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

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

FIGURES

TABLES

VOLUME 2

FIGURES

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.

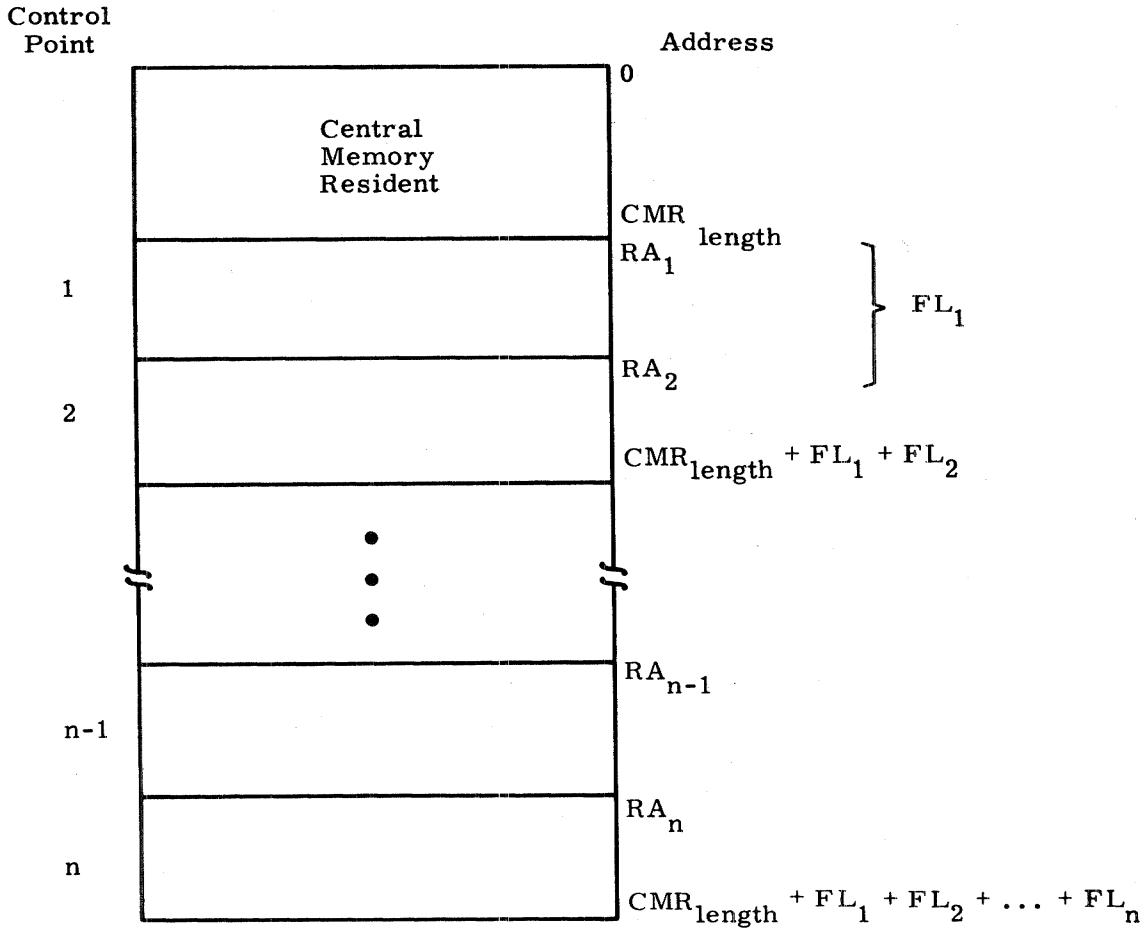


Figure 1-1-1. Central Memory Layout

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.

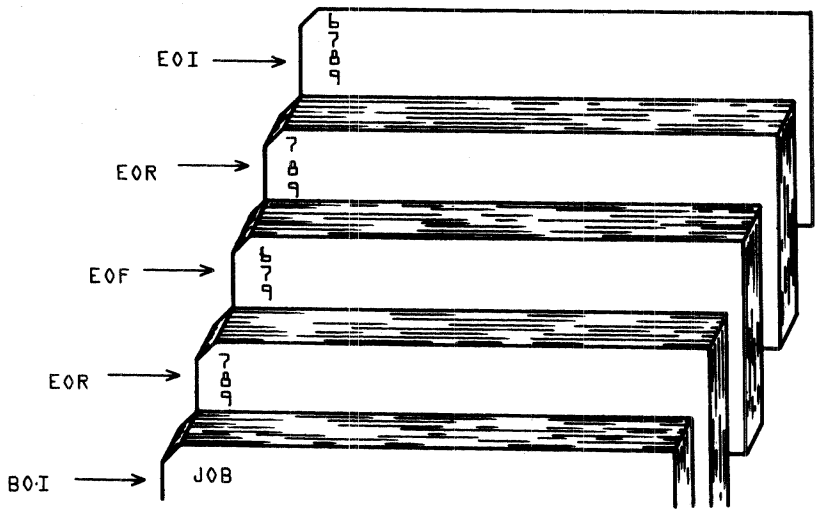


Figure 1-2-1. Sample Card File Structure

† 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.

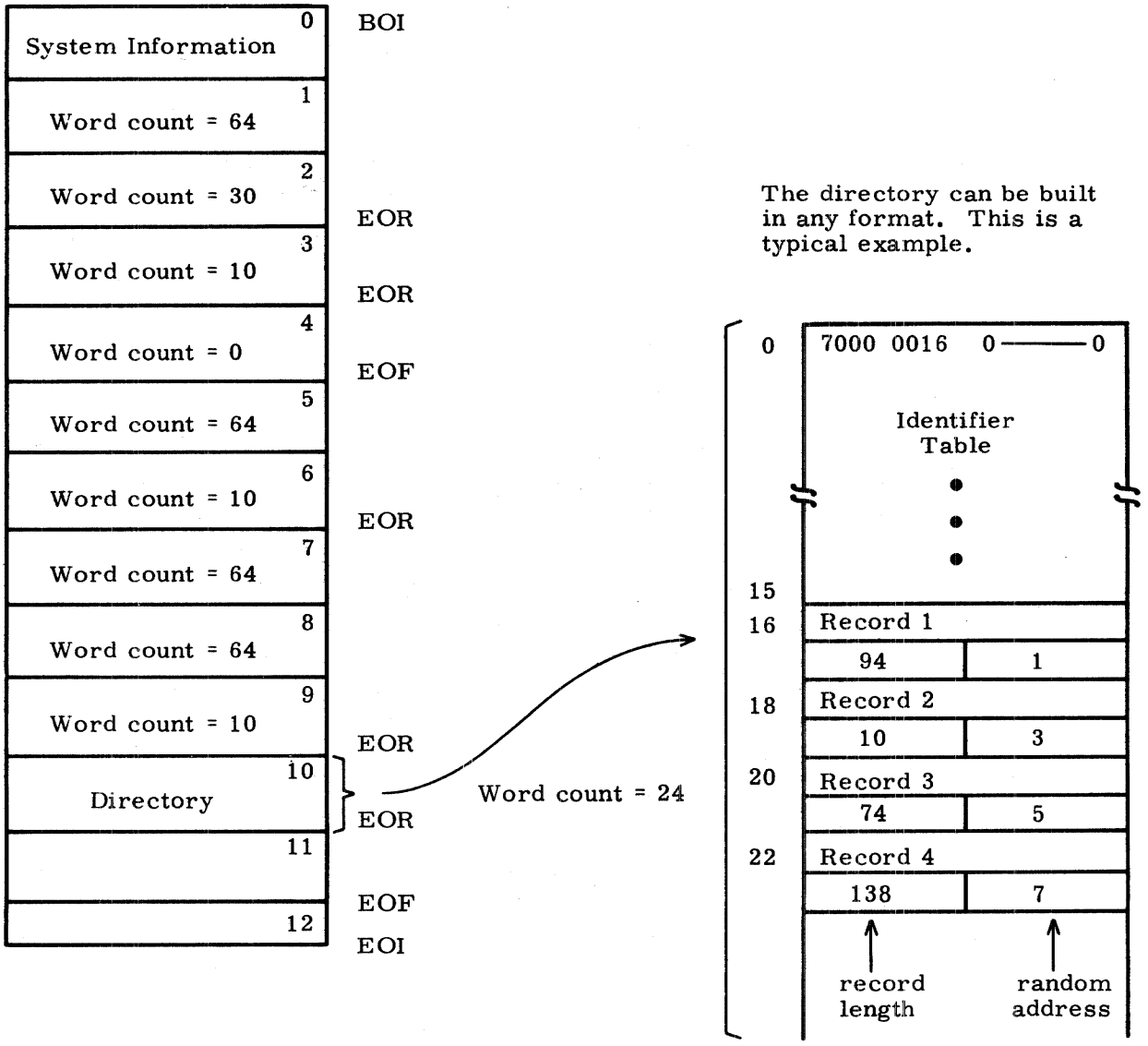


Figure 1-2-2. Sample Random Access File Format

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.

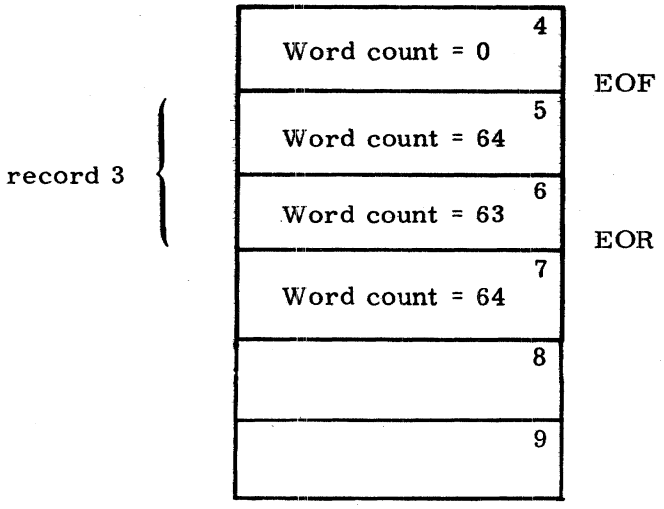
In summary, to access record 3 sequentially, four PRUs must be read. To access the record randomly, only three PRUs are read: two PRUs to position for the directory and one PRU to read the directory.

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.

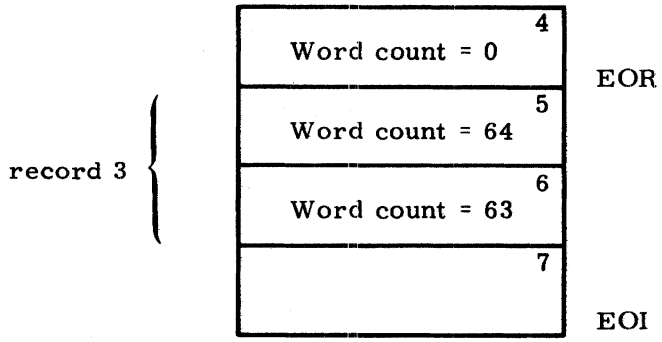
The directory can be placed anywhere in the file. The only requirement is that those users who wish to access the file randomly know where to position the file in order to read the directory. However, the directory usually precedes the EOF/EOI.

WRITING FILES

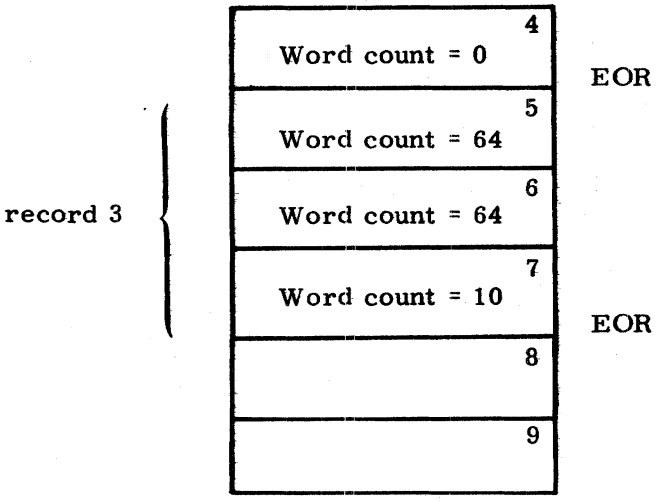
After reading and modifying record 3 of the sample file, the user may wish to rewrite the record in the file. If the modifications have not changed the number of PRUs required, a write operation can be used to replace the existing record with the modified record. This write operation must be issued as 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.

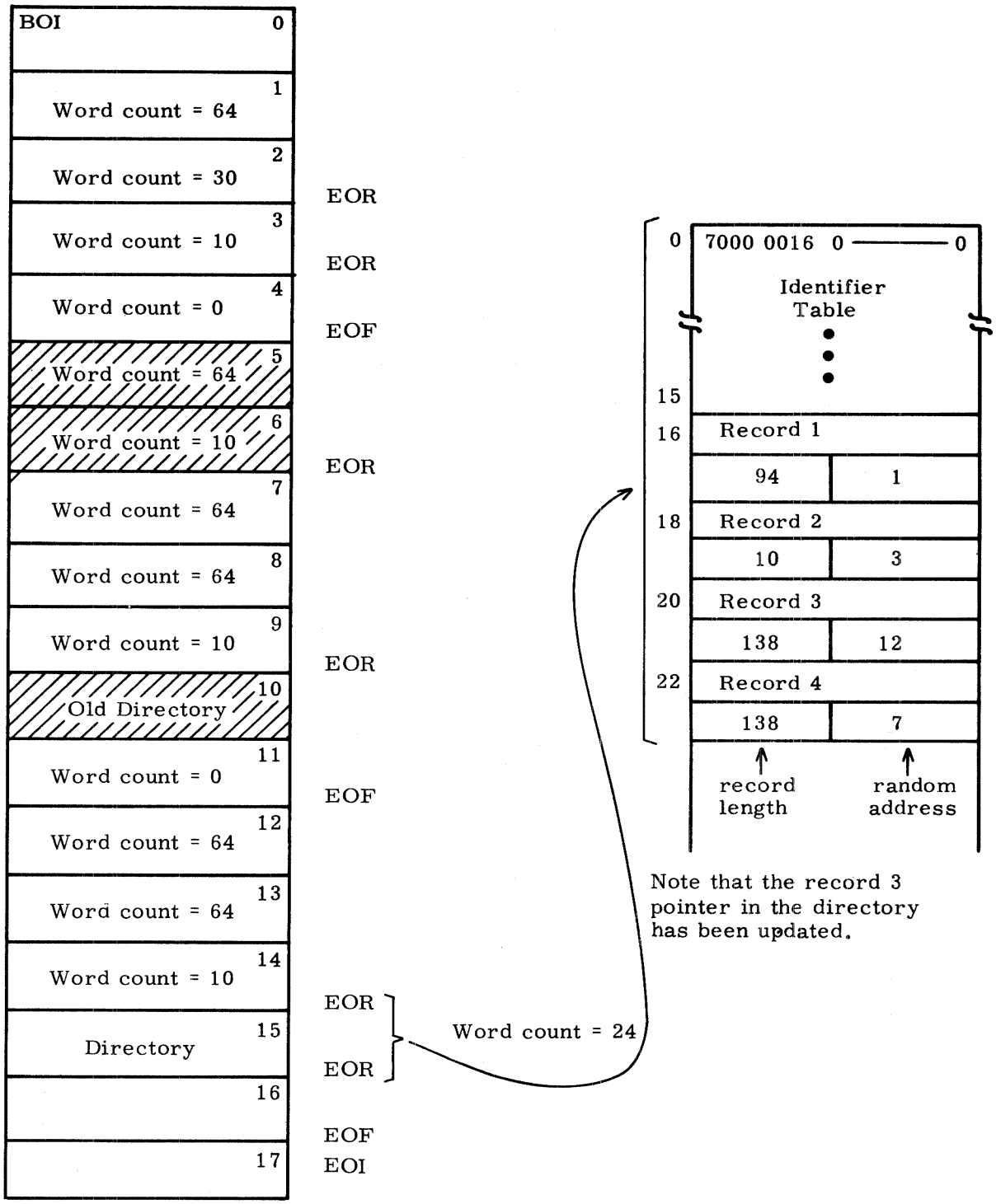


Figure 1-2-3. Modified Sample Random Access File

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.

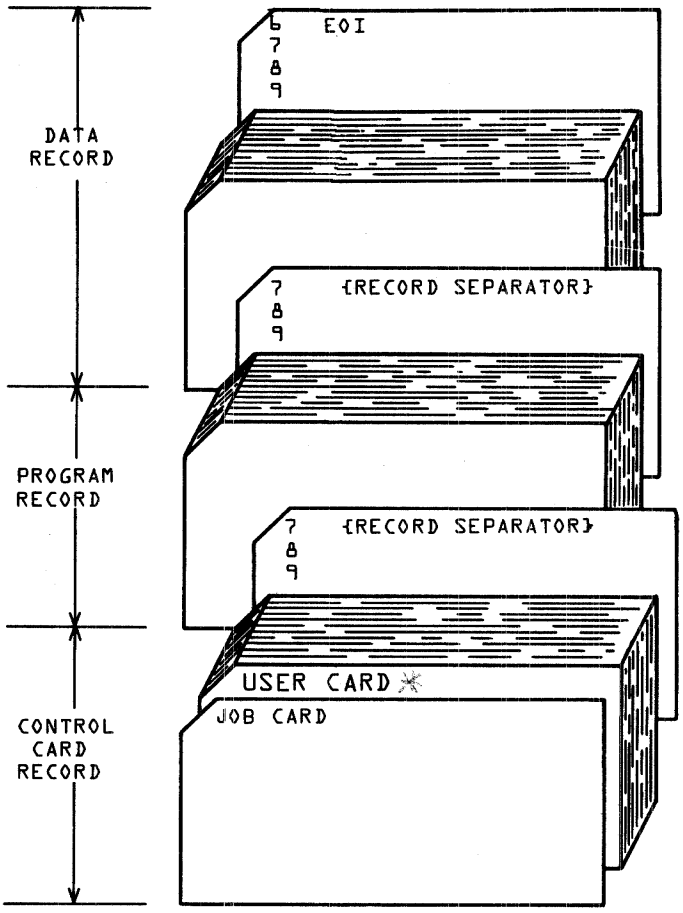


Figure 1-3-1. Basic Job Deck

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.

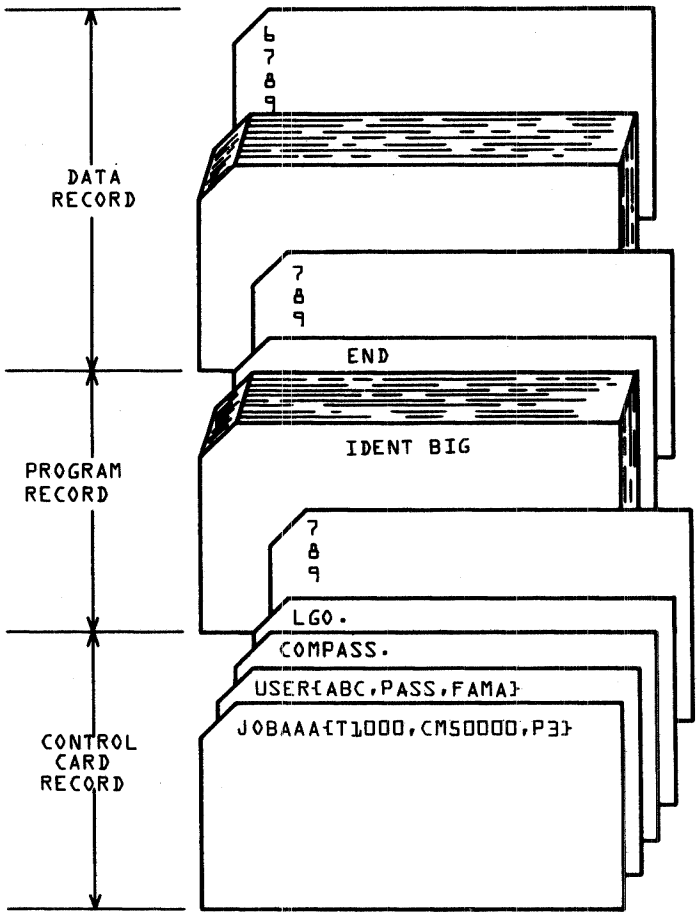


Figure 1-3-2. COMPASS Assemble and Execute Deck

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.

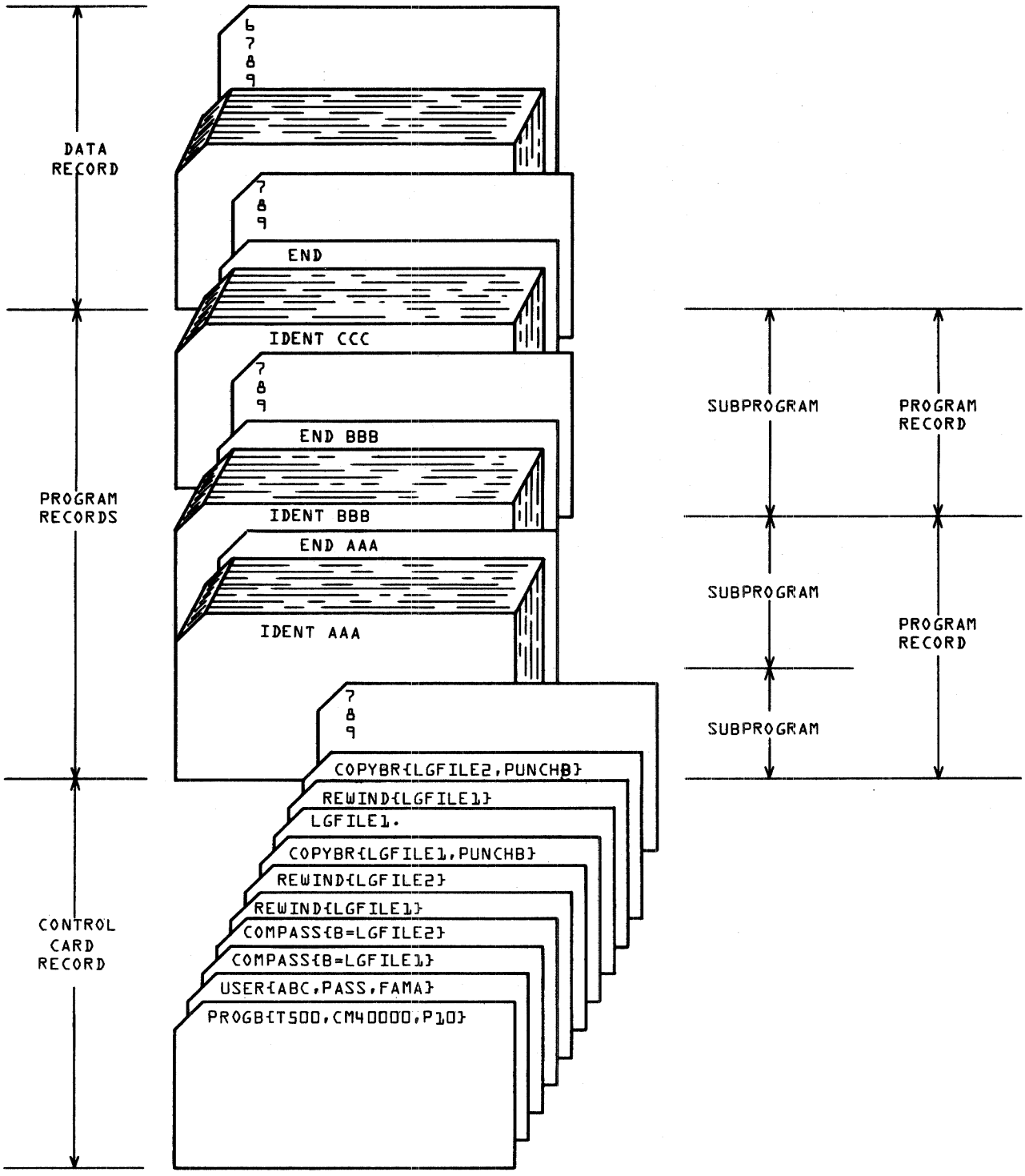


Figure 1-3-3. COMPASS Assemble, Execute, and Punch Binary Deck

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

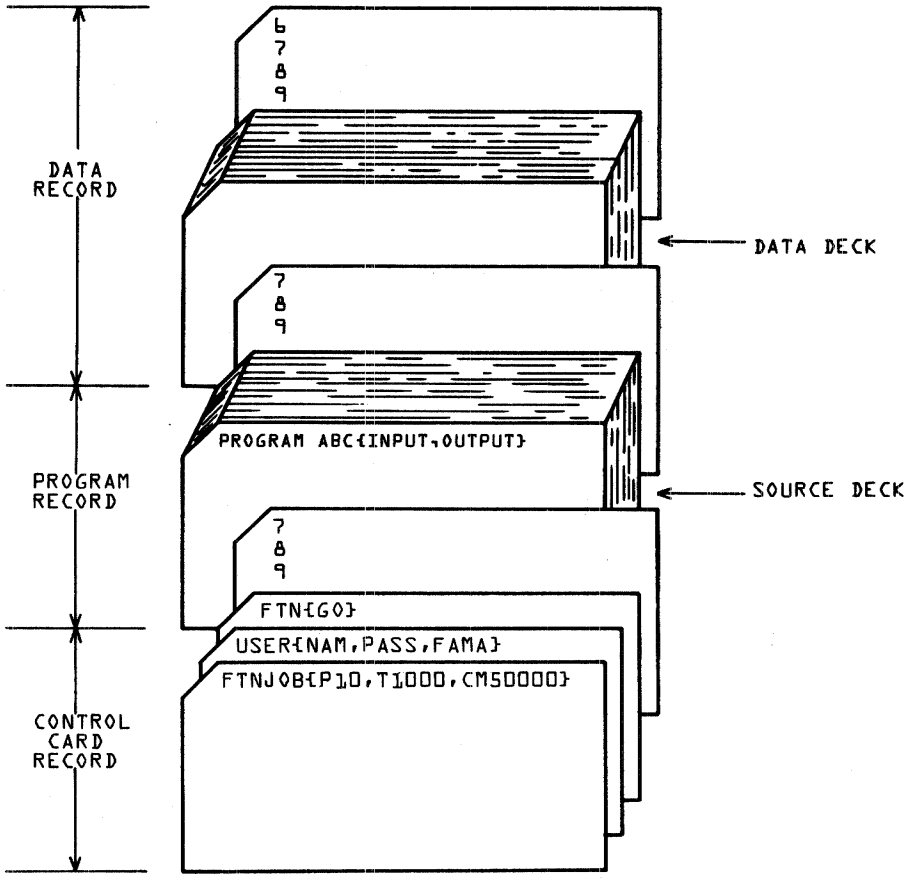


Figure 1-3-4. FORTRAN Compile and Execute Deck

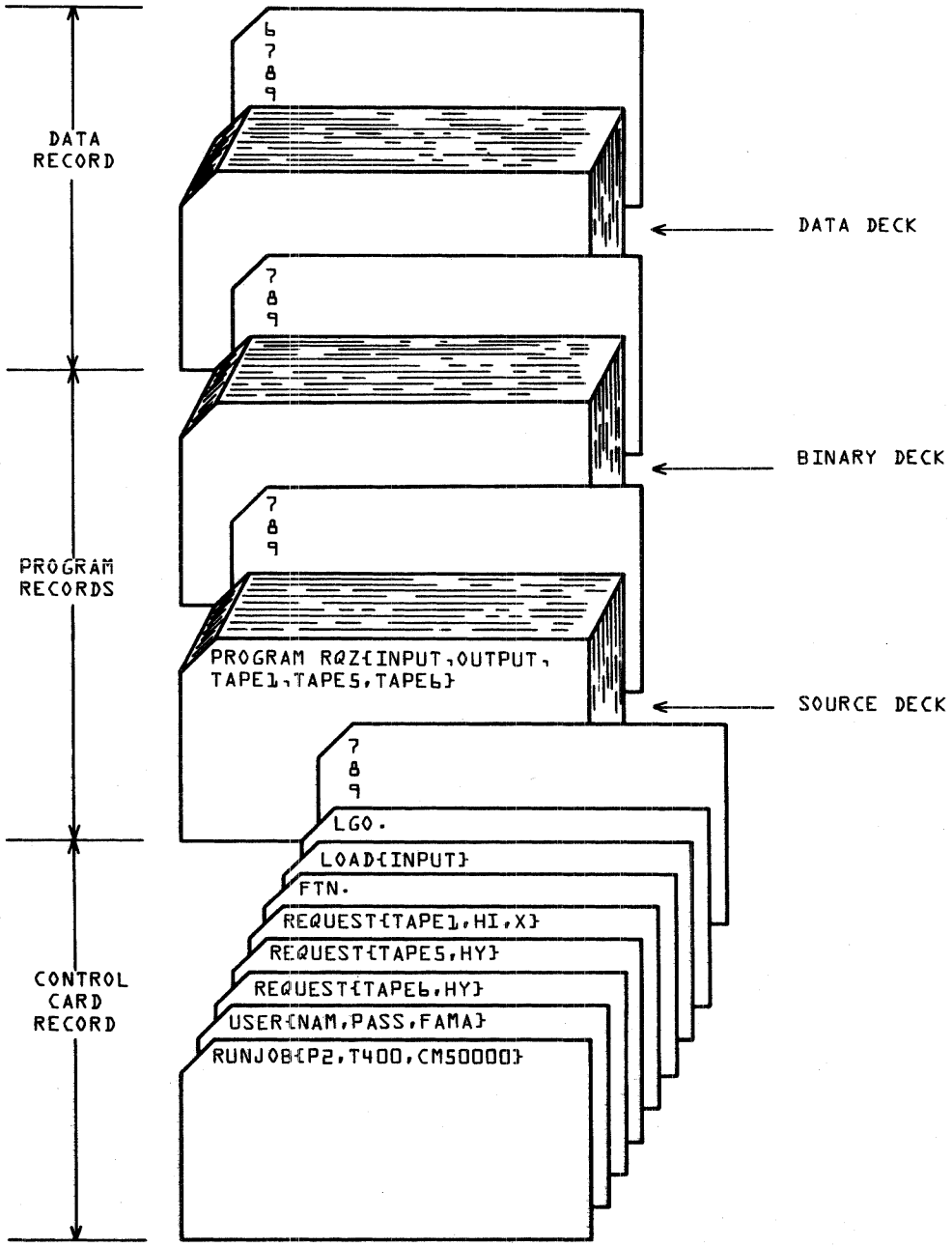


Figure 1-3-5. FORTRAN Load and Run Deck

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.

CONTROL CARD PROCESSING 5

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:
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

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

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	B=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.

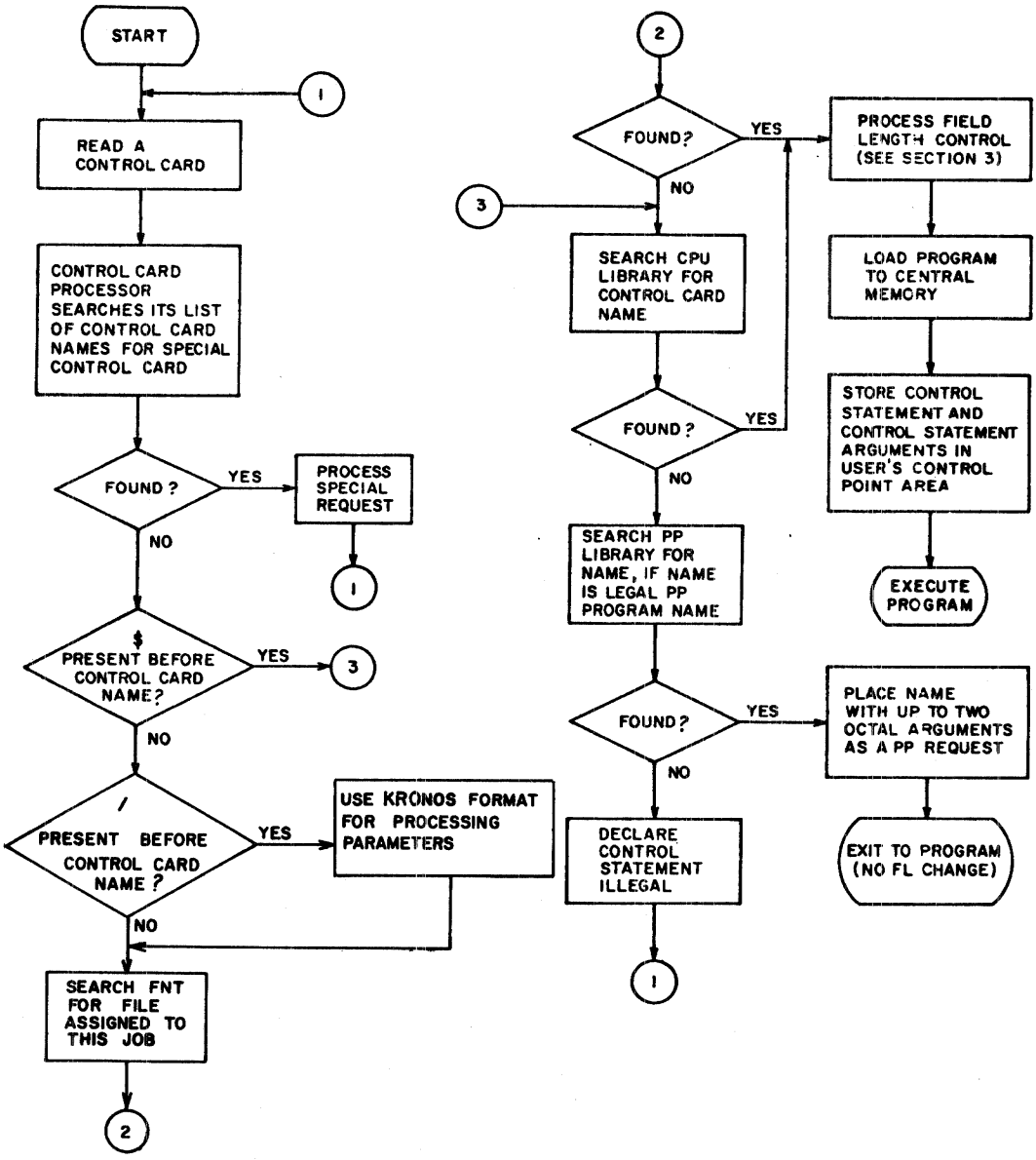


Figure 1-5-1. Control Card Processing Flow

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=$p_1p_2...p_n$, JN=jobname, FN=lfn_1, 0=lfn_2)

or

ENQUIRE($p_1p_2...p_n$)

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 ($0\text{-}67_8$); 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=0000000000000000215

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 nonsharable 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.
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 }
PG	Programmer Number } IUPUI use
PW	Password }
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, $p_1, p_2, \ldots, p_n$)

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.

REC	CATALOG OF SYSTEM NAME	TYPE	FILE LENGTH	1 CKSUM	DATE	COMMENTS
301	PROFILE PROFILE ARG= RFL= SSJ=	ABS	5112	7106	75/04/22.	73/05/24.
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($lfn_1, lfn_2, \ldots, lfn_n$)

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($lfn_1, lfn_2, \ldots, lfn_n$)

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

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($lfn_1, lfn_2, \ldots, lfn_n$)

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($lfn_1=x_1, lfn_2=x_2, \ldots, lfn_n=x_n$)

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.

TDUMP CARD

The TDUMP control card lists a file in octal and/or alphanumeric form.

The control card format is:

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

p_i	Any of the following in any order:	
	I=lfn_1	Input file name (default is TAPE1)
	L=lfn_2	Output file name (default is OUTPUT)
	O	Octal dump only (default is octal and alphanumeric dump)
	A	Alphanumeric dump only (default is octal and alphanumeric dump)
	R=rcount	Number of records in decimal to dump (default is dump to EOI)
	F=fcount	Number of files in decimal to dump (default is dump to EOI)
	N=lines	Maximum number of lines in decimal that can be dumped (if N is omitted, there is no restriction on the number of lines).
	NR	Do not rewind file lfn_1 before dump (default is to rewind lfn_1).

The user has the option of dumping the entire file or of specifying the number of records, files, or lines to dump.

UNLOAD CARD

The UNLOAD control card releases job attachment and/or the file space of the specified file.

The control card format is:

UNLOAD($lfn_1, lfn_2, \ldots, lfn_n$)

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

The UNLOAD card performs the same function as the RETURN control card; for additional information, refer to the description of the RETURN card earlier in this section). Unlike the RETURN card, an UNLOAD of a magnetic tape file cannot be used to decrease the number of tape units scheduled for the job via the RESOURC control card. For magnetic tape files, if the previous operation was a write, the UNLOAD 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 unloads 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 unloads 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 unloads the tape.

Refer to Tape Management Control Cards, section 10, for further information about tape files and to appendix G for a description of an EOF1 label.

UNLOCK CARD

The UNLOCK control card rescinds the LOCK command and clears the write interlock bit for the specified file.

The control card format is:

UNLOCK($lfn_1, lfn_2, \ldots, lfn_n$)

lfn_i Name(s) of local file(s)

The file must be a local file; if it is not, the following message is issued.

ILLEGAL FILE TYPE.

VERIFY CARD

The VERIFY routine performs a binary comparison of all data from the current position of the files specified.

The control card format is:

VERIFY(lfn_1,lfn_2,p_1,p_2,...,p_n)

lfn_1	Name of the first file; if this parameter is omitted, the system assumes TAPE1.
lfn_2	Name of the second file; if this parameter is omitted, the system assumes TAPE2.
p_i	Any of the following in any order:

N=0	Verify terminates on the first empty file encountered on either file.
N=x	Verify x files; default is N=1.
N	Verify terminates when end of information is encountered on either file.
E=y	List the first y errors encountered on the comparison. If E is omitted, the system assumes E=100.
L=lfn_3	List errors on file lfn_3. If L is omitted, the system assumes L=OUTPUT.
A	Abort if errors occur.
R	Rewind both files before and after the verify.

Whenever words on the two files do not match, VERIFY lists the:

- Record number
- Word number within the record
- Words from both files that do not match

If errors are encountered, the following message is issued to the user's dayfile.

VERIFY ERRORS.

If any pair of lfn_1, lfn_2, and lfn_3 are identical, the following message is issued.

FILE NAME CONFLICT.

VFYLIB CARD

The VFYLIB program performs a compare on binary records from the current position of the specified files.

The control card format is:

VFYLIB(lfn_1, lfn_2, lfn_3, NR)

lfn_1	Name of the first file; if this parameter is omitted, the system assumes OLD.
lfn_2	Name of the second file; if this parameter is omitted, the system assumes NEW.
lfn_3	Name of the file to receive output; if this parameter is omitted, the system assumes OUTPUT.
NR	If specified, lfn_1 and lfn_2 are not rewound.

The VFYLIB program lists:

- Replacements
- Deletions
- Insertions

on the output file lfn_3. A program is defined as being replaced when the actual binary code is changed. Information in the prefix (77) table such as last modification date and last assembly date are skipped in VFYLIB's comparison.

WRITEF CARD

The WRITEF control card directs the system to write a specified number of file marks on the named file.

WRITEF(lfn, x)

lfn	Name of the file to be written on
x	Number of file marks to be written; if this parameter is omitted, the system assumes x=1.

WRITER CARD

The WRITER control card directs the system to write a specified number of empty records on the named file.

The control card format is:

WRITER(lfn, x)

lfn	Name of the file to be written on
x	Number of empty records to be written; if this parameter is omitted, the system assumes x=1.

PERMANENT FILE CONTROL CARDS 8

The permanent file control cards allow the user to utilize the permanent file system.† The control cards included in this category are:

APPEND	DEFINE	PERMIT	SAVE
ATTACH	GET	PURGALL	
CATLIST	OLD	PURGE	
CHANGE	PACKNAM	REPLACE	

The cards described in the following section allow the user to create permanent files (DEFINE) and make local files permanent (SAVE, REPLACE). These files can be accessed (ATTACH, GET), added to (APPEND), and destroyed (PURGE, PURGALL). Requests are directed to a specified auxiliary device by the PACKNAM card. Certain parameters can be changed with the CHANGE card without attaching and redefining the file or retrieving and saving it.

Information on permanent files is obtained through the CATLIST card. Part of that information is the permission status of the user as granted by another user by means of the PERMIT card.

The following pages list options available on the control cards. Unless otherwise stated, the options described apply to all of the permanent file control cards. For a detailed description of permanent file structure, refer to section 2. Errors encountered during permanent file control card processing cause error messages to be issued to the user's dayfile. For a description of these messages, refer to appendix B.

† The batch user is unable to access permanent files unless he has included a USER card in the job deck.

Keyword	Option	Description
UN=	usernum	Alternate user number. This parameter is necessary only if the permanent file involved resides in another user's catalog. To be able to access other catalogs, the user must be granted explicit permission (refer to the PERMIT control card), the file must be a semiprivate or public file, or the user must have automatic permission. A user has automatic permission to files in catalogs of other users if his user number contains asterisks, and all nonasterisk characters match the other user's user number.
PW=	passwrd	The user has the option of specifying a 1-to-7-character password for a file. This password must be specified whenever alternate users access the file.
CT=	ct	Permanent files fall into three categories which specify the method of access. This option must be selected when the file is saved or defined. The categories are:

P or PRIVATE	Private files are available for access only by the originator or those to whom the originator has explicitly granted permission (refer to the PERMIT control card).
S or SPRIV	Semiprivate files are available for access by all users who know the file name, user number, and password. The system records in the originator's catalog the user number of each user who accessed the file, the number of accesses, and the date and time of the last access.
PU or PUBLIC	Public files† are available for access by all users who know the file name, user number, and password. The system records the number of times the file was accessed but does not record user numbers or the last access date and time.

†CT=LI can also be used to specify public files.

Keyword	Option	Description
M=	m	File or user permission mode:
	W or WRITE	Allows the user to write, read, append, execute, modify, and/or purge the file. This mode can be specified for direct or indirect access files.
	M or MODIFY	Allows the user to modify, append, read, and/or execute a direct access file. Adding new information within the existing boundaries of the file is legal but the file size must be maintained.
	A or APPEND	Allows the user to append information to the end (EOI) of the file. This mode can be specified for direct or indirect access files.
	R or READ	Allows the user to read and/or execute the file. This mode can be specified for direct or indirect access files.
	RM or READMD	Allows the user to read and/or execute a direct access file with the implication that another user may currently be accessing the same file in MODIFY mode. This mode can be specified only for direct access files.
	RA or READAP	Allows the user to read and/or execute a direct access file with the implication that another user may currently be accessing the same file in APPEND mode. This mode can be specified only for direct access files.
	E or EXECUTE	Allows the user to execute the file. If the file is attached to the user's job in EXECUTE mode, the file must be in absolute format. This mode can be specified for direct or indirect access files. Relocatable files with EXECUTE permission may be loaded and executed only via a stand-alone file name call (such as LGO), which is not preceded by a loader control.
	N or NULL	Removes permission previously granted via PERMIT control cards. This mode can be specified for direct or indirect access files.

Keyword	Option	Description
R=	r	Specifies the type of device on which the permanent file resides or is to reside; r can be any of the following.

r	Device
DE	Extended Core Storage
DIi	844 Disk Storage Subsystem ($1 \leq i \leq 8$)
DP	Distributive Data Path to ECS
MDi	841 Multiple Disk Drive ($1 \leq i \leq 8$)

The R keyword can be used in two ways.

1. It can be used on the DEFINE control card to specify the family device on which the direct access permanent file is to reside.
2. It can be used in conjunction with the PN and NA keywords on any permanent file control card (including DEFINE) to identify the auxiliary device on which the permanent file resides or is to reside. R is required only if the desired device has a device type different from that of the available device and the installation has defined the desired device as removable. If PN and NA are specified but R is not specified, the system default device type is used. If the specified device type cannot be recognized or does not exist in the system, the following message is issued to the user's dayfile.

   ILLEGAL DEVICE REQUEST, AT nnn.

S= space

Specifies the amount of space in decimal PRUs desired for the direct access file. Refer to the DEFINE control card.

PN= packname

A 1-to-7 character pack name used in conjunction with the R keyword to identify the auxiliary device to be accessed in the permanent file request. This parameter is specified only when the file to be accessed resides on an auxiliary device. If the device is currently not available and the NA keyword was not specified, the following message is issued to the user's dayfile.

DEVICE UNAVAILABLE, AT nnn.

An auxiliary device is a mass storage device that supplements the normal family of permanent file devices. A RESOURC control card must be included in any job that uses two or more disk packs concurrently.

Keyword	Option	Description
NA		The NA keyword can be used in two ways. 1. Normally, if the user attempts to access a file that is interlocked or if an error occurs in an attempt to process the file, the system aborts the job. With the NA option the user can bypass a job abort and continue processing. If lfn is busy and the NA option is specified on an ATTACH control card, the system automatically suspend the job until the file becomes available. If NA is specified and an error other than pfn BUSY occurs in processing file lfn_i, the system issues the appropriate error message to the user's dayfile and then continue with file lfn_{i+1}. If the error occurred on the last file specified on the card, the system continues with the next card. 2. If the user requests an auxiliary device that is currently not available, the system will abort his job. The NA keyword enables him to bypass this abort and direct the system to make the desired device available.
ND		The ND keyword prevents releasing of the user's working files upon processing of an OLD control card.

Several files can be accessed with one control card. A slash (/) is used to separate the files being accessed and the options described previously. The special options are order-independent and are indicated by the keywords described. If special options are specified on the control card, they apply to all files that appear on the card.

APPEND CARD

The APPEND control card allows the user to add supplementary information to an existing indirect access file.

The control card format is:

APPEND(pfn, lfn_1, lfn_2, . . . , lfn_n/PW=passwrd, UN=usernum, PN=packname, R=r, NA)

pfn	Name of the indirect access permanent file to which the local files are to be appended
lfn_i	Name(s) of local file(s) to be appended to pfn

The logical structure of the two files is retained; that is, EORs and EOFs are appended as well as data. If the file is appended to a file in an alternate user's catalog, a password must be supplied if one is required.

ATTACH CARD

The ATTACH control card allows a user to access a direct access file.

The control card format is:

ATTACH(lfn_1=pfn_1, lfn_2=pfn_2, . . . , lfn_n=pfn_n/UN=usernum, PW=passwrd, M=m, PN=packname, R=r, NA)

lfn_i	Local file name given to the direct access file while it is attached to the user's job. A working copy is not generated since user access is made directly to the permanent file. Thus, lfn_i is used only when it is desirable to reference the attached file by a name other than its permanent file name, pfn_i.
pfn_i	Name of direct access file to be attached. If pfn_i is omitted, the system assumes pfn_i=lfn_i.
m	File or user permission mode, where m can be W, M, A, E, R, RM, or RA. If m is omitted, the system assumes m is R. This option must be specified by all users, including the originator, if the file is to be modified or new information is to be added to the file. If pfn_i is attached in W mode, the date is recorded as last modification date even if the file was not altered.

A read/write interlock is provided to ensure that only one user at a time can access the file in write mode. Several users may access the file in read mode simultaneously. The user should return the file as soon as possible to enable other users to access the file. If lfn_i is present before this command is issued, it is returned to the system even if an error is encountered when processing the command.

The following tables show the resulting current access modes when the user attempts to attach an active file. (FB means the file is busy and cannot be attached at that time. Refer to the NA option.)

TABLE 1-8-1. RESPONSE TO CURRENT ACCESS WRITE/READ ACCESS DESIRED

	Current Access		
Access Desired	W	M	A
R	FB	FB	FB
E	FB	FB	FB
RM	FB	M/R	A/R
RA	FB	FB	A/R

TABLE 1-8-2. RESPONSE TO CURRENT ACCESS READ/READ ACCESS DESIRED

	Current Access			
Access Desired	R	RM	RA	Free
R	R	R	R	R
E	R	R	R	R
RM	R	RM	RA	RM
RA	R	RA	RA	RA

TABLE 1-8-3. RESPONSE TO CURRENT ACCESS READ/WRITE ACCESS DESIRED

	Current Access			
Access Desired	R	RM	RA	Free
W	FB	FB	FB	W
M	FB	M/R	FB	M
A	FB	A/R	A/R	A

The resulting current access listed in these tables should not be confused with the mode the user will have when requesting file access. For example, in Table 1-8-1, the user desires RM (read/allow modification) access and the current access is M (modify). The resulting current access (M/R) indicates that one user is modifying the file and at least one user is reading the file. The user requesting RM access is permitted read-only access.

Note that if an auxiliary device has been previously specified by a PACKNAM card, the system attempts to attach pfn_i from the auxiliary device rather than the normal system devices.

CATLIST CARD

The CATLIST control card lists information about the user's permanent files or those permanent files he can access in the catalogs of alternate users.

The control card format is:

CATLIST(LO=p, FN=pfn, UN=usernum, PN=packname, R=r, L=lfn, NA, DN=dn)

p — One of the following list options:

F — Selects a listing of pertinent information about each file in the user's catalog. If an alternate user number is specified (UN option), the user obtains a listing of all files that he can access in the alternate user's catalog. Note that the password for files in an alternate user's catalog is not included in the listing. The password to files in an alternate user's catalog must be obtained directly from that user.

FP — Selects a listing of permission information recorded for each alternate user of a specified file in the user's catalog. This option requires that a file name be specified (FN option). If an alternate user number is specified (UN option), only the permission information for that user of the specified file is listed.

The user numbers listed include those that have been granted explicit permission to the file (private file only) and those that have accessed the file because of implicit permission (semiprivate files only).†

† User numbers are not recorded for accesses to public files.

0 (zero) Selects a short list that includes only the names of the files in the user's catalog. If an alternate user number is specified (UN option), the user obtains only the names of the files that he can access in the alternate user's catalog. If no LO keyword is specified, the system assumes this value.

P Selects a short list that indicates only the user numbers of alternate users who have accessed the specified private or semiprivate file. This option requires that a file name be specified (FN option).

pfn Permanent file name. This option specifies that catalog information is desired only for this permanent file. This parameter is required when listing permit information (LO=FP, LO=P). If the short list options are selected (LO=0, LO=P), the message

pfn FOUND, AT nnn.

is issued if the file (or user number) is located. The message

pfn NOT FOUND, AT nnn.

is issued if the specified file (or user number) is not located.

usernum User number. This parameter has two purposes.

1. For LO=F and LO=0. Indicates the alternate catalog for which the user desires catalog information.
2. For LO=FP and LO=P. Indicates the permission information recorded for the specified alternate user.

packname This parameter specifies an auxiliary device that contains catalog information for all users with files on that device. The PN keyword must be specified if the user wishes to obtain the following information from his catalog on the specified auxiliary device.

- Pertinent information about each file (LO=F)
- Only the name of each file (LO=0)
- Permission information for each alternate user that has accessed a specific file (LO=FP)
- Only the user number of each alternate user that has accessed a specific file (LO=P)

The PN parameter can also be specified to allow alternate users to obtain a list of files they can access on the auxiliary device, as well as pertinent information about each file.

lfn Output file name. This is the name of a local file to which the CATLIST information is written. If this parameter is omitted, the system assumes L=OUTPUT. If lfn exists and is positioned at BOI, the contents of that file is purged before the CATLIST information is written. However, if lfn exists and is positioned at EOI, the CATLIST information is appended to the file as a new logical record.

NA	No abort option. CATLIST continues processing if errors are encountered during processing.
dn	Device number (0 through 77_8). List file residing on specified device number dn.

If no entries are present in the specified catalog, the message

EMPTY CATALOG.

is issued to the user's dayfile.

CHANGE CARD

The CHANGE control card allows the originator of a direct or indirect access file to alter any of several parameters without having to attach and redefine the file or retrieve and save it.

The control card format is:

CHANGE(nfn=ofn/CT=ct, M=m, PW=passwrd, PN=packname, R=r, NA)

nfn	New permanent file name
ofn	Old permanent file name. If no name change is desired, only ofn is specified.

The CT, M, and PW keywords should be specified only if a change in the value associated with that keyword is desired. To clear the password for an existing file, the user must set PW=0. The PN and R keywords cannot be used to specify a new auxiliary device. They are used only to specify the device on which ofn resides. CHANGE also updates the last modification date and last access date for the specified file.

The following messages may be issued to the user's dayfile in response to a CHANGE request.

Message	Description
ofn NOT FOUND, AT nnn.	The specified permanent file, ofn, was not found in the user's catalog.
nfn ALREADY PERMANENT, AT nnn.	The new permanent file, nfn, already exists in the user's permanent file catalog.

DEFINE CARD

The DEFINE control card allows the user to define direct access permanent files.

The control card format is:

DEFINE(lfn_1=pfn_1,lfn_2=pfn_2,...,lfn_n=pfn_n/PW=passwrd,CT=ct,M=m,R=r,
S=space,PN=packname,NA)

lfn_i	If DEFINE is to be used to create an empty direct access permanent file, lfn_i is specified only if the user desires to reference the file by a name other than its permanent file name. If DEFINE is to be used to define an existing local file as a direct access file, lfn_i is the name of the local file. Also, if lfn_i exists, its position is not altered.
pfn_i	Permanent file name. If pfn_i is omitted, the system assumes lfn_i=pfn_i.
r	Type of device on which the permanent file is to reside. The device must be a permanent file mass storage device on which direct access files are allowed.

The user can either create an empty permanent file or define an existing local file as a direct access file. If the user releases the file and wishes to access it at some time in the future, the ATTACH control card must be included.

If lfn_i does not exist, the device on which pfn_i resides depends on the r and space parameters.

r	space	Residency
Specified	Not specified	The file resides on the device of type r with the most space available.
Specified	Specified	The file resides on the device of type r with the most space available, provided that device has as many PRUs available as specified by the space parameter.
Not specified	Specified	The file resides on the device with the most space available, provided that device has as many PRUs available as specified by the space parameter.
Not specified	Not specified	The file resides on the device with the most space available.

If an auxiliary device has been previously specified by a PACKNAM card, pfn_i resides on that auxiliary device rather than a system device.

If the optional parameters are omitted, the system assumes the following values.

Keyword	Default
PW	None
CT	PRIVATE
M	WRITE
PN	None

If the S option is selected and no device has the specified amount of space available, the request is aborted and the following message is issued to the user's dayfile.

PRUS REQUESTED NOT AVAILABLE, AT nnn.

Unused space is not guaranteed to be available if the user attempts to expand the file at a later time.

If lfn_i already exists on a device other than that specified by r, or an illegal device is specified, the system issues the following message to the user's dayfile.

DIRECT ACCESS DEVICE ERROR, AT nnn.

GET CARD

The GET control card enables the user to retrieve a copy of file pfn_i for use as a local file.

The control card format is:

GET(lfn_1=pfn_1, lfn_2=pfn_2, . . . , lfn_n=pfn_n/UN=usernum, PW=password, PN=packname, R=r, NA)

lfn_i	Local file name given the file while in use
pfn_i	Permanent file name. If pfn_i is omitted, lfn_i=pfn_i

If the request is made with no parameters specified, the user's primary file is assumed.

Each pfn specified must be an indirect access file. File lfn_i is returned to the system if it is present before this command is issued even if an error is encountered in processing the command. The new file is rewound. No interlock is provided to prevent other users from obtaining working copies of the same file simultaneously. If the name of the user's current primary file is specified as an lfn, the corresponding pfn is made the new primary file and any subsystem associated with it becomes the user's new current time-sharing subsystem.

If the request is for a file in another user's catalog (UN option specified), the permission mode is that which the user has been permitted for private files or that specified in the catalog for semiprivate and public files.

If an auxiliary device has been previously specified by a PACKNAM card, the system attempts to retrieve the copy of pfn_i from the auxiliary device rather than the normal system devices.

OLD CARD

The OLD control card retrives a copy of a permanent file and makes it the primary file.

The control card format is:

OLD(lfn=pfn/UN=usernum, PW=passwrd, PN=packname, R=r, NA, ND)

lfn	Local file name given the file while in use
pfn	Permanent file name. If pfn is omitted, lfn=pfn.

The OLD card performs the same operation as the GET card, and additionally makes lfn the primary file. Any currently existing primary file is released. All working files are also released unless the ND parameter is specified.

If an auxiliary device has been specified previously by a PACKNAM card, the system attempts to retrieve the copy of pfn from the auxiliary device rather than the normal system devices.

PACKNAM CARD

The PACKNAM card directs subsequent permanent file requests to the specified auxiliary device.

The control card format is:

```
PACKNAM(PN=packname)
        or
PACK(packname)
```

packname — A 1- to 7-character name used to identify the auxiliary device to be accessed in subsequent permanent file requests

PACKNAM allows the user to omit the PN keyword from requests for files that reside on the specified device. However, if permanent files on another auxiliary device are to be accessed, the PN keyword can be specified in the request or another PACKNAM request can be issued. Refer to Permanent File Devices, section 2 for information concerning auxiliary permanent file devices.

The user cannot access permanent files residing on the normal system devices while the PACKNAM request is in effect. To access these files, he must include a PACKNAM card in either of the following formats.

```
PACKNAM
    or
PACKNAM(PX=0)
```

PERMIT CARD

The PERMIT control card allows a user to explicitly permit another user to access a private file in his permanent file catalog.

The control card format is:

PERMIT(pfn, $usernum_1$=m_1, $usernum_2$=m_2, . . . , $usernum_n$=m_n, PN=packname, R=r, NA)

pfn	Permanent file name
$usernum_i$	User number to be permitted access to pfn
m_i	Permitted mode of access. If m_i is omitted, the system assumes mode R.

If pfn is a public file, the following message is issued.

PFM ILLEGAL REQUEST, AT nnn.

PURGALL CARD

The PURGALL control card purges all permanent files in the user's catalog that satisfy the criteria specified by the parameters.

The control card format is:

PURGALL(CT=ct, AD=ad, MD=md, CD=cd, DN=dn, TY=ty, TM=tm, PN=packname, R=r, NA)

ct	File category
ad	Last access date; format of date is yymmdd
md	Last modification date; format is yymmdd
cd	Creation date; format is yymmdd
dn	Device number (0 through 77_8). The device number is assigned during system configuration time when the device is defined. It uniquely identifies a device within a family.†
ty	File type: I or INDIR — Purge all indirect access files D or DIRECT — Purge all direct access files A or ALL — Purge all files If this parameter is omitted but other parameters are specified, the system assumes ty is ALL. If no other parameters are specified and the user wishes to purge all files, he must specify TY=A.
tm	Time of day on the date specified by ad, md, or cd parameter. The time of day is expressed in the format hhmmss.

† Refer to section 2 for further information about families of permanent file devices.

packname	Name of auxiliary device on which the files to be purged reside. The PN option cannot be selected if a device number was specified.
r	Type of auxiliary device on which the files to be purged reside. The R option cannot be selected if a device number was specified

The AD, MD, and CD keywords are used to purge any files whose last access, last modification, or creation occurred before the specified date. To purge all files in his catalog, the user must enter:

PURGALL(TY=A)

CT, CN, TY, TM, and either AD, MD, or CD may be entered simultaneously.

PURGE CARD

The PURGE control card allows a user to remove a file from the permanent file device.

The control card format is:

PURGE($pfn_1, pfn_2, \ldots, pfn_n$/UN=usernum, PW=passwrd, PN=packname, R=r, NA)

pfn_i	Permanent file name

If the request is made with no parameters specified, the user's primary file is assumed.

When a PURGE command is issued for a direct access file which is not being used, the file is purged and the permanent file catalog altered accordingly. If the direct access file is in use, the catalog is altered to reflect purging of the permanent file but the actual file is not purged until the last user returns it.

To purge a file in an alternate user's catalog, the user must have write permission or the file must be semiprivate or public with write mode. If pfn_1 does not exist, the following message is issued.

pfn NOT FOUND, AT nnn.

REPLACE CARD

The REPLACE control card enables the user to place a copy of a local file in the permanent file system as an indirect access file.

The control card format is:

REPLACE($lfn_1=pfn_1, lfn_2=pfn_2, \ldots, lfn_n=pfn_n$/UN=usernum, PW=passwrd, PN=packname, R=r, NA)

lfn_i	Local file name
pfn_i	Permanent file name. If pfn_i is omitted, $lfn_i=pfn_i$.

If the request is made with no parameters specified, the user's primary file is assumed.

If pfn_i already exists, it is purged and replaced by the new file. The new file is in the same category as the file it replaced. If pfn_i does not exist, the new file is saved as a private file. Permission information and alternate user access data for the file are not lost when a file is replaced.

A user who has been granted write permission to another user's file can replace that file only if he is validated to create indirect access permanent files (refer to LIMITS control card, section 6.)

SAVE CARD

The SAVE control card allows the user to retain a copy of a local file as an indirect access file.

The control card format is:

SAVE(lfn_1=pfn_1, lfn_2=pfn_2, . . . , lfn_n=pfn_n/PW=passwrd, CT=ct, M=m, PN=packname, R=r, NA)

lfn_i	Local file name
pfn_i	Permanent file name. If pfn_i is omitted, the system assumes lfn_i=pfn_i.

If the request is made with no parameters specified, the user's primary file is assumed. If the name of the user's current primary file is specified as an lfn, the user's current subsystem is stored in the file's catalog entry.

The local files are rewound when the save operation is completed. If the optional parameters are omitted, the system assumes the following values.

Keyword	Default
PW	None
CT	PRIVATE
M	WRITE

If an auxiliary device has been previously specified by a PACKNAM card, the system saves pfn_i on the auxiliary device rather than a normal system device.

If pfn_i already exists in the user's catalog, the following message is issued.

pfn ALREADY PERMANENT, AT nnn.

LOAD/DUMP CENTRAL MEMORY UTILITY CONTROL CARDS 9

The load/dump central memory utility control cards allow the user to transfer information that resides in his job field length to a peripheral device or to transfer information from that device into central memory. The following cards are included in this category.

DMP	LOC	RBR
DMD	PBC	WBR
LBC		

The DMP and DMD control cards dump central memory in octal representation and/or display code equivalences. These cards are particularly helpful in creating dumps for debugging purposes. (Refer to Debugging Aids, section 13.) Other transfers of data from central memory use the PBC card which dumps a binary record to PUNCHB and the WBR card which writes a binary record on a specified file.

Data is loaded to central memory by the LBC, LOC, and RBR cards. The LBC control card is useful in loading binary data in an unknown format. All numeric parameters may be expressed in octal (post radix is B) or decimal (post radix is D) notation. If no radix is specified, octal is assumed. If the parameter is expected to be numeric and only B or D is specified, the value is assumed to be zero.

DMP CARD

The DMP control card requests a dump on file OUTPUT of central memory in four words per line.

The control card format is:

DMP(fwa,lwa) or

DMP(lwa) or

DMP.

fwa — First word address of memory to be dumped; fwa is relative to RA. If fwa is absent, dump mode depends on presence or absence of lwa.

lwa — Last word address plus 1 of memory to be dumped; lwa is relative to RA. If lwa alone is present, DMP assumes fwa=0. If neither fwa nor lwa is present, DMP dumps the exchange package and 40_8 locations before and after the program address register in the exchange package.

The DMP routine dumps on file OUTPUT central memory according to the DMP call parameters in four words per line. If lines are duplicated, they are suppressed with the following notation.

DUPLICATE LINES.

The DMP card must immediately follow the program to be dumped. No control card other than EXIT card may intervene.

Dumping will always stop at FL if lwa > FL. If a DMP control card image is entered under the batch subsystem from a time-sharing terminal, the dump will be formatted for 72-column output. If either fwa or lwa is nonnumeric, the request is interpreted as

DMP.

If fwa $\geq$ FL, fwa is set to FL-10_8. If both fwa and lwa > FL, fwa is set to FL-10_8 and lwa is set to FL. If fwa=lwa, the system adds 10_8 to lwa and proceeds with the operation. If fwa $\geq$ 400000_8, the first dump address is fwa-400000_8, memory from the first dump address through lwa is dumped, and the job is aborted. If fwa > lwa, the system issues the following message to the user's dayfile.

DUMP FWA.GE. LWA+1.

DMD CARD

The DMD control card requests a dump similar to that of the DMP card but additionally contains the display code equivalences to the right of the octal representations.

The control card format is:

DMD(fwa,lwa) or

DMD(lwa) or

DMD.

fwa	First word address of memory to be dumped; fwa is relative to RA. If fwa is absent, dump mode depends on presence or absence of lwa.
lwa	Last word address plus 1 of memory to be dumped; lwa is relative to RA. If lwa alone is present, DMD assumes fwa=0. If neither fwa nor lwa is present, DMD dumps the exchange package and 40_8 locations before and after the program address in the exchange package.

The user should not enter the DMD card image from a time-sharing terminal; the data transmitted for the display code equivalences may cause the terminal to be disconnected.

LBC CARD

The LBC control card is intended for loading binary data of unknown format.

The control card format is:

LBC(addr)

addr	Address relative to RA at which binary load begins; if addr is omitted, 0 (RA) is assumed.

LBC reads only one record from file INPUT. The user must make an LBC call for each record of data to be loaded. If addr is specified in the program call, binary data is loaded beginning at that address; otherwise, loading begins at the reference address (RA).

The following messages may be issued to the user's dayfile in response to an LBC card.

LBC ARGUMENT ERROR.	The load address, addr, is not numeric.
LBC FWA .GE.FL.	The load address is greater than or equal to the user's field length.
RECORD TOO LONG.	The record is too long for available memory. Available memory is filled and the excess data is skipped.

LOC CARD

The LOC control card calls the LOC program and specifies address parameters used by LOC to read octal card images from file INPUT and enter them in CM.

The control card format is:

LOC(fwa,lwa) or
LOC(lwa) or
LOC.

fwa	First word address of an area to clear (zero) before loading correction cards. If fwa is absent, LOC assumes 0.
lwa	Last word address plus 1 of the area to be cleared. If lwa is absent, LOC assumes 0.

The correction card images consist of octal address and data fields. The address field specifies the location to be corrected, and the data field contains the data to be placed in that location. Both fields may start at any column as long as the address precedes the data. The address field consists of a one- to six-digit address. If it is five characters or less, it is separated from the data field by a nonoctal character (for example, a blank). If it is six characters, no separator is required.

The data field consists of 1 to 20 octal chracters. If it is less than 20 characters, it is terminated by a nonblank, nonoctal character and is stored right-justified. If it is 20 characters, no terminator is required. Embedded blanks in the data field are ignored

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

LOC ARGUMENT ERROR.	The fwa or lwa parameter is not numeric.
LOC RANGE ERROR.	Either fwa > lwa or lwa > FL.
ADDRESS OUT OF RANGE, aaaaaa.	The address aaaaaa on a correction card is greater than or equal to the user's field length. The correction card is ignored and LOC continues.

If both addresses are specified and both are nonzero, storage is cleared from fwa to lwa and the octal line images are loaded at the specified addresses. LOC can be called to clear storage by providing an empty (zero-filled) record on file INPUT.

PBC CARD

The PBC routine writes one record from the specified area of CM to file PUNCHB.

The control card format is:

PBC(fwa,lwa) or
PBC(lwa) or
PBC.

fwa — Address relative to RA at which the binary deck begins; if this parameter is omitted, the PBC operation depends upon the presence or absence of lwa.

lwa — Last word address plus 1 of the binary deck. If lwa alone is present, PBC assumes that fwa=RA. If lwa=fwa, and a nonzero value is specified, PBC adds 10_8 to lwa. If fwa and lwa=0 or are omitted, RA contains lwa in the lower 18 bits. If the upper 12 bits of RA are 7700_8, lwa is the lower 18 bits of the location following the prefix (77) table plus the length of the prefix table.

CM is not altered by PBC.

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

PBC ARGUMENT ERROR.	Either fwa or lwa is nonnumeric.
PBC FWA .GT.LWA.	The fwa parameter is greater than lwa.
PBC RANGE ERROR.	The lwa parameter is greater than or equal to the user's field length.

RBR CARD

The RBR routine loads one binary record from a specified file.

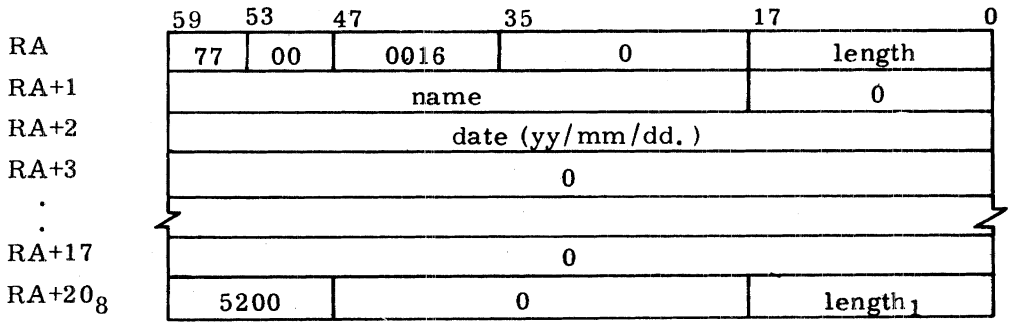
The control card format is:

RBR(n,name)

n — n is used in constructing the name of the file containing the binary record to be read. If n is less than four characters and is numeric, TAPEn is the file name. If n contains a nonnumeric character or is four or more characters long, then n itself is used as the file name. If n is absent, TAPE is the file name.

name — A 1- to 7-character name used in a record prefix.

The RBR routine loads one binary record from the specified file. If the name parameter is included, a record prefix is placed in memory starting at RA. The record itself follows. The following is the format of the record prefix.



length Record length including the prefix

length$_1$ Record length minus words RA through RA+17$_8$

If the record is too long for available memory, memory is filled, excess data is skipped, and the following message is issued to the user's dayfile.

RECORD TOO LONG.

WBR CARD

The WBR routine writes a binary record from CM to a file at its current position.

The control card format is:

WBR(n, rl)

n	n is used in constructing the name of the file on which the binary record is to be written. If n is less than four characters and is numeric, TAPEn is the file name. If n contains a nonnumeric character or is four or more characters long, then n itself is used as the file name. If n is absent, TAPE is the file name.
rl	Record length in words. If rl is 0 or absent, the length is taken from the lower 18 bits of RA.

WBR begins writing from RA.

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

WBR ARGUMENT ERROR.	The rl parameter is nonnumeric.
RECORD TOO LONG.	The rl parameter is greater than or equal to the user's field length.

TAPE MANAGEMENT 10

This section is devoted primarily to the control cards necessary to create and manage files on magnetic tape. Following the control cards are descriptions of the various types of tape formats available to the user.

The control cards described in this section are:

ASSIGN	LISTLB
BLANK	REQUEST
LABEL	VSN

The ASSIGN, LABEL, and REQUEST control cards cause files to be assigned to tape units or devices. The REQUEST card requires operator action unless the VSN is specified. In this case, if the tape has already been mounted, assignment is automatic. LABEL and ASSIGN also cause automatic assignment.

ANSI tape labels can be read using the LISTLB card and blank labeled for installation control using the BLANK card.

The control cards available to the user for assigning a file to magnetic tape are also used to create new and access existing 7- and 9-track labeled and unlabeled tapes. The following terms are used in describing these cards.

Volume	Reel of magnetic tape
Volume serial number	Number that uniquely identifies a reel of tape
Block	One physical record unit (PRU); that is, a group of contiguous characters recorded on and read from magnetic tape as a unit.
Noise	Any block less than the minimum acceptable block size is considered noise and discarded by the system.
Label	Field of characters that identifies and/or delimits a volume or file. Labels may be written in ANSI standard or nonstandard format. ANSI labels are 80-character blocks recorded at the beginning of a volume (VOL1), the beginning of a file (HDR1), the end of a file (EOF1), and the end of a volume (EOV1). Labels which do not conform to ANSI standards in format and/or content are defined as nonstandard.
Tape mark	Special configuration recorded on magnetic tape indicating the boundary between files and/or labels
Owner	Owner of a KRONOS 2.1 written tape identified in the VOL1 label by the combination of his family name and user number
KRONOS 2.1 written tape	Tape with ANSI labels written by KRONOS 2.1 and identified as such in the system code field of each HDR1, EOF1, and EOV1 label

The format and contents of ANSI labels are described in appendix G. A RESOURC control card must be included in any job that uses two or more tape units concurrently.

The following is a list of the parameters that may appear on one or more of the tape management control cards.

Keyword	Parameter	Default	Valid on	Description
LO		Installation parameter	ASSIGN REQUEST	200-bpi tape density (implies 7-track tape)
HI		Installation parameter	ASSIGN REQUEST	556-bpi tape density (implies 7-track tape)
HY		Installation parameter	ASSIGN REQUEST	800-bpi tape density (implies 7-track tape)
HD		Installation parameter	ASSIGN REQUEST	800-cpi tape density (implies 9-track tape)
PE		Installation parameter	ASSIGN REQUEST	1600-cpi tape density (implies 9-track tape)
D=	den	Installation parameter	ASSIGN BLANK LABEL LISTLB REQUEST	Specifies tape density: LO 200 bpi (7-track) HI 556 bpi (7-track) HY 800 bpi (7-track) HD 800 cpi (9-track) PE 1600 cpi (9-track) 200 200 bpi (7-track) 556 556 bpi (7-track) 800 800 bpi (7- or 9-track) 1600 1600 cpi (9-track)

Keyword	Parameter	Default	Valid On	Description
FC=	fcount	None	ASSIGN LABEL REQUEST	Frame count. Specifies the maximum size block (in frames) that may be read or written. This parameter applies only to E, B, and F data formats (refer to the F keyword).
C=	ccount	None	ASSIGN REQUEST	Character count. Specifies the maximum size block (in 6-bit characters) that may be read or written. This parameter applies only to E, B, and F data formats (refer to the F keyword).
CV= or N=	conv	Installation parameter	ASSIGN LABEL REQUEST	Specifies the conversion† mode for 9-track tapes: AS ASCII/display code conversion US Same as for AS EB EBCDIC/display code conversion
MT		Installation parameter	ASSIGN BLANK LABEL LISTLB REQUEST	Specified file resides on or is to reside on 7-track tape
NT		Installation parameter	ASSIGN BLANK LABEL LISTLB REQUEST	Specified file resides on or is to reside on 9-track tape
PO=	$p_1p_2...p_n$	A (nonsystem origin jobs) AU (system origin jobs)	ASSIGN LABEL REQUEST	A string of characters specifying processing options: A Job will be automatically aborted on an irrecoverable read or write parity error. N Job will not be aborted if an irrecoverable read or write parity error occurs; on a read operation, data will be passed to the program. R Enforce ring out. If the tape is mounted with the write ring in, job processing will be suspended until the operator remounts the tape correctly.

†Refer to appendix A for the character sets showing translation between display code and ASCII (USASI) and display code and EBCDIC.

Keyword	Parameter	Default	Valid On		Description
				W	Enforce ring in. If the tape is mounted without the write ring in, job processing will be suspended until the operator remounts the tape correctly.
				U	Inhibit unload. Do not unload at the end of usage. For system origin jobs, the inhibit unload option is selected by default; for all other origin type jobs, omission of the U option causes the tape to be unloaded at end of usage.
				F	Force unload. Unload at the end of usage. This option is useful for system origin jobs where otherwise U (inhibit unload) would be the default.
				E	Error inhibit. All hardware read/write errors are ignored and processing continues. The system does not attempt error recovery, issue error messages, nor return error status. This option is not intended for the normal user. However, it can be used to recover portions of data from a bad tape, for hardware checkout purposes, and to write on tape without skipping bad spots; in the latter case, the user is responsible for verifying that the data was written correctly.
				B	Directs the system to write system noise blocks when performing write error recovery. This option is ignored for 1600-bpi tapes. In addition, this option should not be used for tapes which are to be interchanged with other systems.

Keyword	Parameter	Default	Valid On	Description
				I Directs the system to ignore the block being read when the EOT is encountered. †
				P Directs the system to accept the block being read when the EOT is encountered. †
				S Specifies where the system is to stop on an exit condition. For unlabeled tape, it directs the system to stop at the first tape mark after the EOT is sensed. For labeled tape, it directs the system to stop at the tape mark plus EOF1 or the tape mark plus EOV1 when the EOT is encountered.
F=	format	I format for LABEL card. I format for ASSIGN card if VSN keyword is included; otherwise, default is X format. I format for REQUEST card if VSN keyword is included or VSN control card has been processed; otherwise, default is X format.	ASSIGN LABEL REQUEST	Specifies the data format I Internal X External B Blocked E Line image S SCOPE stranger tape L SCOPE long block stranger tape SI SCOPE internal F Foreign
NS=	ns	For all data formats except I, SI, and X, the default for ns is 18 frames. The NS keyword should not be specified for I, SI, and X format tapes because the definition of noise block size is implied by the format. (Refer to the K keyword.)	ASSIGN LABEL REQUEST	Noise size. Any block containing fewer than ns frames is considered noise and discarded by the system; the maximum value of ns is 31_{10} frame. If a noise size of zero is specified, the default noise size is used.

† For further information, refer to End-Of-Tape/End-Of-Reel Processing in section 10.

Keyword	Parameter	Default	Valid On	Description
LB=	ℓ	KL for LABEL card. KL for ASSIGN card if VSN keyword is included; otherwise, default is KU. KL for REQUEST card if VSN keyword is included or VSN control card has been processed; otherwise, default is KU.	ASSIGN LABEL REQUEST	Specifies whether the tape is to be treated as labeled or unlabeled. KU KRONOS unlabeled. KL KRONOS labeled. The tape is treated as having ANSI labels. If the tape is a KRONOS 2.1 tape, volume and header accessibility restrictions will be enforced. NS Nonstandard labels. The system skips over labels based on tape marks but does not process the labels. This option can also be used in processing a non KRONOS tape which although designated as unlabeled contains a tape mark prior to the beginning of data.
VSN=	vsn	Refer to appendix G.	ASSIGN BLANK LABEL REQUEST	A 1- to 6-character volume serial number that uniquely identifies a reel of tape (refer to the VSN control card).
CK		Refer to REQUEST card.	ASSIGN LABEL REQUEST	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		Refer to REQUEST card.	ASSIGN LABEL REQUEST	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.
FI= or L=	fileid	Refer to appendix G.	LABEL	A 1- to 17-character file identifier recorded in HDR1.

Keyword	Parameter	Default	Valid On	Description
FA=	fa	Unlimited access	BLANK LABEL	File accessibility. One character that indicates who has access to the labeled file (refer to appendix G, HDR1 label). A — Only the owner of the KRONOS 2.1 written tape can access the file. omitted — FA omitted indicates unlimited access. other — All future accesses to this tape must specify this character as the fa parameter. File accessibility is not checked for system origin jobs.
OFS=	fa	Unlimited access	BLANK	One character that indicates the current file accessibility of a labeled tape which is to be blank labeled. (Refer to FA description for explanation of fa characters.)
SI= or M=	setid	Refer to appendix G.	LABEL LISTLB	1- to 6-character set identifier for a multifile set (refer to appendix G, HDR1 label).
SN= or V=	secno	Refer to appendix G.	LABEL	1- to 4-digit file section number (refer to appendix G, HDR1 label).
QN= or P=	seqno	Refer to appendix G.	LABEL LISTLB	1- to 4-digit file sequence number (refer to appendix G, HDR1 label).
G=	genno	Refer to appendix G.	LABEL	1- to 4-digit generation number (refer to appendix G, HDR1 label).
E=	gvn	Refer to appendix G.	LABEL	1- to 2-digit generation version number (refer to appendix G, HDR1 label).
CR= or C=	cdate	Current date	LABEL	Creation date in the form yyddd where $1 \leq ddd \leq 366$. Creation date is meaningful only on read operations; on write operations, the current date is always used. (Refer to appendix G, HDR1 label).
RT=	rdate	Current date	LABEL	Retention date in the form yyddd (used to derive expiration date described in appendix G, HDR1 label).

Keyword	Parameter	Default	Valid On	Description
T=	retcycle	Refer to appendix G.	LABEL	1- to 3-digit retention cycle specifying the number of days from the current date that the file is to be retained (used to derive the expiration date if the RT keyword is not specified. Refer to appendix G, HDR1 label).
R		R	LABEL	Directs the system to read the existing ANSI label. The parameters on the LABEL card are compared with the values recorded on the file labels. If the comparison fails, the job is aborted.
W		R	LABEL	Directs the system to write standard ANSI labels using the parameters specified on the LABEL card or their default values. It is not necessary to specify the PO=W option to enforce ring in. If the tape is mounted without the write ring, job processing is suspended until the operator remounts the tape correctly. However, if the PO=R option is specified, the job is aborted.
VA=	va	Unlimited access	BLANK	Volume accessibility. One character that indicates restrictions on who may have access to information on the reel (refer to appendix G, VOL1 label).

omitted	VA omitted indicates unlimited access.
other	Whenever this reel is processed under KRONOS as a KRONOS 2.1 labeled tape (LB=KL), volume accessibility restrictions are imposed. Thus, the user cannot change or destroy the VOL1 label on the tape. This feature enables an installation to BLANK label new tapes and be assured that the volume serial number field of VOL1 cannot be changed by a user. If VA is nonblank, only a system origin job can change VOL1.

Keyword	Parameter	Default	Valid On	Description
OWNER=	usernum/ familyname	Refer to appendix G.	BLANK	Identifies the owner of the KRONOS 2.1 labeled tape (refer to appendix G, VOL1 label).
LSL=	lsl	The labels and data format of the specified volume require the agreement of the interchange parties.	BLANK	Label standard level (refer to appendix G, VOL1 label). 1 The labels and data format of this volume conform to the requirements of the ANSI standard. blank LSL omitted indicates that the labels and data format of the specified volume require the agreement of the interchange parties.
LO=	ltype	R	LISTLB	Specifies the type of labels to be listed (refer to appendix G). A List all required and optional ANSI labels. R List all required labels (VOL1, HDR1, EOF1, and if present, EOV1) O List all optional labels (VOL2-9, HDR2-9, EOF2-9, EOV2-9, UVL1-9, UHLa, and UTLa) V List all VOL1-9 labels. H List all HRD1-9 labels. F List all EOF1-9 labels. E List all EOV1-9 labels. U List all UVL1-9, UHLa, and UTLa labels.
L=	out	OUTPUT	LISTLB	Specifies the file on which the labels are to be listed.
U=			BLANK	Unload tape after blank labeling it; tape is always returned after blank labeling.

The system allows continuation cards for ASSIGN, BLANK, LABEL, REQUEST, and VSN cards that require more than 80 characters. If, in processing one of these cards, the system does not encounter a termination character prior to the end of the line, it assumes the next line is a continuation line. All continuation lines must contain a blank in column 1.

NOTE

The system accepts continuation lines from a time-sharing terminal only if they are contained in procedure files.

The programmer can use a literal for any parameter on a tape management control card that contains nonalphanumeric characters. Characters other than letters, numbers, and asterisks are defined as nonalphanumeric. A literal is a character string delimited by dollar signs. Blanks within literals are retained. If the literal is to contain a dollar sign, two consecutive dollar signs must be included. Thus, the literal

```
$A B$$41$
```

is interpreted as:

```
A B$41
```

If continuation cards are used, a literal cannot extend from one card to another.

Generally, if more than one parameter of a given type is specified, the last one encountered in a left-to-right scan is used. The two exceptions to this rule are in the processing option parameters. If both ring enforcement options (PO=R and PO=W) or more than one EOT option (PO=I, PO=P, PO=S) is specified, the ARGUMENT ERROR message is issued to the user's dayfile.

ASSIGN CARD

The ASSIGN control card can be used to create new and access existing 7- or 9-track unlabeled tapes.

The control card format is:

$$\text{ASSIGN(nn, lfn, D=den, } \left\{\begin{matrix}\text{FC=fcount}\\ \text{C=ccount}\end{matrix}\right\}\text{, CV=conv, } \left\{\begin{matrix}\text{MT}\\ \text{NT}\end{matrix}\right\}\text{, PO=}p_1p_2\cdots p_n\text{,}$$

$$\text{F=format, NS=ns, LB=1, VSN=vsn, } \left\{\begin{matrix}\text{CK}\\ \text{CB}\end{matrix}\right\}$$

nn — Device or device type to which the specified file is to be assigned; nn may be either the EST ordinal† of a magnetic tape unit or one of the device types MT or NT. MT is defined as a 7-track magnetic tape drive; NT is a 9-track magnetic tape drive.

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

Although the user can also include this card to assign a labeled tape to his job, he cannot use it to create or verify tape labels. It is suggested that the user include LABEL cards for all tapes whenever possible.

The job must be of system origin or the user must be validated for system origin privileges. The user must also be validated for use of magnetic tapes.†† 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.

Before performing the assignment, the system issues a RETURN on lfn.

Example:

```
ASSIGN(51, TAPE1)
```

This card assigns the file TAPE1 to the magnetic tape unit identified by EST ordinal 51.

†Contact installation personnel for a list of EST ordinals.
††Refer to LIMITS control statement, section 6.

BLANK CARD

The control card format is:

BLANK(D=den, { MT / NT }, VSN=vsn, FA=fa, OFA=ofa, VA=va, OWNER=usernum/familyname, LSL=lsl, U)

With the BLANK control card, an installation can establish control over the use of labeled tapes. The values supplied on the card are used to blank label a tape with standard ANSI volume header (VOL1), first file header (HDR1), and first end-of-file (EOF1) labels. The labels are written as follows:

	VOL1	HDR1	*	*	EOF1	*	*		

In writing these labels, the system uses default values for all fields except those fields for which there are corresponding parameters on the BLANK card. The VA and FA keywords can be used to restrict access to information on the reel and the specified files, respectively. If the tape to be blank labeled is a labeled tape which has a file accessibility other than A, this old file accessibility must be specified by the OFA parameter. When the tape is blank labeled, the file accessibility is that specified by the FA parameter. The default track type may be set by the installation to either MT or NT.† If a track type other than the default is desired, it must be specified.

Once a tape has been blank labeled, the user can modify the labels as follows:

1. If the volume accessibility field of VOL1 indicates unlimited access (that is, VA is blank), the user can:
   - Include another BLANK card to change VOL1, HDR1, or EOF1 values.
   - Request the tape as unlabeled (that is, LB=KU) and write it in whatever format he specifies.
   - Include a LABEL card to change HDR1 by specifying one or more of the parameters associated with that label and selecting the write label (W) option.
2. If the volume accessibility field is nonblank, the user can:
   - Include a LABEL card to change HDR1. However, in requesting a tape in which VA is nonblank, the user must specify an NOS labeled tape (that is, LB=KL), and therefore, cannot change or destroy the VOL1 label.
   - Submit a system origin job to change VOL1.

†Contact installation personnel for the default track type.

LABEL CARD

LABEL(lfn, D=den, FC=fcount, CV=conv, {MT / NT}, PO=$p_1p_2 \ldots p_n$, F=format, NS=ns,
LB=ℓ, VSN=vsn, {CK / CB}, {FI=fileid / L=fileid}, FA=fa, {SI=setid / M=setid}, {SN=secno / V=secno},
{QN=seqno / P=seqno}, G=genno, E=gvn, {CR=cdate / C=cdate}, {RT=rdate / T=retcycle}, {W / R})

lfn — Name of the file that resides on or is to reside on magnetic tape

The LABEL control card directs the system to assign file lfn to a tape unit. This assignment occurs using VSN only; the file identifier is not considered in assigning. If a file by the name lfn already exists, the following action is taken.

1. If lfn is assigned to a device other than a tape unit, job processing continues with the next control card.
2. If lfn is an existing tape file and the read label (R) parameter is specified, the system compares the parameters on the LABEL card with the values recorded on the file labels. If the comparison fails, the job is aborted.
3. If lfn is an existing tape file and the write label (W) parameter is specified, the system compares the vsn specified on the LABEL or VSN card with the vsn in the VOL1 label. If the vsns match, the system writes file labels (refer to appendix G) using the parameters specified on the LABEL card. If the vsns do not match, the job is aborted.

To assign to tape an lfn that was previously assigned in the same job to another device, the user must make sure that lfn is RETURNed before the LABEL card is processed. Note that the default track type may be set by the installation to either MT or NT.† If a track type other than the default is desired, it must be specified.

If lfn is to be used for checkpoint dumps and the dumps are to be written on labeled tape, the CK or CB parameter must be included on the LABEL card. For further information about checkpoint dumps, refer to the REQUEST control card.

Example 1:

```
LABEL(NEWFILE,VSN=TP01,FI=FILEA,W)
```

This card creates an ANSI-labeled tape which the job can access by the filename NEWFILE. Default values are used for all fields of HDR1 except the file identification, FILEA. Any data written is recorded in 512 CM word blocks.

Example 2:

```
LABEL(OLDFILE,VSN=TP01,FI=FILEA)
```

This card assigns the tape file created in a previous job (refer to example 1) to the file OLDFILE. The system compares the vsn in VOL1 and the file identification in HDR1 with the values on the card.

Example 3:

```
LABEL(CFILE,VSN=TP02,FI=THIRDFILE,SI=ABCD,QN=0003)
```

This card assigns the tape with vsn TP02 to the file CFILE. The system positions the tape to the beginning of data of THIRDFILE, the third file in the multifile set ABCD.

†Contact installation personnel for the default track type.

LISTLB CARD

The control card format is:

$$\text{LISTLB (lfn, } \left\{\begin{matrix}\text{SI=setid}\\ \text{M=setid}\end{matrix}\right\}, \left\{\begin{matrix}\text{QN=seqno}\\ \text{P=seqno}\end{matrix}\right\}, \text{ LO=ltype, L=out)}$$

The LISTLB control card directs the system to read the ANSI labels on the tape file specified by lfn and write them on the user specified file out. The setid and seqno parameters enable the user to list only those HDR1-9 and EOF1-9 labels which have the specified multifile set identifier and/or file sequence number. The ltype parameter allows the user to specify the type of labels to be listed. Refer to **appendix G** for a description of each type of label.

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:

$$\text{REQUEST(lfn, D=den, } \left\{\begin{matrix}\text{FC=fcount}\\ \text{C=ccount}\end{matrix}\right\}, \text{CV=conv, } \left\{\begin{matrix}\text{MT}\\ \text{NT}\end{matrix}\right\}, \text{PO=}p_1, p_2 \ldots p_n,$$
$$\text{F=format, NS=ns, LB=}\ell\text{ , VSN=vsn, } \left\{\begin{matrix}\text{CK}\\ \text{CB}\end{matrix}\right\})$$

This comment is displayed at the system console, directing the operator to make the requested assignment. If the user has previously specified a vsn via a VSN control card or if he has included the VSN keyword on the REQUEST card, the system initiates automatic tape file 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.

The REQUEST card can be used to create new and access existing 7- or 9-track unlabeled tapes. Although the user can also include this card to assign a labeled tape to his job, he cannot use it to create or verify tape labels. It is suggested that LABEL cards be used for all tapes whenever possible. The default track type may be set by installation of either MT or NT.† If a track type other than the default is desired, it must be specified.

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)

†Contact installation personnel for the default track type.

- 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)

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 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 CCCCCCC, and selects the CK option. For a subsequent restart job, however, the system assumes the user has made the checkpoint file available.

VSN CARD

The control card format is:

VSN(lfn_1=vsn_1,lfn_2=vsn_2,...,lfn_n=vsn_n)

lfn_i	Name of the file with which the specified vsn is to be associated
vsn_i	1- to 6-character volume serial number to be associated with lfn_i. If the vsn_i is zero, absent, or SCRATCH, any available scratch tape will be automatically assigned to lfn_i. If characters other than letters and numbers are used, vsn_i must be specified as a literal.

The system allows tape assignment to be performed either by the system or by the operator. By supplying a vsn uniquely identifying every tape (labeled, unlabeled, and nonstandard labeled), the user enables the system to assign tapes without operator intervention.

A vsn is provided via the VSN keyword on a LABEL or REQUEST card or via a VSN card. With a VSN card the user can:

- Omit the VSN keyword from his LABEL or REQUEST cards and specify lfn/vsn associations on the VSN card instead. This allows the user to specify new vsns without changing LABEL or REQUEST cards.

- Override the vsn specified on subsequent ASSIGN, LABEL, REQUEST, or VSN cards. For example, the sequence:

  ```
  VSN(FILEA=123)
  VSN(FILEA=124)
  LABEL(FILEA)
  ```

  directs the system to assign FILEA to the tape with vsn 123. However, the user can redeclare an lfn/vsn association by returning the file. Thus, the following sequence

  ```
  VSN(FILEA=123)
  RETURN(FILEA)
  VSN(FILEA=124)
  LABEL(FILEA)
  ```

  directs the system to assign to FILEA the tape with vsn 124.

- Associate the vsns of two or more duplicate reels with one file.† If any of several duplicate reels can be used (that is, they differ only in vsns), the vsns should be separated by equal signs. Thus, the card

  ```
  VSN(FILE1=VOL100=VOL101)
  ```

  indicates that either the tape with the vsn of VOL100 or the tape with the vsn of VOL101 can be assigned to FILE1.

- Specify the vsns of a multireel file.† If the file extends to more than one reel, the vsns for all reels required must be separated by slashes. The system assigns the reels in the order indicated on the card. For example, the card

  ```
  VSN(FILE2=VSN23/VSN24/VSN25)
  ```

  indicates that FILE2 may extend to the three reels identified by the vsns of VSN23, VSN24, and VSN25.

The system processes tape requests as follows:

1. Whenever a tape is mounted, the system checks for labels. If the tape was labeled, the system keeps a record of the vsn read from VOL1 and the equipment on which the tape is mounted.

† Up to 55 vsns can be specified for a single file in any combination of duplicate reel and/or multireel configurations.

2. If, when a request is made for tape assignment, an lfn/vsn association is encountered, the system compares the vsn associated with the file (or one of its equivalences) with the vsns read from mounted tapes. If a match is found, the system automatically assigns the tape to the requesting job, provided a deadlock would not occur. If the tape is not mounted, the system rolls out the job until a tape with the required vsn is mounted. For a mounted, unlabeled tape, the operator enters a VSN command specifying the required vsn. The system is then able to automatically assign the tape.
3. If no lfn/vsn association is encountered when the request is made, the system directs the operator to assign an available unit.
4. For an ASSIGN card, the method of assignment depends on the nn parameter. If nn is a device type (MT or NT), the operator must assign an available unit. If nn is the EST ordinal of a tape unit, the system automatically assigns the specified unit.

The following is a summary of the system and/or operator action taken in response to an ASSIGN, LABEL, or REQUEST card. The VSN column indicates whether or not the user has specified an lfn/vsn association via the VSN keyword or a VSN card. The mode column shows the mode as determined by the system in checking for labels.

Card	VSN	Mode	Action
ASSIGN	Yes	Labeled	If the nn parameter is MT or NT, the operator must assign an available unit. If the nn parameter is the EST ordinal of a tape unit, assignment is automatic.
	Yes	Unlabeled	Same as when the vsn is specified and the tape is labeled.
	No	Labeled	Same as when the vsn is specified and the tape is labeled.
	No	Unlabeled	Same as when the vsn is specified and the tape is labeled.
REQUEST	Yes	Labeled	The system matches the vsn read from VOL1 with the vsn on the REQUEST or VSN card. Assignment is automatic.
	Yes	Unlabeled	The operator enters a VSN command specifying the vsn included on the REQUEST or VSN card.† Assignment is automatic.
	No	Labeled	The operator assigns an available unit to lfn.
	No	Unlabeled	The operator assigns an available unit to lfn.
LABEL	Yes	Labeled	The system matches the vsn read from VOL1 with the vsn on the LABEL or VSN card. Assignment is automatic.
	Yes	Unlabeled	The operator enters a VSN command specifying the vsn included on the LABEL or VSN card.† Assignment is automatic.
	No	Labeled	The operator assigns an available unit to lfn.
	No	Unlabeled	The operator assigns an available unit to lfn.

The LB keyword is not used in assigning a tape. Rather, it is used in processing the data on the tape once the assignment has been made.

†A VSN which contains special characters should not be specified in a request for an unlabeled tape. It is not possible to enter special characters via the VSN,xx,aaaaaa. operator command.

MAGNETIC TAPE FORMATS

The standard magnetic tapes used are 7-track, 1/2-inch tape and 9-track, 1/2-inch tape. Each type of tape can be written in binary or coded mode. Unless specified otherwise, tapes are assumed to be in binary mode. The user can select 200, 556, or 800 bits per inch (bpi) density for 7-track tapes or 800 or 1600 characters per inch (cpi) density for 9-track tapes, provided these densities are available with the hardware. Tape density can be specified by a LABEL, ASSIGN, or REQUEST control card, the LABEL macro (refer to section 4, volume 2), or an IPRDECK installation option (refer to the Installation Handbook). The system normally performs automatic processing of tape parity errors and end-of-tape conditions. However, the user can control the processing of these functions via the PO keyword on LABEL, ASSIGN, and REQUEST control cards or the po field of the FET (FET+8, bits 36 through 47).

DATA FORMATS

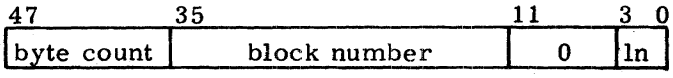
Data can be recorded on magnetic tape in any of eight formats.

Format	Description
I	Internal
SI	SCOPE internal
X	External
S	SCOPE stranger
L	SCOPE long block stranger
E	Line image
B	Blocked
F	Foreign

The control card user specifies the data format via the F keyword of a LABEL, ASSIGN, or REQUEST control card. The LABEL macro user specifies the data format via FET+10, bits 30 through 35. The following is a description of the physical and logical characteristics of each format. Note that the user can define maximum block size, end-of-reel conditions, and noise for any format via control card or FET parameters; the following description of these characteristics defines the suggested (and default) values.

I (INTERNAL) FORMAT

Characteristics	Description
Mode	Binary
Block size (PRU size)	Actual data block size can range from 0 to 512 (1000_8) CM words in exact multiples of CM words. All blocks except those containing labels include a 48-bit block terminator formatted as follows:



byte count	Total number of bytes in the block including the block terminator
block number	Number of blocks since the last HDR1 label†
ln	Level number
	0 End-of-record
	17_8 End-of-file

User-specified frame or character counts have no meaning.

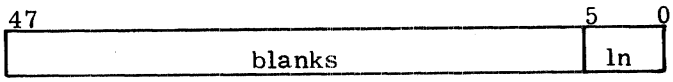
Logical end-of-record	Any block with fewer than 512 (1000_8) CM words of data is considered a logical end-of-record. During a write operation, the level number field of the block terminator contains the level number obtained from FET+0, bits 14 through 17, or the WRITECW macro control word. During read operations, the system will return end-of-record status and the contents of the block terminator level number field. If the level number is 17_8, the system will also return end-of-file status. Some blocks may consist only of a block terminator.
Logical end-of-file	Any block consisting of only a block terminator with a level number of 17_8 is considered a logical end-of-file. The system ensures that an end-of-record will always precede an end-of-file by writing, if necessary, a block terminator with a level number of zero prior to the end of file.
Logical end-of-information	If the tape is unlabeled, there is no logical end-of-information. For labeled tapes, a tape mark followed by an EOF1 label is considered the end-of-information. The system issues a label content error if it encounters a tape mark without a valid label following it.

† Refer to appendix G.

Characteristics	Description
End-of-reel	Refer to option 3 under End-Of-Tape/End-Of-Reel Conditions.
Noise	Any block containing fewer than eight frames for 7-track tapes or six frames for 9-track tapes is considered noise, and therefore, ignored.
Special considerations	• All 9-track tapes are written in an even multiple of bytes. • On all read operations, the system checks for fill status and compares the number of bytes read and the block number expected with the byte count and block number values in the block terminator. If the specified condition does not occur, the system handles it as if it were a parity error. This method is designed to prevent dropped or fragmented blocks; in general, it provides a much higher degree of reliability than any other format.

S (SCOPE INTERNAL) FORMAT

Characteristics	Description
Mode	Binary or coded as indicated by FET+0, bit 1.
Block size (PRU size)	For binary mode, the block size can range from 0 to 512 (1000_8) CM words in exact multiples of CM words. Any block smaller than the maximum size except those containing labels will contain a 48-bit block terminator. This terminator has the same format as that for I format. For coded mode, the block size can range from 0 to 128 (200_8) CM words in exact multiples of CM words. Any block smaller than the maximum size except those containing labels will contain a 48-bit block terminator formatted as follows:



ln Blank if level is 0

1 through 17_8 for all other levels

User-specified frame or character counts have no meaning.

Characteristics	Description
Logical end-of-record	For binary mode, any block containing fewer than 512 (1000_8) CM words represents a logical end-of-record. For coded mode, any block containing fewer than 128 CM words represents a logical end-of-record. If a logical record consists of an exact multiple of 512 (binary) or 128 (coded) CM words, the block that denotes the logical end-of-record consists solely of a block terminator. During write operations, the level number field of the block terminator contains the level number from FET+0, bits 14 through 17, or the WRITECW macro control word. During read operations, the system will return end-of-record status and the contents of the block terminator level number field. If the level number is 17_8, the system will return end-of-file status.
Logical end-of-file	Same as for I format.
Logical end-of-information	Same as for I format.
End-of-reel	Refer to option 3 under End-Of-Tape/End-Of-Reel Conditions.
Noise	Same as for I format.
Special considerations	• The system writes all 9-track tapes with 3n+2 mode. • The system does not perform block checking via block terminators as is done for I format. • For read and write operations on a coded 7-track tape, KRONOS is incompatible with SCOPE. The system converts data from display code to external BCD on write operations and from external BCD to display code on read operations. SCOPE converts data from external BCD to internal BCD on both read and write operations. • For 7-track tapes, standard code conversion is performed. For 9-track tapes, no code conversion will be performed. • For read operations, if a coded 7-track tape contains external BCD 1632 in byte 4 of a CM word, the system converts it to an end-of-line (0000 in display code). The converse is true for write operations. • The FET device type is returned in SCOPE format (refer to the description of the CIO OPEN macro in section 3, volume 2).

X (EXTERNAL) FORMAT

Characteristics	Description
Mode	Binary
Block size (PRU size)	Actual data block size can range from 0 to 512 (1000_8) CM words in exact multiples of CM words.
Logical end-of-record	Any block containing fewer than 512 CM words represents a logical end-of-record. If a logical record consists of an exact multiple of 512 words, the block that denotes the logical end-of-record consists solely of a block terminator in the same format as that for I format.
Logical end-of-file	Tape mark

Characteristics	Description
Logical end-of-information	None
End-of-reel	Refer to option 1 under End-Of-Tape/End-Of-Reel Conditions.
Noise	Same as for I format.
Special considerations	• X-formatted tapes cannot be labeled. • All 9-track tapes are written in an even multiple of bytes.

S (SCOPE STRANGER) FORMAT

Characteristics	Description
Mode	Binary or coded as indicated by FET+0, bit 1.
Block size (PRU size)	No explicit multiple of frames is required. The maximum block size may be specified in the mlrs field of the FET (FET+7, bits 0 through 17). If no block size is specified in the mlrs field, it is assumed to be 1000_8.
Logical end-of-record	On a CIO READ(010) or READSKP(020) request, each PRU is considered an end of record.
Logical end-of-file	None
Logical end-of-information	If the tape is unlabeled, there is no logical end-of-information. If the tape is labeled, the logical end-of-information is a tape mark followed by an EOF1 label.
End-of-reel	Refer to option 2 under End-Of-Tape/End-Of-Reel Conditions.
Noise	Any block containing fewer than 18 frames is considered noise, and therefore, ignored.
Special considerations	• Level numbers 1 through 16_8 are interpreted as level number 0. • Standard code conversion is performed for 7- or 9-track tapes in coded mode. • For CIO READ (010), WRITE (014), WRITER (024), and WRITEF (034) functions, a one-block (PRU) operation is performed with the unused bit count (FET+7, bits 24 through 29) taken from and returned to the FET. • The FET device type is returned in SCOPE format (refer to the description of the CIO OPEN macro in section 3, volume 2).

L (SCOPE LONG BLOCK STRANGER) FORMAT

The characteristics and descriptions are the same as for S format tapes except that if no block size is specified in the mlrs field (FET+7, bits 0 through 17), it is assumed to be LIMIT-FIRST-1.

E (LINE IMAGE) FORMAT

Characteristics	Description
Mode	Coded
Block size (PRU size)	The block size cannot exceed 5120 frames. If the tape unit will not allow an odd number of frames to be written, the system will append a space. Unless the user specifies otherwise when he requests the tape, the system assumes the maximum block size is 136 frames.
Logical end-of-record	For a write operation, there is no logical end-of-record. For a read operation, end-of-record status is returned when a tape mark is encountered. An additional read operation returns end-of-file status.
Logical end-of-file	Tape mark
Logical end-of-information	If the tape is unlabeled, there is no logical end-of-information. If the tape is labeled, the end-of-information is a tape mark followed by an EOF1 label.
End-of-reel	Refer to option 2 under End-Of-Tape/End-Of-Reel Conditions.
Noise	Same as for S-formatted tapes.
Special considerations	• E-formatted tapes cannot be labeled. • For a write operation, a block of data will stop either at a zero byte (end-of-line) in byte 4 of a CM word or at the multiple of CM words (rounded up) based on the frame or character count. The system will then space-fill the buffer to the number of frames specified. Thus, the amount of data written will exactly equal the amount specified. • For a read operation, if there is an odd number of characters, the system will space-fill the last six bits of the last byte and delete all trailing spaces. This ensures that data will be stored in the buffer with at least one 12-bit byte of zeros. • For a control word write operation, no end-of-line processing is done. Data is blocked on tape using the specified frame count. Likewise for a control word read operation, no end-of-line processing is done; data is transferred to the user as it is read.

B (BLOCKED) FORMAT

Characteristics	Description
Mode	Coded
Block size (PRU size)	The block size cannot exceed 5120 frames. If the tape unit will not allow an odd number of frames to be written, the system will append a space. Unless the user specifies otherwise when he requests a tape, the system will assume the maximum block size is 150 frames.

Characteristics	Description
Logical end-of-record	For a write operation, there is no logical end-of-record. For a read operation, end-of-record status is returned when a tape mark is encountered. An additional read operation returns end-of-file status.
Logical end-of-file	Tape mark
Logical end-of-information	If the tape is unlabeled, there is no logical end-of-information. If the tape is labeled, the end-of-information is a tape mark followed by an EOF1 label.
End-of-reel	Refer to option 2 under End-Of-Tape/End-Of-Reel Conditions.
Noise	Same as for S-formatted tapes.
Special considerations	• B-formatted tapes cannot be labeled. • A write operation will stop either at the first zero byte encountered in any byte position of a CM word or at a multiple of CM words (rounded up) based on the frame or character count. • For a read operation, the system ensures that data will be stored in the buffer with at least one 12-bit byte of zeros. • For a control word write operation, no end-of-line processing is done. Data is blocked on tape using the specified frame count. Likewise for a control word read operation, no end-of-line processing is done; data is transferred to the user as it is read.

F (FOREIGN) FORMAT

Characteristics	Description
Mode	Binary or coded, as needed, for 7-track tapes and binary for 9-track tapes
Block size (PRU size)	The block size cannot exceed the CM buffer size. No explicit multiple of frames is required. The maximum block size must be specified at tape request time. The block size is used to determine whether to continue read or write operations based on the amount of data versus the space in the buffer. For example, if the maximum block size is 1000_8 CM words, the read operation will stop any time less than 1001_8 words remain. It is recommended that the user specify a buffer size equal to the largest block.
Logical end-of-record	None
Logical end-of-file	Tape mark
Logical end-of-information	None
End of reel	Refer to option 1 under End-Of-Tape/End-Of-Reel Conditions.
Noise	Any block containing fewer than 18 frames is considered noise, and therefore, ignored.

Characteristics	Description
Special considerations	• F-formatted tapes cannot be labeled. • For 7-track tapes, if a parity error is detected because the tape is being read in the opposite mode, the mode will be switched. • F-format operations are only done using control word reads and writes. On read operations, the control words are transferred to the user regardless of the operation being used.

END-OF-TAPE/END-OF-REEL CONDITIONS

The following is a description of the processing options for end-of-tape conditions. The user can select one of these options by default by specifying the data format or he can specify an option via the PO keyword on a LABEL, ASSIGN, or REQUEST control card or the processing option field of the FET (FET+8, bits 36 through 47). In addition, the user processing option (FET+1, bit 45) gives the macro user control over end-of-reel conditions. For further information, refer to the CLOSER, REWIND, and UNLOAD macros described in section 3, volume 2.

Option	PO= Option	Description
1	I	If, during a write operation, the system senses the end-of-tape, it rewrites the block on which the EOT occurred as the first block on the following reel. No trailer information is written on the current reel. During a read operation, the block on which the EOT occurred is ignored and reading continues on the next reel. If a tape mark and the EOT are sensed at the same time, the EOT is ignored.
2	P	If, during a write operation, the system senses the end-of-tape, the system writes a trailer sequence, consisting of a tape mark, following the block on which the EOT was sensed. Any data that occurs following the block on which EOT was sensed, yet before the tape mark, is ignored. During a read operation, the system transfers to the user the block on which the EOT was sensed. The read operation resumes on the next reel. If a tape mark and the EOT are sensed at the same time, the EOT is ignored.
3	S	If, during a write operation, the system senses the end-of-tape, the system writes a trailer sequence following the block on which the EOT was sensed. This trailer sequence consists of a tape mark followed by an EOV1 label for labeled tapes, and tape marks for unlabeled tapes. The next block is written on the next reel. During a read operation, the EOT is noted and the system transfers to the user the block on which the EOT was sensed plus all following blocks until a trailer sequence (as described above) is recognized. Reading resumes on the next reel.

For options 1 and 2, the system is concerned only with the block on which the EOT is sensed. If tapes written using these options are transferred to another system, any data that occurs on the reel after this block should be ignored.

PRODUCT SET CONTROL CARDS 11

KRONOS supports the following products. †

FORTRAN Extended 4	Sort/Merge 4
COBOL 4	PERT/TIME 1
ALGOL 3	SIMULA 1
ALGOL 4	SIMSCRIPT 3

Although they are generally used in time-sharing jobs, the KRONOS interactive compiler (BASIC) and the KRONOS interactive interpreter (APL) can also be used in batch jobs. The BASIC control card call is described in this section. The APL control card is described in the APL Reference Manual.

FORTRAN Extended, COBOL 4, and Sort/Merge 4 use the Record Manager for accessing files. In addition, KRONOS supports the indexed sequential (IS), direct access (DA), and actual key (AK) file management capabilities of the Record Manager. The user interfaces with the Record Manager via the FILE control card. This card is used to update the file information table (FIT) which is required for files which the Record Manager accesses. For further information, refer to the Record Manager Reference Manual.

This section describes the user library feature available to the KRONOS user and the control card calls for the product supported by KRONOS.

USER LIBRARIES

KRONOS offers the user the option of specifying a library other than the product set default library. The user can then write library routines to perform special functions to meet his own requirements. † † Routines can also be given names identical to routines from another library without causing a system conflict. This enables a user to compare the performance of library routines without modifying his software.

The libraries from which externals are to be satisfied can be specified as parameters on the LDSET card as follows:

LDSET(LIB=lib_1/lib_2/.../lib_n)

LOAD(lfn)

lib_i	Library from which externals are to be satisfied. The system checks through the specified libraries sequentially.
lfn	Name of the file to be loaded

Libraries can also be specified by using the LIBRARY card to define the global library set. (Refer to Loader Reference Manual.) The default system library, SYSLIB, is used to satisfy the externals if no library is specified or if unsatisfied externals exist after using the libraries specified or defaulted.

† Refer to the preface for a list of product set reference manuals.
† † Refer to the Loader Reference Manual for information about the generation of a user library.

CONTROL CARD FORMATS

The following is a description of the program call cards for the product set members supported by KRONOS. The reference manuals listed in the preface provide further information concerning these products.

NOTE

The following product set control cards are processed in product set format.

FTN	SORTMRG	FILE
COBOL	SIMULA	
ALGOL	SIMI5	

Product set format does not allow file names beginning with a numeric character (refer to Control Card Format, section 5).

FTN CARD

The FORTRAN Extended control card calls the compiler to a control point. The minimum memory requirement for FORTRAN Extended is 46,000 octal locations. FTN externals are satisfied from the user library FORTRAN.

The control card format is:

FTN, $p_1, p_2, \ldots, p_n$.

p_i	Description
A	Branch to EXIT control card if fatal compilation error occurs. If there is no EXIT card, the job terminates.
A=0	Control transfers to the next control card, regardless of the installation default, if fatal compilation errors occur.
B	Object code is written on file LGO. If the B parameter is omitted, this option is assumed.
B=lfn_1	Object code is written on file lfn_1.
B=0	Suppresses object code output.
BL	Generates output listing that is easily separable into components by issuing page ejects between source code, error summary (if present), cross reference map, and object code (if requested); and ensures that each program unit listing contains an even number of pages (page parity) by issuing a blank page at the end if necessary. If omitted, BL=0 is assumed.
BL=0	Listings are produced in compact format.
C	Use COMPASS assembler for compiler-generated code.
C=0	Selects the FTN assembler regardless of the installation default.
D	Debug mode of compilation; a minimum of $61,000_8$ locations is required if this option is selected. Debug input is obtained from INPUT source.
D=lfn_2	Debug input is to be obtained from lfn_2.
D=0	Debug statements are ignored.
E	Compiler-generated object code on file COMPS is output as COMPASS line images for input to Update.
E=lfn_3	Compiler-generated object code on lfn_3 is output as COMPASS card images for input to Update.
E=0	Normal binary object file is generated. If the E parameter is omitted, this option is assumed.
EL=ℓ	Lists diagnostics according to list specification ℓ :

p_i		Description
	ℓ	Description
	A	List diagnostics indicating all non-ANSI usages, as well as fatal diagnostics. Also, list informative diagnostics if compiling under OPT=0, 1, or 2; list note and warning diagnostics if compiling in TS mode.
	I	List informative and fatal diagnostics if compiling under OPT= 0, 1, or 2; list note, warning, and fatal diagnostics if compiling in TS mode.
	N	List note, warning, and fatal diagnostics if compiling in TS mode; list fatal diagnostics if compiling under OPT = 0, 1, or 2.
	W	List warning and fatal diagnostics if compiling in TS mode; list fatal diagnostics if compiling under OPT = 0, 1, or 2.
	F	List fatal diagnostics.
	If this parameter is omitted, EL=I is assumed.	
G	Loads the first system text overlay from the sequential binary file SYSTEXT. A maximum of seven system texts can be specified by any combination of the G, S, and C parameters.	
G=lfn_4	Loads the first system text overlay from the sequential binary file lfn_4.	
G=lfn_4/ovl	Searches the sequential binary file, lfn_4, for a system text overlay with the name ovl and loads the first such overlay encountered.	
G=0	No system text is loaded. If the G parameter is omitted, G=0 is assumed.	
GO	Binary object file is loaded and executed at end of compilation.	
GO=0	Binary object file is not loaded and executed. If the GO parameter is omitted, GO=0 is assumed.	
I	Source input is on file COMPILE. If I is omitted, input is on file INPUT.	
I=lfn_5	Source input is on file lfn_5.	
L	Listable output (specified by list control options BL, EL, OL, R, and SL) is to be written onto file OUTPUT. If list control options are not specified, the listing consists of the source program, informative and fatal diagnostics, and a short reference map. If the L parameter is not specified, this option is assumed.	
L=lfn_6	Listable output is to be written onto file lfn_6.	
L=0	Fatal diagnostics and the statements that caused them are listed on the file OUTPUT. All other compile-time output, including intermixed COMPASS, is suppressed. List control options are ignored.	

p_i	Description
LCM=D	Selects 17-bit address mode for level 3 data. This method produces more efficient code for accessing data assigned to level 3. User ECS field length must not exceed 131,071 words. If the LCM parameter is omitted, this option is assumed.
LCM=I	Selects 21-bit address mode for level 3 data. This mode depends heavily upon indirect addressing. LCM=I must be specified if the execution ECS field length exceeds 131,071 words. In TS mode, all LCM addressing is done in 21-bit mode, regardless of the LCM parameter.
ML	Current date in the form yyddd is used for the MODLEVEL micro. If the ML parameter is omitted, this option is assumed.
ML=nnn	Specifies nnn as the value of the MODLEVEL micro used by COMPASS. nnn consists of 1 to 7 alphanumeric characters.
OL	Generated object code is listed on the file specified by the L parameter.
OL=0	Object code is not listed. If the OL parameter is omitted, OL=0 is assumed.
OPT=n	Level of optimization: n=0 Fast compilation (automatically activates T option) n=1 Standard compilation and execution. (Default value). n=2 Fast execution. OPT=2 is equivalent to OPT.
P	Page numbering is continuous from subprogram to subprogram, including intermixed COMPASS. If P is omitted, page numbers begin at 1 for each subprogram.
P=0	Page numbers begin at 1 for each subprogram.
PL=n	n is the maximum number of records produced by the user program at execution time which can be written on the file OUTPUT. n does not include the number of records in the source program listing, and compilation and execution time listings; $n \leq 999\ 999\ 999$.
PL=nB	An octal number must be suffixed with a B; $n \leq 77\ 777\ 777$.
Q	Compiler performs full syntactic scan of the program, but no object code is produced. No code addresses are provided if a reference map is requested. This mode is substantially faster than a normal compilation; but it should not be selected if the program is to be executed.
Q=0	Normal compilation. If the Q parameter is omitted, this option is assumed.
R=n	Selects the kind of reference map required:

_Pi_		Description
	n=0	No map
	n=1	Short map (symbols, addresses, properties, and a DO-loop map)
	n=2	Long map (short map plus references by line number)
	n=3	Long map with printout of common block members and equivalence groups
	In TS mode, R=3 is identical to R=2; common and equivalence groups are not listed.	
ROUND=s	Directs the compiler to produce code that rounds arithmetic operations involving the following operators: (s=*/+ or -).	
ROUND=0	Computation for the indicated operators is not rounded. If the ROUND parameter is omitted, this option is assumed.	
ROUND	Implies ROUND=+-*/	
S=ovl	The system text overlay, ovl, is loaded from the job's current library set.	
S=lib/ovl	The system text overlay, ovl, is loaded from lib. lib can be a user library file or a system library.	
S=0	When COMPASS is called to assemble any intermixed COMPASS programs, it will not read in a system text file. If the S parameter is omitted, the default is S=SYSTEXT.	
SEQ	Source input file is in sequenced line format (as used in TSRUN FORTRAN). Specifying this option automatically activates the TS option.	
SEQ=0	Source input file is in standard FORTRAN format. If the SEQ parameter is omitted, this option is assumed.	
SL	Source program is listed on the file specified by the L parameter. If the SL parameter is omitted, this option is assumed.	
SL=0	Source program is not listed.	
SYSEDIT	All input/output references are accomplished indirectly through a table search at object time. File names are not entry points in main program, and subprograms do not produce external references to the same file.	
SYSEDIT=0	Input/output references accomplished directly; file names are used as entry points in the main program, and subprograms produce external references to the file name. If the SYSEDIT parameter is omitted, this option is assumed.	
T	If this parameter is specified, full error traceback occurs. This is primarily used for programs in debug stages. Selecting the D parameter or OPT=0 automatically activates the T option.	
T=0	No traceback occurs when an error is detected.	

p_i	Description
	If the T parameter is omitted, this option is assumed.
TS	Timesharing mode. Compilation speed and field length are optimized at the expense of execution speed and field length. TS mode is preferable to the optimizing compilation modes (OPT=1, 2, or 3) for the debugging stages of a program. Specifying the TS option together with the C, D, E, or Q option constitutes a fatal control card error.
X	File OPL is the source of external text (XTEXT) when location field of XTEXT pseudo instruction is blank. Only one X parameter may be specified. If omitted, external text is on OLDPL file.
$X=lfn_7$	External text on file lfn_7.
Z	When Z is specified, all subroutine calls having no parameters are forced to pass a parameter list consisting of a zero word. This feature is useful to COMPASS-coded subroutines expecting a variable number of parameters. Z should not be specified unless necessary, since programs require less memory if Z is omitted.
Z=0	A zero word parameter list is not passed. If the Z parameter is omitted, this option is assumed.

COBOL CARD

The COBOL control card calls the COBOL compiler to the control point. The minimum memory requirement for COBOL is 52,000 octal locations. External references are satisfied from COBOL.

The control card format is:

COBOL, $p_1, p_2, \ldots, p_n$. comments

If the control statement does not fit on one card, it can be continued on the next card. However, the last character on the first card must be separator, such as (, +- or /.

The following parameters can be supplied.

p_i	Description
A	Leading blanks are treated as zeros in arithmetic statements and comparisons.
B	Relocatable binary object code is written to file LGO. If the B parameter is omitted, this option is assumed.
$B=lfn_1$	Binary object code is written to file lfn_1. The file name should specify a mass storage file.
B=0	Suppresses binary output of object code.
BUF	In COBOL 4 buffer sizes are based on the record description; a minimun size of 514 words has been established. The BUF parameter selects the Version 3 method which does not use record description or ALTERNATE AREAS to determine the minimum block size.

p_i	Description
C	Specifies that a copy is to be made from source rather than from the library, which is the default condition.
D	Inhibits COBOL program execution when an E diagnostic is encountered.
DB	Checks for subscript range errors. If an error is encountered, the run is terminated. If DB is not specified, no check is made for subscript range errors.
DB1	Allows generation of object code which calls paragraph trace feature to trace the flow of the COBOL program.
E	Allows output of a COBOL compilation to be added to the system library with EDITLIB. This parameter has no application for KRONOS.
E=program-name	The main overlay of the program is named by program-name, which must not exceed five characters.
F	All data name entries described as COMPUTATIONAL are interpreted as COMPUTATIONAL-1 when this parameter is included.
H	Increases sort efficiency if no OPEN statement is executed during SORT input or output procedure. If this parameter is not used, unnecessary space is reserved for all program files. If a file is opened during input/output and H is specified, the run is terminated.
I	Specifies that compiler input is to be obtained from file INPUT. If this parameter is omitted, file INPUT is also assumed.
$I=lfn_2$	Compiler input is obtained from file lfn_2. Tape files must be BCD.
L	Specifies that the listing is to be written on file OUTPUT. If the L parameter is omitted, this option is assumed.
$L=lfn_3$	Output is to be written on file lfn_3.
L=0	Suppresses output except for errors.

p_i	Description

The L parameter may appear with one of the following suffixes to produce special listings.

Suffix	Meaning
C	Listing of items copied from user libraries
M	Data map
O	Object code in octal
R	Data-name and procedure-name cross-reference list with pointers to source lines
X	Extended diagnostics

N	Directs COBOL compiler to issue an E type diagnostic if a non-ANSI feature is detected.
OB	Separates binary overlay segments from main program and writes them on LGO. If this parameter is omitted, this option is assumed.
OB=lfn_4	Specifies that binary output from overlay segments is to be written on lfn_4.
P	Allows the user to execute a strictly ANSI program.
S	Specifies that external references are to be satisfied from the source library file COLIB. If this parameter is omitted, this option is assumed.
S=file-name	Specifies that external references are to be satisfied from the file identified, which contains the COBOL source library.
SUB	Suppresses data division binary output that duplicates output from a separately compiled main program, except for working storage and constant sections, so that the subprogram and main program can be loaded together.
SUBM	Identifies the COBOL program as a subprogram so that it can be called from a main program written in another language.
T	Requests a tape sort rather than a disk sort. This requires four files which may be assigned to the disk.
U	Allows use of a collating sequence other than the standard CDC sequence.
V	If the loaded program is to be saved using NOGO with the file name specified, the V parameter must be specified for all COBOL/SORT programs. Specifying this parameter causes the SORT code to be included in the program rather than being loaded dynamically.
W	An independent segment (priority number 50 through 99) may overlay or be overlayed by an overlayable fixed segment or another independent segment. In COBOL version 4 and for ANSI programs, an independent segment is made available in its initial state. To override this usage and provide independent segments in their last used state so that COBOL 3 programs can be run without change, the W parameter must be included.
Z	Ensures compatibility with COBOL version 3 and turns on the C and W parameters.

ALGOL 3 CARD

The ALGOL control card is used to call the ALGOL compiler to the control point. ALGOL requires a minimum of 27,000 octal locations. External references are satisfied from SYSMISC

The control card format is:

$$\underset{I}{ALGOL}, p_1, p_2, \ldots, p_n.$$

The following parameters may be supplied. Except for the I parameter, the absence of any parameter suppresses the corresponding option.

p_i	Description
A	Specifies that the assembly language encoded form of the object code is to be written on file OUTPUT.
A=lfn_1	Specifies that the assembly language encoded form of the object code is to be written on file lfn_1.
B	Specifies that the assembly language encoded form of the object code is to be written on file PUNCH.
B=lfn_2	Specifies that the assembly language encoded form of the object code is to be written on file lfn_2.
C	Performs checks on actual parameters in object program to ensure correspondence with number and kind of formals. Also perform real to integer transformations on call-by-name parameters. These checks are performed in the absence of C option except when Q is selected.
D	Debugging mode. Examine source program for debugging directives and install debugging code where required. If D is omitted, debugging directives in source program are ignored.
F	Compile with full checking of array bounds, formal parameter specifications, and real to integer transformations, if necessary, on call-by-name parameters. Absence of F implies F. This parameter is not necessary; it is supplied only for compatibility with ALGOL 2.0.
G†	Specifies that file LGO contains user subprogram input exclusively. This option can be specified only when the S option is specified and the U option is not specified. This option suppresses the explicit or implicit I option.
G=lfn_3†	Specifies that file lfn_3 contains user subprogram input exclusively.
I	Input is to be obtained from file INPUT. If this parameter is omitted, file INPUT is assumed.
I=lfn_4	Input is to be obtained from file lfn_4.
L	Specifies that a listing of the source input is to be written on file OUTPUT.

†KRONOS does not support segmentation.

P_i	Description
$L=lfn_5$	Specifies that the source input is to be written on file lfn_5.
M	Produce a map, at compile time, of source input block structure and relative stack locations of identifiers and descriptors. The map is produced, following the source listing, on the file declared by the L option or on the OUTPUT file if L is absent.
N	Suppresses array bound checking in the object program.
O	Optimization of generated code. The code which would be generated in absence of this option is optimized by eliminating duplicate fetches and stores and by scheduling instructions. Provides maximum utilization of X registers (all 6000 series CPU) and functional units (6600 CPU only).
P	Specifies that the binary form of the object code is to be written on file PUNCHB. If P is omitted, no file is generated.
$P=lfn_6$	Specifies that the binary form of the object code is to be written on file lfn_6 in PUNCHB format.
Q	Perform optimization of code produced for loops and array addressing. Do not check array bounds in the object program. Do not check correspondence of actual parameters with formal specifications in the object program.
R†	Specifies that the program on file SEGMENT is to be executed in segmented form. If the S option is not specified, the system assumes that the segmented object code already exists. The use of this option suppresses the I, U, and G options. The file must be a disk file.
$R=lfn_7$†	Specifies that the object program on file lfn_7 is to be executed in segmented form.
S†	Specifies that the object program is to be written in segmented form on file SEGMENT. This option suppresses the X option but not the P parameter. The file must be a disk file.
$S=lfn_8$†	Specifies that the object program is to be written in segmented form on file lfn_8.
U	Specifies that file LGO contains user input, supplementary to the file specified with the I option. This is valid only when the S option is specified.
$U=lfn_9$†	Specifies that file lfn_9 contains user input, supplementary to the input file.
Z	Syntax errors only. Compile according to other options selected unless the program has syntax errors (detected by PASS 1). If syntax errors are detected, compilation terminates when the end of program is read and syntax diagnostics are listed. This provides a preprocessing mode of compilation for debugging purposes which may overcome error recovery problems.

†KRONOS does not support segmentation.

ALGOL 4 CARD

The ALGOL control card is used to call the ALGOL 4 compiler to the control point. ALGOL 4 requires a minimum of 46,000 octal locations to compile. External references are satisfied from ALGOLIB.

The control card format is:

ALGOL, $p_1, p_2, \ldots, p_n$.

The following parameters may be supplied. The absence of any parameter suppresses the corresponding option.

p_i	Description
A	Specifies that the assembly language form of the object code is to be written on the file specified by the L option.
A=0	No assembly language listing.
B	The output object program is to be written on file LGO.
B=lfn_1	The output object program is to be written on file lfn_1.
B=0	No binary object program.
C=n	Comments interpretation for special delimiters. This option requires the compiler to search comments for special delimiters interpretation.

n	Description
0	No comments interpretation.
1	Debugging directives which are present in comments are detected by the compiler and cause debugging code to be inserted into the object program.
2	Overlay directives which are present in comments are detected by the compiler and cause overlay directives in loader input format to be inserted into the object program.
3	Array bound checking directives which are present in comments are detected by the compiler.

Multiple selection for the C option can be performed by separating each value by a slash. For example: C=3/2/1 is acceptable.

D	The symbol file is created on file DUMPFIL.
D=lfn_2	The symbol file is created on file lfn_2.
D=0	The symbol file is suppressed.
E	The job is aborted to an EXIT control card if a fatal error occurs during compilation.
E=0	Suppress abort in the event of a fatal error.

p_i	Description
F	If a fatal error is found in the first pass, compilation is terminated at the end of this pass.
F=0	Continue until the normal end of compilation.
G	Compilation will consider stack swapping to ECS and when the program is executed, the swapping procedures will be activated automatically.
G=0	No swapping will be considered. This option must not be selected when using a machine without ECS; otherwise, unpredictable results leading to a fatal error will be obtained.
I	Source input is on file INPUT.
I=lfn_3	Source input is on file lfn_3.
I=0	No source input.
K=n	Input record size. n — The number of significant characters to be interpreted by the compiler on the source card image. The default value of K is 72.
L	The source program is listed with fatal diagnostics on file OUTPUT.
L=lfn_4	The source program is listed with fatal diagnostics on file lfn_4.
L=0	Fatal diagnostics only are listed on file OUTPUT.
N	A listing of advisory diagnostics is generated on the file specified by the L option.
N=0	Advisory diagnostics are suppressed; only diagnostics fatal to code generation are listed.
O=n	Specifies the level of compiled optimization. n — Description 0 — The program is compiled in fast compile mode. 1 — Linguistic optimization is performed by optimizing procedure calling. 2 — Optimizations of O=1 and also subscript and for statement optimizations.

p_i	Description
P	Specifies that the assembly language form of the object code is to be punched in standard assembly language card format on file PUNCH.
P=lfn_5	Assembly language is to be punched on file lfn_5.
P=0	Assembly language punching is suppressed.
Q	Create interactive file on file QFILE for ALGOL Interactive Debugging Aids (AIDA).
Q=lfn_6	Create interactive file on file lfn_6.
Q=0	Suppress interactive compilation and file. Q may not be specified if the S option is selected.
R	A cross-reference map is produced and listed at compile time for identifiers in the source program on the file specified by the L option.
R=0	No cross-reference map is produced.
S=n	Array storage location. n — Description 0 — All arrays are allocated to CM. 1 — Virtual arrays are allocated to ECS. S may not be specified if the Q parameter is specified.
U	Specifies that the file COMPILE contains user implicit outer block head input, supplementary to the file specified with the I option.
U=lfn_7	The source program is preceded by the implicit outer block head list on file lfn_7.
U=0	There is no file for implicit outer blocks.
X	Allow real-integer (or integer-real) correspondence between formal and actual parameters, and in the case real to integer perform the conversion.
X=0	Forbid any real-integer (or integer-real) correspondence between formal and actual parameters. Selection of this option will significantly degrade the performance of the program.

SORTMRG CARD

The SORTMRG control card calls Sort/Merge to process a logical record of directives. The minimum memory requirements for Sort/Merge is 25,000 octal locations.

The control card format is:

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

p_i	Description
I	Sort/Merge directives are on file COMPILE. If I is omitted directives are on file INPUT.
I=lfn_1/r_1	Sort/Merge directives are on file lfn_1 with the following rewind options:

r_1	Description
R	File is rewound before opening. If system INPUT file is selected, R should not be specified.
NR	File is not rewound before opening. This option is assumed if r_1 is not specified.

O	Listings will be written to the file OUTPUT. If the O parameter is omitted, this option is assumed.
O=lfn_2/r_2	Listings are written to file lfn_2, with the following rewind options:

r_2	Description
R	File is rewound before opening. If the system OUTPUT file is indicated, R should not be specified.
NR	File is not rewound before opening. This option is assumed if r_2 is not specified.

OWN	Owncode binaries are on file LGO. If the OWN parameter is omitted, owncode binaries are on file INPUT.
OWN=lfn_3/r_3	Owncode binaries are located on file lfn_3, with the following rewind options:

r_3	Description
R	File is rewound before opening. If the system INPUT file is indicated, R should not be specified.
NR	File is not rewound before opening. This option is assumed if r_3 is not specified.

p_i	Description
MO=n	Intermediate merge order; $2 \leq n \leq 64$. In general, higher merge orders produce faster sorts at the expense of greater field length requirements. If insufficient core is available to merge at the requested order, a fatal error occurs; and a diagnostic indicates how much additional core is required.
MO=*n	Intermediate merge order; $2 \leq n \leq 64$. If insufficient core is available to merge at the requested order, merge will take place at a smaller order and an informative diagnostic will be issued.
MO omitted	The installation default merge order is used (release system default is 5).

PERT66 CARD

The PERT66 card calls PERT/Time to process the user-supplied directives. The minimum memory requirement is 72,000 octal locations.

The control card format is:

PERT66.

PERT/Time 1.2 is not provided with the KRONOS 2.1 base system release package; refer to the Installation Handbook for a description of the special procedures for using PERT/Time.

SIMULA CARD

The SIMULA control card calls the compiler to the control point. The minimum memory requirement for SIMULA is 30,000 octal locations. Externals are satisfied from user library SYSMISC.

The control card format is:

SIMULA, p_1, p_2, . . . , p_n.

The following parameters may be supplied. Except for the I parameter the absence of any parameter suppresses the corresponding option.

p_i	Description
A	Specifies that the assembly language encoded form of the object code is to be written on file OUTPUT.
A=lfn_1	Specifies that the assembly language encoded form of the object code is to be written on file lfn_1.
B	Specifies that the assembly language encoded form of the object code is to be written on file PUNCH.
B=lfn_2	Specifies that the assembly language encoded form of the object code is to be written on file lfn_2.

p_i	Description
D	Specifies that debugging code is to be generated.
G†	Specifies that file LGO contains user subprogram input exclusively. This option can be specified only when the S option is specified and the U option is not specified. This option suppresses the explicit or implicit I option.
G=lfn_3†	Specifies that file lfn_3 contains user subprogram input exclusively.
I	Input is to be obtained from file INPUT. If this parameter is omitted, file INPUT is assumed.
I=lfn_4	Input is to be obtained from file lfn_4.
L	Specifies that a listing of the source input is to be written on file OUTPUT.
L=lfn_5	Specifies that the source input is to be written on file lfn_5.
N	Suppresses array bound checking in the object program.
P	Specifies that the binary form of the object code is to be written on file PUNCHB. If P is omitted, no file is generated.
P=lfn_6	Specifies that the binary form of the object code is to be written on file lfn_6 in PUNCHB format.
R†	Specifies that the program on file SEGMENT is to be executed in segmented form. If the S option is not specified, the system assumes that the segmented object code already exists. The use of this option suppresses the I, U, and G options. The file must be a disk file.
R=lfn_7†	Specifies that the object program on file lfn_7 is to be executed in segmented form.
S†	Specifies that the object program is to be written in segmented form on file SEGMENT. This option suppresses the X option but not the P parameter. The file must be a disk file.
S=lfn_8†	Specifies that the object program is to be written in segmented form on file lfn_8.
U	Specifies that file LGO contains user input, supplementary to the file specified with the I option. This is valid only when the S option is specified.
U=lfn_9†	Specifies that file lfn_9 contains user input, supplementary to the input file.
X	Specifies that the object code is to be written on file LGO.
X=lfn_{10}	Specifies that the object code is to be written on file lfn_{10}.

† KRONOS does not support segmentation.

SIMSCRIPT CARD

The SIMSCRIPT control card calls the compiler to process the user's program. The minimum memory requirement for SIMSCRIPT is 55,000 octal locations. External references for SIMSCRIPT programs are satisfied from SIMLIB.

The control card format is:

SIMI5(I=lfn_1, L=lfn_2, A=lfn_3, B=lfn_4, G=g, D=d)

I=lfn_1	Specifies that lfn_1 is the input file. If the file name or the entire parameter is omitted, the input is obtained from file INPUT.
L=lfn_2	Specifies that output is to be written on lfn_2. If the file name or the entire parameter is omitted, output is written on file OUTPUT. Output is suppressed when the user specifies L=0.
A=lfn_3	lfn_3 is the file on which the intermediate assembler code is written. If A=0, no listing is generated (default is A=0).
B=lfn_4	lfn_4 is the file on which the binary object code is written. If B=0, no binary file is produced (default is B=0).
G=g	Compile error override flag 0 Abort if compile error (default) 1 Load and execute regardless of compile errors
D=d	Debug mode flag 1 Normal (debug) mode (default) 0 Production mode

BASIC CARD

The BASIC control card is used to call the BASIC compiler to the control point. The minimum field length required is 30,000 octal locations. Normally, BASIC compiles directly to central memory. All object-time routines are contained within the compiler; therefore, no external references are generated. However, the user can include the B parameter to generate relocatable code. When this code is loaded, externals are satisfied from SYSMISC.

The control card format is:

BASIC, p_1, p_2, . . . , p_n.

The following parameters may be supplied.

p_i	Description
L	Source listing, diagnostics, and execution output on file OUTPUT
L=lfn_1	Source listing, diagnostics, and execution output on lfn_1
L omitted	Diagnostics and execution output on file OUTPUT
K	Diagnostics and execution output on file OUTPUT
K=lfn_2	Diagnostics and execution output on lfn_2

P_i	Description
K omitted	Diagnostics and execution output on file OUTPUT
I	Source input from file INPUT
I=lfn_3	Source input from lfn_3
I omitted	Source input from file INPUT
B	Relocatable code on file LGO
B=lfn_4	Relocatable code on lfn_4
B omitted	No relocatable code generated
A	Assembly listing on file OUTPUT
A=lfn_5	Assembly listing on lfn_5
A omitted	No assembly listing
N	Inhibits program execution
N=lfn_6	Inhibits program execution. Absolute code is written on file lfn_6.
N omitted	Program is executed in place (no loading occurs)

If both the K and L options are specified on the control card, K is ignored.

TSRUN CARD

The TSRUN control card calls the time-sharing FORTRAN compiler to the control point. The TSRUN compiler can also be used as a batch compiler. The minimum memory requirement for TSRUN is 46,000 octal locations. Because TSRUN is written in Chippewa format (refer to appendix D), all object-time routines are satisfied from the COS library at compilation time. No user library is specified when loading COS programs.

Chippewa format presents several problems the user should be aware of when modifying TSRUN routines. Object-time routines in Chippewa format are relocatable. Bits 15, 16, and 17 (the upper three bits of the address field) of 30-bit instructions are used for loader control. If constants extend into these bits, they are treated as addresses and relocated when loaded. Constants that extend into this field should be defined outside the block that is identified as that to be relocated. The relocatable block is identified in the first two words of the routine as shown in the following example.

```
+ VFD    42/0Lentry, 18/end
+ VFD    18/erc, 18/end, 24/n
  BSSZ   n
```

entry	Entry point and beginning of relocatable code
erc	End of relocatable code
end	End of code
n	Number of parameters passed to subroutine

Addresses between entry and erc are modified according to bits 15, 16, and 17 of 30-bit instructions; constants that interfere with these bits should be defined between erc and end. The addresses pointing to these constants are modified, but the constants are not.

The control card format is:

TSRUN, cm, if, rf, bl, fl, cl.

The following are the order-dependent parameters.

cm Compile mode option; if this parameter is omitted, the system assumes cm=G; if this parameter is unrecognizable, the system assumes cm=S.

cm	Mode
D	Compiles with binary output; the program is not listed or executed.
G	Compiles and executes the program. No listing is generated unless LIST cards appear in the deck.
L	Compiles with source and object list; the program is not executed.
O	Compiles and executes the program; no listing is generated.
P	Compiles with source list and punches deck on file PUNCHB; the program is not executed.
S	Compiles with source list; the program is not executed.

if Name of file containing compiler input; if this parameter is omitted, the system assumes if=INPUT.

rf Name of file to receive binary information; if this parameter is omitted, the system assumes rf=LGO.

bl Object program I/O buffer length (octal); if this parameter is omitted, the system sets bl = 1001_8.

fl Object program field length (octal); if this parameter is omitted, the system sets fl=field length at compile time.

cl Object program common length.

CHECKPOINT/RESTART 12

A job may be terminated at any time as the result of system, operator, or programmer error. For some jobs it becomes more advantageous to accept the overhead of checkpoint procedures than to run the risk of losing the entire job output. The checkpoint/restart feature is implemented through the CKP control card and the RESTART control card.

CKP CARD

The CKP control card causes a checkpoint dump to be taken.

The control card format is:

CKP(lfn_1,lfn_2,...,lfn_n)

lfn_i Specifies a file to be included in the checkpoint dump. If no files are specified, all files local to the job at the time the CKP card is processed will be checkpointed.

Each time a CKP card is processed, the system takes a checkpoint dump. The dump is written on the tape or mass storage checkpoint file specified on a REQUEST, ASSIGN, or LABEL control card. The dump consists of a copy of the user's central memory, the system information used for job control, and the names and contents of all assigned files explicitly or implicitly identified by the CKP card. These files are:

- INPUT, OUTPUT, PUNCH, PUNCHB, P8, CCCCCC0, and LGO. These files are always included in the checkpoint dump.
- Library type files, working copies of indirect access files, and some direct access files. If one of these types of files is specified on the CKP card, it is included in the checkpoint dump, and all other files of that type are excluded. If no files are specified, all files of these types assigned to the job are included in the dump.

Each checkpointed file is copied according to the last operation performed on it. If the last operation was a write, the file is copied from the BOI to its position at checkpoint time; only that portion will be available at restart time. The file is positioned at the latter point.

If the last operation was a read and the EOI was not detected, the file is copied from its position at checkpoint time to the EOI; only that portion will be available at restart time. The file is positioned at the former point. If the last operation was a read and the EOI was detected, no copy is performed.

The exception to this rule is the type of operation performed on execute-only direct access files. If a dump is specified for this type of file, its name and associated system information are copied but the contents of the file itself is not copied. Thus, if the user attempts to resume from such a dump, RESTART will be unable to retrieve that file and will abort. The user can avoid this by selecting the NA and FC options of the RESTART card and retrieving the file himself.

If the checkpoint file is to reside on mass storage, the user must include a SAVE or DEFINE control card in the checkpoint job and a GET or ATTACH control card in the restart job.

If the checkpoint file is to reside on magnetic tape, care should be taken to use a labeled or nonblank tape. An unlabeled blank tape (one which has never been used) cannot be specified as the checkpoint file since the checkpoint program attempts to read the tape to determine the number of the last checkpoint. The tape subsystem then aborts the job with a blank tape read message.

The system numbers checkpoints starting at one and incrementing by one to a limit of 4095. At this point, a second cycle of numbering begins, again starting at one. An example showing how to restart from a specific checkpoint is given in the RESTART control card section.

RESTART CARD

The RESTART control card directs the system to restart a previously terminated job from a specified checkpoint.

The control card format is:

RESTART(lfn, nnnn, x_i)

lfn — Identifies the checkpoint file; the user must have write permission to lfn.

nnnn — Number of the checkpoint from which to restart; if nnnn is *, the last available checkpoint on lfn is used; if nnnn is omitted, the first checkpoint is used. The nnnn parameter can be obtained from the CHECKPOINT nnnn COMPLETE messages issued to the user's dayfile in response to CKP control cards.

x_i — Any of the following in any order:

- RI — If this parameter is included, the control card file on lfn is not restored. The control card file of this restart job at its current position is used instead. If this parameter is not included, the entire control card file of the checkpointed job is restored and set to its position at checkpoint time; any control cards following RESTART are not processed.
- NA — If this parameter is included, RESTART is not aborted if a required file is not available. Also, if NA is included and a read parity error occurs in an attempt to obtain a file from checkpoint nnnn, RESTART selects checkpoint nnnn-1 if it is available.
- FC — Normally RESTART restores all files included in the specified checkpoint. However, if this option is selected, RESTART first checks if a file is already local to the restart job. If it is, RESTART does not replace it with the file on the checkpoint dump.

The user must assign lfn to his job before the RESTART card is processed. He must include a REQUEST, ASSIGN, or LABEL control card if lfn resides on magnetic tape or a GET or ATTACH control card if lfn resides on mass storage.

Checkpoint dumps are numbered in ascending order from 1 to 4095. When nnnn=4095, the numbering sequence begins again at nnnn=1. The value of nnnn depends on the structure of the checkpoint file, as defined by the CK and CB parameters of the REQUEST, ASSIGN, or LABEL control cards.

If CK was specified when the checkpoints occurred, each dump is appended to the checkpoint file, and therefore, all dumps up to the time the job aborted are available for restart. The user may specify a particular checkpoint dump in the following manner.

Assume a CK file of the name CHKFILE is being used and checkpoint number 4095 has been passed. The job is terminated at checkpoint number 10 in the second cycle of numbering. To restart the job from checkpoint 4 of the second numbering cycle, the following control cards can be used.

SKIPR(CHKFILE,8196)	There are two records for every checkpoint and 4098 checkpoints must be skipped to reach checkpoint 4 of the second numbering cycle.
COPYBR(CHKFILE,AA,2)	The fourth checkpoint is copied to file AA. At this point, file CHKFILE is not positioned correctly for subsequent checkpoints. If the user intends to continue checkpointing on this file, a: BKSP,CHKFILE. card should be included.
RESTART(AA...)	The job is restarted from file AA using the fourth checkpoint.

If CB was specified when the checkpoints occurred, each dump is written over the preceding dump, and therefore, only the last dump is available. If two REQUEST, ASSIGN, or LABEL cards are submitted, successive CB-type dumps are alternated between two files; therefore, the last two dumps are available.†

All files copied by RESTART are made local to the restart job. Therefore, the user must make sure that any direct access files are not lost. For example, assume that direct access files X, Y, and Z are attached to a job. The job is then checkpointed and X, Y, and Z are copied to the checkpoint file lfn. To retain these files as direct access files during restart, the user should include the following sequence of control cards.

```
PURGE(X,Y,Z)
DEFINE(X,Y,Z)
RESTART(lfn,nn,x_i)
```

If the information table associated with a file was included on the checkpoint file, but the file itself was not copied, RESTART issues the appropriate commands to retrieve the file.

†If alternate checkpoint files are used and a read parity error occurs in an attempt to read the last checkpoint, RESTART will abort even if the NA option was selected.

This section contains a description of central memory dumps and their use as a debugging aid. This information should be of considerable assistance to the user in finding errors in his program.

CENTRAL MEMORY DUMPS

The first line of a dump gives the boundaries of the memory that is dumped, relative to the user's field length. Four central memory words are printed per line, with the address of the leftmost word printed on the left-hand side of the page. When the phrase DUPLICATED LINES appears within the dump, all groups of four words not printed are exactly like the last group of four words. Each word is divided into four groups of 15_{10} bits, with the octal representation printed. Figure 1-13-10 is an example of a central memory dump. Section 9 describes the options of the DMP control card that can be used to obtain various dumps.

The user may also dump his exchange package. Figure 1-13-1 illustrates the format of the actual exchange package.

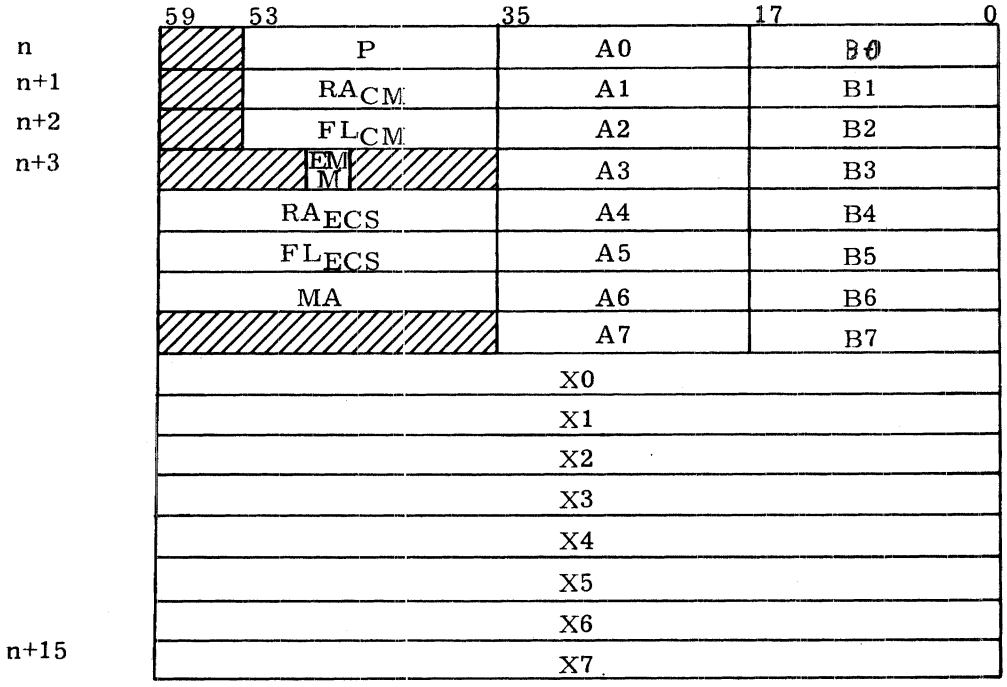


Figure 1-13-1. Exchange Package

P	Program address currently being executed
RA	Reference address, beginning address of the associated field length [central memory (CM), extended core storage (ECS)]
FL	Field length
EM-M	Program error exit mode (refer to section 6)
MA	Monitor address
An	Address registers
Bn	Increment registers
Xn	Operand registers

When the user requests this form of a dump, he also gets the following information.

- The contents of memory at the address contained by the A registers, identified as (An)
- The contents of RA (reference address) and RA+1, identified as (RA) and (RA+1), respectively
- 40_8 locations before and after the address contained in P (100_8 locations total)

Figure 1-13-9 illustrates an example of this exchange package dump.

GENERATING MEANINGFUL DUMPS

The following methods are used to generate meaningful central memory dumps.

- Error Exit Control

  By using the EREXIT macro within his COMPASS program, the user can direct execution when certain errors occur, rather than having his program completely halt execution. This enables him to use it as a checkpoint method (that is, to save generated data to this point). It could also enable him to do further calculations or to write pertinent data to an output file. Refer to section 6, volume 2 for a description of this macro.

- EXIT/NOEXIT/ONEXIT Control

  Once program execution ceases, due to an error condition, and control card processing is resumed, the user can direct which cards are to be processed through the use of the EXIT, NOEXIT, and ONEXIT cards. Upon an error condition, the user can issue the DMP control cards to obtain appropriate dumps. For a detailed description of these control cards, refer to section 6.

- Dumps may also be generated under control of the user's program through the use of the SYSTEM macro. The FORTRAN user can generate dumps by calling the DUMP subroutine.

READING CM DUMPS

Figures 1-13-2 through 1-13-10 are output from a FORTRAN program source deck processed by the following sequence of control cards.

```
TESTA(CM50000,T10)
USER(ABCD,PASS,FAMA)
SETCORE(0)
MAP(FULL)
FTN.
LGO.
OVLA.
DMP.
DMP,10000.
```

The source deck in the example consists of four parts.

- Main program (main overlay)
- Function subprogram
- Subroutine subprogram
- Primary overlay

Each part is listed separately followed by the corresponding address assignments, such as: statement assignments, variable assignments, program length, unused compiler space, etc. (refer to Figure 1-13-2).

Figures 1-13-6 and 1-13-7 illustrate the load map generated by the MAP control card. The load map gives the address and length of all block assignments and the address and references of all entry points. Maps are listed separately for each overlay. Finally, the load map contains the field length required to load and run the program.

Output generated by the program follows the load map (refer to Figure 1-13-8).

Figure 1-13-9 and 1-13-10 illustrate the central memory dumps generated by the DMP. and DMP,5000. control cards, respectively.

The following three examples illustrate the use of these dumps to obtain specific information.

Example 1: (Finding Data Locations in a Core Dump)

Referring to Figure 1-13-2, the variable I is used as the control variable in the DO loop defined by statements 10 through 20. To find the value of I at job termination, the following steps must be performed.

1. Find I in the variable assignments (lower half of Figure 1-13-2), noting I is at relative address 4167_8.
2. Find the first word address (FWA) of the main overlay TESTA. (Refer to the load map, Figure 1-13-6.) The FWA of TESTA is 143_8.
3. Add ($143_8 + 4167_8 = 4332_8$) to obtain the absolute address of I.
4. In Figure 1-13-10, address 4332 contains 0013_8 (11_{10}). This should be the last value of I.

Example 2: (Finding Data Locations in a Core Dump)

To find the variable B(3), the following points must be considered.

- Find B in the variable assignments (lower half of Figure 1-13-2). The value is 12, which means that B begins at relative location 12_8 of common block AAA. By referring to the map (Figure 1-13-6), note that AAA begins at absolute address 101_8. Therefore, $101_8 + 12_8$ (relative location of B) equals 113_8, the beginning address of array B. B(1) is 113_8, and the address of B(3) is 115_8.
- The location in core of the B array is illustrated in Figure 1-13-10.

Example 3: (Finding an Address Within the Program)

By referring to Figure 1-13-9, note that the program stopped at address 10114 (the value of P). To find where this is in the program, the following points must be considered.

- Figure 1-13-6 or 1-13-7 contains the routine addresses.
- Figure 1-13-6 illustrates that routine SYS.RM is at 10114 which means the program ended in routine SYS.RM.

```
      1              OVERLAY(OVLA,0,0)
                      PROGRAM TESTA(INPUT,OUTPUT)
                     COMMON//E(100)
                     COMMON/AAA/A(10),B(10),C(3,3)
      5              COMMON/BLOCKA/BLK(5)
                     DIMENSION N(50)
                     DATA (A(I),I=1,10)/1.,2.,3.,4.,5.,6.,7.,8.,9.,10./
                     DATA (B(I),I=1,5)/100.,200.,300.,400.,500./
                     DATA (B(I),I=6,10)/600.,700.,800.,900.,1000./
     10              DATA C(1,1),C(1,2),C(1,3)/101.,202.,303./
                     DATA C(2,1),C(2,2),C(2,3)/2.1,2.2,2.3/
                     DATA C(3,1),C(3,2),C(3,3)/3.1,3.2,3.3/
                     DATA (N(J),J=1,50)/50*123/
                     CALL OVERLAY(4HOVLA,1,0)
     15              CALL PRNT(BLOCKA)
                     DO 30 I=1,10
                     A(I)=A(I)*A(I)
                     A(I)=TRY(A(I),A(I))
                  30 B(I)=TRI(A(I),A(I))
     20              DO 35 J=1,5
                  35 BLK(J) = A(J)+A(2*J)
                     CALL OVERLAY(4HOVLA,1,0)
                     CALL PRNT(BLOCKA)
                     C(1,1)=J
     25              END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
4107 TESTA

VARIABLES SN TYPE RELOCATION

0	A	REAL	ARRAY	AAA	12	B	REAL	ARRAY	AAA
0	BLK	REAL	ARRAY	BLOCKA	4166	BLOCKA	REAL		
24	C	REAL	ARRAY	AAA	0	E	REAL	ARRAY	/ /
4167	I	INTEGER			4170	J	INTEGER		
4171	N	INTEGER	ARRAY						

Relative address location of variable B

Common block containing variable B

Relative address location of variable I

FILE NAMES MODE

0	INPUT	2041	OUTPUT

EXTERNALS TYPE ARGS

OVERLAY		3	PRNT		1
TRI	REAL	2	TRY	REAL	2

STATEMENT LABELS

0	30	0	35

LOOPS	LABEL		INDEX	FROM-TO	LENGTH	PROPERTIES
4115	30	*	I	16 19	143	EXT REFS
4134	35		J	20 21	48	INSTACK

COMMON BLOCKS LENGTH

/ /	100
AAA	29
BLOCKA	5

STATISTICS

PROGRAM LENGTH	1548	108
BUFFER LENGTH	41038	2115
CM LABELED COMMON LENGTH	428	34
CM BLANK COMMON LENGTH	1448	100

Figure 1-13-2. Main Program of Main Overlay (0,0)

```
      1               FUNCTION TRY(A,B)
                   10 TRY = SQRT(A)+SQRT(B)
                      RETURN
                      ENTRY TRI
      5               IF (A.LE.B) 10,20
                   20 TRY = SQRT(A)-SQRT(B)
                      RETURN
                      END



        SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
   14  TRI              4  TRY

VARIABLES     SN  TYPE           RELOCATION
    0  A         REAL                F.P.          0  B         REAL                F.P.
   34  TRY       REAL

EXTERNALS          TYPE   ARGS
       SQRT        REAL      1 LIBRARY

STATEMENT LABELS
    7  10                                    0  20          INACTIVE

STATISTICS
  PROGRAM LENGTH                  358      29
```

Figure 1-13-3. Function Subroutine of Main Overlay (0, 0)

```
      1               SUBROUTINE PRNT(A)
                      COMMON/ /D(100)
                      COMMON/AAA/P(29)
                      COMMON/A/SUB(5)
      5               B=0
                      DO 50 I=1,29
                   50 B=B+P(I)
                      PRINT 55,B,(SUB(I),I=1,5)
                   55 FORMAT (1X,6F17.7)
     10               RETURN
                      END



        SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
    3  PRNT

VARIABLES     SN  TYPE           RELOCATION
    0  A         REAL       *UNUSED    F.P.          26  B         REAL
    0  D         REAL        ARRAY     / /           27  I         INTEGER
    0  P         REAL        ARRAY     AAA            0  SUB       REAL        ARRAY    A

FILE NAMES         MODE
       OUTPUT      FMT

STATEMENT LABELS
    0  50                                   24  55       FMT

LOOPS  LABEL     INDEX      FROM-TO     LENGTH     PROPERTIES
   11  50        I            6 7          38      INSTACK

COMMON BLOCKS    LENGTH
       / /          100
       AAA           29
       A              5

STATISTICS
  PROGRAM LENGTH                  308      24
  CM LABELED COMMON LENGTH        428      34
  CM BLANK COMMON LENGTH         1448     100
```

Figure 1-13-4. Subroutine of Main Overlay (0, 0)

```
      1                OVERLAY(OVLA,1,0)
                       PROGRAM OVL10
                       COMMON/AAA/W(29)
                       PRINT 105,(W(I),I=1,7)
      5                PRINT 105,(W(I),I=8,14)
                       PRINT 105,(W(I),I=15,21)
                       PRINT 105,(W(I),I=22,28)
                       PRINT 106,(W(29))
                   105 FORMAT(1X,7F17.7)
     10            106 FORMAT (4X,F17.7)
                       END


        SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
    3  OVL10

VARIABLES      SN  TYPE           RELOCATION
   47  I           INTEGER                                  0  W       REAL        ARRAY     AAA

FILE NAMES          MODE
       OUTPUT       FMT

STATEMENT LABELS
   43  105      FMT                              45  106      FMT

COMMON BLOCKS    LENGTH
       AAA           29

STATISTICS
  PROGRAM LENGTH                  47B       39
  BUFFER LENGTH                    1B        1
  CM LABELED COMMON LENGTH        35B       29
```

Figure 1-13-5. Main Program of Primary Overlay (1, 0)

```
LOAD MAP - TESTA                                CYBER LOADER 1.0-U401      75/06/02. 13.24.01.      PAGE      1
OVERLAY(OVLA,0,0)


- OVERLAY(OVLA,0,0)

 FWA OF THE LOAD            101
 LWA+1 OF THE LOAD        17032

 TRANSFER ADDRESS -- TESTA          4252


 PROGRAM AND BLOCK ASSIGNMENTS.        Address of common block /AAA/ containing
                                       variable arrays A, B, and C

 BLOCK       ADDRESS   LENGTH     FILE       DATE      PROCSSR VER LEVEL  HARDWARE   COMMENTS

 /AAA/           101       35     FWA of main overlay
 /BLOCKA/        136        5
 TESTA           143     4257     LGO        75/05/02 FTN      4.4 U401   666X I     OPT=1
 TRY            4422       35     LGO        75/05/02 FTN      4.4 U401   666X I     OPT=1
 /A/            4457        5
 PRNT           4464       30     LGO        75/05/02 FTN      4.4 U401   666X I     OPT=1
 /QB.IO./       4514      134
 /IOCON./       4650       42
 COMIO=         4712       63   SL-FORTRAN   75/03/27 COMPASS  3. 75086              COMMON CODED I/O ROUTINES AND CONSTANTS.
 FLTOUT=        4775      311   SL-FORTRAN   75/03/27 COMPASS  3. 75086              COMMON FLOATING OUTPUT CODE
 FMTAP=         5306      351   SL-FORTRAN   75/03/27 COMPASS  3. 75086              CRACK APLIST AND FORMAT FOR KODER/KRAKER.
 FORSYS=        5657      654   SL-FORTRAN   75/03/27 COMPASS  3. 75086              FORTRAN OBJECT LIBRARY UTILITIES.
 GETFIT=        6533       42   SL-FORTRAN   75/03/27 COMPASS  3. 75086              LOCATE AN FIT GIVEN A FILE NAME.
 KODER=         6575      457   SL-FORTRAN   75/03/27 COMPASS  3. 75086              OUTPUT FORMAT INTERPRETER.
 OUTC=          7254      172   SL-FORTRAN   75/03/27 COMPASS  3. 75086               FORMATTED WRITE FORTRAN RECORD
 OUTCOM=        7446      153   SL-FORTRAN   75/03/27 COMPASS  3. 75086              COMMON OUTPUT CODE
 OVERLAY        7621      143   SL-FORTRAN   75/03/27 COMPASS  3. 75086              OVERLAY LOADING ROUTINE.
 SQRT           7764       43   SL-FORTRAN   75/03/27 COMPASS  3. 75086              COMPUTE THE SQUARE ROOT OF X. OPT=ALL.
 SYSAID=       10027        1   SL-FORTRAN   75/03/27 COMPASS  3. 75086              LINK BETWEEN SYS=AID AND INITIALIZATION CODE.
 SYS=1ST       10030       82   SL-FORTRAN   75/03/27 COMPASS  3. 75086              MATH LIBRARY LINK TO ERROR MESSAGE PROCESSOR.
 SYS.RM        10112       50   SL-SYSIO     75/03/27 COMPASS  3. 75086              PROCESS SYSTEM REQUEST.
 UCLOAD        10162      255   SL-SYSIO     75/03/27 COMPASS  3. 75086              L75086 LOADER USER CALL INTERFACE ROUTINE.
 /JMPS.RM/     10440       11
 LBUF.SQ       10451      133   SL-SYSIO     75/03/27 COMPASS  3. 75086
 /CON.RM/      10604        5
 CIO.RM        10612       34   SL-SYSIO     75/03/27 COMPASS  3. 75086
 /AOB.RM/      10646       10
 ERR.RM        10656      404   SL-SYSIO     75/03/27 COMPASS  3. 75086
 MOVE.RM       11262       64   SL-SYSIO     75/03/27 COMPASS  3. 75086
 CHWR.SQ       11346        7   SL-SYSIO     75/03/27 COMPASS  3. 75086
 MCT.RM        11355      233   SL-SYSIO     75/03/27 COMPASS  3. 75086
 /MEMC.RM/     11610        3
 /OPES.FO/     11613        1
 /OPEN.FO/     11614        7
 OPEN.RM       11623      234   SL-SYSIO     75/03/27 COMPASS  3. 75086
 OSUB.RM       12057       73   SL-SYSIO     75/03/27 COMPASS  3. 75086
 OPEN.SQ       12152      260   SL-SYSIO     75/03/27 COMPASS  3. 75086
 OPEX.SQ       12432       14   SL-SYSIO     75/03/27 COMPASS  3. 75086
 /PUT.RT/      12446       11
 RLEQ.RM       12457       42   SL-SYSIO     75/03/27 COMPASS  3. 75086
 /TERM.RM/     12521        1
 /PUT.FO/      12522        7
 PUT.SQ        12531     1277   SL-SYSIO     75/03/27 COMPASS  3. 75086
 WAR.SQ        14030      260   SL-SYSIO     75/03/27 COMPASS  3. 75086
 /CLSF.FO/     14310        7
 CLSF.RM       14317       23   SL-SYSIO     75/03/27 COMPASS  3. 75086
 CLSF.SQ       14342      131   SL-SYSIO     75/03/27 COMPASS  3. 75086
 /CLSV.FO/     14473        7
 CLSV.SQ       14502      123   SL-SYSIO     75/03/27 COMPASS  3. 75086
 /GET.FO/      14625        7
 /GET.BT/      14634        5
 /GET.RT/      14641       11
 GET.SQ        14652     1027   SL-SYSIO     75/03/27 COMPASS  3. 75086
 Z.SQ          15701      101   SL-SYSIO     75/03/27 COMPASS  3. 75086
 FSU.SQ        16002      106   SL-SYSIO     75/03/27 COMPASS  3. 75086
 BTRT.SQ       16110      114   SL-SYSIO     75/03/27 COMPASS  3. 75086
 LXER.SQ       16224      220   SL-SYSIO     75/03/27 COMPASS  3. 75086
 WEOX.SQ       16444      144   SL-SYSIO     75/03/27 COMPASS  3. 75086
 /SKFL.FO/     16610        7
 SKFL.SQ       16617       47   SL-SYSIO     75/03/27 COMPASS  3. 75086
  //           16666      144
```

Figure 1-13-6. Loader Map of Main Overlay (0, 0)

```
ENTRY POINTS.

ENTRY          ADDRESS          PROGRAM   REFERENCES

INPUTE           143            TESTA
OUTPUT=         2204                     PRNT          4503
TESTA           4252
TRY             4426            TRY      TESTA         4264
TRI             4436                     TESTA         4270
PRNT            4467            PRNT     TESTA         4256     4306
O8NTRY.         4541            FORSYS=  TESTA         4252
FECCHR.         4717            COMIO=   KODER=         6653     6740
FECPRT.         4737                     FMTAP=        5575
FEOFAL.         4775            FLTOUT=  KODER=        7035     7054     7075
FEOEOV.         5017                     KODER=        7054
FEOEXP.         5021                     KODER=        7036
FEORND.         5056                     KODER=        7023     7051     7072
FEOSGA.         5113                     KODER=        7013     7046     7066
FEOZRO.         5177                     KODER=        7024     7052     7073
FECNAP.         5316            FMTAP=   KODER=        6650     6735     6741     6761     6776     7236     7246
FECAP.          5324                     OUTC=         7360
FECFMT.         5343                     KODER=        6655     7201     7203     7213     7243
FECFMU.         5346                     KODER=        6614     6720
FECJP=          5451                     OUTC=         7353
FECLP.          5452                     KODER=        6647
FECRP.          5475                     KODER=        6650
FECEE.          5512                     KODER=        7001
FECV.           5560                     KODER=        6624
FECBUG.         5567                     KODER=        6575     6576
RANDOM.         5667            FORSYS=
LITE.           5681
FLSCM.          5682
FLLCM.          5683
END.            5715                     TESTA         4311

EXIT            5741
STOP.           5743
ABNORM.         5752
SYSAPG=         5773
IOERR.          6012                     OUTC=         7433
SYSEND.         6036
SYP=5           6037
CLSLNK.         6053
SYSERR.         6114                     COMIO=        4756     4771
                                         FMTAP=        5614
                                         GETFIT=       6562
                                         OUTC=         7440
                                         OVERLAY       7730
SYST1A.         6155
SYP=1           6161
SYP=2           6167
SYP=3           6236
SYP=4           6244
SYS=6           6274
SYS2=           6317
COD.            6356
CBD.            6365
COD.            6372
BFN.            6401
FECOPE.         6406                     OUTC=         7335
LINLIM.         6451                     OUTC=         7417
MSGAD.          6465
DBGFIT.         6476                     OVERLAY       7662
GETFIT.         6536            GETFIT=  OUTC=         7277
NAME.           6573
KOJPT.          6576            KODER=   OUTC=         7267
KODWRT=         7237                     OUTC=         7347
KOPEP.          7244                     OUTC=         7270
OUTCI.          7271            OUTC=    PRNT          4501
OUTCR.          7367
FEOL.           7446            OUTCOM=  KODER=        6613     6656
FEOT.           7451                     KODER=        6610     6611
FEOXFL.         7516                     FLTOUT=       5020     5210
                                         KODER=        7077     7146
FEOAFM.         7524                     FLTOUT=       5044     5047     5051     5055     5206
FEOBLS.         7531                     FLTOUT=       5000     5001     5002     5004     5174     5175     5201
                                                       5203
                                         KODER=        7030     7155
FEOCNV.         7544                     FLTOUT=       5016
FEOR1F.         7575                     FLTOUT=       5172
FEOR10.         7601
FEONTL.         7606
OVERLAY         7623            OVERLAY  TESTA         4254     4304
SQRT            7765            SQRT
SQRT.          10004                     TRY           4431     4433     4447     4451
SYSAID=        10027            SYSAID=  OB.IO.        4540
                                         SYS=1ST      10047
SYS1ST.        10033            SYS=1ST  SQRT         10017
MORGUE.        10035
```

Figure 1-13-6. Loader Map of Main Overlay (0, 0) (Cont'd)

```
SYS=            10114       SYS.RM      Q8.IO.      4536    4546
                                        FORSYS=     5701    5733    5772
                                        ERR.RM     11067   11106   11237
RCL=            10127
WNB=            10133
MSG=            10143       Q8.IO.      4544
                            FORSYS=     5726    5731    5761    5763    5770    6267    6306
                                        6314
                            ERR.RM     11024   11044   11115
                            PUT.SQ     13262   13316
                            LXER.SQ    16355
CIO=            10155
LOADER=         10162       UCLOAD
LOADER          10162       OVERLAY     7704
LBUF.SQ         10451       LBUF.SQ     LXER.SQ    16344
RM.CIO          10613       CIO.RM      ERR.RM     11103
                            OPEN.SQ    12275   12410
                            OPEX.SQ    12437
                            PUT.SQ     12600   13223   13310   13330   13541   13551   13634
                                       14020
                            WAR.SQ     14132   14306
                            CLSF.SQ    14404   14471
                            CLSV.SQ    14535   14545   14561   14571   14603
                            GET.SQ     15104   15271   15364   15521   15533
                            WEOX.SQ    16532   16552
RM.RCLA         10623       ERR.RM     11073
                            PUT.SQ     13522   13657   14007
                            WAR.SQ     14232
                            CLSF.SQ    14372
                            GET.SQ     14752   15026
                            RTRT.SQ    16210
RM.RCLP         10630       PUT.SQ     13236   13256
                            GET.SQ     15245
RM.SYS=         10637
ERR.RM          10734       ERR.RM      JMPS.RM    10441   10442   10443   10444   10445   10446   10450
                            MEMC.RM    11610   11611   11612
                            OPES.FO    11613
                            OPEN.FO    11614   11615   11616   11617   11620   11621   11622
                            OPEN.RM    11625   11716   11765   12031   12040   12042   12046
                                       12050   12051
                            OSUB.RM    12140   12141   12142   12143   12144   12145   12146
                                       12147   12150
                            OPEN.SQ    12226   12230   12233   12274   12334   12336   12373
                                       12423   12424
                            OPEX.SQ    12433
                            RLEQ.RM    12462
                            PUT.FO     12522   12523   12524   12525   12526   12527   12530
                            PUT.SQ     12531   12536   12572   12613   12616   12623   12672
                                       12734   12770   13017   13225   13254   13255   13275
                                       14015   14023
                            CLSF.FO    14310   14311   14312   14313   14314   14315   14316
                            CLSF.RM    14321   14332   14340
                            CLSF.SQ    14350   14427   14472
                            CLSV.FO    14473   14474   14475   14476   14477   14500   14501
                            GET.FO     14625   14626   14627   14630   14631   14632   14633
                            GET.SQ     14652   14656   14673   14700   14705   15405   15527
                                       15562   15605
                            LXER.SQ    16371
                            WEOX.SQ    16454   16464   16467   16500
                            SKFL.FO    16610   16611   16612   16613   16614   16615   16616
                            SKFL.SQ    16621   16624   16627
ERR1.RM         10740
ERR2.RM         11116
MOVE.RM         11262       MOVE.RM     PUT.SQ     13111
                            FSU.SQ     16061
CHWR.SQ         11346       CHWR.SQ     OPEN.SQ    12162
                            PUT.SQ     13026
MCT.RM          11362       MCT.RM      OPEN.RM    11646   11650   11752   11754   12022   12026
                            CLSF.RM    14324
OPEN.RM         11623       OPEN.RM     FORSYS=     6447
OSUB.RM         12057       OSUB.RM     OPEN.RM    11766
OPEN.SQ         12152       OPEN.SQ     FORSYS=     6447
OPXX.SQ         12267                   OPEX.SQ    12445
OPEX.SQ         12432       OPEX.SQ     OPEN.SQ    12266
RLEQ.RM         12457       RLEQ.RM     PUT.SQ     12644   12726
                            WAR.SQ     14031
PUT.SQ          12532       PUT.SQ      FORSYS=     5331
                            OUTC=       7423
FLSH.SQ         13643
WAR.SQ          14030       WAR.SQ      PUT.SQ     12622
                            WEOX.SQ    16477
REPO.SQ         14227
CLSF.RM         14317       CLSF.RM     FORSYS=     6101    6140
                            OUTC=       7327
CLSF.SQ         14342       CLSF.SQ
RSPT.SQ         14430                   OPEN.SQ    12345
                            OPEX.SQ    12434
                            CLSV.SQ    14521   14615
                            GET.SQ     15027   15513
```

Figure 1-13-6. Loader Map of Main Overlay (0, 0) (Cont'd)

```
-------- OVERLAY(OVLA,1,0)

         FWA OF THE LOAD          17033
         LWA+1 OF THE LOAD        17103

         TRANSFER ADDRESS -- OVL10          17036


         PROGRAM AND BLOCK ASSIGNMENTS.

         BLOCK       ADDRESS   LENGTH     FILE      DATE      PROCSSR VER LEVEL  HARDWARE    COMMENTS

         OVL10         17033       50     LGO       75/06/02 FTN      4.4 U401   666X I      OPT=1


         ENTRY POINTS.

         ENTRY         ADDRESS          PROGRAM   REFERENCES

         OUTPUT=          2204          TESTA     OVL10          17052     17056     17062     17066     17072
         QENTRY.          4541          FORSYS=   OVL10          17036
         END.             5715                    OVL10          17051
         OUTCT.           7271          OUTC=     OVL10          17040     17042     17044     17046     17050
         OVL10           17036          OVL10
```

Figure 1-13-7. Loader Map of Primary Overlay (1, 0)

```
   1.0000000       2.0000000       3.0000000       4.0000000       5.0000000       6.0000000       7.0000000
   8.0000000       9.0000000      10.0000000     100.0000000     200.0000000     300.0000000     400.0000000
 500.0000000     600.0000000     700.0000000     800.0000000     900.0000000    1000.0000000     101.0000000
   2.1000000       3.1000000     202.0000000       2.2000000       3.2000000     303.0000000       2.3000000
     3.3000000
6177.2000000       0.0000000       0.0000000       0.0000000       0.0000000       0.0000000
   2.0000000       4.0000000       6.0000000       8.0000000      10.0000000      12.0000000      14.0000000
  16.0000000      18.0000000      20.0000000       2.8284271       4.0000000       4.8989795       5.6568542
   6.3245553       6.9282032       7.4833148       8.0000000       8.4852814       8.9442719     101.0000000
   2.1000000       3.1000000     202.0000000       2.2000000       3.2000000     303.0000000       2.3000000
     3.3000000
 795.7498875       0.0000000       0.0000000       0.0000000       0.0000000       0.0000000
```

Figure 1-13-8. Program Output

```
EXCHANGE PACKAGE.
P    10114  A0    2204  B0       0      (A0)  1725 2420 2524 0000 0000
RA  124100  A1       1  B1       1      (A1)  0516 0420 0000 0000 0000
FL   17200  A2    6530  B2  777735      (A2)  1717 0631 4631 4640 3615
EM       7  A3    6531  B3    2450      (A3)  2000 0000 0000 0000 0012
RAX      0  A4    5670  B4      24      (A4)  5555 5555 5555 5733 3640
FLX      0  A5    5674  B5   10444      (A5)  2000 0000 0000 0003 1571
MA    2200  A6       1  B6    6102      (A5)  0516 0420 0000 0000 0000
            A7    2236  B7      30      (A7)  0000 0000 0000 0000 0000

X0  0000 0000 0000 0000 0000
X1  0000 0000 0000 0000 0000
X2  1717 0631 4631 4640 3615
X3  2000 0000 0000 0000 0012
X4  2000 0000 0000 0000 0000
X5  0000 0000 0000 0000 0003
X6  0516 0420 0000 0000 0000
X7  2000 0000 0000 0000 0001

(RA)    0000 0000 0000 0000 0000
(RA+1)  0516 0420 0000 0000 0000


DUMP FROM  10054 TO 10154
 10054    52577 77676 66622 53455    06600 10056 54355 55431    03410 10057 56330 56431    03610 10060 56330 57431
 10060    10633 55341 22704 55431    76165 51600 10103 55761    10633 74260 22704 55671    21111 55761 55071 46000
 10064    01000 00000 61000 46000    51300 10076 61100 00001    43002 55231 26050 63735    21322 55121 63635 55411
 10070    21322 53040 04000 10030    00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000
 10074    00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000    11160 61116 11240 50000
 10100    55012 20725 15051 62455    11160 40506 11161 12405    00000 00000 00000 00000    00000 00000 00000 00000
 10104    00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 10074
 10110    00000 00000 00000 00000    00000 00000 00000 00000    04000 10125 00000 00000    01300 00000 00000 00000
 10114    04000 05734 00000 00000    51100 00001 03110 10115    54610 04000 10113 46000    51100 00066 03310 10121
 10120    51100 10112 04000 10122    71100 00130 20160 46000    13661 13161 13661 46000    51600 10113 10611 46000
 10124    51100 00001 01000 10112    20652 01000 10114 46000    51100 00001 03110 10126    04004 10127 61000 46000
 10130    51100 00001 03110 10127    71602 20314 04000 10125    20150 36661 01000 10114    04004 10133 61000 46000
 10134    71602 20314 20652 36662    53160 20173 03310 10133    03010 10133 51100 00001    03110 10135 71100 00001
 10140    04000 10132 61000 46000    71603 24616 12661 20651    01000 10114 61000 46000    04000 05732 00000 00000
 10144    20622 12161 20651 20123    03260 10141 71600 00301    20645 13116 04000 10141    01000 10133 61000 46000
 10150    43652 71100 00002 12661    53120 11161 71600 31117    03270 10153 14777 27606    12717 20652 53720 12662
 10154    73220 01000 10114 46000    04004 10155 61000 46000    53120 20173 03310 10150    03110 10147 52120 00001
```

Figure 1-13-9. Exchange Package Dump

```
DUMP FROM      0 TO 10000
     0      00000 00000 00000 00000    05160 42000 00000 00000    11162 02524 00000 00143    17252 42025 24000 02204
     4      00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000
        DUPLICATED LINES.
    54      56110 03110 00054 54710    51100 00001 03110 00055    64550 02550 00000 46000    00000 00000 00000 00000
    60      15051 52000 00000 00061    00000 17200 00000 00001    00000 00000 00000 00000    00000 00000 00000 00000
    64      17261 40100 00000 00000    40000 00000 00000 17103    40242 00000 01000 17032    40000 00000 40000 00000
    70      17261 40157 55000 00000    00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000
    74      00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000
   100      00000 00001 71030 04252/AAA(A)17214 00000 00000 00000    17224 00000 00000 00000    17226 00000 00000 00000
   104      17234 00000 00000 00000    17235 00000 00000 00000    17236 00000 00000 00000    17237 00000 00000 00000
   110      17244 00000 00000 00000    17244 40000 00000 00000    17245 00000 00000 00000 (B) 17215 52023 63147 74736
   114      17224 00000 00000 00000    17224 71421 60500 44411    17225 52023 53147 74736    17226 24613 01655 51333
   120      17226 73317 27205 41143    17227 36735 20426 40772    17234 00000 00000 00000    17234 17416 66315 75547
   124      17234 36156 74671 37646 (C) 17226 00000 00000 00000    17214 14631 46314 63146    17216 14631 46314 63146
   130      17275 24000 00000 00000    17214 31463 14631 46315    17215 31463 14631 46315    17304 57000 00000 00000
   134      17214 46314 63146 31463    17216 46314 63146 31463    17226 00000 00000 00000    17236 00000 00000 00000
   140      17244 40000 00000 00000    17246 00000 00000 00000    17247 40000 00000 00000    11162 02524 00000 00000
   144      00000 00000 00000 00165    00000 00042 60000 00000    00000 00000 20010 00000    00000 00000 00000 00000
   150      00000 00002 00000 00203    00000 00000 01400 00000    00000 00000 00000 00000    00000 00000 00000 00000
   154      00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000
        DUPLICATED LINES.
  2204      17252 42025 24000 00000    00000 14700 40630 02227    00000 00042 60010 00000    00000 00000 20010 00000
  2210      00000 00000 00000 06103    00000 00003 00000 02244    00000 22600 03400 04521    00000 00000 00000 00000
  2214      00000 00000 00000 00000    00000 00000 00000 00000    00000 00002 00000 00000    00000 00002 00000 00000
  2220      00000 10000 00000 00000    00300 00000 00000 02244    00000 00000 00000 00000    00000 00000 00000 02244
  2224      00000 00000 00002 24400    00000 00000 00000 00000    00000 00000 00000 00000    17252 42025 24000 00131
  2230      04021 46000 00040 02244    00000 00000 00000 02244    00000 00000 00000 02244    37520 00000 01000 04245
  2234      00000 00000 00000 00000    00000 02237 00000 00000    00000 00000 00000 00000    00000 00000 00000 00001
  2240      00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000
  2244      55555 55555 55555 55534    57333 33333 33333 35555    55555 55555 55355 73333    33333 33333 55555 55555
  2250      55555 53657 33333 33333    33335 55555 55555 55555    37573 33333 33333 33355    55555 55555 55554 05733
  2254      33333 33333 33555 55555    55555 55541 57333 33333    33333 35555 55555 55555    55425 73333 33333 33333
  2260      00000 00000 00000 00000    55555 55555 55555 55543    57333 33333 33333 35555    55555 55555 55445 73333
  2264      33333 33333 55555 55555    55553 43357 33333 33333    33335 55555 55555 53433    33573 33333 33333 33355
  2270      55555 55555 35333 35733    33333 33333 33555 55555    55553 63333 57333 33333    33333 35555 55555 55537
  2274      33335 73333 33333 33333    00000 00000 00000 00000    55555 55555 55554 03333    57333 33333 33333 35555
  2300      55555 55541 33335 73333    33333 33333 55555 55555    55423 33357 33333 33333    33335 55555 55555 54333
  2304      33573 33333 33333 33355    55555 55555 44333 35733    33333 33333 33555 55555    55343 33333 57333 33333
  2310      33333 35555 55555 55534    33345 73333 33333 33333    00000 00000 00000 00000    55555 55555 55555 55535
  2314      57343 33333 33333 35555    55555 55555 55365 73433    33333 33333 55555 55555    55353 33557 33333 33333
  2320      33335 55555 55555 55555    35573 53333 33333 33355    55555 55556 55553 65735    33333 33333 33555 55555
  2324      55553 63336 57333 33333    33333 35555 55555 55555    55355 73633 33333 33333    00000 00000 00000 00000
  2330      55555 55555 55555 55555    55553 65736 33333 33333    33000 00000 00000 00000    55555 55555 55413 44242
  2334      57353 33333 33333 35555    55555 55555 55335 73333    33333 33333 55555 55555    55555 53357 33333 33333
  2340      33335 55555 55555 55555    33573 33333 33333 33355    55555 55555 55553 35733    33333 33333 33555 55555
  2344      55555 55533 57333 33333    33333 30000 00000 00000    55555 55555 55555 55535    57333 33333 33333 35555
  2350      55555 55555 55375 73333    33333 33333 55555 55555    55555 54157 33333 33333    33335 55555 55555 55555
  2354      43573 33333 33333 33355    55555 55555 55343 35733    33333 33333 33555 55555    55555 53435 57333 33333
```

Figure 1-13-10. Central Memory Dump

```
2360      33333 35555 55555 55555    34375 73333 33333 33333    00000 00000 00000 00000    55555 55555 55555 53441
2364      57333 33333 33333 35555    55555 55555 34435 73333    33333 33333 55555 55555    55553 53357 33333 33333
2370      33335 55555 55555 55555    35574 33543 37354 23455    55555 55555 55553 75733    33333 33333 33355 55555
2374      55555 55537 57434 44344    42444 05555 55555 55555    55405 74140 41434 03735    00000 00000 00000 00000
2400      55555 55555 55555 55541    57363 53740 40403 55555    55555 55555 55415 74435    43353 33635 55555 55555
2404      55555 54257 37433 63634    37435 55555 55555 55555    43573 33333 33333 33355    55555 55555 55554 35737
2410      43403 54334 37555 55555    55555 55543 57443 73735    42344 45555 55555 55534    33345 73333 33333 33333
2414      00000 00000 00000 00000    55555 55555 55535 55535    57343 33333 33333 35555    55555 55555 55365 73433
2420      33333 33333 55555 55555    55353 33557 33333 33333    33335 55555 55555 55555    35573 53333 33333 33355
2424      55555 55555 55553 65735    33333 33333 33555 55555    55553 63336 57333 33333    33333 35555 55555 55555
2430      55355 73633 33333 33333    00000 00000 00000 00000    55555 55555 55555 55555    55553 65736 33333 33333
2434      33000 00000 00000 00000    55555 55555 55554 24440    57423 74443 43424 05555    55555 55555 55335 73333
2440      33333 33333 55555 55555    55555 53357 33333 33333    33335 55555 55555 55555    33573 33333 33333 33355
2444      55555 55555 55553 35733    33333 33333 33555 55555    55555 55533 57333 33333    33333 30000 00000 00000
2450      00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000
          DUPLICATED LINES.
4244      00000 00000 00000 00000    00000 22600 00000 04521    00000 22600 00000 04521    77777 77777 77777 66167
4250      24052 32401 55550 04252    00000 00000 00000 00000    51100 04245 01000 04541    51100 04312 46000 46000
4254      01000 07623 00160 04250    51100 04316 46000 46000    01000 04467 00170 04250    71700 00001 51700 04332
4260      51500 04332 52450 00100    72750 00100 40644 46000    51700 04320 51700 04321    54640 51100 04320 46000
4264      01000 04426 00220 04250    51500 04332 72750 00100    52650 00100 51700 04320    51700 04321 51100 04320
4270      01000 04436 00230 04250    51500 04332 72750 00001    72077 77764 52650 00112    54750 03300 04260 46000
4274      71700 00001 51700 04333    51500 04333 61600 00005    36055 63750 62500 00100    51570 00100 56450 30045
4300      51550 00002 24700 46000    51770 00135 61770 00001    06670 04277 76770 46000    51700 04333 51100 04323
4304      01000 07623 00260 04250    51100 04316 46000 46000    01000 04467 00270 04250    51500 04333 51100 04250
4310      27005 24700 51700 00125    04000 05715 46000 46000    00000 00000 00000 04416    00000 00000 00000 04327
4314      00000 00000 00000 04330    00000 00000 00000 00000    00000 00000 00000 04331    00000 00000 00000 00000
4320      00000 00000 00000 00112    00000 00000 00000 00112    00000 00000 00000 00000    00000 00000 00000 04420
4324      00000 00000 00000 04327    00000 00000 00000 04330    00000 00000 00000 00000    00000 00000 00000 00001
4330      00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00013 I  00000 00000 00000 00000
4334      00000 00000 00000 00173    00000 00000 00000 00173    00000 00000 00000 00173    00000 00000 00000 00173
          DUPLICATED LINES.
4414      00000 00000 00000 00173    00000 00000 00000 00173    17261 40155 55555 55555    00000 00000 00000 00000
4420      17261 40155 55555 55555    00000 00000 00000 00000    24223 15555 55550 04426    00000 00000 00000 17200
4424      51400 04456 10644 46000    51300 04423 52030 00000    04000 04271 00000 00000    74600 54010 51600 04423
4430      46000 46000 46000 46000    54500 53150 01000 10004    50500 03001 53150 46000    51600 04454 01000 10004
4434      51500 04454 30056 24700    51700 04456 04000 04424    04000 04271 00000 00000    51200 04441 10622 46000
4440      51600 04430 04000 04427    04000 04442 61000 46000    51100 04455 51200 04436    10611 22702 51600 04430
4444      51700 04426 61000 46000    54500 50400 00001 53150    53340 31031 24700 46000    03270 04431 01000 10004
4450      50500 00001 53150 46000    51600 04454 01000 10004    51500 04454 31056 24700    51700 04456 04000 04424
4454      17224 36156 74671 37646    46000 46000 46000 46000    17234 36156 74671 37646    00000 00000 00000 00000
4460      00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000
4464      20221 62455 55550 04467    00000 00000 00000 17200    51300 04465 52030 00000    04000 04307 00000 00000
4470      74600 54010 51600 04465    43700 71600 00001 46000    51700 04512 51600 04513    51500 04513 61600 00035
4474      63750 46000 46000 46000    51500 04512 51470 00100    61770 00001 30045 24700    54750 06670 04475 76770
4500      51700 04513 51100 04503    01000 07271 00100 04464    04000 04466 46000 46000    00000 00000 00000 02204
4504      00000 00000 00000 04510    00030 00000 01000 04912    00030 00000 05000 04457    00000 00000 00000 00000
4510      55404 05555 55000 00000    51343 04105 34425 74252    17716 15677 76120 00744    00000 00000 00000 00036
4514      00000 00000 00000 04245    00000 00000 00000 00000    03172 03122 11071 02455    00000 00000 00000 00000
4520      00000 00000 00000 00000    55555 55555 55554 24440    57423 74443 43424 05555    55555 55555 55335 73333
4524      33333 33333 55555 55555    55555 53357 33333 33333    33335 55555 55555 55555    33573 33333 33333 33355
4530      55555 55555 55553 35733    33333 33333 33555 55555    55555 55533 57333 33333    33333 33333 33333 33333
4534      55355 73633 33333 33333    20552 12661 43101 20151    12661 01000 10114 46000    51200 04531 10722 46000
4540      00000 00000 00000 04505    00000 00000 74410 04315    00000 00000 00000 02204    00000 00000 00000 04504
4544      00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00006    77777 77777 77777 77776
4550      00000 00000 00000 00000    00000 00000 00000 00007    00000 00000 00000 00000    00000 00000 00000 00001
4554      00000 00000 00000 00004    00000 00000 00000 00000    00000 00000 00000 00226    20060 00000 00000 00021
4560      11701 54335 06020 04576    51200 04626 11403 37623    00000 00000 00000 00002    00000 00000 00000 00000
4564      20120 00001 01000 04512    00000 00000 00000 00000    00000 00000 00000 04521    04000 07244 61000 46000
4570      00000 00000 00000 00057    51000 04627 56710 54300    01000 04640 61000 46000    51500 04627 61500 00001
4574      20120 00000 00000 04511    43052 04000 04623 46000    57225 03310 04623 53210    03120 04600 53710 46000
4600      10255 52510 00005 46000    71600 00001 43773 20535    15667 11775 36667 20637    54650 52510 00006 46000
4604      71600 00006 43771 20546    15567 11775 36667 20626    54650 52510 00002 46000    71600 00002 43772 20532
4610      15667 11775 36667 20642    54650 52510 00002 46000    71600 00002 43772 20536    15667 11775 36667 20636
4614      54650 52510 00006 43752    15457 52510 00005 43744    21544 15657 61500 00001    03140 04621 71600 00226
4620      71400 04521 61000 46000    20544 12764 54710 10611    22502 54640 54115 54445    03210 04560 07020 04560
4624      03350 04532 14155 46000    04000 04532 61000 46000    20140 00000 00000 00002    00000 00000 00000 00000
4630      22227 36727 20503 15213    20701 12661 36771 53525    20506 15150 63410 22244    73113 11505 03320 04630
4634      56357 04400 04637 66322    03150 04640 04430 04637    67545 05520 04640 56300    43400 04300 04640 10677
4640      00000 00000 61000 46000    71400 07774 43056 76600    66211 13777 61307 77744    20425 66500 71300 00007
4644      04000 04632 61000 46000    00000 00000 00000 00000    00000 00000 00000 00000    00000 00000 00000 00000
4650      55555 55555 55555 55555    77777 77777 77777 77777    77777 77777 77777 77700    77777 77777 77777 70000
4654      77777 77777 77770 00000    77777 77777 77000 00000    77777 77777 00000 00000    77777 77700 00000 00000
4660      77777 70000 00000 00000    77770 00000 00000 00000    77000 00000 00000 00000    00000 00000 00000 00000
4664      17204 00000 00000 00000    17235 00000 00000 00000    17265 20000 00000 00000    17317 64000 00000 00000
4670      17354 70400 00000 00000    17406 06500 00000 00000    17437 50220 00000 00000    17474 61132 00000 00000
4674      17525 75360 40000 00000    17557 34654 50000 00000    17614 52013 71000 00000    17645 54416 67200 00000
4700      17677 21522 45040 00000    17734 43023 47124 00000    17765 53630 40751 00000    20027 06576 51143 20000
4704      20064 34157 11576 02000    20115 43212 74135 42400    20146 74050 53164 73100    20204 25434 43011 04750
```

Figure 1-13-10. Central Memory Dump (Cont'd)

PROGRAM LIBRARY AND SYSTEM UTILITY CONTROL CARDS 14

PROGRAM LIBRARY UTILITY CONTROL CARDS

The following control cards are provided by KRONOS to enable the user to maintain a program library.

MODIFY	Edits a Modify-formatted program library file
OPLEDIT	Removes modification decks and identifiers from a Modify-formatted program library file
UPDATE	Edits an Update-formatted program library file
UPMOD	Converts an Update-formatted program library file to a Modify-formatted program library file

MODIFY CARD

The MODIFY control card edits a Modify-formatted program library file.

The control card format is:

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

p_i Any of the following in any order:

I	Use directive input from file INPUT. If the I option is omitted, file INPUT is assumed.
$I=lfn_1$	Use directive input from file lfn_1.
I=0	Use no directive input.
P	Use file OPL for the old program library. If the P option is omitted, file OPL is assumed.
$P=lfn_2$	Use file lfn_2 for the old program library.
P=0	Use no old program library
C	Write compile output to file COMPILE. If the C option is omitted, file COMPILE is assumed.
$C=lfn_3$	Write compile output to file lfn_3.
C=0	Write no compile output.
N	Write new program library on file NPL.
$N=lfn_4$	Write new program on file lfn_4.
N=0	Write no new program library. If this option is omitted, N=0 is assumed.
S	Write source output on file SOURCE.
$S=lfn_5$	Write source output on file lfn_5.
S=0	Write no source output. If this option is omitted, S=0 is assumed.

L	List output on file OUTPUT. If the L option is omitted, file OUTPUT is assumed.
L=lfn_6	List output on file lfn_6.
L=0	List no output.
LO	Select list options: ECTMWDS
LO=chars	Selects up to seven list options which can be any of the following: E Errors C Directives other than INSERT, DELETE, RESTORE T Input text M Modifications made W Compile file directives D Deck status S Statistics I Inactive cards A Active cards
A	Write compressed compile file.
D	Ignore errors.
F	Modify all decks.
U	Modify only decks mentioned on DECK directives; F overrides the U option.
NR	Do not rewind the compile file.
X	Rewind input and output files, set A option, and call the COMPASS assembler when modification is complete.
X=prog	Rewind input and output files, set A option, and call the processing program prog when modification is complete.
X=0	Do not call another processing program. If this option is omitted, X=0 is assumed.
Q	Rewind the output file, set A option, and call the COMPASS assembler when modification is complete.
Q=prog	Rewind the output file, set A option, and call the prog assembler when modification is complete.
Q=0	Do not call another processing program. If this option is omitted, Q=0 is assumed.
Z	If this parameter is present, the MODIFY control card contains the input directives following the terminator. When this parameter is specified, the I parameter is ignored. NOTE Do not place another terminator after the directives.

CV=63	Convert 64 character set OPL to 63 character set OPL.
CV=64	Convert 63 character set OPL to 64 character set OPL.

The following parameters can be entered only if the X or Q options are selected.

CB	Set assembler argument B=LGO. If the CB option is omitted, B=LGO is assumed.
CB=lfn_7	Set assembler argument B=lfn_7.
CB=0	Set assembler argument B=0.
CL	Set assembler argument L=OUTPUT.
CL=lfn_8	Set assembler argument L=lfn_8.
CL=0	Set assembler argument L=0. If this option is omitted, L=0 is assumed.
CS	Set assembler argument S=SYSTEXT. If the CS option is omitted, S=SYSTEXT is assumed.
CS=lfn_9	Set assembler argument S=lfn_9. †
CS=0	Set assembler argument S=0.
CG	Set assembler argument G=SYSTEXT.
CG=lfn_{10}	Set assembler argument G=lfn_{10}. ††
CG=0	Set assembler argument G=0. If this option is omitted, CG is defined by the CS option.

For a more detailed description of Modify, refer to the Modify Reference Manual.

OPLEDIT CARD

The OPLEDIT control card removes modification decks and identifiers from a Modify-formatted program library file.

The control card format is:

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

p_i Any of the following in any order:

I	Use directive input from file INPUT. If the I option is omitted, file INPUT is assumed.
I=lfn_1	Use directive input from file lfn_1.
I=0	Use no directive input.
P	Use file OPL for the old program library. If the P option is omitted, file OPL is assumed.
P=lfn_2	Use file lfn_2 for the old program library.
P=0	Use no old program library.
N	Write new program library on file NPL.

† The desired file is retrieved from the system.
†† The desired file is a local file.

$N=lfn_3$	Write new program library on file lfn_3.
N=0	Write no new program library. If this option is omitted, N=0 is assumed.
L	List output on file OUTPUT. If the L option is omitted, file OUTPUT is assumed.
$L=lfn_4$	List output on file lfn_4.
L=0	List no output.
$M=lfn_5$	Write output from *PULLMOD directives on file lfn_5. If this option is omitted, M=MODSETS is assumed.
LO=x	Set list options x; each bit in x, if set, turns on the corresponding option.

001	Errors
002	Directives
004	All other input cards
010	Modifications made
020	Directives processed from the program library
040	Deck status
100	Directory lists
200	Inactive cards
400	Active cards

If this option is omitted, x=177 is assumed (that is, the first seven options listed).

F	Modify all decks.
D	Debug; ignore errors.
U	Generate *EDIT directives for all decks.
U=0	Generate no *EDIT directives. If the U option is omitted, generate *EDIT directives for common decks.

The OPLEDIT routine is used to completely remove specified modification decks and modification identifiers from an OPL. It can also be used to determine the contents of a specified modification set on an OPL file. The OPLEDIT parameters are similar to the MODIFY parameters. However, the only input directives are as follows:

*PURGE name

PURGE name,

name	Modification identifier to be purged
*	Indicates that name and all identifiers thereafter are to be purged

*EDIT dck1

*EDIT dck1.dckn

dck1	Deck to be edited
dck1.dckn	Edit all decks from dck1 to dckn

The EDIT directive requests OPLEDIT to edit a program deck and transfer it to the new program library. The deck names specified normally are the decks that contain the modification identifiers.

The PULLMOD directive is available to reconstruct a modset to a deck and write it on a specified file.

```
*PULLMOD d moda,modb,...,modn
```

d	Valid OPLEDIT delimiter
moda, modb, ...,modn	Modnames to be pulled and generated onto the file specified by the M option (MODSETS is assumed by default)

The PULLALL directives also make it possible to generate a modset that contains the net effect of all current modsets or all modsets added after and including a specific modset.

```
*PULLALL
*PULLALL,modn
```

modn	First modset to be included; all modsets following modn are also included.

Example 1

The user requires a composite modset of all mods that have been added to a deck. The following cards perform this function.

```
job card
USER(...)
    :
    :
OPLEDIT(M=MOD,...)
-EOR-
*PULLALL
*EDIT 1TD
-EOI-
```

File MOD would contain:

```
*******
*IDENT*******
*DECK 1TD
    :
    :
-EOI-
```

File MOD plus the 1TD OPL, which contains no mods, builds the latest 1TD OPL.

Example 2

The user requires 3 modsets to be combined into one composite modset. The following cards perform this function.

```
job card
USER(...)
OPLEDIT(M=MOD,...)
-EOR-
*PULLALL,1TD27
*EDIT,1TD
-EOI-
```

File MOD would contain:

```
*******
*IDENT*******
*DECK 1TD
    :
-EOI-
```

File OPL contains 1TD,1TD1,1TD2,...,1TD27,1TD28. The resulting MOD file contains 1TD27 and 1TD28.

Example 3

The user requires a noncomposite mod for file MOD. The following cards perform this function.

```
job card
USER(...)
    :
OPLEDIT(M=MOD,...)
-EOR-
*PULLMOD,1TD27,1TD28
*EDIT,1TD
-EOI-
```

File MOD would contain:

```
1TD27
*IDENT 1TD27
*DECK 1TD
    :
1TD28
*IDENT 1TD28
*DECK 1TD
    :
-EOI-
```

It is not possible to pull modsets implicitly; that is, *PULLMOD A.B is illegal. Also, *PULLMOD A,* does not pull modset A and all modsets that follow (as on the *PURGE directive). Rather, it pulls modset A and modset *.

Modnames requested are pulled only from edited decks. The *PULLMOD and *PURGE directives may be used in any order and combination.

The modsets generated are in a form suitable for use by MODIFY as follows:

```
*READ,file,*
     or
*READ,file,ident
```

That is, each modset is a separate record, with ident being the first card. The *PULLALL modset, if any, is the first record on the file. The file specified by the M option is returned before, and rewound after use.

UPDATE CARD

The UPDATE control card edits an Update-formatted program library file.

The control card format is:

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

p_i Any of the following in any order:

A	Sequential-to-random program library copy
B	Random-to-sequential program library copy
C	Write compile file output on COMPILE. If the C option is omitted, file COMPILE is assumed.
C=lfn_1	Write compile file output on lfn_1.
C=0	Write no compile output.
D	Compile output has 80 columns for data; if D is omitted, compile output has 72 columns for data.
E	Update rearranges the directory to reflect the actual order of decks on the program library. If E is omitted, the old program library directory is not edited.
F	Full update; all decks are compiled. If F is omitted, corrected decks and those named on COMPILE directives are processed.
G=lfn_2	Output from PULLMOD directives is written on lfn_2. Any rewind option applying to the source file also applies to this file. OUTPUT is not a valid file for this option. If G is omitted, pulled modifications are appended to the source file.
I	Input is on file INPUT. If the I option is omitted, file INPUT is assumed.
I=lfn_3	Input comprises next record on lfn_3.
K	Compile output decks to be written on file COMPILE in COMPILE directive sequence.
K=lfn_4	Compile output decks to be written on lfn_4 in COMPILE directive sequence. If this option is omitted, output is determined by the C option.
L=char	char is a string that specifies any of the A, F, and 0 through 9 list options. If this option is omitted, options A, 1, 2, 3, and 4 are selected. Any use of 0 suppresses listing.
M	Merge input is on file MERGE.
M=lfn_5	Merge input is on file lfn_5. If M option is omitted, there is no merge file.
N	New program library to be written on file NEWPL.

$N=lfn_6$	New program library to be written on file lfn_6. If N option is omitted, no new program library is written.
O	List output to be written on OUTPUT. If the O option is omitted, OUTPUT is assumed.
$O=lfn_7$	List output to be written on lfn_7. If O option is omitted, OUTPUT is assumed.
P	Use file OLDPL for the old program library. If the P option is omitted, OLDPL is assumed.
$P=lfn_8$	Use file lfn_8 for the old program library. If this option is omitted, OLDPL is assumed.
Q	Only decks on COMPILE directives are processed. If Q is omitted, corrected decks and those named on COMPILE directives are processed.
R	No rewinds are issued for the program libraries, compile file, or source file.
R=char	Each character in the string char indicates a file to be rewound before and after the Update run. C Compile N New program library P Old program library and merge library S Source and PULLMOD Files not specified in char are not rewound. If R is omitted, files are rewound before and after the Update run.
S	Source output written on file SOURCE.
$S=lfn_9$	Source output written on file lfn_9. If S is omitted, Update does not generate a source output file unless the source output is specified by T.
T	Source output excluding common decks on file SOURCE.
$T=lfn_{10}$	Source output excluding common decks on file lfn_{10}. If T is omitted, no source output unless source output is specified by S.
U	Update execution is not terminated by normally fatal errors. If U is omitted, Update execution terminates upon encountering a fatal error.
W	The new program library (refer to N option) will be a sequential file. If W is omitted, the new program library will be a random file (unless it is a magnetic tape file).

X	Compile file is in compressed format. If X is omitted, the compile file is not in compressed format.
Z	The input file (refer to I option) is assumed to be in PCS compressed format. This parameter applies to the directives input file only; it does not apply to files specified by READ directives. If Z is omitted, the input file is a normal, coded file.
8	Compile file output is composed of 80-column line images. If this option is omitted, compile file output is composed of 90-column line images.
*=char	The master control character (first character of each directive) for this Update run is char which can be any character having a display code octal value in the range 01 through 54 except for 51 and 52 (the open and close parentheses). If this option is omitted, the master control character is *.
/=char	The comment control character for this Update run is char which can be A through Z, 0 through 9, or +-*/$=. Note, however, that the character should not be changed to one of the abbreviated forms of directives unless NOABBREV is in effect. If this option is omitted, the comment control character is a slant bar.

Note that the UPDATE control card is processed in product set format. For a more detailed description of Update, refer to the Update Reference Manual.

UPMOD CARD

The UPMOD control card converts an Update-formatted program library file to a Modify-formatted program library file.

The control card format is:

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

p_i Any of the following in any order:

P	Update program library from file OLDPL. If the P option is omitted, file OLDPL is assumed.
P=lfn_1	Update program library from file lfn_1.
N	Modify program library on file OPL.
N=lfn_2	Modify program library on file lfn_2.
M	Modify program library name is OPL. If the M option is omitted, file OPL is assumed.
M=lfn_3	Modify program library name is lfn_3.
F	Convert to file mark
NR	Do not rewind file lfn_1.

The Update file must be in sequential format. A random Update file must first be changed to sequential format via Update before being submitted to UPMOD for conversion. Unless otherwise specified, only one record from the Update file is converted. After the Modify OPL has been created, no references should be made to modset identifiers present on the Update library. The new OPL should be treated as any other program library created by a Modify creation run.

SYSTEM UTILITY CONTROL CARDS

The following utility control cards aid the system programmer in manipulating the system.

FAMILY	Specifies the name of the system's normal family of permanent file devices. FAMILY can be used by any system origin job.
KRONREF	Generates a cross-reference listing of system symbols.
SYSEDIT	Modifies the system library. SYSEDIT can be used by system origin jobs or users with system origin privileges (if the system is in DEBUG mode).

FAMILY CARD

The FAMILY control card allows the user to change the family name associated with his job.

The control card format is:

FAMILY(familyname)

familyname	1- to 7-character name of a family of permanent file devices

If an alternate family of permanent file devices is introduced into the system without a VALIDUS file, the job to create VALIDUS could include a FAMILY card identifying that family. If the familyname parameter is omitted, the default family name is used. If the normal system has a default family name but the alternate system does not, the user can specify the alternate system by submitting the following card.

FAMILY,0.

The FAMILY card can be used by any system origin job. If the card is included in any nonsystem origin job, the job is aborted and the system issues the following message to the user's dayfile.

ILLEGAL CONTROL CARD.

If the default family is required but not known to the user, a

FAMILY.

card will set the default family name.

Refer to Device Residence, section 2, for further information about families.

KRONREF CARD

The KRONREF control card generates a cross-reference listing of system symbols used by decks on a MODIFY OPL.

The control card format is:

KRONREF(P=lfn_1, L=lfn_2, S=lfn_3, G=lfn_4)

P=lfn_1	OPL input from file lfn_1. If the P option is omitted or P alone is specified, file OPL is assumed.
L=lfn_2	List output on file lfn_2. If the L option is omitted or L alone is specified, file OUTPUT is assumed.
S=lfn_3	System text from overlay lfn_3. If the S option is omitted or S alone is specified, file SYSTEXT is assumed.
G=lfn_4	System text from local file lfn_4. If G is omitted, system text is acquired as specified or defaulted by the S option. If G alone is specified, local file TEXT is used. Use of the G option overrides any S specification.

The names of programs on the OPL are listed for those decks that reference the following.

- PP direct cell locations defined in lfn_3 or lfn_4
- PP resident entry points defined in lfn_3
- Monitor functions
- Central memory pointers (in low core) defined in lfn_3 of lfn_4
- Central memory locations (in low core) defined in lfn_3 or lfn_4
- Control point area words defined in lfn_3 or lfn_4
- Dayfile message options
- File types and mass storage constants
- Job origin types, queue types, and priorities
- Error flags referenced
- Common deck calls
- PP packages called

SYSEDIT CARD

The SYSEDIT control card enables modifications to be made to the system library.

The control card format is:

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

p_i Any of the following in any order:

I	Directive input is on file INPUT. If the I option is omitted, file INPUT is assumed.
I=lfn_1	Directive input is on file lfn_1.
I=0	No directive input.
B	Binary change cards are on file LGO. If the B option is omitted, file LGO is assumed.
B=lfn_2	Binary change cards are on file lfn_2.
B=0	No binary change cards.
L	List output on file OUTPUT.
L=lfn_3	List output on file lfn_3.
L=0	No list output. If the L option is omitted, the system assumes L=0.
R	Restore to initial deadstart system.
R=n	Restore to copy n of the system. The system copy number is printed on the output listing.
R=0	No system file restoration. If the R option is omitted, the system assumes R=0.
C	Checkpoint the system following SYSEDIT. If the C option is omitted, no checkpoint is performed unless the system was generated employing the alternate system library residency feature. System using the alternate library feature are check-pointed automatically following SYSEDIT.

If the job is not of system origin, DEBUG must be set from the system console to allow changes to the operating system. The user must also be validated for system origin privileges.

The following are the valid input directives to SYSEDIT. †

*AD, nn, ty_1/rec_1, ty_2/rec_2, ..., ty_n/rec_n	Specifies the alternate device to be used instead of the system device(s) for storing ABS, OVL, and PP type routines; nn is either the EST ordinal or the device type.
*CM, ty_1/rec_1, ty_2/rec_2, ..., ty_n/rec_n	Define record rec_i of type ty_i as being central memory resident.
*MS, ty_1/rec_1, ty_2/rec_2, ..., ty_n/rec_n	Define record rec_i of type ty_i as being mass storage resident.
*DELETE, ty_1/rec_1, ty_2/rec_2, ..., ty_n/rec_n or *D, ty_1/rec_1, ty_2/rec_2, ..., ty_n/rec_n	Delete record rec_i of type ty_i from the system library. Type ty_i=ULIB is ignored; user libraries cannot be deleted.
*FILE,lfn or *FILE,lfn,NR	Define file lfn as a file containing system changes. If NR is not present, lfn is rewound before processing.
*IGNORE, ty_1/rec_1, ty_2/rec_2, ..., ty_n/rec_n	Do not process record rec_i of type ty_i when it appears on the system change file.
*PROC, rec_1, rec_2, ..., rec_n	Define record rec_i of type TEXT as procedure file.
*RENAME, oe_1-ne_1, oe_2-ne_2, ..., oe_n-ne_n	Rename CPU entry names oe_i to ne_i.
*PPSYN, nam/nam_1, nam_2, ..., nam_n	Add entries to system library to provide synonym nam_i for the PPU program nam.
*SC, ty_1/rec_1, ty_2/rec_2, ..., ty_n/rec_n	Define record rec_i of type ty_i as product set format control cards. The control card parameters will be processed in product set format (refer to Control Card Format, section 5).

† Refer to the CATALOG control card for record types.

LOADERS 15

System loading is done by the following loaders.

- LINK relocatable loader
- CDC CYBER reloctable loader
- Absolute overlay loader (LDR)
- Chippewa-formatted overlay loader (EXU)

A relocatable loader provides high-speed transfer of relocatable binary programs from storage devices to central memory. It is called when a job requires loading of a file or program in relocatable format. The LINK loader is provided for compatibility with previous operating systems. It provides many of the features available with the CDC CYBER relocatable loader but differs in the following respects.

- The LINK loader does not allow multiple sequential loads such as those available with the CDC CYBER loader through the LOAD card.
- It allows only the Cnnnnnn origin optional OVERLAY directive parameter to specify the origin of the overlay (refer to the Loader Reference Manual).
- It does not recognize the global library set specified by the LIBRARY control card.
- It uses the first library of the global library set as an alternate default library. This library is searched after the libraries specified on the LINK card and the libraries specified (if any) in LDSET tables, but before the default library SYSLIB.

Additional information and a description of the LINK control card is discussed later in this section. For a complete description of the CDC CYBER loader, its control cards and processing, refer to the Loader Reference Manual.

The overlay loaders provide direct transfer of absolute overlays or binary data from storage devices to central memory. An absolute record consists of code or data that is not relocatable and must be loaded at specific core locations. Because the overlay loaders perform no address manipulation, absolute code can be loaded more rapidly than relocatable code. For further information on the overlay loaders and the LDR and EXU requests, refer to section 11 in volume 2 of the reference manual.

LINK RELOCATABLE LOADER

The LINK relocatable loader is called by the absolute overlay loader as the result of a LINK call card request to load a file that is in relocatable binary format. Input to the LINK relocatable loader is object code generated by compilers and assemblers according to specifications for loader tables. Subprograms assembled or compiled independently, in relocatable or binary format, can be loaded and linked to each other or to library subroutines by the loader. The loader issues diagnostic messages to the dayfile and generates memory maps when requested.

The loader can generate overlays that are written on a specified file in absolute format. These overlays can be loaded later by the absolute overlay loader.

MEMORY MAP

Following completion of loading, an optional map of the user's field length is produced on file OUTPUT. A full map includes:

- Names, length, and positions of entry points with a sublist of all programs referencing the entry points
- Names and locations of common blocks
- Total length of all loaded programs and common blocks
- Length of the loader and its tables
- Unsatisfied external references

Map options can be selected using the MAP control card. (Refer to Loader Reference Manual.)

LINK CARD

The LINK control card specifies directives for the LINK loader. The LINK routine processes the loading of relocatable programs and generating of absolute binary files for present or future executuion. LINK does not load execute-only mode files.

The control card format is:

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

p_i Any of the following in any order:

F=lfn_1	Loads from file lfn_1.
F omitted	The system assumes lfn_1=LGO.
P	Satisfy external references from program library SYSLIB.
P=lfn_2	Satisfy external references from program library lfn_2.
P omitted	Satisfy external references from program library SYSLIB.

B	Write loaded program on file LGOB.
B=lfn_3	Write loaded program on file lfn_3.
B=0	No loaded program to write.
B omitted	The system assumes B=0.
L=lfn_4	Write load map on file lfn_4.
L=0	No map
L omitted	The default file name is OUTPUT but no map is produced unless either errors occur or some map options (LO parameter) are explicitly selected.
E=name	Load program with the specified entry point name from file lfn_1. If this option is selected, lfn_1 must be a user library file.
LO	Set map options S and B.
LO =c...c	Set map option S for statistics and errors and any of the following characters. B Block assignments E Entry points X External references and entry points.
X	Execute loaded program.

Since the loader resides in the program area, the user must have enough memory available to load the largest overlay in the file.

Many of the options on the control card are self-explanatory; however, the following examples illustrate some of the features of the LINK program.

If the user has generated a library file (refer to the description of the LIBGEN control card), he can load any program on that file.

For example, assume the user has file LIB1 attached to a control point. LIB1 contains routines TEST1, TEST2, and TEST3. TEST3 has the entry point TEST3E. The control card:

```
LINK(F=LIB1,E=TEST3E,X)
```

loads the routine TEST3 and begins execution at entry point TEST3E.

A more common example illustrates a relocatable load that generates an overlay using several different user libraries. The control cards:

```
MAP(FULL)
LIBRARY(MYLIB1)
LINK(F=BINREL, P=MYLIB0, L=MYMAP, B=BINABS)
```

generate an absolute overlay format [level (0, 0) unless BINREL contains overlay control cards specifying other levels] of file BINREL and write it on file BINABS. A full load map is written on file MYMAP. The externals are satisfied from libraries in the following order.

MYLIB0

LDSET tables (if any)

MYLIB1

SYSLIB

NOTE

The CDC CYBER loader does not satisfy externals from libraries in this order. (Refer to Loader Reference Manual.)

The following example illustrates the previous example using the CDC CYBER relocatable loader. The control cards:

```
LIBRARY(MYLIB1)
MAP(FULL)
LDSET(LIB=MYLIB0, MAP=/MYMAP)
LOAD(BINREL)
NOGO(BINABS)
```

generate an absolute overlay format of the file BINREL and write it on file BINABS, with map written on MYMAP. The externals are satisfied from libraries in the following order.

MYLIB1
MYLIB0
LDSET tables (if any)
SYSLIB

KRONOS 64-CHARACTER SET FOR TIME-SHARING TERMINALS

ASCII CODE TERMINAL†				CORRESPONDENCE CODE TERMINAL††				INTERNAL DISPLAY CODE (6/12-BIT OCTAL)
STANDARD PRINT		APL PRINT		STANDARD PRINT		APL PRINT		
CHAR.	CODE (8-BIT OCTAL)	CHAR.	CODE (8-BIT OCTAL)	CHAR.	CODE (7-BIT OCTAL)	CHAR.	CODE (7-BIT OCTAL)	
:	072	:	276	:	153	:	121	00
A	101	A	341	A	171	A	171	01
B	102	B	342	B	166	B	166	02
C	303	C	143	C	172	C	172	03
D	104	D	344	D	052	D	052	04
E	305	E	145	E	112	E	112	05
F	306	F	146	F	163	F	163	06
G	107	G	347	G	043	G	043	07
H	110	H	350	H	046	H	046	10
I	311	I	151	I	031	I	031	11
J	312	J	152	J	103	J	103	12
K	113	K	353	K	032	K	032	13
L	314	L	154	L	106	L	106	14
M	115	M	355	M	141	M	141	15
N	116	N	356	N	122	N	122	16
O	317	O	157	O	105	O	105	17
P	120	P	360	P	013	P	013	20
Q	321	Q	161	Q	133	Q	133	21
R	322	R	162	R	051	R	051	22
S	123	S	363	S	045	S	045	23
T	324	T	164	T	002	T	002	24
U	125	U	365	U	062	U	062	25
V	126	V	366	V	061	V	061	26
W	327	W	167	W	165	W	165	27
X	330	X	170	X	142	X	142	30
Y	131	Y	371	Y	147	Y	147	31
Z	132	Z	372	Z	124	Z	124	32
0	060	0	060	0	144	0	144	33
1	261	1	261	1	040	1	040	34
2	262	2	262	2	020	2	020	35
3	063	3	063	3	160	3	160	36
4	264	4	264	4	004	4	004	37
5	065	5	065	5	010	5	010	40
6	066	6	066	6	130	6	130	41
7	267	7	267	7	150	7	150	42
8	270	8	270	8	070	8	070	43
9	071	9	071	9	064	9	064	44
+	053	+	055	+	023	+	067	45
-	055	-	275	-	067	-	067	46
*	252	*	120	*	070	*	013	47
/	257	/	257	/	007	/	007	50
(	050	(	053	(	064	(	153	51
)	251	)	252	)	144	)	111	52
$	044	$	176	$	004	α	171	53
=	275	=	245	=	023	=	010	54

3AE4A

† THE OCTAL CODES LISTED FOR ASCII CODE TERMINALS ARE SHOWN WITH EVEN PARITY (NORMAL)

†† THE OCTAL CODES LISTED FOR CORRESPONDENCE CODE TERMINALS ARE SHOWN WITH ODD PARITY (NORMAL)

ASCII CODE TERMINAL				CORRESPONDENCE CODE TERMINAL				INTERNAL DISPLAY CODE (6/12-BIT OCTAL)
STANDARD PRINT		APL PRINT		STANDARD PRINT		APL PRINT		
CHAR.	CODE (8-BIT OCTAL)	CHAR.	CODE (8-BIT OCTAL)	CHAR.	CODE (7-BIT OCTAL)	CHAR.	CODE (7-BIT OCTAL)	
(SPACE)	240	(SPACE)	240	(SPACE)	100	(SPACE)	100	55
,	254	,	254	,	073	,	073	56
.	056	.	056	.	121	.	121	57
#	243	¨	041	#	160	¨	040	60
[	333	[	273	¼	001	[	153	61
]	335	]	072	½	001	]	111	62
%	245	÷	173	%	010	÷	023	63
"	042	≠	050	¨	111	≠	070	64
_	137†	_	306	_	067	_	163	65
!	041	∨	251	¢	130	∨	064	66
&	246	∧	137	&	150	∧	144	67
'	047	'	113	'	111	'	032	70
?	077	?	321	?	007	?	133	71
<	074	<	243	NULL	——	<	160	72
>	276	>	047	NULL	——	>	150	73
@	300	≤	044	@	020	≤	004	74
\	134	\	077	NULL	——	\	007	75
^	176	¯	042	NULL	——	¯	020	76
;	273	;	074	;	153	;	073	77
`	140	NULL	——	NULL	——	NULL	——	7600
a	341	α	101	a	171	α	171	7601
b	342	⊥	102	b	166	⊥	166	7602
c	143	∩	303	c	172	∩	172	7603
d	344	⌊	104	d	052	⌊	052	7604
e	145	∈	305	e	112	∈	112	7605
f	146	X	134	f	163	X	023	7606
g	347	∇	107	g	043	∇	043	7607
h	350	Δ	110	h	046	Δ	046	7610
i	151	ι	311	i	031	ι	031	7611
j	152	∘	312	j	103	∘	103	7612
k	353	¢	336	k	032	NULL	——	7613
l	154	□	314	l	106	□	106	7614
m	355	|	115	m	141	|	141	7615
n	356	T	116	n	122	T	122	7616
o	157	0	317	o	105	0	105	7617
p	360	←	100	p	013	←	001	7620
q	161	→	134	q	133	→	101	7621
r	162	ρ	322	r	051	ρ	051	7622
s	363	⌈	123	s	045	⌈	045	7623
t	164	~	324	t	002	~	002	7624
u	365	↓	125	u	062	↓	062	7625
v	366	U	126	v	061	U	061	7626
w	167	ω	327	w	165	ω	165	7627
x	170	⊃	330	x	142	⊃	142	7630
y	371	↑	131	y	147	↑	147	7631

3AE3A

† ON TTY MODELS HAVING NO UNDERLINE, THE BACKARROW (←) TAKES ITS PLACE

ASCII CODE TERMINAL				CORRESPONDENCE CODE TERMINAL				INTERNAL DISPLAY CODE (6/12-BIT OCTAL)
STANDARD PRINT		APL PRINT		STANDARD PRINT		APL PRINT		
CHAR.	CODE (8-BIT OCTAL)	CHAR.	CODE (8-BIT OCTAL)	CHAR.	CODE (8-BIT OCTAL)	CHAR.	CODE (8-BIT OCTAL)	
Z	372	⊂	132	Z	124	⊂	124	7632
{	173	{	140	NULL	——	NULL	——	7633
:	174	≥	246	±	040	≥	130	7634
}	175	}	374	NULL	——	NULL	——	7635
~	176	≡	175	NULL	——	NULL	——	7636
DEL	377	DEL	377	NULL	——	NULL	——	7637
NUL	000	NUL	000	NUL	075	NUL	075	7640
SOH	201	SOH	201	SOA	166	SOA	166	7641
STX	202	STX	202	EOA	064	EOA	064	7642
ETX	003	ETX	003	NULL	——	NULL	——	7643
EOT	204	EOT	204	EOT	174	EOT	174	7644
ENQ	005	ENQ	005	NULL	——	NULL	——	7645
ACK	006	ACK	006	ACK	067	NULL	——	7646
BELL	207	BELL	207	NULL	——	NULL	——	7647
BS	210	BS	210	BS	135	BS	135	7650
HT	011	HT	011	HT	057	HT	057	7651
LF	012	LF	012	LF	156	LF	156	7652
VT	213	VT	213	NULL	——	NULL	——	7653
FF	014	FF	014	NULL	——	NULL	——	7654
CR	215	CR	215	CR	155	CR	155	7655
SO	216	SO	216	UCS	034	UCS	034	7656
SI	017	SI	017	LCS	037	LCS	037	7657
DLE	220	DLE	220	NULL	——	NULL	——	7660
DC1	021	DC1	021	NULL	——	NULL	——	7661
DC2	022	DC2	022	NULL	——	NULL	——	7662
DC3	023	DC3	023	NULL	——	NULL	——	7663
DC4	024	DC4	024	STO	054	STO	064	7664
NAK	225	NAK	225	NAK	001	NAK	001	7665
SYN	226	SYN	226	IL	075	IL	075	7666
ETB	027	ETB	027	EOB	136	EOB	136	7667
CAN	030	CAN	030	DEL	177	DEL	137	7670
EM	231	EM	231	NULL	——	NULL	——	7671
SUB	232	SUB	232	NULL	——	NULL	——	7672
ESC	033	ESC	033	PF	076	PF	076	7673
FS	234	FS	234	NULL	——	NULL	——	7674
GS	035	GS	035	NULL	——	NULL	——	7675
RS	036	RS	036	NULL	——	NULL	——	7676
US	237	US	237	NULL	——	NULL	——	7677
NULL	——	NULL	——	NULL	——	NULL	——	7400
@	300	≤	044	@	020	≤	004	7401
^	176	¯	042	NULL	——	¯	020	7402
NULL	——	NULL	——	CNL	001	CNL	001	7403
NULL	——	NULL	——	NULL	——	NULL	——	7404
NULL	——	NULL	——	NULL	——	NULL	——	7405
NULL	——	NULL	——	NULL	——	NULL	——	7406
NULL	——	NULL	——	NULL	——	NULL	——	7407

3AE5A

KRONOS 63-CHARACTER SET FOR TIME-SHARING TERMINALS

ASCII CODE TERMINAL†				CORRESPONDENCE CODE TERMINAL††				INTERNAL DISPLAY CODE (6/12-BIT OCTAL)
STANDARD PRINT		APL PRINT		STANDARD PRINT		APL PRINT		
CHAR.	CODE (8-BIT OCTAL)	CHAR.	CODE (8-BIT OCTAL)	CHAR.	CODE (7-BIT OCTAL)	CHAR.	CODE (7-BIT OCTAL)	
NULL	——	NULL	——	NULL	——	NULL	——	00
A	101	A	341	A	171	A	171	01
B	102	B	342	B	166	B	166	02
C	303	C	143	C	172	C	172	03
D	104	D	344	D	052	D	052	04
E	305	E	145	E	112	E	112	05
F	306	F	146	F	163	F	163	06
G	107	G	347	G	043	G	043	07
H	110	H	350	H	046	H	046	10
I	311	I	151	I	031	I	031	11
J	312	J	152	J	103	J	103	12
K	113	K	353	K	032	K	032	13
L	314	L	154	L	106	L	106	14
M	115	M	355	M	141	M	141	15
N	116	N	356	N	122	N	122	16
O	317	O	157	O	105	O	105	17
P	120	P	360	P	013	P	013	20
Q	321	Q	161	Q	133	Q	133	21
R	322	R	162	R	051	R	051	22
S	123	S	363	S	045	S	045	23
T	324	T	164	T	002	T	002	24
U	125	U	365	U	062	U	062	25
V	126	V	366	V	061	V	061	26
W	327	W	167	W	165	W	165	27
X	330	X	170	X	142	X	142	30
Y	131	Y	371	Y	147	Y	147	31
Z	132	Z	372	Z	124	Z	124	32
0	060	0	060	0	144	0	144	33
1	261	1	261	1	040	1	040	34
2	262	2	262	2	020	2	020	35
3	063	3	063	3	160	3	160	36
4	264	4	264	4	004	4	004	37
5	065	5	065	5	010	5	010	40
6	066	6	066	6	130	6	130	41
7	267	7	267	7	150	7	150	42
8	270	8	270	8	070	8	070	43
9	071	9	071	9	064	9	064	44
+	053	+	055	+	023	+	067	45
-	055	-	275	-	067	-	067	46
*	252	*	120	*	070	*	013	47
/	257	/	257	/	007	/	007	50
(	050	(	053	(	064	(	153	51
)	251	)	252	)	144	)	111	52
$	044	$	176	$	004	α	171	53
=	275	=	245	=	023	=	010	54

3AE4A

† THE OCTAL CODES LISTED FOR ASCII CODE TERMINALS ARE SHOWN WITH EVEN PARITY (NORMAL)

†† THE OCTAL CODES LISTED FOR CORRESPONDENCE CODE TERMINALS ARE SHOWN WITH ODD PARITY (NORMAL)

ASCII CODE TERMINAL				CORRESPONDENCE CODE TERMINAL				INTERNAL DISPLAY CODE (6/12-BIT OCTAL)
STANDARD PRINT		APL PRINT		STANDARD PRINT		APL PRINT		
CHAR.	CODE (8-BIT OCTAL)	CHAR.	CODE (8-BIT OCTAL)	CHAR.	CODE (7-BIT OCTAL)	CHAR.	CODE (7-BIT OCTAL)	
(SPACE)	240	(SPACE)	240	(SPACE)	100	(SPACE)	100	55
,	254	,	254	,	073	,	073	56
.	056	.	056	.	121	.	121	57
"	042	¨	041	¨	111	¨	040	60
[	333	[	273	¼	001	[	153	61
]	335	]	072	½	001	]	111	62
:	072	:	276	:	153	:	121	63
'	047	'	113	'	111	'	032	64
&	246	X	134	&	150	X	023	65
CR	215	CR	215	NULL	——	NULL	——	66
LF	012	LF	012	LF	156	LF	156	67
↑	336	¯	042	NULL	——	¯	020	70
#	243	¢	336	#	160	≠	070	71
<	074	<	243	NULL	——	<	160	72
>	276	>	047	NULL	——	>	150	73
(ESC 1)	——	NULL	——	NULL	——	NULL	——	74
?	077	?	321	?	007	?	133	75
(ESC 2)	——	NULL	——	NULL	——	NULL	——	76
;	273	;	074	;	153	;	073	77
NULL	——	NULL	——	NULL	——	NULL	——	7600
a	341	α	101	a	171	α	171	7601
b	342	⊥	102	b	166	⊥	166	7602
c	143	∩	303	c	172	∩	172	7603
d	344	⌊	104	d	052	⌊	052	7604
e	145	∈	305	e	112	∈	112	7605
f	146	∧	137	f	163	_	163	7606
g	347	∇	107	g	043	∇	043	7607
h	350	Δ	110	h	046	Δ	046	7610
i	151	ι	311	i	031	ι	031	7611
j	152	∘	312	j	103	∘	103	7612
k	353	←	100	k	032	≤	004	7613
l	154	□	314	l	106	□	106	7614
m	355	→	134	m	141	|	141	7615
n	356	T	116	n	122	T	122	7616
o	157	0	317	o	105	0	105	7617
p	360	≤	044	p	013	≥	130	7620
q	161	≥	246	q	133	?	133	7621
r	162	ρ	322	r	051	ρ	051	7622
s	363	⌈	123	s	045	⌈	045	7623
t	164	≠	050	t	002	~	002	7624
u	365	↓	125	u	062	↓	062	7625
v	366	∪	126	v	061	∪	061	7626
w	167	ω	327	w	165	ω	165	7627
x	170	⊃	330	x	142	⊃	142	7630
y	371	↑	131	y	147	↑	147	7631

3AE3A

ASCII CODE TERMINAL				CORRESPONDENCE CODE TERMINAL				INTERNAL DISPLAY CODE (6/12-BIT OCTAL)
STANDARD PRINT		APL PRINT		STANDARD PRINT		APL PRINT		
CHAR.	CODE (8-BIT OCTAL)	CHAR.	CODE (8-BIT OCTAL)	CHAR.	CODE (8-BIT OCTAL)	CHAR.	CODE (8-BIT OCTAL)	
z	372	⊂	132	z	124	⊂	124	7632
DLE	220	DLE	220	NULL	——	NULL	——	7633
BELL	207	BELL	207	NULL	——	NULL	——	7634
DC2	022	DC2	022	NULL	——	NULL	——	7635
ETX	003	ETX	003	NULL	——	NULL	——	7636
DC4	024	DC4	024	NULL	——	NULL	——	7637
NAK	225	NAK	225	NULL	——	NULL	——	7640
SYN	226	SYN	226	NULL	——	NULL	——	7641
ETB	027	ETB	027	NULL	——	NULL	——	7642
CAN	030	CAN	030	NULL	——	NULL	——	7643
EM	231	EM	231	NULL	——	NULL	——	7644
VT	213	VT	213	NULL	——	NULL	——	7645
SOH	201	SOH	201	NULL	——	NULL	——	7646
!	041	V	251	NULL	——	NULL	——	7647
SI	017	SI	017	NULL	——	NULL	——	7650
BS	210	BS	210	BS	135	BS	135	7651
HT	011	HT	011	HT	057	HT	057	7652
EOT	204	EOT	204	NULL	——	NULL	——	7653
GS	035	GS	035	NULL	——	NULL	——	7654
NUL	000	NUL	000	NUL	075	NUL	075	7655
FF	014	FF	014	,	073	NULL	——	7656
SO	216	SO	216	.	121	NULL	——	7657
STX	202	STX	202	NULL	——	NULL	——	7660
{	173	{	140	NULL	——	←	001	7661
}	175	}	374	NULL	——	→	101	7662
SUB	232	SUB	232	NULL	——	NULL	——	7663
ACK	006	ACK	006	NULL	——	NULL	——	7664
&	246	NULL	——	NULL	——	NULL	——	7665
\	134	\	077	NULL	——	\	007	7666
:	174	⌶	115	¢	130	⌶	141	7667
~	176	~	324	±	040	~	002	7670
#	243	NULL	——	NULL	——	NULL	——	7671
FS	234	FS	234	NULL	——	NULL	——	7672
RS	036	RS	036	NULL	——	NULL	——	7673
DEL	377	DEL	377	NULL	——	NULL	——	7674
US	237	US	237	NULL	——	NULL	——	7675
NL	——	NL	——	NL	155	NL	155	7676
ESC	033	ESC	033	NULL	——	NULL	——	7677
NULL	——	NULL	——	NULL	——	NULL	——	7400
@	300	≡	175	@	020	∧	144	7401
%	245	÷	173	%	010	÷	023	7402
`	140	≠	335	NULL	——	∨	064	7403
_	137 †	_	306	_	067	NULL	——	7404
X-ON	021	X-ON	021	NULL	——	NULL	——	7405
X-OFF	223	X-OFF	223	NULL	——	NULL	——	7406
ENQ	005	ENQ	005	NULL	——	NULL	——	7407

3AE5A

† ON TTY MODELS HAVING NO UNDERLINE, THE BACKARROW (←) TAKES ITS PLACE.

KRONOS STANDARD CHARACTER SET

CDC GRAPHIC	ASCII GRAPHIC SUBSET	DISPLAY CODE	HOLLERITH PUNCH (026)	EXTERNAL BCD CODE	ASCII PUNCH (029)	ASCII CODE
:†	:	00†	8-2	00	8-2	3A
A	A	01	12-1	61	12-1	41
B	B	02	12-2	62	12-2	42
C	C	03	12-3	63	12-3	43
D	D	04	12-4	64	12-4	44
E	E	05	12-5	65	12-5	45
F	F	06	12-6	66	12-6	46
G	G	07	12-7	67	12-7	47
H	H	10	12-8	70	12-8	48
I	I	11	12-9	71	12-9	49
J	J	12	11-1	41	11-1	4A
K	K	13	11-2	42	11-2	4B
L	L	14	11-3	43	11-3	4C
M	M	15	11-4	44	11-4	4D
N	N	16	11-5	45	11-5	4E
O	O	17	11-6	46	11-6	4F
P	P	20	11-7	47	11-7	50
Q	Q	21	11-8	50	11-8	51
R	R	22	11-9	51	11-9	52
S	S	23	0-2	22	0-2	53
T	T	24	0-3	23	0-3	54
U	U	25	0-4	24	0-4	55
V	V	26	0-5	25	0-5	56
W	W	27	0-6	26	0-6	57
X	X	30	0-7	27	0-7	58
Y	Y	31	0-8	30	0-8	59
Z	Z	32	0-9	31	0-9	5A
0	0	33	0	12	0	30
1	1	34	1	01	1	31
2	2	35	2	02	2	32
3	3	36	3	03	3	33
4	4	37	4	04	4	34
5	5	40	5	05	5	35

3AE13A

† TWELVE OR MORE ZERO BITS AT THE END OF A 60-BIT WORD ARE INTERPRETED AS END-OF-LINE MARK RATHER THAN TWO COLONS. END-OF-LINE MARK IS CONVERTED TO EXTERNAL BCD 1632.

CDC GRAPHIC	ASCII GRAPHIC SUBSET	DISPLAY CODE	HOLLERITH PUNCH (026)	EXTERNAL BCD CODE	ASCII PUNCH (029)	ASCII CODE
6	6	41	6	06	6	36
7	7	42	7	07	7	37
8	8	43	8	10	8	38
9	9	44	9	11	9	39
+	+	45	12	60	12-8-6	2B
-	-	46	11	40	11	2D
*	*	47	11-8-4	54	11-8-4	2A
/	/	50	0-1	21	0-1	2F
(	(	51	0-8-4	34	12-8-5	28
)	)	52	12-8-4	74	11-8-5	29
$	$	53	11-8-3	53	11-8-3	24
=	=	54	8-3	13	8-6	3D
BLANK	BLANK	55	NO PUNCH	20	NO PUNCH	20
,(COMMA)	,(COMMA)	56	0-8-3	33	0-8-3	2C
.(PERIOD)	.(PERIOD)	57	12-8-3	73	12-8-3	2E
≡	#	60	0-8-6	36	8-3	23
[	[	61	8-7	17	12-8-2	5B
]	]	62	0-8-2	32	11-8-2	5D
%††	%	63	8-6	16	0-8-4	25
≠	"(QUOTE)	64	8-4	14	8-7	22
→	_(UNDERLINE)	65	0-8-5	35	0-8-5	5F
∨	!	66	11-0	52	12-8-7	21
∧	&	67	0-8-7	37	12	26
↑	'(APOSTROPHE)	70	11-8-5	55	8-5	27
↓	?	71	11-8-6	56	0-8-7	3F
<	<	72	12-0	72	12-8-4	3C
>	>	73	11-8-7	57	0-8-6	3E
≤	@	74	8-5	15	8-4	40
≥	\	75	12-8-5	75	0-8-2	5C
¬	~(CIRCUMFLEX)	76	12-8-6	76	11-8-7	5E
;(SEMICOLON)	; (SEMICOLON)	77	12-8-7	77	11-8-6	3B

3AE6A

†† IN INSTALLATIONS USING THE CDC 63-GRAPHIC SET, DISPLAY CODE 00 HAS NO ASSOCIATED GRAPHIC OR HOLLERITH CODE; DISPLAY CODE 63 IS THE COLON(8-2 PUNCH). THE SELECTION OF THE 63- OR 64-CHARACTER SET FOR TAPES IS AN INSTALLATION OPTION.

ASCII/DISPLAY CODE AND EBCDIC/DISPLAY CODE CONVERSION

DISPLAY CODE		ASCII				EBCDIC			
		UPPERCASE		LOWERCASE		UPPERCASE		LOWERCASE	
OCTAL	CHAR	CHAR	HEX	CHAR	HEX	CHAR	HEX	CHAR	HEX
00	:	:	3A	SUB	1A	:	7A	SUB	3F
01	A	A	41	a	61	A	C1	a	81
02	B	B	42	b	62	B	C2	b	82
03	C	C	43	c	63	C	C3	c	83
04	D	D	44	d	64	D	C4	d	84
05	E	E	45	e	65	E	C5	e	85
06	F	F	46	f	66	F	C6	f	86
07	G	G	47	g	67	G	C7	g	87
10	H	H	48	h	68	H	C8	h	88
11	I	I	49	i	69	I	C9	i	89
12	J	J	4A	j	6A	J	D1	j	91
13	K	K	4B	k	6B	K	D2	k	92
14	L	L	4C	l	6C	L	D3	l	93
15	M	M	4D	m	6D	M	D4	m	94
16	N	N	4E	n	6E	N	D5	n	95
17	O	O	4F	o	6F	O	D6	o	96
20	P	P	50	p	70	P	D7	p	97
21	Q	Q	51	q	71	Q	D8	q	98
22	R	R	52	r	72	R	D9	r	99
23	S	S	53	s	73	S	E2	s	A2
24	T	T	54	t	74	T	E3	t	A3
25	U	U	55	u	75	U	E4	u	A4
26	V	V	56	v	76	V	E5	v	A5
27	W	W	57	w	77	W	E6	w	A6
30	X	X	58	x	78	X	E7	x	A7
31	Y	Y	59	y	79	Y	E8	y	A8
32	Z	Z	5A	z	7A	Z	E9	z	A9
33	0	0	30	DLE	10	0	F0	DLE	10
34	1	1	31	DC1	11	1	F1	DC1	11
35	2	2	32	DC2	12	2	F2	DC2	12
36	3	3	33	DC3	13	3	F3	TM	13
37	4	4	34	DC4	14	4	F4	DC4	3C

3AE7A

DISPLAY CODE		ASCII				EBCDIC			
		UPPERCASE		LOWERCASE		UPPERCASE		LOWERCASE	
OCTAL	CHAR	CHAR	HEX	CHAR	HEX	CHAR	HEX	CHAR	HEX
40	5	5	35	NAK	15	5	F5	NAK	3D
41	6	6	36	SYN	16	6	F6	SYN	32
42	7	7	37	ETB	17	7	F7	ETB	26
43	8	8	38	CAN	18	8	F8	CAN	18
44	9	9	39	EM	19	9	F9	EM	19
45	+	+	2B	VT	0B	+	4E	VT	0B
46	-	-	2D	CR	0D	-	60	CR	0D
47	*	*	2A	LF	0A	*	5C	LF	25
50	/	/	2F	SI	0F	/	61	SI	0F
51	(	(	28	BS	08	(	4D	BS	16
52	)	)	29	HT	09	)	5D	HT	05
53	$	$	24	EOT	04	$	5B	EOT	37
54	=	=	3D	GS	1D	=	7E	IGS	1D
55	SP	SP	20	NUL	00	SP	40	NUL	00
56	,	,	2C	FF	0C	,	6B	FF	0C
57	.	.	2E	SO	0E	.	4B	SO	0E
60	≡	#	23	ETX	03	#	7B	ETX	03
61	[	[	5B	FS	1C	¢	4A	IFS	1C
62	]	]	5D	SOH	01	!	5A	SOH	01
63	%	%	25	ENQ	05	%	6C	ENQ	2D
64	≠	"	22	STX	02	"	7F	STX	02
65	↱	_	5F	DEL	7F	_	6D	DEL	07
66	∨	!	21	}	7D	|	4F	}	D0
67	∧	&	26	ACK	06	&	50	ACK	2E
70	↑	'	27	BEL	07	'	7D	BEL	2F
71	↓	?	3F	US	1F	?	6F	IUS	1F
72	<	<	3C	{	7B	<	4C	{	C0
73	>	>	3E	RS	1E	>	6E	IRS	1E
74	≤	@	40	`	60	@	7C	`	79
75	≥	\	5C	¦	7C	\	E0	¦	6A
76	¬	^	5E	~	7E	¬	5F	~	A1
77	;	;	3B	ESC	1B	;	5E	ESC	27

3AE8A

CARRIAGE CONTROL CHARACTERS

CHARACTER	COMMAND
SPACE	SINGLE SPACE
1	EJECT PAGE BEFORE PRINT
0	SKIP ONE LINE BEFORE PRINT (DOUBLE SPACE)
–	SKIP TWO LINES BEFORE PRINT (TRIPLE SPACE)
+	SUPPRESS SPACE BEFORE PRINT
/	SUPPRESS SPACE AFTER PRINT
2	SKIP TO LAST LINE OF FORM BEFORE PRINT
8	SKIP TO FORMAT CHANNEL 1 BEFORE PRINT †
7	SKIP TO FORMAT CHANNEL 2 BEFORE PRINT †
6	SKIP TO FORMAT CHANNEL 3 BEFORE PRINT †
5	SKIP TO FORMAT CHANNEL 4 BEFORE PRINT †
4	SKIP TO FORMAT CHANNEL 5 BEFORE PRINT †
3	SKIP TO FORMAT CHANNEL 6 BEFORE PRINT †
H	SKIP TO FORMAT CHANNEL 1 AFTER PRINT
G	SKIP TO FORMAT CHANNEL 2 AFTER PRINT
F	SKIP TO FORMAT CHANNEL 3 AFTER PRINT
E	SKIP TO FORMAT CHANNEL 4 AFTER PRINT
D	SKIP TO FORMAT CHANNEL 5 AFTER PRINT
C	SKIP TO FORMAT CHANNEL 6 AFTER PRINT
Q	CLEAR AUTO EJECT; REMAINDER OF LINE IS NOT PRINTED
R	SET AUTO EJECT; REMAINDER OF LINE IS NOT PRINTED
S	SELECT 6 LINES/INCH; †† REMAINDER OF LINE IS NOT PRINTED
T	SELECT 8 LINES/INCH; †† REMAINDER OF LINE IS NOT PRINTED

3AE9A

†No space after print. For all other control characters, a line feed is issued after print.

††Used only on the 512 and 580 line printers. The deselection of auto eject mode on a 512 or 580 line printer results in the deselection of 8 lines per inch, if previously selected.

DAYFILE MESSAGES B

This appendix contains an alphabetical listing of the messages which may appear in a user's dayfile. Lowercase characters are used to identify variable names or fields. If the first word or characters are variable, the message is listed according to the second word. For example, the message:

pfn ALREADY PERMANENT, AT nnn.

is listed alphabetically with the messages beginning with the letter A. This is done because the variable pfn (permanent file name) may change each time the message is issued. All messages beginning with numbers follow the alphabetical listing.

The CIO and PFM file processors utilize the file environment table (FET) as a communication area to contain information about the requests of a user's job. Higher level languages (COBOL, FORTRAN, etc.) automatically establish and use these areas but the COMPASS programmer must define the FET. (Refer to volume 2 for detailed information on the FET.) CIO and PFM error messages contain the address, nnn, of the FET associated with the request and the logical file name, fff, from word zero of the table (FET+0).

When the error processing (ep) bit is set in word 1 of the FET, status information is returned by the function processor when an abnormal situation or error occurs. The abnormal termination codes are returned to bits 10 through 13 of word zero of the FET (bits 10 through 17 of PFM). Following the alphabetical listing of messages is a list of LFM and PFM error codes and explanations. Also included at the end of this section is a table summary of the action taken by PFM when an error is detected while reading mass storage.

Message	Routine	Description
ADDRESS ERROR.	TCS	CM address in call is beyond the field length.
ADDRESS OUT OF RANGE aaaaaa.	CPMEM	The address aaaaaa on a correction statement is greater than or equal to the user's field length. The correction statement is ignored and LOC continues.
pfn ALREADY PERMANENT, AT nnn.	PFM	The user has already saved or defined a file with the name specified.
ARG. ERROR.	LDR	LDR parameters were outside the user's field length.
ARGUMENT ERROR.	RESEX/ISF	A control statement is syntactically incorrect. Recheck parameters. On tape management statements, the system issues this message if both ring enforcement options (PO=R and PO=W) or more than one EOT option (PO=I, PO=P, PO=S) is specified.
ARITH, ERROR x AT yyyyyy.	3AB	The monitor detected an arithmetic error condition x at address yyyyyy.
BAD DECK NAME.	TCS	A deck name of more than 7 characters was encountered.
BINARY SEQ. ERROR, RECxxxx CDyyyy.	CIO	A binary card was found to be out of sequence and the job was terminated. xxxx Number (in octal) of record in which sequence error occurred. yyyy Number (in octal) of card within the record which caused the sequence error.
BLANK TAPE, fff AT nnn.	CIO	A blank tape was encountered on a read operation. (Blank tape is defined as more than 25 feet of erased tape.)
BLOCK COUNT ERROR IN TRAILER LABEL, fff AT nnn.	CIO	The block count in the EOF1 or EOV1 label did not match the block count maintained by the tape executive during the read operation.
BLOCK LENGTH ERROR ON fff AT nnn.	CIO	The software-recorded block length did not match the length of the block read (this applies to I format tapes only).
BLOCK SEQUENCE ERROR, fff AT nnn.	CIO	The software-recorded block length did not match the length of the block read, or the block number did not match the software-record block number. This message applies to I format tapes only.
BLOCK TOO LARGE ON fff AT nnn.	CIO	The tape being read contained a data block greater in size than that allowed by the specified format or by user declaration. (S or L format tapes only.)
BOT/EOT ENCOUNTERED, fff AT nnn.	CIO	Indicates an abnormal tape position.
BUFFER ARG. ERROR.	TCS	CM address in call is not less than the field length minus the word count; buffer extends past the job's field length.
BUFFER ARGUMENT ERROR.	QFM	A buffer pointer did not conform to the following constraints: FIRST $\leq$ IN FIRST $\leq$ OUT OUT $<$ LIMIT $\leq$ FL
BUFFER ARGUMENT ERROR ON fff AT nnn.	CIO	A buffer pointer did not conform to the following constraints: FIRST $\leq$ IN FIRST $\leq$ OUT OUT $<$ LIMIT $\leq$ FL The system provides a dump of the FET on file OUTPUT.

Message	Routine	Description
BUFFER CONTROL WORD ERROR ON fff AT nnn.	CIO	The block length specified during a write operation was greater than the allowable PRU size for the device. For tape operations, this message can also indicate that the unused bit count is illegal.
pfn BUSY, AT nnn.	PFM	The specified direct access file is attached in the opposite mode.
n CARD(S) NOT PROCESSED.	CIO	Errors on n directives prevented them from being processed.
CATALOG OVERFLOW - FILES, AT nnn.	PFM	The number of files in the user's catalog exceeds his limit (refer to LIMITS control statement, section 6).
CATALOG OVERFLOW - SIZE AT nnn.	PFM	The cumulative size of the indirect access files in the user's catalog exceeds his limit (refer to LIMITS control statement, section 6).
CHANNEL MALFUNCTION, fff AT nnn.	CIO	Hardware malfunction.
CHARGE ABORTED.	CHARGE	A central site operator action caused the CHARGE operation to abnormally terminate. Resubmit job.
CHARGE FILE BUSY.	CHARGE	The file which the system uses to validate charge and project number is busy. Resubmit job.
CHARGE ILLEGAL AT THIS HOUR.	CHARGE	The specified project number cannot be used at this time of the day.
CHECKPOINT nnnn COMPLETE.	CHKPT	Indicates that checkpoint nnnn has been completed. Issued if only one checkpoint file is present. For a checkpoint operation, more than two checkpoint files or an illegal combination of checkpoint files was specified.
CHECKPOINT nnnn COMPLETED TO xxxxxxx.	CHKPT	Indicates that checkpoint nnnn has been completed to file xxxxxxx. Issued if alternate CB checkpoint files are used.
CHECKPOINT FILE ERROR.	CHKPT/RESTART	During a restart operation, either the checkpoint file lfn specified on the RESTART control statement was empty or RESTART detected a format error attempting to read the specified checkpoint file.
CHECKPOINT NOT FOUND.	RESTART	The specified checkpoint (nn parameter on RESTART statement) could not be found on the file.
CKP REQUEST.	CHKPT	A checkpoint has been initiated.
CM NOT VALIDATED.	TCS	The number of CM words specified on the job statement exceeds that for which the user is validated.
COMPILER NOT IN LIBRARY.	TCS	An LDC control statement requested loading of a compiler not on the system.
CONTROL CARD ARGUMENT ERROR.	QFSP	An invalid argument was encountered on a control statement.
CONTROL CARD ERROR.		Loader failed to find the requested file.
CONTROL STATEMENT LIMIT.	1AJ	The number of control statements processed for a job has exceeded the limit for which the user is validated.
CONVERSION NOT FOUND.	PFM	The conversion table specified by the TS option was not found.

Message	Routine	Description
CONVERSION NOT SPECIFIED.	PFM	Neither a TS nor 64 option was specified on a CONVERT control statement.
CORE OVERFLOW, JOB ABORTED.	CPM	Table overflow occurred; rerun using more central memory field length.
CPxx,... .		Refer to the EQxx,... series of corresponding messages for full descriptions of messages beginning with CPxx,... .
CPM ARG. ERROR.	CPM	Error(s) encountered and job aborted.
CPM ILLEGAL REQUEST.	CPM	A CPM function was issued without the auto recall specified or job was not of system origin.
CPU ERROR EXIT xx AT yyyyyy.	1AJ	Monitor has detected a CPU error exit condition xx at address yyyyyy. (Refer to Error Control, section 3.)
CRxx,... .		Refer to the EQxx,... series of corresponding messages for full descriptions of messages beginning with CRxx,... .
DATA BASE ERROR.	PROFILE/ CHARGE/ MODVAL	One of the following: The system detected an error in its validation file. The user should contact installation personnel. PROFILE detected a format error in the charge or project number entries of the project profile file.
DATA TRANSFER ERROR, AT nnn.	PFM	An error occurred in a read operation during a file transfer.
DAYFILE TERMINATED	SFM	Informative message issued to the terminated dayfile.
xxxxxx DAYFILE TERMINATED.	SFM	Informative message indicating dayfile xxxxxx has been terminated. (Issued to system and control point dayfiles.)
DExx, Cyy, ec, ann, Sttt, Axxxxxxx.	6DE	An error has been detected on extended core storage. The nature of the error is determined by examining each parameter in the message. xx — EST ordinal of ECS unit yy — Channel number ec — Error code; one of the following: PE Parity error/checkword error AD Address error ST Device status error RS Device reserved NR Device not ready a — Type of operation; one of the following: R Read W Write nn — Retry count; error is considered irrecoverable after following number of retries: PE 10 AD 10 ST 64

Message	Routine	Description
		FN 10 RS Indefinite NR Indefinite tttt Device status; implies there was an incomplete transfer if status does not indicate an error Axxxxxxx Physical address at beginning of block
DEMAND EXCEEDED.	RESEX	The user attempted to assign more units than he scheduled on the RESOURC statement.
DEMAND INSTALLATION ERROR.	RESEX	The user requested more units than exist at the installation.
DEMAND VALIDATION ERROR.	RESEX	The specified number of units exceeds the user's validation limits.
DENSITY CHANGE, fff AT nnn.	CIO	The density of the tape changed during a read or write operation. If this error occurs on the first block on the tape, the additional message: DENSITY SPECIFIED DIFFERENT FROM TAPE. is issued. In requesting the tape, the user should specify the density in which the tape was written.
DEVICE ERROR ON FILE fff AT nnn.	CIO	An irrecoverable error occurred on the mass storage device containing the file fff.
DEVICE UNABAILABLE, AT nnn.	PFM	Access to the permanent file device requested is not possible. User may have attempted to access files on a device not present in the alternate system.
DIxx, Cyy, ec, ann, Stttt, FNqqqq. or DIxx, Cyy, ec, ann, Stttt, Uxx Cxxx Txx Sxx.	6DI	An error has been detected on mass storage device xx. The nature of the error is determined by examining each parameter in the message. xx EST ordinal of 844 disk yy Channel number cc Error code - one of the following: PE Parity error/checkword error AD Address error ST Device status error FT Function timed out with no response RS Device reserved NR Device not ready a Type of operation - one of the following: R Read W Write nn Retry count - error is considered irrecoverable after following number of retries: PE 10 AD 10 ST 64 FT 3 RS indefinite NR indefinite

Message	Routine	Description
		tttt — Device status - implies there was an incomplete transfer if status does not indicate an error qqqq — Function rejected Uxx — Physical unit Cxxx — Physical cylinder Txx — Physical track Sxx — Physical sector } physical address
DIRECT ACCESS DEVICE ERROR, AT nnn.	PFM	The specified file already exists on a device other than the device requested or an illegal device type was specified. The device on which the file resides may not contain direct access files because: 1. The device is not specified as a direct access device in the catalog descriptor table. 2. The device is not specified as ON and initialized in the catalog descriptor table. 3. The device is a dedicated indirect access permanent file device. If on an alternate system, the user's master device may not have been transferred to that system.
DIRECTIVE ERROR.	PROFILE	A delete (DC, DP, or DU) directive was encountered during a create run. The delete directive is ignored; all other directives are processed normally.
DIRECTIVE ERRORS.	PROFILE/ MODVAL/ OPLEDIT	An invalid directive statement was encountered. If the message is issued in response to a PROFILE request, refer to the specific directive errors as listed for output file diagnostics.
n DIRECTIVE ERROR(S).	PROFILE	Occurs when there are conflicts or omissions in implied deletes or insertions.
DIRECTIVE CARD ERROR.	PROFILE	Occurs when an illegal directive statement is encountered; for example, syntax error.
DNdn FMxxxxxxx FNT FULL.	QREC	The FNT was filled while recovering the specified device: dn — Device number xxxxxxx — Family name
DNdn FMxxxxxxx IQFT INTERLOCKED.	QREC	The track interlock on the IQFT file is set. dn — Device number xxxxxxx — Family name
DNdn FMxxxxxxx MS ERROR.	QREC	A mass storage error occurred while processing the IQFT file on the specified device. dn — Device number xxxxxxx — Family name
DNdn FMxxxxxxx NO IQFT FILE.	QREC	No IQFT file exists for the specified device. dn — Device number xxxxxxx — Family name
DNdn FMxxxxxxx UNDEFINED ERROR.	QREC	System failure has occurred generating an erroneous error code. dn — Device number xxxxxxx — Family name

Message	Routine	Description
DPxx, Cyy, ec, ann, Stttt, Axxxxxxx.	6DP	An error has been detected on distributive data path (DDP). The nature of the error is determined by examining each parameter in the message. xx — EST ordinal of DDP/ECS yy — Channel number ec — Error code; one of the following: PE Parity error/checkword error AD Address error ST Device status error RS Device reserved NR Device not ready a — Type of operation; one of the following: R Read W Write nn — Retry count; error is considered irrecoverable after following number of retries: PE 10 AD 10 ST 64 FN 10 RS Indefinite NR Indefinite tttt — Device status; implies there was an incomplete transfer if status does not indicate an error Axxxxxxx — Physical address at beginning of block
DUMP FWA .GE. LWA+1.	CPMEM	The first word address of memory to be dumped was greater than the last word address plus 1 of memory.
DUPLICATE COMMON FILE NAME	LFM	A file of the same name as that specified in a COMMON or STAGE request already exists.
DUPLICATE FILE NAME.	LFM	The file specified already exists in the system.
DUPLICATE LINES.	LFM	Lines being dumped during a DMP operation were duplicated and suppressed.
DUPLICATE PROJECT NUMBER.	PROFILE	During a create run, PROFILE detected two or more identical project numbers within one charge number entry. The first project number is retained; all subsequent duplicate numbers are disregarded. All other project numbers are processed normally.
DUPLICATE USER NUMBER.	PROFILE	This message is printed if PROFILE detected two or more identical user numbers in one project number entry, or the user attempts to update the project profile file by adding a user number that already exists under the specified project number. The entire proejct number entry containing the duplicate user numbers is disregarded.

Message	Routine	Description
ECS LOAD ERROR.	3AE	Bad load address from ECS.
EDITING COMPLETE.		Informative message.
lfn EMPTY, AT nnn.	PFM	The file specified on a SAVE request contains no data.
EMPTY CATALOG.	CATLIST	No entries are present in the catalog.
EMPTY SORT INPUT FILE.	MSORT	File lfn specified on the SORT control statement contains no data.
END OF TAPE, fff AT nnn.	CIO	The end of tape was encountered.
ENTRY POINT NOT FOUND.	3AD	The specified entry point could not be found on the overlay file.
EOF ENCOUNTERED BEFORE TERMINATION.	FILES	An end-of-file was encountered on a CONVERT input file before the specified record count was reached.
EOI ENCOUNTERED BEFORE TERMINATION.	FILES	An end-of-information was encountered on a CONVERT input file before the specified record count was reached.
EQ, Ccc-e-uu, vsn, rw, est, Sss, $scon_1$, $scon_2$. EQ, Ccc-Fff, Iii, Bnnnnnn, Lbbbb, Pppppppp. EQ, Ccc, Eec, H000000000, type.	1MT	Three-line message describing a magnetic tape hardware malfunction occurring on a 604, 607, 657, or 659 tape unit. EQ MT for 604, 607, and 657; NT for 659 The first line provides the following information: cc-e-uu — Channel, equipment (tape controller), and physical unit number of tape unit on which error was encountered. vsn — Volume serial number associated with the tape on the specified unit. rw — Read (RD) or write (WR) operation: any operation not involving an actual read or write is listed as a read. est — EST ordinal of the unit on which the tape was written. This is provided only for labeled tapes generated under NOS 1.0; otherwise, the field is blank. ss — Status of the 6681/6684 interface. First digit represents a^{00} where bit $a=2^{11}$ of status; second digit represents bits 2^2-2^0 of status. $scon_1$ — Status of the tape controller. $scon_2$ — Status-2 of the controller, if available. The second line of the message contains: cc — Channel number; the channel number is repeated to allow the analyst to associate this message with the first message if errors are occurring on more than one tape channel at the same time. ff — Software function on which the error occurred. ii — Error iteration; number of times error has been encountered on this unit without successful recovery. nnnnnn — Block number on which error occurred. bbbb — Length of block on which error occurred, in octal bytes. pppppppp — 1MT internal error parameters.

Message	Routine	Description
		The third line of the message contains the following information: **cc** — Channel number; the channel number is repeated to allow the analyst to associate this message with the first and second messages if errors are occurring on more than one tape channel at the same time. **ec** — Octal error code value. **0000000000** — Controller options selected at the time of the error; each two digits is a function code. **type** — Additional description of the error (one of the following): BAD ERASE. — Error detected after an erase was attempted to recover a write error. BLOCK TOO LARGE — Data block was larger than expected. BUSY. — Unit was still busy after 1 second. CHANNEL ILL. — Channel is not accepting function or status requests properly. CON. REJ. — Connect reject; unable to connect to the unit. CON. REJ. OFF. — Connect reject; unable to connect to unit. Unit turned OFF. DENSITY CHANGE. — Either user error where auto select does not match user selection (0-track only), or hardware error where status does not match user selection. FNnn, Pyyyy. — Function nn was rejected by the controller; yyyy is the address in 1MT where the function was initiated. Lbbbb, Bnnnnnn. — The length (bbbb) and block number (nnnnnn) read from trailer bytes in block did not match the actual length or the block number read; given in previous message line. NO EOP. — No end-of-operation detected from unit within 1 second. NOISE. — A noise block was skipped on the tape. NOT READY. — Tape unit dropped ready status. ON THE FLY. — Error was corrected as the data was read. POSITION LOST. — The last good block written cannot be found during write recovery. RECOVERED. — Previously reported error has been successfully recovered. STATUS. — Error type cannot be determined so actual controller status is returned. WRONG PARITY. — Tape was written in parity opposite that being read.
EQ, Ccc-uu, vsn, rw, est, Ss, GSgggg. EQ, Ccc, Dddd...d EQ, Ccc, Fff, Iii, Bnnnnnn, Lbbbb, Pppppppppp. EQ, Ccc, Eec, Hhhhhhhhh, type.	1MT	Four-line message describing a magnetic tape hardware malfunction occurring on a 667 or 669 tape unit. EQ — MT for 667; NT for 669 The first line provides the following information: **cc-uu** — Channel and physical unit number of tape unit on which error was encountered. **vsn** — Volume serial number associated with the tape on the specified unit.

Message	Routine	Description
		rw — Read (RD) or write (WR) operation; any operation not involving an actual read or write is listed as a read. est — EST ordinal of the unit on which the tape was written. This is provided only for labeled tapes generated under NOS 1.0; otherwise, the field is blank. s — Channel status. gggg — General status of magnetic tape unit. The second line of the message contains: cc — Channel number; the channel number is repeated to allow the analyst to associate this message with the first message if errors are occurring on more than one tape channel at the same time. ddd...d — Detailed status of magnetic tape unit. The third line of the message contains: cc — Channel number; repeated to associate this message with the previous messages. ff — Software function on which the error occurred. ii — Error iteration; number of times error has been encountered on this unit without successful recovery. nnnnnn — Block number on which error occurred. bbbb — Length of block on which error occurred, in octal bytes. pppppppp — 1MT internal error parameters. The fourth line of the message contains: cc — Channel number; repeated to associate this message with the previous messages. ec — Octal error code value. hhhhhhhh — Unit format parameters. Refer to Magnetic Tape Subsystem Reference Manual for descriptions of unit format parameter fields. type — Additional description of the error (one of the following): BAD ERASE. — Error detected after an erase was attempted to recover a write error. B.C.RESTART. — Magnetic tape controller firmware restarted. BLOCK TOO LARGE. — Data block was larger than expected. BUSY. — Unit was still busy after 1 second. CHANNEL ILL. — Channel is not accepting function or status requests properly. CON.REJ. — Connect reject; unable to connect to the unit. CON.REJ.OFF. — Connect reject; unable to connect to unit. Unit turned OFF. DENSITY CHANGE. — Either user error where auto select does not match user selection (9-track only), or a hardware error where status does not match user selection. FNnn, Pyyyy. — Function nn was rejected by the controller; yyyy is the address in 1MT where the function was initiated. Lbbbb, Bnnnnnn. — The length (bbbb) and block number (nnnnnn) read from trailer bytes in block did not match the actual length or the block number read; given in previous message line.

Message	Routine	Description
		NO EOP. No end-of-operation detected from unit within 1 second. NOISE. A noise block was skipped on the tape. NOT READY. Tape unit dropped ready status. ON THE FLY. Error was corrected as the data was read. POSITION LOST. The last good block written cannot be found during write recovery. RECOVERED. Previously reported error has been successfully recovered. STATUS. Error type cannot be determined so actual controller status is returned. WRONG PARITY. Tape was written in parity opposite that being read.
EQxx, CHyy Adddd INCOMPLETE TRANSFER.	1IO	An incomplete data transfer was detected by a local batch equipment driver. EQ One of the following equipment types: CP 415 card punch CR 405 card reader LP 501, 505, 512, or 580 line printer LQ 512 line printer LR 580 line printer xx EST ordinal of local batch equipment yy Channel number dddd Octal byte count not transferred
EQxx, CHyy CONTROLLER HUNG BUSY.	1IO	The specified local batch controller did not drop BUSY status. EQ One of the following equipment types: CP 415 card punch CR 405 card reader LP 501, 505, 512, or 580 line printer LQ 512 line printer LR 580 line printer xx EST ordinal of local batch equipment. yy Channel number
EQxx, CHyy Fzzzz FUNCTION TIMEOUT.	1IO	No response (inactive) was received after issuing a function code to the specified local batch equipment. (Converter and equipment status unavailable.) EQ One of the following equipment types: CP 415 card punch CR 405 card reader LP 501, 505, 512, or 580 line printer LQ 512 line printer LR 580 line printer xx EST ordinal of local batch equipment yy Channel number zzzz Function code

Message	Routine	Description
EQxx, CHyy Fzzzz REJ Paaaa, Cbbbb, Ecccc.	1IO	Detected function reject or transmission parity error on the specified local batch equipment. EQ — One of the following equipment types: CP 415 card punch CR 405 card reader LP 501, 505, 512, or 580 line printer LQ 512 line printer LR 580 line printer xx — EST ordinal of local batch equipment yy — Channel number zzzz — Function code aaaa — Driver (1CD) address bbbb — Converter status cccc — Equipment status
EQxx, CHyy RESERVED.	1IO	The specified local batch equipment is reserved and cannot be connected on channel yy. EQ — One of the following equipment types: CP 415 card punch CR 405 card reader LP 501, 505, 512, or 580 line printer LQ 512 line printer LR 580 line printer xx — EST ordinal of local batch equipment yy — Channel number
EQxx, CHyy TURNED OFF.	1IO	The specified local batch equipment was logically turned off (OFF status set in EST). Note: this message is preceded in the error log by a message for the same equipment which specifies the failing condition. EQ — One of the following equipment types: CP 415 card punch CR 405 card reader LP 501, 505, 512, or 580 line printer LQ 512 line printer LR 580 line printer xx — EST ordinal of local batch equipment yy — Channel number
EQxx, DNdn, DIRECT ACCESS FILE ERROR, AT nnn.	PFM	The system sector data for the file does not match the catalog data. xx — EST ordinal of device dn — Device number
EQxx, DNdn, FILE LENGTH ERROR, AT nnn.	PFM	The length of a file does not equal the catalog length. xx — EST ordinal of device dn — Device number

Message	Routine	Description
		The action taken depends on the type of command issued. Command — Action GET — A local file is created with length being the actual length retrieved. SAVE — If file length is longer than TRT specification, file is truncated. REPLACE — Same as for SAVE.
EQxx, DNdn, MASS STORAGE ERROR AT nnn.	PFM	An error was encountered in reading a portion of the permanent file catalog or permit information. xx — EST ordinal of device dn — Device number
EQxx, DNdn, RANDOM INDEX ERROR, AT nnn.	PFM	The random disk address of the permit sector is in error. xx — EST ordinal of device dn — Device number
EQxx, DNdn, REPLACE ERROR, AT nnn.	PFM	The same file was found twice during a catalog search. This error can occur for APPEND or REPLACE commands after a file is found and purged and the catalog search is continued. xx — EST ordinal of device dn — Device number
EQxx, DNdn, TRACK LIMIT, AT nnn.	PFM	No allocatable tracks remain on equipment xx. xx — EST ordinal of device dn — Device number
EQxx FILE ACTIVE - filename.	IMS/MSI	An attempt to initialize a dayfile on device xx was not completed because it was still in use as an active dayfile. xx — EST ordinal of device filename — Name of dayfile
EQxx FILE BUSY - filename.	IMS/MSI	The inactive dayfile has track interlock set indicating it is attached to another control point and thus may not be initialized. xx — EST ordinal of device filename — Name of dayfile
EQxx NO SYSTEM DAYFILES FOUND.	IMS/MSI	An attempt was made to initialize an inactive dayfile on device xx but no dayfile was found. xx — EST ordinal of device
EQxx PROTECTED QUEUES IGNORED.	IMS/MSI	Device xx was initialized for permanent files and protected queue files were found. xx — EST ordinal of device

Message	Routine	Description
EQxx, RM=mmmmmmm, PF=ppppppp, UI=iiiiii.	PFM	Additional line written only in error log after one of the following messages. EQxx, DNdn, DIRECT ACCESS FILE ERROR, AT nnn. EQxx, DNdn, FILE LENGTH ERROR, AT nnn. EQxx, DNdn, MASS STORAGE ERROR, AT nnn. EQxx, DNdn, RANDOM INDEX ERROR, AT nnn. EQxx, DNdn, REPLACE ERROR, AT nnn. EQxx, DNdn, TRACK LIMIT, AT nnn. xx — EST ordinal of device mmmmmmm — Family name ppppppp — Permanent file name iiiiii — User index
filename EQxx TRACKtttt LENGTH ERROR.	QREC	EOI of disk and EOI of TRT are not identical. filename — Name of file in error xx — EST ordinal of device tttt — First track of file in error
EQUIPMENT NOT AVAILABLE	LFM/RESEX	Requested equipment is either in use or does not exist.
ERASE LIMIT, fff AT nnn.	CIO	The system made 20 erasures (10 feet of tape) without being able to successfully write the tape.
ERROR AT LINE xxx		Issued when errors occur while resequencing a BASIC program. The line containing the error is specified by xxx.
ERROR IN ARGUMENTS		One or more of the following conditions was detected. • More than one date was entered. • No options were selected. • The parameter was illegal or could not be recognized. • The TM option was selected but no data was specified. • Both the device number parameter and the packname or auxiliary device parameter were selected; auxiliary devices do not have device numbers.
ERROR IN CHARGE NUMBER.	PROFILE	The charge number is in error. The charge number entry is disregarded and PROFILE skips to the next charge number entry in the input directives.
ERROR IN COMMAND PARAMETERS.		Either no parameters are allowed or an illegal parameter has been encountered.
ERROR IN DATE.	PURGALL	The format of the date (ad, md, or cd) parameter in a PURGALL request was incorrect.
ERROR IN DEVICE NUMBER.	PURGALL	The file residency as specified by the device number parameter was illegal.
ERROR IN DIRECTORY.		Program library does not have a directory record, or has an incorrectly formatted directory record.
ERROR IN FILE ARGUMENTS	FILES	The parameter could not be recognized.

Message	Routine	Description
ERROR IN FILE CATEGORY.	PURGALL/ PFILES	The user specified an illegal file category.
ERROR IN FILE TYPE.	PURGALL	The user specified an illegal file type.
ERROR IN IDENTIFIER.	PROFILE/ MODVAL	PROFILE cannot recognize a directive identifier. The action taken depends upon the position of the erroneous identifier within the entry. • If the error occurs within a project number entry, the entire project number entry is disregarded. • If the error occurs in a directive that appears after a charge number but before the first project number, only the erroneous directive is disregarded. However, if the error occurs on the first PN directive, the entire project number entry is disregarded. • If the error occurs in any PN directive except the first one, it is treated as an error within the preceding project number entry. Both the project number entry for the erroneous project number and the preceding project number entry are disregarded.
ERROR IN LIMITS ARGUMENT.	TCS	Parameters were included on the LIMITS statement.
ERROR IN LINK ARGUMENTS.	LINK	An invalid parameter was used on the LINK control card.
ERROR IN NUMERIC DATA.	PROFILE/ MODVAL	PROFILE detected nonnumeric data or numeric data exceeding the maximum limit for specified control value. The entire project number entry containing the erroneous directive is disregarded.
ERROR IN PASSWOR ARGUMENTS.	TCS	Parameters specified on a PASSWOR control statement were in error.
ERROR IN PROFILE ARGUMENTS	PROFILE	Error on PROFILE control statement.
ERROR IN PROJECT NUMBER.	PROFILE	The data field on the PN directive does not contain 1 to 20 alphanumeric characters. The entire project number entry associated with the erroneous PN entry is disregarded.
ERROR IN TIME.	PURGALL	The format of the time parameter in a PURGALL request was incorrect.
ERROR IN USER NUMBER.	PROFILE	The data field of the UN directive does not contain 1 to 7 alphanumeric characters (or asterisks). The entire project number entry containing the erroneous UN directive is disregarded.
ERROR - FILE(S) NOT PROCESSED.	CHKPT	One or more files were not checkpointed because CHKPT detected address errors.
FET ADDRESS OUT OF RANGE AT nnn.	CIO	FET extends past job's field length.
FILE ALREADY INTERLOCKED.	QFM	The track interlock for the specified file in the DULL word in the MST is already set.
nnnn FILE DEQUEUED Dndn FMxxxxxxx.	QREC	Indicates the number of files that have been dequeued on the specified device. nnnn Number of files dn Device number xxxxxxx Family name
FILE EMPTY.	LFM/SFM/QFM	The file specified was empty.

Message	Routine	Description
FILE ERROR lfn.	CHKPT/ RESTART	An illegal address was detected on file lfn.
FILE NAME CONFLICT.	QREC	Processing was attempted on two files having the same name.
FILE NAME ERROR, AT nnn.	PFM	File name contains illegal characters.
FILE NOT FOUND	LFM/SFM/QFM/QDL	Requested file was not found.
FILE NOT ON MASS STORAGE.	3AD	The specified file does not reside on mass storage.
FILE NOT OVERLAY FORMAT.	LDR	The first record of the file was not an overlay.
FILE TOO LONG, AT nnn.	PFM	The local file specified for a SAVE, REPLACE, or APPEND command exceeds the length allowed, or the direct access file specified for an ATTACH in write, modify, or append mode exceeds the direct access file length limit for which the user is validated.
nnnn FILES ACTIVATED DNdn FMxxxxxxx.	QREC	Indicates the number of files that have been activated on the specified device. nnnn Number of files dn Device number xxxxxxx Family name
nnnn FILES IGNORED DNdn FMxxxxxxx.	QREC	Indicates the number of queued files remaining inactive on the specified device. nnnn Number of files dn Device number xxxxxxx Family name
nnnn FILES PURGED DNdn FMxxxxxxx.	QREC	Indicates the number of queued files which have been purged on the specified device. nnnn Number of files dn Device number xxxxxxx Family name
FL BEYOND USER LIMIT.	1MA	The requested field length exceeds that for which the user is validated.
FL TOO SHORT FOR LIBRARY GENERATION.	LIBGEN	Additional memory is required for LIBGEN.
FL TOO SHORT FOR LOAD.	LINK	The program requires additional field length.
FL TOO SHORT FOR LOADER, NEED xxx.	LINK	The loader requires xxx additional words of memory.
FL TOO SHORT FOR MAP.	LINK	Additional memory is required for the MAP operation.
FL TOO SHORT FOR OVERLAY GENERATION.	LINK	Additional memory is required for overlay generation.
FL TOO SHORT FOR PROGRAM.	3AE	The user's field length is too short for the program.
pfn FOUND, AT nnn.	PFM	The specified permanent file was found.
FORMAT ERROR ON CONTROL CARD.	TCS	An error was detected in the format of the control statement.

Message	Routine	Description
FORMAT ERROR ON OVERLAY DIRECTIVE	LDR	Illegal overlay directive parameter or no arguments found.
ILLEGAL ACCESS TO EXECUTE ONLY FILE.	3AD	The specified file is an execute-only file.
ILLEGAL CHARACTER NUMBER.	FILES	In a copy request, one of the following was detected. • Last character position was less than first character position. • Last character position was greater than 150. • Either first character position or last character position was unrecognizable.
ILLEGAL CHARGE.	CHARGE	The specified charge or project number does not exist or the project number was not assigned to this user.
ILLEGAL CIO REQUEST.	PACK	An attempt was made to pack a nonmass storage file.
ILLEGAL CONTROL CARD.	TCS	One of the following: • The control statement could not be identified. • An invalid parameter was specified or no terminator was detected. • The user attempted to pass too many parameters on the program call statement (such as LGO). • The user submitted a control statement considered illegal because of his validation. For example, if access option 1 (refer to LIMITS control statement) was not set and the user submitted a PASSWOR control statement. • The user submitted a control statement considered illegal for a particular job type or file type. For example, the use of a FAMILY statement in a nonsystem origin job.
ILLEGAL COUNT.	COPYBF/COPYBR/ COPYX	Number of files in copy request was either illegal or zero.
ILLEGAL DEVICE REQUEST, AT nnn.	PFM	The device type (r parameter) specified on a request for an auxiliary device cannot be recognized or does not exist in the system. If the auxiliary device specified by the pn parameter is not the same type as the system default, the r parameter must be included; if not, the message is issued.
ILLEGAL DISPOSE CODE.	CIO	The queue type (q_i) specified on a DISPOSE control statement was unrecognizable.
ILLEGAL EQUIPMENT.	1IO	File is assigned to illegal equipment for the specific request. For example, the file specified in a COMMON request is not on mass storage.
ILLEGAL EXTENSION ON fff AT nnn.	CIO	The user attempted to lengthen a file that could not be extended.
ILLEGAL FILE NAME fff AT nnn.	CIO	The file name does not conform to established rules.

Message	Routine	Description
ILLEGAL FILE TYPE.	LFM/QDL/QFM	The specified file is of a type not allowed in the requested operation. For example, this message would be issued if the file name in a RELEASE request was not a queue type file (input, print, or punch) or if the user attempted to make a non-local file a library file.
pfn ILLEGAL FILE TYPE, AT nnn.	PFM	The user attempted to define a file that was not a local file.
ILLEGAL INPUT FILE.	CIO	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 I/O REQUEST ON FILE fff AT nnn.	CIO	CIO could not recognize the specified function code, or the code was not valid for the type of device to which the file was assigned. The system provides a dump on the FET on file OUTPUT.
ILLEGAL LEVEL NUMBER.	LDR	One of the following: • Assembly error • Level number greater than 77_8 • First overlay not zero level (0,0) overlay
ILLEGAL LOAD ADDRESS.	3AE	The load address is less than 2.
ILLEGAL MAP OPTION, MUST BE -PFSBEXR-	LINK	The MAP control card contained an invalid parameter.
ILLEGAL MODIFICATION OF fff AT nnn.	CIO	Either the user has attempted to shorten a modify-only file or the file cannot be modified at all.
ILLEGAL ORIGIN.	SFM/QFM/QDL	The origin type specified when releasing a local file to a queue was illegal.
ILLEGAL ORIGIN SPECIFIED.		Origin word count error.
ILLEGAL OVERLAY LOADED.	LINK	Overlay processing was already in progress.
ILLEGAL PROFILE INQUIRE.	PROFILE	The user is not allowed to access the control information for the charge number supplied.
ILLEGAL QUEUE TYPE.	QDL	The queue type specified in the request was illegal.
ILLEGAL RECORD TERMINATION.	COPYX	Illegal format on record terminator.
ILLEGAL SORT PARAMETER.		The SORT control statement is in error.
ILLEGAL TERMINAL REQUEST.		A command intended for time-sharing origin jobs only (refer to Time-Sharing Commands, section 4) has been used in a non-time-sharing origin job.
ILLEGAL USER ACCESS.	LFM/QFM/QFSP	User tried to perform an operation for which he was not validated.
ILLEGAL USER ACCESS, AT nnn.	PFM	The user is not validated to create direct access or indirect access files or to access auxiliary devices.

Message	Routine	Description
IMPROPER ACCESSIBILITY.	RESEX	The user did not specify the correct file accessibility on the LABEL statement.
IMPROPER VALIDATION.	TCS	A validation program (one containing a VAL= entry point, such as that used for CHARGE and USER is required before continuing.
INDEX ADDRESS OUT OF RANGE FOR fff AT nnn.	CIO	The random sector address for a random input/output request was equal to or greater than field length.
INPUT FILE IN NORERUN STATUS.	QFM	Informative message.
INPUT FILE IN RERUN STATUS.	QFM	Informative message.
INQUIRY COMPLETE.	PROFILE/ MODVAL	The inquiry was successfully completed.
deckname-INVALID CS, 63 ASSUMED.	TCS	Character set identification for deck deckname was not recognizable. OPLEDIT assumes 63 character set and uses it for the new program library if one is being created.
I/O EXECUTE-ONLY FILE fff AT nnn.	CIO	The user attempted to read, write, or position an execute-only file. RETURN is the only operation allowed for an execute-only file.
I/O SEQUENCE ERROR.	QFM/QDL	Action was requested by a busy file.
I/O SEQUENCE ERROR, AT nnn.	PFM	A request was attempted on a local file that is currently active. This error can occur, for example, if the user creates two FETs for the same file and issues a second request before the first is completed.
I/O SEQUENCE ERROR ON FILE fff AT nnn.	CIO	The user attempted to perform more than one concurrent function on a single file.
IQFT FILE INTERLOCKED.	IMS/MSI	The IQFT file is interlocked by another job and cannot be initialized.
JOB ABORTED, fff AT nnn.	CIO	The job was aborted while a tape operation was pending.
JOB CARD ERROR. (20 characters)	3AA	The job statement on the file being submitted is in error. The first twenty characters of the statement in error follow the message.
JOB EXECUTING.		The job is either executing or has been rolled out for a higher priority job.
JOB IN INPUT QUEUE.		Informative message.
JOB IN NORERUN STATE ON RECOVERY.	1AJ	Identifies a job recovered on level 0 deadstart that was aborted because it was in a no-rerun mode (due to NORERUN control statement or macro).
JOB IN OUTPUT QUEUE.		Informative message.
JOB IN PUNCH QUEUE.		Informative message.
JOB NOT FOUND.		This message normally indicates that the job has been processed and no longer exists in the system. However, it may also be issued if the jobname was entered incorrectly (misspelled).
JOB REPRIEVED.	SFP	The job has been successfully reprieved.

Message	Routine	Description
LABEL CONTENT ERROR, fff AT nnn.	CIO	A block read was the correct size for a label but one or more required fields (such as the label name) were incorrect. The programmer should use the LISTLB control statement to determine the cause of the problem.
LABEL MISSING, fff AT nnn.	CIO	During a read operation, a required label was missing. The programmer should use the LISTLB control statement to determine the cause of the problem.
LABEL PARAMETER CONFLICT ON OPEN, fff AT nnn.	CIO	Label fields did not match on open request. An additional message: FIELD BEGINNING AT nnn NO COMPARE. specifying the decimal character position in HDR1 of the first field that did not compare correctly is also issued.
LBC ARGUMENT ERROR.	CPMEM	The load address, addr, specified on the LBC control statement was not numeric.
LBC FWA .GE. FL.	CPMEM	The load address specified on the LBC control statement was greater than or equal to the user's field length.
LDR ERROR.	LDR	Issued after one of the following errors. OVERLAY NOT FOUND IN LIBRARY. ARG ERROR. FILE NOT OVERLAY FORMAT.
LEVEL NUMBER MISSING	LDR	First or second level number was blank or unspecified.
LEVEL-3 DATA BASE ERROR.	PROFILE	PROFILE detected a format error in the user number/control value entries of the project profile file.
LFM ILLEGAL REQUEST.	LFM	One of the following: • LFM function detected was not recognized as a legal function. • An LFM function was issued without the auto recall bit set.
LIBGEN ARGUMENT ERROR.	LIBGEN	An invalid parameter was used on the LIBGEN control statement.
LIBRARY GENERATION COMPLETE.	LIBGEN	Informative message.
LIBRARY GENERATION FILE EMPTY.	LIBGEN	The file to be processed is empty.
LINE NUMBER LIMIT EXCEEDED.	RESEQ	The line number encountered or required during a resequencing (RESEQ) operation exceeded 99999.
LIST COMPLETE.	PROFILE	The list operation has been successfully completed.
** LOAD ERRORS.	LINK	During the load operation, xx errors were encountered.
LOAD FILE EMPTY.	LINK	No data on the file to be loaded.

Message	Routine	Description
LOADER MISSING.	TCS	Either CALL or LDR= were not found in the library.
LOC ARGUMENT ERROR.	CPMEM	The first word address or last word address parameter specified on the LOC control statement was not numeric.
LOC RANGE ERROR.	CPMEM	Either the first word address was greater than the last word address or the last word address was greater than the user's field length.
-LO-ERROR. MUST BE IN -BEXCR-.	LINK	The LO parameter specifying the map list option for the LINK control card is incorrect.
LPxx,... .		Refer to the EQxx,... series of corresponding messages for full descriptions of messages beginning with LPxx,... .
LQxx,... .		Refer to the EQxx,... series of corresponding messages for full descriptions of messages beginning with LQxx,... .
LRxx,... .		Refer to the EQxx,... series of corresponding messages for full descriptions of messages beginning with LRxx,... .
MASS STORAGE DIRECTORY NOT WRITTEN.		On a GTR control card, user requested that a mass storage directory record be written on a non-mass storage file.
MDxx, Cyy, ec, ann, Stttt, FNqqqq-r. or MDxx, Cyy, ec, ann, Stttt, Ux Cxxxx Stttt.	6MD	An error has been detected on mass storage device xx. The nature of the error is determined by examining each parameter in the message. xx — EST ordinal of 841 disk yy — Channel number ec — Error code; one of the following: PE — Parity error/checkword error AD — Address error ST — Device status error FN — Function reject for any device connected to data channel converter (6681), or function timed out with no response RS — Device reserved NR — Device not ready a — Type of operation; one of the following: R — Read W — Write nn — Retry count; error is considered irrecoverable after following number of retries. PE 10 AD 10 ST 64 FN 10 RS Indefinite NR Indefinite

Message	Routine	Description
		tttt Device status, implies there was an incomplete transfer if status does not indicate an error qqqq Function rejected r Data channel converter (6681) status, if present Ux Physical unit Cxxxx Upper address Physical address Sxxxx Lower address
MEMORY OVERLOW.		Insufficient storage was allowed for an OPLEDIT run.
MISSING OVERLAY FILE SPECIFICATION.	LINK	No overlay file name specified.
MIXED CHARACTER SET OPL.		Records of more than one character set were encountered on the old program library.
M.T. NOT AVAILABLE ON FILE fff AT nnn.	CIO	The magnetic tape executive is not executing.
MT/NT CONFLICT	RESEX	Conflict exists between 7-track and 9-track tape descriptors. For example, a request for a 9-track tape specifies 200-bpi density. This message can also be issued if the device type specified in FET+1 conflicts with the track type specified in FET+8, bit 56. If dt=MT and bit 56 is set, or if dt=NT and bit 56 is not set, the message is issued.
NO CONNECT TIME AVAILABLE.	CHARGE	The user has accumulated the maximum connect time allowed for the specified project number.
NO CPU TIME AVAILABLE.	CHARGE	The user has accumulated the maximum CPU time allowed for the specified project number.
NO DIRECTIVES.		Directive file was empty.
NO EOR FOUND ON ZZZZZDF.		Illegal file format for ZZZZZDF. The FILE control statement (described in the Record Manager Reference Manual) is used to update the file information table (FIT) which is required for files the Record Manager accesses. The system uses information the programmer supplies on the FILE statement to prestore FIT information in file ZZZZZDF.
NO FILE SPECIFIED FOR LOAD.	LINK	The file to be loaded was not specified on the control card.
NO INPUT FILE	PROFILE	No input directives were present.
NO INPUTFILE FOUND.	QFM	No valid input file exists; functions cannot be performed.
NO IQFT FILE FOUND.	IMS/MSI	An attempt to initialize inactive queues on the device was not completed since an IQFT file could not be found.

Message	Routine	Description
NO LINE NUMBER ON SORT FILE.	SORT	A line on the input file to a SORT request is missing a line number or a line exceeded the 150-character limit.
NO LINE TERMINATOR.		A copy operation was attempted on a line longer than 150 characters which did not contain a line terminator.
NON-MATCHING CONVERSION		The conversion mode required for the tape is not the same as that specified on the control statement. This is only a warning message.
NO READ FILE - lfn.	SUBMIT	The specified file cannot be found.
filename NOT DECLARED RANDOM.		An EOF was encountered on the nonrandom file, filename.
NOT ENOUGH DATA IN BUFFER.	QDL	The CIO buffer does not contain enough words to be written into the system sector.
NOT ENOUGH ROOM IN BUFFER.	QDL	There are not enough words available in the buffer to contain the system sector.
lfn NOT FOUND.	RESTART	RESTART was unable to retrieve a file named, but not included, on lfn.
NORERUN/RERUN IGNORED FROM TTY JOBS.	QFM/CONTROL	User entered NORERUN/RERUN from a terminal. The command is ignored.
xxx NOT IN PP LIB.	SFP	PP package xxx was not found in PP libraries.
xxx NOT IN PP LIB. - CALLED BY yyy.	SFP	PP package xxx was not found in the PP libraries and was called by package yyy.
pfn NOT FOUND, AT nnn.	PFM	One of the following: • The specified permanent file could not be found. • The specified user number could not be found. • The user is not allowed to access the specified file. • The user issued an indirect access file command on a direct access file. • The user issued a direct access file command on an indirect access file. If this message occurs in response to the SAVE request, the specified local file is not attached to the control point, is a direct access file, or is an execute-only file.
lfn NOT ON MASS STORAGE, AT nnn.	PFM	The file to be saved is not on mass storage; the first track of the file is not recognizable.
NT.... .		Refer to the EQ.... series of corresponding messages for full description of messages beginning with NT.

Message	Routine	Description
NO WRITE ENABLE, ON fff AT nnn.	CIO	Either the user attempted to write on a tape mounted with no write ring, or no write was allowed because of additional constraints described in an additional message line: LABEL NOT EXPIRED. — The user attempted to write over a label that had not yet expired. WRITE OVER LABEL ILLEGAL. — The user is not allowed to destroy the VOL1 label. 200 BPI WRITE ILLEGAL. — The tape unit (667 or 669) does not support 200-bpi density.
OLDPL ERROR.		Update program library format was bad.
OPERATOR DROP.	3AB	The job was dropped by the operator.
OPLEDIT COMPLETE.	OPLEDIT	Informative message indicating OPLEDIT completion.
OPLEDIT ERRORS.	OPLEDIT	Errors were encountered while modifying a particular deck.
OUTPUT FILE LIMIT	LFM	The total number of files disposed to the output queue by the job has exceeded the limit for which the user is validated.
OUTPUT FILE LIMIT, FILE fff AT nnn.	CIO	During an attempt to close this file, the number of files disposed to output queues by the job has exceeded the limit for which the user is validated.
OVERLAPPING INSERT OR DELETE.		Insertions and deletions affect the same deck.
OVERLAY FILE EMPTY.	3AD	No data appears in the requested file.
OVERLAY FILE NOT FOUND.	3AD	The specified file was not available.
OVERLAY LEVEL NOT (0,0).	LINK	The overlay was not a zero level (0,0) overlay.
OVERLAY NOT FOUND.	3AD/3AE	The specified overlay was not found.
OVERLAY NOT FOUND IN LIBRARY.	LDR	The specified overlay was not found in the system library.
PACK PARAMETER ERROR.	PACK	The PACK control statement contains an error.
PARITY ERROR - RESTARTED FROM kk.	RESTART	Because RESTART detected a parity error in attempting to restart from the specified checkpoint nn, the alternate checkpoint kk was used instead.
PBC ARGUMENT ERROR.	CPMEM	Either the first word address or the last word address specified on a PBC control statement was nonnumeric.

Message	Routine	Description
PBC FWA .GT. LWA.	CPMEM	The first word address was greater than the last word address.
PBC RANGE ERROR.	CPMEM	The last word address parameter specified on a PBC statement was greater than or equal to the user's field length.
PFM ABORTED, AT nnn.	PFM	Error flag detected at PFM control point.
PFM ILLEGAL REQUEST, AT nnn.	PFM	One of the following: • Illegal command code passed to PFM • Illegal permit mode or catalog type specified • CATLIST request has permit specified without a file name • PERMIT command attempted on a public file
PF UTILITY ACTIVE, AT nnn.	PFM	Because a permanent file utility is currently active, the operation was not attempted; the user should retry the operation.
PL ERROR IN DECK dname.		Error encountered in processing deck dname.
POSITION ERROR ON--xxxxxxx.		File xxxxxxx was not repositioned after being checkpointed because CHKPT detected an address error.
POSITION LOST, fff AT nnn.	CIO	During write error recovery, the system could not find the last good block of data, making it impossible to successfully perform error recovery.
PP CALL ERROR.	3AB	The monitor detected an error in a CPU request for PP action.
PROFILA UPDATED.	PROFILE	The project profile file was successfully updated.
PROFILE ABORTED.	PROFILE	Central site operator aborted PROFILE.
PROGRAM FILE EMPTY.	TCS	A load of an empty data file was attempted.
PROGRAM LIBRARY EMPTY.		The old program library contained no data.
PROGRAM NOT FOUND.	EXU	The program to be loaded was not found on the specified library file.
PROGRAM NOT ON MASS STORAGE.	EXU	The program does not reside on a mass storage device.
PROGRAM STOP AT xxxxxx.	3AB	The monitor detected a program stop instruction at address xxxxxx.
PROGRAM TOO LONG	EXU	The program does not fit in the available storage.

Message	Routine	Description
PROTECTED FILE		The user has attempted to release a locked file.
PRU LIMIT, AT nnn.	PFM	The job's mass storage PRU limit has been exceeded during preparation of a local copy of an indirect access file.
PRU LIMIT, FILE fff AT nnn.	CIO	The job's mass storage PRU limit was exceeded during an attempt to write or extend this file.
PRUS REQUESTED UNAVAILABLE.	3PF	The number of PRUs requested is not available.
PRUS REQUESTED NOT AVAILABLE, AT nnn.	PFM	The number of PRUs specified via the s parameter on the DEFINE request is not available.
QDL ARG. ERROR.	QDL	Parameter addresses passed to QDL were not within the user's field length.
QDL ILLEGAL REQUEST.	QDL	One of the following: • The job was not of system origin and the user did not have system origin privileges. • Job did not have SSJ= entry point. • Auto recall bit was not set. • The RA+1 call to QDL contained an illegal function code.
QFM ARGUMENT ERROR.	QFM	One of the following: • Address is outside field length • Address is equal to one • Origin code is out of range • ID code is out of range
QFM EOI BAD ON ATTACHED FILE.	QFM	The EOI sector cannot be found on the specified file.
QFM FILE ALREADY ATTACHED.	QFM	The specified file is already attached to the control point.
QFM FILE EMPTY.	QFM	The submitted file has not been used.
QFM - FILE IGNORED filename.	QFM	The file was ignored because it had an illegal origin or type code. It could indicate a bad IQFT file.
QFM FILE NAME ERROR.	QFM	The lfn specified does not check as a valid file name.
QFM FILE NOT FOUND.	QFM	The submitted file could not be found.
QFM FILE NOT ON MASS STORAGE.	QFM	The submitted file does not reside on mass storage.
QFM ILLEGAL EQUIPMENT.	QFM	The equipment specified in FET+7 either is not mass storage or is not in the range of the EST.
QFM ILLEGAL FILE TYPE.	QFM	The submitted file is not a local file.
QFM ILLEGAL ID CODE.	QFM	The ID code is out of range.
QFM ILLEGAL ORIGIN TYPE.	QFM	The origin type for the submitted file is not batch or Export/Import.

Message	Routine	Description
QFM ILLEGAL REQUEST.	QFM	One of the following: • Specified function illegal or undefined • Job did not have SSJ= entry point • Auto recall bit was not set
QFM INTERLOCK ERROR.	QFM	Track interlock could not be set due to a conflict.
QFM TRACK MISMATCH.	QFM	The file about to be purged is not the same file that was previously attached. The first track in the FST does not equal the one from the DULL word.
QFM UNABLE TO INTERLOCK MST.	QFM	Informative message.
QREC/QLIST ABORTED. QUEUE STATUS INDEFINITE.	QREC	This pair of messages occurs if QREC aborts for any reason.
QUEUE FILE UTILITY COMPLETE.	QFSP	Informative message.
QUEUED FILES LOST.	QREC	Files which process error conditions were not requeued. This error should never occur but may if QREC was aborted and could not modify its files correctly. A level 0 deadstart will recover the queues.
QUEUES UNRECOVERABLE THIS DEVICE.	QREC	This message is issued in conjunction with the mass storage error message DNdn FMxxxxxxx MS ERROR. Refer to that message for the appropriate device information.
RANDOM ADDRESS NOT ON FILE fff AT nnn.	CIO	The random address specified was not within the bounds of the file. The system provides a dump of the FET on file OUTPUT.
READ AFTER WRITE, fff AT nnn.	CIO	The user attempted to read a tape on which the last operation was a write.
READ FILE BUSY - lfn	SUBMIT	The read file is found to be busy (direct access file only).
nnnnn RECORDS CONVERTED.		Informative message indicating number of records (nnnnn) converted from one character set to another.
n RECORD(S) NOT REPLACED.		Informative message; the job is aborted unless the D option was specified.
RECORD SIZE EXCEEDS 500.		The maximum line length for a record to be converted (500 characters) was exceeded.
RECORD TOO LONG.	CPMEM	The record is too long for available memory. Available memory is filled and the excess data is skipped. In response to a WBR request, the record length parameter was greater than or equal to the user's field length.
REPRIEVE IMPOSSIBLE - BAD CHECKSUM	SFP	Post-recovery checksum does not match pre-recovery checksum.
REQUEST UNDEFINED ON DEVICE fff AT nnn.	CIO	The specified function cannot be performed on the device on which the file resides. The system provides a dump on the FET on file OUTPUT.
RERUN NOT POSSIBLE.	IDS	Operator attempted to rerun a job that is in no-rerun mode.
RESEX DETECTED ERROR.	LFM	The resource executive (RESEX) detected an error.

Message	Routine	Description
RESEX FAILURE, AT nnn.	PFM	The resource executive has detected a fatal error.
RESOURCE DEMAND ERROR.	RESEX	The user attempted to decrease the number of scheduled units to less than the number of currently assigned units, or increase the number of scheduled units to a point where a deadlock would occur.
RESOURCE TYPE ERROR.	RESEX	The user specified an illegal resource type.
jobname RESTARTED FROM yy/mm/dd. hh.mm.ss.	RESTART	The checkpointed job identified by jobname was restarted from the checkpoint taken on the specified data and time. This message is issued whenever a checkpoint job is restarted.
ROLLIN FILE BAD.	1RI	An illegal format was detected in the rollin file.
SFM ARGUMENT ERROR.	SFM	The argument passed to SFM was out of bounds or the FET specified did not specify a buffer of at least 100_8 words.
SFM DAYFILE BUSY.	SFM	Action was requested on a busy dayfile.
SFM ILLEGAL DAYFILE CODE.	SFM	The dayfile code passed in the FET was not within range.
SFM ILLEGAL REQUEST.	SFM	The requested function or origin type specified in the function call was not recognizable, or SFM request was made and the auto recall bit was not set.
SFM TRACK INTERLOCK ERROR.	SFM	Track was either interlocked when it should not have been or not interlocked when it should have been.
SMF UNABLE TO INTERLOCK DEVICE.	SFM	SFM request was not performed because the selected device could not be interlocked.
SFP CALL ERROR.	SFP	SFP was not loaded by default.
SFP.xxx ILLEGAL ORIGIN CODE.	SFP	Function illegal for user's job origin.
SFP/xxx PARAMETER ERROR.	SFP	Parameter address outside FL.
SPCW CALL ERROR.	1AJ	A DMP= type call was made and the program called is either not in the CLG or does not have a DMP= entry point defined.
SPECIAL REQUEST PROCESSING ERROR.	SFP	The SPCW word was busy.
STATUS ERROR, fff AT nnn.	CIO	An irrecoverable error was encountered. A second message line describes the error in more detail. CRC ERROR. — An error was detected in cyclic redundancy character. DATA TIMING PROBLEMS. — Hardware malfunctions. Another unit should be tried. FILL STATUS ILLEGAL. — The system has detected an odd number of frame, a condition which is illegal for the data format of the tape being read. FLAG BIT ERROR. — The tape being read with ASCII conversion contains characters not included in the 128-character set. MEMORY PARITY ERROR. — A parity error was detected in the conversion memory of the tape controller.

Message	Routine	Description
		MULTI-TRACK PHASE ERROR. Multiple tracks were found to be in error at 1600 cpi, making recovery impossible. PARITY ERROR. The tape could not be read/written correctly. UNIT HAS MOTION PROBLEMS. The tape unit cannot properly write the tape. The user should resubmit his job, using a different tape unit.
SYSTEM ABORT.	1MA	PP program detected an irrecoverable system error.
SYSTEM SECTOR ERROR.	QFM/QDL	An error occurred while reading the system sector.
TABLE OVERFLOW. JOB ABORTED.		Resubmit job with increased field length.
TAPE BLOCK DEFINITION ERROR.	RESEX	The user attempted to define data block size via the FC or C keyword or noise block size via the NS keyword in such a manner that the system is unable to correctly define the size of the data block. The omission of the FC or C parameter on a control statement where it is required also causes this message to be issued.
TAPE FORMAT PROBABLY WRONG.	CIO	This message is issued in addition to one of the following messages: BLOCK SEQUENCE ERROR, fff AT nnn. BLOCK TOO LARGE, fff AT nnn. WRONG PARITY, fff AT nnn. if one of these error conditions occurs on the first block.
TIME LIMIT.	3AB	The monitor detected that the time limit for the job has expired.
TIME LIMIT EXCEEDED.	CPM	Job used more CPU time (in seconds) than allowed.
TL NOT VALIDATED.	ACCFAM	The time limits specified on the job statement exceed that for which the user is validated.
TOO MANY ARGUMENTS. or	TCS	The number of arguments on the control statement exceeds that allowed by the program.
TOO MANY ARGUMENTS.	COPYB	More arguments were specified on a copy request than are allowed on that control card.
TOO MANY DEFERRED BATCH JOBS.	QFM	The user is not validated for this function or he has more jobs in the system than he is allowed. (All jobs in batch queues and E/I queues are counted.) The count is ignored if the job is of system origin or the user is validated for system privileges and DEBUG mode is set by the operator.
TRACK ALREADY ASSIGNED	QFM	The track byte for the IQFT file in the DULL word in the MST is already assigned.
TRACK LIMIT, FILE fff AT nnn.	CIO	The device on which the file resides is full.
UNIDENTIFIED LOADER INPUT.	LINK	The file to be loaded is not in correct format.
UNABLE TO READ IQFT FILE.	IMS/MSI	An attempt to initialize inactive queues failed because the IQFT file could not be read.

Message	Routine	Description
UNIDENTIFIED PROGRAM FORMAT.	3AE	The file the user requested to be loaded was not in a recognizable format.
UNRECOVERABLE MS ERROR.	QFM	An irrecoverable mass storage error was detected during an I/O operation.
UPMOD COMPLETE.		Informative message indicating UPMOD completion.
USER NOT VALID TO UPDATE.	PROFILE	A user who attempted an update was not the master user for the specified charge number entry. That entry is ignored and PROFILE skips to the next charge number entry in lfn_1.
VERIFY ERRORS.	VERIFY	Errors were encountered during VERIFY routine.
WBR ARGUMENT ERROR.	CPMEM	The record length parameter specified on a WBR statement was nonnumeric.
WRITE ON READ-ONLY FILE fff AT nnn.	CIO	Either the user attempted to write on a file with write interlock or the direct access file was not attached in write mode.
WRITE OVER LABEL ILLEGAL ON fff AT nnn.	CIO	The user is not allowed to destroy the VOL1 label.
WRONG PARITY, fff AT nnn.	CIO	A 7-track tape is being read in opposite parity from which it was written.
25555 FIELD LENGTH INCREASE.		The job field length was too small for LIBEDIT. Field length was increased to 26K.

LFM ERROR CODES

The following error codes are returned to the error code field of the FET word 0, bits 10 through 13 in response to LFM requests.

Error Codes	Description
1	File not found
2	File name error
3	Illegal file type
4	File empty
5	Waiting for common file
6	Duplicate common file name
7	Illegal equipment
10	Equipment not available
11	Duplicate file name
12	Illegal user access
13	Illegal user number
14	Illegal ID code
15	Resource executive (RESEX) detected an error
16	I/O sequence error
17	Output file limit
20	Local file limit

PFM ERROR CODES

The following error codes are returned to the error code field of the FET word 0, bits 10 through 17 in response to PFM requests.

Error Codes	Description
1	The specified direct access file is attached in the opposite mode.
2	One of the following: • The specified permanent file could not be found. • The specified account number could not be found. • The user is not allowed to access the specified file. • The user issued an indirect access file command on a direct access file. • The user issued a direct access file command on an indirect access file. If this message occurs in response to the SAVE macro, the specified local file is not attached to the control point, is a direct access file, or is an execute-only file.
3	The file specified on a SAVE macro contains no data.
4	The file to be saved is not on mass storage; the first track of the file is not recognizable.

Error Codes	Description
5	The user has already saved or defined a file with the name specified.
6	The user attempted to define a file that was not a local file.
7	File name contains illegal characters.
10	The user is not validated to create direct access or indirect access files or to access auxiliary devices.
11	The device type (r parameter) specified on a request for an auxiliary device cannot be recognized or does not exist in the system. If the auxiliary device specified by the pn parameter is not the same type as the system default, the r parameter must be included; if not, this message is issued.
12	The local file specified for a SAVE, REPLACE, or APPEND command exceeds the length allowed, or the direct access file specified for an ATTACH in write, modify, or append mode exceeds the direct access file length limit for which the user is validated.
13	One of the following: • Illegal command code passed to PFM • Illegal permit mode or catalog type specified • CATLIST request has permit specified without a file name • PERMIT command attempted on a library file
14	Access to the permanent file device requested is not possible.
15	The device on which the file resides may not contain direct access files because: 1. The device is not specified as a direct access device in the catalog descriptor table. 2. The device is not specified as ON and initialized in the catalog descriptor table. 3. The device is a dedicated indirect access permanent file device.
16	Because a permanent file utility is currently active, the operation was not attempted; the user should retry the operation.
17	An error occurred in a read operation during a file transfer.
20	The number of files in the user's catalog exceeds the limit (refer to LIMITS control card, section 6).
21	The cumulative size of the indirect access files in the user's catalog exceeds the limit (refer to LIMITS control card, section 6).

Error Codes	Description
22	The number of PRUs specified via the s parameter on the DEFINE macro is not available.
23	A request was attempted on a local file that is currently active. This error can occur, for example, if the user creates two FETs for the same file and issues a second request before the first is completed.
24	The job's local file limit has been exceeded by an attempt to GET or ATTACH the file.
25	The job's mass storage PRU limit has been exceeded during preparation of a local copy of an indirect access file.
30	The resource executive has detected a fatal error.
31	No allocatable tracks remain on equipment xx, where xx is the EST ordinal.
32	The length of a file does not equal the catalog length; the action taken depends on the type of command issued.

Command	Action
GET	A local file is created with length being the actual length retrieved.
SAVE	If file length is longer than TRT specification, file is truncated.
REPLACE	Same as for SAVE.

33	Permit random address error.
34	The system sector data for the file does not match the catalog data.
35	The same file was found twice during a catalog search. This error can occur for APPEND or REPLACE commands after a file is found and purged and the catalog search is continued.
36	Error flag detected at PFM control point.
37	An error was encountered in reading a portion of the permanent file catalog or permit information.

Table 1-B-1 specifies the action PFM takes if it detects an error while reading mass storage. The symbols used in the table designate the type of response PFM makes and are defined as follows:

Symbol	Description	Code
DTE	DATA TRANSFER ERROR.	17
EOI	Processing continues as if an EOI was encountered.	
MSE	MASS STORAGE ERROR.	37
FNF	pfn NOT FOUND.	2
DAF	DIRECT ACCESS FILE ERROR.	34
FLE	FILE LENGTH ERROR.	32

TABLE 1-B-1. PERMANENT FILE ERROR CONDITIONS

Activity	Command								
	SAVE	GET	PURGE	CATLIST	PERMIT	REPLACE	APPEND	DEFINE	ATTACH
Device-to-device transfer (valid sector)	DTE	DTE				DTE	DTE		
Device-to-device transfer (no valid sector)	EOI†	EOI†				EOI†	EOI†		
Reading PF catalog	MSE	FNF	FNF	EOI	FNF	MSE	FNF††	MSE	FNF
Device-to-device transfer of original file (valid sector)							DTE		
Device-to-device transfer of original file (no valid sector)							EOI†		
Reading a system sector			DAF					DAF	DAF
Reading permit information		FNF	FNF	EOI		FNF	FNF		FNF
Reading permit information for update		MSE			MSE	MSE	MSE		MSE

† Unless the error occurred while the last sector was being read, a FILE LENGTH ERROR message is issued.
†† If the error occurred on a reentrant search of the PF catalog, a MASS STORAGE ERROR message is issued.

LIBEDIT C

LIBEDIT is a binary record management program that is used to:

- Create and maintain a library file
- Copy records to a library file
- Delete records from a library file
- Replace records on a library file

Binary logical records are the basic unit manipulated. LIBEDIT manipulates the records of the old library file and optional replacement files. Records for replacement can be on one or more secondary files. Replacement is the implicit mode of a LIBEDIT run. Additions and no-replacements must be explicitly requested.

LIBEDIT manipulates the following record types.

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

Formats are further described in appendix G, volume 2.

LIBEDIT is called from the operating system library using a control card. The record managing process is controlled by LIBEDIT directives.

LIBEDIT executes in two phases. During the first phase, it reads directives and replacement records. It groups directives by type and file and groups corrections when several insertions take place relative to the same record.

During the second phase, LIBEDIT performs modifications and generates the new library. If LIBEDIT cannot process the specified combination of directives and the D option (refer to the following control card description) was not specified, LIBEDIT lists the conflicting directives (or a simulated form of the directives), issues an error message, and aborts the job. If the D option was specified, LIBEDIT continues processing the directives.

CONTROL CARD FORMAT

The following control cards call the LIBEDIT program to be loaded and executed. Parameters specify mode and files.

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

The optional parameters, p_i, can be in any order within the parentheses. Generally, a parameter can be omitted or can be in one of the following forms.

a (C, R, and V only)

a=lfn

a=0

a is one of the following options: I, P, N, L, LO, B, C, R, and V. lfn is the 1- to 7-alphanumeric character file name. LIBEDIT accepts only one instance of any parameter.

Option	Description
I=lfn	Directives comprise the next record on file lfn
I=0	No directive input
I omitted	Directives are on file INPUT
P=lfn	File lfn contains the old program library
P=0	No old program library file
P omitted	Old program library is on file OLD
N=lfn	New program library will be written on file lfn
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	List output on file lfn
LO omitted	List output on file OUTPUT
B=lfn	Use file lfn 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

LIBEDIT DIRECTIVES

Directives comprise a program record on file INPUT or on the file specified through the I mode parameter on the LIBEDIT control card. Directives control the record management process. A directive begins with an asterisk in column 1 followed immediately by the card identifier. The card identifier is delimited by a comma and/or one or more spaces. Parameters are delimited by -, a blank, an end-of-line, or a comma.

Card parameters have no embedded blanks. If a directive does not begin with an asterisk and a card identifier, LIBEDIT assumes the operation is a continuation of the last directive operation. If the card was not preceded by a directive, the operation is assumed to be:

*BEFORE *, gid_1, gid_2, . . . , gid_n.

Note, however, that gid entries cannot be split between cards. For example, the cards

```
*B,OVL/P1,OVL/P2,...,OVL/P
N
```

do not constitute a valid directive. The last entry would not be processed as OVL/PN. On the other hand, the cards

```
*B,OVL/P1,OVL/P2
OVL/P3
OVL/PN
0
TEXT/T1
```

do constitute a valid directive and would be processed in the same manner as

*B,OVL/P1,OVL/P2,OVL/P3,OVL/PN,0,TEXT/T1

Directives are not required. If they are not provided, LIBEDIT replaces the records of the old program library file that have the same name and type as the records on the correction file, and LIBEDIT writes the new library.

Parameters common to many of the correction directives are the reference record identifier (rid) and the group record identifier (gid).

rid — The rid parameter specifies a reference point for a correction. It can be in one of the following forms.

type/rname	Reference record is of the specified type
rname	Reference record is the implied type (refer to type)
*	Reference point is an end-of-file mark (*BEFORE card only)

gid — One or more gid parameters on a directive indicate records or groups of records to be inserted, deleted, or replaced. A gid can be in one of the following forms.

type/rname	Single record of the specified type
$type_1$/$rname_1$-$type_2$/$rname_2$	Group of records beginning with rname of $type_1$ and ending with $rname_2$ of $type_2$. Types are specified or implied.

rname — Record identifier can be one of the following.

rname	Name of record

	*	If used for $rname_1$ on an INSERT, AFTER, BEFORE, or IGNORE, an * indicates that all records on the library of the specified or implied type are to be inserted or ignored. If used for $rname_2$ on INSERT, AFTER, BEFORE, or IGNORE, an * indicates that all records of $type_1$, starting with $rname_1$, are to be inserted or ignored.
	0	Indicates that a zero-length record is to be inserted.
type		Identifies the type of the named record. When type is absent from a rid or gid parameter, LIBEDIT uses the type most recently specified on a directive. For valid types, refer to the description of the TYPE directive.

LIBEDIT recognizes the following directives.

Directive	Definition
*ADD	Adds records at end of library.
BEFORE or *B	Inserts records before the named record.
*BUILD	Builds an index at end of new file.
*COMMENT	Adds comment to prefix table.
*COPY	Copies new file to old at end of editing.
*DATE	Adds date and comment to prefix table.
*DELETE or *D	Deletes specified records.
*FILE	Declares additional correction file.
*IGNORE	Ignores records when reading correction file.
*INSERT or *I or *AFTER or *A	Inserts records from correction file after named record.
*NOREP	Does not automatically replace records from named file.
*RENAME	Renames record.
*REPLACE	Replaces records on old file with records from correction file. Optionally declares current correction file as no-replace.
*REWIND	Designates file to be rewound before and after editing.
*TYPE or NAME	Sets type of library to be used for default.

FILE

The directive format is:

*FILE lfn

lfn Name of the additional replacement file; subject to operating system restrictions on file names. If lfn is an *, LIBEDIT uses the replacement file specified by the LIBEDIT control card or the default file (LGO), if none is specified.

The FILE directive declares a secondary file as an additional file that contains replacement records. LIBEDIT directives following a FILE card specify records on the declared replacement file.

REWIND

The directive format is:

*REWIND lfn

lfn Name of file to be rewound

LIBEDIT rewinds the specified file before and after editing.

TYPE OR NAME

The formats for the directives are:

*TYPE type
*NAME type

type Specifies default type of internal record format:

REL	Relocatable CPU program
OVL	CPU overlay program
ABS	Multiple entry point overlay
PP	6000 series format peripheral processor unit program
PPU	7600 format peripheral processor unit program
OPL	Modify old program library deck
OPLC	Modify old program library common deck
OPLD	Modify old program library directories
ULIB	User library; begins with a ULIB type record and terminates with OPLD type record
COS	Chippewa format CPU program
TEXT	Unrecognizable as a program

Any explicit use of a type or a rid or gid parameter resets the default value to the new type.

With the TYPE (or NAME) directive, the user specifies the type of record to which subsequent LIBEDIT directives refer. A type specification is in effect until the next TYPE (or NAME) directive is supplied or until a type is explicitly declared on another directive. If no TYPE or NAME directive is supplied or no explicit type is used, the type is TEXT.

For example,

```
*BEFORE *, JANE-*
*DELETE HENRY-IDA
*INSERT GEORGE,MARY
*TYPE COS
```

is equivalent to

```
*BEFORE *, COS/JANE-*
*DELETE COS/HENRY-COS/IDA
*INSERT COS/GEORGE,COS/MARY
```

INSERT OR AFTER

The formats for the directives are:

*INSERT rid, gid_1, gid_2, . . . , gid_n or *I rid, gid_1, gid_2, . . . , gid_n

*AFTER rid, gid_1, gid_2, . . . , gid_n or *A rid, gid_1, gid_2, . . . , gid_n

rid	Identifies the record on the old library file after which the specified records or groups of records are to be inserted
gid_1	Identifies the records or groups of records from the replacement file to be inserted after rid

An INSERT or AFTER directive directs LIBEDIT to insert records or groups of records from the current replacement file after the specified old library record for transcription to the new library file. The current replacement file is the most recent file specified by a FILE directive or by the LIBEDIT control card. Insertion of records causes automatic deletion of the old records having the same names and types from the old library file.

An example of the use of this directive is:

```
*INSERT OPL/LEA,TEXT/OSCAR-*
```

These cards direct LIBEDIT to insert, after the OPL deck LEA on the old library file, all TEXT records from OSCAR until an end-of-file mark is encountered. If any of these TEXT records have the same name as a TEXT record that is already on the old library file, the old TEXT record is not transcribed to the new library file.

BEFORE

The directive formats are:

*BEFORE rid,gid_1,gid_2,...,gid_n or *B rid,gid_1,gid_2,...,gid_n

rid	Identifies the record on the old library file before which the specified records are to be inserted. On the form omitting the directive name, rid is assumed to be * (that is, insert before end-of-file).
gid_i	Identifies records or groups of records from the replacement file to be inserted before rid.

A BEFORE directive causes LIBEDIT to insert records or groups of records from the current replacement file before the specified old library record for transcription to the new library file. The current replacement file is the most recent replacement file specified by a FILE directive or by the LIBEDIT control card. Insertion of records causes automatic deletion of the old records having the same names and types from the old library file.

DELETE

The directive formats are:

*DELETE gid_1,gid_2,...,gid_n or *D gid_1,gid_2,...,gid_n

gid_i	Identifies records or groups of records to be deleted from the old library file. An asterisk cannot be used.

The DELETE directive causes LIBEDIT to suppress copying of the specified records from the old library file to the new library file.

An example of the use of this directive is:

```
*DELETE PPU/LAD-REL/RUN
```

This card directs LIBEDIT to delete records starting with 7600 PPU program LAD through relocatable CPU program RUN.

IGNORE

The directive format is:

*IGNORE gid_1,gid_2,...,gid_n

gid_i	Identifies records or groups of records from the replacement file to be ignored.

The IGNORE directive causes LIBEDIT to ignore a record or group of records on the current replacement file during record processing.

An example of the use of this directive is:

```
    *IGNORE FRAN-*
*FILE WOMAN
```

LIBEDIT ignores program FRAN of the current type and all following programs of the current type until an end-of-file mark on the replacement file WOMAN is encountered.

ADD

The directive format is:

*ADD lib, gid_1, gid_2, . . . , gid_n

lib — Specifies that the library is to be added to the old program library file before the zero-length record for the old library file indicated. A library cannot be added if there is no zero-length record.

LIB1 to LIB63 — Libraries 1 through 63 on the old program library file.

gid_i — Identifies records or groups of records to be added to the specified library.

The ADD directive causes LIBEDIT to append records to the specified library for transcription to the new library. Two libraries are separated by a zero-length record on the new library file.

NOTE

Directories are determined from file OLD; adding a zero-length record does not change the directory of the library being added.

Figure 1-C-1 illustrates where records are inserted with the ADD directive.

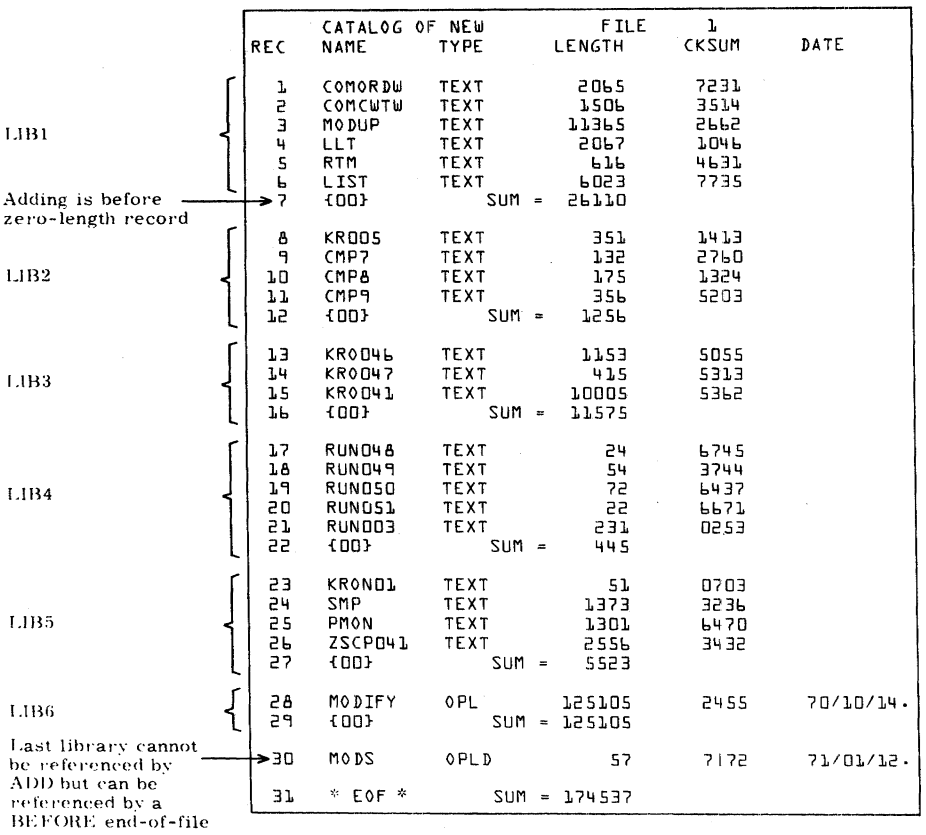


Figure 1-C-1. Adding to the Old Program Library

BUILD

The directive format is:

*BUILD dname

dname	Name of directory record.

The BUILD card requests LIBEDIT to construct and append a directory record in Modify format to the new library file. If the old library file has such a directory, LIBEDIT automatically generates a new directory deck without an explicit BUILD request.†

COMMENT

The directive format is:

*COMMENT rid comment

rid	Name of the record on the replacement or old library file.
comment	A string of up to 40_{10} characters that is suitable as a comment. Additional characters are truncated.

The COMMENT card adds a comment to the prefix (77) table for a program on a replacement file or the old library file. If the program previously did not have a prefix table, LIBEDIT generates one that includes the date and the comment.

DATE

The directive format is:

*DATE rid comment

rid	Name of the record on the replacement or old library file.
comment	A string of up to 40_{10} characters that is suitable as a comment. Additional characters are truncated.

The DATE card adds the current date and the specified comment to the prefix (77) table for a program on a replacement file or the old library file.

NOREP

The directive format is:

*NOREP $lfn_1, lfn_2, \ldots, lfn_n$

The NOREP card declares the specified replacement files to be no-replace files. LIBEDIT does not replace all records of the old library file with records on the no-replace file having identical names, but selectively replaces records from a no-replace file according to REPLACE, INSERT, and BEFORE directives.

---

† BUILD can also be used to change the directory name.

RENAME

The directive format is:

*RENAME rid,name

rid	Name of the record on the replacement or old library file to be renamed.
name	New name of the record (1 to 7 characters).

The RENAME card assigns a new name to a record on the old library or the current replacement file for transcription to the new library file. If the renamed record is referenced by another correction card in the same run, the old name should still be used.

REPLACE

The directive format is:

*REPLACE $gid_1, gid_2, \ldots, gid_n$

gid_i	Name of the record or record group from the replacement file to replace on the old library file.

The REPLACE card directs LIBEDIT to selectively replace records on the old library file with records of the same name from a current replacement file that has been declared a no-replace file (refer to the NOREP card description). Thus, the user can selectively replace records by using the NOREP and REPLACE directives, or he can selectively not replace records by using the IGNORE directive according to the circumstances.

An example of the use of this directive follows: A user has a replacement file named FRUIT containing records APPLE, CHERRY, GRAPE, and ORANGE. Records having the same names are on the old library file. The user wishes to retain records APPLE and CHERRY but replace records GRAPE and ORANGE.

The following two sequences of directives produce the same results.

```
*REPLACE GRAPE-ORANGE
*NOREP FRUIT
*FILE FRUIT
```

```
*IGNORE APPLE-CHERRY
*FILE FRUIT
```

COPY

The directive format is:

*COPY

The COPY card directs LIBEDIT to copy the new library file to the old library file after it has processed all correction cards.

LIBEDIT EXAMPLES

The following two examples illustrate the use of LIBEDIT.

Job 1:

The following steps describe the operations performed by job 1.

1. The job requests an old deadstart tape, OLDSYS.
2. The job modifies decks CPM and SYSEDIT (from OPL) with the changes from the input file, assembles the two decks, and writes the new binary code on file LGO. The MODIFY control card assumes that file OPL is a read-only common file. If, however, OPL is a direct access permanent file, the user must include an ATTACH control card before issuing the MODIFY request.
3. LIBEDIT reads the new binary code from LGO, the old binary code from OLDSYS, and writes a copy of OLDSYS to file NEW with the two decks (CPM and SYSEDIT) replaced with the new code.
4. The job requests a new deadstart tape.
5. File NEW is then copied to the new deadstart tape, NEWSYS, and the copy is verified.

```
JOB1(CM60000,T100)
USER(USERNUM,PASSWRD,FAM1)
REQUEST(OLDSYS)
MODIFY(Q)
LIBEDIT(P=OLDSYS,I=0)
UNLOAD(OLDSYS)
RETURN(OLDSYS)
REQUEST(NEWSYS)
REWIND(NEW,NEWSYS)
COPY(NEW,NEWSYS,V)
UNLOAD(NEWSYS)
RETURN(NEWSYS)
-EOR-
*IDENT CPM****
*/      ****PREVENT TIME LIMIT ON TELEX JOBS
*/      WHEN SETTING TIME LIMIT
*DECK CPM
*I,265
            UJN     STL5
*I,266
 STL5       BSS     0
*IDENT SYSED**
*/      ****FIX ERROR THAT CAUSES CERTAIN
*/      LIBDECK CARDS TO BE IGNORED.
*DECK SYSEDIT
*I,1360
 SL52.1     BSS     0
*D,1379
            SA2     WC
            SB3     X2
            LT      B2,B3,5LS1   IF NOT END OF BUFFER

*EDIT CPM,SYSEDIT
-EOI-
```

Job 2:

The following steps describe the operations performed by job 2.

1. The job copies the source decks for VERIFY and VFYLIB from file INPUT to file VERS. (Note: The 6/7/9 card following VFYLIB makes the COPYBF possible.)
2. It creates the decks VERIFY and VFYLIB and writes them in program library format to file NPL. It then assembles the decks and writes the binary code to file LGO.
3. LIBEDIT reads the program library code from file NPL and writes a copy of the old program library OPL to file N. At the same time, it inserts the code for VERIFY and VFYLIB immediately following the program UPMOD (this assumes that VERIFY and VFYLIB are not already on file OPL).
4. It requests a new program library tape.
5. File N is copied to tape file NEWPL and verified.
6. It requests an old deadstart tape OLDSYS.
7. LIBEDIT reads the binary code for VERIFY and VFYLIB from file LGO and writes a copy of the old system object programs from file OLDSYS to file NEW. Meanwhile, the code for VERIFY and VFYLIB is inserted immediately before the fourth zero-length record on file OLDSYS (this assumes that VERIFY and VFYLIB are not already on file OLDSYS).
8. It requests a new deadstart tape file NEWSYS.
9. File NEW is then copied to tape file NEWSYS and verified.

```
JOB2(CM60000,T7777)
USER(USERNUM,PASSWRD,FAM1)
COPYBF(INPUT,VERS)
COMMON(OPL)
MODIFY(Q,N,CL)
LIBEDIT(P=OPL,B=NPL,N=N)
REQUEST(NEWPL)
REWIND(N,NEWPL)
COPY(N,NEWPL,V)
UNLOAD(NEWPL)
RETURN(NEWPL)
REQUEST(OLDSYS)
LIBEDIT(P=OLDSYS)
UNLOAD(OLDSYS)
RETURN(OLDSYS)
REQUEST(NEWSYS)
REWIND(NEW,NEWSYS)
COPY(NEW,NEWSYS,V)
UNLOAD(NEWSYS)
RETURN(NEWSYS)
-EOR-
VERIFY
    .
    .
Insert VERIFY program source deck here.
    .
    .
-EOR-
VFYLIB
    .
    .
Insert VFYLIB program source deck here.
    .
    .
-EOF-
*REWIND VERS
*CREATE VERS
*EDIT    VERIFY,VFYLIB
-EOR-
*TYPE    OPL
*INSERT  UPMOD,VERIFY,VFYLIB
-EOR-
*ADD     LIB4,ABS/*,OVL/*
-EOI-
```

LIBEDIT issues dayfile messages to report program completion or errors encountered during processing. A fatal error results in job termination. In recreating the card for the error listing, the parameters are changed to reflect the rid (record reference identifier) or gid (group reference identifier) being processed when the error was encountered. These messages are included in Appendix B.

JOB OUTPUT INFORMATION D

Appendix D lists the output information printed for the sample job below. The notes in the right margin identify the various format conventions of KRONOS output.

The following control cards are submitted.

```
TESTA(CM50000,T10)
USER(JEANCOM,PASSWOR,SYS172)
FTN.
-EOR-
      PROGRAM CONVER(INPUT,OUTPUT)
C         THIS PROGRAM CONVERTS OCTAL TO DECIMAL
C         THE SECOND VALUE PRINTED IS 10 OCTAL TIMES THE FIRST
C         TERMINATE BY TYPING ZERO
    2 CONTINUE
      READ 1,J
    1 FORMAT(08)
      K=J*10B
      PRINT 6,J,K
    6 FORMAT(5X,I10,5X,I10)
      IF(J.EQ.0)3,2
    3 CONTINUE
      STOP
      END
-EOI-
```

```
KRONOS 2.1-484.      75/04/19.
        OPERATING SYSTEM
      JOB ORIGIN = BATCH.

     USER NUMBER = JEANCOM
   JOBCARD NAME = TESTA00
```

The first three lines of the banner page indicate that this local batch job was run under the control of version 2.1.2 of the KRONOS Operating System. The system creation date was April 19, 1975.

The user number is that which was supplied on the USER card. The jobcard name is the name of the particular job which was supplied on the job card.

```
 AAAAAAAAAA    AAAAAAAAAA   FFFFFFFFFFF  IIIIIIIIIII   AAAAAAAAAA    CCCCCCCCCC   FFFFFFFFFFF
AAAAAAAAAAAA  AAAAAAAAAAAA  FFFFFFFFFFF  IIIIIIIIIII  AAAAAAAAAAAA  CCCCCCCCCCCC  FFFFFFFFFFF
AA          AA AA          AA FF            II      AA          AA CC          CC FF
AA          AA AA          AA FF            II      AA          AA CC             FF
AA          AA AA          AA FF            II      AA          AA CC             FF
AA          AA AA          AA FF            II      AA          AA CC             FF
AA          AA AA          AA FF            II      AA          AA CC             FF
AA          AA AA          AA FFFFFFFF      II      AA          AA CC             FFFFFFFF
AAAAAAAAAAAA AAAAAAAAAAAA FFFFFFFF      II      AAAAAAAAAAAA CC             FFFFFFFF
AAAAAAAAAAAA AAAAAAAAAAAA FF            II      AAAAAAAAAAAA CC             FF
AA          AA AA          AA FF            II      AA          AA CC             FF
AA          AA AA          AA FF            II      AA          AA CC             FF
AA          AA AA          AA FF            II      AA          AA CC             FF
AA          AA AA          AA FF            II      AA          AA CC          CC FF
AA          AA AA          AA FF       IIIIIIIIIII  AA          AA CCCCCCCCCCCC FF
AA          AA AA          AA FF       IIIIIIIIIII  AA          AA  CCCCCCCCCC  FF
```

The first four characters of the banner job name are generated from the user index associated with the user number. These four characters are unique to each user and remain the same for subsequent jobs run under the same user number. The last three characters are the job sequence number assigned by the system at the time of processing.

```
75/06/02. 13.26.27.
```

This line specifies the current date (June 2, 1975) and the time (hh.mm.ss.) when the job printing was initiated.

```
      PROGRAM CONVER     73/74   OPT=1                    FTN 4.4+U401      75/06/02. 13.26.24.      PAGE     1

    1                    PROGRAM CONVER(INPUT,OUTPUT)
                    C       THIS PROGRAM CONVERTS OCTAL TO DECIMAL
                    C       THE SECOND VALUE PRINTED IS 10 OCTAL TIMES THE FIRST
                    C       TERMINATE BY TYPING ZERO
    5                   2 CONTINUE
                        READ 1,J
                        1 FORMAT(O8)
                        6 FORMAT(5X,I10,5X,I10)
                        K=J*10B
   10                   PRINT 6,J,K
                        IF(J.EQ.0)3,2
                        3 CONTINUE
                        STOP
                        END


      SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
 4107  CONVER

VARIABLES     SN  TYPE           RELOCATION
 4137  J          INTEGER                          4140  K          INTEGER

FILE NAMES       MODE
    0  INPUT     FMT           2041  OUTPUT    FMT

STATEMENT LABELS
 4124  1        FMT                        4110  2                                  0  3          INACTIVE
 4126  6        FMT

STATISTICS
  PROGRAM LENGTH                368       30
  BUFFER LENGTH               41038     2115
```

The job calls the FORTRAN Extended compiler which compiles the program, CONVER, contained in the program record. The symbolic reference map for program CONVER is printed below.

```
        AAFIACF.   75/06/02.(10) CYBER73      KRONOS

13.26.23.TESTA,CM50000,T10.
13.26.23.USER,JEANCOM,,SYS172.
13.26.24.FTN.
13.26.25.       .089 CP SECONDS COMPILATION TIME
13.26.25.UEMS,       0.573KUNS.
13.26.25.UECP,       0.093SECS.
13.26.25.AESR,       1.000UNTS.
13.26.32.UCLP, 23,       0.192 KLNS.
```

This line specifies the job name, the current date, and the computer system being used.

The dayfile includes a listing of the control cards, system supplied status messages, and program output, if any. Spaces precede status messages and program output. Each line includes the time the message was issued to the dayfile.

The last four lines specify the type and amount of system resources the job used. This job used 0.573 kilo-units of mass storage activity, 0.093 seconds of CPU time, and 1.000 SRU. The job produced 0.192 kilo-lines of printable output. Depending on the resources used, additional information may be included in the dayfile. The formats of these messages are given under Job Completion in section 3.

PERMANENT FILE DEVICE STATISTICS E

The system allocates space for permanent mass storage files in units called reservation blocks. The size of a reservation block depends upon the type of file and/or the type of device on which the file is to reside. For indirect access files, the reservation block size is always one PRU (64 CM words), regardless of the device residence. For direct access files, the reservation block size is a multiple of PRUs and varies according to the device type, as shown in the following table.

Device Type	Device	PRUs/ Block	CM Words	Characters	Maximum No. of Blocks
DA	6603 Disk System	50, 64	3200, 4096	32,000, 40,960	2048
DB	6638 Disk System	49	3136	31,360	2048
DC	863 Drum Storage	25	1600	16,000	256
DD	853/854 Disk Storage Drive ($1 \leq n \leq 8$)	n*26	n*1664	n*16,640	n*400 (854) n*200 (853)
DE	Extended Core Storage	16	1024	10,240	121 for 125K 243 for 250K
DF	814 Disk File	85	5440	54,400	2048
DH	821 Data File	320	20,480	204,800	2048
DI	844 Disk Storage Subsystem ($1 \leq n \leq 8$)	n*107	n*6,848	n*68,480	n*1616
DP	Distributive Data Path to ECS	16	1024	10,240	121 for 125K 243 for 250K
MD	841-n Multiple Disk Drive ($1 \leq n \leq 8$)	n*32	n*2048	n*20,480	n*1600

In this table, n indicates the unit count for multiunit devices.

In general, the largest permanent file the user can create is a direct access file that resides on a nonmaster device within his family of permanent file devices. Such files are restricted in size by the limitations of the device itself and the DS validation parameter which limits the size of direct access files. If no DS restriction is imposed, the maximum file size equals the maximum number of reservation blocks that can be allocated for the device.

All other permanent files reside either on the user's master device or on an auxiliary device. Their maximum size is restricted to the device limit minus any space allocated for catalog information and other files. In addition, an installation can use the FS validation parameter to limit the size of indirect access files.

CARD FORMAT AND CONVERSION PROBLEMS F

Data within the system is stored in binary or coded records. Binary records are variable in length and consist of central memory images. Coded records consist of lines of display-coded characters. Binary and coded data can enter the system in several different formats as some enter the system directly; others must be converted to a recognizable form by the peripheral driver that enters the data. The converted data can reside in the system in several formats. The processing program must recognize the specific format and process the data accordingly.

Described are the formats for punched cards and the format for printed data. In addition, this appendix describes the conversion performed by the system on data transferred between the system and peripheral devices and the method by which time-sharing terminal data is converted in the system.

When using the 64 character set, the user should avoid using consecutive colons (00 characters). It is possible for these colons to be interpreted as an end-of-line. An end-of-line is defined as 12 to 66 bits of zero right-justified in one or two central memory words. If consecutive colons appear in the lower 12 bits of a central memory word, they are interpreted as an end-of-line rather than as colons.

Example:

The following characters are punched on a coded card beginning in column 1:

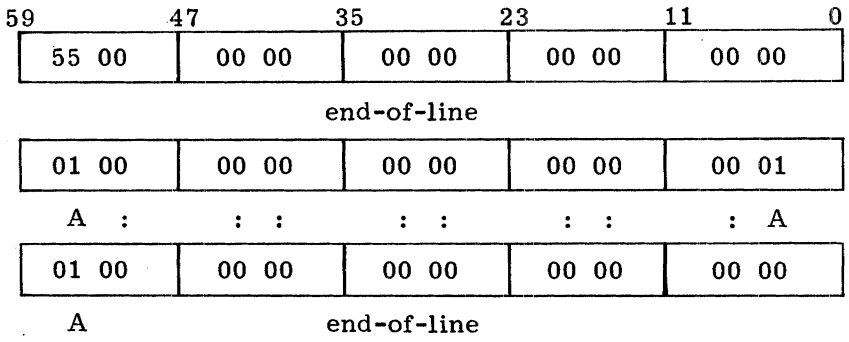
::::::::A::::::::AA

This would appear in memory as follows:

59	47	35	23	11	0
00 00	00 00	00 00	00 00	00 01	
: :	: :	: :	: :	: A	
00 00	00 00	00 00	00 00	01 01	
: :	: :	: :	: :	A A	
00 00	00 00	00 00	00 00	00 00	

end-of-line

However, if the characters were copied with the COPYSBF utility, the following would appear:



NOTE

If a colon is the last character of a line, the system appends a blank character to preserve the colon and then appends an end-of-line. (Two blanks may be added to ensure an even number of characters.)

FORMATS FOR CARDS READ

The system reads cards in coded and binary formats. The following conditions apply in both formats.

1. A card with a 7/8/9 punched in column 1 is an EOR mark.
2. A card with a 6/7/9 punched in column 1 is an EOF mark.
3. A card with a 6/7/8/9 punched in column 1 is an EOI mark.

The reaminder of each card is ignored except for columns 79 and 80 of the EOR and EOF cards. These columns can contain the keypunch conversion mode for the input records that follow. Conversion modes are discussed in the following section.

CODED CARDS

Cards are read in Hollerith punch code. The 3447 card reader controller converts the Hollerith code to internal BCD code and passes the data to the card reader driver. The driver converts the data from internal BCD code to display code. Up to 80 characters can be transferred per card. Trailing spaces are deleted.

Two conversion modes, O26 and O29,† exist for the Hollerith punch code. All data is converted in the system default keypunch mode unless a conversion mode change is specified. This change can be specified on any of the following cards.

The job card, 7/8/9 card (EOR mark), and 6/7/9 (EOF mark) can contain the keypunch conversion mode in columns 79 and 80. A 26 punched in columns 79 and 80 indicates all subsequent coded cards are converted in O26 mode. A 29 indicates subsequent cards are converted in O29 mode. Each conversion change remains in effect until another change card is encountered or the job ends. The user can switch between O26 and O29 mode as often as desired. If 26 or 29 does not appear in columns 79 and 80 of the

†These codes are ignored by a 200 User Terminal since conversion mode is selected by a hardware switch. (Refer to the Export/Import Reference Manual.)

job card, the initial keypunch mode of that job is the system default mode. If 26 or 29 does not appear on a 7/8/9 or 6/7/9 card, no conversion change is made and the most recent keypunch mode remains in effect.

Keypunch mode can also be changed by a card containing a 5/7/9 punch in column 1. A blank (no punch) in column 2 indicates O26 conversion mode; a 9 punched in column 2 indicates O29 mode. The conversion change remains in effect until another change card is encountered or the job ends.

The 5/7/9 card also allows literal input when 4/5/6/7/8/9 is punched in column 2. Literal input allows 80 column binary data to be read while transmitting input in coded mode. Cards are read (16 central memory words per card) until a card identical to the previous 5/7/9 card (4/5/6/7/8/9 in column 2) is read. The next card can then specify the new conversion mode.

BINARY CARDS

Binary cards are denoted by a 7/9 punch in column 1 and can contain up to 15 central memory words. The 3447 card reader controller reads the binary data and passes it to the card reader driver in 12-bit codes. Each card column row corresponds to a bit position. The driver checks the checksum figure if this option is specified. The driver then passes the data to the central memory buffer.

The fields within a binary card are:

Column(s)	Description
1	7/9 punch indicates a binary card
	4 punch ignores checksum punch in column 2
	Rows 0, 1, 2, and 3 contain the binary equivalent of the word count of the card
2	Binary data checksum (modulo 4095)
3 through 77	15 central memory words of binary data
78	Blank
79 and 80	24-bit binary card sequence number

SUMMARY

The following punches appearing in column 1 of a card have the corresponding meaning to the card reader driver.

Punch	Represents
7/8/9	End-of-record (optional conversion mode change)
6/7/9	End-of-file (optional conversion mode change)
6/7/8/9	End-of-information
5/7/9	Conversion mode change/Read 80 column binary
7/9	Binary card
Not 7 and 9	Coded card

FORMAT FOR CARDS PUNCHED

Punched cards can be in three formats.

- Coded (punch Hollerith)
- Binary
- Absolute binary

The following conditions apply to all three formats.

- When an EOR is encountered, a card is punched with a 7/8/9 in columns 1 and 80. This card is offset.
- When an EOF is encountered for a file, a card is punched with a 6/7/9 in columns 1 and 80; the remainder of the card is blank. This card is offset.
- When an EOI is encountered on a file, a card is punched with a 6/7/8/9 in columns 1 and 80; the remainder of the card is blank. This card is offset.
- If a compare error is encountered, the erroneous card and the following card are offset. These two cards are repunched until no error is detected. An EOI card with 6/7/8/9 punches in columns 1 and 80 contains a binary count in column 40 of the number of compare errors.
- During the punching of each file, the system maintains a count of the number of cards punched for the file. If the number exceeds the limit for which the user is validated, punching of the file is terminated. A special banner card with the word LIMIT is punched and offset as the last card of the deck.

The following methods are used by the system to punch each of the three forms of cards.

CODED CARDS (PUNCH)

With the exception of decks punched via the DISPOSE request, the keypunch mode (O26 or O29) of coded cards depends on the job origin type. If the job is of local batch origin, decks are punched in the initial keypunch mode; that is, the mode specified on the job card or set by system default. For all other job origin types, decks are punched in the system default keypunch mode. However, the DISPOSE request allows the user to specify that decks be punched in either O26 or O29 mode regardless of the job's keypunch mode.

BINARY CARDS (PUNCHB)

The card punch driver retrieves 15 words of binary data from central memory. The driver then generates a checksum for the data and issues a card number. The card punch controller receives the binary data and punches it on the card unchanged, that is, in 12-bit codes. Each row in a card column corresponds to a bit position. The driver formats the binary card in the following manner.

Column(s)	Contents
1	7/9 punch denotes binary card Rows 0, 1, 2, and 3 contain the binary equivalent of the word count of the card
2	Binary data checksum (modulo 4095)
3 through 77	15 central memory words of binary data
78	Blank
79 and 80	24-bit binary card sequence number

ABSOLUTE BINARY CARDS (P8)

Absolute binary cards are central memory images in 12-bit codes. Each row in a card column corresponds to a bit position. Sixteen central memory words are punched per card with no special punches or fields added.

PRINTED DATA

All printed data is in coded format. The line printer driver extracts data until an end-of-line mark occurs or until 14 central memory words are retrieved. The end-of-line is denoted by a zero byte as the last byte of a central memory word. The print line consists of a maximum of 136 characters. If an end-of-line mark does not appear after 136 characters, the last four characters of that group are lost. The driver converts the extracted data from display code to internal BCD code (refer to appendix A for the character set equivalences) and forwards the data to the line printer controller.

The driver interprets the first character in a line as the carriage control character (refer to appendix A) and that character is not printed. In most cases, the proper carriage control is issued while the remainder of the line is printed. However, when Q, R, S, or T is specified, no printing takes place for that line. The Q, R, S, and T format controls remain in effect until changed, and all other carriage control options must be supplied for each line they control. Line spacing is normally done in the auto eject mode; that is, creases in the paper are skipped by the line printer controller's automatic line spacing mechanism if the paper is loaded properly.

During the printing of each file, the system maintains a count of the number of lines printed/skipped for the file. If the number exceeds the limit for which the user is validated, printing of the file is terminated. The informative diagnostic LINE LIMIT EXCEEDED is printed. If a job's dayfile is part of the terminated print file, the dayfile is subsequently printed.

The installation can impose an implied page control by setting a certain number of default lines for each page. If less than the default number of lines is printed/skipped on a page, the line limit is still decremented by the default number of lines.

TERMINAL CHARACTER CONVERSION

Normal input mode from a terminal consists of a 64-character set where all lowercase alphabetics are converted to uppercase characters. Under ASCII mode, the characters 74 and 76 represent the beginning of a 74xx or 76xx escape sequence. Under NORMAL mode, the characters 74 and 76 are treated as data rather than escape codes. ASCII and NORMAL modes apply to both input and output.

DATA INPUT

The terminals KRONOS supports can be grouped into ASCII terminals and correspondence code terminals. The manner in which the characters entered from a terminal are interpreted by the system depends on whether the user specifies that the characters belong to the full character set. For example, if the user enters the following characters to be mapped into the full ASCII set:

aAbBcCdDeEfF

the central memory equivalent is:†

59	47	35	23	11	0
76 01	01 76	02 02	76 03	03 76	
04 04	76 05	05 76	06 06	00 00	

However, if a NORMAL command is issued, the characters are mapped into the subset of the ASCII character set; then the central memory equivalent is:

59	47	35	23	11	0
01 01	02 02	03 03	04 04	05 05	
06 06	00 00	00 00	00 00	00 00	

In ASCII mode, †† all 128 characters from an ASCII or correspondence code terminal are recognized. These characters, in addition to the first 64, are processed as 12-bit characters with an escape code conversion as shown previously. Appendix A lists the character set equivalences. The programs that process data must recognize that data is in ASCII mode rather than normal display code and process it accordingly.

DATA OUTPUT

Data output is in either a 64/63 or 128 character set, depending on whether the terminal is in NORMAL or ASCII mode. When the terminal is in NORMAL mode, the codes 74 and 76 represent data rather than escape codes. In ASCII mode, 74 and 76 are treated as the beginning of an escape sequence. All information is transmitted in even parity unless the user specifies odd parity.

For a more detailed description of terminal operation, refer to the Time-Sharing User's Reference Manual.

Data can also be transmitted to or from a terminal through a paper tape reader. The paper tape character mode is always ASCII.

---

† Partial words are zero-filled; partial bytes are blank-filled.

†† Refer to the Time-Sharing User's Reference Manual for descriptions of the ASCII and NORMAL commands.

TAPE LABELS G

The operating system accepts ANSI standard and nonstandard labeled tapes. Labels which do not conform to ANSI standards in format and/or content are defined as nonstandard.

ANSI labels perform two functions. They provide information that uniquely identifies a file and the reel on which it resides, and they mark the beginning and end of a file and the beginning and end of a reel.

ANSI labels are designed to conform to the American National Standard Magnetic Tape Labels for Information Interchange X3.27-1969. All labels are 80 characters in length and are recorded at the same density as the data on the tape. The first three characters of an ANSI label identify the label type. The fourth character indicates a number within a label type.

The following is a summary of each label type, name, function, and whether or not it is required.

Type	No.	Name	Used At	Required/Optional
VOL	1	Volume header label	Beginning-of-volume	Required
UVL	1-9	User volume label	Beginning-of-volume	Optional
HDR	1	File header label	Beginning-of-file	Required
HDR	2-9	File header label	Beginning-of-file	Optional
UHL	†	User header label	Beginning-of-file	Optional
EOF	1	End-of-file label	End-of-file	Required
EOF	2-9	End-of-file label	End-of-file	Optional
UTL	†	User trailer label	End-of-file	Optional
EOV	1	End-of-volume label	End-of-volume	Required when appropriate
EOV	2-9	End-of-volume label	End-of-volume	Optional

REQUIRED LABELS

The VOL1, HDR1, and EOF1 labels are required on all ANSI-labeled tapes. In addition, an EOV1 label is required if the physical end-of-tape reflector is encountered before an EOF1 label is written or if a multifile set is continued on another volume. In the descriptions of the contents of these labels, n is any numeric digit and a is any letter, digit, or any of the following special characters.

† Any member of the CDC 6-bit subset of the ASCII character set.

Δ	)	<
!	*	=
"	+	>
≠	,	?
$	-	@
%	.	[
&	'	\
/	:	]
(	;	^

Some fields are optional. An optional field which does not contain the designated information must contain blanks. Fields which are not described as optional are required and will be written as specified. Note that n-type fields are right-justified and zero-filled, and a-type fields are left-justified and blank-filled.

VOL1 - VOLUME HEADER LABEL

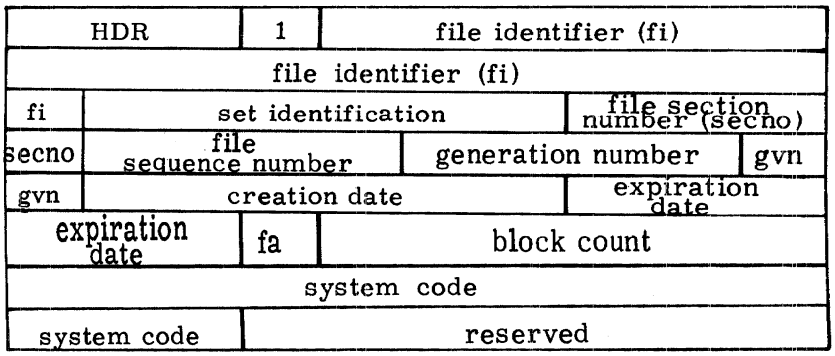
The volume header label must be the first label on a labeled tape. All reels begin with a VOL1 label. If two or more reels belong to a volume set, the file section field in the following HDR1 label gives the actual reel number.

<table>
<tr><td colspan="2">VOL</td><td>1</td><td colspan="3">volume serial number</td></tr>
<tr><td>va</td><td colspan="5">reserved</td></tr>
<tr><td colspan="6">reserved</td></tr>
<tr><td colspan="4">reserved</td><td colspan="2">owner identification</td></tr>
<tr><td colspan="6">owner identification (oid)</td></tr>
<tr><td>oid</td><td colspan="5">reserved</td></tr>
<tr><td colspan="6">reserved</td></tr>
<tr><td colspan="5">reserved</td><td>lsl</td></tr>
</table>

Character Position	Field Name	Length (in characters)	Contents	Default	Checked on Read
1-3	Label identifier	3	Must be VOL		Yes
4	Label number	1	Must be 1		Yes
5-10	Volume serial number	6	Volume identification assigned by owner to identify this physical reel of tape	As read from existing label	Yes, if the file was assigned by volume serial number
11	Accessibility (va)	1	An a character which indicates the restrictions, if any, on who may have access to the information on the tape. A blank means unlimited access. Any other character means special handling, in the manner agreed between the interchange parties. Refer to the BLANK control card.	Blank (unlimited access)	No (refer to BLANK control card)
12-31	Reserved for future standardization	20	Must be blanks		No
32-37	Reserved for future standardization	6	Must be blanks		No
38-51	Owner identification (oid)	14	Any a characters identifying the owner of the physical volume	family name, user number	Refer to discussion of fa field of HDR1.
52-79	Reserved for future standardization	28	Must be blanks		No
80	Label standard level (lsl)	1	1 means the labels and data formats on this volume conform to the requirements of the ANSI standard. A blank means the labels and data formats on this volume require the agreement of the interchange parties.	1	No

HDR1 - FIRST FILE HEADER LABEL

The first file header label must appear before each file. When a file is continued on more than one volume, the file header label is repeated after the volume header label on each new volume for that file. If two or more files are grouped in a multifile set, each HDR1 label indicates the relative position of its associated file within the set.



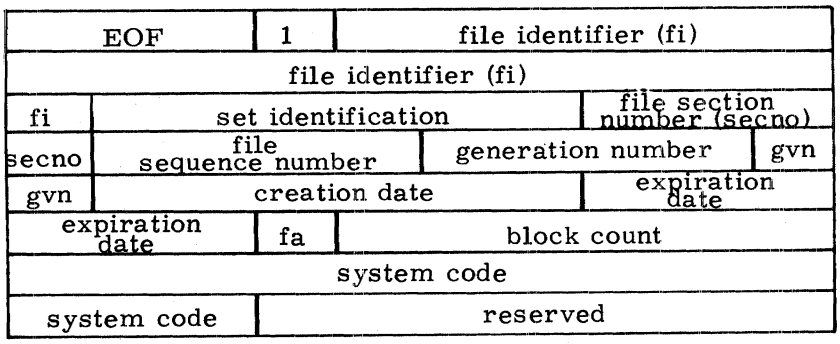
Character Position	Field Name	Length (in characters)	Contents	Default	Checked on Read
1-3	Label identifier	3	Must be HDR		Yes
4	Label number	1	Must be 1		Yes
5-21	File identifier (fi)	17	File identification (fileid) parameter on the LABEL control card	Blank	Checked if specified
22-27	Set identification	6	Set identification as specified by the setid parameter on the LABEL control card. This value must be the same for all files of a multifile set.	Blank	Checked if specified
28-31	File section number (secno)	4	The file section number of the first HDR1 label of a file is 0001. If the file extends to more than one volume, this number is incremented by one for each subsequent volume. This value corresponds to the secno parameter on the LABEL card.	0001	Checked if specified
32-35	File sequence number	4	Position of a file within a file set, as specified by the seqno parameter of the LABEL card. This value is 0001 for the first file, 0002 for the second, and so on. In all the labels for a given file, this field will contain the same number.	0001	Checked if specified

Character Position	Field Name	Length (in characters)	Contents	Default	Checked on Read
36-39	Generation number (optional)	4	Generation number of a file, as specified by the genno parameter of the LABEL card. This value is 0001 for the first generation of a file, 0002 for the second, and so on.	0001	Checked if specified
40-41	Generation version number (gvn)	2	Two n characters used to distinguish successive iterations of the same generation. The generation version number of the first attempt to create a file is 00. This value corresponds to the gvn parameter of the LABEL control card.	00	Yes
42-47	Creation date	6	Date the file was created; it is recorded as a space followed by two n characters for the year followed by three n characters for the day within the year. This value corresponds to the cdate parameter of the LABEL control card.	current date	Yes. The creation date is meaningful only on read operations; on write operations, the current date is always used.
48-53	Expiration date	6	The file is considered expired when today's date is equal to or later than the date given in this field. When this condition is satisfied, the remainder of the volume may be overwritten. Thus, to be effective on multifile volumes, the expiration date of a file must be less than or equal to the expiration date of all preceding files on the volume. The expiration date is written in the same format as the creation date.	current date	Checked if write attempted

Character Position	Field Name	Length (in characters)	Contents	Default	Checked on Read
			It corresponds to the rdate parameter of the LABEL control card.		
54	Accessibility (fa)	1	An a character which indicates the restrictions, if any, on who may have access to the information in this file. A blank means unlimited access. If fa is A, only the owner of the KRONOS 2.1 written tape can access the file. If fa is any other character, all future accesses to the tape must specify this character as the fa parameter. File accessibility is not checked for system origin jobs.	Blank (unlimited access)	Yes, if a KRONOS 2.1 written tape
55-60	Block count	6	Must be zeros		No
61-73	System code	13	13 a characters identifying the operating system that recorded this file. The tape is considered to have been writen under KRONOS 2.1 if the first 10 characters match the default.	KRONOS 2.1-nn (nn is the EST ordinal of the unit on which the file was written)	No
74-80	Reserved for future standardization	7	Must be spaces		No

EOF1 - FIRST END-OF-FILE LABEL

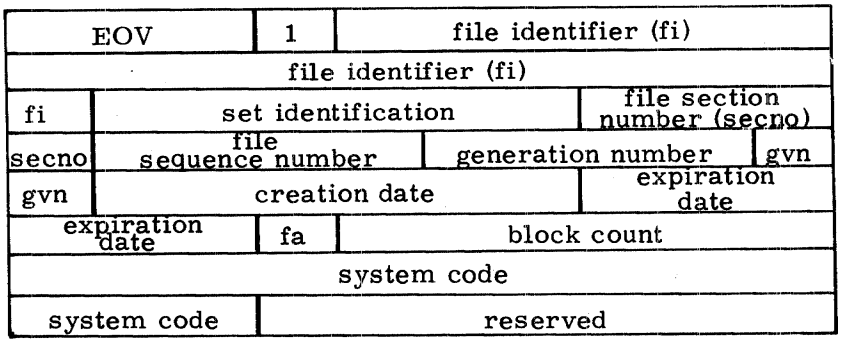
The end-of-file label is the last block of every file. It is the KRONOS end-of-information for the file. A single tape mark precedes EOF1. A double tape mark written after the EOF1 label marks the end of a multifile set.



Character Position	Field Name	Length (in characters)	Contents	Default	Checked on Read
1-3	Label identifier	3	Must be EOF		Yes
4	Label number	1	Must be 1		Yes
5-54	Same as corresponding fields in HDR1 (optional)	50	Same as the corresponding fields in HDR1		Same as HDR1
55-60	Block count	6	Six n characters specifying the number of data blocks between this label and the preceding HDR label group. This total does not include labels or tape marks.		Yes
61-80	Same as corresponding fields in HDR1 (optional)	20	Same as corresponding fields in HDR1		Same as HDR1

EOV1 - FIRST END-OF-VOLUME LABEL

The end-of-volume label is required only if the physical end-of-tape reflector is encountered before an EOF1 label is written or if a multifile set is continued on another volume. EOV1 is preceded by a single tape mark and followed by a double tape mark.



Character Position	Field Name	Length (in characters)	Contents	Default	Checked on Read
1-3	Label identifier	3	Must be EOV		Yes
4	Label number	1	Must be 1		Yes
5-54	Same as the corresponding fields in HDR1 (optional)	50	Same as the corresponding fields in HDR1		Same as HDR1
55-60	Block count	6	Six n characters specifying the number of data blocks between this label and the preceding HDR label group. This total does not include labels or tape marks.		Yes
61-80	Same as the corresponding fields in HDR1 (optional)	20	Same as the corresponding fields in HDR1		Same as HDR1

These labels define four possible file configurations.

- A single file on a single volume
- A single file on two or more volumes
- Two or more files on a single volume
- Two or more files on two or more volumes

Figures 1-G-1 through 1-G-7 illustrate the use of ANSI labels in these configurations.

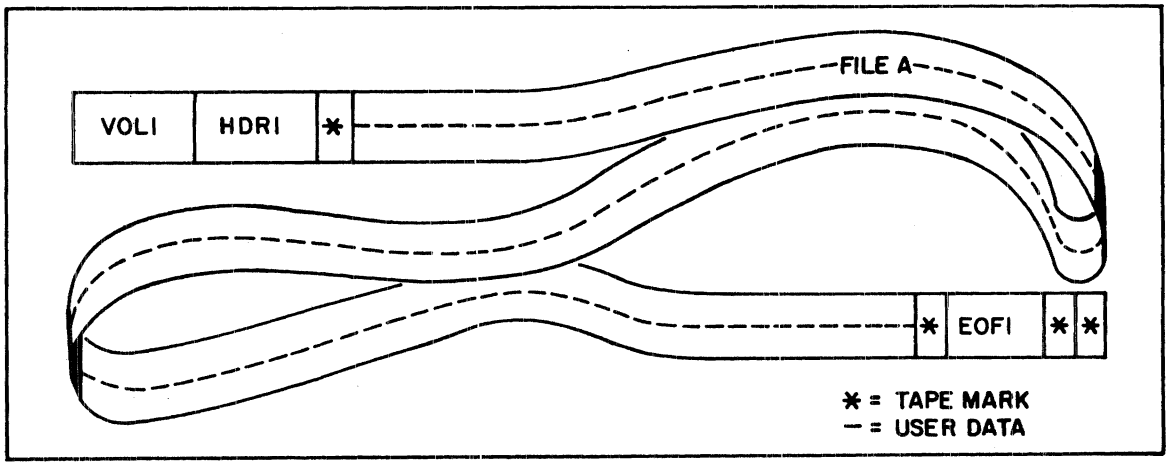


Figure 1-G-1. ANSI Labels: Single File, Single Volume

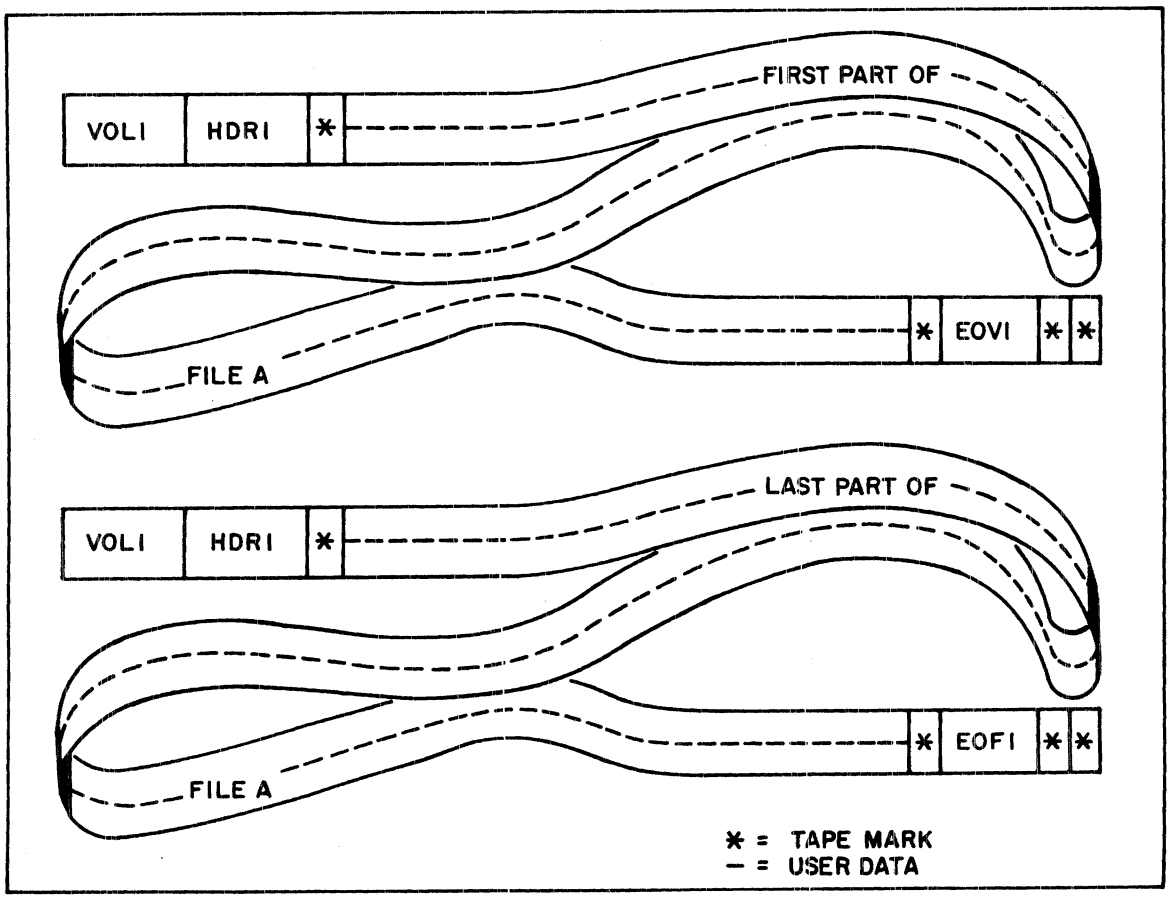


Figure 1-G-2. ANSI Labels: Single File, Multivolume

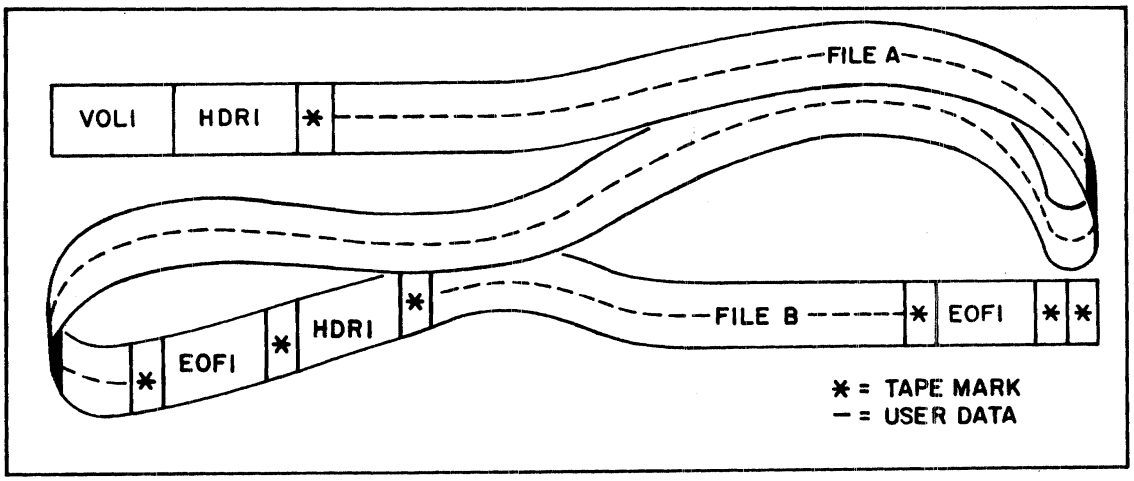


Figure 1-G-3 ANSI Labels: Multifile, Single Volume

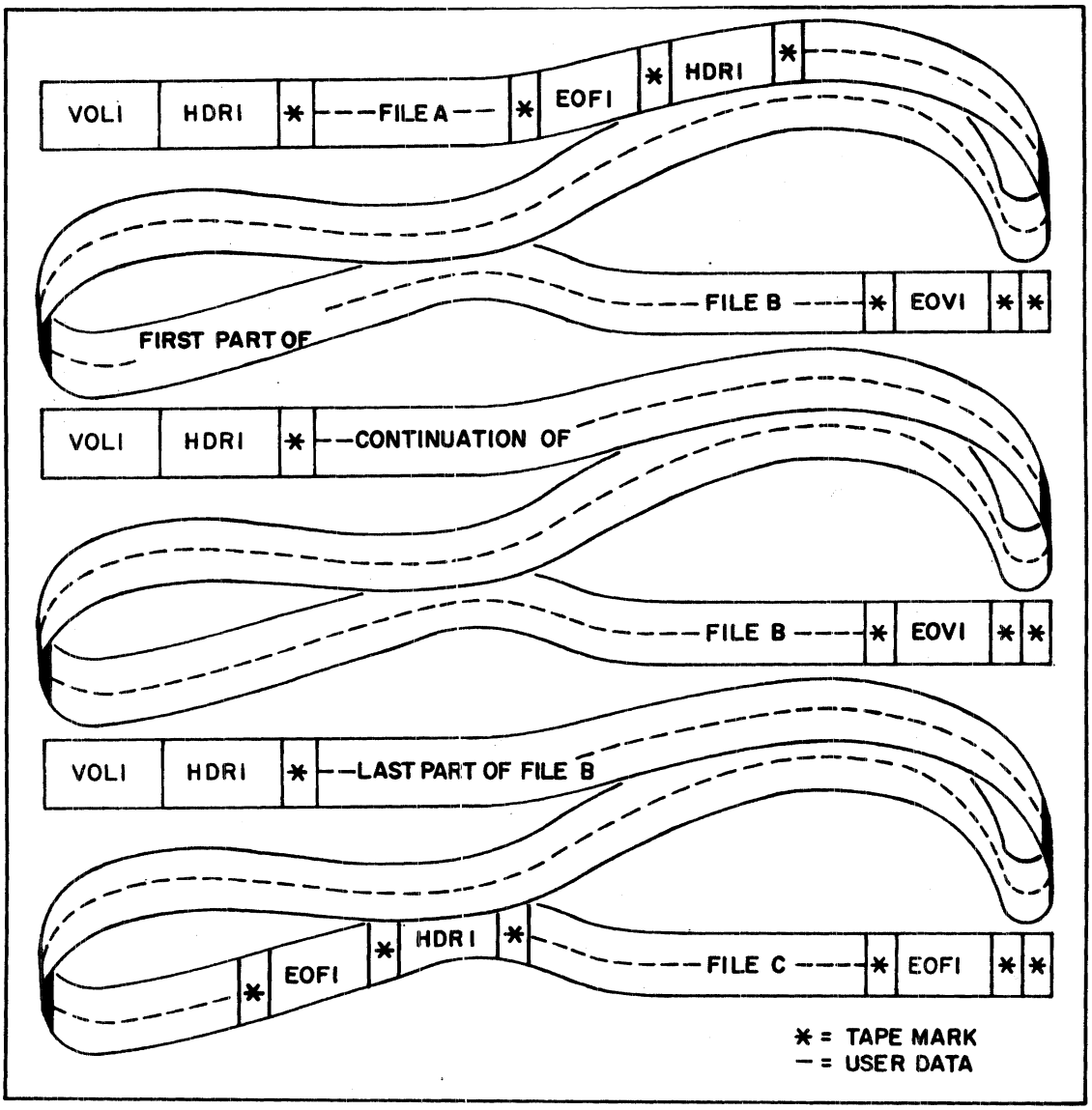


Figure 1-G-4. ANSI Labels: Multifile, Multivolume

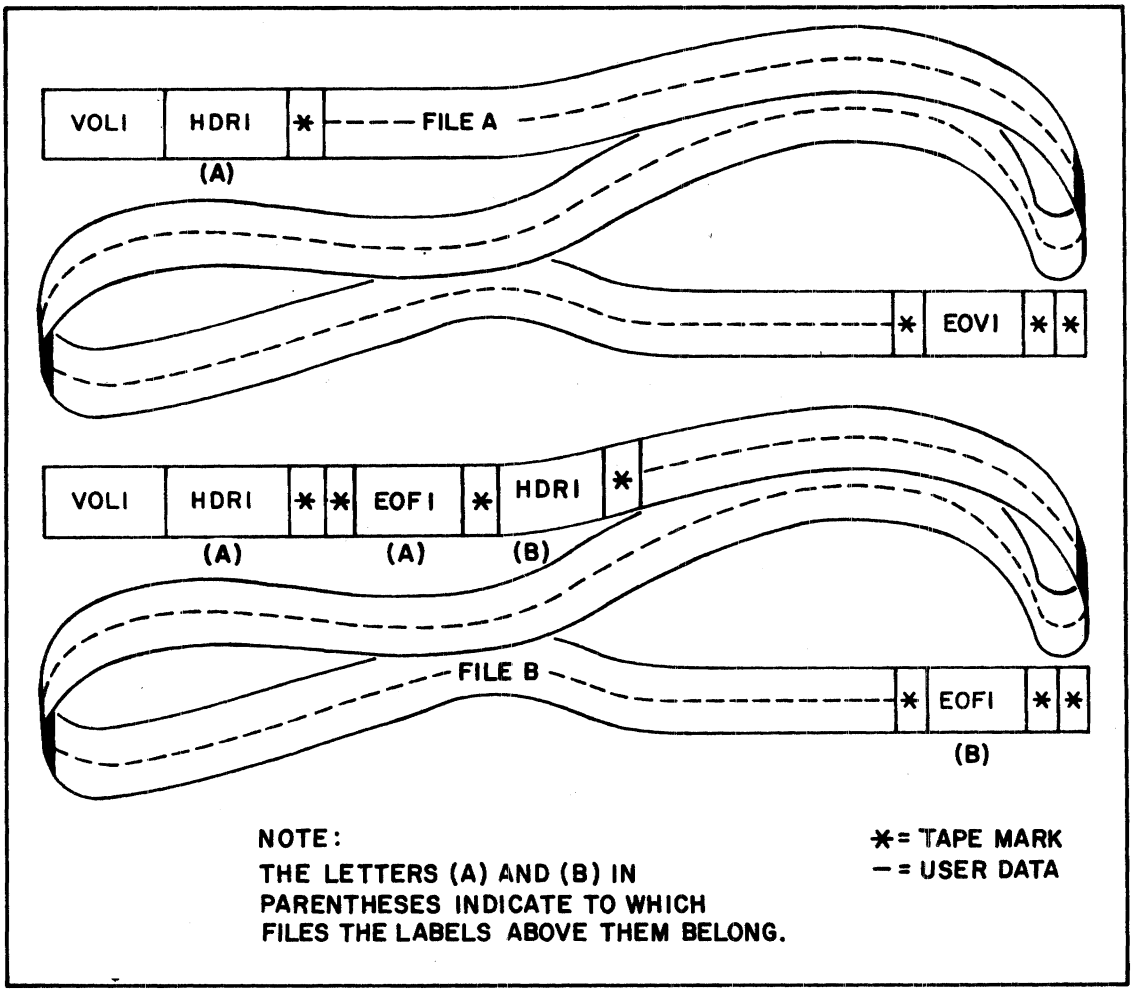


Figure 1-G-5. ANSI Labels: End-of-File, End-of-Volume Coincidence

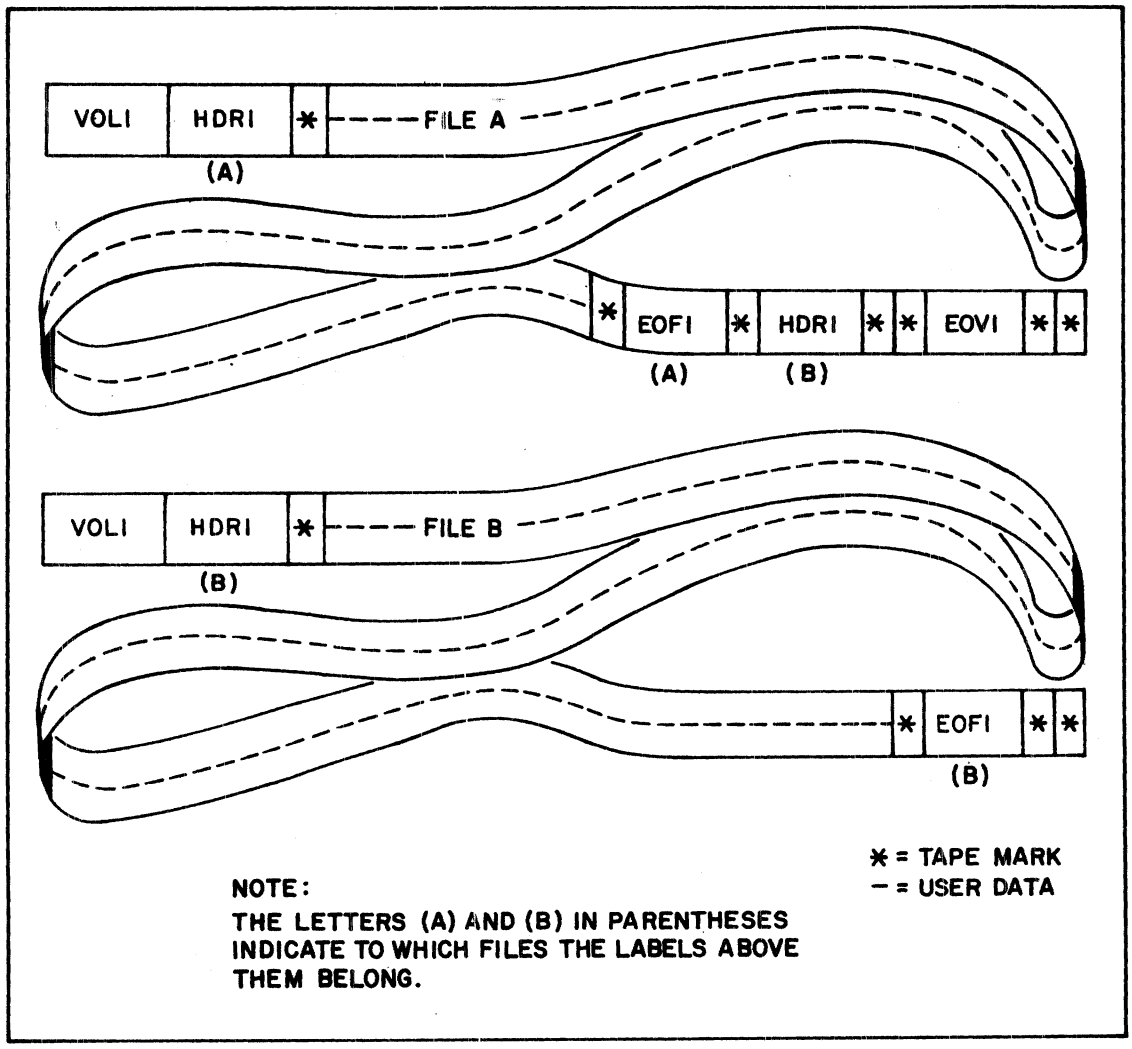


Figure 1-G-6. ANSI Labels: End-of-File, End-of-Volume Coincidence

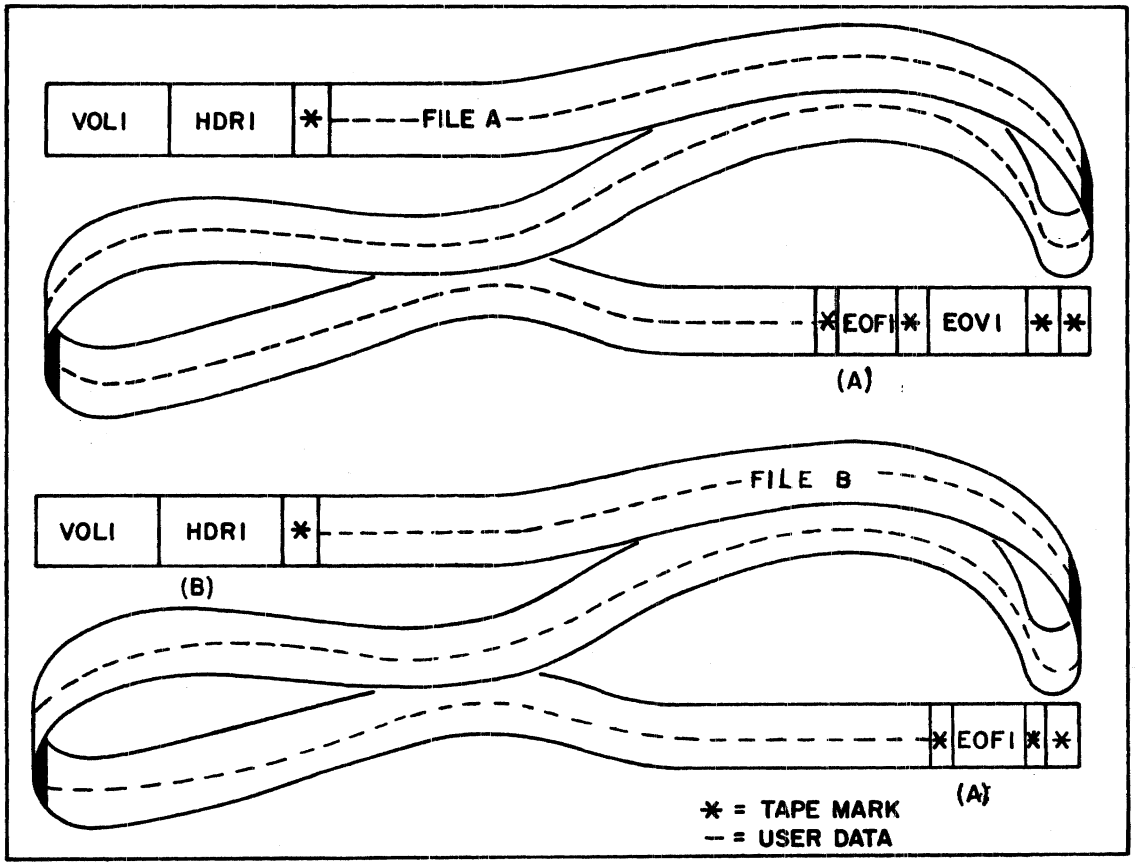


Figure 1-G-7. ANSI Labels: End-of-File, End-of-Volume Coincidence

OPTIONAL LABELS

Six types of optional labels are allowed. They are additional header (HDR2-9), end-of-file (EOF2-9), end-of-volume (EOV2-9), user volume (UVLa), header (UHLa), and trailer (UTLa) labels.

(HDR2-9) - ADDITIONAL FILE HEADER LABELS

HDR2-9 labels may immediately follow HDR1. Their format is:

Character Position	Field Name	Length (in characters)	Contents	Default Written
1-3	Label identifier	3	HDR	HDR
4	Label number	1	2-9	2-9
5-80		76		

Only the label identifier and the label number are checked on read.

(EOF2-9) - ADDITIONAL END-OF-FILE LABELS

EOF2-9 labels may immediately follow EOF1. Their format is:

Character Position	Field Name	Length (in characters)	Contents	Default Written
1-3	Label identifier	3	EOF	EOF
4	Label number	1	2-9	2-9
5-80		76		

Only the label identifier and the label number are checked on read.

(EOV2-9) - ADDITIONAL END-OF-VOLUME LABELS

EOV2-9 labels may immediately follow EOV1. Their format is:

Character Position	Field Name	Length (in characters)	Contents	Default Written
1-3	Label identifier	3	EOV	EOV
4	Label number	1	2-9	2-9
5-80		76		

Only the label identifier and the label number are checked on read.

Refer to section 3, volume 2 for a description of the use of EOV2 labels in conjunction with CLOSER, REWIND, and UNLOAD macros.

USER LABELS

USER labels may immediately follow their associated system labels. Thus, user volume labels (UVLa) may follow VOL1, user header labels (UHLa) may follow the last HDRn label, and user trailer labels (UTLa) may follow the last EOVn or EOFn label. Their format is:

Character Position	Field Name	Length (in characters)	Contents	Default Written
1-3	Label identifier	3	UVL, UHL, or UTL	UVL, UHL, or UTL
4	Label number	1	Must be 1, 2, 3, 4, etc., consecutively for UVL labels. For other labels, any a character.	
5-80	User option	77	Any a characters.	

Only the label identifier and the label number are checked on read. The system checks the number of user labels of a label type; a maximum of 64 is allowed.

INDEX

COMMENT SHEET

MANUAL TITLE CDC KRONOS 2.1 Reference Manual, Volume 1

PUBLICATION NO. 60407000 REVISION D

FROM: NAME:

BUSINESS ADDRESS:

COMMENTS:

This form is not intended to be used as an order blank. Your evaluation of this manual will be welcomed by Control Data Corporation. Any errors, suggested additions or deletions, or general comments may be made below. Please include page number references and fill in publication revision level as shown by the last entry on the Record of Revision page at the front of the manual. Customer engineers are urged to use the TAR.

CUT ALONG LINE

AA3419 REV. 11/69 PRINTED IN U.S.A.

NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE

FOLD

FOLD

FIRST CLASS
PERMIT NO. 8241

MINNEAPOLIS, MINN.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY
CONTROL DATA CORPORATION
Publications and Graphics Division
ARH219
4201 North Lexington Avenue
Saint Paul, Minnesota 55112

CUT ALONG LINE

FOLD

FOLD

CORPORATE HEADQUARTERS, P.O. BOX 0, MINNEAPOLIS, MINNESOTA 55440
SALES OFFICES AND SERVICE CENTERS IN MAJOR CITIES THROUGHOUT THE WORLD

LITHO IN U.S.A.

CONTROL DATA CORPORATION