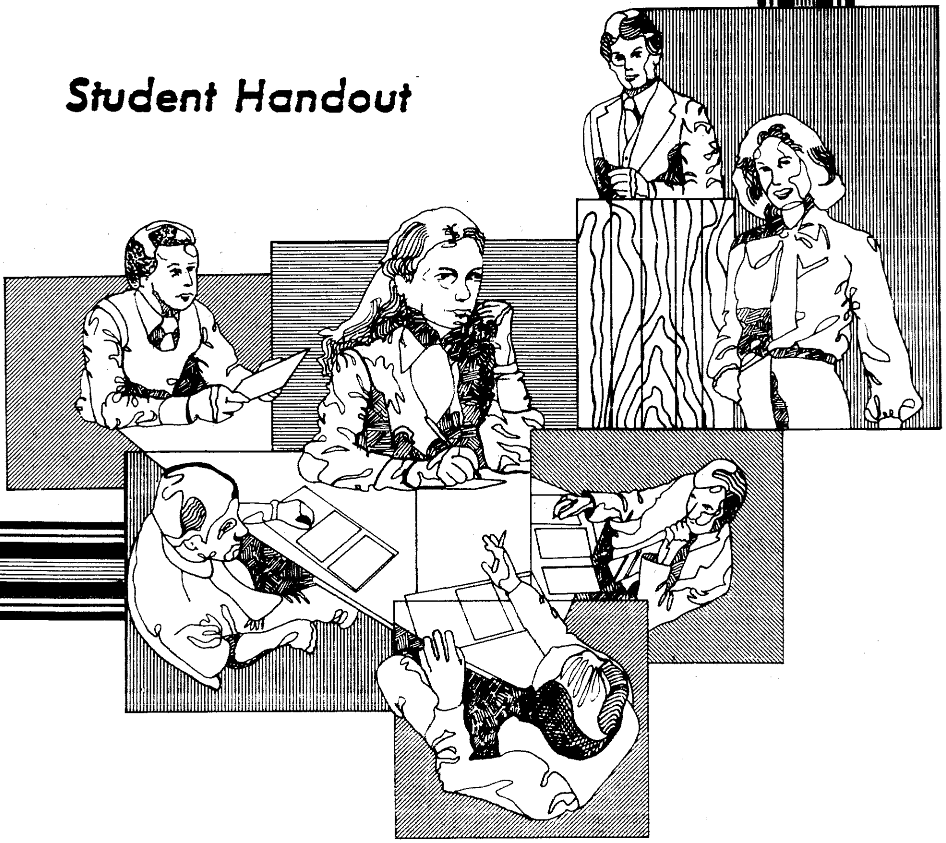
CONTROL DATA
CORPORATION

SEMINARS

Course No. FH4010-1
NOS Analysis

Student Handout



Course No. FH4010-1

NOS Analysis

Student Handout

REVISION C

CONTROL DATA
CORPORATION

PROPRIETARY NOTICE

The ideas and designs set forth in this document are the property of Control Data Corporation and are not to be disseminated, distributed, or otherwise conveyed to third persons without the express written permission of Control Data Corporation.

REVISION RECORD

REVISION	DESCRIPTION
A (06-15-78)	Manual Release
B (10-12-78)	Manual Update
C (04-25-80)	Manual Update
Publication No. FH4010-1	

REVISION LETTERS I, O, Q AND X ARE NOT USED

NOS AANLYSIS

Address comments concerning this manual to:

CONTROL DATA CORPORATION
National Coordinator
5001 West 80th Street
Bloomington, Minnesota 55437
Attn: Curtis Vicha

or use Comment Sheet in the back of this manual.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

GENERAL DESCRIPTION

COURSE TITLE: NOS Analysis

COURSE NUMBER: FH4010

COURSE LENGTH: 10 days

DESCRIPTION:

This course is designed to train the student so that he/she will be conversant in NOS Operating System concepts and its internal workings. The course materials constitute a detailed overview of NOS and are not intended to make the student an "expert" in the details of any one system area.

This is primarily a lecture course. Student assignments consisting of question sets are provided for most of the topics covered by this course.

A set of basic topics should be covered as per the course outline. The block schedule does have "optional" periods that may be filled with one of the optional lessons (to suit the needs of the class) or used as a buffer against running behind "schedule".

PREREQUISITES:

Knowledge of NOS Job Control, NOS Advanced Coding, PP COMPASS.

The student should be familiar with NOS from a user viewpoint in order to be able to grasp the internal concepts presented in the course.

NOS ANALYSIS
COURSE CHART

<table>
<tr><th>HOUR</th><th>DAY 1</th><th>DAY 2</th><th>DAY 3</th><th>DAY 4</th><th>DAY 5</th></tr>
<tr><td rowspan="2">1</td><td>Administrative Details</td><td rowspan="6">CMR (continued)</td><td rowspan="4">Laboratory on CMR</td><td rowspan="2">Laboratory on Monitors</td><td rowspan="4">Laboratory on Storage Move, PPR, Mass Storage</td></tr>
<tr><td rowspan="2">NOS Overview</td></tr>
<tr><td rowspan="2">2</td><td rowspan="3">Storage Move</td></tr>
<tr><td rowspan="3">System Maintenance Concepts</td></tr>
<tr><td rowspan="2">3</td><td rowspan="5">CPUMTR</td><td rowspan="4">Deadstart and Recovery</td></tr>
<tr><td rowspan="3">PP Resident</td></tr>
<tr><td rowspan="2">4</td><td rowspan="6">Central Memory Resident (CMR)</td><td rowspan="5">Exchange Jump</td></tr>
<tr></tr>
<tr><td rowspan="2">5</td><td rowspan="4">Mass Storage Concepts</td><td rowspan="2">Logical Input/Output (CIO)</td></tr>
<tr><td rowspan="3">MTR</td></tr>
<tr><td rowspan="2">6</td><td rowspan="2">Display Console Driver</td></tr>
<tr><td>Delay/Recall</td></tr>
</table>

NOS ANALYSIS
COURSE CHART (CONTINUED)

HOUR	DAY 6	DAY 7	DAY 8	DAY 9	DAY 10
1	Administrative Details Laboratory on DS/Recovery	Job Processing (continued)	Laboratory on Job Processing, Accounting, Perm Files	Laboratory on Resources, Tape Subsystem Unit Record, Q Mgmt	Laboratory on Communications Subsystems
2					
3	Job Processing	Accounting	Resource Control	Networks Overview	Multimainframe
4			Magnetic Tape Subsystem	Time Sharing Subsystem	
5		Permanent Files	Unit Record File Routing Q Management	Remote Batch Facility	MSS Subsystem
6			Other Monitor Requests System Control Point	Transaction Facility	Critique Administrative Details

NOS ANALYSIS
COURSE OUTLINE

LESSON	TOPIC	LENGTH (hours)
1	NETWORK OPERATING SYSTEM OVERVIEW	1
2	SYSTEM MAINTENANCE CONCEPTS	1-1/2
3	CENTRAL MEMORY RESIDENT Low Core Pointers Control Point Area PP Communication Area Dayfile Buffers and Pointers FNT Job Control Block Libraries and Directories	6
4	EXCHANGE JUMP	2-1/2
5	DELAY AND RECALL	1/2
6	CPUMTR	2-1/2
7	MTR	1-1/2
8	STORAGE MOVE	1-1/2
9	PP RESIDENT	1-1/2
10	MASS STORAGE CONCEPTS MST/TRT Mass Storage Driver ECS	2
11	DEADSTART AND RECOVERY THEORY Hardware Deadstart Deadstart Process Mass Storage Recovery System Checkpoints	2
12	LOGICAL INPUT/OUTPUT (CIO)	1
13	DISPLAY CONSOLE DRIVER	1
14	JOB PROCESSING Scheduling Job Initiation Special Entry Point Processing Control Card Processing Error Processing	6-1/2

NOS ANALYSIS
COURSE OUTLINE (Continued)

LESSON	TOPIC	LENGTH (hours)
14	JOB PROCESSING (Continued) Job Completion Field Length Management Rollin/Rollout DMP= and SSJ= Processing Subsystem Scheduling	
15	ACCOUNTING System Resource Unit Validation Files Project Profile Files	2
16	PERMANENT FILES User Index Mappings Indirect and Direct Access Files Catalog and Permit Entries	2
17	RESOURCE CONTROL Deadlock Prevention Resource Files	1
18	MAGNETIC TAPE SUBSYSTEM (MAGNET)	1
19	UNIT RECORD SUBSYSTEM (BATCHIO)	1/2
20	FILE ROUTING AND QUEUE MANAGEMENT	1/2
21	OTHER MONITOR (RA+1) REQUESTS	1/2
22	SYSTEM CONTROL POINT FACILITY	1/2
23	NETWORK PRODUCTS OVERVIEW	1
24	TIME SHARING SUBSYSTEM	1
25	REMOTE BATCH SUBSYSTEM (EI200)	1
26	TRANSACTION SUBSYSTEM (TRANEX/TAF)	1
27	MULTIMAINFRAME	2
28	MASS STORAGE SUBSYSTEM OVERVIEW	1

MATERIALS

STUDENT MATERIAL	PUBLICATION NUMBER
Student Handout NOS Reference Manual, Volume 1	60435400
NOS Reference Manual, Volume 2	60445300
NOS System Programmer's Instant	60449200
NOS Internal Maintenance Specification (IMS)	60454300
NOS Installation Handbook	60435700
NOS Operator's Guide	60435600
NOS System Maintenance Reference Manual	60455380
CYBER Hardware Reference Manual	60456100
MODIFY Reference Manual	60450100
Network Definition Language Reference Manual	60480000
Network Access Methods Reference Manual	60499500

Manual Abbreviations

RM = Reference Manual OG = Operator's Guide IHB = Installation Handbook SMRM = System Maintenance Reference Manual

COMPUTER TIME

Optional. The instruction of this course requires no computer time. However, the instructor may choose to use hands-on machine time for examples, and so forth.

NOMENCLATURE

The nomenclature used to reference sections and pages in the various reference manuals is as follows.

Reference Manual: volume-chapter-page1-page2. Individual pages may be listed as page1,page2,page3,... When the entire section is reference material, no pages will be given.

Installation Handbook: part-section-page1-page2. Individual pages may be listed as page1,page2,page3,... When the entire section is reference material, no pages will be given.

Others: All other manuals will be section-pages, where the page notation is as given above.

LESSON 1
NETWORK OPERATING SYSTEM OVERVIEW

LESSON PREVIEW:

This lesson overviews the Network Operating System, its evolution, its major design philosophies, and its use of the CYBER 170 style hardware.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Describe the evolution of the Network Operating System from its early Chippewa origins to the present day NOS system.
- Describe the control point concept and the meaning of the Reference Address (RA), Field Length (FL) and Program Address (P).
- Identify major NOS system components.
- List the types of files known to NOS and detail how each is used by the system and/or user.

REFERENCES:

NOS IMS - Chapters 1 and 11 NOS Reference Manual, 1-1, 1-2 CYBER 170 Hardware Reference Manual

SUPPLEMENTAL TEXT: HISTORY OF NOS

The CHIPPEWA operating system was the original developmental operating system for Control Data's 6000 computer line. Although there are many descendants of the CHIPPEWA system, Control Data is now primarily concerned with three: NOS, NOS/BE, and CMSE. Figure I-16 shows Control Data 6000, CYBER 70, and CYBER 170 Operating System geneology.

CMSE (Common Maintenance Software Executive) is a operating system- like executive used by Engineering Services to maintain 6000, CYBER 70, CYBER 170 hardware.

NOS/BE evolved from the SCOPE operating system. SCOPE was the name given to the productized version of the CHIPPEWA operating system. Originally CDC planned to make SIPROS the standard 6000 operating system, but when it was delayed and a sizable customer base began using the CHIPPEWA system, SIPROS was abandoned and SCOPE became the major 6000 operating system in 1965.

Several major evolutionary milestones are notable for NOS/BE. In 1967, SCOPE 3.0 was released. This release represented a significant extension of the features previously available. By 1970, when SCOPE 3.3 was released, performance enhancements were needed. The entire I/O system was restructured at that time. In 1972, SCOPE 3.4 was released. This system added significant new features, including the record manager. The final node in the SCOPE to NOS/BE evolution occurred in 1976, with SCOPE being deemphasized and development being directed toward the CYBER 170 mainframe and the NOS/BE operating system.

KRONOS, the father of NOS, had its origin in the CHIPPEWA operating system as well. Unlike SCOPE, it did not immediately become a widely used product. It spent several years, under the name "MACE", as a high-performance benchmark system before it emerged in 1969 as the Time Shared Operating System (TSOS). TSOS was developed for United Computing Systems, a large successful commercial time-sharing company.

The name KRONOS was first used in 1970. KRONOS 1 was the product CYBERNET Services used to expand their existing remote batch capability into a viable time-sharing service. The major emphasis of this productized version of CDC's high-performance operating system was reliability.

KRONOS 2.0, released in 1971, was primarily an extension to the features of KRONOS 1. This emphasis was continued into KRONOS 2.1, a feature rich version of KRONOS 2.0, released in 1973. KRONOS 2.1 incorporated the same product set and data management system as that other operating system (SCOPE), signalling the first real breakthrough in Control Data's movement to common products and operating systems. This release also included the transaction processing subsystem TRANEX.

In 1974, with Control Data's introduction of the CYBER 170 computers and a thrust toward networks, KRONOS begat NOS, the Network Operating System. The first release of NOS consisted of a variety of features primarily directed to support the CYBER 170 RAM (Reliability, Availability, Maintainability) design, to increase its usability (particularily for CYBERNET Services), and to provide more commonality with the other operating system.

The second release of NOS in March, 1976, brought multi-mainframe support into NOS.

The third release of NOS in December, 1976, introduced Control Data's first Network Products offering with the Remote Batch Facility (RBF) and Transaction Facility (TAF, nee TRANEX) applications to the Network Access Method (NAM).

The fourth release of NOS in 1978 expanded upon the networks offerings with the Interactive Facility (IAF, nee TELEX) application. In addition to networks, this release of NOS provides support of full track recording on mass storage devices, and a variety of RAM enhancements.

The fifth release of NOS in 1979 introduced support of the CYBER 176, support of the 7155 mass storage controller and 885 disk drives, and system deadstart from mass storage.

NOS continues enhancement into the 1980s, with a major release on approximately a yearly basis.

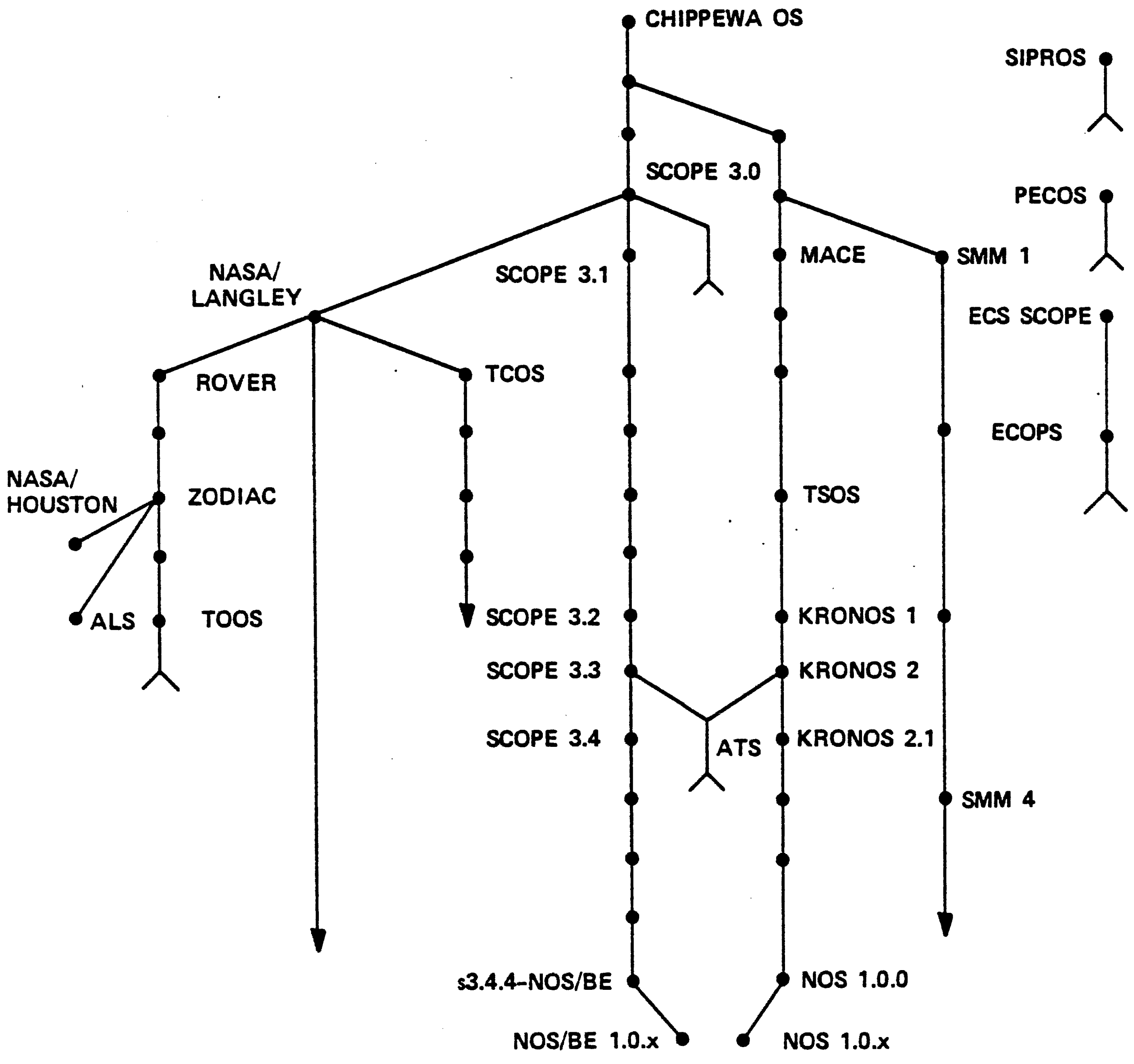
System Naming Conventions

- Operating Systems and the Product Set are generally referred to by a version number and/or the PSR Summary level that corresponds to its release. The nomenclature for NOS is as follows.

	Version	PSR level	Date
KRONOS	2.1.1	387	12/74
NOS	1.0	404	6/75
KRONOS	2.1.2	404	6/75
NOS	1.1	419	3/76
NOS	1.2	439	12/76

COMMON O/S MODULES

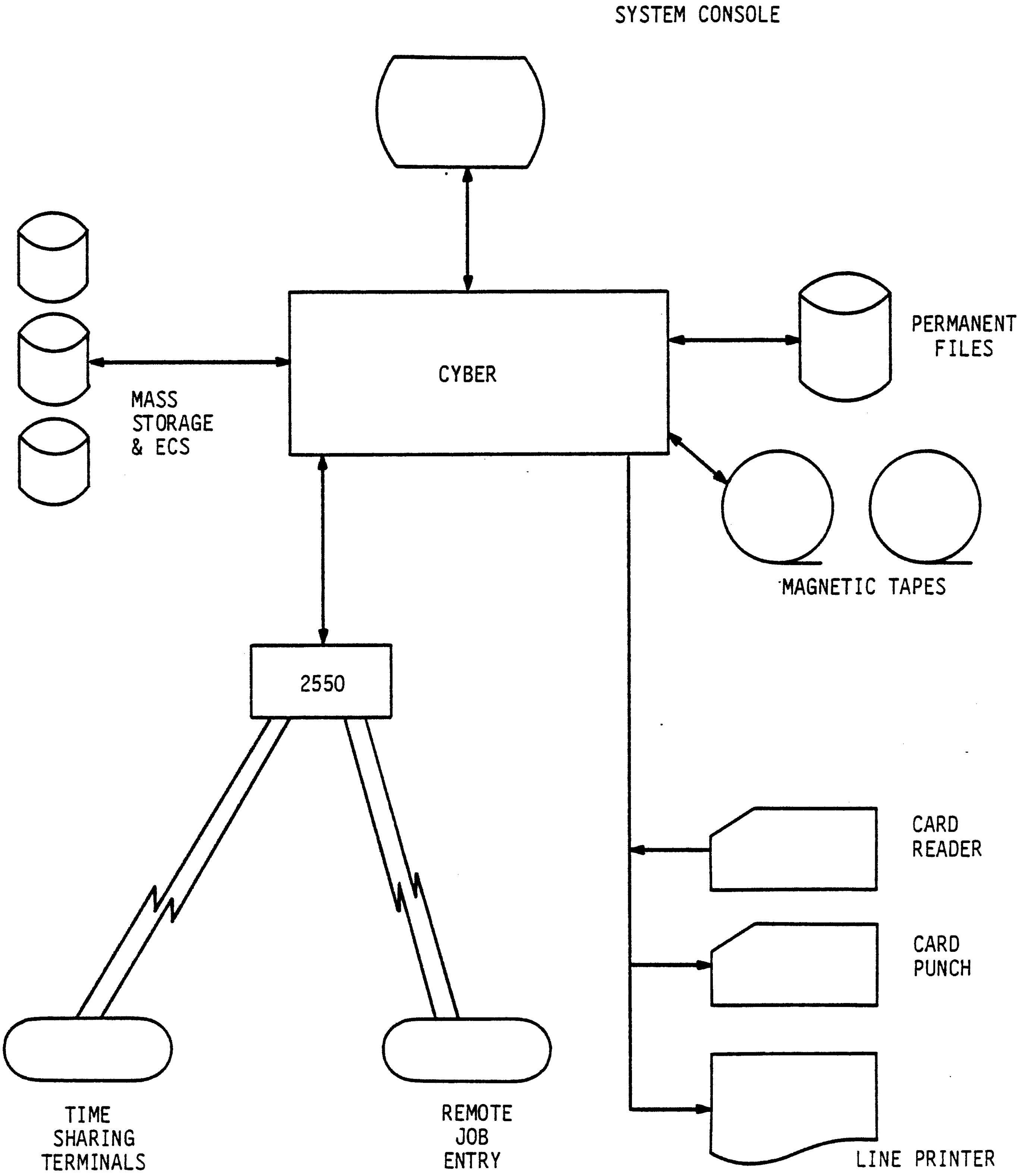
OPERATING SYSTEM GENEALOGY



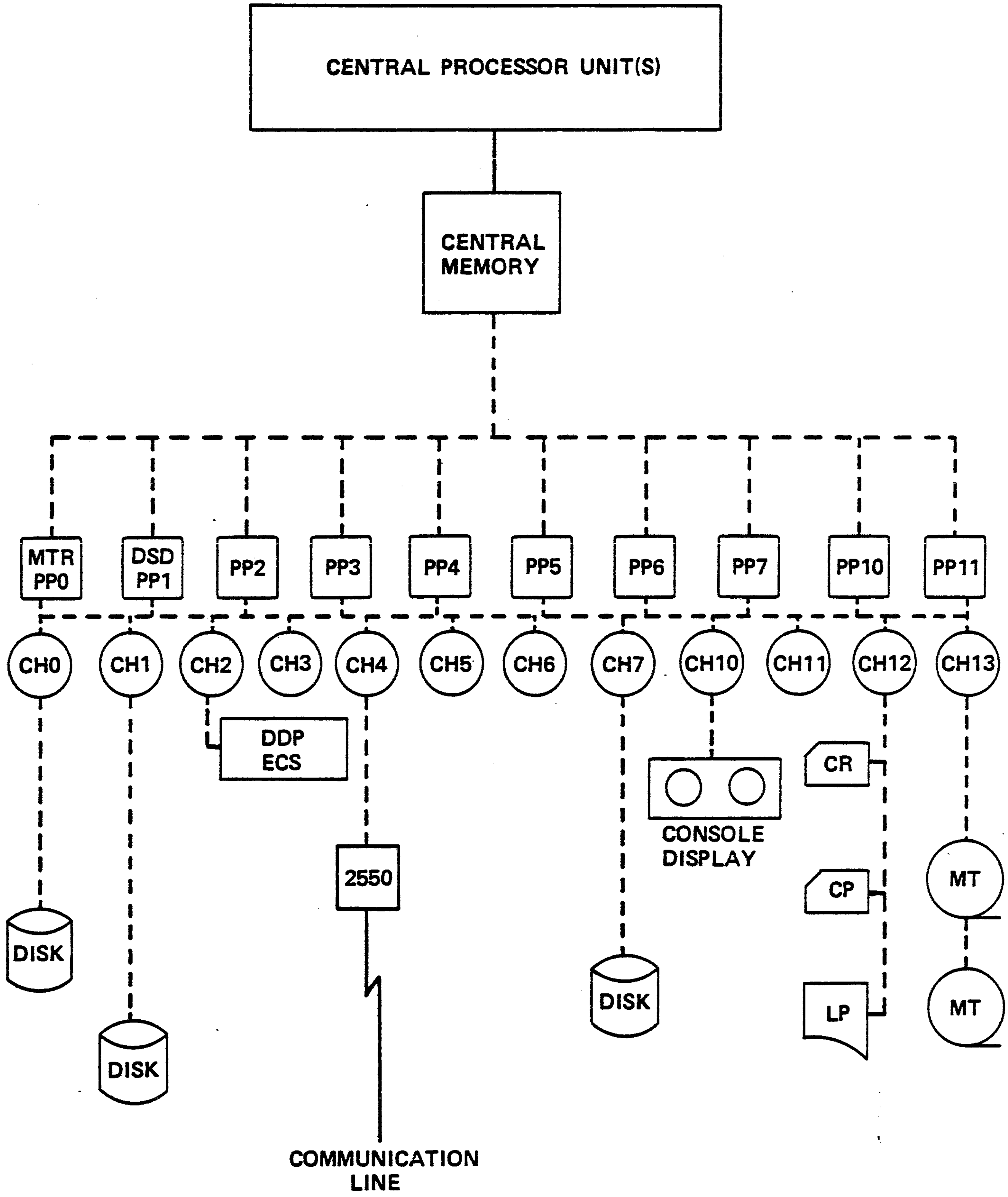
COMMON O/S MODULES

KRONOS/NOS HISTORY

1966	MACE	● BENCHMARK SYSTEM ● EMPHASIS ON PERFORMANCE
1969	TSOS	● INTERACTIVE TIME SHARING ● EMPHASIS ON PERFORMANCE AND RELIABILITY
1970	KRONOS 1	● CDC PRODUCT FROM TSOS ● EMPHASIS ON RELIABILITY
1971	KRONOS 2.0	● MAJOR FEATURE ENHANCEMENTS ● EMPHASIS ON FEATURES
1973	KRONOS 2.1	● SCOPE 3.4 PRODUCT SET ● TRANEX ● EMPHASIS ON FEATURES
1974	NOS 1.0	● 170 FEATURES ● COMMON PRODUCT SET SUPPORT ● EMPHASIS ON FEATURES
1976	NOS 1.1	● MULTIMAINFRAME SUPPORT ● 844-4X SUPPORT
1976	NOS 1.2	● RBF ● TAF
1978	NOS 1.3	● IAF ● FULL TRACKING ● USER ACCESS TO ECS
1979	NOS 1.4	● 7155/885 SUPPORT ● RMS DEADSTART ● CTI SUPPORT ● EXTENDED CHARACTER SET SUPPORT


SYSTEM CONSOLE
MASS STORAGE & ECS
CYBER
PERMANENT FILES
MAGNETIC TAPES
2550
CARD READER
CARD PUNCH
TIME SHARING TERMINALS
REMOTE JOB ENTRY
LINE PRINTER

TYPICAL NOS CONFIGURATION



NOS OPERATING SYSTEM

EFFICIENT PROCESSING

- SYSTEM CONSOLE JOBS
- LOCAL BATCH
- REMOTE BATCH
- TIME SHARING

CPU (CENTRAL PROCESSING UNIT)

- COMPUTATIONAL TASKS
- NO INPUT/OUTPUT CAPABILITY
- COMMUNICATES WITH REST OF SYSTEM THRU CENTRAL MEMORY (RA+1 CALLS)
- USED FOR COMPILATIONS, ASSEMBLIES AND EXECUTION

PPU (PERIPHERAL PROCESSOR UNIT)

- UP TO 20 PPUS - INDEPENDENT
- PERFORMS TASKS FOR REQUESTING PROGRAMS
- 4K MEMORY (12 BIT, 1 BYTE WORDS)
- USED FOR CONTROL OF INPUT/OUTPUT, JOB SCHEDULING, ETC.

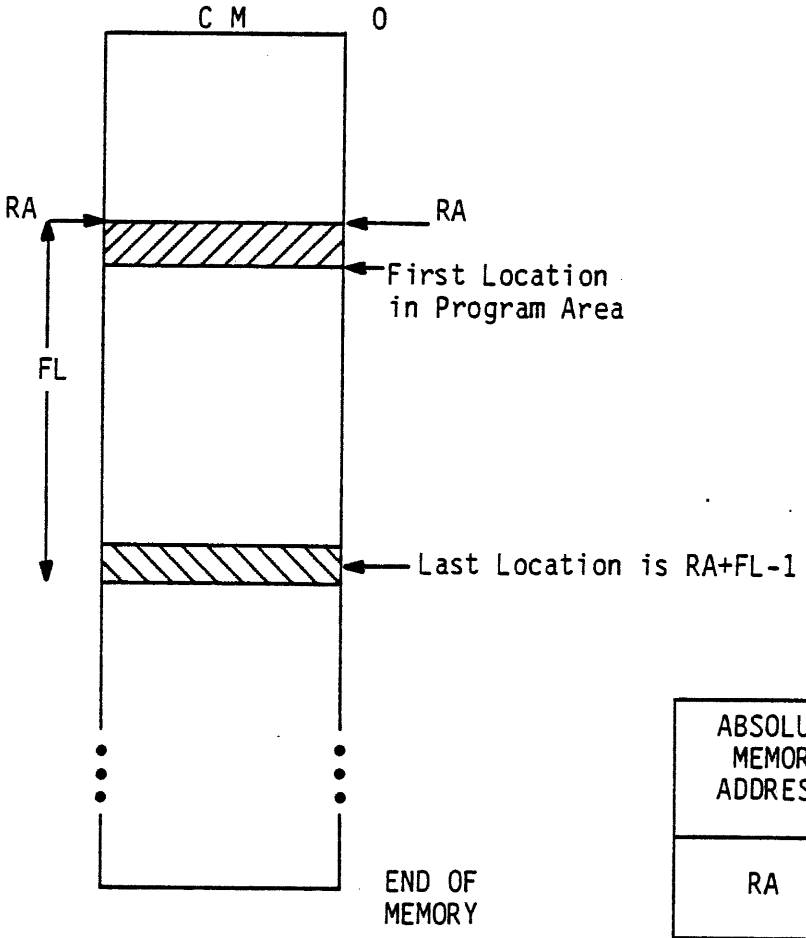
CENTRAL MEMORY

59 48	47 36	35 24	23 12	11 0
BYTE 0	BYTE 1	BYTE 2	BYTE 3	BYTE 4

- 60 BITS LONG
- FIVE 12 BIT BYTES
- BYTES ARE NUMBERED 0-4 (LEFT TO RIGHT)
- 20 OCTAL DIGITS
- 10 DISPLAY CODED CHARACTERS
- 1 OCTAL DIGIT = 3 BITS

CENTRAL MEMORY USAGE

- RA = REFERENCE ADDRESS
  Lower Limit of CM Field
- FL = FIELD LENGTH
  Number of 60 Bit Words
- P = PROGRAM ADDRESS



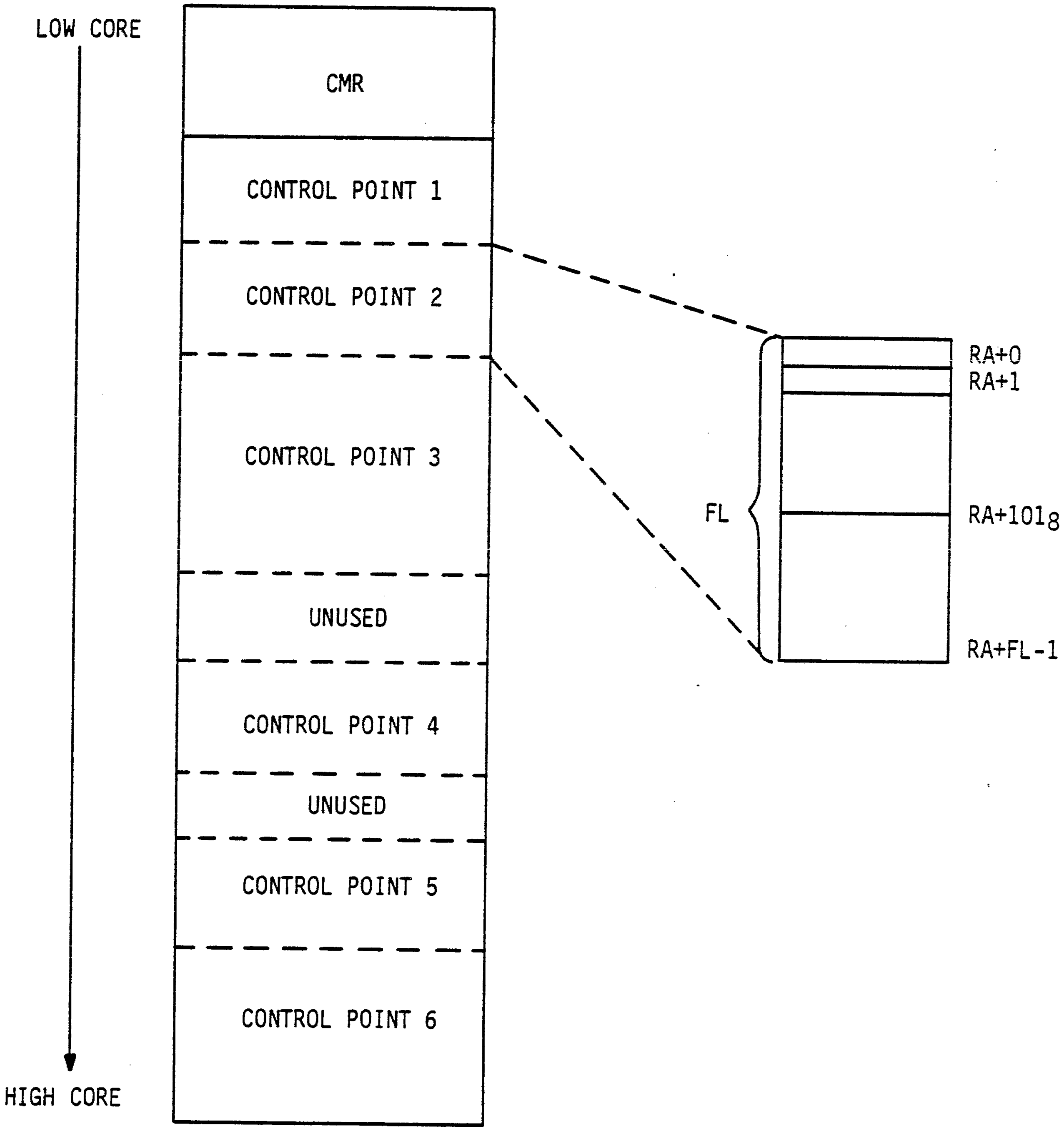
$RA \leq RA+P < RA+FL$

or

$0 \leq P < FL$

ABSOLUTE MEMORY ADDRESS	RELATIVE MEMORY ADDRESS
RA	P=0
RA+P	P<FL
RA+FL	P=FL

CONTROL POINT CONCEPT



CENTRAL MEMORY

LOW CORE

QUEUE PRIORITY		CPU PRIORITY
	CMR	
7777	CPUMTR	100
7776	CONTROL POINT 1 IAF	77
7772	CONTROL POINT 2 TAF	76
4XXX	CONTROL POINT 3 PATSJOB	30
	⋮	
	⋮	
7773	CONTROL POINT n-2 MAGNET	76
7774	CONTROL POINT n-1 BATCHIO	01

HIGH CORE

NETWORK OPERATING SYSTEM

MONITORS

- CPUMTR
- MTR

CONTROL POINTS

- CONTROL POINT 0
- SYSTEM CONTROL POINT
- SUBCONTROL POINTS

JOB FIELD LENGTH

STORAGE MOVE

ROLLOUT/ROLLIN

PROGRAM/SYSTEM COMMUNICATION

RECALL

- PERIODIC
- AUTOMATIC

PRIORITY/RESOURCE TIMES

- CPU PRIORITY
- QUEUE PRIORITY
- CPU TIME SLOT
- CPU TIME SLICE
- CM TIME SLICE

NETWORK OPERATING SYSTEM

JOB ORIGINS

- SYSTEM CONSOLE (SYOT)
- LOCAL BATCH (BCOT)
- REMOTE BATCH (EIOT)
- TIME SHARING (TXOT)
- MULTI TERMINAL (MTOT)

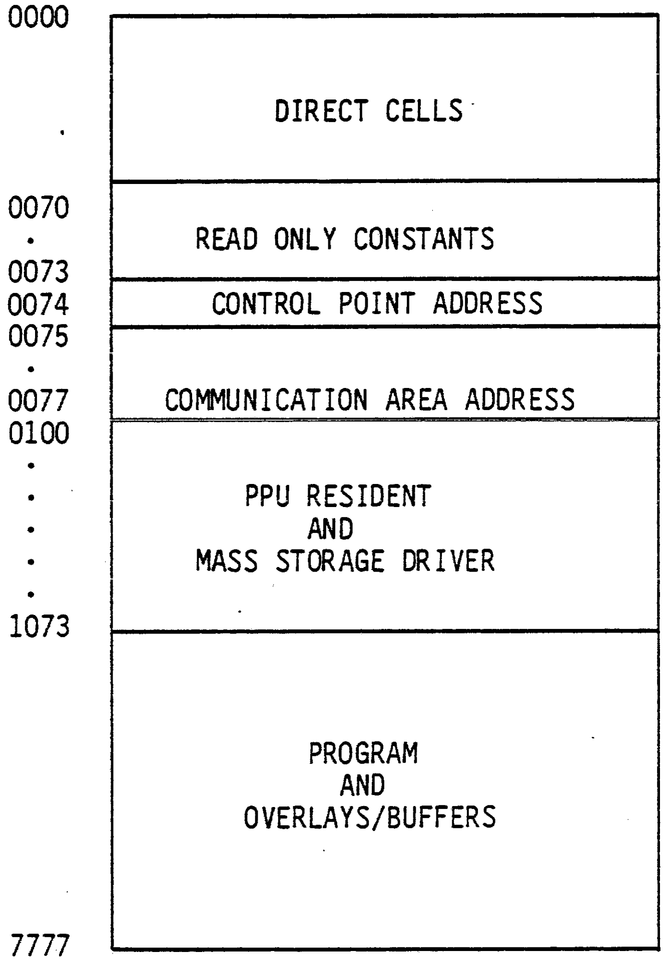
JOB CLASS

- NETWORK SUPERVISOR

PPU USAGE

POOL PROCESSORS

(PP2 through PP11 on 10 PP malchines; PP2 through PP11 and PP20 through PP31 on 20 PP machines.)[+]



RESERVED PP

0	MTR
1	DSD

DEDICATED PP

A Pool PPU permanently assigned to a given control point

[+] PP numbers are in octal notation.

LOCAL FILES

INFT	INPUT
PRFT PHFT	OUTPUT
ROFT TEFT	ROLLOUT
LOFT	LOCAL/SCRATCH
PMFT	DIRECT ACCESS PERMANENT FILE
LIFT	LIBRARY (LOCKED COMMON)
SYFT	SYSTEM
PTFT	PRIMARY

RESERVED NAMES

INPUT
OUTPUT

PUNCH
PUNCHB
P8

SCR
SCR1
SCR2
SCR3
SCR4

Files beginning with five Z; for example, ZZZZZEF

LESSON 2
SYSTEM MAINTENANCE OVERVIEW

LESSON PREVIEW:

The student will review how NOS source code is maintained using the MODIFY utility, how uniform coding techniques and system symbols are employed through various TEXTs to COMPASS and common decks, and the reports generated by the KRONREF utility.

In addition, this lesson introduces the student to the various usages of the term "library" in NOS and how each type of library is maintained.

The concepts discussed in this lesson should give the student a better understanding of why coding standards are enforced and some of the means of presenting a uniform coding approach to help insure the stability and ease of coding of the NOS Operating System.

This lesson also discusses the use of the DSDI (Deadstart Dump Interpreter) to assist the analyst in analyzing system dumps.

OBJECTIVES:

Upon successful completion of this lesson the student should be able to:

- Explain what system symbols and macros are available in various system texts and common decks.
- Describe the naming conventions for common decks and system routines.
- Describe the documentation and COMPASS column conventions required as part of the CODING standard by NOS.
- Explain the reports generated by the KRONREF utility and the value of these reports.
- Distinguish between the various uses of the term "library" and discuss how each "library" is managed in NOS.

REFERENCES:

COMPASS Reference Manual - section 4.3.8 (STEXT) Listing of Lower Cyber Coding standards - CODING from NOS OPL MODIFY Reference Manual NOS RM, 1-2-13-14, 1-13-2, 1-14 NOS RM, 2-1-6-15, 2-A, 2-I NOS IMS - Chapter 32 NOS SMRM, 10 (DSDI)

ASSIGNMENTS:

Read Appendix C of the NOS Reference Manual, Volume 2. Obtain a copy of the Coding Standard from the NOS program library (optional). Obtain a KRONREF of the Operating System. Obtain a CATALOG of the Operating System program library and NOS deadstart tape.

PROGRAM LIBRARY MAINTENANCE

SOURCE CODE FOR NOS IS MAINTAINED USING THE MODIFY UTILITY.

SOURCE CODE IS PASSED THROUGH MODIFY TO GENERATE A "PROGRAM LIBRARY" (PL).

EACH LINE OF SOURCE CODE CARRIES WITH IT A UNIQUE IDENTIFIER AND ACTIVITY INFORMATION.

THE PROGRAM LIBRARY IS THEN A "PERMANENT" MECHANISM TO MAINTAIN SOURCE CODE - RATHER THAN HAVE 600,000 PLUS CARDS, THE PROGRAM LIBRARY IS TYPICALLY MADE A PERMANENT FILE OR COPIES TO MAGNETIC TAPE.

A MODIFY PROGRAM LIBRARY CONSISTS OF THREE RECORD TYPES -

COMMON DECKS	(OPLC)
PROGRAM DECKS	(OPL)
DIRECTORY	(OPLD)

EACH CPU PROGRAM AND PPU ROUTINE RESIDES AS A SEPARATE DECK.

CPUMTR
CIO
DSD
DOPYB
MODIFY

PROGRAM LIBRARY MAINTENANCE

COMMON DECKS ARE ALSO SEPARATE DECKS BUT ARE INCLUDED INTO THE PROGRAM DECK WHEN THE PROGRAM IS BROUGHT TO THE COMPILE FILE.

COMMON DECKS ARE REFERENCED BY
A

```
       *CALL        DECKNAME
```

IN THE SOURCE CODE.

COMMON DECKS ARE USED -

- TO INCREASE EFFICIENCY IN WRITING CODE
- TO INSURE UNIFORMITY OF CODE
- TO DECREASE DEBUGGING TIME
- FOR MODULARITY

```
            IDENT       LOCM
            ENTRY       LCOM
*CALL       COMDECK
            END         LCOM
```

COMMON DECK NAMING

COM Y XXX

XXX = NAME OF ROUTINE

Y = TYPE

C	=	CPU CODE/MACROS
P	=	PPU CODE/MACROS
S	=	SYMBOLS AND MACROS
T	=	TABLES
D	=	DISPLAYS SUBROUTINES
K	=	TAF (TRANEX) CODE/MACROS
M	=	MASS STORAGE
B	=	TAF DATA BASE MANAGER
I	=	INTERACTIVE
A	=	GENERAL MSS
E	=	MSS EXECUTIVE
U	=	MSS UTILITIES
Z	=	MSS DRIVER

SYSTEM TEXT

A SYSTEM TEXT IS AN OVERLAY GENERATED BY COMPASS AS THE RESULT OF THE STEXT PSEUDO INSTRUCTION.

THE SYMBOLS INCLUDED IN THE SYSTEM TEXT OVERLAY ARE ALL SYMBOLS DEFINED IN THE ASSEMBLY EXCEPT THOSE WHICH MEET ONE OF THE FOLLOWING -

- IS RELOCATABLE OR EXTERNAL
- IS QUALIFIED
- IS REDEFINABLE (EG, SET, MIN, MAX, ETC)
- IS READ BY XTEXT
- IS BETWEEN CTEXT AND ENDX
- IS DEFINED BY SST
- IS 8 CHARACTERS LONG WITH ↑↓

SYMBOLS INCLUDED ALSO INCLUDE

- MACROS
- MICROS
- PSEUDO-DEFINED OPCODES

THE PRIMARY SYSTEM TEXT USED BY THE NOS SYSTEM PROGRAMMER IS CALLED

NOSTEXT

IN ADDITION TO SYMBOLS FROM THE SYSTEM TEXT, ADDITIONAL MACROS AND SYMBOLS ARE AVAILABLE TO THE NOS SYSTEM PROGRAMMER THROUGH COMMON DECKS.

COMMON DECK AIDS TO SYSTEM TEXT

NOSTEXT	=	COMMON DECKS	PPCOM CPCOM
SYSTEXT	=	COMMON DECK	CPCOM
PSSTEXT	=	COMMON DECK	COMCMAC COMCCMD

PPCOM - SYMBOL DEFINITIONS FOR CMR AND PPU INTER-COMMUNICATION

- QUEUE PRIORITIES
- DIRECT LOCATION ASSIGNMENT
- PPR ENTRY POINTS
- MS ENTRY POINTS AND ADDRESSES
- MONITOR FUNCTION CODES
- DAYFILE MESSAGE OPTIONS
- FILE TYPES
- JOB ORIGIN/CLASS TYPES
- ERROR FLAGS
- PSEUDO CHANNELS
- SYSTEM POINTERS/LOCATIONS
- CONTROL POINT AREA LOCATIONS
- MST LOCATIONS
- VARIOUS TABLE FORMATS
    - PP COMMUNICATION AREA
    - EST (EQUIPMENT STATUS TABLE)
    - FNT/FST (FILE NAME TABLE/FILE STATUS TABLE)
    - LIBRARY DIRECTOR ENTRIES
    - JOB CONTROL AREA

COMMON DECK AIDS TO SYSTEM TEXT

CPCOM - SYSTEM MACROS FOR COMMONLY USED SYSTEM REQUESTS

- SYSCOM - COMMUNICATION AREA
- SYSTEM MACROS
- FET INITIALIZATION MACROS
- FILE ACTION MACROS
- PERMANENT FILE MACROS
- DATA TRANSFER MACROS

COMCMAC - SYSTEM PROGRAMMER MACROS

COMCCMD - CENTRAL PROGRAM MACROS

COMPMAC - SYSTEM PROGRAMMER MACROS FOR PPU PROGRAMS

S TYPE COMMON DECKS - COMSXXX

THE S TYPE COMMON DECKS CONSIST OF SYMBOL DEFINITIONS THAT HAVE A WIDE USE BUT ARE NOT FREQUENTLY USED TO BE SYSTEM TEXT SYMBOLS.

THE S TYPE COMMON DECKS USUALLY RELATE TO A PARTICULAR ASPECT OF THE SYSTEM.

ACC	ACCOUNTING	NCD	NETWORK COMMUNICATIONS
BIO	BATCHIO	NET	TERMINAL NETWORK
CIO	CIO	PFM	PFM SYMBOLS
CPS	CPUMTR SUBFUNCTIONS	PFS	PFM SUPERVISOR
DSL	DEADSTART	PFU	PF UTILITIES
ESS	DIAGNOSTICS	PRO	PROFILE
EVT	EVENT DESCRIPTIONS	PRD	PRIORITY DEFINITIONS
EXP	EI200	QFS	QUEUE SUPERVISOR
IOQ	I/O QUEUES	REM	INTERACTIVE SUBSYSTEM
JCE	JOB CONTROL DEFAULTS	RSX	RESEX
JIO	JOB I/O SS	SCP	SYSTEM CONTROL POINT FACILITY
JRO	JOB ROLLOUT	SCR	S/C REGISTER
LDR	CPU LOADING	SFS	SPECIAL FILE SUPERVISOR
LFM	LOCAL FILE MANAGER	SRU	SRU PARAMETERS
LSD	LABEL SECTOR	SSE	SYSTEM SECTOR
MMF	MULTIMAINFRAME	SSJ	SPECIAL SYSTEM JOB
MRT	MACHINE RECOVERY	TCM	TELEX COMMUNICATIONS
MSC	MISCELLANEOUS CONSTANTS	TDR	TERMINAL DRIVER
MSI	MS INITIALIZATION	TRX	TRANEX
MSP	MS PROCESSING	WEI	ENHANCED END-OF-INFORMATION
MST	MST INTERLOCKS	ZOL	ZERO LEVEL OVERLAYS
MTR	MTR/COUMTR FUNCTIONS	176	CYBER 176 PARAMETERS
MTX	TAPES	IDS	IDS FUNCTION CODE DEFINITIONS

PROGRAMMING CONVENTIONS

DOCUMENTATION

- DOCUMENT UTILITY

CODING CONVENTIONS

- CODING STANDARD
- COMPASS COLUMN CONVENTION
- NAMING CONVENTIONS

UNIFORM CODING APPROACH

- INCREASES EFFICIENCY OF PROGRAM DEVELOPMENT
- IMPROVES RELIABILITY AND MAINTAINABILITY
- AIDS MAINTENANCE AND TRAINING
- EASES PROGRAM UNDERSTANDING

STABILITY

KRONREF

CROSS REFERENCES SYSTEM SYMBOLS FROM SYSTEM TEXT WITH DECKS FROM A MODIFY PROGRAM LIBRARY AND CROSS REFERENCES COMMON DECKS FROM THAT PL WITH THE DECKS THAT CALL THEM.

LISTED AS OUTPUT OF KRONREF ARE THOSE DECKS THAT REFERENCE:

- PP DIRECT CELLS
- PPR ENTRY POINTS
- MONITOR FUNCTIONS
- CMR POINTERS
- CM LOCATIONS
- CONTROL POINT AREA WORDS
- DAYFILE MESSAGE OPTIONS
- FILE TYPES
- JOB ORIGIN, QUEUE TYPES AND QP
- ERROR FLAGS
- MISC. NOSTEXT SYMBOLS
- COMMON DECK CALLS
- PP ROUTINES CALLED
- SPECIAL ENTRY POINT USAGE
- USE OF SYSTEM MACRO

LOADER TABLES

77	PRFX	PREFIX
70	LDSET	OBJECT DIRECTIVE
60	CAPSULE	RELOCATABLE CAPSULE
54	EACPM	ABS OR OVERLAY W. ECS MULTIPLE ENTRY POINTS
53	ACPM	ABS OR OVERLAY W. ECS
51	EASCM	ABS W. MULTIPLE ENTRY POINTS
50	ASCM	ABS OR OVERLAY
-	6PPM	6000 STYLE PPU PROGRAM BITS 59-42 ARE 3 CHAR NAME OF PROGRAM
70	OPLD	DIRECTORY
76	ULIB	

APPENDIX D OF LOADER MANUAL

APPENDIX G OF REF. MAN. VOL. 2

LOADER TABLES

34	PIDL	PROGRAM LENGTH AND IDENTIFICATION
36	ENTR	ENTRY POINT DEFINITIONS
37	XTEXT	EXTENDED RELOCATABLE TEXT
40	TEXT	RELOCATABLE TEXT
41	XFILL	EXTENDED RELOCATION FILL
42	FILL	RELOCATION FILL
43	REPL	REPLICATION OF TEXT
47	XREPL	EXTENDED REPLICATION OF TEXT
44	LINK	EXTERNAL REFERENCE LINKAGE
45	XLINK	EXTENDED EXTERNAL REFERENCE LINKAGE
46	XFER	TRANSFER POINT

LOADER TABLES

PRFX (77)

<table>
<tr><td>7700</td><td>0016</td><td colspan="6"></td></tr>
<tr><td colspan="4">NAME</td><td colspan="4"></td></tr>
<tr><td colspan="6">DATE</td><td colspan="2"></td></tr>
<tr><td colspan="6">TIME</td><td colspan="2"></td></tr>
<tr><td colspan="8">SYSTEM IDENTIFICATION</td></tr>
<tr><td colspan="5">PROCESSOR</td><td colspan="3">VERSION</td></tr>
<tr><td colspan="3">MODIFICATION LVL</td><td colspan="2">TARGET</td><td colspan="2">VALID</td><td>F</td></tr>
<tr><td>TYPE</td><td colspan="7">HARDWARE REQUIRED</td></tr>
<tr><td colspan="8">COMMENTS</td></tr>
</table>

LOADER TABLES

6PPM

NAME		FWA		WC
PP CODE				

EASCM (51)

5100	L1	L2	FWA	K
ENTRY POINT 1				
⋮				
ENTRY POINT K				
CPU CODE				

ASCM (50)

5000	L1	L2	FWA	ENTRY
CPU CODE				

- LOADER TABLES

RELOCATABLE REL COMCSRT =
SET RECORD TYPE

PRFX

LDSET (OPTIONAL)

PIDL

ENTR TEXT REPL FILL LINK
XTEXT XREPL XFILL XLINK

XFER (OPTIONAL)

ABSOLUTE/OVERLAY ABS
OVL

PRFX

ASCM ACPM
EASCM EACPM

PERIPHERAL PROCESSOR PP

PRFX

6PPM

LOADER TABLES

SYSTEM TEXT	OVL (SPECIAL CASE)
PRFX	
ASCM	LEVELS L1=L2=1
	ORIGIN=ENTRY=0
CAPSULE	CAP
PRFX	
CAPSULE	

LIBRARIES

SYSTEM LIBRARY	SYSEDIT
PROGRAM LIBRARY	MODIFY UPDATE
USER LIBRARY	LIBGEN
	LIBEDIT CATALOG ITEMIZE

LIBRARIES

SYSTEM LIBRARY

- SYSTEM FILE
- DIRECTORIES
    - PLD
    - CLD
    - LBD
- RESIDENCY
    - SYSTEM FILE
    - PPULIB
    - RPL
    - RCL
    - ASR

SYSEDIT

BUILDS DIRECTORY AND RESIDENCY USING ROUTINES FROM THE SYSTEM FILE UNDER DIRECTION OF A LIBDECK

LIBRARIES

PROGRAM LIBRARY

- SOURCE CODE MAINTENANCE
- RECORD TYPES
  - OPL
  - OPLC
  - OPLD
- DIRECTORY SHOWING TYPE AND RANDOM ADDRESS OF ROUTINE WITHIN THE LIBRARY

MODIFY
UPDATE

- BUILD PROGRAM LIBRARY
- COLLECT CODE FROM PROGRAM LIBRARY, ADDING OR DELETING FROM IT, TO COMPILE FILE FOR PROCESSING BY COMPILER OR ASSEMBLER

LIBRARIES

USER LIBRARY

- USED BY LOADER TO SATISFY EXTERNAL REFERENCES - OBJECT TIME ROUTINES
- COLLECTION OF RELOCATABLE SUBROUTINES
- DIRECTORY CONTAINS ENTRY NAME AND RELATIVE ADDRESS WITHIN THE LIBRARY
- FORMAT

LIBRARY NAME	ULIB
ENTRY 1	REL
ENTRY 2	REL
ENTRY 3	REL
LIBRARY DIRECTORY	OPLD

LIBGEN

BUILDS USER LIBRARY FROM A FILE OF RELOCATABLES, PREFIXING THE LIBRARY WITH AN IDENTIFICATION RECORD (ULIB) and APPENDING A DIRECTORY RECORD (OPLD)

LIBRARIES

LIBEDIT

CATALOG/ITEMIZE

GTR

- RECORD TYPES (COMCSRT - SET RECORD TYPE)

  ABS

  CAP

  OPL

  OPLC

  OPLD

  OVL

  REL

  PP

  PPU

  TEXT

  ULIB

  PROC

LIBEDIT MANIPULATES RECORDS ON A LIBRARY FILE (DIRECTORY) EDITING OR REPLACING WITH RECORDS FROM A REPLACEMENT FILE.

CATALOG AND ITEMIZE LIST THE CONTENTS OF A FILE, IDENTIFYING EACH RECORD BY NAME AND TYPE AND LISTING APPROPRIATE INFORMATION ABOUT EACH RECORD.

GTR GETS A RECORD OR RECORDS FROM A FILE BY IDENTIFYING THE RECORD(S) BY TYPE AND NAME.

DSDI

Deadstart Dump Interpreter

Deadstart option dumps hardware registers and memory to a tape

DSDI reads the tape and formats dump data in a more human readable form

DSDI is directive driven -

- control over format of output file
- control over what data is dumped and format it is dumped in
- analysis of system tables
- analysis of major subsystems
    - EI200
    - BATCHIO
    - MAGNET
    - IAF/TELEX

QUESTION SET LESSON 2

1. List the libraries recognized in NOS and the system utilities or mechanisms to manage them.

LESSON 3
CENTRAL MEMORY RESIDENT

LESSON PREVIEW:

This lesson overviews the portion of the NOS operating system that is resident in central memory. The "central memory resident" portion of the system includes all pointers and tables, control point areas, dayfile buffers, PP communication area, PP and CPU routine libraries and directories, and CPUMTR (CPU monitor).

The information in this lesson is organized for a complete overview of medium detail, so that the student has been exposed to the tables and pointers and their contents prior to their use in other lessons.

OBJECTIVES:

Upon completion of this lesson, the student should be able to:

- Trace linkages from table pointers to the actual tables.
- Identify the content of the pointer and constant area (words 0 through 177) of Central Memory Resident.
- Identify the content of the words in the control point area.
- Have familiarity with the:

  Equipment Status Table (EST)

  File Name Table/File Status Table (FNT/FST)

  Mass Storage Table (MST) and Track Reservation Table (TRT)

  Job Control Block (JCB).
- Describe the relationship between the Resident Peripheral Library (RPL), the Resident Central Library (RCL) and the library directories (Peripheral Library Directory (PLD), Central Library Directory (CLD), User Library Directory (LBD)).
- Identify the contents of the words in the Job Communication Area (RA through RA+100) in the job Field Length.

REFERENCES:

NOS System Programmer's Instant, 3 NOS IMS - Chapter 2

CENTRAL MEMORY RESIDENT

CENTRAL MEMORY LAYOUT

000 . . 077	system pointers and control words
100 . . 111	channel status table
112 . . 122	status/control registers
123 . . 126	miscellaneous pointers and data
127 . . 141	reserved
142 . . 177	channel release table
200	control point areas
(n+1)*200	system control point
(n+2)*200	PP communication area (pointer in word 002, byte 4)

dayfile buffer pointers (pointer in word 003, byte 0)
equipment status table (EST) (pointer in word 005, byte 0)
file name/file status table (pointer in word 004, byte 0)
FNT interlock table (pointer in word 004, byte 1)
CDC CYBER 176 exchange package area
mass storage allocation area
mass storage tables (MST)
job control area
dayfile buffers
dayfile dump buffer
ECS/PP buffer
CPUMTR
resident peripheral library (RPL)
resident central library (RCL)
peripheral library directory (PLD)
central library directory (CLD)
system user library directory (LBD)

POINTERS AND CONSTANTS

<table>
<tr><td></td><td>59</td><td></td><td>47</td><td></td><td>35</td><td>29</td><td>23</td><td>17</td><td>11</td><td>5 0</td><td></td></tr>
<tr><td>000</td><td colspan="10">zeros</td><td></td></tr>
<tr><td>001</td><td colspan="4">fwa resident PP library</td><td colspan="2">number of PPUs</td><td colspan="2">†1</td><td colspan="2">memory size/100</td><td>RPLP,PPUL, CPUL,MFLL</td></tr>
<tr><td>002</td><td colspan="4">fwa PP library directory</td><td colspan="2">▨</td><td colspan="2">number of ctrl pts</td><td colspan="2">PP comm area adr</td><td>PLDP,NCPL, PPCP</td></tr>
<tr><td>003</td><td colspan="2">dayfile pntr fwa</td><td colspan="4">fwa dayfile dump buffer</td><td colspan="2">†2</td><td colspan="2">no. excess dayfiles</td><td>DFPP</td></tr>
<tr><td>004</td><td colspan="2">fwa FNT</td><td colspan="2">lwa+1 FNT</td><td colspan="2">▨</td><td colspan="4">fwa job control area</td><td>FNTP,JBCP</td></tr>
<tr><td>005</td><td colspan="2">fwa EST</td><td colspan="2">lwa+1 EST</td><td colspan="2">lwa+1 ms equipment</td><td colspan="4">fwa ECS/PP buffer</td><td>ESTP</td></tr>
<tr><td>006</td><td colspan="4">fwa mass storage allocation</td><td colspan="4">fwa user library directory</td><td colspan="2">▨</td><td>LBDP,MSAP</td></tr>
<tr><td>007</td><td colspan="4">fwa CPU library directory</td><td colspan="4">fwa COS format CPU lib directory</td><td>▨</td><td>†3</td><td>CLDP</td></tr>
<tr><td>010
.
.
017</td><td colspan="10">installation area</td><td>INOL,INSL
.
.
IN7L</td></tr>
<tr><td>020</td><td colspan="8">▨</td><td colspan="2">CMR size /100</td><td>CMRL</td></tr>
<tr><td>021</td><td colspan="7">system name</td><td colspan="2">▨</td><td>†4</td><td></td></tr>
<tr><td>022</td><td colspan="5">▨</td><td colspan="4">job sequence number counter</td><td>▨</td><td>JSNL</td></tr>
<tr><td>023</td><td colspan="2">▨</td><td colspan="2">avail ECS 1000B blocks</td><td colspan="4">▨</td><td colspan="2">available mem/100B</td><td>ACML,AECL</td></tr>
<tr><td>024</td><td colspan="2">job scheduler</td><td colspan="2">CPU recall</td><td colspan="2">PP/auto recall</td><td colspan="2">job activity</td><td colspan="2">job switch</td><td>MSCL</td></tr>
<tr><td>025</td><td colspan="2">†5</td><td colspan="2">ECS first user track</td><td colspan="2">user 1000B word ECS blks</td><td colspan="2">ECS RA/1000B for CPO</td><td colspan="2">ECS FL/1000B for CPO</td><td>ECRL</td></tr>
<tr><td>026</td><td colspan="5">▨</td><td colspan="5">julian date (yyddd)</td><td>JDAL</td></tr>
<tr><td>027</td><td colspan="4">▨</td><td colspan="6">packed date (yr - 1970,mo,da,hr,mn,sc)</td><td>PDTL</td></tr>
<tr><td>030</td><td colspan="10">time of day (Δhh.mm.ss.)</td><td>TIML</td></tr>
<tr><td>031</td><td colspan="10">date (Δyy/mm/dd.)</td><td>DTEL</td></tr>
<tr><td>032
.
.
035</td><td colspan="10">system title line</td><td></td></tr>
<tr><td>036
037</td><td colspan="10">system version name</td><td></td></tr>
<tr><td>040</td><td colspan="8">▨</td><td colspan="2">scheduler cy. intvl.</td><td>JSCL</td></tr>
<tr><td>041</td><td colspan="2">†6 ▨</td><td colspan="4">1CK recall time</td><td colspan="4">1SP recall time</td><td></td></tr>
</table>

	59	47	35	23 17	11 0	
042	†1					IPRL
043	†2					SSTL
044	TELEX/IAF	EXPORT/ IMPORT	BATCHIO	MAGNET	TAF	SSCL
045	STIMULATOR	NETWORK INTER PROC	RBF	CDCS	MCS	
046	MASS STORAGE CONTROL	TRANSACTION STIMULATOR	reserved			
047	reserved					
050	reserved				IR addr next PPU	PPAL
051	idle time					
052 053 054	load code for MS error processors					MSEL
055 056	reserved					
057	ctrl point for move	internal to MTR				CMCL
060	←†3 ▨	▨	CPO ctrl pt assig	CPO exchange address		ACPL
061	←†4 ▨	▨	CPI ctrl pt assig	CPI exchange address		
062	▨	▨	▨	address of PPO exchange package		PXPP
063	first word of PP exchange package					
064 065	reserved					
	zeros					ZERL
067 . . 075	reserved					
076	reserved			CPUMTR exchange address for MTR		MTRL
077	EQ CPSL		PS O			CPSL

<table>
<tr><th></th><th>59</th><th>47</th><th colspan="2">35</th><th>23</th><th>17</th><th>11</th><th>0</th></tr>
<tr><td>100</td><td>CH0</td><td>CH1</td><td colspan="2">CH2</td><td colspan="2">CH3</td><td>CH4</td><td>CTIL †1</td></tr>
<tr><td>101</td><td>CH5</td><td>CH6</td><td colspan="2">CH7</td><td colspan="2">CH10</td><td>CH11</td><td></td></tr>
<tr><td>102</td><td>CH12</td><td>CH13</td><td colspan="2">CH14</td><td colspan="2">CH15</td><td>CH16</td><td></td></tr>
<tr><td>103</td><td>CH17
(unused)</td><td>CH20</td><td colspan="2">CH21</td><td colspan="2">CH22</td><td>CH23</td><td></td></tr>
<tr><td>104</td><td>CH24</td><td>CH25</td><td colspan="2">CH26</td><td colspan="2">CH27</td><td>CH30</td><td></td></tr>
<tr><td>105</td><td>CH31</td><td>CH32</td><td colspan="2">CH33</td><td colspan="2">CH34
(unused)</td><td>CH35
(unused)</td><td></td></tr>
<tr><td>106</td><td colspan="2">seconds</td><td colspan="5">milliseconds</td><td>RTCL</td></tr>
<tr><td>107</td><td colspan="7">reserved</td><td></td></tr>
<tr><td>110</td><td colspan="7">†2</td><td>PFNL</td></tr>
<tr><td>111</td><td colspan="2">†3</td><td colspan="5">▨</td><td></td></tr>
<tr><td>112</td><td colspan="7">†4</td><td>SCRL</td></tr>
<tr><td>113</td><td>4</td><td>3</td><td colspan="2">2</td><td colspan="2">1</td><td>0</td><td>S16L †5</td></tr>
<tr><td>114</td><td>9</td><td>8</td><td colspan="2">7</td><td colspan="2">6</td><td>5</td><td></td></tr>
<tr><td>115</td><td>14</td><td>13</td><td colspan="2">12</td><td colspan="2">11</td><td>10</td><td></td></tr>
<tr><td>116</td><td colspan="4">▨</td><td colspan="2">16</td><td>15</td><td></td></tr>
<tr><td>117</td><td>4</td><td>3</td><td colspan="2">2</td><td colspan="2">1</td><td>0</td><td>S36L †6</td></tr>
<tr><td>120</td><td>9</td><td>8</td><td colspan="2">7</td><td colspan="2">6</td><td>5</td><td></td></tr>
<tr><td>121</td><td>14</td><td>13</td><td colspan="2">12</td><td colspan="2">11</td><td>10</td><td></td></tr>
<tr><td>122</td><td colspan="4">▨</td><td colspan="2">16</td><td>15</td><td></td></tr>
<tr><td>123</td><td>MID</td><td colspan="5">†7</td><td>machine
index</td><td>MMFL</td></tr>
<tr><td>124</td><td colspan="7">reserved</td><td></td></tr>
<tr><td>125</td><td colspan="7">reserved</td><td></td></tr>
<tr><td>126</td><td colspan="5">reserved</td><td colspan="2">flag register</td><td>EFRL</td></tr>
<tr><td>127</td><td colspan="7">†8</td><td>INWL</td></tr>
<tr><td>130</td><td colspan="2">reserved</td><td colspan="2">MXN time</td><td colspan="2">worst case
MTR cycle time</td><td>current MTR
cycle time</td><td>SD0L</td></tr>
<tr><td>131</td><td colspan="3">count of ECS moves</td><td colspan="4">count of CM moves</td><td>SD1L</td></tr>
<tr><td>132</td><td colspan="2">rollout count</td><td colspan="5">count of sectors rolled</td><td>SD2L</td></tr>
<tr><td>133</td><td>reserved</td><td colspan="3">user commits + time slice with output</td><td colspan="3">count of time slices</td><td>SD3L</td></tr>
<tr><td>134</td><td colspan="3">reserved</td><td colspan="4">jobs in recall due to PP priority exchanges</td><td>SD4L</td></tr>
<tr><td>135
⋮
162</td><td colspan="7">reserved</td><td></td></tr>
<tr><td>163
⋮
177</td><td colspan="7">DSD - 1DS communication area</td><td></td></tr>
</table>

<table>
<tr><th></th><th colspan="4">59</th><th colspan="2">47</th><th colspan="2">41</th><th colspan="2">35</th><th colspan="2">29</th><th colspan="2">23</th><th colspan="2">17</th><th colspan="2">11</th><th>5</th><th>0</th><th></th></tr>
<tr><td>000
017</td><td colspan="20">exchange package area</td><td></td></tr>
<tr><td>020</td><td colspan="4">†1</td><td colspan="4">error flags</td><td colspan="4">activity count</td><td colspan="4">RA/100B</td><td colspan="4">FL/100B</td><td>STSW</td></tr>
<tr><td>021</td><td colspan="14">job name</td><td colspan="2">job orgn</td><td colspan="4">operator equipment</td><td>JNMW,OAEW</td></tr>
<tr><td>022</td><td colspan="4">CPU priority</td><td colspan="4">queue priority</td><td colspan="4">†2</td><td colspan="2">▨</td><td colspan="6">CPUs allowable</td><td>JCIW</td></tr>
<tr><td>023</td><td colspan="8">CM residence time limit</td><td colspan="2">†3</td><td colspan="10">CPU time slice limit</td><td>TSCW</td></tr>
<tr><td>024</td><td colspan="20">time entered X status</td><td>CPCW</td></tr>
<tr><td>025</td><td colspan="4">†4</td><td colspan="8">reserved</td><td colspan="4">ECS RA/1000B</td><td colspan="4">ECS FL/1000B</td><td>ECSW,CPIW</td></tr>
<tr><td>026</td><td colspan="20">PP recall register</td><td>RLPW</td></tr>
<tr><td>027</td><td colspan="16">†5</td><td colspan="2">snse swchs</td><td colspan="2">▨</td><td>SNSW</td></tr>
<tr><td>030
●
●
●
034</td><td colspan="20">message 1 area</td><td>MS1W</td></tr>
<tr><td>035
036
037</td><td colspan="20">message 2 area</td><td>MS2W</td></tr>
<tr><td>040
●
●
●
047</td><td colspan="20">installation area</td><td>INOW
●
●
●
IN7W</td></tr>
<tr><td>050</td><td colspan="6">†6</td><td colspan="14">SRU accumulator (micro units *10)</td><td>ACTW,SRUW</td></tr>
<tr><td>051</td><td colspan="20">CP accumulator</td><td>CPTW</td></tr>
<tr><td>052</td><td colspan="7">MS accumulator</td><td colspan="6">MT accumulator</td><td colspan="7">PF accumulator</td><td>IOAW</td></tr>
<tr><td>053</td><td colspan="6">M13=M1*M3</td><td colspan="6">M14=M1*M4</td><td>▨</td><td colspan="7">adder accumulator</td><td>MP1W,ADAW</td></tr>
<tr><td>054</td><td colspan="6">M1*1000</td><td colspan="6">M12=M1*M2</td><td colspan="8">reserved</td><td>ACTWE,MP2W</td></tr>
<tr><td>055</td><td>←</td><td colspan="9">†7 CPM (SRU=SRU+ CPM*CP)</td><td colspan="10">IOM (SRU=SRU+IOM*IO)</td><td>MP3W</td></tr>
<tr><td>056</td><td colspan="6">SRU account block limit</td><td colspan="14">computed SRU job step limit</td><td>STLW</td></tr>
<tr><td>057</td><td colspan="4">reserved</td><td colspan="6">SRU job step limit</td><td colspan="10">SRU at beginning of job step</td><td>SRJW</td></tr>
<tr><td>060</td><td colspan="4">reserved</td><td colspan="6">CP time job step limit</td><td colspan="10">CP time at beginning of job step</td><td>CPJW</td></tr>
</table>

	59	53	47	35	29	23	17	11	0	
061	†1									FPFW
062	†2					rollin FL		FL increase request		FLCW
063	†3					rollin ECS FL		ESC FL increase req		ELCW
064	†4									SSCW
065	TXOT	list of files address		TTY interrupt address †5			output pointer			TXSW,TIOW, TIAW,LOFW
066	auxiliary pack name						†6			PFCW
067	user number †9						← †7 user index			UIDW
070	†8			†11 terminal input pointer			error exit †10 return address			EECW,TINW
071	input FST		primary FST	▨	event descriptor			rollout time		TFSW,TERW
072	†12		control statement count			next statement index		limit index		CSPW
073	†13	eq num	first track	current track		current sector		half sector flag		CSSW
074	job sequence number			control statement address (TCS)			demand file random index			RFCW
075	reserved		†14							ALMW
076	reserved		dayfile msg count	control stmt count		†15	mass storage PRU count			ACLW
077	each bit has a special meaning									AACW
100	buffer 0 length		buffer 0 address		buffer 1 length		buffer 1 address			ICAW
101	special entry point word †16									SEPW
102	system processor call word †17									SPCW
103	EFG	R1G		CCL data			reserved			JCDW
104	EF	R3		R2			R1			JCRW
105	†18	input buffer address		right screen buffer address			left screen buffer address			DBAW
106 107 110	loader control words †19									LB1W LB2W LB3W
111	▨		†20				FWA of dump			PPDW
112	reserved							†21		SSOW
113	computed CP job step limit									CPLW
114 • • • 127	reserved									
130 • • • 177	control statement buffer									CSBW

PP COMMUNICATION AREA

	59	47	41	35	0	
INP REG	name of PP program		† 1	parameters		IA
OUT REG	monitor fnct code	parameters				OA
	message buffer (6 words)					MA

DAYFILE BUFFER POINTERS

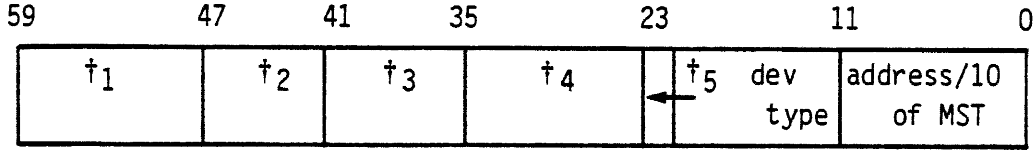
59	47	35	23	11	0
fwa dayfile buffer		no. words in buffer	length of buffer	† 2	
eq no	first track	current track	current sector	▨	

REF	Bit No.	Description
†1	41	Set if called with auto recall.
	40-36	Control point assignment.
†2	11-0	Interlock byte (0 = no dump in progress, 1 = dump in progress).

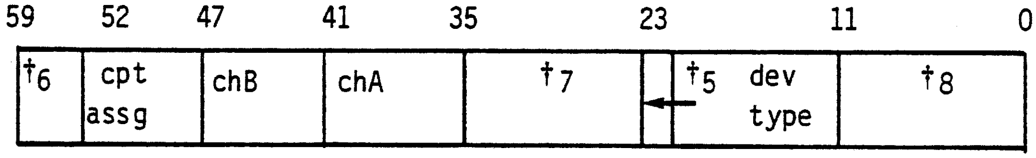
CENTRAL MEMORY TABLES

Equipment Status Table (EST) Formats

Mass Storage Device



Nonmass Storage Device (3000 Type Equipment)



File Name/File Status Table (FNT/FST) Entry

File in Input Queue

59	53	47	35	23	17	11	5	0
job name					job org	type INFT	†1	
id code	eq no	first track	binary card sequence no	field length		queue priority		

File in Print Queue

59	53	47	35	17	11	5	0
job name				job org	type PRFT	†1	
†2	eq no	first track	†3		queue priority		

File in Punch Queue

59	53	47	35	17	11	5	0
job name				job org	type PHFT	†1	
†2	eq no	first track	†3		queue priority		

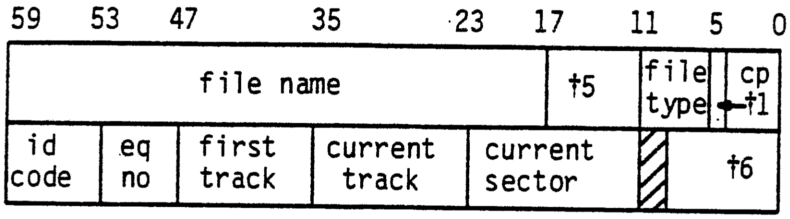
File in Rollout Queue

59	53	47	35	23	17	11	5	0
job name					job org	type ROFT	†4	
id code	eq no	first track	ECS FL/1000B no	field length		queue priority		

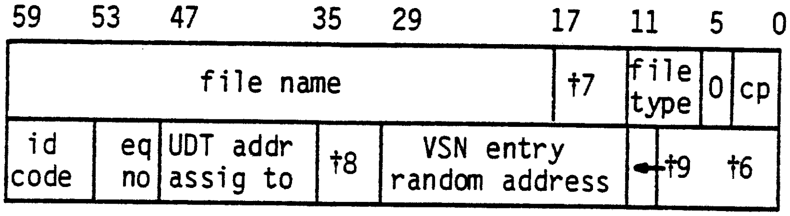
File in Time/Event Rollout Queue

59	53	47	35	23	17	11	5	0
job name					job org	type TEFT	†4	
event des	eq no	first track	event descriptor	field length		rollout time pd		

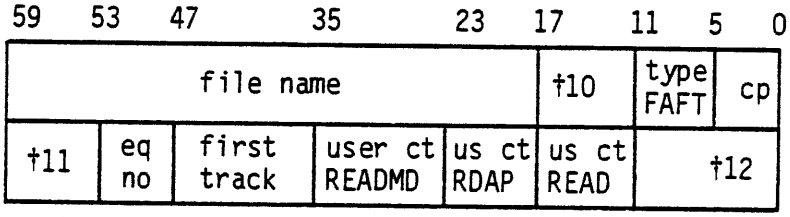
Mass Storage Files
Not in Input, Print, Punch, or Rollout Queue



Magnetic Tape Files



Fast Attach Permanent Files



Ref	Bit No.	Description
†1	5	Set if system sector contains control information.
†2	59-57	Device selection field.
	56-54	External characteristics.
†3	35-33	Forms code.
	32-12	Terminal identification (TID).
†4	5	Set if user job has subsystem connection (either long term connection or wait response).
†5	17	Unused.
	16	Set if extend-only file.
	15	Set if alter-only file.
	14	Set if execute-only file.
	13	Unused.
	12	Write lockout.
†6	10	Unused.
	9	Indicates the track interlock status of LIFT files (mass storage only).
	8	Set if file is opened.
	7	Set if file is written since last open.
	6	Set if file is written on.
	5-4	Unused.
	3-2	Read status (0 = incomplete read, 1 = EOR, 2 = EOF, 3 = EOI).
	1	Set if last operation write.
	0	Clear if busy status.

Mass Storage Allocation (MSA) Area

	59	47 0
000	last temp eq	temporary devices[†]
001	last input eq	input file devices[†]
002	last output eq	output file devices[†]
003	last rollout eq	rollout file devices[†]
004	last dayfile eq	user dayfile devices[†]
005	last primary eq	primary file devices[†]
006	last local eq	local file devices[†]
007	last LGO eq	LGO file devices[†]
008	last secondary rollout eq	secondary rollout file devices[†]

[†] Bit 47-eq is set for each equipment with the allocation type selected.

Mass Storage Table (MST)

<table>
<tr><th></th><th>59</th><th>51</th><th>47</th><th>40</th><th>35</th><th>23</th><th>17</th><th>11</th><th>5</th><th>0 TDGL</th></tr>
<tr><td>000</td><td colspan="2">$\dagger_1$</td><td colspan="2">▨</td><td>TRT length</td><td colspan="2">$\dagger_2$</td><td colspan="2">no. avail. tracks</td><td>TDGL</td></tr>
<tr><td>001</td><td colspan="2">$\dagger_3$</td><td colspan="2">user ECS first track</td><td>file count</td><td colspan="2">IQFT track</td><td colspan="2">$\dagger_4$</td><td>ACGL</td></tr>
<tr><td>002</td><td colspan="4">ECS address of MST/TRT</td><td colspan="4">ECS MST/TRT update cnt</td><td>$\dagger_5$</td><td>SDGL</td></tr>
<tr><td>003</td><td colspan="2">1st track IAF</td><td colspan="2">label track</td><td>permits track</td><td colspan="2">no. catalog tracks</td><td colspan="2">DAT track</td><td>ALGL</td></tr>
<tr><td>004</td><td colspan="6">family or pack name</td><td>DN</td><td>▨</td><td>$\dagger_6$</td><td>PFGL</td></tr>
<tr><td>005</td><td colspan="6">user number for private pack</td><td colspan="3">$\dagger_7$</td><td>PUGL</td></tr>
<tr><td>006</td><td colspan="4">$\dagger_8$</td><td>driver name</td><td colspan="2">0</td><td colspan="2">sector limit</td><td>MDGL</td></tr>
<tr><td>007</td><td colspan="9"></td><td>R1GL</td></tr>
<tr><td>010</td><td colspan="9">installation area (global)</td><td>ISGL</td></tr>
<tr><td>011</td><td colspan="9"></td><td>I2GL</td></tr>
<tr><td>012</td><td colspan="2">activity count</td><td colspan="2">unit interlocks</td><td>current position</td><td colspan="2">MTR internal</td><td colspan="2">ECS error #</td><td>DALL</td></tr>
<tr><td>013</td><td colspan="9">$\dagger_9$</td><td>DILL</td></tr>
<tr><td>014</td><td colspan="2">DAYFILE track</td><td colspan="2">ACCOUNT track</td><td>ERRLOG track</td><td colspan="2">system table track</td><td colspan="2">$\dagger_{10}$</td><td>DULL</td></tr>
<tr><td>015</td><td colspan="5">$\dagger_{11}$</td><td colspan="2">user count</td><td colspan="2">$\dagger_{12}$</td><td>STLL</td></tr>
<tr><td>016</td><td colspan="9">$\dagger_{13}$</td><td>DDLL</td></tr>
<tr><td>017</td><td colspan="9">installation area</td><td>ISLL</td></tr>
</table>

Track Reservation Table (TRT)

Word Format

59	47	35	23	11	0
track link	track link	track link	track link	†1	

REF	Bit No.	Description
†1	11-8	Each bit set indicates corresponding byte (0 through 3) is first track of a preserved file.
	7-4	Track interlock bits.
	3-0	Track reservation bits.

Track Link Byte (Format 1)

Bit	Contents
11	Set.
10-0	Next track in track chain.

Track Link Byte (Format 2)

Bit	Contents
11	Clear.
10-0	End of chain (EOI sector in file).

Track Link Byte (Format 3)

Bit	Contents
3777	Track is flawed.

Machine Recovery Table (MRT)

Word Format

59	31 0
unused	†1

REF	Bit No.	Description
†1	31-0	Each bit represents one logical track (bits 10-5 of the logical track number denote the word number in the MRT and bits 4-0 are the bit numbers within the word).

The meaning of the MRT bit depends upon the state of the track interlock bit in the TRT.

Track Interlock Bit	MRT Bit	Description
0	0	Track is not interlocked or it is local to another machine.
0	1	First track of a file is local to this machine.
1	0	Track is interlocked by another machine.
1	1	Track is interlocked by this machine.

Job Control Area (JCB)

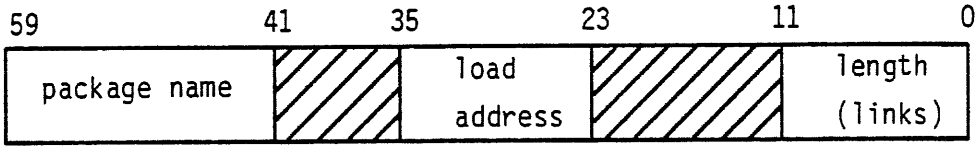
One for each origin type and job class

59	47	35	23	11 0	
in. queue priority	lower bound	upper bound	priority age intvl	cur. intvl count	INQT
in. queue priority	lower bound	upper bound	priority age intvl	cur. intvl count	ROQT
in. queue priority	lower bound	upper bound	priority age intvl	cur. intvl count	OTQT
init. CPU priority	CPU time slice	CM time slice	▨	▨	SVJT
max jobs or users	max FL any job	max FL all jobs	max ECS FL any job	max ECS FL all jobs	
†1	reserved				PFCT
reserved					ETB
▨	▨	▨	▨	▨	

REF	Bit No.	Description
†1	59-48	Index into tables of limits.
	59-57	Index a table of limits for size of each direct access file.
	56-54	Index a table of limits for number of permanent files.
	53-51	Index a table of limits for cumulative size of indirect access files.
	50-48	Index a table of limits for size of each indirect access file.

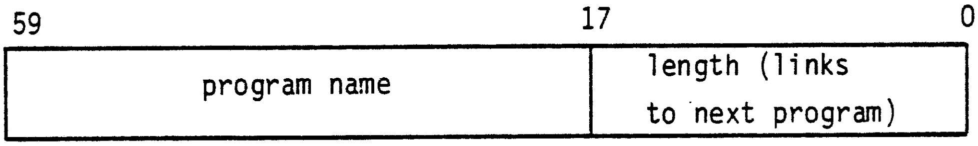
RESIDENT LIBRARIES

Resident PPU Library (RPL)

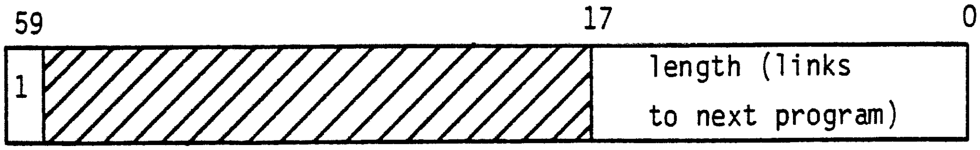


Resident CPU Library (RCL)

Type OVL



Type ABS



DIRECTORIES

PPU Library Directory (PLD)

CM Resident

```
59                41   35                23             11          0
+-----------------+--+-------------------+--------------+-------------+
| package name    |1 | RPL address       |    length    |    load     |
|                 |  |                   |              |   address   |
+-----------------+--+-------------------+--------------+-------------+
```

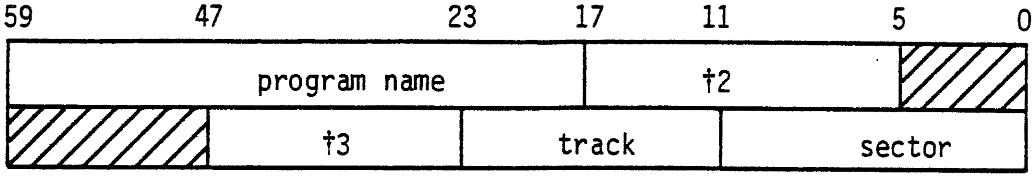
Non-CM Resident

```
59                41        35           23             11          0
+-----------------+---------+------------+--------------+-------------+
| package name    |   †1    |   track    |    sector    |    load     |
|                 |         |            |              |   address   |
+-----------------+---------+------------+--------------+-------------+
```

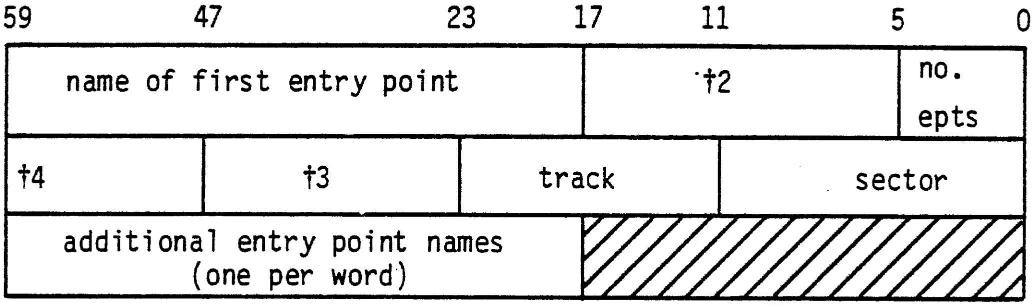
DIRECTORIES

CPU Library Directory (CLD)

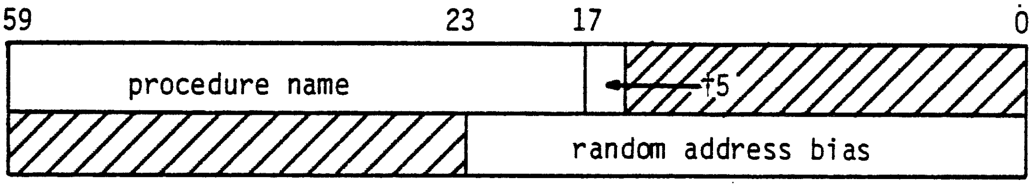
Type OVL



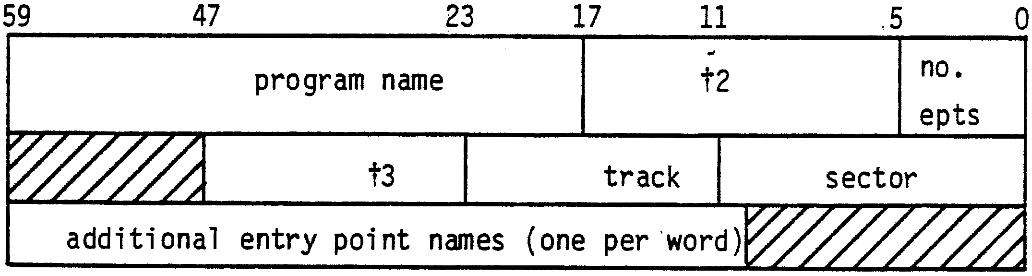
Type ABS



Typer PROC

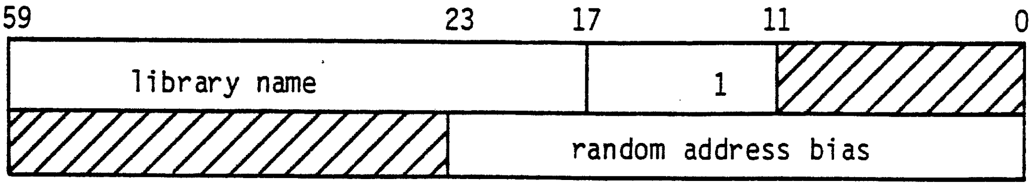


Type REL

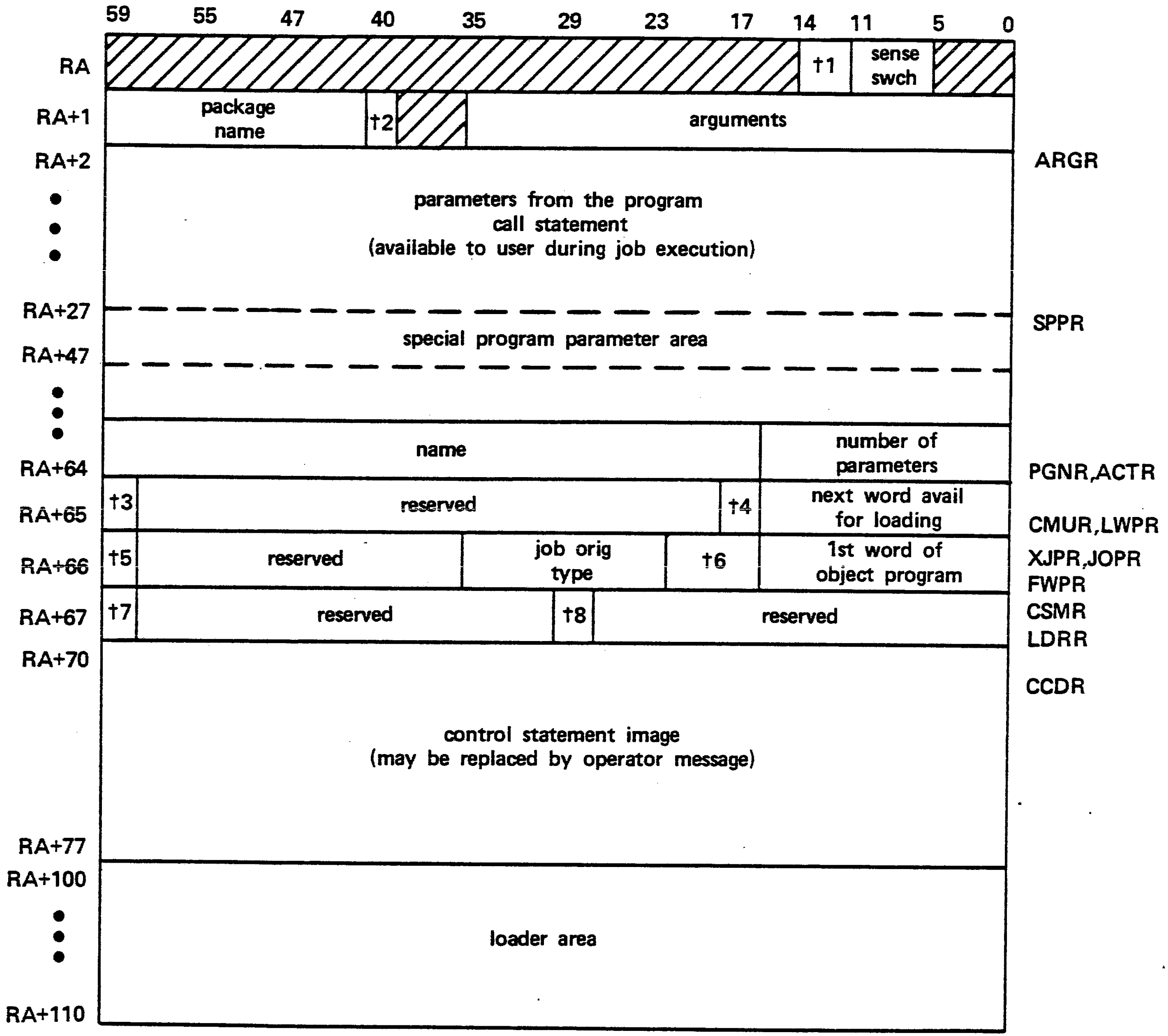


User Library Directory (LBD)

Type ULIB



JOB COMMUNICATION AREA



Ref	Bit No.	Description
†1	14	CFO bit if console forced operator command is allowed.
	13	Subsystem idledown flag.
	12	Pause flag.
†2	40	Auto recall.
†3	59	Set if compare/move unit (CMU) is present.
†4	18	Set if load from system library.
†5	59	Set if CEJ/MEJ option is available.
†6	23-20	Reserved.
	19	Set if program called from DIS.
	18	RSS bit.
†7	59	Set indicates system is in 64-character set mode.
†8	29	Set if load has completed.

QUESTION SET LESSON 3

Obtain a dump of the current system CMR (Central Memory Resident) or use the study dump provided with this handout. (This study dump was obtained during the investigation of a problem that occurred on the inhouse version of NOS 1.4; this dump is being used as the study dump since it has many interesting configuration properties.) Answer the following questions.

1. For this system:
    a. How many PPs?
    b. How many control points?
    c. How much central memory?
    d. Does the CEJ/MEJ option exist?
    e. How long (in words) is CMR?

2. Is there a PP program running in PP3? In PP4? If so, what is the name of the program?

3. Are any of the PPs making monitor requests? If so, which PPS and what are the requests?

4. Describe the peripheral equipment configuration for this system.

5. On what channels are the tape drives and unit record equipment?

6. Locate the system file; i.e., what equipment(s) and which track does the file reside on?

7. Identify the file name, file type, control point assignment, equipment, and file position for the files located at FNT locations:

    6070 6110 6236 6410 6414

8. How much central memory is available?

9. What is the job switch delay?

10. What is the time of day?

11. To which control point(s) is (are) the CPU(s) currently assigned?

12. What is the original input queue priority for a job of batch origin?

13. What is the CPU priority for a remote batch origin job?

14. What are the ROLLOUT queue priorities for time-sharing origin jobs?

15. CIO is a PP program residing in the RPL (Resident Peripheral Library). Find its RPL entry.

16. What is the name and length of the program following CIO in the RPL?

17. What are the base addresses of the PLD (Peripheral Library Directory), the RCL (Resident Central Library), and CLD (Central Library Directory)?

    What are the names of the first entries in each Library or Directory?

18. Where does the system dayfile reside in CMR? Where does the dayfile dump buffer reside in CMR?

19. Which PP has reserved channel 5? Channel 10?

20. What is the first unavailable channel?

21. The following questions refer to control point 12 in the study dump. (If you obtain your own dump, pick any control point that is actively running a job.)

    a. What is the jobname and job origin?
    b. What is the control point status?
    c. How many PPs are assigned? Which ones are they and what function is being performed?
    d. What are the control point RA and FL.
    e. What are the CPU and QUEUE priorities?
    f. How much CPU time has been accumulated?
    g. What is the family, user number, user index for the job?
    h. What is the value of the mass storage acculumator?
    i. What is the first control card in the control card buffer?
    j. What is the next control card to be executed?

22. Which PPs have been locked out (turned off) by the system and how did you arrive at your answer?

23. Why can't the FNT start or end beyond location 4096D?

LESSON 4
EXCHANGE JUMP

LESSON PREVIEW:

This lesson covers the use of the exchange jump hardware by the NOS Operating System. The student will learn what an exchange jump is, how the exchange address is determined, the four types of exchange jumps done by NOS, and how exchange packages are managed by the system.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Explain how the exchange jump hardware instruction works.
- Explain how the exchange address is computed in monitor mode and in program mode.
- Describe in detail each of the four modes of exchange done in NOS; namely,

  MTR exchange

  Pool PP exchange

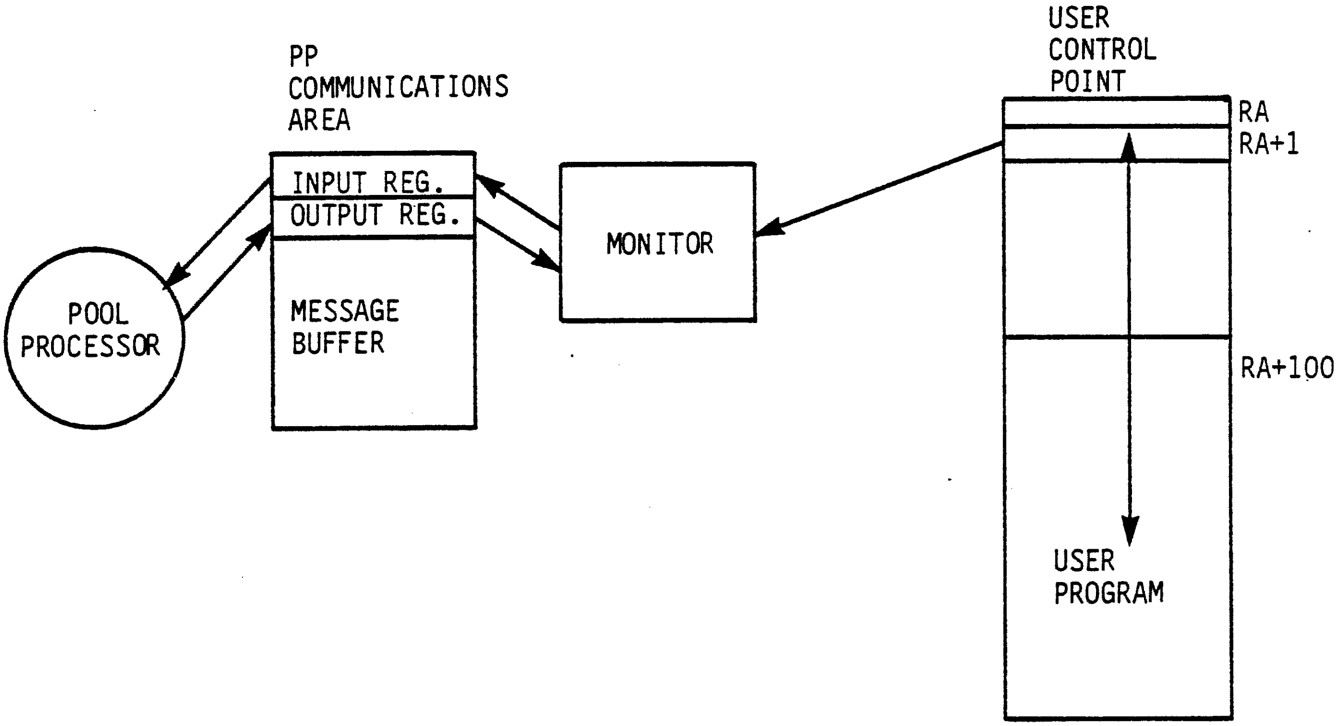
  CPU user program exchange

  Program mode CPUMTR exchange
- Describe in detail how exchange packages are managed when a PP request causes a different CPU program to be brought into execution, rather than resuming the execution of the program that was interrupted by the exchange.

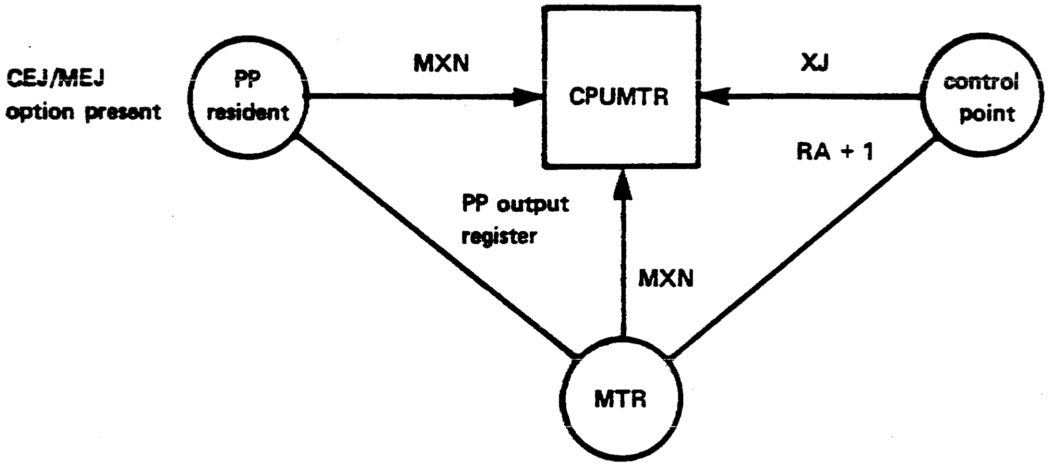
REFERENCES:

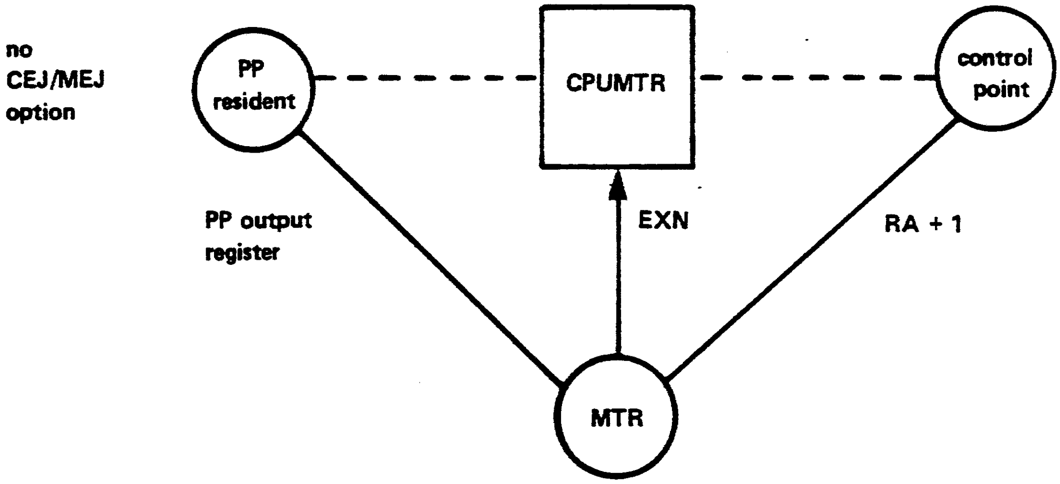
CYBER 170 Hardware Manual, 5-1-7 NOS IMS - Chapter 3

SYSTEM INTERACTION



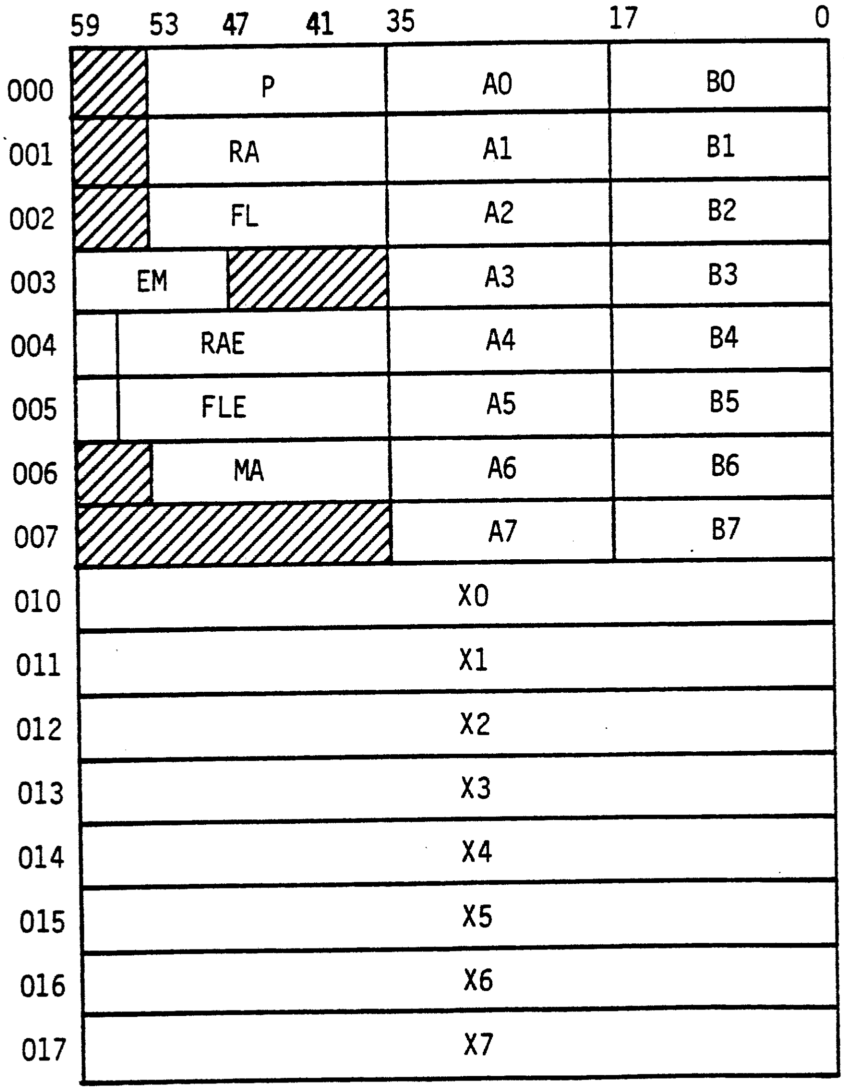
MONITORS INTERACTION





EXCHANGE PACKAGE AREA

Exchange package area for CDC CYBER 170 Series, Models 171, 172, 173, 174,175, 720, 730, 750, and 760; CDC CYBER 70 Series, Models 71, 72, 73, and 74; and CDC 6000 Series Computer Systems.



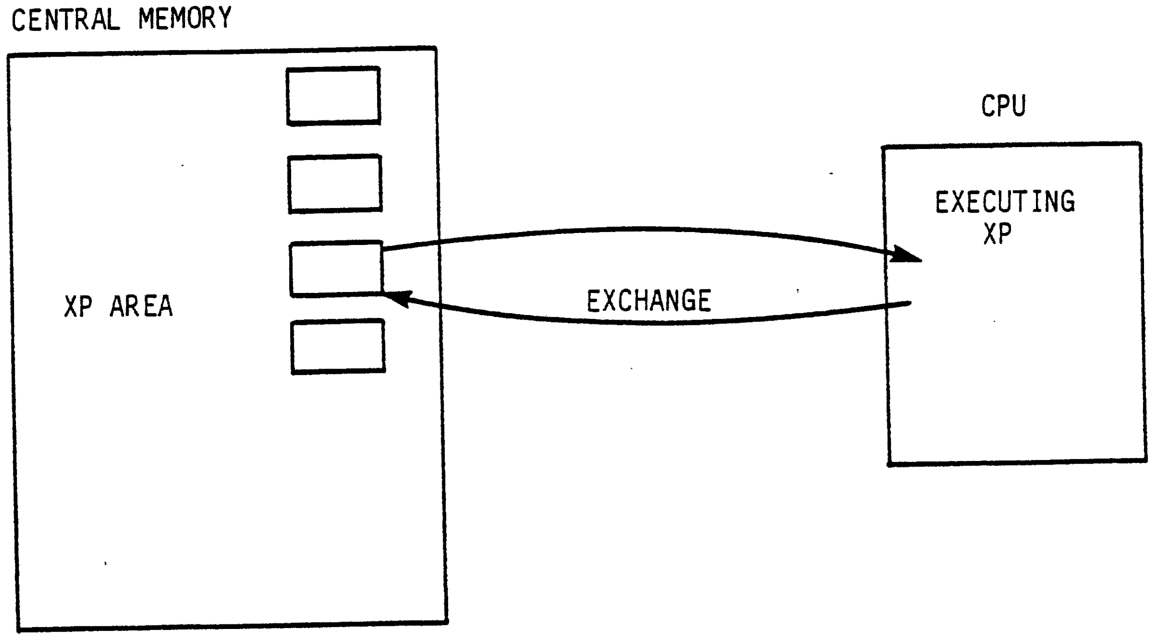
Exchange package area for CDC CYBER 170 Series, Model 176 Computer Systems.

	59	53	35	17	0
000	▨	P	A0	B0	
001	▨	RA	A1	B1	
002	▨	FL	A2	B2	
003	▨	PSD	A3	B3	
004	▨	RAE	A4	B4	
005	▨	FLE	A5	B5	
006	▨	NEA (MA)	A6	B6	
007	▨	EEA	A7	B7	
010	X0				
011	X1				
012	X2				
013	X3				
014	X4				
015	X5				
016	X6				
017	X7				

EXCHANGE JUMP

Take the XP (exchange package) at this address and put it into the CPU hardware registers.

Take the CPU hardware registers and put them in exchange package format at this address.



EXCHANGE JUMP

MONITOR ADDRESS REGISTER (MA)

DEFINES STARTING ADDRESS OF XP FOR EXCHANGE THAT OCCURS WHEN NOT IN MONITOR MODE.

MONITOR FLAG BIT

HARDWARE "FLAG" THAT INDICATES WHETHER CPU IS INTERRUPTABLE (PROGRAM MODE) OR NOT INTERRUPTABLE (MONITOR MODE).

THE MONITOR FLAG ALSO DETERMINES HOW XJ INSTRUCTION IS TO BE EXECUTED.

XJ INSTRUCTION

```
013      XJ    Bj+K
```

IF CPU IS IN MONITOR MODE, THE EXCHANGE IS MADE WITH THE ABSOLUTE ADDRESS Bj+K.

IF CPU IS IN PROGRAM MODE, THE EXCHANGE IS MADE WITH THE ABSOLUTE ADDRESS CONTAINED IN MA.

AN EXCHANGE TOGGLES THE MONITOR FLAG.

EXCHANGE JUMP

MXN INSTRUCTION

```
261        MXN   CP
```

IF CPU IS IN MONITOR MODE, THE EXCHANGE IS BLOCKED AND TREATED AS A PASS (PSN).

IF CPU IS IN PROGRAM MODE, THE EXCHANGE IS MADE WITH THE ABSOLUTE ADDRESS CONTAINED IN THE A REGISTER.

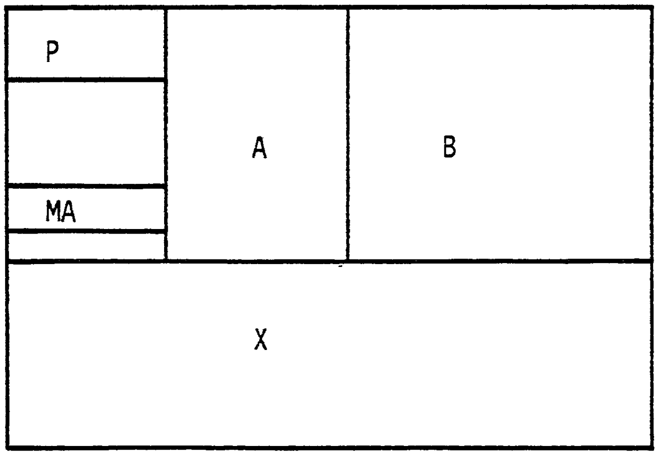
EXCHANGE JUMP

INSTRUCTION	MONITOR	PROGRAM
MXN PPU	PASS	EXCHANGE WITH ADDRESS FROM A REGISTER
XJ CPU	EXCHANGE WITH ADDRESS Bj+K 013jk kkkkk	EXCHANGE WITH ADDRESS FROM MA OF CURRENTLY EXECUTING XP

FOR EACH
CONTROL POINT

(MA) = CONTROL
POINT
ADDRESS

THEREFORE (MA)
OF EXECUTING
CONTROL POINT XP
IS THAT CONTROL POINT'S
ADDRESS



EXCHANGE JUMP

FOUR TYPES OF EXCHANGE JUMP

- MTR

  PPU MTR EXCHANGES WITH CPUMTR

- POOL PPU

  A PPU PROGRAM EXCHANGES WITH CPUMTR

- CPU PROGRAM

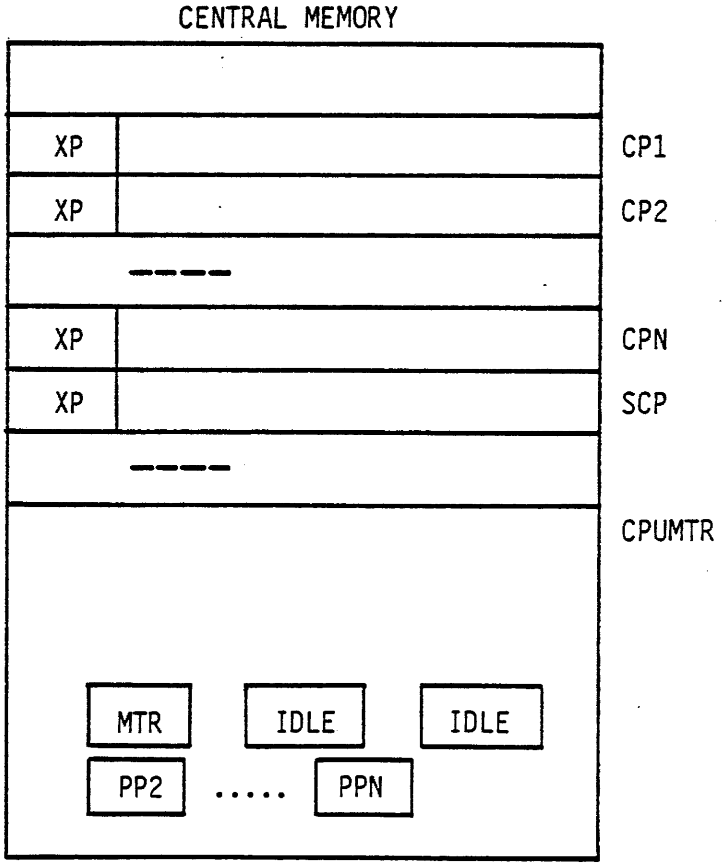
  A PROGRAM AT A CONTROL POINT ACTIVE IN THE CPU EXECUTES AN XJ TO EXCHANGE WITH CPUMTR

- SYSTEM CONTROL POINT

  CPUMTR (IN PROGRAM MODE) EXECUTES AN XJ TO EXCHANGE WITH MONITOR MODE CPUMTR

EXCHANGE PACKAGES

- MTR
- POOL PPUs (2-n)
- IDLE PACKAGE FOR EACH CPU
- EACH CONTROL POINT INCLUDING THE SYSTEM CONTROL POINT



EXCHANGE JUMP

MTR

1. MTR HAS ITS OWN EXCHANGE PACKAGE LOCATED WITHIN CPUMTR.

2. MTR SETS UP TO DO EXCHANGE -

   (P) = PMN   ADDRESS IN CPUMTR WHERE EXECUTION IS TO BEGIN

   (B2) = ADDRESS OF MTR'S EXCHANGE PACKAGE WITHIN CPUMTR

   (X0) = REQUEST TO BE PROCESSED BY CPUMTR

   (B0) ≠ 0   MXN SUCCESSFUL FLAG

3. MTR PERFORMS INSTRUCTIONS -

   LDC ADDRESS OF MTR'S XP

   MXN   CP

   TO EXCHANGE CPU cp (cp=0 FOR CPU 0,
                                        1 for CPU 1)

   WITH THE XP FOR MTR

EXCHANGE JUMP

MTR

4. MTR READS ITS XP'S FIRST WORD.
   IF (B0)=0, THEN THE MXN WAS SUCCESSFUL (CPU WAS IN PROGRAM MODE). OTHERWISE, MTR REPEATS THE EXCHANGE (STEP 3) UNTIL SUCCESSFUL.

5. CPUMTR BEGINS EXECUTION AT PMN TO PROCESS THE REQUEST IN X0.

6. WHEN COMPLETED, CPUMTR EXITS DOING

   ```
   XJ   B2
   ```

   WHICH CAUSES MTR REQUEST PROCESSING TO COMPLETE AND THE XP PREVIOUSLY IN EXECUTION TO BE BROUGHT BACK INTO THE CPU AND RESUME EXECUTION.

7. WHEN MTR SENSES ITS EXCHANGE WAS SUCCESSFUL, IT RESUMES ITS PROCESSING.

   A NEW EXCHANGE REQUEST WILL NOT BE MADE WHILE CPUMTR IS PROCESSING AN MTR REQUEST. MTR WAITS FOR ITS EXCHANGE PACKAGE TO BECOME AVAILABLE BY TESTING THE XP'S MA. IF (MA)=0, THE EXCHANGE SEQUENCE WILL BE EXECUTED.

EXCHANGE JUMP

POOL PPU

1. EACH POOL PPU HAS ITS OWN EXCHANGE PACKAGE LOCATED WITHIN CPUMTR.

2. TYPICALLY, POOL PPU REQUESTS FOR CPUMTR ARE ISSUED BY PPU RESIDENT SUBROUTINE FTN.

   FTN SETS UP TO DO EXCHANGE -

   (P) = PPR ADDRESS IN CPUMTR WHERE EXECUTION IS TO BEGIN

   (B2) = ADDRESS OF PPU's EXCHANGE PACKAGE WITHIN CPUMTR

   (B0) ≠ 0 MXN SUCCESSFUL FLAG

   (OR) = REQUEST TO BE PROCESSED BY CPUMTR

3. FTN PERFORMS INSTRUCTIONS -

   LDC ADDRESS OF PPU's XP

   MXN CP

   TO EXCHANGE CPU cp WITH THE XP FOR THE PPU, cp IS USUALLY 0 UNLESS CPU 0 IS <u>OFF</u>.

EXCHANGE JUMP

POOL PPU

4. FTN READS XP'S FIRST WORD TO DETERMINE IF EXCHANGE WAS SUCCESSFUL. IF (B0)=0, THE MXN WAS SUCCESSFUL. OTHERWISE, FTN REISSUES THE EXCHANGE (STEP 3) UNTIL SUCCESSFUL.

5. CPUMTR BEGINS EXECUTION AT PPR TO PROCESS THE REQUEST FROM THE OUTPUT REGISTER (OR).

6. WHEN COMPLETED, CPUMTR CLEARS BYTE 0 OF THE OUTPUT REGISTER AND EXITS BY DOING

   ```
   XJ  B2
   ```

   WHICH CAUSES THE PREVIOUS XP TO BE BROUGHT BACK INTO EXECUTION (USUALLY).

7. FTN IS LOOPING WAITING FOR BYTE 0 OF THE OUTPUT REGISTER TO BECOME CLEAR. WHEN THIS IS SENSED, FTN RETURNS TO ITS CALLER AND EXECUTION IS RESUMED.

EXCHANGE JUMP

CPU PROGRAM

1. EACH CONTROL POINT HAS ITS OWN EXCHANGE PACKAGE IN ITS CONTROL POINT AREA.

2. THE EXCHANGE JUMP FOR A CPU PROGRAM IS ALWAYS SUCCESSFUL.

3. THE EXCHANGE WILL BE MADE WITH THE XP SPECIFIED BY THE ADDRESS CONTAINED IN MA.

   MA IS ALWAYS THE CONTROL POINT AREA ADDRESS.

   THE EXCHANGE PACKAGE LOCATED AT THE CPA ADDRESS HAS

(P) = MTR	ADDRESS IN CPUMTR WHERE EXECUTION IS TO BEGIN FOR CPU PROGRAM REQUESTS

   (B2) = CPA ADDRESS

   (MA) = 0

4. THE CPU PROGRAM PLACES ITS MONITOR REQUEST IN RELATIVE LOCATION 1 (RA+1) AND DOES AN XJ INSTRUCTION.

   TYPICALLY, THIS IS DONE AS PART OF SYS= (COMCSYS) PROCESSING.

EXCHANGE JUMP

5. CPUMTR BEGINS EXECUTION AT ADDRESS MTR TO PROCESS THE REQUEST CONTAINED IN RA+1.

   CPUMTR WOULD ALSO BEGIN EXECUTION AT MTR IF THE CPU PROGRAM HAD CAUSED AN EXCHANGE TO OCCUR BY DOING AN ILLEGAL OPERATION (HARDWARE DETECTED) SUCH AS EXECUTING A ZERO INSTRUCTION.

6. WHEN CPUMTR IS DONE PROCESSING THE RA+1 REQUEST, IT DOES A

   ```
             XJ  B2
         MTR ....
   ```

   WHICH RETURNS THE CONTROL POINT INTO EXECUTION (USUALLY).

   THIS SEQUENCE OF CODE IN CPUMTR IS SUCH THAT THE NEXT ADDRESS IS MTR SO THE "CPUMTR" EXCHANGE PACKAGE RETURNED TO THE CONTROL POINT AREA IS PROPERLY SET WITH

   (P) = MTR

7. RA+1 REQUESTS NOT PROCESSED BY CPUMTRA ARE DONE BY PPU ROUTINES. EXCEPT FOR CIO, THE CPU IS RELINQUISHED.

EXCHANGE JUMP

SYSTEM CONTROL POINT

5. CPUMTR EXITS WITH THE

```
XJ  B2
```

CAUSING THE SYSTEM CONTROL POINT TO GO INTO EXECUTION AND THE "CPUMTR" EXCHANGE PACKAGE WILL HAVE

```
(P) = MTR
```

6. WHEN PROGRAM MODE COMPLETES, IT DOES A

```
XJ
```

CAUSING AN EXCHANGE TO THE EXCHANGE PACKAGE AT THE MA ADDRESS - THE SYSTEM CPA.

THE (P) WILL BE PRG THE PROGRAM MODE PROCESSOR SO THAT THE NEXT EXCHANGE TO PROGRAM MODE BEGINS EXECUTION AT THIS ADDRESS

EXCHANGE JUMP

SYSTEM CONTROL POINT (CPUMTR PROGRAM MODE)

1. CPUMTR DETERMINES THAT A PROGRAM MODE SUBROUTINE IS REQUIRED TO COMPLETE A REQUEST.

2. THE PROGRAM MODE REQUEST IS ADDED TO THE PROGRAM MODE REQUEST QUEUE (PR).

3. IF PROGRAM MODE IS CURRENTLY ACTIVE, CPUMTR EXITS WITH A

   ```
   XJ  B2
   ```

   SUCH THAT THE "CPUMTR" EXCHANGE PACKAGE WILL HAVE

   ```
   (P) = MTR
   ```

   SINCE THE CPU PROGRAM (CPUMTR PROGRAM MODE) RUNS AT A CONTROL POINT (THE SYSTEM CONTROL POINT) PROGRAM MODE WILL PROCESS THE REQUEST FROM THE PR QUEUE.

4. IF PROGRAM MODE IS NOT ACTIVE, CPUMTR CALLS SUBROUTINE BCP (BEGIN CENTRAL PROGRAM) TO CAUSE PROGRAM MODE (SYSTEM CP) TO GET THE CPU.

   BCP PICKS THE SYSTEM CONTROL POINT AS THE CP TO GET THE CPU - IT IS THE HIGHEST CPU PRIORITY CP. (B2) BECOMES THE ADDRESS OF THE SYSTEM CP, WHICH IS THE PROGRAM MODE XP. THE PREVIOUS ACTIVE CP'S XP (FROM ACPL) IS RESTORED TO ITS CONTROL POINT AREA.

EXCHANGE JUMP

SWITCHING CONTROL POINT EXECUTION

1. ASSUME CP4 EXECUTING IN THE CPU
   - XP AT 1000 (CPA4) HAS (P) = MTR
   - ACPL POINTS TO CPA4 (1000)
   - XP IN CPU HAS (MA) = 1000
   - CPU IS IN PROGRAM MODE

2. PP2 WRITES AN RCPM (REQUEST CPU) REQUEST FOR CP6 INTO ITS OR AND DOES A MXN

   (FIGURE 1)

3. EXCHANGE OCCURS SINCE CPU IS INTERRUPTIBLE
   - XP IN CPU IS NOW PP2's XP WITH (MA)=0
   - CP4's XP (FROM CPU) IS NOW RESIDING AT PPU's XP ADDRESS IN CPUMTR
   - CPUMTR IS EXECUTING AT PPR FOR PP2

   (FIGURE 2)

4. CPUMTR PROCESSES PP2's OR REQUEST.
   - (B2) = PP2 XP ADDRESS

5. RCPM PROCESSING DECIDES TO START UP CP6 BY CALLING BNJ (BEGIN NEW JOB).

SWITCHING CONTROL POINT EXECUTION

6. BNJ MOVES XP FROM (B2) TO THE XP ADDRESS SPECIFIED IN ACPL.

   THIS PUTS CP4's XP INTO CP4's CPA.

   ACPL IS SET FOR CP6's XP ADDRESS -
   CPA6. (1400)

   PP2's XP IS REBUILT FOR FUTURE PPU REQUESTS -

   (RA) = 0 (RAX) = 0
   (FL) = MFLL (FLX) =
   (B1) = 1
   (B2) = PP2 XP ADDRESS
   (MA) = 0
   ERROR MODE SET

   B2 IS THEN ENTERED WITH CP6's XP ADDRESS WHICH IS THE ADDRESS OF CPA6.

   PP2's OUTPUT REGISTER IS CLEARED.

   (FIGURE 3)

7. CPUMTR EXITS DOING A

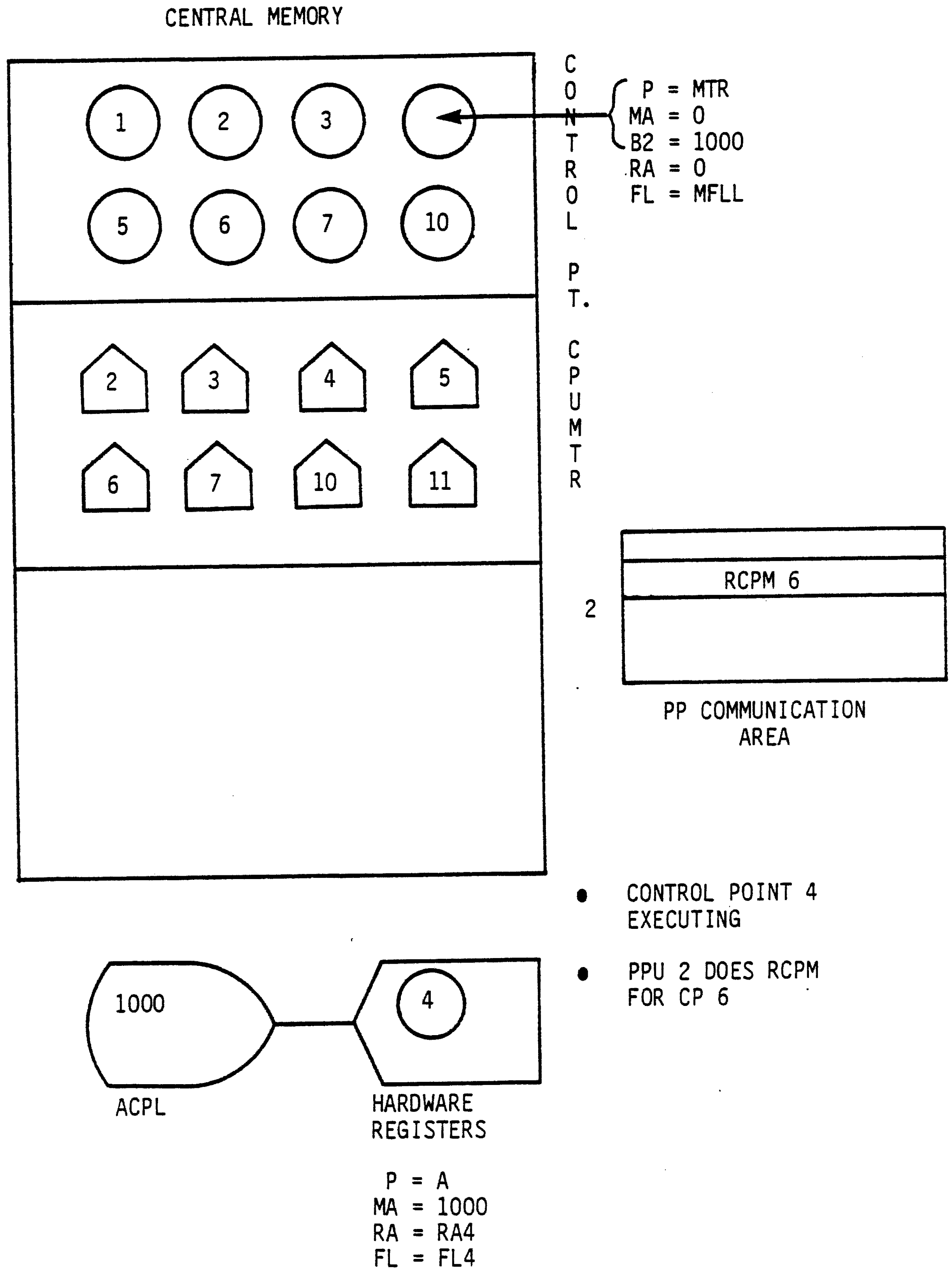
   XJ B2

   CAUSING THE EXCHANGE PACKAGE FOR CP6 TO ENTER THE CPU. THE "CPUMTR" PACKAGE NOW FOUND AT CPA6 HAS

   (P) = MTR

   (FIGURE 4)

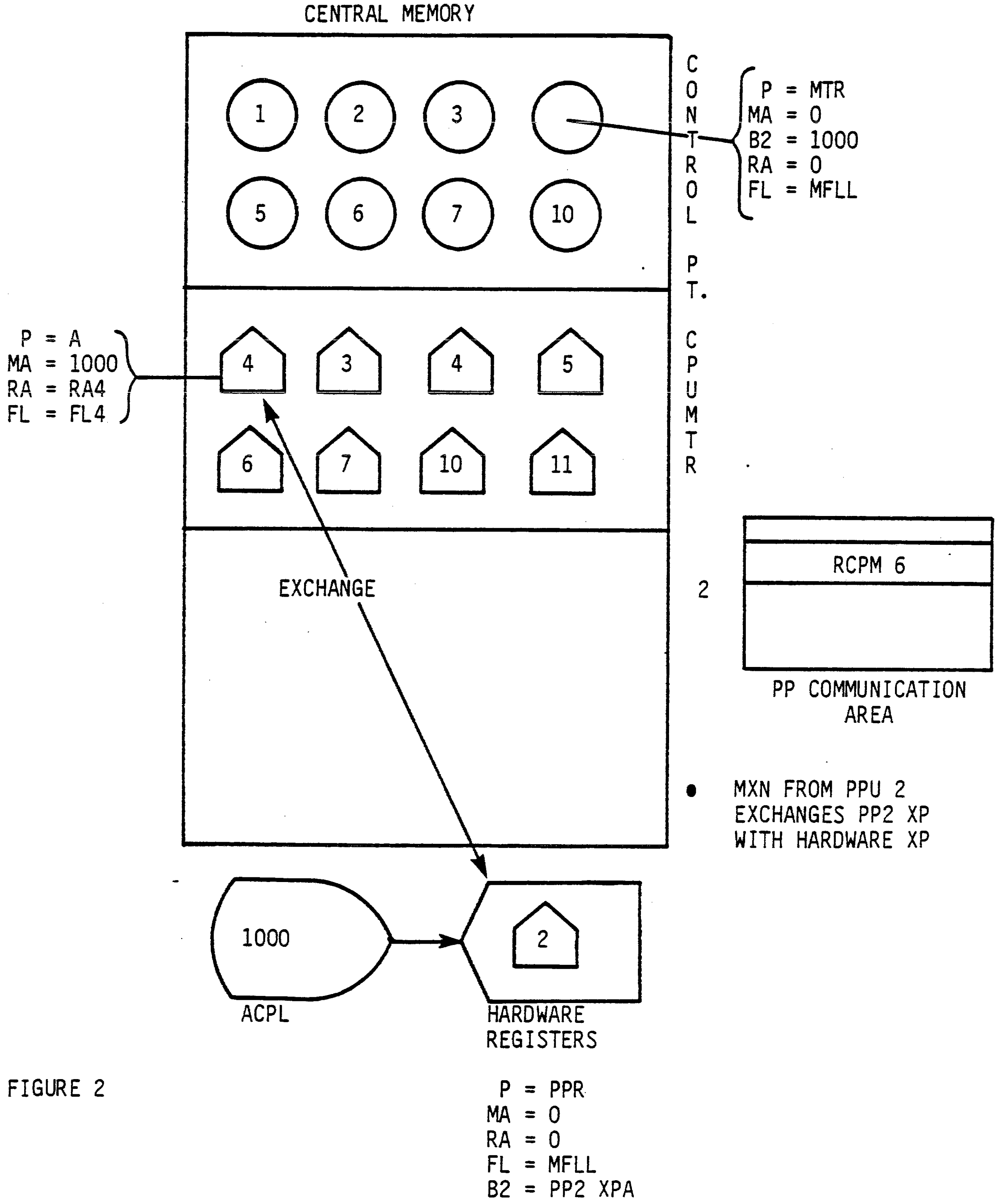
CONTROL POINT SWITCHING



- CONTROL POINT 4 EXECUTING
- PPU 2 DOES RCPM FOR CP 6

FIGURE 1

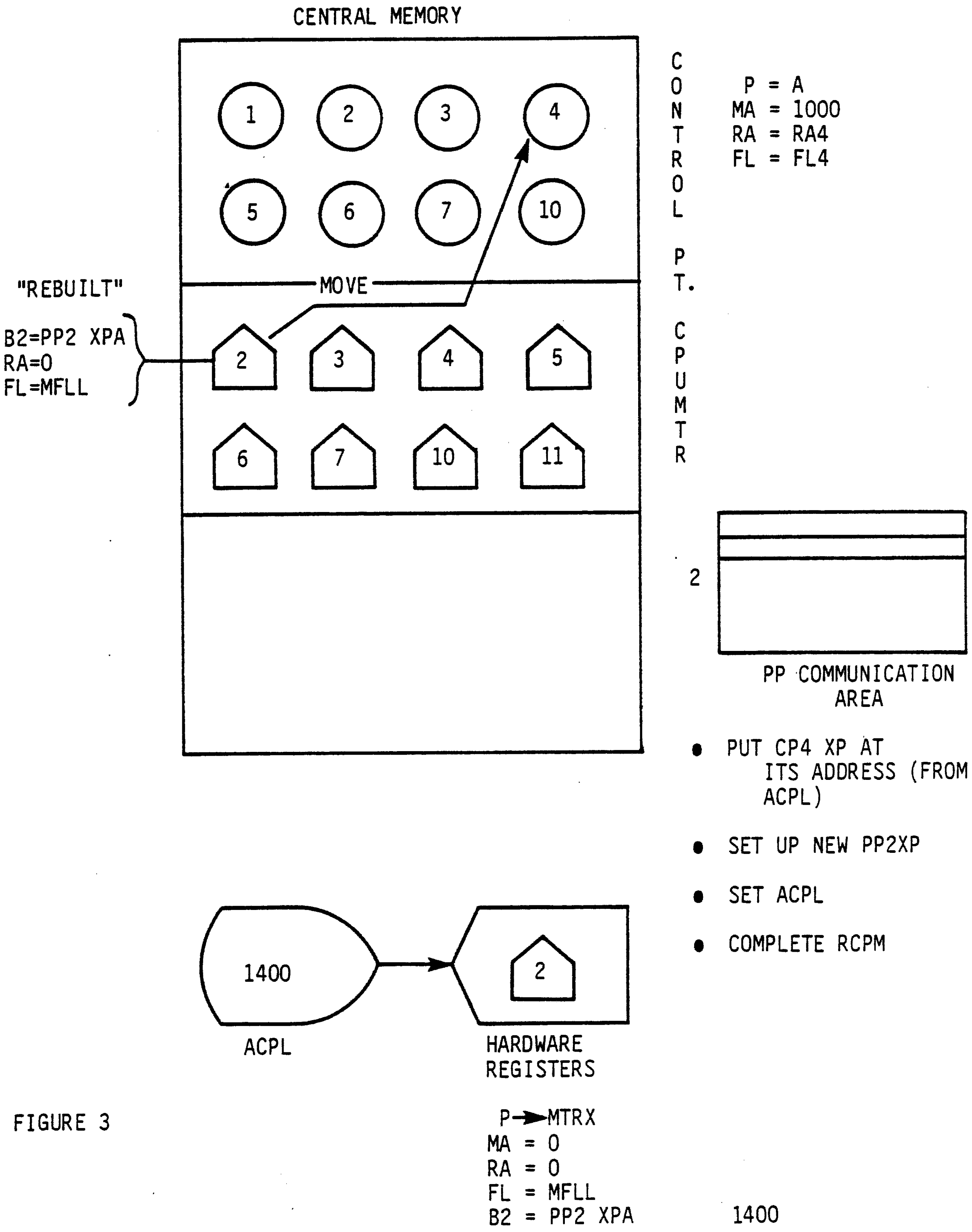
CONTROL POINT SWITCHING



- MXN FROM PPU 2 EXCHANGES PP2 XP WITH HARDWARE XP

FIGURE 2

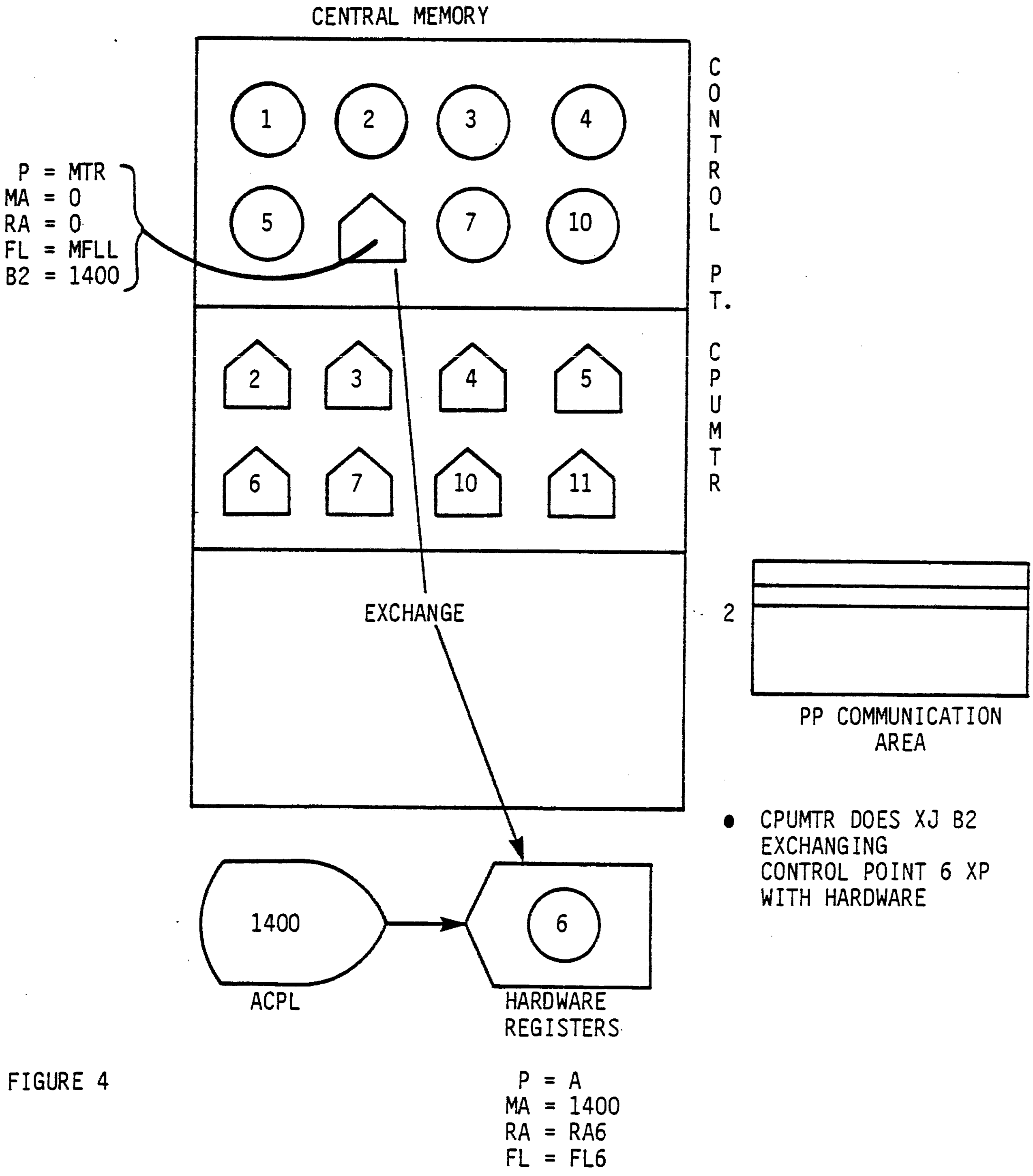
CONTROL POINT SWITCHING



- PUT CP4 XP AT ITS ADDRESS (FROM ACPL)
- SET UP NEW PP2XP
- SET ACPL
- COMPLETE RCPM

FIGURE 3

CONTROL POINT SWITCHING



- CPUMTR DOES XJ B2 EXCHANGING CONTROL POINT 6 XP WITH HARDWARE

FIGURE 4

QUESTION SET LESSON 4

1. Explain the means of communication between Pool PPs, monitor (CPUMTR/MTR), and a Control Point (CPU job).

2. What exchange packages are present when CEJ/MEJ is present?

3. What does a Pool PP do to make a CPUMTR monitor function request?

4. How does MTR make a request to CPUMTR?

5. How does a CPU job make a system (monitor) request?

6. How does CPUMTR manage the exchange packages when activating a "new" control point?

7. What is the difference between CPUMTR program mode and monitor mode?

8. When in monitor mode what does a "XJ" instruction do? In program mode?

LESSON 5
DELAY AND RECALL

LESSON PREVIEW:

This lesson introduces the student to the system delay controls and the various recall mechanisms in the system. The delay controls regulate the frequency at which certain system operations are performed. The various forms of recall allow the user to optimize a CPU program to make efficient use of NOS multi-programming capabilities.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- List the system activities regulated by the delay controls.
- Explain the types of recall available in NOS and give an example of how each may be used.

REFERENCES:

NOS IMS - Chapter 1 NOS RM, 2-1-3-4, 2-11-12 NOS IHB, 8-15-16

ASSIGNMENT:

Write a CPU program using the different types of recall and measure the recall time period where possible.

SUPPLEMENTAL TEXT: RECALL

The CPU or job status in NOS is indicated by the values:

W	Waiting for the CPU
X	Waiting in Periodic Recall
I	Auto Recall
A	Job running in CPU 0
B	Job running in CPU 1
" "	No CPU status but other job activity in progress

Periodic Recall

"Periodic recall" means that a program is placed on recall until a certain period of time (specified by the CR delay value) has elapsed.

A program enters the "periodic recall" state by issuing a RCL RA+1 request. CPUMTR will set X status for the job and set a starting time in the control point area (word CPCW). CPUMTR eventually identifies the "oldest" job in X status. MTR recalls this job when the CR delay expires by issuing a RCLM (Recall) monitor function for that control point. The RCLM causes the job to be given W status.

Automatic Recall

"Automatic recall" means that a program is placed on recall until a particular RA+1 request has completed or is requested to resume execution by a PP program.

A program enters the "automatic recall" state by issuing a RA+1 request with the recall bit (bit 40 of RA+1) set to 1. This is usually an on recall until a particular RA+1 request has completed or is requested to resume execution by a PP program.

A program enters the "automatic recall" state by issuing a RA+1 request with the recall bit (bit 40 of RA+1) set to 1. This is usually an automatic condition since all RA+1 requests except for CIO requests and requests from jobs whose queue priority is greater or equal to MXPS are forced into automatic recall. There may be only one PP in automatic recall for a job at a given time. CPUMTR sets the CPU status to I and clears any existing X status, releases the CPU from the job, and assigns a PP to complete the RA+1 request. The PP program satisfying the RA+1 request causes the job to resume execution by dropping its PP or by clearing RA+1 and doing a RCPM (Request CPU) function for that control point. Either of these operations will cause the control point to reenter W status and wait for CPU assignment.

Auto Recall

"Auto recall" means that a program is placed on recall until the completion bit (bit 0) of a specified address is set to 1.

A program enters the "auto recall" state by issuing a RCLP RA+1 request (RCL with the recall bit (bit 40) set to 1) and a status address specified for completion bit checking. CPUMTR will set the job into X status, leaving RA+1 intact. Periodically, as controlled by the AR delay value, MTR checks for PP recalls. Jobs in I status with RA+1 requests are requested to be restarted. Jobs in X status with RA+1 requests are ignored unless the RA+1 request is an RCLP. If RCLP, that is a "auto recall" request, the completion bit is checked (bit 0 of the specified address), and if set, an RCLM (Recall) is done for the control point. If X or I status is not present, or X or I is set but no RA+1 request is present, the PP recall word RLPW is checked. If a request is present in RLPW, then a PP is requested for that PP program by a RPPM.

DELAY

DELAY	SPECIFY SYSTEM DELAY PARAMETERS FOR PERIODIC OPERATIONS
JS	JOB SCHEDULAR INTERVAL (SECONDS)
CR	CPU PROGRAM RECALL INTERVAL HOW LONG IN X STATUS?
AR	PPU AUTO RECALL INTERVAL
CS	CS JOB SWITCH INTERVAL WHEN MTR DECIDES IT IS TIME TO DO ANOTHER JOB

RECALL

AUTOMATIC RECALL

```
READ        LFN,R
```

NEED READ TO COMPLETE AS COMPUTATION NEEDS DATA FROM LFN

AUTO RECALL

```
READ        LFN

COMPUTE

RECALL      LFN
```

DO NOT NEED DATA FROM LFN IMMEDIATELY; PROGRAM CAN DO SOME COMPUTATION BEFORE NEEDING DATA

PERIODIC RECALL

```
DLY         SUBR
            PDATE       DLYA        START
DLY1        RECALL
            PDATE       DLYB
            SA1         DLYA
            SA2         A1+B1
            SX5         B1+B1
            BX2         X1-X2
            BX5         X5-X2
            ZR          X5,DLY1     IF MORE
            JP          DLYX
```

QUESTION SET LESSON 5

1. What are the five CPU or job status values in NOS?
2. What is meant by the terms: Periodic Recall, Automatic Recall, and Auto Recall?

LESSON 6
CPUMTR

LESSON PREVIEW:

This lesson introduces the student to CPUMTR and overviews the communication between the user CPU program, Pool PP routines, and the system monitors: CPUMTR and MTR.

OBJECTIVES:

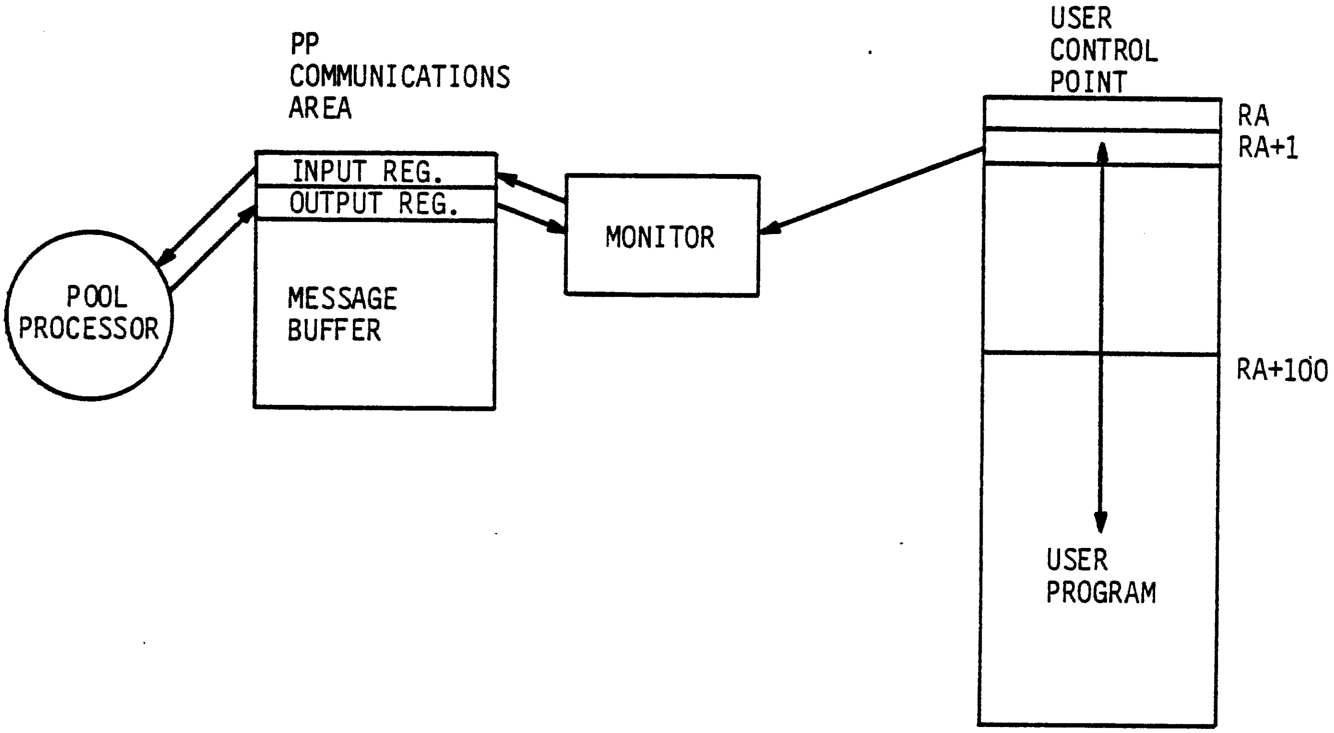
Upon the successful completion of this lesson, the student should be able to:

- Explain the communication between the system monitors (CPUMTR and MTR) and with user CPU programs and Pool PP routines.
- Explain the modular structure of CPUMTR, identifying what hardware/software property requires the loading of a corresponding module of CPUMTR.
- Describe the conditions under which CPUMTR is entered from a CPU program and what processing is done by CPUMTR for this case.
- Describe the conditions under which CPUMTR is entered from MTR and what processing is done by CPUMTR for this case.
- Describe the conditions under which CPUMTR is entered from a Pool PP and what processing is done by CPUMTR for this case.
- Describe the processing done by CPUMTR to begin and suspend execution of a CPU program.
- Describe the processing done by CPUMTR to assign a PP to a job (control point).
- Describe the processing done by CPUMTR to invoke job scheduling.
- Describe the job advancement process.
- Recognize the RA+1 requests, MTR/CPUMTR requests, and PP monitor functions processed by CPUMTR.

REFERENCES:

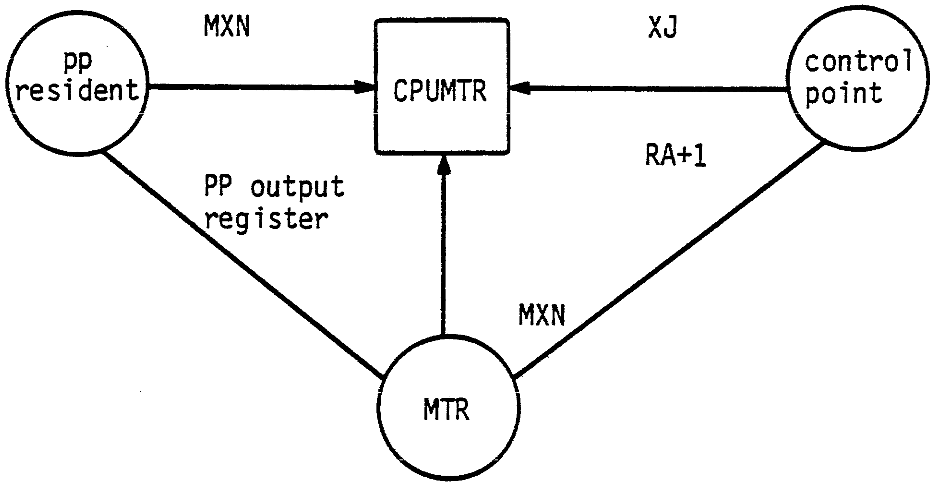
NOS IMS - Chapter 3

SYSTEM INTERACTION

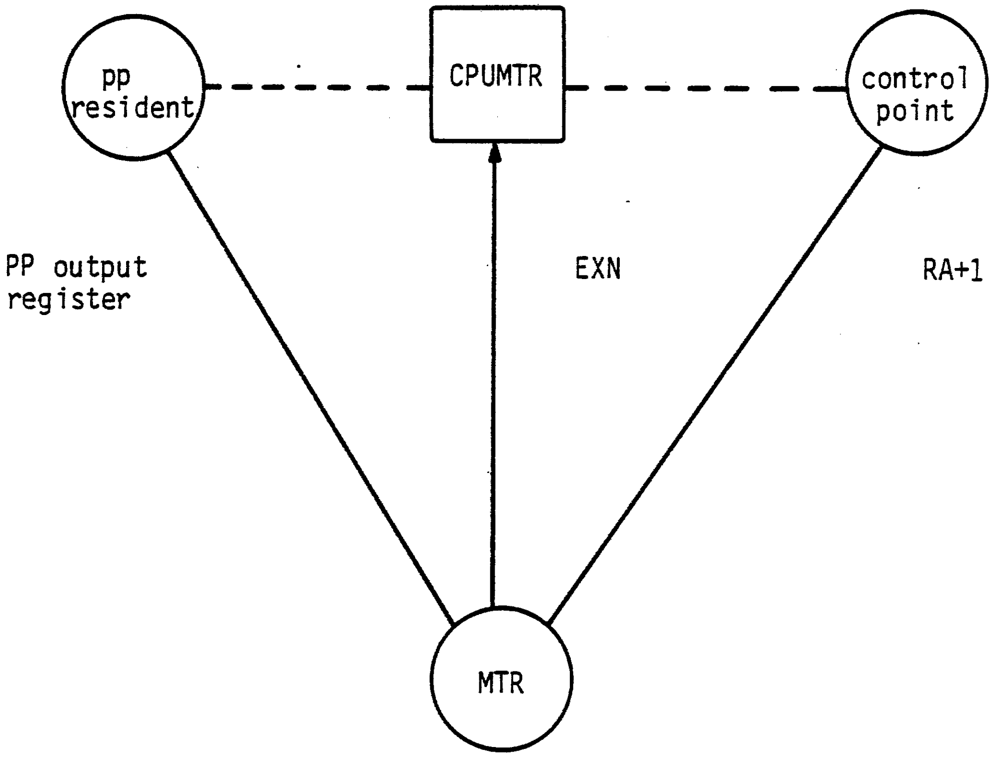


Monitors Interaction

CEJ/MEJ
option present



no
CEJ/MEJ
option



CPUMTR

MONITOR MODE
PROGRAM MODE
"CONDITIONAL" CODE BLOCKS
CMU/OCMU
CMUMTR/OCMUMTR
CEJ/OCEJ
EXPACS
DCP
SUBCP
SCP FACILITY
SCPUEC
ECS
ECSBUF
MMF/OMMF
MMFBUF
UEC
VMS
CP176
XP176

CONTROL WORDS

PX PROGRAM MODE EXIT REQUEST

= 1	PROGRAM MODE COMPLETE
= -1	PP REQUESTED BY PROGRAM MODE
= 0	PX MAY BE USED AS TEMPORARY IF CEJ/MEJ NOT PRESENT

SM STORAGE MOVE REQUEST

INCR		CP

PR PROGRAM MODE REQUEST

STACK OF REQUESTS TO BE DONE IN PROGRAM MODE

S	PPU

S = STORAGE MOVE (BIT 30)
PPU = PPU REQUESTS (BITS 0 - 19)

ST CPU START TIMES (TWO WORDS)

STARTING TIMES FOR EACH CPU ARE UPDATED EACH TIME THAT THE CPU JOB TIME IS UPDATED.

CONTROL WORDS

CR CPU REQUESTS (TWO WORDS)

STACK OF REQUESTS FOR CPU USAGE.
EACH BIT REPRESENTS A CP THAT NEEDS THE CPU.

THE SECOND WORD (CR+1) CONTAINS A MASK BIT WHICH DETERMINES WHERE TO START THE SCAN.

XS CPU RECALLS FOR INITIATED JOBS.

PS CPU RECALL FOR NO PPU.

XS INDICATES THE NEXT JOB TO BE RECALLED BY MTR.

PS INDICATES THE NEXT JOB TO BE RECALLED WHEN PP IS AVAILABLE.

CPA	TIME TO RECALL	XS

CPA	PS

CONTROL WORDS

PP	PP PRIORITY REQUEST
	(PP) = OUTPUT REGISTER ADDRESS OF A PP WITH PENDING RPPM. IF NEGATIVE, 1SP WAS REQUESTED BUT NO PPs WERE AVAILABLE. IF MTR ISSUED THE RPPM, THE OUTPUT REGISTER WILL BE CLEARED BUT PP IS RESERVED FOR MTR.
JA	JOB ADVANCE REQUESTS
	STACK OF BITS REPRESENTING CPS WAITING TO BE ADVANCED BUT NO PPs WERE AVAILABLE TO DO SO.

CPUMTR

CONTROL WORDS

JS INPUT REGISTER FOR 1SJ CALL

SYSTEM CONTROL POINT NUMBER IS SET DURING CPUMTR PRESET

1 S J	SCP	

MP INPUT REGISTER FOR 1MA CALL

1MA IS USED FOR DAYFILE MESSAGES (MSG)
REQUEST FOR STORAGE INCREASES (MEM), AND SYSTEM CONTROL POINT OPERATIONS (SSC,SSF).

1 M A	

RC AUTO RECALL RA+1 REQUEST

USED TO TEST FOR RCLP

RCLP	

AM ACTIVITY MASK FOR PP AND TAPES

USED TO MASK NUMBER OF ASSIGNED PPs AND ACTIVITY COUNT FROM STSW

0037	0000	7777	0000	0000

CPUMTR

ENTRY FROM CPU PROGRAM

```
MTRX
  .
  .
  .
XJ1             XJ              B2              EXCHANGE TO PROGRAM
  .
  .
  .
```

1. READ RA WORD FROM XP
2. READ (RA+1)
3. IF RA=0, CHECK PROGRAM MODE EXIT CALL
4. IF (RA+1) = 0, PROCESS CPU REQUEST (CPR)
5. IF CYBER 176, CHECK PSD REGISTER FOR ERRORS. IF NONE, CONTINUE WITH STEP 6. IF ERRORS, FORMAT (RA) TO LOOK LIKE A NON-CYBER 176 HARDWARE ERROR AND SET ARET (ARITHMETIC) ERROR FLAG.
6. IF (P) > 2, EXCHANGE BACK (GO TO XJ1)
7. CHECK JOB STATUS (CJS)

NOTE

ON A HARDWARE INTRODUCED INTERRUPT (MODE ERROR)

(P) = 0

EXIT MODE BITS = BITS 48 - 53 OF (RA)

P ADDRESS OF ERROR = BITS 30 - 47 OF (RA)

RA =

0	E	P	0

CPUMTR

CHECK JOB STATUS

CHECK FOR: MODE ERRORS, PROGRAM STOPS, TIME LIMIT, SRU LIMIT

1. READ RA, READ P, READ FL.
2. IF P $\geq$ FL, GO SET PSET ERROR FLAG.
3. IF P = 0 OR 1 AND (RA+1) = 0, THE EXIT MODE BITS ARE RETRIEVED FROM (RA) AND EITHER PEET OR ARET ERROR FLAG IS SET.
4. READ (RA+P). IF ZERO INSTRUCTION, SET PSET ERROR FLAG.
5. CHECK SRUW FOR TIME LIMIT OR SRU LIMIT EXCEEDED FLAGS. SET TLET OR SRET ERROR FLAG.

CPUMTR

BEGIN CENTRAL PROGRAM	(BCP)
BEGIN NEW JOB	(BNJ)
END CENTRAL PROGRAM	(ECP)

BCP - BEGIN CENTRAL PROGRAM

1. IF JOB IS IN W STATUS, CALL CJS. IF CJS FINDS NO ERRORS, STEP 6 IS DONE.
2. IF JOB HAS AN RCLP IN (RA+1) AND COMPLETION BIT IS SET, CLEAR RA+1, AND CALL CJS. IF CJS FINDS NO ERRORS, STEP 5 IS DONE. IF COMPLETION BIT IS NOT SET, GO TO STEP 4.
3. FOR NON-RCLP JOBS, CJS IS CALLED. IF CJS FINDS NO ERRORS, STEP 5 IS DONE.
4. ENTER JOB INTO X STATUS.
5. SET W STATUS FOR CONTROL POINT.
6. READ ACPL FOR THIS CPU TO FIND CURRENT ACTIVE JOB.
7. IF THIS JOB'S PR (FROM JCIW BYTE 0) < CURRENT JOB'S PR, THIS CP'S BIT IS MERGED INTO CR CONTROL WORD AND BCP RETURNS TO MTRX.
8. IF THIS JOB'S PR ≥ CURRENT JOB'S PR, THIS CP'S BIT IS MERGED INTO CR.
9. BNJ (BEGIN NEW JOB) IS ENTERED.

CPUMTR

BNJ - BEGIN NEW JOB.

BNJ SETS UP EXCHANGE PACKAGES FOR "NEW" CPU PROGRAM EXECUTION.

1. COMPUTE CPU TIME FOR CURRENT JOB.

2. ACPL IS SET TO REFLECT NEW XP ADDRESS AND (B2) IS SET TO THIS ADDRESS ALSO.

3. IF THE OLD XP WAS NOT WHERE IT BELONGS, IT IS MOVED TO THE ADDRESS SPECIFIED IN THE OLD ACPL. THIS WOULD BE THE CASE IF A PPU CAUSED ANOTHER JOB TO GET THE CPU.

   IF THE XP HAD TO BE MOVED, THEN A "NEW" XP IS BUILT WHERE IT HAD BEEN . . .

   RA = 0
   (B1) = 1
   FL = MACHINE SIZE
   (B2) = ADDRESS OF THIS PACKAGE
   RAX = 0
   FLX = ECS SIZE
   (MA) = 0
   (EM) = EEMC OR EEMC + 60B IF CYBER 176

4. THE ERROR EXIT ADDRESS (EEA) IS SET IN THE XP, IF CYBER 176

   BNJ EXITS TO ADDRESS MTR TO TEST RA AND RA+1 FOR THE JOB BEING GIVEN THE CPU.

CPUMTR

ECP - END CENTRAL PROGRAM

1. REQUEST FOR THIS JOB CLEARED FROM CR.
2. REQUEST SET FOR THIS JOB IN XS.
3. STATUS IN STSW SET TO X OR I.
4. IF JOB WAS NOT RUNNING (ACPL NOT THIS JOB), EXITS CPUMTR.
5. IF JOB WAS RUNNING AND NO OTHER CR REQUESTS WERE FOUND, THE IDLE JOB IS RUN AFTER COMPUTING THE TIME FOR THIS JOB (CPT).
6. THE CR WORD IS SEARCHED FOR OTHER JOBS.

   (CR+1) DETERMINES WHERE TO START EXAMINING JOBS WAITING FOR THE CPU.

   THE PR (JCIW BYTE 0) OF THE CURRENT JOB AND WAITING JOBS ARE COMPARED FOR JOBS WANTING THIS CPU. THE HIGHEST PRIORITY JOB FOR THIS CPU IS FOUND.
7. THE CR REQUEST FOR THE "NEW" JOB IS CLEARED.
8. CPT IS CALLED TO DO ACCOUNTING FOR THE "OLD" JOB WITH CONTROL RETURNING TO BNJ WHEN CPT IS DONE.

CPT - COMPUTE CPU TIME

CPT IS CALLED WHENEVER A JOB RELINQUISHES THE CPU.

1. THE AMOUNT OF TIME IN THE CPU IS DETERMINED BY

   $$\Delta T = RTCL - ST$$

2. THE CURRENT RTCL TIME IS SET INTO THE START TIME (ST) FOR THE CPU.
3. IF CONTROL POINT 0 OR CPU MULTIPLIER (S0 OR S1) IS ZERO, THE ΔT IS NOT SCALED BY S0 OR S1.
4. CPTW (CPU TIME) IS ADVANCED BY ΔT.
5. IF TIME LIMIT HAS BEEN REACHED, THE TIME LIMIT FLAG (BIT 58) IS SET IN SRUW.
6. IF SRU ACCUMULATION IS ENABLED (BIT 59 OF MP3W CLEAR), THE SRU LIMIT IS CHECKED AND SRU LIMIT FLAG (BIT 56) SET IN SRUW, IF THE LIMIT HAS BEEN REACHED. THE CHANGE IN CPU TIME IS REFLECTED IN THE SRU ACCUMULATOR SRUW.

   $$\begin{aligned} SRUW &= \Delta T\text{-}S0 \ * \ CPM + SRUW \\ &= \Delta T\text{-}S1 \ * \ CPM + SRUW \end{aligned}$$

CPUMTR

PP ASSIGNMENT

APJ - ASSIGN PPU JOB
APS - ASSIGN PPU AND SEARCH LIBRARY
SPL - SEARCH PERIPHERAL LIBRARY

APJ - ASSIGN PPU JOB

(RA+1) REQUEST NOT PROCESSED WITHIN CPUMTR. CIO (RA+1) REQUESTS ENTER APJ AT STEP 3

1. FORCE AUTO RECALL IF QP < MXPS.

2. IF OTHER PPUs or TAPE ACTIVITY (USE AM MASK), FORCE X STATUS (IF QP < MXPS).

3. IF PP SATURATION (NP=0) OR IF MTR REQUESTING ONLY PP (PP = 0), SET PS AND X STATUS.

4. IF ROLLOUT SET (JCIW BIT 24 SET) OR IF JOB EXCEEDS NPPS AND LESS THAN 3 PPs ARE AVAILABLE OR IF PLD LOCKED → FORCE X STATUS.

5. SET RECALL BIT AS BIT 41 AND CP NUMBER AS BITS 40-36.

6. CALL APS.

7 IF I STATUS (AUTO RECALL) CALL, SET OUTPUT REGISTER ADDRESS OF PP ASSIGNED INTO RA+1 AND CALL ECP: RETURN OTHERWISE.

APS - ASSIGN PPU AND SEARCH LIBRARY

FROM APJ OR RPPM REQUEST + OTHERS

NP	NEXT AVAILABLE PPU

INPUT REGISTERS OF ALL AVAILABLE PPUs ARE LINKED TOGETHER. THE START OF THIS LINKED STACK IS (NP) = (PPAL). THE END OF THE STACK IS A ZERO ADDRESS. WHEN PP IS ASSIGNED, ITS IR IS PLACED INTO NP. WHEN PP IS DROPPED, (NP) IS STORED INTO ITS IR AND THE IR ADDRESSED IS STORED IN NP.

1. IF NO PP AVAILABLE (NP=0), RETURN
2. IF CP FIELD LENGTH BEING MOVED, RETURN
3. CALL SPL
4. IF LIBRARY LOCKED, RETURN
5. PUT PROGRAM LOAD PARAMETERS INTO THE PP OUTPUT REGISTER
6. PUT COPY OF INPUT REGISTER IN MESSAGE BUFFER
7. ADJUST NP STACK LINKAGE
8. INDICATE DIRECTORY SEARCHED BY CLEARING BYTE 0 UPPER 6 BITS OF INPUT REGISTER
9. UPDATE STSW PP COUNT

SPL - SEARCH PERIPHERAL LIBRARY

CALLED BY APS OR SPLM

1. RETRIEVE PERIPHERAL LIBRARY DIRECTORY (PLDP) POINTER FROM CMR
2. IF LIBRARY LOCKED, I.E., PLD ADDRESS = 0 AS WOULD BE THE CASE DURING A SYSEDIT, RETURN
3. SEARCH DIRECTORY FOR MATCH BY A BINARY SEARCH
4. IF NO MATCH AND 6XX, HANG
5. IF NO MATCH, USE SFP (SCOPE FUNCTION PROCESSOR) AS THE ROUTINE TO BE LOADED
6. RETURN LOAD PARAMETERS FOR ROUTINE (IF MATCH) OR SFP (NO MATCH)

START UP SCHEDULER

SJS - START JOB SCHEDULER

CALLED BY RSJM, DPPM, ARTF, SFL.

RSJM TO START IT UP IF JOBS CHANGE.

DPPM IF 1SP IS WAITING FOR A PPU.

ARTF IF SCHEDULER CYCLE TIME HAS EXPIRED (RTCL-(JSCL +1)).

SFL AFTER ADJUSTING FIELD LENGTH.

1. IF SCHEDULING ALREADY ACTIVE, RETURN.
2. IF 1SP CYCLE TIME HAS ELAPSED, CALL APS FOR 1SP IF PP IS AVAILABLE.
3. IF PP IS NOT AVAILABLE FOR 1SP, SET REQUEST INTO PP CONTROL WORD SO THAT 1SP GETS NEXT AVAILABLE PPU.
4. IF 1SP IS NOT CALLED, 1SJ WILL BE. APS IS CALLED FOR 1SJ. IF PP IS NOT AVAILABLE, THAT'S OK.

CPUMTR

XS/PS CONTROL

XS

1. WHEN ENTERING X STATUS, THE CURRENT RTCL IS SET INTO CONTROL POINT WORD CPCW.
2. IF NO OTHER JOBS ARE IN RECALL ((XS)=0), XS IS ENTERED WITH THE CPA AND MSCL BYTE 1 - CPU RECALL VALUE (DELAY CR).
3. IF XS IS SET, THE JOB WILL HAVE TO WAIT ITS TURN.
4. IF A JOB IS TAKEN OUT OF X STATUS, CPCW IS CLEARED. IF THE JOB WAS IN XS, A NEW XS JOB WILL BE FOUND - THE OLDEST JOB ON X RECALL.
5. XS PROVIDES A FAST MECHANISM TO GET JOBS OUT OF RECALL WITHOUT A LOT OF OVERHEAD.

PS

IF PS IS ALREADY SET, XS CONTROL IS EXERCISED. OTHERWISE, THE REQUEST IS ENTERED INTO PS.

PS CONTROL IS ONLY USED AT SATURATION BY APJ AND SCP FACILITY.

JOB ADVANCEMENT

- BY SCHEDULER STARTING NEW JOB
- NO ACTIVITY AT CONTROL POINT AND W AND X BITS CLEAR
- DIS OR EQUIVALENT WISHES TO PROCESS CONTROL STATEMENT OR ERROR FLAG

JAV - JOB ADVANCE

JOB IS ADVANCED (I.E., CALL 1AJ) IF PP IS AVAILABLE AND JOB ADVANCE CONDITIONS MET. IF NO PP IS AVAILABLE, JOB ADVANCE FLAG (STSW BIT 53) IS SET AND THE CPU DROPPED.

IF ROLLOUT FLAG SET (JCIW BIT 24), THEN

- NO PPs ASSIGNED
- PP AVAILABLE

IF ROLLOUT FLAG NOT SET, THEN

- NO PPs ASSIGNED
- PP AVAILABLE
- NO CPU ACTIVITY
- NO PPU IN RECALL (RLPW)
- NO TAPE ACTIVITY
- NO WAIT RESPONSE/LONG TERM CONNECTIONS
- FNT INTERLOCK NOT BUSY

JA IS SET WITH BIT FOR THIS JOB UNTIL 1AJ IS CALLED FOR THIS JOB.

JOB ADVANCEMENT

1AJ IS CALLED BY A CALL TO APS

THE JAV SUBROUTINE IS CALLED:

- BY SEF (SET ERROR FLAG)
- BY END (RA+1) CALL PROCESSING
- BY ROLF MTR/CPUMTR FUNCTION
- BY CCAM
- BY DCPM
- BY DPPM

CPU PROGRAM REQUESTS

(RA + 1 REQUESTS)

18	6	36
NAME	RECALL	ARGUMENT(S)

ABT	ABORT JOB
CIO	CIRCULAR I/O
CLO	CLOSE (CONVERTS TO CIO)
CPM	CONTROL POINT MANAGER
END	END JOB
LDR	REQUEST OVERLAY LOAD
LDV	REQUEST LOADER ACTION
LOD	REQUEST AUTO LOAD OF RELOCATABLE
MEM	REQUEST MEMORY
MSG	SEND MESSAGE
OPE	OPEN (CONVERTS TO CIO)
PFL	SET (P) AND CHANGE FL

RCL	PLACE PROGRAM ON RECALL
RFL	REQUEST FIELD LENGTH
RPV	RETRIEVE
RSB	READ SUB-SYSTEM BLOCK
SIC	SEND INTER-CONTROL POINT BLOCK
SPC	SPECIAL PPU REQUEST (MXPS $\leq$ QP)
TIM	REQUEST SYSTEM TIME
XJP	INITIATE SUB-CONTROL POINT (SUBCP)
XJR	PROCESS EXCHANGE JUMP REQUEST (SUBCP)
SSC	SUBSYSTEM CALL (SCP)
SSF	SUBSYSTEM FUNCTION CALL (SCP)

ALL OTHER RA+1 REQUESTS ARE ATTEMPTED TO BE SATISFIED BY A PPU PROGRAM HAVING THE SPECIFIED REQUEST NAME.

PPU MONITOR REQUESTS

ARTF	ADVANCE RUNNING TIME
IARF	INITIATE AUTO RECALL
EPRF	ENTER PROGRAM MODE REQUEST
MRAF	MODIFY RA
MFLF	MODIFY FL
SCSF	RESET CPU I STATUS
SMSF	SET MONITOR STEP
CMSF	CLEAR MONITOR STEP
ROLF	SET ROLLOUT FLAG & CHECK JOB ADV.
ACSF	ADVANCE CPU JOB SWITCH
PCXF	PROCESS CPU EXCHANGE REQUEST
ARMF	ADVANCE RUNNING TIME & MMF CLOCKS
MREF	MODIFY ECS RA
MFEF	MODIFY ECS FL

PPU MONITOR REQUESTS

(PROGRAM MODE)

MSTF	STORAGE MOVE
PDMF	PROCESS DOWN MACHINE
PMRF	PROCESS INTER-MACHINE REQUEST
MECF	MOVE ECS STORAGE
TECF	PERFORM ECS TRANSFER (FOR ECXM)

PPU REQUESTS

ABTM	ABORT CONTROL POINT
ACTM	ACCOUNTING
CCAM	CHANGE CONTROL POINT ASSIGNMENT
CEFM	CHANGE ERROR FLAG
CKSM	CHECK SUM SPECIFIED AREA
CSTM	CLEAR STORAGE
DCPM	DROP CPU
DLKM	DELINK TRACKS
DPPM	DROP PPU
DTKM	DROP TRACKS
ECSM	ECS TRANSFERS

IAUM	INTERLOCK AND UPDATE
JACM	JOB ADVANCEMENT CONTROL
LCEM	LOAD CENTRAL PROGRAM
LDAM	LOAD ADDRESS FOR MASS STORAGE DRIVERS
MXFM	MAXIMUM FUNCTION NUMBER
PIOM	PP IO VIA CPU
RCLM	RECALL CPU
RCPM	REQUEST CPU
RDCM	REQUEST DATA CONVERSION
RLMM	REQUEST LIMIT
RPPM	REQUEST PPU TO BE ASSIGNED
RSJM	REQUEST JOB SCHEDULER
RTCM	REQUEST TRACK CHAIN

SFBM	SET FILE BUSY
SFIM	SET FNT INTERLOCK
SPLM	SEARCH PERIPHERAL LIBRARY
STBM	SET TRACK BIT
TDAM	TRANSFER DATA BETWEEN JOB AND MESSAGE BUFFER
TIOM	TAPE I/O PROCESSOR
UDAM	UPDATE CONTROL POINT AREA
VMSM	VALIDATE MASS STORAGE
MXFM	MAXIMUM FUNCTION NUMBER

PP HUNG INDICATES SOMETHING WRONG WAS DETECTED DURING PROCESSING OF THESE FUNCTIONS

QUESTION SET LESSON 6

1. Who recognizes all RA+1 requests?
2. How is a PP assigned to a job to complete a RA+1 request?

LESSON 7
MTR

LESSON PREVIEW:

This lesson introduces the student to MTR, the portion of the system monitor that resides in the reserved PP, PP 0.

In addition, the monitor auxiliary PP routines, 1MA, 1MB, 1MC, will be studied.

This lesson contains a discussion of the philosophy of channel management in NOS.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Describe the processing done or initiated from the main loop of MTR.
- Explain how NOS updates its real time clock and why the clock must be updated at least every four milliseconds.
- Explain why MTR checks CPU programs for RA+1 requests.
- Explain the operation of the various DELAY intervals.
- Explain how MTR processes recalls.
- Describe the logic MTR uses to determine which control point it should request CPUMTR to assign the CPU.
- Recognize PP monitor functions that are processed by MTR.
- Discuss the roles played by the monitor auxiliary PP routines 1MA, 1MB, 1MC.
- Discuss the channel management and interlock philosophy in NOS.

REFERENCES:

NOS IMS - CHAPTER 3

SUPPLEMENTAL TEXT: CHANNEL MANAGEMENT PHILOSOPHY

NOS performance is directly related to channel usage. Consider a very minimal configuration in which there are two mass storage devices and one channel access to them.

All system, temporary, dayfile, permanent file, in fact <u>all</u> activity will occur over that one channel. This channel will almost always be busy. This causes a lot of waiting for that channel to become free.

Adding a second channel is called "dual access" and improves system performance, since the number of paths to the system, temporary, permanent files, etc., has been doubled.

NOS manages its channels and access to the equipments on them by a series of monitor functions and a channel reservation table. All channels must be reserved by the PP performing an operation over them. The PP reserving (owning) the channel is indicated in the channel reservation table. The channel reservation table is indexed by channel number and the PP having the channel is referenced by its PP index, <u>not</u> its PP number. (Remember the PP index is a reference from the beginning of the PP Communication area; thus PP 20 is actually PP index 12.)

To reserve a channel, the PP issues a RCHM (Request Channel) monitor function. If one of the channels in the request is free, it will be given to the requesting PP and an entry made in the channel reservation table. When the PP is done with the channel, it is returned via the DCHM (Drop Channel) monitor function, which clears the channel reservation table entry and issues an operation complete on the channel (if necessary). Monitor function CCHM (Check Channel) is also used in the channel reservation process. CCHM checks a specified channel and if that channel is available, assigns it to the PP and updates the channel reservation table. CCHM is primarily used in channel testing situations.

Channels should be requested and released by the use of the RCHAN and DCHAN macros. The PP program which issues these macros or manipulates channels using monitor functions RCHM and DCHM must allow for a storage move when requesting a channel.

If the channel(s) requested by a RCHAN macro or RCHM monitor function is(are) reserved, MTR will not return control to the calling PP program until the channel (or one of the channels) can be reserved for the calling PP. PP programs should not test channels by the use of the RCHAN/RCHM mechanism, as this could cause the PP to loop infinitely if the channel never becomes available. Monitor function CCHM should be used instead as it returns to the calling PP with an indication of the success or failure in reserving the desired channel and the calling PP can then process this status in an appropriate manner.

If the channel that is requested to be released by a DCHAN macro or the DCHM monitor function is not reserved to the calling PP, a HUNG PP condition will be detected.

Some monitor functions that deal with mass storage, namely, DSWM (Driver Seek Wait) and DEPM (Disk Error Processing Message) return the mass storage channel if held by the requesting PP in order to perform the desired operation without causing a channel conflict (the PP would be trying to get a channel it already has). When control is returned to the PP, the original channel or second channel in dual channel access situations will have been reserved for the PP.

These monitor functions are all performed in MTR. MTR keeps a flag to indicate that the channel reservation table has been altered and will write it back to CMR only once per pass through its main loop.

As mentioned earlier, performance is directly related to channel usage. To minimize the amount of time a PP has a channel reserved, the system (and all system programmers) follow a general rule that all "housekeeping", such as mass storage space allocation, is not done with the channel reserved. This includes such operations as a pause (PRLM) and issuing dayfile messages (DFMM). Unnecessary channel reservations prevent other programs from using that channel or may cause the system to hang on a channel reservation conflict.

NOS also uses the terms "dedicated" and "non-dedicated" channels. A dedicated channel is one whose equipment on it is typically used by only one PP routine. An example of this would be the channel to a multiplexor used by 1TD. A non-dedicated channel is one whose equipment on it is typically used by any PP routine. The prime example of a non-dedicated channel is a mass storage channel. Having only one equipment on the channel is not sufficient for that channel to be dedicated; the usage of the equipment determines the dedication of the channel as well.

NOS never assumes that a particular equipment is on a particular channel. The equipment/channel relationship is specified through the EQ CMRDECK entry. All code that is written to use channels follows the conventions established by common decks COMPCHL and COMPCHI. These decks allow code to be written in a general way with the actual channel number inserted into the instructions through a preset operation at execution time.

MAIN LOOP

1. CHECK PP 1 FOR REQUEST FROM DSD.
2. CHECK EACH PPU OUTPUT REGISTER (OR) FOR REQUEST. PPUs ARE PROCESSED IN ORDER (2,...10,11,20,21,...,31)
3. ADVANCE CLOCK (AVC).
4. WRITE CHANNEL STATUS TABLE TO CMR (IF NECESSARY).
5. CHECK FOR PP SATURATION (CPS).
6. CHECK CENTRAL PROGRAMS (CCP).
7. CHECK CPU SWITCH.
    - READ CPUMTR CX+2.
    - IF IMMEDIATE SWITCH, SEARCH FOR NEXT JOB (SNJ).
    - IF SINGLE CPU OR (CX+2)=0, CHECK ELAPSED TIME SCANNERS (CET).
8. CHECK CYBER 170 STATUS & CONTROL REG. (CSC).
9. CHECK PROGRAM RECALL BY READING CPUMTR XS. RECALL PROGRAM USING RCPM IF TIME TO RECALL.
10. UPDATE MTR CYCLE TIME STATISTICS IN SDOL.
11. ADVANCE CLOCK (AVC).
12. GO TO STEP 1.

ADVANCE CLOCK (AVC)

ENTRY AT LEAST EVERY FOUR MILLISECONDS

1. ADVANCES MILLISECOND CLOCK (RTCL) (TIM).
2. READS CPU 0 EXCHANGE REQUEST -
   CPUMTR CX - IF REQUEST PRESENT, DOES PCXF.
3. READS CPU 1 EXCHANGE REQUEST -
   CPUMTR CX+1 - IF REQUEST PRESENT, DOES PCXF + 10000.
4. IF NEITHER PCXF PERFORMED, THE TIME AND DATE IS ADVANCED (AVT).
5. IF PCXF DONE, AVC LOOPS UNTIL REQUEST HAS BEEN CLEARED BY CPUMTR (GO TO 2).
6. ADVANCES RUNNING TIMES ON EACH CPU.

   ARTF + 10000 FOR CPU 1
   ARTF or ARMF FOR CPU 0

MTR

<u>ADVANCE TIME</u> (AVT)

1. READS MSCL TO HAVE INTERNAL COPY OF IT. THIS ALLOWS MTR TO REACT TO DELAY CHANGES.

2. READS AND UPDATES TIME AND DATE WORDS.

   JDAL
   PDTL
   TIML
   DTEL

<u>UPDATE REAL TIME CLOCK</u> (TIM)

1. UPDATES RTCL EVERY MILLISECOND BY READING CLOCK CHANNEL 14 - IAN 14 - AND UPDATING RTCL WHEN CLOCK OVERFLOWS.

CHECK PP SATURATION (CPS)

1. READ PPAL FROM CMR.
2. IF BYTE 4 OF PPAL = 0, THEN NO PPs ARE AVAILABLE (DEFINITION OF SATURATION) AND RETURNS TO THE MAIN LOOP.
3. IF A PP REQUEST IS PRESENT, IT IS PROCESSED.
4. IF NO PP REQUESTS ARE PENDING, IS A JOB WAITING FOR A PP (CPUMTR WORD XS) IF SO, RECALL IT (RCLM) OTHERWISE RETURN.

CHECK STATUS AND CONTROL REGISTER (CSC)

1. READ FIRST BYTE ON SCR CHANNEL FOR EACH PPS.
2. IF ERROR PRESENT, CALL 1MB TO FORMAT SCR ERROR TO THE ERROR LOG.
3. THE INHIBIT BIT IS SET IN SCRL TO KEEP ONLY ONE COPY OF 1MB ACTIVE. 1MB WILL CLEAR THIS BIT WHEN COMPLETED.
4. 1MB LOGS FIRST TEN ERRORS PER HOUR. AFTER TEN ERRORS, LOGGING IS DISABLED UNTIL NEW HOUR IS BEGUN.

CHECK CENTRAL PROGRAM (CCP)

1. CHECK STORAGE MOVE IN PROGRESS. IF SO, JUMP TO MST TO COMPLETE THE STORAGE MOVE.

2. ADVANCE CLOCK (AVC).

3. READ CPU 1 ACPL.
   IF IDLE PROGRAM (ACPL BYTE 2 = 0), READ CPU 0 ACPL.
   IF IDLE PROGRAM, RETURN.

4. READ STSW FOR THE ACTIVE CP.

5. IF SYSTEM CP, TRY NEXT CPU IF ANY.

6. IF SUBCONTROL POINT, RETRIEVE ITS RA.

7. READ (RA+1) INTO OR
   IF NON-ZERO, ASK CPUMTR TO CHECK IT.

8. REPEAT WITH NEXT CPU IF ANY.

CHECK ELAPSED TIME AND CALL SCANNERS (CET)

1. COMPARE INTERNAL CLOCK AGAINST PP/AUTO RECALL TIME INTERVAL (BYTE 2 OF MSCL: DELAY AR).

   IF TIME EXPIRED, CALL PPL (PROCESS PP RECALLS).

2. COMPARE INTERNAL CLOCK AGAINST CPU JOB SWITCH TIME INTERVAL (BYTE 4 OF MSCL: DELAY CS).

   IF TIME EXPIRED, CALL SNJ (SELECT NEW JOB).

3. OTHERWISE, CHECK CENTRAL PROGRAMS (CCP).

PROCESS PP RECALLS (PPL)

CHECKS UP TO 12_{10} CONTROL POINTS PER PASS (OR ACTUAL IF LESS THAN 12).

1. READ CP'S STSW AND RLPW.
2. IF JOB ADVANCE SET (STSW BIT 53), GO TO NEXT CP AND REPEAT WITH STEP 1.
3. IF NOT IN X OR I RECALL STATUS (STSW BITS 58 OR 57) OR IF IN RECALL STATUS AND NO RA+1 REQUEST, CHECK RLPW.
4. IF NO RLPW, GO TO NEXT CP AND STEP 1.

   IF A RLPW REQUEST IS PRESENT, READ JCIW. IF ROLLOUT SET (BYTE 2 OF JCIW), RETURN.

   IF ROLLOUT NOT SET, REQUEST A PPU FOR THE RLPW REQUEST. IF NONE AVAILABLE, REQUEUE INTERNALLY.
5. IF IN RECALL AND RA+1 REQUEST PRESENT -

   IF I RECALL, INITIATE AUTO RECALL IARF.

PROCESS PP RECALLS (PPL)

6. IF X RECALL, CHECK (RA+1) FOR RCLP.

   IF NOT RCLP, CHECK RLPW (STEP 4).

   IF RCLP, CHECK ADDRESS FOR COMPLETION BIT (BIT 0).

   IF COMPLETION BIT NOT SET, CHECK RLPW (STEP 4).

   IF COMPLETION BIT IS SET, RECALL CPU WITH A RCLM AND RETURN.

SELECT NEW JOB (SNJ)

ENTER WITH CPU TO SELECT.

1. READ ACPL FOR CURRENT JOB.
2. READ JOB'S JCIW TO GET CPU PRIORITY (BYTE 0 OF JCIW).
3. READ CPUMTR CR.
   A BIT IS SET IN CR IF A CONTROL POINT NEEDS THE CPU.

   LOOKS AT 12 CPs AT A TIME - FIRST 12, SECOND 12, REMAINDER.
4. IF A WAITING CP IS FOUND, ITS JCIW IS READ.
5. COMPARE WAITING CP'S CPU PRIORITY AGAINST CURRENT JOB'S (BASE) CPU PRIORITY.

   WAITING   BASE - FORGET IT! (STEP 6)
   WAITING = BASE - IF NO SELECTION MADE OR CP IS HIGHER THAN CURRENT GO TO STEP 7. (THIS ASSURES ROTATION OF EQUAL PRIORITY CPs)
   WAITING   BASE - GO TO STEP 7.
6. ADVANCE TO NEXT WAITING CP (STEP 4).

SELECT NEW JOB (SNJ)

7. IF JOB IN SAME CPU (COMPARE JCIW BYTE 4) OR NO CPU SPECIFIED, RESET BASE, IDENTIFYING CANDIDATE JOB, AND ADVANCE TO NEXT WAITING CP (STEP 4).

8. WHEN LOOPING HAS FINISHED, THE NEW BASE IS SET AND THE CPU REQUESTED TO BE GIVEN TO THAT JOB VIA ACSF.

MTR PP FUNCTION REQUESTS

CCHM	CHECK CHANNEL
DCHM	DROP CHANNEL
DEPM	DISK ERROR PROCESSOR
DEQM	DROP EQUIPMENT
DFMM	PROCESS DAYFILE MESSAGE
DRCM	DRIVER RECALL CPU
DSRM	DSD REQUESTS
DSWM	DRIVER SEEK WAIT
EATM	ENTER EVENT TABLE
ECXM	ECS TRANSFER

MTR PP FUNCTION REQUESTS (Continued)

PRLM	PAUSE FOR STORAGE RELOCATION
RCHM	REQUEST CHANNEL
REMM	REQUEST EXIT MODE
REQM	REQUEST EQUIPMENT
RJSM	REQUEST JOB SEQUENCE NUMBER
ROCM	ROLLOUT CONTROL POINT
RPRM	REQUEST PRIORITY
RSTM	REQUEST STORAGE
RSYM	REQUEST SYSTEM
SCPM	SELECT CPU
SEQM	SET EQUIPMENT PARAMETERS
TGPM	PROCESS REQUEST FOR POT CHAIN
TSEM	PROCESS TELEX REQUEST

HUNG PP INDICATES SOMETHING WRONG WAS DETECTED WHEN PROCESSING THESE FUNCTIONS.

MONITOR AUXILIARY PPU ROUTINES

1MA

- REQUEST STORAGE (MEM/RFL)
- MESSAGES (MSG)

    DAYFILE
    JOB DAYFILE
    ERROR LOG
    ACCOUNT

- SYSTEM CONTROL POINT FACILITY (SSC/SSF)

1MB

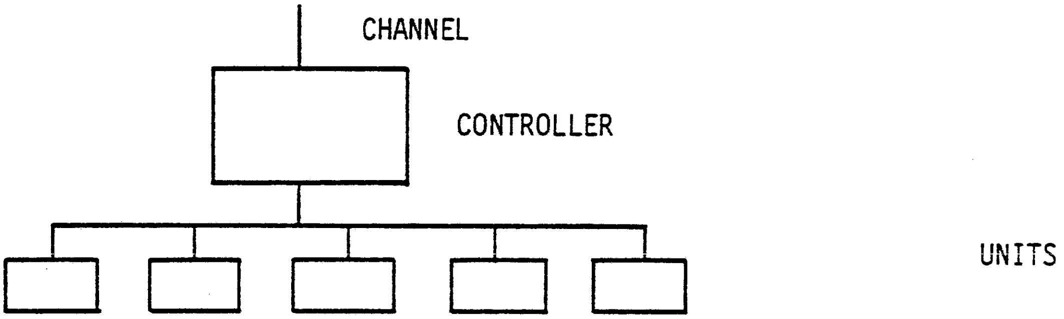
- REPORT STATUS AND CONTROL REGISTER ERRORS

1MC

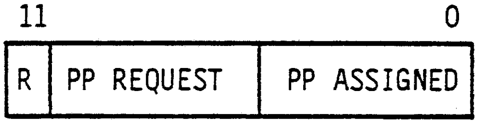
- REPORT ECS PARITY ERRORS

CHANNEL THEORY

- PERFORMANCE
- INTERLOCK
    - PSEUDO CHANNELS
    - CHANNELS TO I/O EQUIPMENT



- CHANNEL RESERVATION TABLE



ONE BYTE PER
CHANNEL

- PP PROGRAM RESERVES CHANNEL VIA MACROS.

RCHAN
DCHAN

- PP MONITOR FUNCTIONS MANIPULATING CHANNELS

RCHM
DCHM
CCHM
DEPM
DSWM

- COMMON DECKS MANIPULATING CHANNELS

COMPCHI
COMPCHL

- NOS DOES NOT HARDCODE CHANNELS

COMMON DECK TECHNIQUES ALLOW PROGRAMS TO MODIFY THEMSELVES TO USE PROPER CHANNELS DURING PRESET OPERATIONS.

THIS ALLOWS FLEXIBILITY IN CONFIGURING SYSTEM WITHOUT REASSEMBLY OF SOFTWARE.

QUESTION SET LESSON 7

1. Explain what each of the scan times (delay cycles) accomplish?

2. How does MTR take a control point out of periodic recall?

3. What is the system mechanism that keeps two PPs from getting the same channel?

LESSON 8
STORAGE MOVE

LESSON PREVIEW:

This lesson details the process whereby the NOS Operating System obtains a contiguous block of memory to serve as a program's (control point's) Field Length. This process is known as "storage move" and is performed by the system monitors in response to a request for storage via the RSTM monitor function recognized by MTR.

This presentation covers only the central memory aspect of storage move. User ECS is also storage moved using the same logic, but with different monitor functions to interlock and adjust the RAE and FLE.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Describe how central memory is allocated among control points and CMR (central memory resident).
- Explain what is meant by "unused field length" and be able to compute the unused field length for each control point.
- Explain how the unused field length is used in the storage request (storage move) process.
- Describe how the storage move control word (CMCL) is set up to manage the movement of control point field lengths to obtain the required amount of memory as a contiguous block.
- Explain the logic used for advancing the storage move from one control point to the next.
- Describe the processing done by MTR and CPUMTR when moving a control point's field length.
- Explain why PPs assigned to a control point must pause for a storage move.

SUPPLEMENTAL TEXT: STORAGE MOVE

As you know (or should know by now), NOS allocates central memory into Central Memory Resident and Control Points. Each control point field length contains consecutive memory locations beginning with the reference address (RA) continuing through the field length (FL). Control point field length is allocated in central memory in ascending control point order; that is, control point 1's FL is a contiguous block before control point 2's FL, and so forth. NOS stores reference addresses and field lengths in multiples of 100B words. This allows RA and FL values to be contained in a 12-bit byte, or more accurately, in one PP word.

Memory Request

Memory for a control point (field length) is requested for the job at the control point by a PP. The CPU program may issue RA+1 requests MEM and RFL to specify a field length, but these requests are trapped by CPUMTR and the request for memory is made by CPUMTR's PP auxiliary 1MA. Monitor request RSTM (Request Storage) is issued by a PP to obtain storage for the control point to which the PP is assigned. RSTM is processed by MTR. This mechanism provides appropriate interlocks to prevent two or more control points from deadlocking themselves attempting to obtain memory.

If not enough memory is available to satisfy a RSTM request, MTR returns a "storage not available" status to the requesting PP. The PP may then take appropriate action to retry the request or call the scheduler to analyze the situation and possibly rollout lower priority jobs so that the desired field length is available.

If enough free memory is available to satisfy the request, it is usually necessary to collect the unused memory into a contiguous block at the requesting control point's relative position of control point memories. This is done by moving other control point memories within central memory, so that when the requesting control point is given the memory requested, the control point's memory will be a contiguous block and in ascending control point order. This technique is called STORAGE MOVE.

Determining the Move

MTR builds a table of the unused field length for each control point. Each entry in the Table of Unused Field Length (TUFL) is built using the formula:

TUFL(i) = RA(i+1) - ¼RA(i) + FL(i)½
where i is the control point number.

As MTR builds the TUFL, it also sums the TUFL entries to determine the amount of available memory (AM). This value will be stored in CMR location ACML.

If a control point requests a decrease in FL, the amount of decrease is taken from the current FL, and RSTM completes. If a job at the last control point, however, decreases its memory requirement, its field length will be moved to the end of memory. There will never be unused FL for the last control point.

If a control point requests additional memory and the unused field length for the control point will satisfy the request, the control point is given the memory increase and RSTM completes.

If the control point unused field length does not satisfy the request, and there is not enough available memory (AM) to satisfy the request, RSTM will be completed with a "storage not available" status. If there is memory available to satisfy the request, then storage move(s) will be done to collect the required amount of unused field length at the control point's relative position of control point field lengths. It should be noted at this time that the purpose of any storage movement is to move control point field lengths so that the unused field length of the requesting control point will satisfy the control point's memory request.

There are three cases of storage movement: 1.) the previous control point; 2.) control points above the requestor; and 3.) all control points. In the following discussion CPi is the requesting control point and MI is the memory increase requested. A control word is set up by MTR to control the movement. This word resides in direct cells MM through MM+4 and is written into CMR at location CMCL. This word has the format:

MM	MM+1	MM+2	MM+3	MM+4
moving CP	move amount	lower CP	request CP	request PP

The first case to consider is the previous control point: CPi-1. Is MI ,= TUFL(CPi-1) + TUFL (CPi)? If not, go on to the next case. If so, the requesting control point CPi's field length is moved downward to get the necessary memory in a continuous block. The CMCL control for this move would be:

MM	MM+1	MM+2	MM+3	MM+4
CPi	TUFL (CPi-1)	CPi-1	CPi	PPz

The second case to consider is the control points above the requestor: CPi+1, CPi+2,...,CPn. The TUFL of each control point is added to the TUFL(CPi) until the last TUFL is reached or the request is satisfied. If the memory increase is not satisfied before reaching the last control point, then the next (and final) case is considered. If the request can be satisfied, the control point field lengths above the requestor that are necessary to be moved are moved upward to get the memory in the contiguous block. The CMCL control for this move would be:

```
  MM        MM+1      MM+2      MM+3      MM+4
+---------+---------+---------+---------+---------+
| CPi+x   | amount  | CPi     | CPi     | PPz     |
|         | CPx     |         |         |         |
+---------+---------+---------+---------+---------+
```

The amount (MM+1) will vary for each control point being moved.

The last case is to consider all control points. After determining the amount of memory available by moving control points above the requestor, MTR begins with the first control point below the requestor working down to the first control point, adding the control point's TUFL until the request can be satisfied. This case must be successful since AM .= MI. If this case does not satisfy the request, MTR will hang to preserve the error for subsequent analysis. The CMCL control for this move will be:

```
  MM        MM+1      MM+2      MM+3      MM+4
+---------+---------+---------+---------+---------+
| CPi+x   | amount  | CPi-y   | CPi     | PPz     |
|         | CPx     |         |         |         |
+---------+---------+---------+---------+---------+
```

The amount (MM+1) will vary for each control point being moved.

Advancing the Move

As each control point field length is moved, MM is adjusted to reflect the next control point to be moved. MM is decremented by 200B when doing an upper move until the the requesting CP (MM+3) is reached. If the requesting CP (MM+3) and the lower CP (MM+2) are not equal, then MM is set to the lower CP. MM will then be advanced by 200B and control points moved until the requesting CP FL has been moved. The amount of the move (MM+1) is positive for upper moves or negative for downward moves and is the amount the moving control point (MM) must be moved.
The reason the upper move is done first is because it is faster to move upward using the CMU (Compare Move Unit) than downward.

Move Storage Subroutine (MST)

The logic for doing storage moves is found in MTR subroutine MST. The flow is as follows.

1. Issue ARTF+10000 function to CPUMTR; this interlocks the system for storage move.

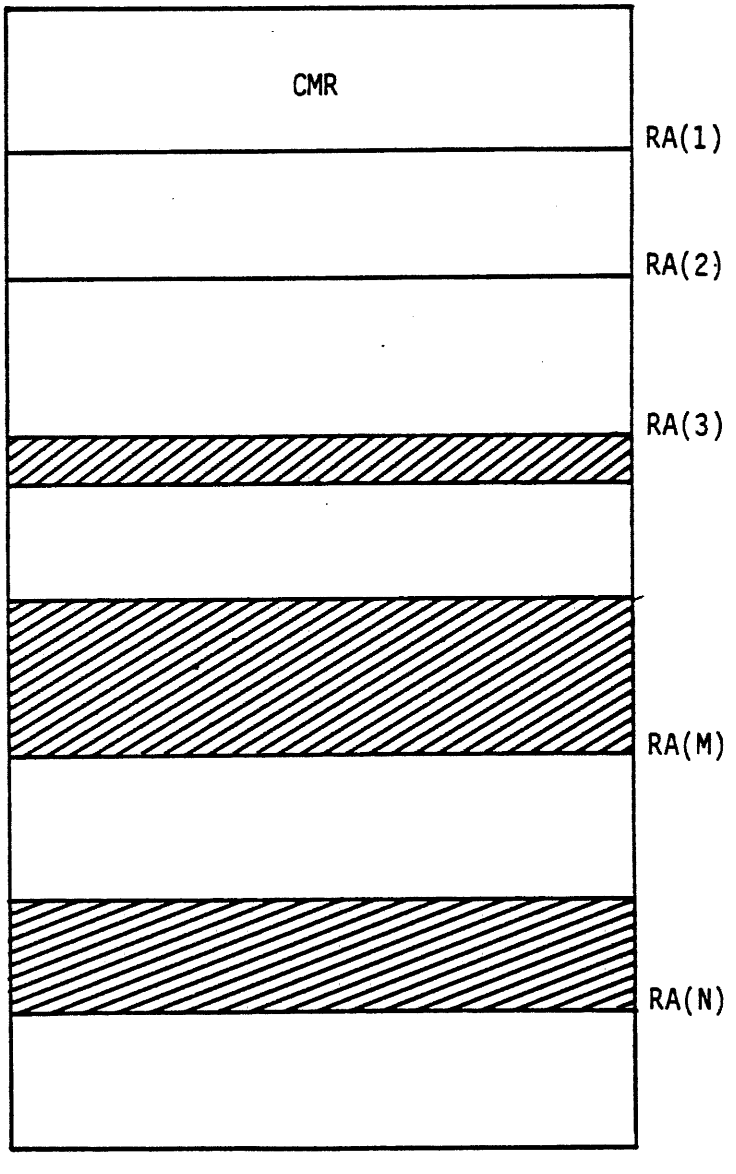
2. Determine the move, setting up direct cells MM - MM+4.

3. If the moving control point (from MM) FL = 0, then
   - modify RA via MRAF
   - advance the move
   - quit when all CPs have been moved
4. If FL non-zero, then
   - issue DCPM on the CP (can not move active CP)
   - save CPU status (byte 0 of STSW)
   - issue MSTF to CPUMTR
   - modify PP scan loop to bypass the requesting PP
   - exit MST
   - CPUMTR sets its location SM from CMCL
   - CPUMTR moves the storage using

     ECS
     CMU
     register-to-register
5. The completion of the storage move by CPUMTR is detected by MTR subroutine CCP (Check Control Point). If the move has completed, control returns to subroutine MST. Otherwise, MTR continues doing other business.
6. After the control point has been moved:
   - clear storage move control
   - modify RA via MRAF
   - reset CPU status, restarting CPU if necessary
   - advance move
   - quit when all CPs have been moved

In order to be moved, a control point with PPs must have those PPs in a pause condition, that is, PPs have issued a PRLM (Pause), DSWM (Driver Seek Wait), DFMM (Dayfile Message), RCHM (Request Channel), or RSTM (Request Storage) MTR function.

STORAGE MOVE



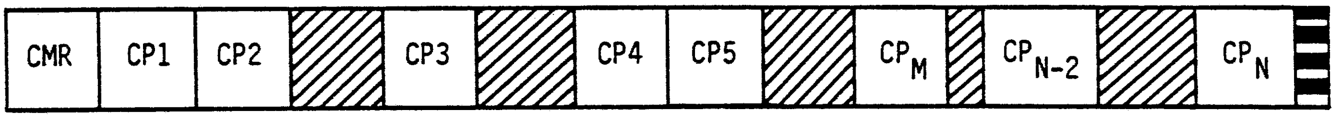
CENTRAL MEMORY

//// = UNUSED

LAST CP ALWAYS
AT END OF MEMORY

RSTM = REQUEST STORAGE

STORAGE MOVE



$$TUFL\ (I) = RA\ (I+1) - \left[RA\ (I) + FL\ (I) \right]$$

$$AM = \sum_{J=1}^{N} TUFL\ (J)$$

AVAILABLE MEMORY

(ACML)

MOVE CONTROL

MM	MM+1	MM+2	MM+3	MM+4
MOVING CP	AMOUNT	LOWER CP	REQUEST CP	REQUEST PPU

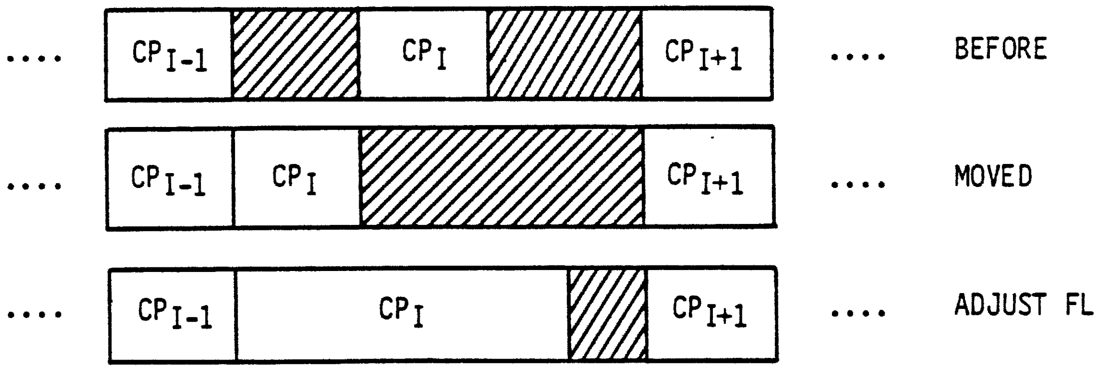
STORAGE MOVE

AM = AVAILABLE MEMORY

CP_I = REQUESTING CP

M_I = AMOUNT OF INCREASE

1. CP_I DECREASES - ADJUST FL
   - EXCEPT IF CP_N, THEN MUST MOVE CP_N TO END OF MEMORY
2. CP_I <u>INCREASES</u> - TUFL (I) $\geq$ MI, ADJUST FL
3. AM $\geq$ MI, IF NOT, "STORAGE NOT AVAILABLE"
4. MI $\leq$ TUFL (I) + FL (I-1)

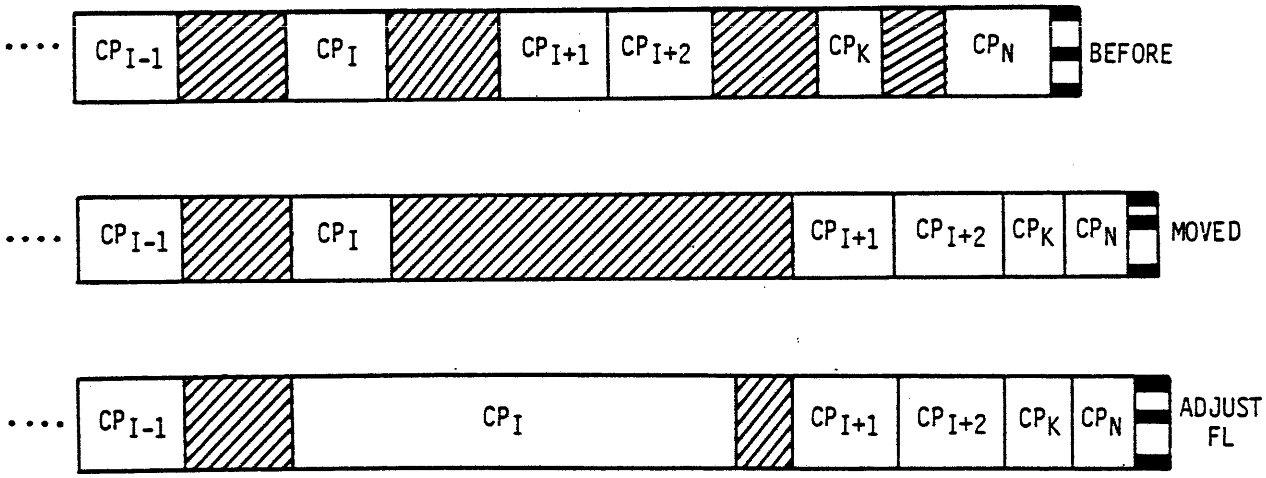


MM	MM+1	MM+2	MM+3	MM+4
CP_I	-TUFL (I-1)	CP_{I-1}	CP_I	

CMCL

STORAGE MOVE

5. $$AM = \leq \sum_{J=1}^{N} \text{TUFL (J)} \quad \text{CONTROL POINTS ABOVE REQUESTOR}$$

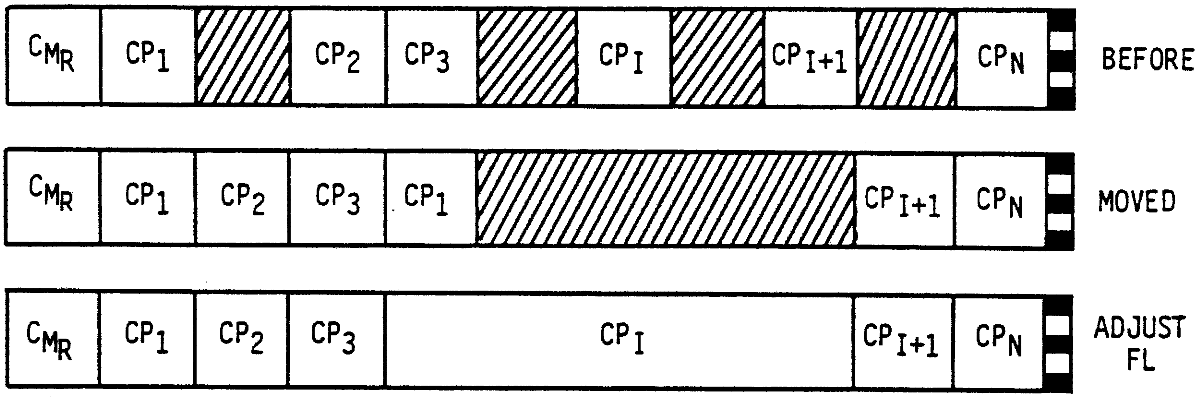


MM	MM+1	MM+2	MM+3	MM+4
CP_K ..	TUFL(X)	CP_I	CP_I	

STORAGE MOVE

6.

$$MI = \leq \sum_{J=1}^{N} TUFL\ (J) + \sum_{K=I-1}^{I} TUFL\ (K)$$ ALL CONTROL POINTS



MM	MM+1	MM+2	MM+3	MM+4	
CP_{I+1}	$TUFL_{(I+1)}$	CP_2	CP_I		FIRST
CP_2	$-TUFL_{(1)}$	CP_2	CP_I		SECOND
CP_3	$-\sum$ TUFL Now TUFL(2)	CP_2	CP_I		THIRD
CP_I	$-\sum_{J=1}^{I-1} TUFL(J)$	CP_2	CP_I		LAST

STORAGE MOVE

1. INTERLOCK SYSTEM WITH ARTF & 10000

2. IF MOVING CP HAS FL=0
    - MODIFY RA VIA MRAF (MREF IF ECS)
    - ADVANCE MOVE

3. IF MOVING CP HAS FL≠0
    - DCPM (CAN'T MOVE ACTIVE CP)
    - SAVE CPU STATUS
    - ASK CPUMTR TO MOVE VIA MSTF (MECF IF ECS)
    - MODIFY PPU SCAN TO SKIP THIS PPU (REQUESTING)
    - EXIT TO WAIT FOR MOVE

4. CPUMTR MOVES USING FASTEST MEDIUM
    - ECS
    - CMU
    - REGISTER TO REGISTER

5. MTR SUBROUTINE CCP (CHECK CONTROL POINT) FINDS MOVE COMPLETE
    - CLEAR MOVE CONTROL
    - MODIFY RA VIA MRAF (MREF IF ECS)
    - RESET CPU STATUS
    - RESTART CPU IF NECESSARY
    - ADVANCE MOVE

6. AFTER MOVING IS DONE,
    - MODIFY REQUESTING CP FL VIA MFLF (MFEF IF ECS)
    - COMPLETE RSTM

7. CP WITH ASSIGNED PPUs MUST HAVE PPUs IN PRLM (PAUSE), DSWM (DRIVER SEEK WAIT), OR DFMM (DAYFILE MESSAGE) IN ORDER TO BE MOVED

QUESTION SET LESSON 8

With the control point usage described below, answer the following questions on storage move:

1. Build a TUFL (Table of Unused Field Length).
2. How much memory is available? Where is this value stored?
3. What will CMCL be (MM - MM+4) if control point 7 wants its field length to be 35500 words? Assume the RSTM request is made by PP 5.
4. What will be the RA and FL values for each control point after the move in c. takes place?
5. Machine size is 200000. Where and in what form is this value stored?
6. What happens if the last (not system) control point makes a RSTM request?
7. Why are the last 100 words of a 262K CM configuration not used by NOS?

CP	RA/100	FL/100	TUFL	RA/100	FL/100
1	300	20	------	------	------
2	320	320	------	------	------
3	640	0	------	------	------
4	650	0	------	------	------
5	700	17	------	------	------
6	1000	0	------	------	------
7	1010	50	------	------	------
10	1130	0	------	------	------
11	1200	100	------	------	------
12	1300	10	------	------	------

CP	RA/100	FL/100	TUFL	RA/100	FL/100
13	1310	50	------	------	------
14	1400	0	------	------	------
15	1410	5	------	------	------
16	1600	200	------	------	------

LESSON 9
PP RESIDENT

LESSON PREVIEW:

This lesson describes the Peripheral Processor Resident (PPR) which resides in each Pool PP. PPR provides the communication link between the Pool PP and the system monitors (CPUMTR and MTR) and jobs at control points. The idle loop in PPR will always be running in the Pool PP if no other PP program is executing in that PP.

This lesson also introduces the student to the concept of zero level overlays (location free routines). These PP routines are used as subroutines by other PP programs.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Describe the operations performed in the main loop of PP Resident; namely, checking the PP's input register (IR) for a PP routine to be loaded into the PP and executed. The student has already seen the actual PP assignment in the lesson on CPUMTR. This discussion of this topic should be directed to what the PP does when it is assigned.
- Discuss the individual subroutines that comprise PP Resident:
  - Peripheral Library Loader
  - Monitor Function Requests
  - Pause for Storage Relocation
  - Request and Drop Channel
  - Issue Dayfile Message
  - Execute Routine
- Discuss programming conventions for interfacing a PP routine to PPR. This discussion should cover the following items:
  - Usage of Direct Cells and their symbols
  - Usage of EXECUTE, MONITOR, PAUSE macros
  - Memory allocation of Pool PPs
  - PP overlaying techniques
- Describe the role 1DD (Dump Dayfile Buffer) plays in the system.
- Describe the role 1RP (Restore PPR) plays in the system.
- Discuss the relocation techniques used by location free routines (zero level overlays).

REFERENCES:

NOS IMS - Chapter 4 and 29

ASSIGNMENT:

Read the text on "System Interface Techniques" that is found in this Handout. Obtain a listing of location free routine OVJ and common decks COMPRLI and COMPREL.

SUPPLEMENT TEXT: SYSTEM INTERFACE TECHNIQUES

The following techniques or rules apply when writing central processor and peripheral processor routines for system interface.

PP routines should be efficient and their use held to a minimum.

PP programs that interface with CPU programs should have some form of validation to ensure proper calling procedures or parameters. An example of this is CIO which checks all buffer parameters before performing any of the tasks requested. Great care should be taken to ensure that errors in arguments set up by the CPU program do not cause the PP routine to destroy an area other than the local field length of the calling program.

PP routines waiting for some system resource to become available always pause for storage relocation to allow other system processing to proceed.

The use of other PPs to assist in performing the assigned task is a practice that should be used only when absolutely necessary and with great caution. The fact that other PPs may not be available must be considered. The situation in which several PPs are waiting for pool processors should be prevented. The only sure way to do this is not to request helper PPs.

PP overlays can be used simply and without difficulty. The advantage of overlays is that a minimal amount of coding is loaded (from disk or CMR) every time a program is executed. Areas of programs that are used infrequently can be made overlays (for example, error processors). Certain system overlays are available for use by any program and are designed to be location-free. For use of the overlays, refer to the COMPASS SEGMENT or IDENT pseudo- ops.

PP memory direct cells 70 through 73 and 75 through 77 are set when PPR is loaded (at deadstart). Care should be taken not to destroy these cells, as these locations are not initialized after each PP program load. Direct cell 74 (CP) is set by PPR when a program is loaded through an input register request.

PP routines preparing for input/output to allocatable devices (mass storage) should request the mass storage space needed before reserving the channel. Also, when input/output is complete, the channel should be dropped before indicating which area of mass storage was not used.

PP routines doing input/output to a device should perform as much housekeeping as possible before the operation is initiated. Unnecessary reservation of channels prevents other programs from using that channel for input/output.

Buffers used by PP and CPU programs should be defined by EQU statements at the end of the program text and not made part of the text by using BSS statements. This eliminates the loading of the buffer area.

PP programs or overlays are loaded from disk. The last sector of a program or overlay must be loaded at an address below that which causes wrap-around.

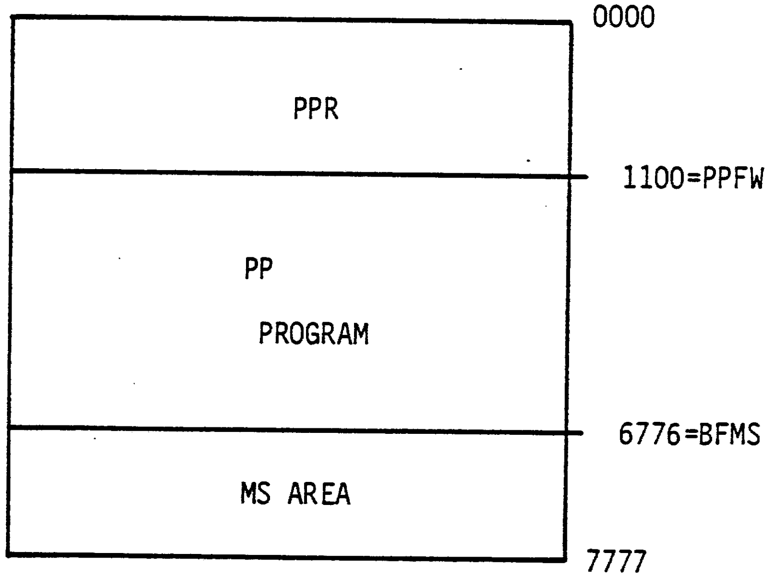
System programmers should follow the guidance offered by any notes and cautions provided for PP and CPU macros and monitor functions. An example of this would be not pausing with a non-dedicated channel reserved.

PERIPHERAL PROCESSOR RESIDENT

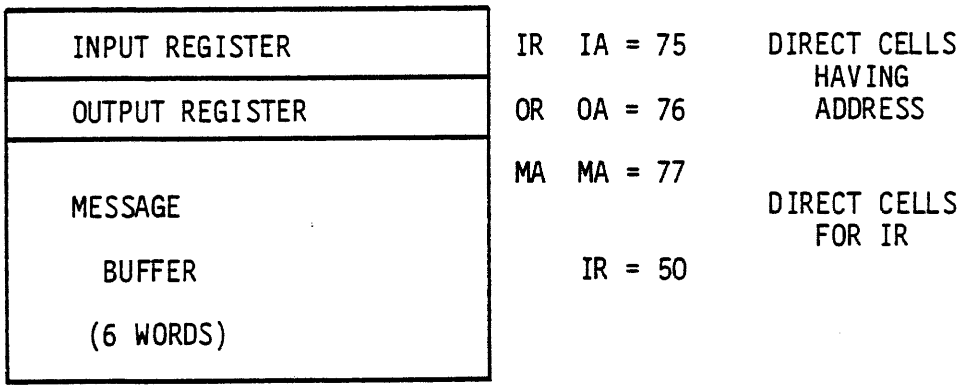
PPR

- PROVIDES COMMUNICATION LINK BETWEEN PP's AND CONTROL POINTS.
- PP IDLE PROGRAM
- LOADER OF PP PROGRAMS
- SOURCE OF COMMONLY USED SUBROUTINES
- PROVIDES CONSISTENCY OF PPU ACTIVITIES
- LOADED DIRECTLY INTO PPU MEMORY AT DEADSTART AND IS NOT CHANGED.
- MTR, DSD, AND SOME OTHERS DO NOT CONTAIN A COPY OF PPR.

PPU MEMORY



PPR



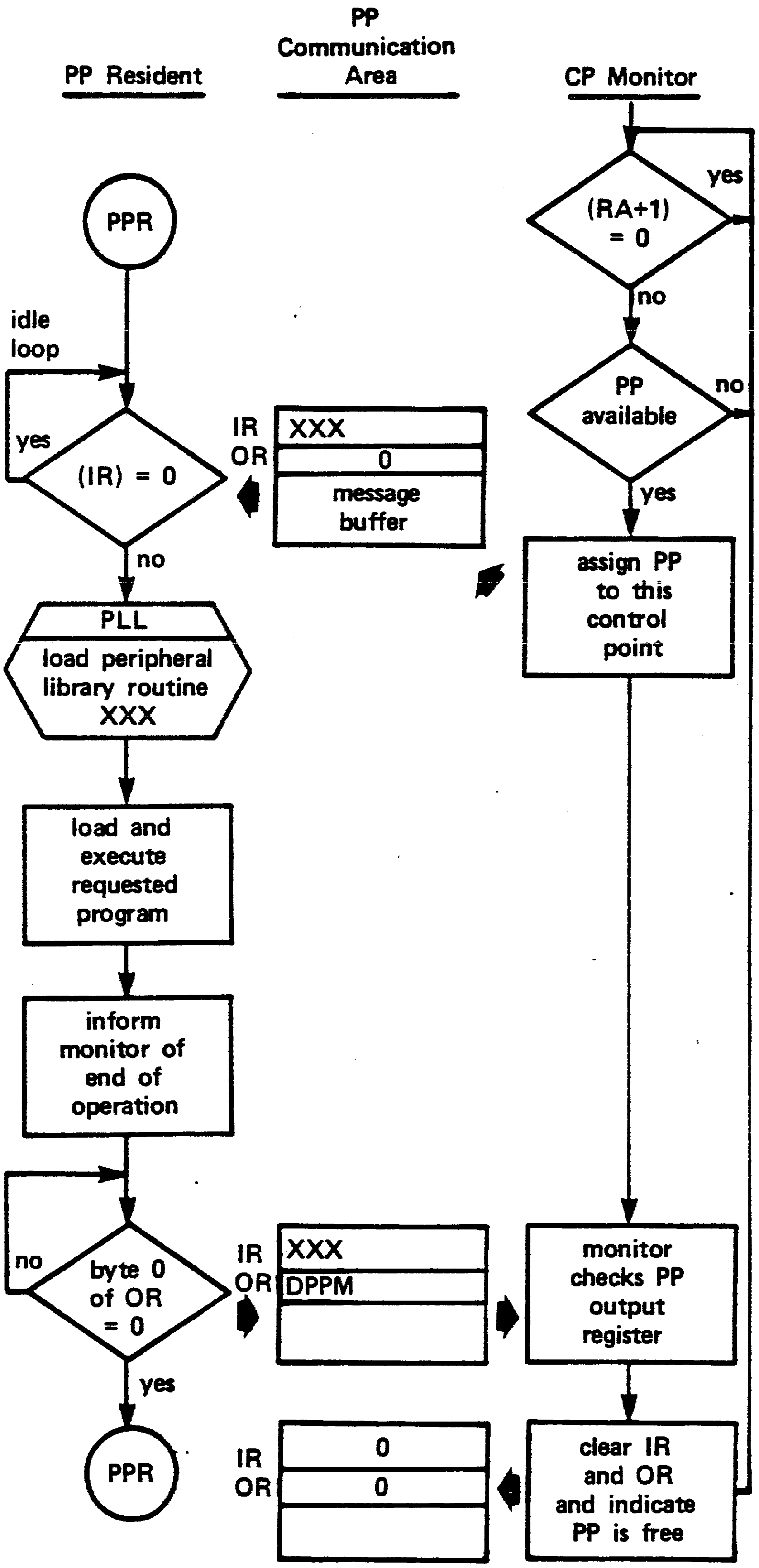
PPR IDLE LOOP CHECKS THE INPUT REGISTER (IR) FOR A REQUEST OF A PP PROGRAM TO BE LOADED.

CPUMTR WILL ENTER THE INPUT REGISTER WITH THE ROUTINE NAME TO BE LOADED INTO THIS PPU. THE EXISTING PP ROUTINE MAY REWRITE THE INPUT REGISTER AND JUMP TO THE IDLE LOOP TO RE-LOAD THE PPU WITH A DIFFERENT ROUTINE.

THE DPPM (DROP PPU) MONITOR FUNCTION WILL CLEAR THE INPUT REGISTER.

THE OUTPUT REGISTER IS USED TO MAKE CALLS TO CPUMTR AND MTR.

THE MESSAGE BUFFER USUALLY CONTAINS PARAMETERS FOR THE FUNCTIONS REQUESTED THROUGH THE OUTPUT REGISTER.


PP Resident
PP Communication Area
CP Monitor
PPR
idle loop
yes
(IR) = 0
no
PLL
load peripheral library routine XXX
load and execute requested program
inform monitor of end of operation
no
byte 0 of OR = 0
yes
PPR
IR
OR
XXX
0
message buffer
IR
OR
XXX
DPPM
IR
OR
0
0
(RA+1) = 0
yes
no
PP available
no
yes
assign PP to this control point
monitor checks PP output register
clear IR and OR and indicate PP is free

PP DIRECT CELL SYMBOLS

LOCATION	SYMBOL	USAGE
0	T0	TEMPORARIES
1	T1	
2	T2	
3	T3	
4	T4	
5	T5	
6	T6	
7	T7	
10-14	CM	CM WORD BUFFER
15	LA	PPU ROUTINE LOAD ADDRESS
16	T8	TEMPORARY
17	T9	TEMPORARY
50-54	IR	INPUT REGISTER
55	RA	REFERENCE ADDRESS
56	FL	FIELD LENGTH
70	ON	CONSTANT 1
71	HN	CONSTANT 100
72	TH	CONSTANT 1000
73	TR	CONSTANT 3
74	CP	CONTROL POINT ADDRESS
75	IA	INPUT REGISTER ADDRESS
76	OA	OUTPUT REGISTER ADDRESS
77	MA	MESSAGE BUFFER ADDRESS
100	DRSW	DRIVER SCRATCH
101	WDSE	WRITE ERROR PROCESSING ADDRESS
102	ERXA	RDS/WDS EXIT ADDRESS
103	RDCT	READ COUNT
104	STSA	DEVICE STATUS
105	UERR	ERROR PROCESSING
106	SLM	SECTOR LIMIT
107	MSD	MASS STORAGE DRIVER DESIGNATOR
110	CHRV	CHANNEL RESERVATION INDICATOR

PPR
SUBROUTINES

1. PPR IDLE LOOP

   PPR READS IR FROM PP COMMUNICATION AREA EVERY 128 MICROSECONDS.

2. PLL PERIPHERAL LIBRARY LOADER

   WHEN IR=0, PLL REQUESTS CPUMTR TO SEARCH THE PLD (PERIPHERAL LIBRARY DIRECTORY) FOR THE REQUESTED PP PROGRAM. THIS IS DONE VIA SPLM MONITOR FUNCTION.
   IF LIBRARY ALREADY SEARCHED, UPPER 6 BITS OF IR=0
   IF FOUND, THE ROUTINE IS LOADED INTO THE PP.
   IF NOT FOUND, ROUTINE SFP IS LOADED.
   IF THE ROUTINE IS NOT PART OF SFP, THE DIAGNOSTICS

	MONITOR CALL ERROR	(IF RA+1 CALL)
OR	XXX NOT IN PP LIB	(IF DIRECT PP CALL)

   ARE ISSUED.

3. LEP LOAD ERROR PROCESSOR

   LOADS MASS STORAGE ERROR PROCESSOR ROUTINES (FROM CM) - PART OF THIS PROCESS RESIDES IN CMR.

4. FTN MONITOR FUNCTION REQUESTS

THE MONITOR FUNCTION (CPUMTR/MTR) IS STORED IN THE OUTPUT REGISTER (OR). FTN EXECUTES A MXN EXCHANGE FOR CPUMTR.

FTN WAITS FOR BYTE 0 OF THE OUTPUT REGISTER TO CLEAR (=0) BEFORE RETURNING CONTROL TO THE PP ROUTINE.

FTN ALSO PERFORMS PAUSE FOR STORAGE RELOCATION PROCESSING. WHEN A CP TO WHICH A PP IS ASSIGNED NEEDS TO BE STORAGE MOVED, THE PP MUST PAUSE TO ALLOW THE STORAGE MOVE TO TAKE PLACE. DIRECT CELLS RA AND FL WILL BE RESET AFTER THE PRLM MONITOR FUNCTION ISSUED BY PRL COMPLETES.

5. RCH REQUEST CHANNEL
   DCH DROP CHANNEL

MONITOR FUNCTIONS RCHM AND DCHM ARE ISSUED TO REQUEST/DROP CHANNELS OR PSEUDO CHANNELS.

THESE ROUTINES WILL BE REMOVED IN A FUTURE RELEASE. THE USAGE OF RCHAN AND DCHAN MACROS IS PREFERRED.

PPR
SUBROUTINES (Continued)

6. DFM DAYFILE MESSAGE

USED BY PP TO ISSUE DAYFILE MESSAGE. THE MESSAGE IS PASSED IN MESSAGE BUFFER FOR DFMM MONITOR FUNCTION.

THIS SUBROUTINE SHOULD NOT BE CALLED WITH CHANNELS RESERVED. (NON-DEDICATED)

7. EXR EXECUTE ROUTINE

EXR LOADS AN OVERLAY INTO THIS PP. EXR CALLS PLL TO IDENTIFY AND LOAD THE ROUTINE.

8. SMS SET MASS STORAGE

SMS LOADS MASS STORAGE DRIVER INTO THIS PP. PLL IS CALLED TO FIND AND LOAD THE DRIVER.

9. RDS READ DISK SECTOR
WDS WRITE DISK SECTOR
EMS END MASS STORAGE OPERATION

THESE "SUBROUTINES" ARE DEFINED TO BE AT THE SAME LOCATION FOR ALL MASS STORAGE DRIVERS. THIS ALLOWS PP CODE TO BE MASS STORAGE INDEPENDENT WITH RESPECT TO I/O OPERATIONS.

PPR COMPMAC MACROS

- EXECUTE PROGRAM/OVERLAY CALL

  EXECUTE NAME, ADDRESS

  NAME = NAME OF PPU ROUTINE

  ADDRESS = LOAD ADDRESS FOR ZERO LEVEL OVERLAY

  = * PUT NAME IN A BUT DO NOT EXECUTE

  = = DO NOT GENERATE CODE

  DOES A RETURN JUMP TO EXR.

- MONITOR REQUEST MONITOR FUNCTION

  MONITOR FUNC

  FUNC = FUNCTION TO PERFORM

  DOES A RETURN JUMP TO FTN

- PAUSE PAUSE FOR RELOCATION AND RESET (RA) AND (FL)

  DOES A MONITOR 0 WHICH IS SPECIAL CASED IN FTN

- DELAY DELAY FOR SYSTEM DELAY TIME. (APPROXIMATELY 130 MICROSECONDS)

PP ROUTINE RESIDENCE

- ALL PP ROUTINES RESIDE IN CENTRAL MEMORY AS PART OF THE RPL (RESIDENT PERIPHERAL LIBRARY) OR ON MASS STORAGE (AS PART OF PPULIB).

  THE PERIPHERAL LIBRARY DIRECTORY (PLD) CONTAINS THE RESIDENCE ADDRESS FOR THE PP ROUTINE.

  MONITOR FUNCTION SPLM (SEARCH PERIPHERAL LIBRARY) SEARCHES THE PLD AND RETURNS THE RESIDENCE ADDRESS FROM WHICH TO LOAD THE ROUTINE. THIS ADDRESS MAY BE FROM THE RPL, AN ASR (ALTERNATE SYSTEM RESIDENCE), OR MASS STORAGE.

- SYSTEM PERFORMANCE IS AFFECTED BY THE RESIDENCE OF FREQUENTLY USED ROUTINES.

  THE PPU PROGRAMS - IDD, SFP, ODF, TSE, ALL MASS STORAGE DRIVERS AND ERROR PROCESSORS 1MB (ON CYBER 170s) MUST RESIDE IN THE RPL.

  CONSULT THE SRB OR IHB FOR RECOMMENDED RPL RESIDENT ROUTINES.

PPR

1DD

1DD IS CALLED BY DFM TO DUMP A FULL DAYFILE BUFFER. A PORTION OF THE CURRENT PP ROUTINE IS WRITTEN TO THE DAYFILE DUMP BUFFER IN CMR WHILE 1DD IS LOADED INTO THAT AREA OF THE PPU FOR ITS EXECUTION.

1DD RE-LOADS THE CURRENT PROGRAM FROM THE DUMP BUFFER UPON COMPLETION.

IF THE DUMPING IS PROHIBITED BY AN UNRECOVERED WRITE ERROR, THE DIAGNOSTIC

"1DD ABT"

WILL BE DISPLAYED. THIS IS A WARNING - THE SYSTEM CONTINUES EXECUTION WITH FAULTY LINKAGE IN THE DAYFILE BEING DUMPED. OPERATOR SHOULD DFTERM THE AFFECTED DAYFILE.

1RP

1RP IS CALLED TO RESTORE PPR INTO A PP WHICH HAD DESTROYED IT OR USED ITS AREA.

1TD REQUESTS 1RP VIA RPPM. 1RP IS ASSIGNED A PPU (SAY PP4). 1RP WILL THEN PASS A COPY OF ITS PPR TO 1TD's PP IN 6 WORD BLOCKS THROUGH THE MESSAGE BUFFER. DIRECT CELLS ARE MODIFIED TO REFLECT "HARD" VALUES AND THE REQUESTING PP's COMMUNICATION AREA.

LOCATION FREE ROUTINES
(ZERO LEVEL OVERLAYS)

- BEGIN WITH ZERO
- RELOCATED FROM LOAD ADDRESS (LA)
  - COMPREL
  - COMPRLI
- LOAD VIA EXECUTE MACRO
- LENGTH CHECK VIA OVERFLOW MACRO AND COMSZOL

OAU	UPDATE PROFILA
OAV	VERIFY USER NUMBER
OBF	BEGIN FILE
ODF	DROP FILE
OBP	FORMAT BANNER PAGE
OFA	RELEASE FAST ATTACH FILE
OMF	INITIALIZE MMF LINK DEVICE
ORF	UPDATE RESOURCE FILES
ORP	RELEASE DIRECT PERMANENT FILE
OVJ	VERIFY JOB/USER STATEMENT
OCI	FIRMWARE IDENTIFIER
OTI	TRACK FLAW PROCESSOR
OPI	PACK SERIAL NUMBER PROCESSOR

QUESTION SET LESSON 9

1. Where must a PP routine be coded to run, if it is to interface with PPR (PP Resident)?

2. Why is it desirable to have a Pool PP pause for storage relocation?

3. What is a Pool PP doing when it is waiting for monitor to assign it to a job?

4. How does a Pool PP routine make monitor requests? How does it know if the request has been honored? How does the PP know when the request has been completed?

5. How does a Pool PP determine which control point it is assigned to?

6. How should a PP program write a message to the control point dayfile?

7. After pausing for storage relocation, where will a PP program find the updated RA and FL for the control point to which it is assigned? What programming considerations must the PP routine consider?

8. Since some PP routines will not fit in a PP, individual functions may have to be written as overlays. How can a PP program get one of these overlays loaded?

9. Why should a programmer be very careful about using locations 6776 - 7777 when dealing with mass storage input/output?

10. What is the difference between "PP HUNG" and "HUNG PP"?

11. Design, flowchart, and code a PP routine to list in the control point dayfile all rollout files names found in the FNT.

LESSON 10
MASS STORAGE CONCEPTS

LESSON PREVIEW:

In this lesson, the student will learn the concepts of mass storage in NOS, its allocation and the drivers to position, read and write it. The use of mass storage in this lesson is at the driver level and will be called "direct input/output".

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Discuss the mass storage allocation scheme employed in NOS.
- Describe in detail the Track Reservation Table (TRT) and how track chains are reserved.
- Describe the NOS sector format, complete with its header bytes.
- Describe the system conventions employed by the mass storage driver and the key PPR subroutines and entry points used to do.

  direct Input/output.
- Discuss 6DI - the Mass storage Driver.
- Discuss the error processing options available to the PP programmer when doing direct input/output.
- Discuss the use of Extended Core Storage (ECS) as both a mass storage device and user accessible field length.

REFERENCES:

NOS IMS - Chapter 7 NOS SMRM, 1-12-15

MASS STORAGE

CRM MANIPULATES A FILE VIA THE FIT - FILE INFORMATION TABLE

THE OPERATING SYSTEM INPUT/OUTPUT ACTIVITY IS CONTROLLED THRU THE

FET - FILE ENVIRONMENT TABLE

AND THRU THE

FNT/FST - FILE NAME TABLE
- FILE STATUS TABLE

THE PHYSICAL INPUT/OUTPUT ACTIVITY IS CONTROLLED BY THREE TABLES:

EST - EQUIPMENT STATUS TABLE
MST - MASS STORAGE TABLE
TRT - TRACK RESERVATION TABLE

FNT → CP AREA
FET → FST → EST → MST → TRT

RMS DEVICE COMPARISON

		INTERLACE	SECTORS/ TRACK	TRACKS/ DEVICE	WORDS/ DEVICE	MAXIMUM DATA RATE	MULTI-SPINDLE
844-21	DI	HALF TRACK	107	1632	11,175,936	46.1K WORDS/SEC.	1-8
7054 7152 7154	DK	FULL TRACK	112	1632	11,698,176	92.16K WORDS/SEC.	1--8
844-4X	DJ	HALF TRACK	227	1640	23,825,920	46.1K WORDS/SEC.	1-8
7054 7152 7154 7155	DL	FULL TRACK	227	1640	23,825,920	92.16K WORDS/SEC.	1-8
885	DM	HALF TRACK	640	1682	68,894,720	61.44K WORDS/SEC.	1-3
7155	DQ	FULL	640	1682	68,894,720	128.88K WORDS/SEC	1-3
ECS	DE		16			80K WORDS/SEC (1XPP)	NA
	DP					160K WORDS/SEC (2XPP)	

MASS STORAGE TABLE (MST)

59 51 47 40 35 23 17 11 5 0

<table>
<tr><td>000</td><td>†1</td><td>▨</td><td>TRT length</td><td colspan="2">†2</td><td colspan="2">no. avail. tracks</td><td>TDGL</td></tr>
<tr><td>001</td><td>†3</td><td>user ECS first track</td><td>file count</td><td colspan="2">IQFT track</td><td colspan="2">†4</td><td>ACGL</td></tr>
<tr><td>002</td><td colspan="2">ECS address of MST/TRT</td><td colspan="4">ECS MST/TRT update cnt</td><td>†5</td><td>SDGL</td></tr>
<tr><td>003</td><td>1st track IAF</td><td>label track</td><td>permits track</td><td colspan="2">no. catalog tracks</td><td colspan="2">DAT track</td><td>ALGL</td></tr>
<tr><td>004</td><td colspan="4">family or pack name</td><td>DN</td><td>▨</td><td>†6</td><td>PFGL</td></tr>
<tr><td>005</td><td colspan="4">user number for private pack</td><td colspan="3">†7</td><td>PUGL</td></tr>
<tr><td>006</td><td colspan="2">†8</td><td>driver name</td><td colspan="2">0</td><td colspan="2">sector limit</td><td>MDGL</td></tr>
<tr><td>007</td><td colspan="7">▨</td><td>R1GL</td></tr>
<tr><td>010</td><td colspan="7">installation area (global)</td><td>ISGL</td></tr>
<tr><td>011</td><td colspan="7">▨</td><td>I2GL</td></tr>
<tr><td>012</td><td>activity count</td><td>unit interlocks</td><td>current position</td><td colspan="2">MTR internal</td><td colspan="2">ECS error #</td><td>DALL</td></tr>
<tr><td>013</td><td colspan="7">†9</td><td>DILL</td></tr>
<tr><td>014</td><td>DAYFILE track</td><td>ACCOUNT track</td><td>ERRLOG track</td><td colspan="2">system table track</td><td colspan="2">†10</td><td>DULL</td></tr>
<tr><td>015</td><td colspan="3">†11</td><td colspan="2">user count</td><td colspan="2">†12</td><td>STLL</td></tr>
<tr><td>016</td><td colspan="7">†13</td><td>DDLL</td></tr>
<tr><td>017</td><td colspan="7">installation area</td><td>ISLL</td></tr>
</table>

MASS STORAGE LABEL FORMAT

DEVICE LABEL TRACK FORMAT

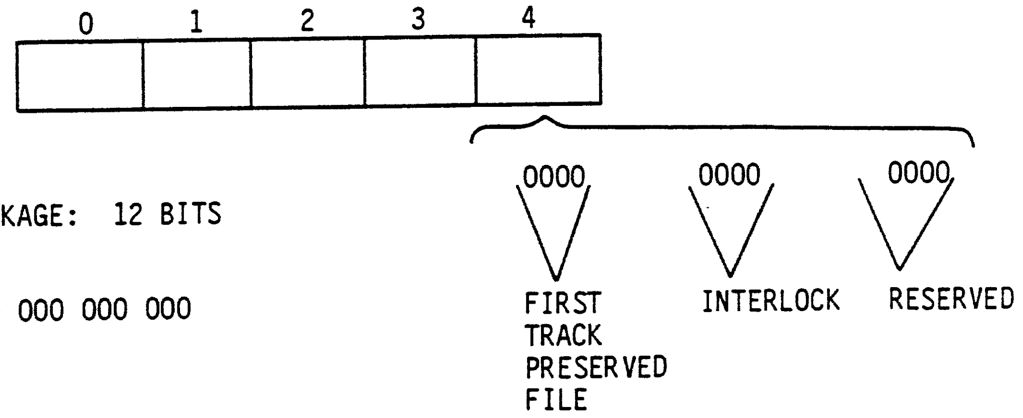
000	label sector
001 . . 012	track reservation table
013	sector of local information (2-word entries)
014	device information sector
015	intermachine communication area (ECS label track only)
016	MMF environment tables (ECS label track only)
017	CPUMTR storage move area for ECS (ECS label track only)

DEVICE LABEL SECTOR FORMAT

000 001 002	reserved		
003	label level	equipment type	reserved
004 005 006 007	reserved		
010 . . . 027	NOS MST		
030 . . . 077	unused		

MASS STORAGE

TRT - TRACK RESERVATION TABLE



LINKAGE: 12 BITS

000 000 000 000

↑ BIT 11=1 INDICATES LINKED TRACK
10-0 IS NEXT TRACK

BIT 11=1 INDICATES TERMINAL CONDITION
10-0 EOI SECTOR
10-1 = 3777 = FLAWED TRACK

WORD BYTE
0/00 000 000 0/00

TO DETERMINE TRT WORD AND BYTE FOR A TRACK:

- IGNORE BIT 11
- BYTE = BITS 0.1
- WORD = BITS 10-2

$$\text{EG:}\quad 4123 = \underbrace{\overset{4}{000}\ \overset{1}{000}\ \overset{2}{000}\ \overset{3}{0}}_{24\ =\ \text{WORD}}\underbrace{00}_{3\ =\ \text{BYTE}}$$

TRT BASE ADDRESS = MST ADDRESS + MSTL (20_8)

MASS STORAGE

	EQ	FIRST TRACK	CURRENT TRACK	CURRENT SECTOR		FST

- EQ POINTS TO EST ENTRY
  BYTE 4 OF EST ENTRY IS MST ADDRESS/10_8

0	1	2	3	
4	4012 5	6	7	0004
10	3777 11	4014 12	13	0006
4015 14	4016 15	4017 16	4020 17	0017
0007 20	21	22	23	0010

	EQ	4005	4020	0007		FST
		FT	CT	CS		

IF DL AT 343_8 SECTORS PER TRACK AND HAS THE ABOVE TRT

FILE IS __________ SECTORS LONG

WHERE IS SECTOR 350_8

CURRENT POSITION IS _________

MASS STORAGE

TRACK ALLOCATION/RESERVATION

RTCM - REQUEST TRACK CHAIN FOR A SPECIFIED NUMBER OF SECTORS ON A SPECIFIED EQUIPMENT OR FROM A COLLECTION OF DESIGNATED EQUIPMENTS (MSAL)
GENERATES OR ADDS TO TRACK CHAIN

DTKM - DROP TRACKS AND SET EOI SECTOR OR DROP ALL TRACKS. OPTIONALLY CAUSES DEVICE CHECKPOINT TO BE REQUESTED

DLKM - DELINK TRACKS - DROPS LINKAGE WITHIN TRACK CHAIN.

4015 → 4016 → 4017 → 4020 → 0007

4015 → 4020 → 0007

USED TO PURGE LARGE INDIRECT FILES FROM DATA CHAIN

STBM - SET TRACK BIT

- INTERLOCK SET/CLEAR
- PRESERVE SET/CLEAR
- FLAW SET/CLEAR

MASS STORAGE

TYPES OF SECTORS	HEADER BYTES LINKAGE CONTROL	NUMBER OF WORDS
SYSTEM SECTOR	3777	0077
FULL DATA SECTOR	NEXT TRACK OR SECTOR	0100
EOR (END OF RECORD)	NEXT TRACK OR SECTOR	$0 \leq WC \leq 0077$
EOF (END OF FILE)	0	NEXT TRACK OR SECTOR
EOI (END OF INFORMATION)	0	0

SYSTEM SECTOR HAS INFORMATION CONCERNING HOW THE FILE IS USED AND THAT INFORMATION AND ITS FORMAT VARIES ACCORDING TO USAGE.

THE END OF INFORMATION (EOI) SECTOR HAS INFORMATION THAT POINTS (LINKS) BACK TO THE SYSTEM SECTOR. THIS INFORMATION IS USED TO VALIDATE THE TRACK LINKAGE FOR THE FILE.

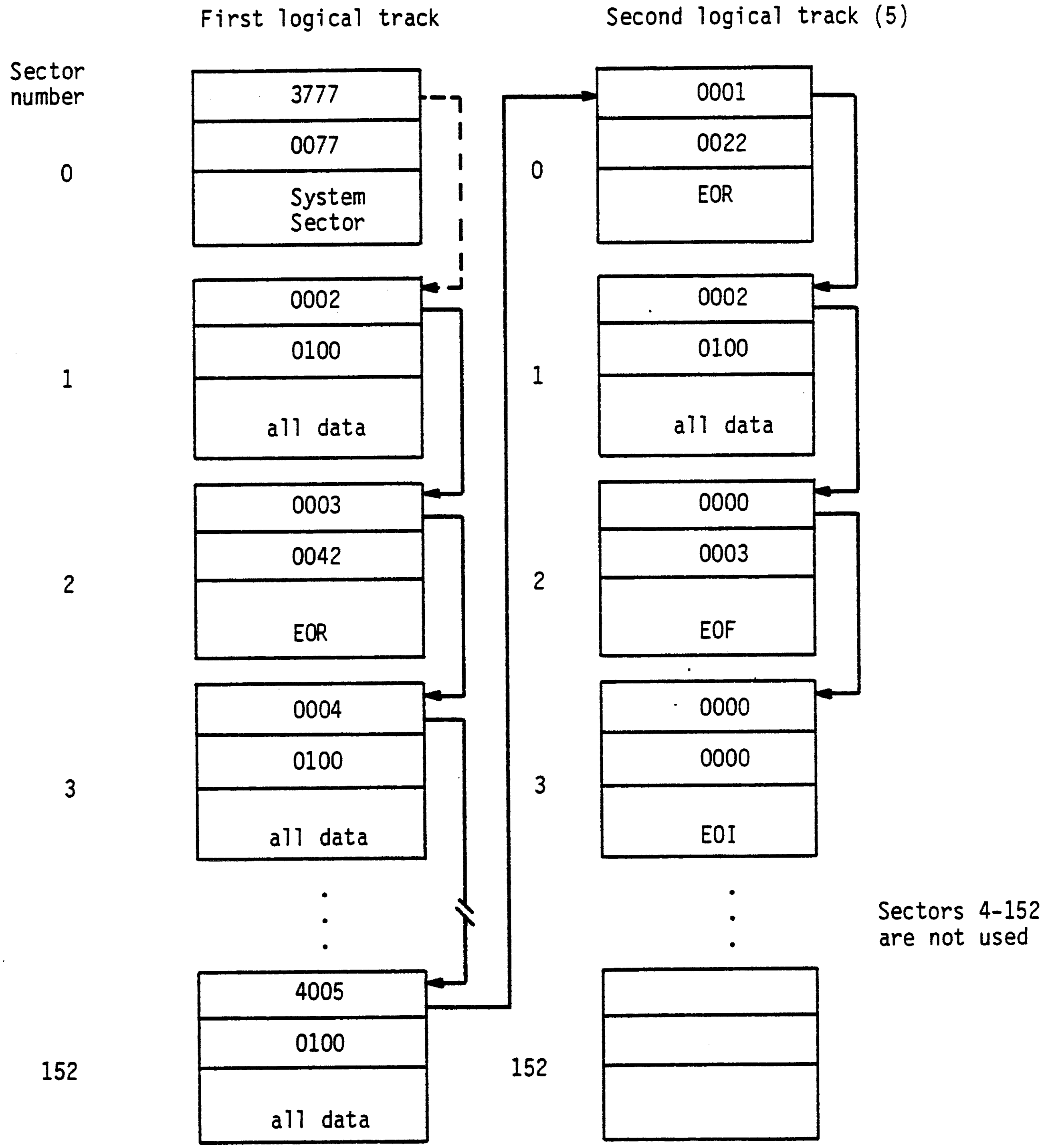
COMMON DECKS

COMSWEI

COMPWSS

COMPWEI

COMPRSS


First logical track
Second logical track (5)
Sector number
0
3777
0077
System Sector
1
0002
0100
all data
2
0003
0042
EOR
3
0004
0100
all data
152
4005
0100
all data
0
0001
0022
EOR
1
0002
0100
all data
2
0000
0003
EOF
3
0000
0000
EOI
152
Sectors 4-152 are not used

SETMS	-	SET MASS STORAGE DRIVE PARAMETERS
ENDMS	-	END MASS STORAGE DRIVER ACCESS

SETMS PERFORMS THE FOLLOWING OPERATIONS

- PASSES READ/WRITE OPERATION TO THE DRIVER
- INSURES CORRECT DRIVER IS LOADED
- PRESETS DRIVER
- INITIALIZES DRIVER RELATED CELLS (DRSW, RDCT, STSA, WDSE)
- SETS WRITE ERROR PROCESSING BUFFER ADDRESS
- SELECTS ERROR PROCESSING OPTIONS
- ELIMINATED POS (POSITION) DRIVER ENTRY POINT

SETMS IS CALLED

- BEFORE ANY INITIAL READ OR WRITE OPERATION
- WHEN SWITCHING BETWEEN READ AND WRITE OPERATIONS
- WHEN SWITCHING LOGICAL TRACKS IF LAST SECTOR WAS NOT THE LAST SECTOR OF A LOGICAL TRACK
- WHEN CHANGING THE ERROR PROCESSING OPTIONS

SETMS CALL FORMAT

```
SETMS       OP,(EP.....EP),WB

WHERE OP    =   READ        READ OPERATION
                READSYS     READ SYSTEM DEVICE
                WRITE       WRITE OPERATION

      EP    =   NE          NO ERROR PROCESSING
                SM          SUPPRESS ERRLOG MESSAGE
                RR          IMMEDIATE RETURN ON RESERVE
                NR          RETURN ON NOT READY

      WB    =   ADDRESS     ADDRESS OF WRITE ERROR
                            PROCESSING BUFFER
```

ENDMS TERMINATES MASS STORAGE OPERATIONS

- RELEASES CHANNEL
- RELEASES CONTROLLER
- RELEASES DRIVE
- RELEASES OTHER RESOURCES RESERVED FOR I/O
- ELIMINATED CHANNEL RELEASE TABLE FROM CMR

```
         LDD     FSTA
         CRD     FS
         LDD     FS                SET EQUIPMENT
         LPN     77
         STD     T5
         LDD     FS+1              SET FIRST TRACK
         STD     T6
         RJM     CRA               CONVERT RANDOM ADDRESS
         SETMS   READ



         LDC     BUF               READ SECTOR
         RJM     RDS
         MJN     ERR               IF ERROR DETECTED
         ENDMS
```

MASS STORAGE

BY SYSTEM CONVENTION

T4 = CHANNEL

T5 = EQUIPMENT (EST ORDINAL)

T6 = TRACK

T7 = SECTOR

MASS STORAGE

PPR DEFINITIONS:

DRSW - DRIVER SCRATCH WORD
WDSE - WRITE RECOVERY BUFFER ADDRESS
ERXA - RDS/WDS EXIT ADDRESS
RDCT - RETRY COUNTER
STSA - STATUS
UERP - ERROR PROCESSING OPTIONS
SLM - SECTOR LIMIT
MSD - DRIVER DESIGNATOR
CHRV - CHANNEL RESERVATION STATUS
SMSA - MMF FLAG
EMS - END MASS STORAGE
RDS - READ DISK SECTOR
WDS - WRITE DISK SECTOR

6DI - MASS STORAGE DRIVER

- PROCESSES EQUIPMENTS DI DJ DK DL DM DQ
- AUTOMATIC TRACK FLOWING USING HARDWARE INFORMATION RECORDED IN THE UTILITY SECTOR

881	(DI/DK)	CYLINDER	410
883	(DJ/DK)	CYLINDER	822
885	(DM/DQ)	CYLINDER	841

MANUFACTURING	FACTORY	UTILITY
0	1	2

- PACK SERIAL NUMBERS RECORDED IN ERRLOG FROM MANUFACTURING SECTOR
- CONTROLWARE REVISION NUMBER RECORDED IN ERRLOG
- SEEK OVERLAP (DSWM)
- LOGICAL TO PHYSICAL ADDRESS MAPPING (LDAM)

LDAM - LOGICAL TO PHYSICAL ADDRESS CONVERSION

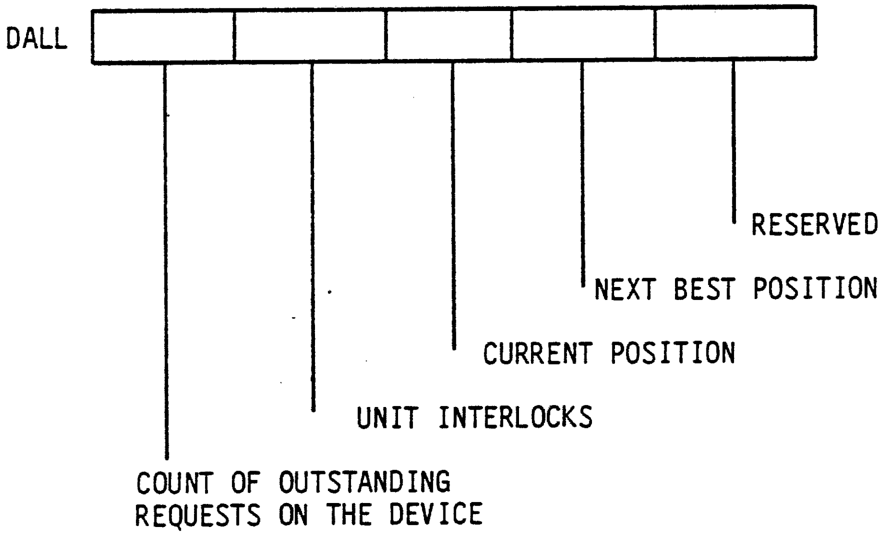
DRIVER USES LDAM MONITOR FUNCTION TO CONVERT LOGICAL TRACK AND SECTOR TO PHYSICAL UNIT, CYLINDER, HEAD, AND SECTOR.

MULTIPLE SPINDLE EQUIPMENTS

- GROUP UNITS TOGETHER
- NUMBER OF SECTORS PER TRACK IS N TIMES THE NUMBER OF SECTORS/TRACK FOR A SINGLE SPINDLE WHERE N IS THE NUMBER OF SPINDLES.
- FOR EXAMPLE, DJ - 3

  SECTORS/TRACK = $3 \times 227 = 671_{10} = 1251_8$

DSWM



ALL THE RESOURCES NEEDED TO DO AN I/O OPERATION ARE REQUESTED VIA DWSM. THE ORDER OF RESOURCE ASSIGNMENT IS:

- IF A READSYS REQUEST IS PENDING AND MORE THAN ONE SYSTEM DEVICE IS PRESENT, THE SYSTEM DEVICE IS SELECTED AS FOLLOWS:
  - DEVICE WITH THE REQUIRED UNIT FREE
  - IF MORE THAN ONE DEVICE HAS A UNIT FREE, THE ONE WITH THE LEAST ACTIVITY IS CHOSEN BASED ON THE ACTIVITY COUNT IN MST WORD DALL
  - IF MORE THAN ONE DEVICE IS FOUND AND ACTIVITY COUNTS ARE EQUAL, THE ONE IS CHOSEN WITH THE LEAST DISTANCE TO MOVE THE POSITIONER

DSWM

- SOFTWARE UNIT INTERLOCK IS OBTAINED IN DALL - THIS INTERLOCK INSURES ONLY ONE REQUEST IS PROCESSED FOR A UNIT AT A TIME. IN THE NON-MMF CONFIGURATION, THIS INTERLOCK IS NEEDED BECAUSE THE DRIVER RELEASES THE DRIVE AFTER INITIATING THE SEEK WHEN THE POSITIONER IS BUSY. THE OPERATION MAY BE CONTINUED ON A CHANNEL DIFFERENT FROM THE ONE ON WHICH THE SEEK WAS INITIATED. IN THE MMF ENVIRONMENT, THE HARDWARE DRIVE RESERVE IS MAINTAINED IN ADDITION TO THE DALL INTERLOCK WHILE SEEKING SO THAT REQUESTS FROM ANOTHER MACHINE DO NOT GAIN ACCESS TO THE DRIVE AND RESEEK.

- CHANNEL IS SELECTED AND RESERVED.

ERROR PROCESSING

PROCESSING OPTION SET IN UERP ALLOWS PROGRAM TO GAIN CONTROL OVER UNRECOVERED ERRORS.

SYMBOL	VALUE	DEFINITION
EPAR	23	RETURN ON ALL ERRORS
EPNE	10	DISABLE RETURN ON ALL ERRORS
EPSM	4	SUPPRESS ERRLOG MESSAGE
EPRR	2	IMMEDIATE RETURN ON RESERVES
EPNR	1	RETURN ON NOT-READY

- SET UP UERP USING SETMS

```
     SETMS      READ   (NE, SM, RR, NR)
                READSYS
```

- PERFORM READ OR WRITE OPERATION

```
     RJM        RDS              READ SECTOR
```

- TEST ERROR (A)   0 IF ERROR

```
     MJN        PDE              IF DISK ERROR
```

MASS STORAGE

ERROR PROCESSING

AN UNRECOVERED ERROR IS DECLARED WHEN THE OPERATION IS NOT SUCCESSFUL WITHIN A SPECIFIED NUMBER OF RETRIES.

NORMAL ERRORS

SYMBOL	MNEMONIC	RETRY	ERROR TYPE
PARE	PE	12	PARITY ERROR
ADDE	AD	12	ADDRESS ERROR
DSTE	ST	77	STATUS ERROR
FTON	FT	3	FUNCTION TIMEOUT

RESERVE ERRORS

RESE	RS	77	DEVICE RESERVED ERROR
CRSE	CR	77	CONTROLLER RESERVED ERROR

AUTO RETRY ERRORS

NRDE	NR	2	NOT READY ERROR

ERROR PROCESSING

DEPM (MTR FUNCTION)

- CHECKS VALIDITY OF DRIVER-SUPPLIED PARAMETERS
- FORMATS ERROR MESSAGE IN MESSAGE BUFFER
- REVERSES CHANNELS IN MTR TABLE (IF NEITHER CHANNEL IS 0)
- SETS UP 7EP PARAMETERS

DRIVER EXECUTES 7EP

- WRITES ERROR MESSAGE TO MSZW
- IF SPECIFIED, RETURNS IMMEDIATELY TO DRIVER
- TERMINATE MASS STORAGE PROCESSING (ENDMS)
- ISSUE DAYFILE AND ERRLOG MESSAGES AS APPROPRIATE
- ABORT JOB IF APPROPRIATE (UNRECOVERED AND NO USER ERROR PROCESSING SELECTED)
- RETURN TO DRIVER FOR RECOVERY ATTEMPT IF POSSIBLE
- DELAY FOR OPERATOR OVERRIDE IF ERROR IS "RETRY AFTER DELAY" TYPE
    - RETURN TO CALLER WITH UNRECOVERED STATUS OR ABORT JOB IF OVERRIDE ENTERED
    - AFTER DELAY, RETURN TO DRIVER TO RETRY OPERATOR IF OVERRIDE NOT ENTERED

DRIVER PERFORMS ERROR RECOVERY IF APPROPRIATE

ERROR PROCESSING

FULL TRACK WRITE ERROR PROCESSING

- ERROR STATUS MAY NOT BE AVAILABLE IMMEDIATELY
- LAST SECTOR WRITE FUNCTION
- PREVIOUS SECTOR ERROR CONDITIONS
- USER PROCESSING OF ERRORS MAY BE SELECTED BY THE USE OF THE SECOND PARAMETER OF THE SETMS MACRO
- A RECOVERY BUFFER MAY BE DEFINED FOR USE IN PREVIOUS SECTOR RECOVERY BY THE USE OF THE THIRD PARAMETER ON THE SETMS MACRO

	USER PROCESSING NOT SELECTED ON SETMS	USER PROCESSING SELECTED ON SETMS
RECOVERY BUFFER SPECIFIED ON SETMS MACRO	. PREVIOUS DATA READ INTO RECOVERY BUFFER . CURRENT SECTOR RETRIED	. READ PREVIOUS DATA INTO BUFFER AND RETRY RECOVERY . (A) = -0 IF PREVIOUS RETRY UNRECOVERED . CONTINUES IF PREVIOUS OK
RECOVERY BUFFER NOT SPECIFIED ON SETMS MACRO	ABORT RECOVERY	. READ PREVIOUS DATA OVER CURRENT AND RECOVER . (A) = -1 TO INDICATE CURRENT MUST BE REGENERATED

7X54/844-21 DISK STORAGE SUBSYSTEMS

Equipment type	DI (Half track, single density)
Sectors/track	107 x n
Tracks/device	1632
Words/device	11,175,936 x n
Maximum data rate	46.1K words per second

DI -- LOGICAL TO PHYSICAL ADDRESS CONVERSION

1 logical track = 107 physical sectors
= 1/2 of 9 physical tracks

4 logical tracks = 1 physical cylinder

LOGICAL

```
TRACK                      SECTOR
+-+--+-+-+                 +-+--------+
| |  tk| ht| hc|           | |    sc    |
+-+--+-+-+                 +-+--------+
11 10   1  0               11 10        0
```

PHYSICAL

```
UNIT           CYLINDER          TRACK            SECTOR
+-+---+        +--+----+         +---+---+        +---+--+
| | pu |       |  | pc |         |   | pt |       |   | ps |
+-+---+        +--+----+         +---+---+        +---+--+
11 5   0       11  8    0        11  4   0        11  4  0
```

Formulae

int(x)	integer portion of x
rem(x/y)	remainder of x divided by y
tk	logical track
sc	logical sector
pu	physical unit number (bits 5 through 0)
pc	physical cylinder number (bits 8 through 0)
pt	physical track number (bits 4 through 0)
lu	logical unit
ht	half track bit (bit 1 of logical track)
hc	half cylinder bit (bit 0 of logical track)
a	intermediate result

lu = int(sc/153B)

a = ht + 2 x rem(sc/153B)

ps = rem(a/30B)

pc = tk (bits 10 through 2)

pu = extracted from physical unit list in DDLL MST word

DI -- PHYSICAL TO LOGICAL ADDRESS CONVERSION

Physical cylinder = TRT ordinal

Physical head = 2 bytes in TRT word

0 <= physical head <= 10

if physical sector is even, UE byte
if physical sector is odd, UO byte

11 <= physical head <= 21

if physical sector is even, LE byte
if physical sector is odd, LO byte

```
                                 +----+----+----+----+----+
TRT ordinal                      | UE | LE | UO | LO |    |
                                 +----+----+----+----+----+
                                    EVEN sector ODD sector
```

DI -- SECTOR ALLOCATION
Logical track xxx0

PHYSICAL SECTOR

PHYSICAL HEAD | TRACK

	00	01	02	03	...	24	25	26	27
00	000		001		...	012		013	
01	014		015		...	026		027	
...	...		...		...	...		...	
07	124		125		...	136		137	
10	140		141		...	152		XXX	
...									
21									
22		THIS TRACK NOT USED BY NOS SOFTWARE							

DI -- SECTOR ALLOCATION
Logical track xxx1

PHYSICAL SECTOR

PHYSICAL HEAD | TRACK	00	01	02	03	...	24	25	26	27
00									
. . .									
10									
11	000		001		...	012		013	
12	014		015		...	026		027	
. . .	...		...		...	...		...	
20	124		125		...	136		137	
21	140		141		...	152		XXX	
22		THIS TRACK NOT USED BY NOS SOFTWARE							

DI -- SECTOR ALLOCATION
Logical track xxx2

PHYSICAL SECTOR

PHYSICAL HEAD TRACK	00	01	02	03	...	24	25	26	27
00		000		001		...	012		013
01		014		015		...	026		027
...		...		...		...	...		...
07		124		125		...	136		137
10		140		141		...	152		XXX
...									
21									
22		THIS TRACK NOT USED BY NOS SOFTWARE							

DI -- SECTOR ALLOCATION
Logical track xxx3

PHYSICAL SECTOR

PHYSICAL HEAD TRACK	00	01	02	03	...	24	25	26	27
00									
...									
10									
11		000		001		...	012		013
12		014		015		...	026		027
...		...		...		...	...		...
20		124		125		...	136		137
21		140		141		...	152		XXX
22			THIS TRACK NOT USED BY NOS SOFTWARE						

Equipment type	DJ (Half track, double density)
Sectors/track	227 x n
Tracks/device	1640
Words/device	23,825,920 x n
Maximum data rate	46.1K words per second

DI -- LOGICAL TO PHYSICAL ADDRESS CONVERSION

1 logical track = 227 physical sectors
                = 1/2 of 19 physical tracks

2 logical tracks = 1 physical cylinder

LOGICAL

```
TRACK                          SECTOR
+-+-+------+-+                 +-+-+----------+
| | |  tk  |ht|                | | |    sc    |
+-+-+------+-+                 +-+-+----------+
11 10      0                   11 10          0
```

PHYSICAL

```
UNIT                CYLINDER          TRACK                SECTOR
+--+-----+-----+-----+----+       +--+-----+-----+--+-----+----+
|  | pu  |     |     |    |  pc   |  |     | pt  |  |     | ps |
+--+-----+-----+-----+----+       +--+-----+-----+--+-----+----+
11 5     0    11     8            0  11    4     0  11    4    0
```

Formulae

int(x)	integer portion of x
rem(x/y)	remainder of x divided by y
tk	logical track
sc	logical sector
pu	physical unit number (bits 5 through 0)
pc	physical cylinder number (bits 9 through 0)
pt	physical track number (bits 4 through 0)
ps	physical sector number (bits 4 through 0)
lu	logical unit
ht	half track bit (bit 0 of logical sector)
a	intermediate result

lu = int(sc/343B)

a = ht + 2 x rem(sc/343B)

pt = int(a/30B)

ps =   rem(a/30B)

pc = tk (bits 10 through 1)

pu = extracted from physical unit list in DDLL MST word

DJ -- PHYSICAL TO LOGICAL ADDRESS CONVERSION

Physical cylinder = 2 x TRT ordinal

INT (physical cylinder / 2) = TRT ordinal

If physical cylinder is even = upper 2 bytes in TRT word

If physical cylinder is odd = lower 2 bytes in TRT word

If physical cylinder is even = upper of the two TRT bytes (E)

If physical cylinder is odd = lower of the two TRT bytes (O)

TRT ordinal

E	O	E	O	

EVEN cyl   ODD cyl

DJ -- SECTOR ALLOCATION
Even logical track

PHYSICAL SECTOR

PHYSICAL HEAD | TRACK

	00	01	02	03	...	24	25	26	27
00	000		001		...	012		013	
01	014		015		...	026		027	
02	030		031		...	042		043	
03	044		045		...	056		057	
⋮	...		...		...	...		...	
17	250		251		...	276		277	
20	300		301		...	312		313	
21	314		315		...	326		327	
22	330		331		...	342		XXX	

DJ -- SECTOR ALLOCATION
Odd logical track

PHYSICAL SECTOR

PHYSICAL HEAD | TRACK

	00	01	02	03	...	24	25	26	27
00		000		001		...	012		013
01		014		015		...	026		027
02		030		031		...	042		043
03		044		045		...	056		057
. . .		...		...		...	...		...
17		250		251		...	276		277
20		300		301		...	312		313
21		314		315		...	326		327
22		330		331		...	342		XXX

7154/844-21 DISK STORAGE SUBSYSTEMS

Equipment type	DK (Full track, single density)
Sectors/track	112 x n
Tracks/device	1632
Words/device	11,698,176 x n
Maximum data rate	92.16K words per second

DK -- LOGICAL TO PHYSICAL ADDRESS CONVERSION

```
1 logical track   =  112 physical sectors
                  =  4 3/4 physical tracks
                  =  4 physical tracks + 16 sectors

4 logical tracks  =  1 physical cylinder
```

LOGICAL

```
TRACK                         SECTOR
+-+-----+--+                  +-+---------+
| | tk  |rt|                  | |   sc    |
+-+-----+--+                  +-+---------+
11 10    0                    11 10       0
```

PHYSICAL

```
UNIT          CYLINDER        TRACK          SECTOR
+--+--+       +--|----+       +--+--+        +--+--+
|  |pu|       |  | pc |       |  |pt|        |  |ps|
+--+--+       +--+----+       +--+--+        +--+--+
11 5  0       11 8    0       11 4  0        11 4  0
```

Formulae

int(x)	integer portion of x
rem(x/y)	remainder of x divided by y
tk	logical track
sc	logical sector
lu	logical unit
pu	physical unit number (bits 5 through 0)
pc	physical cylinder number (bits 8 through 0)
pt	physical track number (bits 4 through 0)
ps	physical sector number (bits 4 through 0)
rt	relative track in physical cylinder (bits 1, 0 of logical track)

lu = int(sc/160B)

ps = rem((rt x 162B + rem(sc/160B))/30B)

pt = int((rt x 162B + rem(sc/160B))/30B)

pc = tk (bits 10 through 2)

pu = extracted from physical unit list in DDLL MST word

DK -- PHYSICAL TO LOGICAL ADDRESS CONVERSION

Physical cylinder = TRT ordinal

```
Physical head
    00 <= head <= 03 = upper byte of TRT word

    head = 04
           00 <= physical sector <= 21 = upper byte
           22 <= physical sector <= 27 = second byte

    05 <= head <= 10 = second byte

    head = 11
           00 <= physical sector <= 13 = second byte
           14 <= physical sector <= 27 = third byte

    12 <= head <= 15 = third byte

    head = 16                        L
           00 <= physical sector <= 05 = third byte
           06 <= physical sector <= 27 = fourth byte

    17 <= head <= 22 = fourth byte
```

TRT ordinal

1/4	1/4	1/4	1/4	

DK -- SECTOR ALLOCATION
Logical track xxx0

PHYSICAL SECTOR

PHYSICAL HEAD | TRACK	00	01	...	17	20	21	...	26	27
00	000	001	...	017	020	021	...	026	027
01	030	031	...	047	050	051	...	056	057
02	060	061	...	077	100	101	...	056	057
03	110	111	...	127	130	131	...	136	137
04	140	141	...	157	* UNUSED *				
05									
. . .									
22									

DK -- SECTOR ALLOCATION
Logical track xxx1

PHYSICAL SECTOR

PHYSICAL HEAD — TRACK

	00	...	10	11	12	...	22	...	27
00									
...									
04							000	...	005
05	006	...	017	020	021	...	030	...	035
06	036	...	047	050	051	...	060	...	065
07	066	...	077	100	101	...	110	...	115
10	116	...	127	130	131	...	140	...	145
11	146	...	157	* UNUSED *					
...									
22									

DK -- SECTOR ALLOCATION
Logical track xxx2

PHYSICAL SECTOR

PHYSICAL HEAD | TRACK

	00	...	03	04	05	...	14	...	27
00									
. . .									
11							000	...	013
12	014	...	017	020	021	...	030	...	043
13	044	...	047	050	051	...	060	...	073
14	074	...	077	100	101	...	110	...	123
15	124	...	127	130	131	...	140	...	153
16	154	...	157	* UNUSED *					
. . .									
22									

DK -- SECTOR ALLOCATION
Logical track xxx3

PHYSICAL SECTOR

PHYSICAL HEAD TRACK

	00	01	...	06	...	24	25	26	27
00									
. . .									
15									
16				000	...	016	017	020	021
17	022	023	...	030	...	046	047	050	051
20	052	053	...	060	...	076	077	100	101
21	102	103	...	110	...	126	127	130	131
22	132	133	...	140	...	156	157	* UNUSED *	

7715X/844-4X DISK STORAGE SUBSYSTEM

Equipment type	DL (Full track, double density)
Sectors/track	227 x n
Tracks/device	1640
Words/device	23,825,920 x n
Maximum data rate	92.16K words per second

DL -- LOGICAL TO PHYSICAL ADDRESS CONVERSION

```
1 logical track    =  227 physical sectors
                   =  9 1/2 physical tracks
                   =  9 physical tracks + 11 sectors

2 logical tracks   =  1 physical cylinder
```

LOGICAL

```
TRACK                          SECTOR
+-+------+--+                  +-+--------+
| |  tk  |rt|                  | |   sc   |
+-+------+--+                  +-+--------+
11 10       0                  11 10      0
```

PHYSICAL

```
UNIT            CYLINDER          TRACK            SECTOR
+--+---+        +---+----+        +---+---+        +---+---+
|  |pu |        |   | pc |        |   |pt |        |   |ps |
+--+---+        +---+----+        +---+---+        +---+---+
11 5   0        11  8    0        11  4   0        11  4   0
```

Formulae

int(x)	integer portion of x
rem(x/y)	remainder of x divided by y
tk	logical track
sc	logical sector
lu	logical unit
pu	physical unit number (bits 5 through 0)
pc	physical cylinder number (bits 9 through 0)
pt	physical track number (bits 4 through 0)
ps	physical sector number (bits 4 through 0)
rt	relative track in physical cylinder (bit 0 of logical track)

lu = int(sc/343B)

ps = rem((rt x 345B + rem(sc/343B))/30B)

pt = int((rt x 345B + rem(sc/343B))/30B)

pc = tk (bits 10 through 1)

pu = extracted from physical unit list in DDLL MST word

DL -- PHYSICAL TO LOGICAL ADDRESS CONVERSION

Physical cylinder = TRT ordinal x 2

INT (physical cylinder / 2) = TRT ordinal

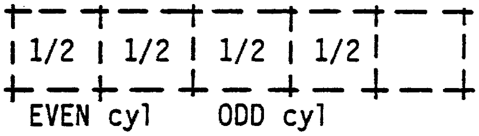
If physical cylinder is even = upper 2 bytes

If physical cylinder is odd = lower 2 bytes

00 <= physical head <= 10 = upper of the two bytes
12 <= physical head <= 22 = lower of the two bytes

head = 11
00 <= physical sector <= 13 = upper of the two bytes
14 <= physical sector <= 27 = lower of the two bytes

TRT ordinal



DL -- SECTOR ALLOCATION
Even logical track

PHYSICAL SECTOR

PHYSICAL HEAD | TRACK

	00	01	...	12	13	14	...	26	27
00	000	001	...	012	013	014	...	026	027
01	030	031	...	042	043	044	...	056	057
02	060	061	...	072	073	074	...	106	107
. . .	...	...	...	...	...	...	...	...	...
10	300	301	...	312	313	314	...	326	327
11	330	331	...	342	* UNUSED *				
12									
. . .									
22									

DL -- SECTOR ALLOCATION
Odd logical track

PHYSICAL SECTOR

PHYSICAL HEAD | TRACK

	00	01	...	13	14	15	...	26	27
00									
...									
10									
11				* UNUSED *		000	...	011	012
12	013	014	...	026	027	030	...	041	042
12	043	044	...	056	057	060	...	071	072
...									
21	263	264	...	276	277	300	...	311	312
22	313	314	...	326	327	330	...	341	342

7155/555 DISK STORAGE SUBSYSTEMS

Equipment type	DM (Half track)
Sectors/track	640 x n (1 = n = 3)
Tracks/device	1682
Words/device	68,894,720 x n
Maximum data rate	61.44K words per sector

DM -- LOGICAL TO PHYSICAL ADDRESS CONVERSION

1 logical track = 640 physical sectors
= 1/2 sectors of 40 physical tracks

2 logical tracks = 1 physical cylinder

LOGICAL

```
TRACK                              SECTOR
+-+-- --+--+                       +-+----------+
| | tk  | ht |                     | |    sc    |
+-+-- --+--+                       +-+----------+
11 10   1  0                       11 10        0
```

PHYSICAL

```
UNIT          CYLINDER         TRACK          SECTOR
+--+--+       +--+---+         +--+--+        +--+--+
|  |pu|       |  | pc|         |  |pt|        |  |ps|
+--+--+       +--+---+         +--+--+        +--+--+
11 5  0       11 8   0         11 4  0        11 4  0
```

Formulae

int(x)	integer portion of x
rem(x/y)	remainder of x divided by y
tk	logical track
sc	logical sector
pu	physical unit number (bits 5 through 0)
pc	physical cylinder number (bits 9 through 0)
pt	physical track number (bits 4 through 0)
ps	physical sector number (bits 4 through 0)
ht	half track bit (bit 0 of logical track)

lu = int(sc/1200B)

pt = int(sc/20B)

ps = ht + rem(sc/20B)

pc = tk (bits 10 through 1)

pu = extracted from physical unit list in DDLL MST word

DM -- PHYSICAL TO LOGICAL ADDRESS CONVERSION

Physical cylinder = 2 x TRT ordinal

INT (physical cylinder / 2) = TRT ordinal

If physical cylinder is even = upper two bytes

If physical cylinder is odd = lower two bytes

If physical sector is even = upper of the two bytes

If physical sector is odd = lower of the two bytes

TRT ordinal

EE	EO	OE	OO	

EVEN cyl   ODD cyl

DM -- SECTOR ALLOCATION
Even logical track

	PHYSICAL SECTOR								
								SPARES*	
PHYSICAL HEAD | TRACK	00	01	02	03	...	36	37	40	41
00	0000		0001		...	0017			
01	0020		0021		...	0037			
02	0040		0041		...	0057			
. . .	...		...		...	...			
22	0540		0541		...	0557			
23	0560		0561		...	0577			
24	0600		0601		...	0617			
. . .	...		...		...	...			
47	1160		1161		...	1177			

* Spare sectors are used automatically when sector flaws exist.

DM -- SECTOR ALLOCATION
Odd logical track

PHYSICAL SECTOR

PHYSICAL HEAD | TRACK								SPARES*	
	00	01	02	03	...	36	37	40	41
00		0000		0001	...		0017		
01		0020		0021	...		0037		
02		0040		0041	...		0057		
...		...		...	...		...		
22		0540		0541	...		0557		
23		0560		0561	...		0577		
24		0600		0601	...		0617		
...		...		...	...		...		
47		1160		1161	...		1177		

* Spare sectors are used automatically when sector flaws exist.

7155/845 DISK STORAGE SUBSYSTEMS

Equipment type	DQ (Full track)
Sectors/track	640 x n (1 < = n < = 3)
Tracks/device	1682
Words/device	68,894,720 x n
Maximum data rate	122.88K words per second

DQ -- LOGICAL TO PHYSICAL ADDRESS CONVERSION

1 logical track = 640 physical sectors
= 20 physical tracks

2 logical tracks = 1 physical cylinder

LOGICAL

```
TRACK                         SECTOR
+-+------+---+                +--+-----------+
| |  tk  | rt|                |  |     sc    |
+-+------+---+                +--|-----------+
11 10    1  0                 11 10          0
```

PHYSICAL

```
UNIT           CYLINDER          TRACK            SECTOR
+--+---+       +----+----+       +----+---+       +----+---+
|  |pu |       |    | pc |       |    | pt|       |    | ps|
+--+---+       +----+----+       +----+---+       +----+---+
11 5   0       11   8    0       11   4   0       11   4   0
```

Formulae

int(x)	integer portion of x
rem(x/y)	remainder of x divided by y
tk	logical track
sc	logical sector
pu	physical unit number (bits 5 through 0)
pc	physical cylinder number (bits 9 through 0)
pt	physical track number (bits 4 through 0)
ps	physical sector number (bits 4 through 0)
ht	half track bit (bit 0 of logical track)
rt	relative track in physical cylinder (bit 0 of logical track)

lu = int (sc/1200B)

pt. = rt x 24B + int(sc/40B)

ps = ht + rem(sc/40B)

pc = tk (bits 10 through 1)

pu = extracted from physical unit list in DDLL MST word

DQ -- PHYSICAL TO LOGICAL ADDRESS CONVERSION

Physical cylinder = 2 x TRT ordinal

INT (physical cylinder / 2) = TRT ordinal

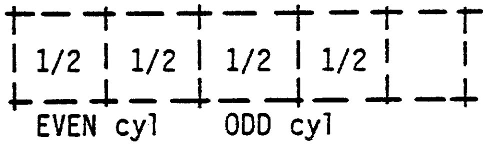
If physical cylinder is even = upper two bytes

If physical cylinder is odd = lower two bytes

00 < = physical head < = 23 = upper of the two bytes

24 < = physical head < = 47 = lower of the two bytes

TRT ordinal



DQ -- SECTOR ALLOCATION
Even logical track

PHYSICAL SECTOR

Odd logical track

SPARES*

PHYSICAL HEAD — TRACK

	00	01	02	03	...	36	37	40	41
00	0000	0001	0002	0003	...	0036	0037		
01	0040	0041	0042	0043	...	0076	0077		
02	0100	0101	0102	0103	...	0136	0137		
...	...	...	...	...	...	...	...		
22	1100	1101	1101	1103	...	1136	1137		
23	1140	1141	1142	1143	...	1176	1177		
24									
...									
47									

* Spare sectors are used automatically when sector flaws exist.

DQ -- SECTOR ALLOCATION
Odd logical track

PHYSICAL SECTOR

								SPARES*	
PHYSICAL HEAD | TRACK	00	01	02	03	...	36	37	40	41
00									
⋮									
23									
24	0000	0001	0002	0003	...	0036	0037		
25	0040	0041	0042	0043	...	0076	0077		
26	0100	0101	0102	0103	...	0136	0137		
⋮	...	...	...	...	...	...	...		
46	1100	1101	1102	1103	...	1136	1137		
47	1140	1141	1142	1143	...	1176	1177		

* Spare sectors are used automatically when sector flaws exist.

ECS

- MASS STORAGE DEVICE
- ALLOCATED BY TRACKS AND SECTORS
- 16_{10} SECTORS/TRACK
- 65_{10} WORD SECTORS (LINKAGE + DATA)
- LINKAGE WORDS GROUPED TOGETHER SO THAT ONE TRACK OF DATA IS CONTIGUOUS WORDS

  LINKAGE WORD ADDRESS = $T \times 2020_8 = S$

  DATA ADDRESS = $(T \times 2020_8) + 20_8 + (S \times 100_8)$

  T = LOGICAL TRACK
  S = LOGICAL SECTOR

- USED FOR ALL MASS STORAGE PROPERTIES
- WHEN USED AS LINK DEVICE IN MMF COMPLEX, LABEL TRACK HAS MMF TABLES

ECS

- USER PROGRAM FIELD LENGTH

    FTN/COBOLS/SYMPL/COMPASS

- DIRECT VS. INTERPRETIVE MODES

    RE WE

  COMCECM - ECSTEXT

  COMCECS

    RD WT

- UEC = CMRDECK ENTRY
- CPU 0
- FIELD LENGTH MANAGEMENT ANALOGOUS TO CENTRAL MEMORY

ECS Direct Access Chain

	59	47	35	23	17	11	5	0
000	** UECS.				▨	LIFT	▨	
001	eqss	ftss	▨					
002	▨		dtss					
003	▨							
004	mid1	ft1	ln1	ra1		lt1		
005	mid2	ft2	ln2	ra2		lt2		
006	mid3	ft3	ln3	ra3		lt3		
007	mid4	ft4	ln4	ra4		lt4		

eqss	Equipment number.
ftss	First track.
dtss	Last modification date and time (packed format).
mid	Machine ID.
ft	First track of subchain.
ln	Length of ECS block.
ra	RAE of ECS block.
lt	Last track of subchain.

ECS

ACCESS UNDER CONTROL OF CPUMTR BY FUNCTION ECSM

- RRES/WRES

  TRANSFER 1 TO 100_8 WORDS TO/FROM RELATIVE ECS TO A SPECIFIED CM BUFFER.

- RECS/WECS

  TRANSFER 1 TO 100_8 WORDS TO/FROM ABSOLUTE ECS TO A SPECIFIED CM BUFFER.

- SFRS/CFRS

  SET/CLEAR FLAG REGISTER BITS (AND SET/CLEAR INDICATION OF WHICH MAINFRAME HELD THE INTERLOCK (FLAG REGISTER BIT))

- RELS

  READ ECS ACCORDING TO A LIST OF ABSOLUTE ADDRESSES. THIS IS USED TO READ DATA FOR ECS DISPLAYS.

ECS

ACCESS UNDER CONTROL OF CPUMTR BY FUNCTION PIOM

- REBS

  REQUEST BUFFER INTERLOCK ON CENTRAL MEMORY ECS PPU BUFFER

- RESS/WESS

  TRANSFER 101_8 WORDS TO/FROM CENTRAL MEMORY ECS PPU BUFFER. THIS WORKS WITH DATA AS IF IT WERE A MASS STORAGE SECTOR: LINKAGE + DATA

ACCESS UNDER CONTROL OF MTR BY FUNCTION ECXM

ECXM IS USED TO READ/WRITE BLOCK OF ECS DATA TO CENTRAL MEMORY. THIS FUNCTION IS USED BY IRI/IRO TO ROLL JOB WITH ECS FIELD LENGTH.

QUESTION SET LESSON 10

1. Explain the purpose and usage of the SETMS and ENDMS macros.
2. How does a PP program request mass storage space?
3. What subroutine within a mass storage driver would you use to read a sector? Write a sector?
4. Why is the last operation when writing on mass storage a DTKM (Drop Track) monitor function?
5. For the file at FNT address 6362 in the study dump, how long is this file (in sectors)? Trace the track linkage.
6. Find the first two flaws on mass storage equipments 6 and 14. Give your answer by logical track.
7. Explain the concept of seek overlap.
8. Explain the recovery techniques used when writing in 1 to 1 interlace (full tracking) mode.
9. Mass storage equipment 14 is a multi-spindle device. How many sectors per track and what are the physical units for equipment.

   14? This equipment would have what mnemonic?

LESSON 11
DEADSTART AND RECOVERY

LESSON PREVIEW:

In this lesson, the processing of deadstarting the NOS Operating System is presented. The particulars of the deadstart process as far as commands and panel setting are detailed in the NOS Installation and Maintenance course.

This lesson also covers the various levels of deadstart/recovery and overviews the operations performed to recover mass storage devices.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Describe how the NOS system uses the hardware deadstart properties to establish the operating system.
- Describe each logical step in the deadstart process, detailing what is done at each step and what routines are involved.
- Explain each type of recovery and tell what is recovered by that level of deadstart.
- Explain why a deadstart should not be done with device checkpoints pending.
- Identify the routines involved with mass storage recovery both at deadstart and on-line.

REFERENCES:

NOS IMS - Chapters 8 and 26 NOS Operator's Guide, 2 NOS IHB, 7, 8 NOS SMRM, 6 NOS Installation and Maintenance course materials

SUPPLEMENTAL TEXT: DEVICE CHECKPOINT

A "device checkpoint" consists of preserving the current CMR tables for a mass storage device (MST/TRT/MRT) in the label track of that device.

Anytime the system alters the mass storage allocation information of a mass storage device, a checkpoint will be requested on that device. This status is set in the device's MST. Periodically, 1SP checks to see if any device checkpoints are pending and if so will call 1CK to checkpoint the mass storage devices.

If "MS VALIDATION" is enabled, 1CK will validate the device and, if the validation is successful, will write all mass storage tables into the label for the device. If the validation is not successful, the operator will be informed by messages at the system control point and the checkpoint will be aborted. This keeps incorrect data from being checkpointed to the device label.

HARDWARE DEADSTART

1. EACH CHANNEL IS CONNECTED TO ITS CORRESPONDING PPU (I.E., CH 1 TO PPU 1, CH 2 TO PPU 2, ETC.)

2. MASTER CLEAR IS DONE ON I/O CHANNELS WHICH SETS EACH CHANNEL ACTIVE AND EMPTY, READY TO ACCEPT INPUTS.

3. (A) OF EACH PPU IS SET TO 10000 SO THAT A PPU CAN INPUT ITS ENTIRE FL BEFORE AUTOMATIC DISCONNECT. P REGISTER CLEARED (SET = 0).

4. DEADSTART PANEL IS SENT ACROSS CHANNEL 0 INTO PPU 0 AT LOCATIONS 1-20 (1-14 NON-170).

   A DCN IS ISSUED AND PPU 0 BEGINS EXECUTION AT (P) + 1 = 0+1 = 1.

5. EACH PPU SIMULATES AN IAM ON ITS CHANNEL.

```
     I.E.,       LDC 10000
                 IAM CH,0000
```

6. CPU DOES A HARDWARE IDLE.

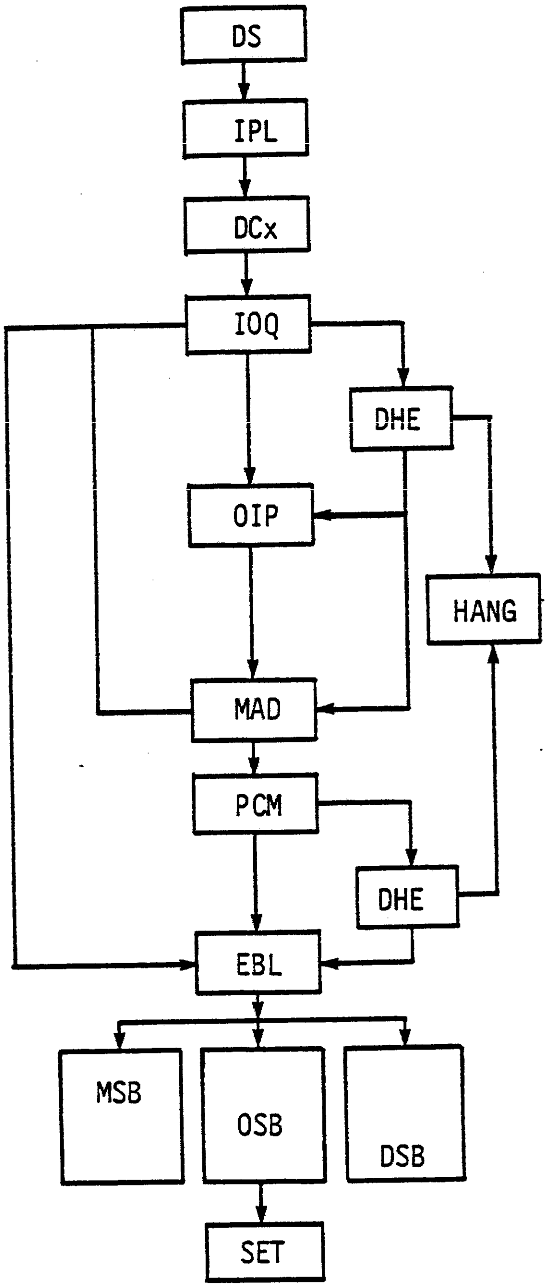
WHEN IAM BEGINS, (P) IS STORED IN LOCATION 0. AS EACH 12 BIT PP WORD IS RECEIVED, (A) IS DECREMENTED. WHEN (A)=0 OR CHANNEL IS DISCONNECTED, THE PPU STORES (0)+1 INTO P AND BEGINS EXECUTION.

THE PROGRAM INPUTTED BY THE PPU IS SUCH THAT THE LAST WORD READ WRAPS INTO LOCATION 0. THIS LAST WORD IS THE ADDRESS TO START EXECUTION.

DEADSTART/CTI

CTI - COMMON TEST AND INITIALIZATION

- ONE PANEL SETTING FOR NOS, NOS/BE, CMSE OVER ALL MAINFRAMES.
- OPERATOR DISPLAYS ALLOW SELECTION OF WHAT TO RUN, CONFIGURATION MODIFICATION, INSTALLATION OF MAINTENANCE MODULES.
- PASSES TO OPERATING SYSTEM OR DIAGNOSTICS A HARDWARE DESCRIPTOR TABLE (HDT) DESCRIBING THE PROPERTIES OF THE MAINFRAME CONFIGURATION, CONTENTS OF DEADSTART PANEL, DISK LOCATION POINTERS FOR CTI, CMSE, DDS, OS IN PPO.
- LOADS/BOOTSTRAPS TO CMSE, DIAGNOSTIC SEQUENCER (DDS) AND OPERATION SYSTEM (SET).



DEADSTART

INITIAL PROGRAM LOAD

COMMON DEADSTART DRIVER

INITIAL OPERATOR QUERY

DISPLAY HARDWARE ERROR

OPERATOR INTERVENTION

IF UNRECOVERABLE ERROR

MACHINE ATTRIBUTE DETERMINATION

PRESET COMPUTER MEMORY

DISPLAY HARDWARE ERROR

ENTER BOOTSTRAP LOADER

CMSE BOOTSTRAP
OPERATING SYSTEM BOOTSTRAP
DIAGNOSTIC SEQUENCER BOOTSTRAP

LOAD FIRST OS MODULE - SET

CTI MODULE DESCRIPTION

IPL - INITIAL PROGRAM LOAD

- CONTAINS POINTER TABLE.
- DETERMINES DS DEVICE TYPE AND LOADS APPROPRIATE DRIVER.

  CD7 - 67X
  CD6 - 66X
  CD3 - 60X/65X
  CD4 - 844
  CD8 - 885

CDX - COMMON DEVICE DRIVER

- LOADS AND TRANSFERS CONTROL TO IOQ.
- CALLED TO LOAD EACH CTI MODULE.

IOQ - INITIAL OPERATOR QUERY

- PRESENTS INITIAL OPTIONS DISPLAY.
- INITIALIZES HARDWARE DESCRIPTOR TABLE (HDT).
- SAVES COPY OF DEADSTART PANEL.

OIP - OPERATOR INTERVENTION DISPLAY

- ALLOWS SELECTION OF DDS SELECTION.
- ALLOWS MODIFICATION OF D.S. PANEL (WORDS 12-14) (MODIFIES DEADSTART PANEL IMAGE).
- ALLOWS SELECTION/DESELECTION OF MAINFRAME ATTRIBUTES (MODIFIES HDT).

MAD - MAINFRAME ATTRIBUTE DETERMINATION

- DETERMINES ACTUAL HARDWARE CONFIGURATION AND SETS APPROPRIATE BITS IN HDT (HAREWARE RECONFIGURATION VIA H DISPLAY OVERRIDES ACTUAL CONFIGURATION).
- DEADSTARTS FIRST LEVEL PPUs.

PCM - PRESET COMPUTER MEMORY

- INITIALIZE PP MEMORIES (0'S THEN 1'S) CHECKING FOR ERRORS.
- INITIALIZES CM MEMORIES (0'S THEN 1'S) CHECKING FOR ERRORS (IF RECOVERY NOT SELECTED).
- IDLES DESELECTED PP'S.

EBL - EXTERNAL BOOTSTRAP LOADER

- ESTABLISHES CTI HANDOFF STATE.
- LOADS SELECTED BOOTSTRAP (OSB, MSB, DSB).

DHE - DISPLAY HARDWARE ERRORS

- CALLED BY IOQ OR PCM IF ERROR STATUS IS DETECTED IN SCR REGISTER.
- CLEARS NON-FATAL ERRORS (SINGLE BIT SECDED) AND RETURNS TO CALLING PROGRAM
- DISPLAYS FATAL ERROR STATUS AND HANGS.

AEI - UTILITY DISPLAY

- DISPLAYS CTI UTILITY OPTIONS.

SAD - SELECT ALTERNATE DEVICE

- ALLOWS DEADSTART FROM DEVICE OTHER THAN ORIGINAL DEADSTART DEVICE.

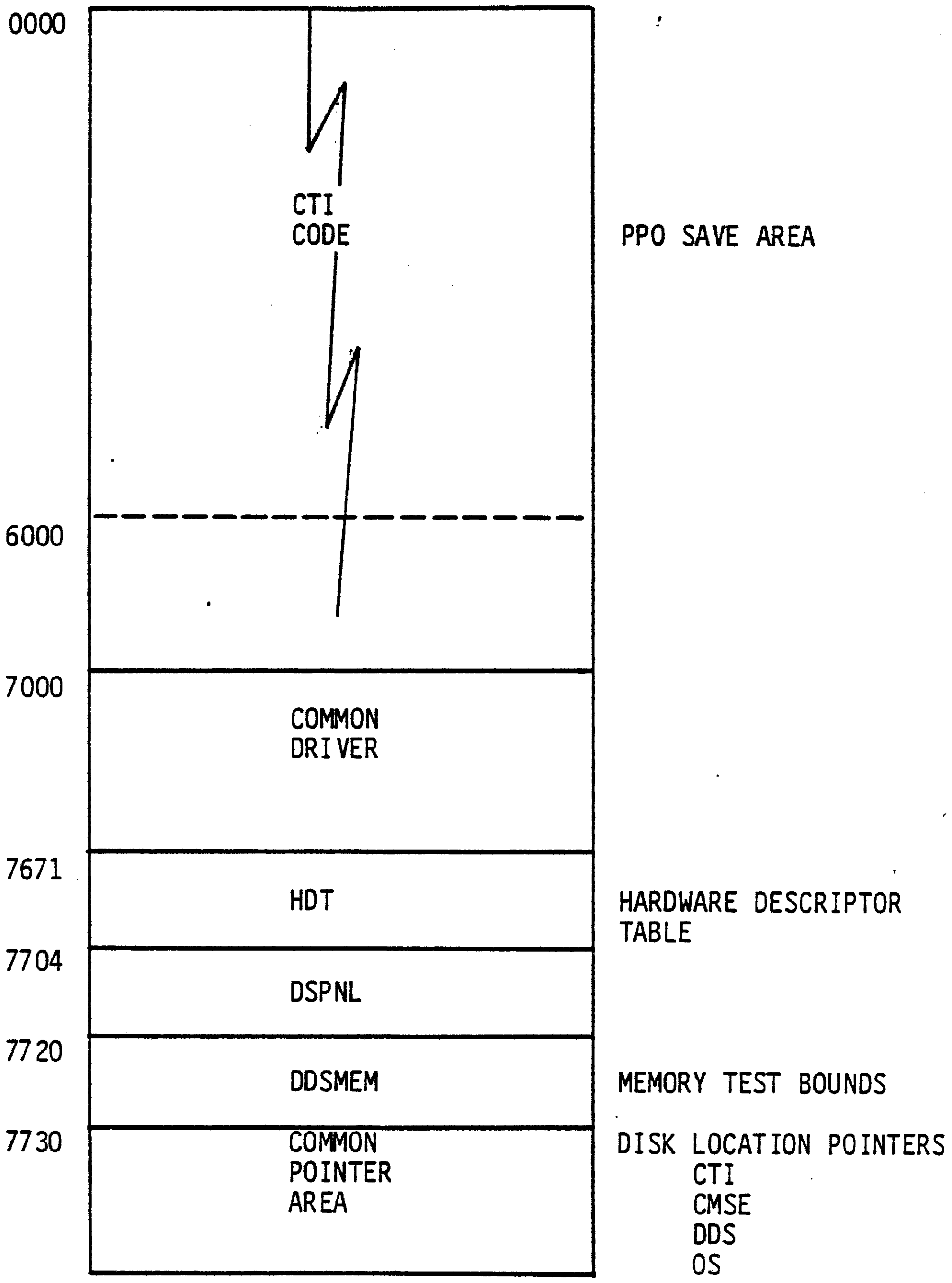
ICD - INSTALL CTI ON RMS DEVICE

- INSTALLS CTI ON SELECTED DEVICE.

EDD - EXPRESS DEADSTART DUMP

- ALLOWS SYSTEM DUMP TO TAPE.

CTI MEMORY MAP



HARDWARE DESCRIPTOR TABLE

WORD 1 WORD 2	CENTRAL MEMORY SIZE	MULTIPLES OF 100B
WORD 3	CPU OPTIONS*	1 BIT FOR EACH OPTION
WORD 4 WORD 5	PHYSICAL PPs PRESENT	1 BIT FOR EACH PP
WORD 6 WORD 7	LOGICALLY ON PPS	1 BIT FOR EACH PP
WORD 8 WORD 9	PHYSICAL PPUs PRESENT	1 BIT FOR EACH PPU
WORD 10 WORD 11	LOGICALLY ON PPs	1 BIT FOR EACH PPU

* BIT ASSIGNMENTS
- CPU0
- CPU1
- CEJ/MEJ
- CMU
- INSTRUCTION STACK
- SCR
- ILR
- C76A
- C76B
- C176

DEADSTART

DIO IS LOADED BY OSB (FROM CTI). DIO READS DEADSTART MEDIUM (TAPE OR DISK) DURING THE DEADSTART PROCESS. DIO LOADS SET INTO PPO AND TRANSFERS CONTROL TO IT.

SET USES PP10 AS A BUFFER FOR CMRDECK AND IPRDECK PROCESSING, AN IDLE PROGRAM, AND BUFFER PROCESSORS.

SET INITIALIZES THE SYSTEM:

1. SET LOADS CMR AND CMRINST AND FINDS THE SPECIFIED CMRDECK. CONSOLE INPUT TO ALTER THE CMRDECK MAY BE ENTERED. A "GO." OR "NEXT." CONTINUES THE DEADSTART SEQUENCE.

2. THE NEXT NON-TEXT ROUTINE (ICM) IS LOADED. ICM (INITIALIZE CENTRAL MEMORY) BUILDS THE FOLLOWING TABLES AND RETURNS TO SET.

   EST
   FNT/FST/FNT INTERLOCK TABLE
   MST/TRT/MRT
   JOB CONTROL AREAS
   CONTROL POINT AREAS
   DAYFILE POINTERS
   PP COMMUNICATION AREA

   SYSTEM
   VALIDUs
   SALVid
   RSXDid
   RSXVid

3. SET LOADS IPR AND IPRINST AND FINDS THE SPECIFIED IPRDECK. IPR SETS APPROPRIATE PORTIONS OF CENTRAL MEMORY RESIDENT - IPRL, JOB CONTROL AREAS, MSAL, ETC. - AS SPECIFIED BY IPRDECK ENTRIES AND RETURNS CONTROL TO SET.

4. SET FINDS NEXT NON-TEXT RECORD (PPR-PPU RESIDENT) AND READS IT INTO ITS PP BUFFER. PPR IS SENT TO PPU2 OVER CHANNEL 2. SET READS THE NEXT RECORD (STL - SYSTEM TAPE LOADER) AND SENDS IT TO PPU2. SET DCNs CHANNEL 2 CAUSING PPU2 TO BEGIN EXECUTING. PPU 0 ISSUES AN IAM ON CHANNEL 0 AND WAITS FOR INPUT.

5. STL IS NOW IN CONTROL. STL LOADS THE NEXT ROUTINES FROM THE DS FILE. THESE ROUTINES - CPUMTR, MASS STORAGE DRIVERS, ETC. - COMPRISE ENOUGH OF THE OPERATING SYSTEM TO ALLOW RECOVERY AND SYSEDIT TO EXECUTE WITHOUT A LOT OF SPECIAL CASING IN THE DS PROCESS.

   - A COPY OF PPR IS LOADED INTO ALL PPUs EXCEPT 0 AND 1.

   - MTR (PPU MONITOR) - NEXT RECORD ON DS TAPE - IS LOADED INTO PPU 0.

- CPUMTR IS LOADED. AN EXN STARTS CPUMTR EXECUTION. CPUMTR PRESENTS ITSELF BASED ON MODULES LOADED.
- DSD IS LOADED INTO PPU 1.
- RMS (RECOVERY MASS STORAGE) IS LOADED INTO NEXT AVAILABLE PPU (PPU 3).
- CHANNEL 0 IS DISCONNECTED WHICH CAUSES MTR TO BEGIN EXECUTION.
- MASS STORAGE DRIVERS AND REMAINING CODE (THROUGH SLL) ARE LOADED INTO A RPL (RESIDENT PERIPHERAL LIBRARY) AND A PLD (PERIPHERAL LIBRARY DIRECTORY) IS BUILT FOR THESE ROUTINES.
- SLL (SYSTEM LIBRARY LOADER) IS LOADED INTO THE NEXT AVAILABLE PPU (PPU4) ON LEVEL 0 OR 2 DEADSTARTS.
- A "LJM PPR" IS SENT TO ALL OTHER PPUs AND THEY ARE DISCONNECTED, CAUSING THEM TO BEGING EXECUTION.
- CONTROLWARE IS THEN LOADED INTO 7X5X CONTROLLERS. BCL IS READ AS AN OVERLAY TO STL; BCL READS BCS, BCF AND FMD AND OUTPUTS THE APPROPRIATE CONTROLWARE ON ALL CHANNELS WITH 7X5Xs. THE ENTIRE COPY OF THE FIRMWARE IS SENT ONE CHANNEL AT A TIME. THE EQUIPMENT TYPE AND LBC ENTRY DETERMINES WHICH CONTROLWARE IS LOADED. BCL ALSO VALIDATES WHETHER FULL TRACKING MAY BE DONE (2XPPU AND FULL TRACK ACCESS).

DEADSTART (Continued)

- LSL (LOAD SYSTEM LIBRARY) IS LOADED, IF LEVEL 0 OR 2, AS AN OVERLAY TO STL.
- LSL CCAM's TO CONTROL POINT 1, CONTROL POINT 1 IS CALLED THE "DEAD START CONTROL POINT." THE XP AND THE CONTROL POINT AREA ARE INITIALIZED; FL IS THE REMAINDER OF MEMORY UP TO 377700.
- LSL SET UP AN INPUT FILE FNT ENTRY SO THAT EVERYTHING RESEMBLES A "RUNNING" SYSTEM (UNLESS DEADSTART IS FROM DISK).
- LSL LOADS SYSEDIT INTO CP1's FIELD LENGTH AND REQUESTS THE CPU (RCPM) SO THAT SYSEDIT MAY BEGIN EXECUTION.
- SYSEDIT READS THE DS FILE VIA DIO OR CIO AND COPIES DATA DIRECTLY TO SYSTEM DEVICES USING SLL. SLL GUARANTEES EACH COPY OF THE SYSTEM USES THE SAME TRACKS.
- AT EOF, LSL RETURNS CONTROL TO STL. THE DS FILE IS REWOUND; STL MOVES TO THE SYSTEM CP VIA CCAM AND DROPS THE PPU VIA A DPPM. SYSEDIT BEGINS LIBRARY BUILDING.
- READING FROM THE FIRST SYSTEM DEVICE, SYSEDIT BUILDS THE RPL ACCORDING TO THE SPECIFIED LIBDECK AND COPIES IT TO CM USING SLL. RPL ENTRIES ARE MADE WITHOUT PREFIX (77) TABLES.

DEADSTART (Continued)

- A PPULIB FILE IS BUILT ON EACH SYSTEM DEVICE OF ALL PPU ROUTINES WITHOUT PREFIX TABLES. THE POINTER TO PPULIB IS CONTAINED IN THE SYSTEM SECTOR FOR THE SYSTEM FILE. THE PLD IS BUILT WITH THE NAME AND RESIDENCE OF ALL PPU ROUTINES IN ALPHABETICAL ORDER.
- THE RCL (RESIDENT CENTRAL LIBRARY) IS BUILT WITH THOSE CPU ROUTINES HAVING *CM LIBDECK DIRECTIVES.
- THE CLD (CENTRAL LIBRARY DIRECTORY) IS BUILT FOR ALL CPU ROUTINES, WITH THEIR NAMES, ENTRY POINTS, AND RESIDENCE.
- THE LBD (USER LIBRARY DIRECTORY) IS BUILT FOR ALL USER LIBRARIES WITH THEIR NAMES AND RESIDENCE.
- SYSEDIT COMPLETES WITH AN ENDRUN.

THE FULL OPERATING SYSTEM HAS BEEN ESTABLISHED AND IS COMPLETELY OPERATIONAL.

WHILE ALL OF THIS IS GOING ON ...

MASS STORAGE RECOVERY IS ACCOMPLISHED IN THE PPUs VIA RMS - RECOVER MASS STORAGE - AND REC - SYSTEM RECOVERY PROCESSOR.

- RMS -
  - READ MSTs FROM DEVICE LABELS.
  - VALIDATES HALF TRACK AND FULL TRACK AVAILABILITY.
  - RECOVERS TRTs.
  - VALIDATES AND RECOVERS SYSTEM DAYFILES.
  - VALIDATES AND RECOVERS PRESERVED FILES.
  - VALIDATES PERMANENT FILE CONFIGURATION.
  - LOADS REC INTO THIS PPU.

- REC -
  - READS SYSTEM TABLES (CHECKPOINT FILE)
  - RECOVERS FNT AND CONTROL POINTS.
  - ACTIVATES/INITIALIZES DAYFILES.
  - WAITS FOR SYSEDIT/SLL TO COMPLETE. (I.E., SYSTEM FILE BECOMES NOT BUSY.)
  - IQFT IS BUILT FROM QUEUED FILES FOUND ON DISK.
  - DAF INTERLOCKS ARE CLEARED (LEVEL 0).
  - DAYFILE MESSAGES ISSUED.
  - CHECKPOINT SYSTEM AND ALL DEVICES.
  - START 1TD IF TELEX RECOVERY REQUIRED.
  - STARTS SCHEDULING VIA RSJM.
  - DROP PPU VIA DPPM.

MASS STORAGE RECOVERY

RMS - RECOVER MASS STORAGE

CMS - CHECK MASS STORAGE

REC - SYSTEM RECOVERY PROCESSOR

DEVICE RECOVERY IMPLIES THAT ALL INFORMATION CONTAINED IN THE DEVICE'S LABEL IS TRANSFERRED TO THE DEVICE'S MST/TRT. AN INITIALIZATION ALTERS ALL OR SOME OF THE INFORMATION FROM THE LABEL WHEN MOVING IT TO THE MST/TRT.

DEVICE LABEL IS DEFINED IN COMMON DECKS COMSLSD.

RMS ATTEMPTS TO RECOVER ALL DEFINED MASS STORAGE EQUIPMENTS AT DEADSTART TIME.

RMS IS CALLED BY STL WITH THE RECOVERY LEVEL AS INPUT.

RMS HAS FIVE LOGICAL PHASES:

- PRESET.
- READ DEVICE LABELS.
- VALIDATE HALF TRACK AND FULL TRACK AVAILABILITY.
- CHECK AND RECOVER DEVICES.
- CALL REC INTO EXECUTION.

CMS VERIFIES THE PROPER DEVICES ARE MOUNTED AND MAKES THEM AVAILABLE FOR USER/SYSTEM ACCESS.

CMS IS CALLED ON A PERIODIC BASIS BY ISP OR WHEN AN ONLINE INITIALIZATION IS NEEDED (BY IMS). 1DS ALSO CALLS CMS AS THE RESULT OF AN UNLOAD OR MOUNT COMMAND.

CMS HAS SEVEN LOGICAL PHASES:

- PRESET.
- READ DEVICE LABELS.
- VALIDATE HALF TRACK AND FULL TRACK AVAILABILITY.
- CHECK AND RECOVER DEVICES.
- CHECK FOR INITIALIZATION REQUESTS.
- COUNT ACTIVE FAMILIES.
- CONTROLS DEADSTART SEQUENCING OPERATIONS.

REC PERFORMS A VARIETY OF RECOVERY FUNCTIONS INCLUDING:

- READING SYSTEM TABLES (CHECKPOINT FILE)
- RECOVERS FNT, CONTROL POINTS, ETC.
- INITIALIZES/ACTIVATES DAYFILES
- BUILDS IQFT (INACTIVE QUEUE)
- CLEARS DAF INTERLOCKS

REC IS LOADED INTO THE SAME PPU AS RMS BUT DOES NOT COMPLETE EXECUTION UNTIL SYSTEM FILE BECOMES NOT BUSY. SYSTEM IS BUSY IF THE SYSTEM IS BEING ESTABLISHED (LOADED) BY SYSEDIT/SLL.

RECOVERY DEADSTARTS

LEVEL 0

- SYSTEM LOADED FROM DS TAPE.
- PRESERVED FILES RECOVERED.

LEVEL 1

- SYSTEM RESTORED FROM CHECKPOINT FILE.
- JOBS AND FILES RECOVERED FROM CHECKPOINT FILE.
- SUCCESSFUL CHECKPOINT SYSTEM.
- ALL MASS STORAGE INTACT.
- CMRDECK ENTRIES SAME AS LAST LEVEL 0 OR MATCH RECONFIGURATIONS (IF ANY).

LEVEL 2

- SYSTEM LOADED FROM DS TAPE.
- JOBS AND FILES RECOVERED FROM CHECKPOINT FILE.
- SUCCESSFUL CHECKPOINT SYSTEM.
- ALL MASS STORAGE INTACT.
- CMRDECK ENTRIES SAME AS LAST LEVEL 0 OR MATCH RECONFIGURATIONS (IF ANY).

RECOVERY DEADSTARTS

LEVEL 3

- SYSTEM, FNT, MST/TRT, LOW CORE CMR AND CONTROL POINTS RECOVERED FROM CENTRAL MEMORY.
- FNT, MST/TRT, LOW CORE CMR AND CONTROL POINTS MUST BE INTACT.
- JOBS AT CONTROL POINTS RESTARTED (RERUN).
- CMRDECK ENTRIES SAME AS LAST LEVEL 0 OR MATCH RECONFIGURATIONS (IF ANY).

RECOVERY DEADSTARTS

WHEN?

LEVEL 0	INITIAL DEADSTART
	RESTART FROM UNSUCCESSFUL LEVEL 3
	AFTER MREC
LEVEL 1	CONTROLLED IDLE DOWN BY CHECKPOINT SYSTEM
	ALLOW SCHEDULED INTERRUPTION OF PRODUCTION FOR MAINTENANCE OR "SYSTEMS TIME"
LEVEL 2	CONTROLLED IDLE DOWN BY CHECKPOINT SYSTEM
	SYSTEMS TEST SITUATIONS
LEVEL 3	EQUIPMENT MALFUNCTION HANG
	SYSTEM HANG

RECOVERY DEADSTARTS

WHEN NOT?

LEVEL 0	
LEVEL 1	● UNSUCCESSFUL CHECKPOINT SYSTEM ● RUNNING AFTER CHECKPOINT SYSTEM ● UNSUCCESSFUL LEVEL 3 ● AFTER MREC
LEVEL 2	● SAME AS FOR LEVEL 1
LEVEL 3	● AFTER LEVEL 0, 1, 2 IS NOT SUCCESSFUL ● MREC HAS BEEN RUN FOR THIS MACHINE ● IF MEMORY HAS BEEN DESTROYED

FILE/MS VALIDATION

A. TRT EOI = DISK EOI
READ DISK SECTOR POINTED TO BY TRT

B. CHAIN FOR EOI
READ DISK BEGINNING AT BOI UNTIL EOI

C. BOI/EOI
READ EOI; READ BOI POINTED TO AN EOI SECTOR

D. CIRCULAR LINKAGE
IS (FIRST) TRACK POINTED TO WITHIN TRT?

E. ONLINE CHECK OF TRT
UNRESERVED TRACKS AGREE
PRESERVED FILE COUNTS AGREE
PERMIT CHAIN RESERVED AND PRESERVED
CATALOG CHAIN RESERVED, PRESERVED, POWER OF 2, CORRECT LENGTH, CONTIGUOUS (IF FLAGGED)
DATA (INDIRECT) CHAIN RESERVED AND PRESERVED

EI - ERROR IDLE (A - D)
NO PFM OR PF/QUEUE UTILITY OPERATIONS
NO NEW FILES

VE - VALIDATION ERRORS (E)

FILE/MS VALIDATION

● LEVEL 0 (OR ONLINE RECOVERY)		
DAYFILES	B (CHAIN FOR EOI)	
PERM FILE IN WRITE MODE	B (CHAIN FOR EOI)	
OTHER PRESERVED FILES	A; B (IF A FAILS) D C (IF EEOI)	
● LEVEL 1, 2		
PRESERVED FILES	A C D (IF MS VALIDATION)	
OTHER FNT ENTRIES	D (IF MS VALIDATION)	
● LEVEL 3		
PRESERVED FILES	A C D	REQUIRES BOTH MS VALIDATION AND PF VALIDATION
OTHER FNT ENTRIES	D	
● DEVICE CHECKPOINT	E (IF MS VALIDATION)	
● PERMANENT FILE ATTACH	C (IF PF VALIDATION)	

RECOVERY DEADSTARTS

LEVEL	JOBS	ACTIVE FILES	PFS	DAYFILES	SYSTEM	FILE/MS VALIDATION
0	QPROTECT	NO	YES	YES	LOADED	ALWAYS
1	CHECKPOINT FILE	CHECKPOINT FILE	YES	YES	CHECKPOINT FILE	MS
2	CHECKPOINT FILE	CHECKPOINT FILE	YES	YES	RELOADED	MS
3	CM DOPY OF FNT	CM COPY OF FNT	YES	YES	YES	MS PF

RECOVERY DEADSTARTS

ALL SYSTEM ACTIVITY SHOULD HAVE CEASED BEFORE DEADSTARTING AND ALL MASS STORAGE CHECKPOINTS SHOULD HAVE COMPLETED.

LEVEL 1/LEVEL 2

CHECKPOINT SYSTEM

LEVEL 3

```
1.ONSW1  }      IF TIME SHARING SUBSYSTEM ACTIVE
1.IDLE.  }
```

ALL LEVELS

```
E,M.            CHECK FOR DEVICE CHECKPOINTS

UNLOCK.

STEP.
```

RECOVERY DEADSTARTS

IF A SYSTEM MALFUNCTION PREVENTS A DEVICE CHECKPOINT FROM BEING DONE, THE CHECKPOINT STATUS WILL BE STILL SET OVER A LEVEL 3 DEADSTART AND WILL BE PERFORMED AFTER THE LEVEL 3 HAS BEEN SUCCESSFULLY COMPLETED.

IF THE LEVEL 3 SHOULD FAIL, AN ANALYSIS OF THE MASS STORAGE TABLES DUMP WILL REVEAL IF ANY CHECKPOINTS ARE PENDING. APPROPRIATE ACTION MAY BE TAKEN ON THE LEVEL 0 DEADSTART TO RESTORE THE FILES ON THE DEVICE.

DEVICE CHECKPOINT

CHECKPOINT - PRESERVE CURRENT CM TABLES IN DEVICE LABEL

- REQUEST CHECKPOINT IF MST/TRT ALTERED
  - CHECKPOINT STATUS IN STLL
- PERIODICALLY, 1CK IS CALLED TO CHECKPOINT DEVICES BY 1SP
- 1CK VALIDATES EACH DEVICE IF MS VALIDATION IS ENABLED
  - UNRESERVED TRACK COUNT AGREES
  - PRESERVED FILE COUNT AGREES
  - PERMIT CHAIN RESERVED/PRESERVED
  - DATA (INDIRECT) CHAIN RESERVED/PRESERVED
  - CATALOG CHAIN RESERVED/PRESERVED, POWER OF 2, CORRECT LENGTH, CONTIGUOUS (IF FLAGGED)
- 1CK WRITES TRT TO LABEL TRACK AND MST TO LABEL SECTOR
- 1CK CLEARS "CHECKPOINT REQUESTED" BIT WHEN FINISHED CHECKPOINTING THE DEVICE

SYSTEM CHECKPOINT

THE SYSTEM TABLE TRACK OR CHECKPOINT FILE CONTAINS A COPY OF CENTRAL MEMORY RESIDENT: LOW CORE POINTERS, THE EST, THE FNT, THE DAYFILE BUFFERS AND POINTERS AND FROM THE BEGINNING OF RPL TO THE END OF CMR.

CHECKPOINT SYSTEM

WHEN THE CHECKPOINT SYSTEM COMMAND IS ENTERED, THE FOLLOWING OPERATIONS OCCUR.

1. SET SENSE SWITCH FOR TIME SHARING SUBSYSTEM TO PLACE ALL USERS IN THE RECOVERY (SALVid) FILE ON AN OPERATOR DROP. TIME SHARING SUBSYSTEM IS THEN "OPERATOR DROPPED".
2. ALL JOB SCHEDULING IS INHIBITED. SAME AS IF IDLE HAD BEEN ENTERED.
3. JOBS WITH QUEUE PRIORITY   MXPS ARE ROLLED OUT.
4. ALL OTHER JOBS EXCEPT MAGNET ARE ABORTED. MOST SUBSYSTEMS WILL CLEAN THEMSELVES UP.

SYSTEM CHECKPOINT

5. MAGNET IS ROLLED OUT.
6. CHECKPOINT FILE IS WRITTEN.
7. ALL DEVICES CHECKPOINTED INCLUDING MRTs FOR NON-REMOVABLE SHARED DEVICES.
8. SYSTEM IS LEFT IN AN IDLE STATE.

CHECKPOINT BY SYSEDIT

THE CHECKPOINT REQUESTED BY SYSEDIT WRITES THE CHECKPOINT FILE WITH THE CMR INFORMATION. THIS TYPE OF CHECKPOINT CANNOT BE USED FOR LEVEL 1 RECOVERY AS FAR AS RESTARTING JOBS IS CONCERNED. IT IS PRIMARILY USED TO GUARANTEE A SUCCESSFUL LEVEL 3 RECOVERY.

QUESTION SET LESSON 11

1. Why are all the PP A registers set to 10000B?
2. What are each of the PPs doing at deadstart time?
3. Why are the positions of the first PP routines on the deadstart file important?
4. What happens to the information represented on the deadstart panel at deadstart time?
5. What does NOS do with the Hardware Descriptor Table (HDT) it receives from CTI?
6. What does SYSEDIT do at deadstart time?
7. What is the PPLIB?
8. The last step in the deadstart process is usually a system checkpoint. Why?
9. Explain the four levels of deadstart recovery and what information is recovered by each level.
10. What happens when MS VALIDATION and PF VALIDATION are specified on a level 0 deadstart? On a level 3 deadstart?
11. Why is it advisable not to deadstart with device checkpoints pending?
12. What does the Deadstart Sequencing priority accomplish?

LESSON 12
LOGICAL INPUT/OUTPUT (CIO)

LESSON PREVIEW:

This lesson overviews the logical input/output processor CIO. The mnemonic CIO means combined input/output but is often called circular input/output because of the way the data buffer in the program's field length is managed.

This lesson does not deal with the end user particulars of doing logical input/output.

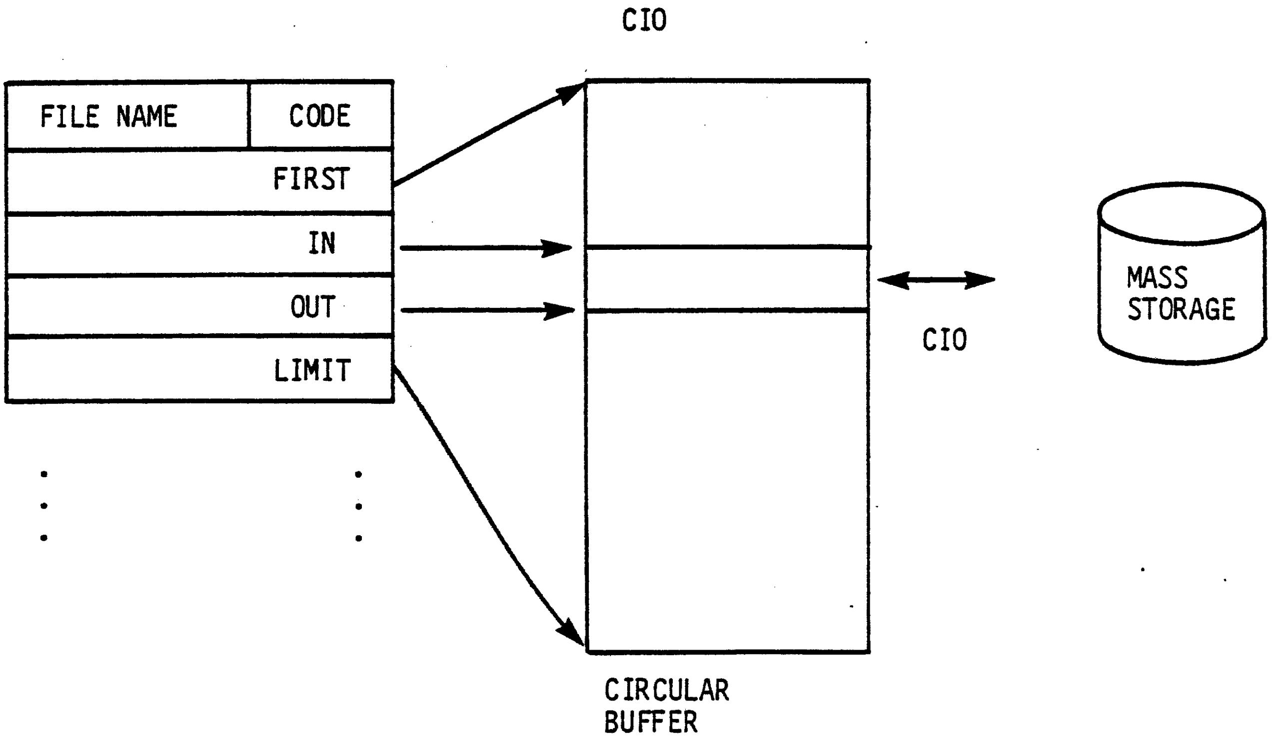
OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Overview the read and write technique to mass storage done by CIO.
- Describe how CIO does random input/output, including discussions of rewrite-in-place techniques.
- Describe how CIO processes input/output requests for magnetic tape resident files.
- Describe how CIO processes input/output requests for TTY resident files.
- Discuss how CIO knows when to terminate a read/write operation.

REFERENCES:

NOS IMS - Chapter 9 NOS Reference Manual, 2-2, 2-3 NOS Advanced Coding Student Handout



MS - I/O DONE BY CIO DIRECTLY

TAPE - CIO CALLS MAGNET

TTY - CONTROL POINT WORDS SET UP AND ROLLOUT JOB

MASS STORAGE FET

	18	14	10	9	2	1	0	
FET+0	LOGICAL FILE NAME	LN	AT	E I O	CODE	F M	1 L K	
FET+1	DEVICE TAPE	R	▨	U P	E P	▨	L	FIRST
2	0	IN						
3	0	OUT						
4	FNT ADDRESS	▨	PRU SIZE	LIMIT				
5	▨	FWA WORKING STORAGE		LIST ADDRESS LWA+1 WORKING STORAGE				
6	CURRENT RANDOM INDEX	W	RANDOM REQUEST					
7	▨	INDEX LENGTH	FWA OF INDEX					

CIO

- FET CREATION MACROS

  FILEB
  FILEC

  RFILEB
  RFILEC

- OPEN

  CLOSE
  CLOSER

- READ/WRITE MACROS

  RPHR
  READ
  READCW
  READN

  READSKP
  READLS
  RPHRLS
  READNS
  READEI

  WPHR
  WRITE
  WRITECW
  WRITEN

  WRITER
  WRITEF

  REWRITE
  REWRITER
  REWRITEF

- POSITIONING

  BKSP
  BKSPRU

  SKIPF
  SKIPFF
  SKIPEI
  SKIPB
  SKIPFB

  REWIND
  UNLOAD
  RETURN
  EVICT

  POSMF

CIO

STRUCTURE

CIO - MAIN ROUTINE

2CA - IDENTIFY SPECIAL REQUEST

2CB - READ MASS STORAGE
2CC - SPECIAL MASS STORAGE READS

2CD - WRITE MASS STORAGE
2CE - SPECIAL MASS STORAGE WRITES

2CF - POSITION MASS STORAGE
2CG - CLOSE MASS STORAGE

2CH - TERMINAL I/O

2CI - MAGNETIC TAPE I/O
2CJ - MULTIFILE TAPE LABELS

2CK - ERROR PROCESSOR

2CL - ISSUE DAYFILE MESSAGES

CIO

1. READ AND VALIDATE FET
2. VALIDATE FUNCTION CODE
3. IDENTIFY REQUEST
4. SET SKIP COUNT, BINARY/CODED STATUS
5. SEARCH FOR FILE
    - CALL 2CJ IF POSMF
    - CREATE FILE IF FILE NOT FOUND
    - SET FILE BUSY
6. CHECK FILE ACCESS
    - CHECK PERMISSIONS
    - FILE RESIDENCE FOR OPERATION LEGALITY
    - SET RANDOM REQUEST
7. VALIDATE FIRST, IN, OUT, LIMIT
8. PROCESS FUNCTION
    - LOAD OVERLAY
9. COMPLETE FUNCTION
    - SET FET COMPLETE AND ANY STATUS
    - SET FILE NOT BUSY
    - SET STATUS INTO BYTE 4 OF FST
    - SET TRACK CURRENT POSITION INTO FST
    - PERFORM ACCOUNTING
    - DROP PP

READ MASS STORAGE (2CB/2CC)

1. IF FILE NOT EXISTENT, COMPLETE REQUEST WITH EOI STATUS.
2. IF RANDOM FUNCTION, POSITION FILE (2CF).
3. IF SPECIAL READ (RPHR, READEI, READNS, READSKP, READCW, RPHRLS, READLS), CALL 2CC TO INITIALIZE READ AND SET TERMINAL CONDITION.
4. SET UP TO READ (SETMS, SET ABSOLUTE ADDRESSES).
5. RECHECK ABSOLUTE ADDRESS (IF PAUSED).
6. LOAD BUFFER.
   - DETERMINE NUMBER OF BLOCKS IN BUFFER
   - READ DATA
   - HAVE FL/100 (STSW FL VALUE) SECTORS BEEN TRANSFERRED. IF SO, COMPLETE
   - PROCESS DATA
     - EOF/EOR
     - REPOSITION IF TRACK CHANGE
   - SEND DATA TO BUFFER
   - ADVANCE IN
   - IF TERMINATION CONDITION MET, QUIT; OTHERWISE, CONTINUE READING

CIO

TERMINAL I/O

<u>WRITE</u>	<u>READ</u>
SET CONTROL INTO CONTROL POINT WORD TIOW (FET ADDRESS OF OUTPUTTING FILE)	SET CONTROL INTO CONTROL POINT WORD TINW (FET ADDRESS OF INPUTTING FILE)

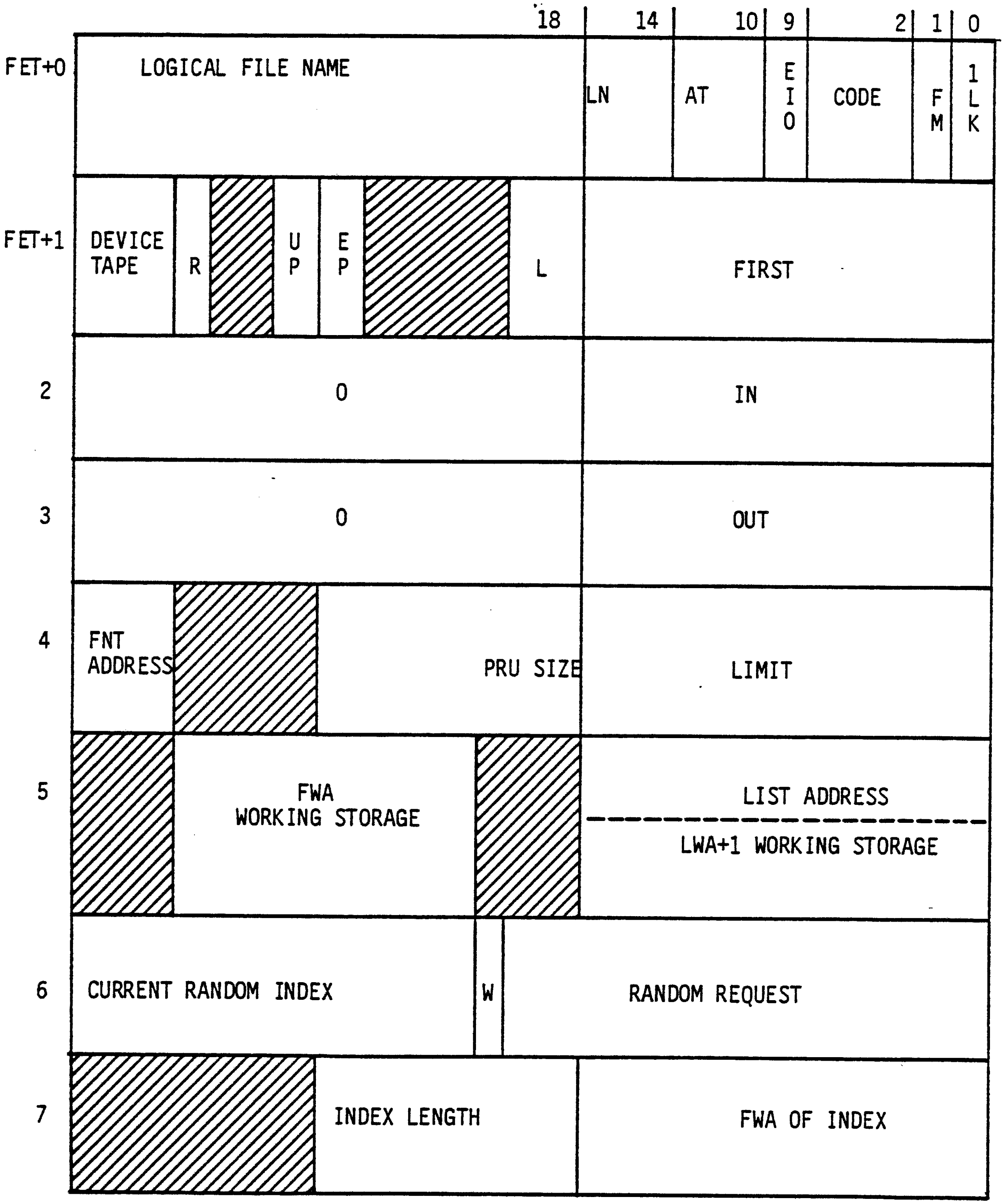
- ROLLOUT CONTROL POINT (ROCM)
- SET FILE COMPLETE
- IF NOT AUTO RECALL, DROP PP
- IF AUTO RECALL, WRITE RCLP INTO RA+1 AND DROP PP

CIO

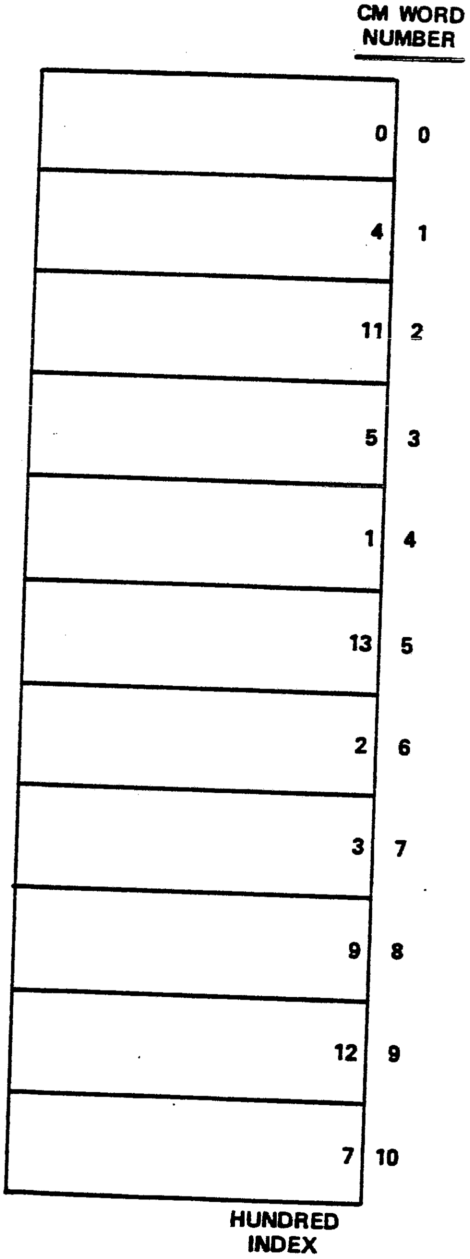
TAPE I/O

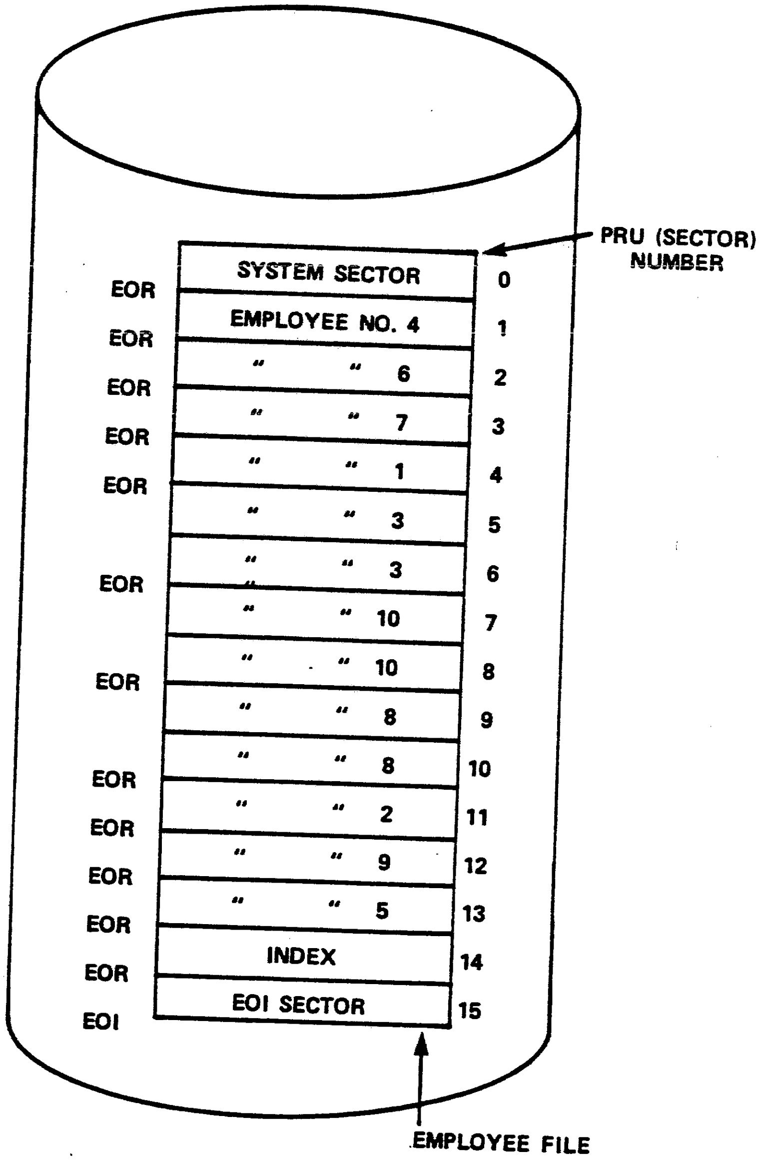
- SET UP INFORMATION FOR TAPE I/O PROCESSING
- SEND REQUEST TO MAGNET VIA TDAM FUNCTION
- IF AUTO RECALL, SET RCLP INTO RA+1
- SET UP "ACCOUNTING" TO SET TAPE ACTIVITY IN STSW
- EXIT NORMALLY

MASS STORAGE FET

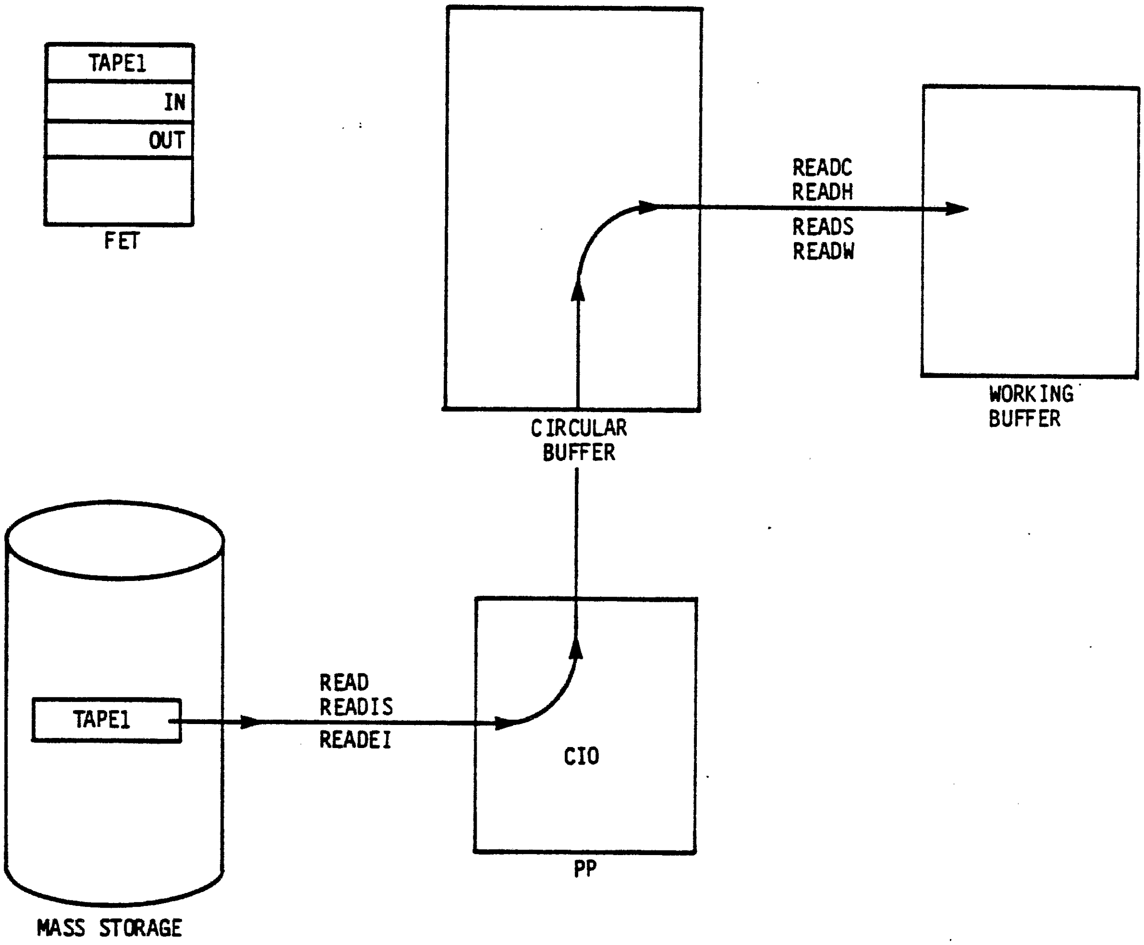


RANDOM INDEX

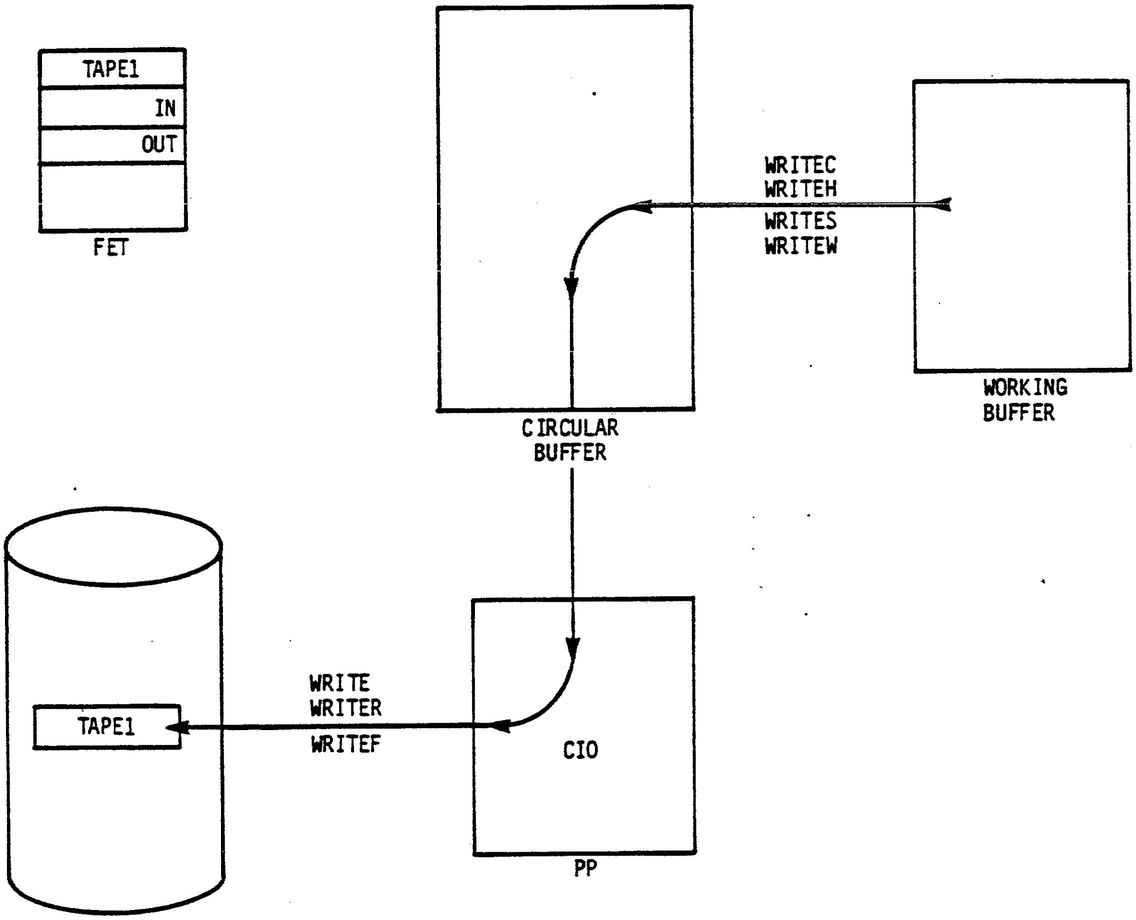




DATA TRANSFER MACROS INPUT DATA FLOW



DATA TRANSFER MACROS OUTPUT DATA FLOW



QUESTION SET LESSON 12

1. If CIO gets a request to process a non-existent file, what happens?

2. Does CIO need help from any other PP routine to do mass storage input/output?

3. Explain what happens during a mass storage read/write operation.

4. How does CIO process random I/O requests? To elaborate on your previous answer, discuss what happens on a random read request.

5. Explain the term "rewrite-in-place".

6. When doing a mass storage read or write operation, CIO will terminate the operation when a termination condition is met.

   What are the termination conditions for a read operation? A

   write operation?

7. What does CIO do when requested to perform an operation on a magnetic tape file?

8. What does CIO do when requested to perform a read or write request on a time-sharing terminal file?

LESSON 13
DISPLAY CONSOLE DRIVERS

LESSON PREVIEW:

This lesson presents the system display console driver, DSD, and the system programmer display console driver, DIS. This lesson will cover display and syntax overlay processing used by these drivers and their sharing of the display channel.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Describe the interpretative syntax technique employed by DSD.
- Map the PP memory allocation for DSD.
- Discuss the technique used to load DSD overlays.
- Explain how DSD and DIS "share" the display channel.
- Describe the function of the auxiliary PP routine 1DS.

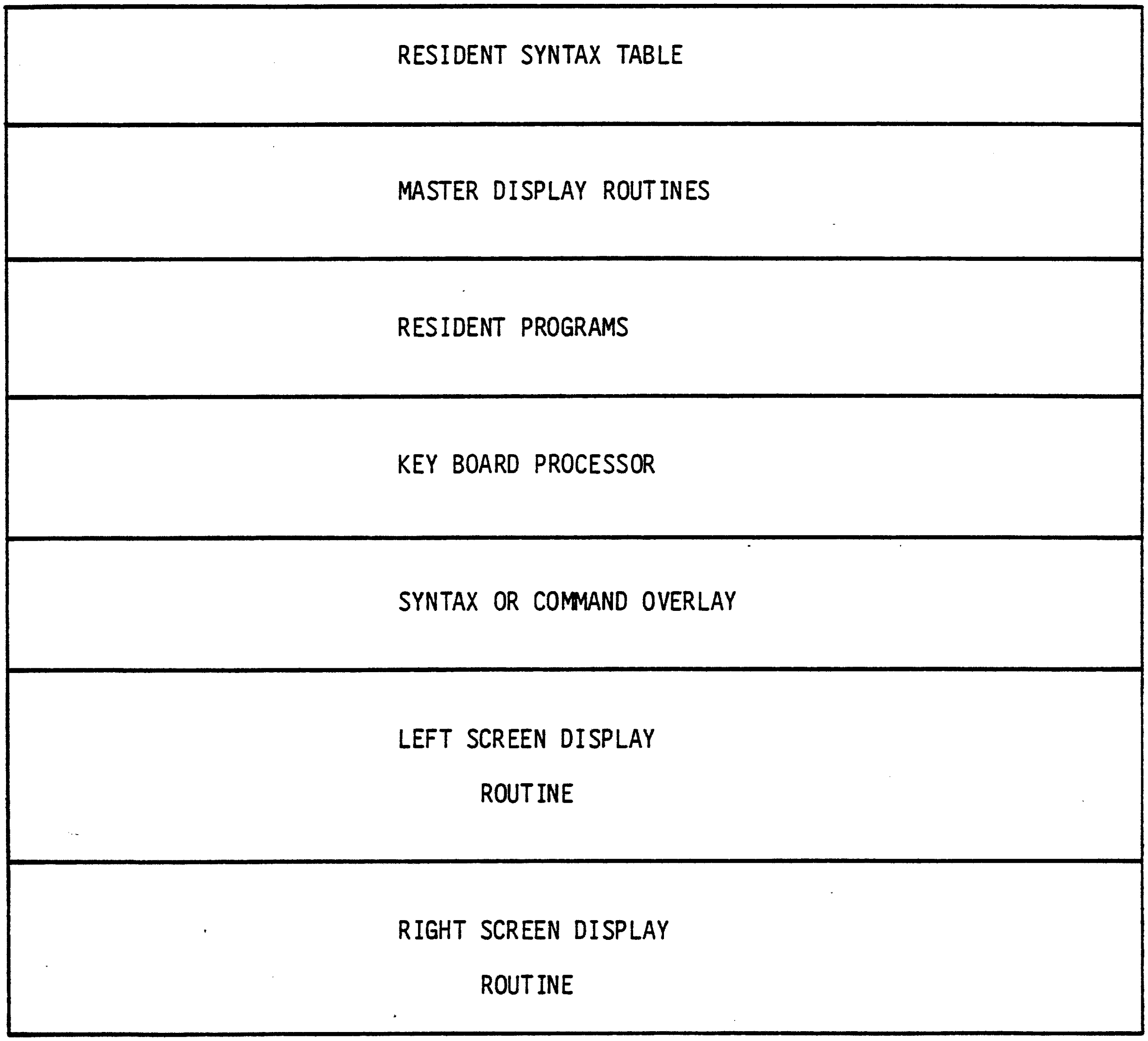
REFERENCES:

NOS IMS - Chapter 27 NOS Operator's Guide, 3, 4, 8 NOS System Programmer's Instant, 1, 2

OVERLAYS

- SYNTAX
- DISPLAY
- COMMAND PROCESSOR

DSD (PP1)



DSD

DSD PROVIDES OVERALL SYSTEM STATUS VIA THE DISPLAY CONSOLE

ACCEPTS OPERATOR COMMANDS TO CONTROL OPERATION OF SYSTEM PERIPHERALS AND SYSTEM PERFORMANCE

INTERPRETIVE SYNTAX TABLES

ENABLE, REMOVABLE PACKS

E SYNTAX OVERLAY FOR "E"

EN SYNTAX OVERLAY FOR "EN"

ENA B L E,

SYNTAX OVERLAY FOR "ENABLE"
REMAINING SYNTAX FILLED IN

ENABLE, R E M O V A B L E _ P A C K S,
REMAINING SYNTAX FILLED IN

ONCE COMMAND IS DETERMINED, THEN DSD LOADS THE PROCESSOR OVERLAY

1DL

DISPLAY OVERLAY LOADER

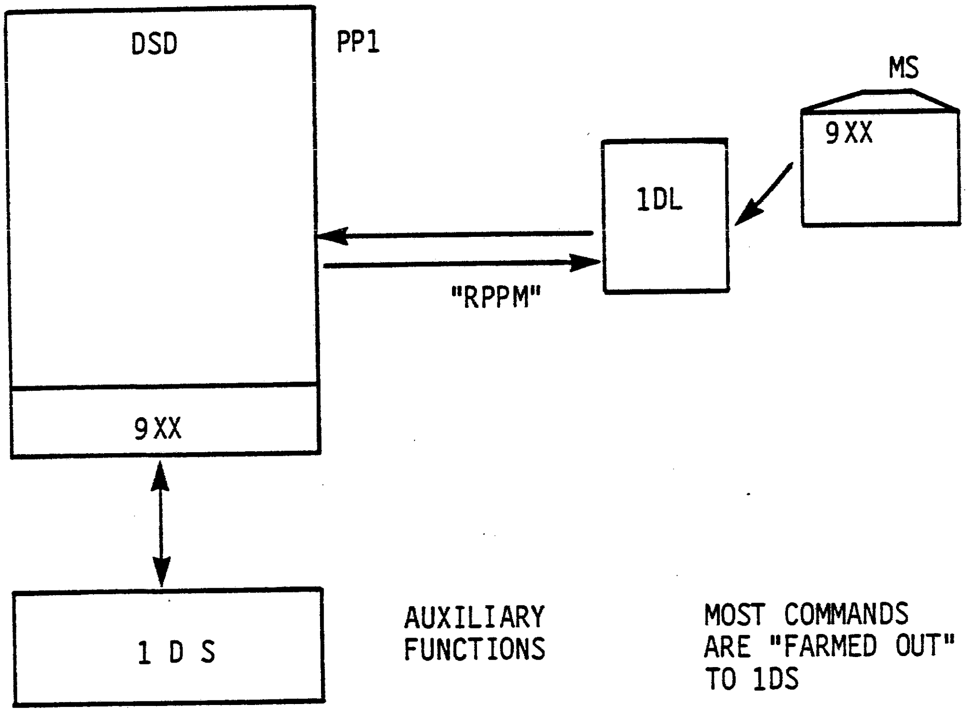
- SYSEDIT BUILDS DISPLAY OVERLAYS WITH ONLY THE FIRST ENTRY OF A GROUP

  9AA . . .
  9BA . . .
  9CA . . .

  EXCEPT FOR RCL OR ASR DISPLAY OVERLAYS

- FROM BASE OVERLAY, 1DL READS DOWN THAT TRACK CHAIN UNTIL DESIRED OVERLAY IS FOUND
- 1DL TRANSFERS THE OVERLAY TO THE REQUESTING PPU OVER A SPECIFIED CHANNEL
- 1DL MAY ALSO TRANSFER THE OVERLAY TO THE REQUESTING PPU ROUTINE USING THAT PPU's MESSAGE BUFFER

DSD



1DS FUNCTIONS ARE FOUND IN COMMON DECK COMS1DS. ANY ROUTINE CAN USE 1DS BY PASSING THE COMMAND TO 1DS IN THE DSD/1DS COMMUNICATION BUFFER IN LOW CORE.

QUESTION SET LESSON 13

1. How and why does DSD use 1DS?
2. Map the memory allocation of the DSD PP.
3. What is meant by "interpretative syntax"?
4. How do DSD and DIS both use the display console and its channel?
5. How does DSD load its overlays?
6. How are the console displays formatted?

LESSON 14
JOB PROCESSING

LESSON PREVIEW:

This lesson describes the flow of a job through the system. The student, having developed a basic foundation of NOS in the preceeding lessons, now begins to see the various system components being brought together to accomplish the processing of user jobs.

This lesson covers all aspects of the flow of a job through the system. The individual topics covered are: FNT interlocking, scheduling, job initiation and completion, rollin and rollout, control card processing, error processing, field length management, special entry point processing with emphasis on DMP= and SSJ=, and subsystem scheduling.

The scheduling, job initiation and completion, and rollin/rollout discussions are coupled with the NOS FNT interlock mechanism and cover how a job is brought to and removed from a control point.

A "rollout" is the copying of all information about a job from central memory and ECS (control point area, dayfile buffer, FNT/FST entries, and program field length (CM and ECS)) to a mass storage file called the "rollout file". A "rollin" is the copying of the information written on a rollout file to a control point (may be the same control point or different from the control point number that was vacated), restoring the control point area, dayfile buffer, FNT/FST entries, and CM and EC program field lengths. The program resumes execution with the rollout/rollin process completely transparent to the user program (with some exceptions for certain types of rollout).

"Special Entry Points" serve as flags and/or values that indicate a requirement for special processing. The special entry points, while entry points in the program in which they are assembled, are entered into the Central Library Directory (CLD) entry for the program as bits in a special entry point word rather than as individual entry points.

The field length management discussion covers the determination of the initial field length for each job step. The algorithm used to determine the initial field length for the job step, various field length related definitions, and the RFL= and MFL= special entry points are covered in detail.

The error processing discussion details the techniques available to recover from hardware/software/limit errors that may be incurred by the user's job.

The subsystem scheduling discussion presents the concept of a NOS "subsystem". The priorities and requirements of subsystems will be discussed, as well as the special operations done to schedule as the them.

The discussion of DMP= and SSJ= special entry points also covers the "special call mechanism". The operations governed by DMP= and SSJ= are quite complex because they each require the transfer of data between job steps and are often used together in the same job step (program). The DMP= entry point causes the previous program field length to be written on a file called "DM*" in a manner similar to the rollout file (except field length is not written in a reverse order). This allows the control point's field length to be used by another CPU program transparent to the user program. The system uses this technique to take advantage of system features available to CPU programs to do system related tasks. The SSJ= entry point causes a block of user validation data to be retrieved from the control point area when the program is loaded and replaced in the control point area when the program terminates. The data being manipulated are words UIDW, ALMW, ACLW, and AACW. SSJ= also carries with it certain permissions for which particular system imposed constraints and validations may be overriden; and the capabiltiy of setting time limit, CPU and Queue Prioirties for the job step. The special call mechanism uses control point area word SPCW to make a request for executing a CPU program in the control point's field length and then returning to the original program. The SPCW mechanism is available only to a PP program but may be triggered by a user program making a RA+1 request, which is then processed by SFP.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Detail the major steps in processing a user's job from its entry into the input queue to the disposal of its output from the output queue.
- Describe in detail the Job Control Block (JCB), what each field means, and how the block is entered with data.
- Explain the concept of "priority aging" and how it is accomplished in NOS.
- Explain how the scheduler (either 1SJ or 1SP) is called into execution.
- Describe the selection criteria for identifying a candidate job to be brought to a control point for subsequent execution, and what technique 1SJ uses to accomplish bringing the candidate to a control point.
- Describe the selection criteria for choosing the control point for the candidate job.
- Identify the various tables used by the scheduler in order to select the candidate job.
- Describe the selection process by major subroutines in the scheduler (1SJ) and the priority evaluator (1SP).
- Explain who determines if a job's CM or CPU time slices have expired and what is done when these values are exceeded.
- Discuss the various mechanisms by which a job may be introduced to the system (i.e., entered into the input queue).
- Describe how NOS determines the name of a job.

- Describe the initialization of the control point area by 1AJ overlay 3AA (Begin Job) when a job is brought to a control point to begin its execution from the input queue.
- Identify the special entry points processed by NOS and what each entry point indicates.
- Describe the CLD entry of a routine with special entry points.
- List the steps involved in processing a control statement.
- Describe the differences between operating system argument processing and product set argument processing.
- Identify steps in the control statement processing where the processing may be altered due to the presence of a special entry point.
- Discuss the loading of programs (both absolutes and relocatables).
- Define the terms: SYSMAX, MAXFL, MFL, RFL, NFL and SYSDEF.
- Describe the initial field length selection process for central memory and for user access ECS.
- Explain the use of the RFL= and MFL= special entry points and the *FL LIBDECK value in field length management.
- Compute a program's initial field length when given the current field length, the contents of FLCW and the program's CLD entry.
- Discuss the job clean-up operations performed by routine 1CJ (Complete Job).
- Discuss extended reprieve, reprieve, and EREXIT processing.
- Describe what happens when a program is aborted due to an error. This discussion should include the function of 1AJ overlay 3AB (Error Processor), file flushing, list of files, and so forth.
- Identify the contents of the rollout file and its system sector.
- Describe the rolling mechanism between mass storage and central memory.
- Describe the DMP= entry point value, detailing the options that may be specified by it.
- Describe the DMP= process when that process is required due to a control statement request from a program that has a DMP=entry point.
- Describe the DMP= process when that process is required due to a SPCW request from a PP program that had been called by a user program RA+1 request.
- Describe the SSJ= entry point value, detailing the options that may be specified by it.

- Describe the SSJ= process, what data is transferred, how priorities and time limit may be specified, and what processing is done at the end of a job step for files created by the SSJ= job step.
- Discuss the processing done when a program combines DMP= and SSJ=processing.
- List the characteristics of a subsystem.
- Explain the queue priority relationship for a subsystem.
- Identify the NOS subsystems, their queue priorities, and any control point requirements.
- Discuss the mechanism by which subsystems are scheduled, detailing the use of the SSCL and SSTL words of CMR and the building of the FNT/FST input queue entry for the subsystem initialization routine by 1DS.

REFERENCES:

NOS IMS - Chapter 5, 6 and 35 NOS RM, 1-2, 1-3, 1-5, 1-6 NOS RM, 2-6, 2-10, 2-F NOS SMRM, 6-5 NOS IHB, 8-5

SUPPLEMENTAL TEXT: FNT INTERLOCKING

The interlocking of a FNT entry provides protection from alteration or processing by another routine while that entry is currently being manipulated. The concepts involved in interlocking an FNT entry include protecting the job while it is moving from one job state to another (transition state), protecting queue entries, and guaranteeing that the FNT is not being accessed simultaneously.

Individual FNT Interlock

Interlocking an individual FNT entry is done by the SFIM monitor function. This function sets or clears an interlock bit in the "FNT Interlock Table" for a particular FNT entry. The individual interlock technique is used in the following circumstances:

Bringing an input file into execution
Performing a job advance
Rolling in or rolling out a job
Terminating a job
Altering the FNT or system sector of a queued file
Moving a file from one queue to another
Assigning a queue file to a control point

Global FNT Interlock

The FNT may be globally interlocked by the reservation of the FNT pseduo-channel FNCT. The use of this mechanism is to avoid conflicts which may occur when more than one system routine attempts to update the FST entry of a queued file. The global interlock is only used when the contents of queued file FSTs are to be altered. As the priority evaluation scheme is disabled by this interlock, it should be used with caution.

In cases where the individual FNT interlock (SFIM) and global interlock (FNCT) are both required, the SFIM interlock should be obtained first. This order must be maintained to avoid deadlock situations.

An example of where the FNCT interlock is used is the DSD command ENQP. 1SP is periodically called to do queue priority evaluation and will update the priority field in queued file FSTs. DSD will update the priority field when performing the ENQP command. If both DSD and 1SP tried to update the same FST, a conflict would occur. DSD performs the following sequence to avoid the possibility of making a conflicting ENQP entry. First, the desired FNT is interlocked via SFIM. Then the entire FNT is interlocked by DSD reserving the FNCT pseudo-channel. When DSD completes, it drops the FNCT pseudo-channel and then clears the SFIM interlock.

FNT Entry Interlock

The FNT is also globally interlocked by those system routines making new FNT entries by the reservation of the FNT Entry pseudo- channel FECT. This mechanism guarantees that a system routine may determine when and where within the FNT to write a FNT/FST entry without being disturbed by another system routine making FNT/FST entries.

An example of the use of the FECT interlock mechanism is found in routine OBF. OBF obtains the FECT interlock by reserving the FECT pseudo-channel. It then scans the FNT for an empty position. OBF writes the new entry at this position and then drops the pseudo- channel, clearing the interlock.

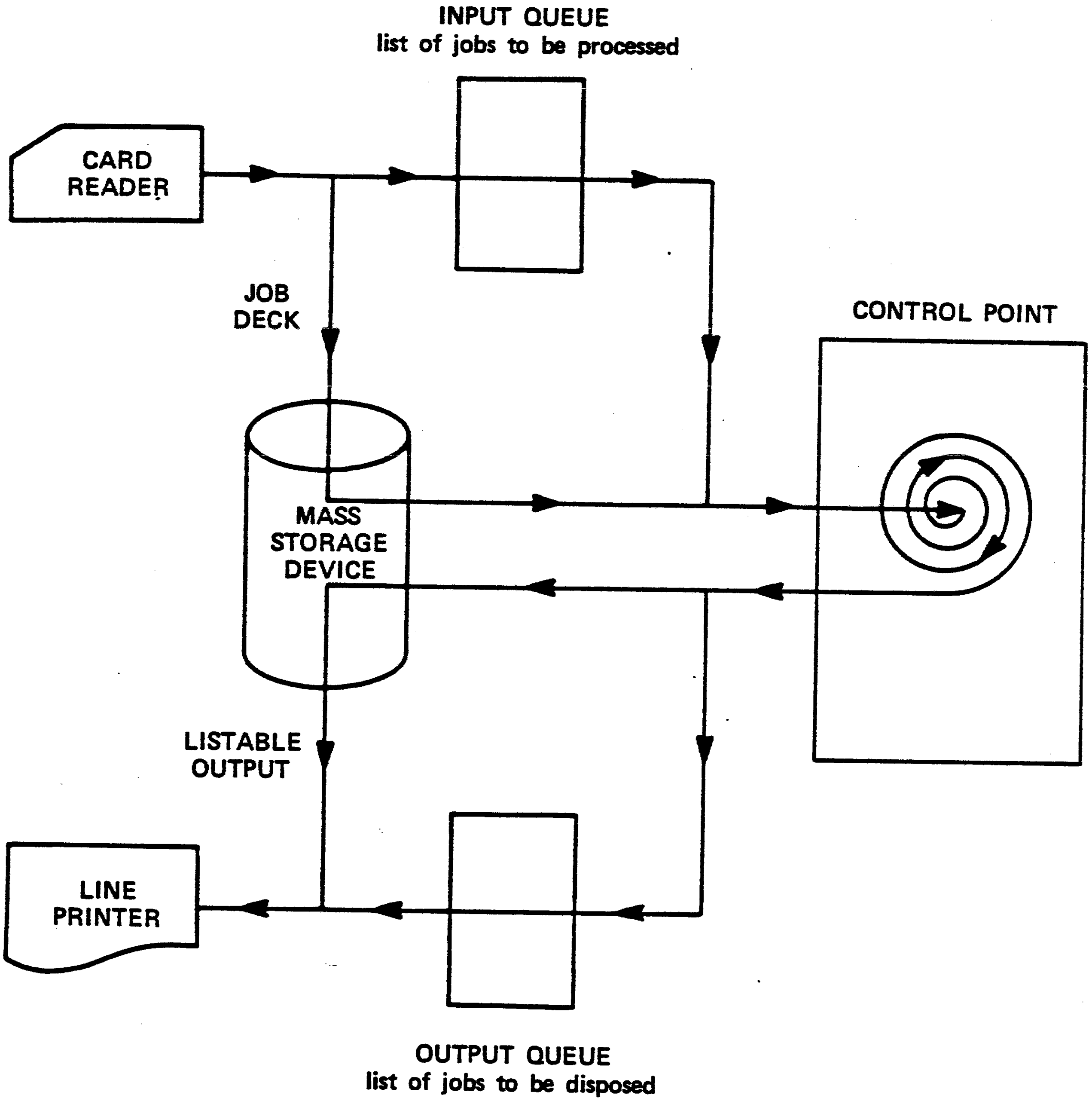
Job Advancement

The individual FNT interlock must be not set on the job's input file in order for the job to be advanced. The job advancement process will automatically set the FNT interlock on the job's input file to indicate that it is in a transition state. Thus, the FNT interlock will always be set for a job if the job advancement flag is set; the converse of this is not true, however. The issuance of the JACM monitor function by the system routines involved in the job advancement process (1AJ, 1RO, 1CJ) will clear the FNT interlock when the job advance flag is cleared. The job advancement process sets the interlock for the FNT/FST position pointed to by control point word TFSW.

Transition State Scheduling

In order to properly control transition state activity, it is necessary to set the FNT interlock on the queued file or input/ rollout file being manipulated before any transition activity may take place. With this structure, system routines are able to identify when transition states are completed by the successful issuance of their own FNT interlock (SFIM) request. This in turn prohibits a transition state from occuring while they perform their specified function on that job or queued file.

GENERAL SYSTEM FLOW



JOB ENTERS SYSTEM

JOB SCHEDULED FOR EXECUTION

JOB BEGINS EXECUTION

CONTROL STATEMENTS PROCESSED

JOB ROLLS OUT

JOB SCHEDULED TO RESUME EXECUTION

JOB ROLLS IN

...

JOB COMPLETES

OUTPUT PROCESSED

1. ISP PERIODICALLY EVALUATES QP-ADVANCES QP

2. 1SJ SCHEDULES JOB FROM INPUT QUEUE

3. 1SJ CALLS 1AJ TO BEGIN JOB

4. CPUMTR CALLS 1AJ TO ADVANCE JOB

   - JACM

5. 1SP EVALUATES QP.

   1SP EVALUATES CPU TIME SLICE AND CM SLICE. IF EITHER HAS EXPIRED, JOB'S QP IS SET TO THE JOB ORIGIN LOWER BOUND FOR TYPE (INPUT OR ROLLOUT).

6. 1RO ROLLS OUT JOB

   THE JOB INPUT FNT/FST ENTRY POINTED TO BY TFSW WILL BECOME THE FNT/FST ENTRY FOR THE ROLLOUT FILE. DURING THE "TRANSITION STATE" THE FNT INTERLOCK HAS BEEN SET ON THIS ENTRY VIA SFIM.

7. 1SJ (CALLED BY 1SP OR CPUMTR) SCHEDULES THE JOB TO BE ROLLED IN FROM THE ROLLOUT QUEUE. THE FNT/FST ENTRY IS INTERLOCKED VIA SFIM DURING THIS TRANSITION STATE.

8. 1RI ROLLS IN THE JOB.

   THE JOB GETS A FRESH CPU TIME SLICE AND CM SLICE. ROLLOUT FNT/FST ENTRY IS REPLACED BY THE INPUT FNT/FST ENTRY. THE FNT INTERLOCK IS RELEASED WHEN THE JOB IS READY TO RESUME EXECUTION.

9. WHEN 1AJ DETECTS THE END OF CONTROL STATEMENTS OR DETECTS A FATAL ERROR CONDITION, IT CALLS 1CJ TO COMPLETE THE JOB.

10. 1CJ RETURNS ALL FILES, DISPOSES OUTPUT FILES TO OUTPUT QUEUES WITH JOBNAME AND PRFT/PHFT, DOES JOB ACCOUNTING, APPENDS DAYFILE TO OUTPUT FILE.

11. OUTPUT IS RETRIEVED FROM OUTPUT QUEUES BASED ON THE ROUTING INFORMATION. THIS OPERATION IS USUALLY PERFORMED BY QAC.

12. 1SP EVALUATES QP FOR OUTPUT FILES...

FNT INTERLOCKING

INDIVIDUAL ENTRY INTERLOCK

- SFIM MONITOR FUNCTION
- TRANSITION STATES
    - BRING JOB INTO EXECUTION
    - JOB ADVANCE
    - ROLLING IN OR OUT A JOB
    - JOB TERMINATION
    - ALTERING FNT OR SYSTEM SECTOR OF QUEUED FILE
    - MOVING FILE FROM ONE QUEUE TO ANOTHER
    - ASSIGNING QUEUE FILE TO CONTROL POINT

GLOBAL INTERLOCK (CHANNEL 15)

- FNCT PSEUDO-CHANNEL

FNT ENTRY INTERLOCK (CHANNEL 14)

- FECT PSEUDO-CHANNEL

JOB CONTROL BLOCK (JCB)

ONE BLOCK FOR EACH ORIGIN/CLASS.

1. QUEUE CONTROL   QUEUE CONSOLE OR IPRDECK ENTRY

FIRST 3 WORDS OF JCB ARE USED FOR JOB AGING - IN INPUT, ROLLOUT AND OUTPUT QUEUES.

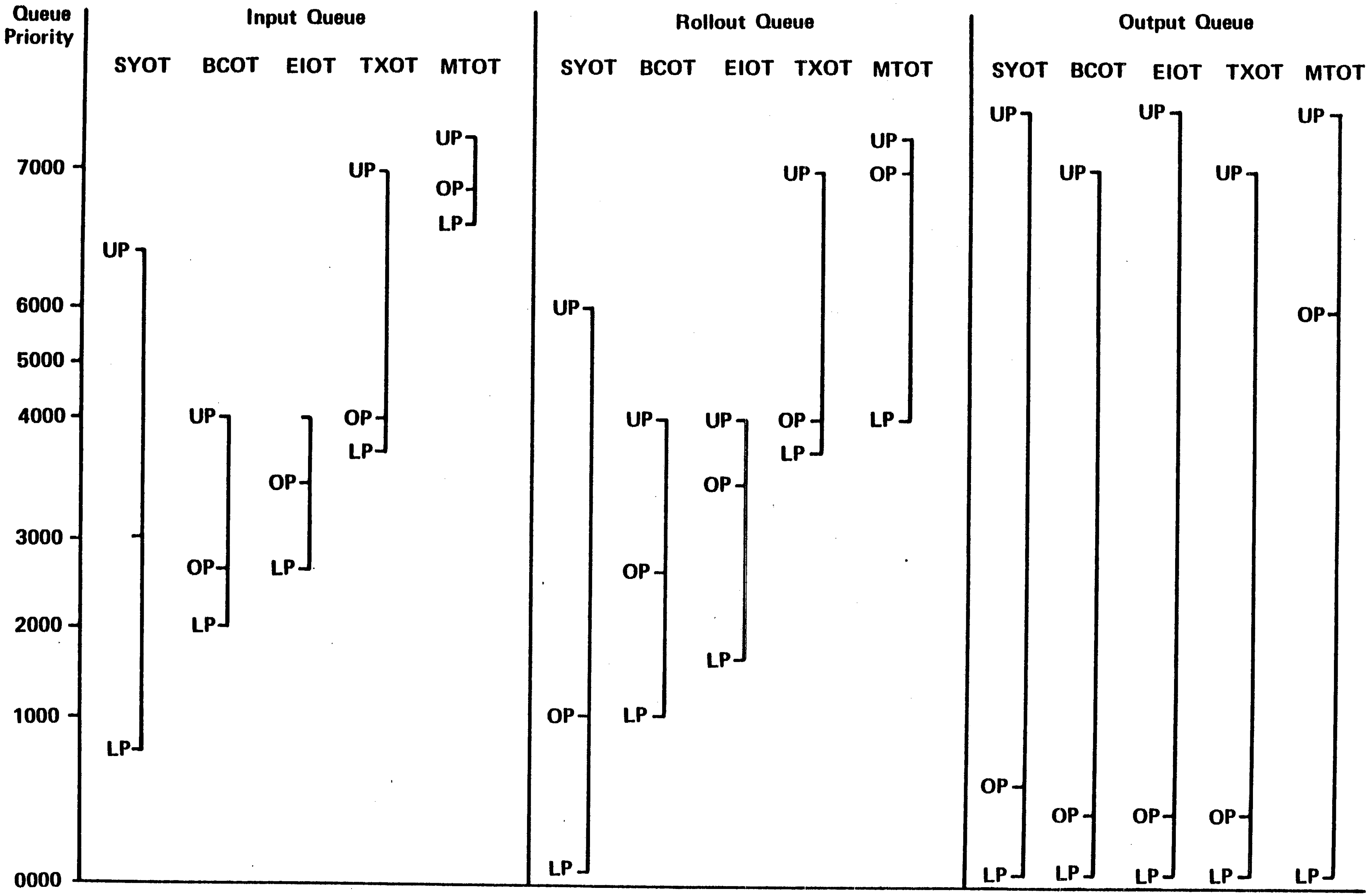
BYTE 0 - ORIGINAL (ENTRY) QUEUE PRIORITY (OP)
BYTE 1 - LOWER BOUND FOR QUEUE PRIORITY (LP)
BYTE 2 - UPPER BOUND FOR QUEUE PRIORITY (UP)
BYTE 3 - PRIORITY INCREMENT (IN)
BYTE 4 - INCREMENT INTERVAL COUNT

OP	LP	UP	IN	COUNT

JOB PRIORITY MUST BE BETWEEN LP AND UP TO BE AGED IN THE QUEUE.

EACH TIME 1SP CYCLES THROUGH THE QUEUES, BYTE 4 IS ADVANCED BY ONE. IF BYTE 3 = BYTE 4, THE QP FOR ALL JOBS OF THIS QUEUE AND ORIGIN TYPE IS INCREMENTED BY 1, BYTE 4 IS CLEARED.

THE QUEUE CONTROL MECHANISM ALLOWS A SITE TO GIVE PREFERRED TREATMENT TO JOBS OF A CERTAIN ORIGIN.


Queue Priority
Input Queue
Rollout Queue
Output Queue
SYOT
BCOT
EIOT
TXOT
MTOT
7000
6000
5000
4000
3000
2000
1000
0000
UP
OP
LP

JOB CONTROL BLOCK (JCB)

2. CONTROL POINT CONTROL SERVICE CONSOLE OR IPRDECK ENTRY

FOURTH WORD OF JCB CONTROLS A JOB WHILE AT A CP (BY ORIGIN)

BYTE 0 - INITIAL CPU PRIORITY SET AT START OF JOB OR LOGIN (PR)
BYTE 1 - CPU TIME SLICE IN MILLESECS/64_{10}(CP)
BYTE 2 - CM TIME SLICE IN SECONDS (CM)

PR	CP	CM		

JOB LEAVES CM BECAUSE:

- COMPLETES OR ABORTS
- TERMINAL INPUT/OUTPUT IS REQUIRED
- CM IS NEEDED BY A HIGHER PRIORITY JOB (KEEPS JOB FROM MONOPOLIZING CM)

SIMILAR LOGIC IS FOLLOWED FOR THE CPU.

A JOB REMAINS IN CM AND MAY USE THE CPU UNTIL/UNLESS A HIGHER PRIORITY JOB NEEDS THE MEMORY OF THE CPU.

JOB CONTROL BLOCK (JCB)

3. MEMORY CONTROL SERVICE CONSOLE OR IPRDECK ENTRY

FIFTH WORD IN JCB HAS MAXIMUM MEMORY ALLOWED FOR JOBS WITHIN THE JOB ORIGIN.

BYTE 0 - MAXIMUM NUMBER OF JOBS (NJ)
BYTE 1 - MAXIMUM FL/100 FOR ANY JOB (FL)
BYTE 2 - MAXIMUM FL/100 FOR ALL JOBS (AM)
BYTE 3 - MAXIMUM ECS FL FOR JOB (EC)
BYTE 4 - MAXIMUM ECS FL FOR ALL JOBS (EM)

NJ	FL	AM	EC	EM

FL/EC AND AM/EM ARE IGNORED WHEN SYSTEM IS IDLE. THE SCHEDULAR WILL ATTEMPT TO KEEP THE SYSTEM "BUSY" REGARDLESS OF FL/EC AND AM/EM.

SCHEDULING

JOB SELECTION

1. HIGHEST PRIORITY (QP) JOB THAT FITS INTO UNASSIGNED OR AVAILABLE MEMORY WITHIN SERVICE CONSTRAINTS (FL/AM, EC/EM) FOR THE CANDIATE'S ORIGIN TYPE.

2. EQUAL CANDIDATES - SELECT JOB ON THE MASS STORAGE DEVICE WITH LEAST ACTIVITY

   - NO FREE CHANNEL
   - CHANNEL REQUESTED MS ACTIVITY
   - FIRST UNIT RESERVED

3. MS ACTIVITY EQUAL - SELECT JOB WITH LARGEST FL.

4. IF NO JOB WAS SELECTED BUT ONE WAS REJECTED BECAUSE OF SERVICE CONSTRAINTS, IT WILL BE SELECTED IF NO JOBS HAVE TO BE ROLLED OUT FOR IT TO FIT. ITS QP WILL BE SET AT THE ORIGIN'S LOWER BOUND.

   THIS STEP PREVENTS THE SYSTEM FROM SITTING IDLE DURING PERIODS OF LOW ACTIVITY.

SCHEDULING

CONTROL POINT SELECTION

1. CONSIDER A CONTROL POINT'S FIELD LENGTH TO BE ALL THE FL OF UNOCCUPIED CPs FOLLOWING THE CANDIDATE. THE SELECTION ORDER IS -

   - EXACT FIX
   - SMALLEST HOLE THAT IS LARGER THAN NEEDED
   - LARGEST HOLE IF NONE IS BIG ENOUGH
   - IF NO CPs ARE AVAILABLE OR ROLLING OUT, THE FIRST CP ENCOUNTERED WITH A LOWER QUEUE PRIORITY (QP) IS SELECTED TO BE ROLLED OUT. IF ALL CPs AHVE A HIGHER QP, NONE IS SELECTED.

2. FL IS OBTAINED BASED ON THE CP SELECTED. THE ABOVE SEQUENCE IS SUCH THAT THE CP SELECTED REQUIRES THE MINIMUM AMOUNT OF STORAGE MOVEMENT.

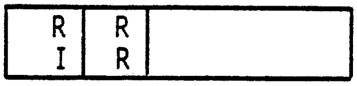
1SJ

TABLES

TACP ACTIVE CONTROL POINTS IN DESCENDING PRIORITY

RI=ROLLOUT IN PROGRESS
RR=ROLLOUT REQUESTED

TRST ROLLOUT STATUS INDEXED BY CONTROL POINT NUMBER



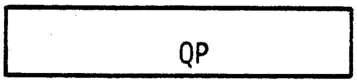
TJFL JOB FIELD LENGTH INDEXED BY CONTROL POINT NUMBER

BYTE 4 OF STSW

TJEC JOB ECS FIELD LENGTH INDEXED BY CONTROL POINT NUMBER

BYTE 4 OF ECSW

TJPR JOB PRIORITY INDEXED BY CONTROL POINT NUMBER



BYTE 1 OF JCIW

TJOT JOB ORIGIN TYPE INDEXED BY CONTROL POINT NUMBER

RIGHT 6 BITS
OF BYTE 3 JNMW

TMFO TOTAL AVAILABLE FIELD LENGTH FOR ALL JOBS OF AN ORIGIN TYPE (INDEXED BY ORIGIN TYPE)

AM SERVICE VALUE

TMEO TOTAL AVAILABLE ECS FIELD LENGTH FOR ALL JOBS OF AN ORIGIN TYPE (INDEXED BY ORIGIN TYPE)

EM SERVICE VALUE

SCHEDULING (Continued)

TAFO	ASSIGNED FIELD LENGTH BY ORIGIN TYPE (INDEXED BY ORIGIN TYPE)	AFL
TAEO	ASSIGNED ECS FIELD LENGTH BY ORIGIN TYPE (INDEXED BY ORIGIN TYPE)	AEC

SCHEDULING

1SJ	TABLES	
TMJO	MAXIMUM FIELD LENGTH PER JOB BY ORIGIN (INDEXED BY ORIGIN TYPE)	$\boxed{\text{FL}}$ FL SERVICE VALUE
TMXO	MAXIMUM ECS FIELD LENGTH PER JOB BY ORIGIN TYPE (INDEXED BY ORIGIN TYPE)	$\boxed{\text{EC}}$ EC SERVICE VALUE
CSTB	CHANNEL STATUS TABLE	EXACT COPY OF CITL FROM LOW CORE CMR
DACT	DEVICE ACTIVITY COUNT, INDEXED BY EQUIPMENT NUMBER	COUNT= NO FREE CHANNEL + CHANNEL REQUEST + FIRST UNIT RESERVE
ESTB	EST - MASS STORAGE ENTRIES	COPY OF EST ENTRY

SCHEDULING

1SJ SUBROUTINES

- SET CONTROL POINT STATUS SCS

  BUILDS TACP, TJFL, TRST, TJOT, TJPR, TAFD,
  TJEC, TAEO

- SET JOB CONTROL SJC

  BUILDS TMJO, TMFO, TMXO, TMEO

- DETERMINE DISK ACTIVITY DDA

  BUILDS CSTB, DACT, ESTB,
  CSTB AND ESTB ARE USED ONLY TO BUILD
  THE DACT.

- SEARCH FOR JOB SFJ

  CHOOSES BEST CANDIDATE. IF NO JOB CHOSEN ON FIRST PASS, SETS TMJO/TMXO AND TMFO/TMEO TO UNLIMITED, DISALLOWS ROLLOUT, AND MAKES A SECOND PASS.

- COMMIT FIELD LENGTH CFL

  SELECTS JOBS NEEDED TO BE ROLLED OUT TO OBTAIN DESIRED AMOUNT OF MEMORY. JOBS OF SAME ORIGIN ARE ROLLED BEFORE JOBS OF DIFFERENT ORIGINS.

- COMMIT CONTROL POINT CCP

  SELECTS CONTROL POINT FOR THE JOB.

- ASSIGN JOB ASJ

  REQUEST STORAGE, SET JNMW, TFSW, JCIW, TSCW AND CALLS 1AJ or 1RI (INTO THIS PP IS ANOTHER IS NOT AVAILABLE)

SCHEDULING

1SP

WHAT 1SP DOES (BY SUBROUTINE)

● ADJUST JOB PRIORITIES CM AND CPU TIME SLICE EVALUATION USER ECS DISABLED	AJP
● ADVANCE TIME INCREMENT ADDS TO BYTE 4 OF QUEUE CONTROL WORDS. IF BYTE 3 = BYTE 4, CLEARS BYTE 4. THIS IS THE FLAG USED TO DETERMINE WHETHER TO ADVANCE QA.	ATI
● ADJUST FILE PRIORITIES IF BYTE 4 OF QUEUE CONTROL FOR QUEUE TYPE AND JOB ORIGIN IS 0, ADD ONE TO QP IN FST ENTRY.	AFP
● CHECK EVENT TABLE IF EVENT HAS OCCURRED FOR TIMED/EVENT ROLLOUT FILES, THEY ARE MADE READY FOR SCHEDULING (QP = UP) AND EVENT TABLE CLEARED. READS SYSTEM SECTOR TO RESET FST ENTRY.	CET
● CHECK MASS STORAGE CALLS CMS TO RECOVER REMOVABLE DEVICES, CALLS 2SP TO POST MASS STORAGE ERRORS, AND SO FORTH.	CMS

SCHEDULING

<u>1SP</u>

● CHECK DEVICE CHECKPOINT CALLS 1CK IF CHECKPOINT BIT SET IN MST FOR A MS DEVICE	CDV
● PROCESS ACCUMULATOR OVERFLOW CALLS OAU TO PROCESS SRU ACCUMULATOR OVERFLOW	POF

SCHEDULING

CPU OR CM SLICE EXCEEDED

1. QUEUE PRIORITY SET TO LOWER BOUND FOR INPUT OR ROLLOUT FOR THE ORIGIN TYPE.

   JOB DOES NOT AUTOMATICALLY GIVE UP CM OR CPU WHEN SLICE EXPIRES.

2. ANY JOB OF HIGHER PRIORITY WILL FORCE JOB OUT OF CPU OR OUT OF CM (ROLLED OUT)

3. JOB AGES IN ROLLOUT QUEUE UNTIL IT BECOMES A CANDIDATE FOR SCHEDULING. AGING WILL OCCUR ONLY IF JOB'S ROLLOUT QP IS WITHIN LP→UP RANGE.

4. WHEN ROLLED BACK IN, JOB IS GIVEN A QP EQUAL TO HIGHER OF THE OP AND UP AND GIVEN FRESH CM AND CPU TIME SLICES.

AUTOROLL.

ENABLES/DISABLES THE AUTOMATIC ROLLOUT OF JOBS SO JOB MAY BE SCHEDULED FROM INPUT OR ROLLOUT QUEUE.

PRIORITY.

ENABLES/DISABLES THE AGING OF THE QUEUE PRIORITY.

INPUT FILE

LOCAL BATCH
REMOTE BATCH
SUBMIT
QUEUE LOADING

OVJ - VERIFY JOB/USER STATEMENT

- CRACK AND VALIDATE JOB CARD PARAMETERS
- INITIALIZE INPUT FILE SYSTEM SECTOR
- CRACK AND VALIDATE USER CARD PARAMETERS
- GENERATE JOB HASH
- RESULTS RETURNED FOR PROCESSING BY CALLING PROGRAM

MASS STORAGE ASSIGNMENT, SYSTEM SECTOR COMPLETED, AND FILE RELEASED TO INPUT QUEUE BY PROCESSING ROUTINE

DSP
QFM
VEJ
QAP
XSP

<table>
<tr><td colspan="5">JOB NAME</td><td>ORIGIN</td><td>INFT</td><td>1</td><td>0</td></tr>
<tr><td>ID</td><td>EQ</td><td>FIRST TRACK</td><td>ERROR CONTROL</td><td colspan="2">FL</td><td colspan="3">QP</td></tr>
</table>

USER INDEX HASH

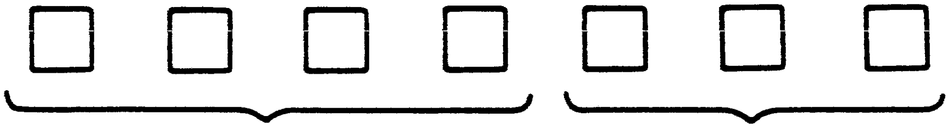
GBN - GENERATE BANNER NAME

ALGORITHM:

1. PAD USER INDEX WITH 3 ZERO BITS ON RIGHT
2. SPLIT INTO 4 GROUPS OF 5 BITS
3. ADD 1 TO EACH GROUP, GIVING DISPLAY CHARACTER

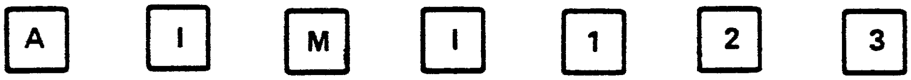
```
2061    =   00  000     010      000      110      001         000

            00000       01000          01100                 01000
            0+1=1       10+1=11        14+1=15               10+1=11

              A           I              M                     I

123456=         01     010     011     10      101     110      000

               10100          01110          01011       10000
               12+1=13        16+1=17        13+1=4      20+1=21

                 K              O              L            Q
```

JOB NAME



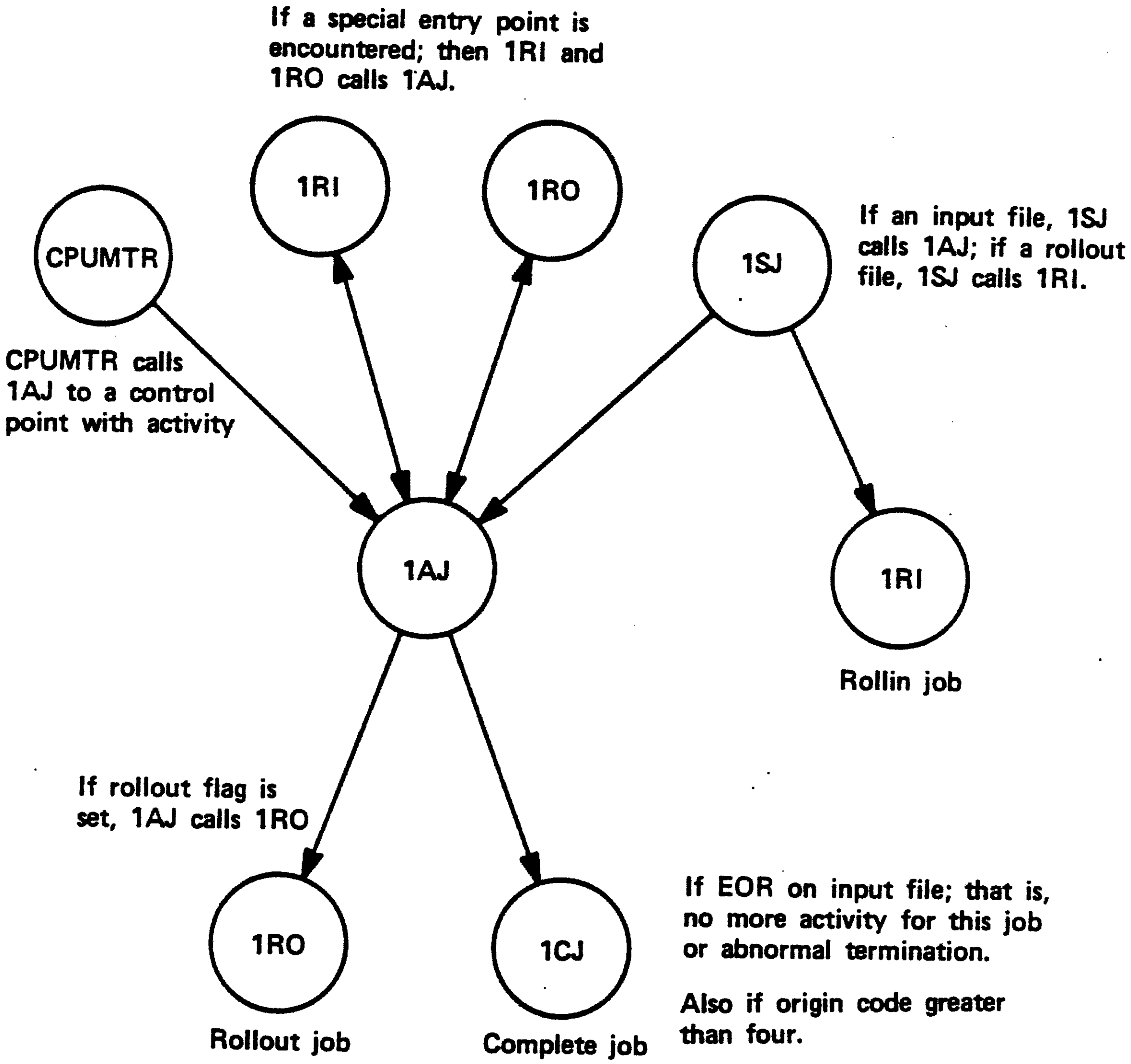
USER INDEX
HASH

1. IF TIME-SHARING JOB, TERMINAL NUMBER (OCTAL) IN DISPLAY CODE
2. NOT-TIME-SHARING JOB, LOWER 3 LETTERS OF JOB SEQUENCE NUMBER (RJSM)



USER: TURKEY, TROT (UI= 2061)

TERMINAL: 123,TTY


If a special entry point is encountered; then 1RI and 1RO calls 1AJ.
1RI
1RO
CPUMTR
1SJ
If an input file, 1SJ calls 1AJ; if a rollout file, 1SJ calls 1RI.
CPUMTR calls 1AJ to a control point with activity
1AJ
1RI
Rollin job
If rollout flag is set, 1AJ calls 1RO
1RO
Rollout job
1CJ
Complete job
If EOR on input file; that is, no more activity for this job or abnormal termination.
Also if origin code greater than four.

JOB BEGINS EXECUTION (1AJ/3AA)

1. CLEAR CONTROL POINT AREA FROM CPCW TO END, EXCEPT FOR ECSW.
2. SET DEFAULT SRU AND TIME LIMITS IN CPTW AND STLW.
3. READ JOB CONTROL STATEMENTS INTO CSBW AREA AND SET CSPW AND CSSW. (INPUT FILE POSITIONED AT EOR).
4. SET TRACK/SECTOR INFORMATION INTO INPUT FILE FST.
5. SET INPUT FILE FST ADDRESS INTO TFSW.
6. SET CONTROL POINT AREA FIELDS:

KEYPUNCH MODE IN SNSW
JOB SEQUENCE NUMBER (FROM SS) IN RFCW
EXIT MODE (EEMC) INTO XP
PERMANENT FILE DEFAULT FAMILY IN PFCW
    ADVANCE FAMILY ACTIVITY COUNT
FIELD LENGTH CONTROL IN FLCW AND ELCW
DEFAULT VALIDATION IN ALMW, ACLW, AACW
VALIDATION REQUIRED IN UIDW
SRU DEFAULTS VIA ABBF/ACTM
TIME LIMIT (FROM SS) INTO STLW
ISSUE JOB CARD TO DAYFILE
ISSUE CARDS READ ACCOUNTING MESSAGE (UCCR)

JOB BEGINS EXECUTION (1AJ/3AA)

7. CHECKS ERRORS

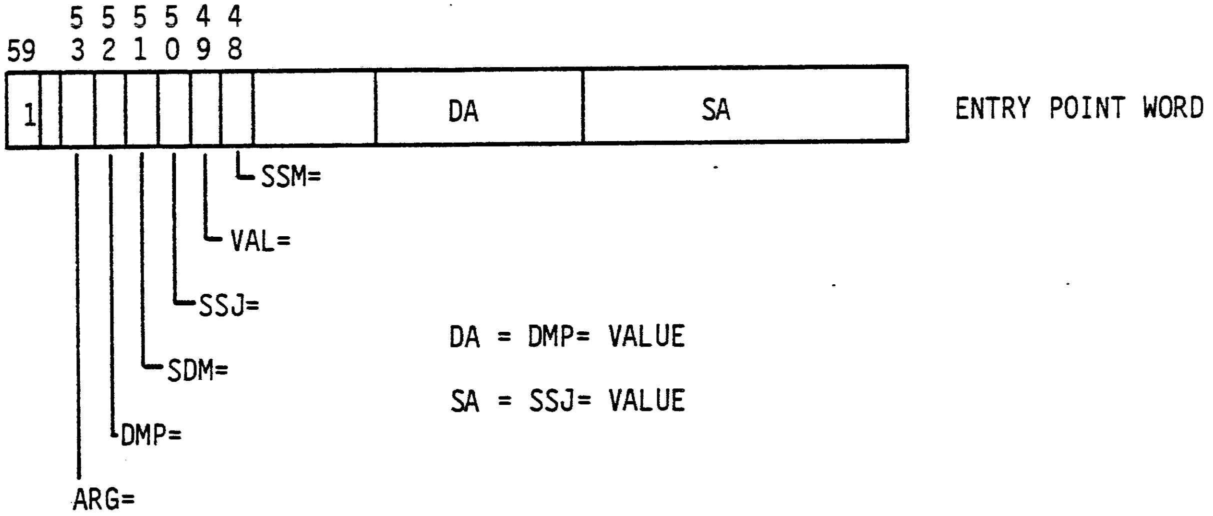
   - JOB CARD ERROR
   - JOB IN NORERUN STATE ON RECOVERY
   - BINARY CARD SEQUENCE ERROR

8. ADVANCE TO NEXT JOB STEP

   EXECUTE TCS

SPECIAL ENTRY POINTS

- ACCESS TO PRIVILEGED INFORMATION
- PERFORM SPECIAL OPERATIONS
- GIVE PROGRAM SPECIAL CONSIDERATION
- 3 CHARACTERS FOLLOWED BY = SIGN
- ASSEMBLED WITH ENTRY PSEUDO
- SYSEDIT TRANSLATES INTO FLAGS OR VALUES IN CLD ENTRY



DA = DMP= VALUE

SA = SSJ= VALUE

- ARG=
  DMP=
  MFL=
  RFL=
  SDM=
  SSJ=
  SSM=
  VAL=

TRANSLATE CONTROL STATEMENT (TCS)

1. READ STATEMENT
   - CONTROL STATEMENT BUFFER
   - MS1W (DIS)
   - JOB'S FL (TCS RA+1 CALL)
2. FIND KEYWORD
3. SEARCH FOR PROGRAM
   - INTERNAL TO TCS CTIME/RTIME/STIME/HTIME
   - LOCAL FILE ($ = IGNORE)
   - CLD
   - PLD

VAL=

4. MANAGE MEMORY

RFL=

MFL=

5. LOAD PROGRAM
   - ABS - DIRECTLY INTO FL
   - RELOCATABLE - CYBER LOADER
6. ISSUE STATEMENT TO DAYFILE

SDM=

7. UPDATE CSPW (IF NECESSARY)
8. ADVANCE JOB (JACM)
   DROP PP (DPPM) (RA+1 CALL)

PROGRAM LOADING

- ABSOLUTES AND OVERLAYS
- RELOCATABLES

1. PROGRAM NAME (KEYWORD) IDENTIFIED
2. STORAGE REQUESTED
3. LOAD PROGRAM (LDR/3AE)
   - ABSOLUTES OR OVERLAYS
     - IF NO MASS STORAGE, THE PROGRAM IS READ FROM SYSTEM LIBRARY OR LOCAL FILE DIRECTLY INTO THE CONTROL POINT FIELD LENGTH. THIS LOAD IS CHARGED THE NUMBER OF SECTORS TRANSFERRED AND A DEFAULT LOAD CHARGE.
     - IF IN CM OR ON ECS (ECS IS AN ASR), THE PROGRAM IS TRANSFERRED INTO THE CONTROL POINT FIELD LENGTH BY THE LCEM MONITOR FUNCTION.
     - IF SYSEDIT IS ACTIVE, THE PROGRAM WILL BE LOADED FROM MASS STORAGE.
     - IF AN ECS ERROR OCCURS, THE ASR RESIDENCY FOR THE ROUTINE IS CLEARED AND ANOTHER SOURCE (MASS STORAGE) IS USED TO LOAD THE PROGRAM.

PROGRAM LOADING

- RELOCATABLES
    - IF THE PROGRAM IS A RELOCATABLE, THE CYBER LOADER (AN ABSOLUTE) IS LOADED. THE CYBER LOADER WILL THEN LOAD THE RELOCATABLE.

4. EXCHANGE PACKAGE IS SET UP.

5. PRIVATE FILES RETURNED (PRIVACY FLAG CLEARED)

6. ARGUMENTS PASSED IN RA+ARGR THRU RA+63

ARG =

7. CLEAR MEMORY CSTM

8. START JOB STEP RLJS/RLMM

ARGUMENT PROCESSING

PRODUCT SET	OPERATING SYSTEM
● LOACL FILES	● CLD DEFAULT
● CLD WITH *SC	● LOCAL FILES WITH "/" PREFIX

PARAMETER	ID

ID=

1	,
2	=
3	≠
4	(
5	+
6	-
7	SPACE
10	;
17	TERMINATOR:) OR.

ID= SEPARATOR CHARACTER

= 0 IF TERMINATOR OR COMMA

ARGUMENTS ARE PROCESSED ONLY IF ARG= IS NOT SPECIFIED.

TCS MACROS

USER PROGRAM MAY CALL TCS DIRECTLY VIA A RA+1 REQUEST

THE MACROS

CONTROL
EXCST

ALLOW THE USER TO READ THE NEXT CONTROL STATEMENT FROM THE CONTROL CARD BUFFER AND ALLOW THE USER TO GIVE THE SYSTEM A CONTROL STATEMENT TO BE EXECUTED AS THE NEXT STATEMENT TO BE PROCESSED.

THIS ALLOWS A USER PROGRAM TO "READ AHEAD" DOING ALL SUBSEQUENT CONTROL STATEMENTS IT CAN PROCESS.

THIS ALSO ALLOWS A USER PROGRAM TO EXECUTE ANOTHER ROUTINE BEFORE REENTERING THE CONTROL CARD STREAM.

FIELD LENGTH MANAGEMENT

SYSMAX	MAXIMUM FL CURRENTLY AVAILABLE FOR ANY JOB (BY ORIGIN) BCOT/EIOT/SYOT: SYSMAX = MACHINE SIZE - CMR SIZE - K TXOT/MTOT SYSMAX = MACHINE SIZE - CMR SIZE - TK TK = K + TIME SHARING EXECUTIVE FL
MAXFL	MAXIMUM FIELD LENGTH THE JOB MAY EVER ATTAIN (BY ORIGIN) MIN $\{$JOB CARD CM, VALIDATION, SYSMAX, SERVICE$\}$
MFL	CURRENT JOB STEP MAXIMUM FIELD LENGTH
RFL	CURRENT FIELD LENGTH (RUNNING FIELD LENGTH)
NFL	NOMINAL FIELD LENGTH ≡ RFL
SYSDEF	DEFAULT INITIAL FIELD LENGTH IF NO OTHER FL CAN BE DETERMINED AND RFL IS 0

FIELD LENGTH MANAGEMENT

INITIAL FIELD LENGTH IS THE FIRST ONE OF THE FOLLOWING THAT APPLIES:

1. FL REQUIRED SPECIFIED BY RFL = /MFL=
2. ROUTINE HAS REQUIRED FL IN A LOADER (54) TABLE
3. RFL HAS BEEN SPECIFIED BY A RFL MACRO OR CONTROL STATEMENT
4. USE THE SMALLER OF MFL AND SYSDEF

AFTER PROCESSING HAS BEGUN, ADDITIONAL FL MAY BE ACQUIRED VIA MEMORY MACRO, UP TO MFL.

$$\text{RFL} \leq \text{MFL} \leq \text{MAXFL}$$

MFL	RFL	MAXFL	ROLLIN FL	FLINCR. REQUEST

FL_{C_W}

FIELD LENGTH MANAGEMENT

ECS FIELD LENGTH IS THE FIRST ONE OF THE FOLLOWING THAT APPLIES:

1. ROUTINE HAS REQUIRED ECS FL IN A LOADER (54) TABLE
2. ECS FL HAS BEEN SPECIFIED BY A RFL MACRO OR CONTROL STATEMENT

AFTER PROCESSING HAS BEGUN, ADDITIONAL ECS FL MAY BE ACQUIRED VIA MEMORY MACRO, UP TO MFL.

$$RFL \leq MFL \leq MAXFL$$

MFL	RFL	MAXFL	ROLLIN FL	FLINCR. REQUEST

$^{E}LC_{W}$

NOTE: ECS FIELD LENGTH IS NOT ROLLED BETWEEN JOB STEPS UNLESS PROTECTED VIA PROTECT MACRO/STATEMENT.

FIELD LENGTH MANAGEMENT

- RFL = MFL = ENTRY POINTS
- *FL LIBDECK SPECIFICATION

<table>
<tr><td>ABS</td><td colspan="3">NAME OF FIRST ENTRY</td><td></td><td>REL</td><td>SC</td><td></td><td>ASR</td><td>NO. EPTS</td></tr>
<tr><td>CLD</td><td>FLC</td><td>RCL/ASR</td><td colspan="5">TRACK</td><td colspan="2">SECTOR</td></tr>
<tr><td>ENTRY</td><td colspan="9">ENTRY POINT 1</td></tr>
<tr><td></td><td colspan="9">⋮</td></tr>
<tr><td></td><td colspan="9">ENTRY POINT N-1</td></tr>
</table>

FLC

<table>
<tr><td>11</td><td>10</td><td>9</td><td>0</td></tr>
<tr><td>R/M</td><td>MFC</td><td colspan="2">FL</td></tr>
</table>

R/M = RFL= /MFL= FLAG. IF R/M=0, THEN
RFL= VALUE IS ASSUMED.

MFC = MFL= CONTROL FLAG. IF MFC=0, THEN REQUIRED FL IS THE MAXIMUM OF EXISTING FL (FROM STSW) AND FL. IF MFC=1, THEN REQUIRED FL IS THE MAXIMUM OF THE CURRENT RFL (FROM FLCW) AND FL.

FIELD LENGTH MANAGEMENT

- RFL = ENTRY POINT

```
          ENTRY RFL =
            .
            .
            .
    RFL = EQU    *
```

THE RFL= VALUE IS ROUNDED UP TO THE NEXT MULTIPLE OF 100_8 AND SET INTO THE CLD ENTRY FOR THE ROUTINE BY SYSEDIT. WHEN THIS ROUTINE IS TO BE LOADED, THE RFL= VALUE WILL BE USED AS THE RUNNING FL (STSW).

- MFL= ENTRY POINT

THE MFL= VALUE IS ROUNDED UP TO THE NEXT MULTIPLE OF 100_8 AND SET INTO THE CLD ENTRY WITH R/M = 1 BY SYSEDIT.

WHEN THIS ROUTINE IS TO BE LOADED, IF THE MFL= VALUE IS $<$ 2000 (MFC=0), THE MAXIMUM OF RFL (FROM FLCW) AND THE MFL VALUE IS USED AS THE RUNNING FL (STSW). OTHERWISE, THE MAXIMUM OF THE RUNNING FL (STSW) AND THE MFL VALUE IS USED AS THE RUNNING FL.

FIELD LENGTH MANAGEMENT

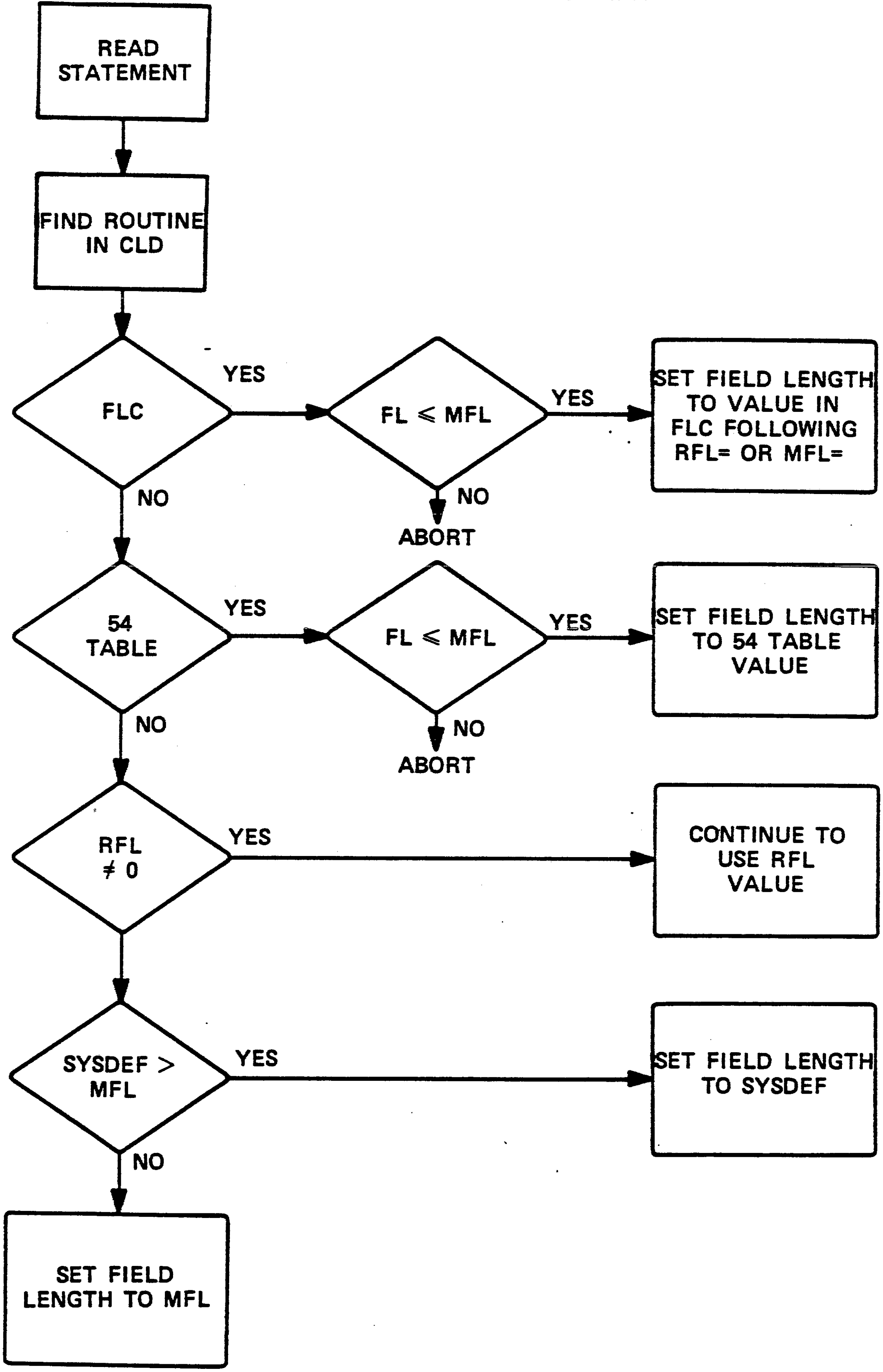
- *FL   LIBDECK DIRECTIVE

```
*FL, TY/REC-FLC
*FL, ABS/FTN-64LO
```

THE *FL DIRECTIVE ALLOWS THE FLC VALUE TO BE SPECIFIED BY SYSEDIT.

THE *FL VALUE IS PRIMARILY USED FOR PRODUCTS THAT DO NOT HAVE RFL= OR MFL= ENTRY POINTS TO OPTIMIZE THEIR MEMORY MANAGEMENT.

FIELD LENGTH MANAGEMENT



COMPLETE JOB

1CJ - JOB TERMINATION

- RELEASE STORAGE (RSTM FL=0)
- RELEASE FILES
    - DROP FILES (ODF) EXCEPT QUEUE TYPES
- RELEASE EQUIPMENT (FROM EST)
- UPDATE DEMAND FILE (ORF)
- RECORD RUNNING DATA
    - UPDATE PROFILA (OAU)
    - ISSUE ACCOUNTING SUMMARY

AESR	UECP	UEAD
UEMS	UEMT	UEPF

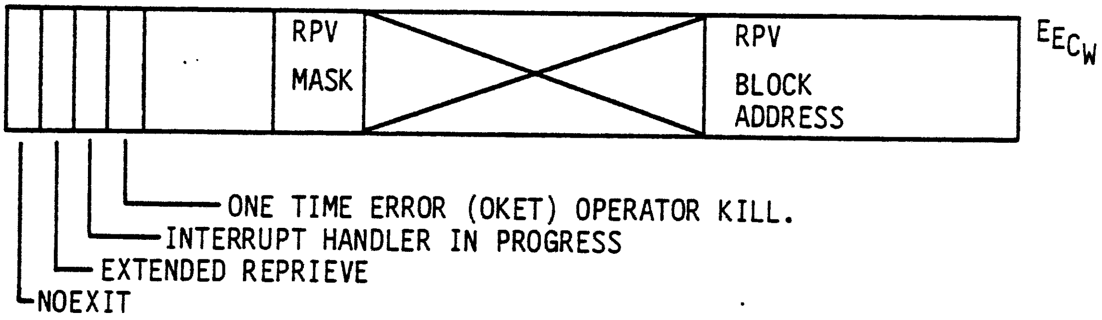
- DISPOSE OUTPUT TO QUEUE
    - COPY JOB DAYFILE TO PRINT FILE
    - ENTER DISPOSAL INFORMATION AND ROUTING DATA IN FNT/FST AND SYSTEM SECTOR
    - QUEUE FILE
- DROP INPUT FILE TRACKS (QPROTECT)
- DROP JOB DAYFILE TRACKS
- DECREMENT FAMILY ACTIVITY COUNT
- CLEAR FNT INTERLOCK
- CLEAR CONTROL POINT AND DROP PPU VIA JACM SUBFUNCTION 3.

ERROR PROCESSING

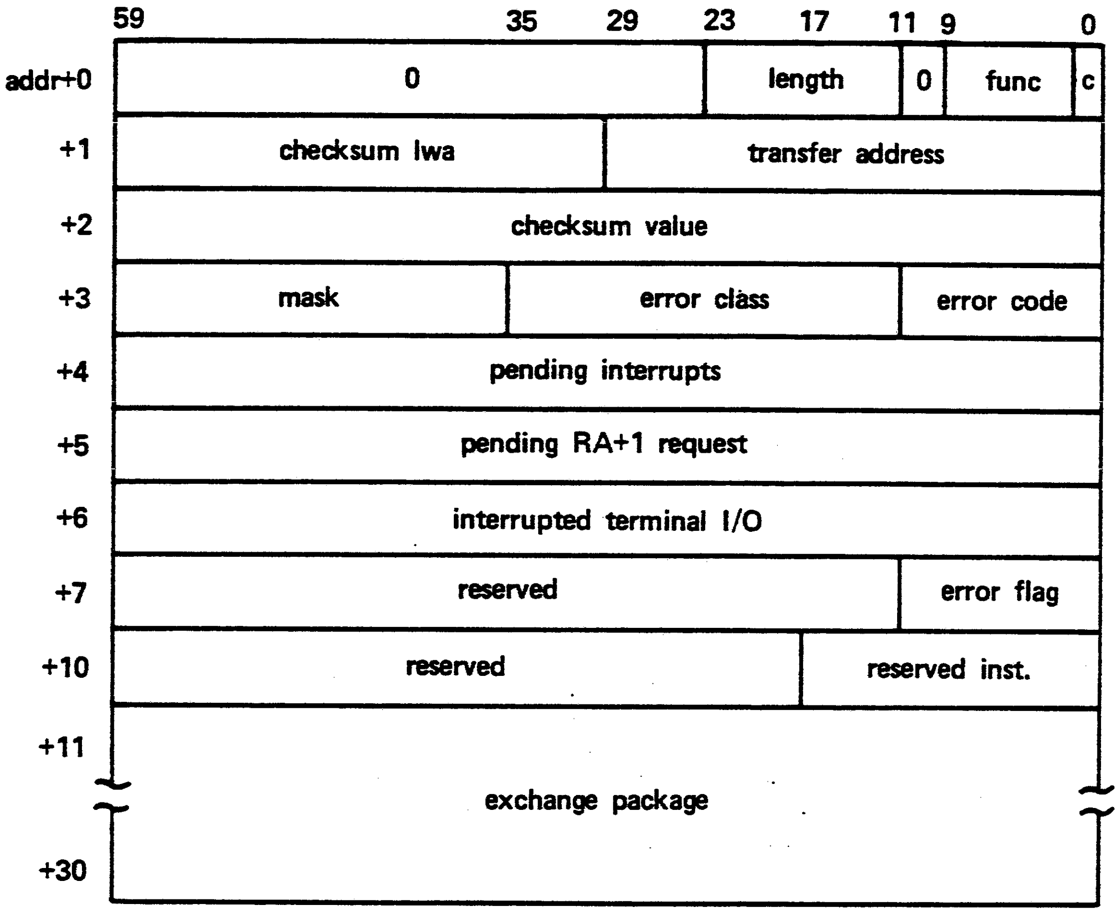
ON ENTRY TO 1AJ, OVERLAY 3AB (PROCESS ERROR FLAG) WILL BE CALLED IF:

- ERROR FLAG IS SET IN STSW
- ERROR EXIT OR REPRIEVE CONTROL IN EECW
    - EXTENDED REPRIEVE
    - EREXIT/REPRIEVE
    - EXIT. STATEMENT (NOEXIT.)
    - ABORT JOB

EXTENDED REPRIEVE



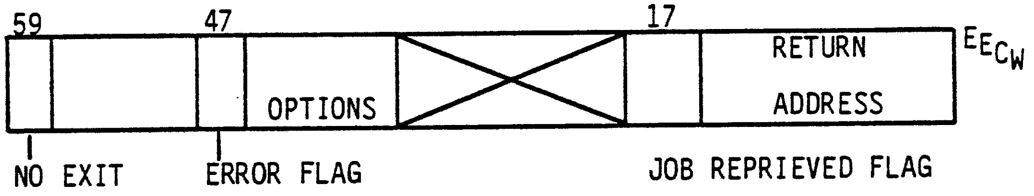
- VALIDATE PARAMETERS AND CHECKSUM
- INTERRUPT HANDLER ACTIVE AND BEING ABORTED, PERFORM EXIT PROCESSING
- INTERRUPT HANDLER NOT ACTIVE,
  - WRITE P, EXCHANGE PACKAGE, ERROR CODES, (RA+1) TO REPRIEVE BLOCK
  - CLEAR RA+1
  - SET INTERRUPT PROCESSING ACTIVE (INTERRUPT HANDLER)
  - RESTART JOB
- INTERRUPT HANDLER ACTIVE,
  - SET PENDING INTERRUPT CONTROL IN REPRIEVE BLOCK
  - RESTART JOB
- RESTART JOB
  - UPDATE EECW
  - INCREMENT TL OR SL
  - CLEAR DUMP CONTROL
  - REQUEST CPU (RCPM)
  - DROP PPU AND CLEAR JOB ADVANCE



length	Length of the parameter block including the exchange package area [minimum of 25 (31B) words].
func	Function code: 1 Setup 2 Resume 3 Reset

ERROR PROCESSING

EREXIT/REPRIEVE



- VALIDATE ADDRESS
- IF REPRIEVE
    - VALIDATE OPTIONS AND CHECKSUM
    - WRITE EXCHANGE PACKAGE, ERROR CODES, (RA+1) TO REPRIEVE BLOCK
    - ISSUE JOB REPRIEVED MESSAGE
    - SET P TO REPRIEVE ADDRESS
    - CLEAR RA+1
    - SET JOB REPRIEVED FLAG
    - RESTART JOB
- IF EREXIT
    - SET (RA) WITH P AND ERROR CODE
    - SET P TO ERROR EXIT ADDRESS
    - CLEAR RA+1
    - CLEAR ERROR EXIT ADDRESS FROM EECW
    - RESTART JOB
- RESTART JOB
    - UPDATE EECW
    - INCREMENT TL OR SL
    - CLEAR DUMP CONTROL
    - REQUEST CPU (RCPM)
    - DROP PPU AND CLEAR JOB ADVANCE

ERROR PROCESSING

EXIT

- IF ERROR < SPET, FIELD LENGTH PRESENT AND NON-MTOT JOB, THEN COMPLETE FILES
    - IF LIST OF FILES PRESENT (LOFW), COMPLETE FILES IN THE LOF LIST, FILES ARE FLUSHED IF
        - FET BUFFER HAS DATA AND WITHIN FL
        - WRITE BIT SET, OPEN ALTER, OR NO CIO CALLS
        - OUTPUT NOT FLUSHED IF TXOT
    - IF LIST OF FILES IS NOT PRESENT, FILE POINTERS FROM RA+2 TO RA+63 ARE EXAMINED FOR FET/FIT POINTERS. FLUSHES UNDER SAME CONDITIONS AS ABOVE FOR

        OUTPUT (IF NOT TXOT)

        PUNCH

        P8

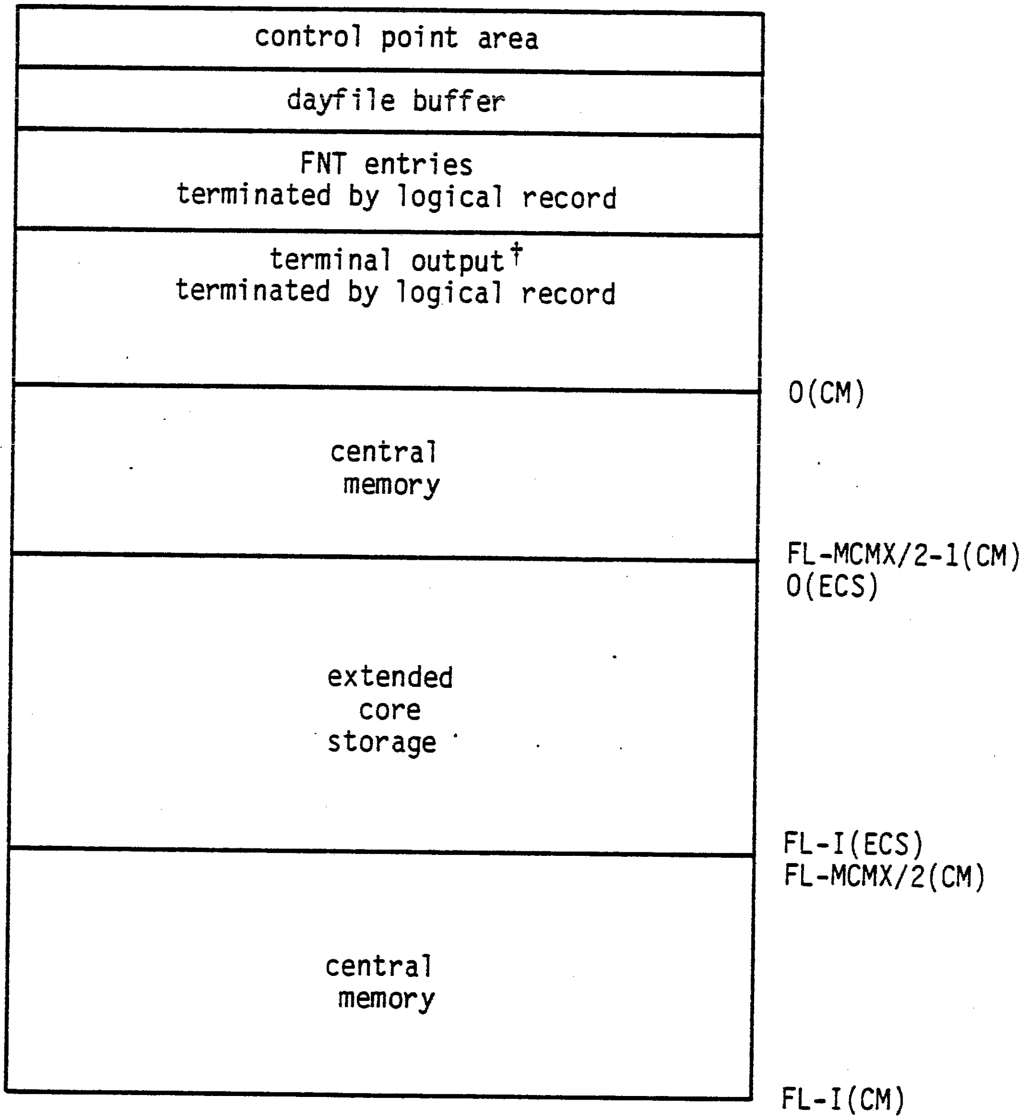
        PUNCHB
    - FLUSHING IS DONE BY A WRITER CIO REQUEST
- SET SKIP FOR EXIT. STATEMENT (UNLESS NOEXIT) AND SET ERROR FLAG IN JCRW
- PROCESS SPECIFIC ERROR AND ISSUE APPROPRIATE DIAGNOSTIC
- REQUEST DUMP VIA SPCW
- AFTER DUMP, JOB ADVANCES TO EXIT. STATEMENT OR TO 1CJ (COMPLETE JOB).

ROLLOUT FILE

System Sector

000 . . . 007	▨
010 011	dayfile buffer pointer
012 013	input file FNT entry
014 . . 027	list of equipment assigned to job (terminated by zero word)
030 . . 037	SSJ=parameter block
040 057	terminal table contents at last rollout
060 077	terminal table contents for recovery

File Format



---

†This part of the rollout file is used only for TXOT jobs.

DMP=

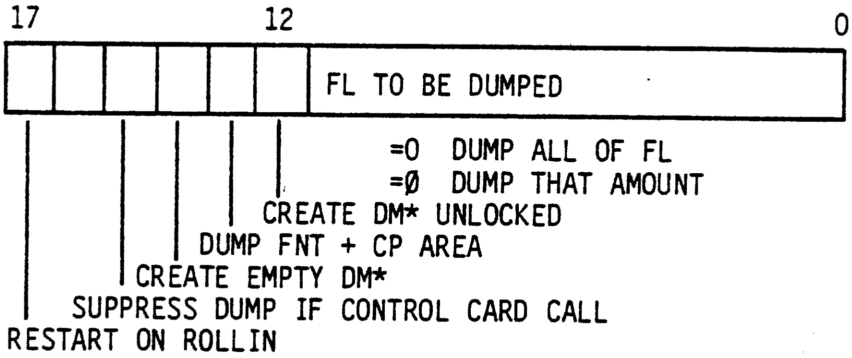
WHY

1. USE CONTROL POINT (IE CPU) FOR CPU TYPE OPERATION
2. SAVE FIELD LENGTH FOR PROCESSING

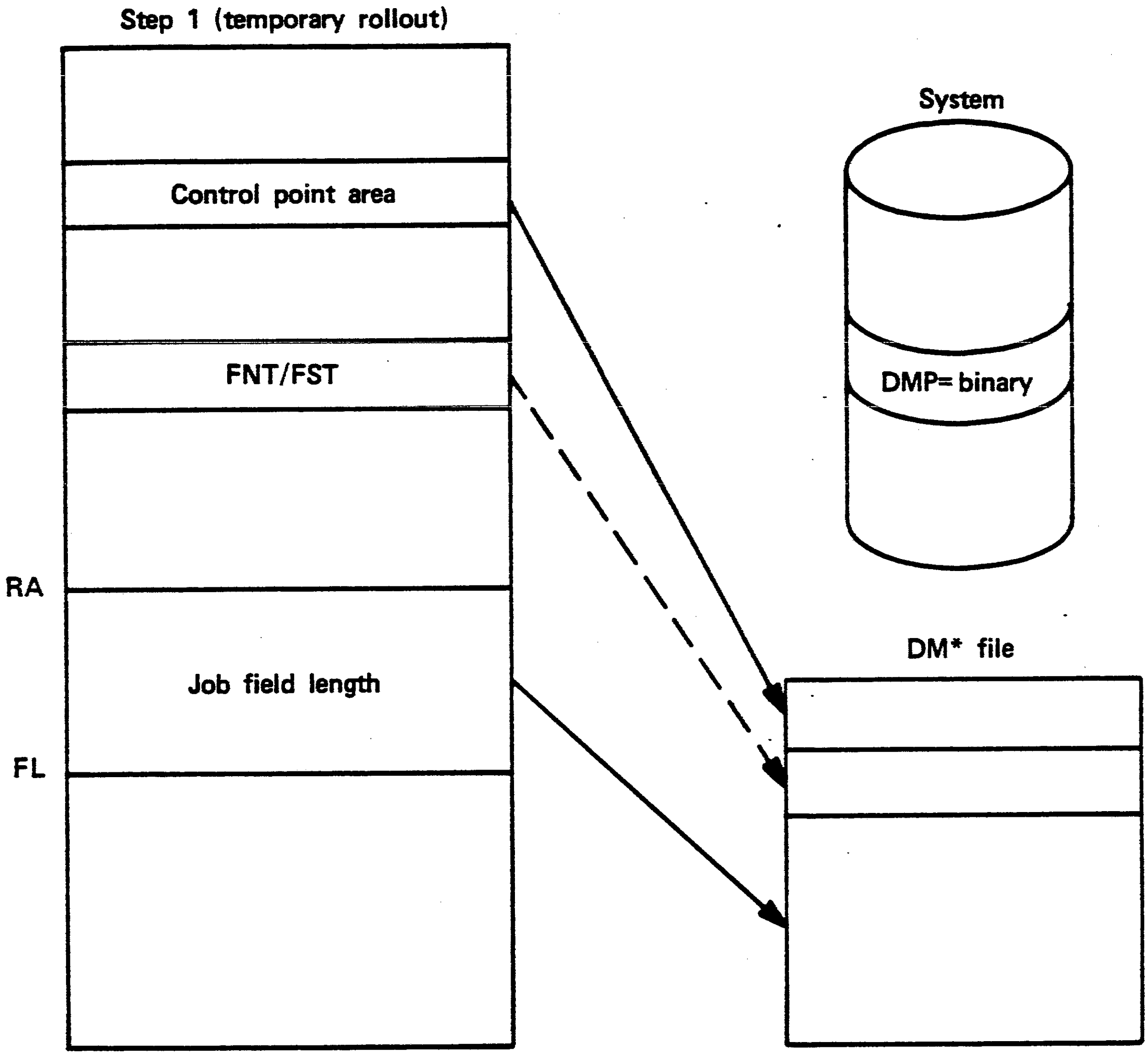
HOW

1. WRITE FL AND PORTIONS OF CONTROL POINT AREA TO DM* ROFT FILE
2. LOAD PROGRAM WITH DMP= ENTRY POINT
3. EXECUTE (PROGRAM MAY MANIPULATE THE DM* FILE)
4. RESTORE ORIGINAL FL AND CONTROL POINT AREA FROM DM*
5. ADVANCE TO NEXT JOB STEP OR RESUME EXECUTION

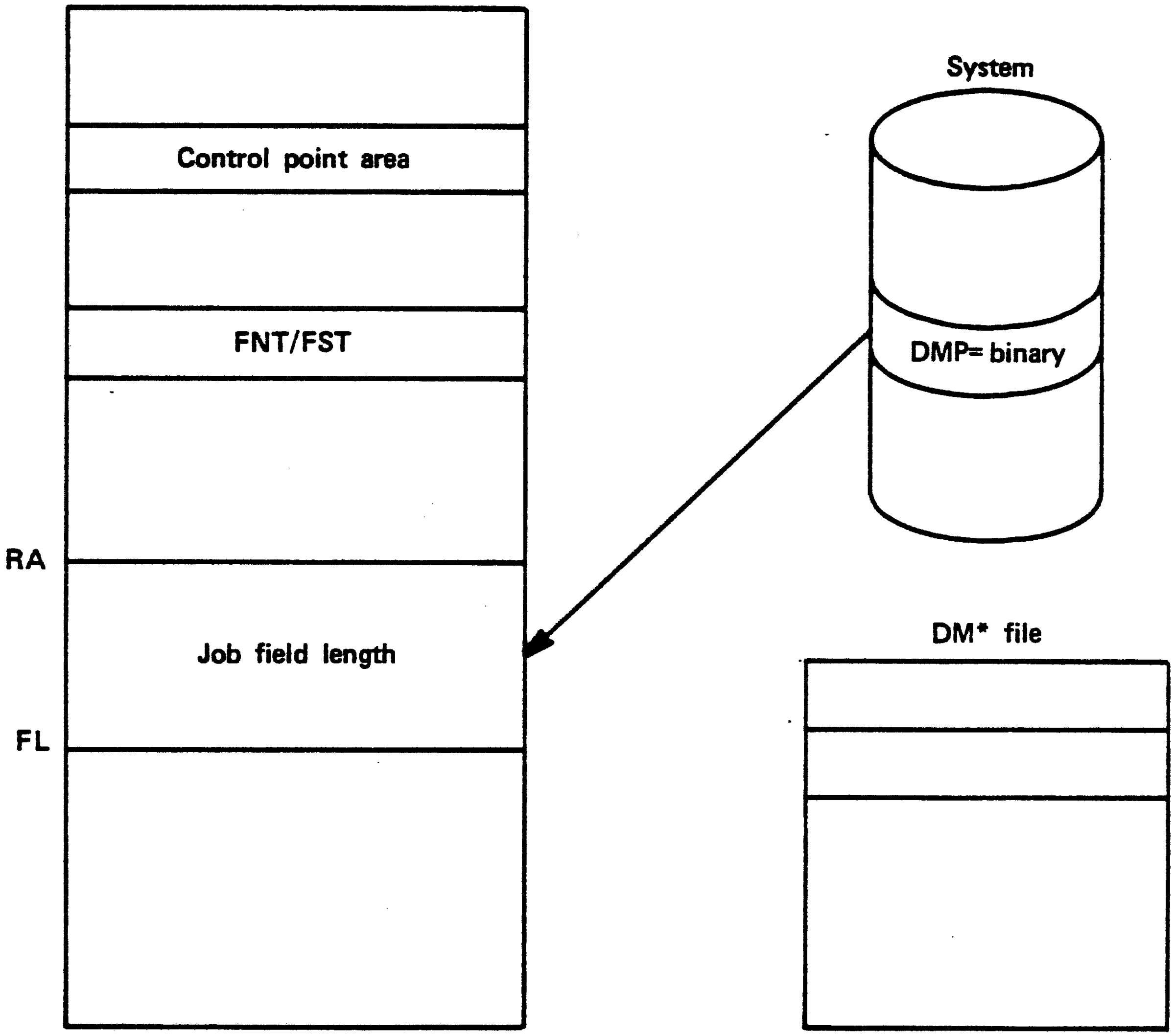
ENTRY POINT

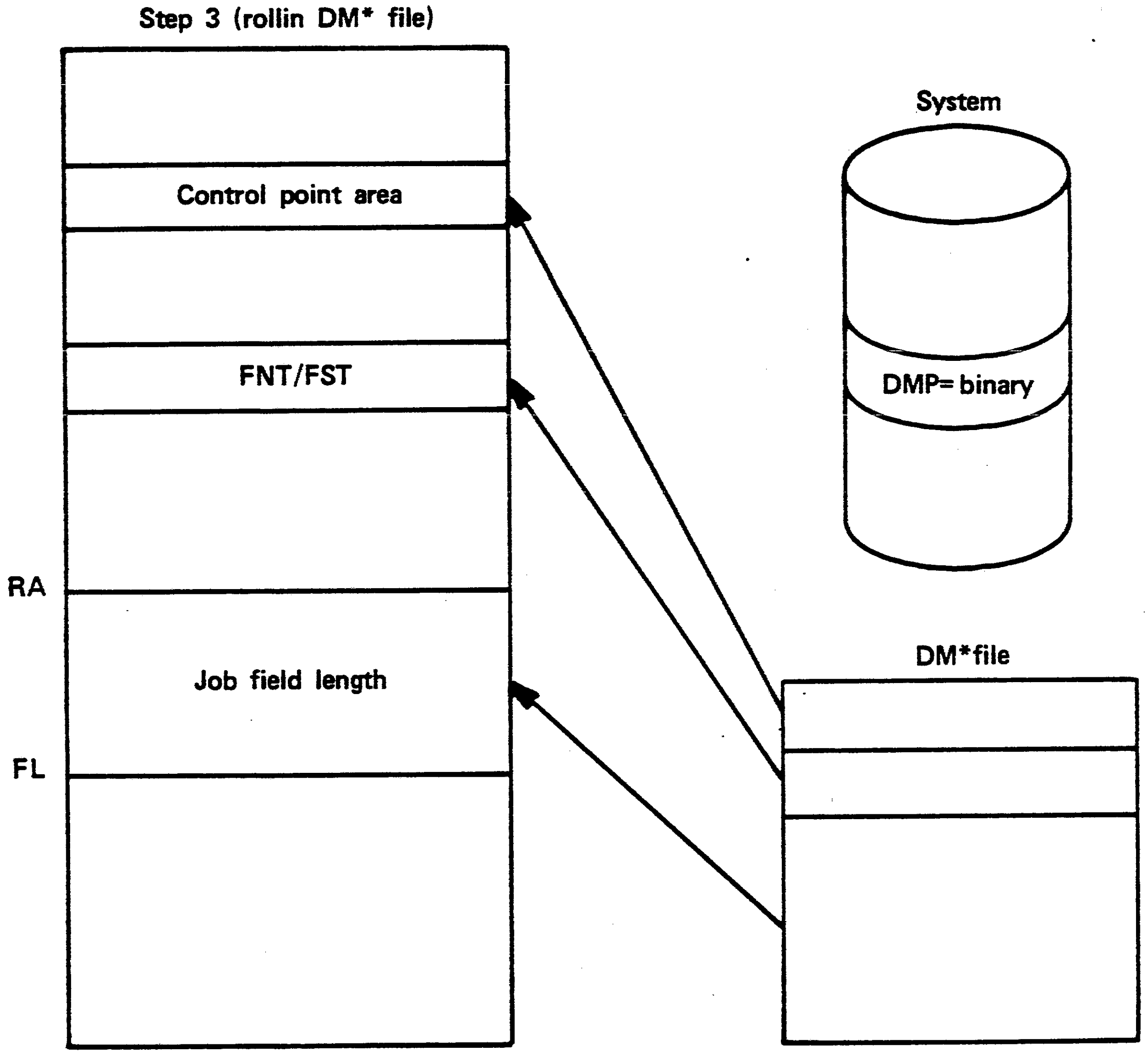


RESEX	100000	CHECKPOINT	010060
CPMEM	010051	RESTART	450000


Step 1 (temporary rollout)
Control point area
FNT/FST
RA
Job field length
FL
System
DMP= binary
DM* file

Step 2 (DMP= job load and execution)




Step 3 (rollin DM* file)
Control point area
FNT/FST
RA
Job field length
FL
System
DMP= binary
DM*file

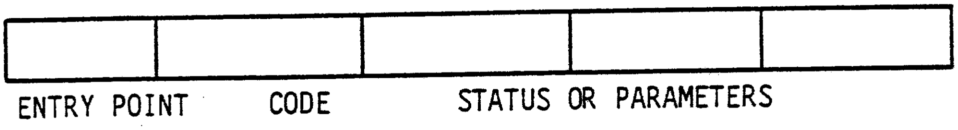
SPCW

SPECIAL CALL FOR DMP= PROCESSOR.

1. CP PROGRAM MAKES RA+1 REQUEST.
2. PPU ROUTINE SATISFYING REQUEST NEEDS A CPU ROUTINE TO ACTUALLY DO THE PROCESSING.
3. PPU ROUTINE PLACES REQUEST INTO CONTROL POINT WORD SPCW.
4. PPU ROUTINE DOES A ROCM - THIS WILL GET 1AJ TO PROCESS AN "ADVANCING" CONDITION.
5. 1AJ SENSES REQUEST IN SPCW AND SEARCHES CLD.
6. FL AND CP AREA ARE DUMPED TO DM* BASED ON DMP= VALUE BY 1RO.
7. 1AJ LOADS DMP= PROCESSOR.
8. PROCESSOR DOES ITS THING.
9. 1AJ REQUEST 1RI TO RESTORE FL AND CP AREA FROM DM* AS REQUIRED.
10. PPU ROUTINE COMES BACK TO COMPLETE REQUEST OR CPU ROUTINE THAT MADE THE RA+1 REQUEST RESUMES OR 1AJ ADVANCES TO NEXT JOB STEP.

DMP=

SPCW FORMAT



CODE 1/A, 1/B, 1/C, 1/D, 1/E, 1/0

* A REQUEST ACTIVE

B CLEAR RA+1 BEFORE RESUMING IF NOT SET

C REMAINDER OF WORD IS PARAMETER LIST, NOT ADDRESS OF PARAMETERS

* D DO NOT RESTART CPU

* E DMP= IN PROGRESS

* SET BY 1AJ

THE SPCW REQUEST WILL BE PASSED ON TO THE DMP= PROCESSOR (WHEN IT IS LOADED) AT ADDRESS SPPR (27) OF FIELD LENGTH.

WHEN PROGRAM TERMINATES SPPR IS USED TO PASS STATUS BACK THROUGH SPCW.

IF A PARAMETER ADDRESS IS SPECIFIED, A BLOCK OF DATA (20 WORDS) WILL BE TRANSFERRED FROM THE ORIGINAL PROGRAM TO THE DMP= PROCESSOR AT ADDRESS SPPR+1.

WHEN DMP= PROGRAM TERMINATES, THIS BLOCK IS RESTORED INTO THE ORIGINAL PROGRAM AT THE PARAMETER ADDRESS

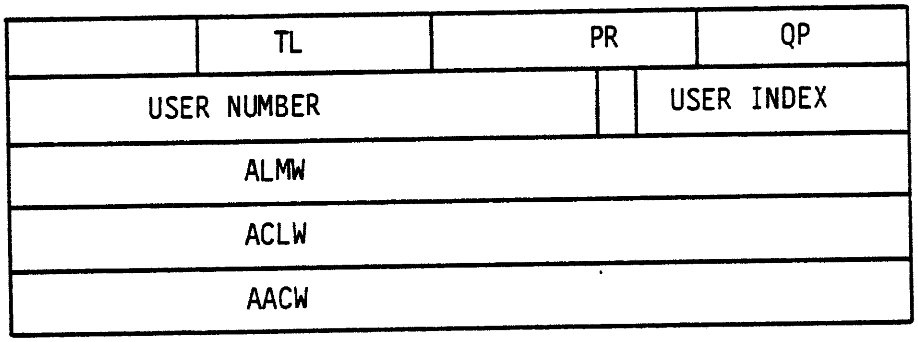
SSJ=

WHY

1. PASS VALIDATION INFORMATION FROM CONTROL POINT AREA TO PROGRAM.
2. PASS VALIDATION INFORMATION TO CONTROL POINT AREA FROM PROGRAM.
3. ALLOW PROGRAM TO HAVE CERTAIN PRIVILEGES.
4. ALLOW PROGRAM TO HAVE TL, PR, QP FOR ONLY ITSELF, NOT THE ENTIRE JOB.

HOW

SSJ= VALUE AND PARAMETER BLOCK



	400000	4AAAAA	0AAAAA	000000
DROP FILES SSID	Y	Y	Y	N
SWAP BLOCK	N	Y	Y	N
SSJ PRIVILEGES	Y	Y	Y	Y
CREATE SSID FILES	N	Y	Y	N

SSJ=

1. IF PARAMETER BLOCK ADDRESS IS SPECIFIED, WHEN PROGRAM IS LOADED:

```
1AJ     SETS       TL, PR, QP    IF SPECIFIED

        RETURNS CURRENT TL, PR,  QP TO PROGRAM

                UIDW
                ALMW
                ACLW
                AACW

        SETS    UIDW TO SYSTEMX, 377777
                ALMW, ACLW, AACW UNLIMITED
```

2. WHEN PROGRAM TERMINATES, 1AJ

```
        RESETS  TL,PR, QP FROM BLOCK
        SETS    UIDW FROM BLOCK
                ALMW      (NEW OR RESTORED VALUES)
                ACLW
                AACW

        RETURNS ALL FILES WITH SSID
```

SUBSYSTEMS

CHARACTERISTICS

1. NOT ROLLABLE - NO ROLLOUT
2. INTER CONTROL POINT COMMUNICATIONS (RSB/SIC)
3. HIGH CPU PRIORITY
4. DOES NOT ABIDE BY JCB CONTROLS
   CM AND CPU SLICES - INFINITE
5. MAY HAVE A USER NUMBER IN UIDW
   TRANEX/TAF
6. MAY RESIDE AT A SPECIFIED CP
7. IMPLICIT SSJ= AND ARE SYOT
8. USE OF SPC (NEE TLX) RA+1 CALL

TO BE A SUBSYSTEM

1. UNIQUE QP DEFINED TO FIT INTO SSCL WORDS OF CNR
2. 1DS TABLE ENTRY TO START UP EXECUTIVE
3. QP IDENTIFIES THE SUBSYSTEM (OPERATING SYSTEM TESTS FOR A GIVEN SUBSYSTEM BY ITS QUEUE PRIORITY)

SUBSYSTEMS

SUBSYSTEM	QP SYMBOL	QP	PP INIT.	CONTROL PT.
TIME-SHARING	TXPS	7776	1SI	1
TELEX				
IAF				
REMOTE BATCH	EIPS	7775	1LS	77
EI200				
UNIT RECORD	BIPS	7774	1IO	76
BATCHIO				
TAPES	MTPS	7773	1MT	75
MAGNET				
TRANSACTION	TRPS	7772		
TRANEX			1TP	
TAF			1SI	
NETWORK INTERFACE	NMPS	7770	1SI	74
NIP				
REMOTE BATCH (NETWORK)	RBPS	7767	1SI	73
RBF				
MESSAGE CONTROL SYS	MCPS	7765	1SI	72
CDCS	CDPS	7766	1SI	71
TIME SHARE STIM	STPS	7771	1SI	77
STIMULA				
MASS STORAGE CONTROL	MSPS	7764	CMS	74
MASS STORAGE SUBSYSTEM	MFPS	7763	1SI	70
MSSEXEC				

SUBSYSTEMS

SCHEDULING

AUTO. MAINTENANCE	1DS FUNCTION 33
MAGNET. ETC.	1DS FUNCTION 32

1DS HAS A TALBE OF SUBSYSTEMS. ON FUNCTION 33 1DS CHECKS TO SEE IF SUBSYSTEM IS ENABLED AND IF SO AUTOMATICALLY INITIATES IT. (SSTL)

SSCL IS CHECKED TO DETERMINE IF SUBSYSTEM IS ALREADY ACTIVE. IF SO, 1DS 33/32 ACTION FOR THAT SUBSYSTEM IS IGNORED.

A FNT/FST ENTRY FOR THE SUBSYSTEM IS BUILT AND ENTERED INTO THE INPUT QUEUE.

1SI	CP	JOB SEQN NUMBER	S Y O T	I N F T	0	FNT

0	77	F	F	F	F	FL	QP	FST

SUBSYSTEMS ARE SCHEDULED BY 1SJ OVERLAY 3SA. SSCL WILL BE ENTERED WITH THE CP GIVEN TO THE SUBSYSTEM.

CP IS ACTUAL CP IF $< 40_8$; OTHERWISE NCP-$\overline{\text{CP}}$-1.

QUESTION SET LESSON 14

1. When and how is 1SJ called into execution?

2. How do queue priorities get aged? What routine does the aging?

3. What routine checks CPU and CM time slices? What happens to the job if either one of these slices expires?

4. Can you disable priority evaluation? Auto rollout? Job Scheduling?

5. Under what conditions might the job scheduler request that a job be rolled out?

6. What criteria does 1SJ use to determine the "best" job for scheduling?

7. What criteria does 1SJ use to determine the "best" control point to be assigned to the job?

8. Why does 1SJ call 1AJ and 1RI? What does 1SJ do if a PP is not available for 1AJ or 1RI?

9. Why does CPUMTR call 1AJ?

10. Explain what the Begin Job overlay (3AA) accomplishes. Why is 3AA called?

11. Explain what the Error Processing overlay (3AB) does and why it would be called.

12. Which 1AJ overlay is called to process control statements?

13. Which control statements are processed entirely within the 1AJ (TCS) PP?

14. Explain the differences between Operating System argument processing and Product Set argument processing.

15. How is a program loaded from the Resident Central Library? From an ECS ASR? From mass storage?

16. Are relocatables and absolute routines loaded in the same manner? Explain your answer.

17. Is it legal to call a PP routine from a control statement? If so, how does 1AJ process it?

18. Explain how 1DS causes subsystems to be initiated.

19. Define the terms:
    SYSMAX

    MAXFL

    MFL

    RFL

    SYSDEF

20. How does the system determine a program's initial central memory field length?

21. How does the system determine a program's initial ECS field length?

22. The CLD entry contains a field length control field. Explain the values this field may assume with respect to RFL=, MFL= and *FL values.

23. Your CPU program does a DMP RA+1 request. How does the system process this request? Your answer should discuss DMP= and SPCW processing.

24. Discuss the processing of the SSJ= entry point. What privileges can the program have? What happens to files created by SSJ= programs when they terminate?

LESSON 15
ACCOUNTING

LESSON PREVIEW:

This lesson presents the accounting and validation philosophy of NOS. The ACCOUNT dayfile and its entries, the System Resource Unit (SRU), the monitor functions and subroutines, and validation files will be discussed. The resource controls placed on the user by the user number and charge/project number will also be discussed.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Detail the standard message format that appears in the ACCOUNT dayfile.
- Explain what a System Resource Unit (SRU) is and give its formula.
- Briefly explain how each value in the SRU formula is computed and the SRU accumulated by CPUMTR.
- Define the term "account block".
- Describe the monitor functions ACTM and RLMM and describe the accounting aspects of monitor functions TIOM and UADM.
- Discuss the validation files VALIDUs and VALINDs and what information they contain.
- Explain the various internal representations for limit values.
- Discuss the project profile file PROFILa and what information it contains.

REFERENCES:

NOS IMS - Chapter 20 NOS SMRM, 5 NOS RM, 1-3-7, 1-6-2,33-34

ACCOUNTING

ACCOUNT DAYFILE

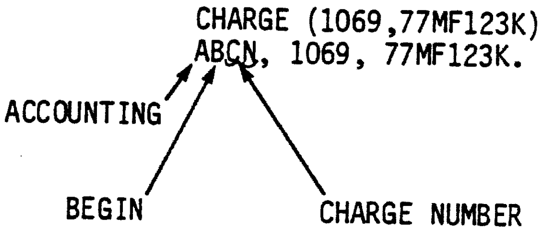
HISTORY OF SYSTEM AND RESOURCE USAGE TO -

- PROPERLY BILL USERS OF THE SYSTEM
- ANALYZE SYSTEM USAGE

STANDARD MESSAGE FORMAT

```
HH.MM.SS. JOBNAMEO, GEAC, INFORMATION.
```

G = GROUP {S = STATISTICAL, U = USAGE, A = ACCOUNTING}
E = EVENT
AC = ACTIVITY



```
CHARGE (1069,77MF123K)
ABCN, 1069, 77MF123K.
```

SRU

BASIC ACCOUNTING UNIT IS THE SYSTEM RESOURCE UNIT OR SRU. IT IS A MEASURE OF THE RESOURCES USED BY A JOB OR TTY SESSION.

- CM FIELD LENGTH
- ECS FIELD LENGTH
- CPU TIME
- MS USAGE
- PF USAGE
- MT USAGE

ACCOUNT BLOCK

SRUs FROM CHARGE TO ANOTHER CHARGE OR END OF JOB.

ACCOUNTING

SRU ALGORITHM

SRU = (M1(CP+M2·IO)+M3(CP+IO)CM+M4(CP+IO)EC)+AD

CP	CENTRAL PROCESSOR USAGE IN MILLIUNITS

CP = S0·CP0 + S1·CP1

S0,S1 ARE MULTIPLIERS TO NORMALIZE CP TIME IN A DUAL CPU MACHINE OR MAINFRAMES AT THE SITE.

CPi IS ACCUMULATED TIME IN CPUi IN MILLISECONDS.

IO	MEASURE OF ACCUMULATED I/O ACTIVITY (MILLIUNITS)

IO = S2·MS + S3·MT + S4·PF

S2,S3,S4 MULTIPLIERS WEIGHT MS,MT,PF AGAINST EACH OTHER

MS = MASS STORAGE ACTIVITY
MT = MAGNETIC TAPE ACTIVITY
PF = PERMANENT FILE ACTIVITY

CM	CENTRAL MEMORY FIELD LENGTH IN WORDS/100_8
EC	ECS FIELD LENGTH IN TRACKS (WORDS/1000_8)
AD	ADDER APPLIED AS AN ACCUMULATION OF INDIVIDUAL UNIT CHARGES
M1	OVERALL SRU SCALE MULTIPLIER
M2	MULTIPLIER TO WEIGHT IO AGAINST CP EC CM
M3	MULTIPLIER TO WEIGHT CM AGAINST CP EC IO
M4	MULTIPLIER TO WEIGHT EC AGAINST CP IO CM

ACCOUNTING

SRU UPDATING IN CPUMTR

AAD - APPLY ADDER
CALLED BY UADM

$SRU = SRU + AD$

AIO - APPLY IO
CALLED BY UADM
TIOM

$IO = S2 \cdot MS+S3 \cdot MT+S4 \cdot PF$
$SRU = SRU + IO \cdot IOM$

CPT - APPLY CP TIME
CALLED BY TIM
OR WHEN CONTROL POINT IS GIVEN OR GIVES UP THE CPU

$CP = S0 \cdot CP0 + S1 \cdot CP1$
$SRU = SRU + CP \cdot CPM$

SRU - CALCULATE IOM AND CPM MULTIPLIERS
CALLED WHEN CM/ECS FL CHANGES OR M1-M4
MULTIPLIERS CHANGE BY NEW ACCOUNT BLOCK

$$SRU = M1(CP+M2 \cdot IO+M3(CP+IO)CM+M4(CP+IO)EC) + AD$$

$$= M1(1+M3 \cdot CM+M4 \cdot EC)CP + M1(M2+M3 \cdot CM+M4 \cdot EC)IO + AD$$

$$= (M1+M1 \cdot M3 \cdot CM+M1 \cdot M4 \cdot EC)CP + (M1 \cdot M2+M1 \cdot M3 \cdot CM + M1 \cdot M4 \cdot EC)IO + AD$$

$$= CPM \cdot CP + IOM \cdot IO + AD$$

$$CPM = M1 + M1 \cdot M3 \cdot CM + M1 \cdot M4 \cdot EC$$

$$IOM = M1 \cdot M2 + M1 \cdot M3 \cdot CM + M1 \cdot M4 \cdot EC$$

ACCOUNTING

CPUMTR FUNCTIONS

ACTM - ACCOUNTING FUNCTIONS

SUBFUNCTIONS

ABBF	BEGIN ACCOUNT BLOCK	FIRST CHARGE OR DEFAULT MULTIPLIERS
ABSF	RESET MULTIPLIERS	IRI RESTARTS JOB •CPM •IOM
ABCF	CHANGE/END ACCOUNT BLOCK	SECOND CHARGE
ABEF	CALCULATE ELAPSED SRUs	IRO COMPLETES TTY JOB STEP
ABVF	CONVERT ACCUMULATORS	ACCOUNT BLOCK END MESSAGES
ABIF	MAKE SRU ACCUMULATOR INTEGER	INCREMENT SRU ACCUMULATOR IN PROFILE

RLMM - REQUEST LIMIT

SUBFUNCTIONS

RLCO	CLEAR OVERFLOW FLAGS
RLIT	INCREMENT JOB STEP TIME LIMIT
RLIS	INCREMENT JOB STEP SRU LIMIT
RLJS	START JOB STEP
RLTL	SET TIME LIMIT
RLSL	SET SRU LIMIT

ACCOUNTING

CPUMTR FUNCTIONS

TIOM - TAPE I/O PROCESSOR CALLS AIO TO ADD MT ACCUMULATOR AFTER INCREMENTING

UADM - UPDATE CONTROL POINT AREA

SUBFUNCTIONS

AISS	INCREMENT MS OR PF ACCUMULATOR	CALLS AIO
AIAD	INCREMENT AD ACCUMULATOR	CALLS AAD

MACROS

DISSR DISABLES SRU ACCUMULATION WHILE ALLOWING OTHER VALUES TO ACCUMULATE. A "ONE TIME CHARGE" MAY BE ADDED TO THE SRU WHEN USING DISSR. DISSR ALLOWS THE NORMALIZATION OF CERTAIN SYSTEM OPERATION, SUCH AS TAPE ASSIGNMENT, IN WHICH SYSTEM OVERHEAD VARIES DUE TO THE ENVIRONMENT.

RENSR ENABLES SRU ACCUMULATION AFTER A DISSR HAS BEEN ISSUED.

ACCOUNTING

VALIDATION FILES

SYSTEM VALIDATION (VALIDUs) AND PROJECT PROFILE (PROFILa) FILES ARE USED TO VALIDATE USER ACCESS TO SYSTEM.

- DETERMINE IF A USER IS ALLOWED TO USE THE SYSTEM.
- CHARGE THE USER FOR HIS RESOURCE USAGE.
- RESTRICT THE USER TO CERTAIN RESOURCE USAGE.
- MAINTAIN PERMANENT FILE CATALOGS FOR THE USER BY MAPPING THE USER'S USER NUMBER INTO A SPECIFIC USER INDEX.

VALIDATION FILES

VALIDUS - HAS USER VALIDATION INFORMATION.

VALINDS - HAS A BIT FOR EACH USER INDEX IN USE (4210_8 WORDS REPRESENT USER INDICES 1 THROUGH 377700 (AUIMX)).

MODVAL - MAINTENANCE UTILITY

VALIDUS IS A TREE STRUCTURED FILE.

TREE STRUCTURED FILES ARE MANAGED VIA COMMON DECK COMSSFS AND SYSTEM ROUTINE SFS - SPECIAL SYSTEM FILE SUPERVISOR.

EACH TREE NODE IS CALLED A LEVEL. TYPICALLY WE HAVE LEVEL-0, LEVEL-1, LEVEL-2 AND LEVEL-3 DATE BLOCKS IN THE VALIDUS AND PROFILA FILES.

VALIDUS LEVEL-0 CONTAINS HISTORY INFORMATION AND THE FIRST USER NUMBER AND RANDOM ADDRESS OF EACH PRIMARY LEVEL-1 BLOCK.
(CREATED ON OP=C)

VALIDUS LEVEL-1 BLOCKS CONTAIN THE USER NUMBER AND RANDOM ADDRESS OF THE LEVEL-2 BLOCKS.

VALIDUS LEVEL-2 BLOCKS CONTAIN THE USER VALIDATION INFORMATION FOR UP TO FOUR USER NUMBERS.

ACCOUNTING

VALIDATION FILES

VALIDATION BLOCK (LAYOUT BY COMSACC)

UN USER NUMBER	1-7 CHARACTERS (ALPHANUMERIC)
UI USER INDEX	SUPPLIED THROUGH VALINDS (377777)
PW PASSWORD	1-7 CHARACTERS (ALPHANUMERIC)
SC SECURITY COUNT	NUMBER OF SECURITY VIOLATIONS
AB ANSWERBACK	ANSWERBACK TERMINAL CODES
MT MAGNETIC TAPE	MAX, MAGNETIC TAPES ALLOWED
RP REMOVABLE PACKS	MAX, REMOVABLE PACKS ALLOWED
TL JOB STEP TIME LIMIT	MAX, SETTING FOR TIME LIMIT
SL JOB STEP SRU LIMIT	MAX, SETTING FOR SRU LIMIT
DF DAYFILE MESSAGES	NUMBER OF MESSAGES ALLOWED
CC CONTROL STATEMENT	NUMBER OF CONTROL STATEMENTS ALLOWED
CM CENTRAL MEMORY	MAX, CM FIELD LENGTH SETTING

EC	ECS	MAX, ECS FIELD LENGTH SETTING	
LP	LINES PRINTED	NUMBER OF LINES PRINTED ALLOWED	
CP	CARDS PUNCHED	NUMBER OF CARDS PUNCHED ALLOWED	
OF	DISPOSED OUTPUT	NUMBER OF OUTPUT QUEUE FILES ALLOWED	
DB	DEFERRED BATCH	NUMBER OF SUBMITS ALLOWED	
MS	MASS STORAGE	AMOUNT OF MASS STORAGE IN USE/JOB	
NF	NUMBER OF FILE	NUMBER OF LOCAL FILES PER JOB	
AW	ACCESS WORD	VARIETY OF INDIVIDUAL PERMISSIONS	
		CPWC (0)	CSRP (10)
		CTPC (1)	CSTP (11)
		CLPF (2)	CTIM (12)
		CSPF (3)	CUCP (13)
		CSOJ (4)	CSAP (14)
		CASF (5)	CBIO (15)
		CAND (6)	CPRT (16)
		CCNR (7)	
AP	APPLICATION	IAF (24)	NOP (29)
		RBF (25)	LOP (30)
		TAF (26)	NOPLOP (31)
		MCS (27)	
		TVF (28)	

ACCOUNTING

VALIDATION FILES

VALIDATION BLOCK

FC	PERM, FILE COUNT	NUMBER OF PFS ALLOWED
FS	IAFSIZE	MAX, SIZE OF IAF
CS	TOTAL IAF SIZE	COMMULATIVE IAF ALLOWED
DS	DAF SIZE	MAX, SIZE OF DAF
PX	TRANSMISSION	FULL OR HALF ECHOPLEX
RO	RUB OUTS	RUBOUT COUNT
PA	PARITY	TERMINAL PARITY (EVEN/ODD)
TT	TERMINAL TYPE	TYPE OF TERMINAL (EG. COR)
TC	CHARACTER SET	DEFAULT CSET (EG. ASCII)
IS	INITIAL SUBSYSTEM	DEFAULT SUBSYSTEM (EG. BASIC)

ACCOUNTING

VALIDATION FILE

LIMIT MANAGEMENT - CONVERT INDEX INTO LIMIT VALUE

- COMCCVI
- COMPCVI

```
YYI$     DEF 1
*CALL    COMxCVI
```

LIMIT INDEX

LIMIT = INDEX·MULTIPLIER+CONSTANT

TLI	LPI	NFI	ECI
SLI	CPI	CMI	DBI

INDEX TABLE

INDEX POINTS TO TABLE ENTRY HAVING LIMIT

FCI	DSI	FSI	CSI

COUNTING LIMIT

VALUE COMPUTED BY THE LIMIT INDEX SCHEME, STORED IN ACLW AND MANIPULATED VIA UADM

MSI	CCI	DFI	OFI

ACTUAL VALUES

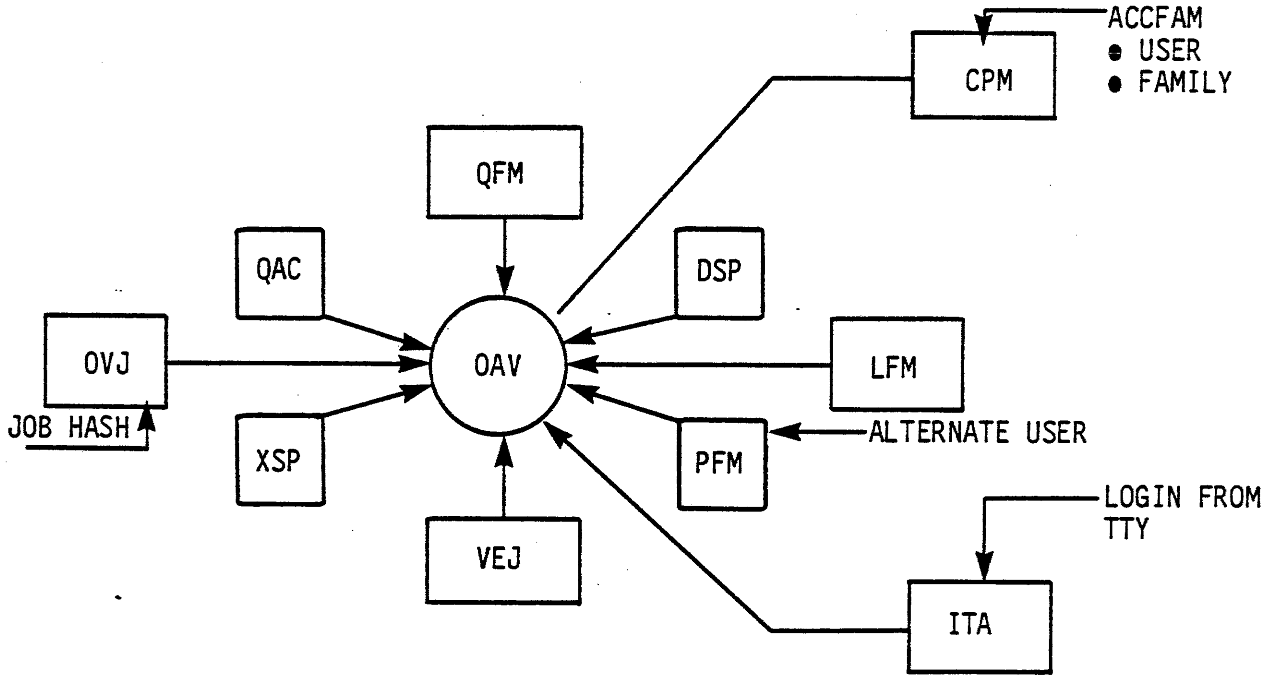
ACTUAL VALUE IS LIMIT - TAPES AND PACKS

ACCOUNTING

VALIDATION FILES

OAV - VALIDATES USER NUMBER AND INDICATES WHERE LEVEL-2 BLOCK IS FOR USE BY ITS CALLER

- OPTIONALLY DECREMENTS SECURITY COUNT



ACCOUNTING

PROJECT PROFILE

PROFILa - CHARGE, PROJECT ACCOUNTING INFORMATION FILE

- SYSTEM ORIGIN - CREATES CHARGE AND MASTER USER SKELETON
- SPECIAL USERS - CSAP IN AACW
- MASTER USER - PROJECT MAINTENANCE WITHIN CHARGE

PROFILE - MAINTENANCE UTILITY

PROFILa LEVEL-0 CONTAINS HISTORY INFORMATION AND FIRST CHARGE NUMBER AND RANDOM ADDRESS OF EACH LEVEL-1 BLOCK.

LEVEL-1 CONTAINS M1-M4,AD VALUES FOR THE CHARGE NUMBER, PROJECT COUNTS, CHARGE EXPIRATION, AND MASTER USER NUMBER. LEVEL-1 BLOCK POINTS TO THE LEVEL-2 BLOCK.

LEVEL-2 BLOCK CONTAINS PROJECT NUMBERS WITH RANDOM ADDRESS FOR THEIR LEVEL-3 DATA.

LEVEL-3 BLOCK HAS SRU ACCUMULATION AND A LIST OF VALIDATED USERS (AS MANY OVERFLOW BLOCKS TO ACCOMMODATE USERS AS NEEDED)

ACCOUNTING

PROJECT PROFILE

LEVEL-3 BLOCK "PROJECT BLOCK"

- PROJECT EXPIRATION DATE
- TIME IN - TIME OUT
- LAST UPDATE DATE/TIME (BY OAU)
- SRU LIMIT AND ACCUMULATOR
- INSTALLATION LIMIT AND ACCUMULATORS
- USER NUMBERS OF USERS IN THE PROJECT

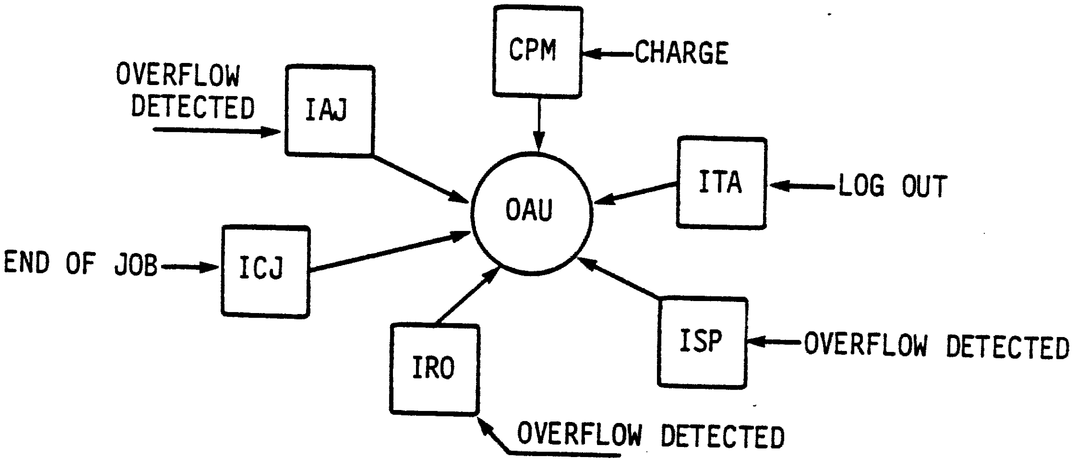
PROJECT PROFILE

CHARGE

1. INDICATE USER'S CHARGE AND PROJECT NUMBERS TO ACCOUNT DAYFILE.
2. ENTER SRU MULTIPLIERS (M1-M4,AD) INTO CONTROL POINT VIA CPM RA+1 REQUEST.
3. ON SUBSEQUENT CHARGE STATEMENTS, ENTER ACCUMULATOR INFORMATION INTO DAYFILE, SRU ACCUMULATION INTO ACCOUNT DAYFILE AND UPDATE LEVEL-3 BLOCK (OAU). MULTIPLIERS ARE RESET (STEP 2).
4. CLEAR SRU ACCUMULATOR BUT LEAVE OTHER ACCUMULATORS (AD, CP, MS, MT, PF) AS IS.

OAU

UPDATES LEVEL-3 BLOCK WITH SRU ACCUMULATION



QUESTION SET LESSON 15

1. What is an SRU? What is the SRU algorithm?

2. CPUMTR processes the SRU in various components. Describe what the following CPUMTR subroutines do:

   Apply Adder (AAD)

   Apply I/O Increment (AIO)

   Compute CP Time (CPT)

   Calculate SRU Multipliers (SRU)

3. What is an "account block"?

4. What information is kept in the validation file VALIDUs? VALINDs?

5. What information is kept in the project profile file PROFILa?

6. What is the relationship between the validation files and the project profile file?

7. What does the "GEAC" code in an ACCOUNT dayfile message indicate?

8. What information does the level-2 block in VALIDUs contain?

9. What information is contained in the level-1 PROFILa block? The level-2 block? The level-3 block?

LESSON 16
PERMANENT FILES

LESSON PREVIEW:

This lesson discusses how the system manages permanent files. The use of permanent files by the user is presented in many reference manuals and user's guides, and will not be covered in this lesson. Thus, it is assumed that the student is familiar with NOS permanent files on the end user level.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Explain the permanent file "family" concept of NOS.
- Define the types of permanent files in NOS and why there are two types of access.
- Discuss in detail the relationship the user's user index has to permanent files.
- Detail the information in a permanent files's catalog and permit entry.
- Describe the process for cataloging an indirect access permanent file.
- Describe the process for cataloging a direct access permanent file.
- Define a "hole" and discuss its use.
- Detail the Mass Storage Table words associated with permanent files.
- Explain what "catalog track overflow" is.
- Explain how more than one user can have access to a direct access file and a indirect access file.

REFERENCES:

NOS IMS - Chapter 14 NOS Reference Manual, 1-2-7, 1-8, 2-5 NOS IHB, 7-31-41 NOS SMRM, 1-1-19

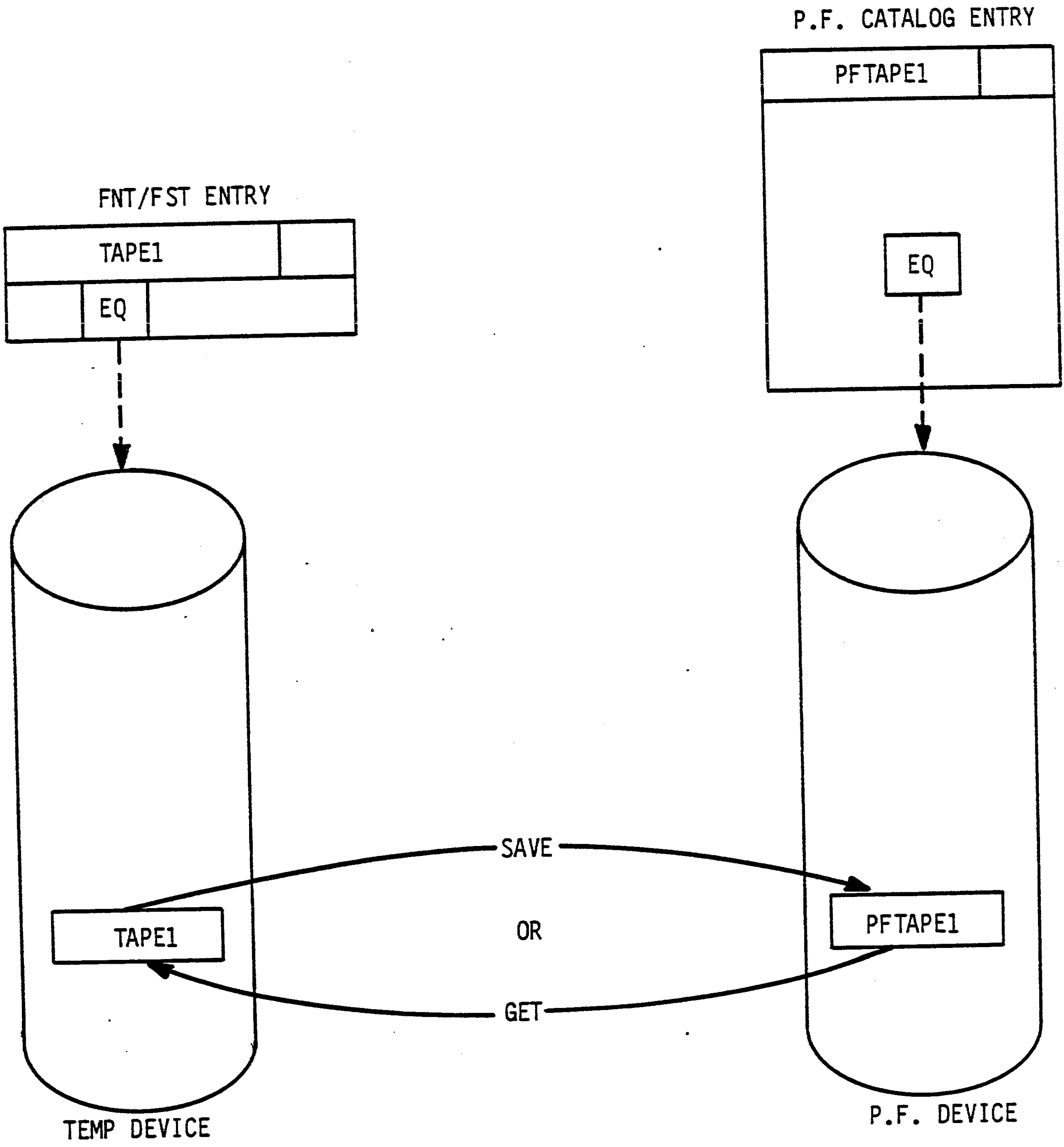
ASSIGNMENT:

Read NOS System Maintenance Reference Manual, pages 1-1 through 1- 19.

NOS PERMANENT FILE CONCEPTS

- NOS SUPPORTS TWO TYPES OF PERMANENT FILES.
  - DIRECT ACCESS FILES (DAFS)
  - INDIRECT ACCESS FILES (IAFS)
- MASS STORAGE DEVICES CONTAINING PERMANENT FILES ARE GROUPED INTO FAMILIES.
  - FILES MAY ALSO RESIDE ON PRIVATE PACKS.
- A PERMANENT FILE FAMILY CONSISTS OF MASTER DEVICES AND SECONDARY DEVICES.
  - MASTER DEVICES CONTAIN PERMANENT FILE CATALOG (PFC) ENTRIES, PERMIT AND ACCESS CHAINS, INDIRECT ACCESS FILES (IAFS), DIRECT ACCESS FILES (DAFS).
  - SECONDARY DEVICES CONTAIN DIRECT ACCESS FILES.
- PERMANENT FILE ACTIVITY IS CONTROLLED BY USER NAME, USER INDEX, AND INSTALLATION DEFINED LIMITS.

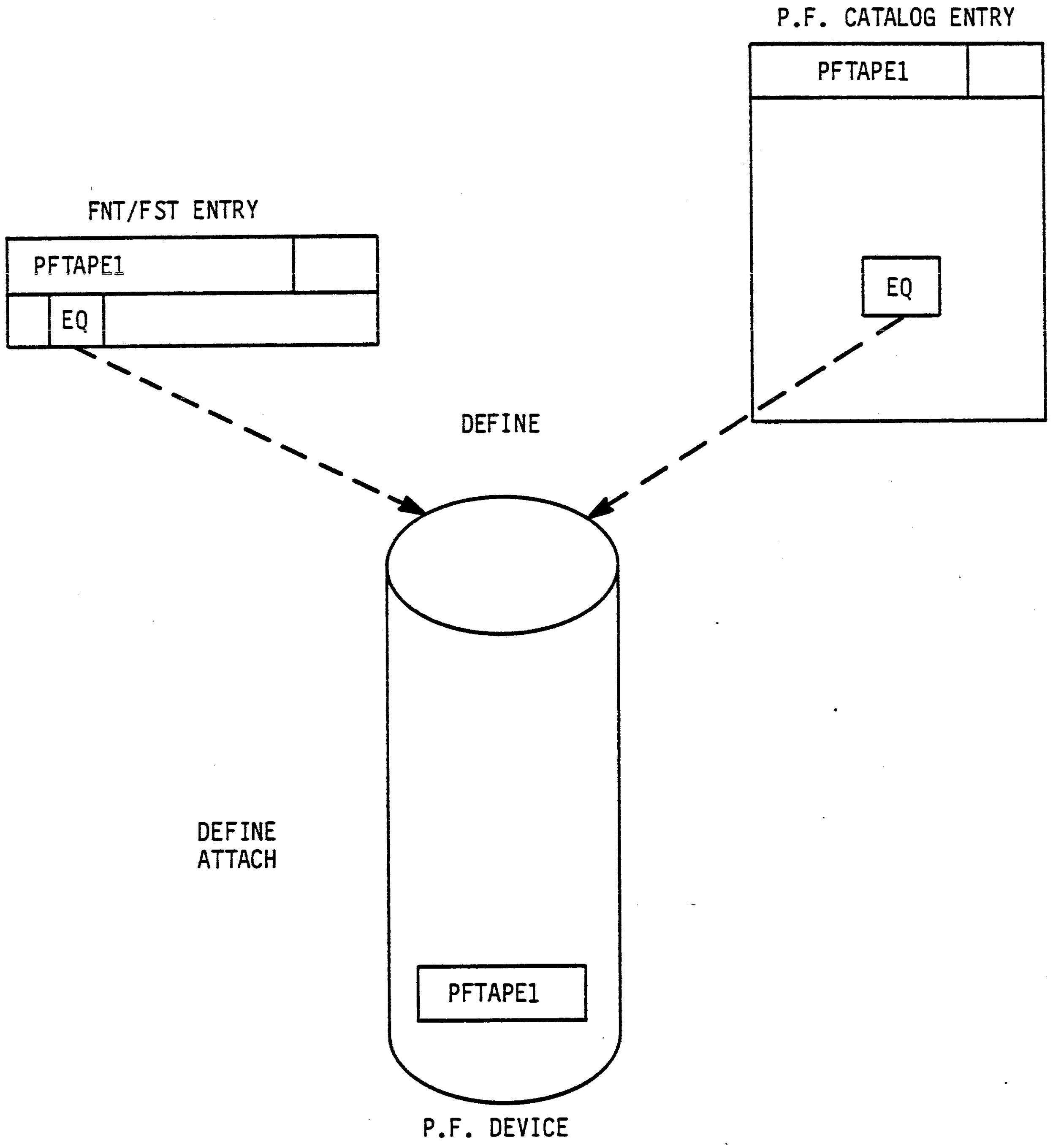
INDIRECT ACCESS PERMANENT FILE



INDIRECT ACCESS FILES

- USER OBTAINS AND MANIPULATES A LOCAL COPY OF THE FILE.
- THE PERMANENT COPY IS NOT CHANGED UNTIL THE USER DIRECTS THE SYSTEM TO REPLACE THE PERMANENT COPY OF FILE WITH THE LOCAL COPY.
- NORMALLY USED FOR RELATIVELY SMALL FILES. INDIRECT ACCESS FILE STORAGE IS ALLOCATED IN MASS STORAGE PRU INCREMENTS (100B CM WORDS).
- ALWAYS RESIDE ON THE OWNER'S MASTER DEVICE.

DIRECT ACCESS PERMANENT FILE



DIRECT ACCESS FILES

- USER ATTACHES AND MANIPULATES THE FILE ITSELF.
- SEVERAL USERS MAY SIMULTANEOUSLY ATTACH THE FILE IN READ MODE.
- NORMALLY USED FOR LARGE FILES. DIRECT ACCESS FILES ARE ALLOCATED IN INCREMENTS OF LOGICAL TRACKS.
- MAY RESIDE ON THE OWNER'S MASTER DEVICE OR ON A SECONDARY DEVICE.

NOS "FAMILY" CONCEPTS

- A FAMILY CONSISTS OF 1-8 MASTER DEVICES AND 0 OR MORE SECONDARY DEVICES. THE NUMBER OF SECONDARY DEVICES IS LIMITED BY PRACTICAL CONSIDERATIONS SUCH AS NUMBER OF MASS STORAGE DEVICES AVAILABLE, EST SIZE, ETC.

- IN A TYPICAL OPERATING ENVIRONMENT, EACH FAMILY CONTAINS A "VALIDUS" FILE WHICH IS A LIST OF VALIDATED USER NAMES AND CORRESPONDING USER INDICES AND OTHER INSTALLATION DEFINED VALIDATION LIMITS.

- ALL FILES AND CATALOG ENTRIES ASSOCIATED WITH A USER NUMBER/USER INDEX RESIDE WITHIN THE FAMILY IN WHICH THE USER NUMBER IS DEFINED.

- MORE THAN ONE FAMILY MAY BE ACTIVE DURING SYSTEM OPERATION. (USERS MUST SPECIFY A FAMILY NAME AT LOGIN TIME, OR ON THEIR USER STATEMENT IF THE DESIRED FAMILY IS NOT THE INSTALLATION DEFINED DEFAULT).

- EACH DEVICE WITHIN A FAMILY HAS A PRIMARY MASK AND SECONDARY MASK ASSOCIATED WITH IT.

  - MASKS ARE DEFINED BY SITE PERSONNEL.

  - MASKS ARE USED TO ASSIGN USER FILES, CATALOG ENTRIES, ETC., TO SPECIFIC DEVICES WITHIN A FAMILY.

PERMANENT FILES

ALL PERMANENT FILE ACTIVITY IS CONTROLLED BY THE USER'S USER NUMBER AND USER INDEX.

USER INDEX

- ABILITY TO USE PFS

DIRECT ACCESS FILES	CLPF
INDIRECT ACCESS FILES	CSPF
AUXILIARY COMMANDS (REMOVABLE PACKS)	CSRP

- DETERMINE WHICH DEVICE IN THE FAMILY IS THE USER'S MASTER DEVICE.
- DETERMINE WHICH DEVICE IN THE FAMILY IS THE USER'S SECONDARY DEVICE.
- DETERMINE WHICH CATALOG TRACK HOLDS THE USER'S PERMANENT FILE CATALOG ENTRIES.

DIRECT ACCESS FILE - USER GETS FILE WITH ITS TRACK/SECTORS "LIVE".

INDIRECT ACCESS FILE - USER GETS COPY OF FILE, ORIGINAL FILE REMAINS UNTOUCHED UNLESS REPLACED/PURGED.

USER INDEX

- EACH USER NUMBER HAS A 17-BIT USER INDEX ASSOCIATED WITH IT. THIS ASSOCIATION IS THROUGH AN ENTRY IN THE *VALIDUS* FILE WHICH IS SET UP BY THE INSTALLATION.
- SPECIAL USER NAMES/USER INDICES

USER NAME	USER INDEX
SYSTEMX	377777B
LIBRARY	377776B

(NOT TO BE CONFUSED WITH THE NOS CONCEPT OF A USER OR SYSTEM LIBRARY OR THE *LIBRARY* CONTROL STATEMENT.)

- THE USER INDEX, TOGETHER WITH DEVICE MASK VALUES, DETERMINE WHICH DEVICE(S) THE USER'S FILES, CATALOG, AND PERMITS RESIDE.
- *SUI* CONTROL STATEMENT AND *SETUI* COMPASS MACRO SERVE TO PROVIDE SYSTEM ORIGIN JOBS WITH A CONVENIENT MEANS OF ACCESSING CATALOGS.

USING THE USER INDEX TO

LOCATE THE MASTER DEVICE

NOTE:

1) THE USER'S MASTER DEVICE CONTAINS ALL CATALOG ENTRIES ASSOCIATED WITH A GIVEN USER INDEX/USER NAME.

2) EACH USER INDEX HAS EXACTLY ONE MASTER DEVICE ASSOCIATED WITH IT.

3) EACH MASTER DEVICE IN THE FAMILY HAS AN 8-BIT PRIMARY DEVICE MASK ASSOCIATED WITH IT.

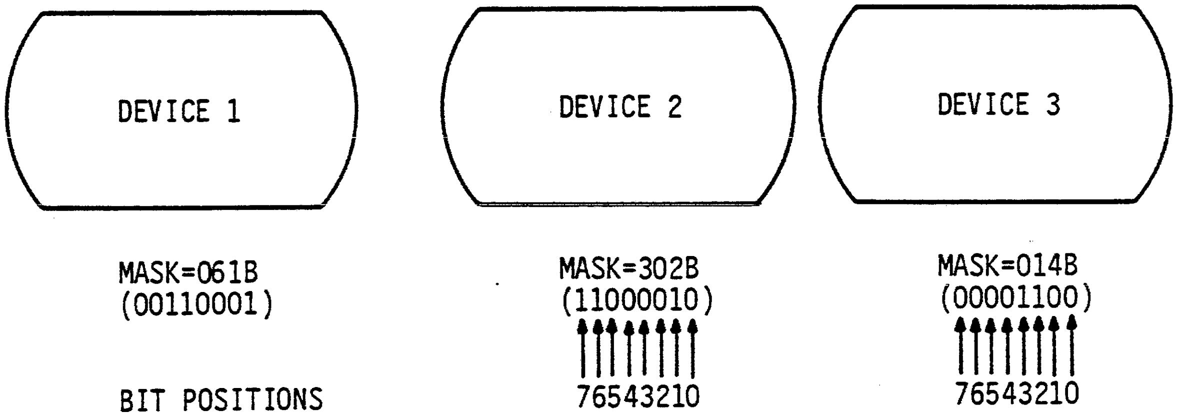
4) A FAMILY CAN CONTAIN AT MOST 8 MASTER DEVICE MASKS.

1) TAKE THE RIGHTMOST 3-BITS (OCTAL DIGIT) OF THE USER INDEX.

2) BITS OF THE DEVICE MASKS ARE NUMBERED 0-7 WITH BIT 0 BEING THE RIGHTMOST BIT IN THE MASK.

3) GIVEN THAT THE RIGHTMOST OCTAL DIGIT OF THE U.I. IS N, THE DEVICE WHOSE DEVICE MASK CONTAINS A 1 IN BIT POSITION N (SEE 2 ABOVE) IS THE MASTER DEVICE FOR THE U.I.

EXAMPLE:



MASTER DEVICE	RIGHTMOST OCTAL DIGIT OF U.I.
1	0
2	1
3	2
3	3
1	4
1	5
2	6
2	7

SAMPLE U.I.	DEVICE NO.
001467B	2
100104B	1
000012B	1

NOTE: THE LOGICAL SUM OF ALL PRIMARY DEVICE MASKS WITHIN A FAMILY <u>MUST</u> BE 377B.

MASTER DEVICE ASSIGNMENTS <u>MUST</u> BE UNIQUE.

FROM OUR PREVIOUS EXAMPLE

DEVICE 1 MASK = 00110001 (061B)
DEVICE 2 MASK = 11000010 (302B)
DEVICE 3 MASK = 00001100 (014B)

LOGICAL SUM = 11111111 (377B)

EACH COLUMN MUST HAVE EXACTLY ONE 1 IN IT.

INVALID DEVICE MASK ASSIGNMENT

DN1 00110001
DN2 01010010
DN3 00001100

NO 1'S IN THIS COLUMN

MORE THAN ONE 1 IN THIS COLUMN

USING THE USER INDEX TO LOCATE THE CATALOG TRACK ON THE MASTER DEVICE

NOTE: 1) EACH MASTER DEVICE CONTAINS A PREALLOCATED NUMBER OF TRACKS WHICH ARE RESERVED FOR CATALOG ENTRIES.

2) THE NUMBER OF CATALOG TRACKS ON EACH DEVICE IS INSTALLATION DEFINABLE.

3) THE NUMBER PREALLOCATED OF CATALOG TRACKS ON EACH DEVICE MUST BE AN EXACT POWER OF 2 IN THE RANGE 1B-200B (2**N, $0 \leq N \leq 7$).

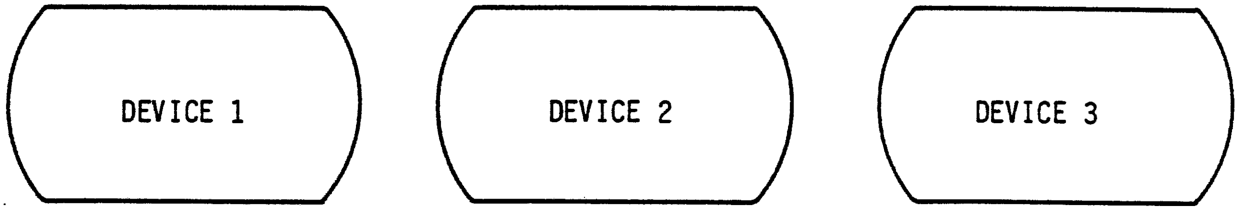
4) PROVISION IS MADE FOR CATALOG TRACK OVERFLOW.

1) DELETE THE RIGHTMOST 3 BITS (OCTAL DIGIT) FROM THE USER INDEX.

2) FORM A MASK BY SUBTRACTING 1 FROM THE NUMBER OF PREALLOCATED TRACKS FOR THE DEVICE. THIS MASK WILL BE N-BITS LONG WHERE N IS THE POWER OF 2 DEFINING THE NUMBER OF CATALOG TRACKS.

3) FORM THE LOGICAL PRODUCT BETWEEN THE MASK IN 2) AND THE RIGHTMOST N-BITS OF THE VALUE OBTAINED IN 1). THE RESULT IS THE NUMBER OF THE LOGICAL TRACK CONTAINING CATALOG ENTRIES FOR THE GIVEN U.I.

EXAMPLE:



MASK = 00110001 (061B)	11000010 (302B)	00001100 (014B)
NC = 100000 (40B)	10000 (20B)	1000000 (100B)

UI	MASTER DEVICE		CATALOG TRACK	
123456 B	2		001010011100101	(12345B)
		NC-1 =	001111	(17B)
		LOGICAL PRODUCT =	101	5B
		CATALOG TRACK =	101	5B
			001100011	(143B)
1433 B	3	NC-1 =	111111	(77B)
		LOGICAL PRODUCT =	100011	(43B)
		CATALOG TRACK =	100011	(43B)

RIGHTMOST OCTAL DIGIT
DETERMINES MASTER DEVICE

REMAINING OCTAL DIGITS
DETERMINES CATALOG TRACK

SECONDARY DEVICE MASKS

1) USED TOGETHER WITH THE U.I. TO ASSIGN DIRECT ACCESS FILES TO DEVICES.

2) THE LOGICAL SUM OF SECONDARY DEVICE MASKS NEED NOT BE 377B.

3) BIT ASSIGNMENTS NEED NOT BE UNIQUE.

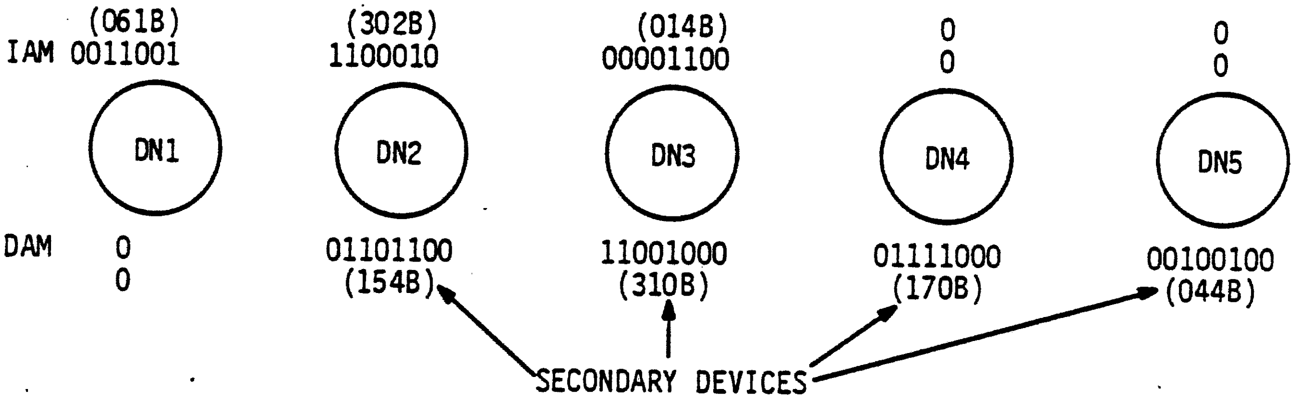
4) MORE THAN 8 SECONDARY DEVICES MAY BE PRESENT WITHIN A FAMILY.

5) A SECONDARY DEVICE MAY ALSO BE A MASTER DEVICE, BUT NEED NOT BE.

6) USER INDICES ARE ASSOCIATED WITH SECONDARY DEVICE MASKS IN A MANNER SIMILAR TO PRIMARY DEVICE MASKS.

EXAMPLE: IAM = PRIMARY DEVICE MASK
DAM = SECONDARY DEVICE MASKS

MASTER DEVICES



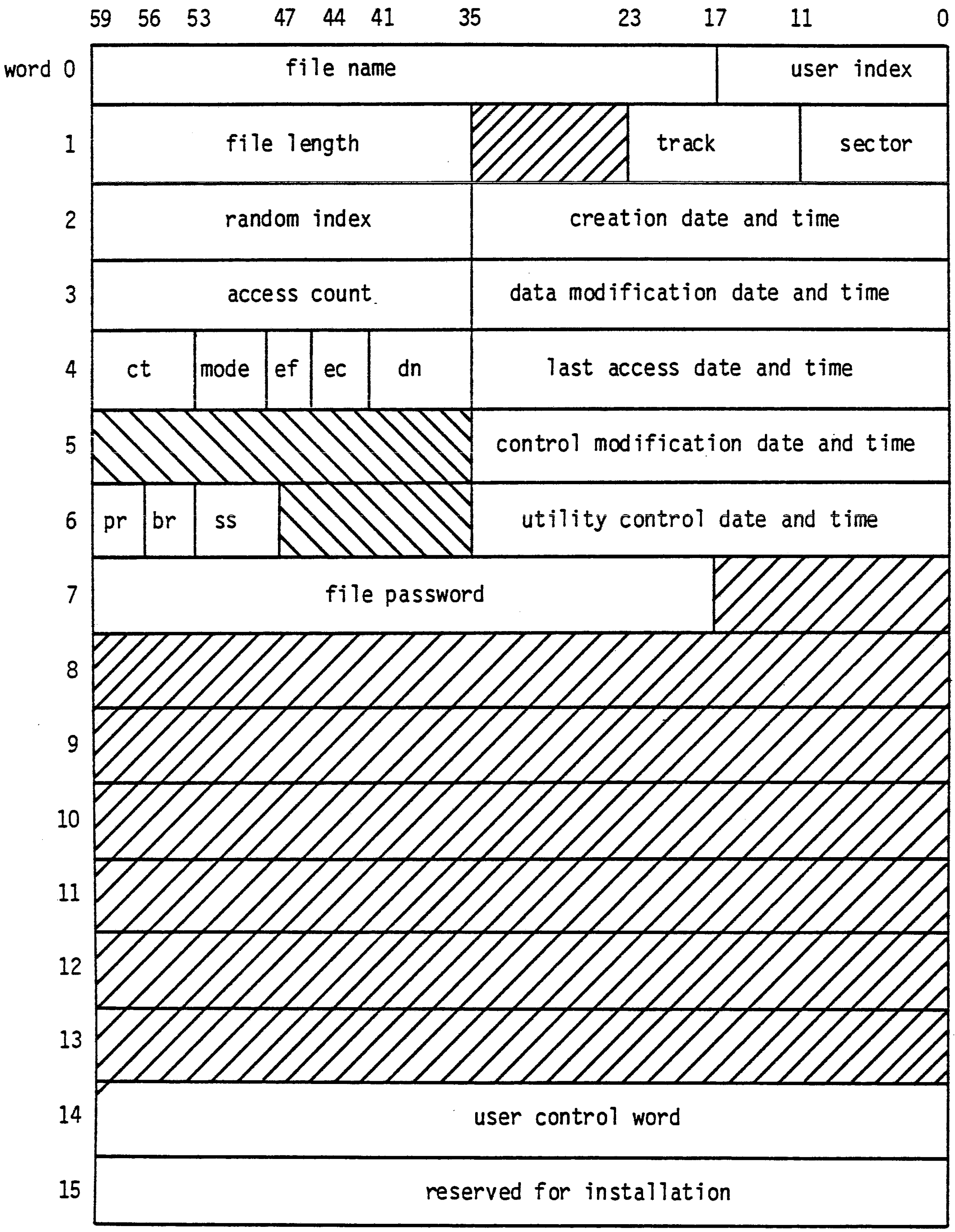
PFFAM

DN	UI'S	SECONDARY MASKS	
1	NONE	00000000	0
2	6, 5, 3, 2	01101100	154B
3	7, 6, 3	11001000	310B
4	6, 5, 4, 3	01111000	170B
5	5, 2	00100100	044B
	LOGICAL SUM =	11111100	

NOTE: UI'S ENDING IN 1 OR 0 CANNOT HAVE DIRECT ACCESS FILES.

THE CATALOG ENTRY FOR A DAF MAY BE ON A DIFFERENT DEVICE FROM THE FILE ITSELF.

PERMANENT FILE CATALOG ENTRY

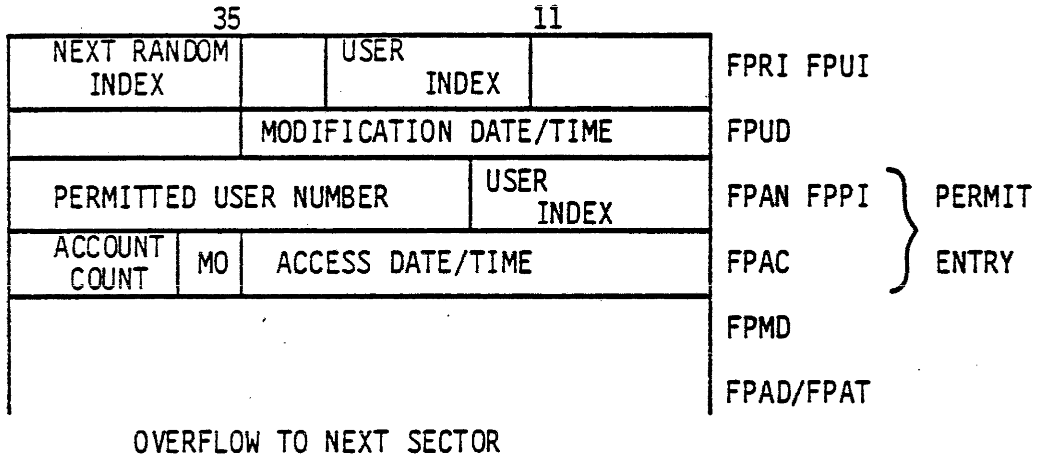


word 0	file name	user index				
1	file length	▨	track	sector		
2	random index	creation date and time				
3	access count	data modification date and time				
4	ct	mode	ef	ec	dn	last access date and time
5	▨	control modification date and time				
6	pr	br	ss	▨	utility control date and time	
7	file password	▨				
8	▨					
9	▨					
10	▨					
11	▨					
12	▨					
13	▨					
14	user control word					
15	reserved for installation					

CATALOG TRACK ORGANIZATION

1) 4 PFC ENTRIES PER PRU.
2) NUMBER OF PRU'S PER TRACK IS DEVICE DEPENDENT.
3) AVAILABLE PFC ENTRIES HAVE UI=0. ENTRIES ARE REUSED WHEN FILES ARE PURGED.
4) CATALOG TRACK IS SEARCHED SEQUENTIALLY.
5) CATALOG TRACK OVERFLOW IS HANDLED USING TRACK LINKING.

PF PERMIT ENTRY



NEXT RANDOM INDEX IF OVERFLOW FOR THIS FILE = 0 IS THE END FOR THIS FILE.

EXPLICIT - OWNER DEFINES PERMIT ALLOWING OR DISALLOWING ACCESS AND MODE FOR THE PERMITTED USER.

IMPLICIT - AUTOMATIC RECORDING OF ACCESS BY ANY USER TO SEMI-PRIVATE FILE WHEN ACCESS IS MADE.

<u>MASTER USER</u>

*IN USER NUMBER OK
NON-* MUST MATCH

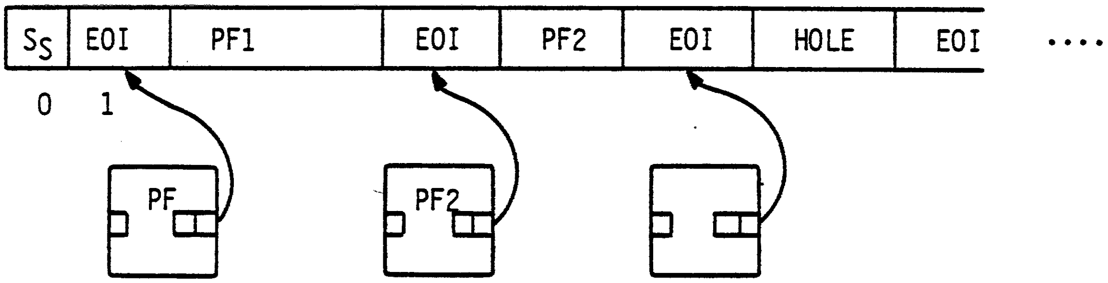
USER*** U*S*E*R
USERABC U1S1E2R

AUTOMATIC READ ONLY PERMISSION TO FILES IN THAT USER'S CATALOG.

```
CATLIST,LO=F.

 CATALOG OF ARR2061           FM/NOSCLSH 79/10/22. 09.35.57.

    FILE NAME ACCESS FILE-TYPE LENGTH DN CREATION  ACCESS    DATA MOD
     PASSWORD MD/CNT INDEX  PERM. SUBSYS DATE/TIME DATE/TIME DATE/TIME
     PR BR RS

    1 CASDATA  DIR. PRIVATE        32  * 79/09/10. 79/10/17. 79/10/17.
                 63         WRITE        15.38.43. 15.50.44. 15.48.56.
      N  Y D
    2 MIKEY    DIR. SEMI-PR       377  * 79/04/23. 79/09/19. 79/07/20.
                184         READ         15.40.13. 16.15.59. 13.47.01.
      N  Y D
    3 FRED     IND. PRIVATE         1    79/08/22. 79/09/06. 79/08/22.
                 10         WRITE FTNTS  16.19.54. 13.07.22. 16.19.54.
      N  Y D
    4 PROCFIL  IND. PRIVATE         2    79/09/07. 79/10/16. 79/09/07.
                 42         WRITE        11.05.19. 16.11.52. 15.54.25.
      N  Y D
    5 CASTPSR  DIR. PRIVATE         3  * 79/09/06. 79/10/17. 79/09/06.
                 51         WRITE        09.22.29. 15.50.41. 09.22.29.
      N  Y D
    6 CNTCRDS  IND. PRIVATE        20    79/09/07. 79/10/16. 79/09/07.
                 32         WRITE        11.05.20. 16.11.53. 11.05.20.
      N  Y D
    7 EJECT    IND. PRIVATE         3    79/09/07. 79/10/16. 79/09/07.
                 32         WRITE        11.05.19. 16.11.53. 11.05.19.
      N  Y D
    8 QUCAT    DIR. PRIVATE        27  * 79/09/07. 79/10/17. 79/09/07.
                 13         WRITE        08.39.51. 15.29.15. 08.39.51.
      N  Y D
    9 1MT15    IND. PRIVATE         2    79/09/19. 79/09/20. 79/09/19.
                  1         WRITE        14.40.31. 14.44.32. 14.40.31.
      N  Y D

         5 INDIRECT ACCESS FILE(S),    TOTAL PRUS =         28.
         4 DIRECT ACCESS FILE(S),      TOTAL PRUS =        439.

READY.
```

IAF CREATION

IAF (DATA) CHAIN ON MASTER DEVICE



- SAVE/REPLACE (NO EXISTING FILE)
- HOLE WITH EXACT FIT
- LARGEST HOLE LARGER THAN FILE
- END OF IAF CHAIN
- BUILD CATALOG ENTRY
- REPLACE (EXISTING FILE)
- SAME SIZE - WRITE OVER EXISTING DATA
- OLD FILE LARGER - WRITE OVER EXISTING DATA; CREATE HOLE ENTRY IF RESIDUE GREATER THAN ONE SECTOR.
- OLD FILE SMALLER - OLD ENTRY MADE INTO HOLE ENTRY, FIND NEW HOLE (AS ABOVE)

HOLE

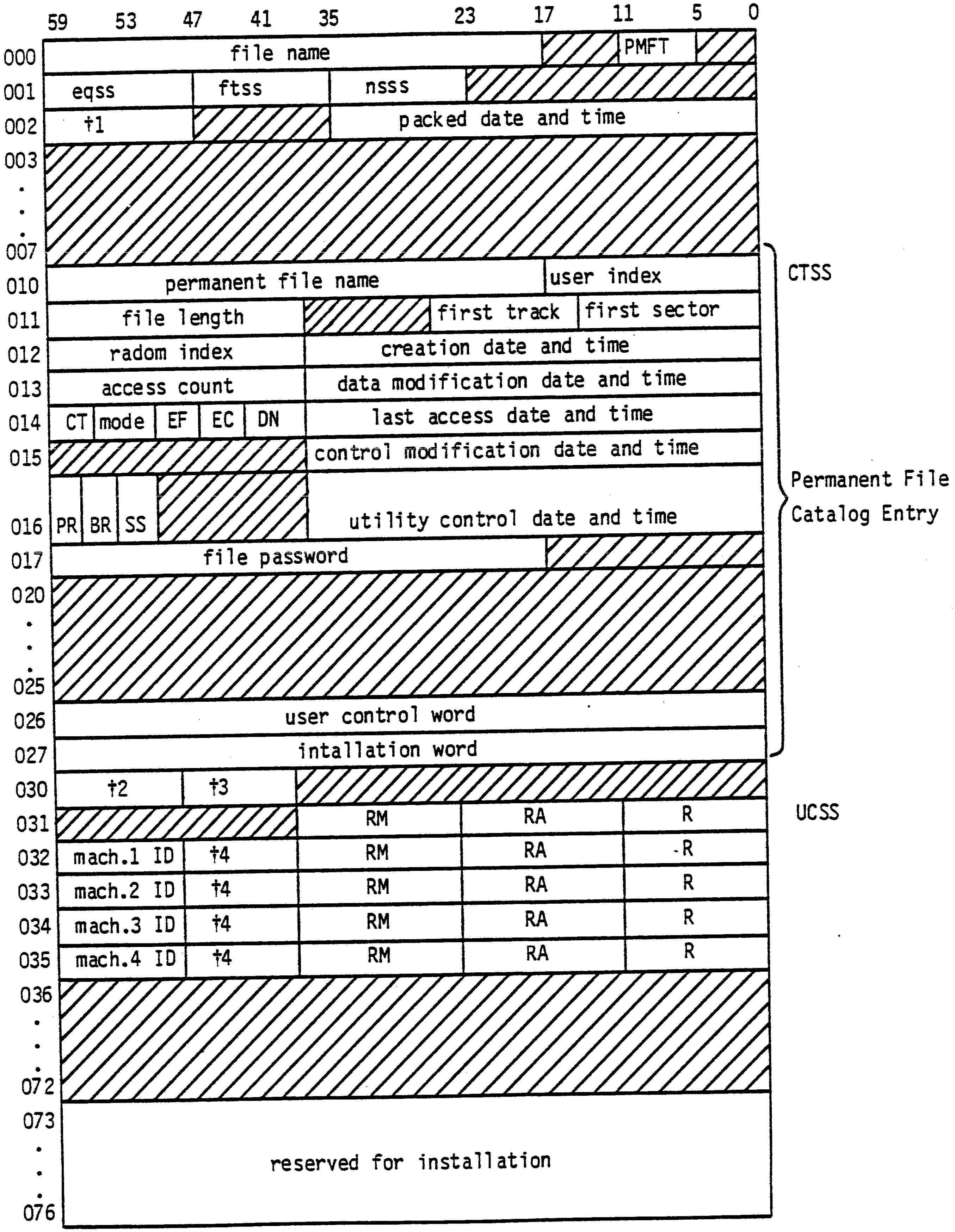
- CATALOG ENTRY POINTING TO UNUSED SPACE
- HAS UI=0 WITH SAME TRACK, SECTOR AND LENGTH VALUES

PERMANENT FILES

DAF CREATION

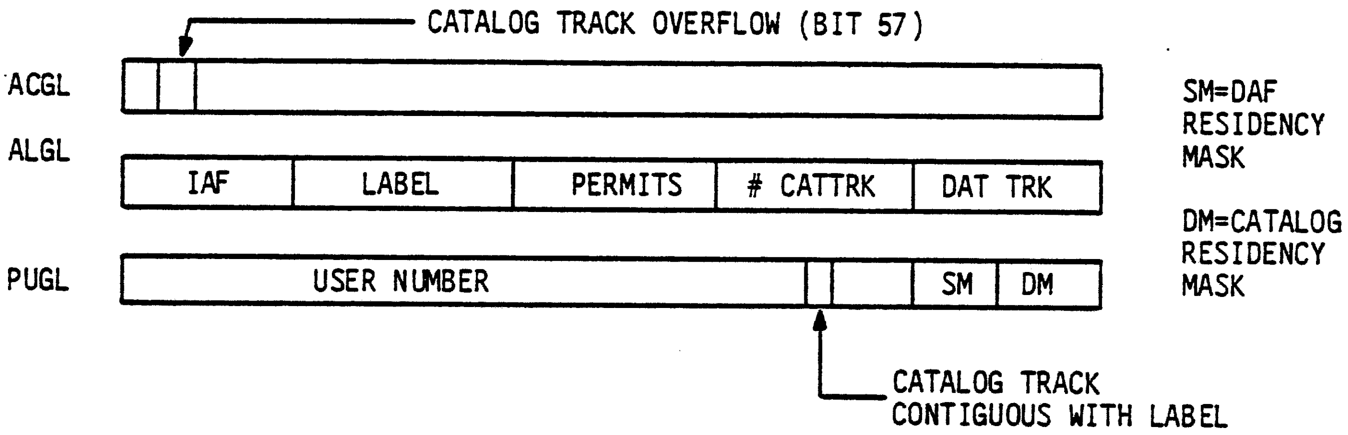
- CATALOG IS ON MASTER DEVICE.
- PERMITS ON MASTER DEVICE.
- FILE RESIDENCY DETERMINED BY SECONDARY MASK.
- DEFINE.
    - IF FILE NOT PRESENT ON SECONDARY DEVICE, CONTINUE; OTHERWISE ABORT.
    - IF NOT PRESENT, SELECT SECONDARY DEVICE IN FAMILY WITH MOST SPACE.
    - REQUEST SPACE ON SELECTED DEVICE.
    - BUILD CATALOG ENTRY ON MASTER DEVICE WITH DN FOR SECONDARY DEVICE.
    - SET FIRST TRACK
    - SET SECTOR TO 4000
    - WRITE SYSTEM SECTOR
    - PRESERVE FILE BY SETTING PRESERVE TRT BIT FOR FIRST TRACK VIA STBM.
    - WRITE EOI AT SECTOR 1.
- WHEN FILE RETURNED, ODF (DROP FILE) DETECTS PMFT AND CALLS ORP WHICH UPDATES SYSTEM SECTOR MODES, DATES, SECTOR LENGTH.

Direct Access File System Sector Format



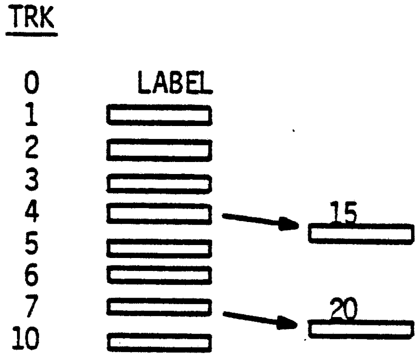
PERMANENT FILES

MST



IN TRT, CATALOG TRACK CHAIN IS LINKED BUT INDIVIDUAL SECTOR LINKAGE IS NOT PRESENT EXCEPT FOR CATALOG TRACK OVERFLOW.

ON CATALOG TRACK OVERFLOW, THE TRT LINKAGE IS TO THE OVERFLOW TRACK BUT ONLY THE TRACK THAT OVERFLOWS POINTS TO AN OVERFLOW TRACK.



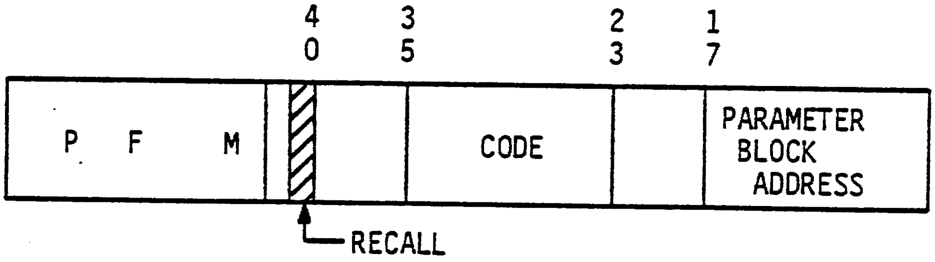
OVERFLOW TRT LINKAGE
1→2→3→4→5→6→
7→10→20→15

PERMANENT FILES

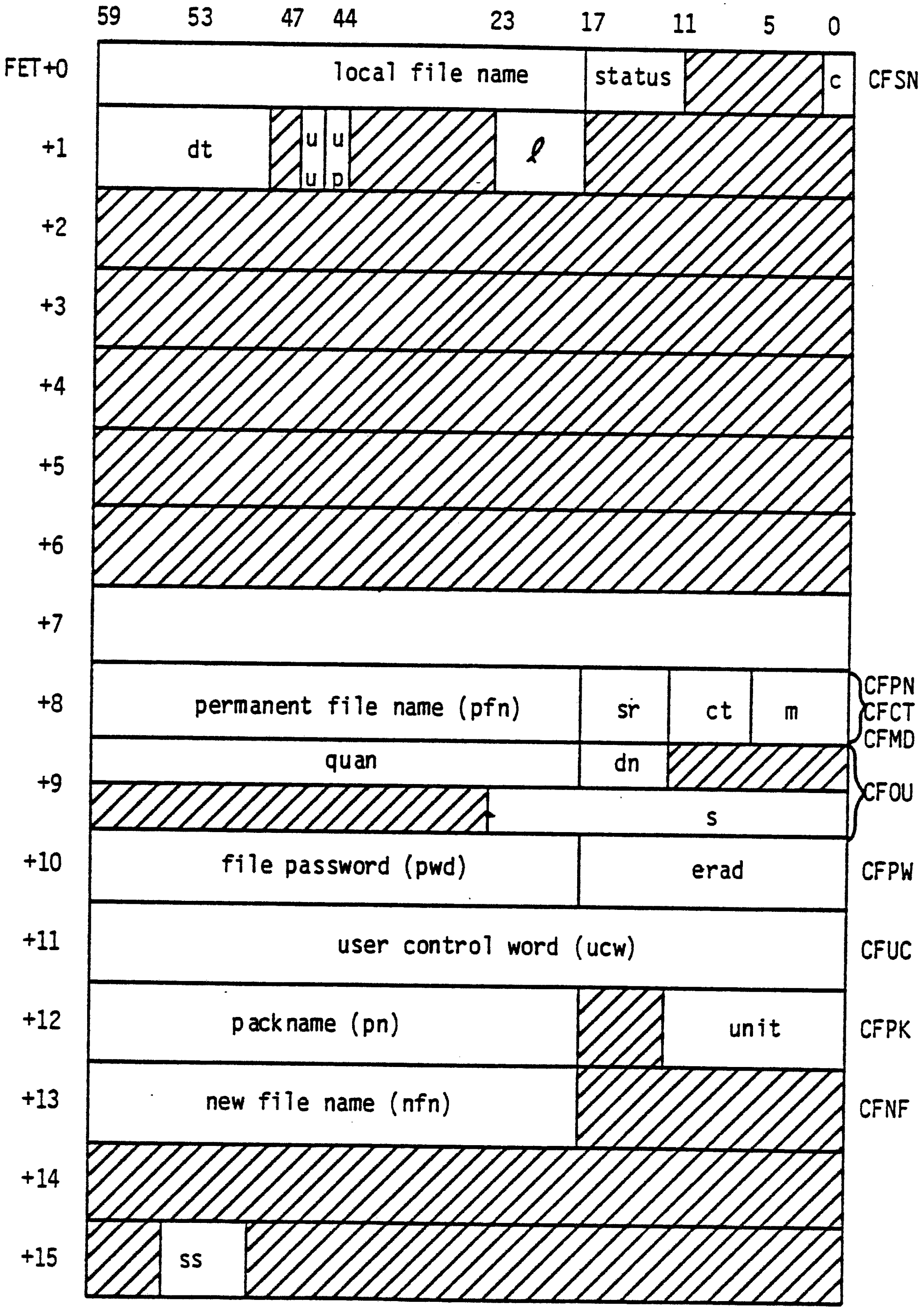
FUNCTION CODE	MACRO	CONTROL STATEMENT
1 CCSV	SAVE	SAVE
2 CCGT	GET	GET
3 CCPG	PURGE	PURGE
4 CCCT	CATLIST	*CATLIST
5 CCPM	PERMIT	PERMIT
6 CCRP	REPLACE	REPLACE
7 CCAP	APPEND	APPEND
10 CCDF	DEFINE	DEFINE
11 CCAT	ATTACH	ATTACH
12 CCCG	CHANGE	CHANGE *PURGALL

2000 + CODE	IGNORE DEFAULT PACKNAME
1000 + CODE	USE DEFAULT FAMILY

*OWN ROUTINES,
OTHERS IN PFILES



PFM FET


59
53
47
44
23
17
11
5
0
FET+0
local file name
status
c
CFSN
+1
dt
u u
u p
ℓ
+2
+3
+4
+5
+6
+7
+8
permanent file name (pfn)
sr
ct
m
CFPN
CFCT
CFMD
+9
quan
dn
s
CFOU
+10
file password (pwd)
erad
CFPW
+11
user control word (ucw)
CFUC
+12
packname (pn)
unit
CFPK
+13
new file name (nfn)
CFNF
+14
+15
ss

PERMANENT FILES

STRUCTURE OF PFM

PRESET	OVERLAYS	
	3PA	COMMAND PROCESSOR
	3PB	SAVE/REPLACE
	3PC	APPEND
	3PD	ATTACH
	3PE	CATALOG LIST
	3PF	DEFINE
	3PG	PERMIT/PURGE
	3PH	ERROR PROCESSOR
	3PI	AUXILIARY ROUTINES
	3PJ	CHANGE
	3PK	DEVICE-TO-DEVICE
	3PL	APPEND - ORIGINAL MOVE
	3PM	DEFINE AUXILIARY

PRESET

- VERIFY PARAMETERS IN CALL BLOCK
- VERIFY VALIDATION CONSTRAINTS
- RECALL IF CATALOG TRACK INTERLOCKED
- ISSUE ACCOUNTING MESSAGES
- CALL RESOURCE EXECUTIVE (RESEX)
- PROCESS FUNCTION (LOAD 3PA OR 3PE)

EXAMPLE

GET

1. DO IPAS/IAUM FUNCTION TO INCREMENT PF ACTIVITY.
2. FIND DEVICE WITH CATALOG.
3. TEST FOR PF UTILITY ACTIVE VIA TUAS/STBM.
4. DETERMINE IF TRACK (CATALOG TRACK) IS INTERLOCKED. COMMON DECK COMPDTS DOES THIS; IN A MMF CONDITION, DTS ISSUES A ECSM TO HAVE CPUMTR READ THE TRT WORD FROM THE ECS COPY OF THE TRT.
5. SET CATALOG TRACK INTERLOCK. COMMON DECK COMPSTI DOES THIS VIA STIS/STBM.
6. READ CATALOG TRACK AND FIND ENTRY.
7. UPDATE CATALOG ENTRY AND PERMITS.
8. REQUEST TRACKS ON TEMP DEVICE VIA RTCM.
9. LOAD 3PK AND DO DEVICE-TO-DEVICE TRANSFER.
10. WRITE EOI SECTOR AND UPDATE TRACK CHAIN VIA DTKM.
11. CLEAR CATALOG TRACK INTERLOCK. COMMON DECK COMPCTI DOES THIS VIA CTIS/STBM.
12. DECREMENT PF ACTIVITY BY DOING A DPAS/IAUM.

WITH THE EXCEPTION OF CODE IN COMPDTS TO READ TRT WORD FROM ECS, THIS LOGIC IS TRANSPARENT TO ANY MMP CONSIDERATIONS.

IN MOST CASES, SYSTEM CODE FOLLOWS SINGLE MAINFRAME DESIGN AND LETS CPUMTR MANAGE THE ACTUAL MAINFRAME ENVIRONMENT BY PRESETTING INDIVIDUAL MONITOR FUNCTIONS (RTCM, DTKM, DLKM, IAUM, STBM) TO REFLECT A STAND-ALONE OR MMF CONDITION.

QUESTION SET LESSON 16

1. How does PFM keep track of "holes" in the indirect chain?

2. What implied permission does user number USER*** have to files belonging to user number USERXYZ?

3. What is the difference between a semi-private and a public (library) file?

4. Must indirect access permanent files reside on a user's master device? Must direct access permanent files reside on a user's master device?

5. Is multi-read access possible with direct access permanent files? If so, how is it implemented?

6. Why is multi-read access no problem when working with indirect access permanent files?

7. What happens when a direct access file is ATTACHed with WRITE permission? If one user attaches a file with WRITE permission, can another user attach this file? Why?

8. What happens when a direct access file is PURGEd? An indirect access file is PURGEd?

9. What happens if the user DEFINEs an existing file that resides on a device not configured to hold direct access permanent files? Could this situation ever occur?

10. What happens when the user SAVEs a file? REPLACEs a file?

The following questions apply to family NOSCLSH and each equipment in this family. Use the study dump to answer these questions.

11. What is the first track of the indirect file chain? The permit chain?

12. How many tracks are allocated to indirect access permanent files? How did you derive your answer?

13. How many tracks are allocated for permit entries? For catalog entries?

14. What are the device masks and secondary masks for each equipment in the family?

15. Has catalog track overflow occurred on any of the equipments? How did you derive your answer?

For the following questions, family PSD has been defined as follows:

```
EQ4=DI-1,ON,0,0,5.
EQ5=DI-1,ON,0,1,5.
EQ6=DI-1,ON,0,2,5.
EQ7=DI-2,ON,0,3,5.
PF=4,F,201,303,PSD,7,10.
PF=5,F,102,303,PSD,6,40.
PF=6,F,44,74,PSD,5.
PF=7,F,30,74,PSD,4.
```

Assume all devices have no tracks flawed.

16. What tracks are in the catalog track chain for EQ4?

17. Complete the following table:

User Index Ending In Digit	Master Device	Secondary Device
0	------	------
1	------	------
2	------	------
3	------	------
4	------	------
5	------	------
6	------	------
7	------	------

18. What is the master device/catalog track mapping for user 1231? For user index 1237?

1231 master ------- catalog track -------
1237 master ------- catalog track -------

19. How many catalog tracks on device 4? On device 5?

20. Track 4234 is the first available track. If the second catalog track on device 7 overflows, describe the TRT and mass storage linkages.

LESSON 17
RESOURCE CONTROL

LESSON PREVIEW:

In this lesson, the student will learn the concepts of the allocation of magnetic tape and removable pack resources in NOS. The lesson consists of discussion on the resource executive (RESEX) and the files it uses, deadlock prevention, and magnetic tape assignment control statements.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Discuss the resource deadlock prevention mechanism employed by NOS.
- Detail the contents of the resource files and the relationships between them and FNT/FST entries for files residing on the magnetic tape and removable pack resource units.
- Discuss the role of routine ORF.
- Identify control statements processed by RESEX.
- Explain the terms "internal call" and "external call" to the resource executive.

REFERENCES:

NOS IMS - Chapter 12 NOS RM, 1-6-18-22

RESOURCE CONTROL

RESEX - RESOURCE EXECUTIVE MANAGES THE ALLOCATION OF MAGNETIC TAPE AND REMOVABLE PACK RESOURCES IN SUCH A WAY AS TO PREVENT A DEADLOCK FROM OCCURRING.

DEADLOCK

RESOURCE UNIT

RESOURCE TYPE

RESOURCE DEMAND

DEMAND COUNT

ASSIGN COUNT

DEMAND FILE

DEMAND FILE ENTRY

VSN FILE

VSN FILE ENTRY

RESOURCE ENVIRONMENT

SHARE TABLE

OVERCOMMITMENT ALGORITHM

PACK SHARING

SMALLEST SPINDLE RESIDUE

RESOURCE CONTROL

DEADLOCK PREVENTION

1. RESOURC CONTROL STATEMENT REQUIRED FOR ANY JOB USING MORE THAN ONE RESOURCE UNIT CONCURRENTLY.

2. RESEX BUILDS DEMAND FILE ENTRY FOR THE JOB AND WRITES IT TO THE DEMAND FILE. THE ENTRY IS USUALLY BUILT FROM THE RESOURC STATEMENT, WHICH TELLS WHAT THE JOB'S CONCURRENT DEMANDS ARE, AND INCLUDES THE JOB SEQUENCE NUMBER (RFCW) AND JOB NAME (JNMW).

3. RESEX VALIDATES THE DEMAND TO DETERMINE IF ENOUGH RESOURCE UNITS EXIST TO SUPPORT THE DEMAND AND THE USER IS VALIDATED TO HAVE THAT MANY RESOURCE UNITS (MT AND RP). THE VERIFICATION OF THE USER'S VALIDATION TO USE THE RESOURCE UNIT (CAND/CSOJ OR CSRP) WILL OCCUR WHEN USER ATTEMPTS TO ASSIGN THE UNIT.

4. WHEN ATTEMPT IS MADE TO ASSIGN RESOURCE UNITS, RESEX VERIFIES THAT THERE EXISTS A SEQUENCE OF JOB COMPLETIONS SUCH THAT <u>ALL</u> JOBS WITH ASSIGNED RESOURCES WILL COMPLETE.

DEADLOCK PREVENTION

5. BUILD "SCRATCH" FILE OF ALL JOBS WITH ASSIGNED RESOURCES.

STEP 1, READ SCRATCH FILE ENTRY

ARE JOB'S DEMANDS SATISFIABLE?

- CHAIN PACKS IF NECESSARY

IF YES, LOGICALLY RETURN ALL RESOURCE UNITS AND GO READ NEXT SCRATCH FILE ENTRY (STEP 1)

IF NO, COPY ENTRY TO SECOND SCRATCH FILE AND GO READ NEXT SCRATCH FILE ENTRY (STEP 1)

AT EOR, A) IF SECOND SCRATCH FILE IS EMPTY, ALL JOBS HAVE COMPLETED AND EVERYTHING IS OK!

B) FIRST SCRATCH = SECOND SCRATCH, OVER-COMMITTED (I.E., DEADLOCK)

C) SWITCH SCRATCH FILES, REWIND, GO TO STEP 1.

6. IF OK, ASSIGN RESOURCE.
IF OVERCOMMITTED, ABORT OR ROLLOUT.

RESOURCE CONTROL

RESOURCE FILES

RSXDid - DEMAND FILE
RSXVid - VSN FILE

EACH FILE HAS ENTRY = ONE PRU

FILE BUILD AND UPDATED BY RESEX
ENTRIES UPDATED/CLEARED BY ORF

ENTRY STRUCTURE DEFINED IN COMMON DECK COMSRSX

HEADER (IDENTIFIES JOB)

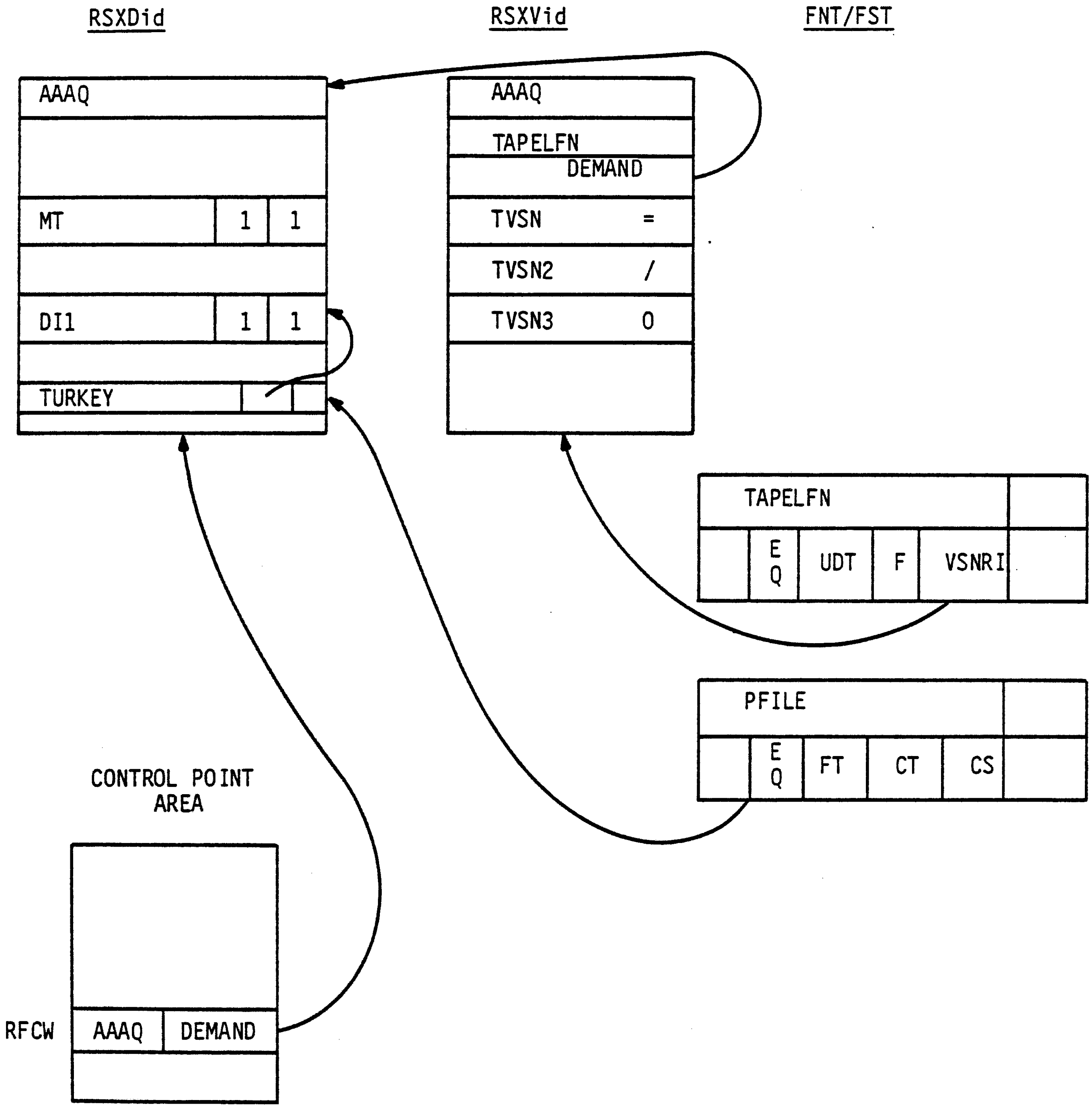
DEMANDS (RSXDid)

- ASSIGN COUNT
- DEMAND COUNT
- VALIDATION MT/RP
- PREVIEW DATA
- SHARE TABLE

VSN (RSXVid)

- FILE NAME
- DEMAND ADDRESS AND INDEX
- VSN WITH SEPARATOR
- 0 = END
  / = MULTIPLE
  = = EQUIVALENCE

RESOURCE CONTROL



Resource Demand File Entry (RSXVid)

59 56 53 47 35 23 17 11 8 5 0

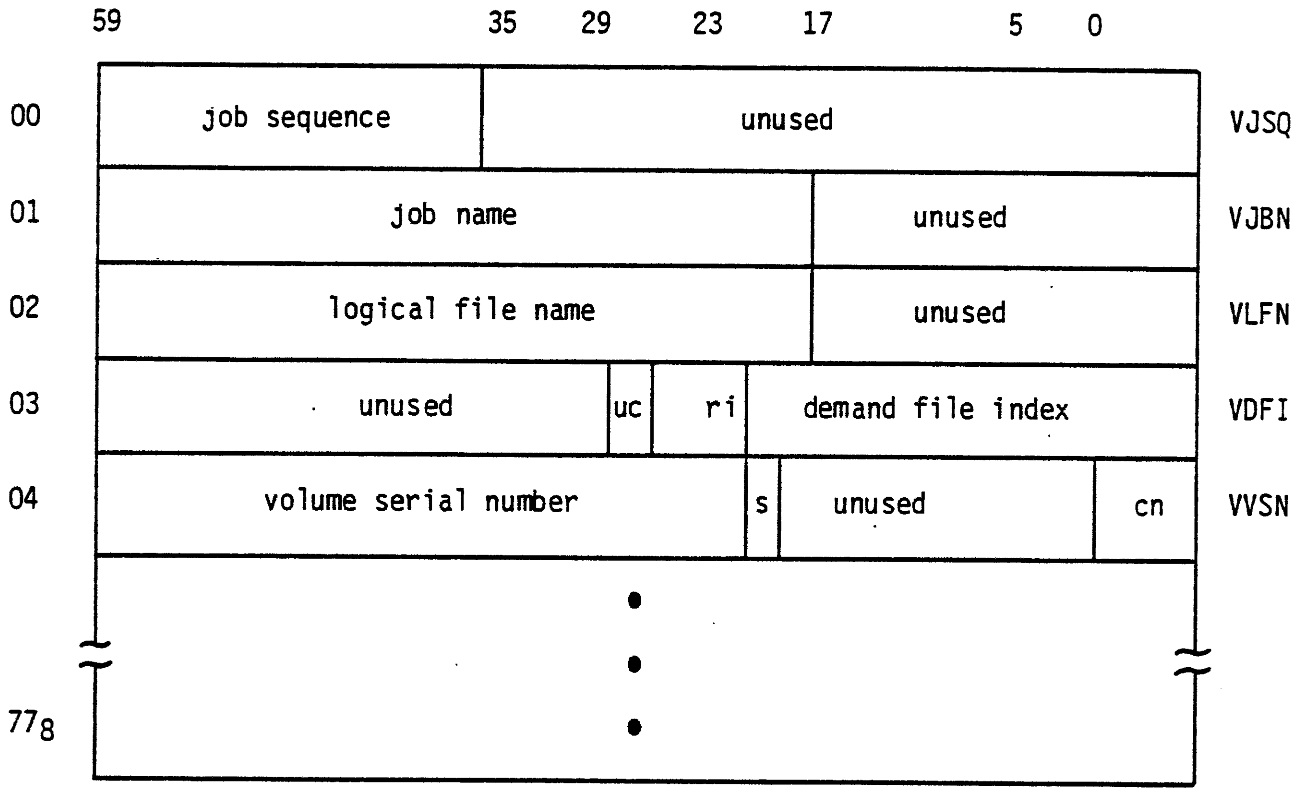
RJSQ	job sequence number	0					0
RJBN	job name	0					1
RVAL	tape val.	pack val.	lost	unused	total assigned	total demand	2
RMTP	MT	assigned	demand	0			3 7-track
RNTP	NT	assigned	demand	0			4 9-track (800/1600 cpi)
RPEP	PE	assigned	demand	0			5 1600 cpi 9-track
RHDP	HD	assigned	demand	0			6 800 cpi 9-track
RGEP	GE	assigned	demand	0			7 6250 cpi 9-track
RRPP	DI	AD(1)	AD(2)	AD(3)	AD(4)		10 844-21 (1 to 8 units)
	mpu	AD(5)	AD(6)	AD(7)	AD(8)		11 half-track
	DJ	AD(1)	AD(2)	AD(3)	AD(4)		12 844-41 (1 to 8 units)
	mpu	AD(5)	AD(6)	AD(7)	AD(8)		13 half-track
	DK	AD(1)	AD(2)	AD(3)	AD(4)		14 844-21 (1 to 8 units)
	mpu	AD(5)	AD(6)	AD(7)	AD(8)		15 full-track
	DL	AD(1)	AD(2)	AD(3)	AD(4)		16 844-41 (1 to 8 units)
	mpu	AD(5)	AD(6)	AD(7)	AD(8)		17 full-track
	MD	AD(1)	AD(2)	AD(3)	AD(4)		20 841 (1 to 8 units)
	mpu	AD(5)	AD(6)	AD(7)	AD(8)		21
RQPD	VSN or pack name	resource type					22 Preview data
RQPU	user number	flags	FST address				23
RQPT	0	time					24
RRPS	pack name	eq	0	uc	ri		25 Share table
	⋮	⋮	⋮	⋮	⋮		⋮ 77

VSN File Entry (RSXVid)

The VSN file (RSXVid) contains the volume serial numbers associated with a particular file (refer to figure below). The entries in the VSN file contain the random index of the job's entry in the demand file and the resource index of the assigned tape within the demand file. The entries in these two files are associated with a particular job and are identified by the job's sequence number. Entries in both files are one PRU (64 words) in length.

These files are updated by RESEX and PP routine ORF. Routine ORF (described later in this section) updates demand and VSN file entries upon the return/unload of a tape file or the job's last direct access file on a pack and at job completion.

VSN File Entry (RSXVid)



uc	Unit count (always 1)
ri	Resource index; points to a word in the demand file entry between RMTP and RGEP; zero if tape is not yet assigned
s	Scratch VSN (if set)
cn	Control byte as follows:

Value	Meaning
/	Multireel
=	Alternate reel
0	End of entries

RESEX/MAGNET Call Block

<table>
<tr><th></th><th>59</th><th>47</th><th>35</th><th>29</th><th>23</th><th>17</th><th>11</th><th>5 0</th></tr>
<tr><td>RCAL+0</td><td>fc</td><td colspan="7">interlock</td></tr>
<tr><td>+1</td><td colspan="2">0</td><td>ec</td><td>0</td><td colspan="2">user options</td><td colspan="2">UDT address</td></tr>
<tr><td>+2</td><td colspan="6">0</td><td>den</td><td>cnv</td></tr>
<tr><td>+3</td><td>word count</td><td>ouc</td><td colspan="2">format</td><td>est</td><td>ns</td><td colspan="2">software options</td></tr>
<tr><td>+4</td><td colspan="2">sequence number</td><td colspan="2">0</td><td>vsn</td><td colspan="3">VSN random address</td></tr>
<tr><td>+5</td><td colspan="5">job name</td><td>org</td><td colspan="2">0</td></tr>
<tr><td>+6</td><td colspan="4">volume serial number</td><td colspan="4">0</td></tr>
<tr><td>+7</td><td colspan="5">user number</td><td>fm</td><td colspan="2">0</td></tr>
</table>

fc	Function code (0=assign unit, 1=set error code in UVSN word of UDT)
ec	Error code for function 1
den	Density
cnv	Conversion
ouc	Overflow unused characters
est	EST ordinal
ns	Noise
vsn	VSN index
org	Job origin
fm	Family ordinal

RMT then updates the job's demand file entry, builds the preview display buffer, and performs any file opening required. RMT exits and control is returned to the routine that called RESEX. The format of the preview display buffer is as follows.

ORF - UPDATE RESOURCE FILES

CLEANS UP VSN AND DEMAND FILES WHEN FILE RETURN/UNLOADED/EVICTED OR AT JOB COMPLETION

- PACKS, CALLED WHEN LAST (DAF) FILE ON PACK (FOR JOB) IS RETURNED/UNLOADED
  - CLEAR SHARE TABLE ENTRY
  - DECREMENT ASSIGN COUNT
  - DECREMENT DEMAND COUNT IF RETURN AND TOTAL DEMAND MET
  - DOES NOT DECREMENT DEMAND COUNT ON UNLOAD
- TAPES, CALLED WHEN FILE RETURNED/UNLOADED
  - CLEAR VSN ENTRY
  - DECREMENT ASSIGN COUNT
  - DECREMENT DEMAND COUNT IF RETURN AND TOTAL DEMAND MET
  - DOES NOT DECREMENT DEMAND COUNT ON UNLOAD
  - CALLS MAGNET TO PHYSICALLY PROCESS THE TAPE (TDAM REQUEST)
- IF ONLY ONE DEMAND, CLEARS ENTIRE ENTRY
- VSN (UNASSIGNED), CALLED WHEN FILE RETURNED OR UNLOADED
  - CLEAR VSN ENTRY
- AT END OF JOB, CLEAR DEMAND FILE ENTRY

RESOURCE CONTROL

TAPE ASSIGNMENT CONTROL STATEMENTS

- ASSIGN
- REQUEST
- LABEL

VSN CONTROL

- VSN

RESOURCE CONTROL

- RESOURCE

CONTROL STATEMENTS PROCESSED BY RESEX

"EXTERNAL" CALLS VIA SPCW

LFM	FUNCTION	0	RENAME
		25	LABEL

- PFM
- REQ (NOS/BE REQUEST FROM SFP)

QUESTION SET LESSON 17

1. List the control statements processed by RESEX?
2. What does "deadlock prevention" mean?
3. Name each resource file and list what information each resource file contains.
4. How does RESEX assign a tape unit to the job? A removable pack?
5. What is an internal call to RESEX? An external call?
6. Define the following terms:

    Deadlock
    Resource unit
    Resource type
    Demand and Assigned counts
    Demand File and VSN File
    Demand File and VSN File Entries
    Resource Environment
    Share Table
    Pack Sharing
    Overcommitment Algorithm

LESSON 18
MAGNETIC TAPE SUBSYSTEM (MAGNET)

LESSON PREVIEW:

In this lesson, the student will study the magnetic tape subsystem which consists of an executive, MAGNET, and a driver, 1MT. This lesson is not meant to be a tutorial on the usage of tapes but rather an overview of the magnetic tape subsystem.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Describe the Unit Descriptor Table (UDT) and what each word in the table means.
- Describe the interfaces between the subsystem (MAGNET/1MT) and other system components, such as RESEX, DSD/1DS, CIO, and so forth.
- Briefly discuss the variety of ANSI standard and optional tape labels.
- Briefly discuss the PROC macro of MAGNET.
- Map the layout of MAGNET's field length.

REFERENCES:

NOS IMS - Chapter 13 NOS Reference Manual, 1-10, 1-G

MAGNET/1MT

1. CONTROLS ALL MAGNETIC TAPE OPERATIONS.
2. MAGNET EXECUTES AT A CONTROL POINT AND MAINTAINS INFORMATION ABOUT TAPES AND FOR RESEX AND SYSTEM.
3. STRUCTURE -

   CP ROUTINES

   MAGNET - EXECUTIVE, PROCESSES REQUESTS FROM RESEX, CIO, DSD.

   MAGNET1 - TERMINATION PROCESSOR

   PP ROUTINES

   1MT - TAPE DRIVER - MANY OVERLAYS (25).

   COMMON DECK

   COMSMTX - UDT LAYOUT, MAGNET FL LAYOUT, FUNCTIONS, ETC.
4. SUBSYSTEM

MAGNET/1MT

MAGNET/1MT BUILDS A UNIT DESCRIPTOR TABLE (UDT) FOR EACH DEFINED TAPE UNIT.

MAGNET CALLS 1MT TO PERFORM CERTAIN TAPE OPERATIONS THAT WERE INITIATED BY A CIO, RESEX, DSD/1DS REQUEST.

IN GENERAL, 1MT SEARCHES UDT FOR REQUESTS TO PROCESS. FIRST WORD OF UDT CONTAINS THE REQUEST. AS REQUESTS ARE COMPLETED, A RETURN CODE IS PLACED IN THE FIRST WORD OF THE UDT.

THERE WILL BE ONE COPY OF 1MT ACTIVE FOR EACH TAPE CHANNEL. EACH 1MT PROCESSES ONLY THOSE UDTS ON ITS CHANNEL.

TAPE LABELS

- UNIQUELY IDENTIFY FILE AND REEL ON WHICH IT RESIDES.
- MARK BEGINNING AND END OF FILE OR REEL.

REQUIRED	OPTIONAL	
VOL1	UVL1 - 9	VOLUME
HDR1	HDR2 - 9 UHLx	HEADER
EOF1	EOF2 - 9 UTLx	END OF FILE
EOV1	EOV2 - 9	END OF VOLUME

TAPE LABELS

VOLUME

LABEL IDENTIFIER
LABEL NUMBER
VOLUME SERIAL NUMBER (VSN)
VOLUME ACCESSIBILITY
OWNER IDENTIFICATION
LABEL STANDARD LEVEL

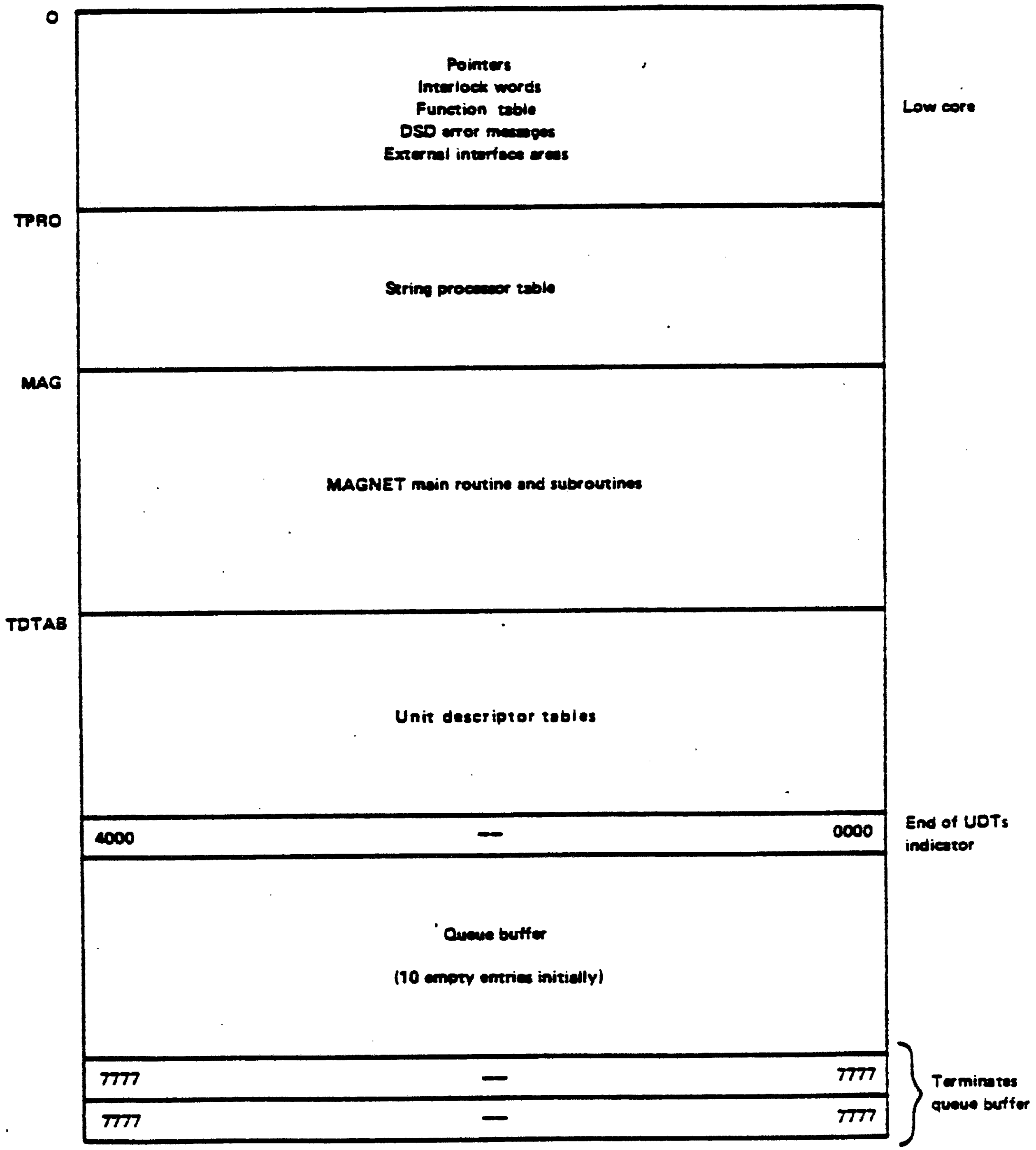
BLOCK COUNT

HEADER

LABEL IDENTIFIER
LABEL NUMBER
FILE IDENTIFIER
SET IDENTIFICATION
FILE SECTION NUMBER
FILE SEQUENCE NUMBER
GENERATION NUMBER
GENERATION VERSION NUMBER
CREATION DATE
EXPIRATION DATE
FILE ACCESSIBILITY
SYSTEM CODE

BLOCK COUNT

OVERVIEW OF MAGNET AFTER INITIALIZATION



DETAILED MAP OF MAGNET LOW CORE

		59	47	35	23	11 0	
RA+3	UBUF	Ptr. first UDT to display	0	lwa UDT	0	fwa UDT	
+4	UQUE	no. units	▨	▨	fwa of queue		
+5	USWP	unit swap flag				UDT addr next VSN	
+6	XREQ	external request buffer					2-word buffer
+7							
+10	URIW	interlocked request word					
+11	CHAN	channel status word				0017	
+12	NTAS	number of tape units currently assigned					
+13	UITW	1MT		1			MAGNET – 1MT interlock words 2 words/ channel 4 channels maximum
+14		1	active flag				
+15		1MT		3			
+16		1	active flag				
+17		1MT		5			
+20		1	active flag				
+21		1MT		7			
+22		1	active flag				
+23		1	end of table				Indicates end of table
+24		1					
+25	FLSW	field length status word					
+26	TMGF	monitoring channel	tape channel index	monitoring 1MT IR address	0	monitored UDT address	
+27 • • • +36	RCAL	RESEX request block buffer					10B words
+37	TFUN	1MT function table					14B words
+40 • • • +50							
+51 • • +64	DMBF	DSD message buffer					14B words
+65 • • +100		▨					
+101 • • +177		preview display buffer					77B words
+200	TPRO	String processor table					

UNIT DESCRIPTOR TABLE FORMAT

		59	53	47	35	29	23	17	11	5	0	
0	UXRQ	rs		fn	mode		ps		pb			
1	UCIA	codes		skip count			FET address					3-word block sent by CIO
2	UCIB	a b external CIO code		FET options	level ▨	record/request/return information MLRS						
3	UCIC	FL		FIRST			LIMIT					
4	UST1	eq down dn	un	hp	error code		es		ds			
5	UST2	error iteration		wp	tape block count				user options			
6	UST3	last good record			error parameter (ep)				den	cv		
7	UST4	word count		ov	format		est ord	f noise	software options			
10	UST5	MTS/ATS detailed status										for 66X/67X units only
11	UST6	MTS/ATS detailed status					MTS/ATS format					
12	UST7	ATS format		ATS unit status								
13	UST8	ATS unit status			block - ID window							
14	UST9	block-ID window										
15	UBLC	blocks read accumulator			blocks written accumulator				blocks skipped			

UNIT DESCRIPTOR TABLE FORMAT (Continued)

59 53 47 35 29 23 17 11 5 0

16	ULRQ	MAGNET last request										
17	UREQ	req	shift	proc addr	B2	B3	X5					
20	UFLA	stack	stack flag									
21	UJSQ	job sequence number	control point no.	VSN index	VSN random index							
22	UBJN	job name	ot	next UDT to display ptr.								
23	UUFN	user number	fam. index	est. writ.	vac							
24	UVSN	vsn	lc	sv	cw	dm	ev	ec	dl	le	mf	reel number
25	UFID	file identifier										
26	UFSN	file identifier (continued)	file section number									
27	USID	set identifier	fac	file sequence number								
30	UGNU	▨	generation version number	generation number								
31	UDAT	creation date	expiration date									

PROC MACRO

THE PROC MACRO DEFINES A "QUEUE" OF SUBFUNCTIONS THAT NEED TO BE PROCESSED TO COMPLETE A TAPE OPERATION. THE "QUEUE" MAY CONSIST OF OPERATIONS THAT MAY BE PERFORMED BY MAGNET, 1MT OR BOTH, OR OTHER PROCEDURE QUEUES.

THE POSITION OF THE QUEUE IS KEPT IN THE UDT.

FOR EXAMPLE,

```
                         PROCESS EOI
PEOI           PROC           (EOI, FET, CET)

                      WRITE MULTI-FILE HDR1
PWHL           PROC           (PTM, PWHD, PWTL, FET1)
```

PREVIEW

MAGNET MAINTAINS BUFFER SPACE FOR THE PREVIEW DISPLAY.

RESEX DETERMINES THE CONTENT OF THE PREVIEW DISPLAY, COMPACTS THE DATA, AND SENDS IT TO MAGNET VIA SIC RA+1 CALL.

DSD E,P DISPLAY PROCESSOR EXTRACTS DATA FROM THE UDTS AND THE PREVIEW BUFFER AND "PAINTS" THE DISPLAY ON THE DISPLAY CONSOLE.

SINCE RESEX IS THE ACTUAL FORMATTOR OF THE PREVIEW DISPLAY, IT (E,P DISPLAY) IS ONLY UPDATED BY A CALL TO REXES. ANY CALL TO RESEX WILL CAUSE A FORMATTING OF THE PREVIEW BUFFER.

QUESTION SET LESSON 18

1. What information is kept in the Unit Descriptor Table (UDT)?

2. How do other system routines make requests to the tape executive MAGNET?

3. The buffer for the preview display (E,P) is kept in MAGNET's Field length. Who updates the information in that buffer?

4. Why do tapes have labels? What information may be found in required ANSI labels?

5. MAGNET/1MT process three different types of tape controllers. How does it handle the function code and operational differences between them?

LESSON 19
UNIT RECORD SUBSYSTEM (BATCHIO)

LESSON PREVIEW:

This lesson introduces the student to the unit record subsystem BATCHIO.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Identify the components of BATCHIO, and explain in detail the flow through 1IO, 1CD, and QAP.
- Discuss "buffer points".
- Describe the "driver control word" and "driver request word".
- Map BATCHIO's control point field length and the FET for active buffer points.

REFERENCES:

NOS IMS - Chapter 17 NOS Operator's Guide, 3 NOS IHB, 7-9-10 NOS SMRM, 9

BATCHIO

BATCHIO IS A SUBSYSTEM THAT PROCESSES DATA TRANSFERS BETWEEN THE MAINFRAME AND THE UNIT RECORD PERIPHERALS ATTACHED TO IT.

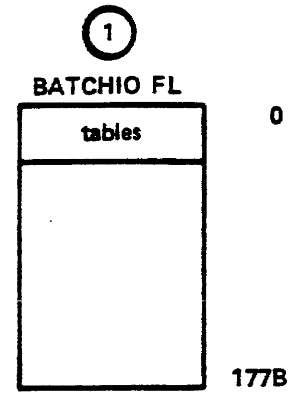
CARD READER	405		
CARD PUNCH	415		
LINE PRINTER	512	580	580PFC

- READS CARDS FROM CARD READER.
  CREATES INPUT FILE.
  ENTERS JOB (INPUT FILE) INTO QUEUE.
- LOCATES FILES IN PRINT OR PUNCH QUEUE AND DISPOSES THEM.
- PROCESSES OPERATOR COMMANDS FOR UNIT RECORD EQUIPMENT (FROM DSD) AND PROVIDES EQUIPMENT INFORMATION FOR THE I DISPLAY.

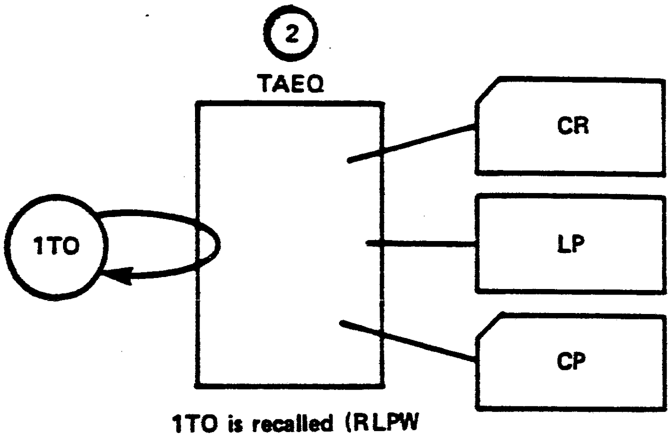
1IO - EXECUTIVE
1CD - PERIPHERAL DRIVER (CR, CP, LP)
QAP - AUXILIARY
COMSBIO - COMMON DECK

NO CPU CODE - FL USED FOR BUFFERS/FETS

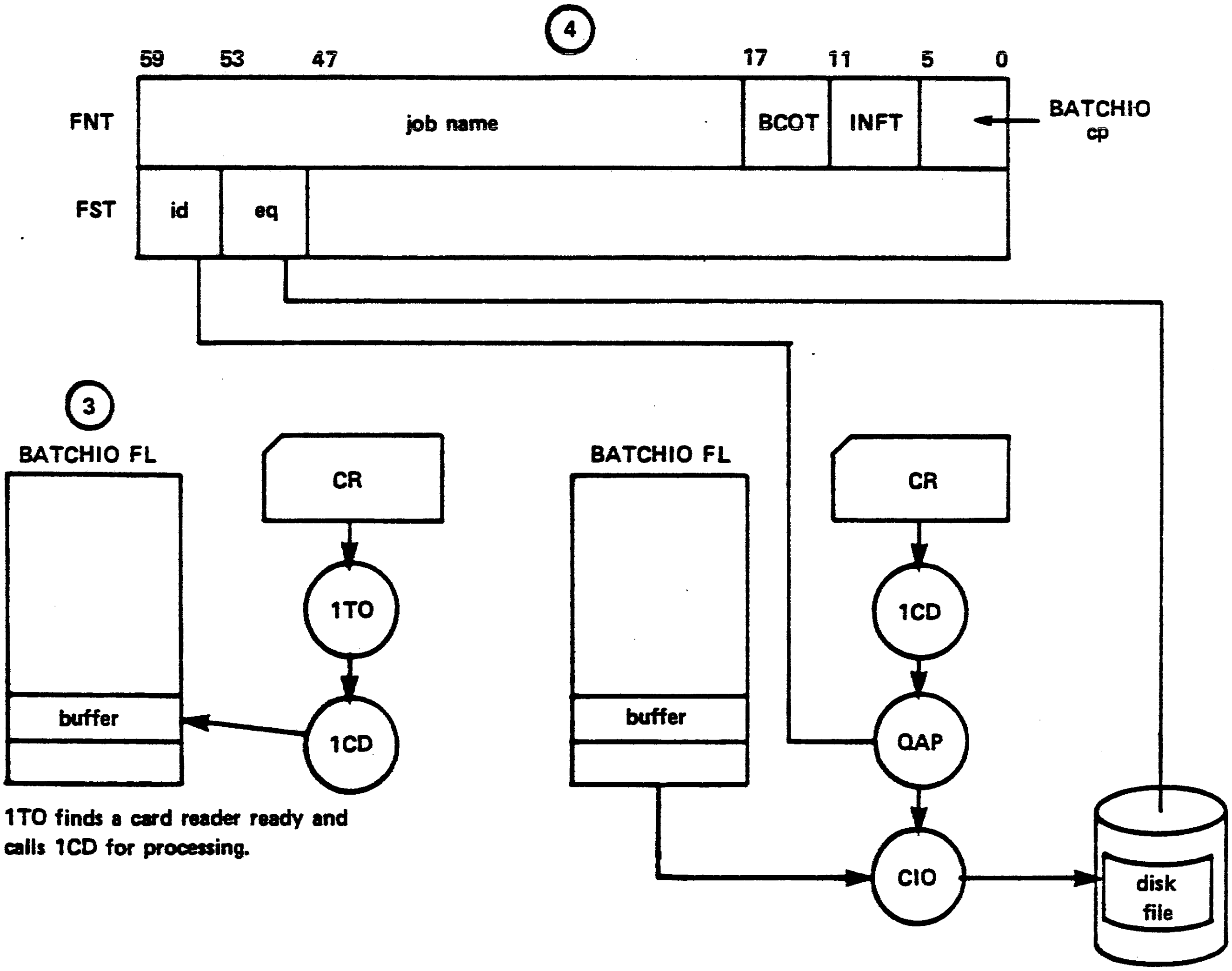
BATCHIO OVERVIEW



BATCHIO control point idle, FL is 200B.



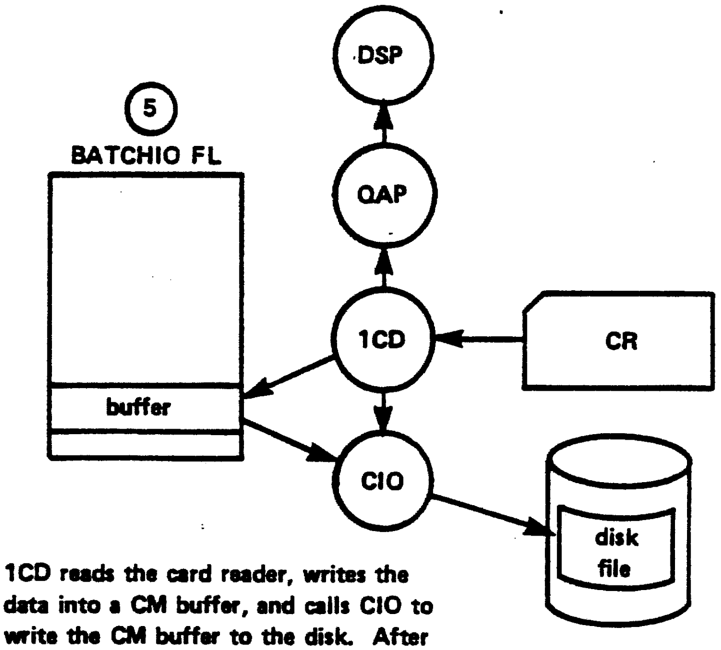
1TO is recalled (RLPW word in control point area) and begins scan of available unit record equipment.



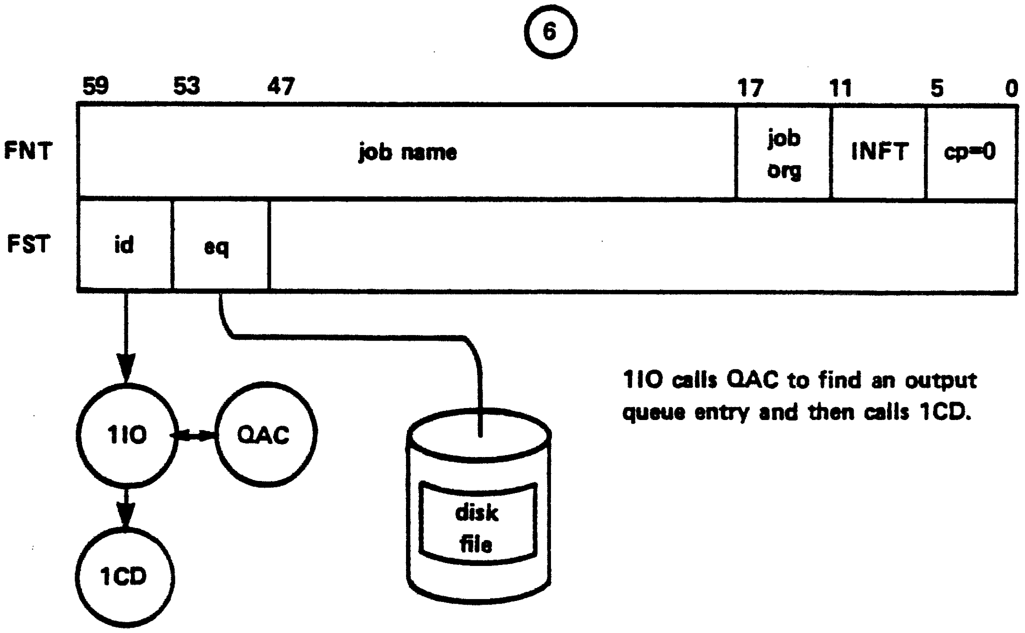
1TO finds a card reader ready and calls 1CD for processing.

1CD calls QAP to build an input file entry and write the data to the disk via CIO.

BATCHIO OVERVIEW (Continued)



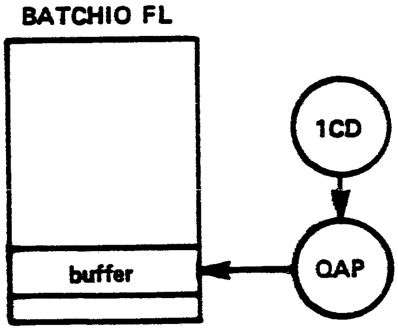
1CD reads the card reader, writes the data into a CM buffer, and calls CIO to write the CM buffer to the disk. After the file has been completely read, 1CD calls QAP who in turn calls DSP to put the file into the input queue.



1IO calls QAC to find an output queue entry and then calls 1CD.

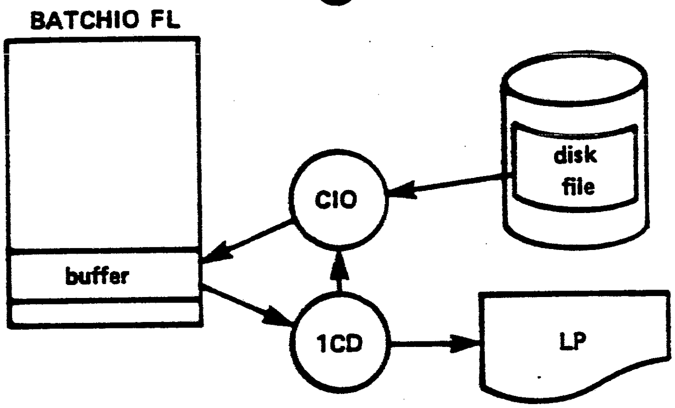
BATCHIO OVERVIEW (Continued)

⑦



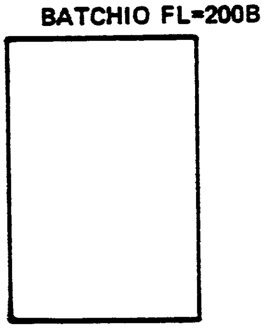
1CD calls QAP to create a banner page (QAP calls OBP) in the CM buffer.

⑧



1CD reads the rest of the file to the CM buffer via CIO and prints the data on the printer.

⑨



1CD completes and drops;
BATCHIO is now idle.

BATCHIO

TABLES AND BUFFERS ARE MAINTAINED IN BATCHIO CONTROL POINT AREA AND FIELD LENGTH FOR USE AND COMMUNICATION BY 1IO/1CD/QAP.

THE CONTROL CARD BUFFER (CSBW) IS USED FOR "BUFFER POINTS", TWO WORDS PER ENTRY FOR EACH PERIPHERAL. BEGINNING WITH NOS 1.3, THESE BUFFER POINTS ARE REFERENCED BY THE OUTSIDE WORLD BY THE EQUIPMENT NUMBER OF THE BUFFER POINT.

<table>
<tr><td colspan="4">FILENAME</td><td>EQ</td><td>RC</td><td>OR</td></tr>
<tr><td>I</td><td>DFM</td><td>MP1</td><td colspan="2">MP2</td><td colspan="2">MP3</td></tr>
</table>

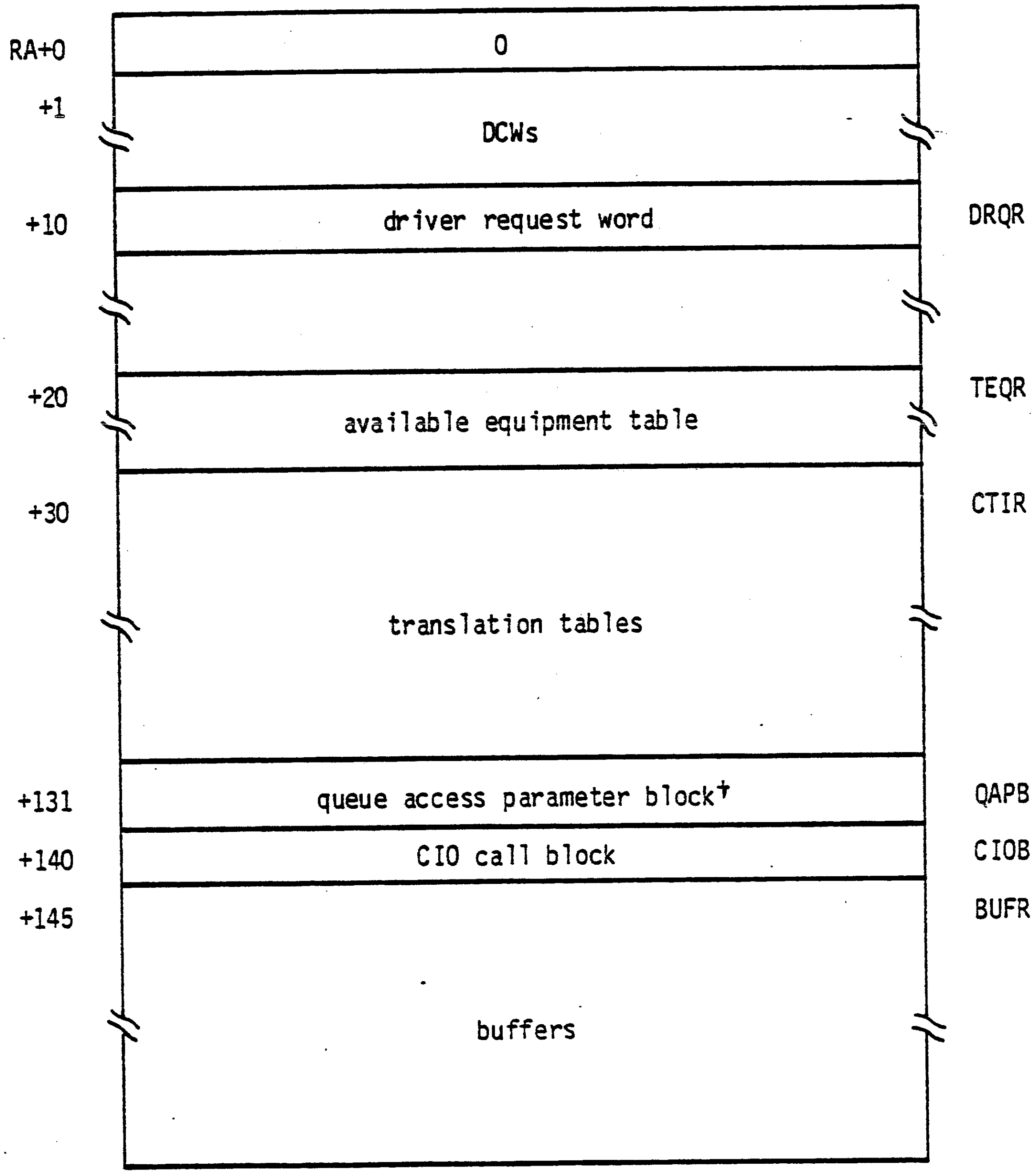
OR = OPERATOR REQUEST

1	END
2	REPEAT
3	SURPRESS
4	RERUN
5	HOLD
6	CONTINUE
7	BKSP PRU
10	BKSP RECORD
11	BKSP FILE
12	SKIP PRU
13	SKIP RECORD
14	SKIP FILE

RC = REPEAT COUNT
I = I DISPLAY MESSAGE CODE
DFM = DAYFILE/ERRLOG MESSAGE CODE
MP1/MP2/MP3 = MESSAGE PARAMETERS

BATCHIO CENTRAL MEMORY LAYOUT

The core Layout of the BATCHIO control point is shown below.



RA+0	0	
+1	DCWs	
+10	driver request word	DRQR
+20	available equipment table	TEQR
+30	translation tables	CTIR
+131	queue access parameter block†	QAPB
+140	CIO call block	CIOB
+145	buffers	BUFR

---

† Built by 1IO for QAC call to locate an unavailable qualifying file.

BATCHIO

FET IN BATCHIO FL

<table>
<tr><td></td><td colspan="4">FILENAME</td><td></td><td colspan="2">STATUS</td></tr>
<tr><td></td><td colspan="4"></td><td colspan="3">FIRST</td></tr>
<tr><td></td><td colspan="7">IN</td></tr>
<tr><td></td><td colspan="7">OUT</td></tr>
<tr><td></td><td>FST/ADDR</td><td colspan="3"></td><td colspan="3">LIMIT</td></tr>
<tr><td></td><td>BN</td><td>P1</td><td>P2</td><td colspan="3"></td><td>OT</td></tr>
<tr><td></td><td>CS</td><td>ES</td><td></td><td colspan="4">USER LIMITS</td></tr>
<tr><td></td><td colspan="7">Q A C PARAMETER BLOCK</td></tr>
<tr><td>FET+13</td><td colspan="7">USER NUMBER</td></tr>
<tr><td>FET+14</td><td colspan="7">JOB NAME</td></tr>
</table>

BUFFER FOLLOWS FET

BUFFER SIZE	1020_8	PRINT
	420_8	PUNCH
	1020_8	READER

BATCHIO

<u>DRIVER CONTROL WORD</u> DCW

ONE DCW FOR EACH COPY OF 1CD

EACH 1CD SUPPORTS UP TO EIGHT PERIPHERALS

1 C D	0	DCW INDEX	NO. OF REQUESTS	0

RA+2
↓
RA+7

<u>DRIVER REQUEST WORD</u> DRQR

REQUEST FOR 1CD TO DO SOMETHING

DCW INDEX	EST ORDINAL	BUFFER POINT	BUFFER ADDRESS

RA+10

BATCHIO

1IO EXECUTIVE

1 I 0	CP	F	RS	BN	AB

F	FILE PREVIOUSLY REQUESTED
RS	RELEASE STORAGE REPEAT COUNT
BN	CURRENT BUFFER POINT
AB	NUMBER OF ACTIVE BUFFERS

1IO WRITES ITS INPUT REGISTER (IR) INTO RPLW AND DROPS
1IO WILL BE RECALLED AT THE DELAY AR INTERVAL BY MTR.

1. BUILD A REQUEST FOR 1CD IN DRQR IF DCW EXISTS AND HAS LESS THAN 8 PERIPHERALS, ADD TO THAT DCW. OTHERWISE, MAKE A NEW DCW AND CALL 1CD OVER ITSELF AFTER SETTING UP RECALL (IR)→(RLPW).

2. CALL QAC FOR OUTPUT. CALLS QAC INTO THIS PP AFTER SETTING UP RECALL (IR)→(RLPW)

BATCHIO

1CD - COMBINED PERIPHERAL DRIVER

1CD CHECKS FOR REQUEST IN DRQR

- READER
  - READ FULL BUFFER OF CARDS (1020_8 WORDS) AND CALL QAP TO PROCESS INPUT FILE.
  - QAP CALLS OVJ TO PROCESS JOB CARD, CREATES INFT FNT/FST ENTRY.
  - 1CD KEEPS READING
  - QAP CALLS CIO TO WRITE DATA TO DISK
  - AT END OF READING, QAP IS CALLED TO CALL DSP AND PUT FILE INTO THE INPUT QUEUE WITH QUEUE SYSTEM SECTOR
  - ACCOUNTS FOR NUMBER OF CARDS READ (PASSED IN SYSTEM SECTOR)
- PRINTER/PUNCH
  - 1CD CALLS QAP TO GET BANNER PAGE OR LACE CARD (OBP) FORMATTED INTO BUFFER
  - QAP CALLS CIO TO READ THE OUTPUT FILE AND FILL THE BUFFER (1020_8 = PRINT, 420_8 = PUNCH, 2040_8 = 199 PRINT)
  - 1CD PRINTS/PUNCHES
  - QAP IS CALLED TO FILL BUFFER
  - AT END OF PRINTING/PUNCHING, QAP IS CALLED TO ACCOUNT FOR IT AND QAP CALLS DSP TO DECREMENT THE REPEAT COUNT (IF ANY)
  - IF REPEAT COUNT IS ZERO → QUIT. OTHERWISE PRINT/PUNCH FROM THE BEGINNING

QUIT: 1IO MAY RELEASE BUFFERS IF NO LONGER REQUIRED AND LAST BUFFERS

BATCHIO

QAP - QUEUE ACCESS PROCESSOR

PERFORM FUNCTIONS FOR BATCHIO AND RBF

	FUNCTION	SYMBOL
DAYFILE MESSAGES (ERROR DIAGNOSTICS)	400	PDMF
INITIATE INPUT FILE WITH WRITE	410	WTIF
INITIATE INPUT FILE WITH WRITE EOR	420	WRIF
INITIATE INPUT FILE WITH WRITE EOF	430	WFIF
CHANNEL ERROR CLEAN-UP	440	CECF
OPERATOR REQUESTS	450	CORF
REQUEUE FILE	460	RQFF
LOAD 580 PFC (INCLUDING USER ARRAYS)	470	PFCF
LOAD INITIAL PRINT DATA (BANNER PAGE)	500	GBPF
LOAD INITIAL PUNCH DATA (LACE CARD)	510	GLCF
LIMIT EXCEEDED	520	PLEF
ACCOUNTING	530	ACTF

PRINT TRAINS AND IMAGE MEMORY

SOFTWARE MAPPING OF PRINTER TRAIN SLUGS IS CALLED IMAGE MEMORY

OVERLAY (1IO)

5IA	595-1	512	SLUG
5IC	595-4, 595-5		MACRO
5IE	596-1	580	
5IG	596-4, 596-5		

580 PFC: PROGRAMMABLE FORMAT CONTROL ARRAYS

5BA	6 LINES PER INCH	SOFTWARE SPECIFIED
5BB	8 LINES PER INCH	"CARRIAGE TAPE"

QUESTION SET LESSON 19

1. What programs are involved in the BATCHIO subsystem?
2. How is the BATCHIO control point and its field length utilized?
3. What is happening when BATCHIO is in its "idle" state?
4. How does the unit record driver (1CD) do input/output to/from mass storage?
5. What does BATCHIO do to submit a job to the system?
6. How does BATCHIO utilize the ID field in an FST entry?
7. What function do the DCW (Driver Control Word) words serve?
8. How is the DRQR one word request stack utilized?
9. Can BATCHIO be rolled out? Why?

LESSON 20
FILE ROUTING AND QUEUE MANAGEMENT

LESSON PREVIEW:

This lesson introduces the student to the file routing and queue management features of NOS. To more fully understand this lesson, the student should review the ROUTE control statement and macro.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Explain what is meant by Queued File Terminal Addressing, Alternate Routings, Device Specification, Forms Control, External and Internal Characteristics, Deferred Route.
- Detail the queued file system sector.
- Discuss the operations performed by QAC (Queue Access Control).
- Discuss the proper technique for creating a queue file.

REFERENCES:

NOS IMS - Chapter 19 and 34 NOS RM, 1-2-4,6, 1-7-19,30,36-40, 2-7, 2-8

ASSIGNMENT:

Design, flowchart and code system utility programs to list the system sector data for a queued output file.

FILE ROUTING/QUEUE MANAGEMENT

QUEUE FILE INFT PRFT PHFT
S1FT S2FT S3FT
WITH CONTROL POINT IN FNT = 0

QUEUE FILE SYSTEM SECTOR

COMSSSE

TERMINAL ADDRESSING

FDSS	DESTINATION FAMILY	TO WHERE
DASS	DESTINATION USER	
DISS	DESTINATION INDEX	

DISS FOR BATCH = FILE ID

FDSS/DASS/DISS FOR EIOT/RBF DEFINE THE RJE TERMINAL FOR DISPOSAL

ALTERNATE ROUTINGS

USER SPECIFIES DESTINATION FAMILY AND USER NUMBER/INDEX

DEVICE SPECIFICATION

SPECIFIES DISPOSAL EQUIPMENT

DVSS	DEVICE CODE
ECSS	EXTERNAL CHARACTERISTICS
FCSS	FORMS CODE (00, AA-AF, OTHERS)

SYSTEM SECTOR FORMAT
Standard Format

<table>
<tr><th></th><th>59</th><th>47</th><th>35</th><th>23</th><th>11 0</th></tr>
<tr><td>000</td><td colspan="5">fnss</td></tr>
<tr><td>001</td><td>eqss</td><td>ftss</td><td>nsss</td><td>▨</td><td>fass</td></tr>
<tr><td>002</td><td colspan="5">dtss</td></tr>
<tr><td>003
.
.
007</td><td colspan="5">▨</td></tr>
<tr><td>010</td><td colspan="4">†1</td><td>jfss/scss</td></tr>
<tr><td>011</td><td>jcss/lcss</td><td>jess</td><td colspan="2">crss</td><td>▨</td></tr>
<tr><td>†2 012</td><td>rcss</td><td colspan="2">rtss</td><td>rbss</td><td>▨</td></tr>
<tr><td>013</td><td>otss</td><td>prss</td><td>miss</td><td colspan="2">flss</td></tr>
<tr><td>014</td><td>lcss</td><td>ecss</td><td>fcss</td><td>dvss</td><td>dcss</td></tr>
<tr><td>015</td><td colspan="5">dass</td></tr>
<tr><td>016</td><td colspan="4">fdss</td><td>odss</td></tr>
<tr><td>017</td><td colspan="2">diss</td><td colspan="3">▨</td></tr>
<tr><td>020</td><td colspan="5">fsss</td></tr>
<tr><td>021</td><td colspan="4">fmss</td><td>ooss</td></tr>
<tr><td>022</td><td colspan="5">acss</td></tr>
<tr><td>023</td><td colspan="5">cdss</td></tr>
<tr><td>024</td><td colspan="5">jnss</td></tr>
<tr><td>025</td><td colspan="5">ohss</td></tr>
<tr><td>026</td><td colspan="5">dhss</td></tr>
<tr><td>027</td><td colspan="5">frss</td></tr>
<tr><td>030
.
.
.
046</td><td colspan="5">vass</td></tr>
<tr><td>047
050</td><td colspan="5">reserved</td></tr>
<tr><td>051
.
.
.
062</td><td colspan="5">ubss
(user data block)</td></tr>
<tr><td>063
.
.
.
077</td><td colspan="5">▨</td></tr>
</table>

†1 For print/punch files, pfss (bits 47-36), rass (bits 35-12); for input files, jsss (bits 59-36), bits 35-24 unused, jtss (bits 23-12).

†2 For input files, bits 59-18 are defined as terminal name (tnss).

The following apply to all system sectors.

fnss	FNT entry.
eqss	Equipment number.
ftss	First track.
nsss	Next sector.
fass	Address of FST entry.
dtss	Last modification date and time (packed format).

The following apply to input files only.

jsss	Job sequence number.
jtss	Job time limit.
jfss	Job flags.
jcss	Job statement CM field length.
jess	Job statement ECS field length.
crss	Cards read.
tnss	Terminal name.

The following apply to print/punch files only.

pfss	Punch format.
rass	Random address of dayfile.
scss	Spacing code for 580 PFC support.
lcss	Lines or statement limit index.
rcss	Repeat count.
rtss	Random index.
rbss	Requeue number.

The following apply to all queued files.

otss	Origin type.
prss	Priority.
miss	Machine ID.
flss	File size (sectors/10_8).
icss	Internal characteristics.
ecss	External characteristics.
fcss	Forms code.
dvss	Device code.
dcss	NOS/BE device code.
dass	Destination user number.
fdss	Destination family name.
odss	Family ordinal of destination (future).
diss	Destination terminal identification (TID).
fsss	FST entry.
fmss	Family name of creator.
ooss	Family ordinal of creator (future).
acss	User number of creator.
cdss	Queued file creation date and time.
jnss	Job statement name.
ohss	Origination host name (future).
dhss	Destination host name (future).
frss	File routing control.
vass	Account file validation block.
ubss	User block.

FILE ROUTING/QUEUE MANAGEMENT

RETRIEVING QUEUE FILE

QAC

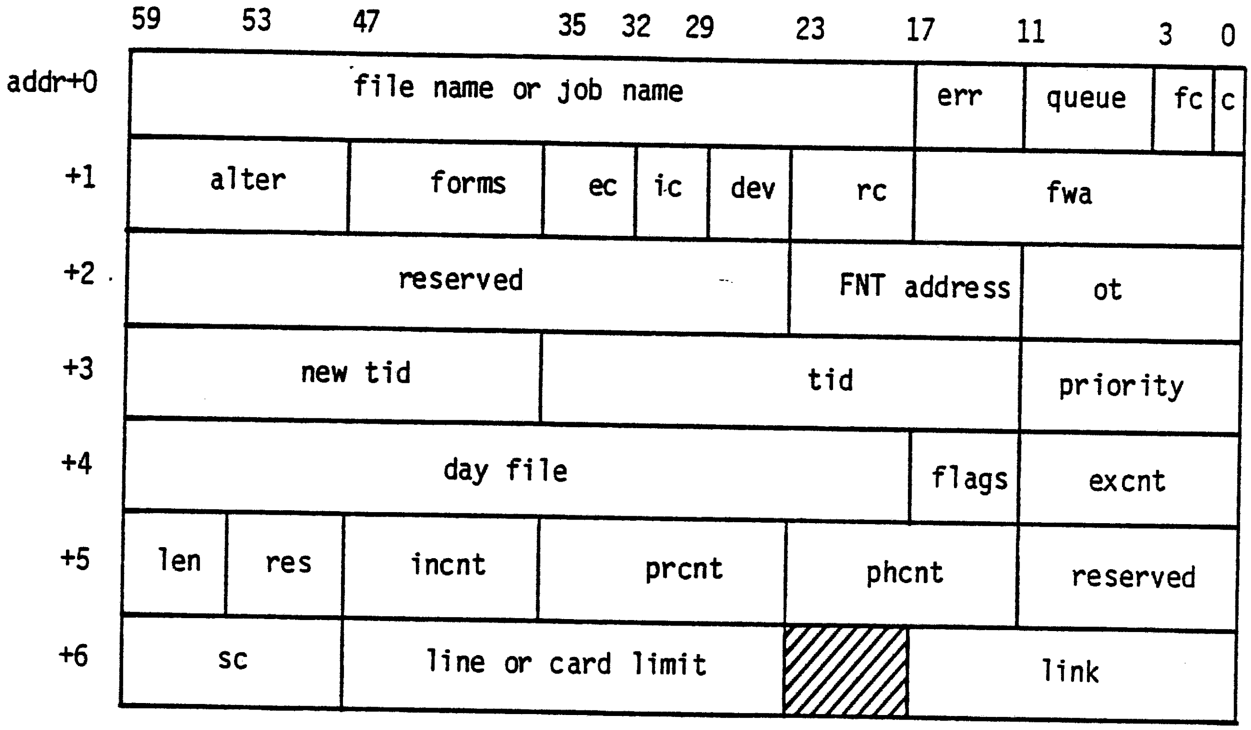
- IDENTIFY FILES FOR DISPOSAL
- ALTER FIELDS
- COUNT/PEEK

VALIDATE CONTROL POINT INFORMATION (VCI)
VALIDATE MASS STORAGE INFORMATION (VMI)

- CHECK FST FOR TID AND D, F, X
- READ SYSTEM SECTOR IF D, F, X EXTENDED CODES
- IF MATCH AND NOT EIOT ⟶ OK
- IF EIOT WHEN FAMILY MUST MATCH (READ SYSTEM SECTOR)

<table>
<tr><td colspan="6">JOBNAME</td><td>ORIGIN</td><td>QT</td><td>1</td><td>0</td><td>FNT</td></tr>
<tr><td>D</td><td>X</td><td>EQ</td><td>FT</td><td>F</td><td colspan="2">TID</td><td colspan="3">QP</td><td>FST</td></tr>
</table>

The parameter block format is as follows.



err Error code returned (complete description follows).

queue Queue type:

Queue	Description
1	Input file
2	Output file
4	Punch file
8	Special output
16	Executing job (at control point or in rollout queue)

fc Function code:

Code	Description
0	ALTER function
1	GET function
2	PEEK function
3	COUNT function

c Completion bit. Must be 0 on call; set to 1 upon completion.

FILE ROUTING/QUEUE MANAGEMENT

CREATING A QUEUE FILE

1. INPUT FILE BY BATCHIO/EI200/RBF

   FDSS/DASS/DISS FROM "CARD READER"
   DESTINATION FAMILY/USER/INDEX

2. QUEUE FILE AT THE "USER CONTROL POINT"

BY	ROUTE	1CJ	
	DISPOSE	CIO	CLOSE-UNLOAD
	SUBMIT		
	OUT		CLOSE-RETURN

THROUGH USE OF:

COMPUSS - UPDATE SYSTEM SECTOR (USS)
- WRITE QUEUED FILE S.S. (WQS)

DSP
- UPDATES SYSTEM SECTOR (USS)
- SETS APPROPRIATE FIELDS
- WRITES SYSTEM SECTOR (WQS)
- WRITE FNT/FST, PUTTING INTO QUEUE IF IMMEDIATE ROUTE. IF DEFERRED ROUTE, FILE IS NOT QUEUED (CONTROL POINT NOT 0) BUT SYSTEM SECTOR INFORMATION BIT (FNT BIT 5) IS SET. ROUTING INFORMATION WILL BE USED WHEN FILE DISPOSED TO QUEUE.

QUESTION SET LESSON 20

1. The most important item in queue management and file routing is the queued file's system sector. Why?

2. What common deck is used to generate the queued file system sector?

3. Explain what is meant by "terminal addressing". What FNT and/or system sector fields are used in terminal addressing?

4. There are three file routing properties that describe what is required of the peripheral device on which a file is to be disposed. Describe these three properties and explain their usage in the FST and system sector.

5. What is a deferred route?

6. Why is QAP (discussed in lesson 19) part of File Routing and Queue Management?

LESSON 21
OTHER MONITOR (RA+1) REQUESTS

LESSON PREVIEW:

This optional lesson presents other RA+1 requests that are found throughout the NOS system. The requests are used by both the system programmer and the user COMPASS programmer and are satisfied by PP routines.

As with most NOS PP routines that are user callable through RA+1, these requests have a macro call and in most cases a corresponding control statement which, in turn, uses the macro.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Identify commonly used PP routines called by the end user or system programmer through RA+1 requests.

REFERENCES:

NOS Reference Manual, all of volume 2 NOS IMS - Chapters 10, 11, 19 and 30

OTHER RA+1 REQUESTS

CPM - CONTROL POINT MANAGER
LFM - LOCAL FILE MANAGER
QFM - QUEUE FILE MANAGER
SFM - SYSTEM FILE MANAGER
TLX - TERMINAL ACTION PROCESSOR
FDL - FAST DYNAMIC LOADER
PFU - PERMANENT FILE UTILITY
VEJ - VERIFY JOB FILE INFORMATION
RDM - REDEFINE MASS STORAGE
RPV - REPRIEVE CENTRAL PROGRAM
CVL - COMMON VALIDATION INTERFACE
ELM - ERROR LOG MESSAGES

STS - STATUS PROCESSOR
MSD - SDA/SIS MESSAGE GENERATOR
PFE - ALTER FUNCTION
ACE - ADVANCE CONTROL CARD
PRM - PERMISSION CHECK
CKP - CHECKPOINT
REQ - EQUIPMENT REQUEST
DMD - DUMP WITH DISPLAY CODE
DMP - DUMP FL
DOO - ERROR TEXT PROCESSOR

SFP

CPM

SETQP
SETPR
SETASL
SETJSL
SETTL
SETJCR
SETSS
SETLC
SETGLS
SETLOF
SETJCI
SETUI
SETRFL
SETMFL
SETSSM

MODE
EREXIT
CONSOLE
ROLLOUT
ONSW
OFFSW
USECPU
PACKNAM

GETQP
GETPR
GETASL
GETJSL
GETTL
GETJCR
GETSS
GETLC
GETGLS
GETLOF
GETACT
GETEM
GETJO
GETJA
GETFLC
GETPFP
GETJCI
GETJN
USERNUM

MACHID
VERSION

PROTECT

DISSJ
DISSR RENSR
EESET
ENFAM

VALID
SETOV

LFM

RENAME
ASSIGN
COMMON
RELEASE
LOCK
UNLOCK
STATUS
FILINFO
REQUEST
LABEL
SETID
ENCSF
PSCSF
GETFNT
PRIMARY

ACCSF

SFM

DAYFILE

- USER
- SYSTEM
- ACCOUNT
- ERRLOG

ESYF

RDVF

DFAT

ENFA

QFM

RERUN
NORERUN
SUBMIT
FORCE EQUIPMENT

VEJ

- VERIFYJ

SFP

- CHECKPT

TLX

CSET
DISTC
PARITY
TSTATUS

LESSON 22
SYSTEM CONTROL POINT FACILITY

LESSON PREVIEW:

This lesson details the internal interfaces of the System Control Point Facility and the role it plays in NOS.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Discuss the requirements necessary for using the System Control Point Facility.
- Identify the components that make up the System Control Point Facility.
- Describe the interfaces between a system control point (subsystem) and a user control point.
- Explain the significance of the subsystem queue priority relative to using the System Control Point Facility.

REFERENCES:

NOS IMS - Chapter 18

SYSTEM CONTROL POINT FACILITY

CENTRALIZED FACILITY WHICH ALLOWS A MODULE OR GROUP OF MODULES TO BE USED BY ONE OR MORE JOBS RESIDING AT OTHER CONTROL POINTS.

PROVIDES -

- CENTRALIZED CONTROL
- REDUCTION OF CENTRAL MEMORY USAGE

SYSTEM CONTROL POINT FACILITY

DEFINITIONS

SUBSYSTEM - MODULE OR GROUP OF MODULES WHICH PERFORM A SPECIFIED SET OF FUNCTION. EXECUTES AT A CONTROL POINT WITH SPECIAL SYSTEM PRIVILEGES.

SYSTEM CONTROL POINT - CONTROL POINT OCCUPIED BY A SUBSYSTEM.

USER CONTROL POINT - JOB AT A CONTROL POINT WHICH MAKES A REQUEST TO A SUBSYSTEM. MAY BE A BATCH JOB OR A SUBSYSTEM.

SUBSYSTEMS

CHARACTERISTICS

1. NOT ROLLABLE - NO ROLLOUT
2. INTER CONTROL POINT COMMUNICATIONS (RSB/SIC)
3. HIGH CPU PRIORITY
4. DOES NOT ABIDE BY JCB CONTROLS CM AND CPU SLICES - INFINITE
5. MAY HAVE A USER NUMBER IN UIDW TRANEX/TAF
6. MAY RESIDE AT A SPECIFIED CP
7. IMPLICIT SSJ = AND ARE SYOT
8. USE OF SPC (NEE TLX) RA+1 CALL

TO BE A SUBSYSTEM

1. UNIQUE QP DEFINED TO FIT INTO SSCL WORDS OF CMR
2. 1DS TABLE ENTRY TO START UP EXECUTIVE
3. QP IDENTIFIES THE SUBSYSTEM (OPERATING SYSTEM TESTS FOR A GIVEN SUBSYSTEM BY ITS QUEUE PRIORITY)

SUBSYSTEMS

SUBSYSTEM	QP SYMBOL	QP	PP INIT.	CONTROL PT.
TIME-SHARING	TXPS	7776	1SI	1
TELEX				
IAF				
REMOTE BATCH	EIPS	7775	1LS	77
E1200				
UNIT RECORD	BIPS	7774	1IO	76
BATCHIO				
TAPES	MTPS	7773	1MT	75
MAGNET				
TRANSACTION	TRPS	7772		2
TRANEX			1TP	
TAF			1SI	
NETWORK INTERFACE	NMPS	7770	1SI	74
NIP				
REMOTE BATCH (NETWORK)	RBPS	7767	1SI	73
RBF				
MESSAGE CONTROL SYS	MCPS	7765	1SI	72
CDCS	CDPS	7766	1SI	71
TIME SHARE STIM	STPS	7771	1SI	77
STIMULA				
MASS STORAGE CONTROL	MSPS	7764	CMS	74
MSS STORAGE SUBSYSTEM	MFPS	7763	1SI	70
EXECUTIVE				
MSSEXEC				

	59	47	35	23 17	11 0	
042	†1					IPRL
043	†2					SSTL
044	TELEX/IAF	EXPORT/ IMPORT	BATCHIO	MAGNET	TAF	SSCL
045	STIMULATOR	NETWORK INTER PROC	RBF	CDCS	MCS	
046	MASS STORAGE CONTROL	TRANSACTION STIMULATOR	reserved			
047	reserved					
050	reserved				IR addr next PPU	PPAL
051	idle time					
052 053 054	load code for MS error processors					MSEL
055 056	reserved					
057	ctrl point for move	internal to MTR				CMCL
060	▨ ←†3	▨	CP0 ctrl pt assig	CP0 exchange address		ACPL
061	▨ ←†4	▨	CP1 ctrl pt assig	CP1 exchange address		
062	▨	▨	▨	address of PP0 exchange package		PXPP
063	first word of PP exchange package					
064 065	reserved					
	zeros					ZERL
067 • • • 075	reserved					
076	reserved			CPUMTR exchange address for MTR		MTRL
077	EQ	CPSL		PS	0	CPSL

SYSTEM CONTROL POINT FACILITY

CPUMTR - INSTALLATION SELECTABLE MODULE

- SSF RA+1 CALL - SUBSYSTEM
- SSC RA+1 CALL - USER CONTROL POINT

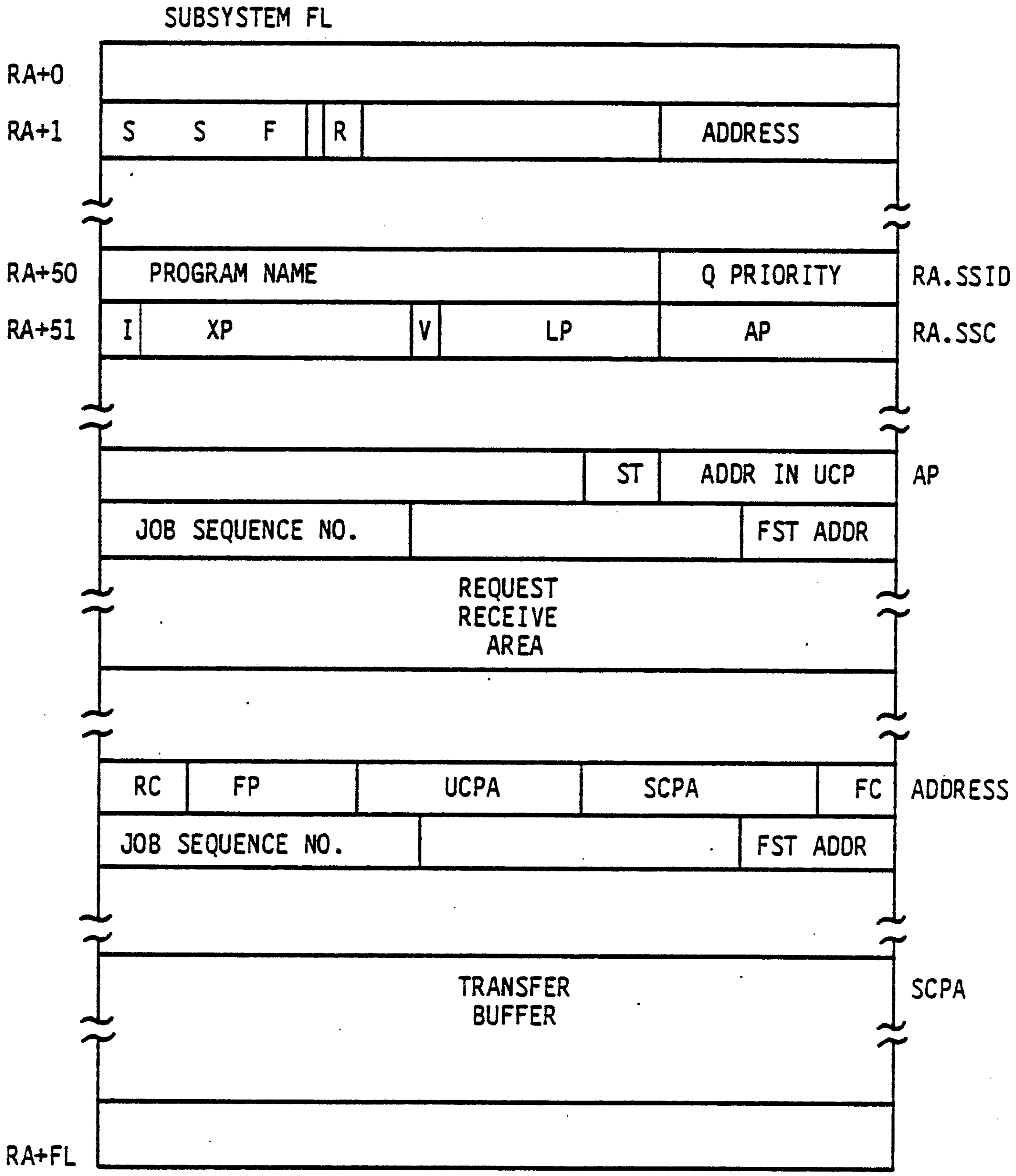
SSYTEXT

- MACRO INTERFACES
    - CALLSS - USER CONTROL POINT
    - SFCALL - SUBSYSTEM
- SYMBOL DEFINITIONS
    - SPECIAL WORD LOCATIONS
    - FUNCTION CODES

COMSSCP - SYSTEM CONTROL POINT EQUIVALENCES

- SPECIAL WORD LOCATIONS
- FUNCTION CODES
- REPLY CODES
- ERROR STATUS CODES

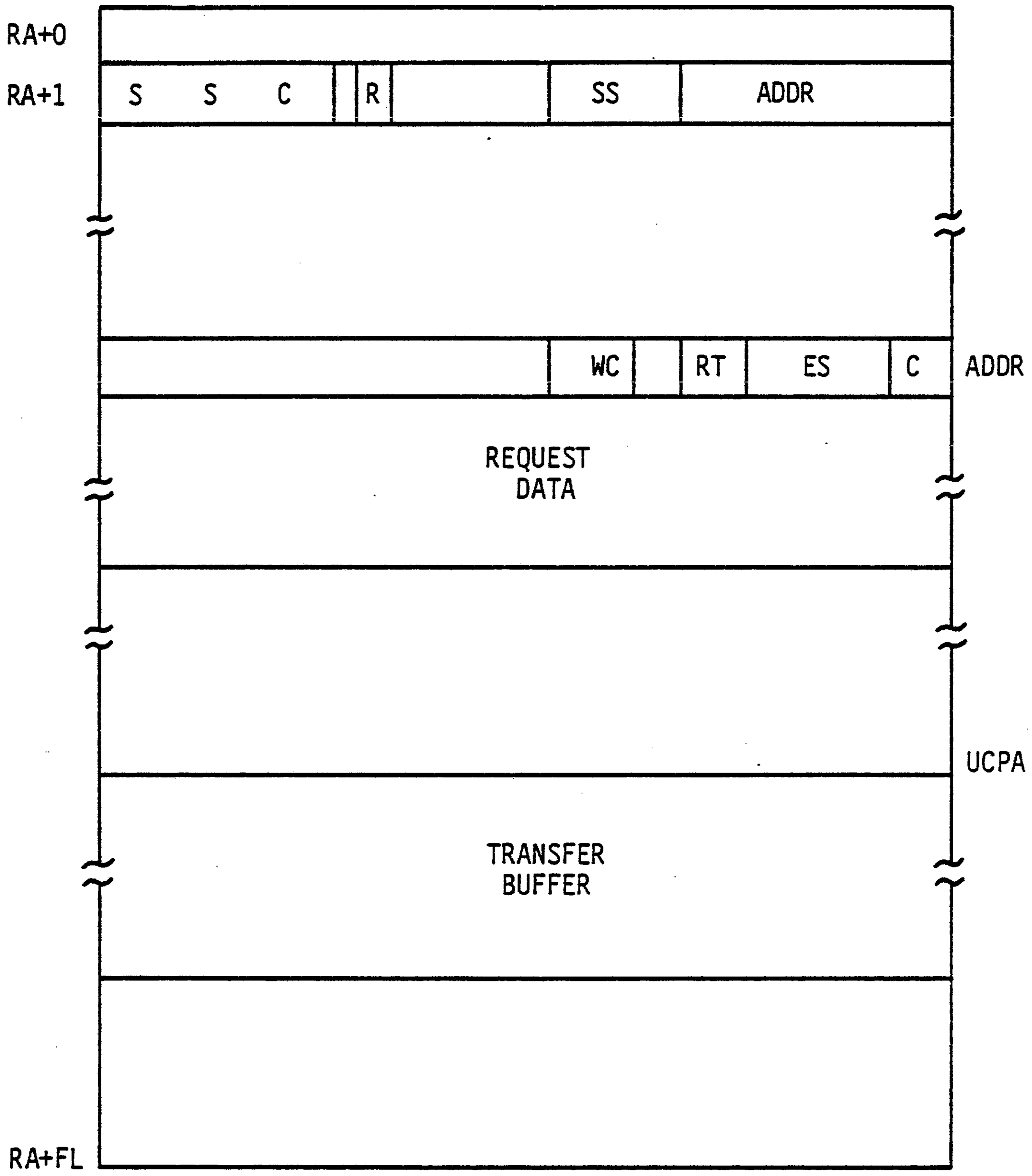
SYSTEM CONTROL POINT FACILITY



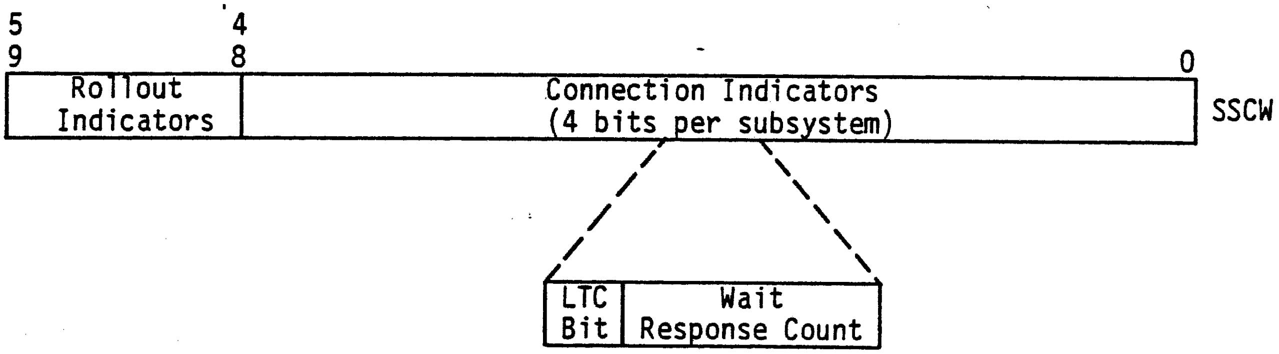
I = Request Present Flag (Set by (CPUMTR)
XP = UCP Exchange Package Address in Subsystem
V = Variable Transfer Flag
LP = Length of Parameter Area
ST = UCP Status (End/Abort/Override)
RC = Response Code
FP = Function Parameter
FC = SSF Function Code

SYSTEM CONTROL POINT FACILITY

USER CONTROL POINT



WC = Word Count -1 of Request
RT = Return Control (Busy/Non Fatal Error)
ES = Error and Status (Present/Busy/Illegal)
C = Completion Bit
SS = Subsystem Queue Priority



Subsystem Control Word in UCP
Control Point Area

Long-Term Connection	N	N	Y	Y
Wait Response Non-Zero	N	Y	N	Y
No Connection	X			
Awaiting Response		X		X
Lock In		X		X
Candidate for Rollout	X		X	
SF. SLTC Allowed		X		
SF. SWPO Allowed		X		X
SF. SWPI Allowed			X	
SF. CLTC Allowed			X	X

User Control Point
Connection State Table

SYSTEM CONTROL POINT FACILITY

SSF FUNCTION CODES

SF.REGR - MESSAGE IN UCP DAYFILE AND/OR ABORT UCP
SE.ENDT - COMPLETE USER REQUEST
SF.READ - READ UCP MEMORY
SF.STAT - REQUEST UCP STATUS
SF.WRIT - WRITE UCP MEMORY
SF.EXIT - EXIT SUBSYSTEM STATUS
SF.SWPO - ALLOW UCP SWAPOUT
SF.SWPI - REQUEST UCP SWAPIN
SF.SLTC - SET LONG TERM CONNECTION
SF.CLTC - CLEAR LONG TERM CONNECTION
SF.LIST - MULTIPLE REQUEST
SF.XRED - EXTENDED READ
SF.XWRT - EXTENDED WRITE
SF.XLST - EXTENDED MULTIPLE REQUEST

SYSTEM CONTROL POINT FACILITY

SSF RESPONSE CODES

00 - NO ERROR

40 - ERROR IN LIST REQUEST

41 - JOB IDENTIFIER INVALID

42 - SCPA OUT OF RANGE

43 - UCPA OUT OF RANGE

44 - UCP SWAPPED OUT

45 - UCP NOT IN SYSTEM

56 - ECS PARITY ERROR

57 - CONNECTION PREVIOUSLY ESTABLISHED

60 - CONNECTION REJECTED

61 - CONNECTION NOT PREVIOUSLY ESTABLISHED

62 - TRANSFER TOO LONG

63 - UCP NOT ESTABLISHED WITH SUBSYSTEM

64 - SUBSYSTEM ESTABLISHED WITH RECEIVER

65 - ATTEMPT TO SET ILLEGAL ERROR FLAG

66 - ILLEGAL DAYFILE PROCESSING FLAG

```
*NIP/SCP ERROR RC=rc  JOBID=jobid*
```

QUESTION SET LESSON 22

1. Explain the difference between a system control point, a user control point, and <u>THE</u> SYSTEM control point.

2. Describe the interfaces provided by the System Control Point Facility for transferring data between a system control point and a user control point.

LESSON 23
NETWORK PRODUCTS OVERVIEW

LESSON PREVIEW:

This lesson introduces the student to the Network Products and the terminology used within the Network Products context. This lesson is not a detailed discussion of Networks, but rather an overview of the concepts of NOS Network Products.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be familiar with the concepts and terminology of Network Products under NOS.

REFERENCES:

Network Access Method Reference Manual NAM Network Definition Language Reference Manual

NETWORK PRODUCTS

NAM NETWORK ACCESS METHOD

- SUPERVISORY MODULES

NS	NETWORK SUPERVISOR
CS	COMMUNICATION SUPERVISOR

- COMMUNICATIONS CONTROL

NPU	NETWORK PROCESSING UNIT (2550)
CCP	COMMUNICATIONS CONTROL PROGRAM
TIP	TERMINAL INTERFACE PROGRAM
HIP	HOST INTERFACE PROGRAM
LIP	LINK INTERFACE PROGRAM

- INTERFACE PROGRAMS

NIP	NETWORK INTERFACE PROGRAM
PIP	PERIPHERAL INTERFACE PROGRAM
AIP	APPLICATION INTERFACE PROGRAM

NETWORK PRODUCTS

NAM	NETWORK ACCESS METHOD

- VALIDATION

NVF	NETWORK VALIDATION FACILITY

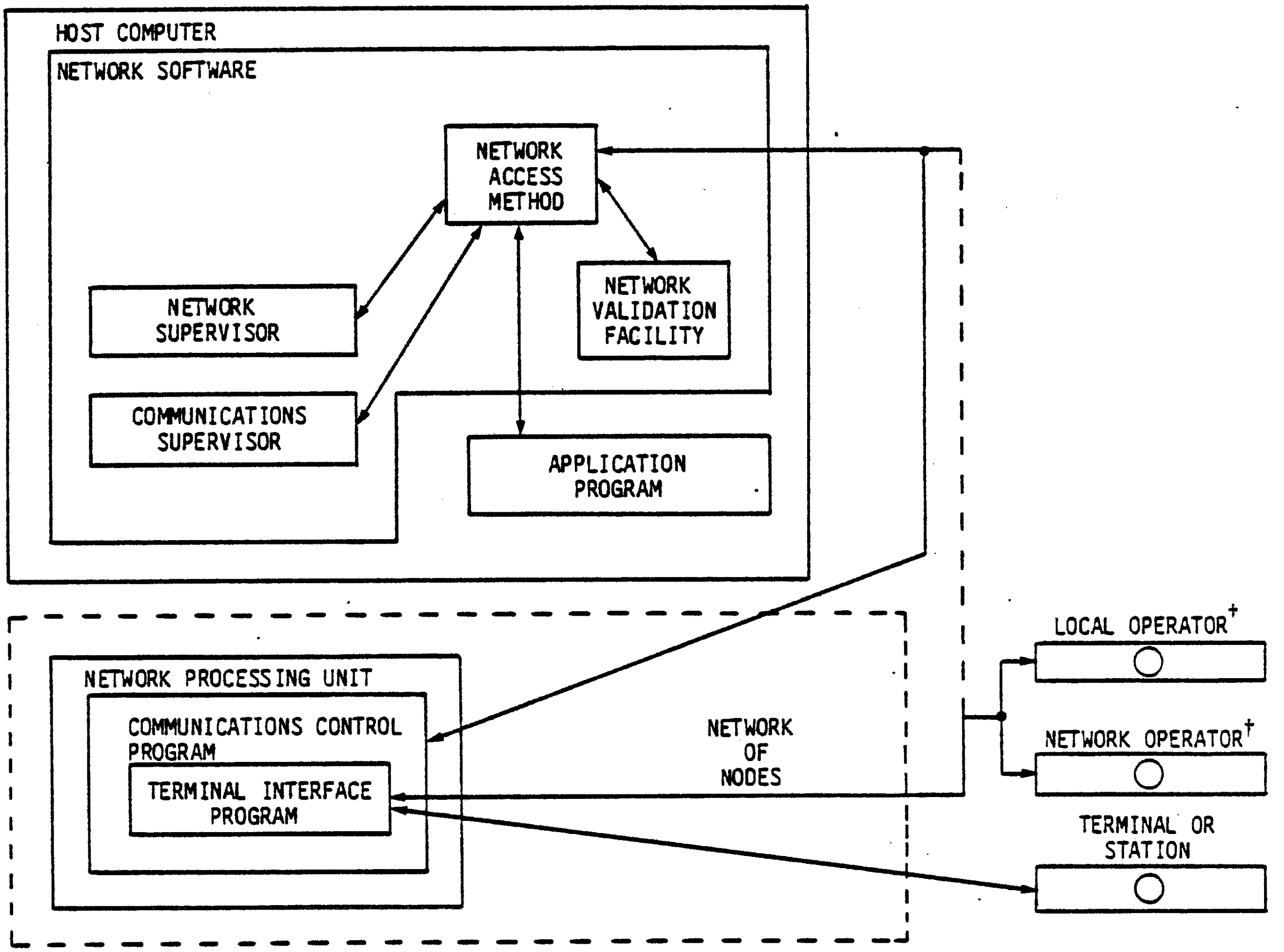
- APPLICATIONS

RBF	REMOTE BATCH FACILITY
TAF	TRANSACTION FACILITY
IAF	INTERACTIVE FACILITY
TVF	TERMINAL VALIDATION FACILITY

- UTILITIES

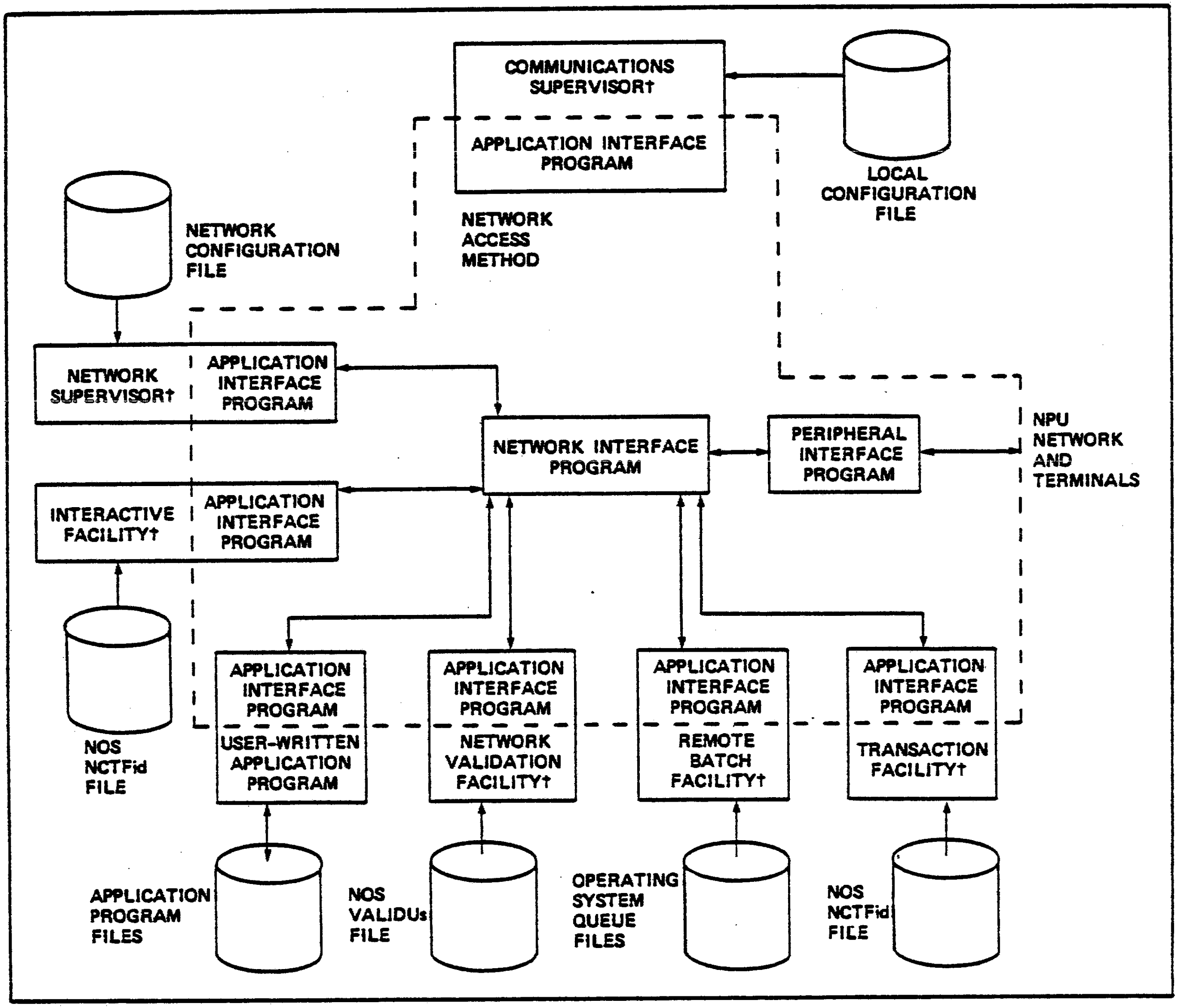
NDL	NETWORK DEFINITION LANGUAGE
NPS	NETWORK PRODUCT STIMULATOR
CROSS	PASCAL COMPILER FOR CCP
NDA	NETWORK DUMP ANALYZER

NETWORK SOFTWARE RELATIONSHIPS



†BOTH ADMINISTRATIVE OPERATORS CAN BE LOCATED AT THE SYSTEM CONSOLE AND COMMUNICATE WITH THE SUPERVISORS VIA DSD, RATHER THAN VIA THE NETWORK AND NAM.

NETWORK ACCESS METHOD COMPONENTS



TYPICAL APPLICATION PROGRAM PROCESSING FLOW

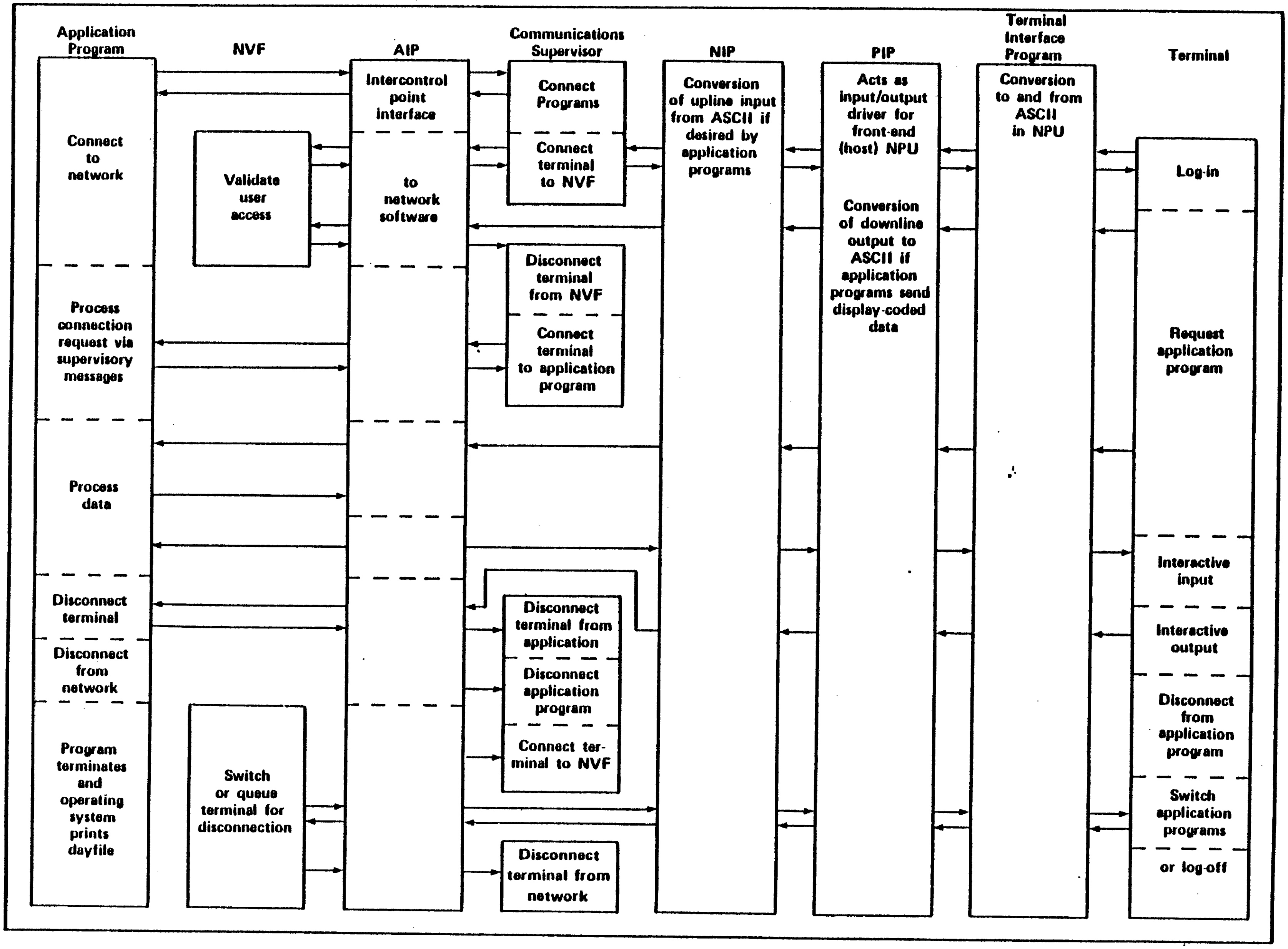


TABLE 1-1. SUPPORTED TERMINAL CLASSES†

Line Protocol	Terminal Class	Device and Archetype Terminal				
		Console	Card Reader	PLine Printer	Card Punch	Plotter
Asynchronous	1	M33				
	2	713				
	3	M1240				
	4	2741				
	5	M40				
	6	H2000				
	7	751††				
	8	T4014				
HASP Synchronous	9	HASP	HASP	HASP	HASP	HASP
Mode 4 Synchronous	10	200UT	200UT	200UT		
	11	214				
	12	711				
	13	714				
	14	731	200UT	200UT		
	15	734	200UT	200UT		

LESSON 24
TIME-SHARING SUBSYSTEM

LESSON PREVIEW:

This lesson introduces the student to the NOS time-sharing subsystem. NOS has two time-sharing subsystems: TELEX and the Interactive Facility (IAF). IAF is a successor to TELEX under the Network Products offering.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Identify the routines that compose the time-sharing subsystem and detail their major features.
- Briefly describe the initialization of the subsystem.
- Describe how the system does input/output with a time sharing terminal.
- Describe how the time-sharing subsystem brings terminal jobs into execution.
- Describe the processing of line numbered data.
- Detail the main loop of the executive program.
- List and detail the various internal queues managed by the time-sharing subsystem.
- Map the time-sharing control point field length allocation.
- Detail the terminal table.
- Describe the acquisition and releasing of POTs and their linkage.
- Discuss the re-entry concepts used in the time-sharing subsystem.
- Describe how the time-sharing executive makes requests for its PP auxiliaries.
- Explain why an output to a terminal is followed by an input from that terminal.

REFERENCES:

NOS IMS - Chapters 15 and 37

TELEX

EXECUTIVE - TELEX

TELEX	-	INITIALIZATION
TELEX1	-	WORKING EXECUTIVE
TELEX2	-	EXECUTIVE ERROR EXIT
TELEX3	-	TERMINATION

DRIVER - 1TD

2TD	-	INITIALIZATION
1TD	-	LOW-SPEED DRIVER 6676 6671 2550-100

AUXILIARY - 1TA

AUXILIARY FUNCTION PROCESSOR CALLED VIA TLX RA+1 CALL BY TELEX

- 1TO

I/O AUXILIARY CALLED WHEN TERMINAL
I/O REQUIRES MASS STORAGE ACTIVITY

- TLX

USER CONTROL RA+1 CALL

COMMON DECKS

COMSREM	-	TELEX SYSTEM PARAMETERS
COMSTCM	-	COMMUNICATION
COMSTOR	-	DRIVER

TELEX

INITIALIZATION

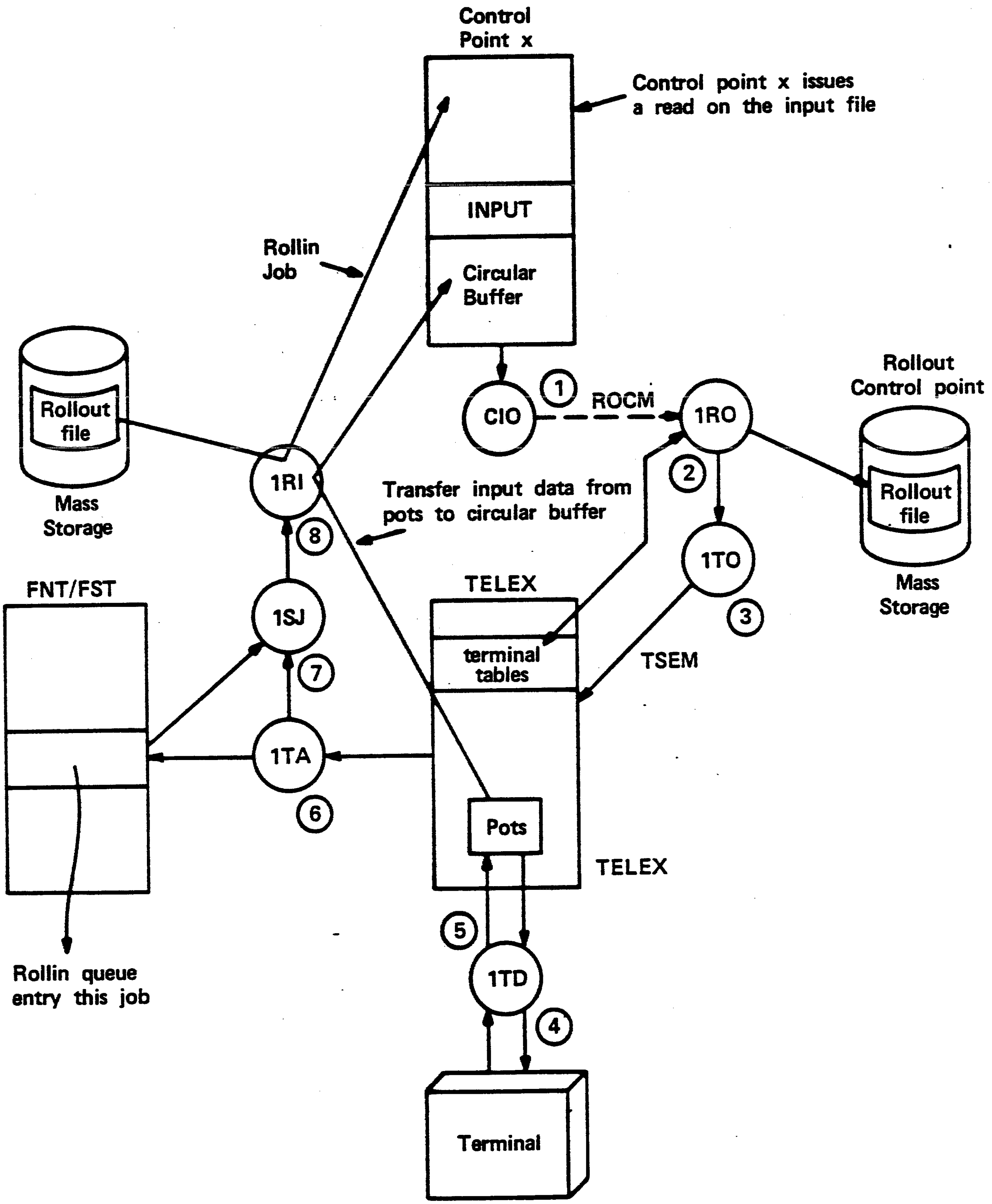
1. IDS SETS UP FNT/FST INPUT QUEUE ENTRY FOR 1SI AND CONTROL POINT 1
2. 1SJ/3SA SCHEDULES CONTROL POINT 1 AND EXECUTES PROCEDURE FILE SPECIFIED THROUGH 1SI
3. TELEX BEGINS EXECUTION IN CP (CP1) AND CALLS INITIALIZATION ROUTINE 1TD
4. NETWORK/SIMULATOR FILE READ AND TERMINAL TABLES BUILT
5. POT CHAIN BUILT BASED ON NUMBER OF TERMINALS
6. STATISTICAL ACCUMULATORS INITIALIZED
7. TELEX QUEUE AREAS INITIALIZED
8. ALL TERMINAL TABLES SET "COMPLETE"
9. HEADER ADDRESS IS SET IN WARN ADDRESS AREA
10. DRIVER QUEUE INITIALIZED
11. DRIVER STARTED AND CONTROL GIVEN TO TELEX1

TELEX

INPUT FROM A TTY

1. JOB ISSUES A READ REQUEST ON INPUT FILE - CALLS CIO. CIO SETS UP TINW AND DOES A ROCM - REQUEST ROLLOUT
2. 1RO DOES ROLLOUT AND CALLS 1TO. NO QUEUE ENTRY IS MADE FOR ROLLOUT FILE
3. 1TO ISSUES TSEM TO INFORM TELEX OF INPUT REQUEST
4. TELEX CALLS 1TD TO ISSUE PROMPT "?" TO USER AT TERMINAL
5. USER INPUTS DATA; 1TD STORES DATA IN POTs
6. WHEN CARRIAGE RETURN IS SENSED, TELEX CALLS 1TA TO REINITIATE THE JOB. 1TA BUILDS ROLLOUT QUEUE FNT/FST ENTRY
7. 1RI ROLLS IN JOB TO A CONTROL POINT AND TRANSFERS DATA FROM THE POTs TO THE JOB's CIRCULAR BUFFER
8. JOB IS GIVEN THE CPU AND CONTINUES EXECUTION . . .

TERMINAL JOB INTERACTION (INPUT)

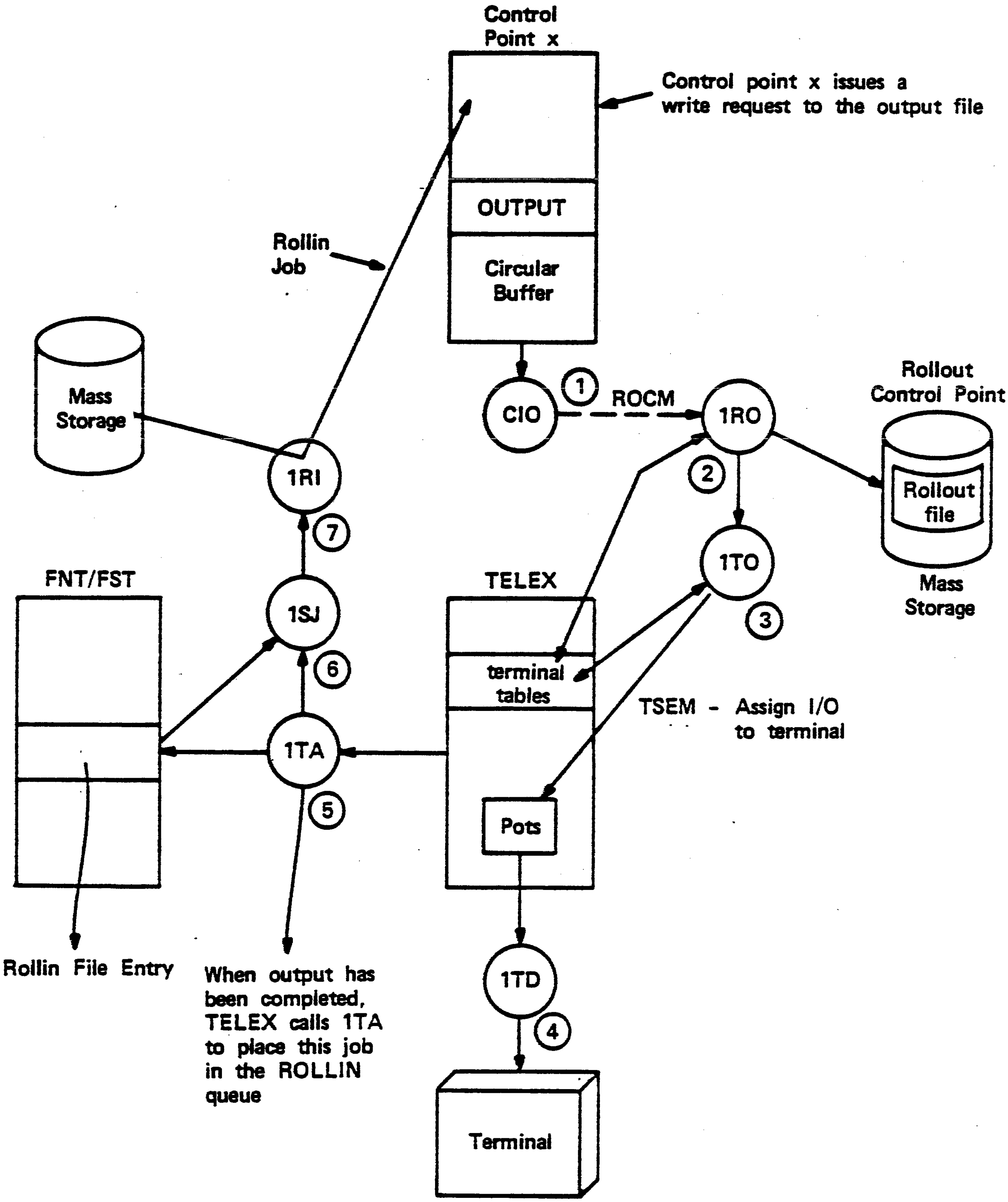


TELEX

OUTPUT TO A TTY

1. JOB ISSUES A WRITE REQUEST ON OUTPUT FILE - CALLS CIO. CIO SETS TIOW AND DOES A ROCM - REQUEST ROLLOUT
2. 1RO DOES THE ROLLOUT BUT MAKES NO ROLLOUT QUEUE ENTRY. 1RO PUTS THE FIRST SECTOR OF DATA INTO ITS PP MEMORY
3. 1RO CAUSES 1TO TO BE LOADED INTO ITS PPU
4. 1TO DOES A TGPM TO GET POTs FOR DATA. 1TO PUTS THE DATA INTO THE POTs AND CALLS TELEX VIA TSEM TO INDICATE OUTPUT IS AVAILABLE
5. TELEX CALLS 1TD TO TRANSMIT DATA IN THE POTs TO THE TTY. 1TD WILL ASK TELEX FOR MORE DATA AND TELEX WILL CALL 1TO TO GET IT UNTIL ALL DATA HAS BEEN TRANSFERRED
6. TELEX CALLS 1TA TO REINITIATE JOB AFTER ALL DATA HAS BEEN SENT. 1TA BUILDS A ROLLOUT QUEUE FNT/FST ENTRY
7. 1RI ROLLS JOB IN; JOB BECOMES A CANDIDATE FOR EXECUTION

TERMINAL JOB INTERACTION (OUTPUT)

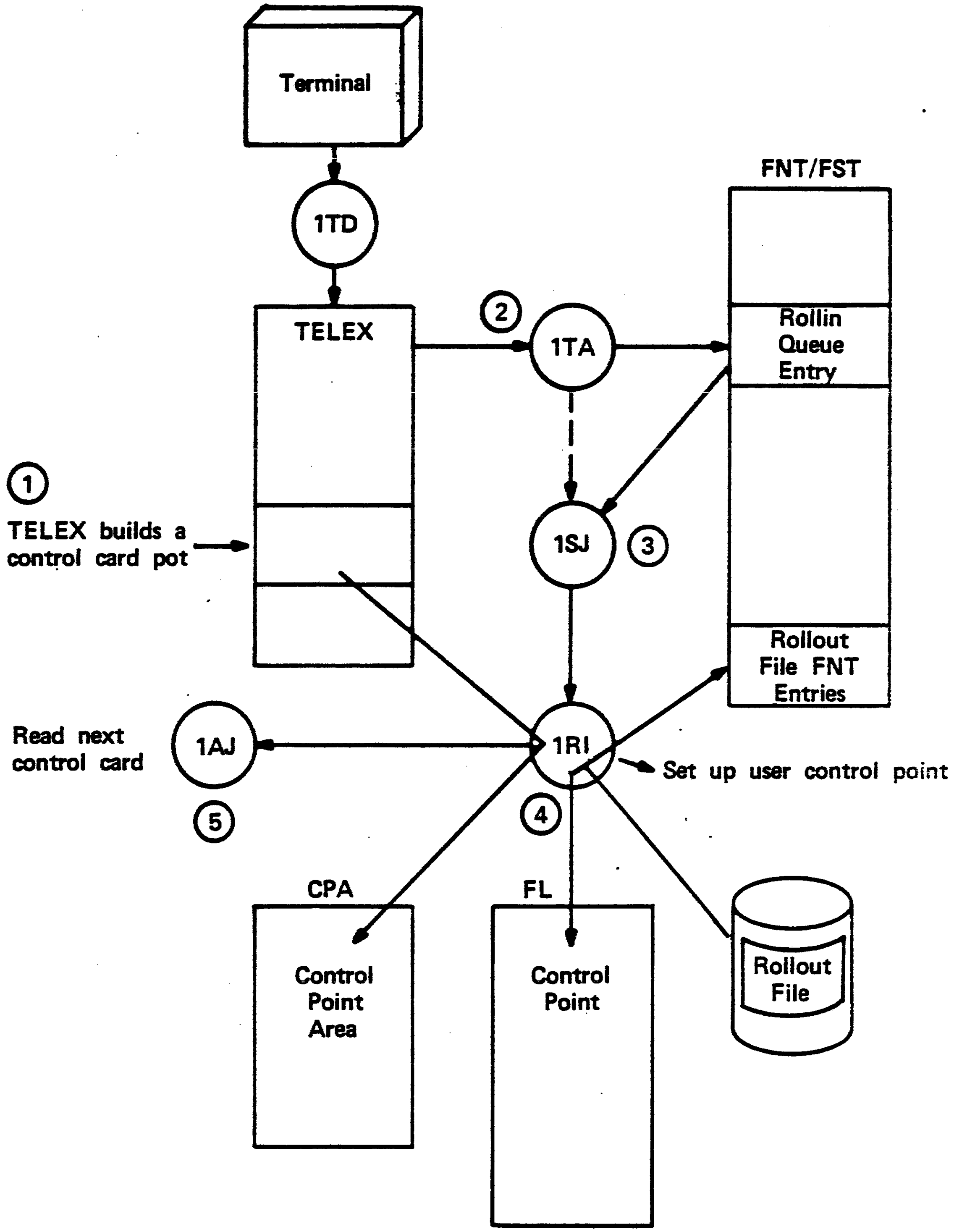


TELEX

JOB INITIATION (JOB STEP)

1. CONTROL STATEMENT PUT INTO POT BY TELEX
2. 1TA CALLED
3. 1TA BUILDS ROLLOUT ENTRY AND PUTS CONTROL STATEMENT INTO MS1W AREA OF CONTROL POINT AREA
4. 1RI ROLLS JOB IN
5. 1RI CALLS 1AJ TO ADVANCE JOB AND 1AJ WILL CALL TCS TO PROCESS THE CONTROL STATEMENT

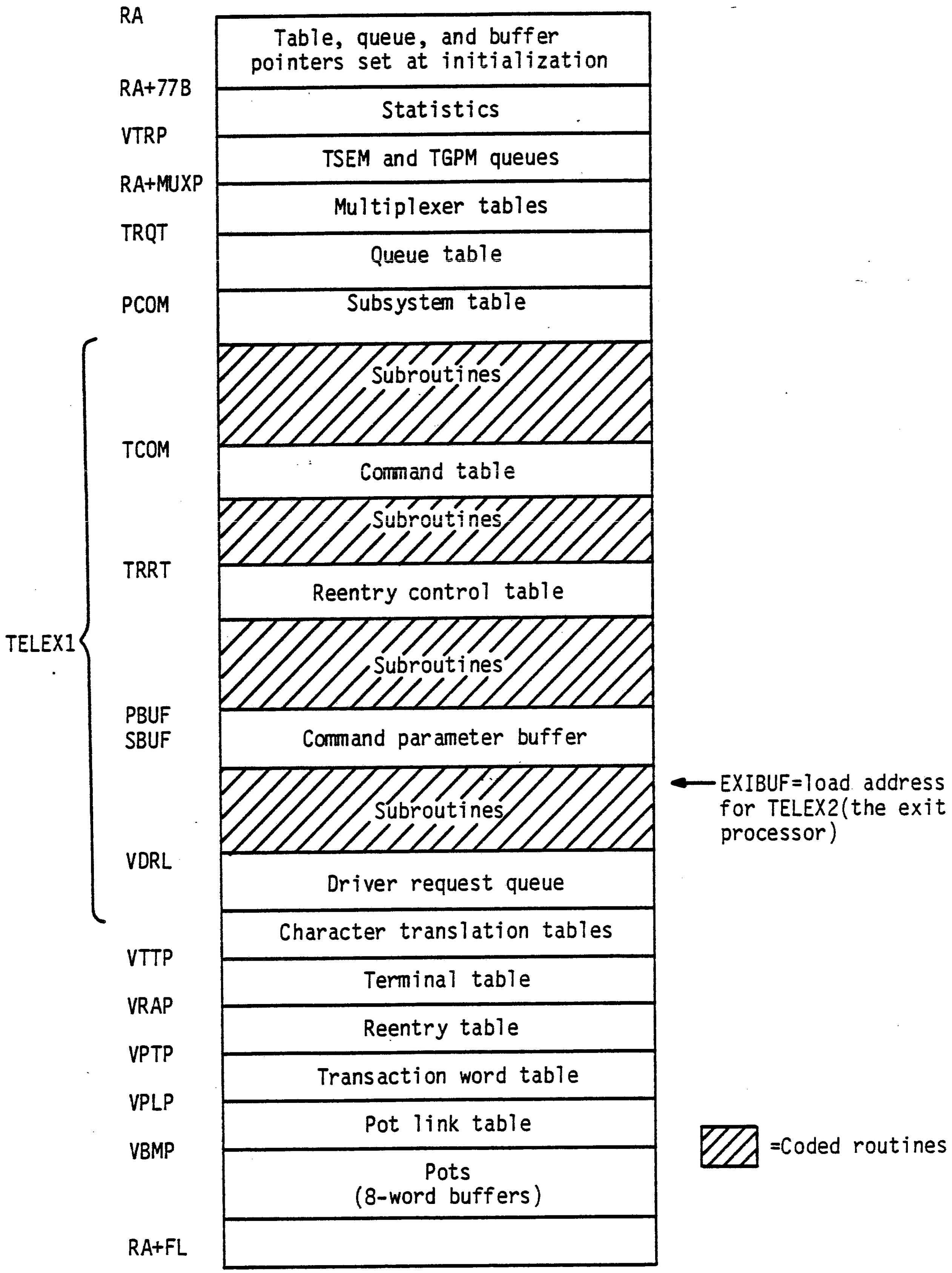
TERMINAL JOB INITIATION



TELEX

LINE NUMBERED DATA

1. USER ENTERS DATA AT TERMINAL
2. DATA BUILD CHARACTER BY CHARACTER AND ENTERED INTO POTs
3. WHEN VIPL POTs HAVE BEEN FILLED, 1TD ISSUES A REQUEST TO DUMP THE POTs (VIPL=2)
4. TELEX SENSES THE DUMP REQUEST AND CALLS 1TO TO DUMP THE POTs TO MASS STORAGE
5. 1TO DUMPS POTs TO ONE MS SECTOR
6. MEANWHILE, 1TD MAY BE FILLING MORE POTs
7. THIS CONTINUES UNTIL USER CAUSES A SORT TO BE DONE
8. MSORT (NOS's ONLY MTOT JOB) IS CALLED
9. MSORT DOES A SHELL SORT AND PACKS DATA INTO FULL SECTORS
10. 1RO SET TERMINAL INTO ACTIVE MODE
11. TELEX PROCESSES THE COMMAND THAT CAUSED THE SORT


RA
Table, queue, and buffer pointers set at initialization
RA+77B
Statistics
VTRP
TSEM and TGPM queues
RA+MUXP
Multiplexer tables
TRQT
Queue table
PCOM
Subsystem table
Subroutines
TCOM
Command table
Subroutines
TRRT
Reentry control table
TELEX1
Subroutines
PBUF
SBUF
Command parameter buffer
Subroutines
EXIBUF=load address for TELEX2(the exit processor)
VDRL
Driver request queue
Character translation tables
VTTP
Terminal table
VRAP
Reentry table
VPTP
Transaction word table
VPLP
Pot link table
VBMP
Pots (8-word buffers)
RA+FL
=Coded routines

POINTER ADDRESSES

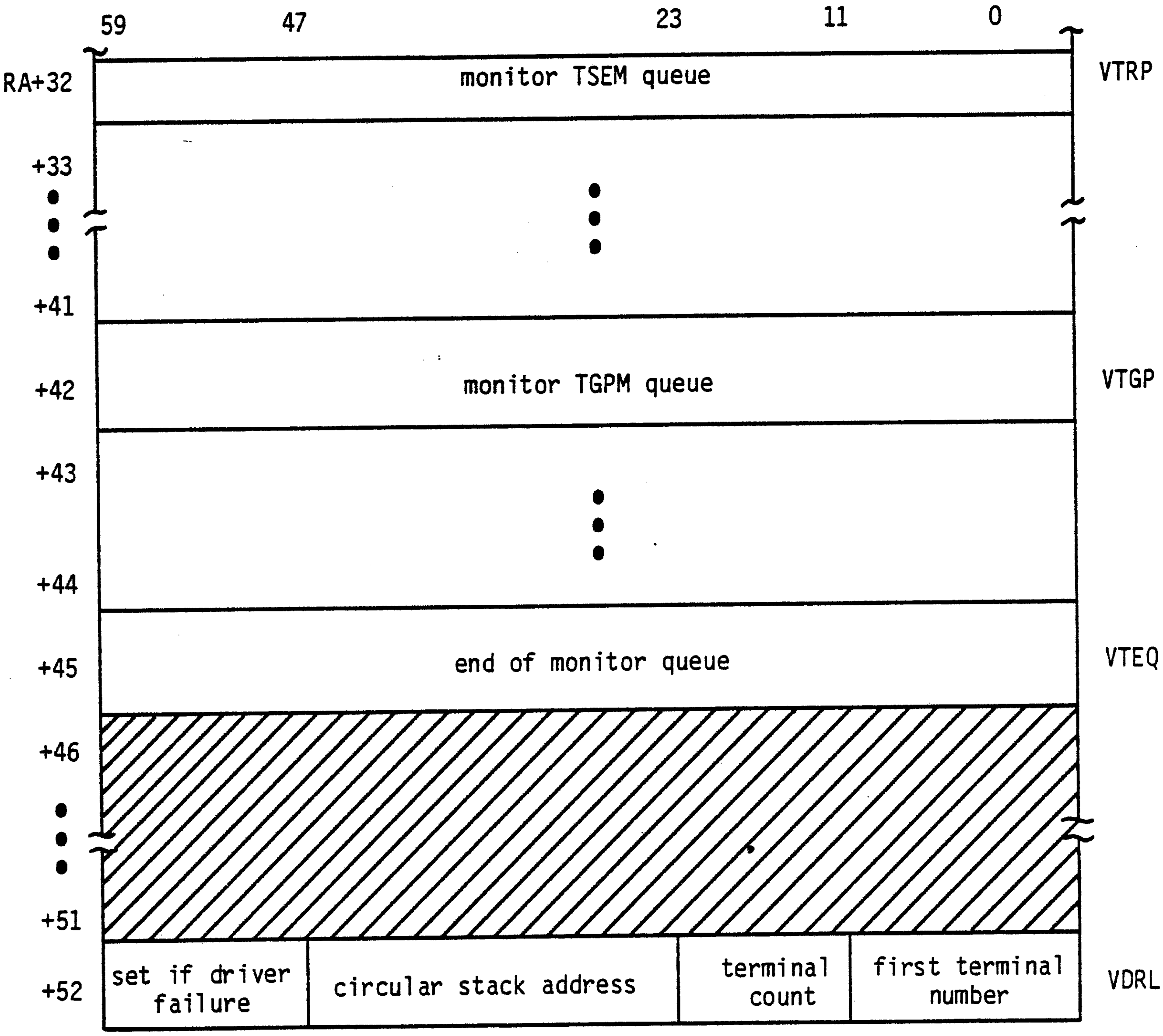
	59	47	41	35	23	17	11	0	
RA+3	▨		FWA terminal tables		▨	LWA+terminal tables			VTTP
+4	first network terminal number				last network terminal number				VNTP
+5	fwa message status table				lwa+1 message activity table				VMST
+6	fwa network activity table				lwa+1 network activity table				VNAT
+7	length pot link table	▨	FWA pot link table		▨	LWA+1 pot link table			VPLP
+10	▨		FWA command table		▨	LWA+1 command table			VCTP
+11	▨					FWA pot memory			VBMP
+12	▨		FWA warn message		▨	FWA header message			VWMP
+13	▨		FWA reentry table		▨	LWA+1 reentry table			VRAP
+14	▨		FWA transaction word table		▨	LWA transaction word table			VPTP
+15	▨		FWA transaction receive buffer		▨	FWA transaction send buffer			UTRN
+16	▨	1TD minimum cycle time		▨	move to byte 4 each cycle		if non zero driver stops *2		DBUG *1
+17	PFNL word								VFNL

*1 Driver debug word.
*2 Useful in debugging 1TD.

POINTER ADDRESSES (Continued)

<table>
<tr><th></th><th>59</th><th>47</th><th>23 0</th><th></th></tr>
<tr><td>RA+20</td><td colspan="3">number of times had to wait for PP</td><td>VPPL</td></tr>
<tr><td>+21</td><td colspan="3">total users since initialization</td><td>VTNL</td></tr>
<tr><td>+22</td><td colspan="3">current active user count location</td><td>VANL</td></tr>
<tr><td>+23</td><td colspan="3">maximum number of possible users</td><td>VMNL</td></tr>
<tr><td>+24</td><td colspan="3">new available pot count during FL change</td><td>VCPL</td></tr>
<tr><td>+25</td><td colspan="2">negative indicates no reload</td><td>real-time clock at last recovery</td><td>VRLL</td></tr>
<tr><td>+26</td><td colspan="3">abnormal occurrence counter</td><td>VABL</td></tr>
<tr><td>+27</td><td>▨</td><td>minimum number spare pots</td><td>maximum number spare pots</td><td>VPLL</td></tr>
<tr><td>+30</td><td colspan="3">number of pots allocated (available)</td><td>VPAL</td></tr>
<tr><td>+31</td><td colspan="3">number of pots in use</td><td>VPUL</td></tr>
</table>

POINTER ADDRESSES (Continued)



TELEX

TELEX MAIN LOOP

DRI	PROCESS DRIVER QUEUE (1TD REQUESTS)
STM	SEND TRANEX MESSAGES
URT	UPDATE RUNNING TIME
STR	PROCESS SYSTEM REQUESTS (TSEM)
RPC	REFILL POT CHAIN QUEUE (TGPM)
TDQ	PROCESS TIME DELAY QUEUE
CSF	CHECK SALVARE FILE
SOR	PROCESS SORT QUEUE
SCH	SCHEDULE JOBS (JOB QUEUE)
TSH	CHECK REQUEST COMPLETIONS (WAIT QUEUE)
PPU	PROCESS PPU REQUESTS
CTB	CHECK TRANSACTION BUFFER
SPR	CHECK FOR FL CHANGE (GET/RETURN POTs)
EPP	ENTER PPU REQUESTS
DRI	
STR	
RPC	
RECALL	
GO BACK TO TOP	

DRIVER REQUEST QUEUE

CIRCULAR STACK BUILT BY 1TD (SIMILAR TO A FET I/O BUFFER, 100 WORD QUEUE).

EACH REQUEST IS BIASED BY 2000_8 SO THAT TABLE INDEX IS EASILY RETRIEVED BY AN UNPACK INSTRUCTION UX1,B7 X2

B7 = REQUEST

X1 = ARGUMENTS/PARAMETERS

RQ+2000		P2	P1	TN

MONITOR REQUEST QUEUE

BUILT VIA TSEM FUNCTION (10 WORD QUEUE)

FN+2000	P1	P2	P3	P4

FUNCTIONS DEFINED IN COMSREM

POT REQUEST QUEUE

BUILT VIA TGPM FUNCTION (3 WORD QUEUE)

TERMINAL TABLE

EIGHT WORD ENTRY FOR EACH POSSIBLE ACTIVE USER (IE, ONE FOR EACH MUX PORT)

0 VUIT	USER NUMBER	USER INDEX

1 VFNT	PRIMARY FILE NAME	M	BFL

M = MODE 2^0 - WRITE LOCKOUT

2^2 - EXECUTE ONLY

BFL = BATCH FIELD LENGTH

2 VFST	EQ LIST	PRIM EQ	PRIM. FT	CT	CS	0

3 VROT	W_C	ROLL EQ	ROLLOUT FT		FL	SUB STATUS	STATUS

SUBSTATUS (1RI)
LLLLooFTIsss
LLLL=LEVEL
I=INTERRUPT
F=ROLLIN FL
T=TERMINATE SSJ
sss= 1 - EOR
2 - EOF
3 - EOI

STATUS	BIT=VALUE
TELEX IN CONTROL	0=1
SYSTEM IN CONTROL	0=0
JOB IN SYSTEM	1=0
JOB TO BE ROLLED IN	2=1
JOB WAITING INPUT	3=1
OUTPUT AVAILABLE	4=1
TRANEX OUTPUT	4=1
SSJ	5=1
LIST	6=1
MULTI TERMINAL	7=1
SUSPENDED	9=1
ERROR ON LAST OPERATION	11=1

TERMINAL TABLE

4	VDPT	FIRST POT	CURRENT POT	POT POSITION	CONTROL FLAGS	ROUTINE ADDR

POT POSITION

9-11	FIRST WORD IN POT
8	INPUT INITIATED IF = 1
7	NEXT INPUT POT REQUESTED IF = 1
4-6	CURRENT WORD IN POT
0-3	CHARACTER NUMBER IN WORD

CONTROL FLAGS

7-11	TRANSLATION TABLE INDEX
6	FULL DUPLEX
5	ERROR FLAG/UPPER CASE FLAG
4	MONITOR MODE
3	BINARY TRANSMISSION
2	TRANSPARENT INPUT
1	EXTENDED MODE
0	ODD PARITY

5	VCHT	BUFFER	CHR COUNT	SCRATCH	INPUT CC	OUTPUT CC

6	VDCT	FLAGS	TERM, CONTROL	AUTO MONITOR	ACCESS (AACW)	NEXT MESSAGE

TELEX

TERMINAL TABLE

7 VSTT	FLAGS	FIRST POT	COMM, INDEX		SUBS	QUEUED

FLAGS (BIT)

0	LOGOUT IN PROGRESS
1	LOGOUT ABORT
2	WARNING ISSUED
3	RUN COMPLETE MESSAGE
4	SORT FLAG
5	TIME/SRU LIMIT FLAG
6	JOB COMPLETE FLAG
7	INPUT LOST/JOB NOT STARTED
8	
9	CHARGE REQUIRED
10	
11	DISABLE TERMINAL CONTROL

SUBS = SUBSYSTEM

0	NULL
1	BASIC
2	TSRUN
3	FTNTS
4	EXECUTE
5	BATCH
6	ACCESS
7	TRANSACTION

TELEX

POT LINK TABLE

PLT CONTROLS POTS IN A MANNER SIMILAR TO HOW TRT CONTROLS TRACKS

PLP

0	1	2	3	4
7777	0002 1	0003 2	0004 3	0017
0005 4	0000 5	0007 6	0000 7	0017
0000 10	0012 11	0013 12	0014 13	0007
0000 14	0000 15	0000 16	0000 17	0010

$$\overbrace{0000\ 0000\ 0000}$$

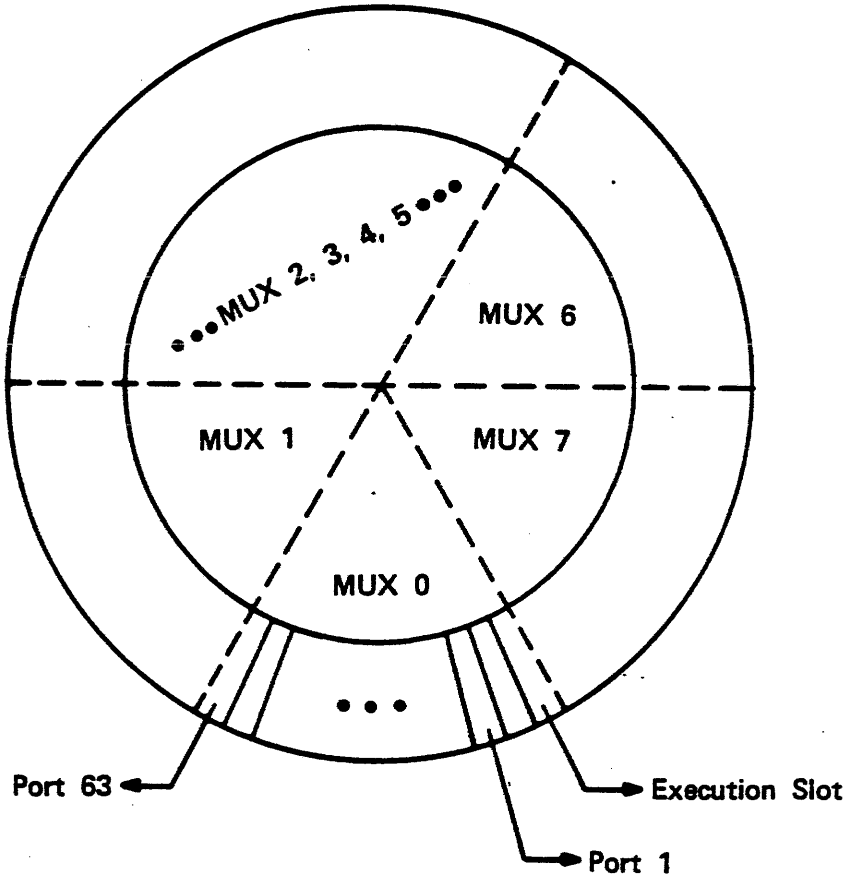
$$\underbrace{0123}$$

RESERVATION BITS

000 000 000 0 | 00
WORD | BYTE

- POT 0 ALWAYS 7777
- LAST POT IN CHAIN = 0000
- UNUSED POT ALSO = 0000 BUT NOT RESERVED

MULTIPLEXER SERVICING CONCEPT



TELEX

DRIVER

1TD

- CAPACITY - 5120 CHARACTERS/SEC
  (SIMULTANEOUS BURST)
  512 10 CPS TERMINALS
- HALF DUPLEX
- MOST ASCII COMPATIBLE TERMINALS
- TRANSLATION TABLES

AUXILIARY - PP ACTION FOR TELEX

1TA CALLED WITH LIST OF REQUESTS IN POTs OR SINGLE REQUEST

FUNCTION	DESCRIPTION
1	ADJUST TELEX FL
2	RETURN TERMINAL JOB
3	CREATE ROLLOUT FILE
4	LOGOUT TERMINAL
5	DISPLAY ACCOUNTING
6	TERMINAL RECOVERY
7	INCREMENT RESOURCE LIMIT (TL, SRU)
10	RECOVERY FILE PROCESSOR
11	SCHEDULE JOBS
12	GATHER STATISTICS
13	CLEAN UP SALVARE

TELEX

1TO AUXILIARY - TTY INPUT/OUTPUT

1TO DOES TERMINAL INPUT/OUTPUT THAT REQUIRES MASS STORAGE REQUESTS

RECOVERY FILE = SALVARE

- BUILT AT INITIALIZATION TIME
- TWO WORDS FOR EACH DEFINED TERMINAL

	EQ	FT	HH MM SS	UI

EQ = ROLLOUT FILE EQUIPMENT
FT = ROLLOUT FILE FIRST TRACK
HHMMSS = LAST ENTRY TIME
UI = USER INDEX

- ENTRIES CLEARED AFTER 10 MINUTES

- IN RECOVERY, BOI + EOI OF USER'S FILES ARE VALIDATED IF SENSE SWITCH SET

TELEX

RE-ENTRY

RE-ENTRY TABLE FOR RETURNING CONTROL OR HAVING FUNCTIONS PERFORMED WHEN A SET OF CONDITIONS ARE MET.

ONE WORD PER TERMINAL WITH INDEX TO TABLE OF RE-ENTRY PROCESSORS DEFINED VIA COMMAND MACRO

1TD RE-ENTRY

ENTRY POINTS WHICH ARE JUMPED TO BY ANY SUBROUTINE THAT CANNOT COMPLETE ITS TASK IN A SINGLE INTERNAL TIME SLICE

RETURN MACRO

IAF

EXECUTIVE - IAFEX

IAFEX - INITIALIZATION
IAFEX1 - WORKING EXECUTIVE
IAFEX2 - DUMP EXECUTIVE
IAFEX3 - TERMINATION
IAFEX4 - INTERFACE TO NAM

INITIALIZATION - 1TN

AUXILIARY - 1TA

AUXILIARY FUNCTION PROCESSOR CALLED VIA SPC RA+1 CALL BY IAF

- 1TO

I/O AUXILIARY CALLED WHEN TERMINAL
I/O REQUIRES MASS STORAGE ACTIVITY

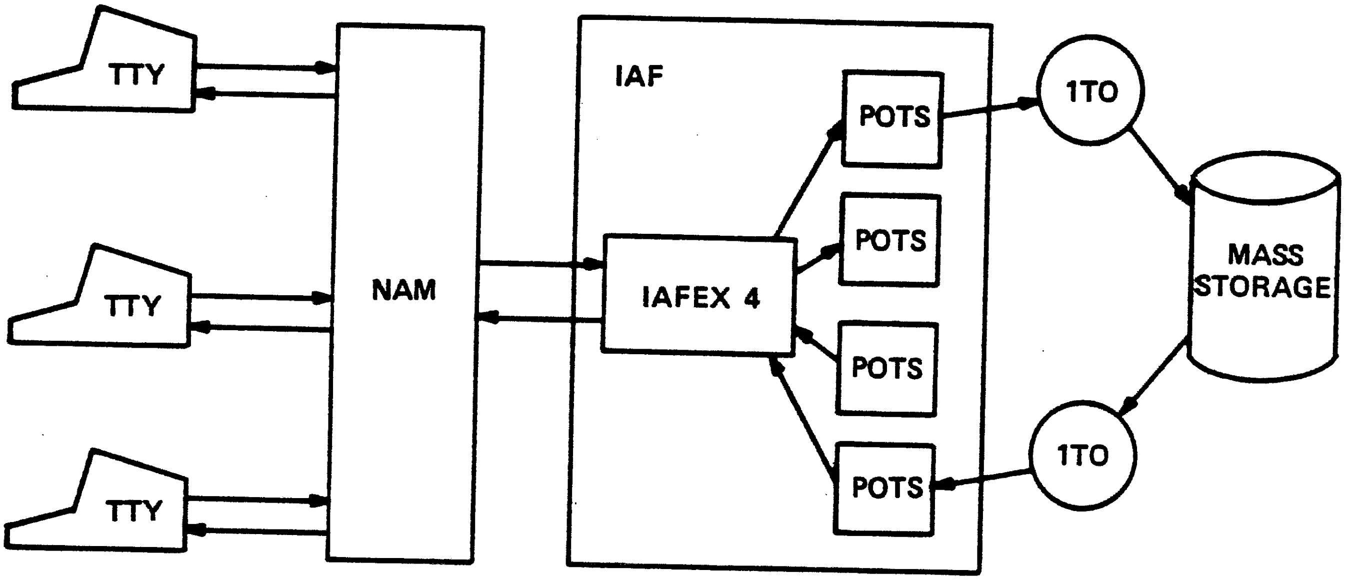
- TLX

USER CONTROL RA+1 CALL

COMMON DECKS

COMSNET - NETWORK PARAMETERS
COMSREM - TELEX SYSTEM PARAMETERS
COMSTCM - COMMUNICATION
COMSTDR - DRIVER
COMIICS - INITIAL CONTROL STATEMENTS
COMIIES - EST SEARCH
COMSNCD - NETWORK COMMUNICATION

IAF


TTY
TTY
TTY
NAM
IAF
IAFEX 4
POTS
POTS
POTS
POTS
1TO
1TO
MASS STORAGE

IAF

INITIALIZATION

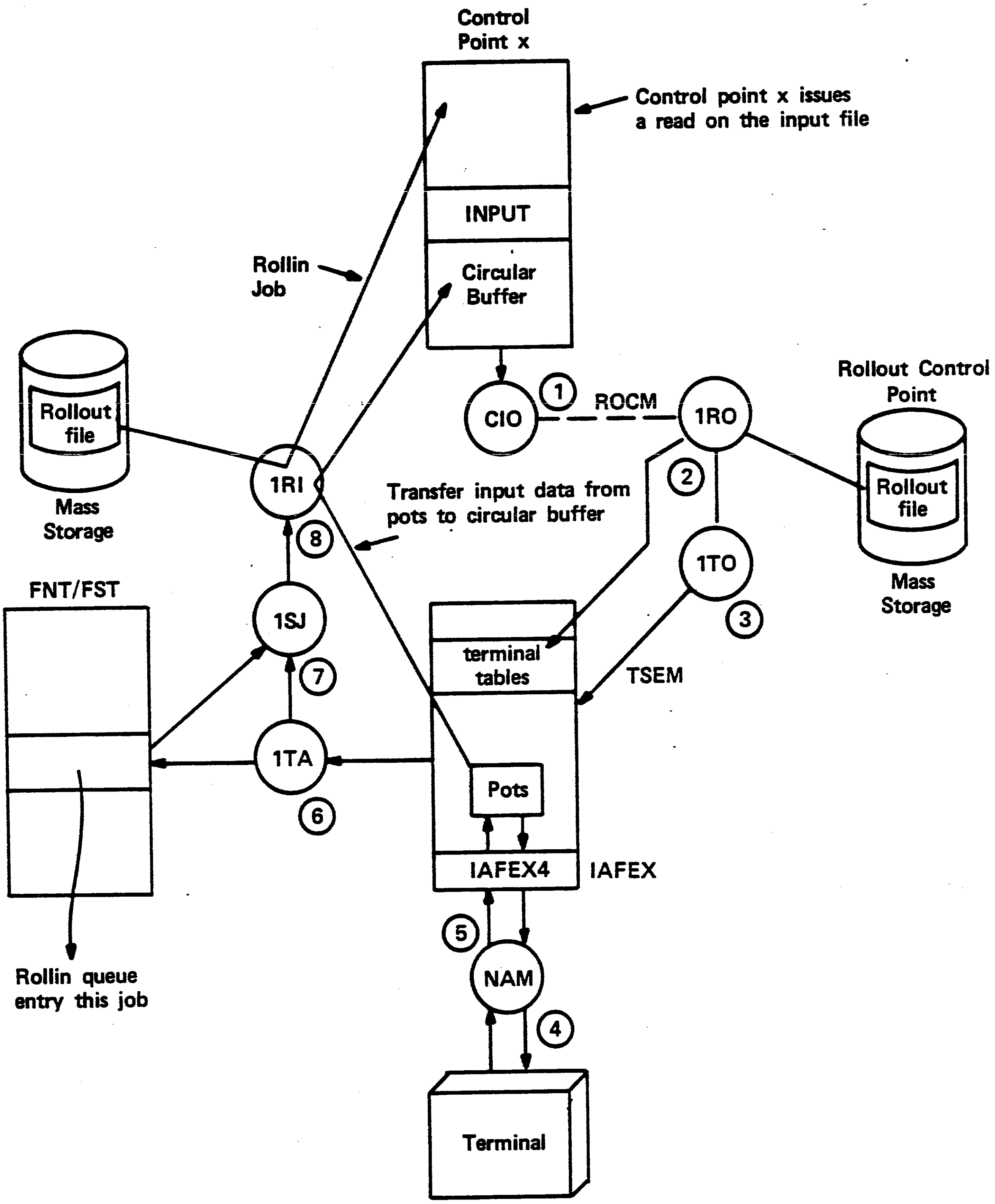
1. 1DS SETS UP FNT/FST INPUT QUEUE ENTRY FOR 1SI AND CONTROL POINT 1
2. 1SJ/3SA SCHEDULES CONTROL POINT 1 AND LOADS EXECUTES PROCEDURE FILE SPECIFIED THROUGH 1SI
3. IAFEX BEGINS EXECUTION IN CP (CP1) AND CALLS INITIALIZATION ROUTINE 1TN
4. EST ENTRY 76B READ AND TERMINAL TABLES BUILT
5. POT CHAIN BUILT BASED ON NUMBER OF TERMINALS
6. STATISTICAL ACCUMULATORS INITIALIZED
7. IAF QUEUE AREAS INITIALIZED
8. ALL TERMINAL TABLES SET "COMPLETE"
9. HEADER ADDRESS IS SET IN WARN ADDRESS AREA
10. DRIVER QUEUE INITIALIZED
11. CONTROL GIVEN TO IAFEX1

IAF

INPUT FROM A TTY

1. JOB ISSUES A READ REQUEST ON INPUT FILE - CALLS CIO. CIO SETS UP TINW AND DOES A ROCM - REQUEST ROLLOUT
2. 1RO DOES ROLLOUT AND CALLS 1TO. NO QUEUE ENTRY IS MADE FOR ROLLOUT FILE.
3. 1TO ISSUES TSEM TO INFORM IAF OF INPUT REQUEST
4. IAFEX4 CAUSES NAM TO ISSUE PROMPT "?" TO USER AT TERMINAL
5. USER INPUTS DATA; IAFEX4 STORES DATA IN POTs
6. WHEN END-OF-LINE IS SENSED, IAF CALLS 1TA TO REINITIATE THE JOB. 1TA BUILDS ROLLOUT QUEUE FNT/FST ENTRY
7. 1RI ROLLS IN JOB TO A CONTROL POINT AND TRANSFERS DATA FROM THE POTs TO THE JOB'S CIRCULAR BUFFER
8. JOB IS GIVEN THE CPU AND CONTINUES EXECUTION . . .

TERMINAL JOB INTERACTION (INPUT)



IAF

OUTPUT TO A TTY

1. JOB ISSUES A WRITE REQUEST ON OUTPUT FILE - CALLS CIO. CIO SETS TIOW AND DOES A ROCM - REQUEST ROLLOUT

2. 1RO DOES THE ROLLOUT BUT MAKES NO ROLLOUT QUEUE ENTRY. 1RO PUTS THE FIRST SECTOR OF DATA INTO ITS PP MEMORY

3. 1RO CAUSES 1TO TO BE LOADED INTO ITS PPU

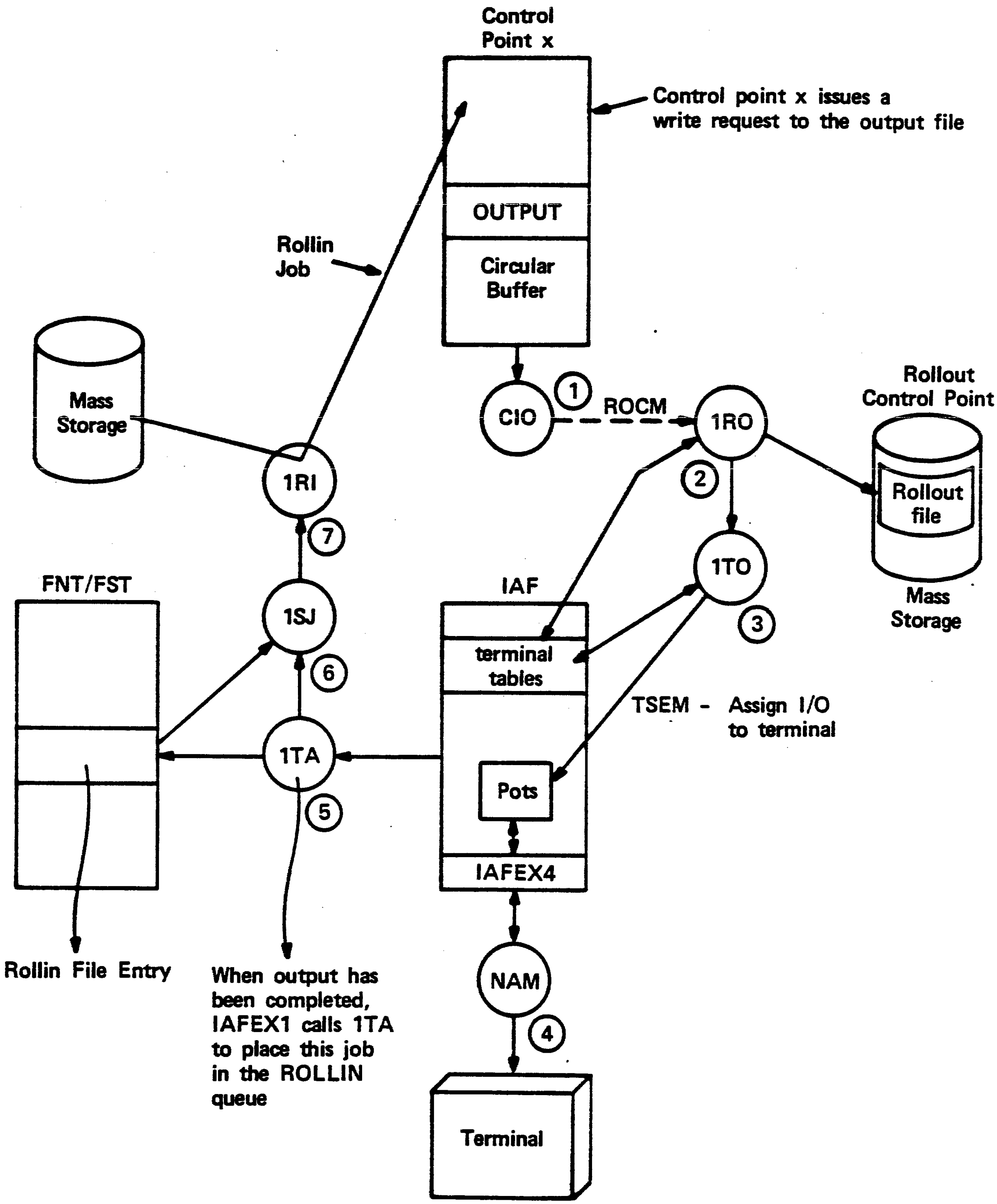
   1TO DOES A TGPM TO GET POTs FOR DATA. 1TO PUTS THE DATA INTO THE POTs AND CALLS IAF VIA TSEM TO INDICATE OUTPUT IS AVAILABLE

4. IAFEX1 CALLS IAFEX4 TO TRANSMIT DATA IN THE POTs TO THE TTY (USING NAM). IAFEX4 WILL ASK IAFEX1 FOR MORE DATA AND IAFEX1 WILL CALL 1TO TO GET IT UNTIL ALL DATA HAS BEEN TRANSFERRED

5. IAF CALLS 1TA TO REINITIATE JOB AFTER ALL DATA HAS BEEN SENT. 1TA BUILDS A ROLLOUT QUEUE FNT/FST ENTRY

6. 1RI ROLLS JOB IN; JOB BECOMES A CANDIDATE FOR EXECUTION

TERMINAL JOB INTERACTION (OUTPUT)

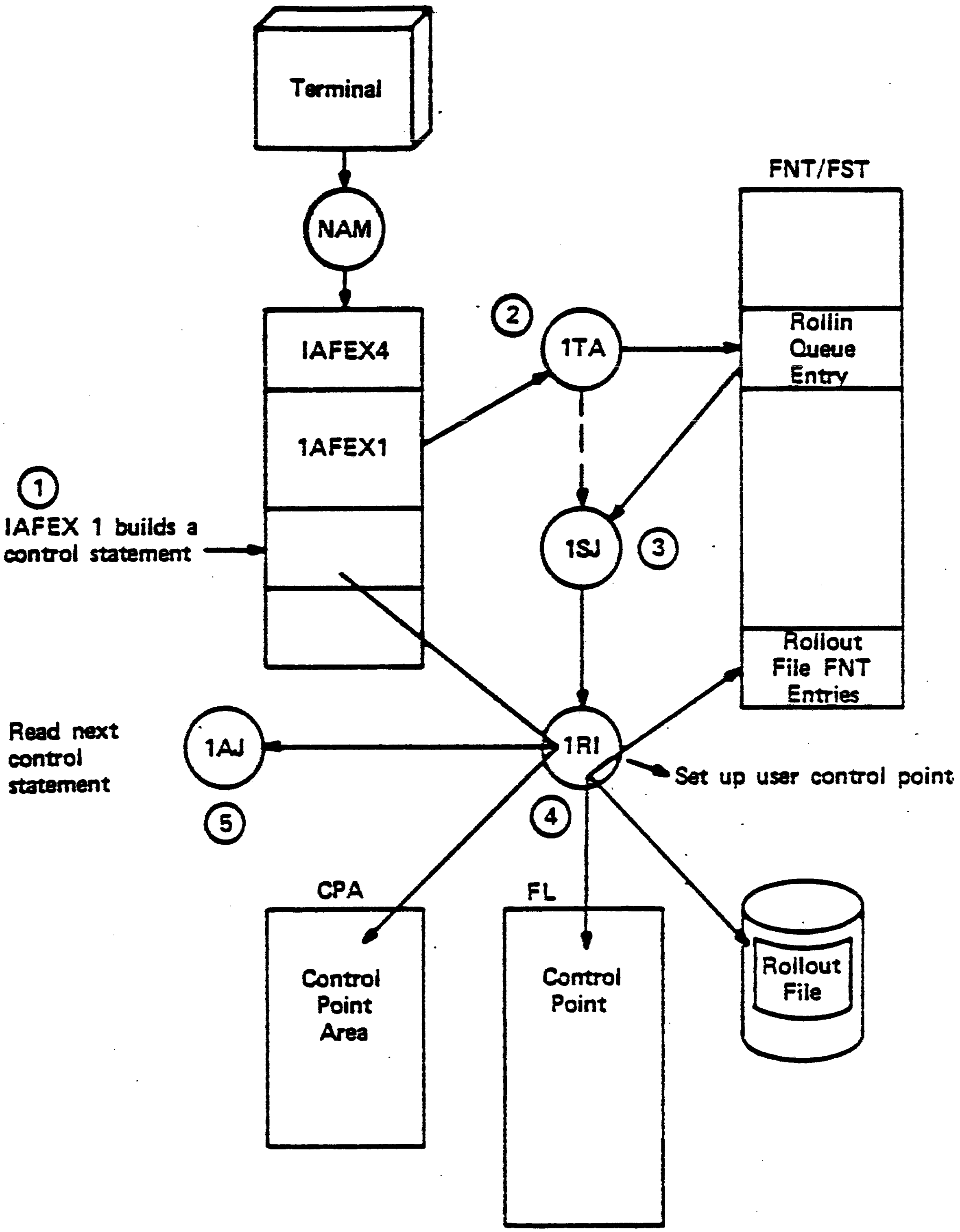


IAF

JOB INITIATION (JOB STEP)

1. CONTROL STATEMENT PUT INTO POT BY IAF AND CALLS 1TA
2. 1TA BUILDS ROLLOUT ENTRY AND PUTS CONTROL STATEMENT INTO MS1W AREA OF CONTROL POINT AREA
3. 1SJ SCHEDULES JOB FOR EXECUTION
4. 1RI ROLLS JOB IN
5. 1RI CALLS 1AJ TO ADVANCE JOB AND 1AJ WILL CALL TCS TO PROCESS THE CONTROL STATEMENT

TERMINAL JOB INITIATION



IAF

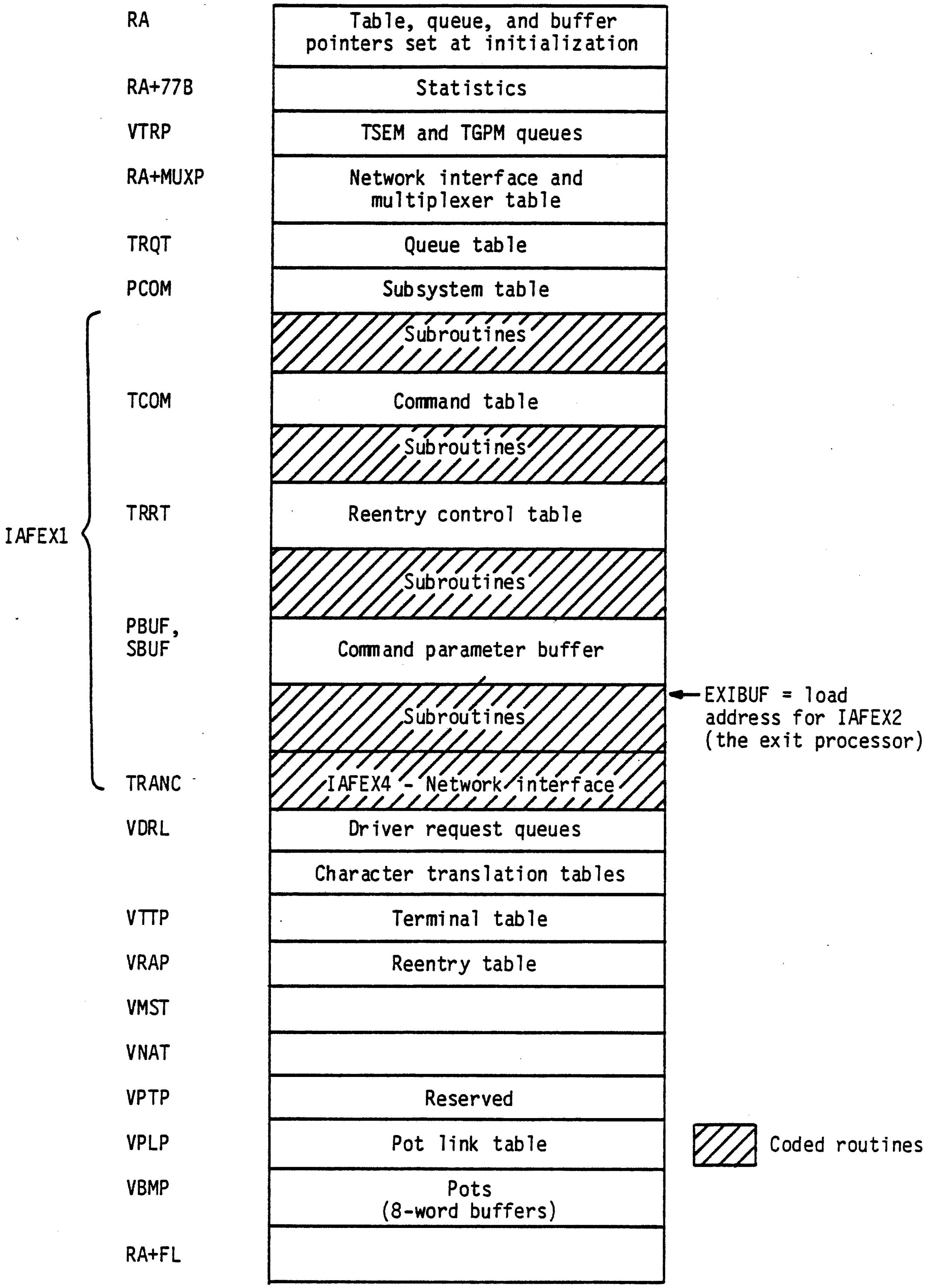
LINE NUMBERED DATA

1. USER ENTERS DATA AT TERMINAL
2. DATA BUILT CHARACTER BY CHARACTER AND ENTERED INTO POTs
3. WHEN VIPL POTs HAVE BEEN FILLED, IAFEX4 ISSUES A REQUEST TO DUMP THE POTs (VIPL=2)
4. IAF SENSES THE DUMP REQUEST AND CALLS 1TO TO DUMP THE POTs TO MASS STORAGE
5. 1TO DUMPS POTs TO ONE MS SECTOR
6. MEANWHILE, IAFEX4 MAY BE FILLING MORE POTs
7. THIS CONTINUES UNTIL USER CAUSES A SORT TO BE DONE
8. MSORT (NOS's ONLY MTOT JOB) IS CALLED
9. MSORT DOES A SHELL SORT AND PACKS DATA INTO FULL SECTORS
10. 1RO SET TERMINAL INTO ACTIVE MODE
11. IAF PROCESSES THE COMMAND THAT CAUSED THE SORT

IAFEX CONTROL POINT

RA	pointers and short queues
RA+TXORG	queue pointers, statistics, and internal messages
	IAFEX1 time-sharing executive
	IAFEX4 ● terminal conversion control ● terminal message control ● NAM interface (AIP)
	● terminal tables ● queues ● pots

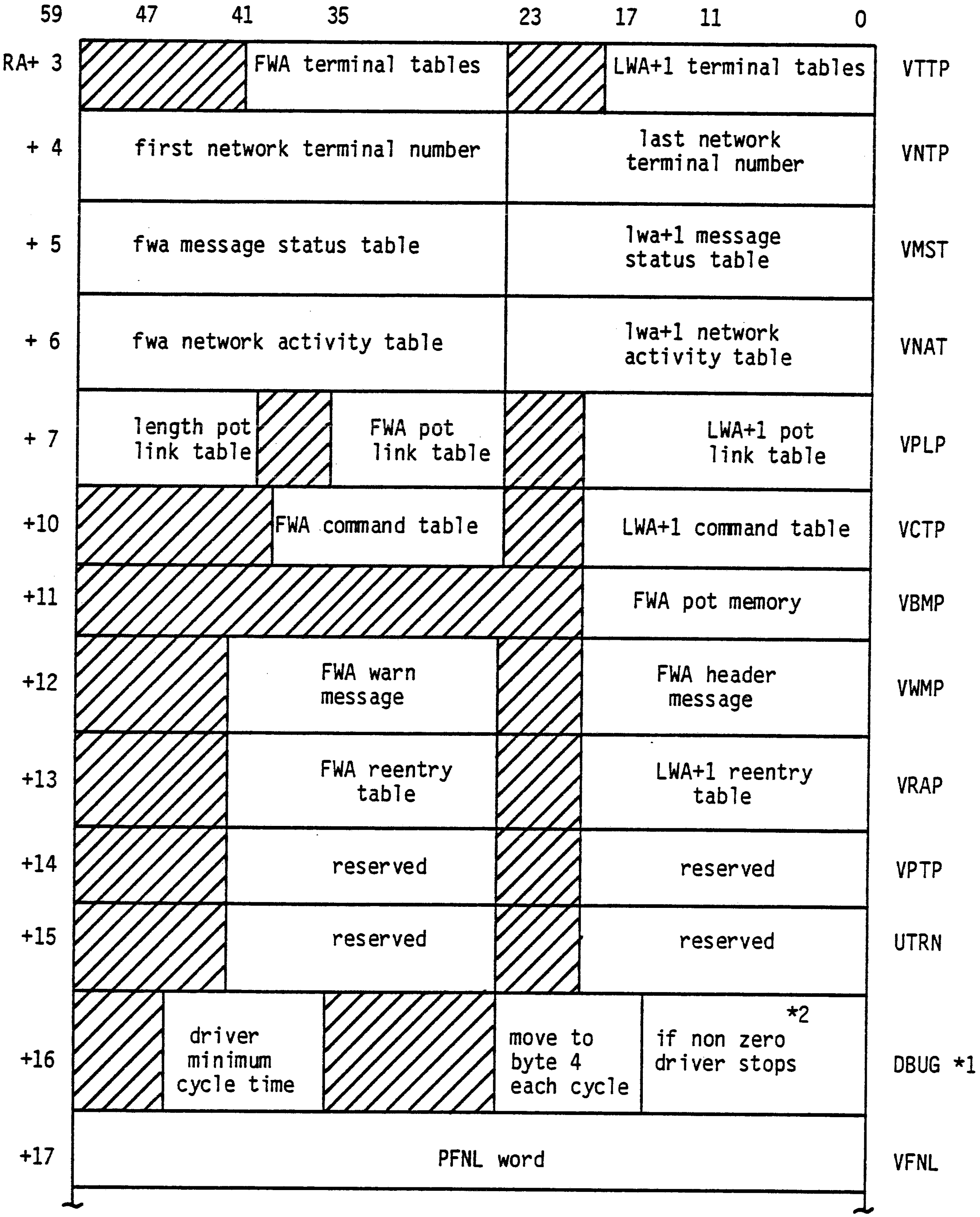
IAFEX1 MEMORY MAP



	RA	Table, queue, and buffer pointers set at initialization	
	RA+77B	Statistics	
	VTRP	TSEM and TGPM queues	
	RA+MUXP	Network interface and multiplexer table	
	TRQT	Queue table	
	PCOM	Subsystem table	
IAFEX1		▨ Subroutines	
	TCOM	Command table	
		▨ Subroutines	
	TRRT	Reentry control table	
		▨ Subroutines	
	PBUF, SBUF	Command parameter buffer	
		▨ Subroutines	← EXIBUF = load address for IAFEX2 (the exit processor)
	TRANC	▨ IAFEX4 - Network interface	
	VDRL	Driver request queues	
		Character translation tables	
	VTTP	Terminal table	
	VRAP	Reentry table	
	VMST		
	VNAT		
	VPTP	Reserved	
	VPLP	Pot link table	
	VBMP	Pots (8-word buffers)	
	RA+FL		

▨ Coded routines

POINTER ADDRESSES



59 47 41 35 23 17 11 0

RA+ 3	▨	FWA terminal tables	▨	LWA+1 terminal tables		VTTP
+ 4	first network terminal number			last network terminal number		VNTP
+ 5	fwa message status table			lwa+1 message status table		VMST
+ 6	fwa network activity table			lwa+1 network activity table		VNAT
+ 7	length pot link table	▨	FWA pot link table	▨	LWA+1 pot link table	VPLP
+10	▨	FWA command table	▨	LWA+1 command table		VCTP
+11	▨			FWA pot memory		VBMP
+12	▨	FWA warn message	▨	FWA header message		VWMP
+13	▨	FWA reentry table	▨	LWA+1 reentry table		VRAP
+14	▨	reserved	▨	reserved		VPTP
+15	▨	reserved	▨	reserved		UTRN
+16	▨	driver minimum cycle time	▨	move to byte 4 each cycle	*2 if non zero driver stops	DBUG *1
+17	PFNL word					VFNL

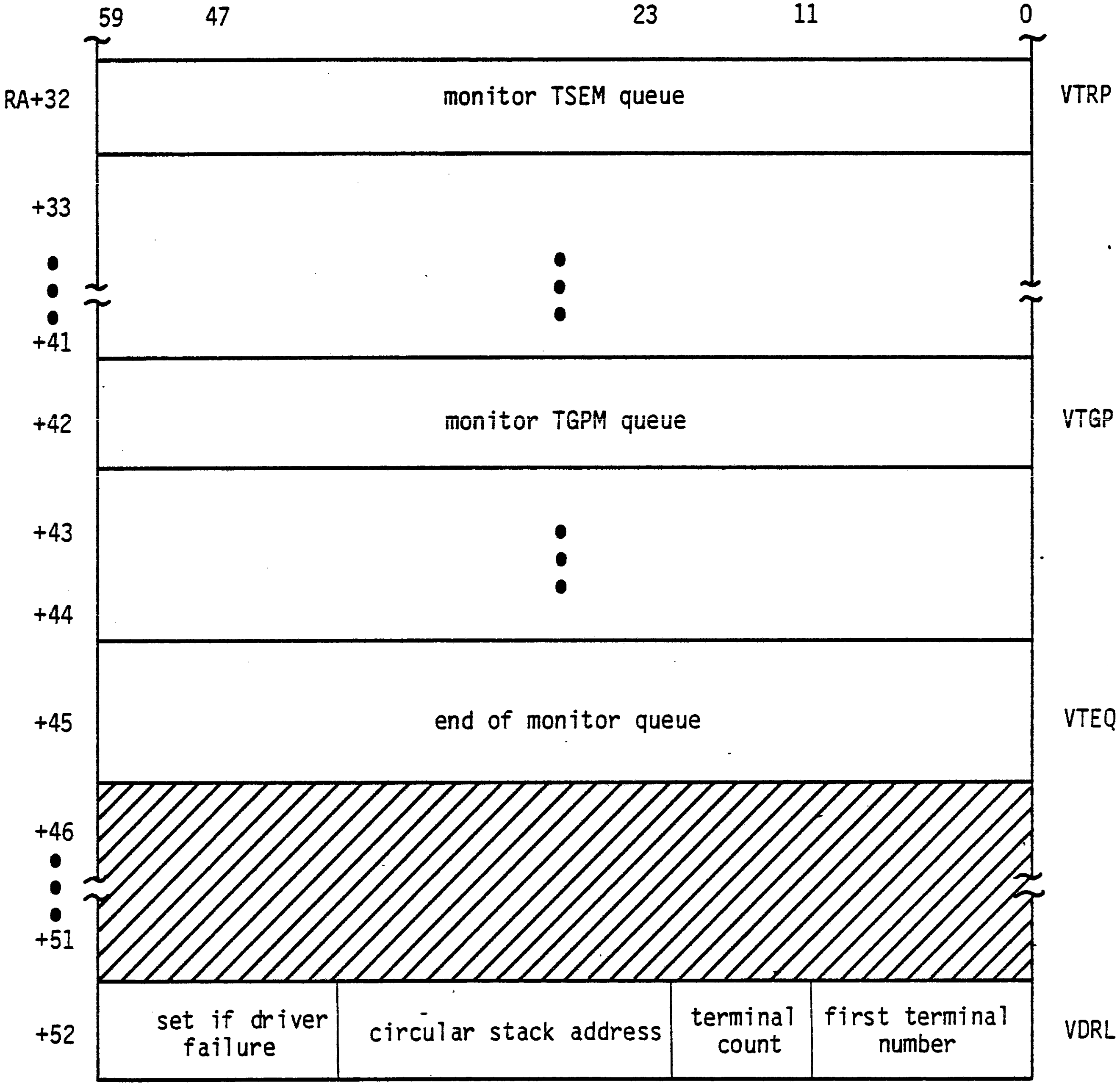
*1 Driver debug word

*2 Useful in debugging 1TD

POINTER ADDRESSES (CONTINUED)

	59	47	23	0
RA+20	number of times had to wait for PP			VPPL
+21	total users since initialization			VTNL
+22	current active user count location			VANL
+23	maximum number of possible users			VMNL
+24	new available pot count during FL change			VCPL
+25	negative indicates no reload		real-time clock at last recovery	VRLL
+26	abnormal occurrence counter			VABL
+27	▨	minimum number spare pots	maximum number spare pots	VPLL
+30	number of pots allocated (available)			VPAL
+31	number of pots in use			VPUL

POINTER ADDRESSES (Continued)



IAF

IAF MAIN LOOP

• DRI	PROCESS DRIVER QUEUE (DRIVER REQUESTS)
• URT	UPDATE RUNNING TIME
• STR	PROCESS SYSTEM REQUESTS (TSEM)
• RPC	REFILL POT CHAINQUEUE (TGPM)
• TDQ	PROCESS TIME DELAY QUEUE
• CSF	CHECK SALVARE FILE
• SOR	PROCESS SORT QUEUE
• SCH	SCHEDULE JOBS (JOB QUEUE)
• TSH	CHECK REQUEST COMPLETIONS (WAIT QUEUE)
• PPU	PROCESS PPU REQUESTS
• NDR	INTERFACE TO NAM
• SPR	CHECK FOR FL CHANGE (GET/RETURN POTs)
• EPP	ENTER PPU REQUESTS
• DRI	
• STR	
• RPC	
• RECALL	
• GO BACK TO TOP	

DRIVER REQUEST QUEUE

CIRCULAR STACK BUILT BY IAFEX4 (SIMILAR TO A FET I/O BUFFER, 100 WORD QUEUE).

EACH REQUEST IS BIASED BY 2000_8 SO THAT TABLE INDEX IS EASILY RETRIEVED BY AN UNPACK INSTURCTION UX1,B7 X2

B7 = REQUEST
X1 = ARGUMENTS/PARAMETERS

RQ+2000		P2	P1	TN

MONITOR REQUEST QUEUE

BUILT VIA TSEM FUNCTION (10 WORD QUEUE)

FN+2000	P1	P2	P3	P4

FUNCTIONS DEFINED IN COMSREM

POT REQUEST QUEUE

BUILT VIA TGPM FUNCTION (3 WORD QUEUE)

TERMINAL TABLE

EIGHT WORD ENTRY FOR EACH POSSIBLE ACTIVE USER (IE, ONE FOR EACH MUX PORT)

0 VUIT

USER NUMBER	USER INDEX

1 VFNT

PRIMARY FILE NAME	M	BFL

M = MODE 2^0 - WRITE LOCKOUT

2^2 - EXECUTE ONLY

BFL = BATCH FIELD LENGTH

2 VFST

EQ LIST	PRIM EQ	PRIM FT	CT		STATUS

AA

- LIST OR PRIMARY FILE CURRENT SECTOR
- CONTROL STATEMENT POT
- ACCOUNTING POT

STATUS

11-6	UNUSED
5	LINE CONTINUATION
4	7400 ESCAPE
3	7600 ESCAPE
2	BINARY
1	TRANSPARENT
0	EXTENDED

TERMINAL TABLE (CON'T)

3 VROT	W_C	ROLL EQ	ROLLOUT FT		FL	SUB STATUS	STATUS

SUBSTATUS (1RI)

LLLLooFTIsss
LLLL=LEVEL
I=INTERRUPT
F=ROLLIN FL
T=TERMINATE SSJ
SSS=1 - EOR
    2 - EOF
    3 - EOI

STATUS	BIT=VALUE
IAF IN CONTROL	0=1
SYSTEM IN CONTROL	0=0
JOB IN SYSTEM	1=0
JOB TO BE ROLLED IN	2=1
JOB WAITING INPUT	3=1
OUTPUT AVAILABLE	4=1
SSJ	5=1
LIST	6=1
MULTI TERMINAL	7=1
SUSPENDED	9=1
ERROR ON LAST OPERATION	11=1

TERMINAL TABLE

4 VDPT	FIRST POT	LAST POT	FW	WC	CONTROL FLAGS	

FW	FIRST WORD OF FIRST POT
WC	LAST POT WORD COUNT

CONTROL FLAGS

11-5	UNUSED
4	SOURCE INPUT
3	BINARY TRANSMISSION
2	TRANSPARENT INPUT
1	UNUSED
0	UNUSED

5 VCHT	NDR REENTRY		POT POINTER	INPUT CC	OUTPUT CC

6 VDCT	FLAGS	TERM. CONTROL	AUTO INCR.	ACCESS (AACW)	NEXT MESSAGE

FLAGS

11	DRIVER REQUEST FROM IAFEX1
10	INTERRUPT COMPLETE
9	USER LOGGED IN
8	INPUT REQUESTED
7	UNUSED
6	READ DATA MODE
5	UNUSED
4	UNUSED
3	EXTERNAL MODE
2	TEXT MODE
1	AUTO MODE
0	TAPE MODE

IAF

TERMINAL TABLE

7 VSTT	FLAGS	FIRST POT	COMM. INDEX		S U B S	QUEUED

FLAGS (BIT)

0	LOGOUT IN PROGRESS
1	UNCONDITIONAL ABORT FLAG
2	WARNING ISSUED
3	RUN COMPLETE MESSAGE
4	SORT FLAG
5	TIME/SRU LIMIT FLAG
6	JOB COMPLETE FLAG
7	INPUT LOST/JOB NOT STARTED
8	
9	CHARGE REQUIRED
10	CONDITIONAL ABORT FLAG
11	DISABLE TERMINAL CONTROL

SUBS = SUBSYSTEM

0	NULL
1	BASIC
3	FTNTS
4	EXECUTE
5	BATCH
6	ACCESS

POT LINK TABLE

PLT CONTROLS POTS IN A MANNER SIMILAR TO HOW TRT CONTROLS TRACKS

PLP	0	1	2	3	4
	7777	0002 1	0003 2	0004 3	0017
	0005 4	0000 5	0007 6	0000 7	0017
	0000 10	0012 11	0013 12	0014 13	0007
	0000 14	0000 15	0000 16	0000 17	0010

$$\overbrace{0000\ 0000\ \underbrace{\overset{0000}{0123}}_{}}^{}$$

RESERVATION BITS

```
000 000 000 0 | 00
    WORD      |  BYTE
```

- POT 0 ALWAYS 7777
- LAST POT IN CHAIN = 0000
- UNUSED POT ALSO = 0000 BUT NOT RESERVED

IAF

RE-ENTRY

RE-ENTRY FOR RETURNING CONTROL OR HAVING FUNCTIONS PERFORMED WHEN A SET OF CONDITIONS ARE MET.

ONE WORD PER TERMINAL WITH INDEX TO TABLE OF RE-ENTRY PROCESSORS DEFINED VIA COMMND MACRO.

NAM INTERFACE - IAFEX4

SEND AND RECEIVE DATA FROM TERMINALS THAT ARE CONNECTED TO IAF THROUGH NAM

- RECEIVES, INTERPRETS, AND SENDS SUPERVISORY MESSAGES
- TRANSFORMS OUTGOING DATA FROM INTERNAL FORMS TO THE APPROPRIATE NETWORK FORMAT
- TRANSFORMS INCOMING DATA FROM NETWORK FORMAT TO INTERNAL FORMAT
- MANAGES DATA TRAFFIC TO OPTIMIZE INTERACTIVE PERFORMANCE

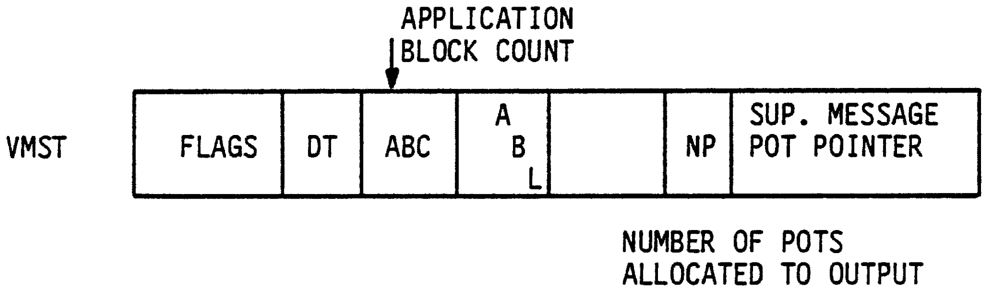
FUNCTIONS

- CONNECTION ESTABLISHMENT (LOG IN)
- COMMAND LINE ENTRY
- SOURCE LINE ENTRY
- INPUT TO RUNNING PROGRAM
- OUTPUT PROCESSING
- SESSION TERMINATION

interface control words interface statistics message headers and fixed messages
IAF/network interface control
terminal manager
supervisory message processor
upline data manager
network interface control subroutines
general subroutines
data translation subroutines
common decks and code conversion tables
interface buffers
application interface program (AIP)

MESSAGE STATUS TABLE

1 WORD/NETWORK TERMINAL

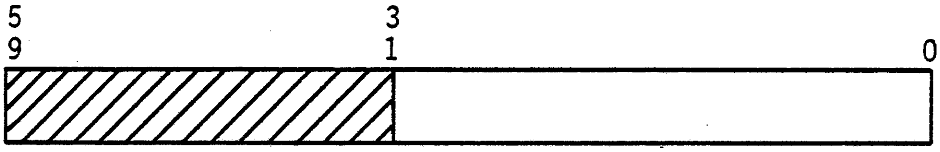


FLAGS

BIT 59 = TERMINAL ON-LINE
58 = SUSPEND TRAFFIC
57 = BREAK IN PROGRESS
56 = SHUTDOWN WARNING SENT
55 = END-CONNECTION IN PROGRESS
54 = DATA RECEIVED
53 = INPUT ENABLED (MSG BLOCK SENT)

NETWORK ACTIVITY TABLE

VNAT - ONE BIT FOR EACH NETWORK TERMINAL
WORD LOCATION = (TERM. NUMBER - 1ST NETWORK TERM. NO.)/32
BIT LOCATION =(TERM. NUMBER - 1ST NETWORK TERM. NO.) MOD 32



AUXILIARY - PP ACTION FOR IAF

1TA CALLED WITH LIST OF REQUESTS IN POTs OR SINGLE REQUEST

FUNCTION	DESCRIPTION
1	ADJUST IAF FL
2	RETURN TERMINAL JOB
3	CREATE ROLLOUT FILE
4	LOGOUT TERMINAL
5	DISPLAY ACCOUNTING MESSAGE
6	TERMINAL RECOVERY
7	INCREMENT RESOURCE LIMIT (TL, SRU)
10	RECOVERY FILE PROCESSOR
11	SCHEDULE JOBS
12	GATHER STATISTICS
13	CLEAN UP SALVARE

AUXILIARY - TTY INPUT/OUTPUT

1TO DOES TERMINAL INPUT/OUTPUT THAT REQUIRES MASS STORAGE REQUESTS

RECOVERY FILE - SALVARE

- BUILT AT INITIALIZATION TIME
- TWO WORDS FOR EACH DEFINED TERMINAL

FO	EQ	FT	HH MM SS	UI

IA		TO

FO = FAMILY ORDINAL
EQ = ROLLOUT FILE EQUIPMENT
FT = ROLLOUT FILE FIRST TRACK
HHMMSS = LAST ENTRY TIME
UI = USER INDEX
IA = INSTALLATION AREA
TO = TERMINAL TABLE ORDINAL

- ENTRIES CLEARED AFTER 10 MINUTES
- IN RECOVERY, BOI + EOI OF USER'S FILES ARE VALIDATED IF SENSE SWITCH SET.

QUESTION SET LESSON 24

1. What routines make up the time-sharing subsystem?

2. How is the dynamic memory associated with POTs managed?

3. Explain in general the time-sharing origin job flow for each of the following:

   a. Job initiation
   b. Terminal input
   c. Terminal output

4. Where is all the information about any active terminal kept?

5. What is a multi-terminal job?

6. How do the terminals get processed as the time-sharing executive progresses around its main loop?

7. How does the time-sharing executive make requests for its auxiliary PP routines? What happens if no PPs are available?

8. Why does the time-sharing executive queue a group of request for the auxiliary routine 1TA?

9. Why are all time-sharing jobs scheduled to the rollout queue instead of the input queue?

10. What happens when a time-sharing origin job step aborts? What happens when the user logs off?

11. What function does the auxiliary routine 1TO perform?

LESSON 25
REMOTE BATCH SUBSYSTEM

LESSON PREVIEW:

This lesson introduces the student to the remote batch entry subsystem, Export/Import 200 or EI200, as it is more commonly called. EI200 has been succeeded by the Remote Batch Facility (RBF) Networks Products offering.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

- Identify the components of the EI200 subsystem and detail their major functions.
- Map the allocation of EI200's control point field length and the port table.
- Detail the job flow through the subsystem from the submittal of the job to the disposition of job output.
- Describe the services performed for the subsystem by the auxiliary PP routine XSP.

REFERENCES:

NOS IMS - Chapter 33

EI200
EXPORT/IMPORT

REMOTE BATCH ENTRY

RJE - REMOTE JOB ENTRY

- MODE 4A 200 UT PROTOCOL
- 2000 2400 4800 (9600) BAUD LINE SPEEDS
- UP TO 16 TERMINALS 6671
  2550-100
- BCD CHARACTER SET (63 CHARACTER SET)

E200CP - CPU PROGRAM
FETS AND BUFFERS IN UPPER FL

1LS - AUXILIARY (TRANSIENT PP)
USES "RPL" OVERLAYS FROM E200CP

1ED - DRIVER (DEDICATED PP)

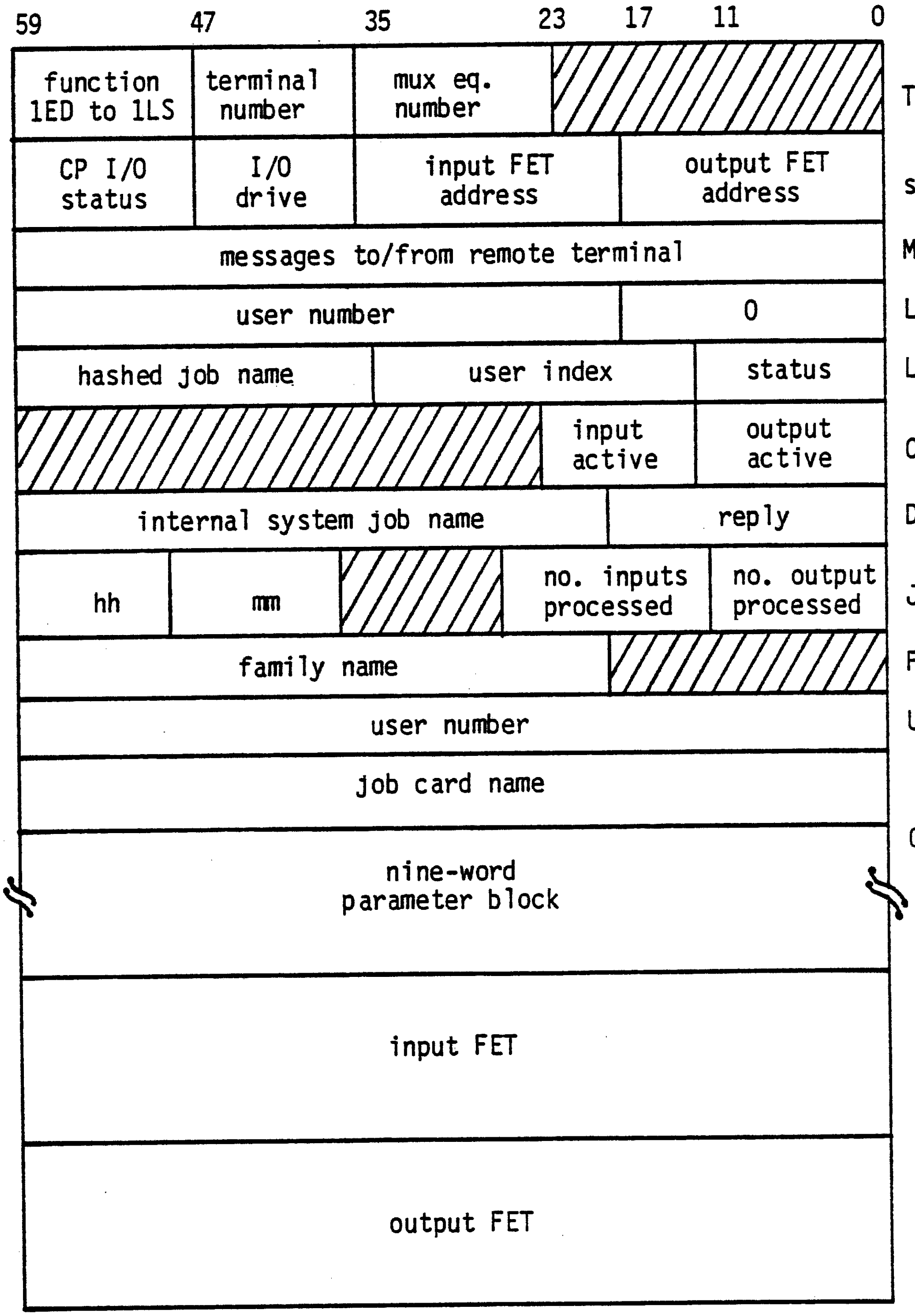
XSP - AUXILIARY (TRANSIENT PP)

COMSEXP - CONSTANTS, POINTERS, TABLE AREA DEFINITIONS

EI200
MEMORY LAYOUT

POINTERS	
FUNCTION STATUS TABLE	TFS
MESSAGE BUFFER AREA	MSGB
LOGIN TABLE	LINF
CPU INTERLOCK TABLE	CPIK
DROP JOB TABLE	DPJT
JOB STATISTICS TABLE	JST
FAMILY NAME TABLE	FAMT
USER NUMBER TABLE	UNJC
QAC PARAMETER BLOCK	QAPB
QAC PARAMETER BLOCK	QAPC
E200CP MAIN LOOP/SUBROUTINES	
LOCAL OVERLAY LIBRARY (MINI RPL)	
ALLOCATED FETS AND BUFFERS	

PORT TABLE LAYOUT



59 47 35 23 17 11 0

function 1ED to 1LS	terminal number	mux eq. number	▨		TFS
CP I/O status	I/O drive	input FET address	output FET address		status word
messages to/from remote terminal					MSGB
user number	0				LINF
hashed job name	user index	status			LINF+1
▨	input active	output active			CPIK
internal system job name	reply				DPJT
hh	mm	▨	no. inputs processed	no. output processed	JST
family name	▨				FAMT
user number					UNJC
job card name					
nine-word parameter block					QAPB
input FET					
output FET					

EXPORT/IMPORT FETs

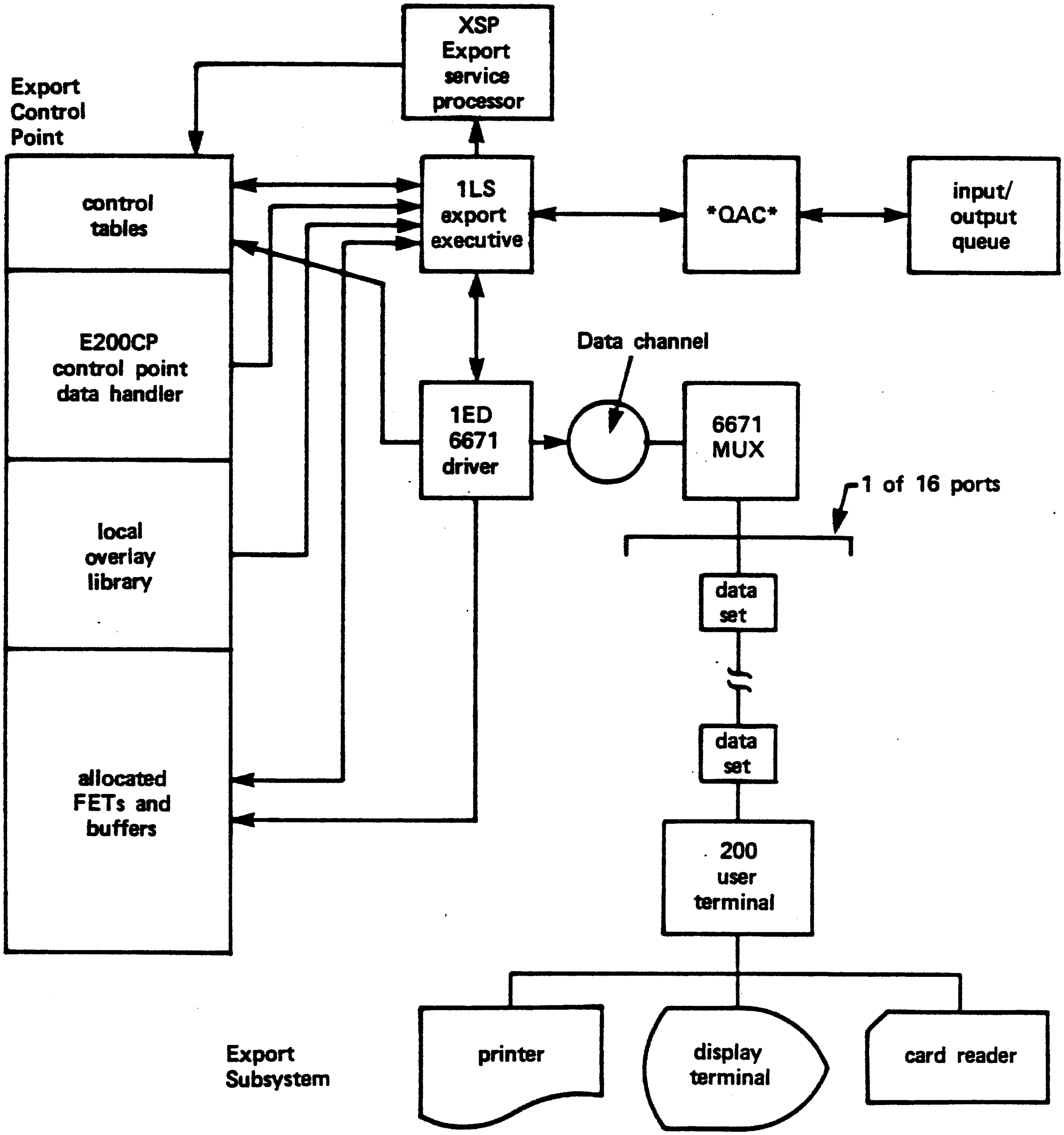
INPUT FET

<table>
<tr><td>FET+0</td><td colspan="3">internal system name</td><td>code/status</td></tr>
<tr><td>1</td><td colspan="3"></td><td>FIRST</td></tr>
<tr><td>2</td><td colspan="3"></td><td>IN</td></tr>
<tr><td>3</td><td colspan="3"></td><td>OUT</td></tr>
<tr><td>4</td><td>FNT address</td><td colspan="2">0</td><td>LIMIT</td></tr>
<tr><td>5</td><td>full/empty driver flag</td><td>job card processing in progress</td><td>address of line following EOR</td><td>address of line following EOF</td></tr>
<tr><td>6</td><td colspan="2">job sequence number</td><td>0</td><td>pointer to next allocated FET</td></tr>
<tr><td>7</td><td>job priority</td><td>job time limit</td><td>job FL 0</td><td>card count</td></tr>
</table>

OUTPUT FET

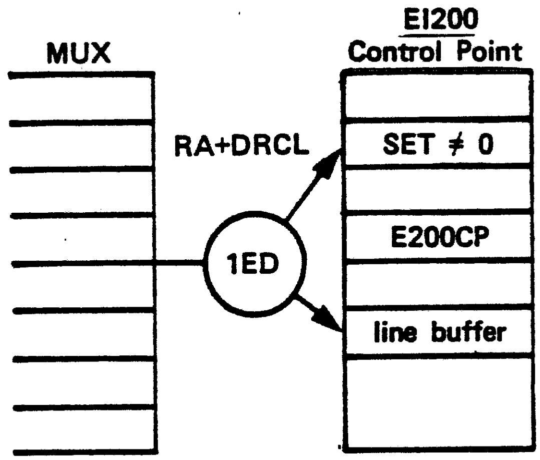
<table>
<tr><td>FET+0</td><td colspan="4">internal system name</td><td>code/status</td></tr>
<tr><td>1</td><td colspan="3"></td><td>01</td><td>FIRST</td></tr>
<tr><td>2</td><td colspan="4">0</td><td>IN</td></tr>
<tr><td>3</td><td colspan="4">0</td><td>OUT</td></tr>
<tr><td>4</td><td>FNT address</td><td>dayfile first track</td><td>dayfile first sector</td><td>0</td><td>LIMIT</td></tr>
<tr><td>5</td><td>full/empty driver flag</td><td colspan="4">0</td></tr>
<tr><td>6</td><td colspan="3">job sequence number</td><td>0</td><td>pointer to next allocated FET</td></tr>
<tr><td>7</td><td colspan="5">print line count</td></tr>
</table>

TAF ENVIRONMENT

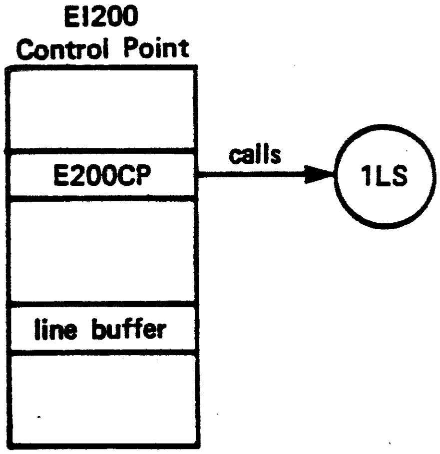


EI200

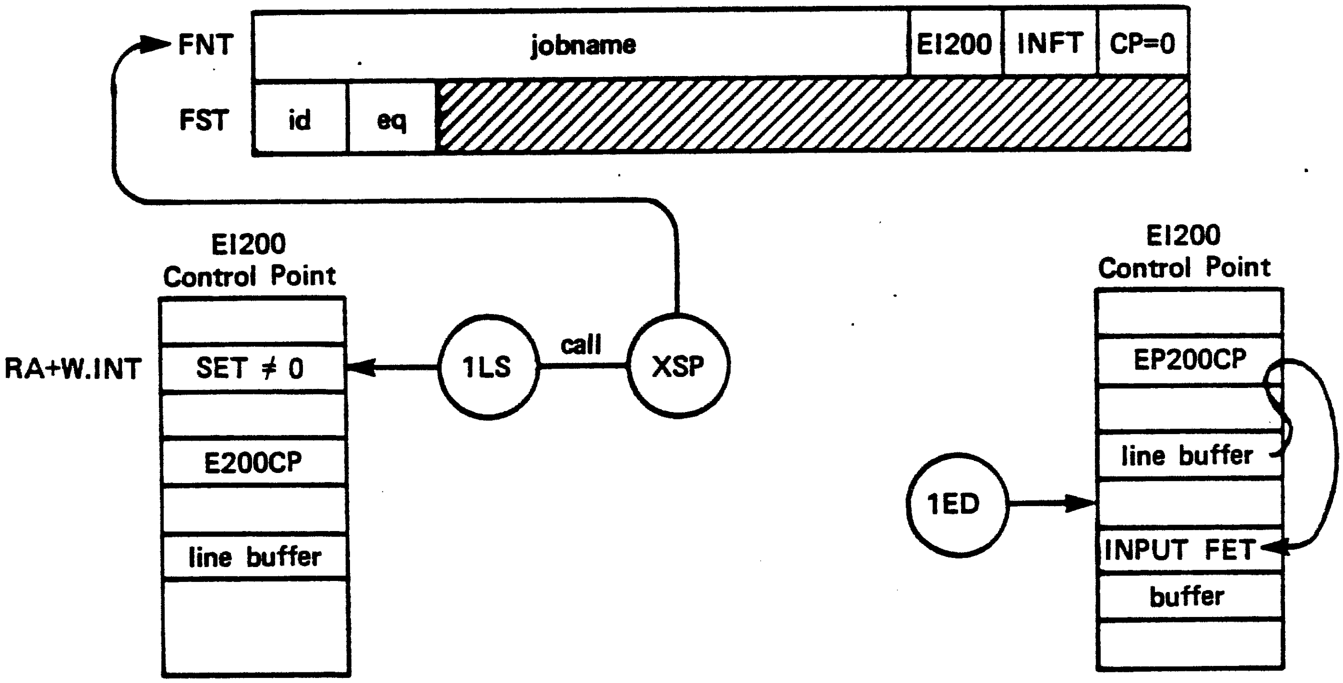
1. 1ED READS DATA FROM MUX TO LINE BUFFER (IN E200CP FL) AND RECALLS E200CP.
2. E200CP BEGINS AND CALLS 1LS AND GOES BACK INTO RECALL.
3. 1LS GETS E200CP OUT OF RECALL INTO EXECUTION.
4. 1LS CALLS XSP TO BUILD INPUT QUEUE ENTRY FOR JOB IN THE LINE BUFFER.
5. E200CP REFORMATS THE LINE BUFFER AS 1ED PASSES IT AND MOVES THE DATA TO THE TERMINAL'S INPUT BUFFER. E200CP CALLS CIO TO EMPTY THE BUFFER. THIS CONTINUES UNTIL ENTIRE JOB DECK HAS BEEN READ (EOI SENSED).
6. RUN THE JOB. OUTPUT AVAILABLE IN OUTPUT QUEUE.
7. 1LS FINDS OUTPUT QUEUE ENTRY WITH TERMINAL ID (TID) MATCHING ONE OF THE ACTIVE PORTS. CALLS OBP TO BUILD BANNER PAGE. TELLS E200CP THAT OUTPUT IS READY.
8. E200CP READS OUTPUT FILE TO TERMINAL'S OUTPUT BUFFER.
9. E200CP FORMATS DATA FROM OUTPUT BUFFER INTO LINE BUFFER AND TELLS 1ED.
10. 1ED SENDS DATA FROM LINE BUFFER TO RJE TERMINAL.
11. E200CP FILLS BUFFER (VIA CIO) AND PASSES FORMATTED LINES TO 1ED UNTIL EOI IS REACHED.
12. E200CP GOES INTO AUTO RECALL AND 1ED KEEPS ROLLING MUX LOOPING FOR INPUT.



1ED reads from multiplexer to line buffer and sets RA+DRCL words to nonzero. This takes E200CP out of autorecall.



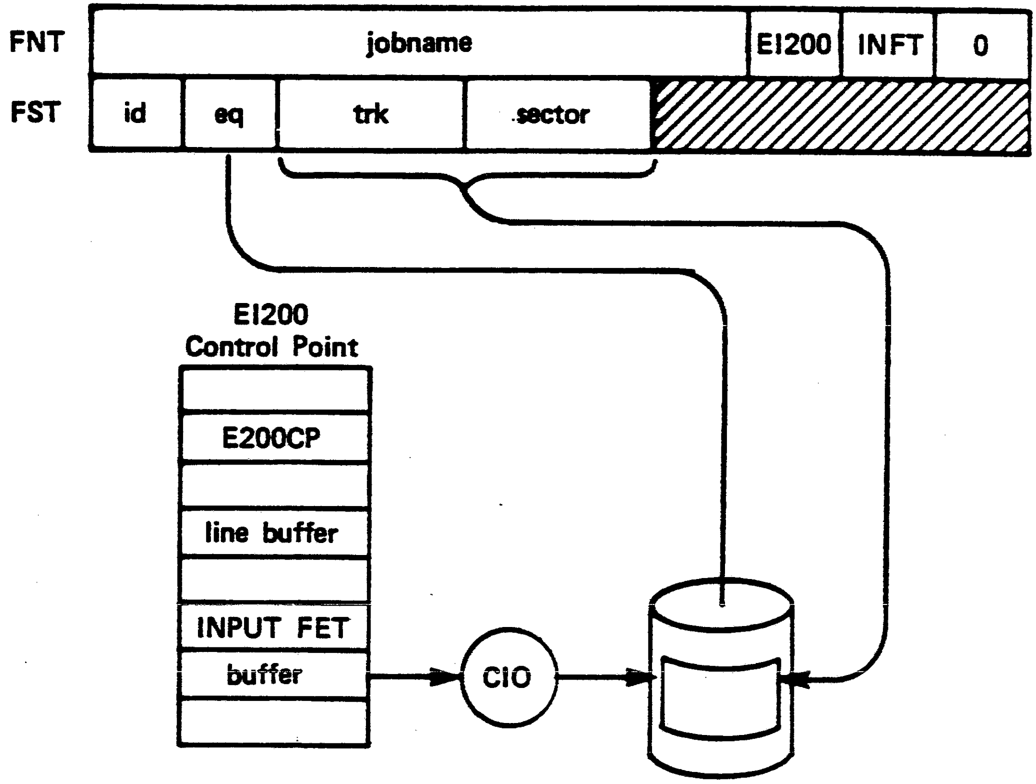
E200CP calls 1LS and goes into autorecall.



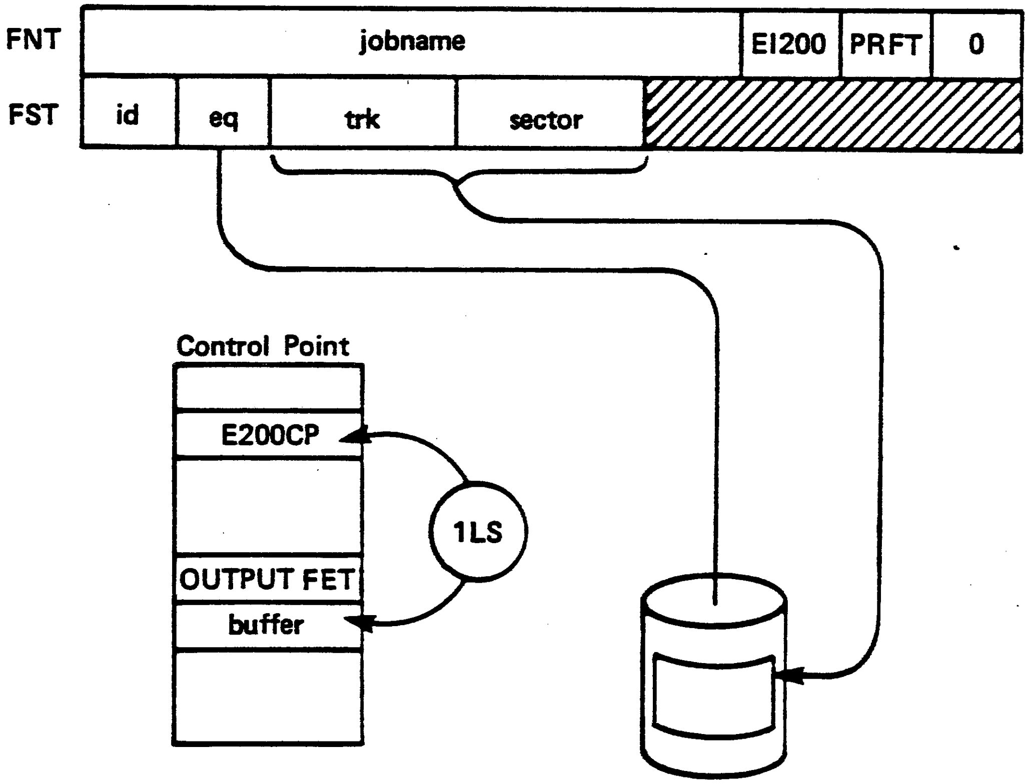
1LS sets RA+W.INT to nonzero, which takes E200CP out of autorecall. 1LS calls XSP to create an FNT/FST input queue entry for the job in the line buffer, using OBF and OVJ to crack the job card.

E200CP reformats the line buffer data as 1ED passes it and moves the data to the input FET buffer.

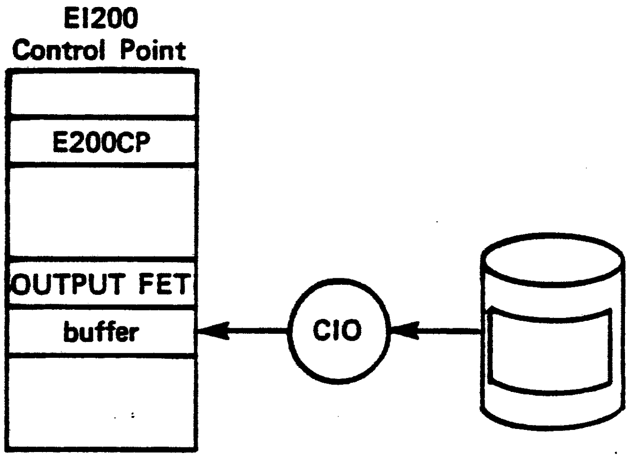
E/I OPERATION (Continued)



E200CP calls CIO to write the data from the input FET buffer to the disk.

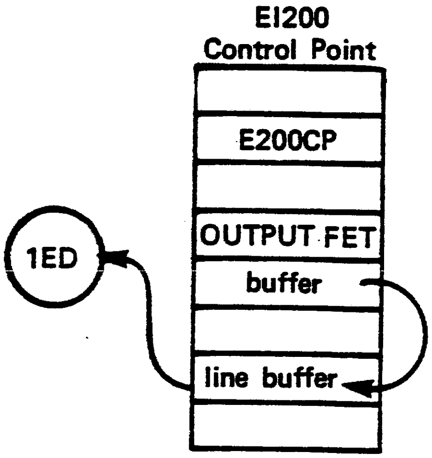


1LS finds an output queue entry and calls OBP to create a banner page in the output FET buffer and informs E200CP.

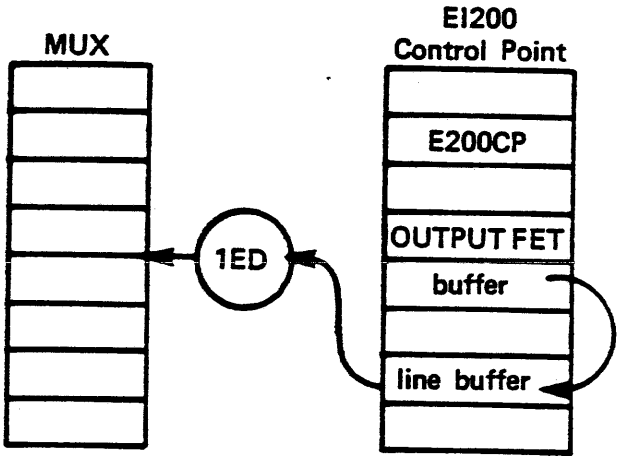


E200CP reads the output file via CIO into the output FET buffer.

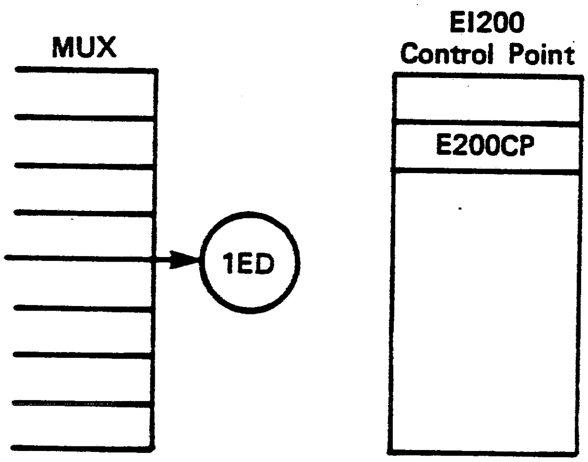
E/I OPERATION (Continued)



E200CP formats the output FET buffer for the remote printer and informs 1ED.



1ED sends the line buffer data, one line at a time, to the remote printer.



E200CP goes into autorecall and 1ED continues to poll the multiplexer.

XSP

1. VALIDATES TERMINAL'S USER NUMBER DURING LOGIN OF TERMINAL

2. MAKE JOB ENTRY (USER JOB SUBMITTAL)

   - OVJ - JOB CARD CRACKER
     - JOB CARD
     - USER CARD
   - JOB SEQUENCE NUMBER (RJSM)
   - BUILDS FNT/FST INPUT ENTRY
   - BUILDS SYSTEM SECTOR - WITH DESTINATION FAMILY, USER NUMBER, USER INDEX OF TERMINAL. THESE ARE USED FOR OUTPUT TID.

QUESTION SET LESSON 25

1. How is the EI200 control point's field length used?

2. What tables are associated with each terminal?

3. How does the EI200 subsystem perform mass storage I/O?

4. Trace the flow of data from the remote batch terminal to the system input queue. Trace the flow of data from the system output queue to the remote batch terminal.

5. How does EI200 determine which terminal a file in the output queue should be routed to?

6. What does the auxiliary routine XSP do for the EI200 subsystem?

7. What characteristic of the driver (1ED) enables the remote batch terminal user to suspend a file which is currently being printed?

LESSON 26
TRANSACTION SUBSYSTEM (TRANEX/TAF)

LESSON PREVIEW:

The purpose of this lesson is to introduce the student to transaction processing and the NOS subsystem dedicated to transaction processing: the Transaction Facility, TAF.

This lesson is not to be an in-depth study of TAF, but to give the student an overview of TAF and its capabilities.

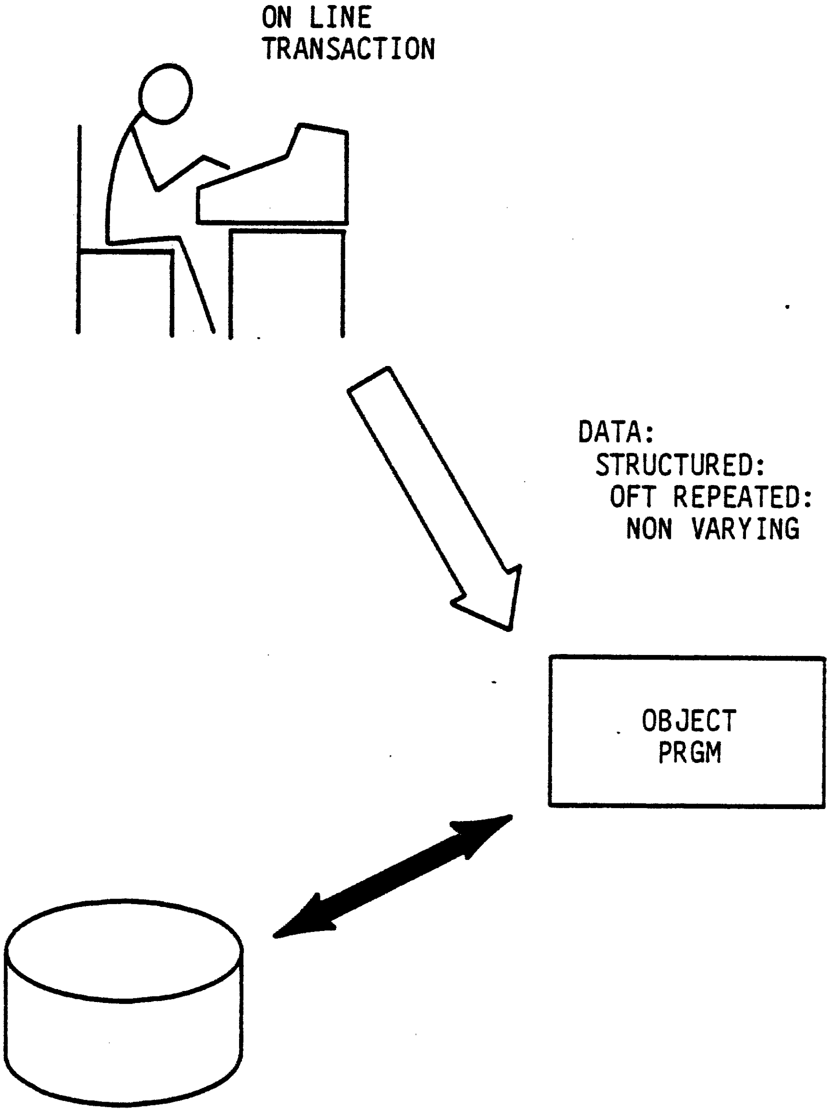
OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

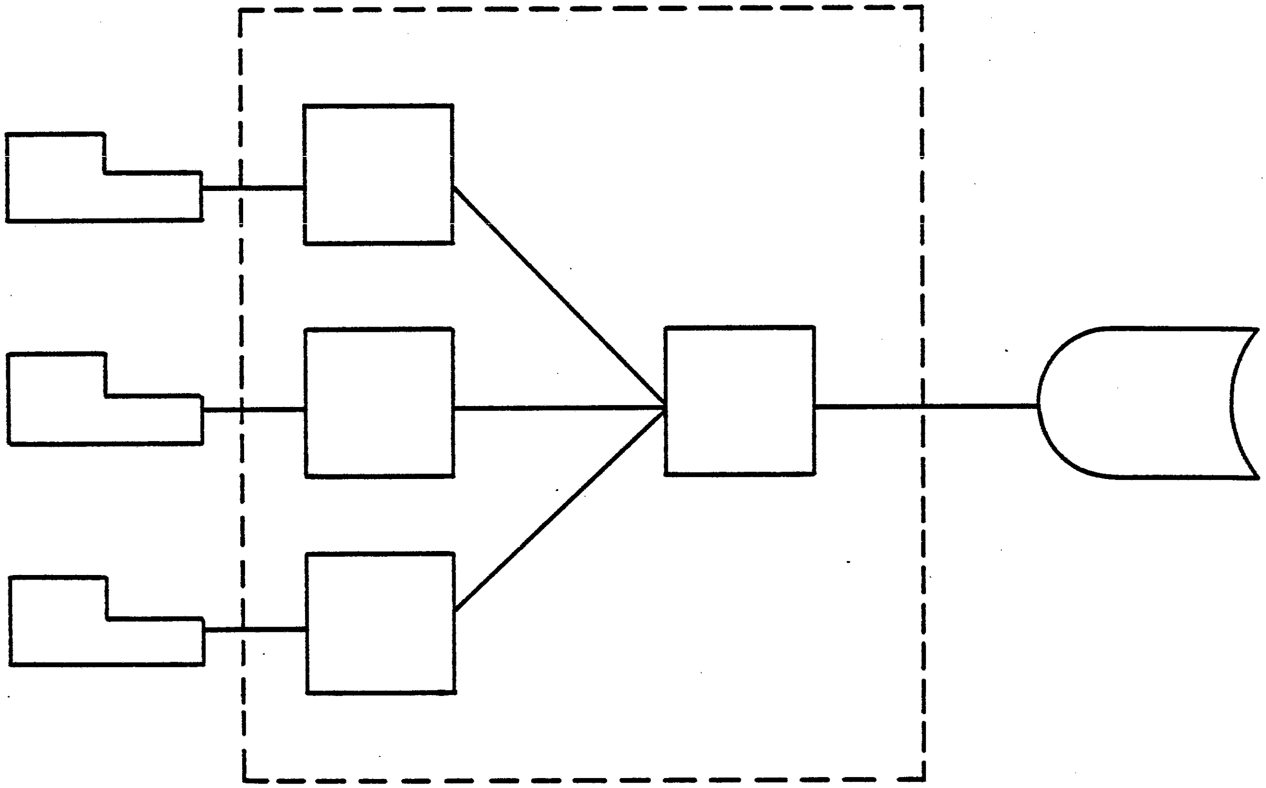
- Define a "transaction".
- List data base managers that interface with TAF.
- Define a "task".
- List the capabilities of TAF.

REFERENCES:

NOS IMS - Chapter 16.


ON LINE
TRANSACTION
DATA:
STRUCTURED:
OFT REPEATED:
NON VARYING
OBJECT
PRGM

Conceptual requirement for a transaction system.



Multiple terminals

One data base

TRANSACTION SYSTEM

1. Characteristics

   Multiple Terminals

   One Data Base

   Repetitive Input with few variations

II. Types

Inquiry/Update	Mostly Inquiry	Mostly Update
Inventory Control	Credit Validation	Consumer Load
Reservation Systems	Stock Quotation	Data Collection
Brokerage Systems		

III. Implementation

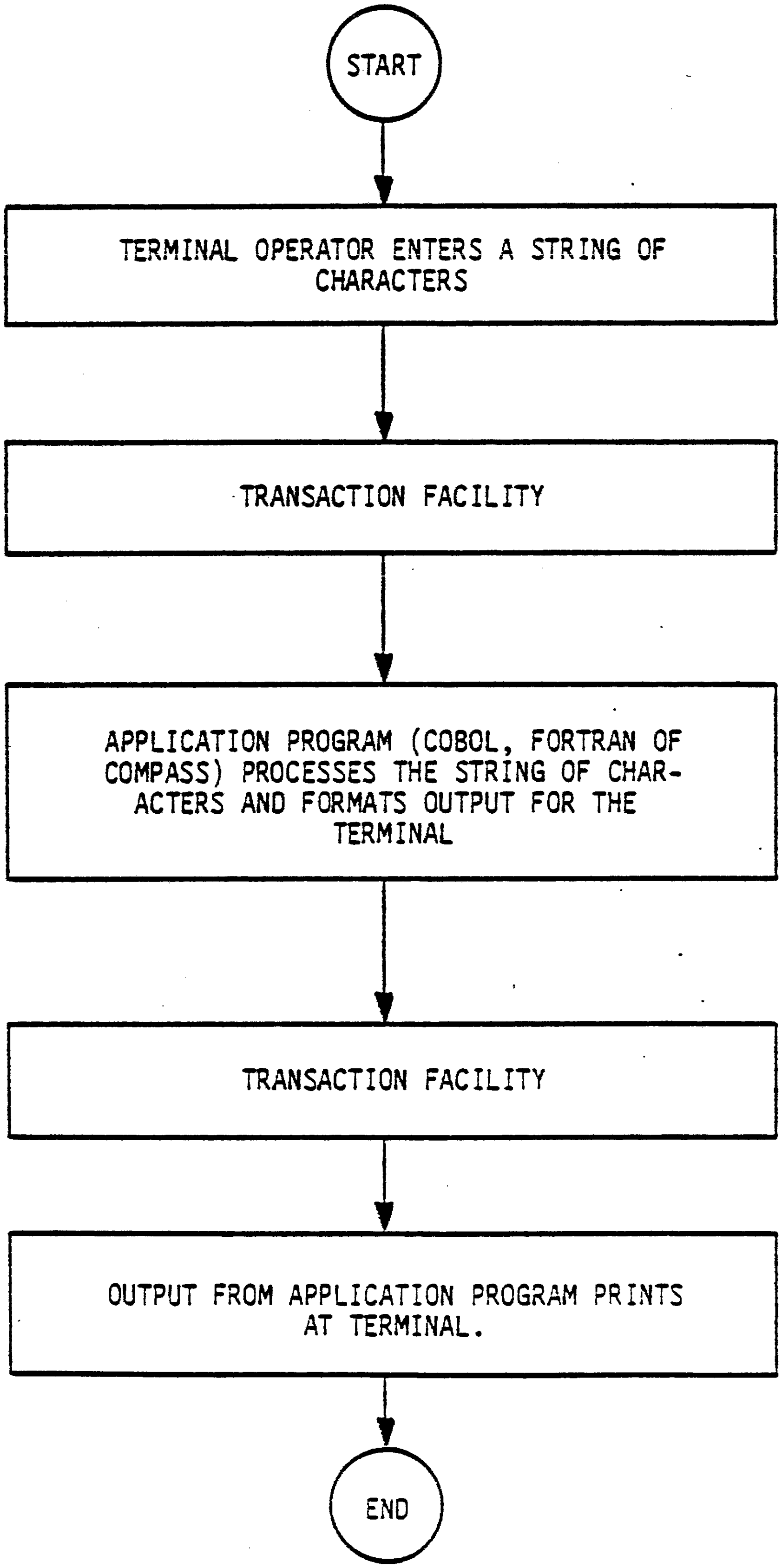
   Dedicated transaction processing systems

   Add transaction processing to an existing batch/time sharing system

TAF

- Looks like one job to the operating system
- Allows up to 31 subcontrol points
- Allows multiple data bases (multiple users)
- Validates data base accesses on a terminal basis
- Uses a separate task library for each data base
- Schedules tasks
- Contains and controls execution of the data manager(s)
- Regulates its field length

A TRANSACTION



FIVE COMPONENTS OF THE NOS TRANSACTION FACILITY

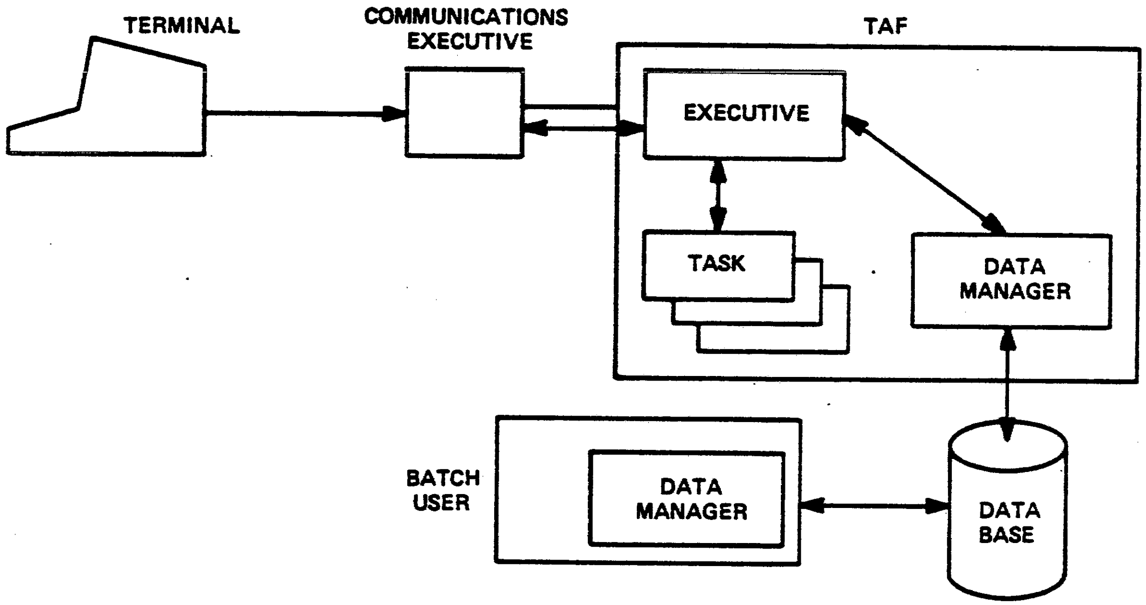
I. NAM or T/S - The NOS communications executive program that controls terminal input/output.

II. TAF - The NOS transaction executive program that coordinates and controls the execution of user application programs and the data manager(s).

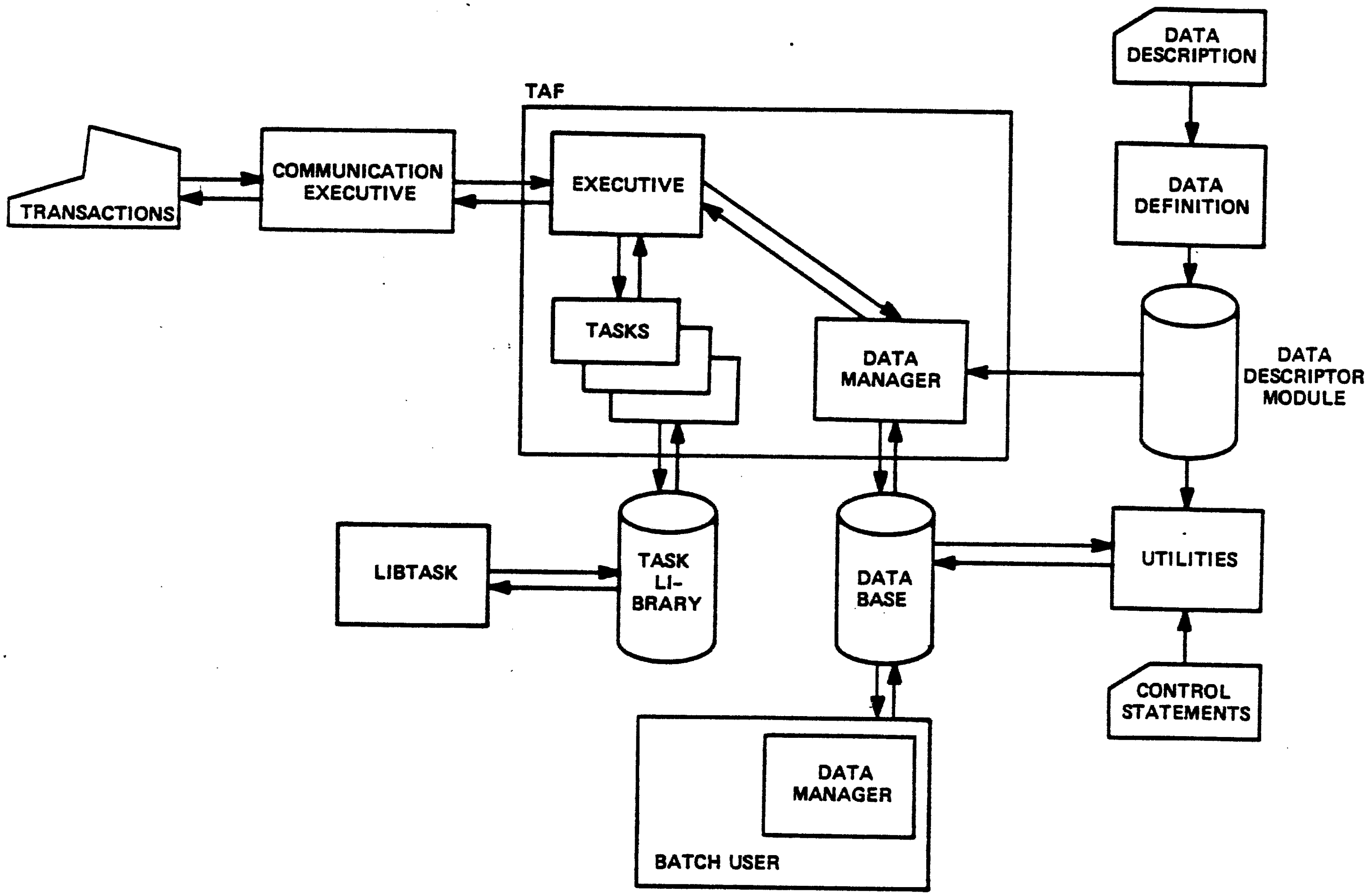
III. Data Manager - When operating as a subset of TAF, processes all data manager requests issued by user application programs. In this manner it can provide record level interlocks that allow concurrently-executing programs to access the same file (but not the same record within the file).

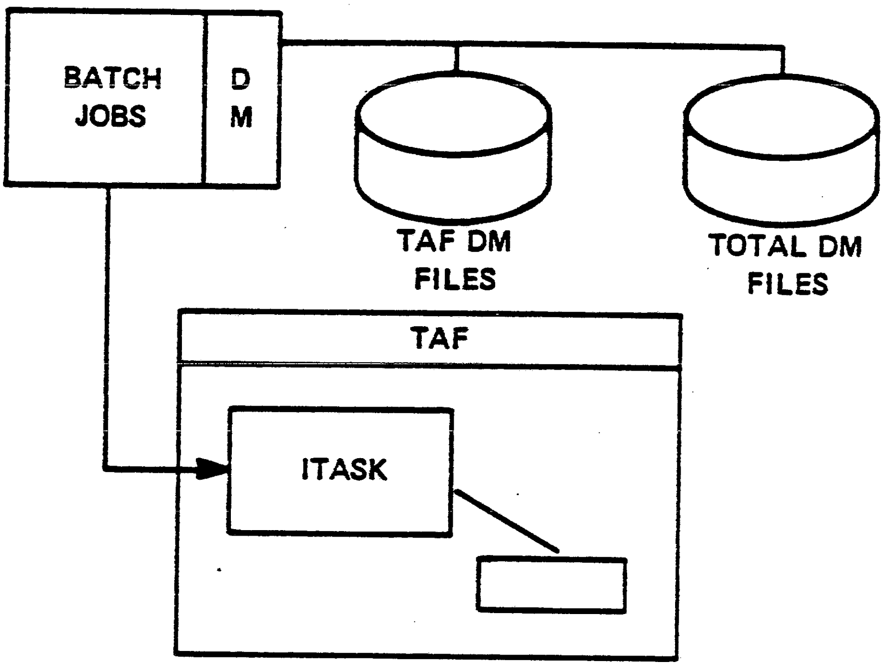
IV. User Tasks - Programs written in COBOL or FORTRAN that process transaction terminal input.

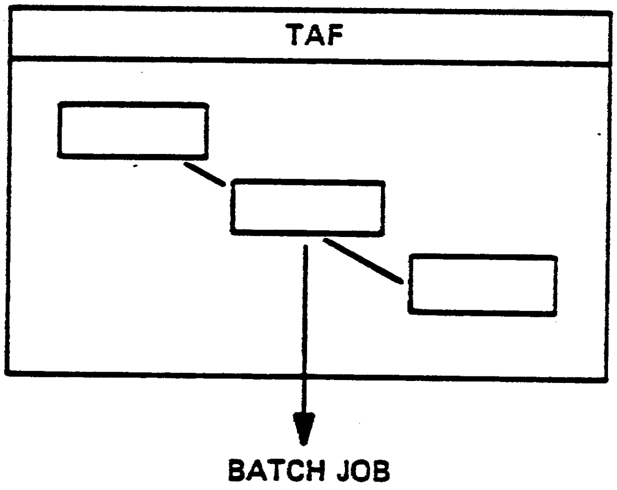
V. Utility Programs - Auxiliary operating system programs used for the creation and maintenance of data bases and task libraries.

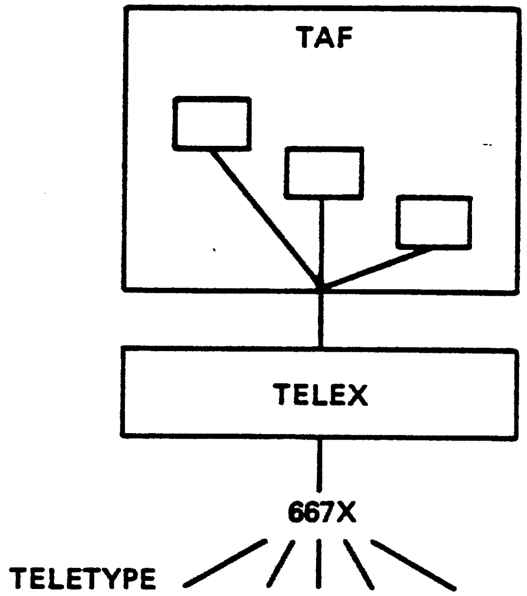

TERMINAL
COMMUNICATIONS EXECUTIVE
TAF
EXECUTIVE
TASK
DATA MANAGER
BATCH USER
DATA MANAGER
DATA BASE

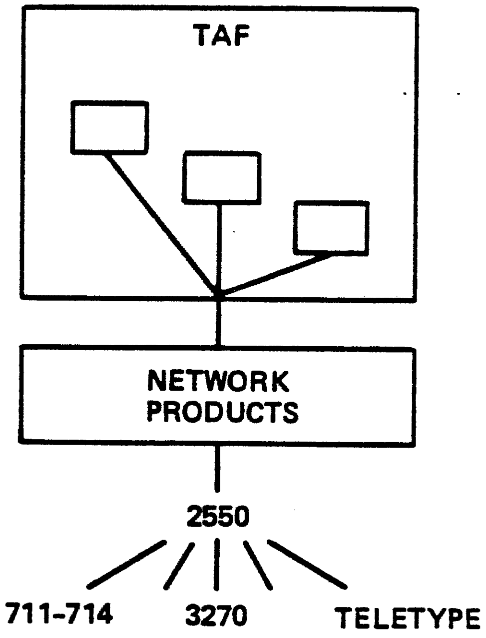
TAF ENVIRONMENT

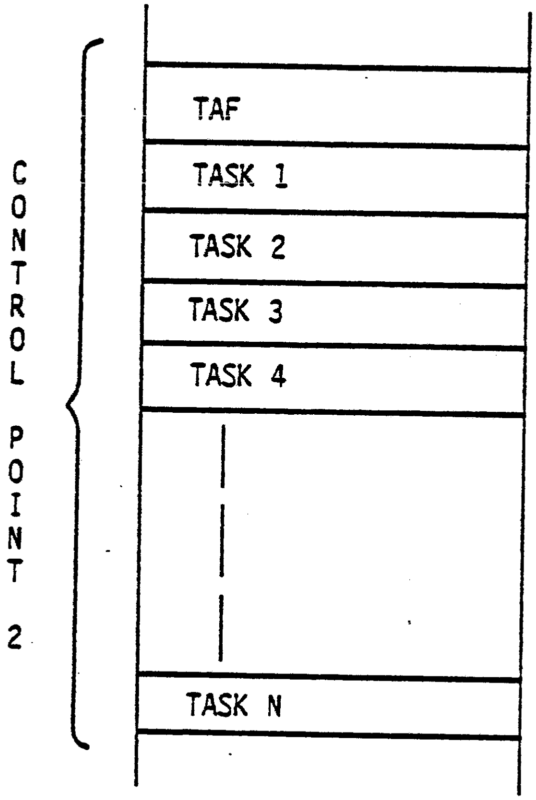


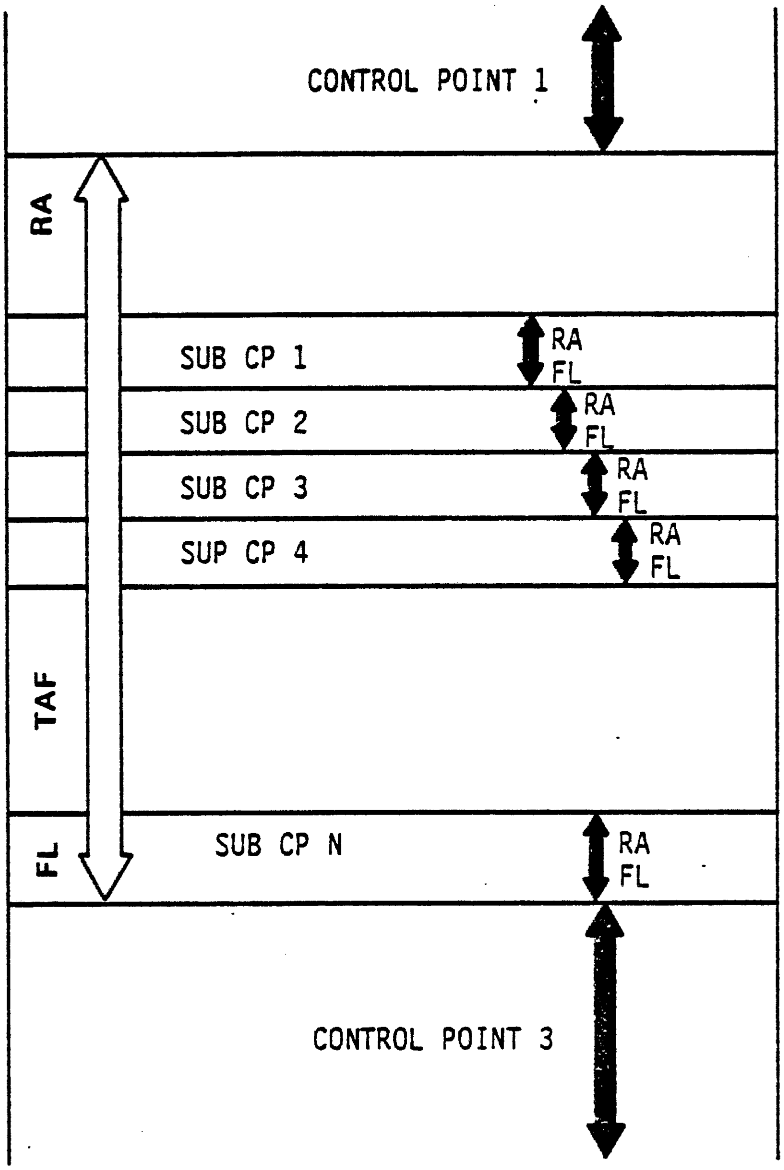

BATCH JOBS
D M
TAF DM FILES
TOTAL DM FILES
TAF
ITASK

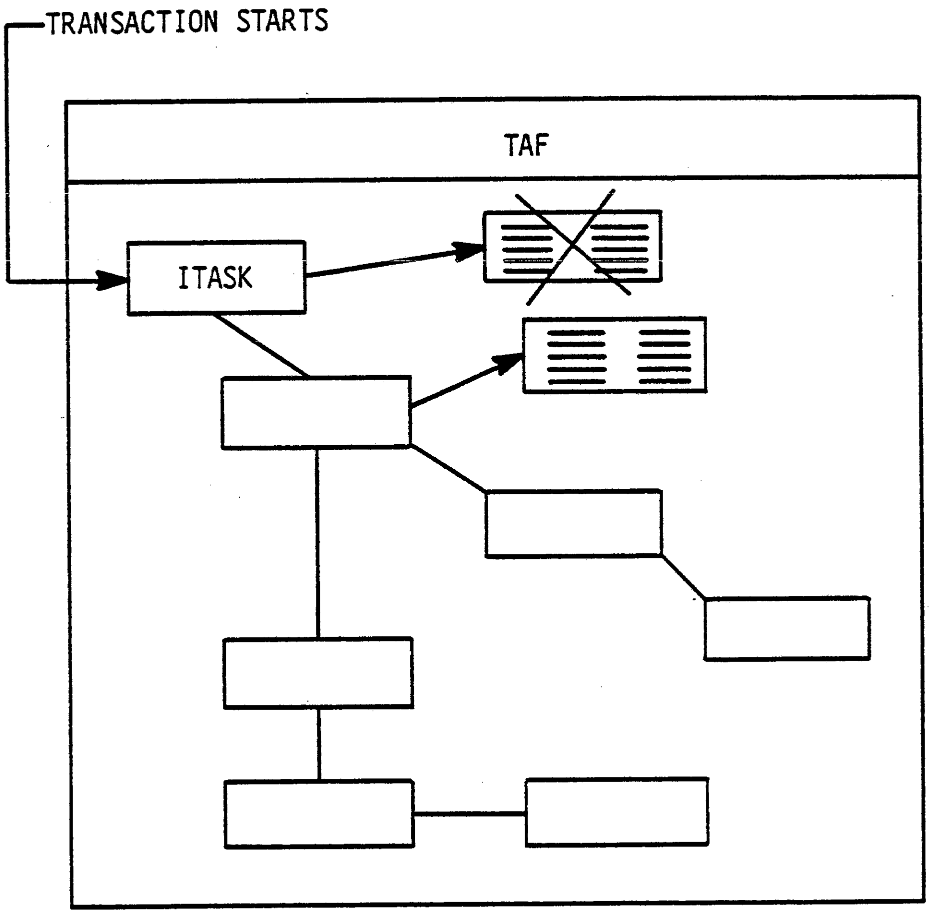

TAF
BATCH JOB

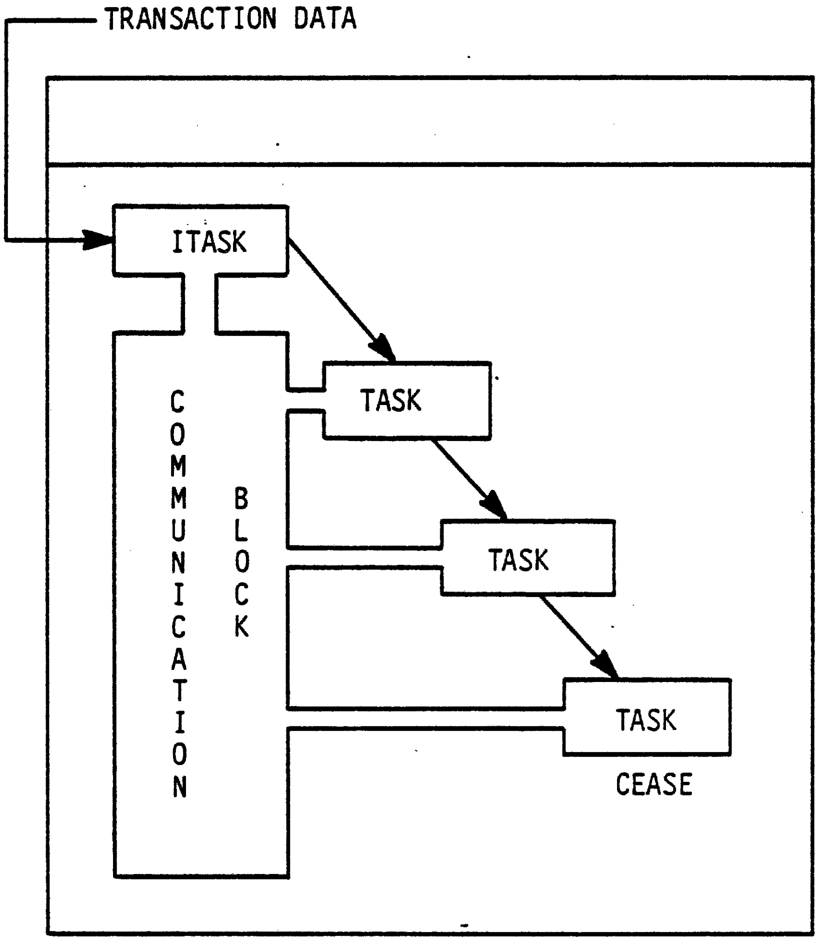

TAF
TELEX
667X
TELETYPE


TAF
NETWORK PRODUCTS
2550
711-714
3270
TELETYPE


CONTROL POINT 2
TAF
TASK 1
TASK 2
TASK 3
TASK 4
TASK N


CONTROL POINT 1
RA
SUB CP 1
RA
FL
SUB CP 2
RA
FL
SUB CP 3
RA
FL
SUP CP 4
RA
FL
TAF
FL
SUB CP N
RA
FL
CONTROL POINT 3


TRANSACTION STARTS
TAF
ITASK


TRANSACTION DATA
ITASK
TASK
TASK
TASK
CEASE
COMMUNICATION BLOCK

QUESTION SET LESSON 26

1. How does an end user (at a terminal) view transaction processing?

2. What is a "transaction"?

3. What is a "task" and how does it relate to a transaction?

4. What are the five components of TAF?

5. Can a data base be accessed from both a batch job and a terminal under TAF?

6. Why use a transaction executive?

LESSON 27
MULTIMAINFRAME

LESSON PREVIEW:

This lesson presents the NOS Multimainframe package: the support theory and how NOS shares permanent files using ECS as a link device.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be able to:

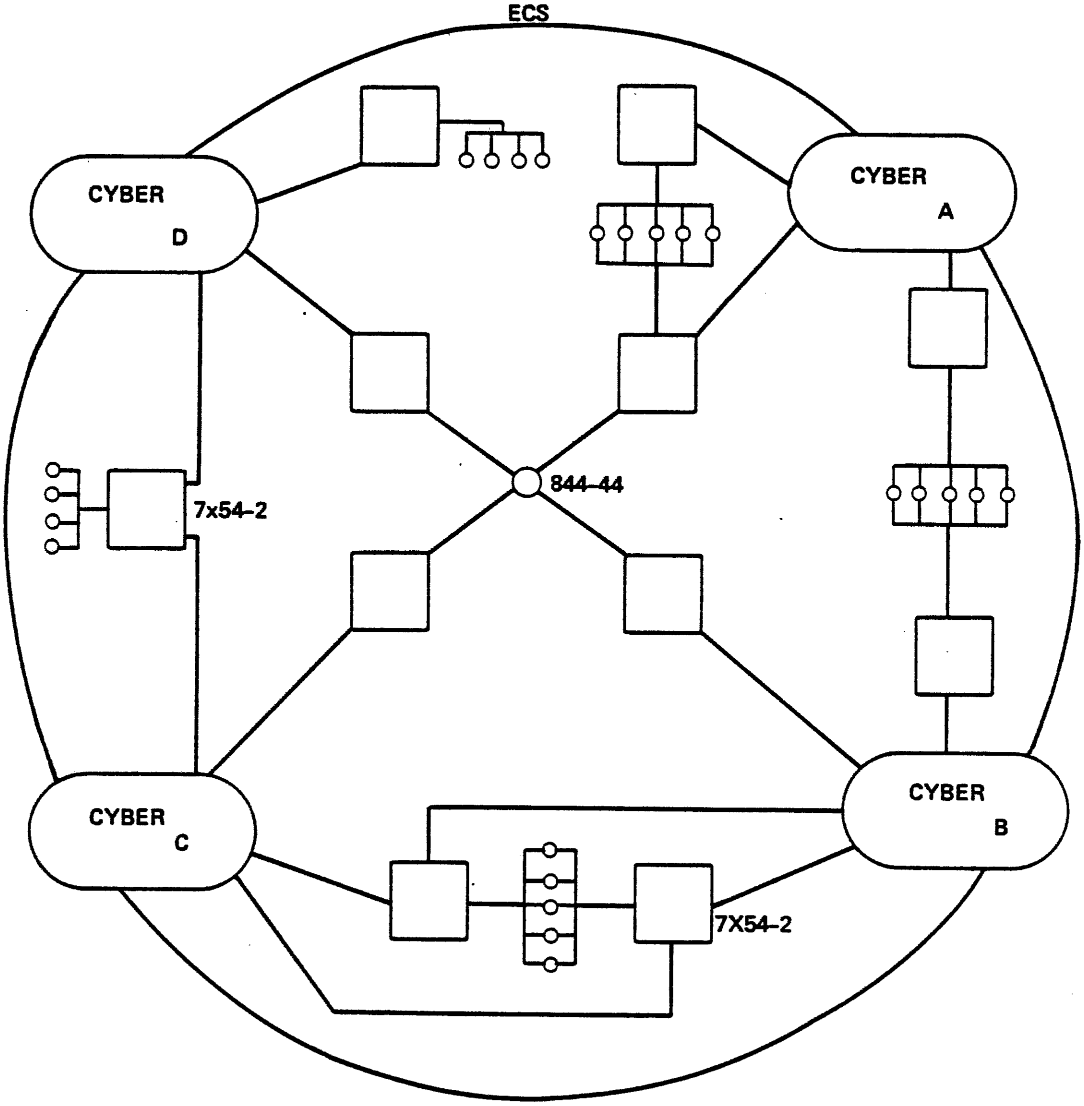
- Configure a MMF shared permanent file system.
- Discuss how the mass storage tables (MST/TRT/MRT) are accessed and updated in a MMF complex.
- Discuss the processing of a down machine, including its removal from and re-introduction into the MMF complex.
- Discuss the usage of ECS as the MMF link device.

REFERENCES:

NOS IMS, Chapter 21 NOS SMRM, 8

MULTI-MAINFRAME

- SHARED PERMANENT FILES
- 2 TO 4 MAINFRAMES
- ECS AS LINK



MULTIMAINFRAME TABLES
INTERMACHINE COMMUNICATION AREA

000 ⋮ 006	communication area 0
007 ⋮ 015	communication area 1
016 ⋮ 067	⋮
070 ⋮ 076	communication area 10

Each communication area has the following format.

	59	47	35	23	11 0
000	FN	▨	MI	MP	MD
001	message word 1				
002	message word 2				
003	message word 3				
004	message word 4				
005	message word 5				
006	message word 6				

FN	Intermachine function number.
MI	Machine initiating request.
MP	Machines to process request.
MD	Machines done processing request.

Sector 16_8 of the ECS label track is defined as follows:

	59	47	11 0
000	MMFL for mainframe 1		
001	MMFL for mainframe 2		
002	MMFL for mainframe 3		
003	MMFL for mainframe 4		
004	multimainframe 1 system time		
005	multimainframe 2 system time		
006	multimainframe 3 system time		
007	multimainframe 4 system time		
010	next DAT track	▨	DAT count
011	▨	▨	FAT count
012 • • •	One word per flag register bit. Each word contains the MMFL word of the machine which currently has the corresponding flag register interlock.		
033			
034	machine 1 requests		
035	machine 2 requests		
036	machine 3 requests		
037	machine 4 requests		
040	machine 1 requests		
041	machine 2 requests		
042	machine 3 requests		
043	machine 4 requests		
044 • • • 067	unused		
070 • • • 077	installation area		

MMF - DAT TRACK CHAIN (ECS)

Track N

0000 ⋮ 0777	device access table (DAT)
1000	fast attach table (FAT)

Track M (same format for each device)

0000 ⋮ 0011	MST for shared device (global area)
0012 ⋮ 0017	local area for machine index 1
0020 ⋮ 0025	local area for machine index 2
0026 ⋮ 0033	local area for machine index 3
0034 ⋮ 0041	local area for machine index 4
0042 ⋮ 0077	unused
0100 ⋮ 1077	TRT for device
1100 ⋮ 1177	MRT1 (machine recovery table)
1200 ⋮ 1277	MRT2
1300 ⋮ 1377	MRT3
1400 ⋮ 1477	MRT4

MMF - ECS FLAG REGISTER FORMAT

59	17 0
0	flag register

Bit Set	Name	Description
17-12	---	Reserved.
11	COMI	CPUMTR intermachine communication request present.
10	CIRI	CPUMTR interlock recovery.
9	FATI, PFNI	FAT and PFNL interlock.
8	IFRI	Intermachine function request interlock.
7	BTRI	Block transfer in progress.
6	PRSI	Deadstart ECS preset in progress.
5	DATI	Device access table interlock.
4	TRTI	TRT interlock; machine. Specified by bits 3-0 is requesting a TRT interlock.
3-0	---	Machine mask indicating which machine has TRT interlock bit set.

DEVICE ACCESS TABLE (DAT) ENTRY

	59	17	11 0
000	family name/pack name	dn	MST pointer
001	0		status

dn	Device number.
MST pointer	If zero, device is not shared.
status	Bits 11-5 are reserved, bit 4 is set if recovery is in progress, and bits 3-0 are machine mask of machines accessing device.

FAST ATTACH TABLE (FAT) ENTRY - GLOBAL

	59	47	35	23	17	11 0
000	fast attach file name				▨	
001	▨	first trk	RM	RA	R	▨
002	mach.1 ID	▨	RM	RA	R	▨
003	mach.2 ID	▨	RM	RA	R	▨
004	mach.3 ID	▨	RM	RA	R	▨
005	mach.4 ID	▨	RM	RA	R	▨
006	family name				dn	▨
007	0					

RM	READMD users.
RA	READAP users.
R	Read/write users.
dn	Device number.

PFNL ENTRY FORMAT - GLOBAL

000	0
001	PFNL (global)
002	PFNL for mainframe 1
003	PFNL for mainframe 2
004	PFNL for mainframe 3
005	PFNL for mainframe 4
006	0
007	0

The first entry of the FAT is an 8-word entry of PFNL words in the preceding format.

ECS

FLAG REGISTER INTERLOCKS

TRTI	SECURED WHENEVER UPDATING MST OR TRT IN ECS. MACHINE MASK BIT IS ALSO SET FOR THE MACHINE.
PRSI	SECURED WHEN UPDATING MACHINE ID WORDS IN ENVIRONMENT TABLE. OCCURS FOR ALL LEVELS OF DEADSTART.
BTRI	SECURED WHEN MACHINE WANTS TO PERFORM A BLOCK TRANSFER VIA ECS STORAGE MOVE AREA.
IFRI	SECURED BY CPUMTR WHEN ENTERING AN INTERMACHINE FUNCTION.
FATI/PFNI	SECURED BY CPUMTR OR MREC TO UPDATE PFNL WORDS OR TO UPDATE EXISTING FAST ATTACH ENTRIES.
DATI	SECURED BY ANY PROGRAM THAT IS SEARCHING, ADDING ENTRIES TO OR DELETING ENTRIES FROM THE FAT OR THE DAT, OR BY ANY PROGRAM ALTERING THE DAET OR FAET ENTRIES.
CIRI	SECURED BY CPUMTR WHEN CLEANING UP FLAG REGISTER INTERLOCKS FOR A DOWN MACHINE.
COMI	SECURED BY CPUMTR WHEN COMMUNICATION PROCESSING IS REQUESTED OF OTHER MACHINES.

ECS FLAG REGISTER

COMMUNICATION REGISTER: COMPUTER TO COMPUTER COMMUNICATION WITHOUT ECS MEMORY REFERENCE.

FLAG REGISTER IS 18 BITS IN LENGTH AND PART OF THE ECS CONTROLLER.

FLAG REGISTER OPERATIONS ARE ECS RE/WE WITH BIT 23 OF ECS ADDRESS SET=1. BITS 22-21 OF ECS ADDRESS SPECIFY THE FLAG REGISTER OPERATION.

BITS 17-0 OF ECS ADDRESS IS THE FLAG WORD. BITS ARE CLEARED/ ENTERED IN THE FLAG REGISTER FROM THE FLAG WORD DEPENDING ON THE FUNCTION.

- BIT BY BIT COMPARISON OF FLAG REGISTER AND FLAG WORD. IF BITS SET IN THE WORD ARE CLEAR IN THE REGISTER, THEN ALL THE BITS SET IN THE FLAG WORD ARE SET IN THE FLAG REGISTER. IF NOT ALL BITS ARE CLEAR IN THE FLAG REGISTER FOR BITS SET IN THE FLAG WORD, THE ERROR HALF-EXIT IS TAKEN.

(R) = 010	(R) = 010
(W) = 101	(W) = 111
(A) = 111	ABORT

ECS FLAG REGISTER

1 BITS IN THE FLAG WORD ARE SET IN THE FLAG REGISTER

(R) = 010
(W) = 100
(A) = 110

2 BIT BY BIT COMPARISON OF FLAG REGISTER AND FLAG WORD. IF ALL BITS IN THE FLAG REGISTER ARE CLEAR FOR BITS SET IN THE FLAG WORD, THEN THE FUNCTION HAS COMPLETED NORMALLY. OTHERWISE, THE ERROR HALF-EXIT IS TAKEN. THIS IS THE SAME LOGIC AS FUNCTION 0 EXCEPT NO DATA IS CHANGED IN THE FLAG REGISTER.

(R) = 010	(R) = 010
(W) = 101	(W) = 111
EXECUTE NEXT WORD	ABORT
(A) = 010	(A) = 010

3 BITS IN THE FLAG WORD ARE CLEARED IN THE FLAG REGISTER.

(R) = 110
(W) = 101
(A) = 010

MRT

THE MACHINE RECOVER TABLE (MRT) WORKS HAND IN HAND WITH THE TRACK RESERVATION TABLE (TRT) TO INDICATE THE LOCAL OR SHARED STATUS OF THE TRACKS ON A MASS STORAGE DEVICE.

TRT	MRT	
0	0	TRACK NOT INTERLOCKED OR IT IS LOCAL TO ANOTHER MF
0	1	FIRST TRACK OF A FILE LOCAL TO THIS MAINFRAME
1	0	TRACK INTERLOCKED BY ANOTHER MAINFRAME
1	1	TRACK INTERLOCKED BY THIS MAINFRAME

THE MRT IS USED TO:

1. CLEAR INTERLOCKS ON PRESERVED FILES THAT ARE HELD BY A DOWN MACHINE.

1. CLEAN UP LOCAL FILE USAGE BY A DOWN MACHINE UNDER MREC CONTROL.

MRT/MST/TRT UPDATING

CPUMTR UPDATES LOCAL MACHINE COPIES OF THE TRT AND MST FROM ECS AND UPDATES THE ECS COPIES OF THE TRT, MST AND MRT WHEN PROCESSING MONITOR FUNCTIONS DLKM, DTKM, RTCM AND MOST SUBFUNCTIONS OF STBM.

EACH MACHINE HAS A LOCAL COPY OF THE MRT AS WELL AS AN ECS COPY. AS THE MRT NEEDS TO BE UPDATED, ONLY THE WORD CONTAINING THE BIT TO BE CHANGED IS WRITTEN TO ECS.

EACH MACHINE HAS A LOCAL COPY OF THE TRT. IF THE MACHINE WAS NOT THE LAST UPDATER OF THE TRT, THE TRT IS READ FROM ECS. WHEN UPDATING THE TRT, THE TRT WORDS FROM THE LOCAL TRT ARE WRITTEN TO ECS. THE AMOUNT WRITTEN IS FROM THE FIRST TRT WORD ALTERED TO THE LAST ALTERED WORD. THE SDGL WORD IN THE MST ECS COPY IS UPDATED TO INDICATE THE LAST UPDATER OF THE TRT.

FOR THE MOST PART, SYSTEM ROUTINES DO NOT KNOW THAT THE MACHINE IS PART OF AN MMF COMPLEX. THEY ISSUE MONITOR FUNCTIONS AND CPUMTR DOES THE REQUIRED OPERATION.

MRT/MST/TRT UPDATING

READING TRT

1. DETERMINE IF DEVICE IS SHARED (IF SO, THERE WILL BE AN ECS ADDRESS IN MST WORD SDGL).

2. OBTAIN TRTI FLAG REGISTER INTERLOCK.

3. READ GLOBAL MST (FIRST THREE WORDS) ONLY FIRST THREE GLOBAL WORDS ARE NECESSARY; REST OF GLOBAL AREA IS ONLY USED TO IDENTIFY THE DEVICE-LABEL RECOVERY.

4. MST/TRT IS INTERLOCKED THROUGH SDGL. IF ANOTHER MACHINE'S MASK IS PRESENT, THE TRTI INTERLOCK IS RELEASED AND THE MONITOR FUNCTION REJECTED.

   IF THIS MACHINE HAD LAST ACCESSED THE MST/TRT (THIS INFORMATION IS ALSO IN SDGL) THERE IS NO NEED TO REREAD THE TRT NOW. OTHERWISE, THE TRT AND MST ARE READ FROM ECS, THEN MOVED TO THE LOCAL TRT/MST AREA.

   THE FIRST THREE GLOBAL MST WORDS ARE WRITTEN TO ECS, THUS <u>INTERLOCKING THE TRT/MST FOR THIS MACHINE</u>.

5. THE TRTI INTERLOCK IS NOW RELEASED.

MRT/TRT/MST UPDATING

RTCM

1. READ TRT AND INTERLOCK IT VIA SDGL IN MST.

2. FIND AVAILABLE TRACK. IF FIRST TRACK, SET MRT BIT (LOCAL AND ECS) FOR THIS TRACK.

3. RESERVE DESIRED NUMBER OF TRACKS IN LOCAL TRT. THE TRT WORD ADDRESSES FOR THE FIRST AND LAST TRACKS RESERVED ARE SAVED.

4. UPDATE THE ECS COPY OF THE TRT BY DOING A BLOCK WRITE FROM THE FIRST TRT WORD USED TO THE LAST TRT WORD USED.

5. CLEAR INTERLOCK IN SDGL BY REWRITING THE THREE GLOBAL MST WORDS.

MRT/TRT/MST UPDATING

DLKM/DTKM

1. READ TRT AND INTERLOCK IT VIA SDGL.
2. PERFORM OPERATION ON LOCAL TRT. THE TRT WORD ADDRESSES FOR THE "FIRST" AND "LAST" TRACKS RESERVED ARE SAVED.
3. IF ENTIRE TRACK CHAIN IS BEING RELEASED, THE MRT BIT (LOCAL AND ECS) IS CLEARED.
4. UPDATE ECS COPY OF THE TRT BY DOING A BLOCK WRITE FROM THE "LOWEST" TRT WORD USED TO THE "HIGHEST" WORD USED.
5. CLEAR INTERLOCK IN SDGL BY REWRITING THE THREE GLOBAL MST WORDS.

MRT/TRT/MST UPDATING

STBM

STBM HAS A VARIETY OF FUNCTIONS THAT UPDATE TRT AND MST FIELDS.

THE LOGIC FOLLOWED FOR TRT FUNCTIONS IS THE SAME AS FOR RTCM/DLKM/DTKM EXCEPT THAT ONLY ONE TRT WORD IS BEING UPDATED.

THE STBM SUBFUNCTIONS STIS/CTIS (SET/CLEAR TRACK INTERLOCK) AND SPFS/CPFS (SET/CLEAR PRESERVED FILE STATUS) WILL CAUSE THE MRT BIT (LOCAL AND ECS) TO BE SET/CLEARED FOR THE TRACK.

IAUM

CPUMTR FUNCTION IAUM PERFORMS A VARIETY OF FUNCTIONS WITH PERMANENT FILE CONTROLS, IN PARTICULAR, FAST ATTACH FILES AND THE PFNL WORD.

IN THE MULTIMAINFRAME ENVIRONMENT, IAUM WORKS ON THE FAT (FAST ATTACH TABLE) TRACK IN ECS.

IAUM WILL

1. OBTAIN FATI INTERLOCK IN ECS FLAG REGISTER
2. READ ENTRY (8 WORDS) FROM FAT. (RECALL THAT THE FIRST ENTRY ON THE FAT IS THE GLOBAL PFNL ENTRY)
3. UPDATE THE ENTRY AND WRITE IT BACK INTO ECS
4. RELEASE FATI INTERLOCK

MMF DEADSTART

THE PRESENCE OF A LINK DEVICE IS THE "SYSTEM FLAG" THAT INDICATES THAT A MACHINE IS TO BE PART OF AN MMF COMPLEX.

THE FOLLOWING CMRDECK ENTRIES ARE RELATED TO MMF PROCESSING:

LINK=EQ.	DEFINES EQ AS THE LINK DEVICE; EQ IS ECS EST ORDINAL.
SHARE=EQ_1,EQ_2,..EQN.	DEFINES EQ_I TO BE SHARED BY OTHER MACHINES IN THE MMF COMPLEX.
PRESET,N.	REQUESTS THAT ECS TABLE SPACE BE ALLOCATED FOR N SHARED DEVICES.

AFTER PROCESSING THE CMRDECK, SET ALLOCATES/RECOVERS SPACE FOR THE TRT/MST/MRT FOR MASS STORAGE DEVICES.

FOR LEVEL 0, 1, 2 RECOVERY, THE MST IS INITIALIZED. FOR LEVEL 3 RECOVERY, ONLY INTERLOCKS (SDGL, PF UTILITY, DISK DRIVER) ARE CLEARED.

MMF DEADSTART

SET PASSES THE MULTIMAINFRAME AND RECOVERY INFORMATION TO STL. SOME OF THIS INFORMATION IS PASSED IN MMFL AND ERFL.

MID	NS	LD			MIN	MMFL

	RS	PI			ELT	ERFL

MID = MACHINE ID (USES CM MMFL IF LEVEL 3)
MIN = MACHINE INDEX
NS = NUMBER OF SHARED DEVICES
LD = LINK DEVICE (FROM CM MMFL IF LEVEL 3)
RS = RECOVERY LEVEL
P = PRESET
I = INITIALIZE
ELT = ECS LABEL TRACK (FROM ALGL LEVEL 3)

STL LOADS AND STARTS UP CPUMTR.

CPUMTR PROCESSES THE LINK DEVICE DEPENDING ON PRESET AND INITIALIZE STATUS AND RECOVERY LEVEL.

MMF DEADSTART

CPUMTR
PRESET MACHINE

CLEARS ECS FLAG REGISTER

SETS PRESET BIT IN FLAG REGISTER .PRSI

IF NOT INITIALIZING, VERIFIES LINK DEVICE LABEL TRACK.

CLEARS COMMUNICATION AND ENVIRONMENT SECTORS IN ECS LABEL TRACK

SETS MMFL AS FIRST ENTRY IN MFET

SETS INTERNAL MMF CLOCKS FROM STET

CLEARS PRESET BIT FROM FLAG REGISTER

MACHINE RECOVERY TABLE (MRT) ADDRESS SET BASED ON MACHINE INDEX

MMF DEADSTART

CPUMTR
NON-PRESET MACHINE

SETS PRESET BIT IN ECS FLAG REGISTER .PRSI

VERIFIES LINK DEVICE LABEL TRACK

IF LEVEL 3 RECOVERY, MMFL MUST AGREE WITH ENTRY IN MFET. IF SO, INTERNAL CLOCKS ARE RESET FROM STET AND FLAG REGISTER PRESET BIT CLEARED. ALL MMF INTERLOCKS PREVIOUSLY HELD BY THIS MACHINE ARE CLEARED AS FOLLOWS:

- .CIRI INTERLOCK IS OBTAINED
- ALL FLAG REGISTER BITS HELD BY THIS MACHINE ARE CLEARED
- .FATI INTERLOCK IS OBTAINED
- THE PFNL FOR THIS MACHINE IS CLEARED AND GLOBAL PFNL IS ADJUSTED
- .FATI INTERLOCK IS CLEARED
- ALL DEVICE INTERLOCKS FOR SHARED DEVICES ARE CLEARED (SDGL)
- LOCAL PF UTILITY ACTIVITY (ACGL) AND DEVICE INTERLOCKS (SDGL) ARE CLEARED FOR ALL MASS STORAGE DEVICES
- .CIRI INTERLOCK IS CLEARED

MMF DEADSTART

IF NOT A LEVEL 3 RECOVERY, THE MFET IS CHECKED TO DETERMINE IF THE MACHINE IS ALREADY IN THE MMF COMPLEX.

IF A MATCH OCCURS BETWEEN THE MACHINE'S MMFL AND AN ENTRY IN THE MFET AND THE DEADSTART IS A LEVEL 0, THE DIAGNOSTIC

"MID CURRENTLY ACTIVE"

IS DISPLAYED, THE .PRSI INTERLOCK RELEASED AND DEADSTART ABORTED.

IF THE MATCH OCCURRED AND THE DEADSTART IS A LEVEL 1 OR 2, THERE MUST BE A LINK EQUIPMENT DEFINED. IF SO, THE MMFL IS RESET IN MFET, THE INTERNAL CLOCKS RESET FROM STET AND THE .PRSI INTERLOCK RELEASED. IF NO LINK DEVICE IS PRESENT, THE DIAGNOSTIC:

"MID UNDEFINED IN ECS"

IS DISPLAYED, THE .PRSI INTERLOCK RELEASED AND DEADSTART ABORTED.

MMF DEADSTART

IF THE MATCH DID NOT OCCUR AND THE DEADSTART IS A LEVEL 0, IF AN EMPTY ENTRY WAS FOUND IN MFET, THE MMFL IS SET INTO MFET, THE INTERNAL CLOCKS RESET FROM STET, THE .PRSI INTERLOCK RELEASED. IF NO ENTRY WAS FOUND, THE DIAGNOSTIC

"MAXIMUM NUMBER OF MIDS ACTIVE"

IS DISPLAYED, THE .PRSI INTERLOCK RELEASED AND DEADSTART ABORTED.

IF THE MATCH DID NOT OCCUR AND THE DEADSTART IS A LEVEL 1 OR 2, THE DIAGNOSTIC

"MID UNDEFINED IN ECS"

IS DISPLAYED, THE .PRSI INTERLOCK RELEASED AND DEADSTART ABORTED.

OTHER DIAGNOSTICS THAT MAY BE ISSUED:

"ECS LABEL TRACK NOT FOUND"

VERIFICATION OF LABEL TRACK FAILED, THE .PRSI INTERLOCK IS RELEASED AND DEADSTART ABORTED.

"ECS READ/WRITE PARITY ERRORS"

ERRORS OCCURRED WHEN READING OR WRITING ECS MMF TABLES. THE .PRSI INTERLOCK IS RELEASED AND DEADSTART ABORTED.

MMF DEADSTART

WHEN CPUMTR COMPLETES, STL STARTS UP RMS. RMS RECEIVES DEADSTART INFORMATION FROM STL IN ITS INPUT REGISTER.

RMS	CP	LD	PI	EC	RS

CP = CONTROL POINT NUMBER

LD = LINK DEVICE EST ORDINAL

P = PRESET

I = INITIALIZE

EC = CPUMTR LINK DEVICE INITIALIZATION ERROR CODE

RS = RECOVERY LEVEL

MMF DEADSTART

RMS OBTAINS THE DATI INTERLOCK IN THE FLAG REGISTER IF LEVEL 0, 1 OR 2 RECOVERY.

IT THEN RECOVERS THE LINK DEVICE. IF NOT BEING PRESET, THE DAT IS READ FROM ECS. IF A PRESET IS BEING DONE, OMF IS CALLED.

OMF INITIALIZES THE DAT TRACK.

RMS THEN READS THE LABELS FOR ALL MASS STORAGE DEVICES. THIS IDENTIFIES WHAT DEVICES EXIST ON THE SYSTEM. LABEL READING CONSISTS PRIMARILY OF READING THE MST TO A CENTRAL MEMORY AREA.

RMS (LEVEL 0, 1, 2) THEN RECOVERS THE MASS STORAGE DEVICES. THE ACTION TAKEN BEGINS WITH CHECKING THE DAT FOR DEVICES AND WHETHER THEY WERE LAST ACCESSED BY THIS MACHINE. IF A DEVICE WAS FOUND ACCESSED BY ANOTHER MACHINE, ITS MST IS READ FROM ECS TO THE CM AREA. THE DAT ENTRY FOR THE DEVICE IS UPDATED BASED ON THIS MACHINE'S ACCESS TO IT AND RECOVERY LEVEL. THE DAT INDEX IS SET IN THE MST MDGL WORD.

MMF DEADSTART

RMS THEN PROCESSES THE TRT. IF A DEVICE IS SHARED, AND HAS A TRT IN ECS, IT WILL BE READ FROM ECS TO THE TRT AREA IN CENTRAL MEMORY RECOVERY AREA. OTHERWISE, THE TRT IS READ FROM THE DEVICE'S LABEL TRACK.

THE TRT IS THEN EDITED (LEVEL 0) UNLESS IT IS A PREVIOUSLY ACCESSED SHARED DEVICE. THE EDITING REMOVES NON-PRESERVED TRACK LINKAGES WHILE COPYING THE TRT FROM THE RECOVERY AREA TO THE ACTUAL TRT RESIDENCE.

FOR NON-LEVEL 0 DEADSTARTS, THE TRT IS SIMPLY COPIED FROM THE RECOVERY AREA TO THE ACTUAL TRT RESIDENCE, UNLESS IT IS A PREVIOUSLY ACCESSED SHARED DEVICE.

THE MST IS THEN WRITTEN TO ITS RESIDENCE FROM THE RECOVERY AREA.

THE TRACK COUNT IS ADJUSTED, UNLESS IT IS A PREVIOUSLY ACCESSED SHARED DEVICE.

THE DAT IMAGE IS THEN WRITTEN BACK TO ECS FROM THE RECOVERY AREA, UNLESS THIS IS A LEVEL 3 RECOVERY.

MMF DEADSTART

RMS NOW UPDATES THE ECS TABLES. IF THE DEVICE HAD NOT BEEN PREVIOUSLY ACCESSED BUT IS SHARED, THE MST/TRT IS WRITTEN TO ECS AND THE MRTs ARE CLEARED.

IF THE DEVICE WAS A SHARED DEVICE PREVIOUSLY ACCESSED (BY THIS MACHINE), THEN THE MRT IS EDITED AND LOCAL TRACK INTERLOCKS ARE CLEARED. IF THIS IS A LEVEL 0 DEADSTART, THE TRACK CHAINS FOR NON-PRESERVED FILES ARE RELEASED.

THE MRT IS THEN WRITTEN TO ECS.

IF THE DEADSTART IS A LEVEL 1 OR 2, THE DATI FLAG REGISTER INTERLOCK IS RELEASED. LEVEL 0 DEADSTART DATI WILL BE RELEASED LATER AND THE DATI INTERLOCK WAS NOT OBTAINED FOR A LEVEL 3 RECOVERY.

IF ANY DEVICES WERE RECOVERED AND THIS IS A LEVEL 0, 1 OR 2 DEADSTART, THE PERMANENT FILE SUBSYSTEM IS VALIDATED. THIS MEANS THAT THERE CAN BE NO DUPLICATES IN DEVICE NUMBERS WITHIN A FAMILY OR DUPLICATE AUXILIARY PACKS MOUNTED.

REC IS THEN LOADED INTO THIS PPU AND CONTROL GIVEN TO IT.

MASS STORAGE DEVICE RECOVERY

	STAND ALONE	SHARED - NOT IN DAT	SHARED - IN DAT	
LEVEL 0	2,4,6, 7,8,14	1,4,6,7 8,9,10,14	3,11	NOT PREVIOUSLY ACCESSED
	N/A	N/A	11,12,13	PREVIOUSLY ACCESSED
1 & 2	2,4,7	1,4,5,7,9	3,11	NOT PREVIOUSLY ACCESSED
	4,7	N/A	11,13	PREVIOUSLY ACCESSED
3	ERROR	ERROR	ERROR	NOT PREVIOUSLY ACCESSED
	4,7	N/A	11,13	PREVIOUSLY ACCESSED

MASS STORAGE DEVICE RECOVERY

1. DATA ENTRY NOT FOUND. MAKE DAT ENTRY INDICATING THIS MACHINE ONLY ACCESSOR.
2. DAT ENTRY NOT FOUND. MAKE DAT ENTRY INDICATING THIS MACHINE IS ACCESSING BUT NO ECS POINTER IN MST (NOT SHARED).
3. ADD INDICATION THAT THIS MACHINE IS ACCESSING TO EXISTING DAT ENTRY.
4. RETRIEVE MST FROM LABEL AND PRESET INTO ECS. RETRIEVE TRT FROM LABEL.
5. SET MRTs FROM DEVICE INTO ECS.
6. EDIT TRT.
7. CLEAR TRACK INTERLOCKS FOR ALL MACHINES.
8. CLEAN UP SYSTEM SECTOR FOR ALL MACHINES.
9. SET TRT INTO ECS.
10. CLEAR MRTs FOR ALL MACHINES.
11. RETRIEVE TRT/MST FROM ECS. GET LOCAL MST FROM LABEL AND CLEAN IT UP.
12. PROCESS MRT DROPPING LOCAL TRACKS.
13. PROCESS MRT CLEARING LOCAL TRACK INTERLOCKS.
14. BUILD IQFT.

MMF DEADSTART

REC RECOVERS REMNANTS OF THE PREVIOUS SYSTEM OPERATION FROM THE SYSTEM TABLE TRACK. THE SYSTEM TABLE TRACK WAS BUILT AS PART OF A SYSTEM CHECKPOINT OPERATION BY 1CK.

FOR LEVEL 1 AND 2 RECOVERIES, THE EST AND FNT ARE RESTORED FROM THE SYSTEM TABLE TRACK. FOR LEVEL 2 RECOVERY, THE EST ENTRIES BUILT BY RMS ARE USED FOR MASS STORAGE EQUIPMENTS INSTEAD OF THOSE FROM THE SYSTEM TABLE. UNTIL THIS TIME, ALL RECOVERY OPERATIONS HAVE TAKEN PLACE USING INFORMATION FROM THE CMRDECK. ANY CHANGES IN THE CMRDECK FROM THE INITIAL LEVEL 0 MUST BE REFLECTED IN THE RECOVERY DEADSTART CMRDECK, OTHERWISE THE RECOVERY MAY NOT BE SUCCESSFUL.

FOR LEVEL 1 RECOVERY, THE DAYFILE POINTERS AND BUFFERS ARE RE-ESTABLISHED FROM THE SYSTEM TABLE.

FOR LEVEL 1 AND 3 RECOVERY, THE LIBRARY TABLES ARE RESTORED FROM THE SYSTEM TABLE. THIS IS THE RE-SETTING OF DIRECTORY TABLES (PLD, CLD, LBD) AND CENTRAL RESIDENT ROUTINES (RPL, RCL). UNTIL THIS TIME, THE ROUTINE USED IN RECOVERY CAME FROM THE TAPE DEADSTARTED.

MMF DEADSTART

REC THEN PROCESSES CONTROL POINTS AND THE FNT; FILES ARE DROPPED OR RE-QUEUED AS NEEDED. CONTROL POINTS ARE RE-INITIALIZED.

THE SYSTEM DAYFILE, ERRLOG AND ACCOUNT FILES ARE RECOVERED/ INITIALIZED WITH THE TIME AND TYPE OF DEADSTART.

AT THIS TIME THE "RECOVERY COMPLETE" MESSAGE IS MADE AVAILABLE AND DSD INFORMED OF THIS FACT.

FOR LEVEL 0 AND 2 DEADSTARTS, REC NOW WAITS FOR THE SYSTEM TAPE TO BE LOADED AND SYSEDIT TO COMPLETE ITS TABLE BUILDING.

ONCE THE TAPE HAS BEEN LOADED (IF REQUIRED), REC RECOVERS PRESERVED FILE CHAINS FOR ALL AVAILABLE MASS STORAGE. IT IS AT THIS POINT IN TIME, THAT THE DATI FLAG REGISTER INTERLOCK FOR A LEVEL 0 DEADSTART IS CLEARED.

JOBS ARE RESTARTED AND SCHEDULAR CONTROL RESET.

FOR A LEVEL 0 DEADSTART, THE SYSTEM TABLE TRACK IS BUILT BY A CHECKPOINT REQUEST TO 1CK WHICH ALSO CAUSES EACH MASS STORAGE DEVICE TO BE CHECKPOINTED.

THE SYSTEM IS NOW FULLY OPERATIONAL.

DOWN MACHINE

CPUMTR IS CALLED BY MRT WITH FUNCTION ARMF TO ADVANCE RUNNING TIME AND STATUS ECS CLOCKS.

CPUMTR WRITES THIS MACHINE'S CLOCK TO ITS PLACE IN STET. THE FLAG REGISTER IS READ AND SAVED IN EFRL.

WHEN THE ECS CLOCKS ARE STATUSED (EVERY TWO SECONDS), IF A MACHINE'S CLOCK HAS NOT CHANGED IN TWO STATUSES (FOUR SECONDS), THE MACHINE IS SAID TO BE "DOWN".

IF A MACHINE IS DOWN, THE CIRI INTERLOCK IS OBTAINED (RELEASING THE DOWN MACHINE'S CIRI INTERLOCK IF HELD).

ALL FLAG REGISTER INTERLOCKS HELD BY THE DOWN MACHINE ARE RELEASED. FRET INDICATES WHICH MACHINE SET WHICH INTERLOCK, SO CPUMTR KNOWS WHO HAS WHAT INTERLOCK BY READING FRET FROM ECS.

CPUMTR THEN READS THE GLOBAL MST FOR EACH SHARED DEVICE AND RELEASES ANY INTERLOCK THE DOWN MACHINE HAD BY CLEARING THE INTERLOCK FIELD IN SDGL AND WRITING THE GLOBAL INFORMATION BACK TO ECS.

THE CIRI INTERLOCK IS RELEASED AND THE REMAINDER OF DOWN MACHINE PROCESSING DONE IN PROGRAM MODE.

DOWN MACHINE

NOW IN PROGRAM MODE, CPUMTR "CLEANS UP" THE DOWN MACHINE'S TRT INTERLOCKS.

THE MST/TRT IS INTERLOCKED IN THE USUAL MANNER FOR EACH SHARED DEVICE (VIA SDGL) AND TRT READ FROM ECS.

THE PF UTILITY INTERLOCK IS CLEARED FROM ACGL (IF HELD BY THE DOWN MACHINE). THE GLOBAL CLEARING WILL OCCUR WHEN THE SDGL INTERLOCK IS RELEASED.

THE MRT IS READ FROM ECS IN 100_8 WORD BLOCKS. IF AN MRT BIT IS SET, THEN THE CORRESPONDING TRACK IS EXAMINED IN THE TRT. IF THE TRACK IS INTERLOCKED, THEN THE INTERLOCK BIT IS CLEARED IN THE TRT AND THE TRACK BIT IS CLEARED FROM THE MRT. THE 100_8 MRT BLOCK IS WRITTEN TO ECS AND NEXT BLOCK IS READ.

WHEN THE MRT HAS BEEN PROCESSED, THE TRT IS WRITTEN TO ECS AND THE MST/TRT SDGL INTERLOCK CLEARED.

THIS LOGIC CONTINUES UNTIL ALL SHARED DEVICES HAVE BEEN PROCESSED.

MREC

THE MAIN FUNCTION OF MREC IS TO CLEAR INTERLOCKS HELD BY AN INTERRUPTED MACHINE WHICH HAVE NOT BEEN CLEARED BY CPUMTR DURING DOWN MACHINE PROCESSING. MREC ALSO RECOVERS SPACE ON A SHARED DEVICE THAT IS NOT ACCESSIBLE BECAUSE ONE OF THE OTHER MACHINES IN THE COMPLEX IS INTERRUPTED.

MREC DISPLAYS THE STATUS OF DEVICES THIS MACHINE IS SHARING AND THE MACHINES IT IS SHARING THEM WITH.

THE OPERATOR SPECIFIES THE MACHINE ID (MID) AND DEVICES TO BE RECOVERED. THE FOLLOWING STEPS ARE PERFORMED:

- CLEAR ANY HARDWARE UNIT RESERVATIONS PREVENTING OTHER MACHINES FROM ACCESSING THE DEVICE. OPERATOR ASSISTANCE MAY BE REQUIRED.
- CLEAR DEVICE ACCESSED BIT FOR INTERRUPTED MACHINE IN THE DAT. THIS PROHIBITS A NON-LEVEL 0 RECOVERY.
- CLEARS INTERLOCKS AND USER COUNTS IN SYSTEM SECTORS FOR DOWN MACHINE.
- PROCESSES MRT TO RELEASE ALL LOCAL SPACE HELD BY DOWNED MACHINE.

MREC

MREC MAY NEED TO BE RUN ON ALL MACHINES TO COMPLETELY REMOVE DOWN MACHINE.

ONCE MREC HAS RUN ON A MACHINE, ONLY A LEVEL 0 CAN BE DONE TO RE-INTRODUCE THE MACHINE INTO THE COMPLEX.

QUESTION SET LESSON 27

1. How is the Machine Recovery Table (MRT) used in conjunction with the TRT?

2. Describe the technique used to retrieve the "up-to-date" copy of the TRT for those CPUMTR operations that reference the TRT.

3. What is the difference between the machine ID, machine mask, and machine index?

4. ECS is used to link machines in the MMF complex. What tables are kept in ECS?

5. How does the system determine when a machine is down?

6. What does MREC do? After an MREC has been performed for a down machine, what must be done to re-introduce that machine into the complex?

LESSON 28
MASS STORAGE SUBSYSTEM OVERVIEW

LESSON PREVIEW:

This lesson presents an overview of Mass Storage Facility support in NOS. The objectives of this support and its hardware and software components are described. This lesson is not a detailed discussion of Mass Storage Facility, but is an overview intended to introduce the student to the terminology and concepts involved with NOS usage of this storage media.

OBJECTIVES:

Upon the successful completion of this lesson, the student should be familiar with:

- The objectives of Mass Storage Facility support in NOS.
- The hardware components of the Mass Storage Facility.
- The software components used in NOS or with NOS to support Mass Storage Facility.
- Mass Storage Facility relationship with NOS permanent files.
- Mass Storage Facility use in Multi-mainframe environments.

REFERENCES:

NOS IHB - Chapters 4, 7, 8 NOS Reference Manual, 1-2, 1-8, 2-5 NOS Operator's Guide, Chapter 3 NOS SMRM - Chapters 1, 8, 13

DISCLAIMER:

Some of the reference material may not be available until this support is officially released under NOS.

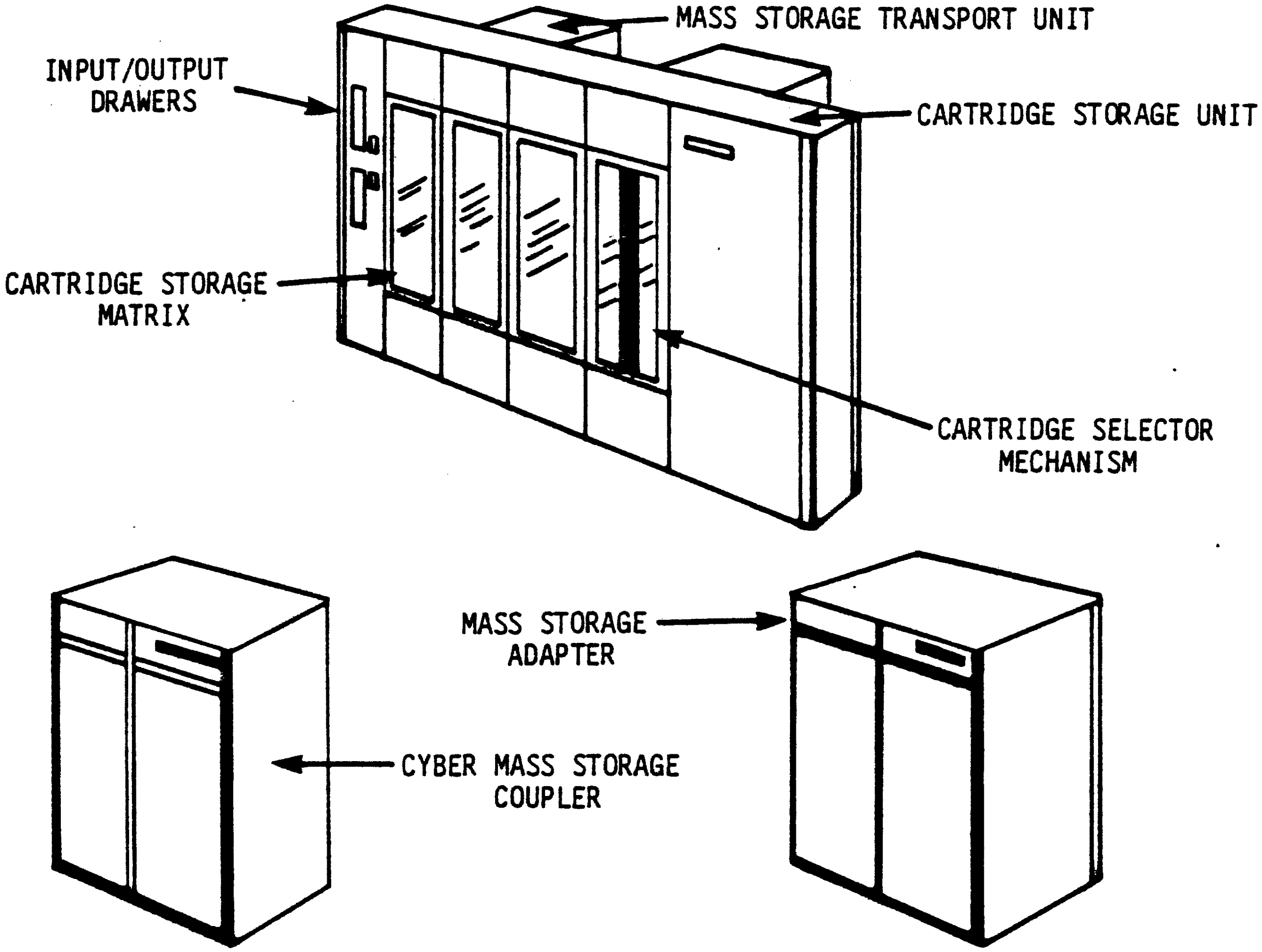
MASS STORAGE SUBSYSTEM OBJECTIVES

- REDUCE TAPE MOUNTS
- REDUCE DISK SPACE FOR INFREQUENTLY USED FILES
- PROVIDE A MULTI-LEVEL FILE SYSTEM
- SUPPORT A LARGE ON-LINE FILE BASE

BASIC MASS
STORAGE COMPONENTS

- CYBER MASS STORAGE COUPLER
- MASS STORAGE ADAPTER
- CARTRIDGE STORAGE UNIT
- MASS STORAGE TRANSPORT
- MASS STORAGE CARTRIDGES

CONTROL DATA CYBER®
MASS STORAGE SYSTEM



CYBER MASS STORAGE COUPLER

INTERFACES 12-BIT CYBER PERIPHERAL
PROCESSOR WITH
8-BIT BYTE ORIENTED MSA

FUNCTIONS

- TWO PP ACCESSES (TWO MORE OPTIONAL)
- 4K X 24-BIT DATA BUFFER
- 12/8 CONVERSION NETWORK
- CONTROL INTERFACE THAT CONNECTS TO MSA

MASS STORAGE ADAPTER

INTERFACES CYBER COUPLER WITH MST/CSU

FUNCTIONS

- CONTROL INTERFACE THAT CONNECTS TO CMSC
- 16-BIT MICROPROCESSOR
- FLEXIBLE DISK DRIVE FOR LOADING CONTROLWARE AND DIAGNOSTICS
- READ/WRITE LOGIC
- DEVICE INTERFACE THAT ACCOMMODATES UP TO 8 DEVICES (CSU OR MST)
- A SECOND DEVICE INTERFACE MAY BE ADDED ALLOWING A MSA TO ACCESS 16 TOTAL DEVICES

MASS STORAGE TRANSPORT

EXCHANGES CARTRIDGES WITH CSU AND
TRANSFERS DATA BETWEEN TAPE AND MSA

FUNCTIONS

- 6250 GCR RECORDING
- TAPE SPEED: 129 IPS
- DATA RATE: 806 KB
- TRACK POSITIONING: 20 MS VOICE COIL
- DUAL PATH TO ALTERNATE MSA
- 5 STATION CARTRIDGE QUEUE

MASS STORAGE CARTRIDGE

CARTRIDGE CONSISTS OF A PLASTIC CASE
CONTAINING 2.7 INCH-WIDE (6.85 CM) MAGNETIC TAPE
WITH USABLE RECORDING LENGTH OF 8.3 FT. (253 CM)

DATA ORGANIZATION

- STREAM IS SMALLEST STORAGE UNIT TO WHICH A SEEK OPERATION CAN BE DIRECTED
- STREAM IS 9 BITS WIDE
- 16 STREAMS PER TAPE (144 TRACKS)
- 614,400 6-BIT CHARACTERS PER STREAM
- OVER 9.8 MILLION 6-BIT CHARACTERS PER CARTRIDGE

MASS STORAGE SUBSYSTEM CAPACITY

	DESCRIPTION	CAPACITY
CARTRIDGE	CHAR/STREAM	614,400
	STREAM/CARTRIDGE	16
	CHAR/CARTRIDGE	9.8 MILLION
CSU	CARTRIDGES/CSU	2,000
	CHAR/CSU	19.6 BILLION
	CHAR/5 CSU'S	98 BILLION
	844-21/CSU	166
	FILES/CSU (RELEASE 1.0)	32,000
	FILES/CSU (FUTURE)	LARGE
9 TRACK TAPE 2400- REEL	CARTRIDGES/TAPE	
	800 BPI	2.3
	1600 BPI	4.6
	6250 BPI	18

MASS STORAGE SOFTWARE APPROACH

- PHASED SOFTWARE DELIVERIES
- OPERATE IN MULTI-COMPUTER ENVIRONMENT
- MAKE TRANSPARENT TO APPLICATION PROGRAMS
- ELIMINATE NEED FOR MANUAL CONTROL OF DATA BASE
- PROVIDE AUTOMATIC MANAGEMENT OF DISK SPACE
- EXTENSION OF CURRENT PERM FILE SYSTEMS
- COMPATIBLE WITH CURRENT PERM FILE SYSTEMS

MASS STORAGE
SUBSYSTEM - RELEASE 1

- STAGED ACCESS
- BASIC MULTI-LEVEL FILE SYSTEM
- MULTI-COMPUTER ACCESS
- CARTRIDGE MAINTENANCE
- DISK SPACE MANAGEMENT
- BASIC MSS UTILITIES
- TAPE BACKUP
- ON-LINE DIAGNOSTICS

MASS STORAGE SUBSYSTEM
COMPONENTS

OPERATING SYSTEM ADDITIONS

- PFM MODIFICATIONS
- 16 WORD PF CATALOG
- SYSTEM TABLE CHANGES
- DSD MODIFICATIONS
- PF UTILITIES MODIFICATIONS

EXECUTIVE CONTROL

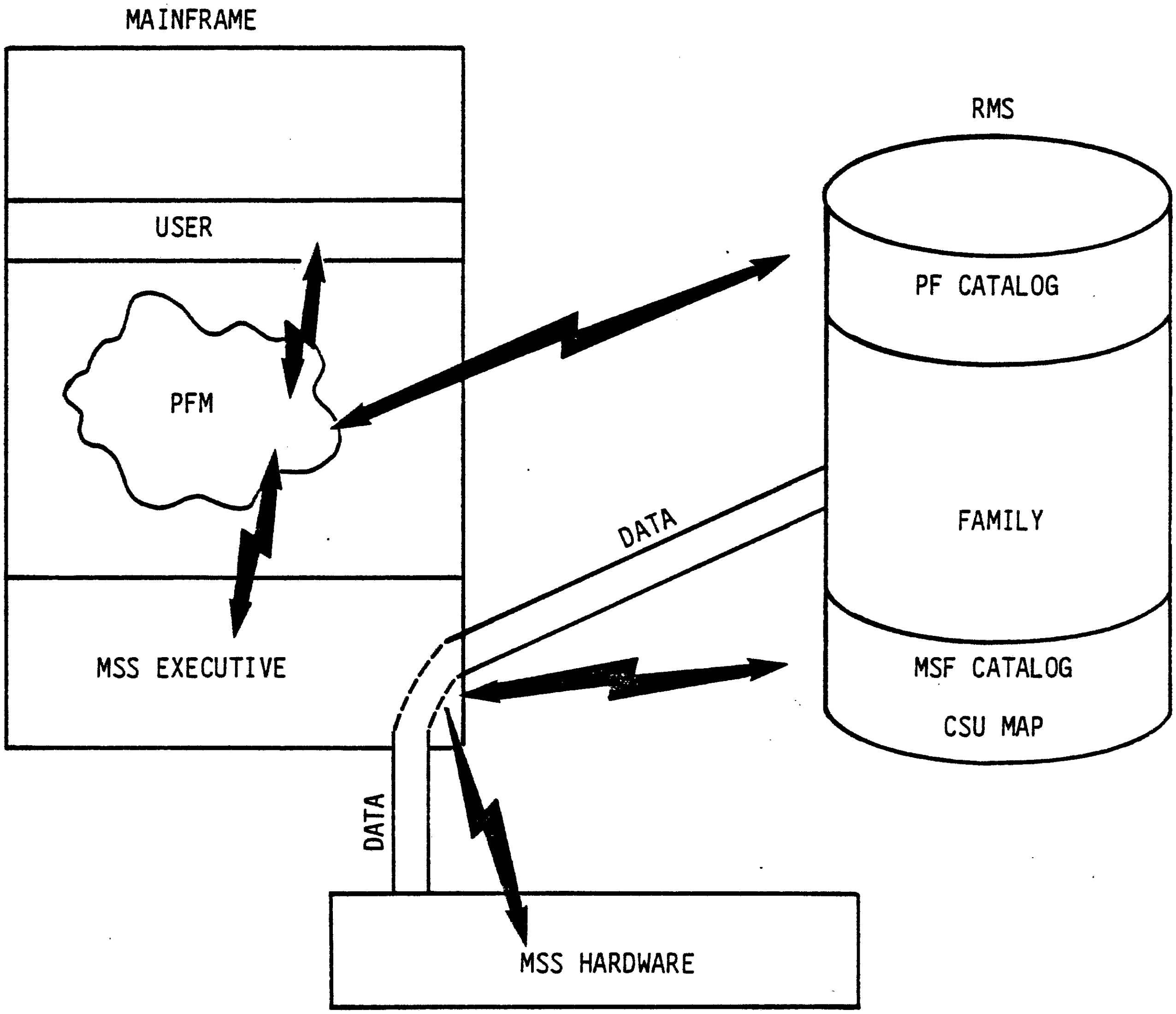
- MSSEXEC (MASTER) - MASTER MSF CONTROL
- MSSEXEC (SLAVE) - SLAVE MSF CONTROL
- PPU DRIVER

MSSEXEC (MASTER)

MSSEXEC IS A CM RESIDENT PROGRAM THAT WORKS WITH THE MSS PPU DRIVER TO PERFORM ALL MSS CARTRIDGE MOVEMENT AND DATA TRANSFERS IN SUPPORT OF THE MSS SOFTWARE SUBSYSTEM

- MOVES DATA TO/FROM MSF
- MAINTAINS MSF CATALOG AND CSU MAP
- MANAGES MSF HARDWARE
- ALLOCATE MSF SPACE AND CREATE NEW MSF FILES IN RESPONSE TO REQUESTS FROM MOVE UTILITY
- STAGE FILES TO DISK IN SUPPORT OF ATTACH REQUESTS FROM MASTER OR SLAVE MAINFRAMES
- PROVIDE ACCOUNTING AND STATISTICAL INFORMATION
- SUPPORT ON-LINE DIAGNOSTICS
- MAINTAIN CARTRIDGE LOAD AND TAPE PASS STATISTICS
- DYNAMICALLY DETECT INCONSISTENCIES BETWEEN MSS CATALOG AND CARTRIDGE CONTENTS.

MSS SYSTEM FLOW



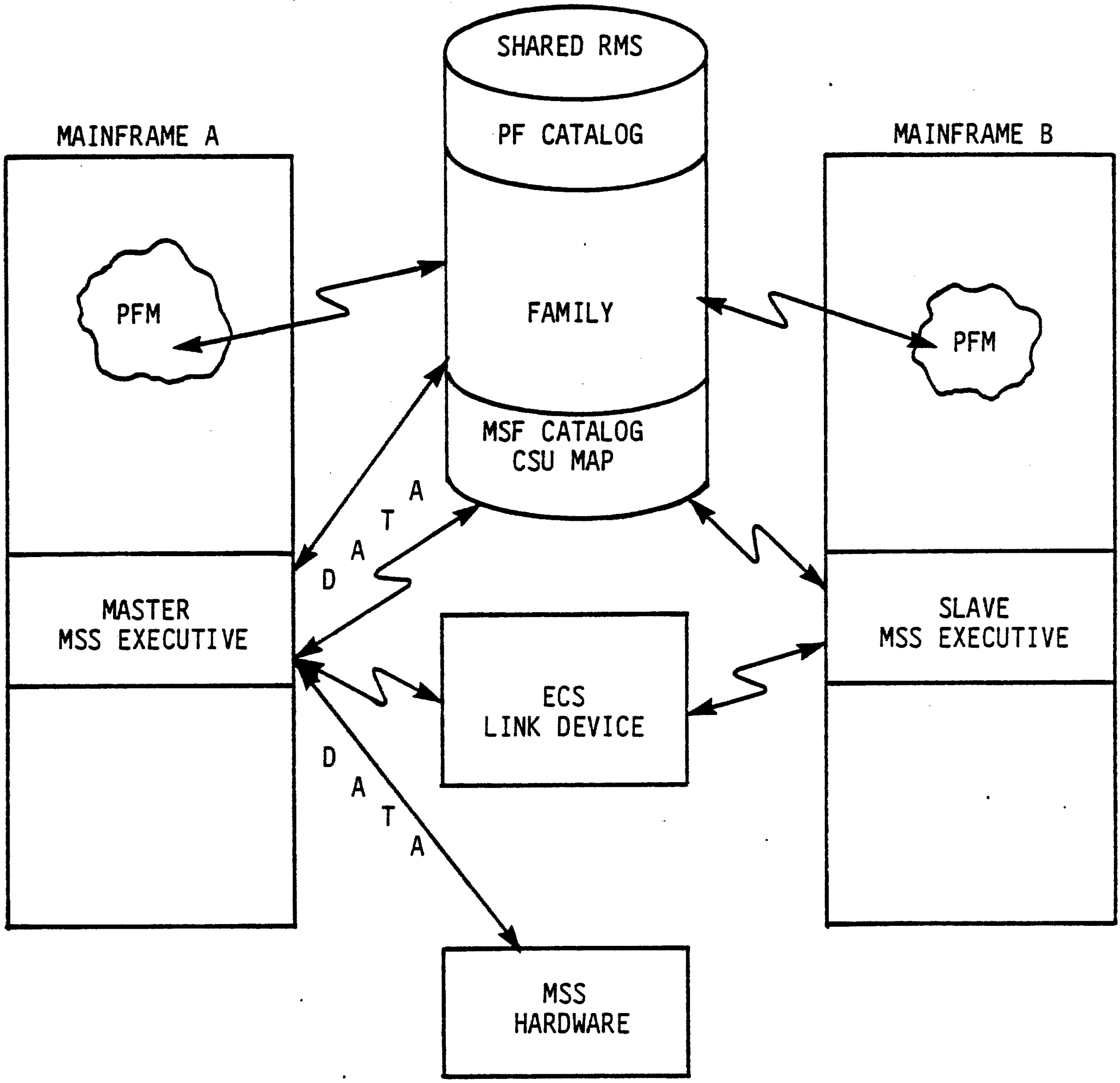
- USER ISSUES ATTACH
- PFM DETECTS FILE RESIDES ON MSF
- PFM INFORMS MSS EXECUTIVE
- MSS EXECUTIVE SENDS REQUEST TO MSF
- MSS EXECUTIVE COPIES FILE FROM MSF TO RMS
- MSS EXECUTIVE RECATALOGS FILE
- USER ATTACH PROCESSING CONTINUES

MSSEXEC (SLAVE)

CM RESIDENT PROGRAM THAT WORKS WITH MASTER MSSEXEC TO PROCESS ATTACH REQUEST FROM JOBS RUNNING IN SLAVE MAINFRAME

- ENTERS STAGING REQUESTS IN COMMUNICATION REQUEST FILE
- PROCESSES RESPONSES FROM MASTER MSSEXEC
- HAS JOB WAITING FOR FILE RESUMED WHEN FILE STAGED
- PROVIDES ACCOUNTING AND STATISTICAL INFORMATION FOR SLAVE MAINFRAME

MSS MULTIMAINFRAME



- PERMANENT, FILE SET MUST BE SHARED
- MASTER AND SLAVE COMMUNICATE VIA LINK DEVICE
- MASTER FUNCTIONS MSS HARDWARE

ASMOVE UTILITY

THE MOVE UTILITY CAUSES PERMANENT FILES TO RESIDE ON DISK AND/OR MSF IN SUPPORT OF THE SITE'S OBJECTIVES FOR DISK SPACE MANAGEMENT AND BACKUP

- DECIDES ON RESIDENCE OF FILES PER A DISK SPACE MANAGEMENT ALGORITHM THAT USES:

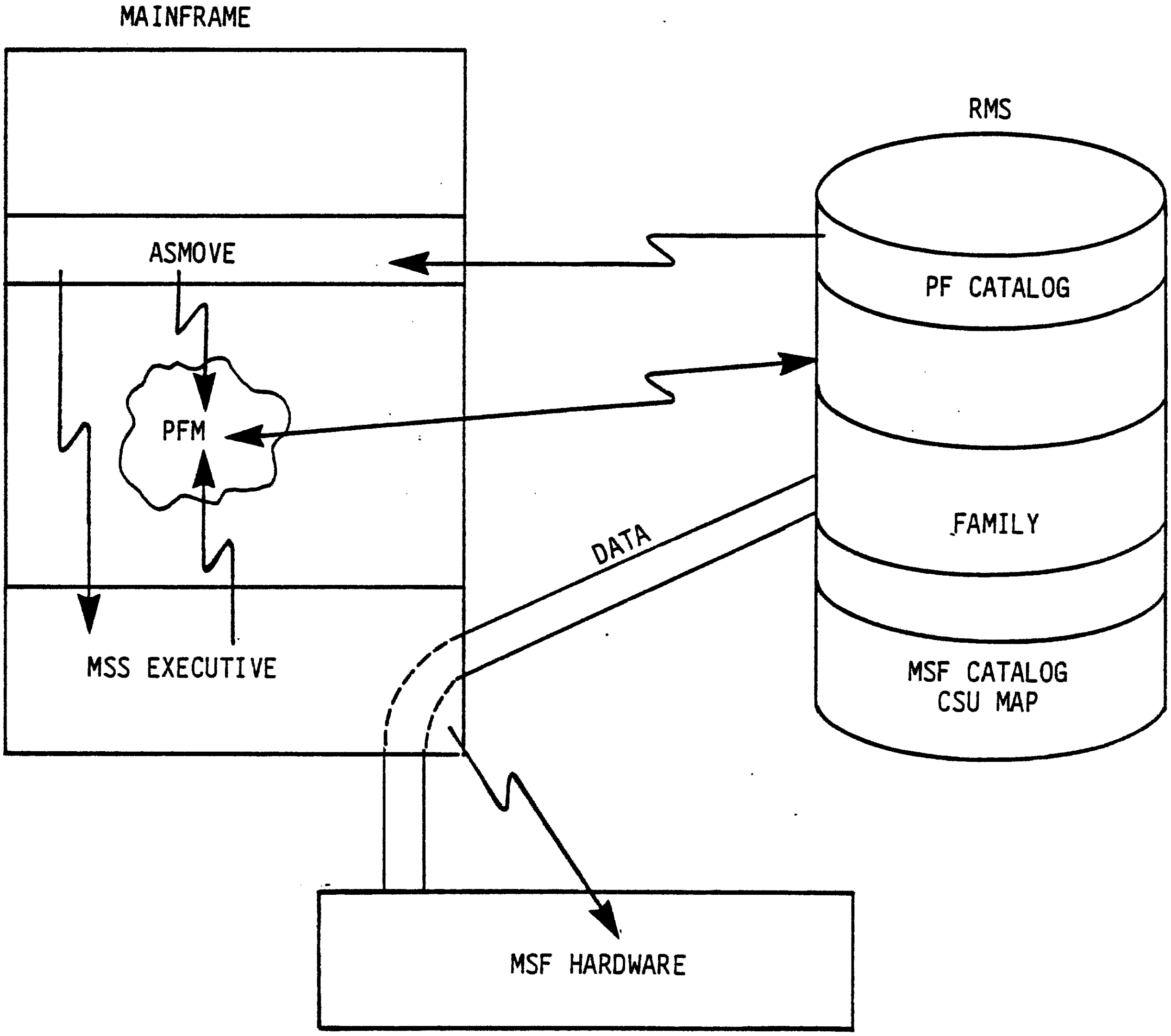
  RUN TIME PARAMETERS
  INSTALLATION PARAMETERS
  NOS TABLE INFORMATION, PFC, ETC.

- CAUSES MSSEXEC TO PERFORM FILE DESTAGING OPERATIONS AND OPTIONALLY RELEASE DISK SPACE

- GENERATES MSS ACTIVITY REPORTS

- INVOKED MANUALLY BY SITE PERSONNEL

ASMOVE UTILITY

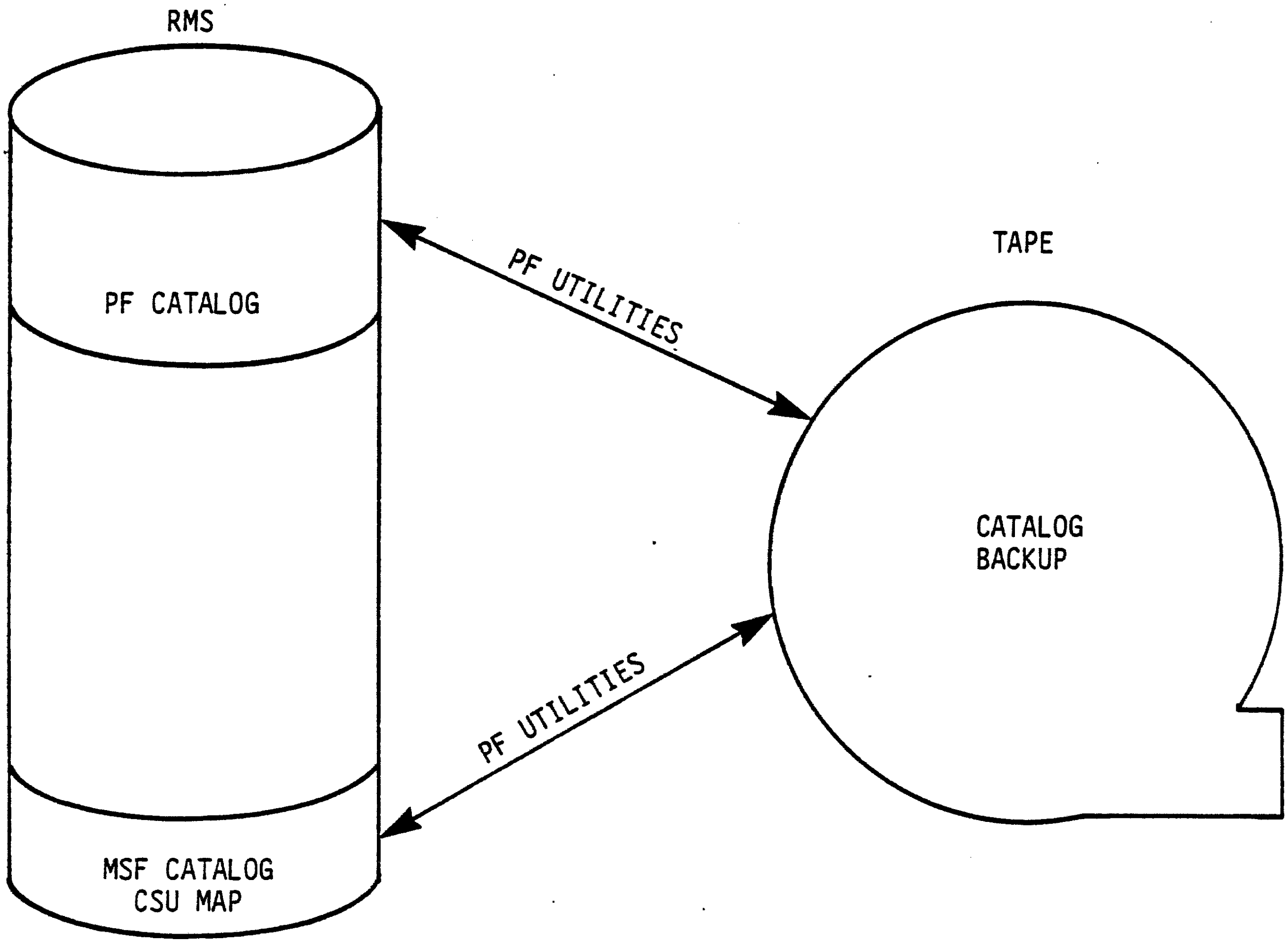


- SCANS PFC FOR MSF FILES
- SELECTS FILES TO BE MOVED TO MSF
- SELECTS FILES TO RELEASE DISK SPACE
- REQUESTS MSSEXEC TO MOVE FILES
- CALLED BY CONSOLE OPERATOR
- MSSEXEC WILL COPY FILE TO MSF AND RECATALOG FILE IN PFC

MASS STORAGE
SUBSYSTEM UTILITIES

- ASLABEL - CARTRIDGE MANAGEMENT
- ASUSE - CARTRIDGE UTILIZATION
- ASDEF - DEFINE MSF CATALOGS AND CSU MAPS
- ASVAL - INSURE MSF CATALOG INTEGRITY
  - RELEASE SPACE FOR PURGED MSF FILES
- ASDEBUG - DUMPS CARTRIDGE DATA IN RAW FORM
  - RESOLVES MSF CATALOG INCONSISTENCIES

TAPE BACKUP



- PF CATALOG BACKED UP BY PF UTILITIES
- MSF CATALOG AND CSU MAP BACKED UP BY PF UTILITIES

MASS STORAGE SUBSYSTEM
RELIABILITY FEATURES

- GCR RECORDING, CRC CODES
- SINGLE BIT ERRORS, MOST TWO BIT ERRORS CORRECTED
- REDUNDANT CONFIGURATION ALTERNATIVES
- AUTOMATIC WRITE ERROR RECOVERY
- FLAGGING OF UNUSABLE STREAMS
- RETRY ON READ ERRORS
- ERRORS RECORDED
- USAGE STATISTICS MAINTAINED
- ON-LINE DIAGNOSTICS
- TAPE BACKUP
- EXTENSIVE LABELING CONVENTIONS
- RECOVERY AND DEBUG UTILITIES

MASS STORAGE
SUBSYSTEM - FUTURE

- ENHANCED DISK SPACE MANAGEMENT
- DIRECT ACCESS
- DATA COMPACTION
- PRIVATE CARTRIDGES
- CARTRIDGE REMOVABILITY
- PERFORMANCE ANALYSIS
- MULTI-FILES PER STREAM
- STAGE/DESTAGE OF INDIRECT ACCESS FILES
- LOAD AND DUMP PERMANENT FILES TO MSS

QUESTION SET LESSON 28

1. Describe the hardware components of the Mass Storage Facility.

2. What is the difference between master and slave versions of MSS Executive?

3. What utilities are provided with the Mass Storage Subsystem and what are their functions?

SECTION 29

ANSWER SETS

ANSWER SET LESSON 2

1. The system library is the collection of all routines contained on the deadstart tape or introduced as part of the system online via SYSEDIT. SYSEDIT builds directories for system routines as part of Central Memory Resident.

   A program library is a collection of source code routines (source code and common decks) maintained by the utility programs MODIFY and UPDATE.

   A user library is a collection of relocatable routines containing a directory for use by the CYBER Loader. A user library can be built only by using the LIBGEN utility. Typically, the routines in a user library are associated with use of a given product, such as the COBOL user library.

   Any file with a directory can be considered to be a "library". The records within a library file can be manipulated by the LIBEDIT utility and the contents of a library file (or any file for that matter) can be listed using the CATALOG and ITEMIZE utilities.

ANSWER SET LESSON 3

The answers in this question set refer to the study dump found in this Handout. If the study dump is not used, the instructor will be supplying you with a set of correct answers for the dump you are using.

1. a. 24B = 20
   b. 24B = 20
   c. 7777B = 262K
   d. yes
   e. 76000B; found in word CMRL

2. PP3: CIO
   PP4: none

3.

PP0	MTR	46
PP1	DSD	53
PP2	CPM	33
PP3	CIO	30
PP7	STM	46
PP11	QAP	33
PP22	QAP	61
PP24	CIO	61
PP26	LDQ	33
PP30	1MA	44

4.

```
EQ0=RD.
EQ75=TT.
EQ76=TE.
EQ77=NE.
```

The above entries are automatic. The following is the CMRDECK used for the system on which the study dump was taken.

```
CMRDC35
NAME= (35) CYBER 174 S/N 620 CLSH.
VERSION=NOS 1-8J03T/R2B.
MID=62.
NCP=24.
IPD=3.
FNT=1756.
CM=7777.
EQ01=DQ-1,ON,0,42,5,7.  SYSTEM, TEMP
EQ02=DQ-1,ON,0,43,5,7.  SYSTEM,TEMP
EQ04=DI-1,ON,0,6,21,20.  TEMP
EQ05=DI-1,ON,0,16,20,21.  TEMP
EQ06=DK-2,ON,4,26,20,21.  PF/40
EQ07=DI-N3,ON,0,24,34,44,22,24.
EQ10=DS,ON,7,0,10.
EQ11=DJ-1,ON,0,2,23,24.  RMVE
EQ12=DQ-1,ON,0,40,7,5.  PF/42
EQ13=DQ-1,ON,0,41,5,7.  PF/43
```

```
EQ14=DJ-N2,ON,0,3,4,21,20.  PF/44
EQ15=DI-1,ON,0,5,23,24.
EQ16=DI-1,ON,0,6,23,24.
EQ17=DI-1,ON,0,4,23,24.
EQ20=DI-1,ON,0,54,24,22.  RMVE
EQ21=DI-1,ON,0,64,22,24.  RMVE
EQ22=DJ-1,ON,0,0,22,23.  FMD DRIVE FOR TESTING
EQ23=DI-1,ON,0,35,22,23.  RMVE
EQ24=DJ-1,ON,0,1,24,22.  RMVE
EQ27=DJ-1,ON,0,3,22,23.  RMVE
EQ30=DK-1,ON,0,46,20,21.  RMVE, FT.
EQ32=DI-1,ON,0,7,23,24.  RMVE
EQ33=DJ-1,ON,0,5,20,21.  RMVE
EQ34=DJ-1,ON,0,0,7,5.  RMVE
EQ35=DJ-1,ON,0,1,5,7.  RMVE
EQ36=DI-1,ON,0,14,22,24.
EQ37=DI-1,ON,0,15,22,23.
EQ40=TT,OFF,7,2,0,,30.  TRANEX STIMULATOR
EQ41=TT,OFF,0,1,0,0,100.  STIMULATOR
EQ42=TT,ON,2,,2,,100.  2550-100 (6676 EMULATOR)
EQ43=ST,OFF,7,,4.  6671
EQ44=NP,ON,7,1,1,,2.  NPU NODE3
EQ45=NP,OFF,7,1,6,,2.  LOCAL/REMOTE NPU NODE7
EQ46=SE,ON,7,,4.  STEM 6671 S/N 183.
EQ47=CR,ON,4,,12.
EQ50=MT-10,ON,0,0,13,32,,,20.  66X
EQ60=MT-10,ON,4,0,31,33,,,10.  67X
EQ70=CP,ON,7,,12.
EQ71=LP,ON,3,,12.  512
EQ72=LP,ON,2,,12.
EQ73=LT-6,ON,5,,12.  580 UPPER/LOWER CASE
EQ74=LS-1P,ON,6,,12.  580-20 PFC
PF=6,F,140,140,NOSCLSH,40.
PF=7,F,010,010,NOSCLSH,41.
PF=12,F,003,003,NOSCLSH,42.
PF=13,F,200,200,NOSCLSH,43.
PF=14,F,024,024,NOSCLSH,44.
TEMP=1,2,4,5.
REMOVE=11,15,16,17,20,21,22,23,24,27,30,32,33,34,35,36,37.
FAMILY=13.
SYSTEM=1,2.
NAMIAF=102.
```

5. Tape channels are 13, 31, 32, 33
   Unit record channel is 12

6. EQ = 1 and 2, track = 4004, FNT address = 6020

7. 6070 file = PROFILB, type = FAFT, control point = 0
       EQ = 13, FT = 5674, FAFT files do not have CT/CS

   6110 file = OUTPUT, type = PRFT, control point = 2
       EQ = 2, FT = 4133, CT = 4133, CS = 2

6236 file = CS00ABR, type = ROFT, control point = 0
EQ = 1, FT = 4206, ROFT files do not have CT/CS
note that the job has a subsystem connection

6410 file = EPLVER, type = PMFT, control point = 14
EQ = 13, FT = 5647, CT = 5647, CS = 1
note that the file is in read only ("write lockout" bit set to 1)

6414 file = AMBY010, type = PRFT, control point = 23
EQ = 5, FT = 5160, CT = 5201, CS = 63
note that the system sector bit is set

8. 303300B from word ACML

9. Normally found in byte 4 of MSCL (word 24); however, this word has been destroyed which is why this system dump was necessary.

10. 16.23.30. from word TIML

11. Idle program, address in ACPL(0) is 45776B (part of CPUMTR) control point 7, address in ACPL(1) is 1600B

12. 2400 from job control block at word 35700

13. 30 from job control block at word 35713

14. OP = 4004, LP = 3740, UP = 7000 from job control block at 35721

15. location 46477: 0311 1700 1073 0000 0340

16. location 47037: 2CA, length 62B

17. PLD at 72015: 0102 2001 4114 0601 1073
RCL - none; would begin at location 71710

CLD at 72464: 0203 2300 0000 0000 0000

18. System dayfile: 40350
Dayfile Pointers: 5640

Dayfile Dump Buffer: 41450

19. CH 5 = PP 3   CH 10 = PP 1

20. None; all channels are available.

21. The control point area for control point 12 is found at address 2400B.

    a. ALMY024   TXOT
    b. I
    c. One PP assigned: PP26 LDQ function 33
    d. RA = 432600   FL = 55700
    e. PR = 30, QP = 7000

f. 2452211750000B quarter nanoseconds
g. user: AD2000 index: 2663B family default (EQ = 13B)
h. 241631B units
i. AD20GX,9,C.
j. EOR

22. None. When "turned off", the upper 18 bits of the PP input register would be "***".

23. FNT pointers are limited to 12 bits in word FNTP. Therefore, the maximum is 7777B = 4095.

ANSWER SET LESSON 4

1. A CPU program communicates with monitor by making RA+1 requests. Pool PPs communicate with monitor by making requests through the PP's output register in the PP's communication area. MTR makes requests to CPUMTR through a central exchange with the request in register X0 of MTR's exchange package.

2. An idle exchange package for each CPU; an exchange package for each Pool PP (PP2 through PP N); an exchange package for MTR; an exchange package for each control point area.

   A CYBER 176 has an additional exchange package for error exits located in CMR, following the FNT.

3. The Pool PP sets up its exchange package in CPUMTR's memory with (P) = PPR and (B0) nonzero; it writes the monitor request into the output register in the communication area; it does a central exchange (MXN); it reads the first word of its exchange package; if (B0) is nonzero, then the exchange is retried as the exchange did not take place; if (B0) = 0, the exchange took place and the Pool PP waits for the function to be completed by waiting for byte 0 of its output register to become zero.

4. MTR sets up its exchange package in CPUMTR's memory with (P) = PMN, (B0) nonzero and (X0) = request; it does a central exchange (MXN); it reads the first word of its exchange package; if (B0) is nonzero, the request is retried; if (B0) = 0, processing continues. MTR does not wait for CPUMTR to complete its request before continuing.

5. The program writes its system request into location 1 of its field length (RA+1) and executes an XJ instruction.

6. The "old" active exchange package is moved to its address from ACPL. "New" exchange package information is set into ACPL and B2. If necessary, CPUMTR rebuilds the PP's exchange package. CPUMTR exchanges to the address in B2, thus starting the new program.

7. Program mode is interruptable by another exchange; monitor mode is not interruptible. Program mode is used for time consuming tasks.

8. In monitor mode, an XJ exchanges with the address specified in the instruction, namely, Bj+K. In program mode, an XJ exchanges with the address contained in the executing exchange package's MA (monitor address) register.

ANSWER SET LESSON 5

1. W = waiting for the CPU
   X = waiting in periodic recall
   I = auto recall
   A = active in CPU 0
   B = active in CPU 1
   " " = no CPU status but other job activity

2. Periodic recall means that the CPU program has relinquished the CPU for a installation specified period of time. Automatic recall means that the CPU program has relinquished the CPU until a pending RA+1 request completes or the program is restarted by the PP routine that satisfied the RA+1 request. Auto recall means that the CPU program has relinquished the CPU until a previous RA+1 request completes, that is, the completion bit (bit 0) in a status words becomes set to 1.

ANSWER SET LESSON 6

1. CPUMTR; some requests are processed directly by CPUMTR, other requests are satisfied by CPUMTR assigning a PP to accomplish the request.

2. CPUMTR will execute subroutine APJ (Assign PP Job) if it does not process the RA+1 request internally. If the job has QP , MXPS, APJ will force the request to be made with auto recall and will not assign a PP if there is other PP or tape activity present. (CIO requests skip these checks.) If not at PP saturation, APJ calls APS (Assign PP and Search Library). APS will search the PLD by executing subroutine SPL (Search Peripheral Library). SPL returns the residency address (load parameters) for the desired routine or for SFP (if the routine was not found). APS retrieves the next PP from the NP stack, sets the routine name in the message buffer, clears the upper six bits of the request as a flag indicating the directory has already been searched, writes the request into the PP's input register, and then updates the NP stack linkage and the PP count in STSW. APS returns to APJ which will end the program's CPU usage (ECP) if an auto recall request and indicate the PP involved by setting the PP's output register address into RA+1.

ANSWER SET LESSON 7

1. The delays balance system activity by causing certain system activities, such as job switching and PP recall, to occur periodically.

2. MTR issues an RCLM monitor request for the waiting control point if the recall delay time has expired.

3. Channels are interlocked by MTR in the channel reservation table. All routines reserving channels must use the RCHAN (reserve channel) or DCHAN (release channel) macros or the RCHM (reserve channel) or DCHM (drop channel) functions to guarantee that channel reservations are properly maintained.

ANSWER SET LESSON 8

1. TUFL Table.

CP	RA/100	FL/100	TUFL	RA/100	FL/100
1	300	20	0	300	20
2	320	320	0	320	320
3	640	0	10	640	0
4	650	0	30	650	0
5	700	17	61	700	17
6	1000	0	10	1000	0
7	1010	50	50	1010	355
10	1130	0	50	1413	0
11	1200	100	0	1413	100
12	1300	10	0	1513	10
13	1310	50	20	1523	50
14	1400	0	10	1573	0
15	1410	5	163	1573	5
16	1600	200	-0-	1600	200

2. 46400 from word ACML in low core CMR

3. The sum of TUFL values from control point 7 to N give 50+50+20+10+163 = 333 which is greater than 305 (amount of increase/100)

   MM = 15, MM+1 = 163, MM+2 = 7, MM+3 = 7, MM+4 = 5

4. See table

5. 2000 in word 1 of CMR (MFLL)

6. A storage move will occur so that this control point's field length is at the end of memory.

7. NOS manages field length in multiples of 100B and stores RA and FL in 12 bit bytes. A 262K CM configuration is 1000000B words; thus with the largest address for NOS being 777700B (stored as 7777B), the last 100B words cannot be accessed.

ANSWER SET LESSON 9

1. At PPFW. It must load at PPFW-5 for the 5 byte header, and if using mass storage drivers, should not use memory beyond BFMS (6776), since that area is the driver's default buffer area.

2. To insure proper operation of the system, it is necessary to move control point field lengths in central memory and ECS. In most cases, a control point cannot be moved while it has PP activity. Therefore, a PP should PAUSE occasionally for storage movement to occur.

3. Pool PPs loop reading their input registers (IR); if the input register becomes non-zero, it is because CPUMTR chose that PP to execute a particular PP routine.

4. A Pool PP program makes monitor requests by using the MONITOR macro, which jumps to PPR subroutine FTN. FTN writes the request into the PP's output register (OR), sets up the PP's exchange package in CPUMTR with (P) = PPR and (B0) nonzero. After doing a MXN, FTN reads the first word of the PP's exchange package and if (B0) = 0, the exchange was successful. FTN then loops reading the output register. Byte 0 of the output register will be set to zero to indicate that the request has completed. Monitor requests processed by MTR are handled in a similar manner, except that the MXN and B0 checks are not done.

5. The control point assignment is found as the lower 5 bits of byte 1 of the input register. PPR will storage the control point address for this control point number in direct cell CP (74).

6. PP routines use PPR subroutine DFM to write messages into all dayfiles, whether they be the system, account, control point or error log dayfiles.

7. The updated RA and FL will be found in direct cells RA and FL, which are set by the PP resident subroutine PRL, which gets these values from bytes 3 and 4 of control point word STSW. After a pause, if the RA or FL has changed, any absolute addresses stored in the program must be re-absolutized. This includes all places where instruction modification was done to use absolute addresses.

8. Overlays may be called into a PP by using the EXECUTE macro, which jumps to PP resident subroutine EXR. EXR in turn jumps to subroutine PLL to load the routine into the PP, then EXR enters the overlay for execution.

9. These locations are used by the mass storage drivers for error processing and as a default buffer.

10. A "PP HUNG" indicates that a PP has issued a monitor function for which CPUMTR finds something wrong during its processing. A "HUNG PP" indicates that a PP has issued a monitor function for which MTR finds something wrong during its processing. Both messages are displayed from the system control point's MS2W message area. The packed time and date (PDTL) is entered into the last word of the "hung" PP's communication area. The system will continue to operate as much as possible, so more than one PP could become "hung" at a time.

11. Left to the student.

ANSWER SET LESSON 10

1. SETMS (Set mass storage driver parameters) performs the following operations: passes read/write operation to the driver insures correct driver is loaded presets the driver initializes DRSW, RDCT, STSA, WDSE sets write error buffer address selects error processing options SETMS is called:

   before any initial read/write
   switching between read and write
   switching logical tracks
   changing error processing options

   ENDMS (End mass storage driver access) performs the following operations:

   releases channel
   releases controller
   releases drive
   releleaes other I/O resources

2. A PP routine requests mass storage space by using the monitor function RTCM (Request Track Chain) for the desired number of sectors on the desired equipment (either the type of equipment or equipment number). CPUMTR will allocate a track chain for at least that many sectors in multiples of tracks.

3. Read sector = RDS; Write sector = WDS.

4. The drop track (DTKM) monitor function is issued after writing a disk file so that any unused tracks may be returned to the pool of allocatable tracks. The DTKM function also enters the sector number of the EOI (end-of-information) in the track linkage byte of the track in which it occurs.

5. The first track is 5702 which links to 5746, 6016, 6121, 6122, 6123, 6124, 5365, 6135 where the EOI is in sector 76. Since this is a DI-3, there are 501B sectors per track; therefore, the file length is 5106B sectors long (10B x 501B + 76B).

6. The flaws are:

EQ=6	111	141
EQ=14	4	166

7. The mass storage driver performs a seek operation to inform the disk controller of an address to position to in preparation for the next data transfer. Once the seek is initiated by the controller, the disk drive can complete the positioning without further direction from the controller. This allows the controller to perform read and write operations or to initiate positioning on other drives that may be accessed through the controller. This overlapping of head positioning with reading, writing, or initiation of head movement is called seek overlap. Seek overlap is managed through the driver seek wait (DSWM) monitor function. There are two options for seek overlap: waiting for position and waiting for unit. Waiting for position

typically occurs if the drive is currently positioning or the position sought is not in the same cylinder as the current position. Waiting for unit typically occurs when another PP is in a waiting for position condition with the unit reserved.

8. When writing in 1-to-1 interlace (full tracking), the error status for a write is not always immediately available. In order to correctly process write errors, some special considerations are needed. If the "last sector write" function is used, the status is immediately available and error processing continues normally. To recover from certain errors when not doing a "last sector write" and the error occurs in the previous sector, the processing done depends on the error processing options selected and the presence of a recovery buffer specified in the SETMS call.

   If a recovery buffer is specified and write error processing is not selected, the previous data is read into the recovery buffer and recovery is attempted. If the recovery is successful, the current sector is retried.

   If a recovery buffer is specified and write error recovery is also specified, the previous data is read into the recovery buffer and recovery is attempted. If the retry is unsuccessful the (A) = -0 is returned. If the retry is successful, writing continues with the current sector.

   If a recovery buffer is not specified and write error processing is not selected, the operation is aborted as an unrecoverable error.

   If a recovery buffer is not specified and write error recovery is specified, the previous data is read over the current sector and recovery is attempted. If recovery is successful, the (A) = -1 is returned to indicate that the current data must be regenerated. A status of (A) = -0 indicates the previous sector recovery was not successful.

9. 706B sectors per track
   physical units 03 and 04 (from MST word DDLL)

   DJ-2

ANSWER SET LESSON 11

1. The A register is set to 10000B so that the PP can input its entire field length before being released from the IAM instruction (disconnected).

2. At deadstart time, each PP is hung on a input on its channel to read into the PP beginning at word 0. That is,

   ```
   IAM    ppch,0
   ```

3. Routine SET and subsequent routines expect to find routines in a certain order.

4. The deadstart panel is read into PPO beginning at word 1 through word 17 (CYBER 17X, word 14 for 6000/CYBER 70) and are used to load the initial program. Panel words 5 through 20 are passed to SET by CTI.

5. NOS uses the HDT to indicate to the system software what hardware features are available for use, such as, central memory size, number and status of PPs and PPs, number of CPUs, CMU, CEJ/MEJ, instruction stack, and so on. Various system tables will be defined and system code loaded and/or executed based on the hardware properties.

6. At deadstart time, SYSEDIT builds the PPLIB, RPL, RCL, PLD, and CLD. It builds these tables in the same manner when it is run "online".

7. PPLIB contains all the PP routines that are not in the RPL. The PP routines have their prefix (77) table stripped from them leaving only the single word PP header. These routines are still complete with prefix tables on the SYSTEM file. The PLD points to the address of the routine in the PPLIB, rather than the address of the routine on the SYSTEM file.

8. The system checkpoint builds the skeleton of the system table or checkpoint file on the first system device. This checkpoint file is used for subsequent deadstart recoveries.

9. A level 0 recovery is used to initialize the system (initial deadstart) and to recover from unsuccessful level 3 deadstarts. Jobs are recovered by the QREC utility if QPROTECT had been enabled when the jobs were introduced to the system, and permanent files and dayfiles are recovered from the mass storage devices on which they reside.

   A level 1 recovery is used to resume system operations after a controlled idle down and a CHECKPOINT SYSTEM. The operating system and active jobs and files are recovered from the checkpoint file.

   A level 2 recovery is used to resume system operations after a controlled idle down and a CHECKPOINT SYSTEM (as a level 1) with the operating system being re-established from the deadstart file. Active jobs and files are recovered from the checkpoint file.

A level 3 recovery is used to resume operations from a system hang or equipment malfunction interruption. Active jobs and files and the operating system are recovered from central memory tables (FNT, libraries and directories).

10. On a level 0 deadstart, MS VALIDATION and PF VALIDATION have no special meaning. Dayfiles, permanent files in write mode, and other preserved files are verified with respect to their EOI.

    On a level 3 deadstart, MS VALIDATION and PF VALIDATION play a major role. Both must be specified, and if so, EOI verification is done for all preserved track chains, and circular linkage is checked for all recoverable files (entries in FNTs).

11. Deadstarting with device checkpoints pending may destroy the integrity of permanent files, since track linkages may be lost during recovery of the devices.

12. The Deadstart Sequencing priority calling of CMS causes CMS to issue certain control statements which must run to completion before any job processing may be done. The scheduler will not schedule any jobs if a job with the Deadstart Sequencing priority is at a control point.

ANSWER SET LESSON 12

1. For a write request, an FNT/FST entry will be created and the write performed. For a read request, an FNT/FST entry will be created and a EOI status returned.

2. CIO performs all mass storage input/output and position operations using its main routine and appropriate CIO overlays.

3. CIO extracts the function code from the FET, loads an appropriate overlay, and performs the requested read/write/ position operation until the operation has completed, the FET buffer becomes full or empty, or a given number of sectors are transferred.

4. Random input/output is accomplished by the user specifying logical addresses of records from which CIO computes a corresponding logical disk address for the particular record.

   For a randon read, the user sets the contents of FET+6 to the logical address of the record desired and issues a random read request with the random bit set in FET+1. CIO will convert the specified address into a logical disk address, position the disk to that address, and begins transferring data from that disk address to the FET buffer.

5. "Rewrite-in-place" is the random output technique of writing a new logical record over an existing logical record. The "rewrite" CIO functions do not alter the end-of-information of the file (unless the file is lengthened) but may alter the file data structure with record or file marks.

6. CIO terminates an operation when the buffer is full/empty, FL/ 100 sectors have been transferred, or when the desired file structure mark (EOR, EOF, EOI, BOI) is reached (for reads only).

7. CIO sends a three word parameter block to a specific unit descriptor table (UDT) address within MAGNET's field length by using the TDAM monitor function. CIO forces the request into recall and MAGNET will eventually complete the request.

8. CIO stores the FET address in the control point area word TIOW (output) or TINW (input), forces the request into recall, and issues the rollout (ROCM) monitor function to roll the job out. 1RO then handles the data transfer with the time-sharing subsystem.

ANSWER SET LESSON 13

1. 1DS is used when DSD must perform a request on a given control point or file, do mass storage activity, or any time consuming task. The text of the command is passed to 1DS in the DSD/1DS communication buffer in CMR.

2. Resident syntax table
   Master display routines
   Keyboard processor
   Resident subroutines
   Syntax and Function Overlay area
   Left Screen program
   Right Screen program

3. Interpretative syntax works as follows. As the operator enters a command, if DSD can determine the remaining characters of the command uniquely from the characters entered so far, it will fill them in or even complete the command phrase and display them for the operator automagitally.

4. They sense an "*" from the keyboard. If "*" is sensed as the first character, the routine releases the channel. When the routine (either DSD or DIS) is not connected to the display channel, it periodically checks the channel status table to determine if the display channel is available. If available, the routine will request the channel via the RCHM monitor function.

5. DSD does not have a PP resident. It also cannot risk blanking the screens by attempting to load one of its overlays from disk. Therefore, to load overlays, DSD calls PP routine 1DL via a RPPM. 1DL selects the desired routine and sends it to DSD via the display channel. CM resident DSD overlays are read directly by DSD.

6. Console displays are formatted and painted on the screen one line at a time. The display overlay issues the display coordinates for each piece of information to be displayed as it processes the information.

ANSWER SET LESSON 14

1. The scheduler should be called whenever there is a change in system resources. The scheduler is requested for this case by the RSJM monitor function. However, the scheduler is also called on a periodic basis by the system monitors. MTR requests CPUMTR to call the scheduler and CPUMTR will assign a PP to 1SJ or 1SP accordingly.

2. Queue priorities are aged by 1SP. Each time 1SP is called, it adds 1 to the increment interval (byte 4) of the queue control word. If byte 4 is equal to the increment (byte 3), byte 4 is cleared and this serves as a flag to 1SP to add 1 to the queue priority for all files of the origin and queue type as long queue priority lies between the lower (LP) and upper (UP) aging bounds.

3. 1SP checks for CM and CPU time slice expirations. If either slice expires, the queue priority for the job is set to the lower bound for the origin.

4. Disable priority evaluation - yes via PRIORITY command; disable autoroll - yes via AUTOROLL command; job scheduling - no.

5. The scheduler may request a job be rolled out if it finds a candidate job with a higher priority needing field length or a control point so that it might begin/resume execution.

6. Job selection is done as follows.

   o Select the highest queue priority job that fits into unassigned or available memory within the service constraints (FL/EC, AM/EM) for the candidate's origin type.

   o For equal candidates, select the job from the device with the least mass storage activity as determined by:

     no free channel
     channel requested
     first unit reserved

   o When the mass storage activity is also equal, select the job with the largest FL requirement

   o If no job was selected, but one was rejected because of the service constraints, repeat the selection process above but consider only those jobs that will fit in central memory without rolling out other jobs. The job selected by this process will have its queue priority set to its origin's lower bound. (This step prevents the system from sitting idle during periods of low activity.)

7. Control point selection is done as follows.

   Considering a control point's field length to be all the FL of unoccupied control points following the candidate, the selection order is: choose the control point that is

   - an exact fit

   - the smallest hole larger than what is needed

   - the largest hole if none satisfies

   - if no control points are available or rolling out, the first control point encountered with a lower queue priority than the candidate is selected to be rolled out. If all control points have a higher QP, no control point is selected.

     Field length is then obtained based on the control point selected. The above sequence is such that the control point selected requires the minimum amount of storage movement to obtain the desired memory.

8. 1AJ is called when scheduling a job from the input queue while 1RI is called when scheduling a job from the rollout queue.

   If a PP is not available, 1SJ writes the 1AJ/1RI request into its input register, requests the scheduler via the RSJM function, and jumps to PP resident which will cause the new PP routine to be loaded into this PP.

9. CPUMTR calls 1AJ when there is no activity (zero status) or the rollout flag is set for the control point. This is called "job advancement".

10. 3AA is called by 1AJ to begin a new job. Typically this is the case on calls to 1AJ from the scheduler. 3AA will read the control statement record into the control statement buffer (CSBW), process the job card (which has already been cracked by OVJ), initialize the control point area, and initiate job processing.

11. The error processing overlay (3AB) is called by 1AJ upon the detection of an error flag in STSW or error exit or reprieve control in EECW. 3AB processes Extended Reprieve, EREXIT/ Reprieve, EXIT., and abort job processing. Depending on the type of abort and recovery options selected by the job, 3AB flushes output files (using the list of files (LOFW) pointer if present), issues diagnostics, takes dumps, and either recovers the job either in the same job step (reprieve and EREXIT) or another job step (EXIT.), or terminates the job.

12. Overlay TCS (Translate Control Statement) processes control statements and may be called via a RA+1 request or by the main 1AJ routine in a job advancement situation.

13. STIME. RTIME. CTIME. HTIME.

14. Operating system argument processing uses the actual separator when passing the arguments while Product Set format uses a code for the separator. This allows the Product Set format to use (eventually) 7 8-bit characters and a 4-bit code in a 60-bit word.

15. If a CPU program to be loaded is in the RCL or in an ECS ASR, the program is transferred into the control point field length by the LCEM monitor function. If SYSEDIT is active, however, the program will be loaded from mass storage. The LCEM function will use the fastest transfer mechanism available on the system (ECS, CMU, register-register). For programs being loaded from mass storage, the program is read from the system library (or local file) directly into the control point field length.

16. Absolutes and overlays are loaded directly into the field length by 1AJ. Relocatables are loaded by the CYBER Loader (which is an absolute).

17. A PP routine may be called from a control statement provided the caller is of system origin or has system origin privileges with DEBUG mode on. 1AJ searches the PLD for the routine and loads it into its PP.

18. 1DS has a table of subsystems. On function 33 (AUTO/MAINTENANCE) 1DS checks the SSTL word to see if the subsystem is enabled and if so, automatically initiates it. 1DS then checks SSCL if determine if the subsystem is already active, and if so, ignores the request to activate it. 1DS then builds an input file FNT/FST entry for the PP routine routine that will initiate the subsystem, including the control point requirement, initial FL, and subsystem QP. The scheduler will select this entry from the input queue and call overlay 3SA to initiate the PP initializer routine.

19. SYSMAX is the maximum FL currently available for any job, computed on a job origin basis. SYSMAX = machine size - CMR size - constant. MAXFL is the maximum field length the job may ever attain, computed on a job origin basis. MAXFL = min ¼ job card CM, validation CM, SYSMAX, service FL ½. MFL is the current job step maximum field length. RFL is the current field length (running field length). NFL is the nominal field length (same as RFL). SYSDEF is the default initial field length if no other FL can be determined and RFL is zero.

20. Initial field length is the first one of the following that applies:

    - o FL required specified by CLD field length control (RFL=or MFL=)
    - o Routine has required FL in loader (54) table
    - o RFL specified by an RFL macro or statement
    - o use the smaller of MFL and SYSDEF

21. Initial field length for ECS is the first one of the following that applies:

    - o routine has required ECS FL in a loader (54) table
    - o ECS FL has been specified by a RFL macro or statement

    ECS field length is not preserved between job steps unless protected via the PROTECT statement or macro.

22. The field length control in the CLD entry represents either a RFL= or a MFL= initial FL specification; the *FL directive simply allows a RFL=/MFL= to be specified if the program does not have either of these entry points.

    For RFL=, the specified field length will be set as the RFL value in FLCW and used as the initial field length.

    For MFL=, if the MFL= control flag is 0, then the maximum of the existing FL (STSW) and the specified MFL will be used as the initial running field length. If the MFL= control flag is 1, the the maximum of the current RFL (FLCW) and the specified MFL will be used as the initial running field length.

23. The DMP RA+1 request is processed by PP routine SFP. SFP makes a SPCW call for DMP, issuing a ROCM on the control point. 1AJ is called and senses a SPCW request. It (1AJ) searches the CLD for entry point DMP, which it finds in routine CPMEM. CPMEM has a DMP=entry point. 1AJ calls 1RO to process the DMP=value, creating a DM* file with the appropriate field length from the calling program. 1AJ returns and loads CPMEM into the field length space written on DM* by 1RO. CPMEM begins execution at the DMP entry point. When CPMEM completes and 1AJ is called to advance the job, it senses the completed SPCW call, reloads the original program field length from the DM* file by calling 1RI. When 1AJ resumes control after 1RI has restored the field length, control returns to the program at the instruction word after the DMP RA+1 call.

24. The SSJ= entry point allows the program containing it to have certain system privileges not normally given to CPU programs, such as using Fast Attach files; allows the program to specify time limit, CPU and Queue priorities; and may transfer user validation between the control point area and the user program field length. The UIDW, ALMW, ACLW, and AACW control point words are transferred to the program's field length when loaded, and are reset in the control point area when the program terminates. Any time limit, CPU or queue priority values specified in the SSJ block are entered into the control point area when the program is loaded, with the current values entered into the SSJ block, from which they will be reset when the job step terminates. Files created by a SSJ= program have special file identification SSID; these files will be automatically returned at the end of the job step.

ANSWER SET LESSON 15

1. A system resource unit (SRU) is a measure of resources used by a job or terminal session. The SRU combines measurement of central memory field length, ECS field length, CPU time, mass storage, magnetic tape and permanent file usage into a single unit. The algorithm is:

   SRU=M1(CP+M2*IO)+M3(CP+IO)CM+M4(CP+IO)EC)+AD

2. ADD applies the adder to the SRU accumulator: SRU = SRU + AD;

   AIO applies the I/O increment to the SRU accumulator:
   IO = S2*MS + S3*MT + S4*PF then SRU = SRU + IO*IOM;

   CPT applies CPU time to the SRU accumulator:
   CP = S0*CP0 + S1*CP1 then SRU = SRU + CP*CPM;

   SRU calculates the SRU multipliers CPM and IOM when field lengths change:
   CPM = M1 + M1*M3 + M1*M4 and IOM = M1*M2 + M1*M3 + M1*M4.

3. An "account block" is the SRU accumulation from one CHARGE statement to the next (or end of job). In effect, it is the SRU accumulation for a given charge/project.

4. The VALIDUs file contains limits on resource usage, various permissions, and default TELEX terminal characteristics for each user number. The VALINDs file indicates the user indices that have been assigned to user numbers. Each bit in this file represents a user index; if the bit is set, the corresponding user index is in use.

5. The PROFILa file contains charge numbers, project numbers, SRU multipliers M1 - M4 and AD, allowed access times, user numbers permitted to use the project numbers, and project SRU accumulation and limit values.

6. The only relationship is that user numbers are used in the project entries to (optionally) restrict project usage to those users.

7. GEAC uniquely describes the conditions that prompted the issuance of the ACCOUNT file entry: G = group, E = event, AC = activity.

8. The VALIDUs level-2 block contains validation entries for up to four user numbers.

9. The level-1 block contains the charge number, master user number, and SRU multipliers M1 - M4 and AD. The level-2 block contains the project numbers. The level-3 block contains detailed project information.

ANSWER SET LESSON 16

1. "Holes" in the indirect chain are pointed to by empty catalog entries (UI = 0).

2. "Master" users have implicit read permission for permanent files cataloged under user numbers that match in non-asterisk positions. Therefore, user USER*** has read permission for USERXYZ's files.

3. Both a semi-private file or a library (public) file can be accessed by any user by specifying the permanent file name, the user number under which it is cataloged, and the password (if defined). The system records the number of times the file was accessed for either file category. However, for semi-private files the user number and last access date/time of the accessor are also recorded.

4. Indirect files must reside on the user's master device since that is where the indirect chain resides. Direct access permanent files may reside on any device within the family that has the appropriate secondary mask bit set for the user's user index.

5. For direct access permanent files, each user attaching the file in read mode will have their own FNT/FST entry for the file. The number of users accessing the file is kept in the DAF's system sector.

6. Each user receives a complete local copy of the permanent file and manipulates that local file rather than the original permanent file.

7. Attaching a direct access permanent file with "write" permission essentially locks the file in that no other user can access it until it is returned. This prevents two users from modifying the file at the same time. However, if any users have attached the file in read mode prior to the attempt to attach in write mode, the write requestor will be aborted. The write accessor may choose to wait until the file becomes available for write access by using the NA (no abort) option.

8. When a direct access file is purged, the user index in the catalog entry is cleared and all tracks released. When an indirect access file is purged, the user index in the catalog entry is cleared (creating a hole in the indirect chain.) If the amount of space returned includes one or more logical track, the track(s) is(are) returned to the system and the indirect chain relinked by the DLKM (Delink Tracks) monitor function. The catalog entry is updated to reflect the new length of the hole.

9. If the appropriate secondary mask is not set, the DEFINE is aborted. This situation easily occurs if the existing file is not on the proper device when the DEFINE is performed.

10. PFM searches the user's catalog track for a hole that is an exact fit (same number of sectors), and if one exists, it is used. If there is no exact fit, then PFM uses the largest hole and creates a new hole entry for the residue (if more than one sector). If no holes satisfy the request, a new catalog

entry is created at the end of the catalog chain. The file is then copied into the hole space or at the end of the data chain. For a REPLACE, the file is copied over the existing entry if it is an exact fit or if the new file is smaller, in which case, a hole will be created for residues larger than one sector. If the new file is larger, a hole will be made of the existing entry, and the process described above will be followed to create a new catalog entry for the file.

11 - 14. The following table contains the answers for questions 11 through 14.

	EQ 6	EQ 7	EQ 12	EQ13	EQ 14
Permit FT (11)	4043	4043	4013	4013	4044
Data FT (11)	4042	4042	4012	4012	4043
Data Length (12)	52D+265B	44D+24B	27D+334B	28D+275B	60D+252B
Permit Length (13)	151	231	250	226	227
Catalog tracks (13)	40	40	10	10	40
Device Mask (14)	140	10	3	200	24
Secondary Mask (14)	140	10	3	200	24

Data length is in tracks + sectors;
Permit length is in sectors.

15. Catalog track overflow is indicated by bit 57 in MST word ACGL. None of these master devices has catalog track overflow.

16. 4001 4002 4003 4004
4005 4006 4007 4010

17. 0, MD = 7, SD = 7 and 6
1, MD = 6, SD = 7 and 6
2, MD = 5, SD = 5 and 4
3, MD = 4, SD = 5 and 4
4, MD = 4, SD = 5 and 4
5, MD = 5, SD = 5 and 4
6, MD = 6, SD = 7 and 6
7, MD = 7, SD = 7 and 6

18. 1231, MD = 6, CT = 23 = 4024
1237, MD = 7, CT = 3 = 4004

19. Device 4 has 40B catalog tracks.
Device 5 has 40B catalog tracks.

20. TRT linkage will link track 4010 to track 4234. Mass storage linkage with the catalog tracks will link from the last sector of track 4002 to track 4234 sector 0.

ANSWER SET LESSON 17

1. Control statements processed by RESEX are:

   LABEL
   ASSIGN
   REQUEST
   RESOURC
   VSN

2. Deadlock prevention is a process performed by RESEX to guarantee that the assignment of a tape or pack will not cause a deadlock on tape/pack resources. All jobs must still be able to complete after the requestor has been assigned the requested resource.

3. RSXDid is the demand file. In addition to a header that identifies the job for the demand file entry, each entry contains assign counts, demand counts, MT/RP validation limits, preview data, and the share table. RSXVid is the VSN file. In addition to the header, it contains the file name, the demand entry address and index to the resource entry within the demand entry, and volume serial numbers (VSNs). The VSN entries end with a code that describes the VSN usage ("0" = end of VSNS, "/" = multiple reels, "=" = equivalenced VSNs).

4. The assignment order within RESEX is:

   a. search environment for VSN or PACKNAME
   b. perform deadlock prevention (overcommitment algorithm)
   c. send call block to MAGNET via SIC with tape data
   d. make share table entry if removable pack
   e. update demand file entry

5. An "internal call" is the assignment of a tape resource when called by a control statement; the arguments are processed, then the resource assignment process is entered. An "external call" is the entry into the assignment process via the special call (SPCW) mechanism because of user RA+1 assignment requests (PFM, LFM, REQ).

6. A "deadlock" or "potential deadlock" condition exists when two or more jobs demand resource units such that no more resources are available (deadlock) or there are not enough free resource units available to satisfy the resource requirements of these jobs (potential deadlock).

   A "resource unit" is any magnetic tape drive or removable mass storage equipment.

   The "resource type" is the mnemonic that refers to a type of resource unit. Tape resource types are the tape densities (LO, HI, HY, HD, PE, GE) and removable pack types are the mass storage mnemonic and number of spindles (DI1, DK2).

The "demand count" is the number of resource units required by the job and may be used to refer to an individual resource type or to the demand for all resource types. The "assigned count" is the number of resource units currently assigned to the job, both on an individual and total resource type basis.

The "demand file" is a fast attach permanent file that contains job demand information for all jobs in the system. The "VSN file" is a fast attach permanent file that contains the file name/VSN correspondence for all tape files in the system.

The "demand file entry" is the job's entry on the demand file containing its resource demands and assignment counts.

A "VSN file entry" is a entry on the VSN file that consists of Volume Serial Numbers associated with a given tape file.

The "resource environment" is a snapshot of all the resource units in the system, what tapes or packs are mounted on then, and the jobs to which they are assigned. The environment is built from the EST, MSTs, and UDTs.

The "share table" is part of a job's demand file entry and is used to keep track of which removable packs are assigned to the job.

"Pack sharing" is the ability for more than one job to use the same removable pack. If the pack is mounted, then it can be shared. A job cannot ever assume that pack sharing will or will not take place.

The term "overcommitment algorithm" is used to describe the entire deadlock prevention mechanism as well as the main subroutine of that process, OCA (Overcommitment Algorithm).

ANSWER SET LESSON 18

1. The UDT contains hardware properties of the tape unit, software and processing properties associated with the tape being manipulated, label information, operation control information for interaction between MAGNET and 1MT, and job assignment information.

2. CIO writes its requests to MAGNET directly into the tape unit's UDT via the TDAM monitor function.

   RESEX makes requests of MAGNET through its call block RCAL using the SIC RA+1 request. The RESEX call block and the preview buffer are the two buffers identified for SIC communication in the ICAW word.

   Other PP routines (1DS, REC, ORF, ...) make requests to MAGNET by writing their requests in the external request buffer XREQ using TDAM. The functions performed via this mechanism are:

   0 - return unit
   1 - enter VSN
   2 - unload VSN
   3 - scratch VSN
   4 - up/down channel
   5 - on/off unit

3. RESEX sends compressed data via a SIC RA+1 request to the preview buffer PBUF in MAGNET's field length. DSD processes this compressed data while painting the E,P display.

4. Tape labels uniquely identify the file and the reel on which it resides, and they mark the beginning and end of a tape file and reel.

   VOL1 contains the VSN, volume accessibility code, and owner identification.

   HDR1 contains the file identifier, set identifier, file section number, file sequence number, generation number, generation version number, file accessibility, creation date, expiration date, and system identification code.

   EOF1 matches the HDR1 label but additionally contains a block count of the number of PRUs in the file.

   EOV1 matches the VOL1 label but additionally contains a block count of the number of PRUs in the reel.

5. 1MT contains many overlays that perform given functions. As 1MT and each overlay are loaded, it determines various tape properties for the tape unit being processed from the UDT and presets operations (function codes, error recovery, etc.) depending upon the type of tape drive. This preset operation allows the processing logic to be generally written; that is, there need not be a separate driver for each type of tape hardware.

ANSWER SET LESSON 19

1. 1IO (executive), 1CD (driver), QAP (auxiliary), and COMSBIO (common deck).

2. BATCHIO's field length is used for communication, FETs and buffers. There is no CPU code for BATCHIO.

3. When in the "idle" state, 1IO is recalled from RLPW to status unit record equipment. If there is no unit record activity to be done, 1IO writes its input register into RLPW and drops its PP, going on recall.

4. 1CD does mass storage input/output by making CIO requests.

5. 1IO calls 1CD which in turn calls QAP to create an input queue entry (a FNT/FST entry of type INFT). QAP call OVJ to validate the job card and generate the jobname hash.

6. Example.

   The LQyy,xx. DSD command allows the operator to set an ID on a particular card reader, punch, or printer. Jobs that enter via a card reader with an ID of xx will have their output processed on printers or punches with IDs of xx. When 1IO calls QAC to search for disposable output, it specifies the TID field (in this case the ID of the peripheral) on which a match should occur.

7. The DCWs serve to indicate whether a copy of 1CD is active and how many peripherals that copy is driving.

8. The DRQR is used to request an active copy of 1CD to perform a given operation. The copy of 1CD that should perform the operation, the operation and the buffer point are specified in the DRQR request.

9. BATCHIO is a subsystem with a queue priority greater than MXPS: it cannot be rolled out.

ANSWER SET LESSON 20

1. All routing and disposition information is found in the system sector.

2. COMPUSS. System sector definitions are found in COMSSSE.

3. "Terminal addressing" is the identification of the family/user number/user index of the remote batch terminal that can process the output or has submitted the input. The terminal ID (TID) field is found in the FST with expanded information being found in system sector words FDSS, DASS, and DISS.

4. F - forms code, X - external characteristics, D - device code.

   There is limited space within the FST for these values. If a value beyond the FST field is necessary, the system sector must be read to obtain the actual value.

5. A deferred route is the setting up of the system sector without disposing the file. The system sector information bit (bit 5 in the FNT) is set when the field is necessary, the system sector must be read to obtain the actual value.

6. A deferred route is the setting up of the system sector without disposing the file. The system sector information bit (bit 5 in the FNT) is set when the system sector already contains the routing and disposal information.

7. QAP is part of File Routing and Queue Management since it is used by both BATCHIO and RBF to process disposed output files in a uniform manner, thus giving consistent accounting, banner pages, and so forth for all printer output.

ANSWER SET LESSON 22

1. A system control point must be a subsystem (i.e., have a unique queue priority assigned and associated with a byte in the SSCL words of CMR). A system control point processes requests from user control points. A user control point requires no special privileges to access the System Control Point Facility but must know the queue priority and protocol of the subsystem it will use. THE SYSTEM control point is the last control point plus 1 and is the "control point" for CPUMTR program mode executions.

2. The user control point uses the SSC RA+1 call or the CALLSS macro to transfer an array of request data to a subsystem. The user control point uses the subsystem queue priority to specify the system control point being accessed.

   The system control point uses the SSF RA+1 call or the SFCALL macro to transfer data to or from a user control point's field length. The SSF call or SFCALL macro accesses a parameter array which includes the transfer buffer address in both the system control point and the user control point. Actual data transfers are done by the SCP module of CPUMTR in program mode.

ANSWER SET LESSON 25

1. EI200's control point field length is used for communication buffers, I/O buffers, port tables, a "mini"-RPL of 1LS overlays, and data conversion subroutines.

2. Port table and input/output FETs and buffers.

3. It uses CIO.

4. Data is transferred from the terminal over the communication lines to the multiplexor. 1ED takes the data from the mux and passes it to a line buffer within the EI200 control point's field length. The data is converted into a FET buffer from which it can be written to mass storage by CIO as the buffer fills. At the end of the file transfer, 1LS and XSP cause the file to be validated and entered into the input queue.

   For output, 1LS retrieves the file for the RJE terminal from the output queue. A banner page is formatted into the FET buffer and the buffer is filled by reading the file using CIO. Data is taken from the FET buffer and converted into the line buffer. 1ED reads the data from the line buffer, sends it to the multiplexor, which in turn passes on to the terminal.

5. The terminal ID and destination information from the system sector are matched with the family/user number/user index of logged-in remote batch terminals. The highest priority file is then disposed by 1LS, based on the destination information.

6. XSP processes drop job requests, logs in terminals, and makes initial input queue entries for jobs read from the remote batch terminal.

7. While 1ED is transmitting print data or receiving card reader input from the terminal, it monitors the keyboard. All keyboard entries except the interrupt key are ignored. If 1ED senses the interrupt key, it will stop transmitting and receive input from the keyboard. At that time, the terminal operator can suspend or end the operation.

ANSWER SET LESSON 26

1. The typical end user knows only a login/logoff procedure and a set of scripts that need to be followed for each type of transaction. The end user does no programming and for the most part simply "fills in the blanks".

2. A transaction is an inquiry or change of a data base. A transaction has three phases: data acquisition, processing of data, and data output.

3. A task is a small piece of coding. One or more of these pieces (tasks) make up a transaction. These tasks are used to build the transaction application system. A task may be written in COBOL, FORTRAN or COMPASS, and a total transaction may involve tasks written in any (or all) of these languages. All tasks may be collected onto a library file (Task Library) maintained by the utility LIBTASK.

4. The five components of TAF are: NAM or TELEX to control terminal input/output; TAF, the executive that controls the execution of applications and data managers; Data Manager; User Tasks; and Utility programs.

5. Yes.

6. The transaction executive coordinates and controls the execution of applications and the data manager and affords task isolation and recovery mechanism as well as interfacing to the input/output mechanisms.

ANSWER SET LESSON 27

1. The MRT indicates the local or shared status of the tracks reserved on a shared mass storage device. The MRT is used to clear interlocks on preserved files that are held by a down machine, and to clean up local file usage by a down machine under MREC control.

2. The following procedure is used to retrieve the "up-to-date" TRT.

   Determine if device is shared by examining SDGL for the ECS address of the MST.

   Obtain the TRTI interlock.

   Read first three words of global MST.

   Reject function if another machine's mask is found in SDGL.

   Examine SDGL to determine if this machine was the last accessor; if so, this machine has the "up-to-date" TRT. Otherwise, the MST and TRT are read from ECS to the ECS buffer and then moved to the MST/TRT area for the device. (This allows error recovery without destroying good data.)

   Set the machine mask in SDGL and write these three words back to ECS to interlock this operation.

   Release TRTI interlock.

3. The "machine ID" is a two character identifier for the machine supplied by CMRDECK entry MID. The "machine index" is the position that the machine's MMFL word is entered in MFET when the machine is introduced into the complex. The "machine mask" is a mask bit 2 ** (machine index - 1) that is used for identifying interlock holders. Machine index may be 1, 2, 3, 4 and machine mask may be 1, 2, 4, 10.

4. The main MMF tables kept in ECS are: the MMF Environment Table, the Intermachine Communication Area, the Device Access Tables, the Fast Attach Table and the global PFNL. The MST/TRT/MRT for shared devices are part of the DAT.

5. Every two seconds, the clock area is read from ECS. If a machine's clock has not changed every other read (four seconds), that machine will be declared "down".

6. MREC clears up hardware reservations preventing other machines from accessing shared devices; clears access information in the DAT; clears interlocks and user counts in system sectors of files shared with the down machine; processes the MRT to release all local space held by the down machine.

   Once MREC has been run for a down machine, all information about that machine has been cleared. The only way to re-introduce the machine is by a level 0 deadstart.

ANSWER SET LESSON 28

1. Hardware components of the Mass Storage Facility are:

   CYBER Mass Storage Coupler which interfaces the 12-bit CDC CYBER peripheral processor with the 8-bit byte-oriented Mass Storage Adapter.

   Mass Storage Adapter is a micro-programmable processor providing the overall control of the Mass Storage Facility in response to functions received from the CYBER PP via the Mass Storage Coupler.

   Cartridge Storage Unit stores cartridges in a matrix and transfers them between input/output drawers, the matrix, and Mass Storage Transport I/O stations.

   Mass Storage Transport exchanges cartridges with the Cartridge Storage Unit and transfers data between the cartridge tape and the Mass Storage Adapter.

   Mass Storage Cartridge consists of a plastic case containing the magnetic tape used for data recording.

2. In a multimainframe complex only one mainframe can be driving the Mass Storage Facility. This mainframe is the master mainframe and contains the Mass Storage Subsystem master executive. The primary differences between master and slave executives are:

   Master executive contains the driver interface and the CPU portion of the driver.

   Master executive requests the actual staging and destaging of files between disk storage and tape cartridges.

   Slave executive processes all file staging requests by transferring the request to the master executive via communication files in the link device.

3. Mass Storage Subsystem utilities are:

ASMOVE	provides the logic for selecting which direct access permanent files are candidates for staging to the Mass Storage Facility
ASLABEL	provides the cartridge management functions for Mass Storage Facility
ASDEF	initializes the Mass Storage Subsystem system files necessary for mass storage processing, i.e., the CSUMAP files and the MSF catalog files.
ASVAL	examines the MSF catalog files and the permanent file catalogs and issues a report on any irregularities or discrepancies found. This utility also releases Mass Storage Facility space for files that have been purged from the permanent file base.
ASUSE	produces reports detailing cartridge usage.
ASDEBUG	dumps data from cartridges in raw form for problem analysis. This utility also releases Mass Storage Facility space or system file entries to resolve previously detailed inconsistencies.

EVALUATION FORM

Course/Seminar Name ____________________ **Date of Attendance** **From** ________ **To** ________

Instructor ________________________________ **Location** ____________________________________

Please place a rating in the box for each area and then add comments explaining your rating.

Rating Key

Excellent	*5*
Very Good	*4*
Good	*3*
Fair	*2*
Poor	*1*

The Course/Seminar

- * How well did the course/seminar cover the stated objectives? ☐
- * To what degree will the course/seminar be helpful in improving on-the-job performance? ☐
- * To what extent were the handout materials and visuals helpful in aiding your understanding of the topic? ☐
- * What is your overall rating of the organization and content of the course/seminar? ☐

The Instructor

- * How do you rate the instructor's knowledge of the material and ability to answer questions? ☐
- * How effective was the instructor in presenting the material in an understandable manner? ☐
- * How effective was the instructor in generating and sustaining interest in the course/seminar? ☐
- * How do you rate the instructor's responsiveness to the needs of participants? ☐
- * What is your overall rating of the instructor? ☐

The Facilities

- * How do you rate the appropriateness of the facilities to the topic and means of presentation? ☐
- * To what extent were the facilities comfortable, well-lighted and heated or cooled? ☐
- * How convenient was the location of the facility? ☐

EVALUATION FORM

Page 2

General Comments

* What changes in the course/seminar would you make if you were the instructor?

* Would you recommend this course/seminar to others in your company or department? Why?

* Please list colleagues or associates who should receive advance notices of similar courses/seminars.

1) Name ____________________

Organization ____________________

Address ____________________

____

Bus. Tel. No. ____________________

2) Name ____________________

Organization ____________________

Address ____________________

____

Bus. Tel. No. ____________________

3) Name ____________________

Organization ____________________

Address ____________________

____

Bus. Tel. No. ____________________

4) Name ____________________

Organization ____________________

Address ____________________

____

Bus. Tel. No. ____________________

* Should this course be offered at your company site? If so, who should be contacted to manage it?

* If we may use your comments in future descriptions of the course/seminar, please sign below.

Signature ____________________

(Optional)

PARTICIPANT INFORMATION FORM

In order for our seminars/courses to be most effective, they need to take into account the characteristics, needs and objectives of the people who attend them. The information asked for below will assist us in keeping our presentations relevant to the participants and in developing and scheduling new presentations that will meet participant needs. Please complete this form and leave it with the presenter at the next break.

Seminar/Course Title ____________________ **Date of Presentation** ____________________

Name ____________________ **Field or Type of Business** ____________________

Title ____________________ **Years of Experience** ____________________

Business Address ____________________ **Supervisor's Title** ____________________

____________________ **Last professional degree** ____________________

____

List your three primary objectives in attending this seminar.

1. ____
2. ____
3. ____

Will this course/seminar be credited toward certification/training requirements? __________

Rank in order of importance in your choice of this seminar session.

Instructor ______ **Date** ______ **Location** ______ **Employer's Preference** ______

Previous courses/seminars attended relating to this topic.

1. ______________________________
2. ______________________________
3. ______________________________

Topics for additional courses/seminars in which you would be interested.

1. ______________________________
2. ______________________________
3. ______________________________

PARTICIPANT INFORMATION FORM

What trade journals/magazines do you regularly read or subscribe to in order to keep abreast in your profession?

1. ________________________________
2. ________________________________
3. ________________________________

How did you become aware of this course/seminar?

Schedule/Catalogue ________,

Direct Mail Brochure ________,

Recommendations of Supervisor ________,

Recommendation of Colleague ________,

Corporate Training Department ________,

Other ________________________________.

COMMENT SHEET

MANUAL TITLE: NOS ANALYSIS

PUBLICATION NO.: FH4010-1 **REVISION:** C

NAME:____________________

COMPANY:____________________

STREET ADDRESS:____________________

CITY:__________ **STATE:**__________ **ZIP CODE:**__________

This form is not intended to be used as an order blank. Control Data Corporation welcomes your evaluation of this manual. Please indicate any errors, suggested additions or deletions, or general comments below (please include page number references).

NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

FOLD ON DOTTED LINES AND STAPLE

CUT ALONG LINE

AA3419 REV. 4/79 PRINTED IN U.S.A.

STAPLE

STAPLE

FOLD

FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 8241 MINNEAPOLIS, MINN.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION
National Coordinator
Bloomington Facility (MNA02B)
5001 West 80th Street
Bloomington, Minnesota 55437
Attn: Curtis Vicha

CUT ALONG LINE

FOLD

FOLD

CONTROL DATA SEMINARS

an education service of
CONTROL DATA CORPORATION

CORPORATE HEADQUARTERS
P.O. BOX 0
MINNEAPOLIS, MINNESOTA 55440