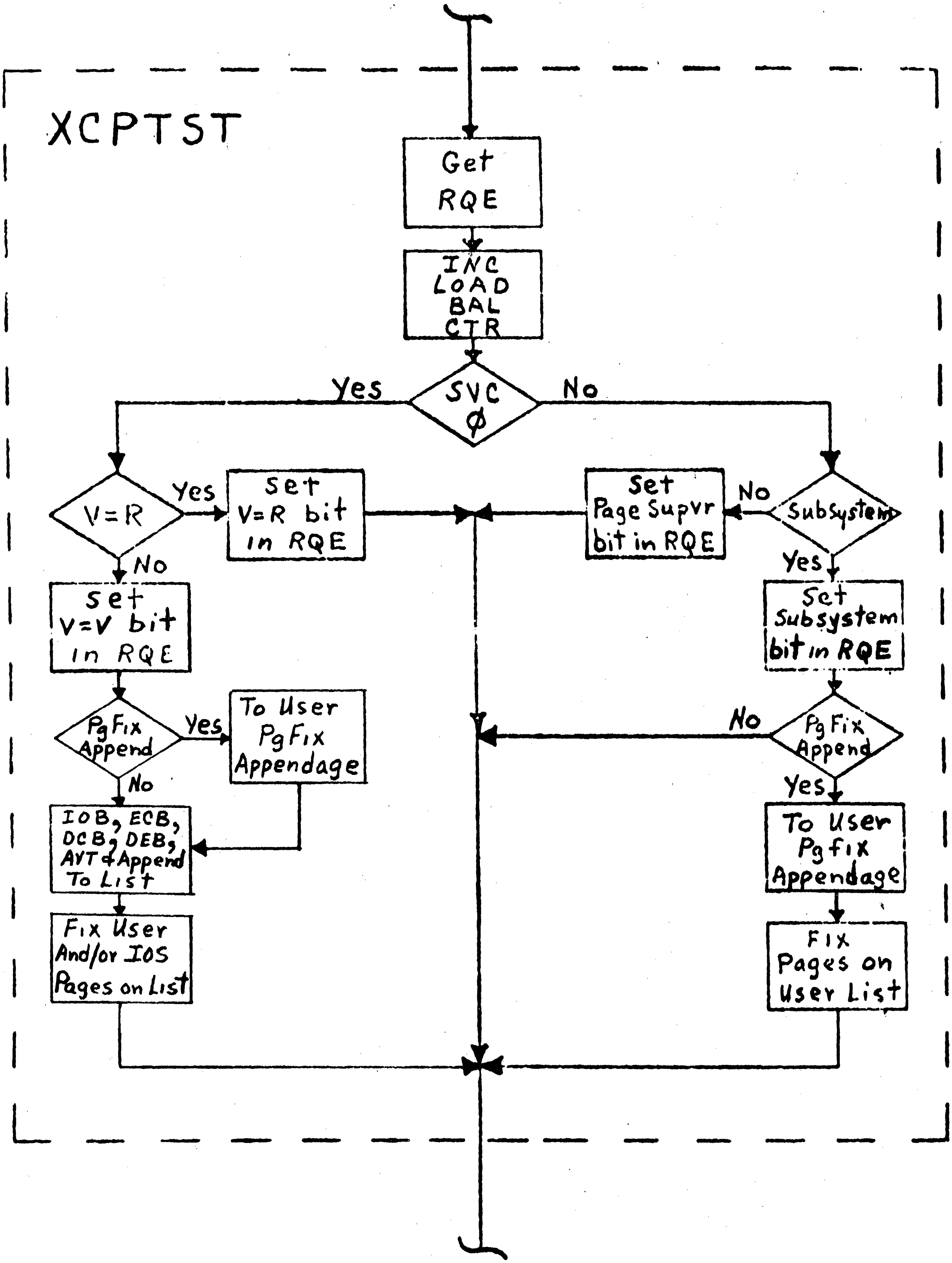


EXCP Supervisor


Entry
Validate Input Data
Build Request Queue Element
Fix Appendage and Control Block
Device Dependent SIO Routines
Start I/O Subroutine
Translate CCWs
Process Condition Codes
Exit

VS1 EXCPVR CHECKING




XCPTST
Get RQE
INC LOAD BAL CTR
SVC Ø
Yes
No
V=R
Yes
Set V=R bit in RQE
No
Set V=V bit in RQE
Pg Fix Append
Yes
To User Pg Fix Appendage
No
IOB, ECB, DCB, DEB, AVT & Append To List
Fix User And/or IOS Pages on List
Subsystem
No
Set Page Supvr bit in RQE
Yes
Set Subsystem bit in RQE
Pg Fix Append
No
Yes
To User Pg fix Appendage
Fix Pages on User List

Type of Requester	CCW Chain Location	CCW Translation Requirements	Page Fixing Requirements	SVC Generated
Virtual	Pageable storage	IOS must translate CCW chains.	Control blocks and appendages must be fixed.	EXCP macro generates an SVC 0.
Fixed (VS subsystems only)	Pageable storage	CCW chains have been translated by the requester.	Control blocks and appendages have been fixed by the requester.	EXCPVR macro generates an SVC 114.
Virtual = Real (V = R)	V = R storage	None	None	EXCP macro generates an SVC 0.
Page Supervisor	V = R storage	None	None	EXCPVR macro generates an SVC 114.

Characteristics of I/O Requesters

0	0	TSTLNK Address of next RQE	TSTUCB Address of UCB
4	4	TSTTCB Task ID	TSTIOB Address of IOB
8	8	TSTPR Priority	TSTDEB Address of DEB
12	0C	TSTKEY Requester's Protection Key	TSTTCBAD Address of TCB
16	10	TSTHDR CCW Translation flags and header address	

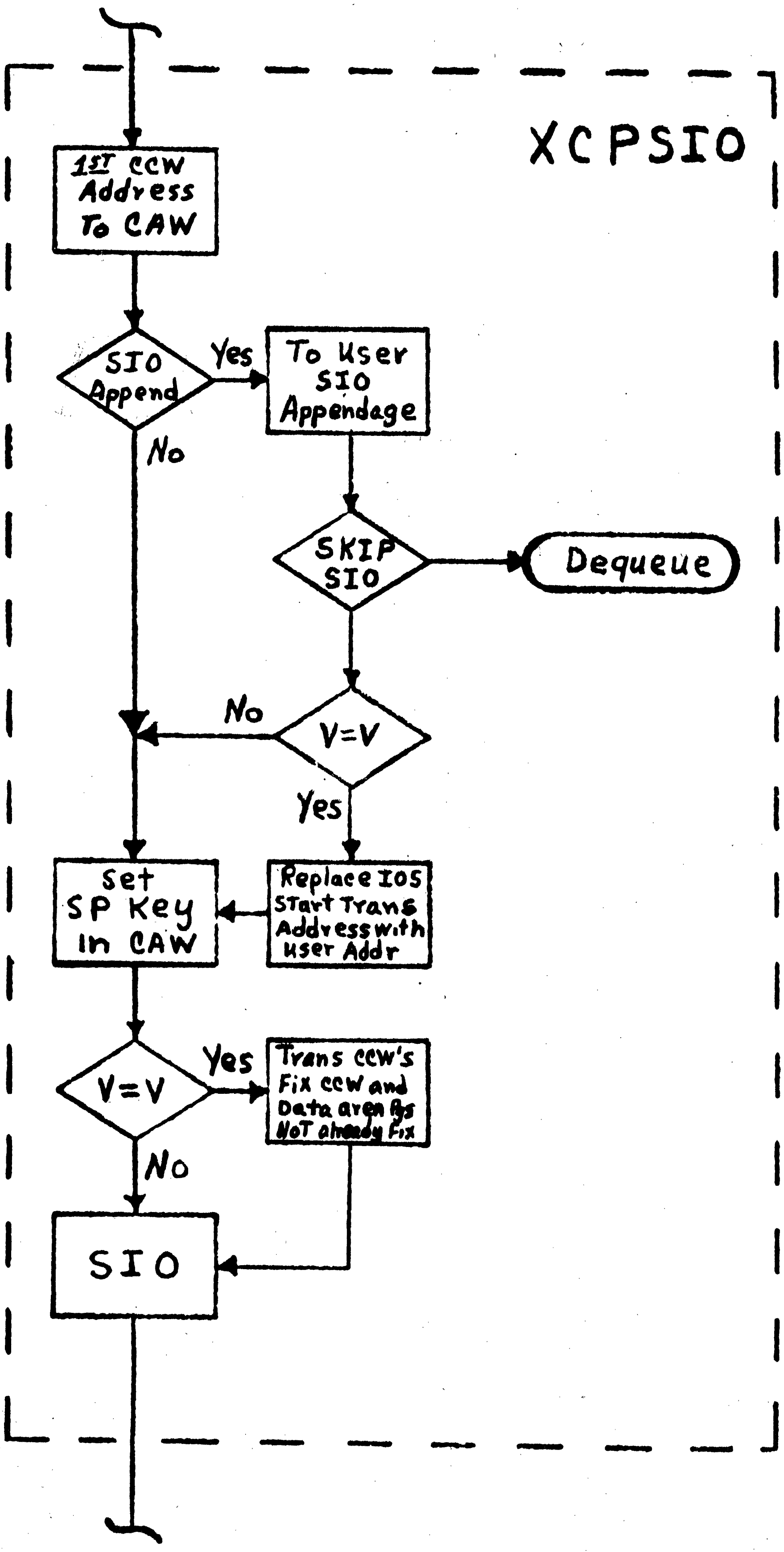
*Flags and Masks

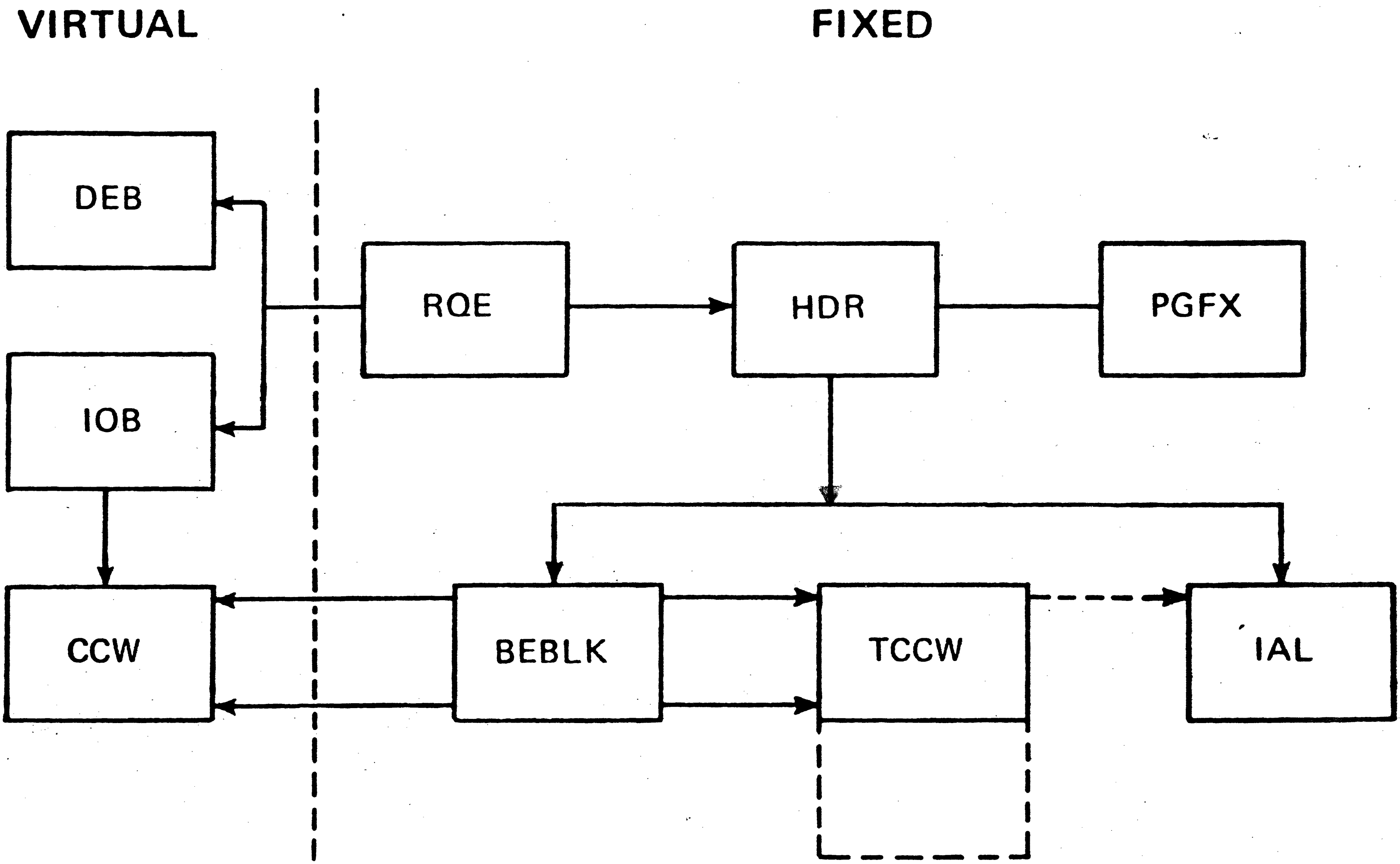
Flag Field	Contains	Hex Value	Means
TSTHDR	CCW Translation Flags	X'80'	Fixed requester (subsystem)
		X'40'	Virtual requester
		X'20'	Virtual=Real requester
		X'10'	Page supervisor is requester

CCW Translation Header Block

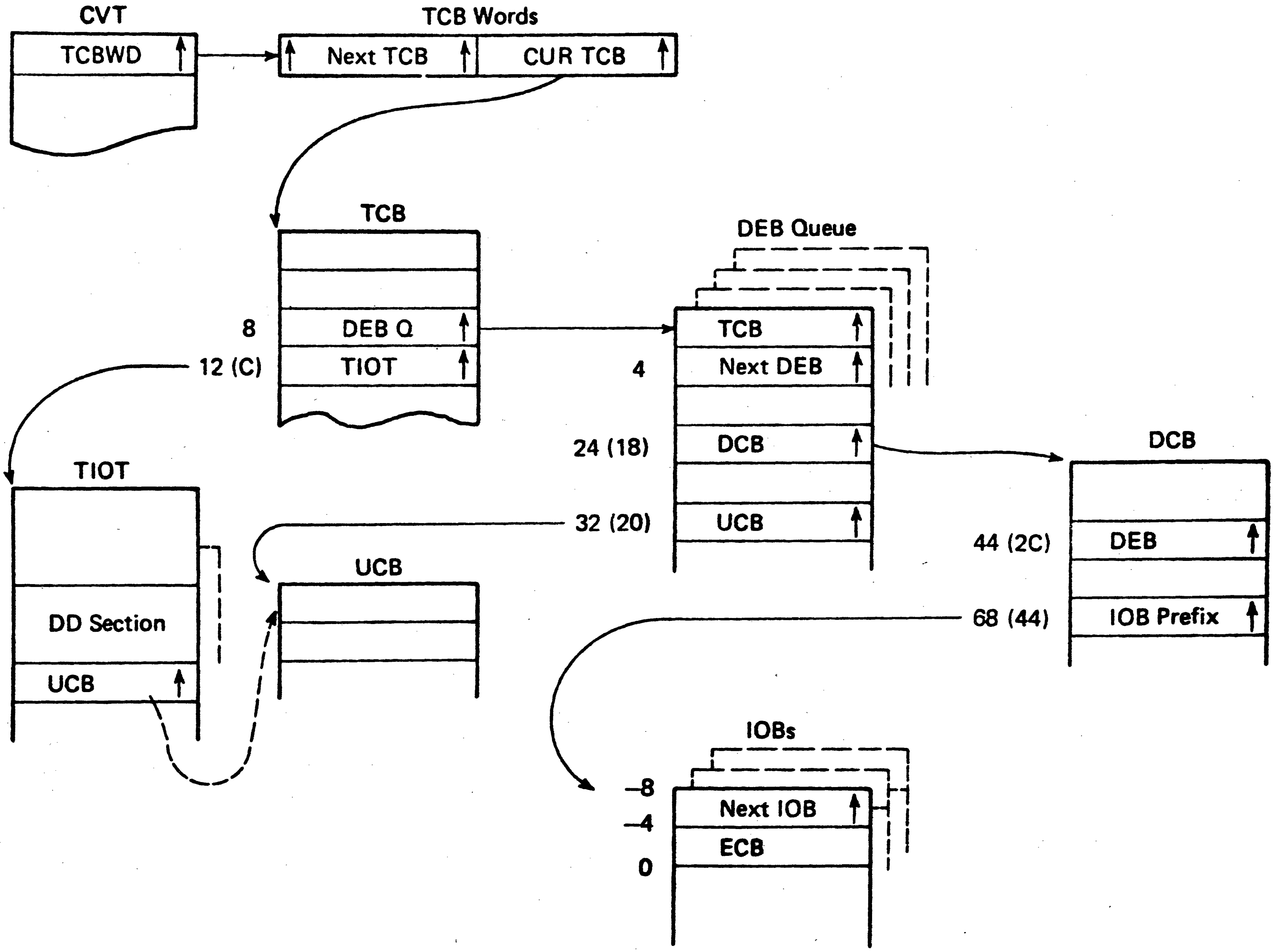
DEC	Hex			
0	0	HDRCHN Header Chain Pointer	HDRSIZE Size of SQA Area Being Used	HDRBEB Address of Area For First BEB
12	0C	HDRINDL Address of Area For Indirect Address List	HDRPGLK Address of Area For Page Fix List	HDRPLKR Address of Next Available Page Fix List Entry
24	18	HDRTCCW Address of CCW Translation Routine Work Area	HDRCAW Address of First Virtual CCW Address	HDRTICL Address of First Unresolved TIC CCW
36	24	HDRREGSV Register Save Area		

Field	DEC	HEX	Field	DEC	HEX
HDRBEB	0008	0008	HDRPLKR	0020	0014
HDRCAW	0028	001C	HDRREGSV	0036	0024
HDRCHN	0000	0000	HDRSIZE	0004	0004
HDRINDL	0012	000C	HDRTCCW	0024	0018
HDRPGLK	0016	0010	HDRTICL	0032	0020

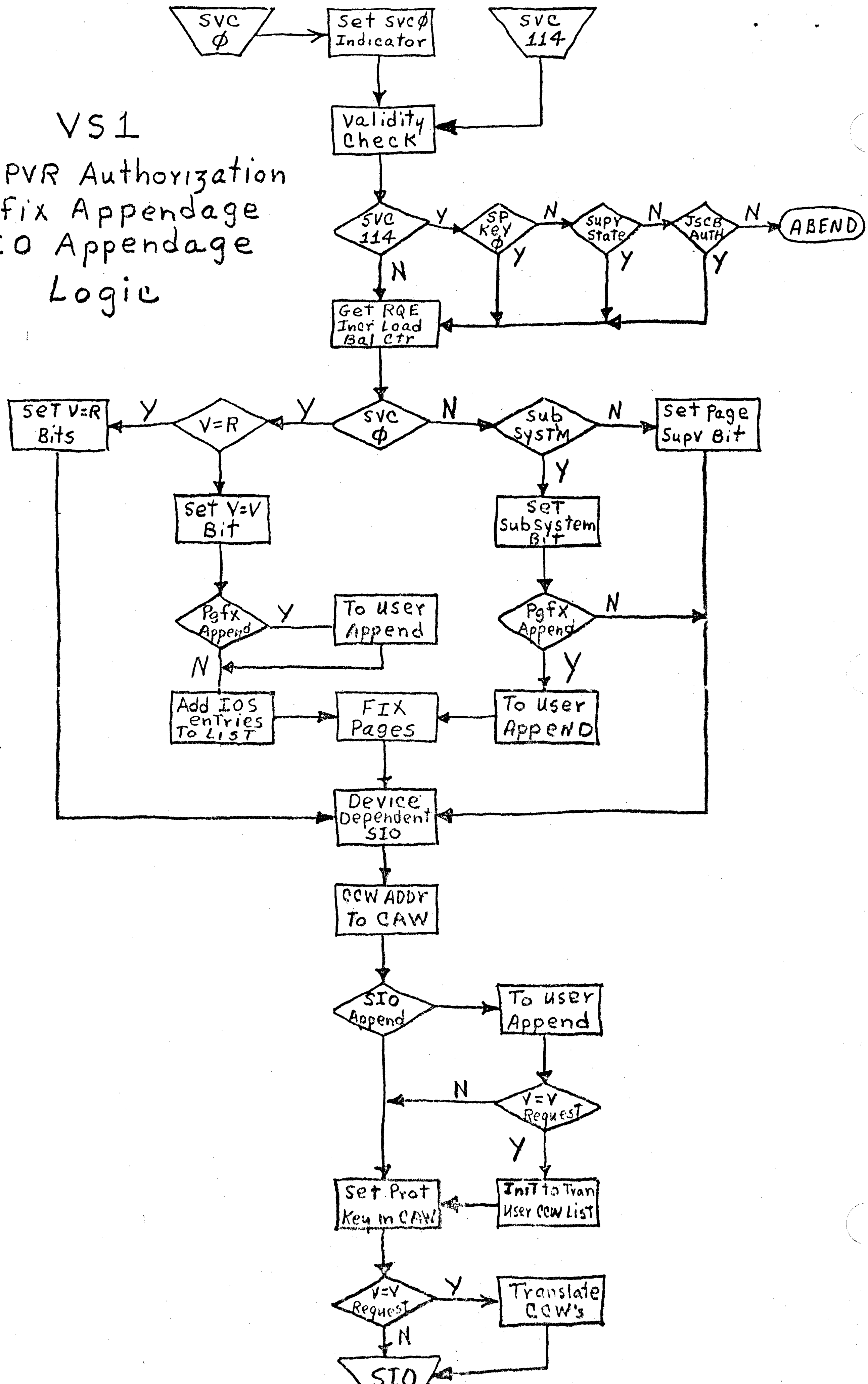

XCPSIO
1ST CCW Address To CAW
SIO Append
Yes
To User SIO Appendage
No
SKIP SIO
Dequeue
No
V=V
Yes
Set SP Key In CAW
Replace IOS Start Trans Address with user Addr
V=V
Yes
Trans ccw's Fix ccw and Data area pgs NoT already Fix
No
SIO


VIRTUAL
FIXED
DEB
RQE
HDR
PGFX
IOB
CCW
BEBLK
TCCW
IAL

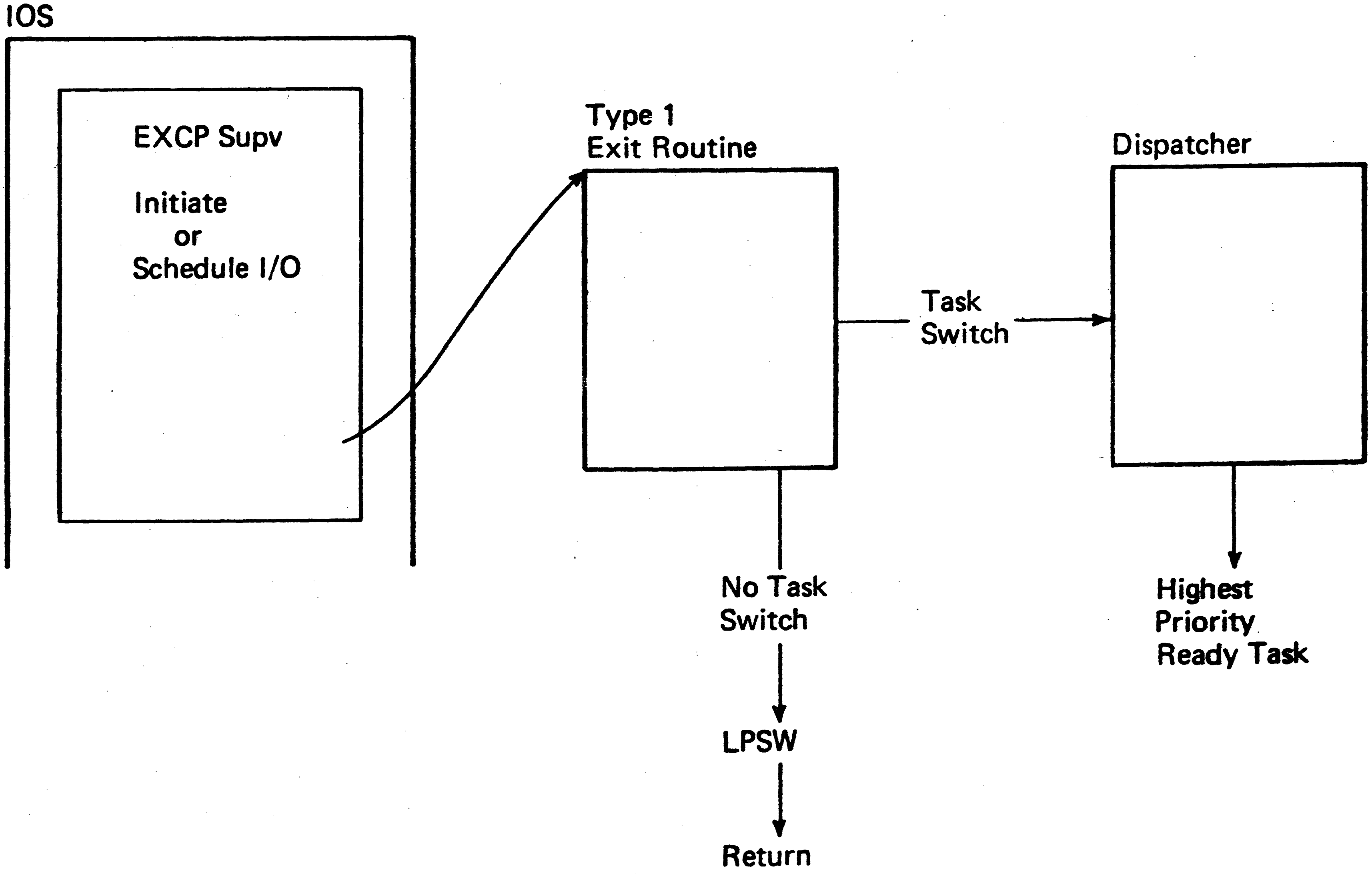
I/O Control Blocks

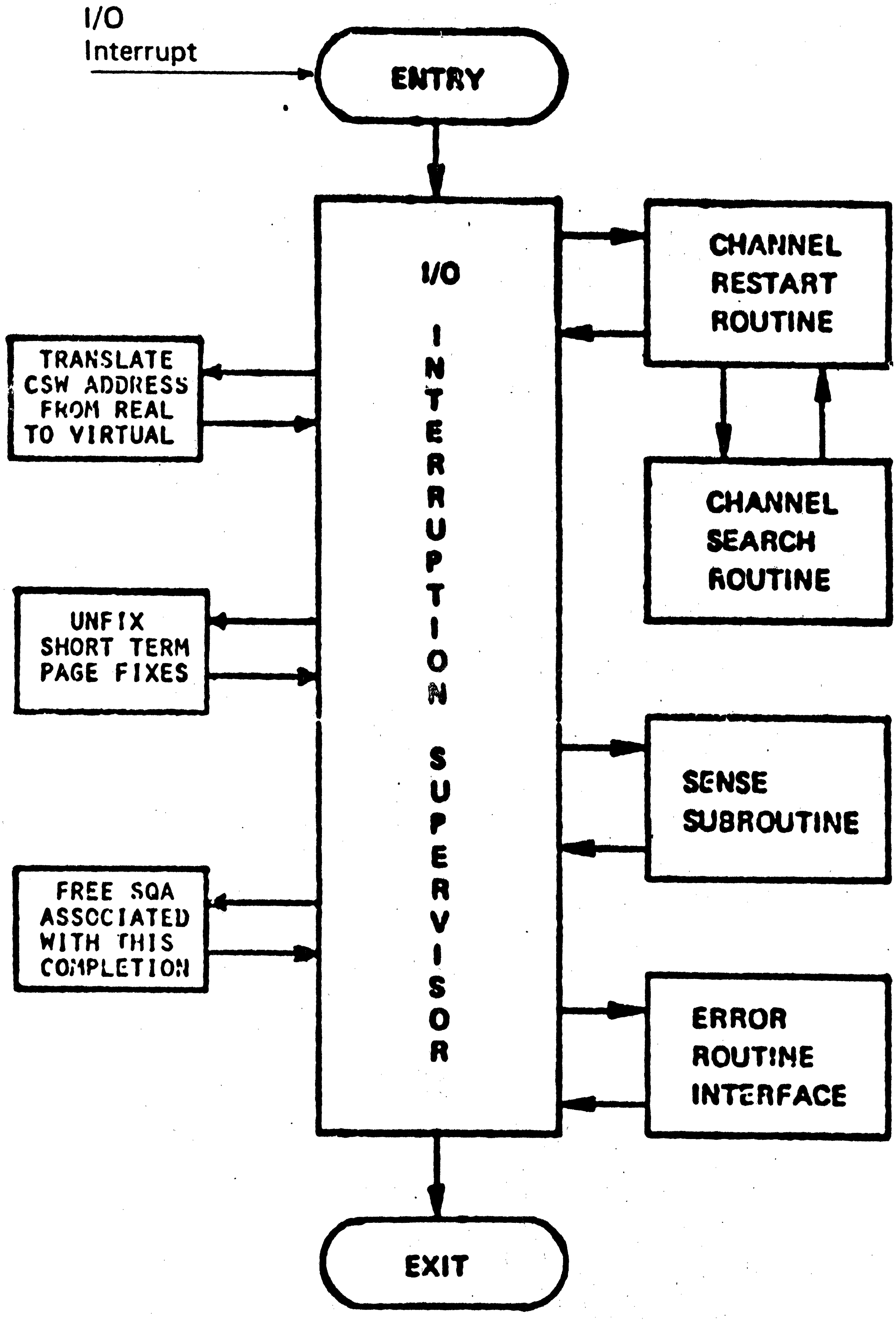


VS1
EXCPVR Authorization Pgfix Appendage SIO Appendage Logic



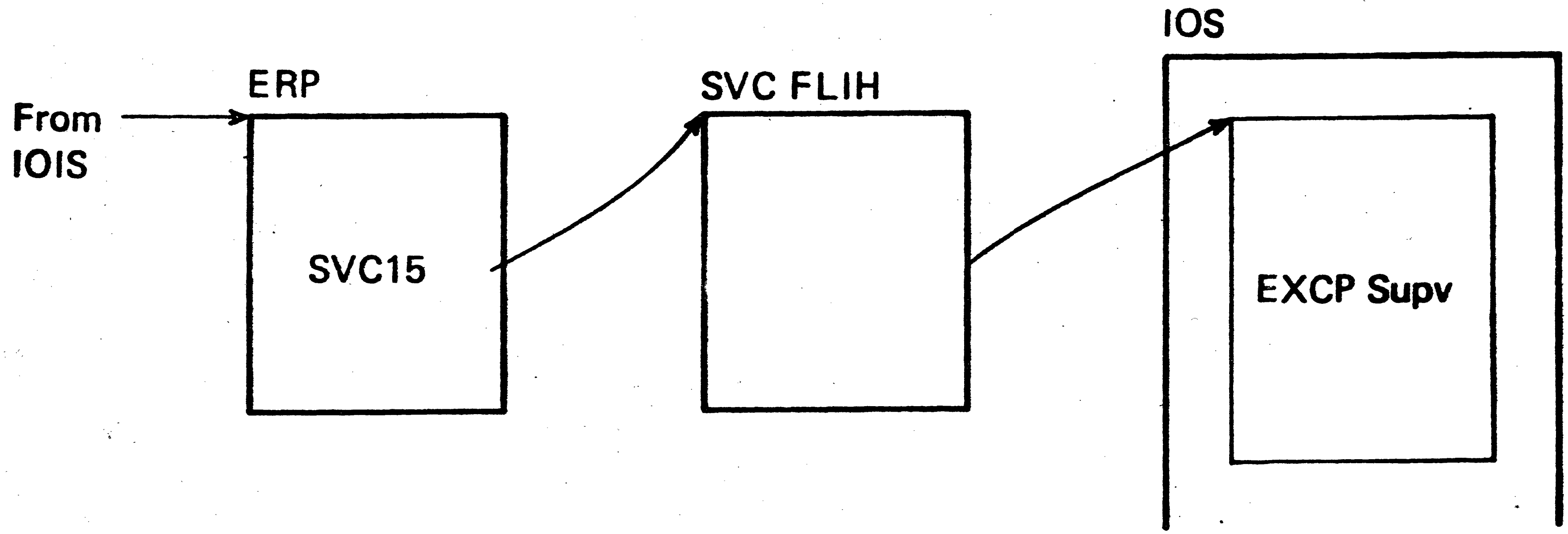
Exit from EXCP Supervisor

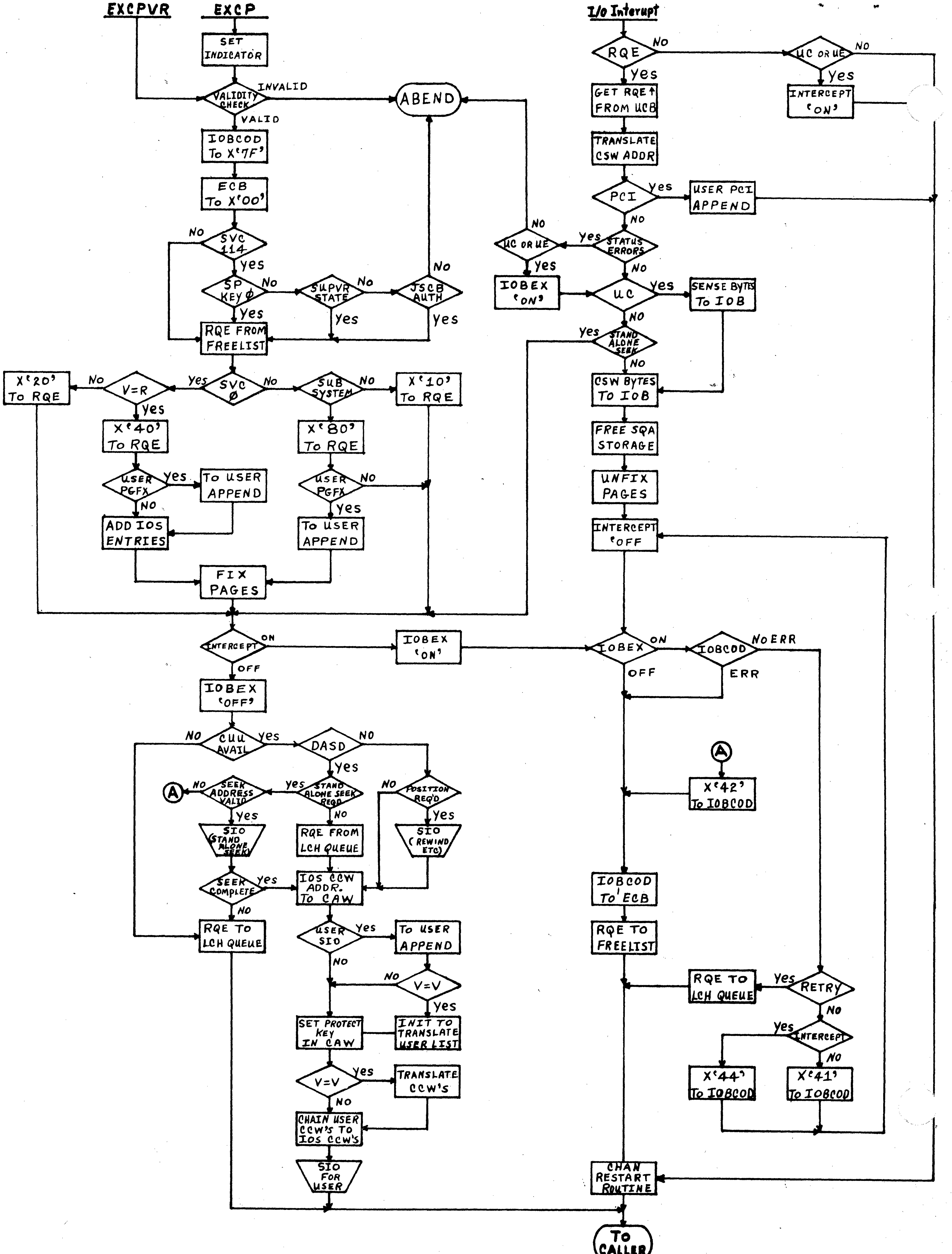


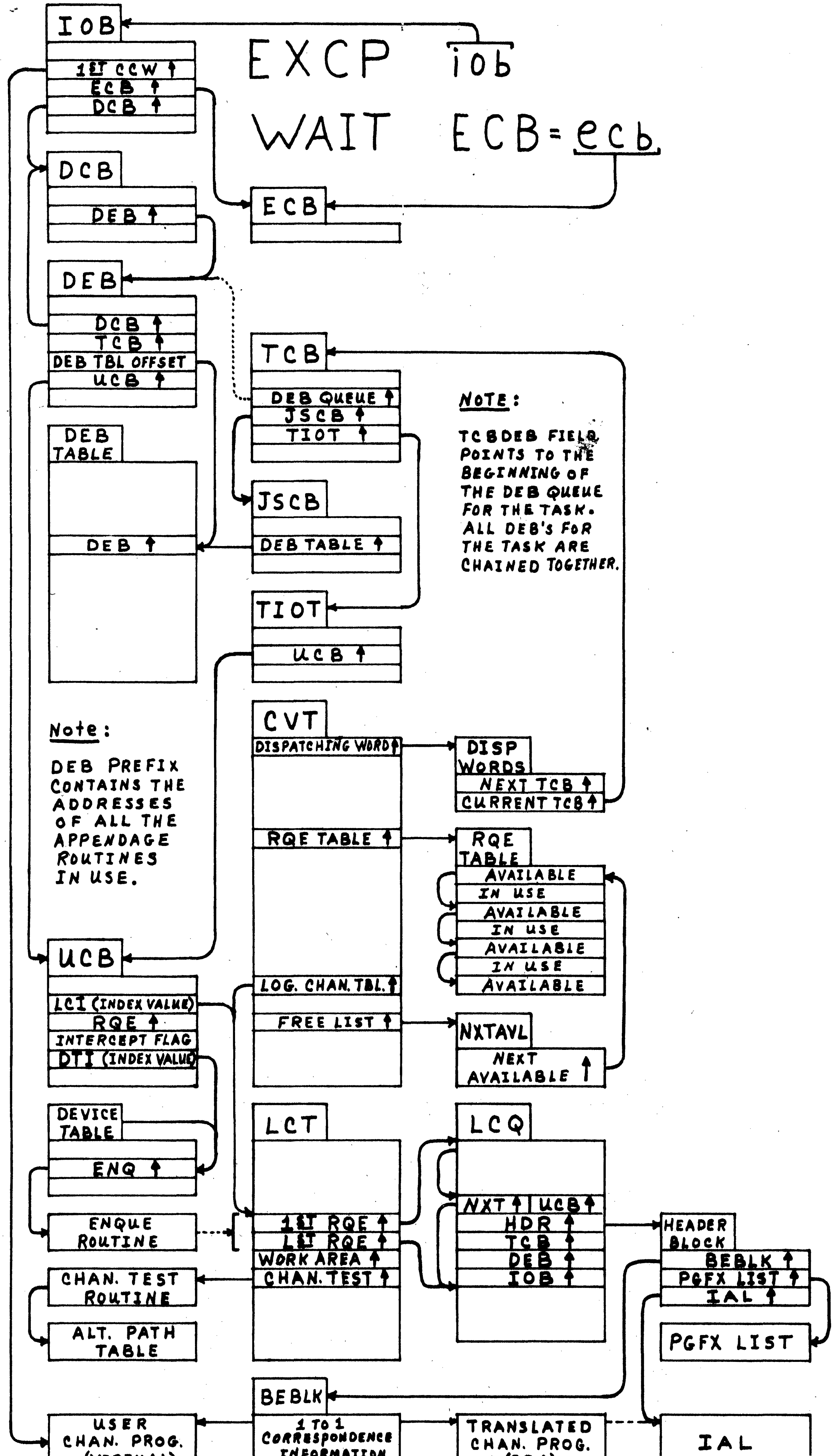

I/O
Interrupt
ENTRY
I/O INTERRUPTION SUPERVISOR
TRANSLATE CSW ADDRESS FROM REAL TO VIRTUAL
UNFIX SHORT TERM PAGE FIXES
FREE SQA ASSOCIATED WITH THIS COMPLETION
CHANNEL RESTART ROUTINE
CHANNEL SEARCH ROUTINE
SENSE SUBROUTINE
ERROR ROUTINE INTERFACE
EXIT

TO SUPERVISOR'S
TYPE 1 EXIT ROUTINE

Restarting I/O




EXCPVR
EXCP
SET INDICATOR
VALIDITY CHECK
INVALID
VALID
ABEND
IOBCOD To X'7F'
ECB To X'00'
SVC 114
NO
yes
SP KEY ∅
No
yes
SUPVR STATE
No
yes
JSCB AUTH
No
yes
RQE FROM FREELIST
SVC ∅
yes
No
V=R
No
yes
X'20' To RQE
X'40' To RQE
USER PGFX
yes
NO
To USER APPEND
ADD IOS ENTRIES
SUB SYSTEM
NO
X'10' To RQE
X'80' To RQE
USER PGFX
NO
yes
To USER APPEND
FIX PAGES
INTERCEPT
ON
OFF
IOBEX 'ON'
IOBEX 'OFF'
CUU AVAIL
NO
yes
DASD
NO
yes
SEEK ADDRESS VALID
NO
yes
A
STAND ALONE SEEK REQD
yes
NO
POSITION REQ'D
NO
yes
SIO (STAND ALONE SEEK)
RQE FROM LCH QUEUE
SIO (REWIND ETC)
SEEK COMPLETE
yes
NO
IOS CCW ADDR. To CAW
RQE To LCH QUEUE
USER SIO
yes
NO
To USER APPEND
V=V
NO
yes
SET PROTECT KEY IN CAW
INIT TO TRANSLATE USER LIST
V=V
yes
NO
TRANSLATE CCW'S
CHAIN USER CCW'S TO IOS CCW'S
SIO FOR USER
I/O Interupt
RQE
No
yes
GET RQE ↑ FROM UCB
UC OR UE
NO
yes
INTERCEPT 'ON'
TRANSLATE CSW ADDR
PCI
yes
NO
USER PCI APPEND
STATUS ERRORS
yes
NO
UC OR UE
NO
yes
IOBEX 'ON'
UC
yes
NO
SENSE BYTES To IOB
STAND ALONE SEEK
yes
NO
CSW BYTES To IOB
FREE SQA STORAGE
UNFIX PAGES
INTERCEPT 'OFF
IOBEX
ON
OFF
IOBCOD
NO ERR
ERR
A
X'42' To IOBCOD
IOBCOD To ECB
RQE TO FREELIST
RQE TO LCH QUEUE
RETRY
Yes
NO
INTERCEPT
Yes
NO
X'44' To IOBCOD
X'41' To IOBCOD
CHAN RESTART ROUTINE
TO CALLER



EXCP iob

WAIT ECB=ecb

NOTE:

TCBDEB FIELD POINTS TO THE BEGINNING OF THE DEB QUEUE FOR THE TASK. ALL DEB'S FOR THE TASK ARE CHAINED TOGETHER.

Note:

DEB PREFIX CONTAINS THE ADDRESSES OF ALL THE APPENDAGE ROUTINES IN USE.

PAGE MGMT

IBM

SYSTEMS PROGRAMMING

Student Materials

This material was produced for Educational purposes only. Changes may obsolete part or all of this publication. No responsibility is assumed for any inaccuracies that occur.

Los Angeles Advanced Education Center
3424 WILSHIRE BOULEVARD • LOS ANGELES, CALIF. 90010

VS/1 PAGE MANAGEMENT

REFERENCES:

OS/VS1 Planning and Use Guide	GC24-5090
OS/VS1 Debugging Guide	GC24-5093
OS/VS1 System Generation Reference	GC26-3791
Operator's Library:OS/VS1 Reference	GC38-0110
OS/VS1 Supervisor Logic	SY24-5155

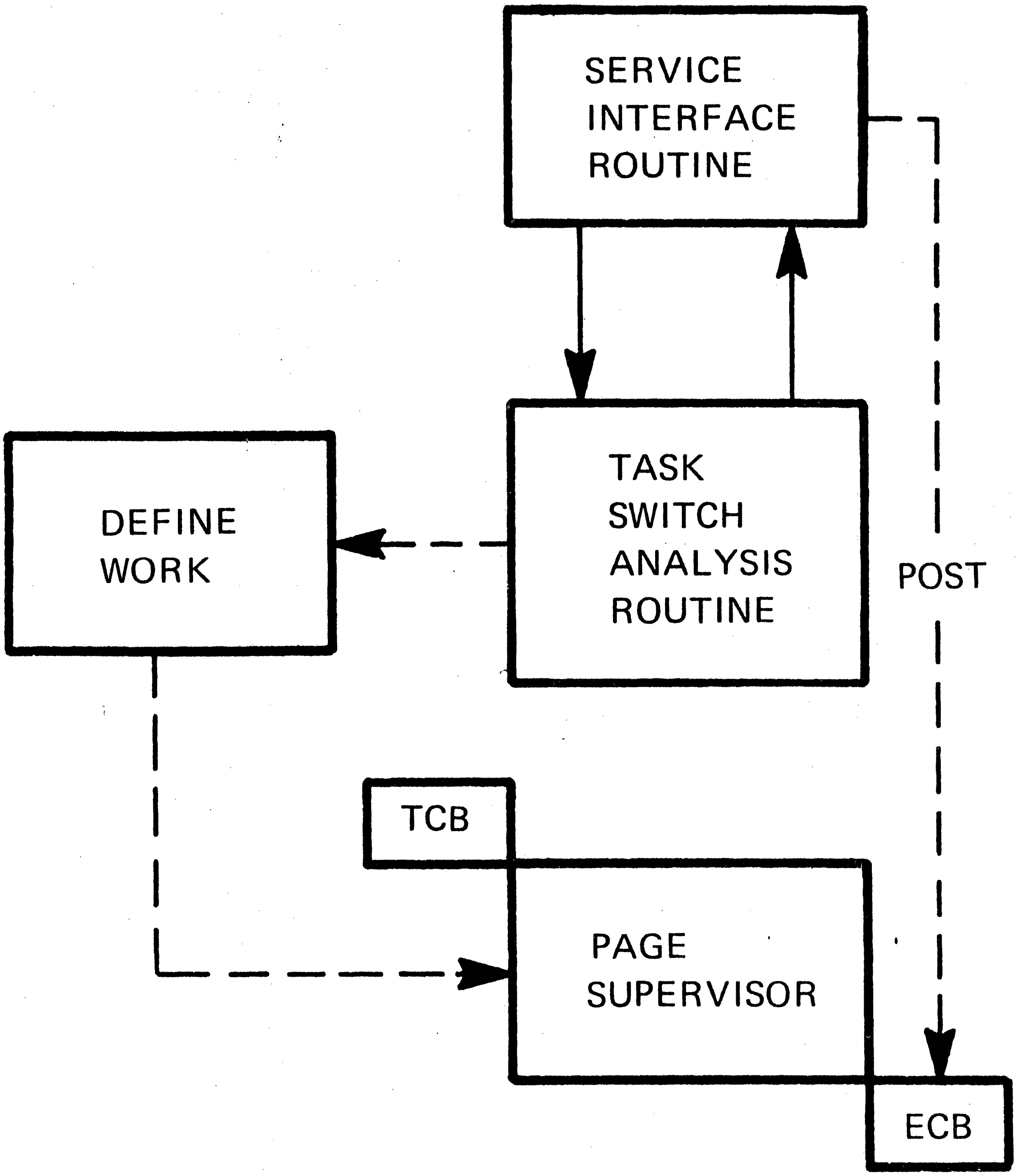
WHY A PAGE SUPERVISOR

- ENSURE THAT CONTENTS OF VIRTUAL STORAGE ARE ADDRESSABLE

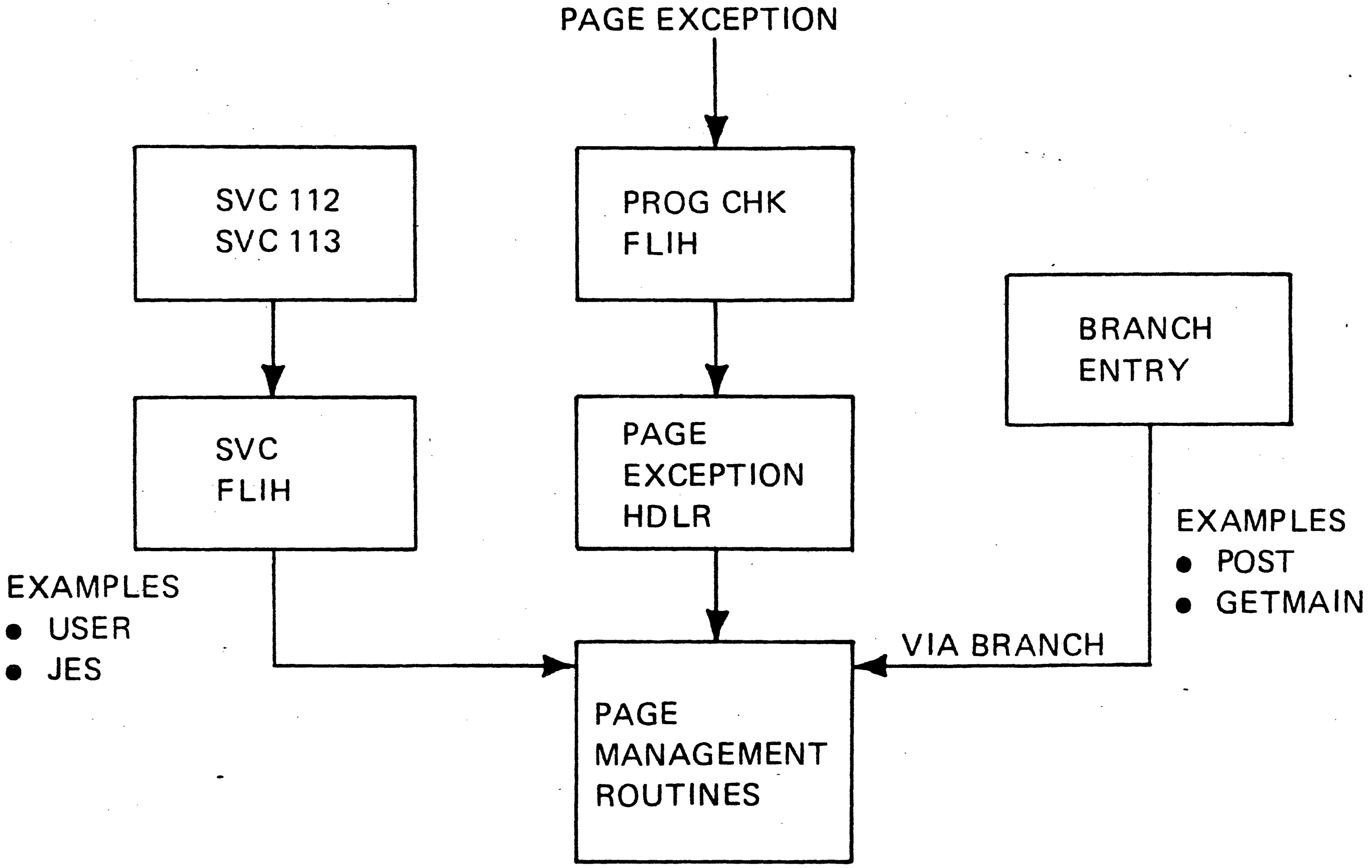
- PROVIDE EFFICIENT EXCHANGE OF PAGES BETWEEN REAL AND EXTERNAL STORAGE

- MAINTAIN STATUS OF
  - REAL STORAGE
  - VIRTUAL STORAGE

PAGE MANAGEMENT ROUTINES



ENTRY TO PAGE SUPERVISOR



PAGE EXCEPTION

SVC 112
SVC 113

PROG CHK
FLIH

BRANCH
ENTRY

SVC
FLIH

PAGE
EXCEPTION
HDLR

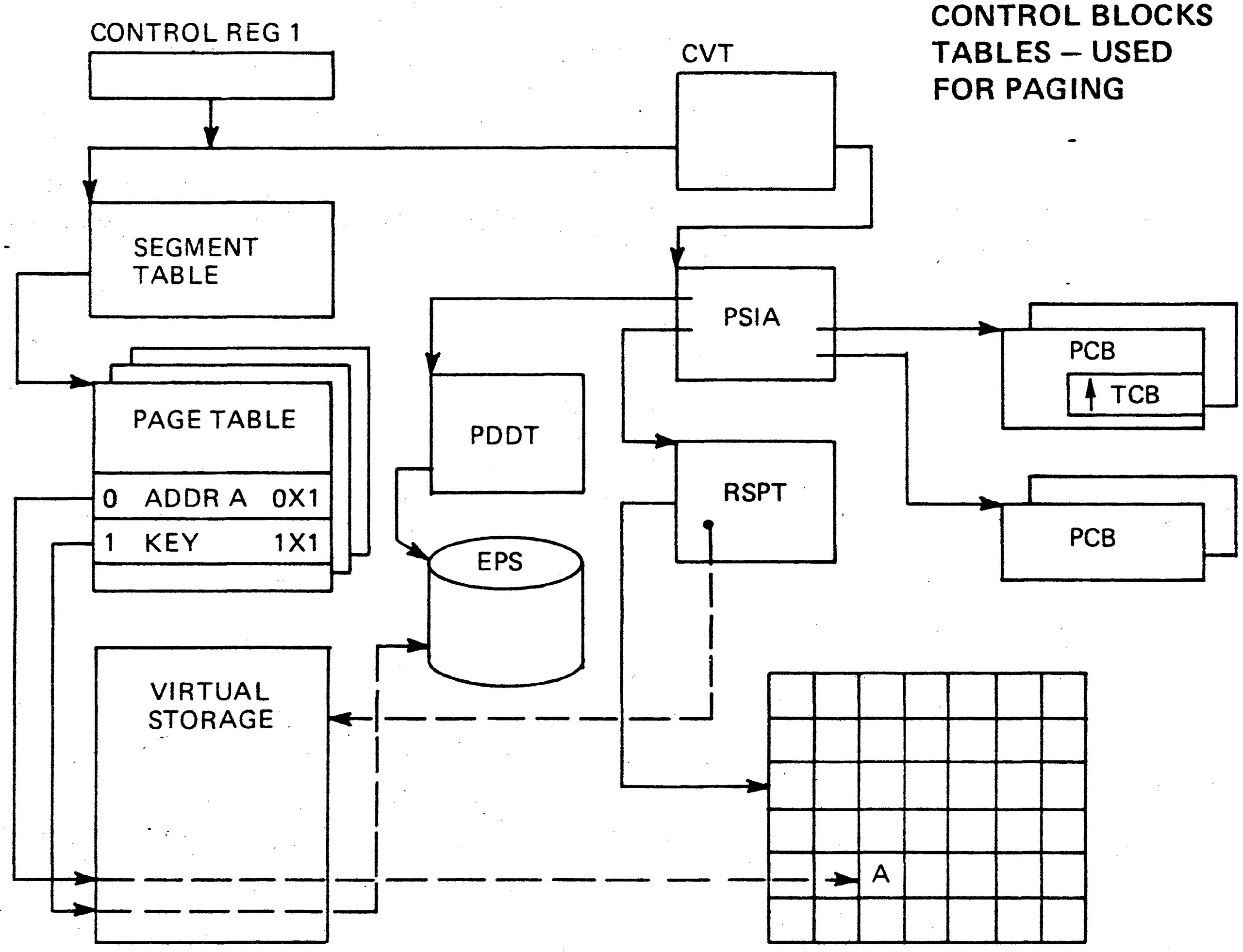
EXAMPLES
- USER
- JES

EXAMPLES
- POST
- GETMAIN

VIA BRANCH

PAGE
MANAGEMENT
ROUTINES

CONTROL BLOCKS TABLES – USED FOR PAGING



PAGE TABLE ENTRY

K		I		U
0	PAGE ADDRESS	1		1

K		I		U
1	KEY	1		0

K		I		U
1	KEY	1		1

K		I		U
0	PAGE ADDRESS	0		1

PSIA

HEX				
0	FLAG BYTE	ADDRESS OF PAGE SUPERVISOR RB	ADDRESS OF REAL STORAGE PAGE TABLE	
8	ADDRESS OF PAGE DEVICE DESCRIPTOR TABLE		ADDRESS OF APCB/APCBE TABLE	
10				
18				
20				
28	POINTER TO PAGEABLE SUPERVISOR BBOX FOR BATCH PAGE POST		NUMBER OF PAGEABLE PAGES	NUMBER OF PAGES FROM END OF NUCLEUS TO V=R BOUNDARY
30 ≈	PCB QUEUE HEADERS			
68		RSPT-DISPLACEMENT TO END OF NUCLEUS	RSPT DISPLACEMENT TO V=R BOUNDARY	RSPT DISPLACEMENT TO END OF REAL STORAGE
70	REAL STORAGE FIX COUNTS			
78			AVAILABLE PAGE QUEUE LOW THRESHOLD VALUE	AVAILABLE PAGE QUEUE HIGH THRESHOLD VALUE
80	AVAILABLE PAGE FRAME COUNT	SHORT TERM FIX THRESHOLD	LONG TERM FIX THRESHOLD	SVC FIX THRESHOLD
88 ≈				
130				

REAL STORAGE PAGE TABLE

<table>
<tr><td colspan="2">FORWARD LINK</td><td colspan="2">BACKWARD LINK</td><td colspan="2">VIRTUAL PAGE NO.</td><td>FIX COUNT</td></tr>
<tr><td>Q IND</td><td>TCB ID</td><td>FLGS</td><td>INDEX</td><td></td><td colspan="2"></td></tr>
</table>

CVT + X'15C' ↑ PSIA

PSIA + X'4' ↑ RSPT

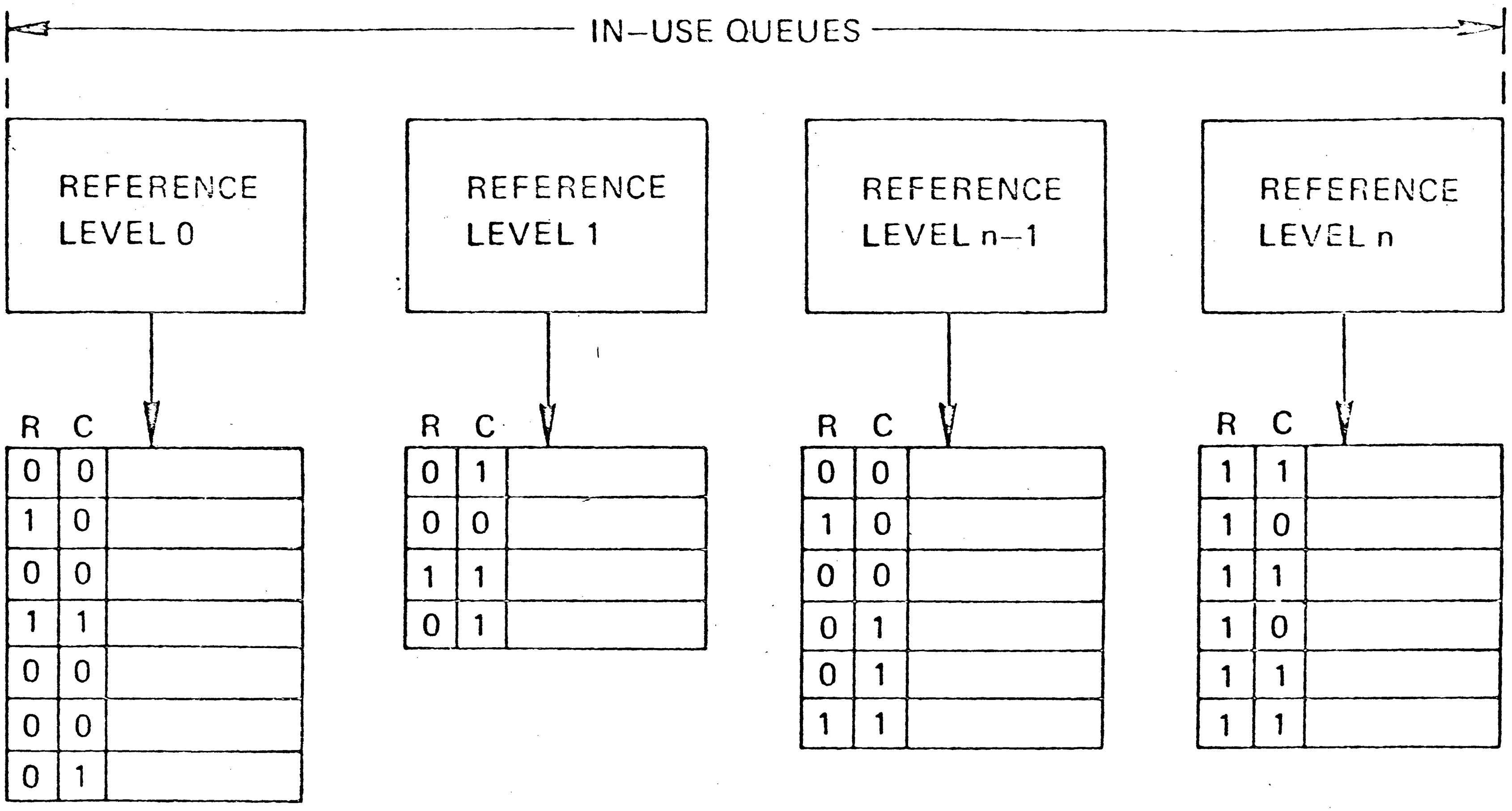
ONE ENTRY PER FRAME OF REAL STORAGE

EACH ENTRY – 16 BYTES

Virtual Page Number

- Bytes 4 and 5 of RSPTE
- Easily Converted to Virtual Address
- Example:

  VPN = 01D9
  SLL – Eleven Bits 0000 0001 1101 1001
  Result = 1110 1100 1000 0000 0000
  Virtual Address = EC800



R	C	
0	0	
1	0	
0	0	
1	1	
0	0	
0	0	
0	1	

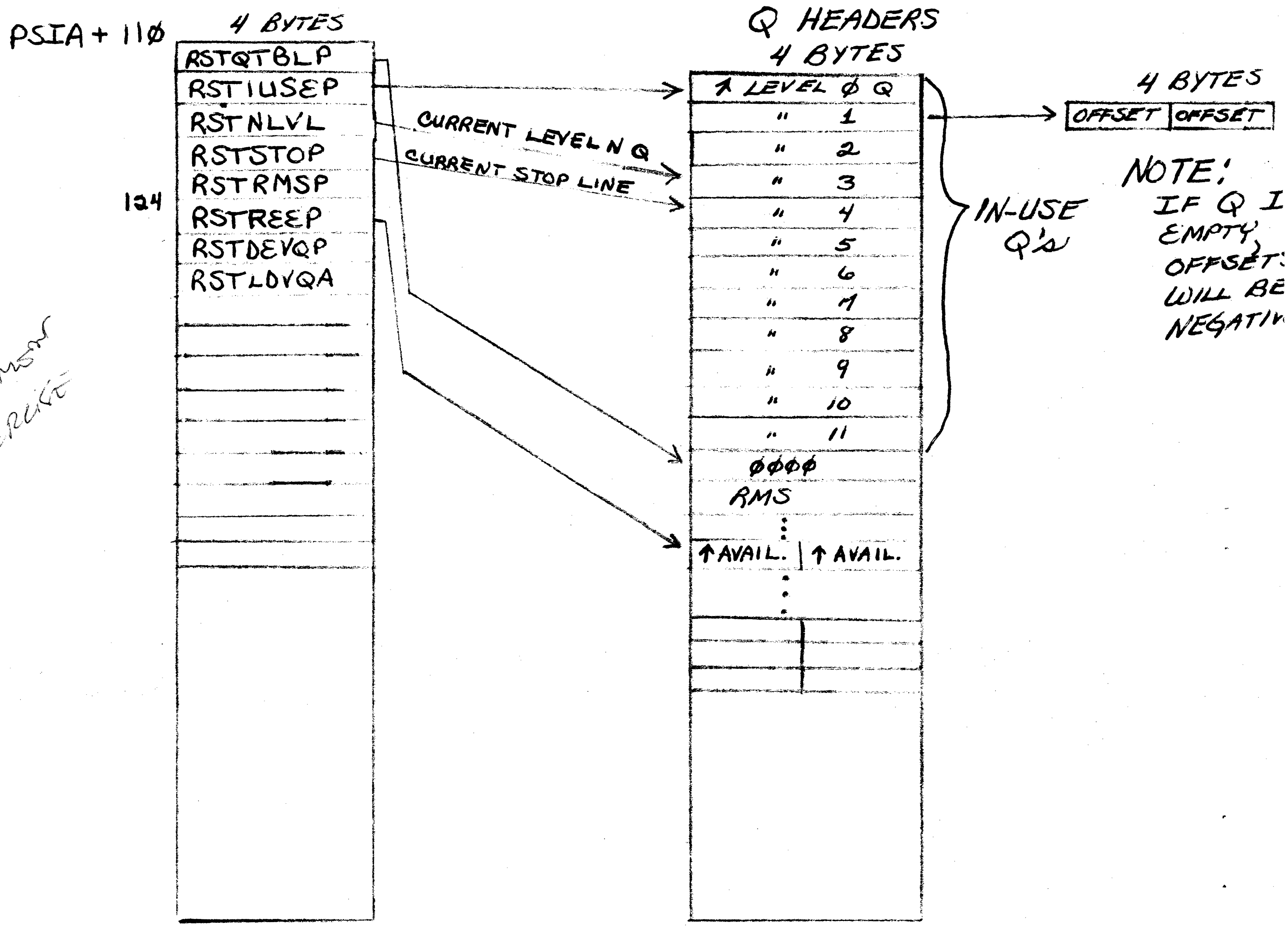
R	C	
0	1	
0	0	
1	1	
0	1	

R	C	
0	0	
1	0	
0	0	
0	1	
0	1	
1	1	

R	C	
1	1	
1	0	
1	1	
1	0	
1	1	
1	1	

R = REFERENCE BIT
C = CHANGE BIT

Q HEADER LOCATION



[illegible]
VS1 FREE MANAGEMENT DUMP EXERCISE

SAMPLE QUEUE STRUCTURE

AVAILABLE PAGE COUNT – 4

AVAILABLE PAGE QUEUE	200	2C0
LEVEL 0 QUEUE	2D0	250
LEVEL 1 QUEUE	210	210
LEVEL N QUEUE	2A0	290

REAL STORAGE PAGE TABLE

1000						
≈						
1200	240	FFFE	PCBE	FFF8	FFF8	RSPTE
1220				2C0	240	PCBE
1240	230	200	PCBE	FFF9	2B0	RSPTE
1260	2B0	2D0	RSPTE			
1280				FFF6	2A0	RSPTE
12A0	290	FFF6	RSPTE	250	260	RSPTE
12C0	FFFE	230	PCBE	260	FFF9	RSPTE
12E0						

PDDT

0	LOW VIRTUAL PAGE NUMBER IN EXTENT	HIGH VIRTUAL PAGE NUMBER IN EXTENT	NUMBER OF RECORD/CYLINDER FOR DEVICE	ADDR. OF DEB DEV. DEPEND. SECTION
8	NO. OF RECORD/TRACK FOR DEVICE	ADDRESS OF DEB's BASIC SECTION	NOTE 1 DEVICE TYPE BYTE	POINTER TO IOB ASSOCIATED WITH DEB NOTE 2

NOTE 1: THIS FIELD CONTAINS THE SAME VALUE FOR EACH DEVICE AS THE UCB DEVICE TYPE BYTE FOR THAT DEVICE.

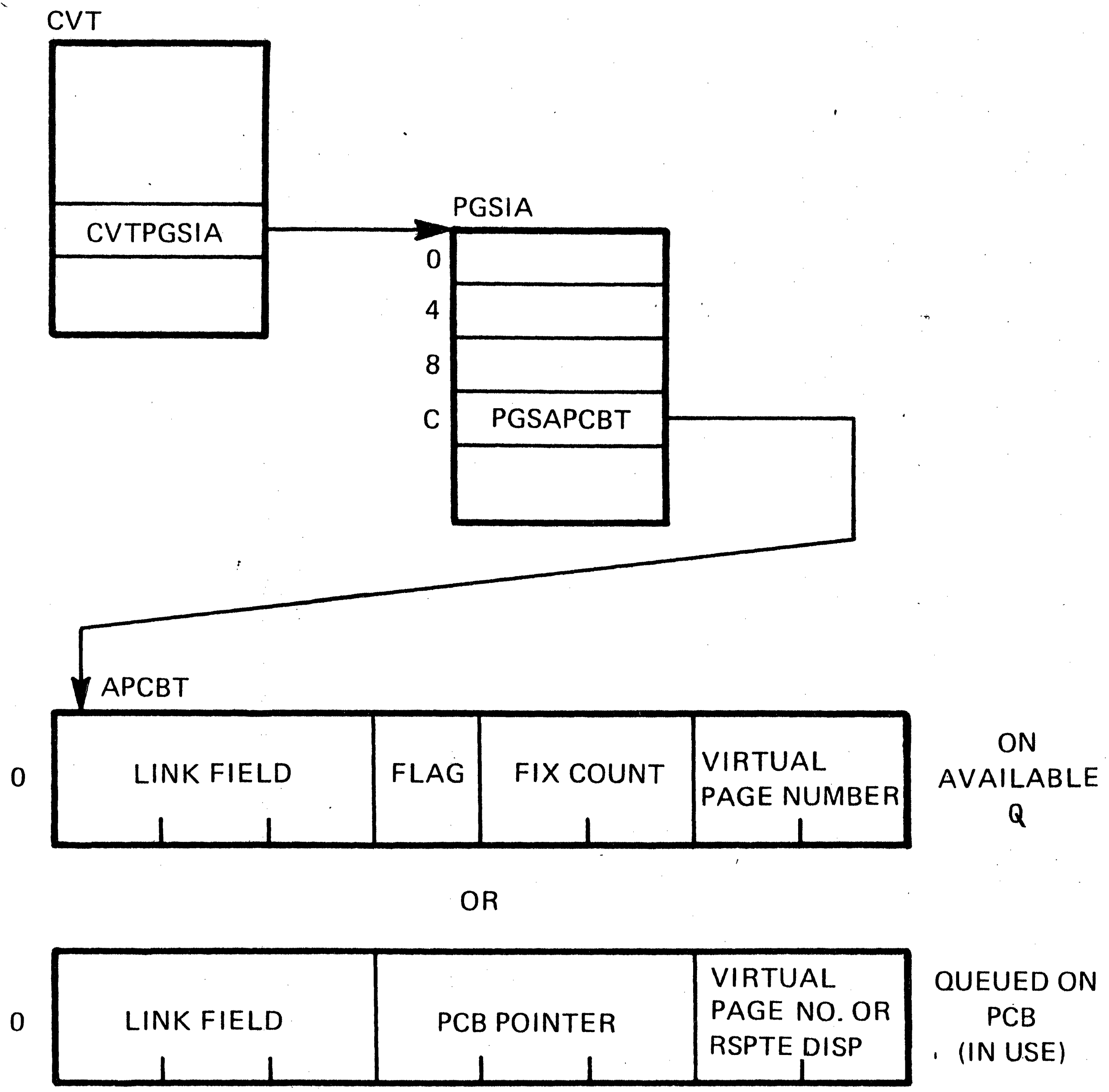
NOTE 2: THIS FIELD POINTS TO THE ADDRESS OF A LIST OF THREE IOB ADDRESSES IF THE DEVICE IS A 2305-2.

DEVICE TYPE BYTE	X'07'	2305-2 DASD
	X'08'	2314/2319 DASD
	X'09'	3330 /3333
	X'0A'	3340 w/o RPS
	X'0D'	3330/3333 MODEL 1
	X'8A'	3340 w/ RPS

PCB

<table>
<tr><th>HEX</th><th colspan="7"></th></tr>
<tr><td>0</td><td>PCBFLG1</td><td colspan="3">FORWARD QUEUE ADDRESS</td><td>PCBFLG2</td><td colspan="2">BACKWARD QUEUE ADDRESS</td></tr>
<tr><td>8</td><td colspan="4">PCBTCB
ADDRESS OF TCB</td><td colspan="3">PCBREG
REGISTER SAVE AREA ADDRESS</td></tr>
<tr><td>10</td><td>PCBENF1
ENTRY
FLAG BYTE</td><td colspan="3">PCBENBG
BEGIN ADDRESS OR ADDRESS
OF FIRST LIST ENTRY</td><td>PCBENF2
ENTRY
FLAG BYTE</td><td colspan="2">PCBENED
END ADDRESS PLUS ONE OR
UNUSED IF LIST</td></tr>
<tr><td>18</td><td colspan="4">PCBPCBE
PCBE LINK FIELD (BEGIN PAGE
SUPERVISOR WORK AREA)</td><td colspan="3">PCBAPCB
FLOATING APCB OR APCBE HEADER</td></tr>
<tr><td>20</td><td colspan="4">PCBAPCB (CONTINUED)</td><td colspan="2">PCBALCT
PAGE ALLOCATION
COUNT</td><td>PCBIOCT OR
PCBRQCT</td></tr>
<tr><td>28</td><td>PCBFLG4</td><td>PCBFLG3</td><td>PCBALID</td><td>RESERVED</td><td colspan="3">PCBECB OR PCBECBA
ACTUAL ECB OR ECB ADDRESS (IF AN
SVC REQUEST) END OF TCB PCB</td></tr>
<tr><td>30</td><td colspan="7">PCBPSW
PSW SAVE AREA</td></tr>
<tr><td>38</td><td colspan="7">PCBSAVE
REGISTER SAVE AREA (16 FULLWORDS)</td></tr>
</table>

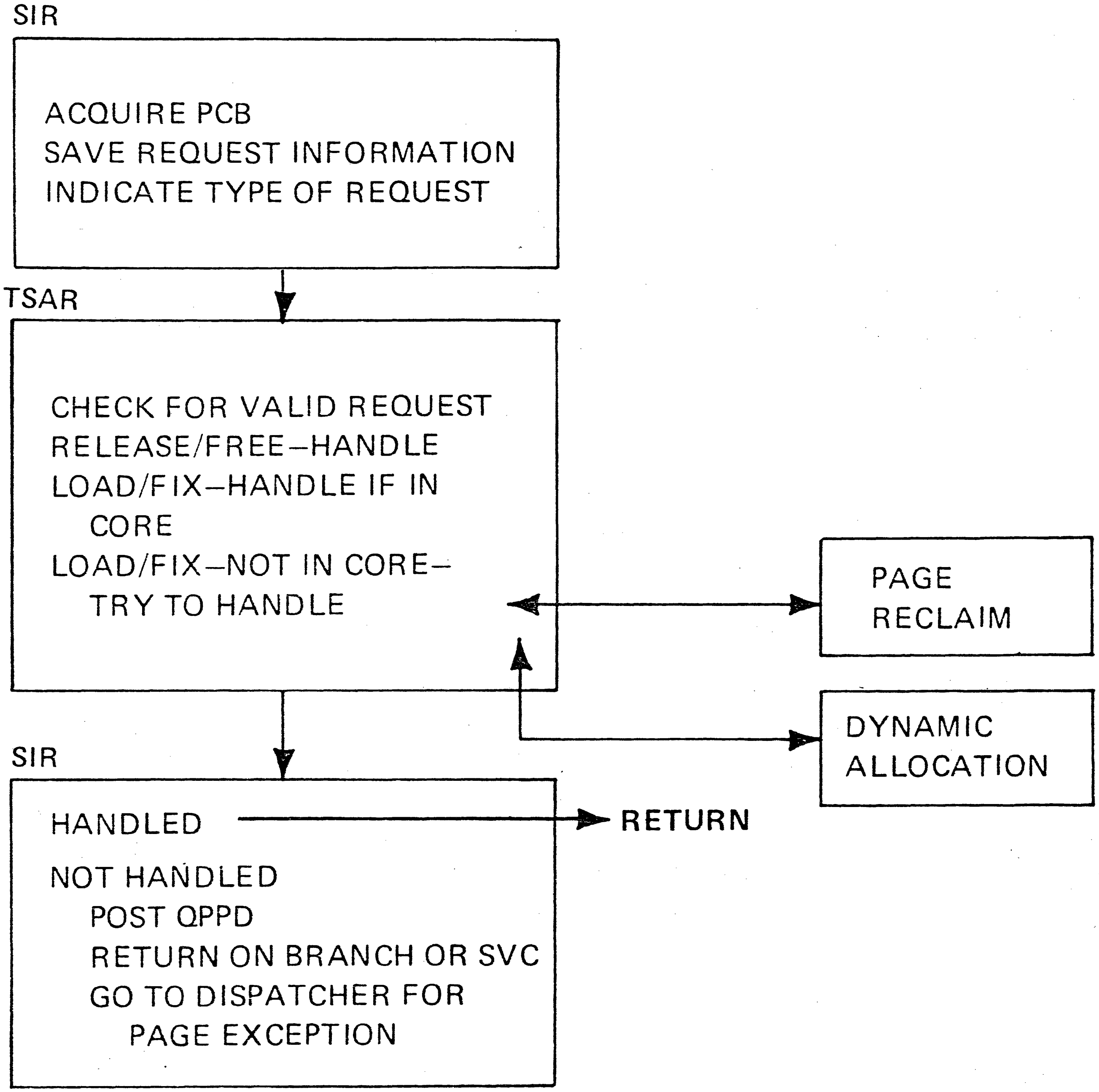
APCBT FORMAT AND LOCATION



- 1 - 8 BYTE ENTRY/PAGEABLE PAGE FRAME

PAGE SUPERVISOR PCB QUEUE HEADERS

HEX	
PSIA + 30	SIR POST QUEUE (SERVICE-OUT QUEUE)
38	PREALLOCATION FIX OVERFLOW QUEUE
40	PREALLOCATION APCBE DEPLETION QUEUE
48	DYNAMIC ALLOCATION INPUT QUEUE
50	I/O PROCESSOR QUEUE = PCB PAGING QUEUE
58	V = R DEFERRED ALLOCATION QUEUE
60	PCB DEACTIVATION QUEUE



SIR

ACQUIRE PCB
SAVE REQUEST INFORMATION
INDICATE TYPE OF REQUEST

TSAR

CHECK FOR VALID REQUEST
RELEASE/FREE–HANDLE
LOAD/FIX–HANDLE IF IN
CORE
LOAD/FIX–NOT IN CORE–
TRY TO HANDLE

PAGE
RECLAIM

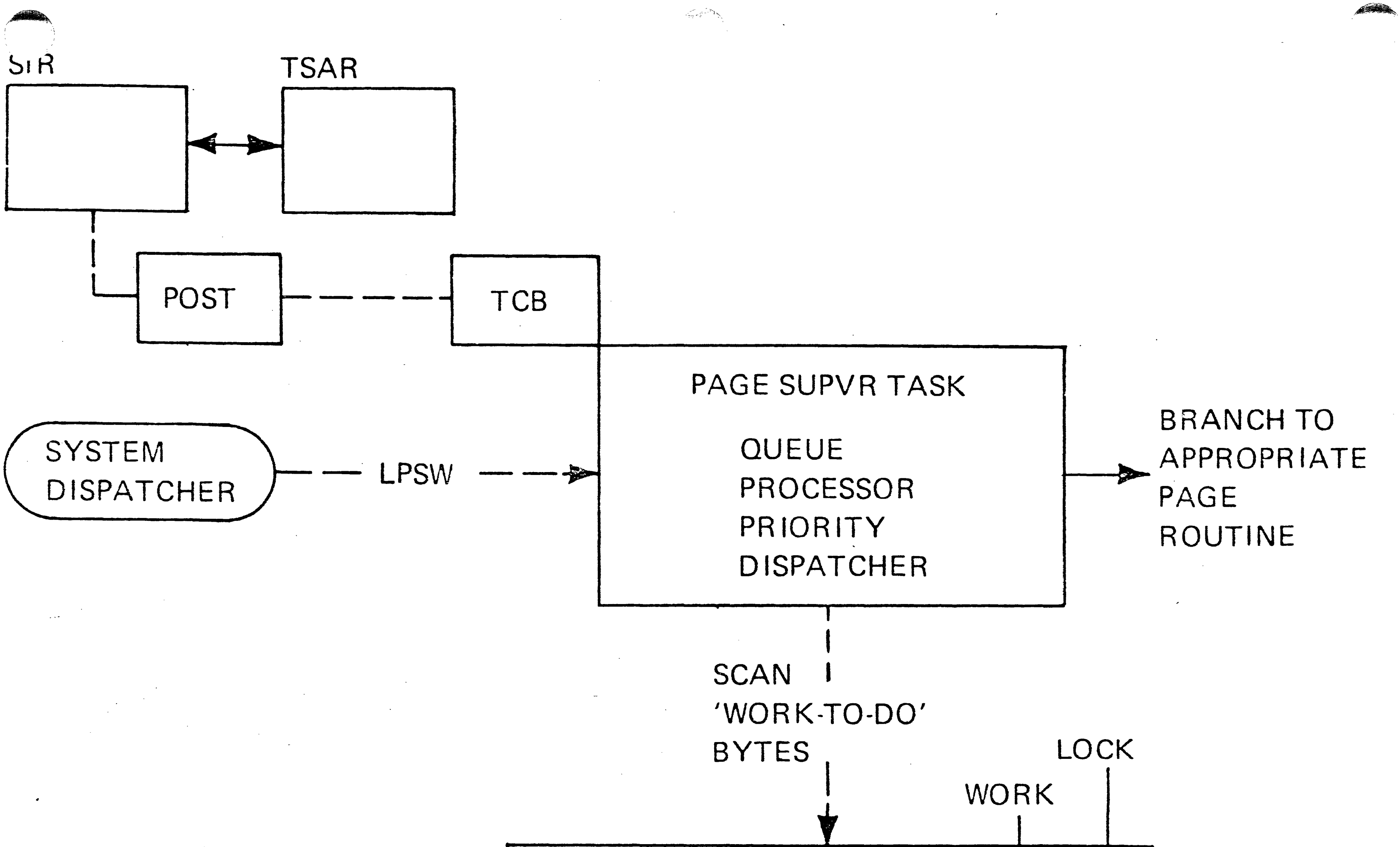
DYNAMIC
ALLOCATION

SIR

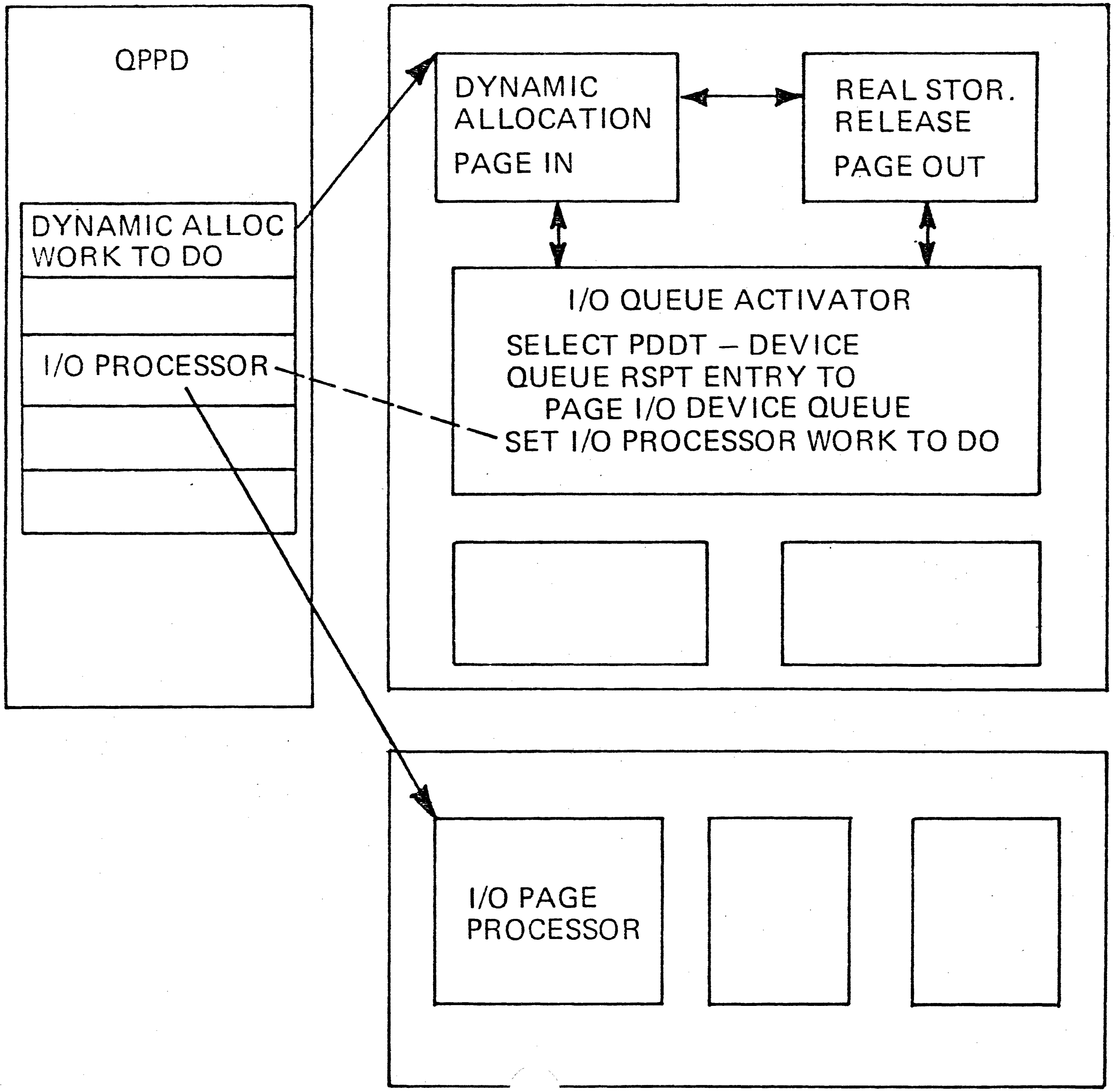
HANDLED

RETURN

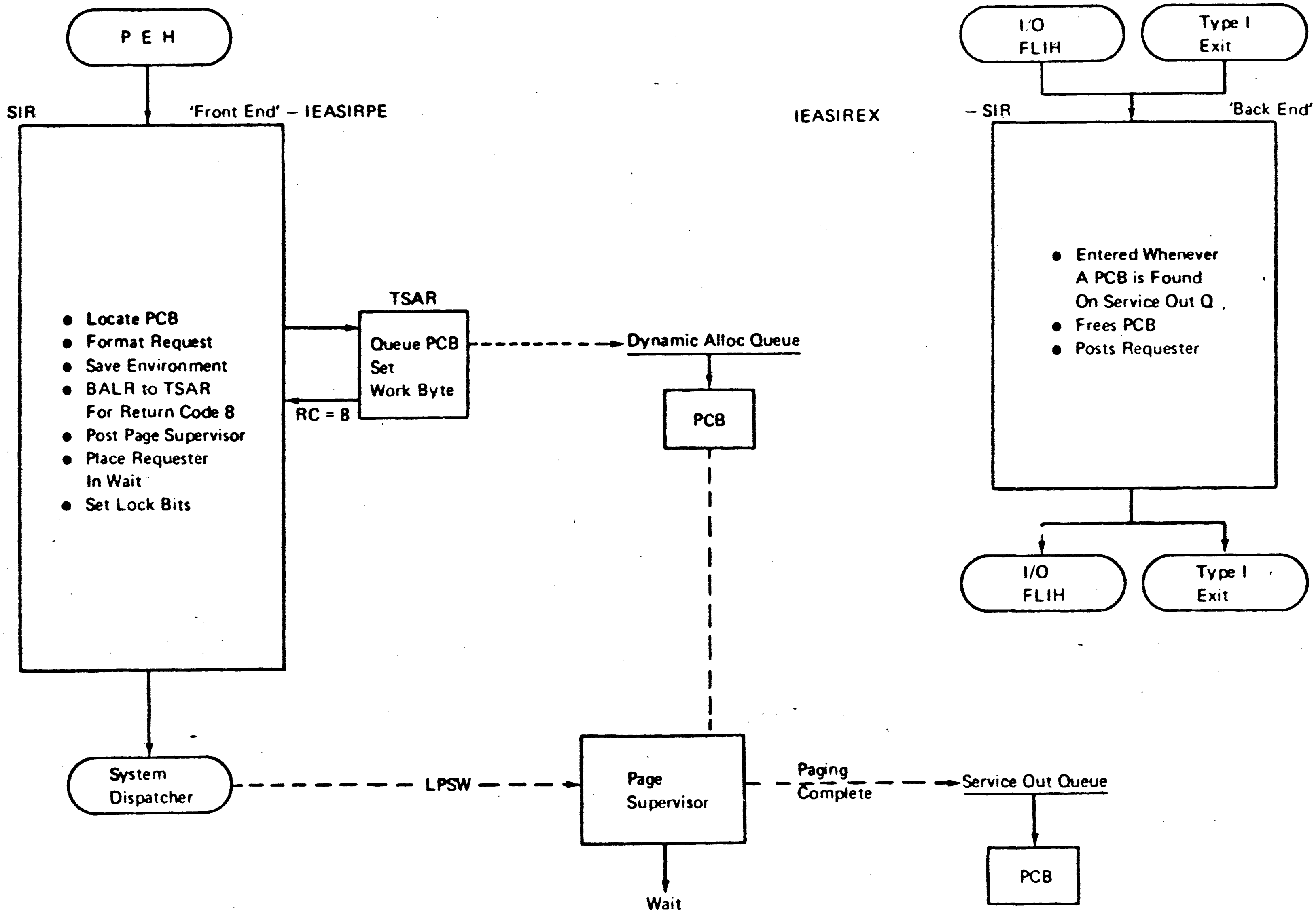
NOT HANDLED
POST QPPD
RETURN ON BRANCH OR SVC
GO TO DISPATCHER FOR
PAGE EXCEPTION

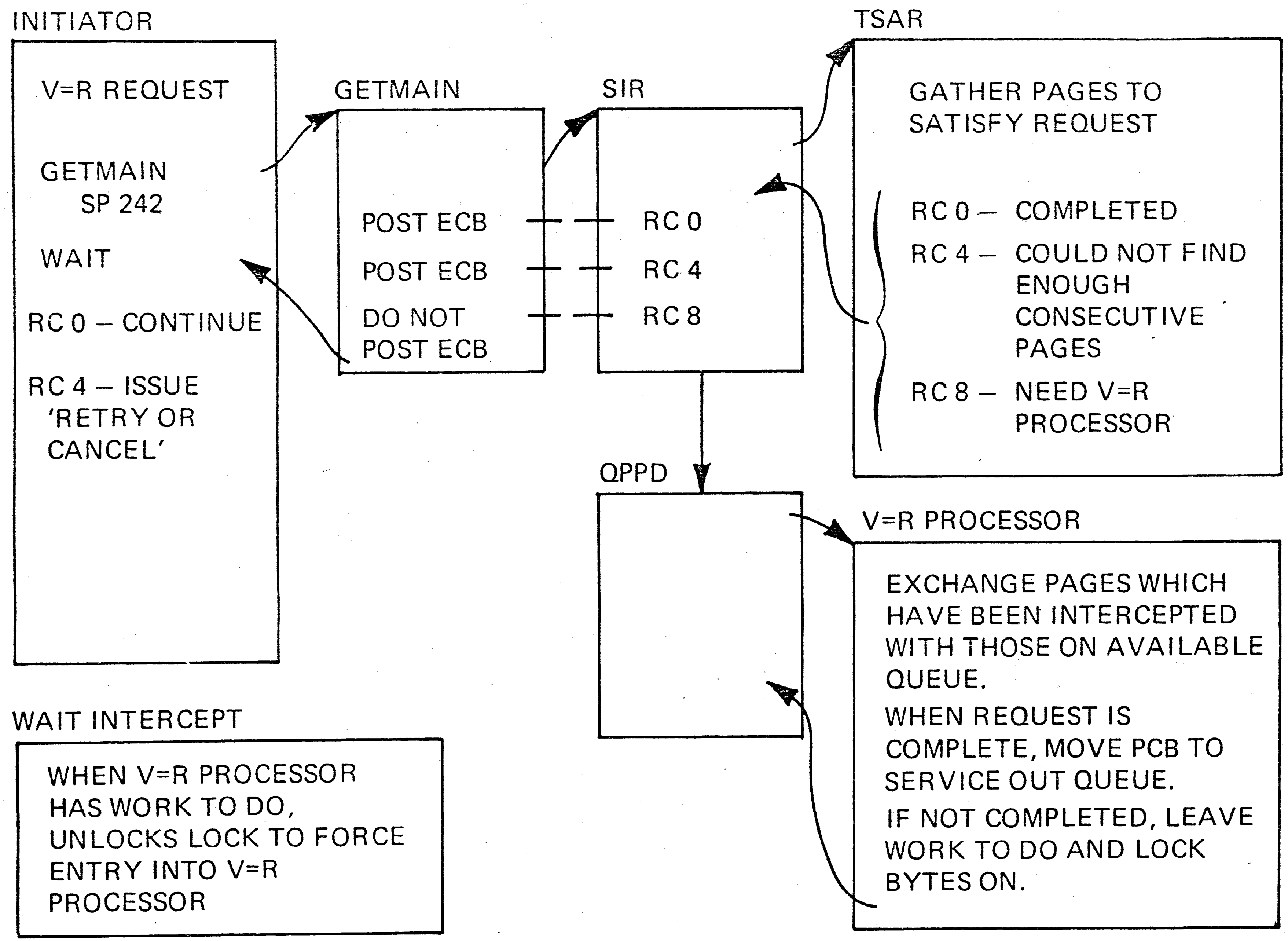


	WORK	LOCK
DYNAMIC ALLOCATION	FF	00
PREALLOCATION	00	00
PAGE I/O PROCESSOR	00	00
V = R ALLOCATION	00	00
PSEUDO PROCESSOR	FF	


QPPD
DYNAMIC ALLOC WORK TO DO
I/O PROCESSOR
DYNAMIC ALLOCATION
PAGE IN
REAL STOR. RELEASE
PAGE OUT
I/O QUEUE ACTIVATOR
SELECT PDDT – DEVICE
QUEUE RSPT ENTRY TO PAGE I/O DEVICE QUEUE
SET I/O PROCESSOR WORK TO DO
I/O PAGE PROCESSOR

Service Interface Routine 'SIR'





INITIATOR

V=R REQUEST

GETMAIN
SP 242

WAIT

RC 0 – CONTINUE

RC 4 – ISSUE
'RETRY OR
CANCEL'

GETMAIN

POST ECB

POST ECB

DO NOT
POST ECB

SIR

RC 0

RC 4

RC 8

TSAR

GATHER PAGES TO
SATISFY REQUEST

RC 0 – COMPLETED

RC 4 – COULD NOT FIND
ENOUGH
CONSECUTIVE
PAGES

RC 8 – NEED V=R
PROCESSOR

QPPD

V=R PROCESSOR

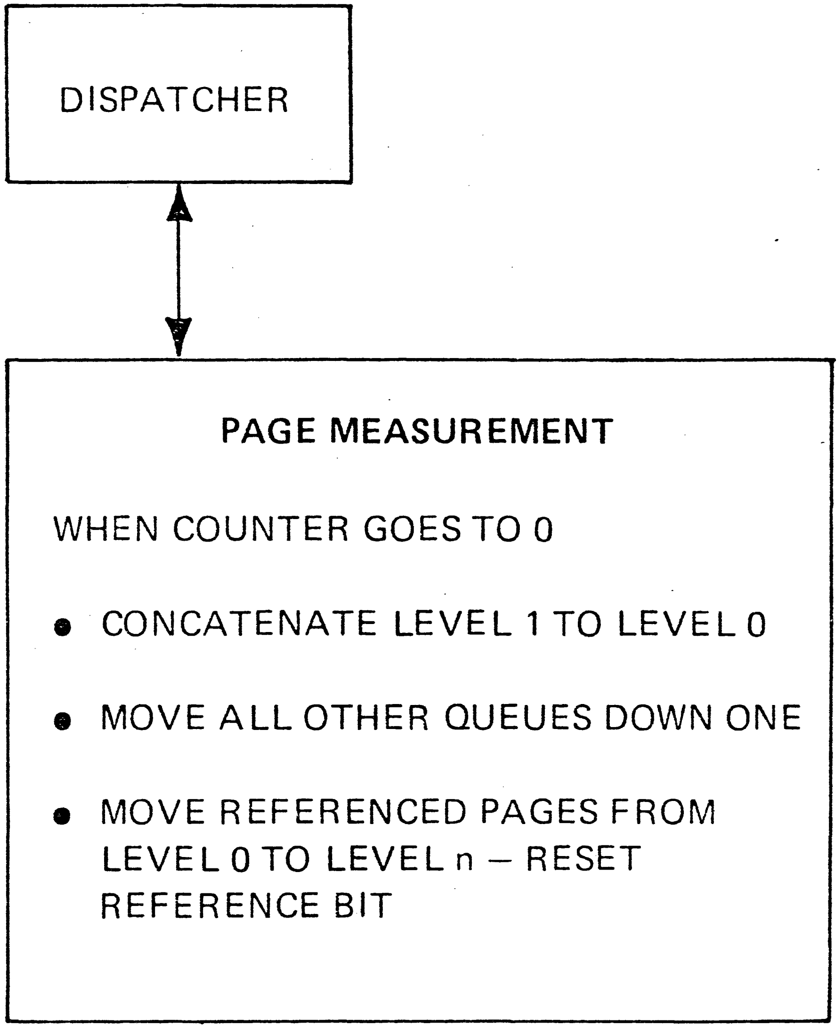
EXCHANGE PAGES WHICH
HAVE BEEN INTERCEPTED
WITH THOSE ON AVAILABLE
QUEUE.

WHEN REQUEST IS
COMPLETE, MOVE PCB TO
SERVICE OUT QUEUE.

IF NOT COMPLETED, LEAVE
WORK TO DO AND LOCK
BYTES ON.

WAIT INTERCEPT

WHEN V=R PROCESSOR
HAS WORK TO DO,
UNLOCKS LOCK TO FORCE
ENTRY INTO V=R
PROCESSOR



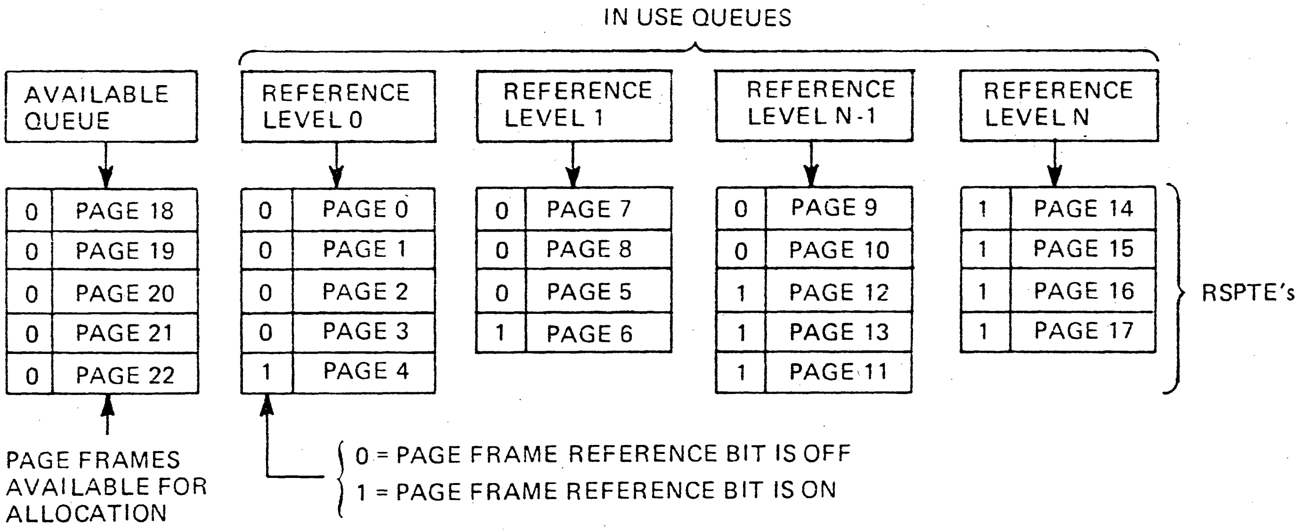
DISPATCHER

PAGE MEASUREMENT

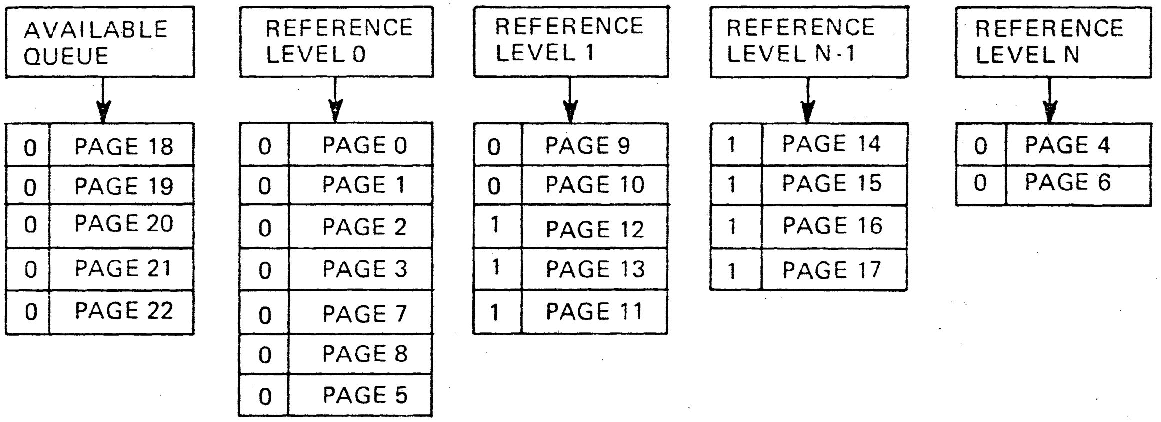
WHEN COUNTER GOES TO 0

- CONCATENATE LEVEL 1 TO LEVEL 0
- MOVE ALL OTHER QUEUES DOWN ONE
- MOVE REFERENCED PAGES FROM LEVEL 0 TO LEVEL n – RESET REFERENCE BIT

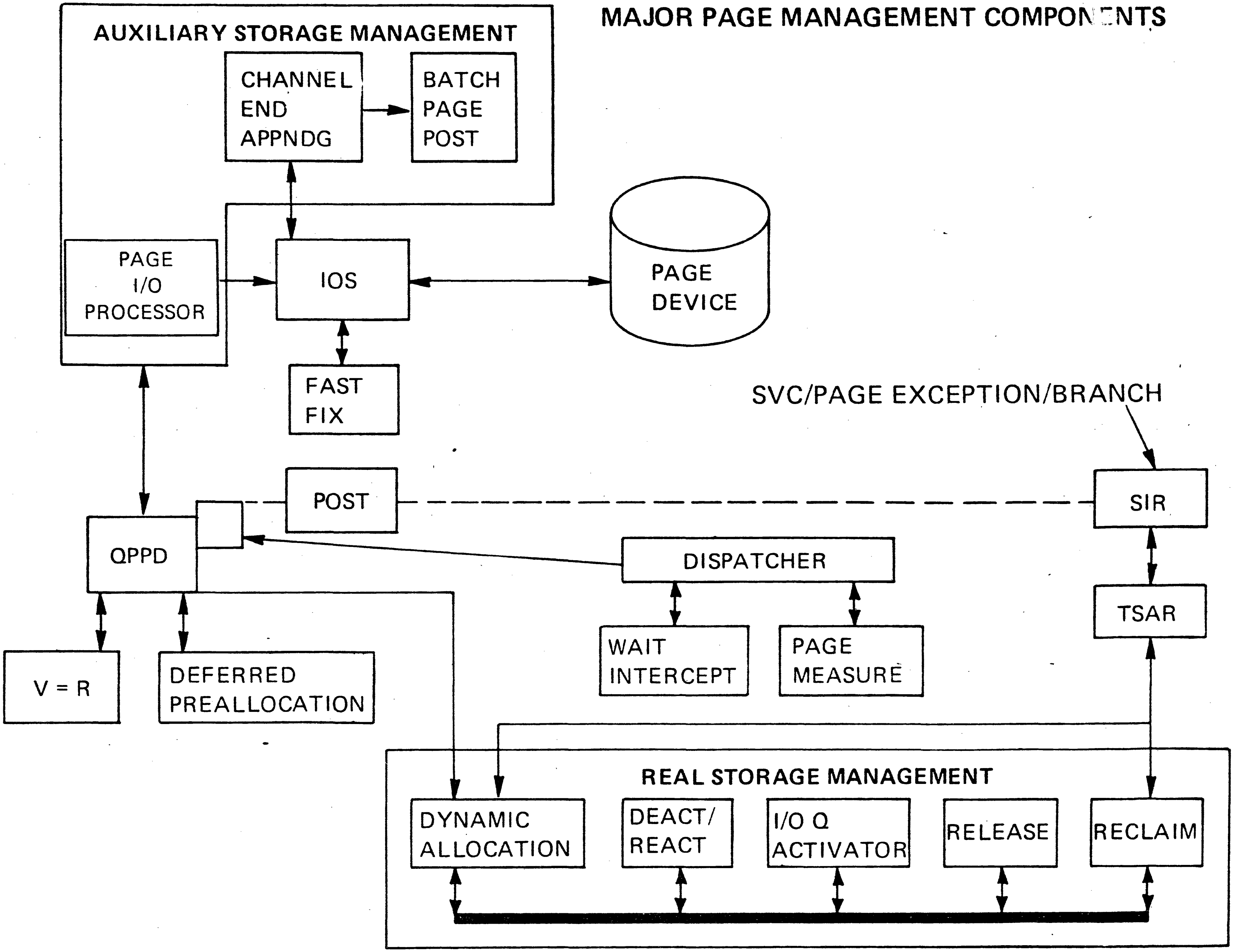
STATUS OF PAGE QUEUES AND PAGE FRAMES AT ACTIVITY MEASUREMENT TIME



STATUS OF PAGE QUEUES AND PAGE FRAMES AFTER ACTIVITY MEASUREMENT TIME



MAJOR PAGE MANAGEMENT COMPONENTS



SERVICE AIDS

SYSTEMS PROGRAMMING

Student Materials

This material was produced for Educational purposes only. Changes may obsolete part or all of this publication. No responsibility is assumed for any inaccuracies that occur.

Los Angeles Advanced Education Center
3424 WILSHIRE BOULEVARD • LOS ANGELES, CALIF. 90010

SERVICE AIDS...VS/1

REFERENCES

OS/VS Service Aids	GC28-0633
OS/VS Service Aids PLM	SY24-0635
OS/VS SYS1.Logrec	GC28-0638

SERVICE AIDS

A – INFORMATION GATHERING

- SADMP
- GTF

B – FORMATTING AND PRINTING

- IFCEREP0
- HMBLIST
- JOBQD
- PRDMP

C – MODIFICATION

- PTFLE
- HMASMP
- IFCDIP00

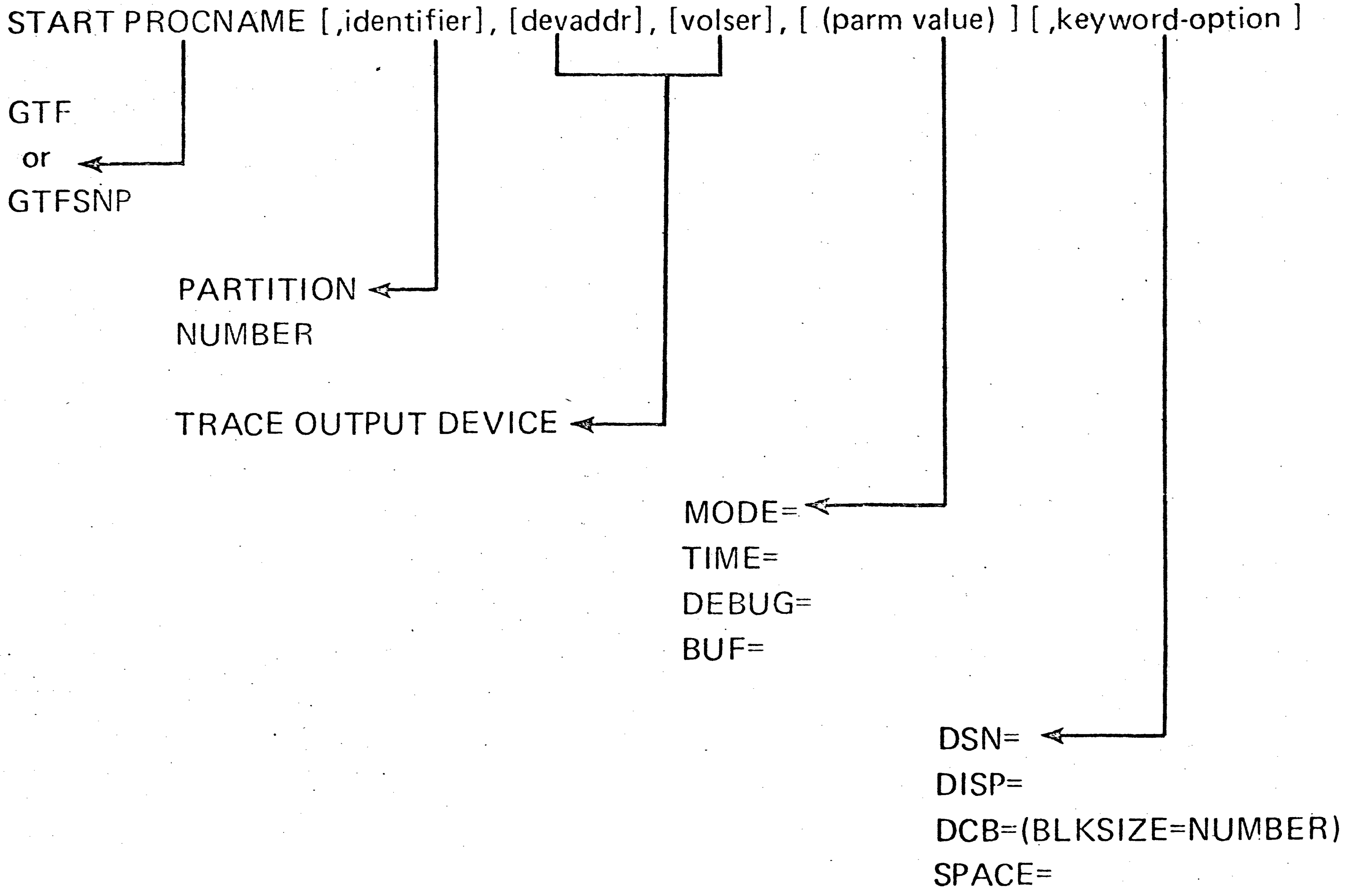
EVENTS TRACED BY GTF

- INPUT/OUTPUT INTERRUPTIONS (IO)
- START I/O OPERATIONS (SIO)
- SUPERVISOR CALL INTERRUPTIONS (SVC)
- PROGRAM INTERRUPTIONS (PI)
- EXTERNAL INTERRUPTIONS (EXT)
- DISPATCHER TASK SWITCH OPERATIONS (DSP)
- USER EVENTS (USR)
- GTF EVENTS (TRC)

SELECTIVE TRACING

- SPECIFIC EVENT TYPES CAN BE SPECIFIED
- FOR SIO AND I/O
    - SPECIFY DEVICES
- FOR SVC
    - SPECIFY SVC'S
- FOR PI
    - SPECIFY INTERRUPTS

STARTING GTF



START PROCNAME [,identifier], [devaddr], [volser], [(parm value)] [,keyword-option]

GTF
or
GTFSNP

PARTITION
NUMBER

TRACE OUTPUT DEVICE

MODE=
TIME=
DEBUG=
BUF=

DSN=
DISP=
DCB=(BLKSIZE=NUMBER)
SPACE=

PROMPTING FOR GTF TRACE OPTIONS

HH100A SPECIFY TRACE OPTIONS

TRACE = option 1[,option 2] . . . . [,option n]

- 3 MODES FOR TRACING FIVE TYPES OF EVENTS
  (I/O, SVC, PI, EXT, SIO)

- SYS - COMPREHENSIVE RECORDS

- SYSM - MINIMAL TRACE DATA

- SYSP - ADDITIONAL PROMPTING FOR DEVICE TYPE, SVC NUMBER, OR PROGRAM INTERRUPT NUMBER

SVC TRACE RECORD FORMAT

MINIMUM:

ID,OLD PSW,R15,R0,R1,OLD TCB

COMPREHENSIVE

ID,OLD PSW,JOBNAME,MODULENAME,OLDTCB,R15,R0,R1
plus SVC dependent Information

GTF PROCEDURE

```
 //GTF          PROC
 //IEFPROC      EXEC       PGM=HHLGTF,
 //             PARM='MODE=EXT,DEBUG=NO,TIME=NO'
 //IEFRDER      DD         DSN=SCRATCH,UNIT=2400,
 //             VOL=SER=TRACE,LABEL=(,NL),DISP=(NEW,KEEP)
 //SYSPRINT     DD         SYSOUT=A
[ //SYSLIB      DD         DSN=SYS1.PARMLIB (membername),]
[ //            DISP=SHR]
```

GTRACE MACRO

-- EFFECTIVE WHEN:

- -- GTF IS ACTIVE
- -- MODE = EXT
- -- TRACE = USR

```
-- [SYMBOL]     GTRACE     DATA=ADDRESS, LNG=number,
                ID = number [ ,FID = value ]
```

FUNCTIONS OF PRDMP.........

- FORMAT/PRINT SADMP OUTPUT
- FORMAT/PRINT SYS1.DUMP DATA SET
- FORMAT/PRINT GTF TRACE DATA

```
//PRDMP      JOB    ,Dump,MSG Level=1
//PRINT      EXEC   PGM=HMDPRDMP
//ANYNAME    DD     Defines Input Data Set
//SYSPRINT   DD     Defines Message Data Set
//PRINTER    DD     Defines Output Data Set
//SYSUT1     DD     Optional work space         \  mutually
//SYSUT2     DD     Saves Data for later proc.  /  exclusive
//SYSIN      DD     Defines Control Statements
```

FORMAT CONTROL STATEMENTS

QCBTRACE
LPAMAP
FORMAT
PRINT
EDIT

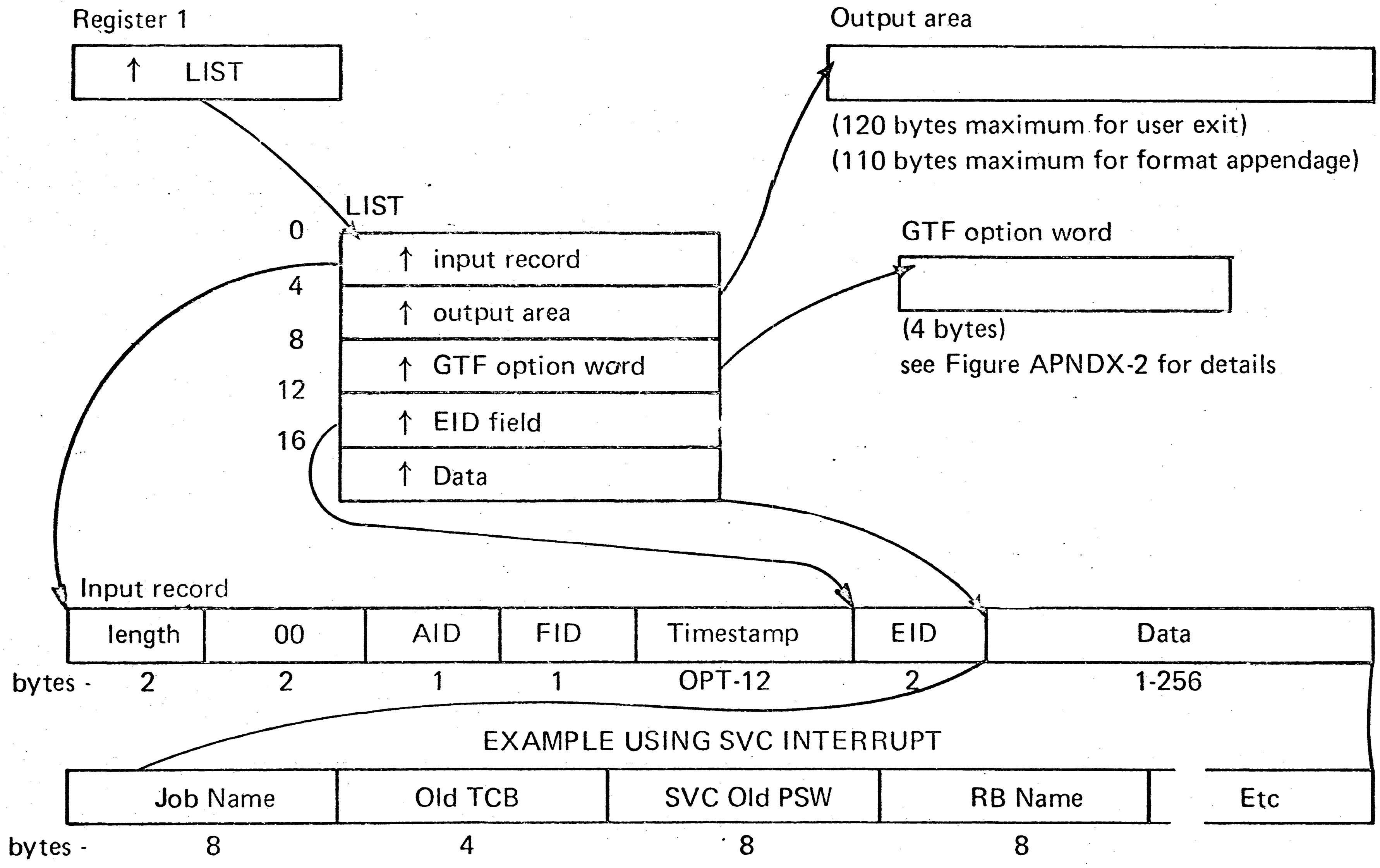
FUNCTION CONTROL STATEMENTS

CVT
SEGTAB
NEWDUMP
GO
ONGO
TITLE
END

EDIT GTF

EXIT	—	*Name*
START	—	*Time*
STOP	—	*Time*
JOBNAME	—	*Names*
TCB	—	*Addresses*
SYS	—	*SVC, SIO, IO, PI, EXT, DSP*
IO:SIO:SIO=IO	—	*All or selected devices*
SVC	—	*All or selected SVCs*
PI	—	*All or selected PI codes*
EXT	—	*All*
DSP	—	*All*
USR	—	*All or selected IDs*

PRDMP EXIT



(120 bytes maximum for user exit)
(110 bytes maximum for format appendage)

(4 bytes)
see Figure APNDX-2 for details

	LIST
0	↑ input record
4	↑ output area
8	↑ GTF option word
12	↑ EID field
16	↑ Data

Input record

	length	00	AID	FID	Timestamp	EID	Data
bytes -	2	2	1	1	OPT-12	2	1-256

EXAMPLE USING SVC INTERRUPT

	Job Name	Old TCB	SVC Old PSW	RB Name	Etc
bytes -	8	4	8	8	

FUNCTIONS OF SPZAP.........

- VERIFY AND/OR REPLACE DATA IN A RECORD ON DASD.
- VERIFY AND/OR REPLACE INSTRUCTIONS IN A LOAD MODULE.
- DUMP A DATA SET OR A LOAD MODULE.
- UPDATE SSI INFORMATION IN THE DIRECTORY OF A PDS.

```
//SPZAP        JOB
//ZAP          EXEC      PGM=HMASPZAP
//SYSPRINT     DD        Defines Message Data Set
//SYSLIB       DD        Defines Input Data Set
//SYSIN        DD        Defines Control Statements
```

SPZAP CONTROL STATEMENTS......

```
 NAME         MEMBER       CSECT

 CCHHR

 VERIFY       OFFSET       CONTENT

 REPLACE      OFFSET       CONTENT

 SETSSI       XXYYNNNN

{DUMP    }    MEMBER      {CSECT}
{DUMPT   }                {ALL  }

{ABSDUMP }   {ST.ADDR}     STOPADDR
{ABSDUMPT}   {MEMBER }
             {ALL    }

 CONSOLE
```

SPZAP AND THE VTOC RECORDS........

```
//SYSLIB  DD  DSN=FORMAT4.DSCB,DISP=OLD,UNIT=....,
//            VOL=SER=.......,DCB=(KEYLEN=44)
```

DATASET SECURITY

SYSTEMS PROGRAMMING

Student Materials

This material was produced for Educational purposes only. Changes may obsolete part or all of this publication. No responsibility is assumed for any inaccuracies that occur.

Los Angeles Advanced Education Center
3424 WILSHIRE BOULEVARD • LOS ANGELES, CALIF. 90010

DATA SET SECURITY

REFERENCES

OS/VS Data Management Services	GC26-3783
OS/VS Data Management for System Prog.	GC28-0631
OS/VS JCL Reference	GC28-0618
OS/VS Utilities	GC35-0005
OS/VS Dadsm PLM	SY26-3787
OS/VS VSAM Programmer's Guide	GC26-3818
OS/VS VSAM Access Method Services	GC35-0009

TYPES OF PROTECTION

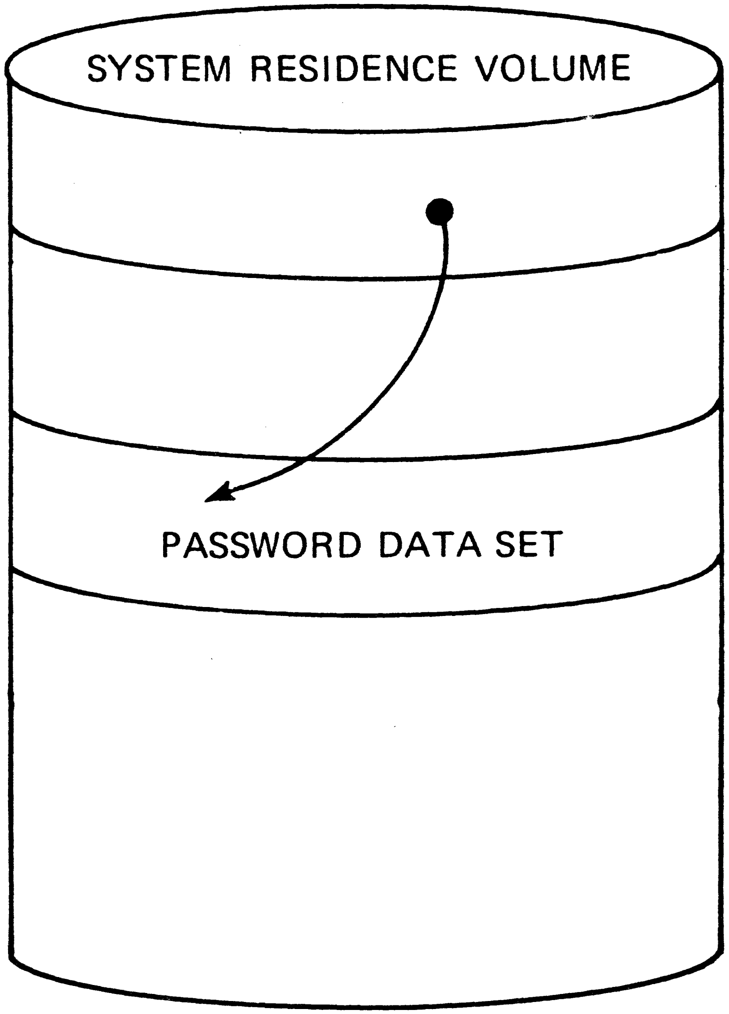
1 PASSWORD REQUIRED TO READ OR WRITE
(This password allows either)

2 PASSWORD REQUIRED TO READ OR WRITE
(This password allows read only)

3 NO PASSWORD REQUIRED TO READ
PASSWORD REQUIRED TO WRITE
(This password allows write)

DSCB PROTECTION BITS

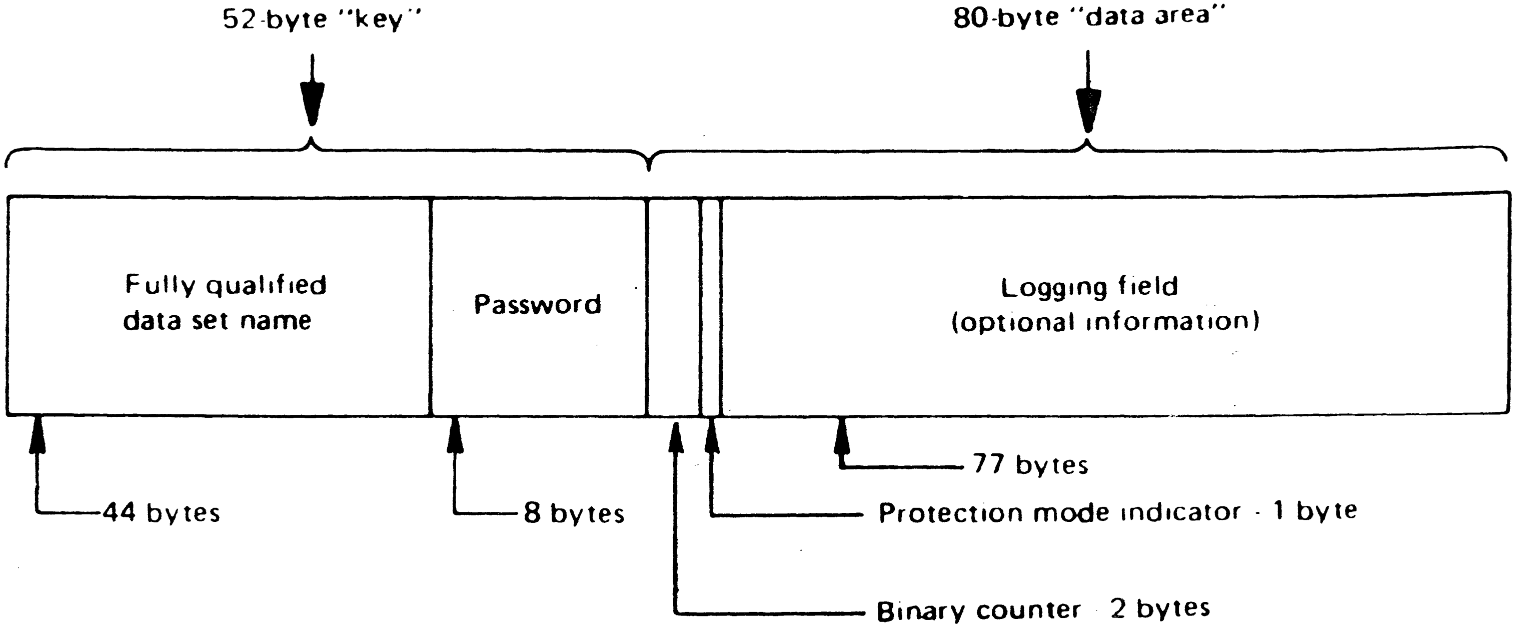
DSCB +93 DS1DSIND field	. . . X . X . .
LABEL= , , PASSWORD	. . . 1 . 0 . .
LABEL= , , NOPWREAD	. . . 1 . 1 . .


SYSTEM RESIDENCE VOLUME
PASSWORD DATA SET

DCB AND DD CODING FOR PASSWORD DATA SET

```
      DCB  DDNAME=DD1,DSORG=PS,LRECL=80,      X
           BLKSIZE=80,KEYLEN=52,RECFM=F,      X


//DD1  DD  DSNAME=PASSWORD,DISP=(NEW,KEEP),
//         VOLUME=SER= sysres volume,
//         SPACE= (one extent large enough to hold all entries)
```

Data Set Record Format


52-byte "key"
80-byte "data area"
Fully qualified data set name
Password
Logging field (optional information)
44 bytes
8 bytes
77 bytes
Protection mode indicator · 1 byte
Binary counter 2 bytes

DSCB PROTECT BIT SETTINGS

. . . 1	. 0 . .	PASSWORD REQUIRED TO READ OR WRITE
. . . 1	. 1 . .	PASSWORD REQUIRED TO WRITE

MODE INDICATOR BIT SETTINGS

X'80'	CONTROL PASSWORD FOR READ
X'81'	CONTROL PASSWORD FOR READ AND WRITE

TYPE	DATA SET PROTECTION	PASSWORD ALLOWS	DSCB	MODE INDICATOR
1	READ/WRITE	READ AND WRITE		
2	READ/NO WRITE	READ ONLY		
3	WRITE	WRITE		

ADD FUNCTION

```
ADD       DSNAME=data set name
          PASWORD2=new password
          CPASWORD=control password
          TYPE=  1
                 2
                 3
          VOL=device=list
          DATA='user data'
```

REPLACE

```
REPLACE     DSNAME=data set name
            PASWORD1=current password
            PASWORD2=new password
            CPASWORD=control password
            TYPE=  1
                   2
                   3
            VOL=device=list
```

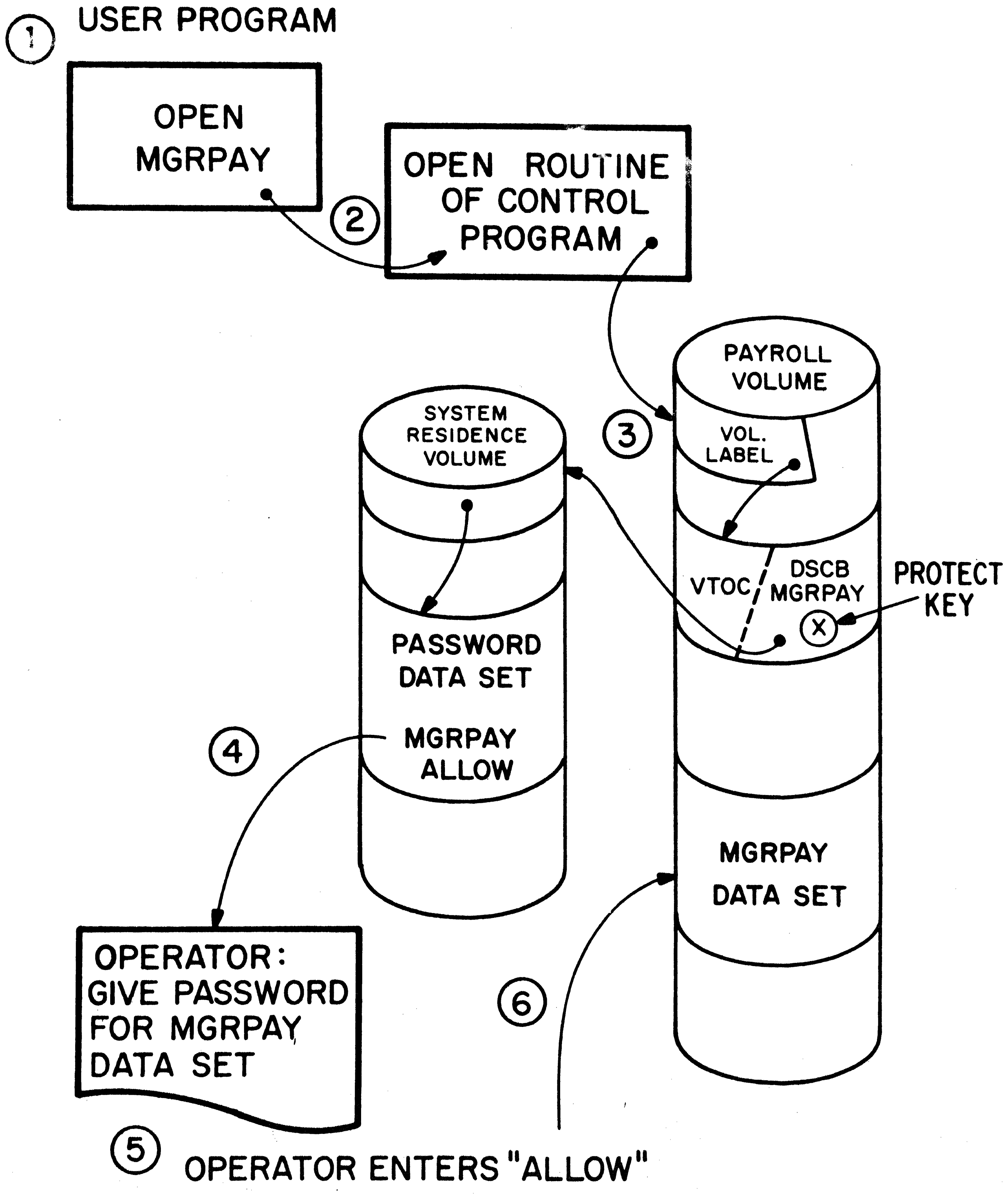
DELETE

```
DELETEP    DSNAME=data set name
           PASWORD1=current password
           CPASWORD=control password
           VOL=device=list
```

LIST

```
LIST          DSNAME=data set name
              PASWORD1=current password
```

PASSWORD PROTECTION



FOUR LEVELS OF PROTECTION

MASTER LEVEL
CONTROL LEVEL
UPDATE LEVEL
READ LEVEL

ENTRY HIERARCHY

CLUSTER - highest level
DATA/INDEX - same level

ONLY READ LEVEL PASSWORD SPECIFIED

```
READPW(ALLOW)
```

RESULTS IN:

MASTER PASSWORD	ALLOW
CONTROL PASSWORD	ALLOW
UPDATE PASSWORD	ALLOW
READ PASSWORD	ALLOW

READ AND CONTROL PASSWORD SPECIFIED

```
 READPW(ALLOW)
CONTROLPW(ALLOWIT)
```

RESULTS:

MASTER PASSWORD	ALLOWIT
CONTROL PASSWORD	ALLOWIT
UPDATE PASSWORD	not specified
READ PASSWORD	ALLOW

AUTHORIZATION PARAMETER

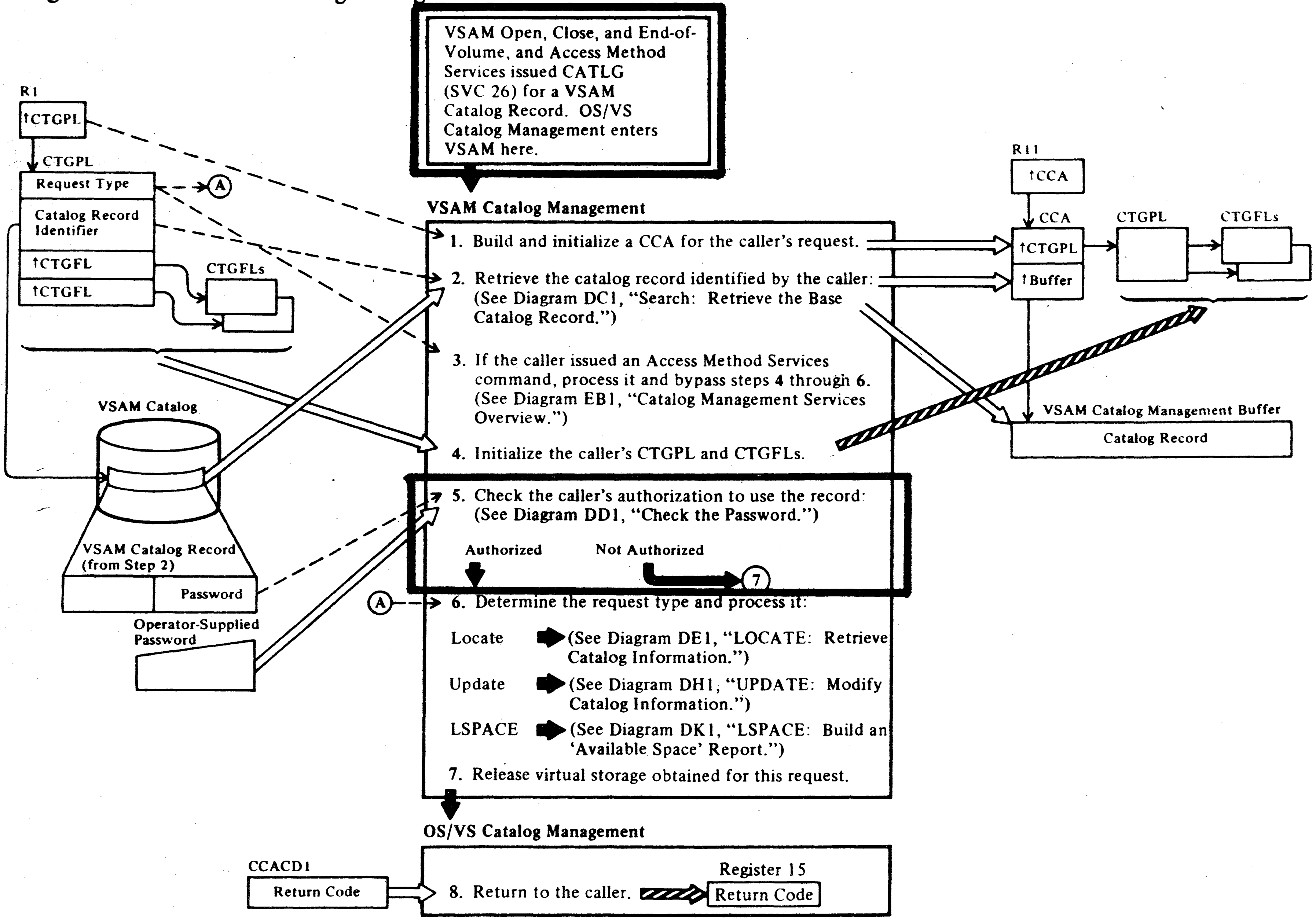
AUTHORIZATION(entrypoint string)

entrypoint is the entry point of your routine on SYS1.LINKLIB

string is your own security information up to 256 bytes

abbr. AUTH(entrypoint string)

Diagram DB1. VSAM Catalog Management Overview



VSAM Open, Close, and End-of-Volume, and Access Method Services issued CATLG (SVC 26) for a VSAM Catalog Record. OS/VS Catalog Management enters VSAM here.

VSAM Catalog Management

1. Build and initialize a CCA for the caller's request.
2. Retrieve the catalog record identified by the caller: (See Diagram DC1, "Search: Retrieve the Base Catalog Record.")
3. If the caller issued an Access Method Services command, process it and bypass steps 4 through 6. (See Diagram EB1, "Catalog Management Services Overview.")
4. Initialize the caller's CTGPL and CTGFLs.
5. Check the caller's authorization to use the record: (See Diagram DD1, "Check the Password.")

   Authorized — Not Authorized → 7

6. Determine the request type and process it:

   Locate (See Diagram DE1, "LOCATE: Retrieve Catalog Information.")

   Update (See Diagram DH1, "UPDATE: Modify Catalog Information.")

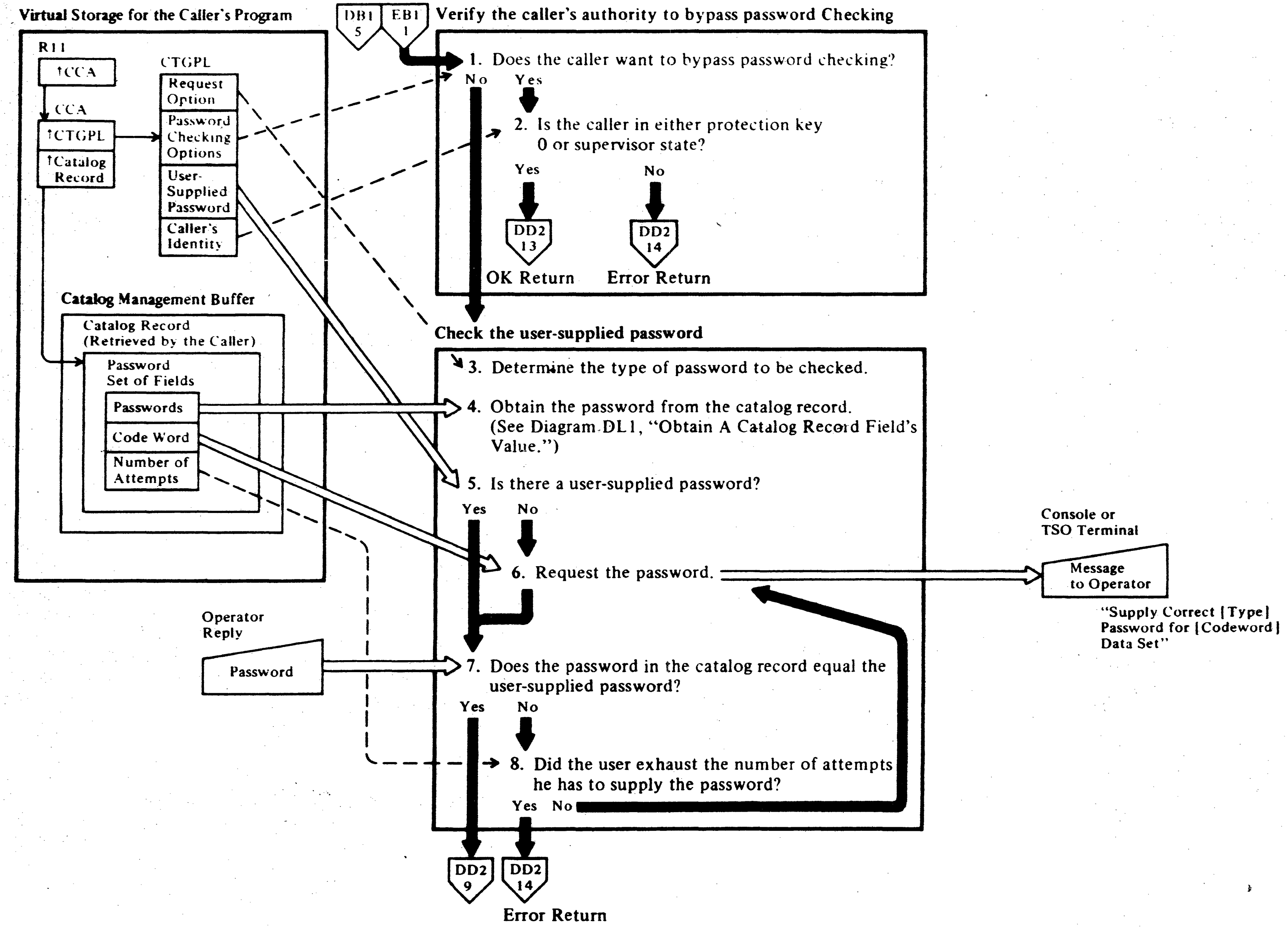
   LSPACE (See Diagram DK1, "LSPACE: Build an 'Available Space' Report.")

7. Release virtual storage obtained for this request.

OS/VS Catalog Management

8. Return to the caller.

Diagram DD1. Check the Password



Verify the caller's authority to bypass password Checking

1. Does the caller want to bypass password checking?
2. Is the caller in either protection key 0 or supervisor state?

OK Return

Error Return

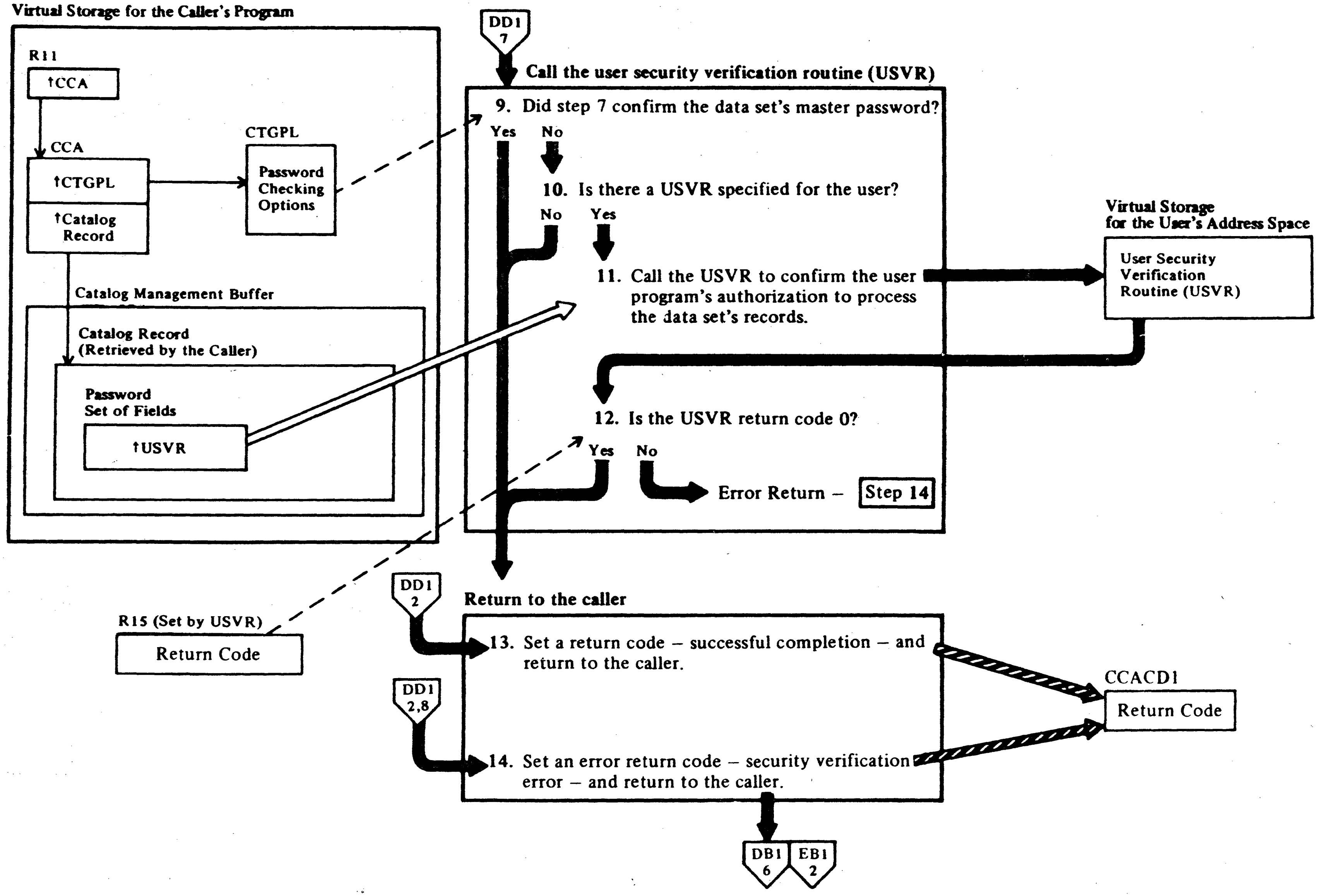
Check the user-supplied password

3. Determine the type of password to be checked.
4. Obtain the password from the catalog record. (See Diagram DL1, "Obtain A Catalog Record Field's Value.")
5. Is there a user-supplied password?
6. Request the password.
7. Does the password in the catalog record equal the user-supplied password?
8. Did the user exhaust the number of attempts he has to supply the password?

Error Return

"Supply Correct [Type] Password for [Codeword] Data Set"

Diagram DD2. Check the Password


Virtual Storage for the Caller's Program
R11
↑CCA
CCA
↑CTGPL
↑Catalog Record
CTGPL
Password Checking Options
Catalog Management Buffer
Catalog Record (Retrieved by the Caller)
Password Set of Fields
↑USVR
R15 (Set by USVR)
Return Code
DD1 7
Yes
No
Yes
No
Yes
No
Error Return – Step 14
Virtual Storage for the User's Address Space
User Security Verification Routine (USVR)
DD1 2
DD1 2,8
CCACD1
Return Code
DB1 6
EB1 2

Call the user security verification routine (USVR)

9. Did step 7 confirm the data set's master password?

10. Is there a USVR specified for the user?

11. Call the USVR to confirm the user program's authorization to process the data set's records.

12. Is the USVR return code 0?

Return to the caller

13. Set a return code – successful completion – and return to the caller.

14. Set an error return code – security verification error – and return to the caller.

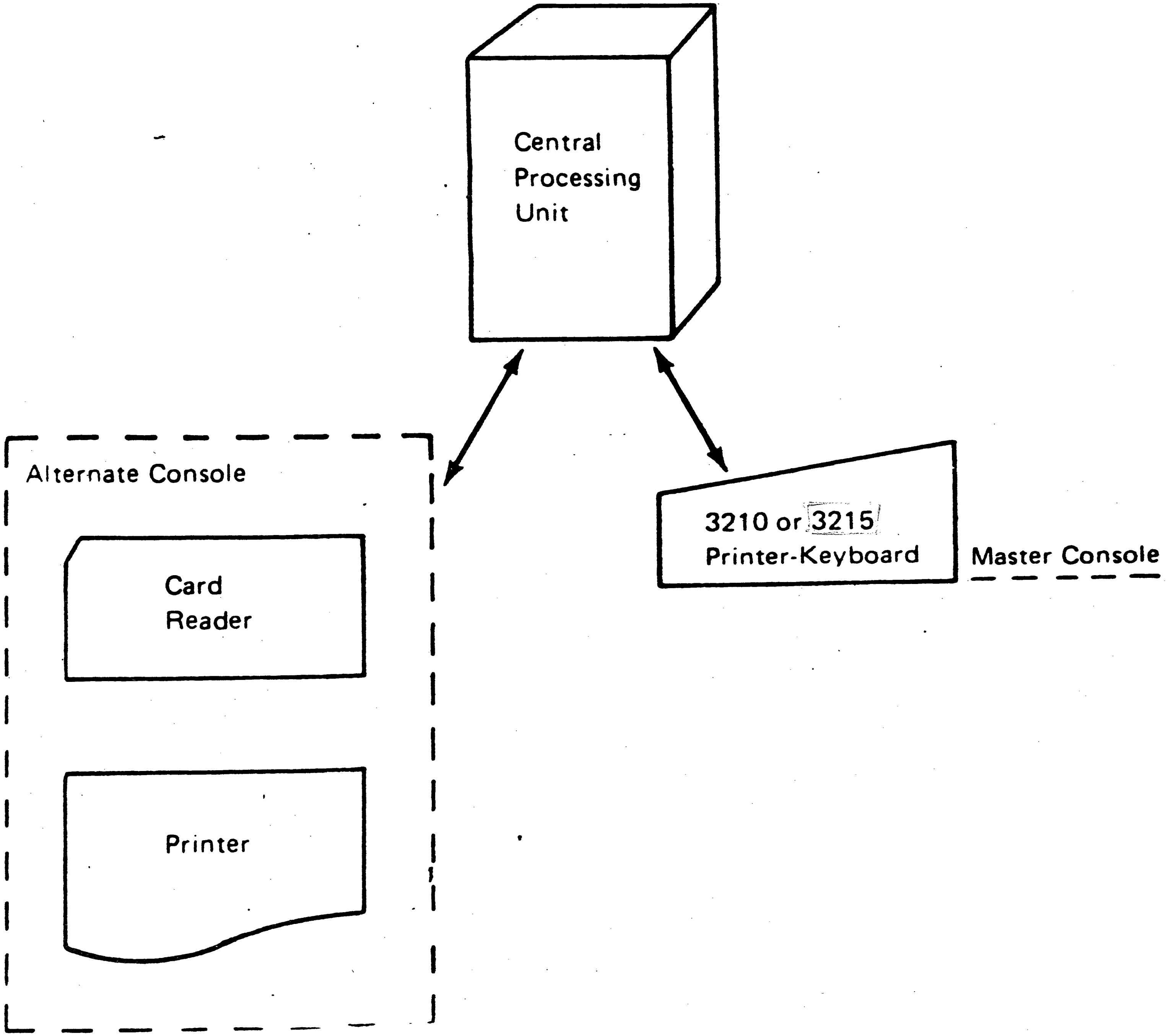
SYSTEM CONSOLES

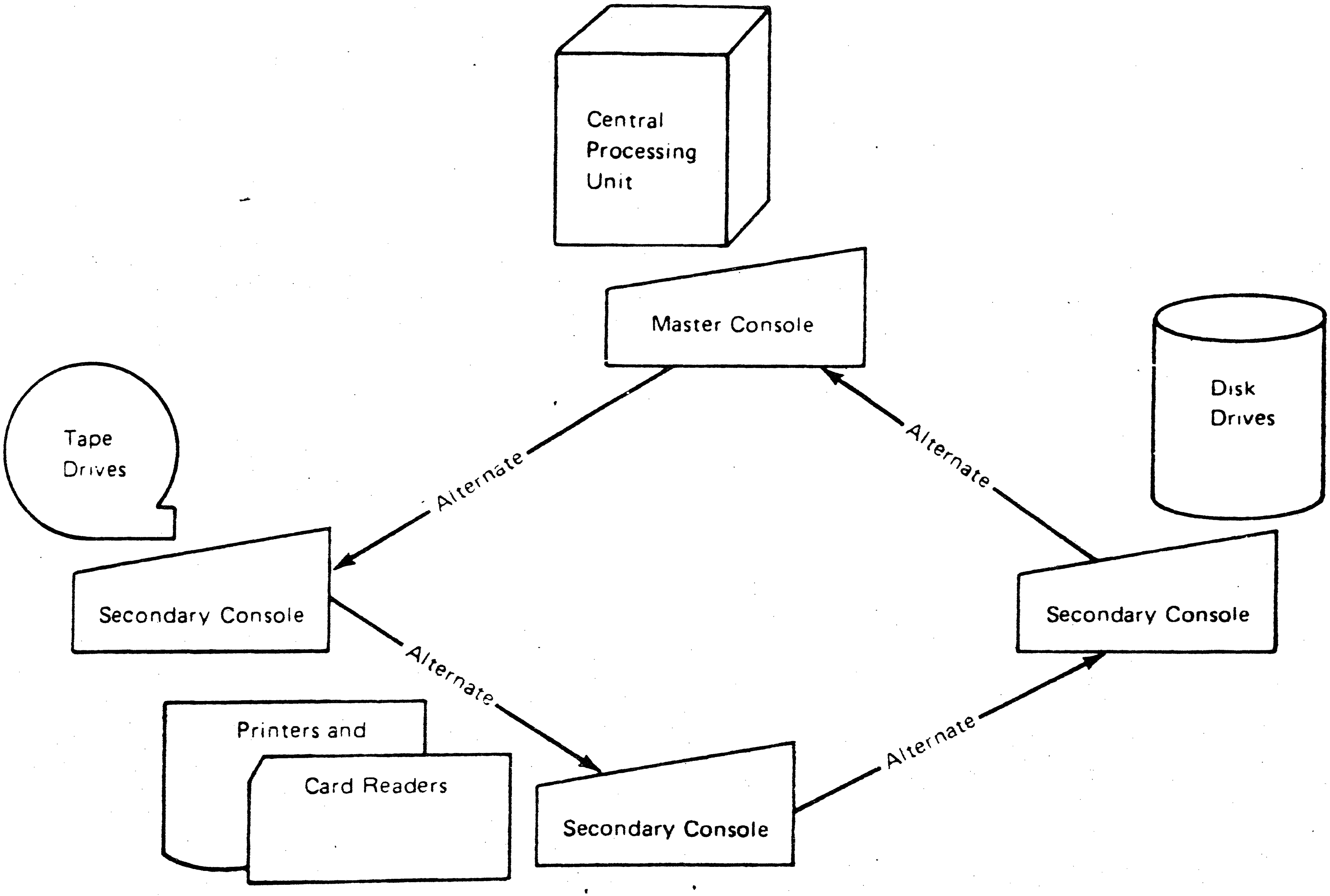
TWO BASIC CONFIGURATIONS

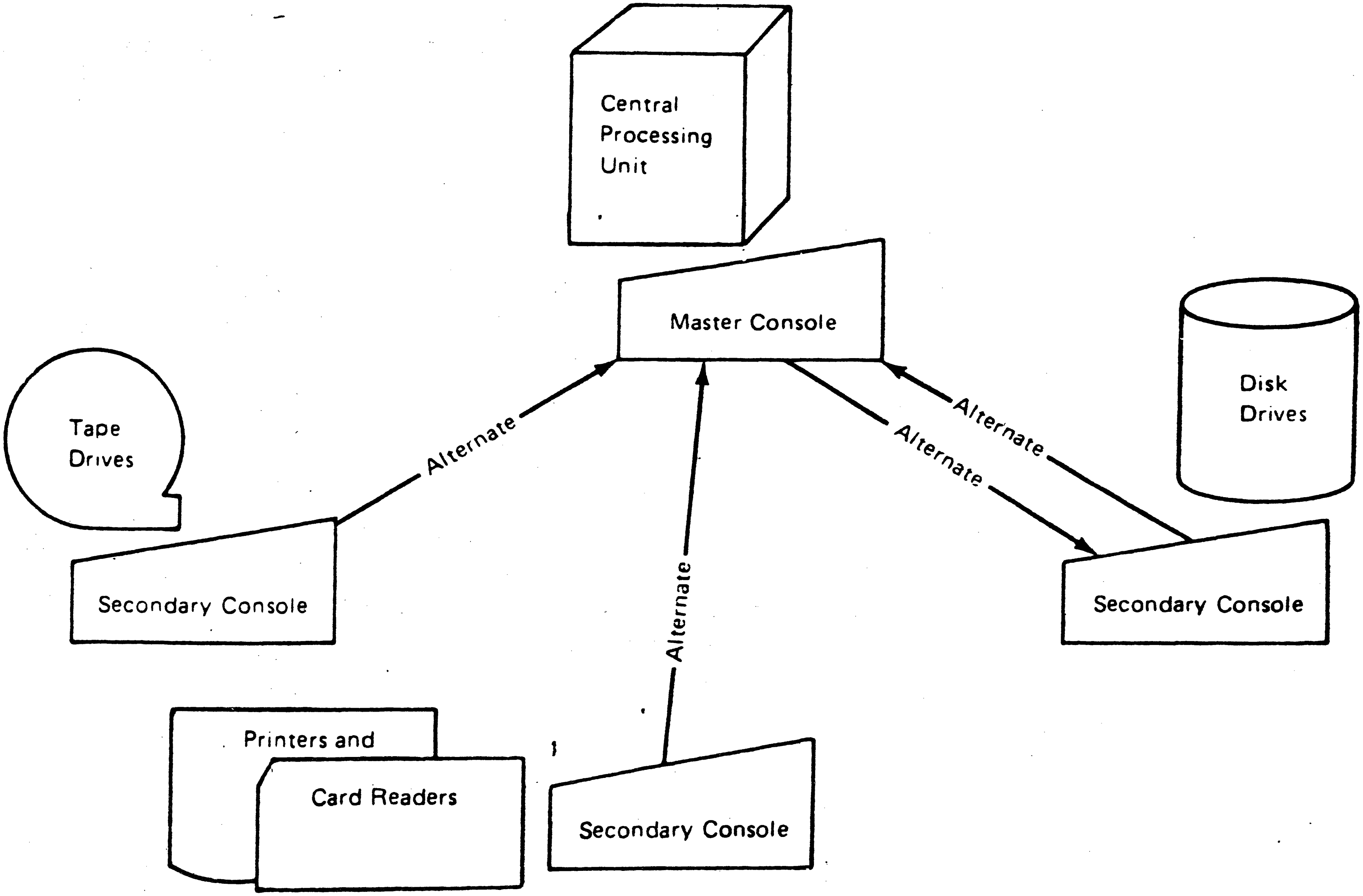
- SINGLE CONSOLE CONFIGURATION
- MULTIPLE CONSOLE CONFIGURATION

SYSTEM I/O DEVICES

- ONLINE
- OFFLINE
- CONSOLE


Central Processing Unit
Alternate Console
Card Reader
Printer
3210 or 3215 Printer-Keyboard
Master Console


Central Processing Unit
Master Console
Disk Drives
Tape Drives
Alternate
Alternate
Secondary Console
Secondary Console
Alternate
Alternate
Printers and
Card Readers
Secondary Console
Alternate


Central Processing Unit
Master Console
Disk Drives
Tape Drives
Alternate
Alternate
Alternate
Alternate
Secondary Console
Secondary Console
Printers and
Card Readers
Secondary Console

SPECIFICATION OF CONSOLES

SCHEDULER	ALTCONS -	{ address I - address, O - address }
	AREA -	(nn,nn)
	CONSOLE -	{ address I - address, O - address }
	HARDCPY -	
	OLDWTOR -	routing codes
	PFK -	number
	ROUTCDE -	routing code

SPECIFYING SECONDARY CONSOLES

SECONSLE

ALTCONS -

AREA -

CONSOLE -

PFK -

ROUTCDE -

VALDCMD -

PROGRAMMER COMMUNICATION

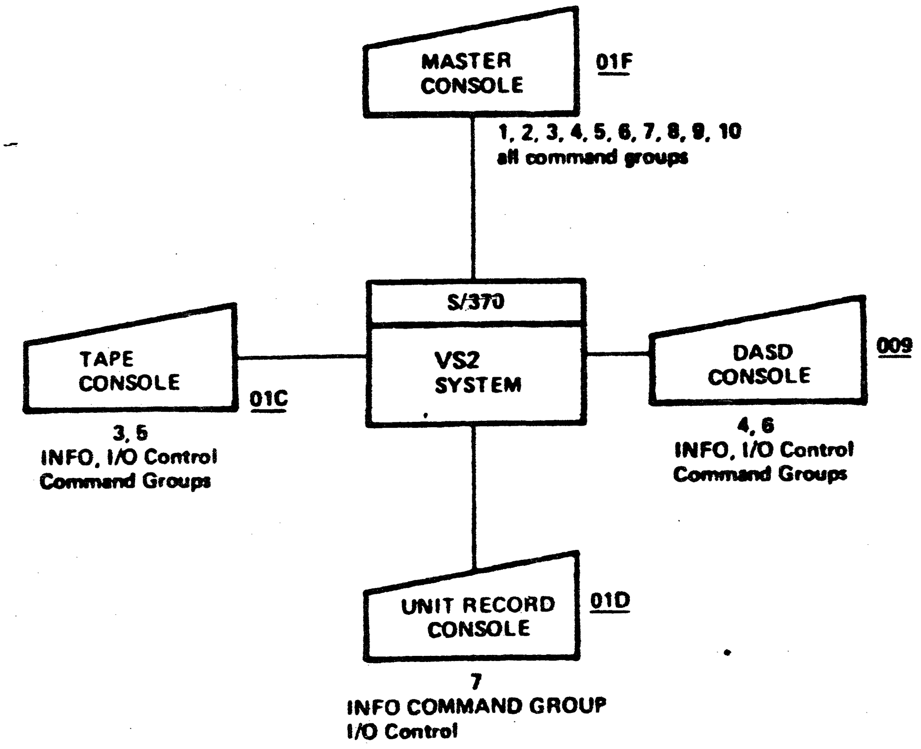
MACRO'S

- WTO
- WTOR
- WTL

COMMAND GROUPS

- INFORMATIONAL COMMANDS (INFO)
- SYSTEM CONTROL COMMANDS (SYS)
- I/O CONTROL COMMANDS (IO)
- CONSOLE CONTROL COMMANDS (CONS)

A SIMPLE MULTIPLE CONSOLE CONFIGURATION



Command Groups

Message Routing Codes

1 Master console action
2 Master console information
3 Tape area
4 DASD area
5 Tape library
6 DASD library
7 Unit Record Area
8 Teleprocessing equipment status
9 System Security
10 System Error Maintenance
11 Sysout device

INFO

DISPLAY
LOG
REPLY
CONTROL'

MSGRT
SEND
MONITOR
STOPMN

I/O Control

MOUNT
UNLOAD
VARY*
SWAP

System Control

CANCEL
DUMP
HALT
HOLD
MODE
MODIFY
RELEASE
SET
RESET
START
STOP
WRITELOG
SWITCH

SPECIFYING A USER EXIT

SCHEDULR   OPTIONS = EXIT

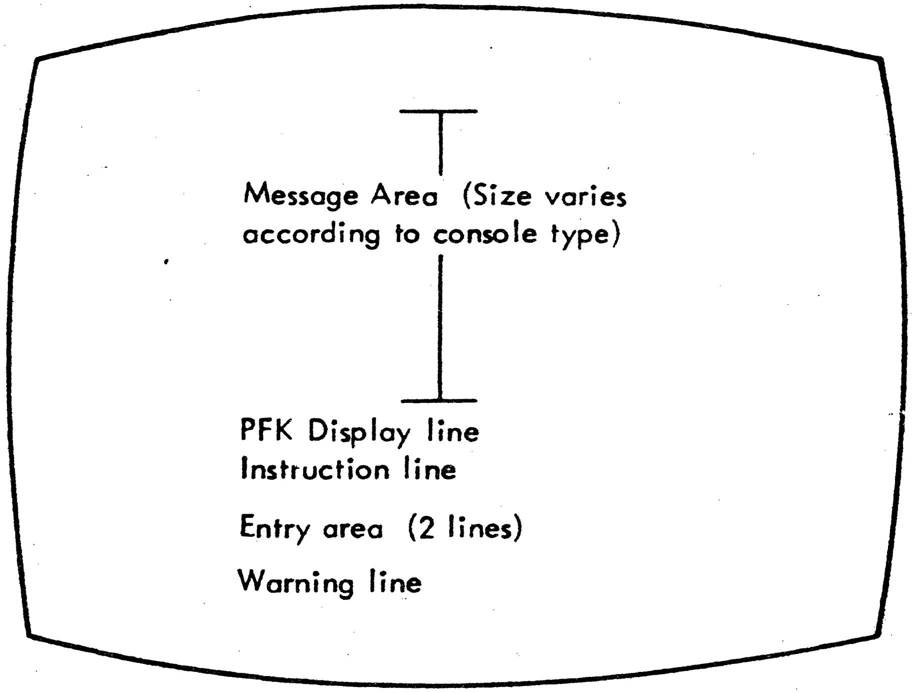
BEFORE SYSTEM GENERATION

REPLACE DUMMY WTO ROUTINE IEECVXIT IN SYS1.AOSB

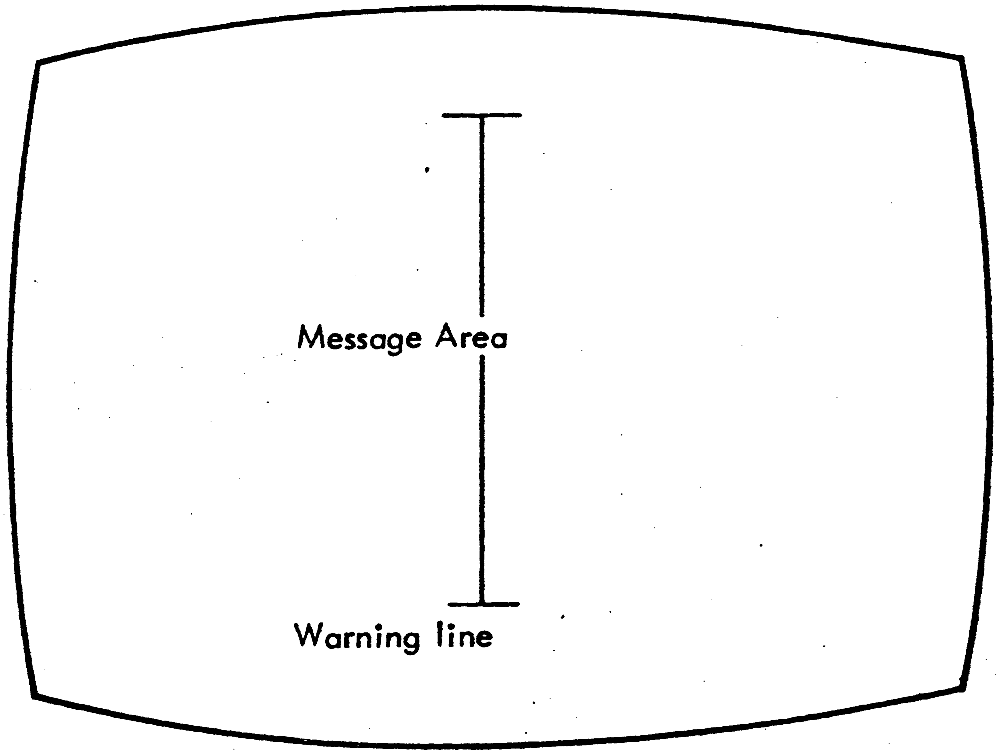
AFTER SYSTEM GENERATION

REPLACE EXIT ROUTINE IEECVCTE in SYS1.NUCLEUS

THEN RE-IPL



Full-capability display console screen



Output-only display console screen

	Full-capability Mode	Output-Only Mode	Light Pen*	Audible Alarm*	Program Function* Keyboard (PFK)	Visual Alarm
2250	YES	NO	YES	YES	YES	NO
2260	YES	YES	NO	NO	NO	NO
Model 168	YES	NO	NO	YES	NO	YES
3277 (1)	NO	YES	NO	NO	NO	NO
3277 (2) and Model 158	YES	YES	YES	YES	YES	NO

* Optional Feature

Figure INTRO-1. Summary of display console features

DIDOCS SPECIAL PROCEDURES

- MESSAGE DELETION

  Manual Deletion
  Automatic Deletion
  Roll Mode
  Roll-Deletable Mode

- CONTROLLING STATUS DISPLAY

CONTROLLING DISPLAY CONSOLES

CONTROL COMMAND

Request or Cancel Mode of Message Deletion

Change Time Interval for Roll or Roll-Deletable Mode

Check Specifications in Effect

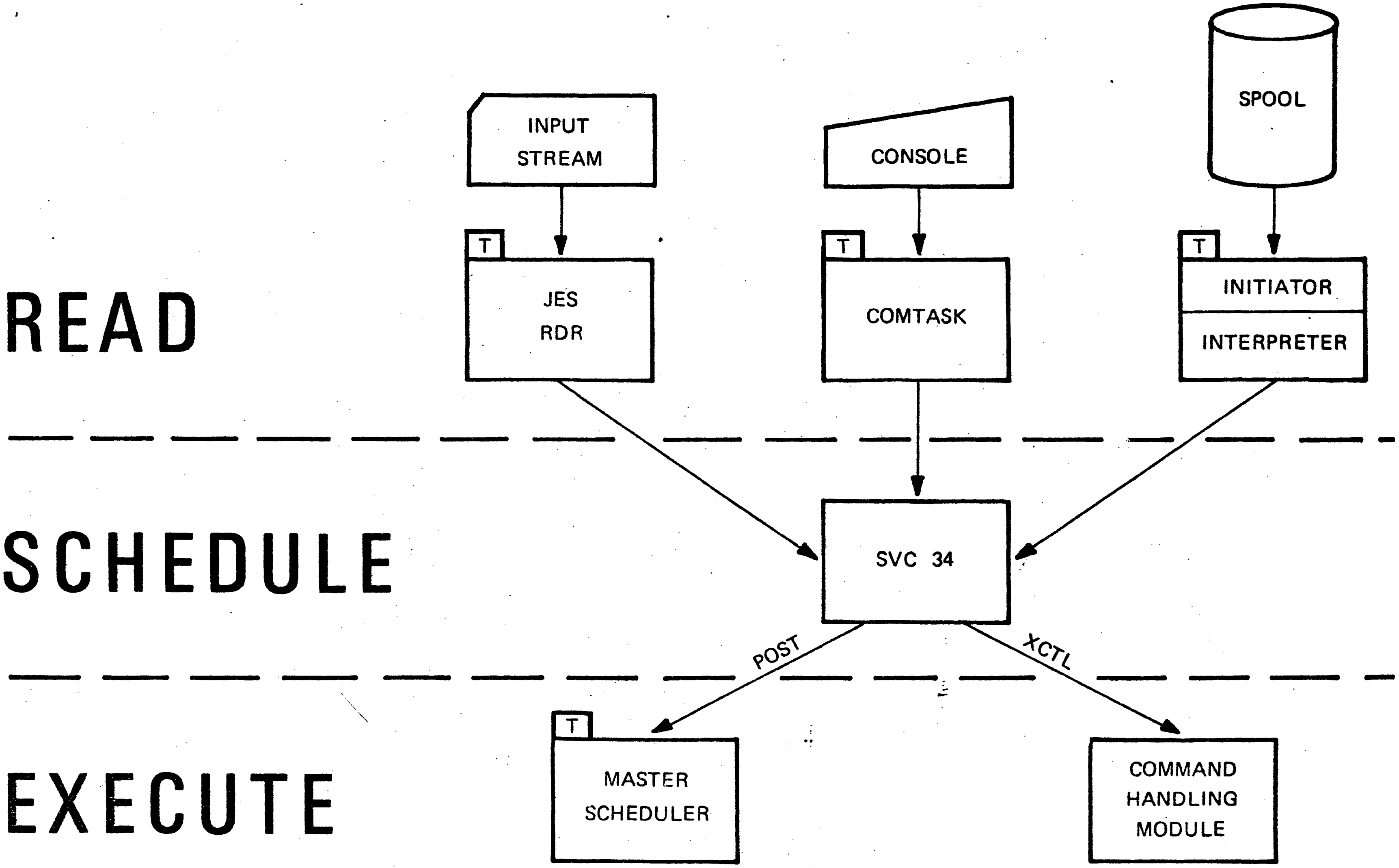
Request Message Numbering

Request Message Renumbering after Message Deletion

Establish Display Areas

Frame Status Disp~~al~~lay Areas

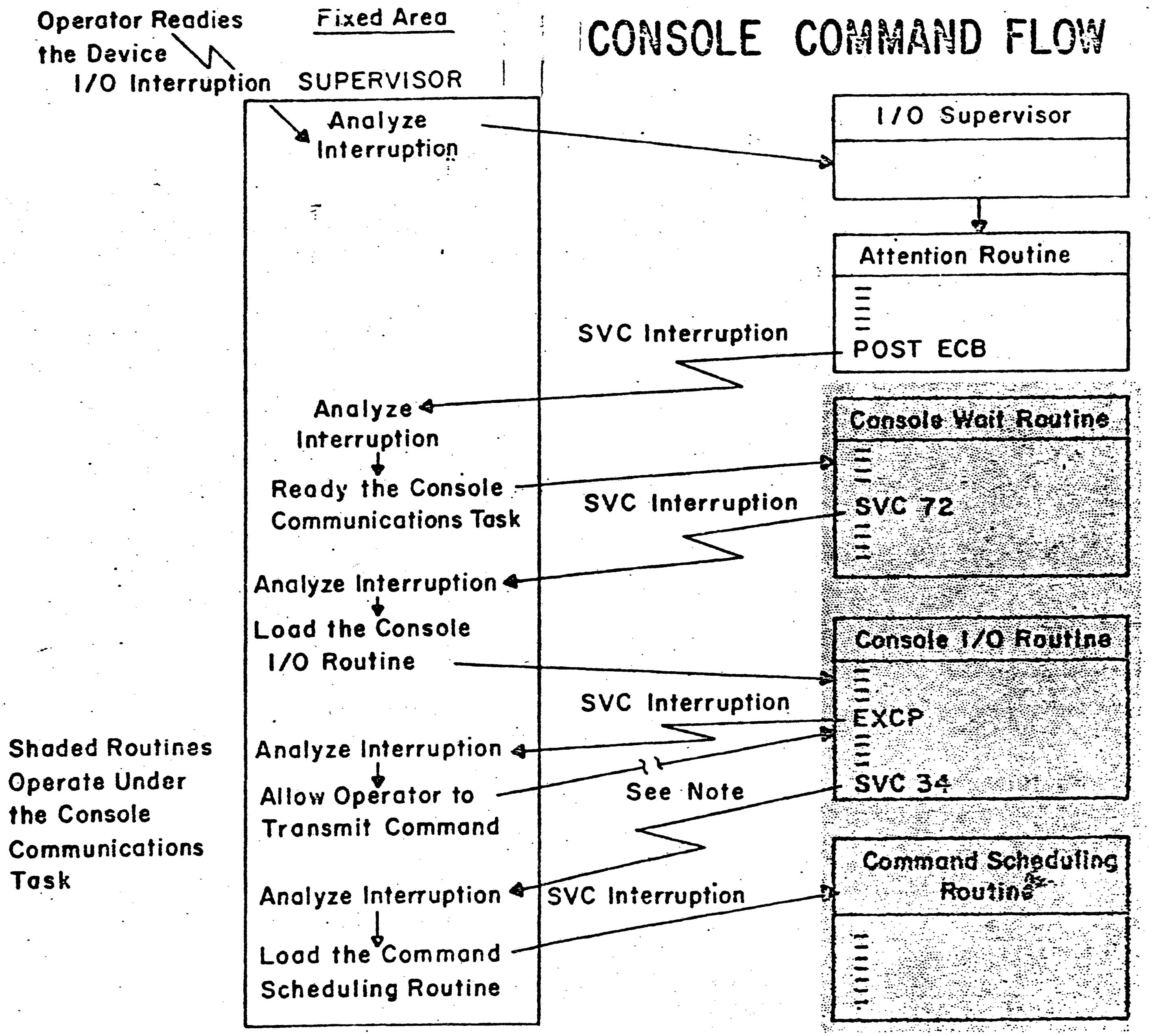
VS1 COMMAND PROCESSING



T : COMPONENT RUNS AS A TASK

1

CONSOLE COMMAND FLOW



Operator Readies the Device

I/O Interruption

Fixed Area

SUPERVISOR

Analyze Interruption

Analyze Interruption

Ready the Console Communications Task

Analyze Interruption

Load the Console I/O Routine

Analyze Interruption

Allow Operator to Transmit Command

Analyze Interruption

Load the Command Scheduling Routine

I/O Supervisor

Attention Routine

POST ECB

SVC Interruption

Console Wait Routine

SVC 72

SVC Interruption

Console I/O Routine

EXCP

SVC Interruption

SVC 34

See Note

SVC Interruption

Command Scheduling Routine

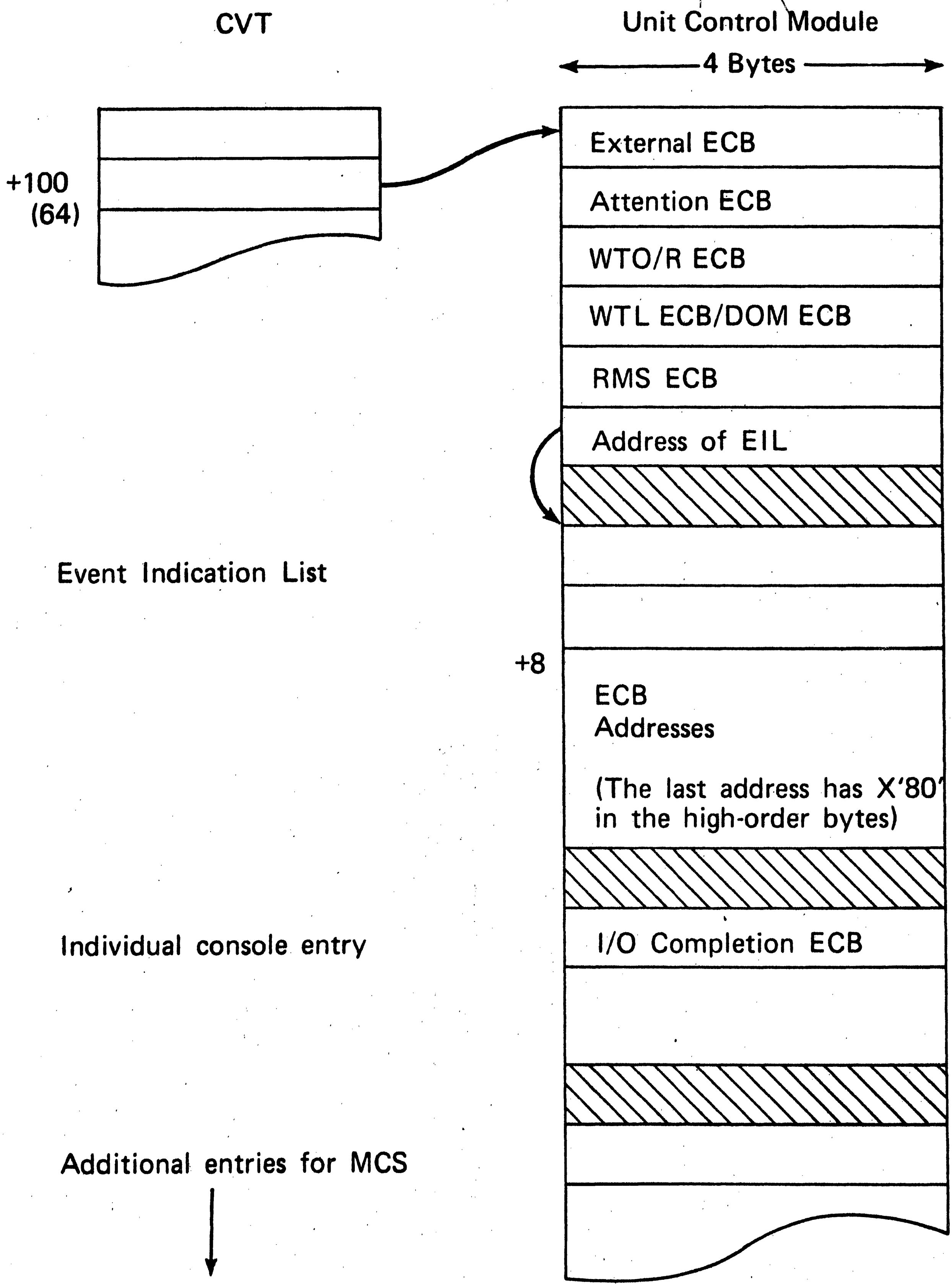
Shaded Routines Operate Under the Console Communications Task

NOTE: While Command is being transmitted, other processing is performed. Control returns to the Consolo I/O routine after the command is trnnsmitted.

© IBM Corp. 1974

2

UCM



CVT

+100
(64)

Unit Control Module

4 Bytes

External ECB

Attention ECB

WTO/R ECB

WTL ECB/DOM ECB

RMS ECB

Address of EIL

Event Indication List

+8

ECB
Addresses

(The last address has X'80' in the high-order bytes)

Individual console entry

I/O Completion ECB

Additional entries for MCS

© IBM Corp. 1974

UCM Event Indication List (EIL)

0(0) UCMEIL Length in Doublewords	1(1) UCMRPYL Last assigned reply ID	2(2) UCMRTCT Route count	3(3) Reserved
4(4) Addr of 2K NIP Message Buffer			
8(8) Addr of External ECB			
12(C) Addr of Attention ECB			
16(10) Addr of WTO/R ECB			
20(14) Addr of DOM ECB			
24(18) Addr RMS ECB			
28(1C) List of all I/O ECB Addresses One word for each console device SYSGENed with a minimum of two entries. The list is variable at SYSGEN only. Last entry has a high order byte = X'80' (Maximum of 128 bytes).			

4

Write Queue Element (WQE)

4 Bytes		
0(0) WQEUSE WQE Use Count	1(1) WQELKPTR Addr of Next WQE	
4(4) WQENBR Message Length		
8(8) WQETXT Message Text (maximum 128 bytes)		
136(88) WQEXA Disposition Flags (see description below)	137(89) Reserved	139(88) WQEAVAIL Buffer Flags (see description below)
140(8C)		

Description of Flags

WQEXA

Disposition Flags with following meanings:

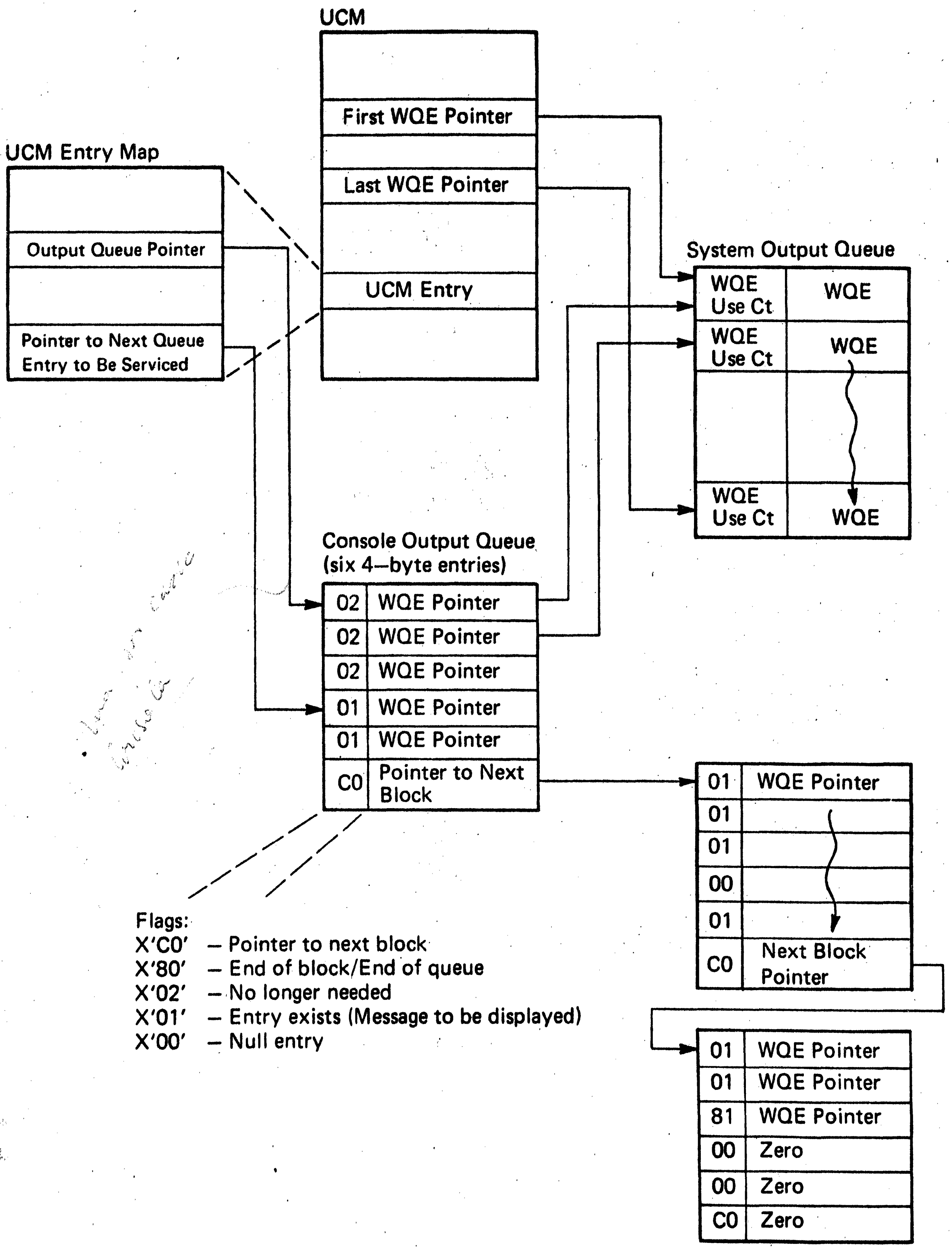
WQEPURGE	X'80':	Purge.
WQEQFHC	X'40':	Queue for hard copy.
WQERQE	X'20':	RQE exists for this WQE.
WQEQDFHC	X'10':	Queue for hard copy.
WQEXWTOR	X'08':	WQE created for WTOR.

WQEAVAIL

Buffer Flags with the following meanings:

RQEBUFA	X'80':	Buffer is free.
RQEBUFB	X'40':	Buffer is in use.
	X'20':	Reserved.
RQEBUFD	X'10':	Buffer obtained dynamically.
RQEBUFE	X'08':	Buffer has been serviced.

System and Console Output Queues (with MCS)



Flags:

X'C0'	– Pointer to next block
X'80'	– End of block/End of queue
X'02'	– No longer needed
X'01'	– Entry exists (Message to be displayed)
X'00'	– Null entry

[illegible]

Reply Queue Element (RPQE)

4 Bytes

<table>
<tr><td colspan="4">0(0)
RQELKP
Addr of next reply queue element</td></tr>
<tr><td colspan="2">4(4)
RQEID
Reply identification no.</td><td>6(6)
RQEXA
Purge Flags (see description below)</td><td>7(7)
RQEAVAIL
Buffer Flags (see description below)</td></tr>
<tr><td>8(8)
RQETJID1
First half of TSO user ID</td><td colspan="3">RQETCB
Addr of TCB task that issued message for which this RPQE represents a reply</td></tr>
<tr><td colspan="4">12(C)
RQEXB
Addr of Communications Task emergency message to cancel replies</td></tr>
<tr><td>16(10)
RQELNGTH
Maximum length of reply</td><td colspan="3">RQEPTR
Addr of user's buffer</td></tr>
<tr><td>20(14)
RQETJID1
Second half of TSO user ID</td><td colspan="3">RQEELB
Addr of user's ECB</td></tr>
</table>

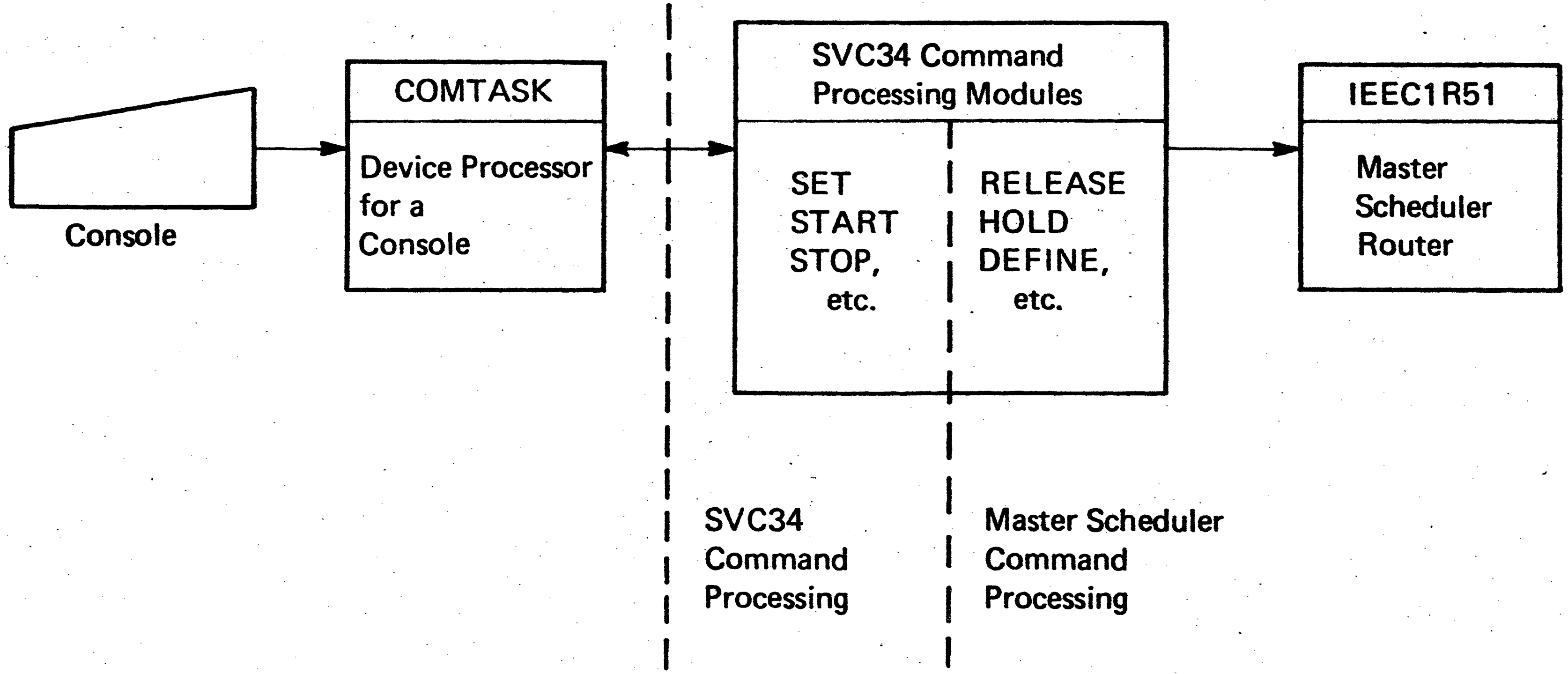
Description of Flags

RQEXA Purge Flags, with the following meanings when on:
Bit 6.0 – Associated reply will be purged.

RQEAVAIL Buffer Flags with the following meanings:
RQEBUFAX'80' – Buffer is free.
RQEBUFBX'40' – Buffer is in use.
X'20' – Reserved.
RQEBUFDX'10' – Buffer has been obtained dynamically.
RQEBUFEX'08' – Buffer has been serviced.

© IBM Corp. 1974

Command Processing



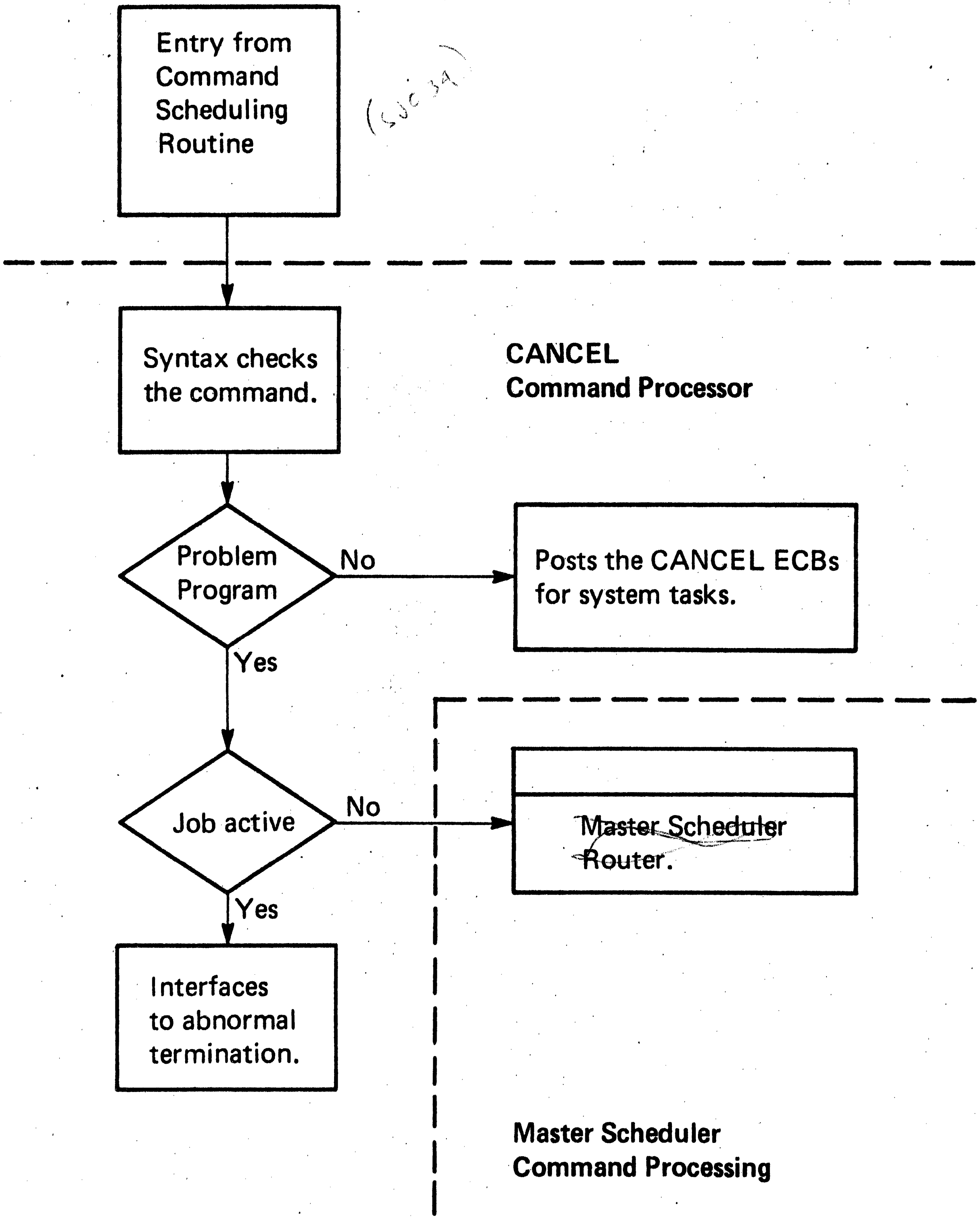
Command Processor Module – SVC34 Command

Command	SVC34 Command Processor Module
CANCEL	IEE3703D
CONTROL	IEE7503D
DISPLAY RT	IEE8803D
DISPLAY U	IEE20110
DUMP	IEE60110
LISTBC	IEELIST
LOG	IEE1603D
LOGOFF	IEE8703D
LOGON	IEELGON
MODE	IGF2603D
MODIFY	IEE4503D
MONITOR	IEE7103D
MOUNT	IEE1903D
MSGRT	IEE6303D
REPLY	IEE1A03D
ROUTE	IEERTE
SEND	IEEVSEND
SET	IEE0603D
START	IEE1903D
STOP	IEE4503D
STOPMN	IEE4503D
SWAP	IGF2503D
UNLOAD	IEE1103D
VARY	IEE3203D
WRITELOG	IEE1603D
WRITER	IEE9903D

Command Processor Module
Master Scheduler Command

Command	SVC34 Command Processor Module
~~CANCEL (job on a queue)~~	~~IEE3703D~~
DEFINE	IEESD571
DISPLAY	IEE3503D
~~DISPLAY ACTIVE~~	~~IEESD566~~
HALT	IEE1403D
HOLD	IEE3503D
MONITOR ACTIVE	IEESD566
RELEASE	IEE3503D
RESET	IEE3503D
SWITCH	IEE1403D

CANCEL Command



Entry from Command Scheduling Routine

(SJC 34)

CANCEL Command Processor

Syntax checks the command.

Problem Program

No

Posts the CANCEL ECBs for system tasks.

Yes

Job active

No

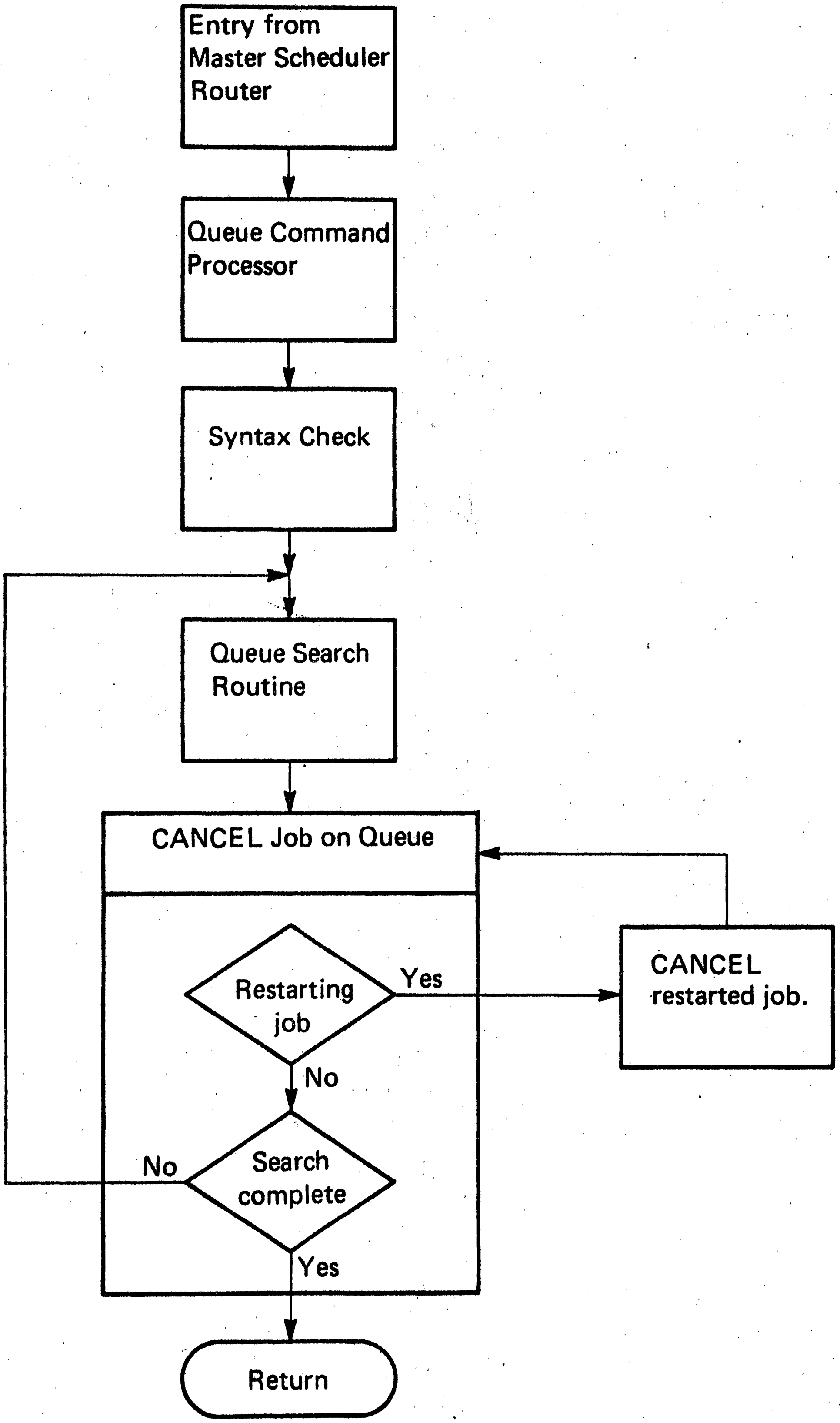
~~Master Scheduler Router.~~

Yes

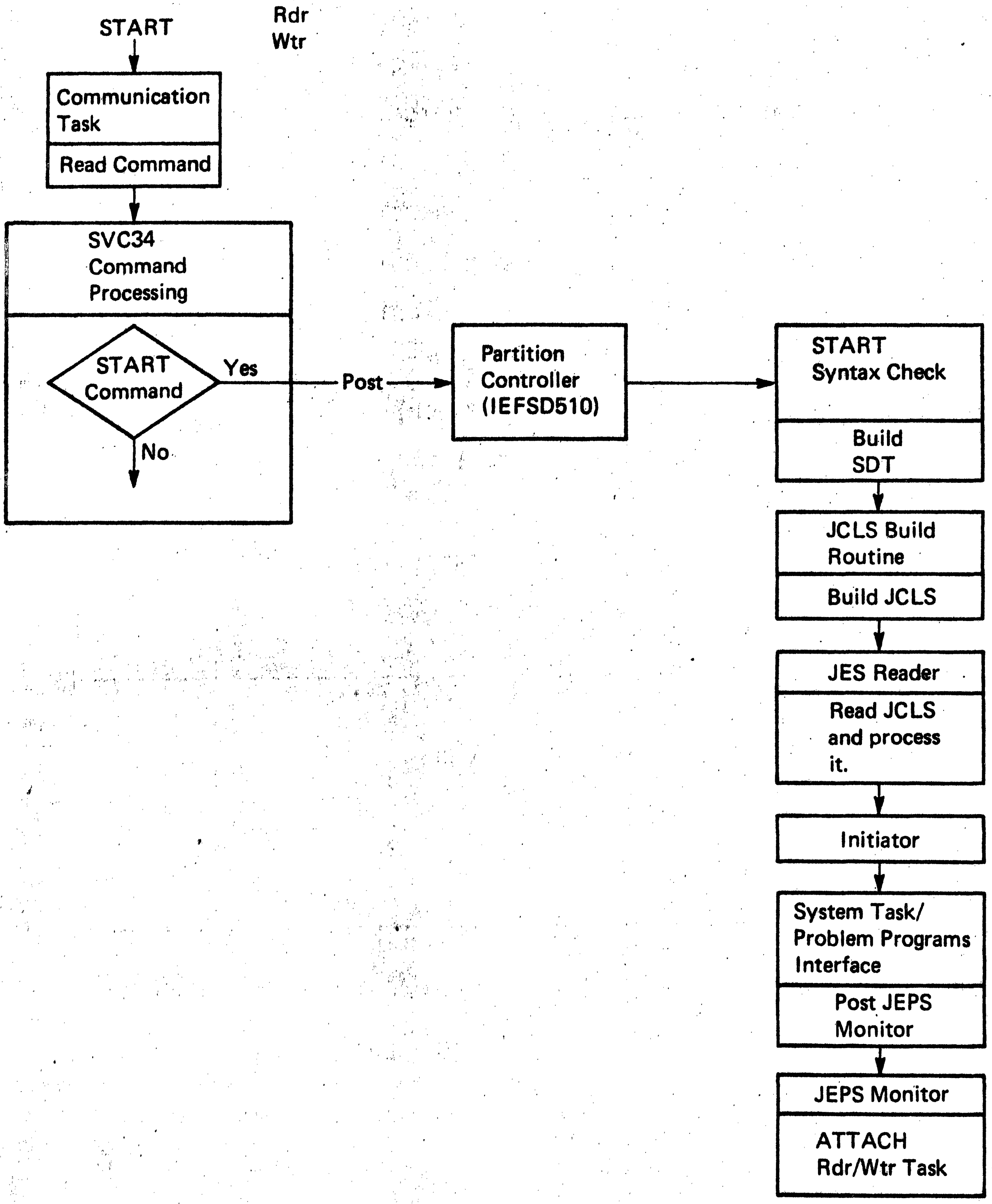
Interfaces to abnormal termination.

Master Scheduler Command Processing

CANCEL Job on Queue



Start Command Processing



Entry Point	Module-IEEVLNKT	Load Module-IEEVRCTL
IEEVLNKT	IEEVMNT2	(MOUNT)
	IHKRJBGM	(RJERDR)
	IEFVRRC	(CH PT RST)
	IKDINPRO	(GRAPHICS)
	IKDSGJP	(GRAPHICS)
	IKAGFX	(GRAPHICS)
	IKAGJP	(GRAPHICS)
	IHKBGN	(GRAPHICS)
	IEFIIC	(INITIATOR)
	IEFDSO	(DSO WTB)
	HHLGTF	(GTF)
	IHLGTF	(GTF)
	IGFTMCHK	(MISSING INTERRUPT)
	IFSRTAM	(RTAM)
	X'00'	
	IEFOSC01	(WTR)
	IEFDSO	(DSO PGM)
	IEFVMA	(MONITOR)
	IEFSRTAM	(RTAM)
	X'00'	

Each entry is 8 bytes. The two tables are delimited by X'00'.

Table 1: Labels the PGM name entered on the execute card as a system task.

Table 2: Those programs named on the execute card which are NOT to have Data Set Integrity.

LAB INTRODUCTION

1. Each student will be assigned a team number which will be associated with the jobnames that he uses during the class.

2. The following JOB statement format is required:

```
//Exxxxtn  JOB  3yyyyyz1F3251L,'member names',CLASS=C
```

t = Team number (A-Z) L
n = Exercise number (1-9)
xxxx = Class number (provided by the instructor) ØØØ5
yyyyy = Course code H374Ø
z = Number provided by instructor 1

3. The following macros are available to provide standard linkage instructions:

```
       ┌    ┌3┐          ┌0              ┐                                 ┐
ENTER  │ R= │2│,EQ=Y,STG=│(begin,end,...)│,CB=YES,CLOSE=(closeoperands)    │
       │    └1┘          └               ┘                                 │
       └                                                                   ┘
```

R= Base register assignment (Default is 1)
1 - Assigns Base Register 12
2 - Assigns Base Register 12 and 11
3 - Assigns Base Register 12, 11 and 10

EQ= Requests Register Equates (Default is NO). Example:

```
R0  EQU 0
R9  EQU 9    (equates exist for all 16 registers)
R15 EQU 15
```

STG= If this parameter is not coded the following information is printed if ABEND occurs:

Floating Point Register 0 contains the ABEND PSW
Floating Point Register 2 contains System Code
Floating Point Register 4 contains User Code
Floating Point Register 6 contains the contents of general registers 0 and 1 respectively at ABEND
General Registers 0-15 (0-1 invalid, see FP6 above)
Problem Program Module (Subpools not printed)

(begin,end,...) causes only programmer selected areas of storage to be listed if ABEND occurs.

0 causes a normal SYSUDUMP if ABEND occurs.

CB= System control blocks to be printed if ABEND occurs. (Default is NO).

CLOSE Datasets to be closed if ABEND occurs. Use normal CLOSE operand format.

LEAVE RC=n (To be used in conjunction with the ENTER macro)

RC= Return code. The n value may be any self-defining term 0-4095 or any general register if register notation is used. If this parameter is not coded, the default is RC=0.

Example: (Each illustration results in return code of 8)

```
a. LEAVE    RC=8

b. LA       15,8
   LEAVE    RC=(15)
```

Note: This macro restores caller Regs, frees Save Area GETMAINed by ENTER macro and returns to caller.

4. The following macro is available for your convenience in converting data fields to printable HEX character format.

$$\text{CTH}\quad \left\{\begin{matrix}\text{(r1)}\\ \text{name1}\end{matrix}\right\},\left\{\begin{matrix}\text{(r2)}\\ \text{name2}\end{matrix}\right\},\left\{\begin{matrix}\text{(r3)}\\ \text{n}\end{matrix}\right\}$$

r1 Any register 2-12. The register must contain the address of the first byte of the field into which the converted (printable) characters are to be placed.

name1 The symbolic name of the first byte of the field into which the converted characters are to be placed.

r2 Any register 2-12. The register must contain the address of the first byte of the field which contains the data to be converted.

name2 The symbolic name of the first byte of the field which contains the data to be converted.

r3 Any register 2-12. The register must contain a binary value of the NUMBER OF BYTES of data to be converted to printable HEX character format from the area specified by r2 or name2.

n The decimal value of the NUMBER OF BYTES of data to be converted to printable HEX character format from the area specified by r2 or name2.

Note: The length in bytes of the field specified by r1 or name1 must be twice the value specified in r3 or n.

If the register notation (r) format is used in any operand, only the decimal value of the register number is valid. Register equates are invalid.

This macro modifies general registers 0, 1, 14 and 15.

5. Note: The message portion of a WTO macro begins at NAME+8.

MACHINE EXERCISE 1

1. Code a TYPE 3 reentrant SVC routine to monitor the unassigned virtual storage in a problem program partition.

2. Your routine will be invoked and tested for reenterability and accuracy by an SVC Driver program.

3. Use a procedure named TESTSVC to assemble and test your routine. Your Job Stream should appear as follows:

```
//stepname    EXEC  TESTSVC,TEAM=t
//ASM.SYSIN   DD    *

                    SVC routine source code
```

4. Registers upon entry to your SVC routine will be as follows:

    Register 3 → Pointer to CVT
    Register 4 → Pointer to TCB
    Register 5 → Pointer to SVRB
    Register 14 → Return address

5. Name your routine to conform with the standard TYPE 3 naming conventions. Use 255 as the SVC number.

6. Return the accumulated total of unassigned storage as a binary value in Register Ø.

MACHINE EXERCISE 2

SPIE EXIT

A main task has been provided to allow you to extend VS through a SPIE EXIT routine. A SPIE macro has been issued by the main line code (named TASK). You are to write a SPIE EXIT routine that will trap all program check interrupts (1 - F) that occur in TASK.

The requirements for this routine are:

1. Name this routine SPIEXIT
2. Issue a WTO 'SPIE EXIT IC nnn,' ROUTCDE=11 where nnn is the Interrupt Code
3. Issue a snap with DCB=SNAP1, ID=30, SDATA=(CB), and PDATA=(PSW,REGS)
4. Return to the main line at RETURN

After a successful return from the SPIEXIT, the message 'CORRECT RETURN FROM SPIE' will be issued.

To assemble, link edit and execute your routine with TASK use the following JCL:

```
//S1    EXEC  ALGSPIE
//ASM.SYSIN  DD   *
 [ SPIEXIT ]
   END CARD
```

```
STMT    SOURCE STATEMENT                                                              ASM 0102 11.16 06/

   1            PRINT NOGEN                                                                           0
   2  *  THIS TASK ISSUES A SPIE MACRO                                                                0

   4 TASK       CSECT                                                                               000

   6            USING *,11                                                                          000
   7            STM   14,12,12(13)                                                                  000
   8            LR    11,15                                                                         000
   9            LR    12,13                                                                         000
  10            LA    13,SAVE                                                                       000
  11            ST    12,SAVE+4                                                                     000
  12            ST    13,8(12)                                                                      000

  14            OPEN  (SNAP1,(OUTPUT))                                                              000

  21            SPIE  SPIEXIT,((1,15))                                                              000

  26            ST    1,SAVESPIE                                                                    000

  31            SP    BLANKS,BLANKS                                                                 000

  33            WTO   'INCORRECT RETURN FROM SPIE',ROUTCDE=11      is msg.                          000

  44 RETURN     EQU   *                                                                             000
  45            WTO   'CORRECT RETURN FROM SPIE',ROUTCDE=11                                           0

  56            CLOSE SNAP1
  62            L     13,SAVE+4                                                                     000
  63            LM    14,12,12(13)                                                                  000
  64            BR    14                                                                            000
  65            DS    0D                                                                            000
  66 SAVESPIE   DC    F'0'                                                                          000
  67 SAVE       DC    18F'0'                                                                        000
  68 BLANKS     DC    CL4' '                                                                        000
  69 SNAP1      DCB   DDNAME=SNAP1,DSORG=PS,RECFM=VBA,BLKSIZE=882,                                 X00
                      LRECL=125,MACRF=(W)                                                           000
 101            LTORG                                                                               000
```

MACHINE EXERCISE 3

EXITS

A main task and a subtask have been provided to allow you to extend VS through EXIT routines. The exits which these tasks support are:

STAE EXIT and RETRY ROUTINE

Attach End-of-Task Exit

It is recommended that this exercise be done in two parts.
Part one is the STAE and RETRY.
Part two is the Attach End-of-TASK.

NOTE: If you use ENTER, then specify STG=0.

Part 1

STAE EXIT

In the parameter list passed by STAE, the first address will be the address of the RETRY routine and the second address will be the return address to continue processing.

The requirements for this routine are:

1. Name this routine STAEXIT
2. Issue a WTO 'STAE EXIT ROUTINE',ROUTCDE=11
3. Issue a snap with DCB=SNAP, ID=21, SDATA=(CB), and PDATA=(PSW,REGS)
4. Indicate RETRY is to be entered

STAE RETRY

The requirements for this routine are:

1. Name this routine RETRY
2. Issue a WTO 'STAE RETRY ROUTINE',ROUTCDE=11
3. Issue a snap with DCB=SNAP, ID=22, SDATA=(CB), and PDATA=(PSW,REGS)
4. Free the STAE workarea
5. Return to main (JSTASK) using the CONTINUE address in the Stae list

After a successful return from the RETRY routine, main will write a message 'INSTRUCTION AFTER PC FOR STAE'.

To assemble, link edit and execute your routines with JSTASK, use the following jcl:

```
//S1     EXEC    ALGSTAE

//ASM.SYSIN    DD    *
     ( RETRY )
    ( STAEEXIT )
     END CARD
```

Part 2 ATTACH END-OF-TASK ROUTINE (ETXR)

Write an End-of-Task routine that will receive control when the SUBTASK terminates.

The requirements for this routine:

1. Name this routine ATCHEXIT
2. Issue a WRO 'ATTACH END OF TASK EXIT ROUTINE', ROUTCDE=11
3. Issue a snap with DCB=SNAP, ID=23,SDATA=(CB), and PDATA=(PSW,REGS)
4. Return to main

After a successful return from ATCHEXIT the message 'JSTASK LEAVE WAIT' will be issued.

The SUBTASK will issue the message:
'SUBTASK HAS STARTED'

To assemble, link edit and execute your routines use the following jcl:

```
//S1      EXEC   ALGSTAE
//ASM.SYSIN    DD
  ( RETRY )
  ( STAEXIT )
  ( ATCHEXIT )
```

END card

```
DR2  STMT    SOURCE STATEMENT                                             ASM 0102 11.28

        1             PRINT NOGEN
        2 *   THIS TASK ISSUES A STAE MACRO. IT ALSO ATTACHES A SUBTASK
        3 JSTASK      CSECT

000     5             USING *,11
        6             STM    14,12,12(13)
        7             LR     11,15
        8             LR     12,13
        9             LA     13,SAVE
       10             ST     12,SAVE+4
       11             ST     13,8(12)

       13             PRINT GEN
       14             CHOOSE ATCHEXIT,MYATHEXT
       15+            L      5,=A(MYATHEXT)
       16             PRINT NOGEN

       18             OPEN   (SNAP,(OUTPUT))

       25             STAE   STAEXIT,CT,PARAM=STAELIST,XCTL=NO

       37             ATTACH EP=SUBTASK,ECB=ECB1,ETXR=(5)

       62             ST     1,TCBSUB

174    64             AP     BLANKS,BLANKS
184    65 CONTINUE    EQU    *

       67             WTO    'INSTRUTION AFTER PC FOR STAE',ROUTCDE=11

       79             WTO    'JSTASK ENTER WAIT',ROUTCDE=11

       89             WAIT   ECB=ECB1

       95             WTO    'JSTASK LEAVE WAIT',ROUTCDE=11

      104             DETACH TCBSUB

      111             CLOSE SNAP
      117             L      13,SAVE+4
      118             LM     14,12,12(13)
      119             SR     15,15
      120             BR     14


      122 SAVE        DC     18F'0'
      123 STAELIST    DC     A(RETRY)
      124             DC     A(CONTINUE)
168   125             ORG    *-4
      126             DC     X'80'
      127             DS     CL3
      128 ECB1        DC     F'0'
      129 TCBSUB      DC     F'0'
      130 BLANKS      DC     CL4' '
```

```
R     STMT    SOURCE STATEMENT                                                  ASM 0102 11.28

      181 SNAP     DCB    DDNAME=SNAP,DSORG=PS,RECFM=VBA,BLKSIZE=882,                    Y
                          LRECL=125,MACRF=(W)
      183          LTORG
      184                 =A(MYATHEXT)
      185          DS     0D
D8    186 MYATHEXT EQU    *
      187          BR     14
```

MACHINE EXERCISE 4

1. Code an SMF retrieval program to accumulate the following list of information from a cataloged SMF data set named TEST.SYSPROG.SMFDATA.

   a. Accumulated JOB page-ins

   b. Accumulated JOB page-outs

   c. Accumulated JOB CPU time

   d. Accumulated entire SYSTEM page-ins

   e. Accumulated entire SYSTEM page-outs

   f. Start time for SMF interval being processed

   g. Accumulated SYSTEM wait time

   h. Stop time for SMF interval being processed

2. Each accumulator in the above list should be a full word binary format.

3. The accumulators should be arranged in the exact order shown in the above list.

4. All information needed can be obtained from type 1,4 and 12 SMF records.

5. When the program has completed processing the SMF data set CALL an assembler language routine named PRTSMF and pass it the address of the beginning of the list. This routine will format and print a summary report of the accumulated information.

6. Use the procedure named ASMCLG to assemble, link and execute your program. This procedure is the same as ASMFCLG except it has the macro library to allow you to use the ENTER and LEAVE macros concatenated into it.

7. Begin accumulation by accessing only the START time from the first TYPE 1 record in the SMF data set. All other statistics in this and prior records reflect activity during IPL and would distort the purpose of the summary report.

8. The following is an example of the output your program should produce.

VBS

```
TOTAL ELAPSED TIME              15.46 MINUTES

TOTAL JOB CPU TIME                271 SECONDS

TOTAL SCP CPU TIME                364 SECONDS

TOTAL CPU UTILIZATION              68 PERCENT

TOTAL JOB PAGE-INS                939 PAGES

TOTAL JOB PAGE-OUTS               625 PAGES

TOTAL JOB PAGING ACTIVITY        1564 PAGES

TOTAL SCP PAGE-INS                401 PAGES

TOTAL SCP PAGE-OUTS               381 PAGES

TOTAL SCP PAGING ACTIVITY         782 PAGES
```

Tomar el 1er Type 1 y coger the Time y de todos los demás
los ~~time~~ page-in + page-out

MACHINE EXERCISE 6

Ver [illegible]

Requirements: (t = team number)

a. Code a complete PGFX and SIO appendage named IGG019Yt.

b. Modify a PCI appendage (deck provided) and name it IGG019Zt.

2. The appendages are to be used with an existing program that retrieves records from a disk data set using EXCP.

3. Consult the Data Management for System Programmers SRL under the topic APPENDAGES for register contents upon entry to an appendage.

4. Additional pointers and lists are provided by the problem program as indicated on the following page.

5. The PGFX appendage must request all areas referenced in appendages and the channel program to be fixed (CCW's, Lists, Msgs, I/O areas).

6. The SIO appendage must request translation for all exposed CCW's and all unexposed CCW's used in the channel program.

7. The PCI appendage must:

   a. Check the IOB MBBCCHHR field to determine when record number 5 on each track is to be retrieved.

   b. When record 5 is not to be retrieved:

      1) Modify a message DC in the problem program to identify input AREA1 is being used.

      2) Return to supervisor indicating no translation required.

   c. When record 5 is to be retrieved:

      1) Modify a message DC in the problem program to identify input AREA2 is being used.

      2) Alter the TIC ccw at CHANPROG+40 to transfer to ALTCCW.

      3) Return to supervisor indicating the translation required.

8. Use the following JCL to test the appendages:

```
//stepname  EXEC  APENDAGE,TEAM=t

//PGFX.SYSIN  DD  *

      *** SIO source ***

//PCI.SYSIN   DD  *

      *** PCI source ***
```

MACHINE EXERCISE 7

1. Prepare the JCL and Control Statements necessary to cause SPZAP to modify an instruction in a PDS Load Module named &SPZAP(TEAMt). The instruction to be modified is labeled BRANCH in the source listing on the following page.

2. Use the following EXEC statement to invoke a procedure named SUPZAP which will provide all JCL needed to execute this exercise except the DD and Control statements required by the HMASPZAP Service Aid program.

```
//stepname  EXEC  SUPZAP,TEAM=t

t = team number
```

3. The procedure contains the following steps:

```
//ASM   EXEC  (Assemble source program)
//LKED  EXEC  (Create load module &SPZAP(TEAMt)
//GO    EXEC  (Execute program before ZAP)
//ZAP   EXEC  PGM=HMASPZAP
//TEST  EXEC  (Execute program after ZAP)
```

4. Note: The EXEC statement to invoke SPZAP is provided in the procedure thus requiring your JCL to be qualified by the stepname ZAP.

```
  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                ASM

                                      1          PRINT NOGEN
000000                                2 ZAP      CSECT
                                      3          ENTER
0001A8 47F0 C28C      0028C         157 BRANCH   B     MSG
                                    158          WTO   '**********************',ROUTCDE=11
                                    168          WTO   '* I HAVE BEEN ZAPPED *',ROUTCDE=11
                                    178          WTO   '**********************',ROUTCDE=11
                                    188 RETURN   CTH   HEXLIST+08,BRANCH,4
                                    204 HEXLIST  WTO   '        ',ROUTCDE=11
                                    214          LEAVE
                                    225 MSG      WTO   '***************************',ROUTCDE=11
                                    235          WTO   '* TRY AND DELETE THIS MSG *',ROUTCDE=11
                                    245          WTO   '***************************',ROUTCDE=11
00030E 47F0 C218      00218         255          B     RETURN
000000                              256          END   ZAP
```

MACHINE EXERCISE 8

1. Prepare the user EXIT code, Control Statements and JCL necessary to cause PRDMP to list the JOB name, RB name and SVC number (in HEX) for all LOAD, LINK, XCTL and ATTACH records on the GTF Trace Data Set named TEST. SYSPROG.GTFTRACE.

2. Use the following EXEC statement to invoke a procedure named GTFEXIT which will provide all JCL needed to execute this exercise except that required by the PRDMP Service Aid program:

```
//stepname   EXEC  GTFEXIT,TEAM=t

TEAMt = CSECT name of your exit routine
```

3. The procedure contains the EXEC statement for PRDMP with a stepname of GTF thus requiring your JCL to use that name as a qualifier. The necessary STEPLIB statement is also provided by the procedure. Your job stream should appear as follows:

```
//stepname   EXEC  GTFEXIT,TEAM=t
//ASM.SYSIN  DD    *
              ~
              ~    Exit source statements
              ~
//GTF.~~~~    ~    PRDMP DD statements
//GTF.~~~~    ~         and
  etc.        ~    Control statements
```

4. If the ENTER macro is used to provide standard linkage for your EXIT ROUTINE, be sure to include the keyword operand STG=Ø.

5. The CTH macro is available for your convenience in converting the SVC number to printable HEX characters. (See the LAB INTRODUCTION handout for a description of the CTH macro)

VS1 SYSTEM PROGRAMMING CLASS

VIRTUAL STORAGE CONCEPTS AND TASK MANAGEMENT DUMP EXERCISES

Task Management Dump Exercise

Using the dump labeled ABCVS1, answer the following questions.

1. Fill in the addresses (and lengths if applicable) on the attached sheet for the ABLE, BAKER, CHARLIE partition for the following:

   a. Lower partition boundary
   b. Upper partition boundary
   c. Fixed PQA
   d. Problem program area
   e. Pageable PQA

2. Fill in the addresses and lengths on the attached sheet for the following:

   a. Nucleus
   b. SQA
   c. V=R area

3. Fill in the addresses and lengths on the attached sheet for the following:

   a. Pageable SQA
   b. Pageable Supervisor area

4. What is the highest address in virtual storage for this machine?

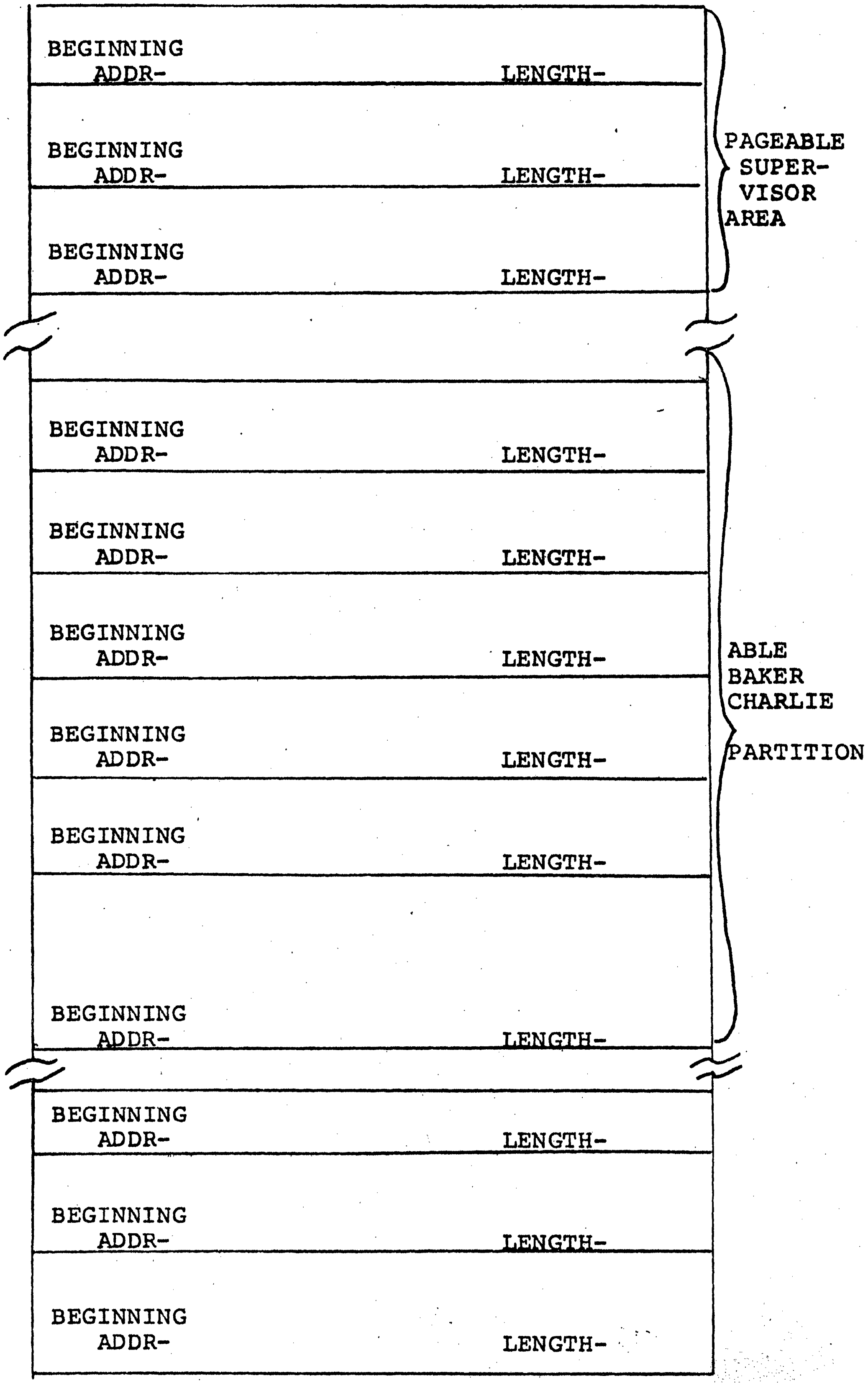
5. What is the TCB ID of the Baker task? of the ABLE task? Why are they not sequential?

6. Indicate where each of the following is located (fixed PQA, pageable PQA, or problem program area)

   a. DEB
   b. TIOT
   c. PRB
   d. LPRB
   e. FSA

7. What is the load point (virtual address) and size of the load module CHARLIE?

Questions 1 - 3



NAME-	BEGINNING ADDR-	LENGTH-
NAME-	BEGINNING ADDR-	LENGTH-
NAME-	BEGINNING ADDR-	LENGTH-
NAME-	BEGINNING ADDR-	LENGTH-
NAME-	BEGINNING ADDR-	LENGTH-
NAME-	BEGINNING ADDR-	LENGTH-
NAME-	BEGINNING ADDR-	LENGTH-
NAME-	BEGINNING ADDR-	LENGTH-
NAME-	BEGINNING ADDR-	LENGTH-
NAME-V=R	BEGINNING ADDR-	LENGTH-
NAME-SQA	BEGINNING ADDR-	LENGTH-
NAME-NUCLEUS	BEGINNING ADDR-	LENGTH-

8. Indicate where each of the free areas listed on the dump is located (fixed PQA, pageable PQA, or problem program area).

9. What control block did the dump program use to determine the free areas referenced in question 8?

10. The registers contained in the SVRB for SVC-601C contain values used by which module? What is the name of the SVC? What type is it?

11. Show the first two entries in the following chains:

    a. FQE for fixed PQA
    b. PFQE for PP
    c. PFQE for pageable PQA
    d. GQE

12. What is the current potential lock setting?

13. Will BAKER go to a SPIE routine on page faults? Why?

14. Construct the RB chain for the active task. List the addresses, the type of RB, and the name of the module associated with each RB.

15. What is the address of the first byte of the SVC transient area?

16. What is the value of the XRBUSE field in the RB for CHARLIE?

17. What is the address of the Partition Information Block?

18. What modules are on the JPQA?

19. Construct the load list for ABLE. List the addresses, the type of RB, and the name of the module associated with each RB.

20. How did the RB for XRAY get created and why is its PSW field zero?

21. List the program check interrupts that are to be handled by the SPIE routine for BAKER.

22. What is the address to be taken if one of the selected program interruptions occurs.

VS/1 PAGE MANAGEMENT DUMP EXERCISE

Please answer the following questions using the SYSABEND DUMP in the VS/1 Release 4 manual(ZR20-4520).

1. What is the address of the Page Supervisor Information Area(PSIA)?

2. What is the address of the Page Device Descriptor Table(PDDT)?

3. What is the address of the Real Storage Page Table(RSPT)?

4. How many RSPTEs are there in this system?

5. How many pageable pages are in this system(Use PSIA)?

6. At the beginning of the RSPT, the entries for the Nucleus and SQA are long term fixed. How many pages indicate LTF?

7. Find the PCB queue headers. If there is no work-to-do, a queue header will point to itself. Are there any PCB's queued up?

8. What are the low and high available page queue threshold values?

9. What is the current available page count?

10. PSIA+X'124' contains the address of the pointer to the available page frame queue. What are the real storage addresses of the first two RSPTEs on the available queue?

11. Examine the RSPTE at X'16EB8'.
    a. What is the Virtual Page Number?

    b. What is the Virtual Address?

    c. What partition is using this page?

    d. Which PDDT entry is associated with this page?

12. The first page device described in the PDDT will contain which virtual pages?

1/1

```
IEA785I FASTNIP ACTIVE
IEE054A DATE=74.322,CLOCK=16.27.26
IEE054A DATE=74.322,CLOCK=16.27.26,GMT
IEA208I DSS FUNCTION INOPERATIVE
 IEE009I LOG NOW RECORDING ON DATA SET X
 IEE303I 230      OFFLINE

 IEE303I 231      OFFLINE
 IEE303I 232      OFFLINE
 IEE303I 233      OFFLINE
 IEE303I 154      OFFLINF
 IEE303I 155      OFFLINE
 IEE303I 156      OFFLINE
 IFE303I 157      OFFLINE
 IEE303I 158      OFFLINE
 IEE303I 159      OFFLINE
 IEE303I 15A      OFFLINE
 IEE303I 15B      OFFLINE
 IEE351I SMF SYS1.MAN RECORDING NOT BEING USED
 IEE048I INITIALIZATION COMPLETED
 IEF403I WTR      STARTED TIME=16.29.35 P01
 IEF236I ALLOC. FOR WTR      00E P01
 IEF237I 00E  ALLOCATED TO IEFRDER P01
 IEF403I H3740RDR STARTED TIME=16.29.39 P02
*00 IEC120A M 00E,P11
r 00,u
 IEF868I 00E WTR WAITING FOR WORK
 IEF236I ALLOC. FOR H3740RDR 00C P02
 IEF237I 00C  ALLOCATED TO IEFRDER P02
 IEF403I INITDVS1 STARTED TIME=16.29.57 P03
*IEA000A 00C,INT REQ,03,0200,4010,,,H3740RDR
 IEF403I INITDVS1 STARTED TIME=16.30.05 P01
 IEF005I PARTITION WAITING FOR WORK  P03
 IEF403I INITDVS1 STARTED TIME=16.30.10 P02
 IEF005I PARTITION WAITING FOR WORK  P01

 IEF005I PARTITION WAITING FOR WORK  P02
```

1/1

```
IEA.50A SPECIFY VIRTUAL STORAGE SIZE

IEA761I PAGE=(V=VS1135,BLK=2048)
IEE054A DATE=74.322,CLOCK=17.14.02

IEE054A DATE=74.322,CLOCK=17.14.02,GMT
IEA101A SPECIFY SYSTEM AND/OR SET PARAMETERS FOR RELEASE 03.0  OS/VS1
r 00,'auto=cold135'
IEA764I NIP45,CMD45,DFN45,JES45,,,SET45,SMF45,
IEA765I  DDG=,DEVSTAT=(MT,DA)
IEA208I DSS FUNCTION INOPERATIVE
 IEE140I SYSTEM CONSOLES
   CONSOLE/ALT  COND AUTH     ID       ROUTCD
     01F/01F      M   ALL     01       1-10,12-16
 IEF032I PARMLIB VALUES TAKEN FOR JES
 IEE866I DEFINE COMMAND BEING PROCESSED
 IEE804I P0=(INACTIVE),P1=(C=ICA,576K,A,E),
 IEE804I P2=(C=JAC,512K,A,F),P3=(C=AC,512K,A,E),
 IEE804I P4=(INACTIVE)
 IEE805I DEFINITION COMPLETED
 IEE101A READY
 IEE029I Q=(,F),SWPRM=(U),JLPRM=(U),SPOOL=(,F)
 IEF070I  SPOOL BEING FORMATTED
 IEE052I V (230,231,232,233,154,155,156,157,158,159,15A,15B),OFFLINE
 IEE009I LOG NOW RECORDING ON DATA SET X
 IEE303I 230     OFFLINE
 IEE303I 231     OFFLINE

 IEE303I 232     OFFLINE
 IEE303I 233     OFFLINE
 IEE303I 154     OFFLINE
 IEE303I 155     OFFLINE

 IEE303I 156     OFFLINE
 IEE303I 157     OFFLINE
 IEE303I 158     OFFLINE
 IEE303I 159     OFFLINE
 IEE303I 15A     OFFLINE
 IEE303I 15B     OFFLINE
 IEE052I MN JOBNAMES,T
 IEE052I S WTR,00E,UCS=P11,FCB=STD2
 IEE052I S H3740RDR,00C
 IEE052I S INITDVS1.P1
 IEE052I S INITDVS1.P2
 IEE052I S INITDVS1.P3
 IEE351I SMF SYS1.MAN RECORDING NOT BEING USED
 IEE048I INITIALIZATION COMPLETED
 IEF403I WTR       STARTED TIME=17.16.34 P01
 IEF236I ALLOC. FOR WTR      00E P01
 IEF237I 00E   ALLOCATED TO IEFRDER P01
 IEF403I H3740RDR STARTED TIME=17.16.37 P02
*00 IEC120A M 00E,P11
r 00,u
 IEF236I ALLOC. FOR H3740RDR 00C P02
 IEF237I 00C   ALLOCATED TO IEFRDER P02
 IEF403I INITDVS1 STARTED TIME=17.16.55 P03
 IEF368I 00E WTR WAITING FOR WORK
 IEA000A  00C,INT REQ,03,0200,4010,,,H3740RDR
 IEF403I INITDVS1 STARTED TIME=17.17.04 P01
 IEF005I PARTITION WAITING FOR WORK  P03
 IEF403I INITDVS1 STARTED TIME=17.17.09 P02
 IEF005I PARTITION WAITING FOR WORK  P01
 IEF005I PARTITION WAITING FOR WORK  P02
```