60407000

CONTROL DATA CORPORATION

KRONOS® 2.1 REFERENCE MANUAL

Volume 1 of 2

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

REVISION RECORD

REVISION	DESCRIPTION
A	Manual released.
(7-1-73)	
B	Revised to describe 580-12 line printer support, 667/669 tape unit support, TCS and QFM
(6-17-74)	macros, and auxiliary device requests, and to correct various technical and typographical
	errors.
C	Revised to update the manual for KRONOS 2.1.1. All modification through level 7 are in-
(12-6-74)	cluded in this revision. The RUN control card and references to FORTRAN 2.3 are removed.
	The ALGOL 4 control card is added. Miscellaneous corrections and clarifications are also
	included.
D	Reprint with revision; obsoletes all previous editions. Manual revised to describe the
(6-17-75)	KRONOS 2.1.2 operating system at corrective code level 404, and reorganize the manual into
	two volumes.
	The following additions have been made due to KRONOS 2.1.2 modifications: enhanced user
	controls as described under Validation and the LIMITS control card; accounting feature (use of
	system resource unit and new dayfile messages); O26/O29 keypunch conversion mode;
	RERUN/NORERUN control cards for I/O Queue protection feature; subsystem symbolic
	name in control language; new control cards (ENQUIRE, LENGTH, STIME, SUMMARY, CLEAR,
	CONVERT, NEW, OLD, and USER).
	This manual contains information which was formerly in the following sections of the manual at
	revision C (sections and appendices in parentheses give the new location of the information): Sections
	1 through 3(remain the same), section 4 (sections 4 and 5), section 5 (sections 6, 7, 8, 9,
	10, and 14), section 6 (section 11), section 8 (section 15), section 9 (section 10, appendix A,
	and appendix F), section 10 (section 13), section 11 (section 10), appendix C (appendix C),
	appendix H (appendix D), appendix I (appendix G), and appendix J (appendix E). A new appendix
	(B) has been added which contains dayfile messages.

Publication No.
60407000

© 1973, 1974, 1975
by Control Data Corporation
Printed in the United States of America

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.

PREFACE

The KRONOS® Time-Sharing System was developed by Control Data Corporation to provide remote interactive job processing for CONTROL DATA® CYBER 70 Series Model 72, 73, and 74 Computer Systems and for CONTROL DATA® 6000 Series Computer Systems. This interactive job processing capability is provided in addition to the local and remote batch processing capabilities available under KRONOS.

This manual describes the external features of KRONOS 2.1.2 for the batch user. Information in this manual should be useful to those who use the programs and utilities supplied with the system and those who wish to write their own. The manual is contained in two volumes to separate information pertaining primarily to the applications programmer from that of interest to the systems programmer.

Volume 1 (publication no. 60407000) contains information for the applications programmer. This includes general information about files, job flow and execution, control card processing, and an extensive discussion on control cards.

Volume 2 (publication no. 60448200) contains information for those who write system or assembly language programs for use with KRONOS. It is primarily intended for the COMPASS programmer; however, several portions contain information for users of higher level languages. For reference, the table of contents of volume 2 follows the table of contents of this volume.

Throughout this manual cross references to the KRONOS 2.1 Reference Manual Volume 2 are of the form, "refer to section (or appendix) n, volume 2". If volume 2 is not stipulated, the reference is to this manual.

This manual does not contain a description of KRONOS system operation, detailed descriptions of the software product set available under KRONOS, or descriptions of the time-sharing commands.

The user is assumed to be familiar with CDC computer systems and with operating systems in general. For further information concerning CDC CYBER 70 and 6000 Series Computer Systems, the KRONOS time-sharing system, and the products supported by KRONOS, consult the following manuals.

Control Data Publication	Publication No.
CDC CYBER 70/Model 72 Computer System Reference Manual	60347000
CDC CYBER 70/Model 73 Computer System Reference Manual	60347200
CDC CYBER 70/Model 74 Computer System Reference Manual	60347400
CDC 6400/6500/6600 Computer Systems Reference Manual	60100000
KRONOS General Information Manual	60407100
KRONOS Instant Manual	60407200
KRONOS Installation Handbook	60407500
KRONOS Time-Sharing User's Reference Manual	60407600
KRONOS Operator's Guide	60407700

Control Data Publication	Publication No.
KRONOS Terminal User's Instant	60407800
TRANEX Reference Manual	60407900
TRANEX Operator's Guide Addendum	60408000
Export/Import Reference Manual	59150500
Text Editor Reference Manual	60408200
BASIC 2 Reference Manual	19980300
Time-Sharing FORTRAN Reference Manual	60408600
APL *CYBER Reference Manual	19980400
Modify Reference Manual	60281700
Modify Instant	60283000
Update Reference Manual	60342500
COMPASS 3 Reference Manual	60360900
FORTRAN Extended 4 Reference Manual	60305600
COBOL 4 Reference Manual	60384100
ALGOL 3 Reference Manual	60329000
ALGOL 4 Reference Manual	60384700
Sort/Merge 4 Reference Manual	60343900
Application Installation Handbook	76071100
PERT/Time 1 Reference Manual	60133600
SIMULA 1 Reference Manual	60234800
SIMSCRIPT 3 Reference Manual	60358500
SYMPL 1 Reference Manual	60328800
APEX III Reference Manual	76070000
GPSS V/6000 1 General Information Manual	84003900
LCGT/IGS 1 Reference Manual	17322800
Math Science Library 1 Reference Manual	60327500
8-Bit Subroutines 1 Reference Manual	60359400
Total Universal Reference Manual	76070300
Common Utilities Reference Manual	60493300
On-Line Maintenance Software Reference Manual	60436600
Record Manager Reference Manual	60307300
Loader Reference Manual	60344200

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

VOLUME 1

SECTION 1	SYSTEM DESCRIPTION	1-1-1
	Central Processor Unit	1-1-1
	Central Memory	1-1-1
	Central Memory Resident	1-1-2
	Extended Core Storage	1-1-3
	Peripheral Processor Units	1-1-3
	Peripheral Hardware	1-1-3
	System Software	1-1-4
	User Programs	1-1-4
SECTION 2	FILES	1-2-1
	Logical/Physical File Structure	1-2-1
	Mass Storage Device File Structure	1-2-2
	Magnetic Tape File Structure	1-2-2
	Punch File Structure	1-2-2
	File Types	1-2-3
	Queue Files	1-2-3
	Special Files	1-2-4
	Permanent Files	1-2-7
	Device Residence	1-2-8
	Accessing Files	1-2-8
	Reading Files	1-2-10
	Writing Files	1-2-11
	Libraries	1-2-14
SECTION 3	JOB FLOW AND EXECUTION	1-3-1
	Job Initiation	1-3-6
	Job Origin Types	1-3-6
	Job Names	1-3-6
	System Origin Type (SYOT) Job Name Format	1-3-6
	Batch Origin Type (BCOT) Job Name Format	1-3-7
	Time-Sharing and Export/Import (TXOT and EIOT) Job Name Format	1-3-7
	Validation	1-3-7
	Accounting	1-3-7
	Job Scheduling	1-3-8
	Job Control	1-3-8
	Field Length Control	1-3-8
	Input File Control	1-3-10
	Time Limit Control	1-3-10
	Rollout Control	1-3-10
	Error Control	1-3-11
	Job Completion	1-3-12
SECTION 4	CONTROL LANGUAGE	1-4-1
	Expressions	1-4-2
	Constants	1-4-2
	Arithmetic Operators	1-4-2

	Relational Operators	1-4-2
	Boolean Operators	1-4-3
	Evaluation of Expressions	1-4-3
	Symbolic Names	1-4-3
	Control Language Statements	1-4-4
	GOTO Statement	1-4-4
	CALL Statement	1-4-5
	DISPLAY Statement	1-4-6
	SET Statement	1-4-6
	IF Statement	1-4-7
	FILE Statement	1-4-8
	NUM Statement	1-4-10
	Procedure Files	1-4-11
	Time-Sharing Commands	1-4-13
	ASCII Statement	1-4-13
	CSET Statement	1-4-13
	PARITY Statement	1-4-13
SECTION 5	CONTROL CARD PROCESSING	1-5-1
	Control Card Format	1-5-1
	Job Card Format	1-5-4
	Control Card Processing Flow	1-5-6
	Exit Processing	1-5-8
SECTION 6	JOB CONTROL CONTROL CARD	1-6-1
	ACCOUNT Card	1-6-2
	CHARGE Card	1-6-2
	COMMENT Card	1-6-3
	CTIME Card	1-6-3
	DAYFILE Card	1-6-3
	ENQUIRE Card	1-6-4
	EXIT Card	1-6-5
	LDI Card	1-6-5
	LENGTH Card	1-6-5
	LIMITS Card	1-6-6
	MODE Card	1-6-8
	NOEXIT Card	1-6-9
	NORERUN Card	1-6-9
	OFFSW Card	1-6-10
	ONEXIT Card	1-6-10
	ONSW Card	1-6-10
	PASSWOR Card	1-6-11
	RERUN Card	1-6-11
	RESOURC Card	1-6-11
	RFL Card	1-6-14
	ROLLOUT Card	1-6-14
	RTIME Card	1-6-14
	SETCORE Card	1-6-14
	SETPR Card	1-6-15
	SETTL Card	1-6-15
	STIME Card	1-6-15
	SUBMIT Card	1-6-16
	SUI Card	1-6-20
	SUMMARY Card	1-6-20
	SWITCH Card	1-6-21
	USECPU Card	1-6-21
	USER Card	1-6-22

SECTION 7	FILE MANAGEMENT CONTROL CARDS	1-7-1
	ASSIGN Card	1-7-2
	BKSP Card	1-7-3
	CATALOG Card	1-7-4
	CLEAR Card	1-7-8
	COMMON Card	1-7-8
	CONVERT Card	1-7-8
	COPY Card	1-7-9
	COPYBF Card	1-7-10
	COPYBR Card	1-7-10
	COPYCF Card	1-7-11
	COPYCR Card	1-7-11
	COPYEI Card	1-7-12
	COPYSBF Card	1-7-12
	COPYX Card	1-7-13
	DISPOSE Card	1-7-14
	DOCMENT Card	1-7-15
	EVICT Card	1-7-16
	GTR Card	1-7-17
	LIBEDIT Card	1-7-18
	LIBGEN Card	1-7-19
	LIST80 Card	1-7-20
	LOCK Card	1-7-21
	L072 Card	1-7-21
	NEW Card	1-7-25
	OUT Card	1-7-25
	PACK Card	1-7-26
	PRIMARY Card	1-7-27
	RENAME Card	1-7-27
	REQUEST Card	1-7-28
	RESEQ Card	1-7-30
	RETURN Card	1-7-31
	REWIND Card	1-7-32
	SETID Card	1-7-32
	SKIPEI Card	1-7-33
	SKIPF Card	1-7-33
	SKIPFB Card	1-7-33
	SKIPR Card	1-7-34
	SORT Card	1-7-34
	STAGE Card	1-7-36
	TDUMP Card	1-7-37
	UNLOAD Card	1-7-38
	UNLOCK Card	1-7-38
	VERIFY Card	1-7-39
	VFYLIB Card	1-7-40
	WRITEF Card	1-7-40
	WRITER Card	1-7-40
SECTION 8	PERMANENT FILE CONTROL CARDS	1-8-1
	APPEND Card	1-8-5
	ATTACH Card	1-8-6
	CATLIST Card	1-8-7
	CHANGE Card	1-8-9
	DEFINE Card	1-8-10
	GET Card	1-8-11
	OLD Card	1-8-11
	PACKNAM Card	1-8-12
	PERMIT Card	1-8-13
	PURGALL Card	1-8-13
	PURGE Card	1-8-14

	REPLACE Card	1-8-14
	SAVE Card	1-8-15
SECTION 9	LOAD/DUMP CENTRAL MEMORY UTILITY CONTROL CARDS	1-9-1
	DMP Card	1-9-1
	DMD Card	1-9-2
	LBC Card	1-9-2
	LOC Card	1-9-3
	PBC Card	1-9-4
	RBR Card	1-9-4
	WBR Card	1-9-5
SECTION 10	TAPE MANAGEMENT	1-10-1
	ASSIGN Card	1-10-11
	BLANK Card	1-10-12
	LABEL Card	1-10-13
	LISTLB Card	1-10-14
	REQUEST Card	1-10-14
	VSN Card	1-10-15
	Magnetic Tape Formats	1-10-18
	Data Formats	1-10-18
	End-of-Tape/End-of-Reel Conditions	1-10-25
SECTION 11	PRODUCT SET CONTROL CARDS	1-11-1
	User Libraries	1-11-1
	Control Card Formats	1-11-2
	FTN Card	1-11-3
	COBOL Card	1-11-7
	ALGOL 3 Card	1-11-10
	ALGOL 4 Card	1-11-12
	SORTMRG Card	1-11-15
	PERT66 Card	1-11-16
	SIMULA Card	1-11-16
	SIMSCRIPT Card	1-11-18
	BASIC Card	1-11-18
SECTION 12	CHECKPOINT/RESTART	1-12-1
	CKP Card	1-12-1
	RESTART Card	1-12-2
SECTION 13	DEBUGGING AIDS	1-13-1
	Central Memory Dumps	1-13-1
	Generating Meaningful Dumps	1-13-2
	Reading CM Dumps	1-13-3
SECTION 14	PROGRAM LIBRARY AND SYSTEM UTILITY CONTROL CARDS	1-14-1
	Program Library Utility Control Cards	1-14-1
	MODIFY Card	1-14-1
	OPLEDIT Card	1-14-3
	UPDATE Card	1-14-7
	UPMOD Card	1-14-10
	System Utility Control Cards	1-14-10
	FAMILY Card	1-14-11
	KRONREF Card	1-14-12
	SYSEDIT Card	1-14-13

SECTION 15	LOADERS	1-15-1
	Link Relocatable Loader	1-15-2
	Memory Map	1-15-2
	Link Card	1-15-2
APPENDIX A	CHARACTER SETS	1-A-1
APPENDIX B	DAYFILE MESSAGES	1-B-1
APPENDIX C	LIBEDIT	1-C-1
APPENDIX D	JOB OUTPUT INFORMATION	1-D-1
APPENDIX E	PERMANENT FILE DEVICE STATISTICS	1-E-1
APPENDIX F	CARD FORMAT AND CONVERSION PROBLEMS	1-F-1
APPENDIX G	TAPE LABELS	1-G-1

FIGURES

1-1-1	Central Memory Layout	1-1-2
1-2-1	Sample Card File Structure	1-2-2
1-2-2	Sample Random Access File Format	1-2-9
1-2-3	Modified Sample Random Access File	1-2-13
1-3-1	Basic Job Deck	1-3-1
1-3-2	COMPASS Assemble and Execute Deck	1-3-2
1-3-3	COMPASS Assemble, Execute, and Punch Binary Deck	1-3-3
1-3-4	FORTRAN Compile and Execute Deck	1-3-4
1-3-5	FORTRAN Load and Run Deck	1-3-5
1-5-1	Control Card Processing Flow	1-5-7
1-7-1	Sample Page of Catalog of SYSTEM	1-7-6
1-13-1	Exchange Package	1-13-1
1-13-2	Main Program of Main Overlay (0, 0)	1-13-5
1-13-3	Function Subroutine of Main Overlay (0, 0)	1-13-6
1-13-4	Subroutine of Main Overlay (0, 0)	1-13-6
1-13-5	Main Program of Primary Overlay (1, 0)	1-13-7
1-13-6	Loader Map of Main Overlay (0, 0)	1-13-8
1-13-7	Loader Map of Primary Overlay (1, 0)	1-13-11
1-13-8	Program Output	1-13-11
1-13-9	Exchange Package Dump	1-13-12
1-13-10	Central Memory Dump	1-13-12
1-C-1	Adding to the Old Program Library	1-C-8
1-G-1	ANSI Labels: Single File, Single Volume	1-G-13
1-G-2	ANSI Labels: Single File, Multivolume	1-G-13
1-G-3	ANSI Labels: Multifile, Single Volume	1-G-14
1-G-4	ANSI Labels: Multifile, Multivolume	1-G-15
1-G-5	ANSI Labels: End-of-File, End-of-Volume Coincidence	1-G-16
1-G-6	ANSI Labels: End-of-File, End-of-Volume Coincidence	1-G-17
1-G-7	ANSI Labels: End-of-File, End-of-Volume Coincidence	1-G-18

TABLES

1-8-1	Response to Current Access Write/Read Access Desired	1-8-6
1-8-2	Response to Current Access Read/Read Access Desired	1-8-6
1-8-3	Response to Current Access Read/Write Access Desired	1-8-7

VOLUME 2

SECTION 1	INTRODUCTION	2-1-1
	Function Processors	2-1-1
	Macros and Common Decks	2-1-1
SECTION 2	PROGRAM/SYSTEM COMMUNICATION	2-2-1
	System Requests	2-2-1
	System Request Processing	2-2-1
	Issuing RA+1 Requests	2-2-3
	Macro Usage	2-2-5
	SYSCOM	2-2-5
	Common Deck Usage	2-2-8
SECTION 3	FILE CREATION AND INPUT/OUTPUT	2-3-1
	File Environment Table (FET)	2-3-1
	Circular Buffers	2-3-2
	FET Description	2-3-4
	FET Creation Macros	2-3-14
	FILEB	2-3-14
	FILEC	2-3-14
	RFILEB	2-3-14
	RFILEC	2-3-14
	CIO - Combined Input/Output	2-3-16
	CIO Function Processing	2-3-20
	Random Processing	2-3-20
	CIO Open and Close Functions	2-3-22
	OPEN	2-3-22
	CLOSE	2-3-26
	CLOSER	2-3-27
	CIO Read Functions	2-3-29
	RPHR (000)	2-3-29
	READ (010)	2-3-29
	READSKP (020)	2-3-30
	READCW (200)	2-3-31
	READLS (210)	2-3-33
	RPHRLS (230)	2-3-34
	READNS (250)	2-3-35
	READN (260)	2-3-35
	READEI (600)	2-3-36
	CIO Write Functions	2-3-37
	WPHR (004)	2-3-37
	WRITE (014)	2-3-37
	WRITER (024)	2-3-38
	WRITEF (034)	2-3-38
	WRITECW (204)	2-3-39
	REWRITE (214)	2-3-39
	REWRITER (224)	2-3-40
	REWRITEF (234)	2-3-40
	WRITEN (264)	2-3-41

	File Positioning Functions	2-3-42
	BKSP (040)	2-3-42
	BKSPRU (044)	2-3-43
	REWIND (050)	2-3-44
	UNLOAD (060)	2-3-46
	RETURN (070)	2-3-48
	POSMF (110)	2-3-49
	EVICT (114)	2-3-52
	SKIPF (240)	2-3-52
	SKIPFF (240)	2-3-53
	SKIPEI (240)	2-3-54
	SKIPB (640)	2-3-54
	SKIPFB (640)	2-3-55
	Data Transfer Macros	2-3-55
	READC	2-3-59
	WRITEC	2-3-60
	READH	2-3-60
	WRITEH	2-3-60
	READO	2-3-61
	WRITEO	2-3-61
	READS	2-3-62
	WRITES	2-3-62
	READW	2-3-63
	WRITEW	2-3-63
SECTION 4	LOCAL FILE MANAGER	2-4-1
	RENAME (000)	2-4-2
	ASSIGN (001)	2-4-3
	COMMON (002)	2-4-4
	RELEASE (004, 005, 006, 007, 016, 030)	2-4-5
	LOCK (010)	2-4-6
	UNLOCK (011)	2-4-7
	STATUS (012)	2-4-8
	STATUS (013)	2-4-8
	REQUEST (014)	2-4-10
	REQUEST (015)	2-4-11
	SETID (017)	2-4-12
	ASSIGN (020)	2-4-13
	ACCSF (021)	2-4-13
	ENCSF (022)	2-4-14
	PSCSF (023)	2-4-14
	LABEL (024)	2-4-15
	GETFNT (025)	2-4-20
	PRIMARY (031)	2-4-22
SECTION 5	PERMANENT FILE MANAGER	2-5-1
	Auxiliary Device Requests	2-5-4
	SAVE (001, CCSV)	2-5-5
	GET (002, CCGT)	2-5-6
	PURGE (003, CCPG)	2-5-7
	CATLIST (004, CCCT)	2-5-8
	PERMIT (005, CCPM)	2-5-12
	REPLACE (006, CCRP)	2-5-12
	APPEND (007, CCAP)	2-5-13
	DEFINE (010, CCDF)	2-5-15
	ATTACH (011, CCAT)	2-5-18
	CHANGE (012, CCCG)	2-5-20

SECTION 6	CONTROL POINT MANAGER	2-6-1
	SETQP (000)	2-6-1
	SETPR (001)	2-6-2
	MODE (002)	2-6-2
	SETTL (003)	2-6-2
	EREXIT (004)	2-6-3
	CONSOLE (005)	2-6-5
	ROLLOUT (006)	2-6-5
	ONSW (011)	2-6-7
	OFFSW (012)	2-6-7
	GETJN (013)	2-6-7
	GETQP (014)	2-6-8
	GETPR (015)	2-6-8
	GETEM (016)	2-6-9
	GETTL (017)	2-6-9
	SETUI (021)	2-6-10
	SETLC (022)	2-6-10
	SETRFL (023)	2-6-11
	GETJCR (024)	2-6-12
	SETJCR (025)	2-6-13
	SETSS (026)	2-6-13
	GETJO (027)	2-6-14
	GETJA (030)	2-6-14
	USECPU (031)	2-6-15
	USERNUM (032)	2-6-15
	GETFLC (033)	2-6-16
	PACKNAM (035)	2-6-16
	PACKNAM (036)	2-6-17
	GETSS (037)	2-6-17
	VERSION (044)	2-6-18
	GETLC (045)	2-6-18
	GETGLS (046)	2-6-18
	SETGLS (047)	2-6-19
SECTION 7	QUEUE FILE MANAGER	2-7-1
	RERUN (015)	2-7-2
	NORERUN (016)	2-7-3
	SUBMIT (017)	2-7-3
	Assign File to Queue Device (020)	2-7-4
SECTION 8	QUEUE DUMP/LOAD PROCESSOR	2-8-1
	Change File to Local (000)	2-8-1
	Release File to Queue (001)	2-8-3
SECTION 9	SYSTEM FILE MANAGER	2-9-1
	DAYFILE (001, 002, 003, 005)	2-9-2
	ESYF (004)	2-9-3
	RDVT (006)	2-9-4
SECTION 10	JOB CONTROL	2-10-1
	Translate Control Statement	2-10-1
	CONTROL (004)	2-10-2
	EXCST (005)	2-10-2
	Checkpoint/Restart	2-10-3
	CHECKPT	2-10-3

SECTION 11	SYSTEM/LOADER REQUESTS	2-11-1
	System Requests	2-11-1
	ABORT	2-11-1
	CLOCK	2-11-1
	DATE	2-11-2
	EDATE	2-11-2
	ENDRUN	2-11-3
	ETIME	2-11-4
	JDATE	2-11-4
	MEMORY	2-11-5
	MESSAGE	2-11-7
	MOVE	2-11-8
	PDATE	2-11-9
	RECALL	2-11-9
	RTIME	2-11-10
	STIME	2-11-11
	SUBR	2-11-11
	SYSTEM	2-11-11
	TIME	2-11-12
	Loader Requests	2-11-13
	EXU	2-11-13
	LDR	2-11-14
	Memory Allocation for Overlay Loaders	2-11-17
SECTION 12	PROGRAM WRITING TECHNIQUES	2-12-1
	Writing Programs under NOS	2-12-1
	Writing Interactive Programs	2-12-1
	Conversion Problems	2-12-1
	Default File Assignments and Special File Treatment	2-12-2
	Special Handling	2-12-2
	Other Special Handling	2-12-2
	Program Control of Terminal Activity	2-12-3
	Control Bytes	2-12-4
	DISTC Macro	2-12-5
	CSET Macro	2-12-8
	Parity Macro	2-12-8
	TLX Macro	2-12-9
	TSTATUS Macro	2-12-9
APPENDIX A	CPU COMMON DECKS	2-A-1
APPENDIX B	EXAMPLES OF RANDOM I/O	2-B-1
APPENDIX C	CODING SPECIFICATIONS	2-C-1
APPENDIX D	PROGRAM EXAMPLE	2-D-1
APPENDIX E	JOB COMMUNICATION AREA	2-E-1
APPENDIX F	SPECIAL ENTRY POINTS	2-F-1
APPENDIX G	BINARY FORMATS	2-G-1
APPENDIX H	COMPASS CONTROL STATEMENT	2-H-1

FIGURES

2-3-1	Circular Buffer	2-3-2
2-3-2	Write Operation	2-3-3
2-3-3	Read Operation	2-3-4
2-3-4	Standard FET for Mass Storage File	2-3-5
2-3-5	Standard FET for Magnetic Tape File	2-3-5
2-3-6	Data Transfer Buffer Arrangement	2-3-56
2-11-1	Absolute Loader Request Assignment	2-11-18
2-B-1	COMPASS Program to Create a Random File	2-B-2
2-B-2	Input File for Program Creating a Random File	2-B-5
2-B-3	Structure of the Random File Created	2-B-6
2-B-4	COMPASS Program Using READLS Macro to Retrieve a List of Records from a Random File	2-B-7
2-B-5	COMPASS Program to Replace Certain Records on a Random File	2-B-9
2-C-1	External Documentation of COPYB	2-C-2
2-C-2	Internal Documentation of COPYB	2-C-5
2-E-1	Job Communication Area	2-E-1
2-G-1	Modify Library File Format	2-G-3
2-G-2	Modification Table Format	2-G-4
2-G-3	Modify Text Format	2-G-4
2-G-4	Common Deck Modification Table Format	2-G-6
2-G-5	Library File Directory Table	2-G-6
2-G-6	Chippewa Record Format	2-G-7
2-G-7	User Library (ULIB) Format	2-G-8
2-G-8	ULIB Record Format	2-G-9
2-G-9	ULIB Deck Entry/Externals Format	2-G-10
2-G-10	Text Record Format	2-G-10

SYSTEM DESCRIPTION 1

The CDC CYBER 70/Models 72, 73, and 74 Computer System and 6000 Series Computer Systems consist of four logical hardware components. They are:

- Central processor unit
- Central memory
- Peripheral processor units
- Associated peripheral equipment

These hardware elements are controlled and coordinated by two basic levels of software, the system software and user programs. This section describes briefly these hardware and software elements and their relationship within the KRONOS Time-Sharing system.

CENTRAL PROCESSOR UNIT

The central processor unit (CPU) performs computational tasks but has no I/O capability. It communicates with the external world through central memory. Under KRONOS, the CPU is used to assemble, compile, and execute user programs and to perform several system functions and utilities.

The CDC CYBER 70 Series and 6000 Series Computer Systems provide two types of central processors. However, the programmer need be concerned only with the distinction between the two types when writing COMPASS programs. Certain instructions, if properly arranged, may be executed simultaneously by the CDC CYBER 70/Model 74 CPU and the 6600 CPU. For more information about CDC CYBER 70 and 6000 systems, refer to the hardware reference manuals listed in the preface.

CDC CYBER 70 Series computers are equipped with a central exchange jump/monitor exchange jump (CEJ/MEJ) feature. This feature enables the system to switch control between the system monitor and a user program. CEJ/MEJ is an option on 6000 Series computers. It should be used when available to improve job performance.

CENTRAL MEMORY

Under KRONOS, central memory (CM) is used for three basic purposes.

- To hold instructions to be executed by the CPU
- To hold data to be manipulated by the CPU
- To buffer data to and from peripheral processors

Several programs can reside in CM simultaneously in hardware-protected areas called control points. The fact that these control points are hardware-protected means that a program cannot reference an address outside its field length. KRONOS supports a maximum of 27_8 control points. The user need be concerned only with the memory

assigned to his own control point. The system assigns the CPU to the control points requiring CPU activity. Normally, the assignment of the CPU is switched rapidly between the control points to allow all programs in memory to execute. The exact amount of time allowed for each control point depends on system activity and system parameters. Thus, a job may take more real time to complete at one time than at another. The user has no control over this switching process.

The user program communicates with the system by placing requests in address 1 (RA+1) of the control point. RA is the reference address that specifies the beginning of the user's control point memory area.

When a user program completes, aborts, or is rolled out, the control point is released and made available to another program.

CENTRAL MEMORY RESIDENT

A portion of CM is reserved for system use. This area is, in effect, a control point with special privileges. This area is called central memory resident (CMR). It contains system tables and directories as well as the CPU portion of the system monitor (CPUMTR).

Figure 1-1-1 illustrates the layout of CM and shows the relationship between CMR and the user control points.

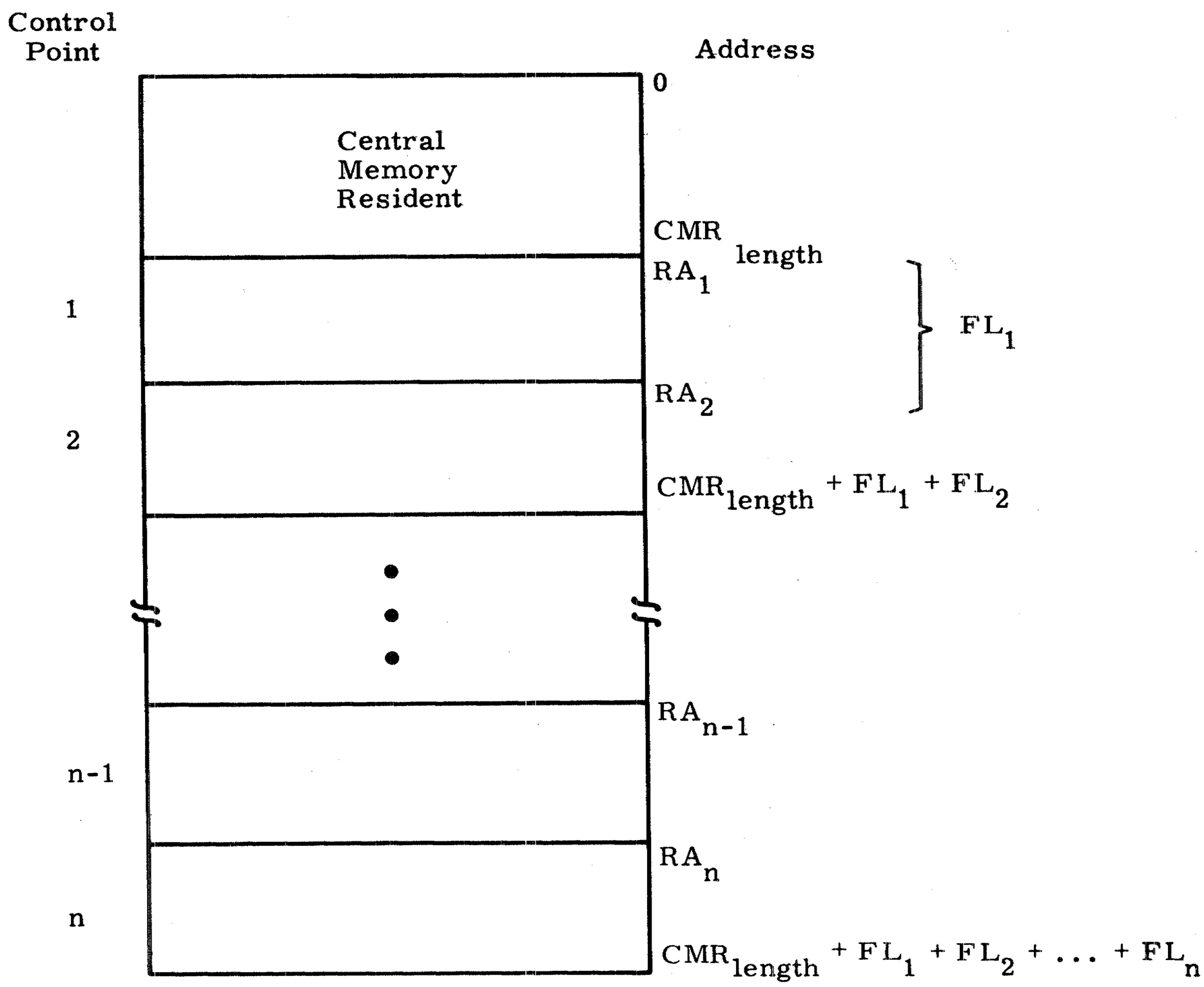


Figure 1-1-1. Central Memory Layout

EXTENDED CORE STORAGE

Extended core storage (ECS), a second, slower form of memory, is also available. KRONOS treats ECS as a mass storage device; it can be used:

- For storing frequently accessed data. Refer to the description of permanent file and equipment/file assignment requests for further information.
- As an alternate system device for storing copies of ABS, OVL, and PP type routines. Refer to the SYSEDIT control card for further information.

The FORTRAN and COMPASS cards for ECS data storage/retrieval are not supported by KRONOS.

PERIPHERAL PROCESSOR UNITS

The peripheral processor units (PPUs) are small processors that provide communication paths between the central processor and individual peripheral equipment. KRONOS supports the 10, 14, 17, and 20 PPU configurations of CDC CYBER 70/Model 72, 73, and 74 computers. The 7, 8, 9, 10, and 20 PPU configurations are supported for 6000 Series computers. A peripheral processor can:

- Read and write CM
- Read and write ECS indirectly via CM or directly via the distributive data path (DDP)
- Transmit data to and receive data from peripheral devices using the data channels

The peripheral processors also perform those system control functions that are better handled by a PPU than by the central processor.

For further information about PPUs, refer to the appropriate system hardware reference manual listed in the preface.

PERIPHERAL HARDWARE

The system peripheral hardware varies from installation to installation but usually includes card readers and punches, line printers, mass storage devices, and magnetic tape units. The following equipment is supported by KRONOS.

405 Card Reader

415 Card Punch

501, 505, 512, and 580 Line Printers

6603 Disk System

6638 Disk System

863 Drum Storage

854 Disk Storage Drive

814 Disk File

821 Data File

841 Multiple Disk Drive

844 Disk Storage Subsystem

Extended Core Storage

604, 607, 657, 659, 667, and 669 Magnetic Tape Units

6671 Multiplexers for communication with 200 User Terminals and 731-12/732-12 Remote Batch Terminals

6671 or 6676 Multiplexers for communication with interactive terminals

The user need be concerned with these devices only to the degree that they affect the format of data being transferred in the system.

SYSTEM SOFTWARE

The system consists of the group of CPU and PPU programs that control the flow of user programs and satisfy any special requests that these programs may make. These special requests include such functions as resource allocation requests and input/output requests.

USER PROGRAMS

A user program is a group of CPU instructions defined by a user to perform a certain task or calculate a specific result. A user program may be written in a language at any of three levels.

- Compiler languages provide the user with a language suited to his particular needs. The program cards are translated by the appropriate compiler (FORTRAN, COBOL, ALGOL, etc.) that generates assembler language or machine language instructions. Programs written in compiler languages are usually machine-independent.

- Assembler languages provide a one-to-one relationship between instructions and machine operation. Mnemonics are provided for each instruction. These languages are normally used by advanced programmers because they are machine-dependent. Most of the KRONOS system is written in COMPASS, the assembler language of the CDC CYBER 70 and 6000 Series computers.

- Hardware instructions are interpreted directly by the computer, and therefore, require no interpretation by a compiler or assembler. Each hardware instruction is a binary number. The programmer is rarely concerned with instructions written at this level. The exception is when program debugging requires that the user scan memory dumps.

FILES 2

A file is the largest collection of information addressable by name. It begins with a beginning-of-information (BOI), an indicator which precedes all data in the file. A file consists of one or more logical records of information. A logical record is a group of related words or characters, of fixed or variable length, which is independent of its physical environment.

The end of a logical record is the end-of-record (EOR). The end of a logical file is the end-of-file (EOF) or the end-of-information (EOI), or both. If both, the EOF precedes the EOI. An EOI is the last physical item of information on a file. Because of this EOF/EOI concept, a file may actually be a multifile file. For example:

(BOI)data....(EOR)....data....(EOR)(EOF)....data....(EOR)(EOF)(EOI)

A typical file is illustrated by the following example.

> One line of an invoice may form an item, a complete invoice may form a record, a set of such records may form a file, and the collection of invoice files may form a multifile file.

LOGICAL/PHYSICAL FILE STRUCTURE

The actual structure of the BOI, EOR, EOF, and EOI indicators depends on the device on which the information is stored.

The user defines the logical format of a mass storage or magnetic tape file when he issues control cards or language specifications to create the file. Once a file is created, it can be transferred from one storage medium to another without affecting its logical format.

To take advantage of the physical characteristics of the medium on which a file is to be stored, the system converts all user-defined logical file structures into a system-defined physical file structure. In general, for higher level language users, this conversion process and the resulting physical file format are transparent. All file-related control cards and language specifications transfer data or position a file according to its logical definition. COMPASS users, on the other hand, have the option of reading, writing, or positioning a file according to its logical or physical format.

The basis of all physical file structures is the physical record unit (PRU). The size of a PRU depends on the storage medium used.

MASS STORAGE DEVICE FILE STRUCTURE

All data stored on mass storage devices† is written in 64 CM word PRUs. A logical record consists of one or more of these PRUs. The last PRU of a mass storage logical record must be a short (less than 64 CM words) or zero-length PRU.

A BOI for a mass storage file is the disk address for the file listed in the file name table (FNT). An EOR is a PRU containing less than 64 words and having a link to the next PRU in the file. An EOF for a mass storage file is a zero-length PRU (that is, a PRU containing no data) with a special link to the next PRU in the file. An EOI is a zero-length PRU with no forward link. The absence of a link signifies the EOI.

MAGNETIC TAPE FILE STRUCTURE

The operating system uses standard 7- or 9-track, 1/2-inch magnetic tape. BOI on magnetic tape is the load point. The definition of PRUs and of the EOR, EOF, and EOI indicators varies according to the format in which the data was recorded. Any of the following formats can be specified: external (X), blocked (B), line image (E), internal (I), SCOPE internal (SI), SCOPE stranger tape (S), SCOPE long block stranger tape (L), and foreign (F). Refer to section 10 for a description of each of these formats.

PUNCH FILE STRUCTURE

Because the physical characteristics of cards define the data, cards do not have a PRU size as previously defined. Refer to appendix F for the conversion procedures used for the various types of punch cards. The logical format of the file is indicated as follows:

- The first card in the deck is the BOI
- A 7/8/9 punch in column 1 represents an EOR
- A 6/7/9 punch in column 1 represents an EOF
- A 6/7/8/9 punch in column 1 represents the EOI

Thus, a deck can consist of many files which can consist of many records, as illustrated in Figure 1-2-1.

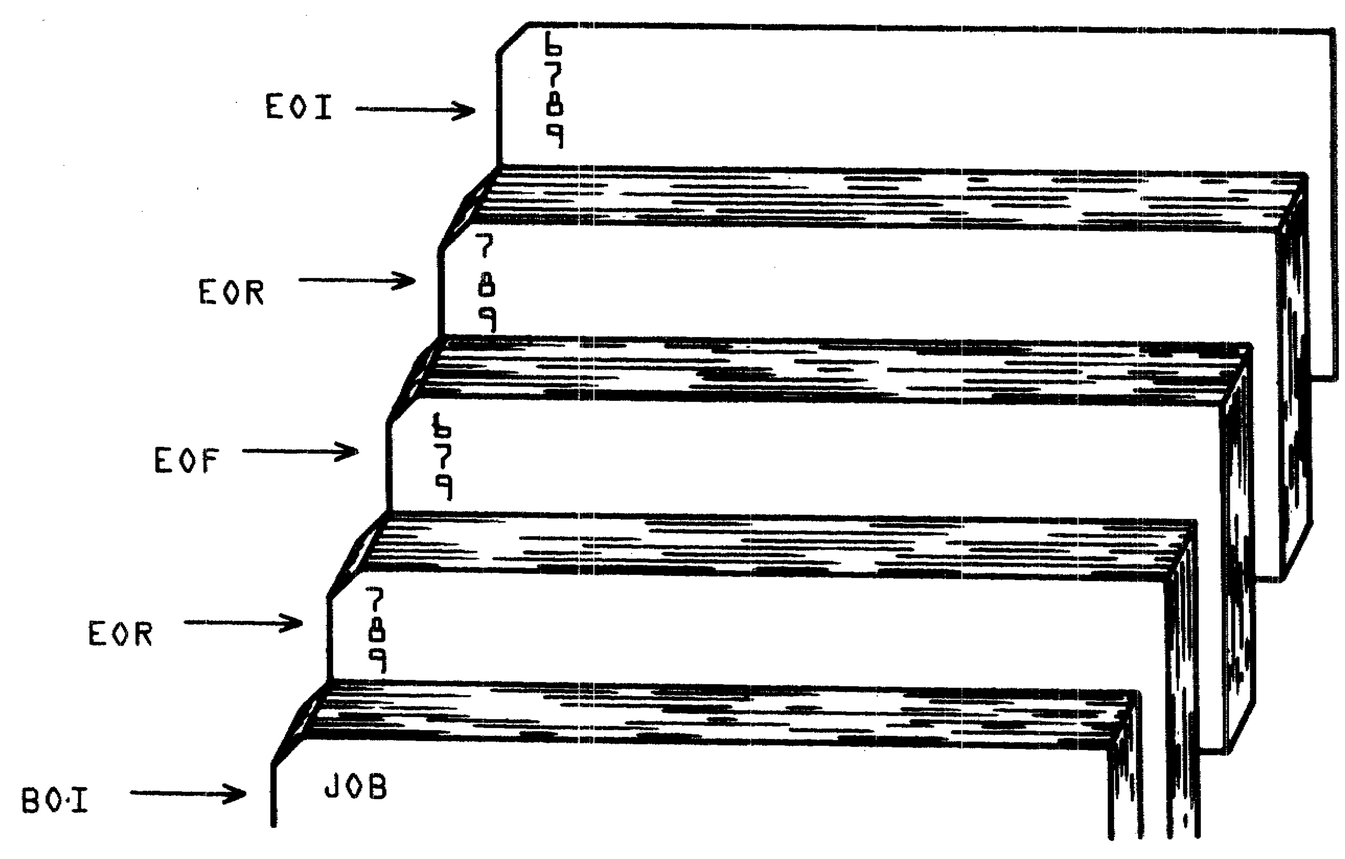


Figure 1-2-1. Sample Card File Structure

† ECS files are allocated in the same manner as all mass storage files.

FILE TYPES

Active files in the system are classified by their file type. Whenever a file is active, one or more entries are made in the file name table (FNT). The FNT entry and the file status table (FST) entry comprise a 2-word description of the file. These two entries contain the name of the file, the device on which the file resides, the file type, the current position, and the current status. All system tasks involving a file use this 2-word entry for control.

In each of the following descriptions, the file type and its mnemonic (such as INFT) which the system uses internally for file classification are listed.

QUEUE FILES

Five types of files are defined as queue files. They are categorized as such because of the kinds of information they contain and the manner in which the system processes them. Queue files always reside on mass storage. When a queue file is ready to be processed, the system or the user places it in a queue where it waits until the required system resource or peripheral equipment becomes available.

INPUT FILES (INFT)

Input files are the job files of the system. They contain all user-supplied control cards and program data. There are two ways a file can be placed in the input queue: directly by the sytem in initiating a local or remote batch job for processing and indirectly by a user job in submitting another job via a SUBMIT control card or an LDI control card.

When central memory space becomes available, either because a job has completed or because a job in the input queue has a higher priority than that of a job being processed, the input file is scheduled for processing (in other words, the job is assigned to a control point in central memory). Refer to section 3 for a description of the elements of jobs and the processes of job initiation and scheduling.

ROLLOUT FILES (ROFT)

At some stage in the processing of a job, the system or the user may determine that the job must be temporarily removed from central memory. When this occurs, the system writes all information concerning the job on a system-defined rollout file. The rollout file includes the contents of the job's central memory field length and the job-related system information from CMR. The file is read back into central memory when the job is again scheduled at a control point (refer to Rollout Control in section 3).

TIMED/EVENT ROLLOUT FILES (TEFT)

A timed/event rollout file is similar to an ROFT file in that it contains all the information concerning a job temporarily removed from central memory. A TEFT file, however, is rolled back into central memory only when a specified event has occurred (such as a file is no longer busy) or a specified time period has elapsed.

A job may be rolled out on a TEFT file as a result of system or user action. The system uses a timed/event file if a job issues certain requests for a file or device that cannot be immediately honored. The COMPASS programmer can use the ROLLOUT macro to roll out his job subject to specified time and/or event dependencies.

PRINT FILES (PRFT)

A print file contains data the user wishes to have printed during his job or upon job completion. The system-assigned name for print files is OUTPUT.† OUTPUT is placed in the print queue either by the system when the job completes or by the user via an OUT control card. The user can also include a DISPOSE control card to place a file in the print queue.

Once a file enters the print queue, it is processed by the local or remote batch printer processor. Then, when a printer becomes available, the PRFT file with the highest priority is printed.

Most system utility reports are written on OUTPUT unless the user specifies an alternate file. OUTPUT has no special internal format. Refer to appendix F for a description of conversion methods and printer control characters and to appendix D for a description of job output information.

PUNCH FILES (PHFT)

Punch files contain data that the user wishes to have punched on cards during his job or upon job completion. The system-assigned names for punch files are:

PUNCH	Contains Hollerith punch output
PUNCHB	Contains binary punch output
P8	Contains 80-column absolute binary punch output

These files are released to the punch queue when the job completes. In addition, the user can include an OUT or DISPOSE control card in the same manner as described for PRFT files to place a file in the punch queue.

Refer to appendix F for a description of the format of the PUNCH, PUNCHB, and P8 files.

SPECIAL FILES

Of the five special files, the first two described (local and direct access permanent files) are general purpose, and the remaining three (library, system, and primary terminal) are special purpose.

LOCAL FILES (LOFT)

All scratch and working files are designated as local files. The user can create a local file in three ways; he can:

1. Implicitly create a local file by making the first reference to it in one of the COPY control cards, any read or write language specification, or an OPEN

† For time-sharing jobs, the name OUTPUT has special meaning. Refer to section 12 volume 2, and to the Time-Sharing User's Reference Manual.

macro. Local files created in this manner always reside on mass storage.

2. Create a local file by preceding any COPY cards, read or write specifications, or OPEN macros with an explicit control card or macro file definition. The ASSIGN control card or the REQUEST control card or macro assigns a local file to mass storage or magnetic tape. The LABEL control card or macro assigns a local file to magnetic tape.

3. Use a GET control card or macro to generate a local mass storage copy of an existing indirect access permanent file. For a description of indirect access permanent files, refer to Permanent Files in this section.

Unless the user includes a control card or macro to change a local file to another type of file, it is released upon job completion.

DIRECT ACCESS PERMANENT FILES (PMFT)

A direct access permanent file is the type of permanent file that can be accessed directly rather than through the use of a working copy. The user creates a direct access file with the DEFINE control card or macro. Once the file is created, the originator or anyone else to whom the originator has given permission can assign the file to his job with an ATTACH control card or macro. The file remains in the system until the originator removes the file with a PURGALL control card, or the originator or any other user with the necessary permission removes the file with a PURGE control card or macro.

For further information about direct access permanent files and their relationship to indirect access permanent files, refer to Permanent Files in this section.

LIBRARY FILES (LIFT)

A library file is a read-only file that can be accessed by several users. A user must be validated to access/create library files. Note that this type of file should not be confused with system library programs or permanent file public (library) files.

A library file is created by performing the following steps.

1. Create a local file lfn.
2. Enter the following directives as control cards or macros.

```
LOCK(lfn)
COMMON(lfn)
```

If a user wishes to read this file and knows the file name, either the COMMON control card or ASSIGN macro is entered. When either of these functions is performed, an FNT entry representing this file as a library type file is created.

A library file cannot be removed from the system once it has been created except by a deadstart. Library files are not retained on initial (level 0) deadstart. They are retained on level 1 or 2 deadstart if a system checkpoint was done after their creation.

For a description of the relationship between LIFT files and other libraries and library files, refer to Libraries in this section.

SYSTEM FILES (SYFT)

The system uses SYFT files for retaining special system information. SYFT files always reside on mass storage. Although the COMPASS programmer who is validated to create system files can do so with an ESYF macro, only special system programs can access them. Once a system file is created, no user including the originator can remove it. System files are lost, however, at system deadstart unless the operator recovers them.

PRIMARY TERMINAL FILES (PTFT)

The primary file is the main working file for the user. Of several files which may be local to his job, the user may designate one file to be the primary file by using a NEW or PRIMARY card. (A copy of an indirect access file may be retrieved and made a primary file using the OLD card.) This becomes the default file if a file name is not specified. Only one primary file is available to the user at a time.

PERMANENT FILES

The user can create, retain, and access files which are available until he specifically decides to remove them from the system. These files are called permanent files. There are two types of permanent files.

- Direct access permanent files are accessed using normal I/O procedures, including random read and write requests. Direct access permanent files are allocated in large blocks;† thus, they are generally used as large data base files. Direct access files have a write interlock. This means that if one user has attached the file in write mode, it cannot be attached by another user. Likewise, if a user wishes to attach the file in write mode, he must wait until all current users have completed using the file. The user should also note that because data is written directly on the file rather than on a working file, care must be taken when modifying a direct access file.

  The maximum size of a direct access file is determined either by the DS validation parameter described in the LIMITS control card, section 6, or, if no DS restriction is imposed, by the device limitations described in appendix E.

- Indirect access permanent files are accessed by using a working copy of the file as a local file attached to the user's job. This working copy is obtained with the GET control card or macro. If the user wishes the working copy to remain permanent after the file has been altered, the SAVE or REPLACE functions must be issued. Indirect access files are allocated in blocks of 64 central memory words (640 characters). Because of this smaller block size and the convenience of a working copy, the indirect access file is generally the method used to create a small permanent file that does not require a write interlock.

  The maximum size of an indirect access file is determined either by the FS validation parameter described in LIMITS Control Card, section 6, or if no FS restriction is imposed, by the device limitations described in appendix E.

To access permanent files, the batch user must specify a user number by entering the USER control card. The user number is a 1- to 7-character value which represents a specific catalog in the permanent file system. Unless specified by an optional (alternate) user number, all permanent file requests are made to this catalog.

User numbers that contain asterisks represent users with automatic read-only permission to files in catalogs of other users. The user number must match the alternate user number in all characters not containing asterisks. For example, the user with the user number *AB*DE* can access the catalogs of the following users.

UABCDEF

UABDDEE

MABCDE1

MAB1DE3

---

† Refer to Permanent File Device Statistics, appendix E.

DEVICE RESIDENCE

For most file operations, the user need not be concerned about the specific device on which his file resides. However, under certain circumstances the user may wish to override the system default device residence for local or permanent files.

With the ASSIGN control card, any user who has the necessary validation can assign a local file to either a specific magnetic tape or mass storage device or to one of a type of magnetic tape or mass storage devices.

Every permanent file the user creates resides either in his family of permanent file devices or on an auxiliary device. Unless the user specifies otherwise, all permanent files are saved in his family.

A family consists of 1 through 63 mass storage devices. Within a family each user has a master device that contains his permanent file catalog, all indirect access files, and some or all of his direct access files.

Normally a system has only one family of permanent file devices. However, because families are interchangeable between KRONOS systems, several families may be active on one system. For example, consider an installation with two systems, A and B. System A provides backup service to system B. If system A failed, its family of permanent file devices could be introduced into system B without interrupting current operations on system B.

The user identifies his family by supplying a 1- to 7-character family name. The family name is included on the USER card in batch jobs and is entered during login in time-sharing jobs. If only one family is active or if another family has been introduced into the user's normal system, he may but need not supply his family name. When the family name is omitted, the system uses the system default family name. If the user's family has been introduced into another system, he must supply his family name.

If the user chooses to save his files on family devices, he has the option of either using the system default device type or of specifying another type of permanent file device.

An auxiliary device is a supplement to the mass storage provided by family devices. It is identified by a 1- to 7-character pack name. An auxiliary device is not necessarily a disk pack that can be physically removed as the pack name implies. Rather, an auxiliary device can be any mass storage device supported by the system and defined as such by the installation. Each auxiliary device is a self-contained permanent file device; all direct and indirect access files represented by the catalogs on the device reside on the device. Auxiliary devices may be defined as public or private. Anyone permitted to use auxiliary devices who supplies the appropriate pack name can create, replace, and access files on a public device. Only one user, the owner, can create and replace files on a private auxiliary device, but others may access those files as permitted by the owner.

ACCESSING FILES

The two methods used to access files attached to a job are sequential and random access. Any file can be accessed sequentially; however, only mass storage files can be accessed randomly.

To read a file randomly, the system reads a portion of the file without reading all information in the file, from the current position to the desired position. Any mass storage file can be read randomly if the user knows which relative PRU (that is, which PRU in relation to the BOI) he wishes to read. The desired PRU can be read by placing the PRU number in the file's communication area (FET) and making the proper I/O requests (refer to section 3, volume 2).

Several methods of random processing exist. The specific method depends on the language being used; however, in all cases, the following points apply.

- Most random I/O operations require a directory or index that contains the relative PRUs of records in the file.
- An EOR or EOF I/O operation transfers one PRU for the EOR or EOF.
- When randomly rewriting data within a file, the user must take care to ensure that data following the area he wishes to write is not destroyed.

Figure 1-2-2 illustrates a typical example of the structure of a random access file.

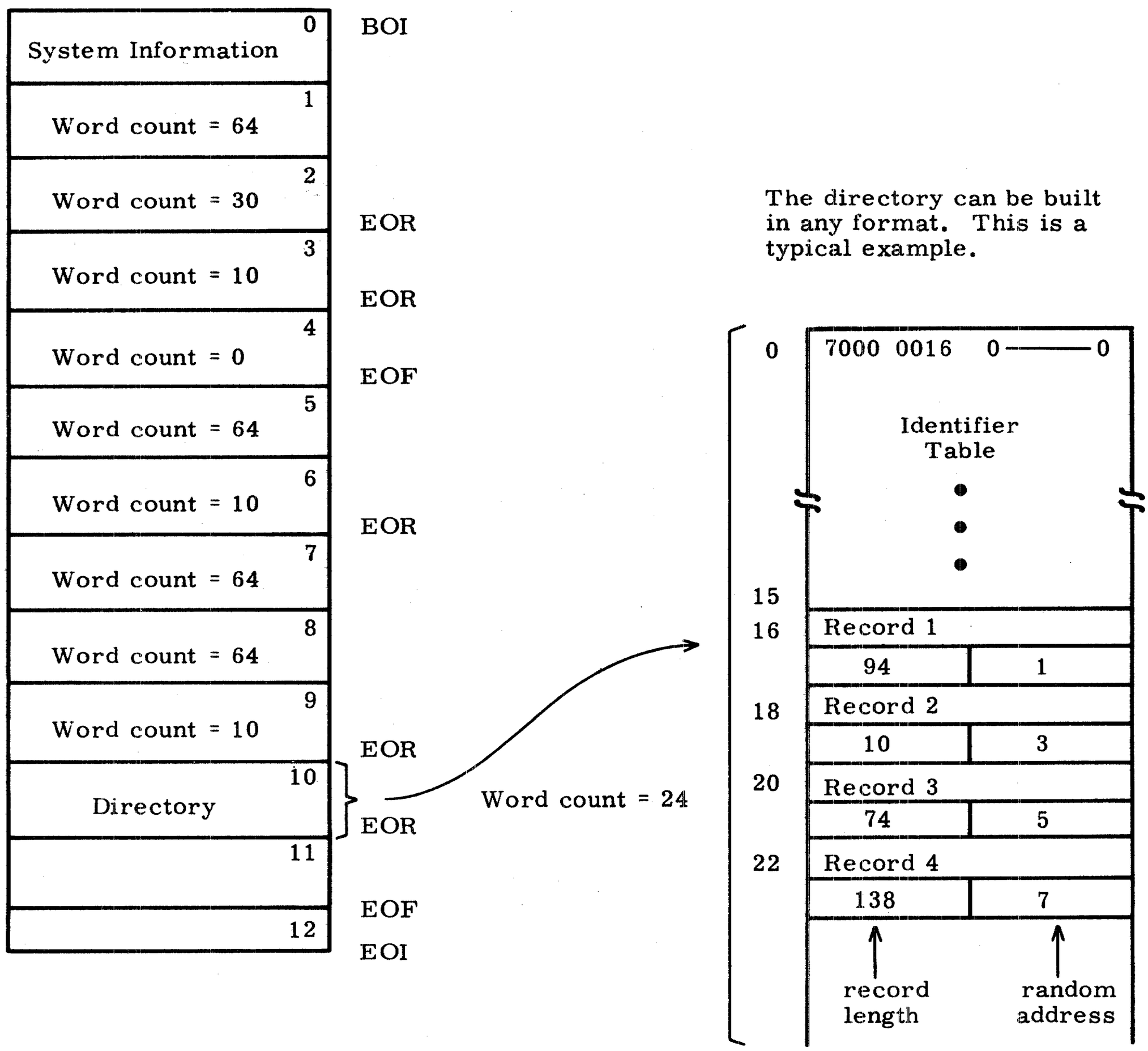


Figure 1-2-2. Sample Random Access File Format

Each directory entry contains the record name, the first PRU of the record (random address), and the record length.

READING FILES

To read record 3 sequentially, the program rewinds the file to BOI and reads the file and counts the number of EORs. System utilities and macros can be used to skip the records; however, the primary consideration is that the data must be read to determine where record 3 begins. Once this is determined, record 3 can be read.

If a directory exists for this file, the only requirement is that the random address of record 3 be obtained from the directory and placed in the FET. The proper random read requests can then be issued. To perform this random read on record 3, the following steps are required.

- Skip to the EOI. This is done by the system without reading the entire file.
- Backspace two logical records (one record for the EOF and one for the directory). The system must read both records to perform this operation.
- Read the directory to obtain the random address to be placed in the FET.

NOTE

The EOF may or may not be used at the end of this file. The language and methods used to build the directory determine whether an EOF is used.

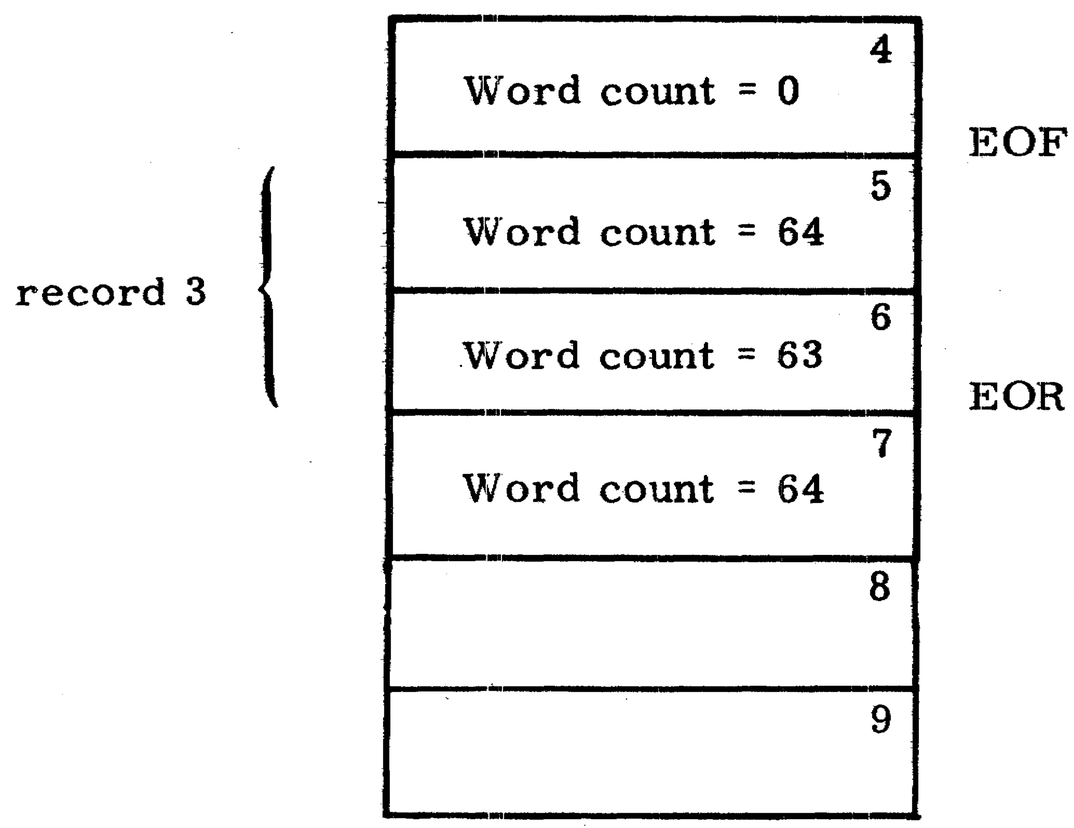
In summary, to access record 3 sequentially, four PRUs must be read. To access the record randomly, only three PRUs are read: two PRUs to position for the directory and one PRU to read the directory.

For additional random accesses to any record in the file, it is not necessary to access the directory again if it remains in central memory.

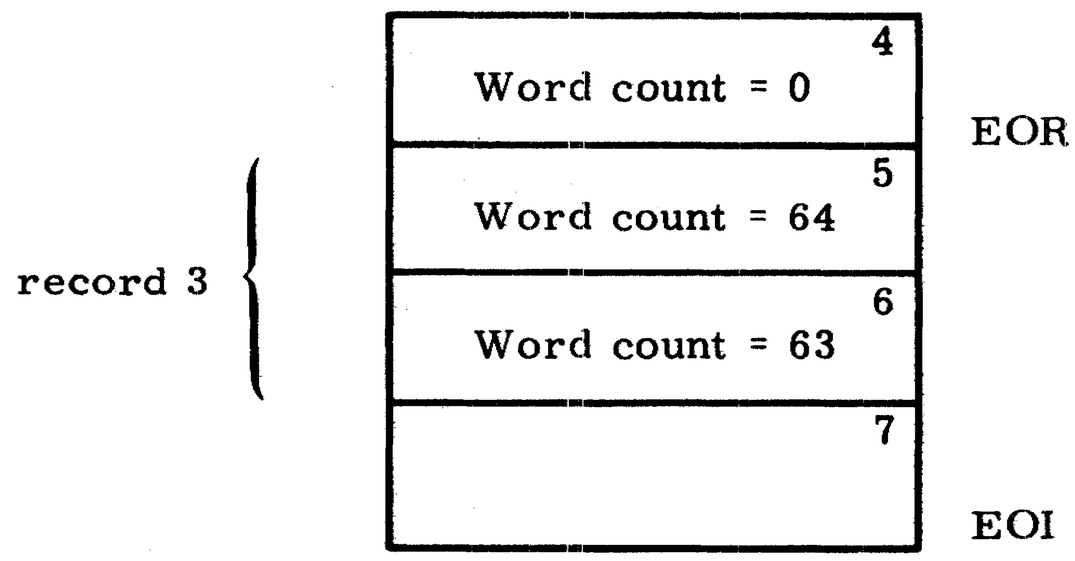
The directory can be placed anywhere in the file. The only requirement is that those users who wish to access the file randomly know where to position the file in order to read the directory. However, the directory usually precedes the EOF/EOI.

WRITING FILES

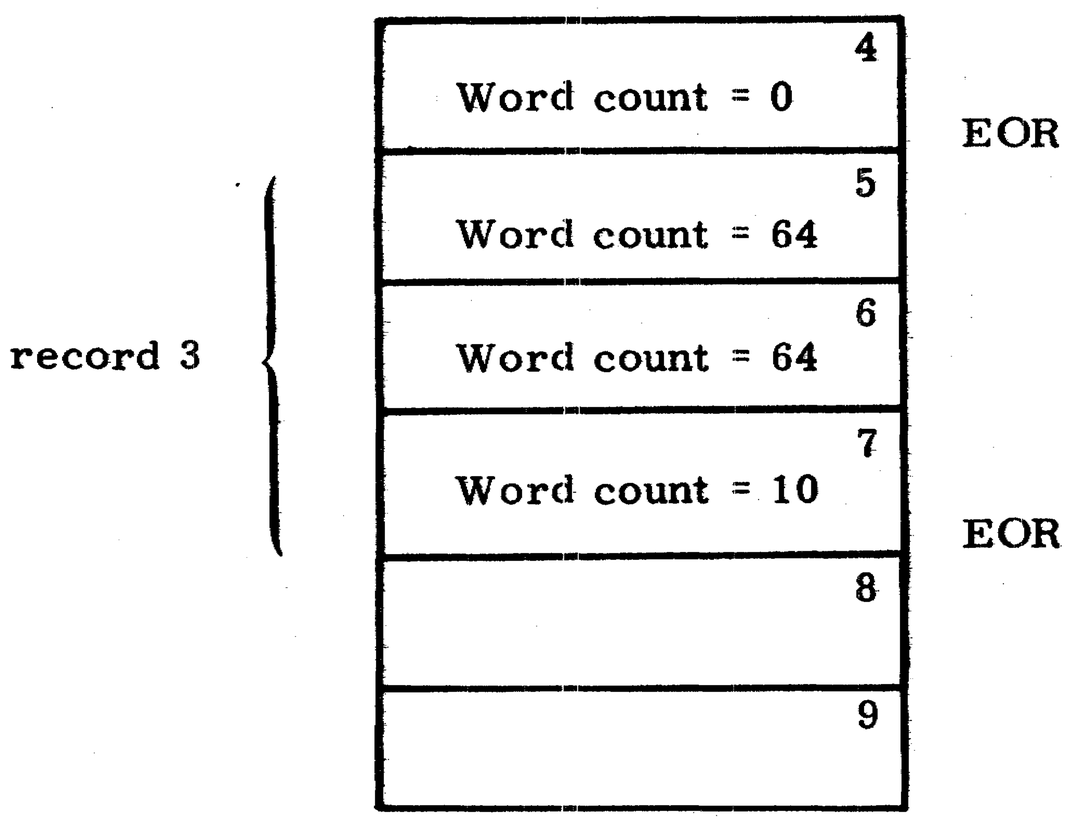
After reading and modifying record 3 of the sample file, the user may wish to rewrite the record in the file. If the modifications have not changed the number of PRUs required, a write operation can be used to replace the existing record with the modified record. This write operation must be issued as a random I/O operation. (Refer to section 3, volume 2 for a complete description of the method.) However, if the modifications have changed the number of PRUs required, data following the record being written is lost. For example, the size of record 3 in the sample file is 74 words or two PRUs. A maximum of 53 words can be added to the record without requiring an additional PRU and destroying data. If a random write request that adds 53 words to record 3 is issued, the file has the following format.



This operation is called a rewrite in place. If the write is issued as a nonrandom write operation, the file has the following format.



All data following the inserted data is destroyed. If the word count for record 3 is increased to 138, the file has the following format.



PRU 7 is destroyed by the write operation. To properly rewrite record 3 without destroying the contents of PRU 7, the user should issue a write request at the end of the file and alter the directory to reflect the change. Figure 1-2-3 illustrates the updated file containing the new directory and the 138-word modified record 3 written at the end of the file.

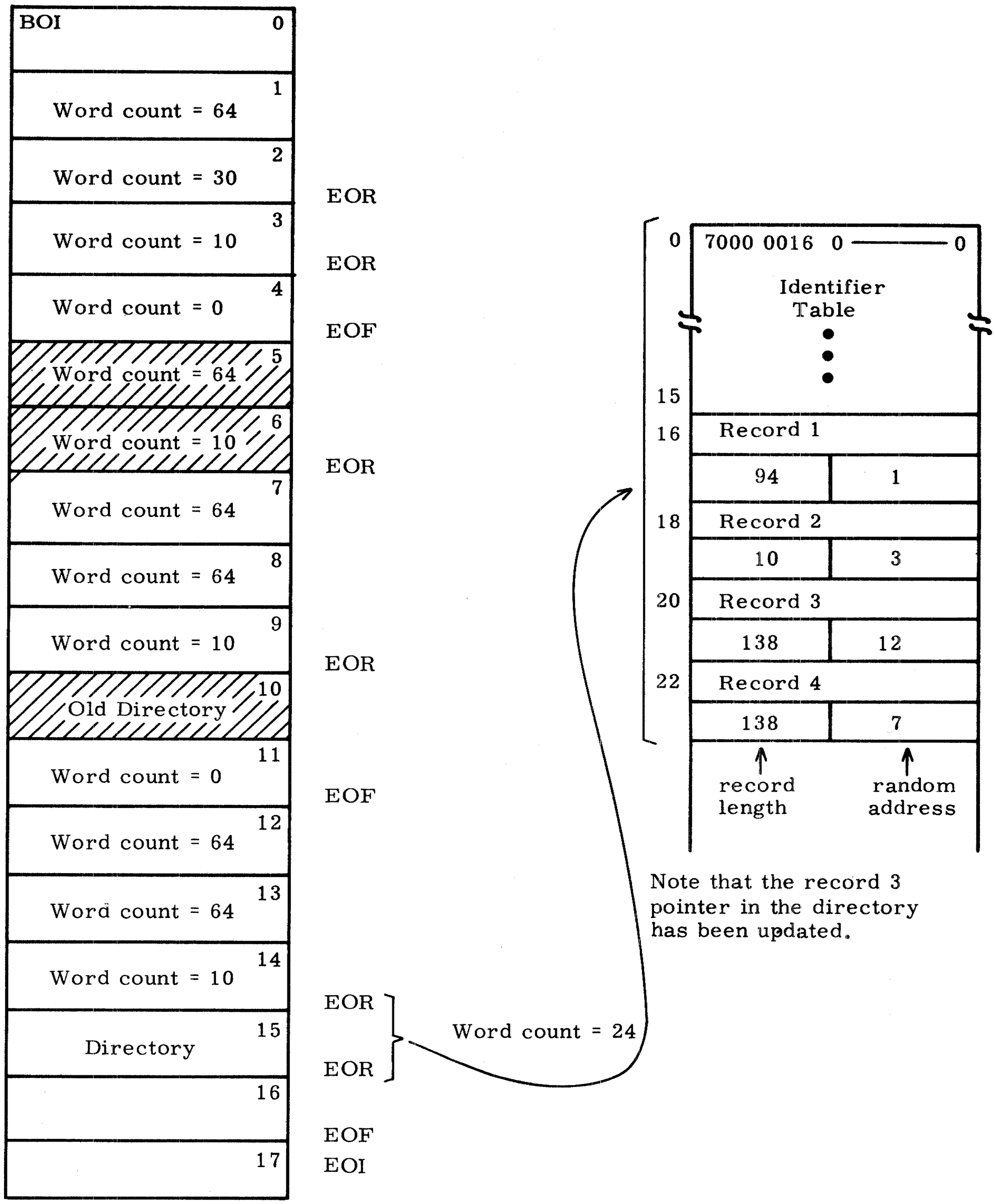


Figure 1-2-3. Modified Sample Random Access File

Appendix B, volume 2 contains examples of COMPASS programs that create, read, and write a random file.

LIBRARIES

The term library can be used in five ways in KRONOS. The following paragraphs define the various types of libraries and the methods, if any, by which the user accesses them.

- System library. The system library consists of the assembled routines that comprise the operating system and its associated product set. System routines may reside in central memory, mass storage, or ECS. The user accesses the system library indirectly when a system routine is executed in response to a control card or macro call. A complete copy of the system library is saved on a read-only file named SYSTEM. Refer to the CATALOG control card in section 7 for a partial list of the system library routines.

- Program library. A program library is a group of source deck images saved on a program library file in compressed format. There are two system-defined program libraries: OPL and OLDPL. OPL contains operating system routines saved and maintained in Modify format via the MODIFY control card. OLDPL contains product set routines saved and maintained in Update format via the UPDATE control card. In addition, the programmer can use a MODIFY or UPDATE control card to create and edit his own program library.

- User library. Before a user's compiled program can be executed, all external references must be satisfied. The loader satisfies externals by searching user libraries. A user library is a group of compiled or assembled object time routines saved on a user library file. There are three types of user library files: user-generated, product set, and system.

  User-generated libraries are created with the LIBGEN control card and can be specified on LIBRARY and/or LDSET control cards. Refer to the Loader Reference Manual for further information. Product set libraries reside as ULIB type records on the system library. They are listed in section 11. If some externals remain unsatisfied after searching these libraries, the loader searches the system default user library SYSLIB, which also resides as a ULIB record on the system library.

- Library files. Library files are read-only files. They are described in Library Files (LIFT) in this section.

- User number LIBRARY. An installation can save under the user number LIBRARY permanent mass storage files containing programs or text of general interest (such as applications programs and games) to time-sharing users. Refer to the Time-Sharing User's Reference Manual listed in the preface for further information.

A job consists of a file of card images grouped into several records. The first logical record contains the control cards that specify the job processing requirements. Each job begins with a job card and ends with an EOI card. All other control cards directly follow the job card. The end of the control cards is marked by an EOR, EOF, or an EOI card. Figure 1-3-1 illustrates a basic job deck.

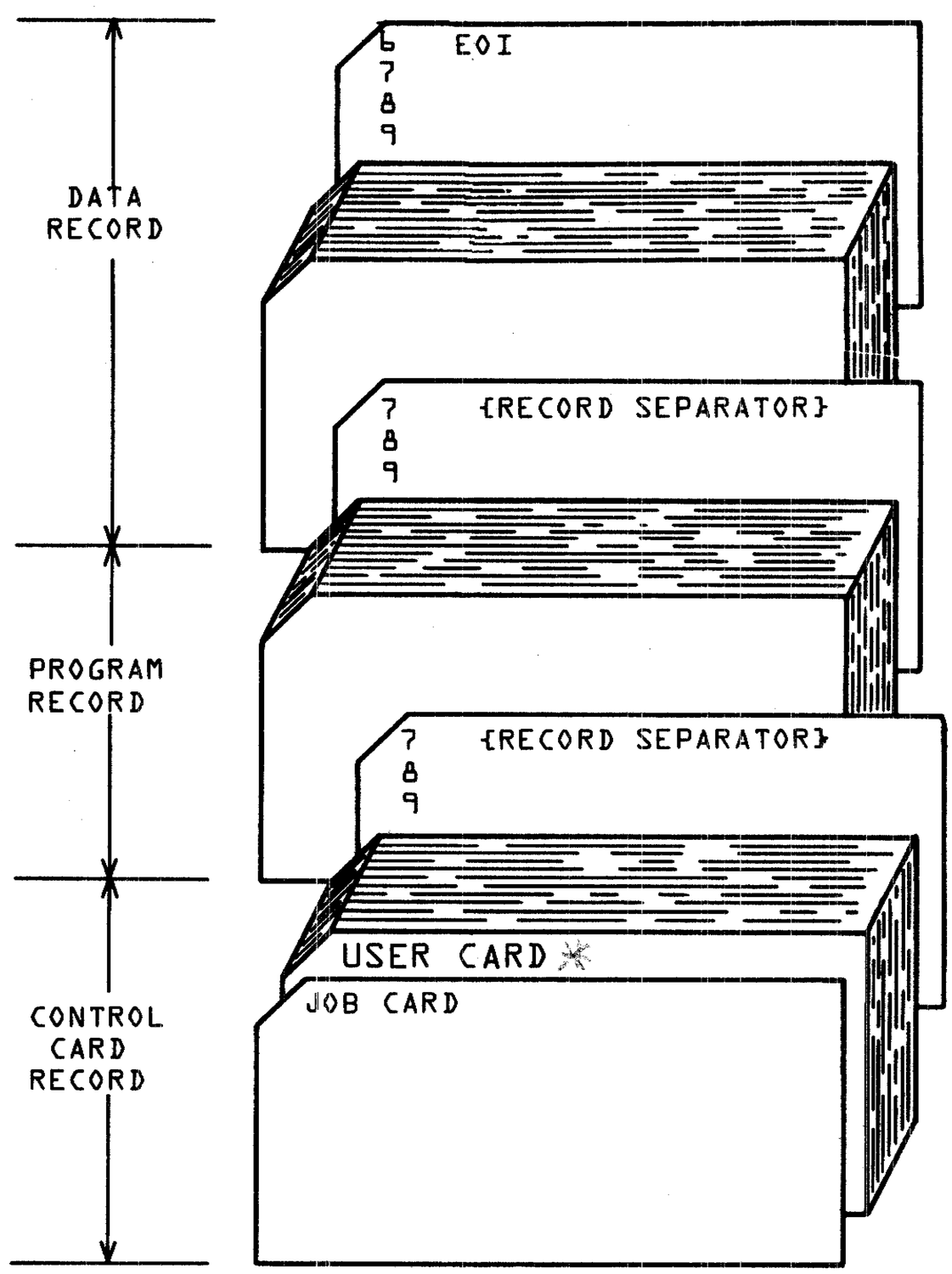


Figure 1-3-1. Basic Job Deck

Figure 1-3-2 illustrates a COMPASS source deck that produces the object code and a listing and executes the binary file using the input data supplied.

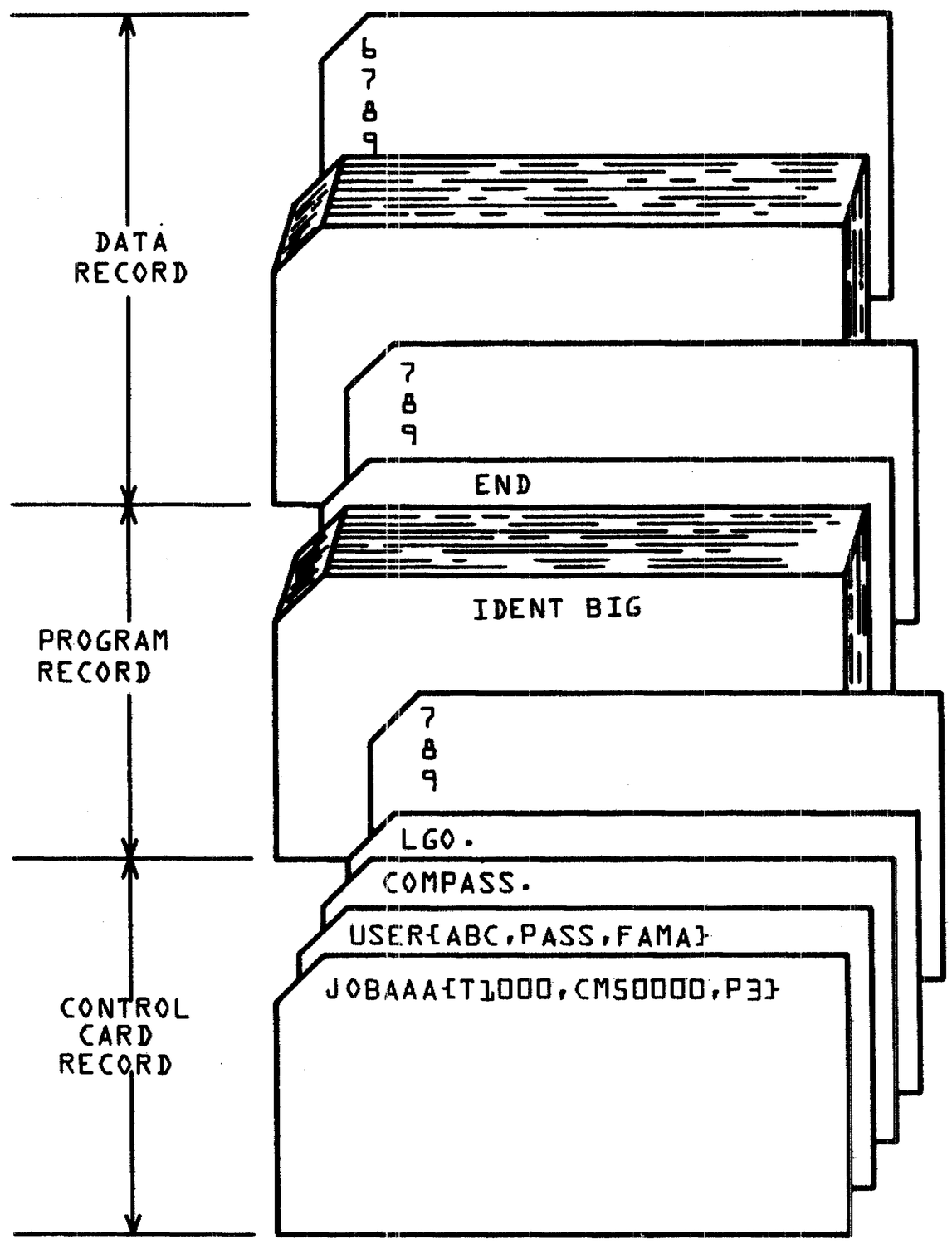


Figure 1-3-2. COMPASS Assemble and Execute Deck

Figure 1-3-3 illustrates a COMPASS deck that assembles the program, produces binary punched files of each subprogram, and executes the object code of the first program record.

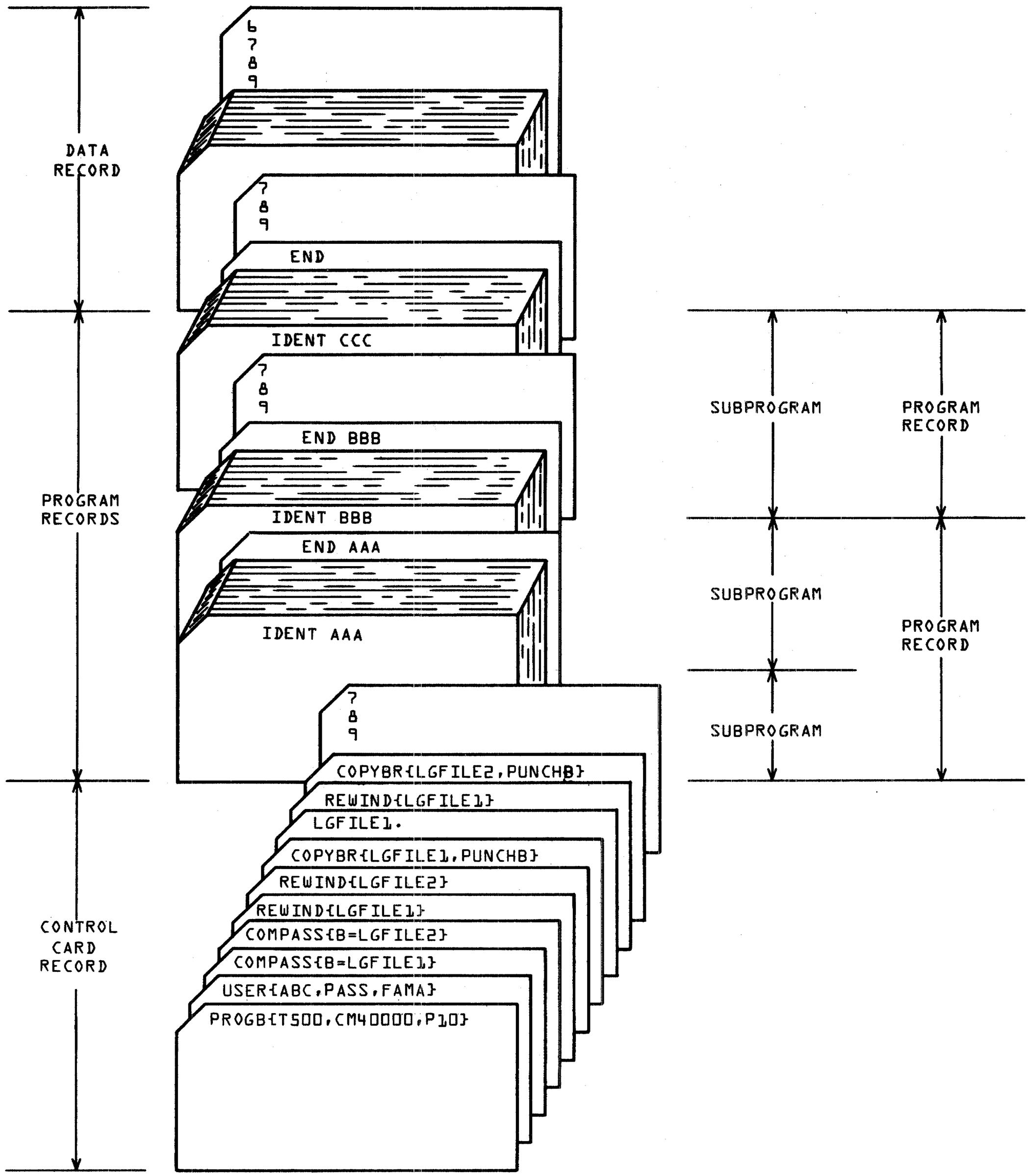


Figure 1-3-3. COMPASS Assemble, Execute, and Punch Binary Deck

Figures 1-3-4 and 1-3-5 illustrate examples of FORTRAN source decks used for computation and user output.

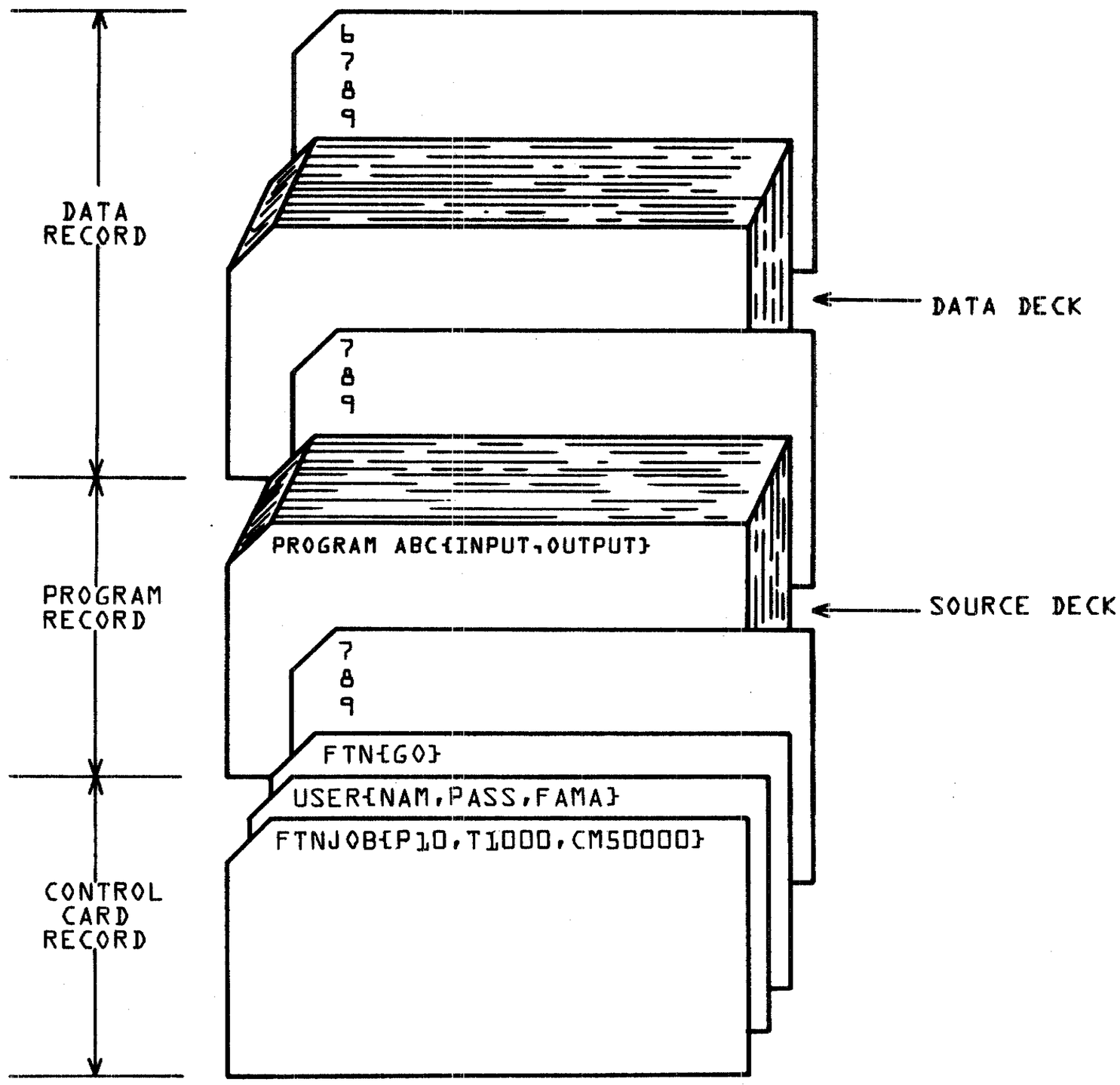


Figure 1-3-4. FORTRAN Compile and Execute Deck

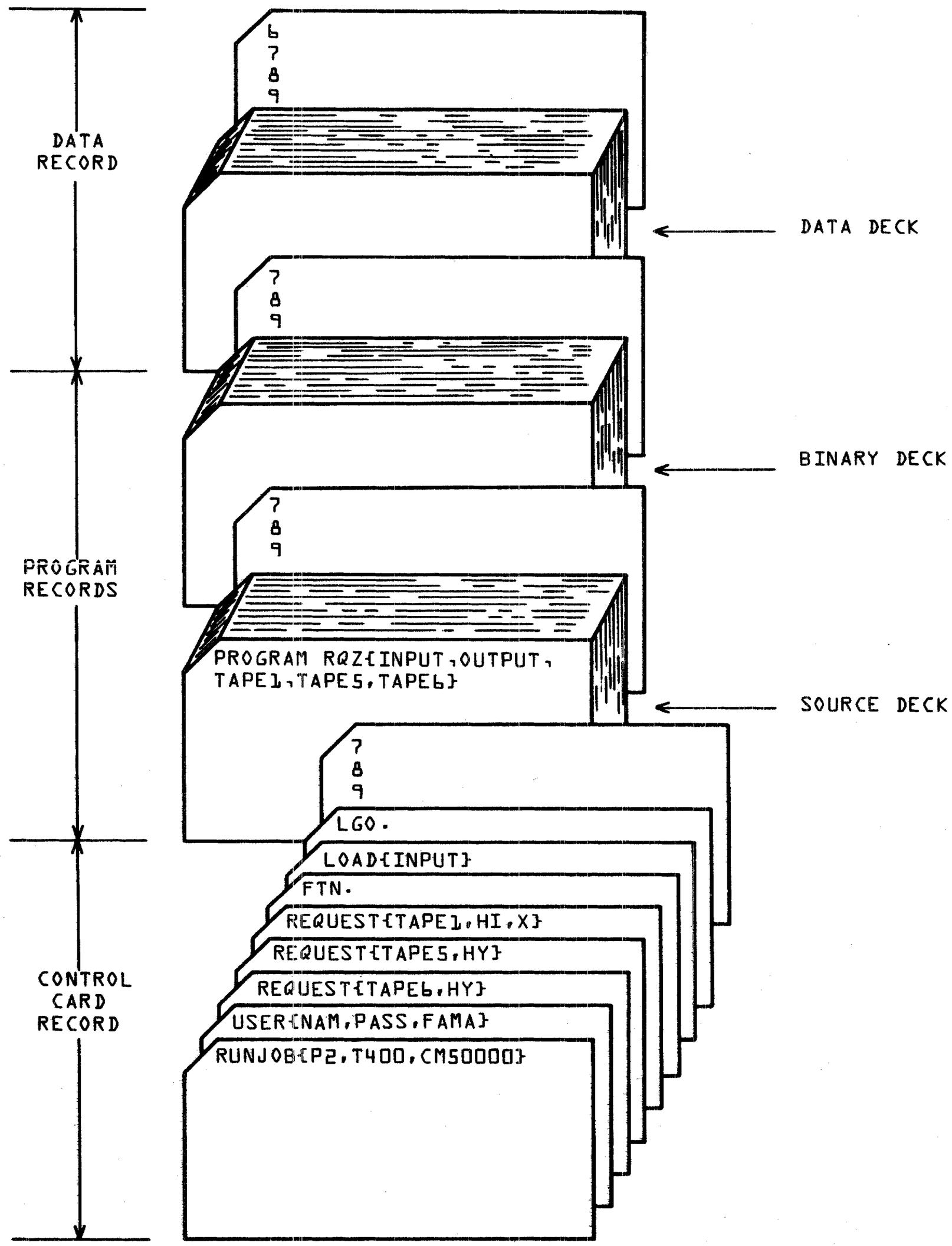


Figure 1-3-5. FORTRAN Load and Run Deck

JOB INITIATION

When a job enters the system, the system determines the job origin type. The job origin type identifies the means by which the job is entered into the system. It is also used to identify the job while it remains in the system. The job origin type is used by the system to control job activity and to aid in directing the job through the system. It also determines the way the job exits from the system.

Jobs are initiated by:

- Reading a card deck in through a card reader, either from a local or a remote batch reader.
- Using the available load utilities (refer to the LDI control card, section 6).
- Using the SUBMIT control card from a job already in the system.

The first two methods of initiation set the job origin type to indicate the method by which the jobs are entered. When using the SUBMIT control card, a parameter is entered with the command specifying the origin type. (Refer to the description of the SUBMIT control card, section 6.)

JOB ORIGIN TYPES

If the job originates from the system console, the job is assigned system origin type (SYOT). If the job is a time-sharing job and enters through the time-sharing executive, it is given time-sharing origin type (TXOT). If the job enters the system through the local card reader, it is a batch origin type (BCOT) job. A job coming into the system from remote batch is entered into the system by the Export/Import package and is assigned Export/Import origin type (EIOT).

If a user is validated to do so, he can submit jobs to the system by using the SUBMIT control card. He can also specify origin types which are different from his own.

JOB NAMES

After entering the system, the job is assigned a unique job name to prevent job name duplication within the system. This job name is a combination of parameters that describe the job; the first seven characters are the system-assigned job name; the eighth character indicates the job origin type. This job name precedes all messages issued to the system dayfile for that job. These messages include normal operating messages, error messages, and accounting information issued by the system.

SYSTEM ORIGIN TYPE (SYOT) JOB NAME FORMAT

The first four characters of a system job name are obtained from the job name entered or are zero filled if fewer than four characters are entered. The next three characters are a unique system sequence number in the range from AAA to 999. The eighth character is an S. For example, if the job entered is DIS, a possible job name is DIS0AABS.

BATCH ORIGIN TYPE (BCOT) JOB NAME FORMAT

The first four characters of a batch origin job name are generated from the user index associated with the user number supplied on the USER control card. These four characters are unique to the user. The next three characters are the job sequence number. The eighth character of a batch origin job name is B.

TIME-SHARING AND EXPORT/IMPORT (TXOT AND EIOT) JOB NAME FORMAT

The first four characters of these job names are generated from the user index associated with the user number supplied by the user when logging into the system. The next three characters represent the number of the terminal on which the user is logged in for TXOT or the system sequence number for EIOT. The eighth character is T for time-sharing origin jobs and E for remote batch jobs.

All jobs entered via the SUBMIT control card derive the first four characters of their job names from the job's current user index in the same manner as EIOT and TXOT jobs. The last three digits are the system sequence number with the eighth character being either E or B, as described previously, depending on the parameter supplied with the SUBMIT card.

VALIDATION

The USER card follows the job card and is used to validate the user as a legal user (refer to USER card, section 6). If the user is validated, a set of control values is set in the control point area; these values are used by the system to control all system requests. In most cases, if the user is not permitted to perform specific functions (such as access nonallocatable devices), his job is aborted and the message

ILLEGAL USER ACCESS.

is issued when the illegal function is attempted.

To determine the extent of his validation, the user can issue the LIMITS command and receive a listing of his current validation control values. Refer to the LIMITS control card in section 6 for an explanation of these values. For further information or to change his validation, the user should contact installation personnel.

Each user number has a unique user index associated with it. Once a user number is validated, the user index is set in the control point area. The system uses this index to determine on which permanent file device (and where) the user's permanent files reside. (Refer to part IV of the Installation Handbook for an explanation of the user index.)

ACCOUNTING

The unit of accounting for the system is the system resource unit (SRU). The SRU is a composite value of central processor time, I/O activity, and memory usage. SRU operations are initiated at the beginning of a job and reinitiated whenever another CHARGE control card is encountered. SRU information includes:

- Central processor time
- Mass storage activity
- Magnetic tape activity
- Permanent file activity

- SRU value
- Application account charges†

This information is written to the user's dayfile at the end of the job or whenever a CHARGE card is processed. The user may request SRU information to be written to his output file at any time during his job by issuing the SUMMARY control card. The format of SRU information written in the dayfile is given under Job Completion in this section.

JOB SCHEDULING

When a job enters the system, it is placed in the input queue on mass storage, where it waits for the required system resources to become available. The job is assigned an input queue priority depending on its origin. The system priorities are system-defined and can be altered only by the system operator. The job queue priority is advanced as the job waits in the queue. The priority ages to a system-defined limit. The job scheduler periodically scans the queues and active jobs to determine whether action is necessary to ensure that the highest priority jobs are being serviced. This action may include rolling out low priority jobs or rolling in higher priority jobs. The job scheduler is also activated to analyze the system status whenever the status of the system changes (for example, when the field length of a job is released, a job enters a queue, or a job completes). Because of this automatic scheduling and analysis of system status changes, a user can increase system performance by releasing memory when all of the assigned memory is not required.

Once a job is brought to a control point, normal control card processing begins. The general flow of the control card processing is illustrated in Figure 1-5-1.

JOB CONTROL

While a job is at the control point, the system exercises several controls over the job.

FIELD LENGTH CONTROL

When a job is active, the system controls the field length in either of two ways. The action the system takes depends upon the requirements of the system routine called in response to a control card request. Some system routines indicate their field length requirements through RFL= or MFL= entry points. If the job issues a request to a routine that does not have a predefined field length, the field length remains at or is restored to the user-specified value. However, if the system does encounter an RFL= or MFL= entry point, it sets the field length according to the rules governing RFL= or MFL= processing (refer to appendix F, volume 2). The system then restores the field length to the user-specified value before processing the next control card, unless that card calls a routine which also has an RFL= or MFL= entry point.

†Not currently supported by the system but reserved for future use.

The running field length for the job is set by the job card which must be validated as previously described. The running field length can be set to any value less than the maximum for which the user is validated by using one of the following methods.

- When the RFL control card is encountered, the running field length is adjusted accordingly.
- If the executing program calls the control point manager (CPM) with the MEMORY macro or a macro call to RFL, the running field length is altered.
- If the program issues the SETRFL macro, the running field length is set to the field length specified.

Examples of field length control:

Control Card	Field Length	Comments
JOB(CM50000, T1000)	50000	Sets running field length to 50000.
USER(USERABC, 1234567, FAM1)	50000	Running field length remains at 50000.
MODIFY(L=0, Z)/*EDIT, FORT	36600	MODIFY increments in 2000 word blocks to the table size required to complete the MODIFY operation.
FTN(I)	50000	Running field length returns to 50000.
LGO.	15000	The loader automatically reduces the job field length after performing a relocatable load. (RFL= is not present in LGO.) The loader requires a field length of 30200 to load itself. It increments in 4000 word blocks until the required table space is available.
FTN(I=TAPE1)	50000	The system restores the job to the running field length because no RFL= is specified in FTN.
REWIND(TAPE1,COMPILE)	1100	The system sets the field length because REWIND utility has RFL= set. (FILES is the utility package.)
SAVE(LGO=BIN)	1000	The SAVE utility also has RFL= set. (PFILES is the utility package.)
RFL(30000)	400	The field length is set to 400 because the utility that sets running field length (CONTROL) has RFL=specified. This control card changes the running field length from 50000 to 30000.
LIBEDIT(P=0, N, V)	30000	Because LIBEDIT has no RFL= specified, the system restores the field length to the running field before processing LIBEDIT.

INPUT FILE CONTROL

All user jobs, when initiated, have a file named INPUT. This file contains the control cards and other input records required for job execution. INPUT is a locked file. As a result, the user may read from it and reposition it, but the system does not allow him to write on it. If for some special reason the user needs to write on INPUT, he should first issue a RETURN(INPUT) control card (refer to section 7). This card changes the name of the file from INPUT to INPUT* and leaves it attached to the user's job. The change of name on RETURN applies only if the input file is of type INFT.

TIME LIMIT CONTROL

The system sets a time limit for a job unless the job card specifies a time limit. This time is the amount of central processor time that the job is allowed. The maximum time allowable on the job card is 77770_8 seconds. Any job in the system with a time limit of 77771_8 through 77777_8 seconds has an infinite amount of central processing time at its disposal. If the user wants to increase a job's time limit, the SETTL control card or macro is used. The user cannot, however, increase the limit beyond that for which he is validated.

While a job is using the central processor, the time of usage is accumulated and checked against the time limit. If the job is not a time-sharing (TXOT) job, the job in execution is aborted when the time limit is reached. Time-sharing origin jobs are rolled out, after which the user can reset the time limit and resume execution from the point where the time limit was exceeded. Refer to the Time-Sharing User's Reference Manual for a more detailed description.

ROLLOUT CONTROL

Each executing program is allowed to reside in CM for a certain amount of time before relinquishing its space to another program. When this CM time slice is exceeded, the program may be rolled out. This means that the contents of the job field length, the job control area, and the control registers (exchange package) are written to mass storage. The program remains on mass storage until it is rolled back into memory. Execution resumes from the point where rollout occurred. The amount of time the job is allowed to occupy CM is called the central memory time slice. The central memory time slice is a system parameter that can be changed only by the system operator. The time slices vary for each origin type. Whether a job is rolled out when its time slice expires depends on several factors.

- Whether there are jobs waiting in the input and rollout queues
- Whether the jobs that are waiting have a lower priority
- Whether jobs that are waiting require more field length than would be available if all jobs of lower priority were rolled out

When a job is rolled out, it is assigned a queue priority. The priority assigned is a system parameter and can be changed only by the system operator. The queue priorities can vary for each origin type. The queue priority is aged (incremented) while the job is in the rollout queue. Normally, all other factors being equal, the job with the highest queue priority is selected to be rolled in.

ERROR CONTROL

The exit mode feature allows the programmer to select conditions that permit the system to discontinue normal processing when errors occur. The error conditions and assoicated condition codes that can occur are:

- Address is out of range (01) — One of the following conditions has occurred.
  - The program attempted to reference CM memory or ECS outside the established limits.
  - The program is attempting to branch to an address outside the user's field length.
- Operand is out of range (02) — Floating-point arithmetic unit received an infinite operand.
- Indefinite operand (04) — Floating-point arithmetic unit attempted to use an indefinite operand.

The user can select any combination of these conditions with the MODE control card (refer to section 5). If one of these errors occurs and the proper mode for that error is selected, the system notes the error by setting the appropriate error flag and exiting from normal processing. The following dayfile error message occurs defining the error exit conditions:

CPU ERROR EXIT xx AT yyyyyy.

This message identifies the error condition by the condition code xx (as listed above) that was detected at location yyyyyy. Regardless of exit mode selection, the program interrupt is unconditional if the error is an illegal instruction or address range error on RNI or branch. If the exit mode is not selected, the central processor stops or proceeds depending on the situation. For a detailed explanation, refer to the 6000 Series Computer Systems Reference Manual or the CDC CYBER 70/Model 72, 73, or 74 Computer System Reference Manuals.

When activity at a control point ceases, the system determines the reason. If an error flag is set, the error is noted and execution is resumed at the error exit address if one was specified.

Once control is transferred, the error flag is cleared. If the error occurs because the central processor time limit is exceeded, the job is given another 10_8 seconds to complete processing. If the error is caused by a central processor abort the address at which the error occurred is specified and normal error processing continues.

When control is transferred from an executing program because of an error, the system determines whether or not to continue with control card processing, perform error processing, or terminate the job.

The system first searches for an EXIT control card. If an EXIT card is found, error processing begins with the card following EXIT. If, prior to the detection of the error, the system encountered a NOEXIT card, no search is made for an EXIT card and processing continues with the next control card. If no EXIT or NOEXIT card was encountered, the system terminates the job.

JOB COMPLETION

When there is no more activity at a control point, no outstanding central processor requests, and no control cards to process, the job is completed in the following manner.

1. All CM assigned to the job is returned to the system.
2. All equipment assigned to the job is returned to the system.
3. All library files attached to the job are returned; other jobs can then access them.
4. All scratch (local) file space used by the job is released.
5. All direct access permanent files attached to the job are returned; the status information for these files is updated.
6. The following information is issued to the user's dayfile and the account dayfile (the associated account dayfile message formats follow each item).
   - Application charge activity in units
     yy.mm.dd. hh.mm.ss. jobname. UEAD, xxxxxx.xxx UNTS.
   - Accumulated central processor time in seconds
     yy.mm.dd. hh.mm.ss. jobname. UECP, xxxxxx.xxx SECS.
   - Mass storage activity in kilo-units
     yy.mm.dd. hh.mm.ss. jobname. UEMS, xxxxxx.xxx KUNS.
   - Magnetic tape activity in kilo-units
     yy.mm.dd. hh.mm.ss. jobname. UEMT, xxxxxx.xxx KUNS.
   - Permanent file activity in kilo-units
     yy.mm.dd. hh.mm.ss. jobname. UEPF, xxxxxx.xxx KUNS.
   - SRU value in units for total job usage including CPU time, I/O activity, and memory usage
     yy.mm.dd. hh.mm.ss. jobname. AESR, xxxxxx.xxx UNTS.
   - Cards read in kilo-cards
     yy.mm.dd. hh.mm.ss. jobname. UCCR, es, xxxxxx.xxx KCDS.
   - Lines printed in kilo-lines
     yy.mm.dd. hh.mm.ss. jobname. UCLP, es, xxxxxx.xxx KLNS.
   - Cards punched in kilo-cards
     yy.mm.dd. hh.mm.ss. jobname. UCPC, es, xxxxxx.xxx KCDS.
7. Control point dayfile is copied to the end of the print file.
8. All output files are released to the output queue.
9. The control point area is cleared for the next job.

CONTROL LANGUAGE 4

The operating system control language allows the programmer to transfer control and to perform arithmetic and test functions within the control card record. Control language consists of statements very similar to FORTRAN statements. These statements are normally composed of a command (as listed below), parameters, symbolic names, and expressions. The following are legal commands.

GOTO	SET	NUM
CALL	IF	
DISPLAY	FILE	

An important feature of control language is the capability to create procedure files. A procedure file is a group of system control cards and/or control statements which can be called much like a subroutine for insertion anywhere within the control card record. It is activated either by the CALL statement or the name of the procedure file. Because control cards or control language or both are allowed in a procedure file, the user is given a much wider range of control for manipulating his files.

The following sections describe the various components and commands of the system control language.

EXPRESSIONS

The expressions allowed are similar to FORTRAN expressions and may contain constants, arithmetic operators, relational operators, Boolean operators, and functions.

CONSTANTS

Numeric constants are assumed to be decimal. If a constant has a post radix of D, it is decimal. If it has a post radix of B, it is octal.

ARITHMETIC OPERATORS

Arithmetic operations are performed in one's complement with 48-bit evaluations. The arithmetic operators processed are:

+	Addition
-	Subtraction
*	Multiplication
/	Division
**, ↑	Exponentiation
Leading -	Negation
Leading +	Ignored

RELATIONAL OPERATORS

Relational operations produce the value 1 if the relation is true and a value of 0 if the relation is false. The relational operators are (either form may be used):

=	.EQ.	Equal to
≠	.NE.	Not equal to
<	.LT.	Less than
>	.GT.	Greater than
≤	.LE.	Less than or equal to
≥	.GE.	Greater than or equal to

BOOLEAN OPERATORS

The Boolean operators are (either form may be used):

≡	.EQV.	Equivalence
∨	.OR.	Inclusive OR
∧	.AND.	AND
↓	.EOR.	Exclusive OR
¬	.NOT.	Complement

EVALUATION OF EXPRESSIONS

The order of evaluation of expressions is:

1. Exponentiation
2. Multiplication, division
3. Addition, subtraction, negation
4. Relations
5. Complement
6. AND
7. Inclusive OR
8. Exclusive OR, equivalence

Nesting of expressions to any depth is allowed within a statement.

SYMBOLIC NAMES

The symbolic names used to reference values pertaining to the job process consist of those with arithmetic values:

ARE	Arithmetic error
BCO	Local batch origin
CPE	CPU abort
EF	Previous error flag
EIO	Remote batch (Export/Import) origin
EM	Current exit mode
FL	Job field length
FLE	File limit error
MNE	Monitor call error

ODE	Operator drop
OT	Job origin type
PPE	PPU abort
PSE	Program stop error
R1	Contents of control register 1
R2	Contents of control register 2
R3	Contents of control register 3
SS	Job subsystem
SYO	System origin
TKE	Track limit error
TLE	Time limit error
TXO	Time-sharing origin

those with Boolean values:

SWn	Setting (1=On, 0=Off) of sense switch n ($1 \leq n \leq 6$)
TRUE	True value
T	True value
FALSE	False value
F	False value

and those using the subsystem (ss) name. The associated name must be one of the following:

NULL
BASIC
FTNTS
EXECUTE
BATCH
ACCESS†
TRANACT†

CONTROL LANGUAGE STATEMENTS

Statements are described in the following paragraphs. Separators and terminators must be used as shown in the statement formats.

GOTO STATEMENT

The GOTO statement transfers control to another location within the control card file.

The statement format is:

GOTO,stmt.

stmt	Name of any control card or a digit (0 through 9) followed by a maximum of six alphanumeric characters, terminated by a period.

†Special validation is necessary to access and use ACCESS and TRANACT. Refer to the LIMITS card, section 6.

Example 1	Example 2
.	.
GOTO, 1WX2.	REQUEST(TAPE1)
.	.
.	.
.	.
.	GOTO, REQUEST.
1WX2, REQUEST(TAPE1)	.
.	REQUEST(TAPE2)
.	.

When stmt appears more than once in the control statement file, the stmt to be executed is the first occurrence of stmt from the beginning of the control statement file. Hence, in both of the previous examples, the REQUEST (TAPE1) card is processed after the GOTO statement.

CALL STATEMENT

The CALL statement allows the user to insert a file consisting of a group of control cards (procedure file) at the specified position in the control card stream. This file is merged, as specified on the CALL card, with the current control card record into a third record. This third record becomes the current control card record. The remainder of the input file is then copied to the new control card record. If the C option is exercised, the current control card record is not used. Only the source file is used to generate a new control card record. All options are order-independent.

The statement format is:

CALL(lfn,C,S=ccc,RENAME(oldnam$_1$=newnam$_1$,oldnam$_2$=newnam$_2$,...,oldnam$_n$=newnam$_n$)

or

CALL(lfn,C,S=ccc(oldnam$_1$=newnam$_1$,oldnam$_2$=newnam$_2$,...,oldnam$_n$=newnam$_n$)

lfn — Procedure file name (refer to the description of procedure files in this section for further information). The system obtains lfn by:

1. Searching for a local file, lfn
2. Searching the system library for lfn
3. Attempting to retrieve a working copy of an indirect access file

C — Clears current statement file before entering statements from the called file. This replaces the entire control card record with lfn.

S=ccc	Sets next control statement to be processed to statement ccc. If S is not specified, the first statement in lfn is processed.
RENAME	Each occurrence of $oldnam_i$ is replaced with $newnam_i$ before the statement is entered into the statement file. As shown by the optional format, the word RENAME does not have to appear.
$oldnam_i$	Old name; name of a file or statement label used in the specified procedure file
$newnam_i$	New name; name to replace $oldnam_i$

DISPLAY STATEMENT

The DISPLAY statement determines the current subsystem or evaluates an expression and displays the result in the dayfile. Numeric results are displayed in both octal and decimal formats. Expression evaluation is significant only to 23 bits. Therefore, the octal representation of a negative number may be incorrect.

The statement format is:

```
DISPLAY(SS)
    or
DISPLAY(expression)
    expression          Any legal expression
```

Example:

```
DISPLAY(SS)
```

If the BASIC subsystem is currently in use, the preceding statement inserts the following message into the dayfile.

```
BASIC
```

Example:

```
DISPLAY((R1+R3) * R2)
```

If R1=5, R2=8, and R3=3, this statement inserts the following data in the dayfile.

```
64        100B
```

(Both decimal and octal values are displayed.)

SET STATEMENT

The SET statement allows the user to specify a subsystem or to set software registers to control the flow of a job. These registers are useful when designing a multipurpose procedure file. They also can be used to select a particular option in the procedure file. These software-defined registers are kept in the job control area and are preserved for the duration of the job. The control register specified in the control statement is set to the value of the expression supplied. This register can be R1, R2, R3, or EF (refer to the description of symbolic names). The R registers are 18-bit quantities whereas the error flag (EF) is a 6-bit quantity. Excess bits are ignored.

The statement format is:

```
SET(Ri=expression)
        or
SET(EF=expression)
        or
SET(SS=ssname)
```

Ri	Software-defined register 1, 2, or 3
EF	An additional register
expression	Any legal expression
ssname	Any legal subsystem name

Example:

This example illustrates the use of the SET statement to control execution of an object program. Because register R1 is set to 1 when file ABC is called, the object program is not executed.

```
SET(R1=1)
CALL(ABC)
FTN.
IF(R1=1)GOTO, 3.
REQUEST(TAPEI)
LGO.
3, REWIND(TAPEI)
        :
```

IF STATEMENT

The IF statement is used to evaluate an expression. If the conditions given in the expression are true, the dependent statement is processed. The expression is considered true if it is evaluated to a nonzero numeric value.

The statement format is:

```
IF(expression)stmt.
        or
IF(SS op ssname)stmt.
        or
IF(SS op ssname expression) stmt.
```

expression	Any legal expression
stmt	Any legal control language statement
op	One of the operators:
	=
	.EQ.
	≠
	.NE.
ssname	Any legal subsystem name

Example:

```
IF(R2=R1.AND.R3)GOTO,REQUEST.
SET(EF=1)
    :
REQUEST(TAPE)
```

If the expression is true, the REQUEST control card is executed; otherwise, the SET statement is executed.

Example:

```
IF(SS.EQ.BASIC.AND.R1=1)GOTO,100.
SET(SS.EQ.BASIC)
    :
100,OLD,BAS.
```

If the statement is true, the OLD control card is processed; otherwise, the SET statement is processed.

FILE STATEMENT

The FILE statement is used to determine the status of any file assigned to the job and is used in conjunction with the SET, IF, and DISPLAY statements.

The statement format is:

FILE(lfn,expression)

lfn	File name
expression	Any legal expression. FILE expressions, however, use different symbolic names.

Symbolic names:

Names with values

EQ	Equipment status table (EST) ordinal† (0 through 77_8)
ID	File ID (0 through 67_8)

Names with true/false values

MS	File is on mass storage
LK	File is locked
OP	File is opened
EX	Execute-only file
AS	File is assigned to user's control point

†Contact installation personnel for a list of EST ordinals.

File Types

LO	Local
PR	Print
IN	Input
PH	Punch
LI	Library
PM	Direct access permanent file
PT	Primary

Device Types

CP	415 Card Punch
CR	405 Card Reader
DA	6603 Disk System
DB	6638 Disk System
DC	863 Drum Storage
DD	853/854 Disk Storage Drive
DE	Extended Core Storage
DF	814 Disk File
DH	821 Data File
DI	844 Disk Storage Subsystem
DP	Distributive Data Path to ECS
DS	6612 Console Display
LP	501, 505, 512, or 580-12 Line Printer
LQ	512 Line Printer
LR	580-12 Line Printer
MD	841 Multiple Disk Drive
MS	Mass Storage
MT	Magnetic Tape Drive (7-track)
NE	Null equipment
NT	Magnetic Tape Drive (9-track)
ST	Remote Batch Multiplexer (6671)
TT	Time-Sharing Multiplexer (6676 or 6671)

Examples:

SET(R1=FILE(TAPE,MT))

If TAPE is a magnetic tape file, R1 is set to 1; otherwise, it is given a value of zero.

IF(FILE(TAPE,MT.OR.LI))GOTO,1ABC.

If TAPE is a magnetic tape file or a library file, the statement at 1ABC is processed.

NUM STATEMENT

The NUM statement is used to determine if the specified parameter name has a numeric value. It is used in conjunction with the SET, IF, and DISPLAY statements.

The statement format is:

NUM(name)

name	Parameter name. If the name is numeric, the statement is true; otherwise, it is false.

Example:

If the following CALL statement were used to call procedure file A:

CALL(A,RENAME(2XY=2,T=TAPE)

the IF statement in A:

IF(NUM(2XY))GOTO,1S.

would be evaluated as true, and control would transfer to 1S.

However, the statement:

IF(NUM(T))GOTO,1S.

would be evaluated as false, and control would pass to the next statement in A.

PROCEDURE FILES

Procedure files are source files consisting of control statements or system control cards, or both. The first statement of a procedure file may be the file name. If the first statement is the same as the file name used in the CALL statement, the first statement is ignored. Procedure files are activated by the CALL statement or by using the name of the procedure file, if the file is in the system.

Example 1:

This procedure file edits the decks MTR and LINK and builds a new deadstart tape containing the edited routines. The procedure file determines whether the system OPL is on mass storage and then modifies and assembles the specified decks.

Input Deck

```
JOB(T7777, P17, CM60000)
USER(USERNUM, PASSWRD, FAM1)
REQUEST(TAPE)
CALL(MOD(T=TAPE,OUT=OUTPUT,BIN=LGO,1NEXT=UNLOAD)
UNLOAD(TAPE)
COMMON(SYSTEM)
LIBEDIT(P=SYSTEM, N)
COMMON(NEW)
-EOR-
*EDIT, LINK, MTR
-EOI-
```

Procedure File on File MOD

```
MOD
IF(FILE(T,MT))GOTO,2.
1,MODIFY,L=0.
COMPASS(I,L=OUT,B=BIN)
GOTO,1NEXT.
2,REWIND(T)
COPYBF(T,OPL)
GOTO,1.
```

After the CALL control card is processed, the input file is:

```
JOB(T7777, P17, CM60000)
USER(USERNUM, PASSWRD, FAM1)
REQUEST(TAPE)
CALL(MOD(T=TAPE,OUT=OUTPUT,BIN=LGO,1NEXT=UNLOAD)
IF(FILE(TAPE,MT))GOTO,2.
1,MODIFY,L=0.
COMPASS(I,L=OUTPUT,B=LGO)
GOTO,UNLOAD.
2,REWIND(TAPE)
COPYBF(TAPE,OPL)
GOTO,1.
UNLOAD(TAPE)
COMMON(SYSTEM)
LIBEDIT(P=SYSTEM, N)
COMMON(NEW)
-EOR-
*EDIT, LINK, MTR
-EOI-
```

Example 2:

This is an example of nested calls. It illustrates the use of one procedure file to skip a specified number of files on a tape (contents of R1) and to copy source data to the tape. The other procedure file retrieves source data from the OPL (old program library) and calls the first procedure file to place that source data on the tape.

Input Deck

```
JOBAAA.
USER(USERNUM,PASSWRD,FAM1)
REQUEST(TAPE)
MODIFY(S,Z)/*EDIT,LINK
SET(R1=0)
CALL(PROC,RENAME(A=TAPE,B=SOURCE,2=2A,3=3A)
SET(R1=R1+1)
CALL(PROB)
-EOR-
```

Procedure File PROB

```
PROB
MODIFY(S=NEW,Z)/*EDIT,MTR
CALL(PROC,RENAME(A=TAPE,B=NEW)
RETURN,NEW.
```

Procedure File PROC

```
PROC
REWIND(A,B)
SET(R2=0)
2,IF(R1=R2)GOTO,3.
SKIPF(A)
SET(R2=R2+1)
GOTO,2.
3,COPYBF,B,A.
```

WARNING

On job initiation the user's input file is a locked file. If the user wishes to call procedure files that write data on the input file, he should enter the RETURN (INPUT) control card before attempting to write on INPUT. For further information, refer to the description of Input File Control in section 3.

TIME-SHARING COMMANDS

The following commands are intended for use only by time-sharing origin jobs but included here for their use in procedure files. For additional information about these commands, refer to the Time-Sharing User's Reference Manual.

ASCII STATEMENT

The ASCII control statement specifies that all subsequent operations are to be done using the ASCII character set.

The control statement format is:

ASCII.

If this control statement is processed while output is still available, the terminal switches to ASCII mode for the remainder of the output.

CSET STATEMENT

The CSET control statement specifies the current character set mode of the terminal.

The control statement format is:

CSET(m)

m	Current terminal character set mode; m may be one of the following:	
	ASCII	Set ASCII character set mode; escape code processing is enabled
	NORMAL	Set NORMAL character set mode; escape code processing is disabled

If this control statement is processed while output is still available, the terminal switches to the new character set mode for the remainder of the output.

PARITY STATEMENT

The PARITY control statement sets the terminal to the indicated parity.

The control statement format is:

PARITY(p)

p	Terminal parity; p may be one of the following:	
	ODD	Set odd parity
	EVEN	Set even parity

If p is omitted, odd parity is assumed.

If this control statement is processed while output is still available, the terminal parity switches to the new parity for the remainder of the output.

Jobs entering the system consist of one or more logical records. The first logical record contains system directives (control cards) which describe the processing that is to occur in the job file (job deck). This section describes control card processing and how the control cards affect other aspects of job processing.

The KRONOS Operating System recognizes three types of control cards.

- Local File Control Cards — These cards call files that are assigned to the job control point. LGO is the system default local file used for retaining object code generated by one of the language processors described in section 11.
- System Control Cards — These cards are divided into eight categories.

  Job control control cards

  File management control cards

  Load and dump central memory utility control cards

  Loader control control cards †

  Permanent file control cards

  Tape management control cards

  Program library utility control cards

  System utility control cards
- Product Set Control Cards — The product set control cards call the various products available under KRONOS (refer to section 11).

CONTROL CARD FORMAT

All control cards may consist of from one to four fields. The first field is the statement label field. If present (the field is optional), it begins with a numeric character and terminates with a separator character. The field is used only in conjunction with the system control language described in section 4.

The second field, also optional, is a $ or / prefix character which precedes the program name. If a $ is present, it indicates that the specified program to be executed must be loaded from the system library. † Therefore, even if a local file of the same name is present, it will not be executed. The / option may be used on local file control card calls. If a / is present, it indicates that the parameters following the program name are to be processed in the operating system format. If a / is not present, the parameters will be processed in product set format. The default is product set format because it is assumed that most programs specified in local file calls have been generated by one of the product set members. The / option does not apply for control card calls to programs residing on the system library. For those types of calls, parameters are processed in the operating system format unless the SC directive to SYSEDIT has been entered. Refer to the SYSEDIT control card for a description of the SC directive.

† Refer to section 15 and the Loader Reference Manual.

The third field contains the name of the program to be executed. The fourth field (optional) contains parameters which further define the operation to be performed. The parameter field is set off from the name field by a separator character. After the fourth field, or the third field if no parameters are present, there must be a valid terminator character.

The following is a comparison of the operating system and product set formats.

Operating System Format	Product Set Format
1. Valid separators are + - " / = , ($	1. Valid separators are + - " / = , (and any other character with a display code value greater than 44_8 except *) $. and blank.
2. Valid terminators are .)	2. Valid terminators are .)
3. Letters, numbers, and the * are the only characters allowed in the parameter field. The one exception to this rule is the use of literals (that is, character strings delimited by dollar signs). Characters other than letters, numbers, and the * can be included in literals. No characters within a literal have special meanings; the system merely checks the syntax of the literal. The called program must do its own processing of the literal. Literals are allowed only on equipment/file assignment control cards and loader control control cards.	3. Any parameter field that includes characters other than letters, numbers, and the * must be expressed as a literal.
4. All embedded blanks within a control statement except those appearing in literals are ignored.	4. All embedded blanks within a control statement except those appearing in literals or after the program name . are ignored. A blank following the program name is considered a valid separator.
5. Comments may appear on the control card but they must follow the terminator. Comments may contain any character.	5. Same as for the operating system format.

Operating System Format	Product Set Format
6. Parameters, separators, and terminators are stored in the user's field length beginning at RA+2. The characters , . and) are stored as zero. For all parameters and all valid separators except the comma, their display code equivalent is stored.	6. Parameters are stored in their display code equivalent beginning at RA+2. Separators and terminators are stored as follows:

Character	Code (Octal)
,	1
=	2
/	3
(	4
+	5
-	6
Blank	7
;	10
) or .	17
Other valid separators	16

7. File names are 1 to 7 alphanumeric characters.	7. File names are 1 to 7 alphanumeric characters. File names beginning with a numeric character are illegal.
8. Not SCOPE compatible	8. SCOPE compatible

In general, no parameter can contain more than seven characters. If a parameter contains more than seven characters, the entire control card is issued to the dayfile, followed by the message:

FORMAT ERROR ON CONTROL CARD.

There are two exceptions to this rule. If a card calls a program from the system library that has an ARG= entry point, parameters in the card can contain more than seven characters. If a parameter contains more than seven characters, the ARG= entry point is not present, and the SDM= entry point is present, the statement name (such as DEFINE) is issued to the dayfile but all parameters are suppressed.

The parameters can appear in either order-dependent or order-independent format. Order-dependent parameters are required when the parameters must be passed in a specific order. An example of order-dependent parameters is

RESEQ(MYFILE,B,,20)

In this example, the system expects the resequencing increment to be passed as the fourth parameter; therefore, a separator must be present for the parameter not specified.

Order-independent parameters may be passed in any order. This is made possible by the use of keywords. Keywords are identifiers which have meaning either by themselves or when used in conjunction with other parameters. Usually, keywords are passed with a parameter and a separator. The separator must not be a comma. When the list of parameters is passed to the called program, all separators except commas are also passed.

Some programs require specific separators (usually =), and others merely require that a separator be present. Examples of keyword notation are:

1. COBOL(I=SFILE,B=BFILE)
2. COBOL(B=BFILE,I=SFILE)
3. COBOL(L=0,A,F)
4. JOBX,T10,CM45000.

In examples 1 and 2, both parameters and separators are passed to the COBOL compiler. Since these parameters are order-independent, both cards produce the same result.

In example 3, two keywords are passed with no separator character or parameter. In example 4, the keyword is the first character of the parameter.

The control statements are processed in the following manner: parameters are extracted from the control card and stored in the user's field length beginning at ARGR (RA+2) through RA+n (n cannot exceed 63_8). † The total number of parameters stored in the user's field length is placed in the lower 18 bits of $RA+64_8$. The name of the control statement is placed in bits 18 through 59 of $RA+64_8$.

The control card image, less any label or prefix field, is stored at $RA+70_8$. If the program being executed was loaded from the system library and has an ARG= entry point, then the entire control card image will be present at $RA+70_8$. Neither the information on arguments nor the argument count, however, will be entered when ARG= is present. This entry point allows for control cards with special parameter requirements (refer to appendix F, volume 2).

An example of how the control card

PERMIT(FILEABC/USERAAA=R,USERBBB=W)

appears in CM is illustrated.

		Memory					Display Code Equivalent	
ARGR	RA+2	0611	1405	0102	0300	0050	FILEABC	/
		2523	0522	0101	0100	0054	USERAAA	=
		2200	0000	0000	0000	0000	R	
		2523	0522	0202	0200	0054	USERBBB	=
	RA+6	2700	0000	0000	0000	0000	W	
	.							
	.							
	.							
ACTR	RA+64							
CCDR	RA+70	2005	2215	1124	5606	1114	PERMIT(FIL	
	.	0501	0203	5026	2305	2201	EABC/USERA	
	.	0101	5422	5625	2305	2202	AA=R,USERB	
	RA+73	0202	5427	5755	0000	0000	BB=W)	

The following control cards would provide exactly the same image in CM.

123,PERMIT(FILEABC/USERAAA=R,USERBBB=W)

123,$PERMIT(FILEABC/USERAAAA=R,USERBBB=W)

JOB CARD FORMAT

The first card of the control card record is always the job card. The job card may be in either order-dependent or order-independent format. When the job card is in order-independent format, the keyword and parameter are passed with no separator character. The format for the job card is:

† The first 101_8 words of the user's field length, from RA through $RA+100_8$, comprise the job communication area. Refer to appendix E, volume 2 for a description of this area.

jobname(Tt, CMfl, Pp) . . . . . cm

jobname(p, t, fl) . . . . . cm

jobname — Alphanumeric job name (1 to 7 characters) which must begin with a letter. This name identifies individual jobs being run under the same user number.

t — Central processor time limit in octal seconds, ranging from 1 to 77770_8. The time limit must be sufficient for completion of the job. If t is absent, the system assumes t equals 100_8 (100_8 seconds is approximately 1 minute).

fl — CM field length (storage requirement) for the job. The system rounds the value to the next highest multiple of 100_8. The field length cannot exceed:

$360,000_8$ on a 131K machine

$163,000_8$ on a 65K machine

$61,000_8$ on a 32K machine

If fl is absent, the system assumes fl = $50,000_8$.

NOTE

The following messages are issued to the user's dayfile if validation limits are exceeded.

CM NOT VALIDATED.	The number of CM words specified on the job card exceeds that for which the user is validated.
TL NOT VALIDATED.	The time limit specified on the job card exceeds that for which the user is validated.

The user may be further restricted by limits placed on him by the validation file or by installation parameters. Also, the user should consult installation personnel for additional restrictions based on the machine configuration and subsystems used.

p — Priority level (octal) at which the job enters the system; $1 \leq p \leq 17_8$.

This parameter is currently ignored since the system will automatically assign priorities specified by the installation parameters.

cm	Conversion mode contained in columns 79 and 80. A 26 indicates coded cards are to be converted in O26 mode; 29 indicates cards are converted in O29 mode. This is the initial keypunch mode of the job but mode may be changed by a conversion change card (refer to Coded Cards, appendix F) when reading cards or a DISPOSE card when punching cards. If this parameter is omitted, the system default keypunch mode is used.

In addition to the regular separator characters, the * may also be used to separate parameters on the job statement.

CONTROL CARD PROCESSING FLOW

The system translates a control statement by:

1. Reading the statement from the control point control card buffer. If necessary, the system reads control statements from file INPUT.

2. Deleting all spaces between the beginning of the statement and the terminator character (a period or a right parenthesis).† In general, the system allows only standard FORTRAN characters to appear before the terminator character, although other characters can appear within a literal or in the comment field.

3. Comparing special control card names with the name of the control card being processed. If the card name is CTIME, RTIME, or STIME, the system processes the control statement.

4. Searching the file name table for a file assigned to the job with a name identical to the name of the control statement. However, if a $ precedes the program name, this step is skipped. If an identical name is found, the program is loaded into memory. The arguments are extracted from the control statement and stored in RA+2 through RA+n+1 (n is the number of parameters). The CPU is requested to begin execution unless special loader control cards follow.

5. Searching the central library directory for a program name that matches the control card name. If the name is found, the system proceeds as in step 4; otherwise, the system searches further.

6. Searching the peripheral processor library directory for a program name that matches the control card name. If found, the name is placed, with a maximum of two arguments, as a peripheral processor request, and the system exits to the program.

7. If the control card name is not found during any of the above searches, the control statement is declared illegal and the job is aborted.

---

†If product set format is used, a blank following the program name is considered a valid separator.

Figure 1-5-1 illustrates the flow of control card processing.

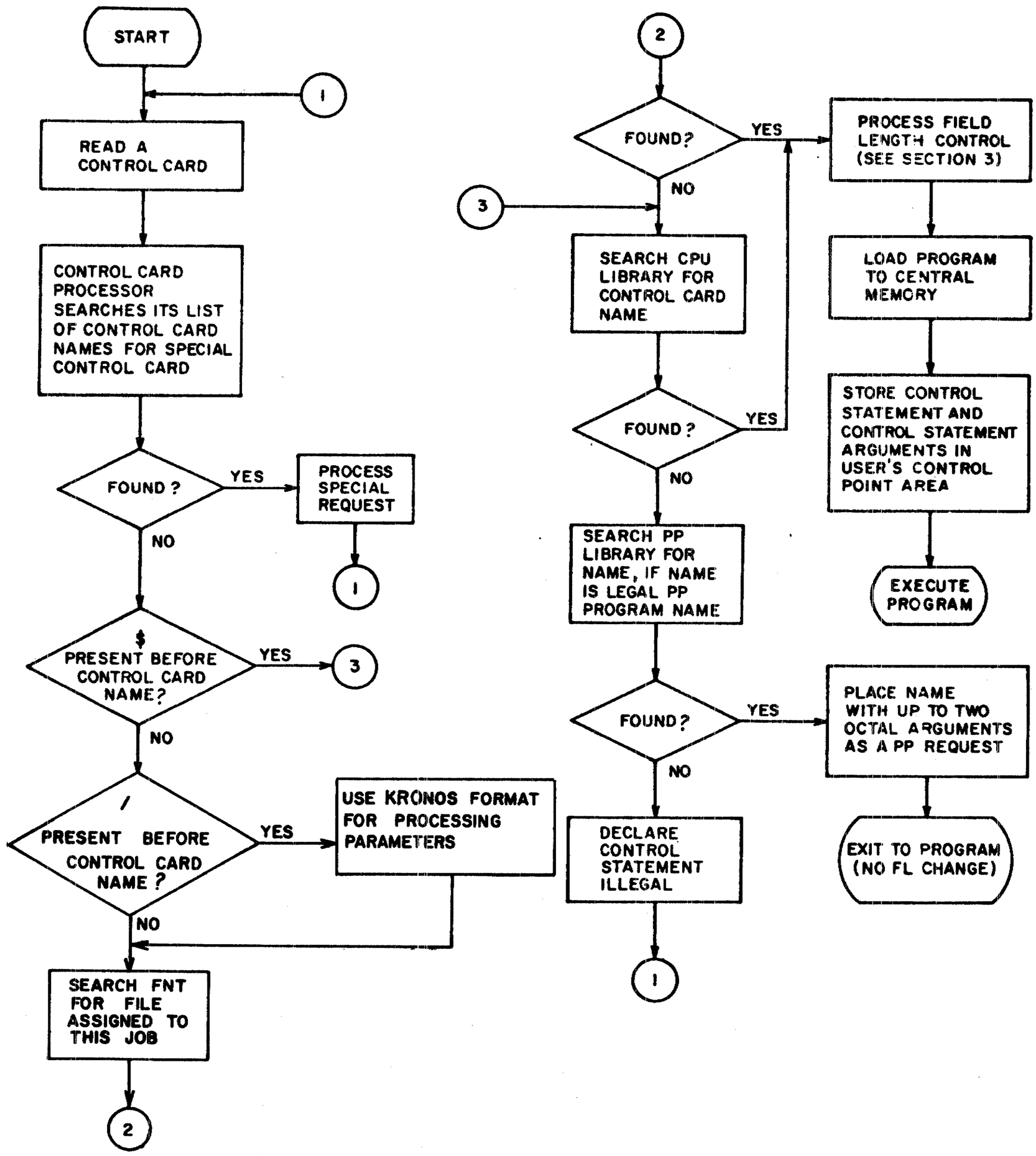


Figure 1-5-1. Control Card Processing Flow

EXIT PROCESSING

When an error condition occurs during job processing, the system searches the control card record for an EXIT card. If the record does not contain an EXIT card, the system terminates the job. If the system finds an EXIT card, it clears the error condition and processes the control cards that follow the EXIT card. If the error was a time limit error, the limit is reset to the time used plus 10_8 seconds. This gives the user time for post-error cleanup operations.

If a NOEXIT card is encountered, normal error processing is not performed. That is, if the no exit flag has been set (by the NOEXIT card) prior to the error, the error flag is cleared, no search is made for an EXIT card, and processing continues with the next control card. An ONEXIT card can be used to return to error processing mode; it clears the no exit flag. For further discussion of possible error conditions, refer to section 3 of this manual.

The following sequence of control cards illustrates this exit processing.

```
JOBABCD(T100,CM60000)
USER(USERNUM,PASSWRD,FAMA)
GET(INPUT,I=INPUT)
MODIFY(N,I=I,X)
LGO.
NOEXIT.
COMMON(SYSTEM)
LIBEDIT(P=SYSTEM,N,I=I)
ONEXIT.
REQUEST(TAPE)
COPY(NEW,TAPE,V)
UNLOAD(TAPE)
EXIT.
DMP(2000, 12500)
-EOR-
```

During the processing of control cards 1 through 5 and control cards 10 through 12, normal error processing is in effect. If an error condition occurs, the control cards are searched for an EXIT card. This is found at the thirteenth card and processing continues with the DMP card. If no errors occur before the NOEXIT card, normal error processing is turned off until the ONEXIT card is processed.

JOB CONTROL CONTROL CARDS 6

The job control control cards enable the user to alter information that controls his job while in the system and to retrieve information concerning the status of his job. The control cards included in this category are:

ACCOUNT	MODE	ROLLOUT
CHARGE	NOEXIT	RTIME
COMMENT	NORERUN	SETPR
CTIME	OFFSW	SETTL
DAYFILE	ONEXIT	STIME
ENQUIRE	ONSW	SUBMIT
EXIT	PASSWOR	SUI
LDI	RERUN	SUMMARY
LENGTH	RESOURC	SWITCH
LIMITS	RFL	USECPU
		USER

All of these cards, with the exception of PASSWOR and SUI, can be issued by the user regardless of his validation. A listing of validation information can be obtained using the LIMIT card. Although the user is allowed to change several control values for his job (such as RFL, SETPR, and SETTL), he can never specify more than that for which he is validated.

The system uses the USER card and CHARGE card for checking user validation and system accounting information. The RESOURC card is also used by the system to prevent deadlocks from occurring when several tapes or packs are used concurrently.

The user is enabled to submit files as batch origin type jobs through the LDI and SUBMIT control cards. He can specify the mode of error exit processing desired through use of the EXIT, ONEXIT, NOEXIT, and MODE cards. He can also set conditions for his program with sense switches (such as ONSW, OFFSW, and SWITCH). In the event of a system malfunction causing jobs to be recovered, he may either allow his job to be run again with the RERUN card or prevent it from being rerun with the NORERUN card. Additional information is returned to the user by the CTIME, RTIME, and DAYFILE cards. The COMMENT card allows the user to provide his own documentation.

ACCOUNT CARD †

The ACCOUNT control card is used to determine if the user is a legal user, which resources he is validated to use, and the extent (limits) to which he may use those resources.

The control card format is:

ACCOUNT(usernum, passwrd, familyname)

usernum	User's user number
passwrd	User's password
familyname	Optional parameter identifying the family†† of permanent file devices that have been or may be transferred from the user's normal system to a backup system.

The ACCOUNT card performs the same function as the USER card and is used in exactly the same way. All uses and restrictions apply to both. For further information, refer to the USER card in this section.

CHARGE CARD

The CHARGE card causes the system to record on the account dayfile all information regarding resources used under a specified charge number/project number combination. Its purpose is to control the accounting activity of the system for a customer or the installation.

The control card format is:

CHARGE(chargenum, projectnum)

chargenum	A 1- to 10-alphanumeric character charge number assigned to the user
projectnum	A 1- to 20-alphanumeric character project number assigned to the user

The CHARGE card is used in conjunction with user accounting control. An installation which implements this feature can impose limits on the SRUs or connect time a user may accumulate or restrict his access to the system to a certain time-of-day interval.

If access option 8 is not set (refer to LIMITS control card in this section), the user must include a CHARGE card immediately following every USER card in his job. If option 8 is set, the user may but is not required to include a CHARGE card. A user assigned more than one charge and/or project number may include additional CHARGE cards in his job to record resources used under each charge number/project number combination. Whenever a new CHARGE card is issued, the SRU information for the previous charge number/project number is written to the account dayfile and then cleared. However, the other accumulators (central processor time, mass storage activity, and so on) are not cleared but continue to increment. The following message is also issued when a new CHARGE card is entered.

yy.mm.dd. hh.mm.ss. jobname. ACCN, chargenum, projectnum.

For a complete listing of messages issued to the user's dayfile, refer to Job Completion in section 3.

†The ACCOUNT control card is included for compatibility with previous systems. The USER control card is recommended.

††Refer to section 2 for a description of permanent file devices.

COMMENT CARD

The COMMENT card is used to enter the specified comment in the system and user's dayfile.

The control card format is:

COMMENT. comments or

*comments

comments	Any combination of characters the user wishes to display

If the

*comment

format is used, the * must appear in column 1.

CTIME CARD

The CTIME control card requests that the accumulated CPU time for the job be issued to the user's dayfile (in seconds).

The control card format is:

CTIME.

DAYFILE CARD

The DAYFILE control card causes the system to write the user's control point dayfile to the file specified.

The control card format is:

DAYFILE(lfn)

lfn	File to which the dayfile is to be written; the default file name is OUTPUT.

ENQUIRE CARD

The ENQUIRE control card gives information about the system to the user. Three forms of the command are allowed.

The control card formats are:

ENQUIRE(OP=$p_1 p_2 \ldots p_n$, JN=jobname, FN=lfn_1, 0=lfn_2)

or

ENQUIRE($p_1 p_2 \ldots p_n$)

or

ENQUIRE.

p_i Any of the following options:

Option	Description
A	Causes all OP= options to be executed.
B	Returns to the user information about his user identification and his priorities, such as USER NUMBER, FAMILY NAME, CPU PRIORITIES, USER INDEX HASH, etc.
F	Status of files at the user's control point.

Example:

FILENAME	LENGTH/PRUS	TYPE	STATUS
BFILE3	81	LO.	EOR
OUTPUT	7	PR.	IC RE/WR
OPL	19349	PM.	EOR

J	Returns the contents of the user's control registers and error flag field.
L	Returns user's loader information.
R	Returns to the user the amount of resources he has used, such as CPU time, MS activity, MT activity, etc. These statistics are the factors that make up the SRU.
S	Gives the user his SRUs. The SRU represents the total usage of the system by the user. This unit is derived from central processor time, I/O activity, and memory usage.
T	Returns accumulated CPU time.
U	Returns the amount of resources remaining available to the user. Included in these resources are dispose files, mass storage, dayfile messages, and control cards.

jobname Last three characters of the name assigned by the system to a remote batch job that has been initiated with the SUBMIT card. When this parameter is specified, the status of the remote batch job is returned. It is only possible to obtain the status of jobs submitted under the current user number.

lfn_1 Local file name. When this parameter is specified, the status of the particular file is returned in the same manner as when the F option is specified.

lfn_2	Name of alternate file to receive output. If omitted, the system assumes OUTPUT.

The third form of the card (ENQUIRE.) defaults to the OP=A option. All OP= options are executed and information is printed on OUTPUT.

EXIT CARD

The EXIT control card indicates the position in the control card record where processing will resume if an error is encountered, or where to terminate normal control card processing if an error is not encountered. For additional information, refer to the description of the NOEXIT and ONEXIT control cards later in this section and to the description of exit processing in section 5.

LDI CARD

The control card format is:

LDI(lfn, id)

lfn	Name of file containing the batch job image to be submitted; if lfn is omitted, LOAD is assumed.
id	Identification code ($0-67_8$); if this parameter is omitted, 0 is assumed.

The LDI routine copies file lfn to mass storage and submits the file to the input queue with identifier id. The file is submitted as a batch origin type job.

LENGTH CARD

The LENGTH control card gives the user the current status of one of his local files.

The control card format is:

LENGTH(lfn)

lfn	Name of local file

The information given for the local file includes its length, type, and current status.

LIMITS CARD

The LIMITS control card directs the system to list validation information on file OUTPUT for the user named on the latest USER card.

The control card format is:

LIMITS.

Generally, validation limits are the internal system controls associated with each user number which govern his use of certain system resources. The listing provided describes both the resources available to the user and the extent to which they may be used. All numeric values listed are decimal unless the post radix B appears, signifying an octal value. The following information is listed.

Field	Description
AB†	Answerback identifier (1 to 10 alphanumeric characters) used for terminal identification
MT	Maximum number of magnetic tape units the user is allowed to have assigned to his job concurrently
RP	Maximum number of removable auxiliary devices the user is allowed to have assigned to his job concurrently
TL	Maximum amount of central processor time (cumulative CPU time slices) in seconds allowed for the user's job. TL represents the actual time limit divided by 10_8.
CM	Maximum number of central memory words that the user is allowed to request. The value stored for CM represents the actual word limit divided by 100_8.
NF	Maximum number of files that the user is allowed to have attached to a job concurrently
DB	Maximum number of deferred batch jobs that the user can have in the system concurrently If the user is validated for system privileges and DEBUG mode is set on the system display console or if the user is submitting jobs from system origin, this parameter is ignored. The user is allowed to submit as many jobs as desired.
FC	Maximum number of permanent files the user can have in each catalog. This limit applies to each catalog being accessed (main, public auxiliary, or private auxiliary).
CS	Maximum number of PRUs available to the user for indirect access files
FS	Maximum number of PRUs available to the user for any one indirect access file
PA†	Terminal parity (EVEN or ODD)
RO†	Specifies the number of rubout characters required for carriage return delay

† For further information about this field, refer to the Time-Sharing User's Reference Manual.

Field	Description
PX†	FULL or HALF duplex transmission mode
TT†	Terminal type
TC	Character set to be used by time-sharing terminal
IS	Initial subsystem for time-sharing terminal
MS	Maximum number of mass storage PRUs the user is allowed to additionally allocate via his job
DF	Maximum number of MESSAGE requests the user can issue to the system and/or job dayfiles
CC	Maximum number of batch control statements processed for a user. (Time-sharing processed control statements are excluded.)
OF	Maximum number of print and punch files the user can dispose to output queues
CP	Maximum number of cards that can be punched from a user's disposed punch file
LP	Maximum number of lines that can be printed from a user's disposed print file
EC††	Maximum number of ECS memory words that the user is allowed to request
SL††	Maximum number of SRUs the user is allowed
CN	Charge number to which the user is assigned
PN	Project number to which the user is assigned
DS	Maximum number of PRUs available to the user for any one direct access permanent file
AW	Access word; controls the user's access within the system according to the following options (assumed values are options 1, 3, and 4)

Option	Signifies
1	User can change his password.
2	User can use the privileged time-sharing commands.†††
3	User is allowed to create direct access files.
4	User is allowed to create indirect access files.

† For further information about this field, refer to the Time-Sharing User's Reference Manual.

†† Not currently used by the system but provided for future expansion of validation control.

††† For further information about privileged time-sharing commands, refer to the operator's guide.

Option	Signifies
5	User can have system origin (SYOT) capability from any job origin if the system console is in DEBUG mode. The user is allowed to assign a device by its EST ordinal although the system need not be in DEBUG mode to do so. The user is allowed to call the customer engineering PPU based diagnostics if ENGINEERING mode (ENGR) is set at the system console.
6	User can access/create library files.
7	User can assign nonallocatable devices. A nonallocatable device is a magnetic tape unit, card reader, card punch, or line printer. Refer to File Management Control Statements in section 7 for further information.
8	User is allowed to access the system without supplying his assigned charge and project numbers.
9	User can define, save, and replace files on auxiliary devices.
10	User can access special transaction functions.
11	Allow no terminal timeout.

The octal value listed for AW corresponds to the preceding options where bit 0 is option 1, bit 1 is option 2, and so on. For example, if the access word listed were:

AW=00000000000000000215

the user would be validated for options 1, 3, 4, and 8.

The LIMITS card is equivalent to the OP=I option of MODVAL. If any parameters are included on the LIMITS card, the system issues the following message to the user's dayfile.

ERROR IN LIMITS ARGUMENTS.

MODE CARD

The control card format is:

MODE(m)

m Exit mode ($0 \leq m \leq 7$)

The following values can be supplied for m.

m	Error Exit Mode
0	Disable exit mode; no selection made
1	Address out of range because: • Attempt was made to reference CM or ECS outside established limits, or • Attempt was made to reference last 60-bit word (word 7) in relative address FL of ECS.

m	Error Exit Mode
2	Operand out of range; floating-point arithmetic unit received an infinite operand
3	Address or operand out of range (mode 1 or 2)
4	Indefinite operand; floating-point arithmetic unit received an indefinite operand
5	Indefinite operand or address out of range (mode 1 or 4)
6	Indefinite operand or operand out of range (mode 2 or 4)
7	Indefinite operand, operand out of range, or address out of range (mode 1, 2, or 4). If no mode is selected, the system assumes m=7.

The MODE card is used to define the error conditions that cause the system to exit from normal processing. When the specified error occurs, the system sets the appropriate error flag and exits from normal processing to perform any error processing required. If an error occurs for which the exit mode is not selected, the system notes the error, skips the operation that is causing the error, and continues normal processing. Note that if exit mode 3, 5, 6, or 7 is specified, a combination of exit modes 1, 2, and 4 is actually selected. For example, if exit mode 5 is specified, an error exit will occur for either a mode 1 or mode 4 error condition. Refer to Error Control, section 3, and to the CDC CYBER 70 and the 6000 Series Computer Systems Reference Manuals for further information about the processing of mode errors.

NOEXIT CARD

The NOEXIT control card suppresses the transfer of control to the statement following the next EXIT statement if an error occurs.

The control card format is:

```
NOEXIT.
```

If a NOEXIT card has appeared in the control card record and an error occurs, processing continues with the next control card, if possible (that is, if error does not cause job to abort). Refer to the description of exit processing in section 4 for further information.

NORERUN CARD

The NORERUN control card allows a user to clear job rerun status.

The control card format is:

```
NORERUN.
```

If the NORERUN card has been issued, the job may not be rerun. This may be desirable to prevent updating of an important data base when the job would otherwise be rerun.

This card is ignored from a time-sharing origin job.

OFFSW CARD

The OFFSW control card clears the pseudo-sense switches for reference by the user's program.

The control card format is:

OFFSW($s_1, s_2, \ldots, s_n$)

s_i	Sense switch to be cleared; $1 \leq s_i \leq 6$. If $s_i=0$ is specified, all sense switches are cleared.

The system stores the sense switch settings in the user's control point area and copies them to RA for use by the central program. The system operator can change these settings by console command.

ONEXIT CARD

The ONEXIT control card causes the transfer of control to the statement following the next EXIT statement if an error occurs.

The control card format is:

ONEXIT.

The ONEXIT card reverses the effect of a NOEXIT card. If an error occurs in processing a card following ONEXIT, control transfers to the card following the next EXIT card. Refer to the description of exit processing in section 4 for further information.

ONSW CARD

The ONSW control card sets the pseudo-sense switches for reference by the user's program.

The control card format is:

ONSW($s_1, s_2, \ldots, s_n$)

s_i	Sense switch to be set; $1 \leq s_i \leq 6$. If $s_i=0$ is specified, all sense switches are set.

The system stores the sense switch settings in the control point area and copies them to RA for use by the central program. The system operator can change these settings by console command.

PASSWOR CARD

The PASSWOR control card is used to change the user's password.

The control card format is:

PASSWOR(oldpswd, newpswd)

oldpswd	Old password
newpswd	New password

The user's password is changed from oldpswd to newpswd. The user can change his password only if access option 1 is set (refer to the LIMITS control card in this section). If option 1 is not set and the user submits a PASSWOR card, the system issues the following message to his dayfile.

ILLEGAL CONTROL CARD.

If the control card parameters are in error, the system issues the following message.

ERROR IN PASSWOR ARGUMENTS.

RERUN CARD

The RERUN control card allows a user to set job rerun status.

The control card format is:

RERUN.

If the RERUN card has been issued, the job may be rerun. This card is ignored from a time-sharing origin job.

RESOURC CARD

The RESOURC control card is necessary in any job that uses more than one tape or pack concurrently in order to prevent deadlocks with other jobs which may need the same resources.

The control card format is:

RESOURC(rt_1=u_1, rt_2=u_2, . . . , rt_n=u_n)

rt_i	Resource type:
	MT — Magnetic Tape Unit (7-track)
	NT — Magnetic Tape Unit (9-track)
	DDi — 854 Disk Storage Drive ($1 \leq i \leq 8$)
	DIi — 844 Disk Storage Subsystem ($1 \leq i \leq 8$)
	MDi — 841 Multiple Disk Drive ($1 \leq i \leq 8$)
u_i	Maximum number of units of resource type rt_i this job will use concurrently; any rt_i-u_i entry can be changed on subsequent RESOURC control cards.

The system manages the use of tape units and disk packs in such a way as to prevent deadlocks from occurring. A deadlock would occur if the system, by assigning a tape unit or pack to one job, prevented another job with currently assigned resources from completing. For example, an installation with two tape units is processing jobs A and B. Each job needs both units during some phase of processing. Job A is assigned unit 1. If job B were assigned unit 2, neither A nor B could complete until the other job relinquished its assigned unit.

The system prevents such situations by requiring that a RESOURC control card be included in any job that uses more than one tape or pack concurrently. When a job that includes a RESOURC card is submitted, the system first checks if the specified number of units exceeds the number of units for which the user is validated† or the number of units available at the installation. If either of these situations occurs, the system issues an error message to the user's dayfile and aborts the job.

When the job requests a tape or pack††, the system compares the number of units that jobs being processed have scheduled via RESOURC cards with the number of units actually assigned. If it determines that the assignment would cause a deadlock, it rolls out the job until a deadlock would not occur. If the assignment would not cause a deadlock, the system searches for the requested tape or pack. If found, it is assigned to the requesting job. If the pack is not found and the NA keyword was included in the request or if the tape is not found, the requesting job is rolled out until the operator makes the pack or tape available.

Thus, in the previous example, a RESOURC card would be required in both jobs. The information supplied by the cards would enable the system to anticipate the deadlock situation and roll out job B until job A no longer needed both units.

Under certain conditions the system overcommits resources, provided all jobs with currently assigned resources can complete. For example, an installation with three tape units is processing jobs A and B. Included in each job is a RESOURC card scheduling two units. Job A requests its first tape. It is assigned the tape (unit 1) because there are enough units available for job A to complete. Job B requests its first tape. It is assigned the tape (unit 2) because either A or B can complete if assigned the last unit, and when the job that is assigned the last unit completes, the other can then use that unit and also complete. Job B then requests and is assigned its second tape (unit 3). It completes its operations (that is, terminates or returns the files on the tape) and makes the unit available for job A to complete.

The system manages resources by keeping totals of the number of scheduled units and assigned units. Each total can vary during job processing. A user can increase the number of scheduled units by RETURNing all files attached to his job residing on resource units not currently needed and then scheduling the required number of units with another RESOURC card. He can decrease the number of scheduled units by including RETURN cards or additional RESOURC cards.

---

† For jobs that use only one tape or pack at a time and do not contain a RESOURC card the system checks validation limits when the request is made.

†† Refer to Permanent File Control Cards for a description of disk pack requests and to Tape Management Control Cards for a description of tape requests.

In the following job, for example, the second RESOURC card increases the number of scheduled disk drives and decreases the number of scheduled tape units.

```
SAMSJOB(CM50000,T40)
USER(SJGREEN,WGT,ALTFAM)
RESOURC(NT=2)
      .
      .
      .
      .
RESOURC(DD1=2,NT=1)
      .
      .
      .
      .
-EOI-
```

At some time during this phase of processing, the job will require two 9-track tape units.

During this phase, the job will require two 854 Disk Storage Drives and one 9-track tape unit. The N=1 entry decreases the number of scheduled tape units from two to one.

If the user decreases the total to less than the number of currently assigned units or increases the total to a point where a deadlock would occur, the system issues an error message to the user's dayfile and aborts his job.

The method of assigning units depends on the resource type. For example, all tapes and all private disk packs not accessible by alternate users can only be assigned to one job at a time. All public packs and those private packs accessible by alternate users are sharable, and therefore, can be assigned to several jobs at the same time.

On indirect access file requests the pack is charged to the job in fulfilling its resource demand only if the request causes the pack to be mounted. For direct access file requests, the pack is charged to the job when the first ATTACH of a direct access file is made.

A unit is assigned to a job until the job terminates or all direct access files residing on the unit that are assigned to the job are RETURNed. At this point a tape or a non-sharable pack can be dismounted. A sharable pack, however, can be dismounted only when there are no files residing on the unit that are assigned to any of the jobs sharing the pack.

NOTE

In GET requests for indirect access files, a pack is assigned to a job only as long as the pack is actually being used (that is, until the system retrieves the local copy of the file). Therefore, during a series of GET requests, the operator may determine that the pack is not being used and dismount it. If the user has a direct access file on the pack, he can avoid this situation by attaching the direct access file before issuing the GET requests.

RFL CARD

The RFL control card allows the user to change the job field length from that specified on the job card. It also sets the restoration field length for all jobs.

The control card format is:

RFL(nnnnnn)

nnnnnn Field length (octal)

The restrictions on nnnnnn are the same as those described for the job card in section 4. If the requested field length exceeds that for which the user is validated, the following error message is issued to the user's dayfile.

FL BEYOND USER LIMIT.

ROLLOUT CARD

The ROLLOUT control card requests that the user's job be rolled out and all memory assigned to the job released.

The control card format is:

ROLLOUT.

The user's job is entered into the rollout queue and is rescheduled by the system.

RTIME CARD

The RTIME control card requests that the current time be read from the realtime clock and issued to the dayfile (in seconds).

The control card format is:

RTIME.

SETCORE CARD

The SETCORE control card presets each word within the field length.

The control card format is:

SETCORE(p)

or

SETCORE(-p)

p Any of the following: (If a minus sign precedes the parameter p, the complement of p is set in core.)

p	Fill Characters
0	0
ZERO	Zeros (0)
INDEF	Indefinite (1777 0000 0000 0000 0000)
INF	Infinite (3777 0000 0000 0000 0000)

Each word within the field length is set to p. If p is omitted, the system assumes p=0.

ROUTE CARD

The ROUTE control card allows a batch user (cards or TELEX) to route a job to a specific OUTPUT and location.

The control format is:

ROUTE(lfn,EJ,C=cc,L=loc,EQ=eq,AT=at,WASTE or SAVE)

lfn	Logical file name. No default.
EJ	If present,ROUTE occurs at end-of-job. Default= immediate.
L=loc	Location to which file is to be directed. Default= location at which job was submitted.
EQ=eq	Output device to which file is to be directed. Default= any printer
AT=at	Attribute.Default- none.
C=cc	Copy count. Default=1.
WASTE or SAVE	Will ensure that each page of printout begins on a new page. Particularly helpful for a printout to be in continuous use.
D=d	Courier Drop Code.
H	Hold in output queue* (zero print priority) *Cyber 172
PP	Previewing Prohibited
PJ	Project Number } IUPUI use
PG	Programmer Number } IUPUI use
PW	Password } IUPUI use
HR	Hold in output queue at remote site

SETPR CARD

The SETPR control card allows the user to specify a new CPU priority for his job.

The control card format is:

SETPR(p)

p	Priority, $1 \leq p \leq 70_8$; if p exceeds that for which the user is validated, it is reduced to that value.

The CPU priority controls the assignment of the CPU to active jobs. If the CPU priority is lower than that of other jobs, the job is assigned to the CPU only when jobs of a higher priority do not need it. The user is validated for a maximum CPU priority. He cannot request a level that exceeds this value or 70_8 (the maximum CPU priority).

SETTL CARD

The SETTL control card allows the user to specify a new CPU time limit for his job.

The control card format is:

SETTL(t)

t	Central processor time limit in octal seconds (maximum is 77777_8); t is accurate to the nearest second.

The CPU time limit is the amount of time (in seconds) that a job is allowed to use the CPU before the error message:

TIME LIMIT EXCEEDED.

is issued by the system.

The user is validated for a maximum time limit. If this is exceeded or $0 \leq t \leq 77777_8$ is not satisfied, the following message is issued.

ILLEGAL USER ACCESS.

If t is between 77770_8 and 77777_8, the time limit is infinite. The user cannot set a time limit greater than that for which he is validated.

STIME CARD

The STIME control card requests that the accumulated SRU value for the job be issued to the user's dayfile.

The control card format is:

STIME.

SUBMIT CARD

The control card format is:

SUBMIT(lfn,q,NR)c

lfn	Name of the file to be submitted to the system for processing as a batch job
q	Specifies disposition of job output as follows: B Job output is disposed to local batch queue to be printed and/or punched at the central site (default value for nontime-sharing origin jobs) N Job output is disposed to local batch queue, but is dropped at job termination (default value for time-sharing origin jobs) E Job output is disposed to Export/Import queue for printing at a remote batch terminal.
NR	No rewind option; inhibits rewind of file specified by reformatting directive cREAD. If omitted, file specified by cREAD directive is automatically rewound.
c	Escape character used to identify reformatting directives in the file to be submitted (lfn). If omitted, the system assumes c=/.

The submit file lfn contains a batch job submitted to the system for processing. The reformatting directives described in this section are provided to aid the user in preparing the submit file. When the SUBMIT card is processed, the submit file can be reformatted according to the directives that appear in the file.

Each line in the submit file preceded by an escape character is recognized by KRONOS as a reformatting directive. The escape character to be used must be specified on the SUBMIT card (/ by default). Throughout this description, the letter c, preceding a directive, denotes the escape character. Reformatting directives may be interspersed throughout the submit file as long as transparent mode is not in effect. Transparent mode is selected by the cTRANS directive and requires that the user observe special rules when inserting subsequent directives into the file (refer to description of cTRANS and cNOTRANS directives).

The system does not process reformatting directives unless the first line of the submit file contains the cJOB directive. In addition, the first two card images following the cJOB directive (second and third cards of the submit file) must be a job and USER card, respectively. All following information is determined by the user. Thus, the first three lines of a submit file that is to be reformatted before processing should be:

```
ln1 cJOB
ln2 jobname,...
ln3 USER,...
```

where ln1, ln2, and ln3 are optional line numbers.

The SEQ and NOSEQ directives are used to determine, during reformatting, if the submit file will contain leading line numbers. Therefore, it is a simple matter to include line numbers on the entire submit file and specify which line numbers are to be removed during reformatting. This is especially useful if the submit file contains a BASIC program where line numbers are a requirement of the language.

The reformatting directives available are described as follows:

cJOB — Indicates that the submit file is to be reformatted and selects the following default reformatting directives. The default directives remain in effect until specified otherwise.

cNOTRANS	(disabled by cTRANS)
cSEQ	(disabled by cNOSEQ)
cPACK	(disabled by cNOPACK)

The cJOB directive must be the first line of the submit file. If omitted, the file is not reformatted.

cEOR — Indicates that an end-of-record mark is to be placed at this point in the submit file during reformatting.

cEOF — Indicates that an end-of-file mark is to be placed at this point in the submit file during reformatting.

cSEQ — Indicates that the following lines are preceded by line numbers and requests that they be removed (default value).

cNOSEQ — Reverses the effect of the cSEQ directive. No attempt is made to remove leading line numbers from subsequent lines.

cPACK — Requests that all succeeding end-of-record and end-of-file marks be removed (default value). This directive applies only to internal EOR and EOF marks that currently exist. The cEOR and cEOF reformatting directives are not affected.

cNOPACK — Reverses the effect of the cPACK directive. Requests the system not to discard succeeding internal end-of-record and end-of-file marks that currently exist.

cTRANS — Indicates transparent mode. When the system encounters this directive, it checks the next line of the submit file for an additional directive. If one exists, it is processed and the next line is checked. This continues until a line that is not a reformatting directive is encountered. Transparent mode is then selected and all directives that exist on subsequent lines are ignored until an internal EOR or EOF is encountered (this pertains only to EOR and EOF marks that currently exist, not cEOR and cEOF directives). The cPACK and cNOPACK directives determine if the internal EOR or EOF mark will be retained. The line following the internal EOR or EOF mark is then checked for a reformatting directive. If one exists, it is processed and the following line is checked. All directives are processed until a line that does not contain a reformatting directive is encountered. This causes transparent mode to be reset unless a cNOTRANS directive was encountered. This process continues until either the end of the submit file is reached or until a cNOTRANS directive following an internal EOR or EOF is encountered.

The cTRANS directive is typically used in conjunction with the cREAD directive. It allows the user to copy the contents of an existing file into the submit file at the location of the cREAD directive. Because the file is read in transparent mode, no check for reformatting directives is attempted until an internal EOR or EOF is encountered. Note that the cREAD directive must follow the cTRANS directive and must be located before the first succeeding line that is not a reformatting directive. If not, transparent mode is selected before the cREAI directive is encountered and the cREAD will be ignored.

The cSEQ or cNOSEQ directive in effect before transparent mode was selected has no effect upon the submit file or the file being read (cREAD) while transparent mode is in effect. Note, however, that the cPACK or cNOPACK directive in effect before transparent mode was selected remains in effect after it is selected.

cNOTRANS

Reverses the effect of the cTRANS directive and informs the system that the submit file is to be examined on a line-by-line basis. All directives encountered in the submit file while the cNOTRANS directive is in effect will be processed. This directive is initially selected by default and remains in effect until a cTRANS directive is encountered in the submit file.

The user should be careful in placing this directive in the submit file. If transparent mode is selected, this directive can possibly be ignored unless it immediately follows either a cREAD directive in the submit file or an internal EOR or EOF mark.

cREAD,lfn

Requests that the system read the entire contents of the specified file, lfn, and insert that file in place of the cREAD directive in the submit file, during reformatting. If the file to be read is not currently local to the job, the system automatically attempts a GET and then an ATTACH on the file. If lfn is not specified in the directive, TAPE1 is assumed. If the file specified cannot be found, the message

NO READ FILE - lfn.

is issued to the user's dayfile, and the job is terminated. If the read file is found to be busy (direct access files only), the message

READ FILE BUSY - lfn.

is issued to the user's dayfile, and the job is terminated. The file specified by lfn in the cREAD directive is automatically rewound before the read operation unless the NR parameter is specified on the SUBMIT control card, In this case, the rewind directive must precede the cREAD directive in the submit file if it is desired to rewind file lfn before the read operation begins. KRONOS returns all files specified in cREAD directives before completion of the job.

If the cPACK directive is in effect at the time of the read, all internal EOR and EOF marks will be removed. If the cNOPACK directive is in effect, all internal EOR and EOF marks are read into the submit file in the proper position during reformatting.

Unless transparent mode is in effect when file lfn is read, each line of that file will also be checked for a reformatting directive. Any directives contained in the file, except another cREAD, will be processed. The cREAD directive cannot be nested. In addition, any directives in effect before the cREAD directive is processed will remain in effect for the file being read, unless transparent mode is selected. Then, only the cPACK or cNOPACK directive remains in effect for the file being read. Moreover, only those directives that immediately follow an internal EOR or EOF in the file being read will be processed.

If the file to be read is a binary file, it is recommended that the cTRANS directive be used. This is to ensure that binary data will not be mistaken for a reformatting directive. The cTRANS directive should immediately precede the cREAD directive in the submit file, if used.

cREWIND,lfn — Requests that the system rewind file lfn to the beginning-of-information (BOI). If lfn is not supplied, TAPE1 is assumed. This directive is required only if the NR parameter is included in the SUBMIT command. Otherwise, file lfn is automatically rewound.

This directive is used in conjunction with the cREAD directive. Thus, if it is desired to rewind a file before the read operation begins, this directive must precede the cREAD directive in the submit file.

c_1EC=c_2 — Indicates that the escape code character is to be changed from c_1 (current escape code) to c_2 (new escape code). The new escape code will be used to recognize all subsequent reformatting directives until further change.

There is no restriction on the maximum number of characters per line for transparent mode. For all other modes, no line can exceed 150 characters.

If the user determines that an error occurred during processing of his job, he may reference a listing of the user's dayfile as an aid in identifying the cause of the error. The user's dayfile contains a record of the job processing activity and is disposed to the local batch queue or the Export/Import queue for printing when the job is terminated. However, all output is normally dropped at job termination when a batch job image is submitted from a time-sharing terminal. This includes the dayfile output as well as the job output. In this event, the user can make provisions within his job to save the contents of the dayfile if an error in processing occurs. This is done by including the following control cards at the end of the control card record.

```
lnx    EXIT.
lny    DAYFILE(lfn)
lnz    SAVE(lfn)
```

For further information about using SUBMIT from a time-sharing terminal, refer to the Time-Sharing User's Reference Manual.

SUI CARD

The SUI control card allows a user to access a permanent file catalog without using the USER statement.

The control card format is:

SUI(n)

n User index desired; $0 \leq n \leq 377777_8$.

The SUI card is useful if validation is not active. Only system origin jobs may issue this control card. If the job is not of system origin, the following message is issued.

CPM ILLEGAL REQUEST.

SUMMARY CARD

The SUMMARY control card gives information about the system to the user. Three forms of the command are allowed.

The control card formats are:

SUMMARY(OP=$p_1p_2 \ldots p_n$, JN=jobname, FN=lfn_1, 0=lfn_2)

or

SUMMARY($p_1p_2 \ldots p_n$)

or

SUMMARY.

p_i Any of the following options:

Option	Description
A	Causes all OP= options to be executed.
B	Returns to the user information about his user identification and his priorities, such as USER NUMBER, FAMILY NAME, CPU PRIORITIES, USER INDEX HASH, etc.
F	Status of files at the user's control point.

Example:

FILENAME	LENGTH/PRUS	TYPE	STATUS
BFILE3	81	LO.	EOR
OUTPUT	7	PR.	IC RE/WR
OPL	19349	PM.	EOR

J	Returns the contents of the user's control registers and error flag field.
L	Returns user's loader information.
R	Returns to the user the amount of resources he has used, such as CPU time, MS activity, MT activity, etc. These statistics are the factors that make up the SRU.
S	Gives the user his SRUs. The SRU represents the total usage of the system by the user. This unit is derived from central processor time, I/O activity, and memory usage.

Option	Description
T	Returns accumulated CPU time.
U	Returns the amount of resources remaining available to the user. Included in these resources are dispose files, mass storage, dayfile messages, and control cards.

jobname — Last three characters of the name assigned by the system to a remote batch job that has been initiated with the SUBMIT card. When this parameter is specified, the status of the remote batch job is returned. It is only possible to obtain the status of jobs submitted under the current user number.

lfn_1 — Local file name. When this parameter is specified, the status of the particular file is returned in the same manner as when the F option is specified.

lfn_2 — Name of alternate file to receive output. If omitted, the system assumes OUTPUT.

The third form of the card (SUMMARY.) defaults to the OP=R option. The resource usage of the job is listed on OUTPUT.

SWITCH CARD

The SWITCH control card sets the pseudo-sense switches for reference by the user's program.

The control format is:

SWITCH($s_1, s_2, \ldots s_n$)

s_i — Sense switch to be set; $1 < s_i \leq 6$. If $s_i = 0$ is specified, all sense switches are set.

The system stores the sense switch settings in the control point area and copies them to RA for use by the central program. The system operator can change these settings by console command.

This control card performs the same function as the ONSW control card.

USECPU CARD

The USECPU control card specifies which central processor is to be used when more than one is available for processing.

The control card format is:

USECPU(n)

n = 0	Either central processor can be used.
n = 1	CPU 0 is to be used (CDC CYBER 74-2X or the 6600 CPU on the 6700).
n = 2	CPU 1 is to be used (CDC CYBER 74-2X or the 6400 CPU on the 6700).

The USECPU card is to be used only when the system is running on a CDC CYBER 74-2X or 6700 system.

USER CARD

The system utilizes the USER control card to determine if the programmer is a legal user, which resources he is validated to use, and the extent (limits) to which he may use those resources. The USER card must follow the job card if user validation is active.

The control card format is:

USER(usernum, passwrd, familyname)

usernum	User's user number
passwrd	User's password
familyname	Optional parameter identifying the family† of permanent file devices that have been or may be transferred from the user's normal system to a backup system

This card defines controls and validation limits for the job and defines the user's permanent file base. The user can specify a different permanent file catalog during job processing by issuing another USER card. However, the access limits (refer to LIMITS control card in this section) for the user named on the first USER card remain in effect for all subsequent USER cards.

Normally, the familyname parameter need not be included on the USER card. However, if the user makes a practice of specifying his family name each time he submits a job, he can be sure that his job will be processed even if his normal system is not available and his permanent file family had to be moved to a backup system. If, after the first USER card, the user doesn't specify a familyname on the USER card, his permanent file family remains the same. If the user specifies the 0 (zero) familyname, his permanent file family becomes the system default family.

Example:

An installation has two systems, A and B. System B provides backup service for system A. The system default family name for system A is AFAM, and the system default family name for system B is BFAM.

During normal operations, system A user CWJONES with password JPWD could enter either of the following USER cards.

USER(CWJONES, JPWD)

USER(CWJONES, JPWD, AFAM)

System B user JDSMITH with password SPWD could enter either of the following cards.

USER(JDSMITH, SPWD)

USER, JDSMITH, SPWD, BFAM)

If system A failed, user CWJONES would be required to enter:

USER(CWJONES, JPWD, AFAM)

to identify his family of permanent file devices. User JDSMITH could enter either of the USER cards as before because the default family name would still be valid.

---

† Refer to section 2 for a description of permanent file devices.

If the user attempts to access permanent files on a device not present in the alternate system, one of the following messages is issued to the user's dayfile.

DEVICE UNAVAILABLE, AT nnn.	This message is issued if the user's master device† was not transferred to the backup system.
DIRECT ACCESS DEVICE ERROR. AT nnn.	This message is issued if the user attempted to reference direct access files on a device (other than his master device) not present in the backup system. †

†Refer to section 2 for a description of permanent file devices.

FILE MANGEMENT CONTROL CARDS 7

The file management control cards enable the user to manipulate files attached to his job. Many of these control cards are processed by the system processor, FILES. The control cards included in this category are:

ASSIGN	COPYEI	NEW	SKIPF
BKSP	COPYSBF	OUT	SKIPFB
CATALOG	COPYX	PACK	SKIPR
CLEAR	DISPOSE	PRIMARY	SORT
COMMON	DOCMENT	RENAME	STAGE
CONVERT	EVICT	REQUEST	TDUMP
COPY	GTR	RESEQ	UNLOAD
COPYBF	LIBEDIT	RETURN	UNLOCK
COPYBR	LIST80	REWIND	VERIFY
COPYCF	LOCK	SETID	VFYLIB
COPYCR	LO72	SKIPEI	WRITEF
			WRITER

The cards in this section allow the user to position his files, copy data from one file to another, specify method and format of input/output, sort his files, and add corrections. He can assign his files to a specific device type; change the file type, identification code, and write interlock status; and release them from job attachment. The user can also receive information about records in a file or documentation in a file containing COMPASS source code.

If an error is encountered in an operation on one file of a multiple file request, the operation is not performed on the following files. For example, if an error occurs in processing file B on the following control card:

```
COMMON(A, B, C, D)
```

files C and D are not processed.

If a file is not specifically assigned through the use of an ASSIGN, LABEL, or REQUEST control card, the system assigns the file to available mass storage. Refer to the ASSIGN and REQUEST cards in this section and Tape Management Control Cards in section 10 for a more detailed description.

ASSIGN CARD

The ASSIGN control card directs the system to assign a file to the specified device or device type.

The control card format is:

ASSIGN(nn, lfn, {CK / CB})

nn — Device or device type to which the specified file is to be assigned; nn may be either the EST ordinal† of a peripheral device or the device type as defined as follows:

Type	Equipment
CP	415 Card Punch
CR	405 Card Reader
DA	6603 Disk System
DB	6638 Disk System
DC	863 Drum Storage
DD	854 Disk Storage Drive
DE	Extended Core Storage
DF	814 Disk File
DH	821 Data File
DI	844 Disk Storage Subsystem
DP	Distributive Data Path to ECS
LP	501, 505, 512, or 580-12 Line Printer
LQ	512 Line Printer
LR	580-12 Line Printer
MD	841 Multiple Disk Drive
MS	Mass Storage Device
NE	Null Equipment
TT	Time-Sharing Multiplexer††

lfn — Name of the file to be assigned to the specified equipment

CK — Specifies that lfn is to be used as a checkpoint file. Each time a checkpoint dump is taken, the new information is written at the previous EOI of lfn.

CB — Specifies that lfn is to be used as a checkpoint file. Each time a checkpoint dump is taken the new information is written at the BOI of lfn.

Before performing the assignment, the system issues a RETURN on lfn. Any job can assign a file to MS and any time-sharing origin job can assign a file to TT. However, to assign any other devices, the job must be of system origin or the user must be validated for system origin privileges.†††

If the user attempts to perform an assignment for which he is not validated, the job is aborted and the following message is issued to the user's dayfile.

ILLEGAL USER ACCESS.

---

† Contact installation personnel for a list of EST ordinals.
†† This device type applies only to time-sharing origin jobs.
††† Refer to LIMITS control card, section 6.

In addition, to assign a file to any nonmass storage device except device type TT, the user must be validated to use nonallocatable devices. If the user does not have this validation or the device is not available, the system aborts the job.

The user should not normally assign any nonallocatable devices to his job. While it is possible to assign a central site card reader, line printer, or card punch directly on-line to the user's job, only a subset of the capabilities of local batch input/output are available through this method of access. Also, there is no need to assign nonallocatable devices to local files named OUTPUT, PUNCH, P8, or PUNCHB and any other local file disposed to an output queue because these files are always processed upon job completion.

Example 1:

ASSIGN(MS, OUTPUT)

This card assigns file OUTPUT to mass storage. With this assignment, a time-sharing user causes output normally printed at his terminal to be written on a mass storage file instead. Here, output means information generated by a program during execution. Informative and error messages are still printed at the terminal. Once this assignment is made, output is written on the mass storage file OUTPUT until the file is returned or reassigned.

Example 2:

ASSIGN(TT, XYZ)

This card assigns file XYZ to the user's time-sharing terminal. The assignment causes output that would normally be written on XYZ to be printed at the terminal instead.

Example 3:

ASSIGN(MD, ABC)

This card assigns file ABC to an 841 Multiple Disk Drive if one is available.

The ASSIGN card can also be used to create or access existing 7- or 9-track unlabeled tapes. For a description of the card as it applies to magnetic tape assignment, refer to Tape Management, section 10.

BKSP CARD

The BKSP control card directs the system to bypass a specified number of logical records in the reverse direction.

The control card format is:

BKSP(lfn, n, m)

lfn	Name of the file to be backspaced
n	Number of logical records (decimal) to backspace; if this parameter is omitted, the system assumes n=1.
m	File mode; C for coded, B for binary. If omitted, the system assumes the file is in binary mode.

The BKSP request can be issued at any point in a logical record. If, for example, FILE1 were positioned within the third record, a

BSKP(FILE1)

request would reposition FILE1 to the beginning of the third record. The system does not backspace past the beginning of information (BOI). However, EOF indicators are considered separate records and are included in the record count. An unrecognizable record count causes the following message:

ERROR IN FILE ARGUMENTS.

to be issued to the user's dayfile.

The BKSP card has no effect on a primary file since that file is rewound after every operation.

CATALOG CARD

The CATALOG control card requests a listing of information about each record in a specified file.

The control card format is:

CATALOG(lfn, p_1, p_2, . . . , p_n)

lfn	Name of the file to be cataloged	
p_i	May be one of the following:	
	N=0	Catalog until an empty file is encountered
	N=x	Catalog x files; default is N=1
	N	Catalog to end of information
	L=frame	Specifies the name of the file to receive output; if this parameter is omitted, the system assumes L = OUTPUT
	U	Select user library list (not given unless selected)
	D	Suppress all comment fields; suppress all page headings after the initial page heading for each individual file.
	R	Rewind lfn before and after cataloging
	CS	Suppress character set list for OPL (old program library) and OPLC (old program library common deck) type records.

The listing for each file of a multifile set begins on a new page with a page heading for that file. If the D option has been specified, the page heading appears only once, at the beginning of the file. The information listed included:

- Number of the record cataloged
- Record name from the first word of the record or the second word of the prefix (77) table, if present.
- Record type (list of valid record types follow this list)

- Length (less 77 table length) in words printed as an octal number
- A checksum (not including the 77 table)
- Dates and comments in 77 table, if present
- Character set mode for OPL/OPLC type records (unless suppressed by CS option)

Type may be one of the following.

• ABS	Multiple entry point overlay
• COS	Chippewa format central processor program
• OPL	Modify old program library deck
• OPLC	Modify old program library common deck
• OPLD	Modify old program library directory
• OVL	Central processor overlay
• PP	6000 series peripheral processor program
• PPU	7600 peripheral processor program
• REL	Relocatable central processor program
• TEXT	Unrecognizable as a program
• ULIB	User library program

Entry points are listed for REL and ABS format records. The entire record is listed for TEXT format records if the name of the record begins with CMRDECK, CMRDC, IPRDECK, IPRDC, LIBDECK, or LIBDC. The first line is listed for TEXT format records if the name of the record begins with OVERLAY. Correction identifiers and their YANK status (refer to the Modify Reference Manual) are listed for OPL and OPLC records.

A ULIB format record suppresses listing of records in the library unless the U option is specified on the control card. Zero-length records cause the length since the last zero-length record to be listed. EOFs cause the length since the last EOF to be listed.

```
        CATALOG OF SYSTEM                FILE      1
REC     NAME        TYPE             LENGTH     CKSUM      DATE      COMMENTS

301     PROFILE     ABS                5112      7106   75/04/22. 73/05/24.
         PROFILE
         ARG=
         RFL=
         SSJ=
302     SFS         OVL 01,00          1716      0262   75/04/19. 73/05/24.
303     0AV         PP (0000)           155      7256   75/04/19. 73/05/05.
304     2TJ         PP (2000)           140      0744   75/04/19. 70/12/13.
305     (00)            SUM =         25023

306     1BA         PP (1100)           401      6467   75/04/19. 74/04/22.
307     3BA         PP (2000)            57      5751   75/04/19. 74/04/22.
308     3BB         PP (2000)           374      0425   75/04/19. 74/04/22.
309     3BC         PP (2000)           136      2771   75/04/19. 74/04/22.
310     3BD         PP (2365)            50      5527   75/04/19. 74/04/22.
311     3BE         PP (2365)            27      6720   75/04/19. 74/11/25.
312     1CD         PP (1100)          1051      2713   75/04/19. 75/03/20.
313     1IO         PP (1100)           165      7162   75/05/08. 74/04/22.
314     3IA         PP (2213)           237      6117   75/05/08. 74/04/22.
315     3IB         PP (4113)            37      2354   75/05/08. 74/04/22.
316     3IC         PP (2213)           163      5366   75/05/08. 74/04/22.
317     5IA         PP (4325)            74      6335   75/05/08. 74/04/22.
318     5IC         PP (4325)            74      7601   75/05/08. 74/04/22.
319     5IE         PP (4325)           117      0462   75/05/08. 74/04/22.
320     5IG         PP (4325)           117      7764   75/05/08. 74/04/22.
321     2LP         PP (2000)           253      7000   75/04/19. 70/12/13.
322     2PC         PP (2000)           270      6063   75/04/19. 70/12/13.
323     2RC         PP (2000)           302      0151   75/04/19. 70/12/13.
324     (00)            SUM =          4723
```

Figure 1-7-1. Sample Page of Catalog of System

```
325   E200CP     OVL 00,00      672      5255  75/04/19. 75/03/23.
326   XSP        PP (1100)      255      3067  75/04/19. 74/05/19.
327   1ED        PP (1100)     1075      1267  75/04/19. 74/05/19.
328   1LS        PP (1100)      123      3727  75/04/19. 74/05/19.
329   9IA        PP (2000)      555      1661  75/04/19. 74/05/19.
330   9IB        PP (6000)      122      2452  75/04/19. 74/05/19.
331   9IC        PP (6000)      121      6122  75/04/19. 74/05/19.
332   9ID        PP (2000)       25      6133  75/04/19. 74/05/19.
333   9IE        PP (2000)       64      7515  75/04/19. 74/05/19.
334   9IF        PP (2401)      131      6712  75/04/19. 74/05/19.
335   9IG        PP (1100)       35      2476  75/04/19. 74/05/19.
336   9IH        PP (7000)       77      0056  75/04/19. 74/05/19.
337   (00)           SUM =     4005

338   CS1        PP (1123)      533      5264  75/04/20. 71/11/16.
339   1TP        PP (1100)      140      2011  75/04/20. 72/09/01.
340   DATADEF    OVL 00,00    14222      7267  75/04/20. 71/10/13.
341   DATAMAP    OVL 00,00    11330      3602  75/04/20. 71/10/04.
342   DBFORM     OVL 00,00    14626      6462  75/05/08. 05/08/72.
343   KTSDMP     ABS           6643      6370  75/04/20. 72/09/23.
       KTSDMP
       RFL=
344   LIBTASK    ABS           7653      4432  75/04/20. 71/09/01.
       LIBTASK
       RFL=
345   PRESIM     OVL 00,00    15446      3173  75/04/20. 73/05/06.
346   TRANEX     OVL 00,00     7037      5646  75/05/20. 75/04/20.
347   TRANEXA    OVL 01,00      260      6643  75/05/20. 75/04/20.
348   TRANEXB    OVL 01,00      170      3744  75/05/20. 75/04/20.
349   TRANEXC    OVL 01,00      235      2473  75/05/20. 75/04/20.
350   TRANEXD    OVL 01,00      253      4042  75/05/20. 75/04/20.
351   TRANEXE    OVL 01,00      252      7276  75/05/20. 75/04/20.
352   TRANEXF    OVL 01,00      174      2614  75/05/20. 75/04/20.
353   TRANDBM    OVL 01,00     6140      2655  75/05/20. 75/04/20.
354   TRANEX1    OVL 00,00     6332      3221  75/05/20. 72/04/29.
355   TRANEX2    OVL 01,00     1007      7120  75/05/20. 75/04/20.
356   TRANSIM    ABS           1455      0562  75/04/20. 71/12/09.
       TRANSIM
       RFL=
       SSJ=
357   TRANLIB    ULIB           113      5272  75/04/20.
375   BDMLIB     ULIB           140      2502  75/05/20.
395   (00)           SUM =   143246

396   DEMUX      ABS           3303      2744  75/04/20. 74/05/07.
       DEMUX
       RFL=
```

Figure 1-7-1. Sample Page of Catalog of SYSTEM (Cont'd)

CLEAR CARD

The CLEAR control card releases all the user's current working files.

The control card format is:

CLEAR.

If a primary file exists, only the file name is retained; information within the file is purged. The empty file remains available as the primary file.

COMMON CARD

The COMMON control card is used to either create or access a library type file.

The control card format is:

COMMON(lfn_1, lfn_2, . . . , lfn_n)

lfn Logical file name

The user must be validated to access/create library files. The specified file must be a local mass storage file. If lfn is not local, a search is made for a library file by that name and an error message issued if the file is not found. If the operation completes successfully, the file is attached to the user's job as a library type file.

Before a local file can be made a library file, it must be locked. Refer to the LOCK control card.

CONVERT CARD

The CONVERT control card converts records from one character set to another.

The control card format is:

CONVERT(p_1, p_2, . . . , p_i)

p_i May be one of the following:

- $P=lfn_1$ Input on file lfn_1; if omitted, file OLD is assumed
- $N=lfn_2$ Output on file lfn_2; if omitted, file NEW is assumed
- $RS=n_1$ Maximum record size in characters (decimal); $1 \leq n \leq 500$. If omitted, 300 is the assumed maximum record size. (Each character is 6 bits.)
- 64 Convert from 63-to 64-character set; if omitted, no conversion takes place. The TS option must be specified if 64 is not.
- $TS=t_1$ Convert from old to new time-sharing character set; t_1 may be one of the following terminal types:

t_1	Terminal Type
TTY	ASCII code terminal with standard print
COR	Correspondence code terminal with standard print

t_1	Terminal Type
CORAPL	Correspondence code terminal with APL print
MEMAPL	Memorex 1240 (ASCII code) terminal with APL print
BLKEDT	ASCII code terminal with standard print, block edit mode

If t_1 is omitted, it is assumed to be TTY. If TS is omitted, no time-sharing conversion takes place. The 64 option must be specified if TS is not.

R — Rewind input and output files prior to processing. If omitted, no rewind takes place.

RC=n_2 — Convert n_2 decimal records. If n_2 is omitted, convert until an EOF is encountered. If RC is omitted, one record is assumed.

The following table lists legal conversion using the appropriate CONVERT parameter.

Type of Record	Legal Conversion Parameters
63-character set, nontime-sharing record	64
Old time-sharing record	TS or 64 and TS
New NORMAL time-sharing record (equivalent to BATCH character set)	64
New ASCII time-sharing record	None

COPY CARD

The COPY control card causes the first file specified to be copied to the second file.

The control card format is:

COPY(lfn_1, lfn_2, x, c)

lfn_1 — Name of the file to copy from; if this parameter is omitted, file INPUT is assumed.

lfn_2 — Name of the file to copy to; if this parameter is omitted, file OUTPUT is assumed.

x — If a third parameter (1 to 7 alphanumeric characters) is present, both files are rewound before the copy begins and rewound, verified, and rewound again after the copy is complete.

c If a fourth parameter (1 to 7 alphanumeric characters) is present, the copy to or from an SI, S, or L format tape is performed in coded rather than binary mode.

The copy begins at the current position of both files, unless the x parameter is specified, and continues until an empty file (a double EOF) or EOI is encountered in lfn_1. If the copy is terminated by a double EOF, the second EOF is detected but is not transferred to lfn_2. That is, if the files are not rewound after the copy (x parameter not specified), file lfn_1 is positioned after the second EOF and lfn_2 after the first EOF.

COPYBF CARD

The COPYBF control card causes a specified number of binary files to be copied from one file to another.

The control card format is:

COPYBF(lfn_1, lfn_2, n, c)

lfn_1 Name of the file to copy from; if this parameter is omitted, file INPUT is assumed.

lfn_2 Name of the file to copy to; if this parameter is omitted, file OUTPUT is assumed.

n Number of files (decimal) on lfn_1 to copy; if this parameter is omitted, n=1 is assumed.

c If a fourth parameter (1 to 7 alphanumeric characters) is present, the copy to or from an SI, S, or L format tape is performed in coded rather than binary mode.

The copy begins at the current position of lfn_1. If lfn_1=lfn_2, n files are skipped but no data transfer occurs. If the EOI is encountered before the file count is satisfied, an EOF is written on lfn_2, and the operation terminates.

COPYBR CARD

The COPYBR control card causes a specified number of binary records to be copied from one file to another.

The control card format is:

COPYBR(lfn_1, lfn_2, n, c)

lfn_1 Name of the file to copy from; if this parameter is omitted, file INPUT is assumed.

lfn_2 Name of the file to copy to; if this parameter is omitted, file OUTPUT is assumed.

n Number of records (decimal) to copy; if this parameter is omitted, n=1 is assumed.

c If a fourth parameter (1 to 7 alphanumeric characters) is present, the copy to or from an SI, S, or L format tape is performed in coded rather than binary mode.

The copy begins at the current position of lfn_1. EOF indicators are considered separate records and are included in the record count. If lfn_1=lfn_2, n records are skipped but no data transfer occurs. If the EOI is encountered before the record count is satisfied, an EOF is written on lfn_2, and the operation terminated.

COPYCF CARD

The COPYCF control card directs the system to copy a specified number of files from one file to another.

The control card format is:

COPYCF(lfn_1, lfn_2, n,fchar, lchar)

lfn_1	Name of the file to copy from; if this parameter is omitted, file INPUT is assumed.
lfn_2	Name of the file to copy to; if this parameter is omitted, file OUTPUT is assumed.
n	Number of files (decimal) to copy; if this parameter is omitted, n=1 is assumed.
fchar	First character position of each line to copy; if this parameter is omitted, fchar=1 is assumed.
lchar	Last character position of each line to copy; if this parameter is omitted, lchar=136 is assumed.

The copy begins at the current position of lfn_1. If lfn_1=lfn_2, n files are skipped but no data transfer occurs. If the EOI is encountered before the file count is satisfied, an EOF is written on lfn_2, and the operation terminates. COPYCF reformats the file into line images if it is blocked in greater than lchar blocks.

If lchar is less than fchar, lchar is greater than 150, or either fchar or lchar is unrecognizable, the following error message is issued to the user's dayfile.

ILLEGAL CHARACTER NUMBER.

If COPYCF is attempted on a line longer than 150 characters which does not contain a line terminator, the following message is issued.

NO LINE TERMINATOR.

If n is illegal or zero, the following message is issued.

ILLEGAL COUNT.

COPYCR CARD

The COPYCR control card directs the system to copy a specified number of records from one file to another.

The control card format is:

COPYCR(lfn_1, lfn_2, n, fchar, lchar)

lfn_1	Name of the file to copy from; if this parameter is omitted, file INPUT is assumed.
lfn_2	Name of the file to copy to; if this parameter is omitted, file OUTPUT is assumed.

n	Number of records (decimal) to copy; if this parameter is omitted, n=1 is assumed.
fchar	First character position of line to copy; if this parameter is omitted, fchar=1 is assumed.
lchar	Last character position of line to copy; if this parameter is omitted, lchar=136 is assumed.

The copy begins at the current position of lfn_1. If lfn_1=lfn_2, n records are skipped but no data transfer occurs. EOF indicators are considered separate records and are included in the record count. If the EOI is encountered before the record count is satisfied, an EOF is written on lfn_2, and the operation terminates. COPYCR is processed in exactly the same manner as the COPYCF control card except that n specifies the number of records rather than the number of files.

If COPYCR is attempted on a line longer than 150 characters which does not contain a line terminator, the following message is issued.

NO LINE TERMINATOR.

COPYEI CARD

The COPYEI control card directs the system to copy one file to another.

The control card format is:

COPYEI(lfn_1, lfn_2, x, c)

lfn_1	Name of the file to copy from; if this parameter is omitted, file INPUT is assumed.
lfn_2	Name of the file to copy to; if this parameter is omitted, file OUTPUT is assumed.
x	If a third parameter (1 to 7 alphanumeric characters) is present, both files are rewound before the copy, and rewound, verified, and rewound again after the copy is complete.
c	If a fourth parameter (1 to 7 alphanumeric characters) is present, the copy to or from an SI, S, or L format tape is performed in coded rather than binary mode.

The copy begins at the current position of lfn_1 and continues until the EOI is encountered. The EOI is not defined for certain tape formats (refer to Data Formats, section 10).

COPYSBF CARD

The COPYSBF control card enables the user to copy a file where the first character of each line is not a printer control character and is to be printed.

The control card format is:

COPYSBF(lfn_1, lfn_2, n)

lfn_1	Name of the file to copy from; if this parameter is omitted, file INPUT is assumed.
lfn_2	Name of the file to copy to; if this parameter is omitted, file OUTPUT is assumed.
n	Number of files (decimal) to copy; if this parameter is omitted, n=1 is assumed.

The COPYSBF routine copies n files beginning at the current position of lfn_1 to file lfn_2, shifting each line image one character to the right and adding a leading space. A page eject character is inserted at the beginning of each logical record (refer to appendix A for a list of carriage control characters). If lfn_1=lfn_2, n files are skipped but no data transfer occurs. If the EOI is encountered before the file count is satisfied, an EOF is written to lfn_2, and the operation terminates.

If COPYSBF is attempted on a line longer than 150 characters which does not contain a line terminator, the following message is issued.

NO LINE TERMINATOR.

COPYSBF results in single-line spacing and a page eject at the beginning of each record when the file is printed.

COPYX CARD

The COPYX control card enables the user to specify certain conditions when copying logical records.

The control card format is:

COPYX(lfn_1, lfn_2, x, b, c)

lfn_1 — Name of the file to copy from; if this parameter is omitted, file INPUT is assumed.

lfn_2 — Name of the file to copy to; if this parameter is omitted, file OUTPUT is assumed.

x — Copy specifications; if omitted, one record is copied. The value for x may be one of the following:

x	Meaning
n	Number of records (decimal) to copy
00	Copy all records up to and including first zero-length record
name	Copy all records up to and including record of specified name (record name is first seven characters of record)
type/name	Copy all records up to and including record of specified type and name (refer to CATALOG control card for list of valid record types)

b — Backspace control; if omitted, 0 is assumed.

b	Meaning
0	No backspace
1	Backspace file lfn_1 one record after copy completes
2	Backspace file lfn_2 one record after copy completes
3	Backspace files lfn_1 and lfn_2 one record after copy completes

c — If a fifth parameter (1 to 7 alphanumeric characters) is present, the copy to or from an SI, S, or L format tape is performed in coded rather than binary mode.

The COPYX routine copies logical records from file lfn_1 to file lfn_2 at the current position of lfn_1 until the condition specified by x is met. It then backspaces the files according to the value specified by the b parameter. If lfn_1=lfn_2, the file is repositioned according to the x parameter; no data is transferred.

DISPOSE CARD

The DISPOSE control card is used to release specified files to the proper output queues.

The control card format is:

DISPOSE(lfn_1=q_1, lfn_2=q_2, . . . , lfn_n=q_n/ot=usernum)

lfn_i	Name of the file to be disposed
q_i	Queue type:
	PR Print
	PH Punch coded O26
	P9 Punch coded O29
	PB Punch binary
	P8 Punch 80-column binary
ot	Origin type to which files are to be disposed:
	BC Local batch
	EI Remote batch (Export/Import)
usernum	Number of the remote batch (that is, ot is EI) user to which the files are to be disposed (ignored if ot is BC). This parameter is valid only if the user is allowed deferred batch jobs. Also, usernum must match the number of the user performing the DISPOSE on all character positions except those containing an *.

The file type for file lfn_i is changed to q_i in the FNT/FST entry for lfn_i. The system then processes the file according to queue type. The user can dispose coded punch files to either O26 or O29 regardless of the job's initial keypunch mode. If the system cannot recognize q_i, the following message is issued.

ILLEGAL DISPOSE CODE.

If the ot and usernum parameters are not specified, a remote batch job disposes the files to the 200 user terminal from which it was submitted, and all other origin types dispose the files to the central site output device. If ot is BC, the usernum parameter is ignored and the files are disposed to the central site device.

DOCMENT CARD

The DOCMENT control card enables the user to extract either the external or internal documentation from a file containing COMPASS source code.

The control card format is:

DOCMENT($p_1, p_2, \ldots, p_n$)

p_i The parameters can be in any order and must be in one of the following forms.

Omitted	The first default value is assumed.
a	The alternate default value is assumed.
a=x	x is substituted for the assumed value.

Any numeric parameter can be specified with a post radix character of either B or D. The values that p_i can assume are:

I=lfn_1 Name of the file that contains the page footing information; this must be a single card in the following format.

Column(s)	Contents
1	Blank
2-45	Document title
46-55	Publication number
56-60	Revision level
61-70	Revision date

S=lfn_2 Name of the file containing the source card images from which to extract the documentation

L=lfn_3 Name of the file on which the file is to be written

N=nn Number of copies to be produced

T=type Documentation type:

INT	Internal documentation (detailed description of the internal features of the software)
EXT	External documentation (detailed description of the external features of the software)

C=cc Key character for documentation

P=pp Number of print lines per page

NT Negate table generator

TC List table of contents

The following are the default values for the parameters described.

Parameter	First Default	Alternate Default	Comment
I	0	INPUT	Page footing information; if I is 0, no footing information is printed.
S	COMPILE	SOURCE	Source card images
L	OUTPUT	OUTPUT	List file
N	1	1	Number of copies (decimal)
T	EXT	INT	Documentation type
C	*	03	Check character (two octal digits)
P	60	80	Number of print lines per page
NT	ON	OFF	Table generator status
TC	OFF	ON	Table of contents status

Refer to appendix C, volume 2 for a detailed explanation of the documentation standards followed. It also contains examples of external and internal documentation for program COPYB.

EVICT CARD

The EVICT control card releases file space for a specified file(s) but does not release file attachment to the job.

The control format is:

EVICT(lfn_1, lfn_2, . . . , lfn_n)

lfn_i Name(s) of the file(s) to be evicted

The operation that EVICT performs depends on the file type. For permanent files, all file space except the first track is released, job attachment remains, and an EOI is written on the first sector of the first track. For all other file types, file space is released and job attachment remains. Also, all files for which write lockout is set are returned to the system. An EVICT of a tape file performs the same function as a RETURN except that EVICT cannot be used to decrease the number of tape units scheduled via the RESOURC card.

GTR CARD

The GTR control card provides directives for specifying certain records to be copied from one file to another.

The control card format is:

GTR(lfn_1, lfn_2, D, NR, S)selection directives

The parameters must be entered in the order shown; they are defined as follows:

lfn_1	File which is searched for the selected records; if this parameter is omitted, file OLD is assumed.
lfn_2	File on which the selected records are written; if this parameter is omitted, file LGO is assumed.
D	If specified, a directory record (OPLD type) is written at the end of lfn_2. In this case, lfn_2 must be a mass storage file. This parameter has special meaning for ULIB type records, as follows: If D is omitted, the first record of the user library, that is, the directory record (UPLD), is not copied to lfn_2; the last record (OPLD type) is copied but is not altered. If D is specified, the first record of the user library (UPLD) is copied to lfn_2, but is not altered, and an additional record, a new directory for the file (OPLD type), is added to lfn_2.
NR	If specified, files lfn_1 and lfn_2 are not rewound before or after the operation. If not specified, lfn_1 and lfn_2 are rewound both before and after the operation.
S	lfn_1 is processed as a sequential file; no attempt is made to read a directory.
selection directives	The user can specify the record types and names that he wants retrieved; these can be:

type/name	Retrieves record of specified type and name (refer to CATALOG control card for a list of valid record types). The record name is the first seven characters of the record.
name	Retrieves the record specified; the type is either TEXT or the type specified previously. If name=*, all records of the specified type are retrieved.
0	Inserts a zero-length record on file lfn_2.
type/$name_1$-$name_2$	Retrieves records $name_1$ through $name_2$ of type specified.

GTR searches file lfn_1 for the records specified by the selection directives. The selected records are then copied to file lfn_2 starting at the current EOI. Note that blanks are not legal between the terminator and the selection directives.

Examples of the use of this control card are:

- GTR(SYSTEM, BIN, D)PP/*

  All records of type PP are retrieved from file SYSTEM and copied to file BIN. A directory is built and placed as the last record on file BIN.

- GTR(OPL, NEW, , NR)OPLC/COMCARG, 0, COMCCIO

  Record COMCARG (type OPLC) is retrieved from file OPL and written on file NEW beginning at the current EOI. Then a zero-length record is written on file NEW. Finally, record COMCCIO (also type OPLC) is retrieved from file OPL and written on file NEW at its current position. File OPL is not rewound either before or after the operation.

- GTR(SYSTEM, SYSLIB, D)ULIB/SYSLIB

  The record named SYSLIB (type ULIB) is retrieved from file SYSTEM and copied to file SYSLIB. The D parameter must be specified to copy the ULIB directory (UPLD) of a ULIB record set. If the D parameter were omitted, the UPLD record would be skipped.

LIBEDIT CARD

The LIBEDIT control card specifies directives for editing and replacing binary records on a file with records from one or more correction files.

The control card format is:

LIBEDIT($p_1, p_2, \ldots, p_n$)

p_i Any of the following parameters in any order:

I=lfn_1	Directives comprise the next record on file lfn_1.
I=0	No directive input.
I omitted	Directives are on file INPUT.
P=lfn_2	File lfn_2 contains the old program library.
P=0	No old program library file.
P omitted	Old program library is on file OLD.
N=lfn_3	New program library will be written on file lfn_3.
N=0	Illegal; no error message is issued, if used.
N omitted	New program library will be written on file NEW.
L=1	Short correction listing (includes only directives, modifications, and errors) on the file specified by the LO parameter.
L=0	No output is listed.
L omitted	Full correction listing is written on the file specified by the LO parameter.
LO=lfn_4	List output on file lfn_4
LO omitted	List output on file OUTPUT.
B=lfn_5	Use file lfn_5 for the replacement file.
B=0	Do not use a default replacement file.
B omitted	Use file LGO as the default replacement file.
C	Copy the new library file over the old library file after processing.
C omitted	Do not copy the new library file over the old library file after processing.

R	Do not rewind library files after processing.
R omitted	Rewind old and new library files after LIBEDIT and VFYLIB processing.
V	Call VFYLIB after LIBEDIT processing.
V omitted	Do not call VFYLIB to verify libraries after LIBEDIT processing.
D	Ignore errors and continue.
D omitted	Do not ignore errors; abort job.

For a description of the LIBEDIT directives, refer to appendix C.

LIBGEN CARD

The LIBGEN control card allows the user to generate a user library file.

The control card format is:

LIBGEN($p_1, p_2, \ldots, p_n$)

p_i Any of the following in any order:

F=lfn_1	Name of source file containing records to be placed on user library file lfn_2.
F	System assumes source file LGO.
F omitted	System assumes source file LGO.
P=lfn_2	Name of the file on which the library is to be written.
P	System assumes library to be written on ULIB.
P omitted	System assumes library to be written on ULIB.
N=lfn_3	Name of the user library being generated; this name becomes the name of the ULIB and OPLD records.
N	System assumes lfn_3=lfn_2.
N omitted	System assumes lfn_3=lfn_2.
NX=n	If n is nonzero, no cross references are given. That is, decks are not cross-linked in the ULIB directory. This can be used to avoid duplicate entry points on loads.
NX omitted	The system assumes n=0.

LIBGEN processes the source file specified and generates a library file on the file specified with the P parameter. The library is given the name specified with the N parameter. If the F and P options specify the same file, the message:

FILE NAME CONFLICT.

is issued.

LIBGEN rewinds and scans the source file and builds a directory of all entry points, program names, and external references for records in the file. When an EOF mark appears, LIBGEN terminates the directory and rewinds lfn_1. LIBGEN then copies lfn_1 to lfn_2, adding the library and directory records. The directory is written as the first record of the new file. It is indicated as a user library type record by a 76 identification table. The identification table also contains the name of the library.

The directory contains all external references within the library and the linkage to routines that reside in the library. This indicates which routines must be loaded when routines from this library are loaded. This means that all externals for routines in a library are automatically satisfied from that library first.

The entire file follows the directory record on the new file. The file index is the last record on the file. This record contains random addresses for each record in the file. The index record has a table identifier of 7000_8. LIBGEN processes only REL and COS type records, bypassing all other record types.

For example, file RELB contains routines that are used at execution time for several application programs. It is desirable to load these routines as needed when executing the application programs. To generate the library, the following control statement:

```
LIBGEN(F=RELB, P=MYLIB, N=APPLIB)
```

is entered. This creates user library APPLIB on file MYLIB. If FORTRAN application programs are compiled using the control card:

```
FTN.
```

the user library can be used by loading the program in the following manner.

```
LDSET(LIB=MYLIB/RUNLIB)
LOAD(LGO).
EXECUTE.
```

This causes the program to be loaded and executed with externals satisfied first from user library MYLIB, then from user library RUNLIB, and finally from the system default library SYSLIB.

LIST80 CARD

The LIST80 routine reads a file containing COMPASS source code and compresses it to 80 columns, which fits on 8-1/2 by 11-inch printer paper.

The control card format is:

LIST80(lfn_1, lfn_2, NR)

lfn_1	File to copy from; if this parameter is omitted, file LIST is assumed.
lfn_2	File to copy to; if this parameter is omitted, file OUTPUT is assumed.
NR	If this parameter is specified, lfn_1 is not rewound.

LOCK CARD

The LOCK control card enables the user to prevent writing on a file.

The control card format is:

LOCK(lfn_1, lfn_2, . . . , lfn_n)

lfn_i	Logical file name of a local file

With the LOCK card, the user can set the write interlock bit in the FNT/FST entry for a local file. Subsequently, the system allows only read operations on the file. The file specified must be a local file; if it is not, the following message is issued.

ILLEGAL FILE TYPE.

The LOCK card may also be used in conjunction with the COMMON card to lock local files before making them library files for multiple user access. Refer to Library Files in section 2 and the COMMON control card.

LO72 CARD

The LO72 control card allows the user to specify the reformatting of his files.

The control card format is:

LO72(p_1, p_2, . . . , p_n)

p_i Any of the following parameters in any order:

I	Reformat parameters are on file INPUT.
I=lfn_1	Reformat parameters are on file lfn_1.
I=0	There is no input file of reformat parameters. If the I parameter is omitted, I=0 is assumed.
S	Data to be reformatted is on file SCR.
S=lfn_2	Data to be reformatted is on file lfn_2. If the S parameter is omitted, SCR is assumed.
L	Reformatted data is listed on file OUTPUT.
L=lfn_3	Reformatted data is listed on file lfn_3. If the L parameter is omitted, OUTPUT is assumed.
T	File to be reformatted is of type B.
T=x	File to be reformatted is of type x, where x is:

M Modify source data

C COMPASS source data

B Other source data

If the T parameter is omitted, B is assumed.

H	Number of characters per output line is 72.
H=xxx	Number of characters per output line is xxx (maximum allowed is 150 characters). If the H parameter is omitted, 72 is assumed.

NOTE

H must be greater than or equal to the number of characters being moved (Nx) plus the starting column number of the destination field (Ox).

LP	Output is formatted for the line printer.
NR	Output file is not rewound.
Nx=y	Specifies the number of characters to be moved (up to 6 fields):

x(1 to 6)	Number of the field being moved
y	Number of characters being moved

Note: N1+N2+N3+N4+N5+N6 must be less than or equal to the number of columns specified in the H parameter.

Ix=y	Specifies the field the data originates from:

x(1 to 6)	Number of the field being moved
y	Starting column of originating field

Ox=y	Specifies the destination field the data is going to:

x(1 to 6)	Number of the field to receive data
y	Starting column of destination field

The following table shows the default values assumed for the N, O, and I parameters for the various source types.

Type	N1	I1	O1	N2	I2	O2	N3	I3	O3
B	72	1	1	0	0	0	0	0	0
C	7	9	1	50	41	8	15	112	58
M	2	6	1	48	10	3	22	82	51

The remaining parameters of these types are defaulted to 0.

LO72 reformats files (output files in general). The user can rearrange each line (all lines must be formatted the same) in the format he chooses. All default values compress output to 72 columns, which is appropriate for terminal output or 8-1/2 by 11-inch printer paper. If a 1 is encountered in column 1 (the page eject printer control character), the next two lines of source data are processed as a two-line header. This header is compressed to 72 columns for all source types. If no page eject control characters are encountered, no headers are processed.

The following values apply to the first line of header and cannot be changed.

N1=42, I1=8, O1=0 (if LP not specified; otherwise, O1=1)

N2=20, I2=90, O2=42

N3=5, I3=115, O3=62

N4=5, I4=121, O4=67

The subheader lines for COMPASS and Modify listings are processed uniquely.

For B listings, the following values apply to the reformatting.

N1=43, I1=8, O1=0 (if LP not specified; otherwise, O1=1)

N2=29, I2=70, O2=43

All parameters are passed to LO72 by the control card. If an input file is specified, LO72 reads it for additional input parameters. If the job originates from a time-sharing terminal, the user is asked if he wishes to change any of the input parameters. If he enters YES, the system prints the current parameter values and allows him to change them individually. Pressing the carriage return key for any parameter leaves the parameter at its former value. In the following examples, the same input parameters are entered in three possible ways.

Control Card

LO72(I=0,S=SOURCE,T=C,L=OUT,N4=1,I4=2,O4=75,H=90)

Time-Sharing Terminal: (underlined data is entered by user)

```
/LO72
DO YOU WANT TO CHANGE ANY CONTROL ARGUMENT VALUES -
ENTER:  YES OR NO
? YES
ARGUMENT                        VALUE
INPUT FILE NAME:                                  ? *CR*
SOURCE FILE NAME:               SCR               ? SOURCE
OUTPUT FILE NAME:                                 ? OUT
SOURCE FILE TYPE:                                 ? C
OUTPUT LINE LENGTH:             72 CHARS.         ? 90
        NO. OF      MOVED FROM                    MOVED TO
         CHARS.        COLUMN                      COLUMN
(X)      (NX)          (IX)                        (OX)
1.        7             9                           1
2.       50            41                           8
3.       15           112                          58
4.        0             0                           0
5.        0             0                           0
6.        0             0                           0
ENTER CHANGES IN THE FOLLOWING FORMAT:
NX=AA*CR*
IX=BB*CR*
OX=CC*CR*
ETC.
TO CONTINUE, ENTER *CR* ONLY.
? N4=1   *CR*
? I4=2   *CR*
? O4=75   *CR*
? *CR*
```

Input File

```
S=SOURCE, L=OUT, T=C.
N4=1, I4=2, O4=75
H=90.
-EOR-
```

Each line in the input file must end with a terminator.

NEW CARD

The NEW control card creates a primary file.

The control card format is:

NEW(lfn/ND)

lfn	Name of file to be made primary file
ND	If this parameter is specified, current working files are not released

The NEW card creates an empty file and makes it the user's new primary file. Any currently existing primary file is released.

Note that all current working files are released unless the ND parameter is specified.

OUT CARD

The OUT control card is used to release output files from the control point to the output queue.

The control card format is:

OUT.

The only files released are those having the names:

OUTPUT
PUNCH
PUNCHB
P8

This control card is used if the user wishes to initiate printing or punching of the files before job termination. The PUNCH file is punched in either O26 or O29 mode depending on the origin of the job. If the job is a local batch job, the coded deck is punched in the initial keypunch mode of the job's control card record. For all other job origin types, the coded file is punched in the system default keypunch mode.

PACK CARD

The PACK control card allows the user to pack a specified file and copy it to another. The control card format is:

PACK(lfn_1, lfn_2, x)

lfn_1	Name of file to be packed
lfn_2	Name of file to receive packed data
x	If a third parameter (1 to 7 alphanumeric characters) is specified, lfn_1 is not rewound before the pack occurs.

The input file, lfn_1, may consist of any number of records and/or files. If no third parameter is supplied, lfn_1 is read from the BOI to the EOI, and all EOR and EOF marks are removed. It is written to file lfn_2 at the current position as one record. File lfn_2 is rewound after the pack; lfn_1 is not. If lfn_2 is not specified, file lfn_1 is packed to itself.

The programmer should note that problems may arise when using PACK with direct access files. For example, if file A resides on a legal direct access file device and the following cards are submitted:

```
PACK(A)
DEFINE(A)
```

PACK may copy file A to a device which does not support direct access files. In this event the DEFINE card would then cause the job to abort and the following message to be issued to the user's dayfile:

```
DIRECT ACCESS DEVICE ERROR, AT nnn.
```

when nnn is the file environment table (FET) address.†

The user can avoid this situation by defining file A as an empty direct access file, creating the file, and then packing it.

```
DEFINE(A)
create file A
PACK(A)
```

The following error messages may be issued to the user's dayfile in response to a PACK card.

Message	Description
PACK PARAMETER ERROR.	The PACK control card contains an error.
ILLEGAL INPUT FILE.	An attempt was made to pack a file that is assigned to a time-sharing terminal (for example, file INPUT for time-sharing origin jobs represents data typed at the terminal keyboard, and therefore cannot be packed).
ILLEGAL CIO REQUEST.	An attempt was made to pack a nonmass storage file.
WRITE ON READ-ONLY FILE fff, AT nnn.	The direct access file was not attached in write mode (refer to the ATTACH control card).

† Refer to Permanent File Manager, section 5, volume 2.

PRIMARY CARD

The PRIMARY control card makes a local file the primary file.

The control card format is:

PRIMARY(lfn)

lfn	Name of local file

The file to be made primary must be a local mass storage file. Any currently existing primary file (other than the lfn specified) is released. If the specified file is already primary, the operation is ignored.

The primary file is rewound after every operation performed on that file. Therefore, the file manipulation cards BKSP, SKIPEI, SKIPF, SKIPFB, and SKIPR cannot be used to position within the file. The user should also remember that the primary file is rewound after the completion of any of the COPY cards. An attempt to add to the file using one of the COPY cards may result in writing over existing data at the BOI.

RENAME CARD

The RENAME control card allows the user to change the name of a local file.

The control card format is:

RENAME($nlfn_1$=$olfn_1$, $nlfn_2$=$olfn_2$, . . . , $nlfn_n$=$olfn_n$)

$nlfn_i$	New name of the local file
$olfn_i$	Existing name of the local file

The RENAME control card is used to change the name of the file $olfn_i$ to $nlfn_i$ in the FNT/FST. This does not change the names of files in the permanent file system.

If a file by the name $nlfn_i$ already exists, it is returned to the system. Under certain conditions the system also changes the file type of $olfn_i$ to that of the file which was returned.

- If $olfn_i$ is a local mass storage file and the returned file was a print, punch, or primary type file, $olfn_i$ is renamed and its file type is changed to that of the returned file.
- If $olfn_i$ is a local mass storage file and the returned file was not a print, punch, or primary type file, $olfn_i$ is renamed but its file type is not changed.
- If $olfn_i$ is not a local file or does not reside on mass storage, an

  ILLEGAL FILE TYPE.

  error message is issued.

For example, the user has only two files assigned to his job. File A is a local mass storage file and file B is a print type file. If the user issues the following request

RENAME(X=A)

file A is renamed file X and its file type (local) is not changed. However, if the user issues the request:

RENAME(B=A)

file B is returned to the system; file A is renamed file B and changed to a print type file.

REQUEST CARD

The REQUEST control card enables the user to assign a file to a device by including in the comment field a description of an acceptable device.

The control card format is:

REQUEST(lfn, $\begin{matrix}CK\\CB\end{matrix}$)

lfn	Name of the file to be assigned to the specified equipment.
CK	Specifies that lfn is to be used as a checkpoint file. Each time a checkpoint dump is taken, the new information is written at the previous EOI of lfn.
CB	Specifies that lfn is to be used as a checkpoint file. Each time a checkpoint dump is taken, the new information is written at the BOI of lfn.

The descriptive comment is displayed at the system console, directing the operator to make the requested assignment.

If lfn already exists when the REQUEST is made, no new assignment is made and job processing continues with the next control card. However, the user can reassign lfn by issuing a RETURN on the file before making the REQUEST.

Any user, regardless of his validation, may use the REQUEST card to assign a file to a mass storage device. However, to assign a file to a nonmass storage device, the user must be validated to use nonallocatable devices.† If the user does not have this validation and attempts to request a nonmass storage device, the system aborts his job.

If lfn is to be used for checkpoint dumps, either the CK or CB keyword is specified. These keywords are used in conjunction with the CKP and RESTART control cards; they allow the user to:

- Save all checkpoint dumps by appending each dump to the checkpoint file:

  REQUEST(lfn, CK)

- Save the last checkpoint dump by writing each dump at the beginning of the checkpoint file:

  REQUEST(lfn, CB)

- Save two consecutive checkpoint dumps by alternately writing on two checkpoint files:

  REQUEST(lfn_1,CB)

  REQUEST(lfn_2, CB)

---

† Refer to LIMITS control card, section 6.

If the CK parameter is specified for alternate files or if more than two checkpoint files are specified, the job is aborted and the following message is issued to the user's dayfile.

CHECKPOINT FILE ERROR.

The CK and CB parameters specify a checkpoint file that is local to the job. The user can make the checkpoint file permanent by placing a DEFINE card† before the REQUEST.

DEFINE(lfn)

REQUEST(lfn, CK)

CKP.

The user is not required to supply a REQUEST card to define a checkpoint file. He can use an ASSIGN or LABEL card, or he can use default values.

If no REQUEST card specifying a checkpoint file has been detected when the first CKP card is encountered, the system requests a device for the user, specifies a file name of CCCCCC, and selects the CK option. For a subsequent restart job, however, the system assumes the user has made the checkpoint file available.

The REQUEST card can also be used to create or access existing 7- or 9-track unlabeled tapes. For a description of the card as it applies to magnetic tape assignment, refer to Tape Management, section 10.

---

† Any mass storage file used as a checkpoint file must have write permission.

RESEQ CARD

The RESEQ control card is used to resequence source files which have leading sequence numbers.

The control card format is:

RESEQ(lfn, t, xxx, yy)

lfn	Name of the file to be resequenced		
t	Type of file:		
		B	BASIC source code
		T	Text source information; a five-digit sequence number plus a blank is added at the beginning of each line; the file text, however, is not inspected
		other or omitted	Any number at the beginning of a line is considered a sequence number and is resequenced according to the xxx and yy parameters; numbers are added to lines where no leading sequence numbers are present. This option can be used with time-sharing FORTRAN statements.
xxx	New line number of the first statement; if this parameter is omitted, the system assumes xxx=100		
yy	Increment to be added to xxx for each succeeding line number; if this parameter is omitted, the system assumes yy=10.		

Files which have leading sequence numbers include Time-Sharing FORTRAN and BASIC source files. If the file has no leading sequence numbers, five-digit numbers are attached to the beginning of each line. If the line number encountered or required exceeds 99999, the following message is issued.

LINE NUMBER LIMIT EXCEEDED.

Some BASIC statements reference the sequence numbers which must also be changed; therefore, it is imperative that the user specify the proper file type (t). When errors occur while resequencing a BASIC program, the following message is issued for all lines containing errors.

ERROR AT LINE xxx.

RETURN CARD

The RETURN control card releases the specified file from job attachment and/or releases its file space.

The control card format is:

RETURN(lfn_1, lfn_2, . . . , lfn_n)

lfn_i Name(s) of the file(s) to be returned

The operation performed depends on the file type.

Type	Operation
Input	The file name is changed to INPUT*. File space is not released; INPUT* remains attached to the job as a local file (refer to Input File Control in section 3 for further information).
Print	Job attachment and file space are released.
Punch	Job attachment and file space are released.
Local	Job attachment and file space are released.
System	Job attachment is released but file space remains.
Library	Job attachment is released but file space remains.
Primary	Job attachment and file space are released.
Permanent	Write interlock is cleared. Job attachment is released but the file space remains.

In addition, the RETURN card can be used to decrease the number of tapes or packs scheduled for the job via the RESOURC control card.

REWIND CARD

The REWIND control card causes files to be rewound and positioned to the BOI (or beginning-of-reel for magnetic tape files).

The control card format is:

REWIND(lfn_1, lfn_2, . . . , lfn_n)

lfn_i	Name(s) of file(s) to be rewound

If the previous operation on the magnetic tape file was a write, a REWIND card causes the following operations to be performed.

1. If the tape is ANSI labeled, the system writes a tape mark, an EOF1 label, and three tape marks and then rewinds the tape.
2. If the tape is unlabeled and the data format specified on the ASSIGN, LABEL, or REQUEST card is X, S, L, E, B, or F, the system writes four tape marks and then rewinds the tape.
3. If the tape is unlabeled and the data format is I or SI, the system writes a tape mark, an EOF1 label, and three tape marks and then rewinds the tape.

Refer to Tape Management Control Cards in section 10 for further information about tape files and to appendix G for a description of EOF1 and EOV1 labels.

SETID CARD

The SETID control card assigns a new identification code for the specified file.

The control card format is:

SETID(lfn_1=x_1, lfn_2=x_2, . . . , lfn_n=x_n)

lfn_i	Logical file name
x_i	New identification code for the file (0 through 67_8). This code must match the device identification code specified in the EST.

The identification code allows the user to route his file to an output device or device group with the same identification code. This is useful when a print file requires special forms.

The file lfn_i must be an input (INFT), local (LOFT), print (PRFT), or punch (PHFT) type file or the following message is issued.

ILLEGAL FILE TYPE.

SKIPEI CARD

The SKIPEI control card directs the system to position the specified file at the EOI.

The control card format is:

SKIPEI(lfn)

lfn	Name of the file to be positioned

On magnetic tapes where no EOI is defined, the operation stops at an EOF.

The SKIPEI card has no effect on a primary file since the file is rewound after every operation.

SKIPF CARD

The SKIPF control card directs the system to bypass, in a forward direction, the specified number of files from the current position of the named file.

The control card format is:

SKIPF(lfn, n, m)

lfn	Name of the file to be positioned
n	Number (decimal) of files to be skipped; if the parameter is omitted, the system assumes n=1.
m	File mode; C for coded, B for binary. If omitted, the system assumes the file is in coded mode.

If an EOI is encountered before n files are bypassed, file lfn remains positioned at the EOI.

The SKIPF card has no effect on a primary file since the file is rewound after every operation.

SKIPFB CARD

The SKIPFB control card directs the system to bypass, in the reverse direction, the specified number of files from the current position of the named file.

The control card format is:

SKIPFB(lfn, n, m)

lfn	Name of the file to be positioned
n	Number (decimal) of files to be skipped; if the parameter is omitted, the system assumes n=1.
m	File mode; C for coded, B for binary. If omitted, the system assumes the file is in coded mode.

The system does not backspace past the beginning -of- information (BOI), in the event that BOI is encountered before n files are bypassed.

The SKIPFB card has no effect on a primary file since the file is rewound after every operation.

SKIPR CARD

The SKIPR control card directs the system to bypass, in a forward direction, the specified number of logical records from the current position of the named file.

The control card format is:

SKIPR(lfn, n, ℓ , m)

lfn	Name of the file to be positioned
n	Number (decimal) of records to be skipped; if this parameter is omitted, the system assumes n=1.
ℓ	EOR level; $0<\ell<17$. If $0<\ell<16$, the system assumes $\ell=0$. If $\ell=17$, n indicates the number of files to skip rather than records.
m	File mode; C for coded, B for binary. If omitted, the system assumes the file is in binary mode.

EOR marks are considered separate records and included in the record count. If the EOI is encountered before n records are bypassed, file lfn remains positioned at the EOI.

The SKIPR card has no effect on a primary file since the file is rewound after every operation.

SORT CARD

The SORT control card enables the user to sort a file of line or card images in numerical order based on leading line numbers consisting of a specified number of digits.

The control card format is:

SORT(lfn, NC=n)

lfn	Logical file name of the file to be sorted; lfn may be a local file or a direct access permanent file.
n	Number of leading line number digits the file is to be sorted on; $n \leq 10$. If the NC parameter is omitted, the system assumes n=5.

In the case of duplicate line numbers, all lines other than the first are considered correction lines. All lines with the same number are deleted from the file except the last line encountered.

For input from a time-sharing terminal, SORT deletes a line or card image if a line number is followed by an empty line or a line number is followed by a blank and a carriage return.

For batch input, SORT deletes a card or line image if a card containing only the line number is submitted.

If a line number contains more than n digits, the user can delete the line either by entering the first n digits of the line number and pressing the carriage return (terminal input) or by submitting a card containing only the first n digits of the line number (batch input).

After the sort, lfn is packed and set at EOI.

The following SORT error messages may be issued to the user's dayfile.

Message	Description
NO LINE NUMBER ON SORT FILE	A line on the input file is missing a line number or a line exceeded the 150-character limit.
ILLEGAL SORT PARAMETER.	The SORT control card is in error.
EMPTY SORT INPUT FILE.	File lfn contains no data.
WRITE ON READ-ONLY FILE fff AT nnn.	The direct access input file was not attached in write mode (refer to the ATTACH control card).

STAGE CARD

The STAGE control card causes files to be copied from the specified device to a file residing on mass storage.

The control card format is:

STAGE(lfn, p_1, p_2, . . . , p_n)

lfn	Name associated with file to be staged from magnetic tape to mass storage
p_i	Any of the following in any order:
	NR — Do not rewind lfn before beginning operation; default is rewind.
	NU — Do not unload lfn after staging operation; default is automatic unload.
	DR — Drop job after staging operation.
	N=n — Copy n files to lfn.
	T=xx — Stage file lfn from device with EST ordinal xx.† This parameter is specified only when tape containing files to be staged is unlabeled (X format and system default density).
	VSN=vsn — Specifies the 1- to 6-character volume serial number of the labeled tape containing the file to be staged
	D=den — Tape density:
	200 — 200 bpi (implies 7-track)
	556 — 556 bpi (implies 7-track)
	800 — 800 bpi/cpi (7- or 9-track)
	1600 — 1600 cpi (implies 9-track)
	F=format — Data format (refer to section 9):
	I — Internal
	X — External
	SI — SCOPE Internal
	MT — 7-track tape (default)
	NT — 9-track tape

If T is not included but VSN is included, n files are copied from the specified tape. If neither T nor VSN is included, a request for lfn is issued to the operator. If DR is not included, STAGE requests the next set of parameters for the next staging operation to be entered by the K display on the system console. When lfn is staged to mass storage, it is designated as a library file. If a library file already exists with the same name as the file being staged, the system issues the following message.

DUPLICATE COMMON FILE NAME.

---

†Contact installation personnel for a list of EST ordinals.