
CONTROL DATA®
CYBER 70 COMPUTER SYSTEMS
MODELS 72, 73, 74
6000 COMPUTER SYSTEMS

SCOPE SYSTEM PROGRAMMER'S REFERENCE MANUAL
MODEL 72, 73, 74 VERSION 3.4
6000 VERSION 3.4

REVISION RECORD

REVISION	DESCRIPTION
A (10-15-71)	Original printing.
B (11-3-72)	The EDITLIB section has been completely replaced to reflect changes and improvements to SCOPE 3.4 EDITLIB. Pages affected: Front Matter; 8-1 thru 8-38; Comment Sheet.
C (5-25-73)	Changes and corrections resulting from product development and documentation evaluation. This is a complete reprint. Pages affected: 1-7; 2-1, 2-5, 2-11 thru 2-15, 2-17 thru 2-31; 3-1, 3-4, 3-6, 3-8, 3-11 thru 3-13, 3-16, 3-17, 3-19, 3-20, 3-24 thru 3-27, 3-29 thru 3-32; 4-5, 4-8, 4-9, 4-14, 4-16 thru 4-16.2, 4-18 thru 4-22, 4-24 thru 4-42, 4-45 thru 4-48, 4-55 thru 4-63; 5-2, 5-4, 5-6, 5-7, 5-16, 5-24, 5-27 thru 5-30, 5-32, 5-33; 6-3, 6-5 thru 6-9, 6-19, 6-20, 6-22, 6-24 thru 6-26, 6-28, 6-32, 6-35; 7-3, 7-5 thru 7-7, 7-11, 7-13 thru 7-16.1, 7-22 thru 7-26, 7-28; 8-6, 8-11 thru 8-13, 8-17; B-1 thru B-23, B-25 thru B-31, B-33 thru B-42; Index-1 thru Index-13, Comment Sheet.
D (1-15-74)	Updated to reflect SCOPE 3.4.1 release and miscellaneous technical changes. Affected pages: iii thru vii; 1-5 thru 1-8; 2-5 thru 2-8, 2-11, 2-12, 2-12.1, 2-13 thru 2-18, 2-23 thru 2-26, 2-29 thru 2-32; 3-17 thru 3-22, 3-25 thru 3-30, 3-33, 3-34; 4-5 thru 4-8, 4-13, 4-14, 4-16.1, 4-16.2, 4-17 thru 4-18.2, 4-23 thru 4-26, 4-29 thru 4-46, 4-49 thru 4-52, 4-57 thru 4-58.1; 5-1 thru 5-2.1, 5-5 thru 5-8, 5-23, 5-24, 5-27 thru 5-30, 5-35; all of section 6; 7-3, 7-4, 7-9 thru 7-12, 7-15 thru 7-24, 7-27, 7-28; 8-13, 8-14, 8-14.1; add new section 9; all of appendix B; entire index; Comment Sheet.
E (5-31-74)	Updated to reflect SCOPE 3.4.2 release and technical changes. Includes SCOPE System Tables from Part II of the Installation Handbook (Pub. No. 60307400). Affected pages: iv thru vii; 1-2; 2-11 thru 2-27, 2-30, 2-32; 3-8, 3-12.1, 3-13, 3-15 thru 3-20, 3-26, 3-34; 4-2 thru 4-3.5, 4-6, 4-7, 4-14, 4-16.1, 4-16.2, 4-18 thru 4-28, 4-31 thru 4-36, 4-39, 4-44; 5-27 thru 5-36; 6-35, 6-37, 6-46; 7-1 thru 7-5, 7-12 thru 7-16, 7-25, 7-29 thru 7-37; 8-20, 8-23, 8-28, 8-34, 8-35, 8-37; B-3 thru B-77; entire Part II; entire Index; Comment Sheet.
Publication No. 60306500	

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 Ave.
Arden Hills, Minnesota 55112

or use Comment Sheet in the back of this manual

© 1971, 1972, 1973, 1974, 1976, 1979
Control Data Corporation
Printed in the United States of America

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	Revision
Cover	-
Title Page	-
ii	L
ii-a/ii-b	L
iii	L
iv	L
iv-a	L
iv-b	L
v	L
vi	L
vii	L
viii	L
ix	L
1-1	A
1-2	F
1-3	L
1-4	A
1-5	D
1-6	L
1-7	H
1-8	D
2-1	C
2-2	G
2-3	A
2-4	L
2-5	L
2-6	G
2-7	F
2-8	F
2-9	L
2-10	L
2-11	L
2-12	H
2-13	F
2-14	F
2-15	H
2-16	F

Page	Revision
2-17	G
2-18	F
2-19	L
2-20	L
2-21	K
2-22	L
2-23	K
2-24	L
2-25	L
2-26	L
2-27	L
2-28	L
2-29	L
2-30	K
2-31	L
2-32	L
2-33	K
2-34	L
3-1	L
3-2	K
3-3	L
3-4	C
3-5	F
3-6	L
3-7	K
3-8	L
3-9	K
3-10	G
3-11	L
3-12	G
3-12.1	E
3-13	G
3-14	L
3-15	G
3-16	G
3-17	L
3-18	L

Page	Revision
3-19	K
3-20	L
3-21	G
3-22	G
3-23	G
3-24	L
3-25	L
3-26	L
3-27	L
3-28	L
3-29	L
3-30	L
4-1	L
4-2	F
4-3	H
4-4	H
4-5	F
4-6	H
4-7	H
4-8	G
4-9	K
4-10	K
4-11	K
4-12	L
4-13	L
4-14	K
4-15	K
4-16	K
4-16.1	K
4-16.2	K
4-16.3	K
4-16.4	K
4-16.5/4-16.6	K
4-17	K
4-18	F
4-19	F
4-20	F

Page	Revision
4-21	H
4-22	F
4-23	K
4-24	K
4-25	L
4-26	L
4-26.1/4-26.2	L
4-27	L
4-28	H
4-29	H
4-30	H
4-31	F
4-32	H
4-33	H
4-34	H
4-35	H
5-1/5-2	L
5-3	L
5-4	C
5-5	A
5-6	L
5-7	D
5-8	A
5-9	A
5-10	A
5-11	A
5-12	A
5-13	A
5-14	A
5-15	L
5-16	C
5-17	A
5-18	A
5-19	A
5-20	A
5-21	A
5-22	A
5-23	D
5-24	H
5-25	H
5-26	H
5-27	L
5-28	K
5-29	K
5-30	K
5-31	K
5-32	K
5-32.1/5-32.2	K

Page	Revision
5-33	K
5-34	K
5-35	H
5-36	H
5-37	H
5-38	H
5-39	H
5-40	H
5-41	H
6-1	F
6-2	F
6-3	K
6-4	F
6-5	F
6-6	F
6-7	L
6-8	H
6-9	F
6-10	F
6-11	H
6-12	A
6-13	H
6-14	H
6-15	L
6-16	A
6-17	H
6-18	H
6-19	F
6-20	H
6-21	D
6-22	D
6-23	H
6-24	G
6-25	H
6-26	H
6-27	D
6-28	D
6-29/6-30	H
6-31	H
6-32	H
6-33	H
6-34	H
6-35	H
6-36	L
6-37	K
6-38	H
6-39	H
6-40	H

Page	Revision
6-41	H
6-42	H
6-43	H
6-44	H
6-45	K
6-46	K
7-1	E
7-2	L
7-3	E
7-4	H
7-5	L
7-6	C
7-7	J
7-8	K
7-9	K
7-10	K
7-11	F
7-12	K
7-13	K
7-14	F
7-15/7-16	L
7-17	H
7-18	H
7-19	F
7-20	F
7-21	F
7-22	F
7-23	F
8-1	L
8-2	A
8-3	A
8-4	A
8-5	L
8-6	L
8-7	B
8-8	L
8-9	B
8-10	B
8-11	C
8-12	C
8-13	C
8-14	H
8-14.1	D
8-15	B
8-16	B
8-17	G
8-18	B
8-19	B

Page	Revision
8-20	E
8-21	B
8-22	H
8-23	L
8-24	B
8-25	B
8-26	B
8-27	K
8-28	F
8-29	B
8-30	B
8-31	H
8-32	B
8-33	B
8-34	L
9-1	D
9-2	D
9-3	D
9-4	D
10-1	L
10-2	L
10-3	L
10-4	L
10-5	L
10-6	F
10-7	F
10-8	F
10-9	F
10-10	F
10-11	F
11-1	L
11-2	L
11-3	L
11-4	L
11-5	L
A-1	C
A-2	C
A-3	C
A-4	C
A-5	C
A-6	C
A-7	C
B-1	K
B-2	K
II-i	L
II-ii	L
II-iii	L

Page	Revision
Divider	-
II-1-1	L
II-1-2	L
II-1-3	L
II-1-4	K
II-1-5	K
II-1-6	L
II-1-7	L
II-1-8	L
II-1-9	L
II-1-10	F
II-1-11	F
II-1-12	F
II-1-13	L
II-1-14	K
II-1-15	H
II-1-16	K
II-1-17	G
II-1-18	K
II-1-19	G
II-1-20	L
II-1-21	F
II-1-22	F
II-1-23	F
II-1-24	H
II-1-25	H
II-1-26	L
II-1-27	K
II-1-28	L
II-1-29	L
II-1-30	L
II-1-31	L
II-1-32	J
II-1-33	K
II-1-34	K
II-1-35	K
II-1-36	H
II-1-37	J
II-1-38	J
II-1-39	K
II-1-40	K
II-1-41	H
II-1-42	K
II-1-43	H
II-1-44	H
II-1-45	J
II-1-46	L
II-1-47	F

Page	Revision
II-1-48	F
II-1-49	F
II-1-50	F
II-1-51	F
II-1-52	F
II-1-53	K
II-1-54	K
II-1-55	K
II-1-56	F
II-1-57	K
II-1-58	H
II-1-59	K
II-1-60	L
II-1-61	K
II-1-62	K
II-1-63	L
II-1-64	K
II-1-65	L
II-1-66	L
II-1-67	L
II-1-68	F
II-1-69	L
II-1-70	F
II-1-71	K
II-1-72	K
II-1-73	L
II-1-74	K
II-1-75	F
II-1-76	G
II-1-77	K
II-1-78	K
II-1-79	F
II-1-80	L
II-1-81	F
II-1-82	F
II-1-83	F
II-1-84	J
II-1-85/II-1-86	J
II-1-87/II-1-88	K
II-1-89	H
II-1-90	F
II-1-91	F
II-1-92	H
II-1-93	F
II-1-94	F
II-1-95	K
II-1-96	K
II-1-97	K

Page	Revision
II-1-98	H
II-1-99	H
II-1-100	H
II-1-101	H
II-1-102	H
II-1-103	L
II-1-104	K
II-1-105	L
II-1-106	L
II-1-106.1/ II-1-106.2	L
II-1-107	K
II-1-108	H
II-1-109	L
II-1-110	K
II-1-111	K
II-1-112	L
II-1-113	H
II-1-114	H
II-1-115	H
II-1-116	H
Divider	-
II-2-1	L
II-2-2	L
II-2-3	L
II-2-4	L
II-2-5	E
II-2-6	L
II-2-7	E
II-2-8	L
II-2-8.1	L
II-2-8.2	L
II-2-9	K
II-2-10	K
II-2-11	K
II-2-12	K
II-2-13	K
II-2-14	K
II-2-15	K
II-2-16	K
II-2-17	K
II-2-18	K
II-2-19	K
II-2-20	L
II-2-21	L
II-2-22	H
II-2-23	E
II-2-24	L

Page	Revision
II-2-25	F
II-2-26	L
II-2-27	F
II-2-28	L
II-2-29	F
Divider	-
II-3-1	H
II-3-2	L
II-3-3	H
II-3-4	E
II-3-5	H
II-3-6	L
II-3-7	H
II-3-8	L
II-3-9	L
II-3-10	H
II-3-11	L
II-3-12	H
II-3-13	H
II-3-14	K
II-3-15	L
II-3-16	H
II-3-17	H
II-3-18	H
II-3-19	H
II-3-20	H
II-3-21	K
II-3-22	K
II-3-23	H
II-3-24	H
II-3-25	H
II-3-26	H
II-3-27	H
II-3-28	H
II-3-29	H
II-3-30	H
II-3-31	H
II-3-32	H
II-3-33	H
II-3-34	H
II-3-35	H
Divider	-
II-4-1	H
II-4-2	L
II-4-2.1/II-4-2.2	L
II-4-3	E
II-4-4	L
II-4-5	E

Page	Revision
II-4-6	K
II-4-7	K
II-4-8	K
Index-1	L
Index-2	L
Index-3	L
Index-4	L
Index-5	L
Index-6	L
Index-7	L
Comment Sheet	L
Back Cover	-

PREFACE

This manual describes the SCOPE 3.4.4 Operating System for the CDC® CYBER 70 Models 72, 73, and 74 and CDC 6000 Series computers. It is written for system programmers who perform system evaluation or program modification.

Part I 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. Part II contains 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 part II 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 about the SCOPE 3.4 operating system that may be useful to a systems programmer.

<u>Control Data Publication</u>	<u>Publication Number</u>
SCOPE Version 3.4 Installation Handbook	60307400
SCOPE Version 3.4 Reference Manual	60307200
SCOPE Version 3.4 Operator's Guide	60327300
SCOPE Version 3.4 Diagnostic Handbook	60386400
SCOPE Version 3.4 Enhanced Station Operator's Reference Manual	60343800
CYBER Common Utilities Reference Manual	60493300
UPDATE Reference Manual	60342500
CYBER Record Manager Reference Manual	60307300
CYBER LOADER Reference Manual	60344200

The SCOPE 3.4 Internal Maintenance Specifications are available on listable magnetic tape.

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 PART I

1	SCOPE 3.4 INTRODUCTION	1-1
	Hardware Characteristics	1-1
	Extended Core Storage	1-2
	Software Elements	1-3
	SCOPE Organization	1-3
	SCOPE System Tape	1-7
2	CENTRAL PROCESSOR AND SCOPE	2-1
	CM Organization	2-1
	Control Point Concept	2-1
	CP-System Communication	2-4
	CP-PP Communication	2-5
	Program Recall	2-6
	Central Memory Resident	2-8
	Summary of Central Memory Resident Areas	2-9
	CMR Segmentation for ECS System	2-13
	Core Layout	2-13
	Segment Loading	2-14
	ECS System Image	2-17
	ECS Error Recovery	2-18
	System Control Point	2-19
3	PERIPHERAL PROCESSORS AND SCOPE	3-1
	Peripheral Processor Organization	3-1
	PP Communications	3-4
	PP Resident	3-5
	Field Access Flag Usage	3-11
	SCOPE System Monitor	3-12
	Monitor Functions (CPMTR)	3-16
	Memory Allocation (PPMTR)	3-22
4	FILES AND FILE TABLES	4-1
	Files	4-1
	System Files	4-1
	Local Files	4-2
	Permanent Files	4-2
	Queue Files	4-3
	ACQUIRE Macro	4-9
	VERIFY Macro	4-16.3
	SCOPE I/O Tables	4-17
	File Tables	4-17
	Device Tables	4-21
	Tape Drive Scheduling	4-25
	RMS Set Terminology	4-27
	Device Sets	4-27
	RMS Tables	4-29

5	INPUT/OUTPUT	5-1
	I/O Philosophy	5-1
	CIO	5-1
	Allocatable Device I/O	5-23
	Stack Processor	5-27
	Stack Request Formats	5-28
	Dismountable Pack Processing – I/O Detail	5-35
	ECS-Buffered I/O	5-40
6	PERMANENT FILES	6-1
	Permanent File Functions	6-1
	Macro Requests	6-2
	Macro Request Calls	6-5
	CATALOG Function	6-9
	SAVEPF Function	6-10
	ATTACH Function	6-11
	GETPF Function	6-13
	ALTER Function	6-13
	RENAME Function	6-13
	EXTEND Function	6-14
	PURGE Function	6-15
	PERM Function	6-16
	Permanent File Utility Routines	6-17
	DUMPF	6-18
	LOADPF	6-23
	TRANSPF	6-25
	Auditing Permanent Files	6-30
	Permanent Files/System Interface	6-33
	Permanent File Interlocks	6-33
	Permanent File Tables	6-35
	Private Device Set Processing	6-37
	Device Set Creation and Maintenance	6-37
	Using Device Sets	6-44
7	JOB PROCESSING	7-1
	Job Flow	7-1
	Job Input Queue	7-1
	JANUS	7-3
	Integrated Scheduler	7-4
	Job Scheduling	7-7
	Job Control Area	7-9
	Job Descriptor Table	7-10
	Job Scheduling Queues	7-11
	Job Advancing	7-12
	Control Card Processing	7-13
	Job Termination	7-15
	Normal Termination	7-15
	Abnormal Termination	7-18
	Job Post-processing Utilities	7-19
	Job Control with Logical Identifiers	7-23

8	EDITLIB	8-1
	Introduction	8-1
	System EDITLIB	8-7
	EDITLIB Files	8-7
	Control Cards	8-14.1
	Directives	8-14
	Files	8-20
	System Security	8-22
	MDI (Move System Directory)	8-22
	Table Formats	8-23
	Directory/Library/Program Limits	8-23
9	SYSTEM BULLETIN UTILITY	9-1
	System Bulletin File (BULLUP Card)	9-1
	BULLUP Data Cards	9-1
	Processing Data Cards	9-2
	Creating a System Bulletin File	9-3
	Updating a System Bulletin File	9-3
	Reducing a System Bulletin File	9-4
	SCOPE/INTERCOM Considerations	9-4
10	LDCMR	10-1
	Introduction	10-1
	LDCMR Control Card	10-1
	LDCMR Files	10-4
	System Security	10-5
	LDCMR Interlock	10-5
	LDC	10-5
	Reserved Names	10-6
	Sample Jobs	10-6
	LDCMR Error Messages	10-7
	Non-Fatal Messages	10-7
	Fatal Messages	10-9
11	SYSTEM DYNAMIC DUMP	11-1
	Interface	11-1
	Dynamic Dump File	11-1
	Listing System Dynamic Dump Files	11-2

APPENDIXES

A	STANDARD SCOPE CHARACTER SETS	A-1
B	SCOPE SYMBOL DEFINITION	B-1

ECS partitioning restructure

844 spare drive support

Segmented CMR for ECS systems

JANUS support of 580 line printers

Support of symmetric links between CYBER 70/Models 72, 73, and 74

844 shared controller/drive support

Support of shared 844 devices in a dual-mainframe system

Implementation of public as well as private device sets

System Control Point

SCOPE/INTERCOM Front End

New features added with this revision to SCOPE 3.4.4 include:

844 factory format support

Restructure of the handling of RMS stack requests

Enhancements to job management and system control point capabilities

On-line Maintenance Software

Modification of 881 disk pack reformatting utility

INTERCOM Restart

Enhanced Station performance improvement

Numeric values given in this document, unless otherwise indicated, are assumed to be as follows:

Bit numbers	Decimal
Byte numbers	Decimal
Word addresses	Octal
Field/table lengths	Octal
Block/page sizes	Octal

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 undefined parameters.

CONTENTS PART I

1	SCOPE 3.4 INTRODUCTION	1-1
	Hardware Characteristics	1-1
	Extended Core Storage	1-2
	Software Elements	1-3
	SCOPE Organization	1-3
	SCOPE System Tape	1-7
2	CENTRAL PROCESSOR AND SCOPE	2-1
	CM Organization	2-1
	Control Point Concept	2-1
	CP-System Communication	2-4
	CP-PP Communication	2-5
	Program Recall	2-6
	Central Memory Resident	2-8
	Summary of Central Memory Resident Areas	2-9
	CMR Segmentation for ECS System	2-13
	Core Layout	2-13
	Segment Loading	2-14
	ECS System Image	2-17
	ECS Error Recovery	2-18
	System Control Point	2-19
3	PERIPHERAL PROCESSORS AND SCOPE	3-1
	Peripheral Processor Organization	3-1
	PP Communications	3-4
	PP Resident	3-5
	SCOPE System Monitor	3-12
	Monitor Functions (CPMTR)	3-16
	Memory Allocation (PPMTR)	3-22
4	FILES AND FILE TABLES	4-1
	Files	4-1
	System Files	4-1
	Local Files	4-2
	Permanent Files	4-2
	Queue Files	4-3
	ACQUIRE Macro	4-9
	VERIFY Macro	4-16.3
	SCOPE I/O Tables	4-17
	File Tables	4-17
	Device Tables	4-21
	Tape Drive Scheduling	4-25
	RMS Set Terminology	4-27
	Device Sets	4-27
	RMS Tables	4-29

5	INPUT/OUTPUT	5-1
	I/O Philosophy	5-1
	CIO	5-1
	Allocatable Device I/O	5-23
	Stack Processor	5-27
	Stack Request Formats	5-28
	Dismountable Pack Processing – I/O Detail	5-35
	ECS-Buffered I/O	5-40
6	PERMANENT FILES	6-1
	Permanent File Functions	6-1
	Macro Requests	6-2
	Macro Request Calls	6-5
	CATALOG Function	6-9
	SAVEPF Function	6-10
	ATTACH Function	6-11
	GETPF Function	6-13
	ALTER Function	6-13
	RENAME Function	6-13
	EXTEND Function	6-14
	PURGE Function	6-15
	PERM Function	6-16
	Permanent File Utility Routines	6-17
	DUMPF	6-18
	LOADPF	6-23
	TRANSPF	6-25
	Auditing Permanent Files	6-30
	Permanent Files/System Interface	6-33
	Permanent File Interlocks	6-33
	Permanent File Tables	6-35
	Private Device Set Processing	6-37
	Device Set Creation and Maintenance	6-37
	Using Device Sets	6-44
7	JOB PROCESSING	7-1
	Job Flow	7-1
	Job Input Queue	7-1
	JANUS	7-3
	Integrated Scheduler	7-4
	Job Scheduling	7-7
	Job Control Area	7-9
	Job Descriptor Table	7-10
	Job Scheduling Queues	7-11
	Job Advancing	7-12
	Control Card Processing	7-13
	Job Termination	7-16
	Normal Termination	7-16
	Abnormal Termination	7-18
	Job Post-processing Utilities	7-19
	Job Control with Logical Identifiers	7-23

8	EDITLIB	8-1
	Introduction	8-1
	System EDITLIB	8-7
	EDITLIB Files	8-7
	Control Cards	8-14.1
	Directives	8-14
	Files	8-20
	System Security	8-22
	MDI (Move System Directory)	8-22
	Table Formats	8-23
	Directory/Library/Program Limits	8-23
	EDITLIB Errors	8-34
9	SYSTEM BULLETIN UTILITY	9-1
	System Bulletin File (BULLUP Card)	9-1
	BULLUP Data Cards	9-1
	Processing Data Cards	9-2
	Creating a System Bulletin File	9-3
	Updating a System Bulletin File	9-3
	Reducing a System Bulletin File	9-4
	SCOPE/INTERCOM Considerations	9-4
10	LDCMR	10-1
	Introduction	10-1
	LDCMR Control Card	10-1
	LDCMR Files	10-4
	System Security	10-5
	LDCMR Interlock	10-5
	LDC	10-5
	Reserved Names	10-6
	Sample Jobs	10-6
	LDCMR Error Messages	10-7
	Non-Fatal Messages	10-7
	Fatal Messages	10-9
11	SYSTEM DYNAMIC DUMP	11-1
	Interface	11-1
	Dynamic Dump File	11-1
	Listing System Dynamic Dump Files	11-2

APPENDIXES

A	STANDARD SCOPE CHARACTER SETS	A-1
B	SCOPE SYMBOL DEFINITION	B-1

SCOPE 3.4 INTRODUCTION

1

The operating system for CONTROL DATA® CYBER 70/Models 72, 73 and 74, and the 6000 Series computers provides Supervisory Control of Program Execution (SCOPE). The SCOPE operating system controls the use of CONTROL DATA CYBER 70/Models 72, 73 and 74, and 6000 Series computer hardware. Therefore, SCOPE is in control of the computer. SCOPE accepts input in the form of jobs submitted by users and processes them as directed by control cards accompanying each job as well as by keyboard commands input by the operator.

Efficient processing of user's jobs is the prime objective of the operating system. This section describes the inherent hardware characteristics, the basic software elements, and how they work together to accomplish the prime objective.

HARDWARE CHARACTERISTICS

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

CENTRAL PROCESSOR

The CPU is designed to perform tasks of a computational nature; it has no input/output capability. It communicates with other system components through the central memory. Under SCOPE, the CPU is used almost exclusively for program compilations, assemblies, and executions. The CPU makes system requests through a CPU request register located at the reference address plus one (RA + 1) of the current program in execution.

PERIPHERAL PROCESSORS

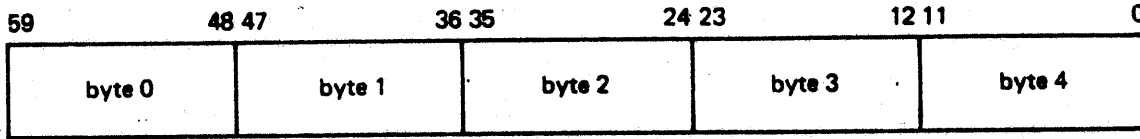
The peripheral processors, of which there may be up to 20 (identified as PP0, PP1, . . . PPn) are identical; they perform many tasks for requesting programs in central memory.

A PP can be assigned to control, input/output, job scheduling, control card interpreting, system housekeeping 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 central memory through which communications with the system are conducted. Each block contains an input register, an output register, and a message buffer. Peripheral processors are discussed in section 3.

CENTRAL MEMORY

Central memory 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:



One or more user programs may be in some state of execution concurrently under SCOPE. These programs are stored in central memory in an assigned user area; a set of system components necessary for the operation of the system is also stored in central memory, forming the central memory resident (CMR) and the record block table (RBT) areas. Central memory is accessible by all PPU's and CPU's and 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 of this manual; the RBT is discussed in section 4.

EXTENDED CORE STORAGE

FEATURES

Under SCOPE 3.4, Extended Core Storage (ECS) is divided into three areas: the System Area, the Dynamic Area, and the Direct Access area. They function as follows:

System Area	Contains system pointers and tables required by ECS software
Dynamic Area	Paged area; contains buffers, library programs, swap files and other files assigned to ECS
Direct Access Area	Assigned to user as result of job card request; used and managed by requestor. Also contains the ECS segment library in ECS control point zero 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 CP code

ECS PAGING

The basic element of ECS paging is a PRU made up of 100B CM words. A group of eight consecutive PRUs (1000B CM words) forms a page. I/O buffering through ECS is made possible by the paging of ECS. A deadstart installation parameter defines the buffer size. 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 usage of ECS through dynamic allocation/deallocation of space

- Availability of ECS to more users by allowing the use of an "over-commitment" algorithm for ECS

SOFTWARE ELEMENTS

Two elements are basic to the SCOPE operating system: files and control points.

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 terminated by an end-of-file indicator. Under the SCOPE operating system, the jobs it processes 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. The control point concept is discussed in section 2.

SCOPE ORGANIZATION

The SCOPE 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 in the library file are in source language form. Installation options are provided to permit flexible selection of system features during the assembly and creation of a deadstart file on tape.

A system monitor is in complete supervisory control of the hardware system. The system monitor is made up of PP overlay OV.MTR, which operates in PPO, and CPMTR, which in a disk system is assembled as part of the central memory resident or in an ECS system consists of a number of separate segments.

SYSTEM LOADING

To load the operating system into a CYBER 70 or 6000 series computer, the deadstart tape is mounted on a device and a small bootstrap loader program is set up on the hardware deadstart panel switches. When the deadstart button of the operator's console is activated, the bootstrap program is transferred to and executed in PP0. The bootstrap loader reads the PP0 save program (CEA) from the first record on the deadstart file, executes it, then 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, then 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 and PP2. CMR is read from the deadstart file into CM, and a display shows all deadstart functions and the options that may be selected.

The functions include:

- Type of deadstart to be performed

- Level of system to be used

- Level of processing to be performed on rotating mass storage (RMS) device labels

- Status of permanent files to be used

- Restriction of permanent files to disk pack devices (854 and 841 only)

- Changes to be made to equipment configuration of the system

- Level of initialization of ECS

- Pre-allocation of RMS devices for customer engineering diagnostic programs

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

When processing of deadstart options is completed, control is passed to MTR and DSD. The system deadstart tape will be rewound to its load point, and it will not be referenced again during operation unless another deadstart is necessary. The tape may be removed and the tape unit cleared for use in other operations. At this point the system is ready to process jobs (see section 7).

This page has been deleted

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

1. 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.
2. The central memory resident (CMR), loaded into the low end of central memory. A set of tables in CMR contain information about the system. Some of the tables are used by the system PP programs to communicate with each other. A record block table (RBT) is built in the upper end of CM.
3. Also, some programs in the system are stored in central memory in an area immediately above the 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 space used to store library programs cannot be used to run user programs; therefore, only the most necessary and frequently used library programs are stored there. For the same reason, CMR is kept as small as possible.
4. The system monitor program (MTR in PPO and CPMTR in CMR) remains in overall control of the system. It controls allocation of system physical resources (CM, ECS, channels, equipments, PPs and CPUs). It handles all communications between user programs and the system and coordinates activities of the other PPU's.
5. PP1 contains the operator console display driver program, DSD. DSD provides a communication path between system and operator. Current system status is displayed by DSD on the two screens in the operator console. The operator can control system operation by typing commands on the console keyboard.
6. Each remaining PP contains 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 is responsible for reading the input register and loading PP overlay programs into its PP as assigned, and for providing communication between the overlays and MTR.

SCOPE SYSTEM TAPE

The released SCOPE 3.4 system (deadstart) tape consists of:

Name	Description
CEA	PP0 save program
CE Diagnostics	See next page
CED	Deadstart PP control program -- resides in PP0
TDR	Read driver for 60x, 65x tape drives -- in PP1
MDR	Read driver for 66x tape drives -- in PP1 and PP3
D	Deadstart dump control program, handles options -- in PP0
DMT	Dump magnetic tape driver for 60x, 65x tape drives -- in PP0
DTS	Dump magnetic tape driver for 66x tape drives -- in PP0
CMR	Central memory resident (up to 8 copies)
COM	Deadstart option matrix generator -- in PP2
IRP	Deadstart RMS driver control program -- in PP2
SCP	Deadstart 6603-I driver
5CQ	Deadstart 6638 driver
5CS	Deadstart 854 driver
5CT	Deadstart 6603-II driver
5CV	Deadstart 821 driver
5CW	Deadstart 841 driver
5CY	Deadstart 844 disk subsystem driver
P	Pre-address 6603-II program
OSY	844 Buffer controlware
OMT	MTS controlware
IRCP	Deadstart main central processor program
STL	Deadstart system initiation program (PP resident)
MTR	System monitor program

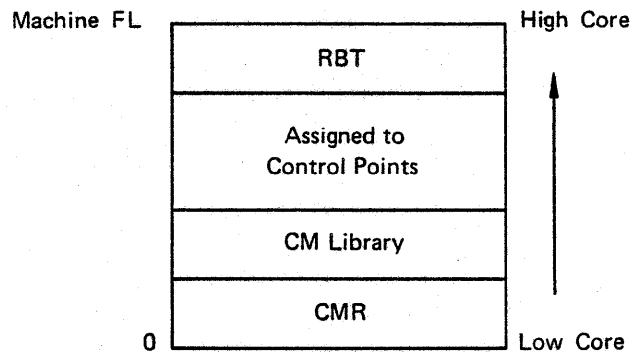
DSD	Display control program
Directory	Library name table
	PP name table
	PP programs — the first must be stack processor's segment
System	Entry point table
	External reference list
	External reference table
Libraries	Program number table
	Program name table
	CP/PP routines

Any installation may expand the above records in two ways:

1. By installing CE Diagnostics. The deadstart diagnostic sequencer, CES, will be placed after CEA, followed by diagnostic routines selected for the site (CUI, ALS, FST, etc.), followed by routine CED.
2. By placing up to seven additional CMR records on the system tape for different equipment configuration.

CM ORGANIZATION

The allocation of central memory is illustrated below:



Low core is allocated to the central memory resident portion of SCOPE, 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.

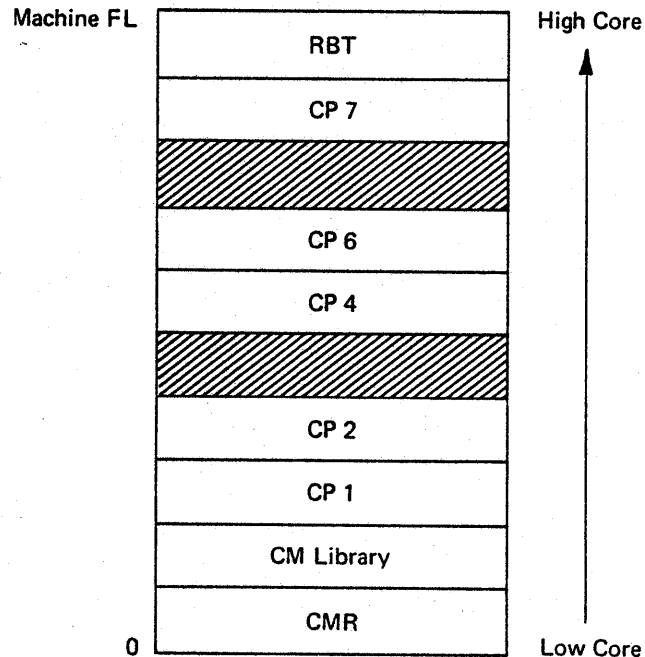
CONTROL POINT CONCEPT

Blocks of central memory storage not allocated for system use are ordered by control point number and assigned to jobs. Each control point number 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 do appear in the system.

Under SCOPE 3.4, any number of control points up to 15 are possible. In the released system, the default value of N.CP is 15 decimal. 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 one time. Once a job is assigned to a control point, system resources such as central memory, 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 core storage block assigned to the job at control point 2 is higher than the block for the job at control point 1, and storage for control point 3 is always higher than that for control point 2, and so on.

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



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

Control point zero is used to identify 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 number is entered into the system table entry for that equipment.

If not assigned to a job, the equipment is assigned to control point zero and is available to be assigned to a job. All active system files are attached to control point zero. 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 monitor (MTR) when it runs a central processor (CP) system program. For example, if MTR needs to move the storage assigned to a control point, it asks CPMTR to initiate the CP storage move program in central memory resident. Since the CP storage move program is not associated with any specific job running at a control point, it is assigned to control point $n+1$. There is no control point area in CMR for control point $n+1$.

JOB DESCRIPTION NUMBER

During the course of 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 back in it is probably associated with a control point other than the control point during its original assignment.

During the time 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 is used to identify the job during its entire execution.

In order to clarify the difference between job descriptor number and control point number, JDT numbers start at $n + 1$ where n is the number of control points.

STORAGE MOVES

Since jobs come and go as they finish processing and new jobs begin, or as jobs are swapped in and out, CM storage must be reallocated and jobs must be moved. If a job at a control point requests additional storage, it may be necessary to move jobs to obtain the required storage. MTR keeps a tally of unassigned CM in a CMR word T.UAS. Storage associated with each control point is of two types, either of which 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 merely 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, will result in a transfer of unallocated storage to allocated; no storage move will take 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 will be scanned. This scan locates the combination of unallocated storage blocks which will result in a move of the least amount of storage.

In the diagram shown under Control Point Concept, if control point 1 needs more storage, it will be 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 will be moved downward to provide the storage. Added storage always extends the field length upward.

STORAGE MOVE EXAMPLE:

Control point 5 requests an FL of 300 (all values are increments of 100 octal).

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

If MTR takes the UAS from control point 7, the 150 units of central memory at control point 6 would have to be moved; however, taking UAS from 3 and 4 requires moving 120 units of central memory at control point 4 and 5. (20 units are moved from 4 to 3 and 100 units are moved from 5 and added on behind the 20 units moved to 3.)

CP - SYSTEM COMMUNICATION

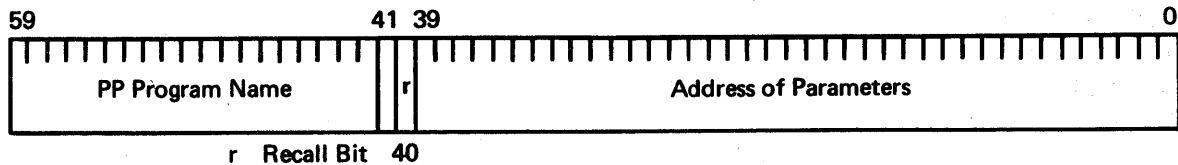
A running CP program must communicate with the system in the following situations:

1. When a CP program is loaded and executed as a result of a control card call, the system must place any parameters specified on the control card in an area where they can be read by the CP program.
2. No CP instructions allow a CP program to perform input/output; therefore a CP must send a request to the system, to load a PP program to execute the input/output.
3. When a CP program terminates, it must advise the system that it may process the next control card.

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 101B locations of each job's field length are reserved for this purpose. The following 10B words are reserved for the loader table. The first program loaded into a user field length is always loaded at location RA+111B because the reserved words are RA+0 through RA+110B.

The RA Communication Area is shown in Part II, Section 2.

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 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:



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 card, they are placed in locations RA + 2 through RA + 63 by 1AJ, which stores the total number of parameters in location RA + 64. When the routine or file indicated on the control card executes, it finds in these locations the information needed to direct execution.

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 36 through 39 of the word, and clear bit 41. If the central exchange jump (CEJ) installation is available, the user's program should use it immediately after placing a call in RA + 1. This will cause CPMTR to begin execution immediately. If CPMTR determines that the RA + 1 call should be assigned to a PP it will pass the call on to MTR.

When a PP is available, MTR will write the word into its PP input register, in CMR. Figure 3-2 shows the format of a PP input register for a transient program called from a CP program. The name, the auto-recall bit, and any parameters in bits zero through 35 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 will find the relative address of the file environment table (FET) to be 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 users program does not use the central exchange jump, or the instruction is not available. When a 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 is used to communicate to the user program if the central exchange jump is available. If the hardware for this instruction is available the bit is set on.

PROGRAM RECALL

The recall program status is provided in SCOPE to enable efficient use of the central processor and to capitalize on the multiprogramming capability of SCOPE. 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 were placed in a loop to await I/O completion, a CP program can ask SCOPE to put the control point into recall status until a later time; and the CPU may be assigned to execute a program at some other control point. The job itself 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. SCOPE will not return control until the specific request has been satisfied. Periodic recall can be used when the program is waiting for any one of several requests to be completed. The program will be activated periodically, so that it can determine which request has been satisfied and whether or not it can proceed.

PERIODIC RECALL

To enter periodic recall, a CP program puts the characters RCL left-justified into RA+1. On encountering the RCL request, CPMTR examines the auto-recall bit (bit 40); if it set, the request is considered to be an auto-recall request. If it is not set, CPMTR checks bits 0-10 (decimal) for a delay count. If this delay count is not set, CPMTR specifies a default value for it. 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 completion bit in the FET to see if the I/O is finished. If so, the CP program may proceed with computations. If I/O is not complete, the CP program will put itself back into recall.

AUTOMATIC RECALL

If a CP program makes a request in RA + 1 and bit 40 of RA + 1 is set to one, the control point will be put into automatic recall after the request has been initiated. Again, the CPU is assigned to another control point as in periodic recall. In this case, however, the program in recall will be 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 most 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 will be put into auto-recall. If bit 40 is set, bits zero through 17 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 will set the completion bit (low order bit) in the reply word. When the completion bit is set, MTR will restart the program.

For a call to CIO, the reply word is the first word of an FET. For other programs the reply word need not be part of an 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 put itself into auto-recall without calling a PP program by putting RCL left justified in RA + 1 and setting bit 40 of RA + 1 to one. Bits zero through 17 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. Figure 2-1 shows 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. 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. He will issue all the requests without recall (using a separate FET for each request); then go into 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 are still incomplete.

Periodic recall may be used also when a CP program can initiate an I/O request and then perform some computation. In some cases, the I/O would be completed before the computation; in others, the computation would be done first. The user would go into recall only when computation was done, 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 PRU's have been processed.

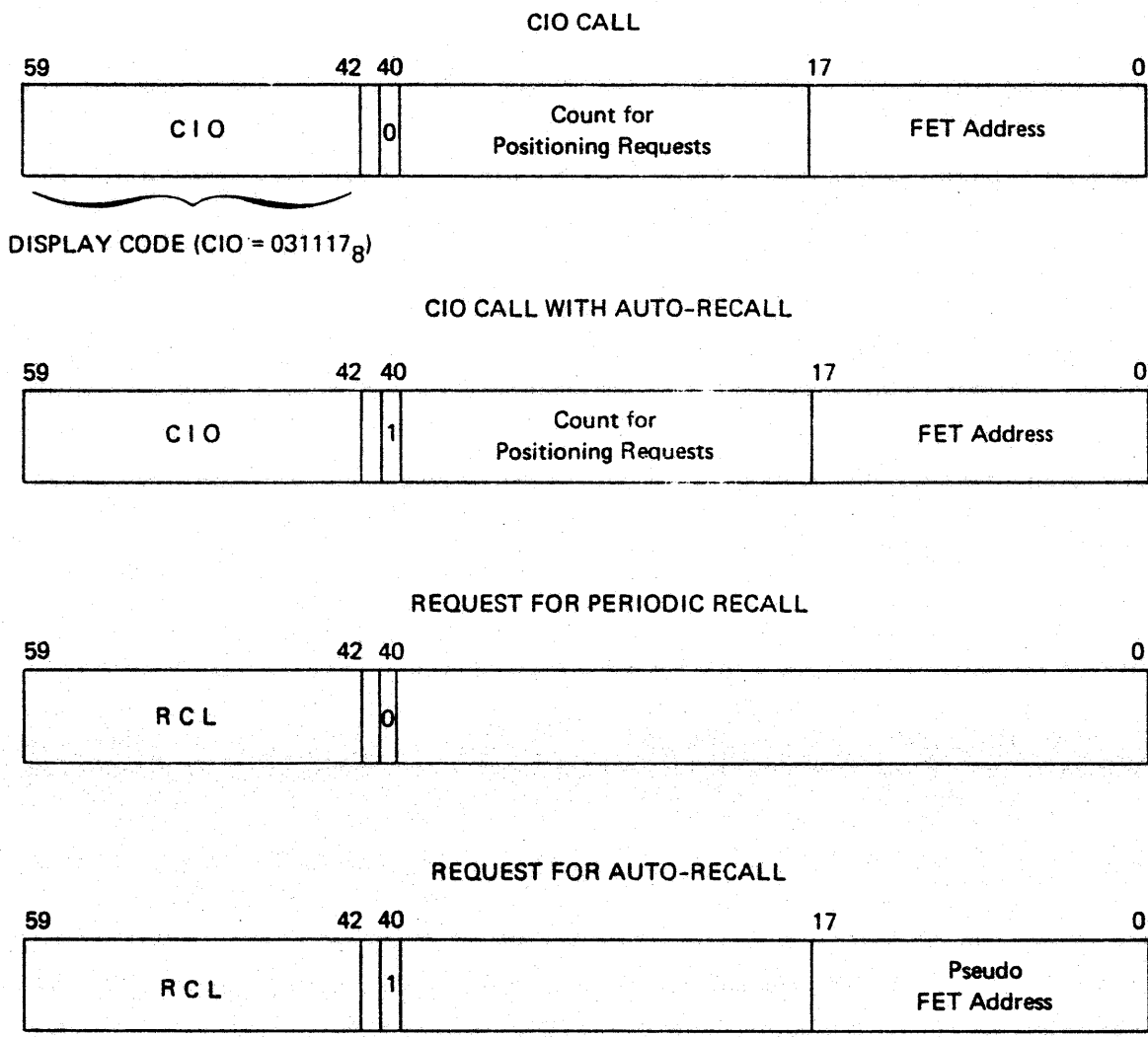


Figure 2-1. Call Formats

CENTRAL MEMORY RESIDENT

The low end of core storage is reserved for the central memory resident portion of SCOPE and the system library portions which reside in central memory. CMR contains pointers, tables and programs. Its length is dependent upon several factors, including the number of peripheral processors and the number of control points, which determine the number of tables in CMR and the length of certain tables. Some CMR tables are optional and may appear only by installation parameter. Figure 2-2 illustrates a typical CMR.

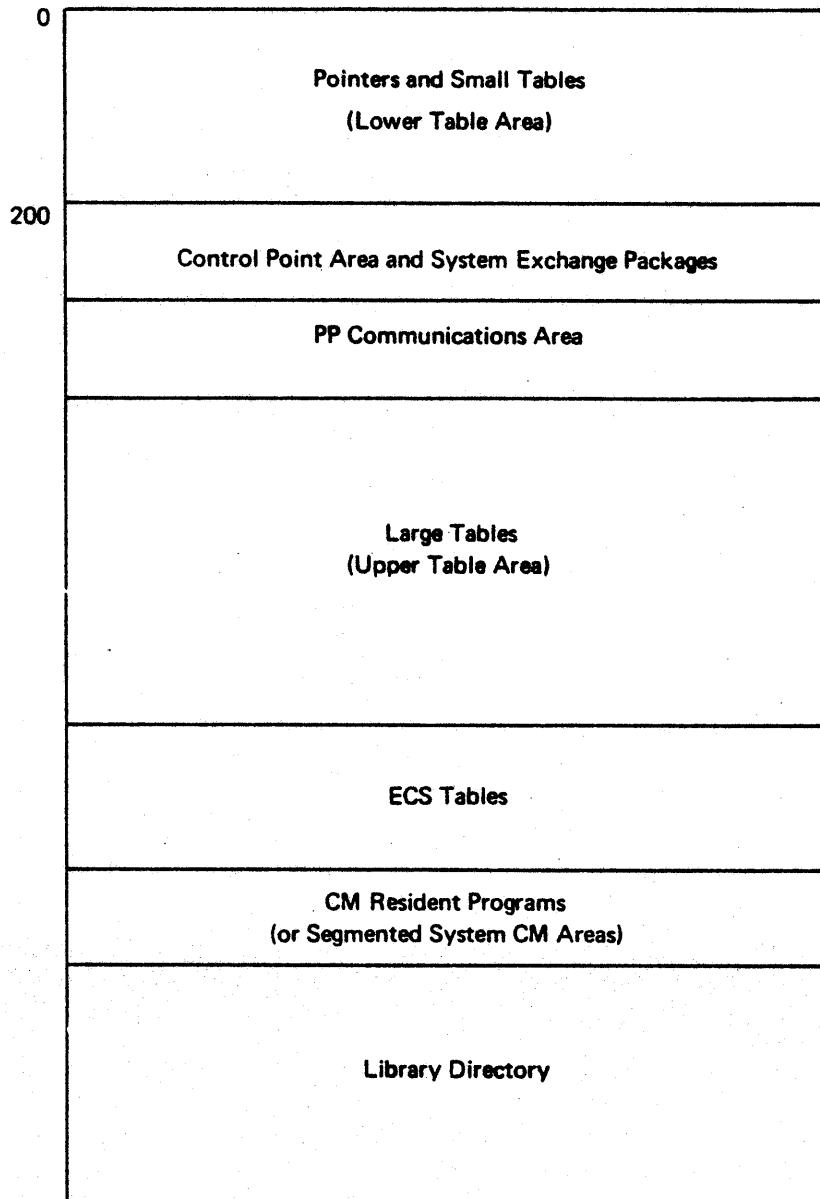


Figure 2-2. Typical CMR Assignments

SUMMARY OF CENTRAL MEMORY RESIDENT AREAS

Lower Table Area Contains pointers to larger tables in the upper table area of CMR, along with various flags, constants, and installation option 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 central memory.

Control Point Area 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 Area Contains eight words for each peripheral processor 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 6-word message buffer.

Upper Table Area Contains major tables pertinent to system and job operation.

ECS Tables Area Contains buffers for transferring PP overlays and RMS files.

CM Resident Program Area 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 CM resident program area is overlaid by the ECS system resident and by the CP code overlay segments.

Library Directory Contains tables related to the system libraries, including library name table, PP program name table and the CM resident library programs.

CENTRAL MEMORY RESIDENT

0		Pointers																																																																	
100	T.CST	Channel Status Table																																																																	
154		PP Status Words																																																																	
200	T.CPA _n	Control Point Areas																																																																	
	T.XPIDLA	System Exchange Packages																																																																	
	T.PPC1	PP Communication Areas																																																																	
	* T.EST	Equipment Status Table																																																																	
	* T.FNT	File Name Table																																																																	
		CIO-CPCIO Special FNTs																																																																	
		Permanent File FNTs																																																																	
	* T.ITABL	INTERCOM Table																																																																	
	* T.DAT	Device Activity Table																																																																	
	* T.RMSBUF	RMS Buffer																																																																	
	* T.STG	Tape Staging Table																																																																	
	* T.APF	Attached Permanent File Table																																																																	
	‡ T.RQS	Request Stack																																																																	
	T.RST	Request Scheduling Table (RMS)																																																																	
	T.RBR	Record Block Reservation Table (Headers)																																																																	
	T.RBRBIT	RBR Bit Table																																																																	
	T.DST	Device Status Table																																																																	
	T.TRB	Trace Buffer																																																																	
	T.SEQ	Sequencer Table																																																																	
	T.INS	Installation Area																																																																	
	T.MST	Mounted Set Table																																																																	
	T.DDT	Dismounted Device Table																																																																	
	T.VRNBUF	VSN Buffer																																																																	
	T.TAPES	Tapes Table																																																																	
	T.MAIL	Scheduler Mailbox Buffer																																																																	
	T.IDT	Logical ID Table																																																																	
	T.DFB	Dayfile Buffers																																																																	
	T.PJT	Parameter Storage for Delayed PP Jobs																																																																	
	T.SSCT	Subsystem Control Table																																																																	
	T.SCHPT	(Optional) Scheduler Statistics																																																																	
	T.SCHJCA	Scheduler Job Control Area																																																																	
	T.SCHJDT	Scheduler Job Descriptor Table																																																																	
	T.BCFAP	CPMTR CEFAP Buffer																																																																	
			<table border="0" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 10%;"></td> <td style="width: 15%;"></td> <td style="width: 40%;">T.PPOVL</td> <td style="width: 45%;">PP Resident Overlay Save Buffer</td> </tr> <tr> <td></td> <td></td> <td>T.BRKPT</td> <td>Breakpoint Table (ECS System)</td> </tr> <tr> <td></td> <td></td> <td>T.AREA</td> <td>Area Table (ECS System)</td> </tr> <tr> <td></td> <td></td> <td>T.ENTRY</td> <td>Entry Table (ECS System)</td> </tr> <tr> <td></td> <td></td> <td>T.DOT</td> <td>Device Overflow Table</td> </tr> <tr> <td></td> <td></td> <td></td> <td>CM Resident Programs (Disk System)</td> </tr> <tr> <td></td> <td></td> <td></td> <td>Segmented System Areas (ECS System)</td> </tr> <tr> <td></td> <td></td> <td>T.EPAGE</td> <td>Empty Page Stack</td> </tr> <tr> <td></td> <td></td> <td>T.ECSPRM</td> <td>ECS Parameters</td> </tr> <tr> <td></td> <td></td> <td>T.LIB</td> <td>Library Directory</td> </tr> <tr> <td></td> <td></td> <td></td> <td>INTERCOM Pointer Area</td> </tr> <tr> <td></td> <td></td> <td></td> <td>INTERCOM Buffers and User Tables</td> </tr> <tr> <td colspan="4" style="text-align: center;">-----</td> </tr> <tr> <td colspan="4" style="text-align: center;">(Job control point user field length)</td> </tr> <tr> <td colspan="4" style="text-align: center;">-----</td> </tr> <tr> <td></td> <td></td> <td>T.RBT</td> <td>RBT Chains</td> </tr> </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.DOT	Device Overflow Table				CM Resident Programs (Disk System)				Segmented System Areas (ECS System)			T.EPAGE	Empty Page Stack			T.ECSPRM	ECS Parameters			T.LIB	Library Directory				INTERCOM Pointer Area				INTERCOM Buffers and User Tables	-----				(Job control point user field length)				-----						T.RBT	RBT Chains
		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.DOT	Device Overflow Table																																																																
			CM Resident Programs (Disk System)																																																																
			Segmented System Areas (ECS System)																																																																
		T.EPAGE	Empty Page Stack																																																																
		T.ECSPRM	ECS Parameters																																																																
		T.LIB	Library Directory																																																																
			INTERCOM Pointer Area																																																																
			INTERCOM Buffers and User Tables																																																																

(Job control point user field length)																																																																			

		T.RBT	RBT Chains																																																																

* Table Must Begin Before 10000B

‡ Table Must Begin Before 20000B

SUMMARY OF TABLES IN UPPER TABLE AREA

Equipment Status Table One entry for each device in the system configuration. Non-allocatable devices can be assigned to one control point at a time; allocatable devices may be attached to many control points simultaneously.

File Name Table 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.

INTERCOM Table Provides multiplexer and port definition information for INTERCOM program use. Assembled only by installation parameter IP.INTCM.

Device Activity Table 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 is used to report an error on an RMS device and ask the operator to idle down the device.

Tape Staging Table Defines availability, assignment, and demand for tape devices.

Attached Permanent File Table 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 3-word entry in this table when a request for its use is active.

Record Block Reservation Table 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 Table Directly related to the request stack; contains a 2-word entry for each allocatable device in the system, plus an additional pseudo-entry for unassigned file processing.

Tapes Table Contains one entry (10B words/entry) for each tape unit defined.

Device Pool Table Contains a 10-word area for each RMS device. Used by associated ISPs to pass their internal pool area to each other.

Sequencer Table Contains 30-word entry for each preallocated RMS device for use for CE diagnostic programs.

Installation Table Reserved for specific needs of installation. Tables are generated in the area only by installation.

Mounted Set Table One entry for each mounted device set, including the public sets.

Dismountable Device Set Entries for each RMS device plus entries for each queuing devices needed by jobs.

ID Table Contains Host ID, Logical IDs, and Physical (link) IDs. The ID table may be zero-length.

Mailbox Used for communications between system and swapped out jobs.

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 zero buffer is at the end of this area.

Peripheral Job Table Contains parameters saved for delayed PP jobs.

Subsystem Control Table Contains names of defined subsystems.

Scheduler Performance Table 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 Table Contains linked entries for each class of job. Entries describe job requirements, current status, accumulated use time of system components, etc. See Job Processing, Chapter 7.

Breakpoint Table 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 Table Eight-word buffer that receives the ECS area table from a segmented system. It describes the system ECS and CM structure.

Entry Table 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.

Library Directory Falls at the end of the CMR upper table area following the CM resident programs and ECS tables; and is of variable length. It contains 2-word entries in the program name table section and 1-word entries in the entry point table. The directory length may expand or contract as programs are added and deleted or as program residence is changed.

CP Resident Programs (in a disk system)

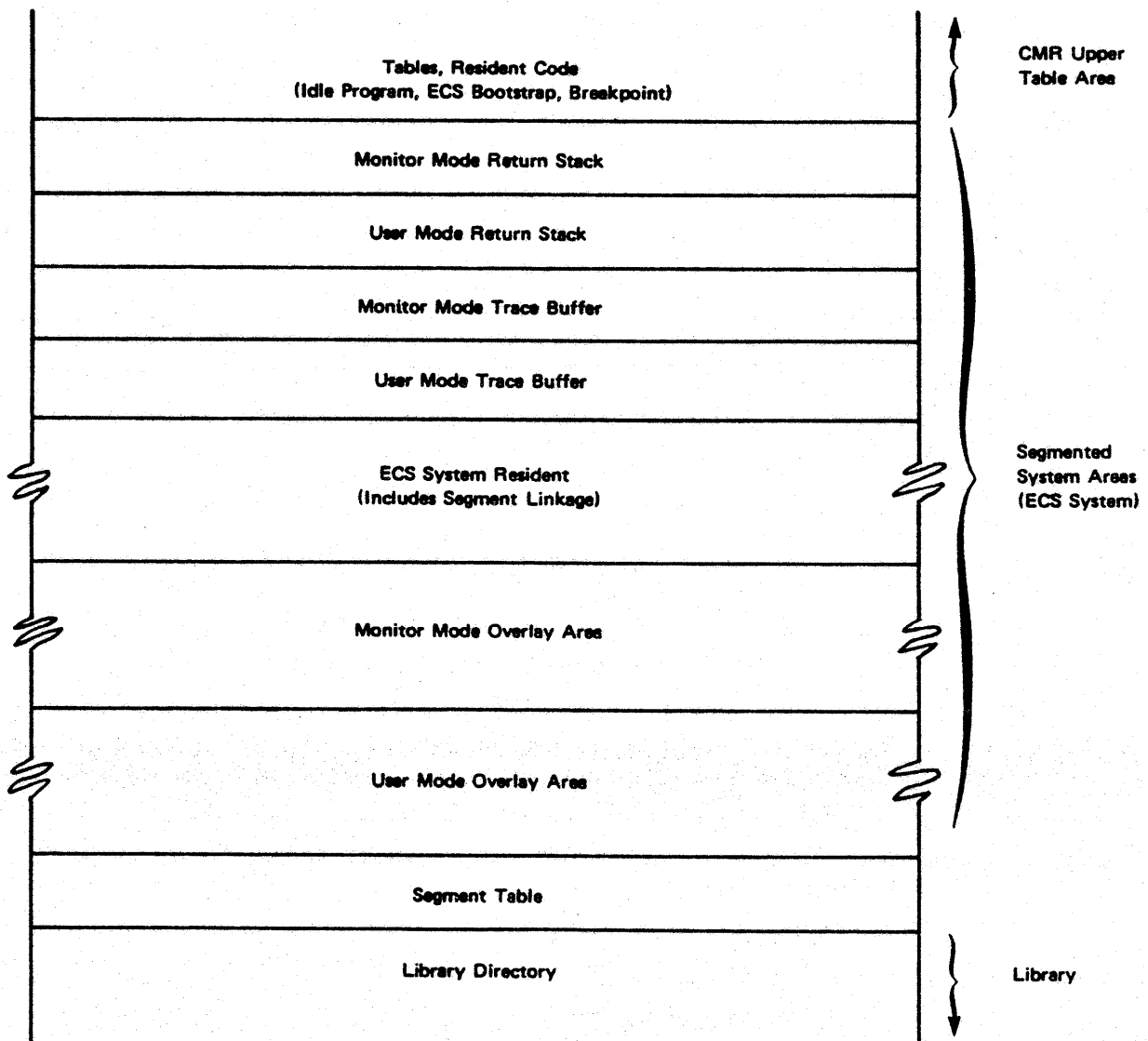
Symbol	Name/Function
CP.MTR	Central processor monitor
CP.SM	Central memory storage move
CP.SPM	Stack processor manager
CP.SCH	Memory manager scheduler

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 in 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 the utility program LDCMR (see section 10).

CORE LAYOUT

For an ECS system, the equivalent of a disk system's system resident programs section in CMR is a number of specialized areas as follows:



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 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 the utility LDCMR to load the ECS system and read the date-time stamp.

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, the Entry table, and the Breakpoint table are detailed in part II, section 1.

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.

GOTO **EPTNAME**

transfers control to the entry point EPTNAME.

GOTO **TABLE,Register**

TABLE GOTOTAB **EPTNAM2**
GOTOTAB **EPTNAM2**

transfers control to the entry point referenced in a GOTOTAB macro in the position indexed by register in TABLE.

RETURN

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

SEGMENT DEFINITION

SEGDEF **SEGMENT,mode,CM**

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

SEGMENT is the segment name.

Mode if USER indicates user mode segment, otherwise monitor mode is assumed.

CM indicates the segment must be CM resident.

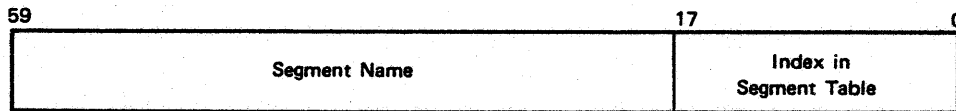
ENDSEG

must be called immediately before the segment END.

The name of the segment also can be the name of an entry point in the segment, but not of a tag. If it is not an entry point, the ENDSEG macro will define 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:

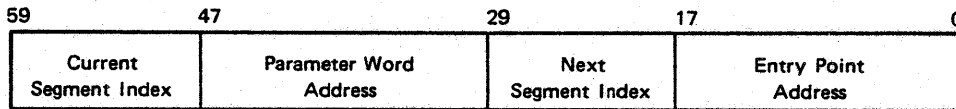


All segments must be reusable serially, otherwise to prevent reuse, they must zero their header word prior to relinquishing control.

PARAMETER WORD

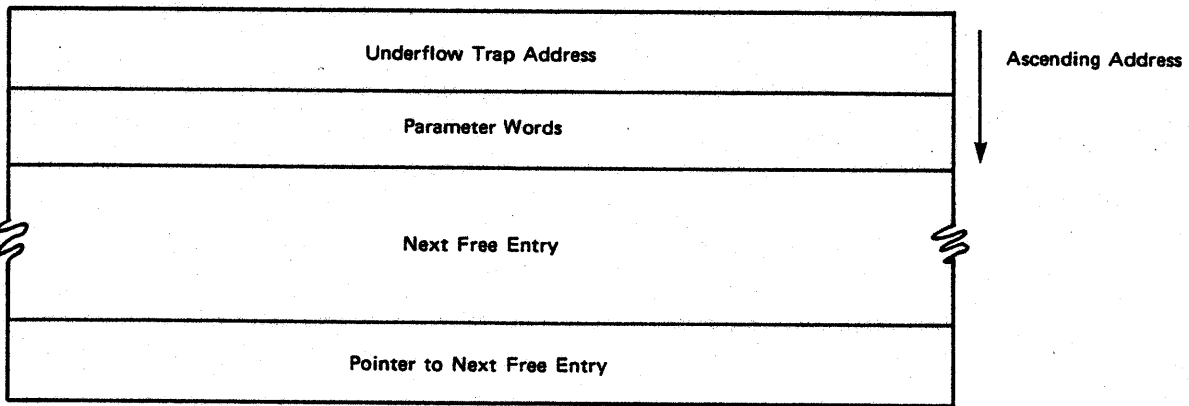
Linkage is done through a parameter word generated by the linkage macros and filled in by LDCMR. For a GOTO or a CALL, A1 is set to the address of the proper parameter word and a jump is made to one of the linkage processor entry points.

Parameter word format:

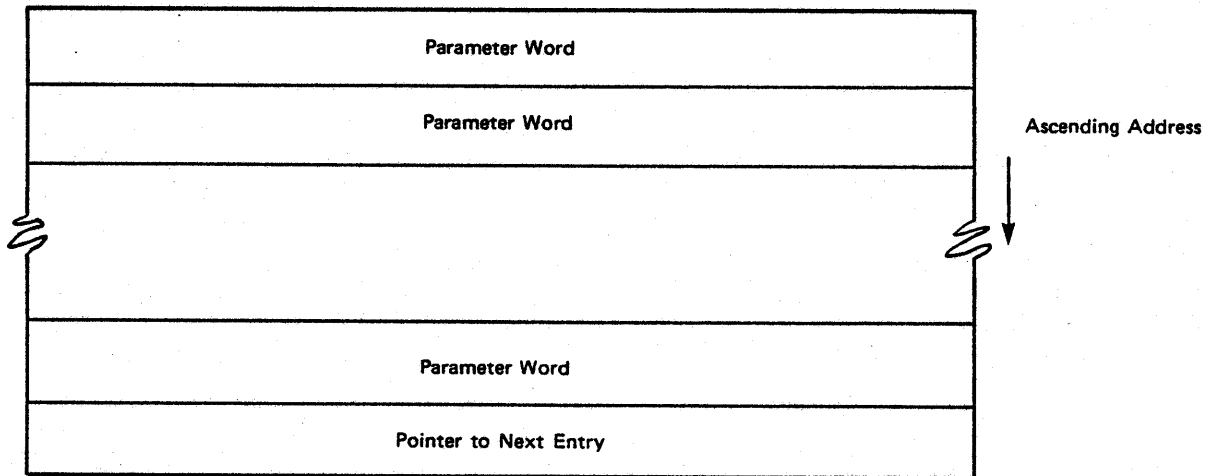


The addresses are absolute. The indexes are in the segment table. The parameter word is transformed into a return descriptor on a CALL by shifting it 30 positions and adding 1; it is stored in the proper return stack. A RETURN loads the last stored descriptor and performs a GOTO on it. If trace buffers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

Return stack format:



Trace buffer format:



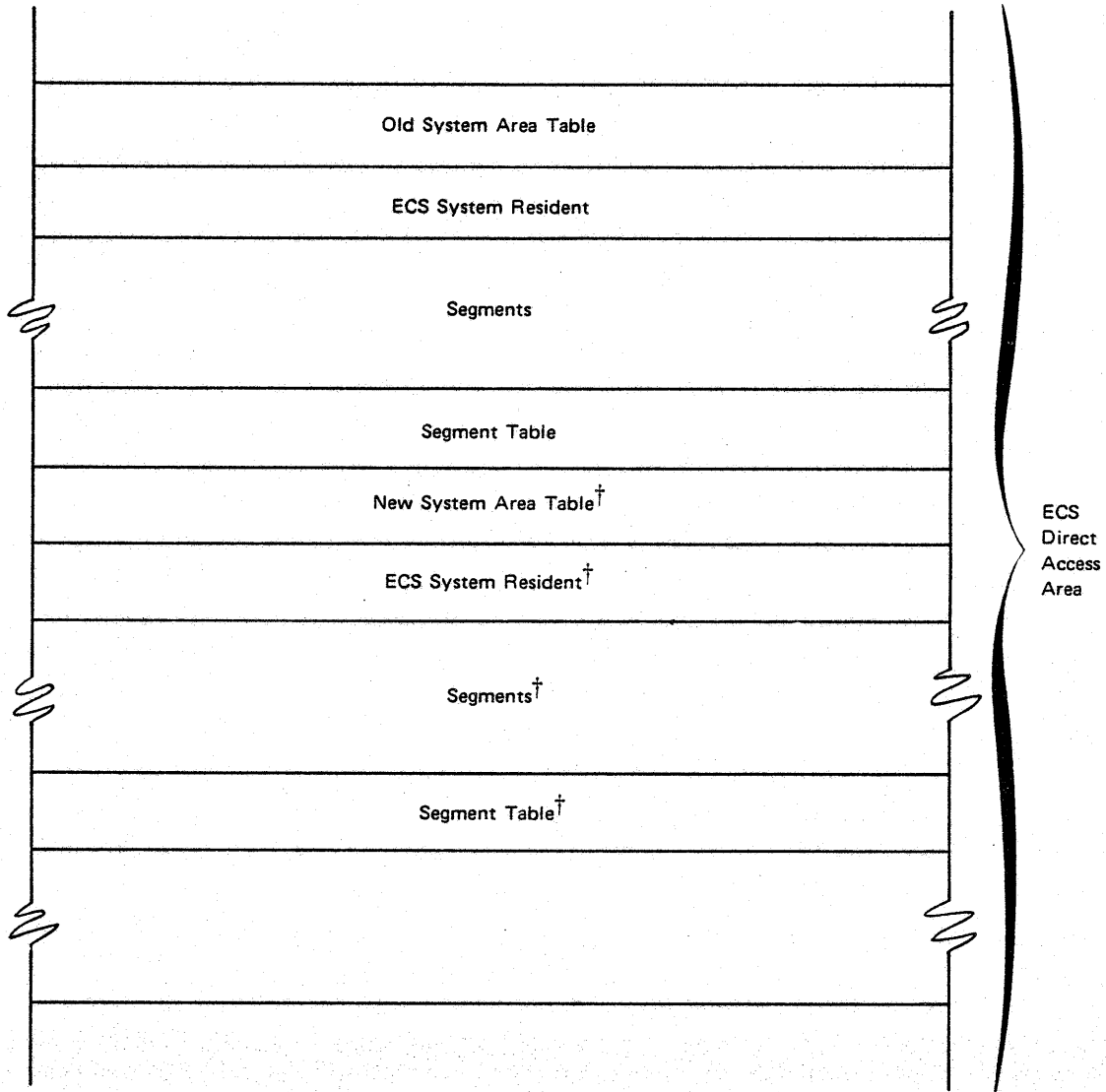
Segment loading is done by using the segment descriptor in the second word of the segment table entry for the segment. The address of the segment table entry for a segment is:

$$\text{Segment table base address} + 2 \times \text{index}$$

The Segment table is detailed in part II, section 1.

ECS SYSTEM IMAGE

The system is written to the ECS direct access area as an extension of control point zero 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 of the following form:

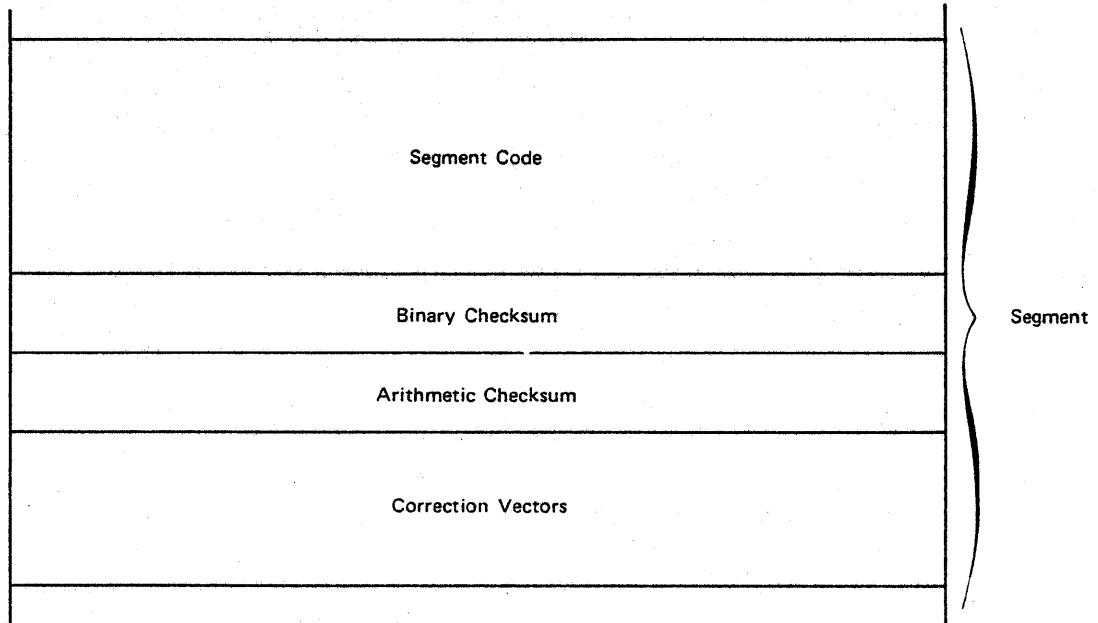


[†]If a terminator replaces the new system area table, the new system sections do not appear.

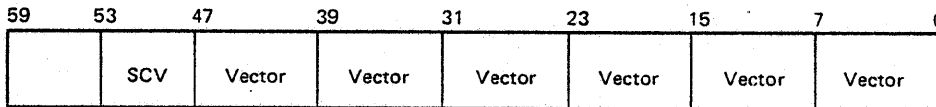
The operating system requires approximately 20K of ECS in the direct access area. For example, if 40K is to be available for user direct access, 60K must be allocated to the direct access partition. If LDCMR is to be used after deadstart, the direct access area must be large enough to contain two operating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

ECS ERROR RECOVERY

Segments are protected against ECS errors in transmission or storage. An auto-corrective hamming code is applied to them to get correction vectors. The correction vectors and two checksums are placed at the end of the segment as shown:

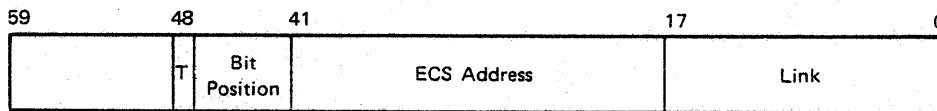


The code is capable of correcting a single bit error every four words. Vectors correct each four words, and a 6-bit secondary correction vector makes vector words self correcting:



Errors are corrected only if they are detected by the ECS parity error mechanism. Detected errors are recorded in a 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 re-applied. 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 = 0 Bit dropped T = 1 Bit picked

SYSTEM CONTROL POINT

A system control point provides a centralized location for a module or group of modules, allowing them to perform functions for one or more jobs at other control points. This facility provides overall reduction of central memory usage. Instead of several jobs having duplicate copies of these specially privileged modules in their field lengths, only one set of these modules needs to occupy central memory. This feature also improves coordination of control and access.

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

A system control point (SCP) is any control point occupied by one of the subsystems. SCP may be used to describe both the control point and the subsystem at the control point.

A user control point (UCP) is any job or module at a control point which makes a request to a system control point. A UCP may be a batch job, INTERCOM job, multi-user 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 two or three hundred words. It should specify a half second periodic recall by using RCL with a 3777B delay period indicated by bits 10-0. The subsystem will then be started immediately when it receives a call.

When a call is received, the following steps must be taken:

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

It is essential that the above steps be taken 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 SCP to make a request and have it successfully passed to the SCP. Both SCPs would then have outstanding SSC calls, preventing both from being swapped out. If the sum of the field lengths of the two UCPs and the required field length of the SCP is greater than what is available, the system will be deadlocked. If this occurs, one of the UCPs must be dropped to resolve the conflict.

CALLSS MACRO

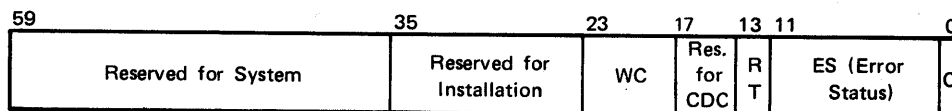
The CALLSS macro is issued by a UCP to request a particular function from a subsystem. A UCP may call more than one subsystem, either serially or in parallel. Also a UCP may make more than one call to an individual subsystem. The contents of registers X1, X2, A1, and A6 are destroyed during execution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

label	CALLSS	ssid,addr,recall
label	is an optional statement label.	
ssid	is a required subsystem code. (This parameter may be a register name.)	
addr	is the address of the parameter block for this request. This parameter is required and must be nonzero. (The parameter may be a register name.) If the address is outside the UCP field length the UCP will be aborted.	
recall	if non-blank, the request will be made with automatic recall. (Processing at the UCP will be suspended until completion of the request.)	

PARAMETER BLOCK

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:



The fields of the first word are defined as follows:

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

- 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 zero.

- Bit 1 0 - subsystem currently running
1 - subsystem not initiated
- Bit 2 0 - subsystem not busy
1 - subsystem busy
- Bit 3 0 - subsystem defined
1 - subsystem not defined
- Bits 4-5 reserved

Bits 6-11- error condition other than those described by bits 1, 2 and 3. Bits 6-11 can assume the following octal values:

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. The meanings of the bits are:

Bit 12 0 - operating system holds the current request until the subsystem is able to accept it. The C bit is not set until the request has been accepted and the processing of this request is finished.

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 non-fatal error conditions. (Fatal errors cause a UCP abort).

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

The remaining bits in this word (14-59) are reserved.

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. It is therefore necessary to re-issue 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) that is to be passed with the request. The maximum is determined by the subsystem but may not exceed 77B.

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 (e.g. refer to RA.SSID not RA+50; to SF.READ not 10).

REQUESTING ACTIVE STATUS

Subsystem notifies the operating system that it is entering active status.

The operating system does not recognize an SCP until a subsystem is loaded and ready to enter SCP status. The subsystem puts its name and code in RA.SSID (RA+50), which 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 validates the following:

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

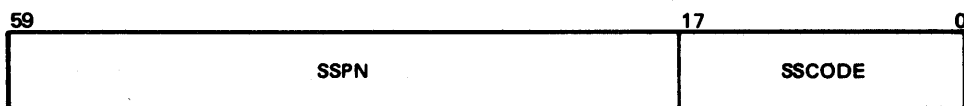
If any of the above conditions are not met, an appropriate diagnostic message is entered in the dayfile and the job is aborted. Once the operating system has assigned SCP status to the subsystem, the C field is set if 1.

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 6-11 of the ES field:

04 - Another control point has SCP status for this subsystem

05 - There are already nine control points with SCP status

The format of SCP word RA.SSID is:



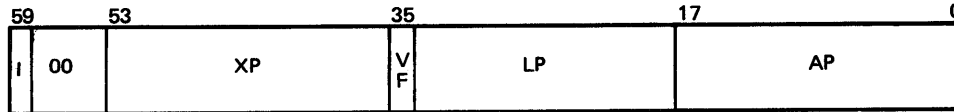
where

SSPN is the subsystem program name and

SSCODE is the 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:



where

- AP is the address of the UCP parameter area
- LP is the length of the request parameter area
- VF is the variable move flag
- XP is the UCP exchange package address
- I is an interlock bit

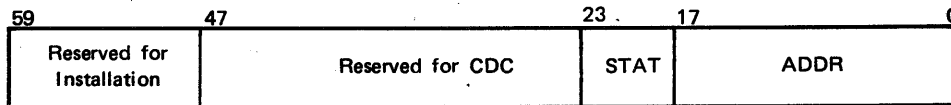
I is set by the operating system when a request has been placed in the parameter area. It is cleared by the subsystem to acknowledge that the request has been received. When the bit is cleared, AP points to a parameter area in which the subsystem is prepared to receive the next request. After clearing I, the subsystem should not attempt to rewrite 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 is set. Attempting a call to itself without this bit being set can result in a subsystem hang.

If I 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) or
- hold the request and periodically attempt to give it to the SCP (RT=0).

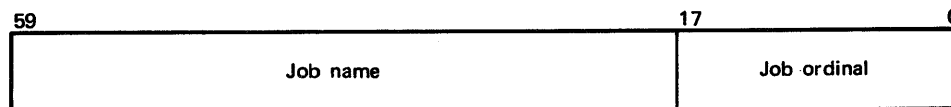
The word at the address AP has the following format:



ADDR is the same as the address in the CALLSS macro. This is a relative address within the UCP field length.

STAT is zero if the call is from a user; 1=normal termination of UCP, 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 UCP parameter word. If WC+3 is greater than LP, only LP words will be moved in.

If XP is non-zero, 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 has the capability of making 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 non-system 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

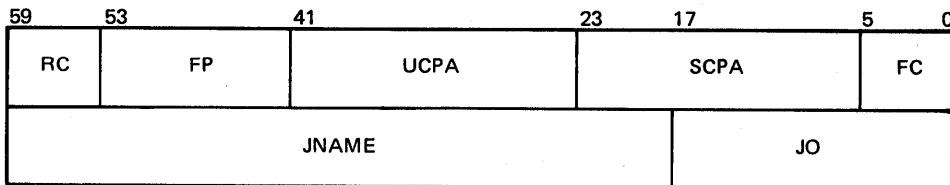
where

label is an optional statement label.

addr is the address of an SFCALL parameter word pair (see below). This parameter can be a register.

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

A typical format of the SFCALL parameter word pair is:



where

RC is the reply code

FP is the function parameter

UCPA is the relative address within the UCP

SCPA is the relative address within the SCP

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

JNAME is the job name and

JO is the job ordinal (JDT or CP)

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

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

Return Code	Meaning
00	No error encountered
01-33	Trivial errors (Reserved for CDC)
34-37	Trivial errors (Reserved for installations)
40	At least one error encountered in list
41	Job identifier invalid
42	SCPA not within the subsystem FL
43	UCPA not within the UCP FL
44	User job swapped out
45	User job not in system
46-56	(Reserved for CDC)
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
67-73	(Reserved for CDC)
74-77	(Reserved for installations)

Possible return codes for each SFCALL function are:

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

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

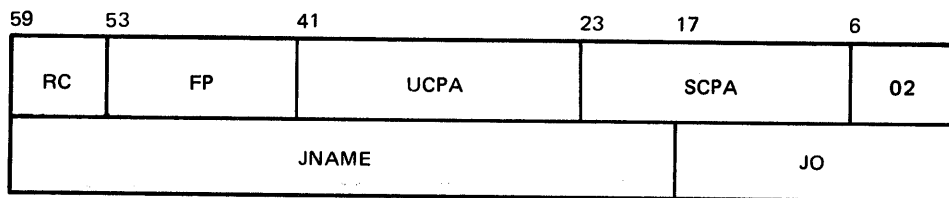
SF.REGR	(02)	Place message into the UCP dayfile and abort the UCP.
SF.TIME	(04)	Obtain accounting data for SCP
SF.ENDT	(06)	Indicate end of task to UCP
SF.READ	(10)	Read from UCP field length
SF.WRIT	(14)	Write to UCP field length
SF.STAT	(12)	Request status of UCP
SF.EXIT	(16)	Exit from SCP status
SF.SWPO	(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
SF.CLTC	(32)	Clear the long-term connection indicator
SF.LIST	(34)	Process a list of SF.xxxx functions

(SF.SWPO and SF.SWPI are not implemented. They are treated as unknown functions (RC=47)).

SF.INS1-4 (70B, 72B, 74B, 76) Reserved for installations.

SF.REGR – REGRETS

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



Definitions of the function codes are:

UCPA = 0 do not abort the UCP

UCPA ≠ 0 abort the UCP. UCPA may have the following nonzero values:

- F.SExx (1) - General subsystem error
- F.SEHU (2) - Hostile user error

SCPA = 0 no message

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

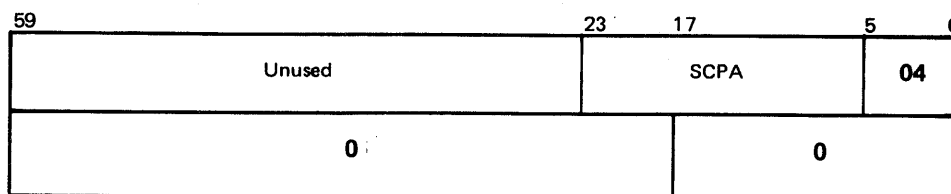
FP dayfile processing flags. The following is a list of the symbolic values and their meanings as defined for the FP field:

- F.SYCP to system dayfile, control point dayfile
- F.NMSN not to control point dayfile

F.JNMN not to control point dayfile, job name in message
 F.CPON to control point dayfile only
 F.ACFN accounting message to system dayfile only
 F.AJNN accounting message to system dayfile only, job name in message
 F.ERLN to error file only
 F.EJNN to error file only, job name 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:



where

SCPA is the relative address within the SCP of a word block for raw accounting data.

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

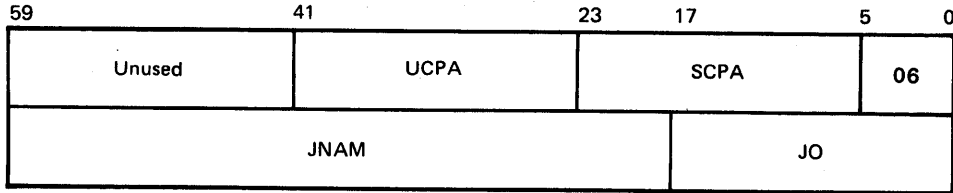
The symbol L.SACT must be used by all subsystems to define the word block lengths (e.g., BSS L.SACT). It allows an installation to add a specially defined area to the word block by modifying the symbol and reassembling the subsystems using this symbol. The operating system is not responsible for setting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



where

UCPA is the relative address within the UCP of the request status word of the task being performed

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

If UCPA > 0:

Set bit zero of the word at UCPA. (Restart UCP if auto-recall was selected). Reduce the request count by one.

If UCPA = 0:

Do not set complete bit (bit 0). Reduce the request count by one.

If UCPA = -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.

If UCPA < -1:

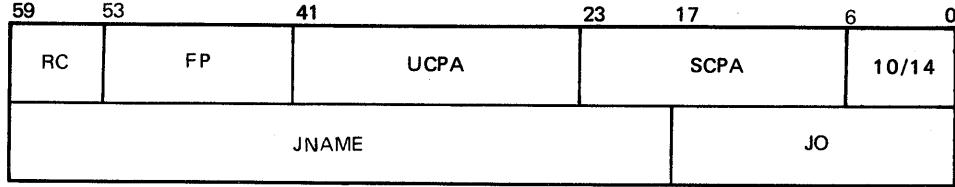
Return error 43.

If SCPA ≠ 0 compute resource usage 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 multi-tasking 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 and SF.WRIT

SF.READ will move FP words from the UCPA to the SCPA whereas SF.WRIT will move FP words from the SCPA to the UCPA.

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



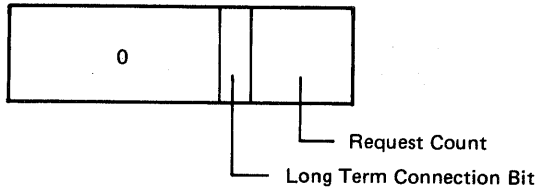
If RC = 42B, the SCPA or SCPA+FP is outside the SCP field length; the SCP is aborted. If RC = 43B, the UCPA or UCPA+FP is outside the UCP field length.

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.

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



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

SF.EXIT – EXIT FROM SCP STATUS

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

- make an SF.EXIT call,
- clear RA.SSID,
- 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, along with the SF.ENDT 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 three 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 below under End Processing for UCPs.

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 central memory 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 one 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 complete until the swap in has been completed. If the swap out request bit is set for a different SCP, the UCP will remain 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 one second.

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 is used to identify the termination notification. If the UCP was terminated by an ENDRUN, the STAT field contains the value one. In the event of an error condition the field contains the value two. All normal calls from the user contain a zero.

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 two seconds and the message CONNECTED TO scpjobname is flashed at the UCP on the B display.

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

If the CONNECTED TO scpjobname 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 UPCs 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.

Stop accepting any requests.

Complete processing any requests already received.

Issue an SF.EXIT.

Do 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 will set the LK bit in RA.SSC and will not send any more requests as long as LK is not cleared.

ABNORMAL SCP TERMINATION

A subsystem should make use of the RECOVER capability (see SCOPE 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.

SCP is aborted with the message EXIT - WITH CONNECTIONS.

The wait response counts and long term connections are nullified.

The message subsystem name ENDED BY SYSTEM is sent to the UCP dayfile.

UCP is aborted.

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:

59	53	41	23	5	0
RC	FP	0	SCPA	34	
JOBID					

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 for proper action.

If FP = 0, the entire list has been processed by the operating system. If FP is nonzero, 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 = 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 zero) accumulate error returns whether or not the entire list is processed on one SFCALL.

Special conditions and notes.

- A. The operating system will abort the SCP during SF.LIST processing if a fatal error occurs in the SF.LIST, or in any member of the list.
- B. The detailed error conditions must be determined by examining the individual list entries whenever the SF.LIST RC field equals 40B.
- C. The individual functions are handled in exactly the same way whether or not the list mode is enabled.
- D. Error status 42B when SCPA is illegal means none of the list entries have been processed. This check is made prior to initiating the list process.
- E. Error status 42B when FP = 0 means that none of the list entries have been processed on this call. If the subsystem improperly handles the FP = 0 condition, the entire list may have been processed prior to the subsystem abort.
- F. List entries are processed sequentially by the operating system and those 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 is provided to allow 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 caused (for example) an excessively long uninterruptable interval.
- G. If the SCP is aborted due to an error in one of the list entries other than the SF.LIST, RC = 40B, SCPA and FP are updated and FC is set complete. The proper return status is also placed in the offending list entry.
- H. The functions SF.REGR, SF.SWPI 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.

HOW TO DEFINE A SUBSYSTEM

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

The SSCT macro is provided for defining subsystems. The macro has three 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 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 reside in the system library or it will not be granted system control point status. The PUF parameter should only be used during subsystem development.

PROGRAMMING TIPS

A system control point runs at a very 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 will not be 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 autorecall bit should not be used in any RA+1 call from an SCP because an autorecall status would prevent it from responding to new requests as they come along.

PERIPHERAL PROCESSOR ORGANIZATION

When SCOPE 3.4 is loaded into the computer at deadstart time, system monitor, MTR, and the system display program, DSD, are loaded into peripheral processors 0 and 1, respectively, where they reside permanently. All peripheral processors in the system contain a group of permanently assigned storage locations called the PP direct cells; 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 may be loaded into the unassigned pool processors.

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 may load additional overlays to help complete its function; on completion, the program may be overlaid by another transient program loaded by PP resident to perform another, often unrelated, function. Figure 3-1 shows a typical layout of a pool PP loaded with a transient PP program and an overlay. The PP direct cells occupy locations 0 to 77; the PP resident is loaded starting at location 103 to approximately 777. The remainder of the PP is occupied by PP transient programs, except for PP0 and PP1 which contain MTR and DSD, respectively.

Specific assignment of the PP direct cells is detailed in the following chart.

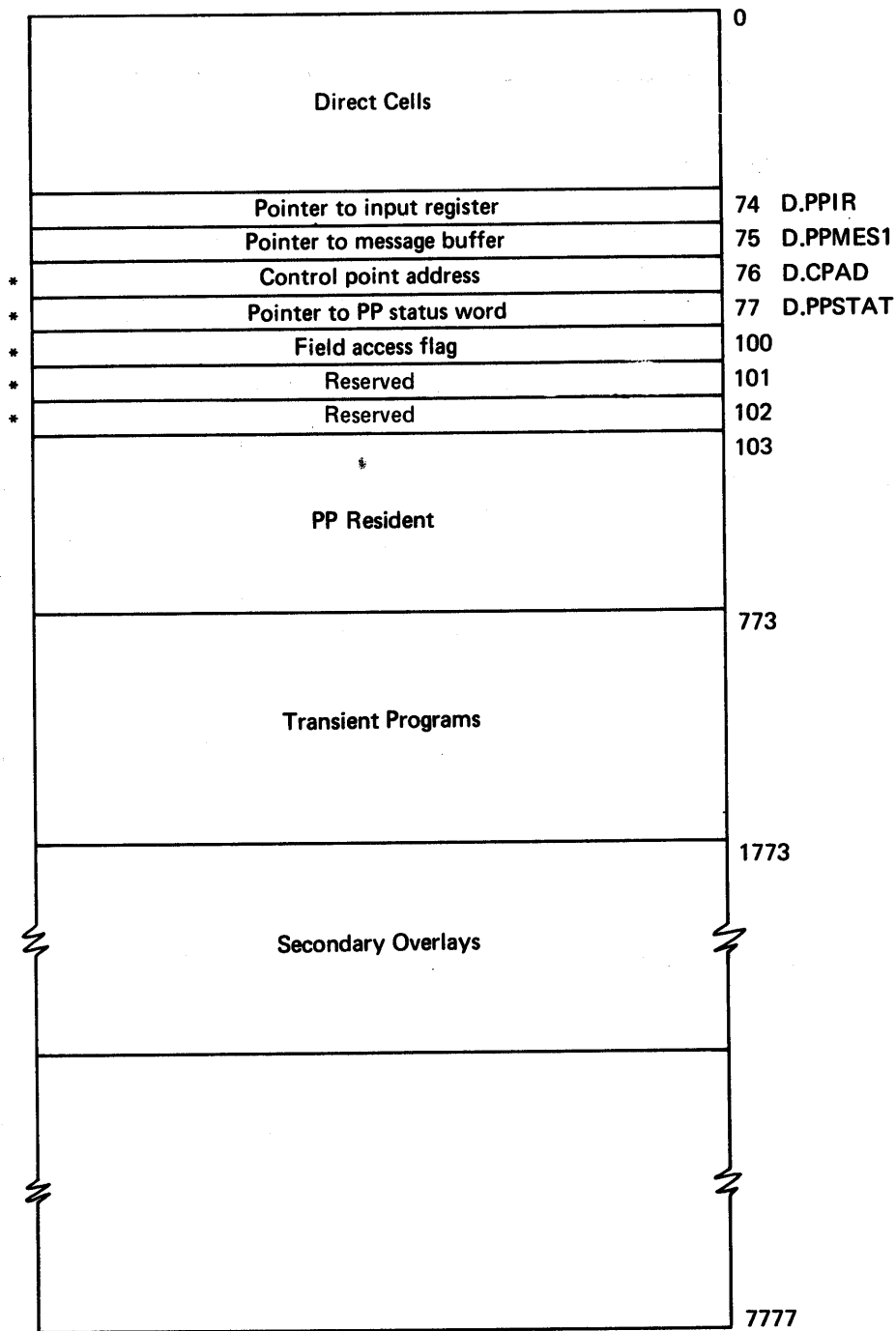
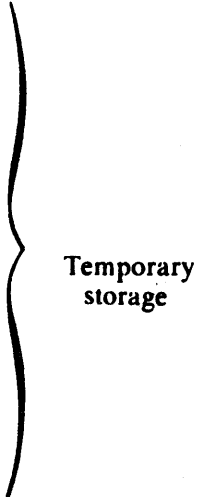


Figure 3-1. Pool PP Layout

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

PP DIRECT CELL ASSIGNMENT

Octal	Identifier	Function
0	D.Z0	
1	D.Z1	
2	D.Z2	
3	D.Z3	
4	D.Z4	
5	D.Z5	
6	D.Z6	
7	D.Z7	
10	D.T0	
11	D.T1	
12	D.T2	
13	D.T3	
14	D.T4	
15	D.T5	
16	D.T6	
17	D.T7	
20	D.FNT/D.TW0	D.FNT through D.FNT+9 contain 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 contain the EST entry in process. D.JPAR contains a job parameter word.
37	D.DTS/D.JFL D.TH7	D.DTS contains device type code in left 6 bits and allocation type code in right 6 bits; D.JFL contains CM field length requirement returned to caller by 2TJ.
40	D.BA/D.FR0	Contains first word of FET in D.BA through D.BA + 4 (buffer address).
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/DFR7	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) control point to which PP is attached.
56	D.FL/D.FF6	CM field length divided by 100 (octal) for job at control point to which PP is attached.
57	D.FA/D.FF7	Address of second word of 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 word OUT in circular I/O buffer.

Octal	Identifier	Function
66	D.LIMIT/D.SX6	This and next cell contain 18-bit address of word LIMIT in circular I/O buffer.
70	D.PPONE/D.SV0	Preset to constant value plus one (+ 1).
71	D.HN/D.SV1	Preset to constant value + 100 (octal).
72	D.TH/D.SV2	Preset to constant value + 1000 (octal).
73	D.TR/D.SV3	Preset to constant value + 3.
74	D.PPIR/D.SV4	PP input register address.
75	D.PPMESI	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		Field access flag.
101		Channel time in seconds.
102		Channel time in milliseconds.

PP COMMUNICATIONS

For each pool PP, CMR has an area used for communication between the PP monitor and the PP. Each area contains a PP input register and a PP output register, each one CM-word long, plus a six CM-word message buffer. In section 2, Figure 2-3 is a diagram of a PP communications area.

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 will load 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 will have inserted the requesting program's control point number into bit 36-39 of the word and cleared bit 41. Bits 0-35 appear in the input register exactly as they did in RA + 1 of the requesting CP program.

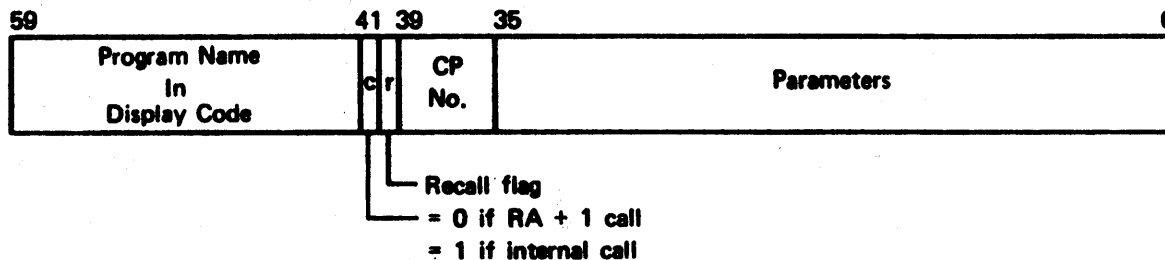


Figure 3-2. PP Input Register

The PP resident in each PP constantly scans its own input register. When it becomes non-zero, the PP resident issues the M.ICE monitor request with EX.PLIB function code and the program name. A search is made for the program in the CMR library directory and the program is prepared for loading. The PP will load the transient program at location 773 and store the address of the control point area to which the PP is assigned in direct cell D.CPAD, then transfer control to location 1000 to start execution. If the transient 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 will clear the PP input register and record 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, signalling that the monitor has processed the request. If CPMTR or PPMTR need to communicate with the PP about the request being processed, it will store the necessary information in the remaining bytes of the output register.

PP RESIDENT

The PP resident program performs two 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

The PP resident is made up of a series of routines, each of which performs a specific function. (See figure 3-3.) These resident routines are used by the transient and overlay programs as required. The routines, their names, locations, calling sequences and functions are described below.

Field Access Flags	R.FAF
PP Resident Idle Loop	R.IDLE
Load Primary PP Overlay	R.OVLJ
Request Access to Control Point Field Length	R.RAFL R.PAUSE
Terminate Access to Control Point Field Length	R.TAFL
Compare Accumulator to Field Length	R.TFL
Process Monitor Function	R.MTR R.PROCES
Wait for Output Register to Clear	R.WAIT
Reserve Channel	R.RCH
Drop Channel	R.DCH
Mask a Byte into Specified Words	R.STBMSK R.STB
Load PP Overlay	R.OVL
Access Request Stack Entry	R.EREQS
Transmit Dayfile Message	R.DFM
Transmit Data To/From PPU	R.WRITEP R.READP
Read/Write Logic (Disk)	R.RWP
Read/Write Segments (Non-disk I/O)	

Figure 3-3. PP Resident Routines

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. R.IDLE destroys direct cells 20 through 22 and some of the temporary storage cells; R.OVLJ and all other PP resident routines destroy only the temporary cells 0 through 17.

R.OVLJ Primary (transient program) Overlay Loader

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

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

R.OVL Overlay Loader

Calling sequence: Load A register with load point address
RJM R.OVL

The M.ICE monitor function with EX.PLIB function is issued along with the overlay name found left justified in D.T6 and D.T7. The overlay is found and prepared for loading. After issuing the request, PP resident waits for byte 0 of the PP output register to become zero and for word 3, byte 0 of the PP message buffer to become non-zero.

If the value of word 3, byte 0 of the PP message buffer is 1, the overlay is disk resident. The stack request has already been issued by CP.MTR; therefore the resident establishes communication with the assigned stack processor and acquires the overlay through a channel. If the value is 5, the overlay is CM resident. The CM address of the overlay and length are given in the same PP message buffer third word. PP resident reads in the overlay. If the value is greater than 5, the overlay is ECS resident. The PP output register and the first four words of its message buffer are used for further communication between CP.MTR and the resident. The overlay is loaded from the CM data buffer area or directly from ECS through DDP according to the type of assigned system circular buffer.

R.READP Transmit Data Via Channel from Stack Processor

R.WRITEP Transmit Data Via Channel to Stack Processor

Calling Sequence: Load L(request)
RJM R.READP (R.WRITEP)

Computes PP word count from first and last word addresses given in the request formatted at the request location and adds the computed word count, the address of the PP message buffer, and the control point number to the request. The request is entered in the stack, and data is transmitted via channel directly to/from PP memory. Upon exit from R.READP (R.WRITEP), the following information is set:

(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	Special entry point to R.READP used by LDR.
R.RWPP	Word in R. READP modified by LDR.
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 and searches the central memory request stack for an empty entry. The monitor function, M.ICE/EX.SPM is called, and PP resident iterates until MTR accepts the request. If the available flag is set (S.STF + S.STFA of byte C.STFB of the second word of the request), R.EREQS exits to R.IDLE. Otherwise, it returns control to its caller.

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 Length 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.

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 A register; otherwise, the A register will contain the resultant absolute CM address (RA + relative address) upon exit. 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

(R.PROCES is identical with R.MTR)

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

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.WAIT PP Wait Loop

Calling Sequence: RJM R.WAIT

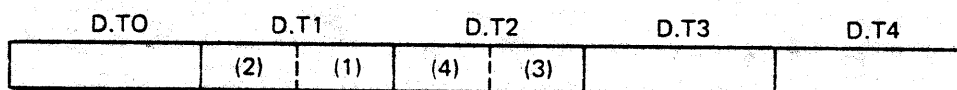
Determines if the monitor function is for MTR or CPMTR. If the MXN instruction is not available, R.WAIT is modified at deadstart causing R.WAIT to assume that all functions are for MTR. If the function is for CPMTR, the PP input register address is written into T.PPID and T.MXNCTL is read in and executed.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

R.RCH Request Channel

Calling Sequence: Load channel numbers
 RJM R.RCH

Channel numbers loaded in the A register will be stored in D.T1, the monitor function M.RCH inserted into D.T4, and then D.T0 through D.T4 will be written to the output register for that PP. Channel numbers in D.T1 and D.T2 will be 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, etc.

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

LDN	0
STD	D.T2
LDC	1312B
RJM	R.RCH

Normally, MTR will stop looking for alternate channels after four have been investigated; in the above example, only two channels will be investigated.

A message is written to the dayfile and/or displayed on console. The flag bits in the high-order 6 bits of the A register are used to determine message destinations. In the flag bit values given below, one or more bits may be on; all are optional. Refer also to M.DFM.

<u>Bit</u>	<u>Description</u>
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

FIELD ACCESS FLAG USAGE

The control point field access flag (R.FAF) is found at location 100g 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 the previous 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, and 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.

†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 is used to identify messages from a task that was executed on a different mainframe.

SCOPE SYSTEM MONITOR

The monitor performs a set of functions that must be performed by a program that is permanently resident. Among these functions are:

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

The monitor is implemented as two distinct components. MTR, which runs continuously in PPO, and CPMTR, which resides in central memory and uses the central processor intermittently for short bursts.

The monitoring tasks are divided between the two processors on a functional basis to distribute the work load in the most efficient manner.

CPU SCHEDULING

A primary function of CPMTR is to assign 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 itself. 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:

W	This bit is set by a M.RCP or M.SETST 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 & D	These bits are 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 will not be restarted until the requested system function is completed.
Z	The job is suspended as it threatened to saturate the system with PP calls.
M, P & 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,

SYSTEM JOB EXCHANGE PACKAGE AREA

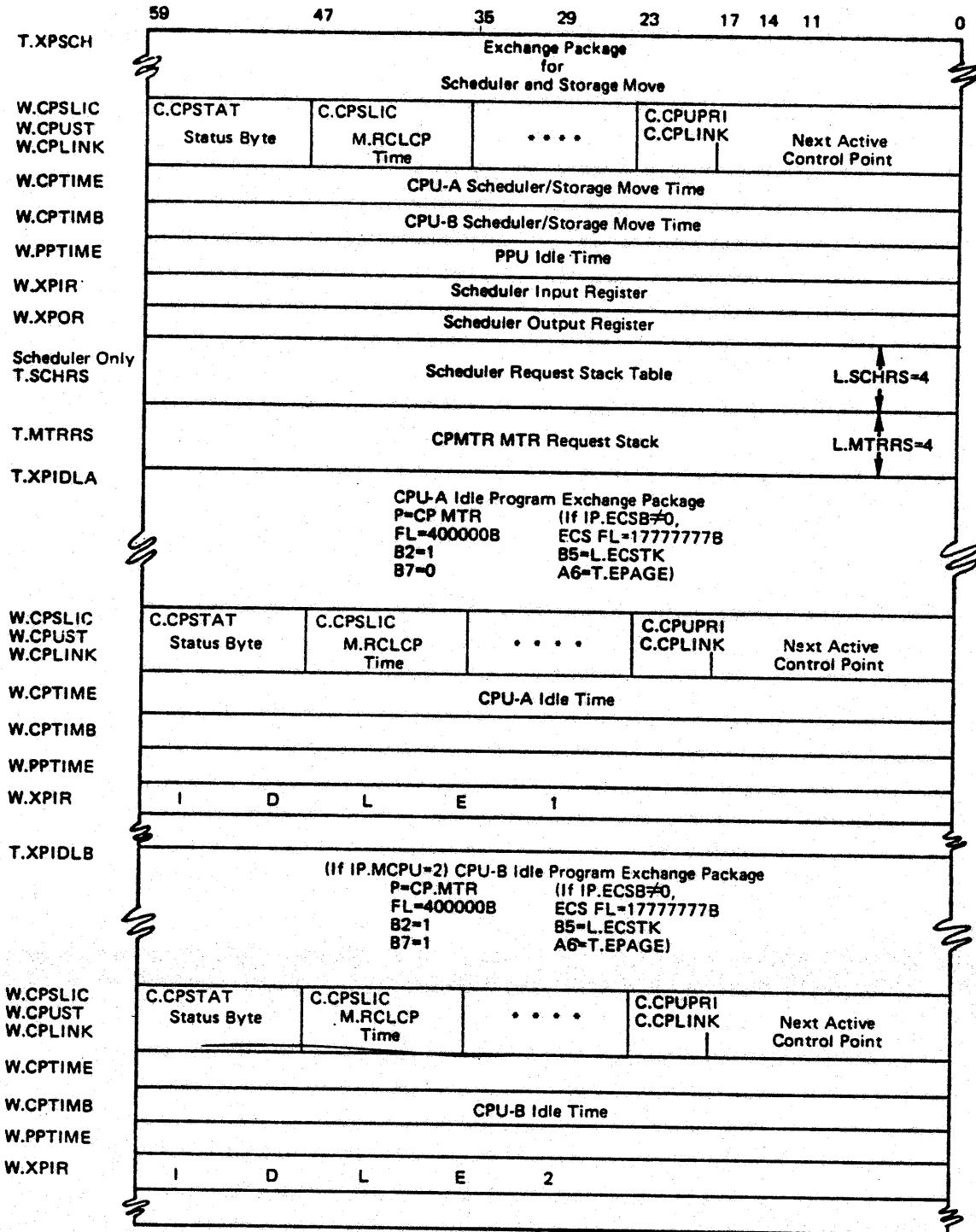


Figure 3-4

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:

PR.IDLE	Zero level default CPU job to be used only in the absence of any other
PR.BATCH	Batch jobs initiated by IIB
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, they will share the CPU on a round-robin basis. Each uses the CPU for BASESLIC milliseconds before control is passed to next.

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, the 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 stack processor to access RMS. More than one PP can be reserved for the stack processors. (See N.SPRPP in CMR.) MTR maintains three lists of PP calls that are not currently assigned to a PP.

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. They are ordered in sequence of their time delay. When the delay is expired, they 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 PPO. The PP calls are kept in the peripheral job table (T.PJT) in central memory. 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.

CHANNEL RESERVATIONS

CPMTR processes channel reservations. The CPU is used for this frequently used function as 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 14g 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 12-35 of T.MSC contain the total number of seconds since the last deadstart, expressed as a 24-bit binary number. Bits 0-11 are the binary fractional parts of a second. Bits 0-35 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 stack processor.

STORAGE REQUESTS

MTR and integrated scheduler perform the memory management tasks in a synchronized procedure. MTR will attempt to process storage requests. If storage is not available, MTR will initiate scheduler to determine if storage will be made available. MTR will not attempt to process any storage requests while scheduler is running.

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 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.MSNCTL. 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 non-zero, 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.

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 roughly 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 14g.

- 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, M.RCH and M.RSTOR, 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 IAJ should be called to advance to the next control statement.

- Check RA+1

Examine RA+1; and 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.

MONITOR FUNCTIONS

The following descriptions of the monitor functions are in alphabetic order. The tables in Part II, section 1, 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.

M.ABORT ABORT CONTROL POINT AND DROP PP (M.ABORT,****,****,****,****)

The job at the requesting PPU is terminated. The requesting processor is responsible for the dayfile message. 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 (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

(M.CCPA,****,****,****,**NN)

The requesting PPU 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 PPU 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

(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. Will cause linkage to or delinkage from chain of control points actively waiting for a CPU.

M.CPJ CAPTURE PERIPHERAL JOB

(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

(M.CPUST,000X,****,****,****)

- X = 0 If either CPU is OFF, it is turned ON. If the CPU was locked OFF at deadstart time, it remains OFF.
- X = 1 If both CPUs are ON, CPU-A is turned OFF.
- X = 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

(M.DCP,****,****,****,****)

Execution of the central processor 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 PPU.

M.DFM PROCESS DAYFILE MESSAGE

(M.DFM,FFFF,MMMM,****,****)

Dayfile flag bits FFFF determines message handling:

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 (no A display)
3	Flag message as an accounting message
4	Send message to hardware error file
5	Do not insert job name in system dayfile

If bits 1 and 4 are both 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 both set, the time is omitted and replaced with blanks in the control point dayfile. This option is used to identify messages from a task that was executed on a different mainframe.

If bit 3 is set, a dollar sign precedes the dayfile entry to identify it as an accounting message. If bit 5 is also set, the dollar sign is suppressed on the assumption that a job which is not identified by a job name cannot be billed for resources.

When the value of MMMM is greater than that in PPOR, it is taken to be the LWA + 1 of the message in the PP message buffer. When the value is less than that in PPOR, MMMM is taken to be a dump index for a requested dayfile dump.

Value of dayfile dump index:

0	System dayfile dump
1 thru N.CP	Control point dayfile dump
N.CP+1	Hardware error file dump

M.DPP DROP PP

(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

(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:

F = 0	Event stack job assigned when bit is off (F.ESOFF)
F = 4	Event stack job assigned when bit is on (F.ESON)
B = 0	AAAAAA is an absolute address (F.ESABS)
B = 1	AAAAAA is relative to RA (F.ESREL)
B = 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 PPU
(M.EESD,00AA,AAAA,FFFF,SYTT)

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

M.ICE INITIATE CENTRAL EXECUTIVE
(M.ICE,PPPP,PPPP,PPPP,EX.xxx)

EX.xxx is a subfunction to be performed. See part II, section 1, under Monitor Functions, for a listing of these subfunctions. The PPPP fields can be used as parameters to the subfunction. Subfunction routines may run in either monitor mode or user mode. Those run in user mode use the exchange package at T.XPSCH. The content of the M.ICE call is placed into the user mode input register at T.SPSCH+W.XPIR before the user mode routine is initiated. This input register is cleared to signal completion.

M.ISP INITIATE STACK PROCESSOR
(M.ISP,000X,000N,****,CCCC)

CCCC	DST ordinal of stack processor to be initiated
X = 0	Initiate 1S5 only if PP active flag is zero
X ≠ 0	Initiate 1S5 regardless of PP active flag setting
N = 0	Initial assignment of 1SP to the DST entry
N = 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 N is 1 or 2 and X ≠ 0, this is a call from 1SP for a partner to work together for a dual access device.

M.KILL BAD FUNCTION REQUEST
(M.KILL,****,****,****,****)

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

M.NTIME ENTER NEW TIME LIMIT
(M.NTIME,TTTT,T***,****,**NN)

A central processor job time limit of TTTTT seconds is entered at the control point. Any previous time limit is superseded. If the requesting PPU is assigned to control point zero, the parameter NN will give the number of the control point to be considered; in any other case this parameter is irrelevant.

M.PASS PPMTR IGNORES FUNCTION REQUEST
(M.PASS,****,****,****,****)

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

M.PATCH INSERT A PATCH IN PPMTR
(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.RACT REQUEST CONTROL POINT ACTIVITY
(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 will be 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
(M.RBTSTO,SSSS,****,****,****)

PPMTR 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
(M.RCH,BBAA,DDCC,***G,RRRR)

AA 1st choice channel number
BB 2nd choice channel number
CC 3rd choice channel number
DD 4th choice channel number

G = 0 Normal charge for channel time.
G = 4 Do not charge control point for channel time.
RRRR = 0000 Request immediate reply.
RRRR ≠ 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 will set byte 0 of the PPU output register to zero. When a channel is granted, its number is returned in the PPU output register byte 1 (location of AA) and byte 4 is set to non-zero value. Thus, programs that request an immediate reply must check that byte 4 is non-zero 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
(M.RCLCP,****,****,****,****)

This request is effective only if the central program associated with the requesting PPU 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
(M.RCP,****,****,****,****)

This request is ignored under the following conditions:

Requesting PPU is assigned to control point zero

Error flag is set for the control point

Job is already in the waiting status.

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

M.RPJ REQUEST PERIPHERAL JOB

(M.RPJ,SSSS,FFFF,****,****)

This function requests that another PPU program be initiated after a specified time delay. The first word of the requesting PPU message buffer contains the input register image of the new PPU program. The time delay is SSSS seconds plus FFFF/10,000B seconds. If the time delay is zero and no PPU 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 PPU

(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 will not be charged for the PP time. The use of the time delay is the same as for M.RPJ.

M.RSTOR REQUEST STORAGE

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

CCCC	Requested CM/100 octal
XXXX	Requested ECS/1000 octal
TT	00 CM request only
	01 ECS request only
	02 CM and ECS request
	03 Request reserved CM
	04 Request CM—will await response
	06 Request CM and ECS—will await response
	07 IP.POSFL requested by swapper
	20 Priority storage requested

Assign CCCC central memory and/or XXXX extended core storage to the control point of the requesting PPU. 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 always is honored immediately. If TT = 02, PPMTR can honor part of the request without honoring the remainder.

MEMORY ALLOCATION

Central memory and direct-access extended core storage is assigned in the same sequence as control point numbers. Storage associated with any control point is contiguous.

All storage is associated with the assigned control points. If no user jobs are assigned to control points, all storage is associated with the central memory resident, which operates at control point zero. All storage, therefore, falls into one of two categories:

Allocated storage, defined by the RA and FL of the control point.

Unallocated storage, defined as that which occurs between the allocated storage of two consecutive control points. All unallocated storage is associated with the lower numbered control point.

PPMTR maintains two tables in PP zero which contain the size of allocated and unallocated CM and ECS storage associated with each control point (0 through N.CP). A request for reduced field length will have the effect of reducing the amount of allocated storage and transferring the value to unallocated status; no storage is actually moved. When the total storage associated with a control point is sufficient, a request for an increased field length will cause transfer of the increased value from the unallocated to allocated state; no actual storage move takes place.

If there is insufficient storage associated with a control point to satisfy an increased field length request, unallocated storage may be transferred from adjacent control points. A scan of the unallocated storage table entries for the adjacent control points will locate a combination of blocks which results in moving the least amount of storage.

When a control point RA and/or FL is to be changed. PPMTR will suspend the control point by setting the M bit of the CP status byte. If a CM move is necessary. PPMTR will set the storage move flag and wait for all PPs assigned to the control point to pause and then initiates the system exchange package to start the storage move program. After completion of the move, PPMTR will modify the control point RA and/or FL and resume the control point by clearing the M status bit.

A PPU pause occurs when the field access flag (C.FAF) in the PP status word is zero. If an ECS parity error occurs during an ECS move, PPMTR will abort both the requesting control point and the control point whose ECS is being moved, if it is no longer intact.

M.SCB SYSTEM CIRCULAR BUFFER SURVEILLANCE (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 (M.SCH,000X,00CC,00JJ,JJJJ)

When X = 2, the contents of this output register are placed into a buffer at T.SCHRS for integrated scheduler's attention during its next normal execution. Integrated scheduler is not initiated immediately. When integrated scheduler processes this request it will link 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 ≠ 2, the output register contents are not passed to scheduler. Both types cause MTR to initiate scheduler.

M.SEF SET ERROR FLAG

(M.SEF, **NN,EEEE,****,****)

Monitor will drop 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 0-5 of byte C.CPMEMO(0). The high order bits (6-11) 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 (-77B) is used to initiate memo mode. Bits 6-11 of C.CPMEMO are set on. Bits 0-5 of C.CPEF are moved to bits 0-5 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) is used to terminate memo mode. Bits 0-5 of C.CPMEMO are moved to C.CPEF and C.CPMEMO is cleared. Error flag zero can also be used to terminate memo mode, but it will clear both the error memo and the error flag fields. If an error memo is already recorded when the F.ERTMM is issued, the memo will be 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 77B will 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 will cause the control point to revert to normal error flag mode.

Cautions

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 card 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 will process it as an error flag.

M.SEQ ASSIGN JOB SEQUENCE NUMBER

(M.SEQ, ****, ****, ****, ****)

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

M.SETST SET STATUS BITS
(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. May cause linkage to or delinkage from chain of control points actively waiting for a CPU.

M.SLICE TERMINATE TIME SLICE PERIOD
(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.SPM SPM CALL FROM 1SP
(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 Change 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 wake up 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 re-initializes itself. When 1SP is idle and detects an outstanding channel request for its channel, it issues a status of 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)

Where - SSSS = 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 0-2 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)

where SSPP, PPPP = Successor call type

PPPP = 0 - No successor call (clear PPOR)

PPPP ≠ 0 and SS = 0 - Set up M.BUFPTR call

PPPP ≠ 0 and SS ≠ 0 - Set up M.SCB call PPPP - successor call parameters.

M.SPRCL STACK PROCESSOR RECALL

(M.SPRCL,00AA,BBBB,000F,CCCC)

CCCC is the control point area address. The I/O time and PP time are updated by the product of AA*BBBB. The count of outstanding stack requests is modified in W.CPSR:

F=0	No adjustment
F=1	Subtract one
F=2	Add one

Execution can be resumed if the control point is in recall.

M.TRACE ENTER MONITOR TRACE MODE

(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 CDC developmental use.

M.TSR TERMINATE STORAGE REQUEST

(M.TSR,****,****,****,****)

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

PP ASSIGNMENT

To control PP assignments, MTR keeps (in PPO) 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 looked at it (refer to M.BUFPTR).
- APLINK Active PPU link. This is a pointer to the next member in a chain of active PPUs. The chain always starts with MTR (PPO) and ends with DSD (PP1). The next PPU is identified by its output register index value.
- JUMPAD This is the address saved for reentry 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 PPU 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 3-5.

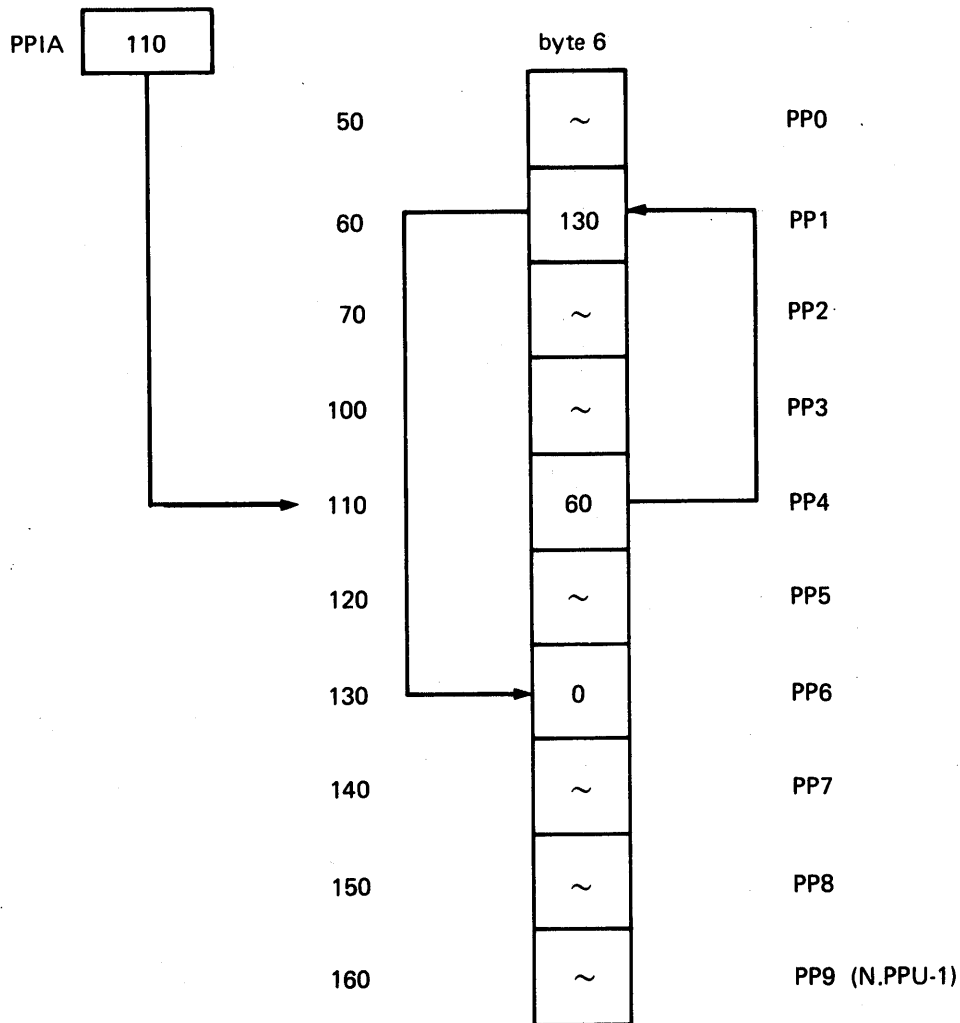


Figure 3-5. PP Chain

MTR assigns a PP by writing a peripheral job name and a control point number into the PP's 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 or M.RPJ 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 IAJ 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 4-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.
- 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 3-6.

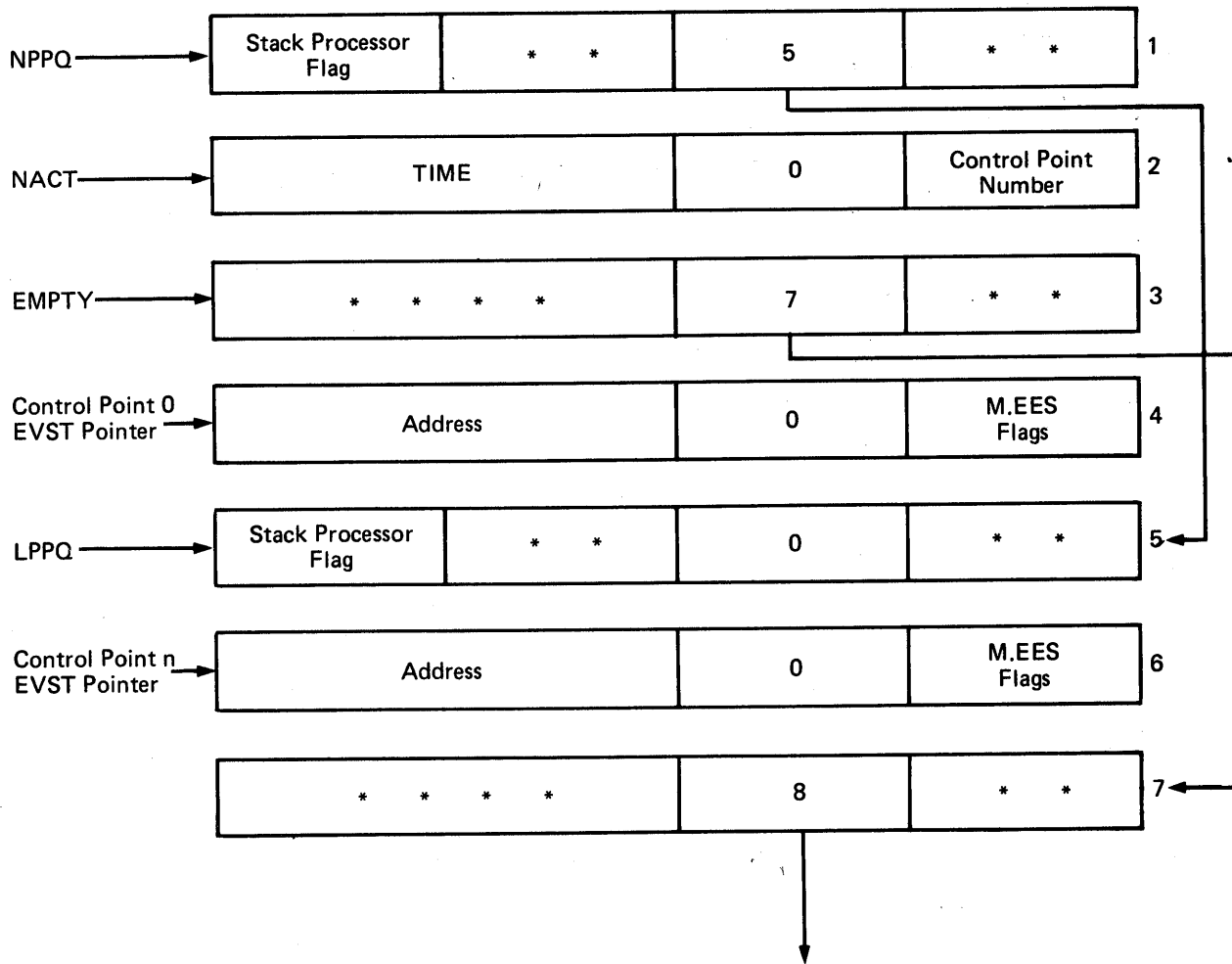


Figure 3-6. PP Job Queue

FILES

A name associated with each file identifies it to the system and to the user. Files are stored on either allocatable or nonallocatable devices. Rotating mass storage units, such as disk or drum, 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. The following paragraphs describe briefly each of the file groups. Files are uniquely known by the lfn, the source or destination ID, and the terminal ID (if applicable).

SYSTEM FILES

The files described below are always in the system; they are always assigned to control point zero and reside on allocatable devices. These files, except for the job dayfile, are maintained on system devices as permanent files.

SYSTEM This file has the name of ZZZZZ04 and contains a copy of the deadstart system tape.

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.

DFILE_n Each user control point in the system has a job dayfile; *n* is the control point number. These files are assigned to control point zero 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 cards 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.

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 by Deadstart and is updated by EDITLIB. It is not permanently attached to control point zero and is used only for deadstart recovery.

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 non-allocatable 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 non-allocatable 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 non-zero 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 job name and the file is assigned to control point zero. A local file with the name PUNCH or PUNCHB will be output to a card punch; the file OUTPUT is printed.

PERMANENT FILES

Permanent files are saved across deadstarts and are therefore considered to be 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.

QUEUE FILES

To provide for the recovery of input and output files from disk tables on non-recovery 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.XbTLOAD) 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 PFC (formerly the RBTC) 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(lfn,DC=IN). 2VJ verifies the job card 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 zero as an input file. The input queue consists of all files assigned to control point zero 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 7-character name. The job name from the job card 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 may be used as sequence characters, therefore 1,296 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 non-zero disposition codes. When a job terminates, such files are cataloged, assigned to control point zero, 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 non-zero 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 card 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 may be the name of the job which created it, and since a job may create several files which go into the output queue, names in the output queue often are not unique.

PLACING FILES IN QUEUES – ROUTE MACRO

The ROUTE function places a file in an input or output queue. This function is available as a control card or as a macro. Use of the ROUTE macro allows a system job additional capabilities denied a user job. A system job can specify a source ID, a seven-character job name, a pre-dayfile lfn, and a priority. Other capabilities are the same as for user jobs. See the SCOPE Reference Manual, publication number 60307200, for a discussion of the ROUTE control card.

The user must construct a parameter block in the format described below before calling the ROUTE macro.

ROUTE tag,recall

- tag Address of the ROUTE parameter block
- recall Optional non-blank character indicating automatic recall

PARAMETER BLOCK FORMAT

	59	47	41	35	23	21	17	11	0	
tag	Logical File Name							Error Code	Unused	A
tag+1	0 0 0 0	Forms Code/ INPUT Flags		Disposition Code	E C	I C	Flags			
tag+2	Station ID- Source		Station ID- Destination		Unused			TID		
tag+3	File Identifier (FID)							B	Priority	
tag+4	Pre-dayfile lfn					Repeat Count		Unused		

Word	Bits	Field	Description																											
tag	59-18	Logical File Name	lfn of file to be ROUTED; must be mass storage file, not a permanent file; does not reside on a dismountable device; must have at least read permission																											
	17-12	Error Code	Code returned by system when bit 12 of flag field is set. A list of the error codes and diagnostic messages follows the description of the parameter block.																											
	11-1	Unused																												
	0	A	Complete bit. Must be zero when macro is issued; system sets to one when function is complete																											
tag+1	59-48	Zero	Twelve bits of zero. Allows compatibility with previous callers of DSP. The old calling sequence puts the lfn in tag+1.																											
	47-36	Forms Code/ INPUT Flags	Two display code letters and/or digits identifying the forms to be used for this file. Default is standard forms. If the file is to be routed to an INPUT queue, this field is defined as: <ul style="list-style-type: none"> 47 Unused 46 Unused 45 Do not catalog INPUT file 44 FID= 7 characters specified. Ignored if not a systems job 43 Send file to INPUT queue even if job card error 42 Use dependency count 41-36 Dependency count 																											
	35-24	Disposition Code	Disposition Code mnemonic in display code																											
	23-21	EC	External characteristics code translated by the following table: <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Value (octal)</th> <th>Print File</th> <th>Punch File</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>EC(default)</td> <td>EC(default)</td> </tr> <tr> <td>1</td> <td>- -</td> <td>EC=SB</td> </tr> <tr> <td>2</td> <td>EC=A4</td> <td>EC=80COL</td> </tr> <tr> <td>3</td> <td>EC=B4</td> <td>- -</td> </tr> <tr> <td>4</td> <td>EC=B6</td> <td>EC=026</td> </tr> <tr> <td>5</td> <td>EC=A6</td> <td>EC=029</td> </tr> <tr> <td>6</td> <td>EC=A9</td> <td>EC=ASCII</td> </tr> <tr> <td>7</td> <td colspan="2">Reserved for installations</td> </tr> </tbody> </table>	Value (octal)	Print File	Punch File	0	EC(default)	EC(default)	1	- -	EC=SB	2	EC=A4	EC=80COL	3	EC=B4	- -	4	EC=B6	EC=026	5	EC=A6	EC=029	6	EC=A9	EC=ASCII	7	Reserved for installations	
Value (octal)	Print File	Punch File																												
0	EC(default)	EC(default)																												
1	- -	EC=SB																												
2	EC=A4	EC=80COL																												
3	EC=B4	- -																												
4	EC=B6	EC=026																												
5	EC=A6	EC=029																												
6	EC=A9	EC=ASCII																												
7	Reserved for installations																													

Word	Bits	Field	Description							
tag+1	20	Unused								
	19-18	IC	Internal characteristics codes transformed by the following table: <table border="0"> <tr><td>0</td><td>IC or IC=DIS (default) – display code</td></tr> <tr><td>1</td><td>IC=ASCII</td></tr> <tr><td>2</td><td>IC=BIN – Binary</td></tr> <tr><td>3</td><td>Reserved</td></tr> </table>	0	IC or IC=DIS (default) – display code	1	IC=ASCII	2	IC=BIN – Binary	3
0	IC or IC=DIS (default) – display code									
1	IC=ASCII									
2	IC=BIN – Binary									
3	Reserved									
tag+2	17-0	Flag Bits	Indicate which parameters are specified:							
			17	File name assigned by system is returned at tag bits 59-18						
			16-15	Unused						
			14	Repeat count						
			13	Dayfile attached for immediate route to output (systems jobs only)						
			12	No dayfile message, return error code in bits 12-17 of word 1 of calling sequence						
			11	Pre-dayfile lfn specified (systems job only)						
			10	Forms code						
			9	Priority						
			8	Internal characteristics						
			7	External characteristics						
			6	FID=* System appends two unique sequence characters to the file identifier						
			5	FID System uses FID specified in tag+3, bits 59-18. Only systems jobs can specify seven characters, other users specify five.						
4	Disposition code									
3	Route to remote station									
2	TID									
1	Route to central site									
0	End-of-job (deferred ROUTE)									
tag+2	59-42	Station ID-Source	Specified by system jobs only. The three display code characters in this field are used as the source ID for a job routed to an input queue. When this field is binary zero, the routed file has no SID. 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.							
	41-24	Station ID-Destination	Display code destination ID. The file is processed by the system with this logical identifier.							
	23-12	Unused								
	11-0	TID	Display code identifier of the INTERCOM terminal to which the file is sent.							

Word	Bits	Field	Description
tag+3	59-18	FID	If the calling job has not been loaded completely from the system library, only a maximum of five characters may be used to specify FID. The additional two character sequence number is determined by flag bits 5 and 6. Seven characters may be specified by calling jobs which are loaded completely from the system library.
	17-13	Unused	
	12	B	Must be set if priority is specified.
	11-0	Priority	Priority for an interactively routed output file being routed to the routing terminal.
tag+4	59-18	Pre-dayfile lfn	The logical file name of the file which contains the pre-dayfile. This parameter is only meaningful for DC=IN, and can only be specified by systems jobs.
	17	Unused	
	16-12	REP	Repeat count.
	11-0	Unused	

ERROR PROCESSING

When an error occurs in processing a ROUTE macro, either a dayfile message explaining the error is issued, or an error code is returned in bits 17-12 of the first word in the parameter block. If the address of the parameter block is outside the field length of the job or if the complete bit is set when the macro is issued, the job aborts. For all other errors, the ROUTE macro is not executed but processing continues. If bit 12 of the flag field is set, an error code is returned and no dayfile message is issued. If bit 12 is not set, a dayfile message is issued and no error code is returned.

When a diagnostic is issued for the ROUTE macro, the message ERROR IN ROUTE FUNCTION LFN= is issued, followed by the message describing the error.

Error Code (octal)	Message
01	INVALID LFN -- DSP
02	CANT ROUTE NON ALLOCATABLE EQP
03	CANT ROUTE PERM FILE
04	NO PERMISSION TO ROUTE THIS FILE
05	ROUTE TO INPUT NOT IMMEDIATE -- IGNORED
06	IMMEDIATE ROUTING -- NO FILE -- IGNORED
07	INVALID DISPOSITION CODE -- ROUTING IGNORED
10	INVALID FID -- ROUTING IGNORED
11	DSP ABORTED BY SYSTEM
12	DSP PARAMETER OUTSIDE FL
13	PRIORITY SPECIFICATION IGNORED
15	EI200 SPECIFIED -- INTERCOM USED (DSP)
16	CAN NOT ROUTE INPUT FILE
17	DSP COMPLETE BIT ALREADY SET
20	FILE ON DISMOUNTABLE DEVICE -- ROUTING IGNORED
21	TID NOT ALPHANUMERIC -- ROUTING IGNORED
22	FORMS CODE NOT ALPHANUMERIC -- ROUTING IGNORED
23	INVALID LINK TYPE -- ROUTING IGNORED (DSP)
24	RESERVED
25	PRE-DAYFILE LFN AND NO DC=IN -- ROUTE IGNORED
26	PRE-DAYFILE FILE NOT FOUND -- ROUTING IGNORED

ACQUIRE MACRO

The acquire macro calls the PP routine QAF to search the input, print, punch, special (non-standard) 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 1-3 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.

ACQUIRE addr,recall,N.

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

recall Optional parameter whose presence indicates recall.

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

The file selection is based on particular parameters that describe the attributes of a file or group of files. The most important are file type (queue type), such as INPUT, OUTPUT, PUNCH, or special output; and priority.

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), terminal ID. Note that 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:

- a) Change routing of INPUT and/or OUTPUT queue files to the central site.
- b) Change routing of INPUT and/or OUTPUT queue files to another terminal.
- c) Change priority of OUTPUT queue files.
- d) Change forms code of OUTPUT queue files except for non-standard output (PLOT, FILM, etc.)
- e) Change repeat count of OUTPUT queue files except for non-standard output.
- f) 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 both a) and b) are specified, the result is as if only a) had been specified. Similarly, f) 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 12B 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 0101B (if an input file is attached) or 0102B (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 02B 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 reply 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 4420B, with reply entries stored in the user field length, beginning at REPBUF. QAF begins searching the FNT at FNT address 4420B 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 zero, 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 4730B, the FNT address is set to 4733B, 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, etc. 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 02B is inserted, and the queue type field remains clear.

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

57						47			41		35		29		23		11		
Filename															Priority			0	
Source ID					Destination ID					Reserved			FNT Ordinal			1			
Job Dependency ID		Dependency Wait Count			Maximum NT Drives		Maximum MT Drives		Reserved			Terminal ID			2				

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

58						53			41		23		11					
Filename															Priority			0
Source ID					Destination ID					Forms Code			FNT Ordinal			1		
Repeat Count		Disposition Code			Size of File					Reserved			Terminal ID			2		

File Interrupt Bit

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

59						41			34		29		23		17		11		
Jobname													Not Used		Priority			0	
Source ID					Time					FL/100B			Job Ordinal			1			
OP ACT					ERR		T Y	CP		STAT		Reserved			TID			2	

Word 0 Bits 59-18 Filename/Jobname
 17-12 Not used
 11-0 Priority

Word 1 Bits 59-42 Source ID
 41-24 Destination ID/Time left in quarter milliseconds for execution
 23-12 Forms code for output; not used for input; Job FL/100B for execution
 11-0 FNT ordinal/JDT ordinal

Word 2			
INPUT	Bits	59-58	Job dependency ID
		57-48	Not used
		47-36	Dependency wait count
		35-30	Maximum number of 9-track drives to be assigned at one time
		29-24	Maximum number of 7-track drives to be assigned at one time
		23-12	Reserved for CDC
		11-0	Terminal ID

Word 2			
OUTPUT	Bits	59	1 means file interrupted
		58-54	Repeat count
		53-42	Disposition code
		41-24	Size of file
		23-12	Reserved for CDC
		11-0	Terminal ID

Word 2			
EXECUTION	Bits	59-42	Operator action codes (SCOPE 2 only)
		41-36	Error flag values
			10B Kill
			4 Drop
			2 Rerun
		35	Type of job (0 = 7600 or CDC CYBER 70 Model 76; 1 = all others)
		34-30	Control point number the job currently occupies
		29-24	Job status
			70B Waiting for MMF action
			60B Waiting for pack mount
			40B Waiting for operator action
			30B Waiting for tape/device assignment
			20B Waiting for permanent file
			10B Waiting for time/event
			02B Executing
		23-12	Reserved for future TID expansion
		11-0	Terminal ID

COUNT

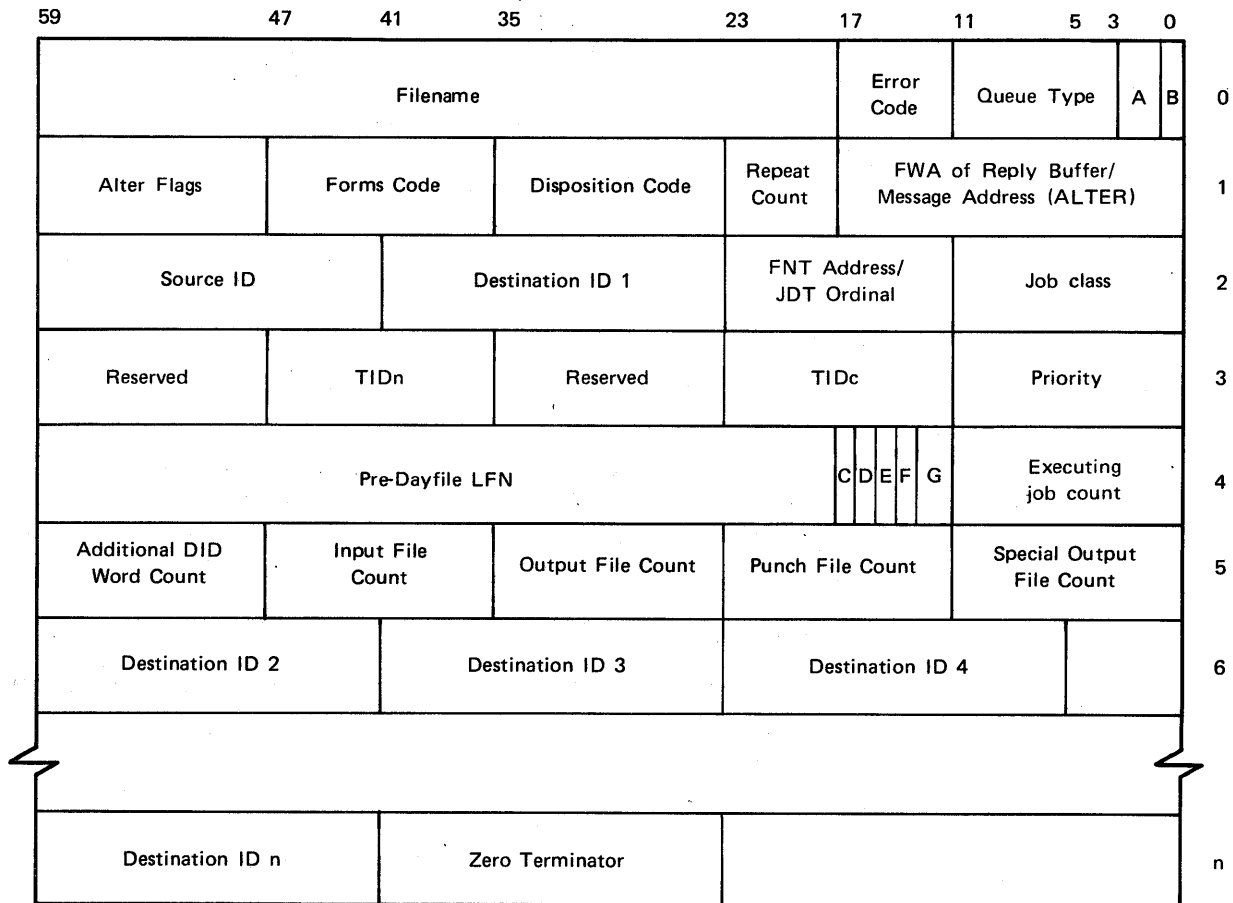
COUNT, function code 3, counts the number of queue entries of a specified type satisfying the selection criteria. Multiple queue type bits can be set on a single call giving the caller the count of each queue type

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 filename, file type, disposition code, source ID, destination ID, FNT address, and priority are returned for the first file that satisfies the selection criteria.

ACQUIRE PARAMETER LIST FORMAT

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 (bits 3-1)
- B Complete bit (bit 0)
- C Pre-dayfile bit (bit 17)
- D Class 2 Input file inhibit bit (bit 16)
- E Class 1 Input file inhibit bit (bit 15)
- F Inhibit duplicate file name search (bit 14)
- G Reserved (bits 13-12)

Parameter List Format

Word 0 Bits 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

Code	Message
------	---------

01B	Invalid queue type
02B	No queue entry found with specified parameters
03B	Function prohibits 7777B priority
04B	No FNT space
05B	Invalid reply entry buffer address
06B	Internal QAF error on FNT address
07B	Illegal Request
10B	Too many extra DID words
11B	PEEK requires single queue type
12B	Duplicate file name on GET
13B	Count of zero is invalid
14B	LFN needed for file having pre-dayfile
15B	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 zero. 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 zero and has a disposition code of 04B (INPUT job), 05B (INPUT tape job), or 06B (INPUT tape job on P display).
OUTPUT file	Has a valid FNT entry, is unlocked at control point zero and has a disposition code of 40B (any 501, 512, or 580), 41B (any 501), 42B (any 512), 43B (any 580-12), 44B (any 580-16), or 45B (any 580-20).
PUNCH file	Has a valid FNT entry, is unlocked at control point zero and has a disposition code of 10B (PUNCH 026 set from display code).
Special output file	Has a valid FNT entry, is unlocked at control point zero and has a disposition code of 20B (film print), 22B (film plot), 24B (hard copy print), 26B (hard copy plot) or 30B (plot).

3-1

Executing job has a valid JDT entry

Function Code

0	ALTER
1B	GET

2B PEEK
3B COUNT

0 Complete Bit

The complete bit must be cleared before any call to QAF. The bit is set on completion of any function.

Word 1 Bits 59-48

ALTER flags

Bit 53 Abort/evict job/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 2 Bits 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 5 Graphics job
4 Express job
3 Multiuser job
2 INTERCOM job
1 Batch job with non-alloc device requirement
0 Batch job without non-alloc device requirement

Word 3 Bits 59-36

TIDn New TID (used by ALTER only)

35-12 TIDc Current Terminal ID (used for search)

11-0 Priority

If zero 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 zero and less than 7777B 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 7777B, 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 03B. 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 4 Bits 59-18

Pre-Dayfile LFN

The pre-dayfile logical file name is required only if the qualifying INPUT file has a pre-dayfile. If the file has a pre-dayfile, a separate FNT entry is created to describe the pre-dayfile entry. If the pre-dayfile lfn is not given, the complete bit is set and the error code 14B is returned. If a pre-dayfile lfn is always in the field, the pre-dayfile flag must be checked after each call to prevent duplicating the file name in the FNT entries.

17 Pre-Dayfile 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 non-allocatable files associated with it. A class 2 INPUT file has at least one non-allocatable file associated with it. If the job card specifies MTxx, NTxx, or ECxxx, a non-allocatable 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

Word 5 Bits 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 – 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 6 Bits 59-42 Destination ID 2

41-24 Destination ID 3

23-6 Destination ID 4

5-0 Not Used

Word n Bits 59-42 Destination ID n

41-24 Zero Terminator

23-6

5-0 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 below:

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	-	-
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
Pre-Dayfile LFN	-	-	-/R†	-	-	-	-	-
Pre-Dayfile 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 pre-day file is present.

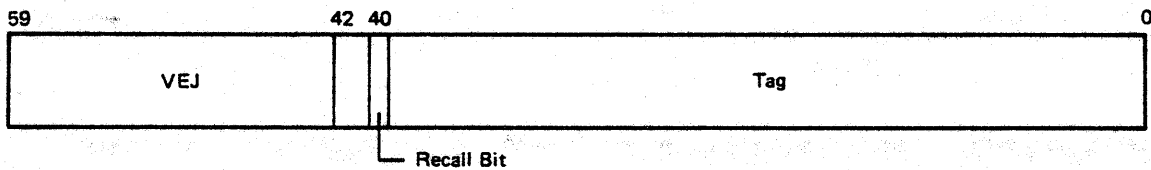
††Second symbol is for the special PEEK function.

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 jobname 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.

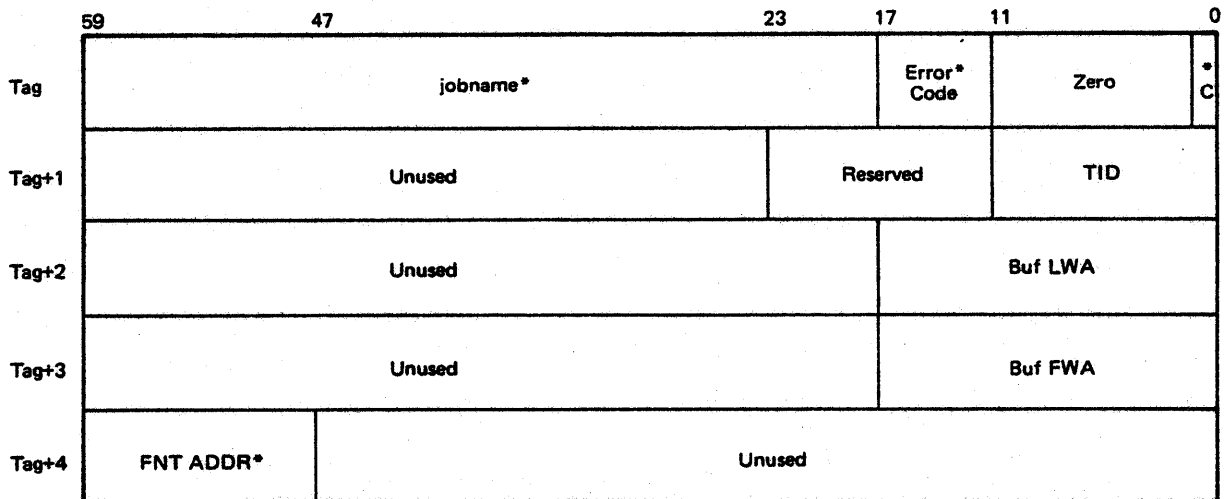
VERIFYJ has the following RA+1 interface:



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

VERIFYJ	tag,recall
tag	Address of the VERIFYJ parameter block
recall	Optional non-blank character indicating automatic recall

VERIFYJ PARAMETER BLOCK FORMAT



*These fields are returned by the VERIFYJ macro.

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 card 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 zero when macro used. 0 On call 1 On completion of call
tag+1	59-24	Unused	
	23-12	Reserved	
	11-0	TID	Terminal identification code
tag+2	59-18	Unused	
	17-0	Buf LWA	Address of the last word + 1 in buffer of the cards to be verified.
tag+3	59-18	Unused	
	17-0	Buf FWA	Address of the first word in the buffer of the cards to be verified.
tag+4	59-48	FNT ADDR	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.

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.

SCOPE I/O TABLES

SCOPE coordinates the requirements of input/output files 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 is 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 addresses of CM. Detailed descriptions of CM resident tables appear in part II, section 1. The FET and FIT are detailed in part II, section 2.

FILE ENVIRONMENT TABLE

Every file for which I/O is to be performed must have a FET. Each FET consists of a basic 5-word entry followed by additional words; the form depends on the type of I/O to be performed.

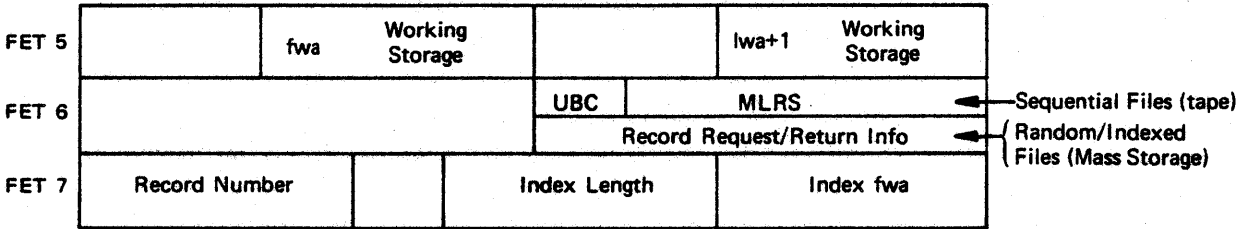
The basic 5-word FET entry is as follows:

	59	47	35	32	23	17	13	8	2	0	
FET 0	Logical File Name in Display Code					Record Levels	Error Flags	Code Status	MB		
FET 1	Device Type	Flag Bits	Disp. Code	LFET -5	FIRST						
FET 2	0				IN						} Buffer Parameters
FET 3	0				OUT						
FET 4	FNT Pointer	RB Size	PRU Size	LIMIT							

B = Busy (Free = 1)
 M = Mode (Binary = 1)

LIMIT ≤ FL
 OUT < LIMIT
 IN < LIMIT
 OUT ≥ FIRST
 IN ≥ FIRST

LFET -5 Length of FET minus 5 words for basic entry



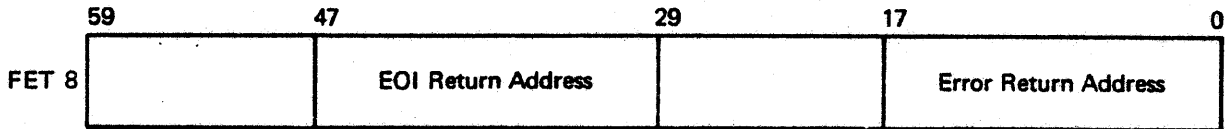
FET 5 is used for input/output blocking/deblocking by CPC

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

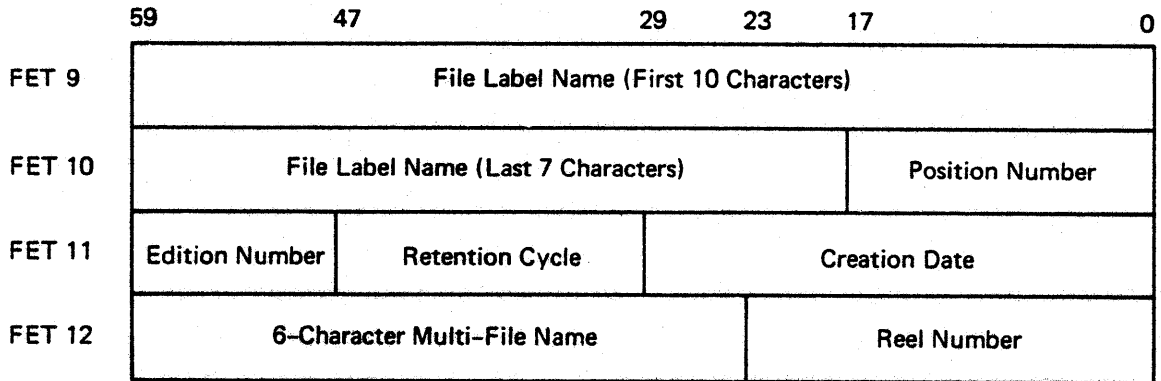
FET 6 and FET 7 are used for indexed file processing by CPC; FET 6 is used to pass RMS address between CP programs and system PP input/output routines

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

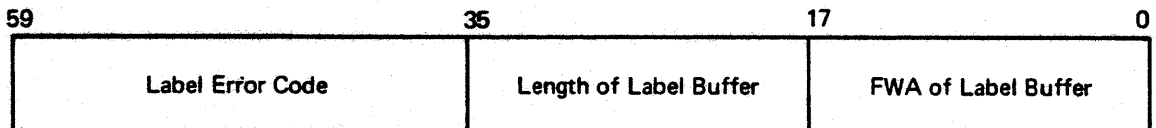
When the UP and/or EP flags are set in FET 1, then FET 8 contains:



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

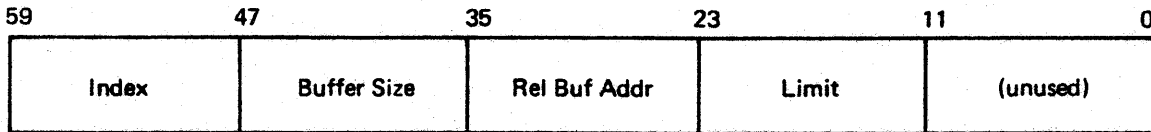


When LFET-5 flag in FET 1 is set to 1 for extended label processing, FET 9 has the following format:



SYSTEM FET ENTRIES

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 zero dayfile buffer. The format of the one-word dayfile FET entries is:

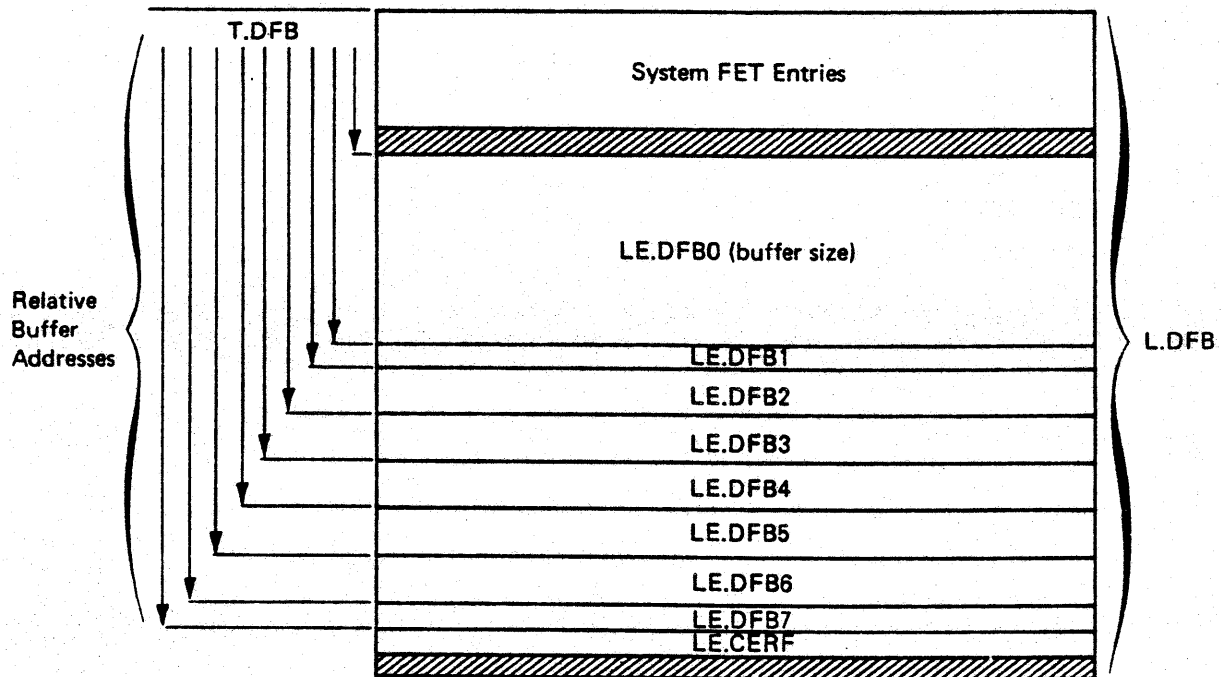


Index is the entry position relative to the table origin (T.DFB). Buffer sizes are set by SCOPE assembly configuration parameters internal to CMR. The origin of each buffer is found by adding the relative buffer address to the T.DFB origin address of the dayfile buffer area in CMR. The CP zero dayfile buffer is preset with a legend in the first eight words as follows: (b represents blank character)

```

b DAYFILE ::
bbb NORMAL b
( b b b b b b b )
DEAD b START
: : : : : : :
COPYRIGHT b
CONTROL b DA
TABCORP. b 1
970 : : : : :
    
```

The system dayfile area in CMR is diagrammed below:



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 the following figure.

T.FNT		Z	Z	Z	Z	Z	0	4	000000	0	0	0
		Control Point Zero										
		0 = Unlocked										
		11										
		0										
		Priority										
Device Type	0	FWA of RBT Word-Pair		Current RBT Word-Pair		Cur RBT Ordinal		Cur Byte		Current PRU		Status
	0	0		0		7 4 0 0		0 0		0		01
		D A Y F I L E		0		0 0 0 0 0 0		0 0		0		0
	0	0		0		7 4 0 0		0 0		0		01
		D F I L E		0 1		0 0 0 0 0 0		0 0		0		0
	0	0		0		7 4 0 0		0 0		0		01
		D F I L E		0 2		0 0 0 0 0 0		0 0		0		0
	0	0		0		7 4 0 0		0 0		0		01
		D F I L E		0 3		0 0 0 0 0 0		0 0		0		0
	0	0		0		7 4 0 0		0 0		0		01
		D F I L E		0 4		0 0 0 0 0 0		0 0		0		0
	0	0		0		7 4 0 0		0 0		0		01
		D F I L E		1 5		0 0 0 0 0 0		0 0		0		0
	0	0		0		7 4 0 0		0 0		0		01
		C E R F I L E		0		0 0 0 0 0 0		0 0		0		0
	0	0		0		7 4 0 0		0 0		0		01
		Z Z Z Z Z		0 3		0 0 0 0 0 0		0 0		0		0
	0	0		0		7 4 0 0		0 0		0 0 0 1		0
		Z Z Z Z Z		0 6		0 1 0 0 0 0		100100100100		0		LE.FNT-1 link addr
device type AX	20	0		0		7 4 4 0		0 0 0 1		0		ECS library entry if IP.ELIB ≠ 0
LINK addr		0 0 7 7 4 4 4 1 0		0		0		0		0		0

FILE NAME TABLE (FNT)

To provide linkage between user programs and all I/O processing routines, SCOPE maintains the FNT 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 6 or 9 words in length. The first word contains the logical 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 part II, section 1. The second and third words of the FNT entry are often called the FST (File Status Table) entry.

DEVICE TABLES

Tables in CMR that provide information on input/output equipment and channels are used by SCOPE to make file assignments. Tables included in this section: 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 part II, section 1.

EQUIPMENT STATUS TABLE (EST)

The Equipment Status Table resides in the upper table area of CMR and is pointed to by P.EST in the CMR pointer area. Table length is variable depending upon the system configuration at deadstart. 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, etc.

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 xx in the operator type-in n.ASSIGNxx.

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 Record Block Reservation (RBR) table, linking that table to the allocatable device it describes.

DISMOUNTABLE DEVICE TABLE (DDT)

The dismountable device table 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 Mounted Set Table 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 device status table 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 (ISP) 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 part II, section 1.

In word 2, if the chain start and chain end pointers are not equal, a PP is assigned to process the stack entry chain. The assigned PP address is given in byte 4 of word 2; when the entry is completed and the PP is dropped, the byte is set to zero.

The device status table is a key table in the processing of allocatable storage files. DST ordinals are found in the Device Activity Table (DAT), the Record Block Reservation (RBR) table header, and the Equipment Status Table (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.

When a device is dual access type, an additional DST entry is assigned to the device. This entry immediately follows the first DST entry and is called the Dual Access Status word pair. The format of a Dual Access Status word pair is shown in part II, section 1.

For a dual access scheme, two ISPs can be assigned to a DST entry and share accesses to the DST entry and the chain of stack requests and communicate with each other through the Device Pool Table.

The Dual Access Status word pair is mainly used for interlocking accesses to these common tables. Also, this word pair is used to hold connection information for the second controller and the location and length of the Device Pool Table.

Through the Unit Lock-out Table, accesses to particular units from particular controllers can be locked out while access is in dual mode.

DEVICE ACTIVITY TABLE (DAT)

The device activity table 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.

DAT entry for a dual access scheme is followed by a dummy entry in the same way that a DST entry is followed by a Dual Access Status word pair.

The formats for DAT entries are shown in part II, section 1.

CHANNEL STATUS TABLE (CST)

The Channel Status Table, 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:

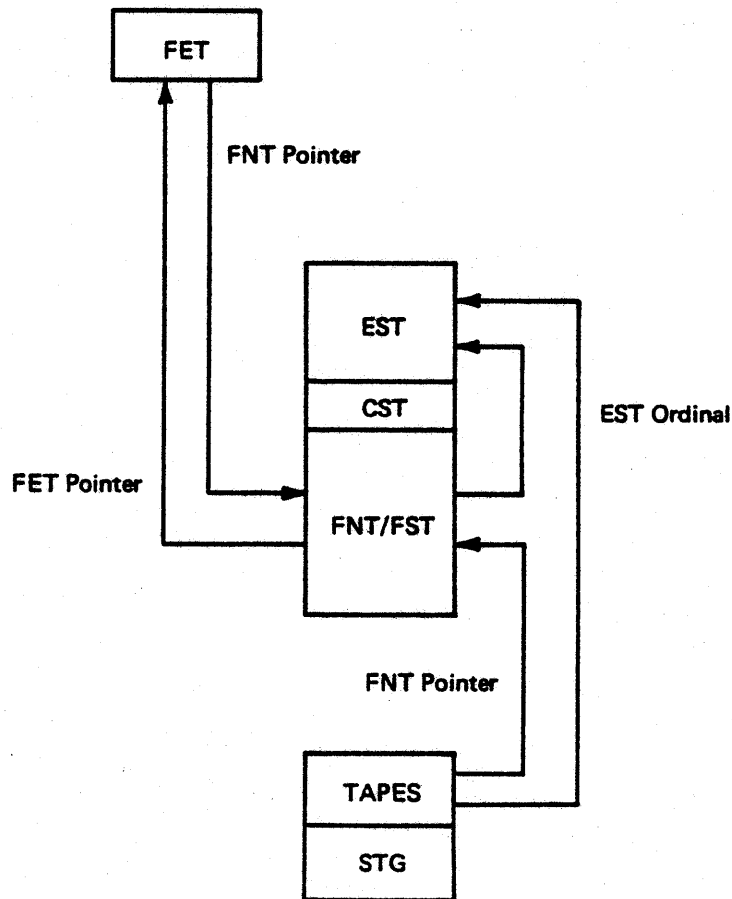
- Entry is added to FNT

- File is assigned to a control point, causing FNT modification

- FST code/status byte is initialized

Details of the Channel Status Table are given in part II, section 1.

Tables related to file processing on non-allocatable devices:



TAPES STAGING TABLE (STG)

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

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

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

The Tapes Staging Table appears in part II, section 1.

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 logical 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 logical 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 defined as one having MT and/or NT parameters on its job card. All incoming tape jobs are entered in a prescheduling queue, a subset of the input queue. The purpose of having a prescheduling queue is to advise the operator of tape reel requirements and to hold jobs in abeyance until such reels 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 type-ins 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 type-in. Once released, the job will be considered by assignment to a control point and execution; it will no longer appear in the pre-scheduling 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 activity uses less than the maximum number of drives necessary for job completion. Therefore, a job is assigned only those 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 in 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. Preventing deadlocks is a function of tape assignment, not job scheduling, although the job scheduling algorithm includes some built-in deadlock prevention features.

As part of REQUEST processing, SCOPE 3.4 provides a deadlock prevention algorithm. 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. Deadlock prevention refuses any tape assignment (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 evade deadlocks, but the resulting refusal of tape assignments would cause operator confusion and loss of efficiency. Job scheduling based on tape drive overcommitment, therefore, attempts to create an optimal situation and deadlock prevention avoids the worst-case situations.

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/non-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.

TAPES TABLE

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, then, 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, per IP.TSG.

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 will be automatically assigned to the requesting job and the job rolled back into CM to continue processing. While the job is rolled out, the operator may make a manual tape assignment which will cause the job to be rolled in automatically.

Dynamic tape drive status checking permits the automatic assignment of unlabeled tapes by volume serial number. 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 may be assigned without operator intervention to 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 his own set of removable disk packs.

NOTE

The SCOPE 3.4.3 category "private pack" no longer exists; it is superseded by private sets. All RMS devices are now allocatable (all devices can be divided among several jobs).

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 typein at any point in processing; masters may be dismounted when no jobs reference them.

Public sets remain mounted at all times and have either PF default, system, queue, or scratch attributes, or a combination thereof (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 may not 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 card to specify system. The user may 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, etc.; these files may not 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 REQUEST(lfn,*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, etc.) 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.

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 non-permanent 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.

A private device set is a group of RMS devices which 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.

Every device in either a public or a private device set can (but need not) be assigned one or more of the following device attributes:

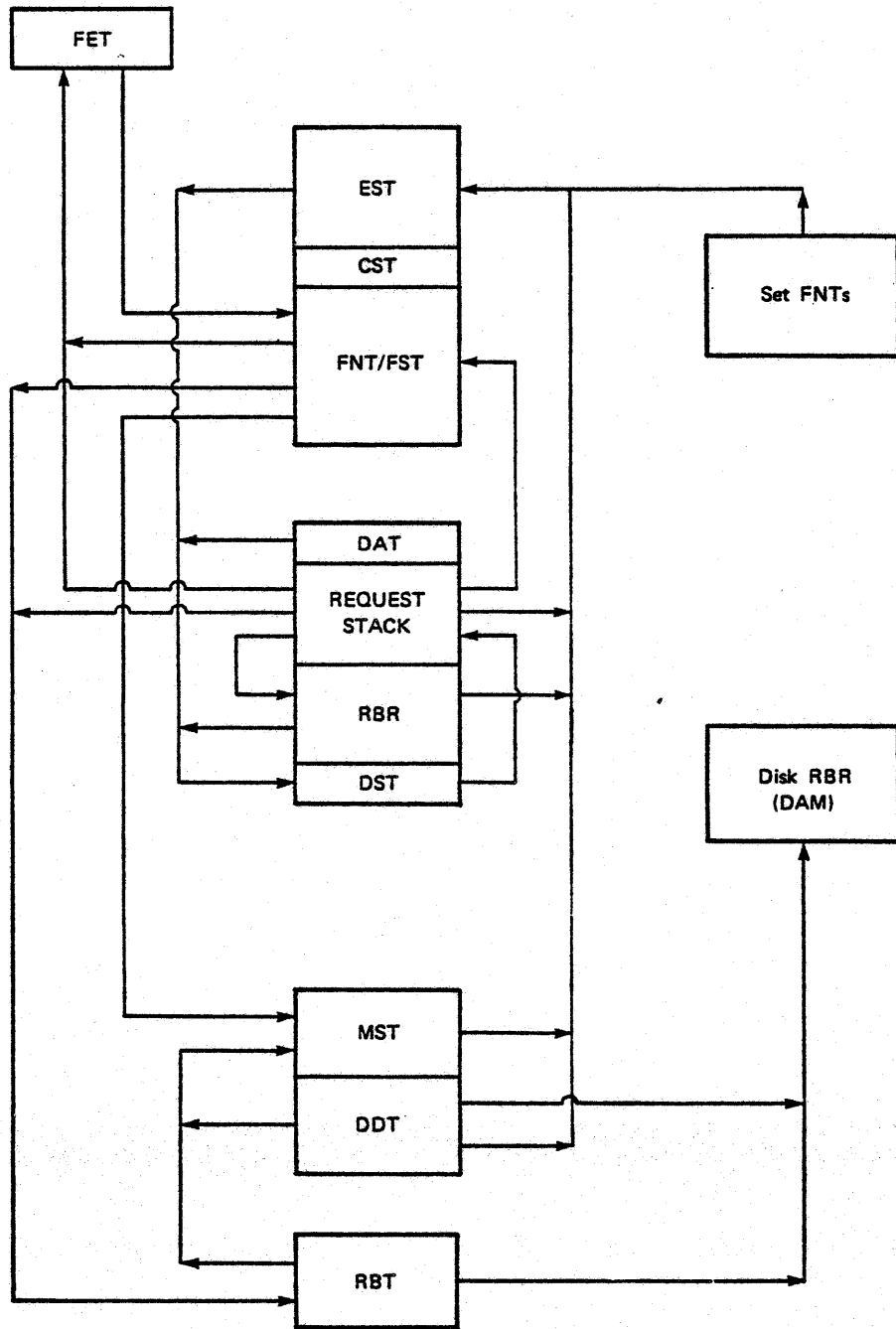
System Device	This device can contain the system files given above for the system set attribute. The system device attribute can be assigned only to public devices that are members of the system set.
Master Device	The master device contains system tables relating to its device set. These tables include the Device Label, the PFD (Permanent File Directory), the PFC (Permanent File Catalog), the SMT (Set Member Table), the DAM (Device Allocation Map), the PFT (Physical Flow Table), and the LFT (Logical Flow Table). Every device set must have a master device.
Permanent File Device	This device can contain files for which the REQUEST control card specifies *PF.
Queue Device	This device can contain files with names such as INPUT, OUTPUT, and PUNCH, files with non-zero disposition codes, and files for which the REQUEST control card specifies *Q.

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.

SHARED DEVICE SETS

In a dual-mainframe system, certain device sets can be shared between mainframes. Such sets must consist entirely of 844 devices and 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, not more than one mainframe can access a device during this process. 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 the 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.

RMS TABLES



To comprehend the functions of the various tables involved, the methods of mass storage space allocation must be understood. Terms are defined below:

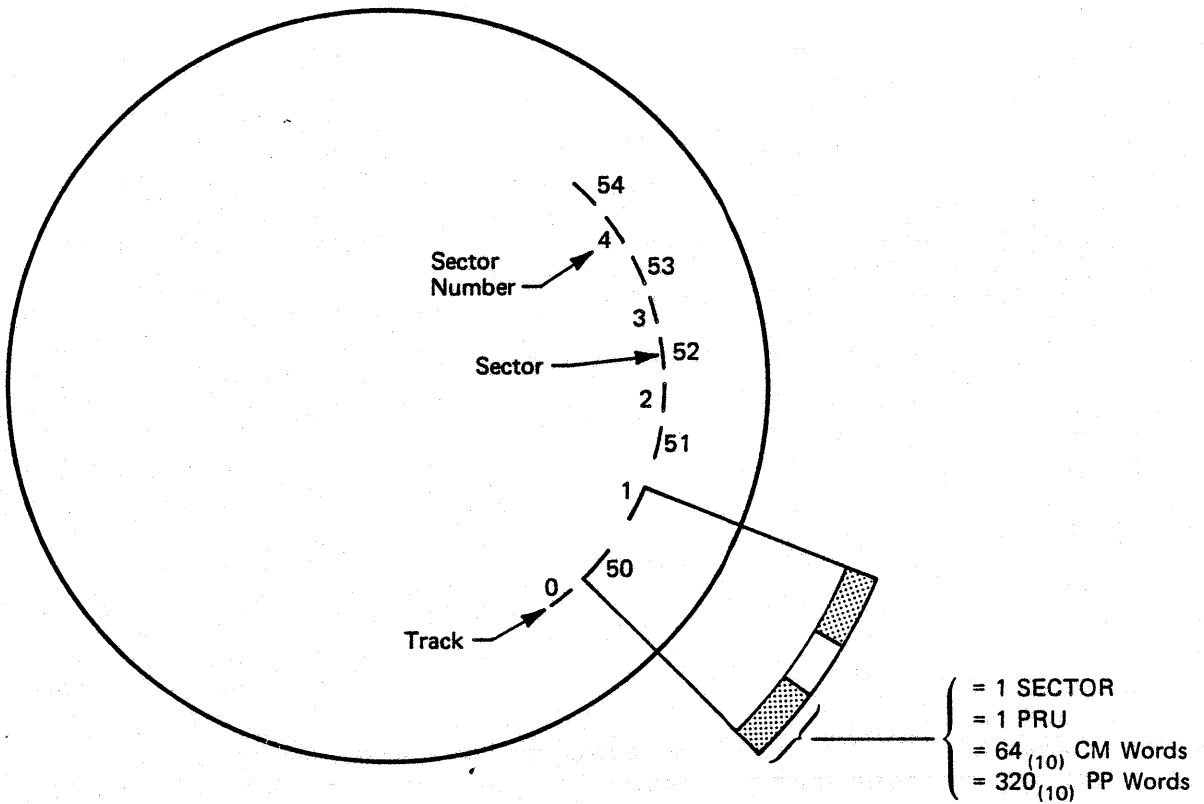
SECTOR: the smallest accessible physical space increment on a track of a rotating mass storage device.

PRU: (physical record unit): the smallest amount of data a user may access; it is 64 decimal (100 octal) central memory words and is usually equal to 1 sector.

RB (record block): the smallest amount of mass storage that can be allocated. An RB, defined in the RBR header, is several PRUs in length.

RBR (record block reservation): a bit-coded table which indicates those RBs on a device which are assigned to a file, flawed (defective), or available for assignment. A 0-bit indicates that the specific RB is available for assignment. The number of physical record units in a standard RB (default RB size) is given in the following table:

Device	RMS Type	Type/Device Codes	Default PRU/RB (decimal)
6603-I	DISK	AA 01	**
6638	DISK	AB 02	50
6603-II	DISK	AC 04	**
821	DISK	AL 05	320
841	DISK PACK	AM 06	56
854	DISK PACK	AP 07	4
844	DISK PACK	AY 13	56
**	INNER ZONE IS 50 PRU/RB OUTER ZONE IS 64 PRU/RB 2 RBRs ARE REQUIRED		



Half-Track Recording Technique

Alternate sectors of a physical recording track on 6603 or 6638 disks are numbered consecutively; intervening sectors are numbered the same way. Essentially, one recording track is divided into two half tracks, one containing the odd-numbered sectors, the other even-numbered sectors. The time required to move to the next numbered sector is used to set up parameters and load or unload PP memory in preparation for the read/write of the next sector.

RECORD BLOCK RESERVATION TABLE

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.

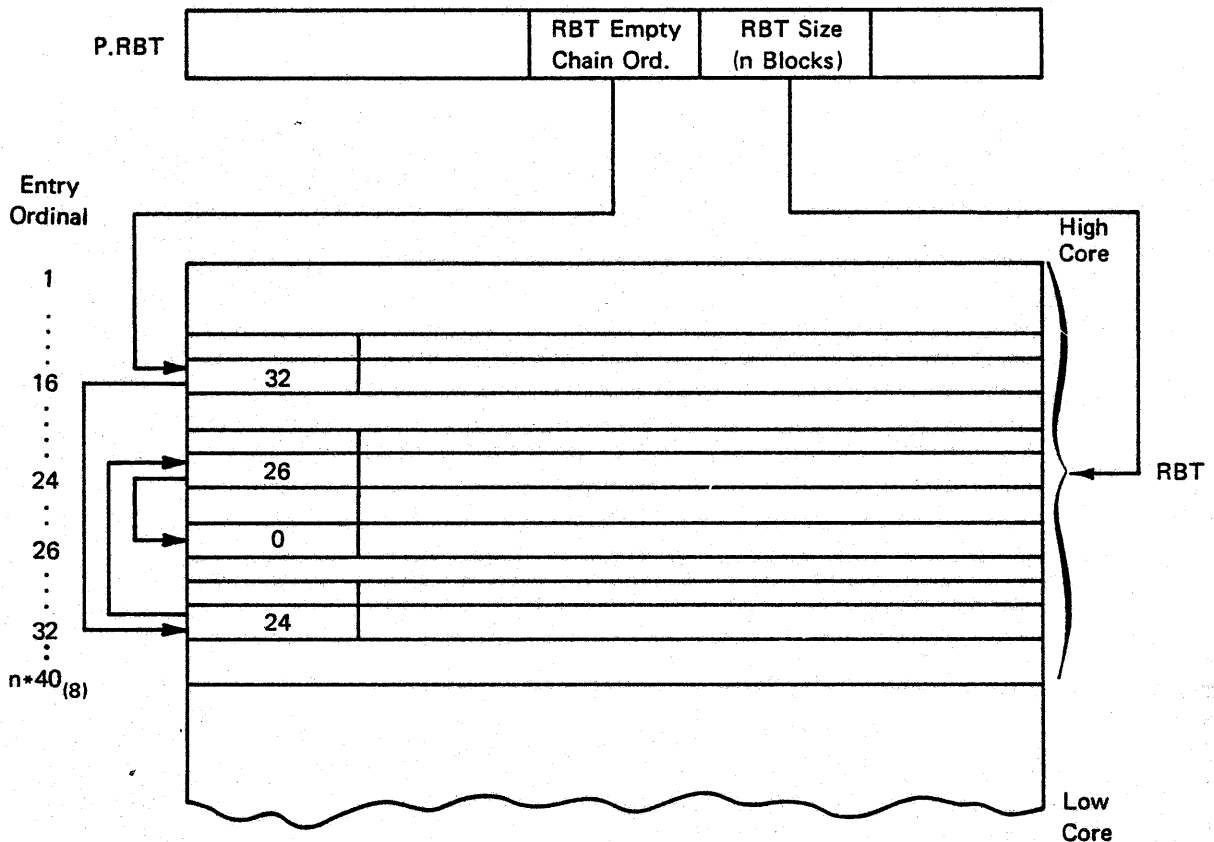
The first word of each RBR header contains a 6-bit allocation style code supplied as a parameter to the RBR macro when the CMR is assembled at an installation. Unique allocation style codes for each RBR may be set by the installation; this code can be used to direct a file to the RBR with a specific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

RECORD BLOCK TABLE (RBT)

The Record Block Table (RBT) is file-oriented. Each mass storage file in the system has an associated RBT chain. The RBT, located in the highest address end of central memory, consists of word pairs which are linked forward to form an RBT chain for each file that exists on an allocatable device currently recognized by the system. The RBT expands and contracts by 100 (octal) word blocks as files are allocated and released. A maximum of 8192 (decimal) central memory 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 central memory.

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 together 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.



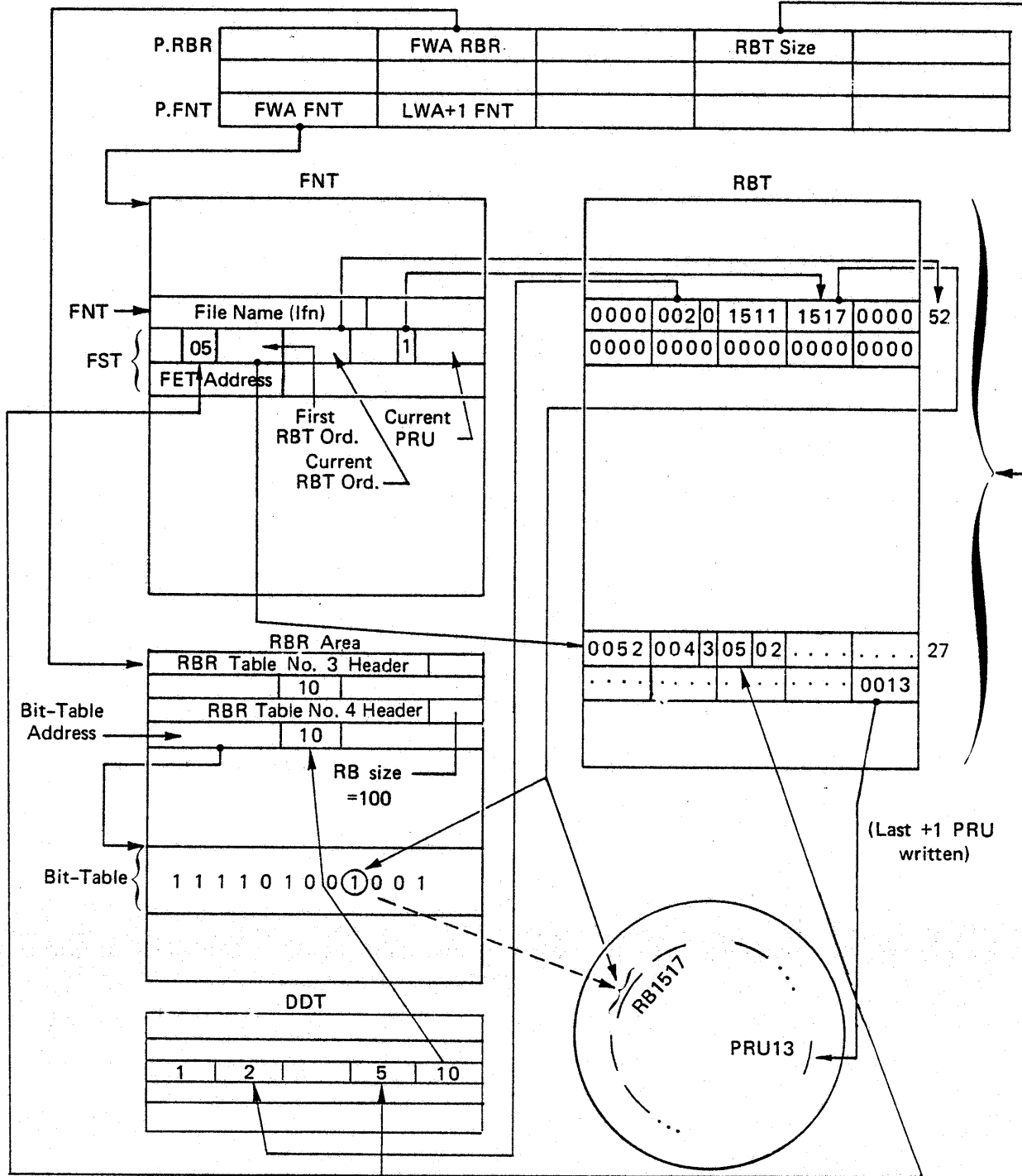
Word 1 of the first word pair assigned to a file contains ordinals, flags, etc. The RB bytes denote the record blocks assigned to the file. These bytes, initially zero, are set as each record block is assigned. The values in the RB bytes are RB numbers which indicate the physical address on the device 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.

The interrelationship of the FNT, RBT, and RBR entries for a file assigned to an outer zone half track on a 6603 disk is illustrated in figure 4-2. Pointers in the lower table area of CMR give the first word address of the FNT and the RBR area. The RBT size, in 100B word blocks, is multiplied by 40B to yield the number of word pair ordinals in the RBT. Multiplying any RBT ordinal by 2 and subtracting it from the LWA+1 address of central memory will produce the address of the word pair entry.

CMR



FNT/RBT/DDT/RBR Interface
for a Sequential File on a
6603 RMS Outer Zone

Figure 4-2. File Table Interfaces
(all numbers are octal)

When the file is established for a job, an entry is made in the FNT; and a pointer to this entry is placed in the job's FET for the file. The FST entry in the FNT for a mass storage file contains the RBT ordinal of the first word pair assigned to the file (27 in this example), and the current word pair assigned. It also includes the RBn byte number representing the current position of the file and the current PRU number in the record block being accessed. RB bytes are numbered from 0 through 7 starting with the middle byte of the first word of the pair. In this example, the current RBT ordinal is 52 and the RBn byte number is 1. Byte 1 in the first word of any file RBT word pair contains the ordinal of the DAM table entry and the MST table entry used by the file. The DDT has a range of DAM ordinals associated with that device. The DDT points to the EST; the RBR points to the EST. Thus, the RBR ordinal can be found from the DAM.

The current record block in the file is found by using the RB number in the RBn byte of the RBT word pair currently being accessed. These RBT pointers are in the FST entry; the example showing word-pair ordinal 52, byte RB1. Only 3 bits in the FST are needed to point to one of the 8 RB bytes in the RBT word pair. The RB number in byte RB1 is 1517 octal or 847 decimal. To find the RBR table entry, find the DDT entry with the same MST ordinal (5 in this example), the DAM within range (2), and the RBR pointing to the same EST (10). Since the first DAM fits on one PRU, the next DAM is the RBR for this file (ordinal 4). To find the corresponding bit in the RBR bit table, subtract one from 847 to get 846, divide by 60 to get 14 and a remainder of 6. Then the RBR bit table entry is bit 53 (59 minus the remainder) in the fifteenth word (fwa + 16 octal) of the bit string. To find the corresponding physical address on the device, subtract one from 1517 to get 1516 and apply the formulas given in Part II, section 1, under RECORD BLOCK TABLE ENTRY. According to the RB address format for a 6603 disk, the physical address is outer zone, head position 151 of head group 3, and consists of even-numbered PRUs. A byte in the RBT points to the last PRU written plus one; a byte in the FST points to the PRU being accessed in the current record block.

INTERCOM TABLES

INTERCOM uses word 16B 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 itself.

The INTERCOM tables are detailed in Part II, section 1.

When the file is established for a job, an entry is made in the FNT; and a pointer to this entry is placed in the job's FET for the file. The FST entry in the FNT for a mass storage file contains the RBT ordinal of the first word pair assigned to the file (27 in this example), and the current word pair assigned. It also includes the RBn byte number representing the current position of the file and the current PRU number in the record block being accessed. RB bytes are numbered from 0 through 7 starting with the middle byte of the first word of the pair. In this example, the current RBT ordinal is 52 and the RBn byte number is 1. Byte 1 in the first word of any file RBT word pair contains the ordinal of the RBR table entry used by the file. The ordinal points to a two-word table header in the RBR area which, in turn, points to the bit string for the area on the device to which the file is assigned.

The current record block in the file is found by using the RB number in the RBn byte of the RBT word pair currently being accessed. These RBT pointers are in the FST entry; the example showing word-pair ordinal 52, byte RB1. Only 3 bits in the FST are needed to point to one of the 8 RB bytes in the RBT word pair. The RB number in byte RB1 is 1517 octal or 847 decimal. To find the corresponding bit in the RBR bit table, subtract one from 847 to get 846, divide by 60 to get 14 and a remainder of 6. Then the RBR bit table entry is bit 53 (59 minus the remainder) in the fifteenth word (fwa + 16 octal) of the bit string. To find the corresponding physical address on the device, subtract one from 1517 to get 1516 and apply the formulas given on the bottom of figure 4-3. According to the RB address format for a 6603 disk, the physical address is outer zone, head position 151 of head group 3, and consists of even-numbered PRUs. A byte in the RBT points to the last PRU written plus one; a byte in the FST points to the PRU being accessed in the current record block.

INTERCOM TABLES

INTERCOM uses word 16B 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 itself.

The INTERCOM buffer area is in two partitions. The lower partition contains 20B-word buffers and the upper partition contains 100B-word buffers. Both partitions are totally dynamic; they are assigned and released on demand. The large buffers are used for the INTERCOM wide band subsystem and are not present if INTERCOM is not running wide band. The small buffers are used for all data transfers except wide band, and the small buffers are always present if INTERCOM is in the system.

The INTERCOM tables are detailed in part II, section 1.

I/O PHILOSOPHY

Input and output request processing depends upon the source of each request. Active user CM programs issue RA + 1 requests for I/O 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.CP + 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 ISP. CIO (circular input/output) processes requests for magnetic tape, Teletype, and unit record I/O; and ISP (stack processor) processes all requests for mass storage I/O.

Another I/O processor, JANUS, exists in SCOPE, 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 of SCOPE. JANUS is discussed under the job processing section of this manual.

CIO

The circular input/output processor consists of the central memory 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; MTR passes it through the CP input register for the CP.MTR. A CP program places a CIO call in the CP request register (RA+1). When MTR 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 one to the code/status field of FET word one. 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).

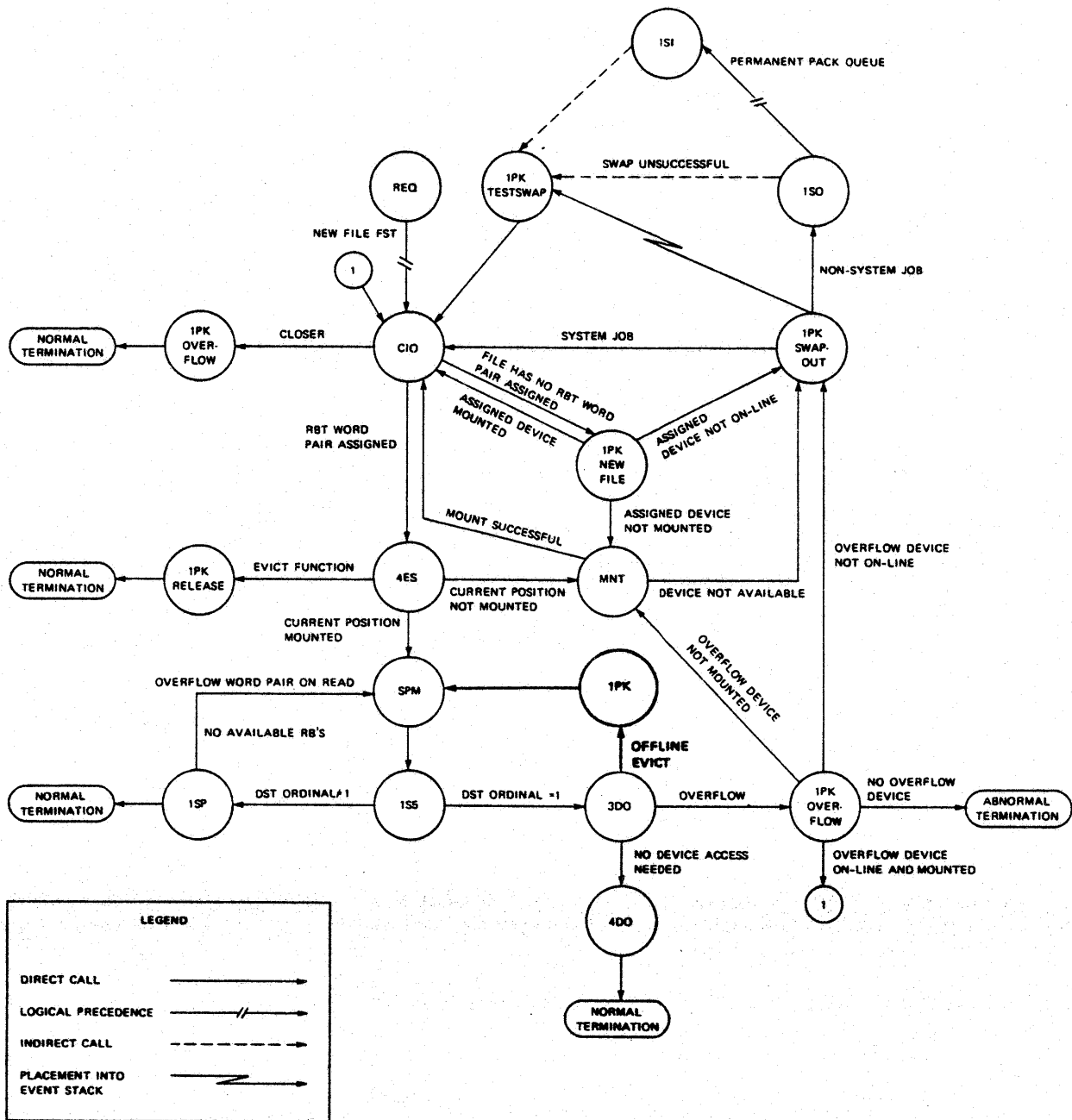


Figure 5-2. Device Set I/O Processing

SCOPE CIO CODES (3.4)

All codes indicated by * are illegal; all reserved codes are illegal. All codes are octal for coded mode operations; add 2 for binary mode. 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	*		

200 Series 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				

300 Series 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

400 Series (Reserved for CDC)

500 Series (Reserved for Installations)

600 Series

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

700 Series (Reserved for CDC)

CIRCULAR BUFFER

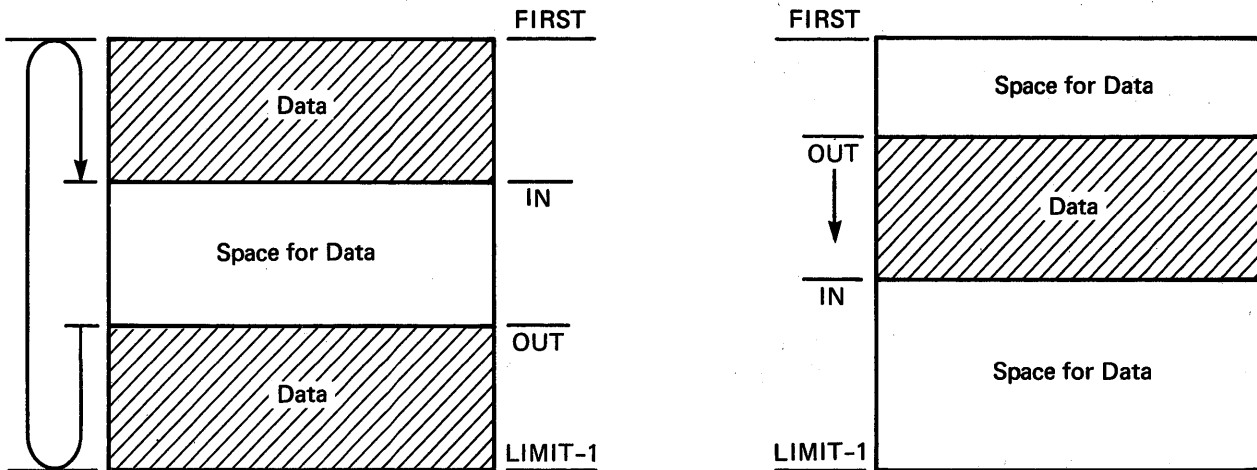
A circular buffer is a temporary storage area in central memory through which data passes during I/O operations. It is termed circular because I/O processing routines treat the last word and the first word of the buffer area as contiguous.

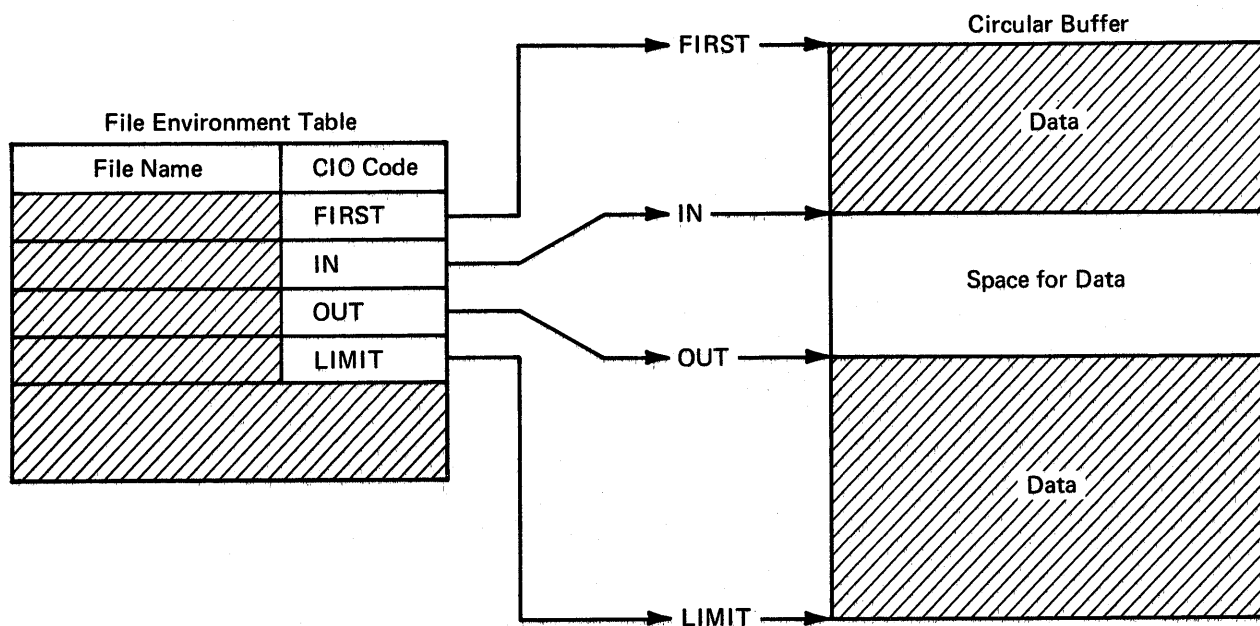
FIRST is the first word address of the circular buffer. Routines that process I/O never change the value of **FIRST**.

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.

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 non-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 will also check 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 will search the entire FNT for a file assigned to that job control point with a matching name. If the file is not found, CP.CIO will create a 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 will perform the I/O and set the completion bit. OV.CIO and its overlays process I/O requests for files on nonallocatable devices.

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

Function routines:

1CL	File close (nontape files)
1OP	File open (nontape files)
1TO	Tape open
1MF	Multifile positioning
1RP/2RP	Reel close -- EOR processor
1TF	Move tape forward (except long record (L) tapes)
2TB	Move tape backward (except long record (L) tapes)
3DO	Mass storage device file open
3IC	File close for MTS tapes
3IF	Multifile positioning for MTS tapes
3IL	Slave for 3IO, 3IC, and 3IV
3IM	Write error message for MTS tape
3IO	Tape open for MTS tapes
3IV	Reel close -- EOR processor for MTS tapes
4ES	Enter stack request (mass storage I/O)
6WM	Write error message

Tape Drivers:

1IT	Main MTS tape driver, calls the nix overlays
1RS	Read 7-track stranger (S) tapes
1RT	Read 7-track SCOPE standard labeled tapes
1MT	Read/write other tapes (7-track)
1NR	Read 9-track stranger (S) tapes
1NW	Write 9-track stranger (S) tapes
1WS	Write 7-track stranger (S) tapes
1WI	Write 7-track SCOPE standard labeled tapes
1R9	Read 9-track SCOPE standard labeled tapes
1W9	Write 9-track SCOPE standard labeled tapes
1TS	Tape sampler
2IA	L tape read -- MTS tapes
2IB	L tape write -- MTS tapes
2IC	Coded SCOPE read (7-track) -- MTS tapes
2ID	Coded SCOPE write (7-track) -- MTS tapes
2IL	Label read/write -- MTS tapes
2IR	Normal read -- MTS tapes
2IW	Normal write -- MTS tapes

Unit record drivers:

2PC	On line card punch
2RC	On line card reader
2LP	On line printer

Tape error recovery drivers:

1P1/3PM	Write error recovery – tape LGR positioning driver
1P2	Write error recovery – erase/rewrite driver
1P3/3PO	Write error recovery – verification driver
1P4/3PS	Write error recovery – final LGNR positioning driver
1RV	Initialize/terminate read error recovery
1R2	Read parity error recovery
1R3	Read error recovery – reposition/reread
1NO	Noise error recovery – read error processing
1N2	Noise error recovery – read recovery driver
1N3	Noise error recovery – skip noise record
1CS	Write CM data – 7/9 track stranger tape read recovery
1CT	Write CM data – 7/9 track SCOPE standard tape read recovery
1CR	Write CM data – 7-track other tape read recovery
3IE	Error diagnosis – MTS tapes
3IR	Read recovery – MTS tapes
3IW	Write recovery – MTS tapes

If the file device code is for a non-allocatable 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 SCOPE standard format 7-track tape, CIO will call the overlay 1RT into its PP. 1RT will reserve one of the hardware channels connected to the equipment. It then issues the function codes to connect the controller and tape drive. 1RT issues functions to transmit one PRU of data from the tape drive over the data channel.

1RT accumulates the PRU of data in a PP buffer. When the entire PRU is transmitted or an end-of-record (short PRU) is encountered, 1RT picks up the pointers to the circular buffer in central memory from the FET. 1RT 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. 1RT updates the PRU count in the file FNT, releases the channel, sets completion bits in the FNT and FET, and drops out.

The following charts depict the logical sequence of events during various CIO tape operations.

READ

	Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded
1. Exit if not enough room in buffer for one maximum size physical record.	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	x	x		
4. Read one physical record into CM.					x	x
5. If physical record exceeds maximum allowable, return error status DEVICE CAPACITY EXCEEDED and perform error procedures.	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
7. If end-of-file mark was read, perform end-of-file mark procedures.	x	x	x	x	x	x
8. If noise records encountered, go to 3.	x	x	x	x	x	x
9. If parity error, perform parity procedures.	x	x	x	x	x	x
10. If end-of-tape reflective spot was encountered and tape is unlabeled, perform end-of-reel procedures.			x	x	x	x
11. If short PRU was read, strip level number.	x	x				
12. If zero length PRU was read, go to 21.	x	x				
13. When 6681 is present, convert data in PP from BCD to display code.		x		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				
16. Transmit data to CM.	x	x	x	x		
17. Update IN.	x	x	x	x	x	x

READ (Continued)

	Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded
18. Fetch OUT from CM.	x	x				
19. Place in word 7 of FET the number of unused bits in the last data word.			x	x	x	x
20. If full PRU, go to 1.	x	x				
21. If last record was level 17 of 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.	x	x	x	x	x	x

READN

	S Binary	S Coded	L Binary	L Coded
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 $> \ell(\text{record}) + 1$ (header) to avoid OUT = IN when buffer is full.	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, 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
7. If end-of-file (tape mark) was read, perform end-of-file mark procedures. Go to 18.	x	x	x	x
8. If noise records encountered, go to 3.	x	x	x	x
9. If parity error, perform parity procedures.	x	x	x	x
10. If end-of-tape reflective spot was encountered on unlabeled tape, perform end-of-reel procedures.	x	x	x	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
13. Transmit data to CM.	x	x		
14. Update IN in PP memory.	x	x	x	x
15. Place in buffer header word, length of record and number of unused bits in last data word.	x	x	x	x
16. Update IN.	x	x	x	x
17. Fetch OUT.	x	x	x	x
18. If last record was tape mark, set end-of-file status and exit.	x	x	x	x
19. Go to 2.	x	x	x	x

READSKP

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

READSKP (Continued)

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

RPHR

	Standard Binary	Standard Coded
1. Set OUT = IN.	x	x
2. Exit if not enough room in buffer for one maximum size physical record.	x	x
3. Read one physical record into PP.	x	x
4. If physical record exceeds maximum allowable, 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
5. If end-of-file mark was read, perform end-of-file mark procedures.	x	x
6. If noise records encountered, go to 3.	x	x
7. If parity error, perform parity procedures.	x	x
8. If zero length PRU was read, go to 13.	x	x
9. Transmit data to CM.	x	x
10. Update IN.	x	x
11. If last record was level 17 or tape mark, set end-of-file status.	x	x
12. Exit.	x	x

WRITE

	Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded
1. Exit if not full PRU.	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	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 present, convert display code to BCD in PP memory.		x		x		
7. When 6681 present, convert from display code to BCD in CM.						x
8. Convert zero byte line terminator to 1632.		x				
9. Write record to tape.	x	x	x	x		
10. Write, from CM to tape, data contained between OUT and IN, adjusted by unused bit count.					x	x
11. When 6681 present, convert data in CM buffer back to display code.						x
12. If parity error, perform parity procedures.	x	x	x	x	x	x
13. If end-of-tape reflective spot, perform end-of-reel procedures.	x	x	x	x	x	x
14. Update OUT.	x	x	x	x	x	x
15. Exit.			x	x	x	x
16. Fetch IN from CM.	x	x				
17. Go to 1.	x	x				

WRITER

	Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded
1. If IN = OUT, exit.			x	x	x	x
2. If PRU not full, insert level number in PP buffer.	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	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 display code to BCD in PP memory.		x		x		
8. When 6681 is present, convert display code to BCD in CM.						x
9. Convert zero byte line terminator to 1632.		x				
10. If IN = OUT, write zero length record. Go to 12.	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.						
13. When 6681 is present, convert data in CM buffer to display code.						x
14. If parity error, perform parity procedure.	x	x	x	x	x	x
15. If end-of-tape reflective spot, perform end-of-reel procedures.	x	x	x	x	x	x
16. Update OUT.	x	x	x	x	x	x
17. Exit.			x	x	x	x
18. If full PRU is not written, exit.	x	x				
19. Go to 1.	x	x				

WRITEF

	Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded
1. If no data from OUT to IN, go to 20.	x	x				
2. If no data from OUT to IN, go to 17.			x	x	x	x
3. If not full PRU, insert 0 level number.	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.			x	x	x	x
6. Fetch one PRU of data starting at OUT or 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 display code to BCD in PP memory.		x		x		
9. When 6681 is present, convert display code to BCD in CM.						x
10. Convert zero byte line terminator to 1632.		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	x
13. When 6681 is present, convert data in CM buffer to display code.						x
14. If parity error, perform parity procedures.	x	x	x	x	x	x
15. If end-of-tape reflective spot, perform end-of-reel procedures.	x	x	x	x	x	x
16. Update OUT.	x	x	x	x	x	x

WRITEF (Continued)

	Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded
17. Write end-of-file mark and exit.			x	x	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 was WRITE, write zero length PRU.	x	x				
21. Go to 17.	x	x				

WRITEN

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

WPHR

	Standard Binary	Standard Coded
1. If IN = OUT, exit.	x	x
2. If more than 512 words in buffer, return DEVICE CAPACITY EXCEEDED to FET.	x	x
3. Fetch data from OUT to IN, or 512 words from OUT, whichever is smaller.	x	x
4. Write record to tape.	x	x
5. If parity error, perform parity procedures.	x	x
6. If end-of-tape reflective spot, perform end-of-reel procedures.	x	x
7. Update OUT and exit.	x	x

SKIPF

	Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded
1. If $n = 0$, set $n = 1$.	x	x	x	x	x	x
2. If L is less than 17, interpret as L equals 0.			x	x	x	x
3. Read a physical record.	x	x	x	x	x	x
4. If noise record encountered, go to 3.	x	x	x	x	x	x
5. If end-of-tape reflective spot encountered on unlabeled tape, perform end-of-reel procedures.			x	x	x	x
6. If record is full PRU, go to 3.	x	x				
7. If end-of-file mark 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
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 $n \neq 0$, 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.	x	x	x	x	x	x

SKIPB

	Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded
1. If $n = 0$, set $n = 1$.	x	x	x	x	x	x
2. If L is less than 17, interpret as L equals 0.			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 encountered, go to 4.	x	x	x	x	x	x
6. If record was 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				
9. If end-of-file mark encountered, assume level number = 17. Otherwise, assume level number = 0.			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	x	x
12. Exit.	x	x	x	x		

BSKP

The BSKP function is identical to SKIPB with n = 1 and L = 0.

BKSPRU

	Standard Binary	Standard Coded	S Binary	S Coded	L Binary	L Coded
1. If at load point or PRU count = 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 not equal to 0, go to 1.	x	x	x	x	x	x
4. Exit.	x	x	x	x	x	x

ALLOCATABLE DEVICE I/O

Most files in the system are stored on allocatable units. The system library ZZZZZ04 is stored on an allocatable unit known as the system device. Each time a non-CM resident CP program or a PP overlay is to be loaded from that library, I/O must be performed on the system device. The job and system dayfiles and the CE error files are stored on allocatable units, as are all input and output queue files and all files created by CIO.

A request for I/O on a mass storage allocatable unit must be placed in a table, called the request stack. The stack is searched, and the request which will require 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 units 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 units 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. Since READC uses inter-sector time to the maximum while reading high-speed mass storage devices, it does not include checks for erroneous programming and control words. READC would only be used by system programmers.

READC lfn,recall

READC transmits PRU's continuously to the circular buffer, with a control word preceding each PRU. READC is a function applicable to all mass storage devices. Reading continues until:

Buffer does not have enough room for the next PRU and its control word.

An error condition occurs.

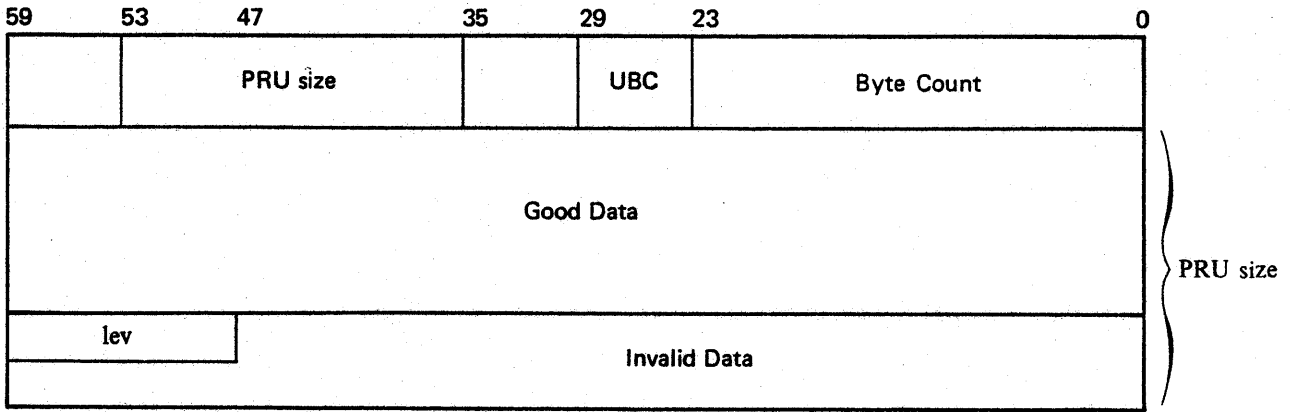
End-of-information is encountered.

Code and status on completion; where x depends on file mode:

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

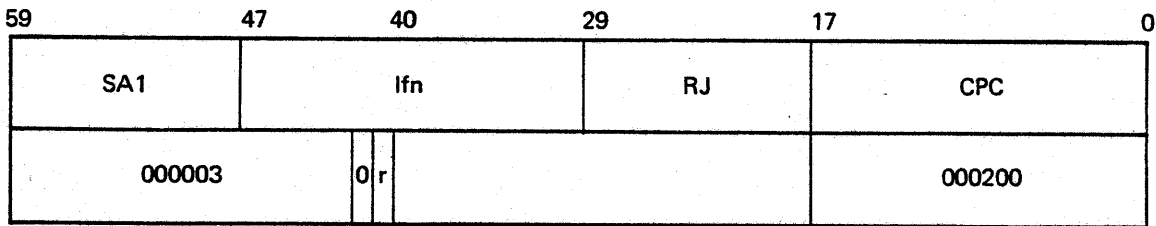
On mass storage, the same amount of data is transmitted for every PRU: the control word and one device standard PRU. The last 12 bits of the control word and the entire standard PRU length are exactly the physical data recorded on the device, including system control information.

Format of the PRU is shown below.



- PRU size 64 central memory 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 central memory words occupied by the data. The value is recorded on disk as 12 bits, but expanded here to 24 bits.
- lev SCOPE logical record level number. If byte count divided by 5 is a full PRU, lev does not exist.

The READC macro generates the following code:



READLS

The READLS function is applicable 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 will reveal boundaries. This function should be used by system programmers only.

READLS lfn, recall

Before READLS is called, bits 0-17 of FET+6 should be set to the address of the list of addresses to be read.

Reading continues until:

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 will contain 01, 10, or 11. 10=end of record, 11=end of file; giving the status at the point where the operation terminated. The operation will terminate with EOI status if the last PRU of the file is read and no EOR or EOF occurs. The contents of bits 5 through 8 do not pertain to this description.

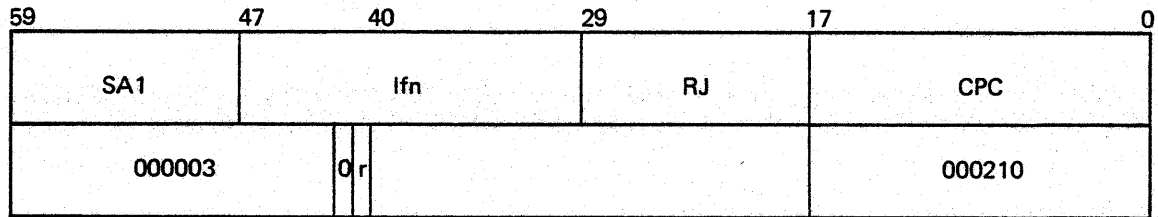
The address pointer is updated by the system when a READLS terminates so that the function can be reissued by the user without the user changing the pointer. The updated pointer will reflect the next record to be read. If reading stopped in the middle of a record, the pointer will reflect the next position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

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

Bits 36-59 contain a PRU number, the same as used in SCOPE indexes. These are the numbers the system returns to the record request/return information field (bits 0-29 of word 7) of the FET when records are written on a mass storage device. Bits 0-35 are zero. A user list in this format will be converted by the system to the next format.

Bits 0-35 contain the SCOPE internal direct access address (RBTA/RBB/PRU) address RBT.

The READLS macro generates the following code:



CONTINUOUS WRITE (WRITEC)

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

WRITEC lfn,recall

WRITEC transmits PRU's from the circular buffer to a mass storage device. Each PRU in the buffer must be preceded by a control word. Writing continues until:

Buffer is empty

An error occurs

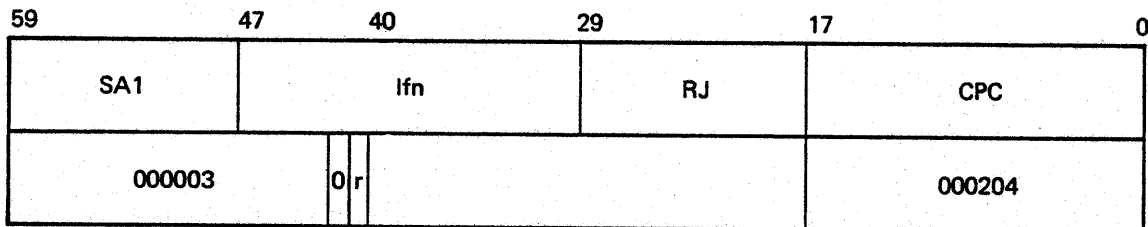
The diagram of the PRU and control word appears with the discussion of READC. PRU size must be standard 64 CM words for mass storage; if not, serious errors will result. The 24 low-order bits of the control word and the full CM words are written to the device.

BC is the count in 12-bit bytes of good data in this PRU and must be a multiple of 5. If BC/5 is less than device PRU size, then the next 12-bit byte after the good data is the SCOPE level number in binary. L must be in the range $0 \leq L \leq 17$ 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 SCOPE logical record format, data resolution is to the nearest full CM word so that the UBC field will always be 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; the level 17 octal must appear as a zero-length logical record.

The WRITEC macro generates the following code:



STACK PROCESSOR

The Stack Processor consists of the CM resident manager CP.SPM, the PP program 1SP and its various overlays. Basically, the components of the Stack Processor and their functions are:

CP.SPM	The Stack Processor Manager
OV.1S5	Examines DST ordinal and loads 3DO if DST ordinal = 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 777777B and O.BPRU.

SPM is called to enter, terminate and reissue stack requests. It performs all RMS I/O functions except file assignment (3DO), non-I/O skipping (4DO), and physical I/O (1SP). Request scheduling and device optimization is performed by SPM. 1SP is a driver and handles one request at a time. A PP or CP system routine initiates I/O by placing a stack request (first two words of 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, the possibility exists that priority stack requests will be pushed down in the DST chain.

RMS I/O is performed by SPM selecting a stack request and assigning it to a 1SP driver that is assigned to a DST ordinal which represents an access to the specified RMS device. If, at the time a stack request is received, no 1SP is up, the request will be entered by SPM and an M.ISP (initiate stack processor driver) request will be sent to MTR. If the MTR communication path is busy, no request is posted and 1SP will be 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 up the proper PP routine.

When 1SP comes up, it initializes the driver by concatenating the driver code letter (from the PPIR) with the characters 3S to form the driver name, 3Sx, and loading that PP overlay into the driver area in the PP. 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 EDITLIBs.) 1SP 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 it to 1SP at this time. Otherwise, it assigns the 1SP to idle. A subsequent stack request for this driver need only be placed in this 1SP's PP communication area to start I/O.

The code for each driver is contained in a common deck in the program library file. The common deck and driver overlay names are listed below:

Common Deck	1SP Overlay	Device Type
RMSA	3SP	6603-I disk
RMSB	3SQ	6638 disk

Common Deck	1SP Overlay	Device Type
RMSP	3SS	854 disk pack
RMSC	3ST	6603-II disk
RMSL	3SV	821 data file
RMSM	3SW	841 multiple disk drive
RMSY	3SY	844 disk pack

In addition to the overlay area for the driver, 1SP has an overlay area for the executive routine that performs the other side of the data transfer or a non-data 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.

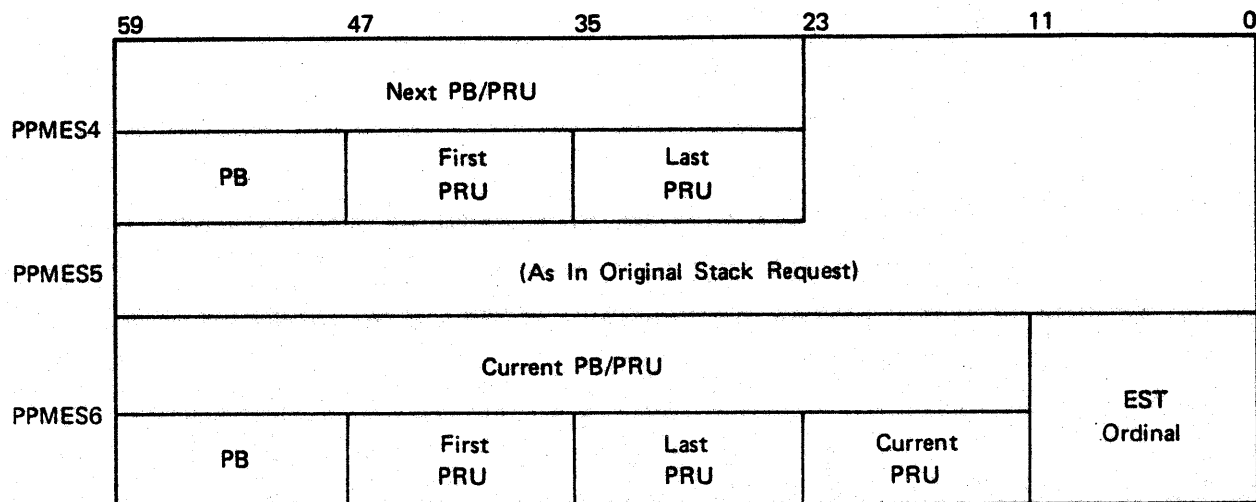
MTR insures that at least one PP contains a stack processor at all times, or that one is reserved for that purpose. This requirement is necessary so that it is always possible to load a PP overlay that resides on mass storage. To avoid unnecessary loading and dropping of stack processors, a stack processor will remain in the PP until MTR needs a free PP. At that point MTR will set a PP request flag for SPM. Idle stack processors check this flag in their idle loop and call SPM if it is set. SPM will then issue 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 go 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 those 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/1SP communication through the 1SP communication area. It is different from the external format because 1SP does not work with record blocks (RBs) but with physical blocks (PBs). SPM converts RBs to PBs for 1SP so that 1SP will not have to access the RBT chains.

The external stack request formats consist 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 part II, section 1.

The internal stack request format consists of three words and is part of the SPM/1SP interface at communication area locations PPMES4-6. The first and third words contain the next and current PB/PRU transfer information in their leftmost three and four bytes, respectively. The second word and rightmost two bytes of the first word are as in the original stack request. The next PB/PRU information in the first word is updated by the NXTPB function of SPM in an overlap asynchronous fashion so that 1SP can continue to do I/O and have it available when needed. At that point, 1SP uses it to update the current PB/PRU in word three which drives 1SP's current operation and the process is then repeated. This is done to facilitate device streaming. The internal stack request format is summarized in the following diagram.



The first and last PRU fields specify the range of the request within the specified PB and the current PRU is the one currently being processed. NEXT PB/PRU means the next one to be processed by ISP and in the case of skipping backward is the logically preceding one in the file.

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 and standard system symbols are given below:

Central Memory Read/Write Orders

- 00 O.READ Corresponds to READ macro, CIO codes 010 and 012. Read data from device to CM until (a) end of information is reached, (b) a short PRU is read, or (c) next PRU does not fit into the buffer.†
- 01 O.RDSK Corresponds to READSKP macro, CIO codes 020 and 022. Read as for O.READ until (a) or (b) above, or (c) the CM buffer is completely full; then change to O.SKF with N = 1 unless reading was stopped by (b) with record level \geq request level.†
- 02 O.RCMR No corresponding CPC macro or CIO code. Read as for O.READ but do not transmit the first n CM words of the PRU, where 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.
- 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.

† See logical sequence of events charts appearing earlier in this section.

- 03 O.RDNS Corresponds to the READNS macro, CIO codes 250 and 252. Read data from device into CM buffer until (a) end of information is reached, (b) a short PRU with record level 16 or 17 has been read, or (c) 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.
- 24 O.WCTNU
O.WCTU Corresponds to WRITEC macro, CIO codes 204 and 206. Provides non-stop writing from CM to device without releasing/reloading PP between records. User's buffer must contain at least two records. Writing stops when buffer is empty.
- 20 O.RCTNU
O.RCTU Corresponds to READC macro, CIO codes 200 and 202. Provides non-stop 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.
- 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.[†]
- 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, corresponds to WRITEF system macro, CIO codes 034 and 036. Same as above, but short PRU is followed by zero-length level 17 record (logical end of file mark).[†]

Peripheral Processor Read/Write Orders

- 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 Orders

- 12 O.SKF Corresponds to SKIPF macro, CIO codes 240 and 242. Skip forward until N short PRUs with level \geq the level specified in the request have been read, or end of information is reached. With N = 777777, the file is positioned at end of information.[†]
- 13 O.SKB Corresponds to SKIPB macro, CIO codes 640 and 642. Skip backward one or more PRUs until N short PRUs with level \geq 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).[†]
- 16 O.BPRU Corresponds to the BKSPRU macro, CIO codes 044 and 046. Skip backward N PRUs. This repositioning is by physical record units rather than logical records.[†]

[†]See logical sequence of events charts appearing earlier in this section.

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/ISP Communication Orders (Internal Format Only)

35 O.IDLE SPM has determined that there is no work for this ISP and that MTR is not requesting a PP. ISP checks the order code in its idle loop, waiting for SPM to assign a stack request. When another order code appears, ISP will process it.

36 O.DROP SPM has determined that this ISP is either idle or going idle and that MTR is requesting a PP. ISP will drop out.

37 O.SEEK SPM has stack requests for this ISP but none are on-cylinder. ISP will issue overlap seeks (up to 5), monitor the units and reserve the first one to come on cylinder. (Overlap seeks may 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.)

The following table is a summary of stack processor orders:

Octal Code	System Symbol	Order Function
00	O.READ	Read into central memory
01	O.RDSK	Read-skip into central memory
02	O.RCMPR	Read into central memory, drop first 3 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 PPU memory
11	O.RDPNP	Read into PPU, drop first 3 CM words
12	O.SKF	Skip forward
13	O.SKB	Skip backward
14	O.WRP	Write from PPU memory
15	O.WRPR	Write EOF/EOR from PPU memory
16	O.BPRU	Backspace n PRUs
17	O.RCHN	Evict
20	O.RCTNU	Read nonstop (comparable to tape READN)
	O.RCTU	
24	O.WCTNU	Write nonstop (comparable to tape WRITEN)
	O.WCTU	
35	O.IDLE	Wait for stack request
36	O.DROP	Drop PP
37	O.SEEK	Issue overlap seek

STACK PROCESSOR/SYSTEM INTERFACE

The system tables, system routines, and MTR functions used by the stack processor are described in this section.

TABLES

Control Point Areas: Control point error flag, storage move flag, RA, and FL. The stack processor accesses but never changes these fields.

DST: All fields of the DST entry whose ordinal is placed by MTR in the 1SP input register are used. SPM makes all DST changes except one made by MTR during 1SP assignment to a PP.

EST: Mass storage flag, unloaded flag, off flag, and DST ordinal are checked but not altered.

FET: Code and status field in first word, error processing flag in second word are accessed by SPM. IN and OUT pointers in third and fourth words are accessed by 1SP. Code and 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, code and status field in second word are accessed by SPM. The code/status field has been processed the same as for the FET.

RMSBUF: Stack processor (1SP) 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 all 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 re-issued, 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. 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 an 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.

SYSTEM ROUTINES/PROGRAMS

PP Programs

1SX Stack processor auxiliary program is called by MTR request for tasks that 1SP cannot handle or does not have time to do. For example, 1SP does not issue dayfile messages, because if the dayfile buffer is full, 1SP and MTR could loop endlessly waiting for each other.

- MTR** System Monitor calls 1SP initially (via 1S5) when a request has been made for an inactive DST entry and performs various functions for 1SP 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 xx has an outstanding request for an idled device.

PP Resident Routines:

- R.DCH** Releases a channel reservation.
- R.IDLE** Entered when 1SP releases its PP.
- R.MTR** Used for all MTR functions other than reserve or drop channel.
- R.TAFL** Terminates access to the control point field length. When necessary, it is used to interlock storage moves during execution of a request, and when a request is terminating (except at control point zero) to switch 1SP back to control point zero.
- 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 Functions:

- M.DPP** Releases PP assignment.
- M.SPM** Used by 1SP 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 is highly structured. This area 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 PRU/PB count in the right-most 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 there is to be done.

The format of the 1SP communication area is:

	59	47	35	29	23	21	17	11	0
PPIR	1 S	P O	Driver Name	DST Ordinal	E	O	CH		
PPOR	M.SPM							EX.xxx	
	M.xxx								
PPMES1*	Unit Ordinal	Unit Ordinal	Unit Ordinal	O			-		
	1SX Error Information								
PPMES2*	PB _x	PB _y	PB _z	-			-		
	1SX Error Information								
PPMES3	SPM Internal Information							PRU/PB	
PPMES4	Internal						SR Order		
PPMES5	Format								
PPMES6	Stack Request								

*EXAMPLE SHOWS THREE UNITS SPECIFIED FOR OVERLAP SEEK WITH ZERO TERMINATOR

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. With IP.DEBUG=0, these conditions are treated similarly to other errors: an error code is placed in the code and status field of FET and FST entries and the control point is aborted if error processing (EP) bit in FET is zero. With IP.DEBUG ≠ 0, an invalid MTR function is issued with the 1SP output register having 77 in byte 0 and an error code in byte 1.

END OF INFORMATION

Puts 01 into bits 9-13 of code/status, but does not issue a message or abort control point. (Non-fatal 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

attempts (3 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 9-13 of code/status, and the control point is aborted if 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 nonzero, request execution terminates with the bad PRU being the last one read.

BUFFER PARAMETER ERROR

This error occurs when a request is processed that references an FET when not all the following conditions are satisfied:

$0 \leq \text{FIRST} < \text{LIMIT} \leq \text{field length}$
 $\text{FIRST} \leq \text{IN} < \text{LIMIT}$
 $\text{FIRST} \leq \text{OUT} < \text{LIMIT}$

Calls 1SX with code 11B (dayfile message BUFFER ARGUMENT ERROR), puts 22B into bits 13-9 of code/status, and aborts control point if EP bit is zero.

NOT ASSEMBLED FOR ECS

This error occurs when a request references a DST entry for an ECS device. 1SP issues a bad MTR request (code 77B).

UNDEFINED ORDER CODE

A request contains order code 07. Calls 1SX with code 22B (dayfile message INVALID STACK ENTRY), puts 22B into bits 13-9 of code/status and aborts control point if EP bit is zero.

NO FET FOR O.RMR

A request contains order code 06 (O.RMR), but no FET was specified. Calls 1SX with code 11B (dayfile message BUFFER ARGUMENT ERROR), puts 22B into bits 13-9 of code/status and aborts control point if EP bit is zero.

ADDRESS OUT OF FL FOR O.RMR

Address for table of disk addresses for O.RMR is out of field length. Call 1SX with code 22B (dayfile message INVALID STACK ENTRY), puts 22B into bits 13-9 of code/status and aborts control point if the EP bit is zero.

INTERLOCK BROKEN

A group of interlocked stack requests were interrupted by a malfunction of a controller and/or unit. 1SP puts 24B into bits 13-9 of code/status.

RMS HARDWARE ERROR

An RMS hardware error is reported when any of the following errors occur on a device:

Unit not ready	6603-II positioning failure
Positioner not ready	No status returned
6681 internal/external reject	Address byte not accepted
Unit busy too long	No disconnect on status request
Channel stays active after 'connect' or 'function'	Abnormal on seek
Unable to connect	Channel not active after ACN

These error conditions are reported by ISP (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, 22B is returned to bits 13-9 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 (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 5-1 shows the flow of control of disk I/O, including the processing of dismountable devices.

NORMAL CALLS FOR READ/WRITE

The user requests I/O by calling CIO in RA+1. CIO calls 3DO if the file is new to assign it to a specific pack as discussed below under SPM. If it is an existing assigned disk file, CIO loads 4ES which:

- 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 1SP.

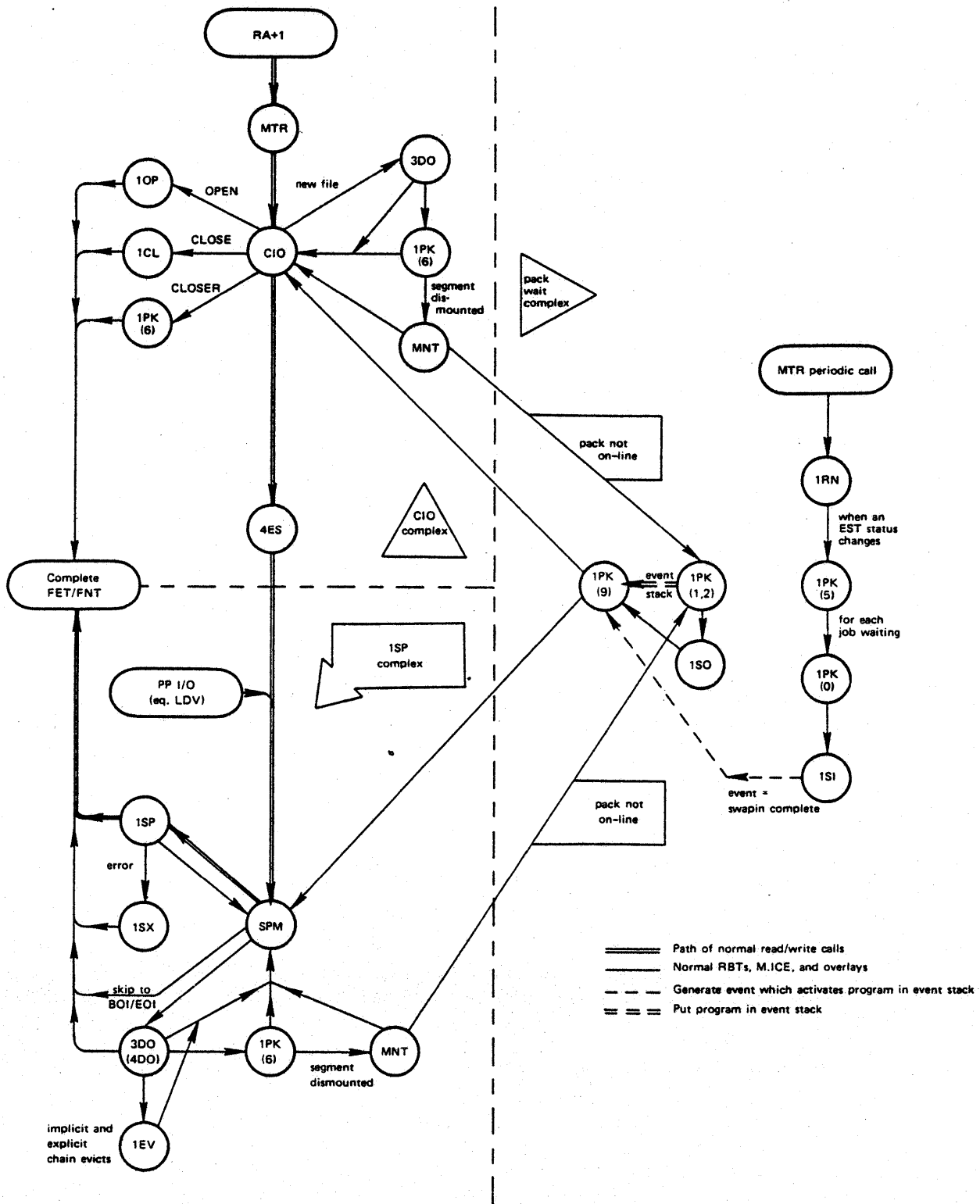


Figure 5-1. Device Set I/O Processing

1SP processes I/O to an arbitrary point, for example, to a cylinder boundary, and then may re-issue the request to SPM. 1SP always returns the request to SPM if the DAM ordinal in the RBT changes; thus the ordinal 777 in overflow wordpairs 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 above description: BKSPRU; release-chain not mounted; SKIPF or SKIPB with count=777777B; file is new and has no RBTs (hasn't been assigned); position is at an overflow wordpair (DAM=777B); etc. 3DO masquerades 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.

3DO DEVICE ASSIGNMENT

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=777777B are sent to 4DO for completion. If the current segment of the file is not on a mounted pack, or if 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 wordpair and changes the format from new-file to 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 was 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:

1. the device must belong to the required set
2. if *PF, *Q, or *SYS, the device must also have that attribute
3. if a device type or allocation style or VSN was 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 wordpair (for all but new files) and an empty wordpair for the selected device, and re-issues 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 CHOOSING AND MOUNTING MEMBERS

1PK may be called by CIO, MNT, ADS or independently via M.RPJ. 3DO may not 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 by M.RPJ. The calling routine places a function number 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. If not CLOSER, if the function is a write (not REWRITE) and if the input register byte 4 is zero indicating a write at EOI, space must be assigned. Assuming 3DO checked the mounted devices, read the SMT and assign the file to a pack with available space which is not mounted, considering first the online 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 re-issues the stack request. If the pack is not online, 1PK function 9 is put on the event stack with the stack request in its message buffer; see the following discussion on functions 5, 0, and 9.

If UP is on and there is no space, complete the FET with error 10 – device capacity exceeded. If the current position is not online on a public set, abort the job.

If not CLOSER and not a write, the first step is a mount. First, advance position past any overflow wordpair; check if required disk is mounted (if yes, re-initiate the function by calling CIO, or SPM if CIO not in input register); if not CIO, call MNT which will call 1PK back and the process will repeat if MNT found the pack online 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 will be called by 1RN which periodically checks ON/OFF changes. Function 5 will cause the removal of all 1PKs from the event stack which were waiting and they re-issue the stack request; MNT finds the pack online 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 wordpair is attached on the end of the RBT. The DAM and VSN fields will be zero. Both EOI and EOVS status are set in the FET.

If the current position is not EOI, the file is positioned to the next EOVS wordpair, or to EOI if that comes first. If the current position is an EOVS wordpair, nothing is done. EOVS status is returned to the FET if the new position is an EOVS wordpair, or EOI if 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 re-issues the stack request. This continues until the pack goes offline or is remounted.

functions 1, 2

These functions are called by MNT when a required pack is not online 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 that DDT. 1PK function 9 is put in the event stack on the JDT swapin, and the job is swapped out by the macro CISO. The initial PP input register from the caller is sent on in 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 online 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, 9

Function 8 swaps the job in. Function 9 re-initiates the I/O function. If the input register was CIO, function 9 will re-issue it to CIQ. If not CIO, the stack request is in the message buffer and is re-issued 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; operator dismount, 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 may 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; then 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 end of job) decrements the set's activity and resets it to dismounted status if no jobs reference it. The operator typein 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 may mount the disk. Only an RMNT typein or a RECOVER clears this flag.

An operator dismount proceeds as follows:

If the pack is mounted on this mainframe—

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 (signalled by FB being set to 11); then it clears request-idle
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; the DDT is written in dismounted format (S.DDSN=1)
4. If the device is shared, the RBR is simply 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 3 above.

If the pack is not mounted but is online on this mainframe, DSM simply 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 RMNT (remount). 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 online). The operator may 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 central memory 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 central memory or through a distributive data path (DDP). 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 card or REQUEST macro. On the control card, the user includes an EC parameter in addition to the normal parameters in one of the following forms:

EC	For a default (IP.BUF) size buffer
EC(XXXX) EC(XXXXK)	For a buffer of XXXX-thousand (octal) words
EC(XXXXP)	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 into bits 0-11 and the display code character K or P must be set into bits 12-17.

In a write sequence, the user first puts data into a central memory 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.

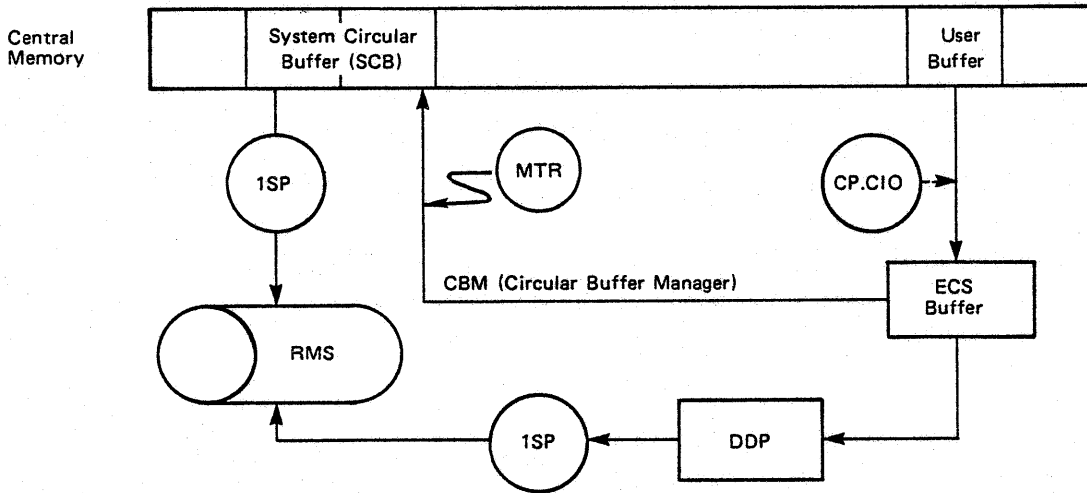
CP.MTR recognizes the CIO call and passes it to CP.CIO for processing. ECS-buffered file I/O causes CP.CIO to perform a validity check on the FET and activates the proper ECS driver. The data is then written directly from the user's central memory buffer to the user's buffer in ECS.

The above 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 1SP loads the appropriate ECS executive routines into an area of 1SP memory.

At a request of 1SP for 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 central memory buffer path is selected, 1SP 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, 1SP 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 1SP from missing disk revolutions during an I/O buffer transfer.

When the distributive data path is selected, the processing is very similar. CP routines do the same bookkeeping as in the central memory case, but only point to the data in ECS instead of transferring it to central memory. As a result, much less central memory is used for the transfer: N words instead of 65xN words for an SCB containing N PRUs.

The following illustrates the general flow of output to an ECS buffered RMS file. Either data path may be assigned dynamically, depending on availability.



A permanent file is a mass storage file cataloged so that its location and identification are always known to the system. The installation designates rotating mass storage devices to contain files made permanent by the user. When creating a private device set, a user can designate permanent file devices. Any file, regardless of content, which is not already permanent and which resides on a permanent file device, may be made permanent by explicit request. Files on magnetic tape or ECS may not be made permanent.

In a multi-mainframe environment, permanent files may be transferred between linked mainframes. The MMF station is used to stage files between mainframes.

PERMANENT FILE FUNCTIONS

The following permanent file functions are available as system macros:

CATALOG	an existing local mass-storage file, thereby making it a permanent file.
SAVEPF	stage out a local copy of the file and catalog the file.
ATTACH	a previously cataloged file to a control point.
GETPF	stage in a local copy of the permanent file.
RENAME	a file in the permanent file directory.
EXTEND	a currently attached file by making permanent an extension to it.
PURGE	a file from the permanent file directory.
ALTER	allows the logical end-of-file to be set to the current file position.
PERM	allows a running program to determine what permissions have been granted to an attached file.
FDB	generates a file description block required for interface with the permanent file system.

In addition, two SCOPE system macros and control cards may be used on an attached permanent file to logically detach it prior to job completion.

macro:	CLOSE Ifn, UNLOAD or CLOSE Ifn, RETURN
control card:	RETURN (Ifn) or UNLOAD (Ifn)

MACRO REQUESTS

The permanent file functions CATALOG, SAVEPF, ATTACH, GETPF, RENAME, EXTEND, ALTER, and PURGE are available either as control cards or running program macro calls. The same parameters are used for both; they differ in the call format and in the capability to test status in the running program.

All permanent file macro requests share a common format:

name	fdbaddr.RC.RT.NR
<i>name</i>	Macro name written in a COMPASS instruction mnemonic field
<i>fdbaddr</i>	Address of fifth word in a file definition block (FDB)
<i>RC.RT.NR</i>	Optional parameters
<i>RC</i>	Returns a code to the FDB and gives control to the requestor on non-fatal errors.
<i>RT</i>	Returns a code to the FDB and inhibits permanent file queuing. (Queueing occurs when a file cannot be attached immediately.)
<i>NR</i>	Specifies no recall. (All permanent file macro calls are issued in recall unless NR is specified.)

Error messages will be written to the job dayfile, unless the RT or RC parameter is specified.

PARAMETERS

The parameters described below are common to both control cards and macro functions. With the exception of lfn and pfn, parameters may be given in any order. Each is written in the form: cc = value or password where cc is a two-letter parameter code. The lfn and pfn parameters are position dependent; if one or the other is omitted, the lfn and pfn are considered to be the same string of characters. In the case when a pfn contains more than 7 characters in the parameter, the lfn will be the first 7 characters given in the pfn.

<i>lfn</i>	Logical file name; 1-7 character alphanumeric name (first character alphabetic) by which a file is known and referenced at a control point. Once a permanent file is attached to a control point, it is referenced by this name.
<i>pfn</i>	Permanent file name; 1-40 alphanumeric characters, assigned by file creator, under which a file is cataloged.
<i>RP</i>	Retention period in days, (0-999) specified by creator; indefinite retention indicated by 999. Installation default value is defined by the installation.

AC Account parameter; 1-9 letters or numbers.

CY Cycle number (1-999) assigned by creator. Default value on initial CATALOG is 1, and on ATTACH, the highest number cataloged.

LC Lowest cycle number cataloged is referenced when value given is non-zero; default is highest number cataloged.

PW List of passwords used to establish user's access permission. Written as:

PW = psw₁, psw₂, psw₃, , psw_n

PW also is used in a CATALOG request when a new cycle is added or PUBLIC file created, and in a PURGE request in permanent file name mode; up to five passwords allowed, each 1-9 letters or numbers.

TK	Turnkey	} Password definition 1-9 letters and numbers
CN	Control	
MD	Modify	
EX	Extend	
RD	Read	
XR	Common	

MR If non-zero, gives read permission only which will permit read access of the file by other users.

RW If non-zero, multi-read with single rewrite will be allowed. If zero, and either CN, MD or EX permission is requested, exclusive access will be given.

ID Identifies file creator; 1-9 letters and numbers. The ID name PUBLIC is reserved for PUBLIC files, and SYSTEM is reserved for SYSTEM files.

ST Staging ID; three letters and numbers that specify logical ID, Host ID or link ID. The staging ID identifies the mainframe where the file resides. When both ST and SN appear, ST takes precedence and SN is ignored.

FO File structure (ordering of data) is checked so that extend and modify permissions will have meaning for direct access (DA), actual key (AK), or indexed sequential (IS) files. Applies to Record Manager files only.

RB If nonzero and S.PCRB in PFC entry is set, PURGE will not allow the mass storage associated with the file to be released. RB is used when the routine RECOVER has encountered RBT chain conflicts with other files.

<i>EC</i>	Allocates ECS for permanent file I/O buffer.
<i>EC=K</i>	Allocate a standard number of 1K blocks.
<i>EC=nnnn</i>	Allocate an octal number of 1K blocks.
<i>EC=nnnnK</i>	Same as above. If K is omitted and P is not given, K is assumed.
<i>EC=nnnnP</i>	Allocate an octal number of ECS pages.
<i>SV</i>	Device set name 1-7 letters and digits, first character a letter. If specified on an ATTACH or PURGE, the specified permanent file resides on the device set.

STAGING ID

The staging ID, ST, identifies the mainframe where the file resides. It may be either a linked mainframe or the local mainframe that is executing the calling routine. The host ID (local mainframe), the link ID (linked mainframe), or a logical ID may be specified for ST.

When the host or the link ID is specified and the station is running, the local or the linked mainframe is given the task of satisfying the file request. When a logical ID is specified, the task is assigned to the mainframe having that ID in its logical ID table.

Example:

Three system files, FILEA, FILEB, and FILEC normally reside at mainframe A, MFA, but occasionally will be moved to mainframe B, MFB.

A user who knows the physical location of the files can set the ST parameter (on a GETPF, SAVEPF, or PURGE) to the proper value (example, ST=MFB) before submitting a job.

By using a logical ID in the ST parameter, however, the routine could remain the same no matter where the file reside. The installation would associate these three files with a logical ID, such as ABC. The caller then would specify ST=ABC and mainframe having ABC in its logical ID table would be assigned to the task.

UNIVERSAL PERMISSION

An installation may define a combination of one or more permissions to be granted automatically by activating the universal permission option (IP.UP). A nine-character password is defined by the installation for such a permission combination. When this password is given on an ATTACH request, the permissions are granted.

PUBLIC FILES

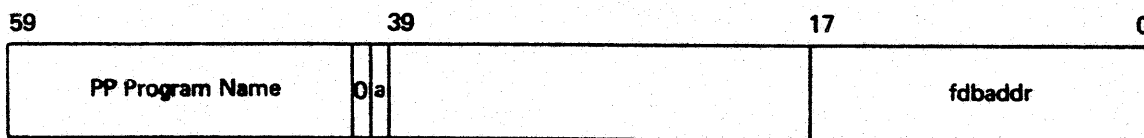
If the public permission password is correctly specified, a file may be cataloged under the ID of PUBLIC; then the user can omit the ID parameter on all permanent file requests. Attaching a PUBLIC file does not preclude the necessity of using correct permission codes.

As all cycles of a file share the same ID, when the first cycle of a file is cataloged as PUBLIC, all subsequent cycles will become PUBLIC. With the RENAME function, a PUBLIC file can be given a new owner ID, making it a private file.

MACRO REQUEST CALLS

Each permanent file request macro expansion (except FDB) will generate an RA + 1 call to the permanent file PP program. All permanent file macro requests are issued with recall status set (bit 40 in word 2 of the request) unless NR (no recall) parameter is present in the macro call.

An RA + 1 call to a permanent file PP program has the following format:



a = 0 If the NR parameter is present in the macro call.

a = 1 If the NR parameter is not present in the macro call.

FILE DEFINITION BLOCK

Parameters necessary for the execution of a permanent file function are contained in a central memory table called the file definition block (FDB). Error codes are returned to the user via the FDB. The FDB is generated automatically for control card calls, but it must be generated by an FDB macro for use by other permanent file macro calls. The format of the FDB is shown in part II, section 2.

The macro for generating an FDB has the following format:

fdbaddr	FDB lfn,pfn,parameters
<i>fdbaddr</i>	is the symbol associated with word 4 of the FDB; it must be present in the location field.
<i>parameters</i>	are separated by commas, and the list is terminated by a blank. Parameters include any of the 2-letter parameter codes listed above. Parameters are entered into the FDB as they are encountered in the list.

The FDB is generated in-line during assembly whenever the macro is called.

If the RC and RT parameter is specified in a macro call, a return code will be available to the user in word 4, fdbaddr, bits 9-17.

Word 4, bits 0-8, contain the PFM request code stored in the following format:

Bit	Content (when set)
8	NR option given
7	RT option given (implies RC given also)
6	RC option given when bit is zero; no RC or RT option when bit is one
5-2	Function code
	0010 ATTACH, GETPF
	0100 CATALOG, SAVEPF
	0110 EXTEND
	0111 ALTER
	1000 PURGE, PURGE(ST=xxx)
	1010 RENAME
	1100 PERM
1	Not used
0	Function completed when set to one

Request code examples (octal):

020	Catalog; return error code and control to user on non-fatal error.
130	Extend file; abort job on any error.
210	Attach file; return error code on any error and inhibit file queuing.

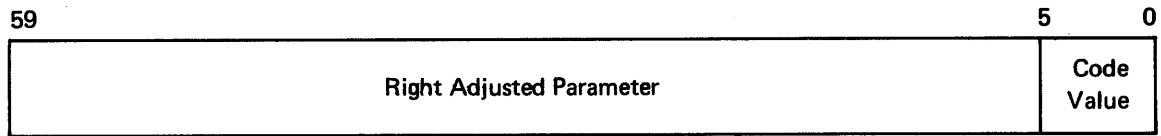
Word 4, bits 9-17, contain a 9-bit return code which can assume the following octal values; the associated message is written to the job dayfile.

000	Function successful
001	ID error
002	lfn already in use
003	Unknown lfn
004	No room for extra cycle (limit is five)
005	Permanent file catalog (PFC) full
006	No lfn or pfn
007	(not used)
010	Latest index not written for a random file
011	File not on PF device
012	File not in system
013	Archive retrieval aborted
014	Bad LPF communication
015	Cycle number limit reached. Maximum value of cycle number is 999
016	Permanent file directory (PFD) full
017	Function attempted on non-permanent file
020	Function attempted on non-local file
021	(Not used)
022	File never assigned to a device
023	Cycle incomplete or dumped
024	File already attached
025	File unavailable
026	(Not used)
027	Illegal lfn
030	File dumped
031-032	(Not used)
033	ALTER needs exclusive access
034	(Not used)
035	File already in system
036	(Not used)
037	(Not used)
041	Device set not mounted at control point
042	RBT chain too large for PFC
043-067	(Not used)
070	PFM stopped by system
071	Incorrect permission
072	FDB address invalid
073	I/O error on PFD/PFC read/write
074	ST parameter illegal with private device set (MMF-PURGE)

On control card requests, all errors are fatal; on macro requests, unless an RC or RT parameter is specified, all errors are fatal. If RC or RT is specified on a macro call, an error code will result in control being returned to the user. Any job that attempts a privacy breach is terminated.

On control card or macro requests for GETPF, SAVEPF, or PURGE(ST=xxx), all errors are fatal. The job is terminated and an error message is written to the job dayfile.

The parameter words of the FDB have the following format:



As shown above, parameters are stored, one per word in any order. Parameter code values are placed in bits 0-5. The format of values entered in the parameter field is indicated in parentheses in the following list.

00		End of FDB list
02	RP	Retention period in days (binary)
03	CY	Cycle number (binary)
04	TK	Turnkey password (display code)
05	CN	Control password (display code)
06	MD	Modify password (display code)
07	EX	Extend password (display code)
10	RD	Read password (display code)
11	MR	Multi-read access only (binary)
13	XR	Control, modify, and extend password definition (display code)
14	ID	User identification (display code)
16	AC	Account parameter (display code)
17	EC	Request ECS buffering of I/O (display code)
20-24	PW	Submitted passwords (display code)
25	FO	File organization (display code)
27		Used internally for checkpoint/restart
31	LC	Lowest cycle (binary)
32	ST	Staging ID
33	RW	Multi-read with single rewrite
40	SN	Setname (left justified, display code)
41		Reserved for VSN parameter
43	RB	PURGE RB conflict parameter

The CATALOG macro references FDBA which contains the necessary parameters to make LFI permanent.

Since RC is specified on the CATALOG macro, control will be returned to the user on a non-fatal error; and a return code will be made available in location FDBA (bits 9-17). If RC is not specified, all errors result in termination and a diagnostic message.

New Cycle Catalog Example:

	CATALOG	FDB5
	.	
	.	
FDB5	FDB	LF16,MFILE,CY-12,PW-Y,Z,ID-ABC

This job will add a second cycle to permanent file MFILE, created in the preceding example. File LF16 is assumed to be a valid local file on a permanent file device. The PW parameter is used to submit the passwords needed to obtain control permission. Had the initial catalog attempt aborted, MFILE would not exist; and this new cycle attempt would be processed as an initial cataloging. If successful as an initial catalog, the file would be unprotected as no passwords are defined in the FDB. An alternate form of the FDB could be used:

FDB5	FDB	LF16,MFILE,CY-12,PW-Y,Z,ID-ABC,TK-Y,CN-Z,MD-X
------	-----	---

The above FDB would perform equally as well for a new cycle catalog because the TK, CN and MD parameters would be ignored. If initial cataloging had failed, this FDB would catalog the file with protection and the PW parameter list would be ignored.

SAVEPF FUNCTION (multi-mainframe only)

SAVEPF	fdbaddr,RC,RT,NR
--------	------------------

The SAVEPF request requires the lfn, pfn, ID and ST parameters in the FDB. The optional parameters are the same as those for the CATALOG function. Refer to the CATALOG function for more details.

SAVEPF stages a local file specified by the lfn to be cataloged at the mainframe associated with the ST parameter. The mainframe where the file is to be cataloged may be either a linked mainframe or the local mainframe. In both cases, two files exist after the call: a new or updated permanent file and the local file attached to the calling routine.

ATTACH FUNCTION

ATTACH **fdbaddr,RC,RT,NR**

The ATTACH request requires the lfn, pfn and ID parameters in the FDB. The following parameters are optional:

CY	Cycle number to be attached.
PW	Password list
MR	Multi-read access
LC	Lowest cycle number
RW	Multi-read/rewrite access
EC	ECS buffering for I/O
SN	Setname

RC parameter is the same as for CATALOG. When RT is specified, a code of 25 is returned to the FDB if the file is currently in use. If the RT parameter is not specified, the following circumstances will cause a job issuing an attach request to be queued for the desired file:

File not available for exclusive access by requesting job

Attached permanent file (APF)table full

Archived file temporarily unavailable. The ATTACH request will cause a LOADPF job to be set up and scheduled through the tape scheduler. The job requesting the ATTACH will be swapped out until the file is available. Permanent file utility is running.

If the CY parameter is zero or not present and the permanent file has multiple cycles, the default cycle attached is the one with the largest cycle number, presumably the latest cataloged. If the CY parameter is present and that particular cycle number is not known to the system, the request cannot be honored. If both LC and CY are specified, LC is ignored and the conflict is resolved.

System evaluation of passwords establishes the type of access granted to the user for each file. Subsequent to ATTACH, the user cannot access the file in any way for which he does not have permission. For example, if ATTACH results in only READ permission, the user cannot subsequently attempt to use MODIFY or EXTEND.

ATTACH does not preclude opening the file. The success of an OPEN request depends upon the permission granted when the file is attached. If the file is attached to another control point and multi-read access is not possible, PFM will wait for access to the file.

Attach Example:

```
FDBZ                    FDB                    LF,MFILE,MR=1,PW=Y,ID=ABC,CY=1
.
.
ATTACH                    FDBZ,RC,RT
```

The permanent file MFILE is referenced again. Explicitly stating CY=1 ensures that cycle one will be attached.

In the FDB, only the password for turnkey appears; EXTEND and READ permissions will be granted by default. In this example, the macro contains MR=1; therefore all permissions except READ are ignored, making the file available for multi-read access.

In the macro request, the presence of RC and RT parameters would result in the octal code 25 being returned at location FDBZ if the file is unavailable.

```

          CATALOG          FDB1
          CLOSE            LF1,UNLOAD,RECALL
          .
          .
          .
          ATTACH           FDB1
          .
          .
          .
FDB1     FDB              LF1,PERMF,TK-T,MD-M,EX-E,CN-C,PW-T,ID-ABC

```

The above example illustrates several points. Assuming that local file LF1 has been created, it is cataloged as cycle 1 (by default) of permanent file PERMF. The file is protected by turnkey, control, modify and extend passwords. The PW parameter in the FDB is ignored in cataloging.

After cataloging is complete, a CLOSE/UNLOAD logically detaches the file from the job.

As illustrated, the file PERMF now can be re-attached. Although not mandatory, the same FDB is used to conserve CM space. When PERMF is attached, the default cycle number is the largest cataloged; therefore cycle 1 is the only cycle present. The PW parameter contains the turnkey password giving READ access permission by default. This example illustrates an implicit read-only attach.

The same example is shown with two FDBs:

```

FDB1     FDB              LF1,PERMF,TK-T,MD-M,EX-E,CN-C,ID-ABC
FDB2     FDB              LF1,PERMF,PW-T,ID-ABC
          .
          .
          .
          CATALOG          FDB1
          CLOSE            LF1,UNLOAD,RECALL
          .
          .
          .
          ATTACH           FDB2

```

GETPF FUNCTION (multi-mainframe only)

GETPF fdbaddr,RC,RT,NR

The GETPF request requires the lfn, pfn, ID, and ST parameters in the FDB. The optional parameters are the same as those for the ATTACH function except for RW, which is ignored if present. See the ATTACH function for more detail.

GETPF prepares for staging the requested permanent file from the mainframe specified by the ST parameter and attaching a local copy of that file to the calling routine. The file must be opened before actual staging occurs. The mainframe specified by the ST parameter may be either the linked mainframe or the local mainframe. In both cases, two files exist after the file is staged: the permanent file and a local copy of the file attached to the calling routine.

The local copy of the file is processed, even if the permanent file specified resides at the same mainframe.

ALTER FUNCTION

ALTER fdbaddr,RC,RT,NR

The ALTER function causes the current position of an attached permanent file (designated by lfn in the FDB) to be recorded as end-of-information in the RBTC of that file. Permissions needed to perform the ALTER function depend upon the context in which function is issued. If the current position of the file is less than the file EOI (as attached), modify permission, extend permission and exclusive access are required. If the current position is greater than the file EOI, ALTER operates similarly to the EXTEND function and extend permission is required.

RENAME FUNCTION

RENAME fdbaddr,RC,RT

Any or all information cataloged by the user can be replaced through the RENAME function. The file must be attached to the requesting job with all permissions granted. A file owner can change permanent file name, cycle numbers, passwords, and even the user ID.

In the FDB for this request, lfn is the only required parameter. The specified parameters will cause replacement of existing parameter information if they contradict the cataloged information. If they duplicate the cataloged information, the parameters are ignored.

Parameters which could result in replacement:

pfn	Permanent file name	EX	Extend password
ID	Owner identification	MD	Modify password
RP	Retention period	CN	Control password
CY	Cycle number	AC	Account name
TK	Turnkey password	XR	Common password definition
RD	Read password		

The PW parameter may be specified to submit the public password if the file ID is to be renamed PUBLIC. Other parameters in the FDB will be ignored. Changing the ID, pfn, or passwords for any cycle cataloged will change all cycles. No ID, pfn, or CY changes are permitted if any of the cycles have been dumped (mode 2 dump) or archived as retrieval of such files would be impossible. RC and RT are as for CATALOG.

An attempt to rename PFN/ID when the new PFN/ID pair currently exist will be ignored; however, the remainder of the specified changes will still occur.

Rename Example:

	ATTACH	FDBA
	RENAME	FDBB
	.	
	.	
FDBA	FDB	DFILE,MFILE,PW-Z,Y,X,ID-ABC
FDBB	FDB	DFILE,PFILE2,RD-W,MD-,CN-ZZ

Assuming that MFILE was cataloged with X, Y and Z as passwords for modify, turnkey and control, read access would be given by default when DFILE is attached as a local file. By RENAME action, the cataloged permanent file name will be replaced with PFILE2. A new password, ZZ, will replace the existing password for control permission; a read password, W, will be cataloged for the non-existent read password. The password for modify permission will be removed, and none will replace it. The owner's ID remains unchanged. Since no cycle number was given in FDBA, the cycle with the largest number will be attached; renaming will not change the existing cycle number, as no replacement is given in FDBB.

FDB1	FDB	LFILE,MFILE,CY-9,RD-Y,ID-ABC
FDB2	FDB	LFILE,MFILE,CY-8,PW-X,Y,Z,ID-ABC
FDB3	FDB	LFILE,MFILE,RD-Z
	.	
	.	
	ATTACH	FDB2
	RENAME	FDB1
	.	
	.	
	RENAME	FDB3

This example illustrates that for renaming purposes, the same file can be called more than once in a job. If the read password was originally cataloged as X, it is changed to Y when the file is renamed as cycle 9, and then finally changed to Z. The appearance of an identical ID parameter in FDB1 will be ignored.

EXTEND FUNCTION

EXTEND fdbaddr,RC,RT,NR

Local extensions can be written at the end-of-information point of an attached permanent file, and an extend function issued, thus extending the length of the permanent file. The file must be attached with EXTEND permission granted.

In the FDB, the required parameter is lfn; the extend password, if defined, must appear in the PW list. The extended section of the file will acquire the privacy controls of the permanent file.

If lfn is an indexed file, the current index will be assumed to be the only valid index for the entire file. Random files must be closed before an EXTEND request is made.

Extend Example:

```

ATTACH          FDBX
.
.
.
EXTEND          FDBX
.
.
.
FDBX            FDB            LF1,PROGLIB1,ID=XYZ

```

The program that attaches permanent file PROGLIB1 as the local file LF1 writes beyond the end of information. Assuming that no password was required for extend permission, it would be given by default. Prior to program termination, the EXTEND request would make permanent any additions written to the file.

PURGE FUNCTION

```

PURGE          fdbaddr,RC,RT,NR

```

A cycle of a file can be removed from the catalog of permanent files by the PURGE function. In the macro, fdbaddr is required; RC and RT are as for CATALOG. In the FDB, the lfn is the only required parameter if the file was attached before the purge function was issued; all other parameters are ignored. The optional parameters are: CY, SN, LC, EC, RB, and PW. Control permission must be granted, or the job will be terminated. For the macro request, the FDB referenced may be one used at attach time; or it may be a new FDB. Only one cycle of a file may be purged at a time. When the last cycle of the file is purged, the entire permanent file name entry is removed from the directory and catalog.

A user attempting to purge a permanent file already attached must specify the lfn under which the file was attached. This purge is by local file name, and the user need provide only the lfn in the FDB.

To purge a file not already attached, the user must specify a local file name not in use at his control point. The permanent file name, ID and password list must be given. If the file resides on a private device set, the SN parameter must be given.

To PURGE a permanent file at a linked mainframe, the user must specify the ST parameter. If both the ST and SN parameters are specified, the VSN parameter is ignored and the PURGE takes place at the mainframe specified (either the linked mainframe or the local mainframe). If RB=1 is specified in the FDB and the RB conflict flag has been set in the PFC by RECOVER, the RBs in the chain will be zeroed and thus the storage will not be released to the system. The FNT permissions will be reduced to control only to prevent further use of the file.

PERM FUNCTION

The PERM function is available only as a system macro. A running program can determine if a file is a non-permanent local file or what permissions have been granted to a currently attached permanent file.

PERM **fdbaddr,RC**

The lfn of an attached permanent file should be given in the FDB. This macro produces a 5-bit code in the fdbaddr return code field. The bits represent the following information:

Bits 0-3

1000	CONTROL permission
0100	MODIFY permission
0010	EXTEND permission
0001	READ permission

Bit 4

1	non-permanent file
0	permanent file

A return code of zero signifies that the lfn was not in the FNT for the control point (a non-existent file) or that some other error was detected.

Perm Example:

```
FDBA                      FDB                      DFLN, PFILE, CY=1, PW=XXX, ID=ABDC
.
.
.
ATTACH                      FDBA, RC
.
.
.
PERM                      FDBA
```

Assuming the file had been cataloged with passwords required for control and modify permissions, the ATTACH request would have generated control permission by the password XXX and read and extend permissions by default. A subsequent PERM request would cause an octal 13 value to be returned to the FDB. The code indicates a permanent file is attached with read, extend and control permissions.

PERMANENT FILE UTILITY ROUTINES

The utility routines provide the capability for 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 following routines perform these functions:

DUMPF Copies all mass storage resident permanent files to tape. The purpose is either to clear all permanent files from mass storage devices or to provide periodic back-up files for an installation. May be used with system devices or user device sets.

LOADPF Counterpart of DUMPF; required to reload permanent files from tape to mass storage. Loads files onto system devices or user device sets.

TRANSPF Enables permanent files and/or related tables to be transferred to a public RMS device or to a private device set. Cannot move system files.

AUDIT Produces a formatted output file containing statistics on permanent files in the system. User device sets and system devices can be audited.

A permanent file named DUM, having an ID of PUBLIC, must be cataloged before a DUMPF or TRANSPF function may be called. Control cards calling these functions must include required passwords for DUM, for which an implicit attach will be performed. DUM must reside on the device set on which DUMPF or TRANSPF will be working. The DUM passwords are defined by the installation and can be changed as often as desired. Since these passwords are used to determine the mode and legality of the utilities, an installation can define (and redefine) the passwords to prevent unauthorized users from calling the utilities and thus maintain the integrity of the data bases.

Four passwords are meaningful to DUMPF, turnkey, read, modify, and control. The turnkey password is assembled into DUMPF and is specified only for the initial catalog of DUM. The other three passwords are specified for the DUMPF utility to determine the mode. If any of the read, modify, turnkey, and control passwords are defined, they must be specified on the TRANSPF call.

The DUMPF routine writes a labeled tape in S format. The job deck must contain a REQUEST card specifying a file DUMTAPE on a new S tape before the call to DUMPF. A job deck must contain a REQUEST card specifying a file DUMTAPE on an existing S tape before a call to LOADPF. Example:

```
REQUEST(DUMTAPE,HY,S,N)   for DUMPF
REQUEST(DUMTAPE,HY,S,E)   for LOADPF
```

DUMPF

The DUMPF utility is used to clear permanent files from a mass storage device or to maintain selectively a file back-up system. Directives may be used to specify the files to be dumped. Files must be dumped to a labeled S tape.

Before DUMPF can be run, a permanent file named DUM with an ID of PUBLIC must be cataloged on the device set to be dumped. DUM must have passwords as defined below. These passwords provide the installation some security for its files.

TK=DUMPF	Value assembled into DUMPF program
RD=xxxxx	Password used to call DUMPF for mode 1 dump
MD=yyyyy	Password used to call DUMPF for mode 2 dump
CN=zzzzz	Password used to call DUMPF for mode 3 dump

Several copies of the DUMPF utility may be executing at the same time. All copies running simultaneously must be the same mode and type; the DUMPF parameters must be identical. If an attempt is made to run a DUMPF with different parameters than one already running, all jobs having issued a DUMPF, except the first one, will be aborted.

DUMPF must be on a system library. Parity errors will be flagged. No files should be attached to a job containing DUMPF operations during DUMPF execution.

Before the DUMPF is issued, a REQUEST card must define the dump file, DUMTAPE, as a new S tape.

The format of the DUMPF control card is:

DUMPF(p1,p2,....,pn)

PW is the only required parameter. The parameters listed below may be included. An unrecognizable parameter will cause DUMPF to abort.

PW=x	Password that allows DUMPF to run. The DUMPF password is defined when the file DUM is cataloged. The password specified must correspond with the mode of dump desired.
LF=lfm	File to receive output listing. Default is OUTPUT.
MO=n	Mode in which DUMPF runs. The value of n may be one of the following: <ol style="list-style-type: none">1 Permanent files are dumped as backup, leaving the PFC, PFD, and all associated disk space intact. The read password must be specified. MO=1 is the default.2 Permanent files are dumped and associated disk space is released; the PFD and PFC entries for the files are left intact. Flags are set in the PFD to indicate the file is no longer disk resident. The modify password must be specified. This is called an archive dump.

- 3 Permanent files are dumped and the PFD, PFC, and all associated disk space is released as the permanent files are dumped. The control password must be specified. Before DUMPF will execute with MO=2 or MO=3, the operator must respond to a message:

**THIS WILL BE A DESTRUCTIVE PERMANENT FILE DUMP,
TYPE GO OR DROP**

CL	Complete list option designator. If CL is specified, all files in the directory are listed, whether or not they are dumped. If CL is not specified, only those files dumped are listed.
DP=A	Dump all files; this is the default.
DP=X	Dump expired files.
DP=C	Dump files modified, renamed, created, or extended since the last DP=C dump was taken.
ID=owner	Files with this owner identification are dumped.
PF=pfn	Files with this permanent file name are dumped. ID=owner must be specified also.
CY=nnn	Permanent file with specified pfn, ID, and cycle will be dumped. If PF and ID are not specified, CY is ignored. If the requested cycle of the permanent file cannot be found, a message is issued and DUMPF continues.
IN=ddd	Files inactive this number of days are dumped. See also the TI parameter description.
JN=yyddd	Files inactive since this ordinal date are dumped. See also the TI parameter description.
LA=mmddy	Files last attached before this date are dumped. See also the TI parameter description.
DA=yyddd	Files created, modified, renamed, or extended after this date are dumped. Also see the TI parameter description.
CD=mmddy	Files created, modified, renamed, or extended after this date are dumped. See also the TI parameter description.
TI=hhmm	This four-digit time parameter modifies the time associated with CD, DA, JN, LA, or IN parameters. If none of the latter is specified, TI is ignored. Example: LA=031573, TI=1200 files attached since noon on March 15, 1973 will not be dumped. Files not attached since noon on March 15, 1973 will be dumped.
SN=setname	Only files on this device set will be dumped. The master device for this set must have been mounted before the DUMPF card is issued.

VSN=vsn

Only files residing on this device will be dumped. If dumping from a device set, the SN parameter also must be specified. If dumping from a system device, VSN specifies an EST ordinal and no SN parameter is used.

I=ifn

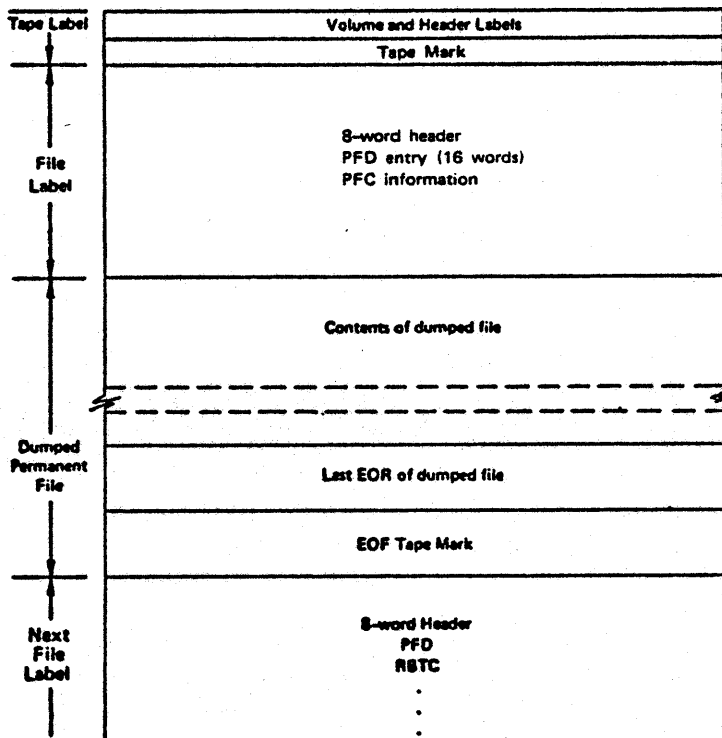
Directives will be read from this file. If I is specified, but not equivalenced, INPUT is used. If I is not specified, no directive file is used. The directive file is used to specify particular files to be dumped. If a directive file is used, the only other parameters that can be specified are SN, MO, CL, and PW. The only valid parameters for directives are permanent file name (PF=pfm), cycle number (CY=nnn), and owner identification (ID=owner). If directives are used, the date and time of the last dump are not modified. Directives can be used only for mode 1 dumps. The format of a directive card is:

PF=pfm,CY=nnn,ID=owner

The output listing will contain the following items for each cycle:

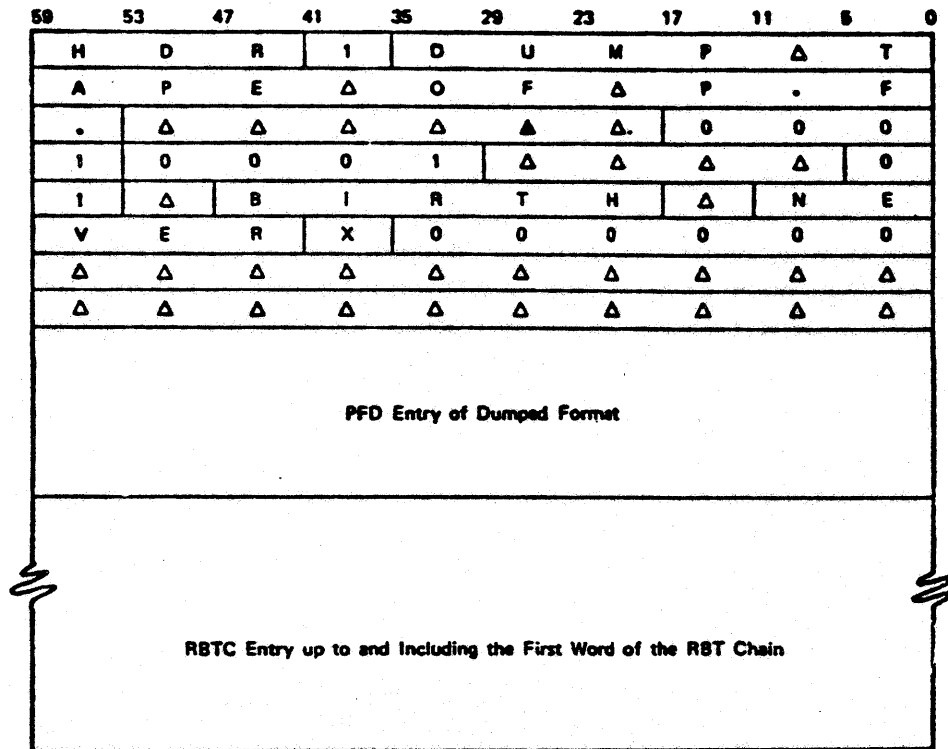
- Permanent file name
- owner ID
- cycle number
- volume serial number
- date of last dump
- comment

DUMPF writes the dump to a multi-volume magnetic tape file having the format shown in the following figure:



Permanent File Dump Tape Format

8-Word Header (format is same as file header label on tape)



Information in the first 20 characters of tape label:

HDRIDUMPF TAPE- NEW

Or:

VOLIDUMPF TAPE- UNIT

FIELD LENGTH (No. of characters)	FIRST CHARACTER POSITION	FIELD DESCRIPTION	NAME OF FIELD
3	1	HDR	Label identifier
1	4	1	Label number
17	5	DUMP TAPE OF P.F.	File label name
6	22	(blank)	Multifile identification
4	28	0001	Reel number
4	32	0001	Multifile position number
4	36	(blank)	Reserved for tape compatibility
2	40	01	Edition number
1	42	(blank)	Reserved for tape compatibility
5	43	BIRTH	Creation date
1	48	(blank)	Reserved for tape compatibility
5	49	NEVER	Expiration Date
1	54	X	Security
6	55	000000	Block count
20	61	(blank)	Reserved for tape compatibility

DUMPF EXAMPLES:

1. JOB.
REQUEST(DUM,*PF)
COPYBR(INPUT,DUM)
CATALOG(DUM,DUM,TK=DUMPF,RD=PERM1,MD=PERM2,CN=PERM3,PW=public password)
7/8/9
.
.
7/8/9
6/7/8/9

This job catalogs DUM on the system set prior to running DUMPF.

2. Full dump of all permanent files, assuming DUM was cataloged as in example 1. Files on user device sets will not be dumped.

JOB.
REQUEST(DUMTAPE,HY,S,N)
DUMPF(PW=PERM1)
6/7/8/9

3. Dump cycle 3 of permanent file TESTFILE with ID of SAM, releasing disk space and the PFD and PFC entries for this cycle.

JOB.
REQUEST(DUMTAPE,HY,S,N)
DUMPF(PW=PERM3,PF=TESTFILE,CY=3,MO=3,ID=SAM)
6/7/8/9

4. Dump all files cataloged under the ID of MARLOWE, releasing disk space but keeping directories intact.

JOB.
REQUEST(DUMTAPE,HY,S,N)
DUMPF(PW=PERM2,ID=MARLOWE,MO=2)
6/7/8/9

Dump certain files of device set SPECPRJ whose master device has volume serial number 479.

JOB.
MOUNT(SN=SPECPRJ,VSN=479) Mount master
REQUEST DUM,*PF,SN=SPECPRJ.
REWIND DUM.
CATALOG DUM,TK=DUMPF,RD=PERM1,
MD=PZ,CN=P3,PW=public password.
RETURN(DUM)
REQUEST(DUMTAPE,HY,S,N)
DUMPF(I,PW=PERM1,SN=SPECPRJ)
7/8/9
PF=COBOL,CY=39,ID=RON
PF=FTN,ID=SLEITER
PF=RUN,CY=79,ID=CDCSVL
6/7/8/9

LOADPF

The LOADPF utility loads permanent files that have been dumped to tape. All (or a selected portion of) files on the tape may be loaded. An optional directive file specifies individual files to be loaded. Multiple copies of LOADPF may execute at the same time. A job may access a file as soon as it is entered into the permanent file tables. For each cycle loaded, LOADPF makes an output listing entry that contains the permanent file name, owner ID, cycle number, date of last dump, and a comment.

If the dump tape contains a duplicate permanent file name, owner ID combination for a file to be loaded, a message is sent to the operator and the file is ignored unless LP=R is specified. If a parity error occurs during loading, the file is flagged in the permanent file tables. LOADPF aborts if it encounters an unrecognizable parameter in the program call statement.

Before LOADPF is called, a REQUEST or LABEL control card must define a load file with the name DUMTAPE, residing on an existing labeled S tape.

The format of the LOADPF control card is the following:

LOADPF(p1,p2,...,pn)

All parameters are optional. Permissible parameters are:

LP=A	Load all files; this is the default. Existing files are not replaced unless the file is incomplete or not disk resident.
LP=P	Load files only if present in the permanent file directory and not disk resident (archived).
LP=X	Do not load expired files.
LP=R	Replace existing files. Both X and R may be specified in the form LP=X,R.
LP=O	Permanent file dump tape is in SCOPE 3.2 or 3.3 format. If LP=O is not specified, the tape is assumed to be a SCOPE 3.4 permanent file dump tape. The O option may be used with other LP parameters in the form LP=R,O,X.
LF=lfn	Use logical file name lfn as output file. The default is the file OUTPUT. The output listing will be written to this file.
CL	Complete list designator. If this parameter is specified, all files on the dump tape will be listed, whether or not they are loaded. If CL is omitted, only loaded files will be listed.
SN=setname	Specifies the name of the device set to which the files are loaded. The master device of this set must have been previously mounted.
VSN=vsfn	If loading a device set, specifies the volume serial number of the RMS device onto which the permanent files are loaded. The SN parameter must also be included, and the master device of the set must have been previously mounted.
VSN=eq	If loading a system device, specifies the EST ordinal of the RMS device onto which the permanent files are loaded.

ID=owner All files with this owner ID will be loaded.

PF=pfm All files with this permanent file name and the specified owner ID will be loaded. ID=name must also be supplied.

CY=nnn The cycle of the specified permanent file name/owner ID is loaded. ID=owner and PF=pfm must also be given. Unless a specific cycle number is given, all cycles of the file will be loaded.

I=ifn Read directives from a file with this logical file name. If I is not specified, no directives are used. If I is specified but not equivalenced, directives are read from the file INPUT. The permanent file name (PF), cycle number (CY), and owner identification (ID) may be specified. If a directive file is used, only those files specified are loaded. The format of the directives is the following:

PF=pfm,CY=nnn,ID=owner

If a directive file is used, the only other parameters that can be specified are CL and SN.

ARCHIVE FEATURE

The archive feature is automatic, and the user will be unaware of this system activity when loading is from tape. When an attempt is made to attach an archived permanent file, the operator may type n.GO which causes the file to be loaded from the appropriate dump tape. An archived file can be purged without actually loading the file.

LOADPF EXAMPLES

```
JOB1.
REQUEST(DUMTAPE,HY,S,E)
LOADPF.
6/7/8/9
```

This job loads all files on the tape unless LOADPF finds the owner ID, permanent file name, and cycle number combination already in the system; such files are skipped.

JOB2.
REQUEST(DUMTAPE, HY, S, E)
LOADPF(LP=X)
6/7/8/9

This job loads all non-expired permanent files from tape.

JOB3.
REQUEST(DUMTAPE, HY, S, E)
LOADPF(PF=STARTREK, ID=SPOCK)
6/7/8/9

All cycles of the permanent file STARTREK with owner ID SPOCK are loaded unless one of the following conditions arises:

The permanent file name/owner ID combination already exists in the system with different passwords.

A duplicate cycle number is encountered.

The permanent file name/owner ID combination already has five cycles cataloged.

JOB4.
REQUEST(DUMTAPE, . . .)
LOADPF(I)
7/8/9
PF=PASSERIFORMES, CY=21, ID=VEERY
PF=ANATINAE, ID=GADWALL
PF=PROCELLARIIFORMES, ID=FULMAR
6/7/8/9

This job loads the specified permanent files from tape.

TRANSPF

The TRANSPF utility can do either of the following:

Change the residence of files and permanent file tables within a device set so that all permanent file information may be removed from a device. (single device set TRANSPF)

Copy files from one device set to another. (dual device set TRANSPF)

The format of the control card is the following:

TRANSPF(p1, p2, . . . , pn)

The TS parameter is always required. The PW parameter is required if passwords have been defined for the file DUM. The format of the parameters is the following:

PW=a,b,c Specifies read, control, turnkey, and modify passwords if they are defined for the permanent file DUM. If passwords have been defined for the file DUM, they must be specified with this parameter or the utility will abort. No default exists.

- FS=set1 Name of the device set from which the permanent file information is to be transferred.
- TS=set2 Name of the device set to which the permanent file information is to be transferred.
- FM=vsn1 Volume serial number (VSN) of the member device from which the permanent file information is to be transferred. Required when set1=set2. Cannot be specified when set1≠set2. Assumed value when set1≠set2 is all devices in set1.
- TM=vsn2 Volume serial number (VSN) of the member device to which the permanent file information is to be transferred. Data that cannot be contained on this device will overflow to another member of the device set specified by TS. Cannot be specified when set≠set2. If set1=set2, files will not overflow into the FM device. Default is all devices in set2. If set1=set2 and FM specifies a master device, TM is required.
- LF=lfm Name of the file on which the output listing is written. Default is OUTPUT.

Before the TRANSPF utility can be executed, a permanent file with the name of DUM and ID of PUBLIC must be cataloged on the device set specified by the FS parameter. If this is not done, TRANSPF will abort.

TRANSPF will issue an internal ATTACH of the permanent file DUM; the passwords submitted in this ATTACH will be those that are submitted via the PW parameter on the TRANSPF request. If TRANSPF is unable to attach the permanent file DUM, the function will abort.

To prevent overflow on PF assignment, the FM device is changed to a non-PF device prior to data transfer.

Before the TRANSPF utility is called, a MOUNT must be issued for the master devices of the device sets specified by the FS and TS parameters. TRANSPF must have exclusive access to the set; if the set is mounted on shared drives, no other MF can be logged in (have set mounted).

SINGLE DEVICE SET TRANSPF (set1=set2)

A single device set TRANSPF is requested if the device set specified by the FS parameter is the same as the device set specified by the TS parameter.

TRANSFERRING FROM A MEMBER

If the FM parameter does not specify a master device, permanent files residing on the FM device will be moved to the TM device. A file will be moved if any part resides on the FM device. Once the file has been transferred, the disk space associated with the old copy will be released. If the file cannot be completely contained on the TM device, the file will overflow to another device in the set, but will not overflow onto the FM device. If the transfer of a file is unsuccessful, that file will be skipped, but TRANSPF will not abort. A file transfer may be unsuccessful because of:

- uncorrectable parity errors;
- not enough space in the device set to accommodate two copies of the file simultaneously;
- permanent file catalog full.

When all permanent file information is successfully transferred from the FM device, that device will be designated as a non-pf device.

TRANSFERRING FROM A MASTER

When the FM parameter specifies a master device, the device set tables will be moved to the device specified by TM, and the device labels for both devices will be updated to reflect the new organization of the device set. If the tables cannot be successfully moved, the device set will not be changed by the TRANSPF utility. Table transfers may fail for one of the following reasons:

- uncorrectable parity errors;
- not enough space on the TM device to completely contain the disk tables.

After the master device is successfully changed, permanent files residing on the FM device will be moved to the TM device as described above. When all permanent file information is transferred from the FM device, that device will be designated as a non-PF device.

Examples

```
FIRST.  
MOUNT(SN=TEST,VSN=999) Mount master  
TRANSPF(FS=TEST,TS=TEST,FM=999,TM=111,PW=A)  
6/7/8/9
```

This job transfers all permanent files and permanent file tables from the master device with VSN of 999 to the member device with VSN of 111. Both devices belong to device set TEST. Note that the member device with VSN 111 was not explicitly mounted. The system initiates the mount of the member when actual I/O is requested by TRANSPF. If this job runs successfully, device 111 will be the master device of set TEST.

If the tables do not fit on the device with VSN 111, the set will not be changed, and the job will end. If the tables are successfully transferred, but the permanent files do not fit on the device with VSN 111, the files will overflow to any devices in the set TEST except the device with VSN 999.

The permanent file DUM is assumed to have been previously cataloged with password A on device set TEST.

```
SECOND.  
MOUNT(SN=TEST1,VSN=555) Mount master  
TRANSPF(FS=TEST1,TS=TEST1,FM=888,TM=222,PW=Q)  
6/7/8/9
```

This job transfers all permanent files from the member device with VSN 888 to the member device with VSN 222. Both members belong to the device set TEST1. Note that the members with VSNs of 888 and 222 were not explicitly mounted. The system initiates the mount of these members when actual I/O is requested by TRANSPF.

DUAL DEVICE SET TRANSPF (set1#set2)

A dual device set TRANSPF is requested if the device set specified by the FS parameter is different from the device set specified by the TS parameter.

TRANSPF transfers permanent files by simulating the following sequence of control cards:

```
REQUEST(lfn2,SN=set2)
ATTACH(lfn1,pfn,ID=owner,SN=set1)
COPY(lfn1,lfn2)
CATALOG(lfn2,pfn,ID=owner)
RETURN(lfn1,lfn2)
```

All files residing on the device set specified by the FS parameter are transferred to the device set specified by the TS parameter. The FM and TM parameters cannot be used; no member devices can be specified.

A permanent file transfer will be unsuccessful if one of the following conditions exists:

- file1 has an uncorrectable parity error;
- CATALOG is unsuccessful, e.g. PFD or PFC full in TS set;
- not enough disk space is available on the TS device set to contain file1.

After a successful transfer of a file, two copies of the file exist, one in the FS device set and one in the TS device set.

Note that in a dual device set TRANSPF the disk tables are not moved as a separate entity; critical tables are only moved within a device set and never from one device set to another.

Example

```
JOB.
MOUNT(SN=BOB,VSN=1944) Mount master
MOUNT(SN=TOM,VSN=1984) Mount master
TRANSPF(FS=BOB,TS=TOM,PW=YES)
6/7/8/9
```

This job moves permanent files from the device set BOB to the device set TOM.

AUDITING PERMANENT FILES

With the AUDIT utility, the user can obtain the status of permanent files. The user can restrict the AUDIT to a specific user ID, permanent file name, or device set. The AUDIT utility provides printed reports containing basic accounting information. AUDIT runs in two modes. Full mode produces a printed report containing all the information listed below. A partial audit produces only the items marked with *.

*Setname	*Account name	Archive flag (VSN printed)
*Permanent file name	*Size in number of PRUs	Time of last attach
*Owner ID	*Public device EST ordinal	Time of last alteration
*Cycle number	Number of attaches	VSNs of files dumped
*Creation date	Number of extends	Loading parity error flag
*Expiration date	Number of rewrites/alters	Positioning flag
*Date of last attach	File size (number of RBs)	Random flag
*Date of last alteration	Subdirectory	SAAM flag
	RB conflict flag	New version flag

The format of the AUDIT control card is:

```
AUDIT(keyword=value, ...)
```

All parameters are optional. The following parameters may be used with AUDIT:

LF=lfm	Logical file name to receive the output listing created by AUDIT. Default is OUTPUT.
MO=m	The mode of the AUDIT. Only one mode may be specified. The value of m may be one of the following: A AUDIT all files; this is the default value. X AUDIT expired files. D AUDIT inactive cycles. I AUDIT incomplete files. P AUDIT files with parity errors. R AUDIT archived files.
ID=name	Owner identification; AUDIT all files with a particular owner identification.
PF=pfm	Permanent file name; AUDIT all files with this pfm. If this parameter is used, the ID=name parameter must also be used.
AI=F	Produces a full two-line output for each file audited. This is the default.
AI=P	Produces a partial, one-line output for each file audited.
SN=setname	Device set to be audited. The master device for this set must have been previously mounted.
VSN=vsf	Volume serial number of device to be audited. All files residing on the device will be audited. The master device for this set must have been previously mounted. The SN=setname parameter must also be specified.
AC=n	Account number; AUDIT all files with this 1-to-9-character account number.



	All Files	Archived Files	Expired Files	Files of Same ID	Files on a Certain Unit	Partial Audit	Full Audit or Account
Owner ID	X	X	X	X	X	X	X
Permanent File Name	X	X	X	X	X	X	X
Cycle Number	X	X	X	X	X	X	X
EST ORDINAL (If on a Public Device)	X		X	X		X	X
Creation Date (ordinal)	X	X	X	X	X	X	X
Expiration Date (ordinal)	X	X	X	X	X	X	X
Date of Last Attach (ordinal)	X	X	X	X	X	X	X
Date of Last Alteration (ordinal)	X	X	X	X	X	X	X
Account Parameter	X	X	X	X	X	X	X
Number of Attaches	X	X	X	X	X		X
Number of Extends	X	X	X	X	X		X
Number of Rewrites/Alters	X	X	X	X	X		X
VSN of dump tapes (first/last)	X	X	X	X	X		X
Length (No. PRUs determined by Inst. Par.)	X		X	X	X	X	X
Flags†	X	X	X	X	X		X
Subdirectory Number	X	X	X	X	X		X
Length in RBs	X		X	X	X		X
Time of Last Attach	X	X	X	X	X		X
Time of Last Alteration	X	X	X	X	X		X

†Flags set will be represented by the following letters produced on AUDIT output:

R Random file
S SAAM file
A Archived file

N New version file
E Parity error in file
P Positioned
C RB conflict

Figure 6-2. Table of AUDIT Outputs

Permanent File Utility Function Codes (octal) for system interface between PF Utilities and PFM routines:

AUDIT	166
DUMPF	172
LOADPF	174
TRANSPF	176

Permanent File Utility Keyword Codes (octal):

Keyword	Code	Description
		(...) indicates format of values in FDB
TR	57	Dump tape type 7 or 9 track (binary)
MO	60	Load or audit dump mode (display code)
UN	61	Dump, load or audit unit EST ordinal (binary)
DA	62	Date parameter for file dumps (display code)
DP	63	Dump mode parameter (display code)
IN	64	Inactive file dump parameter (binary)
E1	66	Transfer files tables from unit EST ordinal (binary)
E2	67	Transfer files tables to unit EST ordinal (binary)
LP	70-73	Load type parameter (display code)
JN	74	Date parameter for file dumps (display code)
TI	75	Time parameter for file dumps (display code)
AI	76	Audit type parameter (display code)
LF	77	Audit output file parameter (display code)
ID	14	Load by file ID name parameter (display code)
PF	30	Load by ID and pfn names parameters (display code)
PW	20-24	Passwords to attach file DUM (display code)
CY	03	Load by ID, pfn and cycle parameters

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 can be set using an interlocked stack request and released by making a stack request without interlocks. They are retained when interlocked stack requests continue from the same PP or to the same FNT as with the first stack request.

Set interlocks in the label are used to prevent set routines (such as ADDSET) 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 will access 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 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 set routines from running on the same mainframe. (Note that 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 member devices, 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, etc. 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 dismounted 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 dismounted flag, and then rewriting the SMT releasing the hardware interlock. They can be released by reading the SMT with hardware interlock (necessary because the SMT interlock only interlocks the SMT entry, not the entire PRU containing the entry), testing and clearing the interlock if not already cleared, and then rewriting the interlock.

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 may allow jobs 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 multi-mainframe environment, the 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 interlocks in the MST can be set by reserving the MST channel, setting the interlock, and then releasing the MST channel. They can be released by reserving the MST channel, reading the MST entry, testing and clearing the interlock if still set or else allowing error exit for the interlock if not still set, and finally releasing the MST channel.

The utility interlock bit in the MST is used to TRANSPF in conjunction with the set interlock bit in the master device label after testing that no other mainframe has the set logged-in and that the permanent file manager activity is quiet on the set. This will both assure the host mainframe exclusive access to the set while transferring disk tables and also prevents CATALOG and ATTACH functions on the same mainframe. This bit 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 finally releasing the MST channel.

The APF interlock (CH.APF) is used for interlocking the APF table, preventing two or more routines from simultaneously accessing or updating the table. With the single PFM interlock, an empty APF slot is reserved before the PFD search in the PFA, PFC, and LPF; furthermore, the PFD, APF, and PFC processing are all performed under the same PFM interlock. 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 formerly called the RBT catalog (RBTC). These names may 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's 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, four per PRU.

The PFD body will have n ID hash points. These hash points will be equally spaced within the body and the space between any two ID hash points will be an integral multiple of pages and this multiple will be a power of two. 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 will be a PFN hash point.

A sub-directory 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 multi-mainframe environment exclusive access can be granted. The first flag signifies that this cycle has been dumped and is unavailable (retained for 3.3 files by CONVPF only); the second, 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 may specify the number for the cycle or permit a cycle number to be generated automatically. Unless the user specifies the cycle when attaching a file, he will get the one with the highest cycle number. 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's ID.

The entry-in-use flag in word zero of the PFD is set when an entry is built. Detailed formats of the PFC and PFC entry appear in part II, section 3.

Detailed formats of other tables relating to device sets are included in part II 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 flaw table and physical flaw tables reserve space on a device for areas with hardware flaws. PFC entries are always allocated in blocks of 64 words. One PFC entry may occupy several blocks which must be contiguous.

Figure 6-7 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 first word of an FNT entry or cataloged for a permanent file relates that entry to the corresponding APF entry. The APF table is an interlock list which allows multi-read access without reloading the RBT chains and also controls queuing for exclusive use of the file. An empty APF entry can be reserved by setting the APF reserved flag in bit 0 of word 2.

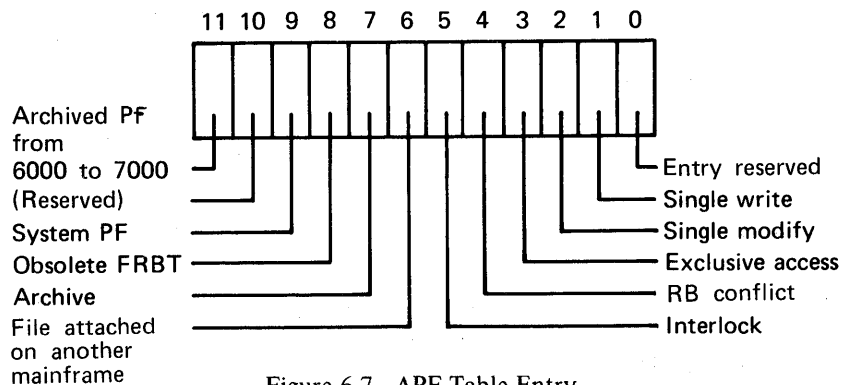
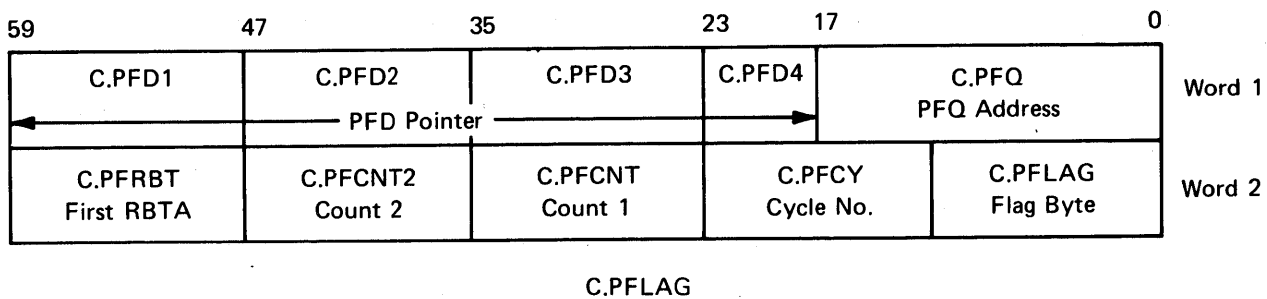


Figure 6-7. APF Table Entry

INCOMPLETE CYCLE

An incomplete cycle is a file that has a PFD entry but no RBTC 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 two. The PFC body is initialized to binary zeros.

PFC entries are always allocated in blocks of 64 words. One PFC entry may occupy several blocks which must be contiguous.

PERMANENT FILE ACCOUNTING

When W.CPFACT in the control point area is non-zero, 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 SCOPE 3.4 Reference Manual. The ACCOUNT card will set W.CPFACT non-zero.

PRIVATE DEVICE SET PROCESSING

A device set is a group of one or more rotating mass storage devices. Files may be written to device sets and made permanent. Device sets may be used simultaneously by many jobs. A file will not be written on a private device set unless specifically directed to the set.

The control cards described in this section are for creating, maintaining, and using device sets. With the exception of REQUEST, these functions are not available through macros. Private device sets are supported only on removable devices such as CDC 841, 854, and 844 disk packs.

DEVICE SET CREATION AND MAINTENANCE

The first step in establishing a private device set is to blank-label the packs using LABELMS. LABELMS must be run in any case, even if DELSET has been run on the device.

The next step is to use the ADDSET control card to establish a master device for the set. The master device contains all the device set related tables.

Once a master device has been established, the device set exists. Packs can then be added to the set with ADDSET or deleted with DELSET. The RECOVER function can be used to verify the device set tables.

The LABELMS and ADDSET control cards may not be entered through INTERCOM.

DEVICE SET LABELING (LABELMS)

The LABELMS utility is used to blank-label a rotating mass storage device. When LABELMS is executed (except during deadstart), the system requests the equipment status table (EST) ordinal and volume serial number (VSN); these must be assigned by the operator. If a device type (DT) is not specified in the statement parameter list, the operator may specify the device to be blank-labeled. LABELMS can also be used to inhibit preallocation of space for C.E. diagnostics, specify allocation information for subsequent access to the device, and record known flaws on a device so that such areas are not accessed. 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.

The statement format is:

LABELMS(DT=tt,NPR,DS,EST=est,I=ifn)

Parameters are optional

DT=tt Device to be labeled; device codes for tt are:

AA	6603-I Disk
AB	6638 Disk
AC	6603-II Disk
AL	821 Disk
AM	841 Disk Pack
AP	854 Disk Pack
AY	844 Disk Pack

NPR No space is to be preallocated for C. E. diagnostics. If NPR is not specified, space will be allocated at the following cylinder numbers (in octal):

Device	Cylinder
6638	37
6603	177
821	777
841	311
854	311
844	632

DS Deadstart parameter, not available to users, used only by the operating system to call LABELMS during post-deadstart.

EST=est EST ordinal used only when DS is specified; not available to users.

I=ifn Logical file name for input directives containing allocation and flaw information. If I is specified but not equivalenced to a file, the default file name is INPUT. If I is omitted entirely, no directives are expected, default allocation information is used and the disk is presumed to be free of flaws. DT must be specified if I is specified.

The directives have the following format:

Allocation directives: Aas,Rpru,Nrbs.

Flaw directives:	{	Ppn,Hhn,Ssn.	} Devices 6638 and 6603
		Ppn,Hhn,Sfsn-lsn.	
	{	Ttn,Ccn,Ssn.	} Devices 841, 821, 854, and 844
		Ttn,Ccn,Sfsn-lsn.	

Use one allocation directive for each DAM (RBR) desired on the device.

as	Allocation style. The limits of as are 0 to 77 octal; default is as=0. The allocation style enables the user to request a specific allocation feature, such as directing a file to a specific portion of a device which may have a particular record block size and/or recording technique.	
pru	Number of PRUs per record block. The limits of pru are 2 to 7777 octal, except that for every device, pru must be greater than or equal to 1/32 of the RB and less than or equal to 32 times the RB. Default depends on the device; see the table below. If the default value for pru is not desired, the value specified should be either an integral multiple or an integral factor of the default value for the device.	
rb	Number of record blocks in the RBR (record block reservation table) for this device or portion of device. The RBR, maintained by SCOPE in central memory, contains information indicating its associated allocation style and the status (available or not available for assignment) of all record blocks governed by this RBR. The limits of rb are 1 to 7777(octal). Default depends on the device; see the table below.	
pn	Position number.	} Limits depend on device; see the table below.
hn	Head group number.	
tn	Track number.	
cn	Cylinder number.	
sn	Sector number.	} Indicate several contiguous flawed sectors.
fsn	First sector number.	
lsn	Last sector number.	

NOTE

User packs may have one RBR per pack, with $R \geq$ standard and $N \leq$ standard, and all packs in a set must be the same.

For both user and public packs: No more DAMs may be created than are configured in CMR for the drive; and total RBs for the device must not exceed the total size of the RBRs configured in CMR.

DEVICE RELABELING

RELABEL is used by deadstart to rewrite the label on RMS devices in order to add or delete flaws or to recreate a label that has been destroyed. The format of the RELABEL control card is:

RELABEL(VSN=vsn,SN=setname,MP=mp,I=0,O=0,DS)

vsn Volume serial number of device to be relabeled.

setname Name of the device set containing the device.

mp Volume serial number of the master device of the device set.

I=0 Specifies no input file.

O=0 Specified no output file.

DS Specifies that RELABEL is being run under deadstart.

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.

Device	pru Default	rbs Default	pn, tn limits	hn, cn limits	sn limits
6638	62	4000	0 to 37	0 to 37	0 to 143
6603-I	62	2000	outer 0-3 inner 4-7	0 to 177	0 to 177 0 to 143
6603-II	100	2000	outer 0-3 inner 4-7	0 to 177	0 to 177 0 to 143
841	70	1750	0 to 23	0 to 307	0 to 15
821	466	4000	0 to 77	0 to 777	0 to 23
854	4	5050	0 to 11	0 to 307	0 to 17
844	70	6240	0 to 22	0 to 632	0 to 27

All values in the above table are octal. All values in the directives are assumed to be octal, but are interpreted as decimal if suffixed with a D.

Each directive must begin in column one and end with a valid terminator. Valid control separators must appear between the elements of a directive. Successive allocation directives must refer to successive portions of a device; allocation directives may be intermixed with flaw directives. A maximum of 8 allocation directives can be provided.

ADDING DEVICES TO A DEVICE SET (ADDSET)

The ADDSET control card establishes a master device and adds members to a device set. A master device, and therefore a device set, is created by ADDSET when the MP and VSN parameters indicate the same volume serial number. A member device is added to an existing device set when the MP and VSN parameters specify different volume serial numbers. A job that adds a member to a previously created device set must issue a MOUNT of the master device before the ADDSET is issued. The format of the ADDSET control card is:

ADDSET (p1,p2, ... ,pn)

The following parameters are required:

- MP=vsn Volume serial number of master device for set (1-6 alphanumeric characters, leading zeros assumed). If MP and VSN parameters specify the same volume serial number, a master device is established. If MP and VSN specify different volume serial numbers, a member is added to an existing device set.
- VSN=vsn Volume serial number of device being added to the set (1-6 alphanumeric characters, leading zeros assumed). If vsn is the same for VSN and MP, a master device is established. If the vsn specified by the VSN parameter does not match vsn in the device label, a message identifying the device vsn is generated, and the job is terminated.
- SN=setname Name of device set created or the device set to which a member is added (1-7 alphanumeric characters beginning with a letter). When a member is added, if the setname does not match the setname on the master device a message is generated and the job terminated.

The following parameters are optional:

- NF=n Maximum number of permanent files that can exist on the device set. The default is 300. The value of n cannot be less than one nor so large that the permanent file tables do not fit on the master device. Tables must reside on the first DAM (RBR). This parameter has meaning only for an ADDSET of a master device.
- NM=m Maximum number of members allowed in the device set. The default is 50. The value of m cannot be less than one and cannot exceed 50 (decimal). NM is used by ADDSET to preallocate on the master device system tables for the member devices. For each member RBR, the system will need one PRU if the RBR is less than 62 words long, or two PRUs otherwise. ADDSET will reserve for system tables a number of PRUs equal to twice NM. If there are to be several RBRs for each member device, NM should be specified as somewhat larger than the actual number of member devices. This parameter has meaning only for an ADDSET of a master device.
- RP=ddd Retention period for the device set. The default is 31 days. ddd must be a decimal number (0 to 999) indicating the number of days before the device set expires. 999 indicates an infinite retention period. Thus parameter has meaning only for an ADDSET of a master device.

- *PF Permanent files may reside on the member added to this device set. A device set may have members that can contain permanent files and members that cannot. Although the master device need not be a permanent file device, at least one device in the set must be.
- *Q Queue files may reside on the member added to this device set.
- DS Deadstart parameter, not available to users.
- EST=est EST ordinal specified at deadstart; not available to users.

DELSET cannot be executed on a device set currently being shared.

DELETING MEMBERS (DELSET)

The DELSET control card deletes devices from a set. The master device must be mounted before the DELSET is issued. All member devices must be deleted before a DELSET is issued for the master device. Permanent files and local files residing on the device must be purged or returned before the DELSET is issued; if any portion of a local file or permanent file resides on the device to be deleted, the DELSET request is aborted. The format of the control card is:

DELSET (SN=setname,MP=vsn1,VSN=vsn2)

All parameters are required:

- SN=setname Name of the device set having the member to be deleted.
- MP=vsn1 Volume serial number of master device of this device set.
- VSN=vsn2 Volume serial number of member to be deleted from the device set.

MAINTENANCE (RECOVER)

The RECOVER control card 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.

The format of the RECOVER control card is:

RECOVER(SN=setname,VSN=vsn,L=ifn,MFID=mfid,TERM,MODE=mode)

- setname Name of the device set to be processed.
- vsn Volume serial number of the master device of the device set.
- ifn Logical file name of the file on which any errors encountered in the validation process will be listed; default is file OUTPUT. When MODE = 1 or 2, this parameter need not be specified and is ignored.

mfid In mode 1, id of this mainframe. In mode 4, primary logical id of the mainframe that has become inoperative. When MODE = 0 or 5, this parameter need not be specified and is ignored.

TERM Sequence number assigned to RECOVER by GENDJ for identification when RECOVER runs under the control of deadstart. When MODE = 0 or 4, this parameter need not be specified and is ignored.

mode Specifies mode of RECOVER processing:

MODE=0 (default)

- called by control card
- runs only on an unmounted device set
- performs a complete interlock cleanup
- rebuilds the DAM using the LFT and the PFC
- detects the RB conflicts
- cross checks the PFD and the PFC
- resets the available RB counts in the SMT
- gives a comprehensive listing of all errors
- flags the PFC entries for which RB conflicts have been detected

MODE=1 or MODE=INITIAL (First MF to Deadstart)

- called by GENDJ at deadstart, which provides TERM parameter
- runs only on an unmounted public device set
- 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 and either gives a GO option to the operator (non-fatal errors) or aborts (fatal errors)
- calls HDS at the end of processing and transmits the TERM parameter value and the number of RB conflicts

If the device set is being shared, the mainframe initiating the RECOVER must be the first mainframe in the multi-mainframe system to be brought up.

MODE=2 or MODE=SECONDARY (Not First MF to Deadstart)

- called by GENDJ at deadstart, which provides the MFID and TERM parameters
- runs on a shared device set
- performs interlock cleanup for local MF
- 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 and either gives a GO option to the operator (non-fatal errors) or aborts (fatal errors)
- calls HDS at the end of processing and transmits the TERM parameter value and the number of RB conflicts

MODE=4 or MODE=CLEANUP (Resource Recovery)

- called by control card
- runs on mounted or unmounted device sets
- performs a cleanup of the interlocks left by the inoperative mainframe
- rebuilds (two mainframes) or validates (more than two mainframes) the DAM using the LFT and the PFC and the RBRs and RBTs of the host mainframe
- detects the RB conflicts
- gives a comprehensive listing of all errors

MODE=5 or MODE=ERROR LIST

- called by GENDJ to get a full listing of the errors detected during the preceding call to RECOVER in MODE 1 or 2
- runs on mounted or unmounted device sets
- validates the DAM against the LFT and PFC
- detects the RB conflicts
- cross checks the PFD and the PFC
- gives a comprehensive listing of all errors
- flags the PFC entries for which RB conflicts have been detected

USING DEVICE SETS

Once a device set has been created with the LABELMS and ADDSET control cards, it can be used by jobs. To access a device set, a job must issue a MOUNT control card. Then REQUEST cards with an SN parameter may be used to direct files to the device set. A file will be written on a system device unless a REQUEST card has previously directed that file to a device set. Files may be made permanent on a device set with the CATALOG control card. Files that were previously made permanent may be accessed with the ATTACH control card. The other permanent file control cards are applicable to device sets also. Except for REQUEST, the following functions are not available through macros.

ASSOCIATING A DEVICE SET WITH A JOB (MOUNT)

To access a device set, the user must issue a MOUNT control card for the master device. MOUNT is a logical operation; if the device is physically available, no operator intervention is required. If the device is not physically available, the device name is placed in an operator display, and the job is swapped out until the device is mounted.

When the master device is mounted, the device set tables are read into the system and all files and member devices become logically accessible to the job. When the master is mounted, the system issues a MOUNT for other member devices as needed. The user also may issue a MOUNT for a member device. The format of the MOUNT control card is:

MOUNT(VSN=vsn,SN=setname,DS,EST=est)

The VSN and SN parameters are required; DS and EST are specified only at deadstart.

VSN=vsn	Volume serial number of device to be mounted
SN=setname	Name of device set to which this device belongs
DS	Deadstart flag, used by operating system at deadstart to MOUNT public devices
EST=est	EST ordinal, used by deadstart

For 844 (AY), MNT issues a stack request (with the factory data flag set) to read the pack serial number and manufacture date, and 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 will contain a pack number of *NO S/N* and a date of 000000.)

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.

The CE error file entry is described in detail in the On-line Maintenance Software Reference Manual.

DISASSOCIATING A DEVICE SET FROM A JOB (DSMOUNT)

When a job has finished processing a user device set, the set may be logically disassociated from the job by issuing a DSMOUNT for the master device. If the user does not dismount the set, the system will do so at job termination. A dismount is a logical operation; other jobs using the same device set will not be affected. DSMOUNT also may be issued for a member device.

When a DSMOUNT specifying the master device of a private device set is executed, a close unload is issued for every file in use by the job and residing on the user device set, each mounted member device is dismounted, and finally the master device is logically dismounted. DSMOUNT is inoperative on public device sets.

The DSMOUNT control card has the format:

DSMOUNT,VSN=vsn,SN=setname.

Both parameters are required and have the same meaning as for MOUNT.

DIRECTING FILES TO DEVICE SETS (REQUEST)

The REQUEST card must be used to direct files to specific device sets. All files are written on public devices unless a REQUEST card with the SN parameter has previously directed the file to a specific device set.

The format of the REQUEST card follows:

REQUEST (lfn, { SN=xxx } [,OV,PF,VSN=yyy,dtaa])
 SN

The meaning and use of the parameters are described in the SCOPE Reference Manual. Note that the EC parameter cannot be specified on this form of the REQUEST card.

IMPLICIT REFERENCE TO DEVICE SETS (SETNAME)

The SETNAME control card indicates the device set to be referenced implicitly by subsequent control cards. Before a SETNAME card is executed, system devices are used unless a device set is explicitly specified. Once a SETNAME control card is executed, the specified device set is implicitly referenced by following control cards. A second SETNAME control card overrides the first. When no device set is specified on the SETNAME card, the system devices are assumed. The only control cards affected by a previously executed SETNAME are REQUEST cards with an unequivalenced SN parameter and permanent file control cards. The format of the SETNAME control card is:

SETNAME(setname) or SETNAME.

setname Device set to be referenced implicitly. The master device of this set must be mounted before SETNAME is issued.

ADDRESSING PUBLIC SETS BY SET ATTRIBUTE

A standard FDB is used by the system to return the device set name of public sets by attribute.

The calling sequence is as follows:

SYSTEM FSN,R,fdbadd,code*100B

where

FSN	Program name
R	Recall (mandatory)
fdbadd	Address of FDB into which the correct SN should be placed. A dummy SN should be used when creating the FDB.
code	Set to find
	0 = system
	1 = pf default
	2 = queue
	3 = (reserved)

FSN will place the set name in the SN parameter slot and set the FDB complete bit. FSN will abort if no set exists except for code 3 where a scratch set will be assigned if no buffer set can be found.

JOB FLOW

SCOPE processes jobs in the system in three sequential, independent stages: Input, Scheduling, 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. When the job has terminated, JANUS is assigned to process the OUTPUT, PUNCH, and PUNCHB files produced by the job.

A job comes under scheduler control after all system resources specified on the job card have been assigned to it. When a job is under scheduler control, it has an assigned job description table entry and is in one of the scheduler queues; or it has been assigned to a control point. When it is assigned to a control point, the control point status values will indicate either the type of activity or that activity at the control point has been suspended. With the job swapping/rolling features of SCOPE 3.4, several jobs may be considered to be assigned 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 SCOPE executes the job as directed by the control card record of the job's input file.

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 the PP overlay 2VJ to translate the JOB control card. 2VJ scans the card, 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 11B 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 card, except for central memory. A job will remain 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 7 and/or 9-track tapes (as defined by MT or NT parameters on the job card) are entered in the pre-input queue, a subset of the input queue. The operator communicates with the pre-input queue through DSD type-ins and the P display. Each time the operator requests a P display, the jobs displayed are the highest priority tape jobs in the queue. A job requiring tapes is not placed in the normal job input queue until the operator releases it with a type-in. Once released, the job will be considered for assignment to a control point and execution; it no longer appears in the prescheduling display. Tape drive scheduling is described in Section 4 under non-allocatable devices.

LOADING JOBS FROM TAPE

The CP routine TLOAD is used to load jobs onto disk via magnetic tape. TLOAD is called to a control point by DSD when the operator initiates a keyboard entry in the format:

n.X TLOAD,g,p.

n is the number of the control point to be used.

g is an optional parameter specifying the number of the file group to be loaded.

p is an optional parameter specifying the number of the job in file g to be loaded.

Both g and p are decimal digits. Numbering begins with one, not zero.

A file group is a collection of jobs on tape, terminated by a double end-of-file.

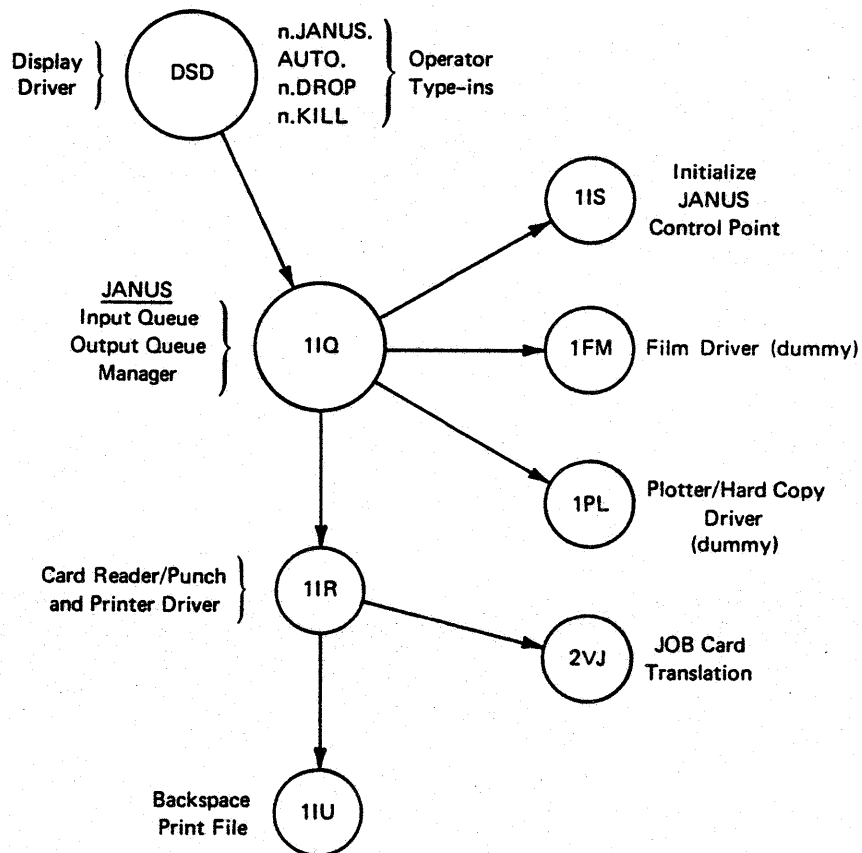
A standard circular buffer technique is used to process input files. The first card of the first record in each file is presumed to be a JOB card, and it is processed accordingly. This assumption is made for each job on the tape. CIO and its overlays are called to read the tape into memory and write memory to disk.

The number of control cards which may appear in a given job is not limited. If the user fails to place an end-of-record (7/8/9) card behind the control card record of a job, the omission will not be detected by either TLOAD, JANUS, or INTERCOM until the job runs.

JANUS

The unit record I/O package - JANUS - consists of the following PP programs:

IIQ	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 type-ins and displays
IIS	Called by IIQ to load a group of overlays into the field length for use by IIR
IIR	Card reader, card punch, and printer driver
IIU	Called by IIR to backspace print files when necessary
IPL	Dummy output file driver for plotter
IFM	Dummy output file driver for film/hard copy devices



JANUS Interfaces

INTEGRATED SCHEDULER

The job scheduling scheme implemented in SCOPE 3.4 is made up of the following:

Scheduler (MMGR)

CPU program operating at control point zero in 2-to 3-millisecond cyclical intervals. It allocates control points and central memory to jobs and determines which jobs should be swapped/rolled in and/or out.

LIB

Brings jobs from input queue to CM queue; builds JDT entries for jobs.

LRN

PP program operating at control point zero at about one-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 (ISI) or out (ISO) and to initiate new INTERCOM jobs.

1DM

PP program which processes job entries in the device queue.

1PF

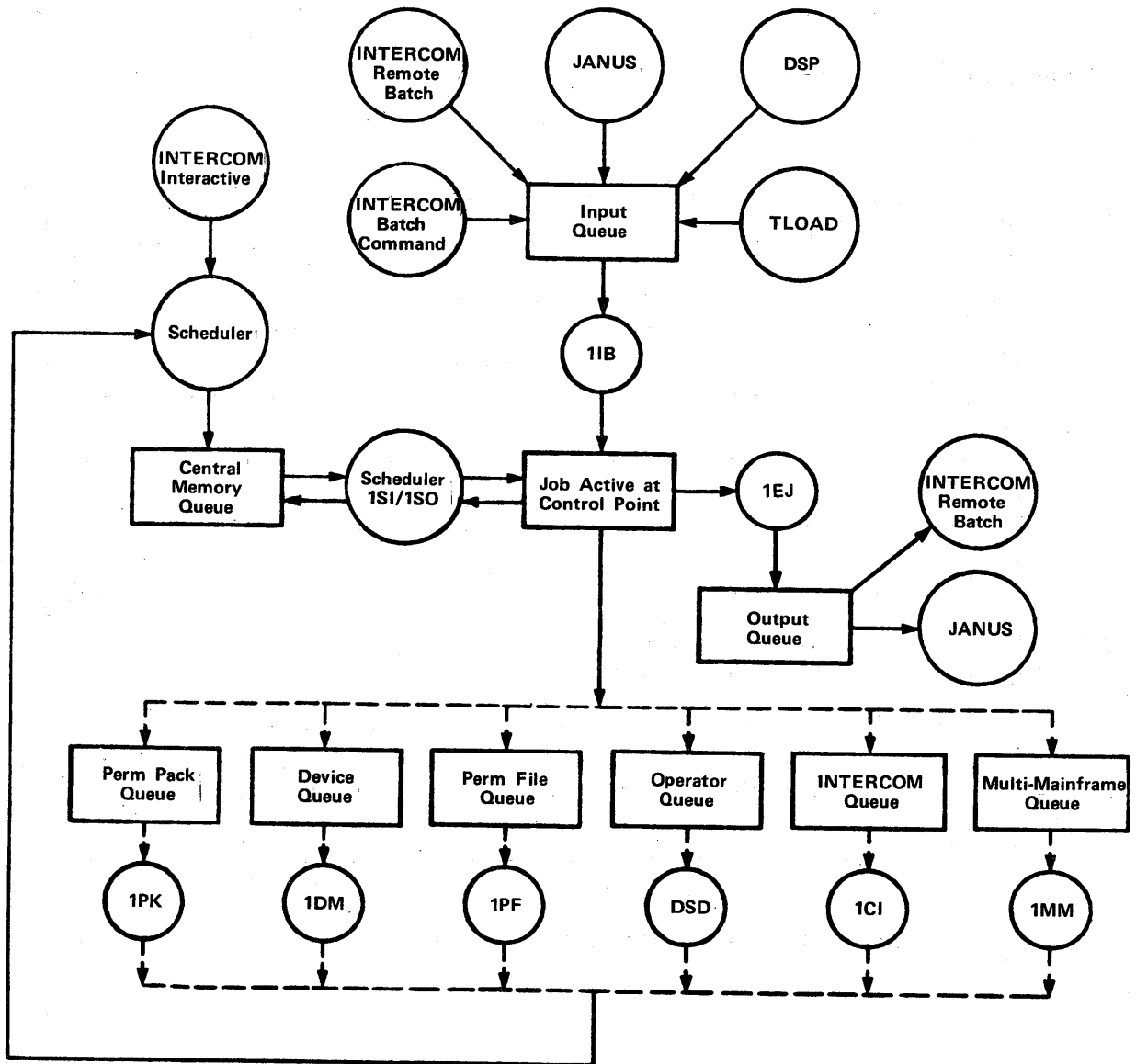
PP program which processes job entries in the permanent file queue.

1CI

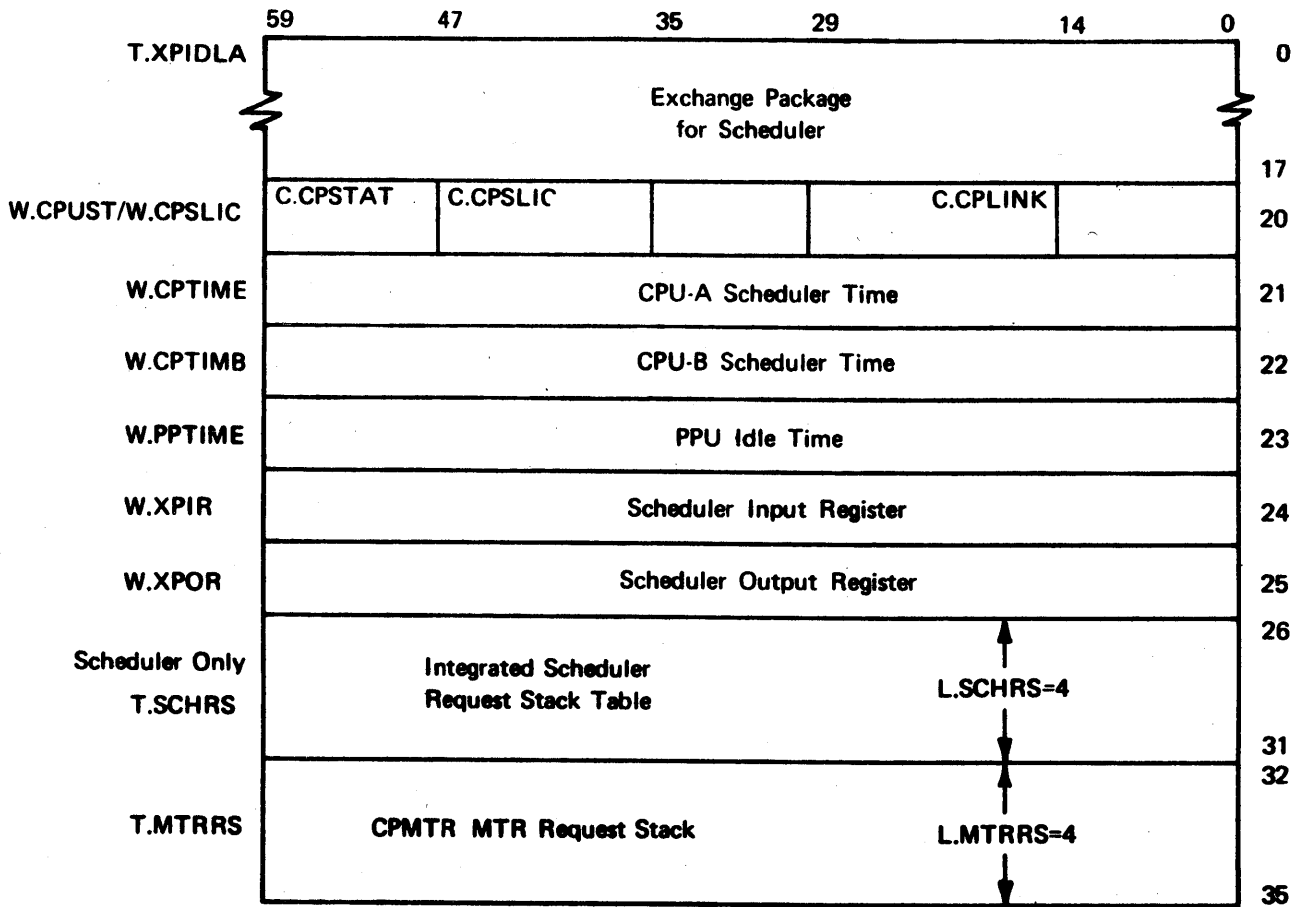
PP program which processes job entries in the INTERCOM queue.

1QP

PP program which processes multi-user job entries in the INTERCOM queue.



INTEGRATED SCHEDULER INTERFACES



SCHEDULER REQUEST STACK IN SYSTEM EXCHANGE PACKAGE AREA

JOB SCHEDULING

In SCOPE 3.4, 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. System jobs such as JANUS and LOADER can not be swapped or rolled.

ROLLING

A job will be rolled out instead of swapped out under the following conditions:

- Job has direct-access ECS assigned

- Job has non-allocatable equipment assigned

- Job is in device queue

- Job is in operator action queue

- Job's field length plus its required positive field length for swapout exceeds IP.MFL (maximum field length a job can request) plus IP.POSFL or 131K.

When a job is rolled out, its field length is written to an allocatable file and all of its 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 begins active use of its control point.

SWAPPING

A job is swapped out under the following conditions:

Job is waiting for CM

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 multi-mainframe action

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. Both the control point and central memory are released and made available for reassignment. The JDT is placed in the appropriate queue; it contains a pointer to the swap file containing the job. The job, when swapped in, may be assigned a different control point and a different area in central memory.

The scheduler can be initiated by a PP program using a M.SCH or M.ICE monitor request. The scheduler is called in the following cases:

1. When a PP program has an entry to be added to the central memory queue, the scheduler will determine if the job has a priority high enough to cause any programs 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 central memory queue.
2. When the field length of a job executing at a control point is reduced, the scheduler is called to search the central memory queue for a new job to be scheduled.
3. The scheduler is called periodically to determine if any job assigned to a control point has completed its quantum time period for CP and/or PP and if any jobs in the central memory queue have been aged enough to cause a swap-out. A job will be swapped out during its quantum time period only if a job with a very high class priority entered the central memory queue.

JOB CONTROL AREA

Jobs entering the system for execution are placed into one of six classes:

1. Batch jobs using no non-allocatable resources
2. Batch jobs using one or more non-allocatable resources
3. INTERCOM (interactive) jobs
4. Multi-user jobs
5. Express jobs (for which operator has entered DROP, KILL or RERUN)
6. Graphics jobs

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 to 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 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 part II, section 1.

A description of the scheduling parameters appears in section I-1 of the SCOPE 3.4 Installation Handbook.

JOB DESCRIPTOR TABLE

Each job to be scheduled for execution, has a job descriptor table (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 scheduler 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 it to be in one of the following job scheduling queues:

Central Memory Queue

Device Queue

Permanent File Queue

Permanent Pack Queue

Operator Action Queue

INTERCOM Queue

Multi-mainframe Queue

JOB SCHEDULING QUEUES

CENTRAL MEMORY QUEUE

The CM queue is made up of jobs waiting for central memory and control point assignment. They are rolled/swapped jobs waiting to be reassigned a control point and/or central memory.

The PP routine IIB scans the job input queue for jobs to be assigned to a control point. When such a job is found, IIB creates the JDT entry and requests scheduler to set the entry in the CM queue. Jobs in other 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 will put the JDT into the device queue and call a swapper to roll out the job. The PP program ITS is responsible for detecting when the requested device is ready and for calling the PP program IDM to put the JDT entry back into the CM queue. When the job is rolled back in, REQ will be 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 will appear on the B display.

PERMANENT FILE QUEUE

When a job executing at a control point makes an ATTACH request for a permanent file, the PP routine PFA is called. If PFA determines that the file cannot be attached and that the job will have to wait, it places the JDT entry into the permanent file queue and calls a swapper to swap out the job.

Whenever an attached permanent file is detached, IPF is called. It checks the permanent file queue for jobs waiting for the detached file. If any exist, IPF will select the highest priority waiting for the file and call the scheduler to put its JDT back into the CM queue. When the job is swapped back in, PFA will be called to attach the permanent file to the job.

If a permanent file is purged, the PURGE routine will search the permanent file queue and call the scheduler to put all jobs waiting for that file back into the CM queue. As each job is swapped in, PFA will be called. It will detect that the file has been purged and respond accordingly.

All entries in the permanent file queue waiting for a given permanent file will be linked together in order of priority. The APF table entry for the file will contain a pointer to the first JDT entry in the chain. When PFA adds an entry to a chain in the queue, it will age all entries in the chain and insert the new entry. Pseudo channel CH.PFM will be used to interlock the queue.

If a job requests a permanent file to be attached with read-only permission and the requested file is already attached to a job with multi-read access, the requesting job can attach the file only if it has a higher priority than any job in the permanent file queue waiting to attach the file.

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 will appear 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 the job is to be put into some other queue. For example, REQ will set the pause bit and then request 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 will be rolled in and initiated 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 set up a job at a control point, the job advancing routine 1AJ is called into a PP. 1AJ reads one PRU of data from the control card record of the job's INPUT file into the control card buffer in the control point area, starting at location W.CPAF. Since the file INPUT is on an allocatable device, one PRU will be 100 CM words long and will extend to location W.CPCAL in the control point area. If the control card record has more than one PRU, 1AJ stores the INPUT file FST entry for the next PRU of the control card record in word W.CPFST in the control point area.

The job card has been processed by the routine 1TJ; 1AJ scans the control card buffer for the card following the JOB card. When it is found, 1AJ sets a pointer in word W.CPCC of the control point area to the next control card and then processes the first card found. The processing sequence for control cards is illustrated in figure 7-1 .

When the first control card has been processed, 1AJ is loaded again. Using the next-control-card pointer, it will read that card from the buffer, delineate the next card, set the pointer to the next card, and then process the control card just read. When all cards in the buffer are processed, 1AJ reads another PRU from the INPUT file control card record into the buffer and resets the next-control-card pointer to the beginning of the buffer.

1AJ continues to process control cards until all have been processed, the job is terminated because of error, or an EXIT card is encountered.

CONTROL STATEMENT PROCESSING

When 1AJ begins to process a control statement in the control point buffer, it first tries to match the keyword with a list of names that it treats as special cases. If there is a match, 1AJ completes the processing of the statement and looks for another statement in the buffer. If the verb is not a special case name, 1AJ next checks whether the keyword is the name of a local file. If it is, the statement is cracked; that means, it is issued to the dayfile, copied to RA + 2 (RA.ARG) through RA + 54, and finally the loader is loaded to perform the load of the file. If the keyword is not a local file name, 1AJ next checks whether the keyword is a control card callable PP routine name. If it is, the parameters are separated and put in RA.ARG, the statement is issued to the dayfiles, the first two parameters are converted to binary and stored in bits 35 through 0 of the input register (except for the routine VSN) and finally the PP routine is loaded on top of 1AJ. If in turn the keyword is not a PP routine name, 1AJ sequentially searches the CM resident global library set to see if there is a control card callable entry point with the same name. If there is, and the program is an absolute overlay, 1AJ itself cracks the statement, loads the program at RA + 100 (RA.ORG), and starts it up. In any other case the loader is loaded to find something to load and execute.

The special case names recognized by 1AJ and the processing for each name are as follows:

COMMENT	The comment card, less the keyword COMMENT, is issued as a dayfile message. If the message is longer than 40 characters, it is issued in two parts.
MODE	The exit mode is set as specified on the MODE card. If the mode value is greater than 7, a control card error message is issued.
LIMIT	This card 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 card, 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 a corresponding bit to be flipped in the C.CPSSW byte of word W.SSW in the control point area.
RFL	The new field length value is set in byte C.CPNFL of word W.CPCC and the reduce bit is cleared. The new field length must not exceed the field length given on the job card or the current value in C.MFL or P.MFL.
SUMMARY	The overlay OV.6BM is loaded to issue job statistics to the dayfile.
LOGIN	Entries to dayfile are suppressed before loading this routine, bypassing the FNT search.

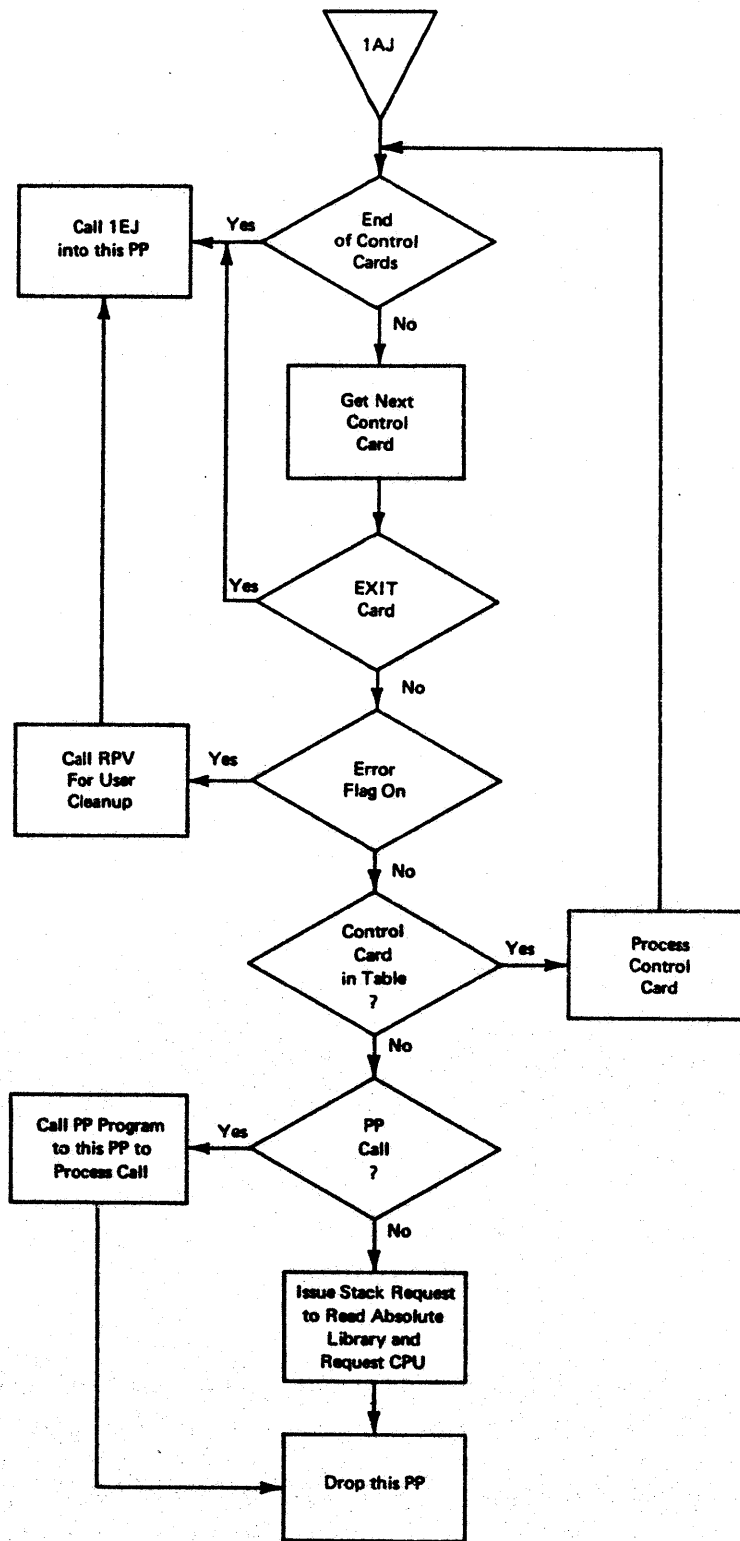


Figure 7-1. Control Card Processing Flowchart

JOB TERMINATION

NORMAL TERMINATION

Normal job termination occurs when the error flag value is zero, and all control cards for the job have been processed (an end-of-record mark for the control card record on the job INPUT file has been read) or when any form of EXIT card (except EXIT,U. or EXIT,C.) has been encountered. The end-of-job processor, 1EJ, is called by 1AJ to terminate the job as follows:

1. Dispose of all files associated with the job as follows:

Drop local files on system devices and all equipment or storage associated with them.

Attach nonzero disposition files to control point zero for further processing of output files.

Update the TAPES table; rewind and unload tapes as appropriate.

When tape scheduling is used, return reserved tapes to the system pool.

Call overlay 1PC for disposal of permanent files and local files on user device sets. 1PC will call 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, set the job name to NEXT. The clear bit will be honored by scheduler when it clears the control point area.

Files are disposed of as follows:

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 will delete the FNT/FST entry for the file. The file will still exist on a mass storage device. The APF entry for the file will be updated or deleted, as appropriate. A purged file is no longer permanent and is processed like any other mass storage file.

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 JANUS. 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, the OUTPUT file FNT is modified so that the file name is replaced by the job name, and the file is assigned to control point zero and rewound. The file is then released for processing by JANUS.

If the output file is on a non-allocatable device or pack, the equipment is dropped and the FNT is zeroed because JANUS cannot process files on non-allocatable 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 zero 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.

OTHER FILES

If the file is on a non-allocatable 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 local files residing on private device sets.

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.

PACK QUEUE

If a job attempts to MOUNT a member of a device set and the device is not on-line, the job will be swapped out or rolled out (operator initiated jobs will abort). When the device being requested is ON and FREE, all jobs waiting for the pack will be 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.

ABNORMAL TERMINATION

Abnormal termination of a job occurs when error flag value is non-zero. If the error flag value is other than KILL (value 7) or RERUN (value 10), 1EJ will dump the contents of the exchange package and the contents of memory in addition to doing the normal termination processing described above. The dump includes from 100 words before the error stop through 100 words after it (P-100 to P+100) and 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) card follows in the control card record, 1AJ will resume job processing after that card is encountered. If an EXIT(C) card is encountered prior to an EXIT, EXIT(S), or EXIT(U) card, under the conditions mentioned above, the job will terminate at that point.

If abnormal termination occurs as the result of an ABORT macro call, job processing will proceed in accordance with the ABORT call parameters as discussed in the SCOPE 3.4 Reference Manual.

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 non-zero disposition files. 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 typed 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 pre-output file is created as a local file to control point zero; it is not rewound; it has a file name which is the same as the job name. The dayfile is copied to the pre-output file, which becomes the output file for the job when it is rerun.

JOB POST-PROCESSING UTILITIES

During execution of a user program, various errors, such as arithmetic mode error or exceeding time limit, may cause the program and the job to terminate abnormally. At this point, the operating system will place a message in the dayfile, dump the exchange package and central memory (100 words on each side of the error stop location) and either resume the job processing at an EXIT or EXIT(S) card or terminate the job.

At the request of the user, the operating system will return control to the job after normal termination or after an error condition is detected and before EXIT card processing is performed. At that time, the system will return control to a user supplied post-processor routine which will print out data in appropriate format, etc. The user's post-processor routine may specify procedures to be performed after normal or abnormal job termination.

REPRIEVE

The reprieve function RPV is a PP program called by the user and by IAJ. It allows the user to receive control at a location inside the job field length after program termination, at which time a cleanup/recovery routine may be executed.

The user should call RECOVER during the job initialization phase, at which time the RPV call, generated by RECOVER as follows, is placed in RA+1:

VFD 18/3HRPV,8/2OB,12/mask,24/fwa

RPV	Is the PP routine name
2OB	Sets the CPU in auto-recall
mask	Is a bit list of error classes for recovery
fwa	Is the first word address of the recovery routine in the user job

System Error	Error Code (octal)	Reprieve Mask (octal)
Normal termination	0	100
Requested time limit exceeded	1	004
Arithmetic mode error	2	001
PP program requested abort (M.ABORT)	3	020
CP program requested abort	4	040
Attempt to call illegal PP routine	5	002
Operator dropped control point	6	010
Operator KILL	7	010
Operator initiated rerun of job	10	010
CP abort (ABT+bit 36) skips to EXIT(S) cards	11	040
ECS parity error	12	020
Required auto-recall status missing	15	002
Job hung in auto-recall	16	002
Required mass storage limit exceeded	17	004
xxx not in program library	20	002
I/O time limit exceeded	21	004

Job terminations are divided into classes: each class is assigned an octal value which may be specified singly or in combination to form the mask generated as part of the call. The values are:

0001	Arithmetic mode error
0002	RA + 1 error (PP call error, auto-recall error; PP name unknown)
0004	Time and/or storage limit exceeded
0010	Operator drop, kill, or rerun
0020	System abort (PP abort; ECS parity error; illegal PP parameter)
0040	CP abort
0100	Normal termination

The first word address points to a 17-word (decimal) block which will receive the exchange package and RA + 1 contents. Control is returned to fwa + 21.

Upon a user call to RPV, if the value of the upper 30 bits of the word at fwa is non-zero, it is assumed to be the last word address of the recovery routine in the user's job. A checksum of the area fwa + 21 to lwa will be made and the sum stored in fwa + 1.

The call to RPV by 1AJ occurs when 1AJ is entered with reprieve mask non-zero. 1AJ calls RPV with bit 41 of the input register set to one.

Upon exit from a user call to RPV, the mask and fwa fields of the call will be stored in the control point area, in word W.CPCC. The contents of the word at fwa will be cleared; and if a checksum was performed, fwa + 1 will contain the sum.

Upon normal exit from a call by 1AJ, the mask in the control point area will be cleared, the exchange package with the error flag set in register B0 will be in fwa to fwa + 17, and the contents of RA + 1 will be in the next word, fwa + 20. The system guarantees an installation defined minimum amount of time and mass storage for reprieve (and EXIT) processing. The CP time limit, IO time limit, and mass storage limit are each set to the larger of their current value or the sum of the amount used and the guaranteed minimum. No limit is allowed to be increased more than once per job.

RECOVR

The central memory program RECOVR is an interface between user program and operating system for users who wish to gain control at the end of a job step, regardless of the manner in which it terminated. RECOVR should be called during the initialization phase of the job, at which time the address of the post-processing routine is given. RECOVR formats the arguments in the call into a stack entry, checksums the user post-processor routine if requested, saves the checksum in the stack, and then calls the PP program RPV to inform the system that the job should be restarted in case an error occurs.

When the system detects an error condition, it will issue two messages to the dayfile: the first will report the error condition and the second will report that the job has been restarted only if flags set in RECOVR call match those that cause the abort. It will give control to RECOVR along with the contents of the exchange package and RA + 1 and the code for the condition that caused the system to stop program execution. RECOVR will call the post-processor program in a last in-first out order of calls in the entry stack and pass to it the address of the array area in which the exchange package, RA + 1 contents and system error code has been placed. Upon termination of the user's post-processing routine, RECOVR will make an abort request to the system so that EXIT card processing will follow, unless the post-processor has specified that the program be terminated normally.

The calling sequences for RECOVR:

FORTAN: CALL RECOVR(name, flags, checksum)

COMPASS: RECOVR name, flags, checksum

name	Name of the user routine to which control is returned after a system abort
flags	Octal value specifying condition under which control is to be returned.
	001 Arithmetic mode error
	002 RA + 1 error
	004 Time/storage limit exceeded
	010 Operator drop or rerun
	020 System abort
	040 CP abort
	100 Normal termination
checksum	Should be zero for no checksumming; otherwise, it should be the last word address+1 of the recovery program to be checksummed.

The user post-processing routine to which control is returned by RECOVR should provide for three arguments. (1) The address of a 17-word array to receive the contents of the exchange package upon abort in the first 16 words and the contents of RA + 1 in the 17th word. (2) A flag to indicate normal (ENDRUN) termination of the job or abnormal (ABORT) termination after post-processing is completed. A non-zero value indicates normal termination. (3) An address of a word to contain the contents of RA + 1. If the P register contains zero in the exchange package, the address of the program error stop will be returned in bits 30-47 of RA + 0.

Under no circumstance should the post-processor routine store into or modify the contents of the first argument.

The system error code is returned in bits 0-17 of word 1 of the first argument. The codes are explained in the REPRIEVE section.

If the user program is executed in overlay mode, both the call to RECOVR and the post-processing routine should reside in the 0,0 level overlay.

When the system stops normal execution because the job time limit has been exceeded, the time limit will be incremented by a value set per installation option so that the post-processing routine can be executed. Post-processing of an error condition should not be confused with OWNCODE (user processing of parity errors); the two concepts are not related.

JOB CONTROL WITH LOGICAL IDENTIFIERS

Logical identifiers specify logical groups or functions. They allow extended control over job initiation and job/file processing, especially when the environment is a multi-mainframe system. The ID table, resident in central memory, contains the information needed to process logical identifiers (LIDs). The ID table may have a length of zero, in which case LIDs will not be allowed by the operating system. For a single operating system, the LIDs can be used, for instance, to group all the jobs for a special printer. Those jobs must enter the system with the same logical ID. Only when the special printer is available may the operator place the logical ID in the system's ID table and allow the group of 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 multi-mainframe system consists of linked mainframes. Each linked mainframe in a multi-mainframe 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 multi-mainframe system consisting of two linked mainframes, one could have an ID table listing a host ID of MFA and a physical ID of MFB, and the other 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 multi-mainframe system is unique, because the third character of each host ID must be unique and must be a letter. As many as 58 decimal LIDs can be associated with the host mainframe.

If a single mainframe system is operating with an ID table of zero length, a job/file which does not specify a logical ID is processed on that system. Specification of a logical ID causes the system to flag the control card error and terminate processing.

If a single mainframe system is operating with an ID table containing the host ID and the list of associated logical IDs, job/files which do not specify a logical ID are processed on that system. Job/files which specify a logical ID cause the system to inspect the ID table for a matching logical ID. If a matching LID is found, the job/file is processed. If no match is found, processing waits until the operator places the specified LID in the ID table and the match is found.

If a multi-mainframe system is operating, an ID table exists for each linked mainframe. Job/files which do not specify a logical ID are processed by the host system. Job/files which specify a logical ID cause the host system to inspect its ID table. If a match is found in the list of LIDs, 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 LID. 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.

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.

Assuming the user is starting from scratch, the user obtains an unconfigured deadstart tape from Software Manufacturing and Distribution. This tape contains the binary decks of the entire SCOPE operating system, to be used to build a new system. EDITLIB is then called by selected control cards and directives to modify the current system after the appropriate assemblies and compilations have been performed. Thus a new deadstart tape and/or running system can be created which reflects the system as it is physically configured. (See figure 8-1).

It is not required that all system code 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 distinct forms: system EDITLIB, which is discussed in this section; and user EDITLIB, which is contained in the SCOPE 3.4 Reference Manual.

System EDITLIB can also function like user EDITLIB; directive statements before the first READY, or after the last COMPLETE, or between a COMPLETE and the next following READY, are interpreted as they would be in user EDITLIB.

All read and write operations required by EDITLIB are accomplished by RA+1 requests to CIO only; I/O macros or other means are not used. EDITLIB can handle object code only of the type processed by the CDC CYBER loader; 7000-type object code cannot be processed. All object code for EDITLIB input must be in SCOPE logical format, having a zero-byte terminator. Object code in W-type (word count) records such as that produced by the Record Manager, or in S, X, L, or other non-SCOPE format tapes cannot be processed by EDITLIB.

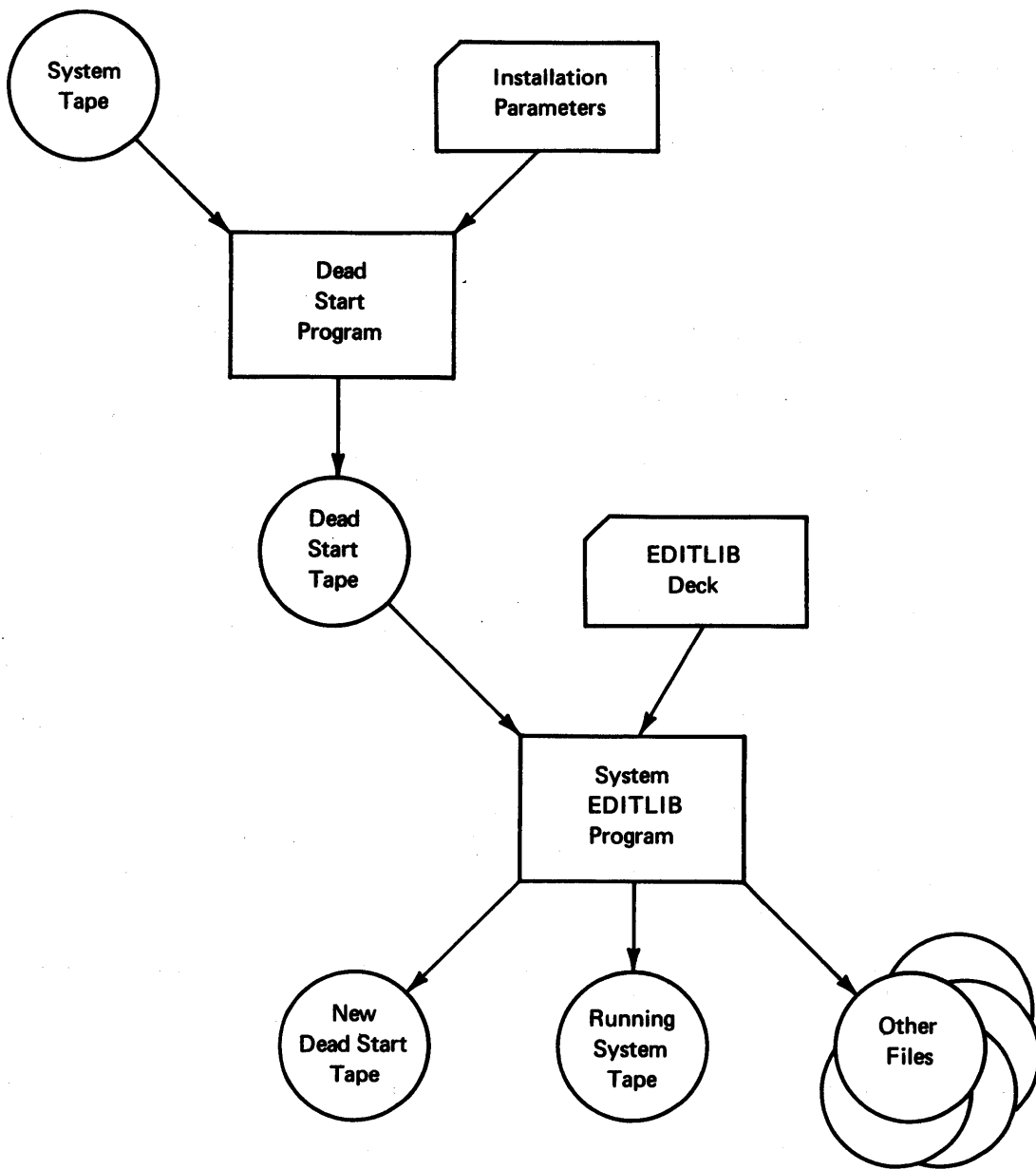


Figure 8-1. Basic Usage of System EDITLIB

CHARACTER SET

The EDITLIB character set is that which is supported by SCOPE 3.4. EDITLIB will interpret the \$ () = + - * / , and . as delimiters. Blanks are free characters and will be removed. From the character set, the user is able to create:

1. Symbol -- a single character or a string of characters
2. Vocabulary Word -- a symbol that has meaning as defined by the semantics.
3. Name -- a symbol that is defined by syntax and semantics.

When any delimiter and/or a blank is used to form a symbol, the symbol must be enclosed between \$ signs. (see Example 1). When a \$ is to be used as a character within a symbol, an additional \$ must be written immediately adjacent to each \$ desired. The only character that will be removed is the first \$ of each pair of \$ encountered. (See example 2.)

Whenever a name could be interpreted as a number or a vocabulary word, the name should be enclosed between \$ signs.

Examples: \$--DSD---\$
 Symbol is --DSD---
 \$D\$\$\$S\$\$\$\$S--\$
 Symbol is DSI\$\$\$\$--
 \$\$\$\$CIOUS
 Symbol is \$CIOUS

The input to EDITLIB is the first 72 columns of an 80 or 90 column card image. A directive must be completely contained on one card.

SYNTAX AND SEMANTICS

Presented here is the format of the various directives and the meaning associated with the symbols. 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 will set the abort flag. After EDITLIB has completed a given task, such as checking all directive formats, it will check the abort flag. When set, EDITLIB will abort. EDITLIB will continue to check a directive even if it has located an error in that directive. EDITLIB will proceed 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.

SYMBOLS USED IN SYNTAX

	The enclosed items are optional.
↓	or
()	The enclosed item is the actual symbol.
p	name of a routine
lfn	Logical 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 Central Memory.
Default	Disk

Interval in a directive will be interpreted to mean

$P_1 + P_2$ or
 $P_1 - P_2$ or
 $P_1/P_2/P_3$ or
 P_1

where P_i are single program names. $P_1/P_2/P_3$ is valid only for ADD and REPLACE directives. It is acceptable to replace either P_1 or P_2 with $[*]$, but not both. EDITLIB will ignore a directive for 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.

* This may be used to replace p in the syntax. When * is encountered, EDITLIB will consider 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.

+ This symbol is used to designate 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.

- This symbol is used to designate 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.

/ This symbol is used to delimit routine names.

Notes

1. EDITLIB will assign a number to each directive it processes.
2. EDITLIB will list each input card it receives on the file designated for listable output by the L parameter on the EDITLIB statement. The directive as interpreted by EDITLIB will appear directly below the input card. The interpreted directive will be prefaced by the number assigned to this directive. All comments about the directive will follow the interpreted directive. Since EDITLIB is a two-pass program, the results between an EDITLIB run and a LIBEDT run will vary. LIBEDT may stop if it finds an error in a directive it is executing; 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 will be generated by referencing the number that is assigned to each directive. Error messages will be issued whenever EDITLIB finds it cannot comply with the user's request. EDITLIB will also issue messages when it is unable to determine what the user is trying to accomplish.
3. The directives ADD and REPLACE will produce a CONTENT directive on the routines involved.
4. Any interval specified in a directive will be interpreted to mean:

In a reference to a file which is a library, the interval will be formed by the entries in the order they appear in the Program Name Table. The interval begins when P1 is encountered and ends when either P2 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 will be formed by the order of the routines on the file. To find P1, 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, P2 or end-of-file. The file is handled circularly only when searching for P1.

5. In a System EDITLIB, all directives that do not specifically name a particular library and are not contained between a LIBRARY and FINISH directive will refer to the PP library. This means that the PP library is the default library when no library is specified. In a user EDITLIB, no PP programs are allowed.
6. EDITLIB will not support TEXT, SEGLOAD records and/or DATA input (7000 LIBEDT). If TEXT/SEGLOAD record/DATA input is found, a message will be issued and the unrecognized records within the interval will be ignored.
7. Miscellaneous 7000 LIBEDT directives: TYPE, ERROR, PCOPY, PMOVE, MOVEC, COPYC, and DELETEC. CDC CYBER EDITLIB will ignore these and print a warning message on each directive.

SYSTEM EDITLIB

EDITLIB FILES

EDITLIB will use the following lfn:

lfn	Cataloged	Use
ZZZZZ01	yes	RESET
ZZZZZ02	yes	RESTORE
ZZZZZ03	yes	SYSTEM EXTEND
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
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 will produce the following type of files:

File Use	lfn	pfm
RESET	ZZZZZ01	ZZZZZ01
RESTORE	ZZZZZ02	ZZZZZ02
System Extension	ZZZZZ03	ZZZZZ03
SYSTEM	ZZZZZ04	ZZZZZ04
ECS Library	ZZZZZ06	
Current Directory	ZZZZZ23	ZZZZZ23

The files will be cataloged using the following keys:

File Use	ID	RP	TK	CN	MD	EX
RESET	SYSTEM	999	SSSSSSSS	XNOX	XNOX	
RESTORE	SYSTEM	999	SSSSSSST			
System Extension	SYSTEM	999	SSSSSSSU	XNOX	XNOX	
SYSTEM	SYSTEM	999	SSSSSSSV	XNOX	XNOX	XNOX
Current Directory	SYSTEM	999	DIRECTORY	XNOX	XNOX	

In SCOPE 3.4, 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 will not duplicate this effort to set up ECS files. Due to this change, a new procedure will be required to place routines into ECS at initial deadstart time.

Another task that EDITLIB performs in conjunction with deadstart is that of recovery. Deadstart will call EDITLIB to accomplish a C type recovery. In this situation, no operator intervention is required. The Deadstart Loader will set bit S.SYSEDT in byte C.DSFLAG in word P.LIB. This bit when set informs MDI that it can go ahead and change the directory without issuing the GO/DROP message. When MDI has moved the directory from the CP to CMR, it will clear S.SYSEDT and set bit S.EDTRUN in the same byte. S.EDTRUN will be set by EDITLIB whenever the directory is changed.

EDITLIB will also maintain a permanent file (ZZZZZ23) for the Deadstart Loader. This file will contain the current system directory as it appears in CMR.

The files ZZZZZ01, ZZZZZ03, ZZZZZ04, ZZZZZ06, and ZZZZZ23 will be cataloged such that by attaching them, CN (Control) and MD (Modify) permission are not granted. That is, the above permission will be granted if the correct passwords are specified, but not by just specifying the ID (Identification) and TK (Turnkey). EDITLIB will request the CN or MD options on file ZZZZZ23. When EDITLIB modifies the running system, it will do so by adding the new routines to the end of the file called ZZZZZ03. By using this method, EDITLIB will not keep others from accessing the files.

SPECIAL HANDLING OF LOCAL FILE NAME SYSTEM

In SCOPE 3.4, the system file has a lfn of ZZZZZ04. In an effort 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 lfn SYSTEM is dependent upon the context in which it is used.

If the lfn SYSTEM does not appear between the directives READY and COMPLETE, it is treated as a local file whose name just happens to be SYSTEM.

When the lfn SYSTEM does occur between a READY and COMPLETE directive, the CONTENT directive will treat it as a local file name; also ADD and REPLACE will treat it as a lfn 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,FL=1)

 Add the interval from the running system file (ZZZZZ04 or ZZZZZ03), specifically the CP library called NUCLEUS.

CONTROL CARDS

PROGRAM CALL CARD

EDITLIB(SYSTEM,t₁,t₂, . . . t_n)

where t_i can be

RESET	L=lfm
RESTORE	MSGL=n
I=lfm	ERROR=n

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. The key can not exceed nine characters and will allow restricted access to the running system directory.

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 will use the directory from the RESET permanent file to replace the directory presently in CMR. It will then change this directory as prescribed by the directives in the input file. EDITLIB will then replace the current directory with the directory it just modified. A copy of the current directory will be saved on the RESTORE permanent file before the change takes place.

RESTORE This parameter informs EDITLIB that a user wishes to replace the current system directory with the directory as it appeared before the last EDITLIB took place. Whenever the RESTORE parameter is encountered, EDITLIB will use 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 will not be saved on the RESTORE file.

I=lfm This symbol is used to designate the file containing EDITLIB directives. If it is not present, the lfm that will be used is INPUT. If INPUT is used, any binary input to EDITLIB on this file must be in the order that it will be requested. This file will not be searched. I appearing alone is equivalent to writing I=INPUT.

L=lfm This symbol is used to designate the file of listable output from EDITLIB. If it is not present, the lfm that will be used is OUTPUT. L appearing alone is equivalent to writing L=OUTPUT.

MSGL=n where n = 0 to 8. The default value is 0. This parameter determines the type of list or dayfile output generated during execution.

n	DIRECTIVE INTERPRETATION		DIRECTIVE EXECUTION	
	MESSAGE TO DAYFILE	LIST INPUT CARD	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	

ERROR=n

where n = 0 to 8. This parameter determines when an abort will occur. Each type of error has been assigned a degree of severity which will be checked against the error flag set by the ERROR parameter. This check will be performed only when the COMPLETE directive is encountered when 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
bit	 	3	2	1	0

When no ERROR=n specification is given, any error or warning will cause job termination.

See the end of this chapter for a list of all EDITLIB errors.

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)

DIRECTIVE PARAMETERS

OPTIONAL DIRECTIVE PARAMETERS

Optional parameters are not positionally dependent.

AL=l where l = 0 to 7777B. Default = 0. AL stands for access level and is a parameter of the ADD and REPLACE directives. 11 of the 12 bits are meaningful to INTERCOM only. The right most bit of this field is used by the system to regulate access to program through control card requests. When this bit is set to 0, the routine or entry cannot be called from a control card. 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 3 sections. The first section is one bit in size and is the right most 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) will be used as permission bits. When a user requests the use of a command, his 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 where k = 0-377777. Default = 0. FL is a parameter of the ADD and REPLACE directives and stands for Field Length. It is meaningful for CM programs only. SCOPE uses the value in allocating memory when the program is called. This field length must taken into account the CM needed by the loader and all programs, overlays and segments that could be loaded by the specified program.

FLO=m where m = 0 or 1. Default = 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 may override the FL value set in the library.

In allocating memory for the execution of a library resident program, SCOPE must determine which of the following execution field lengths is to be used:

Field length (PFL) as specified by the FL parameter; or nominal field length (NFL) which can be the last RFL request, or the FL specified on the job card, or the default value IP.SFL. However, the field length allocated cannot exceed the job card field length. In determining which field length to allocate, SCOPE uses the FLO flag and the automatic field length flag (Reduce bit). The conditions and results are as follows:

When PFL is zero, use NFL

When PFL is not zero and FLO is zero, use PFL.

When PFL is not zero and FLO is one and reduce bit is off, use the NFL

When PFL is not zero and FLO is one and reduce bit is on, use the PFL

Since the maximum field length that will be allocated is NFL, any job that does not have a sufficient field length in which to be loaded for execution will be terminated abnormally.

LIB=pn where pn is the name of a library. This is a parameter which 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. This parameter specifies what library on the deadstart file will 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 will contain 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.

†Either NEW or OLD is mandatory, not both.

DIRECTIVE RESTRICTIONS

The following table indicates where directives must occur relative to LIBRARY-FINISH and READY-COMplete sequences. Any violations of these restrictions will be diagnosed by EDITLIB before any directives are executed and will cause an abort accompanied by diagnostic error messages. (See SCOPE 3.4 Diagnostic Handbook.)

Directive	Restriction
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
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

DIRECTIVES

ADD(Interval,lfn[,r][,(AL =)l][,(FL =)k][,(LIB =)pn][,(FLO =)m])

This directive adds the specified program(s) from the lfn to the library under construction or modification. If the lfn contains a library, the Program Name Table of the library will be searched to locate the specified program(s) to be added. If the file is not a library, the search for the specified program(s) will begin at the current record position on the file and proceed to EOF. If the program(s) are not found, the file will be rewound and the search will continue until all programs have been searched. If the program(s) are not found in the library or on the file, an error message will be issued. If the program(s) to be added already exist(s) in the library, an error message will be issued. It is not the function of an ADD directive to replace the current program with the new program. This function is accomplished by a REPLACE directive.

Example: ADD(HIGH,LOW,ECS,AL = 70,FL = 1000)

 ADD(HIGH,LOW,FLO = 1)

 ADD(HIGH,SYSTEM,LIB = SCOPE,ECS,AL = 123)

CHANGE(pn[,r])

This directive is used to change the residence of the library tables in the LNT. If the library already has the specified residency or if the library is not part of the system, EDITLIB will issue an appropriate comment.

Example: CHANGE(FNT,CM)

COMPLETE.

This directive informs EDITLIB that the directive list for the directory whose name appeared in the READY directive has been exhausted. Simply, it means that there are no more modifications or additions to be performed on the directory. This directive must have been preceded by a READY directive.

CONTENT(Interval,lfn)

The CONTENT directive is used to obtain information about a program or series of programs on a specified file (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 card. In a multi-file situation, CONTENT must be specified for each file.

Example: CONTENT(ONE,FILE1)

DELETE(Interval)

This removes all entries from the library tables that refer to the specified programs.

Example: DELETE(FEAR)

 DELETE(FEAR + HATE)

ENDRUN.

This directive indicates that execution of directives should stop. All directives following ENDRUN will be checked, but not processed. This is an optional directive; if it is not found, one will be generated. If this directive is found and EDITLIB finds an error in a directive after the ENDRUN card, it will be flagged but will not stop EDITLIB from executing directives which precede ENDRUN.

FINISH.

This directive designates the end of the list of directives that will affect the library defined by the last LIBRARY directive. Each LIBRARY directive must have a FINISH directive. It is permitted to have multiple (LIBRARY, FINISH) sequences within a single directive sequence.

INCLUDE(pn,lfn[,r])

This directive will add the library which is on the specified file to the directory. This allows a user to make his user library a part of the running system.†

Example: INCLUDE(SMOG,AIR,ECS)

 INCLUDE(SMOKE,LUNGS)

INCLUDEP(lfn)

This directive allows an old PP program library to be added to the current directory model. The PP program library in the current model must be empty when the directive is encountered. Residency and access levels of each entry will be that of the entries in the source PP program library.†

LIBRARY(pn,(OLD) ↓ (NEW)(,r))

The LIBRARY directive during a system EDITLIB run performs two functions. When it is encountered outside of a READY and COMPLETE directive, the action taken is equivalent to a user's EDITLIB. When this is the 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 it is between a READY and COMPLETE directive, the action is also equivalent to that of a user's EDITLIB, but the LIBRARY will be found using the Library Name Table (OLD) or will be added to the Library Name Table (NEW).

Example: LIBRARY(FNT,OLD)

 LIBRARY(RUN,NEW,ECS)

 LIBRARY(COBOL,NEW)

†When creating a new deadstart tape from the running system, programs that were ECS resident will become disk resident.

LISTLIB(Interval,lfn,[pn])

The LISTLIB directive is used to list a library which can be a user's library, running system, new system and/or deadstart file.

Example: **LISTLIB(COPY,SYSTEM,SCOPE)**

When between READY and COMPLETE, the lfn that the list will be performed on is the running system and/or the new system under construction. When this directive 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 will send information about the specified program(s) on the lfn or on the lfn and in the pn to the output file. The information will come from the object deck being processed.

Example: **LISTLIB(COPY,FILE1,SCOPE)**

LISTLIB(1AJ,SYSTEM)

LISTLNT.

The LISTLNT directive must appear between a READY and a COMPLETE directive. It will list 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 will not list the LNT on an old Deadstart file. There are no parameters.

MOVE(Interval,r)

This directive will change the residence of CP and PP programs within a system directory. EDITLIB will move PP programs to DS, CM, or ECS. Movement of CP programs depends on the residence of the system library of which the CP program is a part. If CM resident, movement is unrestricted; but for DS and ECS resident libraries, the new residence can be specified only as DS or ECS (the library directory residence must be at least as fast as the new residence).

Example: **MOVE(STOCKS,DS)**

MOVE(BONDS+CASH,CM)

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 truly is resident in ECS.

RANTOSEQ(lfn1,lfn2)

This directive will place a random user library on a sequential file in library format. lfn1 is the source file name; lfn2 is the destination file name. All records not referenced by the PNT are lost.

READY(lfn[, (OLD) | (NEW)])

This is the first directive of any directive list pertaining to the lfn specified. Only those directives between a READY and a COMPLETE will affect the lfn. The READY directive is used in conjunction with a system EDITLIB run. The READY directive specifies that the operation will be performed on a directory. The lfn is either the running system or the name of the file which will contain the new system. If the name SYSTEM appears as the lfn, it will mean 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); 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)

REMOVE(pn)

This directive removes the library from the running system. The library's entry in the Library Name Table is removed.

Example: REMOVE(FTN)

REPLACE(Interval,lfn[,r] [(AL=)l] [(FL=)k] [(LIB=)pn] [,FLO=)m])

The REPLACE directive in a system EDITLIB will retain the r, AL, FL, and FLO values of the replaced routine if not otherwise specified.

Example: REPLACE(IBM,CDC,AL=70)

REPLACE(XDS-RCS,NCR,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.

REWIND(lfn[/lfn] . . .)

The result of this directive will be the positioning of the specified file at the first record on the file. To accomplish this, EDITLIB will issue a CIO request to do a REWIND. In system mode, the file name SYSTEM refers to the running system.

SEQTORAN(lfn1,lfn2)

This directive will place a sequential user library on a random file in library format. lfn1 is the source file name; lfn2 is the destination file name.

SETAL(Interval,I)

where I = 0 to 7777 (octal). This directive permits the user to change the access level of a program already in the directory.

Example: SETAL(IBM+NCR,7770)

 SETAL(CDC,1)

SETFL(Interval,k)

where k = 0 to 377777 (octal). This directive permits the user to change the field length required to run the specified program already in the directory.

Example: SETFL(BYTE,100000)

 SETFL(BIT + WORD,100000)

SETFLO(Interval,k)

where k = 0 or 1. This directive permits the user to change the field length override bit of programs already in the library. The default value is 0 (override is not permitted).

Example: SETFLO(BIT11,0)

 SETFLO(BYTE-WORD,1)

SKIPF/SKIPB(N,lfn,[F] ↓ [P])

This is the skip-forward and skip backward directive. The meaning of the parameters is as follows: If N is numeric, it represents a number of records to be skipped unless the third parameter is present; then N indicates a number of files. If N is alphanumeric, then 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 will be positioned before the record of that name (N). If N is not found, 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.

TRANSFER(Interval ↓ n,lfn)

This directive is used to create deadstart tapes. It will move records whose prefix table has been removed or is to be removed. A check will be 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; e.g. program name is 026, the directive will be TRANSFER(\$026\$,SYSTEM).

Example: TRANSFER(*,SYSTEM)

 TRANSFER(10,SYSTEM)

***/comment**

This symbol is used to denote a comment field; whenever encountered in columns one and two, EDITLIB treats it as a comment and will not check the card.

Example: ***/THIS ADDS A PROGRAM THAT WILL DO A/B*C**

LIBRARY DIRECTORY ACCESS

PP ROUTINES

When a PP routine wishes to access the library directory, it will first check bit 59 of P.LIB. When bit 59 is set to 1, EDITLIB 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

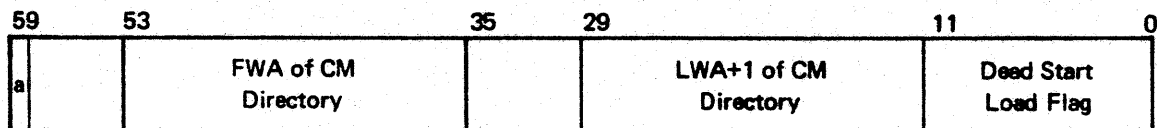
All CP routines accessing the directory should set bit S.CPLDAF in byte C.CPFLAG in word W.CPFLAG of their control point area. To access the directory the following procedure should be followed. First check bit 59 in P.LIB; if not set, then set bit S.CPLDAF. Then check bit 59 in P.LIB again. If it is 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

When EDITLIB wishes to access the directory, it will issue a monitor request to have the PP routine MDI assigned to its control point. By passing MDI the appropriate code, MDI will try to change the directory. The procedure is as follows:

MDI will set 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 will go into the event stack until the bit is cleared. After MDI has completed its task, it will adjust the FWA and the LWA + 1 pointers in P.LIB and set bit 59 to 0.

LIBRARY DIRECTORY POINTER FORMAT

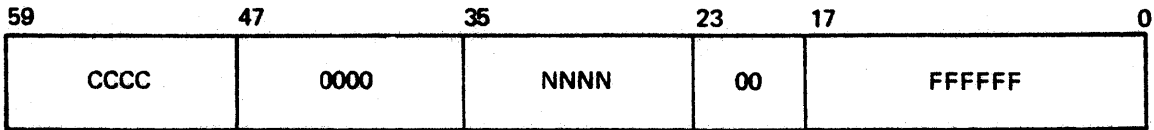


a = Library Change Flag

P.LIB

The use of the first word has been expanded to hold a Library Change Flag (1 bit). This flag was added to increase system integrity and efficiency. The Library Change Flag tells the PP and any CP program using the directory that an EDITLIB run wishes to change the directory or is changing the directory. Only when the flag in the control point areas is zero will 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 will set this word to zero and wait one second before changing the directory.

FILES

SYSTEM FILES

A copy of all the routines referenced by the directory will reside on a permanent file called ZZZZZ04 after initial deadstart. If a program or library subsequently is to be included into the directory, it will be placed in the permanent file called ZZZZZ03. EDITLIB will never write, modify or extend the permanent file ZZZZZ04. The new version of replaced routines will reside on the file ZZZZZ03. Since EDITLIB will not alter any record currently on the file ZZZZZ04 or the file ZZZZZ03, routines may be read from either file while EDITLIB is adding new records to ZZZZZ03.

USER FILES

The user is responsible for the contents of his library. EDITLIB will not overwrite any record on the file. It will only add records at EOF/EOI. This means that the user will have unused records on his file when he deletes or replaces programs on his library. It is up to the user to remove this dead space by building a new library using the old one as a source. EDITLIB will issue no permanent file request. The user must attach, catalog and extend his file. 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.

GENERAL

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: <ol style="list-style-type: none">1. ***LIBRARY2. 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 EOF	Routines

SYSTEM SECURITY

EDITLIB will use the reprieve function to protect the system files and its scratch files. Under any normal or abnormal termination condition, EDITLIB will have control turned over to a program that will return or release 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 will perform the following tasks:

1. Request field length for directory in CMR.
2. Set directory change flag.
3. Get length of directory.
4. Place CM address into LNT for CM resident libraries.
5. Copy directory to control point.
6. Copy directory to CMR.
7. Check CP activity flags at each control point.
8. Write current directory file.
9. Attach FNT to ZZZZZ06 to EDITLIB control point.
10. Monitor EDITLIB error flag after GO.
11. Catalog ZZZZZ01 and ZZZZZ02 at control point zero with an ID of SYSTEM.

A parameter list will be provided in EDITLIB for MDI. This list will contain MDI request code and all necessary information. MDI will return 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 will issue the appropriate message to the operator when any of the following EDITLIB directives have been 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.

TABLE FORMATS

CMR DIRECTORY

59	41	35	23	17	0
T.LIB					LWA+1 of LNT
Library Name Table (LNT) (5 Words per Entry)					
	LWA+1 of PP Program Bodies			LWA+1 of PPNT	
PP Program Name Table (PPNT) (2 Words per Program)					
PP Program Bodies					
CM Resident Libraries					

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 external references.

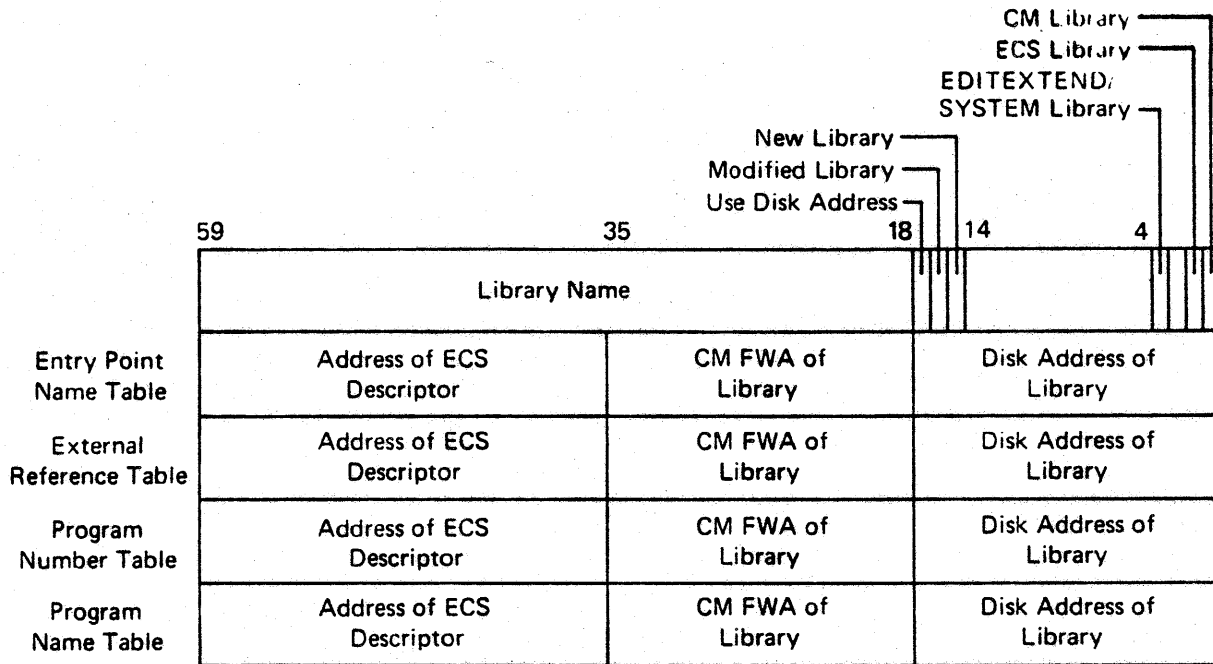
A particular program in the library can contain a maximum of 250 entry points and 500 external references. The PP program name table allows a maximum of 450 PP programs.

CM RESIDENT LIBRARY FORMAT

59	47	35	17	11	0
Length of PNT	Length of PNUT	Length of ERT	0	Length of EPNT	
Library Entry Point Name Table (EPNT) (1 Word per Entry Point; 2047 entries maximum)					
Library External Reference Table (ERT) (1 Byte per External Reference; 2047 entries maximum, plus 2047 entries in a table extension)					
Library Program Number Table (PNUT) (1 Byte per Entry Point; 2047 entries maximum)					
Library Program Name Table (PNT) (2 Words per Program; 2047 entries maximum)					
Library Program Bodies					

All information that is a part of the system directory and resides in ECS will be on the ECS file called ZZZZZ06. This is a special case system file.

LIBRARY NAME TABLE ENTRY



Word 1:

Bit 0 0 Library is not CM resident.

 1 Library is CM resident.

Bit 1 0 Library is not ECS resident.

 1 Library is ECS resident.

If bits 0 and 1 are both zero, the directory is on disk.

Bit 3 0 Library on file ZZZZZ04

 1 Library on file ZZZZZ03

Bit 15 Library is new.

Bit 16 Library is modified.

Bit 17 0 Residency addresses are valid.

 1 Use disk address only.

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 that is 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.

Bits 18-59 The library name; can contain up to 7 alphabetic and/or numeric characters of which the first must be alphabetic.

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.

Example:

Bits

2	1	0
0	0	0
0	0	1
0	1	0

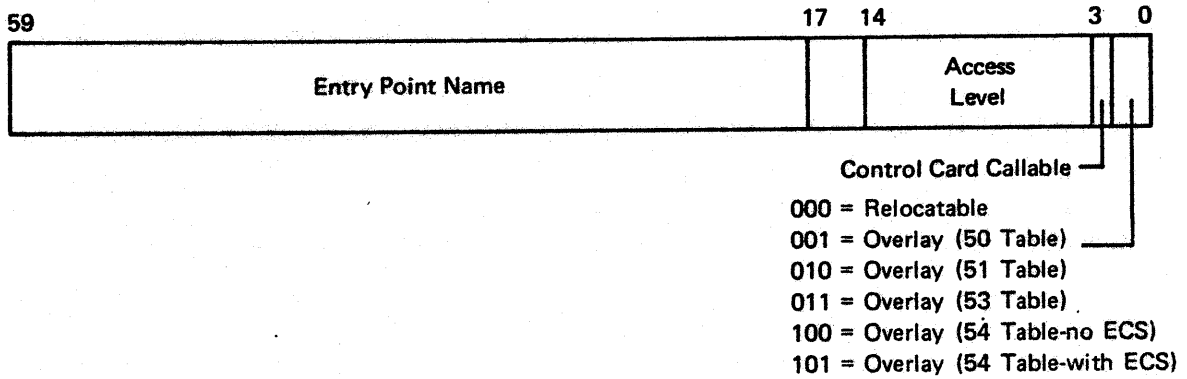
Library is disk resident.

Library is CM resident.

Library is ECS resident.

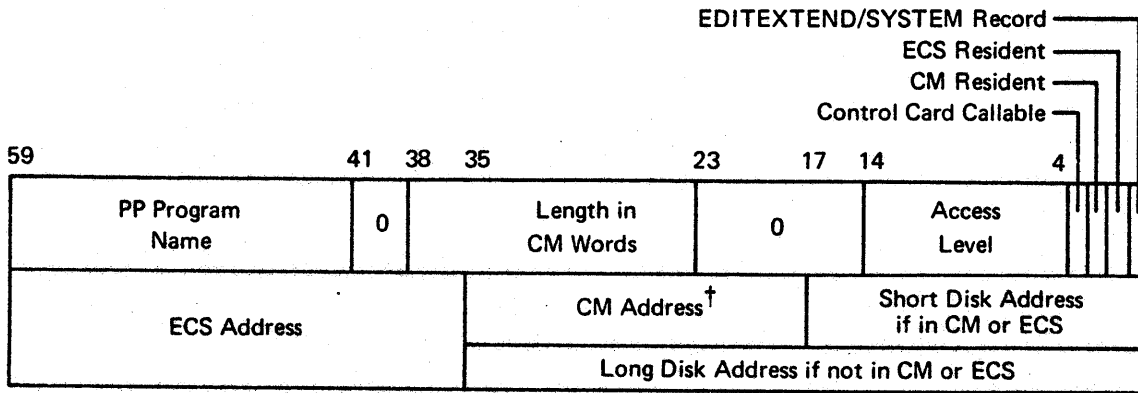
Note: Fields not being used contain zero.

EPNT ENTRY FORMAT



- Bits 0-2
- 000 The entry point belongs to a relocatable program
 - 001 The entry point belongs to an absolute overlay which begins with a 50 table.
 - 010 The entry point belongs to an absolute overlay which begins with a 51 table.
 - 011 The entry point belongs to an absolute overlay which begins with a 53 table.
 - 100 The entry point belongs to an absolute overlay which begins with a 54 table and has no ECS image.
 - 101 The entry point belongs to an absolute overlay which begins with a 54 table and has an ECS image.
- Bit 3
- 0 This entry point cannot be called from a control card.
 - 1 This entry point can be called from a control card.
- Bits 4-14
- These are the Access Level bits for INTERCOM.
- Bits 18-59
- 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.

PP PROGRAM NAME TABLE ENTRY



Word 1

- Bit 0
 - 0 If the record is on (pfn) ZZZZ04
 - 1 If the record is on (pfn) ZZZZ03.
- Bit 1
 - 0 Record/routine is not ECS resident.
 - 1 Record/routine is ECS resident.
- Bit 2
 - 0 Record/routine is not CM resident.
 - 1 Record/routine is CM resident.
- Bit 3
 - 0 This program cannot be called from a control card.
 - 1 This program can be called from a control card.
- Bits 4-14 Access Level bits for INTERCOM.

Word 2

If the routine is disk resident, there will be a special disk address in bits 0-35 for use by PP resident.

If the routine is central memory or ECS resident, bits 17-0 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

59	47	35	23	11	0
Entry Point Number + 1	Entry Point Number + 1	Entry Point Number + 1	Entry Point Number + 1	Entry Point Number + 1 or Continuation	

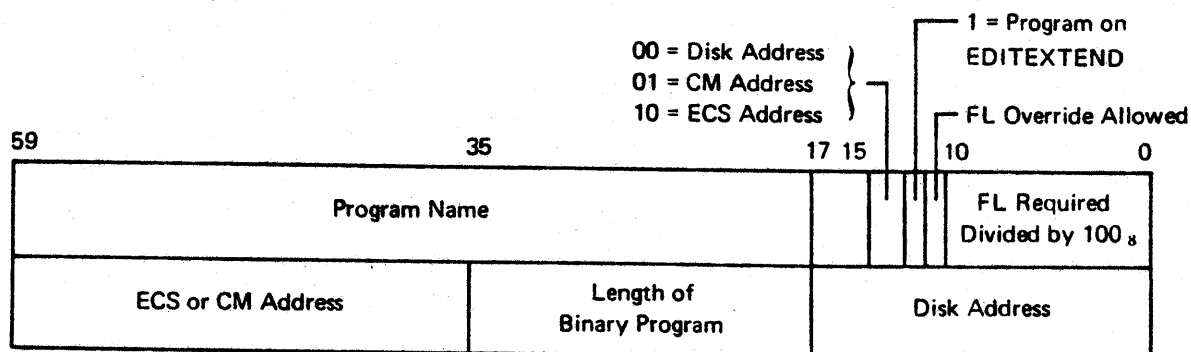
The first entry of this table is associated with the first entry in the Entry Point Name Table. The chain of External References is terminated by 12 bits of zero or by the 5th parcel not having bit 11 set to 1. If bit 11 is set to 1, the value contained in this parcel, less bit 11, is the relative address 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 will be shown in this table.

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	

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 5 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 programs's entry in the Program Name Table. Address is relative to the start of the Program Name Table.

PNT ENTRY FORMAT



Word 1:

Bits 0-10 Contain the amount of field length divided by 100(octal) to execute this program. If not specified during an EDITLIB run, it will be set to DFAULTFL/100(octal) where DFAULTFL is a program parameter.

Bit 11 INTERCOM field length override bit:

- 0 do not override
- 1 may be overridden

Bit 12 1 Program is on (pfn) ZZZZ03.

0 Program is on (pfn) ZZZZ04.

Bits 13-14 00 Disk address

01 CM address

10 ECS address

Bits 18-59 Program Name in display code.

The CM address is relative to the LWA + 1 of this table. Fields not being used contain zero.

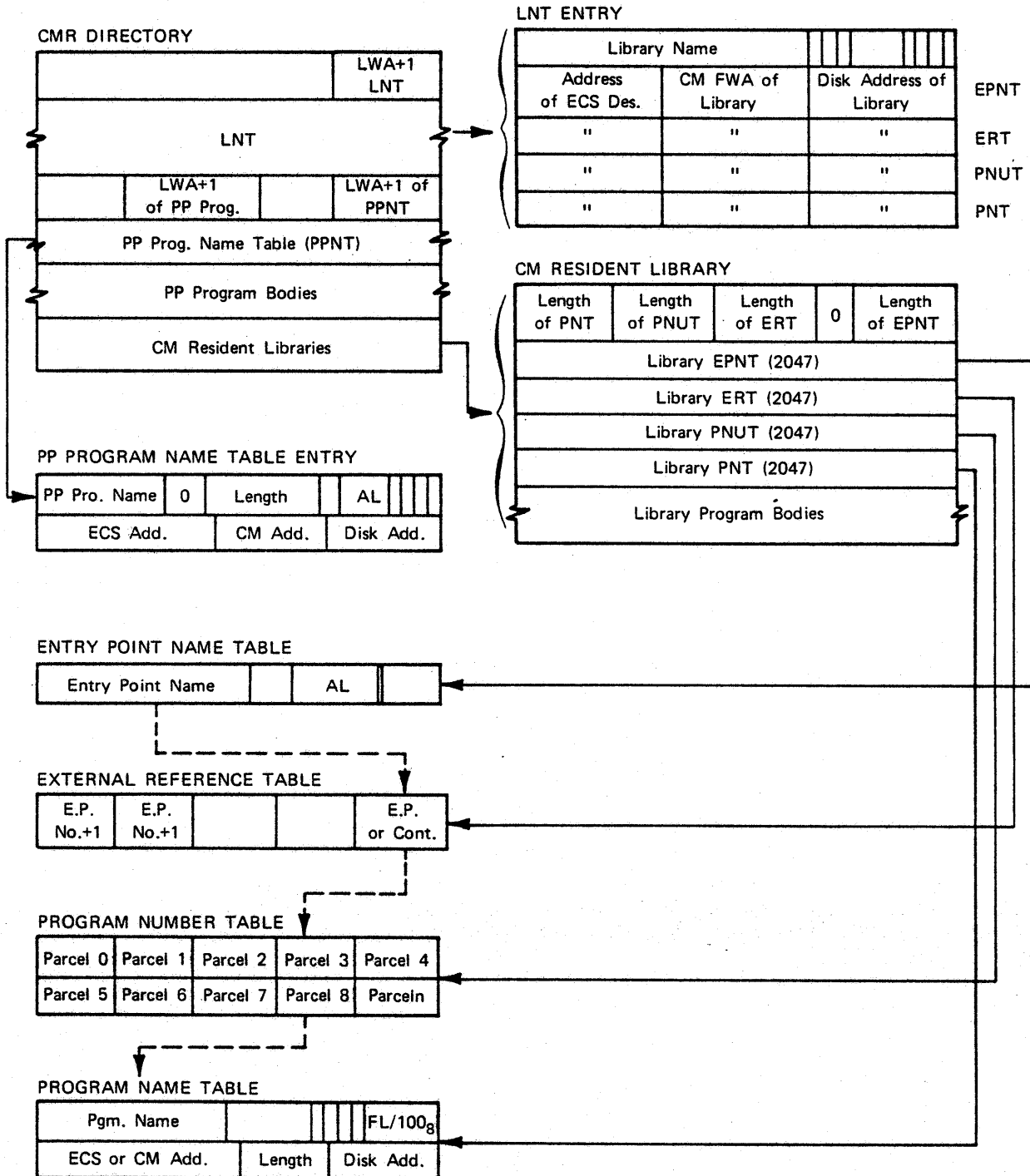
ENTRY POINT AND EXTERNAL REFERENCE LIST FORMAT

	Program Number
Entry Point	
Word of Zeros	
External Reference	
Word of Zeros	

The Entry Point and External Reference List has an entry for each program that is a part of the library. This list will be maintained by EDITLIB. The entry point and external references for each routine will be in alphabetical order and will be left justified with zero fill.

The index to this record, when the library is not part of the running system, will be 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 will be placed following the Entry Point Name Table record. By using the index to the Entry Point Table it will be possible to access the External Reference List. A CIO Read Skip request will be used with the index to the Entry Point Name Table. Then a read with a zero index will be used to obtain the Entry Point and External Reference List.

LIBRARY TABLE INTERFACES



EXAMPLES

Deadstart Creation

```
READY(NEWSYS,NEW)
TRANSFER(17,SYSTEM)
TRANSFER(CMR,NEWCMR)
SKIPF(1,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(ISP,LGO,AL=0,CM)
LISTLIB(*,SYSTEM)
COMPLETE.
LISTLIB(*,SYSTEM,NUCLEUS)
LISTLIB(*,SYSTEM,FORTRAN)
ENDRUN.
```

Deadstart Creation From The Running System

```
READY(NEWSYS).
REWIND(NEWSYS/SYSTEM).
TRANSFER(*,SYSTEM).
INCLUDE(SYSTEM).
INCLUDE(NUCLEUS,SYSTEM,CM).
INCLUDE(SYSOVL,SYSTEM,CM).
INCLUDE(SYSIO,SYSTEM).
```

- . INCLUDE directive for all other libraries that are to
- . be a part of the new system.

```
INCLUDE(SYSMISC,SYSTEM).
```

- . Directives that will modify any of the libraries
- . just included into the new system or directives
- . that will create new libraries.

```
COMPLETE.
ENDRUN.
```

System EDITLIB

```
RFADY(SYSTEM,OLD)
ADD(*,NEWPP)
LIBRARY(SCOPE1,NEW)
ADD(*,SYSTEM,LIB=NUCLEUS,DS,AL=1,FL=40000,FLO=1)
RELPACE(LOADER,NEW,DS,AL=1,FL=20000)
FINISH.
LISTLIB(*,SYSTEM,NUCLEUS)
COMPLETE.
READY(SYSTEM,OLD)
ADD(1ZZ,LGO,AL=0)
LIBRARY(NUCLEUS,OLD)
DELETE(HOHUM)
FINISH.
COMPLETE
ENDRUN.
```

EDITLIB error messages are explained in the SCOPE 3.4 Diagnostic Handbook.

INTERVAL CAN NOT BE SATISFIED – PP PROGRAM LIBRARY IS FULL. PROGRAM MAX. REACHED.
ADJUST MAXLPNT IN EDITLIB – INSTALL NEW VER.

INTERVAL CAN NOT BE SATISFIED – PP PROGRAM LIBRARY IS FULL. TABLE OVERFLOW ERROR
– EDITLIB REQUIRES MORE FL.

LIBRARY MUST BE ON A RANDOM FILE IF OLD PARAMETER SPECIFIED.

ONLY LFN SYSTEM CAN BE MODIFIED.

TABLE OVERFLOW ERROR – EDITLIB REQUIRES MORE FL.

TABLE OVERFLOW WHILE PERFORMING A DELETE. ENLARGE BUF3. BUF. SIZE ASSEMBLED INTO
EDITLIB. INSTALL NEW VERSION.

XREF LIST OVERFLOW – INCREASE THE SIZE OF BUF5. BUF. SIZE ASSEMBLED INTO EDITLIB.
INSTALL NEW VERSION.

xxxxxx LIBRARY CONTAINS THE MAXIMUM NO. OF PROGRAMS.

xxxxxx LIBRARY IS ALREADY IN THE LIBRARY NAME TABLE.

xxxxxx LIBRARY NOT ADDED TO LIBRARY NAME TABLE. PROGRAM MAX. REACHED. ADJUST
MAXLPNT IN EDITLIB – INSTALL NEW VER.

xxxxxx LIBRARY NOT ADDED TO LIBRARY NAME TABLE. TABLE OVERFLOW ERROR – EDITLIB
REQUIRES MORE FL.

Class – CRITICAL

BUF3 UNABLE TO HOLD PROGRAM NAME TABLE. ENLARGE BUF3. BUF. SIZE ASSEMBLED INTO
EDITLIB. INSTALL NEW VERSION.

CIO IS NOT A MEMBER OF THE PP LIBRARY.

COMPLETE ENCOUNTERED BEFORE DSD WAS TRANSFERRED.

INCLUDEP INVALID IF PPNT IS NOT EMPTY.

PARTIAL EDITLIB PERFORMED – SYSTEM MAY BE INOPERATIVE.

PP PGM NAME TABLE IS EMPTY.

PPLIB IS BOTH SOURCE AND DESTINATION LIBRARY.

Class - SERIOUS

BUF3 UNABLE TO HOLD LIBRARY NAME TABLE. ENLARGE BUF3.

DSD WAS TRANSFERRED BEFORE INTERVAL OR COUNT SATISFIED.

EOF/EOI ENCOUNTERED BEFORE ALL THE RECORDS WERE TRANSFERRED.

*** ERROR DUPLICATE ENTRY POINT NAME IN PGM NAME xxxxxx.
ENTRY POINT NAME(S)

LOCAL FILE xxxxxx CONTAINS AN INCOMPLETE LIBRARY.

OBJECT DECK CONTAINS ONLY A PREFIX TABLE.

PREFIX TABLE MISSING OR RECORD IS NOT AN OBJECT DECK.

RELOCATABLE PGM NAME xxxxxx DOES NOT HAVE AN ENTRY POINT.

THE LFN SPECIFIED IS NOT A DEADSTART TAPE.

UNRECOGNIZABLE TABLE FOLLOWING A PREFIX TABLE.

USER LIBRARY COULD NOT BE FOUND ON FILE xxxxxx.

ZERO WORD COUNT IN A LOADER TABLE.

xxxxxx IS NOT A RANDOM FILE.

xxxxxx IS NOT A SEQUENTIAL FILE.

xxxxxx IS NOT IN THE SOURCE LIBRARY - NO PROGRAMS ADDED.

xxxxxx LIBRARY IS ALREADY IN THE LIBRARY NAME TABLE.

Class -- MINOR

CP PROGRAMS MAY NOT RESIDE IN THE PP LIBRARY.

FILE NAME xxxxxx CONTAINS AN INCOMPLETE BINARY RECORD.

INTERVAL CAN NOT BE SATISFIED -- UNABLE TO LOCATE FIRST PGM.

INTERVAL CAN NOT BE SATISFIED -- UNABLE TO LOCATE LAST PGM.

INTERVAL IMPROPERLY FORMED -- xxxxxx PRECEDES xxxxxx.

LOCAL FILE xxxxxx IS NOT A DS TAPE OR USER LIBRARY.

PP PROGRAMS NOT IN PPLIB CANNOT BE ACCESSED BY PP RESIDENT.

PP PROGRAM xxxxxx ALREADY IN PP LIBRARY.

RESIDENCY PARAMETER -- ECS -- INVALID DURING DEADSTART CREATION.

UNABLE TO LOCATE PGM xxxxxx ON LOCAL FILE NAME xxxxxx.

USER LIB. xxxxxx NOW CONTAINS AN EMPTY PNT. FINISH DIRECTIVE NOT PROCESSED -- EDITLIB WILL CONTINUE.

xxxxxx ALREADY IN xxxxxx LIBRARY.

xxxxxx IS NOT THE NAME OF THE NEXT RECORD ON FILE INPUT.

xxxxxx LIBRARY IS EMPTY. LIBRARY WILL BE REMOVED.

xxxxxx LIBRARY IS NOT A MEMBER OF THE LIBRARY NAME TABLE.

xxxxxx LIBRARY IS NOT A MEMBER OF THE LIBRARY NAME TABLE. LIBRARY PARAMETER IS NOW NEW.

xxxxxx LIBRARY IS NOT INCLUDED IN THE xxxxxx FILE.

xxxxxx RESIDES IN ECS SINCE xxxxxx LIBRARY NOT IN CM.

xxxxxx RESIDES ON DISK SINCE LIBRARY xxxxxx NOT IN CM.

FIRST LIBRARY NOT CM RESIDENT, 1AJ CAN-T ACCESS

Class . WARNING

LAST DEADSTART RECORD ALREADY TRANSFERRED. DIRECTIVE IGNORED.

LISTLIB IS VALID FOR NEWSYS/RUNSYS BETWEEN READY - COMPLETE.

LNT IS EMPTY.

TRANSFER DIRECTIVE VALID ONLY IN DEADSTART CREATION MODE.

Under SCOPE 3.4, 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 card SYSBULL. BULLUP has the capability to (1) create a system bulletin file if none exists, (2) update an existing bulletin file, and (3) reduce the size of an existing bulletin file. When BULLUP creates or updates a bulletin file, it also builds an index so that bulletins can be accessed by name. Updating the bulletin file consists of adding new bulletins to the file and/or updating the information in an existing bulletin. Reducing the size of the bulletin file consists of reading the active information from the bulletin file and writing it to a new file.

SYSTEM BULLETIN FILE (BULLUP CARD)

The control card for calling BULLUP is as follows:

BULLUP.

or

BULLUP(input,output).

Default input and output files are INPUT and OUTPUT. Alternate files may be used by specifying the alternate file names. SYSBULL does not have this capability; output from SYSBULL always goes to the file OUTPUT.

BULLUP DATA CARDS

BULLUP accepts data in card format. Data cards for BULLUP are as follows:

Name card	first bulletin
Contents card	
.	
.	
.	
Contents card	
Name card	second bulletin
Contents card	
.	
.	
.	

7/8/9

Name cards have the following format:

Column	1	2	3 through 9	36 through 80
Entry	*	\$	bulname	bulletin description

The bulletin name, bulname, may be 1 to 7 letters and digits, and must begin in column 3. The first blank terminates the field. Any combination of characters is allowed except the reserved names INDEX and ALL. Up to 100 names may be specified for the system bulletin file.

Bulletin description, which may contain up to 45 characters, is generally a brief description of the bulletin. The bulletin description, which is listed with the index, is the second line of the bulletin.

Contents cards are any that do not have * \$ in columns 1 and 2. Column 1 is used either for printer carriage control or as a continuation card indicator. If card column 1 is not a comma, it is considered to be a printer carriage control character, and card columns 2 thru 80 are to be printed in print line positions 1 through 79. A card with a comma in column 1 is considered to be a continuation card, and card columns 2 through 58 are to be printed in print line positions 80 through 136. Multiple continuation cards are allowed.

PROCESSING DATA CARDS

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 cards for a creation run are processed in the same way as for an update run.

A name card is read and validated. If an error occurs in the name card, succeeding bulletin contents cards are skipped up to the next name card; and a diagnostic is printed. If the name card 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 card, contents card, 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 card is encountered. After all data cards 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. Control cards are required to create a system bulletin file:

```
jobcard
REQUEST(ZZZZZIN,*PF)
BULLUP.
CATALOG(ZZZZZIN,SYSTEMBULLETINFILE,ID=SYSBUL,...)
7/8/9
Name card
Contents card
.
.
.
6/7/8/9
```

The permanent file name and the ID of the system bulletin file may be changed. If changed, the attach function in SYSBULL must also be changed. Only the permanent file name and the ID are defined in SYSBULL. 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 may be specified if desired, without any changes.

UPDATING A SYSTEM BULLETIN FILE

BULLUP updates a bulletin file by adding new bulletins and/or by 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.

Control cards are required to update a system bulletin file:

```
jobcard
ATTACH(ZZZZZIN,SYSTEMBULLETINFILE,ID=SYSBULL,...)
BULLUP.
EXTEND(ZZZZZIN)
7/8/9
Name card
Contents card
.
.
.
6/7/8/9
```

Permanent file considerations for an update run are the same as for a creation run. Data card requirements are the same for updating an existing bulletin, as for creating a new bulletin. The old contents of the bulletin are not carried forward to the updated bulletin and, if desired, must be re-entered via contents cards. Information is deleted from an existing bulletin by using a name card and omitting succeeding contents cards. Since disk space for this bulletin still exists, even though it contains no information, a single contents card 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 use re-write 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 may then be cataloged. The following control cards are required to reduce the size of the system bulletin file:

```
jobcard
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 cards 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.

SCOPE/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 simply continues. However, installations with INTERCOM users may wish to keep their users informed via these named bulletins, since INTERCOM automatically initiates an attempt to find them.

Under SCOPE 3.4, 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 simply continues. However, installations may wish to keep their users informed via BATCH since SCOPE automatically attempts to find the BATCH bulletin.

INTRODUCTION

The purpose of segmenting central memory resident (CMR) is to move operating system central processor code from central memory to extended core storage (ECS) in systems having ECS. The code is loaded back to central memory and executed as needed. This reduces the amount of central memory required for the operating system and releases that central memory 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 may 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 may 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 CDC CYBER Loader to reload LDCMR; this program must be a relocatable binary. 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 zero ECS FL; therefore, the length of the direct access area will change 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 2.B 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 lockout EDITLIB operations. On completion of the move, ZZZZZ23, ZZZZZ01, and ZZZZZ02 are modified to reflect the current directory.

If S.SYSEDIT 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 CARD

LDCMR (p₁,p₂, . . . p_n)

p Each p is a nonpositional parameter as described:

(Input file parameter)

F=filename If P0=P is not selected, this file contains CMR segment relocatable binaries and/or CMRDIR. When segment binaries needed to build a CMR are not on the named file, they are taken from library CMRLIB. If two or more copies of the same

segment exist on the named file, 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, the named file contains the same information as ZZZZCMR (area table, segment table, and CMR without the low core tables).

- F** **Use file LGO.**
- F=0** **No load. The B, L, LO parameters are ignored. PO options A and T are ignored.**
- Absent** **No input file.**

(System image file parameter)

- B=filename** **LDCMR writes the system image to the named file. This file can be used later to create a new system (see PO options).**
- B** **File CMR is used.**
- B=0 or**
Absent **No file is generated.**

(List file parameter)

- L=filename** **All output goes to the named file.**
- L or**
Absent **All output goes to file OUTPUT.**

(List options parameter)

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 CDC 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.

<u>Option</u>	<u>Map Contents</u>
S	CDC CYBER Loader statistics
B	CDC CYBER Loader block map
E	CDC CYBER Loader entry point map
X	CDC CYBER Loader entry point cross references
L	LDCMR map
N	No CDC CYBER Loader maps are generated

- LO** **Equivalent to LO=SBEXL**
- Absent** **Equivalent to LO=L**

(Processing options parameter)

PO=string Each character selects an option. order of execution is IPTRDCAB.

The following characters in the string select the options:

- 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. D is completed before A, so PO=DA has the same effect as PO=A.
 - I Initialize system. Used only at deadstart time to examine deadstart options and perform indicated action. This option overrides all other PO options.
 - P Input file specified by F parameter has been pre-processed 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 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.
- Absent No options are used.

LDCMR FILES

lfn	Cataloged	Use
ZZZZZ03	No	Read system library binaries from this file. CMR directory may be on this file.
ZZZZZ04	No	Same as ZZZZZ03.
ZZZZZL0	No	CDC CYBER Loader input file containing CM resident binaries.
ZZZZZL1	No	CDC CYBER Loader input file containing the monitor area binaries.
ZZZZZL2	No	CDC CYBER Loader input file containing the user area binaries.
ZZZZZL3 } ZZZZZL4 } ZZZZZL5 } ZZZZZL6 }	No	Reserved for CDC.
ZZZZZL7	No	File containing LDCMR= and a dummy (0,0) overlay. This forces CDC CYBER Loader to load at the correct address and return to LDCMR when loading completes.
ZZZZZL8	No	CDC CYBER Loader output file where all the loaded segments are written.
ZZZZZL9	No	LDCMR scratch file used to save its tables while CDC 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	Save the area table, segment table, and segments on this file so that LDCMR does not need to reload them.

PERMANENT FILE PARAMETERS

lfn	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 will return or release all LDCMR files.

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 has signalled 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:

1. Read absolute CM.
2. Assign or release control point 0 ECS from the direct access area.
3. Catalog permanent files with ID=SYSTEM at control point zero.
4. Reload LDCMR after CDC CYBER Loader completes loading each stage.
5. Bootstrap a new or old ECS system.
6. Move the directory.
7. 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, see section 8 on EDITLIB. LDC uses the same procedure.

RESERVED NAMES

LDCMR uses the following names which should not be used by the operating system:

DUMMY_{nn}

nn A number from 00B to 77B

The names DUMMY_{nn} 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.

SAMPLE JOBS

Adding a new system to ECS and booting the new system. The new binaries are on file X. A map will also be generated.

LDCMR (PO=AB,F=X)

Replace the old system with the new and release the ECS space.

LDCMR (F=0,PO=R)

To get a load map of the deadstart system.

LDCMR.

To save the preloaded system for later use.

REQUEST(x,*PF)

LDCMR(B=x)

CATALOG(x,pfn,ID=user)

To load and bootstrap a new system from the preloaded file.

ATTACH(x,pfn,ID=user)

LDCMR(F=x,PO=PAB)

LDCMR ERROR MESSAGES

NON-FATAL MESSAGES

BAD ARGUMENT TO LDCMR

B=filenam IGNORED

Use of F=0,B=filenam causes LDCMR to issue this message.

CMR LIBRARY SHOULD BE DISK RESIDENT

COPYING INPUT TO RANDOM FILE

DUPLICATE PROGRAM IN INPUT-progname

There is an extra copy of progname for each occurrence of this message.

ECS COMPARE ERROR. RA+X0=nnnnnnnn
WROTE - XXXXXXXXXXXXXXXXXXXXXXXX
READ - YYYYYYYYYYYYYYYYYYYYYY

LDCMR will try to load the segment or table at the address nnnnnnn+1. This may continue until LDCMR runs out of ECS. If an ECS parity error occurred, then this will be preceded by a parity error message.

ILLEGAL CHARACTER IN ARGUMENT IS DISCARDED

LDCMR PERMANENT FILE ERROR

LFN=lfm nn FC STAT=mm
PFN=pfn

The permanent file given by pfn was used with the permanent file function of nn and received a status of mm.

LDCMR WAITING FOR pfn

The permanent file pfn is attached to another control point.

LOCKING EDITLIB OUT

LDCMR is attaching ZZZZZ01, ZZZZZ02, and ZZZZZ23.

MISSING SYSTEM CMR LIBRARY

LDL was called to find the system library as given by the CMR on input or by the CMR entry T.ENTRY+W.SGLIB.

MORE THAN SEVEN CHARACTERS -- EXCESS IGNORED

LDCMR control card cracker will only handle parameters up to seven characters.

segname MUST BE IN CM RESIDENT

The segment given will be moved to the CM resident area because the SEGDEF macro specifies this to be a CM resident segment.

NO FILE TO CATALOG

LDCMR was called with F=0,PO=C which means there is no ZZZZCMR file to catalog.

NO SYSTEM TO ADD TO ECS

LDCMR was called with F=0,PO=A which means there is no ZZZZCMR to build an ECS system.

PARITY ERROR K=nnnnnn X0=xxxxxxxx

An ECS read parity error occurred with X0=xxxxxxxx and the length of nnnnnn.

PO=A OPTION IGNORED

If a CMR was on the input file, LDCMR will not add this system to ECS because it may not match the running system.

PURGE A CYCLE OF ZZZZCMR THEN TYPE NOGO

There are already five cycles of ZZZZCMR. Purge one or drop this job.

PURGING ALL ZZZZCMR FILES

A deadstart message issued if LDCMR cannot use a cycle of ZZZZCMR to build a system (2.B deadstart with date-time stamps matching).

REMOVING EDITLIB LOCKOUT

Modify and return system files ZZZZZ01, ZZZZZ02, and ZZZZZ23.

TERMINATOR MISSING

The LDCMR control card must be terminated by a right parenthesis or period.

UNDEFINED LOADER MAP OPTION-x

The letter x is not a valid loader map option.

USING CMR FROM INPUT FILE

System entry points and date-time stamp are taken from the CMR on the input file.

USING LAST LOADER MAP OPTION SPECIFIED

Multiple loader map options specified.

USING SYSTEM LIBRARY xxxxxxx

Using the alternate system library xxxxxxx for all segments and routines.

= WITH NO FOLLOWING VALUE -- IGNORED

ZZZCMR CATALOGED

FATAL MESSAGES

AREA DEFINITION OUT-OF-ORDER IN CMRDIR

Specify the areas in increasing order.

BAD CMR ON INPUT FILE

Pointers in word P:AREA are illegal or the binary is not a complete 50 table.

BAD FILE - xxxxxxx

File xxxxxxx does not conform to LDCMR specifications.

CMR IS NOT A LEGAL SEGMENT NAME

Remove segment CMR from CMRDIR.

COMMON BLOCKS NOT PERMITTED - xxxxxxx

Segment xxxxxxx contains a common block which LDCMR cannot handle at this time.

COULD NOT RECOVER ECS AREA TABLE

The new ECS system has a parity error in the area table. We cannot move the area table without changing the ECS and CM alternate ECS addresses.

DATE-TIME STAMPS DO NOT MATCH

The running system has a different date-time stamp than the one just read on the file LDCMR.

DUPLICATE ENTRY POINT NAME - xxxxxxx

Entry point xxxxxxx has occurred in a previous segment or is defined in the T.ENTRY table.

ECS WRITE ABORT

ENTRY xxxxxxxx IS IN THE WRONG MODE

A GOTO or CALL was made to xxxxxxxx which is not in the same mode as the calling segment.

INCREASE LBUF IN LDCMR

INCREASE LWSA IN LDCMR

ILLEGAL FILE NAME -xxxxxxx

INIT NOT IN SEGMENT TABLE

Segment INIT is required for the system initialization after a bootstrap.

INSUFFICIENT FL. (nnnnnn USED)

LDCMR requires more field length.

LDCMR CANNOT RUN ON THIS SYSTEM (T.AREA=0)

The CMR being used has P.AREA with zero pointers for T.ENTRY and T.AREA. LDCMR must find the entry points, date-time stamp and alternate library.

LDCMR SYSTEM ERROR

Get a dump for a systems analyst to examine. LDCMR has been betrayed by the operating system or itself.

MISSING AREA HEADER IN CMRDIR

LDCMR is expecting an area header after the lengths of the stack and trace buffers.

MISSING ENTRY POINT - xxxxxxxx

MISSING PROGRAM -- xxxxxxxx

MISSING 34 TABLE IN SEGMENT xxxxxxxx

PIDL table missing from the segment binary.

MISSING 36 TABLE IN SEGMENT xxxxxxxx

ENTR table missing from segment binary.

MISSING 40 TABLE IN SEGMENT xxxxxxxx

TEXT table missing from the segment binary.

MISSING 50 TABLE IN CMRDIR

CMRDIR must be an absolute program with the entry address at the LWA of CMRDIR.

PO=I ON INITIAL LDCMR

PO=I is required on first LDCMR and is illegal on later LDCMRs.

PO=P WITHOUT INPUT FILE

If PO=P is selected, the F=filename option must be used.

SEGMENTS MISSING FROM FILE - xxxxxxx

File xxxxxxx does not contain all the segments it should.

TABLE LENGTH ERROR IN CMRDIR

The lengths of the stack and trace buffers are larger than 10000B, which usually means CMRDIR is bad.

SYSTEM DYNAMIC DUMP

The SCOPE operating system includes software which allows a facility to take an on-line core-image dump of PPU and central memory and to list the dump on printed output.

This facility is currently used by INTERCOM to document the PPU and central memory contents at the time that a detected error forces an INTERCOM Restart.

INTERFACE

All dumps are written, directly by stack request, to 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 66B. 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 ident 602B. Current RMS position is kept in the supplement.

Peripheral processor memory dumps are written directly from the PPU using macros defined on system text SDDTEXT to establish the interface for access to the file. CATALOG, EXTEND, and PURGE of the file and CM dumps are performed by PPU service program 1DD. Text SDDTEXT includes macros which provide a standard interface to call 1DD.

DYNAMIC DUMP FILE

Each memory dump request is written as a separate record on the system dynamic dump file.

Peripheral processor dumps are unlabeled and are written as single records with end-of-record level zero.

Each CM dump includes a label followed by a contiguous block of central memory core image terminated by a short PRU with end-of-record level 15B. The label and the core-image dump are separate records. The label format is:

59	0	Contents of:																		
<table border="1"><tr><td>C M (x x x x x x ,</td><td>0</td></tr><tr><td>y y y y y y)</td><td>1</td></tr><tr><td> H H . M M . S S</td><td>2</td></tr><tr><td> M M . D D . Y Y</td><td>3</td></tr><tr><td> </td><td>4</td></tr><tr><td> System Label.</td><td>5</td></tr><tr><td> </td><td>6</td></tr><tr><td> </td><td>7</td></tr><tr><td> </td><td>10</td></tr></table>		C M (x x x x x x ,	0	y y y y y y)	1	H H . M M . S S	2	M M . D D . Y Y	3		4	System Label.	5		6		7		10	T.CLK
		C M (x x x x x x ,	0																	
		y y y y y y)	1																	
		H H . M M . S S	2																	
		M M . D D . Y Y	3																	
			4																	
		System Label.	5																	
			6																	
			7																	
			10																	
	T.DATE																			
	T.SLAB2																			
	T.SLAB3																			
	T.SLAB4																			
	T.SLAB5																			
	T.SLAB6																			

All entries are in display code. Words 0-1 of the label define the CM address bounds of the following dump. Words 2-10 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:

59	0	Contents of:
E N D O F F I L	0	
E N O n n n n n	1	
H H . M M . S S	2	T.CLK
M M . D D . Y Y	3	T.DATE
System Label.	4	T.SLAB2
	5	T.SLAB3
	6	T.SLAB4
	7	T.SLAB5
	10	T.SLAB6

All entries are in display code. Word 1 contains the dump number referenced by LISTCID. Words 2-10 correspond to the same words in the CM dump label.

LISTING SYSTEM DYNAMIC DUMP FILES

LISTCID (list core-image dump) is a control card callable system utility which creates listable output from the central site console. LISTCID requests a central site operator GO or DROP to provide security for access to absolute system dumps.

LISTCID will list all, or selected portions, of a core-image dump file, depending on the specified control card arguments.

Calling Sequence:

The LISTCID control card has the following format:

LISTCID (p1,p2,...,pn)

where p may be a keyword or a keyword equated to a value. Keywords, listed below, may be specified in any order.

Keyword	Significance
I	Input file designation
omitted	Core-image input is taken from the system dynamic dump file, ZZZZZDD.
or	
I	
I=lfm	core-image input is taken from file lfm.
O	Output file designation
omitted	Listable output is placed on file OUTPUT and in the output queue at end-of-job (unless otherwise disposed by the calling job).
or	
O	
O=lfm	Listable output is placed on file lfm. Disposition of lfm must be provided by the calling job.
C	List available dumped central memory in byte format.†
omitted	No specific CM dump.
C	List all central memory dumps encountered.
C=L	List central memory from address 0 to address L.
C=F-L	List central memory from address F to address L.
CD	List available dumped central memory in display code format.†
omitted	No specific CM dump.
CD	List all central memory dumps encountered.
CD=L	List central memory from address 0 to address L.
CD=F-L	List central memory from address F to address L.

†If no list control arguments (C, CD, or P) are specified, all of the core-image input file will be listed in byte 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.

P List available dumped peripheral processors in byte format.†

F File selector parameter

Each dynamic dump is concluded with an end-of-file. As multiple files may exist on the dynamic dump file, the F parameter is used to select which of the multiple files to read. File numbers are assumed decimal unless a radix is specified.

omitted All files selected.
or F

F=F File number F is selected. (first file is number 1)

F=F-L File numbers F through L are selected.

All numeric values may be octal or decimal. The character B or D immediately following the number specifies octal or decimal. If not specified, all addresses are assumed to be octal.

When the dump input file is ZZZZDD, LISTCID issues the following message:

SYSTEM DUMP LIST REQUESTED/GO OR DROP

The central site operator must enter j.GO to allow LISTCID to continue.

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.

When LISTCID requests an operator GO or DROP, the operator must enter:

n. GO

To list all peripheral processor dumps and any central memory locations between 0 and 10000₈, a batch job might contain the control card:

LISTCID (P,CM0-10000)

†If no list control arguments (C, CD, or P) are specified, all of the core-image input file will be listed in byte 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.

LISTCID issues the following error messages:

LISTCID ARGUMENT ERROR

Unrecognized keyword.

LISTCID NUMERIC CONSTANT ERROR

A numeric value is ill-formed.

LISTCID ARGUMENT CANT BE EQUATED

A P parameter was followed by an = sign.

LISTCID ARGUMENT FWA .GT. LWA

CM parameter (C or CD) first address specified is greater than second address.

LISTCID-TOO MANY PARAMETERS

Too many parameters were entered on the LISTCID control card.

LISTCID-ONLY ONE F PARAMETER ALLOWED

More than one F parameter was entered on the LISTCID control card.

INCOMPLETE DUMP

The dynamic dump file contains an incomplete dump. A CM or ECS header specified an address range for which the following record on the dynamic dump file did not contain sufficient words.

DUMP FORMAT ERROR OR EMPTY FILE

The dynamic dump file was not formatted properly, or it was empty. The dynamic dump file is listed up to the point where the error is detected.

STANDARD SCOPE CHARACTER SETS

A

The character set selected when the system is installed should be compatible with the printers.

With an installation parameter, the installation keypunch format standard can be selected as 026 or 029; the installation parameter can also allow a user to override the standard; a user may select a keypunch mode for his input deck by punching 26 or 29 in columns 79 and 80 of his JOB card or any 7/8/9 end-of-record card. The mode remains set for the remainder of the job or until it is reset by a different mode selection on another 7/8/9 card.

The physical end of a unit record (card, print line or any other significant character string) is delimited by 12 zero bits. The use of two colons in succession will produce 12 zero bits interpreted as a unit record terminator. This general restriction on the use of the colon character applies throughout SCOPE 3 and all of its product sets.

SCOPE 3.4
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
: 1	:	00†	8-2	00	8-2	3A	6	6	41	6	06	6	36
A	A	01	12-1	61	12-1	41	7	7	42	7	07	7	37
B	B	02	12-2	62	12-2	42	8	8	43	8	10	8	38
C	C	03	12-3	63	12-3	43	9	9	44	9	11	9	39
D	D	04	12-4	64	12-4	44	+	+	45	12	60	12-8-6	28
E	E	05	12-5	65	12-5	45	-	-	46	11	40	11	2D
F	F	06	12-6	66	12-6	46	*	*	47	11-8-4	54	11-8-4	2A
G	G	07	12-7	67	12-7	47	/	/	50	0-1	21	0-1	2F
H	H	10	12-8	70	12-8	48	((51	0-8-4	34	12-8-5	28
I	I	11	12-9	71	12-9	49))	52	12-8-4	74	11-8-5	29
J	J	12	11-1	41	11-1	4A	\$	\$	53	11-8-3	53	11-8-3	24
K	K	13	11-2	42	11-2	4B	=	=	54	8-3	13	8-6	3D
L	L	14	11-3	43	11-3	4C	blank	blank	55	no punch	20	no punch	20
M	M	15	11-4	44	11-4	4D	, (comma)	, (comma)	56	0-8-3	33	0-8-3	2C
N	N	16	11-5	45	11-5	4E	. (period)	. (period)	57	12-8-3	73	12-8-3	2E
O	O	17	11-6	46	11-6	4F	≡	#	60	0-8-6	36	8-3	23
P	P	20	11-7	47	11-7	50			61	8-7	17	12-8-2	5B
Q	Q	21	11-8	50	11-8	51			62	0-8-2	32	11-8-2	5D
R	R	22	11-9	51	11-9	52	%††	%	63	8-6	16	0-8-4	29
S	S	23	0-2	22	0-2	53	≠	" (quote)	64	8-4	14	8-7	22
T	T	24	0-3	23	0-3	54	→	' (underline)	65	0-8-5	35	0-8-5	5F
U	U	25	0-4	24	0-4	55	v	! (exclamation)	66	11-0 or 11-8-2†††	52	12-8-7 or 11-0†††	21
V	V	26	0-5	25	0-5	56	^	&	67	0-8-7	37	12	26
W	W	27	0-6	26	0-6	57	↑	' (apostrophe)	70	11-8-5	55	8-5	27
X	X	30	0-7	27	0-7	58	↓	? (question)	71	11-8-6	56	0-8-7	3F
Y	Y	31	0-8	30	0-8	59	<	<	72	12-0 or 12-8-4 or 12-8-2†††	72	12-8-4 or 12-0†††	3C
Z	Z	32	0-9	31	0-9	5A	>	>	73	11-8-7	57	0-8-6	3E
0	0	33	0	12	0	30	∇	@	74	8-5	15	8-4	40
1	1	34	1	01	1	31	∇	~ (circumflex)	75	12-8-5	75	0-8-2	5C
2	2	35	2	02	2	32	∇	∇	76	12-8-6	76	11-8-7	5E
3	3	36	3	03	3	33	∇	∇	77	12-8-7	77	11-8-6	3B
4	4	37	4	04	4	34	∇	∇					
5	5	40	5	05	5	35	∇	∇					

† 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 the CDC 63-graphic set, display code 00 has no associated graphic or Hollerith code; display code 63 is the colon (8-2 punch).

††† The alternate Hollerith (026) and ASCII (029) punches are accepted for input only.

CDC CHARACTER SET COLLATING SEQUENCE									
Collating Sequence Decimal/Octal		CDC Graphic	Display Code	External BCD	Collating Sequence Decimal/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	v	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/Octal		ASCII Graphic Subset	Display Code	ASCII Code	Collating Sequence Decimal/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	\$	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.

**CONTROL DATA CHARACTER SETS
SHOWING TRANSLATIONS BETWEEN DISPLAY CODE AND ASCII/EBCDIC**

DISPLAY CODE		ASCII				EBCDIC				DISPLAY CODE		ASCII				EBCDIC			
		UPPER CASE		LOWER CASE		UPPER CASE		LOWER CASE				UPPER CASE		LOWER CASE		UPPER CASE		LOWER CASE	
OCTAL	CH	CH	HEX	CH	HEX	CH	HEX	CH	HEX	OCTAL	CH	CH	HEX	CH	HEX	CH	HEX	CH	HEX
00			3A	SUB	1A		7A	SUB	3F	40	5	5	35	NAK	15	5	F5	NAK	3D
01	A	A	41	a	61	A	C1	a	81	41	6	6	36	SYN	16	6	F6	SYN	32
02	B	B	42	b	62	B	C2	b	82	42	7	7	37	ETB	17	7	F7	ETB	26
03	C	C	43	c	63	C	C3	c	83	43	8	8	38	CAN	18	8	F8	CAN	18
04	D	D	44	d	64	D	C4	d	84	44	9	9	39	EM	19	9	F9	EM	19
05	E	E	45	e	65	E	C5	e	85	45	+	+	28	VT	0B	+	4E	VT	08
06	F	F	46	f	66	F	C6	f	86	46	-	-	2D	CR	0D	-	60	CR	0D
07	G	G	47	g	67	G	C7	g	87	47	*	*	2A	LF	0A	*	5C	LF	25
10	H	H	48	h	68	H	C8	h	88	50	/	/	2F	SI	0F	/	61	SI	0F
11	I	I	49	i	69	I	C9	i	89	51	((28	BS	08	(4D	BS	16
12	J	J	4A	j	6A	J	D1	j	91	52))	29	HT	09)	5D	HT	05
13	K	K	4B	k	6B	K	D2	k	92	53	S	S	24	EOT	04	S	5B	EOT	37
14	L	L	4C	l	6C	L	D3	l	93	54			3D	GS	1D		7E	IGS	1D
15	M	M	4D	m	6D	M	D4	m	94	55	SP	SP	20	NUL	00	SP	40	NUL	00
16	N	N	4E	n	6E	N	D5	n	95	56	.	.	2C	FF	0C	.	6B	FF	0C
17	O	O	4F	o	6F	O	D6	o	96	57	,	,	2E	SO	0E	,	4B	SO	0E
10	P	P	50	p	70	P	D7	p	97	60	=	=	23	ETX	03	=	7B	ETX	03
21	Q	Q	51	q	71	Q	D8	q	98	61			58	FS	1C		4A	IFS	1C
22	R	R	52	r	72	R	D9	r	99	62	!	!	5D	SOH	01	!	5A	SOH	01
23	S	S	53	s	73	S	DA	s	A2	63	%	%	25	ENO	05	%	6C	ENO	2D
24	T	T	54	t	74	T	DB	t	A3	64	/	/	22	STX	02	/	7F	STX	02
25	U	U	55	u	75	U	DC	u	A4	65	.	.	5F	DEL	7F	.	6D	DEL	07
26	V	V	56	v	76	V	DD	v	A5	66	~	~	21	;	7D	~	4F	;	0D
27	W	W	57	w	77	W	DE	w	A6	67	^	^	26	ACK	06	^	50	ACK	2E
30	X	X	58	x	78	X	DF	x	A7	70	! &	&	27	BEL	07	!	7D	BEL	2F
31	Y	Y	59	y	79	Y	EA	y	A8	71	>	>	3F	US	1F		6F	IUS	1F
32	Z	Z	5A	z	7A	Z	EB	z	A9	72	<	<	3C	!	7B	<	4C	!	C0
33	0	0	30	DLE	10	0	F0	DLE	10	73	>	>	3E	RS	1E	>	6E	IRS	1E
34	1	1	31	DC1	11	1	F1	DC1	11	74	< @	@	40	.	60	@	7C	.	79
35	2	2	32	DC2	12	2	F2	DC2	12	75	< \	\	5C	!	7C	\	E0	!	6A
36	3	3	33	DC3	13	3	F3	TM	13	76	< ^	^	5E	!	7E	^	5F	!	A1
37	4	4	34	DC4	14	4	F4	DC4	3C	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 (55g).

HEXADECIMAL-OCTAL CONVERSION TABLE

		First Hexadecimal Digit															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Second Hexadecimal Digit	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								

SCOPE SYMBOL DEFINITION

B

SCOPE Text (SCPTEXT) contains system macros, micros, and symbols used by COMPASS CPU and PPU programs that comprise the SCOPE operating system. SCPTEXT contains the following common decks:

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	PPU 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 PPU 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 card (including a request for MTO1
REQUEST(OLDPL,E,HY) SCOPEPL1
UPDATE(Q)
COMPASS(S=0,I=COMPILE)
7/8/9
*COMPILE PPTEXT
6/7/8/9
```

Refer to the SCOPE 3.4 Reference Manual, section 6, 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.

C Byte number in CM word (0-4). C identifiers are used for flags and parameters of 12 bits or less.

CH Pseudo channel assignments

D Direct cells

EX M.ICE or M.SPM parameter values

F Error flag values

L Lengths

LE Length of table entries

M PPU request of monitor

N Number

O Stack processor orders

OV PPU overlays; mnemonic is the overlay name

P CM location of pointer words

R PPU resident entry points

S Number of bits to right shift a parameter to right justify it in a PPU 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 70B, which contains binary zeros.

SECTION 1

CENTRAL MEMORY RESIDENT TABLES

CONTENTS PART II

1	CENTRAL MEMORY RESIDENT TABLES	
	Central Memory Resident	II-1-1
	CMR Pointer Area	II-1-2
	Channel Status Table	II-1-15
	PP Status Words	II-1-16
	Exchange Package	II-1-16
	Control Point Area	II-1-17
	System Job Exchange Package Area	II-1-21
	PP Communication Area	II-1-22
	PP Program Name Reservations	II-1-24
	Monitor Functions	II-1-33
	Equipment Status Table	II-1-35
	Device Codes	II-1-37
	File Name Table	II-1-39
	Disposition Code Values	II-1-52
	Link Identification Values	II-1-54
	INTERCOM Multiplexer Table	II-1-55
	Device Activity Table Entry	II-1-59
	Dummy Entry for Dual Access Device	II-1-59
	Tapes Staging Table	II-1-60
	Attached Permanent File Table	II-1-60
	Request Stack Entry and Stack Processor Order Codes	II-1-61
	Request Scheduling Table	II-1-62
	Record Block Reservation Table	II-1-62
	Device Status Table (Master Entry)	II-1-63
	DST Multiple Access Member Entries	II-1-63
	Sequencer Table	II-1-64
	Overview of Mounted Set Table (MST)	II-1-65
	Overview of Dismountable Device Table (DDT)	II-1-66
	VSNBUF	II-1-68
	Tapes Table	II-1-69
	Mailbox	II-1-70
	ID Table	II-1-71
	Subsystem Control Table	II-1-72
	Peripheral Job Table	II-1-73
	Dayfile FET and Buffer Area	II-1-74
	Scheduler Performance Table (SCHPT)	II-1-75
	Job Control Area	II-1-76
	Job Descriptor Table Entry	II-1-77
	Breakpoint Table	II-1-81
	Area Table	II-1-83
	Entry Table	II-1-84
	Segment Table	II-1-85

	CEFAP Buffer Area	II-1-87
	Empty Page Stack	II-1-89
	Subpage Buffer and Record Descriptors	II-1-90
	SCB (System Circular Buffer)	II-1-92
	CMR Library Directory	II-1-93
	Library Name Table Entry	II-1-93
	PP Program Name Table Entry	II-1-94
	CM Resident Library Format	II-1-94
	EPNT/ERT/PNUT/PNT Entry Formats	II-1-95
	INTERCOM Pointer and Buffer Area	II-1-96
	INTERCOM Driver Communication Area (DCA)	II-1-100
	User Table	II-1-103
	Auxiliary User Table	II-1-107
	Multi-User Job Table	II-1-112
	274 Interactive Graphics System (IGS) User Table	II-1-113
	Record Block Table Entry	II-1-114
2	JOB CONTROL POINT TABLES	
	RA Communication Area	II-2-1
	File Environment Table	II-2-3
	SCOPE CIO Codes in Octal	II-2-4
	Local File Names	II-2-5
	Entry Point Names	II-2-7
	FET Codes	II-2-8
	File Information Table	II-2-10
	File Definition Block	II-2-20
	REQ Function Parameter List	II-2-22
	Subchannel Table (SCHT)	II-2-24
	SPOT Name Table (SNT)	II-2-26
	DSD/INTERCOM Communication through Station (MSG Table)	II-2-28
3	DISK TABLES AND FILE FORMATS	
	Permanent File Directory Overview	II-3-1
	PFD Entry	II-3-2
	Permanent File Catalog Overview	II-3-5
	Permanent File Catalog (PFC) Entry	II-3-6
	Permanent File Set Device Label	II-3-13
	Set Member Table (Mass Storage)	II-3-18
	Overview of Device Allocation Map	II-3-20
	Logical Flaw Table	II-3-22
	Physical Flaw Table	II-3-22
	Subdirectory Table (pre 3.4.4)	II-3-25
	PFC Allocation Matrix (PAM)	II-3-26
	W Record Header	II-3-27
	I Block Header	II-3-27
	Block Format (DA File)	II-3-27
	Index Block Format (IS File)	II-3-28
	Data Block Format (IS File)	II-3-28

	FSTT (IS File)	II-3-29
	FSTT (DA File)	II-3-31
	FSTT (AK File)	II-3-33
	Swap File Format	II-3-35
4	EXTENDED CORE STORAGE TABLES	
	Overall ECS Format	II-4-1
	ECS Label	II-4-1
	ECS Partition Format (within each partition)	II-4-2
	Inter-Computer Area	II-4-3
	System Pointer Area	II-4-4
	Flaw Table	II-4-5
	System Page	II-4-5
	Segments in an ECS System	II-4-6

	FSTT (IS File)	II-3-29
	FSTT (DA File)	II-3-31
	FSTT (AK File)	II-3-33
	Swap File Format	II-3-35
4	EXTENDED CORE STORAGE TABLES	
	Overall ECS Format	II-4-1
	ECS Label	II-4-1
	ECS Partition Format (within each partition)	II-4-2
	Inter-Computer Area	II-4-3
	System Pointer Area	II-4-4
	Flaw Table	II-4-5
	System Page	II-4-5
	Segments in an ECS System	II-4-6

CMR POINTER AREA

	59	47	35	29	23	11	0		
P.AAZ	ABSOLUTE ADDRESS ZERO							0	
P.LIB	C.DIRFWA A	FWA of Library Directory			LWA+1 Library Directory		C.DSFLAG Deadstart Load Flag	1	
P.RBR P.RBT P.CMLWA	C.RBRAD FWA of RBR Area		C.RBTEC RBT Ordinal of Empty Chain		Length/100B of RBT Area		C.CMLWA (LWA+1)/100B of CM	2	
P.NPP P.NCP P.DFB	FWA/10B of Dayfile Buffer		(Reserved)		C.NPP No. of PPs		C.NCP No. of CPs	3	
P.SEQ P.FNT P.HEC	C.FNT FWA of FNT	C.FNTLWA LWA+1 of FNT		C.SEQ T.SEQ/10B		C.SEQL L.SEQ		C.HEC Hardware Error Count	4
P.CST P.PCOM P.EST	C.EST FWA of EST		LWA+1 of EST		C.CST FWA of CST		C.CSTL LWA+1 of CST	C.PCOM Address of Comm Area PP1	5
P.PFM1	(Reserved)		C.APFL No. of APF Entries		C.APF FWA of APF		C.PFMCH Interlock Byte	6	
P.MST P.DDT P.DSMO	C.DSMO sys. set def. PF set		C.NDDT N.FDDT N.VDDT		C.DDT T.DDT/10B		C.NMST L.MST	C.MST T.MST/10B	7
P.INS	(Reserved for Installations)							10	
P.EIRPR	C.LEPAGE L.ECSTK+1		C.ECSPRM T.ECSPRM		ICC Area Address			11	
P.ELBST	Maximum Length/ 1000B of ECS Library File		ECS Flaw Table Address			ECS Page Stack Address		12	
P.RQS	C.DAT T.DAT		C.DATL L.DAT		C.RQSFS FWA/2 of Request Stack		No. of DST Entries	FWA/10B of DST	13
P.TAPES	C.TAPES T.TAPES/10B		L.TAPES		Reserved			14	
P.STG							C.STG T.STG	15	
P.INT	C.INT/C.IFL (LWA+1)/100B of INTERCOM		C.ITABL FWA of Multiplexer Table		C.IBUFF FWA of INTERCOM Pointer and Buffer Area		C.ILTABL Length of Multiplexer Table		16
P.MFL							C.MFL max. job FL/100B	17	

NOTES: CMR POINTER AREA

These descriptions begin with the location number and the symbol. Contents of words may 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	Compare/move availability indicator 0=Not available 1=Available
4	S.ECSLVL	ECS level 0=No ECS 1=ECS up
5	S.USETS	1=Validate user sets
6-7	S.IOLVL	00=Recover I/O queues 01=Do not recover I/O queues 10=Initialize
8-9	S.SYSLVL	System level 00=A 01=B 10=C 11=D
10-11	S.ACTION	System action 00=Load 01=Recovery 02=Checkpoint recovery

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.

NOTES: CMR POINTER AREA (CONT'D)

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 PP's 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 diagnostic sequencer table. 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

Number of empty attached permanent file table entries, FWA of attached permanent file table and the permanent file interlock byte are in bytes 1 to 4, respectively.

Bits in the byte C.PFMCH are:

0-10		Reserved
11	S.PFCIOQ	PFC full flag - input jobs 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 dismountable device table	
C.NMST	L.MST is the number of entries in the mounted set table	
C.MST	T.MST is the first word address divided by 10 octal of the mounted set table	

NOTES: CMR POINTER AREA (CONT'D)

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 and page stack processor address. 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

Bytes 0 and 1 contain the FWA/10 and length of the tape configuration table.

15 P.STG

The FWA of the tape staging table is in byte 4. The rest of the word is currently unused.

16 P.INT

Byte 0 contains the (LWA+1)/100B 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 4 contains the maximum job field length (MFL)/100B.

NOTES: CMR POINTER AREA (CONT'D)

	59	47	41	35	23	17	11	0		
T.JDATE	(Leading Zeros) Y Y D D D							20		
P.NRBR	C.NROS 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 (YYDDD)		Reserved			Time in Binary (HHMMSS)			22	
P.EVICT P.RMSBUF P.SSCT P.SXDT P.ELST P.CMFL		C.ELST T.ELST/10B	C.SSCT T.SSCT/10B	C.RMSBUF T.RMSBUF FWA of RMSBUF		Trace Buffer T.TRB/10B			23	
							Machine FL/100B		24	
	^ S Y S T E M ^ ^ ^									25
T.CPJOBN P.PJT P.SPDRIP	Job Sequence Number	C.SPDRIP DST Ordinal for 1SP Drop	Job Count		C.PJTFWA T.PJT/10B	C.PJTLWA T.PJT/10B+ L.PJT/10B			26	
T.EPBL P.ECSFL	C.ECSPL ECS Page Length		C.ECSBL ECS Buffer Length			C.CPECFL Machine ECS FL/1000B			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	SCOPE									34
T.SLAB5	Version									35
T.SLAB6	3.4									36
T.MSP	Reserved	PP Name if in Step Mode			C P N	Reserved		Step Flag	37	

Reserved 1 = Step Mode

NOTES: CMR POINTER AREA (CONT'D)

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 in binary form is stored in bits 42-59; the current time, representing hours, minutes, and seconds in binary form, is stored in bits 0-17.

23 P.EVICT/P.RMSBUF/P.SSCT/P.SXDT/P.ELST

The FWA of the trace buffer, divided by 8, is stored in byte 4. The first word address of the rotating Mass Storage Buffer is stored in byte 3. The FWA of the subsystem control table, divided by 8, is stored in byte 2.

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.CPJOB/P.PJT/P.SPDRDP

Bytes 0 and 2 contain the job sequence number and job count, respectively. FWA and LWA+1 of the parameter storage area for 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 may be used to reserve additional PPs for 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.

NOTES: CMR POINTER AREA (CONT'D)

30 T.CLK

Current display clock time in the format:
hh.mm.ss.

31 T.DATE/T.SLAB1

Current calendar date is in the format:
mm/dd/yy (entered by operator)
This is the first of a 6-word system display label.

32 (IP.SYSL1) T.SLAB2

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

Bits in word 37 are:

59-48	Reserved
47-30	Name of PPU routine in step mode
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

NOTES: CMR POINTER AREA (CONT'D)

	59	47	35	23	11	0	
T.MSC	Count of PP Job Queue Entries	Number of Idle PPs	Number of Seconds*4096				40
P.CHRQ				C.CHRQ First 10 Channels	C.CHRQ2 Second 10 Channels		41
P.PPLIB	Position of CIO	0 0 0 0	Number of Programs		Address of First PPLIB Entry		42
P.VRNBUF	C.VRNFWA T.VRNBUF/10B	C.VRNFIN Pointer to First VSN	C.STGFLG Stage ON/OFF	C.VRNINT Buffer Interlock	C.VRNFUL Buffer Full Flag		43
T.CPSTA	Idle Exchange Package Address	**	Next Slice Time	2 0	Active XP Address		44
		*****		0 3 0 3	* * * *		
T.CPSTB				0 0 0 L	* * * *		45
T.MXNCTL	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 P						46
	STL Code		20	Active XP Address		2 6 1 P	46
				* * * * *		2 6 2 P	46
T.PPID	* * * *	* * *			PP Input Register Address		47
T.PPIP	* * *	* * *			PP Input Register Address		50
T.CMPID	Computer ID for ECS Partitioning						51
	(Reserved)						52
T.SPF		C.SNTLWA Length of Spot Name Table	C.SNTFWA FWA of Spot Name Table	C.CPN Station Control Point Number			53
	(Reserved)						54
T.RCHN	SPM-1RN Communications Word				First RBT Word Pair to Release		55
T.CPT1 } T.UAS }	Unassigned CM/100B	Unassigned ECS/1000B	ECS Size		Initial CPMTR Address		56
T.ECSPAR } P.EPAGE }		C.EPAGE T.EPAGE	ECS Flaw Table Flag	ECS Parity Flag	ECS Parity Address/1000B		57

L = { 0 Turned Off
1 Locked Off } P = { 0 CPUA
1 CPUB }

NOTES: CMR POINTER AREA (CONT'D)

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; 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:

59	41	35	23	17	0
Idle Exchange Package		Next Time Slice	2 0	Active Exchange Package Address	

Value while an MXN version of CPMTR is selecting the next job mode:

59	41	23	0
Idle Exchange Package	* * * * *	0 3 0 3 * * * *	

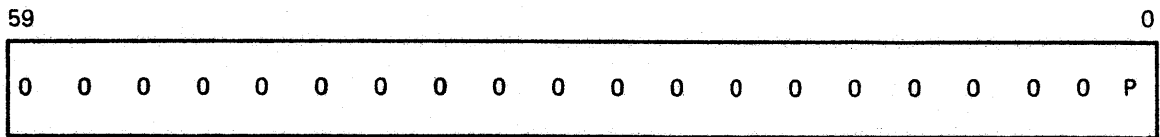
Value while the CPU is turned off:

59	41	23	0
Idle Exchange Package	* * * * *	0 0 0 L * * * *	

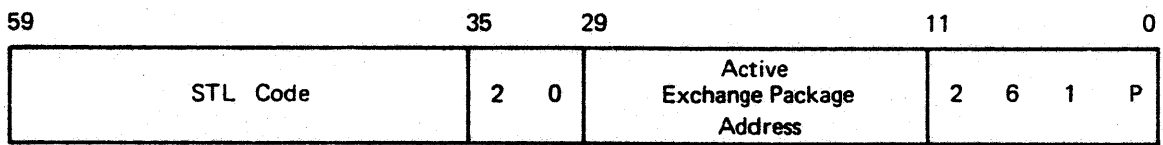
NOTES: CMR POINTER AREA (CONT'D)

46 T.MXNCTL

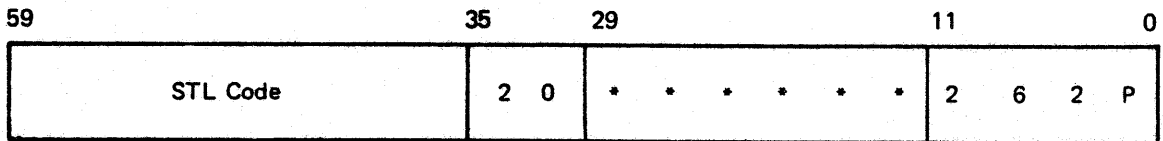
Value while in EXN mode:



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 Currently reserved.

NOTES: CMR POINTER AREA (CONT'D)

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 Currently reserved.

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

Right adjusted in bytes 0 and 1 is the FWA of the ECS page stack. 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.

NOTES: CMR POINTER AREA (CONT'D)

	59	47	35	23	11	0
P.SCH	C.LEJDT LE.JDT	C.SRS T.XPSCH/10B	C.JCA T.SCHJCA/10B	C.LJDT L.SCHJDT	C.JDT T.SCHJDT/10B	60
P.STR	C.NFL Needed FL/100B	C.JQP Queue Priority of Job in Counter	C.RFL Reserved FL/100B	C.RCL C.STMF SCH Recall	C.AFL Available FL/100B	61
T.SCHCP	Interlock Word (Scheduler)					62
T.SCHPP	Interlock Word (PP Routines)					63
	(Reserved)					64
P.MAIL P.SWPECS P.SCHPT }	C.MAILF T.MAIL/10B	C.MAILL L.MAIL	C.SWPECS L.ECSSWP	C.SCHPT T.SCHPT/10B	C.ISIZLN INTERCOM User Table	65
P.LNK P.IDT }	C.ECSLNK	(Reserved)		C.LIDT Length of ID Table	C.IDT T.IDT/10B	66
P.AREA P.ENTRY }	FWA of Breakpoint Table T.BRKPT		FWA of Entry Table T.ENTRY		FWA of Area Table T.AREA	
P.ZERO	(zeros)					70
	(Reserved)					71-76
P.PPOVL P.FDD	C.FDDCT Dump count	C.FDDLK Dynamic dump recall flag	C.FDD Dynamic dump FNT address	C.PPOVL T.PPOVL/10B	Reserved	

NOTES: CMR POINTER AREA (CONT'D)

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.

61 P.STR

Information relative to job scheduling. Communication word for scheduler/MTR.

62 P.SCHCP

Interlock word for integrated scheduler.

NOTES: CMR POINTER AREA (CONT'D)

63 P.SCHPP

Interlock word for PP routines.

64 Currently reserved.

65 P.MAIL/P.SWPECS/P.SCHPT

Pointer to and length of scheduler mailbox buffer in bytes 0 and 1, respectively. The ECS swap flags are in byte 2. Pointer to the job scheduler is in byte 3. INTERCOM user table page size or line length is in byte 4.

The ECS swap flag values 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

66 P.IDT/P.LNK

ECS link restart time control mask is in byte 0. Length of the ID Table is in byte 3. T.IDT/10B is in byte 4.

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 will 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.FDDLK(0) set by routines that wait for IDD to complete, and cleared by IDD.
Byte 2 contains the FNT address of the system dynamic dump file.
Byte 3 contains the FWA of the PP overlay table.

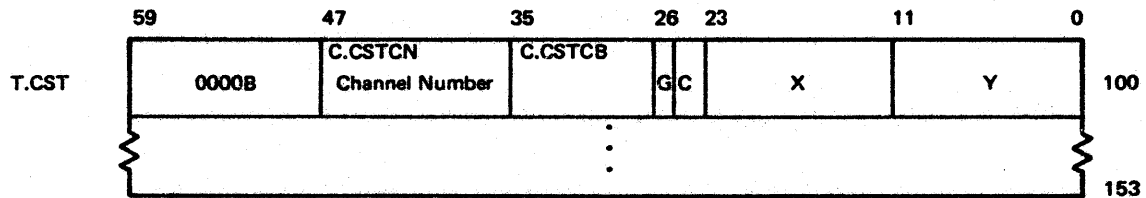
100 T.CST

Entries for the channel status table.

154 T.PPSn

One word entries containing status information for up to 20 PPs.

CHANNEL STATUS TABLE

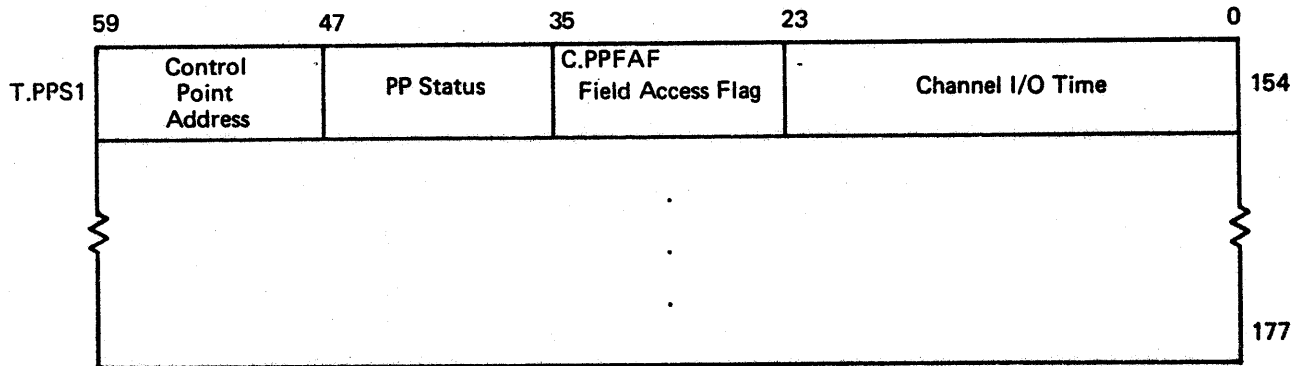


- C** MMT Conversion Table, MMT Equipment Numbers 4-7
- 00 No Table
 - 01 EBCDIC
 - 10 ASCII
 - 11 Reserved
- G** 0 Normal Charge for Channel Time
1 Do not Charge Control Point for Channel Time (S.CSNC)
- X** Address of this word
- Y** Same as X when channel not reserved
PPIR address when reserved

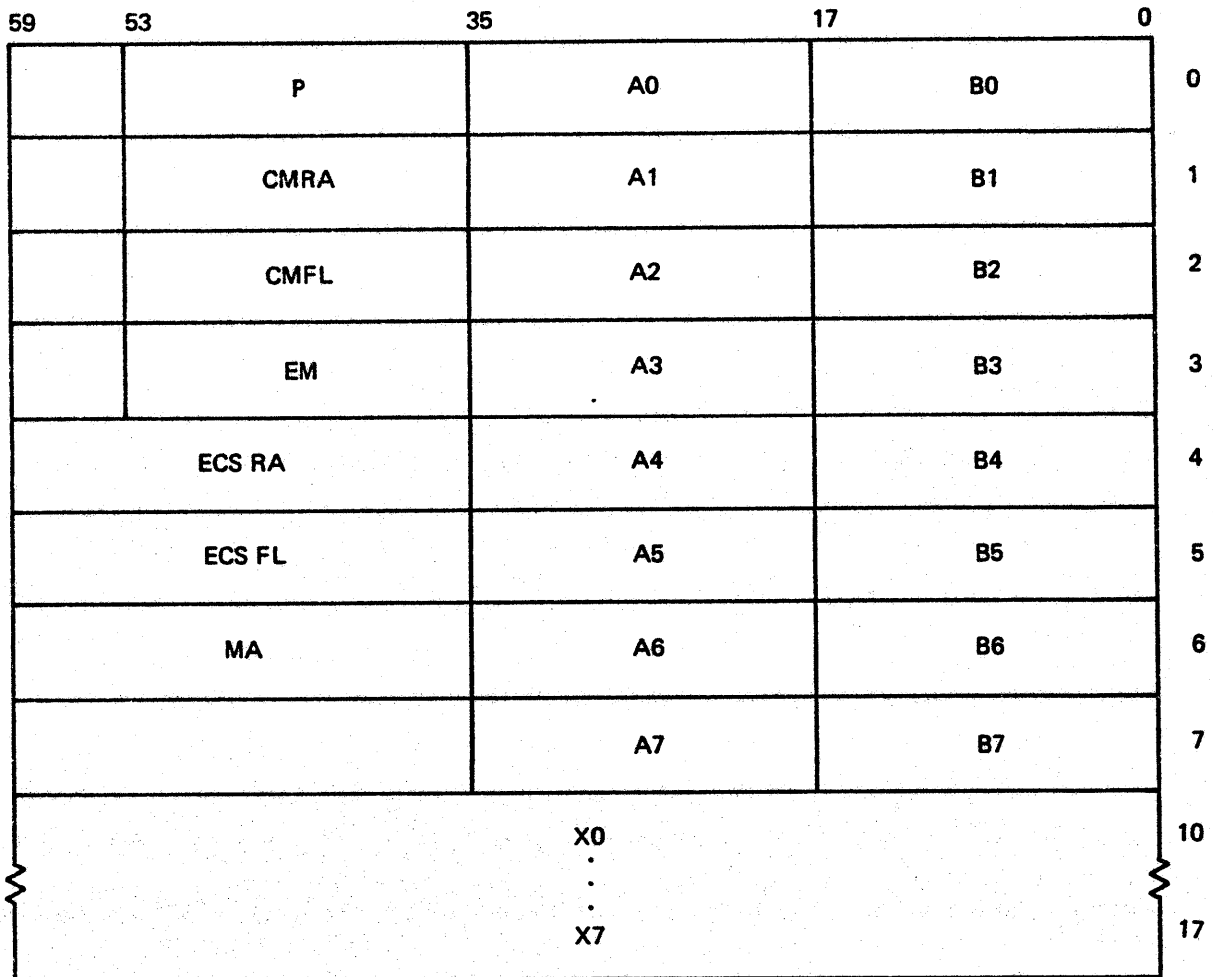
CHANNELS

00-13	Hardware channels	
14	CH.FST	(Controls access to FST)
15	CH.FNT	(Controls access to FNT)
16	CH.DDT	(Dismountable device table 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 attached PF table channel)
36	CH.INS	(For use by installation)
37	CH.MST	(Mounted set table interlock)
40	CH.EST = CH.TAPE	(Controls access to EST/TAPES table)
41	CH.ICOM	(INTERCOM/SCOPE communication interlock)
42	CH.IEMBF	(INTERCOM empty buffer channel)
43	CH.IUSER	(INTERCOM user table channel)
44	CH.SCH	(Scheduler channel)
45	CH.IHUSR	(High speed user table channel)
46	CH.IHSMT	(High speed empty buffer channel)

PP STATUS WORDS



EXCHANGE PACKAGE



EM = Exit mode
 MA = Monitor address

CONTROL POINT AREA

	59	47	44	41	35	29	23	17	11	5	0	
W.CPAn	Exchange Package											
W.CPSLIC } W.CPUST } W.CPLINK }	C.CPSTAT Status Byte	C.CPSLIC M.RCLCP Time			* * * *			C.CPUPRI C.CPLINK	Next Active Control Point			
W.CPTIME	C.CPUQS CPU-A Seconds*4096 This Quantum			C.CPUQMS CPU-B Seconds*4096 This Quantum								C.CPUAS Total CPU-A Time as Number of Seconds*4096
W.CPTIMB	C.CPUQS CPU-B Seconds*4096 This Quantum			C.CPUQMS CPU-B Seconds*4096 This Quantum								C.CPUBS Total CPU-B Time as Number of Seconds*4096
W.PPTIME } W.CPPTM }	C.CPPQS PP Seconds*4096 This Quantum			C.CPPQMS PP Seconds*4096 This Quantum			C.CPPTS Total PP Time as Number of Seconds*4096					
W.CPSTAT } W.CPFL } W.CPEF }	C.CPMEMO Error Memo	C.CPEF Error Flag		C.CPSM Storage Move			C.CPRA RA/100B		C.CPFL FL/100B			
W.CPJNAM	C.CPJNAM Job Name									JDT Ordinal		
W.CPCC	C.CPRPV Retrieve CKSM Value		C.CPRPA Retrieve Address				C.CPNFL Nominal FL/100B		C.CPNCSP Next Control Card Pointer			
W.CPECS	A					C.CPECRA ECS RA/1000B		C.CPECFL ECS FL/1000B				
W.CPDFM	Last Dayfile Message											
W.CPPRI } W.CPJCP } W.CPTIML } W.CPIOL }	C.CPTIML Current Time Limit (15 Bits)		C.CPIOL IO Time Limit (15 Bits)		C.CPPRI Job Class		C.CPECSI Initial ECS FL/1000B		C.CPFLI Initial FL/100B			
W.CPSWP } W.CPINT }	C.CPQNT Quantum			C.CPUTA User Table Address				C.CPORG C.CPEVNT Job Flags Origin				
W.CPSCH } W.CPRO }	C.CPFLG Swap Flags		C.CPJQP Job Queue Priority		C.CPRFL Reserved FL		C.CPJDA JDT Address (Absolute)					
W.SSW } W.CPSSW }	Saved SPOT Error Flags		Reserved				C.CPSSW Sense Switches					
W.CPMST									C.CPMST Setname MST Ordinal			
W.CPCSF	Core Seconds Factor (Floating Point)											
W.CPACS	(Reserved) Accumulated CM Core Seconds (Integer)											
W.CPACSE	(Reserved) Accumulated ECS Core Seconds (Integer)											

A if set, do not update exchange package ECS RA and FL

CONTROL POINT AREA (CONT'D)

	59	47	41	35	23	17	11	5	
W.CPFACT	Account Parameter for Permanent Files								50
W.CPFST W.FSTCC	FST Entry for Next Control Card PRU								51
W.CKP W.CPCKP W.CPID	C.CPDID Destination ID		C.CPSID Source ID		C.CPCON Console Checkpoint Flag		C.CPCKP Number of Checkpoints		52
W.CPOAE	C.CPREQ Req Flag		A	B	Relative Address of Tape Label Information		C.CPOAE Equipment Assigned		53
W.CPVRNO	VSN Assignment 66x VSN Type-in								54
W.CPLDR1	C.CPLW C.CPLT Loader Flags		Global Library						55
W.CPLDR2	Set								56
W.CPLDR3	Indicators								57
W.CPAR	RA+1 Contents (and Control Point Number of Last Auto-recall Request)				C.CPAR		Reply Word Address		60
W.CPIOQ W.CPTAPE W.CPSTG	C.CPTMT MT		C.CPTNT NT		C.CPIOQ MST/PFC of Input File		C.CPMNT MT Max NT Max		61
W.CPDFMC W.CPDV W.CPDSMO W.CPIRB	C.CPDFMC Dayfile Message Count		C.CPDSMO Default Set MST Ordinal		C.CPRBID INTERCOM Batch Routing ID		C.CPDV Job Dependency ID		62
W.CPFP W.CPOUT W.CPFLAG W.CPERT	C.CPFLAG Flags				C.CPFST FST Address		C.CPFP C.CPOUT Flags		63
W.CPMSLM	C.CPMSLM MS Limit in PRU's			C.CPMSMX Maximum PRU Count		C.CPMSRC Running PRU Count			64
W.CHTIM	Channel Time as Number of Seconds*4096								65
W.CPMSI	C.CPSITM Time of Swap-In								66
W.CPSR	C.CPSCPT System CP Count		C.CPSR Stack Requests		C.CPSCPL Long-term Connections		C.CPSCPA Wait-response Connections		67
W.CPCAF	Start								70
W.CPCAL W.CPINS	End								167 170
	Reserved for Installations								177

A S.YNRDY B S.YNNO C Extended label format

NOTES: CONTROL POINT AREA

W.CPLINK(20)		C.CPSTAT	
Bit	0	S.CPUSTM	Move flag – move in progress
	1	S.CPUSTY	Auto recall
	2	S.CPUSTA	CPUA assigned only
	3	S.CPUSTB	CPUB assigned only
	4	S.CPUSTX	Recall status
	5	S.CPUSTW	Wait status
	6	S.CPUSTR	Real time job
	7	S.CPUSTC	Active CPUA
	8	S.CPUSTD	Active CPUB
	9	S.CPUSTS	Control point activity suspended
	10	S.CPUSTP	Suspended by check point
	11	S.CPUSTZ	Suspended by MTR (too many PP calls)

W.CPEF(24)		C.CPEF Values:	
	0001	F.ERTL	CP time limit exceeded, sensed by MTR
	0002	F.ERAR	Arithmetic error, sensed by MTR
	0003	F.ERPP	PPU 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 (batch job only)
	0010	F.ERRN	Operator rerun (batch job only)
	0011	F.EREX	Control card error, set by 1AJ
		F.ERCC	Control card error for INTERCOM job
	0012	F.ERECF	ECS parity error, sensed by MTR
	0013	F.ERJC	Job card error
	0014	F.ERPA	Pre-abort (batch job only)
	0015	F.ERRCL	Auto-recall error, bad PP call
	0016	F.ERHANG	Job hung in auto-recall
	0017	F.ERMSL	Mass storage limit exceeded (batch job only) by stack processor
	0020	F.EROVL	PP overlay not in PP LIB
	0021	F.ERIOD	I/O time limit exceeded, sensed by MTR
	0022	F.ERRMS	Dayfile lost on idled device
	0061	F.ERPARF	Swap-in parity error for graphics
	-77(7700)	F.ERMOMO	Enter MEMO Mode
	-0(7777)	F.ERTMM	Terminate MEMO Mode

W.CPSWP(41)		C.CPORG Values:
Value	4	Real time
	10	Graphics
	20	Multi-user
	40	INTERCOM
Bit	6	Swap out event bit

W.CPSCH(42)		C.CPFLG Values:	
Bit	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.CPFRFL	Storage request
	56	S.CPPROP	Roll out in progress

NOTES: CONTROL POINT AREA (CONT'D)

W.CPSCH(42) C.CPFLG Values: (continued)

Bit	57	S.CPS1P	Swap in in progress
	58	S.CPSOP	Swap out in progress
	59	S.CPSWC	Swap out complete

W.CPFACT(50)

Bit	11-0	If nonzero, permanent file accounting messages are issued
	59-12	Left-justified account number

W.CPCKP(52)

Bit	0	C.CPCON	Console checkpoint request
-----	---	---------	----------------------------

W.CPLDR1(55)

Bit	0	C.CPLW	Reserved for installations
	1	S.CPLP	Program loaded from nonsystem library
	2-3		Reserved for CDC
	4	S.CPLT	Debugging aid flag
	5	S.CPLR	Reduce flag
	6-9	S.CPLM	Map options
	10		Reserved for CDC
	11	S.CPLV	Map options validity flag

W.CPLDR1(55)

W.CPLDR2(56)

W.CPLDR3(57) Global Library Set Indicators:

00	End of global library set
01-76	LNT ordinal of system library
77	lfn of first user library in W.CPLDR3; lfn of second user library in W.CPLDR2

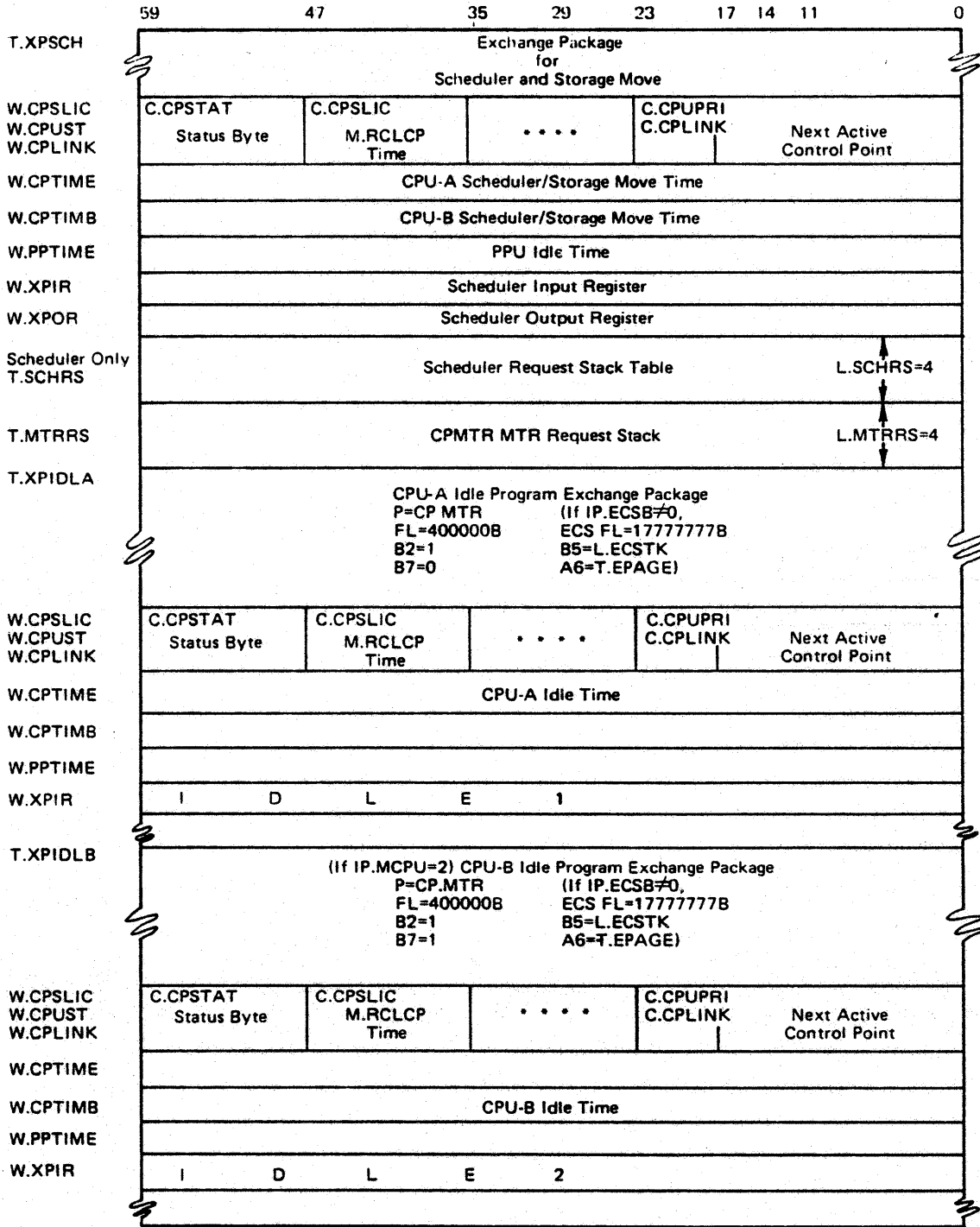
W.CPFLAG(63)

Bit	0	C.CPFLAG	MDI interlock
	1-2	S.CPLDAF	Reserved
	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 time limit previously set
	7	S.CPXTS	Look for next EXIT(S) card
	8	S.CPDMPX	Give no DMPX
	9-11	Reserved	

W.CPFP(63)

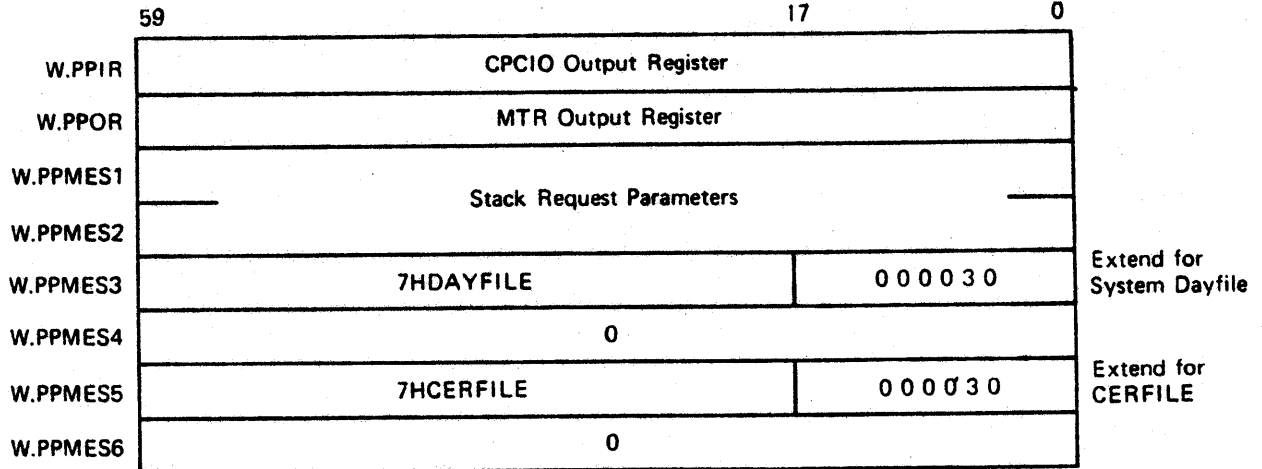
Bit	0	C.CPFP	Reprocess
	1	S.CPL	Abort
	2	S.CPG	No rerun
	3	S.CPA	Sequencer
	4	S.CPS	Checkpoint taken
	5	S.CPN	Look for EXIT card
	6	S.CPX	Private disk pack
	7	S.CPDP	Control card EOR
	8	S.CPEOR	Job card field length assigned
	9	S.CPJFL	JANUS
	10	S.CPJ	Remote Batch
	11	S.CPR	Intercom

SYSTEM JOB EXCHANGE PACKAGE AREA

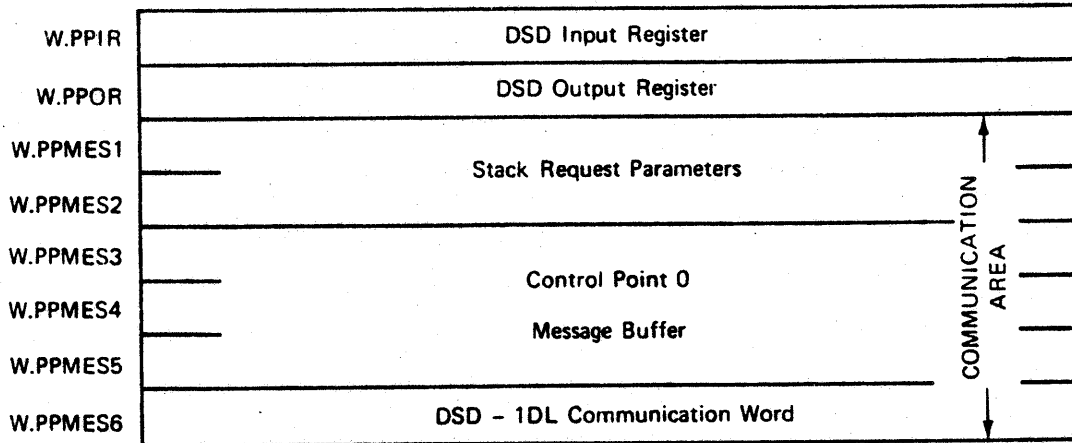


PP COMMUNICATION AREA

FOR PP0

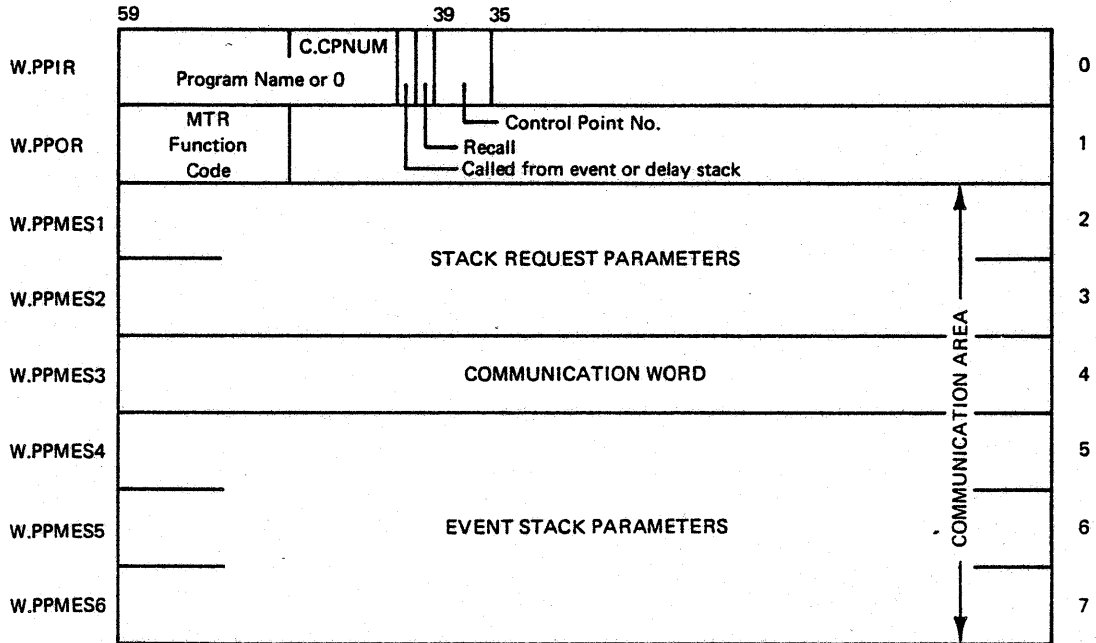


FOR PP1

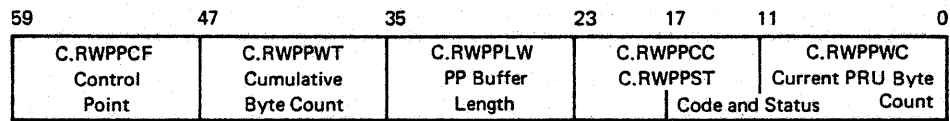


PP COMMUNICATION AREA (CONT'D)

FOR PP2 THROUGH PPn



COMMUNICATION WORD



PP PROGRAM NAME RESERVATIONS

Routine Name	Description
A†	Stack processor segment
ABS	Dump CM – absolute address
ABT	Program abort
ACE	Advance control card
ACT	Helper for program ACCOUNT
ADS	ADDSET processor – add member to PF set
APR	C.E. diagnostic sequencer
CEA	Deadstart PPO save program
CED	Deadstart PP control program
CEJ	MTS coldstart bootstrap
CEM	Central error manager for ECS
CEY	MTS coldstart bootstrap
CIO	Circular I/O processor
CKP	Saves information necessary to restart a checkpoint job
CLO	Dummy program used to call CIO
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
CY1	Resets FNT of file being processed by restart
D†	Deadstart dump
DF4	C.E. 3234 test
DF7	C.E. 3553 test
DF8	C.E. 808 test
DIS	Console display program for a control point
DLE	C.E. diagnostics
DLM	DELSET processor – delete member from PF set
DMP	Dump CM
DMT	Deadstart dump for 60x, 65x drivers
DSD	System display
DSM	Dismount pack
DSP	ROUTE/DISPOSE function processor
DTS	Deadstart dump for 66x drivers
D00	Diagnostic for COBOL
D03	C.E. 6603 test
D21	C.E. 821 test
D41	C.E. 841 test
D44	C.E. 844 test
END	Normal termination
EPF	Send audit information to CM

† Zero-filled

PP PROGRAM NAME RESERVATIONS (CONT'D)

Routine Name	Description
FAD	INTERCOM
FNT	INTERCOM FNT alter routine
FSN	Find set name
FTP	C.E. 580 printer test
GBJ	INTERCOM 274 Graphics begin job
GEJ	INTERCOM 274 Graphics end job
GES	INTERCOM 274 IGS SIGNON service program
GPF	GETPF(MMF)
HDS	Help deadstart
IAP	INTERCOM initiate another program
IEF	Routine for CEFAP
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 dump
LDC	LDCMR utility helper
LDL	Loader utility program
LDV	Loads CPU absolute overlays
LDW	Loads CPU absolute overlays in conjunction with LDV
LOC	Load octal corrections
LPF	In conjunction with LOADPF, reloads permanent files
LPT	C.E. 501 line printer test
LP1	C.E. 512 line printer test
MAC	INTERCOM
MDI	Used by EDITLIB to handle I/O involved in changing and moving directory
MDR	Deadstart 66x driver
MEM	Process memory function
MES	INTERCOM writes messages to remote terminal
MNT	MOUNT processor
MSD	Direct Access Module of Record Manager
MSG	Issues dayfile messages
MTR	Monitor
MTT	C.E. 60X tape test
MTZ	C.E. 66X tape drive test
MUJ	INTERCOM multi-user job
M71	C.E. 6671 multiplexor test

PP PROGRAM NAME RESERVATIONS (CONT'D)

Routine Name	Description
NSV	PP helper for CPVSN processor
OPE	Dummy program used to call CIO
OUX	TRANSPF and DUMPF utility helper
P†	Deadstart preaddressed 6603
PAK	Disk pack management routine
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
PFV	Permanent file manager PURGE function
PFR	Permanent file manager RENAME function
PFS	Permanent file manager POSITION function
PPI	Reserved
PRM	Permission checking function
QAC	I/O queue acquire file
QAF	Queue Access Function
QAJ	Reserved
RCL	Temporarily relinquish CPU
REQ	Makes nonallocatable device assignment and formats FNT entries for allocatable devices in response to REQUEST control card or a REQUEST macro call
RMS	Routine for CERMS
RPV	Reprieve central program
RST	Restores control point area of restart job
RWE	INTERCOM checks for INTERCOM job
SAC	ECS Segment Activity counts
SLT	Reserved
SPF	SAVEPF(MMF)
SPY	Count P-register samples for CP programs
SPZ	PP helper for SPY
SRB	Used by EDITLIB to complete the disk address of a record
SSC	Sub-system call
SSF	Sub-system function
SSH	Station system helper
STD	Enhanced station channel coupler driver
STL	Deadstart system execution PP resident
STS	Used by CP program to obtain certain status
TAT	PF set table system access
TBL	INTERCOM get table
TDR	Deadstart MT-NT tape driver
TDS	Terminate deadstart
TIM	Get current time, date, etc.

† Zero-filled

PP PROGRAM NAME RESERVATIONS (CONT'D)

Routine Name	Description
TMT	Table maintenance helper
T5X	C.E. 65X tape drive test
T6X	C.E. 66X tape drive test
T76	INTERCOM
Uxx	Reserved for installations
VEJ	Verify job statement
VSM	STIMULATOR routine
XDQ	PP portion of dump queue
nUx	Reserved for installations
0FA-0FZ	INTERCOM Front End Initializer
0F0-0F9	INTERCOM Front End 2550 software
0ZA-0ZS	LCC drivers
0ZT-0ZZ	LCC initializer
0Z1-0Z9	LCC drivers
1AB	Identifies recovered jobs
1AJ	Advance job
1BO	Asynchronous job terminator
1BR	INTERCOM Buffer Manager
1BT	Blank label tape routine
1CC	Enhanced station CIO overlay
1CI	INTERCOM Queue Manager
1CL	Close function for all non-tape or non-permanent files
1CR	Tape read recovery – write CM for 9-track tapes
1CS	Tape read recovery – write CM for S tapes
1CT	Tape read recovery – write CM for SCOPE tapes
1C9	Write CM for tape read recovery – NT SCOPE tapes
1DD	System dynamic dump processor
1DF	Dump dayfile
1DL	Overlay loader and dayfile message processor for DSD
1DM	Device queue manager
1DS	INTERCOM H-display
1DU	DUMPF initialization
1EJ	End of job processor
1EV	Off line evict processor
1FC	Creates an RB entry for PFC
1FE	INTERCOM Front End Driver
1FM	Dummy film/hardcopy processor
1GJ	INTERCOM
1GM	Issues GOOD MORNING when time changes from 23.59 to 00.00
1GR	INTERCOM
1GS	INTERCOM 274 IGS SIGNON initializer
1IB	Initiate batch job from input queue
1ID	INTERCOM send dayfile message to terminal, complete swap
1IM	INTERCOM send message to terminal

PP PROGRAM NAME RESERVATIONS (CONT'D)

Routine Name	Description
1IQ	Initiate JANUS control point
1IR	Main JANUS routine; drives readers punches, printers, etc.
1IS	Initialize overlay setup
1IT	Integrated tape driver for (66x) main overlay
1IU	Called by JANUS to backspace print file
1I1	INTERCOM Initialization
1LX	INTERCOM
1MF	Multifile positioning routine
1MH	Tape scheduling/prescheduling routine
1MM	Multi-mainframe job queue manager
1MN	MNT overlay, PFD cleanup after compatibility mode used
1MT	Long record stranger tape driver
1NO	Tape read recovery noise record verifier
1NR	NT read driver
1NS	Notify station of SPOT completion
1NW	NT write driver
1N2	Tape noise record read recovery, read forward 1
1N3	Tape noise record read recovery, read forward 2
1OP	File open routine for non-tape files
1PC	Close permanent file mass storage
1PD	Called by PFA to either enter event stack, call another PP routine, or swap out
1PF	Permanent file error recovery
1PG	PURGE(MMF)
1PK	PF set coordinator
1PL	Dummy plot program
1P1	Tape recovery to LGR positioning driver
1P2	Tape recovery write driver
1P3	Tape recovery verification driver
1P4	Tape recovery to LGNR positioning driver
1QF	I/O file manager
1QM	INTERCOM check for MUJ swap-out completion
1QP	INTERCOM quantum calculator and MUJ servicer
1RC	Restores field length of a checkpointed job
1RN	Ages queues, manages RBT chains and statuses tape drives
1RP	End of reel processor
1RS	Read stranger tape driver
1RT	Read SCOPE tape driver
1RV	Tape I/O read recovery driver initializer and terminator
1R2	Tape read recovery - tape parity error recovery 1
1R3	Tape read recovery - tape parity error recovery 2
1R9	SCOPE tape 9 track (659) read tape driver
1SI	Routine to swap-in or roll-in a job
1SO	Swap-out or roll-out a job
1SP	Mass Storage I/O processor (stack processor)
1SX	Error message and abort function for stack processors
1S5	Load and execute 1SP or 3DO at second entry

PP PROGRAM NAME RESERVATIONS (CONT'D)

Routine Name	Description
1TF	Tape forward motion routine
1TJ	Translate job card
1TO	Tape open routine
1TR	Read labels for non-66X drives
1TS	Tape sampler
1VG	STIMULATOR routine
1WB	INTERCOM wideband driver
1WI	SCOPE internal tape write driver
1WS	Stranger tape write driver
1W9	SCOPE tape 9-track (659) write tape driver
1XG	INTERCOM 1XP overlay used for graphics
1XP-6XP	INTERCOM high speed EXPORT processor
1ZA-1Z9	INTERCOM drivers
2CC	1CI overlay -- process command
2CS	1CI overlay -- create user table
2CU	1CI overlay -- create user table
2FC	1FC overlay replace mode
2FE	INTERCOM Front End Driver
2GJ	INTERCOM
2IA	66x read driver for L tapes
2IB	66x write driver for L tapes
2IC	66x read driver for 7-track coded SCOPE tapes
2ID	66x write driver for 7-track coded SCOPE tapes
2IL	66x 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
2IW	66x basic write overlay
2II	INTERCOM overlay to 1I1
2LF	LPF overlay broken connect
2MN	MNT overlay error processing
2PA	PFA utility processor
2RP	Overlay to 1RP-End-of-reel processor
2ST	MMF CIO staging processor
2TB	All backward tape motion
2TC	Extended trailer label group processor
2VJ	Translate job card
2WB	INTERCOM overlay to 1WB
2XP	INTERCOM wideband directive processor
3AM	ADS overlay member processing
3DO	Initialize allocatable device file
3FC	1FC overlay compatibility replacement
3FE	INTERCOM Front End Driver

PP PROGRAM NAME RESERVATIONS (CONT'D)

Routine Name	Description
3IC	66x close processor
3IE	66x basic error processor
3IF	66x multi-file processor
3II	66x system initialization
3IL	66x label write processor
3IM	66x message processor
3IO	66x open processor
3IP	66x positioning within a logical file
3IR	66x read error recovery
3IS	JANUS overlay
3IV	66x close volume processor
3IW	66x write error recovery
3LX	INTERCOM overlay to 1LX
3MG	Format request for REQ
3MN	REQ overlay for tape assignments
3PA	PFA swapper status check segment
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
3RQ	REQ overlay containing 2TACOM
3R2	Process mispositioning at load point for 1R2
3R3	Process mispositioning at load point or hardware errors for 1R3
3SP	Stack processor for 6603-I driver
3SQ	Stack processor for 6638 driver
3SS	Stack processor for 854 driver
3ST	Stack processor for 6603-II driver
3SV	Stack processor for 821 driver
3SW	Stack processor for 841 driver
3SY	Stack processor for 844 driver
3TT	INTERCOM transmit data from CPU to terminal
3T1-3T2	INTERCOM overlays to 3TT
3WB	INTERCOM overlay to 1WB
3XP	INTERCOM wideband batch input processor
4AM	ADSETT add number overlay
4DO	Process device independent requests for allocatable devices
4ES	Enter stack request
4FC	1FC overlay compatibility replacement
4FE	INTERCOM Front End Driver
4LB	ANSI standard label processor
4LC	3000 label processor
4LX	INTERCOM overlay to 1LX
4PA	PFA segment backspace pru
4SD	Enhanced station channel coupler driver overlay to STD
4WB	INTERCOM overlay to 1WB
4XP	INTERCOM wideband batch output processor

PP PROGRAM NAME RESERVATIONS (CONT'D)

Routine Name	Description
5CP	IRP overlay 6603-I driver
5CQ	IRP overlay 6638 driver
5CS	IRP overlay 854 driver
5CT	IRP overlay 6603-II driver
5CV	IRP overlay 821 driver
5CW	IRP overlay 841 driver
5CY	IRP overlay 844 driver
5FE	INTERCOM Front End Driver
5LX	INTERCOM overlay to 1LX
5PA	PFA segment error subroutine
5WB	INTERCOM overlay to 1WB
5XP	INTERCOM wideband batch banner page
6BM	Billing message overlay
6BR	ANSI label processor read function code overlay
6BW	4LB overlay
6CR	3000 label processor read function code overlay
6CW	3000 label processor write function code overlay
6FE	INTERCOM Front End Driver
6IM	Issue conflict and abort messages for REQ
6LC	Segment of 4LB or 4LC to load conversion table into MMTC
6LM	Segment of 4LB used to construct tape label messages
6LX	INTERCOM overlay to 1LX
6L1	4LB overlay to convert PRU count
6L2	BCD conversion table overlay for 4LB
6L3	4LB overlay to check that proper conversion table is in the MMTC
6L4	4LB overlay for debug message writer
6L5	4LB overlay to format the label information
6L7	4LB overlay to pack and write label to tapes table
6MD	Dummy EDITLIB overlay
6MN	REQ overlay for tape assignments
6NO	Tape error recovery debug segment assembled to give more detail about segment being read by 1NO
6PA	PFA segment-compatibility overlay
6PD	Write pre-dayfile
6RD	Disposed file accounting overlay
6PM	Permanent file accounting overlay
6SI	Process swap-in parity errors
6WM	Outputs dayfile error messages for I/O requests
6XP	INTERCOM wideband batch lace card
7AJ	1AJ overlay for EXIT card processing
7CC	Enhanced station CIO overlay to 1CC
7EC	Generate ECS buffers
7FE	INTERCOM Front End Driver
7ID	Auxiliary error processor for RMS I/O
7NO	Debug routine
7RQ	REQ Set Processor
7T1	ASCII/Display code conversion table
7T2	EBCDIC/Display code conversion table
7W1-7W2	Overlay for 6WM

PP PROGRAM NAME RESERVATIONS (CONT'D)

Routine Name	Description
8AA-8A9	Reserved
8BA-8B9	Reserved
8CA-8C9	C.E. reserved names
8DA	A, I, J display overlay for DSD (dayfile buffers, REQUEST cards, 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)
8DU	U display overlay for DSD (ID table)
8DV	V display overlay for DSD (RMS)
8DW	W display overlay for DSD (pack requests)
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 Front End 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-8PS	Reserved
8NO	Segment to 1N3 that writes debug messages to dayfile if IP.DBUG=1
8PU-8W9	Reserved
8T3	Overlay to load MMTC memory
8XA	Channel commands overlay for DSD
8XB	Debugging commands overlay for DSD
8XC	PPU 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
8XG	Priority and tape staging job control commands overlay for DSD
8XH	INTERCOM commands for DSD
8XI	Miscellaneous commands overlay for DSD

PP PROGRAM NAME RESERVATIONS (CONT'D)

Routine Name	Description
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 Front End Driver
9FF-9PS	Customer Engineering
9PU-9Y9	Customer Engineering
9ZA-9Z9	INTERCOM

MONITOR FUNCTIONS

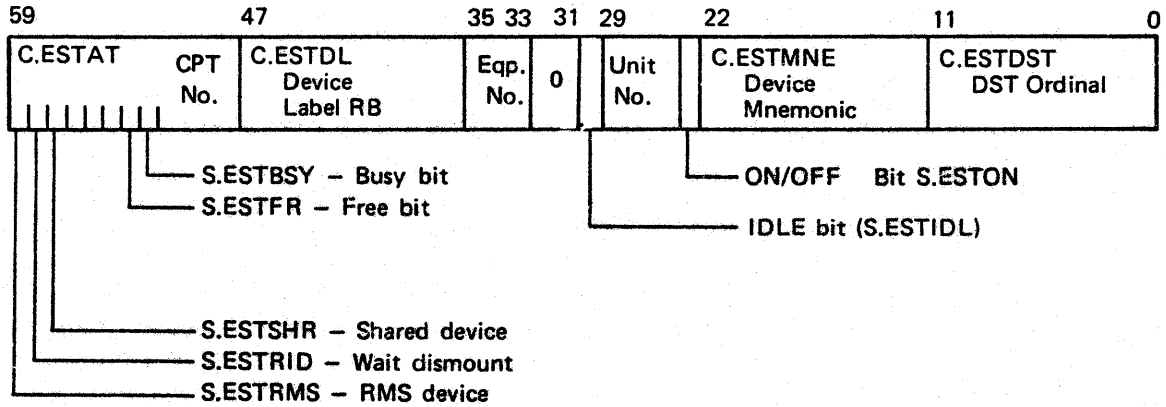
01	M.SETST	Set CPU status bits
02	M.CLRST	Clear CPU status bits
03	M.RCP	Request central processor
04	M.DCP	Drop central processor
05	M.RCLCP	Recall central processor
06	M.ICE/M.SPM	Initiate central executive/SPM call from 1SP
00	EX.CMSM	CM storage move
01	EX.ECSM	ECS storage move
02	EX.PLIB	PP library search
03	EX.SPM	Call stack processor manager
04	EX.SS	Systems second calculation
05	EX.SCH	Call scheduler
06	EX.SCH1	Call scheduler (storage request entry)
07	EX.REQEB	Request ECS buffer
10	EX.RELEB	Release ECS buffer
11	EX.CBM	Circular Buffer Manager
12	EX.SPRCU	Stack processor recall
13	EX.STAT	Change status
14	EX.NXTPB	Get next PB/PRU

MONITOR FUNCTIONS (CONT'D)

15	EX.FLHB	Flush buffer
16	EX.CSWAP	Clean ECS after ECS RPE in swap file
17	EX.AUTEB	Terminate automatic allocation
20	EX.ECD	Display ECS
21	EX.ECR	Release display
22	EX.ECW	Modify ECS
23	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.BOOT	Start ECS system
32	EX.TAT	Lock RBR/RBT processing
33	EX.RBT	PRU conversion
34	EX.SSF	Subsystem function
07	M.CPUST	Change CPU status (IP.MCPU \neq 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.DFM	Process dayfile message
15	M.STEP	Enter step mode
16	M.RBTSTO	Request RBT storage
17	M.RSTOR	Request storage
20	M.TSR	Terminate storage request (IP.RTMTR \neq 0)
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.SEF	Set error flag
27	M.ISP	Initiate stack processor
30		Not used
31	M.CCPA	Change control point assignment
32	M.RPJ	Request peripheral job
33	M.EES	Enter event stack
34	M.CPJ	Capture peripheral job
35	M.SCH	Initiate integrated scheduler
36	M.PASS	MTR ignores it - to be 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, cleared immediately. Used as break point
43	M.PPCH	Request channel surveillance
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

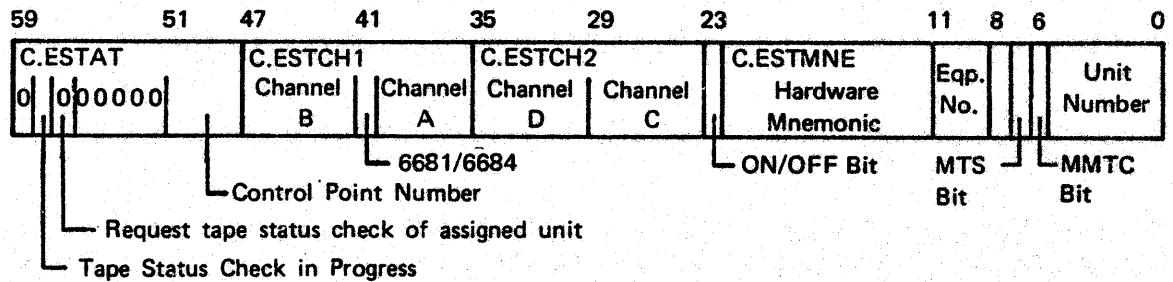
EST FOR RMS DEVICE



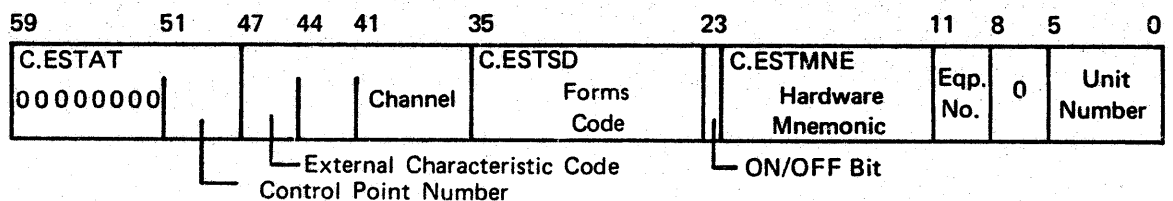
USE OF FREE, BUSY BITS

Free	Busy	Description
0	0	Unavailable device
1	0	Dismounted device
0	1	Mounted device
1	1	Device in process of being dismounted

MAGNETIC TAPE ENTRY

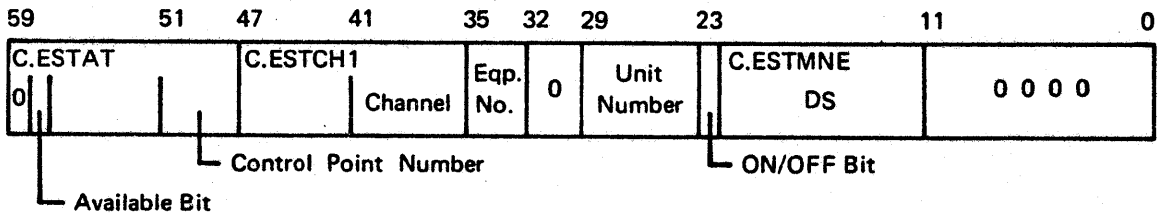


UNIT RECORD EQUIPMENT ENTRY

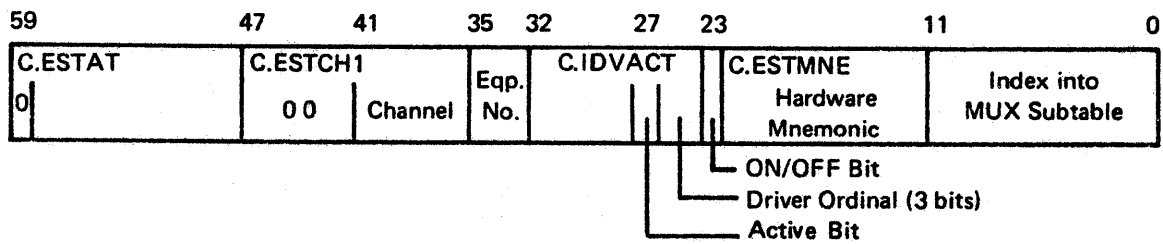


EQUIPMENT STATUS TABLE (CONT'D)

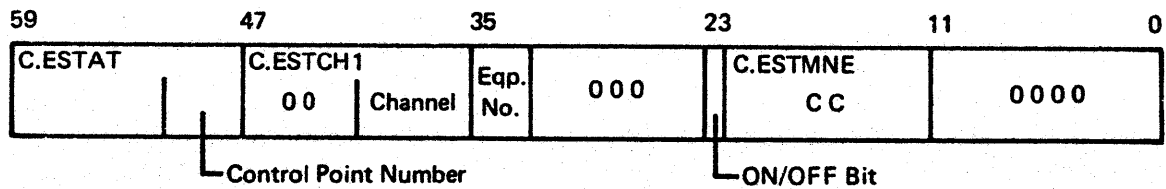
6612 DISPLAY CONSOLE ENTRY



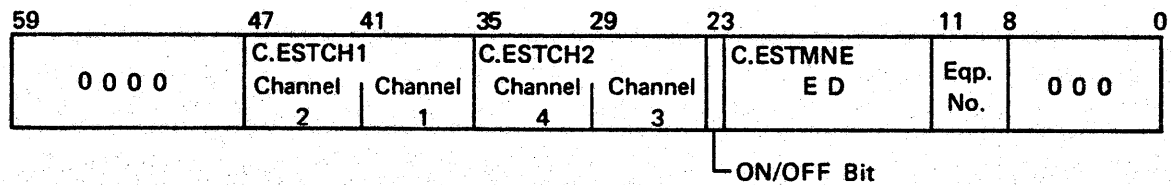
MULTIPLEXER ENTRY



6000/7000 CHANNEL COUPLER



DDP ENTRY



DEVICE CODES

Mnemonic	Device Type	Description
AA	01	6603-I disk
AB	02	6638 disk
--	03	CDC reserved
AC	04	6603-II disk
AL	05	821 data file
AM	06	841 multiple disk drive
AP	07	3234/854 disk pack drive
--	10	CDC reserved
--	11	
--	12	
AY	13	Reserved for 844 disk pack
--	14	CDC reserved
--	15	
--	16	
--	17	ECS resident file
AX*	20	
--	21	
--	22	CDC reserved
--	23	
--	24	
--	25	Link medium file Packet file FNT (MMF)
LM	26	
--	27	
--	30	Reserved for installations, RMS devices only
--	31	
--	32	
--	33	
--	34	
--	35	
--	36	
--	37	7-track magnetic tape†
MT	40 xx	
NT	41 xx	
--	42 xx	
--	43 xx	Member file 7-track tape†
--	43 xx	Member file 9-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
L1**	51	501, 505 line printer
LQ**	52	512 line printer
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	7-track multi-file set tape†
--	63 xx	9-track multi-file set tape†

†See following page

*No EST entry for this device code

DEVICE CODES (CONT'D)

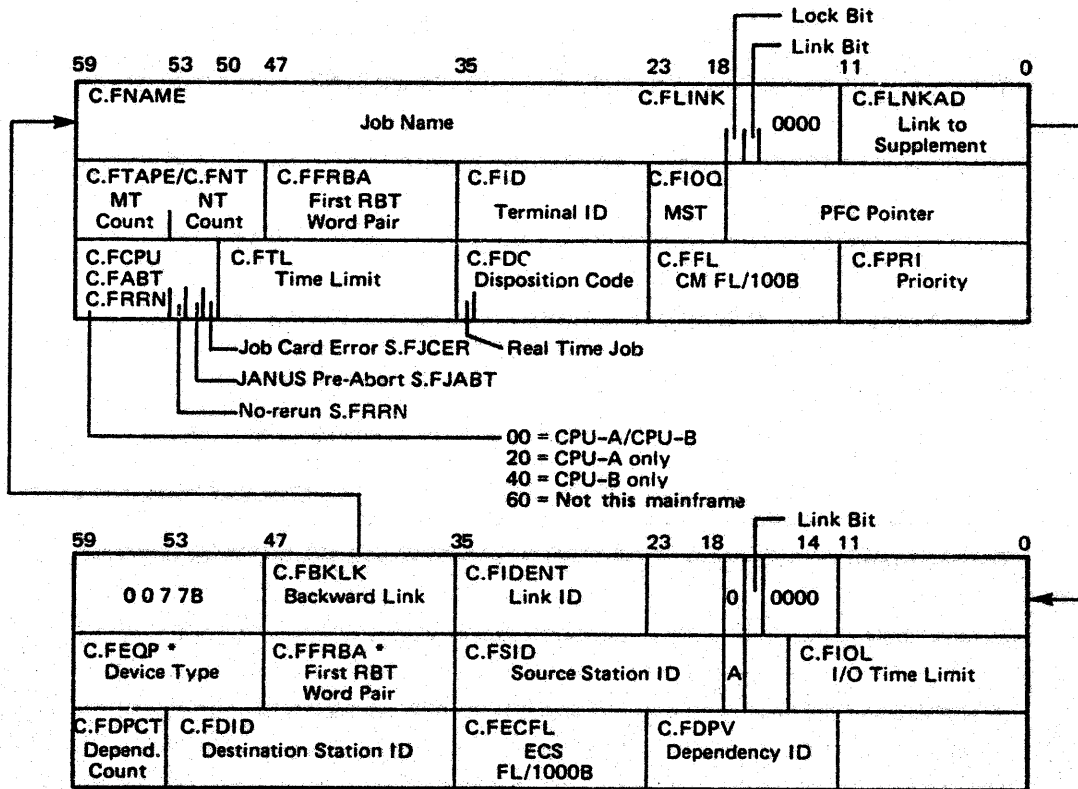
Mnemonic	Device Type	Description
--	64	Pseudo code for tape staging
--	65	CDC reserved
--	66	Reserved for installations
--	67	Reserved for installations
CP**	70	415 card punch
DS	71	6612 keyboard/display console
GC**	72	252-2 graphic console
HC**	73	253-2 hard copy recorder
FE	--	2550 Communications NPU
FM**	74	254-2 microfilm recorder
PL**	75	Plotter
--	76	Reserved for installations
--	77	Reserved for installations
DC	--	6671 DSC
IX	--	Reserved for installations
Wx	--	Reserved for installations
Xx	--	Reserved for installations
SC	--	6673/6674 DSC
YC	--	6676 DSC
CC	--	6000/7000 channel coupler
CS	--	7077-1 communications station (LCC)
ED	--	6642-1 distributive data path (DDP)

**Device type is defined but not supported by standard software.

† Low order 6 bits (xx)	7-track	9-track
----00	HI-density (556 bpi)	CDC-reserved
----01	LO density (200 bpi)	CDC-reserved
----10	HY density (800 bpi)	HD density (800 bpi)
----11	CDC-reserved	PE density (1600 bpi)
--00--	Unlabeled	Unlabeled
--01--	U- or Z-labeled	U- or Z-labeled
--10--	Y-labeled	Y-labeled
--11--	CDC-reserved	CDC-reserved
00----	SCOPE data format	SCOPE data format
01----	CDC-reserved	CDC-reserved
10----	S tape	S tape
11----	L tape	L tape

FILE NAME TABLE

ENTRY AND OPTIONAL SUPPLEMENT FOR FILE IN INPUT QUEUE



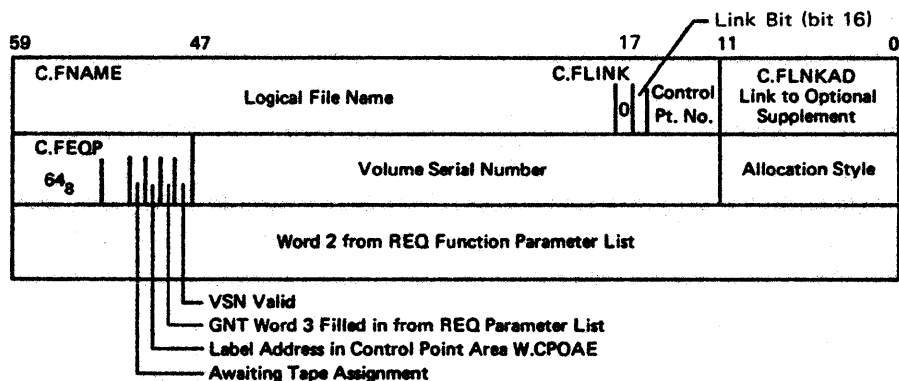
*If non-zero, fields apply to pre-dayfile

A = Keep bit

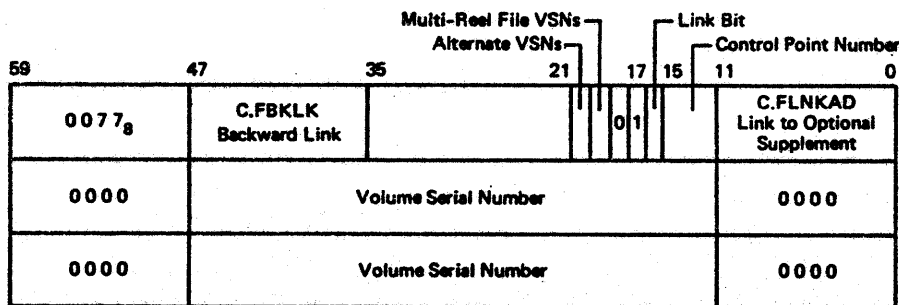
FILE NAME TABLE (CONT'D)

TAPE FILE ENTRIES

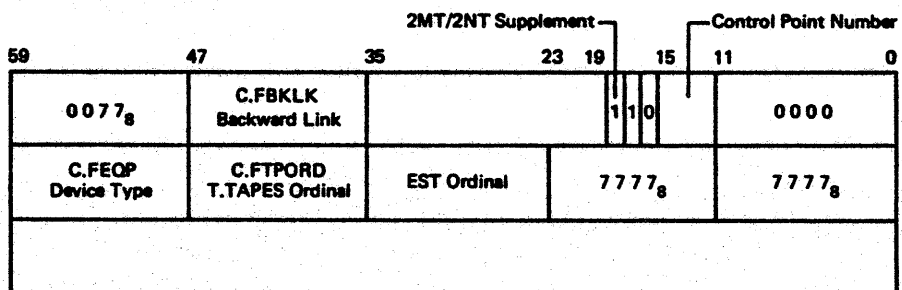
BEFORE EQUIPMENT ASSIGNMENT (GNT)



SUPPLEMENT(S) IF MORE THAN ONE VSN GIVEN

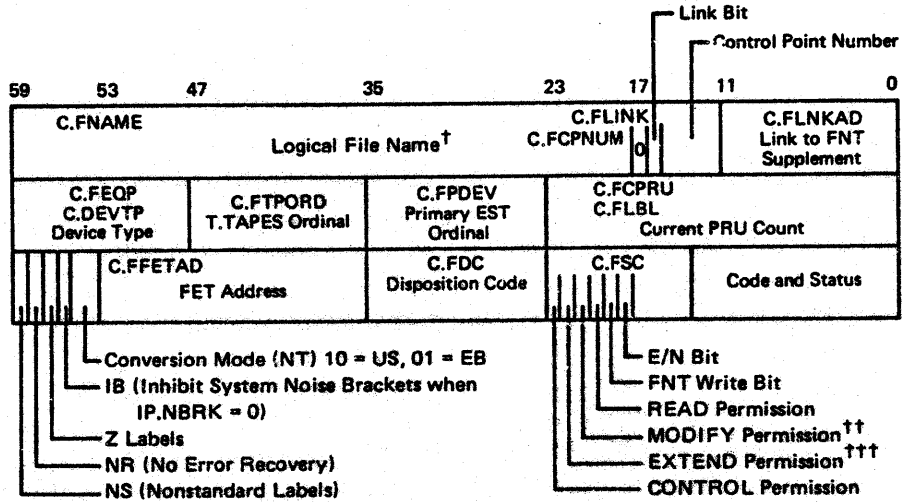


SUPPLEMENT IF 2MT/2NT DECLARED



FILE NAME TABLE (CONT'D)

TAPE FILE ENTRY DURING PROCESSING



†Multi-file name if multi-file set after assignment and before first POSMF.

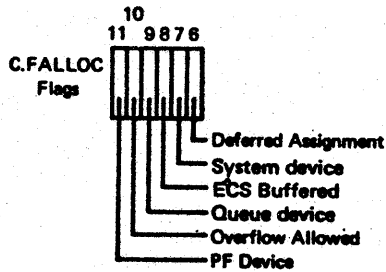
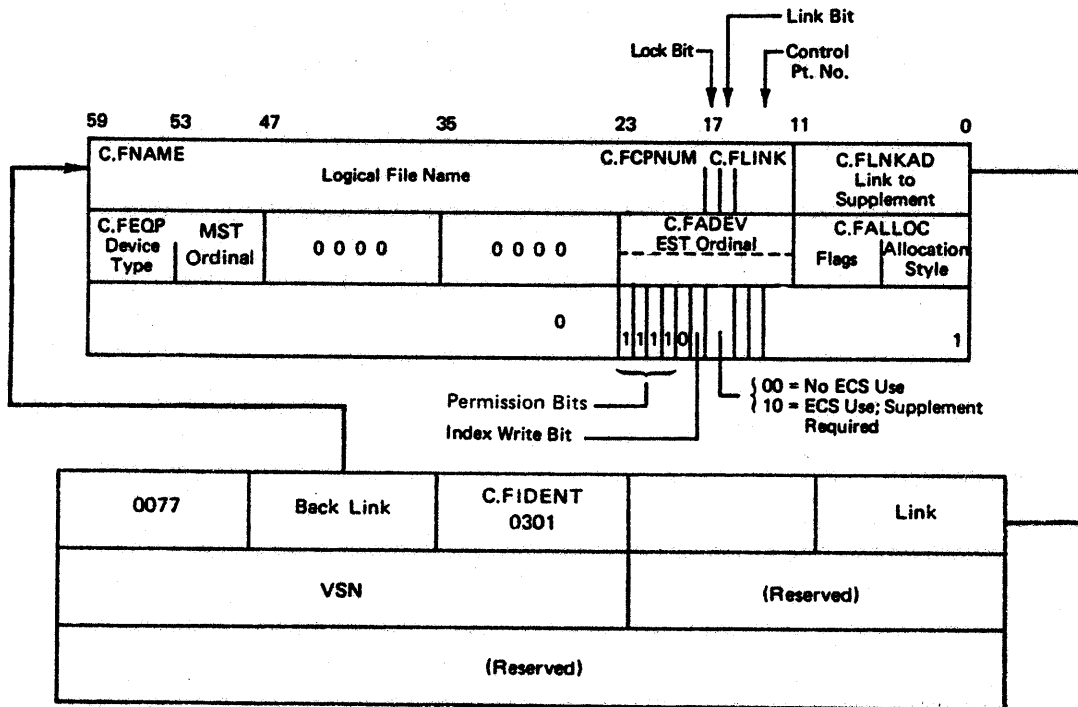
††NORING is not specified.

†††Expired tape label or operator override on unexpired tape label.

FILE NAME TABLE (CONT'D)

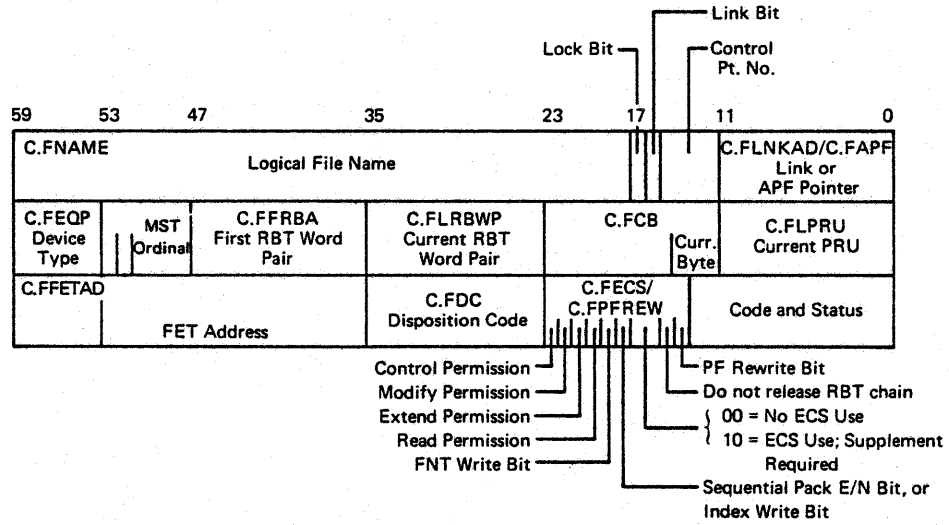
ENTRY FOR LOCAL RMS FILE

BEFORE ASSIGNMENT TO A DEVICE



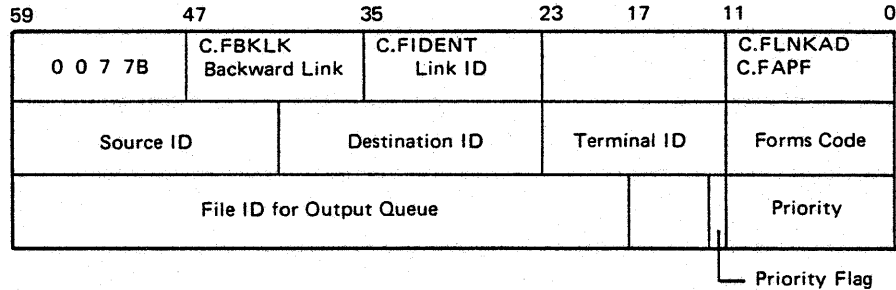
FILE NAME TABLE (CONT'D)

AFTER ASSIGNMENT TO A DEVICE AND THROUGHOUT PROCESSING

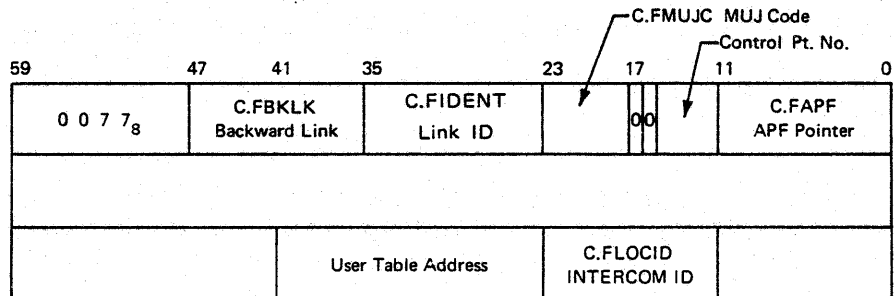


OPTIONAL SUPPLEMENTS FOR LOCAL RMS FILES

FILE ROUTING SUPPLEMENT

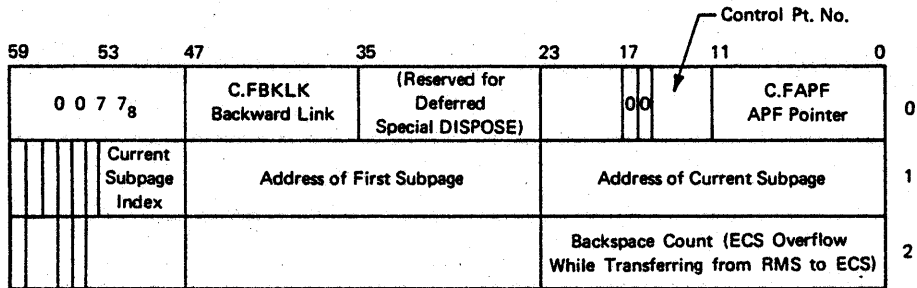


INTERCOM-USER FILE SUPPLEMENT (Required only when file to be attached to swapped-out job)



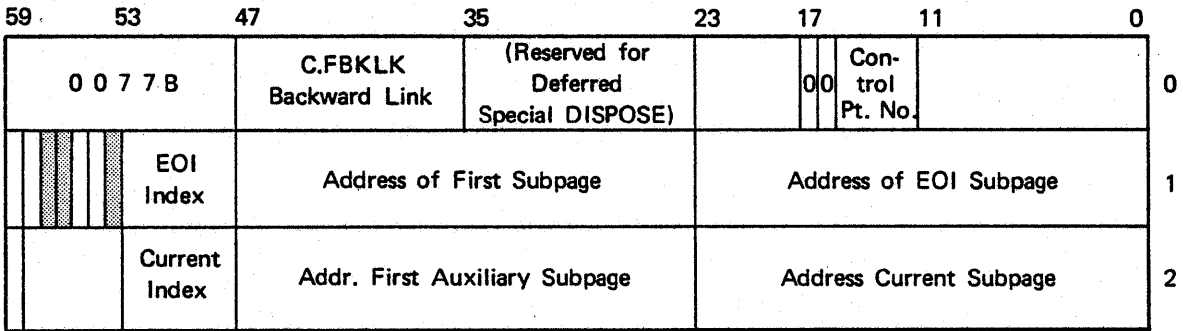
FILE NAME TABLE (CONT'D)

ECS FILE SUPPLEMENT (ECS resident files or IO buffers)



Word 1	Bit 54	Outstanding PPCIO Request	Word 2	Bit 55	Release ECS Buffer
	55	ECS Preallocation Flag		56	After the Current SR
	56	1 = Release Bit		56	Index Written (Close Random File)
	57	0 = Output Buffer		59	Transfer in Progress (ECS
		1 = Input Buffer			Resident Random Files)
	58	Buffer Overflow			
	59	1 = ECS-Buffered File			

ECS FILE SUPPLEMENT (ECS Resident Library)



Word 1	Bit 55	ECS Preallocation Flag
	56	1 = Release Bit
	59	1 = ECS Buffered File

Word 2	Bit 59	Transfer in Progress
--------	--------	----------------------

FILE NAME TABLE (CONT'D)

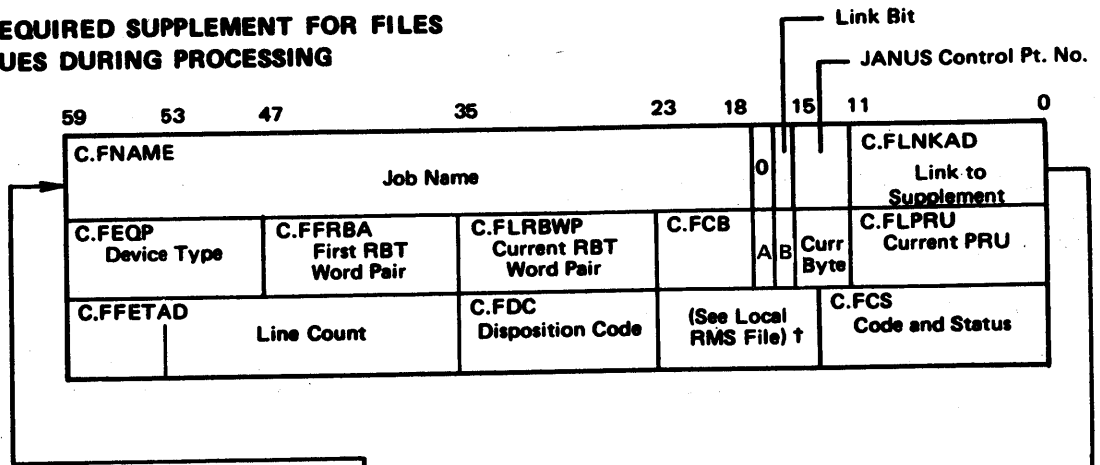
DEVICE SET ENTRY

One entry for each device set each job has mounted.

	59	47	35	23	17	15	11	0
W.FSN	A	C.FNAME Device Set Name			C.FCP		C.FMST MST Ordinal	
W.FVSN		C.FVSN Master Device VSN						
W.FORD			C.FEST EST ordinal of master device					

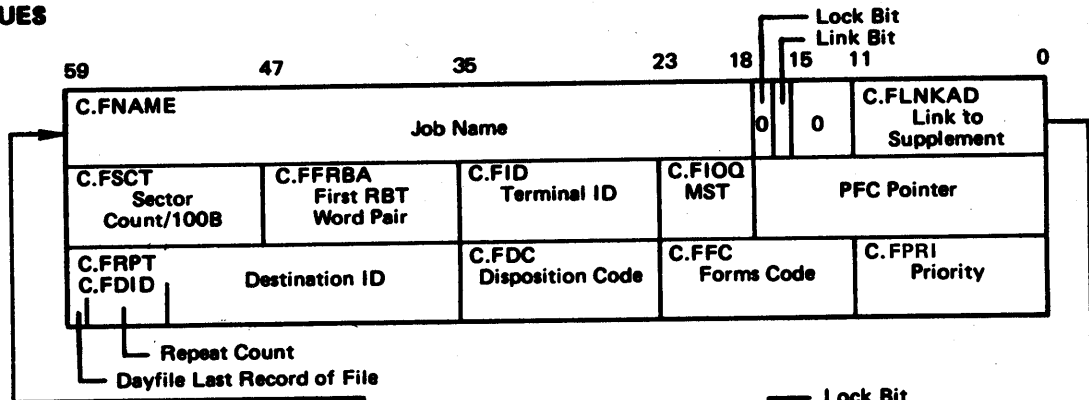
A = 1 set name flag (S.FSET)

**ENTRIES AND REQUIRED SUPPLEMENT FOR FILES
IN OUTPUT QUEUES DURING PROCESSING**



A = 1 if 8 lines per inch specified (S.F8LPI)
 B = 1 if automatic page eject specified (S.FAUTO)
 C = 1 if link FST1-FST2 contains information related to file routing

**ENTRIES AND OPTIONAL SUPPLEMENT FOR FILES
IN OUTPUT QUEUES**



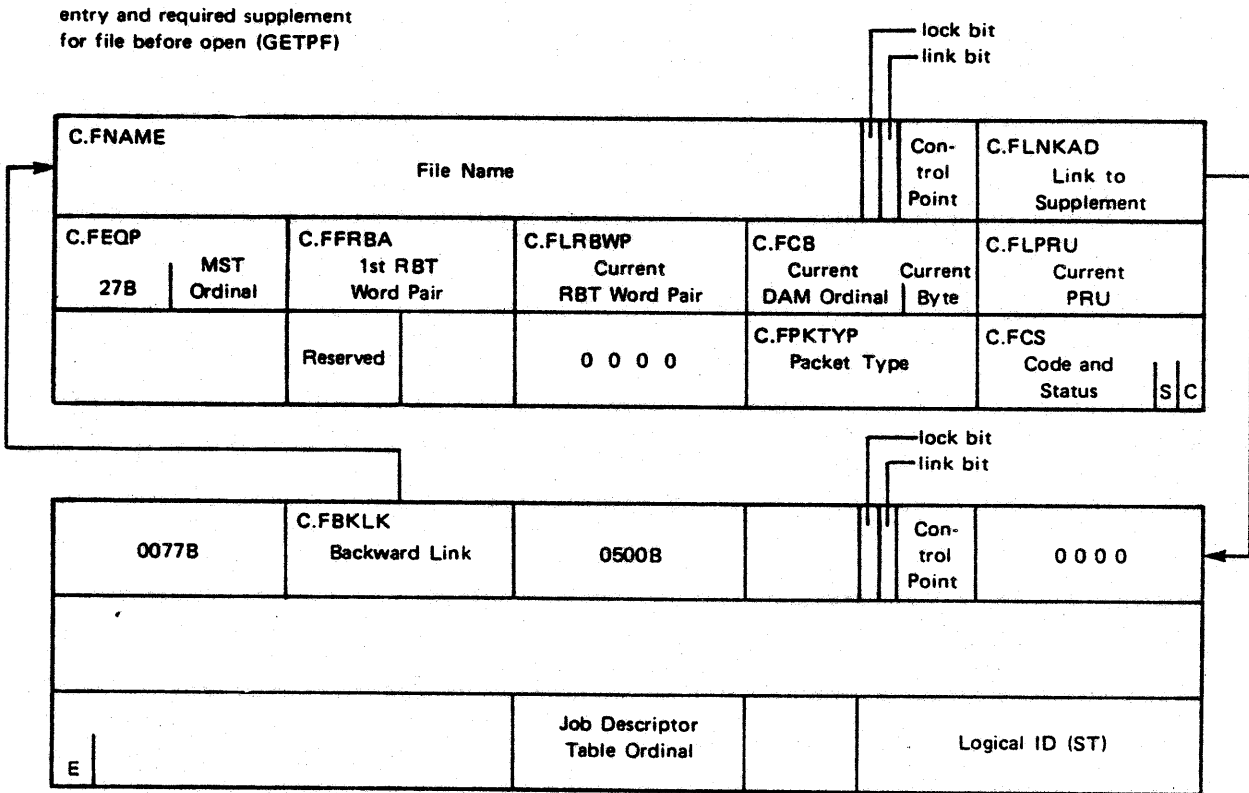
A = 1 if 8 lines per inch specified (S.F8LPI)
 B = 1 if automatic page eject specified (S.FAUTO)

Supplement will be present if file processing has been interrupted by abort or deadstart recovery.
 NOTE: File is not rewind.

FILE NAME TABLE (CON'T)

MMF PACKET FILE ENTRIES

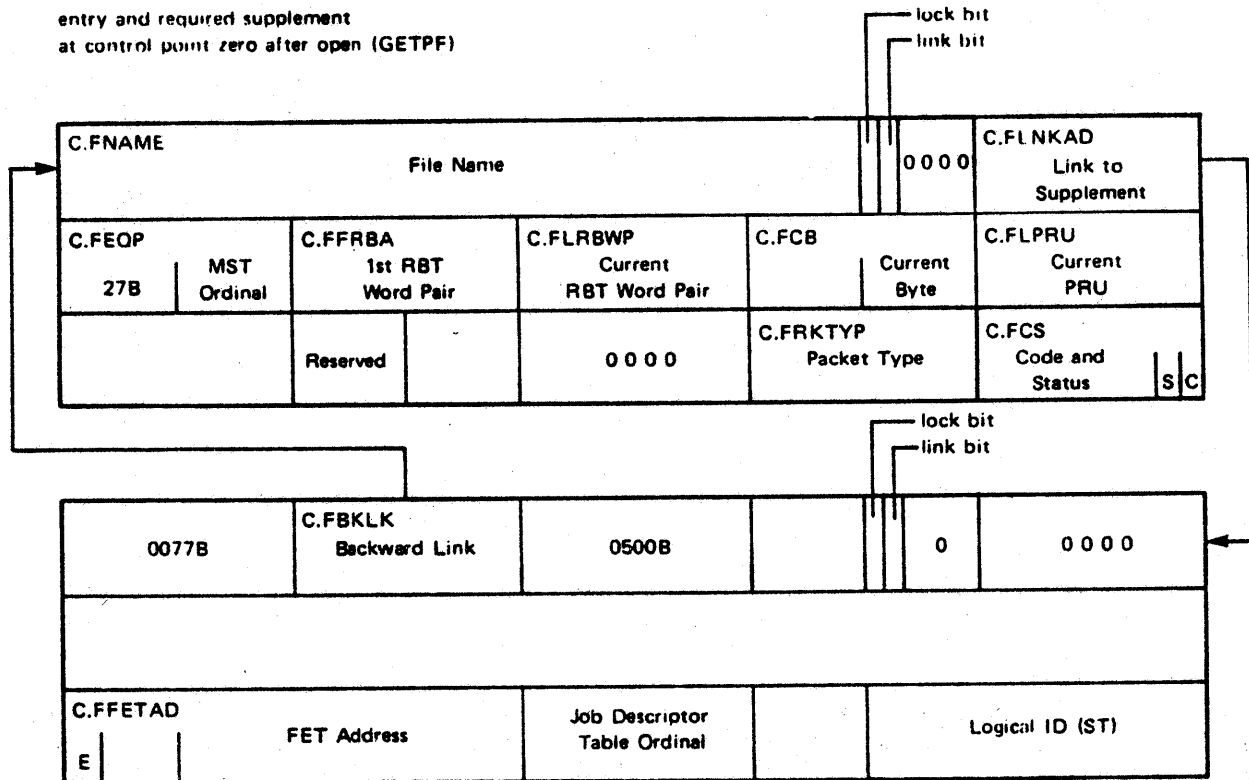
ENTRY AND REQUIRED SUPPLEMENT FOR FILE BEFORE OPEN (GETPF)



FILE NAME TABLE (CONT'D)

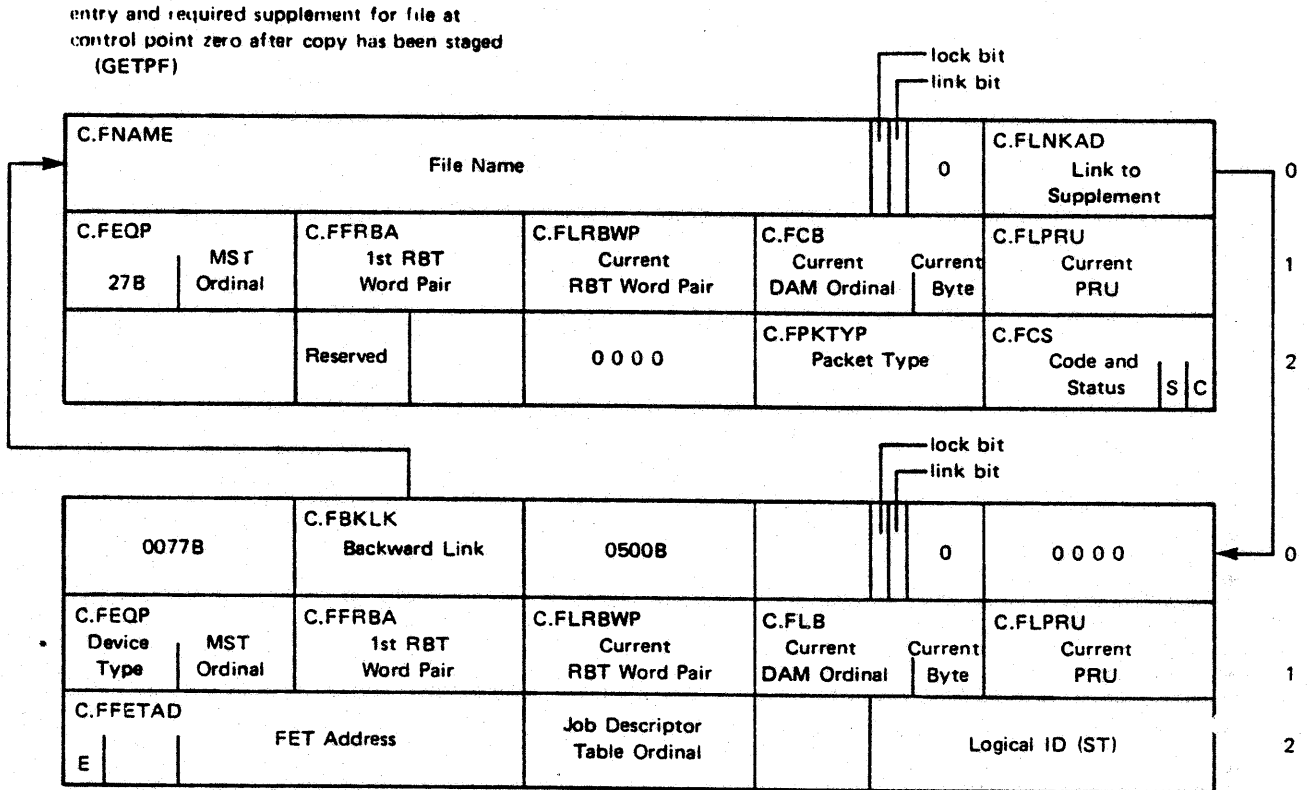
ENTRY AND REQUIRED SUPPLEMENT AT CONTROL POINT ZERO AFTER OPEN (GETPF)

entry and required supplement
at control point zero after open (GETPF)



FILE NAME TABLE (CONT'D)

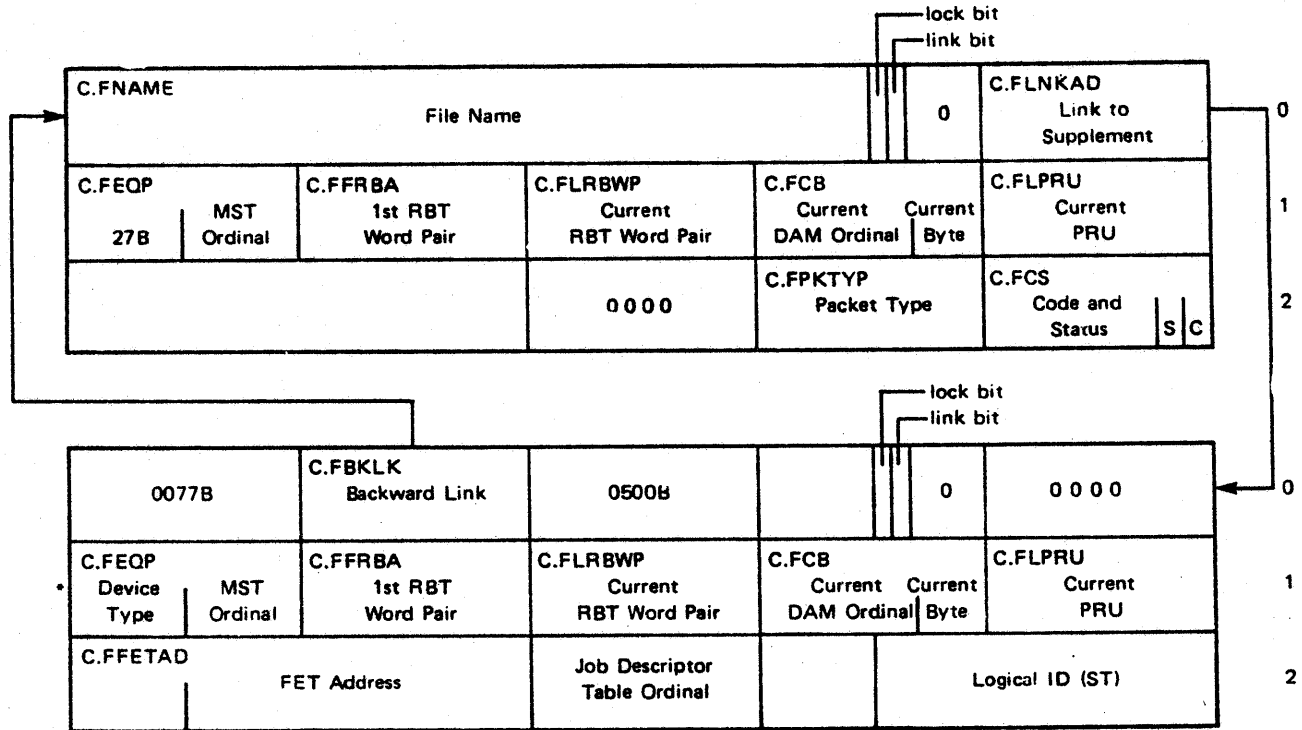
ENTRY AND REQUIRED SUPPLEMENT FOR FILE AT CONTROL POINT ZERO AFTER COPY HAS BEEN STAGED (GETPF)



*Actual file information is in word 1 of the supplement

FILE NAME TABLE (CONT'D)

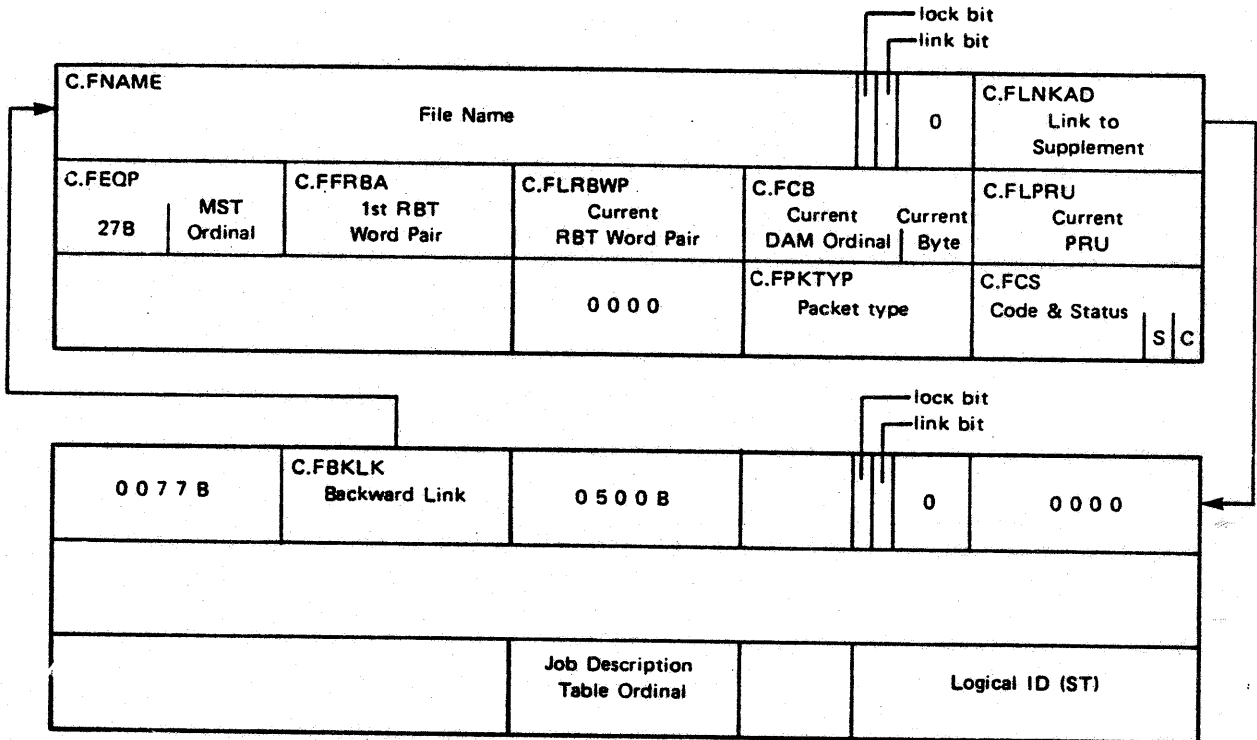
ENTRY AND REQUIRED SUPPLEMENT FOR FILE AT CONTROL POINT ZERO (SAVEPF)



*Actual file information is in word 1 of the supplement

FILE NAME TABLE (CONT'D)

ENTRY AND REQUIRED SUPPLEMENT FOR FILE AT CONTROL POINT ZERO (PURGE)



C – complete bit
E – the EC flag saved from the user 'FDB
S – Packet status bit

PACKET Type codes are: 0100 B ATTACH
0200 B CATALOG
0300 B PURGE

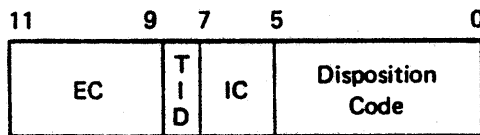
DISPOSITION CODE VALUES (C.FDC)

Non-Allocatable Devices

CK	xxx1	Checkpoint
IU	xxx2	Inhibit unload
CI	xxx3	Checkpoint and inhibit unload
SV	xxx4	Save
CS	xxx5	Checkpoint and save
	xxx6-7777	Reserved

Allocatable Devices

For allocatable devices, the byte C.FDC is divided into four fields:



Code	Value	Description
------	-------	-------------

EC (Bits 11-9) External Code

(Print/Punch)

Default	000	Default print train/Default punch character set
-- /EC=SB	001	Reserved/Punch SCOPE binary
EC=A4/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

TID (Bit 8) Terminal Identification

Relevant only for local files, not queue files.

TID=xy	0	Route file to remote user
TID=C	1	Ignore remote ID in file routing

xy = terminal identification

DISPOSITION CODE VALUES (C.FDC) (CONT'D)

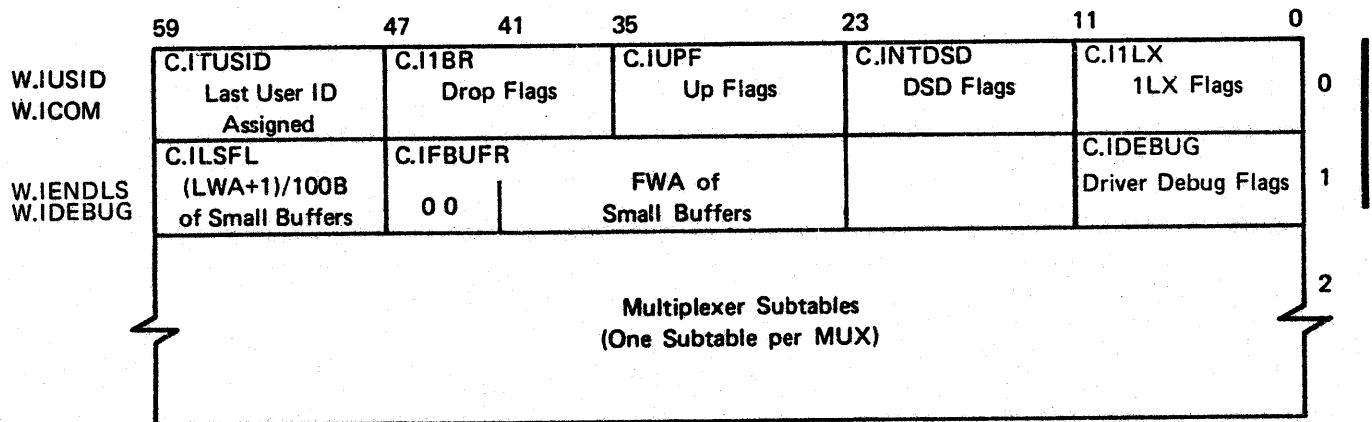
Code	Value	Description
IC (Bits 7-6) Internal Code		
IC=DIS	00	File format is display code
IC=ASCII	01	File format is ASCII
IC=BIN	10	File format is binary
--	11	Reserved
Disposition Code (Bits 5-0)		
	01	Reserved
	02	Reserved
	03	Reserved
	04	Input job ready for scheduling
	05	Input tape job
	06	Input 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	Hard copy print
	25	Reserved
HL†	26	Hard copy plot
	27	Reserved
PT†	30	Plot
	31-37	Reserved
PR	40	Any available printer
P1	41	Any available 501, 505, only EC=B4 or B6 are valid
P2	42	Any available 512, only EC=B4, B6, A6 or A9 are valid
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 for CDC
	66	System dynamic dump file
	67	Reserved for CDC
	70-77	Reserved for installations

†Recognized but not supported by SCOPE 3.4.

LINK IDENTIFICATION VALUES (C.FIDENT)

Value	Description
0000B	Reserved for CDC
0100B	Input queue link
0101B	Input queue during processing link
0102B	Output queue during processing link
0103B	Output queue file interrupted by deadstart recovery or ABORT
0104B } 0177B }	Reserved for CDC
0200B	Tape with alternate VSNs link
0201B	Tape with multiple VSNs link
0202B	Tape with 2MT(NT) link
0203B } 0277B }	Reserved for CDC
0300B	Permanent pack link
0301B	Disk VSN for file assignment
0302B } 0377B }	Reserved for CDC
0400B	Editor file during processing link
0401B	File to be replaced link
0402B	File to be deleted link
0403B	SPOT dayfile link
0404B } 0477B }	Reserved for CDC
0500B } 0577B }	Reserved for CDC
0600B	Routing information link
0601B	Reserved for CDC
0602B	System dynamic dump file link
0603B- 0677B.	Reserved for CDC
0700B	Reserved for ECS resident file or I/O buffered file link
0701B	Reserved for ECS resident library link
0702B- 6777B	Reserved for CDC
7000B- 7777B	Reserved for Installations

INTERCOM MULTIPLEXER TABLE



NOTES: INTERCOM MULTIPLEXER TABLE

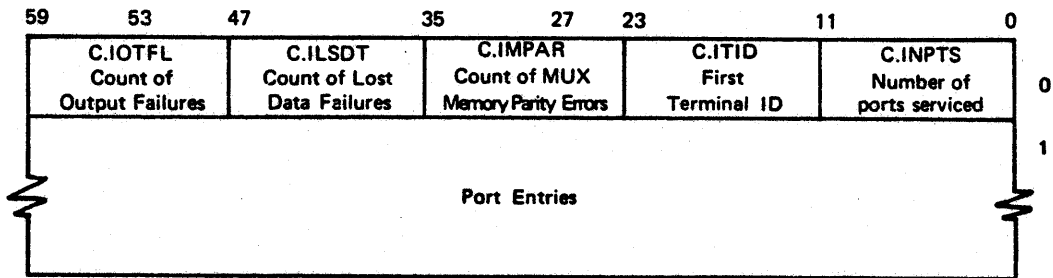
W.ICOM(0)	C.I1BR(1)	Multiplexer Table Header – Communications
Bit	0	S.IEDIT EDITLIB active
	1	S.I1I1 I1I initialization active
	2	S.I1BR 1BR drop flag
	3	S.IDI Driver-initializer active

W.ICOM(0)	C.IUPF(2)	
Bit	5	S.I1CI 1CI up
	6	S.IDD Driver up

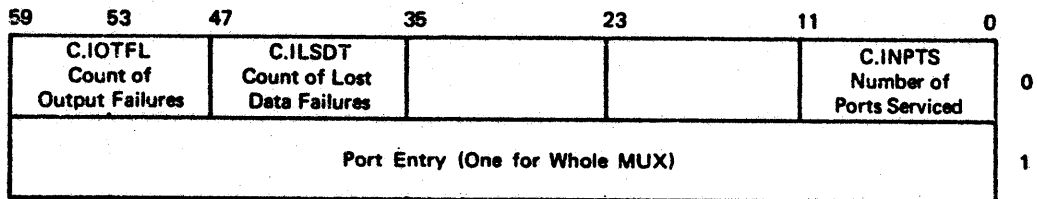
W.ICOM(0)	C.INTDSD(3)	
Bit	0	S.INTDRP INTERCOM drop requested flag
	1	S.INRST Restart in progress
	4	S.I1KOUT LOGIN Locked out
	5	S.I1CI 1CI drop requested
	6	S.IDD Driver/1LX drop requested

W.ICOM(0)	C.I1LX(4)	Two bit count of active 1LXs for each driver
------------------	------------------	--

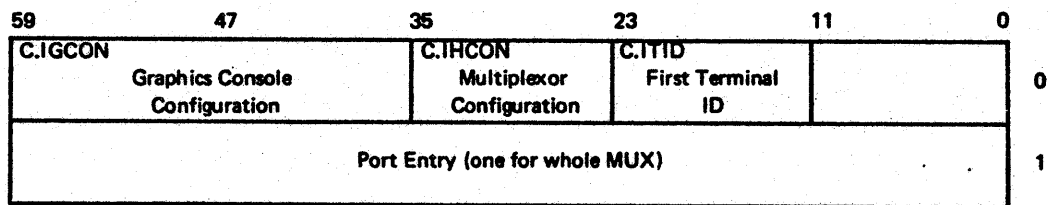
6671 MULTIPLEXER SUBTABLE



6676 MULTIPLEXER SUBTABLE

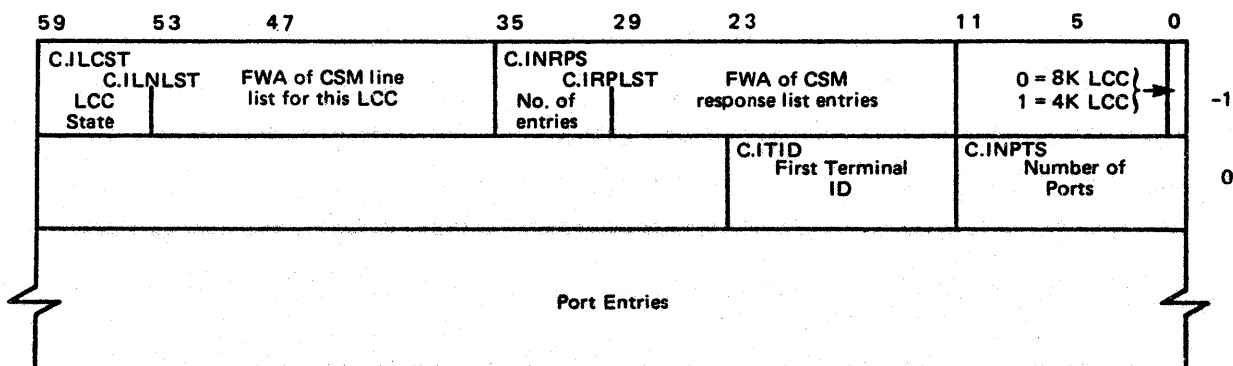


6673/6674 MULTIPLEXER SUBTABLE

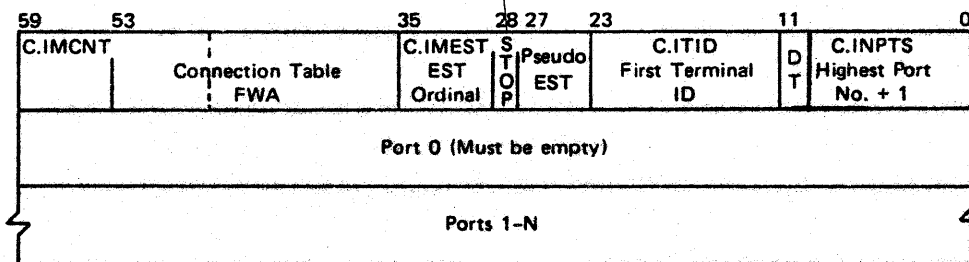


Byte C.IGCON 24 bits representing 24 possible consoles (00 to 36) from left to right.

LCC MULTIPLEXER SUBTABLE



2550 NPU MULTIPLEXER SUBTABLE



C.IMCNT

Bits 05-00 Upper 6 bits of Connection Table FWA

C.IMCNT+1

Bits 11-00 Lower 12 bits of Connection Table FWA

C.IMEST

Bits 11-06 EST Ordinal
 05 Reserved
 04 Request-To-Stop-NPU-Service Bit
 03-00 Pseudo EST

C.ITID

Bits 11-00 First terminal ID

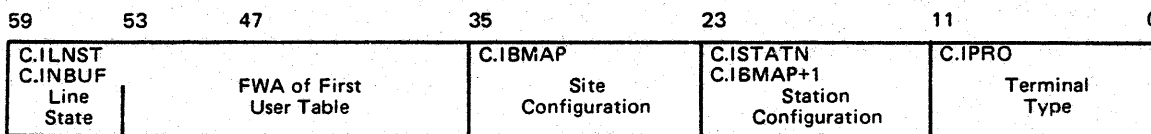
C.INPTS

Bits 08-00 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

S.IBDMP

Bit 11 Dump type: 0 means display code dump; 1, binary dump

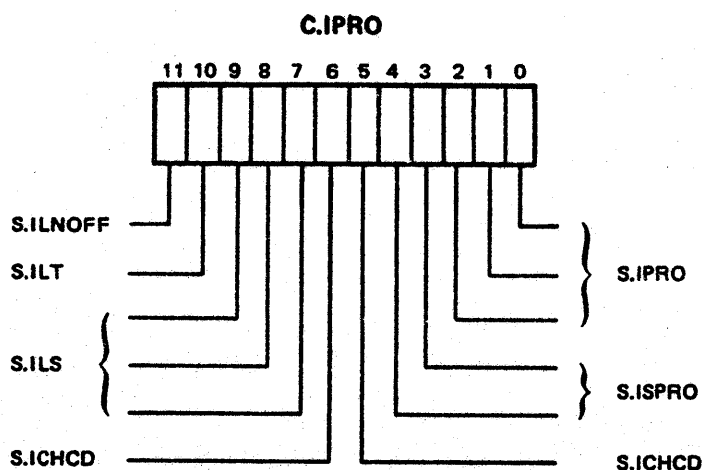
SUBTABLE PORT ENTRY



Bit map of streams configured by 73x-10. Replaces site and station address for these terminals.

C.IBMAP+1(3)

Bit	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)
	8-15	Unused			
	16	Stream	1	Configured	(IDS input)
	17	Stream	3	Configured	(KBD)
	18	Stream	7	Configured	(CR2)
	19	Stream	5	Configured	(CR1)
	20-22	Unused			



S.ILNOFF

Bit 11 Line offed = 0 on
= 1 off

S.ILT

Bit 10 Line type = 0 Dial-up
= 1 Hard-wired

S.ILS

Bits 9-07 Line speed = 0 110 BPS
= 1 134.5 BPS
= 2 150 BPS
= 3 300 BPS
= 4 2000/2400 BPS
= 5 4800 BPS
= 6 9600 BPS
= 7 19.2 UP KB

S.ICHCD Character Conversion = 0 ASCII
 = 1 External BCD
 = 2 Display Code

S.ISPRO Subprotocol
 If mode 4 terminal = 1 Mode 4A
 = 2 Mode 4C
 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

DEVICE ACTIVITY TABLE ENTRY

	59	57	47	41	35	23	11	0
	X	C.DATDST DST Ordinal	C	C.DATEQP Eqp. Type	C.DATACT Activity			Count Maintained By SPM
W.DATSTA			C.DATNFC New File Count	C.DATPRU PRU Weight	C.DATWAT 1SP			C.DATSR Stack Req. Weight
W.DATSUM		C.DATACT Controller Activity	C.DATNFC Pre. Newfile Count	C.DATPRU Previous PRU Count/10B	C.DATWAT Previous (Bypass Count)			C.DATSR Previous S.R. Count
W.DATIL*	Set By SPM				Set By 1SP			
		I/L Address	Member ID+1	DDT Ord	EST Ord			I/L Broken If ≠0 (EST)

X = Controller on = 0, off = 1 C = 1 = Shared controller (used only by Deadstart)
 Y = Controller auto-loaded, yes=0, no=1
 * W.DATIL can be cleared only by 1SP

DUMMY ENTRY FOR DUAL ACCESS DEVICE

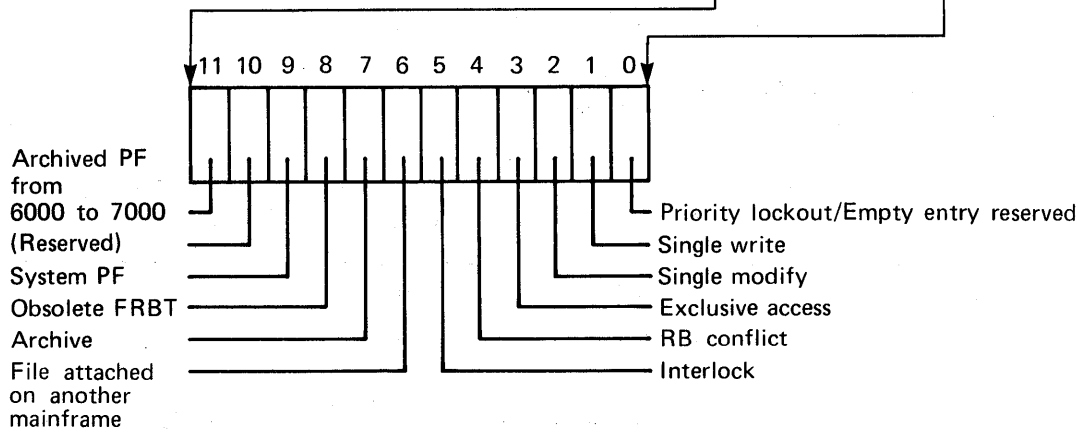
	59	57	47	35	23	11	0
X		0	(Reserved)	(Reserved)	(Reserved)	(Reserved)	(Reserved)
Y							
	(Reserved)						
	(Reserved)						
	(Reserved)						

TAPES STAGING TABLE

	59	C.STGMT	47	C.STGNT	35	23	11	0			
W.STGMAX		Number of MT Defined		Number of NT Defined		(Reserved)		(Reserved)		(Reserved)	(Total)
W.STGFRE		Number of MT ON and Unassigned		Number of NT ON and Unassigned							(Available)
W.STGUFD		Unfilled MT Demand		Unfilled NT Demand							(Unfilled Demand)
W.STGSAT		Number of MT Held by Satisfied Jobs		Number of NT Held by Satisfied Jobs							(Assigned)
		No Tape Status Flag	(unused)								
W.STGTLE	Useless information from T.MSC				Copy of T.MSC						
W.STGTLR											
W.STGTLT											

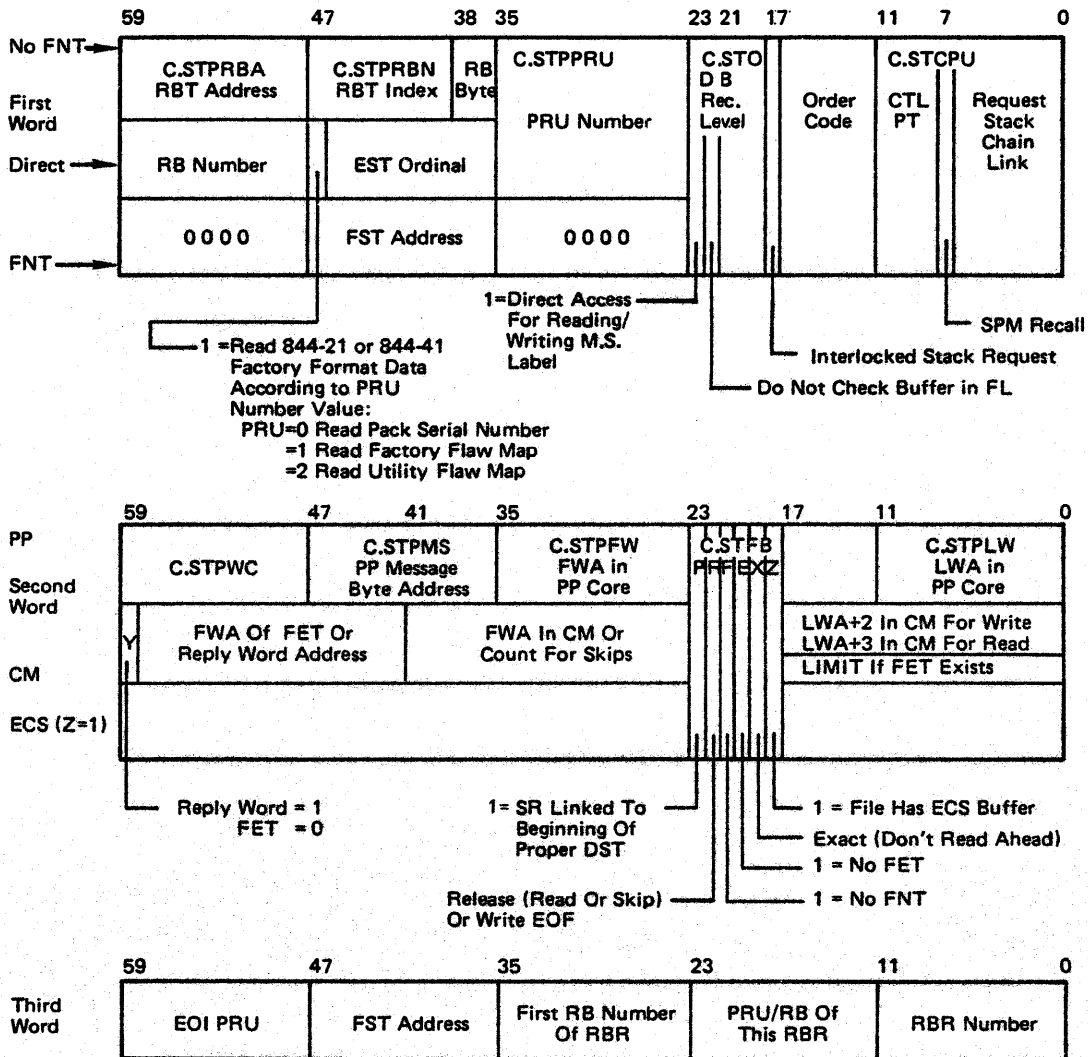
ATTACHED PERMANENT FILE TABLE

59	47	35	23	17	0	
C.PFD1	C.PFD2	C.PFD3	C.PFD4	C.PFQ* PFQ Address		Word 1
← PFD Pointer →						
C.PFRBT First RBTA	C.PFCNT2 Count 2	C.PFCNT Count 1	C.PFCY Cycle No.	C.PFLAG Flag Byte		Word 2



*If ≠ 0, PFQ Address = JDT Address of a job waiting for permanent file.

REQUEST STACK ENTRY



(Supplied by SPM) Note: SPM converts FNT format into no-FNT format but does not change the no-FNT bit.

STACK PROCESSOR ORDER CODES

00	O.READ	Read into central memory	13	O.SKB†	Skip backward
01	O.RDSK	Readskip into central memory	14	O.WRP	Write from PPU memory
02	O.RCMPR	Read into central memory, drop first 3 CM words	15	O.WRPR	Write EOF/EOR from PPU memory
03	O.RDNS	Read non-stop	16	O.BPRU†	Backspace PRU
04	O.WRT	Write from central memory	17	O.RCHN†	Evict
05	O.WRTR	Write EOF/EOR from central memory	20	O.RCTNU	Read nonstop (comparable to tape READN)
06	O.RMR	Read multiple records to central memory	24	O.WCTNU	Write nonstop (comparable to tape WRITEN)
07					
10	O.RDP	Read into PPU memory	35	O.IDLE	Wait for stack request
11	O.RDPNP	Read into PPU, drop first 3 CM words	36	O.DROP	Drop PP
12	O.SKFT†	Skip forward	37	O.SEEK	Issue overlap seeks

†Setting or not setting the interlock stack request bit has no affect on the interlock.

REQUEST SCHEDULING TABLE

Used to schedule stack requests in the request stack (T.RQS).

	59	47	35	23	11	0
Header	1SP Drop Request Flag		0	0	SR Entry Count	Empty Chain

	59	50	47	35	11	0
Entry	DDT Ordinal	S	PB Number	SPM Working Scratch		Link

S = status

RECORD BLOCK RESERVATION TABLE

	T.RBR	53	47	41	35	23	21	19	11	0
W.RBRTPA W.RBRUNT	C.RBRTPA Device Type [†]		C.RBRUNT DST Ordinal	Unit	Starting Device Address	A		C.RBRAL Allocation Style	PRU/RB	
W.RBRLAV	Length/2 Bit Table	RBR Bit Table Starting Address			C.RBREST EST Ordinal			C.RBRLAV Available RBs In CM	Available In DAM ^{††}	
	Any Additional Header Word Pairs									
T.RBRBIT	Bit Table Starting Address Pointed To By Header Word									
Length										

A Overflow flag

† Device type codes are the same as those used for the Equipment Status Table

†† During deadstart this byte contains total number of RBs. This field is only updated when DAM is referenced.

DEVICE STATUS TABLE (MASTER ENTRY)

Device Status Table entries are numbered, starting from 1, to identify DST ordinals.

59	47	41	35	31	29	23	20	17	11	0
Channel Time Accounting Factor	Reserved	Reserved For Inst.	Res.	W		M		N		Start Of Chain Pointer
1S	P*0		Driver Name	DST Ordinal	E	S	Channel		Nonzero If a PP Is Assigned	

*1S5 for DST ordinal one.

Second word used for PPIR.

- W 1SP working flag (when 1SP is active)
- 0 = 1SP idle
- S = Stack request ordinal S assigned
- M Member number (0 for master entry)
- N Number of member DST entries
- S Scheduling options
- 0 = FIFO
- 4 = Selection only
- 6 = Overlap seek and selection
- E Equipment

Driver Name A display code letter which is added to the characters 3S to form the overlay name

Valid letters are

- W 841 Disk Pack
- Y 844 Disk Pack

DST MULTIPLE ACCESS MEMBER ENTRIES

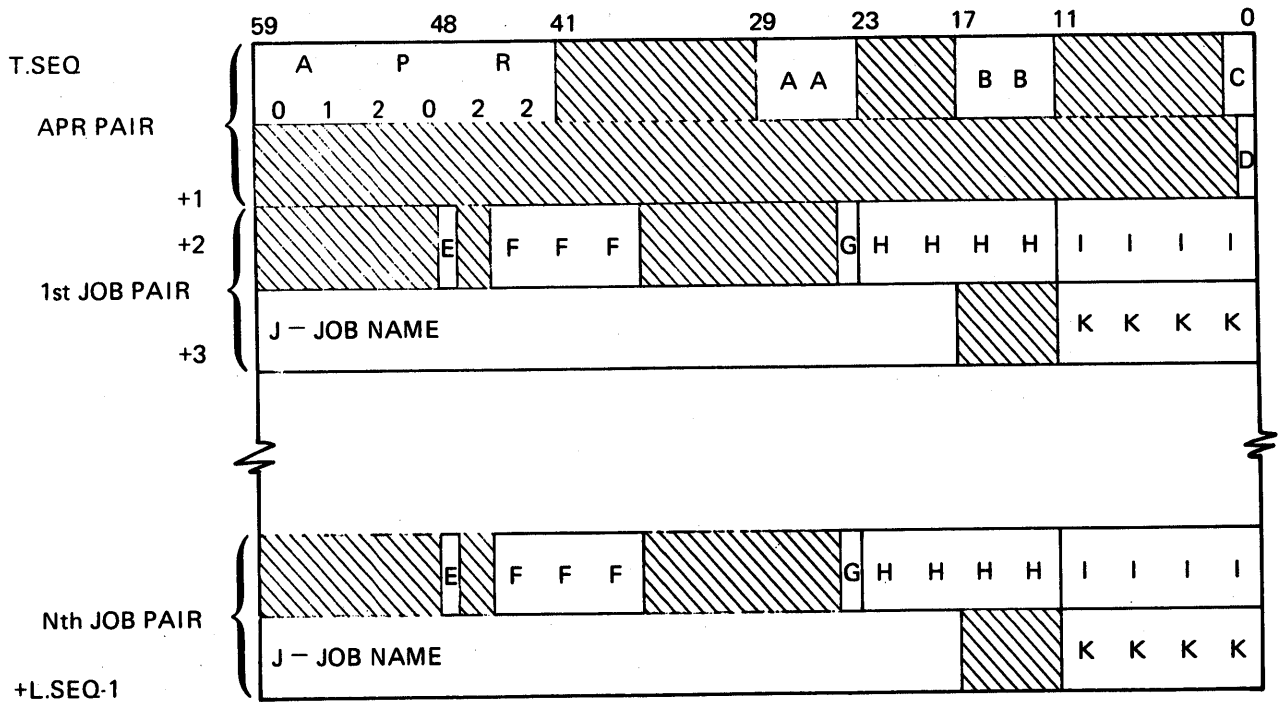
DST entries for dual access to a set of mass storage units.

59	47	41	35	31	29	23	20	17	11	0
Channel Time Accounting Factor	Reserved	Reserved For Inst.	Res.	W		M		O		Nonzero If Work Is Available
1S	PO		Driver Name	DST Ordinal	E	*		Channel		Nonzero If a PP Is Assigned

Second word used for PPIR.

- M Member number (1-3)

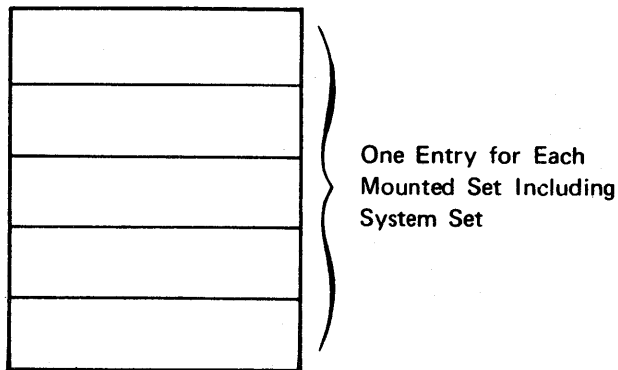
SEQUENCER TABLE



- 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 Full Flag
- F = Diagnostic Flag Bits
 - Bit 0 = CT3
 - Bit 1 = MY1
 - Bit 2 = CM6
 - Bit 3 = CU1
 - Bit 4 = ALS
 - Bit 5 = FST
 - Bit 6 = EC2
 - Bit 7 = ALX
 - Bit 8 = CEFAP
 - Bit 9 = reserved
 - Bit 10 = reserved
 - Bit 11 = reserved
- G = Entry Drop Flag
- H = Interval
- I = Clock
- J = Job Name
- K = Last known FNT Address

OVERVIEW OF MOUNTED SET TABLE (MST)

The Mounted Set Table follows the CMR Installation Area.



Entry Size = 5 CM Words

MOUNTED SET TABLE (MST) ENTRY

	59	47	35	29	23	17	11	0
W.MSVSN	C.MSVSN †Master Device VSN			C.MSMFD Mainframe Ordinal		C.PFMIL L K J I H G F E D C B A		
W.MSSN	C.MSSN †Set Name			C.MSPEOI Reserved				
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/100B	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/10B)	M	C.MSCEOI 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 C.MSHPS Number of Hash Points	Hashing Left Shift Count	Reserved		Reserved	

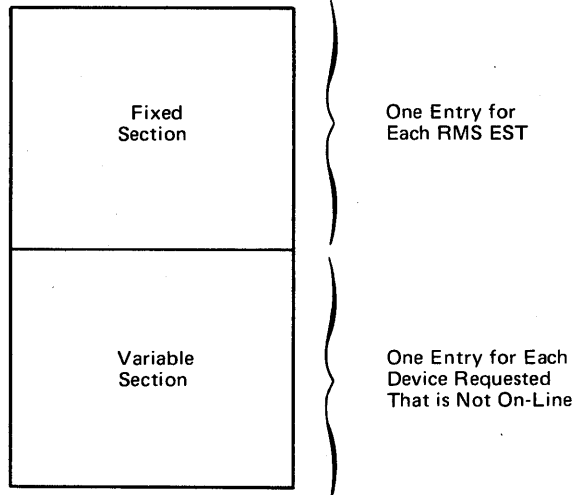
A (reserved)
 B = PFM Interlock (S.MSPFMI)
 C = Utility Interlock (S.MSUTIL)
 D = Set Interlock (S.MSSETI)
 †E = System Set Bit (S.MSSYS)
 †F = PF Default Set (S.MSPF)
 †G = Queue Set (S.MSQ)

†H = System Scratch Set (S.MSSCR)
 †Jl = Deadstart Action (S.MSACT):
 00 Check
 01 Initialize
 10 Modify
 K = Reserved (S.MSBF)
 *†L = RB Conflict (S.MSTRCF)
 M = Wrap-around Flag (S.PFCW)

†Set up by deadstart for post deadstart.
 *Not used by system after post deadstart.

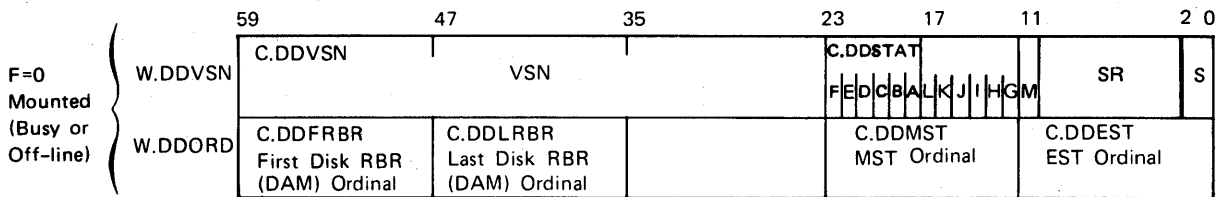
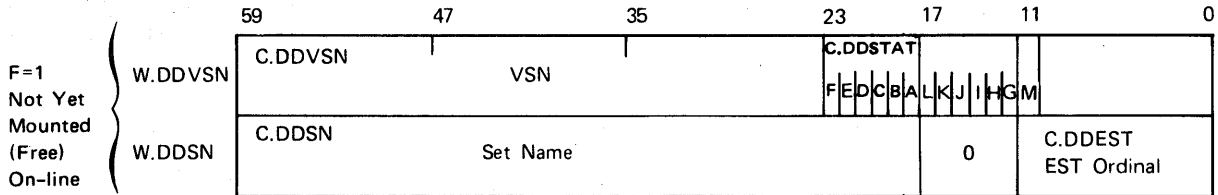
OVERVIEW OF DISMOUNTABLE DEVICE TABLE (DDT)

The dismountable device table follows the MST in CMR.



Entry Size = 2 CM Words for Variable Section
= 4 CM Words for Fixed Section

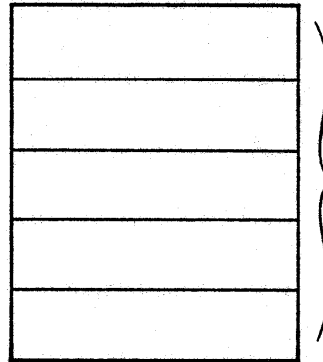
DDT ENTRY (FIXED SECTION)



- | | |
|--|---|
| C.DDSTAT (EST Status) | G = System Device (S.DDSYS) |
| A = Request Idle (S.DDIDLE) | H = PF Device (S.DDPF) |
| B = Busy (S.DDBUSY) | I = Queue Device (S.DDQ) |
| C = Free (S.DDFREE) | J = Master Device (S.DDMSTR) |
| D = Off (S.DDOFF) | K = Preallocated (S.DDPRE) |
| E = Relabel during deadstart (S.DDRLB) | L = Reserved (S.DDBF) |
| F = SN bit; device not mounted if set (S.DDSN) | M = Overlap seek in progress |
| | S = Overlap seek status |
| | SR = Stack request ordinal assigned to unit |

OVERVIEW OF MOUNTED SET TABLE (MST)

The Mounted Set Table follows the CMR Installation Area.



One Entry for Each
Mounted Set Including
System Set

Entry Size = 5 CM Words

MOUNTED SET TABLE (MST) ENTRY

	59	47	35	29	23	17	11	0
W.MSVSN	C.MSVSN †Master Device VSN			C.MSMFD Mainframe Ordinal		C.PFMIL L K J I H G F E D C B A		
W.MSSN	C.MSSN †Set Name			C.MSPEOI		Reserved		
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/100B	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/10B)		M	C.MSCEOI 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 C.MSHPS Number of Hash Points	Hashing Left Shift Count		Reserved		Reserved

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 = Deadstart Action (S.MSACT):

00 Check

01 Initialize

10 Modify

K = Reserved (S.MSBF)

*†L = RB Conflict (S.MSTRCF)

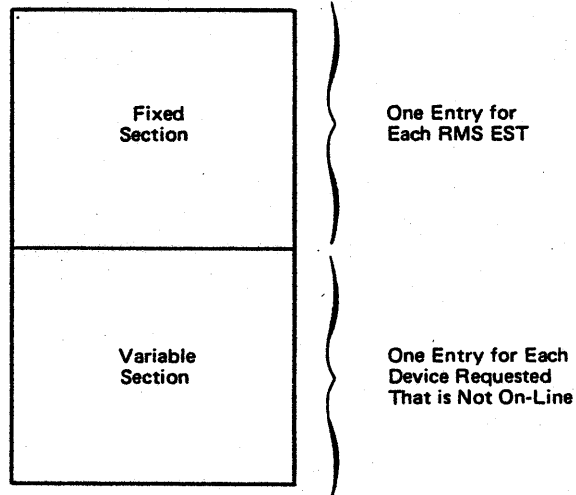
M = Wrap-around Flag (S.PFCW)

†Set up by deadstart for post deadstart.

*Not used by system after post deadstart.

OVERVIEW OF DISMOUNTABLE DEVICE TABLE (DDT)

The Dismountable Device Table follows the MST in CMR.



Entry Size = 2 CM Words for Variable Section
 = 4 CM Words for Fixed Section

DDT ENTRY (FIXED SECTION)

	59	47	35	23	17	11	0	
F=1 Not Yet Mounted (Free) On-line	W.DDVSN	C.DDVSN		VSN		C.DDSTAT		
						F E D C B A L K J I H G		
	W.DDSN	C.DDSN				Set Name	0	C.DDEST EST Ordinal

	59	47	35	23	17	11	0
F=0 Mounted (Busy or Off-line)	W.DDVSN	C.DDVSN		VSN		C.DDSTAT	
						F E D C B A L K J I H G	
	W.DDORD	C.DDFRBR First Disk RBR (DAM) Ordinal	C.DDLRBR Last Disk RBR (DAM) Ordinal		C.DDMST MST Ordinal		C.DDEST EST Ordinal

- C.DDSTAT (EST Status)
- A = Request Idle (S.DDIDLE)
 - B = Busy (S.DDBUSY)
 - C = Free (S.DDFREE)
 - D = Off (S.DDOFF)
 - E = Relabel during deadstart (S.DDRLB)
 - 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 = Reserved (S.DDBF)

Statistic Area	W.DDSTA	C.DDNFC New File Count	C.DDWAT Bypass Count	*	C.DDSR Number or Accesses	*
	W.DDSUM	C.DDACT Unit Activity	C.DDNFC * Previous File Count	C.DDWAT * Bypass Count	*	C.DDSR Pre number of Accesses

*Used as temporary save area by deadstart

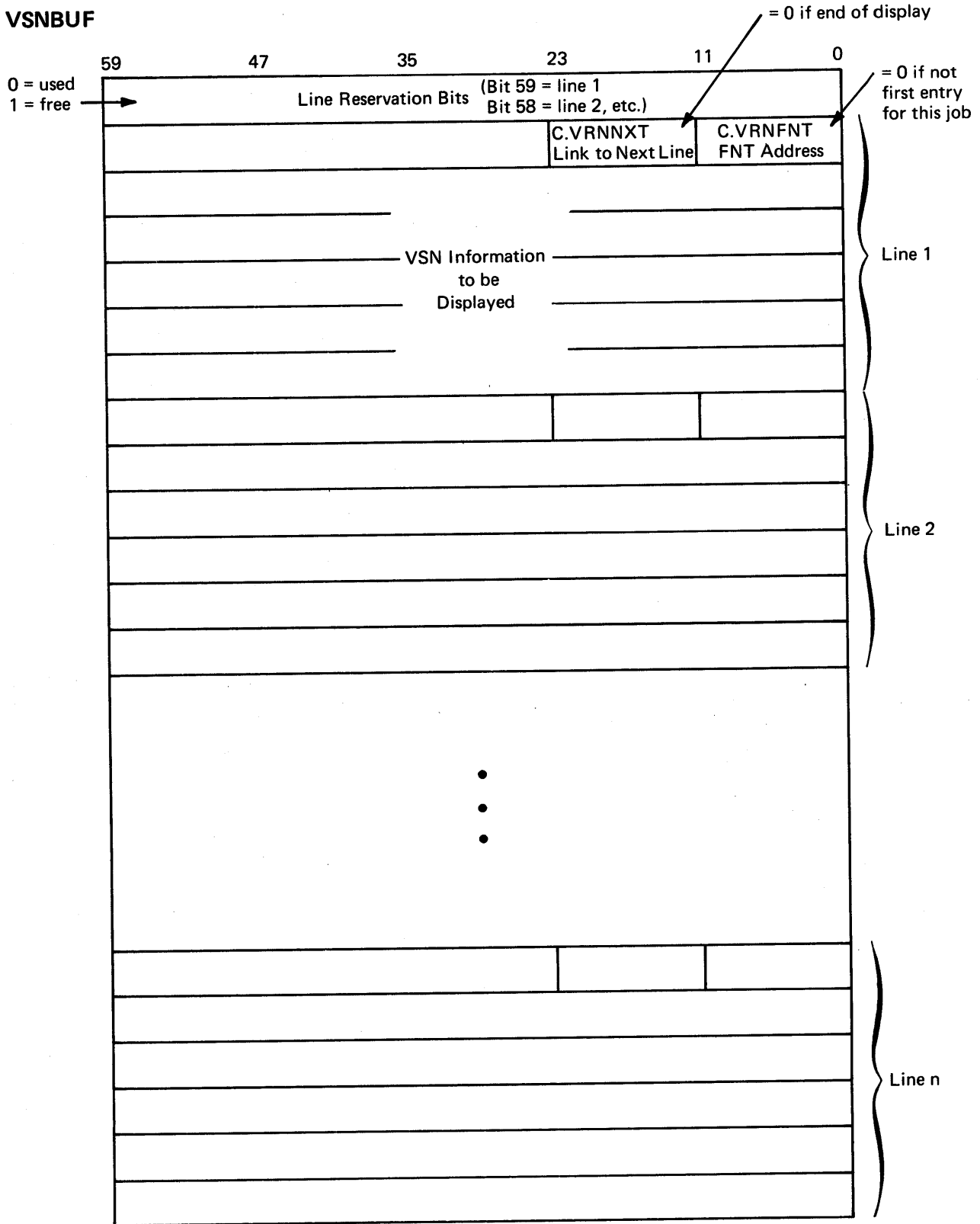
DDT ENTRY (VARIABLE SECTION)

Master Device	W.DDVSN	59	47	35	23	17	11	0
	W.DDSN	C.DDVSN	VSN		C.DDSDQA F	SDQ Address (First JDT Entry)		
		C.DDSN			Set Name		0	0

Member Device	W.DDVSN	59	47	35	23	17	11	0
	W.DDORD	C.DDVSN	VSN		C.DDSDQA F	SDQ Address (First JDT Entry)		
		C.DDFRBR First Disk RBR (DAM) Ordinal	C.DDLRBR Last Disk RBR (DAM) Ordinal	C.DDSDQC SDQ Count	C.DDMST MST Ordinal	0		

F = SN bit; device not mounted if set

VSNBUF



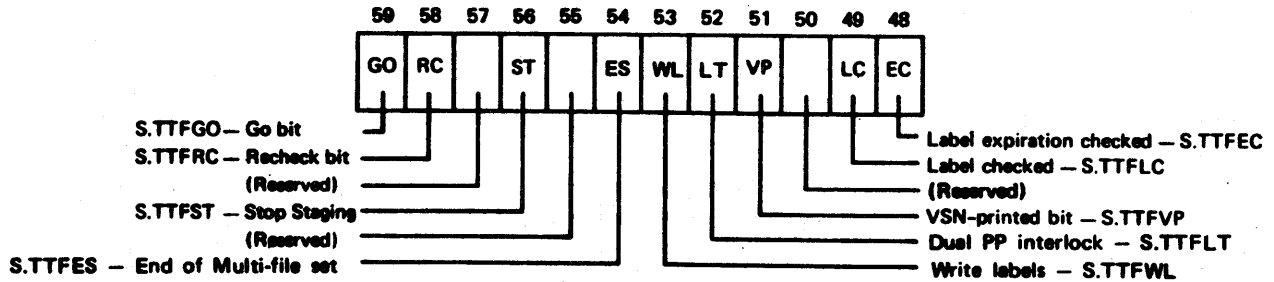
TAPES TABLE

	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	Multi-File Name				Reel Number			
W.TFLGS	C.TFLGS Flag Bits			Reserved			21A/21B 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			

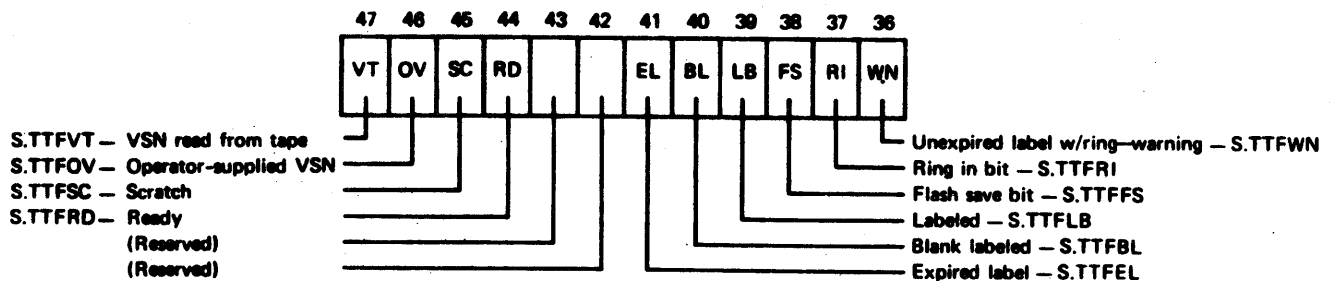
†These bytes are not used by 66x drivers.

NOTES: TAPES TABLE

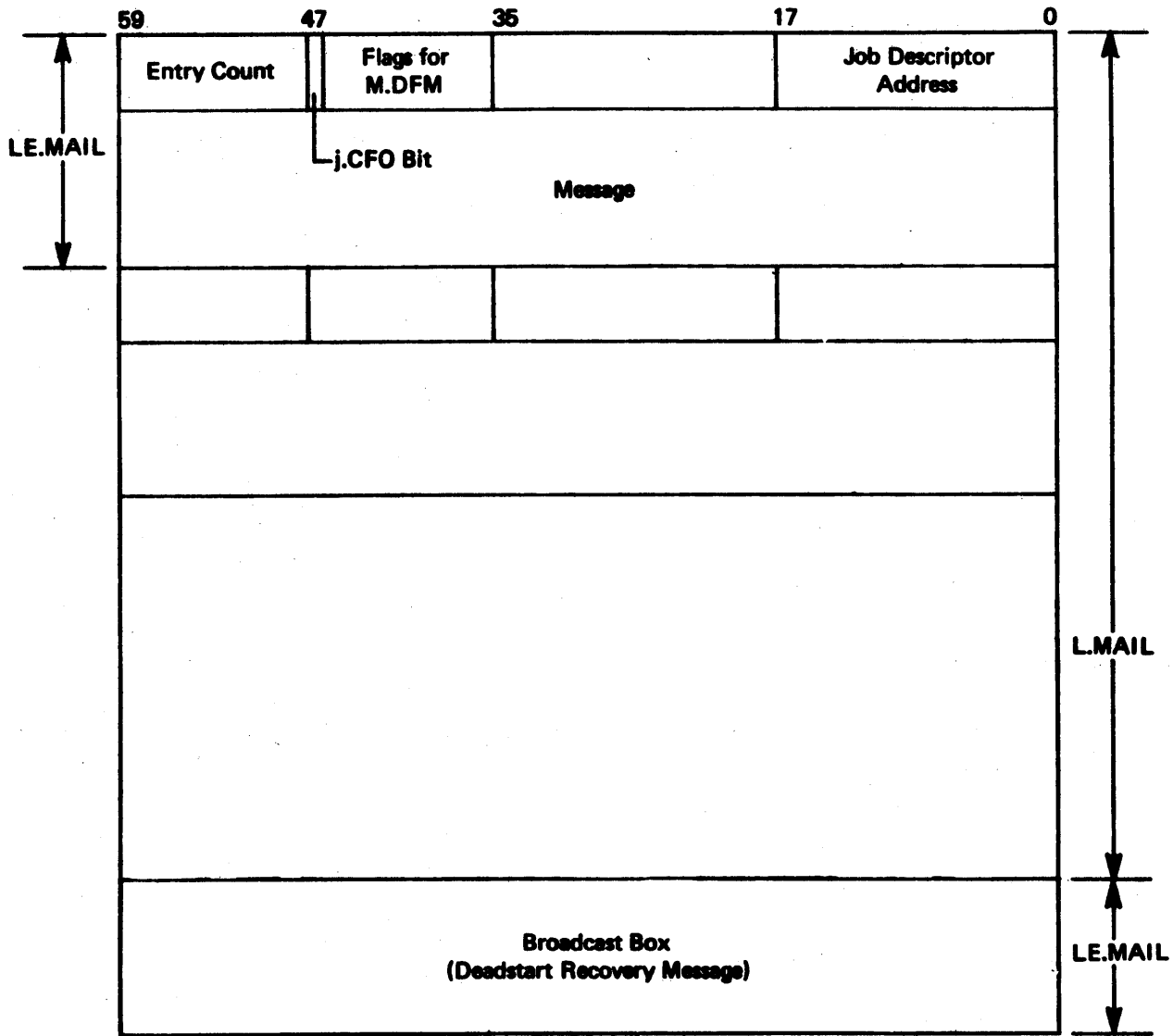
C.TFLGS — Job-Oriented Flag Bits



C.TFLGS+1 — Unit-Oriented Flag Bits



MAILBOX



ID TABLE

	59	47	35	29	23	17	0
W.IDTHID	C.IDTL L.IDT	C.IDTPID First PID Ordinal	C.IDTFLL LID Ordinal First Last			Host ID	
W.IDTLL	(Reserved)						
W.IDTFLO						Logical ID 1	
						Logical ID 2	
						Logical ID 3	
						Logical ID n	
	C.IDSIDT Station IDT RA	C.IDTFNO	C.IDTEST EST Ordinal			Physical ID	
						Physical ID	
						Physical ID	
						Physical ID	

NOTES: ID TABLE

W.IDTHID	C.IDTL	(Host word)	Contains the length of the ID table. Default is 40B. (L.IDT)
	C.IDTPID		Contains the first physical ID Ordinal.
	C.IDTFLL		Contains the first logical ID Ordinal in the upper six bits and the last logical ID Ordinal in the lower six bits.
		Bits 17-0	Contain the host ID as 3 letters or digits. Third character must be a letter.
W.IDTLL			Reserved.

W.IDTFLO

(Logical ID entries)

Bits 17-0

Contain a logical ID as 3 letters or digits.

C.IDSIDT

(Link words)

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 IDs associated with linked mainframes.

C.IDTFNO

Contains relative address of station PID information area.

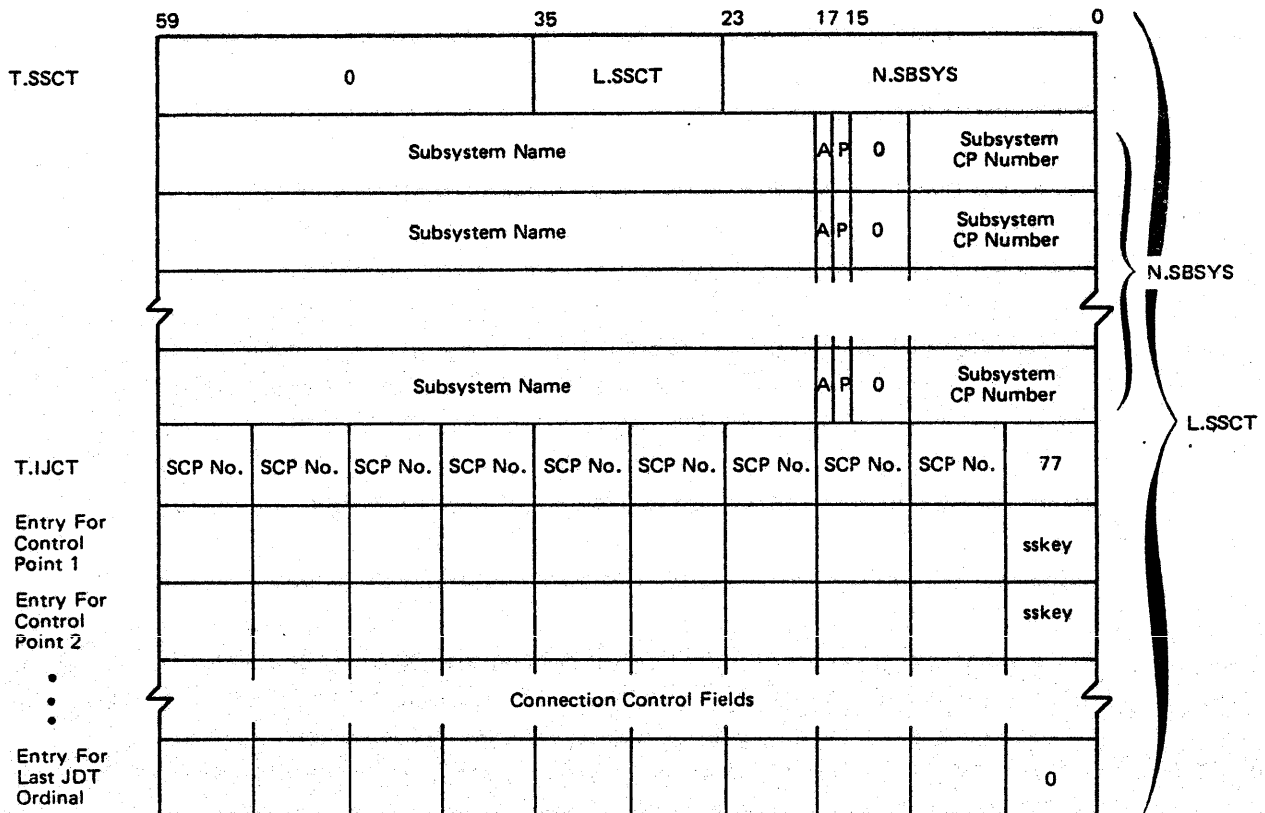
C.IDTEST

Bits 17-0

Contains the EST ordinal of the PID.

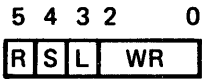
Contain a physical or link ID as 3 letters or digits.

SUBSYSTEM CONTROL TABLE



sskey values: 6 14B 22B 30B 36B 44B 52B 60B 66B

Connection Control Field



	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	CALL SS	SF.ENDT

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 the operator).

When a subsystem is active:

Bit 17 (A) is set to 1.

An skey value is assigned which specifies a column of connection control fields.

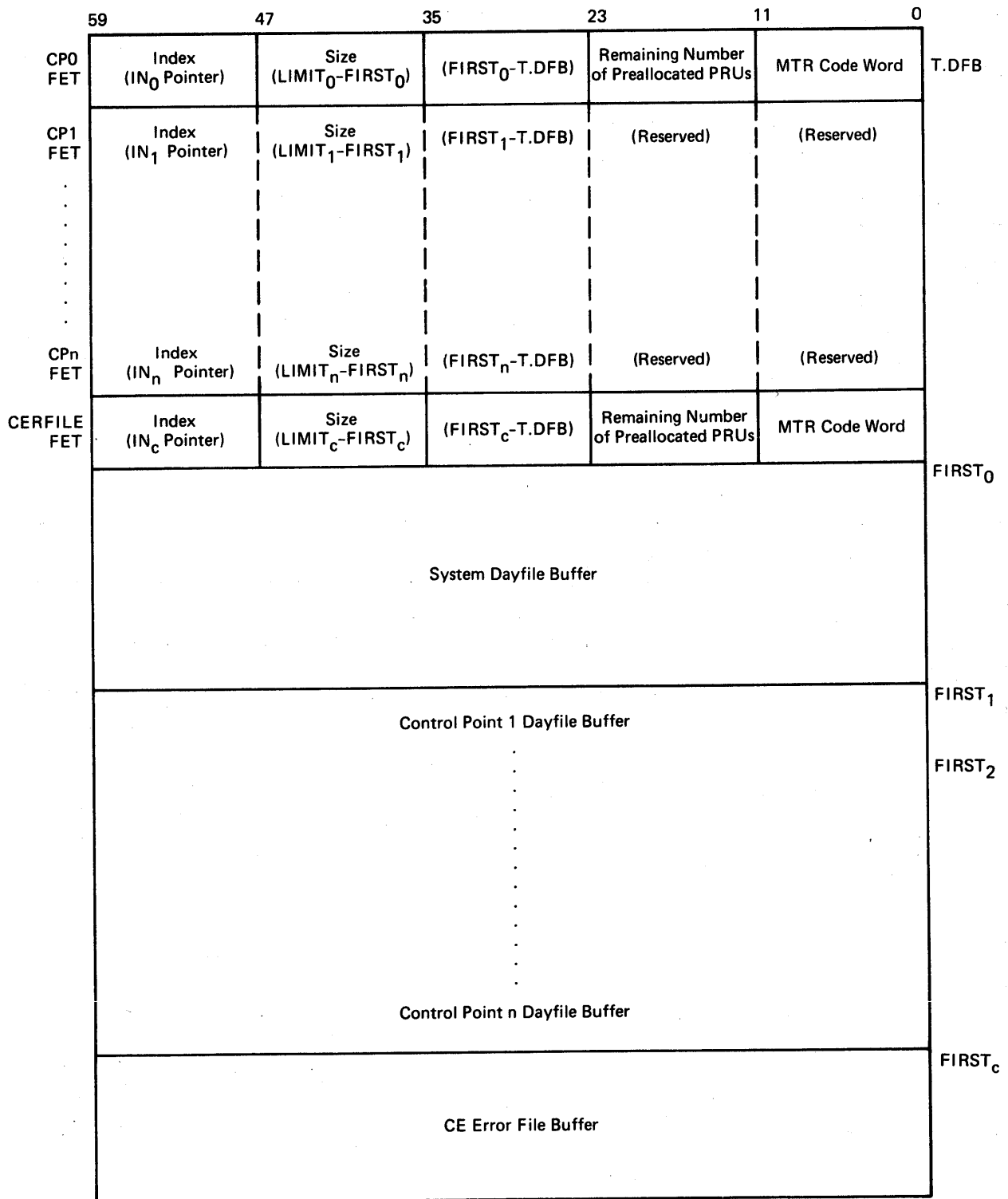
This skey value is put in the word for the subsystem job control point number.

The subsystem CP number is put in the subsystem word and the T.IJCT field specified by skey.

PERIPHERAL JOB TABLE

PP Input Register Image	} One Entry
W.PPMES4 Image	
W.PPMES5 Image	
W.PPMES6 Image	

DAYFILE FET AND BUFFER AREA



JOB CONTROL AREA

	59	47	35	23	17	11	0
Input Queue Entry	C.JCMXB Max # Class 1 Batch	C.JCMTB Max # Class 2 Batch	C.JCCLK/ C.JCAFL a b c d	C.JCQP Quantum Priority	C.JCBQ Quantum Value		
	C.JCCNB Current # Class 1 Batch	C.JCCTB Current # Class 2 Batch	C.JCEMC # Empty FNT Entry	e	C.JCNTJ # Ready-to-run Tape Jobs	f # Ready-to-run Non-tape Jobs	
Class 1 (No Non-allocatable Devices)	C.JCMIN Minimum Class Priority	C.JCMAX Maximum Class Priority	C.JCAR Aging Rate	C.JCQP Quantum Priority	C.JCBQ Quantum Value		
	C.JCNAM Name of Job Class				C.JCFRST Address of First JDT in Chain		
Class 2 (Non-allocatable Devices)	C.JCMIN	C.JCMAX	C.JCAR	C.JCQP	C.JCBQ		
	C.JCNAM				C.JCFRST		
Class 3 (Interactive Jobs)	C.JCMIN	C.JCMAX	C.JCAR	C.JCQP	C.JCBQ		
	C.JCNAM				C.JCFRST		
Class 4 (Multi-user Jobs)	C.JCMIN	C.JCMAX	C.JCAR	C.JCQP	C.JCBQ		
	C.JCNAM				C.JCFRST		
Class 5 (Express Jobs)	C.JCMIN	C.JCMAX	C.JCAR	C.JCQP	C.JCBQ		
	C.JCNAM				C.JCFRST		
Class 6 (Graphics Jobs)	C.JCMIN	C.JCMAX	C.JCAR	C.JCQP	C.JCBQ		
	C.JCNAM				C.JCFRST		

- a = Anticipated FL/1000B
- b = 1 Increment to AFL for INTERCOM (AFL.INT) has been added (S.JCAFL)
- c = Reserved
- d = 1 11B failed to initiate a job last time called (S.JC11B)
- e = 1 Fixed priority tape jobs in Input Queue
- f = 1 Fixed priority non-tape jobs in Input Queue

JOB DESCRIPTOR TABLE ENTRY

	59		47		38	35		23	20	17		11		0	
W.JDNAM W.JDLNK	Job Name										C.JDLNK		Link to Next JD in Chain		
W.JDSWP	C.JDEQC Eq. Code MST ord.		C.JDFRB First RBT Word Pair		C.JDIFLG C.JDIFLG Flags		C.JDFL FL/100B		C.JDPFL Positive FL						
W.JDSD	C.JDCPN CP# Priority		C.JDORD J.D. Ordinal		C.JDTL Time Left		C.JDOPF Operator Flags		C.JDORG SSW Origin						
W.JDMGR	C.JDJST Job St. Class		C.JDPFM C.JDTIN PFM Bits		Time into Chain		C.JDBP Base Priority		C.JDLPFL C.JDRU PFL/100B PP/CP						
W.JDINT	C.JDID INTERCOM User ID		C.JDCPT CPU Time		C.JDSID Source Mainframe ID		C.JDIUTA		User Table Address						

NOTES: JOB DESCRIPTOR TABLE

W.JDNAM(0)	Bits 18-59	Job name as found in job input FNT
W.JDLNK	Bits 00-17	Contains 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)	C.JDEQC(0) C.JDFRB(1)	Contain the equipment code, MST ordinal, and first RBT number of the swap file (F.JDSWI in word W.JDMGR is set); else contain first subpage address of ECS swap file
	C.JDFLG(2) C.JDIFLG(2)	Bit 35 S.JDBC(43B) Set if recovery took place
		Bit 34 S.JDNRR(42B) Set if job cannot be rerun
		Bit 33 S.JDLGI(41B) Set if no swap file exists and control point area must be initialized (e.g. LOGIN)
		Bit 32 S.JDLGO(40B) Set if no swap file is to be generated for this INTERCOM job (e.g. LOGOUT command)
		Bit 31 S.JDNJ(37B) Set to indicate control cards must be read from INTERCOM area
		Bit 30 S.JDECS(36B) Set if swap file is on ECS
		Bit 29 S.JDSKFL(35B) Set if FL is to be skipped on swap file

NOTES: JOB DESCRIPTOR TABLE (CONT'D)

	Bit 28	S.JDROLL(34B)	Set if job cannot be swapped out
	Bit 27	S.JDNFNT(33B)	Set if 1AJ should not search FNT table
	Bit 26	S.JDFAZ(32B)	Set if file at control point 0
	Bit 25	S.JDINTR(31B)	Set to terminate INTERCOM job on recovery deadstart
	Bit 24		Unused
	C.JDFL(3)		Contains the FL/100, including the job control block which is created when job is swapped out, needed to swap this job in
	C.JDPFL(4)		Contains the relative starting address/100 of the job control block when present for job swapping; otherwise, contains zero
W.JDDSD(2)	C.JDCPN(0)		Contains control point number in upper 6 bits. (Set when job is in execution or rolled out.) In lower 6 bits, job or rerun priority
	C.JDORD(1)		Contains job descriptor (JDT) ordinal
	C.JDTL(2)		Contains job time left (upper 12 bits)
	C.JDOPF(3)		Contains flags set by operator
	Bits 23-21		Low order 3 bits of time left
	Bits 20		Currently not used
	Bit 19	S.JDTLI(23B)	If job is to be temporarily locked in to a CP
	Bit 18	S.JDEXP(22B)	If job is to be placed in express queue
	Bit 17	S.JDGO(21B)	If operator typed GO
	Bit 16	S.JDNS(20B)	If job must not be swapped out or rolled out when at a control point
	Bit 15	S.JDLOK(17B)	If job must not be brought to control point
	Bits 12-14		Error codes: F.JDKILL, F.JDDROP, F.JDRRUN, F.JDRRNP

NOTES: JOB DESCRIPTOR TABLE (CONT'D)

	CJDORG(4)	Bits 11-06	Sense switches	
		Bit 05	S.JDINT	Set for a standard INTERCOM job
		Bit 04	S.JDMUJ	Set for a multi-user job
		Bit 03	S.JDGR	Set for a graphics job
		Bit 02	S.JDRT	Set for a real time job
		Bits 01-00		Currently not used
W.JDMGR(3)	C.JD JST(0)	Bits 54-59		Values describe the job status
		7x	F.JDWMM	If job is waiting for MMF action (GETPF, SAVEPF, PURGE)
		6x	F.JDWPK	If job is waiting for permanent pack
		5x	F.JDWIA	If job is waiting for INTERCOM action (in INTERCOM queue)
		4x	F.JDWOA	If job is waiting for operator action (in operator action queue)
		3x	F.JDWDA	If job is waiting for device assignment (in device queue)
		2x	F.JDWPF	If job is waiting for a permanent file availability (in permanent file queue)
		1x	F.JDWCM	If job is waiting for central memory (in central memory queue)
		0x	F.JDLMB	If job is waiting for entry in a scheduling structure (in limbo)
		x3	F.JDWCC	If job is being swapped or rolled out
		x2	F.JDACT	If job is currently executing at a control point (is active)
		x1	F.JDSWI	If job is being swapped or rolled in
		x0		If job is swapped or rolled out
		Bits 48-53		Contain the job class values
		06	F.JDGRA	For graphics job

NOTES: JOB DESCRIPTOR TABLE (CONT'D)

05	FJDEXP	If express handling was requested for this job
04	FJDMUJ	For multi-user job
03	FJDINT	For standard INTERCOM job
02	FJDBNA	For batch job with non-allocatable device requirements
01	FJDBAT	For batch job with no non-allocatable device requirements

C.JDPFM(1) Contains information used by permanent file manager

Bit 47	Purge bit
Bit 46	Exclusive access desired
Bit 45	Control permission desired
Bit 44	Modify permission desired
Bit 43	Extend permission desired
Bit 42	Read permission desired

C.JDTIN(1) Bits 41-24 Contain the time at which job descriptor was attached to job class chain

C.JDBP(3) Bits 23-12 Contain job base priority

C.JDRU(4)
C.JDLPFL(4) Bits 5-0 Contain ratio of PPU 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).

Bits 11-6 Positive field length/100B of job control block created at swapout time

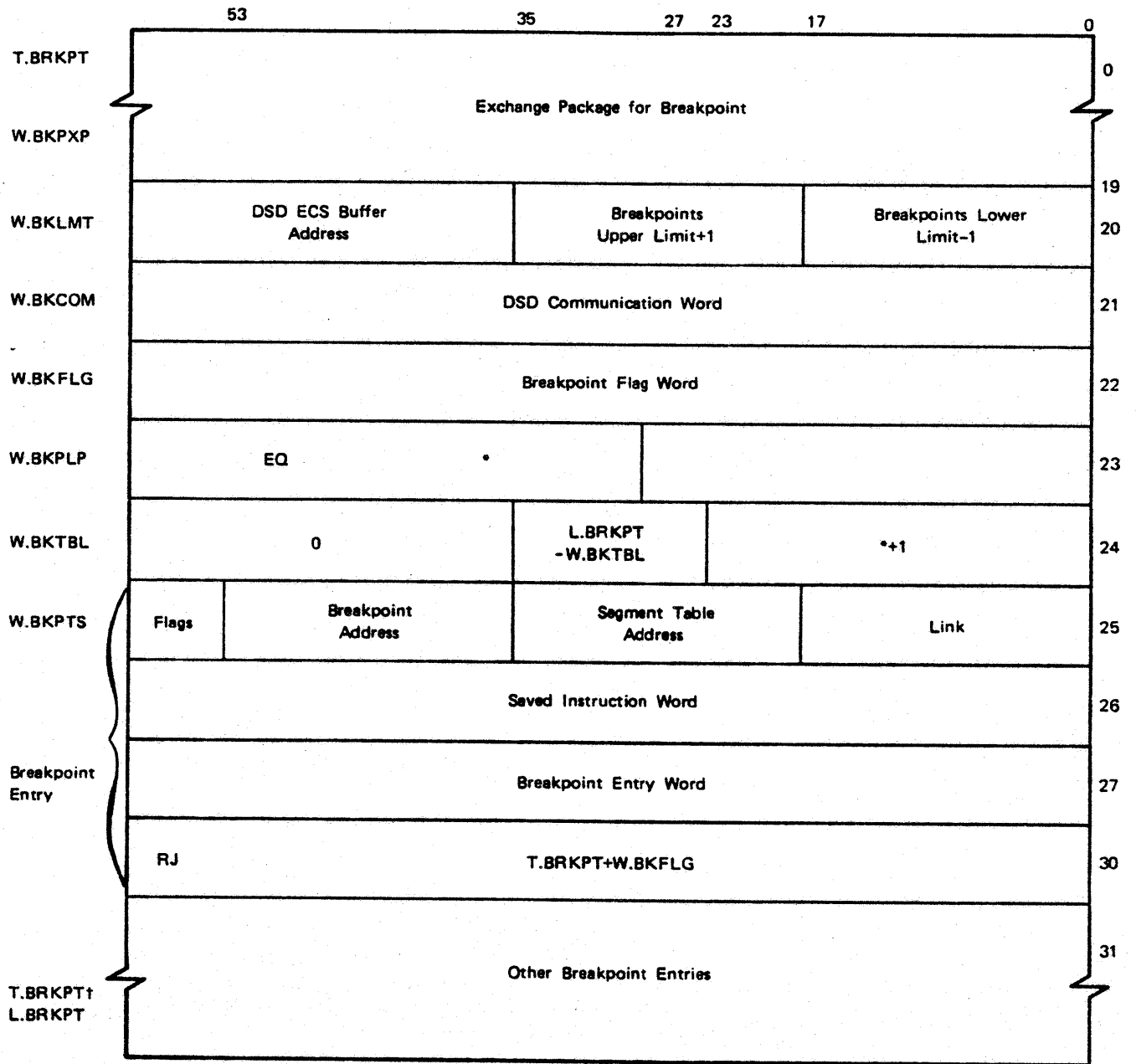
W.JDINT(4) **C.JDID(0)** Bits 59-48 INTERCOM user ID associated with remote batch job

C.JDCPT(1) Bits 47-36 CPU time, in seconds, used by this job

C.JDSID(2) Bits 35-18 Source mainframe identifier

C.JDIUTA(3)
C.JDIUTA+1 Bits 17-0 Address of User Table for interactive or graphics jobs, or MUJ Table for multi-user jobs

BREAKPOINT TABLE



NOTES: BREAKPOINT TABLE

W.BKLMNT (20B)	Bits 0-17	First absolute address that may be breakpointed -1
	18-35	Last absolute address that may be breakpointed +1
	36-59	DSD ECS display buffer
W.BKCOM (21B) (Set by DSD)	Bits 0-11	Mode flag = 1 monitor = 0 user
	12-23	Breakpoint entry relative address +4 (to be processed)
	24-59	Non-zero if bits 0-23 are zero
W.BKFLG (22B)	Bits 0-30	Reserved
	30-47	Breakpoint entry table address +3 (if flag = 400B)
	48-59	Processing flag 400B breakpoint bit 200B breakpoint processed 100B restart CPU 1000B release breakpoint
W.BKPLP (23B)		Breakpoint wait loop
W.BKTBL (24B)	Bits 0-23	Start of breakpoint entries
	24-35	Number of breakpoint entries *4
	36-59	0
Breakpoint Entry		
Word 0	Bits 0-17	Link to next breakpoint entry for the segment (end of chain=0)
	18-35	Address of segment table entry for segment or 0
	36-53	Breakpoint address
	54-59	Processing flag
Word 1		Saved instruction word (from breakpoint address)
Word 2		Entry word; breakpoint word is replaced with a return jump to this word
Word 3		Return jump to breakpoint wait loop

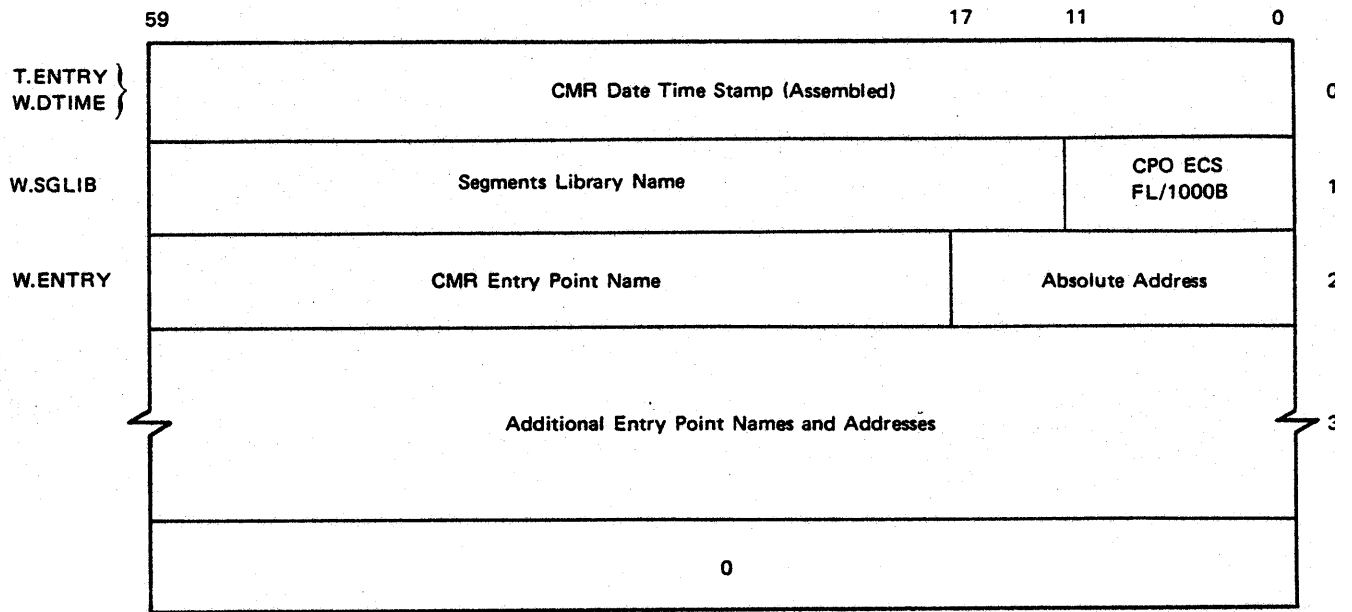
AREA TABLE

	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

NOTES: AREA TABLE

- 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 (3 digits), two digits hour, two digits minutes, and two digits seconds, all in display code.

ENTRY TABLE

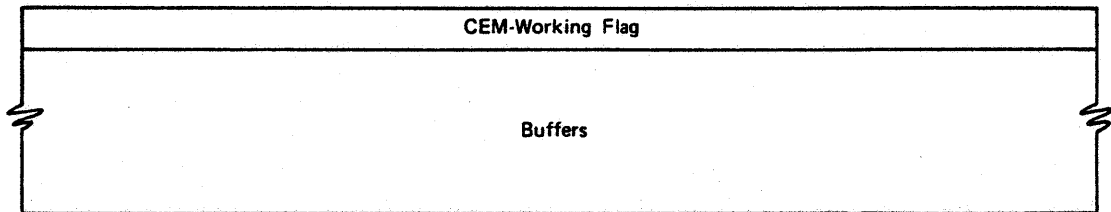


NOTES: ENTRY TABLE

- W.SGLIB (1) Library containing segments to be used with this CMR. Control point 0 FL/1000B is used for ECS system code.
- W.ENTRY (2) List of entry points defined in CMR

CEFAP BUFFER AREA

T.BCFAP



Buffer for Error Codes 0-3

X0 = ECS FWA of Transfer
A0 = CM FWA of Transfer
Bj+K = Length of Transfer
Exact ECS Address of Transfer
CEFAP Error Code (Unweighted)
Pointer to Next Buffer (0 = End of Chain)

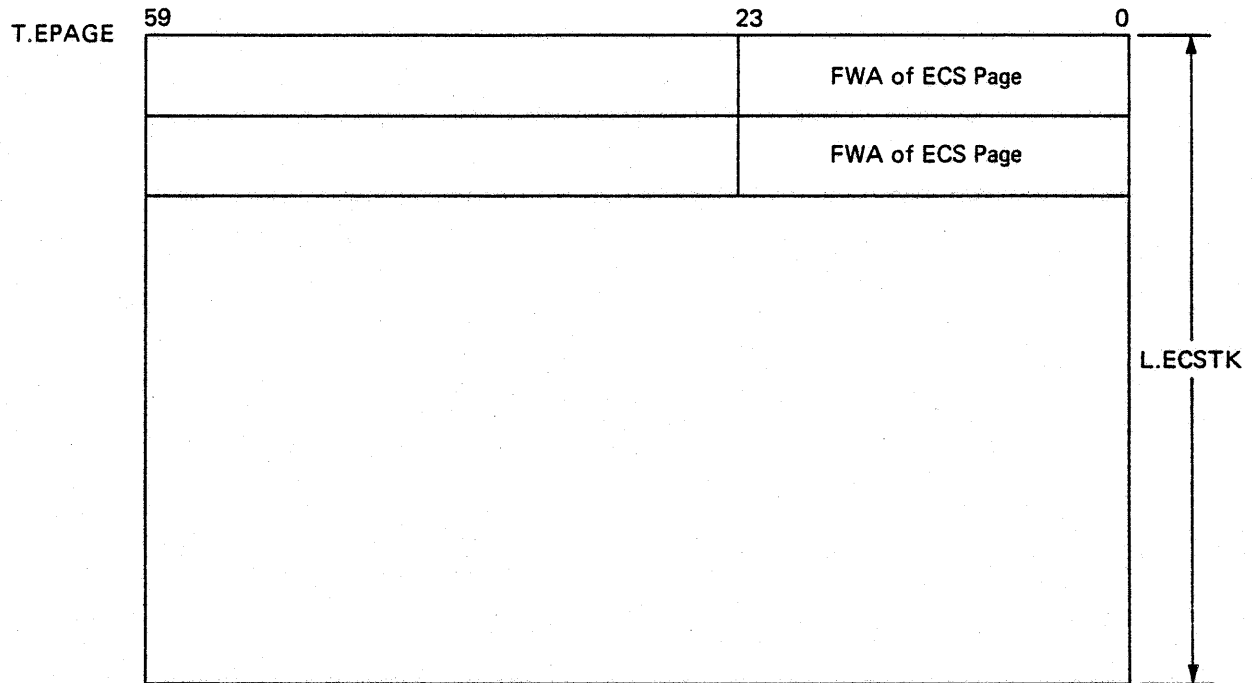
Buffer for Error Codes 20-27

ECS FWA of Transfer (or Unused)
Port Channel
Length of Transfer (or Unused)
PPNT Address + 1
CEFAP Error Code
Pointer to Next Buffer (0 = End of Chain)

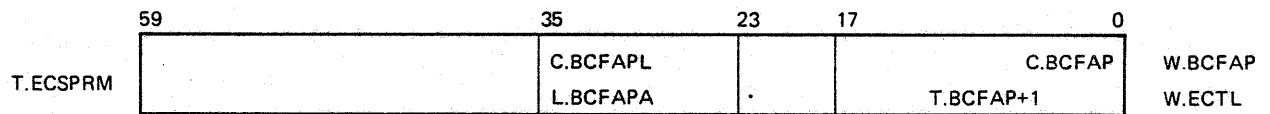
Error Code (Unweighted)

- | | |
|--|---------------------------|
| 0 = Write Abort | 32 = Positioner not ready |
| 1 = Read Parity Error | 33 = 6681 INT/EXT reject |
| 2 = Recovered Read Parity | 34 = Unit busy too long |
| 3 = Interlock Register Abort | 35 = Recovered RMS error |
| 20 = Unclassified DDP Error (DDP Reply = 0) | 36 = Unrecovered RMS err |
| 21 = DDP Found ECS Abort (DDP Reply = 1) | 37 = Channel stays active |
| 22 = DDP Channel Drops Active | 40 = Unable to connect |
| 23 = DDP Channel Stays Full | 41 = 6603II position fail |
| 24 = DDP Found ECS Read Parity Error (DDP Reply = 4) | 42 = Reserved for RMS |
| 25 = Unclassified DDP Error (DDP Reply = 5) | 43 = Reserved for RMS |
| 26 = DDP Unable to Connect | 44 = No status returned |
| 27 = DDP Logically OFF | 45 = Reserved for RMS |
| 30 = Reserved for RMS | 46 = ADDR byte not accept |
| 31 = Unit not ready | |

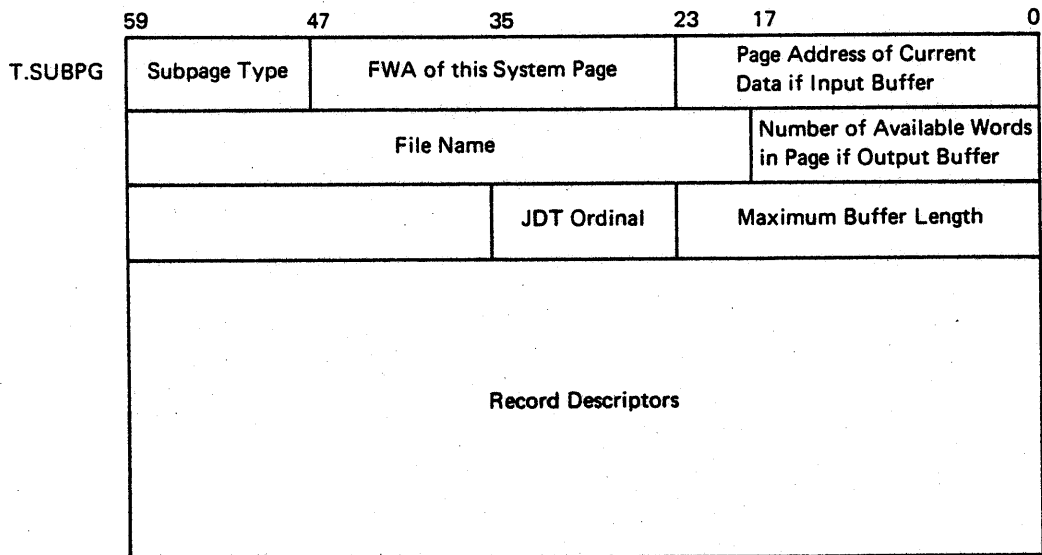
EMPTY PAGE STACK



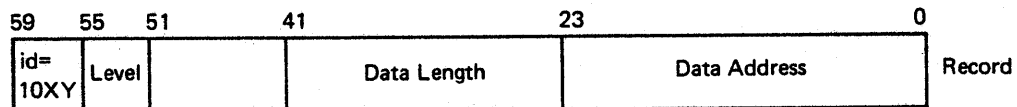
(A6 + B5 in CP.MTR exchange
package points to next entry)



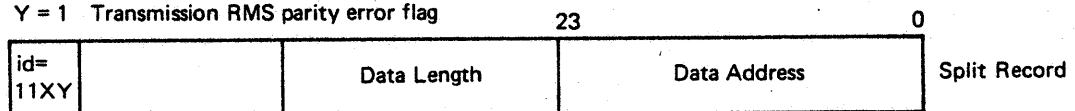
SUBPAGE BUFFER



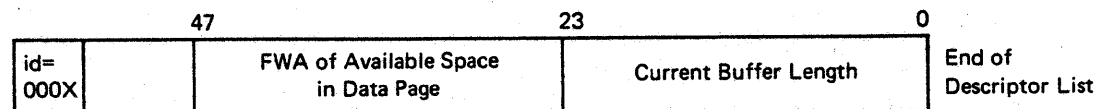
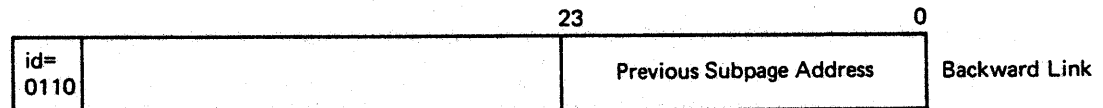
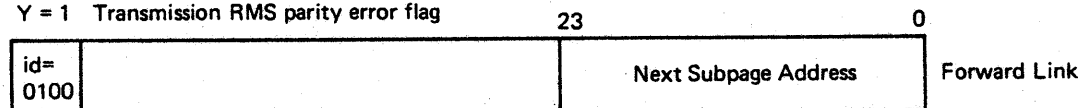
RECORD DESCRIPTORS



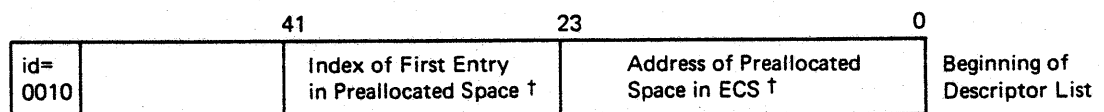
X = 1 Beginning of a new data page
 Y = 1 Transmission RMS parity error flag



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



† Used only for the library.

NOTES: SUBPAGE BUFFER

Subpage Type (bits 59-48)

Subpage Position (bits 49-48)

- 00 = continuation subpage
- 01 = first subpage in a file
- 10 = last subpage

Data Type

Bit	50	release data as read
	51	(reserved)
	52	I/O buffer (with bit 50)
	53	library file (ZZZZZ06)
	54	ECS resident file
	55	swap file
	56	auxiliary file for ECS resident random file
	57	index for random ECS file
	58	(reserved)
	59	(reserved)

id (bits 59-56)

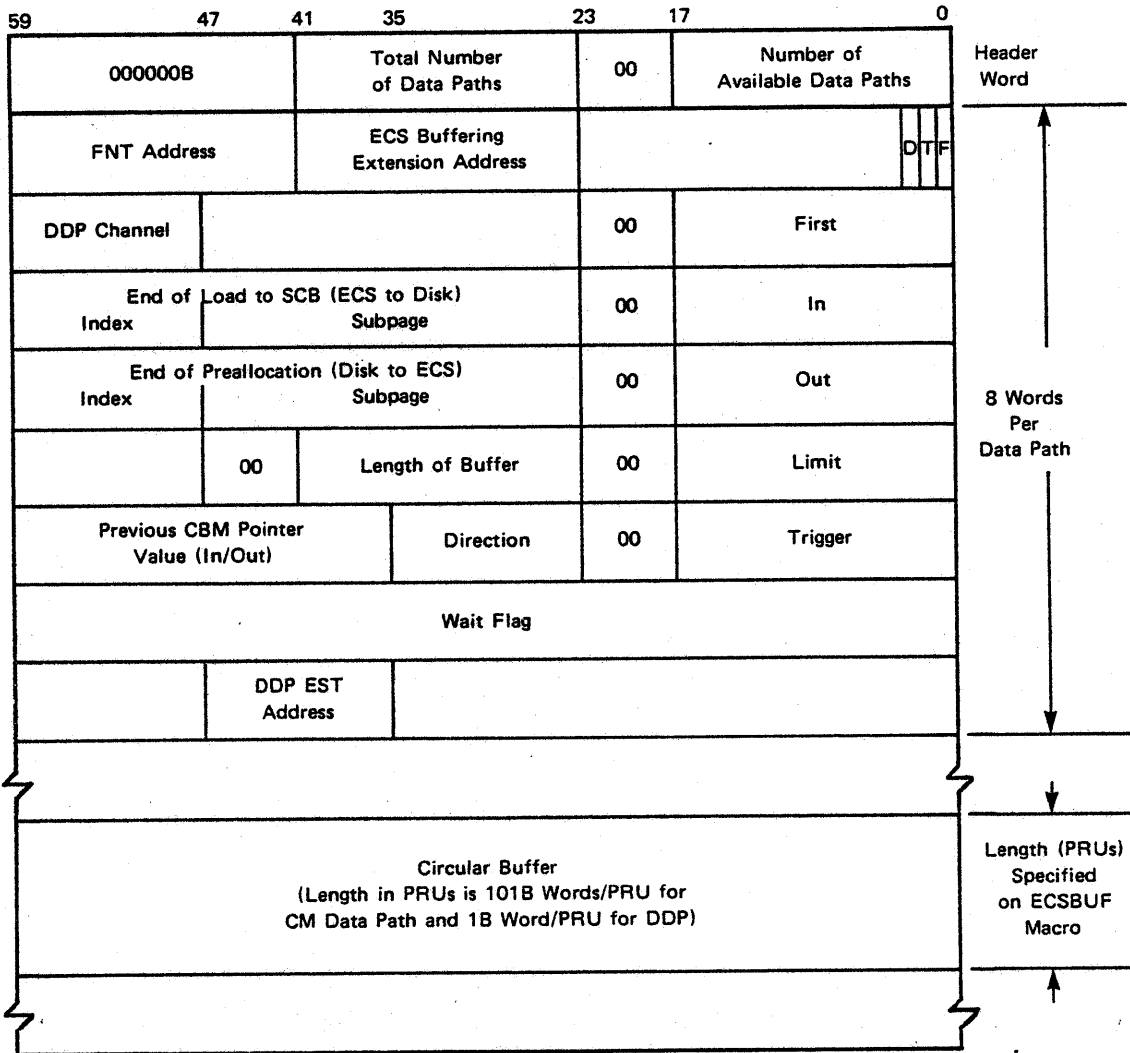
0XXX = system descriptor

- 0000 = end of list; continued on disc
- 0001 = end of list; EOI on file
- 0010 = beginning of list
- 0100 = forward link pointer
- 0110 = backward link pointer

1XXX = data descriptor

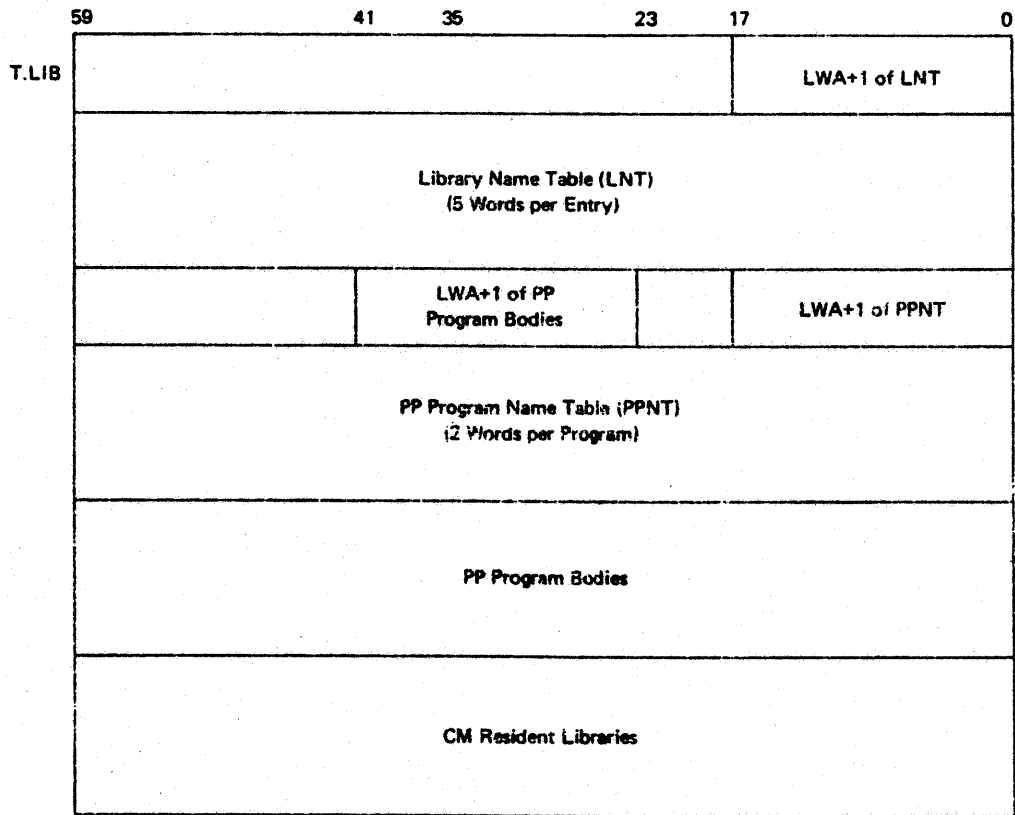
- 10XX = full record descriptor
- 11XX = split record descriptor (full record described by this and next descriptor)
- 1X0X = current data page
- 1X1X = new data page
- 1XX0 = no parity error
- 1XX1 = parity error in record

SCB (System Circular Buffer)

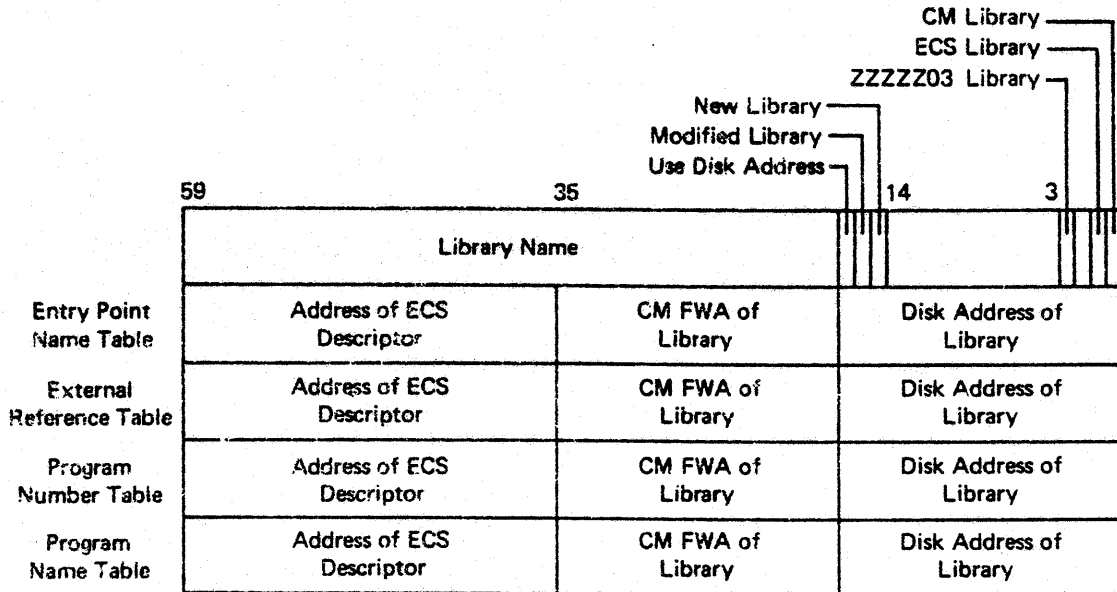


- D - Direction (0-Disk to ECS, 1-ECS to Disk)
- T - Type (0-CM Data Path, 1-DDP)
- F - Free/Busy (0-Busy, 1-Free)

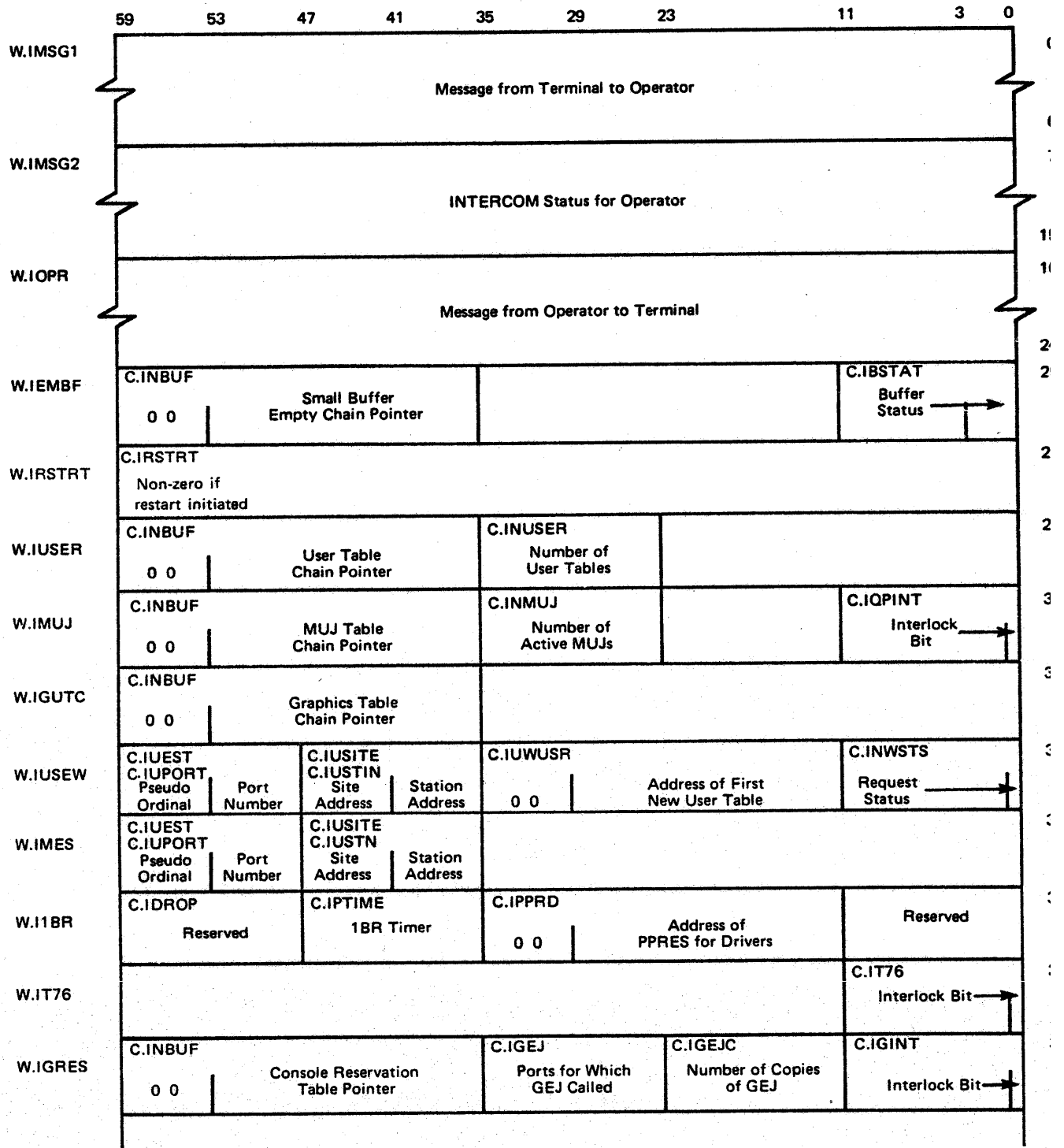
CMR LIBRARY DIRECTORY



LIBRARY NAME TABLE ENTRY



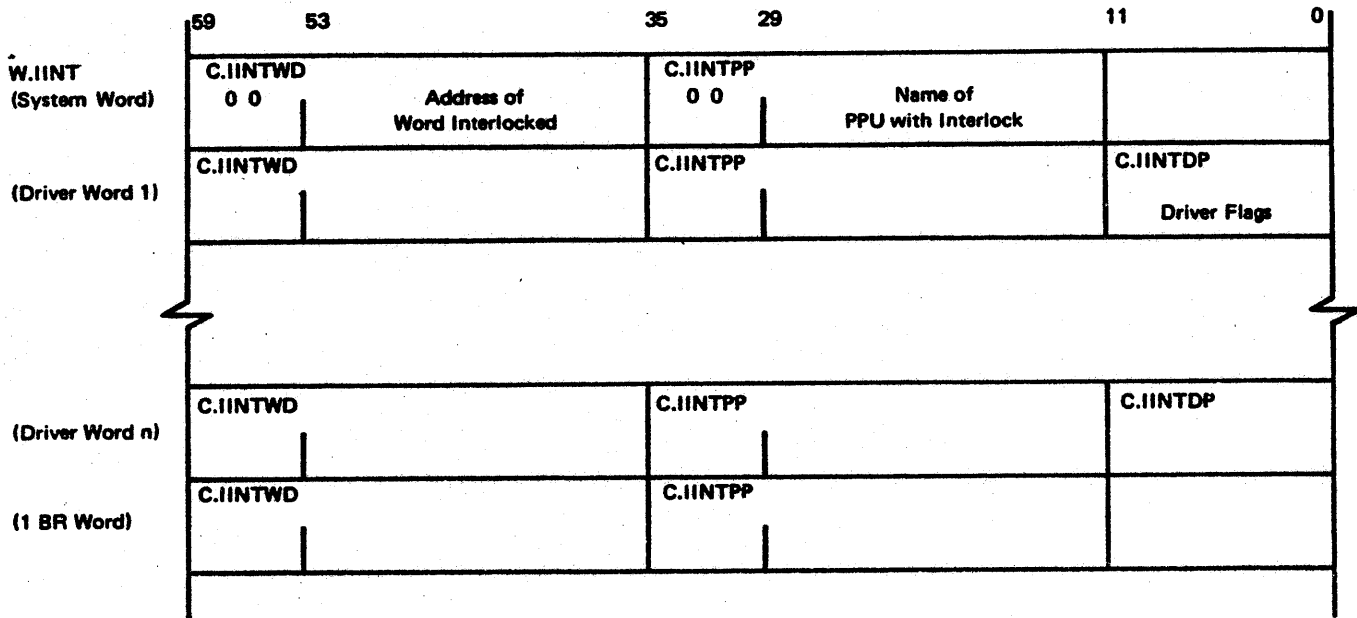
INTERCOM POINTER AND BUFFER AREA



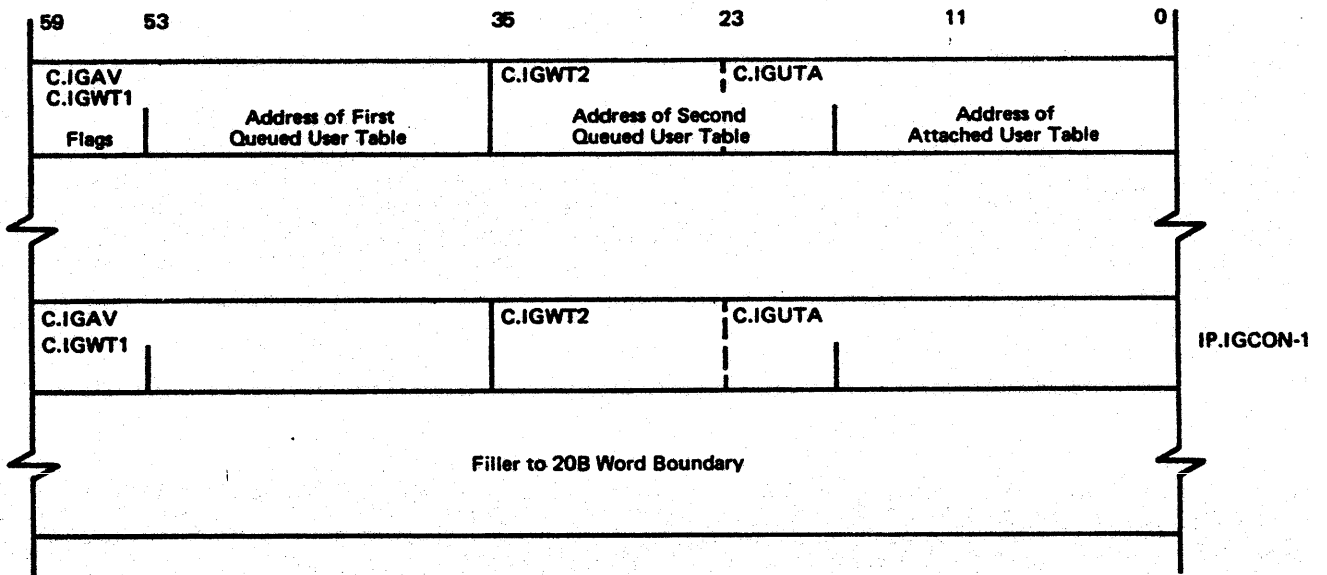
INTERCOM POINTER AND BUFFER AREA (CONTD)

	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.IPEST		Pseudo EST 1		Pseudo EST 2		Pseudo EST 3		Pseudo EST 4		Pseudo EST 5		41
W.IPES2		Pseudo EST 6		Pseudo EST 7		Pseudo EST 8		Pseudo EST 9		Pseudo EST 10		42
W.IPES3		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.IC59ZD	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)											
Only Present If LCC's in the EST	LCC 8ZZ Area											
	LCC Overlay Area											
Only Present If 2550's in the EST	NPU 8FE Area											
	NPU Connection Table											
Only Present If Graphics Defined	Console Reservation Table											
	PP Resident for Drivers											
	9ZD-Driver Dump Program											
	1LX CM Buffers											

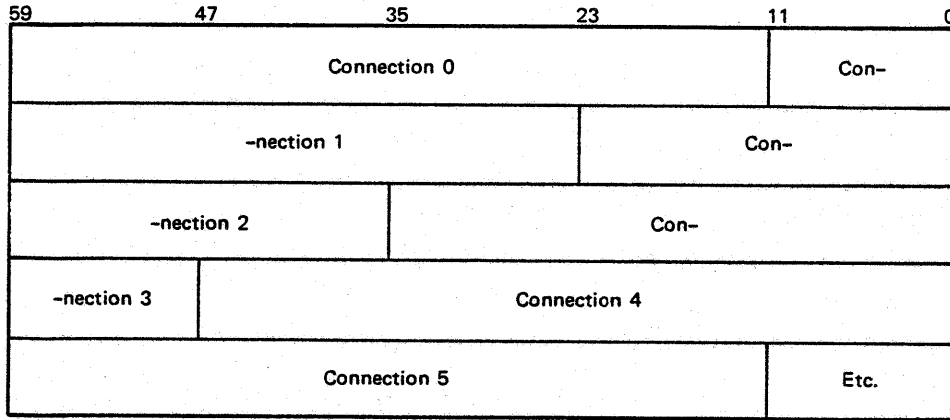
INTERCOM INTERLOCK TABLE



CONSOLE RESERVATION TABLE

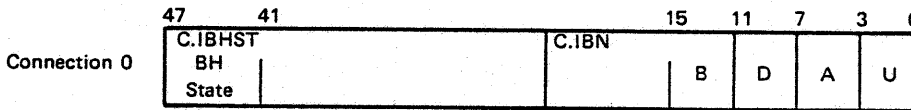


INTERCOM CONNECTION TABLE



Each Connection Table entry contains 4 bytes (48 bits).
It is used by the Front End.

CONNECTION TABLE ENTRY FORMATS



C.IBHST

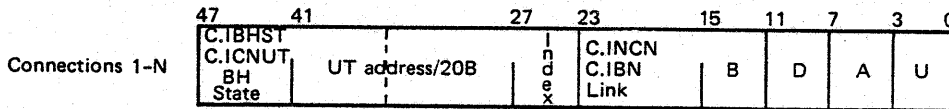
Bits	11-06	S.IBHST	Block Handler state
		L.IBHST	Block Handler state length

C.IBN

Bits	03-00	S.ILBOBN	Block Serial Number (BSN) of last BACK-ed downline Service Message (SM) block. (B)
------	-------	----------	--

C.IBN+1

Bits	11-08	S.ILSOBN	BSN of last downline SM block sent. (D)
	07-04	S.ILBIBN	BSN of last BACK-ed upline SM block. (A)
	03-00	S.ILRIBN	BSN of last upline SM block received. (U)
		L.IBN	BSN field length



C.IBHST

Bits	11-06	S.IBHST	Block Handler state
		L.IBHST	Block Handler state length

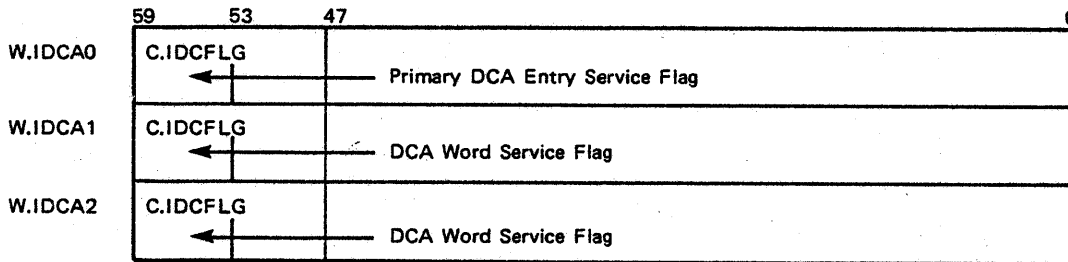
C.ICNUT

Bits	05-00		Upper 6 bits of User Table (UT) address/20B
------	-------	--	---

C.ICNUT+1			
Bits	11-04		Lower 8 bits of UT address/20B
	03-00		Stream Index
C.INCN			
Bits	11-04	S.INCN	Ordinal of next linked Connection Table entry.
C.IBN			
Bits	03-00	S.ILBOBN	Block Serial Number (BSN) of last BACK-ed downline block. (B)
C.IBN+1			
Bits	11-08	S.ILSOBN	BSN of last downline block sent. (D)
	07-04	S.ILBIBN	BSN of last BACK-ed upline block. (A)
	03-00	S.ILRIBN	BSN of last upline block received. (U)
		L.IBN	BSN field length.

INTERCOM DRIVER COMMUNICATION AREA (DCA)

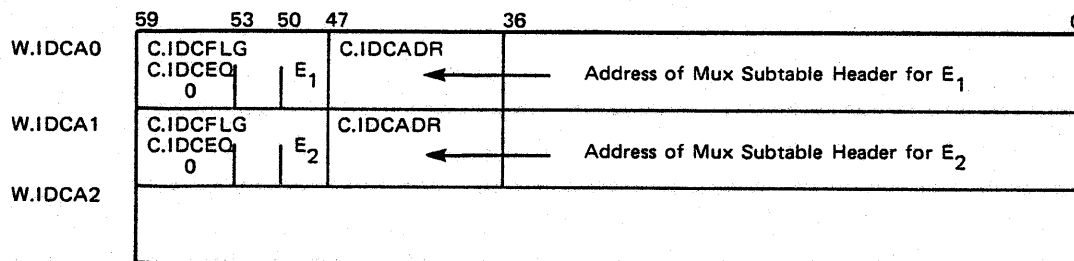
STANDARD DCA ENTRY



W.IDCA0
C.IDCFLG
 Bits 11-06 S.IDCFLG
 0=No service required from standard INTERCOM service routines for any word in this DCA entry.
 1-77B=Service required for entire DCA entry.

W.IDCA1 and W.IDCA2
C.IDCFLG
 Bits 11-06 S.IDCFLG
 0=No service required from standard INTERCOM service routines for this word.
 1-77B=Service required for this word.
 The primary DCA entry flag (in W.IDCA0) must be non-zero for service to occur.

2550 FRONT END NPU DRIVER DCA ENTRY



W.IDCA0

C.IDCFLG

Bits 11-06

S.IDCFLG

0=No service required from standard INTERCOM service routines.

L.IDCFLG

Service flag field length.
Reserved.

05-03

C.IDCEQ

Bits 02-00

S.IDCEQ

Equipment number of 1st NPU on channel (E₁).

L.IDCEQ

Equipment number field length.

C.IDCADR

Bits 11-00

Address of Mux subtable of 1st NPU on channel.

W.IDCA1

C.IDCFLG

Bits 11-06

S.IDCFLG

0=No service required from standard INTERCOM service routines.

05-03

Reserved.

C.IDCEQ

Bits 02-00

S.IDCEQ

Equipment number of 2nd NPU on channel (E₂).

C.IDCADR

Bits 11-00

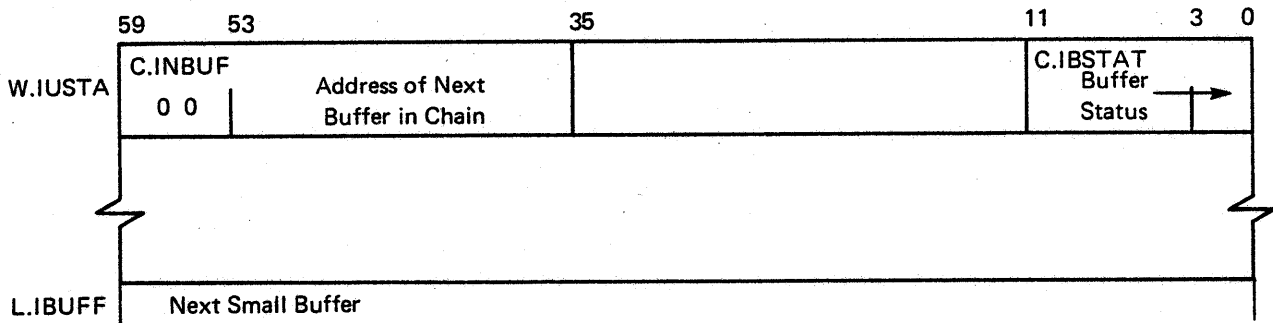
Address of Mux subtable of 2nd NPU on channel.

W.IDCA2

Reserved.

INTERCOM BUFFER AREA

SMALL BUFFERS (20B Word Boundary)



Byte C.IBSTAT

Bits 3-0 S.ITYPE

Buffer Status

- =0 Empty
- =1 User table
- =2 Data output/job card
- =3 Data input
- =4 Interactive control cards
- =5 Multi-user job table (MUJ)
- =6 Auxiliary user table (AUT)
- =7 Graphics job
- =12B Limbo buffer

USER TABLE

	59	53	47	41	35	29	23	11	5	0					
W.IUSTA	C.INBUF 0 0		Address of Next User Table			C.IUSID User ID		C.ITIME Timer		C.IBSTAT Buffer Status	0				
W.IUCMD W.IUIUP W.IUTID	C.IUCMD Command Ordinal		C.IUTID Terminal ID		C.IUIUP C.IUCCA Flags		Address of Control Card Buffer		C.IUSTAT User Status		1				
W.IUMUJ	C.IUMORD MUJ Ordinal				C.IUMJP 0 0		Address of MUJ Table		C.IUMJS MUJ Status		2				
Swap In	C.IUJDA 0 0				Address of JDT				C.IUFILE						
W.IUFST	C.IUEQC Swap Equipment		C.IUFRB Swap First RBT		C.IUFL Swap Field Length/100B		C.IUPFL Positive Field Length/100B		C.IUFILE Flags File Count		3				
W.IINPUT	C.IDINOT C.IDPPTR A				Interactive Input OUT Pointer and Byte				C.IDININ C.IDDPTR B		Interactive Input IN Pointer and Byte		C.ISIZPG Page Size	4	
W.IOTPUT	C.IDOTIN C.IDPPTR A C D				Interactive Output IN Pointer and Byte				C.IDOTOT C.IDDPTR B		Interactive Output OUT Pointer and Byte		C.ISIZLN C.IMXL E	Line Length	5
W.IDSPUT W.ISDOT W.ISDIN W.IGFLGS	Special Directive									C.IGSON C.IGFLGS		6			
W.IUSTAT W.IJBCRD W.IFLGS	C.IDVTLP Line Printer Divert ID		C.IDVTCR Card Punch Divert ID		C.IJBCRD 0 0		Address of Job Card Buffer		C.IXSTAT Export Status		7				
W.IUEQP	C.IUEST C.IUOPT EST Ordinal Port Number		C.IUSITE C.IUSTN Site Address Station Address		C.IUAUT C		Address of Auxiliary User Table		C.IUPRO A		Terminal Type	10			
W.IUDRV1	Driver Word 1 (Driver Dependent)										11				
W.IUDRV2	Driver Word 2 (Driver Dependent)										12				
W.IUAFT	C.ICUFL Current FL/100B		C.IMXFL Maximum FL/100B		C.ICUTL Current Time Limit		C.IMXTL Maximum Time Limit		C.IMXFI/C.IUVC A Re-served Maximum Files		13				
W.IUAPP W.IUACP	C.IUACS C.IUACCS Access Level		C.IUFLGS User Flags		C.IUACP Accumulated CP Time						14				
W.IULCMD	Last Command for Operator										15				
W.IUATIM	0 0		Date of LOGIN (YYDDD)				Time of LOGIN (HHMM)				16				
W.IINS	Reserved for Installations										17				

USER TABLE DRIVER WORDS

6671/6676 DRIVER

	59	47	35	29	23	17	11	0	
W.IUDRV1	C.IUDRET Return Address In PPU		C.IUDCNT Number of Character Left on CRT		C.IDSTAT C.IUDRXT RXT Counter Status		C.IUDECD C.IUDCCD Last Con- trol Code Last E Code		C.IUDSTA Station Info/ Message Type
W.IUDRV2	Partially Transmitted Output Word								
6671	-----								
6676	Current Input/Output Word								

6673/6674 DRIVER

	59	47	35	23	11	0				
W.IUDRV1	C.IHOPAS Pass Counter		Driver Temporary Storage							
W.IUDRV2	C.ISYNC Number of Sync Errors		C.ICYCL Number of Cyclic Errors		C.IIO Number of I/O Errors		C.IDSEQ Number of Sequence Errors		C.ITOTRX Number of Retransmissions	

LCC DRIVER

	59	53	47	41	35	29	23	17	11	5	0
W.IUDRV1	C.ISTST Output State Input State		C.IBCNT Input Byte Count		C.IBPOS C.IDININ Physical Line Input IN Pointer and Byte				C.IHCNT Number of Character on CRT		
W.IUDRV2	First AUT Stream State Number		Second AUT Stream State Number		Third AUT Stream State Number		Fourth AUT Stream State Number		Fifth AUT Stream State Number		

2550 DRIVER

	59	53	47	39	35	11	0
W.IUDRV1	C.ISTST Stream State		C.ISTFL Connection Number		C.IDOTOT Temporary Pointer, Display Output		C.IHCNT Number of Characters on CRT
W.IUDRV2	C.IBSS1 Batch Stream States		C.IBSS2 C.ISTEX		C.IDININ C.IXOTOT Temporary Pointer		C.IHCNT Number of Characters on CRT, Last Write

NOTES: USER TABLE

W.IUSTA(0)		C.IBSTAT(4)	
Bit	4	S.IDISC	Terminal disconnected
	5-6	S.ILOGO	00 = user logged out 01 = user logged in 10 = auto-logout requested 11 = auto-logout in progress
	7-8	S.ISTATE	00 = transmission state 01 = waiting input 10 = waiting output 11 = active or assigned to control point
	9	S.IRDIS	Request disconnect
	10	S.IUTAPE	Request paper tape reading
	11	S.IABRT	Abort request
W.IUIUP(1)		C.IUCCA(2)=C.IUIUP(2)	
Bit	6	S.IUCCP	Control cards moved to control point area
	7	S.INCT	Command in ICI table
	8	S.IUECP	Editor control statements processed
	9	S.IUEXS	Execution started
	10		Unused
	11	S.IUEDC	Buffer contains control cards sent from EDITOR
W.IUCMD(1)		C.IUSTAT(4)	
Bit	0-5	S.IUHDIS	Used by IDS to determine type of H display wanted
	6	S.ICMES	MES Active
	7	S.ICACT	ICI Active
	8	S.IBCTHW	Allow commands before LOGIN
	9	S.IMPORT	IMPORT on at 200 UT
	10	S.INOCOM	Do not issue command
	11	S.ICTAPE	Paper tape on flag
W.IUMUJ(2)		C.IUMJS(4)	
Bit	0	S.IMNUS	New user
	1	S.IMRUN	RUN command in progress
	2	S.IMDIS	Reconnected after disconnect
	3-6		(Unused)
	7	S.IMBKS	Break sent
	8	S.IMLGS	Logout sent
	9	S.IMWO	Waiting for output to complete
	10	S.IMWI	Waiting for input
	11	S.IMUJ	Attached to MUJ
W.IUFST(3)		C.IUFILE(4)	
Bits	0-5	S.IUCNT	Count of FNT entries in swap file
	6	S.IUEOE	End of execution
	7	S.IUDMP	SAVEFL flag
	8	S.IUPS	Pause bit
	9	S.IUECS	Swap file on ECS
	10	S.IUFNT	FNT to be associated on next swap
	11	S.IURED	REDUCE flag

NOTES: USER TABLE (CONT'D)

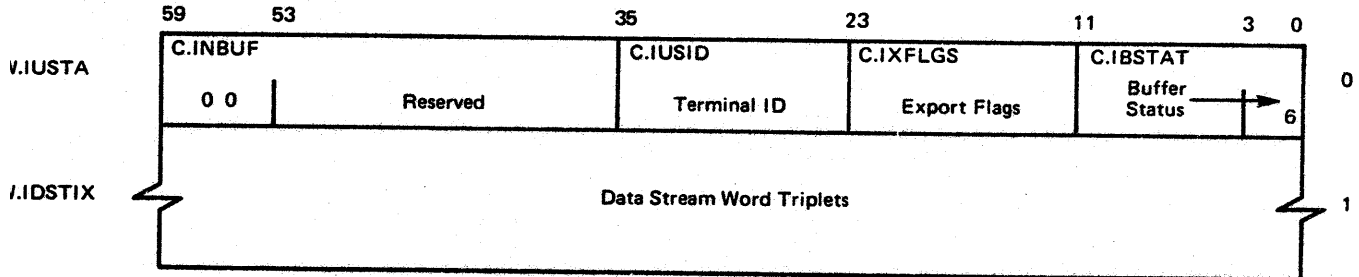
W.IINPUT(4)			C.IDPPTR(0)	
A	Bit	11	S.IINTLK	Interlock bit for 3TT and I/O macros
		6-8	S.IBYTE	Byte position in current word
W.IOTPUT(5)			C.IDDPTR(2)	
B	Bit	10	S.IRLS	Input line no longer in progress
		9	S.IDRACT	Driver not finished
		6-8	S.IBYTE	Byte position in current word
W.IOTPUT(5)			C.IDPPTR(0)	
A	Bit	11	S.IINTLK	Interlock bit for 3TT and I/O macros
C	Bit	10	S.IPWOFF	Page-wait-off flag for processor
D	Bit	9	S.IIMF	1IM interlock
		6-8	S.IBYTE	Byte position in current word
			C.IDDPTR(2)	
B	Bit	11	S.IPRNOP	Driver not finished or previously inoperative
		6-8	S.IBYTE	Byte position in current word
			C.ISIZLN(4)	
E	Bit	11	S.ILNCHG	Line size change bit
W.IGFLGS(6)			C.IGFLGS(4)	(274 graphics only)
	Bit	9-11	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 interrupt output
W.IGFLGS(6)			C.IGSON(4)	(High speed INTERCOM terminal only)
	Bit	8	S.IXOFF	Terminal off line
		7	S.IGSON	274 graphics streams defined
		6	S.IGRMUX	Graphics MUX
		0-5	S.IGDSON	Graphics data stream number
W.IUSTAT=W.IFLGS(7)			C.IXSTAT(4)	
	Bit	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 by 1CI
		7	S.IOVFL	Message overflow (1XP)
		6	S.IMAX	Maximum message buffers assigned (1XP)
		5	S.IDVTLF	Divert print files
		3	S.IDVTCP	Divert punch files

NOTES: USER TABLE (CONT'D)

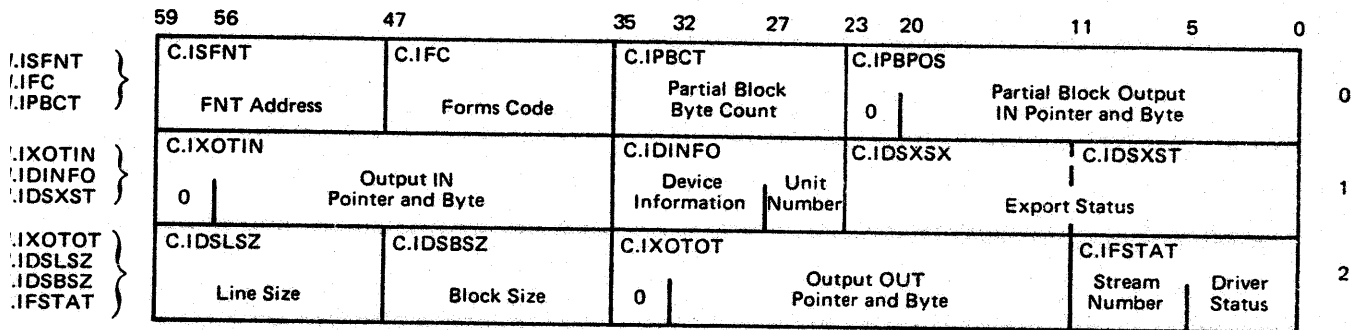
W.IUEQP(10)		C.IUEST(0)	
	Bits	8-11	S.IUEST Pseudo EST ordinal
		0-7	S.IUPORT Port number
		C.IUSITE(1) = C.IUSTN(1)	
	Bits	6-11	S.IUSITE Site address
		0-5	S.IUSTN Station address
		C.IUAUT(2)	
C	Bit	11	S.IUAUTR AUT request bit
		C.IUPOR(4)	
A	Bit	11	S.IUNOP Inoperative bit
W.IUAFT(13)		C.IUVE(4) = C.IMXFI	
A	Bit	11	S.IUVC VC carriage control
		6-10	Reserved
		0-5	Maximum number of files
W.IUAPP(14)		C.IUFLGS	
	Bit	11	S.IUUNR Login with unrestricted password
		10	S.ITLOK Lock bit
		9	S.IRLOK Request job to be locked out
		8	(unused)
		7	S.INOLK Job locked out
		6	S.IUREU Reduce mode flag
		0-5	S.IUSSW Sense switch setting changes

AUXILIARY USER TABLE

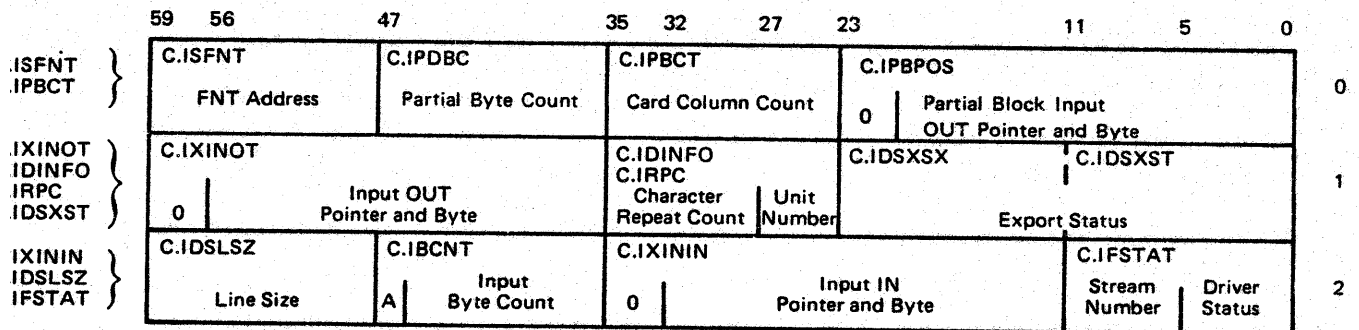
The auxiliary user table is for batch terminals on 7077/791 and 6671.



OUTPUT DATA STREAM WORD TRIPLET



INPUT DATA STREAM WORD TRIPLET



NOTES: AUXILIARY USER TABLE

W.IUSTA(0)		C.IXFLGS(3)	
		S.ITXOUT	1LX taking 200 UT out of transmission mode
W.IDINFO(1)		C.IDINFO	(for output streams)
Bits	7-11	S.IFCMO	FCM ordinal
	4-6	S.ITRAIN	Train type
			=3 B4
			=4 B6
			=5 A6
			=6 A9
	03	S.IDSLUN	Logical unit number
			=1.2 CR1 and 2
			=3 CP
			=4-7 LP1-4

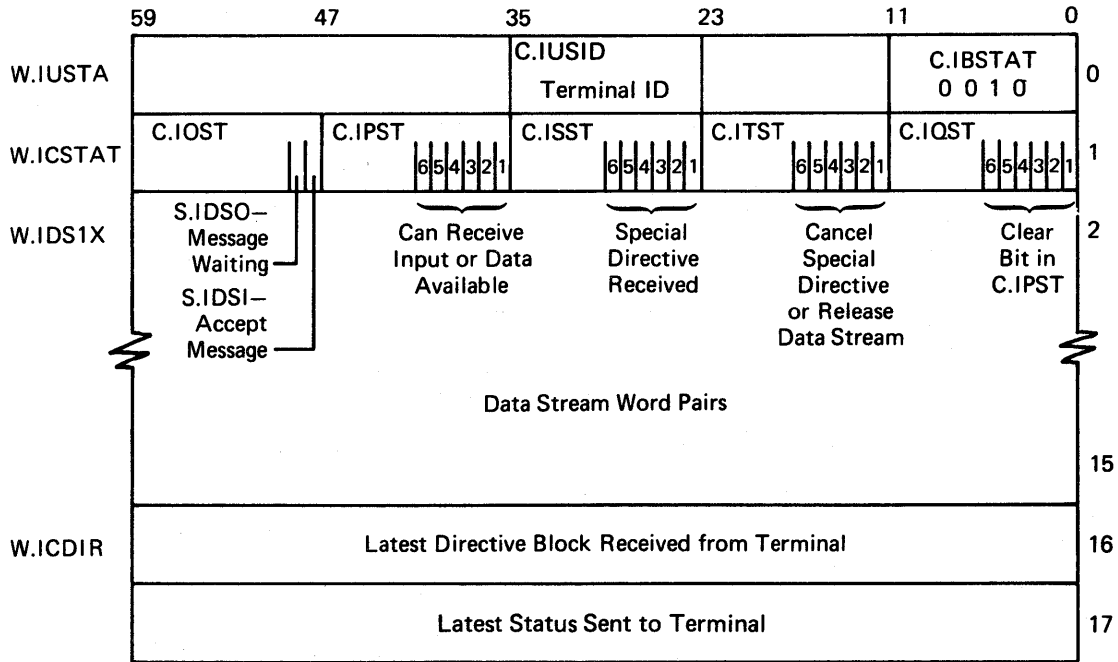
NOTES: AUXILIARY USER TABLE (CONT'D)

W.IDSXST(1)		C.IDSXST(3)	(for output stream)
Bit	8	S.ISNT	No banner/lace card
	6-7	S.IIC	Internal code =0 Display =1 ASCII =2 Binary
	9-11	S.IEC	External code =1 SB =2 80COL =3 B4 =4 B6 or 026 =5 A6 or 029 =6 A9 or ASCII
	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
	0-1	S.IVCC	V carriage control =00 No V detected =01 V detected; check for VC =10 V detected; no C =11 VC detected
		C.IDSXST(4)	(for output file)
Bit	1	S.INEEDB	Start next PRU with a blank
	0	S.IREADF	READ,filename in progress
		C.IDSXST(4)	(for output file)
Bit	11	S.IWAITX	Wait-for-driver-stop
	10	S.IWEQJ	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
	2	S.IETX	ETX sent
	1	S.ISTOP	Request driver stop stream
	0	S.IXABT	Request driver abort stream
		C.IDSXST(4)	(for input stream)
Bit	11	S.IWAITX	Wait-for-driver-stop
	10	S.IWEOJ	Wait-end-of-job
	9	S.IBIN	Binary mode
	8	S.IASC	ASCII mode

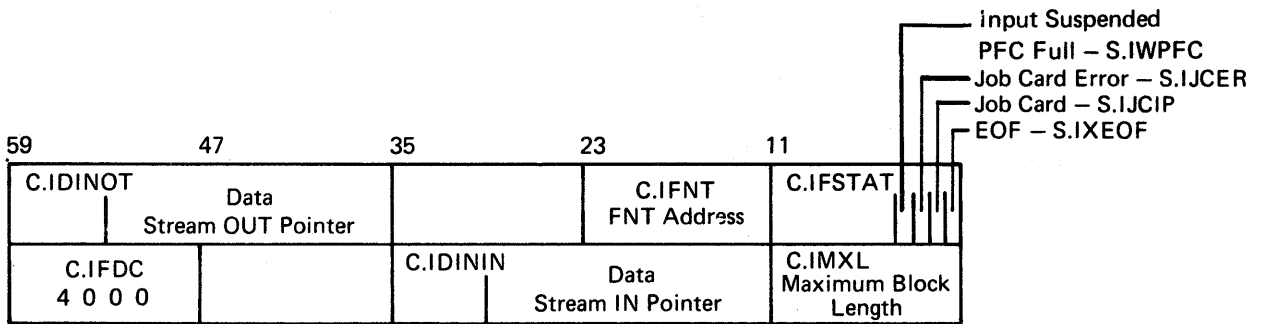
NOTES: AUXILIARY USER TABLE (CONT'D)

W.IDSXST(1)	C.IDSXST(4)	(for input stream)
Bit 7	S.IJ CER	Error
5	S.IX EOF	End-of-file
4	S.I OFF	Request driver off stream
3	S.I WPFC	Wait PFC full
1	S.I STOP	Request driver stop stream
0	S.I XABT	Request driver abort stream
W.IXININ(2)	C.IBCNT(1)	(for input stream only)
A Bit 11	(S.I ETB)	=1 ETB, =0 ETX sent
W.IFSTAT(2)	C.IFSTAT(4)	(for both input and output)
Bit 6-9	S.IDSN	Data stream number/site address
5	S.I XABTI	Abort issued
4	S.I OFF	Stream off
3	S.I NREDY	Device not ready (mode 4)
2	S.I ETX	ETX block sent
1	S.I STOP	Stream stopped
0	S.I XABT	Stream aborted

HIGH SPEED AUXILIARY USER TABLE



INPUT DATA STREAM WORD PAIR



MULTI-USER JOB TABLE

	59	47	35	29	23	11	0
W.IUSTA	C.INBUF Address of Next MUJ Table		C.IUSID MUJ ID		C.ITIME ICI Timer for Swaps		C.IBSTAT 0 0 0 5
W.IUCMD			C.IUCCA A Address of Control Card Buffer				
W.IMQP	C.IMUC Total Number of MUJ Users	C.IMAC Activity Count	C.IMSTAT MUJ Status Byte		C.IMSIF Swapin Flag	C.IMSOF Swapout Flag	
W.IUFST	C.IUJDA MUJ JDT Address						C.IUFILE ↑ End of Execution
W.IMDES	C.IMED EDITOR Flag	C.IMSID Swapin Delay	C.IMSOD Swapout Delay	C.IMFL MUJ FL/100B			
W.IMTIN	C.IMTIP Address of TERMIN		C.IMTIL Length of TERMIN		C.IMTHDR Address of TERMIN/TERMOUT Header		
W.IMTOUT	C.IMTOP Address of TERMOUT		C.IMTOL Length of TERMOUT		C.IMTHDR Address of TERMIN/TERMOUT Header		
	(Reserved for CDC)						
W.IMNAME	MUJ Name in Display Code						
W.IINS	(Reserved for Installation)						

W.IMQP(2) C.IMSTAT

0001 = Waiting for I/O
0003 = MUJ Active

A S.IUCCP Control Cards moved to CPA Flag

274 INTERACTIVE GRAPHICS SYSTEM (IGS) USER TABLE

	59	53	47	35	29	23	11	0	
W.IUSTA	C.INBUF Address Next User Table In Chain			C.IUSID User ID			C.IBSTAT Flags 7 0 1 1 1		0
				C.IUCCA A Address of Control Card Buffer					1
									2
W.IUFST	C.IUIDA Job Descriptor Addr.			C.IUCLAS Job Class Before Entering IGS Queue					3
W.IINPUT	C.IDINOT Graphics Input Data Out Pointer			C.IDININ Graphics Input Data In Pointer					4
W.IOTPUT	C.IDOTOT Graphics Output Data Out Pointer			C.IDOTIN Graphics Output Data In Pointer					5
W.IGFLGS							C.IGFLGS Graphics Flags		6
									7
W.IGHSU W.IUEQP	C.IUPORT EST Ordinal		Port No.	C.IGHSU Pointer to High Speed User Table			C.IUMSG 0 0 1 0		10
									11
									12
									13
W.IUGCAC W.IUGEF				C.IUGCAC Console Access		C.IUGEF Error Flags	C.IUGEN Console No.		14
W.IUACP	C.IUACP Control Card Buffer Address								15
									16
									17

NOTES: 274 IGS USER TABLE

W.IGFLGS(6)

Bit	5	S.IG3TT
	1	S.IGQUE
	2	S.IGSNON
	3	S.IGDRP
	9-11	S.IGCON

C.IGFLGS(4)

3TT Interlock flag
Job queued for first console flag
SIGNON user table flag
IGS job termination flag
Console No. for which job is queued

RECORD BLOCK TABLE ENTRY

The RBT entries exist in the highest addresses of CM above the jobs 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	

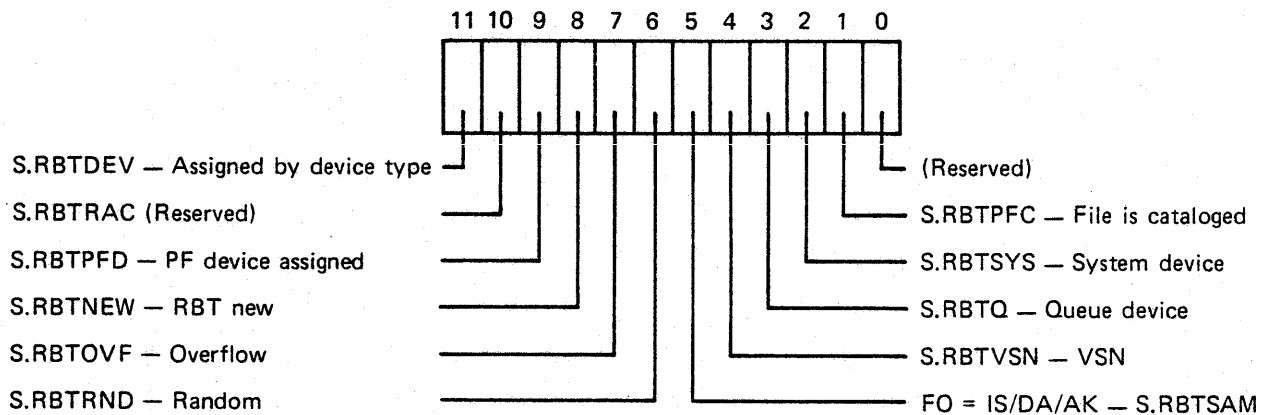
OTHER WORD PAIRS EXCEPT OVERFLOW

59	47	38	35	23	11	0
CRBTWPL 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

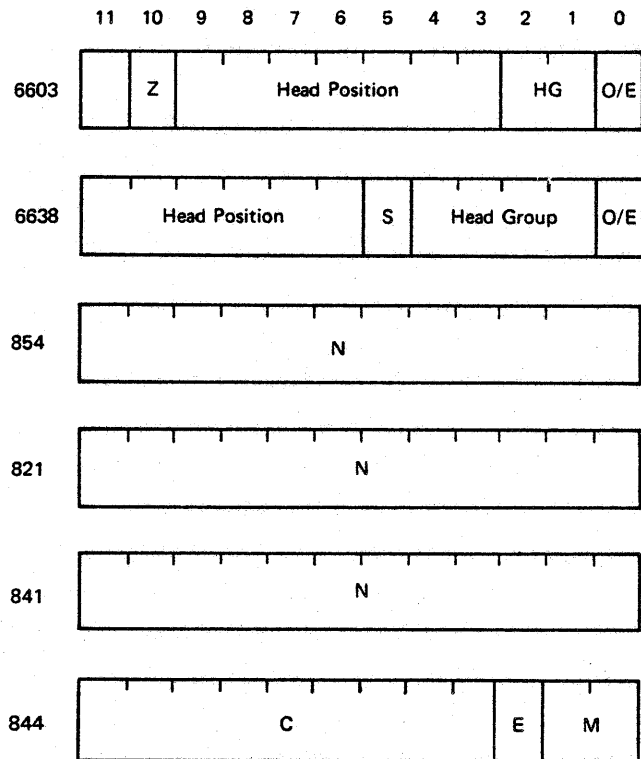
59	47	38	35	29	23	11	0
C.RBTWPL Next Word Pair	777	0	C.RBTMST MST Ordinal		End of Volume PRU + 1	C.RBTODD DAM Ordinal	
C.RBTAUS PRUs/RB	C.RBTVSN Volume Serial Number					0 0 0 0	

C.RBTBIT



RECORD BLOCK TABLE BYTE MINUS ONE

COMPUTATION OF PHYSICAL ADDRESSES FOR DEFAULT ALLOCATION STYLES



Z Zone 0 = Outer
 1 = Inner

HG Head Group

O/E Odd/Even 0 = Even
 1 = Odd

S Stack

SG Subgroup

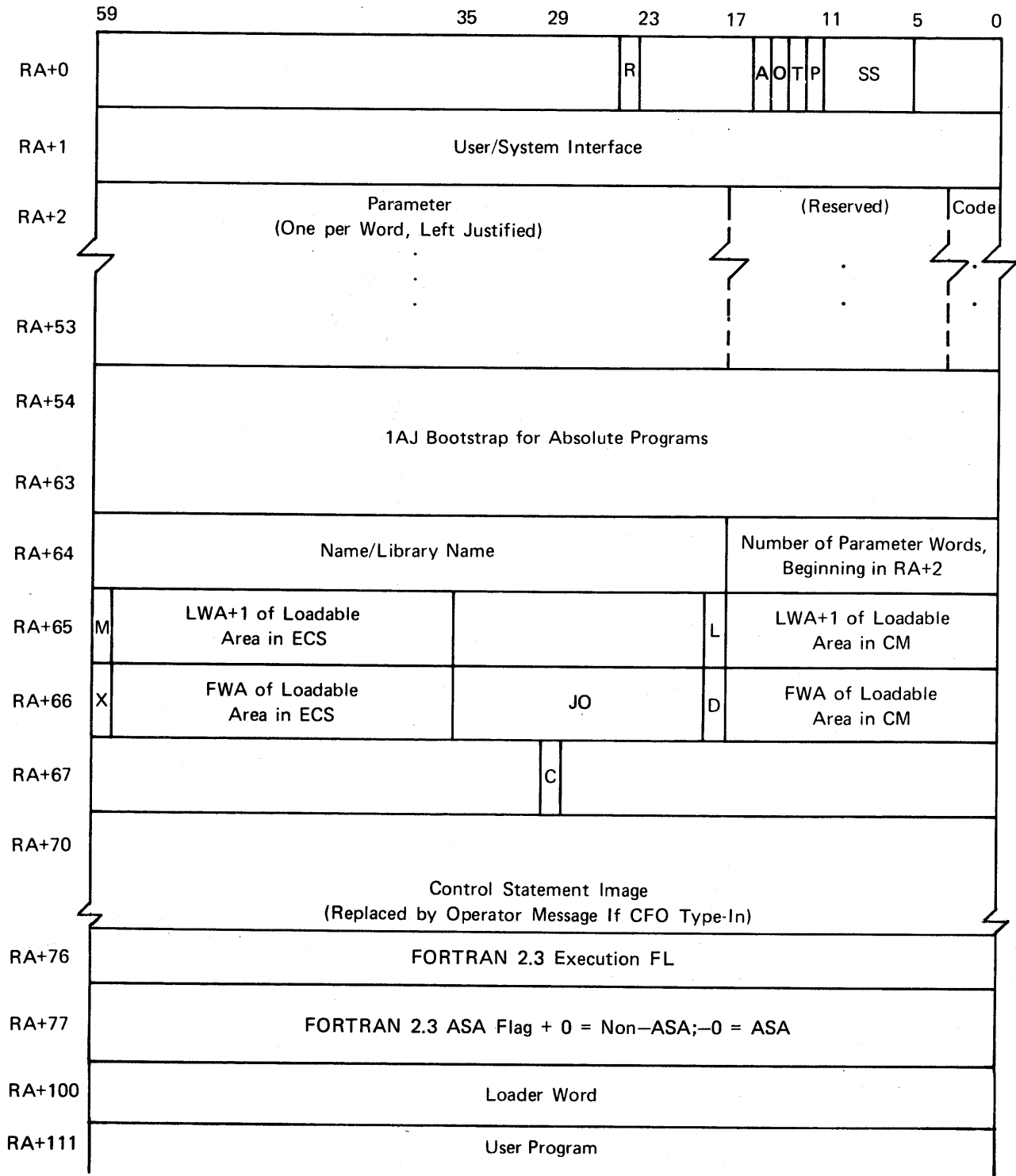
COMPUTATION OF PHYSICAL ADDRESSES (CONT'D)

821	Cylinder	=	$\frac{N}{2}$
	Track	=	0
	Sector	=	0 if N is even 1 if N is odd
841	Cylinder	=	$\frac{N}{5}$
	Track	=	8 * remainder of $\frac{N}{5}$ modulo 20
	Sector	=	0 if remainder of $\frac{N}{5} \leq 2$ 1 if remainder of $\frac{N}{5} < 2$
844	Cylinder	=	C
	Track	=	$\frac{114 * M}{24}$
	Sector	=	E + remainder of $\frac{114 * M}{24}$ (E = 0 for even/E = 1 for odd)
854	Cylinder	=	$\frac{N}{13}$
	M	=	remainder of $\frac{N}{13}$
	S	=	remainder of $\frac{M * 24}{159}$
	Track	=	$\frac{S}{16}$
	Sector	=	remainder of $\frac{S}{16}$

SECTION 2

JOB CONTROL POINT TABLES

RA COMMUNICATION AREA



NOTES: RA COMMUNICATION AREA

R	Job dependency recheck bit
A	Job swapout to operator action queue flag (1=job will be placed under operator action queue upon swapout regardless of job origin)
O	CFO flag (1 = accept comment from operator)
T	Storage move flag (1 = move being attempted)
P	Pause flag (1 = control point pausing)
SS	Sense switches
CODE	00 = Continuation
	01 = Comma
	02 = Equals sign
	03 = Slash
	04 = Left parenthesis
	05 = Plus sign
	06 = Minus sign
	07 =
	10 = Semi-colon
	11 =
	12 =
	13 = (reserved)
	14 =
	15 =
	16 = Other
	17 = Termination
L	Library/file flag (1 = name is library name)
X	XJ flag: if XJ = 1, and XJ can be issued
C	LDV completion flag (bit 29)
D	DIS RSS flag (bit 18)
M	CMU Bit; if M=1 and CMU can be issued
JO	Job origin:
	0 = System
	1 = Batch
	2 = Remote batch
	3 = Terminal

FILE ENVIRONMENT TABLE

	59	53	47	41	35	32	29	23	17	13	8	0			
	LOGICAL FILE NAME										LEVEL NO.	ERROR CODE	CODE/STATUS	0	
	DEVICE TYPE	R	U	E	M	X	X	E	N	I	I	DISPOSITION CODE	FET LENGTH -5	FIRST POINTER	1
	0										IN POINTER			2	
											OUT POINTER			3	
	FNT POINTER	RECORD BLOCK SIZE				PRU SIZE				LIMIT POINTER				4	
6RM →	Reserved for 6RM												5		
CPC →	FWA WORKING STORAGE AREA						LWA+1 WORKING STORAGE AREA						6		
	DETAIL ERROR CODE (XP=1)	POINTER TO FET EXTENSION (XP=1)				UBC	MLRS (S/L TAPES ONLY)					6			
	RECORD REQUEST/RETURN INFORMATION (RANDOM RMS ONLY)												6		
6RM →	Reserved for 6RM												7		
CPC →	RECORD NUMBER (CPC)					SCOPE INDEX LENGTH				FWA OF SCOPE INDEX				7	
6RM →	6RM FET EXTENSION (XP=1)												8		
CPC →	CPC EOI ADDRESS						CPC ERROR EXIT ADDRESS						8		
XL=1	LABEL ERROR CODE				LENGTH OF LABEL BUFFER				FWA OF LABEL BUFFER				9		
XL=0	FIRST 10 CHARACTERS OF FILE LABEL NAME												9		
XL=1	(RESERVED)												10		
XL=0	LAST 7 CHARACTERS OF FILE LABEL NAME								POSITION NUMBER				10		
XL=1	(RESERVED)												11		
XL=0	EDITION NUMBER	RETENTION CYCLE				CREATION DATE						11			
XL=1	(RESERVED)												12		
XL=0	MULTI-FILE SET NAME								REEL NUMBER				12		
							RESIDUAL SKIP COUNT		PERM BITS	LENGTH OF EXTENSION L				L	
													L		

SCOPE CIO CODES IN OCTAL

All codes are shown for coded mode operations; add 2 for binary mode. Example: 010 is coded READ, 012 is binary READ.

000	RPHR	170	CLOSE,UNLOAD
004	WPHR	174	CLOSE,RETURN
010	READ	200	READC
014	WRITE	204	WRITEC
020	READSKP	210	READLS
024	WRITER	214	REWRITE
030	-	220	-
034	WRITEF	224	REWRITER
040	BKSP	230	-
044	BKSPRU	234	REWRITEF
050	REWIND	240	SKIPF
054	-	244	-
060	UNLOAD	250	READNS
064	-	254	-
070	RETURN	260	READN
074	-	264	WRITEN
100	OPEN,NR	270-274	-
104	OPEN WRITE,NR	300	OPEN,NR
110	POSMF	304-324	-
114	EVICT	330	CLOSER
120	OPEN,NR	334	-
124	-	340	OPEN
130	CLOSE,NR	350	CLOSER
134	-	354-364	-
140	OPEN	370	CLOSER,UNLOAD
144	OPEN WRITE	374	CLOSER,RETURN
150	CLOSE	400-474	-
154	-	500-574	Reserved for installations
160	OPEN	600	Reserved for KRONOS READEI
164	-	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 3.4 product set file.

NAME	3.4 PRODUCT SET	USED FOR
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)	
ZZZZZ10	(EDITLIB)	Program name table spill file
ZZZZZ11		External reference table spill file
ZZZZZ12		External reference collection spill file
ZZZZZ13		Library or deadstart program collection file
ZZZZZ14		Scratch
ZZZZZ15		PP program name table spill file
ZZZZZ16		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	(CDC reserved)	

LOCAL FILE NAMES (CONT'D)

ZZZZAA-A9	(Index Processor)	
ZZZZBA-B0	(Index Processor)	
ZZZZC3	(CDC Special Systems)	
ZZZZC4	(CDC Special Systems)	
ZZZZCB	(DDL)	Scratch file
ZZZZCC	(DDL)	Scratch file
ZZZZCD	(DDL)	Scratch file
ZZZZCP	(INTERCOM)	Copy permanent files
ZZZZDB	COBOL Debug File	
ZZZZDD	(System Dynamic Dump)	Core image dump file
ZZZZDF	(6RM/LOADER)	File control card processor
ZZZZDM	(SCOPE)	FNT used by SPM to access DAM
ZZZZEF	(6RM)	Error message file
ZZZZFC	(FTN4)	Symbolic object code file
ZZZZI1	(ALGOL)	Scratch file
ZZZZI2	(ALGOL)	Scratch file
ZZZZIN	(INTERCOM Utility/QU)	Connected files
ZZZZL1-L9	(LDCMR)	Scratch files
ZZZZOP	(FTN4/COMPASS)	
ZZZZOU	(INTERCOM Utility/QU)	Connected files
ZZZZQU	(QUERY/UPDATE)	
ZZZZQ1-Q6	(QUERY/UPDATE)	
ZZZZPA	(PFM)	Scratch file
ZZZZPB	(PFM)	Scratch file
ZZZZPC	(PFM)	Attached RBTC
ZZZZPD	(PFM)	Attached PFD
ZZZZPE	(PFM)	Reserved
ZZZZPF	(PFM)	Attached PF
ZZZZPG	(PFM)	Reserved
ZZZZPK	(MMF-PFM)	Multimainframe packet file
ZZZZPT	(PFM)	PF dump tape
ZZZZPW	(PFM)	Attached PF DUM
ZZZZRE	(INTERCOM)	Restricted passwords
ZZZZRL	(FTN4/COMPASS)	
ZZZZRM	(FTN4/COMPASS)	
ZZZZRN	(PAGE Utility)	Interim random page file
ZZZZSA-SD	(SIFT)	
ZZZZSE	(EDITOR)	
ZZZZSF	(EDITOR)	
ZZZZSG	(EDITOR)	
ZZZZSH	(EDITOR)	
ZZZZUN	(INTERCOM)	Unrestricted passwords
ZZZZVx-Zx	(Installations)	
ZZZZECS	EDITLIB	System ECS resident library creation job (permanent file)

ENTRY POINT NAMES

AGxxxxx }	ALGOL
ALxxxxx }	
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
FMxxxxx	FORM
FTxxxxx	RUN
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 }	
OTxxxxx }	OPTIMA
PLxxxxx	PL1
PTxxxxx	PERT/TIME
QUxxxxx	QUERY UPDATE
RMxxxxx	6RM
SCxxxxx	SCOPE
SIxxxxx	SIMSCRIPT
SMxxxxx }	
SOxxxxx }	SORT/MERGE
SSxxxxx	SIMSCRIPT
SUxxxxx	SIMULA
Uxxxxxx }	
Vxxxxxx }	
Wxxxxxx }	
Xxxxxxx }	
Yxxxxxx }	
Zxxxxxx }	
	Reserved for Installation

FET CODES

WORD 0 – Error Codes

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 multi-file 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	47	(R)	Process SCOPE index if OPEN/CLOSE; else random read/write
	46		Reserved
	45	(UP)	User processing at end of volume
	44	(EP)	User processing on error condition
	43		Reserved
	42	(IN)	INTERCOM (multi-user job or graphics)
	41	(XL)	Extended label processing
	40	(XP)	Extended error processing
	39	(EC)	Disallow automatic allocation of ECS buffer
	38	(NS)	File has non-standard labels; processing of label records is left to user
	37	(IIM)	Inhibit implicit MOUNT

WORD 6

Detail Error Codes (bits 48-59)

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 1-77 (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	Multi-file set ill-formed
32	Write attempt on protected volume
33	Write at 200 bpi not allowed on 66x tape drive
35	Multi-file name not found on multi-file device
36	Next volume unknown
37	File not allowed on assigned device

Codes 100-177 (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 200-277 (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

Codes 400-477 (octal) are errors resulting from hardware failure between the PPU and the physical tape unit. These codes and subsequent errors are as follows:

Error Code (Octal)	Data Path Error
400	Hardware – 668x malfunction
401	Hardware – MMTC memory parity error
402	Hardware – 6681 failed, no data on IAN
403	Hardware – transmission parity error
404	System error

Codes 1000-1005 (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 6000-7777 (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, etc.)
- 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 non-user 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.

FET CODES (CONT'D)

WORD 6 – Error Codes (cont.d)

Hardware Malfunction

- 0050 MMTC memory parity error
- 0051 Transmission parity error

Position Uncertain

- 0100 Valid data probably destroyed – highly unlikely that a close will be successful
- 0101 Valid data probably intact – likely that a close will be successful

FILE INFORMATION TABLE

59											53											47											41											35											29											23											17											11											5											0										
LFN Logical File Name																						Reserved																						0																																																																												
RL Current Record Length											P	M	FO			N	D	X	BT		B	C	D	RT		D	K	I	PD		FDT File Description Table											1																																																																														
PTL Partial Transfer Length											OF		VF	CF		LT	ULP		FP				LX Address of Label Routine											2																																																																																						
HL Header Length of T Record MNR Minimum Record Length											BFS Buffer Size (in Words)											DX Address of End-of-Data Routine											3																																																																																							
TL Trailer Length of T Record											F	N	E		SES				IRS				EX Address of Error Routine											4																																																																																						
VNO	ECT Error Count				ERL Error Limit				F	S	S	E	S	O	C	F	W	O	L	C	R	K	N	E	H	LVL		FWB FWA of User Buffer											5																																																																																	
FL Length of F/Z Record MRL Maximum Record Length											Reserved											C	M	EO		WSA (WSAL) FWA of Working Storage Area											6																																																																																			
KP	KL Key Length				MKL Major Key Length				RKP		RKW				PNA (PNAL) Addr. of Partition Name											7																																																																																														
MFN Multi-file Name											IP				DP				KA Key Address											7																																																																																										
MNB Minimum Block Length											RMK Record Mark Character				PC		LA FWA of Label Area											8																																																																																												
LP BCP of D Record Length Field											RB No. of Records in K Block											PAR Parameter List Address											9																																																																																							
CP BCP of T Record Trailer Count Field											RC Record Count											10																																																																																																		
P	S	Reserved		C	N	S	LL		W	LOP		RC Record Count											10																																																																																																	
D	B	Reserved		C	N	S	CL		P	LOP		RC Record Count											10																																																																																																	
F	F	Reserved		C	N	S	CL		P	LOP		RC Record Count											10																																																																																																	
LBL Length of Label Area											MUL				BN Current Block Number											11																																																																																														
MBL Maximum Block Length											NL				O	TRC		ECL Error Code Location											11																																																																																											
IBL Index Block Length											WAC Word Address Char. Pos.				WA Current Word Address											13																																																																																														
Reserved for Installation																						14																																																																																																		
HMB Number of Home Blocks											Reserved				HRL Address of Key Hashing Routine											15																																																																																														
CDT											Reserved				DCT											15																																																																																														
XN Index File Name											Reserved				KR Key Repeat											16																																																																																														
Reserved for Record Manager																						17																																																																																																		

Figure 3-1. File Information Table

NOTES: FIT

The following fields are explained as they apply to CYBER Record Manager under SCOPE 3.4.

FIT TABLE

Word 0

59-18 LFN Logical file name

17-0 Reserved for Record Manager

Word 1

59-36 RL Current record length in number of characters
If RL=1, for multiple index file, position file only

35 PM Processing mode (Record Manager only)
0 = random
1 = sequential

34-32 FO File organization
000 sequential (SQ) default
001 word addressable (WA)
011 indexed sequential (IS)
101 direct access (DA)
110 actual key (AK)

31 NDX Index flag
0 = data file is accessed
1 = index file is accessed

30-28 BT Blocking type
000 default/internal (I)
001 internal (I)
010 character count (C)
011 record count (K)
100 exact records (E)

27 BCK Block checksum flag
0 = no checksums
1 = checksums

26 Unused

NOTES: FIT (CONT'D)

25-22	RT	Record type 0000 control word (W) 0001 fixed length (F) 0010 record mark (R) 0011 zero byte (Z) 0100 decimal character count (D) 0101 trailer count (T) 0111 undefined (U) 1000 System logical records (S)
21	DKI	Duplicate key indicator; indicates duplicate key permission on an IS file 0 no 1 yes
20-18	PD	Processing direction 000 input (INPUT) 001 input (INPUT) 010 output (OUTPUT) 011 input/output (I/O)
17-0	FET	Pointer to FET
Word 2		
59-36	PTL	Partial transfer length, set by GETP, PUTP macros. Number of keys moved to WSA for a GET or GETN on an alternate key index.
35-34	OF	Open flags (positioning of file at OPENM time) 00 rewind (R) 01 rewind (R) 10 no rewind (N) 11 extend (E)
33-32	VF	End of volume flags (positioning of file at volume CLOSEM time) 00 unload (default) 01 rewind (R) 10 no rewind (N) 11 unload (U)

NOTES: FIT (CONT'D)

31-30	CF	Close flags (position at file close) 00 Rewind (R) 01 Rewind (R) 10 No rewind (N) 11 Unload (U)
29-28	LT	Label type 00 ANSI standard (S) 01 Non-standard (NS) 10 Unlabeled (UL) 11 Any (ANY)
27-25	ULP	User label processing 000 None 001 VOL/EOV (V) 010 HDR/EOF (F) 011 VOL/HDR/EOV/EOF (VF) 100 UVL/UHL/UTL (U) 101 VOL/UHL/UVL/EOV/UTL (VU) 110 HDR/UVL/EOF/UHL/UTL (FU) 111 All (VFU)
24-18	FP	File position 000 Mid-record 001 End of label group or beginning of information (EOL/BOI) 002 Beginning of file/volume (BOF/BOV) (only set on SKIPBu in connection with DX) 004 End of volume (EOV) 010 End of section (EOS) 020 End of record (EOR) 040 End of partition (EOP) 100 End of information (EOI)
17-0	LX	Label routine address
Word 3		
59-36	HL	Character length of fixed header of a type T record
	MNR	Minimum character length of type R record
35-18	BFS	CIO buffer size (number of words)
17-0	DX	End of data routine address

NOTES: FIT (CONT'D)

Word 4

59-36	TL	Character length of trailer portion of type T records
35	FNF	Fatal/non-fatal flag (1 = fatal)
34	PEF	Parity error flag (1 = parity error)
33-31		Reserved
30-27	SES	System error subfield
		01 Read parity error level 1
		02 Read parity error level 2
		03 Read parity error level 3
		04 Read parity error level 4
		05 Write parity error level 1
		06 Write parity error level 2
26-18	IRS/ES	Invalid request subfield (CYBER Record Manager/ octal error status error code)
17-0	EX	Error Routine address

Word 5

59-54	VNO	Current volume number of multi-volume sequential file
53-45	ECT	Current trivial error count
44-36	ERL	Trivial error limit
35	FPB	File position bit (for user use)
34	SVO	0 Indicates IS version 2 user 1 Indicates IS version 1 user
33	SPR	Suppress read ahead
		0 Read ahead/write behind (buffered sequential I/O)
		1 No read ahead/no write behind
32	EXD	Extended diagnostic flag
		0 No detailed explanation
		1 More detailed diagnostics are produced on the file ZZZZEF
31	SDS	System message disposition
		0 For SQ and WA files, all error messages are written on the system error file, ZZZZZEF. Fatal errors are written on the job dayfile (NO)
		For IS, DA and AK files all messages and file statistics are written on the system error file

NOTES: FIT (CONT'D)

		1	For SQ and WA files, error messages appear on the job dayfile as well as ZZZZZEF (YES)
			For IS, DA and AK files, all nonfatal messages are written on ZZZZZEF, and file statistics are written on the job dayfile.
30-29	OC		Open/close 00 never opened 01 opened 10 closed
28	FWI		Forced write indicator for IS, AK, and DA file blocks 0 no forced write (NO) 1 forced write (YES)
27	ON		Old/new IS, DA, or AK file (1 = new-creation run) (0 = old)
26	LCR		Label action on PD tape (PD = I/O) 0 create new labels (N) 1 check existing labels (E)
25	KNE		Key not equal (multiple-index file) 0 key match 1 no key match found
24	HB		User header option AK files 0 do not return header 1 return header (default)
23-21	LVL		Level number of an EOS
20-18	REL		File position key relation (IS files) 1 EQ equal 2 LE less than or equal 3 GE greater than or equal 4 NE not equal 5 LT less than 6 GT greater than
17-0	FWB		First word address of user I/O buffer
Word 6			
59-36	MRL		Maximum record length in characters
	FL		Length of an F or Z type record (characters)
35-26			(Reserved for CDC)

NOTES: FIT (CONT'D)

25	CM	Conversion mode 1 convert from EC to IC (YES) 0 no conversion (NO)
24-22	EO	Error options 000 terminate file (T) 001 drop erroneous data (D) 010 accept erroneous data (A) 011 (unused) 100 display data and terminate file (TD) 101 drop and display data (DD) 110 accept and display erroneous data (AD) 111 (unused)
21-0	WSA	First word address of user work storage area (user record area)
Word 7		
59-56	KP	Starting character position of record locator key (IS, DA files) specified in KA
55-47	KL	Length of key in characters (IS, DA files) Length of key in bits (AK files) Length of primary or alternate key in bits prior to open of new multiple index file; after open, length in characters (AK files)
46-38	MKL	Major key length in characters (IS files)
37-34	RKP	Relative key position in RKW (DA files or alternate key IS, DA, or AK files)
33-22	RKW	Relative key word position (DA files or alternate key IS, DA, or AK files)
35-29	IP	Index block padding percent (IS files)
28-22	DP	Data block padding percent (IS, AK files)
21-0	KA	Address of record locator keys (key field address)
59-24	MFN	Multi-file name
23-0	PNO	Multi-file position number
Word 8		
59-36	MNB	Minimum block length in characters
35-28	RMK	Record mark character

NOTES: FIT (CONT'D)

27-22	PC	Padding character for sequential file blocks
	KT	Key type for indexed sequential files (octal) 00 symbolic (S) 01 symbolic (S) 02 integer (I) 03 floating (F) 04 computational-1 06 computational-2 07-77 reserved for CDC use
21-0	LA	First word address of user's label area
Word 9		
59-36	CP	Beginning character position of trailer count field (type T record) (numbered from zero)
	LP	Beginning character position of record length field (type D record) (numbered from zero)
35-22	RB	Number of records per K type block in SQ files; records per block in AK file; average number of records for IS or DA file
21-0	PAR	Reserved
Word 10		
59	PDF	Set by SETFIT, cleared by 'open
58	SBF	1 if suppressed buffer I/O
57-45	Reserved (CDC)	
44	CNF	File is connected to terminal if 1
43	SB	For D-type records, length field has sign overpunch. For T-type records, the trailer count field has sign overpunch
42	C1	For D-type records, length field is binary For T-type records, trailer count field is binary
41-36	LL	Length in characters of record length field (type D record)
41-36	CL	Length in characters of record trailer count field (type T record)

NOTES: FIT (CONT'D)

Word 13

59-36	IBL	Index block length (characters)
35-30	WAC	Word address character position
29-0	WA	Current position word address set by GET, PUT, GETP, and PUTP macros

Word 14 Reserved for users

Word 15

59-30	HMB	Number of home blocks of a DA file
21-0	HRL	Address of key hashing routine for a DA file
51-30	CDT	Collating sequence to display code conversion table address for IS file
21-0	DCT	Address if display code to collating sequence conversion table address for IS file

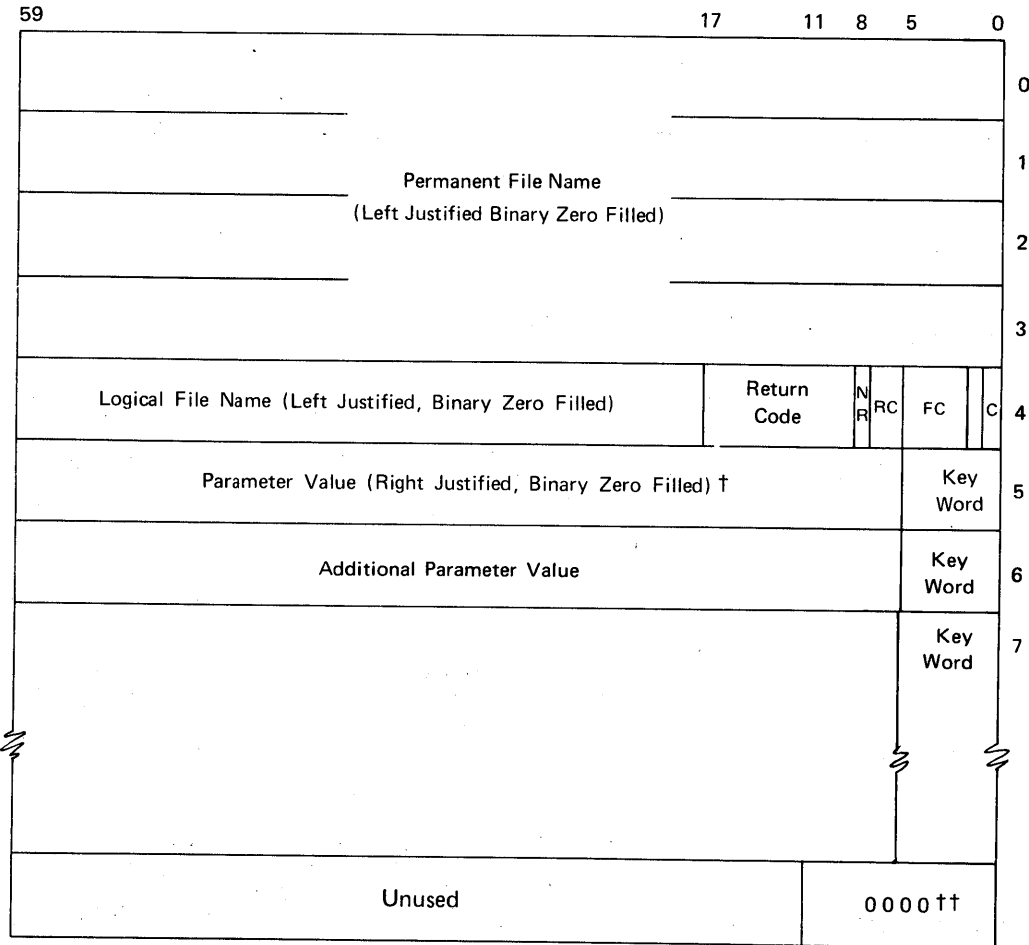
Word 16

59-18	XN	Name of index file associated with multiple-index file
17-15		Unused
14-12		Reserved for KI
11-0	KR	Number of times key value repeats in current record of multiple-index file

Word 17

59-0		Scratch area used by CDC CYBER Record Manager
------	--	---

FILE DEFINITION BLOCK



NR (Bit 8)

1 = NR option specified; no automatic recall

RC (Bits 7-6)

01 = No RC or RT specified
 00 = RC option specified
 10 = RT option specified (implies RC as well)

FC (Bits 5-2) Function Code

0001 = SETP	0111 = ALTER
0010 = ATTACH,GETPF	1000 = PURGE,PURGE(ST=xxx)
0100 = CATALOG,SAVEPF	1010 = RENAME
0110 = EXTEND	1100 = PERM

C (Bit 0) Complete Bit

1 = Function completed

†If VSN (keyword 41) is specified, a 6-character volume serial number is contained in bits 59-24, right-justified, display code zero filled.

††The system checks only bits 0 through 11 of this word.

NOTES: FILE DEFINITION BLOCK

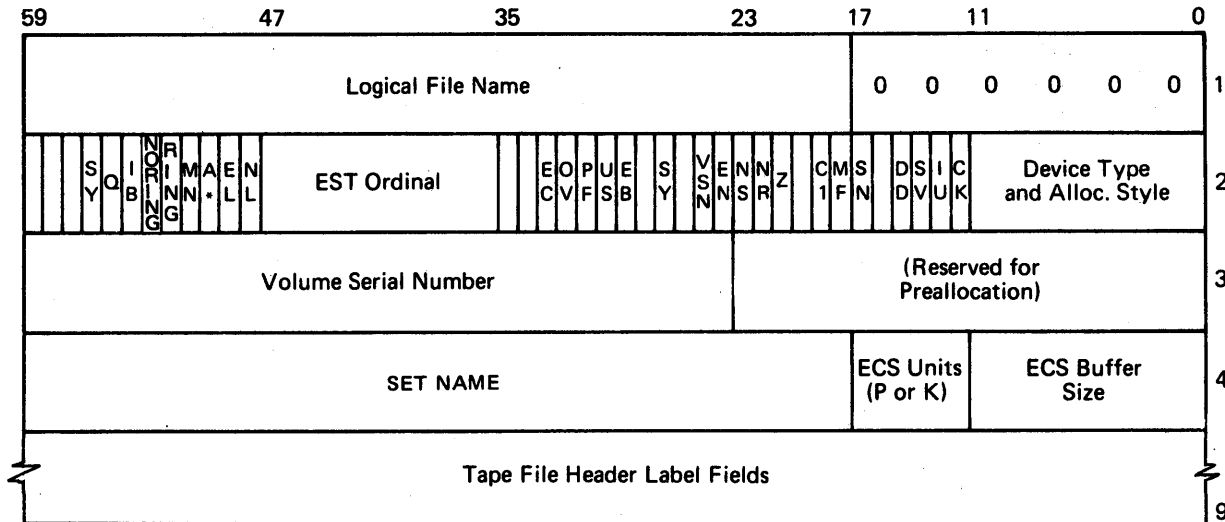
Return code (Bits 17-9 of word 4) and message written to the job dayfile

000	Function successful	035	File already in system
001	ID error	040	Illegal setname specified
002	lfn already in use	041	Device set not mounted at control point
003	Unknown lfn	042	RBT chain too large for PFC
004	No room for extra cycle (limit is five)	070	PFM stopped by system
005	PFC Full	071	Incorrect permission
006	No lfn or pfn	072	FDB address error
010	Latest index not written for a random file	073	I/O error on PFD/PFC read/write
011	File not on PF device	074	ST illegal with user set (MMF-PURGE)
012	File not cataloged, SN=xxxxxxx		
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		
020	Function attempted on nonlocal file		
022	File never assigned to a device		
023	Cycle incomplete or dumped		
024	PF already attached		
025	File unavailable		
027	Illegal lfn		
030	File dumped		
031	Illegal function code		
033	ALTER needs exclusive access		

Key Word (Bits 5-0 of any parameter word) and parameter value

01	PP –Privacy Procedure PP name	20	} PW –Passwords submitted
02	RP –Retention Period (days) (binary)	21	
03	CY –Cycle Number (binary)	22	
04	TK –Turnkey Password Definition (display)	23	
05	CN –Control Password Definition (display)	24	
06	MD –Modify Password Definition (display)	25	FO –File Organization (display)
07	EX –Extend Password Definition (display)	26	PS –Position
10	RD –Read Password Definition (display)	30	PF –Permanent File Name
11	MR –Multi-Read Parameter (binary)	31	LC –Lowest Cycle (binary)
12	SD –Reserved	32	ST –Staging ID (display)
13	XR –Control, Modify, Extend Password Definition (display)	33	RW –Multi-Access Rewrite (binary)
14	ID –Owner-Identification (display)	40	SN –Setname (display left-justified)
15	RN –Reserved	41	Reserved for VSN parameter
16	AC –Account Parameter (display)	43	RB –RB conflict on Permanent File
17	EC –ECS Buffering (display)		

REQ FUNCTION PARAMETER LIST



Bits	59-57	Reserved
56		1 = assign file to system device in system set
55	(Q)	1 = assign file to queue device (in queue set if no (SN) given)
54	(IB)	1 = inhibit system noise records if IP.NBRK=0
53	(NORING)	1 = write enable ring prohibited in tape
52	(RING)	1 = write enable ring required in tape
51	(MN)	1 = accept either MT or NT assignment
50	(A*)	1 = assign any RMS device (overrides device type specification)
49	(EL)	1 = extended label fields in parameter words 5-9
48	(NL)	1 = normal label fields in parameter words 5-8
35		Reserved
34		Reserved
33	(EC)	1 = ECS buffering requested; parameter word 4
32	(OV)	1 = allow overflow to a different device
31	(PF)	1 = assign file to a PF device
30	(US)	1 = ASCII conversion mode on 9-track tape
29	(EB)	1 = EBCDIC conversion mode on 9-track tape
28		1 = assign automatically†
27	(SY)	1 = print card image from RA+70
26		1 = assign 2 devices
25	(VSN)	1 = VSN declared in parameter word 3
24	(EN)	1 = tape has existing labels
23	(NS)	1 = tape has non-standard labels
22	(NR)	1 = disable standard tape parity recovery procedure
21	(Z)	1 = SCOPE 3.3 labeled tape
20		1 = return error code without dayfile message or operator intervention
19	(C1)	1 = console checkpoint request
18	(MF)	1 = multi-user tape request
17	(SN)	1 = set name request
16		Reserved
15	(DD)	1 = default density for labels and data
14	(SV)	1 = save tape
13	(IU)	1 = inhibit physical unload
12	(CK)	1 = checkpoint tape

† An * preceding a parameter will cause automatic assignment.

NORMAL LABEL FIELDS

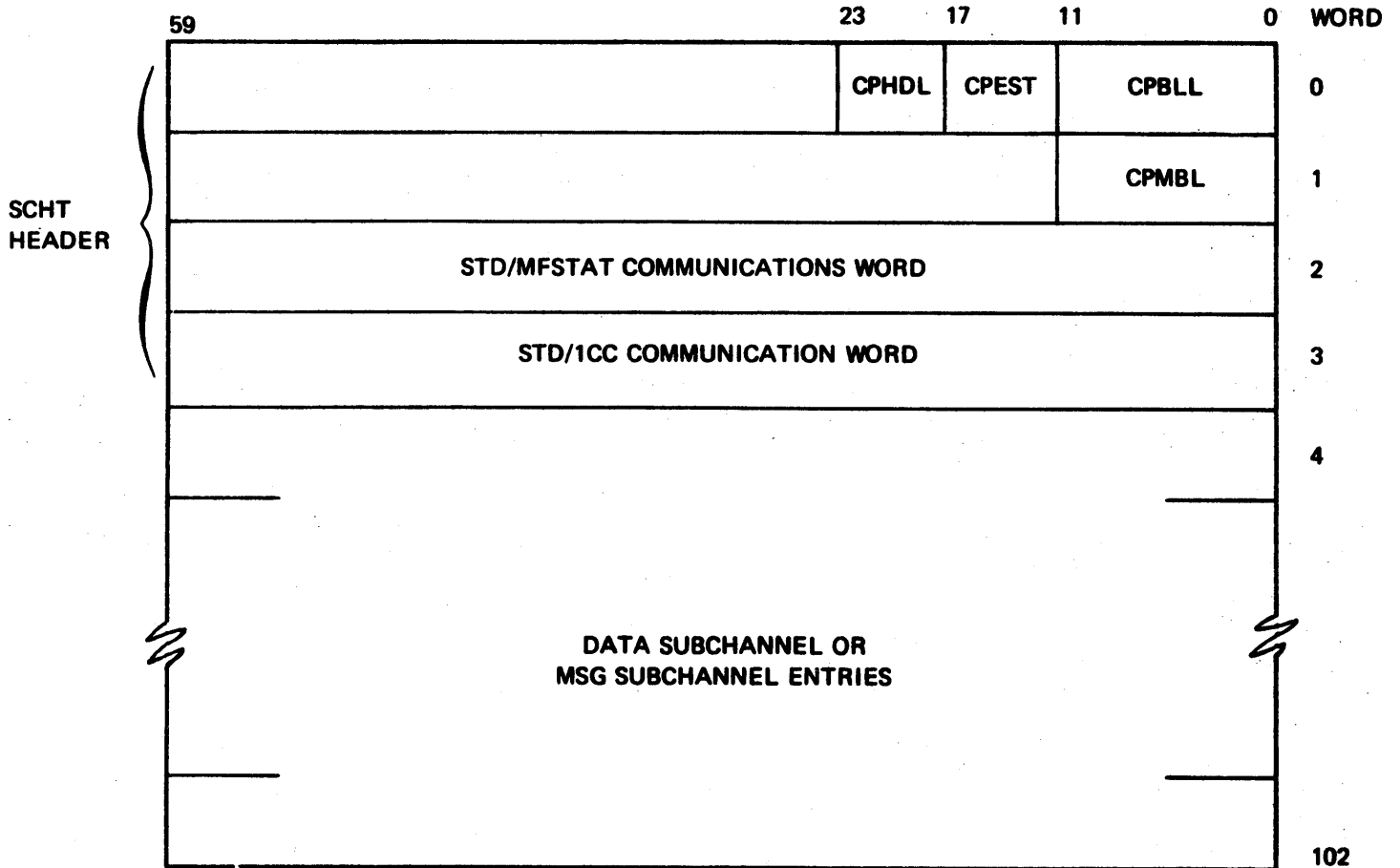
59	47	29	23	17	0	
File Label Name						5
				Position Number		6
Edition Number		Retention Cycle		Creation Date (YYDDD)		7
Multi-File Set Name				Reel Number		8

EXTENDED LABEL FIELDS

59	53	35	29	17	5	0
H D R 1		File Label Name				5
						6
File Label Name	Multi-File Set Name			Reel Number		7
Reel Number	Position Number		Generation Number		Edition Number	8
Edition Number	Creation Date (ΔYYDDD)					9

SUBCHANNEL TABLE (SCHT)

The SCHT is used by the CDC CYBER Enhanced Station in a multi-mainframe configuration. The SCHT consists of two parts, the header and word 1 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 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 ICC.



NOTES: SUBCHANNEL TABLE

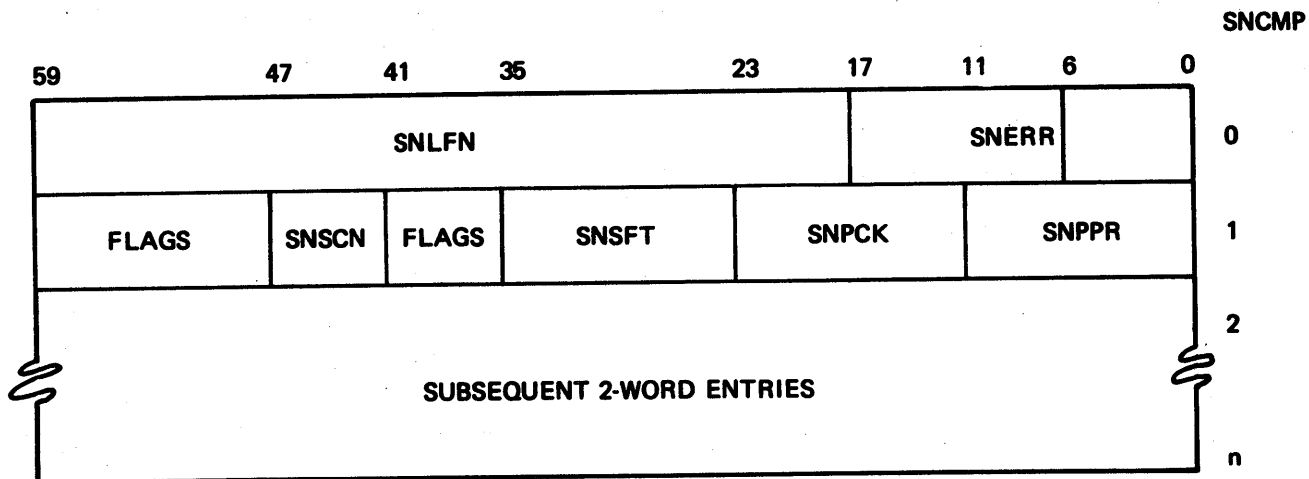
Field	Field Name	Description
Word 0		
23-18	CPHDL	SCHT header size minus 4
17-12	CPEST	EST ordinal
11-0	CPBLL	Current standard buffer size
Word 1		
11-0	CPMBL	Maximum standard buffer size
Word 2		
59-54	CPRCD	Request code (station only)
52-48	CPRSC	Subchannel number
Word 3		
59-54	PPRCD	Request code
58	PPBUS	Request word busy
53-48	PPRSC	Subchannel busy
47-36	PPRCP	Control point number
17-0	PPRFT	FET address
Word 4-102 Data and message subchannel entries		
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

NOTES: SUBCHANNEL TABLE (CONT'D)

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

SPOT NAME TABLE (SNT)

The SNT is used by the CDC CYBER Enhanced Station in a multi-mainframe 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 ICC acts as the communicator for the task. The opening, closing and backspacing of the staged file is coordinated through the SNT. Routines that access this table are 1NS, 1CC, MFSTAT, and DSD.



NOTES: SPOT NAME TABLE

Field	Field Name	Description
Word 0		
59-18	SNLFN	Name of spun off task; is mnnncc. 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
6-17	SNERR	IEJ error return code
00	SNCMP	Complete indicator set to 0 when task is spun off, and set to 1 when IEJ goes through end-of-job processing
Word 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 17-00 contain backspace word count
53	SNBUS	ICC busy bit; if on, open, close, or backspace operation is in progress (set and cleared by ICC only)
52	SNDOP	Open procedure for the task dayfile is complete
51	SNFIN	Dayfile transfer completed
49	SNRFL	Buffer space obtained for dayfile transmission
49	SNSPR	A request made to linked mainframe to stage a spooled file
48	SNRWI	File direction 0 = transmit 1 = receive
47-42	SNSCN	Subchannel number to be used for file I/O

NOTES: SPOT NAME TABLE (CONT'D)

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 FILED (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

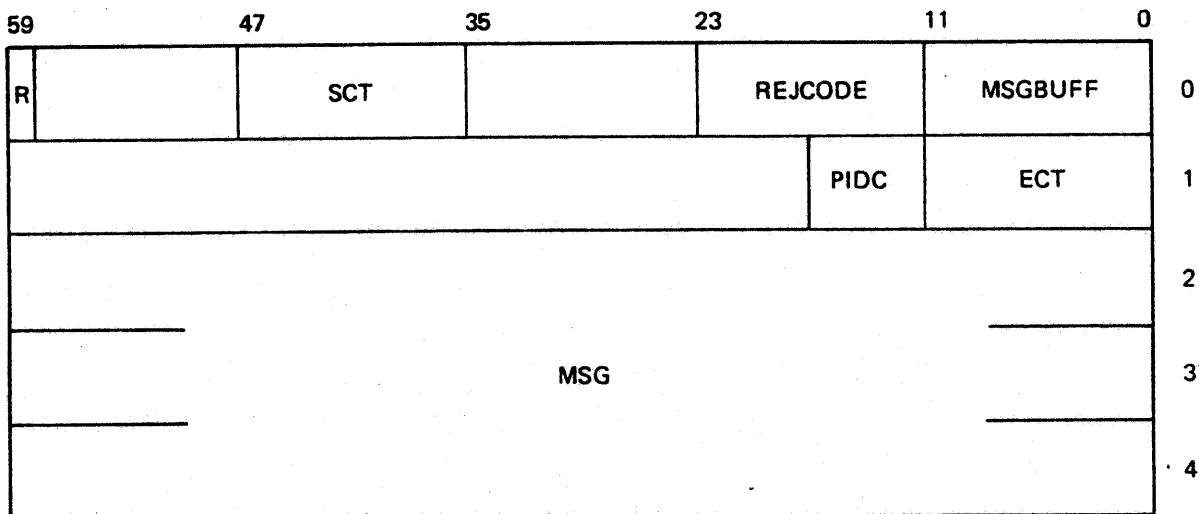
Word 2

ENTRY

More 2-word entries following

DSD/INTERCOM COMMUNICATION THROUGH STATION (MSG TABLE)

MSG is used by the CDC CYBER Enhanced Station in a multi-mainframe configuration. This table, residing within the station field length, serves as a communication area for DSD, INTERCOM, and the station.



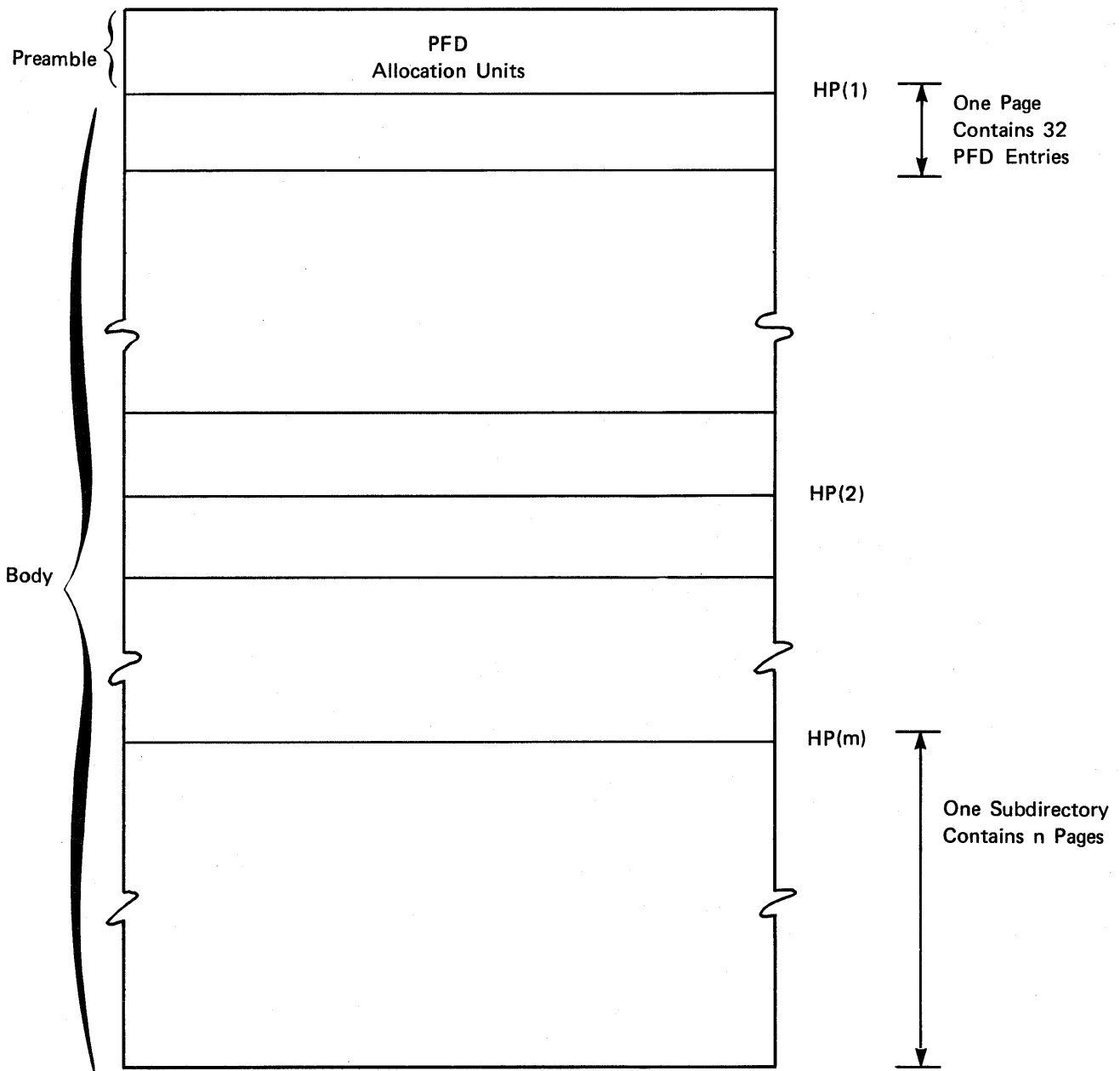
NOTES: MSG TABLE

Field	Field Name	Description
Word 0		
59	R	Command or display rejected by linked mainframe
47-36	SCT	Station sequence count
23-12	REJCODE	Type of error
11-00	MSGBUFF	Address reply to the message (relative to RA of station). Reply is available when SCT equals ECT
Word 1		
17-12	PIDC	Identifying character of PID, to which message is sent
11-00	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)
Word 2		
	MSG	Message request from external processor to be sent to designated mainframe. MSG starts in word 2 and can be up to 3 words long

SECTION 3

DISK TABLES AND FILE FORMATS

PERMANENT FILE DIRECTORY OVERVIEW



HP_i ID Hash Point

$m =$ Number of ID Hash Points

$n =$ Number of Pages between ID Hash Points; must be an Integral Power of two

$m*n =$ Number of Pages in the PFD Body

PFD ENTRY

	59	47	35	29	23	17	11	5	0											
W.PDHDR/W.PDSD W.PDCPFN W.PDEF	7	7	7	7	P	F	D	4	SUB DIR. NO.	Chars. in PFN	S	F							0	
W.PDID	OWNER-ID										148									1
W.PDN1																			2	
W.PDN2																			3	
W.PDN3	PERMANENT																		4	
W.PDN4	FILE NAME																		5	
W.PDCY W.PDIC W.PDAC W.PDPE W.PDPFC	LEFT JUSTIFIED, ZERO FILLED																		6	
	CYCLE NUMBER	Mainframe ordinal (if I=1 only)				D	I	A	R	H	PE	PFC POINTER				6				
	CYCLE NUMBER	Mainframe ordinal (if I=1 only)										PFC POINTER				7				
	CYCLE NUMBER	Mainframe ordinal (if I=1 only)										PFC POINTER				8				
	CYCLE NUMBER	Mainframe ordinal (if I=1 only)										PFC POINTER				9				
	CYCLE NUMBER	Mainframe ordinal (if I=1 only)										PFC POINTER				10				
W.PDPW	TURNKEY																		11	
	CONTROL																		12	
	PASSWORDS									MODIFY									13	
	EXTEND																		14	
	READ																		15	

D=1 Cycle dumped
 I=1 Cycle incomplete
 S=1 End of subdirectory flag, S.PDESD
 F=1 Entry in use flag, S.PDEF
 ARH=1 Cycle archived
 PE=1 Parity error

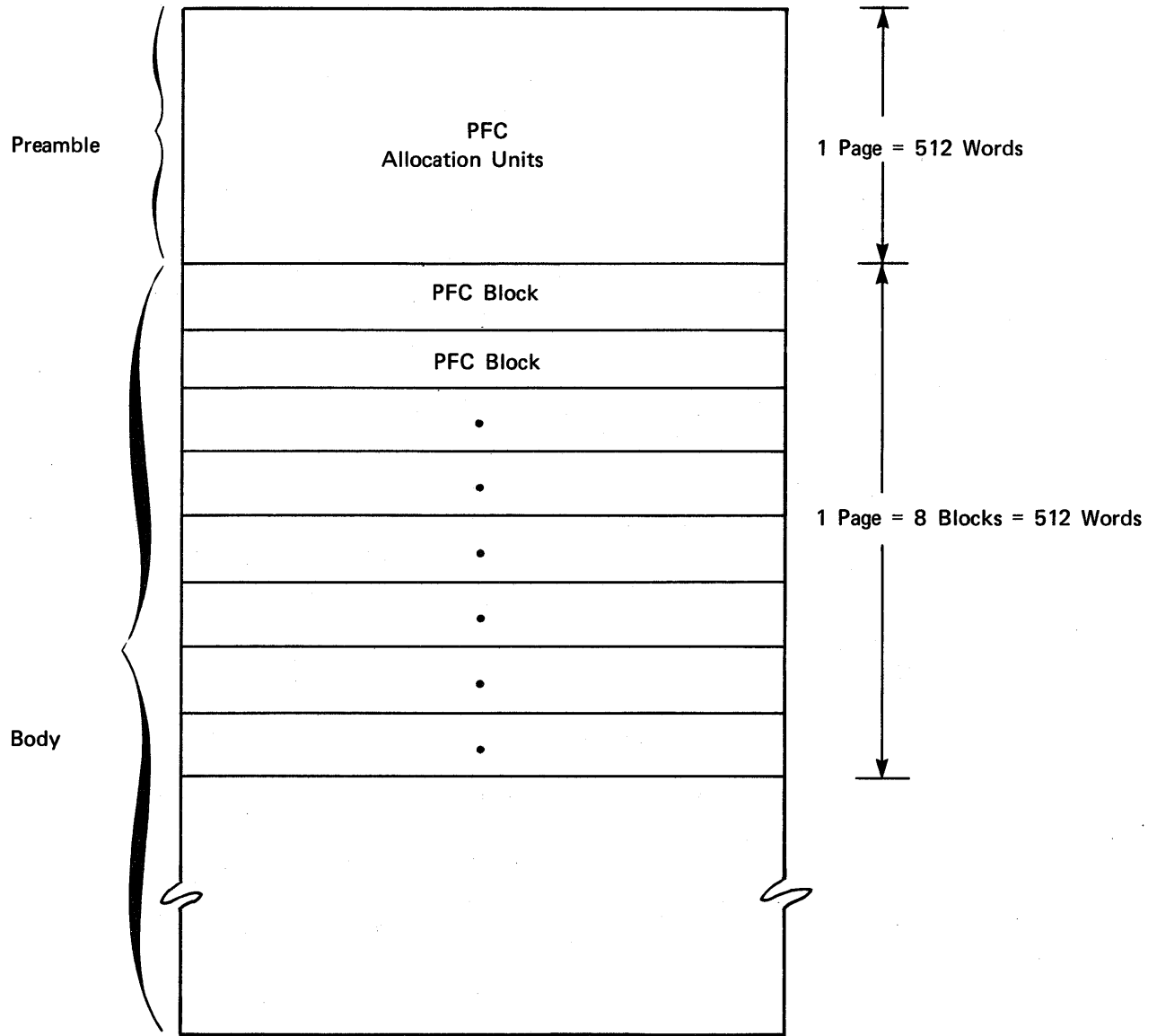
NOTES: PFD

Symbol	Word	Position Within Word	Contents
W.PDHDR	0	59-48 47-36 35-24	7's in binary PF in variable code D4 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.PDEF	0	3	0 = entry is free 1 = entry is in use
W.PDID	1	59-6 5-0	Owner ID, right-justified, blank filled, display code 14B
W.PDN1	2	59-0	Permanent file name, left-justified, trailing binary zeros with no nested bytes of zeros
W.PDN2	3	59-0	
W.PDN3	4	59-0	
W.PDN4	5	59-0	
W.PDCY	6-10 6-10	59-48 27	Cycle number (binary) 0 = cycle is not dumped 1 = cycle is dumped
W.PDFMO	6-10	47-36	Mainframe ordinal (incomplete cycle only)
W.PDIC	6-10	26	0 = cycle is complete 1 = cycle is incomplete
W.PDAC	6-10	25	0 = cycle is not archived 1 = cycle is archived
W.PDPE	6-10	24	0 = cycle has no parity errors 1 = cycle has a parity error
W.PDPFC	6-10	17-0	PFC pointer as a 100B word offset (binary)
W.PDPW	11	59-6	Password parameter for TURNKEY permission, right-justified, binary zero filled, display code Password parameter for CONTROL permission, right-justified, binary zero filled, display code
	12	59-6	

NOTES: PFD (CONT'D)

Symbol	Word	Position Within Word	Contents
W.PDPW (continued)	13	59-6	Password parameter for MODIFY permission, right-justified, binary zero filled, display code.
	14	59-6	Password parameter for EXTEND permission, right-justified, binary zero filled, display code
	15	59-6	Password parameter for READ permission, right-justified, binary zero filled, display code

PERMANENT FILE CATALOG OVERVIEW†



Each PFC entry occupies one or more consecutive PFC blocks.

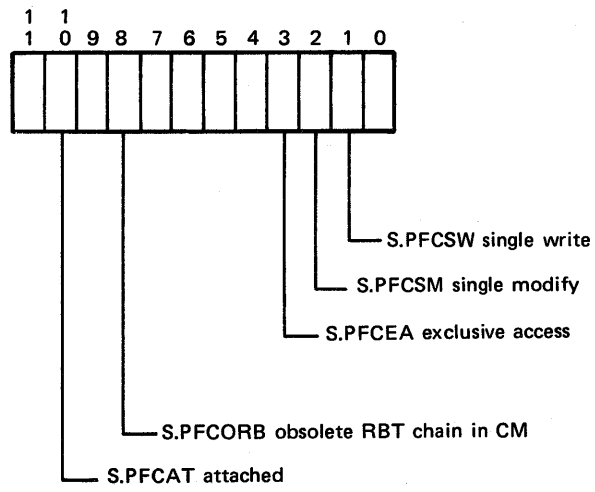
†PFC was formerly called the RBTC (Record Block Table Catalog).

PERMANENT FILE CATALOG (PFC) ENTRY FOR PERMANENT FILES

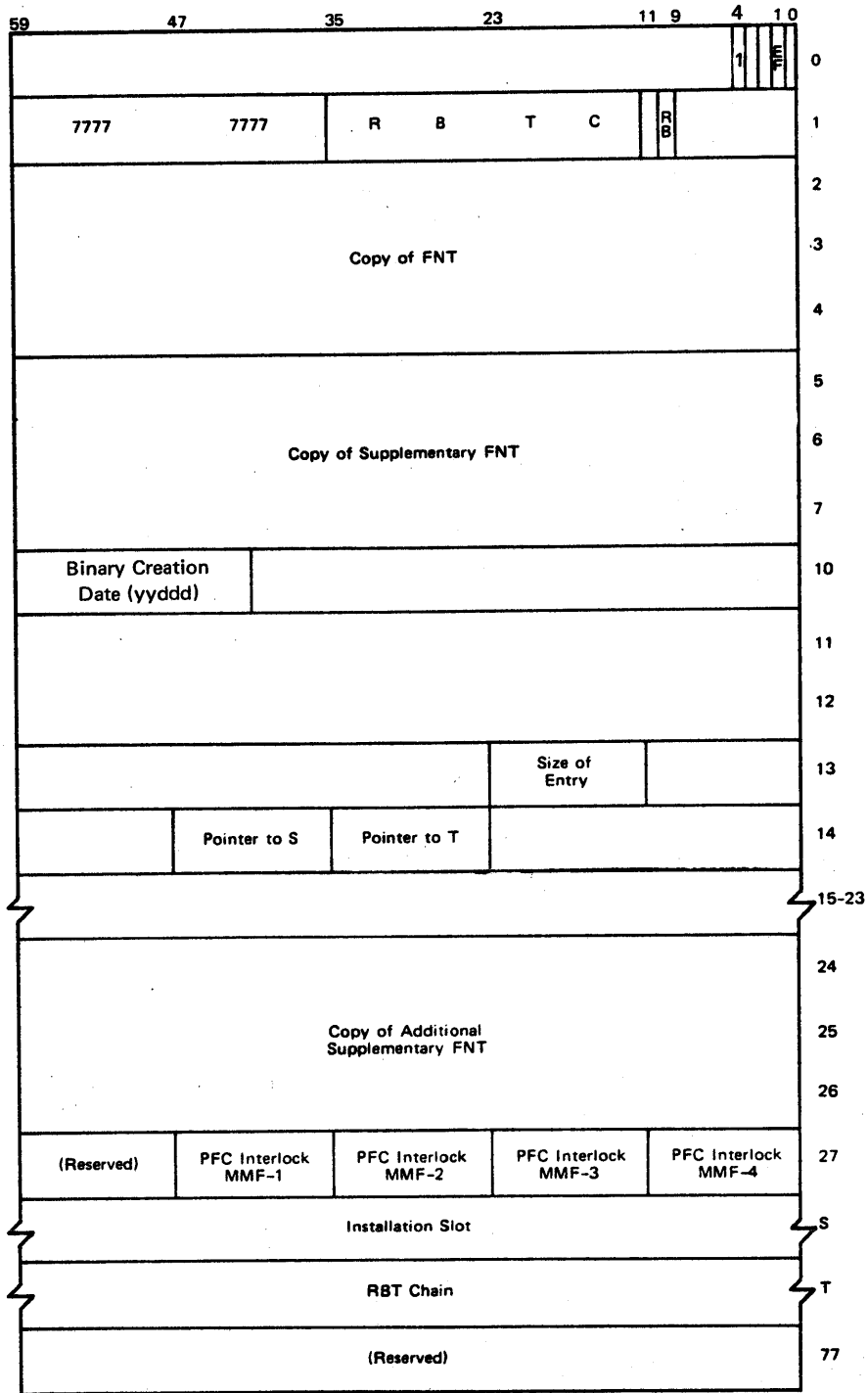
		59	53	47	41	35	29	23	17	11	8	6	4	2	0		
W.PCFC	W.PCEF													O	E	F	0
W.PCARC	W.PCID	7777			7777			R	B	T	C	RT	POS	ARC	NV	1	
W.PCFO		Owner ID (Right Justified, Blank Filled)												1	4	2	
W.PCRA																3	
W.PCPS																4	
W.PC9T																5	
W.PCHDR															6		
	W.PCN1															7	
	W.PCN2	Permanent File Name (Left Justified, Zero Filled)														4	
	W.PCN3															5	
	W.PCN4															6	
W.PCCY	W.PCDE	Cycle				PFD Entry Offset				PFD Pointer (in PRU Offset)				7			
W.PCDE																10	
W.PCDE	W.PCDE	Binary Creation Date (YYDDD)				Retention Period (Binary)								10			
W.PCDE	W.PCDE	Binary Date of Last Attach (YYDDD)				Binary Time of Last Attach (HHMMSS)								11			
W.PCDE	W.PCDE	Binary Date of Last Alteration (YYDDD)				Binary Time of Last Alteration (HHMMSS)								12			
W.PCNA	W.PCNE	Number of Attaches		Number of Extends		Number of Modifies		Size of Entry						13			
W.PCNM		Pointer to S		Pointer to T		Subdirectory Number						14					
W.PCESZ	W.PCS															15	
	W.PCT															16	
	W.PCSD															17	
	W.PCACT	Account Parameter														15	
		Dump Tape VSN 1				SETP Pointer								16			
		Dump Tape VSN 2				CKP Pointer								17			
		(Reserved)														20	
W.PCDLD	W.PCTLD	Binary Date of Last Dump (YYDDD)				Binary Time of Last Dump (HHMMSS)								21			
W.PCTLD																22	
	W.PCPW	Turnkey Password												00		22	
		Control Password												00		23	
		Modify Password												(Right Justified, Binary Zero Filled)		00	24
		Extend Password												00		25	
		Read Password												00		26	
		Reserved	PFC Interlock MMF-1		PFC Interlock MMF-2		PFC Interlock MMF-3		PFC Interlock MMF-4						27		
		Installation Slot														S	
		RBT Chain														T	
		(Reserved)														77	

EF=1	Entry free	RAN=1	Random
NV=1	New version	FO=1	File organization IS/DA/AK
9T=1	Nine-track	ARC=1	Archived
POS=1	Positioned	RB=1	RB conflicts

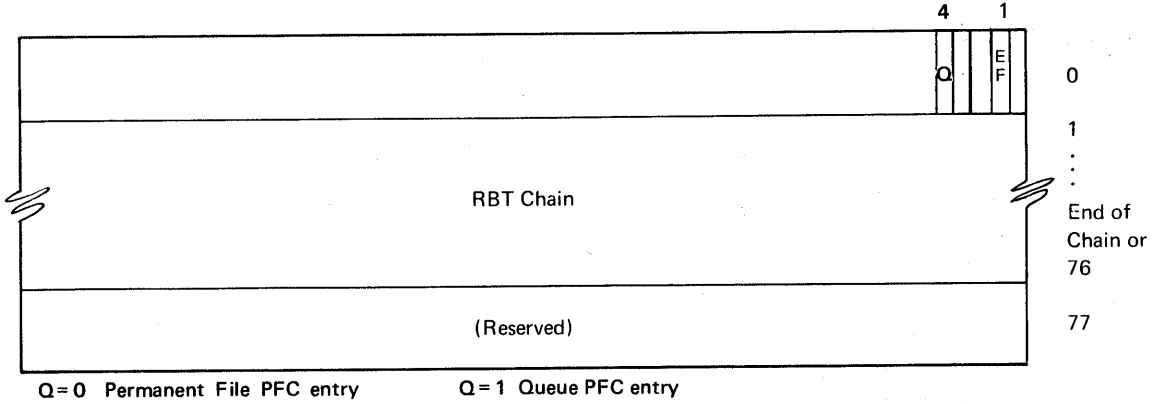
Detail: PFC I/L Byte



PFC ENTRY FOR I/O QUEUES.



PFC ENTRY OVERFLOW



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 Volume Serial Number					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

59	47	38	35	23	11	0	
C.RBTWPL	777	7		End of Volume PRU + 1			
C.RBTDRB DAM Ordinal	C.RBTVSN Volume Serial Number					0 0 0 0	

FIRST RBT WORD PAIR: TYPE 0 (PRE-SCOPE 3.4.4 CREATED PUBLIC DEVICE FILES)

59	47	S.RBTRBR	35	29	23	11	0
C.RBTWPL Next Word Pair	C.RBTRBR C.RBTFB RBR Ordinal	3	C.RBTAL Alloc. Type	C.RBTPRU Last PRU + 1	C.RBTBIT Flags		
RB3	RB4		RB5	RB6	RB7		

SUBSEQUENT WORD PAIRS:

59	47	35	23	11	0
C.RBTWPL Next Word Pair	C.RBTRBR C.RBTFB RBR Ordinal	0	RB0	RB1	RB2
RB3	RB4		RB5	RB6	RB7

For queue files the end of the input file RBT chain is flagged with 7777B in the next word pointer field (C.RBTWPL) if a predayfile follows. In all cases a zero in C.RBTWPL indicates end of chain.

NOTES: PFC

Symbol	Word	Meaning
W.PCEF	0	Bit 1 1 = free 0 = locked
		Bit 4 0 = permanent file entry 1 = queue file entry
W.PCFC	1	Bit 2 1 = new 0 = dumped
W.PCARC	1	Bit 4 0 = cycle is not archived 1 = cycle is archived
W.PCPO	1	Bit 5 0 = file is not a SAAM file 1 = file is a SAAM file
W.PCRA	1	Bit 6 0 = file is not random 1 = file is random
W.PCPS	1	Bit 7 0 = file is not positioned 1 = file is positioned
W.PC9T	1	Bit 8 0 = 7-track tape 1 = 9-track tape
W.PCRB	1	Bit 9 0 = no RB conflicts 1 = RB conflicts exist on this file
W.PCHDR	1	RBTC in display code
W.PCID	2	14 octal in bits 5-0. Owner ID in bits 59-6
W.PCN1 — W.PCN4	3-6	Permanent file name in display code, left justified, zero filled
W.PCCY	7	Cycle number in binary
W.PCPDE	7	PFD entry number
W.PCPFD	7	PFD pointer as a sector offset (binary)
W.PCCD	10	Bits 59-42 contain the creation date in binary in the format YYDDD
W.PCRT	10	Bits 23-12 contain the retention period in binary
W.PCDLA	11	Bits 59-42 contain the date of the last attach in binary in the format YYDDD
W.PCTLA	11	Bits 17-0 contain the time of the last attach in binary in the format HHMMSS
W.PCDLME	12	Bits 59-42 contain the date of the last alter in binary in the format YYDDD

NOTES: PFC (CONT'D)

Symbol	Word	Meaning
W.PCTLME	12	Bits 17-0 contain the time of the last alter in binary in the format HHMMSS
W.PCNA	13	Bits 59-48 contain the number of attaches in binary
W.PCNE	13	Bits 47-36 contain the number of extends in binary
W.PCNM	13	Bits 35-24 contain the number of modifies in binary
W.PCESZ	13	Bits 23-12 contain the size on the entry in binary number of words
W.PCS	14	Number of words from word 13 to the user area
W.PCT	14	Number of words between word 13 and start of RBT chain
W.PCSD	14	Bits 23-12 contain the subdirectory number in binary
W.PCACT	15	Account parameter in display code, left-justified, blank fill
	27	PFC interlock for MMF configuration; each byte contains the following flags: <ul style="list-style-type: none"> bit 1 single write (S.PFCSW) bit 2 single modify (S.PFCSM) bit 3 exclusive access (S.PFCEA) bit 8 obsolete chain in CM (S.PFCORB) bit 10 file attached (S.PFCAT)

PERMANENT FILE SET DEVICE LABEL

	59	53	35	29	23	17	11	0		
W.LBLD W.LBTYPE W.LBDATE	DEV 1		02	YYDDD					0	
W.LBEX				YYDDD					1	
W.LBVSN	VSN								2	
W.LBSN W.LBMEM	Set Name				* Maximum Number [†] of Members				3	
W.LBPF					* PFD Alloc. Unit Index				4	
	Reserved for CDC								5	
W.LBPFC					* PFC Alloc. Unit Index				6	
	Reserved for CDC								7	
W.LBDAM					* DAM Alloc. Unit Index				8	
W.LBSMT					* SMT Alloc. Unit Index				9	
W.LBFLW			Number of Extra Flaw Tables		* Flaw Table Alloc. Unit Index				10	
W.LBSFT	Reserved for CDC								11	
W.LBDSR	(Reserved for Deadstart Recovery)				* Deadstart Recovery RB Number				12	
W.LBSD					* Subdirectory Table Alloc. Unit Index				13	
W.LBPCM	U					* PFC Alloc. Map Alloc. Unit Index				14
W.LBFLT					* Physical Flaw Table Alloc. Unit Index				15	
W.LBNPFC W.LBNPFP W.PBNSD		* Number of PFC Words/100B	* Number of PFD Pages per Subdirectory	* Number of Subdirectories				16		
W.LBCK							Checksum		17	

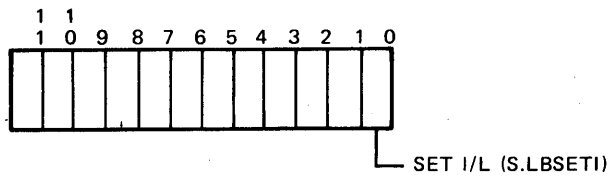
* Field will contain non-zero value only on master device
 † The length of the SMT is twice the value contained in this field
 U-Bit 59 indicates subdirectory counts are invalid

PERMANENT FILE SET DEVICE LABEL (CONT'D)

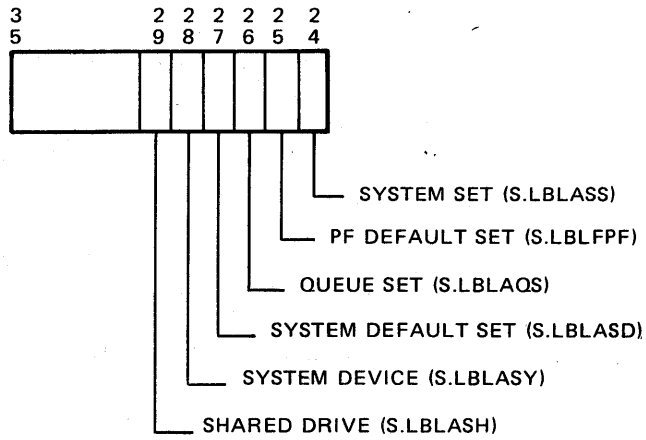
	59	47	35	23	11	0	
W.LBAUS						Alloc. unit size in words/100B	18
W.LBDUP	UNIVERSAL PERMISSION PASSWORD (RESERVED)						19
W.LBDPI	PUBLIC PERMISSION PASSWORD (RESERVED)						20
W.LBDFR						†Default file retention period (Reserved)	21
W.LBDIAG	CE AREA PREALLOCATION 0 = Preallocated All 7's = Not Preallocated						22
W.LBIL	C.LBNMF number of MNTed MFs	C.LBSETI MF 1	MF 2	MF 3	MF 4		23 set I/L
W.LBLI1	C.LBEO EST ord	C.LBMO MF ord	C.LBATT local attr.	No. of DUMPFs	C.LBLID host MF PIO MF 1		24
W.LBLI2							25
W.LBLI3							26
W.LBLI4							27
W.LBDSM					C.LBDSM DSMNT flag		28
W.LBGLA					C.LBGLA Global Attri.		29
W.LBCHS	date 1st DUMPF	time 1st DUMPF	dump mode				30 DUMPF synchr.

† Field contains non-zero value only on master device.

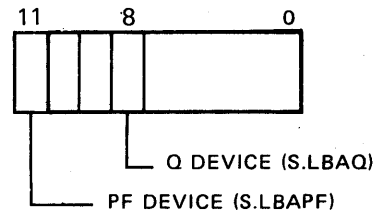
Detail: Set I/L Byte in W.LBIL



Detail: Local attributes in W.LBLI1 through W.LBLI4



Global Attributes in W.LBGLA



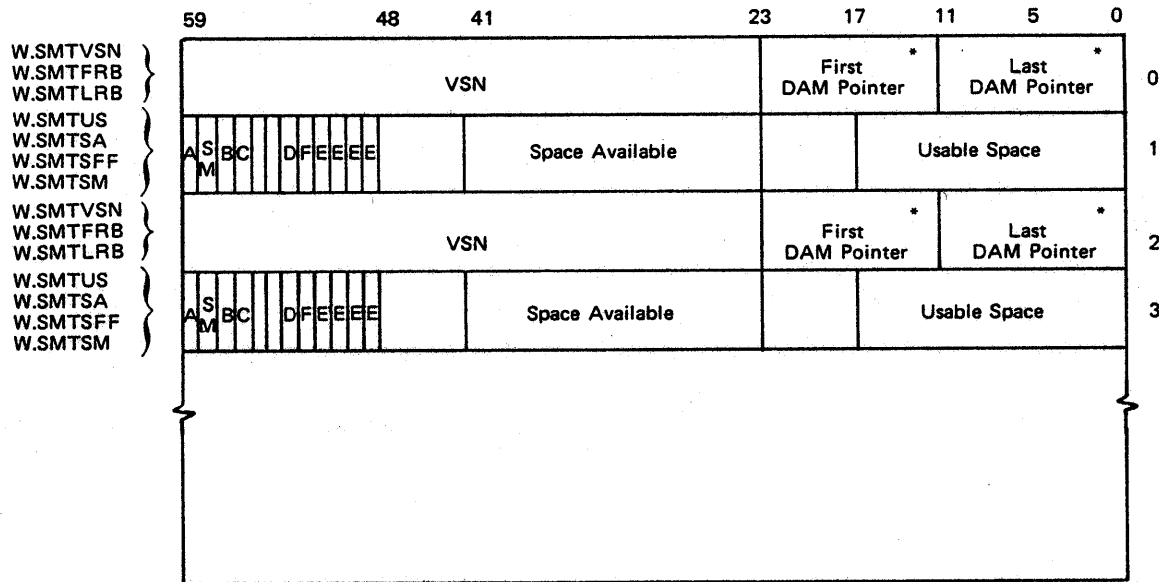
NOTES: PERMANENT FILE SET DEVICE LABEL

Symbol	Words	Position Within Word	Contents
W.LBLD	0	59-36	DEV1 in display code
W.LBTYPE	0	35-30	02 = 6000/7000
W.LBDATE	0	29-0	The date in the form YYDDD (display code)
W.LBEX	1	29-0	The expiration date in the form YYDDD (display code)
W.LBVSN	2	59-24	VSN (display code) right-justified, display zero filled
W.LBSN	3	59-18	Setname (display code) left justified, binary zero filled
W.LBMEM	3	17-0	Maximum number of members (binary)
W.LBPFD	4	17-0	PFD allocation unit index (binary)
	5	59-0	CDC reserved
W.LBPFC	6	17-0	PFC allocation unit index (binary)
	7	59-0	CDC reserved
W.LBDAM	8	17-0	DAM allocation unit index (binary)
W.LBSMT	9	17-0	Set member table allocation unit index (binary)
W.LBFLW	10	17-0	Flaw table allocation unit index (binary)
	11	59-0	CDC reserved
W.LBDSR	12	17-0	Deadstart recovery RB number
W.LBSD	13	59	1 = pack created under SCOPE 3.4.x
W.LBPCM	14	17-0	PFC allocation map allocation, unit (binary)
W.LBFLT	15	17-0	Physical Flaw Table allocation unit index (binary)

NOTES: PERMANENT FILE SET DEVICE LABEL (CONT'D)

Symbol	Words	Position Within Word	Contents
W.LBNPFC	16	53-36	Number of PFC words/100B
W.LBNPFP	16	35-18	Number of PFD pages per subdirectory (binary)
W.LBNSD	16	17-0	Number of subdirectories (binary)
W.LBCK	17	11-0	Checksum of this label PRU

SET MEMBER TABLE (MASS STORAGE)



* For 844 device First DAM pointer = Last DAM pointer.

- 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 MF ordinal 1-4 (bit 51 = 1, 50 = 2, 49 = 3, 48 = 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.

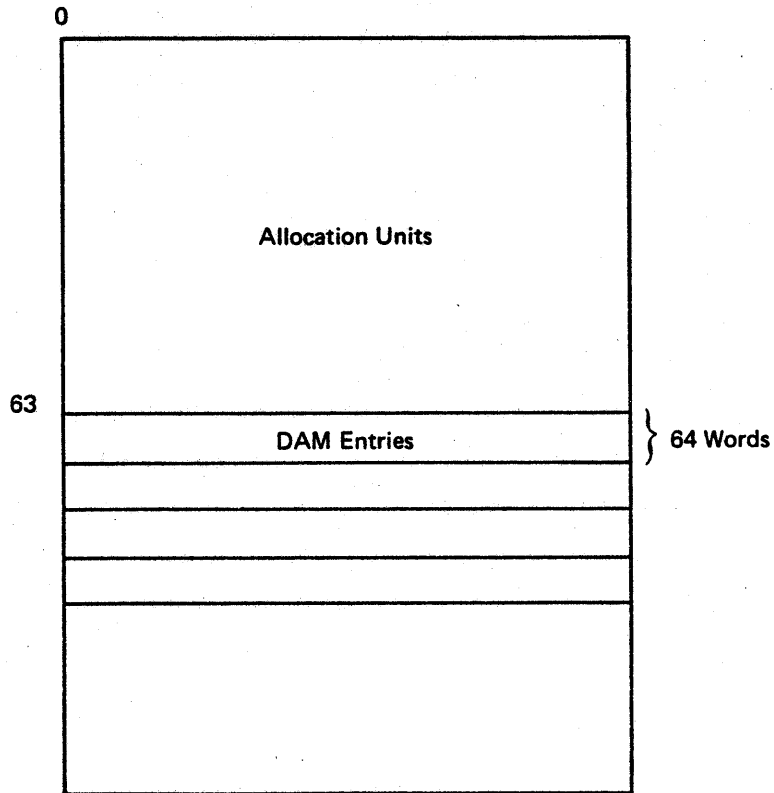
Zero entries can be intermixed with VSN entries for member devices.

NOTES: SET MEMBER TABLE

Symbol	Words	Position Within Word	Contents
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)
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	SM = 0 Device is not mounted SM = 1 Device is mounted
	Odd	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
W.SMTSA	Odd	41-24	Total allocatable space available on this device (binary)=number of allocation units currently not in use on the device
W.SMTUS	Odd	17-0	Total number of allocation units minus number of flaws and minus number of system table allocation units

A two word entry will exist for each member in this set.

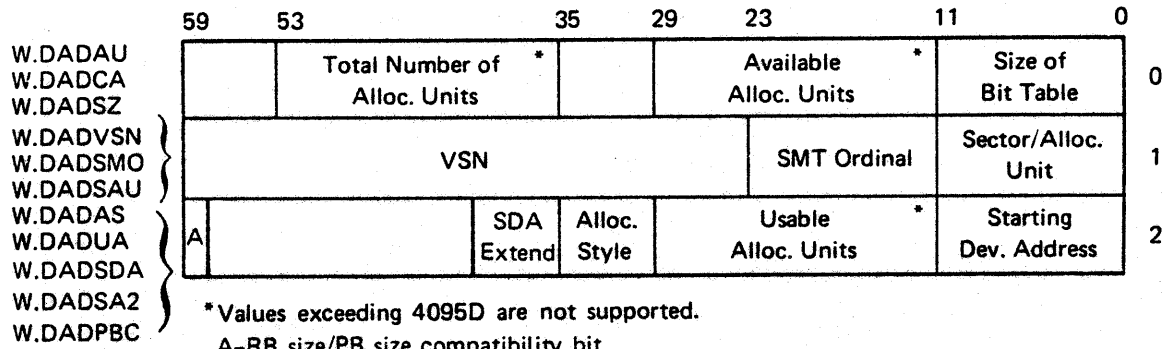
OVERVIEW OF DEVICE ALLOCATION MAP



Every DAM starts on a sector boundary. If the entry needs to be greater than 64 words, it will always be overflowed to the next PRU or PRUs.

An 844 DAM may contain a maximum of 50_{10} entries; i.e., the entire DAM requires one allocation unit.

DAM HEADER



NOTES: DAM

Symbol	Words	Position Within Word	Contents
W.DADAU	0	53-36	Total number of allocatable units
W.DADCA	0	29-12	Currently available allocatable units
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 allocatable unit
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 112D for 844s
W.DADAS	2	35-30	Allocation style
W.DADUA	2	29-12	Usable allocatable units = total allocatable units - flaws - system allocatable units
W.DADSDA	2	11-0	Starting Dev. address
W.DADRCA	53	7-0	Preventive Maintenance (844 s)
W.DADRCB	54	59-0	Reserved cylinders

LOGICAL FLAW TABLE

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

PHYSICAL FLAW TABLE FORMAT: 7638/844

	59	29	11	0	
W.PHFNF				Number of Flaws	0
W.PHFAR W.PHFAL	Flaw Address		Flaw Address		1
	Flaw Address		Flaw Address		2
	⋮				⋮
	Flaw Address		Flaw Address		777 ₈

PHYSICAL FLAW TABLE FORMAT: 6603/6633/821/841/854

Note: The Physical Flaw Table may contain a maximum of 1000_g words.

Note: Each device contains its own Logical Flaw Table and Physical Flaw Table.

	11	0	
	F ₁		0
	F ₂		
	F ₃		
	F ₄		
	⋮		
	F _n		
	0		End of List
	0		777 _g

Each F_(i) is a 24-bit physical address corresponding to a particular flaw.

PHYSICAL FLAW TABLE ENTRIES

	59		41	35	29			11	5	0
7638 Address	Track Number			Initial Sector	No. of Sectors	Track Number			Initial Sector	No. of Sectors

	59	56		47	41	35	30	26		17	11	5	0
844 Address		Cylinder Number	Track Number	Initial Sector	No. of Sectors		Cylinder Number	Track Number	Initial Sector	No. of Sectors			

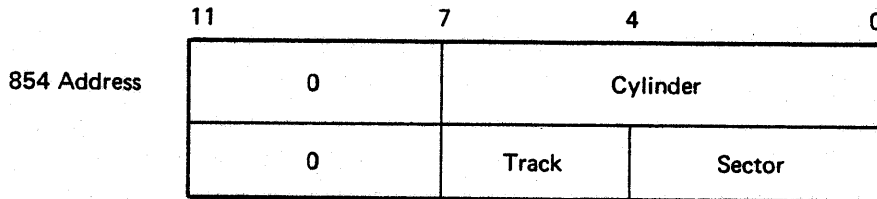
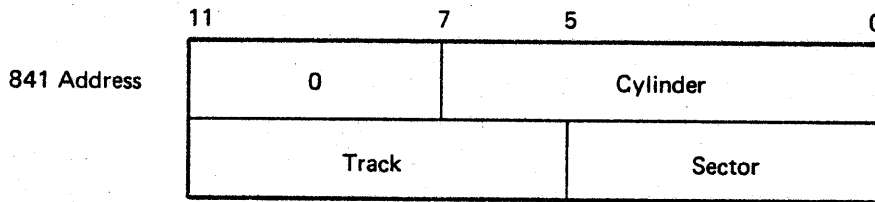
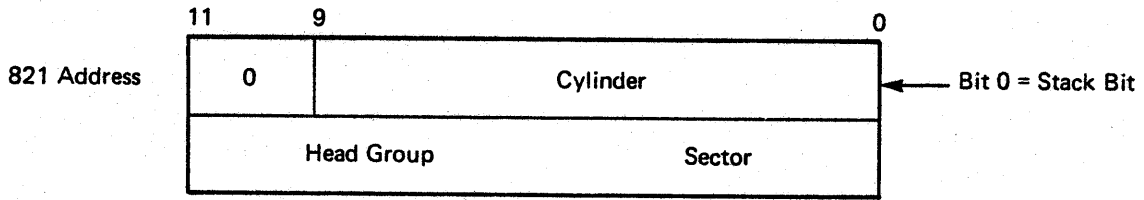
	11	10		7	5		0
6638 Address	0	Position			Head Group		
	0			Sector			

	11		9	8		6			2	1	0
6603 Address	0	H3	Position				H2	H1			
	0			Sector							

PHYSICAL FLAW TABLE ENTRIES (CONT'D)



↑
Lower 5 Bits of Sector



SUBDIRECTORY TABLE

	Interlock Bit	Interlock Bit	Interlock Bit	Interlock Bit	Interlock Bit
Word	59	47	35	23	11 0
0	# of Entries Hashed to This SD Ordinal	# of Entries Hashed to This SD Ordinal	# of Entries Hashed to This SD Ordinal	# of Entries Hashed to This SD Ordinal	# of Entries Hashed to This SD Ordinal
1					
.					
.					
.					
n					

Subdirectory Description

Symbol	Words	Position Within Word	Contents
	0-n	59, 47, 35, 23, 11	Interlock bit; always zero on RMS copy of the table. Reserved for 7000 use only
	0-n	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)

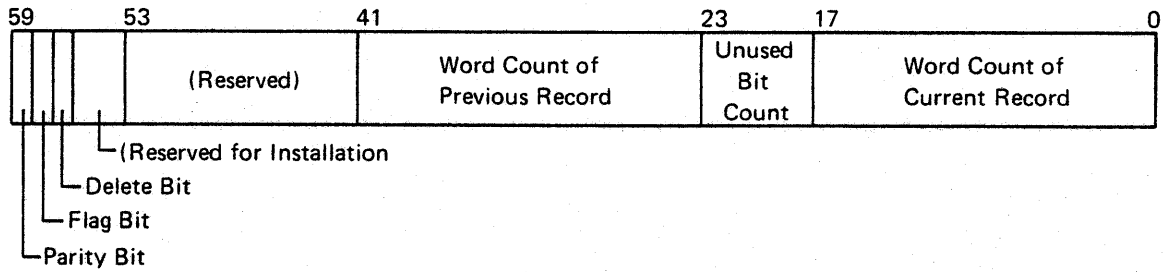
	59	47	41	23	17	0
W.PAMNW						Number of Words in PAM Body = n-1
W.PAMPEI W.PAMCEI				Purge EOI Pointer		Current EOI Pointer
	7 7 7 7	PAM Body: One Interlock Bit per PFC Entry. I/L Bit=1 Indicates Entry Is in Use.				
	7 7 7 7					
	.					
	.					
	7 7 7 7					

NOTES: PAM

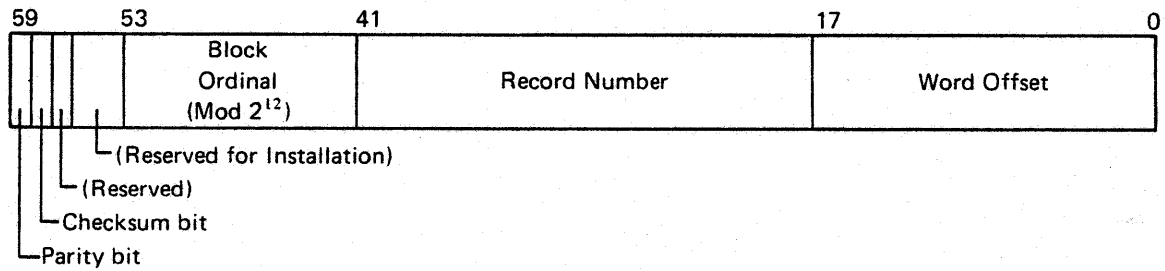
Symbol	Words	Position Within Word	Contents
W.PAMNW	0	17-0	Number of words in PAM body
W.PAMPEI	1	41-24	Pointer to empty 100B word block created by purge as a PFC sector offset
W.PAMCEI	1	17-0	PFC sector offset to first 100B word block not in use after the last 100B word block in use

The PAM body represents one interlock bit per PFC entry. A PAM bit set implies that an entry is in use. Only bits 47-0 of the body are used.

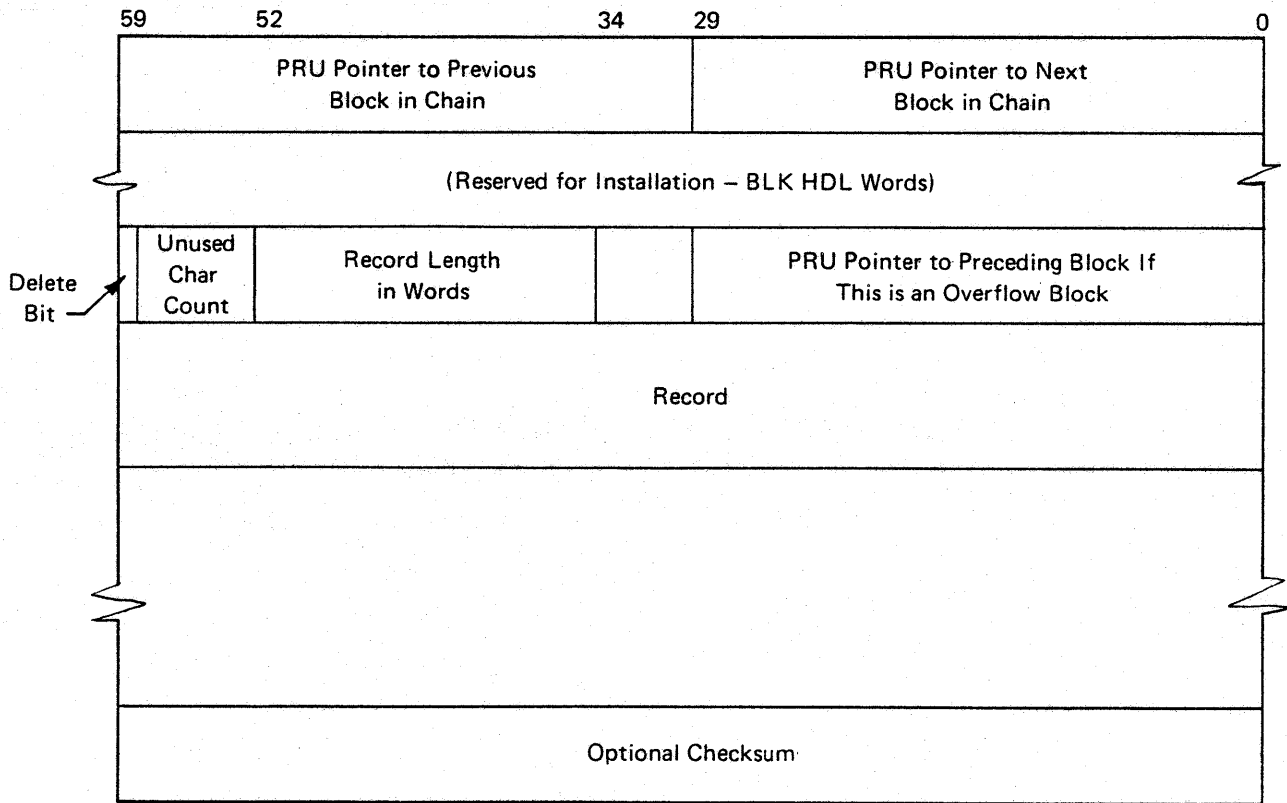
W RECORD HEADER



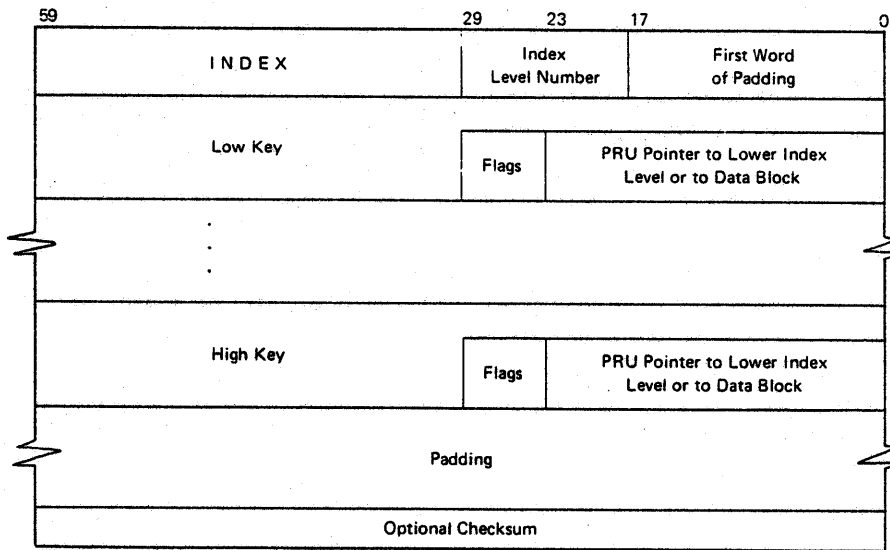
I BLOCK HEADER



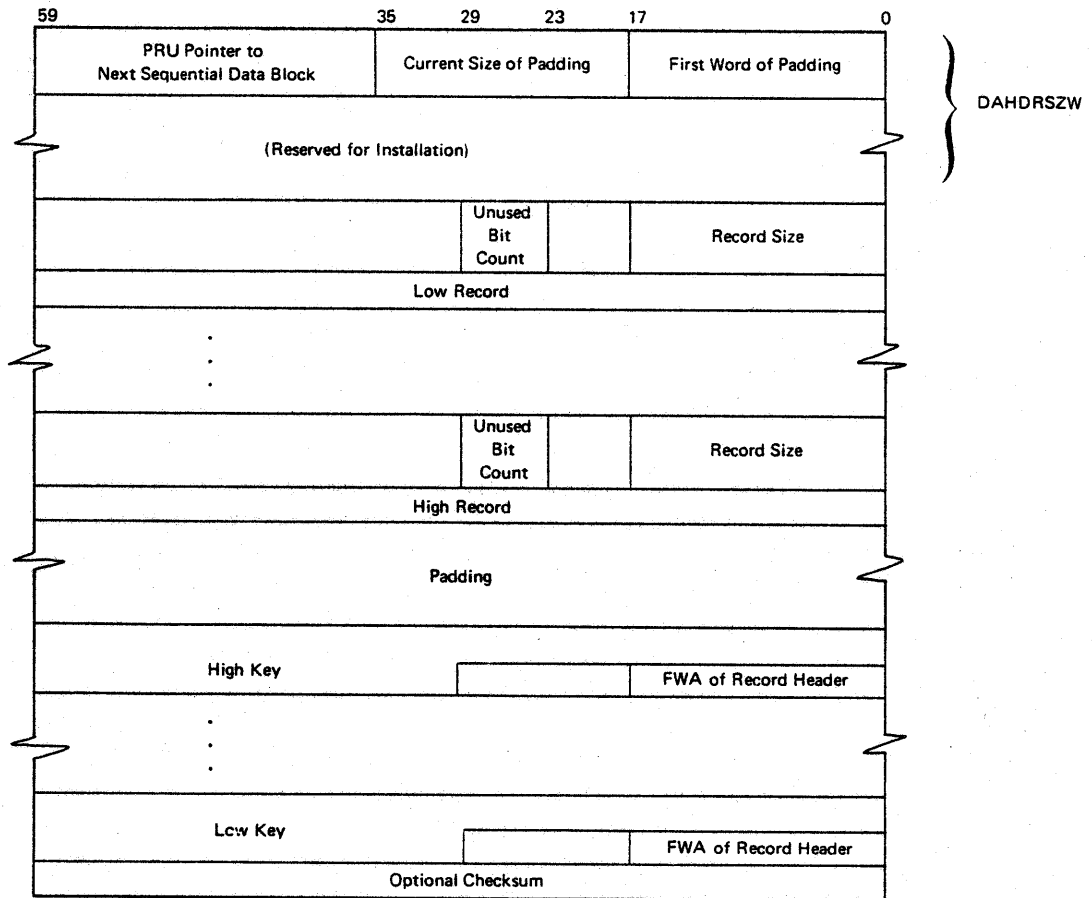
BLOCK FORMAT (DA FILE)



INDEX BLOCK FORMAT (IS FILE)



DATA BLOCK FORMAT (IS FILE)



FSTT (INDEXED SEQUENTIAL FILE)

	59	44	35	23	17	0					
CHEKSUM	FSTT Checksum						0				
HEADWORD	S	A	A	M	-	S	I	S	V	2	1
MIPWORD	(Reserved for MIP)						2				
AVALDABK						PRU Number of Next Empty Data Block	3				
AVALINBK						PRU Number of Next Empty Index Block	4				
CHKSUMB	← Checksum File						5				
CODITABL	Collating Sequence to Display Code Conversion Table						6				
CODITAB2							7				
CODITAB3							10				
CODITAB4							11				
CODITAB5							12				
CODITAB6							13				
CODITAB7							14				
CODITAB8							15				
CURFILPP	PTREE Index		Key Displacement in Data Block		Address of Buffer Catalog		16				
DABKSZP						Data Block Size in PRUs	17				
DABKSZW						Data Block Size in Words	20				
DAPADSZW						Data Block Padding Size in Words	21				
DELETSTO	Total DELETES						22				
DICOTABL	Display Code to Collating Sequence Conversion Table						23				
DICOTAB2							24				
DICOTAB3							25				
DICOTAB4							26				
DICOTAB5							27				
DICOTAB6							30				
DICOTAB7							31				
DICOTAB8							32				
DUPKEYB	← Duplicate Keys Allowed						33				
FEMDABK						Deleted Data Block Chain Pointer	34				
FEMINBK						Deleted Index Block Chain Pointer	35				
FILDATE	File Creation Date						36				
FILNAME	Logical File Name						37				
FIRDABK						First Data Block Pointer	40				
INBKSZP						Index Block Size in PRUs	41				
INBKSZW						Index Block Size in Words	42				

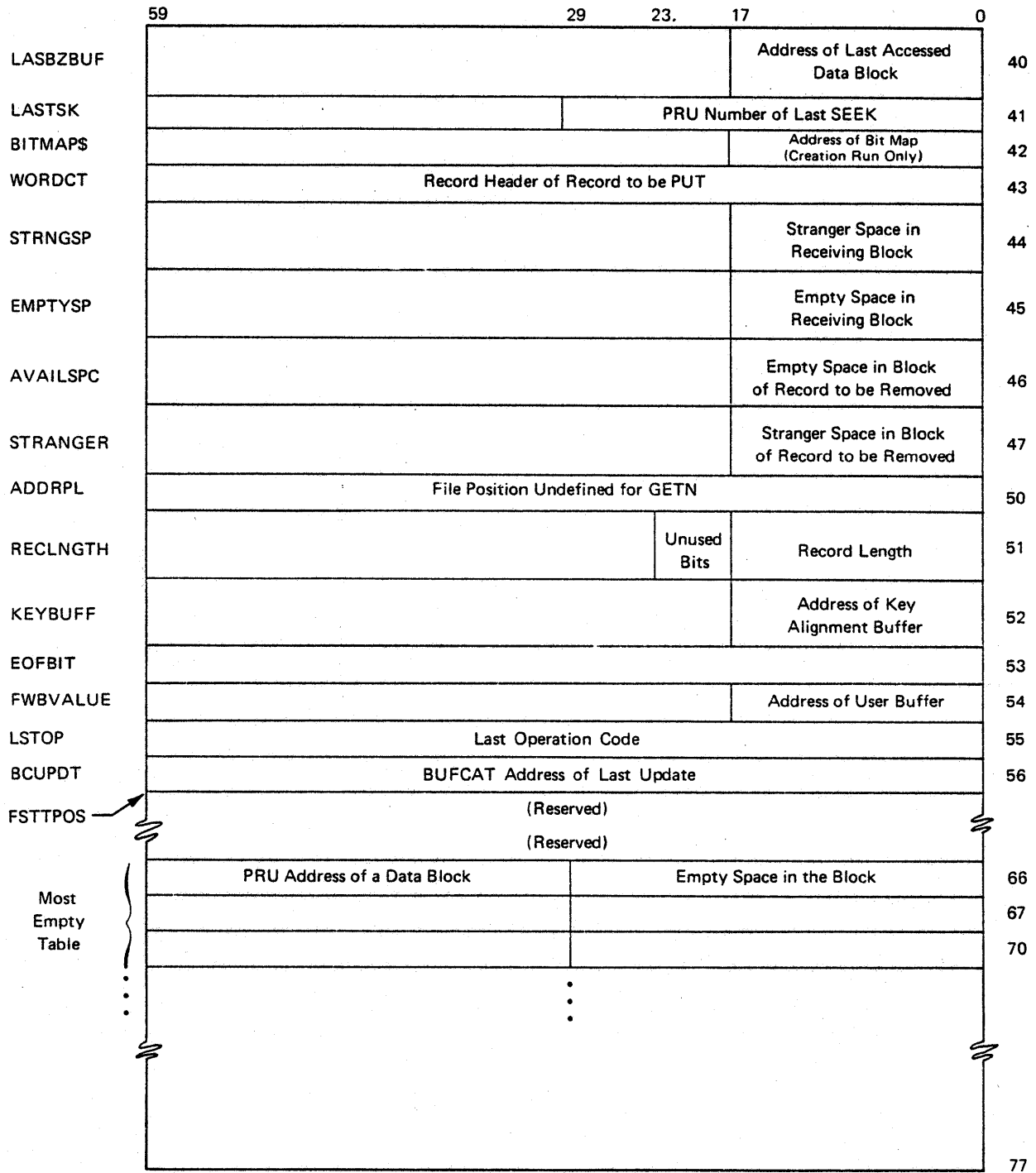
FSTT (INDEXED SEQUENTIAL FILE) (CONT'D)

	59	29	23	11	8	2	0	
INPADSZW	Index Block Padding Size in Words							43
KEYBIAS				COMP-1 Floating Bias				44
KEYENSZW				Key Entry Size (words)				45
KEYPACKB	← Key Ends in Character Position 0-4 of Last Word							46
KEYSZC				Key Size in Characters				47
KEYTYPE				Key Type →				50
LASDABK	Final Data Block Pointer							51
MAXRECSZ	Maximum Record Size in Chars							52
MINRECSZ	Minimum Record Size in Chars							53
NRACSTO	Total GETs							54
NRDABKTO	Total Number of Data Blocks							55
NRDAREC	Total Number of Data Records							56
NREMDABK	Number of Deleted Data Blocks							57
NREMINBK	Number of Deleted Index Blocks							60
NRINBKTO	Total Number of Index Blocks							61
NRINSTO	Total PUTs							62
NRLVLSF				Number Index Block Level				63
NRWUDISK	Number of Unused Words on Disk							64
NXTUNASP	Next Unassigned PRU Number							65
PRINBK	Primary Index Block PRU Number							66
REPLACTO	Total REPLACES							67
SEQINSS	Limit of Sequential INSERTs			Number of Sequential INSERTs				70
TERMDATE	Date of Last CLOSEM							71
TERMTIME	Time of Last CLOSEM							72
UEPRINBK	Number of Unused Key Entries in Primary Index Block							73

FSTT (DIRECT ACCESS FILE)

	59	53	35	29	23	20	17	0			
CKSWORD	FSTT Checksum								0		
HEADER	b	F	S	T	T	b	S	D	A	b	1
MIPWORD	(Reserved for SIP)								2		
CSFLAG	← Block Checksum Option (BCK at OPENM NEW)								3		
STATADD	Total Number of PUTs								4		
STATDLT	Total Number of DELETes								5		
STATRPL	Total Number of REPLACES								6		
STATRET	Total Number of GETs and GETNs								7		
STATOVF	Total Number of Overflow Blocks								10		
STATHOM					Number of Home Blocks in Use				11		
STATHR					Total Number of Records in Home Blocks				12		
STATOR					Total Number of Records in Overflow Blocks				13		
FILESZ					Total Number of Home Blocks				14		
KEYS			Relative Key Position (RKP)		Relative Key Word (RKW)		Key Length (KL)		15		
MINMAXS			Minimum Record Size (MNR)		Maximum Record Size (MRL)				16		
NXTUNASP					Next Unassigned PRU Number				17		
NXTUPDSK					Next Usable PRU on disk				20		
DATABKSZ					Data Block Size (words)				21		
PRUBLK					Number of PRUs per Block				22		
OVFOP	Overflow Manager								23		
ENDFIXD →	Number of PUTs - Current OPENM								24		
RUNADD	Number of DELETes - Current OPENM								25		
RUNDLT	Number of GETs and GETNs - Current OPENM								26		
RUNRET	Number of REPLACES - Current OPENM								27		
RUNRPL					Number of Overflow Blocks Created - Current OPENM				30		
RUNOVF					Number of Home Blocks in Use First Time - Current OPENM				31		
RUNHOM					Number of Records in Home Blocks - Current OPENM				32		
RECHOM					Number of Records Placed in Overflow Blocks - Current OPENM				33		
RECOVF	Total Number of CIO Calls								34		
NRCIOCLS					Current PRU for Use by GETN				35		
CURPRU							Relative Data Block Word for Use by GETN		36		
RELDWB							Address of Buffer Catalog		37		
BUFCATS							↑				
	← Length of Buffer Catalog										

FSTT (DIRECT ACCESS FILE) (CONT'D)



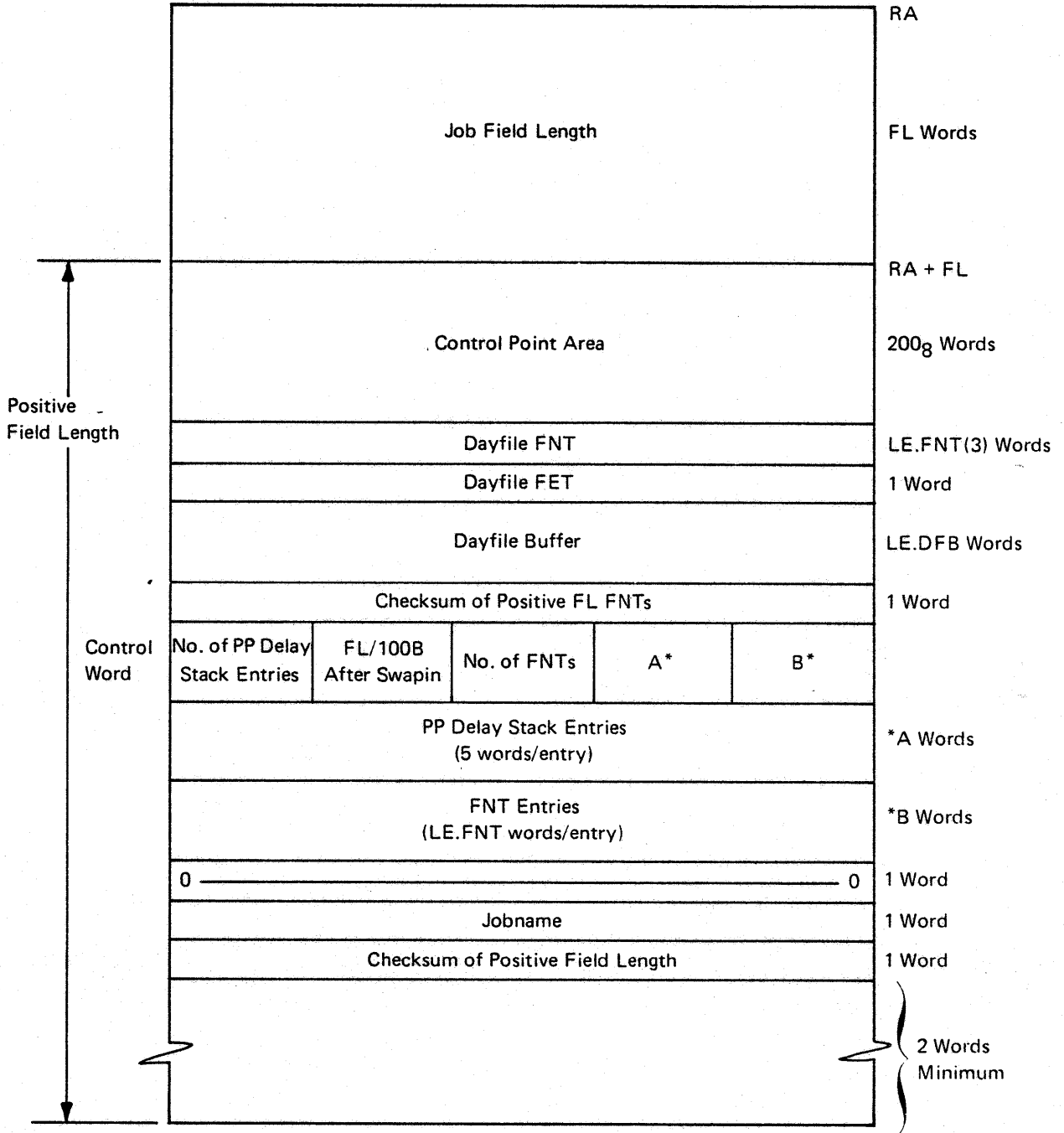
FSTT (ACTUAL KEY FILES)

	59	35	23	17	0	
CKSWORD	FSTT Checksum Word					0
HEADER	S A A M - S A K					1
MIPWORD	MIPWORD to be Used by SIP					2
CSFLAG	←	Bit Set in Data Block to be Checksummed				3
KEYL	Key Length in Bits					4
SLOTL	Length of Slot - Number Field in Bits					5
NXTBLK	Next Unassigned Block Number					6
ENDBLK	Relative Pointer to First Header					7
MINMAX\$				Maximum Record Size	Minimum Record Size	10
NXTUPDSK					Next Used PRU No. on Disk	11
PRUBLK	No. of PRU's Per Block					12
KEYM	Key Mask (Right Justified)					13
SLOTM	Slot-Number Field Mask					14
NOREC	Number of Records in the File					15
OVFREC	Number of Overflow Records in File					16
BLKFCTR	Blocking Factor					17
DBLKPD	Data Block Padding (In Words)					20
CURBCK	Current Block Number (SAKOVFM)					21
RANDOM	Current Random Number (Used by SAKOVFM)					22
EMTABLE	Empty Space Table					23
						24
						25
						26
						27
						30
						31
						32
						33
						34
						35
						36
						37
						40

FSTT (ACTUAL KEY FILES) (CONT'D)

	59	35	17	0	
METABLE	Most Empty Table				41
					42
					43
					44
					45
					46
					47
					50
					51
ENDFIXD	Symbol for End of Permanent Data				52
RUNADD	Number of Puts This Open				53
RUNDLT	Number of Deletes This Open				54
RUNRET	Number of Accesses This Open				55
RUNRPL	Number of Replaces This Open				56
BUFCAT\$		Length of BUFCAT	Address of BUFCAT		57
RECLNGTH	Record Length in SAK Format				60
FWBVALUE		FWB (FIT)			61
CURPOS	Current File Position				62
HDADD1		Pointer to Record Header			63
BUFCAT1		BUFCAT Address (HDADD1)			64
HDADD2		Pointer to Overflow Header			65
BUFCAT2		BUFCAT Address (HDADD2)			66
FSTHDR		Pointer to First Header in List			67
LSTHDR		Pointer to Last Header in List -1			70
RECHD	Current Record Header				71
RECADD		Add. of Current Record			72
BKPTR		Back Pointer For Current Record			73
EXTENDB	←	Permanent File Extend Bit			74
EOIBIT	←	Positioned at EOI Bit			75
EOFMARK	←	Set By SAKCEOI			76

SWAP FILE FORMAT

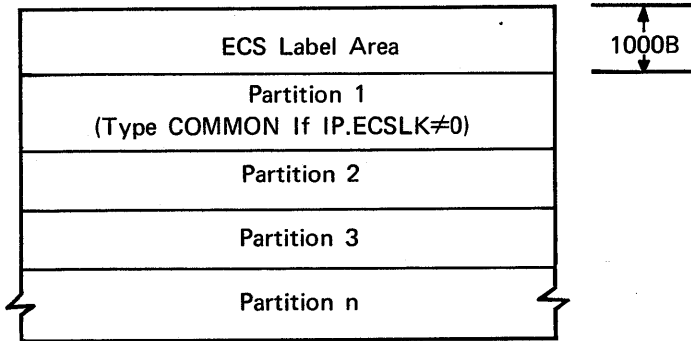


SECTION 4

EXTENDED CORE STORAGE TABLES

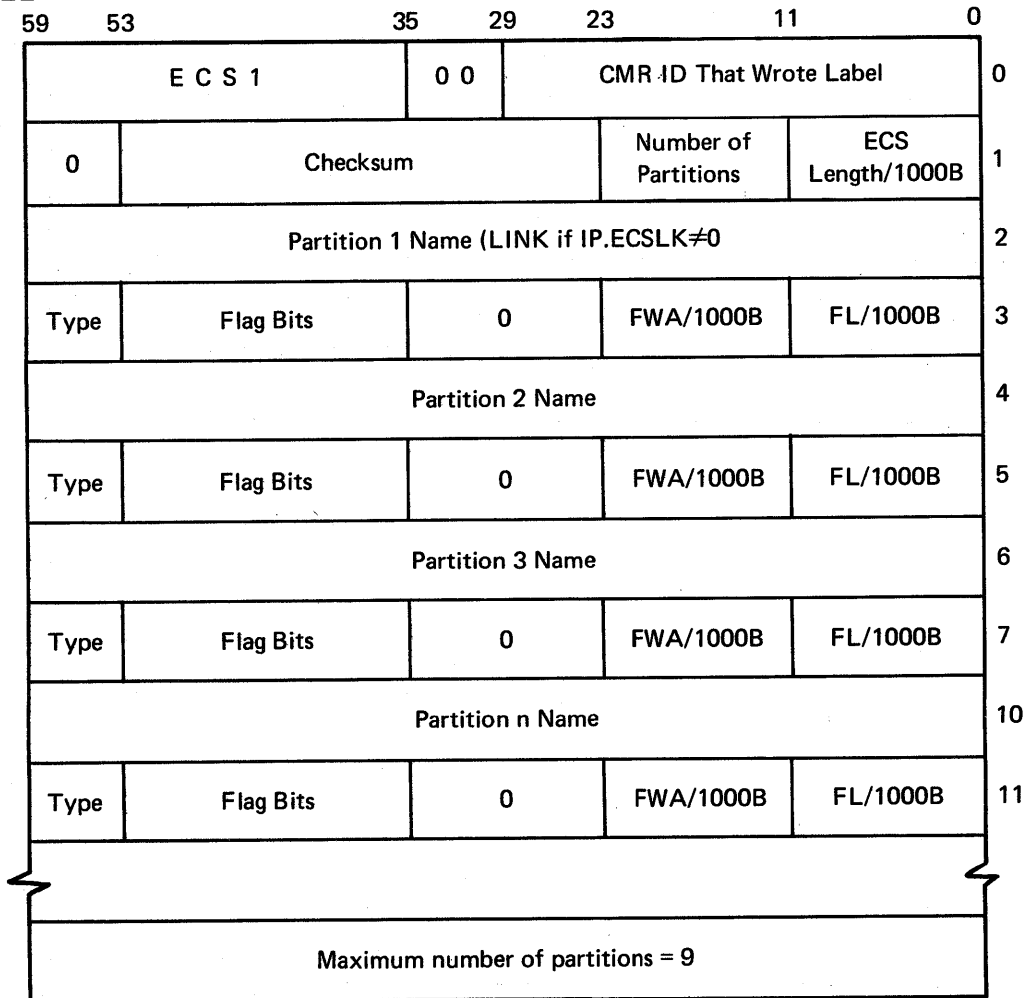
OVERALL ECS FORMAT

ECS FORMAT



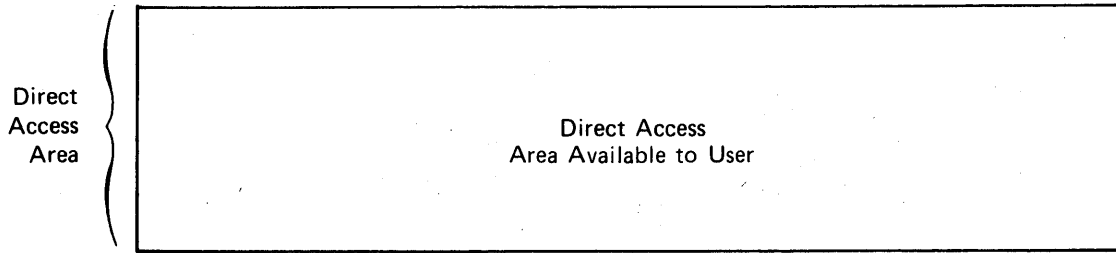
ECS label is written to one of the areas starting at 120B, 230B, 340B, 450B, 560B, or 670B.

ECS LABEL

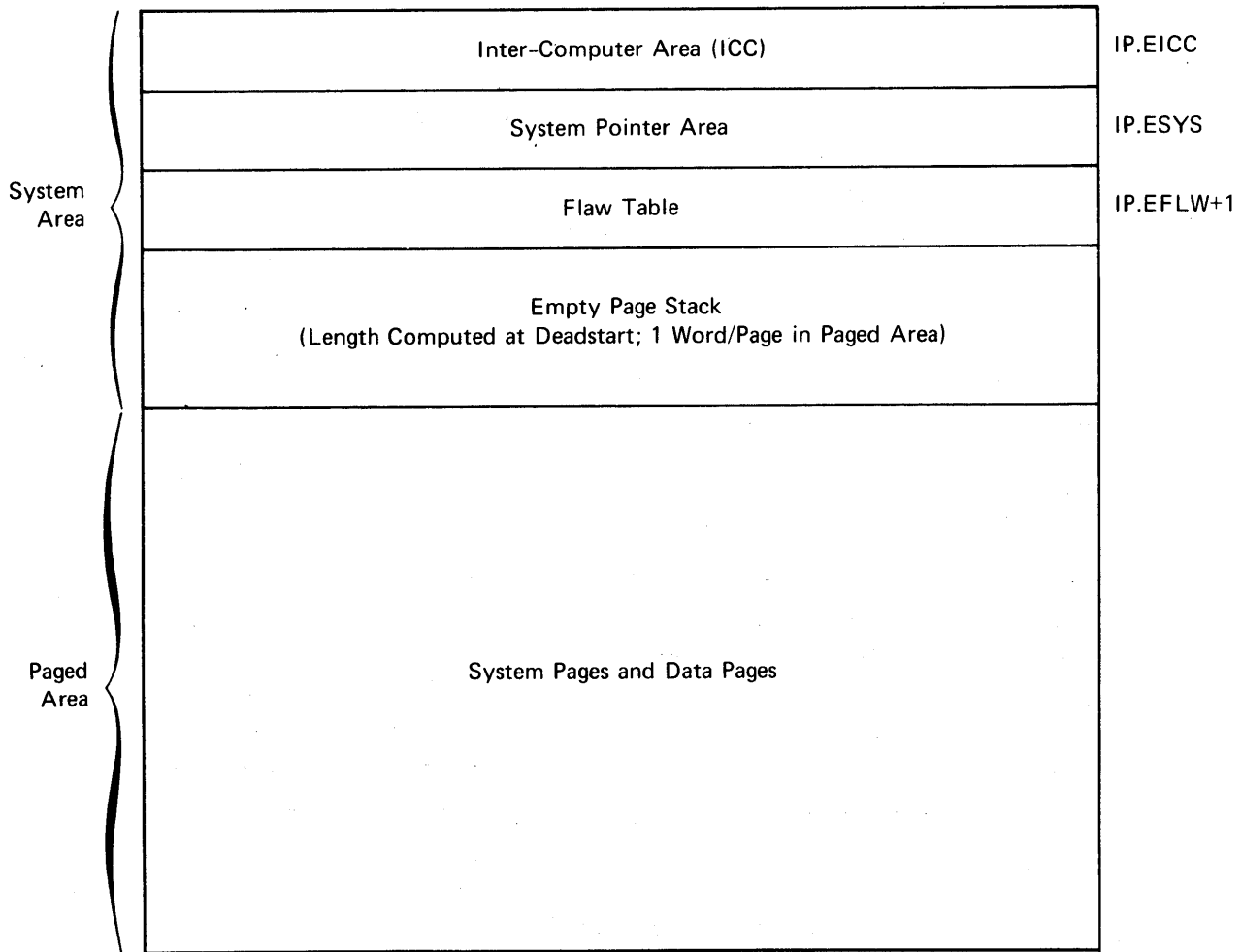


Type = 1B Direct access area
 2B Allocatable device area (system and paged area)
 3B COMMON

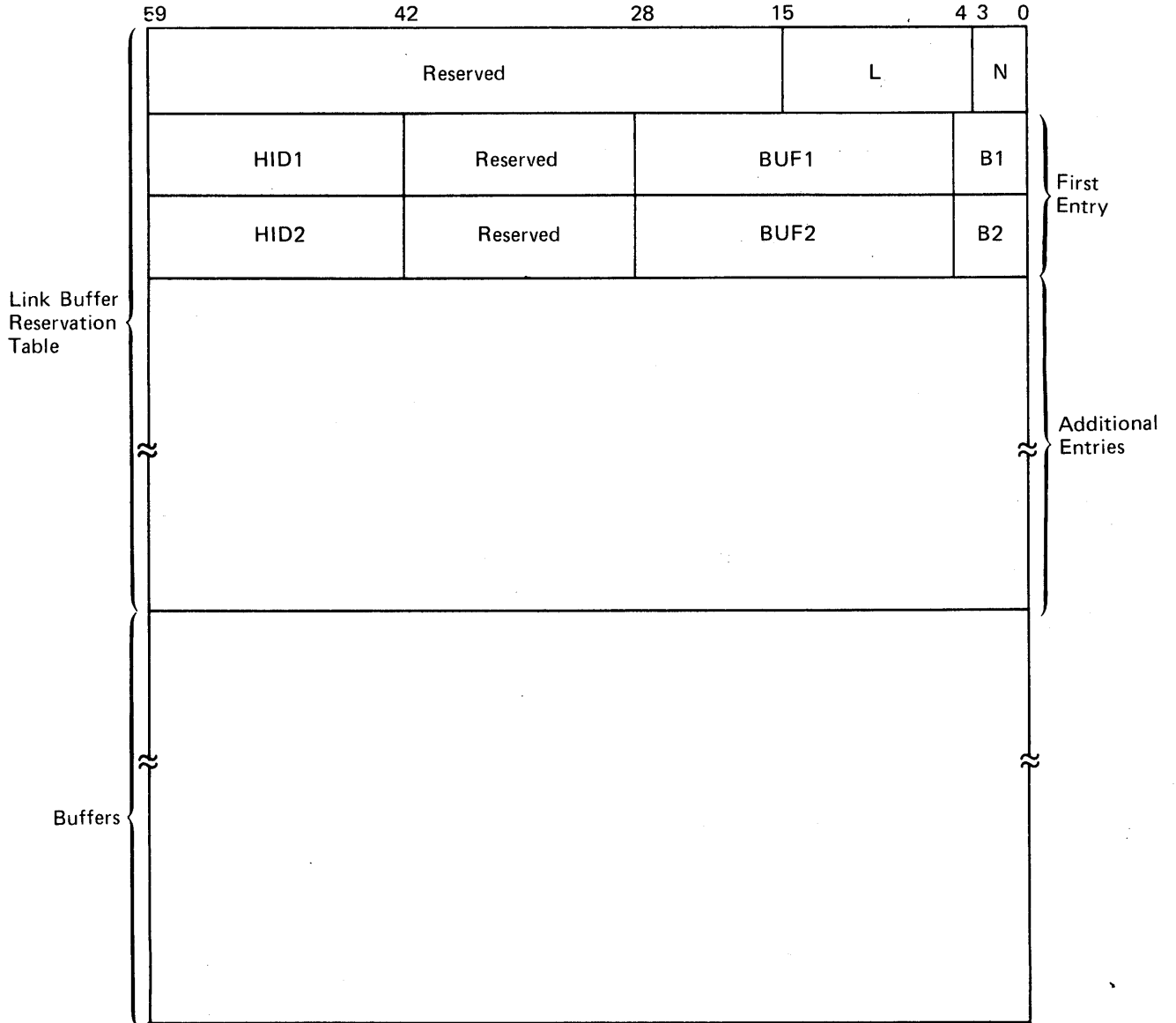
DIRECT ACCESS AREA (TYPE 1 PARTITION)



ALLOCATABLE DEVICE AREA (TYPE 2 PARTITION)

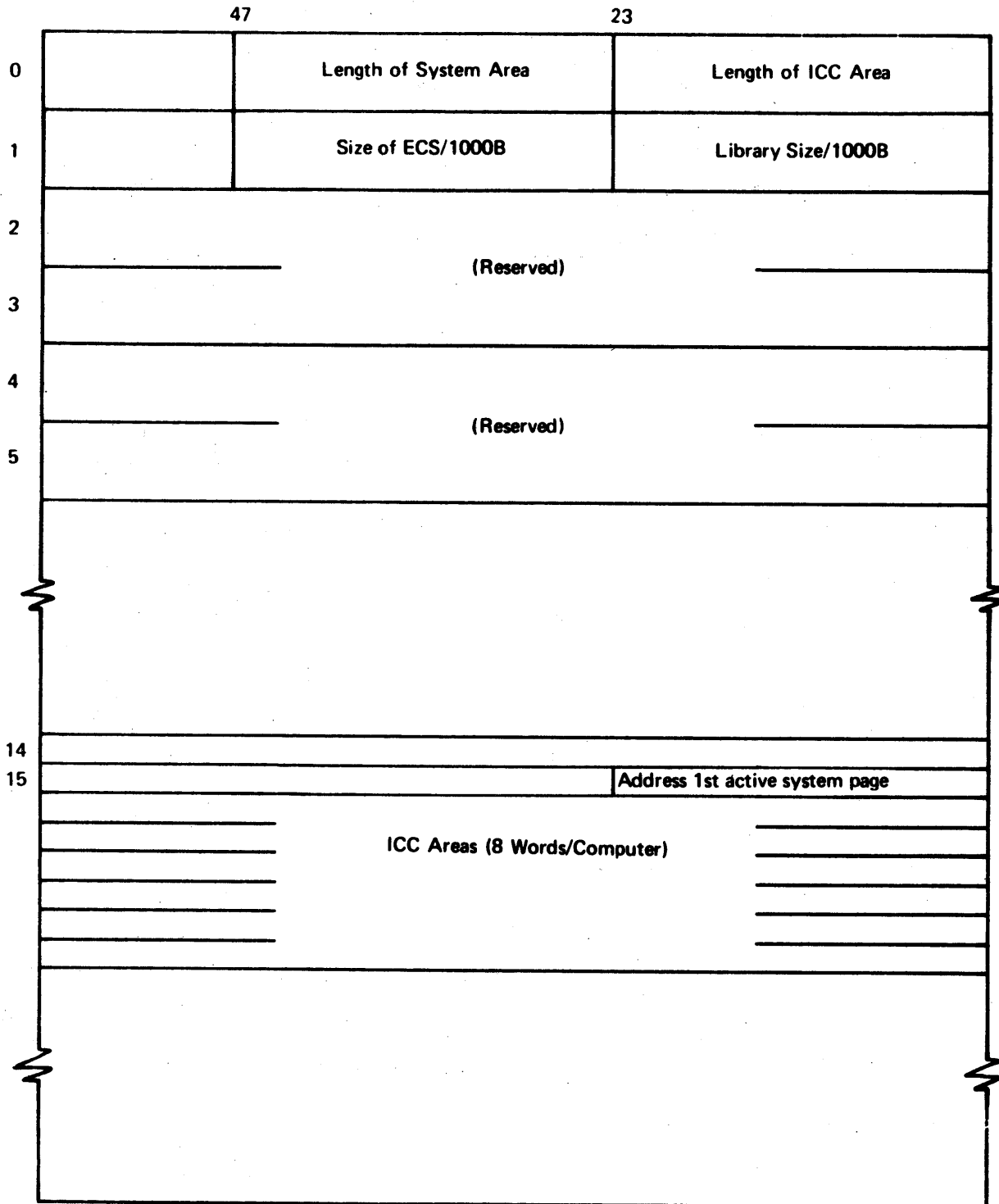


COMMON AREA (TYPE 3 PARTITION)

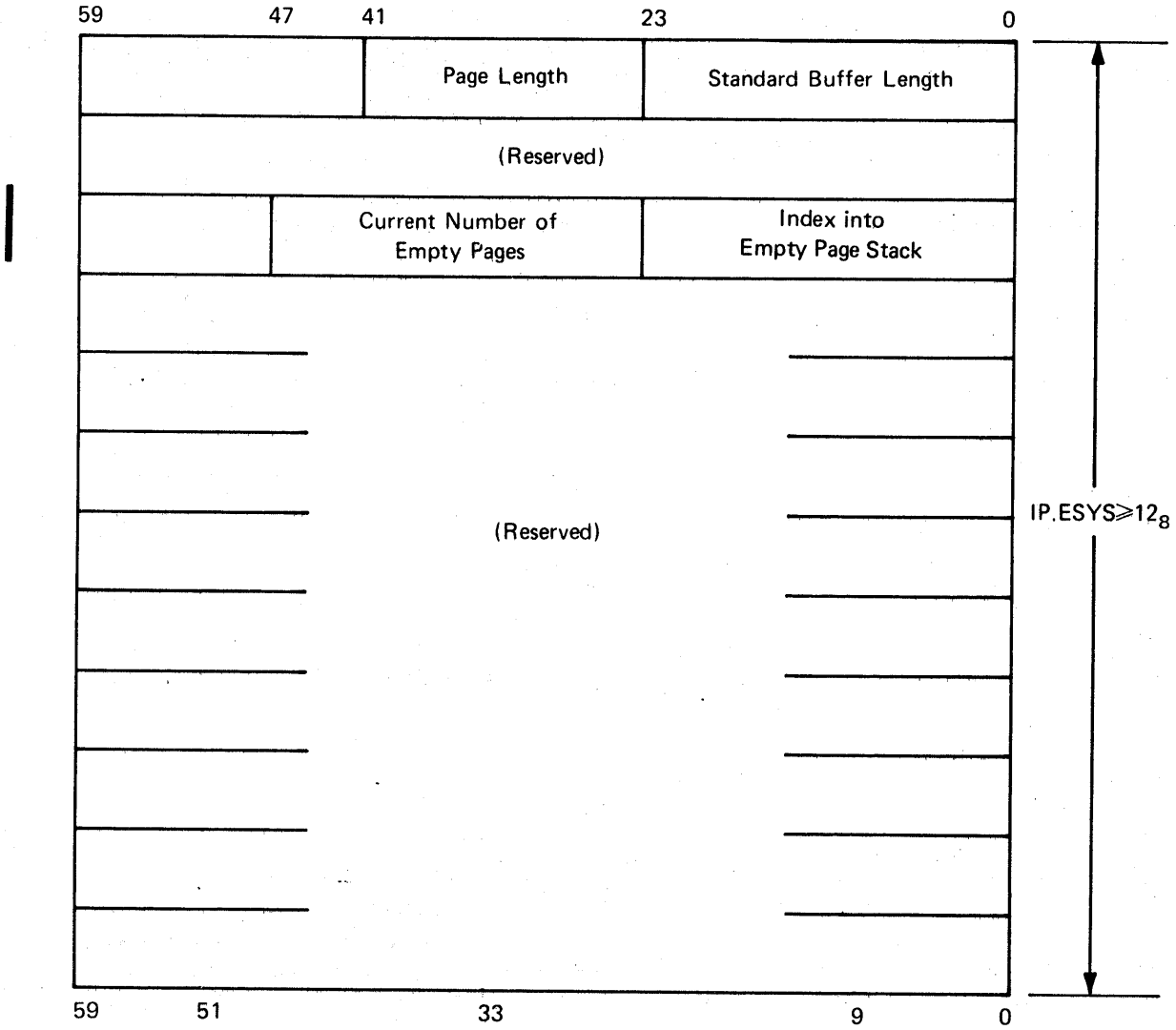


- L** Length of each buffer
- N** IP.LNKBF; number of entries in table
- HID1** Mainframe ID of first mainframe to reserve a buffer for this link
- HID2** Mainframe ID of second mainframe to reserve a buffer for this link
- BUF1** FWA of receiving buffer for first mainframe (HID1) and send buffer for 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

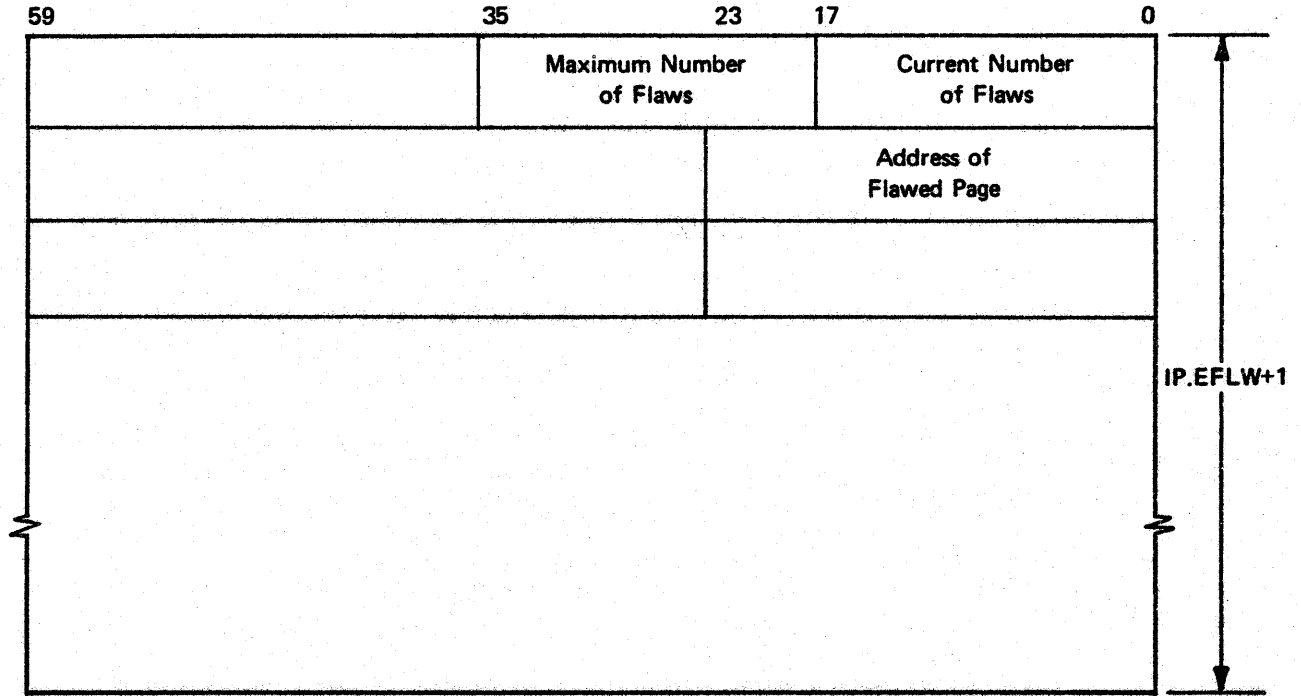
INTER-COMPUTER AREA



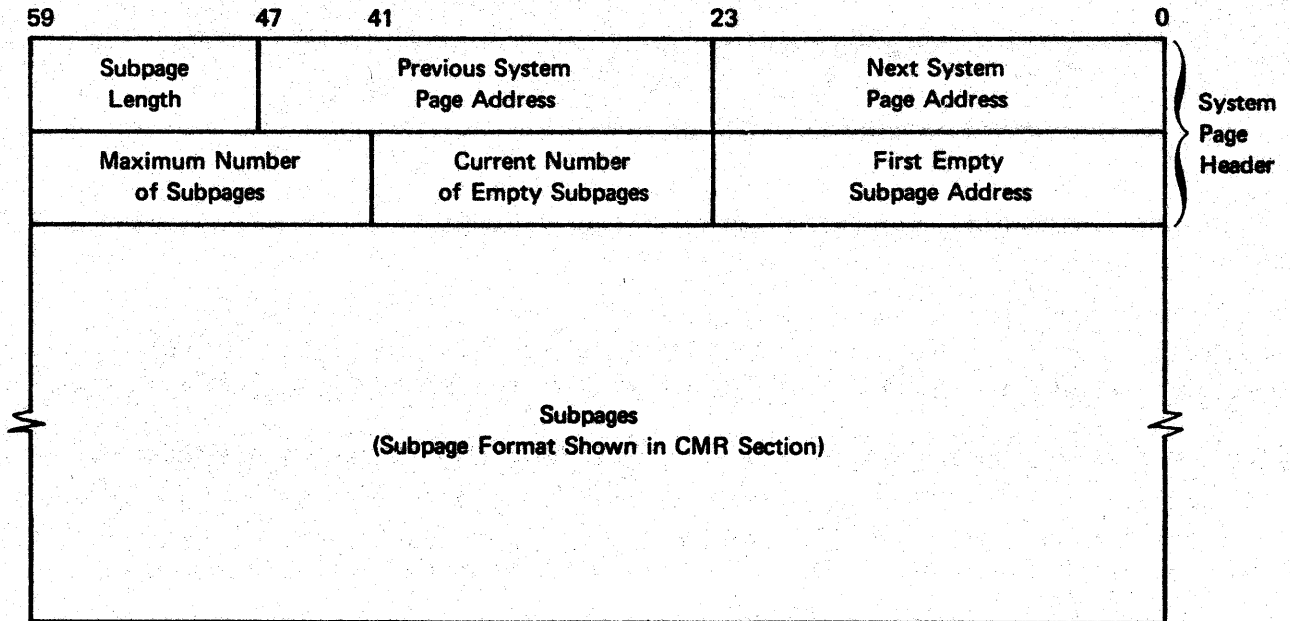
SYSTEM POINTER AREA



FLAW TABLE



SYSTEM PAGE



SEGMENTS IN AN ECS SYSTEM

<u>SEGMENT</u>	<u>DECK</u>	<u>PURPOSE</u>
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
CHKSPAC	RACDAC	SUBROUTINE TO CHECK IF ECS SPACE IS AVAILABLE
CLRCM	ECS	CLEAR CEM WORKING FLAG
CMDATA	ECDATA	COMMON DATA AREAS
CMRDIR	CMRDIR	DIRECTS 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 ISI/ISO PPU SWAPPERS
DAC	RACDAC	DEALLOCATE ECS OVERLAY PAGES
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
ECSDS	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
GETDESC	ECS	SUBROUTINE TO READ AN ECS PRU DESCRIPTOR

<u>SEGMENT</u>	<u>DECK</u>	<u>PURPOSE</u>
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
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 CTL PT 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
PLBECT	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
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
SEGPARG	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

<u>SEGMENT</u>	<u>DECK</u>	<u>PURPOSE</u>
SSFSEG	SCPT	SUBSYSTEM FUNCTION PROCESSOR
SREQ	ECLINK	SUBROUTINE TO HANDLE ECS LINK SUBCHANNEL REQUESTS
SSFS2	SCPT	SECONDARY SET OF SUBSYSTEM FUNCTIONS
STOPRU	ECS	STORE PRU
STORE	ECS	MOVE DATA FROM USERS CM BUFFER TO ECS BUFFER
SWCLEAN	ECS	CLEAN UP AFTER PARITY ON ECS SWAP FILE
SYSPROG	RESCH	CHECK STATUS OF A USER MODE SYSTEM PROGRAM
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	INITIATES EXECUTION OF USER MODE SYSTEM PROGRAMS
USERR	RESCH	RESIDENT USER MODE ENTRY/EXIT ROUTINES
WRITE	CPCIO	CPCIO WRITE EXECUTIVE
XJRSEG	RAP1	PROCESS XJR OR SAC CALLS
XSPM	SPM2	EXTENSION OF STACK PROCESSOR MANAGER

INDEX

- Abnormal job termination
 - KILL, RERUN 7-18
- Accessing device sets
 - MOUNT control card 6-4
- Accounting
 - Permanent files 6-37
- ACQUIRE macro
 - Functions GET, PEEK, COUNT 4-9
 - Searching local files 4-9
- ADDSET control card 6-41
- Allocatable device
 - Area II-4-2
 - I/O: READC, READLS, WRITEC 5-23
 - Tables related to file processing 4-29
- Allocation
 - CM 1-2; 2-1; 3-22
 - ECS 1-2
 - Overview of device allocation map (DAM) II-3-20
 - PFC allocation matrix (PAM) II-3-26
- ALTER
 - ACQUIRE macro function 4-9
- ALTER macro functions 6-13
- APF
 - Permanent file entries 6-33
 - Table II-1-60
- Archive permanent files 6-24
- Area table II-1-83
- ASCII character set A-4
 - Punched card codes
 - EBCDIC translation A-5
 - ASCII translation A-6
- Assignment
 - Automatic tape drive 4-25
 - Permanent file to RMS devices 6-1
 - PP 1-1; 3-27
- ATTACH
 - Macro function 6-11
 - Permanent files, see APF
- Attached permanent file table II-1-60
- Attribute
 - Address public sets by set attribute 6-46
 - Attribute (see RMS set terminology) 4-27
- AUDIT
 - Macro to obtain permanent files status 6-29
 - Permanent file utility routine 6-17
- Automatic
 - Generate FDB using control card 6-5
 - Recall 2-6
- Auxiliary user table II-1-107
- Bit
 - Byte and bit number 1-2
- Block
 - Block format (DA file) II-3-27
 - I block header II-3-27
 - Index block format (IS file) II-3-28
 - Record block reservation table II-1-62
 - Record block table entry II-1-114
- Breakpoint table II-1-81
- Buffer
 - CEFAP buffer area II-1-87
 - Dayfile FET and buffer area II-1-74
 - Default size for deadstart 1-3
 - INTERCOM pointer and buffer area II-1-96
 - SCB (system circular buffer) II-1-92
 - Sizes 4-19
 - Sub-page buffer II-1-90
- BULLUP
 - Create and update system bulletin file 9-3
 - File and data statements 9-1
 - Interface with INTERCOM 9-4
 - Reduce systems bulletin file 9-4
- Byte and bit number 1-2
- CALLSS macro 2-20
- Carriage control II-1-109
- CATALOG macro functions 6-9
- CEFAP buffer area II-1-87
- Central memory, see CM
 - Resident, see CMR
- CERFILE
 - System CERFILE error file 4-2

- Chain, PP 3-27
- Channel coupler entry II-1-36
- Channel reservations 3-14
- Channel status table (CST) II-1-15
- Character set
 - ASCII A-4
 - EDITLIB 8-3
 - Standard A-3,7
- CIO
 - Circular buffering 5-4
 - Codes 5-3; II-2-4
 - Concepts and codes 5-1
 - Functions 5-6
 - Use of circular buffer 5-4
- Circular input/output, see CIO
- CM allocation to control point 2-1; 3-22
- CM resident library format II-1-94
- CMR 2-8; II-1-1
 - Areas summary 2-9
 - Library directory II-1-93
 - Loader, see LDCMR 10-1
 - Pointer area II-1-2
 - Segmentation for ECS system 2-13
 - Table 2-10
 - Table function summary 2-11
- Codes
 - CIO codes 5-3; II-2-4
 - CIO concepts and codes 5-1
 - Device codes II-1-37
 - Disposition code values II-1-52
 - FET codes II-2-8
 - Punched card codes
 - ASCII translation A-6
 - EBCDIC translation A-5
 - Stack processor order codes II-1-61
- Common area II-4-2.1
- Communication word II-1-23
- Console display, see DSD
- Control
 - Job control area II-1-76
- Control Card
 - Automatic generation of FDB 6-5
 - Buffer 7-13
 - LDCMR parameters 10-1
 - Processing flowchart 7-14
 - Processing terminated by EXIT 7-15
- Control point
 - CM allocation to control point 2-1; 3-22
 - Control point area II-1-17
- Conversion table
 - Hexadecimal-octal A-8
- COUNT
 - ACQUIRE macro function 4-9
- CP
 - Monitor 3-12
 - RA+ 1 requests; CP-system communication 2-4
- Create device sets 6-41
- Creating and updating system bulletin file,
 - BULLUP 9-3
- CST channel status table II-1-15
- D
 - PP direct cells, D. symbols 3-1,3
- Dayfile
 - Dayfile FET and buffer area II-1-74
 - System dayfile 4-1; 7-17
- DDP entry II-1-36
- DDT (dismountable device table) II-1-66
- Deadstart
 - Creating using EDITLIB 8-1
 - Default buffer size 1-3
 - Loading operating system 1-4
 - PP usage in deadstart load 1-6
 - Tape records and overlays 1-7
- Delay count 2-6
- DELSET control card
 - Deleting device sets 5-39; 6-39
- Descriptors, record II-1-90
- Device
 - Activity table entry II-1-59
 - Allocation map (DAM) II-3-20
 - Codes II-1-37
 - Permanent file set label II-3-13
 - Tables summarized 4-21
- Device sets 4-27
 - Accessing sets, MOUNT 6-44
 - Creating and processing 6-37
 - Creating sets using ADDSET 6-41
 - Deleting sets using DELSET 6-42
 - Directing files to specific sets, REQUEST 6-45
 - Disassociating sets, DSMOUNT 6-45
 - Implicit reference to sets, SETNAME 6-46
 - I/O processing 5-35
 - Reconstructing tables, RECOVER 6-42
 - Shared device sets 4-28
- Device status table II-1-63
 - Multiple access entries II-1-63

- Diagnostics
 - ROUTE macro 4-4
- Direct access area II-4-2
- Direct cells
 - PP, D. symbols 3-1,3
- Directive
 - EDITLIB directives 8-12
 - EDITLIB parameters 8-9
 - Used with DUMPF utility 6-18
- Directory
 - CMR library directory II-1-93
- Dismountable device table (DDT) II-1-66
- Dismountable pack processing, I/O detail 5-35
- Display console, see DSD
 - Entry II-1-36
- Disposition code values II-1-52
- DSD/INTERCOM communication through station,
 - MSG table II-2-28
- DSMOUNT
 - Disassociating device sets 5-39; 6-45
- Dual access
 - Device entry II-1-59
- Dummy entry for dual access device II-1-59
- Dump (System Dynamic Dump) 11-1
- DUMPF utility 6-18
 - Clearing permanent files 6-18
 - Directives used with utility 6-20
 - S tape usage required 6-18

- EBCDIC with punched card codes and ASCII translation A-6
- ECS
 - Allocation 1-2
 - Buffering of RMS files 5-40
 - CMR segmentation for ECS system 2-13
 - Error recovery 2-18
 - Format II-4-1
 - Label II-4-1
 - Paging 1-3
 - Partitioning formats II-4-2,2.1
 - Overall ECS format II-4-1
 - Segments II-4-6
- EDITLIB
 - Basic usage of system EDITLIB 8-1
 - Deadstart creation 8-2
- Empty page stack II-1-89
- ENDSEG
 - Macros for CMR segmentation 2-14

- Entry formats
 - EPNT, ERT, PNUT, PNT II-1-95
- Entry point names II-2-7
- Entry table II-1-84
- EPNT entry format II-1-95
- Equipment status table (EST) II-1-35
- Error
 - Codes, reprieve processing 7-21
 - Conditions detected by stack processor 5-32
 - ECS error recovery 2-18
 - RMS hardware error 5-35
 - System CERFILE error file 4-2
- Error file
 - CEFAP buffer area analyzer program II-1-87
- ERT entry format II-1-95
- EST (equipment status table) II-1-35
- Exchange jumps
 - Exchange package II-1-16
 - Scheduler request stack in system exchange package area 7-6
 - System job exchange package area II-1-21
- EXIT card processing terminated 7-15
- EXTEND macro functions 6-14
- Extended core storage, see ECS

- FDB
 - Automatic generation 6-5
 - Table II-2-20
- FET
 - Codes II-2-8
 - Dayfile FET and buffer area II-1-74
 - Table II-2-3
- Field access flag 3-11
- Field length, see FL
- Field length override 8-12
- File
 - Definition block II-2-20
 - File environment table, see FET
 - File information table, see FIT
 - Swap file format II-3-35
- File name table (FNT) II-1-39
- Files
 - EDITLIB 8-7
 - LDCMR 10-4
- FIT table II-2-10
- FL
 - Allocated storage definition 2-3

Flaws

- Device relabeling 6-40
- Flaw table II-4-5

FLO 8-12

FNT (file name table) II-1-39

FSTT

- AK file II-3-33
- DA file II-3-31
- IS file II-3-29

GET

- ACQUIRE macro function 4-9
- GETPF macro function 6-13
- GOTO macros for CMR segmentation 2-14

Header

- I block header II-3-27
- W record header II-3-27

Hexadecimal-octal conversion table A-8

Host ID local mainframe 6-4; 7-23

ID table II-1-71

Index block format (IS file) II-3-28

Input queue 4-3

Interactive graphics user table II-1-113

INTERCOM 4

- Auxiliary user table II-1-107
- Buffer area II-1-102
- BULLUP interface, SYSBULL 9-4
- Driver communication area II-1-100
- Input/output interface with JANUS 7-3
- Multiplexer table, subtables II-1-55
- Pointer and buffer area II-1-96
- User table II-1-103

Interlock LDCMR 10-5

Inter-computer area II-4-3

I/O

- Allocatable device I/O; READC, READLS, WRITEC 5-23
- Device set processing 5-35
- Dismountable pack processing, I/O detail 5-35
- ROUTE macro used in queueing 4-4
- Tables summarized 4-17

JANUS interfaces 7-3

Input/output interface with INTERCOM 7-3

JDT 7-10

- Swap job information 2-3; 7-8
- Table entry II-1-77

Job

- Control area II-1-76
- Descriptor number 2-3
- Descriptor table II-1-77
- Flow 7-1
- Function of VSN card in prescheduling 4-25
- Load from tape, TLOAD routine 7-2
- Origin II-2-2
- Post-processing utilities 7-19
- Roll-out 7-7
- Scheduling execution through class assignment in job control area 7-9
- Scheduling queues 7-10
- Termination, abnormal KILL, RERUN 7-18
- Termination, normal 7-15

KILL, abnormal job termination 7-18

Label

- LABELMS 6-37
- Permanent file set device II-3-13

LDCMR

- Control card parameters 10-1
- Files 10-4
- Interlock 10-5
- Overview 10-1
- Reserved names 10-6

Library

- CM resident library format II-1-94
- CMR library directory II-1-93

Library name

- Table entry 8-25; II-1-93
- Table interfaces 8-32

LID (logical identifiers) 7-23

Link

- ID linked mainframe 6-4; 7-23
- Identification values II-1-54
- Staging, local and link 6-4
- List core-image dump utility 11-2

List system dynamic dump file 11-2
 LISTCID utility 11-2
 Load
 Job load from tape, TLOAD routine 7-2
 PP usage in deadstart load 1-6
 LOADPF utility 6-17,23
 Loading permanent files from tape 6-23
 Local
 Host ID local mainframe 6-4; 7-23
 Staging, local and link 6-4
 Local files 4-2
 ACQUIRE macro usage in searching local files 4-9
 Local file names II-2-5
 Logical file name
 Logical identifiers (LID) 7-23
 Logical Flaw Table II-3-22

 M.xx monitor functions 3-16,23
 Macros
 CMR segmentation 2-14
 Permanent file functions, parameters 6-1
 Magnetic tape entry II-1-35
 Mailbox II-1-70
 Mainframe
 Host ID local 6-4; 7-23
 Link ID linked 6-4; 7-23
 User assignment of system files to mainframe
 B 6-4
 Mask, reprieve code 7-19
 Monitor
 CP processing 3-14
 Functions 3-16; II-1-33
 PP processing 3-22
 Structure 3-15
 MOUNT control card 6-44
 Mounted set table (MST) II-1-65
 MSG table
 DSD/INTERCOM communication through
 station II-2-28
 MST (mounted set table) II-1-65
 MTR, see monitor
 Multiplexer entry II-1-36
 Multi-user job table II-1-112

 Name reservations, PP II-1-24
 NFL 8-12
 Nominal field length 8-12

 Normal job termination 7-15
 Normal reprieve processing 7-21,22
 Null function 3-20

 Output queue 4-3
 Overcommitment of tape drives 4-26

 Pack queue 7-17
 Page
 Empty page stack II-1-89
 Paging
 ECS paging 1-3
 Parameters
 Permanent file functions 6-2
 Password
 Public 6-5,35
 Universal 6-4,35
 PEEK
 ACQUIRE macro function 4-10
 Periodic recall 2-6
 Peripheral processor, see PP
 Peripheral job table II-1-73
 PERM macro 6-16
 Permanent files 7-15
 APF entries 7-15
 Assignment to RMS devices 6-1
 AUDIT macro to obtain status 6-29
 Automatic archive feature 6-24
 Catalog, see PFC
 Clear files using DUMPF 6-18
 Copy or transfer files, TRANSPF 6-25
 Directory II-3-1,2
 Functions, parameters, and macros 6-1
 Files interlocks 6-33
 Loading files from tapes, LOADPF 6-23
 Set device label II-3-13
 Utility routines 6-17
 Permission, universal 6-4,35
 PFC
 Allocation matrix (PAM) II-3-26
 Entry II-3-6
 Overview II-3-5
 PFD
 Entry II-3-2
 Overview II-3-1
 Usage 6-33
 PFL 8-12

Physical flaw table II-3-22

PNT
 Entry format II-1-95
 PP program name table entry II-1-94

PNUT entry format II-1-95

Pointer
 CMR pointer area II-1-2
 INTERCOM pointer and buffer area II-1-96

PP
 Assignment 1-1; 3-27
 Chain 3-27
 Communication area II-1-22
 Communication register 3-4
 Communication with CP monitor 2-5
 Direct cells, D. symbols 3-1,3
 Job queue 3-29
 PNT II-1-93
 Program name reservations II-1-24
 Program name table entry 8-28; II-1-94
 Resident, R. functions 3-5
 Status words II-1-16
 Usage in deadstart load 1-6

PPLIB 3-5

Program name
 PP PNT entry II-1-94
 PP reservations II-1-24
 PP table entry 8-28

Public password 6-5,35

PURGE macro functions 6-15

Queue
 Definitions and processing 4-3
 Files 4-3

R, PP resident functions 3-5

RA
 RA communication area II-2-1
 RA used in allocated storage definition 2-3
 RA+ 1 requests, CP-system communication 2-4

READC
 Allocatable device I/O 5-23

READLS
 Allocatable device I/O 5-24

Recall
 Automatic 2-6
 Periodic 2-6

Record
 Block reservation table 4-32; II-1-62
 Block table entry 4-32; II-1-114
 Descriptors II-1-90
 W record header II-3-27

RECOVER control card
 Reconstruct device sets tables 6-42

Recovery
 ECS error recovery 2-18

RECOVR macro 7-21

Reference address, see RA

Register
 PP communication register 3-4

RENAME macro functions 6-13

REPRIEVE macro 7-19

Reprieve processing 7-19
 Error codes 7-19
 Mask 7-19

REQ function parameter list II-2-22

Request scheduling table II-1-62

Request stack entry II-1-61

RERUN 7-18

Reserved name LDCMR 10-6

Resident
 CM resident library format II-1-94
 PP resident, R. functions 3-5

Resource scheduling conflicts 4-26

RMS
 ECS buffering of RMS files 5-40
 Hardware error 5-35
 Permanent file assignment to RMS devices 6-1
 Set terminology 4-27
 System circular buffer (SCB) II-1-92
 Tables 4-29,32

Roll-out job 7-7

Rotating mass storage, see RMS

ROUTE
 Macro 4-4
 Macro diagnostics 4-7

RPV program 7-19

SCB II-1-92

Scheduler
 Integrated scheduler 7-4,5
 Performance table (SCHPT) II-1-75
 Request stack in system exchange package
 area 7-6

Scheduling
 Job scheduling queues 7-10
 Scheduling execution through class assignment in
 job control area 7-9

SCP 2-19

SEGDEF
 Macro for CMR segmentation 2-14

Segment table II-1-85

Segmentation
 CMR segmentation for ECS system 2-13; II-4-6
 Macros for CMR segmentation 2-14

Sequencer table II-1-64

Set
 RMS set terminology 4-27
 Set member table (mass storage) II-3-18

SETNAME
 Implicit reference to device sets 6-46

Spot name table (SNT) II-2-26

Stack processor
 Empty page stack II-1-89
 Error conditions detected 5-33
 Order codes II-1-61
 Processing request 5-27
 Request stack entry II-1-61
 Scheduler request stack in system exchange
 package area 7-6
 Table usage 5-32

Staging, local and link 6-4

Standard character set A-3,7

Status
 AUDIT macro obtains status of permanent files
 6-29
 PP status words II-1-16

Storage move 2-3

Subchannel table (SCHT) II-2-24

Subdirectory flag II-3-2

Subpage buffer II-1-90

Swap
 File format II-3-35
 Job information in JDT 2-3; 7-7

Syntax, EDITLIB 8-4

SYSBULL
 BULLUP interface with INTERCOM 9-4

Systems bulletin file
 Reduce file, BULLUP 9-4

System bulletin utility, BULLUP 9-1
 Create and update system bulletin file,
 BULLUP 9-3

System circular buffer II-1-92

System control point 2-19

System dynamic dump 11-1

System files, see dayfile and CERFILE

System job exchange package area II-1-21

System page II-4-5

System pointer area II-4-4

Tables
 Related to file processing on RMS devices 4-29,32
 Stack processor table usage 5-32

Tapes
 Automatic tape drive assignment 4-25
 Job load from tape, TLOAD routine 7-2
 Overcommitment of tape drives 4-26
 Staging table II-1-60
 Tapes table 2-11; 4-26.1; II-1-69

TLOAD routine
 Job load from tape 7-2

TRANSPF utility 6-17,25

UCP 2-19

Unit record equipment entry II-1-35

Universal password 6-4,35

Universal permission 6-4,35

User control point 2-19

User table
 Auxiliary table II-1-107
 Driver words II-1-104
 High speed auxiliary user table II-1-111
 IGS user table II-1-113
 Multiuser job table II-1-112

Utility routines, permanent files
 AUDIT 6-29
 DUMPF 6-18
 LOADPF 6-23
 TRANSPF 6-25

VERIFYJ Macro 4-16.3

VSN card in job prescheduling 4-25

VSNBUF, volume serial number buffer II-1-68

WRITEC
 Allocatable device I/O 5-26

ZZZZZxx (Local File Names) II-2-5

Local

Staging, Local and Link 6-4
Host ID Local Mainframe 6-4 7-23

Local Files

Local Files 4-2
ACQUIRE Macro Usage in Searching Local Files 4-9
Local Files - Names II-2-5

Logical File Name

Logical File Name
Logical Identifiers, (LID) 7-23

M.xx

M.xx Monitor Functions 3-16, 3-25

Macros

Macros for CMR Segmentation; GOTO, SEGDEF, ENDSEG 2-14
Permanent File Functions; Parameters and Macros 6-1

Mailbox

Mailbox II-1-70

Mainframe

Host ID Local Mainframe 6-4, 7-23
Link ID Linked Mainframe 6-4, 7-23
System Files that Reside at Mainframe-A 6-4
User Assignment of System Files to Mainframe-B 6-4

Monitor

CP Monitor Processing 3-14
PP Monitor Processing 3-23
Monitor Structure 3-15

Monitor Functions 3-16, II-1-34

MOUNT

Accessing Device Sets Using MOUNT Control Card 6-44

Mounted Set Table

Mounted Set Table (MST) II-1-65.1

MSG Table

DSD/INTERCOM Communication through Station - MSG Table II-2-28

MST

Mounted Set Table (MST) II-1-65.1

MTR

MTR; Also See Monitor

Name

PP Program Name Reserved 3-11

Normal

Normal Job Termination 7-15

Overcommitment

Overcommitment of Tape Drives 4-26

Pack

Pack Queue 7-17

Page

Empty Page Stack II-1-89

Paging

ECS Paging 1-3

Parameters

Permanent File Functions; Parameters and Macros 6-1

PEEK

ACQUIRE Macro Functions GET, PEEK, COUNT 4-9

Periodic

Periodic Recall 2-6

Peripheral Processor

Peripheral Processor; See PP

Peripheral Job Table II-1-74

PERM

PERM Macro - Description 6-16

Permanent

Permanent File Interlocks 6-33

Permanent File Assignment to RMS Devices 6-1

Permanent File Functions; Parameters and Macros 6-1

Permanent File Utility Routines: DUMPF, LOADF, TRANSPF, AUDIT 6-17

Automatic Archive Feature of Permanent Files 6-24

Clearing Permanent Files Using DUMPF 6-18

Loading Permanent Files from Tapes Using LOADPF 6-19

Copying or Transferring Permanent Files Using TRANSPF 6-20

Use of AUDIT Macro to Obtain Status of Permanent Files 6-30

Permanent File Creation During System Deadstart Load 6-35

APF Entries for Permanent Files 7-16

Permanent File Catalog; See PFC

Permanent File Set Device Label II-3-13

PFC

PFC Entry II-3-6

PFC Overflow II-3-5

PFC Allocation Matrix (PAM) II-3-26

PFD

PFD Usage 6-35

PFD Overview II-3-1

PFD Entry II-3-2

PFMOVE

Permanent File Utility Routines: DUMPF, LOADF, TRANSPF, AUDIT 6-17

Physical Flow Table

Physical Flow Table II-3-22

PNT

PP Program Name Table Entry (PNT) II-1-94

EPNT, ERT, PNUT, PNT Entry Format II-1-95

PNUT

EPNT, ERT, PNUT, PNT Entry Format II-1-95

Pointer

CMR Pointer Area II-1-2
INTERCOM Pointer and Buffer Area II-1-96

PP

PP Assignment 1-1
PP Usage in Deadstart Load 1-6
PP Communication with CP Monitor 2-5
PP Direct Cells; D. Symbols 3-2
PP Communication Register 3-4
PP Resident; R. Functions 3-5
PP Monitor Function Codes and Mnemonics 3-5
PP Program Name Table Entry 8-72
PP Status Words II-1-16
PP Communication Area II-1-22
PP Program Name Reservations II-1-24
PP Program Name Table Entry (PNT) II-1-94

Program Name Table

PP Program Name Table Entry 8-72
PP Program Name Reservations II-1-24
PP Program Name Table Entry (PNT) II-1-94

PURGE

PURGE Macro - Functions Described 6-15

Queue

Queue Definitions and Processing 4-3
Control Card Processed in Job Advancing Queue 7-12, 7-13, 7-15

Queues

Job Scheduling Queues 7-11

R

PP Resident; R. Functions 3-5

RA

RA Used in Allocated Storage Definition 2-3
RA + 1 Requests; CP / SCOPE Communication 2-4
RA Communication Area II-2-1

READC

Allocatable Device I/O; READC, READLS, WRITEC 5-23

READLS

Allocatable Device I/O; READC, READLS, WRITEC 5-23

Recall

Automatic Recall 2-6
Periodic Recall 2-6

Record

Record Block Reservation Table 4-32, II-1-62
Record Descriptors II-1-90
Record Block Table Entry 4-32, II-1-114
W Record Header II-3-27

RECOVER
 Reconstructing Tables of Device Sets Using RECOVER Control Card 6-42

Recovery
 ECS Error Recovery 2-18

RECOVR
 User Call to RECOVR Generates Call to RPV 7-19,7-21

Reference Address
 Reference Address; See RA

Register
 PP Communication Register 3-4

RENAME
 RENAME Macro - Functions Described 6-13

REQ
 REQ Function Parameter List II-2-22

RERUN
 Abnormal Job-Termination; KILL, RERUN 7-18

Reserved
 PP Program Name Reserved 3-11

Reserved Name
 LDCMR Reserved Name 10-6

Resident
 CM Resident Library Format II-1-94

Resident
 PP Resident; R. Functions 3-5

RMS
 Set Terminology 4-27
 RMS Recording Technique 4-31
 RMS Tables Summarized 4-29
 RMS Hardware Error 5-35
 ECS Buffering of RMS Files 5-40
 Permanent File Assignment to RMS Devices 6-1
 System Circular Buffer (SCB) II-1-92

Roll-Out
 Job Roll-Out 7-7

Rotating Mass Storage
 Rotating Mass Storage; See RMS 4-27

ROUTE
 Use of ROUTE Macro in I/O Queueing 4-2
 ROUTE Macro 4-4
 ROUTE Macro Diagnostics 4-7

RPV
 User Call to RECOVR Generates Call to RPV 7-19,7-21

Scheduler
 Integrated Scheduler 7-4, 7-5
 Scheduler Request Stack in System Exchange Package Area - Illustration 7-6
 Scheduler Performance Table (SCHPT) II-1-75

Scheduling
 Scheduling of Execution through Class Assignment in Job Control Area 7-9
 Job Scheduling Queues 7-11

SEGDEF
 Macros for CMR Segmentation; GOTO, SEGDEF, ENDSEG 2-14

- Segment
 - Segment Table II-1-85
 - Segment in an ECS System II-1-86
- Segmentation
 - CMR Segmentation for ECS System 2-13
 - Macros for CMR Segmentation; GOTO, SEGDEF, ENDSEG 2-14
- Sequencer Table
 - Sequencer Table II-1-65
- Set
 - RMS Set Terminology 4-27
- SETNAME
 - Implicitly Referencing Device Sets Using SETNAME Control Card 6-46
- Set Member Table
 - Set Member Table (Mass Storage) II-3-18
- Spot Name Table
 - Spot Name Table (SNT) II-2-26
- Stack
 - Processing Stack Request 5-27
 - Stack Processor Order Codes 5-29, II-1-61
 - Stack Processor Table Usage 5-31
 - Error Conditions Detected by the Stack Processor 5-33
 - Scheduler Request Stack in System Exchange Package Area - Illustration 7-6
 - Request Stack Entry II-1-61
 - Empty Page Stack II-1-89
- Staging
 - Staging, Local and Link 6-4
- Status
 - Use of AUDIT Macro to Obtain Status of Permanent Files 6-30
 - PP Status Words II-1-16
 - Dual Access Status Word Pair II-1-63
- Storage Move
 - Storage Move 2-3
- Subchannel
 - Subchannel Table (SCHT) II-2-24
- Swap
 - Swap Job Information Contained in JDT 2-3, 7-8
 - Swap File Format II-3-35
- Syntax
 - EDITLIB Syntax 8-4
- SYSBULL
 - BULLUP Interface with INTERCOM; SYSBULL 9-5
- Systems Bulletin File
 - Reducing a Systems Bulletin File Using BULLUP 9-4
- System Bulletin Utility
 - System Bulletin Utility; See BULLUP 9-1
 - Creating and Updating a System Bulletin File Using BULLUP 9-3
- System Control Point 2-19
- System Files
 - System Files; See Dayfile and CERFILE
- System Page
 - System Page II-4-5
- System Pointer Area
 - System Pointer Area II-4-4
- S Tape
 - S Tape Usage Required with DUMPF Utility 6-17, 6-18

Table

Stack Processor Table Usage 5-31

Tables

Tables Related to File Processing on RMS Devices 4-27

Tape

Automatic Tape Drive Assignment 4-25

Overcommitment of Tape Drives 4-26

Job Load from Tape Using TLOAD Routine 7-2

Tapes Table

Tapes Table II-1-69

Tape Staging Table

Tape Staging Table II-1-60

TLOAD

Job Load from Tape Using TLOAD Routine 7-2

TRANSF

Copying or Transferring Permanent Files Using TRANSF 6-20

TRANSPF

Permanent File Utility Routines: DUMPF, LOADF, TRANSPF, AUDIT, 6-17

User

User Table Driver Words II-1-103

Auxiliary User Table II-1-107

High Speed Auxiliary User Table II-1-111

Multi User Job Table II-1-112

IGS User Table II-1-113

Utility

Permanent File Utility Routines: DUMPF, LOADF, TRANSPF, AUDIT 6-17

Directives Used with DUMPF Utility 6-18

VSN

Function of VSN Card in Job Prescheduling 4-25

VSNBUF

VSNBUF - Volume Serial Number Buffer II-1-68

WRITEC

Allocatable Device I/O; READC, READLS, WRITEC 5-23

COMMENT SHEET

MANUAL TITLE CDC SCOPE System Programmer's Reference Manual

PUBLICATION NO. 60306500 REVISION L

FROM: NAME: _____
BUSINESS
ADDRESS: _____

COMMENTS:

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

PRINTED IN U.S.A.

AA3419 REV. 1/79

NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE

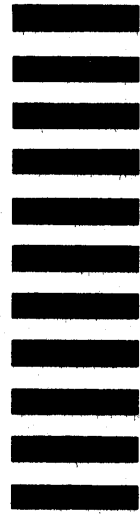
FOLD

FOLD

FIRST CLASS
PERMIT NO. 8241
MINNEAPOLIS, MINN.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

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

CONTROL DATA
CORPORATION

CORPORATE HEADQUARTERS, 8100 34th AVE. SO., MINNEAPOLIS, MINN, 55440
SALES OFFICES AND SERVICE CENTERS IN MAJOR CITIES THROUGHOUT THE WORLD

Litho in U.S.A.