60494100

CONTROL DATA CORPORATION

NOS/BE VERSION 1 SYSTEM PROGRAMMER'S REFERENCE MANUAL

CDC® COMPUTER SYSTEMS:
CYBER 170 SERIES
CYBER 70
MODELS 71, 72, 73, 74
6000 SERIES

PAGE INDEX TO MACROS

Macro	Page	Macro	Page
ACCSF	7-14	READSKP	5-9, 24
ACQUIRE	4-5	RECOVR	7-18
		REPRIEVE	7-18
BKSP	5-17	RETURN	2-16
		ROUTE	4-3
CALL	2-16		
CALLSS	2-22	SEGDEF	2-16
CEVAL	4-1	SEGMFL	2-37
		SETLC	7-25
DISPOSE	4-3	SETLOF	7-18
		SFCALL	2-26
ENCSF	7-15	SKIPB	5-17, 26
ENDSEG	2-16	SKIPF	5-16, 25
		SSCT	2-37
GETLC	7-25	SYSTEM	6-28; 7-3
GETLOF	7-18		
GOTO	2-16; 10-3	VERIFYJ	4-16
GOTOTAB	2-16		
		WRITE	5-11, 25
READ	5-6, 24	WRITEC	5-20, 25
READC	5-18, 25	WRITEF	5-13, 25
READLS	5-19, 25	WRITER	5-12, 25
READNS	5-24		

60494100

CONTROL DATA CORPORATION

NOS/BE VERSION 1 SYSTEM PROGRAMMER'S REFERENCE MANUAL

CDC® COMPUTER SYSTEMS:
CYBER 170 SERIES
CYBER 70
MODELS 71, 72, 73, 74
6000 SERIES

REVISION RECORD

REVISION	DESCRIPTION
A	Manual released.
(11-01-75)	
B	Revised to reflect features and PSRs added with the release of NOS/BE Version 1.1. Features included
(07-16-76)	are Factory Format Support (83), Stack Processor Enhancements (133), INTERCOM Restart, 844-41 Support (145),
	844 Expander Support (169), Job Management and Systems Control Point Enhancements (159, 163), 2X PPU
	Support, and FORMAT/FDP Version 2 Enhancements.
C	Manual revised to support NOS/BE Version 1.2 at PSR level 447 and to make editorial and technical
(04-22-77)	corrections. The following features are documented: CYBER Control Language (CCL), Fast Dynamic Loader
	(FDL), Reliability Feature Utilization (System Idle), CE Validation, Product Set Support, INTERCOM
	Enhancements, Advanced Tape Subsystem (ATS), 580 Line Printer Programmable Format Control (PFC), 844 Full
	Tracking, and support of the CYBER 170 Model 176 and 819 disk drive. References to 604 and 607 tape units, and
	the 501 line printer are removed.
D	Updated to support NOS/BE Version 1.2 at PSR level 454 and to make editorial and technical correc-
(08-19-77)	tions. Support of CYBER 170 Model 171 is included.
E	Manual revised to support NOS/BE Version 1.3 at PSR level 473 and to make editorial and technical
(06-13-78)	corrections. New features documented are permanent file utilities PFLOG GENLDPF; user capability to assign
	universal password and permissions to private sets; user reprieve processing; support of all 677/679 tape units;
	option to schedule tapes by density; support of INTERCOM 5; SF.RERN function; SETMFL, GETLOF, SETLOF,
	GETLC, and SETLC macros. This edition obsoletes all previous editions.
F	Manual revised to support NOS/BE Version 1.3 at PSR level 481. Information is added in the INTERCOM
(10-20-78)	5 Pointer and Buffer Area table to support the capability to turn communication lines on and off. Miscellaneous
	technical corrections and clarifications are made.
G	Manual revised to support NOS/BE Version 1.3 at PSR summary level 488. New features documented are
(02-19-79)	FNT space threshold; direct access user ECS swapping; sequencer jobs; and VSN parameter on GETPF, SAVEPF,
	and PURGE. Miscellaneous technical corrections and clarifications are included.
H	Manual revised to support NOS/BE Version 1.3, at PSR summary level 499. New features documented
(07-02-79)	are EXPORT High Speed (PP routines and EXPORT Multiplexer Subtable) and Improved Load Leveling (STF
	routine). Miscellaneous technical corrections and clarifications are included.
J	Manual revised to support NOS/BE Version 1.4, at PSR summary level 508. New features documented
(12-21-79)	are Common Test and Initialization (CTI), deadstart dump analyzer, 885 disk drive support, enhancements to the
	CEVAL macro, and reorganization of monitor functions. Information duplicated in the NOS/BE Reference Manual
	on reprieve processing and permanent file utilities is removed. Miscellaneous technical corrections and
	clarifications are included. This edition obsoletes all previous editions.
K	Manual revised to support NOS/BE Version 1.4 at PSR summary level 518. New features documented are
(05-19-80)	downline load utility, remote batch accounting for INTERCOM 5, and user capability to log error information for
	ECS errors. Information duplicated in the NOS/BE Reference Manual on permanent file macros is removed.
	Miscellaneous technical corrections and clarifications are included.

Publication No.
60494100

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

Address comments concerning this manual to:
Control Data Corporation
Publications and Graphics Division
4201 North Lexington Avenue
St. Paul, Minnesota 55112

or use Comment Sheet in the back of this manual.

LIST OF EFFECTIVE PAGES

New features, as well as changes, deletions, and additions to information in this manual, are indicated by bars in the margins or by a dot near the page number if the entire page is affected. A bar by the page number indicates pagination rather than content has changed.

PAGE	REV
Front Cover	-
Title Page	-
ii	K
iii	K
iv	K
v	K
vi	J
vii	K
viii	K
ix	K
x	K
xi	K
1-1	J
1-2	J
1-3	J
1-4	J
1-5	J
1-6	J
2-1	K
2-2	K
2-3	J
2-4	J
2-5	J
2-6	K
2-7	J
2-8	K
2-9	K
2-10	J
2-11	K
2-12	K
2-13	J
2-14	J
2-15	J
2-16	J
2-17	J
2-18	J
2-19	J
2-20	K
2-20.1	K
2-20.2	K
2-21	J
2-22	J
2-23	J
2-24	J
2-25	J
2-26	J
2-27	J
2-28	J
2-29	J
2-30	J
2-31	J
2-32	J
2-33	J
2-34	J
2-35	J
2-36	J
2-37	J
2-38	J
3-1	J
3-2	J
3-3	J
3-4	J
3-5	J
3-6	J
3-7	J
3-8	J
3-9	J
3-10	J
3-11	J
3-12	J
3-13	J
3-14	J
3-15	J
3-16	J
3-17	J
3-18	J
3-19	J
3-20	J
3-21	J
3-22	J
3-23	K
3-24	J
3-25	K
3-26	K
3-26.1/3-26.2	K
3-27	J
3-28	J
3-29	J
3-30	J
3-31	J
3-32	J
4-1	J
4-2	J
4-3	J
4-4	J
4-5	J
4-6	J
4-7	J
4-8	J
4-9	J
4-10	J
4-11	J
4-12	J
4-13	J
4-14	J
4-15	J
4-16	J
4-17	J
4-18	J
4-19	J
4-20	J
4-21	J
4-22	J
4-23	J
4-24	J
4-25	J
4-26	J
4-27	J
4-28	J
4-29	J
4-30	J
4-31	J
4-32	J
4-33	J
4-34	J
4-35	J
4-36	J
4-37	J
4-38	J
5-1	J
5-2	J
5-3	J
5-4	J
5-5	J
5-6	J
5-7	J
5-8	J
5-9	J
5-10	J
5-11	J
5-12	J
5-13	J
5-14	J
5-15	J
5-16	J
5-17	J
5-18	J
5-19	J
5-20	J
5-21	J
5-22	J
5-23	K
5-24	J
5-25	J
5-26	J
5-27	J
5-28	K
5-29	J
5-30	K
5-31	K
5-32	K
5-32.1/5-32.2	K
5-33	J
5-34	J
5-35	J
5-36	J
5-37	J
5-38	J
5-39	K
5-40	J
5-41	K
5-42	J
5-43	J
5-44	J
5-45	J
5-46	J
5-47	J
5-48	J
5-49	J
5-50	J
5-50.1/5-50.2	K
5-51	J
5-52	J
5-53	J
5-54	J
5-54.1/5-54.2	K
6-1	K
6-2	K
6-3	K
6-4	K
6-5	K
6-6	K
6-7	K
6-8	K
6-9	K
6-10	K
6-11	K
6-12	K
6-13	K
6-14	K
6-15	K
7-1	J
7-2	J
7-3	J
7-4	J
7-5	J
7-6	J
7-7	J
7-8	J
7-9	J
7-10	J
7-11	J
7-12	J
7-13	J
7-14	J
7-15	J
7-16	K
7-17	J
7-18	J
7-19	J
7-20	J
7-21	J
7-22	J
7-23	J
7-24	J
7-25	J
8-1	K
8-2	K
8-3	J
8-4	J
8-5	J
8-6	J
8-7	J
8-8	J
8-9	J
8-10	J
8-11	K
8-12	K
8-13	J
8-14	J
8-15	J
8-16	J
8-17	J
8-18	J
8-19	J
8-20	J
8-21	J
8-22	J
8-23	J
8-24	J
8-25	J
8-26	J
8-27	J
8-28	J
8-29	J
8-30	K
8-31	K
9-1	J
9-2	J
9-3	J
9-4	J
10-1	J
10-2	J
10-3	J
10-4	J
10-5	J
10-6	J
11-1	J
11-2	J
11-3	J
11-4	J
11-5	J
11-6	J
11-7	J
11-8	J
11-9	J
11-10	J
12-1	K
12-2	K
12-3	K
12-4	K
12-5	K
12-6	K
12-7	K
12-8	K
A-1	J
A-2	J
A-3	J
A-4	J
A-5	J
A-6	J
A-7	J
A-8	J
B-1	K
B-2	K

PAGE	REV
B-2.1/B-2.2	K
B-3	J
B-4	J
B-5	K
B-6	K
B-7	J
B-8	J
B-9	J
B-10	J
B-11	K
B-12	J
B-13	J
B-14	K
B-15	K
B-16	K
B-17	K
B-18	K
B-19	K
B-20	K
B-21	J
B-22	J
B-23	J
B-24	J
B-25	J
B-26	K
B-27	K
B-28	J
B-29	K
B-30	J
B-31	K
B-32	J
B-33	J
B-34	K
B-35	K
B-36	K
B-37	J
B-38	K
B-39	K
B-40	K
B-41	K
B-42	K
B-43	K
B-44	J
B-45	K
B-46	K
B-47	J
B-48	K
B-49	J
B-50	K
B-51	J
B-52	K
B-52.1/B-52.2	K
B-53	J
B-54	J
B-55	K
B-56	J
B-57	J
B-58	J
B-59	J
B-60	J
B-61	J
B-62	K
B-63	K
B-64	K
B-65	J
B-66	J
B-67	J
B-68	J
B-69	J
B-70	J
B-71	J

PAGE	REV
B-72	K
B-73	J
B-74	J
B-75	J
B-76	J
B-77	J
B-78	J
B-79	J
B-80	K
B-81	K
B-82	J
B-83	J
B-84	J
B-85	J
B-86	K
B-87	J
B-88	J
B-89	J
B-90	J
B-91	J
B-92	K
B-93	K
B-94	J
B-95	K
B-96	J
B-97	K
B-98	K
B-98.1/B-98.2	K
B-99	K
B-100	J
B-101	K
B-102	K
B-103	J
B-104	K
B-105	J
B-106	J
B-107	J
B-108	J
B-109	K
B-110	K
B-111	J
B-112	K
B-113	K
B-114	K
B-114.1	K
B-114.2	K
B-115	K
B-116	K
B-117	J
B-118	K
B-119	J
B-120	K
B-121	J
B-122	J
B-123	J
B-124	J
B-125	K
B-126	K
B-127	K
B-128	K
B-129	K
B-130	K
B-131	K
B-132	K
B-133	K
B-134	K
B-135	K
B-136	K
B-137	K
B-138	K
B-139	K

PAGE	REV
B-140	K
B-141	K
B-142	K
B-143	K
B-144	J
B-145	J
B-146	J
B-147	J
B-148	J
B-149	J
B-150	J
B-151	J
B-152	J
B-153	J
B-154	K
B-155	J
B-156	J
B-157	J
B-158	J
B-159	J
B-160	J
B-161	J
B-162	J
B-163	J
B-164	J
B-165	J
B-166	J
B-167	J
B-168	K
B-169	J
B-170	J
B-171	J
B-172	J
B-173	J
B-174	J
B-175	K
B-176	K
B-177	J
B-178	J
B-179	J
B-180	K
B-181	J
B-182	K
B-183	K
B-184	J
B-184.1/ B-184.2	K
B-185	K
B-186	K
B-187	J
C-1	J
C-2	J
C-3	K
C-4	J
C-5	J
C-6	J
C-7	K
C-8	K
C-9	K
C-10	K
C-11	K
C-12	K
C-13	J
C-14	J
C-15	J
C-16	J
C-17	J
C-18	J
C-19	J
C-20	J
C-21	J

PAGE	REV
C-22	J
C-23	J
C-24	J
C-25	J
C-26	J
C-27	J
C-28	J
D-1	K
D-2	K
D-3	J
D-4	K
D-5	J
D-6	K
D-7	J
D-8	J
D-9	J
D-10	K
D-11	J
D-12	J
D-13	J
D-14	J
D-15	J
D-16	J
D-17	K
D-18	K
D-19	J
D-20	K
D-21	J
D-22	J
D-23	J
D-24	J
D-25	J
D-26	J
E-1	J
E-2	J
E-3	K
E-4	K
E-5	J
E-6	J
E-7	J
E-8	K
E-9	K
E-10	K
E-11	K
E-12	K
E-13	K
E-14	K
E-15	K
E-16	K
F-1	J
F-2	J
Index-1	K
Index-2	J
Index-3	K
Index-4	K
Index-5	J
Index-6	J
Index-7	K
Index-8	K
Index-9	K
Index-10	J
Index-11	J
Index-12	J
Comment Sheet	K
Back Cover	-

PAGE	REV

PREFACE

This manual describes the NOS/BE Version 1.4 Operating System for the CDC® CYBER 170 Series, CDC CYBER 70, Models 71, 72, 73, 74, and CDC 6000 Series computers. It is written for systems programmers who perform system evaluation or program modification.

The manual describes the system interface with the central processor and peripheral processors, files and file tables, input/output, job processing, permanent file manipulation, and various system utilities. Appendixes B through E contain system tables and file formats divided into four general areas: central memory, job control point, disk and files, and extended core storage. In general, the central memory tables, extended core storage tables, disk tables, and file formats are of interest only to system programmers. The job control point tables are of interest to all users of the product set. Job control point tables can be used by central processor programs running at any control point. The tables in the appendixes serve as reference material for those familiar with the system and its product set. More detailed information is available in the various reference manuals and internal maintenance specifications.

RELATED PUBLICATIONS

The following manuals contain additional information that may be useful to a systems programmer. NOS/BE Manual Abstracts is an instant-sized manual that contains a brief description of the contents and intended audience of every manual documenting NOS/BE and its product set. The abstracts manual may be useful in determining which manuals would be of greatest interest to a particular user.

Control Data also publishes a Software Release History Report of all software manuals and revision packets it has issued. This history lists the revision level of a particular manual that corresponds to the level of software installed at the site.

Control Data Publication	Publication Number
NOS/BE Manual Abstracts	84000470
NOS/BE Version 1 Installation Handbook	60494300
NOS/BE Version 1 Reference Manual	60493800
NOS/BE Version 1 Operator's Guide	60493900
NOS/BE Version 1 Diagnostic Handbook	60494400
NOS/BE Version 1 Diagnostic Index	60456490
INTERCOM Version 5 Reference Manual	60455010
INTERCOM Version 4 Reference Manual	60494600
EXPORT High Speed Reference Manual	60456880
SCOPE Version 2 Operator's Guide	60455090

Control Data Publication	Publication Number
Update Reference Manual	60449900
CYBER Record Manager Basic Access Methods Version 1.5 Reference Manual	60495700
CYBER Record Manager Advanced Access Methods Version 2 Reference Manual	60499300
CYBER Loader Reference Manual	60429800

The NOS/BE to NOS/BE link is described in the NOS/BE Version 1 Operator's Guide. The NOS/BE to SCOPE 2 link is described in the SCOPE Version 2 Operator's Guide.

The NOS/BE Internal Maintenance Specifications are available on listable magnetic tape.

Extended memory for CYBER 170 Models 171, 172, 173, 174, 175, 720, 730, 750, and 760 is extended core storage (ECS). Extended memory for CYBER 170 Model 176 is analogous to CYBER 70 Model 76 large central memory (LCM) or large central memory extended (LCME). ECS and LCM/LCME are functionally equivalent, except LCM/LCME cannot link mainframes and does not have a distributive data path (DDP) capability. (An appendix in the NOS/BE Reference Manual describes other minor differences.) In this manual, the acronym ECS refers to all forms of extended memory on the CYBER 170 Series. However, in the context of a multimainframe environment or DDP access, model 176 is excluded. The acronym LCM refers to both LCM and LCME in the discussion of 819 disk I/O processing in section 5, because ECS cannot be used on a model 176.

Unless otherwise indicated, bit and byte numbers are given in decimal; word addresses, field and table lengths, and block and page sizes are given in octal. Unless reserved for a specific purpose or group, all currently unused fields, names, codes, and so on are reserved for future development.

DISCLAIMER

This product is intended for use only as described in this document. Control Data cannot be responsible for the proper functioning of undescribed features or parameters.

CONTENTS

1. INTRODUCTION 1-1

Hardware Characteristics 1-1
- Central Processor (CPU) 1-1
- Peripheral Processors (PPs) 1-1
- Central Memory (CM) 1-2
- Extended Core Storage (ECS) 1-2
  - Features 1-2
  - ECS Paging 1-3

Software Elements 1-3
- Files 1-3
- Control Points 1-3

System Organization 1-3
- System Loading 1-4
- System Tape 1-5

2. CENTRAL MEMORY AND THE CENTRAL PROCESSOR 2-1

CM Organization 2-1
Control Points 2-1
- Job Descriptor Number 2-2
- Storage Moves 2-3

CP - System Communication 2-4
CP - PP Communication 2-4
Program Recall 2-5
Central Memory Resident (CMR) 2-7
- Summary of CMR Areas 2-10
- Summary of Tables in Upper Table Area 2-10

STF - System Table Find 2-13
CMR Segmentation for ECS Systems 2-15
- Segment Loading 2-16
  - Segment Linkage 2-16
  - Segment Definition 2-16
  - Parameter Word 2-17
- ECS System Image 2-18
- ECS Error Recovery 2-20
- ELM - Error Log Messages 2-20.1

System Control Point 2-21
- Managing Subsystem Resources 2-21
- CALLSS Macro 2-22
- System Control Point Interfaces 2-23
  - Requesting Active Status 2-24
  - Subsystem Request Acknowledgement 2-25
  - Special Subsystem Requests to the Operating System 2-26
- End Processing for UCPs 2-35
- Normal SCP Termination 2-36
- Abnormal SCP Termination 2-36
- How to Define a Subsystem 2-37
- SETMFL Macro 2-37
- Programming Tips 2-38

3. PERIPHERAL PROCESSORS 3-1

Peripheral Processor Organization 3-1
PP Communications 3-5
PP Resident 3-6
- R.IDLE - PP Resident Idle Group 3-6
- R.OVLJ - Primary Overlay (Transient Program) Loader 3-6
- R.RAFL - Request Control Point Field Length Access 3-6
- R.TAFL - Terminate Control Point Field Access 3-6
- R.TFL - Test Field Length 3-8
- R.MTR - Process Monitor Function 3-8
- R.WAIT - PP Wait Loop 3-8
- R.RCH - Request Channel 3-8
- R.DCH - Drop Channel 3-9
- R. STBMSK 3-9
- R.STB - Store Byte 3-9
- R.OVL - Overlay Loader 3-10
- R.EREQS - Enter Stack Request 3-10
- R.DFM - Enter Dayfile Message 3-10
- R.READP - Transmit Data Via Channel from Stack Processor 3-11
- R.WRITEP - Transmit Data Via Channel to Stack Processor 3-11
- R.RWP - Performs Read/Write Logic for R.READP/R.WRITEP 3-11
- Field Access Flag Usage 3-12

System Monitor 3-12
- MTR Structure 3-12
- CPMTR Organization 3-13
- Operations 3-14
- CPU Scheduling 3-14
- Assignment of the PPs 3-15
- Channel Reservations 3-18
- Time Accounting 3-18
- Storage Requests 3-18

Monitor Functions 3-18
- M.ABORT - Abort Control Point and Drop PP 3-19
- M.BUFPTR - Watch Buffer Pointer Word 3-20
- M.CCPA - Change Control Point Assignment 3-20

M.CLRST - Clear Status 3-20
M.CPJ - Capture Peripheral Job 3-20
M.CPUST - Change CPU Status 3-21
M.DCP - Drop Central Processor Job 3-21
M.DFM - Process Dayfile Message 3-21
M.DPP - Drop PP 3-22
M.EES - Enter Event Stack 3-22
M.EESD - Enter Event Stack and Drop PP 3-23
M.ICE - Initiate Central Executive 3-23
M.ISP - Initiate Stack Processor 3-23
M.KILL - Bad Function Request 3-23
M.MFLA - Monitor Field Length Access at Control Point 3-24
M.NOTE - Null Function 3-24
M.NTIME - Enter New Time Limit 3-24
M.PASS - PPMTR Ignores Function Request 3-24
M.PATCH - Insert a Patch in PPMTR 3-24
M.PPLIB - PP Library Search Function 3-25
M.RACT - Request Control Point Activity 3-25
M.RBTSTO - Request Bit Storage 3-26
M.RCH - Request Channel Reservation 3-26
M.RCLCP - Recall Central Program 3-26.1
M.RCP - Request Central Processor 3-26.1
M.RPJ - Request Peripheral Job 3-27
M.RPJD - Request Peripheral Job and Drop PP 3-27
M.RSTOR - Request Storage 3-27
M.SCB - System Circular Buffer Surveillance 3-28
M.SCH - Initiate Integrated Scheduler 3-28
M.SEF - Set Error Flag 3-28
M.SEQ - Assign Job Sequence Number 3-29
M.SETST - Set Status Bits 3-29
M.SLICE - Terminate Time Slice Period 3-30
M.SLPER - Initiate Central Monitor in Other CPU 3-30
M.SPM - SPM Call from 1SP 3-30
M.SPRCL - Stack Processor Recall 3-31
M.TRACE - Enter Monitor Trace Mode 3-32
M.TSR - Terminate Storage Request 3-32

4. FILES AND FILE TABLES 4-1

Files 4-1
System Files 4-1
Permanent Files 4-2
Local Files 4-2
Queue Files 4-2
Input Queue 4-3
Output Queue 4-3
ROUTE Macro - Additional Capabilities 4-3
ACQUIRE Macro 4-5
VERIFYJ Macro 4-16
I/O Tables 4-18
File Tables 4-18
File Environment Table (FET) 4-19
File Name Table (FNT) 4-23
Device Tables 4-23
Equipment Status Table (EST) 4-23
Dismountable Device Table (DDT) 4-23
Mounted Set Table (MST) 4-24
Device Status Table (DST) 4-24
Device Activity Table (DAT) 4-24
Channel Status Table (CST) 4-24
Tapes Staging Table (STG) 4-25
Tape Drive Scheduling 4-26
Automatic Tape Drive Assignment 4-26
Tape Job Prescheduling 4-26
Job Scheduling with Tape Drive Overcommitment 4-27
Dynamic Tape Drive Status Checking 4-27
RMS Set Terminology 4-28
Device Sets 4-28
Public Device Sets 4-29
Private Device Set 4-29
Shared Device Sets 4-30
RMS Tables 4-31
Record Block Reservation Table (RBR) 4-32
Record Block Table (RBT) 4-32
INTERCOM Tables 4-38

5. INPUT/OUTPUT 5-1

I/O Philosophy 5-1
CIO 5-1
CIO Codes 5-1
Circular Buffer 5-2

CIO Operation 5-4
Allocatable Device I/O 5-18
Stack Processor 5-21
Dismountable Pack Processing - I/O Detail 5-34
ECS-Buffered I/O 5-39
819 Disk I/O Processing 5-41
Logical I/O Processing 5-41
General Description 5-41
Logical I/O Segments 5-44
Physical I/O Processing 5-45
General Description 5-45
Physical I/O Segments 5-46
PPIO Processing 5-48
LCM Buffer Management 5-50
Tables 5-50
Transfer Buffer Table 5-50
TBT Address Table 5-50.1
Unit Queue Table 5-51
Channel Table 5-51
CE Error File 5-51
819 Subsystem Flush Function 5-54.1

6. PERMANENT FILES 6-1

Permanent Files - System Interface 6-1
Permanent File Interlocks 6-1
Permanent File Tables 6-2
Permanent File Accounting 6-4
Permanent File Utility Routines 6-5
DUMPF Utility 6-6
PFLOG Utility 6-8
Private Device Set Processing 6-10
LABELMS Control Statement 6-10
ADDSET Control Statement 6-10
MOUNT Control Statement 6-11
RELABEL Control Statement 6-11
RECOVER Control Statement 6-12
Addressing Public Sets by Set Attribute 6-15

7. JOB PROCESSING 7-1

Job Flow 7-1
Job Input Queue 7-1
Tape Job Scheduling 7-1
Loading Jobs from Tape 7-2
Sequencer Jobs 7-2
JANUS 7-3
Integrated Scheduler 7-4
Job Scheduling 7-6
Rolling 7-6
Swapping 7-6
Job Control Area 7-8
Job Descriptor Table (JDT) 7-9
Job Scheduling Queues 7-9
Central Memory Queues 7-9
Device Queue 7-10
Permanent File Queue 7-10
Permanent Pack Queue 7-10
Operator Action Queue 7-11
INTERCOM Queue 7-11
Job Advancing 7-11
Control Statement Processing 7-13
Job Control Statement Source File 7-14
Job Termination 7-16
Normal Termination 7-16
Permanent Files 7-17
Local Files 7-17
Input File 7-17
Output File 7-17
Dayfile 7-17
Abnormal Termination 7-17
Job Post-Processing Utilities 7-18
List-of-Files Address - GETLOF and SETLOF Macros 7-18
Job Control With Logical Identifiers 7-19
CEVAL Macro 7-20
Access to Loader Word - GETLC and SETLC Macros 7-25

8. EDITLIB 8-1

Introduction 8-1
Character Set 8-1
Syntax and Semantics 8-3
System EDITLIB 8-5
EDITLIB Files 8-5
Special Handling of Local File Name System 8-7
Control Statement 8-7
Directives 8-10
Optional Directive Parameters 8-10
Directive Restrictions 8-11
Directive Formats 8-12
Library Directory Access 8-18
PP Routines 8-18
CP Routines 8-18
EDITLIB 8-18
Library Directory Format 8-18
PP Library Pointer Format 8-19
Files 8-19
System Files 8-19
User Files 8-19
File and Library Positioning 8-19
System Security 8-20
MDI (Move System Directory) 8-21
Directory/Library/Program Limits 8-22
Table Formats 8-22
Examples 8-30

9. SYSTEM BULLETIN UTILITY 9-1

BULLUP Statement 9-1
BULLUP Data Statements 9-1
Name Statements 9-2
Contents Statements 9-2
Processing Data Statements 9-2
Creating a System Bulletin File 9-3
Updating a System Bulletin File 9-3
Reducing a System Bulletin File 9-3
NOS/BE-INTERCOM Considerations 9-4

10. LDCMR 10-1

Introduction 10-1
LDCMR Control Statement 10-1
LDCMR Files 10-3
System Security 10-4
Reserved Names 10-4
LDCMR Interlock 10-5
LDC 10-5
Examples 10-5

11. SYSTEM DUMPS 11-1

Deadstart Dump Analyzer 11-1
DSDUMP Control Statement 11-1
Analysis Directives 11-2
Display Directives 11-2
Dump Directives 11-2
Special Dump Directives 11-5
Default Directives 11-6
DSDUMP Messages 11-6
System Dynamic Dump 11-7
Interface 11-7
Dump File Format 11-7
Listing Dump Files 11-7
Error Messages 11-10

12. DOWNLINE LOAD UTILITY 12-1

BCPROC 12-1
BCLOAD 12-2
DLEB 12-5
WPPF 12-5
ANSWER Buffer 12-6
DLL 12-6
Building a Controlware File 12-7
Examples 12-7

APPENDIXES

A. STANDARD CHARACTER SETS A-1
B. CENTRAL MEMORY RESIDENT TABLES B-1
C. JOB CONTROL POINT TABLES C-1
D. DISK TABLES AND FILE FORMATS D-1
E. EXTENDED CORE STORAGE TABLES E-1
F. SYMBOL DEFINITION F-1

INDEX

FIGURES

2-1	Allocation of CM	2-1
2-2	Sample Control Point Storage	2-2
2-3	Call Formats	2-6
2-4	Typical CMR Assignments	2-7
2-5	ECS System Areas	2-15
2-6	ECS System Image	2-19
2-7	ECS Segment	2-20
3-1	Pool PP Layout	3-4
3-2	PP Input Register	3-5
3-3	PP Resident Routines	3-7
3-4	PP Chain	3-17
3-5	PP Job Queue	3-19
4-1	System Dayfile Area	4-21
4-2	System File Entries	4-22
4-3	Nonallocatable Device File Processing	4-25
4-4	RMS Tables	4-31
4-5	Record Block Table	4-33
4-6	File Table Interfaces - FNT Points to RBT Chain	4-34
4-7	File Table Interfaces - RBT Points to RBR Via DDT	4-36
4-8	File Table Interfaces - RB Byte Points to RB	4-37
5-1	Circular Buffer Interface	5-3
5-2	FET - Circular Buffer Interface	5-3
5-3	Device Set I/O Processing	5-35
5-4	Output Flow to RMS File	5-40
5-5	CYBER 176 Computer System with 819 Disk	5-41
5-6	Logical I/O Processing	5-42
5-7	Physical I/O Processing	5-46
5-8	PPIO Processing	5-49
5-9	CE Error File	5-51
6-1	APF Table Entry	6-4
6-2	Permanent File Dump Tape Format	6-6
6-3	File Header	6-7
6-4	PFLOG Dump Tape Format	6-8
7-1	JANUS Interfaces	7-4
7-2	Integrated Scheduler Interfaces	7-5
7-3	Scheduler Request Stack in System Exchange Package Area	7-7
7-4	Control Statement Processing Flowchart	7-12
8-1	Basic Usage of System EDITLIB	8-2
8-2	Library Table Interfaces	8-29

TABLES

2-1	Example of Storage Moves	2-3
2-2	SFCALL Return Codes	2-28
3-1	PP Direct Cell Assignment	3-1
4-1	ACQUIRE Macro Parameters	4-10
4-2	Default RB Size	4-30
5-1	READ Macro Logical Sequence	5-6
5-2	READN Macro Logical Sequence	5-8
5-3	READSKP Macro Logical Sequence	5-9
5-4	RPHR Macro Logical Sequence	5-10
5-5	WRITE Macro Logical Sequence	5-11
5-6	WRITER Macro Logical Sequence	5-12
5-7	WRITEF Macro Logical Sequence	5-13
5-8	WRITEN Macro Logical Sequence	5-15
5-9	WPHR Macro Logical Sequence	5-16
5-10	SKIPF Macro Logical Sequence	5-16
5-11	SKIPB Macro and BKSP Macro Logical Sequence	5-17
5-12	BKSPRU Macro Logical Sequence	5-17
5-13	Stack Processor Orders	5-26
5-14	Ranges of Cylinders Used	5-32.1
6-1	Header Fields	6-7
8-1	Directive Interpretation and Execution	8-8

INTRODUCTION 1

The Batch Environment Network Operating System (NOS/BE) is an operating system for the CYBER 170 Series, the CYBER 70, Models 71, 72, 73, and 74, and the 6000 Series computers. NOS/BE accepts input in the form of jobs submitted by users, processes jobs as directed by the accompanying job control statements, and provides operations control in accordance with command instructions that are input at the console's keyboard. This section describes the inherent hardware characteristics, the basic software elements, and how they work together to accomplish efficient processing of users' jobs.

HARDWARE CHARACTERISTICS

The operating system uses peripheral processor units (PPs) for system and input/output (I/O) tasks and a central processor unit (CPU) to execute user and system jobs. Central memory (CM) contains the user programs. System software areas are located at the upper and lower ends of CM. An extended core storage (ECS) unit may contain system libraries and other items (such as file buffers for rotating mass storage and swap files), which may not be contained in CM or on other mass storage devices.

CENTRAL PROCESSOR (CPU)

The CPU performs tasks of a computational nature. It has no I/O capability. It communicates with other system components through CM. The CPU is used almost exclusively for program compilations, assemblies, and executions. The CPU makes system requests through a CPU request word located at the reference address plus one (RA+1) of the current program in execution. The CPU is discussed in section 2.

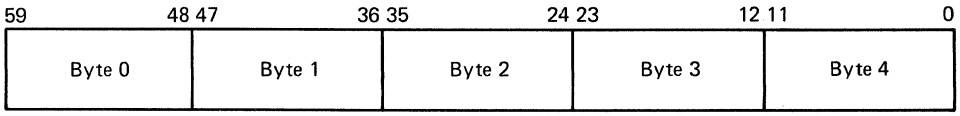
PERIPHERAL PROCESSORS (PPs)

The PPs, of which there may be up to 20 (identified as PP0,PP1,...,PPn), are identical. They perform many tasks for requesting programs in CM. A PP can be assigned to control, I/O, job scheduling, control statement interpreting, and other tasks as required. Tasks are assigned one at a time to each PP by the system monitor (MTR). When an assigned task is completed, the PP signals the system. MTR waits for this signal before assigning another task to the PP.

Each PP is assigned a block of eight words in the system area of CM through which communications with the system are conducted. Each block contains an input register, an output register, and a message buffer. PPs are discussed in section 3.

CENTRAL MEMORY (CM)

CM words are 60 bits long. Each has five 12-bit PP memory words called bytes. Each 12-bit byte in a CM word is numbered 0 through 4, from left to right as follows:



One or more user programs may be in a state of execution concurrently. These programs are stored in CM in an assigned user area. A set of system components necessary for the operation of the system is also stored in CM, forming the central memory resident (CMR) and the record block table (RBT) areas. Central memory is accessible by all PPs and CPUs; it forms the communications link between all processor units in the computer system.

Central memory resident (CMR) contains system communications areas, system tables, CPU resident routines, the library directory, and information about each job currently in execution. The CMR is discussed in section 2, and the RBT is discussed in section 4.

EXTENDED CORE STORAGE (ECS)

Features

ECS is divided into the system area, the dynamic area, and the direct access area. These areas function as follows:

Area	Function
System	Contains system pointers and tables required by ECS software.
Dynamic	Paged area; contains buffers, library programs, swap files, and other files assigned to ECS.
Direct access	Assigned to user as result of job statement request; used and managed by the user; contains the ECS segment library in ECS control point 0 field length.

Such division of ECS allows the following features to be provided.

- I/O buffering through ECS (via CM buffers or distributive data path).
- Library residence in ECS.
- Job swapping to/from ECS by the integrated scheduler.
- Compatibility with BNL-ECS.
- Segmentation of system central processor code.

ECS Paging

I/O buffering through ECS is made possible by the paging of ECS. The basic element of ECS paging is a PRU made up of 100_8 CM words. A group of eight consecutive PRUs (1000_8 CM words) forms a page. A deadstart installation parameter defines the buffer size. The parameter IP.EBUF determines the default buffer size for deadstart processing and has a default value of 16 decimal pages.

ECS paging provides the following advantages:

- More efficient use of ECS through dynamic allocation/release of space.
- Availability of ECS to more users by allowing the use of an over-commitment algorithm for ECS.

SOFTWARE ELEMENTS

Files and control points are basic to the operating system.

FILES

A file is an organized collection of data known to the system by a given name. Data is organized in one or more logical records and is terminated by an end-of-file indicator. The jobs processed by the operating system and all intermediate and final results are contained in files or parts of files. Files are discussed in section 4.

CONTROL POINTS

The system can control execution of several jobs at one time. When placed into CM before execution, each job is assigned a value, which is the control point number and the index to a control point. Jobs at control points are assigned to a processor for execution. Each control point has a control point area in the CMR, which holds all information necessary to process the assigned job. Control points are discussed in section 2.

SYSTEM ORGANIZATION

The operating system consists of PP programs, CP programs, macro definitions, and symbol definitions. The entire system is contained on program library files produced by the library maintenance program Update. Programs on these library files are in source language form. Installation options are provided to permit flexible selection of system features during extraction of programs and texts from the libraries. These software components can be assembled for subsequent creation of a deadstart file on tape by the system maintenance program EDITLIB. EDITLIB is discussed in section 8.

A system monitor controls the operating system. The system monitor consists of PP overlay MTR (operates in PP0) and CPMTR. In a disk system, CPMTR is assembled as part of CMR; in an ECS system, it consists of a number of separate segments.

SYSTEM LOADING

To load the operating system, the deadstart tape or disk is mounted on the appropriate unit, and a small bootstrap loader program is set up on the hardware deadstart panel switches. When the deadstart button on the operator's console or deadstart panel is pressed, the bootstrap program is transferred to and executed in PP0. The bootstrap loader reads the first record on the deadstart file and executes the routine contained there. This routine reads in the remainder of the common test and initialization (CTI) routines. The CTI routines determine the machine attributes and run a confidence test to ensure that the mainframe is operating properly. When confidence testing is complete, the first operating system routine (OSB) is read into PP0. It reads the deadstart control program (CED) from the next record into PP0 and sets it into operation. CED determines the type of deadstart to be performed and loads the required routines into all PPs involved in the deadstart process. The routines include a display routine in PP0 and I/O routines in PP1, PP2, and PP3. CMR is read from the deadstart file into CM, and a display shows all deadstart functions and options that may be selected. The functions include the following:

- Library reload option.
- Queue file recovery level.
- User device set processing.
- Equipment configuration changes to the system.
- Initialization level of ECS.
- First mainframe to deadstart.

The operator may select specific options or take the default option for each function.

After deadstart options are processed, control is passed to MTR and the display routine, DSD. If a deadstart tape was used rather than disk, the tape is rewound to its load point and is not referenced again during normal system operation. The tape can be removed and the tape unit can be cleared for other operations. At this point, the system can process jobs (refer to section 7).

Upon completion of system loading, the computer contains the following:

- Initial system libraries stored on one or more mass storage devices. Programs can be loaded from any system library into PPs or CM as needed.
- CMR loaded into the low end of CM. A set of tables in CMR contains information about the system. Some tables are used by the system PP programs to communicate with each other. An RBT is built into the upper end of CM.
- Some system programs stored in CM in an area immediately following CMR. Such programs can be loaded into PPs or into other CM areas much faster than they can be loaded from the system storage device. Storage space in CM is costly, and storage space for library programs cannot be used to run user programs. Therefore, only the most frequently used library programs are stored. For the same reason, CMR is kept as small as possible.
- The system monitor program (MTR in PP0 and CPMTR in CMR) which controls the system. It controls allocation of system physical resources (CM, ECS, channels, equipments, PPs, and CPUs), handles all communications between user programs and the system, and coordinates activities of the other PPs.

- PP1 which contains the operator console display driver program DSD. DSD provides a communication path between the system and the operator. Current system status is displayed by DSD on the operator console display(s). The operator can control system operation by typing commands on the console keyboard.

- Remaining PPs which contain pointers to a PP communications area, an area in CMR used for communications between each PP and MTR, and a PP resident program loaded into each PP. The resident program is responsible for reading the input register, loading PP overlay programs into its assigned PP, and providing communications between the overlays and MTR.

SYSTEM TAPE

The released system deadstart tape consists of the following programs.

Name	Description
IPL-ZZZ	CTI routines.
OSB	Operating system bootstrap routine; resides in PP0.
CED-CEF	Deadstart PP control programs; reside in PP0.
DDR	Read driver for 844 or 885 disk subsystem; resides in PP1.
MDR	Read driver for 66x and 67x tape drives; resides in PP1 and PP3.
CMR	Central memory resident (up to 64 copies).
COM	Deadstart option matrix generator; resides in PP2.
IRP	Deadstart RMS driver control program; resides in PP2.
5CY	Deadstart driver for 844-21, 844-41, and 885 disk subsystems.
LFP	Driver for 819 disk subsystem (CYBER 176 only).
0SY	844 buffer controlware for the 7054 controller.
0SZ	844 buffer controlware for the 7154 controller.
0SJ	885 buffer controlware for the 7155 controller.
0MT	66x magnetic tape subsystem controlware.
IRCP	Deadstart main CP program.
STL	Deadstart PP initiation program (PP resident).
MTR	System monitor program.
DSD	Display control program.
Directory	Library name table. PP name table. PP programs; the first must be stack processor's segment.

Name	Description
System Libraries	Entry point table.
	External reference list.
	External reference table.
	Program number table.
	Program name table.
	CP routines.

An installation may expand the preceding records by placing up to 63 additional CMR records on the system tape for different equipment configurations.

CENTRAL MEMORY AND THE CENTRAL PROCESSOR 2

CM ORGANIZATION

Figure 2-1 shows the allocation of CM.

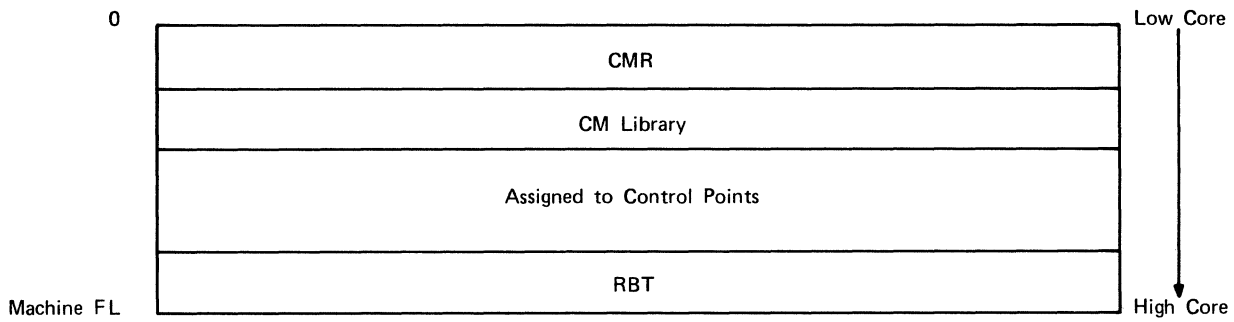


Figure 2-1. Allocation of CM

Low core is allocated to the central memory resident (CMR) portion of the operating system, executable system programs, and INTERCOM buffers. The length of the INTERCOM buffers area varies dynamically when INTERCOM is running. High core is allocated to the record block table (RBT). Its length varies dynamically with the load of the system. The remaining area can be assigned to control points.

CONTROL POINTS

Blocks of CM storage not allocated for system use are ordered by control point number and assigned to jobs. Each control point has a corresponding table in CMR called the control point area. A control point is not a physical entity but rather a concept used to facilitate bookkeeping. The control point number and the control point area, however, are physical quantities that appear in the system.

Any number of control points up to 15 decimal are possible. In the released system, the default value of N.CP is 15. In an installation with n control points for user jobs, they are numbered from 1 to n. Only one job can be assigned to a control point at any time. Once a job is assigned to a control point, system resources such as CM, ECS, channels, equipments, and processors may be assigned to the control point for use by the job.

Storage assigned to a single control point is contiguous; storage for all control points is not necessarily contiguous. The storage block assigned to the job at control point 2 is higher than the block for the job at control point 1, storage for control point 3 is always higher than that for control point 2, and so on.

In figure 2-2, no storage is assigned to control points 3 and 5; unassigned storage appears between assigned storage.

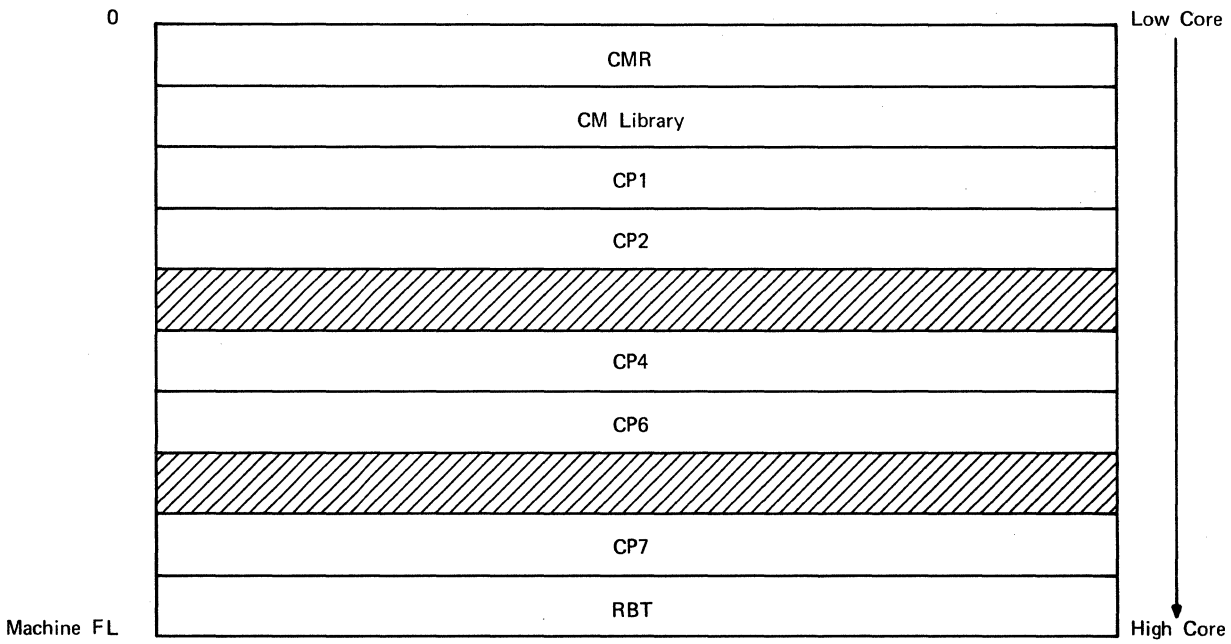


Figure 2-2. Sample Control Point Storage

In addition to the n control points used for running jobs, two pseudo control points (zero and n+1) are used by the system.

Control point 0 identifies system resources not allocated to a job at a control point. They are unallocated or allocated to the system. If an equipment is assigned to a control point, that control point number is entered into the system table entry for that equipment. If not assigned to a job at a control point, the equipment is assigned to control point 0 and is available to be assigned to a job. All active system files are attached to control point 0. They include the system file, any job files that have been read in and are waiting for scheduling, and all output files waiting to be processed by JANUS and remote batch processors.

Control point n+1 is used by CPMTR for executing system jobs such as the integrated scheduler or storage move routines. Control point n+1 has an abbreviated control point area that consists primarily of an exchange package. The field length of control point n+1 is all available memory.

JOB DESCRIPTOR NUMBER

During execution, a job might not remain continuously at the same control point. It is possible for the job to be swapped out while it is only partially executed. When a job is swapped out, it is not associated with a control point. When a job is swapped in, it may be associated with a different control point.

While a job is swapped out, the only table in CMR that contains information about the job is the job descriptor table (JDT). When a job is initialized at a control point, it is also assigned to an entry in the JDT. The job descriptor number is constant and identifies the job during its entire execution.

To clarify the difference between job descriptor number and control point number, JDT numbers start at n+1 (n is the number of control points).

STORAGE MOVES

CM storage must be reallocated and jobs must be moved as jobs finish processing and new jobs begin or as jobs are swapped in and out. If a job at a control point requests additional storage, it may be necessary to move jobs to obtain the required storage. CPMTR keeps a tally of unassigned CM in CMR word T.UAS.

Storage associated with each control point is allocated or unallocated. Either storage may have a zero value. Allocated storage is defined by the reference address (RA) and field length (FL) of the control point. Unallocated storage (UAS) lies between the allocated portions of two consecutive control points. This area is associated with the lower of the two control points, but it may be transferred to neighboring control points by moving any intervening allocated storage.

A request for a reduced field length transfers storage to UAS (no storage moved). A request for an increased field length, when the total already associated with the control point is adequate, results in a transfer of unallocated storage to allocated storage; no storage move takes place.

If it is necessary to take unallocated storage from other control points to satisfy a request for increased field length, control points above and below the requesting control point are scanned. This scan locates the combination of unallocated storage blocks that result in a move of the least amount of storage.

If control point 1 in figure 2-2 needs more storage, it is necessary to move control point 2. If control point 6 needs storage, sufficient unallocated storage may be available to make a control point move unnecessary. If, however, control point 7 needs additional storage, control points 4, 6, and 7 are moved downward to provide the storage. Added storage always extends the field length upward.

Example:

Control point 5 requests an FL of 300 (refer to table 2-1). All values are increments of 100 octal. If CPMTR takes the UAS from control point 7, the 150 units of CM at control point 6 must be moved. However, taking UAS from control points 3 and 4 requires moving 120 units of CM at control points 4 and 5 (20 units are moved from 4 to 3; 100 units are moved from 5 and added onto the 20 units moved to 3).

TABLE 2-1. EXAMPLE OF STORAGE MOVES

Control Point	Before			After		
	RA	FL	Unallocated Storage (UAS)	RA	FL	UAS
0	0	142	0	0	142	0
1	142	33	0	142	33	0
2	175	31	0	175	31	0
3	226	0	500	226	0	0
4	726	20	130	226	20	0
5	1076	100	0	246	300	430
6	1176	150	0	1176	150	0
7	1346	0	430	1346	0	430

CP — SYSTEM COMMUNICATION

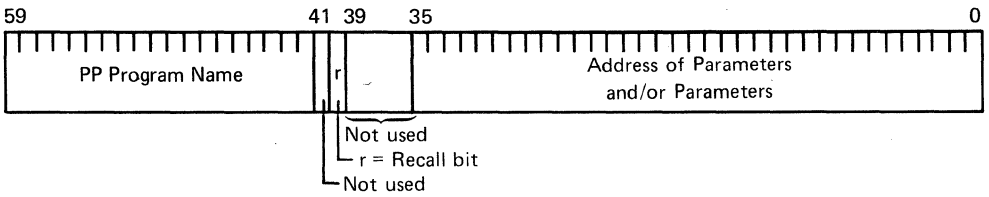
A running CP program must communicate with the system in the following situations.

- When a CP program is loaded and executed as a result of a control statement call. The system must place any parameters specified on the control statement in an area where they can be read by the CP program.
- When a CP program needs to perform input/output. No CP instructions allow a CP program to perform I/O. The CP program must send a request to the system to load a PP program to execute the I/O.
- When a CP program terminates. The program must advise the system that the system may process the next control statement.

Since a CP program cannot access memory locations outside its field length, any area reserved for communication between a CP program and the system must be within the field length of the job. The first 101_8 locations of each job's field length are reserved for this purpose. The following 10_8 words are reserved for the loader table. The first program loaded into a user field length is always loaded at location RA+111_8 (for the user, this is location 111_8 because the reserved words are RA+0 through RA+110_8).

The RA communication area is shown in appendix C.

The first word of a user field length (location RA+0) is reserved for use of hardware and software flags in event of error. Other locations in the first hundred octal words of a user field length store information needed for execution of a system program. Monitor regularly scans location RA+1, which is presumed to contain a request from the central processor for monitor to summon a peripheral program. The form of the request is as follows:



Loader information is placed by the first of several loader routines in words RA+64 through RA+67. This information is used and modified as additional loader routines complete specific tasks.

When parameters are encountered on a control statement, they are placed in locations RA+2 through RA+63 by IAJ, which stores the total number of parameters in location RA+64. When the routine or file indicated on the control statement executes, it finds the information needed to direct execution in these locations.

CP — PP COMMUNICATION

If a user's program places a call for a PP program in RA+1, CPMTR will pick up the RA+1 call, insert the control point number of the caller into bits 39 through 36 of the word, and clear bit 41. If the central exchange jump (CEJ) installation option is available, the user's program should use it immediately after placing a call in RA+1. This causes CPMTR to begin execution immediately. If CPMTR determines that the RA+1 call should be assigned to a PP, it passes the call on to MTR.

When a PP is available, MTR writes the word into its PP input register in CMR (refer to figure 3-2 for the format of a PP input register of a transient program called from a CP program). The name, the auto recall bit, and any parameters in bits 35 through 0 appear in the input register exactly as they did in RA+1. Parameters are passed from a CP program to a PP program through this parameter field.

For example, if the PP program CIO is called, CIO finds the relative address of the file environment table (FET) used in the operation by reading its input register. It can find the RA of the control point field length by reading the control point number from its input register, computing the address of the control point area, and reading the value of RA from the control point area. By adding the RA to the relative FET address, CIO obtains the absolute address of the start of the FET. CIO then reads the parameters for the I/O operation from the FET. In ECS systems, CPMTR traps all CIO calls and sends them to CPCIO where the device type is checked. If the device is RMS or ECS, the request is processed; otherwise, the request is sent to CIO.

MTR continually scans RA+1 in the event that the user's program does not use the central exchange jump, or the instruction is not available. When an RA+1 call is found, MTR initiates CPMTR. Less CPU time is used by letting CPMTR process the call than if MTR did it directly.

Bit 59 of RA+66 communicates to the user program if CEJ is available. If the hardware for this instruction is available, the bit is set.

PROGRAM RECALL

The recall program status enables efficient use of the CP and capitalizes on the multiprogramming capability of the operating system. Often, a CP program must wait for an I/O operation to be completed before more computation can be performed. To eliminate the CPU time wasted if the CP program is placed in a loop to await I/O completion, a CP program can request that the control point be put into recall status until a later time, and the CPU can be assigned to execute a program at another control point. The job may be rolled out or swapped out, as necessary.

Recall may be automatic or periodic. Auto recall should be used when a program requests I/O or other system action and cannot proceed until the request is completed. Control is not returned until the specific request has been satisfied. Periodic recall can be used when the program is waiting for one of several requests to be completed. The program is activated periodically so that it can determine which request has been satisfied and whether or not it can proceed.

To enter periodic recall, a CP program inserts the characters RCL left-justified into RA+1. Upon encountering the RCL request, CPMTR examines the auto recall bit (bit 40). If set, the request is considered to be an auto recall request. If it is not set, CPMTR checks bits 10 through 0 (decimal) for a delay count. The delay count is specified in units of 0.244 millisecond (the same as the real-time clock). The largest delay time that can be specified is 2047 (decimal) or approximately 0.5 second. If this delay count is not set, CPMTR specifies a default value for it. The current default, as defined by symbol RCLPER in the CMR internal configuration parameters, is 25 milliseconds. The delay count of the control point in periodic recall is examined regularly by the advance control point routine (ACP) of MTR. When the delay count expires, the control point loses its recall status, and the CPU is again assigned to execute the program at the control point. At this time, the CP program can check the completion bit in the FET to see if the I/O is finished. If so, the CP program can proceed with computations. If I/O is not complete, the CP program can go into recall.

To enter auto recall, a CP program makes a request in RA+1 with bit 40 of RA+1 set to one. The control point is put into auto recall after the request has been initiated. The CPU is assigned to another control point. The program in recall is restarted by MTR after the completion bit in the FET has been set. MTR, not the user, checks the completion bit in the FET.

Recall and auto recall are often used while waiting for CIO to process an I/O request. However, any time a PP program is called from RA+1, with bit 40 of RA+1 set to one, the control point is put into auto recall. If bit 40 is set, bits 17 through 0 of RA+1 must contain the address of a word in the program's field length called a reply word. When the PP has completed its function, it sets the completion bit (low order bit) in the reply word. When the completion bit is set, MTR restarts the program.

For a call to CIO, the reply word is the first word of a FET. For other programs, the reply word need not be part of a FET.

Some PP programs (DMP and MSG) set the completion bit only when they are called with auto recall. Periodic recall cannot be used for these programs.

A CP program can go into auto recall without calling a PP program by putting RCL left-justified into RA+1 and setting bit 40 of RA+1 to one. Bits 17 through 0 of RA+1 must contain the address of a reply word. A program which has already initiated one or more I/O operations might go into auto recall in this way, using the first word of the FET associated with one of the I/O operations as the reply word. The formats of RA+1 for a normal CIO call, a request for periodic recall, a CIO call with auto recall bit set, and an RCL call with auto recall bit set are shown in figure 2-3. For periodic recall, a user must issue a normal CIO call followed by an RCL request. For auto recall, only one request is required.

Normally, CP programs use auto recall for convenience, but only one request involving auto recall can be processed at one time. For example, to initiate I/O action on several files at once, a user must employ the periodic recall technique. All the requests are issued without recall (using a separate FET for each request). Then the user requests periodic recall. Each time the CP program is restarted by the system, it can check all the files for completion and go back into periodic recall if any files are incomplete.

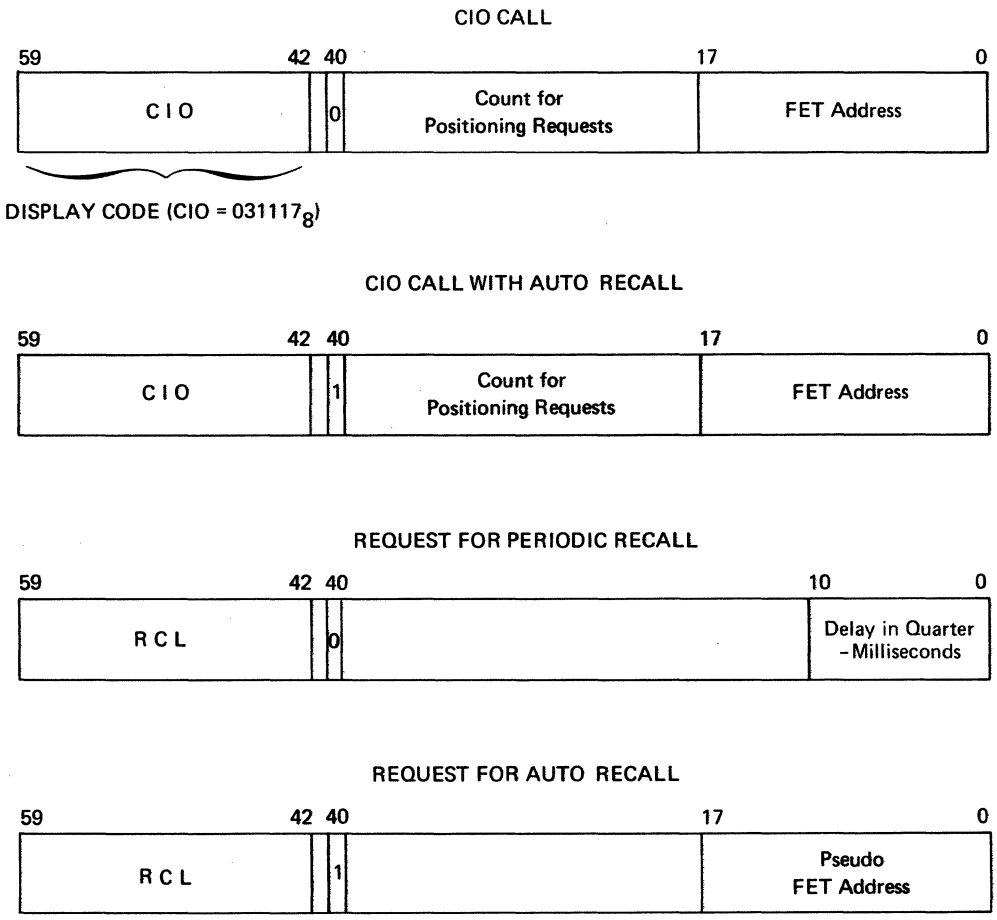


Figure 2-3. Call Formats

Periodic recall can also be used when a CP program can initiate an I/O request and perform computation. In some cases, the I/O would be completed before the computation; in others, the computation would complete first. The user would go into recall only after computation was completed and then only if the I/O was still in process.

Periodic recall should also be used, if possible, to continue processing while only part of the data buffer has been read or written by the I/O driver. Some of the I/O drivers coordinate with MTR so that a program in periodic recall is restarted after one or two PRUs have been processed.

CENTRAL MEMORY RESIDENT (CMR)

The low end of central memory is reserved for the CMR portion of the operating system and the system library portions which reside in CM. CMR contains pointers, tables, and programs. Its length depends upon several factors, including the number of PPs and the number of control points, which determine the number of tables and the length of certain tables in CMR. Some CMR tables are optional and appear only by installation parameters. Figure 2-4 illustrates a typical CMR.

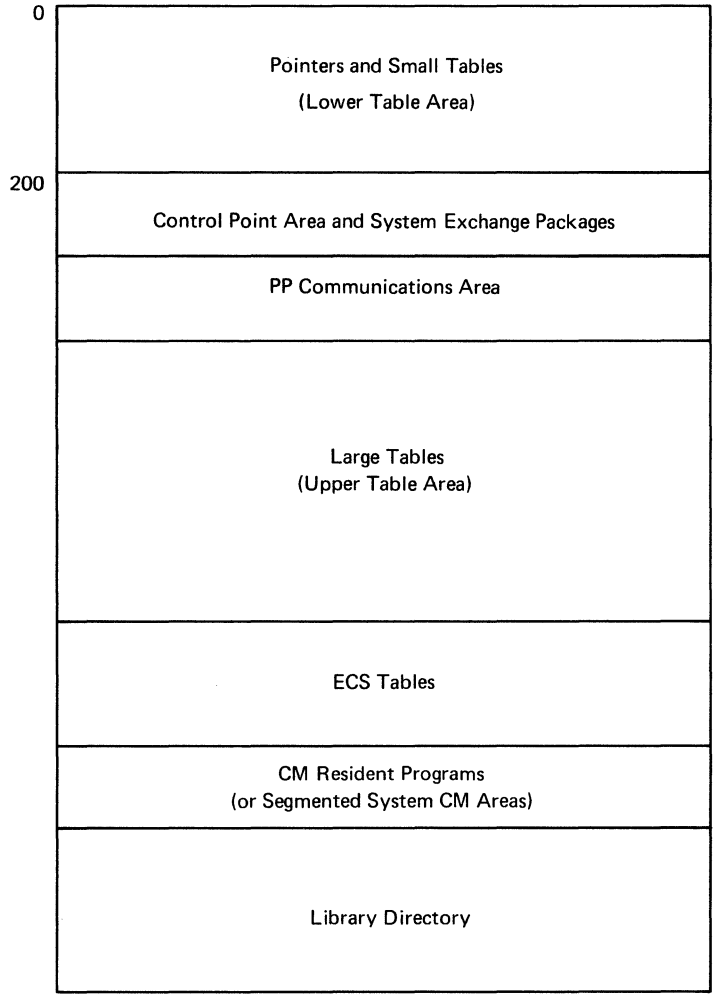


Figure 2-4. Typical CMR Assignments

The CMR contains the following tables.

First Word Address	Table Name	Description
0		CMR pointer area.
100	T.CST	Channel status table.
154	T.PPS1	PP status words.
200	T.CPA$_n$	Control point areas.
	T.XPIDLA	System job exchange package area.
	T.PPC1	PP communication areas.
	T.EST†	Equipment status table.
	T.FNT†	File name table. CIO-CPCIO special file name tables. Permanent file name tables. T.ELIBD - ECS resident library descriptor word
	T.ITABL†	INTERCOM table.
	T.DAT †	Device activity table.
	T.RMSBUF†	RMS buffer.
	T.STG†	Tapes staging table.
	T.APF	Attached permanent file table.
	T.EXPIO	CYBER 176 exchange package and I/O buffers.
	T.CHT	Channel table.
	T.UQT	Unit queue table.
	T.RQS††	Request stack.
	T.RST	Request scheduling table.
	T.RBR	Record block reservation table (headers).
	T.RBRBIT	RBR bit table.
	T.DST	Device status table.
	T.DOT	Device overflow table.
	T.SEQ	Sequencer table.
	T.INS	Installation area.
	T.MST	Mounted set table.
	T.DDT	Dismountable device table.

† Table must begin before 10000_8.
†† Table must begin before 20000_8.

First Word Address	Table Name	Description
	T.TRB	Trace buffer.
	T.VRNBUF	VSN buffer.
	T.TAPES	Tapes table.
	T.URT	Tape unit recovery table.
	T.MAIL	Scheduler mailbox buffer.
	T.IDT	Logical ID table.
	T.DFB	Dayfile buffers.
	T.PJT	Parameter storage for delayed PP jobs.
	T.MAB	Mainframe attribute block.
	T.SSCT	Subsystem control table.
	T.SCHPT	(Optional) scheduler performance table.
	T.SCHJCA	Scheduler job control area.
	T.SCHJDT	Scheduler job descriptor table.
	T.ELST	Error logging status table.
	T.PPOVL	PP resident overlay save buffer.
	T.BRKPT	Breakpoint table (ECS system).
	T.AREA	Area table (ECS system).
	T.ENTRY	Entry table (ECS system).
	T.BCFAP	CEFAP buffer.
	T.EPAGE	Empty page stack.
	T.ECSPRM	ECS parameters.
	T.SCBHDR	System circular buffer.
	T.SUBPG	Subpage buffer.
	CM resident programs (disk system).	
	Segmented system areas (ECS system).	
	T.LIB	Library directory.
	INTERCOM pointer area. INTERCOM buffer and user tables.	
	Job control point user field length.	
	T.RBT	RBT chains.

SUMMARY OF CMR AREAS

Area	Description
Lower table	Contains pointers to larger tables in the upper table area of CMR, various flags, constants, and installation options parameters. It includes accounting information, calendar and Julian dates, the system display label, and other small tables. The lower table area occupies the first 200 words of CM.
Control point	Contains a 200-word area for each control point in the system. Each area contains the job name, exchange package, and other information related to the job running at that control point. The system exchange package is also contained in the same area.
PP communications	Contains eight words for each PP in the system, through which they communicate with the system monitor and with each other. Each area contains the PP input and output registers and a six-word message buffer.
Upper table	Contains major tables pertinent to system and job operation.
ECS table	Contains buffers for transferring PP overlays and RMS files.
CMR program	Contains four resident programs: CP.MTR — Central processor monitor. CP.SM — Central processor storage move. CP.SPM — Central processor stack processor manager. CP.SCH — Central processor memory manager (scheduler). In a segmented system, the CMR program area is overlaid by the ECS system resident and the CP code overlay segments.
Library directory	Contains tables related to the system libraries, including library name table, PP program name table, and CMR library programs.

SUMMARY OF TABLES IN UPPER TABLE AREA

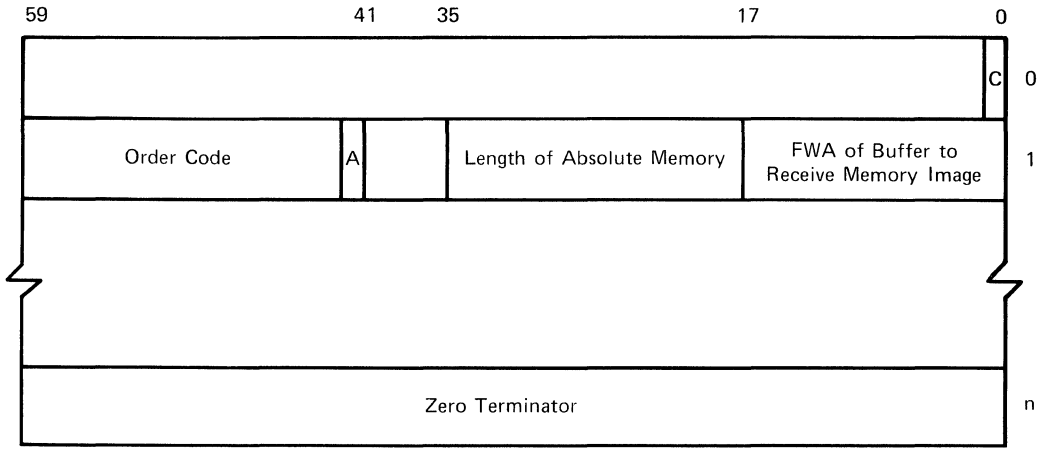
Table	Description
Equipment status	Contains one entry for each device in the system configuration. Nonallocatable devices can be assigned to one control point at a time; allocatable devices can be assigned to many control points simultaneously.
File name	Contains an entry for each file in the system; created when the file is created. Several entries are preset and remain in the system for duration; these entries are for the system library (deadstart) file, the system and control point dayfiles, and the hardware error file.

Table	Description
INTERCOM	Provides multiplexer and port definition information for INTERCOM program use.
Device activity	Contains a four-word entry for each RMS device in system. Each entry provides dynamic information related to current activity of the RMS device.
Rotating mass storage buffer	Holds a message to be flashed on the bottom line of the B display. The message reports an error on an RMS device and asks the operator to idle down the device.
Tape staging	Defines availability, assignment, and demand for tape devices.
Attached permanent file	Provides information for the permanent file manager and job use. Control and status information entries are created when a permanent file is cataloged initially or attached to a qualified CP program.
Request stack	Requests for data transfers, device positioning, or logical file operations. Each allocatable device in system has at least one three-word entry in this table when a request for its use is active.
Record block reservation	Provides continuous information as to assignment/availability of record blocks in which file data is recorded on allocatable devices. Strings of bits in the RBRBIT table denote current status of record blocks in each device.
Device status	Directly related to the request stack; contains a two-word entry for each allocatable device in the system, plus an additional pseudo entry for unassigned file processing.
Sequencer	Contains 30-word entry for each preallocated RMS device for use for CE diagnostic programs.
Installation	Reserved for specific needs of installation. Tables are generated in the area only by installation.
Mounted set	Contains one entry for each mounted device set, including the public sets.
Dismountable device set	Contains entries for each RMS device, plus entries for each queueing device needed by jobs.
Tapes	Contains one entry (10_8 words per entry) for each tape unit defined.
Tape unit recovery	Contains one entry (five words per entry) for each 66x/67x tape unit.
Mailbox	Used for communications between system and swapped out jobs.
ID	Contains host ID, logical IDs, and physical (link) IDs. The ID table can be zero-length.

Table	Description
Dayfile buffer area	Contains dayfile buffers and file environment table entries of the system dayfile, the control point dayfiles, and the hardware error file. The control point 0 buffer is at the end of this area.
Peripheral job	Contains parameters saved for delayed PP jobs.
Mainframe attribute block	Contains attributes of a mainframe, such as number of PPs/PPUs and the presence of ILR/SCR, CMU, CEJ/MEJ, and CPU-1/CPU-0.
Subsystem control	Contains names of defined subsystems.
Scheduler performance	Optional table used to collect execution data to study the efficiency of the integrated scheduler. Created by installation parameter IP.SPT set to 1.
Job control area	Contains entries pertinent to the scheduling of jobs by class and queue priorities.
Job descriptor	Contains linked entries for each class of job. Entries describe job requirements, current status, accumulated use time of system components, and so on (refer to section 7).
Error logging status	Optional table to control status/control register error reporting on CYBER 170 systems. Created by CMR configuration parameter L.ELST set to 20_8. Table can be zero length.
Breakpoint	Contains the breakpoint code exchange package, the breakpoint wait loop, flags and data used by DSD, and the breakpoint entries. It is used by DSD and breakpoint processing.
Area	Contains an eight-word buffer that receives the ECS area table from a segmented system. It describes the system ECS and CM structure.
Entry	Contains the date-time stamp for this CMR, the associated segment library name, and a list of the entry points defined in the tables with their addresses.
CM resident programs (in a disk system)	Symbol — Name/Function CP.MTR — Central processor monitor. CP.SM — Central memory storage move. CP.SPM — Stack processor manager. CP.SCH — Memory manager scheduler.
Library directory	Falls at the end of the CMR upper table area following the CM resident programs and ECS tables. It contains two-word entries in the program name table section and one-word entries in the entry point table. The directory length can expand or contract as programs are added and deleted or as program residence is changed.

STF — SYSTEM TABLE FIND

The PP routine STF copies a specified portion of central memory to the system user buffer area. This allows functions to be performed by CP programs without the need for a special PP program. Bits 17 through 0 of the input register point to a table of commands that the user specifies in the following format.



Word	Bit	Description
0	0	Complete bit. The complete bit must be cleared before STF can be called. The bit is set on completion of any function.
1	59-42	Order code. The order code is either an absolute CM address, or a three-character table name or copy directive as described below. If the order code specifies an absolute CM address of a block of memory to be copied, the absolute memory flag (bit 41) must be set. If the order code specifies one of the following table names or copy directives (CPS and CPx), the absolute memory flag must not be set. The following names must be specified in display code.

Name	Description
APF	Attached permanent file table.
AUT	Auxiliary user table.
CPS	Copy a specific word of the user's control point into the buffer. The word address (relative to the start of the control point) is specified in the field length (bits 35 through 18).
CPx	Copy control point area x into the buffer. Normally, x is the binary value of the desired control point. However, when x is 0 (in either binary or display code), the user's own control point is copied.
CST	Channel status table.

Word	Bit	Description

Name	Description
DAT	Device activity table.
DDT	Dismountable device table.
DST	Device status table.
EST	Equipment status table.
FDT	DDT (fixed section).
FNT	File name table.
IDT	ID table.
IUT	INTERCOM user table.
JCA	Job control area.
JDT	Job descriptor table.
MST	Mounted set table.
MUX	Multiplexer table.
RBR	Copy RBRs.
RBT	Copy RBTs.
SEQ	Sequencer table.
SPT	Scheduler performance table.
STG	Tapes staging table.
TPS	Tapes table.
URT	Unit recovery table.
VDT	DDT (variable section).

Word	Bit	Description
	41	Absolute memory flag. This flag must be set if the order code is an absolute memory address.
	40-36	Unused.
	35-18	Length of absolute memory if the absolute memory flag is set; unused if the absolute memory flag is clear.
	17-0	FWA of buffer to receive memory image.
n	59-0	Zero terminator word.

When the order code is a table name, the length of the table in memory is used as the length of the user buffer. The information is read from memory without any system interlocks set; therefore, the data returned may be inconsistent.

CMR SEGMENTATION FOR ECS SYSTEMS

CMR segmentation is implemented for installations with ECS. A segmented system is intended to have most of its CP code ECS resident. Sections of code (segments) are loaded as needed into CM overlay areas. A segmented system is started from an ECS system image by a bootstrap monitor function, which overlays an existing system (disk or ECS) with a new system. The ECS system image for this new system must have been created in a previous step using utility program LDCMR (refer to section 10). For an ECS system, the equivalent of a disk system's system resident programs section in CMR is a number of specialized areas as shown in figure 2-5.

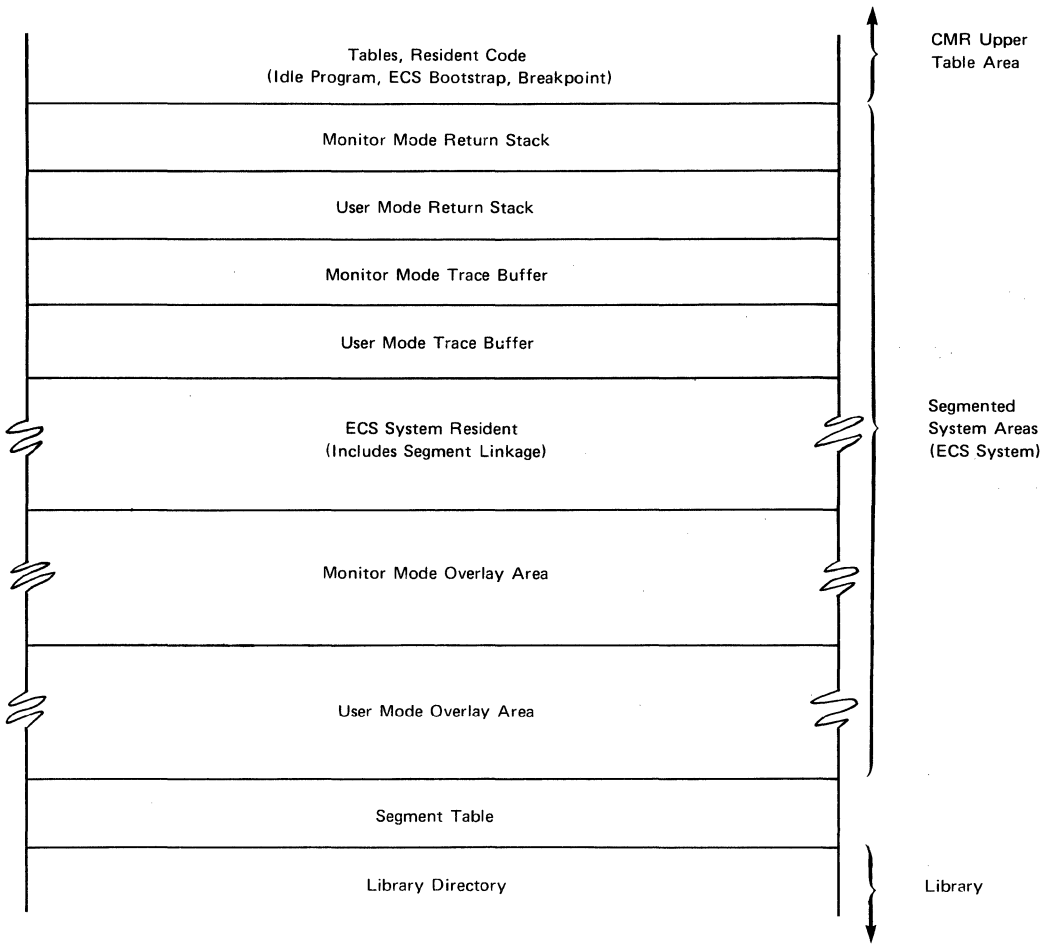


Figure 2-5. ECS System Areas

These areas are set up by the initialization segment INIT, loaded by the bootstrap in the monitor mode overlay area. The position of the library directory is adjusted if necessary by LDCMR. The ECS system resident contains the segment linkage program which loads new segments and passes control to them, the ECS parity error recovery routine, and some heavily used code (CPMTR start, CPMTR return to user). The trace buffer and return stacks are used by segment linkage.

The breakpoint table is initially empty. The breakpoint (N) display and commands allow breakpoints (temporary halts) to be set and released in the operating system during system execution. CM, ECS, and the operating system exchange package can be observed while the operating system is at a breakpoint.

The area table completely describes an ECS system. It is read in from the ECS image of a segmented system by the bootstrap monitor function.

The entry table contains the entry points for tables in CMR which are not directly accessible by a text symbol of the T. type. This table is used by utility LDCMR to load the ECS system and read the date-time stamp.

The breakpoint table, area table, and entry table are detailed in appendix B.

SEGMENT LOADING

Segment Linkage

Linkage is done through the GOTO (GOTOTAB), CALL, and RETURN macros. These macros can be used only in a segment defined through the SEGDEF and ENDSEG macros.

The following macro transfers control to the entry point EPTNAME.

```
GOTO            EPTNAME
```

The following macros transfer control to the entry point referenced in a GOTOTAB macro in the position indexed by register in TABLE.

```
GOTO            TABLE,Register

TABLE GOTOTAB   EPTNAM2
GOTOTAB         EPTNAM2
```

The following macro returns to the last address and segment saved by a CALL macro. The previous address and segment saved will be used by the next RETURN.

```
RETURN
```

Segment Definition

The following macro must be called immediately after the segment IDENT (first group).

```
SEGDEF          SEGMENT,mode,CM
```

SEGMENT	Segment name.
Mode	USER indicates user mode segment; otherwise, monitor mode is assumed.
CM	Segment must be CM resident.

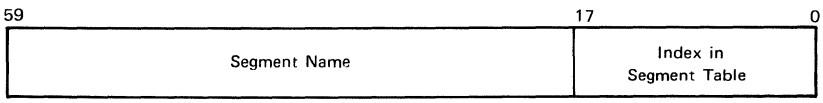
The following macro must be called immediately before the segment END.

```
ENDSEG
```

The name of the segment can also be the name of an entry point in the segment but not a tag. If it is not an entry point, the ENDSEG macro defines an entry point by that name referencing the second word of the segment.

The SEGDEF macro generates a segment header word with the tag ...REUSE.

Header format after processing by LDCMR is

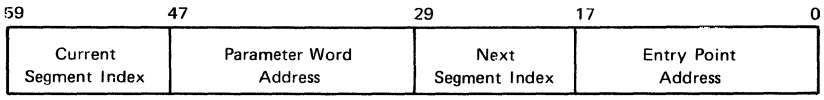


A segment should be serially reusable. If not, the segment must set its header word to zero before relinquishing control to prevent its reuse.

Parameter Word

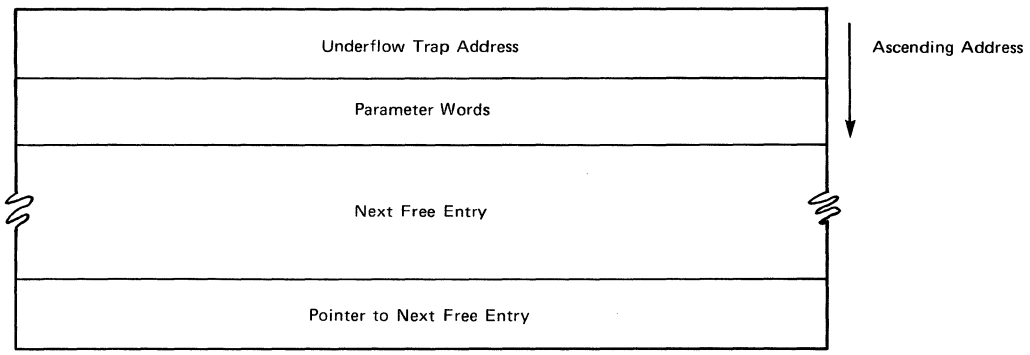
Linkage is done through a parameter word generated by the linkage macros and filled in by LDCMR. For a GOTO or a CALL, A1 is set to the address of the proper parameter word and a jump is made to one of the linkage processor entry points.

Parameter word format:

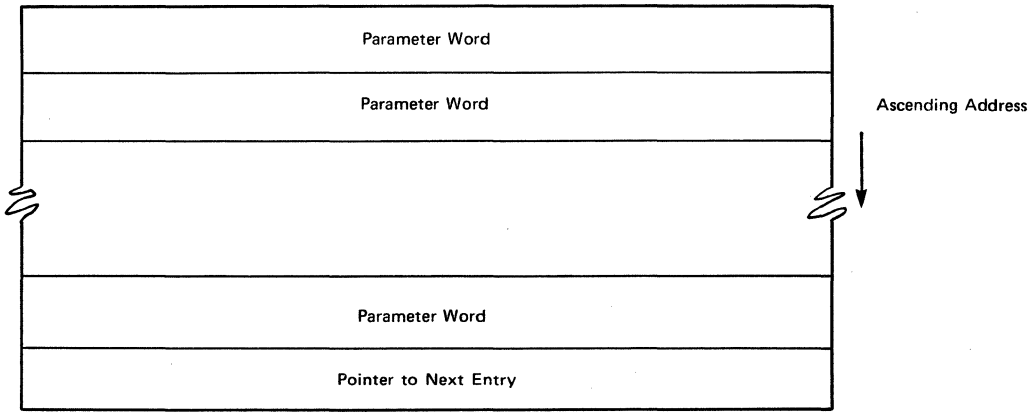


The addresses are absolute. The indexes are in the segment table. The parameter word is transformed into a return descriptor on a CALL by shifting it 30 positions and adding 1; it is stored in the proper return stack. A RETURN loads the last stored descriptor and performs a GOTO on it. If trace buffers are defined (refer to section 10), all parameter words processed are stored in the trace buffer for the current mode.

Return stack format:



Trace buffer format:



Segment loading is done by using the segment descriptor in the second word of the segment table entry for the segment. The address of the segment table entry for a segment is

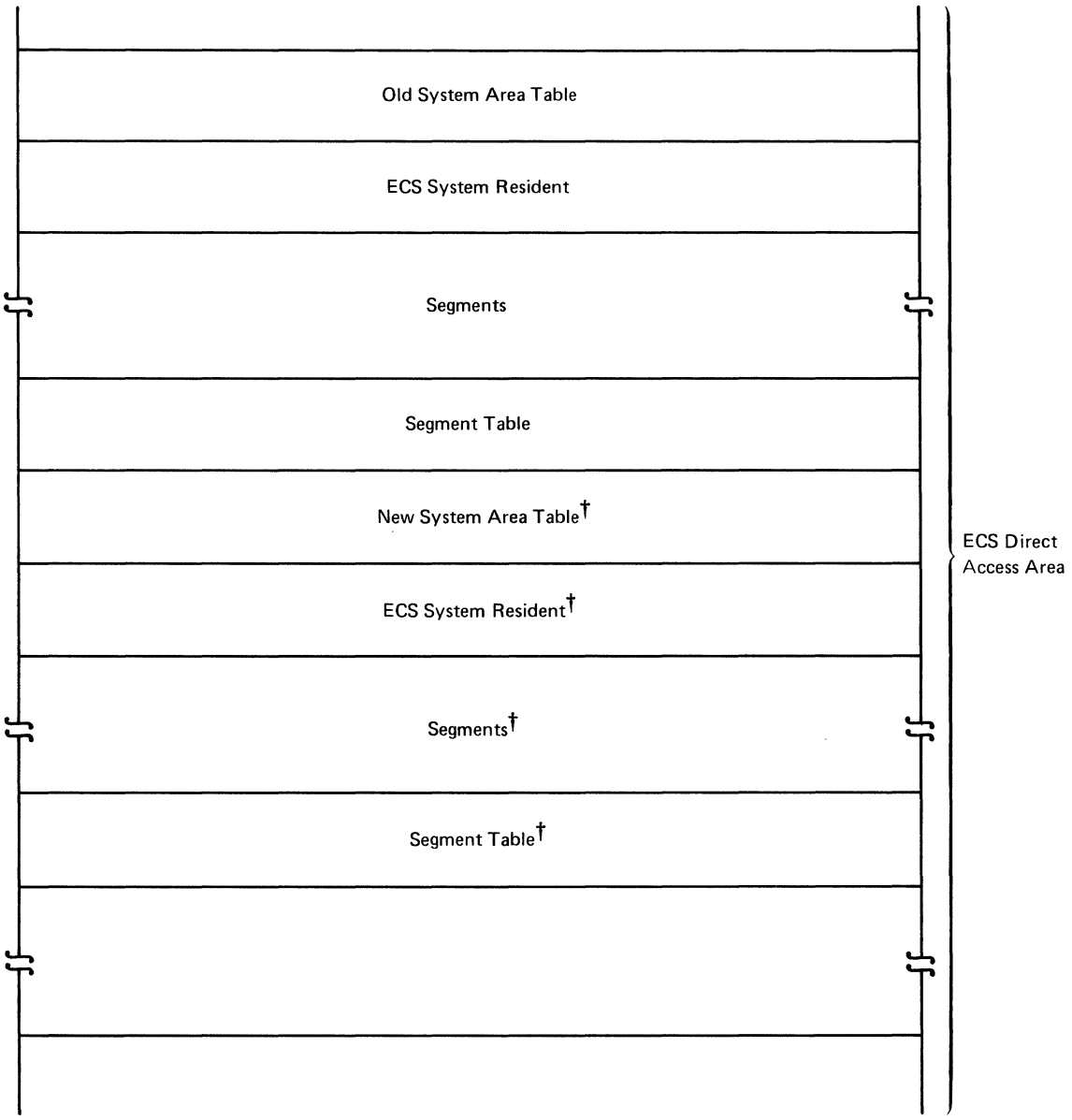
Segment table base address + 2 * index

The segment table is described in appendix B.

ECS SYSTEM IMAGE

The system is written to the ECS direct access area as an extension of control point 0 ECS field length. Two systems, named the old system (lower system) and the new system (upper system), can coexist in ECS. LDCMR creates ECS system images as shown in figure 2-6.

The operating system requires approximately 20K of ECS in the direct access area. For example, if 40K is to be available for user direct access, 60K must be allocated to the direct access partition. If LDCMR is to be used after deadstart, the direct access area must be large enough to contain two operating system CP code versions, approximately 40K, minimum.



† If a terminator replaces the new system area table, the new system sections do not appear.

Figure 2-6. ECS System Image

ECS ERROR RECOVERY

Segments are protected against ECS errors in transmission or storage. An autocorrective code is applied to obtain correction vectors. The correction vectors and two checksums are placed at the end of the segment as shown in figure 2-7.

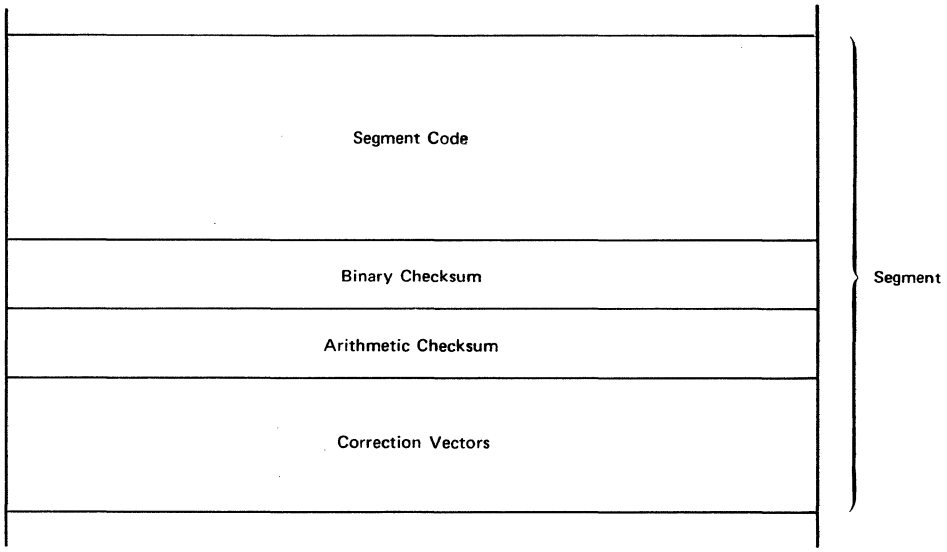
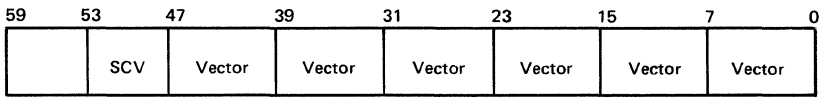


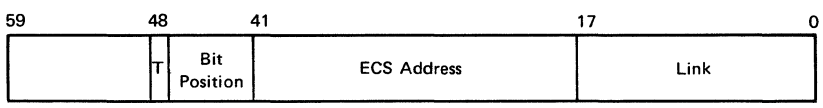
Figure 2-7. ECS Segment

The code is capable of correcting a single bit error every four words. Vectors correct each four words, and a 6-bit secondary correction vector makes vector words self-correcting.



Errors are corrected only if they are detected by the ECS parity error mechanism.† Detected errors are recorded in an error directory, then displayed in the dayfile (by a call to CEM, function 10). Error directory entries for a segment are linked to that segment's segment table entry. When an ECS error is encountered on a segment that has error directory entries, a first correction attempt uses the information in these entries. If this fails, the error entries are released and the full correction process is reapplied. The test of a successful recovery is the comparison of the existing checksums with the checksums for the recovered segment. The system is killed if a segment cannot be satisfactorily recovered. A recovered segment is written back to ECS, as it is possible the ECS error is transient.

Error directory entry format:

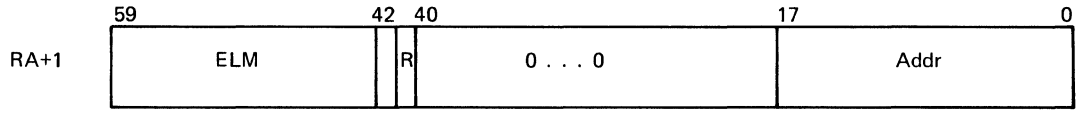


T	Meaning
0	Bit dropped.
1	Bit picked.

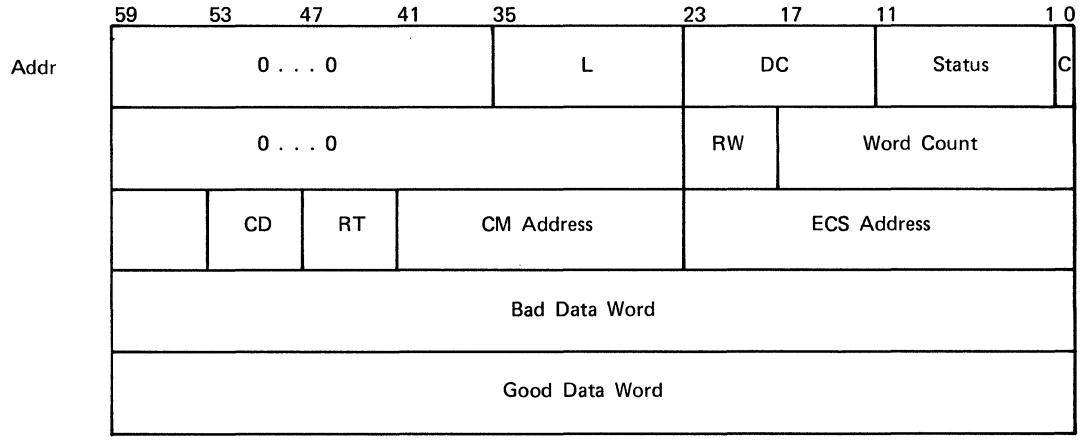
† The system ECS read parity error recovery routine executes before the correction vectors are used. This routine attempts a standard recovery algorithm and records the results in the CERFILE.

ELM - ERROR LOG MESSAGES

The ELM RA+1 request allows a user program to log error information related to the processing of direct user access to ECS errors. The format of the ELM RA+1 request is as follows:



The format of the parameter block is:



Word	Bit	Field	Description
0	35-24	L	Length minus one of parameter block.
	23-12	DC	Device code. EC ECS/Coupler Error.
	11-1	Status	Status flag. 0 No errors. 1 CERFILE message limit reached.
	0	C	Complete bit. This bit must be set to zero when the request is issued.
1	23-18	RW	Read/write flag. 1 Read. 2 Write.
	17-0	Word Count	Number of CM words transferred.

Word	Bit	Field	Description
2	53-48	CD	Error packet description code. Bits 53-49 are unused. Bit 48 equals 0 if error is recovered. Bit 48 equals 1 if error is unrecovered.
	47-42	RT	Retry count. This count is the total number of recovery retry operations associated with the error being reported in this request.
	41-24	CM Address	CM First Word Address of transfer.
	23-0	ECS Address	ECS First Word Address of transfer.

SYSTEM CONTROL POINT

A module or group of modules that performs a specific set of functions is known as a subsystem. A subsystem has the ability to make privileged requests (reserved by subsystems) in addition to any requests that a standard control point is allowed. Typical subsystems are a data base management system or Record Manager. Each subsystem has a unique four-digit ordinal by which it is referenced.

A system control point (SCP) is any control point occupied by one of the subsystems. The term SCP can describe both the control point and the subsystem at the control point. An SCP provides a centralized location for a subsystem, allowing it to perform functions for one or more jobs at other control points. This facility provides overall reduction of CM usage. Instead of several jobs having duplicate copies of these specially privileged modules in their field lengths, only one set of these modules occupies CM. This feature also improves coordination of control and access.

A user control point (UCP) is any job or module at a control point that makes a request to an SCP. A UCP can be a batch job, INTERCOM job, multiuser job, or another SCP.

MANAGING SUBSYSTEM RESOURCES

A subsystem at a control point may receive requests irregularly, leaving it idle for long periods. Since a system control point cannot be swapped out, some other action must be taken to reduce its memory requirement while it is idle.

The idle subsystem should be organized so that it can reduce its field length to 200 or 300 words. It should specify a 1/2-second periodic recall by using RCL with a 3777_8 delay period indicated by bits 10 through 0. When it receives a call, the subsystem is started immediately.

When a call is received, the following steps should be taken.

1. Issue a MEM call to acquire the field length necessary to process the requested information.
2. Acknowledge the request by resetting RA.SSC.
3. Load the subsystem overlay(s) required to process the request.
4. Process the request.

Take these steps in the order specified. If steps 1 and 2 are taken out of sequence, a memory deadlock could occur. If the request is acknowledged before the memory is requested for the SCP, it is possible for a second UCP to make a request and have it successfully passed to the SCP. Both UCPs would then have outstanding SSC calls. If the UCPs cannot be swapped out, and the sum of the field lengths of both UCPs and the required field length of the SCP is greater than what is available, the system is deadlocked. If this occurs, one of the UCPs must be dropped to resolve the conflict.

CALLSS MACRO

The CALLSS macro is issued by a UCP to request a particular function from a subsystem. A UCP can call more than one subsystem, either serially or concurrently. Also a UCP can make more than one call to an individual subsystem. Registers X1, X2, A1, and A6 are destroyed during execution of the macro and should not be used as parameters. The format of the macro is

```
label    CALLSS    ssid,addr,recall
```

label — An optional statement label.

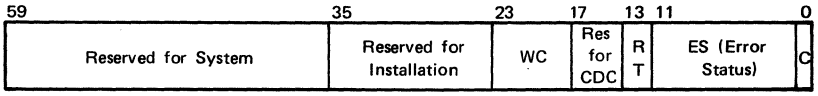
ssid — A required subsystem code. This parameter can be a register name.

addr — Address of the parameter block for this request. This parameter is required and must be nonzero. (The parameter can be a register name.) If the address is outside the UCP field length, the UCP is aborted.

recall — If nonblank, the request is made with auto recall, and processing at the UCP is suspended until completion of the request.

The parameter block pointed to by the addr parameter of the CALLSS macro is used by the UCP to pass parameter information to the SCP. The parameter block must be at least one word long.

The first word of the parameter block is used for calling and status information. The second and subsequent words of the parameter block are used for data which is passed to the SCP by the operating system. The format of the first word is



C — This bit indicates whether or not the current request has been completed. The user program must set this bit to 0 prior to executing the CALLSS macro. If the request has been completed, the operating system sets the bit to 1; otherwise, it remains set to 0.

ES — The operating system sets this field to indicate the presence of an error or unavailable system condition. This field is not set (all zeros) if the RT field is 0.

Bit 1	0	Subsystem currently running.
	1	Subsystem not initiated.
Bit 2	0	Subsystem not busy (that is, it has not acknowledged receipt of the last request).
	1	Subsystem busy.
Bit 3	0	Subsystem defined.
	1	Subsystem not defined.
Bits 4 and 5	Reserved.	

Bits 6 through 11 — Error condition other than bits 1, 2, and 3. Bits 6 through 11 can assume the following octal values.

Value	Significance
00	No other error.
01-17	Reserved for system errors.
20-67	Reserved for subsystem errors.
70-77	Reserved for installation.

RT — This field is set by the user prior to making a subsystem call.

Bit 12
- 0 — Operating system holds the current request until the subsystem is able to accept it. The UCP is placed in periodic recall and is not assigned the CPU until the request is accepted. The C bit is not set until processing of this request is complete.
- 1 — Operating system returns control to the user if the subsystem is present but is unable to accept the user's call. In this event, bit 2 in the ES field is set to 1.

Bit 13
- 0 — ES field is not set (with the possible exception of bit 2), and subsequent errors cause the UCP to abort.
- 1 — Operating system sets the ES to indicate which error condition occurred and returns control to the user on nonfatal error conditions (fatal errors cause a UCP abort).

  A message is issued to the UCP dayfile indicating the error condition.

When either of the RT bits is set and a condition is encountered that causes any of the bits in the ES field to be set, the operating system sets the C field and considers the operation complete. Therefore, it is necessary to reissue the CALLSS macro.

WC — Some subsystems require the user to specify the length of the parameter list. WC is the number of words (excluding the first word) to be passed with the request. The maximum is determined by the subsystem but may not exceed 77_8.

The remaining bits in this word are reserved.

If a call is issued with auto recall, the user's program is not restarted until the C field has been set.

A subsystem can call another subsystem as long as this does not result in a circular chain reaction. An SCP cannot make a call to itself, unless bit 12 of the RT field is set. An attempt to call itself without bit 12 set can result in a subsystem hang.

SYSTEM CONTROL POINT INTERFACES

Three types of system communication interfaces are unique to subsystems.

- Subsystem notification to the operating system that it is entering active status.
- Subsystem request acknowledgement.

- Special requests to the operating system allowing the subsystem to have access to the UCP field length, control and forward accounting data, obtain UCP swap and error status, transmit dayfile messages and error conditions to the UCP, and exit from active status.

Subsystems should use a symbolic reference for SCP locations and operations (for example, refer to RA.SSID rather than RA+50 and SF.READ rather than 10).

Requesting Active Status

The operating system does not recognize an SCP until a subsystem is loaded and ready to enter SCP status. To notify the operating system that it is entering active status, the subsystem puts its name and code in word RA.SSID (RA+50). RA.SSID is used by the operating system to identify the subsystem and must be maintained at all times. A CALLSS macro using automatic recall is then executed with the ssid field equal to SS.SYS.

Prior to assigning SCP status to the requesting control point, the operating system ensures that the following conditions are true:

- Request is made with recall.
- Program is called by a system origin job.
- Subsystem name in RA.SSID matches that in the corresponding subsystem control table entry.
- The same subsystem is not in SCP status at another control point and the maximum number of SCPs has not been reached.

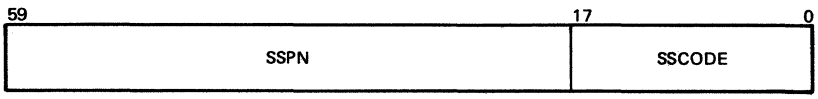
If any of the preceding conditions are not met, an appropriate diagnostic message is entered in the dayfile, and the job is aborted. Once the operating system has assigned SCP status to the subsystem, the C field is set to 1.

If RT bit 13 is set, failure to meet either of the last two tests will not abort the job but will set the following codes in bits 11 through 6 of the ES field.

04 Another control point has SCP status for this subsystem.

05 Nine control points have SCP status.

The format of SCP word RA.SSID is

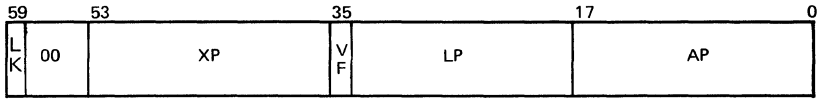


SSPN Subsystem program name.

SSCODE Identification code (SS.XXX).

Subsystem Request Acknowledgement

A word in the SCP field length, RA.SSC (RA+51), is set by the subsystem and used as a pointer to indicate where incoming requests from the UCP are put. The format of RA.SSC is



AP Address of the UCP parameter block.

LP Length of the request parameter block.

VF Variable move flag.

XP UCP exchange package address.

LK Interlock bit.

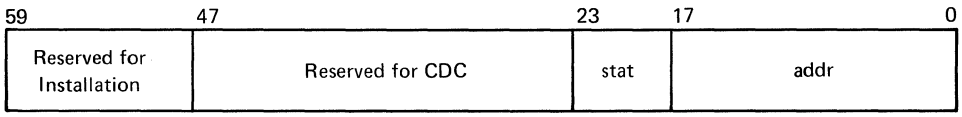
LK is set by the operating system when a request has been placed in the parameter area. It is cleared by the subsystem to acknowledge that the request has been received. When the bit is cleared, AP points to a parameter area in which the subsystem is prepared to receive the next request. After clearing LK, the subsystem should not attempt to rewrite word RA.SSC until the next request has been received.

If it is necessary to force a request, it is possible for a subsystem to call itself if bit 12 of the RT field in the first word of the parameter block is set. Attempting a call to itself without this bit being set can result in a subsystem hang.

If LK is set at the time of the request, one of the following actions is performed, depending on bit 12 of the RT field.

- Return control to the UCP and indicate a busy status in the ES field (RT=1).
- Hold the request and periodically attempt to give it to the SCP (RT=0).

The word at the address AP has the following format:



addr Same as the address in the CALLSS macro. This is a relative address within the UCP field length.

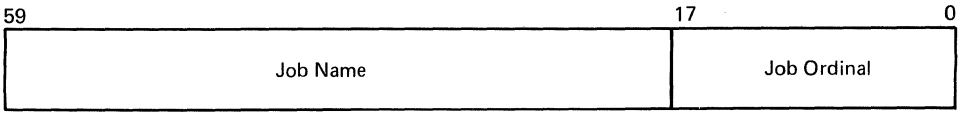
stat Status values.

0 Call is from a user.

1 Normal termination.

2 Error termination of UCP.

AP+1 is the job identifier whose format is



AP+2 through AP+LP-1 (minimum value of LP is 2) contains LP-2 words which are from the UCP parameter list starting at addr. The UCP parameter list address is taken from the CALLSS macro call.

If the VF bit is set, the length of the move is determined by the WC field in the first word of the UCP parameter block. If WC+3 is greater than LP, only LP words are moved in.

If XP is nonzero, the UCP exchange package is stored in the 16 words starting at XP.

Special Subsystem Requests to the Operating System

A subsystem at the SCP can make special requests of the operating system. These special requests, called subsystem functions (SFCALLs), are allowable from an SCP only. If an SFCALL is issued from a nonsystem control point, the control point is aborted with the error message PP CALL ERROR.

The SFCALL macro call has the following format:

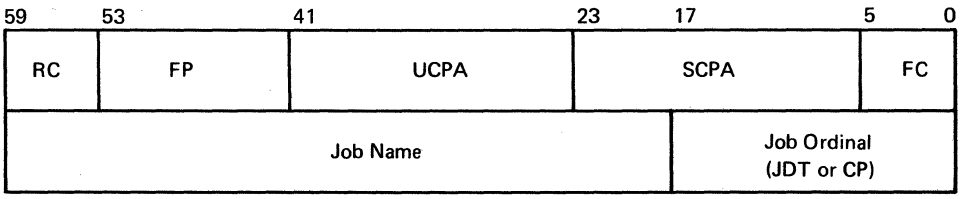
label SFCALL addr,recall

label An optional statement label.

addr Address of an SFCALL parameter word pair (refer to following SFCALL format). This parameter can be a register.

recall If nonblank, the job is put in automatic recall. Although this option is allowed, its use by a subsystem is discouraged.

A typical format of the SFCALL parameter word pair is



RC Reply code.

FP Function parameter.

UCPA Relative address within the UCP.

SCPA Relative address within the SCP.

FC Function code (an even number, incremented by one when the function has been completed).

If a parameter error prevents the processing of the function, RC will be set to a value in the range 40 through 77.

The following list of SFCALL return codes (RC) gives all codes which have been defined. Not all of these codes apply to this operating system. Codes 40 through 77 indicate that the function was not processed.

Return Code	Meaning
00	No error encountered.
01 through 33	Trivial errors (reserved for Control Data).
34 through 37	Trivial errors (reserved for installations).
40	At least one error encountered in list.
41	Job identifier invalid (NOS/BE considers all IDs to be valid; returns 45 if ID is not found).
42	SCPA not within the subsystem FL (code not returned by NOS/BE; the system aborts the SCP and issues a message to the dayfile).
43	UCPA not within the UCP FL.
44	User job swapped out.
45	User job not in system.
46	Reserved for Control Data.
47	Unknown function code.
50 through 56	Reserved for Control Data.
57	Connection previously established.
60	Connection rejected.
61	Connection not previously established.
62	Word transfer too long. †
63	UCP not established with subsystem. †
64	Subsystem not established with receiver. †
65	Attempt to set illegal error flag. †
66	Illegal dayfile processing flag. †
67 through 73	Reserved for Control Data.
74 through 77	Reserved for installations.

Possible return codes for each SFCALL function are shown in table 2-2.

† Not checked under NOS/BE.

TABLE 2-2. SFCALL RETURN CODES

Function	Return Code													
	40	41	42	43	44	45	57	60	61	62	63	64	65	66
SF.REGR			X		X	X								
SF.TIME			X			†								
SF.ENDT			X	X	X	X								
SF.READ			X	X	X	X								
SF.STAT					X	X								
SF.WRIT			X	X	X	X								
SF.EXIT						†								
SF.SLTC					X	X	X	X						
SF.CLTC					X	X			X					
SF.SWPO					X	X								
SF.SWPI					††	X								
SF.LIST	X		X		X	X								
SF.RERN					X	X								

† Return code of 45 may be returned if the user ID word is not zero.
†† Return code of 44 is returned if job is locked out.

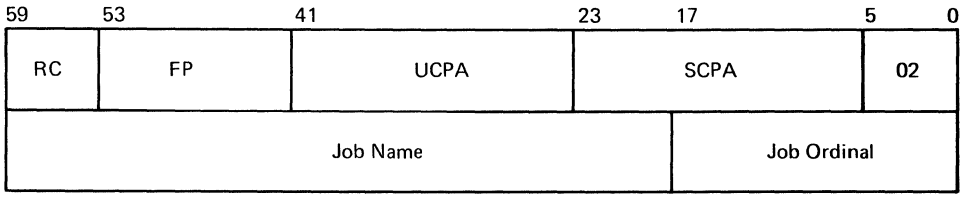
The function codes (octal) used with SFCALL are as follows:

Function	Code	Description
SF.REGR	02	Place message into the UCP dayfile and/or abort the UCP.
SF.TIME	04	Obtain accounting data for SCP.
SF.ENDT	06	Indicate end of task to UCP.
SF.READ	10	Read from UCP field length.
SF.WRIT	14	Write to UCP field length.
SF.STAT	12	Request status of UCP.
SF.EXIT	16	Exit from SCP status.
SF.SWPO	24	Indicate UCP as candidate for swap-out.
SF.SWPI	26	Request swap-in of UCP.
SF.SLTC	30	Set the long-term connection indicator.

Function	Code	Description
SF.CLTC	32	Clear the long-term connection indicator.
SF.LIST	34	Process of list of SF.xxxx functions.
SF.RERN	36	Set/clear rerun status of UCP.
SF.INS1-4	70,72,74,76	Reserved for installations.

SF.REGR - Regrets

The SF.REGR function code places a message of up to 40 characters into the dayfile of the UCP and/or aborts the UCP. It has the following format:



UCPA = 0 Do not abort the UCP.

UCPA ≠ 0 Abort the UCP. The F.ERPP error flag is set at the UCP.

SCPA = 0 No message.

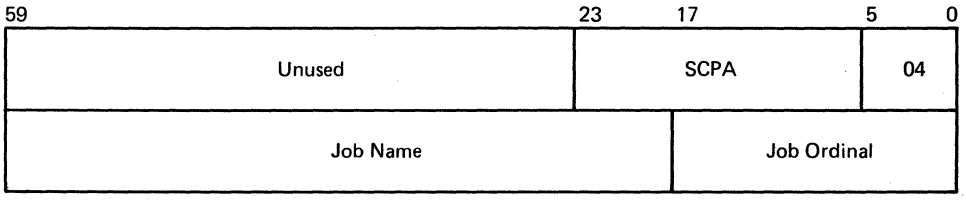
SCPA ≠ 0 Address of a message that is to be sent to the UCP dayfile.

FP Dayfile processing flags. The following is a list of the symbolic values and their meanings as defined for the FP field.

Value	Significance
F.SYCP	Send message to system dayfile and control point dayfile.
F.NMSN	Do not send message to control point dayfile.
F.JNMN	Do not send message to control point dayfile; job name is in message.
F.CPON	Send message to control point dayfile only.
F.ACFN	Accounting message to system dayfile only.
F.AJNN	Accounting message to system dayfile only; job name is in message.
F.ERLN	Send message to error file only.
F.EJNN	Send message to error file only; job name is in message.

SF.TIME - Accounting

The SF.TIME function code allows the SCP to obtain the accumulated accounting totals at its own control point. The format is



SCPA	Relative address within the SCP of a word block for accounting data.
Job name	Set equal to zero.
Job ordinal	Set equal to zero.

The accounting totals are returned to SCPA through SCPA+5.

SCPA+0	CPA time.
SCPA+1	CPB time.
SCPA+2	I/O time.
SCPA+3	CM field length.
SCPA+4	ECS field length.
SCPA+5	PP time.

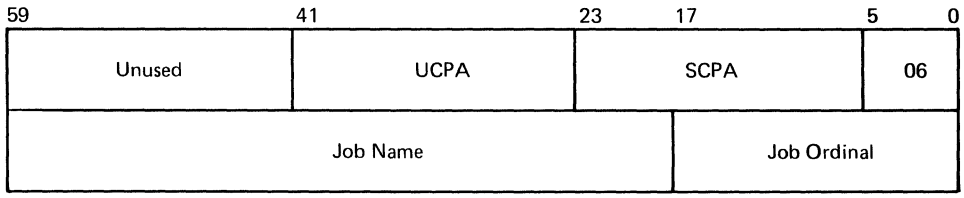
The symbol L.SACT must be used by all subsystems to define the work block lengths (for example, BSS L.SACT). It allows an installation to add a specially defined area to the word block by modifying the symbol and reassembling the subsystems using this symbol. The operating system is not responsible for setting or clearing the installation area.

When an installation defines one or more words for installation usage, these words must be located by using the last address of each word block (first + L.SACT-1) and referencing installation words backwards from this address.

Since the data delivered to this area varies among operating systems, a module must be provided for each operating system to process this area. Multitask users can use the data in this area to charge a particular UCP for the SCP resources used in processing the UCP's task. The resource data sent to the SCP can be the accumulated totals. The subsystem at the SCP has the responsibility for storing the previous totals of used resources and for calculating the differences.

SF.ENDT - Subsystem Task Complete

The SF.ENDT function informs the operating system and the UCP that the subsystem task has been completed, and allows the SCP to distribute the accumulated resource costs back to the UCP. Its format is



UCPA Relative address within the UCP of the request status word of the task being performed.

- > 0 Set bit 0 of the word at UCPA. (Restart UCP if auto recall was selected.) Reduce the request count by one.
- = 0 Do not set complete bit (bit 0). Reduce the request count by one.
- = -1 Do not set complete bit. Reduce activity count to zero, no matter how many requests are outstanding. (This activity count is the number of requests from the UCP JOBID to this subsystem.) Clear the long-term connection bit if it is set.
- < -1 Return error 43.

SCPA Relative address within the SCP of a word block of accounting data. The content is the same as specified by the SF.TIME function.

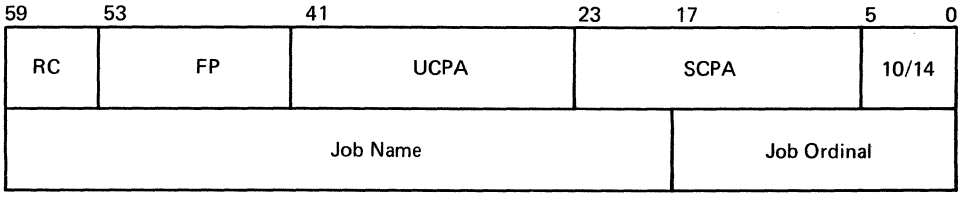
If SCPA is not 0, computer resource usage is based upon the data provided at SCPA. The resource accumulators at the UCP are incremented with this computed resource data. The core seconds that are computed and added to the UCP are based on the SCP field length at SCPA+3. An SCP that is multitasking should not charge a single user for the full SCP field length, but use a somewhat smaller value in this field. Each of these fields can be passed to the SF.ENDT function exactly as returned to the SCP by the SF.TIME function, or they can be adjusted as required by the SCP. At a later time, the normal computation of the core seconds at the UCP includes the CP and I/O time of the SCP. The result is that the effective field length of CP and I/O time used by the SCP is the sum of the UCP and SCP field length.

SF.READ - Read from the UCP Field Length

SF.WRIT - Write to UCP Field Length

SF.READ moves FP words from the UCPA to the SCPA, whereas SF.WRIT moves FP words from the SCPA to the UCPA.

The format of SF.READ and SF.WRIT is as follows:



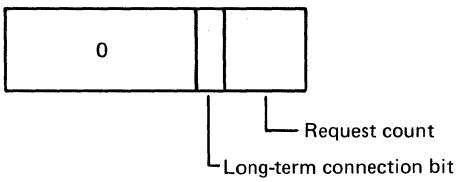
If RC is 42_8, the SCPA or SCPA+FP is outside the SCP field length. The SCP is aborted. If RC is 43_8, the UCPA or UCPA+FP is outside the UCP field length.

Transfers of large blocks of data between the UCP and SCP should be avoided because they may cause significant system response delays. Block transfers take place when the UCP calls an SCP or when an SCP makes an SF.READ or SF.WRIT request to read or write data to the UCP field length. Generally, no block transfer should attempt to transfer more than 64 words in one request or call.

SF.STAT - Status

The SCP uses the SF.STAT function to request the current status of the user job. The RC and FP fields are used for reply.

If RC is 0, the UCP is not swapped out or being swapped out. The state of the connection indicator is returned in FP as follows:



The UCPA and SCPA fields are not used but should be set to 0 in case optional uses are assigned in the future.

SF.EXIT - Exit from SCP Status

SF.EXIT removes the SCP from SCP status. The following actions should be taken in the order specified.

1. Make an SF.EXIT call.
2. Clear RA.SSID.
3. Issue ENDRUN or ABORT.

SF.SWPO - Swap Out

SF.SWPI - Swap In

SF.SLTC - Set Long-Term Connection

SF.CLTC - Clear Long-Term Connection

These functions, with the SF.ENDT function described previously, control the setting and clearing of bits in the inter-job connection table (T.IJCT). The T.IJCT is a part of the subsystem control table (T.SSCT) used to define and control SCP status.

Each user job has a corresponding word in the T.IJCT that contains nine connection control fields. When a control point is assigned SCP status, it is assigned one of the nine fields in each T.IJCT word. The connection control field contains the following connection indicators.

WAIT RESPONSE COUNT (alternately called request count)

This 3-bit field contains the number of unanswered requests submitted to the SCP by the UCP. The count is incremented each time a CALLSS request is passed to the SCP and decremented by an SF.ENDT function with UCPA=0.

LONG TERM CONNECTION

This bit is set by SF.SLTC and cleared by SF.CLTC. An SCP sets this bit to be notified when the UCP is terminating because of either an ENDRUN or any abnormal termination. The method of notification is described under End Processing for UCPs in this section.

SWAP OUT REQUESTED BIT

This bit is set by the SF.SWPO and cleared by the SF.SWPI. If the swap out bit is set in any one of the connection control fields for a user control point, that job is treated as if its CM time quantum has expired. This causes it to be swapped out unless it is locked in or there are no other jobs in the CM queue that could use the memory that would be released by swapping it out.

While a job is swapped out, it is not aged as long as any of its connection control field swap-out bits remain set. When an SF.SWPI function causes the swap-out bit to be cleared, the UCP is artificially aged so that it can be swapped in again promptly.

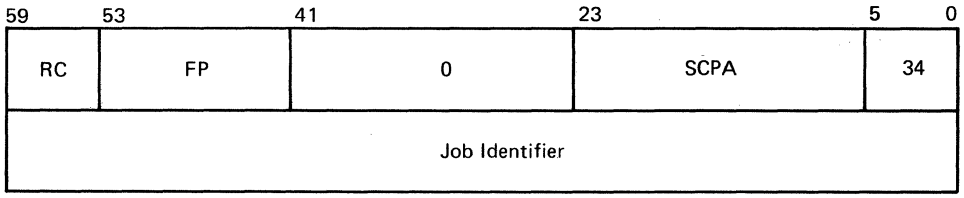
Subsystems should use the SF.SWPO and SF.SWPI carefully, because misuse can cause unnecessary and inefficient swapping among its user jobs.

Neither the wait response count nor the long term connection is considered when a job is selected to be swapped out. If, however, any of the connection control fields has a nonzero wait response count and the swap-out bit is not set, the swapping of that job will be delayed for over 1 second, if necessary, to give the subsystem a chance to complete the request before the user job is swapped out.

An SF.SWPI function is not set until the swap-in has been completed. If the swap-out request bit is set for a different SCP, the UCP remains a good candidate to be swapped out again. The existence of a wait response can, however, guarantee that it will be held at a control point for at least 1 second.

SF.LIST - Presents a List of SF.xxxx Functions

The multiple request capability is invoked through the use of the function code SF.LIST. The format of the SFCALL parameter word pair in this case is



RC Reply code.

FP Number of entries in the list.

SCPA First word address of the contiguous parameter list.

When using the list function, the entries in the list are each one word in length. The entry consists of the first word as described for each function. Only one UCP may be addressed for each list processed and this is the UCP indicated in the SF.LIST word pair. An SF.LIST function may not be included as a member of a list.

When the FC field is set complete by the operating system, the RC and FP fields must be examined to determine the action to be taken. If FP is 0, the entire list has been processed by the operating system. If FP is not 0, processing of the list was abandoned, and FP contains the number of entries remaining in the list. SCPA is set to the address of the first entry in the remaining list. The subsystem reissues the SF.LIST call by resetting the FC field and executing the SFCALL macro until FP equals 0.

The operating system will set the RC field only if an error is detected. Multiple issues of the same SF.LIST request (until FC is set complete and FP is 0) accumulate error returns whether or not the entire list is processed on one SFCALL.

The user of SF.LIST should consider the following notes and special conditions.

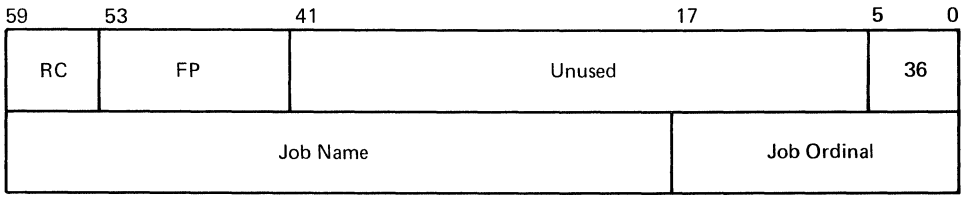
- The operating system aborts the SCP during SF.LIST processing if a fatal error occurs in the SF.LIST or in any member of the list.
- The detailed error conditions must be determined by examining the individual list entries whenever the SF.LIST RC field equals 40_8.
- The individual functions are handled in the same way whether or not the list mode is enabled.
- Error status 42_8 when SCPA is illegal means none of the list entries has been processed. This check is made prior to initiating the list process. Illegal SCPA is a fatal error.
- Error status 42_8 when FP is 0 means that none of the list entries has been processed on this call. If the subsystem handles the FP equal to 0 condition improperly, the entire list may have been processed prior to the subsystem abort. Illegal SCPA is a fatal error.
- List entries are processed sequentially by the operating system and entries detected as erroneous for any reason are considered completed. It is expected that in most cases the entire list will be processed on one SFCALL. The option of abandoning the list allows the operating system to take corrective action if it decides that either the length of the list, the complexity of the processing, or other reasons have possibly caused a long uninterruptable interval.

- If the SCP is aborted due to an error in one of the list entries other than the SF.LIST, RC equals 40_8, SCPA and FP are updated, and FC is set complete. The proper return status is also placed in the offending list entry.

- The functions SF.REGR, SF.RERN, and SF.EXIT are performed by a PP program instead of CPMTR. When any of these are included in a list, processing is transferred to the PP for that function. When it is completed, the list is abandoned rather than attempting to transfer processing back to CPMTR. To process the remainder of the list, it is necessary to reissue the SF.LIST function.

SF.RERN - Set/Clear Rerun Status

The SF.RERN function code sets or clears the status bit in the UCP area and input file's permanent file catalog (PFC) entry, which determines whether the job can be rerun. This bit is sometimes referred to as the no-rerun status bit. SF.RERN has the following format.



RC Reply code.

FP = 0 Sets status such that job can be rerun (clears no-rerun status).

FP = 1 Sets status such that job cannot be rerun (sets no-rerun status).

A user may wish to set the no-rerun status bit if he is performing operations that could destroy the validity of his files if done more than once. An example is the updating of a data base. If an error occurs and the job is stopped before the data base changes are complete, the user may not want to risk rerunning the job and making incorrect changes to a data base that is already partially modified.

END PROCESSING FOR UCPS

If a program running at a UCP is terminated while there is a long-term connection or a nonzero wait response count, the SCP is notified in the form of a two-word call to the SCP. The stat field in the word at AP+0 identifies the termination notification. If the UCP was terminated by an ENDRUN, the stat field contains the value 1. In the event of an error condition, the field contains the value 2. All normal calls from the user contain a 0. If the UCP aborts and reprieves, the SCP is not notified. If the UCP is aborted by an SF.REGR function, the SCP is notified as for any other error condition.

When a subsystem receives termination notification, it should complete all requests as quickly as possible, issuing an SF.ENDT for each wait response and an SF.CLTC if the long-term connection is set. Until this is done, the termination notification is repeated every 2 seconds and the message CONNECTED TO jobname is flashed at the UCP on the B display (jobname is the SCP job name).

If the subsystem is unable to complete the outstanding requests, it should issue an SF.ENDT to that UCP with a -1 (777776_8) in the UCPA field to unconditionally release the UCP.

If the CONNECTED TO jobname message continues to flash for an extended period of time, it can be assumed that the SCP is not functioning correctly. The SCP and all of its connected UCPs can be terminated by an operator drop on the SCP.

NORMAL SCP TERMINATION

The following steps should be taken to terminate execution of a subsystem.

1. Stop accepting any requests.
2. Complete processing any requests already received.
3. Issue an SF.EXIT.
4. Issue an ENDRUN.

To force a subsystem to stop accepting requests, the user can have the subsystem send a dummy request to itself. The RT bit (bit 12) must be set whenever a CALLSS is issued. If bit 2 of the ES field is set, the operating system has delivered a request from another UCP, which must be processed along with any other uncompleted requests before the SF.EXIT is issued. The operating system sets the LK bit in RA.SSC and does not send any more requests as long as LK is not cleared.

ABNORMAL SCP TERMINATION

A subsystem should make use of the RECOVR capability (refer to NOS/BE Reference Manual) so that the subsystem will be reprieved if an error condition occurs. During reprieve processing, it can attempt to complete all outstanding requests, so as to cause as little user interruption as possible. The subsystem retains SCP status during reprieve processing.

If a subsystem has completed reprieve processing, or was not reprieved and attempts to terminate without issuing the SF.EXIT, the operating system performs the SF.EXIT and issues the message SYS CTL PT STATUS CANCELLED.

When performing the SF.EXIT function, the operating system determines if the SCP still has any long-term connection or active wait-response counts with a UCP. If there are any, the following actions will be performed by the operating system.

1. SCP is aborted with the message EXIT - WITH CONNECTIONS.
2. The wait-response counts and long-term connections are nullified.
3. The message subsystem name ENDED BY SYSTEM is sent to the UCP dayfile.
4. UCP is rerun or if rerun is not allowed, aborted.
5. UCP can perform reprieve processing.

HOW TO DEFINE A SUBSYSTEM

The CMR symbol N.SBSYS determines if T.SSCT is assembled in CMR. The default value is 0, which causes T.SSCT not to be assembled. N.SBSYS is the maximum number of subsystems that may be defined. If set to a value not 0, that value must be at least 2. Installations that want to define their own subsystems should use position ordinals 10_8 through 17_8.

The SSCT macro is provided for defining subsystems. The macro has the following parameters:

SSPN	Subsystem program name.
SSCODE	Position ordinal for the subsystem.
PUF	Permit user files.

SSCODE is the unique code that identifies the subsystem and determines the position within T.SSCT at which the defining entry is assembled. Its value may not be greater than N.SBSYS. This is the same code that the subsystem uses in word RA.SSID when it requests system control point status. It is also used as the ssid for the CALLSS macro to identify which subsystem is to be called.

If the PUF parameter is omitted, the subsystem program must be called by a system origin job or it will not be granted system control point status. The PUF parameter should only be used during subsystem development.

System control point macros are contained in SSYTEXT.

SETMFL MACRO

Several programs, such as CDCS, that execute as system origin jobs with SCP status, require a maximum field length similar to that specified by the CM parameter on a batch origin job statement. The SETMFL macro sets the maximum field length for system origin jobs. The format of the macro is

```
label    SETMFL    maxfl
```

label	An optional statement label.
maxfl	Maximum field length; maxfl can be a constant, symbol, or register name.

The SETMFL macro is defined in the system text SSYTEXT. When SETMFL executes, it makes an RA+1 system request with auto recall set.

Under certain conditions, the maximum field length (MFL) established may differ from the field length requested. If the requested field length is 0, less than 0, or larger than 400000_8, the system default field length (set by IP.SFL) is used. If the requested field length exceeds the system MFL (set by IP.MFL), the system MFL is used.

The SETMFL macro should be executed only once in a system origin job. Subsequent calls to SETMFL or any calls to SETMFL from a nonsystem origin job are ignored unless the requested field length exceeds the current MFL. In that case, the job is aborted.

PROGRAMMING TIPS

A system control point runs at a high CPU priority level. When it receives a request from a user, the CPU is immediately assigned to process the request. In most cases, the CPU is not reassigned to any of the lower priority jobs until the SCP releases it by issuing an RCL. If the SCP uses the CPU inefficiently, monopolizing it for long periods of time, the throughput of the whole system will suffer. When subsystem action is blocked, waiting for actions from other parts of the system, the CPU must be relinquished.

It is also important that the subsystem be ready to accept and process requests from other users. Before issuing an RCL, it should be certain that the LK bit is not set, indicating that it is ready to receive a new request. The auto recall bit should not be used in any RA+1 call from an SCP because an auto recall status would prevent it from responding to new requests as they come along.

PERIPHERAL PROCESSORS 3

PERIPHERAL PROCESSOR ORGANIZATION

When the operating system is loaded into the computer at deadstart time, MTR and DSD are loaded into PP0 and PP1, respectively, where they reside permanently. All PPs in the system contain a group of permanently assigned storage locations called the PP direct cells. Refer to table 3-1 for PP direct cell assignment. The contents of the direct cells are not guaranteed from one PP overlay to the next. Each PP, except PP0 and PP1, contains a copy of the PP resident program, which handles common service functions for the PP programs that are loaded into the unassigned pool processors.

TABLE 3-1. PP DIRECT CELL ASSIGNMENT

Location (Octal)	Identifier	Function
0	D.Z0	Temporary storage.
1	D.Z1	Temporary storage.
2	D.Z2	Temporary storage.
3	D.Z3	Temporary storage.
4	D.Z4	Temporary storage.
5	D.Z5	Temporary storage.
6	D.Z6	Temporary storage.
7	D.Z7	Temporary storage.
10	D.T0	Temporary storage.
11	D.T1	Temporary storage.
12	D.T2 (PP output register buffer)	Temporary storage.
13	D.T3	Temporary storage.
14	D.T4	Temporary storage.
15	D.T5	Temporary storage.
16	D.T6	Temporary storage.
17	D.T7	Temporary storage.

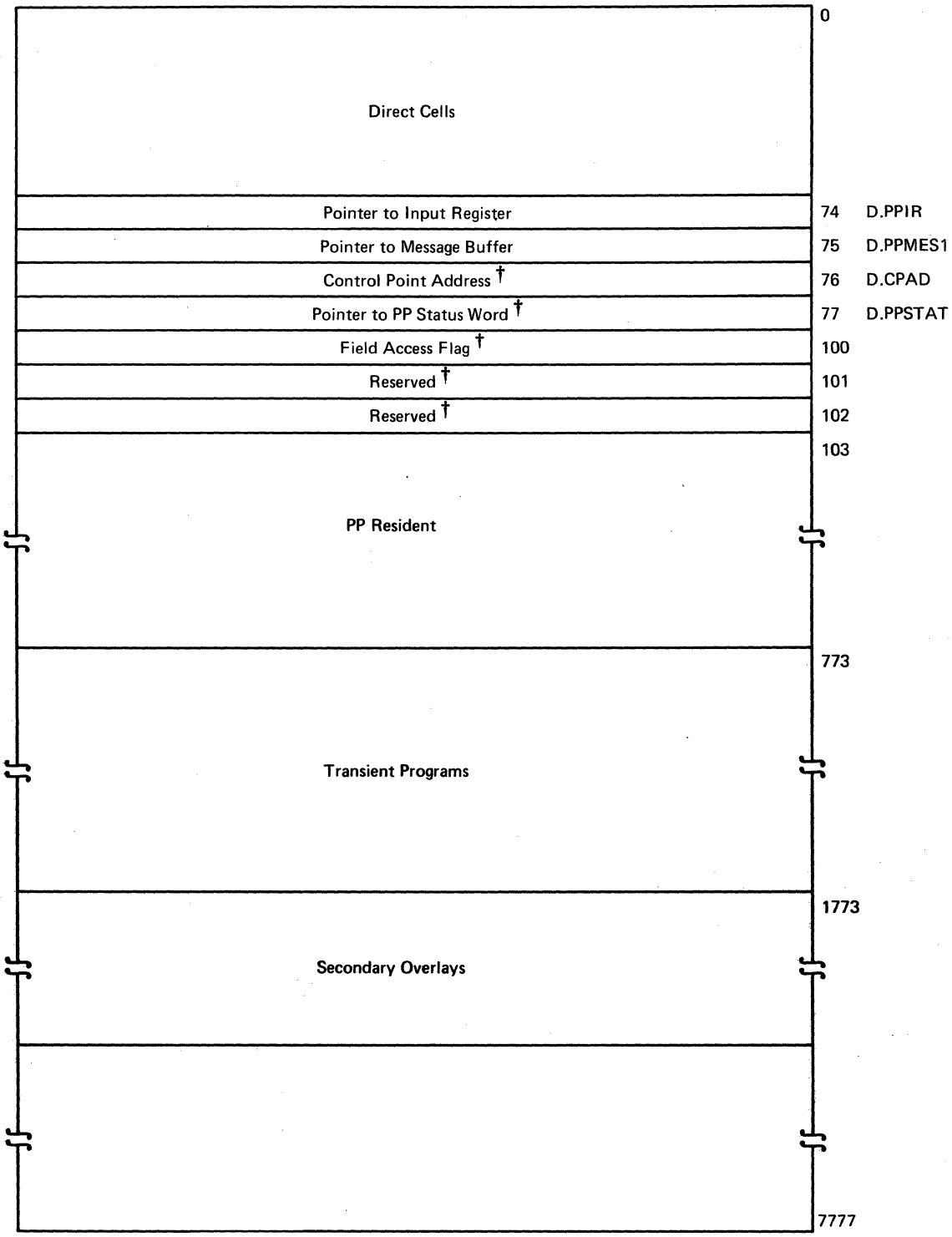
TABLE 3-1. PP DIRECT CELL ASSIGNMENT (Contd)

Location (Octal)	Identifier	Function
20	D.FNT/D.TW0	D.FNT through D.FNT+11 (octal) contains words 2 and 3 of the FNT, referred to as the file status table (FST).
32	D.EST/D.JPAR, D.TH2	D.EST through D.EST+4 contains the EST entry in process. D.JPAR contains a job parameter word.
37	D.DTS/D.JFL, D.TH7	D.DTS contains the device type code in the left 6 bits and the allocation type code in the right 6 bits. D.JFL contains the CM field length requirement returned to the caller by 2TJ.
40	D.BA/D.FR0	D.BA through D.BA+4 (buffer address) contains the first word of the FET.
45	D.JECS/D.FR5	ECS field length returned by 2TJ to caller.
46	D.JPR/D.FR6	Computed job priority returned to caller by 2TJ.
47	D.JTL/D.FR7	Job time limit returned to caller by 2TJ.
50	D.PPIRB/D.FF0	D.PPIRB through D.PPIRB+4 contain PP input register contents.
55	D.RA/D.FF5	Reference address divided by 100 (octal) for the control point to which the PP is attached.
56	D.FL/D.FF6	CM field length divided by 100 (octal) for the job at the control point to which the PP is attached.
57	D.FA/D.FF7	Address of the second word of the FNT entry in process.
60	D.FIRST/D.SX0	This and next cell contain 18-bit CM address of word FIRST in circular I/O buffer.
62	D.IN/D.SX2	This and next cell contain 18-bit CM address of word IN in circular I/O buffer.
64	D.OUT/D.SX4	This and next cell contain 18-bit CM address of word OUT in circular I/O buffer.
66	D.LIMIT/D.SX6	This and next cell contain 18-bit address of word LIMIT in circular I/O buffer.

TABLE 3-1. PP DIRECT CELL ASSIGNMENT (Contd)

Location (Octal)	Identifier	Function
70	D.PPONE/D.SV0	Can be set to constant value +1.
71	D.HN/D.SV1	Can be set to constant value +100 (octal).
72	D.TH/D.SV2	Can be set to constant value +1000 (octal).
73	D.TR/D.SV3	Can be set to constant value +3.
74	D.PPIR/D.SV4	PP input register address.
75	D.PPMES1	Address of first word of PP message buffer.
76	D.CPAD	Address of control point area in use by PP.
77	D.PPSTAT	Pointer to PP status word.
100	R.FAF	Field access flag.
101		Reserved.
102		Reserved.

PP programs are loaded into a pool processor by the PP resident and remain in a PP only until they have completed a specific function. When loaded, the program can load additional overlays to help complete its function; on completion, the program can be overlaid by another transient program loaded by PP resident to perform another, often unrelated, function. A typical layout of a pool PP loaded with a transient PP program and an overlay is shown in figure 3-1. The PP direct cells occupy locations 0 to 77; the PP resident is loaded starting at location 103 to approximately 773. The remainder of the PP is occupied by a PP transient program.



† Cells 76-102 constitute the five bytes of the PP status word in central memory.

Figure 3-1. Pool PP Layout

PP COMMUNICATIONS

For each pool PP, CMR has an area used for communication between the PP monitor (PPMTR) and the PP (refer to figure 2-4). Each PP communications area contains a PP input register and a PP output register (each one-CM-word long), plus a six-CM-word message buffer.

When a PP is idle, the input register in the communications area contains zero. When PPMTR assigns a PP to load and run a transient PP program, it loads a request word into the assigned PP input register. Figure 3-2 shows the format of a PP input register for a transient program called from a CP program. CPMTR inserts the requesting program's control point number into bits 39 through 36 of the word and clears bit 41. Bits 35 through 0 appear in the input register exactly as they did in RA+1 of the requesting CP program.

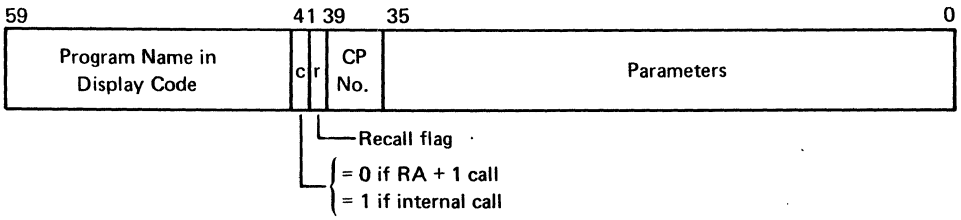


Figure 3-2. PP Input Register

The PP resident in each PP constantly scans its own input register. When it becomes nonzero, the PP resident issues the M.PPLIB monitor request with the program name and load address. A search is made for the program in the CMR library directory and the program is prepared for loading. The PP loads the transient program at location 773, stores the address of the control point area to which the PP is assigned in direct cell D.CPAD, and transfers control to location 1000 to start execution. If the transient program needs to load an overlay, it calls a subroutine in the PP resident. PP resident loads the overlay. Since the input register is not cleared until the PP becomes idle, parameters transmitted by PPMTR in the input register can be read by the transient program and/or any overlay. When the PP transient program has completed its function, it sends a request to PPMTR to drop the PP. PPMTR clears the PP input register and records the fact that the PP is idle and that another program can be loaded into the PP. The transient program terminates by executing a jump to the idle loop of the PP resident which scans the input register for the next assigned task.

When PP resident has a monitor request, it places a message into the PP output register in the PP communication area. The leftmost byte contains a number which identifies the function requested; other bytes may contain parameters for the request. Additional information or parameters for the request may be placed in the message buffer in the PP communications area. After making the request, PP resident waits for the first byte of the output register to be set to zero, signaling that the monitor has processed the request. If CPMTR or PPMTR need to communicate with the PP about the request being processed, the PP stores the necessary information in the remaining bytes of the output register.

PP RESIDENT

The PP resident program performs the following main functions.

- Handles all communication between MTR and the transient and/or overlay program.
- Loads transient programs and overlays and initiates execution of these programs.
- Controls PP access to the assigned control point's field length.

The PP resident consists of a series of routines, each of which performs a specific function (figure 3-3). These resident routines are used by the transient and overlay programs as required. R.OVLJ and all other PP resident routines, except R.IDLE, destroy temporary cells 0 through 17. R.IDLE destroys direct cells 20 through 22 and some of the temporary storage cells. The names, locations, calling sequences, and functions of the routines follow.

R.IDLE — PP RESIDENT IDLE LOOP

Calling sequence: LJM R.IDLE

In the idle loop, PP resident continually scans its input register for an assigned task. When R.IDLE finds an assigned task, the control point address (D.CPAD) is set, the field access flag (R.FAF) is cleared, and R.OVLJ is entered to load the overlay.

R.OVLJ — PRIMARY OVERLAY (TRANSIENT PROGRAM) LOADER

Calling sequence: Store name of overlay left-justified in D.T6,D.T7.
LJM R.OVLJ

When this routine is called, PP resident loads a new primary overlay at C.PPFWA minus L.PPHDR and transfers control to location C.PPFWA.

R.RAFL — REQUEST CONTROL POINT FIELD LENGTH ACCESS

Calling sequence: RJM R.RAFL

The storage move flag for the control point is tested. If set, a call is made to R.TAFL. When clear, the field access flag in the PP status word is set, the RA in D.RA is reset, and the FL in D.FL is reset.

R.PAUSE is the same as R.RAFL.

R.TAFL — TERMINATE CONTROL POINT FIELD ACCESS

Calling sequence: RJM R.TAFL

This routine is called to clear the field access flags in the PP byte R.FAF and in the PP status word.

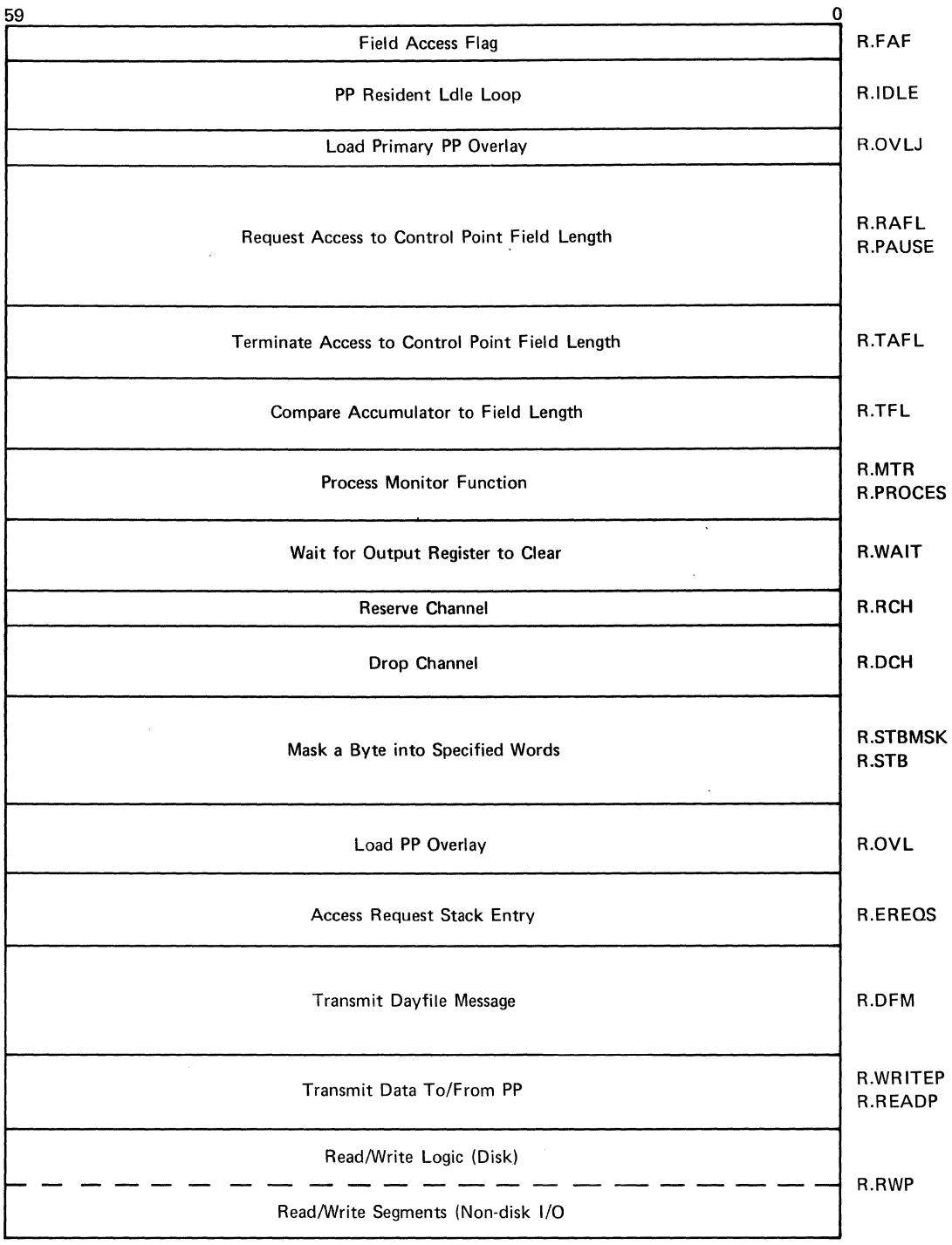


Figure 3-3. PP Resident Routines

R.TFL — TEST FIELD LENGTH

Calling sequence: Load relative address.
RJM R.TFL

This routine ensures that a relative address is within the field length limits. The 18-bit address is added to the control point reference address and compared with the field length. If the resultant address is out of range, R.TFL exits with a zero in the A register; otherwise, R.TFL exits with the resultant absolute CM address (RA + relative address) in the A register. A call to R.RAFL sets a flag which enables R.TFL to return a reliable result. R.TAFL clears the flag. Therefore, the transient and its overlays must not call R.TFL until R.RAFL has been called.

R.MTR — PROCESS MONITOR FUNCTION

Calling sequence: Store function parameters in D.T1 to D.T4.
Load function code.
RJM R.MTR

This routine places the function code in D.T0, writes D.T0 through D.T4 to the output register, and waits for the output register to clear via a call to R.WAIT.

R.PROCES is the same as R.MTR.

R.WAIT — PP WAIT LOOP

Calling sequence: RJM R.WAIT

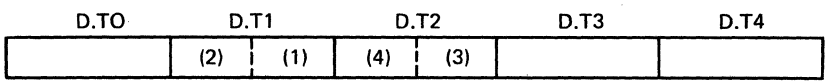
This routine determines if the monitor function is for MTR or CPMTR. If the MXN instruction is not available, R.WAIT is modified at deadstart causing R.WAIT to assume that all functions are for MTR. If the function is for CPMTR, the PP input register address is written into T.PPID and T.MXNCTL is read in and executed. If the function is for MTR, the input register address is written in T.PPIP.

After either action, R.WAIT idles until byte zero of the output register is cleared. If the field access flag (R.FAF) is set, R.WAIT pauses for relocation via calls to R.RAFL.

R.RCH — REQUEST CHANNEL

Calling sequence: Load channel numbers.
RJM R.RCH

This routine stores the channel numbers loaded in the A register in D.T1, inserts the monitor function M.RCH into D.T4, and writes D.T0 through D.T4 to the output register for that PP. Channel numbers in D.T1 and D.T2 are assigned by monitor on the following priority basis.



The highest priority is given to the channel number in the rightmost 6 bits of D.T1; the second highest to the channel number in the leftmost 6 bits of D.T1, and so on.

When assigning alternate channels, monitor discontinues its search of D.T1 and D.T2 when it encounters 6 zero bits. If only one alternate channel is required, the programmer must clear D.T2 before calling R.RCH. As an example, the coding for requesting primary channel 12, alternate channel 13 is:

```
LDN     0

STD     D.T2

LDC     1312B

RJM     R.RCH
```

Normally, MTR stops looking for alternate channels after four have been investigated. In the preceding example, only two channels are investigated.

When R.RCH is called, the function is not considered complete until byte 0 of the output register is cleared, signaling that a channel has been assigned.

DSD and a few other programs need an immediate reply on a channel request, even if the channel is already reserved by another PP. These programs do not use R.RCH; they issue an M.RCH function through R.MTR. These requests are made with a zero in D.T4 and, if the channel is available, the monitor assigns it and sets D.T4 to nonzero in the reply. If the channel is not assigned to the requesting PP, D.T4 remains zero in the reply.

R.DCH — DROP CHANNEL

Calling sequence:
```
Load channel number.
RJM     R.DCH
```

The specified channel is dropped. Since more than one PP can request the same channel at the same time, an MTR request must be used to reserve a channel. Only the PP reserving the channel can release it by making an R.DCH call. The function modifies the CST entry for the channel to indicate that it is free.

R.STBMSK

Address in PP resident of a logical mask used by R.STB routine. This mask is initially 7700 octal. The value should be restored by any routine which substitutes an alternate mask.

R.STB — STORE BYTE

Calling sequence:
```
Load L (list).
RJM     R.STB
```

List has the following form.

```
L(BYTE)
L(WORD1)
L(WORD2)
    .
    .
    .
L(WORDn)
ZERO
```

A logical AND is performed on the mask at location R.STBMSK for each word in the list before an exclusive OR is performed with word BYTE. R.STB is used primarily to substitute channel numbers in driver overlays.

R.OVL — OVERLAY LOADER

Calling sequence:
```
Store name of overlay left-justified in D.T6,D.T7.
Load A register with load point address.
RJM     R.OVL
```

An M.PPLIB monitor function is issued along with the overlay name and load address via an R.MTR call. CPMTR determines overlay residence (disk, ECS, or CM) and sets up the load accordingly. For disk resident overlays, a stack request is set up in the PP message buffer and the M.PPLIB function is changed to an M.ICE/EX.SPM function. For ECS resident overlays, a system circular buffer (DDP or non-DDP) is reserved and the buffer parameters are passed in words 1 and 2 of the PP message buffer. For CM resident overlays, the CM address and length of the overlay is passed in word 3 of the PP message buffer.

After the M.PPLIB request is acknowledged, R.OVL calls R.READP to load the overlay. If an illegal program name has been requested, R.READP returns an error status in D.T4+C.RWPPST. R.OVL issues an M.ABORT function and exits to R.IDLE. Otherwise, R.OVL returns to the caller.

R.EREQS — ENTER STACK REQUEST

Calling sequence:
```
Store L (request) in D.TO.
RJM     R.EREQS
```

This routine adds the control point number to the already formatted request, writes it in words 1 and 2 of the message buffer, clears word 3, and issues the request via the M.ICE/EX.SPM function.

R.DFM — ENTER DAYFILE MESSAGE

Calling sequence:
```
Load L (message) + flag bits.
RJM     R.DFM
```

A message is written to the dayfile and/or displayed on the console. The flag bits in the high-order 6 bits of the A register are used to determine message destinations. In the following flag bit values, one or more bits may be on; all are optional (refer to M.DFM). When a bit is set, the corresponding action occurs.

Bit Set	Description
0	Do not send message to B display.
1†	Do not send message to control point dayfile.
2††	Do not send message to system dayfile (A display).
3	Flag the message as an accounting message.
4†	Send the message to the hardware error file.
5††	Do not put the job name in the message.

† If bits 1 and 4 are set, the message is not sent to an INTERCOM control point dayfile but is sent to other control points.

†† If bits 2 and 5 are set, the time is omitted and replaced with blanks in the control point dayfile. This option identifies messages from a task that was executed on a different mainframe.

R.READP — TRANSMIT DATA VIA CHANNEL FROM STACK PROCESSOR

R.WRITEP — TRANSMIT DATA VIA CHANNEL TO STACK PROCESSOR

Calling sequence:
```
Load L (request).
Load 0 if request already issued.
RJM    R.READP
       or
RJM    R.WRITEP
```

These routines control transmission of data to/from the PP via a channel-stack processor interface or a CM system circular buffer interface. R.READP or R.WRITEP calls R.RWP to perform the read/write logic after which the following is returned to the caller.

D.T3+C.RWPPLW	LWA+1 of data transmitted.
D.T3+C.RWPPST	Upper 6 bits of status.
D.T3+C.RWPPWT	Number of PP words transmitted.
D.T4+C.RWPPST	Lower 12 bits of status.

R.RWP — PERFORMS READ/WRITE LOGIC FOR R.READP/R.WRITEP

Calling sequence:
```
Store L (request) in D.T0.
Load IAM/OAM instruction.
RJM    R.RWP
```

This routine is called by R.READP/R.WRITEP to perform the read/write logic. If D.T0=0, read/write logic is entered directly. Otherwise, the PP message buffer address is added to the stack request which is issued by R.EREQS.

Read/write logic idles on byte 0, word 3 of the PP message buffer. If the field access flag is set, R.RWP pauses via R.RAFL while waiting. The following values may be found in byte 0.

Value	Significance
1 through 4	Indicates channel-stack processor transfers.
5	Indicates that word 3 contains parameters to load a PP overlay from CM.
6 or greater	Indicates a system circular buffer interface.

A system circular buffer interface requires the load of a segment into the area between the end of R.RWP and C.PPFWA. Unless the information required to load the segment is already loaded, the information is contained in word 3 of the message buffer.

Segments exist for loading ECS-resident PP overlays either through CM buffers or directly from ECS via a DDP. Segments also exist for communicating with the 819 disk stack processor, HSP (applicable to CYBER 176 only). Segments are loaded from T.PPOVL in CMR.

R.RWP returns to R.READP/R.WRITEP after the request is complete.

FIELD ACCESS FLAG USAGE

The control point field access flag (R.FAF) is found at location 100_8 in PP resident. A copy of the flag is kept in CM in the PP status table entry (T.PPS1) for each PP. It is used by the PP to prevent storage moves at a control point while the PP is accessing the control point's field length. R.RAFL and R.TAFL (refer to descriptions in this section) obtain and release field access.

The field access flag must be set whenever data is read or written within a control point's field length. If a PP program is looping, waiting for an external event to occur, the loop must be performed while the field access flag is not set, or the loop must include a call to R.RAFL. When no field access is required for a major operation (such as searching a CMR table), it is advisable to call R.TAFL before the process.

Execution of the R.MTR subroutine or any resident routine that calls R.MTR (that is, R.RCH, R.OVL, R.EREQS, R.DFM, R.READP, R.WRITEP, or R.RWP) may result in a call to R.RAFL. If an absolute CM address within a control point's field length has been computed and saved, the address will be invalidated because the control point may have been moved.

SYSTEM MONITOR

The system monitor consists of MTR, which runs continuously in PP0, and CPMTR, which resides in CM and uses the CP intermittently for short bursts. The monitoring tasks are divided between MTR and CPMTR on a functional basis to distribute the work load in the most efficient manner.

MTR STRUCTURE

Unlike CPMTR, MTR is not initiated to perform a specific function. It runs continuously and must keep searching for requests directed to it. The frequency with which it scans for each type of request can have major impact on system efficiency. The following major responsibilities are listed in an order which corresponds approximately to the frequency with which they should be performed.

- Advance system clocks.

  The accuracy of the system clocks in T.CLK and T.MSC is directly related to the frequency with which MTR accesses the real-time clock on channel 14_8.

- Check T.PPIP.

  This is the word into which PP resident writes its input register address when it has a monitor function for MTR. Frequent checking reduces the MTR response time and reduces chances of conflict between two PPs in the use of this word.

- Check T.MTRRS.

  This is the short buffer through which CPMTR passes PP calls taken from RA+1. It is also used for some PP monitor functions called by CPMTR or scheduler (M.SEF or M.ISP). MTR should keep this buffer clear so that it will not inhibit the efficient execution of CPMTR.

- Check individual PP output registers.

  T.PPIP is used for quick attention from MTR on monitor functions. It is still necessary for MTR to scan the output registers because if two PPs make requests in near unison, one will be lost from T.PPIP. Also several monitor functions, such as M.BUFPTR, M.DFM, and M.RCH may not be completed on their first processing. These functions will be processed more quickly if the output registers are scanned more frequently.

- Advance control point.

  Examine each control point in turn to see if a PP program should be initiated from the event stack, if the CPU should be restarted from recall status, or if 1AJ should be called to advance to the next control statement.

- Check RA+1.

  Examine RA+1; if it is nonzero, initiate CPMTR. This function is intended only to initiate CPMTR for a program that does not use the exchange jump capability.

CPMTR ORGANIZATION

CPMTR is the one central program that has no exchange package area of its own. It runs in monitor mode and selects the user mode programs to be run next. When CPMTR is not running, its exchange package is stored in the area reserved for the user mode program selected to be run. The user mode program makes a system request by placing the system call in the word at RA+1 and performing a central exchange jump (XJ) instruction. This reinitiates the execution of CPMTR and saves the register contents of the user mode program in its own exchange package area.

PP programs also can direct system requests to CPMTR. Typically, they use the R.MTR routine of PP resident for such requests. R.MTR places the monitor function in its output register and then determines if the function should be directed to CPMTR or to MTR. When calling CPMTR, the PP input register address is written in T.PPID so CPMTR can identify the calling PP without scanning all the output registers. Then the PP resident routine executes the monitor exchange jump (either MAN or MXN) to initiate CPMTR execution. MAN causes an exchange jump to the address in the CPU's MA register; MXN causes an exchange jump to the address in the PP's A register. When the system is using the MXN (IP.XJ=1), CPMTR maintains a special control word at T.MXNCTL. This word is read and executed by PP resident to ensure that the correct exchange jump address is used with the MXN.

When CPMTR begins, it must determine why it was called. It first checks RA+1 of the user mode program that was running. If RA+1 is nonzero, its content is picked up by CPMTR and RA+1 is cleared. This call is compared against a list of system calls to be performed immediately by CPMTR. If not one of these, the call is placed into the small buffer at T.MTRRS where MTR assigns it to a PP. If the RA+1 call does not have the auto recall bit set, CPMTR immediately returns control to the user program.

If the auto recall bit is set, CPMTR sets that control point into auto recall status and reassigns the CPU to another user program.

The list of system calls performed by CPMTR includes ABT, END, RCL, TIM, XJR, and others.

If RA+1 is empty, T.PPID is checked. If an input register address is in T.PPID, CPMTR clears it and checks the corresponding output register for a function to be performed. The CPMTR functions are described with the other monitor functions.

If RA+1 and T.PPID are both empty, MTR's output register is checked. This extra check is made because MTR does not use T.PPID when it issues a CPMTR function. Otherwise, a function from MTR is handled just like any other PP.

If CPMTR cannot determine why it was called, it returns control to the interrupted user program.

OPERATIONS

The monitor performs operations that must be performed by a program that is permanently resident. Among these operations are

- CPU scheduling.
- Assignment of the PPs.
- Channel reservations.
- Time accounting.
- Storage requests.
- Other operations easily done by a centralized routine.

CPU SCHEDULING

CPMTR assigns the CPU to jobs at control points or to certain system programs that execute in program mode. The CPU status of jobs is controlled by PPs or by the job. CPMTR accepts requests for a change of status and records the current status as a set of bits in the control word (W.CPUST) associated with each exchange package.

The most significant of these bits are

Bit	Description
W	Set by an M.RCP or M.SETST monitor function when a central program is loaded for execution. It remains set until the program posts END in RA+1 or is aborted for any reason.
C and D	Set when the job is being executed in CPU A or B, respectively.
X	The job is in periodic recall status because of an RCL request from the program.
Y	The job is in auto recall status and is not restarted until the requested system function is completed.
Z	The job is suspended because it threatened to saturate the system with PP calls.
M, P, and S	Job execution has been temporarily suspended by storage move, checkpoint, or the job swapper, respectively.

An exchange package exists for each control point, in addition to one for storage move and scheduler, and one for each CPU idle program. An exchange package is ready to use the CPU if W is on and M, P, S, X, Y, and Z are all off. CPMTR assigns the exchange package to the CPU on a priority basis. The priority used is the CPU priority.

The CPU priority is a 6-bit field in the CPU status word. Priority levels, in ascending order, are listed below.

Priority Level	Description
PR.IDLE	Zero level default CPU job to be used only in the absence of any other.
PR.BATCH	Batch jobs initiated by 1IB.

Priority Level	Description
PR.INT	INTERCOM jobs initiated by 1SI.
PR.SCP	System control point jobs initiated by DSD.
PR.SYS	Storage move and scheduler.

When more than one job is at the highest active priority level, the CPU is shared on a round-robin basis. Each uses the CPU for BASESLIC milliseconds before control is passed to the next job.

This combination of priority and round-robin scheduling is overridden during the time that an RMS driver is transferring data to or from a user's buffer. If the user's program is not in auto recall, it is given a slice of CPU time so that it can process the data as it is transferred. In this way, a low priority job may temporarily preempt the CPU away from a high priority job.

When CPMTR assigns the CPU to a job, BASESLIC is added to the current time to produce the projected end-of-slice time. The end-of-slice time is posted in T.CPSTA (or T.CPSTB). When the time arrives, MTR issues an M.SLICE function which causes CPMTR to select the next job for the CPU.

ASSIGNMENT OF THE PPs

PP scheduling is done by MTR. A PP can be requested by a CPU program through a call in RA+1 or by another PP through an M.RPJ or M.EES monitor function.

MTR always reserves at least one PP for the RMS stack processors to ensure that the stack processor is not locked out while all programs in the PPs are waiting for the stack processor to access RMS. More than one PP can be reserved for the stack processor (refer to N.SPRPP in CMR). MTR maintains the following lists of PP calls that are not currently assigned to a PP.

Call	Description
PP job queue	This is the overflow list where PP calls are placed when no PP is available. It is a first in/first out, ordered queue except for stack processors. A PP call that MTR identifies as a stack processor will be added at the front of the list, pushing down all the members already in the list.
Delay stack	This is a list of PP calls for the M.RPJ function with a nonzero time delay. The calls are ordered in sequence of their time delay. When the delay is expired, the calls are removed from the delay stack and assigned to a PP or added to the PP job queue if no PP is available.
Event stack	This is a list of PP calls for the M.EES function. It is searched periodically to find any entries whose event has occurred. When one is found, it is removed from the event stack and assigned to a PP or added to the PP job queue if no PP is available.

MTR keeps the control values for these lists in PP0. The PP calls are kept in the peripheral job table (T.PJT) in CM. Each call consists of the input register and three words for the PP message buffer. When a PJT entry is not in use, the input register word is set to zero.

To control PP assignments, MTR keeps (in PP0) an 8-byte status word for the PP entries which form the PP status table. Each status word has one of two formats, depending upon whether the PP is assigned or unassigned and available.

1	2	3	4	5	6	7	8
CPAD	EXTRA	BUFPTR	APLINK	JUMPAD	PPFLAG	PPSEC	PPMSEC

CPAD	Base address of control point area to which PP is assigned.
EXTRA	Spare byte.
BUFPTR	The low order 12 bits of the buffer pointer the last time that MTR checked it (refer to M.BUFPTR).
APLINK	Active PP link. This is a pointer to the next member in a chain of active PPs. The chain always starts with MTR (PP0) and ends with DSD (PP1). The next PP is identified by its output register index value.
JUMPAD	This is the address saved for re-entry to a partially completed monitor function that has been exited via an RJM MAINLOOP.
PPFLAG	Flag is set when PP contains a stack processor (PP assigned). When idle, PPFLAG contains the link to the next idle PP.
PPSEC	PP starting time in seconds.
PPMSEC	PP starting time in milliseconds.

When the PP is unassigned and available, the PP status word is linked in a chain of unassigned and available PPs, using byte 6 of the status word (PPFLAG). PP direct cell PPIA contains a pointer to the status word at the head of the chain. Byte 6 contains a pointer to the status word of the next available PP. If no more PPs are available, byte 6 contains zero. A chain of three available PPs is illustrated in figure 3-4.

MTR assigns a PP by writing a peripheral job name and a control point number into the PP input register to perform one of the following actions.

- Satisfy a PP program call issued as an RA+1 request.
- Answer a PP request for another PP job (M.EES, M.EESD, M.RPJ, or M.RPJD request).
- Initiate a stack processor when an I/O request is issued for a mass storage device to which no stack processor is currently assigned.
- Call the PP program 1AJ to a control point when all control point activity has ceased.

MTR maintains a PP queue table containing a maximum of 40 entries, each 4 bytes long. Each entry corresponds to a four-word entry in the peripheral job table (PJT) in CMR. In the queue, the following chains are kept.

- A queue of PP jobs that cannot be initiated currently because PPs are not available. This PP job queue is a chain of PP input register images.
- A queue of PP jobs that must be initiated after a given time delay. This queue is a time-ordered chain of PP register images, called the delay stack.

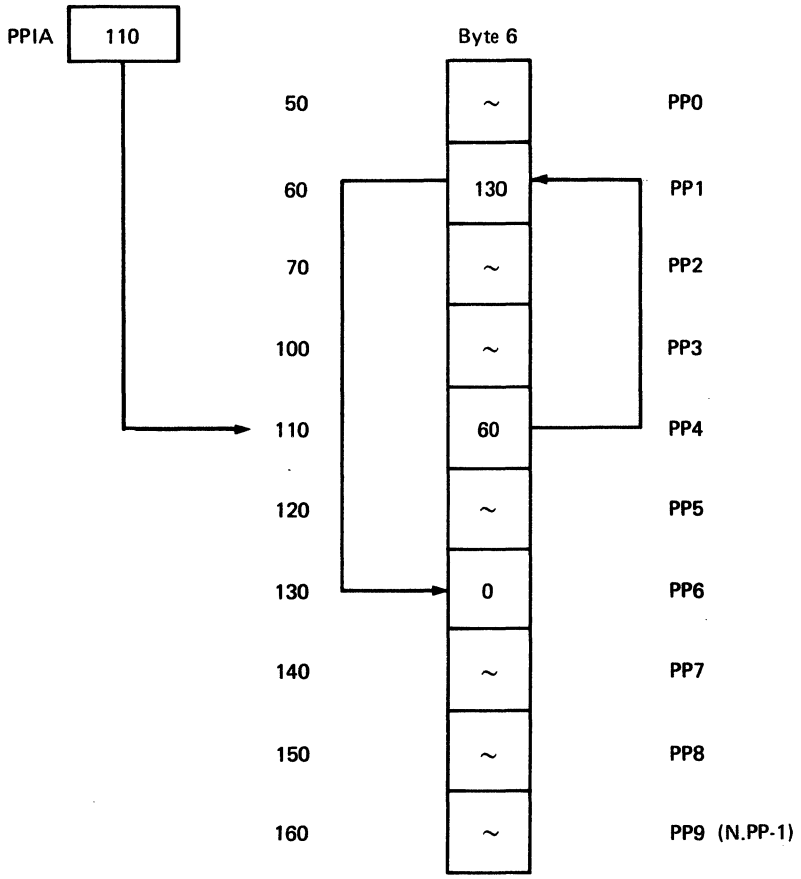


Figure 3-4. PP Chain

- A separate queue of PP jobs for each control point which must be initiated after a specified bit has been set or cleared in CM. This queue is called the event stack.
- An empty queue of all unused entries in the PP queue.

Three pointers in MTR define the beginning of each chain.

NPPQ	PP job queue.
NACT	PP delay stack.
EMPTY	Empty chain.

The pointers to the event stack are in the MTR control point table EVST. Each control point has a separate EVST. A fourth pointer, LPPQ, defines the end of the PP job queue. When the time delay expires for an entry in the delay stack, that entry will be transferred to the end of the job queue.

Chaining of the queue entries uses byte 3 of each 4-byte entry. Bytes 1 and 2 contain the PPFLAG or the maturation time for entries in the delay stack. The end of a chain of entries is signaled by zeros in byte 3. A PP job queue containing a two-entry overload queue, a one-entry delay stack, the empty chain, and two one-entry event stacks is illustrated in figure 3-5.

CHANNEL RESERVATIONS

CPMTR processes channel reservations. The CPU is used for this frequently used function because it is easily accessed from PP resident. If the requested channel is already busy, MTR periodically reissues the request to CPMTR.

TIME ACCOUNTING

MTR uses the real-time clock on channel 14_8 as the source for its time keeping duties. MTR maintains the two basic system time clocks T.CLK and T.MSC. T.CLK is a 24-hour clock that gives the time of day in hours, minutes, and seconds.

Bits 35 through 12 of T.MSC contain the total number of seconds since the last deadstart, expressed as a 24-bit binary number. Bits 11 through 0 are the binary fractional parts of a second. Bits 35 through 0 contain a continuous binary number recording the time as seconds times 4096, which is used as the accounting basis for CPU time, I/O time, and PP time.

CPU time is compiled by CPMTR. Each time the CPU is rescheduled, the current value of T.MSC is recorded so that the elapsed time can be computed the next time the CPU is rescheduled.

MTR accumulates I/O time and PP time, except that portion for RMS devices. This part is accumulated by CPMTR from a PRU count that is passed to it in the M.SPRCL function from the stack processor.

STORAGE REQUESTS

CPMTR processes storage requests using control point n+1 (the system job control point) to execute memory management routines. If storage is not available, the memory management program exits to the integrated scheduler which determines if storage can be made available. Storage allocation and storage moves are described under Control Points, section 2.

When a control point RA and/or FL is to be changed, CPMTR suspends the control point by setting the M bit in the CP status byte and setting the storage move flag. PPMTR is then called. It waits for each PP accessing the control point's field length to clear its field access flag. When all field access flags are clear for the control point, the system job control point is restarted to perform the move. When the move is complete, CPMTR clears the M status bit and restarts the control point that was moved.

MONITOR FUNCTIONS

The following descriptions of the monitor functions are in alphabetic order. The tables in appendix B of this manual list the monitor functions in numerical sequence. Functions with a code number of less than or equal to M.MTRCPU are assigned to CPMTR.

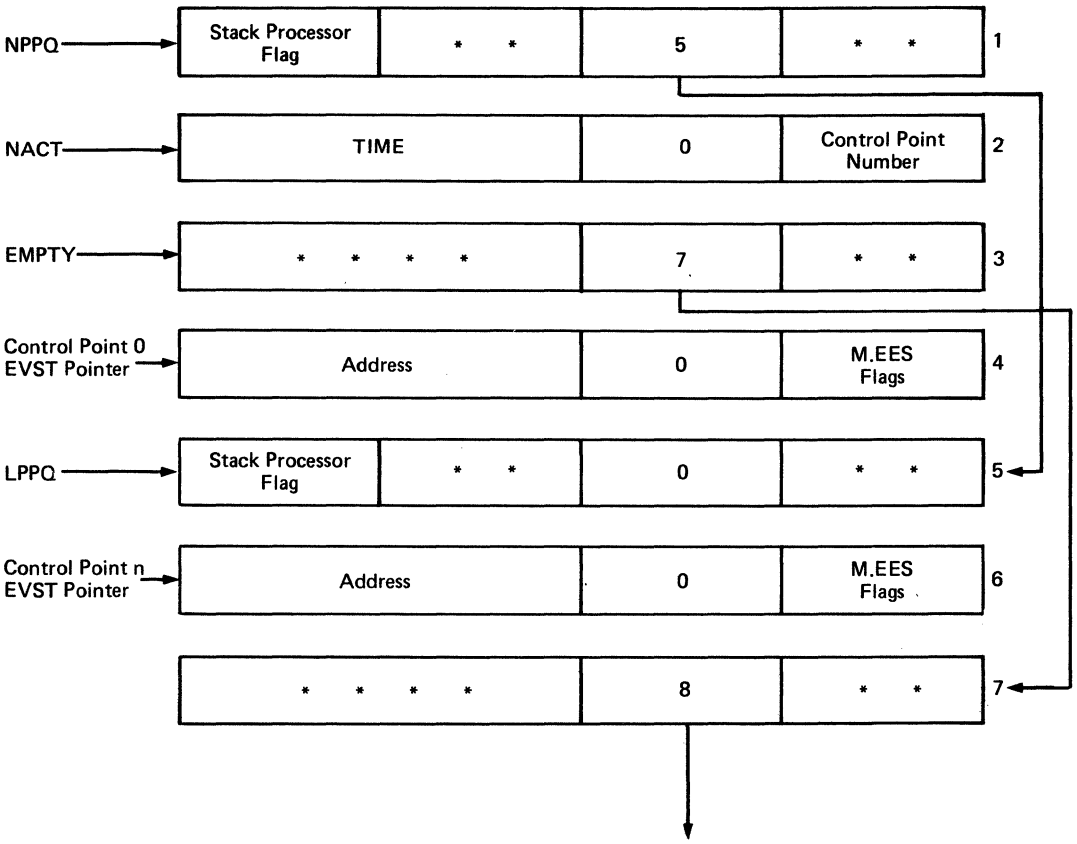


Figure 3-5. PP Job Queue

M.ABORT — ABORT CONTROL POINT AND DROP PP

The format of the function is

M.ABORT,****,****,****,****

The job at the requesting PP is terminated. The requesting processor is responsible for the dayfile messge. Operation of this function is identical with function M.DPP except that the error flag in the control point area is set to F.ERPP to note the abort function.

M.BUFPTR — WATCH BUFFER POINTER WORD

The format of the function is

M.BUFPTR,****,****,00AA,AAAA

AAAAAA Buffer pointer address.

I/O drivers use this function to give MTR the absolute address of the buffer pointer that is being updated. MTR monitors the value of that pointer and when it changes, restarts the control point if it is in periodic recall.

M.CCPA — CHANGE CONTROL POINT ASSIGNMENT

The format of the function is

M.CCPA,****,****,****,**NN

The requesting PP is released from its current control point assignment as if it had issued an M.DPP function, but its input register is not cleared. The PP is assigned to control point NN, and the new control point number inserted in its input register. It is the responsibility of the requesting PP to alter D.CPAD. R.TAFL should also be used to clear the field access flag at the old control point.

M.CLRST — CLEAR STATUS

The format of the function is

M.CLRST,BBBB,****,****,00NN

BBBB Pattern of bits to be cleared.

NN Control point number (only is in MTR output register).

Called to clear CP status bits in byte C.CPSTAT in control point linkage; causes linkage to or delinkage from chain of control points actively waiting for a CPU.

M.CPJ — CAPTURE PERIPHERAL JOB

The format of the function is

M.CPJ,00XX,XXXX,****,****

XXXXXX Buffer address, relative to RA.

This request is issued to find a job for a control point either in the event stack or in the PP delay stack. The event stack is searched first; if a job is found, its data is written to the buffer whose address is given in the request. When the end of the delay stack is reached, the function is completed.

M.CPUST — CHANGE CPU STATUS

The format of the function is

M.CPUST,000X,****,****,****

X	Description
0	If either CPU is off, it is turned on. If the CPU was locked off at deadstart time, it remains off.
1	If both CPUs are on, CPU-A is turned off.
2	If both CPUs are on, CPU-B is turned off.

When the requested function cannot be performed, no action is taken.

M.DCP — DROP CENTRAL PROCESSOR JOB

The format of the function is

M.DCP,****,****,****,****

Execution of the CP job at control point is stopped. The control point status bits W, X, Y, and Z are cleared. The control point status bits set prior to M.DCP are returned in byte 1 of the output register of the requesting PP.

M.DFM — PROCESS DAYFILE MESSAGE

The format of the function is

M.DFM,FFFF,MMMM.****,****

Dayfile flag bits FFFF determine message handling.

FFFF	Description
0	Do not send to B display.
1†	Do not send to control point dayfile.
2††	Do not send to system dayfile (no A display).
3	Flag as an accounting message.
4†	Send to hardware error file.
5††	Do not insert job name in system dayfile.

† If bits 1 and 4 are set, the message is not sent to an INTERCOM control point dayfile but is sent to other control points.

†† If bits 2 and 5 are set, the time is omitted and replaced with blanks in the control point dayfile. This option identifies messages from a task that was executed on a different mainframe.

When MMMM is larger than the address of PPOR, MMMM is taken to be the LWA+1 of the message in the PP message buffer. When MMMM is smaller than the address of PPOR, it is taken to be a dump index for a requested dayfile dump.

Value of dayfile dump index:

Value	Description
0	System dayfile dump.
1 through N.CP	Control point dayfile dump.
N.CP+1	Hardware error file dump.

M.DPP — DROP PP

The format of the function is

M.DPP,FFFF,****,****,****

MTR clears the PP control assignment (the PP status word and the PP input are cleared). If the value of FFFF represents M.DPP, the PP time is not incremented.

M.EES — ENTER EVENT STACK

The format of the function is

M.EES,00AA,AAAA,****,SYTT

AAAAAA	Word address in event stack.
Y	Byte address in word AAAAAA.
TT	Bit address in byte Y.
S	Combined value of F and B.

Value	Description
F is 0	Event stack job assigned when bit is off (F.ESOFF).
F is 4	Event stack job assigned when bit is on (F.ESON).
B is 0	AAAAAA is an absolute address (F.ESABS).
B is 1	AAAAAA is relative to RA (F.ESREL).
B is 2	AAAAAA is a control point area address (F.ESCPA).

This function is used to call a PP program after a specified event has occurred. That event must be defined as a specific bit being set or off. The bit is defined by the parameters in the output registers. W.PPMES1 contains the input register image of the program that is to be assigned when the event occurs. The contents of W.PPMES4, W.PPMES5, and W.PPMES6 are also saved and set in the message buffer when the program is called. This function will not complete if the peripheral job table is full. If possible, use M.EESD instead of M.EES.

M.EESD — ENTER EVENT STACK AND DROP PP

The format of the function is

M.EESD,00AA,AAAA,FFFF,SYTT

Combines the functions of M.EES and M.DPP. This function is completed even if the peripheral job table is full. If FFFF=M.EESD, the control point is not charged for the PP time. All other parameters are identical to M.EES.

M.ICE — INITIATE CENTRAL EXECUTIVE

The format of the function is

M.ICE,PPPP,PPPP,PPPP,EX.xxx

EX.xxx is a subfunction to be performed. These subfunctions are listed under Monitor Functions in appendix B. The PPPP fields can be used as parameters to the subfunction. Refer to the section on INPUT/OUTPUT for a detailed description of the 819 subsystem flush function (EX.SUB).

M.ISP — INITIATE STACK PROCESSOR

The format of the function is

M.ISP,000X,000Y,****,CCCC

CCCC	DST ordinal of stack processor to be initiated.
X is 0	Initiate 1S5 only if PP active flag is zero.
X is not 0	Initiate 1S5 regardless of PP active flag setting.
Y is 0	Initial assignment of 1SP to the DST entry.
Y is 1 (or 2)	A partner call.

A check is made to see if a PP is assigned to this DST ordinal. If there is none, a check for an available PP is made. If an available one is found, set the PP active flag and place the DST ordinal and 1S5 in its input register. If a PP is found to be assigned to the DST ordinal, the setting of X determines if 1S5 is to be initiated or not. If not set, the output register is cleared and an exit made; if set, proceed as for an available PP. If the PP job stack is full, the routine is exited. The 1S5 program is the PP input register DST ordinal checker and stack processor loader. When Y is 1 or 2 and X is not 0, this is a call from 1SP for a partner to work together for a dual access device.

M.KILL — BAD FUNCTION REQUEST

The format of the function is

M.KILL,****,****,****,****

MTR flags the function request as bad and automatically enters STEP 0 mode. The requesting PP is hung.

M.MFLA — MONITOR FIELD LENGTH ACCESS AT CONTROL POINT

The format of the function is

M.MFLA,****,****,****,CCCC

CCCC Address of control point being moved.

This function is used by CPMTR to wait for field access flags at a control point being moved to clear. When all field access flags have been cleared, PPMTR restarts the system job control point (n+1) to perform the move (refer to Storage Requests in this section).

M.NOTE — NULL FUNCTION

The format of the function is

M.NOTE,****,****,****,****

This is a null function used with STEP mode as a breakpoint. During normal execution, MTR clears the output register of the requesting PP.

M.NTIME — ENTER NEW TIME LIMIT

The format of the function is

M.NTIME,TTTT,T***,****,**NN

A CP job time limit of TTTTT seconds is entered at the control point. Any previous time limit is superseded. If the requesting PP is assigned to control point 0, the parameter NN gives the number of the control point to be considered; in any other case, this parameter is irrelevant.

M.PASS — PPMTR IGNORES FUNCTION REQUEST

The format of the function is

M.PASS,****,****,****,****

Indicates a no-operation by PPMTR which is cleared by another routine.

M.PATCH — INSERT A PATCH IN PPMTR

The format of the function is

M.PATCH,AAAA,BBBB,CCCC,DDDD

The routine inserts a patch in the monitor program at the address indicated.

AAAA Insertion address for patch BBBB.

CCCC Insertion address for patch DDDD.

M.PPLIB — PP LIBRARY SEARCH FUNCTION

The formats of the function are

```
M.PPLIB,AAAA,AA**,LLLL,****
M.PPLIB,000X,****,****,****
```

This function is used by PP resident routines only. Its initial use is to locate and set up the loading of a PP overlay.

AAAAAA	Name of overlay.
LLLL	PP load address.

Subsequent use of the function is to control the loading process.

X	Description
0	Release system circular buffer and DDP.
1	Continue loading SCB.
2	No DDP code; default to disk copy.
3	No ECS code; default to disk copy.
4	DDP block transfer error; start error recovery process.
5	Report DDP recovery error.
6	Report completion of error recovery process.
7	Load next overlay to continue error processing.
10	Obtain event number for use by 1SP/1SQ in logging DDP errors.
11	Report 1SP/1SQ DDP error.
12	Report STL channel error.

M.RACT — REQUEST CONTROL POINT ACTIVITY

The format of the function is

```
M.RACT,**NN,IIII,****,****
```

This request provides the various activity counts of control point NN at a given time (NN cannot be zero). If IIII is nonzero, the pseudo activity count is incremented or decremented by the constant IIII (after sign extension). Monitor replies through the PP output register.

Byte 1	Control point status byte.
Byte 2	General activity count.

Byte 3	Count of outstanding delayed PP requests.
Byte 4	Pseudo activity count.

M.RBTSTO — REQUEST RBT STORAGE

The format of the function is

M.RBTSTO,SSSS,****,****,****

CPMTR sets SSSS*100 as the new RBT starting address. If the request cannot be honored, the old RBT starting address is returned in SSSS.

M.RCH — REQUEST CHANNEL RESERVATION

The format of the function is

M.RCH,BBAA,DDCC,***G,RRRR

AA	First choice channel number.
BB	Second choice channel number.
CC	Third choice channel number.
DD	Fourth choice channel number.
G is 0	Normal charge for channel time.
G is 4	Do not charge control point for channel time.
RRRR is 0000	Request immediate reply.
RRRR is not 0000	No reply until a requested channel has been reserved.

When channel zero is requested, it must be field AA. Zero BB, CC, or DD implies no more choices. If none of the requested channels is available, PPMTR sets byte 0 of the PP output register to 0. When a channel is granted, its number is returned in the PP output register byte 1 (location of AA) and byte 4 is set to a value not 0. Thus, programs that request an immediate reply must check that byte 4 is not 0 before using the channel.

On exit, if a channel has been reserved, the output register appears.

0000 XXXX TTTG TTTT YYYY

XXXX	Channel number.
TTTG TTTT	Information from the channel status word, where G is the charge/no charge bit for channel time.
YYYY	PP input register address.

M.RCLCP — RECALL CENTRAL PROGRAM

The format of the function is

M.RCLCP,****,****,****,****

This request is effective only if the central program associated with the requesting PP is in recall status, and no error flag is set at the control point. The status of the control point is set to waiting (W). In any other case, the status of the control point is not altered.

M.RCP — REQUEST CENTRAL PROCESSOR

The format of the function is

M.RCP,****,****,****,****

This request is ignored under the following conditions.

- Requesting PP is assigned to control point 0.
- Error flag is set for the control point.
- Job is already in the waiting status.

If none of these conditions exist, CPMTR sets the job in waiting status (W).

M.RPJ — REQUEST PERIPHERAL JOB

The format of the function is

M.RPJ,SSSS,FFFF,****,****

This function requests that another PP program be initiated after a specified time delay. The first word of the requesting PP message buffer contains the input register image of the new PP program. The time delay is SSSS seconds plus FFFF/$10\,000_8$ seconds. If the time delay is zero and no PP is available, the request is entered in the PP job queue. If no space is available in the PP job queue buffer of PPMTR, the entire request remains pending until a queue entry becomes free. M.RPJD should be used in preference to M.RPJ whenever possible.

M.RPJD — REQUEST PERIPHERAL JOB AND DROP PP

The format of the function is

M.RPJD,DDDD,DDDD,FFFF,****

Combines the functions of M.RPJ and M.DPP. This function will be completed even if the peripheral job table is full. If FFFF=M.RPJ, the control point is not charged for the PP time. The use of the time delay is the same as for M.RPJ.

M.RSTOR — REQUEST STORAGE

The format of the function is

M.RSTOR,CCCC,XXXX,00TT,****

CCCC	Request CM/100 octal.	
XXXX	Request ECS/1000 octal.	
TT	00	CM request only.
	01	ECS request only.
	02	CM and ECS request.
	04	Request CM - awaits response.
	06	Request CM and ECS - awaits response.
	07	IP.POSFL requested by swapper.

This function assigns CCCC central memory and/or XXXX extended core storage to the control point of the requesting PP. Monitor replies to this request by setting CCCC and/or XXXX to the values actually assigned to the control point and by setting byte 0 to zero. These values should be compared with the original values requested to determine whether these requests have been honored or not. A request for more storage is rejected if not enough storage is available or if a storage move is already in progress. A request for less storage is always honored. If TT is 02 or 04, CPMTR can honor part of the request without honoring the remainder.

M.SCB — SYSTEM CIRCULAR BUFFER SURVEILLANCE

The format of the function is

M.SCB,****,00BB,BBBB,EX.CBM

BBBBBB System circular buffer address.

The system circular buffer is an FET-like table that has a trigger and a direction flag in addition to FIRST, IN, OUT, and LIMIT. MTR uses IN, OUT, and the trigger and direction (RMS-to-ECS or ECS-to-RMS) to determine if a threshold has been reached. If not, no action is taken. If so, MTR issues an M.ICE/EX.CBM function to start CBM for processing of the system circular buffer.

M.SCH — INITIATE INTEGRATED SCHEDULER

The format of the function is

M.SCH,000X,00CC,00JJ,JJJJ

When X is 2, the contents of the output register are placed into a buffer at T.SCHRS for the integrated scheduler. When the integrated scheduler processes this request, it links the JDT at location JJJJJJ to the job queue for JCA ordinal CC. If CC is zero, the job class is taken from the JDT.

When X is not 2, the contents of the output register are not passed to the scheduler.

In both cases, CPMTR initiates the scheduler immediately.

M.SEF — SET ERROR FLAG

The format of the function is

M.SEF,**NN,EEEE,****,****

Monitor drops the central program at control point NN by putting the program in zero status, and setting the error flag to the value EEEE.

The M.SEF function recognizes two special control values in the error flag field that are used to initiate and terminate the memo mode.

When a control point is in memo mode, error flags are not set in byte C.CPEF(1), but are recorded in bits 5 through 0 of byte C.CPMEMO(0). The high order bits 11 through 6 of C.CPMEMO are set on when the control point is in memo mode, and are used by MTR to recognize the mode.

F.ERMEMO (-77_8) initiates memo mode. Bits 11 through 6 of C.CPMEMO are set on. Bits 5 through 0 of C.CPEF are moved to bits 5 through 2 of C.CPMEMO and C.CPEF is cleared. If the control point is already in memo mode, the effect of the F.ERMEMO is to clear an error flag memo without terminating the memo mode.

F.ERTMM (-0) terminates memo mode. Bits 5 through 0 of C.CPMEMO are moved to C.CPEF and C.CPMEMO is cleared. Error flag zero can also terminate memo mode; it clears both the error memo and the error flag fields. If an error memo is already recorded when the F.ERTMM is issued, the memo is made an error flag causing the normal error flag processing to take place.

When a control point is in memo mode, the M.SEF function with error flag values 1 through 77_8 does not cause the CPU to be dropped as when in normal error flag mode; however, there are some exceptions. The following error codes are caused by errors in the central program and render the CPU useless.

2	F.ERAR	Arithmetic error.
4	F.ERCP	CPU abort (ABT in RA+1).
5	F.ERPCE	PP call error (garbage in RA+1).
15	F.ERRCL	Auto recall error.

Any of these codes cause the control point to revert to normal error flag mode.

When entering memo mode, it is possible that an error flag had been set just prior to the processing of the F.ERMEMO. The PP program that initiates memo mode should immediately check the error memo field after completion of the F.ERMEMO. If an error memo is set, it should be assumed that it occurred as an error flag prior to the F.ERMEMO. Usually the best action at this point is an F.ERTMM. Since the program is not yet committed to its critical stage, it is best to allow the error flag processing to continue.

Memo mode is restricted to use during single control statement executions only. It is the responsibility of the program that initiates memo mode to terminate it. If 1AJ finds a control point in memo mode, it is processed as an error flag.

M.SEQ — ASSIGN JOB SEQUENCE NUMBER

The format of the function is

M.SEG,****,****,****,****

Monitor returns in byte 1 of the PP output register a job sequence number (in display code).

M.SETST — SET STATUS BITS

The format of the function is

M.SETST,BBBB,****,****,00NN

BBBB	Pattern of bits to be set.
NN	Control point number (only if in MTR output register).

Called to set CP status bits in byte C.CPSTAT in control point NN area; can cause linkage to or delinkage from chain of control points actively waiting for a CPU.

M.SLICE — TERMINATE TIME SLICE PERIOD

The format of the function is

M.SLICE,****,****,****,****

Only the PPMTR can issue this function request. It is issued to interrupt an executing user mode program so that CPMTR can reschedule the use of CPUs.

M.SLPER — INITIATE CENTRAL MONITOR IN OTHER CPU

The format of the function is

M.SLPER,****,****,****,***C

C CPU in which CPMTR is to execute next instruction.

0 Initiate CPMTR in CPU-A.

1 Initiate CPMTR in CPU-B.

This request is used by CPMTR to cause the next execution to take place in the opposite CPU of a dual CPU system. If CPMTR is executing in CPU-A and this request is made with C set, the next execution of CPMTR takes place in CPU-B. A possible use would be to terminate a job executing in CPU-B while CPMTR is executing in CPU-A.

M.SPM — SPM CALL FROM 1SP

The format of the function is

M.SPM,PPPP,PPPP,PPPP,EX.xxx

The PPPP fields are subfunction parameters. EX.xxx is the subfunction to be performed as follows:

EX.SPRCL Stack processor recall. SPM is called to terminate the actual I/O portion of the current stack request. SPM will terminate, reissue, or otherwise further process the stack request and issue a new stack request or special order (O.IDLE, O.DROP, O.SEEK) to 1SP and complete the function. Call format is

M.SPM,****,****,****,EX.SPRCL

EX.STAT Changes status. SPM is called to change 1SP status in the DST and take appropriate action. DST status is

0 No PP assigned.

1 PP assigned but not ready; PPIR - PP assigned and ready.

When PPMTR assigns 1SP to a PP, it changes the DST status from zero to one. After initialization, 1SP issues an EX.STAT with PPIR status. This call may be issued again later to activate SPM if 1SP is idle and the PPMTR stack processor drop flag is set. A pending EDITLIB or LDCMR will set a wait flag for 1SP. When 1SP is idle and encounters the flag, 1SP issues a status of one to SPM and reinitializes itself. When 1SP is idle and detects an outstanding channel request for its channel, it issues a status to zero to SPM to request a drop. SPM will then issue an O.DROP to 1SP so that 1SP can give up the PP. Call format is

M.SPM,0000,0000,SSSS,EX.STAT

SSSS	Status
0	Request drop.
1	Request inactive status (reinitializing).
PPIR	Request active status (initialized).

EX.NXTPB Get next PB/PRU. This is a time-critical call made during the I/O transfer. This call is entered by 1SP into its PPOR just prior to starting transfer of the current PB/PRU chunk of data. While the current chunk is being transferred, PPMTR sees the PPOR call and initiates SPM. If the current transfer is a write, SPM will allocate more RBs, if needed. In any case, SPM then converts the current RB position in the RBT chain to a PB position and stores this in PPMES4, bytes 0 through 2 of the 1SP communication area. When 1SP completes the current PB/PRU transfer, it updates PPMES6 (current PB/PRU) from PPMES4 (next PB/PRU), issues the next EX.NXTPB call to the PPOR, and continues the transfer. Call format is

M.SPM,2,SSPP,PPPP,EX.NXTPB

SSPP,PPPP	Successor Call Type
PPPP is 0	No successor call (clear PPOR).
PPPP is not 0 SS is 0	Set up M.BUFPTR call.
PPPP is not 0 SS is not 0	Set up M.SCB call PPPP; successor call parameters.

M.SPRCL — STACK PROCESSOR RECALL

The format of the function is

M.SPRCL,****,****,000F,CCCC

CCCC Control point area address.

F Modification of the count of outstanding stack requests in W.CPSR.

F	Description
0	No adjustment.
1	Subtract one.
2	Add one.

M.TRACE — ENTER MONITOR TRACE MODE

The format of the function is

M.TRACE,AAAA,FFFF,NNNN,****

AAAA Absolute address of buffer within requesting field length of requesting job.

FFFF Length of buffer.

NNNN Pointer to next available word-pair in buffer.

A buffer must be provided by the trace mode requestor into which this PPMTR function will write trace records. Each record is a two-word entry containing function and PP status information. This function is reserved for Control Data developmental use.

M.TSR — TERMINATE STORAGE REQUEST

The format of the function is

M.TSR,****,****,****,****

Request is valid if real-time monitor is installed (IP.RTMTR is nonzero); it terminates wait period involving an M.RSTOR request.

FILES AND FILE TABLES 4

FILES

A name associated with each file identifies it to the system and to the user. Files are uniquely known by the file name, source or destination ID, and the terminal ID (if applicable). Files are stored on either allocatable or nonallocatable devices. Rotating mass storage (RMS) units, such as disks, are allocatable because files on these devices may be allocated to more than one control point. Other devices, such as magnetic tapes, card readers, punches, and line printers are nonallocatable because they can process only one file at a time.

Files associated with a job running at a control point are assigned or attached to a control point. Files not associated with running jobs are assigned to control point zero.

Files are associated with one of the following groups: system, local, permanent, and queue (input and output). The following paragraphs describe briefly each of the file groups.

SYSTEM FILES

The following files are always in the system. They are always assigned to control point 0 and they reside on allocatable devices. These files, except for the job dayfiles, are maintained on system devices as permanent files. Each job dayfile exists only for the duration of the job.

File	Description
System	The file has the name of ZZZZZ04 and contains a copy of the deadstart system tape.
System dayfile	The system dayfile contains a complete record of all activity in the system. Normally, when a message is sent to any job dayfile, it also is sent to the system dayfile. At intervals, the system dayfile can be dumped to a line printer, punch, or magnetic tape.
Job dayfile (DFILEn)	Each user control point in the system has a job dayfile; n is the control point number. These files are assigned to control point 0 and a user cannot access them directly. When a user job terminates, the content of the job dayfile is copied to the end of the job output file. A job dayfile contains images of all control statements processed, appropriate system messages concerning the job run, plus messages sent to the dayfile by the job.
CERFILE	If hardware errors are discovered by running programs, a message is written to this file. Periodically, the file is dumped for examination by customer engineers so they can take remedial action.
ZZZZCMR	This file contains the absolute segments of CMR in an ECS system. Depending on the options selected, this file is used by LDCMR at deadstart or when the system is reloaded.

File	Description
ZZZZZ06	If the installation parameter IP.ELIB is one, this file is created to contain the ECS library.
ZZZZZ23	This file contains a copy of the CM resident library area. It is created at deadstart and is updated by EDITLIB. It is not permanently attached to control point 0 and is used only for deadstart recovery.

PERMANENT FILES

Permanent files are saved across deadstarts and are therefore considered permanent to the system. Controls over file access and mode of use are provided to define various degrees of privacy. When a permanent file is created, the privacy defined determines which user can access it and the kind of processing allowed.

LOCAL FILES

Any files, other than permanent files, attached to a job running at a control point are local files. They may be on allocatable or nonallocatable devices.

Local files assigned to a control point must have unique names. Two local files named INPUT and OUTPUT are associated with each job. INPUT contains the job file; the job name is changed to the name INPUT when the job is assigned to a control point. OUTPUT is assigned to the job when the first reference to it occurs. It has a disposition code which indicates the job output is to be produced on peripheral devices in the area from which the user submitted the job.

When a local file is detached from a control point, disposition depends on its control point, disposition code, and device type on which it resides. For local files on nonallocatable devices, the device and the related table space will be released. Assigned storage and related table space is released for local files on allocatable devices, other than private disk packs, having a zero disposition code. Files with the special names OUTPUT, PUNCH, PUNCHB, P80C, FILMPR, FILMPL, HARDPL, HARDPR, and PLOT are assigned nonzero disposition codes when created. All other files are assigned a zero disposition code when created. For local files with other disposition codes the file name is changed to the job name and the file is assigned to control point 0. A local file with the name PUNCH or PUNCHB is output to a card punch; the file OUTPUT is printed.

QUEUE FILES

To provide for the recovery of input and output files from disk tables on nonrecovery deadstarts, the input and output queues are kept as permanent files. All input files that enter the system via JANUS, INTERCOM, Remote Batch processor, or load tape (n.X TLOAD) are automatically cataloged. They are not purged until the job has completed execution and all output files of the job have been cataloged. When the permanent file catalog (PFC) becomes full, a message is issued to the operator and input halts until there is more space for files to be cataloged.

The file name table entry of queue files contains permanent file information and the file description parameters. 1TJ is the common routine for entering a file into the input queue called by JANUS, INTERCOM Import processors, and DSP for ROUTE(filename,DC=IN). 2VJ verifies the job statement parameters. 1QF is the PP routine which catalogs and purges queue files.

Input Queue

Jobs may enter the system from sources such as card reader, magnetic tape, or remote devices. In every case, a job file is read by a system package operating at a control point which then writes the job to a local file on an allocatable device. When the file has been written, its entry in the file name table is altered to indicate an input disposition code; the file is cataloged and released to control point 0 as an input file. The input queue consists of all files assigned to control point 0 with an input disposition code.

The system packages that read in batched local jobs ensure that each job file in the system has a unique seven-character name. The job name from the job statement is truncated to the first five characters (or extended with zeros to five characters) and two unique sequence characters are added. All numerals and letters can be used as sequence characters; therefore, 1296 sequence combinations are possible for a single five-character job name. Even though unique combinations are exhausted, duplication of names is not significant unless the earlier job has not been completely processed when the duplicate enters the system.

Output Queue

Output files originate from local files on allocatable devices; they have nonzero disposition codes. When a job terminates, such files are cataloged, assigned to control point 0, and given either the job name or the name in the file ID field of the FNT file routing supplement. These files then form a system output queue which is, essentially, a list of files waiting to be output to unit record equipment.

In each output queue file name table entry, fields define the destination of the file. The characteristics that are defined are device type, terminal ID, destination ID, external and internal characteristics codes, disposition codes, and forms codes.

Local files can be put into the output queue as follows:

- The user gives the file a special name. When a file with a special name is created, it receives a nonzero disposition code. These files are sent to the corresponding destination when the file is released for output processing. For example, the file named OUTPUT receives a print disposition code. A file named PUNCH receives a punch disposition code.

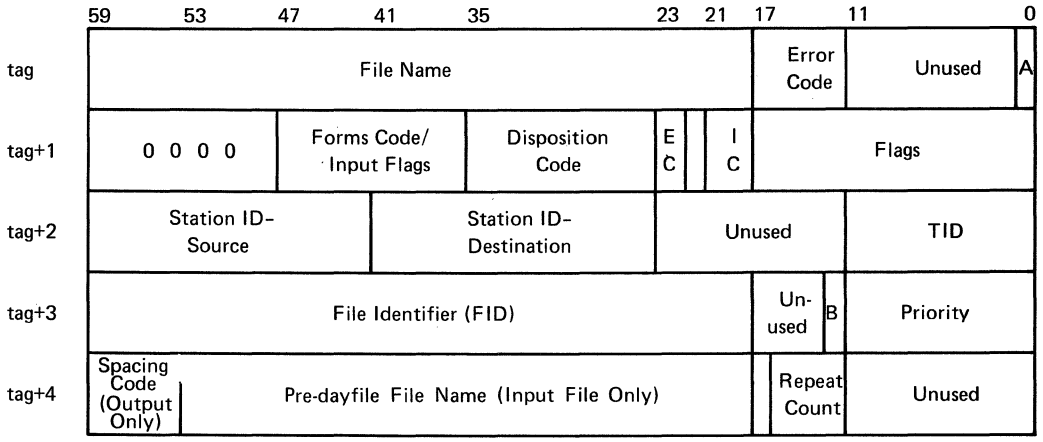
- The user can specify file disposition with a DISPOSE or ROUTE control statement or macro. The file can have any name. Files must reside on allocatable devices that are members of the queue set.

Files in the output queue must be on allocatable devices. A file is put into the output queue when the job terminates or when a CLOSE,UNLOAD, or CLOSE,RETURN is performed Since the name of an output queue file can be the name of the job which created it, and since a job can create several files which go into the output queue, names in the output queue often are not unique.

ROUTE Macro — Additional Capabilities

A system job using the ROUTE macro has additional capabilities that a user job does not. A system job can specify a source ID, a seven-character job name, and a predayfile file name. The other ROUTE capabilities are the same as for user jobs. The ROUTE control statement and macro are described for user jobs in the NOS/BE Reference Manual. The remainder of this discussion describes only the differences for system jobs.

The parameter block that must be set up to use the additional capabilities for system jobs has the following format.



The fields in the parameter block are identical to those described for the ROUTE macro in the NOS/BE Reference Manual with the following exceptions.

Word	Bits	Field	Description
tag+1	47-36	Forms code/input flags	If the file is to be routed to an input queue, a value of 44 indicates a file identifier (FID) of seven characters is specified. This value is ignored if the job is not a system job.
	17-0	Flag bits	Indicates which parameters are specified (in addition to the values given in the NOS/BE Reference Manual). 13 Dayfile is attached for immediate routing to output. 11 Predayfile file name is specified. 5 System uses FID specified in tag+3, bits 59-18. Only system jobs can specify seven characters; user jobs specify five characters.
tag+1	59-42	Station ID-source	Three display code characters used as the source ID for a job routed to an input queue. When this field is binary 0, the routed file has no source ID. When DC=IN, the job's source ID is used as the setting of this field. A job's source ID is found in the control point area.
tag+4	59-18	Predayfile file name	Name of the file which contains the predayfile. This parameter is meaningful only for DC=IN.

ACQUIRE Macro

The ACQUIRE macro calls the PP routine QAF to search the input, print, punch, special (nonstandard) output, and execution queues looking for entries that satisfy given selection criteria. The user specifies one of four functions specified by a function code in bits 3 through 1 of word 0 in the ACQUIRE parameter list: ALTER, modify queue entries; GET, attach a file to the caller's control point; PEEK, return information about the queue entries; or COUNT, count the entries in the specified queue(s). QAF can be called only by a routine resident in the system library. The format of the ACQUIRE macro is

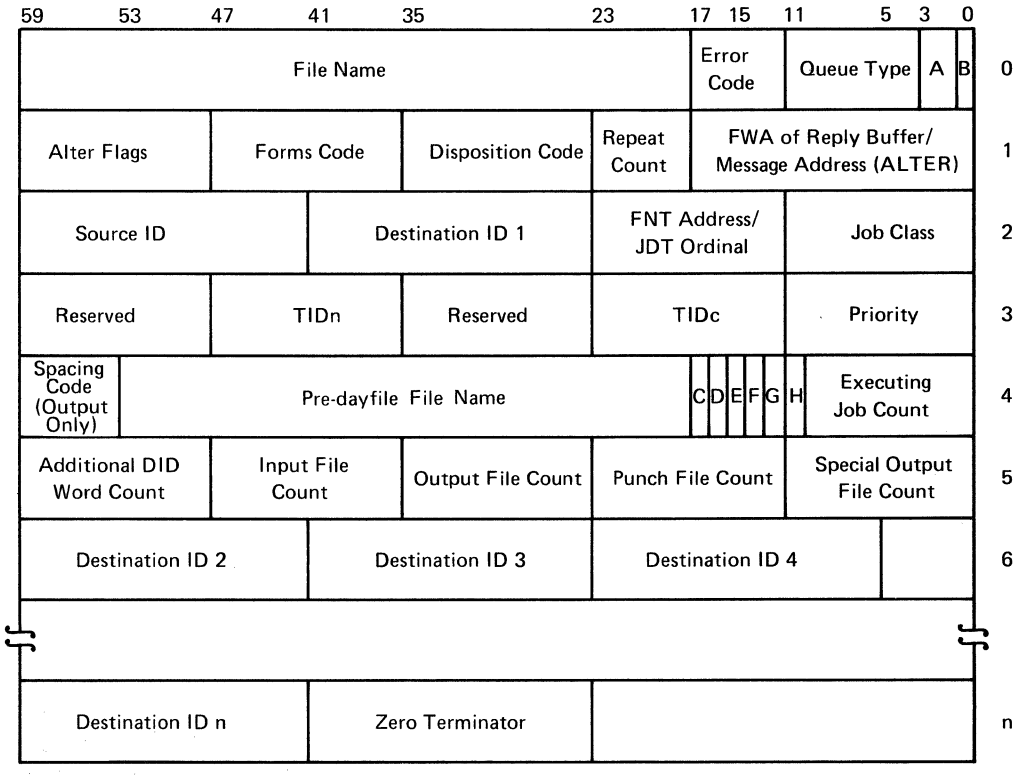
ACQUIRE addr,recall,N.

addr The address of the first word of the parameter list.

recall Optional parameter specifying auto recall.

N Required parameter to distinguish this new macro from an older version.

QAF requires the parameter list to be at least six words in length. The list can be longer if there are additional destination IDs. The additional length is specified in the additional destination ID word count field.



A Function code.
B Complete bit.
C Predayfile bit.
D Class 2 input file inhibit bit.
E Class 1 input file inhibit bit.
F Inhibit duplicate file name search.
G Reserved.
H Reserved.

Word	Bits	Description
0	59-18	File name. If the file name of a particular file is specified on any function, each FNT entry is examined until the specified file name is found. This file must meet the specified criteria to qualify as the selected file.
	17-12	Error code (in octal).

Code	Significance
01	Invalid queue type.
02	No queue entry found with specified parameters.
03	Function prohibits 7777_8 priority.
04	No FNT space.
05	Invalid reply entry buffer address.
06	Internal QAF error on FNT address.
07	Illegal request.
10	Too many extra DID words.
11	PEEK requires single queue type.
12	Duplicate file name on GET.
13	Count of 0 is invalid.
14	LFN needed for file having predayfile.
15	Invalid FNT address/JDT ordinal.

11-4 Queue type. The queue type must be supplied on all functions except the special PEEK function when it must be 0. The binary values are: 00000001 (INPUT), 00000010 (OUTPUT), 00000100 (PUNCH) or 00001000 (special output), or 00010000 (execution).

INPUT file	Has a valid FNT entry, is unlocked at control point 0, and has a disposition code of 04_8 (INPUT job), 05_8 (INPUT tape job), or 06_8 (INPUT tape job on P display).
OUTPUT file	Has a valid FNT entry, is unlocked at control point 0, and has a disposition code of 40_8 (any 512 or 580 line printer), 42_8 (any 512), 43_8 (any 580-12), 44_8 (any 580-16), or 45_8 (any 580-20).

Word	Bits		Description
		PUNCH file	Has a valid FNT entry, is unlocked at control point 0, and has a disposition code of 10_8 (PUNCH 026 set from display code).
		Special output file	Has a valid FNT entry, is unlocked at control point 0, and has a disposition code of 20_8 (film print), 22_8 (film plot), 24_8 (hardcopy print), 26_8 (hardcopy plot), or 30_8 (plot).
		Executing job	Has a valid JDT entry.
	3-1	Function code (in octal).	
		0 ALTER	
		1 GET	
		2 PEEK	
		3 COUNT	
	0	Complete bit. The complete bit must be cleared before any call to QAF. The bit is set on completion of any function.	
1	59-48	ALTER flags.	
		Bit Set	Significance
		53	Abort and/or evict job and/or file.
		52	Change repeat count.
		51	Change forms code; for other functions, means compare forms codes.
		50	Change priority.
		49	Change terminal ID to TIDn.
		48	Send to central site.
	47-36	Forms code.	
	35-24	Disposition code.	
	23	Not used.	
	22-18	Repeat count.	
	17-0	FWA of reply buffer or a message to be issued when aborting a job. Limited to 30 characters.	

Word	Bits	Description
2	59-48	Source ID (SID).
	41-24	Destination ID 1 (DID).
	23-12	FNT address/JDT ordinal. This field in the parameter list is an absolute FNT address whenever a queue type other than job queue is specified. It is a JDT ordinal whenever only the job queue is specified. The field is an FNT address and is required on a special PEEK function and optional on all other functions.
	11-0	Job class.

Bit Set	Significance
5	Graphics job.
4	Express job.
3	Multiuser job.
2	INTERCOM job.
1	Batch job with nonallocatable device requirement.
0	Batch job without nonallocatable device requirement.

3	59-36	TIDn. New TID (used by ALTER only).
	35-12	TIDc. Current Terminal ID (used for search).
	11-0	Priority. If 0 priority is specified and the other criteria are satisfied, GET attaches the first file found; PEEK writes a reply entry for each file; and COUNT increments the file type count. If the priority is greater than 0 and less than 7777_8 and the other criteria are satisfied, GET attaches the first file having a priority greater than or equal to the specification; PEEK writes a reply entry for each file that has a priority greater than or equal to the specification; and COUNT increments the file type count when a file has a priority greater than or equal to that specified. If the priority is equal to 7777_8, GET attaches the file having the highest priority among those that satisfy all requirements; PEEK and COUNT do not allow this priority and return an error code of 03_8. ALTER does not use the priority as a search criteria. This value replaces whatever the entry had before if bit 50 in word 1 is set.

Word	Bits	Description
4	59-18	Predayfile file name. The predayfile file name is required only if the qualifying INPUT file has a predayfile. If the file has a predayfile, a separate FNT entry is created to describe the predayfile entry. If the predayfile file name is not given, the complete bit is set and the error code 14_8 is returned. If a predayfile file name is always in the field, the predayfile flag must be checked after each call to prevent duplicating the file name in the FNT entries.
	59-54	Spacing code (output file only). The spacing code associated with the output file is returned on a GET function. It cannot be specified as a parameter but will be returned.
	17	Predayfile flag bit.
	16	Class 2 INPUT file inhibit flag. The class 1 and class 2 INPUT file inhibit flags are used to achieve selectivity in terms of file classes for GET, PEEK, or COUNT functions. A class 1 file has no nonallocatable files associated with it. A class 2 INPUT file has at least one nonallocatable file associated with it. If the job statement specifies MTxx or NTxx, a nonallocatable file is associated with the INPUT file. If the caller sets the class 2 inhibit flag on a GET call and also sets the INPUT file type bit, only class 1 INPUT files are returned.
	15	Class 1 INPUT file inhibit flag.
	14	Inhibit duplicate file name search flag. If the flag is 1, no search is made for a duplicate file name.
	13-12	Not used.
	11-0	Executing job count.
5	59-48	Additional DID word count. The additional word count is used whenever more than one is needed in the parameter list. Additional DIDs are packed three per word and terminated by a byte of zeros. A maximum of 64 (decimal) is allowed. The count is the number of CM words required to hold the additional DIDs. It is not necessary to allocate an additional CM word simply to hold a terminating byte of zeros. To hold six additional DIDs, for a total of seven DIDs in the parameter list, the DID word count is two to three DIDs in the first word and three in the second.
	47-36	INPUT file count. Only one of the four file count fields can be specified on a PEEK function; the other fields are not used. File type determines which file count field is to be used.
	35-24	OUTPUT file count.
	23-12	PUNCH file count.
	11-0	Special output file count.

Word	Bits	Description
6	59-42	Destination ID 2.
	41-24	Destination ID 3.
	23-6	Destination ID 4.
	5-0	Not used.
n	59-42	Destination ID n.
	41-24	Zero terminator.
	23-0	Not used.

Summary of Parameter List Usage

The required and optional parameter list field usage for each QAF function and the corresponding returned parameters are shown in table 4-1.

TABLE 4-1. ACQUIRE MACRO PARAMETERS

Field Name	Field Usage							
	ALTER		GET		PEEK		COUNT	
	Call	Return	Call	Return	Call	Return	Call	Return
File/job name	O	X	O	X	-	X	-	X
Error code	-	X	-	X	-	X	-	X
Queue type	R	X	R	X	R	X	R	X
QAF function code	R	-	R	-	R	-	R	-
Complete bit	R	X	R	X	R	X	R	X
ALTER flags	R	-	-	-	-	-	-	-
Forms code	O	-	O	-	O	-	O	-
DISP code	-	-	O	X	O	X	O	X
Repeat count	O	-	-	X	-	X	-	-

TABLE 4-1. ACQUIRE MACRO PARAMETERS (Contd)

Field Name	Field Usage							
	ALTER		GET		PEEK		COUNT	
	Call	Return	Call	Return	Call	Return	Call	Return
FWA reply buffer	O	-	-	-	R	-	-	-
Source ID	O	-	O	X	O	X	O	X
DID 1	O	-	O	X	O	X	O	X
FNT add/JDT ord	O	X	O	X	O/R††	X	O	X
Terminal IDn	O	-	-	-	-	-	-	-
Terminal IDc	O	-	O	-	O	-	O	-
Priority	O	-	R	X	R	X	R	X
Spacing code (SC)	-	-	-	X	-	-	-	-
Predayfile file name	-	-	-/R†	-	-	-	-	-
Predayfile flag	-	-	-	X	-	X	-	X
Class 2 inhibit	O	-	O	-	O	-	O	-
Class 1 inhibit	O	-	O	-	O	-	O	-
Executing job count	-	-	-	-	R/-††	X	-	X
Add DID word count	O	-	O	-	O	-	O	-
Input file count	-	-	-	-	R/-††	X	-	X
Output file count	-	-	-	-	R/-††	X	-	X
Punch file count	-	-	-	-	R/-††	X	-	X
Special out file count	-	-	-	-	R/-††	X	-	X
DID 2	O	-	O	-	O	-	O	-
DID n	O	-	O	-	O	-	O	-

Explanation of symbols:

- O Optional parameter.
- X Parameter returned by QAF.
- - Parameter not used.
- R Required parameter.

† Required only if predayfile is present.
†† Second symbol is for the special PEEK function.

ALTER

ALTER, function code 0, gives the user the ability to change various fields within the queue entries that match the selection criteria specified in the parameter list. Required parameters are queue type and the ALTER flag bits which indicate the actions to be performed. Optional parameters are queue entry name, address of an abort message, source ID, destination ID(s), and terminal ID. The forms code and priority fields contain the new values and, thus, cannot be used as a search criteria.

The actions that can be performed are:

- Change routing of INPUT and/or OUTPUT queue files to the central site.
- Change routing of INPUT and/or OUTPUT queue files to another terminal.
- Change priority of OUTPUT queue files.
- Change forms code of OUTPUT queue files except for nonstandard output (PLOT, FILM, and so forth).
- Change repeat count of OUTPUT queue files except for nonstandard output.
- Abort/evict queue entries and issue supplied error message.

The bits that indicate these actions may be set in any combination, but certain combinations are mutually exclusive. For example, if the first two actions are specified, the result is as if only the first action had been specified. Similarly, the last action overrides all other actions.

The user may also set the queue type bits in any combination but the combinations used when aborting a job can make a difference. For example, if all queue types are specified, the job is killed rather than dropped.

GET

GET, function code 1, selects the file that best meets selection criteria and attaches it to the control point of the calling routine. Required fields are function code, priority, file type, and a zeroed completion bit.

Before a file is attached, a search is made to ensure that no file having the same name is already attached. If a duplicate file is found, an error code of 12_8 is returned and the completion bit is set. The search for a duplicate file name can be suppressed by setting the inhibit search flag.

When the selected file is attached, an FNT supplement of type 0101_8 (if an input file is attached) or 0102_8 (if the attached file is output) is created and linked to the base FNT. The control point number of the job is written into the FNT. When the file is returned by the calling job, the FNT supplement is erased.

After the file is attached, the complete bit is set to one, the file name and FNT address are inserted, and the source ID and the destination ID are entered. Should no file satisfy all the selection criteria, the complete bit is set to one, the FNT address is zeroed, and an error code of 02_8 is returned.

PEEK

PEEK, function code 2, creates a list of three-word reply entries built from the queue entries matching the selection criteria. Required fields are function code, priority, zeroed completion bit, the first word address of the rely buffer in the user's field length where the reply entries are returned, the queue type count of the number of reply entries to be returned, and the queue type. Only one queue may be specified in the queue type. Optional queue entry selection criteria also include the starting FNT address or JDT ordinal from an earlier PEEK request for the same queue.

PEEK begins examining the FNT entries at the point specified by the FNT address or at the start of the FNT if no address is provided. For each file that matches the file selection criteria, a three-word reply entry is built from the file's FNT. The reply entry is placed in the reply buffer, and the file type count is incremented by one. PEEK continues searching until the requested number of reply entries is found or the end of the FNT is encountered. The function works in a similar manner for the execution queue using an optional starting JDT ordinal.

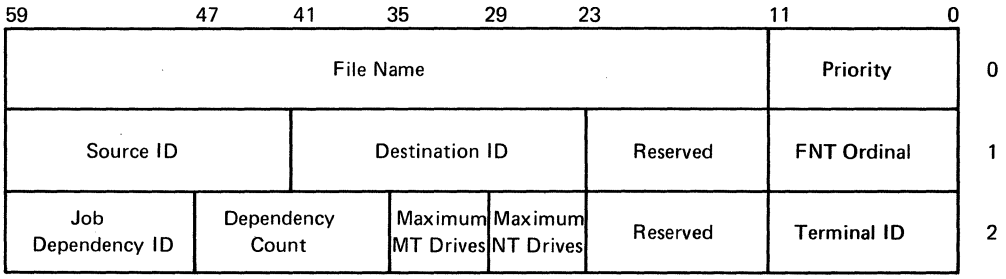
On return to the calling routine, the reply buffer, beginning at the first word address specified, contains the three-word reply entries. The count field for the queue type requested contains the number of reply entries built. The count is either the number requested or the number of entries built upon reaching the end of the FNT or JDT. For example:

> A user calls the QAF PEEK function to obtain 20 input queue reply entries for files having a destination ID of ABC. The search is to begin at FNT address 4420_8, with reply entries stored in the user field length, beginning at REPBUF. QAF begins searching the FNT at FNT address 4420_8 looking for input queue having a destination ID of ABC. Assuming that only 15 entries are found before reaching the end of the FNT, the file count is set to 15, the FNT address is set to 0, and REPBUF contains 15 three-word input queue file reply entries built from the FNT of the 15 qualifying files. If 20 entries are found with the last qualified at FNT address 4730_8, the FNT address is set to 4733_8, ready to begin the next search, the input queue file count remains 20, and REPBUF contains 20 three-word reply entries.

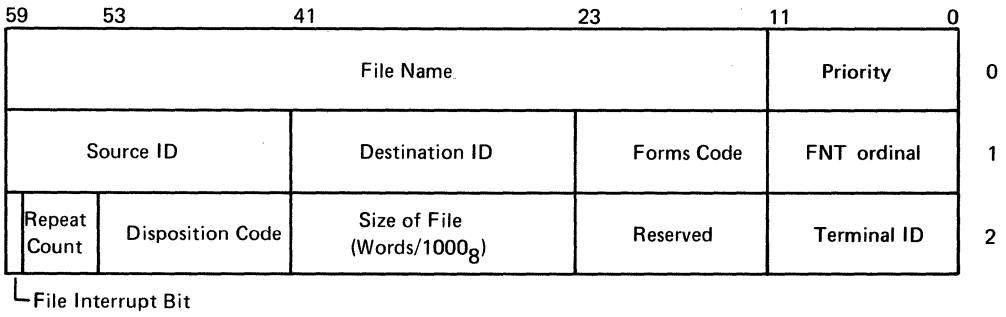
A special PEEK function is defined with the file type field zero. The caller may check a particular FNT entry at the address specified to determine whether or not the entry matches the file selection criteria. Required fields are function code, a zeroed completion bit, the queue type field cleared, the first word address of the reply buffer, and the FNT address of the file. Optional parameters are any of the file selection criteria.

If the file at the specified address qualifies, the queue type, the complete bit, priority, and so forth are inserted into the fields and a single three-word reply entry, built from the FNT entry, is placed in the reply buffer. If the file does not qualify, the complete bit is set, an error code of 02_8 is inserted, and the queue type field remains clear.

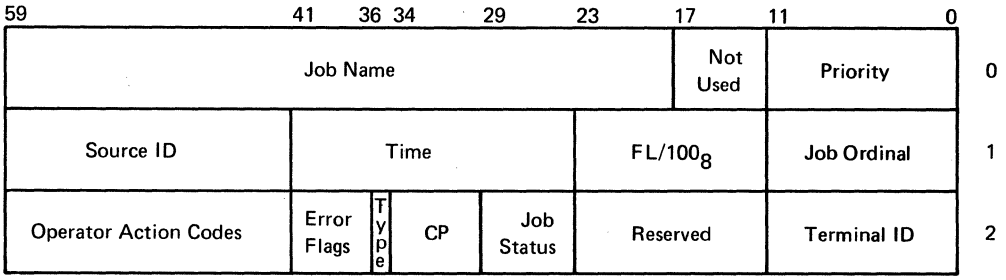
Format of the three-word reply entry for an input queue file is



Format of the three-word reply entry for an output queue (print, PUNCH, or special output) file is



Format of the three-word reply entry for an execution queue entry is



Word	Bits	Description
0	59-18	File name/job name.
	17-12	Not used.
	11-0	Priority.

Word	Bits	Description
1	59-42	Source ID.
	41-24	Destination ID/time left for execution in seconds.
	23-12	Forms code for output; not used for input; job field length/100_8 for execution.
	11-0	FNT ordinal/JDT ordinal.
2 INPUT	59-48	Job dependency ID.
	47-36	Dependency count.
	35-30	Maximum number of seven-track drives to be assigned at one time.
	29-24	Maximum number of nine-track drives to be assigned at one time.
	23-12	Reserved.
	11-0	Terminal ID.
2 OUTPUT	59	One means file interrupted.
	58-54	Repeat count.
	53-42	Disposition code.
	41-24	Size of file (words/1000_8).
	23-12	Reserved.
	11-0	Terminal ID.
2 EXECUTION	59-42	Operator action codes (SCOPE 2 only).
	41-36	Error flag values (in octal): 10 Kill. 4 Drop. 2 Rerun.
	35	Type of job (0 = 7600 or CYBER 70 Model 76; 1 = all others).
	34-30	Control point number the job currently occupies.

Word	Bits	Description
	29-24	Job status (in octal):
		70 Waiting for MMF action.
		60 Waiting for pack mount.
		40 Waiting for operator action.
		30 Waiting for tape/device assignment.
		20 Waiting for permanent file.
		10 Waiting for time/event.
		02 Executing.
	23-12	Reserved for future TID expansion.
	11-0	Terminal ID.

COUNT

COUNT, function code 3, counts the number of queue entries of a specified type satisfying the selection criteria. Multiple queue type bits can be set on a single call giving the caller the count of each queue type desired. Required fields are queue type, function code, a zeroed completion bit, and priority. Optional fields are the rest of the file selection criteria.

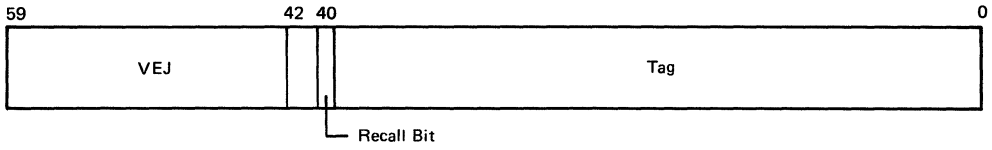
The counts of the queue types specified are returned to caller, and the complete bit is set. The file name, file type, disposition code, source ID, destination ID, FNT address, and priority are returned for the first file that satisfies the selection criteria.

VERIFYJ Macro

The VERIFYJ macro performs verification of basic job statement information for a new job input file. The job statement information is obtained from a buffer in the user's field length. This macro verifies the information, creates a system name for the new job, assigns the file to a queue device, and returns an FNT address in addition to the job name and verified status. VERIFYJ is available only to routines loaded from the system library.

The VERIFYJ macro formats a call to the PP routine VEJ (verify job statement) to perform the required functions. VEJ can be called only by a routine resident in the system library.

VERIFYJ has the following RA+1 interface.

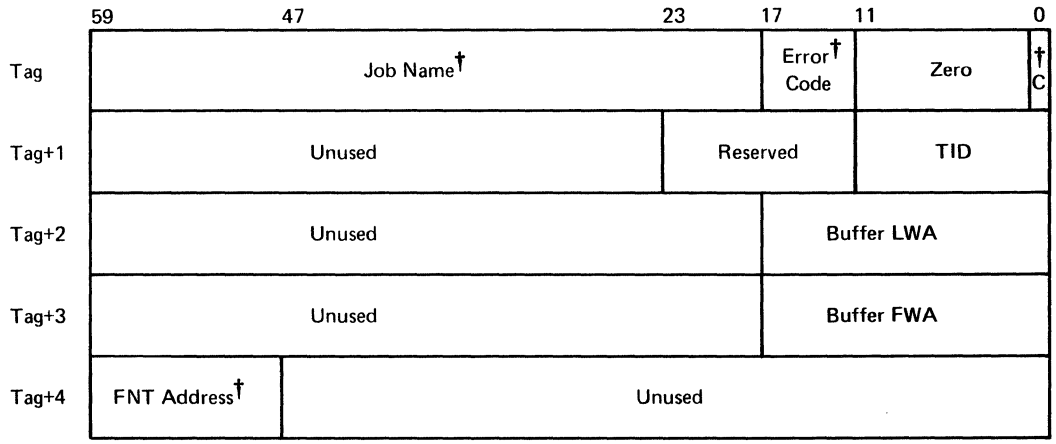


The user must construct a parameter block in the following format before calling the VERIFYJ macro.

VERIFYJ tag,recall

tag Address of the VERIFYJ parameter block.

recall Optional nonblank character specifying automatic recall.



Word	Bits	Field	Description
tag	59-18	jobname	The name assigned to the new job file which will be an input queue file once the file contents are complete.
	17-12	Error code	Status return.
			0 Successful.
			1 Job statement error.
			2 Buffer out of field length.
			3 Reserved.
			4 FNT full.
			5 VERIFYJ parameter block outside FL.
			6 Complete bit already set.
			7 No permission to call VEJ.
	11-1		Zero.
	0	C	Complete bit. Must be 0 when macro used.
			0 On call.
			1 On completion of call.

† These fields are returned by the VERIFYJ macro.

Word	Bits	Field	Description
tag+1	59-24		Unused.
	23-12		Reserved.
	11-0	TID	Terminal identification code.
tag+2	59-18		Unused.
	17-0	Buffer LWA	Address of the last word+1 in buffer of the statements to be verified.
tag+3	59-18		Unused.
	17-0	Buffer FWA	Address of the first word in the buffer of the statements to be verified.
tag+4	59-48	FNT Address	The location of the FNT for this file. The name of the file will be that one returned as jobname above.
	47-0		Unused.

Error code and corresponding fields returned are as follows:

Error Code	Fields Returned
0	All * fields as indicated in the parameter block.
1	All * fields as above, except if there is a jobname error, then jobname will be ERROR xy, where xy is the system sequence ID.
2	Only error code and complete bit.
4	Only error code and complete bit.
5	None; job aborted.
6	Only error code; job aborted.
7	Only error code and complete bit.

I/O TABLES

The input/output file requirements are coordinated with the status of input/output devices. File tables and device tables are updated continually to provide interface for user jobs and system programs.

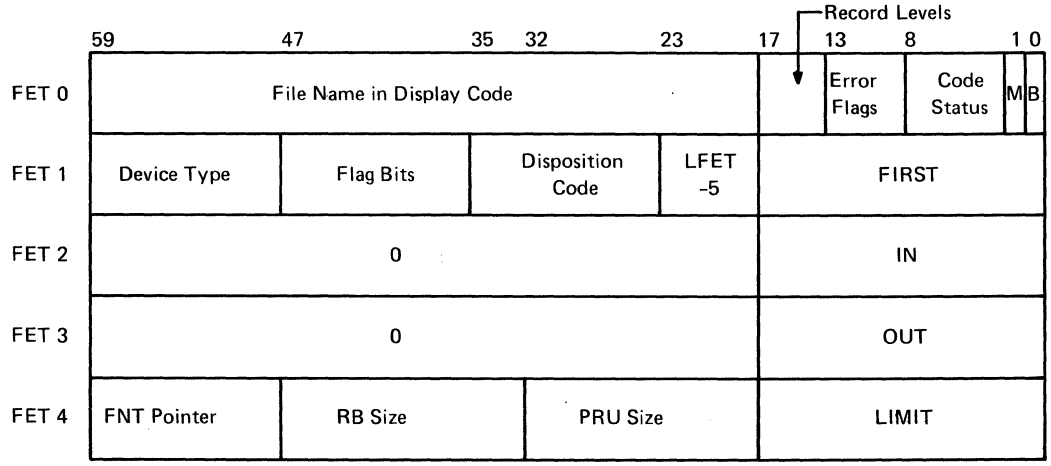
FILE TABLES

The status and requirements of files are kept in the following tables: file environment table (FET), file name table (FNT), file information table (FIT), and record block table (RBT). The FET and FIT are created within the job field length; the other tables are CM resident. The CM resident tables are in the upper table area of CMR, except the RBT which resides at the highest address of CM. Detailed descriptions of CM resident tables appear in appendix B. The FET is detailed in appendix C. The FIT is described in the CYBER Record Manager reference manuals listed in the preface.

File Environment Table (FET)

Every file for which I/O is to be performed must have a FET. Each FET consists of a basic five-word entry followed by additional words; the form depends on the type of I/O to be performed.

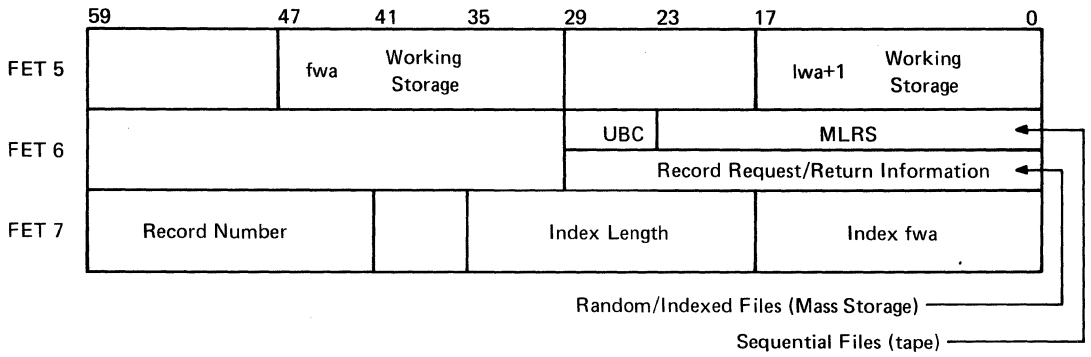
The basic five-word FET entry is as follows:



LFET -5	Length of FET minus five words for basic entry.
B	Busy (free is 1).
M	Mode (binary is 1).

Buffer parameters

$LIMIT \leq FL$
$OUT < LIMIT$
$IN < LIMIT$
$OUT \geq FIRST$
$IN \geq FIRST$



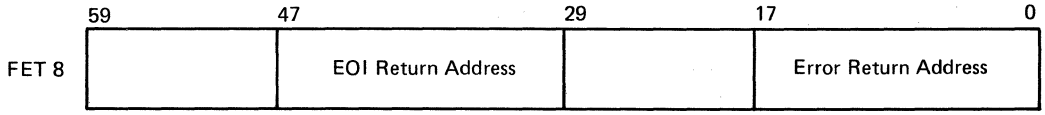
FET 5 is used for input/output blocking/deblocking by CPC.

FET 5 and FET 6 are used for S and L tape file processing.

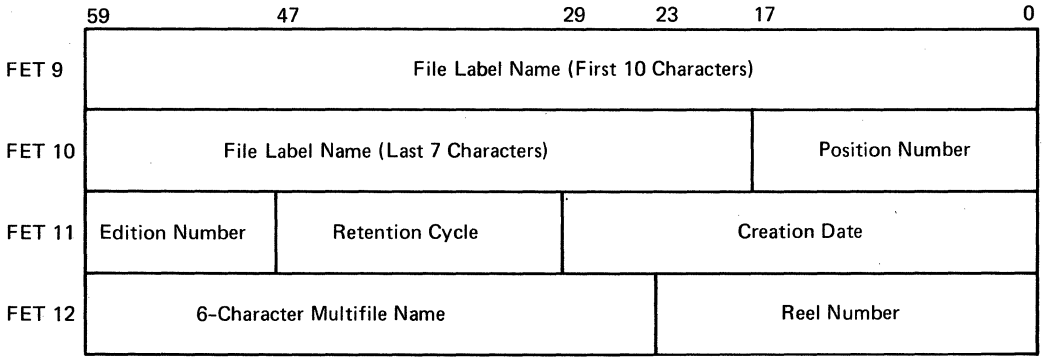
FET 6 and FET 7 are used for indexed file processing by CPC; FET 6 is used to pass RMS address between CP programs and system PP input/output routines.

UBC	Unused bit count.
MLRS	Maximum logical record size (S/L tapes only).

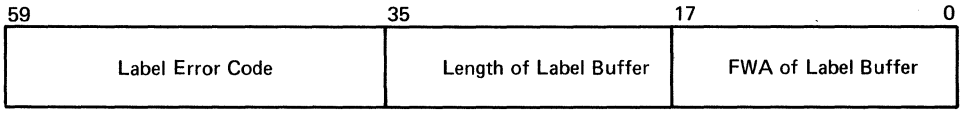
When the UP and/or EP flags are set in FET 1, then FET 8 contains



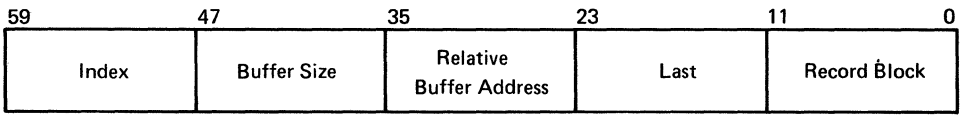
When standard file labels are to be written, the following FET words are filled with information from the LABEL control statement or macro. When a labeled file is read, the fields contain data read from the label.



When LFET-5 flag in FET 1 is set to 1 for extended label processing, FET 9 has the following format:



FET entries for the system dayfile, the hardware error file, and the control point dayfiles are kept in the upper table area of CMR, adjacent to the control point 0 dayfile buffer. The format of the one-word dayfile FET entries is



Buffer sizes are set by the operating system assembly configuration parameters internal to CMR. The origin address of each buffer is calculated by adding the relative buffer address to the T.DFB origin address of the dayfile buffer area in CMR. The current position within the buffer is determined by adding the index value to the buffer origin address. The field labeled last (byte 3) contains the value of the index when the buffer was last flushed to disk. For the system dayfile and hardware error file, the record block field contains the end-of-information record block when the file was last extended. For control point dayfiles, the record block field is not used.

The first five words of the control point 0 dayfile buffer are preset as follows (b represents a blank):

```
b D A Y F I L E : :
b b b N O R M A L b
( b b b b b b b b )
D E A D b S T A R T
: : : : : : : : : :
```

The system dayfile area in CMR is diagrammed as shown in figure 4-1.

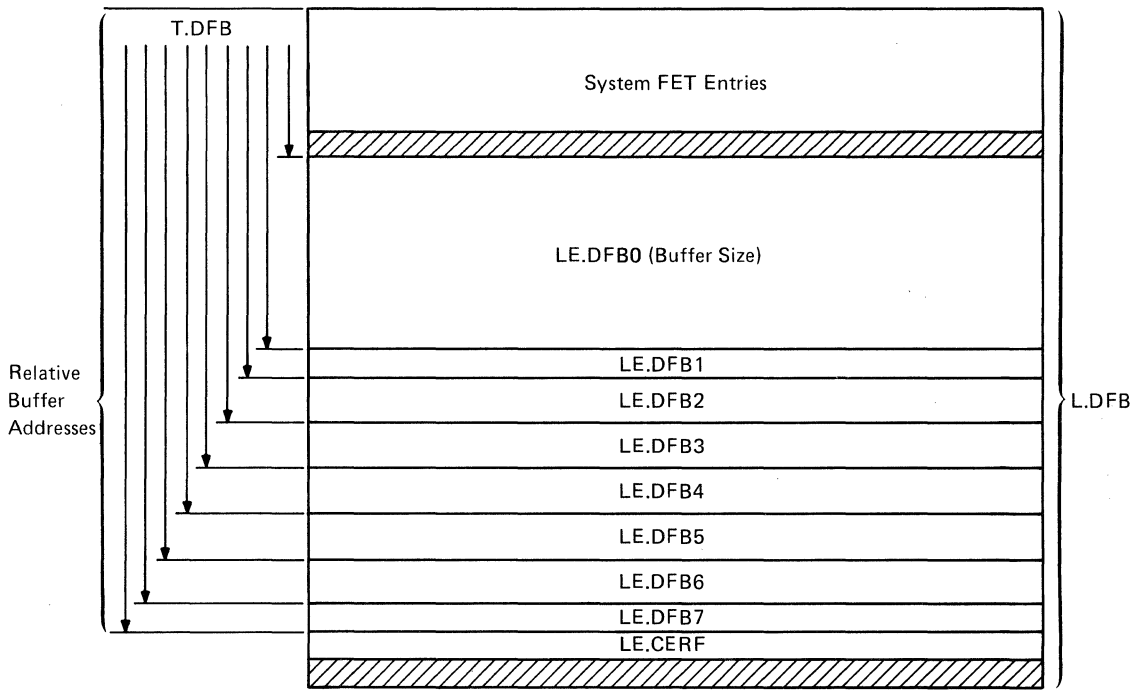


Figure 4-1. System Dayfile Area

When the system is assembled, several system file entries are built into the FNT/FST for control point dayfiles, system (library) file, and hardware error file. Their initial entries are diagrammed in figure 4-2.

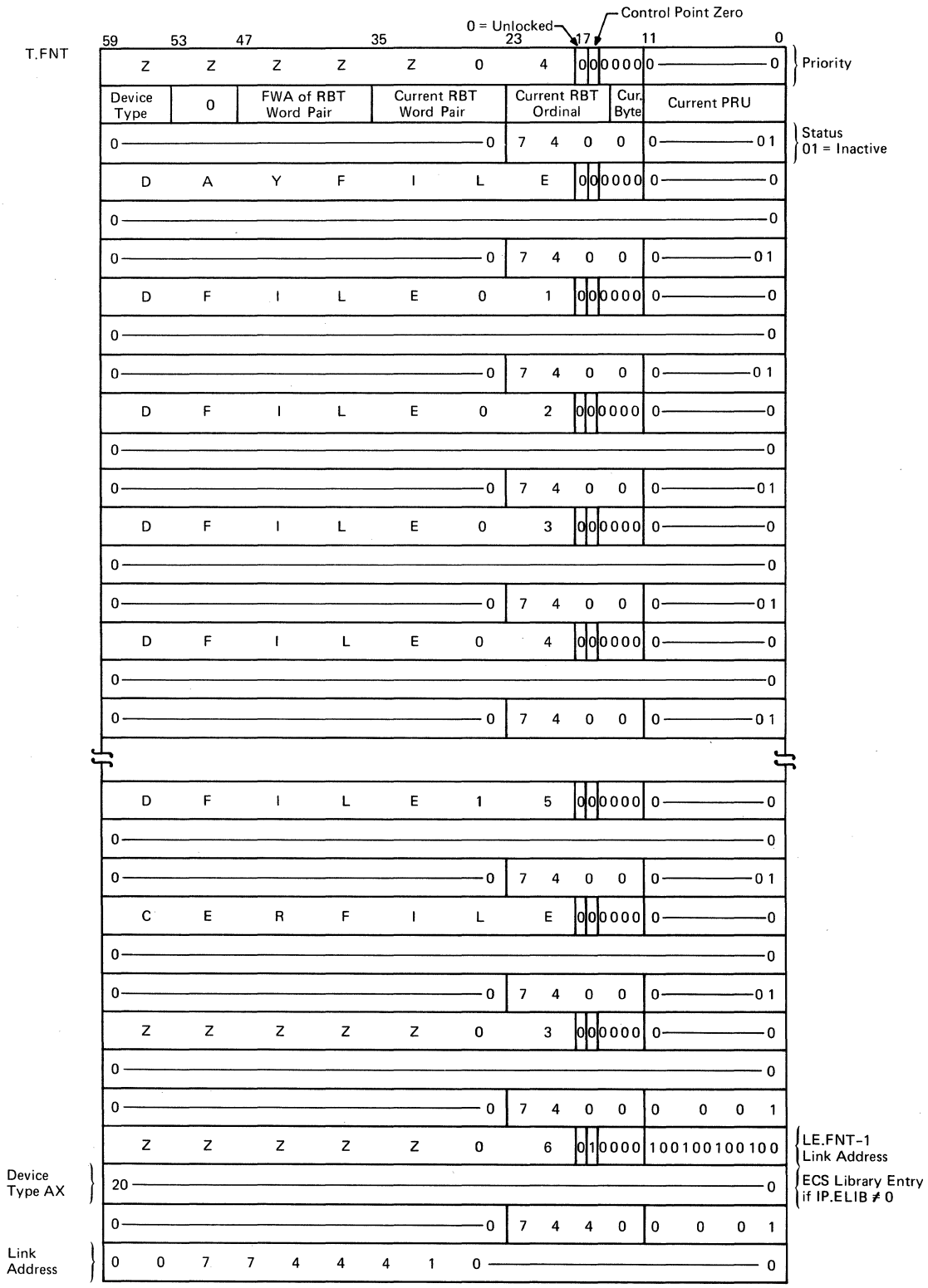


Figure 4-2. System File Entries

File Name Table (FNT)

To provide linkage between user programs and all I/O processing routines, the FNT is maintained in CMR upper table area. Each basic entry in the file name table consists of three words; one or two three-word extensions to entries may occur in some instances, extending the entry to six or nine words in length. The first word contains the file name, control point number to which it is assigned, as well as other pertinent information. The second and third words constitute the file status information; the format differs depending upon the type of file and where it resides. The various forms of the FNT entry are detailed in appendix B. The second and third words of the FNT entry are often called the file status table (FST) entry.

DEVICE TABLES

Tables in CMR that provide information on input/output equipment and channels are used by the operating system to make file assignments. Tables included in this section are the equipment status table (EST), containing entries for all I/O equipment in the configured system; device status table (DST) and device activity table (DAT), providing information related to mass storage devices and controllers; record block reservation (RBR) and record block table (RBT) containing information on each record block in a mass storage device; the dismountable device table (DDT) and mounted set table (MST) containing information related to the recording surfaces. The channel status table (CST) provides I/O channel availability information and serves as an interlock for major file tables, which prevents modification of the same table entry by two or more programs. Also included are the TAPES table and the tape staging table (STG), the device pool table (DPT), and the INTERCOM table (ITABL). These tables are detailed in appendix B.

Equipment Status Table (EST)

The EST resides in the upper table area of CMR and is pointed to by P.EST in the CMR pointer area. Table length depends on installation parameters. Therefore, the CMR pointer word also includes the LWA+1 address of the EST.

The EST contains a one-word entry for each device configured in the system, including consoles and remote terminal MUX devices. Each entry describes current status of the device and includes the device hardware mnemonic name, channels to which it is attached, device unit number, and so on.

Entries in the EST are numbered starting with one; an entry number, called the EST ordinal, is used to identify the table position of each equipment entry. The EST ordinal of the equipment being assigned is given as xxx in the operator command n.ASSIGNxxx. Only RMS devices may have EST ordinals greater than 77_8.

The EST is the basic reference for most other I/O tables. EST ordinals are found in the FNT/FST entries for linking file entries to their assigned equipment. EST ordinals in the TAPES table link tape entries to related equipment entries in the EST. Likewise, EST ordinals are found in the RBR, linking that table to the allocatable device it describes.

Dismountable Device Table (DDT)

The DDT is used to maintain the status of rotating mass storage devices that are logically removable from the system. The fixed section of the DDT is used to relate the status of an RMS drive to the status of the pack mounted on that drive. The variable section of the DDT is used to store pack requests that have not been satisfied. The second word of a fixed section entry has a pointer to the EST entry for the drive. Whenever the physical status of the drive changes, the EST is updated. 1RN compares the status bits in the EST with the status bits in the DDT and calls 1PK when a difference is detected. 1PK updates the DDT to reflect the new status of the drive and checks the variable section of the DDT to see if any

pack requests can be satisfied. If a requested pack has been mounted, 1PK updates the fixed section to include the DAM and MST ordinals, deletes the variable section entry, and recalls the job that had requested the pack. When a new pack request is made, 1PK checks the DDT to see if the device is already mounted. If it is mounted, 1PK satisfies the request. If the pack is not mounted, 1PK makes an entry in the variable section of the DDT and swaps the job out.

Mounted Set Table (MST)

The MST is used to keep pointers for each mounted device set in the system. Each MST entry has a corresponding set subdirectory table entry in the FNT. Entries are made by MNT when a master device is mounted and deleted by DSM when a master device is dismounted.

Device Status Table (DST)

The stack processor uses the DST in processing of mass storage files. The DST is located adjacent to the request stack in CMR upper table area. Each controller has one DST two-word entry which specifies the overlay to be used by the stack processor for each controller, pointers to a chain of requests entered in the request stack for that controller, and device availability information.

Each entry is numbered, starting from 1, to identify DST ordinals. The format of a DST entry is shown in appendix B.

The DST format reflects the new SPM/1SP working relationship and the multiple access approach. The first word contains multiple access information and the request stack chain pointer. The second word is the stack processor input register. It contains the DST ordinal used by SPM when 1SP calls, equipment and channel numbers, and PPIR activity pointer (used only as part of the DST).

The DST is a key table in the processing of allocatable storage files. DST ordinals are found in the DAT, RBR table header, and EST. A DST ordinal appears in each DST entry; it is placed into the input register of the PP assigned to process an entry for that device in the request stack.

Multiple access uses a DST master entry with DST multiple access memory entries. SPM assigns and tracks each 1SP independently at each monitor call. Each 1SP works on only one stack request at a time. All 1SPs operate independently of each other and are unaware of any other 1SP activity.

Device Activity Table (DAT)

The DAT is directly related to the device status table. It has one entry for each DST entry and is referenced by the mass storage device open overlay (3DO) in determining the best RBR to assign to a new or overflowing file.

The format for DAT entries are shown in appendix B.

Channel Status Table (CST)

The CST residing in the lower table area of CMR, contains a one-word entry for each hardware channel and each pseudo channel in the system. For a reserved channel, the PP reserving the channel is identified in the entry.

The channel number is obtained by a PP program from the EST entry for the type of equipment. The length of the CST includes entries for a minimum of 12 hardware channels (optionally 24 maximum) and 13 pseudo channel numbers.

Access to the FST/FNT/RBT is controlled by an interlock scheme which prevents two or more programs from attempting to modify the same table entry at the same time. Not all table accesses require pseudo channel reservations. Some of the conditions which require pseudo channels are:

- Entry is added to FNT.
- File is assigned to a control point, causing FNT modification.
- FST code/status byte is initialized.

Details of CST are given in appendix B.

Refer to figure 4-3 for tables related to file processing on nonallocatable devices.

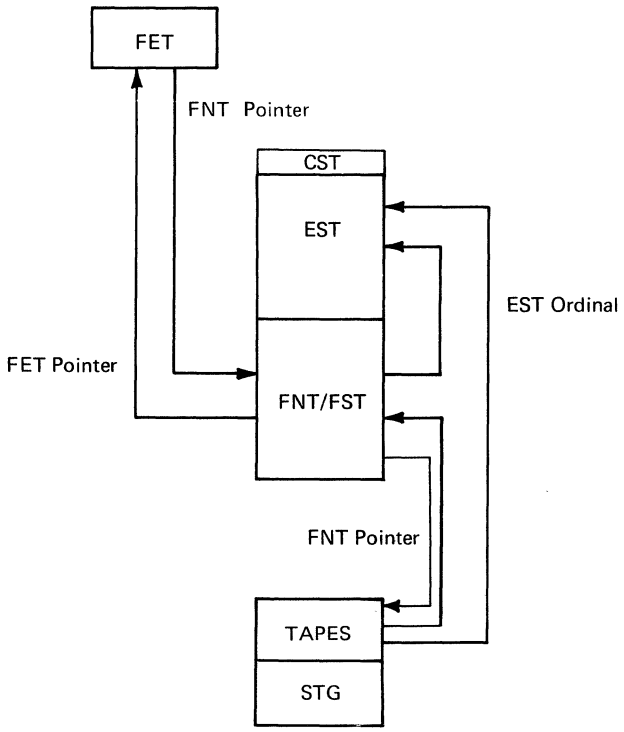


Figure 4-3. Nonallocatable Device File Processing

Tapes Staging Table (STG)

A satisfied job has all the tapes requested on its job statement. Unfilled demand is the sum of the job statement reservations of active jobs, less the tapes assigned to them.

The NO TAPE STATUS FLAG makes is possible to issue tape channel functions through DSD without interference from tape status processing. Status processing is not performed if the byte is nonzero. Normal system operation resumes when the byte is zeroed.

The three clocks are used to make event triggers for automatic assignment.

The STG appears in appendix B.

TAPE DRIVE SCHEDULING

Tape drive scheduling improves overall system throughput, particularly as it relates to tape job setup and execution. Automatic assignment, prescheduling, and overcommitment options are controlled by the value of IP.TSG.

AUTOMATIC TAPE DRIVE ASSIGNMENT

ANSI tape labels include a volume serial number (VSN) field. The user can have tape drives assigned automatically to his ANSI-labeled tapes by specifying the VSN on a VSN statement, in the REQUEST function, or as a parameter on the REQUEST or LABEL control statement. The VSN statement relates the external sticker or VSN to the file name and also provides information required for the tape job prescheduling display. When used with the REQUEST or LABEL control statements or the REQUEST function, it relates a VSN to a file name, which is relevant to equipment assignment. By itself, however, the VSN serves no purpose. When a VSN control statement provides the first reference to a file, a dummy FNT entry is set up using equipment code 64. If no subsequent REQUEST or LABEL control statement or REQUEST function provides additional information about the file, CIO finds the 64 equipment code in the FNT entry, releases that entry, and creates a default disk file. This feature does not encroach upon automatic assignment by label. The VSN parameter declares the tape label as either type U (full ANSI-standard label) or type Z (SCOPE 3 nonstandard label). The Z labels are not ANSI standard because the recording density field (character 12 of the volume header label) is not standard.

For automatic assignment of unlabeled tapes, the VSN must be entered by the operator. The tape is then assigned automatically to all jobs naming its VSN. Y-labeled tapes do not contain VSN information; however, to achieve automatic tape assignment, the operator can enter a VSN for a Y tape through the console. No automatic assignment is provided for 2MT or 2NT parameters.

TAPE JOB PRESCHEDULING

The tape job prescheduling display is an extension of the P display and lists, by VSN, the tape reels required by each tape job. A tape job is any job which contains one or more of the tape parameters MT, NT, HD, PE, or GE on its job statement. All incoming tape jobs are entered in a prescheduling queue, a subset of the input queue. The purpose of a prescheduling queue is to advise the operator of tape reel requirements and to keep jobs from being processed until the required tapes can be obtained from the tape library. This arrangement also allows the operator some control over the selection of tape jobs for execution.

The operator communicates with the prescheduling queue through DSD commands and the P display. Each time the P display is requested, tape jobs having the highest priority are displayed. A job requiring tapes is not placed in the normal job input queue until the operator releases it with a command. Once released, the job is considered for assignment to a control point and execution; it no longer appears in the prescheduling display.

JOB SCHEDULING WITH TAPE DRIVE OVERCOMMITMENT

Job scheduling based on tape drive overcommitment assumes that a tape job does not always need its maximum tape requirement for the duration of the job and that most processing uses fewer than the maximum number of drives requested. Therefore, a job is assigned only the drives it needs to continue execution at any instant in time; excess drives, at that instant, are made available to run other jobs. Such a job scheduling algorithm permits the total tape requirements of all active jobs to exceed the total number of drives in the installation. However, a system deadlock could occur if two or more jobs have unfilled tape demands, such that every tape drive is assigned but no job has enough tapes to run to completion. Such a deadlock could be broken only by rerunning or killing one of the competing jobs. Although the job scheduling algorithm includes some built-in deadlock prevention features, preventing deadlocks is a function of tape assignment, not job scheduling.

As part of REQUEST processing, a deadlock prevention algorithm is provided. A potential deadlock exists if at least two jobs have unsatisfied tape requirements and the number of free tapes is less than the maximum required to satisfy any one job. The deadlock prevention algorithm refuses any tape asignment (manual or automatic) if such assignment would create a potential deadlock. Tape jobs could be scheduled at random without regard to tape drive availability and the deadlock algorithm would prevent deadlocks, but the resulting refusal of tape assignments would cause operator confusion and loss of efficiency.

Depending on the installation option to enable or disable scheduling by density (IP.SCHDE), deadlocks involving 679 tape units may occur. If the option to schedule by density is disabled (IP.SCHDE = 0), the system assumes all nine-track tape units have the same recording capabilities. However, models 679-2/3/4 tape drives are capable of 800/1600-cpi density operations while models 679-5/6/7 tape drives are capable of 1600/6250-cpi density operations. Without careful scheduling, a single job can cause a deadlock. For example, assume an installation has two 800/1600-cpi 679 tape units and two 1600/6250-cpi 679 tape units. A job requires two tapes recording at 1600 cpi and one tape at 800 cpi. If the two tapes requiring 1600 cpi are assigned to the 800/1600-cpi units, the tape requiring 800 cpi cannot be assigned. Procedures to resolve schedule deadlocks involving a mixture of 679 tape units and other nine-track tape units are described in part I of the NOS/BE Installation Handbook.

If the option to schedule by density is enabled (IP.SCHDE = 1), nine-track tape units are scheduled by the system according to the density parameters specified on the job statement.

DYNAMIC TAPE DRIVE STATUS CHECKING

Information concerning the physical status of tape drive units is entered into the TAPES table and updated by periodic checks of unassigned units for a ready/not ready status. This information is displayed in the top half of the P display. The period for status checking is set by the installation; it must be short enough to preclude the possibility of an operator dismounting a tape from a tape drive and mounting another without detection. Such periodic checking of unassigned tape drives makes automatic assignment more efficient and flexible.

Initially a tape drive is set to not-ready status as noted in the TAPES table. When a drive is made ready, the TAPES table is updated with information from the tape label. (If the tape is unlabeled, this fact is noted in the table.) A search is made for a job that needs the tape, and the tape is assigned to it, providing such an assignment will not cause a deadlock. This action applies to both labeled tapes and tapes qualifying as scratch.

Whenever a requested tape cannot be located immediately, the requesting job is rolled out until the operator mounts the tape. When the tape is found, it is automatically assigned to the requesting job and the job is rolled back into CM to continue processing. While the job is rolled out, the operator can make a manual tape assignment which causes the job to be rolled in automatically.

Dynamic tape drive status checking permits the automatic assignment of unlabeled tapes by VSN. A VSN entered by the operator is recorded in the TAPES table; as long as that drive remains in the ready status, the system knows that the tape is still mounted and that it can be assigned without operator intervention by any job requesting that VSN.

RMS SET TERMINOLOGY

All disks used in the operating system are divided into sets. The term disk includes fixed disks and removable packs and is distinct from a drive which can hold different disk packs at different times. A set is an independent group of disks; a disk belongs to only one set, and files do not overflow to another set. Any user may own a set of removable disk packs.

Private sets are removable and mountable by job requests and operator action. Each member is mounted as needed, and members (other than the master) may be dismounted by operator command at any point in processing; masters may be dismounted when no jobs reference them.

Public sets remain mounted at all times and have either permanent file default, system, queue, or scratch attributes, or a combination. These can all be combined in one set. Also, the members individually have SYS, PF, and Q attributes to further delimit file allocation. All these attributes are set by the operator at deadstart, and the individual devices can be given PF and Q attributes only by initialization deadstarts.

Members of public sets cannot be dismounted; however, empty members can be deleted by DELSET, and new members added by ADDSET.

The system set is used for the system file and its related files created by post-deadstart use of EDITLIB and LDCMR, and the dayfile and CERFILE. There is no parameter on the REQUEST statement to specify system. The user can request the system set and VSN by name.

The PF default set is assigned when a file requests PF and no setname; only the PF default set is consulted on an ATTACH when no SN (setname) is supplied.

The Q set is assigned for special name files such as OUTPUT, PUNCH, and so forth; these files cannot be assigned to another set. Deadstart consults only the Q set to retrieve the queues. If a file is to be moved via DISPOSE or ROUTE, it must first be assigned to the Q set with a RESULT(filename,Q) request.

Scratch sets are unlike the other sets as several sets may have the scratch attribute. Files not assigned by REQUEST and not special-named (OUTPUT, and so on) are assigned to a scratch set.

DEVICE SETS

Every RMS device is a member of a group of devices known as a device set. Such device sets can be either public sets or private (user) sets. All members of a private set must be the same device type such as all 844-21 disks or all 844-41 disks; a combination of device types is not allowed.

Public Device Sets

Each public device set is assigned one or more of the following set attributes:

System set	This set contains system files such as ZZZZZ04, ZZZZZ23, the system dayfile, and the C.E. diagnostic file.
Permanent file default set	This set contains permanent files for which an alternate device set is not explicitly assigned.
Queue set	This set contains the INPUT, OUTPUT, and PUNCH queue files.
System default (scratch) set	This set contains nonpermanent files for which a device set residence is not explicitly assigned.

Device set attributes are assigned at deadstart. All four attributes must be assigned for each mainframe. Only the system default attribute can be assigned to more than one device set on a mainframe.

Every device in a public device set can (but need not) be assigned one or more of the device attributes listed below. Devices within a private set can be assigned master device and permanent file device attributes. System device and queue device attributes are prohibited within private sets.

System device	This device can contain the system files given previously for the system set attribute. The system device attribute can be assigned only to public devices that are members of the system set.
Master device	The master device contains system tables relating to its device set. These tables include the device label, the PFD, the PFC, the SMT, the DAM, the PFT, and the LFT. Every device set must have a master device.
Permanent file device	This device can contain files for which the REQUEST control statement specifies PF.
Queue device	This device can contain files with names such as INPUT, OUTPUT, and PUNCH, file with nonzero disposition codes, and files for which the REQUEST control statement specifies Q. This device attribute cannot be assigned to devices within a private set.

The system device attribute is assigned at deadstart; the master, permanent file, and queue device attributes are assigned when the device set is created. Attributes assigned to devices (except for the system device attribute) need not match the attributes assigned to the device sets of which they are members.

Private Device Set

A private device set is a group of RMS devices that can contain permanent files and be logically and physically removed from a running system. Permanent files stored on a private device set can therefore be transferred from one computer to another without moving the entire system. Specific attributes cannot be assigned to private device sets.

Shared Device Sets

In a dual-mainframe system, certain device sets can be shared between mainframes. Such sets must consist entirely of 844-21, 844-41, or 855 devices. They cannot have the system set attribute. When a device set is shared, all devices within that set are shared. Devices can be shared at either the unit or the controller level. The system uses the hardware reserve feature to reserve access to critical tables during an update; consequently, only one mainframe can access a device during an update.

A pool of free space is maintained in the RBR of each mainframe sharing the device. Additional space is maintained in the DAM on the master device. The pool is replenished when it gets low, and excess is returned to DAM. If a stack request is outstanding but all local space is used, the request is chained into the device overflow table (DOT) contained in CMR. Permanent file access between mainframes is coordinated through the PFC.

To comprehend the functions of the various tables described in figure 4-4, the terms used in mass storage space allocation must be understood. Terms are defined below.

Sector — The smallest accessible physical space increment on a track of a rotating mass storage device.

PRU — The smallest amount of data a user can access; it is 64 decimal (100 octal) CM words and is usually equal to 1 sector.

RB — The smallest amount of mass storage that can be allocated. An RB, defined in the RBR header, is several PRUs in length.

RBR — A bit-coded table which indicates those RBs on a device which are assigned to file, flawed (defective), or available for assignment. A zero bit indicates that the specific RB is available for assignment. The number of PRUs in a default RB is given in table 4-2.

TABLE 4-2. DEFAULT RB SIZE

Device	RMS Type	Device Type		Default PRU/RB (Decimal)
		Mnemonic	Code	
844-21	Disk pack	AY	13	57
844-41	Disk pack	AZ	14	57 †
819	Disk pack (fixed)	AH	15	160
885	Disk pack (fixed)	AJ	17	160 †

† Using the default PRU/PB size, the 885 and 844-41 devices require two RBRs to fully describe the available disk space.

RMS TABLES

The record block reservation table (RBR) and the record block table (RBT) contain information about each record block in an RMS device. Figure 4-4 shows the interface between the RMS tables and the file/device tables described earlier in this section.

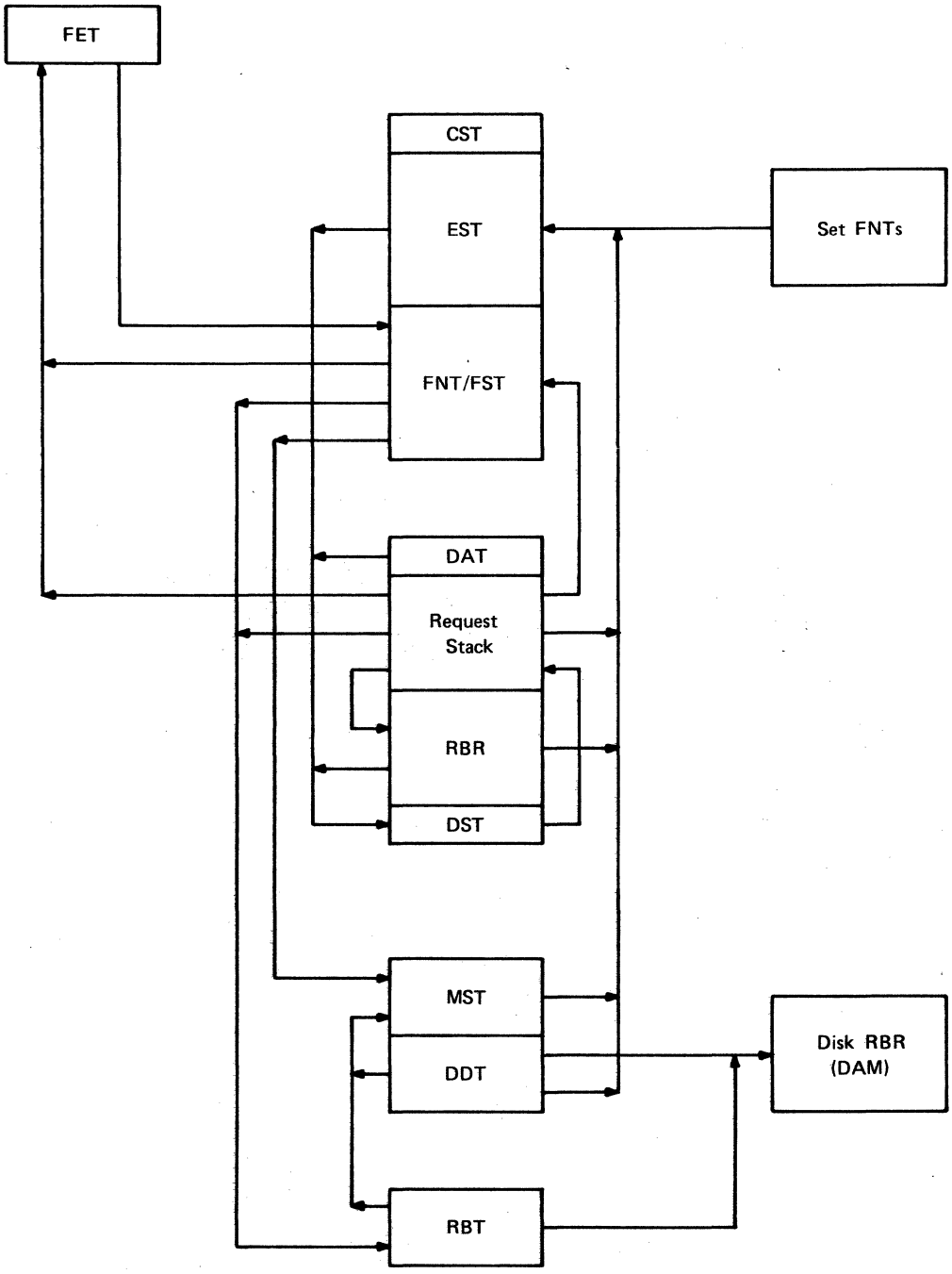


Figure 4-4. RMS Tables

Record Block Reservation Table (RBR)

A record block on a mass storage device is allocated to a file before any data can be written to that file. As data is written and a record block is filled, another record block must be assigned. Before the stack processor can select a record block to assign to a file, it must determine availability of record blocks. A record block reservation table maintained in CMR provides this information.

Each mass storage device is represented by at least one entry in the RBR. Several RBRs can be generated for a single device, each describing a unique area on the device. Each entry is made up of a two-word header and a variable length bit table. Each bit represents the availability of the corresponding record block. If a bit is zero, the RB is available for assignment; if a bit is one, the RB is not available.

The first word of each RBR header contains a 6-bit allocation style code supplied as a parameter to the RBR macro when the CMR is assembled at an installation. Unique allocation style codes for each RBR can be set by the installation; this code can be used to direct a file to the RBR with a specific RB size and/or recording technique.

Record Block Table (RBT)

The RBT is file oriented. Each mass storage file in the system has an associated RBT chain. The RBT, located in the high address end of CM, consists of word pairs which are linked to form an RBT chain for each file that exists on an allocatable device currently recognized by the system (refer to figure 4-5). The RBT expands and contracts by 100 (octal) word blocks as files are allocated and released. A maximum of 8192 (decimal) CM words may be assigned to contain all the RBT entries active at any one time.

When a mass storage file is established, a two-word RBT entry is created for that file; additional entries are assigned and linked in a chain as the file expands and entries are needed. Each entry consists of ten 12-bit bytes; some are used as pointers to additional entries in the chain and to other tables. Remaining bytes in the entry contain the RB numbers of record blocks assigned to the file. RB numbers are placed in sequential RB bytes in order of their assignment. An RB number serves as the address of a bit in the RBR and DAM bit tables representing the availability of that record block; it is also the address of the corresponding physical record block on the mass storage device. RBT entries are addressed by RBT word-pair ordinals. The word-pair ordinals are numbered sequentially starting from the highest address in CM.

The CMR pointer word P.RBT contains the current size of central memory divided by 100 (octal), as well as the current length of the RBT in 100 (octal) word increments. The same word also contains the RBT word pair ordinal of the first member of the RBT empty chain. Unused word pairs in the RBT are linked to form the empty chain. As record blocks are released from an evicted file, the dropped word pairs are linked into the empty chain. Word pairs are assigned to files from the head of the RBT empty chain, and the new first-member word pair ordinal is entered into the CMR pointer word. The RBT channel is requested as an interlock before a word pair is removed from the empty chain.

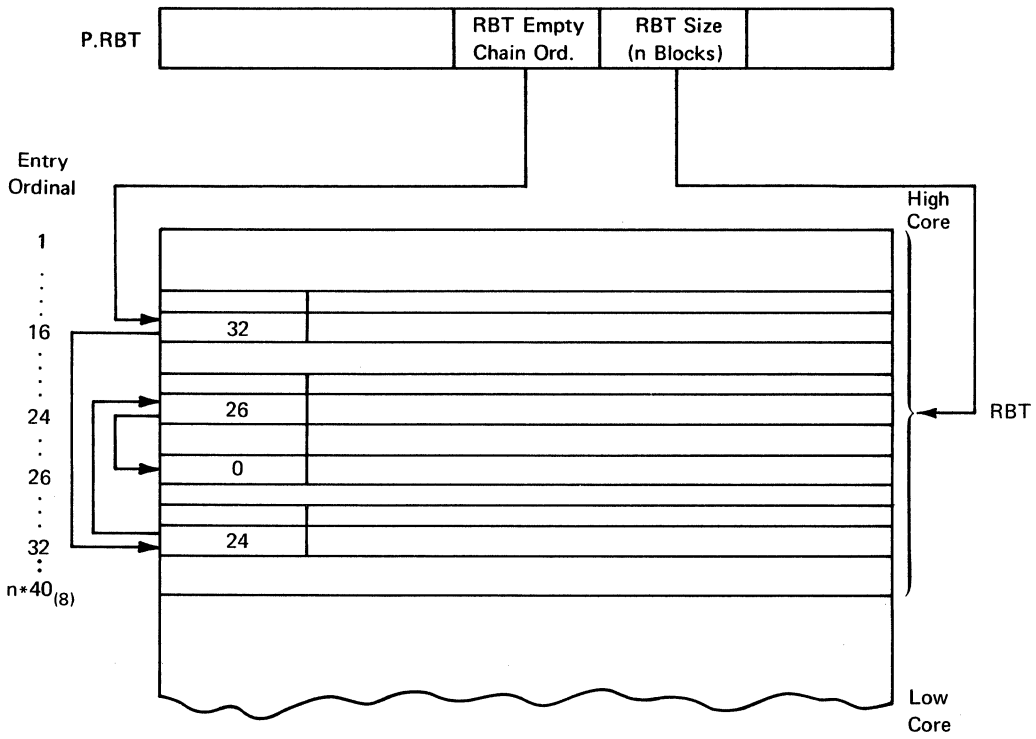


Figure 4-5. Record Block Table

Word 1 of the first word pair assigned to a file contains ordinals, flags, and so forth. The RB bytes denote the record blocks assigned to the file. These bytes, initially zero, are set as each record block is assigned. The values in the RB bytes are RB numbers which indicate the physical address on the device and a corresponding bit in the bit tables. As a file expands, additional RBs are entered into the RB byte fields until the word pair is filled; in this case, another word pair is assigned to the file and linked to the current word pair. If no more record blocks are assignable from the RBR/DAM table, an overflow condition occurs; in this case, a word pair in overflow format is linked into the chain, another word pair is linked to the overflow word pair, and processing continues with the remaining RB byte fields in the last link on the overflowed device set to zero.

As a file is evicted or record blocks are dropped, the RB bytes are cleared. When an entire word pair is emptied, it is linked into the RBT empty chain.

The end of a file's RBT chain is a word pair having zeros in byte 0 of the first word. The last word pair in the empty chain contains all zeros.

When a file is established for a job, an entry is made in the FNT table. The FST part of the FNT entry for a mass storage file contains the ordinal of the first RBT word pair (WP) of the RBT chain that describes the file. The same FST word also contains the current file position.

Figure 4-6 shows the FNT pointing to the RBT chain for the file. The RBT chain is made up of word pairs that are forward linked (byte 0). That is why the first word pair must be known. In the example, the file is described by the contents of two word pairs. The first word pair is 27 (octal) and the second is 52 (octal) to represent 27 and 52 word pairs from the end of CM, respectively. The first word pair contains some additional information besides RBs, such as the EOI PRU. This is the last PRU+1 of the last RB of the last word pair in the RBT chain that describes the file. The first word pair also contains the MST ordinal of the set that the file resides on. Each word pair contains the DAM ordinal of that part of the set that contains the RBs in that word pair.

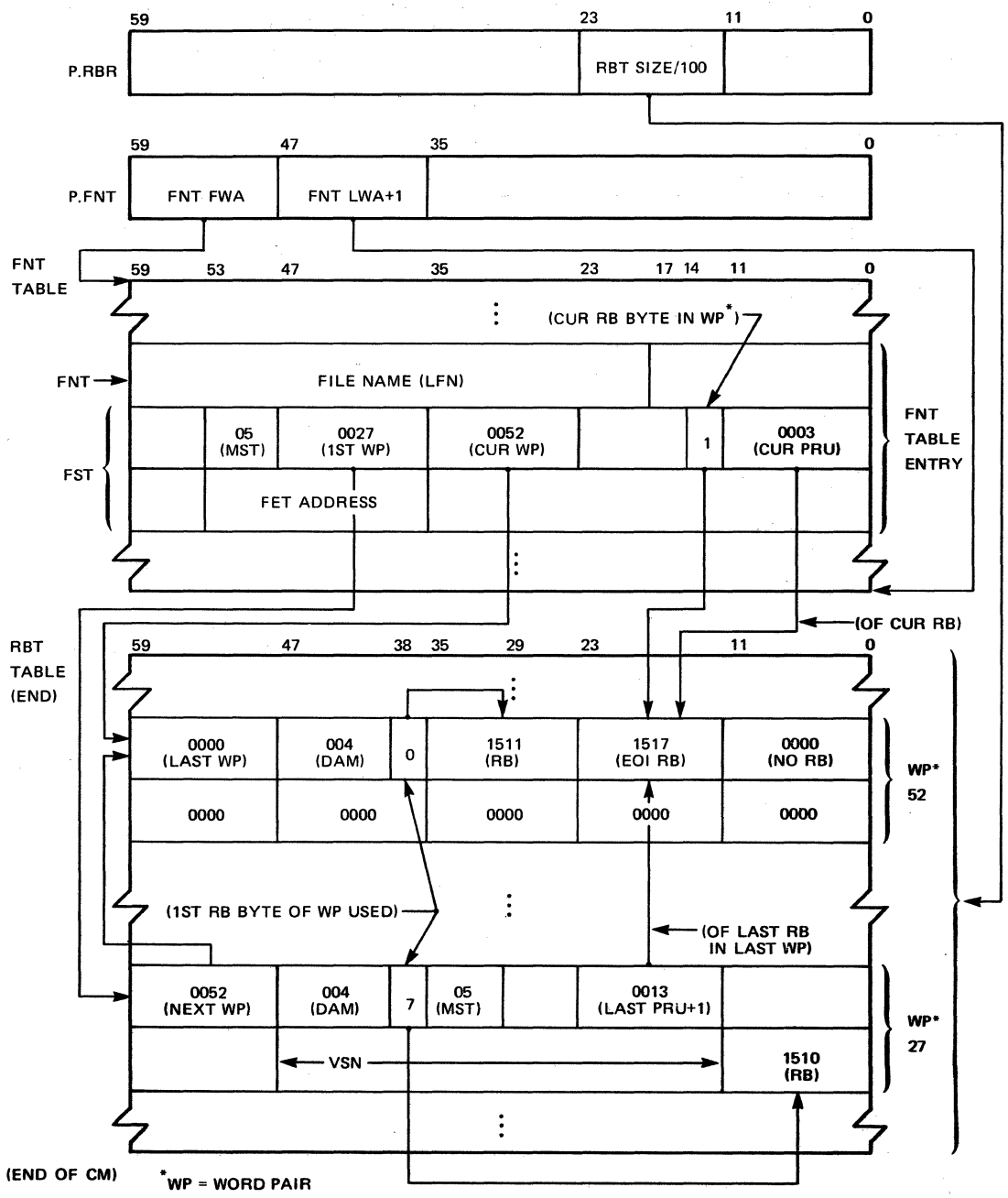


Figure 4-6. File Table Interfaces - FNT Points to RBT Chain

The FST in the example points to the first RBT word pair (0027) and the current position, which is specified as the current RBT word pair (0052), the current RB byte of that word pair (1), and the current PRU (0003) within the RB specified by that RB byte. The current position of the file in the example is seven PRUs from EOI.

Figure 4-7 shows how information in the RBT first word pair is used to find the correct RBR via the DDT. The DAM ordinal, rather than the RBR ordinal, is used to identify files because it is associated with the device (pack) in the set, while the RBR ordinal is associated with the drive (unit). When a pack is moved from one unit to another, its RBR ordinal changes but its DAM ordinal does not. The DAM ordinal is actually the relative PRU within the set DAM table on the master device. This table contains the DAMs for all members of the set. A DAM starts on a PRU boundary and may take one or more PRUs. Only those DAM (PRU) ordinals corresponding to the start of a DAM are valid as DAM ordinals.

The RBT first word pair in the example specifies DAM ordinal 004 and MST ordinal 05. A linear search is first made of the DDT table for an entry with the corresponding MST ordinal and a DAM ordinal range that includes the one specified in the RBT first word pair. (If no entry is found, the device is not mounted.) If only one DAM is associated with this DDT entry, the first and last DAM ordinals designated there will be the same number. Once the DDT entry is located, the DAM ordinal range (003 to 006 in the example) and the EST ordinal (10 in the example) found in that entry are saved.

A linear search of the RBR table headers is now made for the first entry with the same EST ordinal that was found in the DDT (10 in the example). When found, this RBR corresponds to the first DAM ordinal from the DDT (003 in the example). If this does not match the DAM ordinal from the RBT first word pair, the DAM ordinal corresponding to the next RBR is determined. This is done by taking the length of the bit map from the RBR header plus 3 for the DAM header (on disk), adding 77 (octal) and dividing by 100 (octal) to determine the number of PRUs this DAM takes, and adding that to the DAM ordinal this RBR represents to give the DAM ordinal for the next RBR. The DAM ordinal for this RBR is now compared to the DAM ordinal from the RBT first word pair and the search continues until the correct RBR is found. In the example, the search concludes at the second RBR.

The DAM ordinal range from the DDT was 003 to 006 and the DAM ordinal from the RBT first word pair was 004. The last DAM ordinal of the range 006 means either there are two more RBRs corresponding to DAM ordinals 005 and 006, or that the DAM with ordinal 004 requires two PRUs and only one RBR follows corresponding to DAM ordinal 006. As above, the correct situation is determined by examining the length field in the RBR header corresponding to DAM ordinal 004.

Figure 4-8 shows how the RB bytes in the RBT word pairs point to the RBs in the RBR bit table. Bits in the RBR bit table (pointed to from the RBR header) are allocated contiguously, 60 bits per word for the entire allocation space represented by this RBR. When an RB is allocated, the bit in the RBR bit table is set and its position is converted to an RB ordinal and placed in an RB byte in an RBT word pair. When an RB is deallocated, the RB ordinal from the RB byte is converted back to a bit position. That bit in the RBR bit table and the RB byte in the RBT word pair are both cleared.

To convert the current file position of the example to its equivalent RB bit position, check the FST current position from figure 4-6 to find word pair 0052 and RB byte 1. Figure 4-6 shows that RB (1517 in the example) pointing to the RBR bit table. The RB ordinal of 847 decimal (1517 octal) is first converted to an RB number by subtracting 1. Dividing 846 by 60 gives word 14 with a remainder of 6. Since RBR bit table bits are allocated left to right, this corresponds to bit 53 of word 14 of the RBR bit table. To find the corresponding physical address on the device, use the RB number and apply the appropriate formula given under Record Block Table Entry in appendix B.

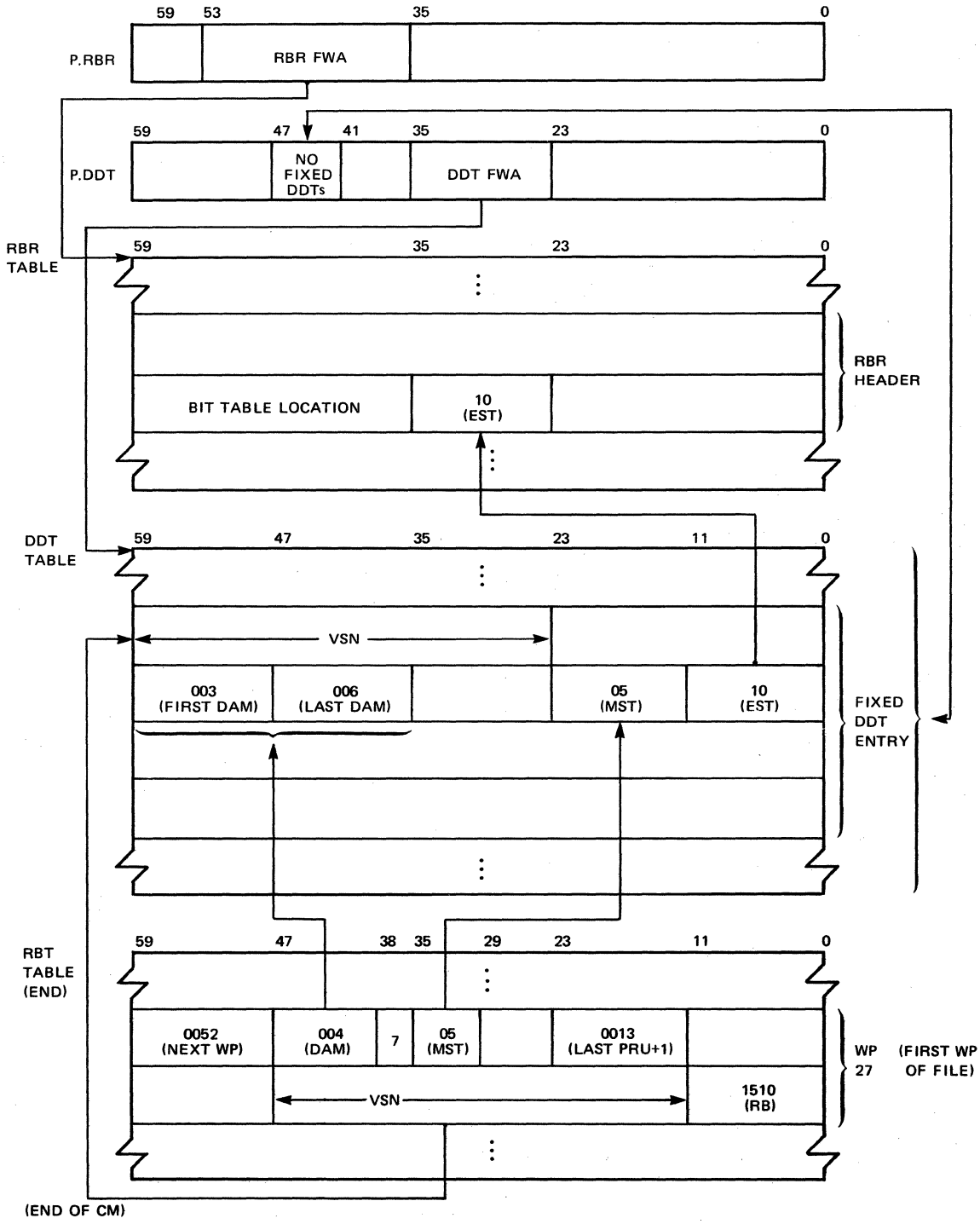


Figure 4-7. File Table Interfaces - RBT Points to RBR Via DDT

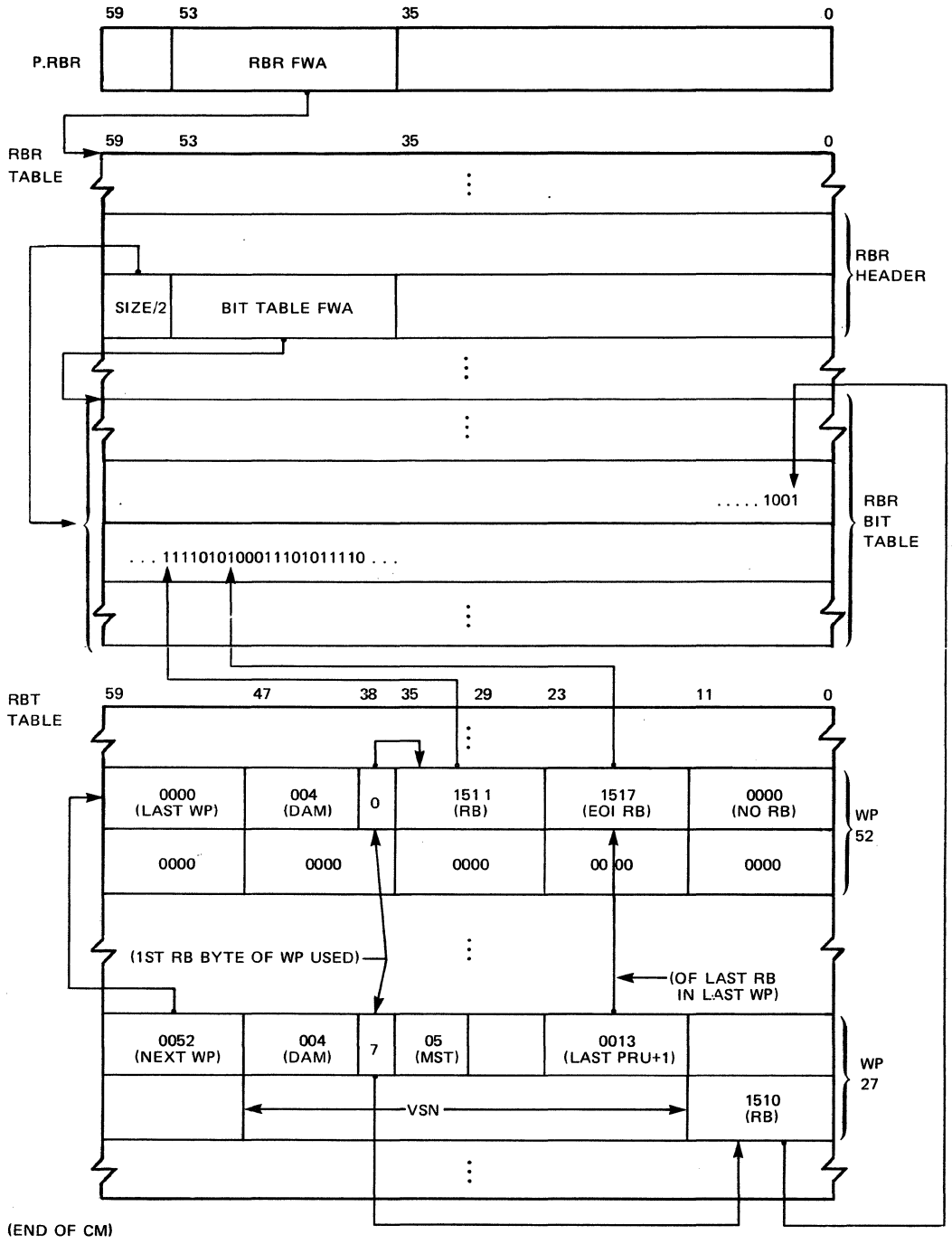


Figure 4-8. File Table Interfaces - RB Byte Points to RB

INTERCOM TABLES

INTERCOM uses word 16_8 of the CMR pointer area as the pointer to the INTERCOM multiplexer table and the INTERCOM pointer area. The multiplexer table and subtable entries contain a complete description of the communications equipment to be serviced by INTERCOM. The multiplexer table is central memory resident.

The INTERCOM pointer and buffer area is generated when INTERCOM is initialized and is not resident when INTERCOM is not running in the system. This area contains pointers to the various chains in the INTERCOM buffer area.

The INTERCOM tables are detailed in appendix B.

I/O PHILOSOPHY

Input and output request processing depends upon the source of each request. Active user programs request I/O through RA+1 requests, which are cycled through CPMTR. PP programs request I/O by placing a monitor request into their PP output register. System programs, which run at control point n+1, cannot make monitor requests through RA+1. Since they run as CM service functions for PP programs, they make such requests through the output register of the PP servicing the program.

CPMTR assigns the I/O request to CP.CIO which, in turn, assigns it to the proper processor, CIO or 1SP. The circular input/output processor (CIO) processes requests for magnetic tape, teletypewriter, and unit record I/O; the stack processor processes all requests for mass storage I/O.

Another I/O processor, JANUS, exists in the operating system, but its function is limited to processing unit record I/O for the system input and output queues. The queues contain job input and output files and are related to the job processing activities. JANUS is discussed in section 7, Job Processing.

CIO

CIO consists of the CM program CP.CIO, the PP program OV.CIO, and several PP I/O drivers. A system programmer can write his own input/output software, or he can have his program generate a call to CIO. Before calling CIO, the program must set up circular buffer parameters and the CIO operation code in the file environment table (FET) for the file. The relative address of the FET is placed in the CIO call.

A PP routine places a CIO call in its PP output register. PPMTR passes the call through the CP input register for the CP.MTR. A CP program places a CIO call in the CP request register (RA+1). When PPMTR accepts the CIO call, it assigns a PP and clears byte 0 of the PP output register.

When CP.MTR detects a CIO call, it passes it to CP.CIO for validation and selection of the proper CP.CIO routine to supervise execution of the function. The CIO call is then reissued via the request stack and CP.MTR to be processed by the required CIO driver; RA+1 is cleared. When the I/O operation is completed, CP.CIO adds 1 to the code/status field of FET word 1. As all CIO codes placed in the FET code/status field are even numbers, an odd number in that field signals completion of the operation (or that the file is not busy).

CIO CODES

All codes indicated by * are illegal; all reserved codes are illegal. All codes are octal for coded mode operations; 2 is added for binary mode. For example, 010 is coded READ; 012 is binary READ.

000	RPHR	054	*	130	CLOSE,NR
004	WPHR	060	UNLOAD	134	*
010	READ	064	*	140	OPEN
014	WRITE	070	RETURN	144	OPEN WRITE
020	READSKP	074	*	150	CLOSE
024	WRITER	100	OPEN,NR	154	*
030	*	104	OPEN WRITE,NR	160	OPEN
034	WRITEF	110	POSMF	164	*
040	BKSP	114	EVICT	170	CLOSE,UNLOAD
044	BKSPRU	120	OPEN,NR	174	CLOSE,RETURN
050	REWIND	124	*		

The 200 series is for special read or write (reverse, skip, nonstop, rewrite, and so on).

200	READC	230	*	254	*
204	WRITEC	234	REWRITEF	260	READN
210	READLS	240	SKIPF	264	WRITEN
214	REWRITE	244	*	270	*
220	*	250	READNS	274	*
224	REWRITER				

The 300 series is for tape OPEN and CLOSE.

300	OPEN,NR	324	*	354	*
304	*	330	CLOSER	360	*
310	*	334	*	364	*
314	*	340	OPEN	370	CLOSER,UNLOAD
320	*	350	CLOSER	374	CLOSER,RETURN

The 400 series is reserved for Control Data.

The 500 series is reserved for installations.

The 600 series is as follows:

600	*	630	*	654	*
604	*	634	*	660	*
610	*	640	SKIPB	664	*
614	*	644	*	670	*
620	*	650	*	674	*
624	*				

The 700 series is reserved for Control Data.

CIRCULAR BUFFER

A circular buffer is a temporary storage area in CM through which data passes during I/O operations (figures 5-1 and 5-2). It is termed circular because I/O processing routines treat the last word and the first word of the buffer area as contiguous.

FIRST is the first word address of the circular buffer. Routines that process I/O never change the value of FIRST.

LIMIT is the last word address+1 of the buffer area. No data is stored in this word. When LIMIT is reached, the next address accessed is FIRST. Routines that process I/O never change the value of LIMIT.

OUT is the next location from which data is removed from the circular buffer. CIO or the calling program changes OUT depending on whether the operation is read or write.

IN is the next location into which data is written. CIO or the calling program changes IN depending on whether the operation is read or write. When IN=OUT-1, the buffer is full. A partly filled buffer extends from OUT to IN-1.

The circular buffer must be at least one word larger than the length of one PRU. For a write operation, at least one PRU of data should be in the buffer. For a read operation, the buffer must have room to receive one PRU of data. Less than one PRU may be transmitted only if an end-of-record is read or written.

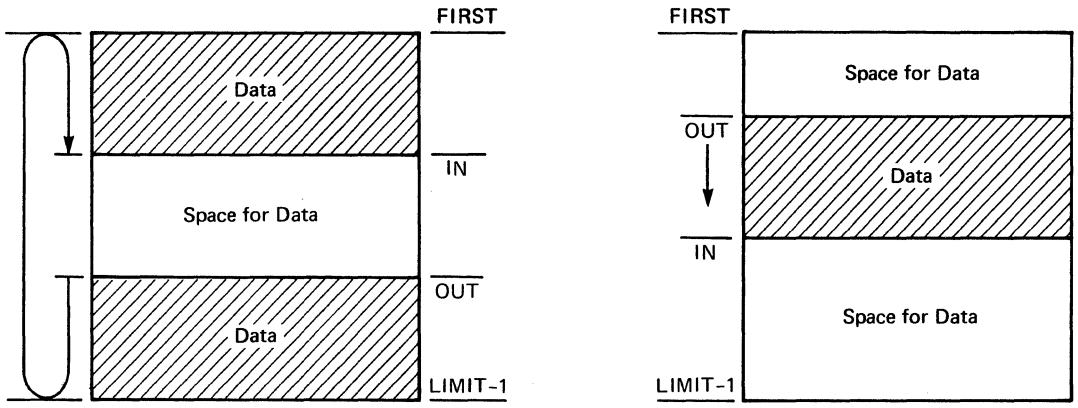


Figure 5-1. Circular Buffer

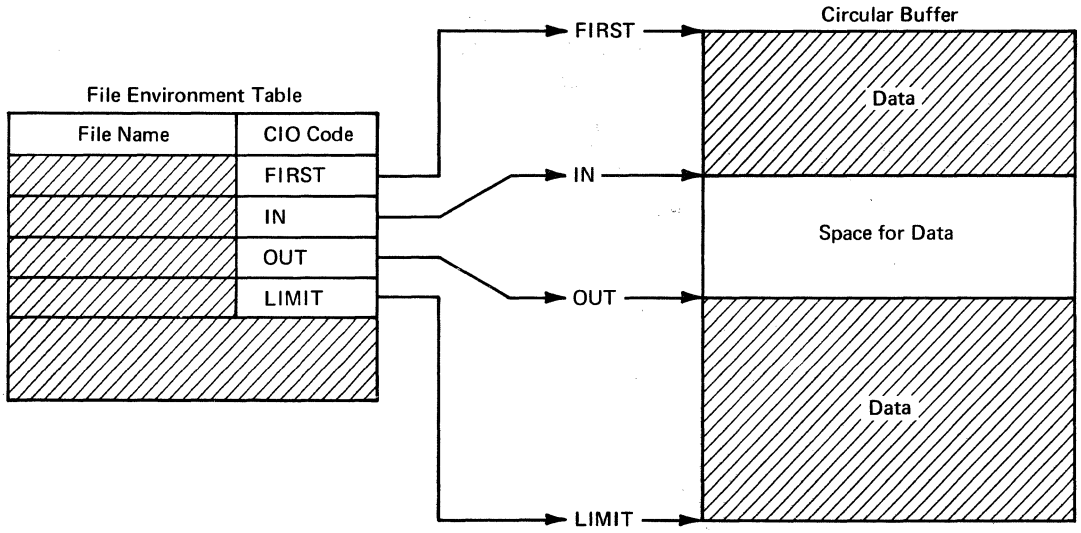


Figure 5-2. FET - Circular Buffer Interface

CIO OPERATION

When MTR initiates CP.CIO to perform file I/O, CP.CIO locates the FNT for the file. If the FNT pointer in the FET is not zero, CP.CIO checks the FNT entry indicated by the pointer to determine if the file name in the FNT entry is the same as the file name in the FET. It also checks that the file is assigned to the job control point. If the names do not match or if the FNT pointer is zero, CP.CIO searches the entire FNT for a file assigned to that job control point with a matching name. If the file is not found, CP.CIO creates an FNT entry for the file. Such files are always local and assigned to allocatable devices. Once the FNT entry is found or created, CP.CIO stores the address of the FNT entry in the FET. The FNT pointer in the FET facilitates the FNT search.

If file status is busy, CP.CIO posts the request for rescheduling and exits. Otherwise, CP.CIO checks the code field in the FET against the last code/status field in the FNT to ensure the requested operation can legally follow the preceding operation. If not, CP.CIO replaces the RA+1 call with a request for the PP program CEM which handles error messages, then reissues the RA+1 call to be processed again by CP.MTR. If the operation is legal, CP.CIO transfers the code/status field in the FET to the last code/status field in the FNT. The proper CP.CIO routine is selected to supervise function execution.

When the file is opened, CP.CIO determines if the file is on an allocatable or nonallocatable device or is ECS resident by checking the device code in the second word of the FNT. If the file is ECS resident, an ECS extension routine is called to process the request. If the file is on an allocatable device, CP.CIO calls CP4ES, which calls SPM to enter the request in the I/O request stack in CMR. The stack processor 1SP schedules I/O on allocatable devices; it performs the I/O and sets the completion bit. OV.CIO and its overlays process I/O requests for files on nonallocatable devices.

When OV.CIO is required, PPMTR assigns an available PP and causes OV.CIO to be loaded and initialized. Depending upon the operation, OV.CIO calls one or more of the following overlays.

Function routines:

1OP	File open (nontape files).
3DO	Mass storage device file open.
3IC	File close for 66x/67x tapes.
3IF	Multifile positioning for 66x/67x tapes.
3II	66x/67x initialization and setup.
3IJ	System calls to 1IT.
3IL	Slave for 3IO, 3IC, and 3IV.
3IM	Write error message for 66x/67x tapes.
3IN	VSN message processor for 66x/67x tapes.
3IO	Tape open for 66x/67x tapes.
3IV	Reel close; EOR processor for 66x/67x tapes.
4ES	Enter stack request (mass storage I/O).

Tape drivers:

1IT	Main 66x/67x tape driver, calls the nIx overlays.
1LC	Load conversion tables into 66x/67x controller.
1TS	Tape sampler.
2IA	L tape read for 66x/67x tapes.
2IB	L tape write for 66x/67x tapes.
2IC	Coded read (seven-track) for 667/677 tapes.
2ID	Coded write (seven-track) for 667/677 tapes.
2IL	Label read/write for 66x/67x tapes.
2IP	66x/67x tape positioning.
2IR	66x read driver.
2IT	67x read driver.
2IW	66x write driver.
2IX	67x write driver.

Tape error recovery drivers:

3IE	Error diagnosis for 66x/67x tapes.
3IR	Read recovery for 66x/67x tapes.
3IW	Write recovery for 66x/67x tapes.

If the file device code is for a nonallocatable device, CIO loads an I/O driver into its PP to perform the actual I/O. The overlay selected is determined by the operation requested. For example, if a user issues a request to read data from a file on a standard format seven-track tape from a 667 tape unit, CIO calls the overlays 1IT, 2IR, and 3II into its PP. 2IR reserves one of the hardware channels connected to the equipment. It then issues the function codes to connect the controller and tape drive. 2IR issues functions to transmit one PRU of data from the tape drive over the data channel.

2IR accumulates the PRU of data in a PP buffer. When the entire PRU is transmitted or an end-of-record (short PRU) is encountered, 2IR picks up the pointers to the circular buffer in CM from the FET. 2IR continues to transfer PRUs of data from the tape through the PP buffer to the circular buffer until the buffer is full or an end-of-record is encountered. 2IR and 1IT update the PRU count in the file FNT, release the channel, set completion bits in the FNT and FET, and drop out.

Tables 5-1 through 5-12 list the logical sequence of events during various CIO tape operations.

TABLE 5-1. READ MACRO LOGICAL SEQUENCE

Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded	Sequence of Events
x	x					1. Exit if not enough room in buffer for one maximum size physical record.
		x	x	x	x	2. Exit if not enough room in buffer for MLRS words.
x	x	x	x			3. Read one physical record into PP.
				x	x	4. Read one physical record into CM.
x	x					5. If physical record exceeds maximum allowable size, return error status DEVICE CAPACITY EXCEEDED and perform error procedures.
		x	x	x	x	6. If physical record exceeds maximum logical record size, return error status DEVICE CAPACITY EXCEEDED and perform error procedures. If a long record is encountered, excess information is discarded without notification to user.
x	x	x	x	x	x	7. If end-of-file mark is read, perform end-of-file mark procedures.
x	x	x	x	x	x	8. If noise records are encountered, go to 3.
x	x	x	x	x	x	9. If parity error, perform parity procedures.
		x	x	x	x	10. If end-of-tape reflective spot is encountered and tape is unlabeled, perform end-of-reel procedures.
x	x					11. If short PRU is read, strip level number.
x	x					12. If zero length PRU is read, go to 21.

TABLE 5-1. READ MACRO LOGICAL SEQUENCE (Contd)

Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded	Sequence of Events
	x		x			13. When 6681 is present, convert data in PP from BCD to display code.
					x	14. When 6681 is present, convert data in CM from external BCD to display code.
	x					15. Convert 1632 line terminator to 0000.
x	x	x	x			16. Transmit data to CM.
x	x	x	x	x	x	17. Update IN.
x	x					18. Fetch OUT from CM.
		x	x	x	x	19. Place in word 7 of FET the number of unused bits in the last data word.
x	x					20. If full PRU, go to 1.
x	x	x	x	x	x	21. If last record was level 17 or tape mark, set end-of-file status.
x	x	x	x	x	x	22. Set end-of-record in status field of FET and exit.

TABLE 5-2. READN MACRO LOGICAL SEQUENCE

S Binary	S Coded	L Binary	L Coded	Sequence of Events
x	x	x	x	1. Fetch size of MLRS from word 7 of FET.
x	x	x	x	2. Exit if not enough room in circular buffer for one logical record plus header word. Buffer size must be greater than record length plus one (header word) to avoid OUT equal to IN when buffer is full.
x	x			3. Read one physical record into PP.
		x	x	4. Read one physical record into CM.
x	x			5. If physical record exceeds maximum allowable size, return error status DEVICE CAPACITY EXCEEDED and perform error procedures.
		x	x	6. If logical record exceeds MLRS, return error status DEVICE CAPACITY EXCEEDED and perform error procedures.
x	x	x	x	7. If end-of-file (tape mark) is read, perform end-of-file mark procedures. Go to 18.
x	x	x	x	8. If noise records are encountered, go to 3.
x	x	x	x	9. If parity error is encountered, perform parity procedures.
x	x	x	x	10. If end-of-tape reflective spot is encountered on unlabeled tape, perform end-of-reel procedures.
	x			11. When 6681 is present, convert data in PP from BCD to display code.
			x	12. When 6681 is present, convert data in CM from BCD to display code.
x	x			13. Transmit data to CM.
x	x	x	x	14. Update IN in PP memory.
x	x	x	x	15. Place length of record and number of unused bits in last data word in buffer header word.
x	x	x	x	16. Update IN.
x	x	x	x	17. Fetch OUT.
x	x	x	x	18. If last record is tape mark, set end-of-file status and exit.
x	x	x	x	19. Go to 2.

TABLE 5-3. READSKP MACRO LOGICAL SEQUENCE

Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded	Sequence of Events
x	x	x	x			1. Read one physical record into PP.
x	x	x	x			2. If physical record exceeds maximum allowable size (512 CM words, and so on), return error status DEVICE CAPACITY EXCEEDED and perform error procedures.
				x	x	3. Read one physical record directly from tape to CM buffer, stopping without error when available buffer space is full.
x	x	x	x	x	x	4. If end-of-file (tape mark) is read, perform end-of-file mark procedures.
x	x	x	x	x	x	5. If noise records encountered, go to 1.
x	x	x	x	x	x	6. If parity error is encountered, perform parity procedures.
		x	x	x	x	7. If end-of-tape reflective spot is encountered on unlabeled tape, perform end-of-reel procedures.
x	x					8. If short PRU is read, strip level number.
x	x					9. If zero length PRU is read, go to 10.
	x		x			10. When 6681 is present, convert data in PP from BCD to display code.
					x	11. When 6681 is present, convert data in CM from BCD to display code.
	x					12. Convert 1632 line terminator to 0000.
x	x	x	x			13. Transmit data to CM. If record exceeds circular buffer, stop without error at buffer full.
		x	x	x	x	14. Place number of unused bits in last data word in word 7 of FET.
x	x	x	x	x	x	15. Update IN.
x	x					16. Fetch OUT from CM.
x	x					17. If any unused space exists in circular buffer, go to 1.

TABLE 5-3. READSKP MACRO LOGICAL SEQUENCE (Contd)

Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded	Sequence of Events
x	x					18. If last record is full PRU, set n to 1 and proceed to SKIPF.
		x	x	x	x	19. If L is less than 17, set L to 0.
		x	x	x	x	20. If record is end-of-file mark (tape mark), assume level is 17.
x	x	x	x			21. If level number is less than 1, set n to 1 and proceed to SKIPF.
				x	x	22. If level number is less than L, set n to 1 and skip to first end-of-file mark (tape mark).
x	x	x	x	x	x	23. If last record is level 17, set end-of-file status and exit.
x	x	x	x	x	x	24. If last record is not level 17, return end-of-record status and exit.

TABLE 5-4. RPHR MACRO LOGICAL SEQUENCE

Standard Binary	Standard Coded	Sequence of Events
x	x	1. Set OUT to IN.
x	x	2. Read one physical record.
x	x	3. If end-of-file mark is read, perform end-of-file procedures.
x	x	4. If noise records are encountered, go to 2.
x	x	5. If parity error is encountered, perform parity procedures.
x	x	6. If zero length PRU is read, go to 10.
x	x	7. Transmit data to CM.
x	x	8. Update IN.
x	x	9. If last record is level 17 or tape mark, set end-of-file status.
x	x	10. Exit.

TABLE 5-5. WRITE MACRO LOGICAL SEQUENCE

Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded	Sequence of Events
x	x					1. Exit if not full PRU.
		x	x	x	x	2. If data from OUT to IN exceeds maximum logical record size from FET, return DEVICE CAPACITY EXCEEDED and perform error procedures.
		x	x	x	x	3. Fetch number of unused bits in last data word from FET and adjust record length. If record length constitutes a noise record, return DEVICE CAPACITY EXCEEDED and perform error procedures.
x	x					4. Read one PRU of data starting at OUT from CM to PP.
		x	x			5. Read data contained between OUT and IN from CM to PP. Adjust by unused bit count.
	x		x			6. When 6681 is present, convert from display code to BCD in PP memory.
					x	7. When 6681 is present, convert from display code to BCD in CM.
	x					8. Convert zero byte line terminator to 1632.
x	x	x	x			9. Write record to tape.
				x	x	10. Write, from CM to tape, data contained between OUT and IN, adjusted by unused bit count.
					x	11. When 6681 is present, convert data in CM buffer back to display code.
x	x	x	x	x	x	12. If parity error is encountered, perform parity procedures.
x	x	x	x	x	x	13. If end-of-tape reflective spot is encountered, perform end-of-reel procedures.
x	x	x	x	x	x	14. Update OUT.
		x	x	x	x	15. Exit.
x	x					16. Fetch IN from CM.
x	x					17. Go to 1.

TABLE 5-6. WRITER MACRO LOGICAL SEQUENCE

Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded	Sequence of Events
		x	x	x	x	1. If IN equals OUT, exit.
x	x					2. If PRU is not full, insert level number in PP buffer.
		x	x	x	x	3. If data from OUT to IN exceeds maximum logical record size from FET, return DEVICE CAPACITY EXCEEDED and perform error procedures.
		x	x	x	x	4. Fetch number of unused bits in last data word from FET and adjust record length. If record length constitutes a noise record, return DEVICE CAPACITY EXCEEDED and perform error procedures.
x	x					5. Read one PRU starting at OUT or between OUT and IN, whichever is smaller, from CM to PP.
		x	x			6. Read data between OUT and IN from CM to PP. Adjust by unused bit count.
	x		x			7. When 6681 is present, convert from display code to BCD in PP memory.
					x	8. When 6681 is present, convert from display code to BCD in CM.
	x					9. Convert zero byte line terminator to 1632.
x	x					10. If IN equals OUT, write zero length record. Go to 12.
x	x	x	x			11. Write record to tape.
		x	x	x	x	12. Write data between OUT and IN from CM to tape, adjust by unused bit count.

TABLE 5-6. WRITER MACRO LOGICAL SEQUENCE (Contd)

Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded	Sequence of Events
					x	13. When 6681 is present, convert data in CM buffer to display code.
x	x	x	x	x	x	14. If parity error is encountered, perform parity procedure.
x	x	x	x	x	x	15. If end-of-tape reflective spot is encountered, perform end-of-reel procedures.
x	x	x	x	x	x	16. Update OUT.
		x	x	x	x	17. Exit.
x	x					18. If full PRU is not written, exit.
x	x					19. Go to 1.

TABLE 5-7. WRITEF MACRO LOGICAL SEQUENCE

Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded	Sequence of Events
x	x					1. If no data from OUT to IN, go to 20.
		x	x	x	x	2. If no data from OUT to IN, go to 17.
x	x					3. If not full PRU, insert level number 0.
		x	x	x	x	4. If data from OUT to IN exceeds maximum logical record size, return DEVICE CAPACITY EXCEEDED and perform error procedures.
		x	x	x	x	5. Fetch number of unused bits in last data word from FET and adjust record length. If record length constitutes a noise record, return DEVICE CAPACITY EXCEEDED and perform error procedures.

TABLE 5-7. WRITEF MACRO LOGICAL SEQUENCE (Contd)

Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded	Sequence of Events
x	x					6. Fetch one PRU of data starting at OUT or fetch data between OUT and IN, whichever is smaller, from CM to PP.
		x	x			7. Read data contained between OUT and IN from CM to PP. Adjust by unused bit count.
	x		x			8. When 6681 is present, convert from display code to BCD in PP memory.
					x	9. When 6681 is present, convert from display code to BCD in CM.
	x					10. Convert zero byte line terminator to 1632.
x	x	x	x			11. Write record to tape.
		x	x	x	x	12. Write data between OUT and IN from CM to tape, adjust by unused bit count.
					x	13. When 6681 is present, convert data in CM buffer to display code.
x	x	x	x	x	x	14. If parity error is encountered, perform parity procedures.
x	x	x	x	x	x	15. If end-of-tape reflective spot is encountered, perform end-of-reel procedures.
x	x	x	x	x	x	16. Update OUT.
		x	x	x	x	17. Write end-of-file mark and exit.
x	x					18. If full PRU is not written, write zero length level 17 record and exit.
x	x					19. Go to 3.
x	x					20. If last operation is WRITE, write zero length PRU.
x	x					21. Go to 17.

TABLE 5-8. WRITEN MACRO LOGICAL SEQUENCE

S Binary	S Coded	L Binary	L Coded	Sequence of Events
x	x	x	x	1. If OUT equals IN, exit.
x	x	x	x	2. Fetch header word from OUT. Set PPOUT to OUT plus 1. Set PPIN to PPOUT plus the number of CM words in logical record. If PPIN has passed IN, exit.
x	x			3. If data from PPOUT to PPIN exceeds maximum physical record size, return DEVICE CAPACITY EXCEEDED and perform error procedures.
x	x	x	x	4. Adjust record length by number of unused bits in last data word (from header word). If noise record is encountered, return DEVICE CAPACITY EXCEEDED and perform error procedures.
x	x			5. Fetch data contained between PPOUT and PPIN. Adjust by unused bit count.
	x			6. When 6681 is present, convert from display code to BCD in PP memory.
			x	7. When 6681 is present, convert from display code to BCD in CM.
x	x			8. Write record to tape.
		x	x	9. Write data between OUT and IN from CM to tape. Adjust by unused bit.
			x	10. When 6681 is present, convert data in CM buffer back to display code.
x	x	x	x	11. If parity error is encountered, perform parity procedures.
x	x	x	x	12. If end-of-tape reflective spot is encountered, perform end-of-reel procedures.
x	x			13. Update PPOUT.
x	x	x	x	14. Update OUT. Fetch IN. Go to 1.

TABLE 5-9. WPHR MACRO LOGICAL SEQUENCE

Standard Binary	Standard Coded	Sequence of Events
x	x	1. If IN equals OUT, exit.
x	x	2. Fetch data from OUT to IN.
x	x	3. Write record to tape.
x	x	4. If parity error is encountered, perform parity procedures.
x	x	5. If end-of-tape reflective spot is encountered, perform end-of-reel procedures.
x	x	6. Update OUT and exit.

TABLE 5-10. SKIPF MACRO LOGICAL SEQUENCE

Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded	Sequence of Events
x	x	x	x	x	x	1. If n is 0, set n to 1.
		x	x	x	x	2. If L is less than 17, interpret as L equals 0.
x	x	x	x	x	x	3. Read a physical record.
x	x	x	x	x	x	4. If noise record is encountered, go to 3.
		x	x	x	x	5. If end-of-tape reflective spot is encountered on unlabeled tape, perform end-of-reel procedures.
x	x					6. If record is full PRU, go to 3.
		x	x	x	x	7. If end-of-file mark is encountered on unlabeled tape, assume level number equals 17.
		x	x	x	x	8. If record is not end-of-file mark, assume level number equals 0.
x	x	x	x	x	x	9. If end-of-file mark encountered on labeled tape, perform end-of-file procedures.
x	x	x	x	x	x	10. If level number is less than L, go to 3.
x	x	x	x	x	x	11. Subtract 1 from n. If n is not equal to 0, go to 3.
x	x	x	x	x	x	12. Return end-of-record to status. If last level number was 17, return end-of-file to status. Exit.

TABLE 5-11. SKIPB MACRO AND BKSP MACRO LOGICAL SEQUENCE

Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded	Sequence of Events
x	x	x	x	x	x	1. If n is 0, set n to 1.
		x	x	x	x	2. If L is less than 17, interpret as L equals 0.
x	x	x	x	x	x	3. If reel is at beginning of data (either physical load point or zero physical record count), set beginning-of-information and exit.
x	x	x	x	x	x	4. Read one physical record backward.
x	x	x	x	x	x	5. If noise record is encountered, go to 4.
x	x					6. If record is full PRU, go to 3.
x	x					7. If this is first read backward, go to 3.
x	x					8. Position forward over short PRU.
		x	x	x	x	9. If end-of-file mark is encountered, assume level number equals 17. Otherwise, assume level number equals 0.
x	x	x	x	x	x	10. If level number is less than L, go to 3.
x	x	x	x	x	x	11. Subtract 1 from n. If n is not equal to zero, go to 3.
x	x	x	x			12. Exit.

TABLE 5-12. BKSPRU MACRO LOGICAL SEQUENCE

Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded	Sequence of Events
x	x	x	x	x	x	1. If at load point or if PRU count equals 0, set beginning-of-information in FET and exit.
x	x	x	x	x	x	2. Backspace one physical record.
x	x	x	x	x	x	3. Subtract 1 from n. If n is not equal to 0, go to 1.
x	x	x	x	x	x	4. Exit.

Allocatable Device I/O

Most files in the system are stored on allocatable devices. The system library ZZZZZ04 is stored on an allocatable device known as the system device. Each time a PP overlay or a CP program not resident on CM is to be loaded from the library, I/O must be performed on the system device. The job and system dayfiles and the CE error files are stored on allocatable devices, as are all input and output queue files and all files created by CIO.

A request for I/O on a mass storage allocatable device must be placed in a table, called the request stack. The stack is searched, and the request which requires the minimum amount of overhead to access the data is chosen. Overhead involves switching head groups on the disk or physically moving heads. By using a priority-incrementing scheme for scheduling disk I/O, overhead is kept to a minimum.

All mass storage devices are connected to controllers which are connected to hardware channels of the computer. For some disk devices, the controller and the disk unit form one piece of equipment. In most cases, however, the controller and the disk are physically separate units. All mass storage devices connected to a single controller must be of the same type.

READC

The READC function is intended primarily for system use with mass storage files. It applies to all mass storage devices. Since READC uses intersector time to the maximum while reading high-speed mass storage devices, it does not include checks for erroneous programming and control words. READC should be used by system programmers only. The format is

```
READC lfn,recall
```

READC transmits PRUs continuously to the circular buffer, with a control word preceding each PRU. Reading continues until one of the following occurs.

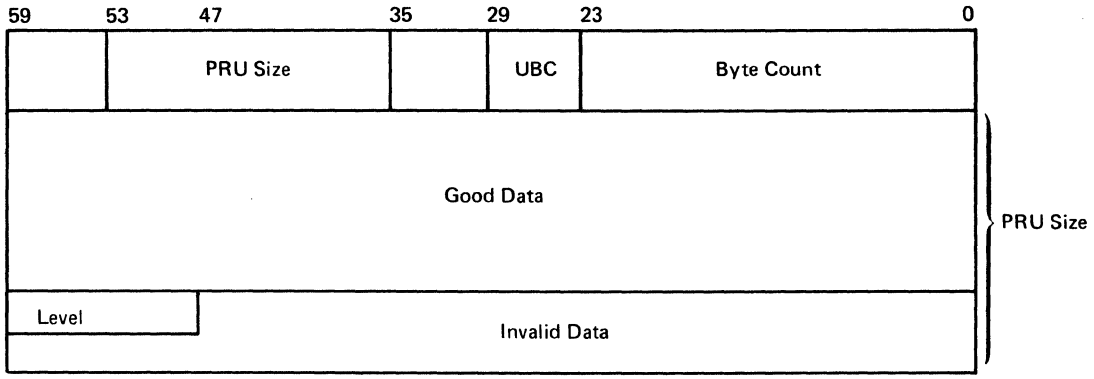
- The buffer does not have enough room for the next PRU and its control word.
- An error condition occurs.
- End-of-information is encountered.

Code and status on completion (x depends on file mode):

00020x	Normal completion.
0ee20x	Error code ee.
74123x	EOI.

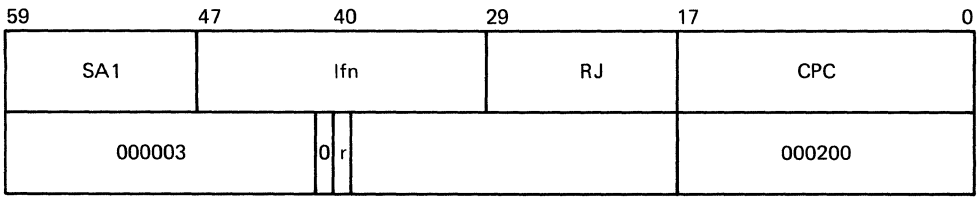
On mass storage, the same amount of data is transmitted for every PRU: the control word and one device standard PRU. The last 12 bits if the control word and the entire standard PRU length are exactly the physical data recorded on the device, including system control information.

The following diagram shows the format of the PRU.



PRU size	64 CM words in each PRU on the device.
UBC	Unused bit count; always 0.
Byte count	Count of the number of 12-bit bytes of data. It must be equal to 5 times the number of CM words occupied by the data. The value is recorded on disk as 12 bits, but expanded here to 24 bits.
Level	System logical record level number. If byte count divided by 5 is a full PRU, level does not exist.

The READC macro generates the following code.



READLS

The READLS function applies only to mass storage files. READLS reads several random records into the file circular buffer according to the list of direct access addresses provided by the user. No information in the buffer reveals boundaries. READLS should be used by system programmers only.

The format is

READLS lfn,recall

Before READLS is called, bits 17 through 0 of FET+6 should be set to the address of the list of addresses to be read. Reading continues until one of the following occurs.

- The list of addresses is exhausted.
- End-of-information is encountered while reading a record.
- The buffer is full.
- An error condition occurs.
- The request is discontinued for device repositioning.

Code and status on completion:

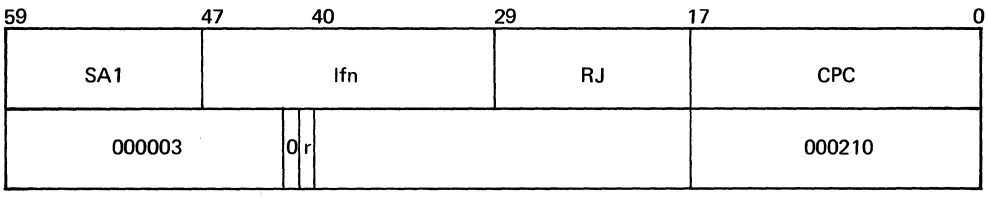
Bits 3 and 4 contain 01, 10, or 11, giving the status at the point where the operation terminated (10 is end-of-record, 11 is end-of-file). The operation terminates with EOI status if the last PRU of the file is read and no EOR or EOF occurs. The contents of bits 8 through 5 do not pertain to this description.

The address pointer is updated by the system when READLS terminates so that the function can be reissued by the user without the user changing the pointer. The updated pointer reflects the next record to be read. If reading stopped in the middle of a record, the pointer reflects the next position to be read.

The words in the list of addresses to be read can have one of two formats, but the formats of the entire list must be the same. A word of all zeros must terminate the list. Either of the following formats can be used.

- Bits 59 through 36 contain a PRU number, the same as used in the system indexes. These are the numbers the system returns to the record request/return information fields (bits 29 through 0 of word 7) of the FET when records are written on a mass storage device. Bits 35 through 0 are zero. A user list in this format is converted by the system to the next format.
- Bits 35 through 0 contain the internal direct access address (RBTA/RBB/PRU) address RBT.

The READLS macro generates the following code.



WRITEC (Continuous Write)

The WRITEC function is intended primarily for system use. Since it uses intersector time to the maximum on high-speed mass storage devices, it does not include checks for erroneous programming and control words. The format is

WRITEC lfn,recall

WRITEC transmits PRUs from the circular buffer to a mass storage device. Each PRU in the buffer must be preceded by a control word. Writing continues until one of the following occurs.

- The buffer is empty.
- An error occurs.

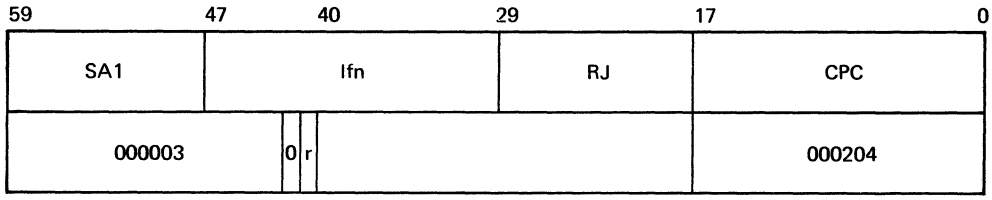
The diagram of the PRU and control word appears with the discussion of READC. PRU size must be standard 64 CM words for mass storage; if not, serious errors result. The 24 low-order bits of the control word and the full CM words are written to the device.

Byte count is the count in 12-bit bytes of good data in the PRU and must be a multiple of 5. If byte count/5 is less than the device PRU size, the next 12-bit byte after the good data is the system level number in binary. Level must be in the range $0 \leq \text{level} \leq 17$ octal.

The unused bit count field (UBC) in the header word represents the number of unused bits in the last data word of a PRU. Since mass storage files are in system logical record format, data resolution is to the nearest full CM word so that the UBC field is always zero. This field is reserved for future expansion.

If the file has any data at all, it must be terminated by some end-of-logical-record; level 17 octal must appear as a zero-length logical record.

The WRITEC macro generates the following code.



Stack Processor

A stack processor consists of the CMR manager CP.SPM, the PP program 1SP, and its various overlays. Basically, the components of a stack processor and their functions are:

CP.SPM	The stack processor manager.
OV.1S5	Examines DST ordinal and loads 3DO if the DST ordinal is 1; otherwise, loads 1SP.
OV.1SP	The stack processor driver supervisor.
OV.1RN	Requests/releases RBT storage; merges released chains to the empty chain.
OV.3DO	Assigns a device and an RBT word pair to a new or overflowing file.
OV.4DO	Processes stack requests which require no device access, specifically O.SKPF/O.SKPB with skip counts of 777777_8 and O.BPRU.

If SPM finds a request for an 819 disk, it calls HSP, the 819 disk stack processor (refer to 819 Disk I/O Processing). The remainder of this discussion applies to only 844 and 885 disk drives.

SPM is called to enter, terminate, and reissue stack requests. SPM performs request scheduling, device optimization, and all RMS I/O functions except file assignment (performed by 3DO), non-I/O skipping (performed by 4DO), and physical I/O (performed by 1SP/1SQ). 1SP and 1SQ are RMS device drivers. 1SP is used with 7054 and 7154 controllers; 1SQ is used with 7155 controllers. Each program handles one request at a time.

A PP or CP system routine initiates I/O by placing a stack request (first two words of the format) in the first two words of its associated communication area (T.PPCn for PPn) and calling the CP monitor function SPM (M.ICE/EX.SPM). SPM picks up the stack request from the communication area, generates the third word, puts the three-word stack request into the request stack area, and links it to the proper DST chain. If the priority bit in the stack request is set, that stack request becomes the first stack request in the proper DST chain. The priority bit should be used with discretion; otherwise, priority stack requests could be pushed down in the DST chain.

To perform RMS I/O, SPM selects a stack request and assigns it to an RMS device driver. The driver must be assigned to a DST ordinal that represents an access to the specified RMS device. If a stack request is received and no device driver is operational, the request is entered by SPM and an M.ISP request (initiate stack processor driver) is sent to MTR. If the MTR communication path is busy, no request is posted and the device driver is assigned later during the normal MTR DST scan for work outstanding with no PP assigned. The PP is assigned by MTR by using the second word of the DST as the PPIR entry to call the proper PP routine.

When a device driver comes up, it initializes itself for the proper device (844 or 885). Channel and equipment numbers are in the PPIR. Initialization is completed and SPM is called for a stack request. (A special procedure is used during EDITLIB operations.) The device driver performs the I/O requested, obtaining field access as necessary, and at I/O completion, returns the stack request to SPM for termination processing. If there is another stack request outstanding for this driver, SPM assigns the request to the driver. Otherwise, it assigns the driver to idle. A subsequent stack request for this device need only be placed in this device driver's PP communication area to start I/O.

Device dependent code for 844 devices is in overlay 3SY which is contained in common deck RMSY. Device dependent code for 885 devices is in overlay 3SJ which is contained in common deck RMSJ. RMSY and RMSJ are in PL1A.

In addition to the overlay area for device dependent code, the stack processor has an overlay area for the executive routine that performs the other side of the data transfer or a nondata transfer related function. There are currently four such executive routines.

- CM input/output.
- PP input/output.
- Positioning.
- ECS I/O via CM buffers or DDP.

As each stack request is processed, the appropriate executive routine is loaded if necessary. No load occurs if the proper routine is already present.

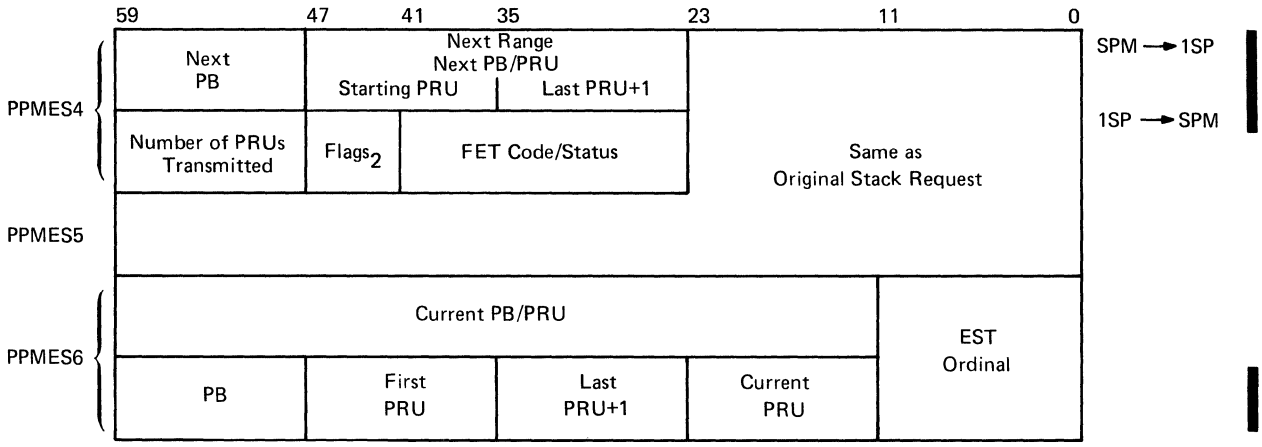
MTR ensures that at least one PP contains a stack processor at all times, or that one is reserved for that purpose. This requirement is necessary so that it is always possible to load a PP overlay that resides on mass storage. To avoid unnecessary loading and dropping of stack processors, a stack processor remains in the PP until MTR needs a free PP. At that point, MTR sets a PP request flag for SPM. Idle stack processors check this flag in their idle loop and call SPM if it is set. SPM then issues a drop order to the first idle stack processor it finds. If all stack processor PPs are busy, no action is taken until one is about to become idle. When the monitor drop flag is set, SPM issues a drop order to the PP instead of an idle order.

Stack Request Formats

There are two kinds of stack request formats, external and internal. The external formats are used by PP and CP routines in submitting I/O initiation requests to SPM through the communication area of the caller. The internal format is used for SPM/stack processor communication through the stack processor communication area. It is different from the external format because the stack processor does not work with RBs but with PBs. SPM converts RBs to PBs for 1SP and RBs to physical addresses for 1SQ. This is done so that stack processors do not have to access the RBT chains.

The external stack request format consists of three words. The first two words are supplied by the caller; the third word is added by SPM. The first word specifies the type of stack request (order code, interlock flag, direct) and source of RMS file or device information (FST pointer, RBT chain pointer, equipment pointer). The second word specifies the other side of the request (PP, CM, ECS). The third word is SPM internal information. External stack request formats are summarized in the Request Stack Entry illustration in appendix B.

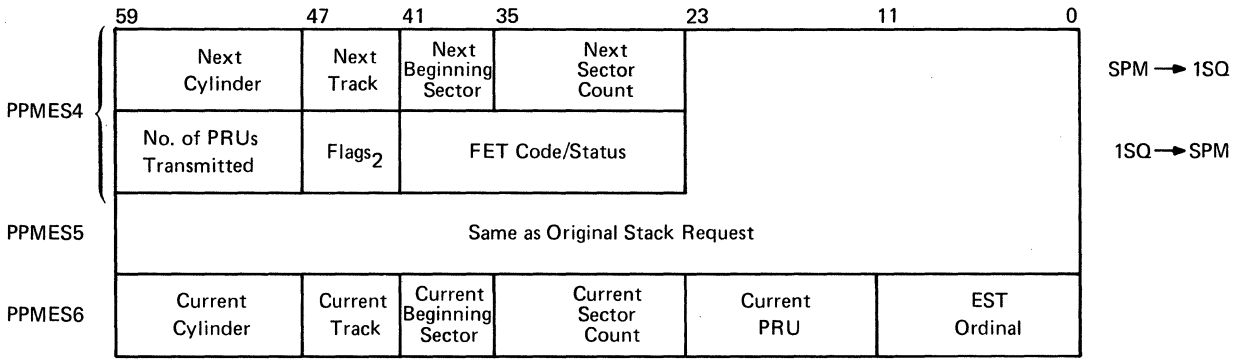
The internal stack request format consists of three words. It is part of the SPM/stack processor interface at communication area locations PPMES4 through PPMES6. The format varies depending on whether 1SP or 1SQ is used. The following diagram shows the format of the internal stack request for the SPM-1SP interface.



The first and third words contain the next PB/PRU and current PB/PRU transfer information. The next PB/PRU is the next one to be processed by 1SP and, in the case of skipping backward, is the PB/PRU which logically precedes the current PB/PRU being processed. The NXTPB function of SPM updates the next PB/PRU in an overlap synchronous manner so that 1SP can continue to do I/O and have the next PB/PRU information available when needed. 1SP uses the next PB/PRU information to update the current PB/PRU in word 3 and thus determine the current operation. The process is then repeated.

The second word and rightmost two bytes of the first word are the same as the original stack request. The first PRU and last PRU fields specify the range of the request within the specified PB; the current PRU is the one currently being processed. The $flags_2$ field (bits 47 through 42) is described under SPM-1SQ Interface in this section.

The following diagram shows the format of the internal stack request for the SPM-1SQ interface.



The SPM-1SQ interface is the same as the SPM-1SP interface except physical addresses are passed instead of PBs. The $flags_2$ field (bits 47 through 42 of the first word) is described under SPM-1SQ Interface in this section.

Stack Processor Order Codes

Order codes used in mass storage I/O request stack entries differ from those used in the CIO code/status fields of FET and FST entries. The three groups of codes correspond to the formats for the second word of a request stack entry. Order codes, standard system symbols, and functions are as follows.

CM Read/Write Order	Description
00 O.READ	Corresponds to READ macro, CIO codes 010 and 012. Read data from device to CM until end-of-information is reached, a short PRU is read, or next PRU does not fit into the buffer. (Refer to table 5-1.)
01 O.RDSK	Corresponds to READSKP macro, CIO codes 020 and 022. Read as for O.READ until end-of-information is reached, a short PRU is read, or the CM buffer is completely full. Change to O.SKF with n=1 unless reading was stopped by a short PRU with record level greater than or equal to request level. (Refer to table 5-3.)
02 O.RCMPR	No corresponding CPC macro or CIO code. Read as for O.READ, but do not transmit the first n CM words of the PRU. n is the number of CM words in the PRFX (77) table. Used for loading programs from a system library in which the first n words represent the PRFX (77) table in each record and contain information of interest only to EDITLIB and deadstart.
03 O.RDNS	Corresponds to the READNS macro, CIO codes 250 and 252. Read data from device into CM buffer until end-of-information is reached, a short PRU with record level 16 or 17 has been read, or next PRU does not fit into CM buffer. Used by loader when reading a relocatable binary field, since it does not stop at an ordinary end of logical record.

CM Read/Write Order		Description
04	O.WRT	Corresponds to WRITE macro, CIO codes 014 and 016. Write data from CM to device until CM buffer contains less than a full PRU. (Refer to table 5-5.)
05	O.WRTR	Corresponds to WRITER macro, CIO codes 024 and 026. Write data from CM until CM buffer is empty, ending with short PRU (zero-length if necessary) with level number specified in request. If EOF flag bit is set, this corresponds to the WRITEF system macro, CIO codes 034 and 036. Same action as for WRITER macro, but short PRU is followed by zero-length level 17 record (logical end of file mark). (Refer to table 5-6.)
06	O.RMR	Corresponds to READLS macro, CIO codes 210 and 212. Read several records for which disk addresses are given in a table; pointer to table is in FET+8. Address of table must be in the user's field length. Read records until EOR is reached, buffer capacity is exceeded, or the addresses are exhausted.
20	O.RCTNU, O.RCTU	Corresponds to READC macro, CIO codes 200 and 202. Provides nonstop reading from device to CM without releasing/reloading PP between logical records. Buffer must provide space for at least two records and their header words.
24	O.WCTNU, O.WCTU	Corresponds to WRITEC macro, CIO codes 204 and 206. Provides nonstop writing from CM to device without releasing/reloading PP between logical records. User's buffer must contain at least two records. Writing stops when buffer is empty.

PP Read/Write Order		Description
10	O.RDP	Same as O.READ except read data from device into requesting PP's memory.
11	O.RDPNP	Same as O.RCMPR except read data from device into requesting PP's memory. Used for loading mass storage resident PP programs and overlays.
14	O.WRP	Same as O.WRT except write data from requesting PP's memory to device.
15	O.WRPR	Same as O.WRTR except write data from requesting PP's memory to device.

Positioning Order		Description
12	O.SKF	Corresponds to SKIPF macro, CIO codes 240 and 242. Skip forward until n short PRUs, with level greater than or equal to the level specified in the request, have been read or until end-of-information is reached. With n=777777, the file is positioned at end-of-information. (Refer to table 5-10.)

Positioning Order		Description
13	O.SKB	Corresponds to SKIPB macro, CIO codes 640 and 642. Skip backward one or more PRUs until n short PRUs, with level greater than or equal to the level specified in the request, have been read, then move forward over the last of these. With n=777777, the file is positioned at beginning of information (rewound). (Refer to table 5-11.)
16	O.BPRU	Corresponds to the BKSPRU macro, CIO codes 044 and 046. Skip backward n PRUs. This repositioning is by PRUs rather than logical records. (Refer to table 5-12.)
17	O.RCHN	Release allocatable storage and RBTs (processed by SPM). For permanent file set RBT chains, the first word pair to be evicted must be a first word pair or an overflow word pair.

SPM-Stack Processor Communication Order†		Description
35	O.IDLE	SPM has determined that there is no work for this stack processor and that MTR is not requesting a PP. The stack processor checks the order code in its idle loop, waiting for SPM to assign a stack request. When another order code appears, the stack processor processes it.
36	O.DROP	SPM has determined that this stack processor is either idle or going to become idle, and that MTR is requesting a PP. The stack processor drops out.
37	O.SEEK	SPM has stack requests for this stack processor but none are on-cylinder. The stack processor issues overlap seeks (up to five), monitors the units, and reserves the first one to come on-cylinder. Overlap seeks can also be issued with other stack requests. In this case, no monitoring is done. The seeks are issued before stack request execution. Status is taken after stack request execution and returned to SPM with the stack request.

Table 5-13 is a summary of stack processor orders.

TABLE 5-13. STACK PROCESSOR ORDERS

Octal Code	System Symbol	Order Function
00	O.READ	Read into CM.
01	O.RDSK	Read-skip into CM.
02	O.RCMPR	Read into CM, drop first three CM words.

† Internal format only.

TABLE 5-13. STACK PROCESSOR ORDERS (Contd)

Octal Code	System Symbol	Order Function
03	O.RDNS	Read nonstop.
04	O.WRT	Write from CM.
05	O.WRTR	Write EOF/EOR from CM.
06	O.RMR	Read multiple records to CM.
10	O.RDP	Read into PP memory.
11	O.RDPNP	Read into PP, drop first three CM words.
12	O.SKF	Skip forward.
13	O.SKB	Skip backward.
14	O.WRP	Write from PP memory.
15	O.WRPR	Write EOF/EOR from PP memory.
16	O.BPRU	Backspace n PRUs.
17	O.RCHN	Evict.
20	O.RCTNU, O.RCTU	Read nonstop (comparable to tape READN).
24	O.WCTNU, O.WCTU	Write nonstop (comparable to tape WRITEN).
35	O.IDLE	Wait for stack request.
36	O.DROP	Drop PP.
37	O.SEEK	Issue overlap seek.

Stack Processor-System Interface

MTR functions used by the stack processor are described in this section.

The following system tables are used by the stack processor.

Tables	Description
Control Point Areas	Contain control point error flag, storage move flag, RA, and FL fields. The stack processor accesses but never changes these fields.
DST	All fields of the DST entry whose ordinal is placed in the stack processor input register by MTR are used. SPM makes all DST changes except one made by MTR during stack processor assignment to a PP.

Tables	Description
EST	Mass storage flag, unloaded flag, off flag, and DST ordinal are checked but not altered.
FET	Code and status field in the first word and error processing flag in the second word are accessed by SPM. IN and OUT pointers in the third and fourth words are accessed by the stack processor. Code/status is marked busy (even value) before a request enters the stack and is marked complete (odd value) when the request is executed.
FST	RBT/RB/PRU position pointers in first word and code/status field in the second word are accessed by SPM. The code/status field has been processed the same as for the FET.
RMSBUF	Stack processor (1SP/1SQ) formats and stores RMS hardware error diagnostics in the RMSBUF three-word area for DSD to display.
RST	Request scheduling table is parallel to the request stack area and is used by SPM to hold request scheduling parameters.
DDT	First and last DAM ordinals, MST ordinal, and EST ordinal are used by SPM in determining RBR ordinal of a permanent file set member and whether or not it is mounted.
RBR Area	All the first header word, the EST ordinal, and available RB count bytes in the second header word are used by SPM. SPM assigns record blocks for a write request by searching the RBR table for available bits. When the request is terminated or reissued, SPM sets the corresponding bits in the RBR for all record blocks assigned for the write operation. SPM also clears RBR bits when record blocks are released and updates the available RB count.
RBT	All fields are used. The pseudo channel CH.RBT is reserved only when RBT word pairs are being removed from the RBT empty chain.
SCB	FIRST, IN, OUT, and LIMIT are accessed in the same manner as FET when transferring data between RMS and ECS.
R.STBMSK (PP resident)	Contains appropriate mask when calling R.STB; always returned to 7700 octal, its normal value.

The following system routines and programs are used by the stack processor.

PP Program	Description
1SX	Stack processor auxiliary program called by MTR request for tasks that the stack processor cannot handle or does not have time to do. For example, the stack processor does not issue dayfile messages, because if the dayfile buffer is full, 1SP/1SQ and MTR could loop endlessly waiting for each other.
CEM	Auxiliary program called by MTR request to handle CERFILE logging and dayfile messages for DDP errors encountered by 1SP/1SQ.

PP Program	Description
MTR	System monitor initially calls the stack processor (via 1S5), when a request has been made for an inactive DST entry, and performs various functions for the stack processor while it is processing the request.
7ID	Stack processor auxiliary program called by 1SX. 7ID informs the operator (via a flashing message at the bottom of the B display) that the job associated with control point x has an outstanding request for an idle device.

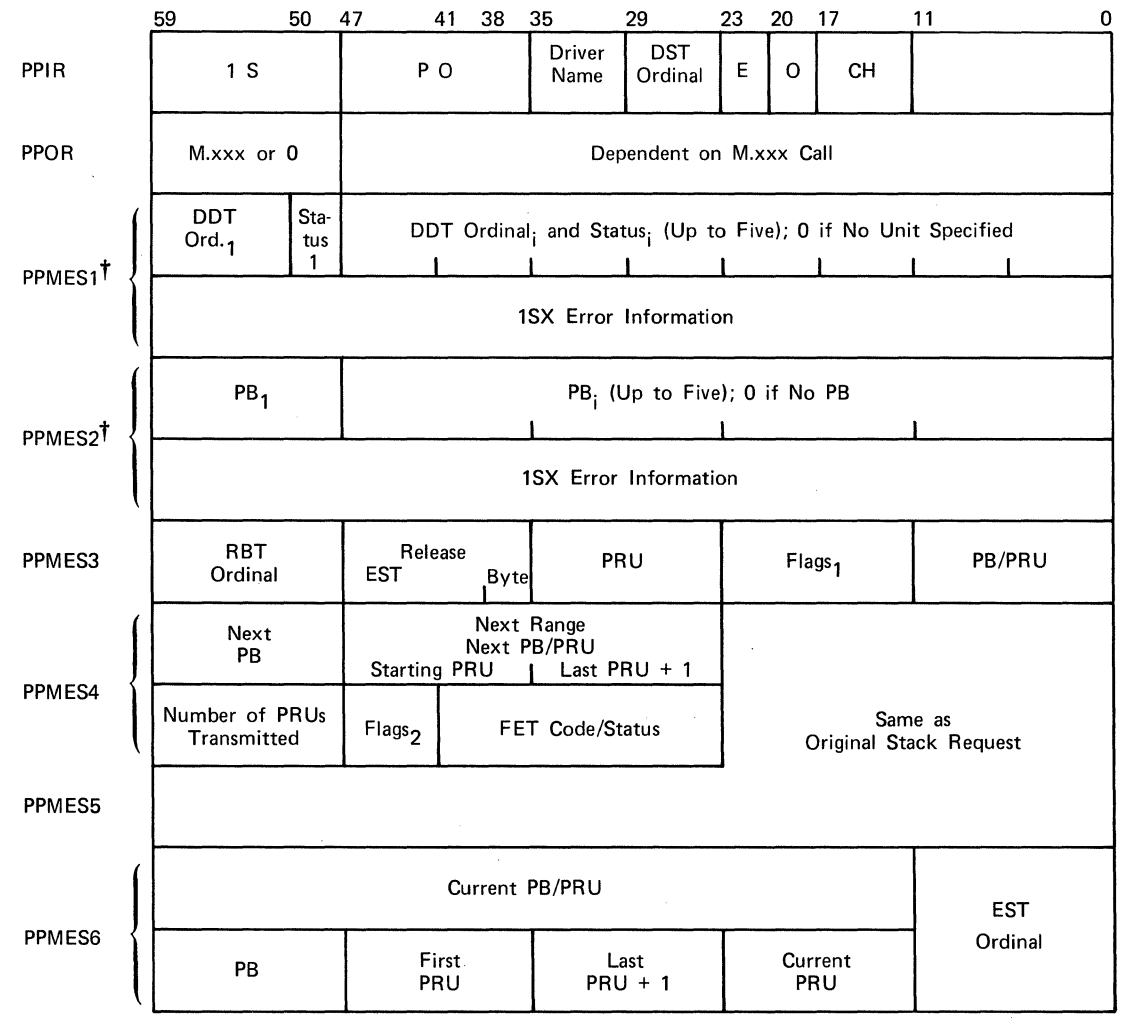
PP Resident Routine	Description
R.DCH	Releases a channel reservation.
R.IDLE	Entered when a stack processor releases its PP.
R.MTR	Used for all MTR functions other than to reserve or drop a channel.
R.TAFL	Terminates access to the control point field length. When necessary, it interlocks storage moves during execution of a request and, when a request is terminating (except at control point 0), switches the stack processor back to control point 0.
R.OVL	Loads driver overlay 3Sx.
R.RCH	Reserves a channel.
R.STB	Inserts controller equipment number into device function codes and channel number into I/O instructions.
R.TFL	Computes an absolute CM address, from one that is relative to a control point's RA, and checks whether or not a relative CM address is within the control point's FL.

Monitor Function	Description
M.DPP	Releases PP assignment.
M.SPM	Used by the stack processor to call SPM. The three executives are: • EX.SPRCL — Stack processor recall; terminate a stack request. • EX.STAT — Change status; get next stack request. • EX.NXTPB — Get next PB/PRU.
M.RCH	Used (rather than R.RCH) with zero in byte 4 to reserve pseudo channel CH.RBT only if it is immediately available.
M.RPJ	Used for calling 1SX to another PP.
M.KILL	Used when a bad monitor request has been made.

SPM-1SP Interface

The PP communication area through which SPM and 1SP communicate consists of a PP input register (PPIR), PP output register (PPOR), and a six-word PP message buffer (PPMES1-6). The PPIR, used to call 1SP, comes directly from the second word of the DST and contains initialization information such as drive, name, equipment, and channel number. The PPOR is used by 1SP to initiate SPM via the M.SPM monitor call in addition to other monitor calls. PPMES1-2 are used for up to five byte pairs (corresponding bytes of PPMES1-2) of overlap seek information and for the 1SX error information interface. PPMES3 is for SPM internal information although 1SP supplies the device PB/PRU count in the rightmost byte during initialization. PPMES4-6 is the internal format stack request that SPM gives to 1SP. The order field in PPMES4 is monitored by 1SP to determine what work is to be done. The $status_1$, $flags_1$, and $flags_2$ fields are described under SPM-1SQ Interface in this section.

The format of the 1SP communication area is:

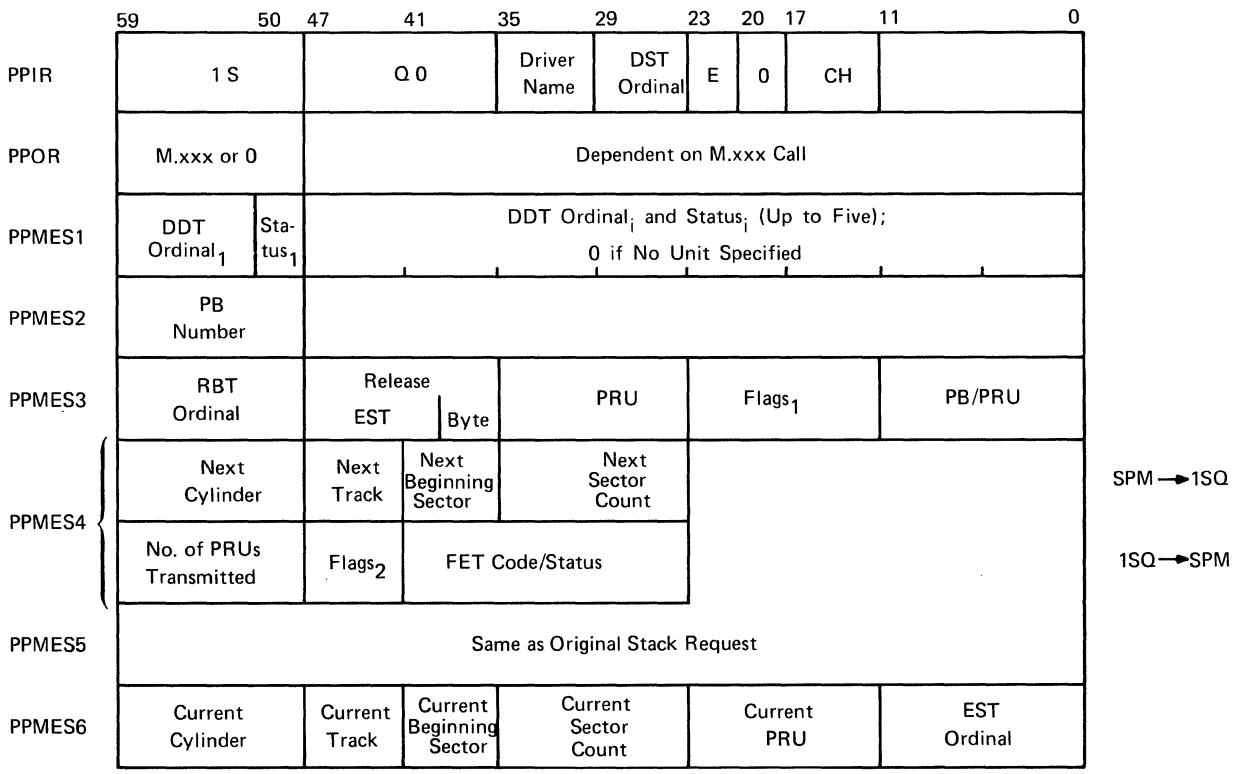


†Example shows three units specified for overlap seek with zero terminator.

SPM-1SQ Interface

SPM and 1SQ use the PP comunication area similar to SPM and 1SP. The difference is that SPM sends and receives physical disk addresses to and from 1SQ in cylinder, track, sector, and number-of-sectors format rather than PB, first, and last PRU format. This moves the conversion of physical addresses to the CPU saving time and space in the PP.

The format of the 1SQ communication area is:



The following coded values are contained in the $status_1$ field of PPMES1 (bits 50 through 48).

Code	Significance
7	Controller/unit reject.
6	Unit busy.
2	On cylinder.

The following flags are contained in the $flags_1$ field (PPMES3, bits 23 through 12).

Bits Set	Significance
23	End of RB flag; indicates end of RB.
22	Recording mode flag.
21	Gap sector flag; if set, the pack was written with gap sectors.
20	DST flag.

Bits Set	Significance
19-18	Device type. 0 All devices other than 844 or 885 disk drives. 1 844-21 disk drive. 2 885 disk drive. 3 844-41 disk drive.
17	Release flag; if set, 1SP releases the interlock unit.
16	Interlock flag; if set, stack request is from interlock user.
15	Access flag; if set, CM access is set for the CM I/O request.
14	EOF flag; if set, the current cylinder is at EOI (for forward) or BOI (for backward).
13-12	Type of stack request; set by SPM. 0 Normal write. 1 Skip backward. 2 Direct I/O. 3 Others.

The following flags are contained in the $flags_2$ field (PPMES4, bits 47 through 42).

Bits Set	Significance
47	If set, 1SQ desires to drop. SPM does not assign a stack request if it recognizes this flag. Otherwise, SPM automatically assigns a stack request following an EX.SPRCL request from 1SQ.
46	If set, no PB was assigned at the last NXTPB request.
45	If set, interlock is broken.
44	If set, the unit is idle.
43	If set, a fatal error was detected.
42	If set, EOR/EOF was read.

RMS Device Capacity Limitations

The stack processor does not use all the space physically available on 844 and 885 devices. The innermost cylinders are unconditionally reserved for use by on-line disk diagnostics.

The ranges of cylinders used by the stack processor on each type of disk device are shown in table 5-14. These limitations must be observed by any installation that wishes to use CDC on-line disk maintenance software.

TABLE 5-14. RANGES OF CYLINDERS USED

Device	Cylinders Used		Reserved for CTI	Cylinders Physically Available
	Octal	Decimal	Decimal	Decimal
844-21	0-623	0-403	407	0-410
844-4x	0-1447	0-807	819	0-822
885	0-1506	0-838	840	0-842

Stack Processor Error Conditions

This section describes the error conditions that can be detected by the stack processor. In some cases, the action depends on debug mode. If IP.DEBUG is zero, these conditions are treated similarly to other errors. An error code is placed in the code and status field of the FET and FST entries and the control point is aborted if the error processing (EP) bit in the FET is zero. If IP.DEBUG is not zero, an invalid MTR function is issued with the stack processor output register having 77 in byte 0 and an error code in byte 1.

END OF INFORMATION - The error code 01 is inserted in bits 13 through 9 of the code/status field, but no message is issued and the control point is not aborted (nonfatal condition).

PARITY ERROR - A parity error is reported when any possibly recoverable device error occurs during a read or write operation. These include actual parity error, lost data, and mispositioning. The PRU is reread or rewritten up to 10 times (3 times for ECS). Whether success is attained or not, request execution continues after setting a flag. When request execution is completed, or the request is about to be reissued to the stack, the flag is examined. If the error was recovered, 1SX is called with code 03 (dayfile message RECOVERED PARITY ERROR), but this condition does not affect code/status or abort the control point. If all 10 attempts fail, 1SX is called with code 04 (dayfile message UNCORRECTABLE PARITY ERROR), 04 is put into bits 13 through 9 of code/status, and the control point is aborted if the EP bit is zero. A request at control point 0 is not aborted.

When an uncorrectable parity error occurs for a READ request and the EP bit is not zero, request execution terminates with the bad PRU being the last one read. When an uncorrectable parity error occurs for a WRITE request and the EP bit is not zero, request execution terminates with the FET pointer positioned after the last good write operation and the file positioned after the bad PRU.

BUFFER PARAMETER ERROR - This error occurs when a request is processed that references an FET when not all the following conditions are satisfied.

$0 \leq \text{FIRST} < \text{LIMIT} \leq \text{field length}$
$\text{FIRST} \leq \text{IN} < \text{LIMIT}$
$\text{FIRST} \leq \text{OUT} < \text{LIMIT}$

1SX is called with code 11_8 (dayfile message BUFFER ARGUMENT ERROR), error code 22_8 is put in bits 13 through 9 of the code/status field, and the control point aborts if the EP bit is zero.

NOT ASSEMBLED FOR ECS - This error occurs when a request references a DST entry for an ECS device. The stack processor issues a bad MTR request (code 77_8).

UNDEFINED ORDER CODE - A request contains order code 07. 1SX is called with code 22_8 (dayfile message INVALID STACK ENTRY), error code 22_8 is put in bits 13 through 9 of the code/status field, and the control point aborts if the EP bit is zero.

NO FET FOR O.RMR - A request contains order code 06 (O.RMR), but no FET is specified. 1SX is called with code 11_8 (dayfile message BUFFER ARGUMENT ERROR), error code 22_8 is put in bits 13 through 9 of the code/status field, and the control point aborts if the EP bit is zero.

ADDRESS OUT OF FL FOR O.RMR - The address for a table of disk addresses for O.RMR is out of field length. 1SX is called with code 22_8 (dayfile message INVALID STACK ENTRY), error code 22_8 is put in bits 13 through 9 of the code/status field, and the control point aborts if the EP bit is zero.

INTERLOCK BROKEN - A group of interlocked stack requests are interrupted by a malfunction of a controller and/or unit. The stack processor puts 24_8 into bits 13 through 9 of code/status field.

RMS HARDWARE ERROR - An RMS hardware error is reported when any of the following errors occur on a device.

- Unit not ready.
- Positioner not ready.
- 6681 internal/external reject.
- Unit busy too long.
- Channel stays active after connect or function.
- Unable to connect.
- No status returned.
- Address byte not accepted.
- No disconnect on status request.
- Abnormal on seek.
- Channel not active after ACN.
- Irrecoverable write error between 7154 coupler buffer and disk.

These error conditions are reported by the stack processor via a flashing message at the bottom of the B display. The function on which the error condition occurred is retried until it is recovered or until the device is idled down by the operator. In either case, the error diagnostic is cleared. If the device is idled, 22_8 is returned to bits 13 through 9 of code/status and 7ID is called. The operator is notified of the job name associated with the idled equipment via a flashing message at the bottom of the B display. 1SX is called when the operator acknowledges the message (the message is also sent to the dayfile), and the control point is aborted if the EP bit is zero.

Dismountable Pack Processing — I/O Detail

Figure 5-3 shows the flow of control of disk I/O, including the processing of dismountable devices.

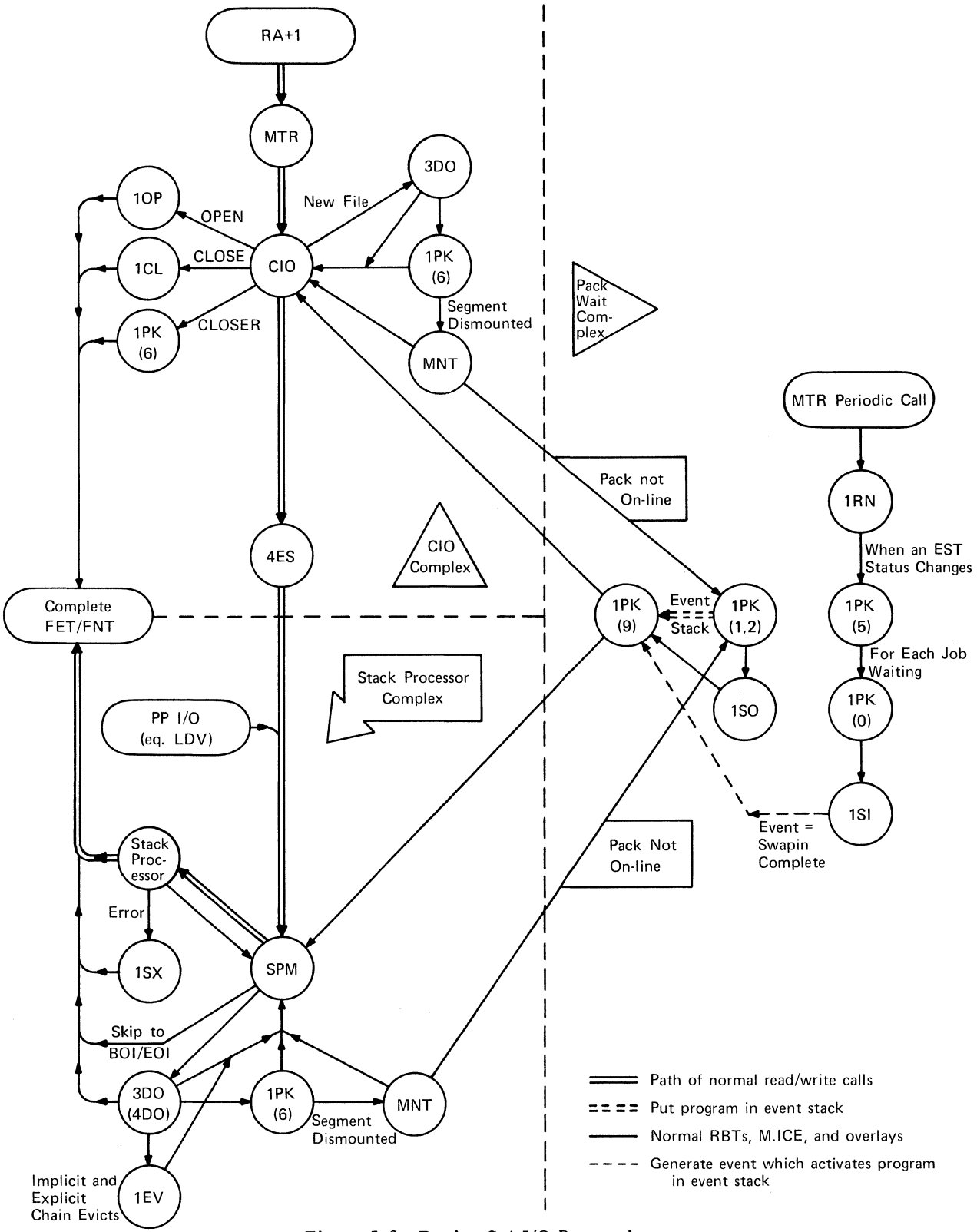


Figure 5-3. Device Set I/O Processing

Normal Calls for Read/Write

The user requests I/O by calling CIO in RA+1. If the file is new, CIO calls 3DO to assign it to a specific pack. If it is an existing assigned disk file, CIO loads 4ES which performs the following actions.

- Performs random positioning.
- Generates the stack request which accomplishes the function.
- Issues the function to SPM via the PP resident routine R.EREQS.

SPM processes all stack requests. Some system routines (JANUS, overlay loads, batch terminals) send requests directly to SPM rather than use CIO. SPM performs release-chain functions except for those file segments which are not on line.

SPM first determines if the function is a read or write function and whether the current segment of the file is on line. If the function is a normal write (not REWRITE), SPM assigns RBs to the file based on the amount of data in the buffer. If there is no space in the file and no free space in the current RBR, SPM sends the request to 3DO. When the write completes and too much space has been assigned, SPM is again called to remove any extra RBs.

SPM next determines which DST this disk belongs to and adds the request to that DST's chain. MTR ensures that a stack processor or a DST is active by checking the DST chain pointers. If there is no activity, MTR activates a stack processor.

The stack processor processes I/O to an arbitrary point, for example, to a cylinder boundary, and then may reissue the request to SPM. The stack processor always returns the request to SPM if the DAM ordinal in the RBT changes; thus, the ordinal 777 in overflow word pairs always causes a request to return to SPM and then to 3DO.

SPM-3DO Interface

SPM forwards to 3DO all requests which fail at any point in the preceding description: BKSPRU, release-chain not mounted, SKIPF or SKIPB with count=777777_8, file is new and has no RBTs (has not been assigned), position is at an overflow word pair (DAM=777_8), and so on. 3DO acts as the stack processor for the first DST, which is reserved; actual disk controllers begin at the second DST. Thus, SPM forwards 3DO requests via the same mechanism as normal requests by putting them on the first DST chain.

The only task of 3DO is to select a disk for files requiring space to write, such as new files. Other tasks are passed on. BKSPRU and SKIPF/SKIPB with count=777777_8 are sent to 4DO for completion. If the current segment of the file is not on a mounted pack, the function is not a write (as opposed to REWRITE), or if the file still has space available to write in, 1PK is called.

If the FNT indicates a new file (no RBTs), 3DO obtains a word pair and changes the format from a new file to an existing file by moving the flags from the FNT to the RBT. Next, the set for the file is chosen. If SN was specified, no selection is made. If SN is not specified, PF, Q, or SYS (SYS is available only by macro) is selected as specified. If none is specified, a flag is set to indicate that a scratch set is required.

Order of device selection is:

- The device must belong to the required set.
- If PF, Q, or SYS, the device must also have that attribute.
- If a device type, allocation style, or VSN is specified, attempt to match it; if no match, but DV was specified, repeat the attempt ignoring the requirements of VSN, device type, and allocation style.

Selection is limited to the allowed devices. Controller and unit activity from the DAT and available space in the RBR are factors in selection of the optimal device. 3DO creates an overflow word pair for all but new files, creates an empty word pair for the selected device, and reissues the request to SPM. 3DO considers only mounted devices. If 3DO finds no space available and this is not a public set request, 3DO calls 1PK to consider selection of an unmounted member of the set.

Once the file is written, its device set attribute is fixed.

1PK may be called by CIO, MNT, ADS, or independently via M.RPJ. 3DO cannot load 1PK via R.OVL because 3DO is considered a stack processor and, therefore, not allowed to use M.DFM or other I/O because it could lead to a deadlock condition. 1PK issues dayfile messages and can be accessed by 3DO only via a call to M.RPJ. The calling routine places one of the following function numbers in the PP input register byte 2 to specify the 1PK function desired.

Function 6 - Calls from 3DO are always made using function 6, and 1PK is called via M.RPJ. The call from CIO for CLOSER is also done in this way, so 1PK must determine if CIO is called and if the function is CLOSER. Space must be assigned if the function is not CLOSER, the function is a write (not REWRITE), and input register byte 4 is zero indicating a write at EOI. Assuming 3DO checked the mounted devices, the SMT is read and the file assigned to a pack with available space which is not mounted, considering first the on-line devices, then those not present. If there is no space and the user has not set UP, the user is aborted with error 10 (device capacity exceeded). 1PK selects a device and calls MNT as an overlay. When the mount is successful, MNT reloads 1PK which reissues the stack request. If the pack is not on line, 1PK function 9 is put on the event stack with the stack request in its message buffer (refer to functions 5 and 9).

If UP is on and there is no space, the FET is completed with error 10 (device capacity exceeded). If the current position is not on line on a public set, the job is aborted.

If not CLOSER and not a write, the first step is a mount. First, advance position past any overflow word pair; check if required disk is mounted. If it is, reinitiate the function by calling CIO, or SPM if CIO is not in input register. If not CIO, call MNT, which calls 1PK back and the process repeats if MNT found the pack on line and mounted it. If not, MNT calls 1PK function 2 to swap the job out while waiting for the pack. When the operator puts the pack on and turns on the EST, 1PK function 5 is called by 1RN, which periodically checks on/off changes. Function 5 causes the removal of all waiting 1PKs from the event stack and the stack request is reissued. MNT finds the pack on line and I/O proceeds.

CLOSER is processed by CIO calling 1PK function 6. When 1PK detects a CLOSER call, 1PK processing depends on whether the position is EOI. If EOI, a dummy overflow word pair is attached on the end of the RBT. The DAM and VSN fields are zero. Both EOI and EOV status are set in the FET.

If the current position is not EOI, the file is positioned to the next EOV word pair or to EOI, if it comes first. If the current position is an EOV word pair, nothing is done. EOV status is returned to the FET if the new position is an EOV word pair or EOI is encountered.

If MNT finds the operator dismount flag set in the SMT, it loads 1PK mode 7, which calls 1PK mode 8 with a delay, which reissues the stack request. This continues until the pack goes off line or is remounted.

Functions 1 and 2 - These functions are called by MNT, when a required pack is not on line, to delay the job until the pack becomes available. The SN/VSN are put into the variable area of the DDT if not there. The JDT enters the queue with others that may be waiting on the DDT. 1PK function 9 is put in the event stack on the JDT swap-in, and the job is swapped out by macro C1SO. The initial PP input register from the caller is sent to the message buffer. Stack requests are added to the message buffer.

Function 5 - Function 5 is called by 1RN when an EST free, busy, or off status changes (becomes different from the DDT). The fixed DDT for this EST is updated. If a pack has come on line and appears in the variable DDT area, 1PK function 0 is called for each job queued on the DDT and the job is cleared.

Functions 8 and 9 - Function 8 swaps the job in. Function 9 reinitiates the I/O function. If the input register was CIO, function 9 reissues it to CIO. If not CIO, the stack request is in the message buffer and is reissued to SPM.

Removing a Pack - DELSET and DSMOUNT

Both DELSET and DSMOUNT can be used to make a disk unavailable to the system. DELSET removes a disk from set membership and requires that there be no files on it. DSMOUNT makes a disk unavailable to a job. The operator dismount command, DMNT, permits the operator to remove a pack from a drive. DELSET uses PP routine DLM; DSMOUNT and DMNT use PP routine DSM.

DLM and DSM must halt all stack request activity before a disk can be removed from mounted status. To do this, they set the request idle bit in the EST. 1PK checks the stack request area for requests for this unit, and if it finds none, sets the EST status to FB=11 to indicate there is no activity. Any subsequent requests for the pack cause the requesting job to be swapped, and DLM or DSM can put the pack into unavailable status.

DELSET removes a disk from a set by zeroing its entry in the SMT (which defines set membership). Before this is done, DLM searches the RBT for local files and the PFC for permanent files resident on this disk and checks that the SMT usable count matches the total DAM available counts (if the disk is in dismounted status). If any test fails, DLM aborts the job.

A DSMOUNT call from a job (including the automatic one at the end of job) decrements the set's activity and resets it to dismounted status if no jobs reference it. The operator command DMNT resets the disk to dismounted status as soon as all I/O clears and sets the operator dismount flag in the SMT so that no other mainframe can mount the disk. Only an RMNT command or a RECOVER clears this flag.

If the pack is mounted on this mainframe, an operator dismount proceeds as follows:

1. The EST request idle bit is set, which locks out all I/O.

2. The routine waits until all activity stops on the disk (signaled by FB being set to 11); then it clears the request idle bit.

3. If the device is not shared, the RBR is translated into a DAM and written on the disk, the mounted flag is cleared from this mainframe's bit in the SMT entry, and the DDT is written in dismounted format (S.DDSN=1).

4. If the device is shared, the RBR is not written to the disk. The PP reads the DAM, clears all bits in the DAM which are clear in the RBRs, and then writes the DAM back. Other processing is the same as in step 3.

If the pack is not mounted but is on line on this mainframe, DSM sets the EST to FB=11.

If another mainframe has the pack mounted, DSM terminates with an operator message that the pack cannot yet be removed. It sets the EST to FB=10 (free). The pack cannot now be mounted unless the operator enters the RMNT (remount) command. The operator can attempt DMNT again later.

If no other mainframe has the member mounted, DSM sets the EST to off and FB=0 (no pack on line). The operator can then remove the pack.

ECS-Buffered I/O

Reading and writing of large sequential RMS files is greatly enhanced by the use of ECS buffers. Such operations involve the use of a small CM buffer in the user's field length and a large user's buffer in ECS. The data is transferred between ECS and the RMS device either through a system circular buffer (SCB) in CM or through a distributive data path (DDP). ECS buffering is not available on a CYBER 176. Refer to the section on 819 Disk I/O Processing for a description of 819 buffered input/output. The following describes a write sequence involving an ECS buffer; a read sequence is essentially the reverse.

The user requests ECS buffering on a file-by-file basis through the REQUEST control statement or macro. On the control statement, the user includes an EC parameter in addition to the normal parameters in one of the following forms.

Parameter	Use
EC	For a default (IP.BUF) size buffer.
ECxxxx ECxxxxK	For a buffer of xxxx-thousand (octal) words.
ECxxxxP	For a buffer of xxxx (octal) pages.

In the REQUEST macro, the user must set bit 33 to one in the second word of the parameter list. In the fourth word of the parameter list, the buffer size must be set in bits 11 through 0, and display code character K or P must be set in bits 17 through 12.

In a write sequence, the user first puts data into a CM buffer, which need be only about 200 words long, then issues a CIO call through RA+1. If an XJ instruction follows the request in RA+1, the job is exchange-jumped out of execution, and CP.MTR begins processing the request.

CP.MTR recognizes the CIO call and passes it to CP.CIO for processing. ECS-buffered file I/O causes CP.CIO to perform a validity check on the FET and activates the proper ECS driver. The data is then written directly from the user's CM buffer to the user's buffer in ECS.

The preceding process continues until the user's ECS buffer is full; then a stack request is generated by CP.CIO, requesting that the ECS buffer be written to an RMS device. In processing the request, the stack processor loads the appropriate ECS executive routines into an area of stack processor memory.

When the stack processor requests an SCB, CBM (system circular buffer manager) assigns the first one available from the list of SCBs. Each SCB has an integral number of PRUs, so that PRUs are not split across the end of the buffer. When a CM buffer path is selected, the stack processor uses the SCB in the normal circular I/O mode, with the following modification. In addition to FIRST, IN, OUT, and LIMIT, the FET-like SCB control table also contains a TRIGGER and a DIRECTION field. Before the transfer, the stack processor puts an M.SCB in its PPOR. During the transfer, MTR checks to see if the trigger has been reached. If so, it calls CBM to process the SCB; if not, no action is taken. This circular I/O continues until the ECS buffer has been emptied and all data has been written out to the RMS file. The processing is designed to prevent the stack processor from missing disk revolutions during an I/O buffer transfer.

When the DDP is selected, the processing is similar. CP routines do the same bookkeeping as in CM, but only point to the data in ECS instead of transferring it to CM. As a result, less CM is used for the transfer, n words instead of 65 * n words for an SCB containing n PRUs.

Figure 5-4 illustrates the general flow of output to an ECS-buffered RMS file. Either data path may be assigned dynamically, depending on availability.

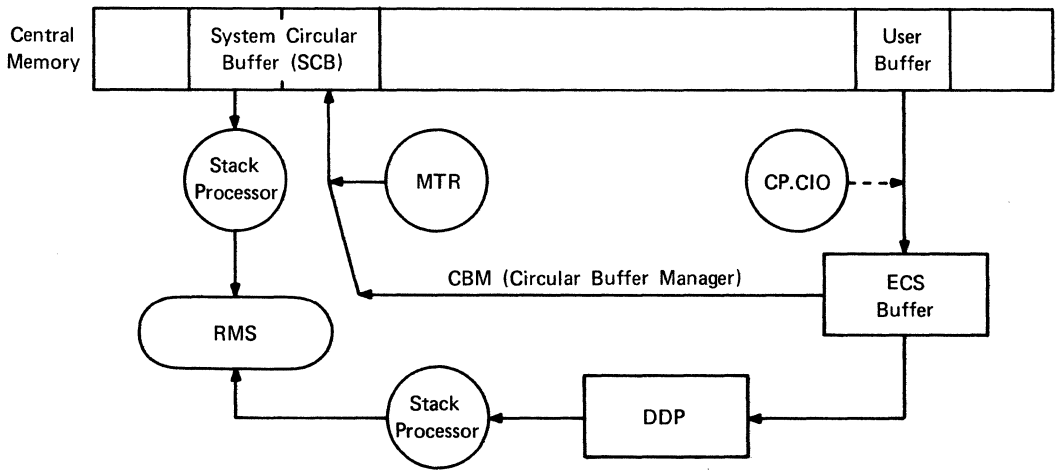


Figure 5-4. Output Flow to RMS File

819 DISK I/O PROCESSING

The CYBER 170 Model 176 computer uses a PPS consisting of 10 or 20 peripheral processors (PPs) and up to six first level peripheral processors (PPUs) to perform I/O tasks (figure 5-5). The PPS communicates with all standard NOS/BE equipment such as card readers, line printers, magnetic tape units, and so on. The PPU can communicate only with an 819 disk drive. Because a PP cannot access the 819 disk directly, a special method of I/O processing is necessary when a request involving the 819 disk is encountered. This method uses LCM as an intermediate buffer area. Data is transferred between the 819 disk and LCM through the PPU and a hardware buffer in CM and between LCM and the requesting control point or PP. The remainder of this section describes the logical and physical I/O processing involved in the data transfers, PPIO processing for the 819 disk, LCM buffer management and tables, and the CE error files.

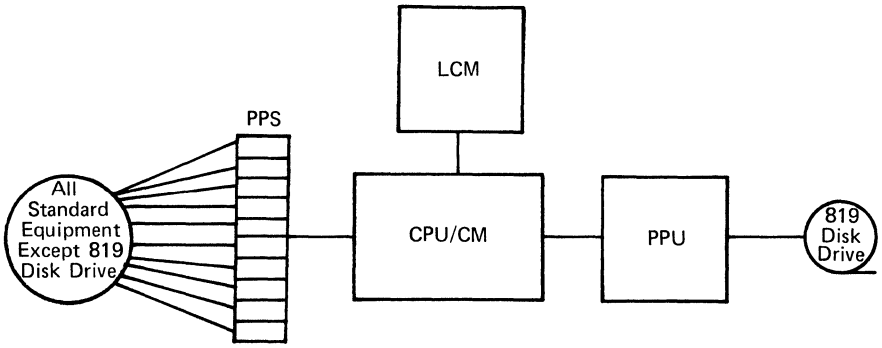


Figure 5-5. CYBER 176 Computer System with 819 Disk

LOGICAL I/O PROCESSING

General Description

The following discussion describes the flow of 819 disk logical I/O processing (figure 5-6). The segments labeled HDRV, IH, and HDC are part of a sequence of routines which handle the physical I/O. Physical I/O is discussed in detail in this section.

When a user requests that an 819 disk file be read, written, or repositioned, CPCIO/CP4ES makes a stack entry and assigns the request to SPM. SPM calls the segment BFM which allocates a buffer area and a transfer buffer table (TBT) in LCM. (Refer to LCM Buffer Management in this section.) The TBT contains the current stack request and information about the data in LCM for each 819 disk file. If the required number of buffers in LCM are not immediately available for allocation, the TBT is put on an empty TBT chain where it can be processed at a later time. CPMTR recalls BFM periodically to process the empty chain.

Segment BFM adds the TBT address to the TBT address table (TAT). BFM also initially creates TAT in the paged area of LCM.

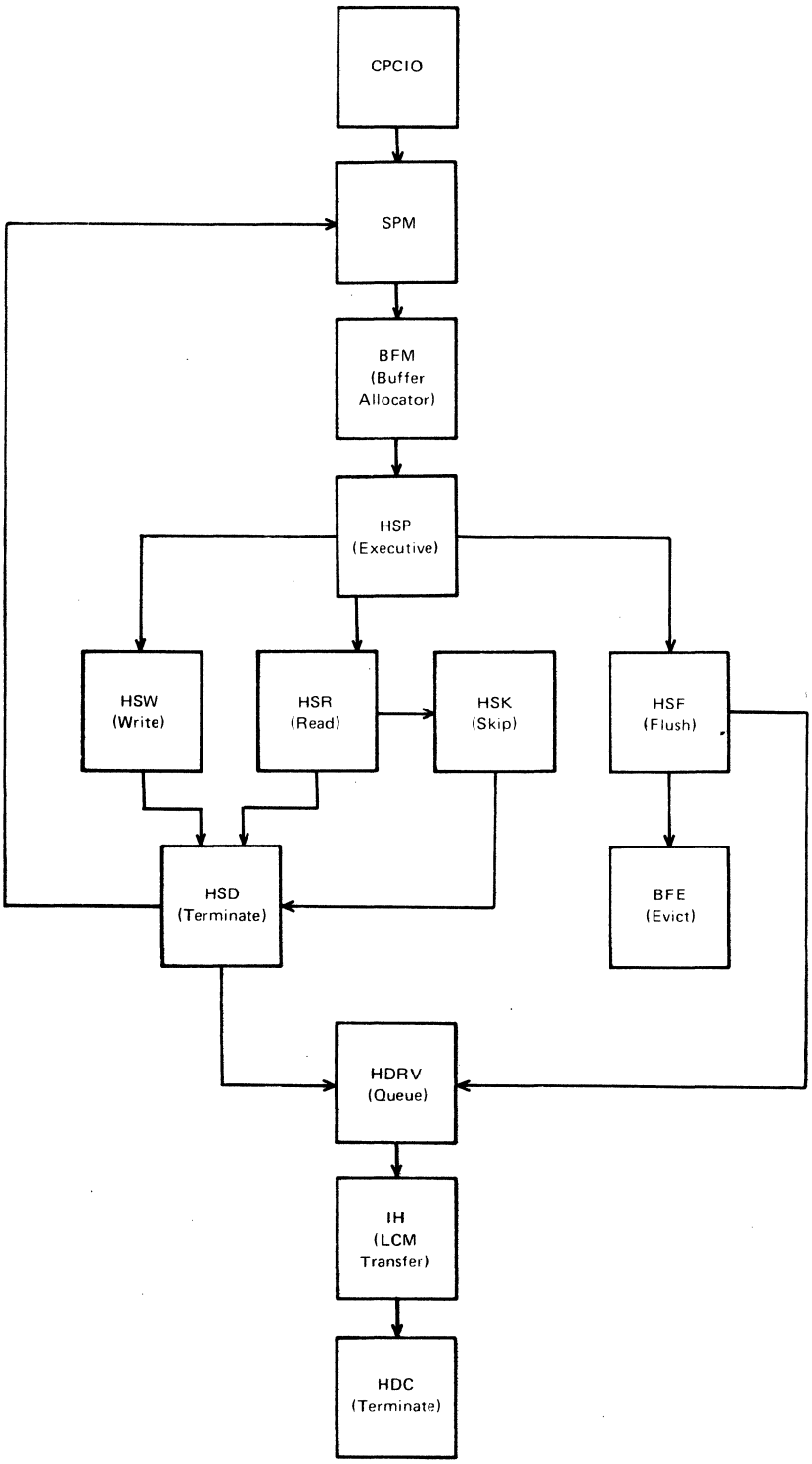


Figure 5-6. Logical I/O Processing

BFM then calls HSP, the 819 disk executive. Depending on the operation requested, HSP calls one of the following segments.

Segment	Description
HSW	Write operation.
HSR	Read operation.
HSK	Skip operation (called through HSR).
HSF	Flush operation.

If, for example, the user wants to write a file on an 819 disk, HSP calls the segment HSW. HSW gets the starting PRU from the stack processor and begins transferring data from CM to the buffer in LCM. HSW saves short PRU headers, builds the sector header PRU flags, and updates IN† and INW† in the TBT. When the end of the 819 sector is reached, one of the following occurs.

- If the limit PRU is equal to the current PRU and the next PB† is zero, SPM is called to allocate another PB. If another PB is not available, the request is terminated for reissue.
- The next PB is moved to current PB and data transfer continues until the user buffer is empty and/or the LCM buffer is full.

When the LCM buffer threshold is reached (buffer is half full), HSD calls HDRV with a request to initiate transfer of data to the 819 disk. HDRV places the request in the unit queue table (UQT). The UQT holds all 819 disk requests received from HDRV for each unit. When the proper channel is available, HDRV assigns the channel, formats the request for the PPU driver, and calls the interrupt handler. The interrupt handler (IH) moves data between LCM and a hardware buffer in CM and sends a request to transfer data to the PPU. The PPU disk driver (HCD) writes the data from the hardware buffer to the 819 disk. HDC is discussed further under Physical I/O Processing in this section.

If the write request was not completed when the LCM buffer was filled, HSD sets the wait-on-disk flag before calling HDRV. When the interrupt handler reads or writes a sector from a TBT which has wait-on-disk status set, it calls HSP which recalls HSW to continue processing the request. HSW transfers data until the request is satisfied or the CM buffer is empty. This sequence of calls allows the LCM buffers to be circular buffered. When the request is satisfied or the CM buffer is empty, HSW sets the stack request completion flag and returns control to HSD. HSD calls SPM which drops the stack request and sets the FST and FET completion bits.

If the user wants to read a file on an 819 disk, HSP calls the segment HSR and essentially the write operation is reversed. The request to read the disk is given to HDRV, which places the request in the UQT. The PPU driver, HCD, reads data from the 819 disk to the hardware buffer and IH transfers the data to LCM. HSR then moves the data from LCM to CM.

HSR gets the starting PRU number from the stack request and checks the beginning PRU and PB values in the TBT to determine if data has been read from the disk. It stores the level number to be transferred to the FET by SPM and transfers the buffer, updates OUT† and OUTW† in the TBT, and processes sector error flags. When the end of a PB is reached (the next PB in the TBT is zero), HSR gets the next PB from SPM.

---

†Refer to the transfer buffer table in appendix E for an explanation of fields.

HSR transfers data to the user's field length in CM until the next PRU will not fit in the area or the request is satisfied. When the request is satisfied, HSR sets the stack request completion flag and returns control to HSD. HSD calls SPM which drops the stack request and sets the FST and FET completion bits.

If the file was found to have a request currently outstanding before the read or write operation, HSR checks the outstanding operation. If it is a request to read, and the PRU being read is not the requested PRU, HSR calls HSF, which calls HDRV to terminate the unit queue entry and set the flag to stop the transfer.

The HSK segment is called by HSR to skip logical records forward or backward. HSR gets the correct PRU from the stack request and searches for the end-of-record. If the desired PRU position is found in the LCM buffers, HSK performs the repositioning forward or backward depending on the stack request code. If the end-of-record is not found in the buffers, HSK returns control to HSR which repeats the process until EOI or BOI is found.

HSP calls the HSF segment when it receives a request which indicates a mode change (read to write operation or write to read operation). HSF is called directly from CPMTR for a monitor flush command or an evict request. A flush command is made when a file must be returned (such as before swap-out or roll-out) between job steps, and at the end of the job. When the job is swapped or rolled back in, the first reference to the file reinitializes the TBT.

For empty buffers and inactive read buffers (last operation performed was a read), HSF calls BFE to clear the TBT and release the LCM buffers. For buffers currently active and write buffers (last operation performed was a write), HSF queues the TBT to the physical I/O segments (HDRV, HDC, and so on), which later recalls HSF. If I/O is complete, HSF flushes the inactive write buffers. The TBT is cleared.

Logical I/O Segments

The following segments are involved in 819 disk logical I/O processing.

Segment	Function
BFM	Allocates a TBT and LCM buffer areas for each file.
HSP	819 executive; calls routines to perform allocation, reading and writing data between LCM and user's field length in CM, repositioning within files, and flushing of data. HSP performs the following sequence of actions. 1. Checks whether storage access is allowed. 2. Checks FET for valid parameters. 3. Compares new request to last request. If there is a mode change (read to write or write to read) or evict request, calls HSF. 4. If read request, calls HSR. 5. If write request, calls HSW.
HSF	Flushes and evicts files. Performs the following actions. 1. Drops inactive read buffers and associated TBT. 2. Flushes write buffers. 3. Drops inactive write buffers and associated TBT.

Segment	Function
HSR	Initiates read operation, or if skip request, calls HSK. If read request, performs following actions. 1. Compares buffer position to requested position. 2. If positions are the same, calls CHR to transfer data from LCM to CM. 3. If positions are incorrect, HSR sets up a disk request to read data from disk.
HSW	Initiates write operation. Performs following actions. 1. Compares buffer position to requested position. 2. If positions are correct, calls CHW to transfer data from CM to LCM. 3. If positions are incorrect, calls HSF to flush the buffer.
HSK	Skips logical records forward or backward. HSR fills several LCM buffers and calls HSK when the requested position is in the buffers. Based on the stack request code, HSK calls SKF to skip forward or SKB to skip backward. For a skip backward, the LCM buffer area is subtracted from the file position so that the maximum amount of data behind the current empty buffer position is read.
HSD	Terminates stack request if the request is completed. If the requested operation has caused a buffer threshold to be reached, HSD puts the request on the unit queue (in the UQT) to be processed.

PHYSICAL I/O PROCESSING

General Description

The preceding section on logical I/O described the sequence of segment calls leading to HDRV and IH. This section describes, in detail, the segments involved in actual physical I/O processing for the 819 disk (figure 5-7).

HDRV is called to queue I/O requests for the 819 disk. It calls HDIN, only once, to initialize the UQT and channel table (CHT) after a deadstart. HDRV enters the request in the unit queue (UQT) and, if it is the only request in the queue, calls HDSL. HDSL selects the best request for a unit to perform I/O, selects the best channel, and calls the interrupt handler (IH). IH puts the request in the hardware I/O buffer and contacts the PPU through the disk driver HCD. When the PPU acknowledges the request, IH calls HDRQ. HDRQ checks for a request on another unit and, if it finds one, calls HDSL. HDSL sends the request to the PPU as described previously.

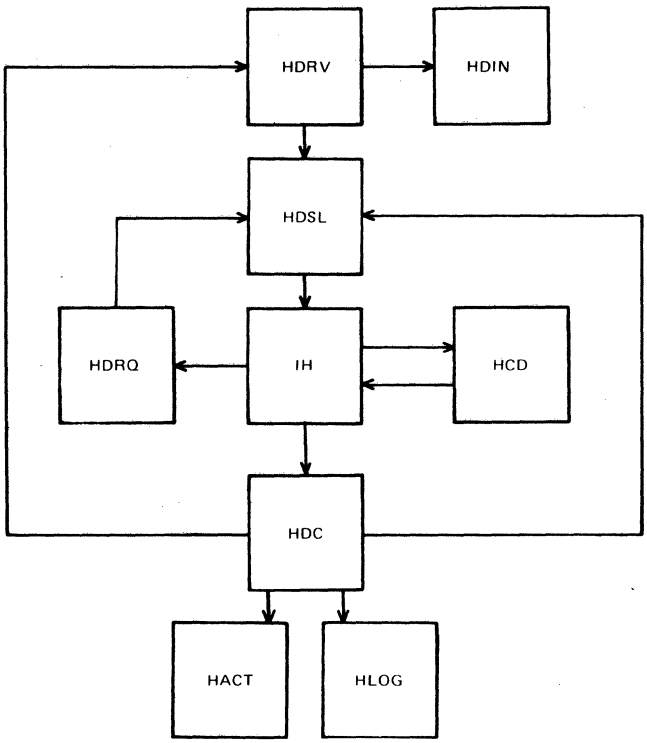


Figure 5-7. Physical I/O Processing

When the PPU finds one of the units on the proper cylinder and sector, it communicates this to the IH, and data is transferred to or from the disk. At the end of the data transfer, IH calls HDC to terminate the request. HDC may call HDRV to requeue the request, HACT to record activity status, or HLOG to record disk errors. (Refer to CE Error File in this section.) If there are any other requests on any unit, HDC returns to HDSL.

Physical I/O Segments

The following segments are involved in 819 disk physical I/O processing.

Segment	Function
HDRV	Queues 819 disk requests. Enters the TBT in the UQT according to ascending order of cylinder addresses. Within the cylinder, TBTs are positioned in the order in which they were received. If there are no other TBTs in the queue, I/O is initiated for that request.
HDIN	Initializes the UQT and CHT based on the EST and DST. HDIN is called only once after a deadstart.
HDSL	Selects best TBT for I/O transfer and best channel. HDSL bases its selection on the cylinder address of the forward TBT, backward TBT, and the current position of the unit. It assigns the unit to the best channel, sets up the request for HDC, and calls the interrupt handler.

Segment	Function
IH	Transfers data between LCM and the PPU I/O buffers. The following sequence of events occurs. 1. Master output IH sends disk request to master PPU. 2. Master PPU receives request and initiates master input IH. 3. Slave PPU calls slave input IH to start I/O link-up. 4. Master and slave PPUs call output IH to transfer data from LCM to the hardware buffer or input IH to transfer data from the hardware buffer to LCM. 5. Slave PPU calls input IH to terminate all input and output disk requests. The interrupt handler can perform the following actions. • Handle 513-word transfer size, picking up first word from the TBT and remainder from LCM buffer. • Transfer to LCM buffer based on TBT pointers; update the buffer flags, IN and OUT fields. • Recognize a switch to a new PB and send the next head/sector to the PPU in the middle of a request. • Decide one sector in advance whether to do continuous transfer based on the following conditions. If the next TBT in the UQT is on the same cylinder and is the same type of request, IH automatically switches to that request. HDC is notified to terminate the previous request. If next PB on the current file is on the same cylinder. IH continues on the current TBT. IH sets last PB equal to current PB and current PB equal to next PB. • Calls HSP after a sector transfer if the wait-on-disk flag is set.
HDRQ	Searches for requests. HDRQ is called by IH when the PPU acknowledges the request. HDRQ searches for an unassigned unit with a TBT entry in the queue. If one is found, HDRQ calls HDSL to initiate I/O on the unit.
HDC	Completes request. When a request is completed, HDC performs the following actions. 1. Processes termination status from the PPU. 2. Sets the channel to idle mode. 3. Releases PPUs by calling IH. 4. Updates the current PB, cylinder, headgroup, and sector. 5. Removes TBT from UQT.

Segment	Function
	6. Sets unassigned status for the unit. 7. Checks conditions for calling HSP. 8. If threshold is reached or flush process was not complete, requeues the TBT in UQT. 9. Calls HDSL to select the next TBT for I/O processing.
HACT	Logs activity status for CE file. HACT records the number of sectors transferred on a unit and sets up a CE message in the CEFAP buffer, which contains the number of sectors transferred since the last recorded message. It calls CEM through CALCEM to complete logging of the CE message.
HLOG	Logs 819 disk errors. HLOG retrieves recovered and unrecovered errors recorded in the CM hardware I/O buffer. It reformats the errors to the CE message format and puts them in the CEFAP buffer. CEM is called through CALCEM to complete logging of the CE messages.
HDAY	Calls CEM to send message to job and system dayfile and conditionally abort the job.

PPIO PROCESSING

PPIO stack requests control data transfers directly to and from PP memory. Figure 5-8 shows the logical I/O segments (as shown in figure 5-6) plus the segments used in PPIO processing. The calling PP program builds a stack request and passes its location to the PP resident routines R.READP or R.WRITEP. These routines call the subroutine R.RWP to perform the data transfer. R.RWP returns control to the calling PP program when the stack request is complete.

Although the type of mass storage involved in the transfer is transparent to the calling PP program, a special interface with the 819 stack processor HSP has been defined for PPIO. This interface consists of reserving a CM system circular buffer (SCB) for the PPIO request and then emulating CM FET I/O.

The following segments are used in PPIO processing.

Segment	Function
RWH (PP resident)	Interfaces with HSP and performs data transfers to and from the SCB.
BFP	Executes during 819 stack request initialization. BFP reserves the SCB, linking it to a TBT, and sets the W.RWPPCW in the calling PP's communication area to load and execute the segment RWH.
HPO	Initiates data transfer operation when called by RWH via an M.ICE/EX.PPIO function. HPO determines the type of request and calls HSP to transfer data between the SCB and the PP. If a read operation is being terminated or reissued, HPO exits to HPP.

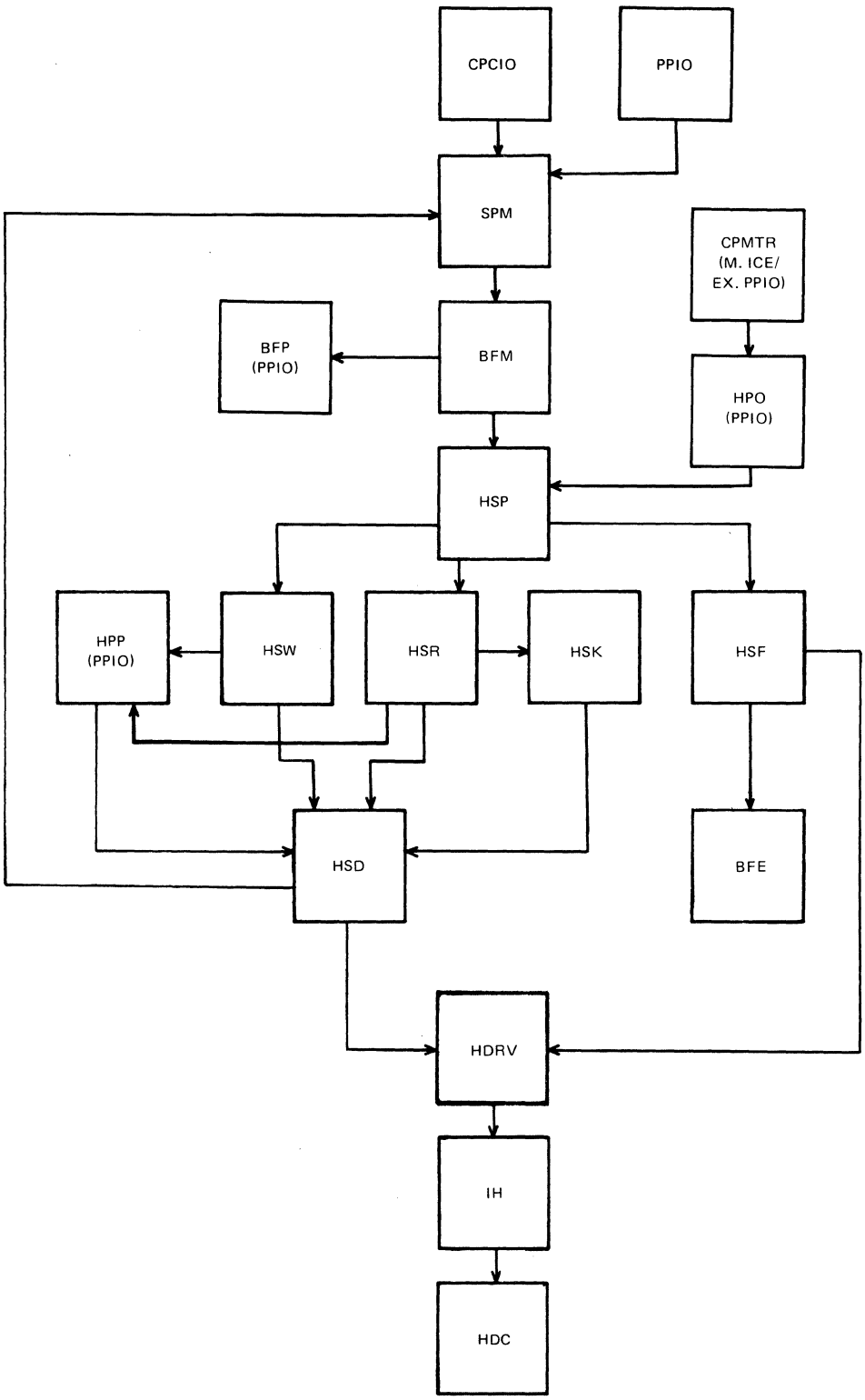


Figure 5-8. PPIO Processing

Segment	Function
HPP	Called by the 819 read and write routines HSR and HSW after a data transfer. If there is data to be transferred between the SCB and the PP, HPP sets W.RWPPCW to execute RWH in the calling PP and then returns to CPMTR. If the request must be reissued, HPP sets up W.RWPPCW to cause PP resident to loop in R.RWP waiting for communication to be reestablished. If the request is to be terminated, HPP sets complete status in W.RWPPCW. R.RWP exits to R.READP or R.WRITEP which returns to the caller. HPP releases the SCB and calls HSD to terminate or reissue the request.

LCM BUFFER MANAGEMENT

The LCM partition, part of the dynamic paged area, consists of 512-word buffers, each of which can hold an 819 disk sector. From 2 to 24 buffers can be allocated for each file. The contents of the TBT fields FIRST, IN, OUT, and LIMIT are pointers to these buffers.

When a request for a new file is made, BFM must initialize a TBT. If an empty TBT cannot be found, another system page of 512 words is allocated. If there are not enough LCM buffers to allocate area for the new file, BFM must call BFE to evict an inactive file. Selection of a file is based on FNT order. The 819 file evict code keeps track of the position in the FNT where the last file was evicted. The search for the next file to be evicted takes place ahead of this entry in an end-around search of the FNT. When the file is selected, BFE clears the TBT and releases the LCM buffers. If the buffer is empty or a read buffer, the new file is assigned to the released area immediately. If the buffer is a write buffer, the new file request must wait on the recall chain until I/O on the write buffer is complete and the buffer has been evicted.

TABLES

Transfer Buffer Table

The transfer buffer table (TBT) contains information necessary to move data through the LCM buffers. One TBT is allocated for each file and remains until the file is evicted. It contains the addresses of the LCM buffers and the latest stack request for the file. TBTs are allocated until a 1000_8-word page is full. A CM location contains a pointer to the first TBT page. Subsequent TBTs are pointed to by the first word of the page. That word contains a zero if it is the end of the chain. The format of the TBT is given in appendix E.

TBT Address Table

The TBT address table (TAT) contains the addresses of TBTs for all active files. Segment BFM creates the TAT which resides in the paged area of LCM. The TAT contains a one-word-long entry corresponding to each entry in the FNT. When initially assigning a TBT to a file, segment BFM adds the address of the TBT to the appropriate entry in the TAT. Segment BFE clears this address when the TBT is released.

During processing of the monitor flush function, the system saves in the TAT the address of the output register of the PP, issuing the function or the address of the third word of the FNT entry for the file being flushed. The format of the TAT is described in appendix E.

Unit Queue Table

The unit queue table (UQT) is used by HDRV to queue disk requests for each unit. The UQT contains a header and a four-word entry for each unit. It points to the first and current TBT in the unit queue and is also linked to a particular channel table entry (primary/secondary channels). The format of the UQT is given in appendix B.

Channel Table

The channel table (CHT) describes the activity of each data path. There is one entry for each data path. The CHT is used primarily by the interrupt handler. It points to the UQT for the current unit. The format of the CHT is given in appendix B.

CE ERROR FILE

After each disk request, the PPU sends status to the CPU. If the status shows any error, the segment HLOG logs the status information, along with other information it has gathered, in the CE error file. Generally, one error file entry describes an attempt to recovery one error, but if other errors occurred during the recovery procedure, the additional errors are also recorded in the entry. At least 1 bit in the first or second group of flags in the error file is set to describe the type of error. All statuses in the entry reflect the status of the first failure for the error. Figure 5-9 shows the format of the CE error file.

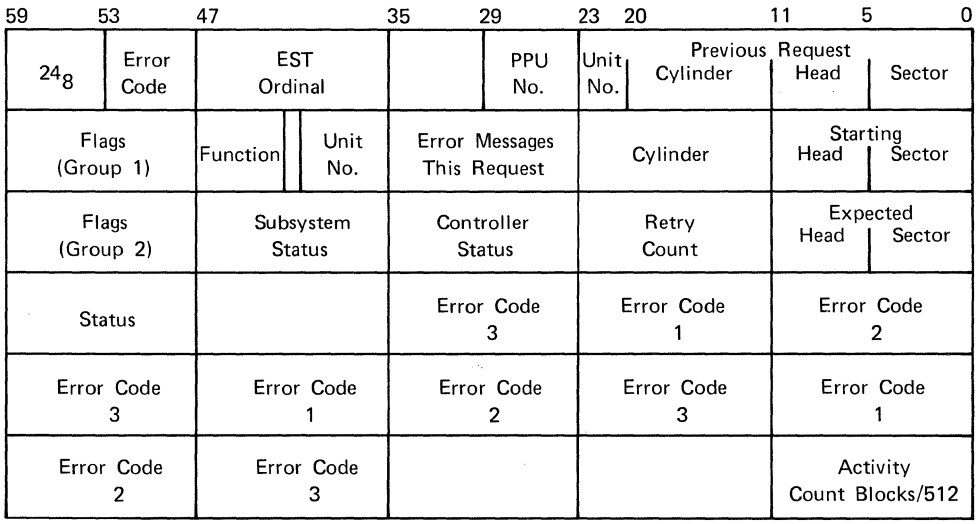


Figure 5-9. CE Error File

Word	Bits	Description
0	59-54	24g; type of error.
	53-48	Error code (in octal).

Code	Description
06	Status message.
17	RMS address.
20	RMS checkword error.
71	RMS abnormal error.

	47-36	EST ordinal.
	35-30	Unused.
	29-24	Number of PPUs which encountered the error.
	23-0	Previous request on this unit.
		23-21 Unit number.
		20-12 Cylinder address.
		11-6 Head group.
		5-0 Starting sector.
1	59-48	Flags (group 1).

Bit Set	Significance
59	Request aborted; error unrecovered.
58	Last sector transferred with unrecovered checkword error.
57	Request completed successfully.
56	No resumption on control channel.
55	No resumption to RF sent to disk.
54	Slave aborted the request.
53	Unit never on cylinder.
52	Unit down.
51	Channel down.
50	CPU request error.
49	Hardware error.
48	Partner PP down.

Word	Bits	Description
	47-42	Function code.

Code	Description
0	Read.
1	Write.

	41	Verify sector header address on write request.
	40-36	Physical unit number.
	35-24	Number of error messages for this request.
	23-12	Cylinder address.
	11-6	Starting head group.
	5-0	Starting sector.
2	59-48	Flags (group 2).

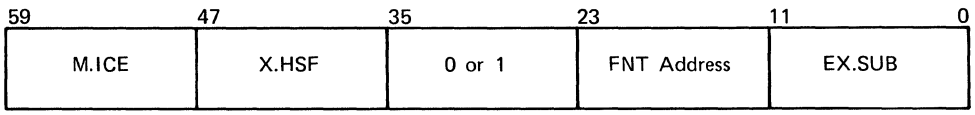
Bit Set	Significance
59	Slave encountered error.
58	Error during read or write of data; refer to controller status for type of error.
57	Not-on-cylinder status occurred without previous position function.
56	Unused.
55	Subsystem busy.
54	Error correction attempted (includes error code statuses).
53	Unit not ready (maximum includes unit fault and interlock statuses).
52	Unit, track, head, or sector in header not in desired position.
51	Cylinder address in the cylinder status is not the desired cylinder.
50	Head address in the head status is not the desired head.
49	Either the unit number in the controller status is not the desired unit or a controller error occurred during a seek.
48	Seek error.

Word	Bits	Description
	47-36	Subsystem status.
	35-24	Controller status if controller error detected.
	23-12	Retry count.
	11-6	Expected head group.
	5-0	Sector address expected.
3	59-48	Contents vary depending on error. • Cylinder status if status is incorrect. • Head status if status is incorrect. • Cylinder address recorded on sector if address is incorrect. • Unit fault status if not ready. • Error code 1 if error correction was attempted.†
	47-39	Contents vary depending on error. • Head and sector recorded on sector if incorrect. • Interlock status if not ready. • Error code 2 if error correction.
	35-24	Error code 3 if error correction.
	23-0	Error codes 1, 2, and 3 for up to three more channels of error correction (multiple checkword errors).†
4	59-0	
5	59-36	
	35-12	Not used.
	11-0	Activity count; number of blocks/1000 since last activity count was reported.

† The controller status tells which channels had checkword errors. The order of error correction for checkword errors on multiple channels is 3, 2, 1, 0. If one channel cannot be corrected, the message does not include any further error code statuses.

819 SUBSYSTEM FLUSH FUNCTION

A PP routine can use the M.ICE function with the subfunction EX.SUB to flush the LCM buffers associated with a file residing on an 819 device. A PP invokes this function by setting up the PP output register as follows:



The FNT address field contains the FNT address of the file to be flushed. The PP sets byte 2 of the output register equal to 0 or 1 with the following implications.

Setting	Significance
0	The system does not clear the PP output register until the flush is complete. Thus, it is not necessary for the PP to check for a completion bit.
1	The system clears the PP output register after segment HSF receives the flush function. Thus, the PP must check bit 0 (completion bit) of the third word of the FNT to determine flush completion. Only PP routine 1S0 issues the flush function with byte 2 equal to 1.

PERMANENT FILES 6

PERMANENT FILES-SYSTEM INTERFACE

Two pointer words in the CMR pointer area contain pertinent system information about files. Word 6 contains the number of attached permanent file table entries, the starting address of the attached permanent file table, and a byte of flags. Word 7 contains the MST ordinal of the system set and PF default set; the starting address and number of entries in the dismountable device table, and the starting address and number of entries in the mounted set table.

PERMANENT FILE INTERLOCKS

Several types of interlocks are used to prevent conflicts during the accessing of permanent files on RMS devices:

- Hardware interlocks.
- Set interlocks in the label.
- Set interlocks in the mounted set table (MST).
- Interlocks in the set member table (SMT).
- Permanent file manager (PFM) interlocks in the MST.
- Utility interlocks in the MST.
- Attached permanent file (APF) interlocks.

Hardware interlocks are used by RECOVER, TRANSPF, SPM, and PFM to test and set interlocks residing on a device and to prevent any changes to critical tables (SMT, DAM, PFC, and PFD) on the master device, if the set is already mounted on any mainframe. They are set by issuing an interlocked stack request and released by issuing a stack request without interlock. They are retained when interlocked stack requests continue from the same PP or to the same FNT as with the first stack request.

The set interlock in the label is used to prevent device set routines (such as ADS,MNT) on other mainframes from simultaneously updating critical tables (SMT, DAM). When this interlock is set, no other mainframe is allowed to run a MOUNT on the master device until the interlock is cleared. Thus, a program can ensure that no other mainframe accesses a particular set by first checking that no other mainframe has yet mounted the master device and then setting the set interlock in the label. To set this interlock, first the label should be read with hardware interlock, then the set interlock should be tested and set, and finally the label should be rewritten without hardware interlock. To release this interlock, first the label should be read, then the interlock should be tested and cleared if still set, and finally the label should be rewritten.

Set interlocks in the MST are used to prevent two or more device set routines from running on the same mainframe. (This function can also be performed by the set interlock in the label.) The purpose of keeping the central memory set interlock is to allow jobs to swap out while waiting for the set interlock in central memory to clear; thus, the action, if the interlock in central memory is not set, is different from the action required if the interlock on the device is not set. The set interlocks in the MST can be set by reserving the MST channel, setting the interlock, and then releasing the MST channel. The interlocks can be cleared by reading the MST entry, testing, and clearing the interlock if not already cleared.

Interlocks in the SMT are used to interlock a member device, preventing other mainframes from mounting the device so that it can eventually be physically dismounted. They are also used by RELABEL to ensure exclusive access to the device's SMT, DAM, and so on. There are five such interlock bits in the SMT: four correspond to the mounted/unmounted flag on the four possible mainframes; the fifth is a request dismount flag. This last flag, when set, prevents any other mainframe from mounting the device; thus, it can be used to interlock a member which is not yet mounted. These bits can be set by reading the SMT with hardware interlock, setting the request dismount flag, and then rewriting the SMT releasing the hardware interlock. They can be released by reading the SMT with hardware interlock, testing, and clearing the interlock if not already cleared, and then rewriting the SMT.

PFM interlocks in the MST are used to prevent two or more PFM routines from simultaneously accessing the PFD or PFC within a single mainframe. This can also be achieved with interlocked stack requests; however, the central memory interlock allows a PP to enter the event stack to wait for the PFM interlock in the MST to clear. Furthermore, it also prohibits TRANSPF's table transfer mode from running in the same mainframe at the same time.

In a multimainframe environment, permanent file disk tables PFD and PFC are accessed with interlocked stack requests, so as to provide interlocking between mainframes; however, within a single mainframe, the same PFM interlock is used to interlock both the PFD and also the PFC. The PFM interlock is set based on the device set (that is, in the MST) and is always cleared along with the device interlock whenever the PP takes an abnormal exit. When a broken connect occurs during the process of reading or writing the PFD or PFC, new sequences are executed to perform extensive checking and clean-ups for recovery attempts.

For a complete permanent file cycle, the type of access permission currently granted to each mainframe is recorded in the PFC interlock word in the PFC entry. To access an individual permanent file, it must be ensured that there are no access permission conflicts with the other mainframes. For an incomplete permanent file cycle, only exclusive access permission is granted to one mainframe by recording its mainframe ordinal in the PFD entry.

The PFM interlock in the MST is set by reserving the MST channel, setting the interlock, and then releasing the MST channel. It is released by reserving the MST channel, reading the MST entry, testing and clearing the interlock if still set or allowing error exit for the interlock if not still set, and releasing the MST channel.

The utility interlock in the MST is used by TRANSPF in conjunction with the set interlock in the master device label, after testing that no other mainframe has the set mounted and that the permanent file manager activity is quiet on the set. This ensures that the host mainframe has exclusive access to the set while transferring disk tables and also prevents CATALOG and ATTACH functions on the same mainframe. The utility interlock is set by reserving the MST channel, setting the interlock bit, and then releasing the MST channel. The interlock is cleared by reserving the MST channel, clearing the interlock if set or allowing error exit for the interlock if not set, and releasing the MST channel.

The APF interlock is used for interlocking an APF entry, preventing two or more routines from simultaneously accessing or updating the APF entry. The APF interlock is set by reserving the APF channel, setting the APF flag in the APF entry, and then releasing the APF channel; it is cleared by clearing the flag in the APF entry.

PERMANENT FILE TABLES

There are four system files created as permanent files during system deadstart: the system dayfile, the CE error file, the edit-extension file, and the system (library) file. Each has an entry in the permanent file directory (PFD) and the permanent file catalog (PFC). The PFC was called the RBT catalog (RBTC) and these names can be used interchangeably.

PFD and PFC are of fixed length, and reside on the same mass storage device. A user device set has its own PFD and PFC on the master device of the set. Pointers to these tables are kept in the device label.

The APF table is CM resident and consists of two-word entries, up to a predefined size for the installation (L.APF, default of 30 decimal).

The PFD has a header at the beginning of subdirectory 1, followed by a number of subdirectories of equal length. The PFD header is a copy of the PFD RBT chain for use in recovery. Each subdirectory consists of PFD entries.

The PFD is a file whose first allocation unit is recorded in the device label of a master device. This file has two fixed length parts. The first part, called the preamble, is always 512 words long. It contains a complete list of record blocks assigned to the PFD. The second part, called the PFD body, contains the individual entries. The size of the PFD body must always be an integral multiple of pages. Also, this multiple must be a power of 2. The size of the PFD body must be less than 511 pages. The PFD body is initialized to binary zeros.

The size of a PFD entry is always 16 words, 4 entries per PRU.

The PFD body has n ID hash points. These hash points are equally spaced within the body and the space between any two ID hash points is an integral multiple of pages. This multiple is a power of 2. The number (1 to n) of the hash point is the ID hash.

Since space between two ID hash points is a multiple of pages, the start of each of these pages is a PFN hash point.

A subdirectory is a subset of PFD entries which have a common ID hash.

Up to five cycles or versions of each permanent file can be maintained. The PFD entry, therefore, can have up to five pointers to the PFC, each corresponding to an entry for a cycle. Each cycle entry contains the cycle number, the mainframe ordinal (for incomplete cycles only), four flags, and the PFC pointer. The mainframe ordinal indicates the mainframe currently accessing the cycle, so that in a multimainframe environment, exclusive access can be granted. The first flag is not used; the second signifies that the entry is incomplete, since no PFC pointer exists; the third, that the cycle is archived; and the fourth, that the cycle has a parity error.

When cataloging, the user can specify the number for the cycle or permit a cycle number to be generated automatically. Unless the user specifies the cycle when a file is attached, the one with the highest cycle number is attached. Cycle numbers range from 1 to 999.

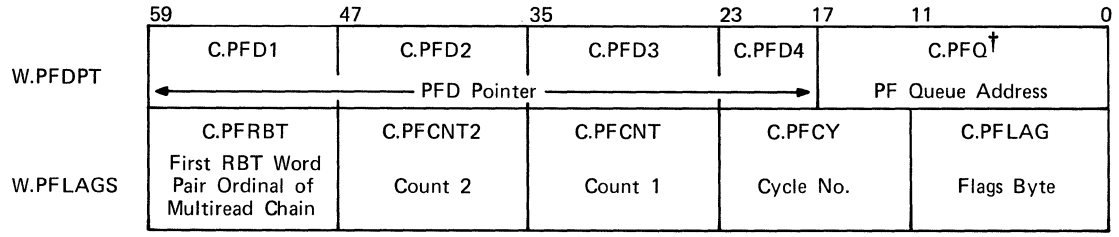
The permanent file name (pfn) can consist of up to 40 characters. The PFD entry also contains the passwords given when the file was cataloged. PFD entries are placed into appropriate subdirectories by a hashing algorithm applied to the file ID.

The entry-in-use flag in word 0 of the PFD is set when an entry is built. Detailed formats of the PFC and PFD entry appear in appendix D.

Detailed formats of other tables relating to device sets are included in appendixes B and D of this manual. The set member table (SMT) is a directory of the members of a device set. The device allocation map (DAM) functions like an RBR for a device. The DAM is copied to central memory when the device is mounted, maintained in central memory while the device is being used, and copied back to disk when the device is dismounted. The subdirectory table is used to keep track of the number of permanent files in each subdirectory.

The logical flaw table is a bit map of space reserved for system tables on the device and space unusable because of hardware flaws on the device. The physical flaw table is a list of hardware flaws for a device. PFC entries are always allocated as one or more PRUs. One PFC entry may occupy several PRUs which must be contiguous.

Figure 6-1 is the format of the APF entry. Whenever a permanent file is attached to a control point, an entry is made for it in the APF table. A pointer in the FNT entry of an attached permanent file relates that entry to the corresponding APF entry.



C.PFLAG

Bit 11	Unused	Bit 05	Interlock (S.PFIL)
Bit 10	Full dump (S.PFFD)	Bit 04	RB conflicts (S.PFRBC)
Bit 09	System file (S.PFSYS)	Bit 03	Exclusive access (S.PFEA)
Bit 08	RBT chain obsolete (S.PFOBS)	Bit 02	Single modify (S.PFSM)
Bit 07	Archived file (S.PFARC)	Bit 01	Single write (S.PFSW)
Bit 06	Multimainframe permission conflicts (S.PFMMF)	Bit 00	Priority lockout (S.PFPL) Reserved entry (S.PFRES)

† If nonzero, PF queue address equals JDT address of a job waiting for access to the permanent file.

Figure 6-1. APF Table Entry

An incomplete cycle is a file that has a PFD entry but no PFC entry. It can result from either a CATALOG or LOADPF job that does not terminate normally. An incomplete cycle can be attached and purged if the CY parameter is used to specify the cycle and control permission is established.

The PFC, like the PFD, is a file whose first allocation unit is recorded in the device label of a master device. This file has two fixed length parts. The first part, called the preamble, is always 512 words long. It contains a complete list of allocation units assigned to the PFC. The second part, called the PFC body, must always be an integral multiple of pages. This multiple need not be a power of 2. The PFC body is initialized to binary zeros.

PFC entries are always allocated in blocks of 64 words. One PFC entry can occupy several blocks which must be contiguous.

PERMANENT FILE ACCOUNTING

When W.CPFACT in the control point area is nonzero, accounting messages will be sent to both the system and user dayfiles whenever the status of a permanent file changes in the job. These messages are given in the NOS/BE Reference Manual. The ACCOUNT statement will set W.CPFACT nonzero.

PERMANENT FILE UTILITY ROUTINES

Permanent file utility routines provide capabilities of dumping permanent files to tape, loading dumped files from tape, transferring permanent files and tables from one mass storage device to another, and producing printed reports on the status of each permanent file. The permanent file utilities are:

Utility	Description
DUMPF	Copies selected permanent files from mass storage to tape, either to clear all permanent files from mass storage devices or to provide periodic backup files for an installation. DUMPF must be on a system library. No files should be attached to a job containing DUMPF operations during DUMPF execution.
LOADPF	Reloads permanent files from the tape created by DUMPF to mass storage. LOADPF allows automatic retrieval of an archived file. The user is unaware of this activity if loading is from tape. When an attempt is made to attach an archived permanent file, the operator can type n.GO, which causes the file to be loaded from the appropriate dump tape. An archived file can be purged without loading the file.
PFLOG	Writes an image of the permanent file catalog (PFC) to tape.
GENLDPF	Reads the PFLOG tape, creates a permanent file directory (PFD) entry for each archived file entry, and calls the routine LPF to recatalog the file. Remaining unarchived file entries are sorted by dump tape VSN in ascending order. GENLDPF creates a LOADPF job for each VSN and routes it to the input queue.
TRANSPF	Transfers permanent files and/or related tables within a set or from one set to another. TRANSPF cannot move system files.
AUDIT	Produces a formatted output file containing statistics on permanent files on a set.

The following permanent file utility function codes (octal) are for system interface between permanent file utilities and PFM routines.

Utility	Function Code
AUDIT	166
DUMPF	172
LOADPF	174
TRANSPF	176

The control statements which call the permanent file utilities are described in the NOS/BE Reference Manual. The remainder of this discussion describes the dump tape formats for DUMPF and PFLOG.

DUMPF UTILITY

DUMPF writes the permanent file dump to a multivolume magnetic tape file in the format shown in figure 6-2.

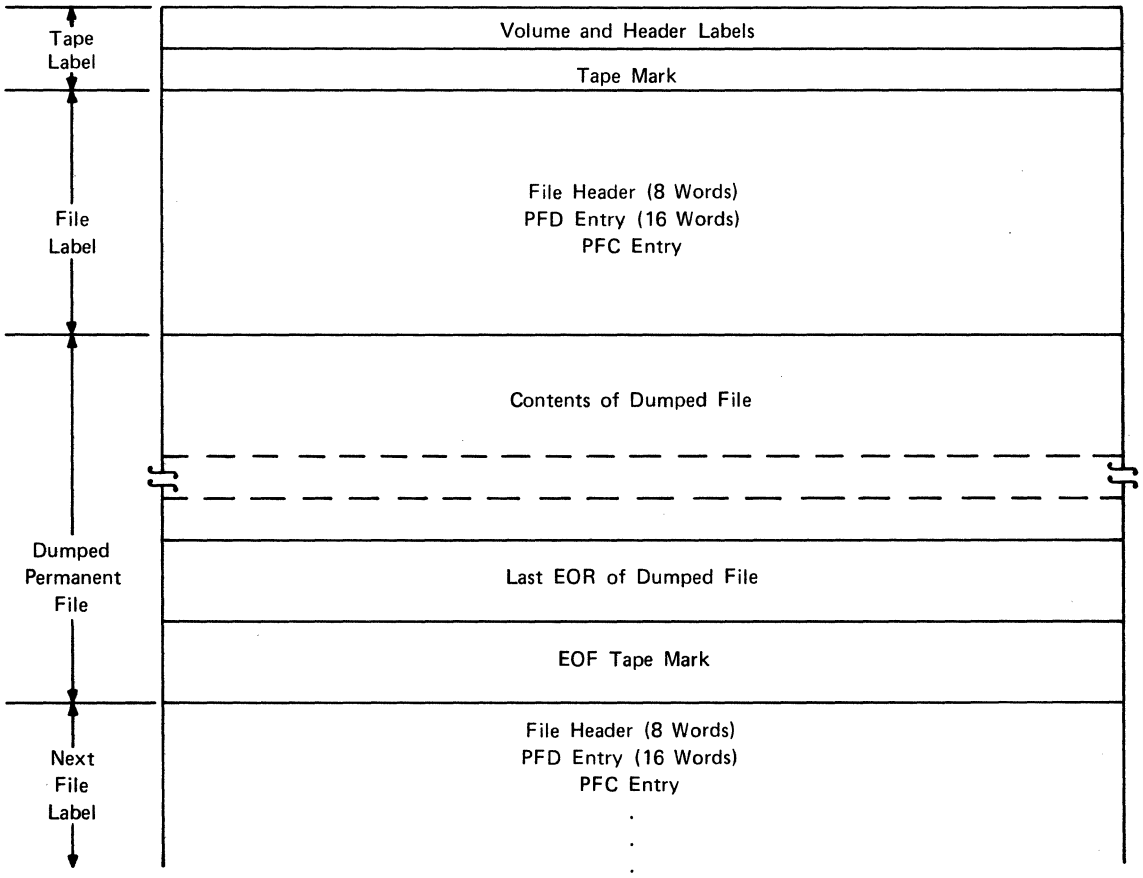


Figure 6-2. Permanent File Dump Tape Format

The first 20 characters of the tape label are as follows:

HDR1DUMPF TAPE- NEW

VOL1DUMPF TAPE-UNIT

Figure 6-3 shows the file header. The fields in the header are in table 6-1.

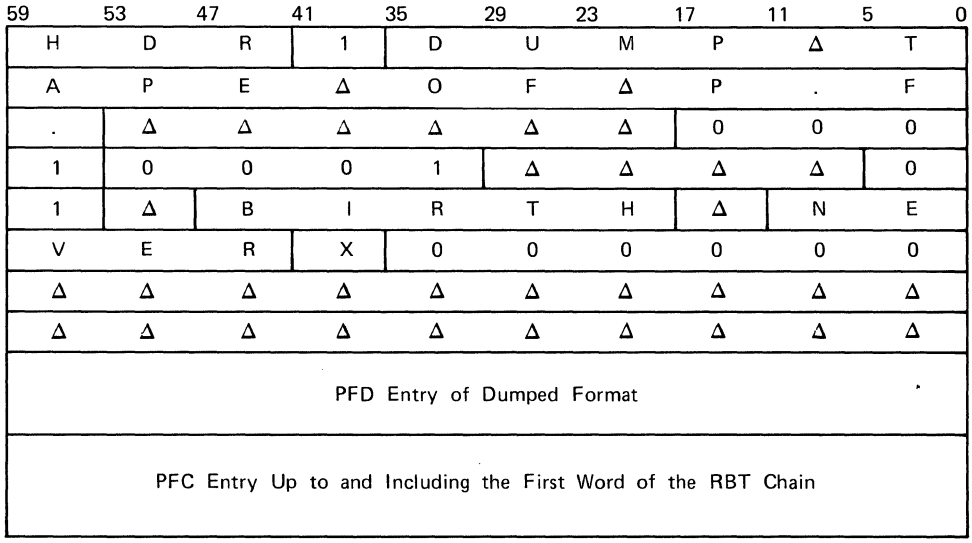


Figure 6-3. File Header

TABLE 6-1. HEADER FIELDS

First Character Position Within Header	Field Length (Characters)	Field Contents	Field Description
1	3	HDR	Label identifier.
4	1	1	Label number.
5	17	DUMP TAPE OF P.F.	File label name.
22	6	Blank	Multifile identification.
28	4	0001	Reel number.
32	4	0001	Multifile position number.
36	4	Blank	Reserved for tape compatibility.
40	2	01	Edition number.
42	1	Blank	Reserved for tape compatibility.
43	5	BIRTH	Creation date.
48	1	Blank	Reserved for tape compatibility.
49	5	NEVER	Expiration date.
54	1	X	Security.
55	6	000000	Block count.
61	20	Blank	Reserved for tape compatibility.

PFLOG UTILITY

PFLOG calls the PP routine OUX with function code 10_8 to create an FNT entry for the PFC. This makes the PFC a local file at the control point where the PFLOG job is running. PFLOG reads the PFC and writes it to a multivolume magnetic tape file in the format shown in figure 6-4.

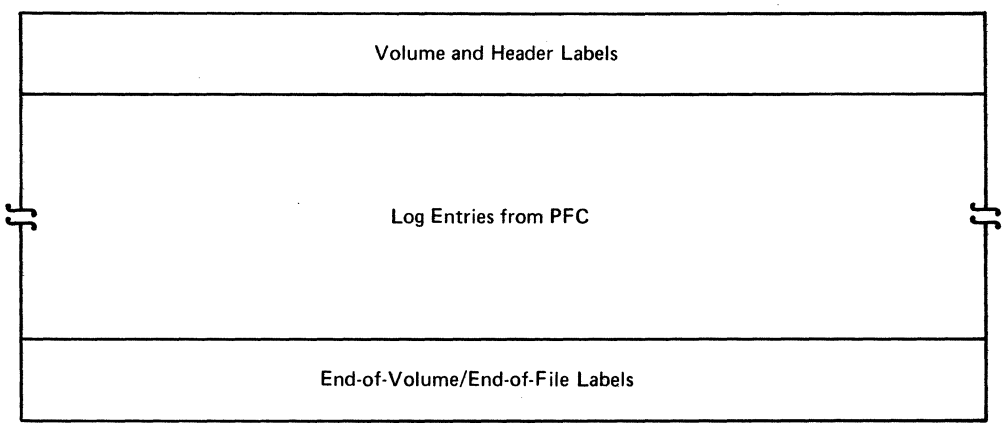
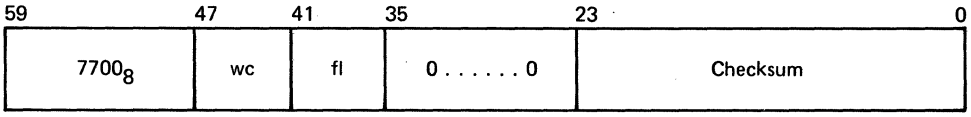


Figure 6-4. PFLOG Dump Tape Format

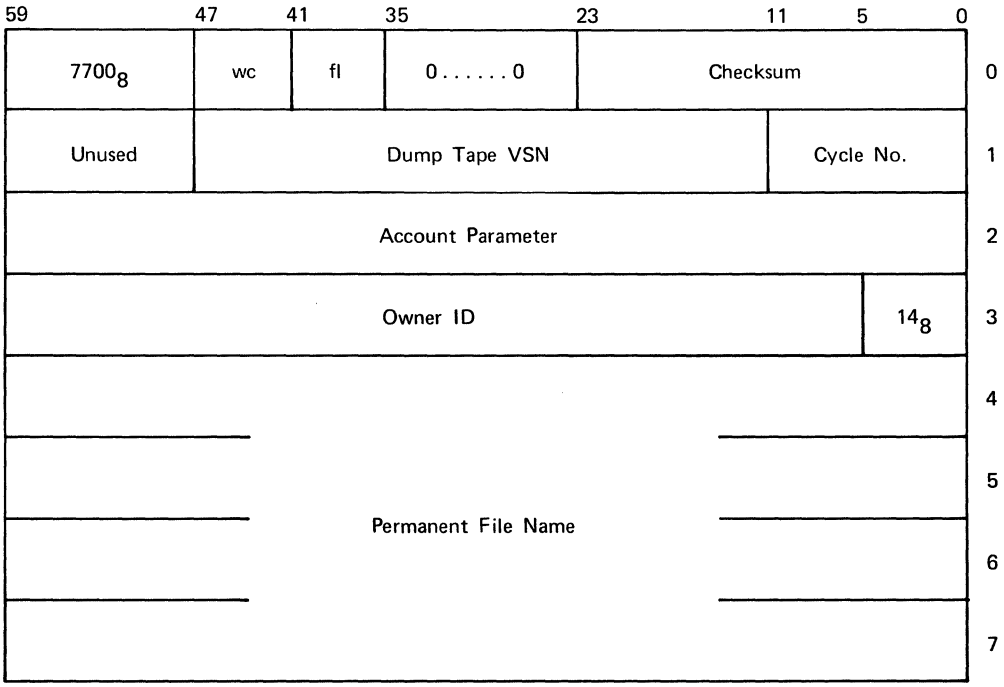
The first 20 characters of the tape label are as follows:

HDR1 LOG-FILE-TAPE

The log entry for an archived file is identical to the PFC entry for permanent files (refer to PFC entry in appendix D) with the exception of the first word (word 0) and the RBT chain. Instead of the entire RBT chain, the log entry contains only the first RBT word pair; this word pair is the last word of the log entry. The format of word 0 of the log entry for an archived file is as follows:



The log entry for a file which is not archived contains only eight words as follows:



The wc and fl fields are the same for each entry.

Field	Significance
wc	Word count of log entry.
fl	Flags.

Bit	Significance
41	If set, indicates the log entry is for an archived file.
40	If set, indicates nine-track tape.
39	If set, indicates 6250-cpi density tape.

PRIVATE DEVICE SET PROCESSING

Device set creation and usage are described in the NOS/BE Reference Manual. The remainder of this section describes additional capabilities of the LABELMS, ADDSET, MOUNT, RELABEL, and RECOVER statements and the FSN program to address public sets.

LABELMS CONTROL STATEMENT

In addition to the parameters described for the LABELMS control statement in the NOS/BE Reference Manual, system-generated jobs at postdeadstart time can also specify the NPR, DS, and EST=est parameters.

Parameter	Description
NPR	Allows the system to use the entire range of cylinders normally usable on the device. If CTI is on the device, the system does not specify the NPR parameter when it generates the LABELMS statement. If NPR is not specified, and if DDS or MSL is on the disk, LABELMS preallocates space to prevent DDS or MSL being overwritten by the system. Refer to RMS Device Capacity Limitations in section 5 for the usable cylinders on each disk device. The NPR parameter has meaning only when the DS parameter is specified.
DS	Deadstart parameter; used only by the operating system to call LABELMS during postdeadstart.
EST=est	EST ordinal of device; used only when the DS parameter is specified.

If DT=AH (819 disk drive) is specified, the system ignores the NPR, DS, mode, and EST=est parameters.

The deadstart routine GENDJ uses the LABELMS utility during an initial deadstart to blank label RMS devices and to determine flaws from factory format information.

ADDSET CONTROL STATEMENT

In addition to the parameters described for the ADDSET control statement in the NOS/BE Reference Manual, system-generated jobs at postdeadstart time can also specify the *Q, DS, and EST=est parameters.

Parameter	Description
*Q	Queue files may reside on the member added to this device set. This parameter is allowed only if the set is the public queue set.
DS	Deadstart parameter.
EST=est	EST ordinal specified at deadstart.

MOUNT CONTROL STATEMENT

In addition to the parameters described for the MOUNT control statement in the NOS/BE Reference Manual, system jobs can also specify the DS and EST=est parameters.

Parameter	Description
DS	Deadstart flag; used by the operating system at deadstart to mount public devices.
EST=est	EST ordinal used at deadstart.

For the 844-21 disk drive (AY), 844-41 disk drive (AZ), and 885 disk drive (AJ), MNT issues a stack request with the factory data flag set to read the pack serial number and manufacture date. It then issues a CE error file message containing the serial number and date. If a parity error occurs or the factory data does not exist on the pack, the error file entry contains a pack number of *NO S/N* and a date of 000000. The CE error file entry is described in detail in the On-Line Maintenance Software Reference Manual.

MNT processes the serial number and date only during the initial mount of the pack, not when the pack is mounted by subsequent jobs at other control points.

RELABEL CONTROL STATEMENT

RELABEL is used only by deadstart to rewrite the label on RMS devices to add or delete flaws, or to re-create a label that has been destroyed. The format of the RELABEL control statement is

RELABEL(VSN=vsn,SN=setname,MP=mp,I=0,O=0,DS,mode,mpmode)

Parameter	Description
VSN=vsn	Volume serial number of the device to be relabeled.
SN=setname	Name of the device set containing the device.
MP=mp	Volume serial number of the master device of the device set.
I=0	No input file is used.
O=0	No output file is used.
DS	RELABEL is to be run during deadstart.
mode	Recording mode (full-track or half-track) of the disk pack to be relabeled.

mode	Recording Mode
FT	Full-track; read/write contiguous sequential sectors.
HT	Half-track; read/write alternate sectors.

Refer to the description of mpmode for restrictions on using full-track mode.

Parameter	Description
mpmode	Recording mode of the master pack of the device set.

mpmode	Recording Mode
MF	Full-track; same effect as FT.
MH	Half-track; same effect as HT.

If the master pack of a device set is to be relabeled, it is not necessary to specify mpmode.

NOTE

Full-track recording mode (FT or MF) can be specified only if the system is using 2XPP speed, and if the device to be relabeled is one of the following:

- 844-21 disk pack accessible by a 7154 controller.
- 844-41 disk pack accessible by a 7154 or 7155 controller.
- 885 disk pack accessible by a 7155 controller.

RELABEL run under deadstart obtains a complete new set of flaws from IRCP through CMR, rewrites the PFT and the device label, and rewrites the DAM and LFT for the device on the master device. It detects any conflicts between the new flaws and already existing files and system tables, and advises the operator when such conflicts exist.

RECOVER CONTROL STATEMENT

The RECOVER control statement is used to validate a device set and reconstruct critical tables. In a dual-mainframe system, it can also be used to recover space on a device set when one of the mainframes becomes inoperative.

RECOVER can have up to six continuation cards or lines. The last nonblank character of the card or line to be continued must be a separator; parameter fields (keyword=nnn) cannot be split between cards or lines.

The format of the RECOVER control statement is

RECOVER(SN=setname,V=vsn,L=filename,MF=mfid,T=term,MO=mode)

Parameter	Description
SN=setname	Name of the device set to be processed.
V=vsn	Volume serial number of the master device of the device set.

Parameter	Description
L=filename	Name of the file on which any errors encountered in the validation process are listed; default is file OUTPUT. When MO=1 or 2, this parameter need not be specified and is ignored.
MF=mfid	If MO=1, ID of this mainframe. If MO=4, primary logical ID of the mainframe that has become inoperative. If MO=0 or 5, this parameter need not be specified and is ignored.
T=term	Sequence number assigned to RECOVER by GENDJ for identification when RECOVER runs under the control of deadstart. When MO=0 or 4, this parameter need not be specified and is ignored.
MO=mode	Specifies mode of RECOVER processing.

mode	Description
0	Default mode. It is called by control statement or, if the device set is shared, by operator command. Mode 0 runs only on an unmounted device set. It performs the following actions. • Performs a complete interlock cleanup. • Rebuilds the DAM using the LFT and the PFC. • Cross checks the PFD and the PFC. • Detects the RB conflicts. • Resets the available RB counts in the SMT. • Flags the PFC entries for which RB conflicts have been detected. • Gives a comprehensive listing of all errors.
1	Mode for first mainframe to deadstart. It is called at deadstart by GENDJ which provides the term parameter. Mode 1 runs only on an unmounted public device set. If the device set is being shared, the mainframe initiating the RECOVER must be the first mainframe in the multimainframe system to be brought up. It performs the following actions. • Performs a complete interlock cleanup. • Rebuilds the DAM using the LFT and the PFC. • Detects the RB conflicts. • Resets the available RB counts in the SMT. • Flashes a message on the B display when an error is detected; either gives a GO option to the operator (nonfatal errors) or aborts (fatal errors). • Calls HDS at the end of processing; transmits the term parameter value and the number of RB conflicts.

Parameter	Description

mode	Description
2	Mode for mainframe other than the first to deadstart. It is called at deadstart by GENDJ, which provides the mfid and term parameters. Mode 2 runs on a shared device set. It performs the following actions. • Performs interlock cleanup for the local mainframe. • Rebuilds the DAM using the LFT and the PFC if the device set has not been mounted by another mainframe; otherwise, validates the DAM against the LFT and PFC. • Detects the RB conflicts. • Flashes a message on the B display when an error is detected; either gives a GO option to the operator (nonfatal errors) or aborts (fatal errors). • Calls HDS at the end of processing; transmits the term parameter value and the number of RB conflicts.
4	Resource recovery mode. It is called by control statement or, if the device set is shared, by operator command. Mode 4 runs on either mounted or unmounted device sets. It performs the following actions. • Performs a cleanup of the interlocks left by the inoperative mainframe. • Rebuilds (if two mainframes) or validates (if more than two mainframes) the DAM using the LFT, the PFC, and the RBRs and RBTs of the host mainframe. • Detects the RB conflicts. • Gives a comprehensive listing of all errors.
5	Called by GENDJ to get a full listing of the errors detected during the preceding call to RECOVER in mode 1 or 2. Mode 5 runs on either mounted or unmounted device sets. It performs the following actions. • Validates the DAM against the LFT and PFC. • Cross checks the PFD and the PFC. • Detects the RB conflicts. • Flags the PFC entries for which RB conflicts have been detected. • Gives a comprehensive listing of all errors.

ADDRESSING PUBLIC SETS BY SET ATTRIBUTE

The PP program FSN places the set name in the SN parameter slot and sets the FDB complete bit. The SYSTEM macro can be used to call FSN from an executing program using the following format.

SYSTEM FSN,R,fdbaddr,code

FSN	Program name.
R	Recall (mandatory).
fdbaddr	Address of FDB into which the correct set name should be placed. A dummy set name should be used when creating the FDB.
code	Device set to find.

0	System set.
100_8	Permanent file default set.
200_8	Queue set.
300_8	Reserved.

FSN aborts if no set exists unless the code 300 was specified. In that case, a scratch set is assigned if no buffer set can be found.

JOB PROCESSING 7

JOB FLOW

Jobs in the system are processed in three sequential, independent stages: input, scheduling, and output. Many jobs may be in any one of the three stages.

Jobs can be loaded into the computer by reading card decks into the system using the system package JANUS. Alternately, they may be input from tape with the tape loader or from a user terminal through INTERCOM.

A job comes under scheduler control after all system resources specified on the job statement have been assigned to it. When a job is under scheduler control, it has an assigned job descriptor table (JDT) entry and is either in one of the scheduling queues or assigned to a control point. When it is assigned to a control point, the control point status values indicate the job's activity. With the job swapping/rolling features of the operating system, several jobs can be assigned alternately to each control point. A job is in execution only when it is assigned a control point and central memory and is actively using a central processor. Then the job is executed as directed by the control statement record of the job's input file.

When the job has terminated, JANUS is assigned to process the OUTPUT, PUNCH, and PUNCHB files produced by the job.

JOB INPUT QUEUE

As each job is read into the computer by JANUS, the tape loader, or INTERCOM, it is placed into the job input file and a FNT entry for the job is created in the job input queue.

JANUS and INTERCOM call PP overlay 2VJ to translate the job statement. 2VJ scans the statement, checks the parameters for validity, computes a priority value for the job, determines the job class, and passes all this information back to the PP program which called it. TLOAD, the tape loader, and other CP programs call DSP to enter a job into the input queue.

The PP program 1IB scans the job input queue, looking for jobs eligible for execution scheduling. To be eligible, a job must have all resources requested on the job statement, except for central memory. A job remains in the input queue if it is dependent on a job which has not run, if it is waiting for operator tape staging, if the job is not to be brought to a control point, or if the amount of ECS it requested is not available.

Tape Job Scheduling

All incoming jobs requiring a specific number of seven-/nine-track tapes (as defined by parameters MT, NT, HD, PE, and GE on the job statement) are entered in the preinput queue, a subset of the input queue. The operator communicates with the preinput queue through console entries and the prescheduling (P) display. Each time the operator requests a P display, the highest priority tape jobs in the queue are displayed. A job requiring tapes is not placed in the normal job input queue until the operator releases it with a command. Once released, the job is considered for assignment to a control point and execution. It no longer appears in the P display. Tape drive scheduling is described in section 4.

Loading Jobs from Tape

The CP routine TLOAD is used to enter jobs into the input queue via magnetic tape. TLOAD is called to a control point by DSD when the operator enters the following command.

n.X TLOAD,x,y.

n Number of the control point to be used.

x Optional parameter specifying the number of the file group to be loaded (decimal digit; numbering begins with 1, not 0).

y Optional parameter specifying the number of the job in file x to be loaded (decimal digit; numbering begins with 1, not 0).

A file group is a collection of jobs on tape, terminated by a double end-of-file.

A standard circular buffer technique is used to process the job files. The first statement of the first record in each file is presumed to be a job statement, and it is processed accordingly. This assumption is made for each job on the tape. CIO and its overlays are called to read the tape into memory and write memory to disk.

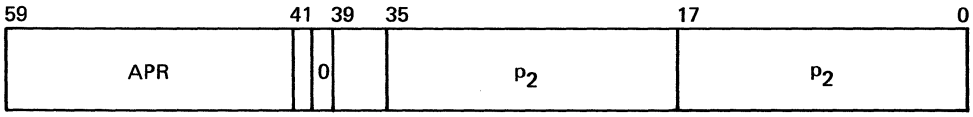
The number of control statements that can appear in a given job is not limited. If the user fails to place an end-of-record (7/8/9) statement behind the control statement record of a job, the omission is not detected by TLOAD, JANUS, or INTERCOM until the job runs.

Sequencer Jobs

Jobs running under the control of the sequencer remain in the input queue with a locked FNT. When the clock interval for the job has expired, the FNT is unlocked and the job becomes eligible for execution under the same constraints as other jobs in the input queue.

When the job completes execution, it returns to the input queue as a locked FNT until its clock interval expires again.

The sequencer is called to a control point using the following RA+1 request format.



Bit 40 Auto recall; must be zero.

P1	P2	Effect	Called by
0	0	Place sequencer program (APR) in delay request stack.	System job
0	1	Turn sequencer on.	Operator
0	2	Turn sequencer off.	Operator
1	xxxxnn	Add job to sequencer table at ordinal nn, with interval xxxx minutes.	Control statement

p_1	p_2	Effect	Called by
2	nn	Run job at ordinal nn.	Operator
3	nn	Drop job at ordinal nn from sequencer table.	Operator
4	xxxxnn	Set interval xxxx minutes for job at ordinal nn in sequencer table.	Operator
5	addr	Write real-time clock value in RA+addr.	System job
10	addr	Write RA, RA ECS, and FL ECS in RA+addr.	System job
11		Evict OUTPUT file for the job.	Control statement

Range of p_2 parameters (in octal):

$0 < nn < L.SEQ/2$

$0 < xxxx < 3777_8$

$addr < RA+FL$

For p_1 parameters (functions) 1 and 11, APR can be called from a control statement as follows:

APR(p_1,p_2)

Functions 5, 10, and 0,0 can be called only by jobs in the system libraries using the SYSTEM macro as follows:

SYSTEM APR,,p_2,p_1

All other functions can be called only from the operator's console.

JANUS

The unit record I/O package, JANUS, consists of the following PP programs (refer to figure 7-1 for JANUS interfaces).

Program	Description
1IQ	Input/output queue manager; sets up a control point to read jobs from card readers and to print or punch job output files; handles DSD commands and displays.
1IS	Called by 1IQ to load a group of overlays into the field length for use by 1IR.
1IR	Card reader, card punch, and printer driver.
1IU	Called by 1IR to backspace print files when necessary, or find proper spacing code array.
1PL	Dummy output file driver for plotter.
1FM	Dummy output file driver for film/hardcopy devices.

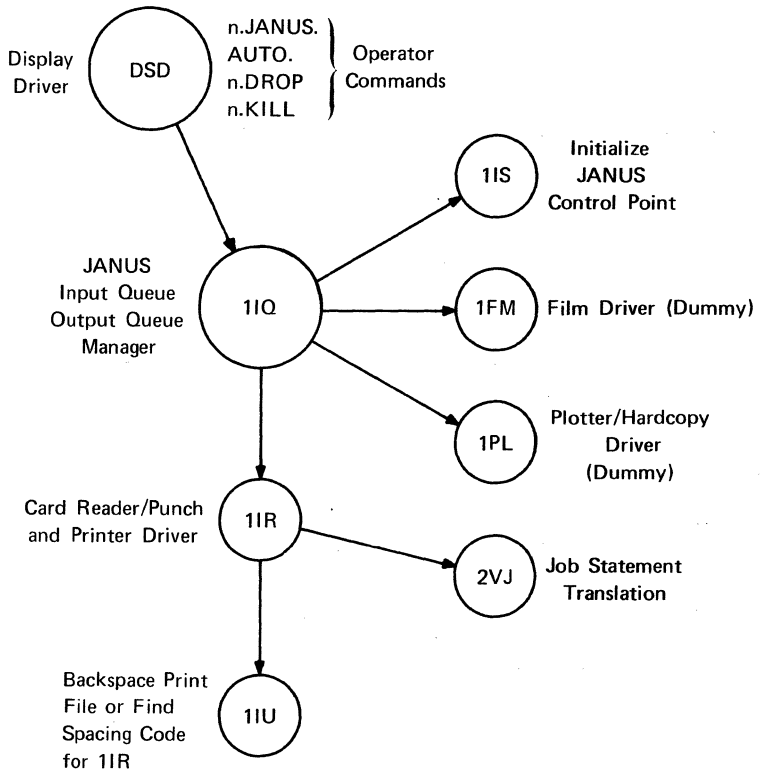


Figure 7-1. JANUS Interfaces

INTEGRATED SCHEDULER

The job scheduling scheme is made up of the following segments (refer to figure 7-2 for scheduler interfaces).

Segment	Description
Scheduler (MMGR)	CPU program operating at control point 0 in 2- to 3-millisecond cyclical intervals. It allocates control points and central memory to jobs and determines which jobs should be swapped or rolled in and/or out.
1IB	Brings jobs from input queue to CM queue; builds JDT entries for jobs.
1RN	PP program operating at control point 0 at about 1-second intervals. One of its functions is to age jobs in the input and output queues.
Swappers	PP programs called to swap or roll jobs in (1SI) or out (1SO) and to initiate new INTERCOM jobs.

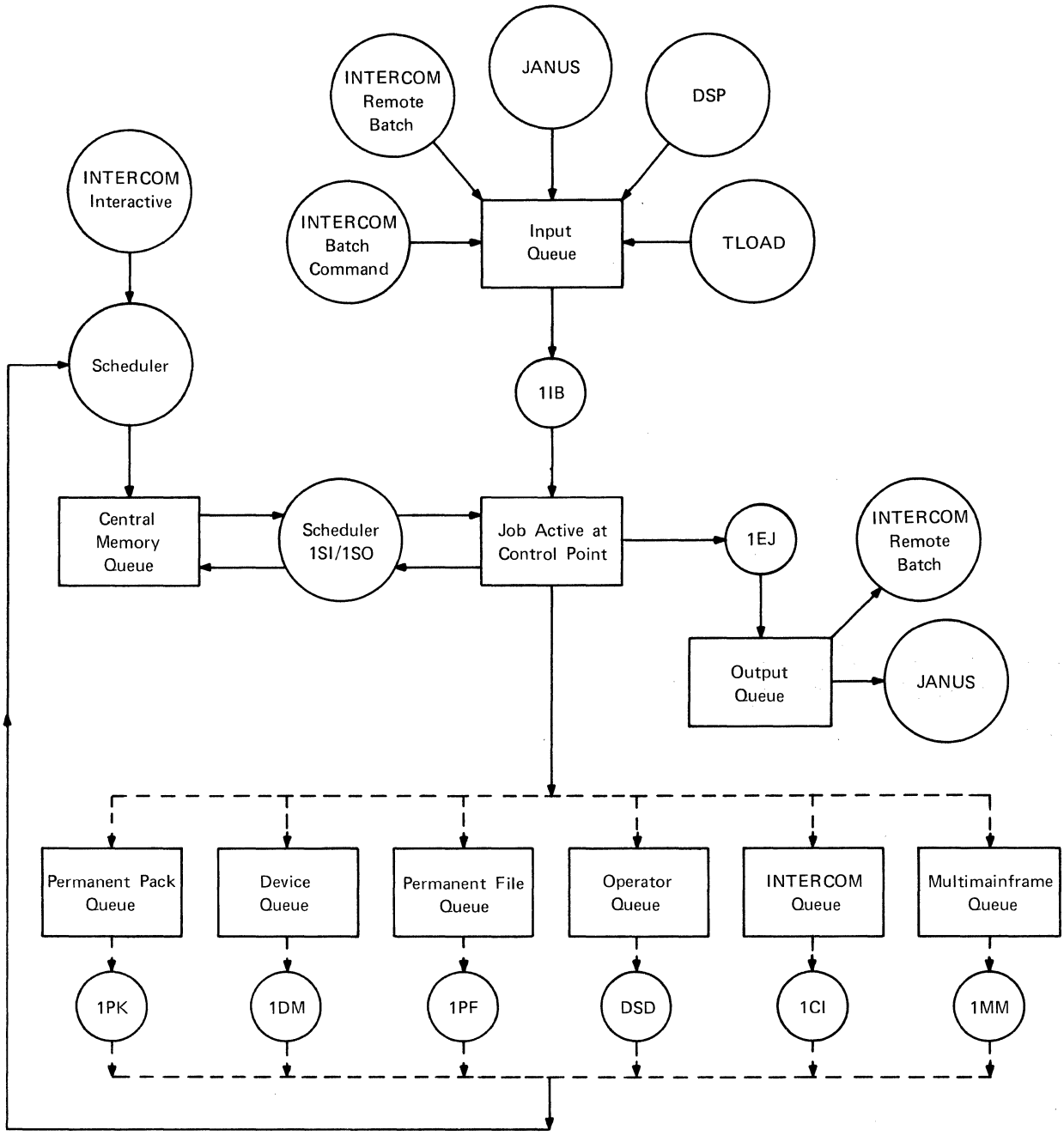


Figure 7-2. Integrated Scheduler Interfaces

Segment	Description
1DM	PP program which processes job entries in the device queue.
1PF	PP program which processes job entries in the permanent file queue.
1CI	PP program which processes job entries in the INTERCOM queue.
1QP	PP program which processes multiuser job entries in the INTERCOM queue.

Figure 7-3 shows the scheduler request stack in the system exchange package area.

JOB SCHEDULING

The user jobs in execution can be either swapped or rolled in or out of central memory by the integrated scheduler or by the operator. This feature of the job scheduling scheme allows dynamic allocation of control points and central memory. If the scheduler needs a control point or central memory, it need not wait for a job to terminate. Such a scheme permits more efficient servicing of a mixed group of jobs, such as INTERCOM, remote batch, local batch, and time-critical. A system job such as JANUS cannot be swapped or rolled.

ROLLING

When a job is rolled out, its entire field length is written to an allocatable file, and all of its central memory, except RA+0 through RA+77, is released. The control point remains assigned to the job, and its JDT entry is placed in the appropriate queue. When a job is rolled in, it resumes use of its control point.

A job is rolled out instead of swapped out under the following conditions.

- Job has direct-access ECS assigned and IP.ECSW is zero (user ECS swapping disabled).
- Job has nonallocatable equipment assigned.
- Job is in device queue.
- Job is in operator action queue.
- Job field length plus its required positive field length for swap-out exceeds IP.MFL (maximum field length a job can request) plus IP.POSFL or 131K.

SWAPPING

When a job is swapped out, all information concerning the job is written to an allocatable file. This information includes the job field length, control point area, dayfile buffer, dayfile pseudo FET, FNT entries assigned to the control point, and entries in the PP event stack and delay stack. The control point, CM, and ECS are released and made available for reassignment. The JDT entry is placed in the appropriate queue; it contains a pointer to the swap file containing the job. The job, when swapped in, can be assigned a different control point and a different area in CM and ECS.

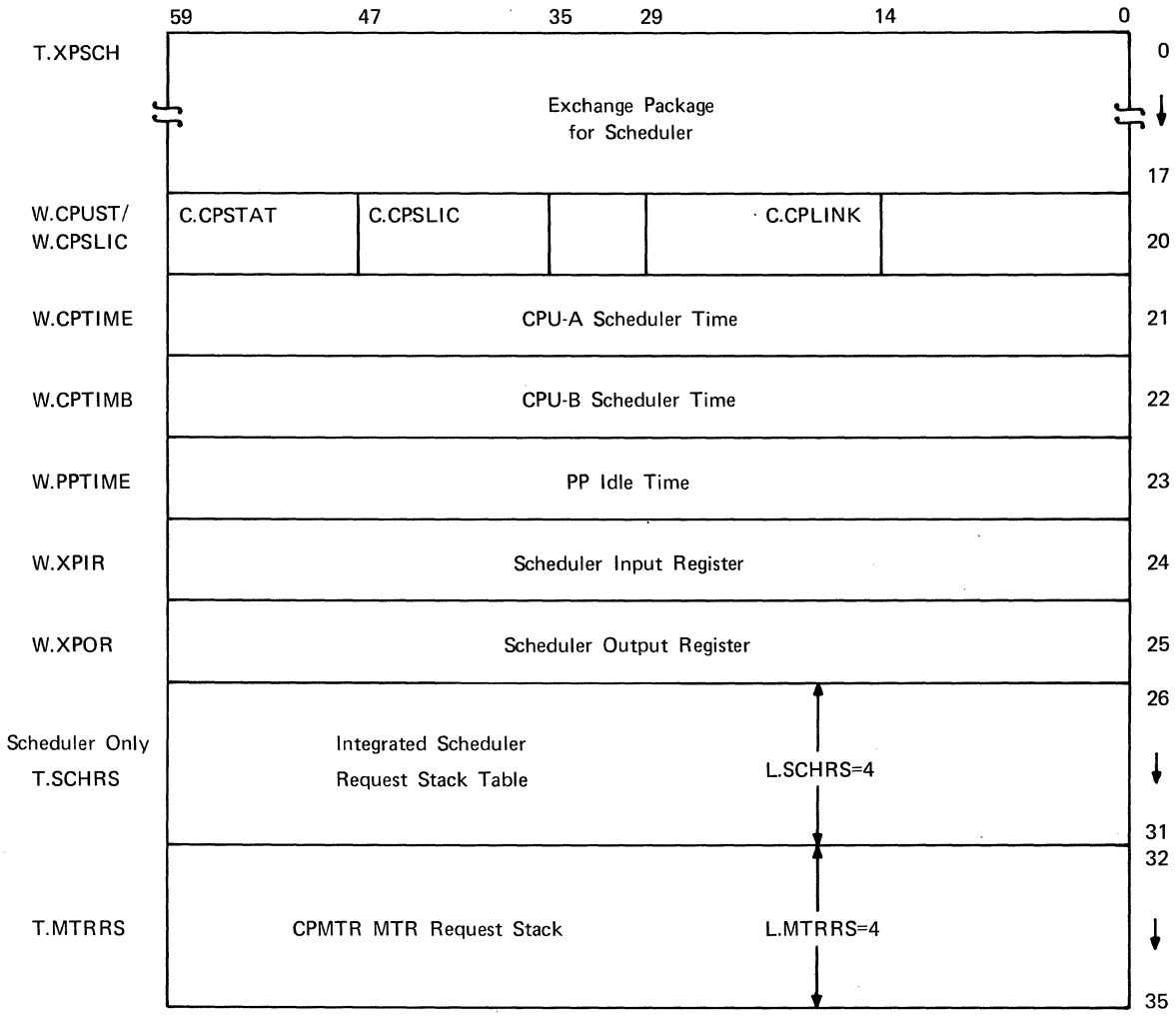


Figure 7-3. Scheduler Request Stack in System Exchange Package Area

A job is swapped out under the following conditions.

- Job is waiting for CM or ECS.
- Job is in permanent file queue.
- Job is waiting for permanent pack.
- Job is waiting for INTERCOM or graphics terminal I/O (job is in INTERCOM queue).
- Job is waiting for multimainframe action.

The scheduler can be initiated by a PP program using an M.SCH request. The scheduler is called in the following cases.

- When a PP program has a job to be added to the CM queue, the scheduler determines if the job has a priority high enough to cause any other jobs to be swapped out. If so, the scheduler initiates the swap-out and assigns the job to a control point. Otherwise, the scheduler adds the entry to the CM queue.
- When the field length of a job executing at a control point is reduced, the scheduler is called to search the CM queue for a new job to be scheduled.
- The scheduler is called periodically to determine if any job assigned to a control point has used up its quantum time period for CP and/or PP, and if any jobs in the CM queue have been aged enough to cause a swap-out. A job is swapped out during its quantum time period only if a job with a higher class priority has entered the CM queue.

JOB CONTROL AREA

Jobs entering the system for execution are placed into one of seven classes.

- Batch jobs using no nonallocatable resources.
- Batch jobs using one or more nonallocatable resources.
- INTERCOM (interactive) jobs.
- Multiuser jobs.
- Express jobs (for which operator has entered DROP, KILL, or RERUN).
- Graphics jobs.
- Batch jobs using direct access ECS.

An entry for each job class appears in a CMR table called the job control area. Each entry contains parameters affecting the scheduling of jobs in that class. These parameters include:

- Minimum queue priority.
- Maximum queue priority.
- Aging rate (time) for incrementing job priority.
- Quantum priority value.
- Quantum time length value.
- Pointer of first JDT in job class scheduling chain.

When a job is swapped out, it is given a queue priority value equal to the minimum queue priority for its class. This value is incremented with time at the aging rate for the class. When the incremented priority value reaches that of the maximum queue priority for the class, aging stops.

When a job is swapped in, it is given a queue priority value equal to the quantum priority for the class. When the quantum time length for the class has elapsed, the job priority is set to the minimum priority for the class.

A system operator can control job swapping overhead by adjusting the quantum time lengths for the different job classes in the job control area. Special DSD commands and a display allow the operator to adjust the job control area parameters.

The pointer to the job control area is T.SCHJCA, which is in byte C.JCA of word P.SCH of the CMR pointer area. The format of the job control area appears in appendix B.

Scheduling parameters are further described in the NOS/BE Installation Handbook.

JOB DESCRIPTOR TABLE (JDT)

Each job to be scheduled for execution has a JDT entry containing information pertinent to its scheduling. If a job is not active at a control point, its JDT must be in one of the scheduling queues. Each job JDT in the CM queue is linked into one of the six job class scheduling chains originating in the job control area. A linking pointer to the next JDT in the chain appears in the first word of the JDT.

Entries in one of the job class scheduling chains of the CM queue are linked in order of their entry into the chain. Each entry contains the time when it was added to the chain. The current priority of an entry is computed by multiplying the length of time the job has been in queue by the aging rate for its class and adding the minimum class priority. Job JDT entries in the permanent file queue are also linked; JDT entries in other queues are not linked.

The job status code in the JDT is of prime importance. When a job is assigned to a control point, its JDT ordinal is in byte 4 of word W.CPJNAM in the control point area. If a job is not assigned to a control point, its status code in word W.JDMGR of the JDT indicates that it is in one of the following job scheduling queues.

- Central memory queue.
- Device queue.
- Permanent file queue.
- Permanent pack queue.
- Operator action queue.
- INTERCOM queue.
- Multimainframe queue.

JOB SCHEDULING QUEUES

Central Memory Queues

The CM queue consists of rolled-out and swapped-out jobs waiting for CM, ECS, and/or control point assignment.

The PP routine 1IB scans the job input queue for jobs to be assigned to a control point. When such a job is found, 1IB creates the JDT entry and requests that the scheduler put the entry in the CM queue. Jobs in scheduling queues, such as the permanent file queue, for which requested action has been completed, also are eligible to re-enter the CM queue.

Device Queue

If a job executing at a control point requests a tape that must be mounted, the REQ routine puts the JDT into the device queue and calls a swapper to roll out the job. The PP program 1TS is responsible for detecting when the requested device is ready and for calling PP program 1DM to put the JDT entry back into the CM queue. When the job is rolled back in, REQ is recalled to create a FNT entry for the file.

Pseudo channel CH.SCH is used as an interlock in the device queue. Jobs in the queue are rolled out rather than swapped out so that the message to the operator appears on the B display.

Permanent File Queue

When a job issues an ATTACH request for a permanent file, it may be denied access to the file because of permission conflicts with jobs already having the file attached on the same or any sharing mainframes. When this occurs, PFA calls 1PF to link the job's JDT entry to the APF entry of the file and swap the job out. The permissions requested by this job are stored by 1PF in the job's JDT entry. If exclusive access or control permission are requested, 1PF sets the priority lockout bit (S.PFPL) in the APF entry and in the PFC entry. This causes queuing of all subsequent attaches that do not require exclusive access or control permission.

Whenever a permanent file is returned, 1PC checks for jobs queued on the file. If any exist, 1PC calls 1PF to swap in as many as possible. 1PF checks the priority lockout bit and, if set in either the APF or PFC entries, selects the first job in the queue requesting control permission or exclusive access. If no other jobs in the queue require exclusive access or control permission, the priority lockout bit is cleared in the APF entry and in the PFC interlock byte for this mainframe in the PFC entry.

If the priority lockout bit is clear in both the APF entry and the PFC interlock byte of all sharing mainframes, 1PF scans the entire queue, selecting as many candidates for swap-in as possible. These candidates cannot conflict with each other nor with any jobs having the file attached on this or any sharing mainframe.

If PFA detects permission conflicts with jobs having the file attached on other mainframes, it calls 1PF to queue the job on the APF entry and to set bit S.PFMMF in the APF entry. 1RN scans the APF table periodically, looking for entries having bit S.PFMMF set and calling 1PF when such an entry is found. 1PF then reads the PFC entry for any file having the bit set. If 1PF determines that any jobs can be swapped in, it does so, clearing S.PFMMF if no jobs remaining in the queue conflict with jobs on other mainframes.

Permanent Pack Queue

If a job attempts to mount a member of a device set and the device is not on line, the job is swapped out or rolled out. Operator-initiated jobs abort. When the device being requested is on and free, all jobs waiting for the pack are returned to the CM queue.

Routine 1PK is responsible for processing the queue. If the request for the device is generated by an I/O request, the I/O function continues from the point of interruption.

Operator Action Queue

When a job makes an operator action request, its pause bit is set in the control point area, its JDT entry is placed into the operator action queue, and then it is rolled out. Such jobs are rolled out rather than swapped out, so that the message to the operator appears on the B display. Whenever the scheduler finds a job pause bit set, it calls a swapper to put the job in the operator action queue, unless the PP routine processing the request indicates that the job is to be put into some other queue. For example, REQ sets the pause bit and then requests the job to be put into the device queue rather than the operator action queue.

When the operator takes appropriate action, the job is again eligible for the CM queue. It is rolled in and resumes execution at its control point.

INTERCOM Queue

Jobs waiting for input from an interactive INTERCOM or graphics terminal or waiting until output can be sent to a terminal are swapped out and put into the INTERCOM queue.

JOB ADVANCING

When the scheduler has assigned a job to a control point, the job advancing routine 1AJ is called into a PP. This section describes how the 1AJ routine processes the control statements in a batch origin job.

1AJ reads one PRU (100_8 CM words) from the control statement record of the job's input file into the control statement buffer of the control point area (locations W.CPAF to W.CPCAL). 1AJ stores the input file's FST entry for the next PRU of the control statement record in word W.CPFST of the control point area. The source file from which the control statements are read can be changed from the job's input file to another file using CYBER Control Language (CCL). For further information on specifying a different control statement file, refer to Job Control Statement Source File in this section.

The job statement of a batch origin job is processed by routine 1TJ and is skipped by 1AJ. 1AJ reads the next statement from the control point area buffer and sets an index to the buffer in word W.CPCC. The statement just read is processed as described in Control Statement Processing in this section (the sequence of operations is shown in figure 7-4). After that statement has been processed, 1AJ is reloaded and, using the buffer index in W.CPCC, repeats the procedure. When all statements in the buffer have been read and processed, 1AJ reads another PRU from the input file's control statement record into the buffer and resets W.CPFST to the next PRU. The 1AJ routine continues either until all statements in the input file's control statement record have been processed, the job is terminated because of an error, or an EXIT statement is encountered.

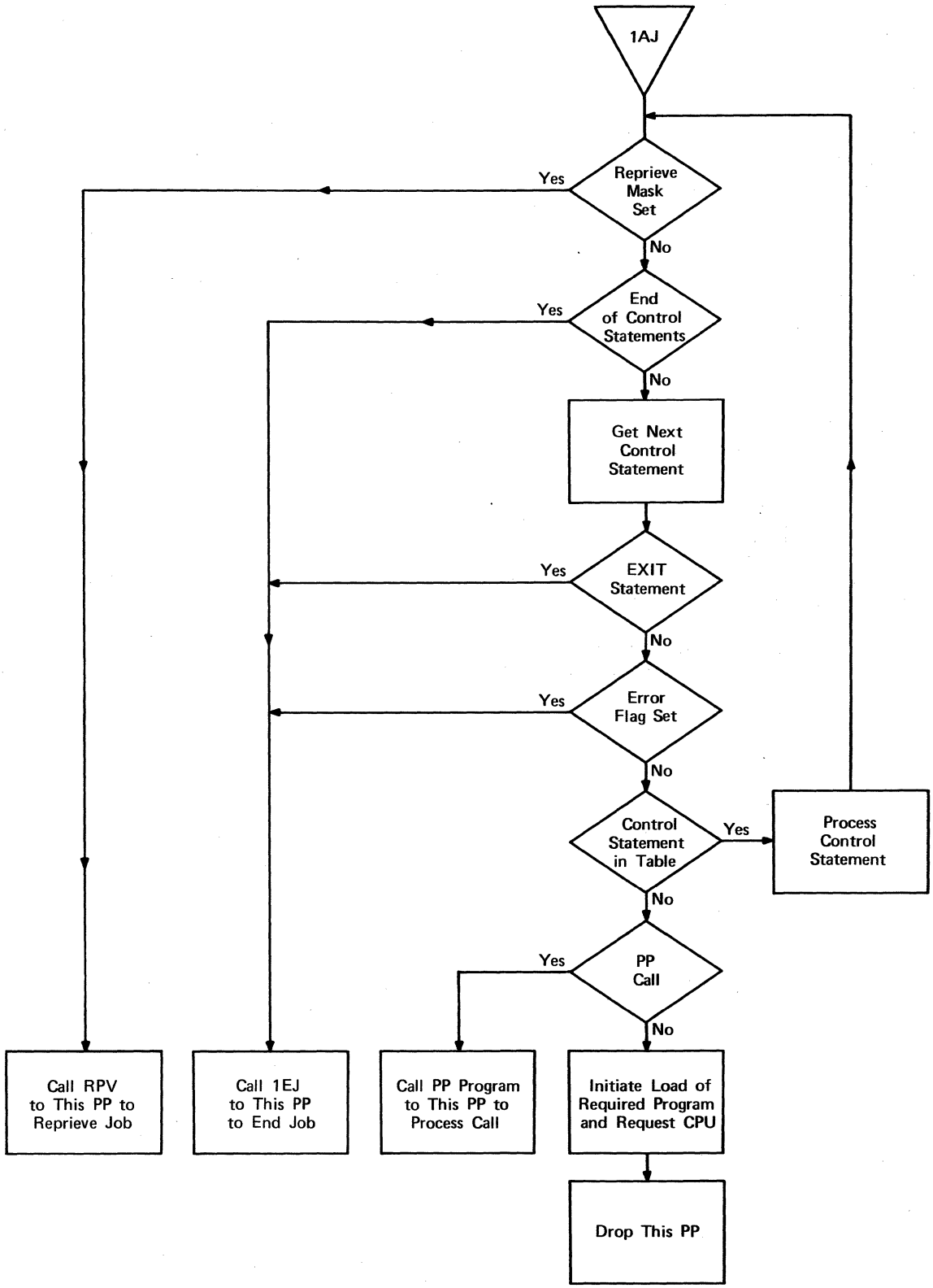


Figure 7-4. Control Statement Processing Flowchart

CONTROL STATEMENT PROCESSING

When 1AJ processes a control statement, it performs the following sequence of operations.

1. 1AJ compares the verb with a list of verb names that are treated as special cases. If there is a match, 1AJ completes the processing of the statement and continues with the next statement in the buffer.

2. If a match is not found, 1AJ attempts to find a local file of the same name as the verb. If a file is found, the statement is issued to the dayfile, copied to RA+70_8 (RA.CCD) through RA+77_8, and the parameters are separated and stored in RA+2 (RA.ARG) through RA+52_8. Then the loader is called to load the file.

3. If the verb is not a file name, 1AJ looks for a PP routine that can be called by a control statement and that has the same name as the verb. If such a PP routine is found, the statement is issued to the dayfile, and the parameters are separated and stored in RA.ARG. For all routines except the routine VSN, the first two parameters are converted to binary and stored in bits 35 through 0 of the input register. Then the PP routine is loaded on top of 1AJ.

4. If the verb is not a PP routine name, 1AJ searches in order the global library (as long as the libraries are resident in CM) for an entry point that can be called by a control statement and that has the same name as the verb. If such an entry point is found and the program is an absolute overlay, 1AJ separates and stores the statement, loads the program at RA+100 (RA.ORG), and starts it.

5. If an entry point is not found or the program is not an absolute overlay, 1AJ loads the system loader, which completes processing of the control statement.

The following special case names are recognized and processed by 1AJ.

ACCOUNT

Routine ACCOUNT is loaded from the system library, and execution is initiated. 1AJ does not issue a dayfile message.

COMMENT

The control statement, without the keyword COMMENT, is issued as a dayfile message. If the message is longer than 40 characters, it is issued in 2 parts.

MODE

The exit mode is set as specified on the MODE statement. If the mode value is greater than 7, a control statement error message is issued.

LIMIT

This statement sets a limit on the amount of mass storage that can be allocated to the job. The value is given as a decimal number of 4096 CM-word blocks and is stored in the control point area, word W.CPMSLM, as the mass storage limit in number of PRUs. If the job does not contain this statement, an installation-defined value (IP.SMS) is used. A running PRU count is kept in the same control point word; if the limit is exceeded, the job is terminated.

SWITCH

The parameter value causes the corresponding bit to be toggled in the C.CPSSW byte of word W.SSW in the control point area.

RFL,fl or RFL,CM=fl

The new CM field length value (fl) is set in byte C.CPNFL of word W.CPCC, and the CM reduce bit in W.CPLDR1 is cleared. The new field length must not exceed the field length given on the job statement or the current value in C.MFL in P.MFL.

RFL,EC=fle

The new ECS field length (fle) is requested immediately, and the ECS reduce bit in W.CPLDR1 is cleared. The new ECS field length must not exceed the field length given on the job statement or the current value of C.MFLE in P.MFL.

Permanent file control statements (CATALOG, ATTACH, EXTEND, RENAME, PURGE, and so on)

Each permanent file control statement is copied into the job field length, starting at RA+70. No dayfile message for the statements is issued, since the password must be removed from the statement for security. CM program PFCCP is loaded to process the statement.

EXIT

The exit flag bit is in byte C.CPFP of word W.CPFP in the control point area. If a control statement error occurs or if a binary deck containing fatal assembly or compilation errors is loaded, the abort flag bit in the same byte is set. If the EXIT statement contains a parameter, the exit flag bit is set when the error flag in byte C.CPEF of word W.CPEF on the control point area is set. If the EXIT statement does not contain a parameter, the exit flag bit is set only when the error flag is set and the abort bit is not set. PP routine 1EJ is called to do end-of-job processing.

SETNAME

This statement sets the control point default set either to the name specified or, if no name is specified, to the system default set.

SUMMARY

PP overlay 6BM is loaded to issue job statistics to the dayfile.

LOGIN

Routine LOGIN is loaded and execution is initiated. 1AJ does not issue a dayfile message.

JOB CONTROL STATEMENT SOURCE FILE

Control statements for a job are usually read from the first record of the job's input file. However, CCL can change the source file so that control statements (usually CCL procedure files) can be read from a different file.

The ACCSF macro retrieves the name and position of the current control statement file. If a different control statement is used, the current file can be restored later.

The format of the macro is

ACCSF lfn,save

lfn Address of word to receive name of the current control statement file.

save Address of word to receive current position of the control statement file.

The ENCSF macro activates a different control statement file at the position specified by the calling program.

The format of the macro is

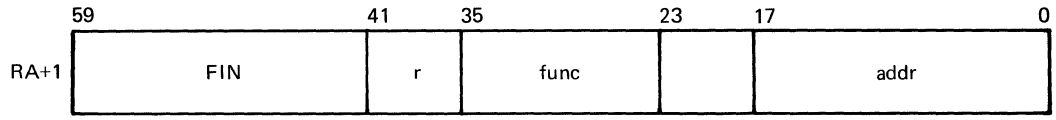
ENCSF lfn,restore

lfn Address of word containing the name of the new control statement file.

restore Address of word containing new position of control statement file.

The ACCSF and ENCSF macros are defined in the common deck ACTCOM.

Execution of either macro results in the following RA+1 request for PP routine FIN.



r Recall; set to 20_8 by macros.

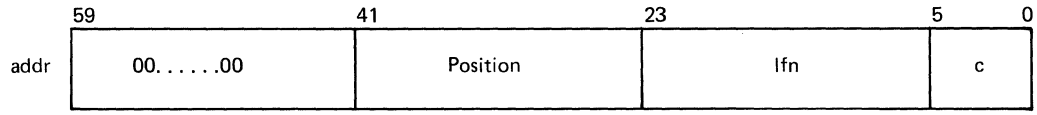
func Function code.

2 ACCSF

3 ENCSF

addr Address of a pointer word generated by ACCSF or ENCSF. The pointer word contains the address of the words containing lfn and the position of the control statement file.

The pointer word addr has the following format.

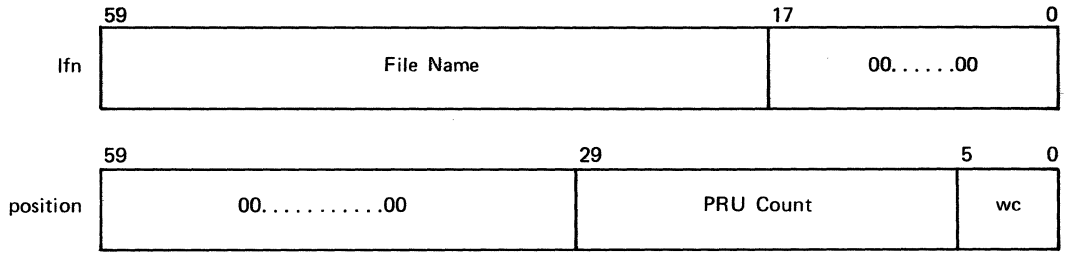


position For ACCSF, address of word to receive current control statement file position. For ENCSF, address of word containing new control statement file position.

lfn For ACCSF, address of word to receive name of current control statement file. For ENCSF, address of word containing name of new control statement file.

c Completion flag; must be zero before macro execution. Set to one when function completes.

The words containing lfn and control statement file position have the following format.



file name	A legal file name, one to seven characters. ACCSF sets file name to zero if the current control statement file is invalid or missing. Both ACCSF and ENCSF ensure that the next control statement is null (end-of-job) if the file name is invalid or missing.
PRU count	Position of control statement file in PRUs counting from the beginning of information (BOI=1).
wc	Word count; position of word within current PRU ($0\text{-}77_8$).

JOB TERMINATION

NORMAL TERMINATION

Normal job termination occurs when the error flag value is zero and all control statements for the job have been processed (an end-of-record mark for the control statement record on the job input file has been read) or when an EXIT or EXIT,S statement has been encountered. The end-of-job processor, 1EJ, is called by 1AJ to terminate the job by performing the following steps.

1. Dispose of all files associated with the job as follows:
   a. Drop local files on system devices and all equipment or storage associated with them.
   b. Release nonzero disposition files to control point 0 for further processing of output files. Scratch files are zero disposition files and are dropped.
   c. Update the TAPES table; rewind and unload tapes as appropriate.
   d. When tape scheduling is used, return reserved tapes to the system pool.
   e. Call overlay 1PC for disposal of permanent files and local files on user device sets. 1PC calls DSM if private sets are associated with the job.
2. Transfer the job dayfile to the OUTPUT file associated with the job.
3. Release storage for the job and set the job name to NEXT. The control point clear bit will be honored by the scheduler when it clears the control point area.

The following sections describe the disposal of specific files.

Permanent Files

Each time a permanent file is to be disposed of, an entry is made in a list to be processed by 1PC. 1PC deletes the FNT/FST entry for the file. The file still exists on a mass storage device. The attached permanent file (APF) entry for the file is updated or deleted, as appropriate. A purged file is no longer permanent and is processed like any other mass storage file.

Local Files

If the file is on a nonallocatable device, the equipment is dropped. If the file is on an allocatable device, the file space is evicted. In either case, the FNT/FST entry is cleared. 1PC disposes of local files residing on private device sets.

Input File

The job input file is not dropped until after the dayfile has been transferred to the job output file and the output file has been released to control point 0. Then the disk space for the job input file is evicted and its FNT/FST is cleared.

Output File

After the dayfile has been copied to the job output file, output file FNT is modified so that the file name is replaced by the job name, and the file is assigned to control point 0 and rewound. The file is then ready for processing by JANUS.

If the output file is on a nonallocatable device or pack, the equipment is dropped and the FNT is zeroed because JANUS cannot process files on nonallocatable devices.

If the file is on an allocatable device, the FNT is rewritten so that the file name is replaced by the job name, and the file is assigned to control point 0 and rewound. Its disposition code is retained. For INTERCOM batch files, the user ID is picked up from the control point area and stored in the FNT.

Dayfile

After the dayfile has been copied to the job output file, its FET is reset. If part of the dayfile is on disk, the disk space is released and its FST is cleared.

ABNORMAL TERMINATION

Abnormal termination of a job occurs when the error flag value is nonzero. If the error flag value is other than KILL (value 7) or RERUN (value 10), 1EJ performs the following steps.

1. Attempts to flush each local file (that is, to send unwritten data from the file's CM buffer to mass storage) that has a disposition code other than scratch (print, punch, and so on), or has both the disposition code of scratch and the file flush flag set in its FET.
2. Dumps the contents of the exchange package and the contents of memory.
3. Performs normal termination processing as described in the previous section.

The dump includes 100_8 words before the error stop through 100_8 words after the error stop (P-100_8 to P+100_8). It is written to the output file immediately preceding the job's dayfile. A dayfile message describes the error.

If an error flag value other than KILL or RERUN occurs and an EXIT, EXIT(S), or EXIT(U) statement follows in the control statement record, 1AJ resumes job processing after that statement is encountered. If EXIT(C) is encountered prior to EXIT, EXIT(S), or EXIT(U) and the conditions mentioned above are true, the job terminates at that point.

If the operator has killed the job, the error flag value is set to 7. Permanent files are processed as for normal termination. The FNT is zeroed, and disk space or equipment is released for all other files, including output and nonzero disposition files. The message JOB KILLED is issued to the system dayfile, the control point area is reset, and the job name is cleared. No output is produced for killed jobs.

If an operator has entered a RERUN command, the error flag value is set to 10, and the job input file is returned to the job input queue. Permanent files and family disk pack files are processed as for normal termination. All other files are dropped. A predayfile is created with its location contained in a supplement to the input file FNT. The predayfile becomes part of the job's dayfile when the job is rerun.

If abnormal termination occurs as the result of an ABORT macro call, job processing proceeds in accordance with the ABORT call parameters as discussed in the NOS/BE Reference Manual.

JOB POST-PROCESSING UTILITIES

During execution of a user program, various errors (such as an arithmetic mode error or exceeded time limit error) could cause the program and job to terminate prematurely. Normally, the operating system places a message in the dayfile and dumps the exchange package and 100_8 words of CM preceding and following the error location. The system then resumes job processing at an EXIT statement (if one is included) or terminates the job.

However, at the request of the user, the operating system returns control to the user program when an error condition is detected or the program terminates normally. Control is given to a routine supplied by the user which specifies procedures to be performed after abnormal or normal program termination. The NOS/BE Reference Manual describes the RECOVR and REPRIEVE macros and direct calls to the PP program RPV, which performs program recovery.

LIST-OF-FILES ADDRESS — GETLOF AND SETLOF MACROS

The list-of-files address is an 18-bit field in word 43 of the control point area. Currently it is used only by system programs (such as CYBER Record Manager) to access files which must be processed by a recovery routine if a job step ends with an error. (The system clears the list-of-files address when each job step begins.) The GETLOF and SETLOF macros provide access to the field containing the list-of-files address.

The format of the GETLOF macro is

```
label    GETLOF    addr,recall
```

label An optional symbolic address.

addr Address of the word to receive the current value in the list-of-files address field.

recall Optional recall parameter.

The format of the SETLOF macro is

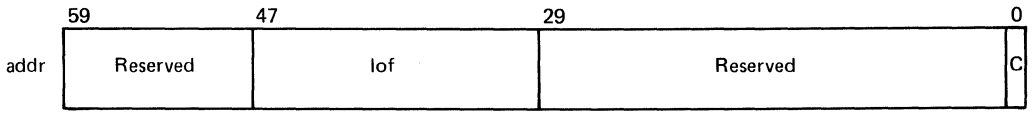
```
label    SETLOF    addr,recall
```

label An optional symbolic address.

addr Address of the word which contains the new value to be placed in the list-of-files address field.

recall Optional recall parameter.

For both GETLOF and SETLOF, addr has the following format



lof An 18-bit field containing current value in list-of-files address field (returned by GETLOF) or new value to be placed in list-of-files address field (set by user with SETLOF).

C Completion flag; cleared by macro initially and set to one when operation completes.

GETLOF and SETLOF are defined in the common deck ACTCOM.

JOB CONTROL WITH LOGICAL IDENTIFIERS

Logical identifiers (IDs) specify logical groups or functions. They allow extended control over job initiation and job/file processing. For example, logical IDs can be used in a single operating system to group all the jobs for a special printer. Those jobs must enter the system with the same logical ID.

The ID table, resident in CM, contains the information needed to process logical IDs. When the ID table has a length of zero, logical IDs are not allowed by the system. A job specifying a logical ID must wait to be processed until the operator enters that logical ID in the ID table. In the preceding example of grouping jobs for a special printer, the operator should wait until the printer is available, then enter the logical ID in the system's ID table and thus allow the jobs to execute.

A mainframe, in this discussion, is a hardware configuration controlled by a dedicated operating system. A linked mainframe communicates with a mainframe other than itself, and a multimainframe system consists of linked mainframes. Each linked mainframe in a multimainframe system communicates with the other mainframes. The ID table contains the name of the host mainframe (host ID), the logical IDs associated with the host mainframe, and the names of the other linked mainframes (physical or link IDs). A physical ID stored in the ID table is identical to the host ID of one of the other linked mainframes. For example, in a multimainframe system consisting of two linked mainframes, one could have an ID table listing a host ID of MFA and a physical ID of MFB. The other mainframe would then have an ID table listing a host ID of MFB and a physical ID of MFA.

The host IDs and logical IDs consist of three letters or digits. The host ID in a multimainframe system is unique because the third character of each host ID must be unique and must be a letter. As many as 58 decimal logical IDs can be associated with the host mainframe.

In a single mainframe system operating with an ID table of zero length, a job/file that does not specify a logical ID is processed on the system. A job/file that does specify a logical ID causes the system to flag the control statement as an error and terminate processing.

In a single mainframe system operating with an ID table containing the host ID and the list of associated logical IDs, a job/file that does not specify a logical ID is processed on that system. A job/file that specifies a logical ID causes the system to inspect the ID table for a matching logical ID. If a matching logical ID is found, the job/file is processed. If no match is found, processing waits until the operator places the specified logical ID in the ID table and the match is found.

In a multimainframe system, an ID table exists for each linked mainframe. A job/file that does not specify a logical ID is processed by the host system. A job/file that specifies a logical ID causes the host system to inspect its ID table. If a match is found in the list of logical IDs, the job/file is processed on the host system. If a match is not found, the host system uses the physical IDs in its ID table to seek a match for the logical ID. If a match is found, the job/file is processed by the appropriate linked mainframe. If a match is still not found, the job/file waits in queue on the host system until a match for its logical ID is found, either for the host system or for a linked mainframe system.

CEVAL MACRO

The CEVAL macro enables a hardware diagnostic routine to determine the status of equipment it is going to use. Before the user calls the macro, the diagnostic routine must set up an eight-word parameter block describing the equipment to be used and the type of operations (read-only or read/write) that will be performed. The system returns a code in the parameter block granting or denying the request. The routine should use this response code to generate an appropriate message for the user.

When the system grants a CEVAL request for a device that is nonallocatable and not an operator-assigned magnetic tape unit, it reserves the device by setting the maintenance flag and control point number in the device's EST entry. The system cannot access the device until the job releases it with the CEVAL request to return the unit to the system (function code 3). Up to five nonallocatable devices can be reserved at a time. All reserved devices are released automatically at the end of the job.

When the system grants a CEVAL request for a magnetic tape unit that was assigned previously by the operator, the system does not set the maintenance flag in the tape unit's EST entry. Therefore, it cannot be released with a CEVAL request (return unit to system request). If necessary, the unit can be released with the system control statements or macros RETURN or UNLOAD. When requesting an operator-assigned tape unit, only the mnemonics MT or NT should be specified in the parameter block; the EST ordinal, channel, equipment, and unit numbers must not be specified or the request is denied. CEVAL can request only one seven-track and one nine-track operator-assigned tape unit at a time.

The diagnostic routine must be allowed system access; that is, either it must reside in the system library and be called when the system is in engineering mode (enabled by the ENGR console command), or the job calling the routine must be initiated from the console. If either the routine is not allowed system access, the parameter block is outside the job's field length, or the routine improperly requests or releases a device, the system sends an error message to the dayfile and aborts the job without setting the complete flag.

The CEVAL macro is defined in the common deck ACTCOM.

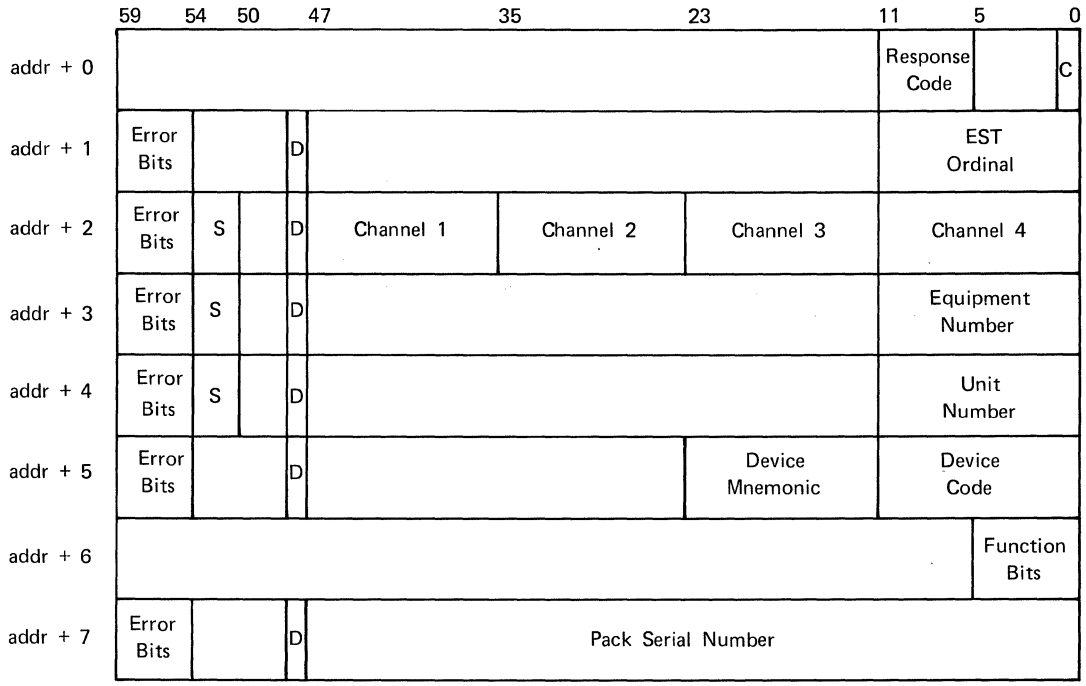
The format of the macro is

CEVAL addr,vo

addr Location of the eight-word parameter block.

vo Optional parameter ignored unless vo=VO. When VO is specified, the system aborts the job if it is not allowed system access; otherwise, the system completes the macro call by setting the complete flag in the first word of the parameter block.

The format of the parameter block is



Word	Bits	Description
addr+0	11-6	Response code.

Code	Significance
0	Validation granted.
2	Validation granted; element might be shared with another mainframe.
4	Validation denied; could not find element.
5	Validation denied; critical information may be destroyed.
6	Validation denied; element in use.
7	Validation denied; share byte not set.

0 Completion bit.

Word	Bits	Description
addr+1 through addr+5, addr+7	59-55	Validation error bits.

Bit Set	Significance
59	Error in word.
58	Error in byte 1.
57	Error in byte 2.
56	Error in byte 3.
55	Error in byte 4.

addr+1 through addr+5, addr+7	48	Data word.
addr+1	11-0	EST ordinal of element.
addr+2 through addr+4	54-51	Share bits; requests exclusive access if bits are set.

Bit Set	Significance
54	Error in byte 1.
53	Error in byte 2.
52	Error in byte 3.
51	Error in byte 4.

addr+2	47-36 35-24 23-12 11-0	Channel numbers.
addr+3	11-0	Equipment (controller) number.
addr+4	11-0	Unit number.
addr+5	23-12	Device mnemonic as shown under device code description.
	11-0	Device code.

Code	Mnemonic	Device
0001	AM	Reserved for installation.
0002	AY	7054/844-2x disk.
0003	AZ	7054/844-4x disk.
0004	AYF	7154/844-2x disk.

Word	Bits	Description		
		Code	Mnemonic	Device
		0005	AZF	7154/844-4x disk.
		0006	AH	819 disk.
		0007	AJ	885 disk.
		0010 thru 0017		Reserved.
		0020	CR	405 card reader.
		0021	CP	415 card punch.
		0022	LQ	Reserved for installation.
		0023	LR	580-12 line printer.
		0024	LS	580-16 line printer.
		0025	LT	580-20 line printer.
		0026	LRP	580-12 PFC line printer.
		0027	LSP	580-16 PFC line printer.
		0030	LTP	580-20 PFC line printer.
		0031 thru 0037		Reserved.
		0041	MTM	Reserved for installation.
		0042	MTS	667 seven-track tape.
		0043	MTB	667 seven-track tape with block ID.
		0044	MTA	677 seven-track tape.
		0045 0046 0047		Reserved.
		0051	NTM	Reserved for installation.
		0052	NTS	669 nine-track tape.
		0053	NTB	669 nine-track tape with block ID.

Word	Bits	Description		
		Code	Mnemonic	Device
		0054	NTA	679 nine-track tape.
		0055	NTG	679 GCR nine-track tape.
		0056 0057		Reserved.
		0060	DC	Reserved for installation.
		0061	YC	Reserved for installation.
		0062	DCX	Reserved for installation.
		0063	YCX	Reserved for installation.
		0064	FE	255x host communications processor.
		0065	CS	7077-1 LCC.
		0066	SC	6673 data set.
		0067	CC	6683 coupler.
		0070	ED	DDP.
		0071	AX	ECS coupler.
		0072 thru 7677		Reserved.
		7700 thru 7777		Reserved for QSEs.
addr+6	6-0	Function bits.		
		Bit	Significance	
		0	Load controller memory (controller share byte must be set).	
		1	Will write on unit.	
		2	Will not use preallocation area.	
		3	Return unit to system (overrides other function bits; EST ordinal is required.	
		6	Designated elementis not in system EST.	
addr+7	47-0	Pack serial number (binary).		

ACCESS TO LOADER WORD — GETLC AND SETLC MACROS

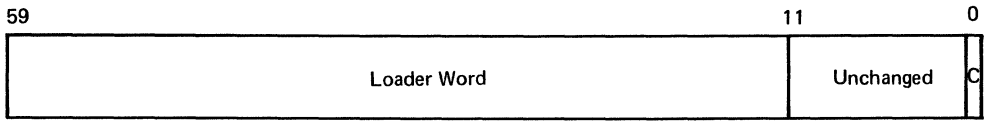
Certain system programs, such as the Loader and CYBER Interactive Debug, use information contained in W.CPLDR1, the loader word, in the control point area. The GETLC and SETLC macros provide access to this word. Although any program can call these macros, they are intended for use only by system programs.

The GETLC macro format is

GETLC addr

addr Address of the word to receive the current loader word.

The GETLC request is made with auto recall set. The initial contents of addr is ignored. When the operation is complete, addr has the following format.



Loader word — Bytes 0 through 3 of W.CPLDR1. Refer to word 55 (W.CPLDR1) of the control point area table in appendix A for the format of the loader word.

C — Completion flag; initially cleared by macro; set to one when the function completes.

The SETLC macro format is

SETLC addr

addr Address of the word which contains the new value of the loader word.

The SETLC request is made with auto recall set. The user sets bits 59 through 24 of addr to correspond to the desired loader word settings. The remainder of addr is ignored except for bit 0. Bit 0 is the completion flag, similar to the flag for GETLC. It is cleared by the macro initially and set to one when the function completes.

Before setting bits in the loader word, the user should read the current contents of the word via the GETLC macro. In this way, the user can avoid setting bits unnecessarily and changing bits unintentionally. Bit 49 (S.CPLP), which indicates a nonsystem library program, can be set but never cleared.

GETLC and SETLC are defined in the common deck ACTCOM, which is included in CPCTEXT, CPUTEXT, and SCPTEXT.

EDITLIB 8

INTRODUCTION

The purpose of EDITLIB is to:

- Create deadstart tapes.
- Create and maintain system libraries.
- Manipulate the running system to effect temporary changes.
- Modify the running system or the deadstart tape by merging in permanent changes and creating a new deadstart tape.
- Create and modify a user library consisting of a group of central processor routines or overlays.

To create a new deadstart tape and/or running system for a specific configuration, the user obtains an unconfigured deadstart tape from Software Manufacturing and Distribution. This tape contains the binary decks of the entire operating system to be used to build the new system. After appropriate assemblies and compilations have been performed, EDITLIB is called by selected control statements and directives to modify the current system. Thus, a new deadstart tape and/or running system can be created, which reflects the system as it is physically configured (figure 8-1).

All system code is not required to be located in one massive system library. Object code can reside in either a system library or a user library. Thus, EDITLIB can be used in two forms, SYSTEM EDITLIB, which is described in this section, and USER EDITLIB, which is described in the NOS/BE Reference Manual.

SYSTEM EDITLIB can function like USER EDITLIB. Directive statements before the first READY, after the last COMPLETE, or between a COMPLETE and the next following READY are interpreted as they would be in USER EDITLIB. However, USER EDITLIB does not allow the addition of PP programs.

EDITLIB can handle object code only if it is the type processed by the CYBER loader; 7000-type object code cannot be processed. All object code for EDITLIB input must be in system logical format. All procedure records must be in zero-byte terminator format. S, L, or other nonsystem format tapes cannot be processed by EDITLIB.

CHARACTER SET

EDITLIB interprets $ ( ) = + - * / , and . as delimiters. Blanks are free characters and are removed unless in a literal delimited by dollar signs ($). From the character set, the user can create:

Symbol	Single character or a string of characters.
Vocabulary word	Symbol that has meaning as defined by the semantics.
Name	Symbol that is defined by syntax and semantics.

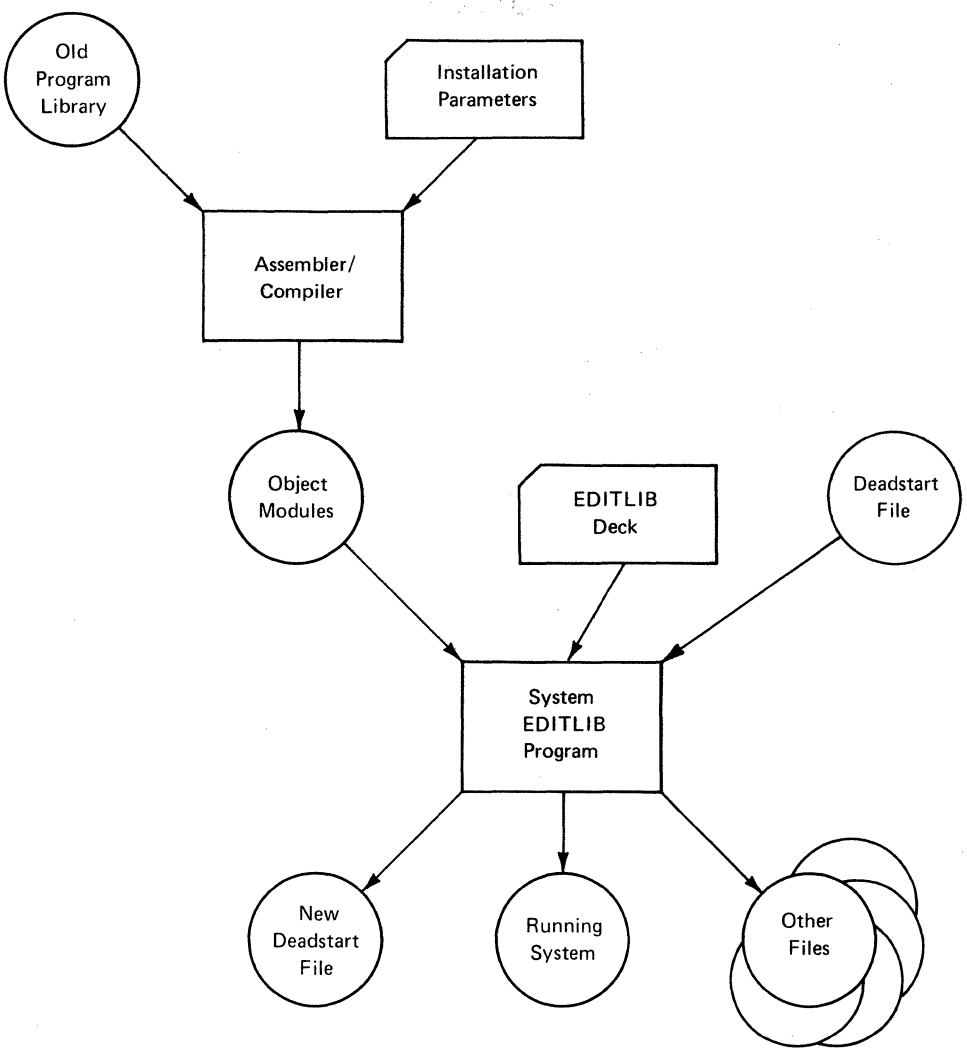


Figure 8-1. Basic Usage of System EDITLIB

When any delimiter and/or a blank is used to form a symbol, the symbol must be enclosed between $ signs (refer to example). When a $ sign is to be used as a character within a symbol, an additional $ sign must be written immediately adjacent to each $ sign desired. The only character that is removed is the first $ sign of each pair of $ signs encountered (refer to example).

Whenever a name could be interpreted as a number or a vocabulary word, that name should be enclosed between $ signs.

Examples:

$-DSD-$	Symbol is -DSD-.
$$$CIO$	Symbol is $CIO.
$D$$I$$$$$$-$	Symbol is D$I$$$-.

The input to EDITLIB is the first 72 columns of an 80- or 90-column card image. A directive must be completely contained in one statement.

SYNTAX AND SEMANTICS

The formats of the various directives and the meanings associated with the symbols are presented here. The syntax, as illustrated in the examples, must be followed.

Any directive having parameters must be terminated by a right parenthesis. If the directive has no parameters, it must be terminated by a period.

Whenever a violation of syntax is encountered, EDITLIB sets the abort flag. After EDITLIB has completed a given task, such as checking all directive formats, it checks the abort flag. When set, EDITLIB aborts. EDITLIB continues to check a directive, even if it has located an error in that directive. EDITLIB proceeds to the next directive only after it has completed analyzing the current directive or after it is unable to determine the syntax of the directive.

The following are the symbols used in the syntax.

Symbol	Significance
[ ]	Enclosed items are optional.
⊥	Or.
( )	Enclosed item is the actual symbol.
p	Name of a routine.
lfn	File name.
n	Positive or negative integer.
pn	Library name.
r	Residence:

CM	Central memory.
DS	Disk.
ECS	Extended core storage; if no ECS available, then default to disk.
EM	Extended core storage; if no ECS available, then default to central memory.
Default	Disk.

Interval — Program name or range of program names in one of the following forms.

P_1

P_1+P_2

P_1-P_2

$P_1/P_2/P_3$

P_1 is a single program name. $P_1/P_2/P_3$ is valid only for ADD and REPLACE directives. It is acceptable to replace either P_1 or P_2, but not both, with [*]. EDITLIB ignores a directive under the following conditions.

- Unable to locate P_1.
- Unable to locate P_2.
- Unable to locate P_1 when P_2 = [*].
- Unable to locate P_2 when P_1 = [*].
- Unable to locate P_1 when there is no interval.

* — Replaces p in the syntax. When * is encountered, EDITLIB considers all remaining routines on the file proceeding from the current position. If the file is a library, the current position is the first entry in the PNT.

\+ — Designates an inclusive interval.

Example: Given the ordered set of records A, B, C, D, E, F, and G on a file, then A + G represents all members of the file.

\- — Designates an exclusive interval.

Example: Given the ordered set of records A, B, C, D, E, F, G, H, I, and J on a file, then D - H represents all members of the file except D, E, F, G, and H. This leaves A, B, C, I, and J.

/ — Delimits routine names.

Notes:

- EDITLIB assigns a number to each directive it processes.
- EDITLIB lists each input statement it receives on the file designated for listable output by the L parameter on the EDITLIB statement. The directive, as interpreted by EDITLIB, appears directly below the input statement. The interpreted directive is prefaced by the number assigned to this directive. All comments about the directive follow the interpreted directive. EDITLIB is a two-pass program. EDITLIB interprets and checks all directives during the first pass; it then proceeds to completion, unless a fatal error is encountered. EDITLIB may stop then, however, in the second pass. Error messages are generated by referencing the number that is assigned to each directive. Error messages are issued whenever EDITLIB finds it cannot comply with the user's request. EDITLIB also issues messages when it is unable to determine what the user is trying to accomplish.

- The directives ADD and REPLACE produce a CONTENT directive on the routines involved.
- Any interval specified in a directive is interpreted to mean:

  In a reference to a file which is a library, the interval is formed by the entries in the order they appear in the program name table (PNT). The interval begins when P_1 is encountered and ends when either P_2 or the last entry in the PNT is encountered. The table is not handled circularly.

  In a reference to a file that is not a library, the interval is formed by the order of the routines on the file. To find P_1, EDITLIB searches from current file position to end-of-file, then rewinds and searches the file from the beginning, if necessary. The interval ends at whichever occurs first, P_2 or end-of-file. The file is handled circularly only when searching for P_1.

- In a system EDITLIB, all directives that do not specifically name a particular library and are not contained between a LIBRARY and FINISH directive refer to the PP library. This means that the PP library is the default library when no library is specified. In the user EDITLIB, no PP programs are allowed.
- EDITLIB does not support TEXT, SEGLOAD records and/or DATA input (7000 LIBEDT). If TEXT/SEGLOAD record/DATA input is found, a message is issued and the unrecognized records within the interval are ignored.
- CYBER EDITLIB ignores miscellaneous 7000 LIBEDT directives TYPE, ERROR, PCOPY, PMOVE, MOVEC, COPYC, and DELETEC, and prints a warning message on each directive.

SYSTEM EDITLIB

EDITLIB FILES

EDITLIB uses the following files.

File Name	Cataloged	File Use
ZZZZZ01	Yes	RESET.
ZZZZZ02	Yes	RESTORE.
ZZZZZ03	Yes	System extension.
ZZZZZ04	Yes	SYSTEM.
ZZZZZ05	No	Interpreted directives.
ZZZZZ06	No	ECS resident routines library file (special case system file).
ZZZZZ07	No	Entry point name table spill file.
ZZZZZ08	No	Program number table spill file.
ZZZZZ10	No	Program name table spill file.

File Name	Cataloged	File Use
ZZZZZ11	No	External reference table spill file.
ZZZZZ12	No	External reference collection spill file.
ZZZZZ13	No	Library or deadstart program collection.
ZZZZZ14	No	Scratch.
ZZZZZ15	No	PP program name table spill file.
ZZZZZ16	No	Library name table spill file.
ZZZZZ23	Yes	Current directory file.

System EDITLIB produces the following type of files.

File Use	lfn	pfn
RESET	ZZZZZ01	ZZZZZ01
RESTORE	ZZZZZ02	ZZZZZ02
System extension	ZZZZZ03	ZZZZZ03
SYSTEM	ZZZZZ04	ZZZZZ04
ECS library	ZZZZZ06	
Current directory	ZZZZZ23	ZZZZZ23

The files that are made permanent are cataloged using the following values.

File Name	ID	RP	TK	CN	MD	EX
ZZZZZ01	SYSTEM	999	SSSSSSSSS	XNOX	XNOX	---
ZZZZZ02	SYSTEM	999	SSSSSSSST	---	---	---
ZZZZZ03	SYSTEM	999	SSSSSSSSU	XNOX	XNOX	---
ZZZZZ04	SYSTEM	999	SSSSSSSSV	XNOX	XNOX	XNOX
ZZZZZ23	SYSTEM	999	DIRECTORY	XNOX	XNOX	---

The allocation of ECS is controlled by Monitor. Since the allocation of ECS involves the creation of tables and pointers by Monitor, the deadstart loader does not duplicate this effort to set up ECS files. An EDITLIB job is required to place routines into ECS after deadstart.

EDITLIB maintains a permanent file (ZZZZZ23) for the deadstart loader. This file contains the current system directory as it appears in CMR.

When EDITLIB modifies the running system, it adds the new routines to the end of file ZZZZZ03. Using this method, EDITLIB does not prevent others from accessing the file, except during the short period of time while the new directory is written to CMR. Bit S.EDTRUN in byte C.DSFLAG or word P.LIB is set when the directory is changed.

SPECIAL HANDLING OF LOCAL FILE NAME SYSTEM

The system file is named ZZZZZ04. To allow the use of the local file name SYSTEM and to clarify the handling of the libraries on a system-type file, the user must understand that the use of the file name SYSTEM depends on the context in which it is used.

If the file name SYSTEM does not appear between the directives READY and COMPLETE, it is treated as a local file whose name just happens to be SYSTEM.

If the file name SYSTEM does occur between a READY and COMPLETE directive, the CONTENT directive treats it as a local file name. Also, ADD and REPLACE treat it as a local file name whenever the LIB parameter is absent from the optional parameter list on those directives.

Examples:

ADD(interval,SYSTEM,AL=1)

Add the interval from the local file named SYSTEM.

ADD(interval,SYSTEM,LIB)

Add the interval from the running system file (ZZZZZ04 or ZZZZZ03), specifically the PP library.

ADD(interval,SYSTEM,LIB=NUCLEUS,FLO=1)

Add the interval from the running system file (ZZZZZ04 or ZZZZZ03), specifically the CP library called NUCLEUS, with the field length override (FLO) option.

CONTROL STATEMENT

The program call statement is

EDITLIB(SYSTEM,t_1,t_2,...,t_n)

Parameter	Description
SYSTEM	This parameter is a password. It is used by EDITLIB to determine what kind of EDITLIB run is to be performed. This key is a program parameter, which cannot exceed nine characters; it allows restricted access to the running system directory.
t_i	One of the following:

t_i	Description
RESET	This parameter instructs EDITLIB to replace the current system directory with the directory as it appeared after initial deadstart. Whenever the RESET parameter is encountered, EDITLIB uses the directory from the RESET permanent file to replace the directory presently in CMR; it changes this directory as prescribed by directives in the input file. EDITLIB replaces the current directory with the directory just modified. A copy of the current directory is saved on the RESTORE permanent file before the change takes place.

Parameter	Description
t_i	Description
RESTORE	This parameter informs EDITLIB to replace the current system directory with the directory as it appeared before the last EDITLIB took place. When the RESTORE parameter is encountered, EDITLIB uses the directory on the RESTORE permanent file to replace the directory presently in CMR. The rest of the procedure EDITLIB follows is the same as for an EDITLIB (SYSTEM, RESET), except that the current directory is not saved on the RESTORE file.
I=lfn	This symbol designates the file containing EDITLIB directives. If it is not present, lfn INPUT is used. If INPUT is used, any binary input to EDITLIB on this file must be in the order in which it is requested. This file is not searched. I, appearing alone, is equivalent to I=INPUT.
L=lfn	This symbol designates the file of listable output from EDITLIB. If it is not present, lfn OUTPUT is used. L, appearing alone, is equivalent to L=OUTPUT.
MSGL=n	This parameter determines the type of list or dayfile output generated during execution; n equals 0 to 8. The default value is 0. Refer to table 8-1.

TABLE 8-1. DIRECTIVE INTERPRETATION AND EXECUTION

n	Directive Interpretation		Directive Execution	
	Message to Dayfile	List Input Statement	List Prefix Table	List Optional Parameters
0	No	Yes	Yes	Yes
1	Yes	Yes	Yes	Yes
2	No	No	No	Yes
3	Yes	No	No	Yes
4	No	Yes	Yes	No
5	Yes	Yes	Yes	No
6	No	No	No	No
7	Yes	No	No	No
8	No listing produced		No listing produced	

Parameter	Description

t_i	Description
ERROR=n	This parameter determines when an abort occurs. The degree of severity of each error is checked against the error flag set by the ERROR parameter. Bit 3 set indicates a critical error, bit 2 set indicates a serious error, bit 1 set indicates a minor error, and bit 0 set indicates a warning error. Checking is performed only when the COMPLETE directive is encountered in system mode.

n	Fatal	Critical	Serious	Minor	Warning
0	Yes	No	No	No	No
1	Yes	Yes	Yes	Yes	No
2	Yes	Yes	Yes	No	Yes
3	Yes	Yes	Yes	No	No
4	Yes	Yes	No	Yes	Yes
5	Yes	Yes	No	Yes	No
6	Yes	Yes	No	No	Yes
7	Yes	Yes	No	No	No
8	Yes	No	Yes	Yes	Yes

When ERROR=n is not specified, any error or warning causes job termination.

EDITLIB errors are explained in the Diagnostic Handbook.

Examples:

```
EDITLIB(SYSTEM)

EDITLIB(SYSTEM,RESET)

EDITLIB(SYSTEM,MSGL=1,RESTORE)

EDITLIB(SYSTEM,I=INFILE,MSGL=0)

EDITLIB(SYSTEM,I=INFILE,L=OUTFILE)

EDITLIB(SYSTEM,I,L)

EDITLIB(SYSTEM,MSGL=7,I,L=INFILE)
```

OPTIONAL DIRECTIVE PARAMETERS

Optional parameters are positionally independent.

Parameter	Description
AL=al	al is one to four octal digits from 0 to 7777. Default is 0. AL stands for access level and is a parameter of the ADD and REPLACE directives. Of the 12 bits, 11 are meaningful to INTERCOM only. The rightmost bit of this field is used by the system to regulate access to the program through control statement requests. When this bit is set to 0, the routine or entry cannot be called from a control statement. The setting of this 12-bit field is determined by the binary representation of the octal digits specified by AL. INTERCOM has divided the 12-bit field into three sections. The first section is 1 bit in size and is the rightmost bit of the field. The remaining 11 bits are divided into two sections according to the installation parameter, IP.ACES. IP.ACES=n defines the number of bits to be reserved for the access level. The remaining bits (11-n) are used as permission bits. When the use of a command is requested, the user's access level is checked to determine whether it is greater than or equal to the access level of the command. If it is, the permission bits of the user are compared against the permission bit of the command. If a match is found, the user is given access to the command. If either test fails, permission is not granted.
FL=k	k is from 0 to 377777 octal. Default is 0. FL is a parameter of the ADD and REPLACE directives and stands for field length. It is meaningful for CM programs only. The value is used in allocating memory when the program is called. This field length must take into account the CM needed by the loader plus all programs, overlays, and segments that can be loaded by the specified program. (Refer to FLO parameter description.)
FLO=m	m is 0 or 1. Default is 0. This parameter can only appear on an ADD or REPLACE directive. It sets or clears the field length override bit, which determines the execution field length. If set, the user's field length can override the FL value set in the library. The value used by the system to assign memory for execution of a library resident program is either the field length specified by the EDITLIB FL parameter (PFL) or the nominal field length (NFL). NFL is the last RFL request (if one was made), the field length specified on the job statement, or the default value IP.SFL. The system determines which field length to assign depending on the setting of the FLO parameter and the automatic field length flag (reduce bit).

Conditions	Field Length Assigned
PFL=0, reduce bit is off.	NFL
PFL=0, reduce bit is on.	IP.SFL
PFL≠0, FLO=0.	PFL
PFL≠0, FLO=1, reduce bit is off.	NFL
PFL≠0, FLO=1, reduce bit is on.	PFL

Parameter	Description
	The field length assigned cannot exceed that specified on the job statement. Any job that does not have sufficient field length for the library program is terminated abnormally.
LIB=pn	pn is the name of a library. This parameter can only appear on an ADD or REPLACE directive. It is used when the lfn specified in the directive is SYSTEM (meaning the running system) or a deadstart file. LIB specifies which library on the deadstart file is to be used to satisfy the directive.
NEW †	Mandatory parameter for the LIBRARY directive and optional parameter for the READY directive. When a deadstart file is being created, the lfn specified contains a new library or directory.
OLD †	Mandatory parameter for the LIBRARY directive and optional parameter for the READY directive. It is required in these directives when an existing library or an existing deadstart file is referenced by the lfn parameters.
r	Represents residency as follows:
	CM — Central memory. ††
	ECS — Extended core storage; if no ECS available, then default to disk. ††
	DS — Disk (default).
	EM — Extended core storage; if no ECS available, then central memory.

DIRECTIVE RESTRICTIONS

The following list indicates where directives must occur relative to LIBRARY-FINISH and READY-COMPLETE sequences. Any violations of these restrictions are diagnosed by EDITLIB before any directives are executed and can cause an abort accompanied by diagnostic error messages.

Directive	Restrictions
ADD	Inside LIBRARY-FINISH or inside READY-COMPLETE.
CHANGE	Inside READY-COMPLETE.
COMPLETE	Outside LIBRARY-FINISH and after READY.
CONTENT	None.
DELETE	Inside LIBRARY-FINISH or inside READY-COMPLETE.
ENDRUN	Outside LIBRARY-FINISH and outside READY-COMPLETE.
FINISH	After LIBRARY.
INCLUDE	Inside READY-COMPLETE.
INCLUDEP	Inside READY-COMPLETE.

† Either NEW or OLD is mandatory, not both.
†† Not allowed for procedure files.

Directive	Restrictions
LIBRARY	Not inside LIBRARY-FINISH.
LISTLIB	Outside LIBRARY-FINISH.
LISTLNT	Inside READY-COMPLETE.
MOVE	Inside READY-COMPLETE.
RANTOSEQ	Outside LIBRARY-FINISH and outside READY-COMPLETE.
READY	Not inside READY-COMPLETE.
REMOVE	Outside LIBRARY-FINISH and inside READY-COMPLETE.
REPLACE	Inside LIBRARY-FINISH or inside READY-COMPLETE.
REWIND	None.
SEQTORAN	Outside LIBRARY-FINISH and outside READY-COMPLETE.
SETAL	Inside LIBRARY-FINISH or inside READY-COMPLETE.
SETFL	Inside LIBRARY-FINISH or inside READY-COMPLETE.
SETFLO	Inside LIBRARY-FINISH or inside READY-COMPLETE.
SKIPF	None.
SKIPB	None.
TRANSFER	Inside READY-COMPLETE.
TRANS77	Inside READY-COMPLETE.

DIRECTIVE FORMATS

The following directive adds the specified program(s) from the lfn to the library under construction or modification.

ADD(interval,lfn[,r][,(AL=)al][,(FL=)k][,(LIB=)pn][,(FLO=)m])

If the lfn contains a library, the program name table of the library is searched to locate the specified programs to be added. If the file is not a library, the search for the specified programs begins at the current record position on the file and proceeds to end-of-file. If the programs are not found, the file is rewound and the search continues until all programs have been searched. If the programs are not found in the library or on the file, an error message is issued. If the programs to be added already exist in the library, an error message is issued. The ADD directive should not be used to replace the current program with the new program. This function is accomplished by a REPLACE directive.

Examples:

ADD(HIGH,LOW,ECS,AL=70,FL=1000)

ADD(HIGH,LOW,FLO=1)

ADD(HIGH,SYSTEM,LIB=A,ECS,AL=123)

The following directive is used to change the residence of the library tables in the LNT.

CHANGE(pn[,r])

If the library already has the specified residence or if the library is not part of the system, EDITLIB issues an appropriate comment.

Example:

CHANGE(FNT,CM)

The following directive informs EDITLIB that the directive list for the directory whose name appeared in the READY directive has been exhausted and there are no more modifications or additions to be performed on the directory.

COMPLETE.

This directive must be preceded by a READY directive.

The following directive is used to obtain information about a program or series of programs on a specified file (lfn).

CONTENT(interval,lfn)

The information is obtained from the program-expanded prefix table and the program itself. The CONTENT directive applies to all files other than a deadstart file or a LIBRARY file. If information is desired on PP programs, the SYSTEM parameter must be entered on the EDITLIB call statement. In a multifile situation, CONTENT must be specified for each file.

Example:

CONTENT(ONE,FILE1)

The following directive removes all entries from the library table that refer to the specified programs.

DELETE(interval)

Examples:

DELETE(FEAR)

DELETE(FEAR+HATE)

The following directive indicates that execution of directives is to stop.

ENDRUN.

All directives following ENDRUN are checked, but not processed. This is an optional directive; if it is not found, one is generated. If this directive is found and EDITLIB finds an error in a directive after the ENDRUN card, it is flagged but does not stop EDITLIB from executing directives which precede ENDRUN.

The following directive designates the end of the list of directives that affects the library defined by the last LIBRARY directive.

FINISH.

Each LIBRARY directive must have a FINISH directive. It is permitted to have multiple (LIBRARY, FINISH) sequences within a single directive sequence.

The following directive adds the library on the specified file to the directory.

INCLUDE(pn,lfn[,r])

This allows a user library to be made part of the running system.†

Examples:

INCLUDE(SMOG,AIR,ECS)

INCLUDE(SMOKE,LUNGS)

The following directive allows an old PP program library to be added to the current directory model.

INCLUDEP(lfn)

The PP program library in the current model must be empty when the directive is encountered. Residency and access levels of each entry are that of the entries in the source PP program library.†

During a system EDITLIB run, the following directive performs two functions.

LIBRARY(pn,(OLD)⊥(NEW)(,r))

When encountered outside of a READY and COMPLETE directive, the action taken is equivalent to a user's EDITLIB. In this case, the user is in a user's EDITLIB mode and thus (NEW=n) is supported, where n is the size of the PNT required by LIBEDT. When the directive is between a READY and COMPLETE, the action is also equivalent to that of a user's EDITLIB, but the LIBRARY is found using the library name table (OLD) or is added to the library name table (NEW).

Examples:

LIBRARY(FNT,OLD)

LIBRARY(RUN,NEW,ECS)

LIBRARY(COBOL,NEW)

The following directive is used to list a library which can be a user's library, running system, new system, and/or deadstart file.

LISTLIB(interval,lfn,[pn])

Example:

LISTLIB(COPY,SYSTEM,A)

When between READY and COMPLETE, the lfn listed is the running system and/or the new system under construction. When LISTLIB appears outside of a READY and COMPLETE, only a user's library and a deadstart tape can be listed. pn is meaningful when listing the running system, new system, or deadstart file. If not present, the PPLIB is assumed. LISTLIB cannot be present between a LIBRARY and a FINISH directive. LISTLIB must be called for each library on a file.

These directives send information about the specified program(s) on the lfn or on the lfn and in the pn to the output file. The information comes from the object deck being processed.

---

† When creating a new deadstart tape from the running system, programs that were ECS resident become disk resident.

Examples:

LISTLIB(COPY,FILE1,A)

LISTLIB(1AJ,SYSTEM)

The following directive must appear between a READY and a COMPLETE directive.

LISTLNT.

It lists the library name table of the file referenced on the READY directive. It can be used to list either the old system or new system LNTs. It does not list the LNT on an old deadstart file. There are no parameters.

The following directive changes the residence of CP and PP programs within a system directory.

MOVE(interval,r)

EDITLIB moves PP programs to disk (specified by DS), ECS (specified by ECS or EM), or CM. Movement of CP programs depends on the residence of the system library of which the CP program is a part. If the library is in CM, movement is unrestricted; if the library is on disk or in ECS, the new residence can be specified only as DS, ECS, or EM. The library directory residence must be at least as fast as the new residence.

After a library program is requested to be ECS resident and a GO response is received, the user should request LISTLIB to determine if the program is resident in ECS. This is advisable because output is generated before the actual move is made and, if ECS is not available, the program may reside on disk (if ECS was specified) or in CM (if EM was specified).

Examples:

MOVE(FILE1,DS)

MOVE(FILE1+FILE2,CM)

The following directive places a random user library on a sequential file in library format.

RANTOSEQ(lfn_1,lfn_2)

lfn_1 is the source file name; lfn_2 is the destination file name. All records not referenced by the PNT are lost.

The following directive is the first directive of any directive list pertaining to the lfn specified.

READY(lfn[,(OLD)⊥(NEW)])

Only those directives between a READY and a COMPLETE affect the lfn. READY is used in conjunction with a system EDITLIB run.

READY specifies that the operation is performed on a directory. The lfn is either the running system or the name of the file which contains the new system. If the name SYSTEM appears as the lfn, it means the running system whether the OLD parameter is specified or not; if OLD is specified, lfn must be SYSTEM. Directives following the READY directive can be classified into four major functions: those that modify the library name table, those that modify the PP library, those that modify the system library (they must have LIBRARY and FINISH directives to delimit them), and those that do not change any part of the system directory.

Examples:

Directive	Identical to
READY(FUTURE,NEW)	READY(FUTURE)
READY(SYSTEM,OLD)	READY(SYSTEM)

The following directive removes the library from the running system.

REMOVE(pn)

The library's entry in the library name table is removed.

Example:

REMOVE(FTN)

The following directive replaces the program(s) specified if it is already in the library, or adds the program(s) if it is not in the library.

REPLACE(interval,lfn[,r][,(AL=)al][,(FL=)k][,(LIB=)pn][,(FLO=)m])

The REPLACE directive in a system EDITLIB retains the AL, FL, FLO, and r values of the replaced program, unless otherwise specified.

Examples:

REPLACE(FIRST,MASTER,AL=70)

REPLACE(FIRST-FIFTH,MASTER,LIB=NUCLEUS)

In the first example, the previous FL, FLO, and r values are retained. In the second, the previous AL, FL, FLO, and r values are retained.

The following directive positions the specified file to the first record on the file.

REWIND(lfn[/lfn]...)

To accomplish this, EDITLIB issues a CIO request to do a REWIND. In system mode, the file name SYSTEM refers to the running system.

The following directive places a sequential user library on a random file in library format.

SEQTORAN(lfn_1,lfn_2)

lfn_1 is the source file name; lfn_2 is the destination file name.

The following directive permits the user to change the access level of a program already in the directory.

SETAL(interval,al)

al = 0 to 7777 (octal).

Examples:

SETAL(SECOND+THIRD,7770)

SETAL(FIRST,1)

The following directive permits the user to change the field length required to run the specified program already in the directory.

SETFL(interval,k)

k = 0 to 377777 (octal).

Examples:

SETFL(BYTE,100000)

SETFL(BIT+WORD,100000)

The following directive permits the user to change the field length override bit of programs already in the library.

SETFLO(interval,k)

k = 0 or 1.

The default value is 0 (override is not permitted).

Examples:

SETFLO(BIT11,0)

SETFLO(BYTE-WORD,1)

The following directive is the skip-forward and skip-backward directive.

SKIPF/SKIPB(N,lfn,[F]⊥[P])

If N is numeric, it represents a number of records to be skipped, unless the third parameter is present, in which case N indicates a number of files. If N is alphanumeric, the function is skip-by-name and the direction implied by the directive is ignored. The file is searched in the forward direction; if the name N is not found, the file is rewound and the search continues in the forward direction. When found, the lfn is positioned before the record of that name (N). If N is not found, the original position is maintained. The alphanumeric form is applicable to sequential files only. While in system mode, the file name SYSTEM means the running system.

The following directive is used to create deadstart tapes.

TRANSFER(interval⊥n,lfn)

It moves records whose prefix table has been removed or is to be removed. A check is made to determine if the records being moved have a prefix table. Because a program name can be all digits, it is necessary to delimit such programs for this directive; for example, program name is 026, the directive is TRANSFER(026,SYSTEM).

Examples:

TRANSFER(*,SYSTEM)

TRANSFER(10,SYSTEM)

The following directive is used to create deadstart tape records with prefix tables.

TRANS77(interval⊥n,lfn)

It performs the same functions that the TRANSFER directive does, with the exception that it does not remove prefix tables.

The following symbol is used to denote a comment field; whenever encountered in columns 1 and 2, EDITLIB treats it as a comment and does not check the statement.

*/

Example:

*/THIS ADDS A PROGRAM THAT WILL DO A/B*C

LIBRARY DIRECTORY ACCESS

PP Routines

To access the library directory, a PP routine first checks bit 59 of P.LIB. When bit 59 is set to 1, EDITLIB either is waiting to change or is changing the directory. Thus, the directory is unavailable. When bit 59 is set to 0, the directory is available.

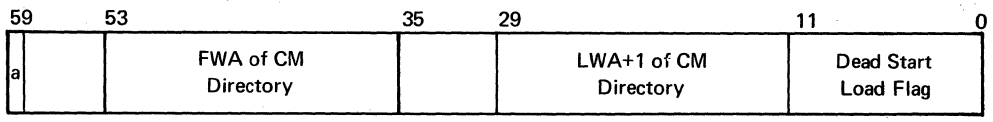
CP Routines

All CP routines accessing the directory should set bit S.CPLDAF in byte C.CPFLAG in word W.CPFLAG of their control point area. To access the directory, the following procedure should be followed. First, check bit 59 in P.LIB; if not set, then set bit S.CPLDAF. Then check bit 59 in P.LIB again. If it is now set, clear S.CPLDAF and wait until bit 59 in P.LIB is clear. After access is obtained and directory access is complete, clear bit S.CPLDAF.

EDITLIB

To access the directory, EDITLIB issues a monitor request to have PP routine MDI assigned to its control point. By passing MDI the appropriate code, MDI tries to change the directory. MDI sets bit 59 in P.LIB to 1 to lock out all other routines from initiating a directory access. It then checks bit S.CPLDAF in byte C.CPFLAG in word W.CPFLAG in each control point area. If the bit is set, it means the job has not completed its last directory access, and MDI goes into the event stack until the bit is cleared. After MDI has completed its task, it adjusts the FWA and the LWA+1 pointers in P.LIB and sets bit 59 to 0.

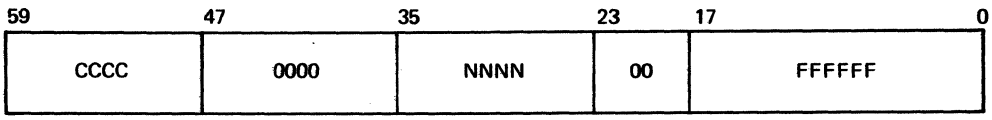
Library Directory Pointer Format



a Library change flag.

P.LIB The first word holds a library change flag (1 bit). This flag tells the PP and any CP program using the directory that EDITLIB either wants to change the directory or is changing the directory. Only when the flag in the control point areas is zero does EDITLIB proceed to change the directory.

PP Library Pointer Format



P.PPLIB FFFFFF is the address of the first entry in the PP program name table. NNNN is the number of entries in the PP program name table. CCCC is the position of CIO in the PP program name table. This value specifies the number of entries preceding CIO.

MDI sets this word to zero and waits 1 second before changing the directory.

FILES

SYSTEM FILES

A copy of all the routines referenced by the directory resides on a permanent file called ZZZZZ04 after initial deadstart. If a program or library subsequently is to be included into the directory, it is placed in the permanent file called ZZZZZ03. EDITLIB never writes, modifies, or extends permanent file ZZZZZ04. The new version of replaced routines resides on the file ZZZZZ03. Since EDITLIB does not alter any record currently on the file ZZZZZ04 or the file ZZZZZ03, routines can be read from either file while EDITLIB is adding new records to ZZZZZ03.

USER FILES

The user is responsible for library contents. EDITLIB does not overwrite any record on the file. It only adds records at EOF/EOI. This means that the user has a file with unused records when library programs are deleted or replaced. It is up to the user to remove this dead space by building a new library using the old one as a source. No permanent file requests are issued by EDITLIB. The user's file must be attached, cataloged, and extended. If the library is on tape, the user must build a new library each time an EDITLIB is performed on the library, using the old library as a source.

FILE AND LIBRARY POSITIONING

EDITLIB rewinds all files except INPUT before executing any directives. After a random library is written, it is rewound. When a new sequential library is written, it is left positioned after the end-of-file. New directories are rewound when completed.

Index to user's libraries on a random file (six words):

Word	Address
1	Entry point name table.
2	External reference table.
3	Program number table.
4	Program name table.
5	Entry point and external reference list.
6	Unreferenced PRU count (not an address).

Format of user's library on a sequential file:

Record	Contents
1	This record is three words long. 1. ***LIBRARY. 2. Library name (blank-filled). 3. Date of creation.
2	Entry point name table.
3	Entry point and external reference list.
4	External reference table.
5	Program number table.
6	Program name table.
7 to end-of-file	Routines.

SYSTEM SECURITY

EDITLIB uses the reprieve function to protect the system files and its scratch files. Under any normal or abnormal termination condition, EDITLIB control is turned over to a program that returns or releases all EDITLIB internal files.

MDI (MOVE SYSTEM DIRECTORY)

This PP overlay includes the program 6MD, which is used to control changes to the CMR directory. If 6MD is deleted, the running system may not be changed, but all other functions can be performed. MDI performs the following tasks.

1. Requests field length for directory in CMR.
2. Sets directory change flag.
3. Gets length of directory.
4. Places CM address into LNT for CM-resident libraries.
5. Copies directory to control point.
6. Copies directory to CMR.
7. Checks CP activity flags at each control point.
8. Writes current directory file.
9. Attaches FNT to ZZZZZ06 to EDITLIB control point.
10. Monitors EDITLIB error flag after GO.
11. Catalogs ZZZZZ01 and ZZZZZ02 at control point 0 with an ID of SYSTEM.

A parameter list is provided in EDITLIB for MDI. This list contains MDI request code and all necessary information. MDI returns information to EDITLIB in this list. This list is the MDI INPUT/OUTPUT register.

An EDITLIB change to the CMR library directory is incompatible with INTERCOM operation. If INTERCOM is running, MDI issues the appropriate message to the operator when any of the following EDITLIB directives are encountered for the reasons noted.

EDITLIB Directive	Reason for Incompatibility
ADD	Library is CM resident.
DELETE	Library is CM resident.
REPLACE	Library is CM resident.
CHANGE	CM is involved in the residency change.
INCLUDE	Unconditionally incompatible.
LIBRARY	Library is new or CM is involved in a residency change.
MOVE	CM is involved in the residency change.
REMOVE	Unconditionally incompatible.
INCLUDEP	Unconditionally incompatible.

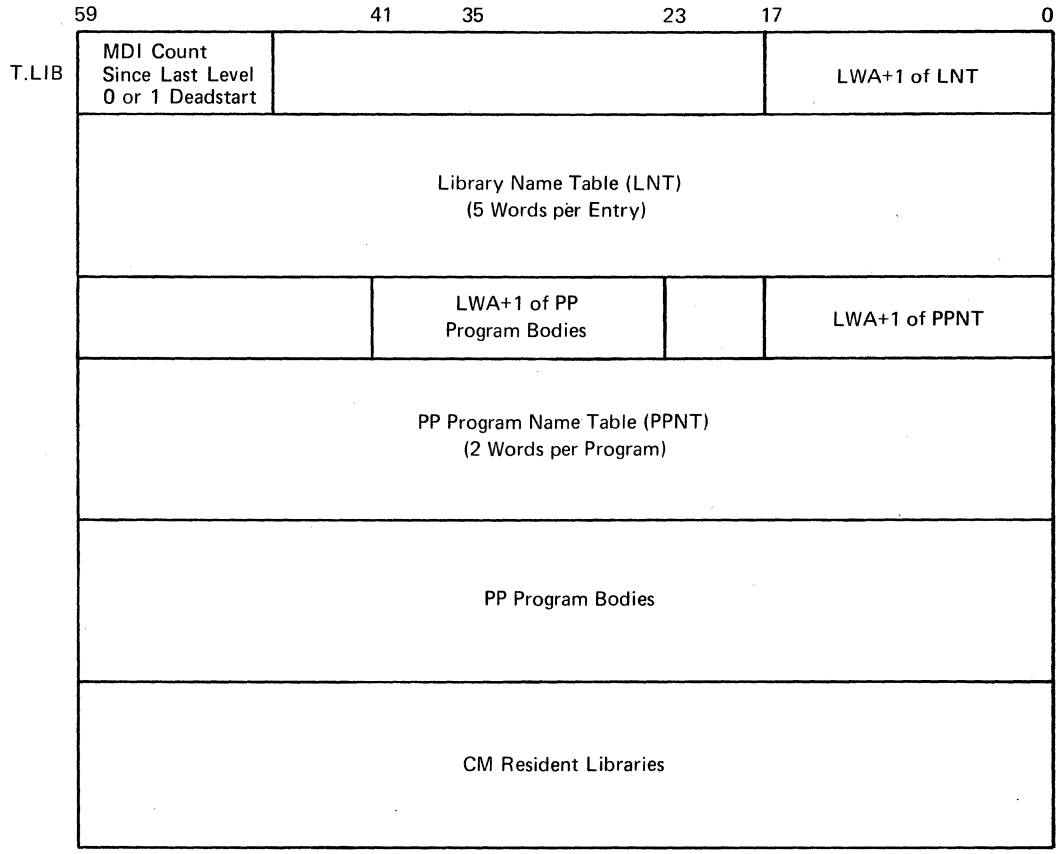
DIRECTORY/LIBRARY/PROGRAM LIMITS

The LNT allows a maximum of 27 libraries. A library can contain a maximum of 2047 programs, 2047 entry points, and 2047 internal references.

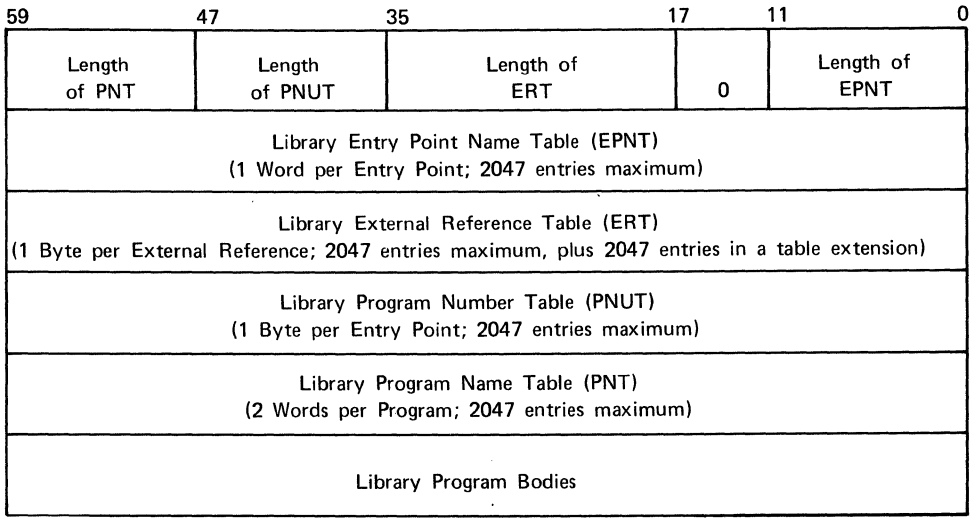
A particular program in the library can contain a maximum of 250 entry points and 500 external references. The PP program name table allows a maximum of 450 PP programs.

TABLE FORMATS

CMR DIRECTORY

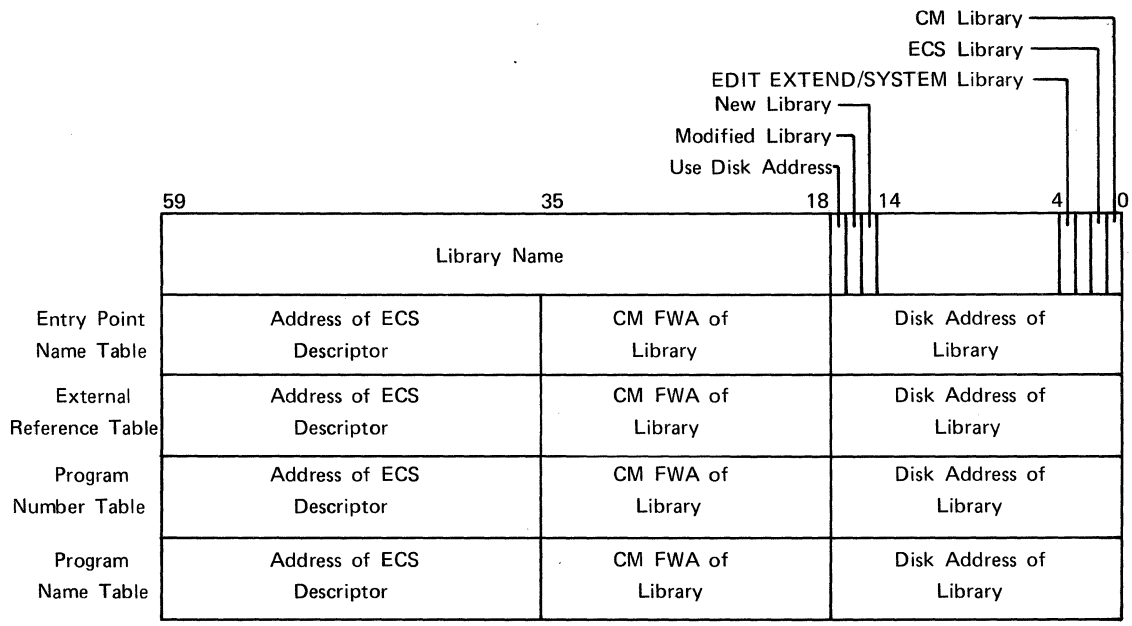


CM RESIDENT LIBRARY FORMAT



All information that is a part of the system directory and resides in ECS is also on the ECS file called ZZZZZ06. This is a special case system file.

LIBRARY NAME TABLE ENTRY



Fields not being used contain zero.

Word 1:

Bits 59-18		Library name; can contain up to seven alphabetic and/or numeric characters of which the first must be alphabetic.
Bit 17†	0	Residency addresses are valid.
	1	Use disk addresses only.
Bit 16†		Library is modified.
Bit 15†		Library is new.
Bit 3	0	Library on file ZZZZZ04.
	1	Library on file ZZZZZ03.
Bit 1††	0	Library is not ECS resident.
	1	Library is ECS resident.
Bit 0††	0	Library is not CM resident.
	1	Library is CM resident.

Word 2:

Address word for the entry point name table.

Word 3:

Address word for the external reference table.

Word 4:

Address word for the program number table.

Word 5:

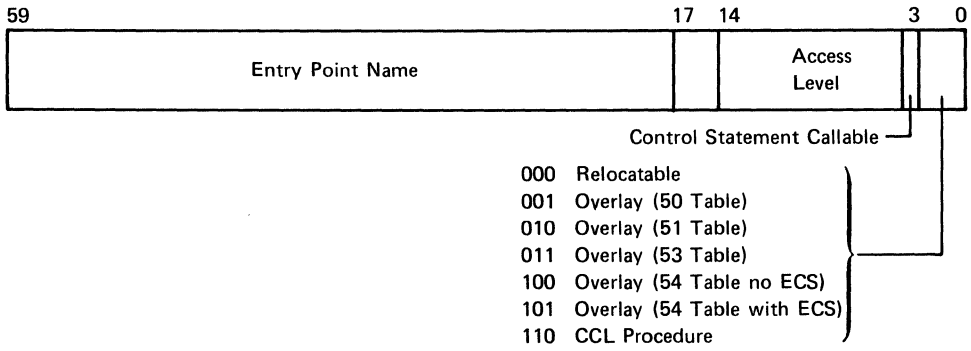
Address word for the program name table.

---

† Bits 15, 16, and 17 are set and cleared by EDITLIB only. They are used when EDITLIB is changing a directory that contains many libraries. Any routine accessing the library that has these bits set must use the disk address to obtain any library table and any routine that is part of this library. These bits are necessary due to the dynamic allocation of ECS and the nature of the new directory.

†† If bits 0 and 1 are both zero, the directory is on disk.

EPNT ENTRY FORMAT



Bits 59-18 Entry point name/entry point number value is equal to its position in the table relative to the start of the table. The table is sorted alphanumerically.

Bits 14-4 Access level bits for INTERCOM.

Bit 3

0 Entry point cannot be called from a control statement.

1 Entry point can be called from a control statement.

Bits 2-0

000 Entry point belongs to a relocatable program.

001 Entry point belongs to an absolute overlay which begins with a 50 table.

010 Entry point belongs to an absolute overlay which begins with a 51 table.

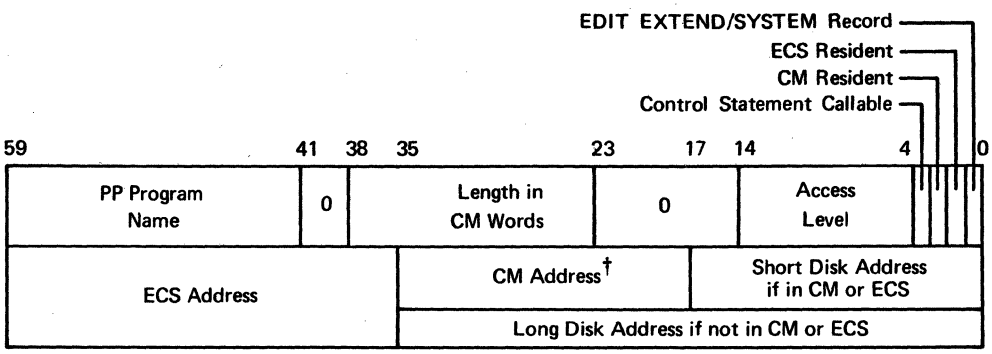
011 Entry point belongs to an absolute overlay which begins with a 53 table.

100 Entry point belongs to an absolute overlay which begins with a 54 table and has no ECS image.

101 Entry point belongs to an absolute overlay which begins with a 54 table and has an ECS image.

110 Entry point belongs to a CCL procedure.

PP PROGRAM NAME TABLE ENTRY



Word 1:

Bits 14-4 Access level bits for INTERCOM.

Bit 3
- 0 Program cannot be called from a control statement.
- 1 Program can be called from a control statement.

Bit 2
- 0 Record/routine is not CM resident.
- 1 Record/routine is CM resident.

Bit 1
- 0 Record/routine is not ECS resident.
- 1 Record/routine is ECS resident.

Bit 0
- 0 Record is on (pfn) ZZZZZ04.
- 1 Record is on (pfn) ZZZZZ03.

Word 2:

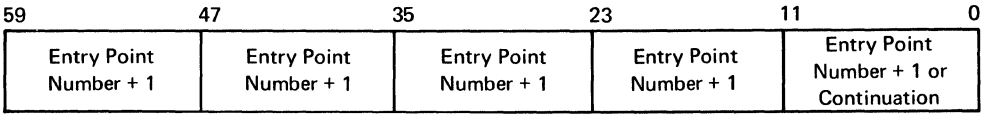
If the routine is disk resident, there is a special disk address in bits 35 through 0 for use by PP resident.

If the routine is central memory or ECS resident, bits 17 through 0 contain the short disk address, which is the PRU offset from the start of the chain. The central memory address is absolute.

Fields not being used contain zero.

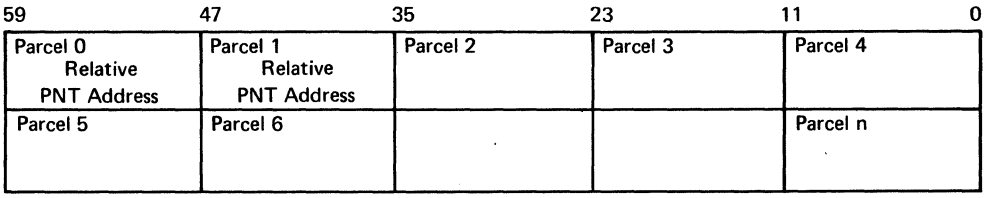
† For ECS resident overlay, this field contains the last consecutive access failure count. The residence is moved to RMS when the failure count exceeds an established threshold.

ERT ENTRY FORMAT



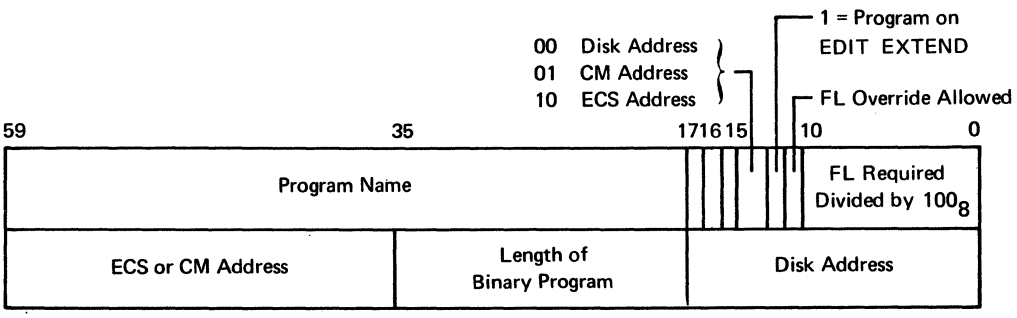
The first entry of this table is associated with the first entry in the entry point name table. The chain of external references is terminated by 12 bits of 0 or by the fifth parcel not having bit 11 set to 1. If bit 11 is set to 1, the value contained in this parcel, less bit 11, is the relative address of the continuation word. The address is relative to the FWA of the external reference table plus the entry point name table. All external calls made directly by a routine are shown in this table.

PNUT ENTRY FORMAT



This table holds pointers relative to the program name table for each entry point number. The entry point number is the parcel number in this table. There are five parcels per word, stored from left to right in the sequential bytes of a word. The value in each parcel is the relative address of the program's entry in the program name table. Address is relative to the start of the program name table.

PNT ENTRY FORMAT



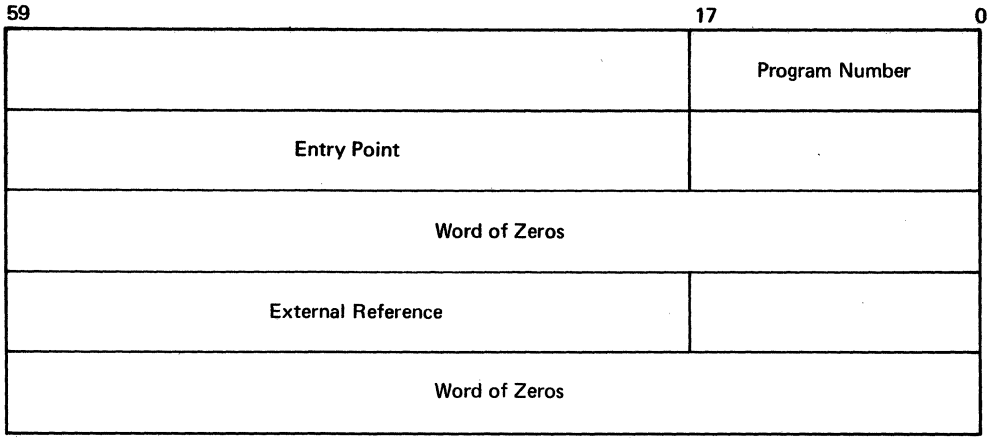
Word 1:

Bits 59-18		Program name in display code.
Bit 17	0	Normal program.
	1	Capsule.
Bit 16	0	Binary program.
	1	CCL procedure.

Bit 15	Not used.	
Bits 14-13	00	Disk address.
	01	CM address.
	10	ECS address.
Bit 12	1	Program is on (pfn) ZZZZZ03.
	0	Program is on (pfn) ZZZZZ04.
Bit 11	INTERCOM field length override bit.	
	0	Do not override.
	1	May be overridden.
Bits 10-0	Contain the amount of field length divided by 100 (octal) to execute this program. If not specified during an EDITLIB run, it is set to DFAULTFL/100 (octal) where DFAULTFL is a program parameter.	

The CM address is relative to the LWA+1 of this table. Fields not being used contain zero.

ENTRY POINT AND EXTERNAL REFERENCE LIST FORMAT



The entry point and external reference list has an entry for each program that is a part of the library. This list is maintained by EDITLIB. The entry point and external references for each routine are in alphabetical order and are left-justified with zero fill.

The index to this record, when the library is not part of the running system, is in the index record for the random file. When the library becomes part of the running system, the index to the entry point and external reference list for the library is destroyed. The entry point and external reference list is placed following the entry point name table record. By using the index to the entry point table, it is possible to access the external reference list. A CIO read skip request is used with the index to the entry point name table. Then a read with a zero index is used to obtain the entry point and external reference list.

Figure 8-2 shows library table interfaces.

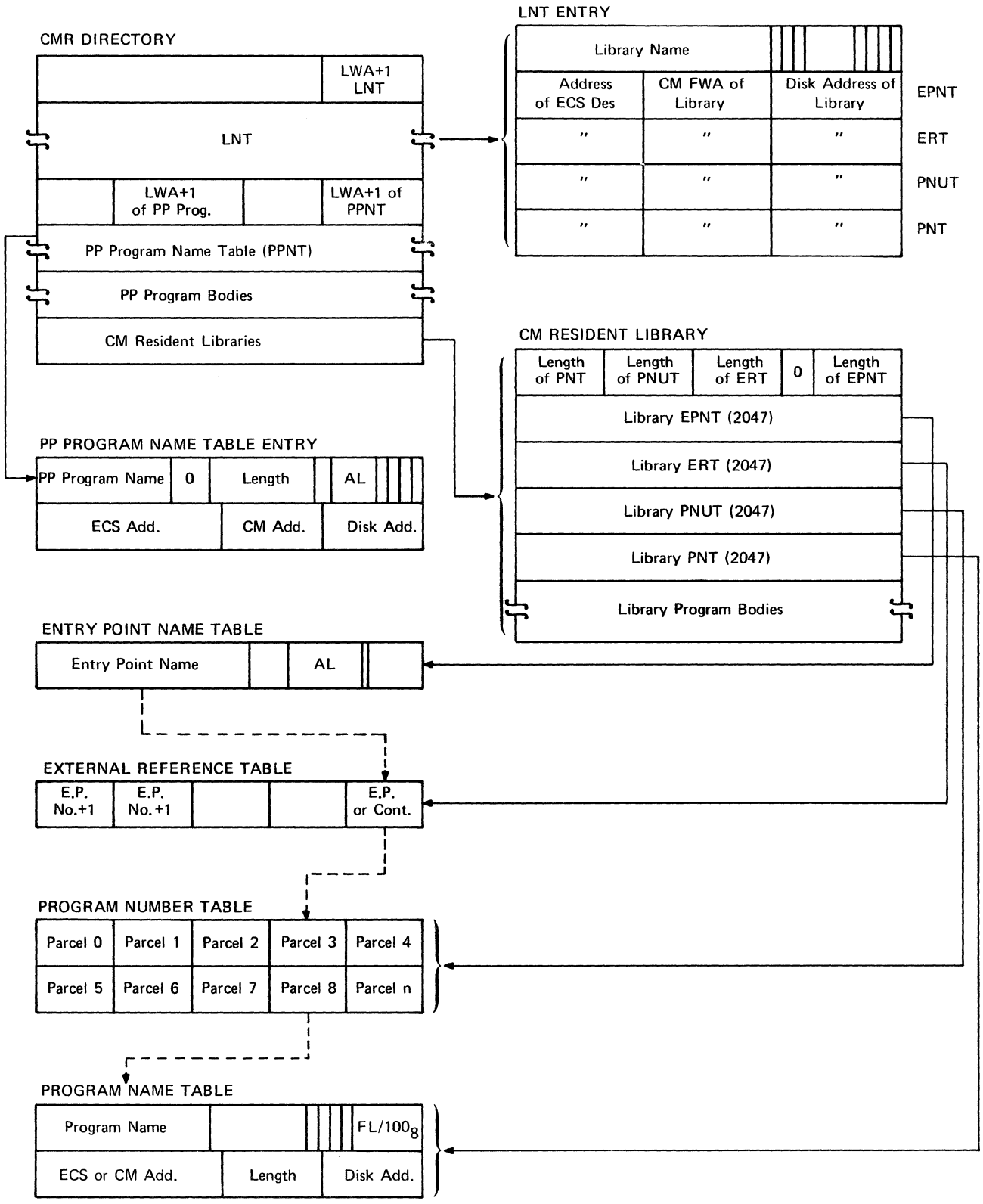


Figure 8-2. Library Table Interfaces

EXAMPLES

Deadstart creation from the running system with changes:

```
READY(NEWSYS,NEW)
TRANS77(IPL+OSB,SYSTEM)
TRANSFER(CED+MDR,SYSTEM)
TRANSFER(CMR,NEWCMR)
SKIPF(1,SYSTEM)
TRANSFER(COM+LFP,SYSTEM)
TRANSFER(0SY,SYSTEM)
TRANSFER(0SZ,SYSTEM)
TRANSFER(0MT,SYSTEM)
TRANSFER(0SJ,SYSTEM)
TRANSFER(*,SYSTEM)
LIBRARY(NUCLEUS,NEW)
ADD(*,SYSTEM,LIB=NUCLEUS,AL=1)
REPLACE(EDITLIB,LGO,AL=1,FL=40000)
REPLACE(LOADER,LGO,AL=1,FL=20000)
FINISH.
LIBRARY(FORTRAN,NEW)
ADD(*,NEWFTN,AL=0,FL=45000)
FINISH.
LISTLIB(*,NEWSYS,NUCLEUS)
LISTLIB(*,NEWSYS,FORTRAN)
ADD(*,SYSTEM,LIB)
REPLACE(1SP,LGO,AL=0,CM)
LISTLIB(*,SYSTEM)
COMPLETE.
LISTLIB(*,SYSTEM,NUCLEUS)
LISTLIB(*,SYSTEM,FORTRAN)
ENDRUN.
```

Deadstart creation from the running system with no changes:

```
READY(NEWSYS)
REWIND(NEWSYS/SYSTEM)
TRANS77(IPL+OSB,SYSTEM)
TRANSFER(CED+MDR,SYSTEM)
TRANSFER(1,SYSTEM)CMR
TRANSFER(COM+LFP,SYSTEM)
TRANSFER(0SY,SYSTEM)
TRANSFER(0SZ,SYSTEM)
TRANSFER(0MT,SYSTEM)
TRANSFER(0SJ,SYSTEM)
TRANSFER(*,SYSTEM)
INCLUDEP(SYSTEM)
INCLUDE(NUCLEUS,SYSTEM,CM)
INCLUDE(SYSOVL,SYSTEM,CM)
INCLUDE(BAMLIB,SYSTEM,DS)
INCLUDE(SYMLIB,SYSTEM,DS)
INCLUDE(FORTRAN,SYSTEM,DS)
    .
    .   INCLUDE directive for all other libraries that are to
    .   be a part of the new system.
    .
INCLUDE(SYSMISC,SYSTEM)
    .
    .   Directives that modify any of the libraries
    .   just included into the new system, or directives
    .   that create new libraries.
    .
COMPLETE.
ENDRUN.
```

System EDITLIB:

```
READY(SYSTEM,OLD)
ADD(*,NEWPP)
LIBRARY(SCOPE1,NEW)
ADD(*,SYSTEM,LIB=NUCLEUS,DS,AL=1,FL=40000,FLO=1)
REPLACE(LOADER,NEW,DS,AL=1,FL=20000)
FINISH.
LISTLIB(*,SYSTEM,NUCLEUS)
COMPLETE.
READY(SYSTEM,OLD)
ADD(1ZZ,LOG,AL=0)
LIBRARY(NUCLEUS,OLD)
DELETE(HOHUM)
FINISH.
COMPLETE.
ENDRUN.
```

SYSTEM BULLETIN UTILITY 9

Installation system bulletins can be created and stored for user access. Bulletins are created and updated with the system utility BULLUP, and are accessed by the user with the control statement SYSBULL. BULLUP has the capability to create a system bulletin file if none exists, update an existing bulletin file, and reduce the size of an existing bulletin file. BULLUP creates and updates the bulletin file using information from data statements. Updating the bulletin file consists of adding new bulletins to the file and/or updating the information in an existing bulletin. When BULLUP creates or updates a bulletin file, it also builds an index so that bulletins can be accessed by name. Reducing the size of the bulletin file consists of reading the active information from the bulletin file and writing it to a new file.

BULLUP STATEMENT

The control statement for calling the BULLUP utility is as follows.

```
BULLUP.
```

or

```
BULLUP(input,output)
```

Default input and output files are INPUT and OUTPUT. Alternate files can be used by specifying alternate file names. The SYSBULL control statement does not have this capability; output from SYSBULL always goes to the file OUTPUT.

BULLUP DATA STATEMENTS

BULLUP accepts data statements in the following format.

```
Name statement
Contents statement
    .                  } First bulletin
    .
    .
Contents statement
Name statement
Contents statement
    .                  } Second bulletin
    .
    .
7/8/9
```

NAME STATEMENTS

Name statements have the following format.

Column	1 2 3 through 9	36 through 80
	* $ bulname	bulletin description

bulname	Bulletin name consisting of one through seven letters and digits; it must begin in column 3. The first blank terminates the field. The reserved names INDEX and ALL are not allowed. Up to 100 names can be specified for the system bulletin file.
bulletin description	Bulletin description consisting of up to 45 characters (generally, a brief description of the bulletin). The description, which is listed with the index, is the second line of the bulletin.

CONTENTS STATEMENTS

Contents statements are any statements that do not have * $ in columns 1 and 2. Column 1 is used either for printer carriage control or as a continuation indicator. If column 1 is not a comma, it is considered a printer carriage control character, and columns 2 through 80 are printed in print line positions 1 through 79. A statement with a comma in column 1 is considered to be a continuation of the preceding statement, and columns 2 through 58 are printed in print line positions 80 through 136. Multiple continuation statements are allowed.

PROCESSING DATA STATEMENTS

The most significant difference between a creation run and an update run is the fact that a bulletin file does not exist before a creation run. Data statements for a creation run are processed in the same way as for an update run.

A name statement is read and validated. If an error occurs in the name statement, succeeding bulletin contents statements are skipped up to the next name statement, and a diagnostic is printed. If the name statement is valid, the bulletin file index is searched for a bulname match. If none is found, the name is added to the end of the index, and the index is changed to reflect the new bulletin. If a matching name is found, the index is changed to reflect the update to this bulletin. BULLUP then fills a buffer with the name statement, contents statement, and other information, including the current date and whether the bulletin was created or updated. The buffer is written to the end of the bulletin file when it becomes full and when a new name statement is encountered. After all data statements have been read, the bulletin file is closed and the bulletin file index is appended to the end of the file.

CREATING A SYSTEM BULLETIN FILE

BULLUP creates a bulletin with the name specified in the bulname position of the name card. An index entry for each bulletin is stored in the bulletin file index. The following control statements are required to create a system bulletin file.

```
job statement
REQUEST(ZZZZZIN,PF)
BULLUP.
CATALOG(ZZZZZIN,SYSTEMBULLETINFILE,ID=SYSBULL,...)
7/8/9
Name statement
Contents statement
    .
    .
    .
6/7/8/9
```

Only the permanent file name and the ID are defined in SYSBULL. Both can be changed; however, if they are changed, the attach function in SYSBULL must also be changed. A turnkey (TK=) and/or a read (RD=) password for the system bulletin file may be added. If so, the attach function in SYSBULL also must be changed to specify the addition. Extend, modify, and control passwords can be specified if desired, without any changes.

UPDATING A SYSTEM BULLETIN FILE

BULLUP updates a bulletin file by adding new bulletins and/or updating information in existing bulletins. When a new bulletin is added to the file, a new index entry is added to the file index. When the information in an existing bulletin is updated, the file index is changed to reflect the fact that the bulletin has been updated.

The following control statements are required to update a system bulletin file.

```
job statement
ATTACH(ZZZZZIN,SYSTEMBULLETINFILE,ID=SYSBULL,...)
BULLUP.
EXTEND(ZZZZZIN)
7/8/9
Name statement
Contents statement
    .
    .
    .
6/7/8/9
```

Permanent file considerations for an update run are the same as for a creation run. Data statement requirements are the same as for creating a new bulletin. The old contents of the bulletin are not carried forward to the updated bulletin. If desired, the contents must be re-entered via contents statements. Information is deleted from an existing bulletin by using a name statement and omitting succeeding contents statements. Since disk space for this bulletin still exists, even though it contains no information, a single contents statement stating that the bulletin contains no information is suggested.

REDUCING A SYSTEM BULLETIN FILE

An index entry and its associated bulletin become inactive when that bulletin is updated. BULLUP does not rewrite in place, since the new information in an updated bulletin may exceed the old information. To maintain minimum disk space for the system bulletin file, BULLUP copies the active bulletins and the current index to a new file, which can then be cataloged.

The following control statements are required to reduce the size of the system bulletin file.

```
job statement
REQUEST(ZZZZZOU,PF)
ATTACH(ZZZZZIN,SYSTEMBULLETINFILE,ID=SYSBULL,...)
BULLUP.
CATALOG(ZZZZZOU,SYSTEMBULLETINFILE,ID=SYSBULL,...)
PURGE(ZZZZZIN)
6/7/8/9
```

ZZZZZOU is the new bulletin file. It must be requested for permanent file device residence before BULLUP is called and it must be an empty file. ZZZZZIN is the current bulletin file with active and inactive information. Data statements should not be used when the active bulletins and the current index are to be copied to a new file. If a system bulletin file backup is desired, PURGE(ZZZZZIN) need not be used.

NOS/BE-INTERCOM CONSIDERATIONS

Under INTERCOM, a call to SYSBULL is initiated when a user logs in. SYSBULL attempts to find the specific bulletin named LOGIN (if SUP was not specified during login procedures) or SUP. If SYSBULL finds the appropriate bulletin, information is immediately displayed at the terminal. Bulletins named LOGIN and SUP are not mandatory on the system bulletin file. If SYSBULL does not find the system bulletin file or the specific bulletin LOGIN or SUP, processing continues. However, installations with INTERCOM users can keep their users informed via these named bulletins since INTERCOM automatically initiates an attempt to find them.

A call to SYSBULL is initiated automatically when a batch job enters the system. SYSBULL attempts to find the specific bulletin named BATCH. If found, the bulletin information is sent to OUTPUT immediately, and processing continues. The bulletin named BATCH is not mandatory on the system bulletin file. If SYSBULL does not find the system bulletin file or the specific bulletin BATCH, processing continues. However, installations can keep their users informed via BATCH, since the system automatically attempts to find the BATCH bulletin.

INTRODUCTION

The purpose of segmenting CMR is to move operating system central processor code from central memory to ECS in systems having ECS. The code is loaded back to CM and executed as needed. This reduces the amount of CM required for the operating system and releases that CM to user jobs running at control points.

Because of CMR segmentation, the system CP code executes in two different configurations, depending on ECS availability. When ECS is not available, the system CP code is all located in CM. When ECS is available, most of the CP code is ECS resident and loaded as needed in overlay areas. The CMR loader, LDCMR, handles the segment relocation and the creation of the ECS segment library. LDCMR is called by deadstart and can also run as a utility. If P.AREA is changed, LDCMR must be reassembled.

The default library used by LDCMR is CMRLIB. An alternate library can be specified in T.ENTRY+W.SGLIB. This library should contain all CMR segments and routines used by these segments. Two other programs are used by LDCMR. Program LDCMR= is loaded by CYBER Loader to reload LDCMR; this program must be a relocatable binary. Program CMRDIR describes the system to be generated and contains the trace and stack buffer lengths, as well as segment names needed for a load. CMRDIR also designates areas where the segments are to be loaded.

LDCMR uses direct access ECS by requesting control point 0 ECS FL. Therefore, the length of the direct access area changes if LDCMR is used to add ECS systems.

LDCMR catalogs a file called ZZZZCMR with ID=SYSTEM, TK=ECSSYSTEM, MD=XNOX, and EX=XNOX. ZZZZCMR has two records: the first record contains a local area table followed by a segment table; the second record contains all the completed segments. On a level 1 deadstart, LDCMR uses the lowest cycle of ZZZZCMR to rebuild an ECS system if the date-time stamp matches the one in T.ENTRY+W.DTIME.

If the library directory is moved, LDCMR will attach ZZZZZ02 to lock out EDITLIB operations. On completion of the move, ZZZZZ23, ZZZZZ01, and ZZZZZ02 are modified to reflect the current directory.

If S.SYSEDT is set in P.LIB, LDCMR does not need operator permission to add a system to ECS or to bootstrap an ECS system. The bit is cleared by a successful bootstrap.

LDCMR CONTROL STATEMENT

The LDCMR control statement has the following format.

LDCMR(p_1,p_2,...,p_n)

Parameters are order independent.

p_i	Description
F=lfn_1	If PO=P is not selected, lfn_1 contains CMR segment relocatable binaries and/or CMRDIR. When segment binaries needed to build a CMR are not on lfn_1, they are taken from library CMRLIB. If two or more copies of the same segment exist on lfn_1, the last one encountered is used. If a deck called CMR is present in the input, all information needed by LDCMR is taken from this CMR. If P.AREA is changed, LDCMR must be reassembled. If PO=P is selected, lfn_1 contains the same information as ZZZZCMR (area table, segment table, and CMR without the low core tables).
F	Use file LGO for input file.
F=0	No load. The B, L, and LO parameters are ignored. PO options A and T are ignored.
F omitted	No input file.
B=lfn_2	LDCMR writes the system image to lfn_2. This file can be used later to create a new system (refer to PO options).
B	LDCMR writes the system image to file CMR.
B=0 or B omitted	No system image file is generated.
L=lfn_3	All output goes to lfn_3.
L or L omitted	All ouput goes to file OUTPUT.
LO=option	The letter or letters selected specify the contents of the map. Any of the options S, B, E, X, and L can be combined; for example, LDCMR(LO=SBL). The N option can be combined only with the L option; for example, LDCMR(LO=LN). If none of the CYBER Loader map options (N, S, B, E, or X) are selected or an error is detected in processing the parameter, the current job default is used.

option	Map Contents
S	CYBER Loader statistics.
B	CYBER Loader block map.
E	CYBER Loader entry point map.
X	CYBER Loader entry point cross references.
L	LDCMR map.
N	No CYBER Loader maps are generated.

LO	Equivalent to LO=SBEXL.
LO omitted	Equivalent to LO=L.

Pi	Description
PO=option	The letter or letters selected specify actions to be performed; order of execution is IPTRDCAB.

Option	Action
A	Add a new CMR to ECS. If the old system is not running, LDCMR bootstraps the old system first.
B	Bootstrap alternate system and move library if necessary. This option requires operator permission.
C	Catalog a new cycle of ZZZZCMR with the lowest cycle.
D	Delete new system from ECS. If the new system is running, LDCMR bootstraps the old system before releasing ECS. Option D is completed before option A, so PO=DA has the same effect as PO=A.
I	Initialize system. This option is used only at deadstart time to examine deadstart options and perform the indicated action. This option overrides all other PO options.
P	Input file specified by F parameter has been preprocessed by LDCMR and is in the same format as the ZZZZCMR file.
R	Replace old system in ECS with the new system. If the old system is running, LDCMR bootstraps the new system before releasing ECS space taken by the old system. The bootstrap moves the ECS copy.
T	Trace all GOTOs. Do not change a GOTO to an entry point in CM resident to a direct jump.

PO	Options ABC are used.
PO omitted	No options are used.

LDCMR FILES

File Name	Cataloged	Use
ZZZZZ03	No	File from which system library binaries are read. CMR library directory may be on this file.
ZZZZZ04	No	Same as ZZZZZ03.
ZZZZZL0	No	CYBER Loader input file containing CM resident binaries.
ZZZZZL1	No	CYBER Loader input file containing the monitor area binaries.
ZZZZZL2	No	CYBER Loader input file containing the user area binaries.

File Name	Cataloged	Use
ZZZZZL3 ZZZZZL4 ZZZZZL5 ZZZZZL6	No	Reserved.
ZZZZZL7	No	File containing LDCMR= and a dummy (0,0) overlay. This forces CYBER Loader to load at the correct address and return to LDCMR when loading completes.
ZZZZZL8	No	CYBER Loader output file where all the loaded segments are written.
ZZZZZL9	No	LDCMR scratch file used to save its tables while CYBER Loader is running.
ZZZZZ01	Yes	Modify reset file in case directory moved.
ZZZZZ02	Yes	Interlock file to lock EDITLIB out. Modify restore file if directory is moved.
ZZZZZ23	Yes	Modify directory file if LDCMR has moved T.LIB.
ZZZZCMR	Yes	File on which the area table, segment table, and segments are saved so that LDCMR does not need to reload them.

The files that are cataloged have the following parameter values set.

File Name	ID	RP	TK	CN	MD	EX
ZZZZZ01	SYSTEM	999	SSSSSSSSS	XNOX	XNOX	XNOX
ZZZZZ02	SYSTEM	999	SSSSSSSST	---	---	XNOX
ZZZZZ23	SYSTEM	999	DIRECTORY	XNOX	XNOX	---
ZZZZCMR	SYSTEM	999	ECSSYSTEM	---	XNOX	XNOX

SYSTEM SECURITY

LDCMR uses the reprieve function to protect the system files and its scratch files. Under normal or abnormal termination, LDCMR returns or releases all LDCMR files.

RESERVED NAMES

LDCMR uses the following names which should not be used by the operating system:

DUMMYnn

nn A number from 00_8 to 77_8.

The names DUMMYnn are entry point names which must not be defined in any segment.

LDCMR also uses the common block name of CMRES to relocate the load address for each area. At the present time, common blocks are not permitted within segments.

LDCMR INTERLOCK

When LDCMR changes the ECS system, LDC is called to assign the ECS. LDC assigns the block of ECS under control point zero in the direct access area.

LDC uses a bit in the control point area to keep CPMTR from changing the ECS FL and ECS RA. This prevents the scheduler from changing LDCMR's exchange package while LDCMR is running. This bit is cleared when LDCMR signals LDC that it has completed the change. The method used to set/clear this bit is as follows:

1. Set/clear bit in control point area.
2. Wait before reading the word.
3. If MTR has changed the word, try again; otherwise, continue.

This interlock is also used to prevent two LDCMRs from bootstrapping or changing the ECS system at the same time.

LDC

This PP overlay is used to help LDCMR. It can perform the following tasks.

- Read absolute CM.
- Assign or release control point 0 ECS from the direct access area.
- Catalog permanent files with ID=SYSTEM at control point 0.
- Reload LDCMR after CYBER Loader completes loading each stage.
- Bootstrap a new or old ECS system.
- Move the directory.
- Issue all operator GO/DROP messages.

A parameter list is provided in LDCMR for LDC. This list contains the LDC function code and all necessary information. LDC returns information and sets the completion bit within this list.

For an explanation of the directory move process, refer to EDITLIB in section 8. LDC uses the same procedure.

EXAMPLES

To add a new system to ECS and bootstrap the new system, use the following LDCMR statement. The new binaries are on file X. A map will also be generated.

```
LDCMR (PO=AB,F=X)
```

To replace the old system with the new and release the ECS space, use the following statement.

```
LDCMR (F=0,PO=R)
```

To get a load map of the deadstart system, use the following statement.

```
LDCMR.
```

To save the preloaded system for later use, use the following sequence of statements.

```
REQUEST(X,PF)
LDCMR(B=X)
CATALOG(X,PERM,ID=USERA)
```

To load and bootstrap a new system from the preloaded file, use the following statements.

```
ATTACH(X,PERM,ID=USERA)
LDCMR(F=X,PO=PAB)
```

SYSTEM DUMPS 11

This section describes the deadstart dump analyzer and the system dynamic dump.

DEADSTART DUMP ANALYZER

The deadstart dump analyzer processes an express deadstart dump (refer to the NOS/BE Operator's Guide for procedures to take an express deadstart dump) and produces as output the dump in readable format. Analysis directives allow the user to specify portions of CM, PPs/PPUs, and/or ECS to be reformatted and printed. The tape file is in stranger format.

DSDUMP CONTROL STATEMENT

The format of the analyzer control statement, DSDUMP, is

DSDUMP(I=lfn$_1$,O=lfn$_2$,T=lfn$_3$,C=lfn$_4$,Z)

I=lfn$_1$ — File containing analysis directives. Directives are read starting at the current position of the file. If I=0 is specified, the directives A, B, IT,† CM, SR, RBTC, M, PP, H, JDT, XP, and WO are assumed. Default is INPUT.

O=lfn$_2$ — File to which output is written. Default is OUTPUT.

T=lfn$_3$ — File containing express deadstart dump (EDD). This file cannot be processed in its original format; the analyzer must reformat it. To save time in subsequent runs using the same file, the resultant file (C=lfn$_4$) can be saved. If the T=lfn$_3$ parameter is omitted, it is assumed that the EDD file has already been reformatted. The reformatting phase is skipped and the file specified by C=lfn$_4$ is used as input. If the keyword T is specified alone, lfn$_3$ is assumed to be TAPE.

C=lfn$_4$ — Reformatted dump file (T=lfn$_3$). Default is CRASH.

Z — Directives are on the DSDUMP control statement. If I=lfn is also specified, directives on the control statement take precedence. Directives can start after the first right parenthesis or period on the control statement; they terminate at the next period encountered. Directives can be continued; if a period is not found on a control statement and column 80 is not blank, the next statement is used as input starting in column 1. If an end-of-record is encountered, reading of directives ends and all legal directives that have been read are processed.

† If INTERCOM is in the system.

ANALYSIS DIRECTIVES

Analysis directives can appear on an input file (specified by the I=lfn$_1$ parameter) or on the DSDUMP control statement (specified by the Z parameter). Directives are separated by commas and terminated by a period or an end-of-record. In the following description, directives are separated into three groups: display directives, dump directives, and special dump directives (such as those used for pattern matching).

Display Directives

The display directives generate tables that are expanded forms of console displays.

Directive	Display	Description of Table Generated
A	A	Contains the most recent system dayfile and dayfiles for all active control points.
B	B	Indicates activity at each control point, channel and PP allocation, JDT address, and so on.
H	F, H	Contains the file name table. The octal representations of the FNT entries are printed but supplements are not.
JDT	R	Contains the job descriptor table showing all jobs in the system.
M	M	Contains the current status of all PPs.
TAPES	P	Contains the entire octal presentation of the TAPES table for every drive in the EST.

Dump Directives

The dump directives generate dumps of specified area. Dumps are in octal, unless otherwise noted in the description.

Directive	Description
BC	Dumps (in hexadecimal) all buffer controllers that were dumped to tape by EDD. Dumps are identified by channel number.
Cx	Dumps central memory from address 0 to the first word address of the CM resident library table.†
	x Specifies number of octal words printed per line; 0 through 4 or M. The values 0, 4, and M specify four words per line.

† In CM and ECS dumps, lines of all zeros are omitted. The next nonzero line contains the symbol ↦ after the address. Also omitted are duplicate lines and lines which are the same, word for word, as the last word of the previous line. The next nonduplicate line contains an equals sign after the address.

Directive	Description
Cx=memstring$_1$/.../memstring$_n$	Dumps specified CM and optionally prints a message at the beginning of the dump.†

x — Specifies number of octal words printed per line; 0 through 4 or M. The values 0, 4, and M specify four words per line.

memstring$_i$ — Specifies either the range of CM to be dumped for the range plus a delimited character string. The range of memory to be dumped is specified in the form

fwa-lwa

where fwa is the first word address of the dump and lwa is the last word address of the dump. A message of up to 50 characters can immediately follow the range. It is printed at the beginning of the dump. The character string must be delimited by one of the special characters + * = ; - $ or %. Both delimiters must be the same; the delimiter cannot appear in the string.

Examples:

```
C2=200-1000
C3=3700-4700*DUMP NO. 1*
CM=200-600$FIRST DUMP$/3000-
4000$SECOND DUMP$
```

The values of fwa and lwa are assumed to be octal, unless followed by D specifying decimal.

If lwa is less than fwa, lwa is reset to fwa + 100_8. If lwa is greater than the memory size, it is reset to the memory size. Memory is read in blocks of 100_8 and therefore some shifting of fwa and lwa may occur. If shifting causes lwa to be greater than memory size, the nonexistent words are printed with the display code equivalent of 10 asterisks.

† In CM and ECS dumps, lines of all zeros are omitted. The next nonzero line contains the symbol ⌐→ after the address. Also omitted are duplicate lines and lines which are the same, word for word, as the last word of the previous line. The next nonduplicate line contains an equals sign after the address.

Directive	Description
Ex=memstring$_1$/.../memstring$_n$	Dumps specified ECS and optionally prints a message at the beginning of the dump.† The parameters for this directive are identical to those for the Cx=memstring directive. Refer to that directive for a description of parameters and their usage.
PP	Dumps all PPs in a format of 16 bytes per line.
PP=pp$_1$/.../pp$_n$	Dumps specified PPs or PPUs in a format of 16 bytes per line. pp$_i$ Specifies PP or PPU to be dumped or a range of PPs or PPUs (in the form pp$_m$-pp$_n$) to be dumped. For example, PP=3-7/10/17 dumps PPs 3, 4, 5, 6, 7, 10, and 17. To specify a first level peripheral processor (PPU), the number is preceded by F. For example, PP=F5 dumps PPU 5. PP numbers are assumed to be decimal; PPU numbers are assumed to be octal. Either can have a post-radix of B or D. PPs/PPUs are dumped and printed in the order they were specified. When PPs other than PP0 or PP1 are printed, three lines of PP resident entry point information are printed in addition. The first line is the entry point address. The second line is the entry point name. The third line is the contents of the memory cell at entry point+1, which reflects the return address of the last return jump to this entry point. No attempt is made to verify that the return address is valid. If a specified PP/PPU is not on the dump file, an error message is generated and the analyzer continues. PP memory cells destroyed by the deadstart process are printed as asterisks.
RBR	Dumps all RBRs.
RBR=n	Dumps record block reservation table. n Number of RBRs to be dumped.

† In CM and ECS dumps, lines of all zeros are omitted. The next nonzero line contains the symbol ↦ after the address. Also omitted are duplicate lines and lines which are the same, word for word, as the last word of the previous line. The next nonduplicate line contains an equals sign after the address.

Directive	Description
RBTC	Validates RBT chains and prints chains and RBT area in a format of two words per line.
UFL	Dumps all user field lengths.
UFL=cp_1/.../cp_n	Dumps user field length at specified control point. cp_i Specifies control point or range of control points (in the form cp_m-cp_n) for which field length is to be dumped. For example, UFL=2/5-7/10 dumps control points 2, 5, 6, 7, and 10.
XP	Dumps the system exchange package, including the status control register.

Special Dump Directives

Directive	Description
IT	If INTERCOM was running at the time of the deadstart dump, this directive dumps the INTERCOM area, tables, buffers, and so on. The dump is printed in the format of four words per line.
PAT=pattern$_1$/.../pattern$_n$	Searches CM, PP memory, and ECS (if it is available) for a word that matches the specified pattern. If a match is found, the type of memory and address are printed. pattern$_i$ Specifies the word to be matched. Patterns are assumed to be octal unless they are followed by D specifying decimal. Up to 50 patterns can be tested.
SEG	If the system was using ECS at the time of the deadstart dump, this directive dumps the segment name and entry point tables in addition to the monitor and user mode segment names. The CM resident, monitor, and user areas are also dumped in the format of four words per line.
SEQ	Dumps the sequencer tables if they are present.

Directive	Description
SR	Dumps stack requests. Stack requests are listed in the following order: 1. By DST/DAT. If all words of the DST/DAT are zero, it is not printed. 2. By DOT. If the DOT is zero, it is not printed. 3. Empty chain, if it exists. 4. Any stack requests that have not yet appeared in the preceding lists.
WO	Searches CM, PP memory, and ECS (if it is available) for a word that matches word 0 of central memory. If word 0 is zero, no match is attempted.

Default Directives

If no directives were found on the DSDUMP control statement or input file, or if I=0 is specified, the following directives are assumed:

A, B, IT,† CM, SR, RBTC, M, PP, H, JDT, XP, WO

DSDUMP MESSAGES

The deadstart dump analyzer issues informative messages indicating the current status of the analysis. The dayfile message

DSDUMP - - ANALYZING CRASH FILE.

is issued after the dump tape is read or as the header of the CRASH file ($C=lfn_4$) is being read. When all directives have been processed, the analyzer issues the dayfile message

DSDUMP - - NORMAL TERMINATION.

If INTERCOM analysis is requested and INTERCOM was running when the deadstart dump was produced, the message

INTRAN - - BEGINNING INTERCOM ANALYSIS.

is issued when INTERCOM analysis begins. When INTERCOM analysis is concluded, the message

INTRAN - - END INTERCOM ANALYSIS.

is issued.

† If INTERCOM is in the system.

If one or more errors occur in the directive, the dayfile message

DSDUMP - - ERROR(S) IN DIRECTIVES.

is issued. Except for the message, errors are ignored and processing continues.

The dump analyzer uses CMR pointers to find many of the tables it formats. If a pointer appears to be incorrect, a message to that effect is printed in the analysis output. If the CMR tables are overwritten, the message

DSDUMP - - LOW CORE WAS OVERWRITTEN.

is issued. If the values in low core are no longer valid, the analyzer uses the following default values.

CM size	777700_8
Number of PPs	20
Number of control points	17_8

SYSTEM DYNAMIC DUMP

The system dynamic dump facility takes an on-line dump of PP or central memory and lists the dump on printed output. This facility is currently used by INTERCOM and EXPORT to document the PP and CM contents at the time that a detected error forces either an INTERCOM restart or an EXPORT abort.

INTERFACE

All dumps are written, directly by stack request, to the system file ZZZZZDD (the system dynamic dump file). ZZZZZDD is initialized and recovered by the operating system.

ZZZZZDD is cataloged and purged as a queue file with special disposition code 66_8. When not in use, the file is unlocked at control point zero. During a dynamic dump, the file is locked by the program requesting the dump. While it is being listed, the file is attached to a control point. The file is associated with a file supplement table in the FNT which includes special link ID 602_8. Current RMS position is kept in the supplement.

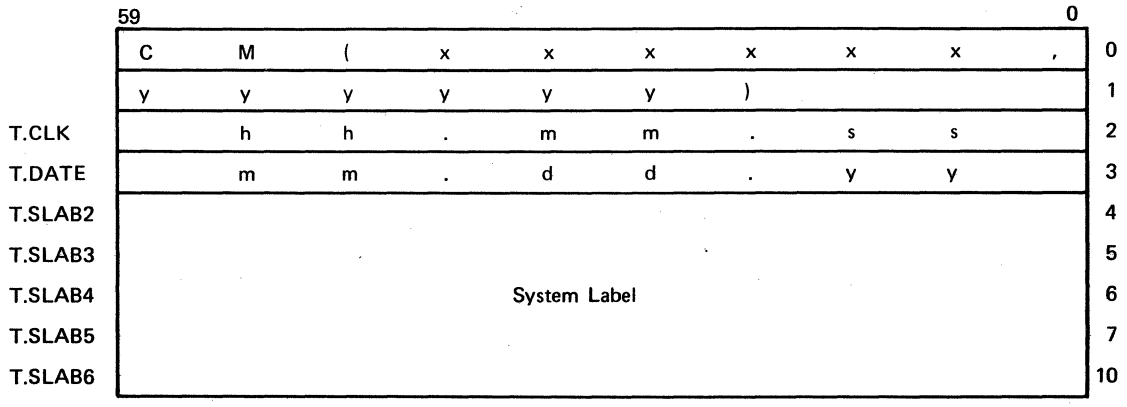
PP memory dumps are written directly from the PP using macros defined on system text SDDTEXT to establish the interface for access to the file. The PP service program 1DD performs CM dumps as well as the cataloging, extending, and purging of the ZZZZZDD file. Text SDDTEXT includes macros that provide a standard interface to call 1DD.

DUMP FILE FORMAT

Each memory dump request is written as a separate record on the system dynamic dump file.

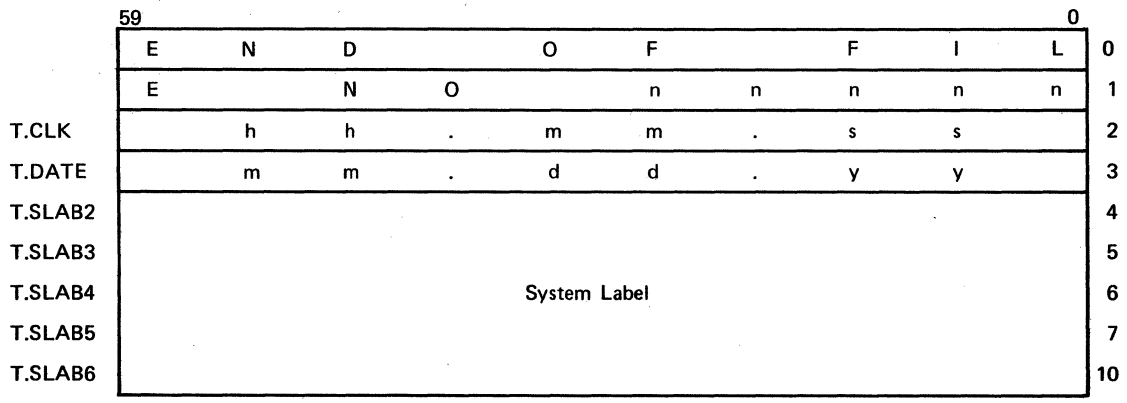
PP dumps are unlabeled and are written as single records with end-of-record level 0.

CM dumps consist of a label followed by a contiguous block of CM image. The label and the CM image dump are separate records, each with end-of-record level 15_8. The label format is as follows.



All entries are in display code. Words 0 and 1 of the label define the CM address bounds of the dump that follows. Words 2 through 10 are taken from CMR at the time of the dump.

At the end of the dynamic dump, a trailer label is written in the following format.



All entries are in display code. Word 1 contains the dump number referenced by LISTCID. Words 2 through 10 correspond to the same words in the CM dump label.

Each dynamic dump ends with an end-of-file. Multiple files may exist on a dynamic dump file. The F parameter on the LISTCID control statement allows the user to select which of these multiple files to read.

LISTING DUMP FILES

LISTCID (list core-image dump) is a system utility that lists output of all or selected portions of a core-image dump file depending on specified parameters. It can be called by a control statement in a batch job or from the central site console. When LISTCID is initiated from a batch job, it requests that the central site operator enter a LOCKIN or DROP command to provide security for access to absolute system dumps.

The LISTCID control statement has the following format.

LISTCID(p_1,p_2,...,p_n)

where p can be a keyword or a keyword equated to an option. Parameters can be specified in any order.

p_i	Description
I=lfn_1	Core-image input is taken from file lfn_1.
I or I omitted	Core-image input is taken from the system dynamic dump file, ZZZZZDD.
O=lfn_2	Listable output is placed on file lfn_2. Disposition of lfn_2 must be provided by the calling job.
O or O omitted	Listable output is placed on file OUTPUT and in the output queue at end-of-job (unless otherwise disposed by the calling job).
C	List all CM dumps encountered in byte format.†
C=addr	List CM from address 0 to address addr in byte format.†
C=$addr_1$-$addr_2$	List CM from address $addr_1$ to address $addr_2$ in byte format.†
C omitted	No specific CM dump.
CD	List all CM dumps encountered in display code format.†
CD=addr	List CM from address 0 to address addr in display code format.†
CD=$addr_1$-$addr_2$	List CM from address $addr_1$ to address $addr_2$ in display code format.†
CD omitted	No specific CM dump.
P	List available dumped PPs in byte format.
F=n	Read file number n. (First file is number 1.)
F=n_1-n_2	Read file numbers n_1 through n_2.
F or F omitted	Read all files.

† If no list control arguments (C or CD) are specified, all of the core-image input file will be listed in display code format. If any of these arguments are specified, each record on the core-image input file is examined and listed according to specified list control arguments.

Numeric values can be octal or decimal. The character B or D immediately following the number specifies octal or decimal. If not specified, addresses are assumed to be octal and file numbers are assumed to be octal.

When the dump input file is ZZZZZDD and it is called from a batch job, LISTCID issues the following message:

SYSTEM DUMP LIST REQUESTED/LOCKIN OR DROP.

The central site operator must enter the n.LOCKIN command to continue the LISTCID operation.

Examples:

To list the contents of the system dynamic dump file from the central site console, the operator must select a clear control point and enter

n.X LISTCID.

To list all PP dumps and any CM locations between 0 and 10000_8, a batch job could contain the control statement:

LISTCID (P,C=0-10000)

ERROR MESSAGES

The error messages issued by LISTCID are explained in the NOS/BE Diagnostic Handbook.

DOWNLINE LOAD UTILITY 12

The downline load utility reloads controlware when the system is running in engineering mode. The system procedure BCPROC, the CP program BCLOAD, and the PP program DLL comprise the downline load utility. Parameters passed to the procedure specify the controller and source of the controlware binary file to be loaded. The controlware can reside on tape, on a sequential mass storage file, on file INPUT, or on the system file ZZZZZ04. Controllers that can be downline loaded are the 7021, 7054, 7152, 7154, and 7155 controllers.

NOTE

Since the autoload function clears all reserved units on the channel for the access, reloading controlware during normal system operation can create broken interlocks. Units reserved by the alternate access are not affected.

BCPROC

BCPROC can be called by the operator from the console or with the BCPROC control statement in a batch job. The format is as follows:

BCPROC,cc,cw,F=ff

Parameter cc must be specified. Parmeters cw and F=ff are optional and have default values.

cc Two-digit octal channel number.

cw Source of controlware binary file. Default is SYSTEM.

cw	Description
BCFILE	Name of the local file attached by the user before calling BCPROC. This can be a tape file or a sequential mass storage file.
INPUT	System file INPUT (valid only when BCPROC is called from a batch job).
SYSTEM	System file ZZZZZ04.

Controller	Deckname
7021	0MT
7054	0SY
7154	0SZ
7155	0SJ

ff Two-digit octal number of files to skip in positioning to the appropriate controlware binary file on BCFILE. This parameter is ignored if cw is INPUT or SYSTEM. The default is zero.

BCPROC generates the data files DATA1, DATA2, and DATA3. These data files contain directives for reading controlware from BCFILE, INPUT, or ZZZZZ04, respectively. BCPROC determines the appropriate file to use and passes the file name as a parameter to BCLOAD.

BCPROC generates the following directives based on parameters the user specified.

ADDRESS30=chff

ch Channel number.

ff Number of files to skip.

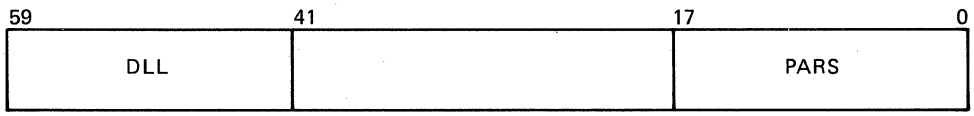
ADDRESS55=00x0

x Controlware source file.

x	Name
1	BCFILE
2	INPUT
3	SYSTEM

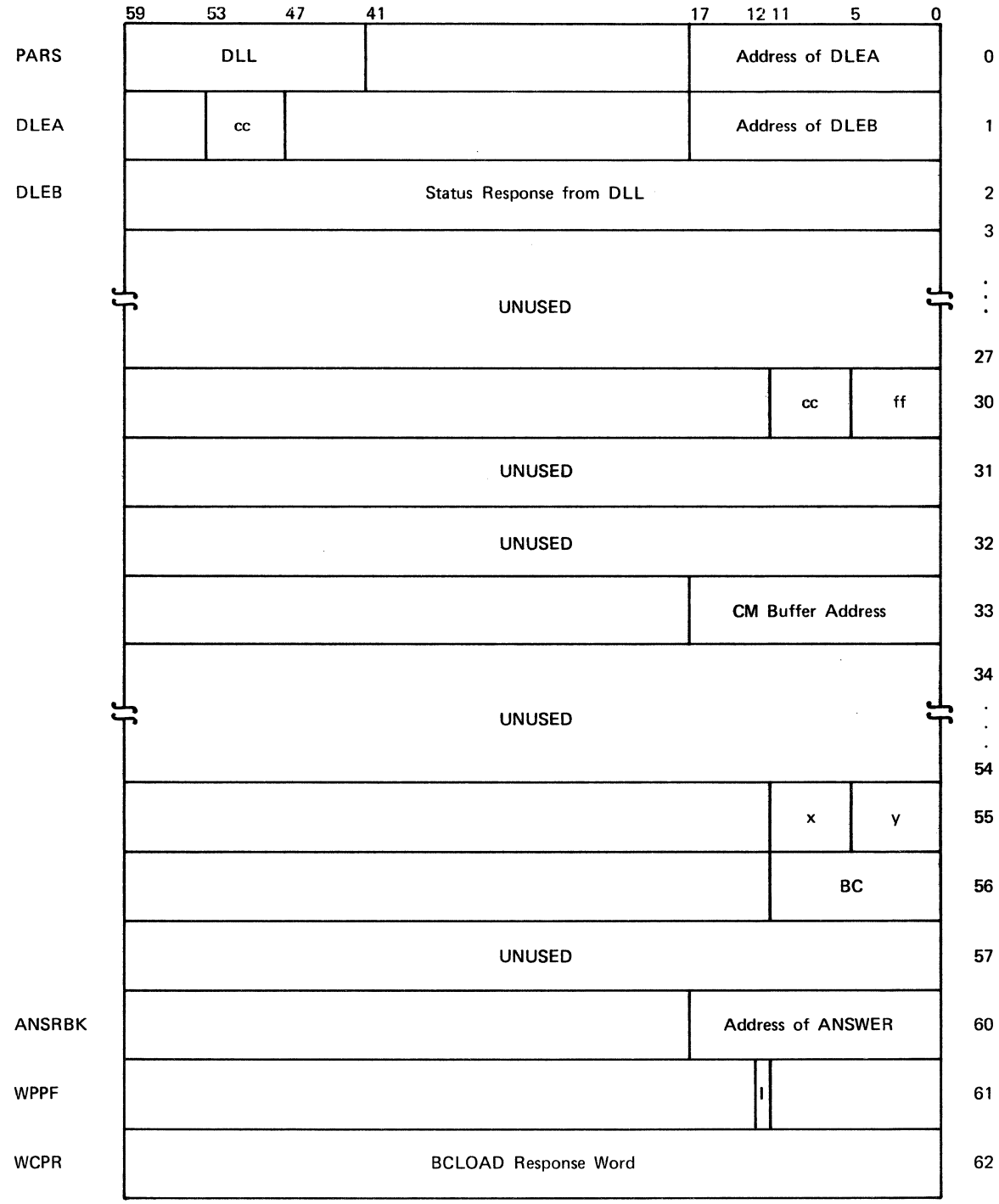
BCLOAD

BCLOAD is a CP program that reads input directives from the procedure data file and passes the information to the PP program DLL. Then BCLOAD reads the controlware from the specified binary file and verifies that the controller can be downline loaded. BCLOAD calls DLL into execution and then goes into a recall state waiting for a response from the PP. BCLOAD and DLL communicate through a parameter buffer area in BCLOAD. The address of this area is passed to DLL in the initial RA+1 request.



PARS is the address of the parameter buffer area.

The format of the parameter buffer area is:



Description of parameter buffer area:

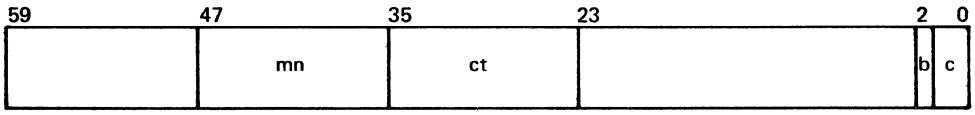
Word	Bits	Field Name	Description
0			Contains the RA+1 request. Bytes 0 and 1 contain the display code for DLL left justified. Bytes 3 and 4 contain the address of DLEA right justified.
1			Bits 53 through 48 contain the channel number cc. Bytes 3 and 4 contain the address of DLEB.
2			Status response from DLL.
30	11-6	cc	Channel number.
	5-0	ff	Number of files to skip.
33	17-0	CM Buffer Address	Address of buffer containing the controlware binary file.
55	11-6	x	File containing controlware. 1 BCFILE 2 INPUT 4 SYSTEM
	5-0	y	Controller type. 1 7054 2 7154 3 7155 4 7021
56	11-0	BC	Display code of characters BC.
60			Bits 17-0 contain the address pointer to PP answer area.
61			PP communication word. If bit 12 is set (I), BCLOAD is waiting for DLL to complete processing. Otherwise, bit 12 contains status codes from DLL.
62			BCLOAD response word; set to nonzero when BCLOAD processes status.

BCLOAD checks the contents of the status words returned by DLL. BCLOAD displays operator action messages and writes error data on the output file when fatal errors occur.

DLL returns status information to the following words in BCLOAD: DLEB, WPPF, and ANSWER.

DLEB

The format of DLEB is:



mn Hardware mnemonic.

(AY, AZ, AJ, MT, or NT)

ct Controller type.

1 7054

2 7154

3 7155

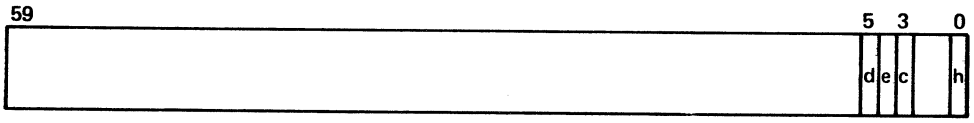
4 7021

b If set to 1, the specified channel number is not in EST.

c Complete bit.

WPPF

The format of WPPF is :



d Operator action required if nonzero.

1 Coupler reserved.

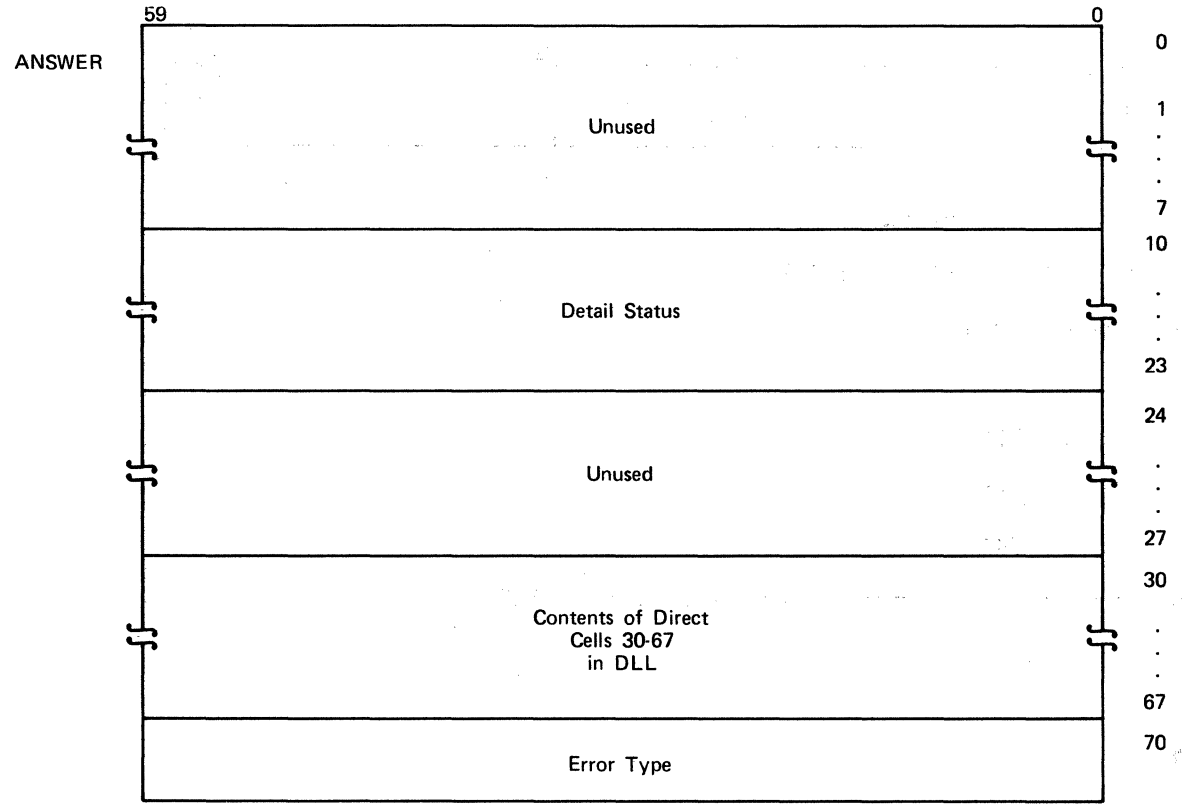
e PP error report. If set, additional status information is returned in the ANSWER buffer.

c Complete bit.

h Recall bit.

ANSWER BUFFER

The format of the ANSWER buffer is:

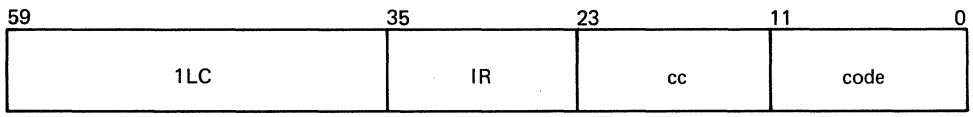


Word 70 Error type.

77	No error; load complete.
Other	Downline load error.

DLL

DLL is a PP program that does the actual downline loading. DLL aborts if the system is not in engineering mode. DLL verifies that the channel number specified is defined in the EST and is a 7021, 7054, 7152, or 7155 controller and returns this information to status word DLEB. DLL checks for coupler reserved errors and returns status to WPPF if such an error occurs. When reloading controlware for the 7021 controller, DLL calls 1LC to load conversion tables. The format of the request to load conversion tables is:



IR Input register address of DLL.

cc Channel number.

code If 6, load conversion tables code number.

Errors occurring in 1LC or DLL during the downline load process are fatal. Status is returned to WPPF. Detail status and the contents of direct cells in DLL are returned to the ANSWER buffer in BCLOAD.

BUILDING A CONTROLWARE FILE

Use the following example to build a controlware file for use with BCPROC. In this example, the controlware records are copied to tape and then to a mass storage file.

```
job statement
REQUEST(BCFILE,....)
COPYBR(INPUT,DISK1)
COPYBR(INPUT,DISK2)
COPYBR(INPUT,DISK3)
COPYBR(INPUT,DISK4)
REWIND(BCFILE)
REWIND(DISK1,DISK2,DISK3,DISK4)
COPYBF(DISK1,BCFILE)
COPYBF(DISK2,BCFILE)
COPYBF(DISK3,BCFILE)
COPYBF(DISK4,BCFILE)
REWIND(BCFILE)
REQUEST(CTLWARE,*PF)
COPYBF(BCFILE,CTLWARE,4)
CATALOG(CTLWARE,...)
7/8/9
        Coldstart binary deck       MB434 - A14       7021 controller
7/8/9
        Coldstart binary deck       MA710 - A13       7054 controller
7/8/9
        Coldstart binary deck       MA401 - A05       7154 controller
7/8/9
        Coldstart binary deck       MA721 - A02       7155 controller
7/8/9
6/7/8/9
```

EXAMPLES

BCPROC can be called after the operator turns on engineering mode.

1. This example shows how to bring BCPROC to a clear control point to load controlware from the system file.

   ```
   n.X BCPROC,07.
   ```

   or

   ```
   n.X BCPROC,07,SYSTEM.
   ```

2. In this example BCPROC is called from a batch job to load controlware for a 7054 controller from any file.

   a. JOB1 loads the controlware contained on the system file.

```
JOB1,....
BCPROC,07.
7/8/9
6/7/8/9
```

   b. JOB2 loads the controlware contained on file INPUT.

```
JOB2,....
BCPROC,07,INPUT.
7/8/9
        Coldstart binary deck    MA710
7/8//9
6/7/8/9
```

   c. JOB3 loads the controlware from tape. The 7054 controlware is the second file.

```
JOB3,....
REQUEST(BCFILE,...)
BCPROC,07,BCFILE,F=01.
7/8/9
6/7/8/9
```

   d. JOB4 loads the controlware from a sequential mass storage file. The 7054 controlware is the second file.

```
JOB4,....
ATTACH,BCFILE,CTLWARE,....
BCPROC,07,BCFILE,F=01.
6/7/8/9
```

STANDARD CHARACTER SETS A

Control Data operating systems offer the following variations of a basic character set.

- CDC 64-character set.
- CDC 63-character set.
- ASCII 64-character set.
- ASCII 63-character set.

The set in use at a particular installation is specified when the operating system is installed.

Depending on another installation option, the system assumes an input deck has been punched either in O26 or in O29 mode (regardless of the character set in use). The alternate mode can be specified by a 26 or 29 punched in columns 79 and 80 of the job statement or any 7/8/9 card. The specified mode remains in effect through the end of the job unless it is reset by specification of the alternate mode on a subsequent 7/8/9 card.

Graphic character representation appearing at a terminal or printer depends on the installation character set and the terminal type. Characters shown in the CDC graphic column of the standard character set table are applicable to BCD terminals; ASCII graphic characters are applicable to ASCII-CRT and ASCII-TTY terminals.

STANDARD CHARACTER SETS

CDC Graphic	ASCII Graphic Subset	Display Code	Hollerith Punch (026)	External BCD Code	ASCII Punch (029)	ASCII Code	CDC Graphic	ASCII Graphic Subset	Display Code	Hollerith Punch (026)	External BCD Code	ASCII Punch (029)	ASCII Code
:†	:	00††	8-2	00	8-2	072	6	6	41	6	06	6	066
A	A	01	12-1	61	12-1	101	7	7	42	7	07	7	067
B	B	02	12-2	62	12-2	102	8	8	43	8	10	8	070
C	C	03	12-3	63	12-3	103	9	9	44	9	11	9	071
D	D	04	12-4	64	12-4	104	+	+	45	12	60	12-8-6	053
E	E	05	12-5	65	12-5	105	-	-	46	11	40	11	055
F	F	06	12-6	66	12-6	106	*	*	47	11-8-4	54	11-8-4	052
G	G	07	12-7	67	12-7	107	/	/	50	0-1	21	0-1	057
H	H	10	12-8	70	12-8	110	(	(	51	0-8-4	34	12-8-5	050
I	I	11	12-9	71	12-9	111	)	)	52	12-8-4	74	11-8-5	051
J	J	12	11-1	41	11-1	112	$	$	53	11-8-3	53	11-8-3	044
K	K	13	11-2	42	11-2	113	=	=	54	8-3	13	8-6	075
L	L	14	11-3	43	11-3	114	blank	blank	55	no punch	20	no punch	040
M	M	15	11-4	44	11-4	115	, (comma)	, (comma)	56	0-8-3	33	0-8-3	054
N	N	16	11-5	45	11-5	116	. (period)	. (period)	57	12-8-3	73	12-8-3	056
O	O	17	11-6	46	11-6	117	≡	#	60	0-8-6	36	8-3	043
P	P	20	11-7	47	11-7	120	[	[	61	8-7	17	12-8-2	133
Q	Q	21	11-8	50	11-8	121	]	]	62	0-8-2	32	11-8-2	135
R	R	22	11-9	51	11-9	122	%	%	63††	8-6	16	0-8-4	045
S	S	23	0-2	22	0-2	123	≠	" (quote)	64	8-4	14	8-7	042
T	T	24	0-3	23	0-3	124	→	_ (underline)	65	0-8-5	35	0-8-5	137
U	U	25	0-4	24	0-4	125	∨	!	66	11-0 or 11-8-2†††	52	12-8-7 or 11-0†††	041
V	V	26	0-5	25	0-5	126							
W	W	27	0-6	26	0-6	127	∧	&	67	0-8-7	37	12	046
X	X	30	0-7	27	0-7	130	↑	' (apostrophe)	70	11-8-5	55	8-5	047
Y	Y	31	0-8	30	0-8	131	↓	?	71	11-8-6	56	0-8-7	077
Z	Z	32	0-9	31	0-9	132	<	<	72	12-0 or 12-8-2†††	72	12-8-4 or 12-0†††	074
0	0	33	0	12	0	060							
1	1	34	1	01	1	061	>	>	73	11-8-7	57	0-8-6	076
2	2	35	2	02	2	062	≤	@	74	8-5	15	8-4	100
3	3	36	3	03	3	063	≥	\	75	12-8-5	75	0-8-2	134
4	4	37	4	04	4	064	¬	^ (circumflex)	76	12-8-6	76	11-8-7	136
5	5	40	5	05	5	065	; (semicolon)	; (semicolon)	77	12-8-7	77	11-8-6	073

†Twelve or more zero bits at the end of a 60-bit word are interpreted as end-of-line mark rather than two colons. End-of-line mark is converted to external BCD 1632.

††In installations using a 63-graphic set, display code 00 has no associated graphic or card code; display code 63 is the colon (8-2 punch). The % graphic and related card codes do not exist and translations from ASCII/EBCDIC % yield a blank (55_8).

†††The alternate Hollerith (026) and ASCII (029) punches are accepted for input only.

CDC CHARACTER SET COLLATING SEQUENCE									
Collating Sequence Decimal	Collating Sequence Octal	CDC Graphic	Display Code	External BCD	Collating Sequence Decimal	Collating Sequence Octal	CDC Graphic	Display Code	External BCD
00	00	blank	55	20	32	40	H	10	70
01	01	≤	74	15	33	41	I	11	71
02	02	%	63 †	16 †	34	42	∨	66	52
03	03	[	61	17	35	43	J	12	41
04	04	→	65	35	36	44	K	13	42
05	05	≡	60	36	37	45	L	14	43
06	06	∧	67	37	38	46	M	15	44
07	07	↑	70	55	39	47	N	16	45
08	10	↓	71	56	40	50	O	17	46
09	11	>	73	57	41	51	P	20	47
10	12	≥	75	75	42	52	Q	21	50
11	13	¬	76	76	43	53	R	22	51
12	14	.	57	73	44	54	]	62	32
13	15	)	52	74	45	55	S	23	22
14	16	;	77	77	46	56	T	24	23
15	17	+	45	60	47	57	U	25	24
16	20	$	53	53	48	60	V	26	25
17	21	*	47	54	49	61	W	27	26
18	22	-	46	40	50	62	X	30	27
19	23	/	50	21	51	63	Y	31	30
20	24	,	56	33	52	64	Z	32	31
21	25	(	51	34	53	65	:	00 †	none†
22	26	=	54	13	54	66	0	33	12
23	27	≠	64	14	55	67	1	34	01
24	30	<	72	72	56	70	2	35	02
25	31	A	01	61	57	71	3	36	03
26	32	B	02	62	58	72	4	37	04
27	33	C	03	63	59	73	5	40	05
28	34	D	04	64	60	74	6	41	06
29	35	E	05	65	61	75	7	42	07
30	36	F	06	66	62	76	8	43	10
31	37	G	07	67	63	77	9	44	11

†In installations using the 63-graphic set, the % graphic does not exist. The : graphic is display code 63, External BCD code 16.

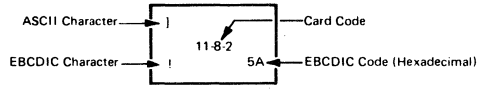
ASCII CHARACTER SET COLLATING SEQUENCE									
Collating Sequence Decimal	Collating Sequence Octal	ASCII Graphic Subset	Display Code	ASCII Code	Collating Sequence Decimal	Collating Sequence Octal	ASCII Graphic Subset	Display Code	ASCII Code
00	00	blank	55	20	32	40	@	74	40
01	01	!	66	21	33	41	A	01	41
02	02	"	64	22	34	42	B	02	42
03	03	#	60	23	35	43	C	03	43
04	04	S	53	24	36	44	D	04	44
05	05	%	63†	25	37	45	E	05	45
06	06	&	67	26	38	46	F	06	46
07	07	'	70	27	39	47	G	07	47
08	10	(	51	28	40	50	H	10	48
09	11	)	52	29	41	51	I	11	49
10	12	*	47	2A	42	52	J	12	4A
11	13	+	45	2B	43	53	K	13	4B
12	14	,	56	2C	44	54	L	14	4C
13	15	-	46	2D	45	55	M	15	4D
14	16	.	57	2E	46	56	N	16	4E
15	17	/	50	2F	47	57	O	17	4F
16	20	0	33	30	48	60	P	20	50
17	21	1	34	31	49	61	Q	21	51
18	22	2	35	32	50	62	R	22	52
19	23	3	36	33	51	63	S	23	53
20	24	4	37	34	52	64	T	24	54
21	25	5	40	35	53	65	U	25	55
22	26	6	41	36	54	66	V	26	56
23	27	7	42	37	55	67	W	27	57
24	30	8	43	38	56	70	X	30	58
25	31	9	44	39	57	71	Y	31	59
26	32	:	00†	3A	58	72	Z	32	5A
27	33	;	77	3B	59	73	[	61	5B
28	34	<	72	3C	60	74	\	75	5C
29	35	=	54	3D	61	75	]	62	5D
30	36	>	73	3E	62	76	^	76	5E
31	37	?	71	3F	63	77	_	65	5F

†In installations using a 63-graphic set, the % graphic does not exist. The : graphic is display code 63.

AMERICAN NATIONAL STANDARD CODE FOR INFORMATION INTERCHANGE (ASCII) WITH PUNCHED CARD CODES AND EBCDIC TRANSLATION

	b_8 b_7 b_6 b_5	0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
b_4 b_3 b_2 b_1	COL / ROW	0	1	2	3	4	5	6	7	8	9	10 (A)	11 (B)	12 (C)	13 (D)	14 (E)	15 (F)
0 0 0 0	0	NUL 12-0-9-8-1 NUL 00	DLE 12-11-9-8-1 DLE 10	SP no-punch SP 40	0 0 0 F0	@ 8-4 @ 7C	P 11-7 P D7	` 8-1 ` 79	p 12-11-7 p 97	11-0-9-8-1 DS 20	12-11-0-9-8-1 30	12-0-9-1 41	12-11-9-8 58	12-11-0-9-6 76	12-11-8-7 9F	12-11-0-8 B8	12-11-9-8-4 DC
0 0 0 1	1	SOH 12-9-1 SOH 01	DC1 11-9-1 DC1 11	! 12-8-7 ! 4F	1 1 1 F1	A 12-1 A C1	Q 11-8 Q D8	a 12-0-1 a 81	q 12-11-8 q 98	0-9-1 SOS 21	9-1 31	12-0-9-2 42	11-8-1 59	12-11-0-9-7 77	11-0-8-1 A0	12-11-0-9 B9	12-11-9-8-5 DD
0 0 1 0	2	STX 12-9-2 STX 02	DC2 11-9-2 DC2 12	" 8-7 " 7F	2 2 2 F2	B 12-2 B C2	R 11-9 R D9	b 12-0-2 b 82	r 12-11-9 r 99	0-9-2 FS 22	11-9-8-2 CC 1A	12-0-9-3 43	11-0-9-2 62	12-11-0-9-8 78	11-0-8-2 AA	12-11-0-8-2 BA	12-11-9-8-6 DE
0 0 1 1	3	ETX 12-9-3 ETX 03	DC3 11-9-3 TM 13	# 8-3 # 7B	3 3 3 F3	C 12-3 C C3	S 0-2 S E2	c 12-0-3 c 83	s 11-0-2 s A2	0-9-3 23	9-3 33	12-0-9-4 44	11-0-9-3 63	12-0-8-1 80	11-0-8-3 AB	12-11-0-8-3 BB	12-11-9-8-7 DF
0 1 0 0	4	EOT 9-7 EOT 37	DC4 9-8-4 DC4 3C	$ 11-8-3 $ 5B	4 4 4 F4	D 12-4 D C4	T 0-3 T E3	d 12-0-4 d 84	t 11-0-3 t A3	0-9-4 BYP 24	9-4 PN 34	12-0-9-5 45	11-0-9-4 64	12-0-8-2 8A	11-0-8-4 AC	12-11-0-8-4 BC	11-0-9-8-2 EA
0 1 0 1	5	ENQ 0-9-8-5 ENQ 2D	NAK 9-8-5 NAK 3D	% 0-8-4 % 6C	5 5 5 F5	E 12-5 E C5	U 0-4 U E4	e 12-0-5 e 85	u 11-0-4 u A4	11-9-5 NL 15	9-5 RS 35	12-0-9-6 46	11-0-9-5 65	12-0-8-3 8B	11-0-8-5 AD	12-11-0-8-5 BD	11-0-9-8-3 EB
0 1 1 0	6	ACK 0-9-8-6 ACK 2E	SYN 9-2 SYN 32	& 12 & 50	6 6 6 F6	F 12-6 F C6	V 0-5 V E5	f 12-0-6 f 86	v 11-0-5 v A5	12-9-6 LC 06	9-6 UC 36	12-0-9-7 47	11-0-9-6 66	12-0-8-4 8C	11-0-8-6 AE	12-11-0-8-6 BE	11-0-9-8-4 FC EC
0 1 1 1	7	BEL 0-9-8-7 BEL 2F	ETB 0-9-6 ETB 26	' 8-5 ' 7D	7 7 7 F7	G 12-7 G C7	W 0-6 W E6	g 12-0-7 g 87	w 11-0-6 w A6	11-9-7 IL 17	12-9-8 GE 08	12-0-9-8 48	11-0-9-7 67	12-0-8-5 8D	11-0-8-7 AF	12-11-0-8-7 BF	11-0-9-8-5 ED
1 0 0 0	8	BS 11-9-6 BS 16	CAN 11-9-8 CAN 18	(12-8-5 (4D	8 8 8 F8	H 12-8 H C8	X 0-7 X E7	h 12-0-8 h 88	x 11-0-7 x A7	0-9-8 28	9-8 38	12-8-1 49	11-0-9-8 68	12-0-8-6 8E	12-11-0-8-1 B0	12-0-9-8-2 CA	11-0-9-8-6 EE
1 0 0 1	9	HT 12-9-5 HT 05	EM 11-9-8-1 EM 19	) 11-8-5) 5D	9 9 9 F9	I 12-9 I C9	Y 0-8 Y E8	i 12-0-9 i 89	y 11-0-8 y A8	0-9-8-1 29	9-8-1 39	12-11-9-1 51	0-8-1 69	12-0-8-7 8F	12-11-0-1 B1	12-0-9-8-3 CB	11-0-9-8-7 EF
1 0 1 0	10 (A)	LF 0-9-5 LF 25	SUB 9-8-7 SUB 3F	* 11-8-4 * 5C	: 8-2 : 7A	J 11-1 J D1	Z 0-9 Z E9	j 12-11-1 j 91	z 11-0-9 z A9	0-9-8-2 SM 2A	9-8-2 3A	12-11-9-2 52	12-11-0 70	12-11-8-1 90	12-11-0-2 B2	12-0-9-8-4 CC	12-11-0-9-8-2 I (LVM) FA
1 0 1 1	11 (B)	VT 12-9-8-3 VT 0B	ESC 0-9-7 ESC 27	+ 12-8-6 + 4E	; 11-8-6 ; 5E	K 11-2 K D2	[12-8-2 ¢ 4A	k 12-11-2 k 92	{ 12-0 { C0	0-9-8-3 CU2 2B	9-8-3 CU3 3B	12-11-9-3 53	12-11-0-9-1 71	12-11-8-2 9A	12-11-0-3 B3	12-0-9-8-5 CD	12-11-0-9-8-3 FB
1 1 0 0	12 (C)	FF 12-9-8-4 FF 0C	FS 11-9-8-4 IFS 1C	, 0-8-3 , 6B	< 12-8-4 < 4C	L 11-3 L D3	\ 0-8-2 \ E0	l 12-11-3 l 93	¦ 12-11 ¦ 6A	0-9-8-4 2C	12-9-4 PF 04	12-11-9-4 54	12-11-0-9-2 72	12-11-8-3 9B	12-11-0-4 B4	12-0-9-8-6 CE	12-11-0-9-8-4 FC
1 1 0 1	13 (D)	CR 12-9-8-5 CR 0D	GS 11-9-8-5 IGS 1D	– 11 – 60	= 8-6 = 7E	M 11-4 M D4	] 11-8-2 ! 5A	m 12-11-4 m 94	} 11-0 } D0	12-9-8-1 RLF 09	11-9-4 RES 14	12-11-9-5 55	12-11-0-9-3 73	12-11-8-4 9C	12-11-0-5 B5	12-0-9-8-7 CF	12-11-0-9-8-5 FD
1 1 1 0	14 (E)	SO 12-9-8-6 SO 0E	RS 11-9-8-6 IRS 1E	. 12-8-3 . 4B	> 0-8-6 > 6E	N 11-5 N D5	^ 11-8-7 ¬ 5F	n 12-11-5 n 95	~ 11-0-1 ~ A1	12-9-8-2 SMM 0A	9-8-6 3E	12-11-9-6 56	12-11-0-9-4 74	12-11-8-5 9D	12-11-0-6 B6	12-11-9-8-2 DA	12-11-0-9-8-6 FE
1 1 1 1	15 (F)	SI 12-9-8-7 SI 0F	US 11-9-8-7 IUS 1F	/ 0-1 / 61	? 0-8-7 ? 6F	O 11-6 O D6	_ 0-8-5 _ 6D	o 12-11-6 o 96	DEL 12-9-7 DEL 07	11-9-8-3 CU1 1B	11-0-9-1 E1	12-11-9-7 57	12-11-0-9-5 75	12-11-8-6 9E	12-11-0-7 B7	12-11-9-8-3 DB	EO 12-11-0-9-8-7 FF

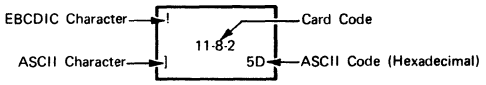
LEGEND



EXTENDED BINARY CODED DECIMAL INTERCHANGE CODE (EBCDIC) WITH PUNCHED CARD CODES AND ASCII TRANSLATION

BITS 0 1 2 3	0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
BITS 4 5 6 7 / 1ST HEX 2ND	0	1	2	3	4	5	6	7	8	9	A (10)	B (11)	C (12)	D (13)	E (14)	F (15)
0 0 0 0	NUL 12-0-9-8-1 NUL 00	DLE 12-11-9-8-1 DLE 10	DS 11-0-9-8-1 80	12-11-0-9-8-1 90	SP no punch SP 20	& 12 & 26	- 11 - 2D	12-11-0 BA	12-0-8-1 C3	12-11-8-1 CA	11-0-8-1 D1	12-11-0-8-1 D8	{ 12-0 { 7B	} 11-0 } 7D	\ 0-8-2 \ 5C	0 0 0 30
0 0 0 1	SOH 12-9-1 SOH 01	DC1 11-9-1 DC1 11	SOS 0-9-1 81	9-1 91	12-0-9-1 A0	12-11-9-1 A9	/ 0-1 / 2F	12-11-0-9-1 BB	a 12-0-1 a 61	j 12-11-1 j 6A	~ 11-0-1 ~ 7E	12-11-0-1 D9	A 12-1 A 41	J 11-1 J 4A	11-0-9-1 9F	1 1 1 31
0 0 1 0	STX 12-9-2 STX 02	DC2 11-9-2 DC2 12	FS 0-9-2 82	SYN 9-2 SYN 16	12-0-9-2 A1	12-11-9-2 AA	11-0-9-2 B2	12-11-0-9-2 BC	b 12-0-2 b 62	k 12-11-2 k 6B	s 11-0-2 s 73	12-11-0-2 DA	B 12-2 B 42	K 11-2 K 4B	S 0-2 S 53	2 2 2 32
0 0 1 1	ETX 12-9-3 ETX 03	TM 11-9-3 DC3 13	0-9-3 83	9-3 93	12-0-9-3 A2	12-11-9-3 AB	11-0-9-3 B3	12-11-0-9-3 BD	c 12-0-3 c 63	l 12-11-3 l 6C	t 11-0-3 t 74	12-11-0-3 DB	C 12-3 C 43	L 11-3 L 4C	T 0-3 T 54	3 3 3 33
0 1 0 0	PF 12-9-4 9C	RES 11-9-4 9D	BYP 0-9-4 84	PN 9-4 94	12-0-9-4 A3	12-11-9-4 AC	11-0-9-4 B4	12-11-0-9-4 BE	d 12-0-4 d 64	m 12-11-4 m 6D	u 11-0-4 u 75	12-11-0-4 DC	D 12-4 D 44	M 11-4 M 4D	U 0-4 U 55	4 4 4 34
0 1 0 1	HT 12-9-5 HT 09	NL 11-9-5 85	LF 0-9-5 LF 0A	RS 9-5 95	12-0-9-5 A4	12-11-9-5 AD	11-0-9-5 B5	12-11-0-9-5 BF	e 12-0-5 e 65	n 12-11-5 n 6E	v 11-0-5 v 76	12-11-0-5 DD	E 12-5 E 45	N 11-5 N 4E	V 0-5 V 56	5 5 5 35
0 1 1 0	LC 12-9-6 86	BS 11-9-6 BS 08	ETB 0-9-6 ETB 17	UC 9-6 96	12-0-9-6 A5	12-11-9-6 AE	11-0-9-6 B6	12-11-0-9-6 C0	f 12-0-6 f 66	o 12-11-6 o 6F	w 11-0-6 w 77	12-11-0-6 DE	F 12-6 F 46	O 11-6 O 4F	W 0-6 W 57	6 6 6 36
0 1 1 1	DEL 12-9-7 DEL 7F	IL 11-9-7 87	ESC 0-9-7 ESC 1B	EOT 9-7 EOT 04	12-0-9-7 A6	12-11-9-7 AF	11-0-9-7 B7	12-11-0-9-7 C1	g 12-0-7 g 67	p 12-11-7 p 70	x 11-0-7 x 78	12-11-0-7 DF	G 12-7 G 47	P 11-7 P 50	X 0-7 X 58	7 7 7 37
1 0 0 0	GE 12-9-8 97	CAN 11-9-8 CAN 18	0-9-8 88	9-8 98	12-0-9-8 A7	12-11-9-8 B0	11-0-9-8 B8	12-11-0-9-8 C2	h 12-0-8 h 68	q 12-11-8 q 71	y 11-0-8 y 79	12-11-0-8 E0	H 12-8 H 48	Q 11-8 Q 51	Y 0-8 Y 59	8 8 8 38
1 0 0 1	RLF 12-9-8-1 8D	EM 11-9-8-1 EM 19	0-9-8-1 89	9-8-1 99	12-8-1 A8	11-8-1 B1	0-8-1 B9	8-1 60	i 12-0-9 i 69	r 12-11-9 r 72	z 11-0-9 z 7A	12-11-0-9 E1	I 12-9 I 49	R 11-9 R 52	Z 0-9 Z 5A	9 9 9 39
1 0 1 0 (A)	SMM 12-9-8-2 8E	CC 11-9-8-2 92	SM 0-9-8-2 8A	9-8-2 9A	¢ 12-8-2 [5B	! 11-8-2] 5D	¦ 12-11 ¦ 7C	: 8-2 : 3A	12-0-8-2 C4	12-11-8-2 C8	11-0-8-2 D2	12-11-0-8-2 E2	12-0-9-8-2 E8	12-11-9-8-2 EE	11-0-9-8-2 F4	¡(LVM) 12-11-0-9-8-2 FA
1 0 1 1 (B)	VT 12-9-8-3 VT 0B	CU1 11-9-8-3 8F	CU2 0-9-8-3 8B	CU3 9-8-3 9B	. 12-8-3 . 2E	$ 11-8-3 $ 24	, 0-8-3 , 2C	# 8-3 # 23	12-0-8-3 C5	12-11-8-3 CC	11-0-8-3 D3	12-11-0-8-3 E3	12-0-9-8-3 E9	12-11-9-8-3 EF	11-0-9-8-3 F5	12-11-0-9-8-3 FB
1 1 0 0 (C)	FF 12-9-8-4 FF 0C	IFS 11-9-8-4 FS 1C	0-9-8-4 8C	DC4 9-8-4 DC4 14	< 12-8-4 < 3C	* 11-8-4 * 2A	% 0-8-4 % 25	@ 8-4 @ 40	12-0-8-4 C6	12-11-8-4 CD	11-0-8-4 D4	12-11-0-8-4 E4	↵ 12-0-9-8-4 EA	12-11-9-8-4 F0	↵ 11-0-9-8-4 F6	12-11-0-9-8-4 FC
1 1 0 1 (D)	CR 12-9-8-5 CR 0D	IGS 11-9-8-5 GS 1D	ENQ 0-9-8-5 ENQ 05	NAK 9-8-5 NAK 15	(12-8-5 (28	) 11-8-5) 29	¯ 0-8-5 _ 5F	' 8-5 ' 27	12-0-8-5 C7	12-11-8-5 CE	11-0-8-5 D5	12-11-0-8-5 E5	12-0-9-8-5 EB	12-11-9-8-5 F1	11-0-9-8-5 F7	12-11-0-9-8-5 FD
1 1 1 0 (E)	SO 12-9-8-6 SO 0E	IRS 11-9-8-6 RS 1E	ACK 0-9-8-6 ACK 06	9-8-6 9E	+ 12-8-6 + 2B	; 11-8-6 ; 3B	>> 0-8-6 > 3E	= 8-6 = 3D	12-0-8-6 C8	12-11-8-6 CF	11-0-8-6 D6	12-11-0-8-6 E6	↳ 12-0-9-8-6 EC	12-11-9-8-6 F2	11-0-9-8-6 F8	12-11-0-9-8-6 FE
1 1 1 1 (F)	SI 12-9-8-7 SI 0F	IUS 11-9-8-7 US 1F	BEL 0-9-8-7 BEL 07	SUB 9-8-7 SUB 1A	| 12-8-7 ! 21	¬ 11-8-7 ^ 5E	? 0-8-7 ? 3F	" 8-7 " 22	12-0-8-7 C9	12-11-8-7 D0	11-0-8-7 D7	12-11-0-8-7 E7	12-0-9-8-7 ED	12-11-9-8-7 F3	11-0-9-8-7 F9	12-11-0-9-8-7 EO FF

LEGEND



CONTROL DATA CHARACTER SETS
SHOWING TRANSLATIONS BETWEEN DISPLAY CODE AND ASCII/EBCDIC

DISPLAY CODE		ASCII				EBCDIC			
		UPPER CASE		LOWER CASE		UPPER CASE		LOWER CASE	
OCTAL	CH	CH	HEX	CH	HEX	CH	HEX	CH	HEX
00	:	:	3A	SUB	1A	:	7A	SUB	3F
01	A	A	41	a	61	A	C1	a	81
02	B	B	42	b	62	B	C2	b	82
03	C	C	43	c	63	C	C3	c	83
04	D	D	44	d	64	D	C4	d	84
05	E	E	45	e	65	E	C5	e	85
06	F	F	46	f	66	F	C6	f	86
07	G	G	47	g	67	G	C7	g	87
10	H	H	48	h	68	H	C8	h	88
11	I	I	49	i	69	I	C9	i	89
12	J	J	4A	j	6A	J	D1	j	91
13	K	K	4B	k	6B	K	D2	k	92
14	L	L	4C	l	6C	L	D3	l	93
15	M	M	4D	m	6D	M	D4	m	94
16	N	N	4E	n	6E	N	D5	n	95
17	O	O	4F	o	6F	O	D6	o	96
20	P	P	50	p	70	P	D7	p	97
21	Q	Q	51	q	71	Q	D8	q	98
22	R	R	52	r	72	R	D9	r	99
23	S	S	53	s	73	S	E2	s	A2
24	T	T	54	t	74	T	E3	t	A3
25	U	U	55	u	75	U	E4	u	A4
26	V	V	56	v	76	V	E5	v	A5
27	W	W	57	w	77	W	E6	w	A6
30	X	X	58	x	78	X	E7	x	A7
31	Y	Y	59	y	79	Y	E8	y	A8
32	Z	Z	5A	z	7A	Z	E9	z	A9
33	0	0	30	DLE	10	0	F0	DLE	10
34	1	1	31	DC1	11	1	F1	DC1	11
35	2	2	32	DC2	12	2	F2	DC2	12
36	3	3	33	DC3	13	3	F3	TM	13
37	4	4	34	DC4	14	4	F4	DC4	3C

DISPLAY CODE		ASCII				EBCDIC			
		UPPER CASE		LOWER CASE		UPPER CASE		LOWER CASE	
OCTAL	CH	CH	HEX	CH	HEX	CH	HEX	CH	HEX
40	5	5	35	NAK	15	5	F5	NAK	3D
41	6	6	36	SYN	16	6	F6	SYN	32
42	7	7	37	ETB	17	7	F7	ETB	26
43	8	8	38	CAN	18	8	F8	CAN	18
44	9	9	39	EM	19	9	F9	EM	19
45	+	+	2B	VT	0B	+	4E	VT	0B
46	-	-	2D	CR	0D	-	60	CR	0D
47	*	*	2A	LF	0A	*	5C	LF	25
50	/	/	2F	SI	0F	/	61	SI	0F
51	(	(	28	BS	08	(	4D	BS	16
52	)	)	29	HT	09	)	5D	HT	05
53	$	$	24	EOT	04	$	5B	EOT	37
54	=	=	3D	GS	1D	=	7E	IGS	1D
55	SP	SP	20	NUL	00	SP	40	NUL	00
56	,	,	2C	FF	0C	,	6B	FF	0C
57	.	.	2E	SO	0E	.	4B	SO	0E
60	≡ ≠	≠	23	ETX	03	#	7B	ETX	03
61	[	[	5B	FS	1C	¢	4A	IFS	1C
62	]	]	5D	SOH	01	!	5A	SOH	01
63	%	%	25	ENQ	05	%	6C	ENQ	2D
64	≠ "	"	22	STX	02	"	7F	STX	02
65	→ _	_	5F	DEL	7F	_	6D	DEL	07
66	∨ ¦	¦	21	}	7D	¦	4F	¦	D0
67	∧ &	&	26	ACK	06	&	50	ACK	2E
70	↑ '	'	27	BEL	07	'	7D	BEL	2F
71	↓ ?	?	3F	US	1F	?	6F	IUS	1F
72	<	<	3C	{	7B	<	4C	¦	C0
73	>	>	3E	RS	1E	>	6E	IRS	1E
74	≤ @	@	40	`	60	@	7C	`	79
75	≥ \	\	5C	¦	7C	\	E0	¦	6A
76	¬ ^	^	5E	~	7E	¬	5F	~	A1
77	;	;	3B	ESC	1B	;	5E	ESC	27

NOTES:

1. The terms "upper case" and "lower case" apply only to the case conversions, and do not necessarily reflect any true "case".
2. When translating from Display Code to ASCII/EBCDIC, the "upper case" equivalent character is taken.
3. When translating from ASCII/EBCDIC to Display Code, the "upper case" and "lower case" characters fold together to a single Display Code equivalent character.
4. All ASCII and EBCDIC codes not listed are translated to Display Code 55 (SP).
5. Where two Display Code graphics are shown for a single octal code, the leftmost graphic corresponds to the CDC 64-character set (system assembled with IP.CSET set to C64.1), and the rightmost graphic corresponds to the CDC 64-character ASCII subset (system assembled with IP.CSET set to C64.2).
6. In a 63-character set system, the display code for the : graphic is 63. The % character does not exist, and translations from ASCII/EBCDIC % or ENQ yield blank (55_8).

HEXADECIMAL–OCTAL CONVERSION TABLE

Second Hexadecimal Digit	First Hexadecimal Digit 0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	000	020	040	060	100	120	140	160	200	220	240	260	300	320	340	360
1	001	021	041	061	101	121	141	161	201	221	241	261	301	321	341	361
2	002	022	042	062	102	122	142	162	202	222	242	262	302	322	342	362
3	003	023	043	063	103	123	143	163	203	223	243	263	303	323	343	363
4	004	024	044	064	104	124	144	164	204	224	244	264	304	324	344	364
5	005	025	045	065	105	125	145	165	205	225	245	265	305	325	345	365
6	006	026	046	066	106	126	146	166	206	226	246	266	306	326	346	366
7	007	027	047	067	107	127	147	167	207	227	247	267	307	327	347	367
8	010	030	050	070	110	130	150	170	210	230	250	270	310	330	350	370
9	011	031	051	071	111	131	151	171	211	231	251	271	311	331	351	371
A	012	032	052	072	112	132	152	172	212	232	252	272	312	332	352	372
B	013	033	053	073	113	133	153	173	213	233	253	273	313	333	353	373
C	014	034	054	074	114	134	154	174	214	234	254	274	314	334	354	374
D	015	035	055	075	115	135	155	175	215	235	255	275	315	335	355	375
E	016	036	056	076	116	136	156	176	216	236	256	276	316	336	356	376
F	017	037	057	077	117	137	157	177	217	237	257	277	317	337	357	377
Octal	000 – 037		040 – 077		100 – 137		140 – 177		200 – 237		240 – 277		300 – 337		340 – 377	

CENTRAL MEMORY RESIDENT TABLES B

CENTRAL MEMORY RESIDENT

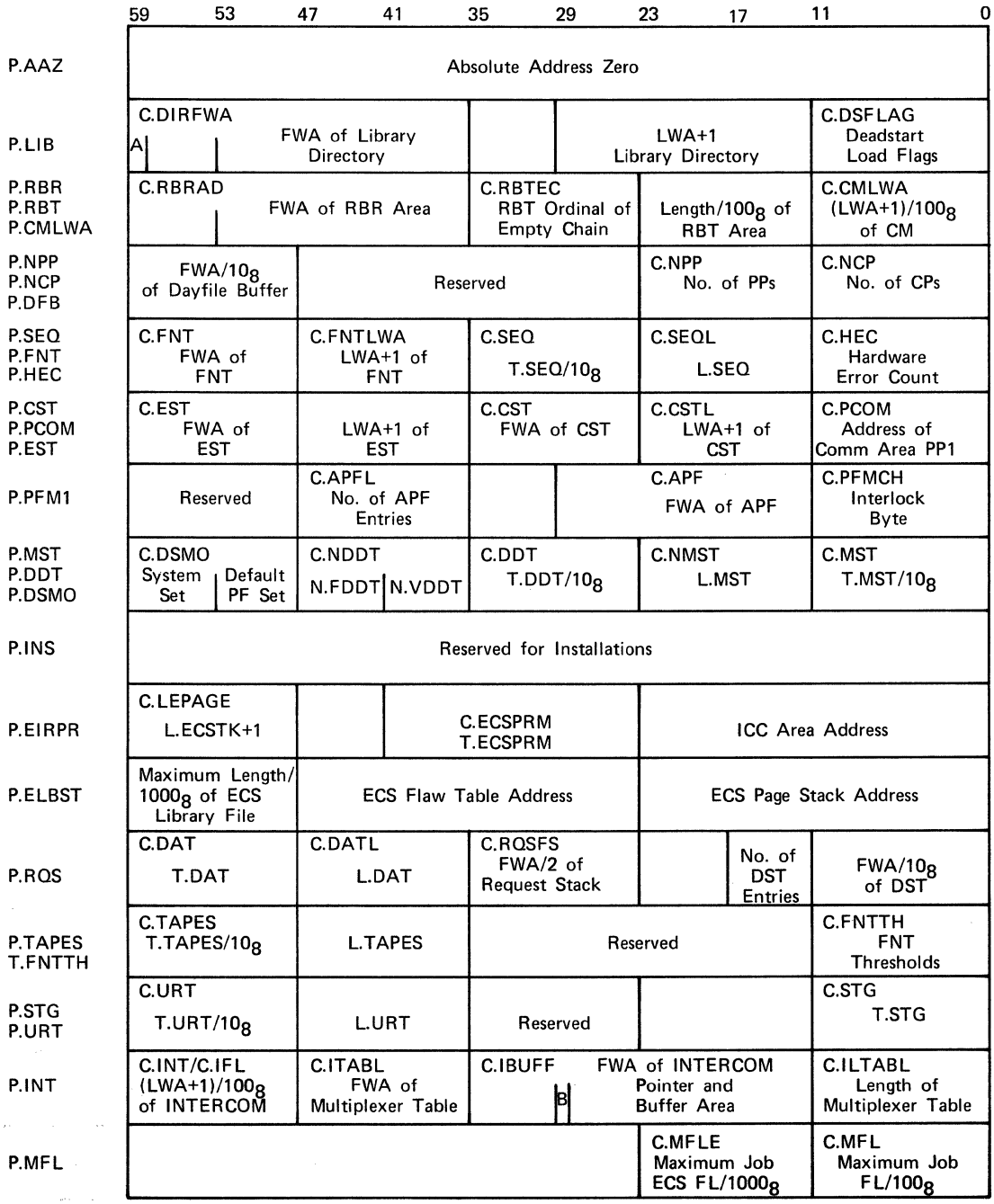
First Word Address	Table Name	Description
0		CMR pointer area.
100	T.CST	Channel status table.
154	T.PPS1	PP status words.
200	T.CPA$_n$	Control point areas.
	T.XPIDLA	System job exchange package area.
	T.PPC1	PP communication areas.
	T.EST †	Equipment status table.
	T.FNT †	File name table. CIO-CPCIO special file name tables. Permanent file name tables. T.ELIBD - ECS resident library descriptor word.
	T.ITABL †	INTERCOM table.
	T.DAT †	Device activity table.
	T.RMSBUF †	RMS buffer.
	T.STG †	Tapes staging table.
	T.APF	Attached permanent file table.
	T.EXPIO	CYBER 176 exchange package and I/O buffers.
	T.CHT	Channel table.
	T.UQT	Unit queue table.
	T.RQS ††	Request stack.
	T.RST	Request scheduling table.
	T.RBR	Record block reservation table (headers).
	T.RBRBIT	RBR bit table.
	T.DST	Device status table.

† Table must begin before 10000_8.
†† Table must begin before 20000_8.

First Word Address	Table Name	Description
	T.DOT	Device overflow table.
	T.SEQ	Sequencer table.
	T.INS	Installation area.
	T.MST	Mounted set table.
	T.DDT	Dismountable device table.
	T.TRB	Trace buffer.
	T.VRNBUF	VSN buffer.
	T.TAPES	Tapes table.
	T.URT	Tape unit recovery table.
	T.MAIL	Scheduler mailbox buffer.
	T.IDT	Logical ID table.
	T.DFB	Dayfile buffers.
	T.PJT	Parameter storage for delayed PP jobs.
	T.MAB	Mainframe attribute block.
	T.SSCT	Subsystem control table.
	T.SCHPT	Scheduler performance table (optional).
	T.SCHJCA	Scheduler job control area.
	T.SCHJDT	Scheduler job descriptor table.
	T.ELST	Error logging status table.
	T.PPOVL	PP resident overlay save buffer.
	T.BRKPT	Breakpoint table (ECS system).
	T.AREA	Area table (ECS system).
	T.ENTRY	Entry table (ECS system).
	T.BCFAP	CEFAP buffer.
	T.EPAGE	Empty page stack.
	T.ECSPRM	ECS parameters.
	T.SCBHDR	System circular buffer.

First Word Address	Table Name	Description
	T.SUBPG	Subpage buffer.
		CM resident programs (disk system).
		Segmented system areas (ECS system).
	T.LIB	Library directory.
		INTERCOM pointer area. INTERCOM buffers and user tables.
		Job control point user field length.
	T.RBT	RBT chains.

CMR POINTER AREA



The contents of the following words can change in the released version of system.

0 P.AAZ

Contains absolute address zero.

1 P.LIB

A library change flag appears in bit 59. Right-justified in bytes 0 and 1 is the first word address of the library directory; bytes 2 and 3 contain the right-justified last word address plus one of the library directory. Byte 4 contains deadstart load flags.

Bits in the byte C.DSFLAG are:

0	S.SYSEDT	1	Bypass EDITLIB GO/DROP message (internal to deadstart).
1		Not used.	
2	S.MFLVL	0	First mainframe to deadstart.
		1	Not first mainframe to deadstart.
3	S.CMU	0	Compare/move not available.
		1	Compare/move available.
4	S.ECSLVL	0	No ECS.
		1	ECS up.
5	S.USETS	0	Do not validate user sets.
		1	Validate user sets.
7-6	S.IOLVL	00	Recover I/O queues.
		01	Do not recover I/O queues.
		10	Initialize I/O queues.
10-8	S.DSLVL	Deadstart level.	
		000	Level 0 deadstart.
		001	Level 1 deadstart.
		010	Level 2 deadstart.
		011	Level 3 deadstart.
		100 thru 111	Not defined.
11	S.RLIB	0	Reload system libraries.
		1	Do not reload system libraries.

2 P.RBT/P.RBR/P.CMLWA

Bytes 0 and 1 contain the right-justified first word address of the record block reservation area. Byte 2 contains the record block table word-pair ordinal of the first member of the RBT empty chain. Byte 3 contains the current length of the RBT area in 100-word blocks. Byte 4 contains the current size of central memory in 100-word blocks.

3 P.DFB/P.NPP/P.NCP

Byte 0 contains the FWA/10 of the dayfile buffer area. Bytes 1 and 2 are reserved for the 250 graphics package. Bytes 3 and 4 contain the number of PPs and control points in the system, respectively.

4 P.FNT/P.SEQ/P.HEC

Bytes 0 and 1 contain the FWA and LWA+1 addresses of the file name table. Bytes 2 and 3 contain the FWA and length of the sequencer table (SEQ). Byte 4 contains the hardware error count.

5 P.EST/P.CST/P.PCOM

Bytes 0 and 1 contain the FWA and LWA+1 addresses of the equipment status table. Bytes 2 and 3 contain the FWA and LWA+1 addresses of the channel status table. Byte 4 contains the FWA of the communications area for PP1.

6 P.PFM1

Byte 1 contains the number of attached permanent file table entries. Bits 29 through 12 contain FWA of attached permanent file table. The permanent file interlock byte is in byte 4.

Bits in the byte C.PFMCH are:

9-0		Reserved.
10	S.FNTTH	FNT space critical; input is halted.
11	S.PFCIOQ	PFC full flag; input jobs are halted.

7 P.MST/P.DDT/P.DSMO

C.DSMO	Bits 11-6	MST ordinal for system set.
	Bits 5-0	MST ordinal for PF default set.
C.NDDT	Bits 11-6	Number of fixed DDT entries.
	Bits 5-0	Number of variable DDT entries.
C.DDT	T.DDT is the first word address divided by 10 octal of the DDT.	
C.NMST	L.MST is the number of entries in the MST.	
C.MST	T.MST is the first word address divided by 10 octal of the MST.	

10 P.INS

Reserved for installation use.

11 P.EIRPR

Byte 0 contains the size of the ECS page stack; right-justified in bytes 1 and 2 is the FWA of the ECS parameter table. Right-justified in bytes 3 and 4 is the FWA of the ICC area address.

12 P.ELBST

Contains the ECS flaw table address and the address of a one-word table in ECS that contains the index to an empty page stack. Byte 0 contains the maximum size of the ECS library file in 1000-word pages.

13 P.RQS

Bytes 0 and 1 contain the FWA and length, respectively, of the device activity table. Byte 2 contains FWA/2 of the request stack. Right-justified in byte 3 is the current number of device status entries. FWA of the device status table is in byte 4.

14 P.TAPES/T.FNTTH

Bytes 0 and 1 contain the FWA/10 and length of the tape configuration table. Byte 4 contains the lower FNT space threshold/10_8 in bits 11 through 6 and the upper FNT space threshold/10_8 in bits 5 through 0.

15 P.STG/P.URT

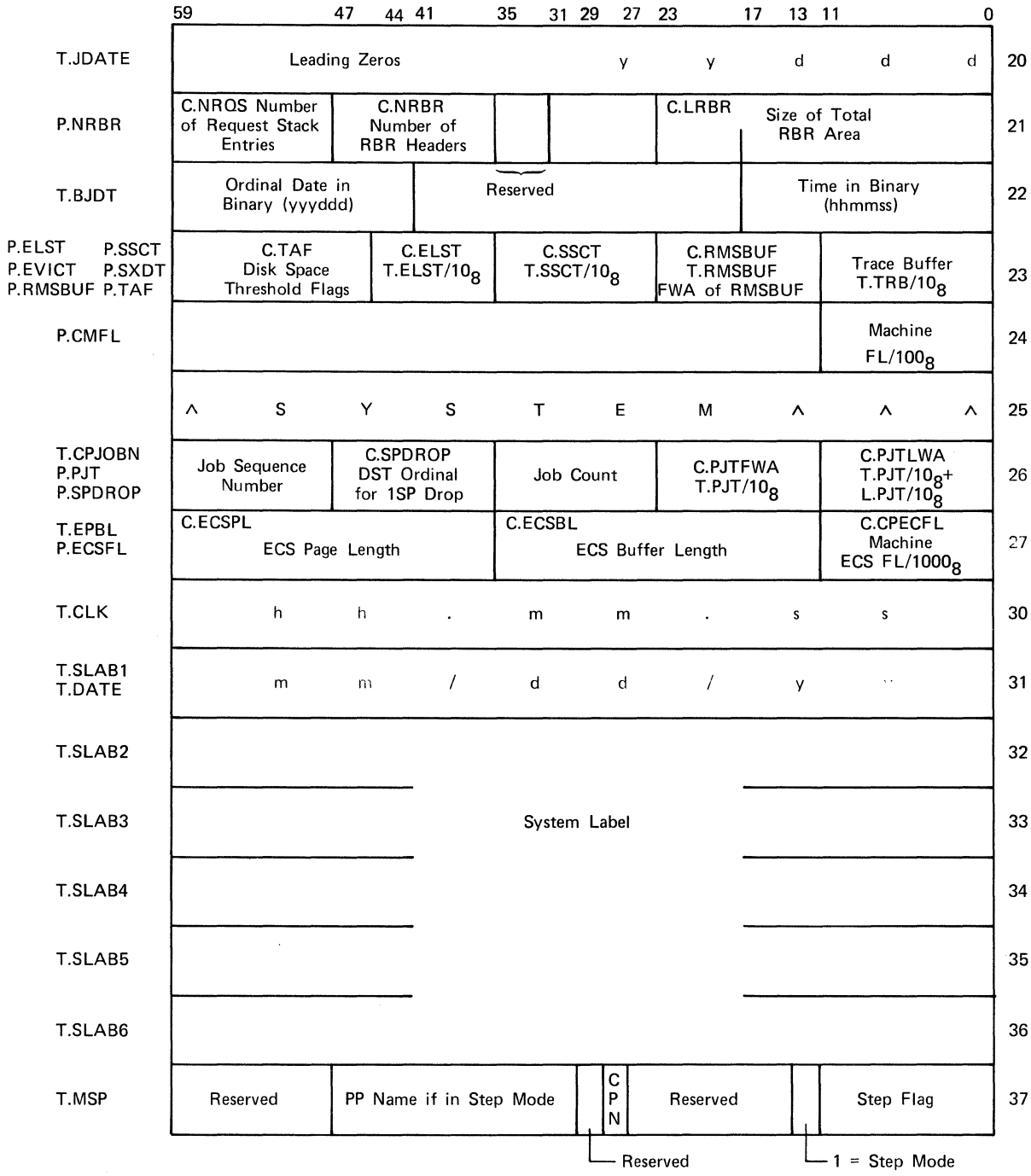
Bytes 0 and 1 contain the FWA/10 and length of the tape unit recovery table. Byte 4 contains the FWA of the tape staging table.

16 P.INT

Byte 0 contains the (LWA+1)/100_8 of the INTERCOM pointer and buffer area. Byte 1 contains the FWA and byte 4 the length of the INTERCOM multiplexer table. Bytes 2 and 3 contain the FWA of the INTERCOM pointer and buffer area. Bit 30 is a flag which is nonzero when deadstart is in progress.

17 P.MFL

Byte 3 contains the maximum job ECS field length (MFLE)/1000_8. Byte 4 contains the maximum job CM field length (MFL)/100_8.


59 47 44 41 35 31 29 27 23 17 13 11 0
T.JDATE
Leading Zeros
y y d d d
20
P.NRBR
C.NRQS Number of Request Stack Entries
C.NRBR Number of RBR Headers
C.LRBR
Size of Total RBR Area
21
T.BJDT
Ordinal Date in Binary (yyyyddd)
Reserved
Time in Binary (hhmmss)
22
P.ELST P.SSCT
P.EVICT P.SXDT
P.RMSBUF P.TAF
C.TAF Disk Space Threshold Flags
C.ELST T.ELST/10₈
C.SSCT T.SSCT/10₈
C.RMSBUF T.RMSBUF FWA of RMSBUF
Trace Buffer T.TRB/10₈
23
P.CMFL
Machine FL/100₈
24
^ S Y S T E M ^ ^ ^
25
T.CPJOBN
P.PJT
P.SPDROP
Job Sequence Number
C.SPDROP DST Ordinal for 1SP Drop
Job Count
C.PJTFWA T.PJT/10₈
C.PJTLWA T.PJT/10₈+ L.PJT/10₈
26
T.EPBL
P.ECSFL
C.ECSPL ECS Page Length
C.ECSBL ECS Buffer Length
C.CPECFL Machine ECS FL/1000₈
27
T.CLK
h h . m m . s s
30
T.SLAB1
T.DATE
m m / d d / y y
31
T.SLAB2
32
T.SLAB3
System Label
33
T.SLAB4
34
T.SLAB5
35
T.SLAB6
36
T.MSP
Reserved
PP Name if in Step Mode
C P N
Reserved
Step Flag
37
Reserved
1 = Step Mode

20 T.JDATE

The current ordinal date is stored here in the form yyddd with leading zeros.

21 P.NRBR

Byte 0 contains the number of entries in the request stack; byte 1 contains the number of RBR headers; the length of the RBR area is right-justified in bytes 3 and 4.

22 T.BJDT

The current ordinal date (yyyddd) in binary form is stored in bits 59 through 42; the current time (hhmmss), in binary form, is stored in bits 17 through 0.

23 P.EVICT/P.RMSBUF/P.SSCT/P.ELST/P.SXDT/P.TAF

Byte 0 contains the activity flags used by 2RN to determine whether disk space thresholds are active (that is, available space exceeds the threshold). Byte 1 contains the FWA of the error logging status table divided by 8. Byte 2 contains the FWA of the subsystem control table divided by 8. Byte 3 contains the FWA of the rotating mass storage buffer. Byte 4 contains the FWA of the trace buffer divided by 8.

24 P.CMFL

Contains, in byte 4, the current machine core size in 100-word blocks.

25 W.CPJNAM

Contains control point zero job name in the form SYSTEM.

26 T.CPJOBN/P.PJT/P.SPDROP

Bytes 0 and 2 contain the job sequence number and job count, respectively. FWA and LWA+1 of the parameter storage area for a delayed PP job is in bytes 3 and 4. Byte 1 contains the current DST ordinal to be used to drop 1SP. Before MTR initialization, byte 1 can be used to reserve additional PPs for the stack processor.

27 T.EPBL/P.ECSFL

Right-justified in bytes 0 and 1 is the ECS page length as set by IP.EPAG; right-justified in bytes 2 and 3 is the ECS buffer length set by IP.EBUF. Byte 4 contains the total number of ECS pages assigned to direct access.

30 T.CLK

Current display clock time in the format:

hh.mm.ss

Starting time is entered by the operator.

31 T.DATE/T.SLAB1

Current calendar date in the format:

mm/dd/yy

The current date is entered by the operator. This is the first of a six-word system display label.

32 (IP.SYSL1) TSLAB5

33 T.SLAB3

Words 32 and 33 provide storage for up to 20 characters for the system display label first line, as given by installation parameter.

34 (IP.VER) T.SLAB4

System version identification.

35 (IP.SYSE) T.SLAB5

System edition date.

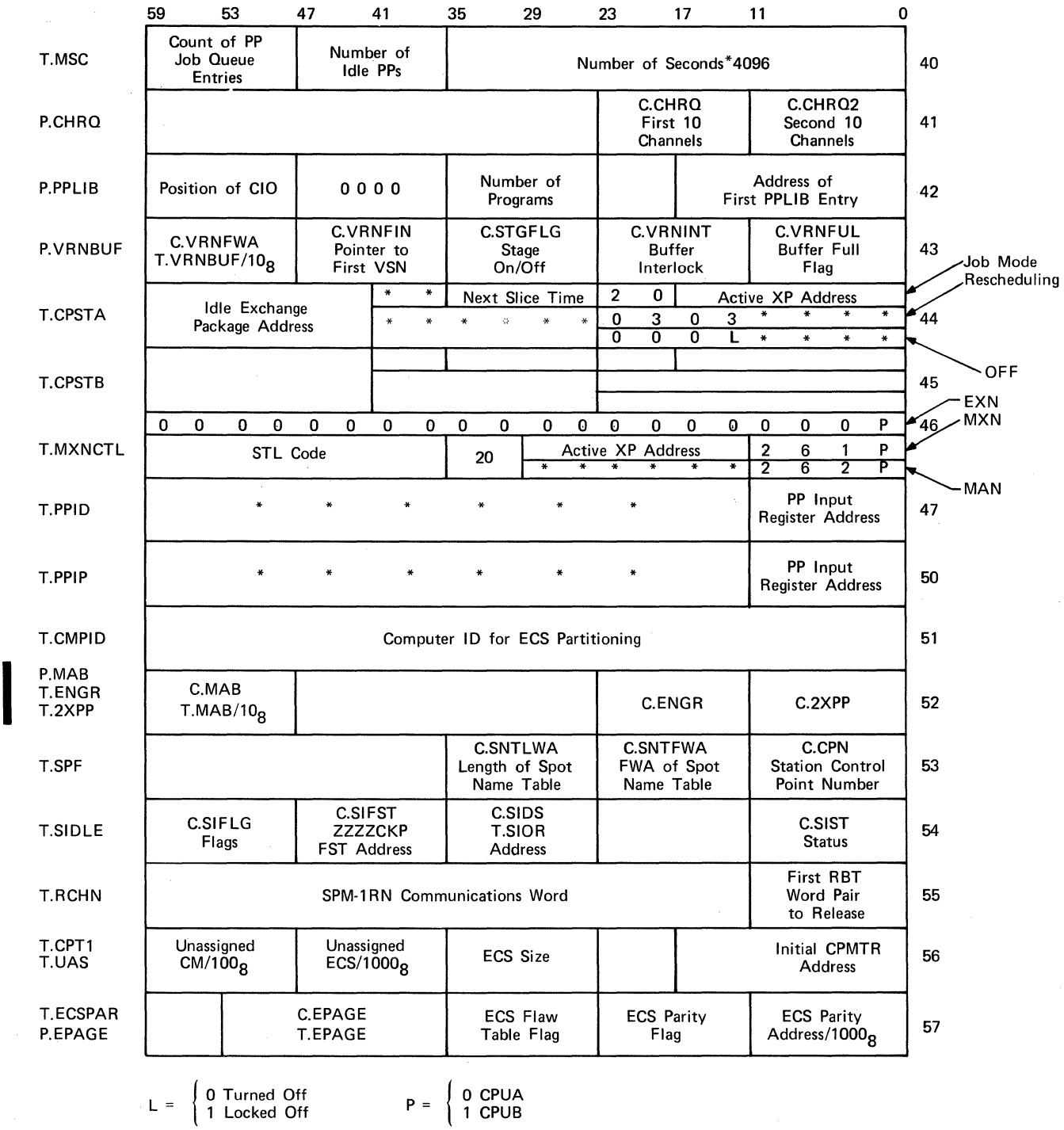
36 T.SLAB6

Used by system dynamic dump.

37 T.MSP

MTR-DSD step mode communication word.

Bit	Meaning
59-48	Reserved.
47-30	Name of PP routine, if any.
29-28	Reserved.
27-24	Control point number (0 = entire system).
23-13	Reserved.
12	1 Step mode.
	0 No step.
11-0	Communication byte.



L = { 0 Turned Off / 1 Locked Off

P = { 0 CPUA / 1 CPUB

40 T.MSC

Real time clock (microsecond count) in bytes 2 through 4. Count of jobs in PP job queue in byte 0; a count of idle PPs is in byte 1.

41 P.CHRQ

Active channel request queue flags for the first 10 channels in byte 3; for the second 10 channels in byte 4.

42 P.PPLIB

Byte 0 contains the current position of CIO; byte 1 is a communication byte between MDI and 1SP/1SQ; the number of programs in the system library is given in byte 2. The address of the first library table entry is right-justified in bytes 3 and 4.

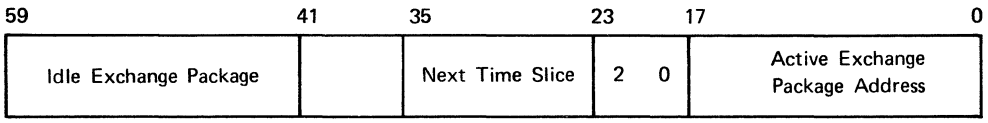
43 P.VRNBUF

Contains volume serial (visual reel) number buffer information and tape staging flags.

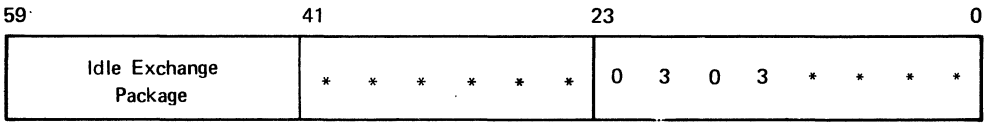
44 T.CPSTA

45 T.CPSTB (one word for each CPU). C.CPUOFF is a pointer to byte 3.

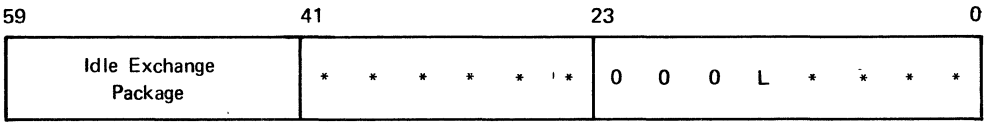
Value while the CPU is running in job mode:



Value while an MXN version of CPMTR is selecting the next job mode:

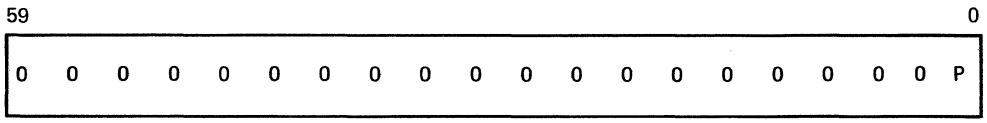


Value while the CPU is turned off:

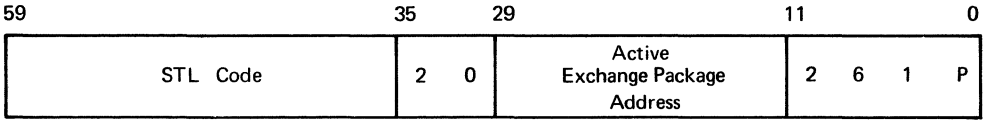


46 T.MXNCTL

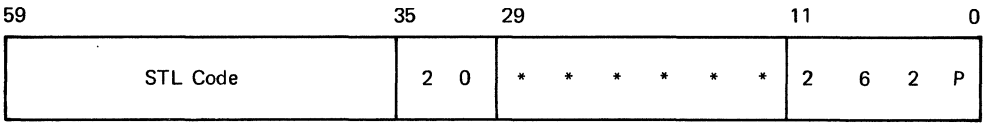
Value while in EXN mode:



Value while in MXN mode:



Value while in MAN mode:



This word contains PP executable code. It is used by PP resident to perform the MXN for any CP monitor function.

P = 0 or 1. The CPU number of the CPU that is running the lower priority job. CPMTR sets this and the active exchange package address at which the job is running.

47 T.PPID

Byte 4 contains the PP input register address of a PP that is waiting for a CPMTR function.

50 T.PPIP

Byte 4 contains the PP input register address of a PP that is waiting for a PPMTR function.

51 T.CMPID

Computer ID for ECS partitioning.

52 P.MAB/T.ENGR/T.2XPP

Byte 0 contains the first word address of the mainframe attribute block divided by 10_8.

Byte 3 contains the engineering mode flag.

0 Indicates engineering mode off.

1 Indicates engineering mode on.

Byte 4 contains the 2X PP speed bit taken from the status and control register (CYBER 170 only).

53 T.SPF

The length and the first word address of the spot name table are in bytes 2 and 3, respectively. The station control point number is in byte 4.

54 T.SIDLE

Byte 0 contains internal flags used by IDLE mode routines. The bits in C.SIFLG are:

Bit	Field	Description
59	S.SICPOF	SYSIDLE turned off a CPU.
58	S.SITDS	TDS was called to initialize ZZZZCKP.
57	S.SICPM	CPMTR initiated SYSIDLE.
56-55	S.SISR	Count of stack requests issued by SYSIDLE.
54	S.SISTEP	SYSIDLE has initiated system STEP mode.

Byte 1 contains the address of FST word 1 for the file ZZZZCKP. Byte 2 contains the address of the SYSIDLE pseudo-PPOR T.SIOR. Byte 4 contains the status of IDLE mode activity as defined by the following codes:

0 SYSIDLE not active.

1 SYSIDLE active.

2 SYSIDLE waiting for acknowledgement from PP which initiated IDLE mode.

3 SYSIDLE waiting for RESUME command.

4 RESUME command entered.

55 T.RCHN

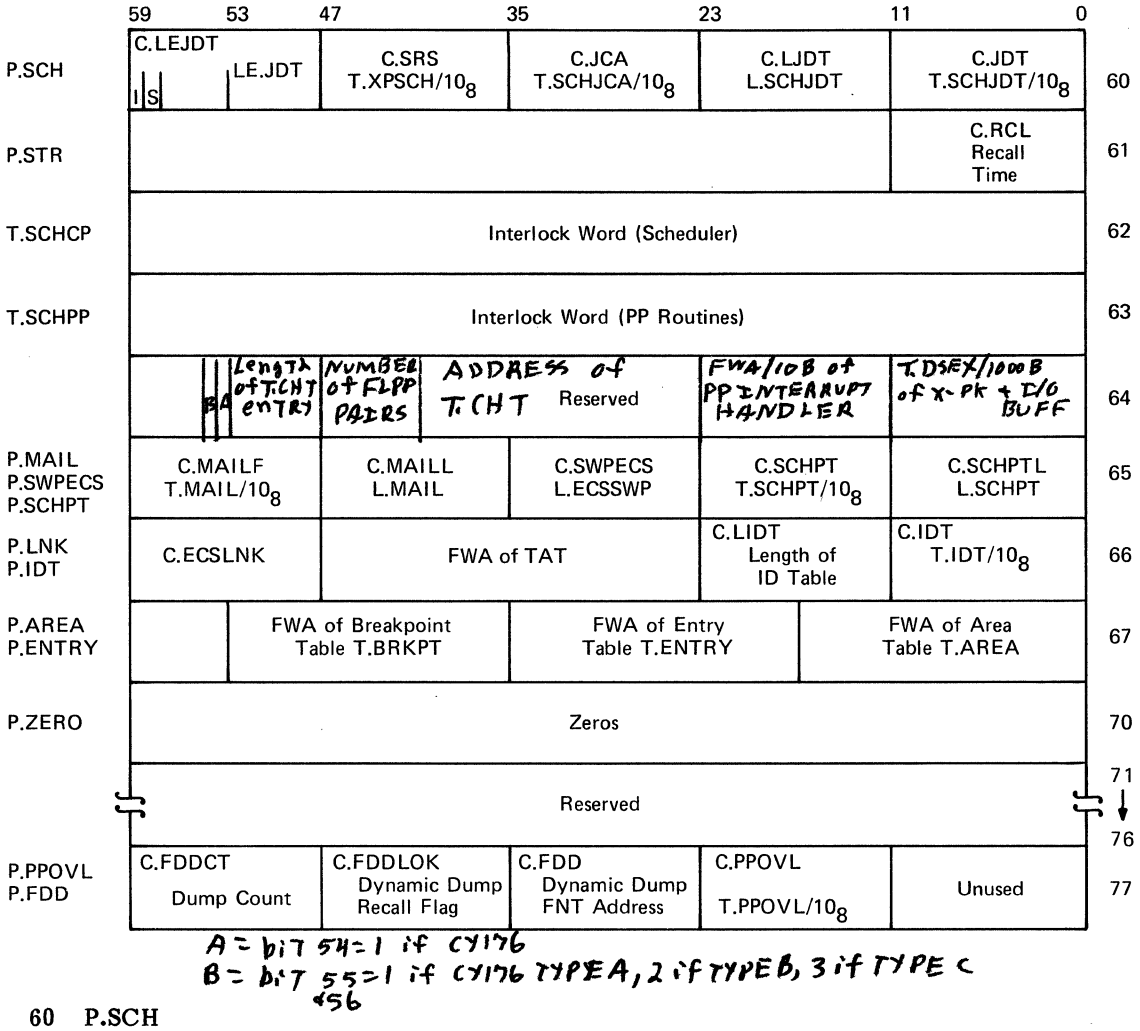
First RBT word pointer of the chain to be released in byte 4. The rest of the word contains the SPM-1RN communications word.

56 T.UAS/T.CPT1

Byte 0 contains the number of unassigned CM 100-word storage blocks; byte 1 contains the number of unassigned ECS 1000-word blocks. Byte 2 contains the current size of ECS. Bytes 3 and 4 contain, right-justified, the initial program address of the CM monitor.

57 T.ECSPAR/P.EPAGE

The FWA of the ECS page stack is right-adjusted in bytes 0 and 1; byte 2 contains ECS flaw table full flag; byte 3 contains ECS parity flag; byte 4 contains ECS block address in which parity error occurred.



A = bit 54 = 1 if CY176
B = bit 55 = 1 if CY176 TYPE A, 2 if TYPE B, 3 if TYPE C
456

60 P.SCH

Contains information relative to the integrated scheduler. Pointer to job scheduler exchange package is in byte 1; a pointer to the job control area is in byte 2; length and pointer to the job description table is in bytes 3 and 4; length of job description table entries is in byte 0. Bit 59 set to one indicates INTERCOM capability; bit 58 set to one indicates that direct access user ECS can be swapped.

61 P.STR

Integrated scheduler recall time in milliseconds.

62 P.SCHCP

Interlock word for integrated scheduler.

63 P.SCHPP

Interlock word for PP routines.

64 Reserved.

65 P.MAIL/P.SWPECS/P.SCHPT

Byte 0 contains the pointer to the scheduler mailbox buffer; byte 1 contains the length of the scheduler mailbox buffer. Byte 2 contains the ECS swap flags. Byte 3 contains the pointer to the scheduler performance table; byte 4 contains the length of the scheduler performance table.

The ECS swap flag bits are:

0 Swap INTERCOM jobs to ECS at end of job (EOJ bit set).

1 Swap batch jobs in central memory queues to ECS.

2 All INTERCOM and graphics jobs are to be swapped.

3 All batch jobs are to be swapped.

4 819 disk is available for all swapping.

6 Swap CM and ECS in parallel.

66 P.IDT/P.LNK

Byte 0 contains the ECS link restart time control mask. Bytes 1 and 2 contain the first word address of the TBT address table for 819 files. Byte 3 contains the length of the ID table. Byte 4 contains T.IDT/10_8.

67 P.AREA/P.ENTRY

Pointers to the first word address of the breakpoint table, the area table, and the entry table used for ECS systems.

70 P.ZERO

Contains a full word of binary zeros; a PP can clear five bytes at a time by reading this location. Changing the contents of this word can destroy system operation.

71-76 Reserved.

77 P.PPOVL/P.FDD

Byte 0 contains a count of dumps on the system dynamic dump file. Byte 1 contains flag bit S.FDDLOK (0) set by routines which wait for 1DD to complete, and cleared by 1DD. Byte 2 contains the FNT address of the system dynamic dump file ZZZZZDD. Byte 3 contains the FWA of the PP overlay table.

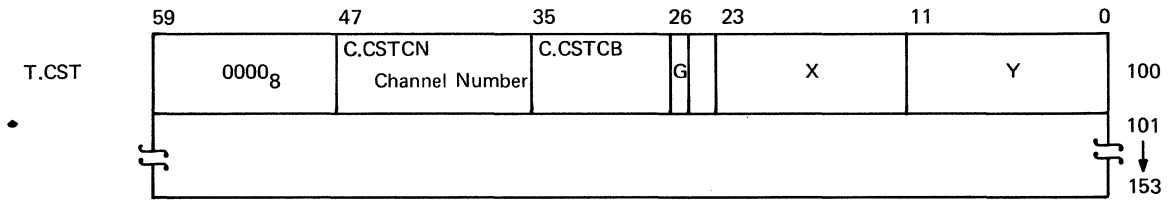
100 T.CST

Entries for the channel status table.

154 T.PPSn

One word entries containing status information for up to 20 PPs, beginning with PP1.

CHANNEL STATUS TABLE (CST)



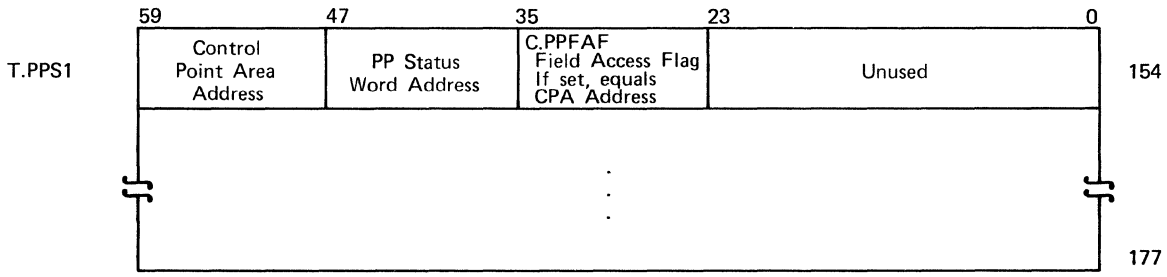
Byte 2 of word 5 in the CMR pointer area contains the first word address of the CST. Byte 3 of word 5 in the CMR pointer area contains the last word address plus 1 of the CST.

Field Name	Description
G	0 Normal charge for channel time.
	1 No charge to control point for channel time (S.CSNC).
X	Address of this word.
Y	Same as X when channel is not reserved; PPIR address when channel is reserved.

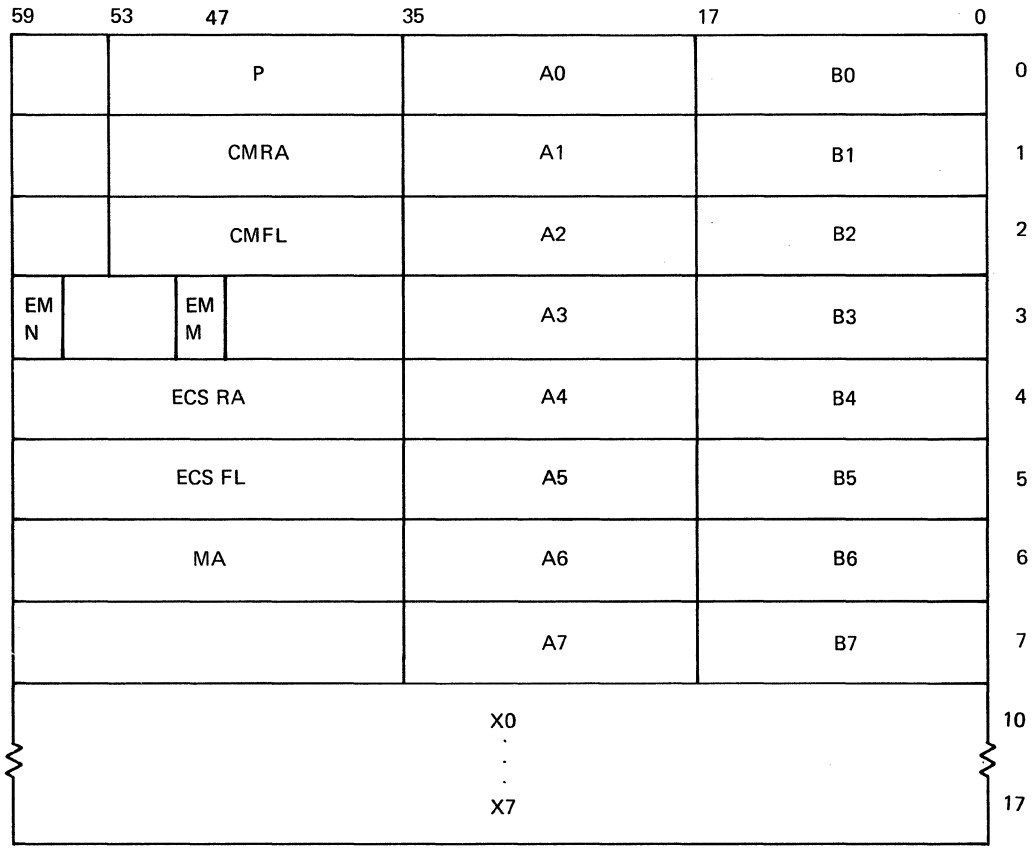
Channel Number	Channel Symbol	Description
00-13		Hardware channels.
14	CH.FST	Controls access to FST.
15	CH.FNT	Controls access to FNT.
16	CH.DDT	DDT interlock.
17	CH.RBT	Controls access to RBT.
20-33		Hardware channels.
34	CH.CPA	Control point area interlock.
35	CH.PFM and CH.APF	Permanent file manager and APF table channel.
36	CH.INS	Reserved for installation.
37	CH.MST	MST interlock.
40	CH.EST = CH.TAPE	Controls access to EST/TAPES table.
41	CH.ICOM	INTERCOM-NOS/BE communication interlock or RMS error buffer interlock.
42	CH.IEMBF	INTERCOM empty buffer channel.

Channel Number	Channel Symbol	Description
43	CH.IUSER	INTERCOM user table channel.
44	CH.SCH	Scheduler channel.
45	CH.IHUSR	Reserved.
46	CH.IHSMT	Reserved.

PP STATUS WORDS

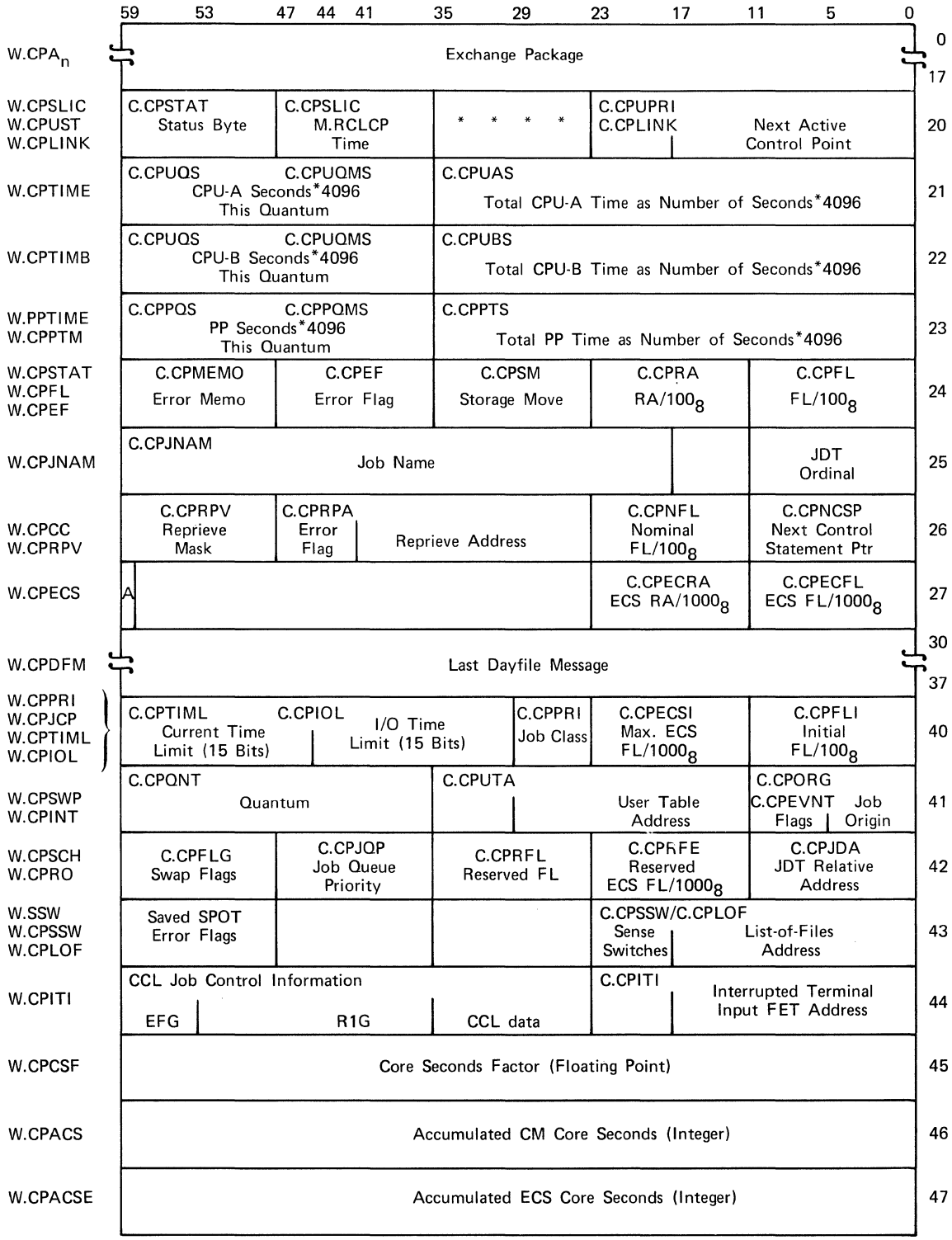


EXCHANGE PACKAGE FOR CONTROL POINT AREA (WORDS 0 - 17)

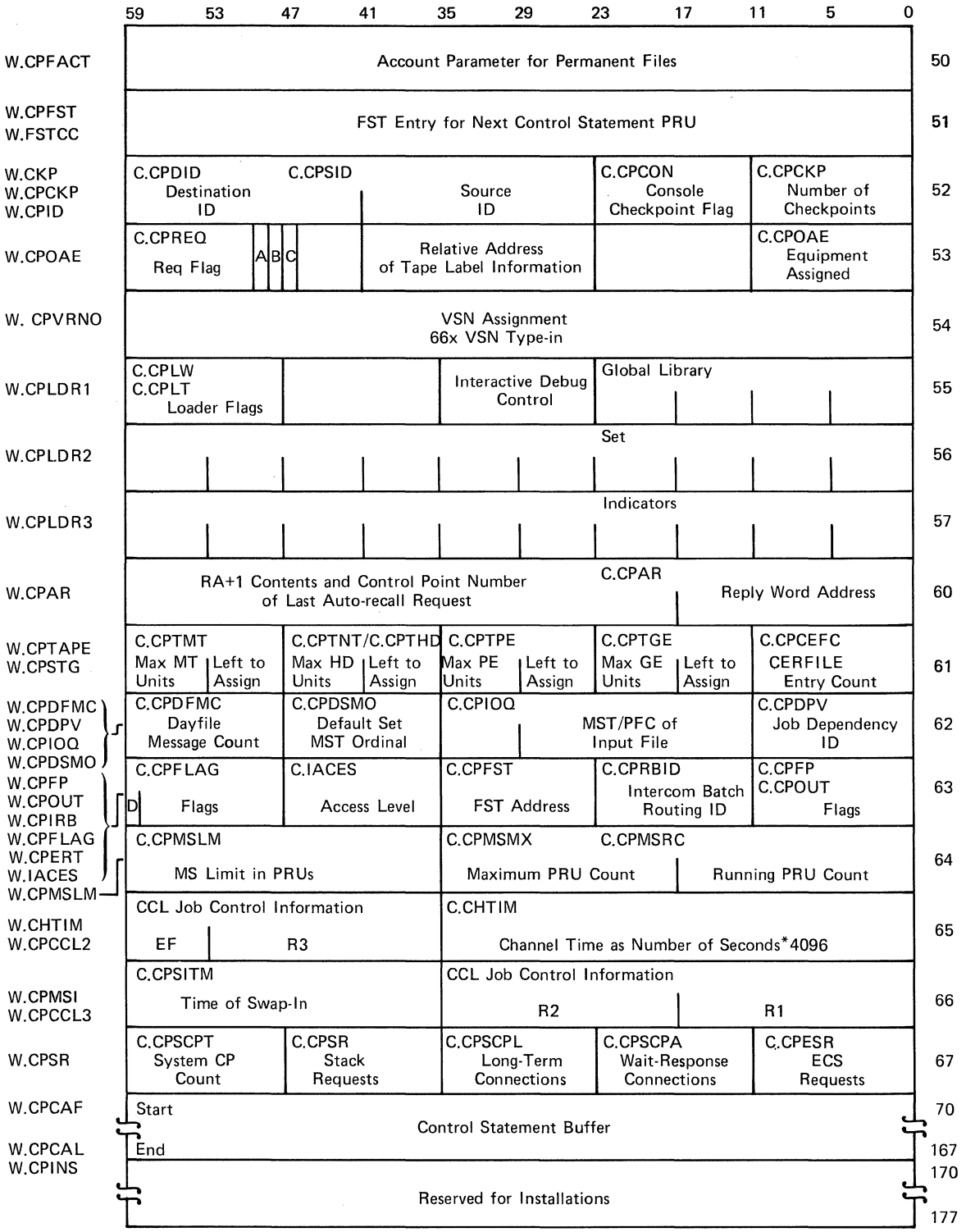


Field Name	Description
EM	Exit mode bits.
	N For CYBER 170 only.
	M For CYBER 170, CYBER 70, and 6000.

CONTROL POINT AREA



A If set, no update to exchange package ECS RA and FL.



A S.YNRDY
B S.YNNO
C Extended label format.
D Do not update user's running PRU count.

W.CPLINK(20) C.CPSTAT

Bit	Field	Description
0	S.CPUSTM	Move flag; move in progress.
1	S.CPUSTY	Auto recall.
2	S.CPUSTA	CPU-A assigned only.
3	S.CPUSTB	CPU-B assigned only.
4	S.CPUSTX	Recall status.
5	S.CPUSTW	Wait status.
6	S.CPUSTR	Real time job.
7	S.CPUSTC	Active CPU-A.
8	S.CPUSTD	Active CPU-B.
9	S.CPUSTS	Control point activity suspended.
10	S.CPUSTP	Suspended by checkpoint.
11	S.CPUSTZ	Suspended by MTR (too many PP calls).

W.CPEF(24) C.CPEF

Value	Field	Description
0001	F.ERTL	CP time limit exceeded; sensed by MTR.
0002	F.ERAR	Arithmetic error; sensed by MTR.
0003	F.ERPP	PP abort (M.ABORT); requested by PP.
0004	F.ERCP	CPU abort (ABT in RA+1); requested by program.
0005	F.ERPCE	PP call error (garbage in RA+1) abort; sensed by MTR.
0006	F.EROD	Operator drop.
0007	F.ERK	Operator kill.
0010	F.ERRN	Operator rerun (batch job only).
0011	F.EREX	Control statement error; set by 1AJ.
	F.ERCC	Control statement error for INTERCOM job.
0012	F.ERECP	ECS parity error; sensed by MTR.
0013	F.ERJC	Job statement error.
0014	F.ERPA	Preabort (batch job only).

Value	Field	Description
0015	F.ERRCL	Auto recall error; bad PP call.
0016	F.ERHANG	Job hung in auto recall.
0017	F.ERMSL	Mass storage limit exceeded by stack processor (batch job only).
0020	F.EROVL	PP overlay not in PP LIB.
0021	F.ERIOL	I/O time limit exceeded; sensed by MTR.
0022	F.ERRMS	Dayfile lost on idled device.
0040	F.ERTI	Terminal interrupt by user.
0061	F.ERPARF	Swap-in parity error for graphics.
-77(7700)	F.ERMEMO	Enter MEMO mode.
-0(7777)	F.ERTMM	Terminate MEMO mode.

W.CPRPV(26)

Bit	Description
41	If nonzero, no checksum is taken.
58	If nonzero, extended RPV is selected.
59	If nonzero, user's reprieve routine is active.

W.CPSWP(41) C.CPORG

Bit	Description
5-0	Job origin. 4 Real-time. 10 Graphics. 20 Multiuser. 40 INTERCOM.
6	Swap-out event bit.

W.CPSCH(42) C.CPFLG

Bit	Field	Description
51	S.CP1IB	1IB operating at control point.
52	S.CPFFL	FNTs in positive FL.
53	S.CPEOJ	End of job.
54	S.CPCLR	Control point area clear request.
55	S.CPRFL	Storage request.
56	S.CPROP	Roll-out in progress.
57	S.CPS1P	Swap-in in progress.
58	S.CPSOP	Swap-out in progress.
59	S.CPSWC	Swap-out complete.

W.CPSCH(42) C.CPJDA Values

Value	Description
0-7776	FWA JDT entry (relative to FWA JDT).
7777	No JDT entry assigned.

W.CPFACT(50)

Bit	Description
11-0	If nonzero, permanent file accounting messages are issued.
59-12	Left-justified account number.

W.CPCKP(52) C.CPCON

Bit	Description
0	Console checkpoint request.

W.CPLDR1(55) C.CPLW

Bit	Field	Description
48		Reserved for installations.
49	S.CPLP	Program loaded from nonsystem library.
50		Reserved.

Bit	Field	Description
51	S.CPLRE	Reduce ECS flag.
52	S.CPLT	Debugging aid flag.
53	S.CPLR	Reduce flag.
57-54	S.CPLM	Map options.
58		Reserved.
59	S.CPLV	Map options validity flag.

W.CPLDR1(55) Interactive Debug Control

Bit	Description
24	Reserved.
25	FTN.
31-26	Reserved.
33-32	Reserved for installations.
34	Copy PIDL, symbol, and line number tables; generate block tables and entry point tables.
35	Load and pass control to the interactive debugger.

W.CPLDR1(55)
W.CPLDR2(56) } Global Library Set Indicators
W.CPLDR3(57)

Value	Significance
00	End of global library set.
01-76	LNT ordinal of system library.
77	File name of first user library in W.CPLDR3; file name of second user library in W.CPLDR2.

W.CPTAPE(61) C.CPTMT(0)=C.CPTHD(1)=C.CPTPE(2)=C.CPTGE(3)

Bit	Description
11-6	Maximum number of tape units of a specific type (MT, HD, PE, or GE) that will be assigned to the control point for a specific job. This value is the number requested on the job statement.
5-0	Number of tape units of a specific type (MT, HD, PE, or GE) left to be assigned to the control point. This value is the maximum number requested minus the number which have been assigned.

W.CPFLAG(63) C.CPFLAG

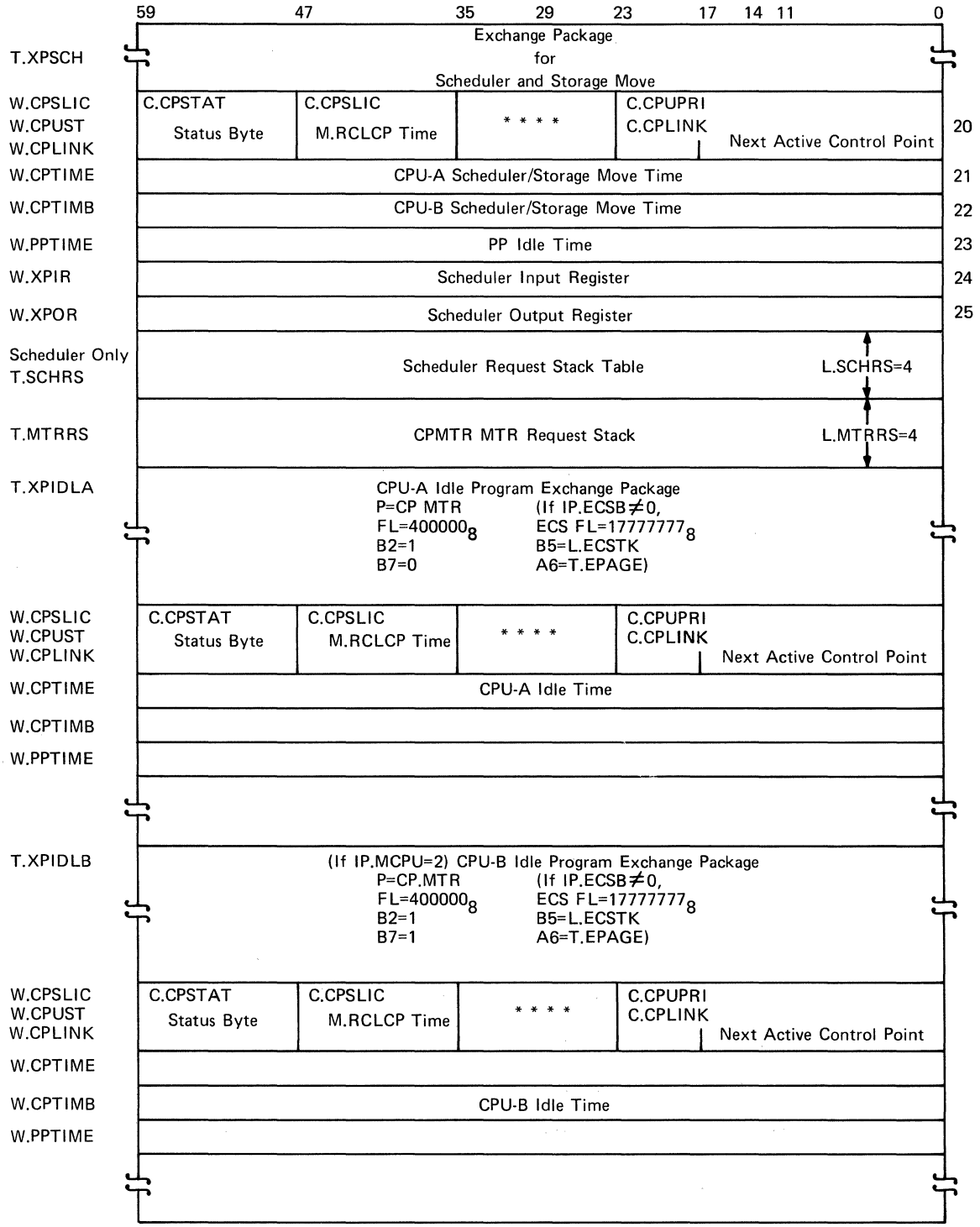
Bit	Field	Description
0	S.CPLDAF	MDI interlock.
1		Reserved.
2	S.CPRK	Previously reprieved after operator KILL.
3	S.CPNFNT	Do not search FNT.
4	S.IOL	I/O time limit previously set.
5	S.CPL	CP time limit previously set.
6	S.MSL	MS limit previously set.
7	S.CPXTS	Look for next EXIT(S) statement.
8	S.CPDMPX	Give no DMPX.
9	S.CPCMM	CMM active flag.
10	S.CPCVL	CVL reserved EST entry for maintenance.
11		Reserved.

W.CPFP(63) C.CPFP

Bit	Field	Description
0	S.CPL	Reprocess.
1	S.CPG	Abort.
2	S.CPA	No rerun.
3	S.CPS	Sequencer.
4	S.CPN	Checkpoint taken.
5	S.CPX	Look for EXIT statement.
6		Unused.
7	S.CPEOR	Control statement EOR.
8	S.CPJFL	Job statement field length assigned.
9	S.CPJ	JANUS.
10		Unused.
11	S.CPE	EXPORT.

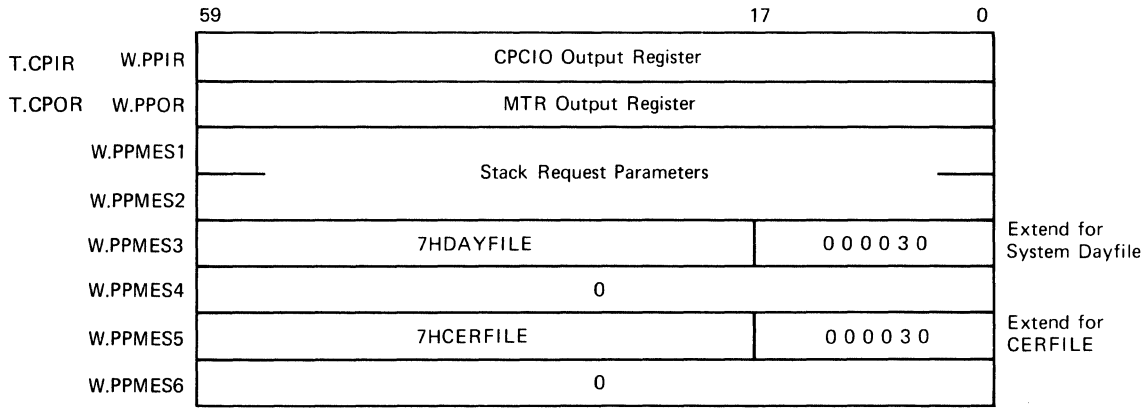
SYSTEM JOB EXCHANGE PACKAGE AREA

Byte 1 of word 60 in the CMR pointer area contains the address of T.XPSCH/10_8. Bits 59 through 43 of word 40 in the CMR pointer area contain the address of T.XPIDLA. Bits 59 through 43 of word 45 in the CMR pointer area contain the address of T.XPIDLB.

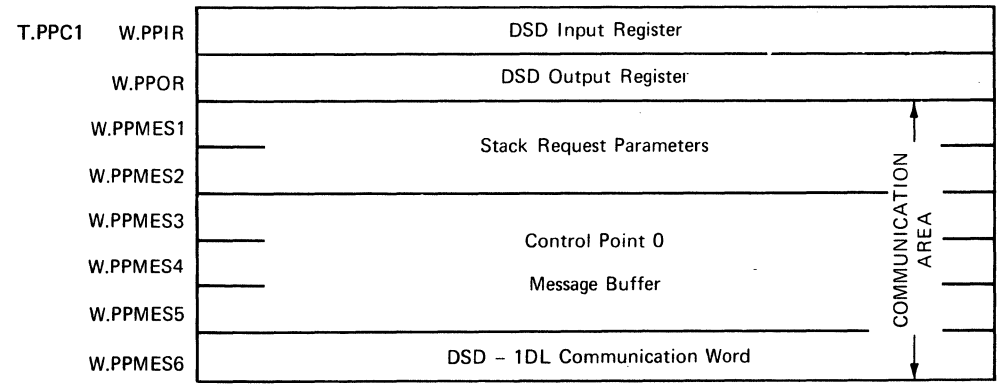


PP COMMUNICATION AREA

FOR PP0

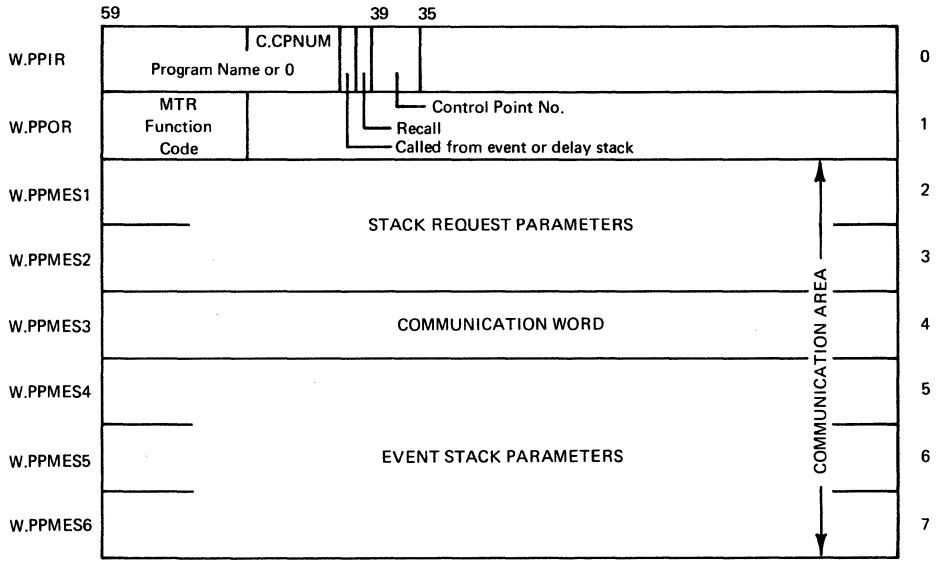


FOR PP1

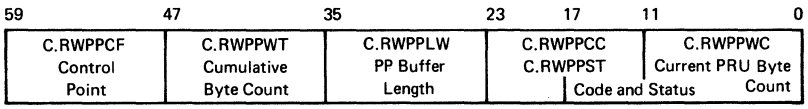


Byte 4 of word 5 in the CMR pointer area contains the address of T.PPC1.

FOR PP2 THROUGH PPn



COMMUNICATION WORD



PP PROGRAM NAME RESERVATIONS

Routine Name	Description
A	Stack processor segment; zero-filled.
ABC	Buffer controller coldstart bootstrap.
ABS	Dump CM (absolute address).
ABT	Program abort.
ACE	Advance control statement.
ACT	Helper for program ACCOUNT.
ADS	ADDSET processor; add member to device set.
AEI	Utility display routine (CTI).
APR	Automatic program sequencer.

Routine Name	Description
CD4	844 disk driver (CTI).
CD6	66x tape driver (CTI).
CD7	67x tape driver (CTI).
CD8	885 disk driver (CTI).
CED	Deadstart PP control program.
CEE	Second part of deadstart PP control program.
CEF	Third part of deadstart PP control program.
CEM	Central error manager for ECS.
CEY	MTS coldstart bootstrap.
CIO	Circular I/O processor.
CKP	Save information necessary to restart a checkpoint job.
CLO	Dummy program used to call CIO.
CMC	Check computer memory (CTI).
COM	Deadstart option matrix.
CON	INTERCOM connect file to remote terminal.
CP1	C.E. 415 card punch test.
CR1	C.E. 405 card reader test.
CT8	Reserved for C.E. diagnostics.
CVL	C.E. diagnostic validation routine.
CY1	Reset FNT of the file being processed by restart.
DDR	Deadstart 885 or 844 disk driver.
DEM	Dump extended memory (CTI).
DF4	C.E. 3234 test.
DF7	C.E. 3553 test.
DF8	C.E. 808 test.
DHE	Determine hardware characteristics (CTI).
DIS	Console display program for a control point.
DLE	C.E. diagnostics.
DLL	Downline load controlware program.

Routine Name	Description
DLM	DELSET processor; delete member from PF set.
DMP	Dump CM.
DPC	Dump peripheral controllers (CTI).
DSD	System display.
DSM	Dismount pack.
DSP	ROUTE/DISPOSE function processor.
DTS	Deadstart dump for 66x and 67x drivers.
D00	Diagnostic for COBOL.
D44	C.E. 844 test.
EBL	External bootstrap loader (CTI).
EDD	Express deadstart dump to tape (CTI).
EDT	Express dump tape driver (CTI).
EHS	Export initialization.
END	Normal termination.
EPF	Send audit information to CM.
FAD	INTERCOM file attach/detach.
FDP	881/883 pack formatting driver.
FIN	CCL function processor.
FNT	INTERCOM FNT alter routine.
FSN	Find set name.
FS8	Reserved for C.E. diagnostics.
FTP	C.E. 580 printer test.
GBJ	INTERCOM 274 Graphics begin job.
GCC	Graphics class change program.
GEJ	INTERCOM 274 Graphics end job.
GES	INTERCOM 274 IGS SIGNON service program.
GPF	GETPF (multimainframe).

Routine Name	Description
HDS	Help deadstart.
IAP	INTERCOM initiate another program.
ICD	Install CTI on disk routine (CTI).
ICE	Second part of CTI installation routine (CTI).
IEF	Routine for CEFAP.
IOQ	Initial operator queries (CTI).
IPL	Initial program loader (CTI).
IPP	INTERCOM password protection.
IRP	Deadstart RMS stack processor.
IUP	INTERCOM initiate user program.
JAC	Job queue acquire information.
JDP	Job dependency count decrementor.
LBK	C.E. load buffer controller.
LBL	LABELMS header.
LCD	INTERCOM LCC/2550 dump.
LDC	LDCMR utility helper.
LDD	Load capsule directory.
LDL	Loader utility program.
LDQ	FDL quick loader.
LDV	Load CPU absolute overlays.
LDW	Load CPU absolute overlays in conjunction with LDV.
LFP	Deadstart first level peripheral processor (PPU) loader.
LIF	Lower CYBER interactive interface.
LOC	Load octal corrections.
LPF	In conjunction with LOADPF, reload permanent files.
MAC	INTERCOM multiuser job accounting.
MAD	Mainframe attribute determiner (CTI).

Routine Name	Description
MDI	Used by EDITLIB to handle I/O involved in changing and moving directory.
MDR	Deadstart 66x and 67x driver.
MEM	Process memory function.
MES	INTERCOM write messages to remote terminal.
MLD	C.E. MALET maintenance language driver.
MNT	MOUNT processor.
MSD	Direct access module of record manager.
MSG	Issue dayfile messages.
MTR	Monitor.
MTZ	C.E. 66x tape drive test.
MUJ	INTERCOM multiuser job.
M71	C.E. 6671/6676 driver.
NSV	PP helper for CPVSN processor.
OIP	Operator intervention processor (CTI).
OPE	Dummy program used to call CIO.
ORD	C.E. diagnostic.
OSB	Operating system bootstrap routine.
OUX	TRANSPF and DUMPF utility helper.
PAK	Disk pack management routine.
PCM	Preset computer memory (CTI).
PFA	Permanent file manager ATTACH function.
PFC	Permanent file manager CATALOG function.
PFD	Attaches permanent file directory to control point.
PFE	Permanent file manager EXTEND function.
PFP	Permanent file manager PURGE function.
PFR	Permanent file manager RENAME function.
PFS	Permanent file manager POSITION function.

Routine Name	Description
PPI	Reserved.
PRM	Permission checking function.
QAC	I/O queue acquire file.
QAF	Queue access function.
QAJ	Reserved.
RCL	Temporarily relinquish CPU.
REQ	Make nonallocatable device assignment and format FNT entries for allocatable devices in response to REQUEST control statement or a REQUEST macro call.
RMS	Routine for CERMS.
RPV	Reprieve central program.
RST	Restore control point area of restart job.
RWE	Check for INTERCOM job.
SAC	ECS segment activity count.
SAD	Select alternate deadstart routine (CTI).
SBP	C.E. MALET stand-by PP used to disconnect a hung channel.
SCE	SCR error processor (CTI).
SLT	Reserved.
SPF	SAVEPF (multimainframe).
SPY	Count P-register samples for CP programs.
SRB	Used by EDITLIB to complete the disk address of a record.
SSC	Subsystem call.
SSF	Subsystem function.
SSH	Station system helper.
STD	Enhanced station channel coupler driver.
STF	Copy central memory into user buffer area.
STL	Deadstart system execution PP resident.
STR	Reserved.
STS	Used by CP program to obtain certain status.

Routine Name	Description
TAT	PF set table system access.
TBL	INTERCOM get table.
TDS	Terminate deadstart.
TIM	Get current time, date, and so on.
TMT	Table maintenance helper.
T6X	C.E. 66x tape drive test.
T7X	C.E. 67x tape drive test.
T76	INTERCOM interface to station control point.
Uxx	Reserved for installations.
VEJ	Verify job statement.
VSM	STIMULATOR routine.
XDQ	PP portion of dump queue.
ZZZ	Termination record (CTI).
nUx	Reserved for installations.
0DA-0DZ	INTERCOM 5 2550 load/dump modules.
0D0-0D9	INTERCOM 5 2550 load/dump modules.
0FA-0FZ	INTERCOM 4 2550 initializer.
0F0-0F9	INTERCOM 4 2550 load/dump modules.
0ND	INTERCOM 5 driver overlay.
0ZA-0ZS	INTERCOM 4 LCC drivers.
0ZT-0ZZ	INTERCOM 4 LCC initializer.
0Z1-0Z9	INTERCOM 4 LCC drivers.
1AB	Identify recovered jobs.
1AJ	Advance job.
1BO	Asynchronous job terminator.

Routine Name	Description
1BR	INTERCOM buffer manager.
1BT	Blank tape label routine.
1CC	Enhanced station CIO overlay.
1CI	INTERCOM queue manager.
1CL	Close function for all nontape or nonpermanent files.
1CR	Reserved.
1CS	Reserved.
1CT	Reserved.
1C9	Write CM for tape read recovery (nine-track system tapes).
1DD	System dynamic dump processor.
1DF	Dump dayfile.
1DI	INTERCOM 5 driver overlay loader.
1DL	Overlay loader and dayfile message processor for DSD.
1DM	Device queue manager.
1DS	INTERCOM H display.
1EJ	End of job processor.
1EV	Off line evict processor.
1FC	Create an RB entry for PFC.
1FE	INTERCOM 4 2550 driver.
1FM	Dummy film/hardcopy processor.
1GJ	INTERCOM 274 graphics.
1GM	Issue GOOD MORNING when time changes from 23.59 to 00.00.
1GR	INTERCOM 274 graphics.
1GS	INTERCOM 274 IGS SIGNON initializer.
1HS	EXPORT main overlay.
1IB	Initiate batch job from input queue.
1ID	INTERCOM send dayfile message to terminal, complete swap.
1IM	INTERCOM send message to terminal.

Routine Name	Description
1IQ	Initiate JANUS control point.
1IR	Main JANUS routine; drives readers, punches, printers, and so on.
1IS	Initialize overlay setup.
1IT	Integrated tape driver (66x/67x) for main overlay.
1IU	Called by JANUS to backspace print file.
1I1	INTERCOM initialization.
1LC	Load tape controller conversion tables.
1LX	INTERCOM 4 export processor.
1MF	Reserved.
1MH	Tape scheduling/prescheduling routine.
1MM	Multimainframe job queue manager.
1MT	Reserved.
1ND	INTERCOM 5 255x driver.
1NI	INTERCOM 5 255x initializer.
1NO	Reserved.
1NP	INTERCOM 5 export processor.
1NR	Reserved.
1NS	Notify station of SPOT completion.
1NW	Reserved.
1N2	Reserved.
1N3	Reserved.
1OP	File open routine for nontape files.
1PC	Close permanent file mass storage.
1PD	Called by PFA to enter event stack, call another PP routine, or swap out.
1PF	Permanent file error recovery.
1PG	PURGE (multimainframe).
1PK	PF set coordinator.
1PL	Dummy plot program.

Routine Name	Description
1P1	Reserved.
1P2	Reserved.
1P3	Reserved.
1P4	Reserved.
1QF	I/O file manager.
1QM	INTERCOM check for MUJ swap-out completion.
1QP	INTERCOM quantum calculator and MUJ servicer.
1RC	Restore field length of a checkpointed job.
1RN	Age queues, manage RBT chains, and check status of tape drives.
1RP	Reserved.
1RS	Reserved.
1RT	Reserved.
1RV	Reserved.
1R2	Reserved.
1R3	Reserved.
1R9	Reserved.
1SC	Record status/control register errors in CERFILE.
1SI	Routine to swap in or roll in a job.
1SO	Swap out or roll out a job.
1SP	Mass storage I/O stack processor for 7054 and 7154 controllers.
1SQ	Mass storage I/O stack processor for 7155 controller.
1SX	Error message and abort function for stack processors.
1S5	Load and execute 1SP or 3DO at second entry.
1TF	Reserved.
1TJ	Translate job statement.
1TO	Reserved.
1TR	Read labels for non-66x drives.
1TS	Tape sampler; contains 2TACOM.
1VG	STIMULATOR routine.

Routine Name	Description
1WB	INTERCOM 4 wideband driver.
1WI	Reserved.
1W5	Reserved.
1W9	Reserved.
1XG	INTERCOM 4 1XP overlay used for graphics.
1XP-6XP	INTERCOM 4 high-speed EXPORT processor.
1ZA-1ZP	INTERCOM 4 drivers.
2CC	1CI overlay; process command.
2CS	1CI overlay; status management.
2CU	1CI overlay; create user table.
2FC	1FC overlay; replace mode.
2FE	INTERCOM 4 2550 driver.
2GJ	INTERCOM 274 graphics.
2IA	66x/67x read driver for L tapes.
2IB	66x/67x write driver for L tapes.
2IC	66x/67x read driver for seven-track coded system tapes.
2ID	66x/67x write driver for seven-track coded system tapes.
2IL	66x/67x labels and tape module.
2IO	Submodule for 3IO, 3IL.
2IP	66x/67x tape positioning.
2IR	66x basic read overlay.
2IS	Reservoir of routines for 1IS.
2IT	67x basic read overlay.
2IW	66x basic write overlay.
2IX	67x basic write overlay.
2I1	INTERCOM overlay to 1I1.
2LF	LPF overlay broken connect.
2MN	MNT overlay error processing.

Routine Name	Description
2ND	INTERCOM 5 driver input command processor.
2NI	INTERCOM 5 initializer overlay.
2NP	INTERCOM 5 batch command processor.
2PA	PFA utility processor.
2PK	1PK overlay.
2QF	1QF overlay; recover queue files.
2RN	1RN overlay; check available disk space.
2RP	Reserved.
2R2	Reserved.
2ST	Multimainframe CIO staging processor.
2TB	Reserved.
2TC	Reserved.
2VJ	Translate job statement.
2WB	INTERCOM 4 overlay to 1WB.
2XP	INTERCOM 4 wideband directive processor.
3AM	ADS overlay member processing.
3DO	Initialize allocatable device file.
3FE	INTERCOM 4 2550 driver.
3IC	66x/67x close processor.
3IE	66x/67x basic error processor.
3IF	66x/67x multifile processor.
3II	66x/67x system initialization.
3IJ	66x/67x system call processor.
3IL	66x/67x label write processor.
3IM	66x/67x message processor.
3IN	VSN message processor.
3IO	66x/67x open processor.
3IP	66x/67x positioning within a logical file.

Routine Name	Description
3IR	66x/67x read error recovery.
3IS	JANUS overlay.
3IV	66x/67x close volume processor.
3IW	66x/67x write error recovery.
3LF	LPF overlay; abnormal termination.
3LX	INTERCOM 4 overlay to 1LX.
3MN	REQ overlay containing 2TACOM for tape assignments.
3MS	EXPORT overlay.
3ND	INTERCOM 5 driver input command processor.
3NI	INTERCOM 5 initializer overlay.
3NP	INTERCOM 5 batch support subroutines.
3PC	PFC helper.
3PM	Segment of 1P1 used for holding code for future use.
3PO	Segment of 1P3 that processes uncorrectable parity error GO or RECHECK code.
3PS	Segment of 1P4 used for holding code for future use.
3QF	1QF overlay; 1QF functions 4 and 5.
3RQ	REQ overlay containing 2TACOM.
3R2	Reserved.
3R3	Reserved.
3SJ	Stack processor for 885 driver.
3SW	Reserved.
3SY	Stack processor for 844-21 and 844-41 driver.
3TT	INTERCOM transmit data from CPU to terminal.
3T1-3T2	INTERCOM overlays to 3TT.
3WB	INTERCOM 4 overlay to 1WB.
3XP	INTERCOM 4 wideband batch input processor.
4AM	ADS add member overlay.
4DO	Process device independent requests for allocatable devices.

Routine Name	Description
4EJ	RERUN overlay to 1EJ to reconstruct input FNT.
4ES	Enter stack request.
4FE	INTERCOM 4 2550 driver.
4HS	EXPORT overlay.
4IS	JANUS overlay.
4LB	Reserved.
4LC	Reserved.
4LX	INTERCOM 4 overlay to 1LX.
4MN	MNT overlay; master not mounted.
4ND	INTERCOM 5 driver output command processor.
4NI	INTERCOM 5 initializer overlay.
4NP	INTERCOM 5 diverted file processor.
4PA	PFA overlay for delay/event stack and 1PF interface.
4QF	1QF overlay; 1QF functions 10 and 11.
4SD	Enhanced station channel coupler driver overlay to STD.
4SR	Reserved.
4WB	INTERCOM 4 overlay to 1WB.
4XP	INTERCOM 4 wideband batch output processor.
5CY	IRP overlay for 844-21, 844-41, and 885 drivers.
5FE	INTERCOM 4 2550 driver.
5LL	C.E. maintenance language driver (MLD) overlay for low level language.
5LX	INTERCOM 4 overlay to 1LX.
5MN	MNT overlay; normal termination.
5MU	C.E. MLD overlay.
5ND	INTERCOM 5 batch support subroutines.
5NP	INTERCOM 5 disconnect/recovery processor.
5PA	PFA segment error subroutine.
5QF	1QF overlay; 1QF functions 0 and 3.

Routine Name	Description
5WB	INTERCOM 4 overlay to 1WB.
5XP	INTERCOM 4 wideband batch banner page.
541	C.E. MLD overlay.
55X	C.E. MLD overlay.
56X	C.E. MLD overlay for 66x tape unit language.
57X	C.E. MLD overlay for 67x tape unit language.
58F	C.E. MLD overlay for 844 full-track disk language.
58H	C.E. MLD overlay for 844 half-track disk language.
58X	C.E. MLD overlay for 580 line printer language.
6BM	Billing message overlay.
6BR	Reserved.
6BW	Reserved.
6CR	Reserved.
6CW	Reserved.
6DS	INTERCOM H display overlay.
6FB	EXPORT buffer manager overlay.
6FE	INTERCOM 4 2550 driver.
6FM	EXPORT file manager overlay.
6IB	EXPORT initial block processor overlay.
6IM	Issue conflict and abort messages for REQ.
6LC	Reserved.
6LM	Segment of 4LB used to construct tape label messages.
6LX	INTERCOM 4 overlay to 1LX.
6L1	Reserved.
6L2	Reserved.
6L3	Reserved.
6L4	Reserved.
6L5	Reserved.

Routine Name	Description
6L7	Reserved.
6MD	Dummy EDITLIB overlay.
6MN	REQ overlay for tape assignments.
6ND	INTERCOM 5 batch support subroutines.
6NO	Reserved.
6NP	INTERCOM 5 batch command processor.
6PC	Restore main PFC code following 7PC call.
6PD	Write predayfile.
6PM	Permanent file accounting overlay.
6PR	Restore main PFR code following 7PR call.
6RD	Disposed file accounting overlay.
6SD	EXPORT special directive overlay.
6SF	EXPORT file search overlay.
6T1	TDS overlay for load deadstarts.
6T2	TDS overlay for recovery deadstarts.
6WM	Output dayfile error messages for I/O requests.
6XP	INTERCOM 4 wideband batch lace card.
7AJ	1AJ overlay for EXIT statement processing.
7A1-7A8	1IU overlays of 580 PFC spacing code arrays.
7CC	Station CIO overlay to 1CC.
7EC	Generate ECS buffers.
7FE	INTERCOM 4 2550 driver.
7ID	Auxiliary error processor for RMS I/O.
7ND	INTERCOM 5 transition processor and system support.
7PC	Read label of private set master device for PFC.
7PR	Read label of private set master device for PFR.
7RQ	REQ set processor.
7SF	EXPORT file search overlay.

Routine Name	Description
7SI	Process swap-in parity errors.
7T1	Reserved.
7T2	Reserved.
7W1-7W2	Overlay for 6WM.
8AA-8A9	Reserved.
8BA-8B9	Reserved.
8CA-8C9	C.E. reserved names.
8DA	A, I, J display overlay for DSD (dayfile buffers, REQUEST statements, JANUS).
8DB	B display overlay for DSD (control point status).
8DC	C, D, G display overlay for DSD (central memory).
8DD	Reserved for DSD.
8DE	E display overlay for DSD (equipment status table).
8DF	F display overlay for DSD (file name table).
8DG	Reserved for DSD.
8DH	H display for DSD (I/O queues).
8DI	Reserved for DSD.
8DJ	Reserved for DSD.
8DK	K display overlay for DSD (pointers and control point area).
8DL	L display overlay for DSD (central programmable).
8DM	M display overlay for DSD (PP communications area).
8DN	N display overlay for DSD (breakpoint).
8DO	O display overlay for DSD (operator message).
8DP	P display overlay for DSD (tapes table and VSN previewing).
8DQ	Q display overlay for DSD (INTERCOM status).
8DR	R display overlay for DSD (JDT tables and queues).
8DS	S display overlay for DSD (job control area).
8DT	T display overlay for DSD (transfer status-linked mainframe).

Routine Name	Description
8DU	U display overlay for DSD (ID table).
8DV	V display overlay for DSD (RMS devices).
8DW	W display overlay for DSD (waiting packs).
8DX	X display overlay for DSD (ECS memory).
8DY	Y display overlay for DSD (command format dictionary).
8DZ	Z display overlay for DSD (display dictionary).
8D0-8D4	DSD.
8EA-8E4	DSD (linked mainframe displays).
8FA-8FD	Reserved.
8FE	INTERCOM 4 2550 driver.
8FF-8I9	Reserved.
8GO	Loaded by 1R3 when GO or DROP operator decision necessary during tape processing.
8JA-8M9	Reserved for DSD alternate overlay names.
8NA-8NC	Reserved.
8ND	INTERCOM 5 driver terminator routines.
8NE-8PS	Reserved.
8NO	Segment to 1N3 that writes debug messages to dayfile if IP.DBUG=1.
8PA	PFA overlay to bring in RBT chain.
8PC	PFC overlay to verify SAAM files.
8PU-8SH	Reserved.
8SI	Segment to 1SI that writes debug messages to dayfile if IP.TF=0.
8SJ-8W9	Reserved.
8XA	Channel commands overlay for DSD.
8XB	Debugging commands overlay for DSD.
8XC	PP calling control points requests commands overlay for DSD.
8XD	Equipment status commands overlay for DSD.
8XE	Control point commands overlay for DSD.
8XF	Deadstart commands overlay for DSD.

Routine Name	Description
8XG	Priority and tape staging job control commands overlay for DSD.
8XH	INTERCOM commands for DSD.
8XI	Miscellaneous commands overlay for DSD.
8XJ	Miscellaneous commands overlay for DSD.
8XK	Tape scheduling commands overlay for DSD.
8XL	Operator action manager commands overlay for DSD.
8XM	Error flag commands overlay for DSD.
8XN	CP/PP interlock commands overlay for DSD.
8XO	Initiate system jobs command overlay for DSD.
8XP	Tape assignment command overlay for DSD.
8XQ	Bring up displays command overlay for DSD.
8XR	Divert a file command overlay for DSD.
8XS	Segment debug command overlay for DSD.
8XT	Segment debug command overlay for DSD.
8XU	RMS commands for DSD.
8XV	Logical ID command overlay for DSD.
8XW	ENID command overlay for DSD.
8XX-8X7	Reserved for DSD.
8X8	DSD command syntax table.
8X9	Reserved for DSD.
8YA-8Y9	DSD; linked mainframe commands.
8ZA-8Z9	INTERCOM PP drivers.
9AA-9FD	Customer engineering.
9FE	INTERCOM 4 2550 driver.
9FF-9NC	Customer engineering.
9ND	INTERCOM 5 255x driver.
9NE-9PS	Customer engineering.
9PU-9Y9	Customer engineering.
9ZA-9Z9	INTERCOM 4 drivers.

MONITOR FUNCTIONS

CPMTR Functions

01	M.SETST	Set CPU status bits.
02	M.CLRST	Clear CPU status bits.
03	M.RCP	Request central processor.
04	M.DCP	Drop central processor.
05	M.RCLCP	Recall central processor.
06	M.ICE/M.SPM	Initiate central executive/SPM call from 1SP. The following subfunctions can be performed.

00	EX.SS	System second calculation.
01	EX.SPM	Call stack processor manager.
02	EX.SPRCL	Stack processor recall.
03	EX.STAT	Change status.
04	EX.NXTPB	Get next PB/PRU.
05	EX.TAT	Lock RBR/RBT processing.
06	EX.RBT	PRU conversion.
07	EX.SIDLE	Initiate system IDLE mode.
10	EX.SSF	Subsystem function.
11	EX.DAM	Initiate DAM processing.
12	EX.BOOT	Start ECS system.
13	EX.REQEB	Request ECS buffer.
14	EX.RELEB	Release ECS buffer.
15	EX.CBM	Circular buffer manager.
16	EX.FLHB	Flush buffer.
17	EX.CSWAP	Clean ECS after ECS RPE in swap file.
20	EX.AUTEB	Terminate automatic allocation.
21	EX.ECD	Display ECS.
22	EX.ECR	Release display.
23	EX.ECW	Modify ECS.

		24	EX.CEM	Clear CEM working flag.
		25	EX.ECLDV	Make successive partial reads of ECS record.
		26	EX.BKSPF	Release data in ECS from input buffer.
		27	EX.LNKON	Restart ECS link driver.
		30	EX.LNKIN	Initialize MMF ECS link driver.
		31	EX.SUB	819 subsystem function.†
		32	EX.PPIO	819 PP I/O function.
		33	EX.ECE	ECS error.
07	M.CPUST	Change CPU status (IP.MCPU ≠ 1).		
10	M.SLICE	MTR interrupts CPMTR at end of time slice for job.		
11	M.SPRCL	Stack processor recall.		
12	M.RCH	Reserve channel.		
13	M.SEF	Set error flag.		
14	M.PPLIB	PP library search.		
15	M.SCH	Initiate integrated scheduler.		
16	M.RBTSTO	Request RBT storage.		
17	M.RSTOR	Request storage.		
20		Not used.		

PPMTR Functions

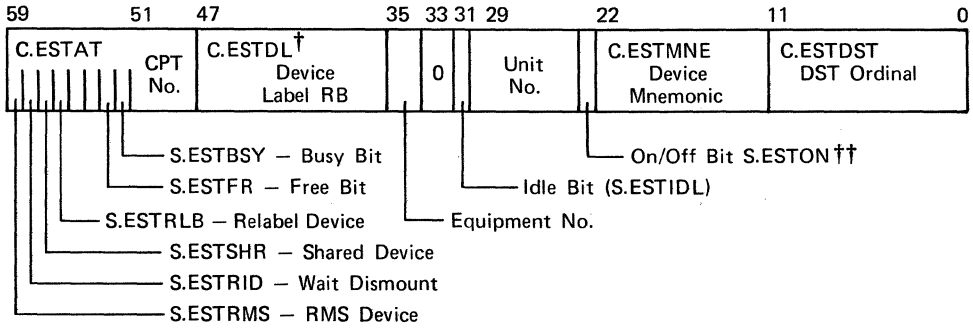
21	M.DPP	Drop PP.
22	M.ABORT	Abort control point and drop PP.
23	M.RPJD	Request peripheral job and drop PP.
24	M.EESD	Enter event stack and drop PP.
25	M.SEQ	Assign job sequence number.
26	M.MFLA	Monitor field length access.
27	M.ISP	Initiate stack processor.
30	M.DFM	Process dayfile message.
31	M.CCPA	Change control point assignment.

† Refer to the section on Input/Output for more details on the 819 subsystem function.

32	M.RPJ	Request peripheral job.
33	M.EES	Enter event stack.
34	M.CPJ	Capture peripheral job.
35	M.TSR	Terminate storage request (IP.RTMTR ≠ 0).
36	M.PASS	Ignored by MTR; cleared by another routine.
37	M.RACT	Request control point activity.
40	M.SCB	System circular buffer surveillance.
41	M.NTIME	Enter new time limit.
42	M.NOTE	Null function that is cleared immediately; used as breakpoint.
43		Not used.
44	M.BUFPTR	Buffer pointer address.
45	M.PATCH	Enter a patch into MTR.
46	M.TRACE	Turn on MTR trace.
47	M.SLPER	XJ to other CPU.
77	M.KILL	Bad monitor request made.

EQUIPMENT STATUS TABLE (EST)

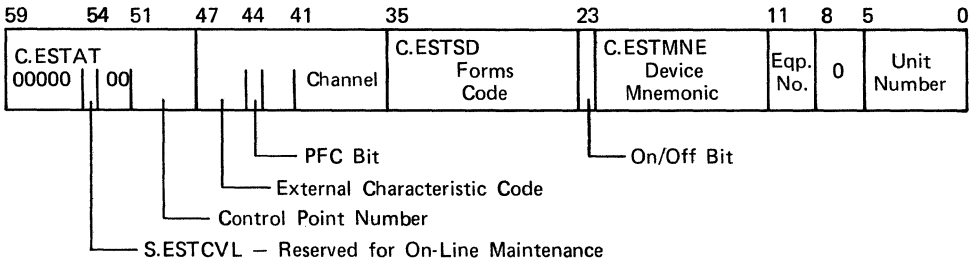
EST FOR RMS DEVICE



Byte 0 of word 5 in the CMR pointer area contains the first word address of the EST. Byte 1 of word 5 in the CMR pointer area contains the last word address plus one of the EST.

Free Bit	Busy Bit	Significance
0	0	Unavailable device.
1	0	Dismounted device.
0	1	Mounted device.
1	1	Device in process of being dismounted.

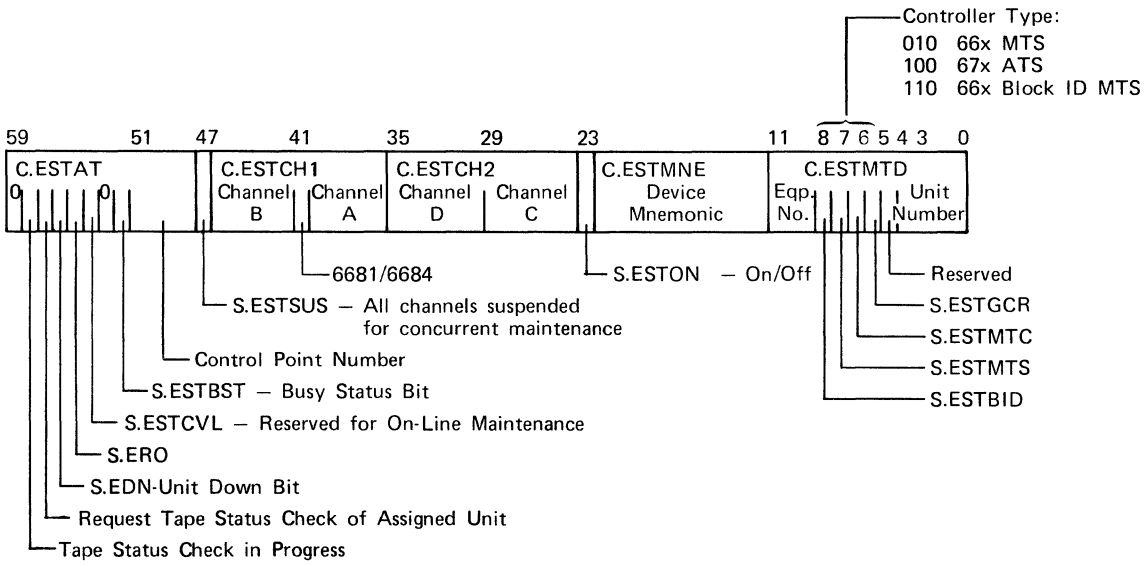
UNIT RECORD EQUIPMENT ENTRY



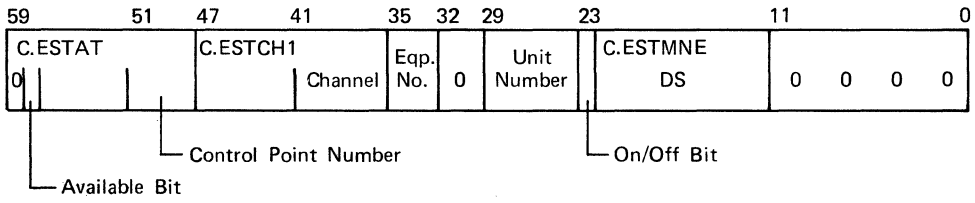
† During a level 0 deadstart, this byte holds the starting cylinder number of CTI/MSL for the preallocation routine in LABELMS.

†† A setting of 0 indicates on; 1 indicates off.

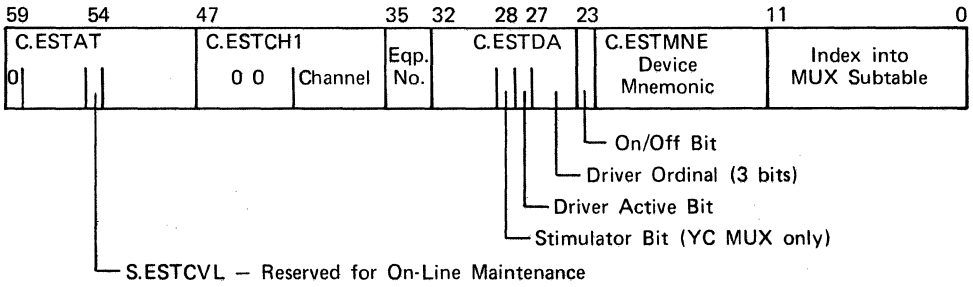
MAGNETIC TAPE ENTRY



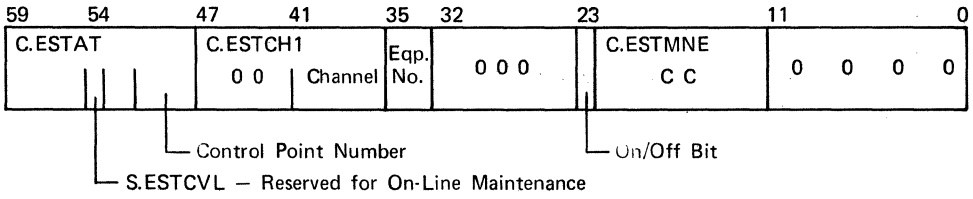
6612 DISPLAY CONSOLE ENTRY



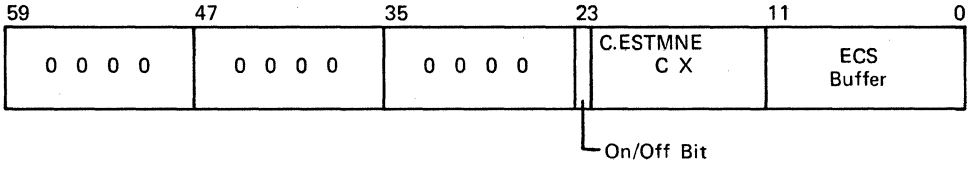
MULTIPLEXER ENTRY



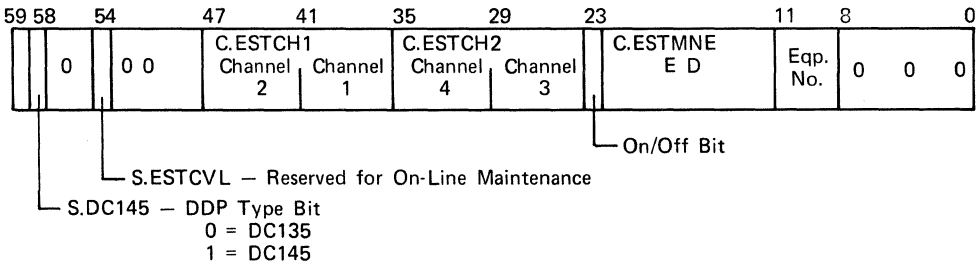
6000/7000 CHANNEL COUPLER



ECS LINK ENTRY



DDP ENTRY



DEVICE CODES

Device Mnemonic	Device Number	Description
--	01	Reserved.
--	02	
--	03	
--	04	
--	05	

Device Mnemonic	Device Number	Description
AM	06	Reserved for installations.†
--	07	Reserved.
--	10	
--	11	
--	12	
AY	13	844-21 disk drive.
AZ	14	844-41 disk drive.
AH	15	819 disk drive.
--	16	Reserved.
AJ	17	885 disk drive.
AX	20	ECS resident file (no EST entry for this device code).
--	21	Reserved.
--	22	
--	23	
--	24	
--	25	
LM	26	Link medium file.
--	27	Packet file FNT (multimainframe).
--	30	Reserved for installations; RMS devices only.
--	31	
--	32	
--	33	
--	34	
--	35	
--	36	
--	37	
MT	40 xx††	Seven-track magnetic tape.

† Device type is defined but not supported by standard software.
†† Explanation of low order 6 bits (xx) follows this listing.

Device Mnemonic	Device Number	Description
NT	41 xx	Nine-track magnetic tape.†
--	42 xx	Member file seven-track tape.†
--	43	Member file nine-track tape.†
TR	44	Paper tape reader.††
TP	45	Paper tape punch.††
--	46	Reserved for installations.
--	47	Reserved for installations.
LP	50	Any available line printer.††
LQ	52	Reserved for installations.††
LR	53	580-12 line printer.
LS	54	580-16 line printer.
LT	55	580-20 line printer.
--	56	Reserved for installations.
--	57	Reserved for installations.
CR	60	405 card reader.††
KB	61	Remote terminal keyboard.
--	62 xx	Seven-track multifile set tape.†
--	63 xx	Nine-track multifile set tape.†
--	64	Pseudo code for tape staging.
--	65	Reserved.
--	66	Reserved for installations.
--	67	Reserved for installations.
CP	70	415 card punch.††
DS	71	6612 keyboard/display console.
GC	72	252-2 graphic console.††
HC	73	253-2 hardcopy recorder.††
FE	--	255x communications NPU.

† Explanation of low order 6 bits (xx) follows this listing.
†† Device type is defined but not supported by standard software.

Device Mnemonic	Device Number	Description
FM	74	254-2 microfilm recorder.†
PL	75	Plotter.†
--	76	Reserved for installations.
--	77	Reserved for installations.
DC	--	Reserved for installations.†
IX	--	Reserved for installations.
Wx	--	Reserved for installations.
Xx	--	Reserved for installations.
SC	--	6673/6674 DSC.
YC	--	Reserved for installations.†
CC	--	6683 channel coupler.
CX	--	ECS link.
CS	--	7077-1 communications station (LCC).
ED	--	6642-1 distributive data path (DDP).

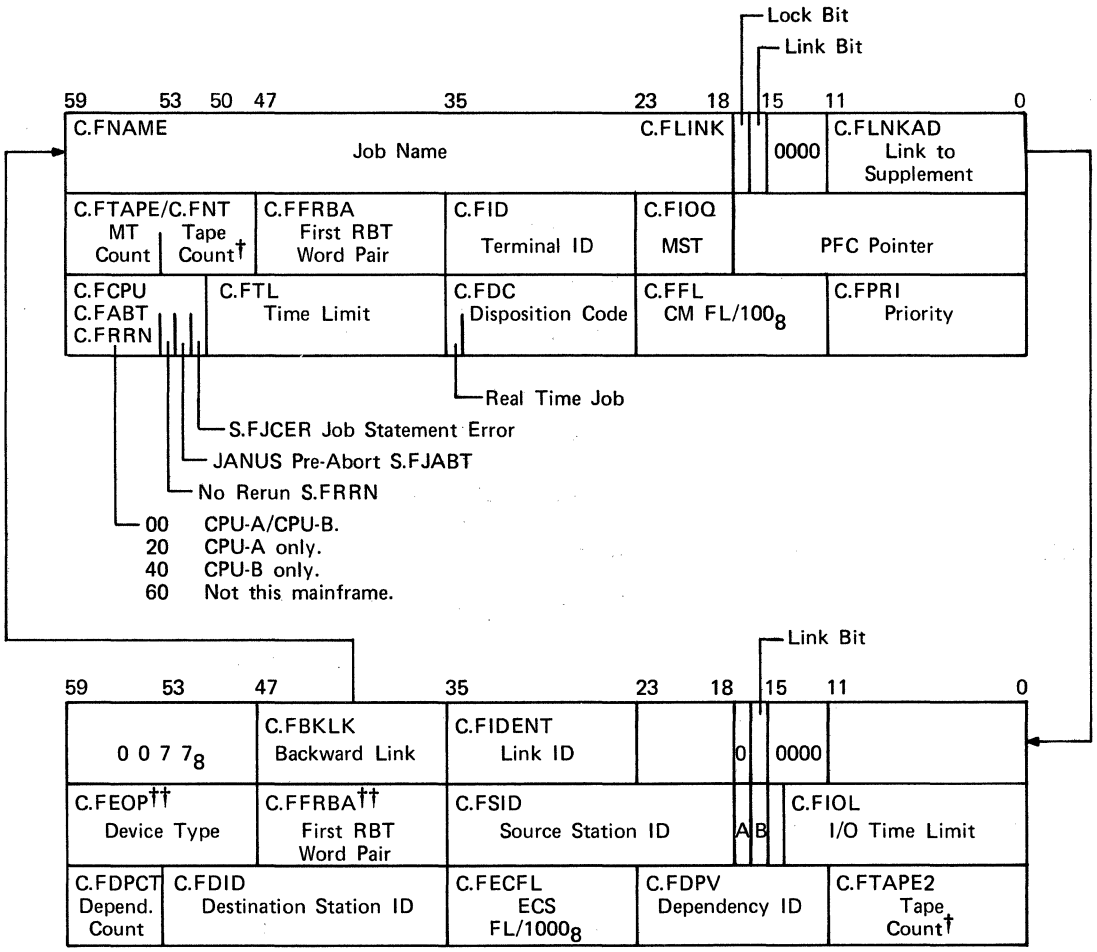
The low order 6 bits for seven-track and nine-track magnetic tape device types are as follows:

xx	Seven-track	Nine-track
----00	HI density (556 bpi).	Reserved.
----01	LO density (200 bpi).	GE density (6250 cpi).
----10	HY density (800 bpi).	HD density (800 bpi).
----11	Reserved.	PE density (1600 bpi).
--00--	Unlabeled.	Unlabeled.
--01--	U- or Z-labeled.	U- or Z-labeled.
--10--	Y-labeled.	Y-labeled.
--11--	Reserved.	Reserved.
00----	Standard data format.	Standard data format.
01----	Reserved.	Reserved.
10----	S tape.	S tape.
11----	L tape.	L tape.

† Device type is defined but not supported by standard software.

FILE NAME TABLE

ENTRY AND OPTIONAL SUPPLEMENT
FOR FILE IN INPUT QUEUE

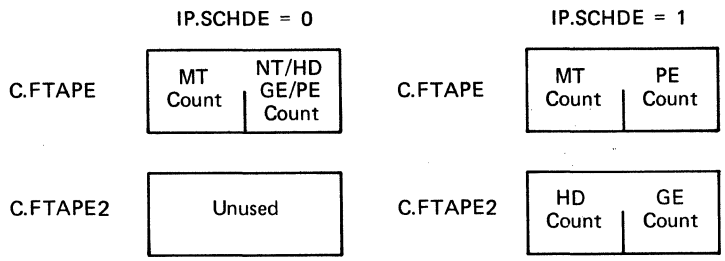


A Keep bit; indicates file is to be kept on this mainframe, not transferred to another mainframe by station or Gemini.

B Reduce ECS flag.

---

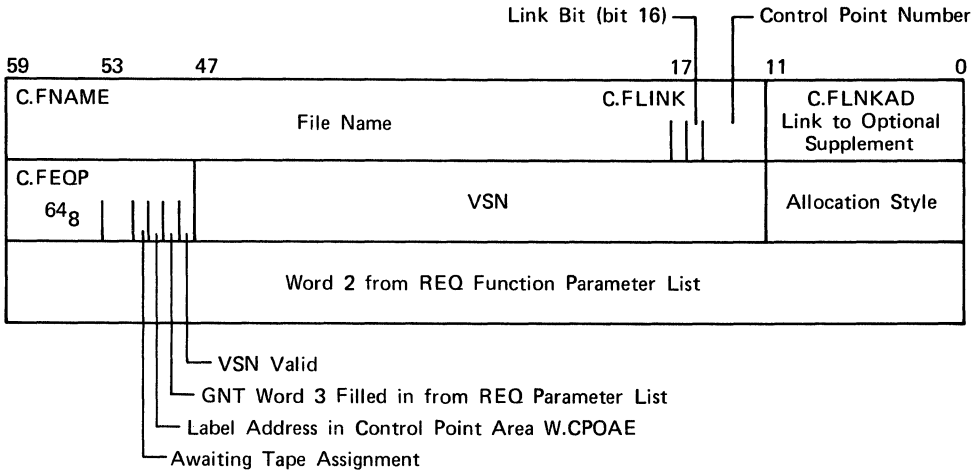
†Dependent on IP.SCHDE scheduling parameter as follows:



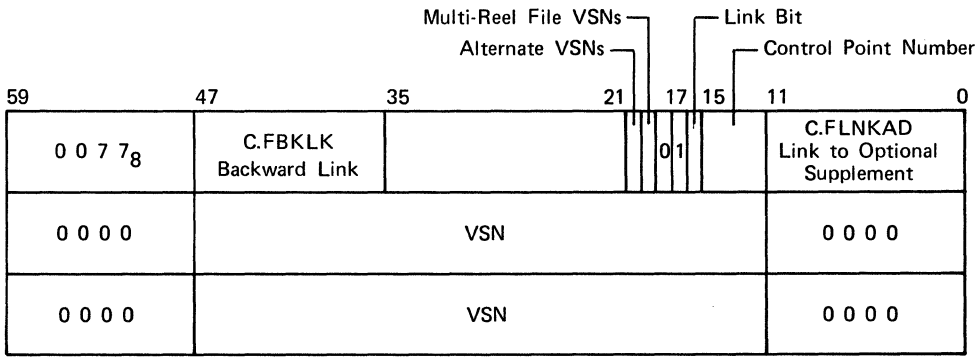
††If nonzero, fields apply to predayfile.

TAPE FILE ENTRIES

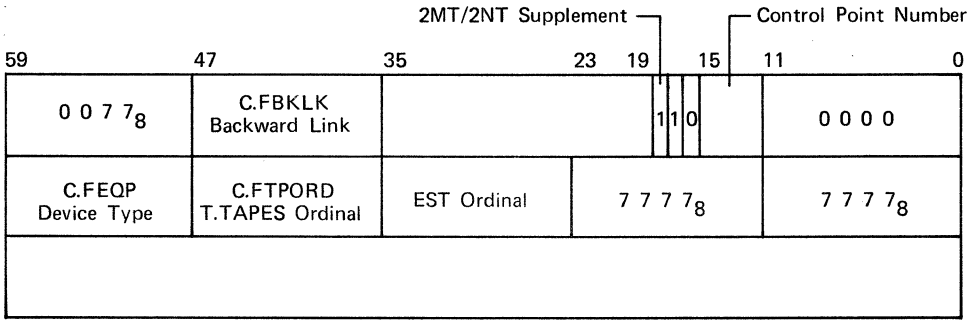
BEFORE EQUIPMENT ASSIGNMENT (GNT)



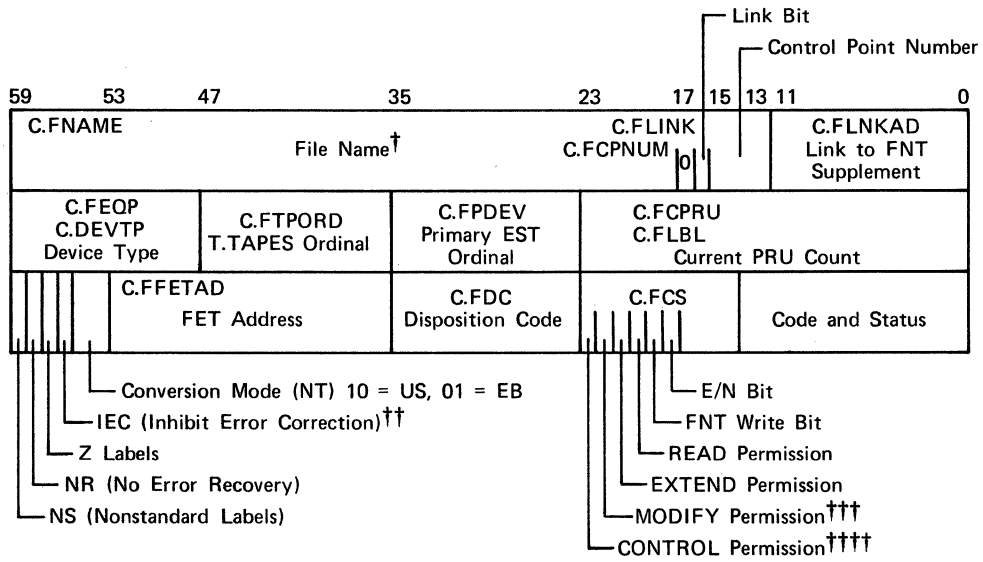
SUPPLEMENT(S) IF MORE THAN ONE VSN GIVEN



SUPPLEMENT IF 2MT/2NT DECLARED



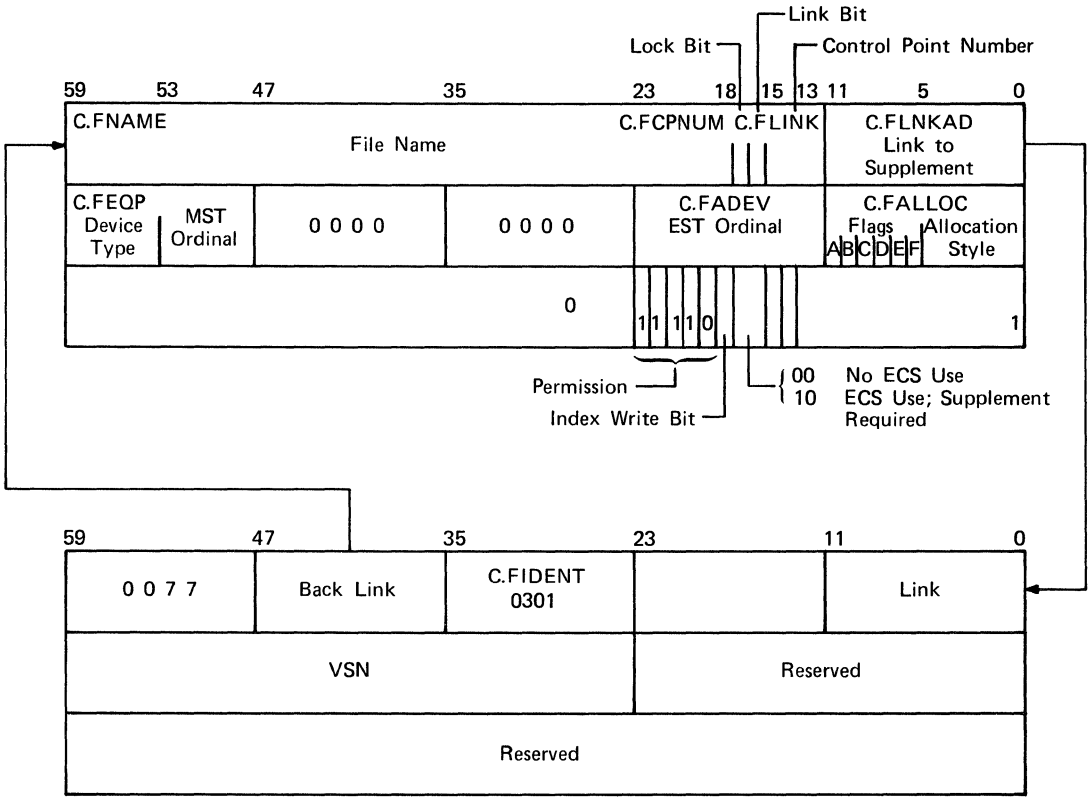
TAPE FILE ENTRY DURING PROCESSING



†Multifile name if multifile set after assignment and before first POSMF.
††Applies only to 679 tape units capable of 6250 cpi density. This density exists on models 679-5, 679-6, and 679-7 tape drives only.
†††NORING is not specified.
††††Expired tape label or operator override on unexpired tape label.

ENTRY FOR LOCAL RMS FILE

BEFORE ASSIGNMENT TO A DEVICE



A PF device.

B Overflow allowed.

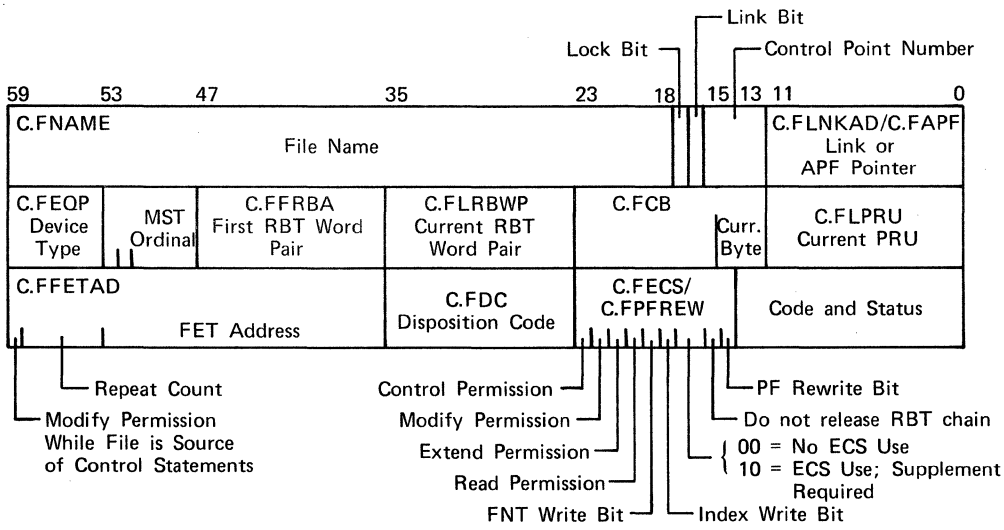
C Queue device.

D ECS buffered.

E System device.

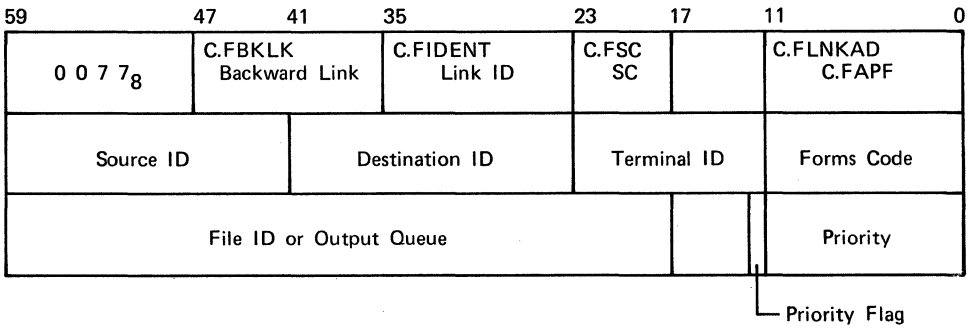
F Deferred assignment.

AFTER ASSIGNMENT TO A DEVICE AND THROUGHOUT PROCESSING

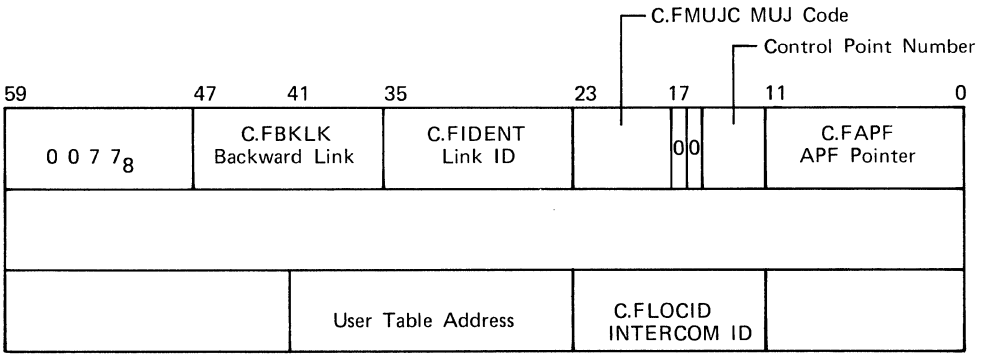


OPTIONAL SUPPLEMENTS FOR LOCAL RMS FILES

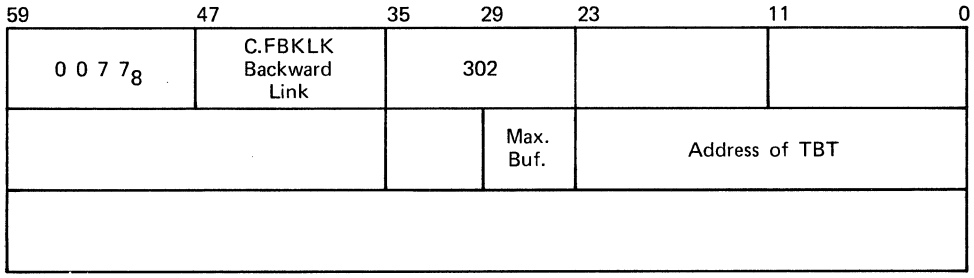
FILE ROUTING SUPPLEMENT



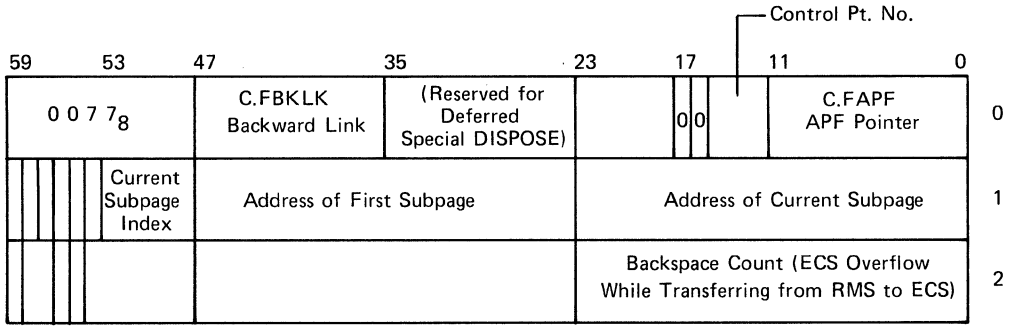
INTERCOM USER FILE SUPPLEMENT†



819 FILE SUPPLEMENT



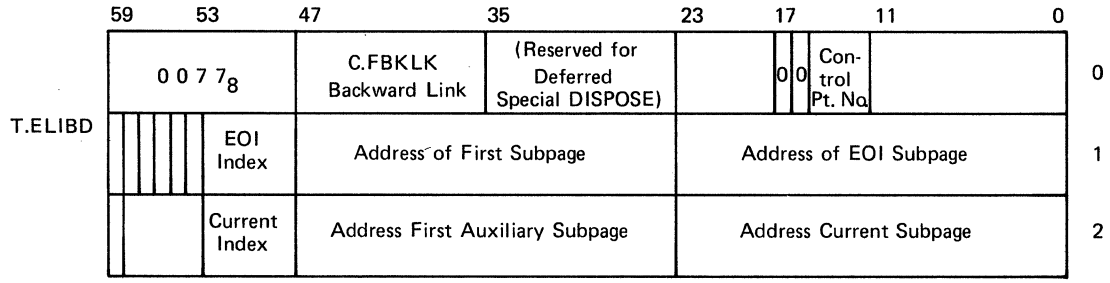
ECS FILE SUPPLEMENT (ECS resident files or I/O buffers)



†Required only when file is to be attached to a swapped-out job.

Word	Bit	Description
1	54	Outstanding PPCIO request.
	55	ECS preallocation flag.
	56	1 Release bit.
	57	0 Output buffer.
		1 Input buffer.
	58	Buffer overflow (R option).
	59	1 ECS buffered file.
2	55	Release ECS buffer after the current SR.
	56	Index written (close random file).
	59	Transfer in progress (ECS resident random files).

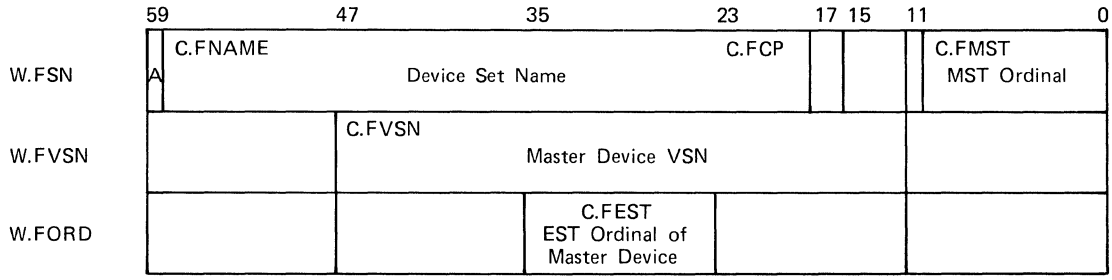
ECS FILE SUPPLEMENT (ECS Resident Library)



Byte 4 of word 77 in the CMR pointer area contains the address of T.ELIBD.

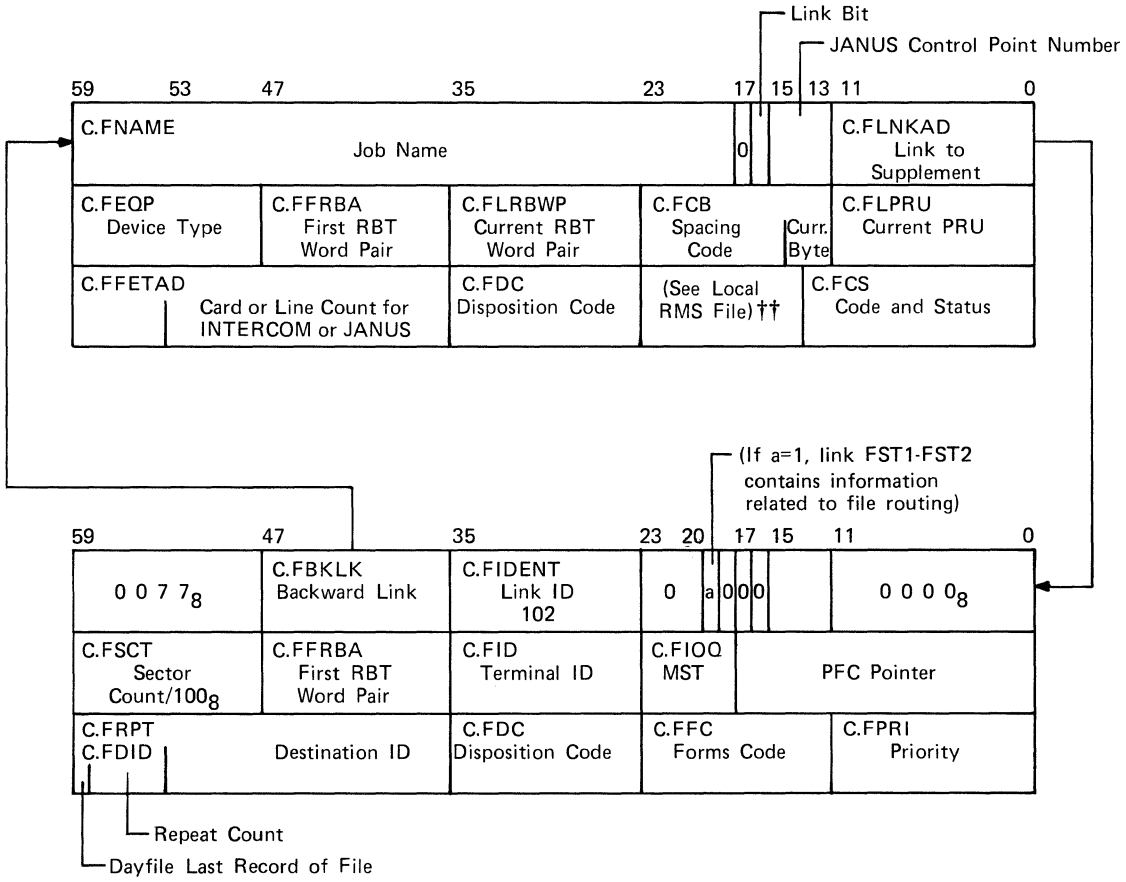
Word	Bit	Description
1	55	ECS preallocation flag.
	56	1 Release bit.
	59	1 ECS buffered file.
2	59	Transfer in progress.

DEVICE SET ENTRY †



Field Name		Description
A	1	Set name flag (S.FSET).

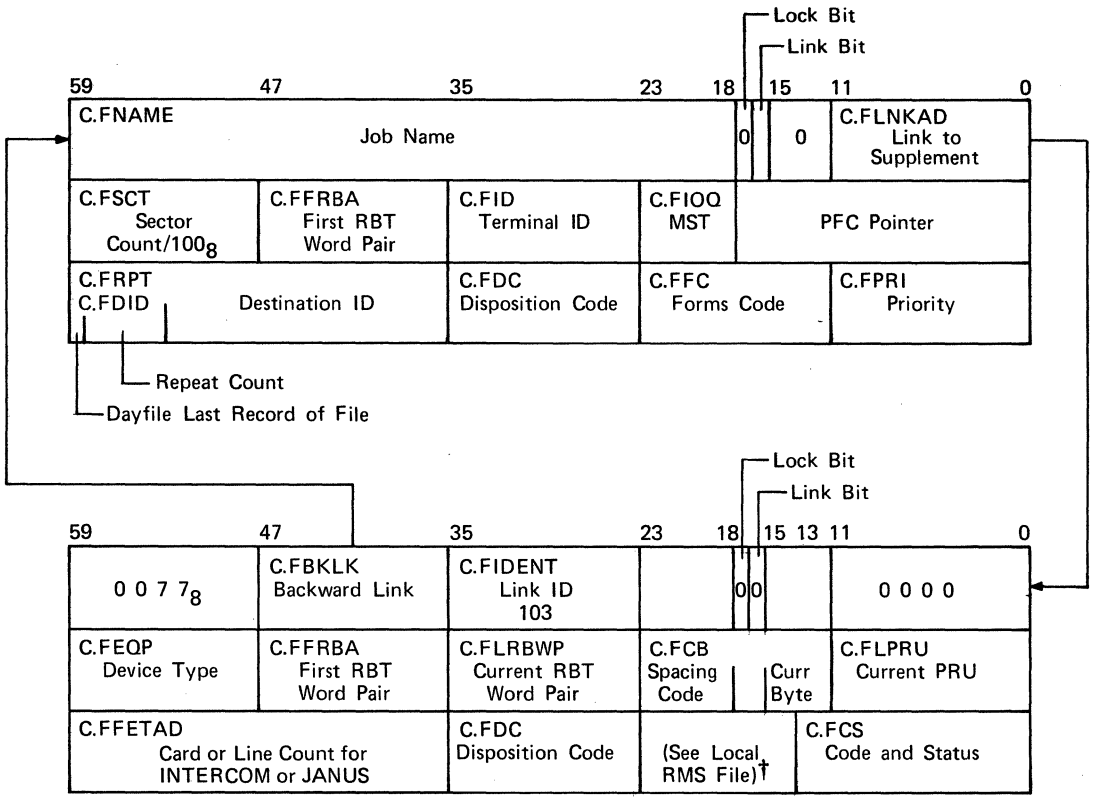
ENTRIES AND REQUIRED SUPPLEMENT FOR FILES IN OUTPUT QUEUES DURING PROCESSING



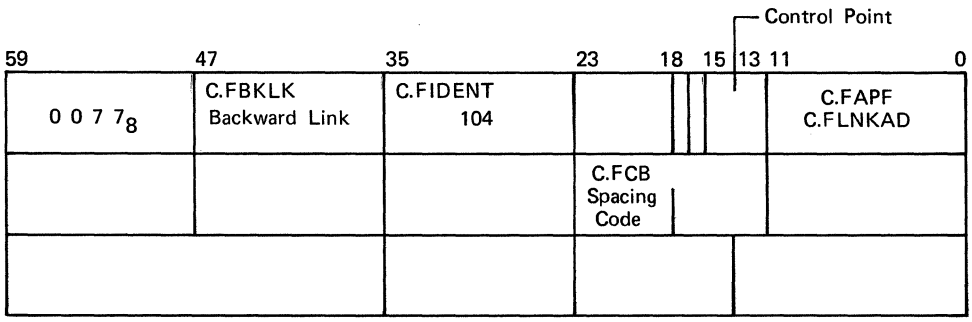
† One entry for each device set that each job has mounted.

†† Supplement will be present if file processing has been interrupted by abort or deadstart recovery. File is not rewound.

ENTRIES AND OPTIONAL SUPPLEMENT FOR FILES IN OUTPUT QUEUES

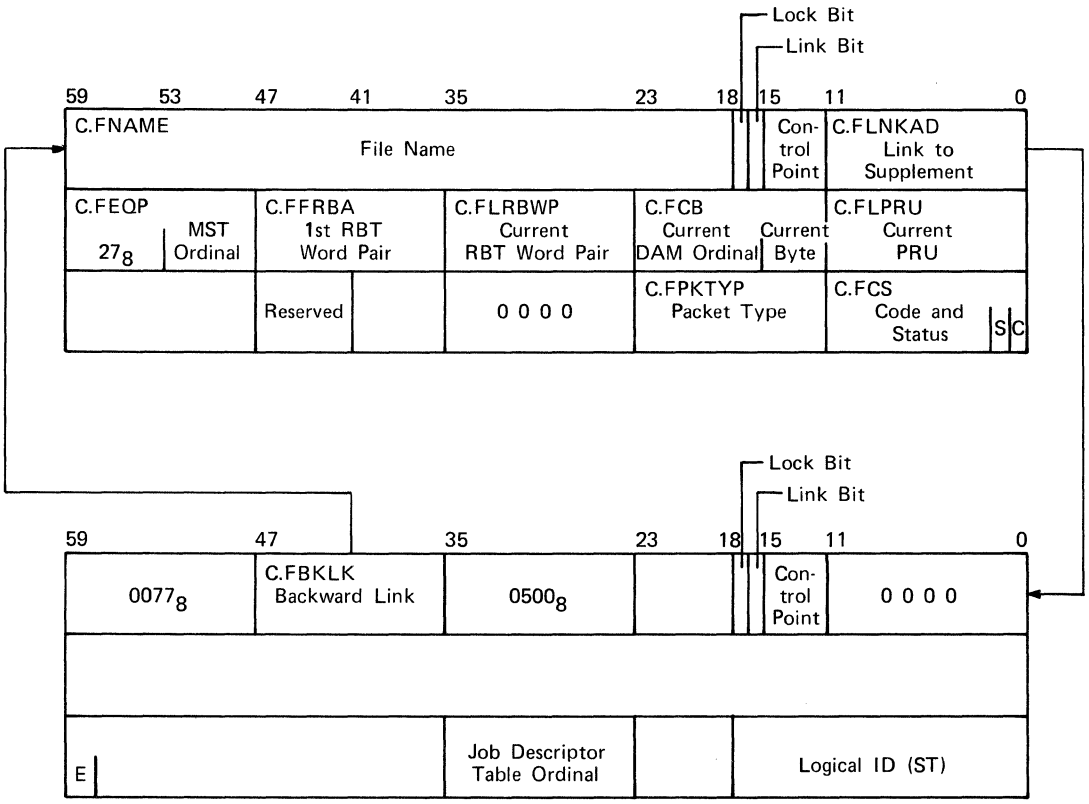


SUPPLEMENT FOR SC SPECIFIED WITH FILE IN OUTPUT QUEUE



† Supplement will be present if file processing has been interrupted by abort or deadstart recovery. File is not rewound.

MULTIMAINFRAME PACKET FILE ENTRIES
ENTRY AND REQUIRED SUPPLEMENT FOR FILE BEFORE OPEN (GETPF)

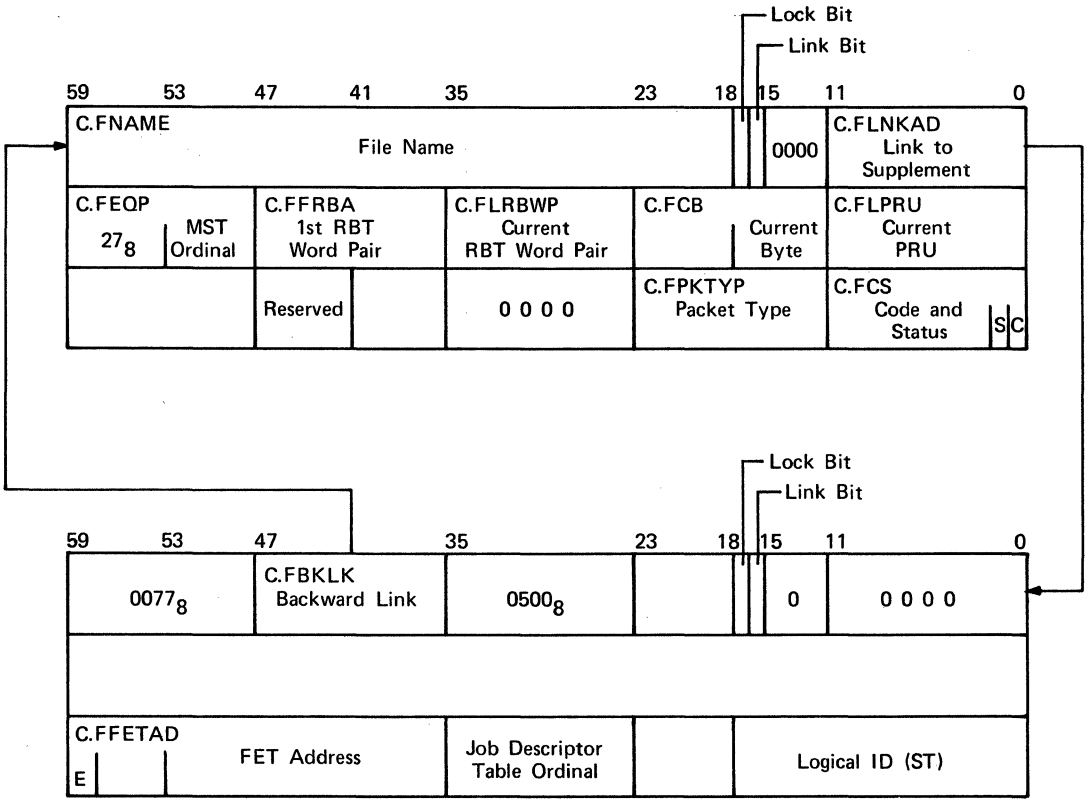


Field Name	Description
S	Packet status bit.
C	Complete bit.
E	EC flag that was saved from the user FDB.

Packet types (C.FPKTYP) are as follows:

Code (in octal)	Type
100	ATTACH
200	CATALOG
300	PURGE

ENTRY AND REQUIRED SUPPLEMENT AT CONTROL POINT 0 AFTER OPEN (GETPF)

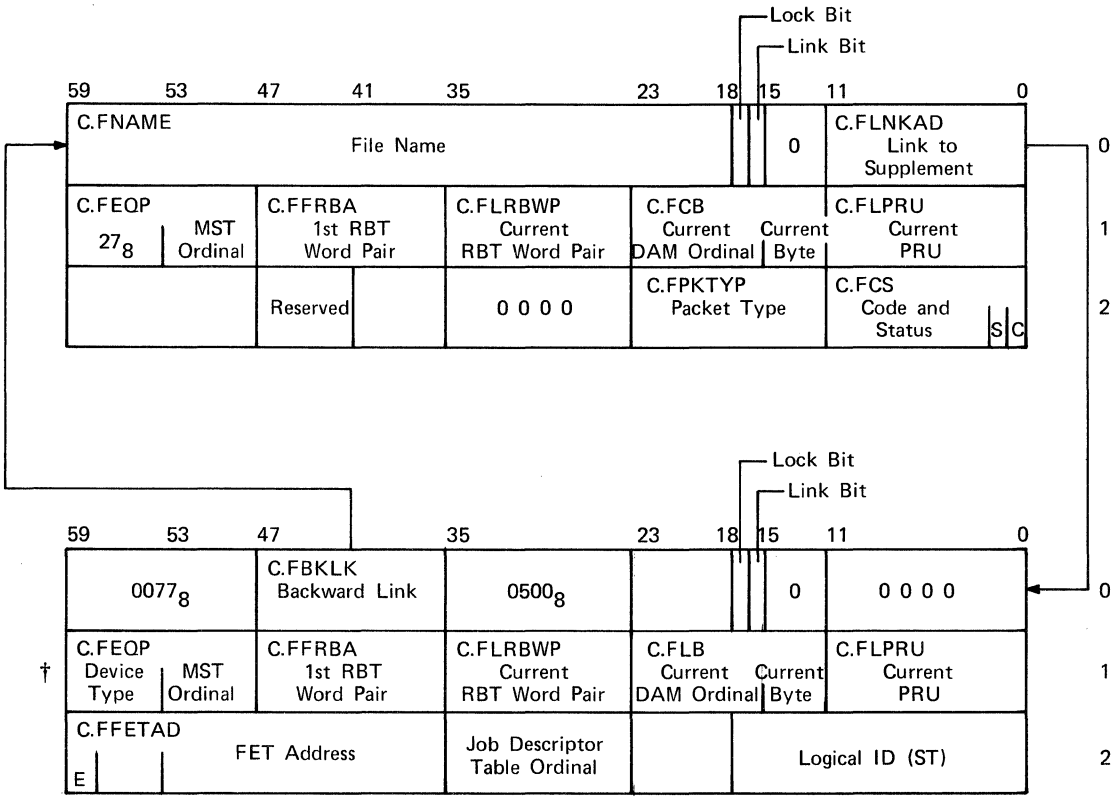


Field Name	Description
S	Packet status bit.
C	Complete bit.
E	EC flag that was saved from the user FDB.

Packet types (C.FPKTYP) are as follows:

Code (in octal)	Type
100	ATTACH
200	CATALOG
300	PURGE

ENTRY AND REQUIRED SUPPLEMENT FOR FILE AT CONTROL POINT 0
AFTER COPY HAS BEEN STAGED (GETPF)



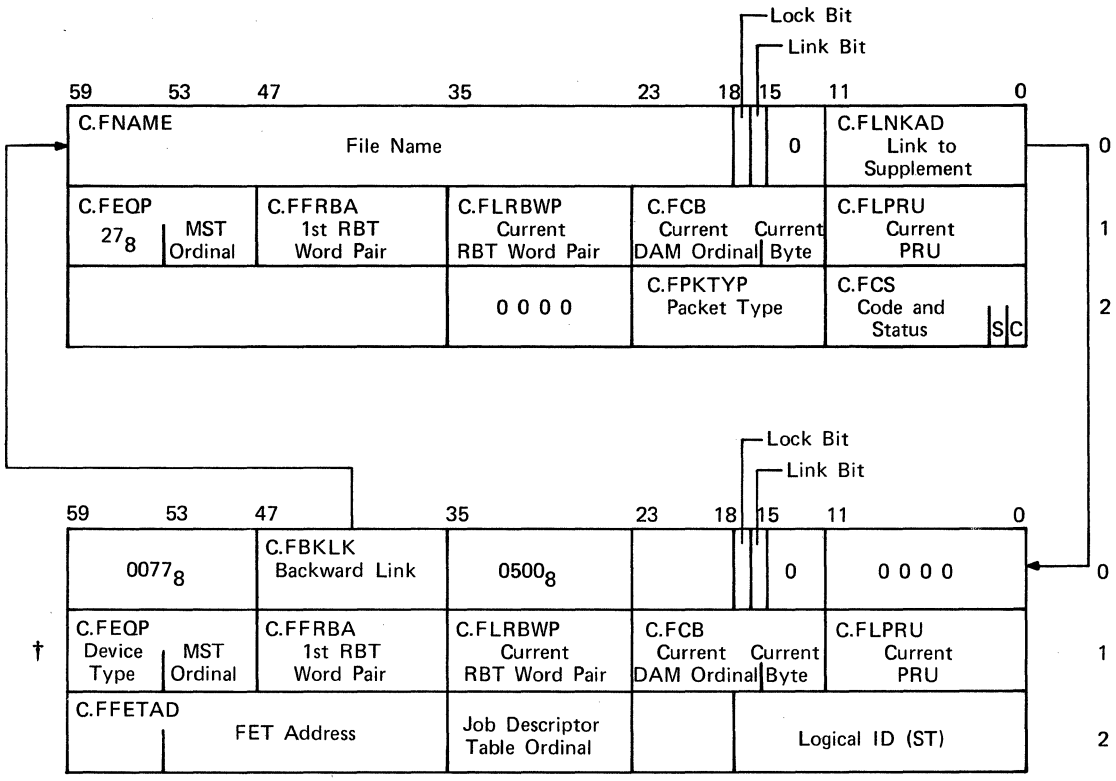
Field Name	Description
S	Packet status bit.
C	Complete bit.
E	EC flag that was saved from the user FDB.

Packet types (C.FPKTYP) are as follows:

Code (in octal)	Type
100	ATTACH
200	CATALOG
300	PURGE

†Actual file information is in word 1 of the supplement.

ENTRY AND REQUIRED SUPPLEMENT FOR FILE AT CONTROL POINT 0 (SAVEPF)



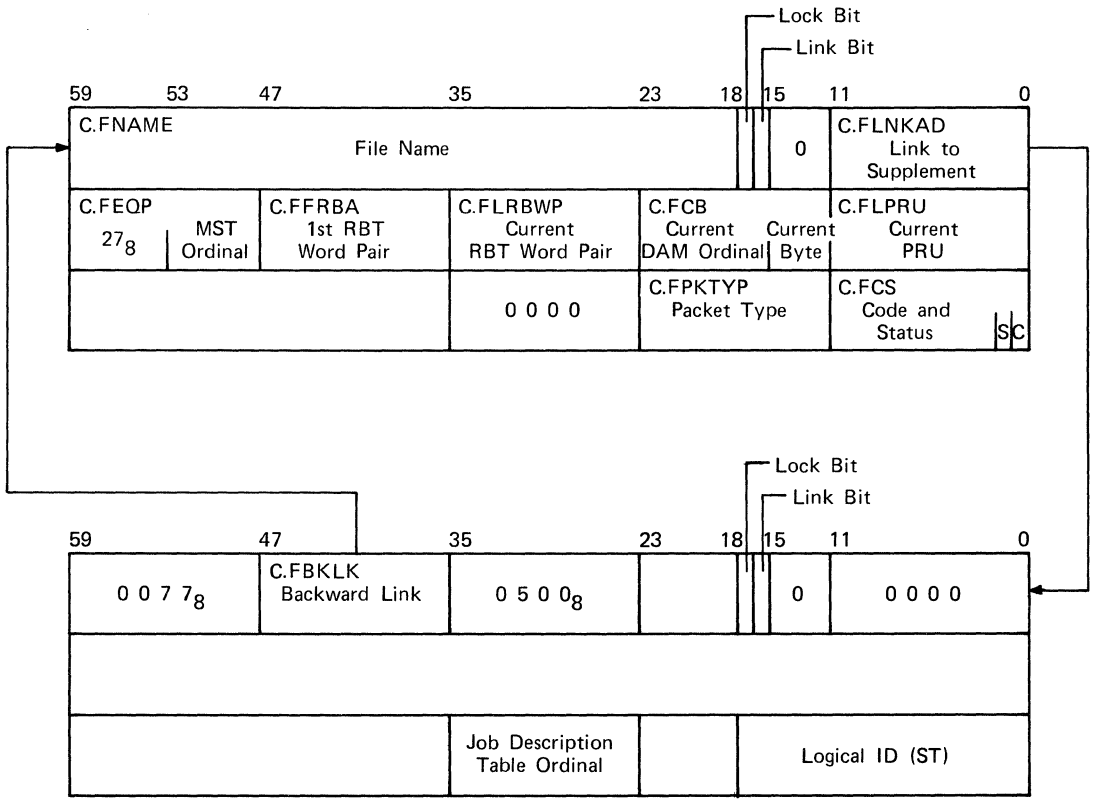
Field Name	Description
S	Packet status bit.
C	Complete bit.

Packet types (C.FPKTYP) are as follows:

Code (in octal)	Type
100	ATTACH
200	CATALOG
300	PURGE

†Actual file information is in word 1 of the supplement.

ENTRY AND REQUIRED SUPPLEMENT FOR FILE AT CONTROL POINT 0 (PURGE)



Field Name	Description
S	Packet status bit.
C	Complete bit.

Packet types (C.FPKTYP) are as follows:

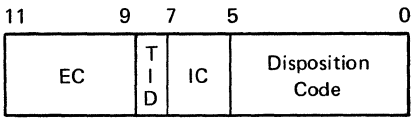
Code (in octal)	Type
100	ATTACH
200	CATALOG
300	PURGE

DISPOSITION CODE VALUES (C.FDC)

Values for nonallocatable devices are:

Code	Value	Description
CK	xxx1	Checkpoint.
IU	xxx2	Inhibit unload.
CI	xxx3	Checkpoint and inhibit unload.
SV	xxx4	Save.
CS	xxx5	Checkpoint and save.
	xxx6	Reserved.
	xxx7	Reserved.

For allocatable devices, the byte C.FDC is divided into four fields.



Bit	Description
11-9	External code. The first code applies to print files, the second to punch files (print/punch).

Code	Value	Description
Default	000	Default print train/default punch character set.
- - /EC=SB	001	Reserved/punch system binary.
EC=4A/EC=80 column	010	ASCII 48-character print train/punch 80-column binary.
EC=B4/ - -	011	BCD 48-character print train.
EC=B6/EC=026	100	BCD 64-character print train/punch 026.
EC=A6/EC=029	101	ASCII 64-character print train/punch 029.
EC=A9/EC=ASCII	110	ASCII 96-character print train/punch ASCII.
- - /- -	111	Reserved for installations.

Bit	Description

8 Terminal identification. Relevant only for local files, not queue files.

Code	Value	Description
TID=xy	0	Route file to remote user (xy is terminal identification).
TID=C	1	Ignore remote ID in file routing.

7-6 Internal code.

Code	Value	Description
IC=DIS	00	File is in display code format.
IC=ASCII	01	File is in ASCII format.
IC=BIN	10	File is in binary format.
IC=TRANS	11	File is in transparent format (INTERCOM 5 only).

5-0 Disposition code.

Code	Value	Description
	01	Reserved.
	02	Reserved.
	03	Reserved.
	04	Input job ready for scheduling (control point 0) or input file (control point 1 through 17).
	05	Input tape job.
	06	Input tape job on VSN display.
	07	Reserved.
PU, PB, or P8	10	Punch, EC=80, SB, 026, or 029.
FR†	20	Film print.
	21	Reserved.
FL†	22	Film plot.
	23	Reserved.
HR†	24	Hardcopy print.
	25	Reserved.

†Recognized but not supported by the operating system.

Code	Value	Description
HL †	26	Hardcopy plot.
	27	Reserved.
PT †	30	Plot.
	31-37	Reserved.
PR	40	Any available printer.
--	41	
P2	42	Reserved for installations. †
LR	43	Any available 580-12; only EC=B4, B6, A4, A6, or A9 are valid.
LS	44	Any available 580-16; only EC=B4, B6, A4, A6, or A9 are valid.
LT	45	Any available 580-20; only EC=B4, B6, A4, A6, or A9 are valid.
	46-65	Reserved.
	66	System dynamic dump file.
	67	Scratch file.
	70-77	Reserved for installations.

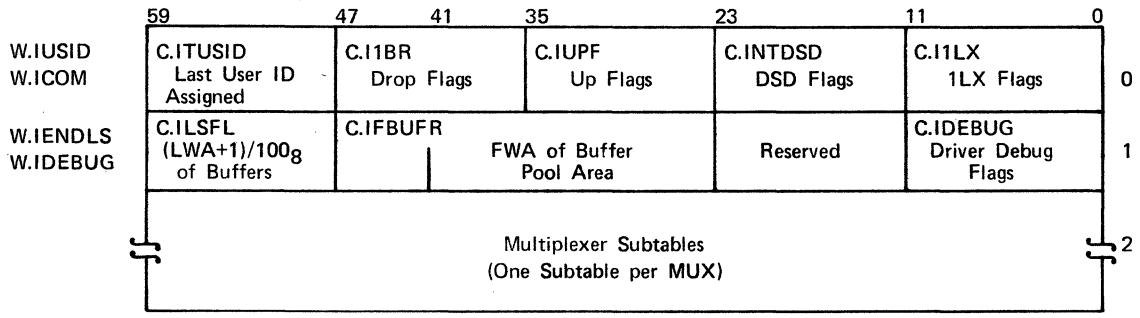
LINK IDENTIFICATION VALUES (C.FIDENT)

Value (in octal)	Description
0000	Reserved.
0100	Input queue link.
0101	Input queue during processing link.
0102	Output queue during processing link.
0103	Output queue file interrupted by deadstart recovery or ABORT.
0104	Output queue file with SC specified.
0105 thru 0177	Reserved.

† Recognized but not supported by the operating system.

Value (in octal)	Description
0200	Tape with alternate VSNs link.
0201	Tape with multiple VSNs link.
0202	Tape with 2MT(NT) link.
0203 thru 0277	Reserved.
0300	Permanent pack link.
0301	Disk VSN for file assignment.
0302	819 file supplement.
0303 thru 0377	Reserved.
0400	Editor file during processing link.
0401	File to be replaced link.
0402	File to be deleted link.
0403	SPOT dayfile link.
0404 thru 0477	Reserved.
0500 thru 0577	Reserved.
0600	Routing information link.
0601	Reserved.
0602	System dynamic dump file link.
0603 thru 0677	Reserved
0700	Reserved for ECS resident file or I/O buffered file link.
0701	Reserved for ECS resident library link.
0702 thru 6777	Reserved.
7000 thru 7777	Reserved for installations.

INTERCOM 4 MULTIPLEXER TABLE



W.ICOM(0) C.I1BR(1) Multiplexer Table Header - Communications

Bit	Field	Description
0	S.IEDIT	EDITLIB active.
1	S.II1I	1I1 initialization active.
2	S.I1BR	1BR drop flag.
3	S.IDI	Driver-initializer active.

W.ICOM(0) C.IUPF(2)

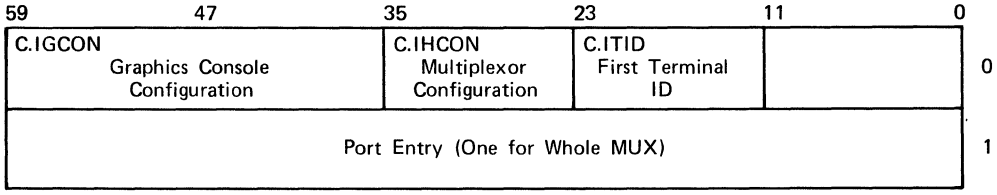
Bit	Field	Description
5	S.I1CI	1CI up.
6	S.IDD	Driver up.

W.ICOM(0) C.INTDSD(3)

Bit	Field	Description
0	S.INTDRP	INTERCOM drop requested flag.
1	S.INTRST	Restart in progress.
4	S.ILKOUT	Login locked out.
5	S.I1CI	1CI drop requested.
6	S.IDD	Driver/1LX drop requested.

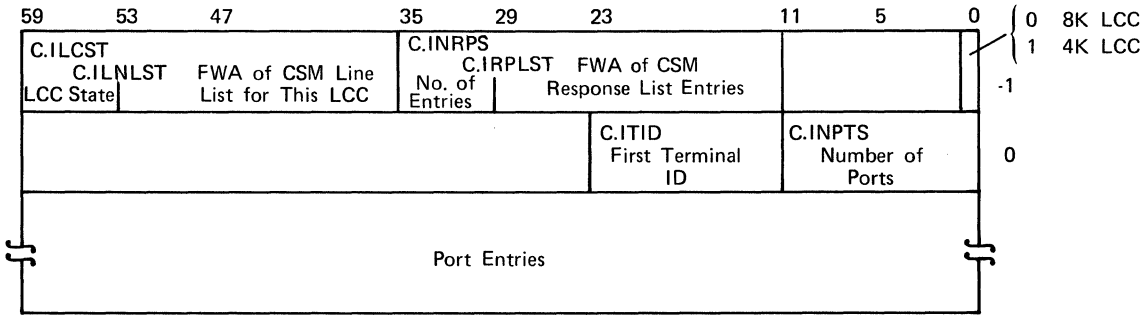
W.ICOM(0) C.I1LX(4) Two bit counts of active 1LXs for each driver.

6673/6674 MULTIPLEXER SUBTABLE

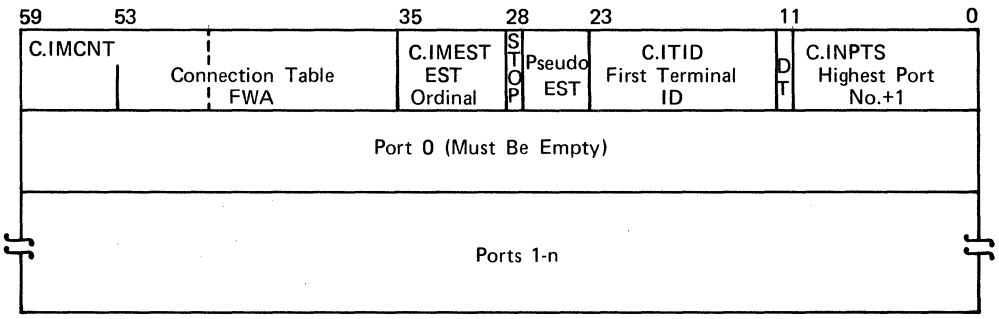


C.IGCON is 24 bits representing 24 possible consoles (00 to 36) from left to right.

LCC MULTIPLEXER SUBTABLE



2550 NPU MULTIPLEXER SUBTABLE



C.IMCNT

Bit	Field	Description
5-0		Upper 6 bits of connection table FWA.

C.IMCNT+1

Bit	Field	Description
11-0		Lower 12 bits of connection table FWA.

C.IMEST

Bit	Field	Description
11-6		EST ordinal.
5		Reserved.
4		Request-to-stop-NPU-service bit.
3		Pseudo EST.

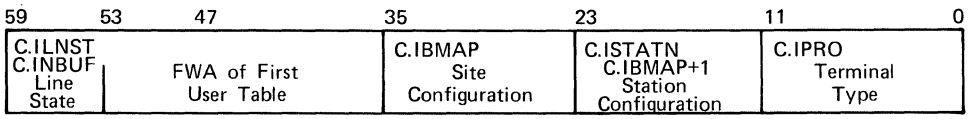
C.ITID

Bit	Field	Description
11-0		First terminal ID.

C.INPTS

Bit	Field	Description
8-0		Highest port number + 1. The line entry for port 0 must be empty. A communication line must not be connected to port 0.
10-9		Reserved.
11	S.IBDMP	Dump type. 0 Display code dump. 1 Binary dump.

SUBTABLE PORT ENTRY

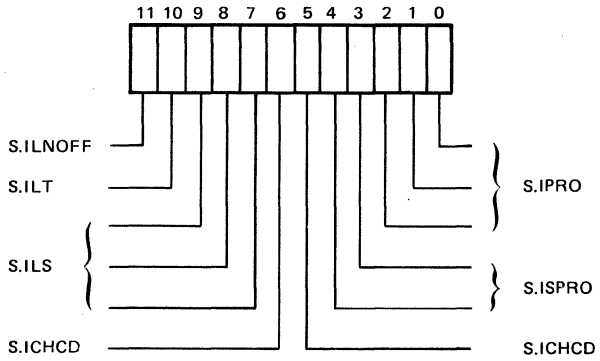


Bit map of streams configured by 73x-10. Replaces site and station address for these terminals.

C.IBMAP+1(3)

Bit	Description
0	Unused.
1	Stream; 0; configured; IDS output.
2	Stream; 2; configured; CRT.
3	Stream; 8; configured; CP1.
4	Stream; 12; configured; LP4.
5	Stream; 10; configured; LP3.
6	Stream; 6; configured; LP2.
7	Stream; 4; configured; LP1.
15-8	Unused.
16	Stream; 1; configured; IDS input.
17	Stream; 3; configured; KBD.
18	Stream; 7; configured; CR2.
19	Stream; 5; configured; CR1.
22-20	Unused.

C.IPRO



S.ILNOFF

Bit	Description
11	Line status.
	0 On.
	1 Off.

S.ILT

Bit	Description
10	Line type.
	0 Dial-up.
	1 Hardware.

S.ILS

Baud rate indicated by the line speed field is protocol dependent.

Bits 9-7

Line Speed	Mode 2	Mode 3	Mode 4	Wide Band
0	-	110	-	-
1	-	134.5†	-	-
2	-	150	-	-

†134.5 bps not supported.

Line Speed	Mode 2	Mode 3	Mode 4	Wide Band
3	-	300	-	-
4	2000/2400	600†	2000/2400	-
5	4800	1200†	4800	-
6	9600	-	9600	-
7	19 200+	††	-	19 200+

S.ICHCD

Character conversion.

0 ASCII.

1 External BCD.

2 Display code.

S.ISPRO

Subprotocol if mode 4 terminal.

0 Auto detect (mode 4A/mode 4C).

1 Mode 4A.

2 Mode 4C.

Subprotocol if wideband terminal.

1 Batch only.

2 Interactive.

S.IPRO

Protocol field.

0 Empty.

1 Mode 3.

2 Mode 4.

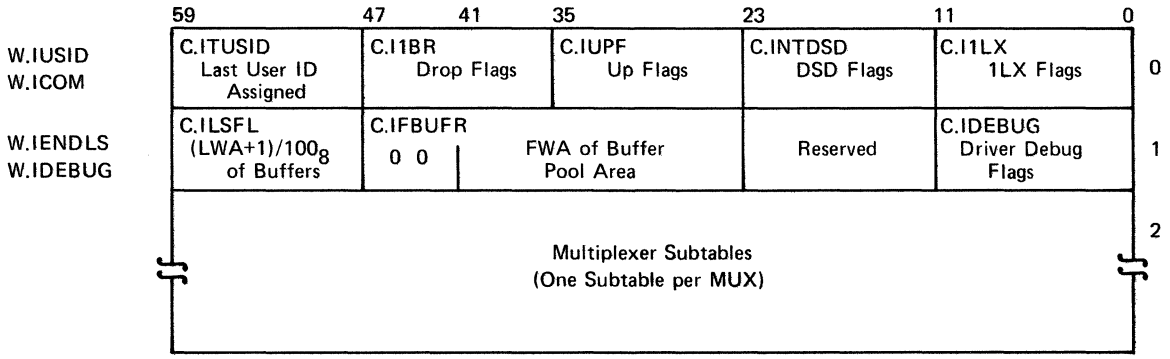
3 Mode 2.

4 Wideband.

---

†600 and 1200 bps allowed for 255x front end only.
††Indicates automatic baud recognition (allowed for 255x front end only).

INTERCOM 5 MULTIPLEXER TABLE



W.ICOM(0) C.I1BR(1) Multiplexer Table Header - Communications

Bit	Field	Description
0	S.IEDIT	EDITLIB active.
1	SI1I1	1I1 initialization active.
2	S.I1BR	1BR drop flag.
3	S.IDI	Driver-initializer active.

W.ICOM(0) C.IUPF(2)

Bit	Field	Description
5	S.I1CI	1CI up.
11-6	S.IDD	Driver up.

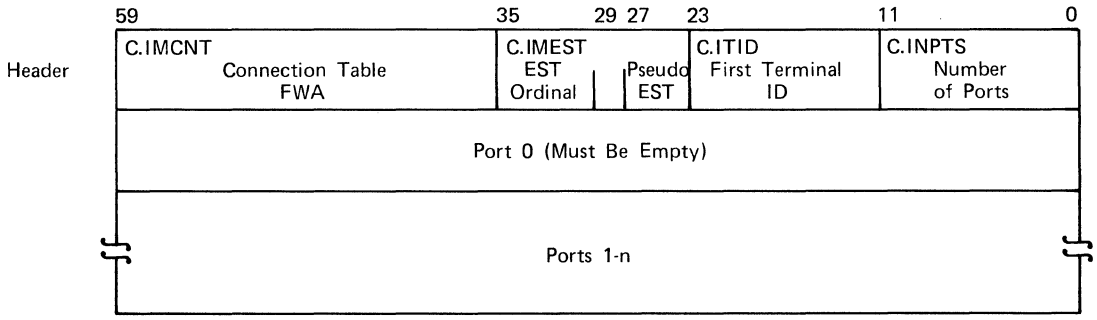
W.ICOM(0) C.INTDSD(3)

Bit	Field	Description
0	S.INTDRP	INTERCOM drop requested flag.
1	S.INTRST	Restart in progress.
3-2		Reserved.
4	SILKOUT	Login locked out.
5	S.I1CI	1CI drop requested.
11-6	S.IDD	Driver 1P drop requested.

W.ICOM(0) C.I1NP(4) 1NP Up Flags

Bit	Field	Description
0	S.I1NP	1NP active.

2550 NPU MULTIPLEXER SUBTABLE



C.IMCNT

Bit	Description
5-0	Upper 6 bits of connection table FWA.

C.IMCNT+1

Bit	Description
11-0	Lower 12 bits of connection table FWA.

C.IMEST

Bit	Description
11-6	EST ordinal.
5	Set if 2550 is already loaded (off-line).
4	Reserved.
3-0	Pseudo EST ordinal.

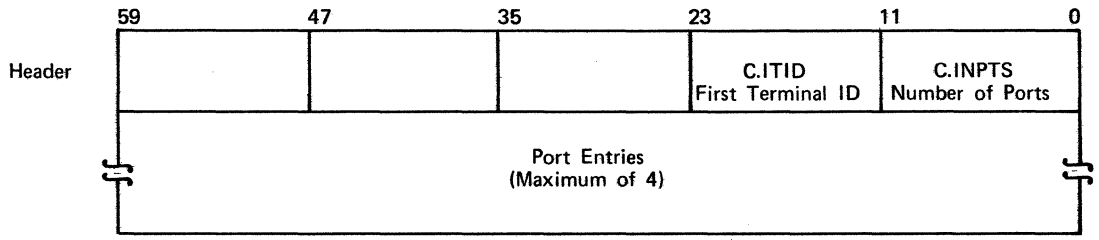
C.ITID

Bit	Description
11-0	First hardwired terminal ID.

C.INPTS

Bit	Description
8-0	Highest port number + 1. The line entry for port 0 must be empty. A communication line must not be connected to port 0.
10-9	Reserved.
11	S.IBDMP; 255x binary dump bit.

EXPORT MULTIPLEXER SUBTABLE



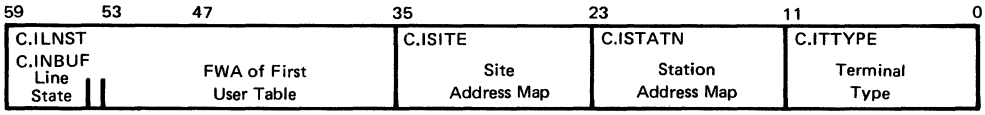
C.ITID

Bit	Description
11-0	First terminal id.

C.INPTS

Bit	Description
11-0	Number of ports (1 through 4).

SYNCHRONOUS TERMINAL PORT ENTRY



C.ILNST(0)
C.INBUF(0)

Bit	Field	Description
11-7		Driver line state.
6	S.IMSUTR	User table requested flag.
5-0		Upper 6 bits of address of first user table on port or configuration buffers.

C.INBUF+1(1)

Bit	Description
11-0	Lower 12 bits of first user table on port or configuration buffers.

C.ISITE(2) Site Address for This Port

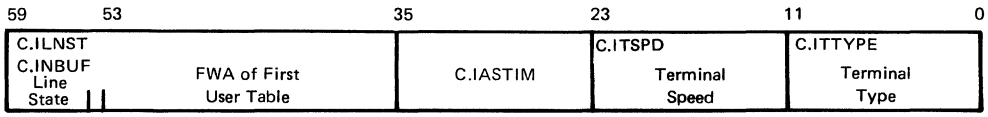
C.ISTATN(3) Station Address for This Port

C.ITTYPE(4)

Bit	Field	Description
11	S.ILNOFF	Line status. 0 On. 1 Off.
10	S.ILT	Line type. 0 Dial-up terminal. 1 Hardwired terminal.

Bit	Field	Description
9	S.IKCAR	Carrier control. 0 Carrier-controlled terminal. 1 Carrier-constant terminal.
8	S.ILNACO	If set, terminal is off externally.
7	S.ILNACI	If set, terminal is off internally.
6-5	S.ICHCV	Character code conversion. 0 ASCII. 1 External BCD. 2 Display code. 3 EBCDIC.
4-0	S.ITTYPE	Terminal type. 00 Empty. 01-07 Asynchronous 01 Mode 3. 02-07 Reserved. 10-17 CDC synchronous. 10 Mode 4A. 11 Mode 4C. 12-16 Reserved. 17 Mode 4A/4C autorecognition. 20-27 Non-CDC synchronous. 20 IBM 2780. 21 IBM 3780. 22 HASP. 23-26 Reserved. 27 IBM 2780/IBM 3780/HASP autorecognition. 30-37 Reserved for installation.

ASYNCHRONOUS TERMINAL PORT ENTRY



C.ILNST(0) Identical to Synchronous Terminal Port Entry

C.INBUF(0,1) Identical to Synchronous Terminal Port Entry

C.IASTIM(2) S.IASTIM ASTIM Terminal

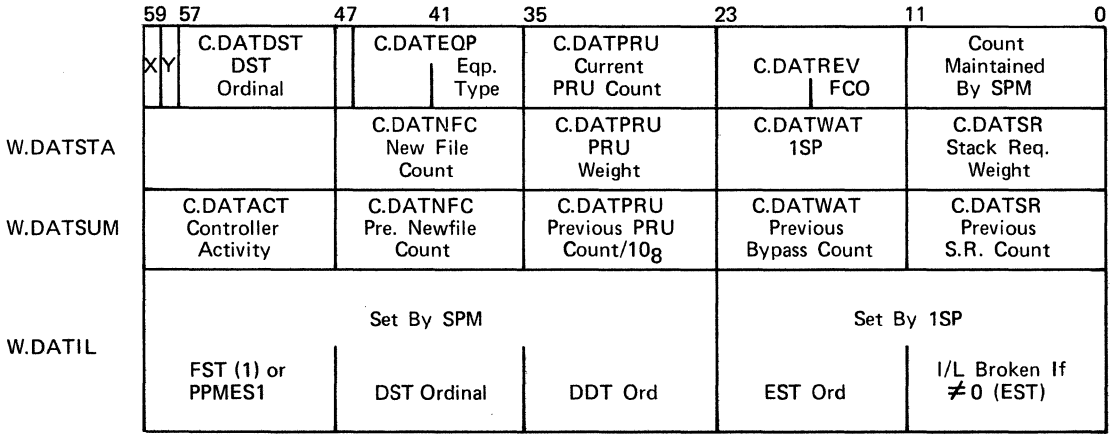
C.ITSPD(3)

Bit	Field	Description
11		Reserved.
10	S.INIC	No initial carrier flag.
9-5		Line or terminal speed if automatic speed detection.
4-0		Line speed.

00	110 baud.
01	134 baud.
02	150 baud.
03	300 baud.
04	600 baud.
05	1200 baud.
06	2400 baud.
07	4800 baud.
10	9600 baud.
11-17	Reserved.
20-36	Reserved for installation.
37	Automatic speed detection.

C.ITTYPE(4) Identical to Synchronous Terminal Port Entry

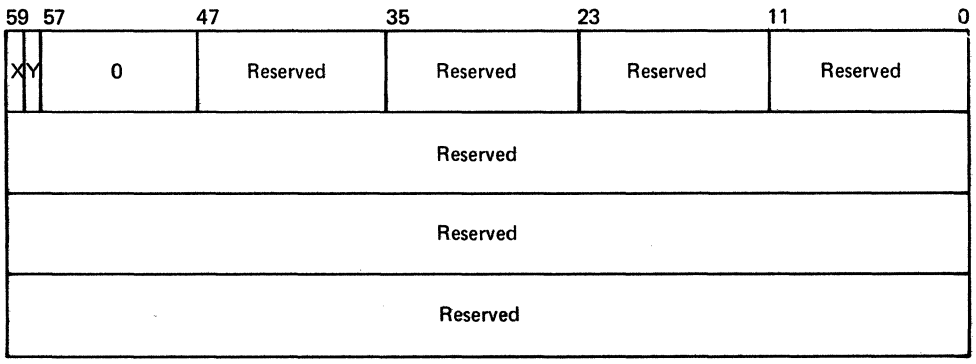
DEVICE ACTIVITY TABLE ENTRY



Field Name	Description
X	Controller on is 0; off is 1.
Y	Controller autoloaded is 0; not autoloaded is 1.
FCO	Controlware revision level.

W.DATIL can be cleared only by 1SP.

DUMMY ENTRY FOR MULTIPLE ACCESS DEVICE



TAPES STAGING TABLE

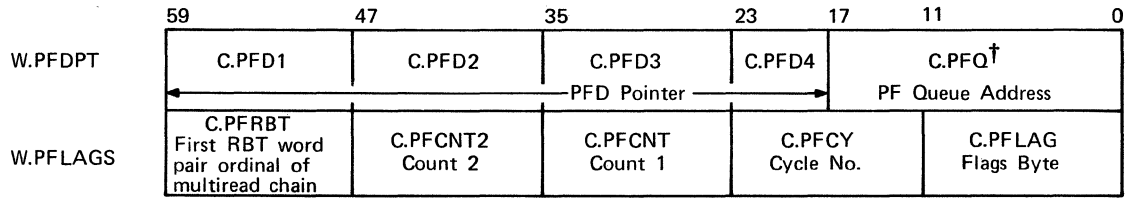
	59 C.STGMT	47 C.STGNT C.STGHD	35 C.STGPE	23 C.STGGR C.STGGE†††	11 0	
W.STGMAX†	Number of MT Defined	Number of NT Defined	Unused	Number of GE Defined	Unused	Total
W.STGFRE†	Number of MT on and Unassigned	Number of NT on and Unassigned	Unused	Number of GE on and Unassigned	Unused	Available
W.STGUFD††	Unfilled MT Demand	Unfilled HD Demand†††	Unfilled PE Demand	Unfilled GE Demand	Unused	Unfilled Demand
W.STGSAT†	Number of MT Held by Satisfied Jobs	Number of NT Held by Satisfied Jobs	Unused	Number of GE Held by Satisfied Jobs	Unused	Assigned
	No Tape Status Flag	Unused				
W.STGTLE	Useless Information from T.MSC		Copy of T.MSC			
W.STGTLR						
W.STGTLT						

† Indexed according to device. NT is defined in this table as a nine-track tape unit, which can write/read 800/1600 cpi. GE is a nine-track tape unit, which can write/read 1600/6250 cpi.

†† Indexed according to user density requests. If NT is requested, the installation's default nine-track density (HE, PE, or GE) will be assigned.

††† Mapped to C.STGNT if installation option OP.SCHDE is zero.

ATTACHED PERMANENT FILE TABLE

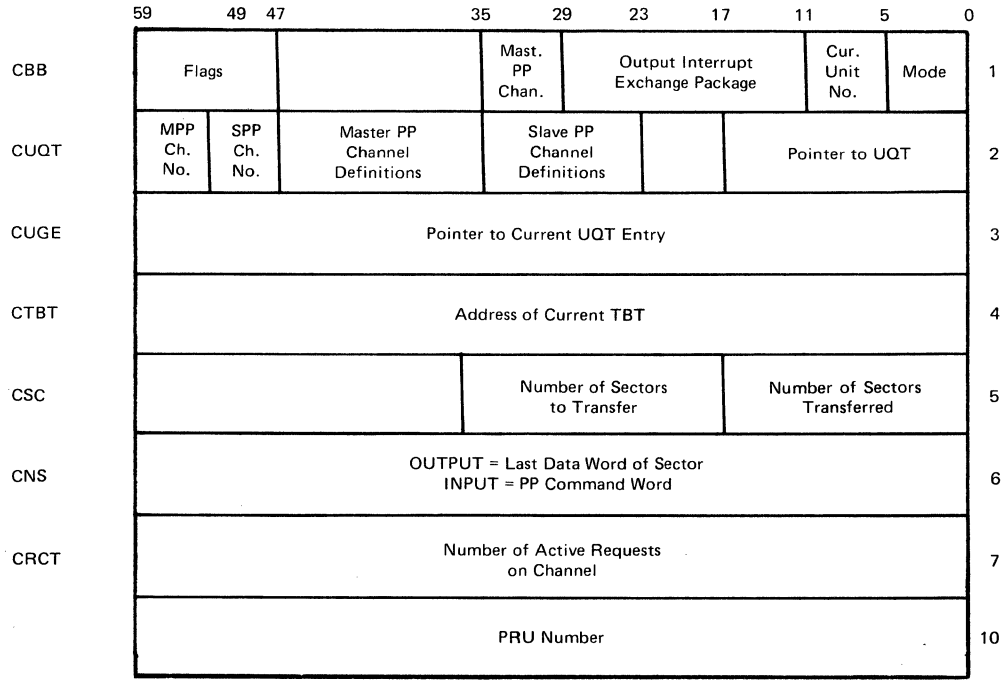


C.PFLAG bits are as follows:

Bit	Description
11	Unused.
10	Full dump (S.PFFD).
9	System file (S.PFSYS).
8	RBT chain obsolete (S.PFOBS).
7	Archived file (S.PFARC).
6	Multimainframe permission conflicts (S.PFMMF).
5	Interlock (S.PFIL).
4	RB conflicts (S.PFRBC).
3	Exclusive access (S.PFEA).
2	Single modify (S.PFSM).
1	Single write (S.PFSW).
0	Priority lockout (S.PFPL). Reserved entry (S.PFRES).

† If nonzero, PF queue address equals JDT address of a job waiting for access to the permanent file.

CHANNEL TABLE



Word 1

Bit	Description
59-48	Flags.

Bit Set	Significance
59	Busy.
58	Down.
57	CHTRQ; transmit request.
56	Term; one more sectors to transmit.
55	BEGPB; next sector to transfer is the end of a PB. Used to maintain a continuous data transfer when switching to NEXTPB.
54	Switch; set for a continuous data switch to a different PB.
53	New TBT; the interrupt handler (IH) has completed switch to a new TBT.
52	Controller error.
51	Initializing; used by IH to start data transfer.

Bit	Description	
	Bit Set	Significance
	50	EOT; IH terminated data transfer on current request.
	49	Active; used by deadstart for loading PPs.
	48	HDC recall.
5-0	Mode.	
	0	Not available.
	1	Idle.
	2	Request.
	3	Data.

Word 2

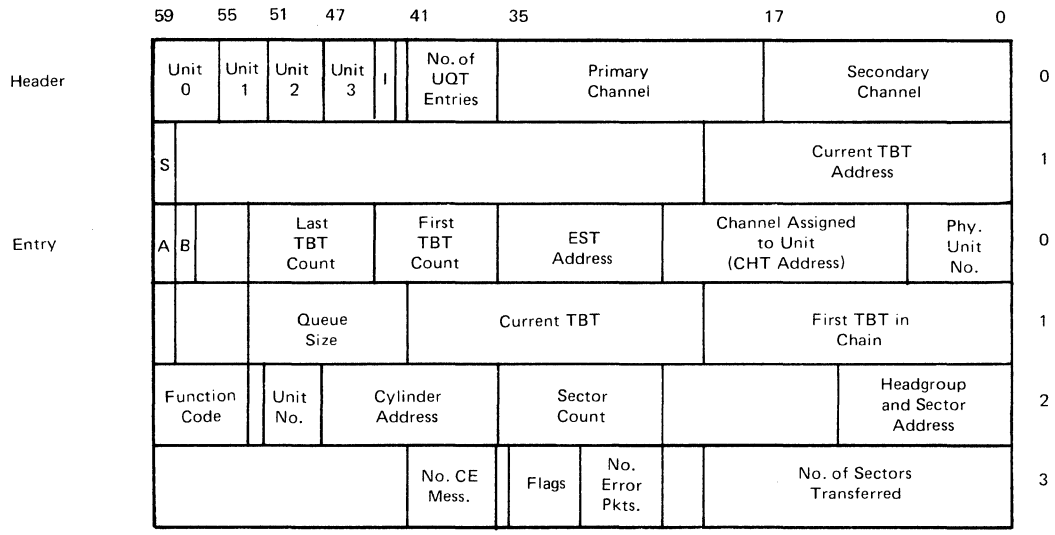
Bit	Description
59-54	Master PP channel.
53-47	Slave PP channel.
47-36	Master PP channel definition as follows:

Bit	Definition
47-45	Disk data channel.
44-42	Disk control channel.
41-39	CPU channel.
38-36	Partner PP channel.

35-24 Slave PP channel definitions.

Bit	Definition
35-33	Disk data channel.
32-30	Disk control channel.
29-27	CPU channel.
26-24	Partner PP channel.

UNIT QUEUE TABLE



Header Word 0

Bit	Description
43	Set if UQT has been initialized.

Header Word 1

Bit	Description
59	Set if IH switched to a new TBT.

Entry Word 0

Bit	Description
59	Set if UQTSW sweep mode.
58	Set if UQTFB backward sweep.

Entry Word 1

Bit	Description
59	Set if current TBT preceded the last request on the queue.

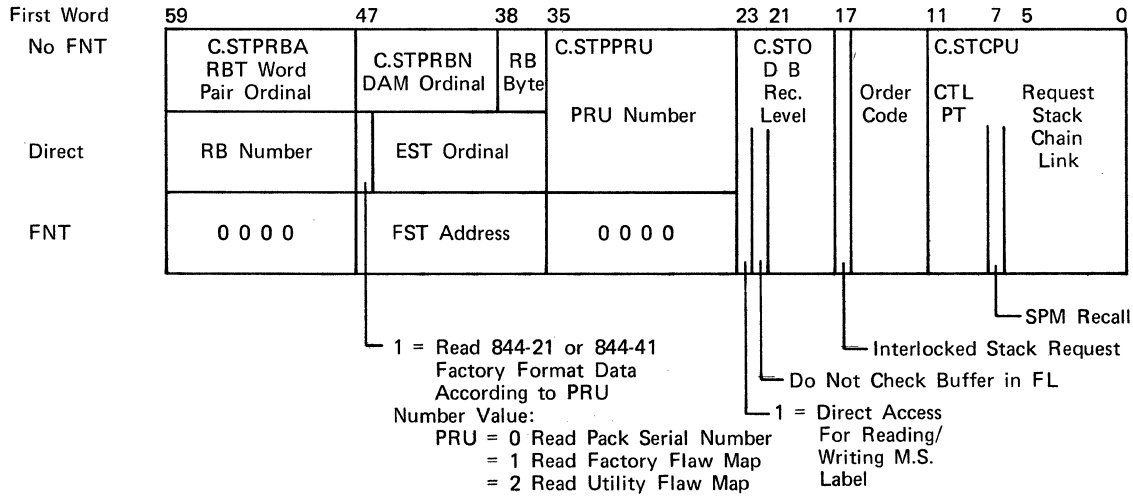
Entry Word 3

Bit	Description
41-36	Number of consecutive requests on this unit which resulted in logging a CE error message.
34-30	Flags.

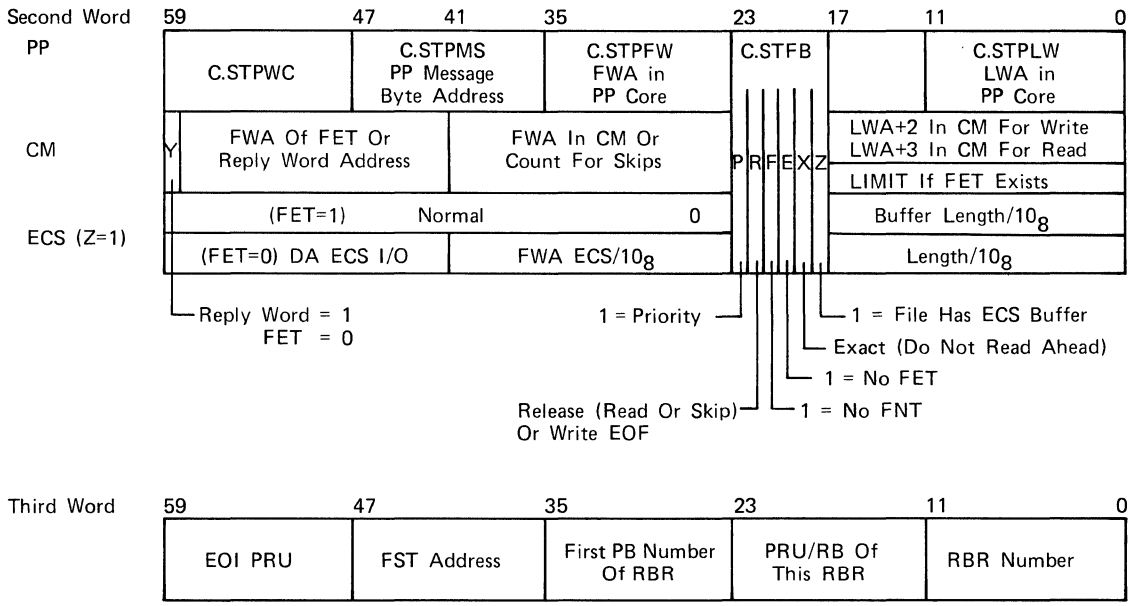
Bit Set	Significance
34	Unrecovered error; flashing ERR message displayed.
33	Unrecovered error.
32	Unrecovered error; no sectors transferred.
31	Unrecovered error; requeue the TBT.
30	Unfinished error logging.

29-24 Number of error packets left to log.

REQUEST STACK ENTRY (T.RQS)



Byte 2 of word 13 in the CMR pointer area contains the first word address divided by 2 of T.RQS. Byte 0 of word 21 in the CMR pointer area contains the number of request stack entries.



The request stack entry is supplied by SPM. SPM converts FNT format into no-FNT format but does not change the no-FNT bit.

Stack processor order codes are as follows:

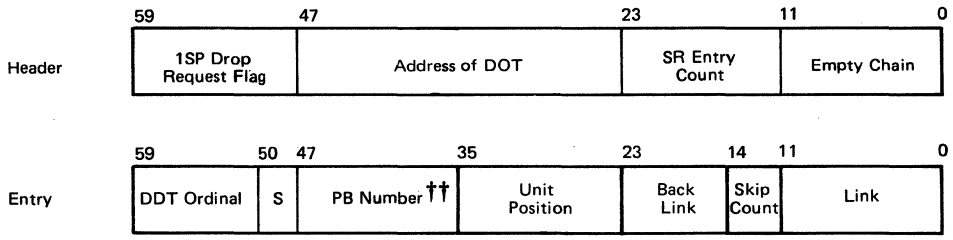
00	O.READ	Read into central memory.
01	O.RDSK	Readskip into central memory.
02	O.RCMPR	Read into central memory; drop first three CM words.
03	O.RDNS	Read nonstop.
04	O.WRT	Write from central memory.
05	O.WRTR	Write EOF/EOR from central memory.
06	O.RMR	Read multiple records to central memory.
07		Not currently defined.
10	O.RDP	Read into PP memory.
11	O.RDPNP	Read into PP; drop first three CM words.
12	O.SKF †	Skip forward.
13	O.SKB †	Skip backward.

† Setting or not setting the interlock stack request bit has no effect on the interlock.

14	O.WRP	Write from PP memory.
15	O.WRPR	Write EOF/EOR from PP memory.
16	O.BPRU†	Backspace PRU.
17	O.RCHN†	Evict.
20	O.RCTNU	Read nonstop (comparable to tape READN).
24	O.WCTNU	Write nonstop (comparable to tape WRITEN).
35	O.IDLE	Wait for stack request.
36	O.DROP	Drop PP.
37	O.SEEK	Issue overlap seeks.

REQUEST SCHEDULING TABLE

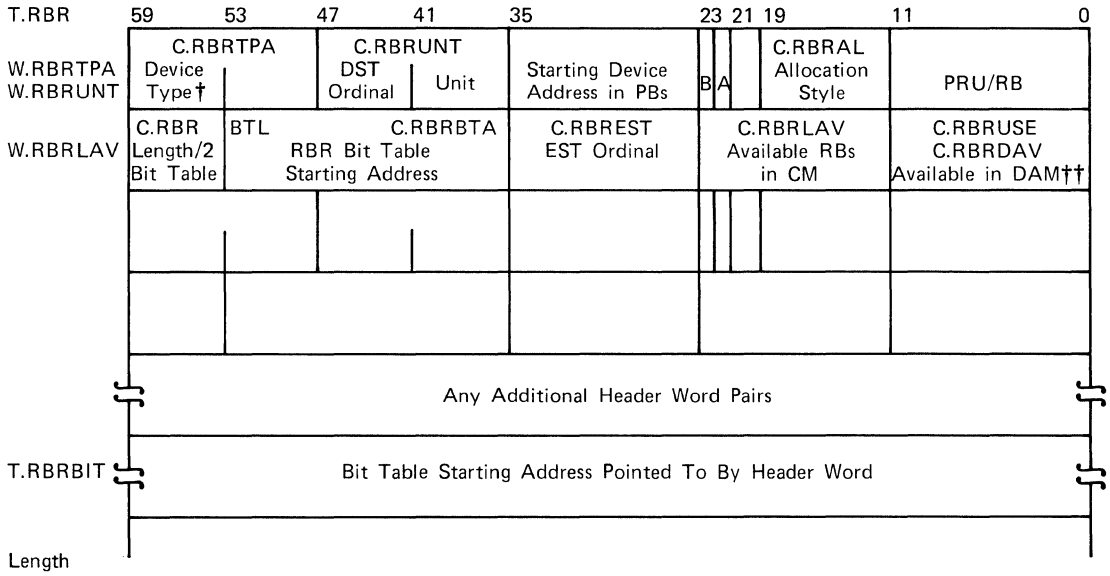
Table is used to schedule stack requests in the request stack (T.RQS).



Field Name	Description
S	Status.

† Setting or not setting the interlock stack request bit has no effect on the interlock.
††Target PB for the stack request; if 7777_8, target PB has not been computed.

RECORD BLOCK RESERVATION TABLE



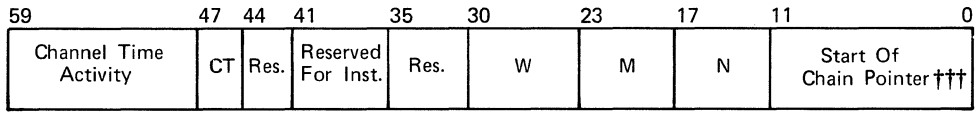
Bits 54 through 36 of word 2 in the CMR pointer area contain the first word address of T.RBR. Byte 1 of word 21 in the CMR pointer area contains the number of RBR headers. Bytes 3 and 4 of word 21 in the CMR pointer area contain the length of the RBR area.

Field Name	Description
A	Overflow flag.
B	Files with no attributes specified may reside here.

DEVICE STATUS TABLE (MASTER ENTRY)

Device status table (DST) entries are numbered, starting from 1, to identify DST ordinals.

C.DSTFLG (First Word)



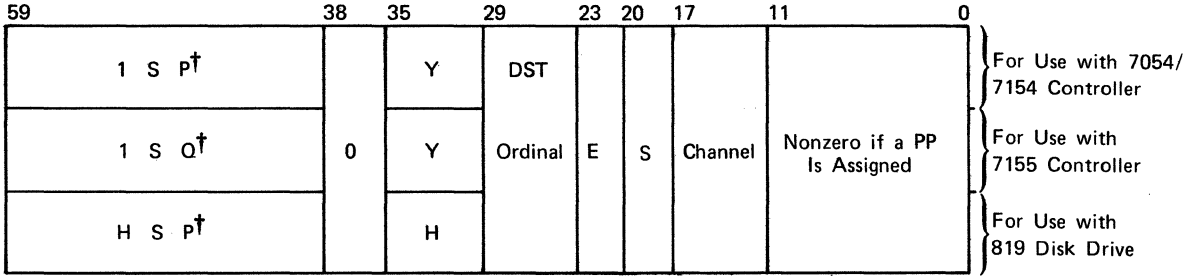
Byte 4 of word 13 in the CMR pointer area contains the first word address divided by 10_8 of the DST. Byte 3 of word 13 in the CMR pointer area contains the number of DST entries.

†Device type codes are the same as those used for the EST.

††During deadstart this byte contains total number of RBs. This field is only updated when DAM is referenced.

†††The field contains the address in the T.RST for the next stack request to process. This field is zero if there is no stack request to process.

C.DSTDRV
(Second Word)



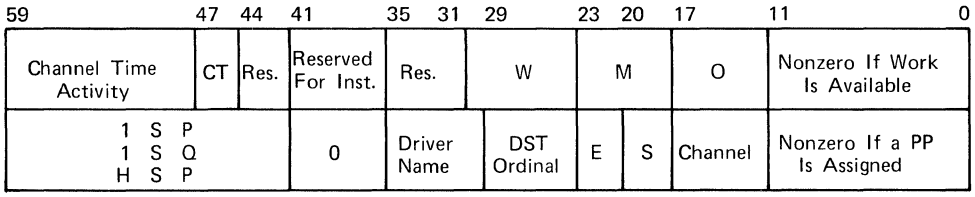
Second word is used as PPIR of the corresponding stack processor.

Field Name	Description
CT	Controller type.
	0 Controller that does not require loading of controlware.
	1 7054 controller.
	2 7154 controller.
	3 7155 controller.
W	1SP/1SQ working flag (when 1SP/1SQ is active).
	0 1SP/1SQ idle.
	S Stack request ordinal S assigned.
M	Member number (0 for master entry).
N	Number of member DST entries.
Y	Display code Y used to form the overlay 3SY.
H	Display code H; currently unused.
E	Equipment.
S	Scheduling options.
	0 FIFO.
	4 Selection only.
	6 Overlap seek and selection.

†1S5 for DST ordinal one.

DST MULTIPLE ACCESS MEMBER ENTRIES

DST entries for second (dual), third, or fourth (844-44 only), access to a set of mass storage units.

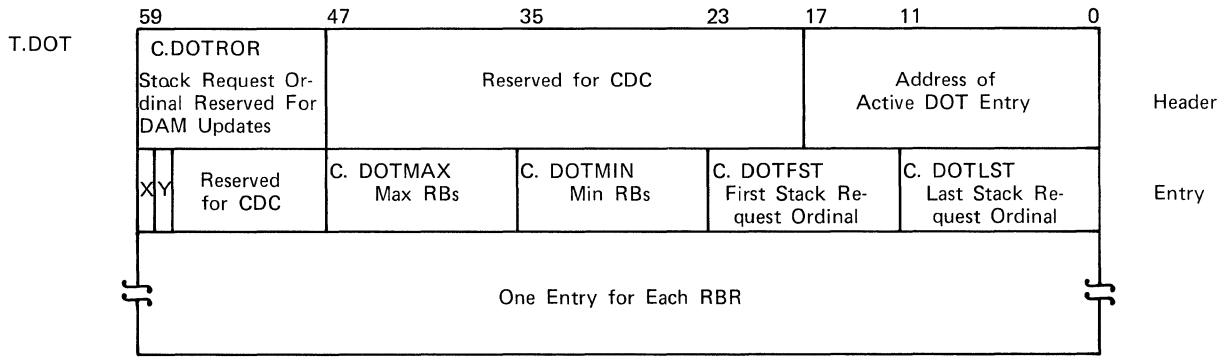


Second word is used for PPIR.

Field Name	Description
M	Member number (1 through 3).

Other field descriptions are identical to those for the master entry.

DEVICE OVERFLOW TABLE



Field Name	Description
X	DAM Update flag.
	0 No DAM update in progress.
	1 DAM update in progress for this entry (active DOT entry).

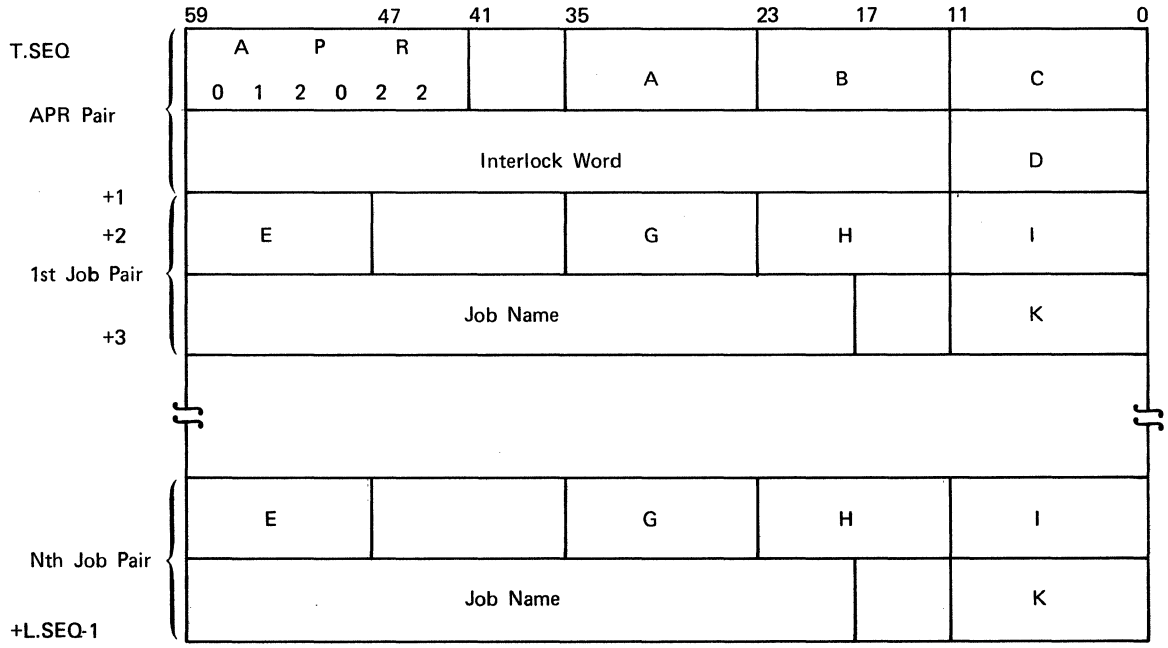
Y Status of RBS.

0 Get RBs.

1 Release RBs.

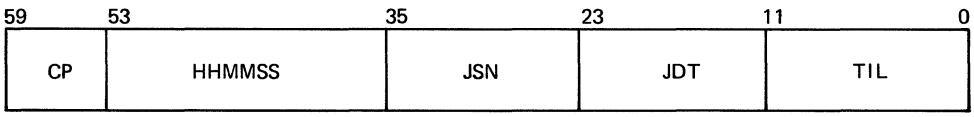
T.DOT is an entry point to CMR and thus is an entry of the table T.ENTRY of CMR. There is also a pointer to T.DOT in the header word of the Request Scheduling Table.

SEQUENCER TABLE



Field Name	Description
A	Maximum number of job entries (L.SEQ-2/2).
B	Number of jobs in sequencer table.
C	On/off/drop flag. 0 Off. 1 On. 2 Drop.
D	Table interlock flag.
E	Entry in use flag.
G	Entry drop flag.
H	Interval.
I	Clock.
K	Last known FNT address.

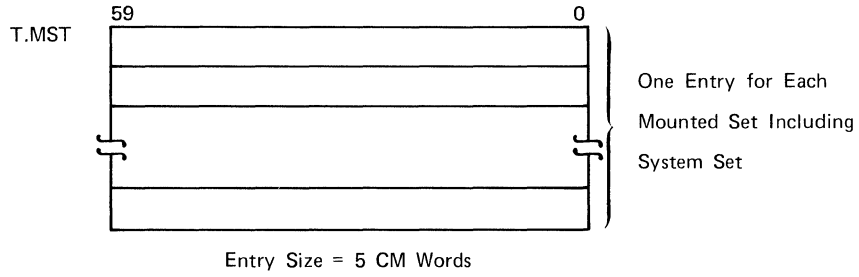
Interlock word when sequencer is active.



Field Name	Description
CP	Control point number.
HHMMSS	Hours, minutes, seconds in binary.
JSN	Job sequence number.
JDT	JDT ordinal.
TIL	Table interlock flag.

MOUNTED SET TABLE (MST) OVERVIEW

The mounted set table follows the CMR installation area.



Byte 4 of word 7 in the CMR pointer area contains the address of T.MST divided by 10_8.

MST ENTRY

	59	47	35	29	23	17	11	0
W.MSVSN	C.MSVSN Master Device VSN†				C.MSMFO Mainframe Ordinal		C.PFMIL L K J I H G F E D C B A	
W.MSSN	C.MSSN Set Name†				C.MSPEOI Purge EOI			
W.MSPTR	C.MSSMT Pointer to First RB of SMT	C.MSRBR First RBT word pair ordinal of DAM	C.MSEQT Master Dev. Equip. Type Max. Number Files/100_8† ††		C.MSEST Master Device EST Ordinal†		C.MSACT No. Active Jobs Accessing Set Maximum No. of Members† ††	
W.MSPFC	C.MSPFC Primary PFC First RBT Word Pair	C.MSPFC1††† Auxiliary PFC First RBT Word Pair	C.MSPFCS PFC Size in PRUs/10_8		C.MSCEOI M N P PFC Current EOI (PRU Offset)			
W.MSPFD	C.MSPFD Primary PFD First RBT Word Pair	C.MSPFD1††† Auxiliary PFD First RBT Word Pair	C.MSHPN Hashing C.MSHPS Number of Hash Points Left Shift Count		C.MSDFR Default File Retention Period		Reserved	

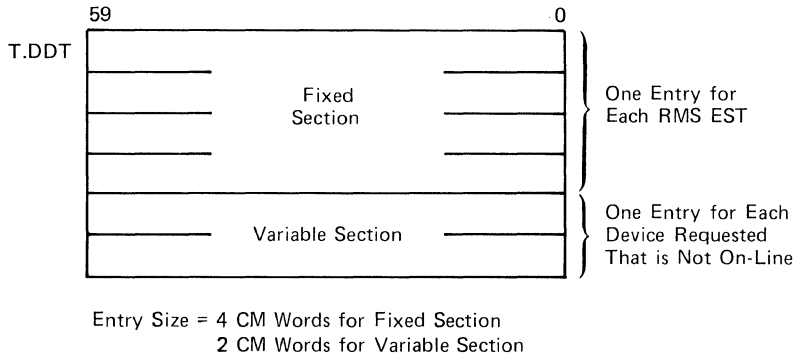
† Set up by deadstart for postdeadstart.
†† Not used by system after postdeadstart.
††† Not used currently but reserved by system.

Field Name	Description
A	Reserved.
B	PFM interlock (S.MSPFMI).
C	Utility interlock (S.MSUTIL).
D	Set interlock (S.MSSETI).
E	System bit (S.MSSYS).†
F	PF default set (S.MSPF).†
G	Queue set (S.MSQ).†
H	System default set (S.MSSCR).†
J I	Deadstart action (S.MSACT).†
	00 Check.
	01 Initialize.
	10 Modify.
K	Reserved (S.MSBF).
L	RB conflict (S.MSTRCF).† ††
M	Wrap-around flag for PFC (S.PFCW).
N	Passwords defined in label (S.MSPDL).
P	Universal permissions (S.MSUP).
	1000 Control.
	0100 Modify.
	0010 Extend.
	0001 Read.

† Set up by deadstart for postdeadstart.
†† Not used by system after postdeadstart.

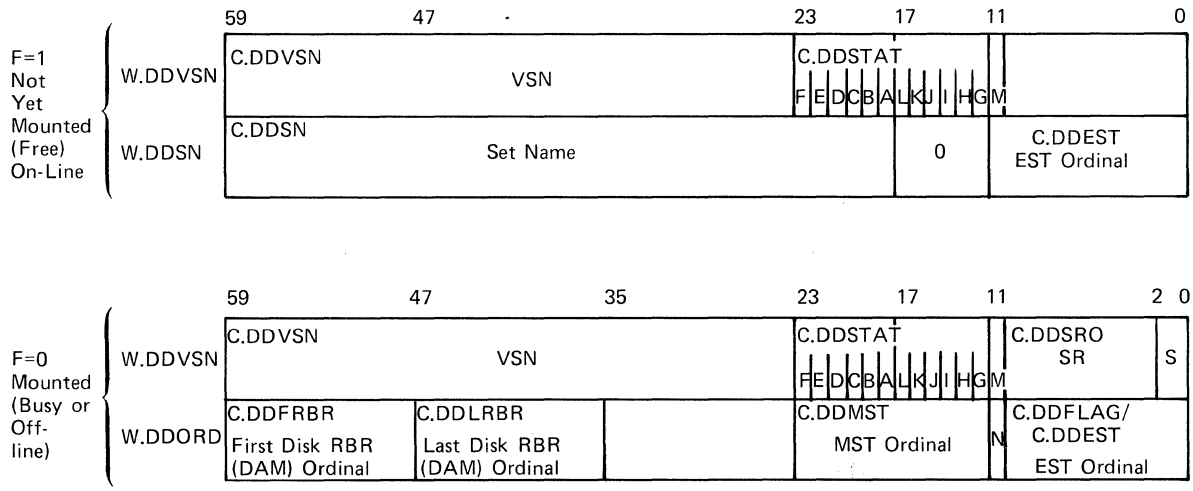
DISMOUNTABLE DEVICE TABLE (DDT) OVERVIEW

The dismountable device table follows the MST in CMR.



Byte 2 of word 7 in the CMR pointer area contains the address of T.DDT divided by 10_8.

DDT ENTRY (FIXED SECTION)



C.DDSTAT contains EST status as follows:

Field Name	Description
A	Request idle (S.DDIDLE).
B	Busy (S.DDBUSY).
C	Free (S.DDFREE).
D	Off (S.DDOFF).
E	Relabel during deadstart (S.DDRLB); during postdeadstart, SR assigned to unit.
F	SN bit; device not mounted if set (S.DDSN).
G	System device (S.DDSYS).
H	PF device (S.DDPF).
I	Queue device (S.DDQ).
J	Master device (S.DDMSTR).
K	Preallocated (S.DDPRE).
L	Recording mode; full-track if set (S.DDRM).

C.DDSRO contains EST status as follows:

Field Name	Description
M	Overlap seek in progress.
S	Overlap seek status.
SR	Stack request ordinal assigned to unit (after deadstart completes).

C.DDFLAG/C.DDEST contains EST status as follows:

Field Name	Description
N	Gap sector flag; no gap sectors if set (S.DDNGS).

Statistic Area

W.DDSTA

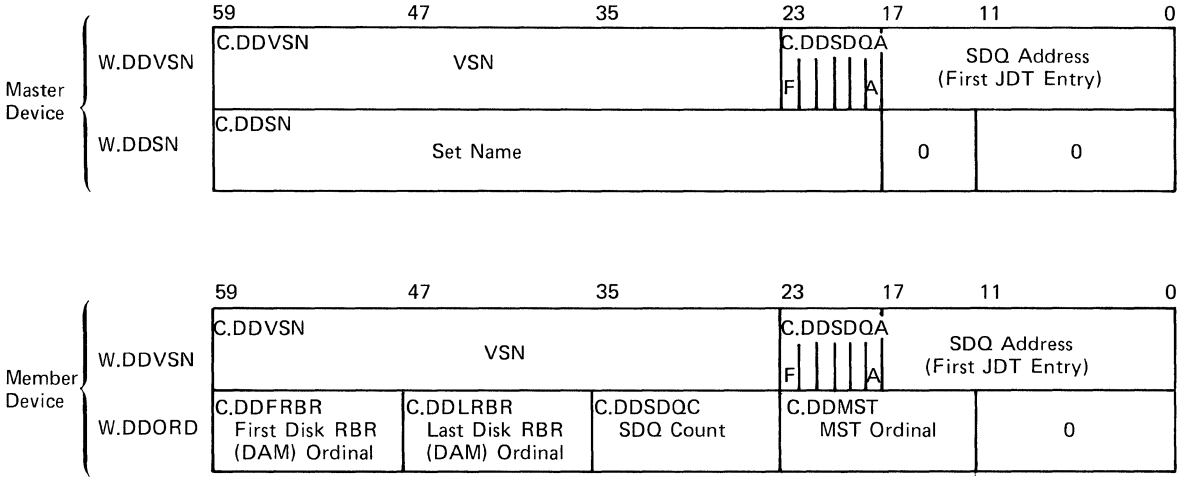
59	47	35	23	11	0
	C.DDNFC New File Count	C.DDWAT Bypass Count	†	C.DDSR † Number of Accesses	

W.DDSUM

59	47	35	23	11	0
C.DDACT Unit Activity	C.DDNFC † Previous File Count	C.DDWAT † Bypass Count	†	C.DDSR † Prenumber of Accesses	

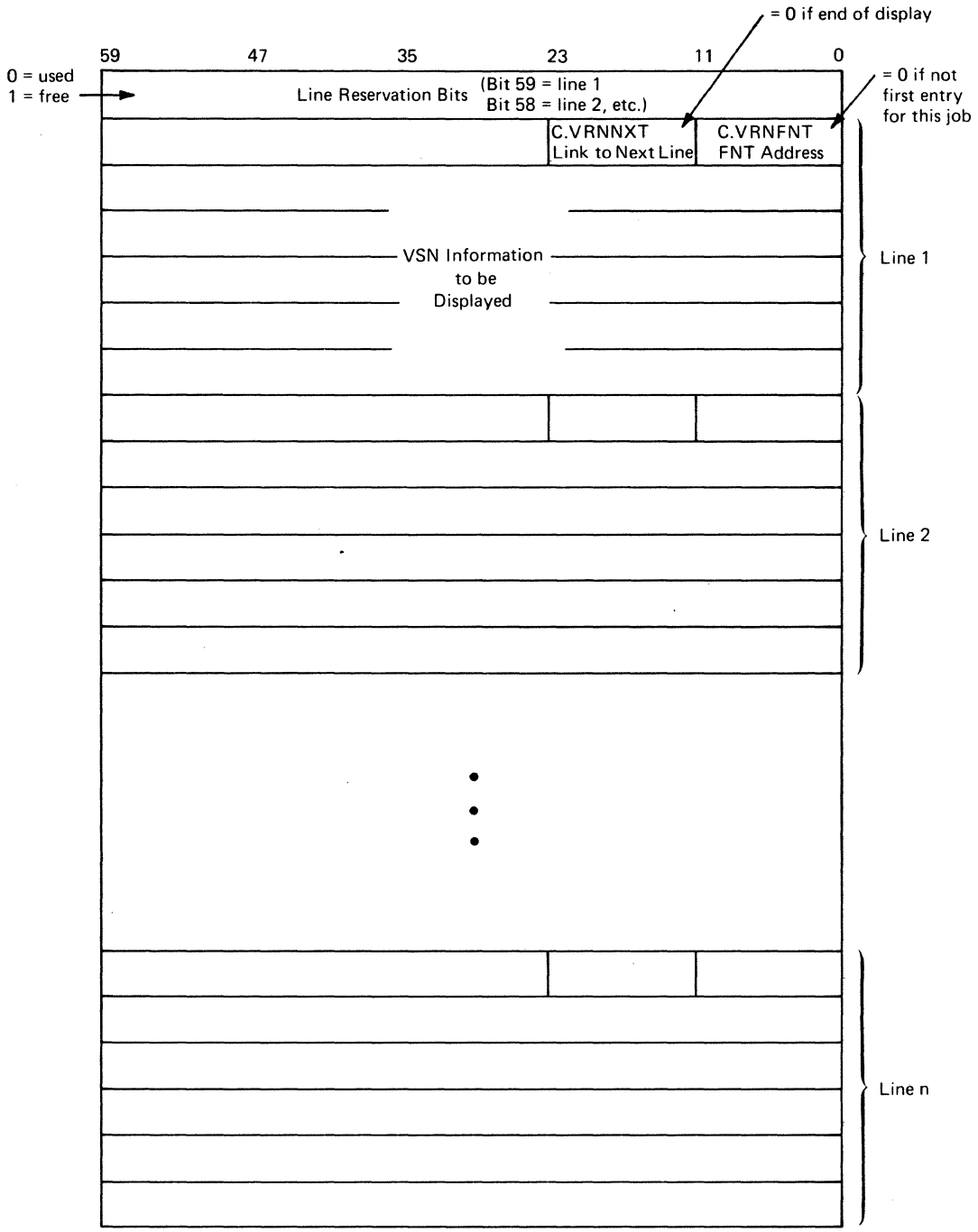
† Used as temporary save area by deadstart.

DDT ENTRY (VARIABLE SECTION)



Field Name	Description
A	Recording mode; full-track if set (S.DDRMW).
F	SN bit; device not mounted if set (S.DDSN).

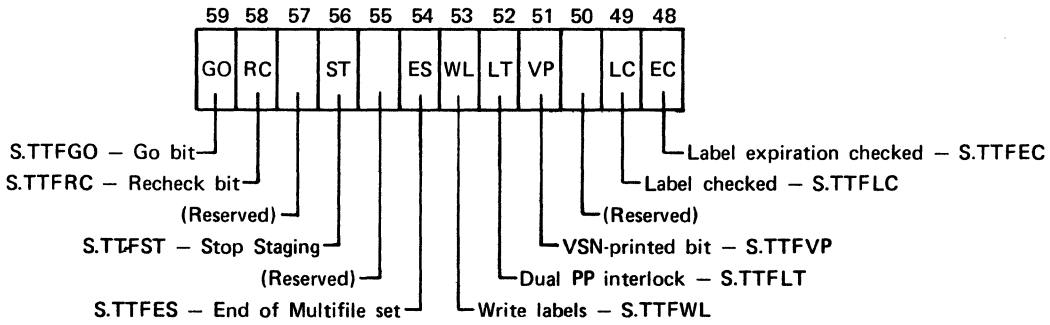
VSN BUFFER


= 0 if end of display
59
47
35
23
11
0
0 = used
1 = free
Line Reservation Bits
(Bit 59 = line 1
Bit 58 = line 2, etc.)
= 0 if not
first entry
for this job
C.VRNNXT
Link to Next Line
C.VRNFNT
FNT Address
VSN Information
to be
Displayed
Line 1
Line 2
Line n

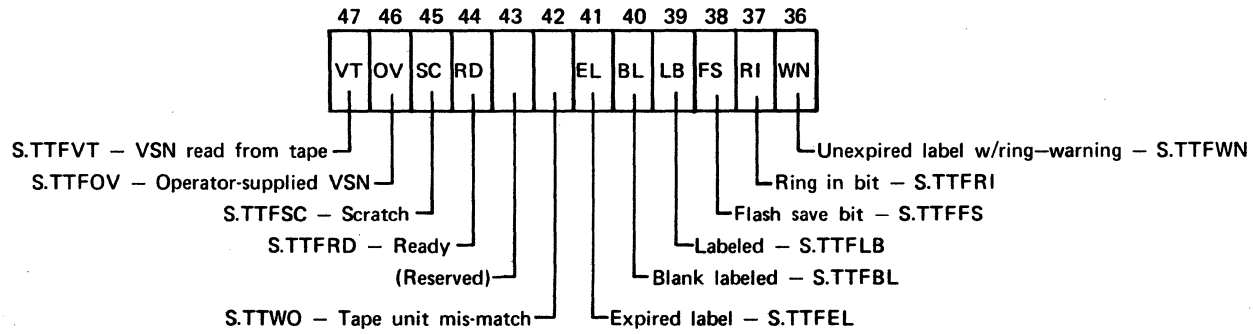
TAPES TABLE

	59	47	35	29	23	17	11	0
	EST Ordinal (Binary)	FNT Address	Control Point Number		MT or NT (Display Code)		EST Ordinal (Display Code)	
W.TFLN1	Label Name							
	Label Name						Position Number	
	Edition Number	Retention Cycle	Creation Date					
W.TREEL	Multifile Name				Reel Number			
W.TFLGS	C.TFLGS Flag Bits		Reserved				Communication Area†	
W.TVRN	Volume Serial Number of Current Reel				Last Good Checksum††		Last Good Byte Count††	
W.TVRN1	Volume Serial Number of First Reel				PRU Number of Last PRU That Got Noise Warning 1			

C.TFLGS - Job-Oriented Flag Bits

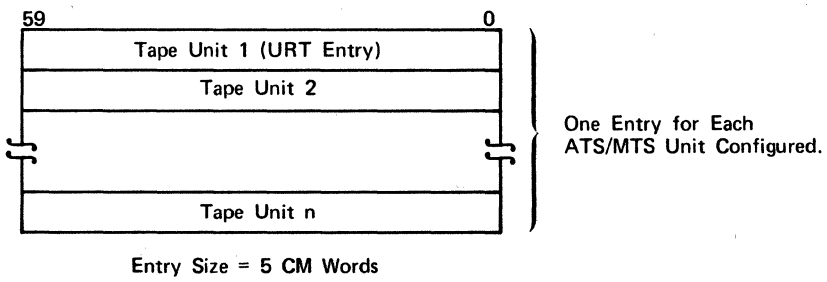


C.TFLGS+1 - Unit-Oriented Flag Bits

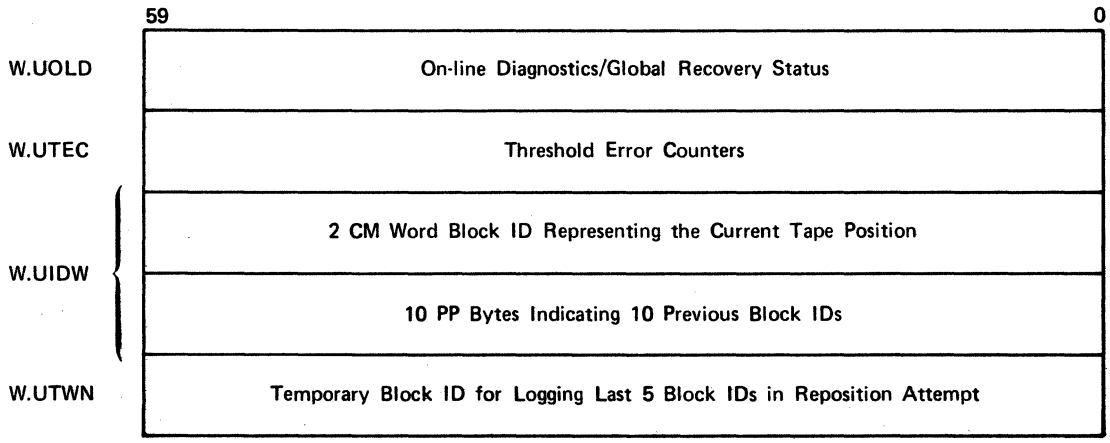


† 21A/21B slave-master PP communication area.
†† These bytes are not used by 66x drivers.

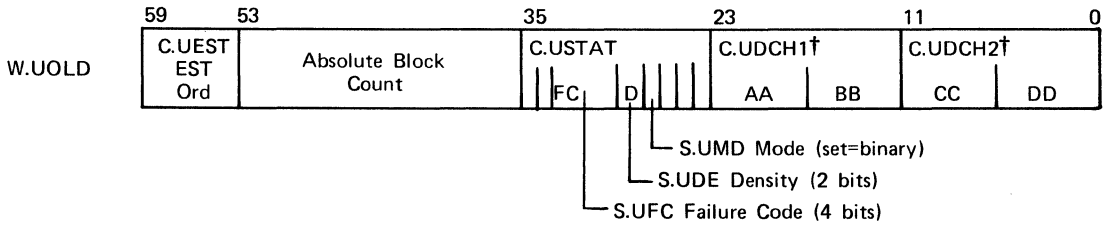
TAPE UNIT RECOVERY TABLE (URT) OVERVIEW



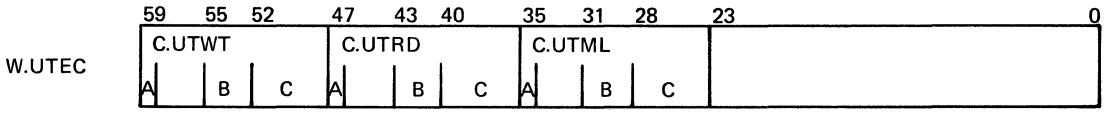
TAPE UNIT RECOVERY TABLE ENTRY



W.UOLD FORMAT

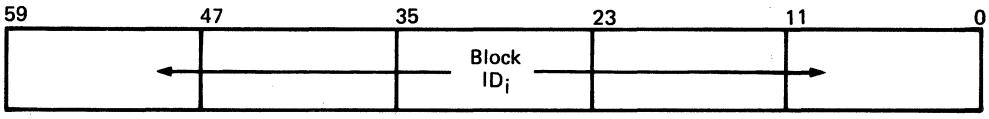


W.UTEC FORMAT



Field Name	Description
A	Reel error reached.
B	Threshold error count (3 bits).
C	Running error count (5 bits).

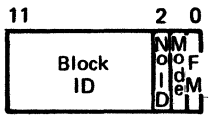
W.UIDW FORMAT



W.UIDW Two words containing the block IDs (five IDs per word) of the previous 10 tape blocks. When updated with most recently processed blocks, new ID is placed in byte 4 of second word, and all preceding block IDs move forward (left shift, end-off); the contents of word 1, byte 0 is lost.

†Channels in the order they are removed from the EST.

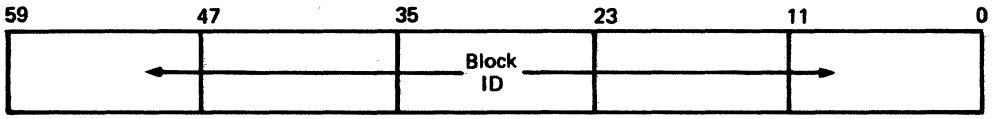
Each 12-bit block ID field has the following format:



Bit	Description
0	If set, indicates file mark.
1	If set, indicates seven-track even parity tape block.
2	If set, indicates bits 11 through 3 contain software code rather than the actual hardware block ID.
11-3	Block ID (if bit 2 clear) or software code (if bit 2 set) as follows:

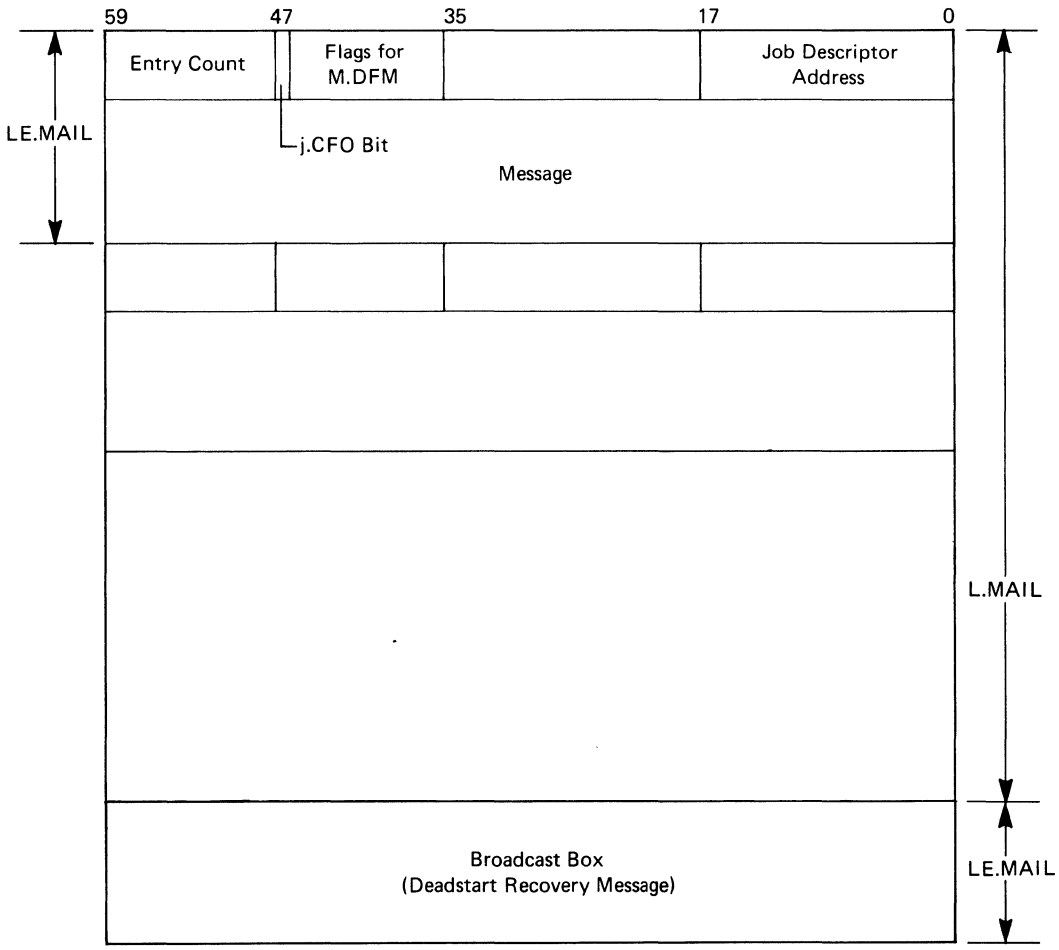
Code	Description
000	Unknown or unusable block ID.
001	Loadpoint.
002	Block resulted from unrecovered read error.
003	Block resulted from unrecovered write error.
004	Block resulted from error on SKIPF.

W.UTWN FORMAT

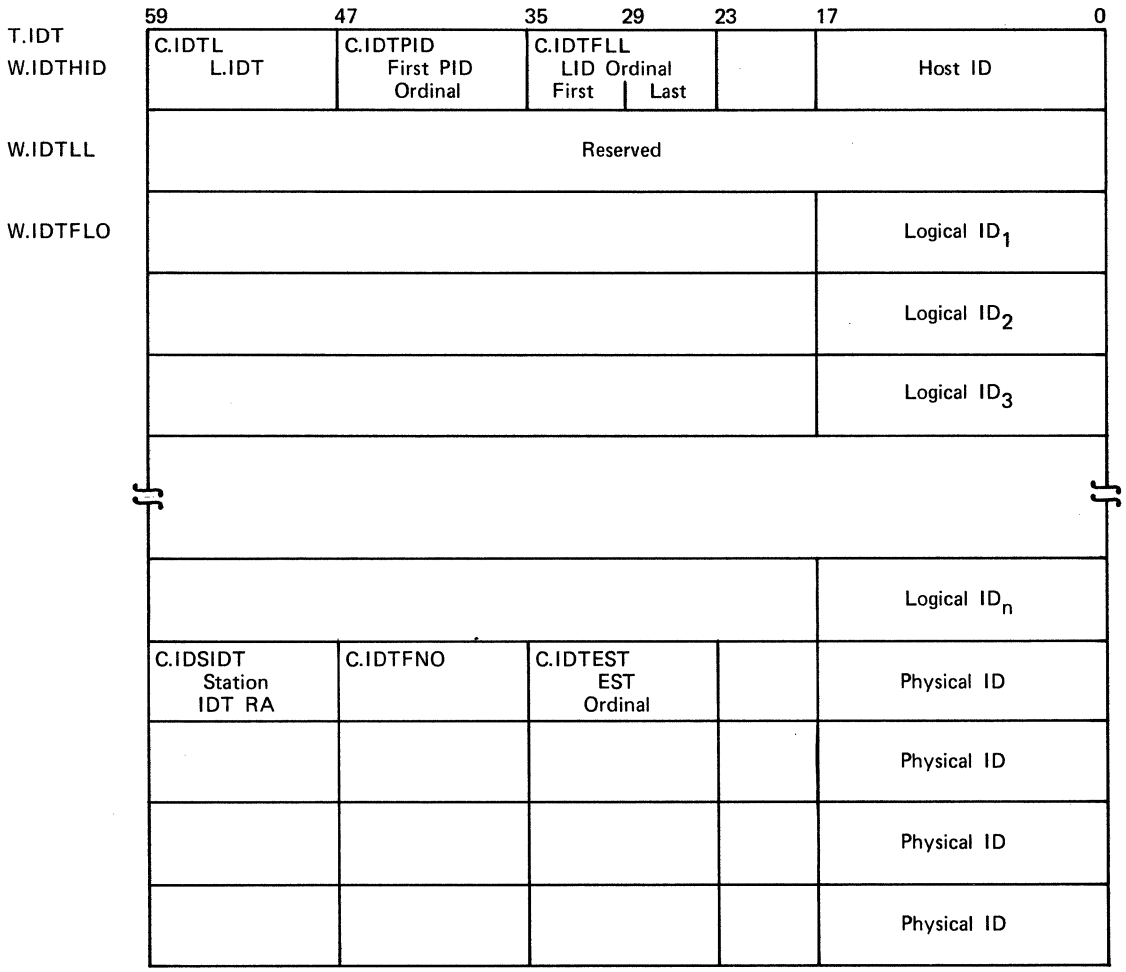


W.UTWN Temporary location of five blocks ID read from tape following reposition attempt; IDs are compared with corresponding IDs in W.UIDW to verify correct repositioning.

SCHEDULER MAILBOX BUFFER



LOGICAL ID TABLE



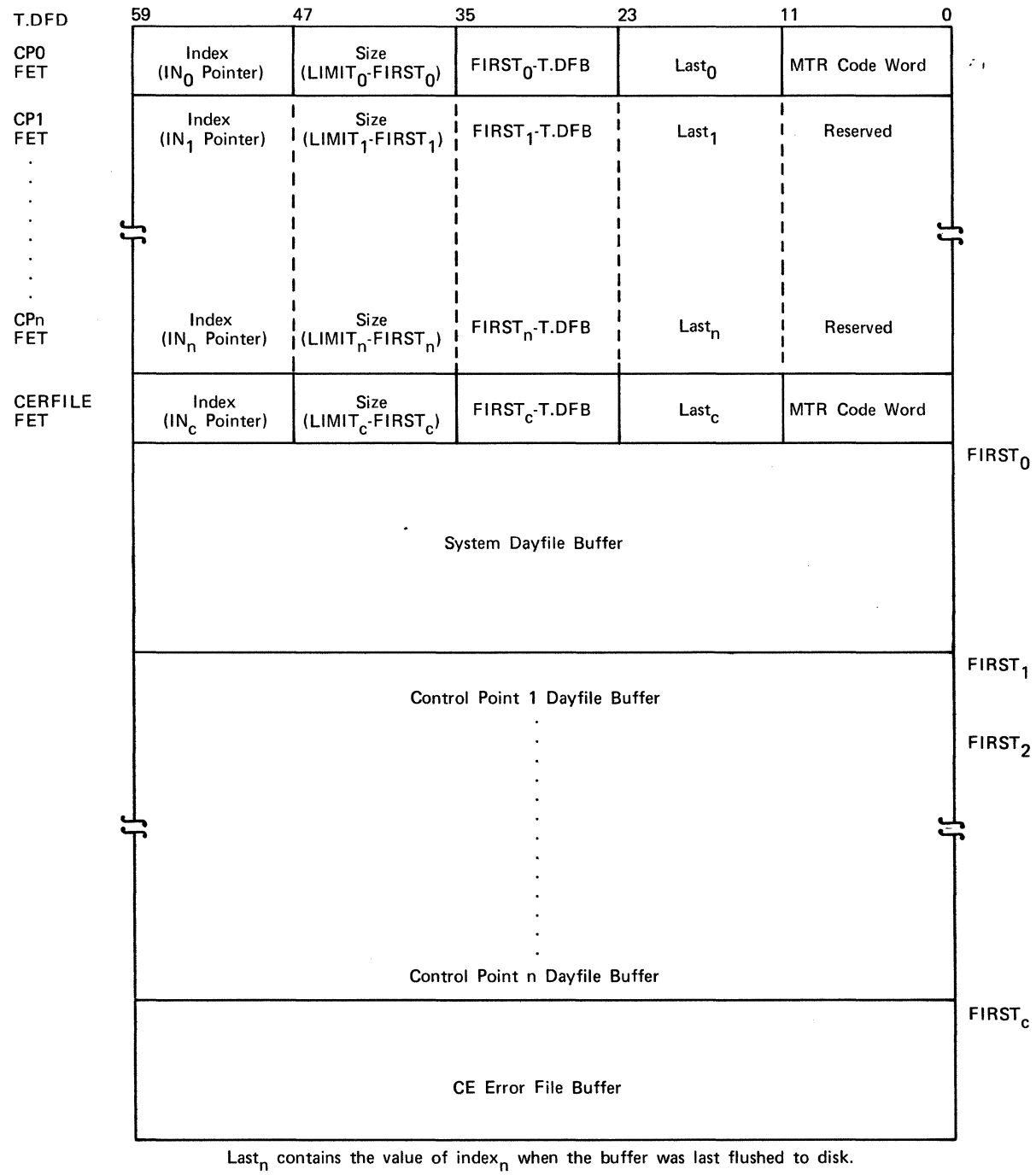
Byte 4 of word 66 in the CMR pointer area contains the address of T.IDT divided by 10_8. Byte 3 of word 66 in the CMR pointer area contains the length of T.IDT.

Word	Byte	Bit	Description
W.IDTHID			Host word.
	C.IDTL	59-48	Contains the length of the ID table. Default is 40_8. (L.IDT).
	C.IDTPID	47-36	Contains the first physical ID ordinal.

Word	Byte	Bit	Description
	C.IDTFLL	35-24	Contains the first logical ID ordinal in the upper 6 bits and the last logical ID ordinal in the lower 6 bits.
		17-0	Contains the host ID as three letters or digits. Third character must be a letter.
W.IDTLL			Reserved.
W.IDTFLO			Logical ID entries
		17-0	Contains a logical ID as three letters or digits.
Link Words	C.IDSIDT	59-48	Contains the relative address of the station ID table. The station ID table is a copy of the central memory resident ID table, but the SIDT also contains lists of the logical ID's associated with linked mainframes.
	C.IDTFNO	47-36	Contains relative address of station PID information area.
	C.IDTEST	35-24	Contains the EST ordinal of the PID.
		17-0	Contains a physical or link ID as three letters or digits.

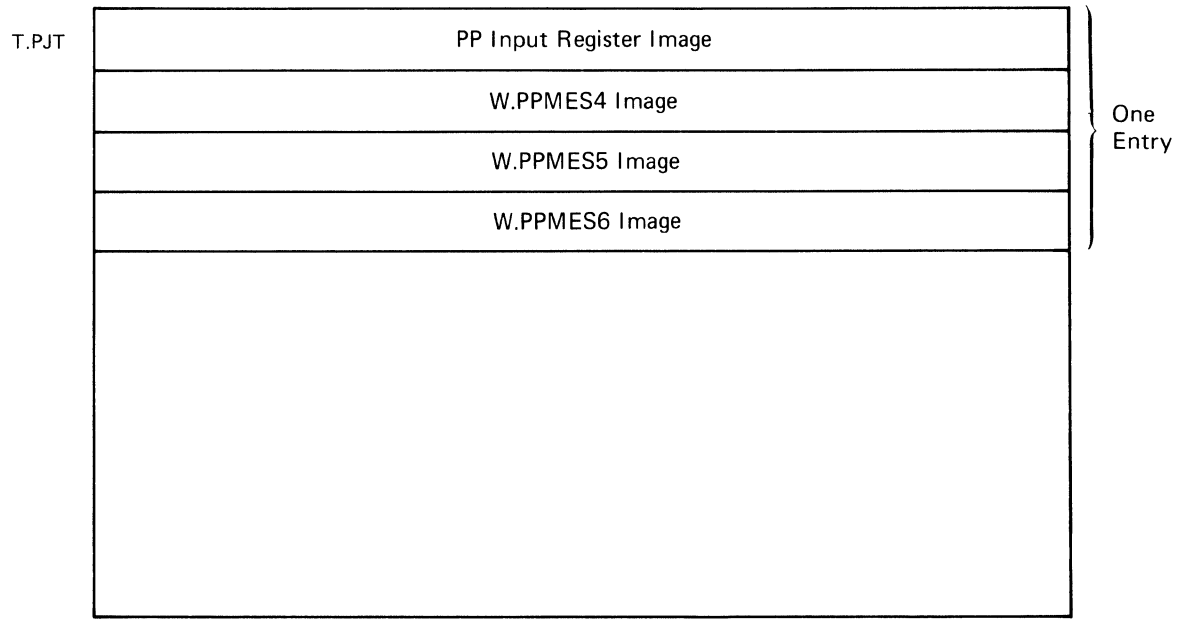
DAYFILE FET AND BUFFER AREA

Byte 0 of word 3 in the CMR pointer area contains the address of T.DFD divided by 10_8.



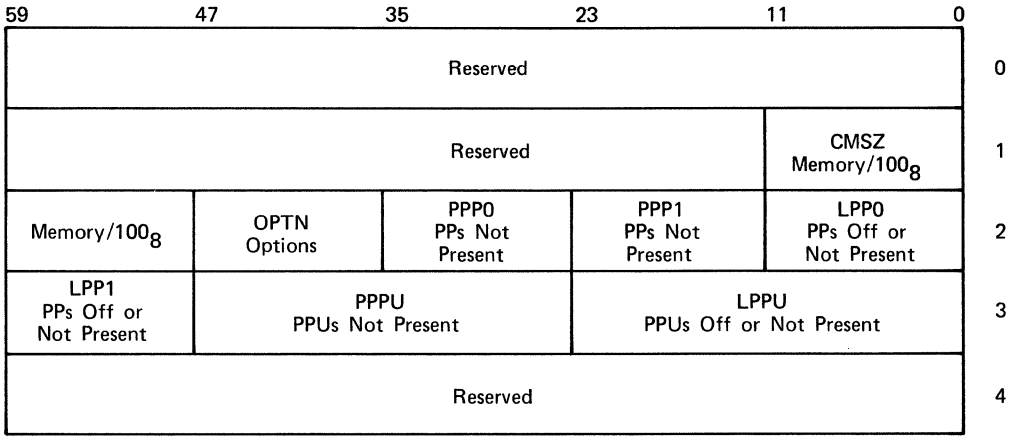
$Last_n$ contains the value of $index_n$ when the buffer was last flushed to disk.

PERIPHERAL JOB TABLE



Byte 3 of word 26 in the CMR pointer area contains the first word address divided by 10_8 of T.PJT. Byte 4 of word 26 in the CMR pointer area contains the last word address divided by 10_8 of T.PJT.

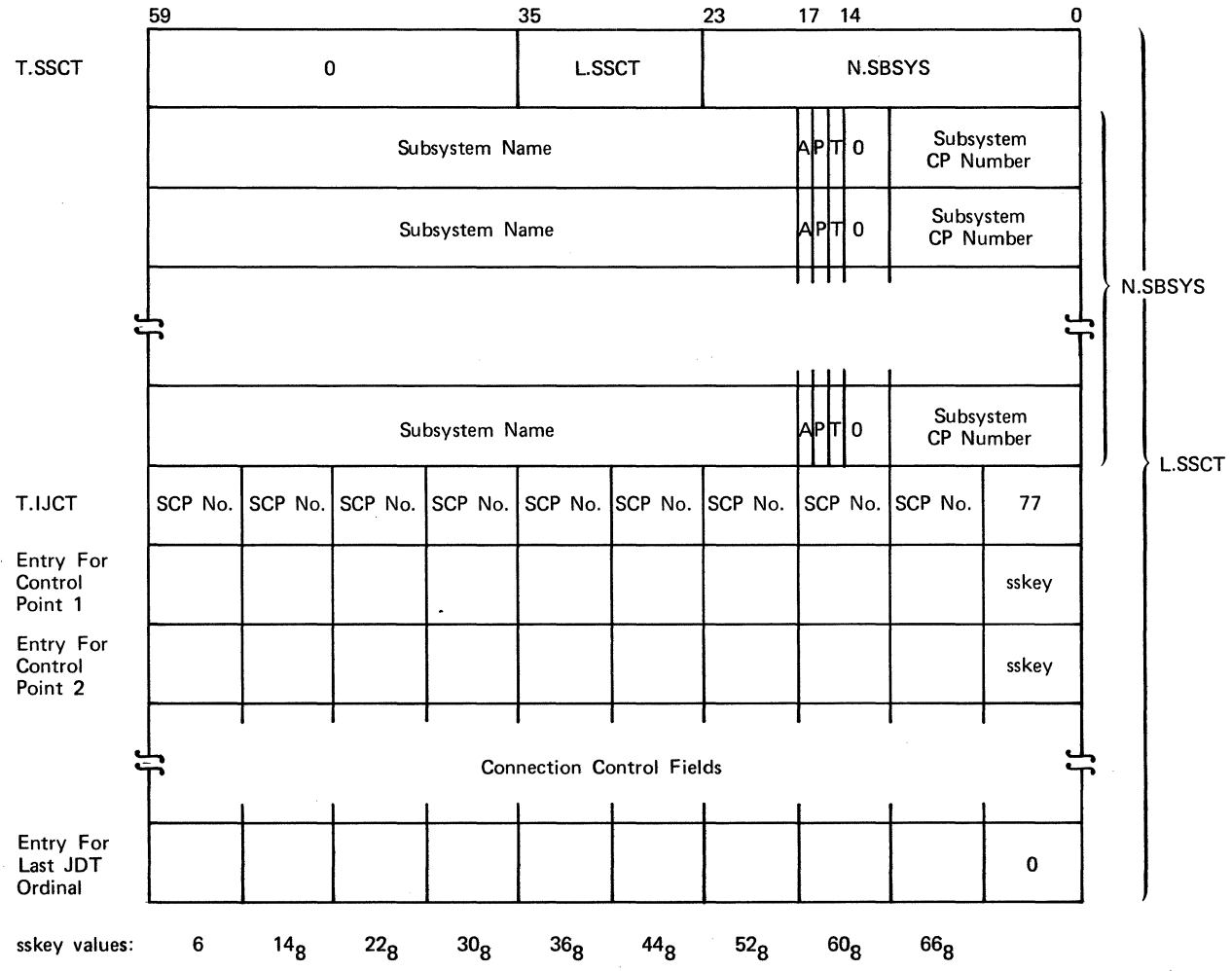
MAINFRAME ATTRIBUTE BLOCK



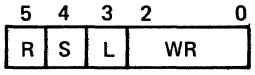
Word	Bits	Description
1, 2	11-0 59-48	Memory size/100_8.
2	47	Reserved.
	46-45	7000 type flags.
		0 Non-7000.
		1 CYBER 76 Model A.
		2 CYBER 76 Model B.
		3 CYBER 176.
	44	Reserved.
	43	ILR.
		1 No ILR.
	42	SCR.
		1 No SCR.
	41	Reserved.
	40	CPU-0 instruction stack.
		1 No stack.
	39	CMU.
		1 No stack.

Word	Bits	Description
	38	CEJ/MEJ. 1 No CEJ/MEJ.
	37	CPU-1. 1 No CPU-1.
	36	CPU-0. 1 No CPU-0.
2	35-34	Not used.
	33-24	A set bit indicates that the corresponding PP numbered 9 through 0 is not physically present.
	23-22	Not used.
	21-12	A set bit indicates that the corresponding PP numbered 19 through 10 is not physically present.
	11-10	Not used.
	9-0	A set bit indicates that the corresponding PP numbered 9 through 0 has been turned off or is not physically present.
3	59-58	Not used.
	57-48	A set bit indicates that the corresponding PP numbered 19 through 10 has been turned off or is not physically present.
	47-37	Not used.
	36-24	A set bit indicates that the corresponding PPU numbered 12 through 0 is not physically present.
	23-13	Not used.
	12-0	A set bit indicates that the corresponding PPU numbered 12 through 0 has been turned off or is not physically present.

SUBSYSTEM CONTROL TABLE



Connection control field is as follows:



Field Name	Description	Set/Increment	Clear/Decrement
R	Reserved	---	---
S	Swap out.	SF.SWPO	SF.SWPI
L	Long-term connection.	SF.SLTC	SF.CLTC
WR	Wait response count.	CALLSS	SF.ENDT

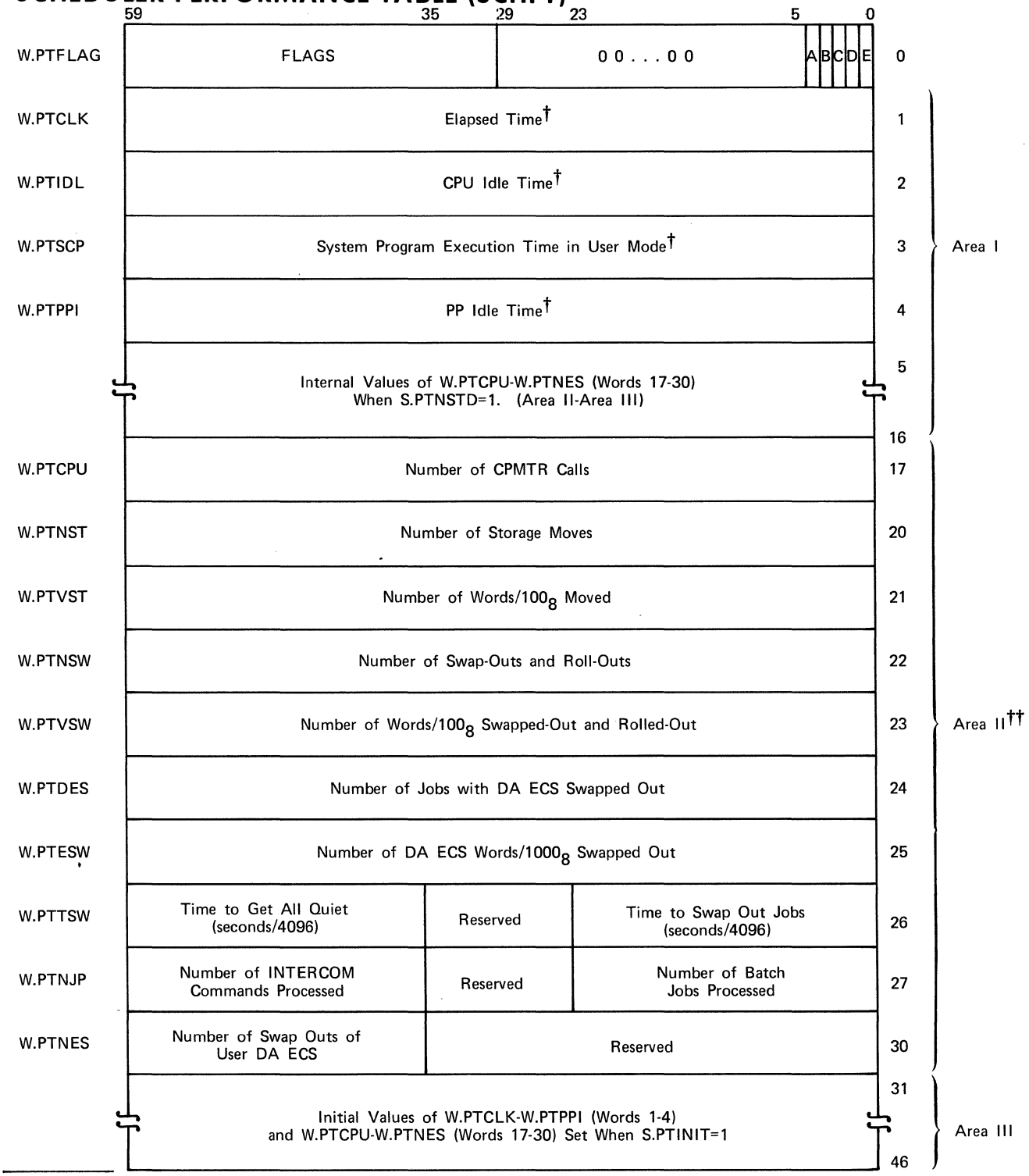
Bit 15 (T) is set by SSF when performing termination processing on a system control point which has its error flag set. (Bit 17 remains set during this time.)

Bit 16 (P) permits a control point to attain system control point status as the specified subsystem even if the job is not system origin (initiated by operator).

When a subsystem is active, the following actions occur.

- Bit 17 (A) is set to 1.
- An sskey value is assigned, which specifies a column of connection control fields.
- This sskey value is put in the word for the subsystem job control point number.
- The subsystem CP number is put in the subsystem word and in the T.IJCT field specified by sskey.

SCHEDULER PERFORMANCE TABLE (SCHPT)



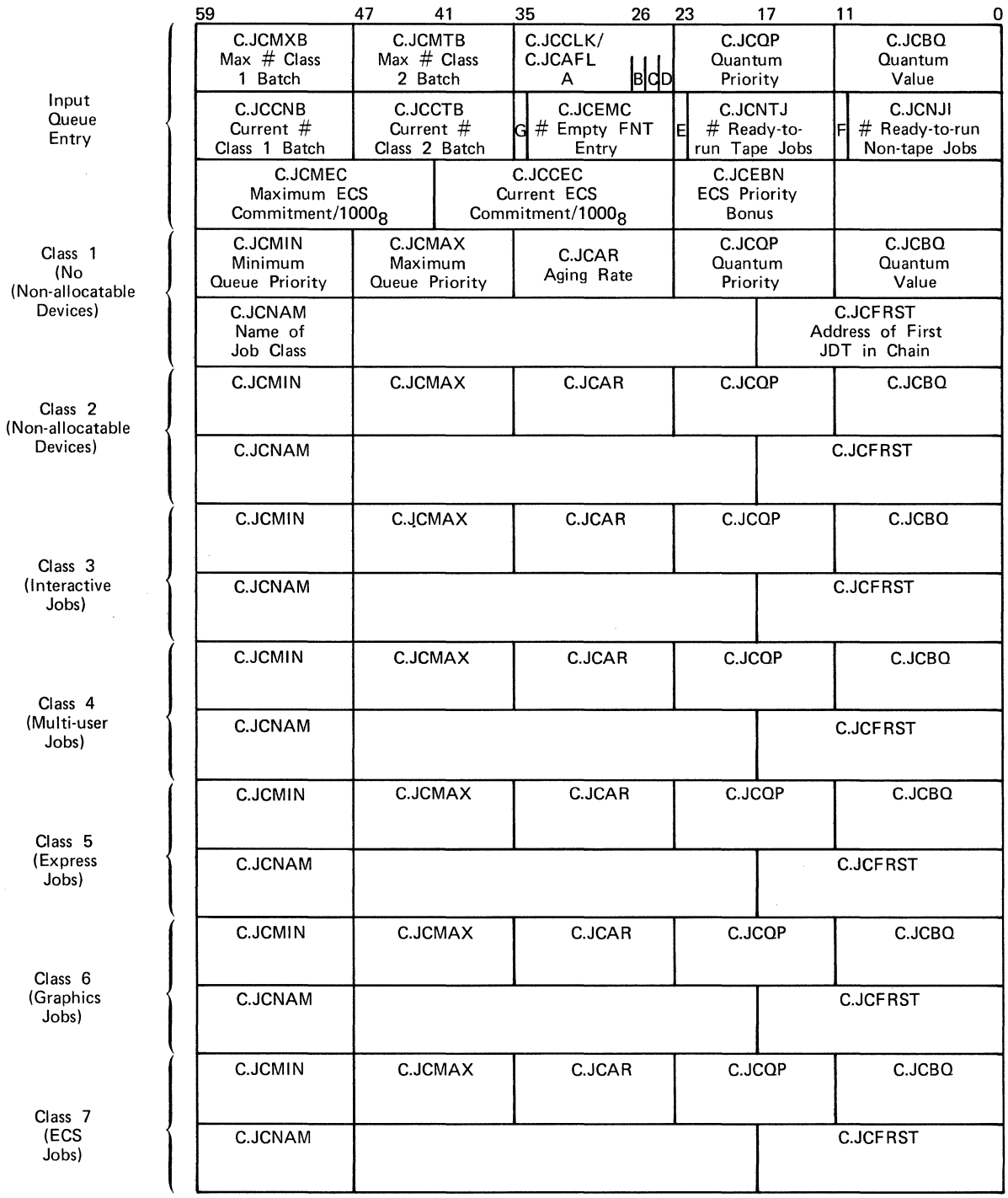
†Area I, words W.PTCLK-W.PTPPI, contains current times when S.PTNSTD=0; interval times when S.PTNSTD=1. The unit used to measure the interval of time is one second/4096.

††Area II always contains current values regardless of the flags.

Address of this table is T.SCHPT.

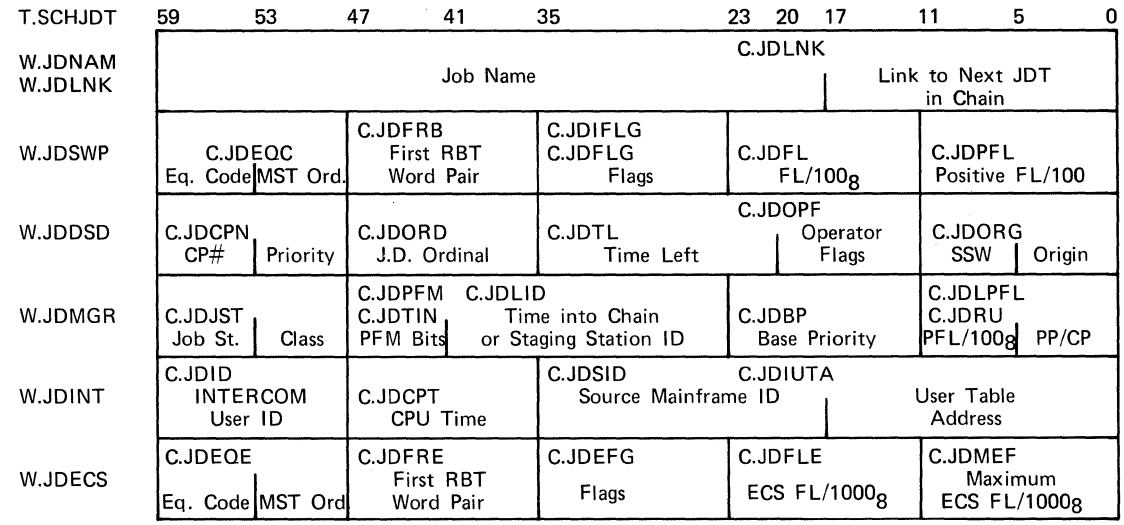
Field Name	Bit Name	Description
A	S.PTFJOB	First job execution started.
B	S.PTBNCH	Benchmark option selected.
C	S.PTSTOP	Stop data gathering.
D	S.PTINIT	Initialization of TSPT.
E	S.PTNSTD	Manual/benchmark option selected.

SCHEDULER JOB CONTROL AREA



Field Name	Description
A	Anticipated FL/1000$_8$.
B	1=Increment to AFL for INTERCOM (AFL.INT) has been added (S.JCAFL).
C	Reserved.
D	1=1IB failed to initiate a job last time called (S.JC1IB).
E	1=Fixed priority tape jobs in input queue.
F	1=Fixed priority nontape jobs in input queue.

SCHEDULER JOB DESCRIPTOR TABLE ENTRY



Byte 4 of word 60 in the CMR pointer area contains the address of T.SCHJDT.

W.JDNAM(0)

Bit	Description
59-18	Job name as found in job input FNT.

W.JDLNK

Bit	Description
17-00	Address of next job descriptor in the chain. (C.JCFRST in JCA points to first entry in chain; a zero field denotes last entry in chain.)

W.JDSWP(1)

Byte	Bit	Field	Description
C.JDEQC(0), C.JDFRB(1)			Equipment code, MST ordinal, and first RBT number of the swap file (F.JDSWT in word W.JDMGR is set); else contains first subpage address of ECS swap file.
C.JDFLG(2), C.JDIFLG(2)	35	S.JDBCB(43_8)	Set if recovery took place.
	34	S.JDNRR(42_8)	Set if job cannot be rerun.
	33	S.JDLGI(41_8)	Set if no swap file exists and control point area must be initialized (for example, LOGIN command).

Byte	Bit	Field	Description
	32	S.JDLGO(40_8)	Set if no swap file is to be generated for this INTERCOM job (for example, LOGOUT command).
	31	S.JDNJ(37_8)	Set to indicate control statements read from INTERCOM area.
	30	S.JDECS(36_8)	Set if swap file is on ECS.
	29	S.JDSKFL(35_8)	Set if FL is to be skipped on swap file.
	28	S.JDROLL(34_8)	Set if job cannot be swapped out.
	27	S.JDFNT(33_8)	Set if 1AJ should not search FNT table.
	26	S.JDFAZ(32_8)	Set if file at control point 0.
	25	S.JDINTR(31_8)	Set to terminate INTERCOM job on recovery deadstart.
	24		Unused.
C.JDFL(3)			FL/100, including the job control block, which is created when job is swapped out, needed to swap in this job.
C.JDPFL(4)			Relative starting address/100 of the job control block when present for job swapping; otherwise, contains zero.

W.JDDSD(2)

Byte	Bit	Field	Description
C.JDCPN(0)			Control point number in upper 6 bits. (Set when job is in execution or rolled out.) In lower 6 bits, contains job or rerun priority.
C.JDORD(1)			Job descriptor (JDT) ordinal.
C.JDTL(2)			Job time left (upper 12 bits).

Byte	Bit	Field	Description
C.JDOPF(3)			Flags set by operator.
	23-21		Low order 3 bits of time left.
	20		Currently not used.
	19	S.JDTLI(23_8)	Set if job is to be temporarily locked into a CP.
	18	S.JDEXP(22_8)	Set if job is to be placed in express queue.
	17	S.JDGO(21_8)	Set if operator typed GO.
	16	S.JDNS(20_8)	Set if job must not be swapped out or rolled out when at a control point.
	15	S.JDLOK(17_8)	Set if job must not be brought to control point.
	14-12		Error codes: F.JDKILL, F.JDDROP, F.JDRRUN, F.JDRRNP, F.JDTI.
C.JDORG(4)	11-6		Sense switches.
	5	S.JDINT	Set for a standard INTERCOM job.
	4	S.JDMUJ	Set for a multiuser job.
	3	S.JDGR	Set for a graphics job.
	2	S.JDRT	Set for a real-time job.
	1-0		Currently not used.

W.JDMGR(3)

Byte	Bit	Field	Description
C.JDJST(0)	59-54		Values describe the job status.

Value	Name	Job Status
7x	F.JDWMM	Job is waiting for MMF action (GETPF, SAVEPF, PURGE).
6x	F.JDWPK	Job is waiting for permanent pack.
5x	F.JDWIA	Job is waiting for INTERCOM action (in INTERCOM queue).

Byte	Bit	Field	Description		
			Value	Name	Job Status
			4x	F.JDWOA	Job is waiting for operator action (in operator action queue).
			3x	F.JDWDA	Job is waiting for device assignment (in device queue).
			2x	F.JDWPF	Job is waiting for a permanent file availability (in permanent file queue).
			1x	F.JDWCM	Job is waiting for central memory (in central memory queue).
			0x	F.JDLMB	Job is waiting for entry in a scheduling structure.
			x3	F.JDWCC	Job is being swapped or rolled out.
			x2	F.JDACT	Job is currently executing at a control point (is active).
			x1	F.JDSWI	Job is being swapped or rolled in.
			x0		Job is swapped or rolled out.
	53-48		Job class values.		
			Value	Name	Job Status
			07	F.JDECS	ECS jobs.
			06	F.JDGRA	Graphics job.
			05	F.JDEXP	Express handling was requested for this job.

Byte	Bit	Field	Description		
			Value	Name	Job Status
			04	F.JDMUJ	Multiuser job.
			03	F.JDINT	Standard INTERCOM job.
			02	F.JDBNA	Batch job with nonallocatable device requirements.
			01	F.JDBAT	Batch job with no nonallocatable device requirements.
C.JDPFM(1)			Information used by permanent file manager.		
	47		Purge bit.		
	46		Exclusive access desired.		
	45		Control permission desired.		
	44		Modify permission desired.		
	43		Extend permission desired.		
	42		Read permission desired.		
C.JDTIN(1)	41-24		Time at which job descriptor was attached to job class chain.		
C.JDLID	41-24		ID of the staging station when a job has been swapped to the multimainframe queue.		
C.JDBP(3)	23-12		Job base priority.		
C.JDRU(4), C.JDLPFL(4)	11-6		Positive field length/100_8 of job control block created at swap-out time.		
	5-0		Ratio of PP time to CPU time used by job during its last execution (if job is swapped out) or since start of job (if job is rolled out).		

W.JDINT(4)

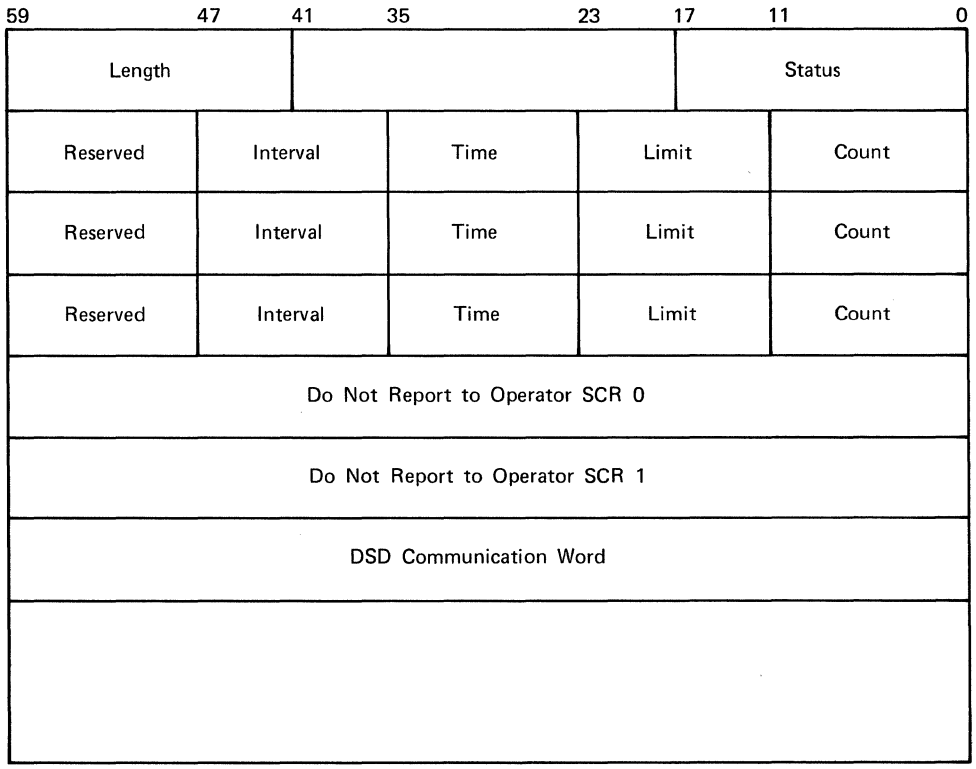
Byte	Bit	Field	Description
C.JDID(0)	59-48		INTERCOM user ID associated with remote batch job.
C.JDCPT(1)	47-36		CPU time, in seconds, used by this job.
C.JDSID(2)	35-18		Source mainframe identifier.
C.JDIUTA(3), C.JDIUTA+1	17-0		Address of user table for interactive or graphics jobs, or MUJ table for multiuser jobs.

W.JDECS(5)

Byte	Bit	Field	Description
C.JDEQE(0), C.JDFRE(1)			Equipment code, MST ordinal, and first RBT word pair number of the direct access ECS swap file for the job.
C.JDEFG(2)	25	S.JDEAS(0)	Set if the job's direct access ECS has been swapped out.
	24	S.JDCPS(1)	Set if swapped-out job is still assigned to a control point (ECS not swapped out).
C.JDFLE(3)			Current amount of direct access ECS assigned to the job, in 1000_8 word blocks.
C.JDMEF(4)			Maximum amount of direct access ECS that the job can obtain, in 1000_8 word blocks.

ERROR LOGGING STATUS TABLE

Byte 1 of word 23 in the CMR pointer area contains the address of T.ELST.



Values for status field are as follows:

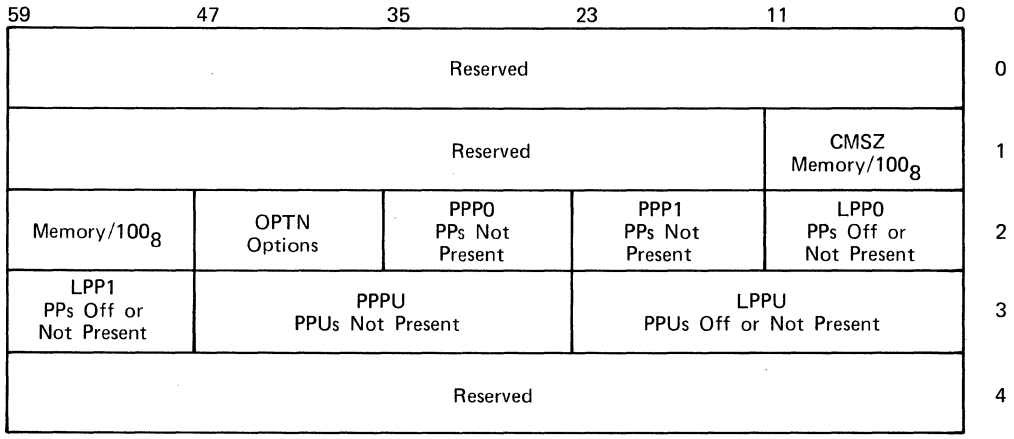
0 MTR testing SCR for errors.

1 1SC processing error in SCR.

3 1SC in dedicated mode.

5 SCR monitoring not activated.

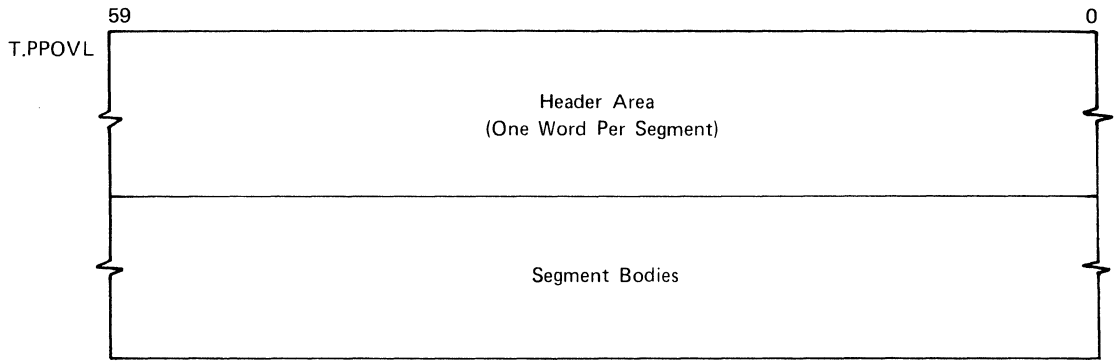
MAINFRAME ATTRIBUTE BLOCK



Word	Bits	Description
1, 2	11-0 59-48	Memory size/100$_8$.
2	47	Reserved.
	46-45	7000 type flags. 0 Non-7000. 1 CYBER 76 Model A. 2 CYBER 76 Model B. 3 CYBER 176.
	44	Reserved.
	43	ILR. 1 No ILR.
	42	SCR. 1 No SCR.
	41	Reserved.
	40	CPU-0 instruction stack. 1 No stack.
	39	CMU. 1 No stack.

Word	Bits	Description
	38	CEJ/MEJ. 1 No CEJ/MEJ.
	37	CPU-1. 1 No CPU-1.
	36	CPU-0. 1 No CPU-0.
2	35-34	Not used.
	33-24	A set bit indicates that the corresponding PP numbered 9 through 0 is not physically present.
	23-22	Not used.
	21-12	A set bit indicates that the corresponding PP numbered 19 through 10 is not physically present.
	11-10	Not used.
	9-0	A set bit indicates that the corresponding PP numbered 9 through 0 has been turned off or is not physically present.
3	59-58	Not used.
	57-48	A set bit indicates that the corresponding PP numbered 19 through 10 has been turned off or is not physically present.
	47-37	Not used.
	36-24	A set bit indicates that the corresponding PPU numbered 12 through 0 is not physically present.
	23-13	Not used.
	12-0	A set bit indicates that the corresponding PPU numbered 12 through 0 has been turned off or is not physically present.

PP RESIDENT OVERLAY SAVE BUFFER

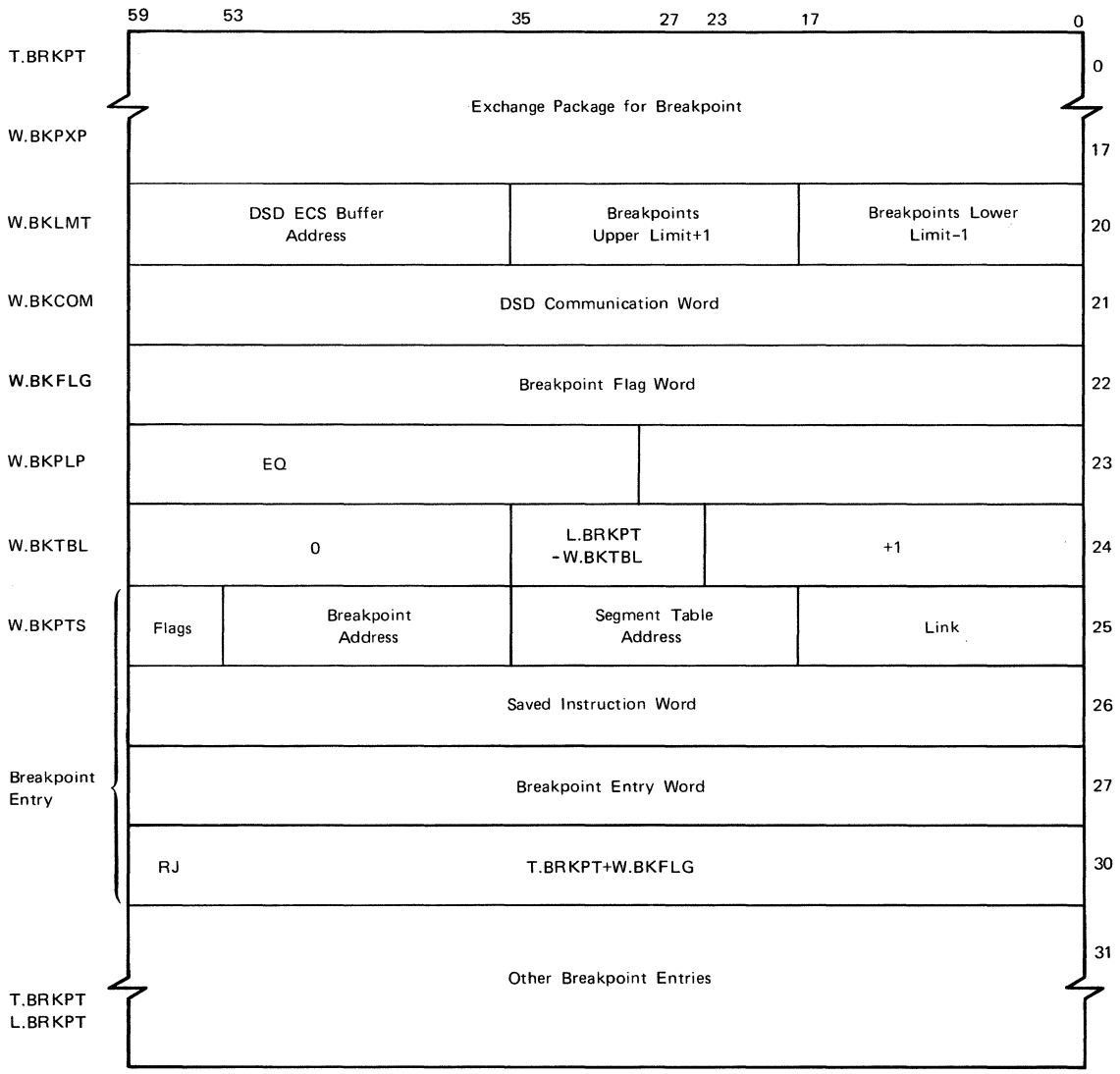


Header entry:

Bit	Description
59-48	Segment header ordinal + 5.
41-24	Absolute CM address of segment.
23-12	Nonzero flag for PP resident to differentiate between segment and overlay.
11-0	Segment length in CM words.

BREAKPOINT TABLE

Bits 59 through 36 of word 67 in the CMR pointer area contain the first word address of T.BRKPT.



W.BKLMNT (20)

Bit	Description
59-36	DSD ECS display buffer.
35-18	Last absolute address that may be breakpointed +1.
17-0	First absolute address that may be breakpointed -1.

W.BKCOM (21) (Set by DSD)

Bit	Description
59-24	Nonzero if bits 23 through 0 are zero.
23-12	Breakpoint entry relative address +4 (to be processed).
11-0	Mode flag. 1 Monitor. 0 User.

W.BKFLG (22)

Bit	Description
59-48	Processing flag (in octal). 400 Breakpoint bit. 200 Breakpoint processed. 100 Restart CPU. 1000 Release breakpoint.
47-30	Breakpoint entry table address +3 (if flag = 400_8).
29-0	Reserved.

W.BKPLP (23)

Breakpoint wait loop.

W.BKTBL (24)

Bit	Description
59-36	0.
35-24	Number of breakpoint entries *4.
23-0	Start of breakpoint entries.

Breakpoint entry:

Word	Bit	Description
0	59-54	Processing flag.
	53-36	Breakpoint address.
	35-18	Address of segment table entry for segment or 0.
	17-0	Link to next breakpoint entry for the segment (end of chain = 0).
1		Saved instruction word (from breakpoint address).
2		Entry word; breakpoint word is replaced with a return jump to this word.
3		Return jump to breakpoint wait loop.

AREA TABLE

Bits 17 through 0 of word 67 in the CMR pointer area contain the address of T.AREA.

	59	35	17	0	
T.AREA W.CURSYS	Current System ECS Address				0
W.ALTSYS	Alternate System Address (or Terminator)				1
W.CMRES	ECS Address	Length	CM Address		2
W.MTRA	0	Length	CM Address		3
W.USERA	0	Length	CM Address		4
W.SEGT	ECS Address	Length	CM Address		5
W.INIT	ECS Address	Length	CM Address		6
W.EDTIM	Date-Time Stamp (from ECS System)				7

W.CMRES(2)

CM resident descriptor word.

W.MTRA(3)

Monitor mode overlay area descriptor word.

W.USERA(4)

User mode overlay area descriptor word.

W.SEGT(5)

Segment table descriptor word.

W.INIT(6)

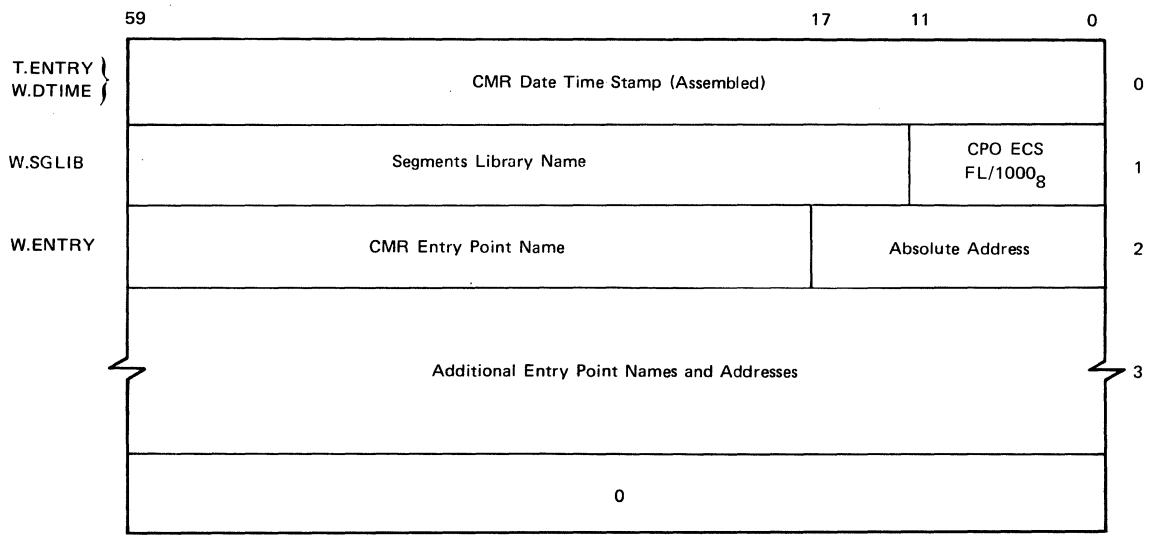
Initialization segment descriptor word.

W.EDTIM(7)

Date-time stamp of CMR for which this ECS system was loaded. Last digit of year, ordinal date (three digits), two-digit hour, two-digit minutes, and two-digit seconds, all in display code.

ENTRY TABLE

Bits 35 through 18 of word 67 in the CMR pointer area contain the address of T.ENTRY.



W.SGLIB(1)

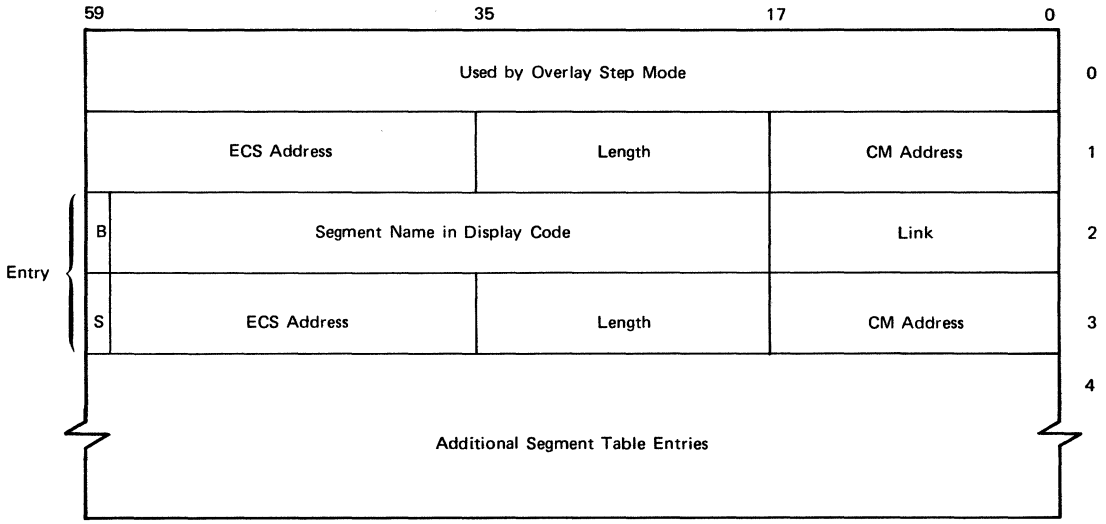
Library containing segments to be used with this CMR. Control point zero FL/1000_8 is used for ECS system code.

W.ENTRY (2)

List of entry points defined in CMR.

SEGMENT TABLE

Word 5 in the area table contains the address of the segment table.



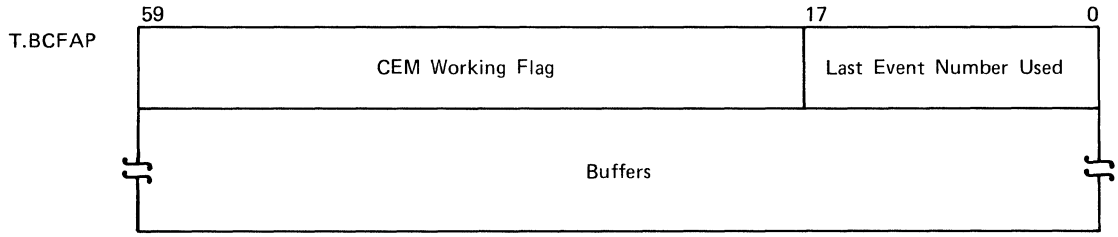
Word	Description
0	Step mode communication word.
1	ECS sytem resident descriptor word.
2 and 3	Segment table entries.

Segment table entry:

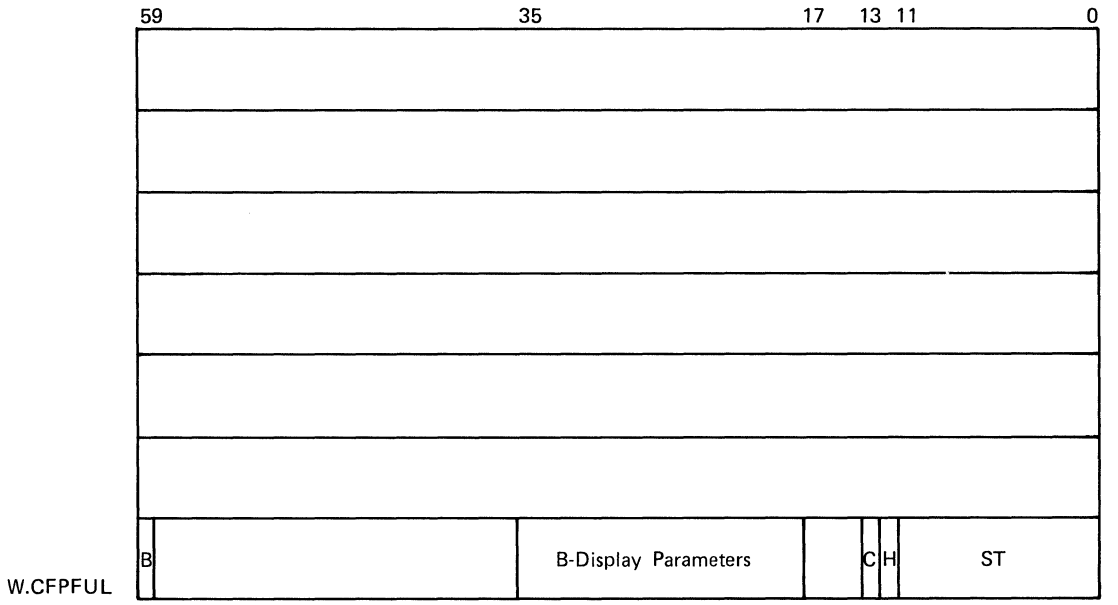
Word	Bit	Description
0	59	Breakpoint flag.
	58-18	Segment name in display code, left-justified, zero-filled.
	17-0	Link to error directory entries or breakpoint entries.
1	59	Step mode flag.
	58-36	ECS address of segment.
	35-18	Segment length (exclusive of ECS error recovery information).
	17-0	CM address where segment should be loaded.

CEFAP BUFFER AREA

T.BCFAP is an entry in the T.ENTRY table.



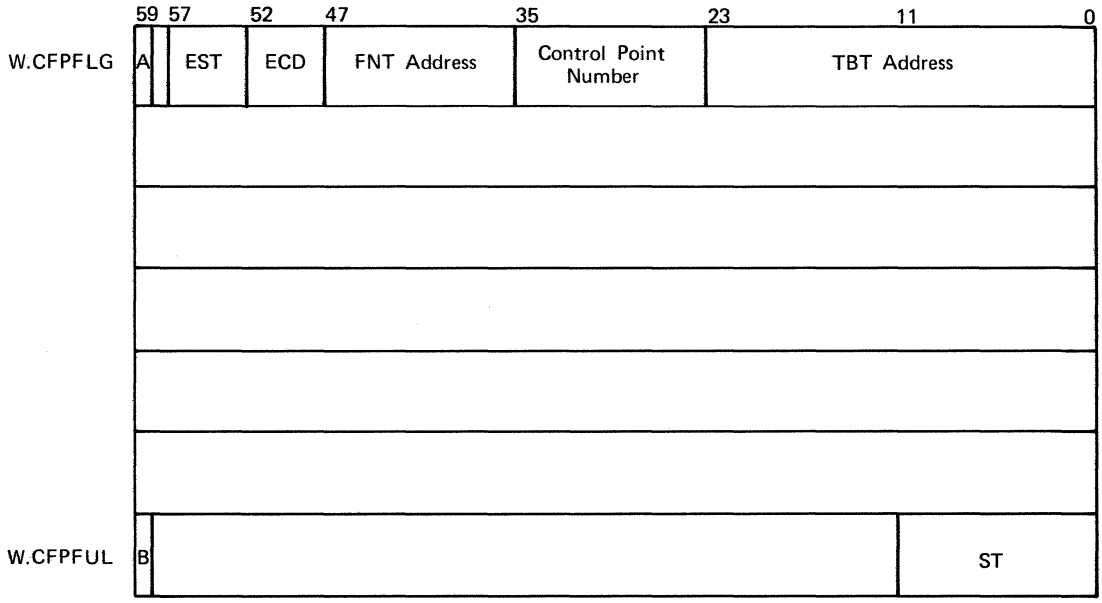
Buffer for 819 Disk Error



Field	Description
B	Busy bit.
C	Set if buffer contains CERFILE message to be recorded.
H	Set if CEM is to call segment HLCN.
ST	Subtype code = 1.

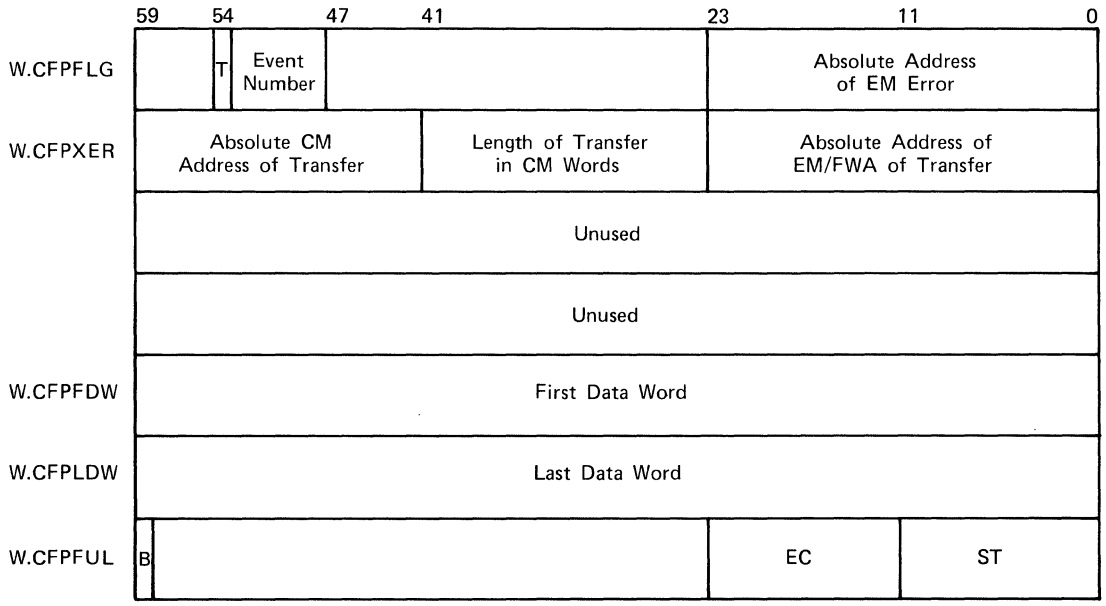
Bits 35 through 18 are parameters for ERRcc, ESTxx, and CHcc messages displayed on the B-display.

Buffer for Dayfile Messages



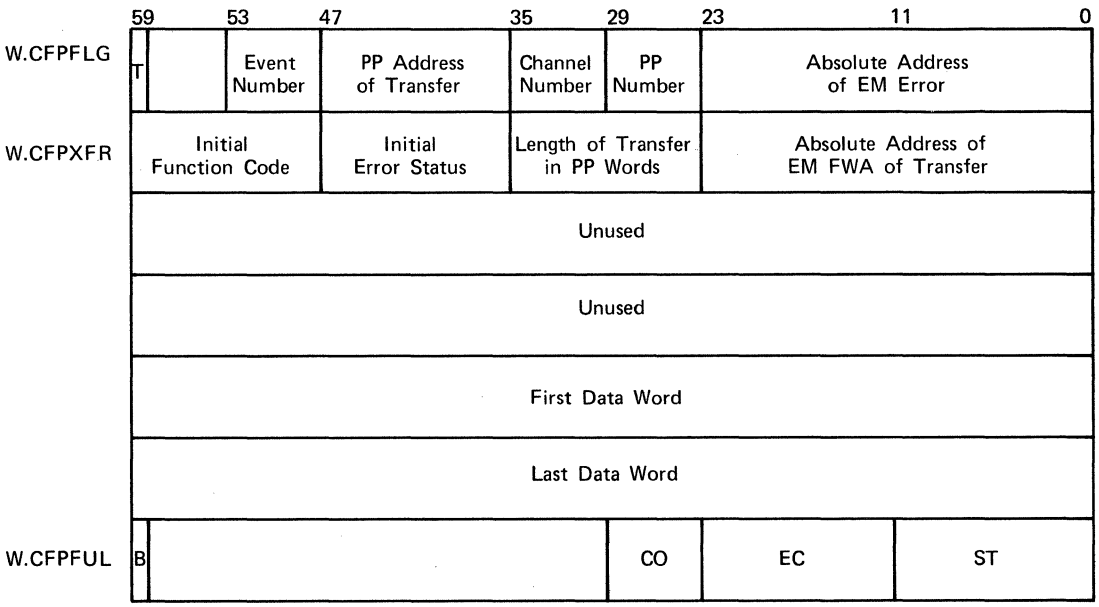
Field	Description
A	Abort flag.
EST	EST ordinal.
ECD	Error codes.
	0 System communication error.
	1 Uncorrectable RMS error.
B	Busy bit.
ST	Subtype code = 2.

CP Read/Write of ECS



Field	Description
T	Type of access.
	0 System.
	1 Subsystem.
B	Busy bit.
EC	Error code.
	0 ECS write abort.
	1 ECS write recovered.
	2 ECS read parity error.
	3 Recovered ECS read parity errror.
ST	Subtype code = 0.

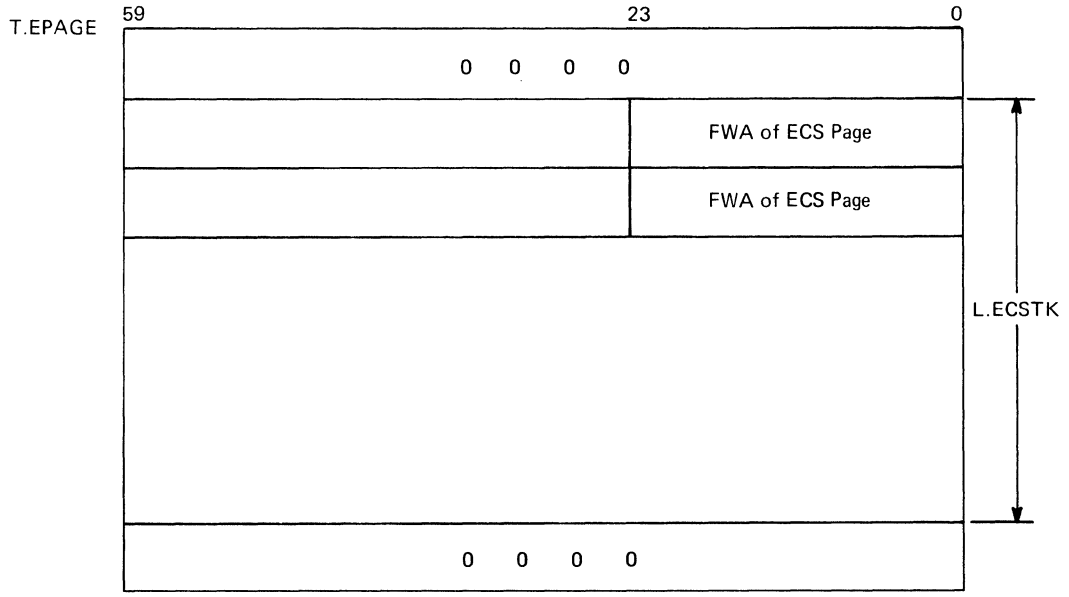
DDP Read of ECS



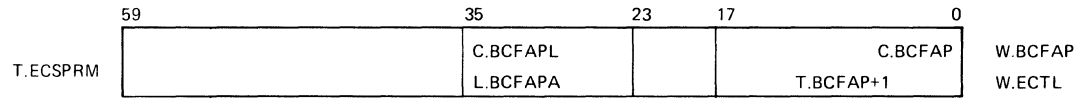
Field	Description
T	Type of access.
	0 System.
	1 Subsystem.
B	Busy bit.
CO	Calling PP overlay. For error codes (EC) 6, 7, 10, and 11, this field contains the display code of the characters P or Q.
	P 1SP
	Q 1SQ
EC	Error code.
	4 DDP read parity error.
	5 Recovered DDP read parity error.
	6 1SP/1SQ DDP read parity error.
	7 1SP/1SQ recovered DDP read parity error.
	10 1SP/1SQ ECS write abort.
	11 1SP/1SQ ECS write recovered.
	12 STL DDP read channel parity error.
	13 1SP/1SQ DDP channel error.

EMPTY PAGE STACK

Bytes 0 and 1 of word 57 in the CMR pointer area contain the address of T.EPAGE.



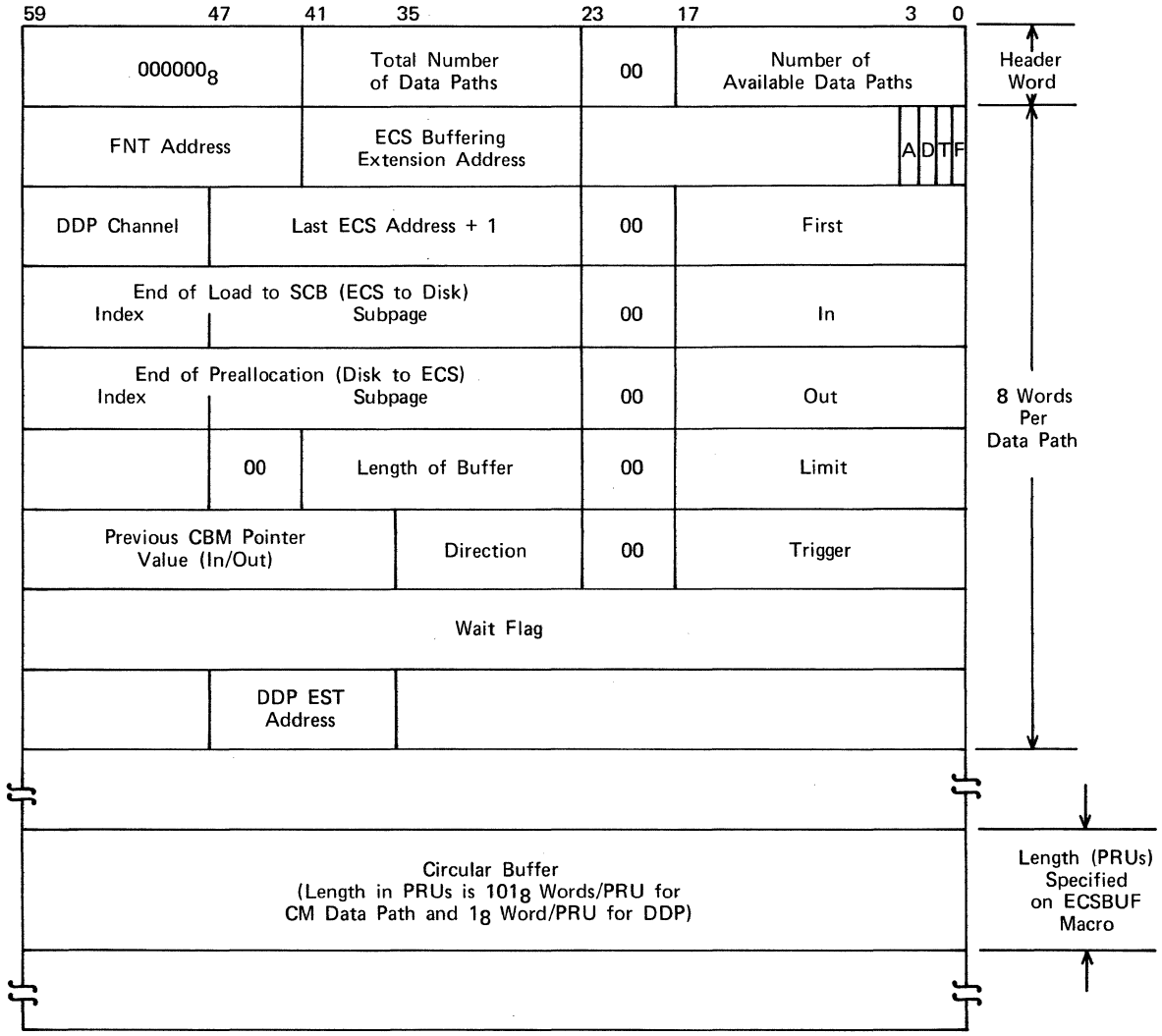
T.EPAGE + (W.B5) Points to Next Entry



SYSTEM CIRCULAR BUFFER (SCB)

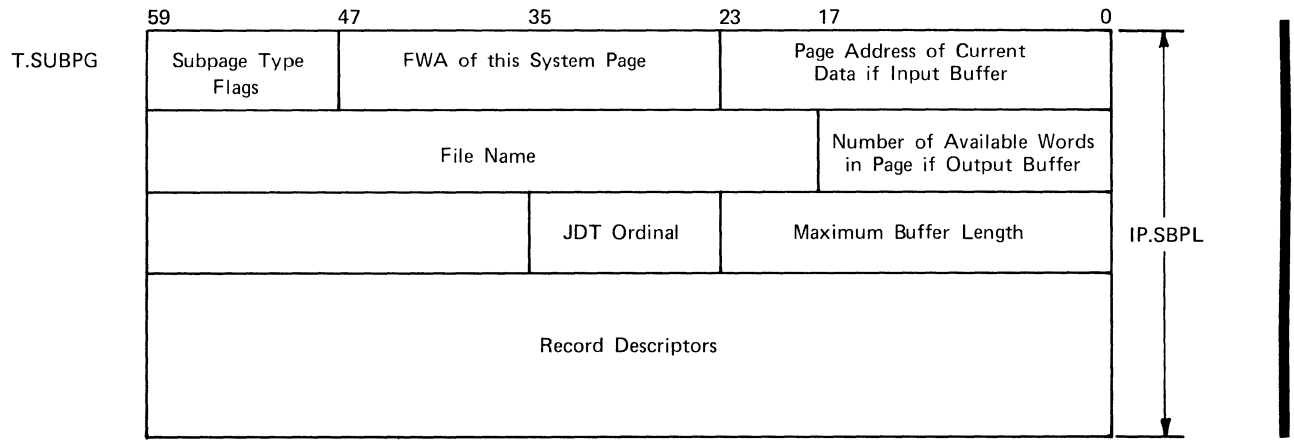
SCBHDR is an entry of table T.ENTRY.

SCBHDR



Field Name	Description
A	Direct access ECS I/O.
D	Direction.
	0 Disk to ECS.
	1 ECS to disk.
T	Type.
	0 CM data path
	1 DDP.
F	Free/busy.
	0 Busy.
	1 Free.

SUBPAGE BUFFER



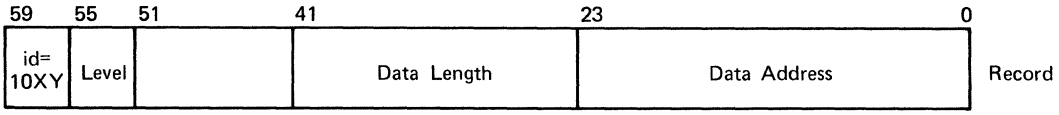
Word	Bit	Description
T.SUBPG	59-50	Subpage data type.

Bit	Description
59	Reserved.
58	Reserved.
57	Index for random ECS file.
56	Auxiliary file for ECS resident random file.
55	Swap file.
54	ECS resident file.
53	Library file (ZZZZZ06).
52	I/O buffer (with bit 50).
51	Reserved.
50	Release data as read.

49-48 Subpage position

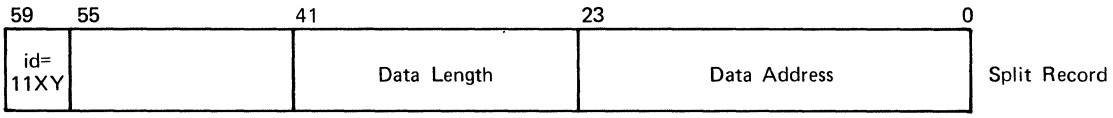
00	Continuation subpage.
01	First subpage in a file.
10	Last subpage.

RECORD DESCRIPTORS



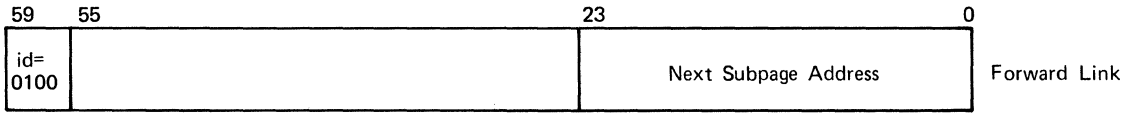
X=1 Beginning of a new data page.

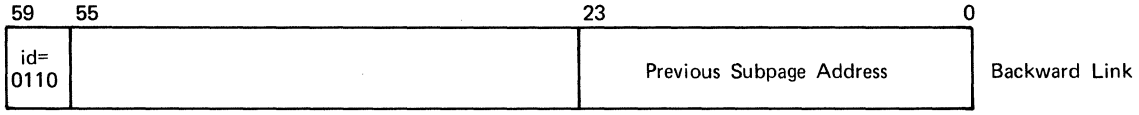
Y=1 Transmission RMS parity error flag.

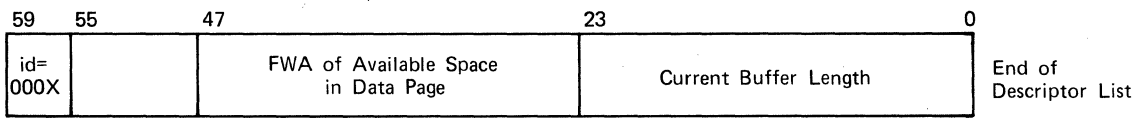


X=1 Beginning of a new data page.

Y=1 Transmission RMS parity error flag.

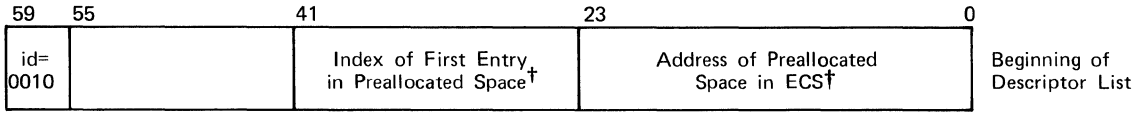






X=0 End of ECS buffer.

X=1 End of information on disk.



id (Bits 59-56)

0xxx System descriptor.

- 0000 End of list; continued on disk.
- 0001 End of list; EOI on file.
- 0010 Beginning of list.
- 0100 Forward link pointer.
- 0110 Backward link pointer.

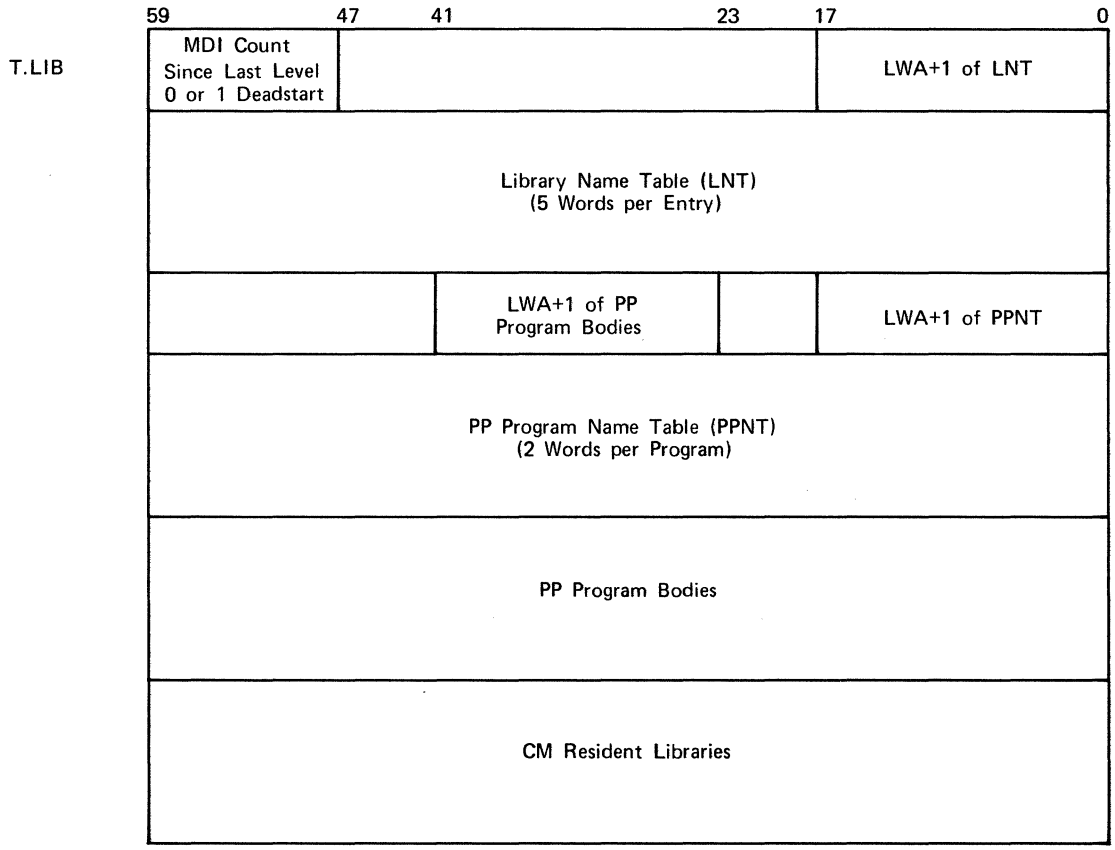
1xxx Data descriptor.

- 10xx Full record descriptor.
- 11xx Split record descriptor (full record described by this and next descriptor).
- 1x0x Current data page.
- 1x1x New data page.
- 1xx0 No parity error.
- 1xx1 Parity error in record.

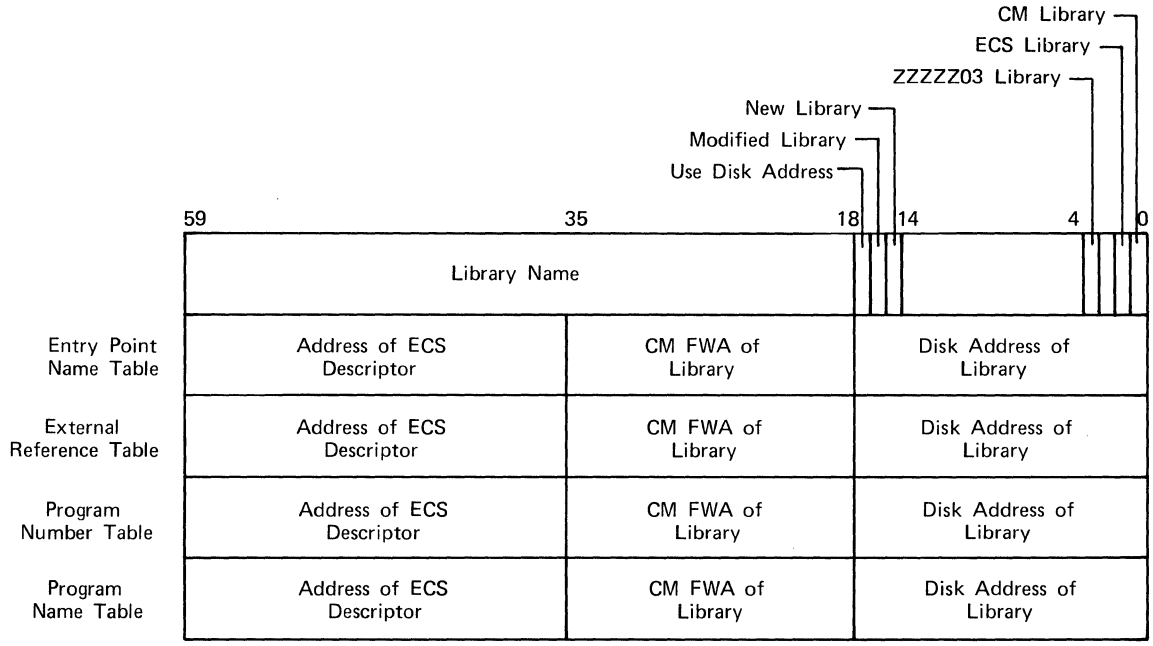
† Used only for the library.

CMR LIBRARY DIRECTORY

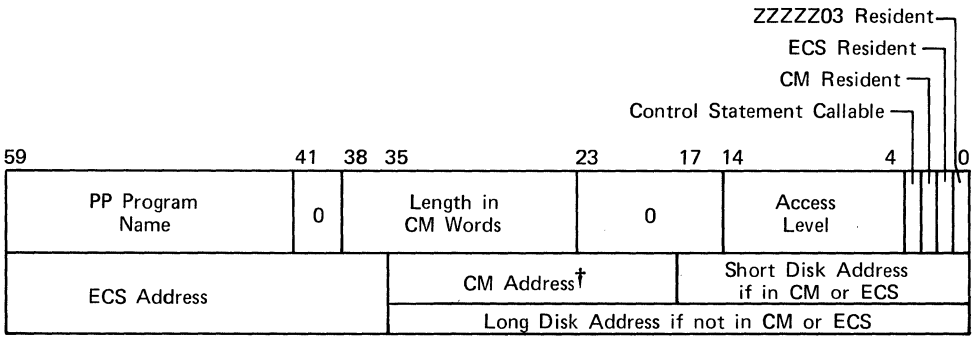
Bits 59 through 38 of word 1 in the CMR pointer area contain the address of T.LIB.



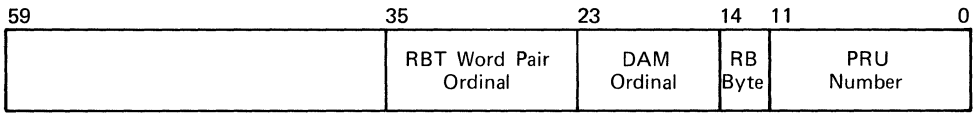
LIBRARY NAME TABLE (LNT) ENTRY



PP PROGRAM NAME TABLE (PPNT) ENTRY

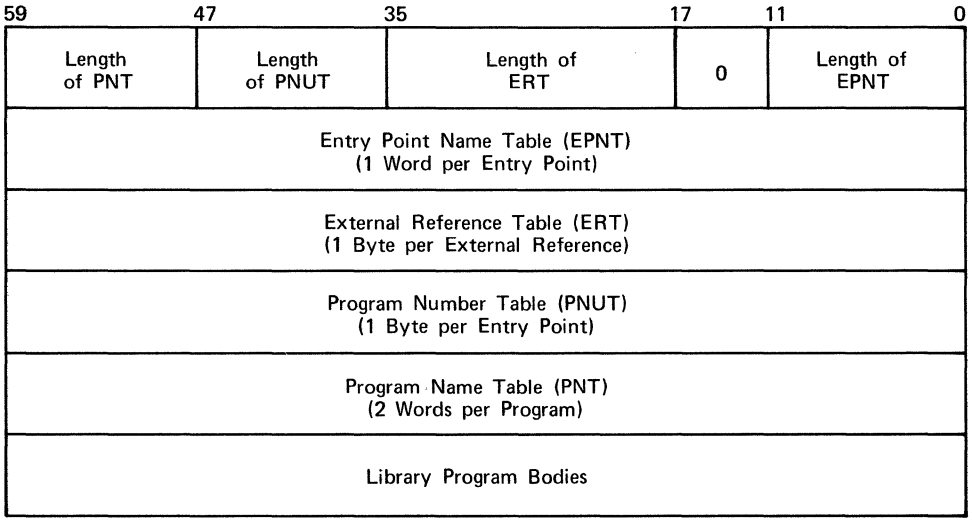


The long disk address has the following format.

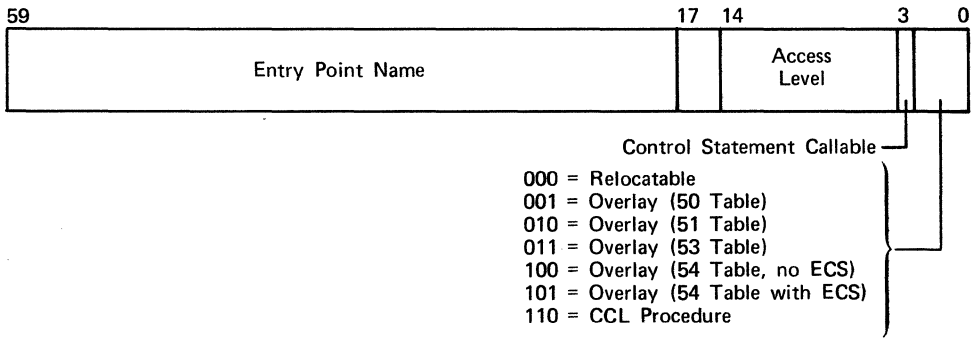


† For ECS resident overlay, field holds last consecutive access failure count. If failure count overflows a set threshold, residence changes to RMS.

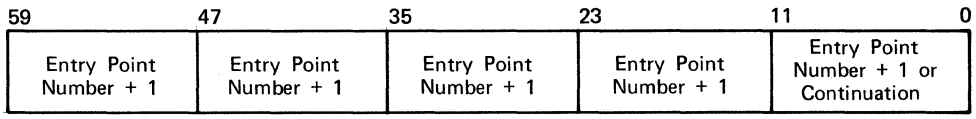
CM RESIDENT LIBRARY FORMAT



EPNT ENTRY FORMAT



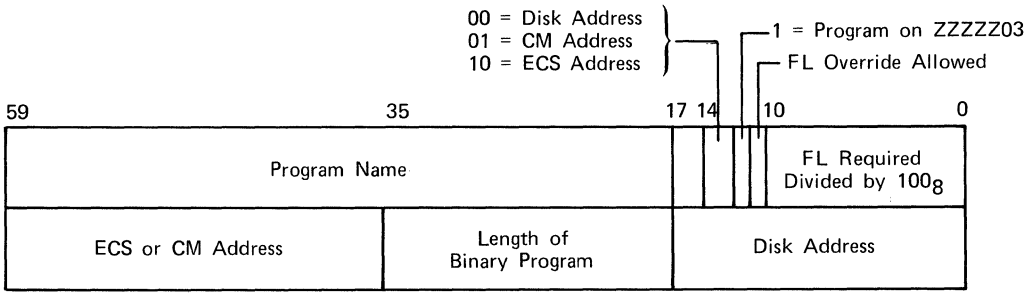
ERT ENTRY FORMAT



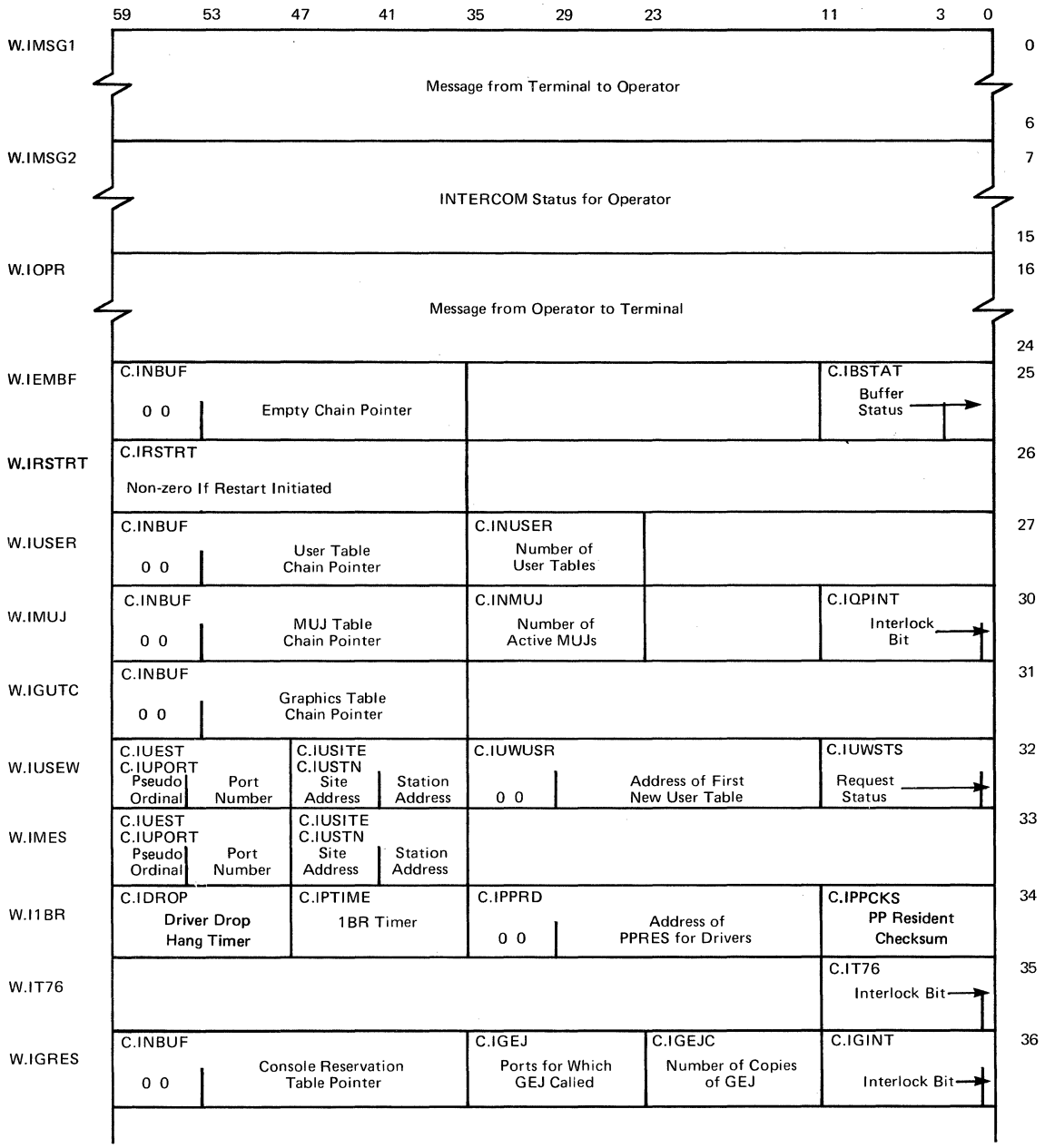
PNUT ENTRY FORMAT

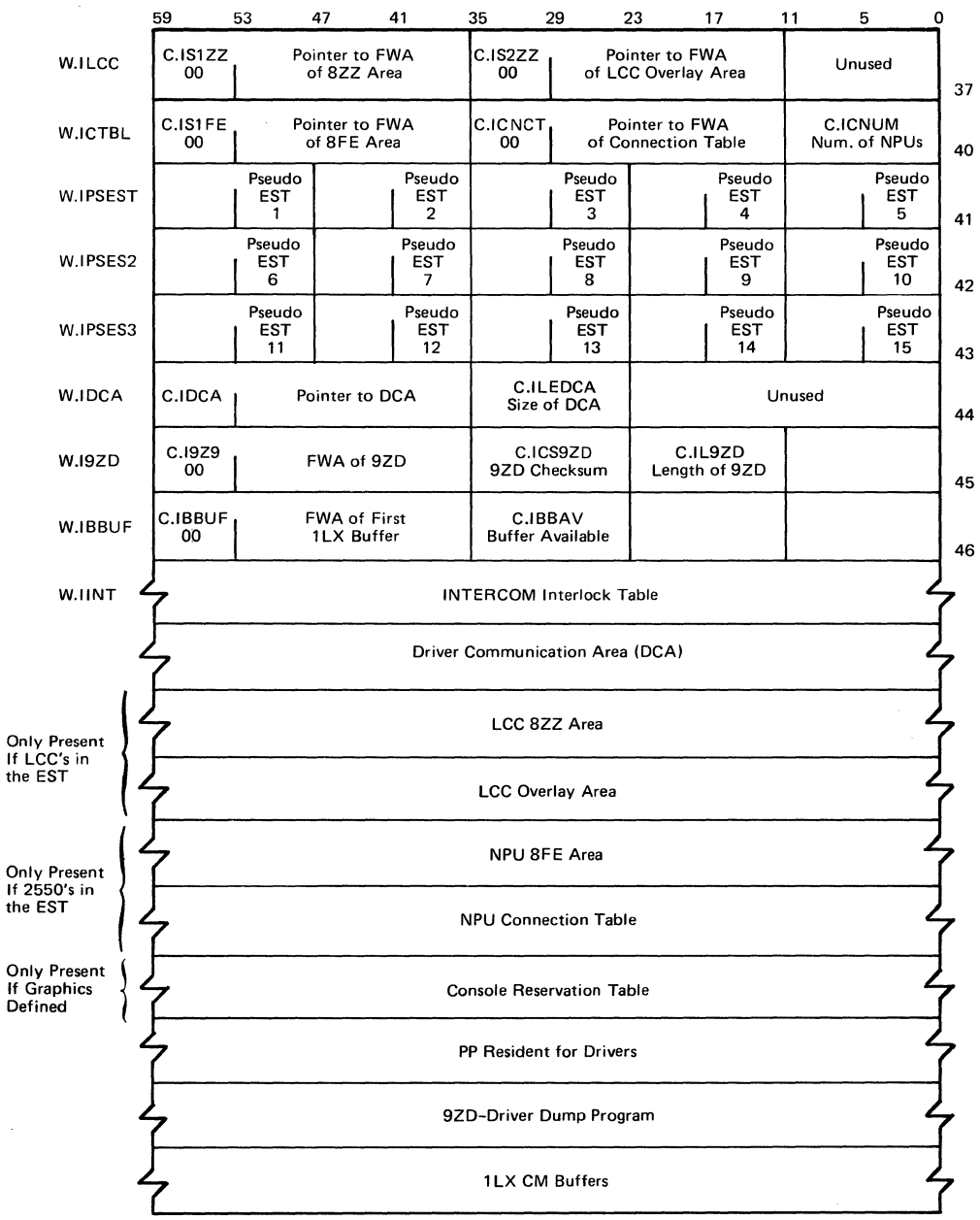
59	47	35	23	11	0
Parcel 0 Relative PNT Address	Parcel 1 Relative PNT Address	Parcel 2	Parcel 3	Parcel 4	
Parcel 5	Parcel 6			Parcel n	

PNT ENTRY FORMAT

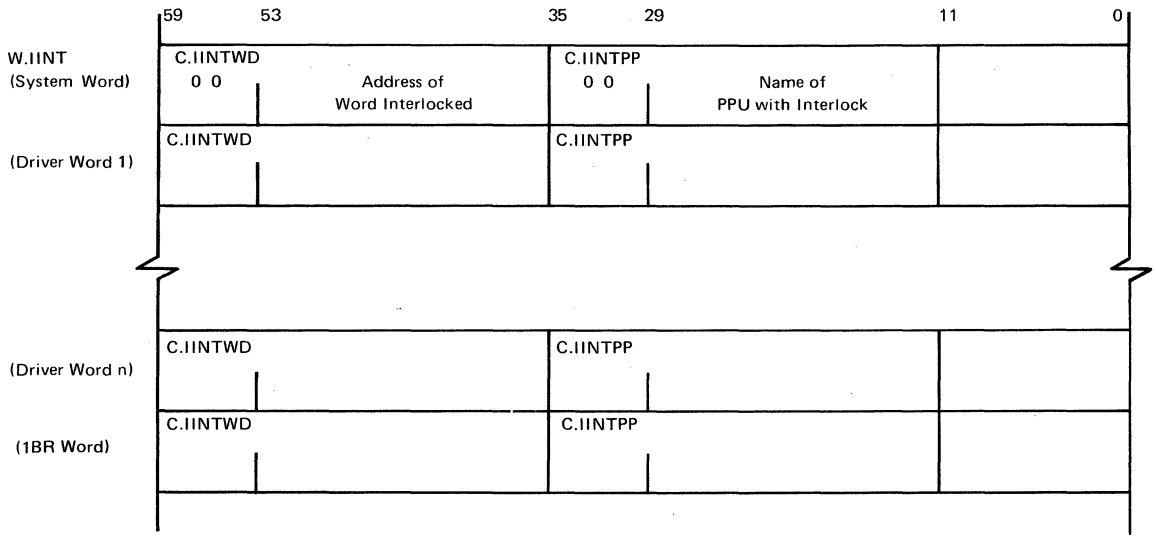


INTERCOM 4 POINTER AND BUFFER AREA

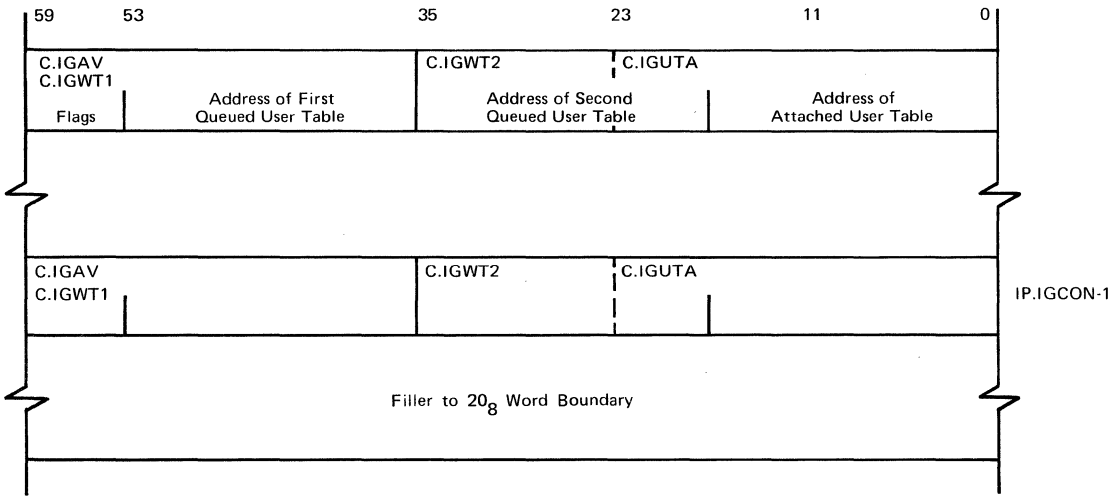



59
53
47
41
35
29
23
17
11
5
0
W.ILCC
C.IS1ZZ 00
Pointer to FWA of 8ZZ Area
C.IS2ZZ 00
Pointer to FWA of LCC Overlay Area
Unused
37
W.ICTBL
C.IS1FE 00
Pointer to FWA of 8FE Area
C.ICNCT 00
Pointer to FWA of Connection Table
C.ICNUM Num. of NPUs
40
W.IPSEST
Pseudo EST 1
Pseudo EST 2
Pseudo EST 3
Pseudo EST 4
Pseudo EST 5
41
W.IPSES2
Pseudo EST 6
Pseudo EST 7
Pseudo EST 8
Pseudo EST 9
Pseudo EST 10
42
W.IPSES3
Pseudo EST 11
Pseudo EST 12
Pseudo EST 13
Pseudo EST 14
Pseudo EST 15
43
W.IDCA
C.IDCA
Pointer to DCA
C.ILEDCA Size of DCA
Unused
44
W.I9ZD
C.I9Z9 00
FWA of 9ZD
C.ICS9ZD 9ZD Checksum
C.IL9ZD Length of 9ZD
45
W.IBBUF
C.IBBUF 00
FWA of First 1LX Buffer
C.IBBAV Buffer Available
46
W.IINT
INTERCOM Interlock Table
Driver Communication Area (DCA)
LCC 8ZZ Area
LCC Overlay Area
Only Present If LCC's in the EST
NPU 8FE Area
NPU Connection Table
Only Present If 2550's in the EST
Console Reservation Table
Only Present If Graphics Defined
PP Resident for Drivers
9ZD-Driver Dump Program
1LX CM Buffers

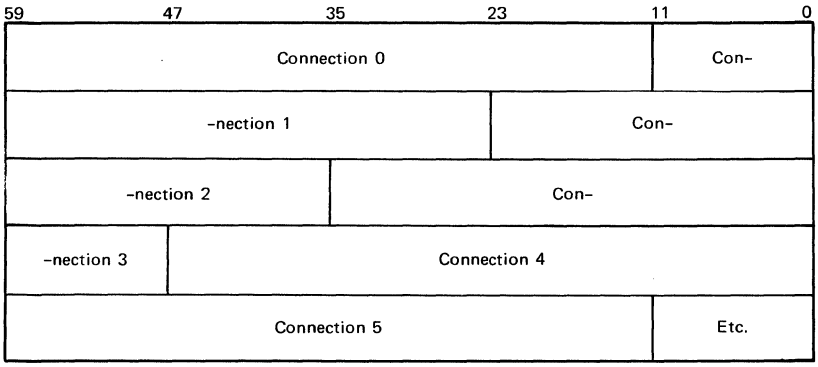
INTERCOM 4 INTERLOCK TABLE



CONSOLE RESERVATION TABLE

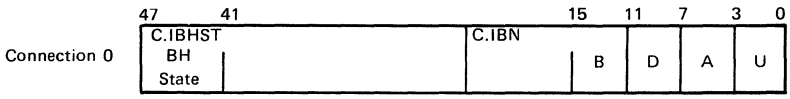


INTERCOM 4 CONNECTION TABLE



Each connection table entry contains four bytes (48 bits).
It is used by the front end.

CONNECTION TABLE ENTRY FORMATS



C.IBHST

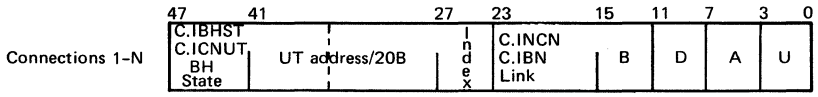
Bit	Field	Description
11-6	S.IBHST	Block handler state.
	L.IBHST	Block handler state length.

C.IBN

Bit	Field	Description
23-0	S.ILBOBN	Block serial number (BSN) of last BACKed downline service message (SM) block.

C.IBN+1

Bit	Field	Description
11-8	S.ILSOBN	BSN of last downline SM block sent.
7-4	S.ILBIBN	BSN of last BACKed upline SM block.
3-0	S.ILRIBN	BSN of last upline SM block received.
	L.IBN	BSN field length.



C.IBHST

Bit	Field	Description
11-6	S.IBHST	Block handler state.
	L.IBHST	Block handler state length.

C.ICNUT

Bit	Description
5-0	Uppper 6 bits of user table (UT) address/20_8.

C.ICNUT+1

Bit	Description
11-4	Lower 8 bits of UT address/20_8.
3-0	Stream index.

C.INCN

Bit	Field	Description
11-4	S.INCN	Ordinal of next linked connection table entry.

C.IBN

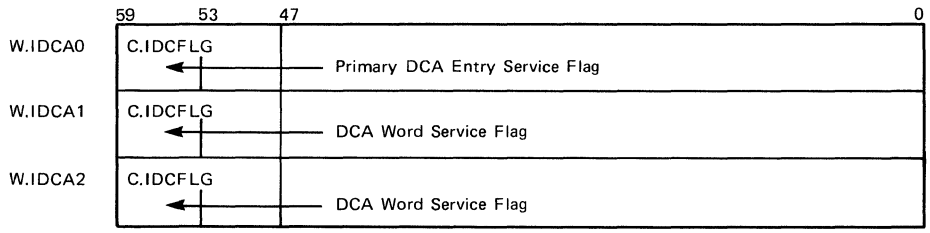
Bit	Field	Description
3-0	S.ILBOBN	Block serial number (BSN) of last BACKed downline block.

C.IBN+1

Bit	Field	Description
11-8	S.ILSOBN	BSN of last downline block sent.
7-4	S.ILBIBN	BSN of last BACKed upline block.
3-0	S.ILRIBN	BSN of last upline block received.
	L.IBN	BSN field length.

INTERCOM 4 DRIVER COMMUNICATION AREA (DCA)

STANDARD DCA ENTRY



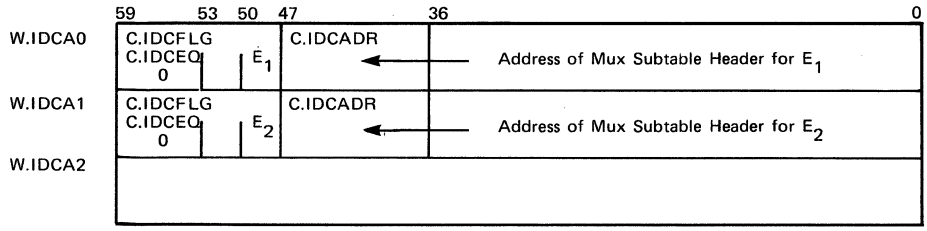
W.IDCA0

Byte	Bit	Field	Description
C.IDCFLG	11-6	S.IDCFLG	0 indicates no service required from standard INTERCOM service routines for any word in this DCA entry. 1 - 77_8 indicates service required for entire DCA entry.

W.IDCA1 and W.IDCA2

Byte	Bit	Field	Description
C.IDCFLG	11-6	S.IDCFLG	0 indicates no service required from standard INTERCOM service routines for this word. 1 - 77_8 indicates service required for this word. The primary DCA entry flag (in W.IDCA0) must be nonzero for service to occur.

2550 FRONT END NPU DRIVER DCA ENTRY



W.IDCA0

Byte	Bit	Field	Description
C.IDCFLG	11-6	S.IDCFLG	0 indicates no service required from standard INTERCOM service routines.
		L.IDCFLG	Service flag field length.
	5-3		Reserved.
C.IDCEQ	2-0	S.IDCEQ	Equipment number of first NPU on channel (E_1).
		L.IDCEQ	Equipment number field length.
C.IDCADR	11-0		Address of multiplexer subtable of first NPU on channel.

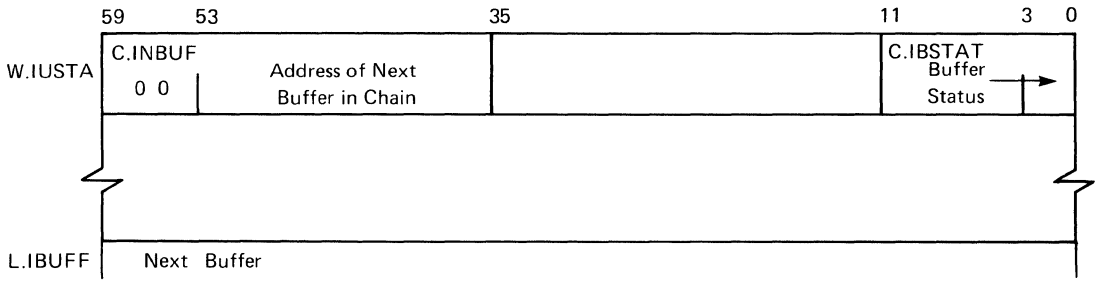
W.IDCA1

Byte	Bit	Field	Description
C.IDCFLG	11-6	S.IDCFLG	0 indicates no service required from standard INTERCOM service routines.
	5-3		Reserved.
C.IDCEQ	2-0	S.IDCEQ	Equipment number of second NPU on channel (E_2).
C.IDCADR	11-0		Address of multiplexer subtable of second NPU on channel.

W.IDCA2 Reserved.

INTERCOM 4 BUFFER AREA

BUFFERS (20_8 Word Boundary)



Byte	Bit	Field	Description
C.IBSTAT	3-0	S.ITYPE	Buffer status (in octal).

0 Empty.

1 User table.

2 Data output/job statement.

3 Data input.

4 Interactive control statements.

5 Multiuser job table (MUJ).

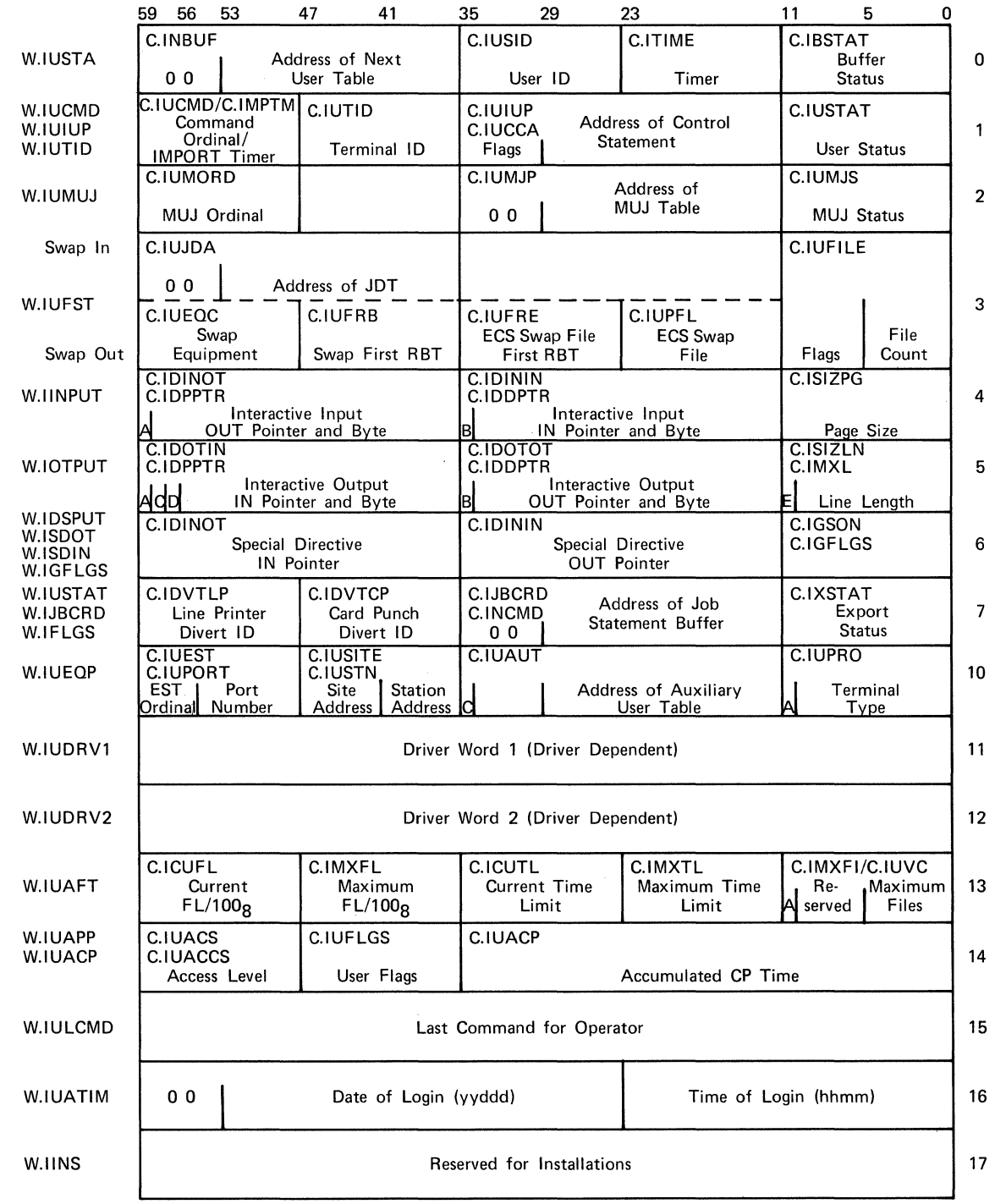
6 Low speed auxiliary user table (AUT).

7 Graphics jobs.

10 Wideband auxiliary user table (AUT).

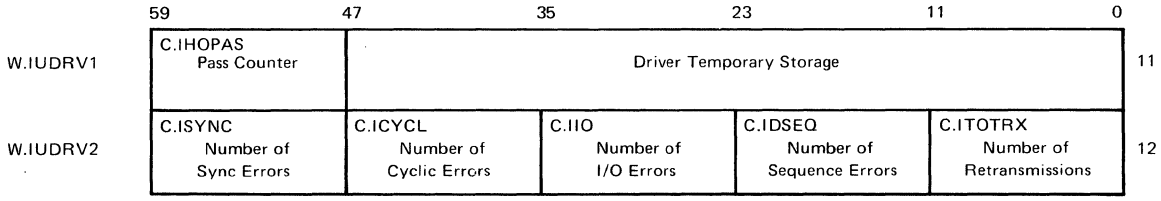
12 Limbo buffer.

INTERCOM 4 USER TABLE

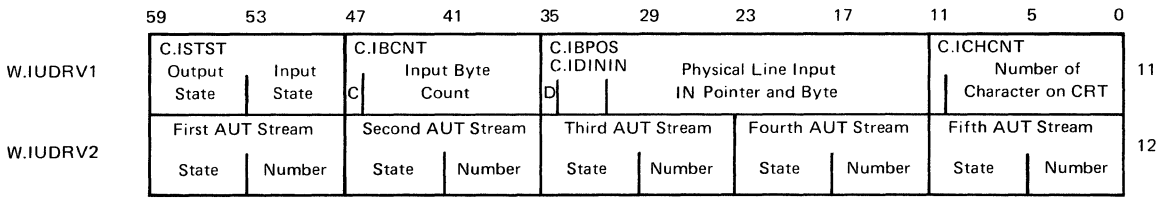


INTERCOM 4 USER TABLE DRIVER WORDS

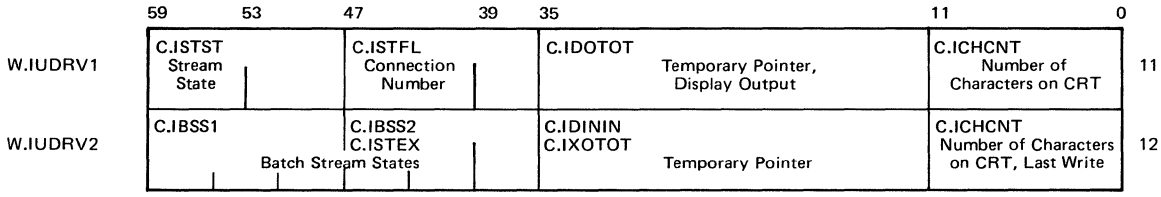
6673/6674 DRIVER



LCC DRIVER



2550 DRIVER



W.IUSTA(0)

Byte	Bit	Field	Description
C.IBSTAT(4)	11	S.IABRT	Abort request.
	10	S.IUTAPE	Request paper tape reading.
	9	S.IRDIS	Request disconnect.

Byte	Bit	Field		Description
	8-7	S.ISTATE	00	Transmission state.
			01	Waiting input.
			10	Waiting output.
			11	Active or assigned to control point.
	6-5	S.ILOGO	00	User logged out.
			01	User logged in.
			10	Automatic logout requested.
			11	Automatic logout in progress.
	4	S.IDISC	Terminal disconnected.	
	3-0	S.ITYPE	Buffer type.	

W.IUCMD(1)

Byte	Bit	Field	Description
C.IUCCA(2)=C.IUIUP(2)	11	S.IUEDC	Buffer contains control statements sent from EDITOR.
	10		Unused.
	9	S.IUEXS	Start execution flag.
	8	S.IUECP	Editor control statements processed.
	7	S.INCT	Command in 1CI table.
	6	S.IUCCP	Control statements moved to control point area.
C.IUSTAT(4)	11	S.ICTAPE	Paper tape on flag.
	10	S.INOCOM	Do not issue command.
	9	S.IMPORT	IMPORT on at 200 UT.
	8	S.ICBTHW	Allow commands before LOGIN.
	7	S.ICACT	1CI active.
	6	S.ICMES	MES active.
	5-0	S.IUHDIS	Used by 1DS to determine type of H display wanted.

W.IUMUJ(2)

Byte	Bit	Field	Description
C.IUMJS(4)	11	S.IMUJ	Attached to MUJ.
	10	S.IMWI	Waiting for input.
	9	S.IMWO	Waiting for output to complete.
	8	S.IMLGS	Logout sent.
	7	S.IMBKS	Break sent.
	6-3		Unused.
	2	S.IMDIS	Reconnected after disconnect.
	1	S.IMRUN	RUN command in progress.
	0	S.IMNUS	New user.

W.IUFST(3)

Byte	Bit	Field	Description
C.IUFILE(4)	11	S.IURED	REDUCE flag.
	10	S.IUFNT	FNT to be associated on next swap.
	9	S.IUECS	Swap file on ECS.
	8	S.IUPS	Pause bit.
	7	S.IUDMP	SAVEFL flag.
	6	S.IUEOE	End of execution.
	5-0	S.IUCNT	Count of FNT entries in swap file.

W.IINPUT(4)

Byte	Bit	Field	Description
C.IDPPTR(0)	11	S.IINTLK	Interlock bit for 3TT and I/O macros.
	8-6	S.IBTYE	Byte position in current word.
C.IDDPTR(2)	10	S.IRLS	Input line not in progress.
	9	S.IDRACT	Driver not finished.
	8-6	S.IBYTE	Byte position in current word.

W.IOTPUT(5)

Byte	Bit	Field	Description
C.IDPPTR(0)	11	S.IINTLK	Interlock bit for 3TT and I/O macros.
	10	S.IPWOFF	Page-wait-off flag for processor.
	9	S.I1IML	1IM interlock.
	8-6	S.IBYTE	Byte position in current word.
C.IDDPTR(2)	11	S.IPRNOP	Driver not finished or previously inoperative.
	8-6	S.IBYTE	Byte position in current word.
C.ISIZLN(4)	11	S.ILNCHG	Line size change bit.

W.IGFLGS(6) 274 Graphics only.

Byte	Bit	Field	Description
C.IGFLGS(4)	11-9	S.IGCON	Console for which job is queued.
	4	S.IGSNIT	SIGNON initialization.
	3	S.IGDRP	Job dropping.
	2	S.IGSNON	SIGNON job.
	1	S.IGQUE	Job queued for console.
	0	S.IG3TT	3TT flag for interrupted output.

W.IGFLGS(6) High Speed INTERCOM terminal only.

Byte	Bit	Field	Description
C.IGSON(4)	8	S.IXOFF	Terminal off line.
	7	S.IGSON	274 Graphics streams defined.
	6	S.IGRMUX	Graphics multiplexer.
	5-0	S.IGDSN	Graphics data stream number.

W.IUSTAT=W.IFLGS(7)

Byte	Bit	Field	Description
C.IXSTAT(4)	11	S.IXACK	Processor drop acknowledge (1XP).
	10	S.IXDRP	Drop requested (1XP).
	9	S.IXUTBY S.ILUTBY	User table release requested (1XP/1LX).
	8	S.IRLAUT	AUT drop requested of 1LX by 1CI.
	7	S.IOVFL	Message overflow (1XP).
	6	S.IMAX	Maximum message buffers assigned (1XP).
	5	S.IDVTLP	Divert print files.
	3	S.IDVTCP	Divert punch files.

W.IUEQP(10)

Byte	Bit	Field	Description
C.IUEST(0)	11-8	S.IUEST	Psuedo EST ordinal.
	7-0	S.IUPORT	Port number.
C.IUSITE(1)=C.IUSTN(1)	11-6	S.IUSITE	Site address.
	5-0	S.IUSTN	Station address.
C.IUAUT(2)	11	S.IUAUTR	AUT request bit.
C.IUPRO(4)	11	S.IUNOP	Inoperative bit.

W.IUAFT(13)

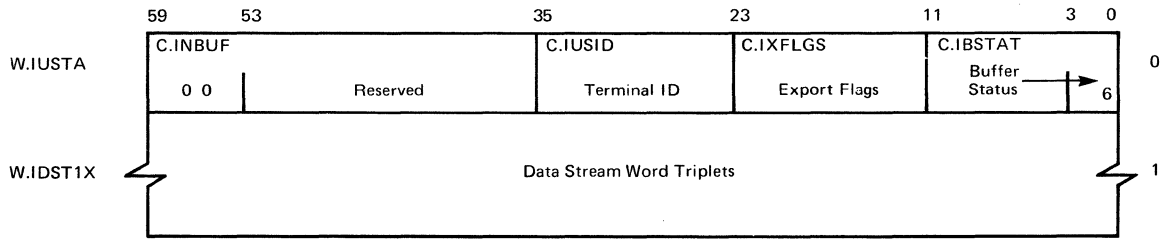
Byte	Bit	Field	Description
C.IUVC(4)=C.IMXFI	11	S.IUVC	VC carriage control.
	10-6		Reserved.
	5-0		Maximum number of files.

W.IUAPP(14)

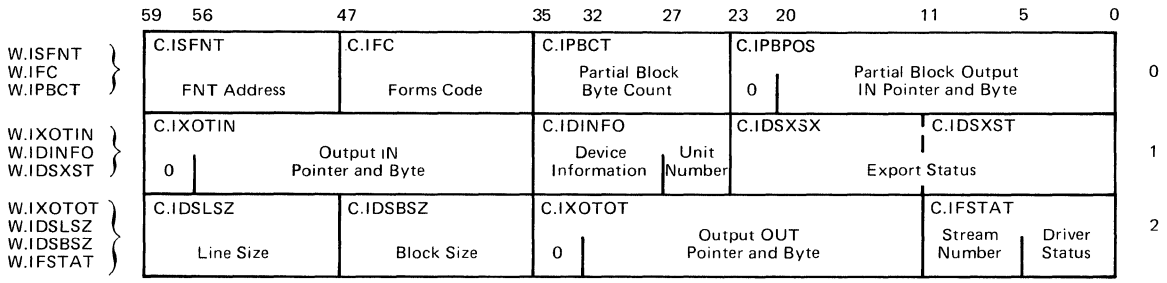
Byte	Bit	Field	Description
C.IUFLGS(4)	11	S.IUUNR	Log in with unrestricted password.
	10	S.ITLOK	Message lock bit.
	9	S.IRLOK	Request job to be locked out.
	8	S.IUAT	Nonterminal interrupt drop.
	7	S.INOLK	Job is locked out.
	6	S.IUREU	Reduce mode flag.
	5-0	S.IUSSW	Sense switch setting changes.

AUXILIARY USER TABLE

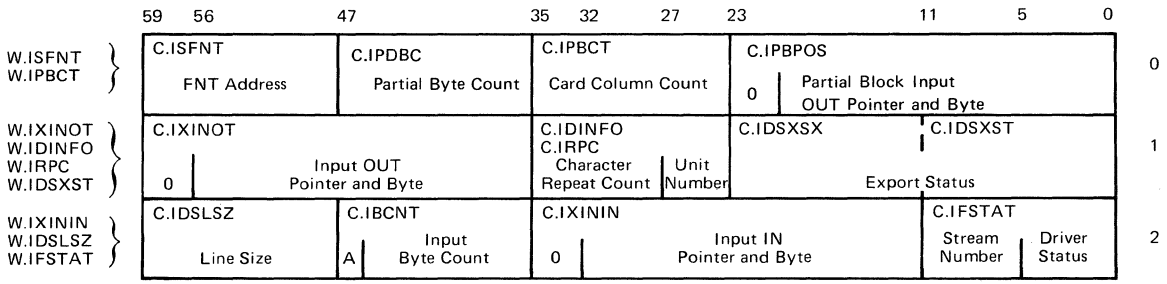
The auxiliary user table is for batch terminals on 7077/791.



OUTPUT DATA STREAM WORD TRIPLET



INPUT DATA STREAM WORD TRIPLET



W.IUSTA(0)

Byte	Field	Description
C.IXFLGS(3)	S.ITXOUT	1LX taking 200 UT out of transmission mode.

W.IDINFO(1) For output streams.

Byte	Bit	Field	Description
C.IDINFO	11-7	S.IFCMO	FCM ordinal.
	6-4	S.ITRAIN	Train type.
			3 B4.
			4 B6.
			5 A6.
			6 A9.

Byte	Bit	Field	Description
	3-0	S.IDSLUN	Logical unit number. 1 and 2 CR1 and CR2. 3 CP. 4 LP1 through 4.

W.IDSXST(1) For output stream.

Byte	Bit	Field	Description
C.IDSXSX(3)	11-9	S.IEC	External code. 1 SB. 2 80 column. 3 B4. 4 B6 or O26. 5 A6 or O29. 6 A9 or ASCII.
	8	S.ISNT	No banner/lace card.
	7-6	S.IIC	Internal code. 0 Display. 1 ASCII. 2 Binary.
	5	S.ILP	Print file (else punch).
	4	S.IPMSG	PM message waiting to be sent.
	3	S.IPMSNT	PM message sent.
	2	S.IZ66	66-bit end-of-line.
	1-0	S.IVCC	V carriage control. 00 No V detected. 01 V detected; check for VC. 10 V detected; no C. 11 VC detected.

W.IDSXST(1) For input file.

Byte	Bit	Field	Description
C.IDSXSX(3)	1	S.INEEDB	Start next PRU with a blank.
	0	S.IREADF	READ, filename command in progress.

W.IDSXST(1) For output file.

Byte	Bit	Field	Description
C.IDSXST(4)	11	S.IWAITX	Wait-for-driver-stop.
	10	S.IWEOJ	Wait-end-of-job.
	9	S.IHDR	Send header.
	8	S.IBAN	No banner.
	7	S.ISUP	Suppress carriage control.
	6	S.IXNDLP	First print end command flag.
	5	S.IEOL	End-of-line.
	4	S.IOFF	Request driver OFF/ON stream.
	3	S.IWPFC	Wait, PFC full.
	2	S.IETX	ETX sent.
	1	S.ISTOP	Request driver stop stream.
	0	S.IXABT	Request driver abort stream.

W.IDSXST(1) For input stream.

Byte	Bit	Field	Description
C.IDSXST(4)	11	S.IWAITX	Wait-for-driver-stop.
	10	S.IWEOJ	Wait-end-of-job.
	9	S.IBIN	Binary mode.
	8	S.IASC	ASCII mode.
	7	S.IJCER	Error.
	5	S.IXEOF	End-of-file.
	4	S.IOFF	Request driver off stream.
	3	S.IWPFC	Wait PFC full.
	1	S.ISTOP	Request driver stop stream.
	0	S.IXABT	Request driver abort stream.

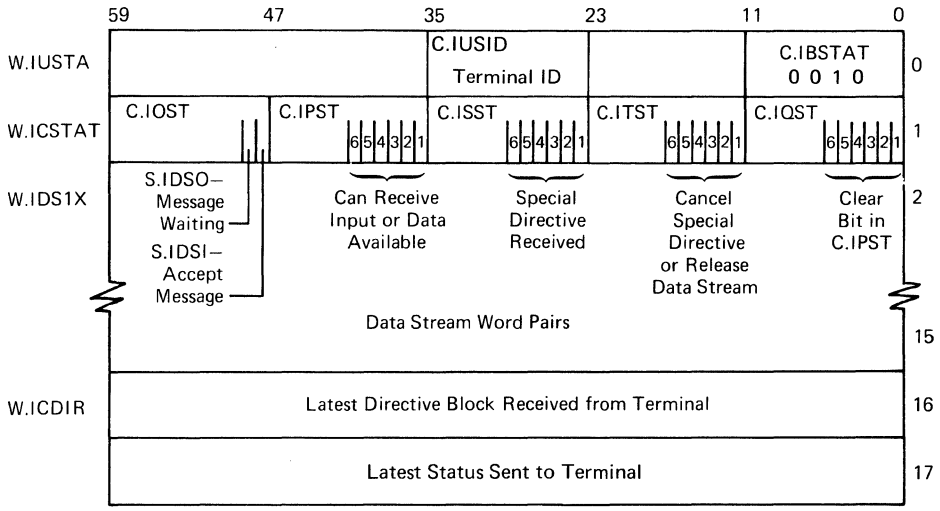
W.IXININ(2) For input stream only.

Byte	Bit	Field		Description
C.IBCNT(1)	11	S.IETB	1	ETB.
			0	ETX sent.

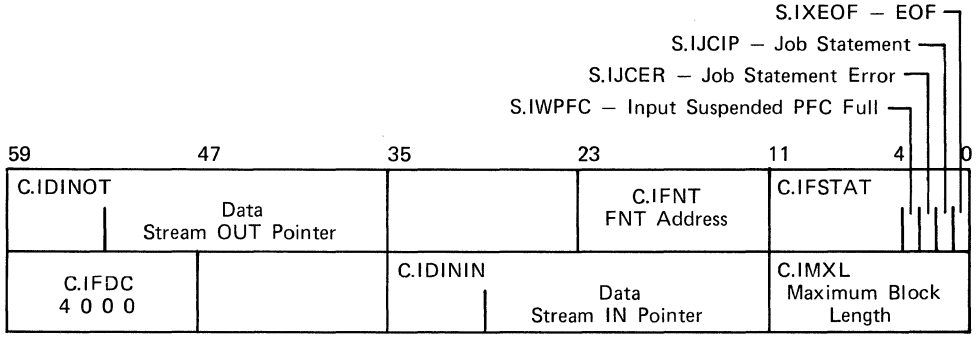
W.IFSTAT(2) For both input and output.

Byte	Bit	Field	Description
C.IFSTAT(4)	9-6	S.IDSN	Data stream number/site address.
	5	S.IXABTI	Abort issued.
	4	S.IOFF	Stream off.
	3	S.INREDY	Device not ready (mode 4).
	2	S.IETX	ETX block sent.
	1	S.ISTOP	Stream stopped.
	0	S.IXABT	Stream aborted.

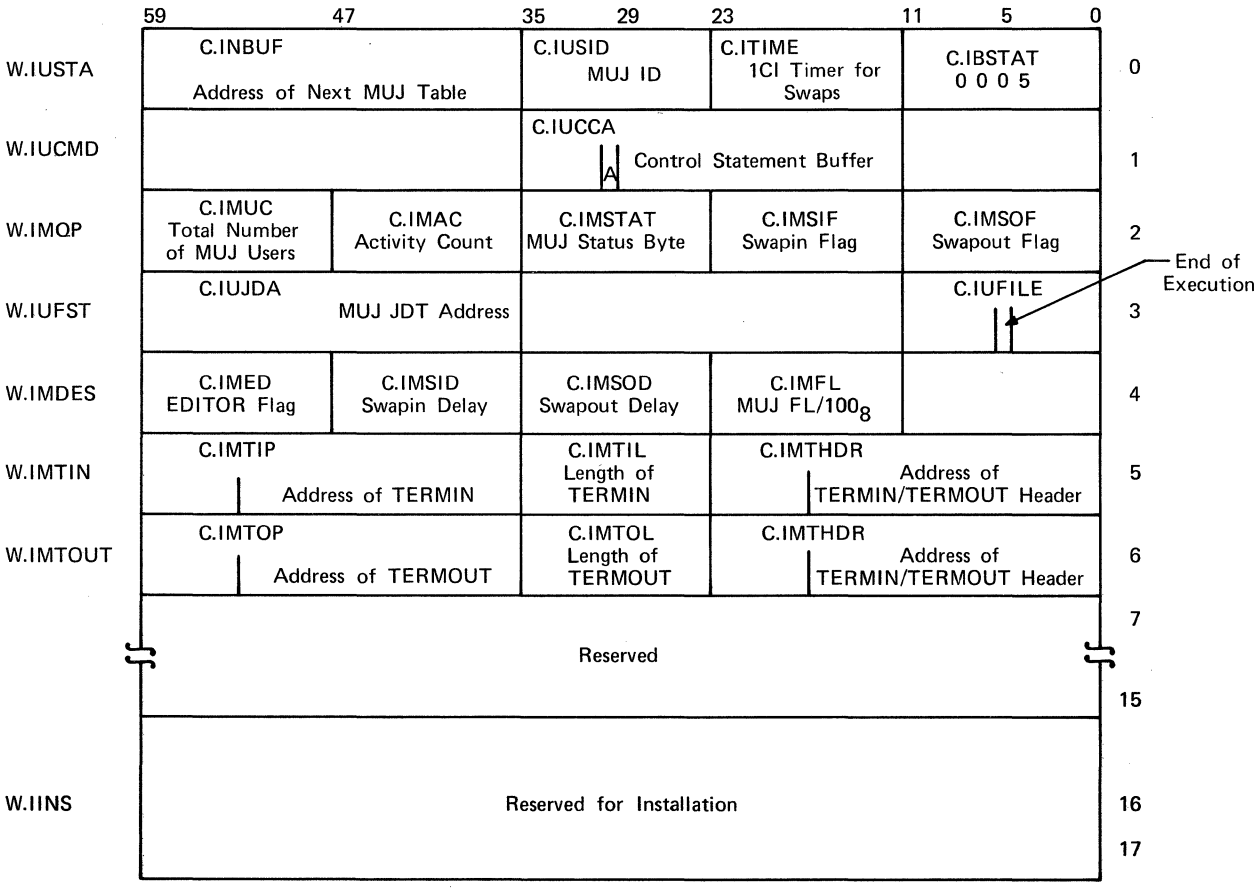
HIGH SPEED AUXILIARY USER TABLE



INPUT DATA STREAM WORD PAIR



MULTIUSER JOB TABLE



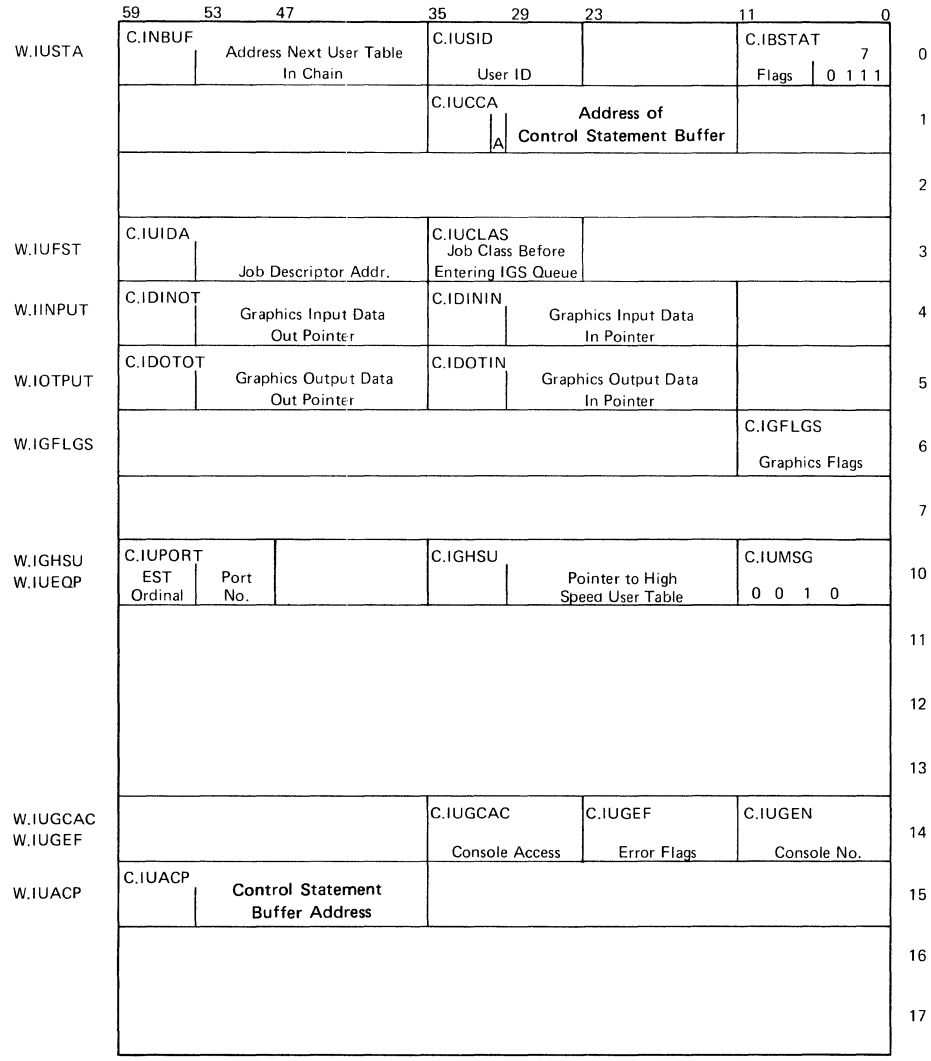
W.IUCMD(1)

Byte	Bit	Field	Description
C.IUCCA	30	S.IUCCP	Control statements moved to CPA flag.

W.IMQP(2)

Byte	Bit	Field		Description
C.IMSTAT	35-24		0001	Waiting for I/O
			0003	MUJ active.

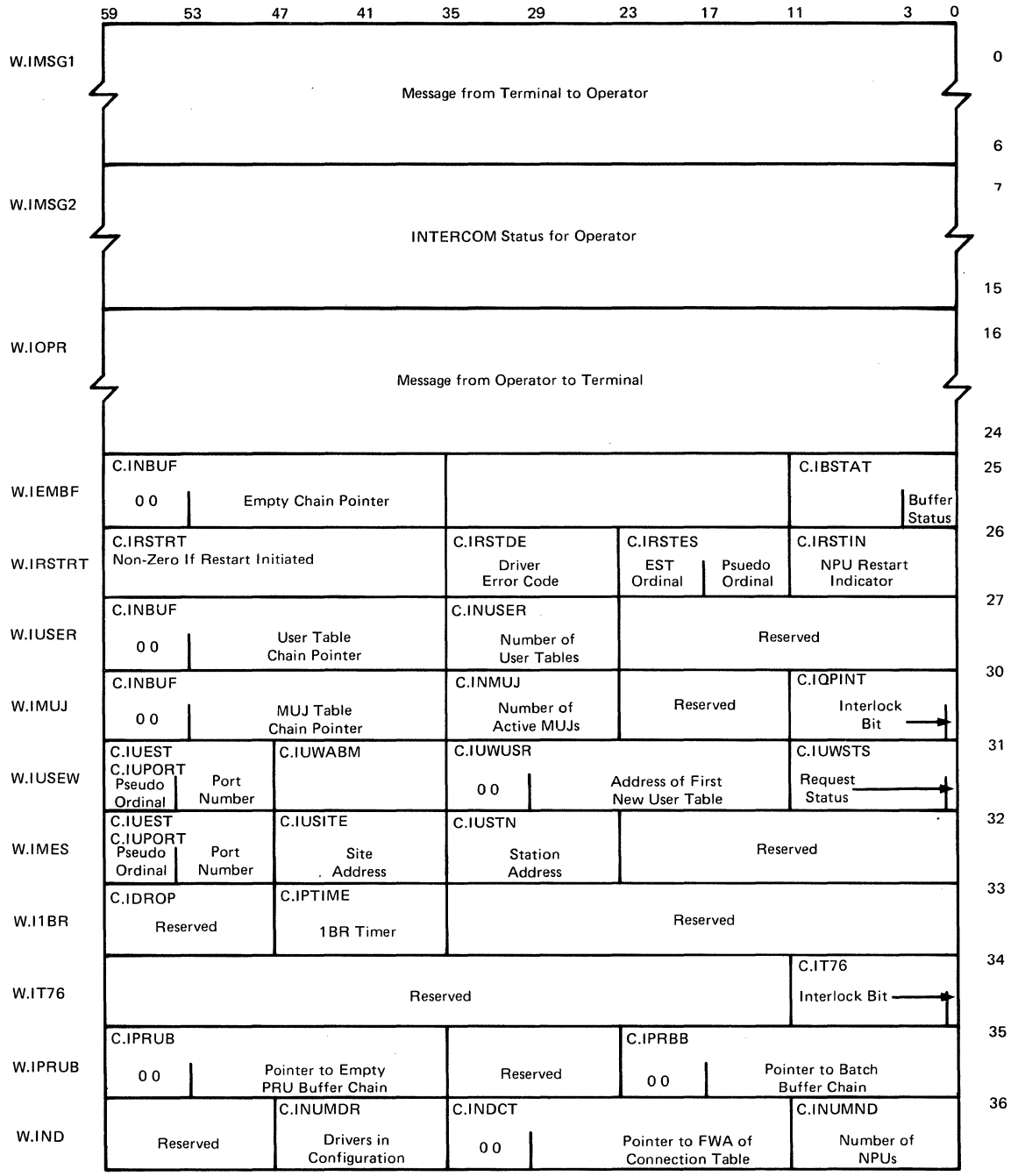
274 INTERACTIVE GRAPHICS SYSTEM (IGS) USER TABLE

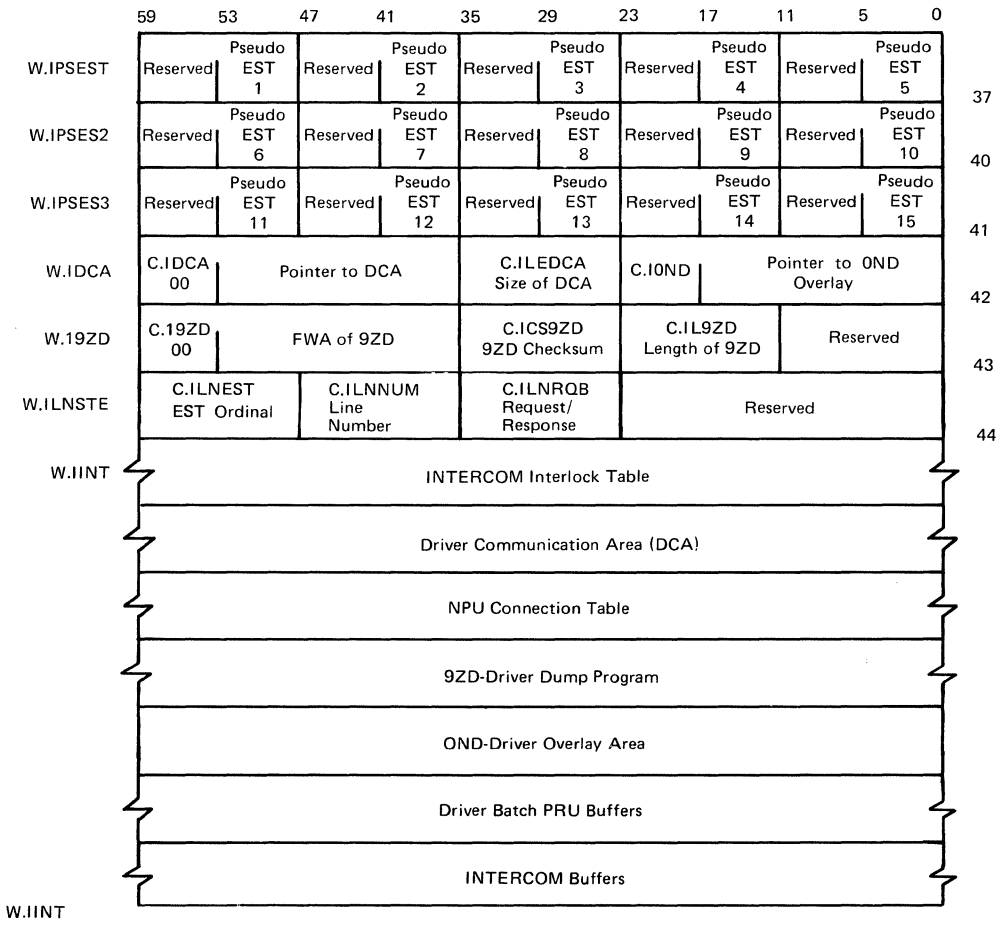


W.IGFLGS(6)

Byte	Bit	Field	Description
C.IGFLGS(4)	11-9	S.IGCON	Console number for which job is queued.
	5	S.IG3TT	3TT interlock flag.
	3	S.IGDRP	IGS job termination flag.
	2	S.IGSNON	SIGNON user table flag.
	1	S.IGQUE	Job queued for first console flag.

INTERCOM 5 POINTER AND BUFFER AREA

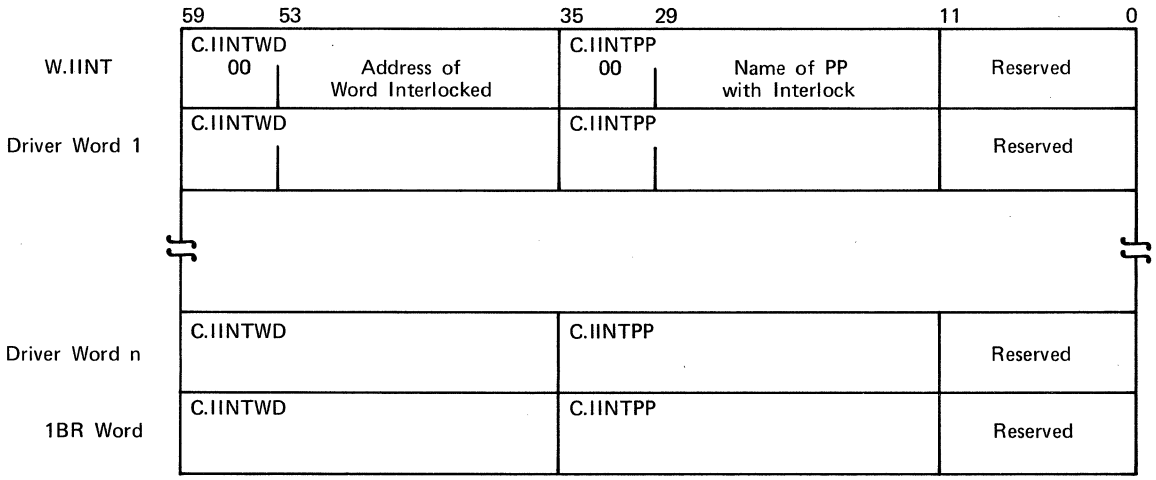




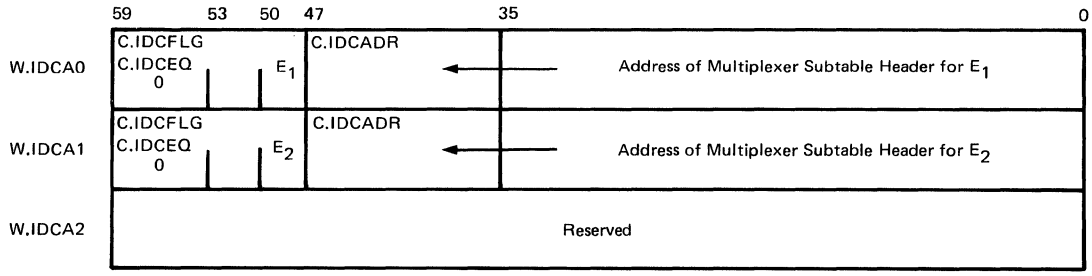
W.ILNSTE(44)

Byte	Bit	Field	Description
C.ILNRQB(2)	11-6	S.ILNRSP	Driver response.
			1 Illegal line number.
			2 Illegal action.
	5-0		Driver request.
			0 Word available (no request pending).
			1 Turn line off.
			2 Turn line on.

INTERLOCK TABLE



NPU DRIVER DCA ENTRY



W.IDCA0(0)

Byte	Bit	Field	Description
C.IDCFLG(0)=C.IDCEQ(0)	11-6	S.IDCFLG	0 indicates no service is required from standard INTERCOM service routines.
	5-3		Reserved.
	2-0	S.IDCEQ	Equipment number of first NPU on channel (E_1).
C.IDCADR(1)	11-0		Address of multiplexer subtable of first NPU on channel.

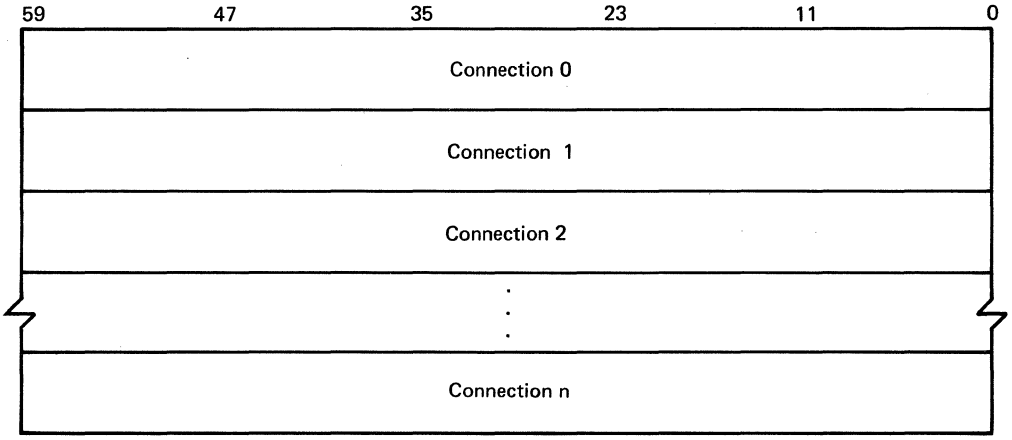
W.IDCA1(1)

Byte	Bit	Field	Description
C.IDCFLG(0)=C.IDCEQ(0)	11-6	S.IDCFLG	0 indicates no service is required from standard INTERCOM service routines.
	5-3		Reserved.
	2-0	S.IDCEQ	Equipment number of second NPU on channel (E_2).
C.IDCADR(1)	11-0		Address of multiplexer subtable of second NPU on channel.

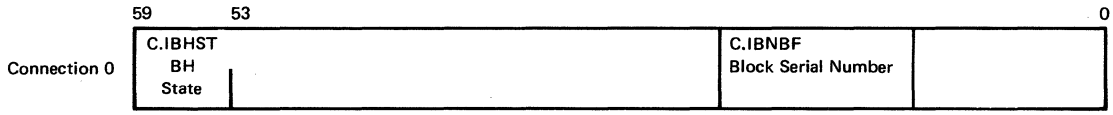
W.IDCA2(2)

Reserved.

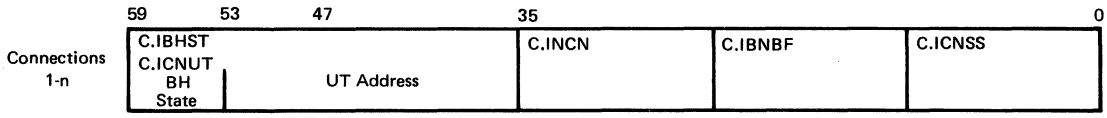
INTERCOM 5 CONNECTION TABLE



CONNECTION TABLE ENTRY FORMATS



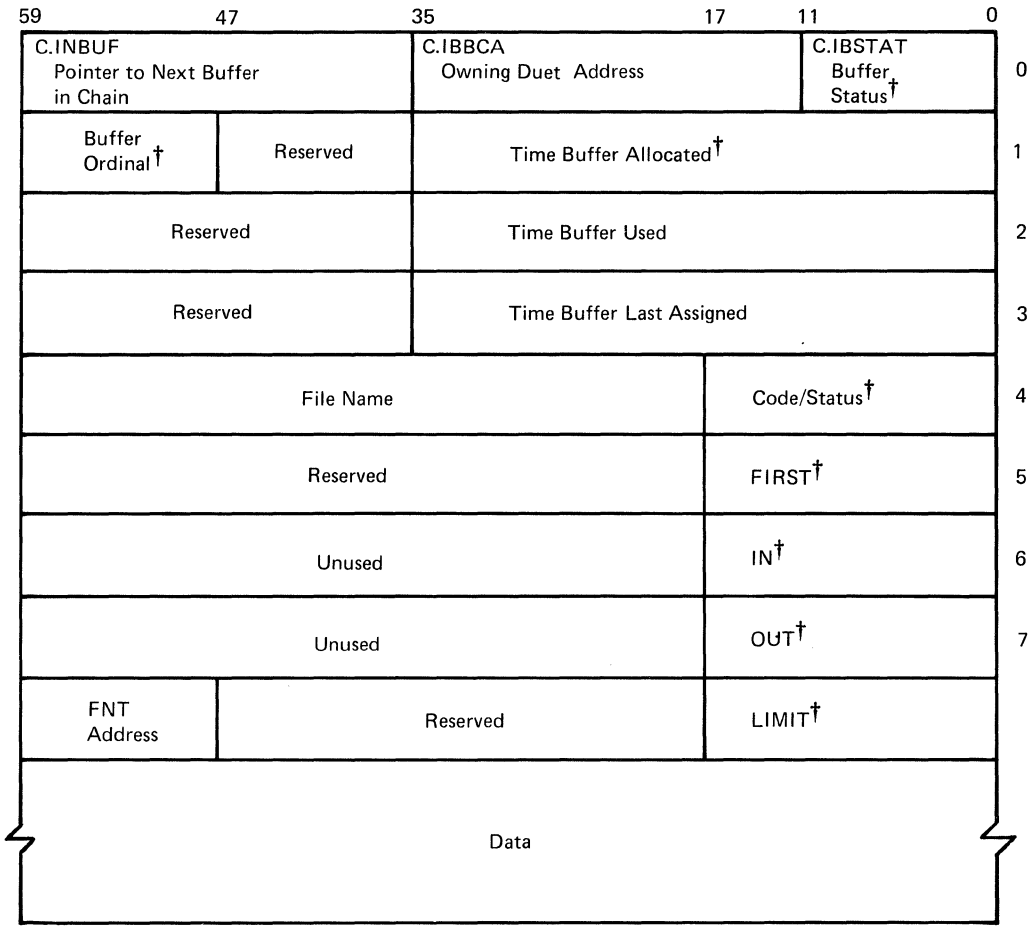
Byte	Bit	Field	Description
C.IBHST(0)	11-6	S.IBHST	Block handler state.
C.IBNBF(3)	11-9		Next upline BSN.
	8-6		Unused.
	5-3		Next downline BSN.
	2-0		Unused.



Byte	Bit	Field	Description
C.IBHST(0)	11-6	S.IBHST	Block handler state position.
C.ICNUT(0)	5-0		Upper 6 bits of user table (UT) address.
C.ICNUT+1(1)	11-0		Lower 12 bits of UT address.

Byte	Bit	Field	Description
C.INCN(2)	11-0	S.INCN	Ordinal of next linked connection table entry.
C.IBNBF(3)	11-9		Next upline BSN.
	8-6		Upline back count.
	5-3		Next downline BSN.
	2-0		Downline back count.
C.ICNSS(4)	11-6		Previous stream state.
	5-0		Current stream state.

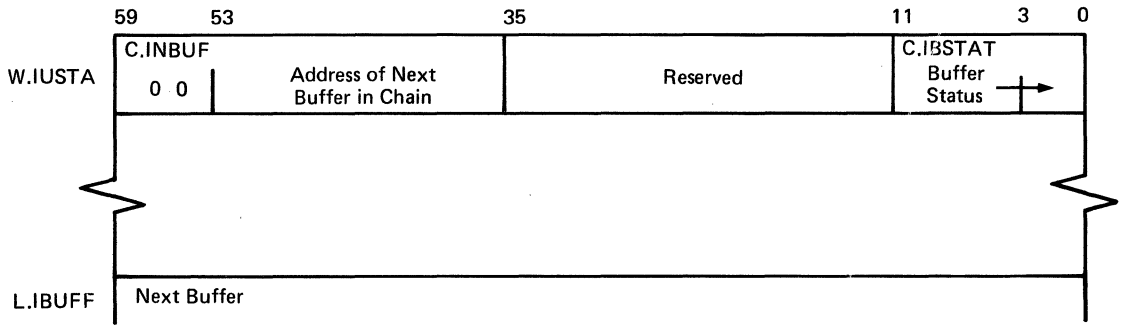
PRU BUFFER



† Initialized by 1I1.

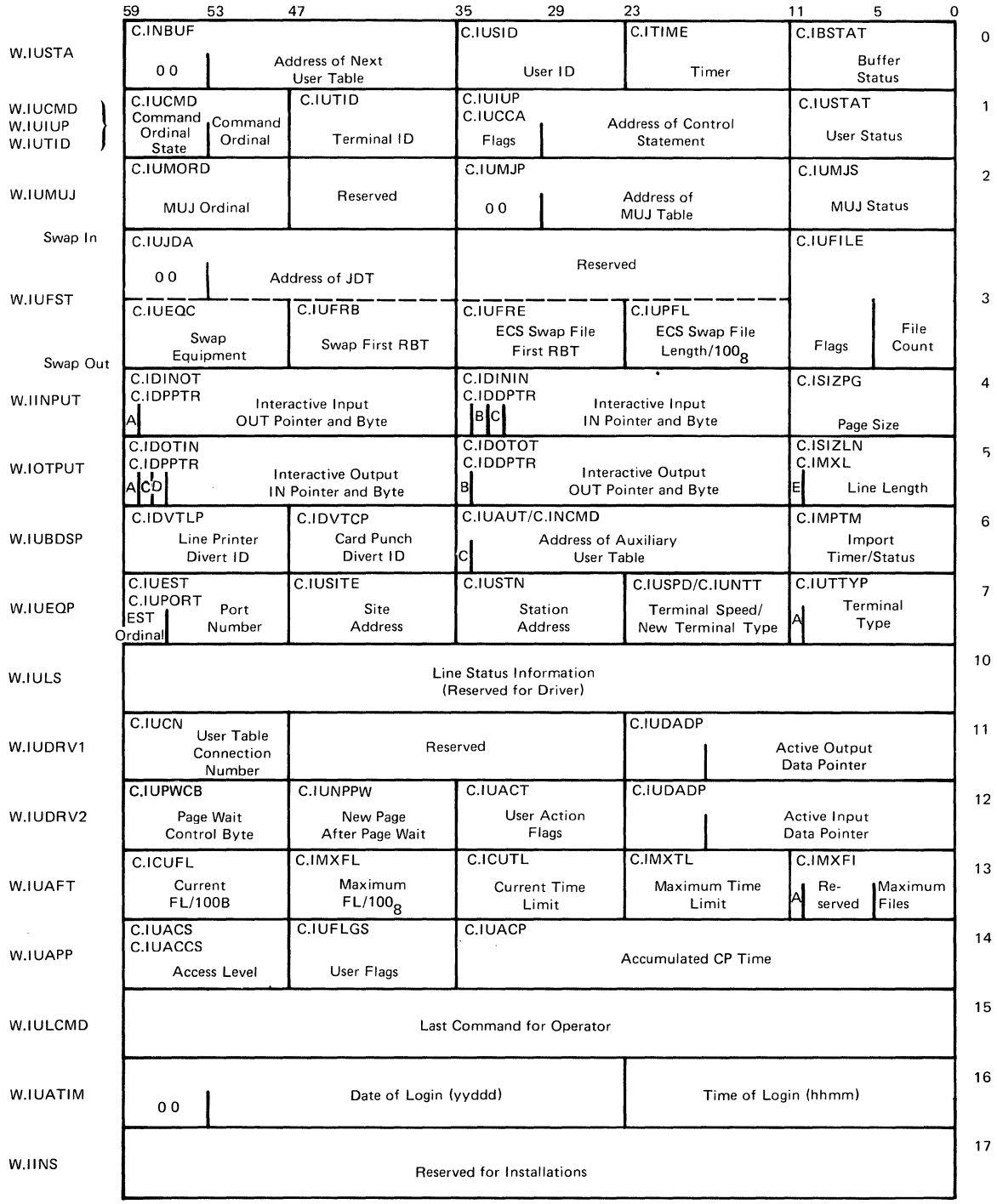
INTERCOM 5 BUFFER AREA

BUFFERS (20_8 Word Boundary)



Byte	Bit	Field	Description
C.IBSTAT	3-0	S.ITYPE	Buffer status (in octal).
			0 Empty.
			1 User table.
			2 Data output/job statement.
			3 Data input.
			4 Interactive control statements.
			5 Multiuser job table (MUJ).
			6 Auxiliary user table (AUT).
			11 Port configuration table.
			12 Limbo buffer.
			13 Batch input buffer.
			14 Batch output buffer.

INTERCOM 5 USER TABLE



W.IUSTA(0)

Byte	Bit	Field	Description
C.IBSTAT(4)	11	S.IABRT	Abort request.
	10	S.IUTAPE	Request paper tape reading.
	9	S.IRDIS	Request disconnected.
	8-7	S.ISTATE	00 Transmission state.
			01 Waiting input.
			10 Waiting output.
			11 Active or assigned to control point.
	6-5	S.ILOGO	00 User logged out.
			01 User logged in.
			10 Automatic logout requested.
			11 Automatic logout in progress.
	4	S.IDISC	Terminal disconnected.
	3-0	S.ITYPE	Buffer type.

W.IUCMD(1)

Byte	Bit	Field	Description
C.IUCMD(0)	59-54		Command ordinal state.
	53-48		Command ordinal.

Ordinal	Command	Description
1B	.ON	Turn on device.
2B	.OFF	Turn off device.
3B	.GO	Go device.
4B	.WAIT	Wait device.
5B	.DEFINE	Define device.
6B	.READFN	Read, filename.
7B	.READ	Read job stream.
10B	.BSP	Backspace output file.

Byte	Bit	Field	Description

Ordinal	Command	Description
11B	.REWIND	Rewind output file.
12B	.RETURN	Return output file.
13B	.REPEAT	Repeat output file.
14B	.END	Terminate file transmission on device.
15B	.SUS	Suspend output activity on device.
16B	.SCREEN	Screen command ordinal to 1NP.
16B	.BLOCK	Block transmission size for bisynchronous terminal.
17B	.CONT	Continue batch operations.

Byte	Bit	Field	Description
C.IUCCA(2)=C.IUIUP(2)	11	S.IUEDC	Buffer contains control statements from editor.
	10		Unused.
	9	S.IUEXS	Start of execution flag.
	8	S.IUECP	Editor control statements processed.
	7	S.INCT	Command in 1CI table.
	6	S.IUCCP	Control statements moved to control point area.
C.IUSTAT(4)	11	S.ICTAPE	Paper tape on flag.
	10	S.INOCOM	1CI should not issue command.
	9	S.IMPORT	IMPORT on at 200 UT.
	8	S.ICBTHW	Allow commands before LOGIN.
	7	S.ICACT	1CI active.
	6	S.ICMES	MES active.
	5-0	S.IUHDIS	Used by IDS for H display ordinal.

W.IUMUJ(2)

Byte	Bit	Field	Description
C.IUMJS(4)	11	S.IMUJ	Attached to MUJ.
	10	S.IMWI	Waiting for input.
	9	S.IMWO	Waiting for output to complete.
	8	S.IMLGS	Logout sent.
	7	S.IMBKS	Break sent.
	6-3		Unused.
	2	S.IMDIS	Reconnected after disconnect.
	1	S.IMRUN	RUN command in progress.
	0	S.IMNUS	New user.

W.IUFST(3)

Byte	Bit	Field	Description
C.IUFILE(4)	11	S.IURED	REDUCE flag.
	10	S.IUFNT	FNT to be associated on next swap.
	9	S.IUECS	Swap file on ECS.
	8	S.IUPS	Pause bit.
	7	S.IUDMP	SAVEFL flag.
	6	S.IUEOE	End of execution.
	5-0	S.IUCNT	Count of FNT entries in swap file.

W.IINPUT(4)

Byte	Bit	Field	Description
C.IDPPTR(0)	11	S.IINTLK	Interlock bit for 3TT and I/O macros.
	8-6	S.IBYTE	Byte position in current word.
C.IDDPTR(2)	10	S.IRLS	Input line not in progress.
	9	S.IDRACT	Driver active.
	8-6	S.IBYTE	Byte position in current word.

W.IOTPUT(5)

Byte	Bit	Field	Description
C.IDPPTR(0)	11	S.IINTLK	Interlock bit for 3TT and I/O macros.
	10	S.IPWOFF	Page-wait-off flag for processor.
	9	S.I1IML	1IM interlock.
	8-6	S.IBYTE	Byte position in current word.
C.IDDPTR(2)	11	S.IPRNOP	Driver not finished.
	8-6	S.IBYTE	Byte position in current word.
C.ISIZLN(4)	11	S.ILNCHG	Line size change bit.

W.IUBDSP(6) Batch data stream pointers for the following:

Byte	Bit	Field	Description
C.IDVTLP(0)			Line printer divert ID.
C.IDVTCP(1)			Card punch divert ID.
C.IUAUT(2)			Auxiliary user table pointer.
	11	S.IUAUTR	AUT requested when set address is multiplexer subtable address.
	10	S.IURAUT	Recover AUT.
	9	S.IURETA	Release AUT (1NP internal flag).

W.IUBDSP(6) Import timer and status byte.

Byte	Bit	Field	Description
C.IMPTM(4)	11	S.IUDCN	EXPORT completed disconnect.
	8	S.IRLAUT	1CI requesting 1NP to release AUT.

W.IUEQP(7)

Byte	Bit	Field	Description
C.IUEST(0)	11-8	S.IUEST	Pseudo EST ordinal.
	7-0	S.IUPORT	Port number.
C.IUSPD(3)=C.IUNTT(3)	11-10		Reserved.
	9-5		Terminal speed.
	4-0		New terminal type.
C.IUTTYP(4)	11	S.IUNOP	Inoperative bit.
	10-0		Terminal type (refer to byte C.ITTYPE of the synchronous terminal port entry of the multiplexer subtable for bit definitions).

W.IULS(10) Reserved for 1ND line status information.

W.IUDRV2(12) User action flags.

Byte	Bit	Field	Description
C.IUACT(2)	11-3		Reserved.
	2	S.IPT	Paper tape mode.
	1	S.ISUP	Suppress output (%S).
	0	S.IABT	User abort required (%A).
C.IUDADP(3)	9	S.IDRACT	Driver active.

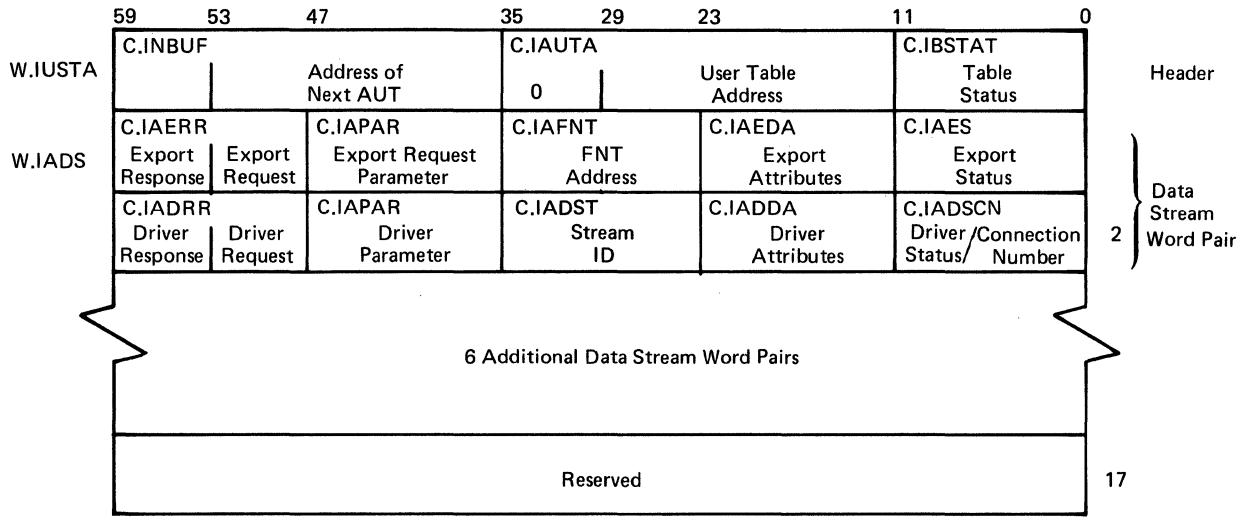
W.IUAFT(13)

Byte	Bit	Field	Description
C.IMXFI(4)=C.IUVC(4)	11	S.IUVC	VC carriage control.
	10-6		Reserved.
	5-0		Maximum number of files.

W.IUAPP(14)

Byte	Bit	Field	Description
C.IUFLGS(1)	11	S.IUUNR	Login with unrestricted password.
	10	S.ITLOK	Lock bit.
	9	S.IRLOK	Request job to be locked out.
	8	S.IUAT	Abort user; no terminal interrupt.
	7	S.INOLK	Job is locked out.
	6	S.IUREU	Reduce mode flag.
	5-0	S.IUSSW	Sense switch changes.

AUXILIARY USER TABLE (AUT)



W.IUSTA(0) Table status.

Byte	Bit	Field	Description
C.IBSTAT(4)	3-9	S.ITYPE	Table type.

W.IAEXWD(0) Export processor word (first of two).

Byte	Bit	Field	Description
C.IAERR(0)	11-6		Export response to driver request.
	5-0		Export request of driver.

W.IAEXWD(0) Export device attributes.

Byte	Bit	Field	Description
C.IAEDA(3)	11-8		Reserved.
	7-3	S.IFCMO	Forms control matrix ordinal.
	2-0	S.ITRAIN	Print train code.

W.IAEXWD(0) Export status.

Byte	Bit	Field	Description
C.IAES(4)	11	S.IPFCFL	Wait PFC full.
	10	S.IENDLP	END,LP command issued.
	9	S.IEOJOF	Turn off device at end-of-file.
	8	S.IREADF	READ,filename command in progress.
	7	S.IOFF	Off.
	6	S.READ	Read in progress.
	5	S.IBAN	Banner on/off. 0 Banner on. 1 Banner off.
	4	S.IO26 †	Character mode for input or punch (IBM HASP/2780/3780 Remote Batch Terminals). 1 O26. 0 O29.
	4	S.IO26 †	Pre/post print for IBM HASP Printer. 0 Pre-print. 1 Post-print.
	3	S.ISUPR	Suppress carriage control on printer or EM option on punch (IBM 2780/3780 Remote Batch Terminals).
	2	S.IMROP	MR option (IBM 2780/3780 Remote Batch Terminals).
	1	S.IWAIT	Wait.
	0	S.ISSTOP	Stream stopped.

W.IADRWD(1) Driver processor word (second of two).

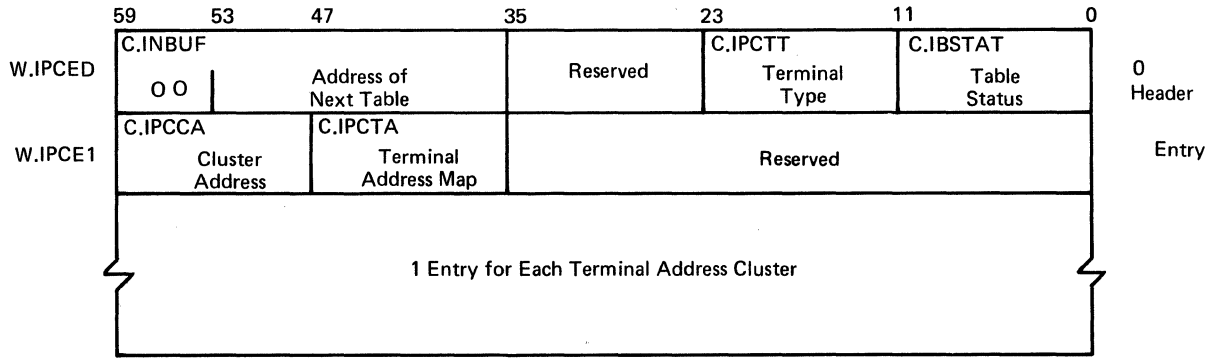
Byte	Bit	Field	Description
C.IADRR(0)	11-6		Driver response to export request.
	5-0		Driver request of export.

† The S.IO26 field has a different meaning for the card devices and the printers.

W.IADRWD(1) Stream identification.

Byte	Bit	Field	Description
C.IADST(2)	11	S.ISTSTP	Stream stopped.
	10	S.IJBCRD	Job statement expected (input).
		S.IOTRAN	Transparent mode (output).
	9	S.IOEXCS	Extended character set (output).
	8	S.IMSERR	Mass storage error.
	7	S.IINPUT	Input stream identification.
	6	S.IOTPUT	Output stream identification.
	5-3	S.ISTYPE	Stream type.
	2-0	S.IDEVNR	Device number.

PORT CONFIGURATION TABLE



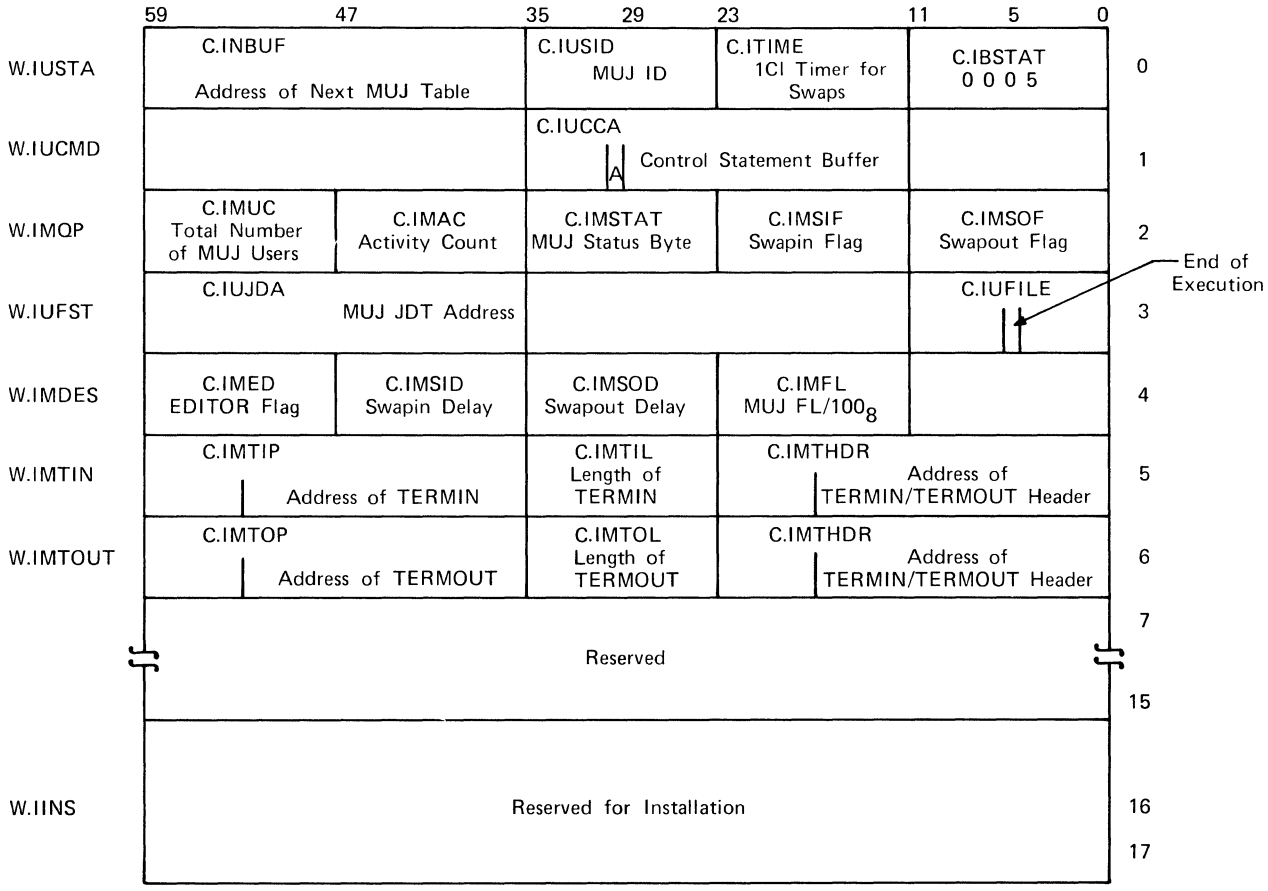
W.IPCED(0)

Byte	Description
C.IPCTT(3)	Terminal type; refer to byte 4 (C.ITTYPE) of synchronous terminal port entry for terminal type definitions.

W.IPCED(0) Buffer status.

Byte	Bit	Field	Description
C.IBSTAT(4)	7-4	S.IPCTWC	Port configuration word count.
	3-0	S.ITYPE	Table type.

MULTIUSER JOB TABLE



W.IUCMD(1)

Byte	Bit	Field	Description
C.IUCCA	30	S.IUCCP	Control statements moved to CPA flag.

W.IMQP(2)

Byte	Bit	Field		Description
C.IMSTAT	35-24		0001	Waiting for I/O.
			0003	MUJ active.

RECORD BLOCK TABLE ENTRY

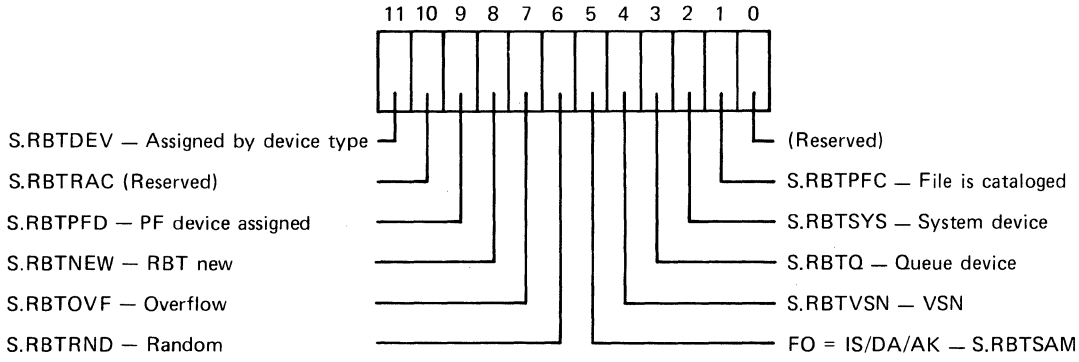
The RBT entries exist in the highest address of CM above the job running at control points.

FILES RESIDENT ON PERMANENT FILE SETS

FIRST RBT WORD PAIR (Type 4 - in CM)

59	47	38	35	29	23	11	0
C.RBTWPL Next Word Pair †	C.RBTDRB DAM Ordinal	7	C.RBTMST MST Ordinal	C.RBTAL Alloc. Type	C.RBTPRU Last PRU + 1	C.RBTBIT Flags	
C.RBTAUS PRUs/RB	C.RBTVSN Volume Serial Number					RB_7	

C.RBTBIT



† When this byte is zero, there are no more word pairs.

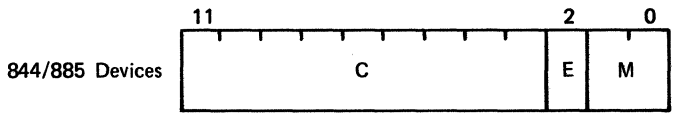
OTHER WORD PAIRS EXCEPT OVERFLOW

59	47	38	35	23	11	0
C.RBTWPL Next Word Pair †	C.RBTDRB DAM Ordinal	0	RB_0	RB_1	RB_2	
RB_3	RB_4		RB_5	RB_6	RB_7	

OVERFLOW WORD PAIRS

59	47	38	35 29	23	11	0
C.RBTWPL Next Word Pair †	777	0	C.RBTMST MST Ordinal	End of Volume PRU + 1	C.RBTODO DAM Ordinal	
C.RBTAUS PRUs/RB	C.RBTVSN Volume Serial Number				0 0 0 0	

RECORD BLOCK TABLE BYTE MINUS ONE



Computation of physical addresses for default allocation styles is as follows:

844/885 half-track (2:1 interlace):

Cylinder = C

Track = (PS*M)/TS

Sector = E+remainder of (PS*M)/TS

E is 0 for even; E is 1 for odd.

† When this byte is zero, there are no more word pairs.

844/885 full-track (1:1 interlace):

Cylinder = C

Track = (PS*EM)/TS

Sector = Remainder of (PS*EM)/TS

Values for PS and TS in the preceding computations are as follows:

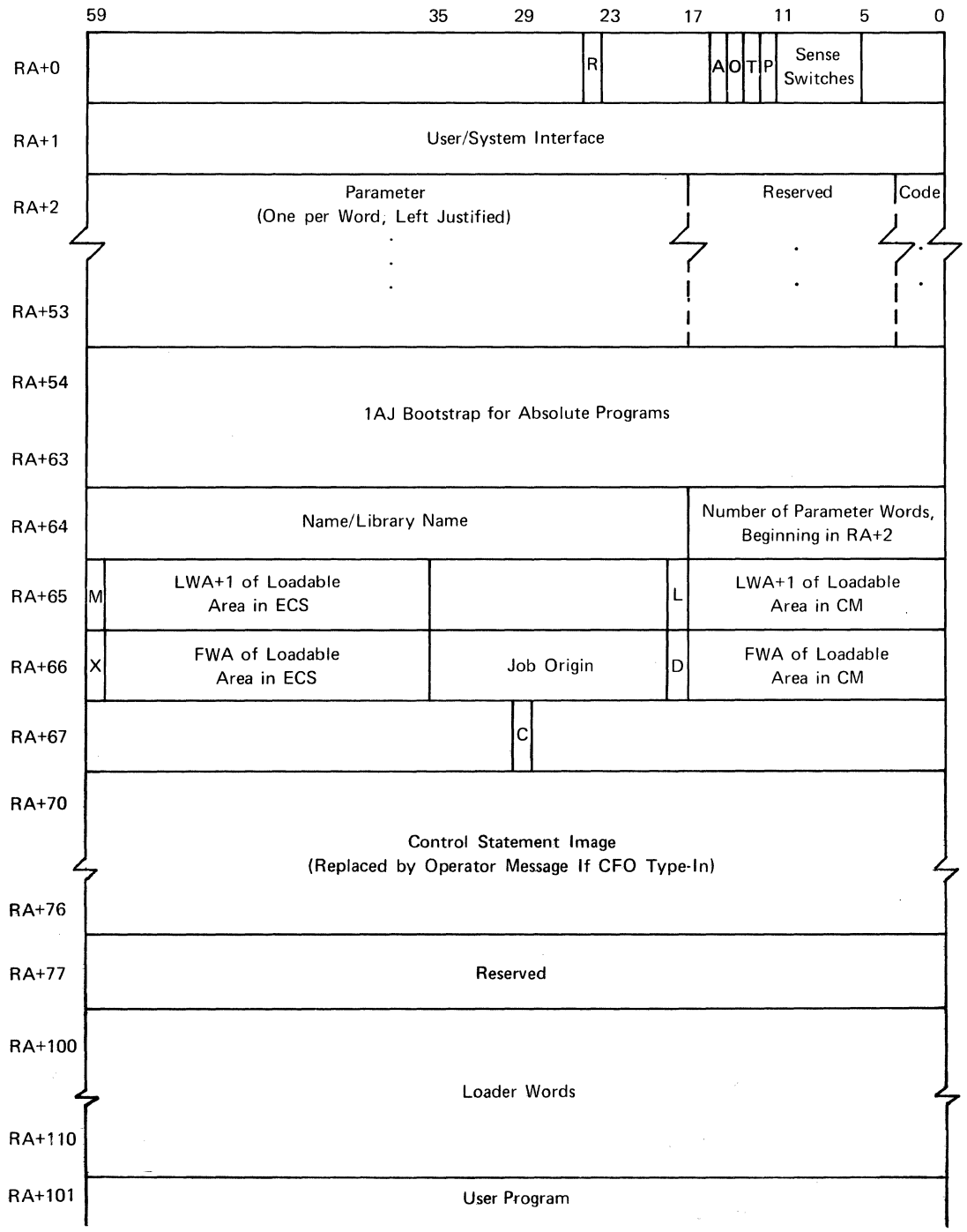
PS PB size in PRUs (decimal).

Device	PRUs
844	114
885	320

TS Track size in PRUs (decimal).

Device	PRUs
844	24
885	40

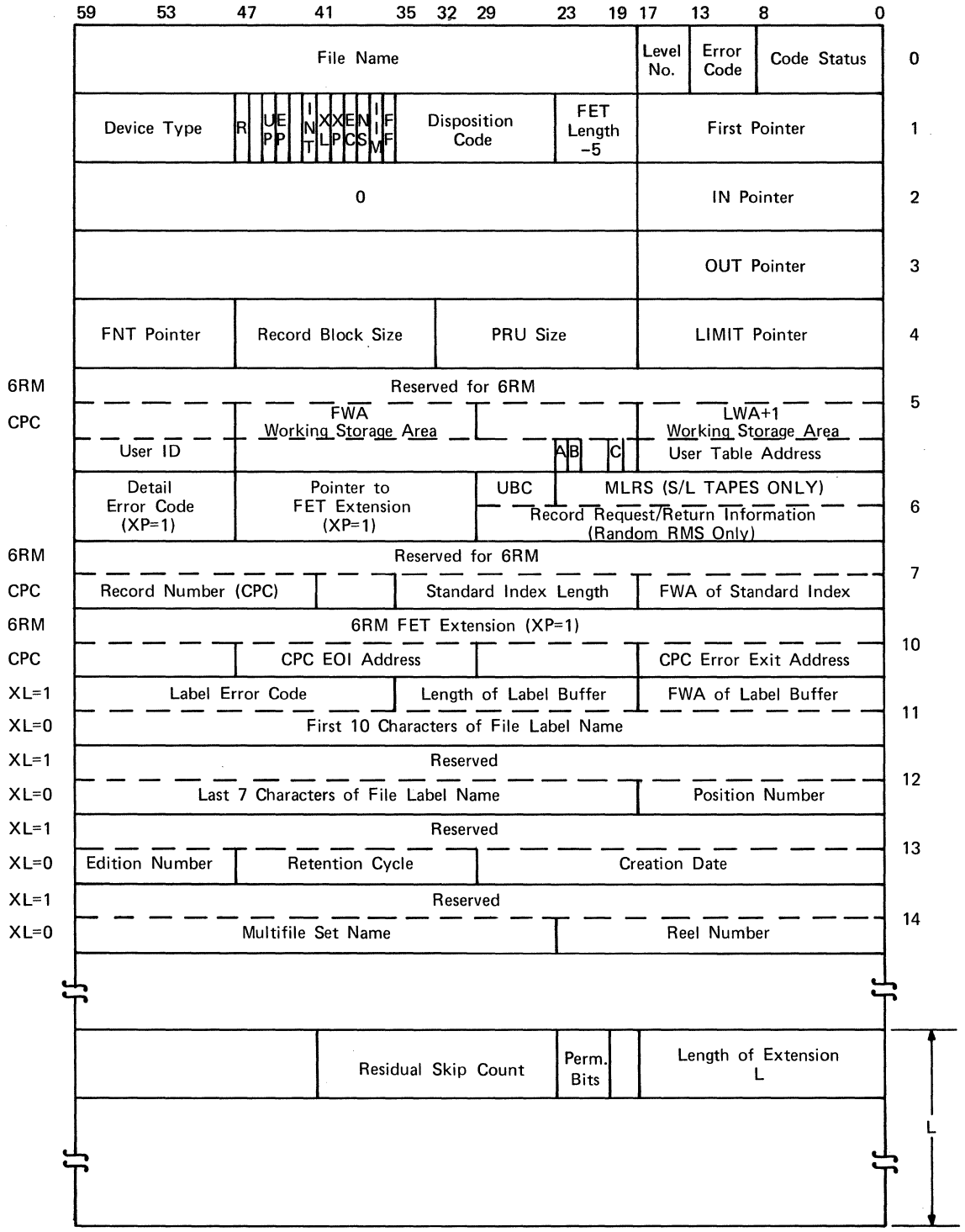
RA COMMUNICATION AREA



Field Name	Description
R	Job dependency recheck bit.
A	Job swap-out to operator action queue flag (1 indicates the job will be placed under operator action queue upon swap-out regardless of job origin).
O	CFO flag (1 is accept comment from operator).
T	Storage move flag (1 is move being attempted).
P	Pause flag (1 is control point pausing).
Code	00 Continuation.
	01 Comma.
	02 Equal sign.
	03 Slash.
	04 Left parenthesis.
	05 Plus sign.
	06 Minus sign.
	07
	10 Semicolon.
	11
	12
	13 Reserved.
	14
	15
	16 Other.
	17 Termination.

Field Name	Description
L	Library/file flag (1 indicates name is the library name).
X	XJ flag; if X is 1, XJ can be issued.
C	LDV completion flag (bit 29).
D	DIS RSS flag (bit 18).
M	CMU bit; if M is 1, CMU can be issued.
Job origin	0 System.
	1 Batch.
	2 Remote batch.
	3 Interactive terminal.

FILE ENVIRONMENT TABLE



CIO CODES IN OCTAL

All codes are shown for coded mode operation; add 2 for binary mode. For example, 010 is coded READ; 012 is binary READ.

Code	Function
000	RPHR
004	WPHR
010	READ
014	WRITE
020	READSKP
024	WRITER
030	-
034	WRITEF
040	BKSP
044	BKSPRU
050	REWIND
054	-
060	UNLOAD
064	-
070	RETURN
074	-
100	OPEN,NR
104	OPEN WRITE,NR
110	POSMF
114	EVICT
120	OPEN,NR
124	-
130	CLOSE,NR
134	-
140	OPEN
144	OPEN WRITE
150	CLOSE

Code	Function
154	-
160	OPEN
164	-
170	CLOSE,UNLOAD
174	CLOSE,RETURN
200	READC
204	WRITEC
210	READLS
214	REWRITE
220	-
224	REWRITER
230	-
234	REWRITEF
240	SKIPF
244	-
250	READNS
254	-
260	READN
264	WRITEN
270-274	-
300	OPEN,NR
304-324	-
330	CLOSER
334	-
340	OPEN
350	CLOSER
354-364	-
370	CLOSER,UNLOAD
374	CLOSER,RETURN

Code	Function
400-474	-
500-574	Reserved for installations.
600	Reserved for NOS READEI.
604-634	-
640	SKIPB
644-774	-

LOCAL FILE NAMES

The following list represents the reserved local file names that appear in a FET for the named product set file.

File Name	Product Set File	Use
ZZCCLAA-ZZ	CCL	CCL files.
ZZZZCKP	NOS/BE	System checkpoint file.
ZZZZECS	EDITLIB	System ECS resident library creation job (permanent file).
ZZZZZ01	EDITLIB	Reset file (permanent file).
ZZZZZ02	EDITLIB	Restore file (permanent file).
ZZZZZ03	EDITLIB	System extend file (permanent file).
ZZZZZ04	EDITLIB	System file (permanent file).
ZZZZZ05	EDITLIB	Interpreted directives.
ZZZZZ06	EDITLIB	ECS resident routines library file.
ZZZZZ07	EDITLIB	Entry point name table spill file.
ZZZZZ08	EDITLIB	Program number table spill file.
ZZZZZ09	DIAXNOS	
ZZZZZ0A	FTN45	Comments file.
ZZZZZ0G	FTN45	Token list file.
ZZZZZ0H	FTN45	Name table file.
ZZZZZ0I	FTN45	Invented name file.
ZZZZZ0J	FTN45	Data statement analysis file.
ZZZZZ10	EDITLIB	Program name table spill file.
ZZZZZ11	EDITLIB	External reference table spill file.

File Name	Product Set File	Use
ZZZZZ12	EDITLIB	External reference collection spill file.
ZZZZZ13	EDITLIB	Library or deadstart program collection file.
ZZZZZ14	EDITLIB	Scratch.
ZZZZZ15	EDITLIB	PP program name table spill file.
ZZZZZ16	EDITLIB	Library name table spill file.
ZZZZZ17	LOADER	Entry point list.
ZZZZZ18	LOADER	Owned global blocks.
ZZZZZ19	FORM	
ZZZZZ1A-1Z	SORT/MERGE	
ZZZZZ20	FORM	
ZZZZZ21	FORM	
ZZZZZ22	6RM	Memory manager.
ZZZZZ23	EDITLIB	Current directory file.
ZZZZZ24	QUERY/UPDATE	
ZZZZZ25	LOADER	Global library set.
ZZZZZ26	GRAPHICS	
ZZZZZ27	LOADER	Overlay/segment generator.
ZZZZZ28	DEBUGGING AIDS	
ZZZZZ29	LOADO	ECS hold file.
ZZZZZ2A-2Z	SORT/MERGE	
ZZZZZ30	LOADER	SEGBILD scratch file (random).
ZZZZZ31	LOADER	SEGBILD sort file (random).
ZZZZZ32	LOADER	SEGBILD sort file (random).
ZZZZZ3A-3Z	SORT/MERGE	
ZZZZZ41-49	COBOL	
ZZZZZ50-59		Reserved for Control Data.
ZZZZZAA-A9	Index Processor	
ZZZZZAD	TRANSPF	
ZZZZZBA-B0	Index Processor	

File Name	Product Set File	Use
ZZZZZC0-C2	CCL	CCL files.
ZZZZZC3	CDC Special Systems	
ZZZZZC4	CDC Special Systems	
ZZZZZCB	DDL	
ZZZZZCC	DDL	Scratch file.
ZZZZZCD	DDL	Scratch file.
ZZZZZCF	SYMPL	Reserved.
ZZZZZCP	INTERCOM	Copy permanent files.
ZZZZZCR	SYMPL	Reserved.
ZZZZZCR	CDCS2	Rollout file.
ZZZZZCS	CDCS2	Rollout file.
ZZZZZDB	COBOL Debug File	
ZZZZZDC	BASIC	Debug binary file.
ZZZZZDD	System Dynamic Dump	Core-image dump file.
ZZZZZDF	6RM/LOADER	File control statement processor.
ZZZZZDI	CID	Reserved for CYBER Interactive Debug.
ZZZZZDM	NOS/BE	FNT used by SPM to access DAM.
ZZZZZDO	CID	Reserved for CYBER Interactive Debug.
ZZZZZDP	FORTRAN 5	Post Mortem Dump core-image dump file.
ZZZZZDS	CID	Reserved for CYBER Interactive Debug.
ZZZZZDT	CID	Reserved for CYBER Interactive Debug.
ZZZZZEF	6RM	Error message file.
ZZZZZEG	CRM	Error message file.
ZZZZZFC	FTN4	Symbolic object code file.
ZZZZZGN	SYMPL	Reserved.
ZZZZZGO-9	NOS	Reserved.
ZZZZZI1	ALGOL 4	Scratch file.
ZZZZZI2	ALGOL 4	Scratch file.
ZZZZZI4-I9	ALGOL 5	Scratch files.

File Name	Product Set File	Use
ZZZZZIC	SYMPL	Reserved.
ZZZZZIL	SYMPL	Reserved.
ZZZZZIN	INTERCOM Utility/QU/CID	Connected files.
ZZZZZL1-L9	LDCMR	Scratch files.
ZZZZZMP	FORTRAN 5	Post Mortem Dump load map file.
ZZZZZNA-NZ	PL/I	Reserved.
ZZZZZOD	TRANSPF	
ZZZZZOP	FTN/COMPASS	
ZZZZZOU	INTERCOM Utility/QU/CID	Connected files.
ZZZZZQU	QUERY/UPDATE	
ZZZZQ1-Q6	QUERY/UPDATE	
ZZZZZPA	PFM	Scratch file.
ZZZZZPB	PFM	Scratch file.
ZZZZZPC	PFM	Attached RBTC.
ZZZZZPD	PFM	Attached PFD.
ZZZZZPE	PFM	Reserved.
ZZZZZPF	PFM	Attached permanent file.
ZZZZZPG	PFM	Reserved.
ZZZZZPK	MMF-PFN	Multimainframe packet file.
ZZZZZPS	SYMPL	Reserved.
ZZZZZPT	PFM	Permanent file dump tape.
ZZZZZPW	PFM	Attached permanent file DUM.
ZZZZZRE	INTERCOM	Restricted passwords.
ZZZZZRL	FTN4/COMPASS	
ZZZZZRM	FTN4/COMPASS	
ZZZZZRN	PAGE Utility	Interim random page file.
ZZZZZRT	PFM	
ZZZZZS1	DDL	
ZZZZZS2	DDL	

File Name	Product Set File	Use
ZZZZZSA-SD	SIFT	
ZZZZZSE	EDITOR	
ZZZZZSF	EDITOR	
ZZZZZSG	EDITOR	
ZZZZZSH	EDITOR	
ZZZZZSY	FORTRAN 5	Post Mortem Dump symbol tables.
ZZZZZTC	TRANSPF	
ZZZZZTD	TRANSPF	
ZZZZZUI	CID	Reserved for CYBER Interactive Debug.
ZZZZZUN	INTERCOM	Unrestricted passwords.
ZZZZZVx-Zx		Reserved for installations.

ENTRY POINT NAMES

Name	Product
AGxxxxx, ALxxxxx	ALGOL
ATxxxxx	APT
BAxxxxx	BASIC
CBxxxxx, COxxxxx	COBOL
CPxxxxx	COMPASS
D.xxxxx	COBOL
DIxxxxx	CE Diagnostics
EBxxxxx	8231 IMPORT
ECxxxxx	EXPORT IMPORT 200
EHxxxxx	6000 EXPORT High Speed
FExxxxx, FXxxxxx	FORTRAN Extended Version 4
FMxxxxx	FORM

Name	Product
G6xxxxx	IGS/6000 EXPORT HS
G7xxxxx	IGS/1700 IMPORT
INxxxxx	INTERCOM
IXxxxxx	Index Processor
ISxxxxx	SIS 1.0
ITxxxxx	INTERCOM
I7xxxxx	1700 IMPORT HS
I8xxxxx	8231 IMPORT HS
JVxxxxx	JOVIAL
MIxxxxx	1700 MSOS IMPORT HS
MRxxxxx	MARS VI
OHxxxxx	OPHELIE
OPxxxxx	OPTIMA
OTxxxxx	
PLxxxxx	PL1
PTxxxxx	PERT/TIME
QUxxxxx	QUERY UPDATE
RMxxxxx	6RM
SCxxxxx	SCOPE
SIxxxxx	SIMSCRIPT
SMxxxxx	SORT/MERGE
SOxxxxx	
SSxxxxx	SIMSCRIPT
SUxxxxx	SIMULA
Uxxxxxx	Reserved for installation.
Vxxxxxx	
Wxxxxxx	
Xxxxxxx	
Yxxxxxx	
Zxxxxxx	

FET CODES

Word 0 — Error Codes

Code	Description
01	End of information.
02	End of reel.
04	Parity error.
10	Device capacity exceeded.
11	Implicit MOUNT inhibited.
20	Additional error status returned.
21	End of multifile set.
22	Fatal error.
23	Index buffer full.
24	Interlock broken for shared RMS.
25	Index full on random read/write of record n.
26	Nonexistent record named on random read.
27	Nonexistent record named on random write and index is full.
30	Function undefined on device.
31	Permission not granted.
32	Function illegal on permanent file.
33	No public set has required attributes.
34-37	Reserved.

Word 1

Meaning if bit is set:

Bit	Field	Description
47	R	Process standard index if OPEN/CLOSE; otherwise, random read/write.
46		Reserved.
45	UP	User processing at end of volume.
44	EP	User processing on error condition.

Bit	Field	Description
43		Reserved.
42	INT	Allows use of the INTERCOM word (FET+5) which is needed to set ASCII 256 mode or ASCII 128 mode, or to enable multiline reads.
41	XL	Extended label processing.
40	XP	Extended error processing.
39	EC	Disallow automatic allocation of ECS buffer.
38	NS	File has nonstandard labels; processing of label records is left to user.
37	IIM	Inhibit implicit MOUNT.
36	FF	File flushing; unwritten data in a sequential file's buffer will be written to mass storage if a job step ends abnormally, even if the file has scratch disposition.

Word 5

Meaning if bit is set:

Bit	Field	Description
23	A	Set for ASCII 256 mode.
22	B	Set for ASCII 128 mode.
19	C	Set for multiline reads. Normally, only one input line is transferred into the circular buffer. When this bit is set, multiple lines are transferred if they are available and if there is room in the buffer.

Word 6 — Detail Error Codes (Bits 59-48)

When the XP bit is set to 1, this field contains extended tape error processing codes which give additional detail of abnormal conditions resulting from the last input/output operation. The user is responsible for clearing this field after reading it.

Codes 1 through 77 (octal) are considered software warnings to the user; they are not results of hardware failures. The tape-related codes and subsequent software warnings are as follows:

Error Codes (Octal)	Software Warning
24	Read error in opposite mode.
25	Function not complete.
27	Record fragment possible.
30	Data read exceeds MLRS/PRU size.
31	Multifile set ill-formed.

Error Codes (Octal)	Software Warning
32	Write attempt on protected volume.
33	Write at 200 bpi not allowed on 66x tape drive.
35	Multifile name not found on multifile device.
36	Next volume unknown.
37	File not allowed on assigned device.

Codes 100 through 177 (octal) are considered cases where the tape unit has lost position. These codes are as follows:

Error Codes (Octal)	Position
100	Position uncertain; data intact.
101	Position uncertain; data destroyed.
102	Physical/logical positions disagree.
103	Position uncertain; ready dropped during last operation.

Codes 200 through 277 (octal) are considered unit-oriented errors. Switching physical tape devices allows the program to continue after repositioning. These codes and subsequent errors are as follows:

Error Codes (Octal)	Unit
200	System error; tape table.
201	Hardware; unit hung busy.
202	Hardware; no end of operation.
203	Hardware density change during I/O.
204	Unit reserved by another buffer controller.
205	Loop fault.
206	Unable to read tape label just written.
207	Marginal transport indication.
210	Lost data.
211	Multiple load points on tape.
212	No read after write.
213	Coldstart.
214	Irrecoverable write reposition error.
215	Tried to use unit which is down.

Codes 400 through 477 (octal) are errors resulting from hardware failure between the PP and the physical tape unit. These codes and subsequent errors are as follows:

Error Code (Octal)	Data Path Error
400	Hardware; 668x malfunction.
402	Hardware; 6681 failed, no data on IAN.
403	Hardware; transmission parity error.
404	System error.

Codes 1000 through 1005 (octal) are errors resulting from a bad tape. These codes and subsequent errors are as follows:

Error Codes (Octal)	Tape (Medium)
1000	Tape parity error.
1001	25 feet erased tape.
1002	Blank tape read.
1003	Incomplete erasure of tape bad spot.
1004	Noise in IRG.
1005	Erase limit reached.

Codes 6000 through 7777 (octal) are reserved for installations.

Codes are combined meanings of the following bits:

11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TM	CE	UE	PL	DE	DE	DE	DE	DE	DE

TM Tape medium.

CE Controller error (controller, 6681, and so on).

UE Unit caused error.

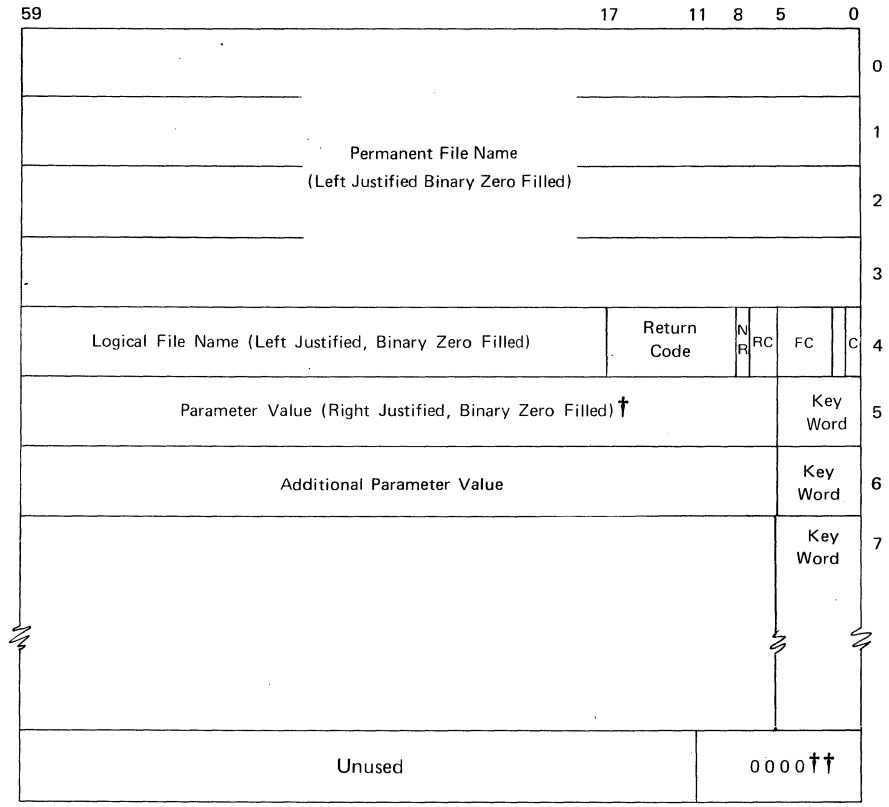
PL Position lost.

DE Detailed error.

The references to system noise record and last good record refer to procedures the system follows in recovery attempts.

Detailed error codes allow a central processor program to take appropriate action when a nonuser-caused error occurs. For example, the message UBC IN FET TOO LARGE does not have a detailed error code because it is a user-caused error. On the other hand, the message TAPE PARITY ERROR is assigned to a detailed error code because the condition is an externally-caused error.

FILE DEFINITION BLOCK



Word 4

Bit	Field		Description
8	NR	1	NR option specified; no automatic recall.
7-6	RC	01	No RC or RT specified.
		00	RC option specified.
		10	RT option specified (implies RC as well).

†If VSN (key word 41) is specified, the one- to six-character volume serial number is contained in bits 59 through 24 with leading display code zero-filled.

††The system checks only bits 11 through 0 of this word.

Bit	Field	Description
5-2	FC	Function code:
		0001 SETP
		0010 ATTACH,GETPF
		0100 CATALOG,SAVEPF
		0110 EXTEND
		0111 ALTER
		1000 PURGE,PURGE(ST=xxx)
		1010 RENAME
		1100 PERM
0	C	Complete bit:
		1 Function completed.

Return code (bits 17 through 9 of word 4) and message written to the job dayfile:

Code	Significance
000	Function successful.
001	ID error.
002	lfn already in use.
003	Unknown lfn.
004	No room for extra cycle (limit is five).
005	PFC full.
006	No lfn or pfn.
010	Latest index not written for a random file.
011	File not on PF device.
012	File not cataloged; SN=setname.
013	Archive retrieval aborted.
014	Bad LPF communication.
015	Cycle number limit reached. Maximum value of cycle number is 999.
016	PFD full.
017	Function attempted on nonpermanent file.

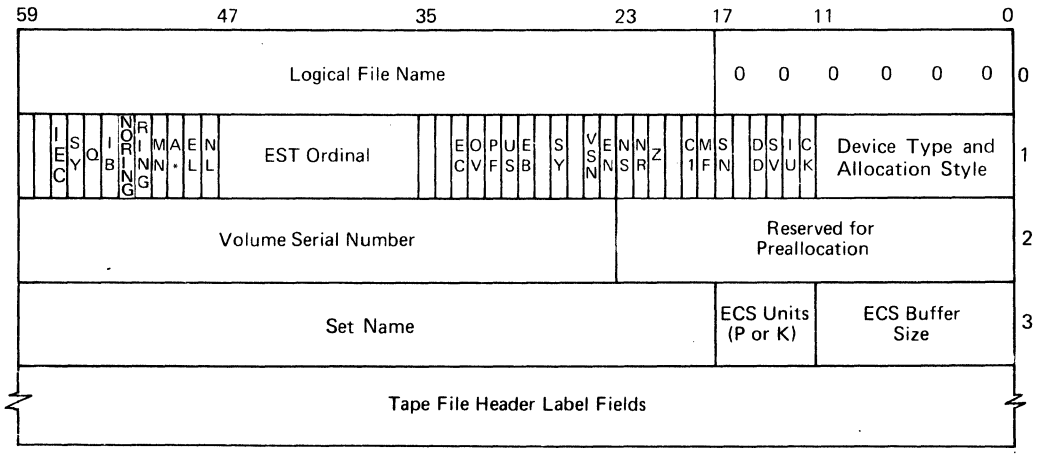
Code	Significance
020	Function attempted on nonlocal file.
021	Improper archive retrieval call.
022	File never assigned to a device.
023	Cycle incomplete or dumped.
024	PF already attached.
025	File archived.
026	Illegal character in FDB parameter.
027	Illegal lfn.
030	File dumped.
031	Illegal function code.
032	Purge attempt ignored; use RB parameter.
033	ALTER needs exclusive access.
034	FDB is too large.
035	File already in system.
036	No APF space.
037	Permission conflicts.
040	Illegal setname specified.
041	Device set not mounted at control point.
042	RBT chain too large for PFC.
043	File resides on unavailable device.
070	PFM stopped by system.
071	Incorrect permission.
072	FDB address error.
073	I/O error on PFD/PFC read/write.

Keyword (bits 5 through 0 of any parameter word) and parameter value:

Value	Parameter
02	RP; retention period in days (binary).
03	CY; cycle number (binary).
04	TK; turnkey password definition (display code).

Value	Parameter
05	CN; control password definition (display code).
06	MD; modify password definition (display code).
07	EX; extend password definition (display code).
10	RD; read password definition (display code).
11	MR; multiread parameter (binary).
13	XR; control, modify, extend password definition (display code).
14	ID; owner-identification (display code).
16	AC; account parameter (display code).
17	EC; ECS buffering (display code).
20	PW; passwords submitted (display code).
21	
22	
23	
24	
25	FO; file organization (display code).
26	PS; position.
30	PF; permanent file name.
31	LC; lowest cycle (binary).
32	ST; staging ID (display code).
33	RW; multiaccess rewrite (binary).
40	SN; set name (display code left-justified).
41	VS; VSN parameter (display code).
43	RB; RB conflict on permanent file.
53	UV; universal password (display code).

REQ FUNCTION PARAMETER LIST

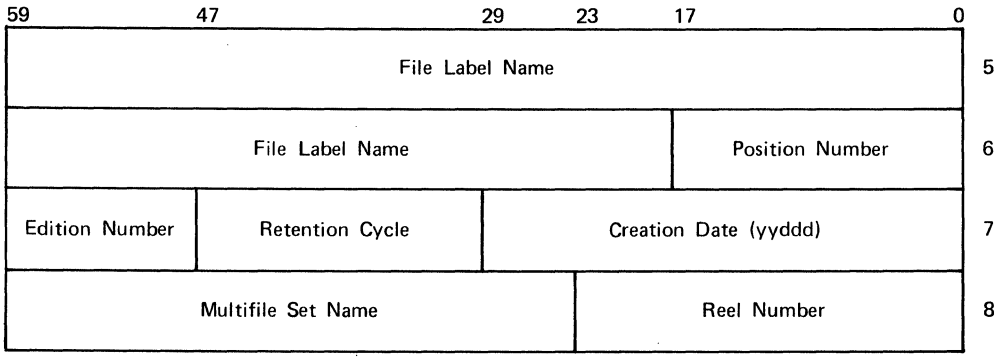


Word	Bit	Field	Description
2	59-58		Reserved.
	57	IEC	1 inhibits error correction on GE (6250 cpi) writes.
	56		1 assigns file to system device in system set.
	55	Q	1 assigns file to queue device (in queue set if no set name given).
	53	NORING	1 prohibits write enable ring in tape.
	52	RING	1 requires write enable ring in tape.
	51	MN	1 accepts either MT or NT assignment; disallows automatic assignment.
	50	A*	1 assigns any RMS device; overrides device-type specification.
	49	EL	1 is extended label fields in parameter words 5 through 9.
	48	NL	1 is normal label fields in parameter words 5 through 8.
	35		Reserved.
	34		Reserved.
	33	EC	1 is ECS buffering requested; parameter word 4.
	32	OV	1 allows overflow to a different device.
	31	PF	1 assigns file to a PF device.

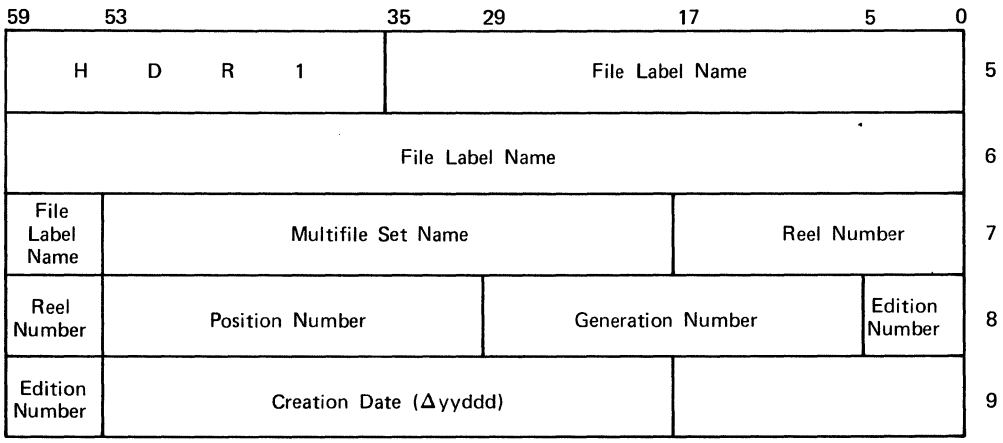
Word	Bit	Field	Description
	30	US	1 is ASCII conversion mode on nine-track tape.†
	29	EB	1 is EBCDIC conversion mode on nine-track tape.
	28		1 assigns automatically.
	27	SY	1 prints card image from RA+70.
	26		1 assigns two devices.
	25	VSN	1 is VSN declared in parameter word 3.
	24	EN	1 is tape has existing labels.
	23	NS	1 is tape has nonstandard labels.
	22	NR	1 is disable standard tape parity recovery procedure.
	21	Z	1 is SCOPE 3.3 labeled tape.
	20		1 returns error code without dayfile message or operator intervention.
	19	CI	1 is console checkpoint request.
	18	MF	1 is multiuser tape request.
	17	SN	1 is set name request.
	16		Reserved.
	15	DD	1 is default density for labels and data.
	14	SV	1 saves tape.
	13	IU	1 inhibits physical unload.
	12	CK	1 is checkpoint tape.
	11-0		Device type and allocation style (binary).

†An * preceding a parameter causes automatic assignment.

NORMAL LABEL FIELDS

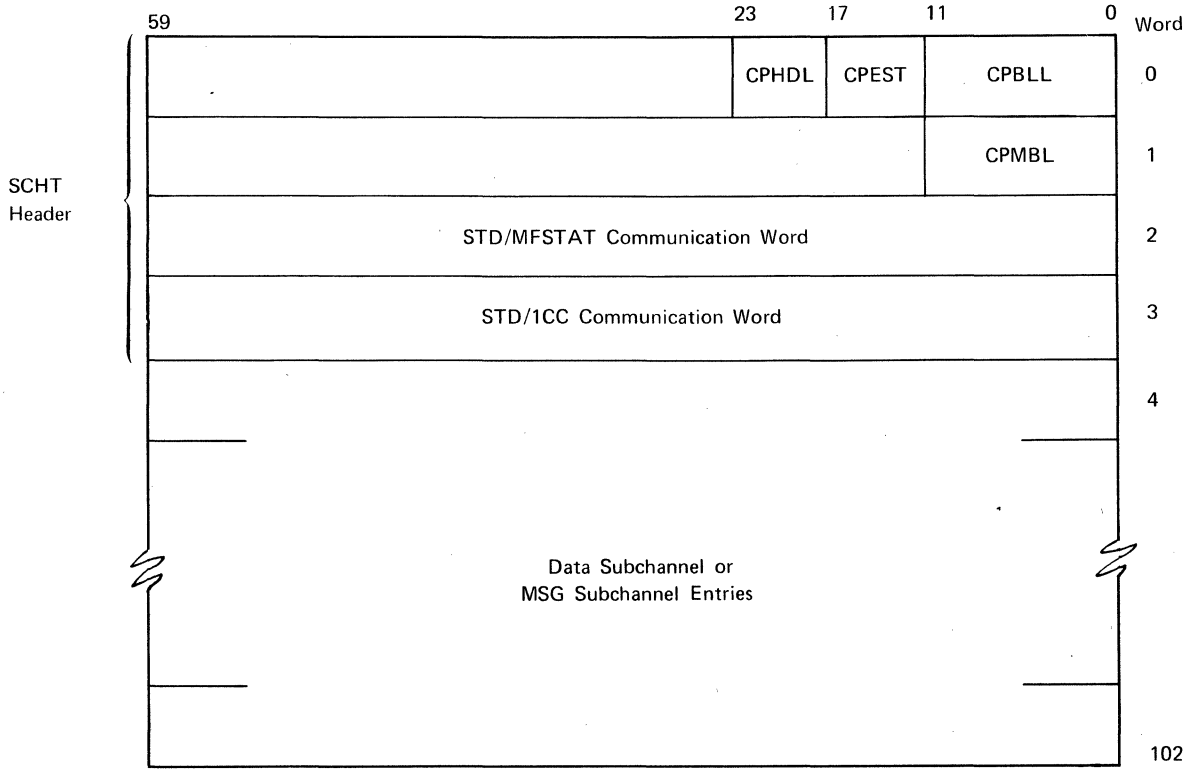


EXTENDED LABEL FIELDS



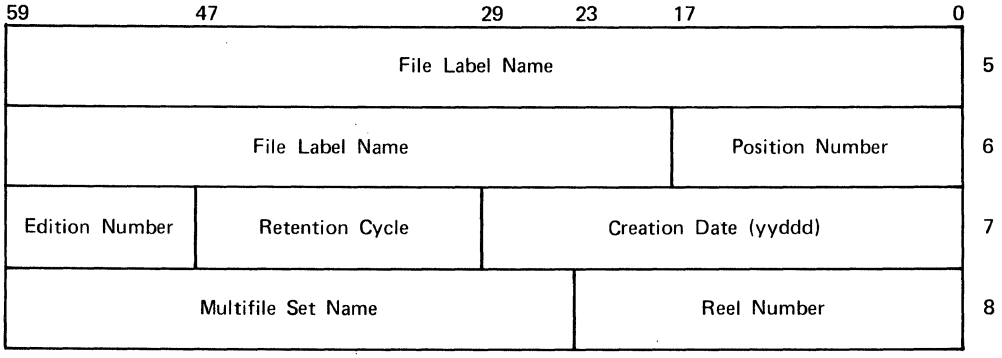
SUBCHANNEL TABLE (SCHT)

The SCHT is used by the CYBER Station in a multimainframe configuration. The SCHT consists of two parts, the header and one-word entries that represent data or message subchannels. This table resides within the field length of the station. Its purpose is to coordinate the transmission/reception of data or messages, to or from the linked mainframe, between the spun-off task (SPOT), STD, and MFSTAT. Word 2 of the header is a communications word between STD and MFSTAT. Word 3 of the header is a communications word between STD and 1CC.

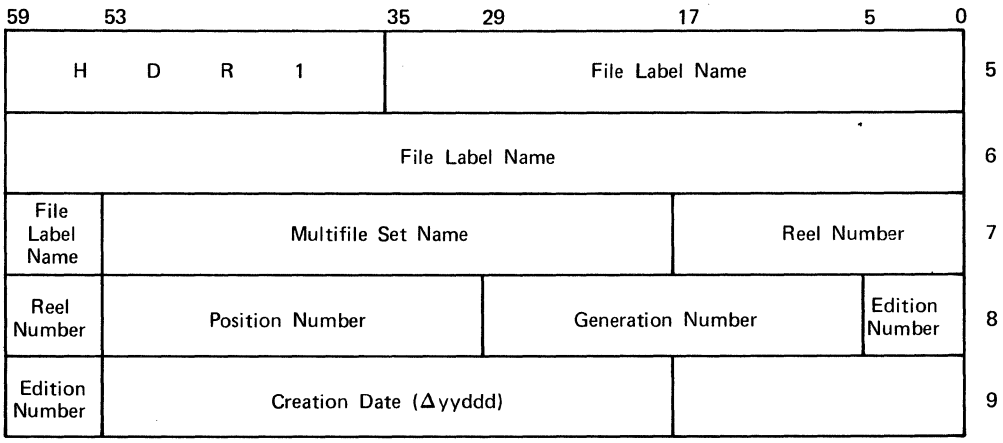


Word	Bit	Field	Description
0	23-18	CPHDL	SCHT header size minus 4.
	17-12	CPEST	EST ordinal.
	11-0	CPBLL	Current standard buffer size.
1	11-0	CPMBL	Maximum standard buffer size.
2	59-54	CPRCD	Request code (station only).
	52-48	CPRSC	Subchannel number.

NORMAL LABEL FIELDS

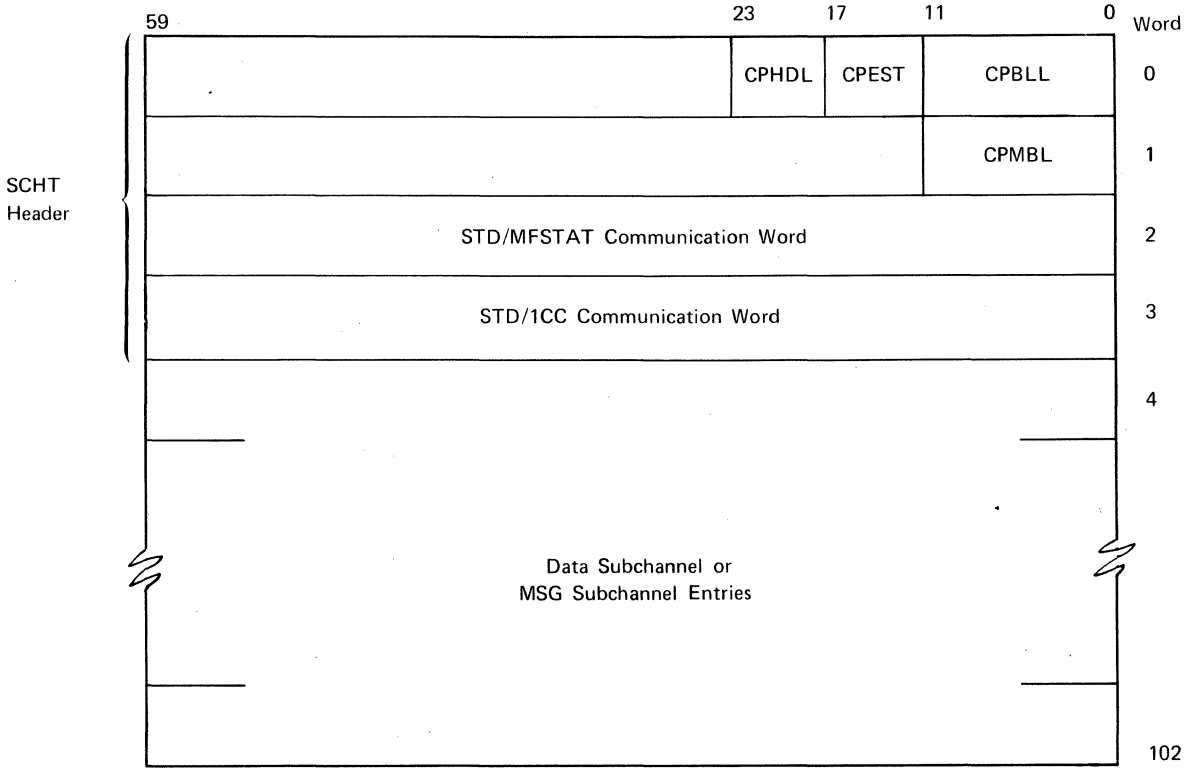


EXTENDED LABEL FIELDS



SUBCHANNEL TABLE (SCHT)

The SCHT is used by the CYBER Station in a multimainframe configuration. The SCHT consists of two parts, the header and one-word entries that represent data or message subchannels. This table resides within the field length of the station. Its purpose is to coordinate the transmission/reception of data or messages, to or from the linked mainframe, between the spun-off task (SPOT), STD, and MFSTAT. Word 2 of the header is a communications word between STD and MFSTAT. Word 3 of the header is a communications word between STD and 1CC.



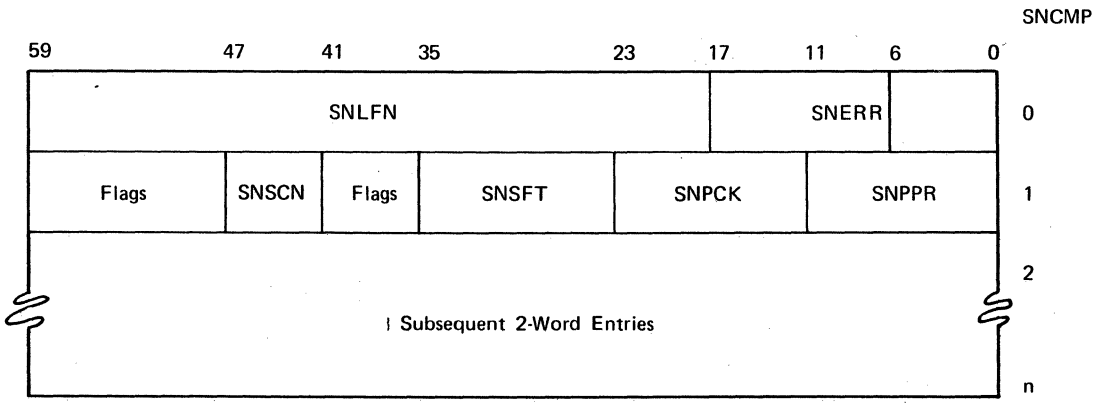
Word	Bit	Field	Description
0	23-18	CPHDL	SCHT header size minus 4.
	17-12	CPEST	EST ordinal.
	11-0	CPBLL	Current standard buffer size.
1	11-0	CPMBL	Maximum standard buffer size.
2	59-54	CPRCD	Request code (station only).
	52-48	CPRSC	Subchannel number.

Word	Bit	Field	Description
3	59-54	PPRCD	Request code.
	58	PPBUS	Request word busy.
	53-48	PPRSC	Subchannel busy.
	47-36	PPRCP	Control point number.
	17-0	PPRFT	FET address.
4-102†	59	LMTTY	Subchannel type. 1 Data. 0 Message.
	58	LMTSB	Indicates transmit buffer full.
	58	LMTDR	1 Receive data subchannel. 0 Transmit data subchannel.
	57	LMTAK	Indicates transmit buffer full and acknowledge expected.
	57	LMTEI	Send end-of-information.
	57	LMTCK	Try to request more data.
	56	LMTCN	End-of-information sent.
	56	LMTRB	Indicate receive buffer full.
	55	LMTUX	Unexpected data.
	54	LMTEX	Excess data.
	52-48	LMTCP	Control point of spun-off task.
	47-36	LMTRL	Receive message buffer length.
	35-18	LMTRC	Receive word count (data subchannel).
	35-18	LMTRF	Receive message buffer address.
	17-0	LMTSF	Transmit message buffer address.
	17-0	LMTFT	FET address.

† Data and message subchannel entries.

SPOT NAME TABLE (SNT)

The SNT is used by the CYBER Station in a multimainframe configuration. The SNT consists of two-word entries that reside within the field length of the station. For each spun-off task (SPOT), there is one entry with the name of the task in the first word of the entry. Its chief purpose is communications between the task and the station, in which 1CC acts as the communicator for the task. The opening, closing, and backspacing of the staged file are coordinated through the SNT. Routines that access this table are 1NS, 1CC, MFSTAT, and DSD.

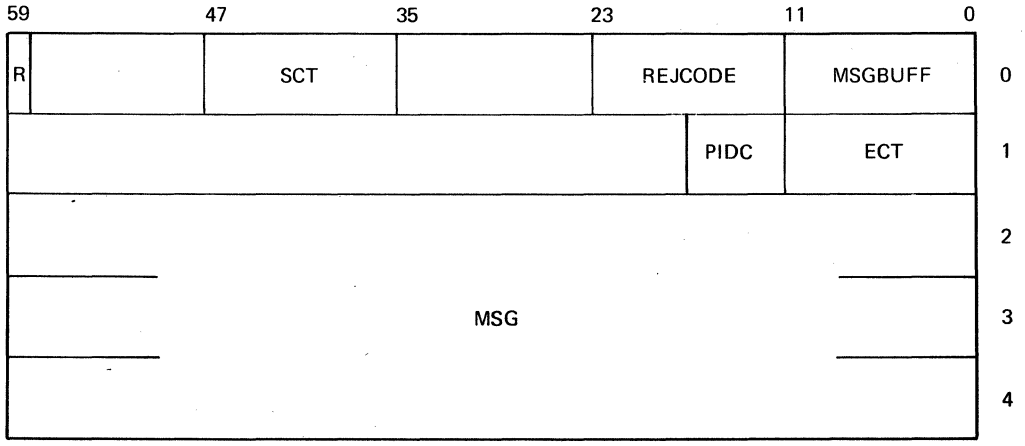


Word	Bit	Field	Description
0	59-18	SNLFN	Name of spun-off task is mnnnncc. m is the identifying character of linked mainframe, and n's are the job ordinal on mainframe m. cc are arbitrary characters assigned by the station.
	17-6	SNERR	1EJ error return code.
	00	SNCMP	Complete indicator set to 0 when task is spun off, and set to 1 when 1EJ goes through end-of-job processing.
1	59	SNIIN	Operation for spooled input file.
	58	SNOIN	Operation for spooled output file.
	57	SNCLO	File close operation.
	56	SNOPN	File open operation.
	55	SNFLK	File linkup chores are in progress or complete (if e is 0).
	54	SNBKS	Backspace operation (for end-of-volume tape stage processing). When this bit is set, bits 17 through 0 contain backspace word count.
	53	SNBUS	1CC busy bit; if on, open, close, or backspace, operation is in progress (set and cleared by 1CC only).

Word	Bit	Field	Description
	52	SNDOP	Open procedure for the task dayfile is complete.
	51	SNFIN	Dayfile transfer completed.
	49	SNRFL	Buffer space obtained for dayfile transmission.
	48	SNRWI	File direction.
			0 Transmit.
			1 Receive.
	47-42	SNSCN	Subchannel number to be used for file I/O.
	41	SNIOR	I/O request message received from linked mainframe.
	41	SNDRM	File is random.
	40	SNODD	Deadstart or dump request received and file can be opened.
	40	SNCAN	Linked mainframe can receive or transmit a spooled file.
	40	SNIOD	I/O delink transmitted to linked mainframe.
	39	SNIOL	I/O linkup transmitted to linked mainframe.
	39	SNCNT	Linked mainframe cannot receive or transmit a spool file.
	38-36	SNSAT	Type of spun-off task.
			1 ATTACH.
			2 CATALOG.
			3 POST STAGE (write).
			4 PRE STAGE (read).
			5 SPOOLED FILE (in or out).
			6 DEADSTART OR DUMP FILE.
			7 LOCAL FILE.
	35-24	SNSFT	File ordinal.
	23-12	SNPCK	Address for spooling file packet (linkage).
	17-0	SNPRU	Number of words to backspace.
	17-0	SNDBF	Dayfile buffer address.
	11-0	SNPPR	Address of subchannel table for mainframe.
2		ENTRY	More two-word entries following.

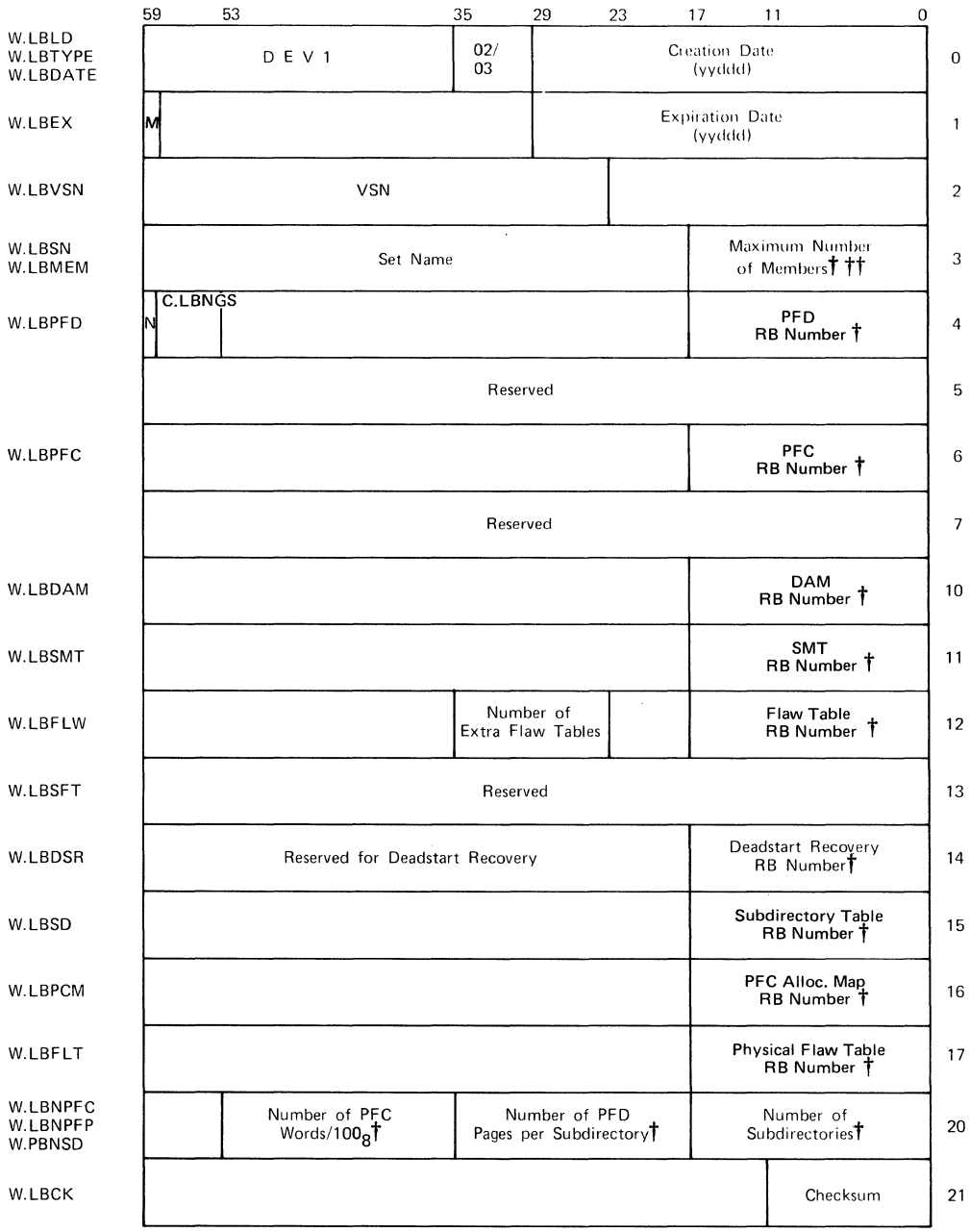
DSD-INTERCOM COMMUNICATION THROUGH STATION (MSG)

MSG is used by the CYBER Station in a multimainframe configuration. This table, residing within the station field length, serves as a communication area for DSD, INTERCOM, and the station.



Word	Bit	Field	Description
0	59	R	Command or display rejected by linked mainframe.
	47-36	SCT	Station sequence count.
	23-12	REJCODE	Type of error.
	11-0	MSGBUFF	Address reply to the message (relative to RA of station). Replay is available when SCT equals ECT.
1	17-12	PIDC	Identifying character of PID, to which message is sent.
	11-0	ECT	External processor (DSD or INTERCOM) sequence count. Message should be stored before the word containing PIDC and ECT (which should be incremented by one).
2		MSG	Message request from external processor to be sent to designated mainframe. MSG starts in word 2 and can be up to three words long.

DEVICE SET LABEL

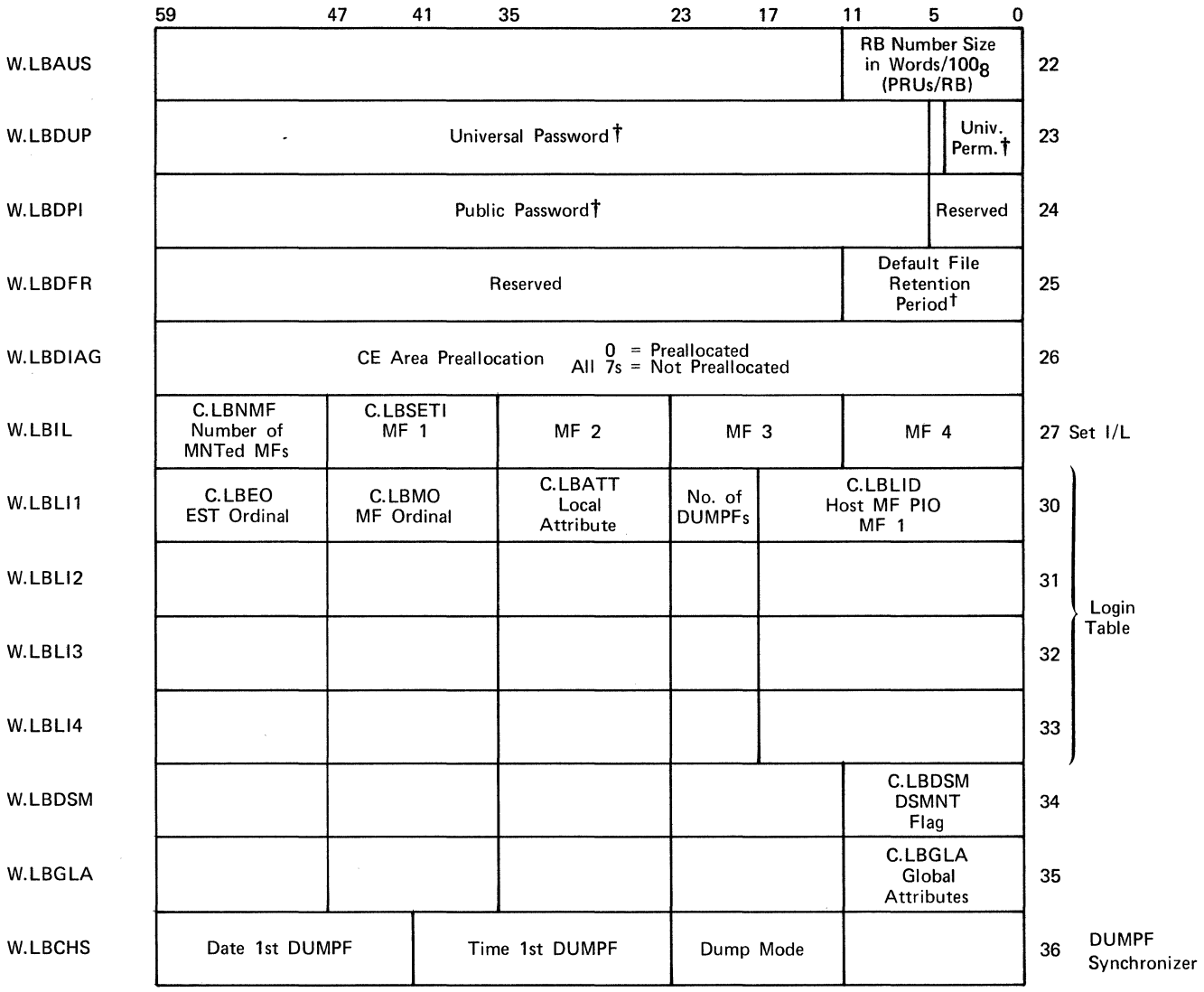


M If set, system of PSR level 430 or higher initialized pack. At this time, PB size changed from 56 to 112.

N Gap sector flag; if set, no gap sectors.

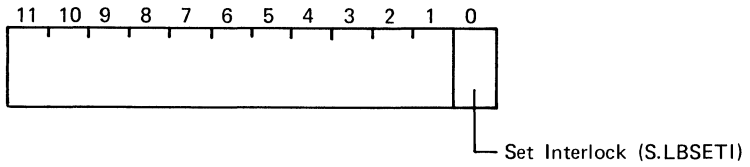
† Field contains nonzero value only on master device.

†† The length of the SMT is twice the value contained in this field.

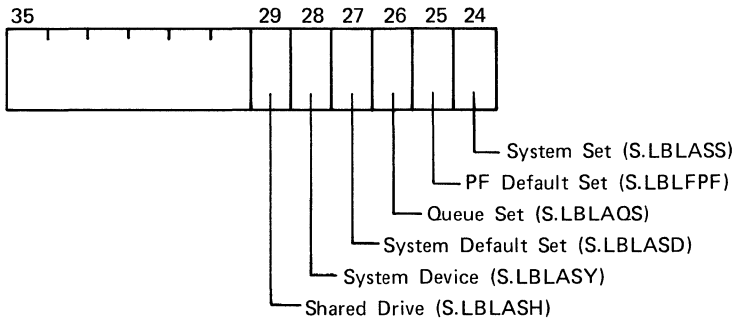


†Field contains nonzero value only on master device.

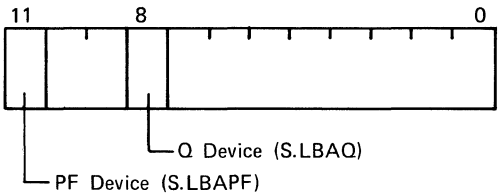
Detail: Set Interlock Byte in W.LBIL (Word 27)



Detail: Local Attributes in W.LBLI1-W.LBLI4 (Words 30-33)



Detail: Global Attributes in W.LBGLA (Word 35)



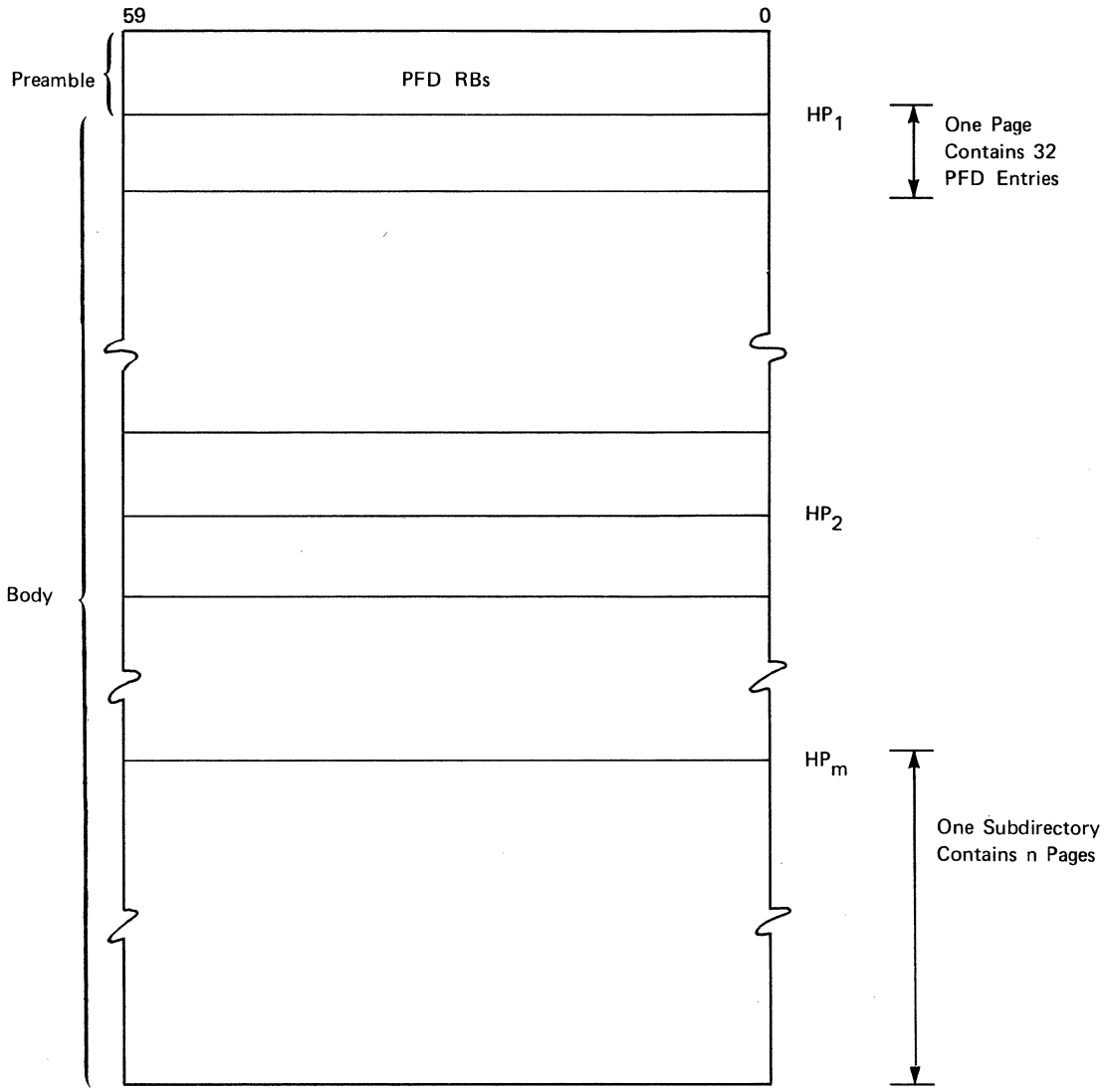
Symbol	Word	Bit	Description
W.LBLD	0	59-36	DEV1 in display code.
W.LBTYPE	0	35-30	02 6000 HT
			03 6000 FT
W.LBDATE	0	29-0	Creation date in the form yyddd (display code).
W.LBEX	1	29-0	Expiration date in the form yyddd (display code).
W.LBVSN	2	59-24	VSN right-justified, display zero-filled (display code).

Symbol	Word	Bit	Description
W.LBSN	3	59-18	Set name left-justified, binary zero-filled (display code).
W.LBMEM	3	17-0	Maximum number of members (binary).
W.LBPFD	4	59	Gap sector flag; if set, no gap sectors.
		17-0	PFD RB number (binary).
-	5	59-0	Reserved.
W.LBPFC	6	17-0	PFC RB number (binary).
-	7	59-0	Reserved.
W.LBDAM	10	17-0	DAM RB number (binary).
W.LBSMT	11	17-0	SMT RB number (binary).
W.LBFLW	12	35-24	Number of extra flaw tables.
		17-0	Flaw table RB number (binary).
-	13	59-0	Reserved.
W.LBDSR	14	17-0	Deadstart recovery RB number.
W.LBSD	15	17-0	Subdirectory table RB number (binary).
W.LBPCM	16	17-0	PFC allocation map RBs (binary).
W.LBFLT	17	17-0	Physical flaw table RB number (binary).
W.LBNPFC	20	53-36	Number of PFC words/100_8.
W.LBNPFP	20	35-18	Number of PFD pages per subdirectory (binary).
W.LBNSD	20	17-0	Number of subdirectories (binary).
W.LBCK	21	11-0	Checksum of this label PRU.
W.LBAUS	22	11-0	RB size.
W.LBDUP	23	59-6	Universal password (display code).
		3-0	Universal permission bits.

Bit	Permission
3	Control (C).
2	Modify (M).
1	Extend (E).
0	Read (R).

Symbol	Word	Bit	Description
W.LBDPI	24	59-6	Public password (display code).
		5-0	Reserved.
W.LBDFR	25	59-12	Reserved.
		11-0	Default permanent file retention period (binary).
W.LBDIAG	26	59-0	CE area preallocation.
W.LBIL	27	59-48	Number of mainframes sharing mass storage.
		47-36	Mainframe 1 set interlock (refer to detail).
		35-24	Mainframe 2 set interlock.
		23-12	Mainframe 3 set interlock.
		11-0	Mainframe 4 set interlock.
W.LBLI1 through W.LBLI4	30-33	59-48	Equipment status table ordinal.
		47-36	Mainframe ordinal.
		35-24	Local attributes (refer to detail).
		23-18	Number of DUMPFs executing.
		17-0	Mainframe identification.
W.LBDSM	34	11-0	DMNT flag.
W.LBGLA	35	11-0	Global attribute (refer to detail).
W.LBCHS	36	59-42	Date of first DUMPF.
		41-24	Time of first DUMPF.
		23-12	Dump mode.

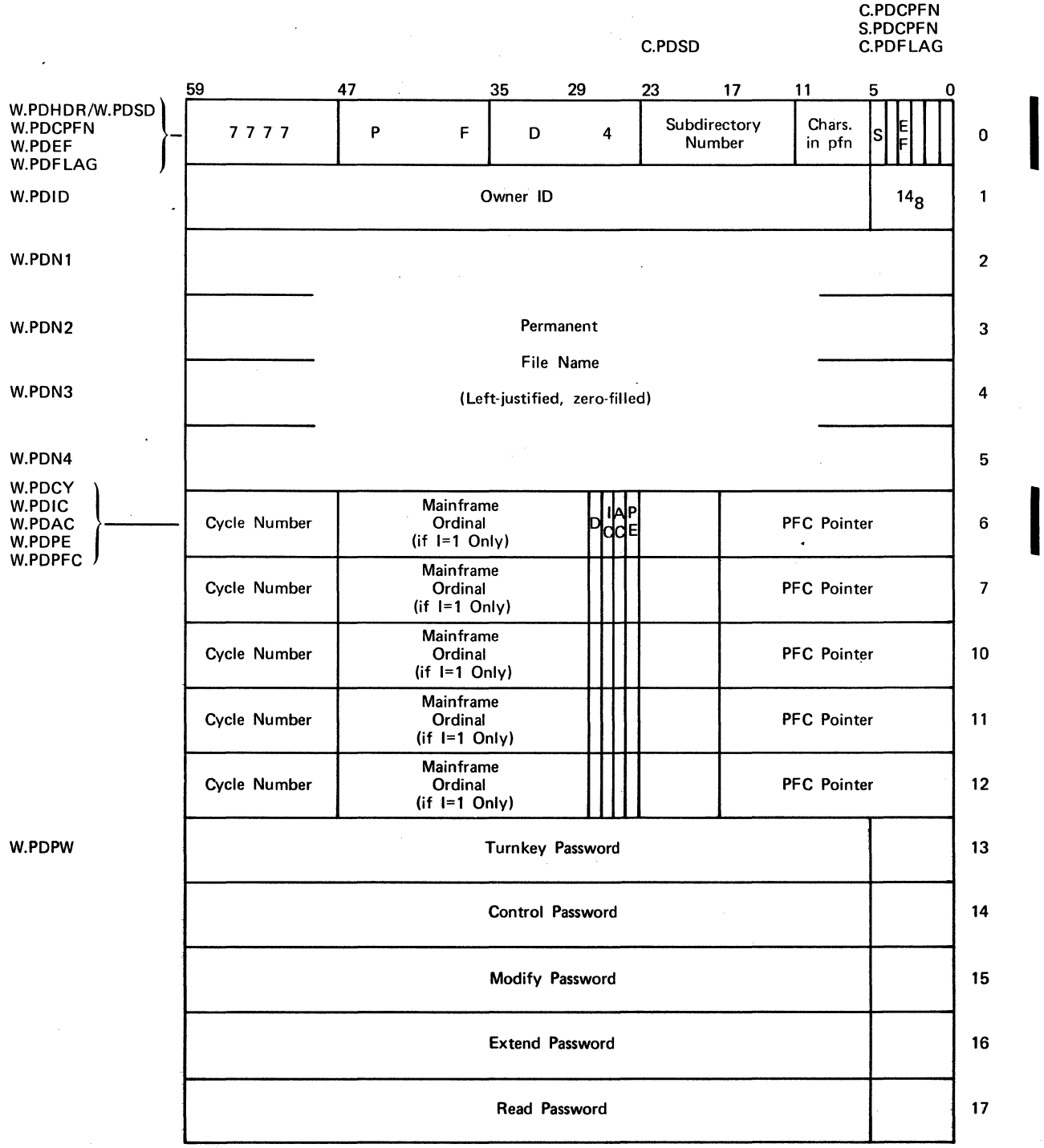
PERMANENT FILE DIRECTORY (PFD) OVERVIEW



HP_1	ID hash point.
m	Number of ID hash points.
n	Number of pages between ID hash points; must be an integral power of 2.

m * n = number of pages in the PFD body.

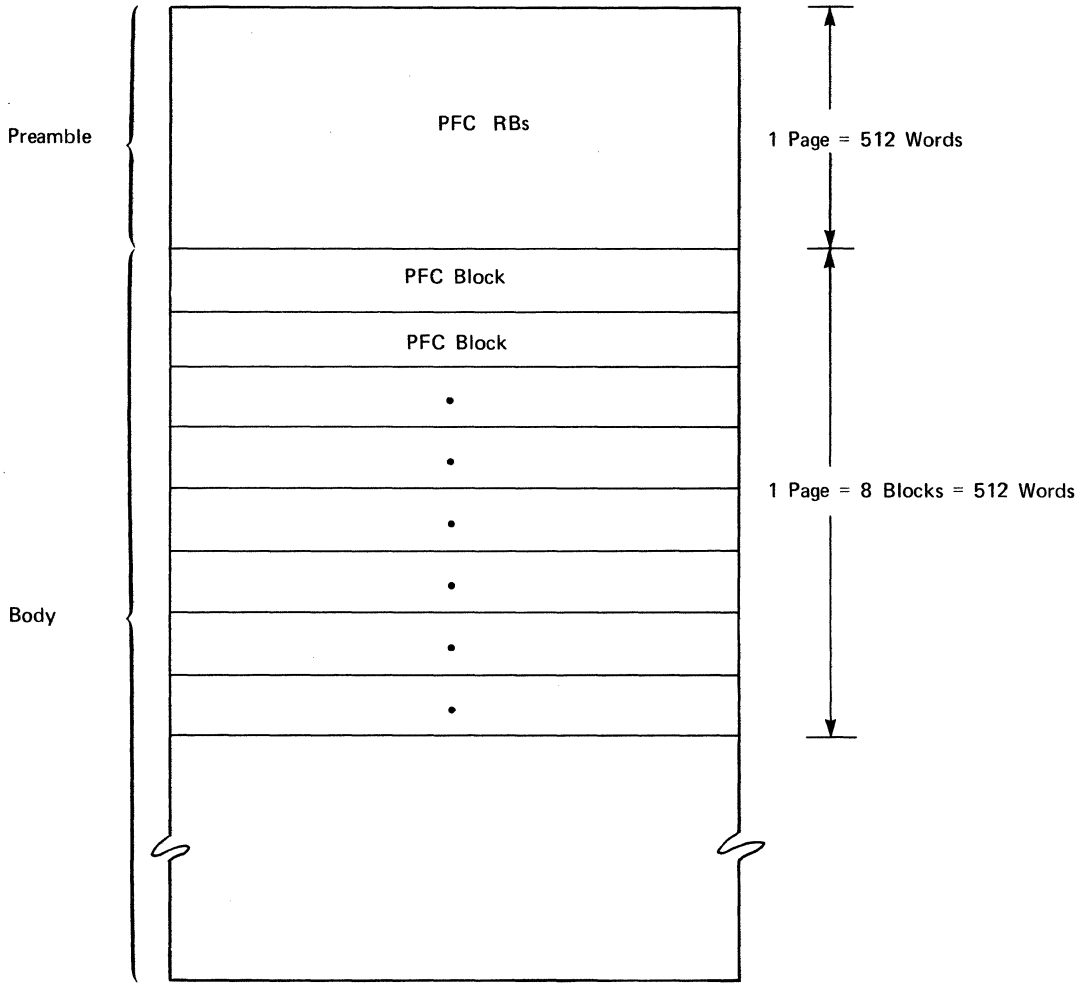
PFD ENTRY


C.PDCPFN
S.PDCPFN
C.PDFLAG
C.PDSD
59
47
35
29
23
17
11
5
0
W.PDHDR/W.PDSD
W.PDCPFN
W.PDEF
W.PDFLAG
7 7 7 7
P
F
D
4
Subdirectory Number
Chars. in pfn
S
EF
0
W.PDID
Owner ID
14_8
1
W.PDN1
2
W.PDN2
Permanent
3
File Name
W.PDN3
(Left-justified, zero-filled)
4
W.PDN4
5
W.PDCY
W.PDIC
W.PDAC
W.PDPE
W.PDPFC
Cycle Number
Mainframe Ordinal (if I=1 Only)
D
IAC
PCE
PFC Pointer
6
7
10
11
12
W.PDPW
Turnkey Password
13
Control Password
14
Modify Password
15
Extend Password
16
Read Password
17

Symbol	Word	Bit	Description
W.PDHDR	0	59-48	7's in binary.
		47-36	PF in variable code.
		35-24	DR in variable code.
W.PDSD	0	23-12	Subdirectory number to which the ID was hashed (binary).
W.PDCPFN	0	11-6	Number of characters in the permanent file name (binary).
W.PDFLAG	0	5	End of subdirectory flag.
W.PDEF	0	3	0 Entry is free.
			1 Entry is in use.
W.PDID	1	59-6	Owner ID, right-justified, blank-filled (display code).
		5-0	14_8.
W.PDN1	2	59-0	Permanent file name, left-justified, trailing binary zeros with no nested bytes of zeros.
W.PDN2	3	59-0	Same as W.PDN1.
W.PDN3	4	59-0	Same as W.PDN1.
W.PDN4	5	59-0	Same as W.PDN1.
W.PDCY	6-12	59-48	Cycle number (binary).
W.PDFMO	6-12	47-36	Mainframe ordinal; for incomplete cycle only.
W.PDCY	6-12	27	0 Cycle is not dumped.
			1 Cycle is dumped.

Symbol	Word	Bit	Description
W.PDIC	6-12	26	0 Cycle is complete.
			1 Cycle is incomplete.
W.PDAC	6-12	25	0 Cycle is not archived.
			1 Cycle is archived.
W.PDPE	6-12	24	0 Cycle has no parity errors.
			1 Cycle has parity errors.
W.PDPFC	6-12	17-0	PFC pointer as a 100_8 word offset (binary).
W.PDPW	13	59-6	Password parameter for turnkey permission, right-justified, binary zero-filled (display code).
	14	59-6	Password parameter for control permission, right-justified, binary zero-filled (display code).
	15	59-6	Password parameter for modify permission, right-justified, binary zero-filled (display code).
	16	59-6	Password parameter for extend permission, right-justified, binary zero-filled (display code).
	17	59-6	Password parameter for read permission, right-justified, binary zero-filled (display code).

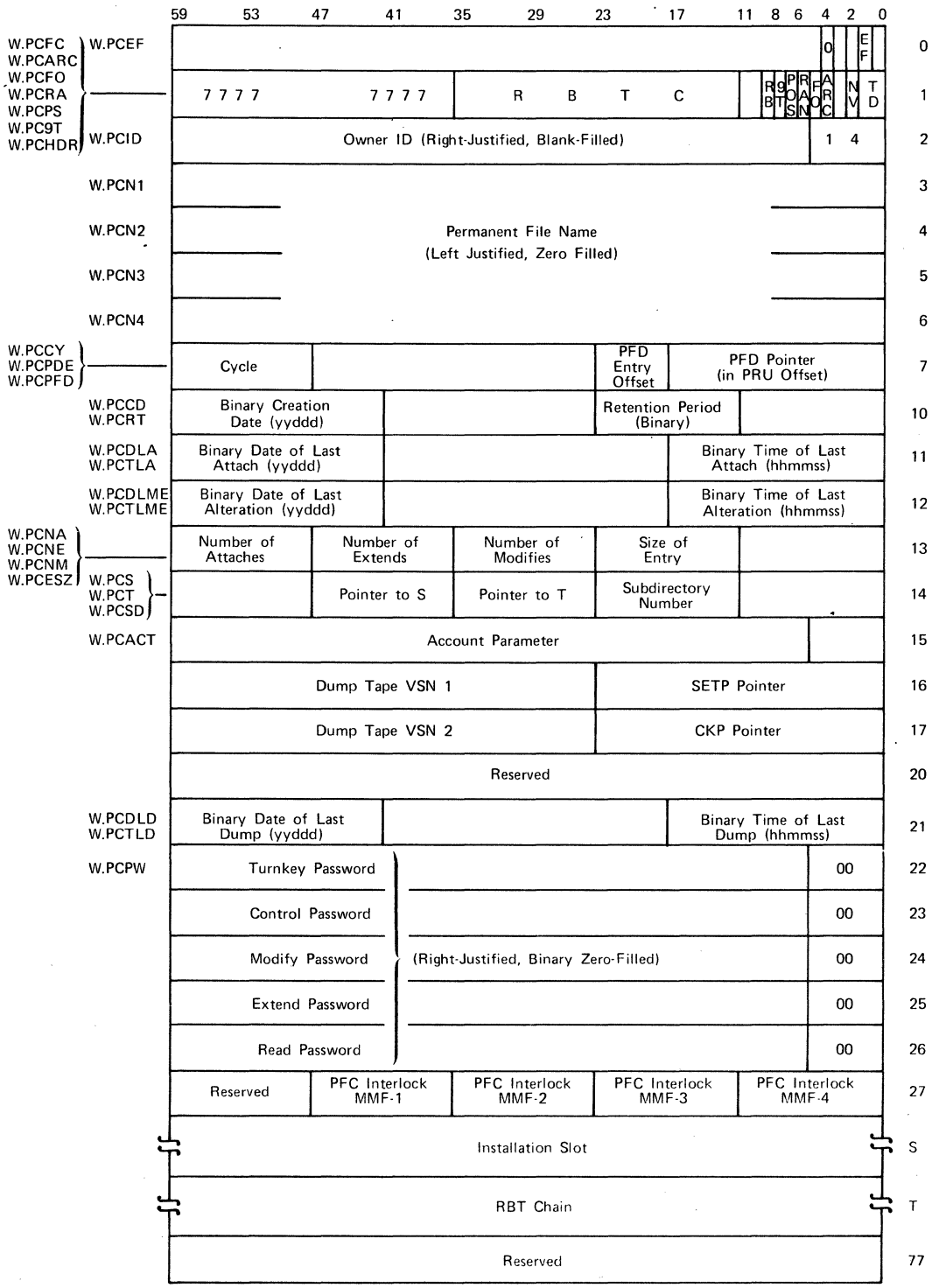
PERMANENT FILE CATALOG (PFC) OVERVIEW†



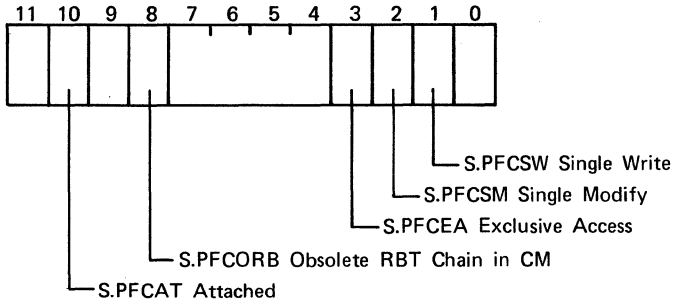
Each PFC entry occupies one or more consecutive PFC blocks.

† The PFC was formerly called the record block table catalog (RBTC).

PFC ENTRY FOR PERMANENT FILES



Detail: PFC Interlock Byte



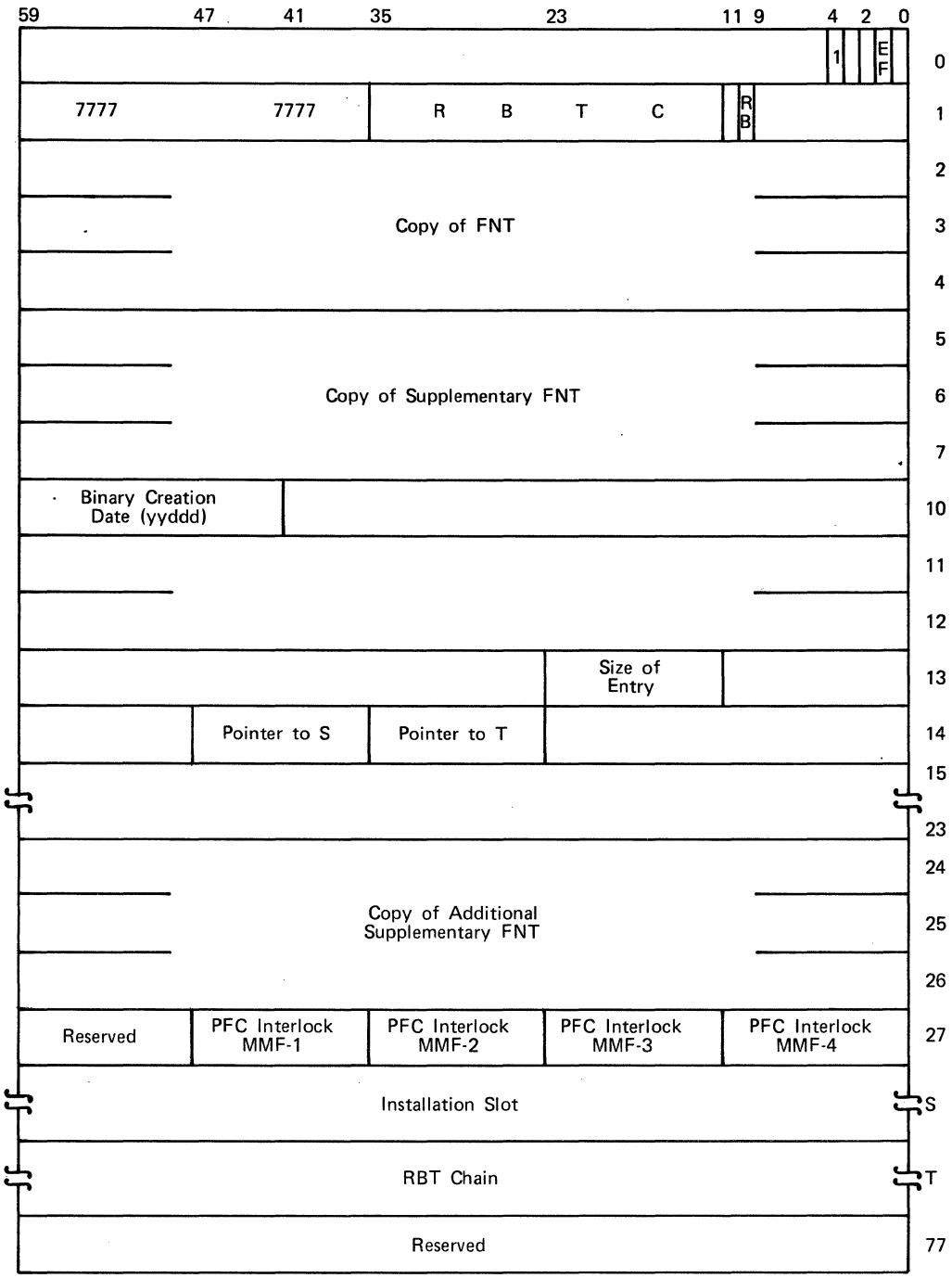
Symbol	Word	Bit		Description
W.PCEF	0	4	0	Permanent file entry.
			1	Queue file entry.
		1	0	Entry free.
			1	Entry in use.
W.PCHDR	1	35-12	RBTC in display code.	
W.PCRB	1	9	0	No RB conflicts.
			1	RB conflicts exist on this file.
W.PC9T	1	8	0	Seven-track tape.
			1	Nine-track tape.
W.PCPS	1	7	0	File is not positioned.
			1	File is positioned.
W.PCRA	1	6	0	File is not random.
			1	File is random.
W.PCPO	1	5	0	File is not an SAAM file.
			1	File is an SAAM file (file organization IS/DA/AK).

Symbol	Word	Bit	Description
W.PCARC	1	4	0 Cycle is not archived.
			1 Cycle is archived.
W.PCFC	1	2	0 Dumped.
			1 New version.
		1-0	Tape density, seven-track.
			00 556 cpi.
			01 200 cpi.
			10 800 cpi.
			11 Default.
			Tape density, nine-track.
			00 Default
			01 6250 cpi.
			10 800 cpi.
			11 1600 cpi.
W.PCID	2	59-6	Owner ID.
		5-9	14_8.
W.PCN1	3	59-0	Permanent file name, left-justified, zero-filled (display code).
W.PCN2	4	59-0	Same as W.PCN1.
W.PCN3	5	59-0	Same as W.PCN1.
W.PCN4	6	59-0	Same as W.PCN1.
W.PCCY	7	59-48	Cycle number (binary).
W.PCPDE	7	23-18	PFD entry number.
W.PCPFD	7	17-0	PFD pointer as a sector offset (binary).
W.PCCD	10	59-42	Creation date in the form yyddd (binary).
W.PCRT	10	23-12	Retention period (binary).
W.PCDLA	11	59-42	Date of last attach in form yyddd (binary).
W.PCTLA	11	17-0	Time of the last attach in the form hhmmss (binary).

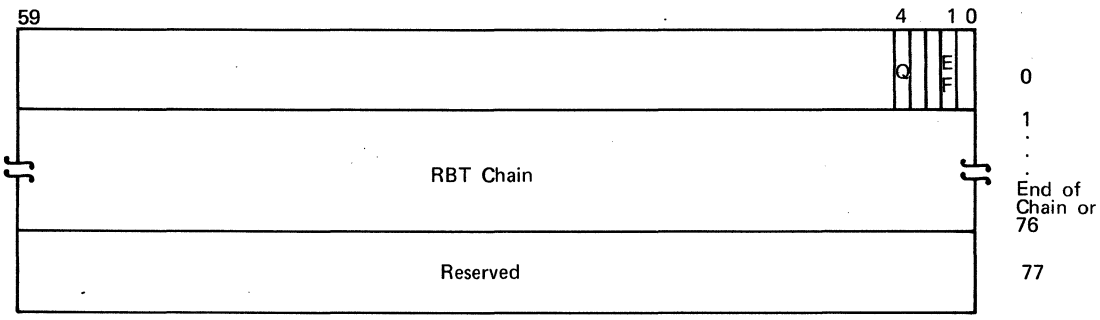
Symbol	Word	Bit	Description
W.PCDLME	12	59-42	Date of the last alter in the form yyddd (binary).
W.PCTLME	12	17-0	Time of the last alter in the form hhmmss (binary).
W.PCNA	13	59-48	Number of attaches (binary).
W.PCNE	13	47-36	Number of extends (binary).
W.PCNM	13	35-24	Number of modifies (binary).
W.PCESZ	13	23-12	Size of the entry in binary number of words.
W.PCS	14	47-36	Number of words from word 13 to the user area.
W.PCT	14	35-24	Number of words between word 13 and start of RBT chain.
W.PCSD	14	23-12	Subdirectory number (binary).
W.PCACT	15	15	Account parameter code, left-justified, blank-filled (display code).
—	27		PFC interlock for MMF configuration; each byte contains the following flags.

Bit	Description
10	File attached (S.PFCAT).
8	Obsolete RBT chain in CM (S.PFCORB).
3	Exclusive access (S.PFCEA).
2	Single modify (S.PFCSM).
1	Single write (S.PFCSW).

PFC ENTRY FOR I/O QUEUES



PFC ENTRY OVERFLOW



Q=0 Permanent file PFC entry.

Q=1 Queue file PFC entry.

DEVICE SET RBT CHAINS IN PFC

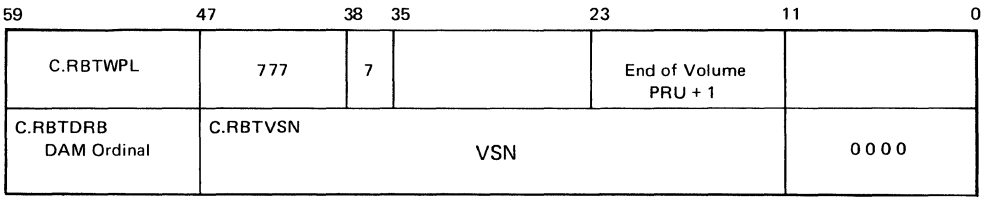
FIRST RBT WORD PAIR

59 – 47	38	35 – 29	23 – 11	0
C.RBTWPL	7	C.RBTAL Device Type / Alloc. Type	C.RBTPRU Last PRU + 1	C.RBTBIT Flags
C.RBTDRB DAM Ordinal	C.RBTVSN VSN			RB_7

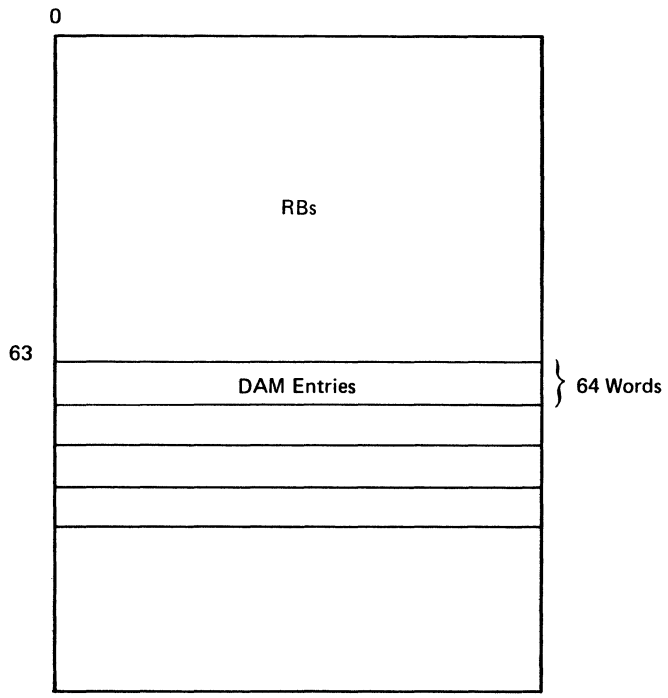
OTHER WORD PAIRS EXCEPT OVERFLOW

59 – 47	38	35 – 23	11 – 0
C.RBTWPL	0	RB_0	RB_1	RB_2
RB_3	RB_4	RB_5	RB_6	RB_7

OVERFLOW WORD PAIRS



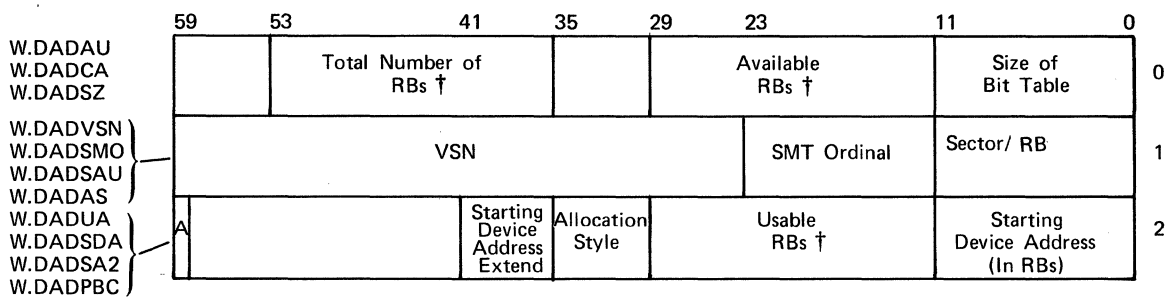
DEVICE ALLOCATION MAP (DAM) OVERVIEW



Every DAM starts on a sector boundary. If the entry is greater than 64 words, it overflows to the next PRU or PRUs.

The DAM for an 844 or 885 device can contain a maximum of 50_{10} entries; that is, the entire DAM requires one RB.

DAM HEADER

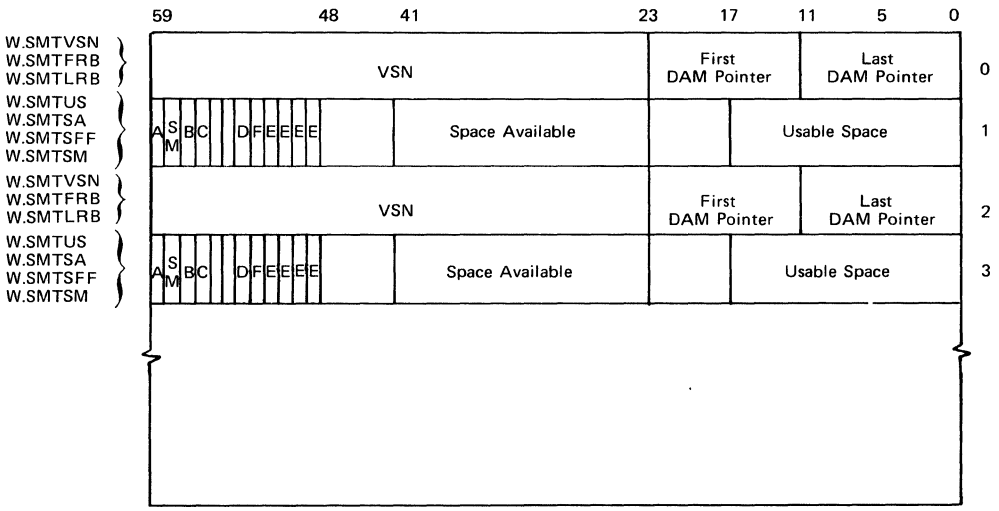


A RB size/PB size compatibility bit.

Symbol	Word	Bit	Description
W.DADAU	0	53-36	Total number of RBs.
W.DADCA	0	29-12	Currently available RBs.
W.DADSZ	0	11-0	Size of bit table in words.
W.DADVSN	1	59-24	VSN.
W.DADSMO	1	23-12	Set member ordinal.
W.DADSAU	1	11-0	Sectors per RB.
W.DADSA2	2	41-36	Extension of starting device address (W.DADSDA).
W.DADPBC	2	59	If set, device was created on a system with PB size of $112_{10}/114_{10}$ for 844 devices.
W.DADAS	2	35-30	Allocation style.
W.DADUA	2	29-12	Usable RBs equal total RBs minus flaws minus system RBs.
W.DADSDA	2	11-0	Starting device address in PBs.
W.DADRCA	53	7-0	Preventive maintenance (844-21 and 844-41).
W.DADRCB	54	59-0	Reserved cylinders.

† Values exceeding 4095_{10} are not supported.

SET MEMBER TABLE (MASS STORAGE)



File Name	Description
A	PF device (S.SMTSFF).
SM	Mounted flag (S.STMTSM).
B	Recover flag.
C	Queue device (S.SMTSFQ).
D	Preallocated for C.E. diagnostics (S.SMTPR).
E	Mounted on mainframe ordinal 1 through 4 (bit 51 equals 1, bit 50 equals 2, bit 49 equals 3, bit 48 equals 4) (S.SMTMF).
F	Operator requested dismount (S.SMTOD).

The length of the SMT is two times the maximum number of members as shown in word 3 of the master device label. A two-word entry will exist for each member in a set.

Zero entries can be intermixed with VSN entries for member devices.

Symbol	Word	Bit	Description
W.SMTVSN	Even	59-24	VSN of the pack, right-justified, display zero-filled.
W.SMTFRB	Even	23-12	First DAM pointer as a sector offset (binary).
W.SMTLRB	Even	11-0	Last DAM pointer as a sector offset (binary).

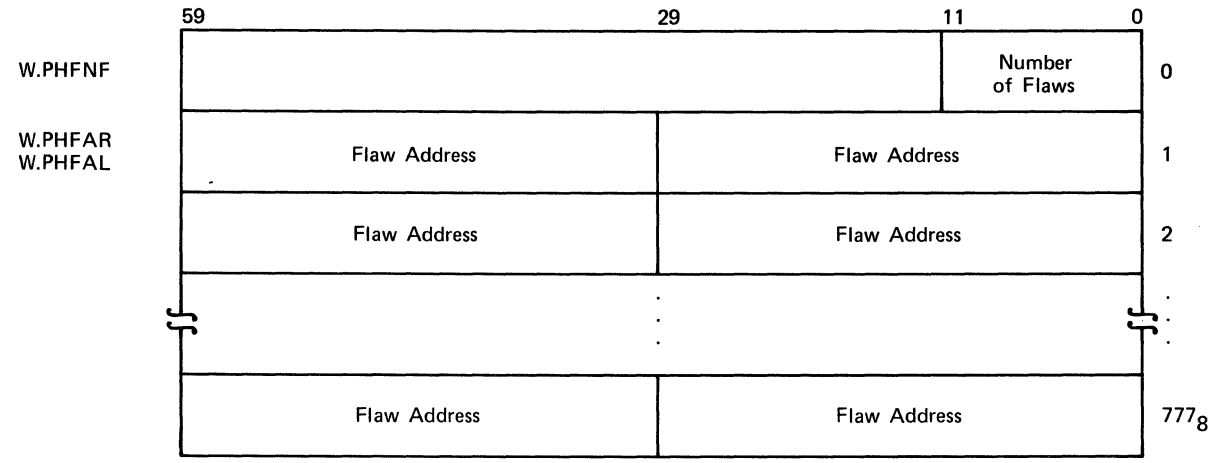
Symbol	Word	Bit		Description
W.SMTSFF	Odd	59	0	Permanent files do not reside on this device.
			1	Permanent files may reside on this device.
W.SMTSM	Odd	58	0	Device is not mounted.
			1	Device is mounted.
		57		Recover interlock (master pack only).
		56	0	Queue files do not reside on this device.
			1	Queue files may reside on this device.
		53	0	Device is not preallocated.
			1	Device is preallocated.
		52	0	Operator dismount not outstanding.
			1	Operator dismount outstanding.
		51-48	0	Not mounted by mainframe with this mainframe ordinal.
			1	Mounted by mainframe with this mainframe ordinal.
				(Bit 51 is mainframe ordinal 1, bit 50 is mainframe ordinal 2, and so on.)
W.SMTSA	Odd	41-24		Total allocatable space available on this device (binary) is number of RBs currently not in use on the device.
W.SMTUS	Odd	17-0		Total number of RBs minus number of flaws and minus number of system table RBs.

LOGICAL FLAW TABLE (LFT)

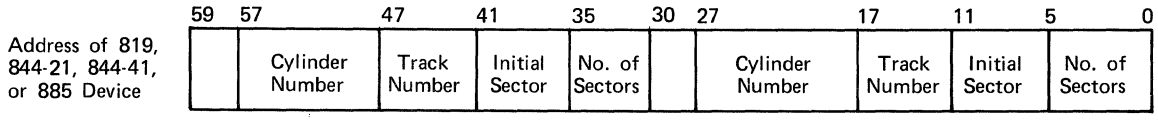
The logical flaw table has the same format as the DAM, but only bits corresponding to flawed sectors (within allocation unit) are set.

PHYSICAL FLAW TABLE

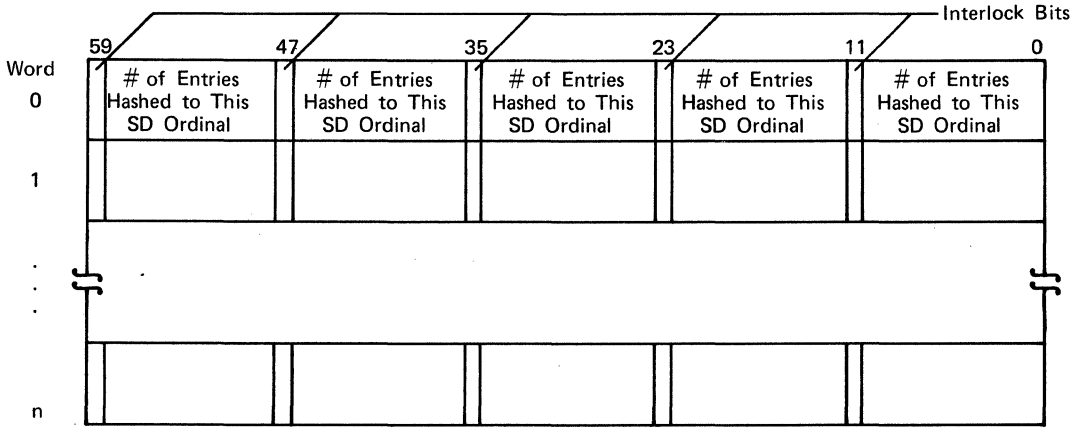
819, 844-21, 844-41, AND 885 DEVICES



PHYSICAL FLAW TABLE ENTRIES



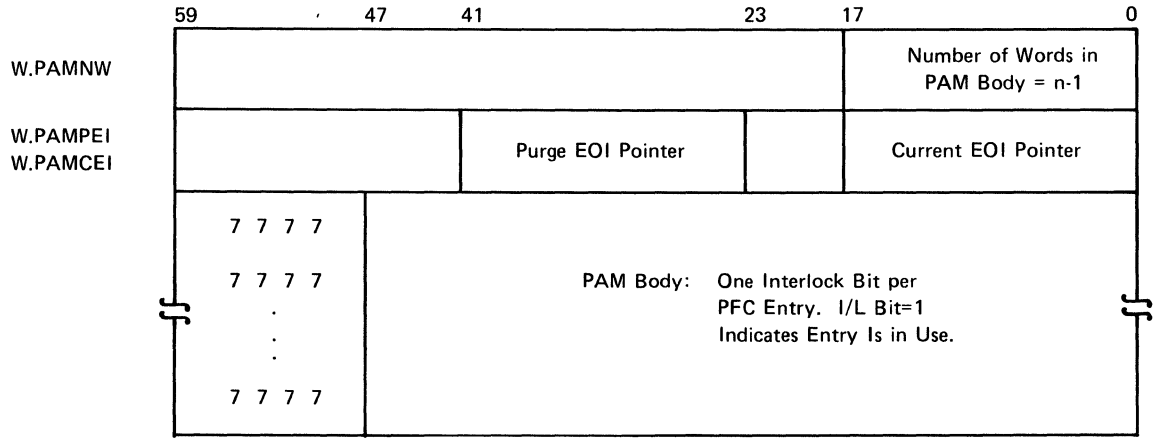
SUBDIRECTORY TABLE



Word	Bit	Description
0-n	59,47,35,23,11	Interlock bit; always zero on RMS copy of the table (reserved for 7000 use only).
	58-48,46-36,34-24,22-12,10-0	Number of entries that have hashed to this subdirectory ordinal (binary).

PFC ALLOCATION MATRIX (PAM)

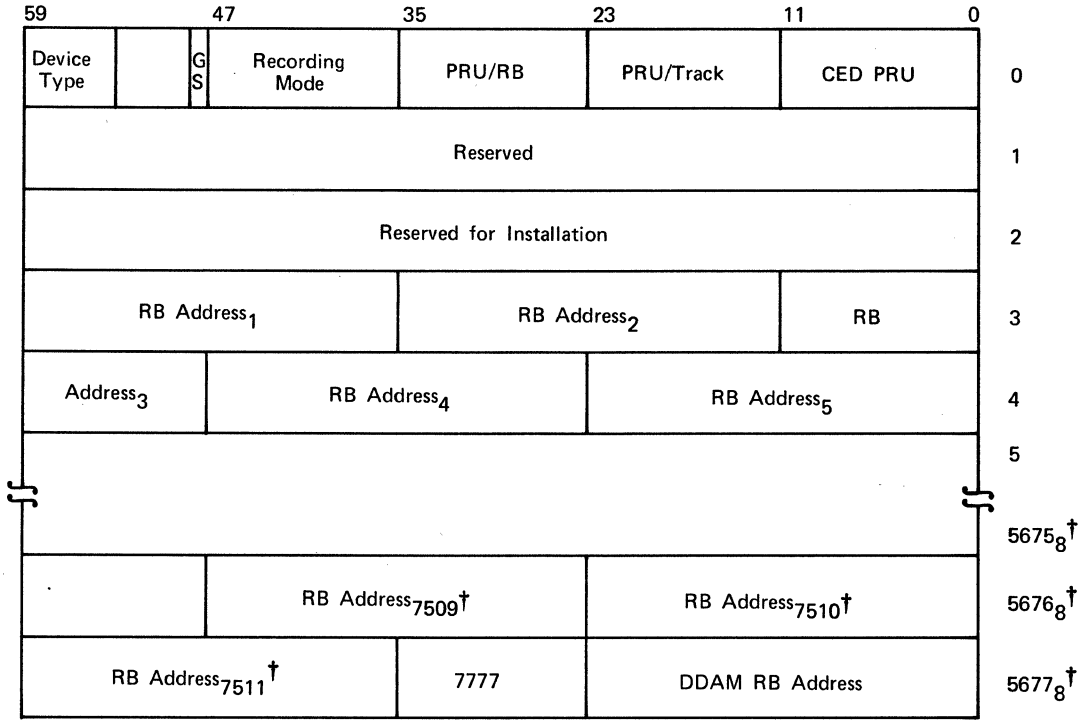
(Reserved for SCOPE 2 processing.)



Symbol	Word	Bit	Description
W.PAMNW	0	17-0	Number of words in PAM body.
W.PAMPEI	1	41-24	Pointer to empty 100_8 word block created by purge as a PFC sector offset.
W.PAMCEI	1	17-0	PFC sector offset to first 100_8 word block not in use after the last 100_8 word block in use.

The PAM body represents one interlock bit per PFC entry. A PAM bit set implies that an entry is in use. Only bits 47 through 0 of the body are used.

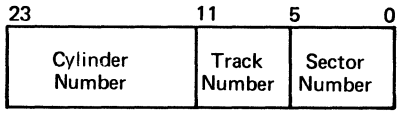
DDAM FILE FORMAT



Word	Bit	Description
0	59-54	Device type.
		13 844-21 disk drive.
		14 844-41 disk drive.
		17 885 disk drive.
	53-49	Not used.
	48	Gap sector flag.
		1 No gap sectors.
	47-36	Recording mode.
		2 Half-track.
		1 Full-track.
	35-24	RB size in PRUs.

†Assumes 57 PRUs/RB.

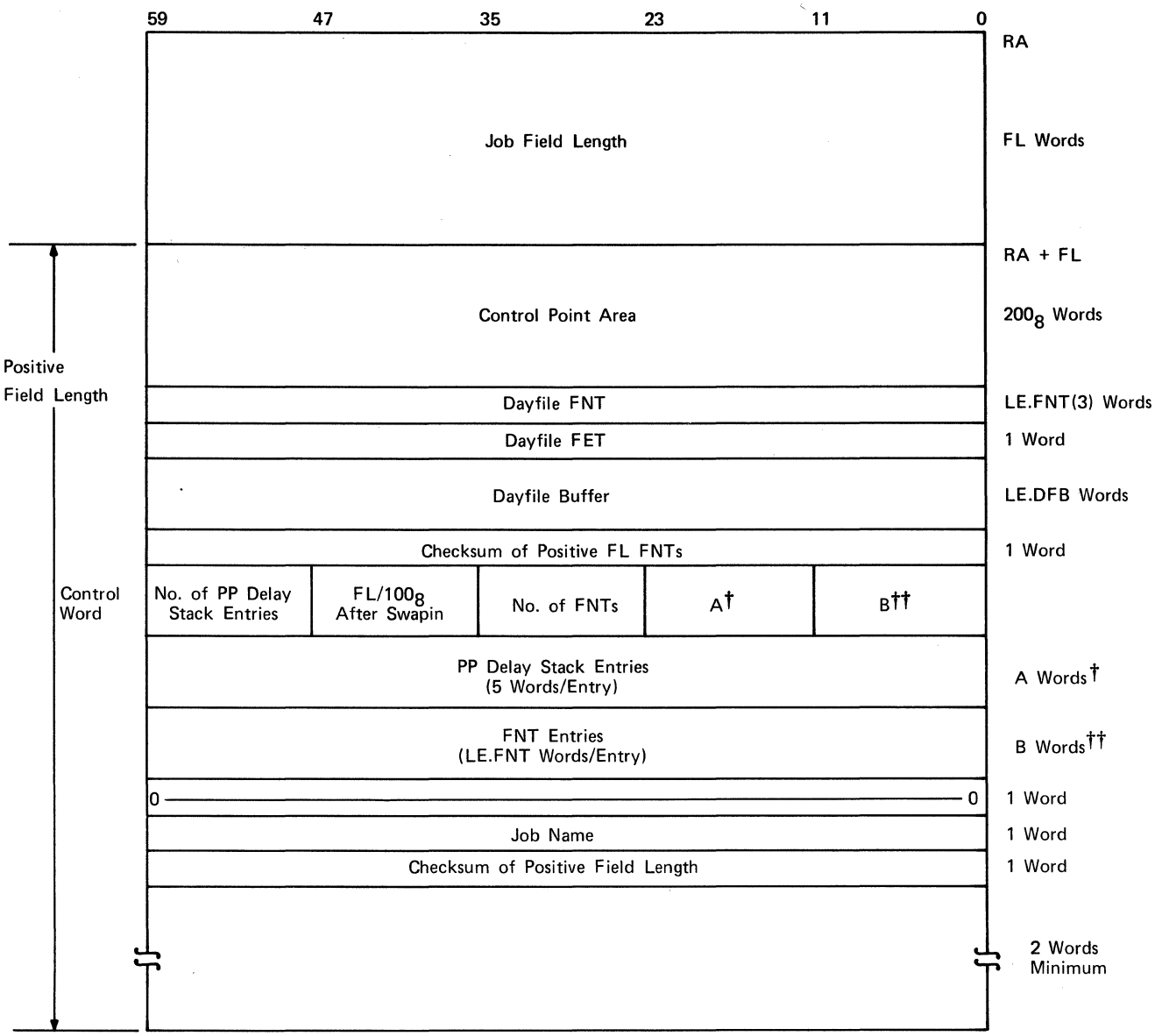
Word	Bit	Description
	23-12	Device track size in PRUs.
	11-0	Number of first PRU of CED in the first RB.
3 through 5677_8†		Physical address of the first PRU of the RB. The format of this address is:



A value of 7777_8 in the cycle field indicates the end of the DDAM RB. If a value of 7777_8 is found in the next byte, it indicates the end of the DDAM file. Otherwise, the next two bytes contain the physical address of the first PRU of the next RB of the DDAM file in the same format. The first RB address of the second and succeeding RBs of the DDAM file is in word 0, bits 47 through 24.

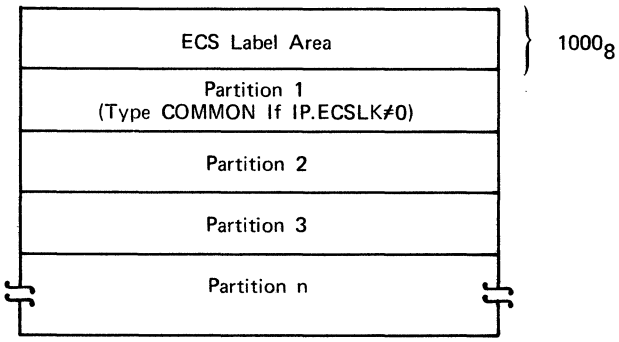
†Assumes 57 PRUs/RB.

SWAP FILE FORMAT



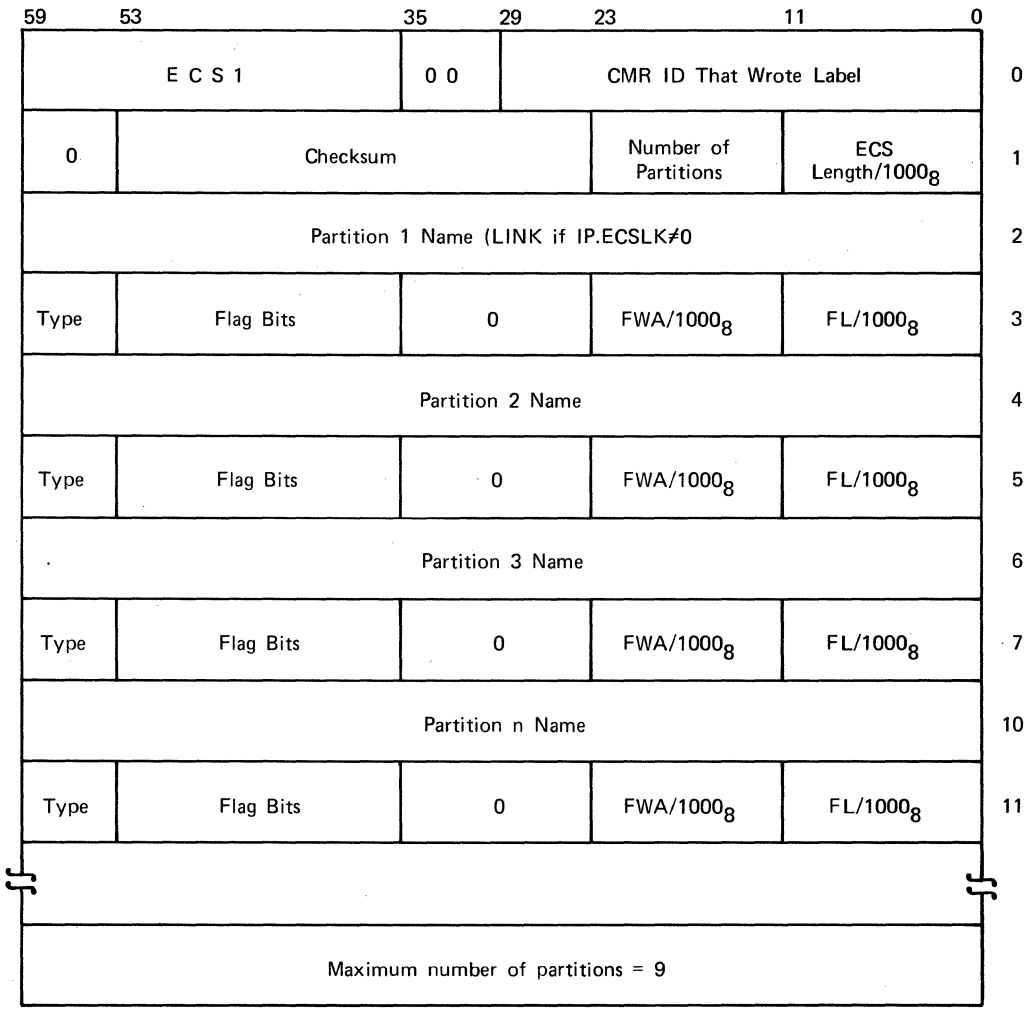
EXTENDED CORE STORAGE TABLES E

OVERALL ECS FORMAT



ECS label is written to one of the areas starting at 120_8, 230_8, 340_8, 450_8, 560_8, or 670_8.

ECS LABEL



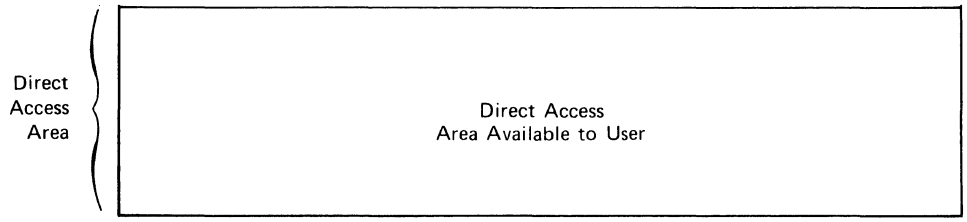
Type is one of the following:

1_8 Direct access area.

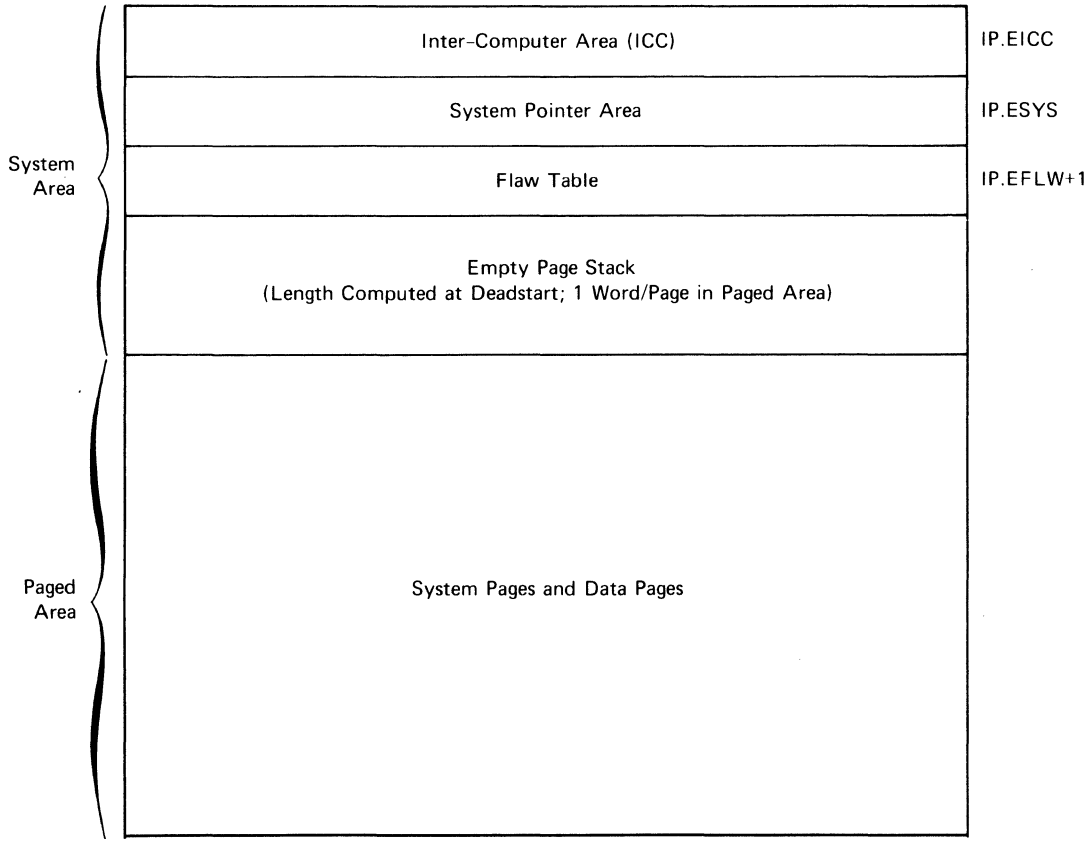
2_8 Allocatable device area (system and paged area).

3_8 COMMON area.

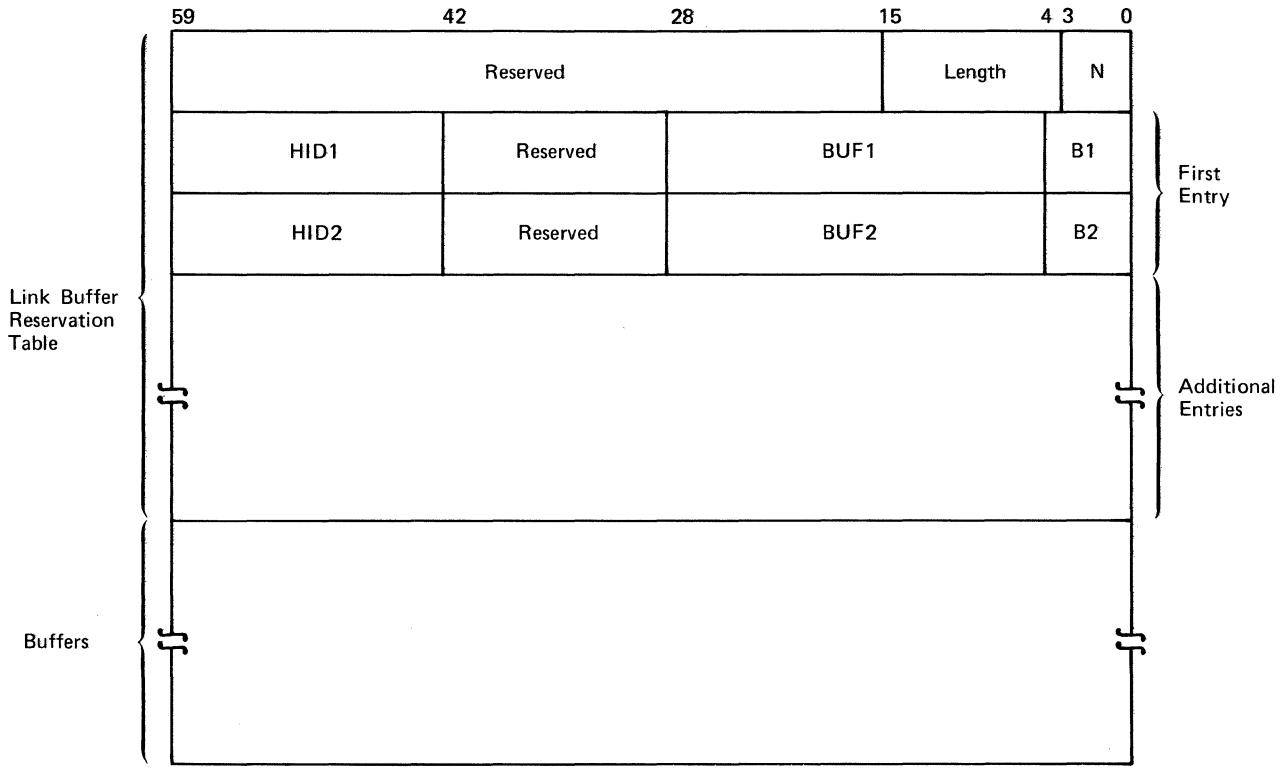
DIRECT ACCESS AREA (TYPE 1 PARTITION)



ALLOCATABLE DEVICE AREA (TYPE 2 PARTITION)

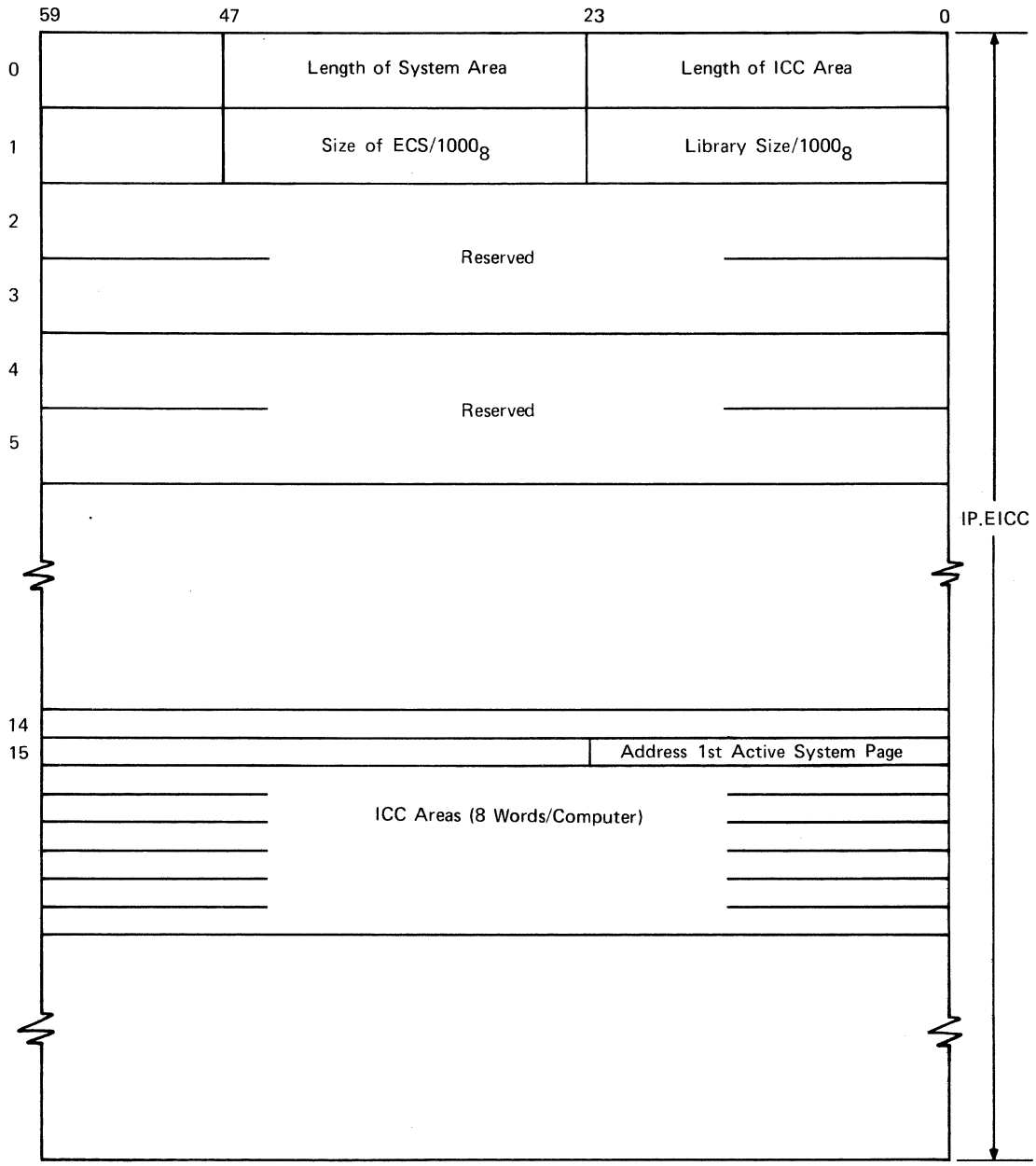


COMMON AREA (TYPE 3 PARTITION)

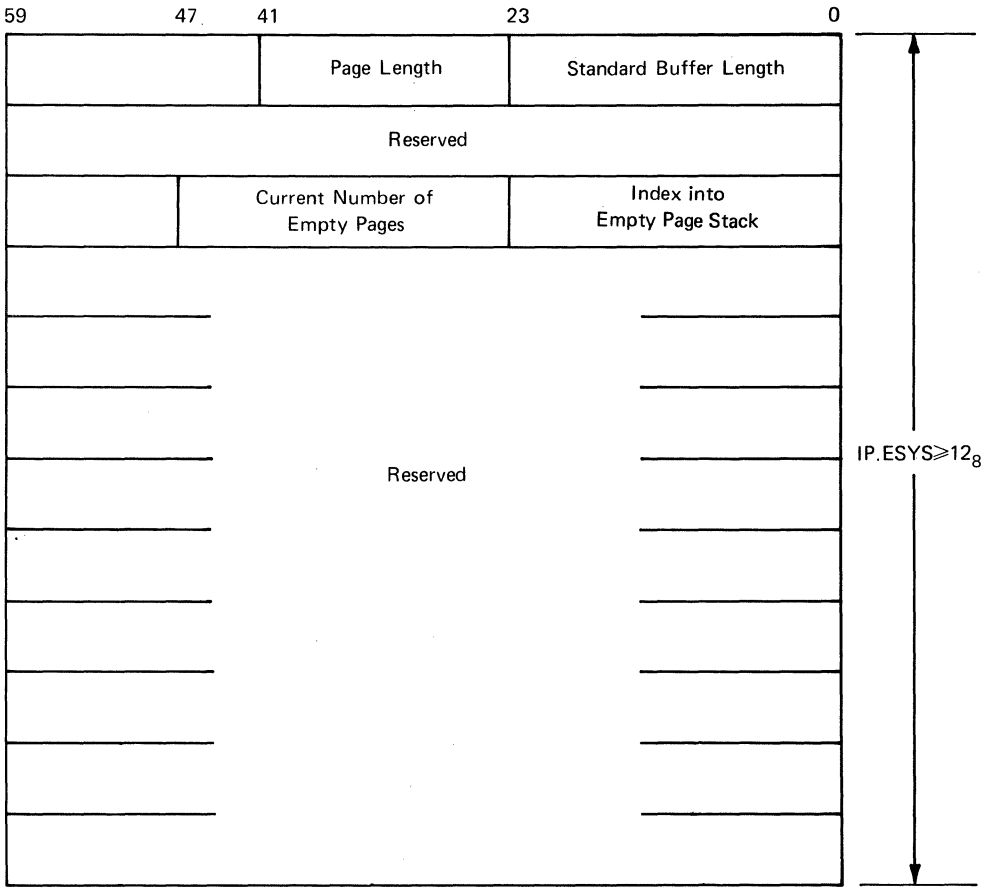


Field Name	Description
Length	Length of each buffer.
N	IP.LNKBF; number of entries in table.
HID1	Mainframe ID of first mainframe to reserve a buffer for this link.
HID2	Mainframe ID of second mainframe to reserve a buffer for this link.
BUF1	FWA of receiving buffer for first mainframe (HID1) and send buffer for second mainframe (HID2).
BUF2	FWA of receiving buffer for second mainframe (HID2) and send buffer for first mainframe (HID1).
B1	TN.BUF1; link status bit in the ECS interlock register for BUF1.
B2	TN.BUF2; link status bit in the ECS interlock register for BUF2.

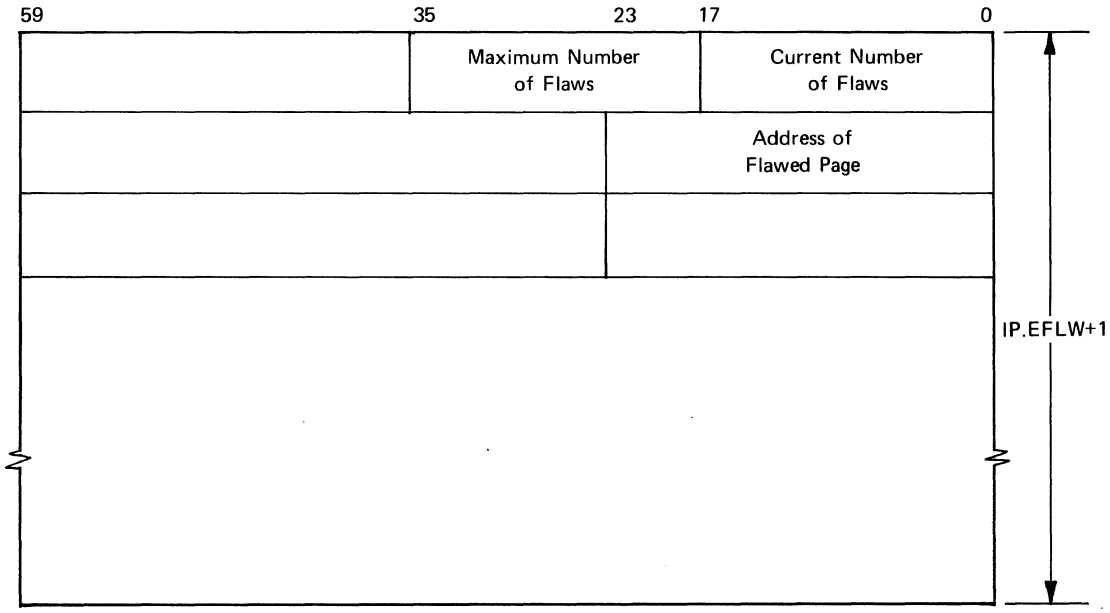
INTERCOMPUTER AREA (ICC)



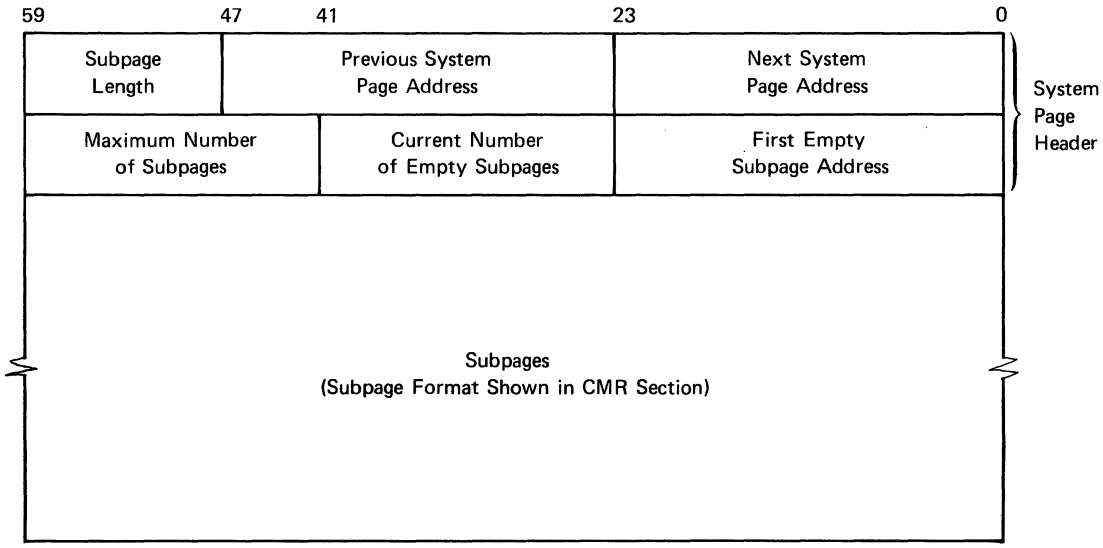
SYSTEM POINTER AREA



FLAW TABLE



SYSTEM PAGE



SEGMENTS IN AN ECS SYSTEM

Segment	Deck	Purpose
ADRB	SPM3	Add record blocks onto RBT.
BBJ	MMGR	Bring in a batch job.
CACT	ECLINK	Subroutine to change job activity count.
CALLDAM	SPM5	Initiate RBR update from DAM.
CBM	ECS	Circular buffer manager.
CBMEND	ECS	Circular buffer manager termination.
CCP	MMGR	Clean up control point area.
CHKPNT	SYSIDLE	Generate system checkpoint.
CHKSPAC	RACDAC	Subroutine to check if ECS space is available.
CLRCEM	ECS	Clear CEM working flag.
CMDATA	ECDATA	Common data areas.
CMRDIR	CMRDIR	Direct LDCMR in assigning segment residence.
CONTDAM	SPM5	Continue DAM processing.
CPCIO	CPCIO	Central processor I/O control.
CPECSM	CPSM	Storage move (ECS).
CPMTR	CPMTR	Identify reason for monitor mode execution.
CPPXT	ECS	ECS executive, central resident routines.
CPSCH	MMGR	Main loop - memory manager.
CPSM	CPSM	Storage move (CM).
CPSPM	SPM1	Stack processor manager.
CPSS	RESCH	Calculate system seconds - EX.SS.
CPSSF	SCPT	EX.SSF function processor.
CPUST	CPMTR	On/off CPU.
CP4ES	CPCIO	Enter stack request.
CSBS	ECLINK	Subroutine to check ECS link buffer space.
CSWP	MMGR	Call 1SI/1SO PP swappers.

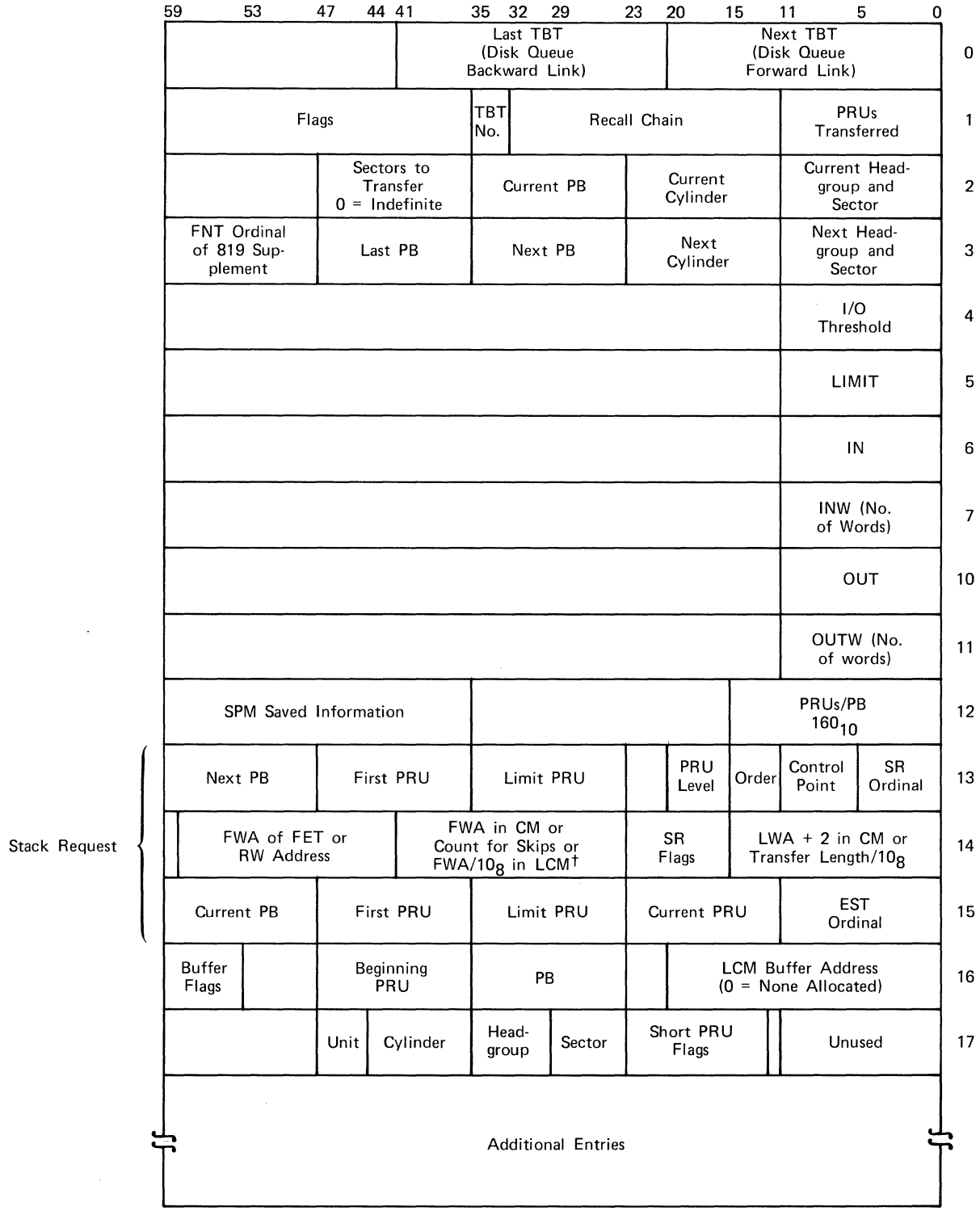
Segment	Deck	Purpose
DAC	RACDAC	Deallocate ECS overlay pages.
DBS	MMGR	Define best swap.
DEAL	ECS	Deallocate PRUs of data in ECS buffer.
DJD	MMGR	Subroutine to delink job descriptor.
DRVR	ECLINK	ECS link receive driver.
DRVRS	ECLINK	Subroutines for ECS link receive driver.
DRVS	ECLINK	ECS send driver.
DRVSS	ECLINK	Subroutines for ECS send driver.
ECPARTY	ECLINK	ECS link parity error processor.
ECSDSD	ECS	DSD ECS commands CP helper.
ECSSWAP	ECSSWAP	ECS job swapper interface.
ECSUB	ECSSUB	ECS executive subroutines.
ELINK	ECLINK	ECS link restart.
EVICTCH	SPM4	Evict on user sets.
EVICTOW	SPM4	Evict on write.
EXBOOT	RESCH	Start ECS system.
EXRBT	SPM4	PRU conversion - M.ICE function.
ERPEREC	ECSSUB	ECS read parity error recovery.
ERS	MMGR	Empty scheduler request stack.
FILLSTK	ECSSUB	Subroutine to fill CM ECS page stack.
FLSHBUF	ECS	Flush ECS buffer to disk.
FLUSHST	ECSSUB	Subroutine to flush CM ECS page stack.
GETDAE	ECS	Direct access ECS I/O (read).
GETDESC	ECS	Subroutine to read an ECS PRU descriptor.
GETPRU	ECS	Get a PRU.
GETPRE	RACDAC	Subroutine to get preallocated page.
GETRAND	ECS	ECS read driver.
GETSUBP	ECS	Subroutine to get an ECS subpage.

Segment	Deck	Purpose
ILINK	ECLINK	Initialize ECS link.
INDEX	SPM4	Subroutine to convert PRU index to RBT chain format.
INIT	LINKCMR	Initialize system.
LINK	RESCH	Subroutine to link a job into active control point ring.
LINKDST	SPM3	Subroutine to link stack request to DST chain.
LINKVAR	ECLINK	Subroutine for ECS link variables initiation.
OPECLO	CPCIO	CPCIO open/close executive.
OPM	MMGR	Optimize FL priority map.
PACKAGE	RESCH	Subroutine to update RA/FL.
PLBCNT	ECS	Continue loading PP overlay from ECS.
PLBECS	ECS	Load PP overlay from ECS.
PLBERR	ECS	Error while loading PP overlay from ECS.
PLBREL	ECS	Release buffer after loading PP from ECS.
PPLIB	RESCH	Search PP library and load PP overlay.
PREALLO	ECS	Preallocate ECS.
PROCERR	ECLINK	Process driver detected errors.
RAC	RACDAC	Request preallocated ECS space.
RAGET	ECLINK	Subroutine to fetch exchange package RA.
RAPLUS1	RAP1	Identification of RA+1 calls.
RBTRB	SPM5	Subroutine to convert RBT chain format to PRU index.
RCH	CPMTR	Request channel.
RCH3	CPMTR	Channel request subroutine.
READ	CPCIO	CPCIO read executive.
READDAM	SPM5	Prepare stack request to read DAM.
RELECS	ECS	Release ECS buffer.
RELPRE	RACDAC	Subroutine to release preallocated ECS pages.
RELSB	ECS	Subroutine to release system buffer.
RELSUBP	ECS	Subroutine to release an ECS subpage.

Segment	Deck	Purpose
REQEBUF	ECS	Request ECS buffer.
RESCH	RESCH	Select the next job for CPU execution.
RESET	CPCIO	Subroutine to logically rewind ECS buffer.
REWIND	CPCIO	CPCIO rewind executive.
RMRLST	CPCIO	Subroutine to process PRU index list for READLS.
RTAFL	SPM3	Subroutine to request or terminate field access.
SCADDT	SPM4	Subroutine to scan DDT and RBR headers.
SCHRES	MMGR	Scheduler CM resident subroutines.
SEGLINK	LINKCMR	Segment CALL/GOTO linkage processor.
SEGPAR	LINKCMR	ECS segment parity error processor.
SETST	CPMTR	CPU status bit manipulations.
SFCP	ECLINK	Subroutine to set file complete.
SKIPB	CPCIO	CPCIO skip backward executive.
SKIPF	CPCIO	CPCIO skip forward executive.
SPMBKSP	SPM3	Backspace related functions.
SPRBMGR	SPM4	Record block manager.
SSCSEG	SCPT	System control point calls from user control points.
SSFSEG	SCPT	Subsystem function processor.
SREQ	ECLINK	Subroutine to handle ECS link subchannel requests.
SSFS2	SCPT	Secondary set of subsystem functions.
STODAE	ECS	Direct access ECS I/O (write).
STOPRU	ECS	Store PRU.
STORE	ECS	Move data from users CM buffer to ECS buffer.
SWAPCP	SYSIDLE	Swap out all control points.
SWCLEAN	ECS	Clean up after parity on ECS swap file.
SYSIDLE	SYSIDLE	System idle control.
SYSPROG	RESCH	Check status of a user mode system program.
TDI	SYSIDLE	Subroutine to test delay interval.

Segment	Deck	Purpose
TOVSH	SPM5	Subroutine to test overflow of a shared device.
TTS	MMGR	Try to schedule.
TRAUTEB	ECS	Terminate automatic ECS buffer allocation.
TIMSEG	RAP1	Process TIM calls.
UPM	MMGR	Update FL priority map.
USERMOD	CPMTR	Initiate execution of user mode system programs.
USERR	RESCH	Resident user mode entry/exit routines.
WOR	SYSIDLE	Wait for operator to type RESUME.
WRITE	CPCIO	CPCIO write executive.
XJRSEG	RAP1	Process XJR or SAC calls.
XSPM	SPM2	Extension of stack processor manager.

TRANSFER BUFFER TABLE (CYBER 176)



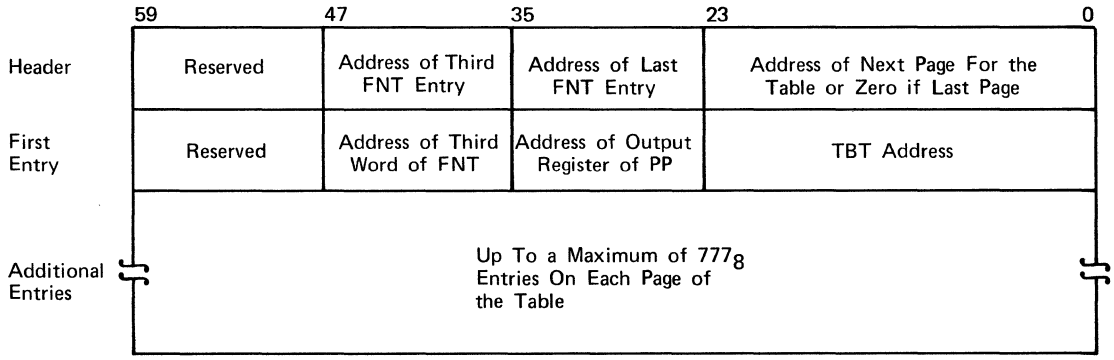
Word	Bit	Description
0	42	TBT is on queue if set.
1	59-36	Flags.

Bit Set	Significance
59	Read/write mode. 0 Read. 1 Write.
58	Stack request completed.
57	Stop flag; set to stop disk transfers on read requests due to new position requirements.
56	Positioning forward flag; set by READSKIP or O.SKF. This flag tells HDC to call HSP when request is complete.
55	Flush flag; set if file switched from write to read (or vice versa).
54	PPIO request.
53	Preread flag; set when prereading on a write request.
52	Postread flag; set when read-filling last sector on write request.
51	Positioning reverse flag.
50	XP bit.
49	Unrecovered error.
48	Evict request.
47	Switch to next PB.
46	Skip first three CM words to read; O.RDPNP or O.RCMPR set flag to skip 77 table.
45	Write in place request.
44	Skip backward over PB.
43	SCB not available.
42	Unrecovered read checkword error.
41	CEM called.
40	CEM complete.
39	Direct access LCM I/O.

Word	Bit	Description
		Bit Set — Significance
		38 — Buffer argument error.
		37 — Special processing for O.RDSK request.
2	59	Wait.
	58	Empty/full.
	35-24	Current PB.†
3	47-36	Last PB.†
	35-24	Next PB.†
13	21-18	PRU level.
	17	Interlock stack request.
	16-12	Order code.
	11-8	Control point.
	7	SPM recall.
	6-0	Stack request ordinal.
14		Same as second word of stack request.
16	59-56	Buffer flags.
		Bit Set — Significance
		59 — Data has been written to disk.
		58 — Data has been read from disk.
		57 — Sector could not be read.
		56 — Sector read with unrecovered checkword error.

† Current PB is the PB being read/written from the disk. When IH switches to the next PB on the same request, it sets last PB equal to current PB and current PB equal to next PB. When a disk request is queued, the desired PB from the stack request is put in current PB by HSP. If this is different from the previous current PB value, the last PB is set to the previous current PB. For read requests, the data in the TBT buffer pertains to last PB or current PB. For write requests, the data in the TBT buffer pertains to current PB or next PB.

TBT ADDRESS TABLE



Header entry:

Bit	Description
59-48	Reserved.
47-36	Address of first FNT entry which has a corresponding entry in this table.
35-24	Address of last FNT entry which has a corresponding entry in this table.
23-0	Address of next page for the table or zero if this is the last page of the table.

Entry:

Bit	Description
59-48	Reserved.
47-36	Address of the third word of the FNT during processing of the flush function issued with byte 2 of the PP output register equal to 1.
35-24	Address of the PP output register during processing of the flush function issued with byte 2 of the PP output register equal to 0.
23-0	TBT address.

SYMBOL DEFINITION F

SYSTEM SYMBOLS

SCPTEXT contains system macros, micros, and symbols used by COMPASS CPU and PP programs that comprise the operating system. SCPTEXT contains the following common decks:

Deck	Contents
ACTCOM	CPU program system action request macros.
COMAFET	File environment table generation macros.
COMSRAS	System communication (RA) symbols.
CPSYS	CPU input/output macros using the Central Program Control (CPC) library routines.
PPSYS	PP program system macros, micros, and symbols.

SCPTEXT is made up of CPCTEXT and PPTEXT. CPCTEXT may be used when only user mode CPU programs are assembled, and PPTEXT may be used when only PP programs are assembled.

Common deck COMSRAS contains definitions of symbols of the form RA.xxx which are addresses of words in the communication area (RA+0 through RA+100).

A listing of system symbols can be obtained with the following job deck:

```
job statement (including a request for MT01)
REQUEST(OLDPL,E,HY) PL1A.
UPDATE(Q)
COMPASS(S=0,I=COMPILE)
7/8/9
*COMPILE PPTEXT
6/7/8/9
```

Refer to the NOS/BE Reference Manual, section 7, for a list of common decks and text overlays.

PPSYS IDENTIFIERS

Common deck PPSYS contains definitions of symbols of the form:

i.mn

i Identifier; one or two alphabetic characters specifying the category to which the symbol belongs.

i	Category
C	Byte number in CM word (0 through 4). C identifiers are used for flags and parameters of 12 bits or less.
CH	Pseudo channel assignments.

i	Category
D	Direct cells.
EX	M.ICE or M.SPM parameter values.
F	Error flag values.
L	Lengths.
LE	Length of table entries.
M	PP request of monitor.
N	Number.
O	Stack processor orders.
OV	PP overlays; mn is the overlay name.
P	CM location of pointer words.
R	PP resident entry points.
S	Number of bits to right shift a parameter to right justify it in a PP word. Some symbols, notably those related to the scheduler, are the number of bits to right shift a parameter to right justify it in a CM word.
SF	Subsystem function.
T	First word address of CM tables. The system programmer should use the P. definition rather than access the table directly with the T. definition.
W	Relative positions in CM tables.

mn Mnemonic; one to six alphanumeric symbols suggesting the use of the symbol. For example, P.ZERO identifies CMR pointer area word 70_8, which contains binary zeros.

INDEX

A directive 11-2
Abnormal job termination 7-17
Abort control point 3-19
Abort flag bit 7-14
Access, exclusive 7-10
Access level 8-10,16
Access loader word 7-25
ACCOUNT control statement 7-13
Accounting, permanent file 6-22
Accounting, time 3-18
ACCSF macro 7-14
ACQUIRE macro 4-5
Active PP link 3-16
ADD directive 8-12
Address public set 6-28
Addresses, list of 5-19
ADDSET control statement 6-23
Aging rate 7-8
Allocatable device area E-3
Allocatable device I/O 5-18
Allocatable devices 4-1
Allocated memory 2-3
Allocation style 4-32; B-187
ALTER function 4-5,12
Analysis directives 11-1,2
APF interlocks 6-19,20
APF table entry 6-21,22; B-88
APR 7-2
Archived file, retrieve 6-14
Area table 2-16; B-131
Area table, local 10-1
ASCII character sets A-1,4
Assigning PPs 3-15
Assigning tapes 4-26,28
Attached permanent file interlocks 6-19,20
Attached permanent file table B-88
Attributes
  Device 4-28,29
  Device set 4-28,29
AUDIT utility 6-14
Auto recall 2-5
Automatic recall (refer to Auto recall)
Automatic tape assignment 4-26,28

B directive 11-2
BATCH bulletin 9-4
Batch jobs 7-8
BC directive 11-2
BKSP macro 5-17
BKSPRU 5-17,26
Block map 10-2
Bootstrap ECS system 10-1,3
Bootstrap program 1-4
Breakpoint table 2-16; B-129
Broken interlock 5-34
Buffer argument error 5-33
Buffer, message 1-1
Buffered I/O 5-39
Bulletin (refer to System bulletin)
Bulletin names 9-2
BULLUP data statements 9-1
BULLUP utility 9-1
Bytes 1-2

CALL macro 2-16
Calls for read/write 5-36
CALLSS macro 2-22
CBM 5-40
CE diagnostic file 4-29
CE error file 5-51
CE error file entry 6-24
CEFAP buffer area B-134
CEJ 2-4
Central exchange jump 2-4; 3-13
Central memory 1-1; 2-1 (also refer to CM)
Central memory resident 1-2; 2-1 (also refer to CMR)
Central processor unit 1-1
CERFILE 4-1
CEVAL macro 7-20
CHANGE directive 8-13
Change running system 8-1
Change user library 8-1
Channel reservations 3-18
Channel status table 4-24; B-16
Channel table 5-45,51; B-89
Character sets A-1
Checkword errors 5-54
CHT 5-45,51; B-89
CIO 5-1
CIO codes 5-1; C-5
CIO function routines 5-4
CIO operation 5-4
CIO tape operations 5-5
Circular buffer 5-1,2

Circular buffer manager 5-40
Circular input/output processor 5-1
CM 1-1,2; 2-1
CM, allocated 2-3
CM, copy 2-13
CM dump 11-2,3
CM field length 7-14
CM match 11-5
CM online dump 11-7
CM organization 2-1
CM queue 7-9
CM read/write order codes 5-24
CM resident library B-142
CM storage moves 2-3
CM, unallocated 2-3
CM words 1-2
CMR 1-2; 2-1,7
CMR areas 2-10
CMR library directory 8-22,23; B-142
CMR loader 10-1
CMR pointer area B-3
CMR segmentation 2-15; 10-1
CMR tables 2-10; B-1
CMRDIR program 10-1,2
CMRES 10-4
CMRLIB library 10-1
COMMENT control statement 7-13
Comment field, EDITLIB 8-18
Common area E-4
Common block name 10-4
Common test and initialization 1-4
Communication area 1-5; 2-4; 5-22
COMPLETE directive 8-13
Console display dumps 11-2
CONTENT directive 8-13
Contents statements 9-2
Control permission 7-10
Control point 1-3; 2-1; 4-1
Control point, abort 3-19
Control point area 1-3; 2-1; 7-11; B-19
Control point dayfile dump 11-2
Control point field length access 3-6
Control point, system 2-21
Control point tables C-1
Control point, user 2-21
Control point 0 4-1
Control statement buffer 7-11
Control statement processing 7-11,13
Control statement record 7-11
Control statement source file 7-11,14
Controller status 5-54
Copy CM 2-13
Core-image dump 11-9
COUNT function 4-5,16
CP routines, EDITLIB 8-18
CPMTR 2-2; 3-12,13
CPMTR functions B-47
CPMTR organization 3-13
CP-PP communication 2-4
CP-system communication 2-4
CPU 1-1
CPU priority 3-14
CPU scheduling 3-14
CPU status 3-21
Crash file 11-6
Create deadstart tapes 8-1,17,30
Create system libraries 8-1
Create user library 8-1
Creating system bulletin file 9-3
CST 4-24; B-16
CTI 1-4
CTI confidence testing 1-4
Cx directive 11-2
CYBER Loader 10-1
Cycles, permanent file 6-21

DAM 4-32; 6-21; D-17
DAM ordinal 4-33
DAT 4-24; B-86
Dayfile buffer area B-114
Dayfile FET B-114
Dayfile, job 4-1; 7-16,17; 11-2
Dayfile message 3-10,21
Dayfile, system 4-1,21,29; 11-2
DDAM file format D-24
DDP 5-39
DDT
    Fixed section 4-23; B-101
    Variable section 4-23; B-103
Deadlock 4-27
Deadstart 1-4
Deadstart dump analyzer 11-1
Deadstart dump, express 11-1
Deadstart functions and options 1-4
Deadstart loader 8-6
Deadstart system tape 4-1
Deadstart tape 1-5
Deadstart tape, create 8-1,17,30
Deadstart tape, modify 8-1
Debug mode 5-32
Default DSDUMP directives 11-6
Delay count, recall 2-5
Delay stack 3-15
DELETE directive 8-13
DELSET 5-38
Device
    Master 4-29
    Permanent file 4-29
    Queue 4-29
    System 4-29; 5-18
Device activity table 4-24; B-86
Device allocation map 6-21; D-17
Device attributes 4-29
Device codes B-53

Device I/O 5-18
Device overflow table 4-30
Device pool table 4-23
Device queue 7-10
Device set 4-28
  Attributes 4-29
  Label D-1
  Permanent file default 4-29
  Private 4-29; 6-23
  Public 4-29
  Queue 4-29
  Scratch 4-29
  Shared 4-30
  System 4-29
  Validation 6-25
Device status table 4-24; B-95,97
Device tables 4-23
Devices
  Allocatable 4-1
  Nonallocatable 4-1
DFILE 4-1
Diagnostic file 4-29
Diagnostic routines, hardware 7-20
Direct access area (ECS) 1-2
Direct access ECS 7-8; 10-1; E-3
Direct cells, PP 3-1
Directives, DSDUMP analysis 11-2
Directory (refer to System directory)
Directory limits 8-22
Disk 4-28
Disk I/O, 819 5-41
Disk tables D-1
Dismountable device table 4-23; B-101
Dismountable pack processing 5-34
Display directives 11-2
DISPOSE control statement/macro 4-3
Dispose files 7-16
Disposition code 4-2; B-70
Distributive data path 5-39
DOT 4-30
DPT 4-23
Drive 4-28
Drop PP 3-19
DSD-INTERCOM communication C-28
DSDUMP control statement 11-1
DSDUMP directives, default 11-6
DSDUMP messages 11-6
DSMOUNT 5-38
DST 4-24; B-95,97
DUMMYnn 10-4
Dump
  Analyzer 11-1
  Buffer controllers 11-2
  CM 11-2,3
  Control point dayfile 11-2
  Directives 11-2
  ECS 11-4
  Entry point table 11-5
  Express deadstart 11-1
  File format, dynamic 11-7
  File name table 11-2
  INTERCOM area 11-5
  INTERCOM buffers 11-5
  INTERCOM tables 11-5
  Job descriptor table 11-2
  PPs 11-4
  PPUs 11-4
  RBRs 11-4
  Request stacks 11-6
  Segment name table 11-5
  Sequencer tables 11-5
  Status control register 11-5
  System 11-1
  System dayfile 11-2
  System dynamic 11-7
  System exchange package 11-5
  TAPES table 11-2
  User field length 11-4,5
Dynamic area (ECS) 1-2
Dynamic dump, system 11-7

ECS vi; 1-1,2
ECS buffered I/O 5-39
ECS buffers 5-39
ECS, direct access E-3
ECS dump 11-4
ECS error recovery 2-20
ECS field length 7-14
ECS label E-2
ECS, match 11-5
ECS paging 1-3
ECS segments E-8
ECS system image 2-18,19
ECS systems, segmentation 2-15
ECS tables E-1
EDD 11-1
EDITLIB
  Character set 8-1
  Control statement 8-7
  CP routines 8-18
  Directive file 8-8
  Directive format 8-12
  Directive restrictions 8-11
  Errors 8-9
  Examples 8-30
  Field length 8-10
  File positioning 8-19
  Files 8-5,19
  Library positioning 8-19
  Limits 8-22

Output 8-8
PP routines 8-18
Syntax 8-3
SYSTEM 8-1,5
Tables 8-22
USER 8-1
Empty page stack B-137
ENCSF macro 7-15
ENDRUN directive 8-13
ENDSEG macro 2-16
Engineering mode 7-20
Entry point 8-28
Entry point cross reference 10-2
Entry point map 10-2
Entry point name table 8-25,28; B-144
Entry point names C-11
Entry point number 8-27
Entry point table dump 11-5
Entry table 2-16; B-132
EPNT 8-25; B-144
Equipment status 7-20
Equipment status table 4-23; B-50
Error file, CE 5-51; 6-24
Error logging status table B-136
Error recovery, ECS 2-20
ERT 8-27; B-144
EST 4-23; B-50
Event stack 3-15
Ex directive 11-4
Examples, LDCMR 10-5
Exchange package 3-14; 11-5; B-18
Exclusive access 7-10
EXIT control statement 7-14
Exit flag bit 7-14
EX.NXTPB subfunction 3-31
EXPORT abort 11-7
Express deadstart dump 11-1
Express jobs 7-8
EX.SPRCL subfunction 3-32
EX.STAT subfunction 3-32
Extended core storage 1-1,2
Extended core storage tables E-1
External reference list 8-28
External reference table 8-27; B-144
External stack request format 5-23

Factory data flag 6-24
FDB C-17
FET 4-19; 5-1; C-4
FET codes C-13
FID 4-4
Field access flag 3-6,12,24
Field length 2-37; 7-14; 8-17
Field length access 3-6,24
Field length, EDITLIB 8-10
Field length override 8-10,17
Field length, user 2-4
File definition block C-17
File disposal 7-16
File environment table 4-19; 5-1; C-4
File formats D-1
File group 7-2
File identifier 4-4
File information table 4-18
File I/O 5-4
File name table 4-23; B-56
File name table dump 11-2
File names, special 4-2,28
File positioning, EDITLIB 8-19
File status information 4-23
File status table entry 4-23
File tables 4-18
Files 1-3; 4-1
Input 7-17
Local 4-1,2; 7-17
Output 7-17
Permanent 4-1,2; 7-17
Public 6-4
Queue 4-1,2
System 2-2; 4-1
Files, EDITLIB 8-5,19
FILMPL 4-2
FILMPR 4-2
FINISH directive 8-13
FIRST 5-2
First level peripheral processors (refer to PPUs)
FIT 4-18
Fixed section 4-23
Flaw table E-7
FLPP (refer to PPUs)
Flush files 7-17
FNT 4-23; B-56
FNT dump 11-2
FSN program 6-28
FST entry 4-23
Function numbers 5-37

GENLDPF utility 6-14
GET function 4-5,12
GETLC macro 7-25
GETLOF macro 7-18
Global library 7-12
GOTO macro 2-16; 10-3
GOTOTAB macro 2-16
Graphic character set A-1
Graphics jobs 7-8

H directive 11-2
HARDPL 4-2
HARDPR 4-2
Hardware diagnostic routines 7-20
Hardware error file 4-1
Hardware interlocks 6-19
Hash points 6-21
Host ID 7-19
HSP 5-43

ICC E-5
ID hash points 6-21
ID table 7-19; B-110
Identifiers 7-19
IDs 7-19
IH 5-43,45
IN 5-2
INCLUDE directive 8-14
Incomplete cycle 6-22
Initialize system 10-3
Input file 7-17
INPUT file 4-2
Input/output 5-1,4,18; 7-3
Input/output tables 4-18
Input queue 4-2,3; 7-1
Input register 1-1; 2-4; 3-5,12
Integrated scheduler 3-28; 7-4
INTERCOM area dump 11-5
INTERCOM buffer area
    INTERCOM 4 B-146
    INTERCOM 5 B-168
INTERCOM buffer dump 11-5
INTERCOM jobs 7-8
INTERCOM multiplexer subtable 4-38
    INTERCOM 4 B-75
    INTERCOM 5 B-81
INTERCOM multiplexer table 4-38
    INTERCOM 4 B-74
    INTERCOM 5 B-80
INTERCOM pointer area 4-38
    INTERCOM 4 B-146
    INTERCOM 5 B-168
INTERCOM queue 7-11
INTERCOM restart 11-7
INTERCOM, system bulletin file 9-4
INTERCOM tables 4-23,38
INTERCOM tables dump 11-5
Intercomputer area E-5
Interjob connection table 2-33
Interlock broken 5-34
Interlock, LDCMR 10-5
Interlocked stack request 6-19,20
Interlocks 6-19
    APF 6-19,20
    Hardware 6-19
    MST 6-19
    PFM 6-19,20
    Set 6-19
    SMT 6-19,20
    Utility 6-19
Internal stack request format 5-23
Interrupt handler 5-43,45
Interval 8-4
Invalid stack entry 5-33
I/O (refer to Input/output)
I/O, ECS buffered 5-39
I/O tables 4-18
IT directive 11-5
ITABL 4-23

JANUS 7-3
JDT 2-2; 7-1,9; B-120
JDT directive 11-2
JDT dump 11-2
Job advancing 7-11
Job classes 7-8
Job control 7-19
Job control area 7-8
Job control point tables C-1
Job control statement source file 7-14
Job dayfile 4-1; 7-16
Job descriptor number 2-2
Job descriptor table 2-2; 7-9; 11-2; B-120
Job descriptor table dump 11-2
Job descriptor table entry 7-1; B-120
Job input file 7-11,14
Job input queue 7-1
Job killed 7-18
Job pause bit 7-11
Job postprocessing utilities 7-18
Job prescheduling 4-26
Job processing 7-1
Job queue, PP 3-15
Job rerun 2-35
Job scheduling 4-27; 7-1,6
Job scheduling queues 7-1,9
Job sequencer 7-2
Job status 2-32
Job status code 7-9
Job termination
    Abnormal 7-17
    Normal 7-16
Jobs
    Batch 7-8
    Express 7-8
    Graphics 7-8
    INTERCOM 7-8
    Multiuser 7-8
    Sequencer 7-2

Kill Job 7-18

Label, rewrite RMS 6-24
LABELMS control statement 6-23
LCM vi; 5-41
LCM buffer 5-50
LCME vi
LDC 10-5
LDCMR 10-1
  Control statement 10-1
  Examples 10-5
  Files 10-3
  Interlock 10-5
  Map 10-2
  Reserved names 10-4
  System security 10-4
Libraries, system 1-1,4; 8-1
LIBRARY directive 8-14
Library directory 8-18; 10-1
Library files 1-3
Library limits 8-22
Library name 8-3,11
Library name table 8-15,23; B-143
Library positioning, EDITLIB 8-19
Library table interfaces 8-29
Library, user 8-1,20
LIMIT 5-2
LIMIT control statement 7-13
Limits, EDITLIB 8-22
Link ID 7-19
List core-image dump 11-9
List of addresses 5-19
LISTCID error messages 11-10
LISTCID utility 11-9
Listing dump files 11-9
LISTLIB directive 8-14
LISTLNT directive 8-15
List-of-files address 7-18
LNT 8-22; B-143
Load dumped files from tape 6-14
Loader, CYBER 10-1
Loader map 10-2
Loader table 2-4
Loader word 7-25
Loading 1-4
Loading jobs from tape 7-2
LOADPF utility 6-14
Local area table 10-1
Local file name SYSTEM 8-7
Local file names C-7
Local files 7-17
Logical flaw table 6-22; D-20
Logical groups 7-19
Logical identifiers 7-19
LOGIN bulletin 9-4
LOGIN command 7-14
Long-term connection 2-33
M directive 11-2
MAB B-125
M.ABORT function 3-19
Macro requests 6-2,5
Magnetic tape (refer to Tape)
Mailbox B-109
Mainframe 7-19
Mainframe attribute block B-125
Mainframe ID 6-4
Map, loader 10-2
Master device 4-29
Match word 0 11-6
Maximum field length 2-37
Maximum logical record size 4-19
Maximum queue priority 7-8
M.BUFPTR function 3-20
M.CCPA function 3-20
M.CLRST function 3-20
M.CPJ function 3-20
M.CPUST function 3-21
M.DCP function 3-21
M.DFM function 3-21
MDI routine 8-21
M.DPP function 3-22
M.EES function 3-22
M.EESD function 3-23
Memo mode 3-29
Memory (refer to CM)
Message buffer 1-1
Message, dayfile 3-10,21
Message, dump 11-3
Message to UCP 2-29
M.ICE function 3-23
Minimum queue priority 7-8
M.ISP function 3-23
M.KILL function 3-23
MLRS 4-19
M.MFLA function 3-24
M.NOTE function 3-24
M.NTIME function 3-24
MODE control statement 7-13
Mode of RECOVER 6-26
Modify deadstart tape 8-1
Modify running system 8-1
Modify user library 8-1
Monitor area 10-3
Monitor field length access 3-24
Monitor functions 3-18; B-47
Monitor mode 3-13
Monitor mode segment name 11-5
Monitor, system 1-1; 3-12; 5-29
Monitor trace mode 3-32
MOUNT control statement 6-24
Mounted set table 4-24; B-99

MOVE directive 8-15
Move permanent files/tables 6-14
Move system directory 8-21
M.PASS function 3-24
M.PATCH function 3-24
M.PPLIB function 3-25
M.RACT function 3-25
M.RBTSTO function 3-25
M.RCH function 3-26
M.RCLCP function 3-26
M.RCP function 3-26
M.RPJ function 3-27
M.RPJD function 3-27
M.RSTOR function 3-27
M.SCB function 3-28
M.SCH function 3-28
M.SEF function 3-28
M.SEG function 3-29
M.SETST function 3-29
MSG table C-28
M.SLICE function 3-30
M.SLPER function 3-30
M.SPM function 3-30
M.SPRCL function 3-31
MST 4-24; B-99
MST set interlocks 6-19
MTR 1-1; 3-12; 5-29
MTR structure 3-12
M.TRACE function 3-32
M.TSR function 3-32
Multiple access 4-24
Multiplexer subtable 4-38
Multiplexer table 4-38
Multiuser job table B-166
Multiuser jobs 7-8

Name, EDITLIB 8-1
Name statements 9-2
NFL 8-10
Nominal field length 8-10
Nonallocatable devices 4-1
Normal job termination 7-16

Online dump, PP/CM 11-7
Operator action queue 7-11
Order codes 5-24,33
  CM read/write 5-24
  Positioning 5-25
  PP read/write 5-25
  SPM communication 5-26
  Stack processor communications 5-26
OUT 5-2
Output file 7-17
OUTPUT file 4-2
Output queue 4-2,3; 7-1
Output register 1-1; 3-5,12; 5-1
Overcommitment of tape drives 4-27
O26/O29 mode A-1

P display 7-1
Pack processing 5-34
Pack removal 5-38
Paged area 1-2
Paging 1-3
PAM D-23
Parity error 5-33
PAT directive 11-5
Pattern matching 11-5
Pause bit 7-11
PEEK function 4-5,13
Periodic recall 2-5
Peripheral job table B-115
Peripheral processors 1-3; 3-1 (also refer to PP)
Permanent dump tape format 6-15
Permanent file 4-2; 6-1; 7-17
  Accounting 6-22
  Catalog 6-14,20; D-10
  Control statements 7-14
  Cycles 6-21
  Default set 4-28,29
  Device 4-29
  Directory 6-20; D-6
  Directory entry 6-14; D-7
  Dump to tape 6-14,15
  Functions 6-1
  Incomplete cycles 6-22
  Interlocks 6-19
  Load 6-14
  Name 6-21
  Queue 7-10
  Statistics 6-14
  Status 6-14
  System interface 6-19
  Tables 6-20
  Transfer 6-14
  Utilities 6-14
  Utility function codes 6-14
  Versions 6-21
Permanent pack queue 7-10
Permission, control 7-10
PFC 6-14,20; D-10
PFC allocation matrix D-23
PFD 6-20; D-6
PFD entry 6-14; D-7
PFL 8-10
PFLOG dump tape format 6-15
PFLOG tape 6-14
PFLOG utility 6-14

PFM interlocks 6-19,20
Physical flaw tables 6-22; D-21
Physical ID 7-19
Physical record unit 4-30
Physical status of tape drive 4-27
PLOT 4-2
Pool PP 3-3
Pool processor 3-3
Positioning order codes 5-25
Postprocessing utilities 7-18
PNT 8-5,27; B-145
PP assignment 3-15
PP calls 3-15
PP communication 2-4; 3-5
PP communication area 1-5; 3-5; B-27
PP direct cells 3-1
PP directive 11-4
PP drop 3-19
PP dump 11-4
PP input register 3-5
PP job queue 3-15
PP library 3-25
PP library pointer 8-19
PP link, active 3-16
PP match 11-5
PP monitor 3-5
PP online dump 11-7
PP organization 3-1
PP output register 3-5; 5-1
PP program name table 8-5,22,26; B-143
PP program names B-28
PP program, transient 3-5
PP read/write order codes 5-25
PP resident 3-1,6
PP resident segment table B-127
RR routines 3-6
PP routines, EDITLIB 8-18
PP status 11-2
PP status table 6-12
PP status words B-17
PPIO processing 5-48
PPMTR 3-5
PPMTR functions B-48
PPNT 8-26; B-143
PPs 1-1; 3-1; 5-41
PPS 5-41
PPSYS identifiers F-1
PPU dump 11-4
PPUs 5-41
Predayfile name 4-3
Prefix tables 8-17
Preinput queue 7-1
Prescheduling display 4-26; 7-1
Prescheduling queue 4-26
Prescheduling tape jobs 4-26
Priority 7-8
Priority lockout bit 7-10
Private device set 4-28,29
Private device set processing 6-23
Program library file 1-3
Program limits 8-22
Program name 8-4
Program name table 8-5,22; B-143
Program recall 2-5
PRU 4-30
Public device set 4-28,29
Public device set addressing 6-28
Public files 6-4
PUNCH 4-2
PUNCHB 4-2
P80C 4-2

Q set (refer to Queue set)
Quantum priority 7-8
Quantum time length 7-8
Queue device 4-29
Queue files 4-2
Queue set 4-28,29
Queues
  Central memory 7-9
  Device 7-10
  Input 4-2,3; 7-1
  INTERCOM 7-11
  Operator action 7-11
  Output 4-2,3; 7-1
  Permanent file 7-10
  Permanent pack 7-10
  Prescheduling 4-26
  Scheduling 7-1

RA communication area 2-4; C-1
Random file 8-16,20
Random user library 8-15
RANTOSEQ directive 8-15
RA+1 requests 5-1
RB 4-30
RB bytes 4-32
RB numbers 4-32
RBR 4-30,32; B-95
RBR directive 11-4
RBT 1-2; 2-1; 4-32
RBT area dump 11-4
RBT catalog 6-20
RBT chain 4-32; 11-4; D-16
RBTC 6-20; 11-4
RBTC directive 11-4
R.DCH routine 3-9
R.DFM routine 3-10
READ macro 5-6,24
READC macro 5-18,25
READLS macro 5-19,25

READN 5-8
READNS macro 5-24
READSKP macro 5-9,24
Read/write calls 5-36
Read/write order codes 5-24
READY directive 8-15
Recall
Auto 2-5
Delay count 2-5
Periodic 2-5
Record block 4-30
Record block reservation table 4-30,32; B-95
Record block reservation table dump 11-4
Record block table 1-2; 2-1; 4-32; B-185
Record descriptors B-139
Recording mode 6-24,25
RECOVER control statement 6-25
RECOVER mode 6-26
Recovered parity error 5-33
Recovery programs 7-18
RECOVR macro 7-18
Recreate RMS label 6-24
Reducing system bulletin file 9-3
Register
Input 1-1; 2-4; 3-5,12
Output 1-1; 3-5,12; 5-1
RELABEL control statement 6-24
REMOVE directive 8-16
Removing a pack 5-38
REPLACE directive 8-16
Replace system directory 8-7
REPRIEVE macro 7-18
Reprieve processing 7-18
REQ function parameter list C-21
Request count, SCP 2-33
Request scheduling table B-94
Request stack 5-18
Request stack entry B-92
Request, system 1-1; 2-4
Request word 1-1
R.EREQS routine 3-10
RERUN command 7-18
Rerun job 2-35
Rerun status 2-35
Reserved names, FET C-7
Reserved names, LDCMR 10-4
RESET 8-7
Resource recovery mode 6-27
RESTORE 8-8
Retrieve archived file 6-14
RETURN macro 2-16
REWIND directive 8-16
Rewrite RMS label 6-24
RFL control statement 7-14
R.IDLE routine 3-6
RMS 4-1
RMS device, rewrite label 6-24
RMS hardware error 5-34
RMS set 4-28
RMS tables 4-31
R.MTR routine 3-8
Rolling 7-6
Rotating mass storage 4-1
ROUTE control statement 4-3
ROUTE macro 4-3
R.OVL routine 3-10
R.OVLJ routine 3-6
RPHR 5-10
RPV program 7-18
R.RAFL routine 3-6
R.RCH routine 3-8
R.READP routine 3-4
R.RWP routine 3-4
R.STB routine 3-9
R.STBMSK routine 3-9
R.TAFL routine 3-6
R.TFL routine 3-8
Running system, modify 8-1
R.WAIT routine 3-8
R.WRITEP routine 3-4

SCB 3-28; 5-39; B-141
Scheduler control 7-1
Scheduler performance table B-116
Scheduling jobs 7-6
Scheduling queues 7-1,9
Scheduling tape jobs 4-27
Scheduling tapes 4-26
SCHPT B-116
SCHT C-24
SCP 2-21
Abnormal termination 2-36
Accounting 2-30
Exit 2-32
Normal termination 2-36
Request count 2-33
Wait response count 2-33
SCPA 2-26
Scratch sets 4-28,29
Sector 4-30
SEG directive 11-5
SEGDEF macro 2-16
Segment 2-15
Definition 2-16
Descriptor 2-18
In ECS system E-8
Linkage 2-16
Loading 2-16
Name table dump 11-5
Table 2-18; 10-1; B-133
Segmentation, CMR 2-15; 10-1
SEGMFL macro 2-37

SEQ directive 11-5
SEQTORAN directive 8-16
Sequencer jobs 7-2
Sequencer table 7-2; B-97
Sequencer table dump 11-5
Sequential file 8-15,20
Sequential user library 8-16
Set 4-28
  Attributes 4-28; 6-28
  Device 4-28
  Interlocks 6-19
  Member table 6-21; D-19
  Permanent file default 4-28
  Private 4-28
  Public 4-28
  Queue 4-28
  RMS 4-28
  Scratch 4-28
  Shared 4-30
  System 4-28
SETAL directive 8-16
SETFL directive 8-17
SETFLO directive 8-17
SETLC macro 7-25
SETLOF macro 7-18
SETNAME control statement 7-14
SF functions list 2-34
SFCALL function codes 2-28
SFCALL macro 2-26
SF.CLTC function 2-33
SF.ENDT function 2-31
SF.EXIT function 2-32
SF.LIST function 2-34
SR.READ function 2-32
SF.REGR function 2-29
SF.RERN function 2-35
SF.SLTC function 2-33
SF.STAT function 2-32
SF.SWPI function 2-33
SF.SWPO function 2-33
SF.TIME function 2-30
SF.WRIT function 2-32
Shared device set 4-30
Short PRU 5-5
Skip backward directive 8-17
Skip forward directive 8-17
SKIPB directive 8-17
SKIPB macro 5-17,26
SKIPF directive 8-17
SKIPF macro 5-16,25
SMT 6-21; D-19
SMT interlocks 6-19,20
SNT C-26
Source file 7-11,14
Source ID 4-3
Special dump directives 11-5
Special file names 4-2,3,28
SPM 5-21
SPM communication order codes 5-26
SPM-1SP interface 5-30
SPM-1SQ interface 5-31
SPM-3DO interface 5-36
Spot name table C-26
Spun-off task C-26
SR directive 11-6
SSCT macro 2-37
Stack entry, invalid 5-33
Stack processor 5-21
Stack processor, error conditions 5-32
Stack processor order codes 5-24,26
Stack processor recall 3-30
Stack processor, system interface 5-27
Stack request dump 11-6
Stack request formats
  External 5-23
  Internal 5-23
Status control register dump 11-5
Status of equipment 7-20
Status of job 2-32
Status of tape drive 4-27
Status, permanent file 6-14
STF routine 2-13
STG 4-25
Storage moves 2-3
Storage requests 3-19
Subchannel table C-24
Subdirectory 6-21; D-22
Subpage buffer B-138
Subsystem
  Control table 2-33; B-112
  Definition 2-21,37
  Functions 2-26,34
  Swap-in 2-33
  Swap-out 2-33
  Task complete 2-31
SUMMARY control statement 7-14
SUP bulletin 9-4
Swap file format D-26
Swap-in 2-33; 7-6
Swap-out 2-2,33; 7-6
SWITCH control statement 7-13
Symbol 8-1
SYSBULL control statement 9-1
System area (ECS) 1-2
System bulletin 9-1
System bulletin file
  Creating 9-3
  Reducing 9-3
  Updating 9-3
System circular buffer 3-28; 5-39,40; B-141
System communication 2-4
System control point 2-21
System dayfile 4-1,29; 11-2
System dayfile area 4-21
System dayfile dump 11-2

System deadlock 4-27
System default set 4-29
System definition F-1
System device 4-29; 5-18
System directory 8-6,21
System directory, replace 8-7
System dumps 11-1
System dynamic dump 11-7
System dynamic dump file 11-7
SYSTEM EDITLIB 8-1,5
System exchange package dump 11-5
SYSTEM file name 8-7
System files 2-2; 4-1
System files, EDITLIB 8-19
System image 2-18,19; 10-2
System libraries 1-1,4; 8-1
System libraries, create 8-1
System loading 1-4
SYSTEM macro 6-28; 7-3
System monitor 1-1,3; 3-12; 5-29 (also refer to Monitor)
System page E-7
System pointer area E-6
System programs 5-1
System requests 1-1
System security
    EDITLIB 8-20
    LDCMR 10-4
System set 4-28,29
System symbols F-1
System table find 2-13
System tape 1-5; 4-1

Tape drive
    Assignment 4-26
    Automatic assignment 4-26
    Overcommitment 4-27
    Physical status 4-27
    Scheduling 4-26
    Status checking 4-27
Tape drivers 5-5
Tape error recovery drivers 5-5
Tape job prescheduling 4-26
Tape job prescheduling display 4-26; 7-1
Tape job scheduling 7-1
Tape jobs, loading 7-2
Tape operations 5-5
Tape unit recovery table B-106
Tape, unlabeled 4-26,28
TAPES directive 11-2
Tapes staging table 4-25; B-87
TAPES tables 11-2; B-105
TBT 5-41,50; E-12
T.IJCT 2-33
Time accounting 3-18
TLOAD routine 7-2
Trace mode 3-32
Transfer buffer table 5-41,50; E-12
TRANSFER directive 8-17
Transfer permanent files/tables 6-14
Transient PP program 3-5
TRANSPF utility 6-14
TRANS77 directive 8-17
T.SSCT 2-33

UCP 2-21
UCP, abort 2-29
UCP, end processing 2-35
UCP, message 2-29
UCP, read 2-32
UCP, write 2-32
UCPA 2-26
UFL directive 11-4,5
Unallocated memory 2-3
Uncorrectable parity error 5-33
Unit queue table 5-43,51; B-91
Unit record I/O 7-3
Universal permission 6-4
Unlabeled tapes 4-26,28
Updating system bulletin file 9-3
UQT 5-43,51; B-91
User area 10-3
User control point 2-21
USER EDITLIB 8-1
User field length 2-4
User field length dump 11-4,5
User files, EDITLIB 8-19
User library 8-1,20
User library, modify 8-1
User library, random 8-15
User library, sequential 8-16
User mode 3-13
User mode segment name 11-5
User program 5-1; 7-18
Utility interlocks 6-19

Validate device set 6-25
Validate RBT chains 11-4
Variable section 4-23
VERIFYJ macro 4-16
Versions, permanent files 6-21
Vocabulary word 8-1
Volume serial number 4-26
VSN 4-26,28
VSNBUF B-104

Wait response count, SCP 2-33
WO directive 11-6

Word 0, match 11-6
Words, CM 1-2
WPHR 5-16
Write calls 5-36
WRITE macro 5-11,25
Write order codes 5-24
WRITEC macro 5-20,25
WRITEF macro 5-13,25
WRITEN 5-15
WRITER macro 5-12,25

XP directive 11-5

ZZZZCMR file 4-1; 10-1,2
ZZZZZDD file 11-7
ZZZZZ04 file 4-29; 5-18
ZZZZZ06 file 4-2
ZZZZZ23 file 4-2,29; 8-6

1PK functions 5-37
1PK routine 7-10
1SP 5-22,30
1SQ 5-22,31
3DO 5-36
819 disk I/O processing 5-41
    Logical I/O processing 5-41
    Physical I/O processing 5-45
    Tables 5-50

COMMENT SHEET

MANUAL TITLE: CDC NOS/BE Version 1 System Programmer's Reference Manual

PUBLICATION NO.: 60494100 **REVISION:** K

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).

CUT ALONG LINE

AA3419 REV. 4/79 PRINTED IN U.S.A.

NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

FOLD ON DOTTED LINES AND STAPLE

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

Publications and Graphics Division
ARH219
4201 North Lexington Avenue
Saint Paul, Minnesota 55112

CUT ALONG LINE

FOLD FOLD

PAGE INDEX TO MEMORY RESIDENT TABLES

Table	Page
Area Table (ECS System)	B-130
Attached Permanent File Table	B-88
Breakpoint Table (ECS System)	B-129
CEFAP Buffer	B-133
Channel Status Table	B-16
Channel Table	B-89
CM Resident Tables	B-1
CMR Library Directory	B-142
CMR Pointer Area	B-3
Control Point Areas	B-19
Dayfile FET and Buffer Area	B-112
Device Activity Table	B-86
Device Overflow Table	B-97
Device Status Table	B-95
Dismountable Device Table	B-101
Empty Page Stack	B-137
Entry Table	B-131
Equipment Status Table	B-50
Error Logging Status Table	B-126
File Name Table	B-56
INTERCOM 4 Multiplexer Table	B-74
INTERCOM 5 Multiplexer Table	B-80
INTERCOM 4 Pointer and Buffer Area	B-146
INTERCOM 5 Pointer and Buffer Area	B-168
INTERCOM Table	B-74
INTERCOM 4 User Table	B-154
INTERCOM 5 User Table	B-175
Library Directory	B-142
Logical ID Table	B-110
Mainframe Attribute block	B-114
Mounted Set Table	B-99
Peripheral Job Table	B-113
PP Communication Areas	B-27
PP Resident Overlay Save Buffer	B-127
PP Status Words	B-17
Record Block Reservation Table	B-95
Record Block Table Entry	B-185
Request Scheduling Table	B-94
Request Stack Entry	B-92
Scheduler Job Control Area	B-118
Scheduler Job Descriptor Table	B-120
Scheduler Performance Table	B-116
Scheduler Mailbox Buffer	B-109
Segment Table	B-132
Sequencer Table	B-97
Subpage Buffer	B-139
Subsystem Control Table	B-114.2
System Circular Buffer	B-138
System Job Exchange Package Area	B-26
Tape Unit Recovery Table	B-106
Tapes Staging Table	B-87
Tapes Table	B-105
TBT Address Table	E-16
Unit Queue Table	B-91
VSN Buffer	B-104

CORPORATE HEADQUARTERS, P.O. BOX 0, MINNEAPOLIS, MINN. 55440
SALES OFFICES AND SERVICE CENTERS IN MAJOR CITIES THROUGHOUT THE WORLD

LITHO IN U.S.A.

CONTROL DATA CORPORATION