3100
3200
3300
3500

COMPUTER SYSTEMS
REAL-TIME SCOPE

REFERENCE MANUAL

CONTROL DATA
CORPORATION

REVISION RECORD

60172500

REVISION	NOTES
(10-66)	Original printing.
A	
(3-69)	Reprint with revision.

Additional copies of this manual may be obtained from the nearest Control Data Corporation sales office.

Pub. No. 60172500

Address comments concerning this manual to:

Control Data Corporation
Software Documentation
4201 North Lexington Avenue
St. Paul, Minnesota 55112

or use Comment Sheet in the back of this manual.

CONTENTS

LIST OF ILLUSTRATIONS

INTRODUCTION 1

The Real-Time SCOPE Monitor System supervises loading and execution of programs on CONTROL DATA® 3100, 3150, 3200, 3300, and 3500 Computer Systems. It permits multiprogramming of two independent programs. Stacked jobs are processed with minimum operator intervention. Priority programs are executed on a real-time and non-real-time basis; batch programs have control when the priority program is not in execution.

The Real-Time SCOPE batch only (BATCH) option features all the elements of the standard system except those specifically for priority processing. Standard Real-Time SCOPE has the following features.

- System control is provided through control statements entered by card or from the console typewriter.
- A priority program can submit job stacks for batch processing.
- High priority interrupts are recognized on channels reserved for real-time applications.
- Loading large programs into core in sections is facilitated by segmenting large priority and batch programs into main, overlay, and segment elements.
- Informative messages, diagnostic messages, recovery procedures, and debugging aids are standardized for major product set members.
- PRELIB generates new or updated system files.
- CIO, the centralized input/output routine, operates in conjunction with standard drivers for the following: magnetic tape, card reader, printer, card punch, console typewriter, paper tape, plotter, and optical character reader.
- The system product set contains ADAPT, TAPE SORT/MERGE, REPORT GENERATOR, COSY, SIPP, BSIPP, SCOPE UTILITY, FORTRAN, COBOL, and COMPASS.

1.1 OPERATION

The operator begins a Real-Time SCOPE (RTS) run by autoloading the resident portions of the system library. Resident contains the internal processing routines and tables that must be continuously available for reference by the batch and priority programs. Control statements direct loading and execution. A SEQUENCE card is the first card in the job stack, and the ENDSCOPE control card indicates the end of a job stack. RTS sequentially processes the series of independent jobs that are stacked on the standard input device.

A job consists of loading and executing one or more related programs; loading and execution of each program constitutes a run. Unless the BATCH option has been chosen, priority and batch programs may be in a common job stack. Each batch job is loaded and executed in turn with equipment assigned according to control statements accompanying the job. Batch equipment is released between jobs.

The priority program periodically performs on an interrupt-controlled basis. Each time it initiates a task, it returns control to RTS which continues stacked job processing. If there is no priority program to be run, RTS processes only the stacked batch programs. Conversely, if there is no batch program to be run while an active priority program is in core, execution is shared only with the RTS monitor. When the operator provides new batch programs, batch processing resumes automatically.

The operator may make calls to load and execute batch and priority jobs from the library. The manual interrupt scheme permits operator communication with RTS or with the batch or priority jobs being processed. The operator may obtain control after a SEQUENCE card is processed to intervene in normal processing and introduce other statements from the console typewriter.

1.2 PROGRAM LINKAGE

Compilers or assemblers independently process each subprogram written in source language. All symbols not declared external are local to the subprograms in which they occur. Locations referenced by other subprograms are called entry points. When the entry point matches the external symbol, the loader establishes linkage; when unmatched symbols remain, the loader searches the library tape for routines with entry points that are the names of these symbols. If all symbols are not matched, the job terminates with an error message.

1.3 PRIORITY PROGRAMS

A priority program (chapter 7) may be real-time or non-real-time; that is, it may or may not need immediate access upon the occurrence of some interrupt condition. It should be as brief and efficient as possible and is typically a special purpose job requiring control on interrupts or is input/output oriented. Real-time priority program interrupts (chapter 6) may interrupt the system input/output routines or may gain control of the processor through an interrupt at any other point in batch processing. Real-time priority interrupts are defined by each installation.

Input/output routines for real-time channels are supplied by the user and the system input/output routines cannot use these channels for general applications. A real-time program may use the system input/output routines for input/output on non-real-time channels.

A priority program:

- Must be in relocatable binary form for loading and execution
- Must be debugged if batch processing is to be performed concurrently
- May not have a common area
- May terminate itself when execution is complete or may be terminated through operator manual interrupt
- Has no time limit imposed by RTS
- May use overlays

1.4 BATCH PROGRAMS

Batch jobs are stacked jobs such as compilation, assembly, object program execution, and library generation. Batch jobs are executed when neither the priority program nor Real-Time SCOPE controls the central processing unit (CPU).

1.5 REAL-TIME OPERATION

Real-Time SCOPE permits one or more channels to be declared real-time channels. These channels are used for programs needing immediate access where data transfer must keep pace with a physical process during a given time period. All input/output is controlled outside the centralized input/output routines; all interrupts receive priority handling.

1.6 JOB STACKING

Real-Time SCOPE sequentially processes a series of independent jobs stacked on the standard input device. Control statements directing loading and execution accompany the stacked jobs.

Priority and batch programs may be in a common job stack. Typically, a priority program remains in core during processing of a number of batch jobs. To prevent either program from referencing the other, Real-Time SCOPE assigns batch and priority equipment independently. Each batch program is loaded and executed in turn with equipment assigned by control statements accompanying the job. Between jobs, Real-Time SCOPE releases batch equipment.

EQUIPMENT 2

2.1 HARDWARE CONFIGURATION

Minimum configuration requirements for running Real-Time SCOPE on a 3100, 3150, 3200, 3300, or 3500 computer system include:

Central processor

16K memory

2 data channels (1 module)

Console typewriter

Magnetic tape controller

3 magnetic tape units

The following devices, or additional tape units as substitutes, are required.

Punch device and controller

Input device and controller

Listable output device and controller

This configuration is sufficient for running an assembler (COMPASS), a compiler (FORTRAN), and a library generation program (PRELIB), but does not include peripheral devices for priority programs. A configuration requiring real-time channel assignments must have additional channels.

Configurations vary between installations; the typical configuration (figure 2-1) is based on Real-Time SCOPE requirements and standard options. The typical system includes a printer as the listable output device and a card reader as input device; neither is required by Real-Time SCOPE. The configuration shown in figure 2-1 also includes extra peripheral devices and channels for the priority program.

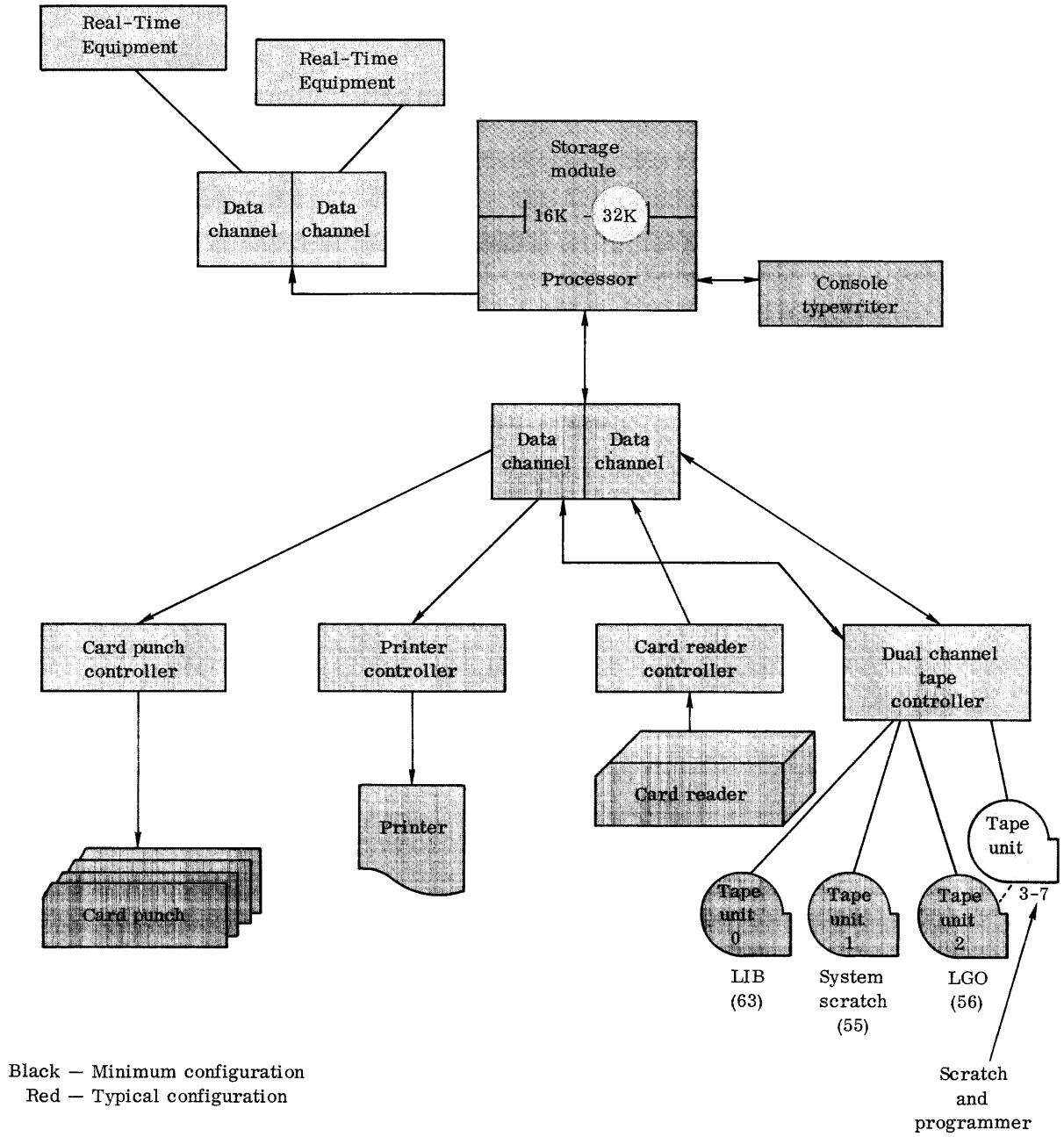


Figure 2.1 Configuration

2.2 EQUIPMENT DESIGNATION

Each input/output unit in a machine configuration is defined by a hardware connect code consisting of hardware type, channel number 0-7, equipment (or controller) number 0-7, and unit number 00-77. Although connect codes for physical units are the equipment designation with which the operator is most familiar, the programmer seldom uses them except for designating real-time equipment. Logical units are used by the programmer to designate standard peripheral units.

Unit Number	Mnemonic	Application	Description
56	LGO	Load and go	Output unit for compilers and assemblers; receives object programs ready for loading and execution.
57	ACC	Accounting record	May contain accounting records.
58	CFO	Comment-from-operator	Transmits messages from the operator to Real-Time SCOPE, the priority program, and the batch program.
59	CTO	Comment-to-operator	Transmits messages to the operator from Real-Time SCOPE, the priority program, and the batch program.
60	INP	Standard input	The source of Real-Time SCOPE control statements that define stacked jobs and of the programs and data (BATCH) when they accompany the control statements.
61	OUT	Standard output	Receives listable output from Real-Time SCOPE and batch programs.
62	PUN	Standard punch	Receives punch output primarily from RTS and batch programs.
63	LIB	Library	Contains the monitor system and all standard programs and subroutines (e.g. COMPASS, FORTRAN) that operate under Real-Time SCOPE. Control cards direct the loading of a priority program from LIB. During execution, priority programs cannot reference the library unit.

When an EQUIP or LED card for the load-and-go unit, 56, specifies magnetic tape but no special unit, Real-Time SCOPE assigns tape unit CcEeU02. When an EQUIP or LED card for scratch unit 55 specifies magnetic tape but no special unit, RTS assigns tape unit CcEeU01. The assignment is not recorded on the CTO nor is the operator asked READY? The assignment holds for the entire batch run. When loading from the LGO unit has been completed, a loaded batch program can use 56 as a scratch unit.

When Real-Time SCOPE operation is initiated, the following system units are automatically equipped to physical units.

System Unit Number	Mnemonic	Application	Hardware Type
58	CFO	Comment-from-operator	Console typewriter
59	CTO	Comment-to-operator	
60	INP	Standard input	Card reader or magnetic tape
61	OUT	Standard output	Printer or magnetic tape
62	PUN	Standard punch	Card punch or magnetic tape
63	LIB	Library	Magnetic tape

2.3 LOGICAL UNITS

When writing programs to be run under Real-Time SCOPE control, programmers use input/output designators called logical units. When using logical rather than physical units, the programmer need not know the connect codes or availability of particular physical units. However, he should know the quantity and types of units so that his program requirements do not exceed the configuration.

Efficient Real-Time SCOPE operation requires familiarity with the concept of logical unit assignments. In addition to being a convenience for the programmer, logical units increase the flexibility of peripheral equipment assignments. For example, magnetic tapes can be substituted for printers, punches, and card readers. The substitution unit must be able to perform a corresponding task.

Logical units, numbered from 1 through 63, are divided into three basic application oriented classifications.

Class	Batch Logical Units	Priority Logical Units
Programmer	1-49	1-49
Scratch	50-55	----
System	56-63	58-59

2.3.1 SYSTEM UNITS

Real-Time SCOPE reserves logical units 56-63 for the system; they must be assigned by unit number. These units, except for 58-59, can be used only by batch programs and RTS. The Real-Time SCOPE loader can load batch and priority jobs from the standard input unit (60), the library unit (63), and any other units named by the user. A protection scheme (section 5.5) prohibits operations that would reposition a standard system unit or destroy information on it. Only physical units with resident drivers can be declared as standard units; only physical units having resident drivers may be declared as priority programmer units.

2.3.2 PROGRAMMER UNITS

The programmer uses logical units 1-49. There are two sets of unit numbers: 49 for batch applications and 49 for priority applications. These units are always rewound after job completion.

2.3.3 SCRATCH UNITS

Batch programs also have access to logical units 50 through 55, designated by Real-Time SCOPE as scratch units. These units are rewound and released at run end. In a subsequent run, the physical units that served as 50-54 can be reassigned; when they are, Real-Time SCOPE asks READY? If the operator was directed to save the information on 50 through 54, he can do so before readying the units for new assignments.

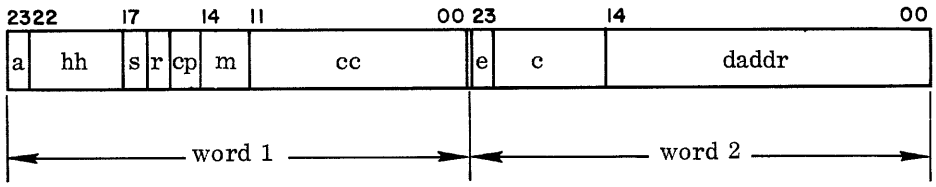
Most standard programs (FORTRAN, COMPASS, COBOL, etc.) use logical unit 55 as a scratch unit. Real-Time SCOPE assigns logical unit 55 to tape unit CcEeU01 unless the programmer requests some other unit. When such programs are run, 55 is recorded on the CTO, and the operator is asked READY? If tape unit CcEeU01 is equipped before execution begins, the program is terminated with a destructive error diagnostic (appendix H).

2.4 EQUIPMENT ASSIGNMENT

Each logical unit used by a program must be assigned a physical unit before execution begins. Real-Time SCOPE assigns physical units to logical units as directed by its own logic, by LED cards in standard programs (section 8.3.1), by programmer and operator EQUIP statements (section 7.2.7 and 3100/3200/3300/3500 Real-Time SCOPE Operator's Manual Pub. No. 60170200), and by XFER statements (section 7.2.9). RTS keeps internal records of these assignments in the Available Equipment Table (AET), the Running Hardware Table (RHT), and the Priority Running Hardware Table (BRHT).

2.4.1 AET

The Available Equipment Table (AET) defines the equipment available to the system. Depending upon the equipment configuration, the AET consists of up to 50 two-word entries. The unit location in the table (1 to 50) is the unit ordinal. Each entry has the following 24-bit word format:



Word	Bit	Field	Significance
1	23	a	0 Unit not assigned to a logical unit 1 Unit assigned to a logical unit
	22-18	hh	Numeric code 1-37 designates hardware type
	17	s	0 Unit operable 1 Unit inoperable
	16	r	0 Not reserved 1 Reserved for another computer
	15	cp	0 No transmission parity error detected 1 Transmission parity error occurred
	14	m	0 Batch interrupt 1 Priority interrupt
	11-00	cc	12-bit connect code for each unit; the hardware code used by RTS in the I/O instruction
2	23	e	0 No action 1 Unit of this hardware type to be assigned
	22-15	c	8-bit channel code; each set bit from 22 through 15 corresponds to a channel (0 through 7, respectively) available to this unit; all zeros indicates no channels available for the unit
	14-00	daddr	0 Nonresident driver ≠0 Driver address for the hardware type

Code	Type	Code	Type
01	Magnetic Tape (MT)	10	Typewriter Station (TS)
02	Card Reader (CR)	11	Plotter (PL)
03	Printer (PR)	12	Satellite Controller (SL)
04	Card Punch (CP)	13	Disk Pack Controller (DP)
05	Console Typewriter (TY)	14	Disk File (DF)
		15	Drum (DR)
06	Paper Tape Reader (TR)	16	Optical Character Reader (OR)
07	Paper Tape Punch (TP)	17-37	Unassigned

By using an AET control statement, the operator can obtain information from the AET or alter its contents (Real-Time SCOPE Operator's Manual). The entries are typed in octal and must be converted to binary (one octal digit = three bits) before a table entry can be interpreted.

With one exception, physical units cannot be shared between priority and batch programs; for logical units 58 and 59 (CFO and CTO), entries in the RHT and BRHT both use the same AET entry — the console typewriter — for comments to or from the operator. The assignment is permanent.

2.4.2 RHT

The Running Hardware Table (RHT) establishes correspondence between physical and logical units used by batch programs. The relative position of each entry (1 through 63) corresponds to its logical unit number. Each entry is composed of a 2-octal digit unit ordinal that refers to the position in the AET and Unit Status Table (UST) of the physical unit. Character position 0 contains the system protect flag. The physical unit is identified in the AET by its connect code. When the RHT entry is zero or illegal (exceeds number of AET entries), the logical unit is unassigned and cannot be used by a program.

2.4.3 BRHT

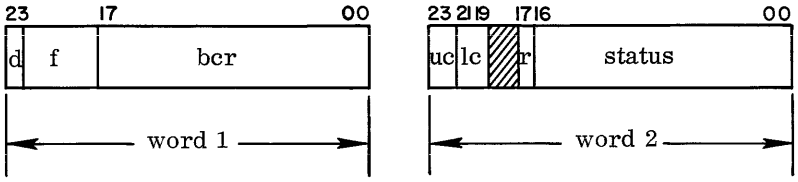
The Priority Running Hardware Table (BRHT) establishes correspondence between physical units and logical units used by priority programs. The relative positions of physical units 1-49, 58, and 59 correspond to the logical unit numbers. Each entry is composed of two octal digits (the unit ordinal) that refer to the position of the physical unit in the AET and the UST. The physical unit is identified in the AET by its connect code. When the BRHT entry is zero or illegal (exceeds number of AET entries), the logical unit is unassigned and cannot be used by the priority program. Character position 0 is a dummy character used as a dump indicator. If it is set to 0, a dump is indicated; otherwise, there is no dump.

2.4.4 RHTD

Running Hardware Table D (RHTD) duplicates the RHT or BRHT in size. When a programmer (using an EQUIP statement or LED card) assigns a logical $unit_i$ to a hardware type but not to a specific device, the request is stored until all requests for specific devices are processed. RTS sets the RHT (or BRHT) entry for u_i to 77_8 (63_{10}) and sets the RHTD entry for u_i to the code for the requested hardware type. After specific device requests are processed, RTS uses the corresponding RHTD entries to select an AET ordinal for each RHT/BRHT entry of 77_8 and sets the RHT/BRHT entry to the corresponding AET ordinal.

2.4.5 UST

The Unit Status Table (UST) consists of up to 50 two-word entries containing unit and channel status information. The position of a unit in the table is its UST ordinal, which is identical to its AET ordinal.



Word	Bit	Field	Significance	
1	23	d	0	Unit is dynamic; a data transfer or control function has been initiated; completion is undetermined
			1	Unit is static; last function is completed
	22-17	f		Last function (other than STATUS) initiated on unit
	16-00	bcr		Character address from buffer control register of current or last read/write operation
2	23,22	uc		Unit/channel status (see Status, Section 5.4.4)
	21,20	lc		Channel last used
	18			Unused
	17	r	0	Controller not reserved by another channel
			1	Controller reserved by another channel
	16-00	status		Status replies received from I/O equipment (status interpretation is given in section 5.4.4)

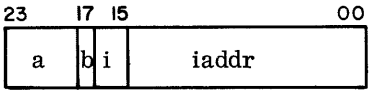
2.4.6 CST

Each of the eight entries in the Channel Status Table (CST) maintains the busy and interrupt status conditions for a particular channel. The position of a channel in the table is its channel number (i.e., first position is channel number 00).

Bit	Field	Significance	
23-18	a	00	No unit connected
		01-50	AET/UST ordinal of hardware unit last connected to channel
17	b	0	Channel not busy
		1	Channel busy; a selected internal interrupt must be recognized before channel can be freed
16-06		zero	Unused
05-00	e		EST ordinal of connected equipment/controller

2.4.7 EST

The Equipment Status Table (EST) has up to 64 one-word entries corresponding to 64 possible controller-channel combinations. The relative location of an equipment in the table is the EST ordinal of that unit.



Bit	Field	Significance	
23-18	a		AET/UST ordinal of last hardware unit connected to this equipment
17	b	0	Equipment not busy; no external interrupt selected
		1	Equipment busy; selected external interrupt must be processed before equipment can be freed
16,15	i	00	No interrupt requested by user (I/O parameters, section 5.6)
		01	User receives control on abnormal termination
		02,03	User receives control at end-of-operation whether normal or abnormal
14-00	iaddr		Address at which user receives control following interrupt (if requested)

LOADING AND MEMORY ALLOCATION 3

Real-Time SCOPE divides computer core memory into system memory and memory available to user programs. System memory consists of resident and variable resident routines.

The operator autoloads resident into memory. Resident calls variable resident routines into memory through RDCKF1. The Real-Time SCOPE loader loads relocatable user programs into available memory.

When loading relocatable user programs, Real-Time SCOPE allocates available memory first to the priority program, if there is one, and then to the batch program.

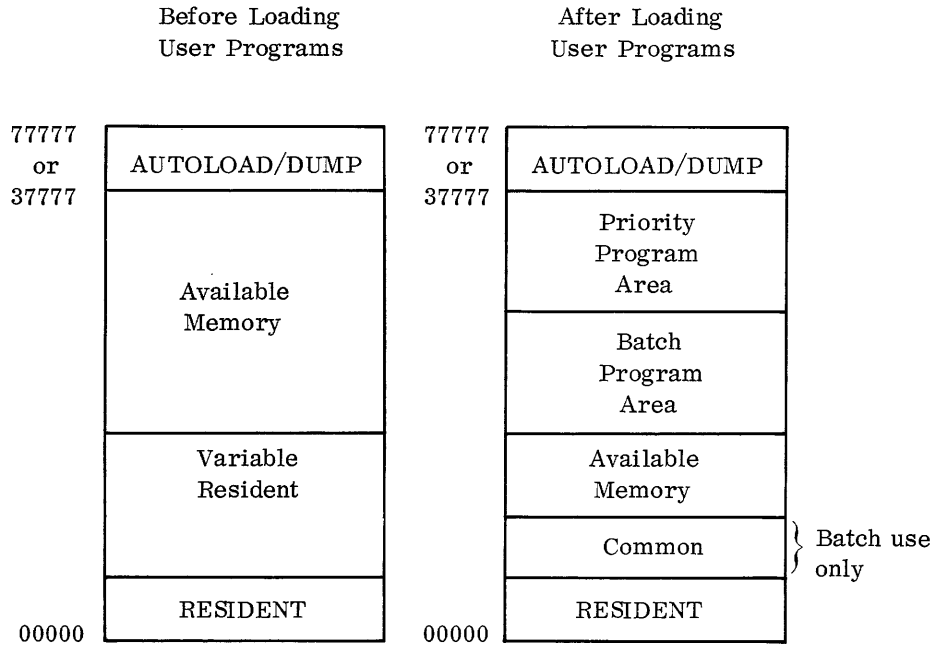


Figure 3.1. Core Memory

3.1 LOADER

The Real-Time SCOPE loader (OVPRO):

- Loads relocatable binary object programs into memory
- Links independently compiled subprograms that reference each other through symbolic external references to named entry points
- Automatically loads and links into a program any externally referenced library routines
- Provides optional operator intervention for standard error recovery on reader, printer, and magnetic tape devices
- Determines from programmer equipment statements whether program requires a nonresident driver and, if so, loads it from the library
- Assigns equipment to logical units as specified on control statements
- Prepares overlays and segments
- Determines requirements for software implementation of BCD and FDP instructions, and loads and links the required routines
- Records violations of established formats and procedures noted during loading

Real-Time SCOPE calls the loader from File 2 of the library once for each batch and priority run. To increase system efficiency by minimizing library tape rewinding, the library includes several copies of the loader.

The following control statements cause the loader to be brought into core.

- Library name
- LOAD
- IDC loader card (if not preceded by LOAD)
- CALL (operator statement)

The loader can also be called by a batch program. Programs are loaded from the standard input unit or programmer-specified units into memory in accordance with the Memory Limits Table (section 3.4). Each subprogram is loaded immediately below the last; the first one loaded is assigned the highest-numbered available memory address.

For library programs recorded in absolute binary, such as the COMPASS assembler and the FORTRAN compiler, Real-Time SCOPE loads a control program which then loads the absolute-format library program.

3.2 SUBPROGRAM ELEMENTS

The loader interprets the control cards described in chapter 8 and allocates memory accordingly. It recognizes the following subprogram elements.

Entry points

External names

Data block

Common block

3.2.1 ENTRY POINTS

If a location in a subprogram is to be used by another subprogram, the label must be declared an entry point of the subprogram. The programmer declares entry points through compiler or assembler statements, or through entry point name (EPT) cards (section 9.2.2). A subprogram may have any number of entry points.

An entry point consists of a label and an address. The one- to eight-character label is supplied by the programmer; the first character must be alphabetic.

3.2.2 EXTERNAL NAMES

A subprogram which refers to an entry point in another subprogram must declare the referenced point an external name. The programmer declares external names through compiler or assembler statements or through external name (XNL) cards (section 9.2.4).

The loader links subprograms by using the external name. When the referenced entry point is in a library program, the loader calls the referenced subprogram from the library and links the subprograms. The loader does not link batch to priority programs or priority to batch programs.

If a subprogram declares as external a name which is not a system entry point, a library entry point, or a subprogram entry point, Real-Time SCOPE issues a diagnostic (appendix H).

3.2.3 DATA BLOCKS

The data block for a priority or batch program is an area of memory which may be shared by subprograms and may be preloaded with data. The loader assigns memory to the data block the first time it is encountered; thereafter, the declared length must not exceed the assigned length. When a subprogram does not refer to the data block, there is none. The data block is defined by an IDC card (section 9.2.1).

3.2.4 COMMON BLOCK

The batch program may use a common block which is an area of memory that may be shared by subprograms. It is assigned to the lowest-numbered available memory, which may be occupied by the loader and its tables during loading; therefore, this block cannot be preloaded with data. The declared length of a common block varies from one subprogram to another; it is restricted only by the amount of memory available. A priority program cannot use a common block. The common block is defined by an IDC card (section 9.2.1).

3.3 RELOCATABILITY

Addresses assigned by the compilers and assemblers are relocatable. They do not identify actual addresses in the computer memory but are used as sequence numbers or reference points. Relocatable addresses are relative to the beginning of the subprogram.

The programmer coding in symbolic language need not be concerned with absolute memory locations. He organizes the program and dictates certain relationships between instructions and data. The compiler and assemblers record the relationships and produce relocatable object subprograms. In a relocatable subprogram, the first location is given a sequence number or starting location of zero.

Successive instructions or data words in the subprogram are assigned successive addresses. When the same increment or decrement is applied to these relocatable addresses or to the sequence number, they may be assigned to any memory location during loading. It is important to retain the relationships established (or detected) during assembly or compilation. By treating subprograms as unified blocks and maintaining a constant interval between the internal parts, it is possible to leave the ultimate assignment of memory to the loader. Entry points and external symbols allow the loader to establish the correct linkage between subprograms.

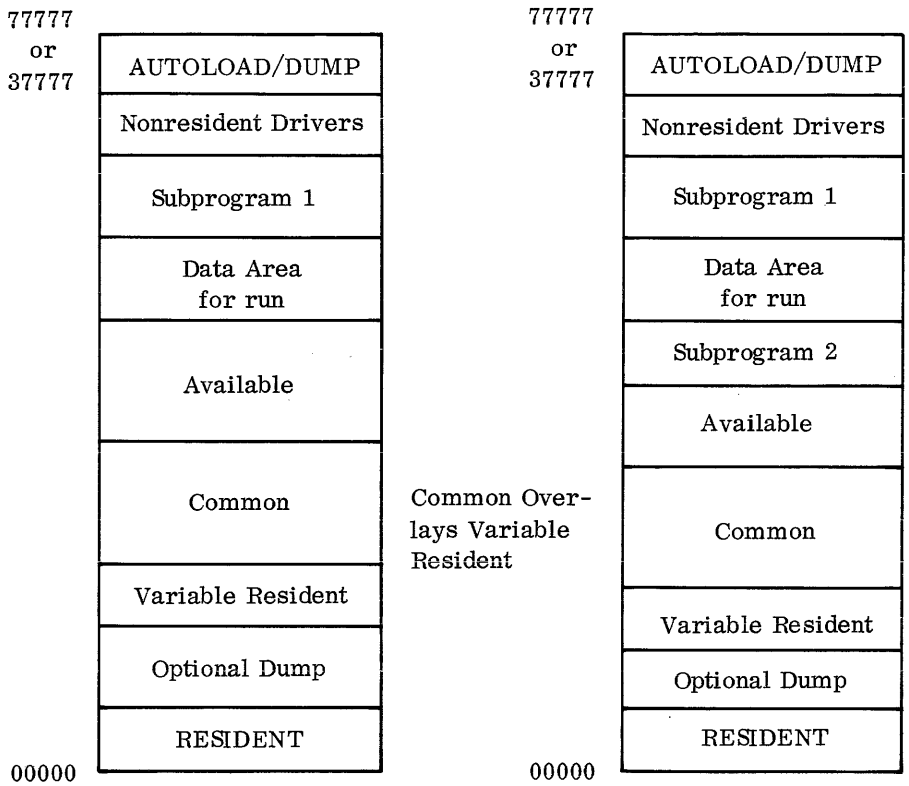


Figure 3.2. Relocatability

Data and common storage are treated similar to subprogram storage. The first location is assumed zero by the assemblers and compilers and subsequent areas are addressed relatively. The loader relocates the addresses; the assemblers and compilers provide a separate relocation factor for each object program area.

3.3.1 RELOCATION OF SUBPROGRAMS

The assembler or compiler determines which relocation factor (subprogram, common, or data) is to be applied during loading. The loader determines which relocation factor to use by word count, or for RIF cards (section 9.2.3), by one of the following relocation bytes.

Subprogram increment

Common area increment

Data area increment

Subprogram decrement

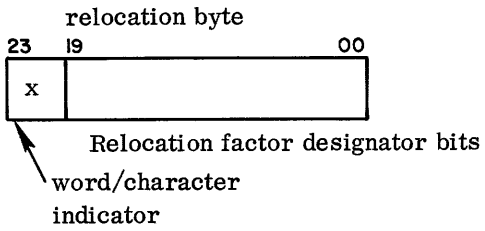
Common area decrement

Data area decrement

Extension area increment (x, specified by the programmer)

3.3.2 RELOCATION BYTES

The relocation byte on the RIF card (section 9.2.3) determines whether the address in the word on the card is to be incremented or decremented, which area is involved, and whether to perform 15-bit or 17-bit arithmetic on the address. A relocation byte (4 bits) on the RIF card has the following form.



Bit	Field	Significance
23-19	x	0 15-bit arithmetic 1 17-bit arithmetic
19-00		Relocation factor designator bits

Relocation Byte	Relocation Factor
x000	Not used; this code constitutes an error
x001	No relocation (absolute)
x010	Subprogram increment
x011	Common block increment
x100	Data block increment
x101	Subprogram decrement
x110	Common block decrement
x111	Data block decrement

Relocation bytes are discussed in further detail under RIF card, section 9.2.3.

3.4 MEMORY LIMITS TABLE

The loader refers to and updates the memory limits table when placing programs in core. The programmer may refer to this table by declaring MEMORY an external name in his program.

The table has the following entries.

MEMBASE	Constant address specifying end of resident
MEMLIMIT	Constant address during batch processing specifying the highest location available for batch loading
MEMTOP	Address during batch processing that specifies the first available location for a batch program; less than MEMLIMIT when RTS loads a nonresident driver; otherwise, equal to MEMLIMIT
MEMORY	Lowest numbered address of available memory
MEMORYE	Highest numbered address of available memory
BUPMEM	Upper limit for priority program loading; equal to highest address of all available memory
BLOWMEM	Lower limit of a loaded priority program; equal to MEMLIMIT+1

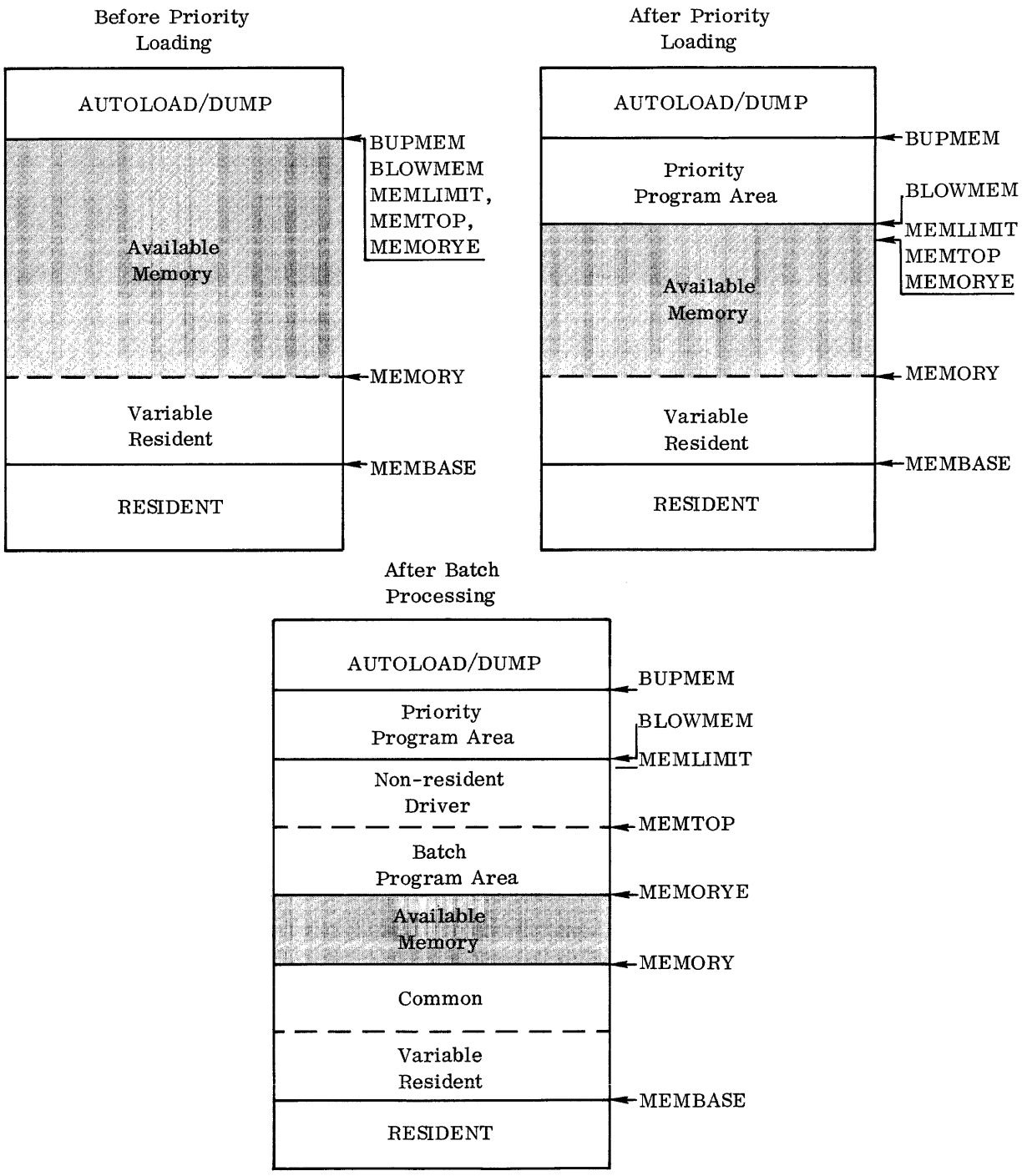


Figure 3-3. Memory Limits Table Addresses

3.5 PRIORITY PROGRAM AREA

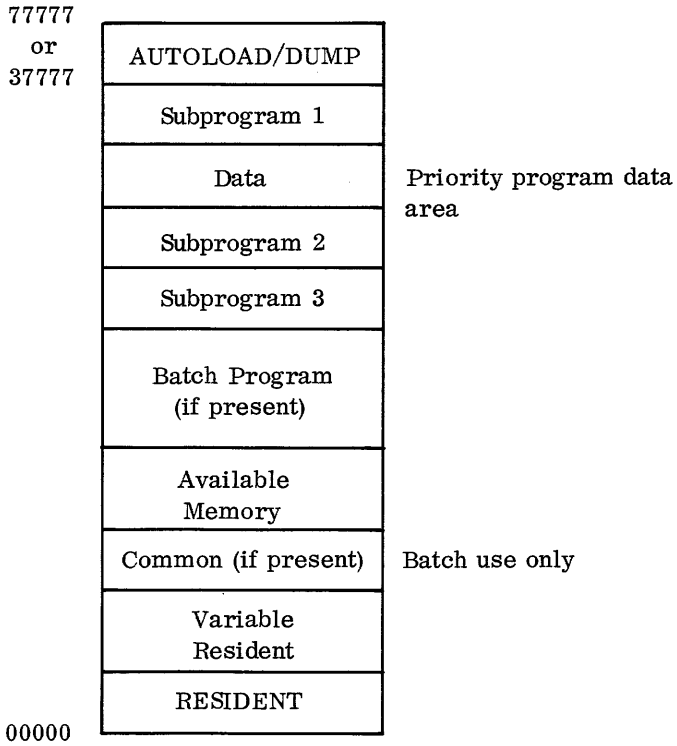
A BACK control statement following a SEQUENCE statement on INP directs Real-Time SCOPE to load the next program as a priority program. This program may consist of subprograms and one data area.

3.5.1 PRIORITY DATA AREA

A priority subprogram may specify a data area which may be preset at load time with instructions or constants.

Every priority subprogram may reference this data area. The maximum length of the priority data area is specified by the first data-area-defining priority subprogram. An error message results if a subsequent priority subprogram attempts to define a larger data area.

The priority data area is located between the subprogram that defines it and the next subprogram.



3.5.2 RESTRICTIONS ON PRIORITY USE OF MEMORY

Priority programs may not use a common area or a program extension area. Priority programs must use drivers which are in resident or which are contained in the priority program (chapter 7).

3.6 BATCH PROGRAM AREA

A JOB card following a SEQUENCE card or following a priority program directs Real-Time SCOPE to load the next program as a batch program in available memory.

Batch programs may consist of subprograms, a data area, and a common area. Before loading a batch program, RTS checks logical unit assignments, and if the program requires a nonresident driver, RTS loads the driver into available memory.

After a batch program is loaded, octal correction (OCC) cards (section 8.2.18) on INP may direct the postload processor to add a program extension area to the batch program.

3.6.1 BATCH DATA AREA

The batch data area resembles the priority data area. The user may preset the area and all batch subprograms may use it. The size of the batch data area is fixed by the first batch subprogram that defines a data area. Definition of a larger area by a subsequent batch subprogram results in an error message.

When presetting the batch data area, the programmer may use the COMPASS pseudo instructions ORGR and BSS (3100/3200/3300/3500 COMPASS Reference Manual, Pub. No. 60236800) to guarantee the proper placement of information in the data area for reference by all batch subprograms.

Example of data area definition:

Two subprograms, BAKER and GEORGE, are to be loaded and executed at the same time. The data area may be organized as follows:

```
LOCATION  OPERATION, MODIFIERS  ADDRESS FIELD  COMMENTS
1      8  10                   20             41
          IDENT                BAKER
           .
           .
           .
          DATA
          BSS                  7              (space reserved for BAKER and GEORGE)
          ORGR                 0
     A    OCT                  1              (preset 1st location)
     B    OCT                  2,3,4          (preset locations 2 through 4)
           .
           .
           .
          PRG
           .
           .
           .
          END
```

```
LOCATION  OPERATION, MODIFIERS  ADDRESS FIELD  COMMENTS
1      8  10                   20             41
          IDENT                GEORGE
           .
           .
           .
          DATA
          BSS                  7              (space reserved for BAKER and GEORGE)
          ORGR                 4              (start at 5th location)
     C    OCT                  5              (preset 5th location)
     D    OCT                  6,7            (preset locations 6 and 7)
           .
           .
           .
          PRG
           .
           .
           .
          END
```

The data area length declared for subprogram BAKER includes the area required by GEORGE.

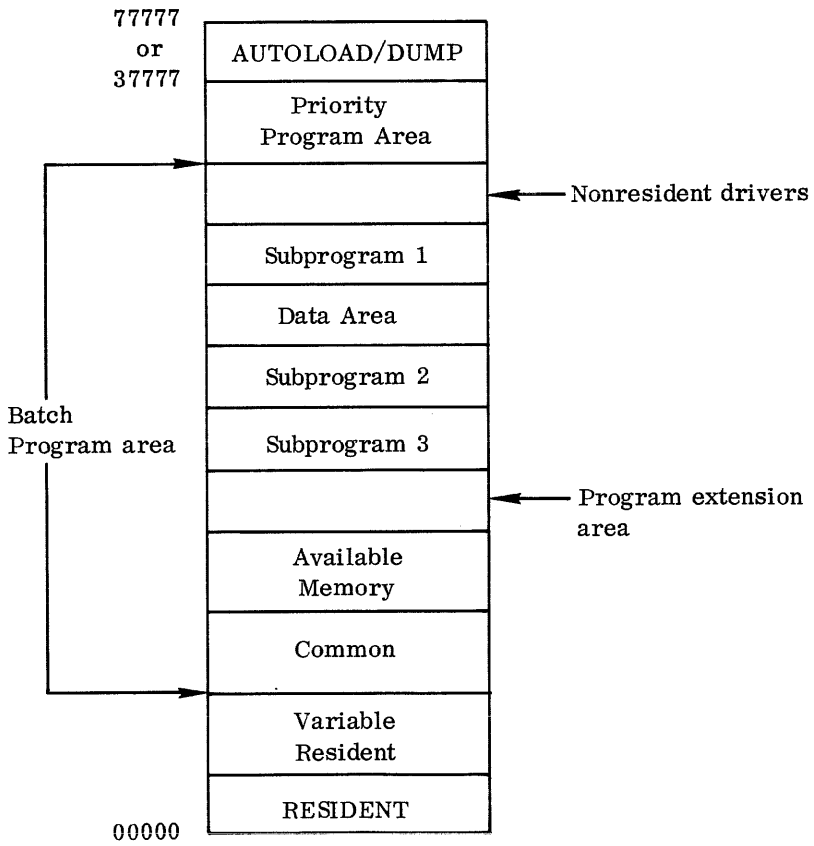
3.6.2 COMMON

Each batch subprogram may define a common area to be used by all batch subprograms. When batch loading is complete, the length of common is the greatest length specified by any batch subprogram.

During loading, the Real-Time SCOPE loader and loader symbol table occupy the area of available memory which common occupies when loading is complete. Thus, common may not be preset with data; an error message results if a job requests the loader to place information in common.

The programmer must ensure that information in common needed by one subprogram is not destroyed by another subprogram.

A map of the batch program area is:

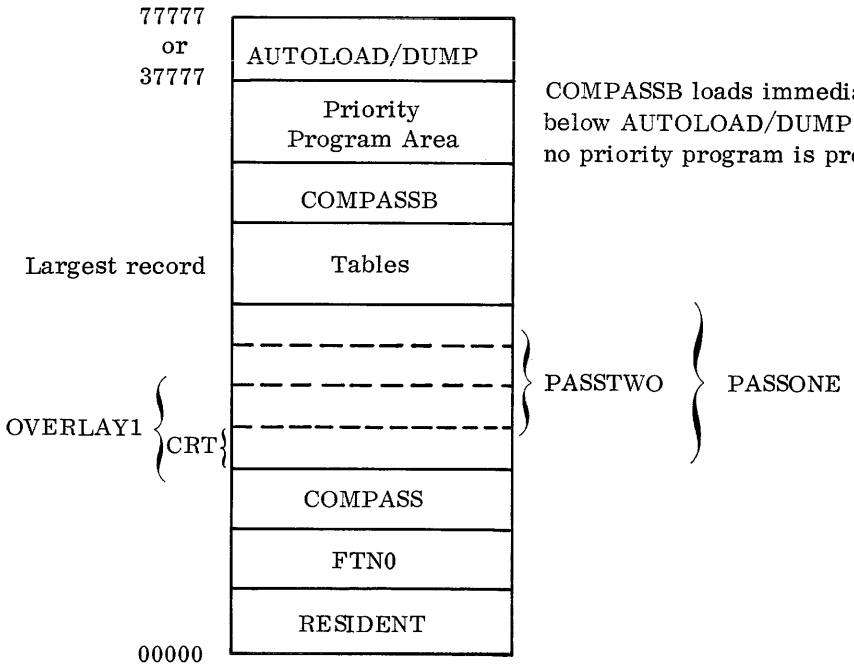


3.7 EXAMPLES OF MEMORY USAGE

3.7.1 COMPASS

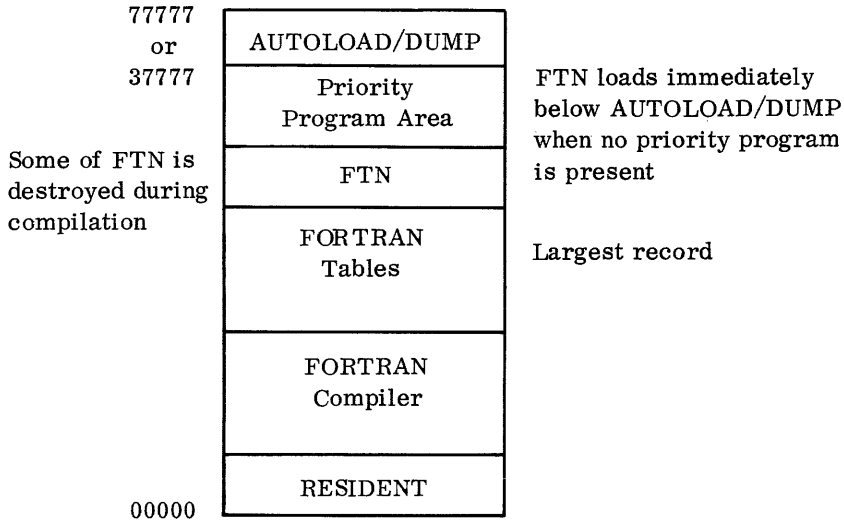
When COMPASS is called from the library by a COMPASS statement, Real-Time SCOPE loads a relocatable binary program, COMPASSB, into high available memory. COMPASSB causes COMPASS and FTN0 to be loaded into low memory through RDCKF1.

COMPASS then receives control and loads the remaining records (OVERLAY1, PASSONE, PASSTWO, and CRT) into low memory through RDCKF1 as required during assembly. The routine COMPASSB is destroyed by symbol tables. Control returns to Real-Time SCOPE through COMPASS.



3.7.2 FORTRAN

When the FORTRAN compiler is called by the FORTRAN statement, Real-Time SCOPE loads a relocatable binary control program, FTN, from the library into high memory. During compilation, the sections of this compiler (FTN0 through FTN6 and FTNE) are loaded into low memory by RDCKF1.



SYSTEM LIBRARY 4

The Real-Time SCOPE System Library Tape contains the control program, system language programs, input/output control programs, and specialized applications programs. It may be modified by the library preparation routine (PRELIB) to include programs developed specifically for an individual installation.

The Real-Time SCOPE control program includes the resident record (Resident) and several variable resident records. Resident is file one of the library tape; it is read in by the autoload program. The variable resident routines are included in file two. They are read in by RDCKF1, a Real-Time SCOPE resident subroutine, as needed. Variable resident routines are in absolute binary form.

The system languages and the remaining programs are contained in file two in relocatable and absolute binary form. The relocatable programs are read in as specified by LOADER, a variable resident executive File 2 routine.

4.1 CONTROL

The Real-Time SCOPE control program consists of resident and those portions of absolute binary records comprising the variable resident routines. RTS routes jobs through the proper sequence of operating routines, establishes I/O equipment environments, processes Real-Time SCOPE control statements, and provides for loading of user programs.

4.2 RESIDENT

Resident includes the following routines and tables which are always available for use by batch and priority programs. References to these resident routines and tables must be declared external by the using program.

- ZERO — Routine that defines linkage between EXEC and first 48 words of memory
- CIC — Central interrupt control routine
- CIO — Central input/output control routine
- DRIVERS — Drivers for standard system peripheral units
- MANUAL INTERRUPT — Routine for entering messages to Real-Time SCOPE, to batch programs, or to priority programs during system operation
- MANUAL INTERRUPT PROCESSORS — Processes of Real-Time SCOPE system messages received through the manual interrupt routine
- CONTROL(IRP) — Multipurpose resident routine that handles loading of variable resident, and routes control through the variable resident routines
- TABLES — Tables which may be referenced but not changed by users:

  AET Available equipment table
  RHT Running hardware table
  BRHT Priority running hardware table
  UST Unit status table
  EST Equipment status table
  CST Channel status table
  CIT Central interrupt table
  Accounts
  Memory limits
  Operating flags
  Transfer addresses
  Message table for manual interrupt
- RDCKF1 — Routine that reads and checks routines written in absolute binary form
- ABNORMAL — Routine that obtains control when a fatal error occurs

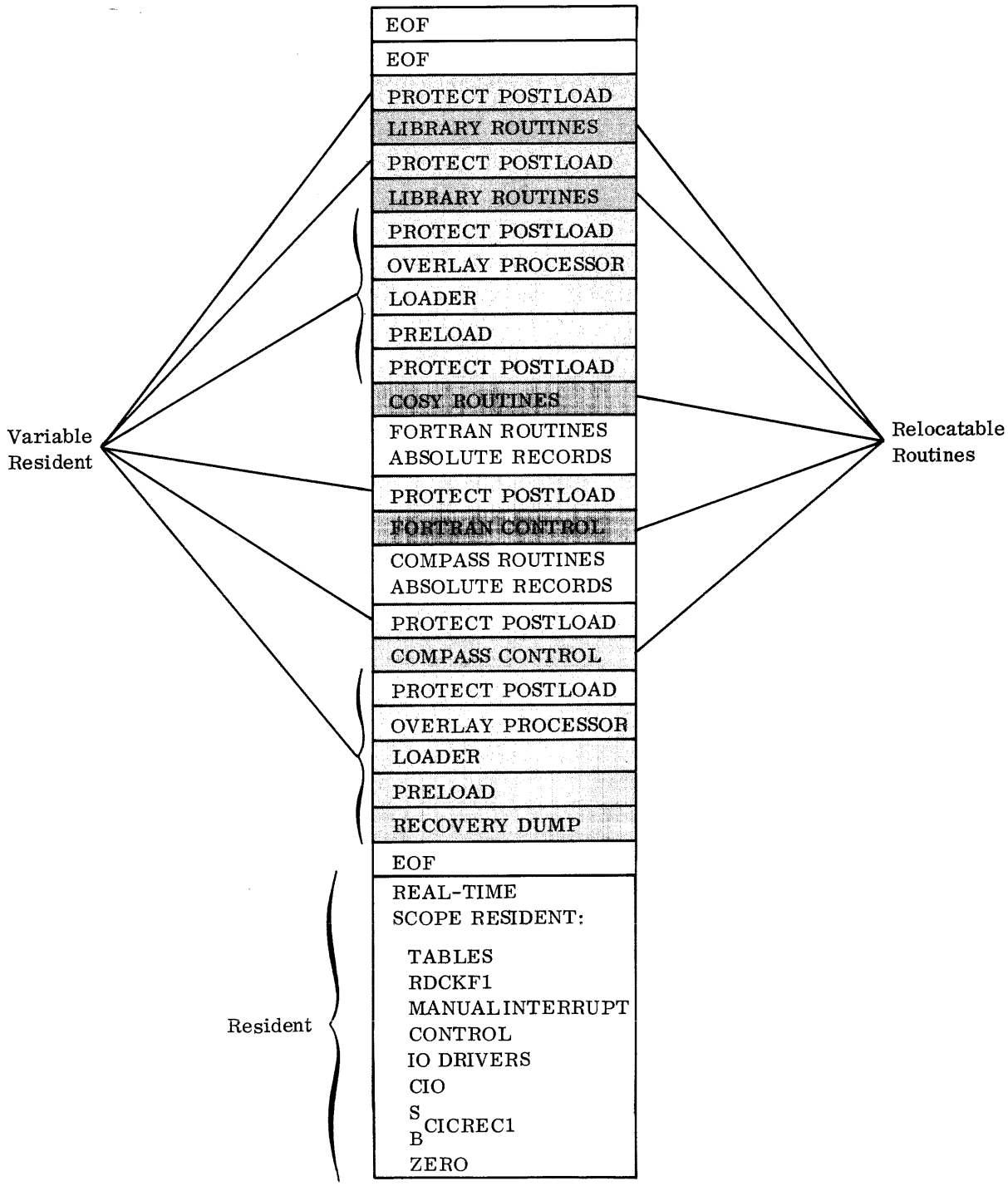
4.3 VARIABLE RESIDENT

Variable resident contains each record not in resident but which must be loaded beginning at a specific core location prior to use. These routines, when loaded, replace one another in core; the overlay structure is possible because the routines are needed only at particular points of processing.

The variable resident routines are:

RDUMP	Recovery dump routine which prints out the contents of console registers and of all batch or priority memory. The routine is executed when a program terminates abnormally or when the operator terminates the program through manual interrupt.
PRELOAD	Clears memory and releases scratch units in preparation for loading; processes control statements.
LOADER	Loads and links relocatable subprograms, library routines, and I/O drivers.
OVPRO	Overlay processor; loads and links overlays.
PROTECT	Prohibits destructive operations on standard system units.
POSTLOAD	Processes control statements concerned with postload operations and prepares Real-Time SCOPE for entry into user programs.

4.4 SAMPLE LIBRARY ROUTINES



4.5 LGO

The system load-and-go (LGO) tape must be assigned when it is needed. Unless the programmer specifies otherwise, logical unit 56 receives assembly and compilation load-and-go output. RTS rewinds the load-and-go tape when it reads a LOAD,56 card, when loading is completed from 56, and when a job terminates.

Using the appropriate logical unit, the user may refer to the system scratch tape and LGO in the following statements: EQUIP, LOAD, XFER, REWIND, UNLOAD (chapter 8).

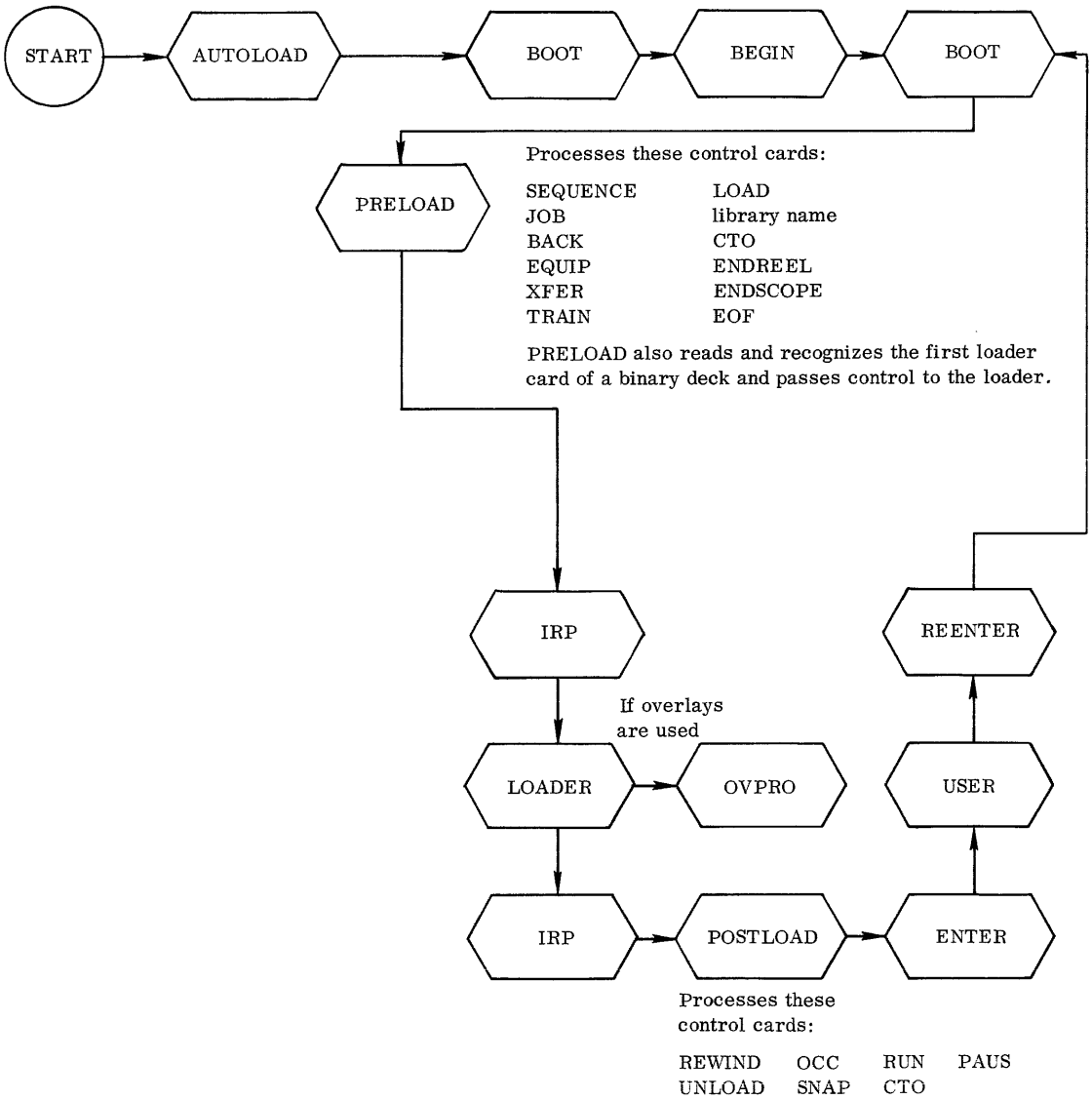
4.6 PRELIB

PRELIB allows the user to modify an existing Real-Time SCOPE system or to create a new system from binary card images.

At the user's discretion, a relocatable library routine may be updated, deleted, or inserted on the library. Resident may be altered, system entry points deleted or added, and variable resident (e.g., those COMPASS or FORTRAN programs requiring an absolute loading address) may be modified.

Frequently used variable resident routines can be duplicated on the tape to reduce system overhead during tape searches. Further information is in the 3100/3200/3300/3500 Real-Time SCOPE V2.0 Installation Handbook, Pub. No. 60237000.

4.7 FLOW OF CONTROL THROUGH RTS



INPUT/OUTPUT 5

The Real-Time SCOPE central input/output control routine (CIO) simplifies I/O processing on peripheral equipment. The user may call CIO for data transfer, control, status checking, and format request functions. In calls to CIO, the user refers to equipment by logical unit numbers. CIO operates in conjunction with drivers for each standard type of peripheral device and in conjunction with the PROTECT routine, which prohibits detrimental operations that would destroy information on the units or interfere with normal job sequencing on standard units. CIO makes use of the I/O tables (section 2.4).

A request for an I/O operation is typically a return jump to CIO. CIO interprets the legality of the logical unit number. If illegal, the job is terminated through ABNORMAL; if legal, CIO checks function/unit for violation of system protection. If function or unit is illegal, CIO terminates the run. If both are legal, control transfers to the proper driver routine. The driver determines if the operation is possible on the hardware type. If it is possible, the driver either prepares a list of functions for CIO to perform or performs the function itself; if not possible, control returns to CIO and the request is rejected. If CIO rejects any request, it transfers control to the user-supplied reject address.

5.1 REAL-TIME I/O

Because of the stringent time requirements imposed by real-time equipment, a real-time priority program usually includes a driver for the equipment. In this case, I/O requests are not routed through CIO. The user is responsible for hardware availability, hardware connection, setting flags in the central interrupt table, status checking, data transfer, etc.

Interrupts on real-time data channels are given priority handling with maximum interrupt recognition time of 300 microseconds, not including I/O. The user should not assign logical unit numbers to equipment directly driven by his program. Additionally, the user-driven equipment should not have an entry in the AET; if it does, the AET entry must be unavailable for system assignment (RTS Operator's Manual). User-driven equipment in RTS has real-time interrupt priority.

Real-time programs may call CIO for I/O functions on equipment not connected through real-time channels. Before calling CIO, however, the real-time program should call RIO, an input/output subroutine that tests CIO availability and is not used by the system. This permits a real-time program to use CIO for I/O operations. The real-time program cannot call CIO directly because CIO is not re-entrant, but may be interrupted to service a real-time program.

Calling sequence to RIO:

```
LOCATION  | OPERATION, MODIFIERS  ADDRESS FIELD                COMMENTS
1       8 | 10                    20                           41
----------+-------------------------------------------------------
p         | RTJ                   RIO
p+1       | AZJ,NE                Address of non-I/O subroutine
p+2       | RTJ                   Address of I/O subroutine
p+3       | Continue
          | real-time
          | program
```

The AZJ tests the contents of A as returned by RIO. If (A) is 0, the real-time user may call CIO. If (A) is not 0, RIO gives the real-time user I/O subroutine control when CIO completes its interrupt processing. In this case, the user must give up control as soon as possible to enable CIO to complete its previous assignment.

5.2 I/O TABLES

Input/output routines communicate through the following tables (section 2.4).

Available Equipment Table (AET)	Describes each piece of physical equipment
Running Hardware Table (RHT) (batch)	Correlates batch and system logical units with physical units
Priority Running Hardware Table (BRHT)	Correlates priority logical units with physical units
Unit Status Table (UST)	Maintains status information for each piece of equipment
Channel Status Table (CST)	Maintains channel activity information
Equipment Status Table (EST)	Maintains controller activity information

5.3 I/O FUNCTIONS

A user requesting CIO to perform an operation on a specified logical unit may use the function codes in the COMPASS calling sequence described in section 5.4. Compilers and the COMPASS I/O macros running under Real-Time SCOPE generate similar CIO calling sequences using the function codes. To determine if a function is possible or has an equivalent function on a particular hardware type, refer to the driver description in section 5.6.

Nonstandard drivers, including the optical character reader, the paper tape station, and the plotter are described in appendix D.

Code	Function
01	Read n words/characters starting at first word address
02	Write n words/characters starting from first word address
03	Read n words/characters backward and store backward starting at first word address plus (n-1)
04	Rewind specified unit
05	Rewind and unload specified unit
06	Backspace one record
07	Space forward past one end-of-file mark
10	Space backward past one end-of-file mark
11	Write or punch end-of-file or page eject
12	Skip bad spot on tape
13	Check and report unit status
14	Select format

5.4 CIO

The programmer may call CIO to perform the following types of functions on peripheral equipment.

Data transfer (read, write, read backward)

Control (rewind, unload, backspace, etc.)

Select format

Check status

The calls for these functions may be in one of the following forms.

COMPASS calling sequence; CIO must be declared external in the calling program

COMPASS library macro (called from library by LIBM cards)

Compiler input/output statement which generates a sequence similar to the COMPASS calling sequence

In calls to CIO, the user refers to peripheral equipment by logical unit numbers (section 2.3).

5.4.1 DATA TRANSFER

CIO processes a data transfer request as follows:

Selects channel and declares it busy (CST/b set to 1)

Selects internal interrupt

Declares unit dynamic (UST/d set to 0)

Selects various external interrupts

Declares equipment busy (EST/b set to 1)

When the channel interrupt is processed, the channel is declared not busy and the bcr and c_1 entries (see static status check) in the unit status table are updated. When the external interrupt occurs, the unit is declared static, the remainder of the UST is updated, and the equipment is declared not busy.

Calling sequence:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
p	RTJ	CIO	Call CIO
p+1	f	u,i	Parameters
p+2	jump	raddr	
p+3	m,C	fwaddr	
p+4	c	n	
p+5	normal	iaddr	
	return		
	if i = 0		
p+6	normal		
	return		Continue program
	if i ≠ 0		

Macro:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	READS	(u,raddr,fwaddr,n,i,iaddr,(m,C),c)	
	WRITES	(u,raddr,fwaddr,n,i,iaddr,(m,C),c)	
	READB	(u,raddr,fwaddr,n,i,iaddr,(m,C),c)	

Parameters include:

f	Input/output function code (see function for each driver)
u	Logical unit number
i	Interrupt selection 0 No interrupt 1 Interrupt only on abnormal end-of-operation 2,3 Interrupt at end-of-operation, normal or abnormal
jump	Nonselective jump, such as UJP or RTJ
raddr	Reject address; in calling sequence on a reject, control returns at p+2
m	Mode selection (see mode for each driver)
fwaddr	Address of first word or character in buffer for data transmission
n	Number of buffered words or characters to be transmitted
iaddr	When interrupt occurs, control transfers to interrupt address specified
,C	Required after mode selection when character addressing is selected
c	Character addressing flag 40_8 Character addressing selected 0 or blank Word addressing selected

5.4.2 CONTROL

Control functions require a channel only for initiation of the function. If the user selects interrupt, CIO processes a control function request as follows:

Declares unit dynamic in UST

Declares equipment busy in EST

At end-of-operation, if interrupt is selected CIC returns control to iaddr

If the user does not select interrupt, the unit will be declared dynamic but the equipment will not have busy status; thus, the user may initiate operations on various units connected to the same controller. Because he receives no indication of end-of-operation, the user must check status before using any of the units with an operation so initiated.

Calling sequence:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
p	RTJ	CIO	Call CIO
p+1	f	u,i	Parameters
p+2	jump	raddr	
p+3	normal return if i = 0	iaddr	
p+4	normal return if i ≠ 0		Continue program

Macro:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	REWIND	(u,raddr,i,iaddr)	
	UNLOAD	(u,raddr,i,iaddr)	
	BKSP	(u,raddr,i,iaddr)	
	SEFF	(u,raddr,i,iaddr)	
	SEFB	(u,raddr,i,iaddr)	
	WEOF	(u,raddr,i,iaddr)	
	ERASE	(u,raddr,i,iaddr)	

Parameters include:

f	Input/output function code (see function for each driver)	
u	Logical unit number	
i	0	No interrupt (always 0 in UNLOAD)
	1	Interrupt on abnormal end-of-operation
	2,3	Interrupt on end-of-operation, normal or abnormal
jump	UJP or RTJ	
raddr	Reject address; control returns at p+2 on a reject	
iaddr	Interrupt address; control transfers to iaddr on interrupt if i≠0	

5.4.3 FORMAT

A format request requires a channel for initiation of the function but selects no interrupts and leaves the requested unit nondynamic (UST/d = 1).

Calling sequence:

```
LOCATION  OPERATION, MODIFIERS  ADDRESS FIELD  COMMENTS
1      8  10                    20             41
p         RTJ                   CIO            Call CIO
p+1       14                    u,fm           ( FORMAT function code
p+2       jump                  raddr          ( and parameters
p+3       normal
          return                                 Continue program
```

Macro:

```
LOCATION  OPERATION, MODIFIERS  ADDRESS FIELD  COMMENTS
1      8  10                    20             41
          FORMAT                (u,raddr,fm)
```

Parameters include:

u	Logical unit number
fm	Format mode designator; listed with driver for each hardware type
jump	RTJ or UJP
raddr	Reject address

5.4.4 STATUS

The status function checks the state of a unit before, during, or after an I/O operation and returns an updated copy of the UST entry for the unit to the A and Q registers.

Calling sequence:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
p	RTJ	CIO	Call CIO
p+1	13	u,d	STATUS function code
p+2	jump	raddr	and parameters
p+3	normal return		continue program

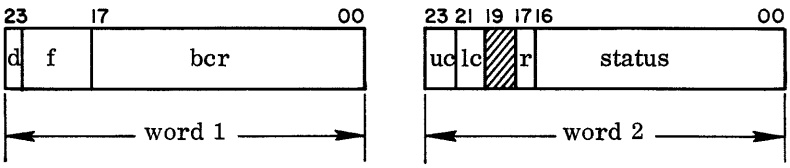
Macro:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	STATUS	(u,d)	

Parameters include:

u	Logical unit number	
d	0	Request static status check
	1	Request dynamic status check

Unit Status Table



Word	Bit	Field	Significance	
1	23	d	0	Unit is dynamic; a data transfer or control function has been initiated, completion is undetermined
			1	Unit is static; last function is completed
	22-17	f		Last function (other than STATUS) initiated on unit
	16-00	bcr		Character address from buffer control register of current or last read/write operation

Word	Bit	Field	Significance	
2	23,22	uc		Unit/channel status (see static status check)
	21-19	lc		Channel last used
	18			Unused
	17	r	0	Controller not reserved by another channel
			1	Controller reserved by another channel
	16-00	status		Status replies received from I/O equipment (see static status check)

Static Status Check

When, in the calling sequence, a user requests a static status check, the UST entry is updated before being placed in the A and Q registers.

1. If the unit is static (UST/d=1), channel availability is determined and the UST/u,c fields appear as follows:

u	c	Significance
0	0	A channel is free for immediate use
0	1	No channel is available

All other fields in the UST reflect the state of the unit at the end of the last operation.

2. If the unit is dynamic (UST/d=0), a previously initiated operation is not known to be complete. CIO determines channel and unit availability and updates the UST/u,c fields as follows:

u	c	Significance
0	0	Unit and channel immediately available
0	1	No channel is available; no further updating is possible
1	0	Unit busy or not ready

When a channel is available and the unit is connected, the d and status fields of the UST are updated to reflect the current status of the unit.

Dynamic Status Check

It is sometimes convenient to request updating the d and status fields even when the unit is static (UST/d=1). For example, all previous operations may be known to be complete, but the unit may not be ready. When parameter d in the STATUS calling sequence is set to 1, CIO attempts to update the d and ready/busy status fields as if the unit were dynamic.

5.4.5 CIO REJECT CONDITIONS

For any I/O request other than status to be initiated by CIO, all of the following conditions must be true.

- A channel is available to which the unit is or can be connected.
- The requested unit is physically ready and not logically busy.
- The requested function is logically defined on the requested unit (it is possible or simulated).
- No user interrupt request has been stacked for the unit.

When one of the above conditions is not met, CIO rejects the request (Error Recovery, Appendix G). It does so by updating the u, c fields of the UST, placing the second word of the UST entry in the Q-register, placing zero, or, if the function is illegal, the function code in the A-register, and transferring control to p+2 in the CIO calling sequence.

Channel Busy

A channel is busy and unavailable for any I/O request if CST/b is one in the associated CST entry (section 2.4.6). This bit is set to 1 when a channel interrupt is selected and is cleared when the interrupt on that channel is cleared.

CIO attempts to clear the busy status on a channel whenever that channel could be used to process a new request.

CIO controls the channels on which I/O functions are performed. It not only prevents any operation from destroying an operation still active on the channel but frees a channel as soon as possible. In addition, CIO accommodates multichannel hardware by having the capability of switching units from one channel to another.

Controller Busy

A controller is busy if a selected interrupt has not yet occurred in a unit to which it is connected.

Multichannel controllers are considered logically separate. A requested unit not reserved to a channel may be connected to another channel that is not busy.

Unit Busy or Not Ready

Unit Logically Busy: Once an operation is initiated on a unit, the unit remains logically busy until an external interrupt occurs, or if interrupt was not selected, until the unit is determined to be not busy by sensing the busy status line.

In both cases, sensing can be accomplished during processing of a new request on that unit or during processing of a request requiring the channel to which that unit is connected. Interrupts requiring linkage to a user procedure must physically occur external to CIO. Therefore, the interrupt will not be cleared.

Any function select, other than an interrupt select, will be rejected while the unit is busy. In this sense a unit is busy while a read/write operation is in progress as well as when other dynamic operations (such as rewind) are in progress.

Unit Not Ready: When a unit is not ready, operator action is required to make the unit ready. No new operation is initiated until the unit becomes ready.

Function Undefined

A function is undefined if:

No hardware operation corresponds to function requested (e.g., rewind of a card reader).

Unit is not ready for the function (e.g., write on a magnetic tape which does not contain a write ring).

Hardware externally rejects a function code processed from a string set up by an equipment driver procedure.

The hardware driver detects the first two error types. The third error type is detected when a function string is processed by CIO. If a hardware reject persists, CIO transfers control to the reject address in the user's calling sequence.

Stacking User Interrupt

Within an interrupt subroutine, either batch or priority, it is possible for other interrupts (chapter 6) to occur, including interrupts set up by the program in control. If such an interrupt occurs and the user has requested interrupt control, the request is stacked by CIC and the UST/uc is set to 11. CIO rejects requests for this unit until the interrupt has been unstacked.

Therefore, when making CIO calls from within an interrupt subroutine, it is necessary to test for this type of CIO reject. Unstacking occurs only after the interrupt subroutine has returned control to CIC.

5.4.6 CIO ABORT CONDITIONS

Classes of errors that cause abnormal job termination:

- Illegal call
- Illegal logical unit designation
- Rejected connect operation
- Rejected select operation
- Rejected data transfer operation

  } For internal rejects generated by CPU

- Request for prohibited function on protected unit

When one of the above conditions causes job termination, a message is printed on CTO and OUT (appendix H).

5.5 SYSTEM UNIT PROTECTION

The protect subroutine, called by CIO in processing I/O requests, prohibits the performance of any function which would destroy information on a standard system unit or would position the unit so as to interfere with normal job sequencing. This system unit protection is in effect unless the user includes the no protection parameter, NP, on his JOB control statement.

Protection applies to standard system units (logical units 58-63) and to any logical unit equated to a standard system unit.

The following functions are prohibited.

Unit	Protection
57 (ACC)	No protection defined.
58 (CFO)	Only READ, WRITE, and STATUS permitted; interrupt may not be selected in the CIO call.
59 (CTO)	Only READ, WRITE, and STATUS permitted; interrupt may not be selected in the CIO call.
60 (INP)	Requests to position the unit outside the file currently being processed or to destroy information on current file are prohibited. When the end-of-file status line is on, the following conditions are true:

Last Function Performed	Function Permitted		
	READ	READB	BKSP
READ	no	yes	yes
READB	yes	no	yes
BKSP	no	no	no

61 (OUT) and 62 (PUN)	REWIND, UNLOAD, SEFF, READ, READB, and WEOF are prohibited. Requests to position units (if tape) to overlap output from a previous job and requests to advance a tape beyond current information are prohibited. BKSP is allowed only if the previous function was WRITE or ERASE.
63 (LIB)	WRITE, UNLOAD, WEOF, and ERASE are prohibited.

5.6 DRIVERS

A typical configuration for Real-Time SCOPE may include the following standard drivers.

DRIVER01	Magnetic tape
DRIVER02	Card reader
DRIVER03	Printer
DRIVER04	Card punch
DRIVER05	Console typewriter

Other drivers are discussed in appendix D.

When a function is requested on a unit, CIO obtains the driver address for the equipment from word 2 of the AET and transfers control to the driver.

A driver routine for a particular hardware type interprets function requests and either executes the function or prepares a list of I/O operations for CIO to perform on the requested unit.

The driver first checks the legality of the function code in the user's calling sequence to determine if the function can be performed on the specified hardware (e.g., unless PROTECT is called, logical unit 62 (PUN) can be rewound if an EQUIP, 62=MT statement is used). If it cannot, the driver returns control to CIO which rejects the request.

If the function can be performed, the driver:

Edits the status field of the unit's UST entry according to hardware type.

Forms a list of hardware function codes to be selected on the unit.

5.6.1 MAGNETIC TAPE

DRIVER01 3228 or 3229/603, 606†, or 3127/601

Operation Performed	Permissible Function Code	Calling Sequence	Macro Name	Logical Unit (u)						
				1-57	58 CFO	59 CTO	60 INP	61 OUT	62 PUN	63 LIB
Read/record	01	data transfer	READS	yes			yes			yes
Write/record (1)	02	data transfer	WRITES	yes				yes	yes	
Rewind	04	control	REWIND	yes						
Rewind & unload	05	control	UNLOAD	yes						
Backspace record (2)	06	control	BKSP	yes			yes	yes	yes	yes
Space forward past EOF	07	control	SEFF	yes						yes
Backspace past EOF (2)	10	control	SEFB	yes						yes
Write EOF (1)	11	control	WEOF	yes						
Erase 6" tape (1)	12	control	ERASE	yes				yes	yes	
Check status	13	status	STATUS	yes			yes	yes	yes	yes
Select mode (3)	14	format	FORMAT	yes			yes	yes	yes	yes
Select density (3)	14	format	FORMAT	yes						

† CONTROL DATA® 3228 and 3229 Magnetic Tape Controllers
CONTROL DATA® 603 and 606 Magnetic Tape Transports

1. WRITE(02), WEOF(11), and ERASE(12) are rejected if write enable (status bit 02) is not set.

2. BKSP(06) and SEFB(10) are rejected when tape is at load point.

3. FORMAT mode/density codes:

0	Illegal
1	BCD mode
2	Binary mode
3	200 bpi (LO)
4	556 bpi (MED)
5,6,7	Illegal

DRIVER01 3228 or 3229/604, or 607, or 608†

Operation Performed	Permissible Function Code	Calling Sequence	Macro Name	Logical Unit (u)						
				1-57	58 CFO	59 CTO	60 INP	61 OUT	62 PUN	63 LIB
Read 1 record	01	data transfer	READS	yes			yes			yes
Write record (1)	02	data transfer	WRITES	yes				yes	yes	
Reverse read 1 record	03	control	READB	yes			yes			yes
Rewind	04	control	REWIND	yes						
Rewind & Unload	05	control	UNLOAD	yes						
Backspace (2)(3) 1 record	06	control	BKSP	yes			yes	yes	yes	yes
Space forward past EOF	07	control	SEFF	yes						yes
Backspace (2) past end EOF	10	control	SEFB	yes						yes
Write EOF (1)	11	control	WEOF	yes						
Erase 6" tape (1)	12	control	ERASE	yes				yes	yes	
Check status	13	status	STATUS	yes			yes	yes	yes	yes
Select mode (4)	14	format	FORMAT	yes			yes	yes	yes	yes
Select density	14	format	FORMAT	yes						

† CONTROL DATA® 604 and 607 Magnetic Tape Transports

1. WRITE(02), WEOF(11), and ERASE(12) are rejected if write enable (status bit 02) is not set.

2. BKSP(06) and SEFB(10) are rejected when tape is at load point.

3. If last function was READB(03) or FORMAT(14) code 6 reverse read, space forward one record.

4. FORMAT mode/density codes:

    0 Illegal
    1 BCD mode
    2 Binary mode
    3 200 bpi (LO)
    4 556 bpi (MED)
    5 800 bpi (HI); 607 and 608 only
    6 Reverse read (see below)
    7 Clear reverse read

Read Reverse: The capability to read backward on the 607 tape units is provided by two control functions: select reverse read and clear reverse read. The selection of reverse reading affects only two tape operations — read and backspace. The driver for the units performs as follows: On a READB request, reverse motion is selected. On any other request, except STATUS, BKSP, or possibly FORMAT, the selection of reverse motion is cleared.

FORMAT selections may be made to select or clear reverse motion independently of any operation on the unit. Such a selection affects only BKSP requests, as other commands except for STATUS control selection of the feature.

EDITED STATUS

Magnetic Tape 601/603/606

Bit	Octal Mask	Name	Bit Set	Significance
00	000001	Ready	0 1	Tape is ready
01	000002	Busy	0 1	Channel, READ/WRITE control or unit busy
02	000004	Write Enable	0 1	No ring on tape or tape is unloaded Write ring in tape
03	000010	EOF mark	0 1	End-of-file mark read or written
04	000020	Load point	0 1	Tape is at load point
05	000040	EOT mark	0 1	Tape passed end-of-tape marker
06	000100	Density	0 1	200 bpi (LO) density selected 556 bpi (MED) density selected
07	000200	Not used	0 1	
08	000400	Lost data	0 1	Hardware failure
09	001000	End of operation	0 1	Operation completed; cleared when new operation init
10	002000	Parity	0 1	Parity Error
11	004000	Binary mode	0 1	Binary mode selected
12	010000	Not used	0 1	
13	020000	Not used	0 1	
14	040000	Not used	0 1	
15	100000	Channel parity	0 1	Channel parity
16	200000	Write history	0 1	Previous function was write

EDITED STATUS

Magnetic Tape 604/607/608

Bit	Octal Mask	Name	Bit Set	Significance
00	000001	Ready	0 1	Tape is ready
01	000002	Busy	0 1	Channel, READ/WRITE control, and/or unit busy
02	000004	Write	0 1	No ring on tape or tape is unloaded Write ring is on tape
03	000010	EOF Mark	0 1	End-of-file mark read or written
04	000020	Load point	0 1	Tape at load point
05	000040	EOT mark	0 1	Tape has passed end-of-tape marker
06	000100	Density	0 1	200 bpi (LO) density selected 556 bpi (MED) density selected
07	000200	Density	0 1	800 bpi (HI) density selected
08	000400	Lost data	0 1	Hardware failure
09	001000	End of Operation	0 1	Operation completed; cleared when new operation initialized
10	002000	Parity	0 1	Parity error
11	004000	Binary mode	0 1	Binary mode selected
12	010000	Not used	0 1	
13	020000	Not used	0 1	
14	040000	Reverse motion	0 1	Reverse motion requested
15	100000	Channel parity	0 1	Channel parity error
16	200000	Write history	0 1	Previous function was write.

5.6.2 CARD READER

DRIVER02 3248/405 and 3649/405† and 3447/405†

Operation Performed	Permissible Function Code	Calling Sequence	Macro Name	Logical Unit (u)						
				1-57	58 CFO	59 CTO	60 INP	61 OUT	62 PUN	63 LIB
Read 1 card	01	data transfer		yes			yes			
Check (1) status	13	status	STATUS	yes			yes			
Select (1) format	14	format	FORMAT	yes			yes			

† CONTROL DATA® 3248 and 3649 Card Reader Controllers
CONTROL DATA® 405 Card Reader

Hollerith to BCD conversion is always in effect. FORMAT and other requests for mode change are used to set the mode (artificial parity) status bit.

1. FORMAT codes:

0	No operation
1	Hollerith
2	Binary
3,4,5	No operation
6,7	Illegal

EDITED STATUS

Card Reader 405

Bit	Octal Mask	Name	Bit Set	Significance
00	000001	Ready	0 1	Reader ready
01	000002	Busy	0 1	Card being fed past read stations
02	000004	Not used	0 1	
03	000010	EOF card	0 1	End-of-file card read (7,8 in col. 1)
04	000020	Not used	0 1	
05	000040	Hopper Empty	0 1	Tray empty
06	000100	EOF switch	0 1	Tray empty and end-of-file switch on 405 set
07	000200	Not used	0 1	
08	000400		0 1	Feed failure, stacker full or jam
09	001000	End-of-operation	1	Operation completed; cleared when new operation initialized
10	002000	Mode error	0 1	Logical parity error
11	004000	Binary card	0 1	Rows 7,9 in column 1 in last card read.
12	010000	Preread or compare error	0 1	Compare error
13	020000	Logical mode selection	0 1	Binary mode requested†
14	040000		0 1	
15	100000	Channel parity error	0 1	Channel parity error active
16	200000	Not used	0 1	

†Driver manipulated

5.6.3 PRINTER

DRIVER03 3659, 3256, 3254, 3152/501 and 3555/512†

Operation Performed	Permissible Function Code	Calling Sequence	Macro Name	Logical Unit (u)						
				1-57	58 CFO	59 CTO	60 INP	61 OUT	62 PUN	63 LIB
Printer 1 line (1)	02	data transfer		yes				yes		
Eject page	11	control		yes						
Check status	13	status		yes				yes		
Select format (2)	14	format		yes				yes		

† CONTROL DATA® 3659, 3256, 3254, 3152, and 3555 Line Printer Controller
CONTROL DATA® 501 and 512 Line Printer

DRIVER03 501

1. First character of print line is used for carriage control; it is not printed.

User Character (PCC)	Action Before Print	Action After Print
1	Page eject	No space
2	Skip to last line	No space
3	Level 6 skip	No space
4	Level 5 skip	No space
5	Level 4 skip	No space
6	Level 3 skip	No space
7	Level 2 skip	No space
8	Level 1 skip	No space
A	Space 1	Page eject
B	Space 1	Last line - skip
C	Space 1	Level 6 - skip
D	Space 1	Level 5 - skip
E	Space 1	Level 4 - skip
F	Space 1	Level 3 - skip
G	Space 1	Level 2 - skip
H	Space 1	Level 1 - skip
Q	Clear auto eject mode	No print
R	Select auto eject mode	No print
(blank) 60B	Space 1	No space
0	Space 2	No space
- (40B)	Space 3	No space
+	No space	No space
Other	Space 1	No space

2. FORMAT codes:

0	Page eject
1	Skip to format level 1
2	Skip to format level 2
3	Skip to format level 3
4	Skip to format level 4
5	Skip to format level 5
6	Skip to format level 6
7	Skip to last line

EDITED STATUS

Line Printer 501

Bit	Octal Mask	Name	Bit Set	Significance
00	000001	Ready	0 1	Printer ready
01	000002	Busy	0 1	Paper in motion
02	000004	Not used	0 1	
03	000010	Last line	0 1	Carriage format tape directly on channel 7
04	000020	Not used	0 1	
05	000040	Paper out	0 1	Sensors signal no paper
06	000100	Not used	0 1	
07	000200	Post print last line	0 1	Post print (last line selected)†
08	000400	Post print page eject	0 1	Post print (page eject selected)†
09	001000	End-of-operation	0 1	Operation completed; cleared when new operation initialized
10	002000	Not used	0 1	
11	004000	Not used	0 1	
12	010000	Not used	0 1	
13	020000	Auto eject clear	0 1	Auto eject not in effect †
14	040000	Auto eject	0 1	Auto eject in effect†
15	100000	Channel parity	0 1	Channel parity error
16	200000	Not used	0 1	

†Driver manipulated

DRIVER03 512

1. First character of print is used for carriage control, it is not printed.

User Character (PCC)	Action Before Print	Action After Print
A	Space 1	Page eject
B	Space 1	Skip to last line
C	Space 1	Skip to level 6
D	Space 1	Skip to level 5
E	Space 1	Skip to level 4
F	Space 1	Skip to level 3
G	Space 1	Skip to level 2
H	Space 1	Skip to level 11
I	Space 1	Skip to level 7
J	Space 1	Skip to level 8
K	Space 1	Skip to level 9
L	Space 1	Skip to level 10
1	Page eject	No space
2	Skip to last line	No space
X	Skip to level 10	No space
Y	Skip to level 9	No space
Z	Skip to level 8	No space
9	Skip to level 7	No space
3	Skip to level 6	No space
4	Skip to level 5	No space
5	Skip to level 4	No space
6	Skip to level 3	No space
7	Skip to level 2	No space
8	Skip to level 11	No space
0 (zero)	Space 2	No space
+	No space	No space
-	Space 3	No space
(blank)	Space 1	No space
Q	Clear Auto Page eject	No print
R	Select auto page eject	No print
S	Select 6 lines per inch	No print
T	Select 8 lines per inch	No print

2. FORMAT codes, perform as defined:

1	Page eject
2	Level 2 skip
3	Level 3 skip
4	Level 4 skip
5	Level 5 skip
6	Level 6 skip
7	Level 7 skip
8	Level 8 skip
9	Level 9 skip
10	Level 10 skip
11	Level 11 skip
12	Last line skip
13	Select extended array
14	Clear extended array
15	Enable load image-memory

EDITED STATUS

Line Printer 512

Bit	Octal Mask	Name	Bit Set	Significance
00	000001	Ready	0 1	Printer ready
01	000002	Busy	0 1	Paper in motion
02	000004	Not used	0 1	Not used
03	000010	Last line	0 1	Positioned on last line of form
04	000020	Level 9	0 1	Level 9 positioned
05	000040	Paper fault	0 1	Paper out or paper jam
06	000100	Memory busy	0 1	Buffer memory busy
07	000200	Post print last line	0 1	Post print last line skip required†
08	000400	Post print page eject	0 1	Post print page eject required†
09	001000	End-of-operation	0 1	Operation complete
10	002000	Print error	0 1	A nonprintable character encountered
11	004000	Compare fault	0 1	A print character is not on the chain
12	010000	6/8 Line Coincide	0 1	May switch from 6-8 L/I without overprinting
13	020000	Auto eject cleared	0 1	Auto eject not in effect†
14	040000	Auto eject	0 1	Auto eject is in effect†
15	100000	Channel parity	0 1	Channel parity error
16	200000	Not used	0 1	

†Driver manipulated

5.6.4 CARD PUNCH

DRIVER04 3446, 3644, and 3245/415†

Operation Performed	Permissible Function Code	Calling Sequence	Macro Name	Logical Unit (u)						
				1-57	58 CFO	59 CTO	60 INP	61 OUT	62 PUN	63 LIB
Punch 1 record	02	data transfer	WRITES	yes					yes	
Punch EOF	11	control	WEOF	yes						
Check status	13	status	STATUS	yes					yes	
Select (1) hardware function	14	format	FORMAT	yes					yes	

† CONTROL DATA® 3644 and 3245 Card Punch Controllers
CONTROL DATA® 415 Card Punch

1. FORMAT codes:

0	No operation
1	Negate BCD-to-Hollerith conversion (set binary)
2	Return to BCD-to-Hollerith conversion
3	Select offset stacker (415 only)
4	Check last card (3446 and 3644)
5	Clear
6,7	No operation

EDITED STATUS

Card Punch

Bit	Octal Mask	Name	Bit Set	Significance
00	000001	Ready	0 1	Card punch ready
01	000002	Busy	0 1	Card punch busy
02	000004	Not used	0 1	
03	000010	Not used	0 1	
04	000020	Not used	0 1	
05	000040	Not used	0 1	
06	000100	Not used	0 1	
07	000200	Not used	0 1	
08	000400	Fail to feed	0 1	Card failed to feed into punch
09	001000	End-of-operation	0 1	Operation completed, cleared when new operation initialized
10	002000	Compare error	0 1	Compare error
11	004000	Binary card	0 1	User requested binary card
12	010000	Not used	0 1	
13	020000	Not used	0 1	
14	040000	Not used	0 1	
15	100000	Channel parity	0 1	Channel parity error
16	200000	Not used	0 1	

5.6.5 CONSOLE TYPEWRITER

DRIVER05

Operation Performed	Permissible Function Code	Calling Sequence	Macro Name	Logical Unit (u)						
				1-57	58 CFO	59 CTO	60 INP	61 OUT	62 PUN	63 LIB
Read max. 80-character buffer	01†	data transfer	READS		yes	yes				
Type max. 80-character buffer	02†	data transfer	WRITES		yes	yes				
Check status	13	status	STATUS		yes	yes				

† Interrupt not allowed.

EDITED STATUS

Console Typewriter

Bit	Octal Mask	Name	Bit Set	Significance
00	000001	Ready	0 1	Console typewriter ready
01	000002	Busy	0 1	Console typewriter busy
02	000004	Not used	0 1	
03	000010	Not used	0 1	
04	000020	Not used	0 1	
05	000040	Not used	0 1	
06	000100	Not used	0 1	
07	000200	Not used	0 1	
08	000400	Not used	0 1	
09	001000	End-of-operation	0 1	Operation completed; cleared when new operation initialized
10	002000	Repeat	0 1	Repeat button depressed
11	004000	Not used	0 1	
12	010000	Not used	0 1	
13	020000	Not used	0 1	
14	040000	Not used	0 1	
15	100000	Not used	0 1	
16	200000	Not used	0 1	

INTERRUPT CONTROL 6

The interrupt control section of the computer system hardware tests for the existence of selected interrupt conditions near the end of each RNI cycle. When an interrupt condition exists and the interrupt system has been enabled, the sequence of instructions currently in execution is interrupted, the contents of the program address register and the interrupt code are stored, and the RTS central interrupt control routine (CIC) is given control. CIC is a general-purpose routine that facilitates priority interrupt control. (In this discussion, CIO is considered a user of CIC.) CIC performs the following functions:

- Saves all displayable register contents (A,Q,IMR, etc.) upon interrupt occurrence
- Clears extraneous interrupt conditions such as fault and overflow
- Routes control to user interrupt routines according to priority
- Stacks low priority interrupts for future processing while processing a higher priority interrupt
- Provides linkage with the accounting routine to stop or start the clock when a priority interrupt is processed
- Diagnoses interrupt error conditions and informs the operator with a diagnostic message

Routing an interrupt through CIC uses from 145 to 300 microseconds if the central processing unit is not being shared with an I/O operation. If the central processor is being shared, interrupt recognition time becomes a function of the time required to complete the I/O operation.

6.1 INTERRUPT PRIORITY

RTS INTERRUPT PRIORITY
1. Real-Time interrupts
2. System interrupts
3. Priority program user interrupts
4. Batch program user interrupts

CIC receives control according to the interrupt priority scanning scheme that is a part of the hardware; CIC routes control to the interrupt-processing routine according to its own priority scheme. CIC recognizes four primary classes of interrupts: real-time, system, non-real-time priority, and batch. Real-time, non-real-time priority, and batch interrupts are user interrupts provided by the system.

6.1.1 SYSTEM INTERRUPTS

System interrupts are given immediate recognition unless a real-time interrupt is being processed; if so, system interrupts are disabled and recognized at completion of real-time interrupt processing. System interrupts include channel interrupts, end-of-operation interrupts, and abnormal end-of-operation interrupts. When a channel interrupt occurs, I/O interrupt processor (IOIP) clears the interrupt and frees the channel for another operation on a different equipment. When an equipment interrupt occurs, IOIP clears the interrupt and releases the equipment.

6.1.2 REAL-TIME INTERRUPTS

Real-time interrupts possess the highest priority level. They may originate from the clock, through manual interrupt, or from an I/O channel on which real-time devices are connected. They may interrupt system I/O routines. Real-time SCOPE requires that a user service his own real-time I/O interrupts (i.e., the real-time program must select and clear real-time interrupts, see Chapter 7).

6.1.3 NON-REAL-TIME PRIORITY INTERRUPTS

Non-real-time priority interrupts, occurring when some non-real-time priority operation reaches a predetermined condition, have priority over batch interrupts; they may not, however, interrupt system I/O processing.

A non-real-time priority interrupt can be from the clock if it is not reserved for real-time use, from an I/O device under CIO control, or from an I/O device under user control.

6.1.4 BATCH INTERRUPTS

A batch interrupt occurs when some batch operation reaches a predetermined condition. Batch users can select interrupt on any non-I/O condition, such as arithmetic overflow or search/move, including the clock if it is not reserved for the priority program. Batch I/O interrupts are similar to non-real-time priority interrupts, but they are not recognized until all real-time and priority processing is completed and the I/O system is not busy.

6.2 CENTRAL INTERRUPT TABLE

For each type of interrupt condition, Real-Time SCOPE maintains an entry in the central interrupt table (CIT) for the interrupt clear mask, priority bits and the user interrupt address. The initial format of the relevant entries in the table is:

CIT	BSS	6	STORAGE AREA
+6	04	ABINRT	CLOCK
+7	20	ABINRT	ARITHMETIC OVERFLOW
+8	20	ABINRT	DIVIDE FAULT
+9	10	ABINRT	EXPONENT OVERFLOW
+10	10	ABINRT	BCD FAULT
+11	40	ABINRT	SEARCH/MOVE
+12	00	ABINRT	MANUAL
+13	00	ABINRT	ADJACENT PROCESSOR
+14	00	address	CHANNEL 0
+15	00	address	1
+16	00	address	2
+17	00	address	3
+18	00	address	4
+19	00	address	5
+20	00	address	6
+21	00	address	7

The entry point, CIT, is a batch interrupt processing flag.

CIT	+0	No batch interrupt processing is in progress
	-0	Batch interrupt processing is in progress

The address field for each channel entry is set to IOIP at installation time for each channel used by system I/O routines. Other channel entry address fields initially contain ABINRT.

When a program terminates normally or abnormally, the interrupt addresses for that program which were initially ABINRT are reset to ABINRT.

6.3 I/O INTERRUPT PROCESSOR

All I/O interrupts selected by CIO are processed through IOIP. IOIP is permanently entered as the user address in the CIT entries for CIO interrupts. When an interrupt occurs, control transfers to IOIP which determines whether the interrupt is internal or external and processes it as follows:

Internal (channel) interrupt:

Determines channel number from the interrupt code†

Clears the interrupt

Sets CST/b to 0 to designate channel is not busy

Updates bcr/f_1 in UST (word 1)

Updates channel status in CST

Returns control to CIC

External (equipment) interrupt:

Senses and processes the channel interrupt

Determines channel and equipment numbers from interrupt code

Updates UST status entry if unit with dynamic status is connected to another controller on the same channel

Connects interrupting unit and updates its UST status entry

Sets UST/d to 1 indicating unit is static

Sets EST/i to nonzero indicating controller is free and, if user requested interrupt control, passes user interrupt address (EST/i addr) to CIC

Returns control to CIC; when CIC routes control to the user interrupt routine, the A and Q registers contain the UST entry for the unit

6.4 INTERRUPT STACKING

After IOIP has processed an I/O interrupt and returned control to CIC, CIC checks to see if it can immediately transfer control to user interrupt routine. If it cannot, the user routine is placed in a priority or batch stack table. Priority table entries are unstacked before batch table entries. CIC unstacks on a first-in/first-out basis. It can stack while unstacking.

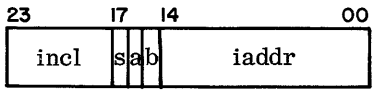
Each stack can contain a maximum of 16 entries. A program that causes a stack to overflow is terminated and a diagnostic is typed on CTO (appendix H).

IOIP processes all interrupts occurring in system I/O operations and, through CIC, routes control to the user interrupt subroutine according to priority.

†An equipment interrupt that occurs before a channel interrupt must wait for the channel interrupt before it can be processed.

6.5 NON-I/O INTERRUPTS

To gain control on non-I/O interrupts, the user must place a declaration of priority and the address of the interrupt subroutine in the entry of the CIT appropriate to the type of interrupt selected. The format of the entry is:



Bits	Field	Significance
23-18	incl	Interrupt clear mask; preset in the CIT (e.g., 04 in the CLOCK entry); CIO clears the non-I/O interrupt conditions with the interrupt clear mask
17	s	0 No system accounting routine 1 System accounting routine
16	a	0 Batch 1 Priority
15	b	0 Non-real-time 1 Real-time
14-00	iaddr	Address of user interrupt subroutine

6.6 DINT. AND EINT. RESTRICTIONS

The disable interrupt (DINT) and enable interrupt (EINT) instructions should not be used in batch programs run under Real-Time SCOPE. They could cause unnecessary or destructive delays in real-time or priority interrupt recognition. Two RTS subroutines simulate the disabling and enabling of interrupts. DINT. forces user routines to be stacked and EINT. initiates their unstacking. DINT. and EINT. are used by COMPASS, FORTRAN, and other standard library programs. In order to reference the DINT. and EINT. subroutines, the user must declare them external.

6.7 PRIORITY INTERRUPT RESTRICTIONS

Immediately after loading, a real-time or non-real-time priority program receives control to initialize itself. During initialization, it must select the interrupt conditions (manual, clock, or equipment) through which it regains control.

After initialization, the priority program transfers control to RTS and regains it only when a selected interrupt occurs. At each turn of control, the priority program must enable some operation that results in an interrupt or it permanently loses control and becomes dormant.

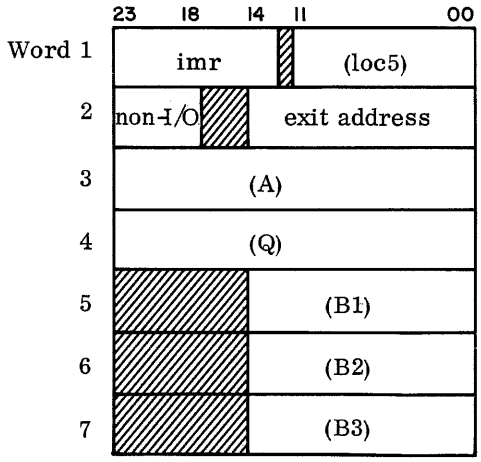
If a non-real-time priority program depends upon an interrupt other than a system I/O interrupt, a unique priority mask must be set during the initialization stage. The setting is retained for the duration of the run.

6.8 REAL-TIME INTERRUPT

During initialization, a real-time priority program must set a unique priority interrupt mask in location CIT.RTM. The real-time user must declare CIT.RTM external.

To CIC, all real-time interrupts have equal priority over all other forms of interrupts. However, any number of levels of real-time interrupts may be defined by the user. To further delineate real-time priorities, the user program must save the current contents of the real-time register storage area for each level encountered and restore it each time control is released. The real-time register storage block is the first seven words of the area CIT.RSA.

To reference the real-time register storage area, the user must declare CIT.RTM external. The real-time register storage area begins at CIT.RTM + 1.†



† When CIT.RSA is a system entry point, a user program may declare CIT.RSA an external name and reference the real-time register storage area through this symbol (CIT.RSA = CIT.RTM + 1).

Word	Bits	Field	Significance
1	23-14	imr	Interrupt mask register code at time of interrupt
	11-00	(loc5)	Contents of location 5; code representing interrupt-causing condition
2	23-19	non-I/O	Bit pattern representing non-I/O conditions present
	14-00	exit address	Address of instruction which was interrupted

6.9 REAL-TIME CIO CALLS

A real-time priority program which calls CIO for I/O operations on non-real-time channels should first call RIO to determine if the priority program gained control on interruption of CIO execution.

If RIO finds CIO busy, RIO sets a flag in CIC and stores the address of the real-time I/O subroutine before returning to the real-time program. The real-time program must give up control as soon as possible to enable CIO to complete its interrupted assignment. When CIO completes its assignment, CIO transfers control to the real-time I/O subroutine.

6.10 UNASSIGNED INTERRUPTS

When an interrupt occurs and the entry for that interrupt contains the address ABINRT, a diagnostic message appears on CTO.

Because CIC is unable to determine which program should receive control, it clears the condition and resumes normal execution, returning control to the interrupted program. If the abnormal condition is I/O and another equipment on the channel has an interrupt awaiting recognition, it also is cleared.

6.11 UNMASKED INTERRUPTS

Adjacent processor and manual interrupts are unmasked. It is possible for these interrupts to occur during execution of a high-priority real-time procedure or some critical phase of RTS. If one occurs, CIC stores it temporarily.

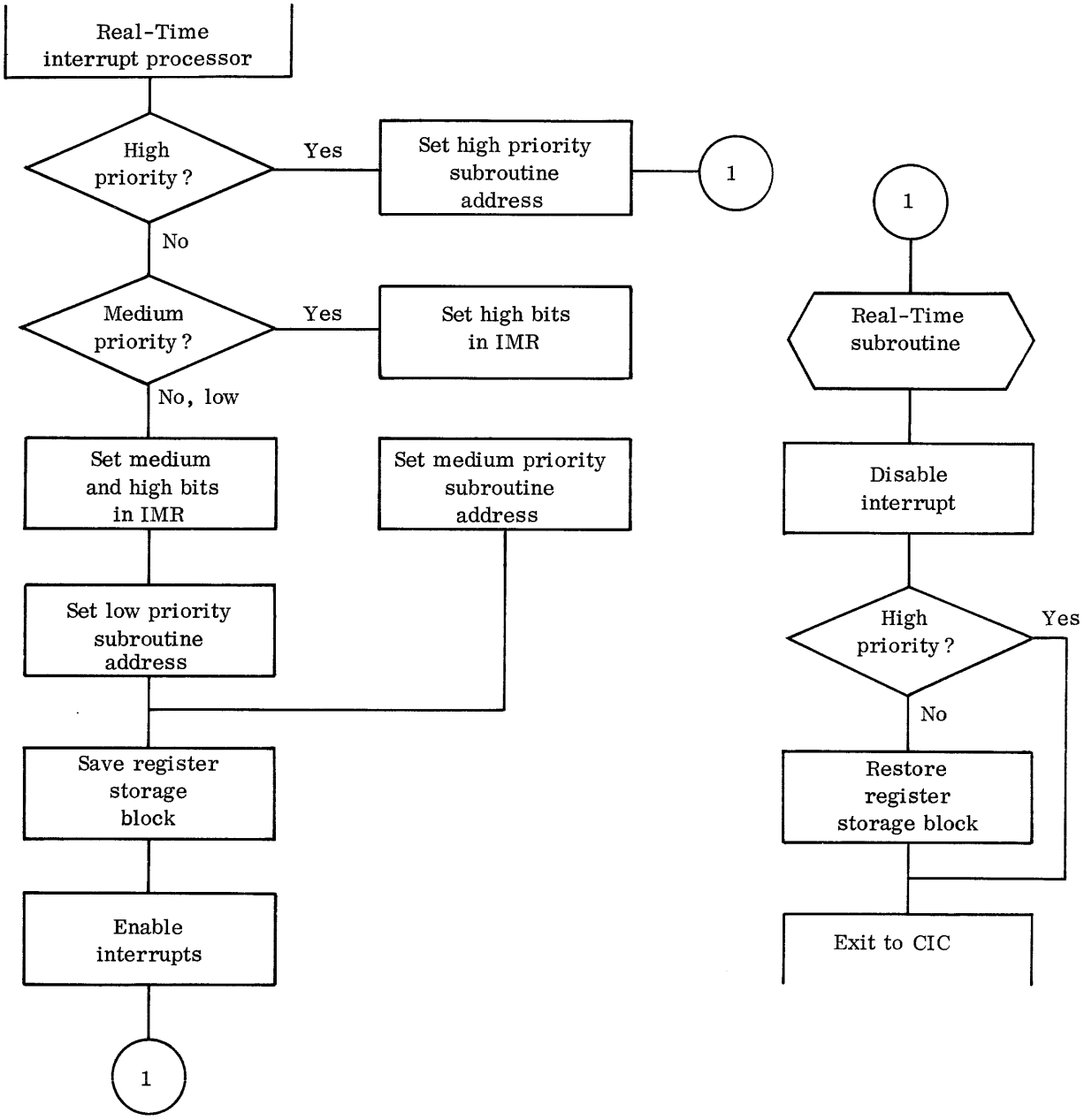


Figure 6-1. Real-Time Interrupts

At the end of each CIO function or interrupt procedure, CIC checks for an unmasked interrupt and, if one has occurred, simulates the condition.

Caution: Priority and batch units should be connected through different channels if possible. When this is not possible, the priority program must give up control for short periods of time through the use of a real-time clock interrupt; otherwise, the priority program has continuous control of the channel because of the higher priority of priority interrupts, and batch processing is not possible.

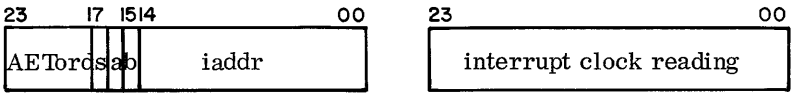
6.12 CIP OPTION

The clock interrupt processor (CIP) option stacks clock interrupts to prevent conflicts between batch and priority requirements. When more than one interrupt requests control at the same time, control is passed according to the following priority level.

1. Priority and real-time programs
2. System routines (including drivers)
3. Batch programs

Control is passed to priority, system, and batch users through a return jump similar to other non-I/O interrupts (section 6.5). The CIP is called by the user to request a clock interrupt or to clear a previously requested interrupt which has not yet occurred. Only CIP is a user of the clock and any routine needing clock interrupt must be a user of CIP. The lost interrupt recovery option requires CIP.

User supplied parameters:



A-register

Q-register

Bits	Field	Significance
23-18	AETord	Supplied by drivers
17	s	1 System routine 0 Not system routine
16	a	1 Priority or real-time routine 0 Not priority or real-time

Q = Desired clock reading at time of interrupt

Q = 0 To clear previous call

Bits	Field	Significance
15	b	Not used
17-15	s a b	Batch routine
14-00	iaddr	Address of user interrupt subroutine

Calling sequence for CIP:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
p	RTJ	CIP	
p+1	reject return		
p+2	normal return		

A reject return (p+1) indicates one of the following conditions:

1. Clear call (Q=0)
   - A register does not match outstanding request for caller or
   - There is no outstanding request
2. Set call (Q≠0)
   - Time requested has occurred or
   - Another request is outstanding for caller

6.13 LOST INTERRUPT DETECTION

Real-Time SCOPE can detect lost equipment interrupts with the TIMEOUT assembly option. If it is used, CIP must be assembled.

Lost interrupt detection uses a clock interrupt to indicate a specified time limit has passed. When an I/O request requires the equipment interrupt, CIO records the real-time clock, a time increment, and a pointer to the equipment entry in the EST.

If a clock interrupt occurs, the specified time limit has elapsed and no equipment interrupt has occurred. This indicates the equipment interrupt is lost, usually due to a hardware malfunction. A diagnostic message (appendix H) is typed, the operator presses FINISH, the lost equipment interrupt is simulated, and processing continues.

REAL-TIME AND PRIORITY OPERATION 7

A priority program is a debugged, relocatable, binary program which obtains control through interrupts and promptly returns control to the operating system. Priority programs may equip logical units 1-49 and may reference logical units 1-49 (when previously equipped), 58, and 59. A real-time priority program normally calls RIO (section 5.1) before calling CIO. Real-time equipment must not share a channel with non-real-time equipment. Equipment connected through a real-time channel must be driven by user provided routines.

SEQUENCE OF EVENTS FOR PRIORITY PROGRAMS

- Loaded by RTS at inter-job time
- Entered through ENTER routine
- Initializes and returns to REENTER
- Obtains further control via interrupts
  - Real-Time
  - I/O
  - Manual Interrupts
  - Real-Time clock
- Performs functions
- Re-initializes and relinquishes control
- Terminates
  - BKEXIT (self)
  - TERM,P (operator)
  - New priority job (INP with operator consent)

Priority programs are loaded from INP, from LIB, and from logical units designated in LOAD statements. A priority program may submit batch job stacks with redesignated INP, OUT, and/or PUN (appendix B). All other priority program references to standard system units are illegal. The priority-submitted INP, OUT, and PUN must be previously equipped priority logical units. The submitting program may not refer to the submitted units while being processed in the priority submitted batch job stack. The priority program may request notification of completion of a priority-submitted batch job stack (appendix B). The priority program should not submit a new batch job stack while a previous priority-submitted batch job stack is in process. A priority program may not submit a priority job.

7.1 PRIORITY PROGRAM OPERATION

A priority program is a special purpose routine that requires control for discrete intervals or that is I/O bound. The program either is loaded from the library in binary form in response to a call on the CFO at inter-job time or is part of a batch job stack on the standard input unit. After the priority program is loaded, RTS gives it control of the CPU. It initializes itself and returns the program control to the operating system; it regains control only on interrupts. When it has gained control through an interrupt, the program must perform its designated function, select another interrupt to regain control, and return control of the CPU to the operating system.

7.1.1 PRIORITY INTERRUPTS

Priority program execution could be initiated and directed through a manual interrupt and subsequently could obtain control by user interrupt, by real-time interrupt, by real-time clock interrupt, or by another manual interrupt. Priority programs cannot select internal interrupt conditions (arithmetic overflow, divide faults, BCD faults, etc.).

7.2 REAL-TIME PRIORITY PROGRAM OPERATION

REAL-TIME PROGRAMS
Highest interrupt priority
Dedicated Real-Time channel
Performs own Real-Time I/O
All interrupts disabled
RIO used for CIO calls
May allow further Real-Time interrupts

Because real-time programs require equipment on a dedicated channel, the user must provide all I/O routines for the real-time equipment. The equipment is neither listed in the AET (section 2.4.1) and the SYST.IOM (see RTS Installation Handbook) nor handled as a typical system driver. A real-time program requiring CIO for a device other than real-time equipment must use RIO (section 5.1).

7.2.1 REAL-TIME PRIORITY INTERRUPTS

Real-time interrupts interrupt the operating system, the I/O system, and the priority, batch, or lower priority real-time program in process. A real-time program receives control upon recognition of a real-time interrupt and immediately disables the interrupt system.

To the I/O system, all real-time interrupts have equal priority over other types of interrupts. The user must define the levels of interrupts, enable the interrupt system, and maintain the real-time register storage area

when multiple levels of real-time interrupts are used. For each level of real-time interrupt, the real-time register storage area (the first seven words of the CIT.RSA area) must be saved at interrupt time and restored upon return to the program.

7.3 PROGRAM TERMINATION

Real-time and priority programs may be terminated in several ways. When a new priority program is requested through CFO or INP, the operator either terminates the priority program in process or allows it to continue until self-termination. The operator terminates a priority program with the TERM,P manual interrupt routine (appendix F). This terminates all priority processing and should be used only in unusual cases where access to all of core is essential.

The system entry point BKEXIT provides another means of terminating a priority program. The BKEXIT routine provides entry into ABNORMAL with a suppressed recovery dump.

When a priority program terminates, memory limits are reset to protect the new priority program or, if no new priority program is loaded, to provide batch programs access to all of core.

7.4 MANUAL INTERRUPT

MANUAL INTERRUPT PROCEDURE

Press: MANUAL INTERRUPT

Type: = priority message

Press: MANUAL INTERRUPT

MIBKADD: Address of manual interrupt processing routine of priority program (appendix F)

MIBUF: 80-character message buffer (appendix F)

1. Period (.) filled
2. First character is a blank

Additional manual interrupt messages are logically locked out during current manual interrupt processing.

The resident manual interrupt routine relays CFO messages and interrupts to the system and the batch or non-real-time priority program. When a priority program uses manual interrupt, initialization, restart, and response messages, CFO communication to the rest of the I/O system should not be inhibited.

The manual interrupt routine uses the buffering characteristics of the console typewriter. When the operator first presses MANUAL INTERRUPT, the I/O system illuminates TYPE LOAD on the console and returns to the interrupted process. The operator types the prefix =, the priority message, (appendix F), and again presses MANUAL INTERRUPT. The CPU transfers control to the appropriate subroutine (appendix F) to process the completed CFO message. Batch and system communication, inhibited until the second manual interrupt, is again enabled.

Batch programs often require extensive communication through the typewriter. Priority programs can use the manual interrupt routine to avoid looping by the processing program. As a result of priority program use of manual interrupt:

1. Batch programs cannot be locked out from typewriter access and thus suspended from operation because of an outstanding priority CFO request.
2. Batch program execution continues although an outstanding CFO request exists. The priority program does not need to wait for the request to be completed before relinquishing control to the batch program.
3. A priority program may enable several CFO messages simultaneously, allowing the operator several input options, depending on needs and requirements. (For example, BSIPP may allow several requests to initiate different BSIPP operations.)

7.5 MULTI-PROGRAMMING

PRIORITY PROGRAM TRAPS

- Priority and batch channel conflicts
- Excessive CPU control
- CFO/CTO conflict
- Batch job run as priority
- Nondebugged priority jobs

Efficient priority and batch multiprogramming hinges on the design and implementation of the priority program because it dominates use of the CPU and gains control from CIC. RTS does not allocate time increments for batch and priority execution. Therefore, since the priority program locks out batch processing, the priority program must return control to the system so the batch processing can continue.

Priority programs must be debugged; a nondebugged program can destroy batch operation and even the operating system. No batch program should be in execution while the priority program is being debugged.

7.6 SAMPLE PRIORITY PROGRAM

The following sample program illustrates priority processing.

```
LOCN      OPERATION,MODIFIERS   ADDRESS FIELD                COMMENTS                                  IDENT
          EXT                   CIØ,RIØ,BKEXIT,MIBKADD
          ENTRY                 TØP
TØP       UJP                   **                          RETURN CØNTRØL TØ BATCH PRØGRAM
          ENA                   ABØRT                       ENTER ADDRESS ØF M/I RØUTINE
          SWA                   MIBKADD                     STØRE IN B/G ADDRESS
          RTJ                   RIØ                         WAS CIØ INTERRUPTED
          AZJ,NE                TØP                         YES RETURN CØNTRØL
          RTJ                   REWIND                      NØ, GØ REWIND THE TAPE
          UJP,I                 TØP                         RETURN CØNTRØL TØ BATCH PRØGRAM
REWIND    UJP                   **
          RTJ                   CIØ                         REWIND LUN 1
          04                    1,3                         WITH INTERRUPTS
          UJP                   *-2                         REJECT ADDRESS
          UJP                   INTADD                      INTERRUPT ADDRESS
          UJP,I                 REWIND                      RETURN
INTADD    UJP                   **
          RTJ                   RIØ                         WAS CIØ INTERRUPTED
          AZJ,NE                INTADD                      YES RETURN CØNTRØL
          RTJ                   STATUS                      NØ, CHECK STATUS AND INIT I/Ø
          UJP,I                 INTADD                      THEN RETURN
STATUS    UJP                   **
          RTJ                   CIØ                         GET STATUS
          13                    1                           LUN 1
```

DOC 366

LOCN	OPERATION,MODIFIERS	ADDRESS FIELD	COMMENTS	IDENT
	AZJ,GE	*-2	LØØP IF BUSY	
	SHAQ	+2	MØVE EØT TØ SIGN BIT	
	AZJ,LT	RWD	EØT THEN REWIND	
	RTJ	CIO	WRITE 1 RECØRD	
	02	1,3	LUN 1 WITH INTERRUPTS	
	UJP	*-2	REJECT	
	60	0	MØDE AND FWA	
	ØCT	77000	NUMBER ØF WØRDS	
	UJP	INTADD		
	UJP,I	STATUS	RETURN	
RWD	RTJ	CIØ	REWIND LUN 1	
	04	1,3	WITH INTERRUPTS	
	UJP	*-2	REJECT	
	UJP	INTADD	INTERRUPT ADDRESS	
	ENA	0	CLEAR A REGISTER	
STØPPER	ASE	**	HAS M/I ØCCURRED	
	UJP,I	STATUS	NØ, KEEP GØING	
	UJP	BKEXIT	YES, THEN EXIT B/G PRØGRAM	
ABØRT	UJP	**		
	ENA	0	CLEAR A REGISTER	
	SWA	STØPPER	CLEAR STØPPER	
	UJP,I	ABØRT	RETURN	
	END	TØP		

CDC 366

REAL-TIME SCOPE CONTROL STATEMENTS 8

Real-Time SCOPE job processing is directed by control statements entered as cards on the standard input unit or typed on the CFO by the operator. With control statements, the user establishes job type and equipment requirements, calls for the loading of batch and priority programs, selects memory dumps, directs the insertion of corrections to the loaded programs, directs execution of the loaded programs, and directs comments to the operator.

Control statements consist of a statement name or mnemonic immediately followed by parameters which define the operation.

Library routine control cards are described in the compiler and assembler reference manuals.

8.1 OPERATOR STATEMENTS

Operator control statements are typed on the CFO by the operator.

Statement	Purpose
SEQUENCE,j	Alter job-processing sequence from standard input unit; next job is j
AET,a,p	Examine or alter Real-Time SCOPE AET
ENDSCOPE	Suspend batch job processing
CALL,u,name,p_1,...,p_n	Call the named program for loading from u and execute according to p_1
CALL,name,p_1,...,p_n	Call the named program for loading from the library and execute according to p_1
BACK	Identify the next job as priority
LOAD,u_1,u_2,u_3	Calls loader to load binary decks
EQUIP	Assign equipment to programmer, scratch, and system logical units

Statement	Purpose
REWIND,u_1,...,u_n	Rewind logical units indicated
UNLOAD,u_1,...,u_n	Rewind logical units indicated and remove them from ready status
TRAIN,n,u	Mount specified train in 512 printer

The use of these statements is more fully explained in the Real-Time SCOPE Operator's Manual.

8.2 PROGRAMMER STATEMENTS

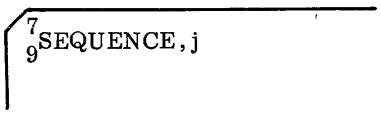
Programmer control statements are punched on cards.

SEQUENCE,j	XFER,u
BACK,ND (not used in BATCH option)	PAUS
JOB,c,i,t,NP,ND	CTO,$^{T}_{L}$,COMMENTS
ENDSCOPE,$^{N}_{R}$	REWIND,u_1,...,u_n
ENDREEL	UNLOAD,u_1,...,u_n
RUN,t,NM	TRAIN,n,u
Library Name,parameters	OCC,parameters
EQUIP,u_1=xx	SNAP,(subp)n,fwa,lwa,mode,id
LOAD,u_1,u_2,u_3	

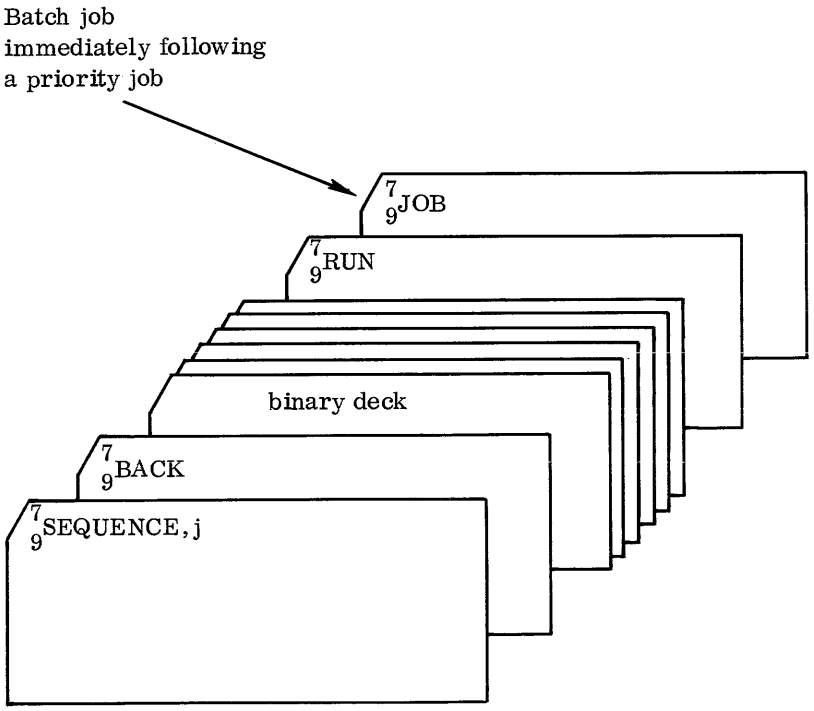
Except for the end-of-file cards, control cards are characterized by a 7,9 punch in column one. Columns 2 through 80, in Hollerith code, contain the statement name and parameters separated by commas. When a control statement has specific parameters and all are expressed, the final parameter may be terminated by a comma followed by comments or a blank. Extra spaces should not be inserted between the statement name and parameters.

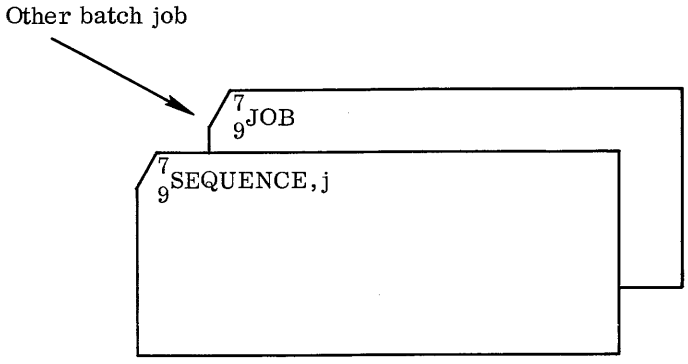
8.2.1 SEQUENCE

The SEQUENCE statement separates stacked jobs on INP and causes an EOF card to be punched at job end. A SEQUENCE card is placed in front of every job except a batch job immediately following a priority job.



j 1- to 3-digit job sequence number; must be present

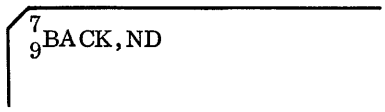




SEQUENCE notifies Real-Time SCOPE of the beginning of a new job. Real-Time SCOPE releases all batch units and, if OUT is a tape, writes an end-of-file. If OUT is a printer, Real-Time SCOPE causes a page eject. The operating system then produces a copy of the SEQUENCE card on CTO and OUT and enters j in the accounting table (appendix E). If SELECT JUMP 6 was set when the SEQUENCE card was read, Real-Time SCOPE is ready to process operator control statements entered from CFO.

8.2.2 BACK

BACK notifies Real-Time SCOPE that a priority program is to be loaded. BACK follows a SEQUENCE card.



ND	No dump; when this parameter is omitted, RTS makes a recovery dump on abnormal termination

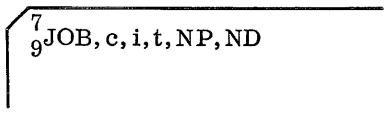
A LOAD card, loader card, EQUIP card, or library name card must follow the BACK statement.

Real-Time SCOPE copies BACK on CTO and OUT. If no priority program is in core, Real-Time SCOPE loads the program or comments on CFO. If a priority program is in core, Real-Time SCOPE requests permission from the operator to release the old priority program. If permission is granted, the new program is loaded; if denied, the new priority program is bypassed and the next JOB card is processed.

The BACK control card and operator statement are not included in the BATCH option.

8.2.3 JOB

The JOB statement identifies a batch job to Real-Time SCOPE. It must follow either a SEQUENCE card or the last priority Real-Time SCOPE control card.

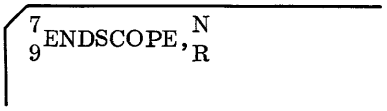


c	Account number, 0-8 characters
i	Programmer identification, any length provided that all specified parameters appear on one card
t	Estimated running time, 0-999 minutes
NP	No system unit protection; when NP is omitted, protection is in effect
ND	No dump; when omitted, recovery dump is taken on abnormal termination

If the c, i, or t field is blank, the comma delineating the field must appear or the job is terminated. If it is assigned, the JOB statement is copied on OUT, CTO, and PUN.

8.2.4 ENDSCOPE

ENDSCOPE indicates the end of a batch job stack. It appears after an end-of-file card and simulates autoload. When the operator presses MANUAL INTERRUPT and types GO, he may reassign INP, OUT, and PUN. If INP, OUT, and PUN are not magnetic tape, N or R is ignored. When the accounting routine is used, ENDSCOPE causes processing of the accounting record for the preceding batch jobs and closes the accounts file prior to unloading ACC.



N	INP, OUT, and PUN maintain current position
R	Rewind INP, OUT, and PUN
blank or other symbol	Rewind and unload INP, OUT, and PUN

8.2.5 ENDREEL

When INP is on magnetic tape, the ENDREEL statement in the job stack signals the end of an input reel (Real-Time SCOPE Operator's Manual); otherwise, it is ignored.

```
7
9ENDREEL
```

8.2.6 LIBRARY NAME

A library name statement calls the loader to load and execute a library routine such as COMPASS or FORTRAN.

```
7
9library name, parameters
```

library name	Primary entry point to a library program
parameters	Variables used by the named library program

A program may be called by a library name statement after PRELIB places the program on LIB. A library program does not require a RUN card; its execution is automatic. Real-Time SCOPE places the following information in the A register before passing control to the routine.

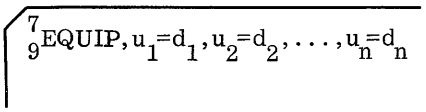
Bits	Contents
23-17	Column number of first character following first comma; zero if no comma
16,15	Zero
14-00	First word address of control card

Example:

```
7
9COMPASS, I, P, X, L, R
```

8.2.7 EQUIP

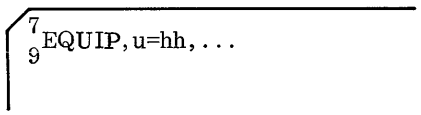
An EQUIP statement assigns physical units to batch logical units or, if preceded by a BACK statement, to priority logical units. EQUIP cards may not make equipment assignments for standard units (57-63). The operator may change standard units through operator EQUIP statements.



u_i	Logical unit number
d_i	Declaration about u_i in one of three forms; any or all may appear on a single EQUIP card

The three forms of the EQUIP statement are:

1. To assign a physical unit to a logical unit by hardware type:



u	Logical unit (batch, 1-56; priority, 1-49)
hh	Hardware type:

MT	Magnetic tape	
CR	Card reader	
PR	Printer	
CP	Card punch	
TY	Console typewriter	
TR	Paper tape reader	
TP	Paper tape punch	
PL	Plotter	
TS	Remote typewriter station	
SL	Satellite controller	
DP	Disk pack controller	Not provided in RTS
DF	Disk file controller	Not provided in RTS
DR	Drum	Not provided in RTS
OR	Optical character reader	

Real-Time SCOPE searches the AET for an unassigned physical unit of the type specified by hh and assigns it to u with the comment on CTO.

u	CcEeUuu (physical unit)
	or
	Logical unit number 1-63
c	Channel number 0-7
e	Equipment number 0-7
uu	Unit number 00-77

If u was previously assigned, its former physical unit is released for reassignment. When no unit of type hh is available or the statement contains an error, Real-Time SCOPE writes a message on OUT and the job is terminated.

2. To assign a physical unit specified by hardware code to a logical unit:

$^{7}_{9}$EQUIP,u=hhCcEeUuu

u	Logical unit (batch 1-56; priority 1-49)
hh	Hardware type
c	Channel number 0-7
e	Equipment number (controller) 0-7
uu	Unit number (device) 00-77

This form of EQUIP statement offers the following options; blank spaces in the table indicate parameters that may be omitted.

hh	Cc	Ee	Uuu
x	x		
x	x	x	
x	x	x	x
	x	x	x
	x	x	

If the hardware code identifies a device already assigned to a logical unit, Real-Time SCOPE prints an error message (Diagnostics, Appendix H) and terminates the job.

Real-Time SCOPE assigns the I/O device specified in the statement Uuu to its equated logical unit. If the specified channel (Cc) is unavailable, Real-Time SCOPE assigns any available channel.

3. To equip two or more logical units to the same physical unit:

```
7
9EQUIP,u1=u2,...,um=un
```

u_i Logical unit number

Real-Time SCOPE assigns the logical unit on the left of the equal sign to the same physical unit as the logical unit on the right. The unit on the right of the equal sign must be previously equipped. For batch programs, the unit on the right may be 1-57 or 60-63. For priority programs, the unit on the right may be 1-49. If the logical unit on the left has been previously assigned, Real-Time SCOPE releases the former physical unit and makes the new assignment. A unit equated to a protected system unit is also protected.

Examples of EQUIP statement:

To assign two units to the card reader:

```
7
6EQUIP,10=26
7
9EQUIP,26=CR
```

or

```
7
9EQUIP,26=CR,10=26
```

To assign two units to the same physical tape:

```
7
9EQUIP,18=16
7
9EQUIP,16=MTC0E0U06
```

or

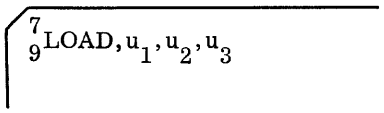
```
7
9EQUIP,16=MTC0E0U06,18=16
```

The card reader is assigned to logical unit 26. Logical units 16 and 18 are equipped as magnetic tape physical unit 06, connected through channel 0 and controller 0.

8.2.8 LOAD

LOAD calls the loader to load binary decks from the specified logical units and from INP. Only one LOAD statement may appear during a run, and no more than three units may be specified on the card. By repeating the unit designation in the LOAD statement, the programmer can cause multiple files to be loaded from the same unit. An end-of-file terminates loading from each specified unit.

If the user has previously defined u as equipment other than magnetic tape, RTS writes an error message on CTO and OUT and terminates the job.



u_i	Previously equipped logical unit (1-56 for batch programs; 1-49 for priority programs)

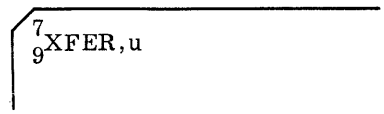
The user must ensure proper positioning of the unit prior to the LOAD statement. If $u_i = 56$, the system rewinds the unit before loading. If no parameters are present, the loader loads from INP.

When Real-Time SCOPE reads a binary card with a nonzero word count on INP, it assumes the card is the first card of a program and begins loading from INP without a load statement. Since Real-Time SCOPE calls the loader only once per run, a LOAD statement must precede the binary decks on INP if loading is required from both INP and a logical unit.

For overlay preparation when the binary deck is on INP, the MAIN card (section 10.1.1) must be preceded by a LOAD card and no parameters are needed.

8.2.9 XFER

XFER is for batch use only. Information between XFER and the next RTS control statement is transferred from INP to magnetic tape by Real-Time SCOPE.



u	Logical unit 1-57

If u is a logical unit previously equipped as magnetic tape, RTS transfers the information on INP (following XFER and up to the next RTS control card) to the specified or RTS selected magnetic tape. RTS writes an end-of-file mark at the end of the information and backspaces over the mark.

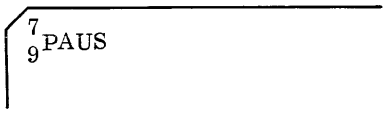
If u is within the legal range for logical units but is unassigned, RTS assigns u to the first available magnetic tape, logs the assignment on the CTO, and waits for a signal from the operator before continuing.

8.2.10 PAUS

PAUS is for batch use only. On encountering PAUS, Real-Time SCOPE suspends processing, copies the statement on CTO and OUT, and types:

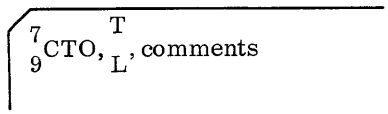
READY?

The operator can then mount new tapes, change paper in the printer, etc. The job resumes when the operator presses FINISH.



8.2.11 CTO

CTO is for batch use only. The CTO card causes the comments appearing on the card to be printed on CTO and/or OUT. The comments may direct the operator to take some action.



T	Comments on CTO only
L	Comments on OUT only
comments	Comments on CTO and OUT

8.2.12 REWIND

REWIND is for batch use only. Real-Time SCOPE copies the statement on CTO and OUT. Unit assignments are not altered.

```
7/9REWIND,u1,u2,u3,...,un
```

u_i Logical unit 1-56

The parameters specify logical units equipped to magnetic tape; Real-Time SCOPE rewinds the specified tapes to load point. If u_i is not magnetic tape or is unassigned, the request for u_i is ignored and the remainder of the statement is processed.

8.2.13 UNLOAD

UNLOAD is for batch use only. The parameters specify logical units equipped to magnetic tape. Real-Time SCOPE rewinds and unloads the specified tapes.

```
7/9UNLOAD,u1,u2,u3,...,un
```

u_i Logical unit 1-56

If u_i is not a magnetic tape, Real-Time SCOPE ignores the request for u_i and processes the remainder of the statement.

8.2.14 TRAIN

A TRAIN card notifies Real-Time SCOPE that a batch or priority program requires a specific train to be mounted in the 512 printer.

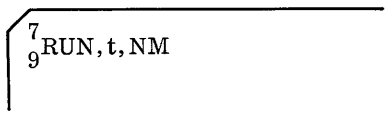
```
7/9TRAIN,n,u
```

n External symbol matches entries in preload and postload tables (See Real-Time SCOPE Installation Handbook).

u Logical unit assigned to 512 printer

8.2.15 RUN

On encountering a RUN statement, RTS copies the statement on CTO and OUT, enters t into the accounts table, and unless NM is specified writes a memory map on OUT. When this processing is complete, control transfers to the object program in memory.



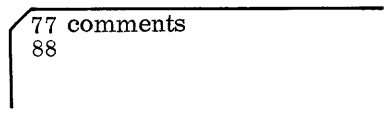
t	Estimated time in minutes, 0-999; always omitted for priority execution
NM	Suppress memory map; when omitted, RTS prints a memory map; always omitted for priority execution

If the object program is on INP, RUN follows the binary deck. If the program is on a unit other than INP, RUN follows the LOAD statement.

If the program runs successfully, RTS processes the next run. If the run is unsuccessful, RTS terminates the entire job.

8.2.16 EOF

An end-of-file (EOF) card must be the last card of each job on INP. Columns 3-80 may contain comments.



8.2.17 SNAP

Through use of a SNAP control card, a batch user may request the snapshot dump routine to periodically printout the contents of selected sections of memory. The requested dump may be in octal, character, or decimal floating-point.

Each time it is called, the routine writes on the system OUT unit, a line containing the 1-4-character BCD dump identifier, the location of the call to snapshot, and contents of the A and Q registers and the three index registers. When the register file option is selected, snapshot prints the subheading REGISTER FILE followed by the contents of the last 32 high-speed registers.

The memory dump consists of eight-word lines of data printed in the designated mode and preceded by a 4-character identifier (provided by the user) and by the absolute octal address and the relocatable word address of the first word of the line. When snapshot detects a line that contains words all identical to the last word of the preceding line, the line is suppressed. The suppressed line of words is noted with the word GAP on the listing.

SNAP control cards may be inserted in a job input deck after a subprogram control card (if the subprogram is not on the INP unit) or after an OCC control card. They cannot follow binary card images on a unit other than the INP unit. For each SNAP card, a calling sequence to snapshot is created and stored in available memory. An instruction in the subprogram specified on the SNAP card is replaced with a return jump to the calling sequence. The instruction is saved within the calling sequence and executed after the dump is taken. Because this instruction is not executed in its normal location, it must not be modified by the program and it must not involve more than one word (e.g., skips, indirectly addressed instructions, and searches).

- To prevent excessive printout, avoid calling snapshot within a loop.
- The location at which the SNAP occurs must not be altered during execution or by OCC cards.
- Do not replace the following types of instruction:
  - Instructions involving more than one word, such as searches and skips
  - Indirectly addressed instructions
  - Instructions modified by program execution

7_{9}SNAP, (subp)n, fwa, lwa, mode, id

(subp)n — Location of the instruction to be replaced. The name (1-8 BCD characters) of the subprogram containing the instruction appears within the parentheses. n is an address of not more than 5 octal digits to be added to subp to obtain the relative address of the instruction.

fwa, lwa — Beginning and ending addresses of the area to be dumped. Unless lwa exceeds or is the same as fwa, the SNAP statement is ignored and an error message is written on the OUT unit. If they are the same, the dump produces only the contents of console registers. Both fwa and lwa must assume one of the following forms:

D/id/n	Dump begins or ends with octal location n (0-77777) in the data area specified by 1-8-character id.
Cn	Dump begins or ends with octal location n (0-77777) in the common area.
(subp)n	Dump begins or ends with octal location n (0-77777) in subprogram subp.

The area to be dumped must reside entirely within data area D/id, the common area(C), or the subprogram area (subp) of subprogram memory.

mode — Format of the dump; if mode is illegal, the SNAP card is bypassed, and an error message is written on OUT.

O	Octal
C	Character (6-bit BCD)
F	Floating-point
R	Register file
OR or RO	Octal; register file
CR or RC	Character; register file
FR or RF	Floating-point; register file

id — 1-4 BCD characters identifying the dump; these four characters precede each line of output on the dump.

The calling sequence generated by a SNAP statement is:

```
LOCATION  OPERATION, MODIFIERS  ADDRESS FIELD                COMMENTS
1       8 10                    20                           41
p                               Beginning address
p+1       Mode                  Ending address
p+2                             Identifier
p+3                             Subprogram relocation factor
p+4       Displaced
          instruction
p+5       UJP                   Return linkage address to program
p+6       RTJ                   SNAPSHOT
```

8.2.18 OCC

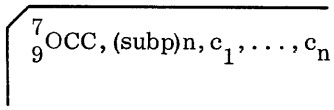
OCC is for batch use only. With an octal correction (OCC) statement, a user can change loaded subprograms by altering and adding instructions or adding and revising data.

OCC cards are generally directed to specific locations. If an OCC card is directed to the same location as a SNAP statement and follows the SNAP statement in the Real-Time SCOPE input, the SNAP dump will not be executed. No error is indicated and the SNAP calling sequence remains in available memory.

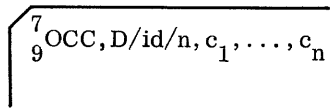
Parameters on the OCC card are free field. If an optional period terminates the statement, comments may follow. When correcting batch programs, use of absolute addresses could destroy the priority program.

The location field (first parameter on OCC card) indicates whether the OCC card:

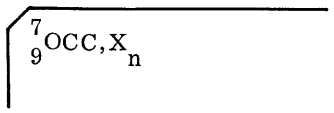
1. Sequentially corrects program instructions beginning at word location n (0-77777) in named subprogram (1-8 BCD characters).



2. Sequentially corrects data words starting at word location n (0-77777) in the specified subprogram data area.



3. Extends program execution area by n word locations (0-77777). If n exceeds available memory for subprogram, program area is extended only by the amount of memory available; a message is written on the OUT file specifying actual extension length.



4. Places information in extension area starting at word location n. This OCC card must be preceded by the card described in 3. If n is larger than the previously defined area, the corrections are not loaded.

$$\begin{matrix}7\\9\end{matrix}\text{OCC,X}_n\text{,c}_1\text{,...,c}_n$$

5. Continues corrections to program or data area begun on card described in 4, starting at n addresses (0-77777) beyond last correction on preceding OCC card. When only n appears, no addresses are skipped.

$$\begin{matrix}7\\9\end{matrix}\text{OCC,+n,c}_{x+1}\text{,...}$$

Parameters include:

$c_1,\ldots,c_n$ Octal correction fields separated by commas. A field may be blank or omitted (two sequential commas) or may contain a 1- to 8-digit octal value which may be accompanied by a positive or negative relocation factor.

Each octal value is stored right justified in a computer word. When the value occupies fewer than eight digits (leading zeros omitted), the loader zero-fills the remainder of the word. Blanks in a correction field are ignored; when a field is blank or omitted, the location represented by the field is unchanged.

The loader stores a correction as it appears, or with word or character relocation of the address, into the memory word determined by the location field and the position of the correction field on the card.

Fields may be combined:

correction	Stores correction by word address as it appears in field without address relocation
correction reloc factor	Stores correction by word address, relocating address field positively relative to relocation factor

correction-reloc factor	Stores correction by word address, relocating address field negatively relative to relocation factor
correction reloc factor C	Stores correction by word address, positively relocating address field as character address relative to relocation factor
correction-reloc factor C	Stores correction by word address, negatively relocating address field as character address relative to relocation factor

Relocation factors:

(subp)	1-8-character subprogram name enclosed in parentheses. The loader relocates the address field relative to the first word address of the named subprogram.
*	The loader relocates the address field relative to the first word address of the last subprogram named in this series of OCC cards.
C	The loader relocates the address field relative to first location of common.
D/id/	Loader relocates the address field relative to the first word address of a data block, identified by 1-8 BCD character id.
X	Loader relocates the address field relative to the first word address of the program extension area.

8.3 OCC AND SNAP

Compile FORTRAN or assemble COMPASS source subprograms; load them along with binary object decks; insert octal changes; request SNAP dump; execute.

BINARY DECK STRUCTURE 9

The Real-Time SCOPE loader loads binary object decks structured according to specifications; assemblers and compilers produce object decks which meet these specifications.

9.1 LOADER CARDS

The loader accepts binary cards in the following order.

IDC Program identification

EPT Entry point name (may appear anywhere between IDC and TRA)

RIF Relocatable information

LRL Local reference list (optional)

XNL External name (optional)

TRA Transfer

The above cards will be produced by assemblers and compilers; the following cards are not part of assembler or compiler output, but the programmer may insert them into binary decks.

LED Loader equipment declaration

EXS External symbol declaration

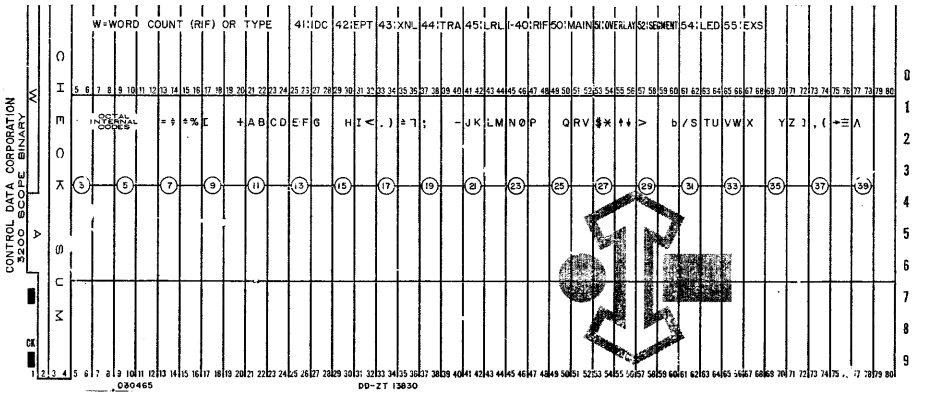
ELD End loader declaration

Overlay preparation

Library preparation

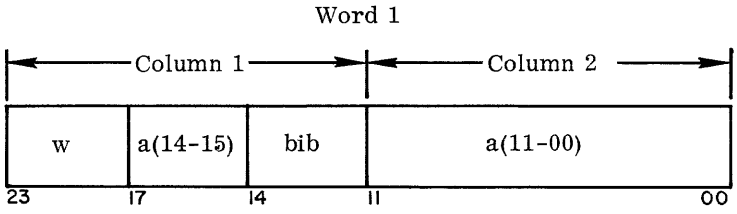
9.2 BINARY CARD STRUCTURE

The number of columns used on a card varies among card types. The first four columns, which are always used, conform to the format shown. The 7 and 9 indicating a binary card, are always punched. The remaining fields are punched as required by the card-type specifications. Common to most cards are the word count field, the address field, and the checksum field.



The fields in columns 1-4 of all cards except the ELD and LED cards must meet certain specifications.

1. Columns 1 and 2 form one 24-bit computer word with the following format:



w — 2 octal digits identifying card type; on an RIF card, w also indicates word count.

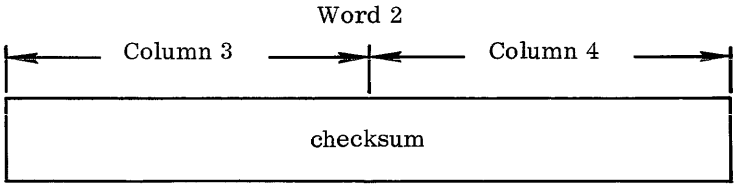
a — Bits 17-15, 11-00, form a 15-bit value whose significance varies with the card type.

b — Bits 14 and 12 (7,9 punch in column 1) are always punched; they identify the card as binary.

i — Bit 13 (8 punch in column 1)

Unpunched	Checksum used by loader
Punched	Checksum ignored by loader

2. Columns 3 and 4 of the card form one 24-bit word with the following format:



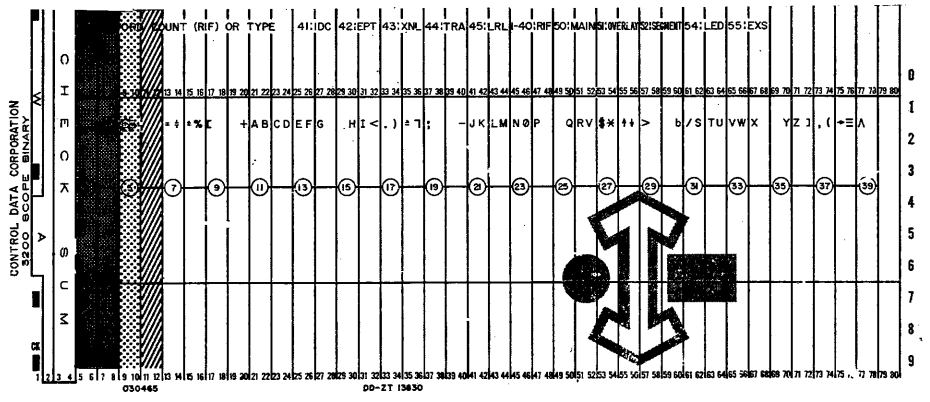
checksum — 24-bit sum with end-around carry of all other columns on card (1,2,5-80)

3. Columns 5-80 contain information which varies with the card type.

9.2.1 IDC CARD

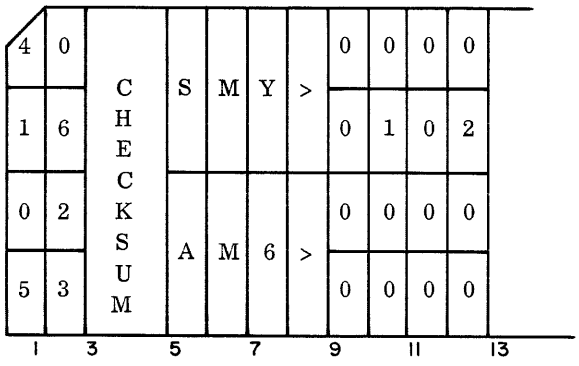
The subprogram identification card provides the loader with the subprogram name and the amount of core storage required for the subprogram with its associated data and common areas.

In the stacked binary decks, the data reserved by the first IDC card is the maximum area available to subsequent subprograms.



w	41_8, identifies card as an IDC card.
a	Octal 0-77777; subprogram length, excluding data and common.
columns 5-8	Subprogram name; 1-8 internal BCD characters left justified. If the name has fewer than 8 characters, octal 60's complete the field.
columns 9, 10	Number of storage locations reserved for common; must be unpunched in priority program.
columns 11, 12	Number of storage locations to be reserved for data.
columns 13-80	Not used.

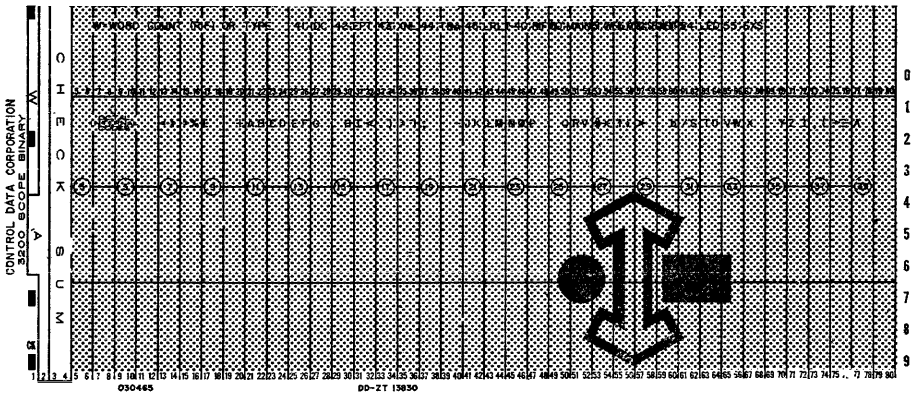
Example:



The subprogram SAMMY6 is 623_8 words long; common is 100_8 words long, and data is 200_8 words long.

9.2.2 EPT CARD

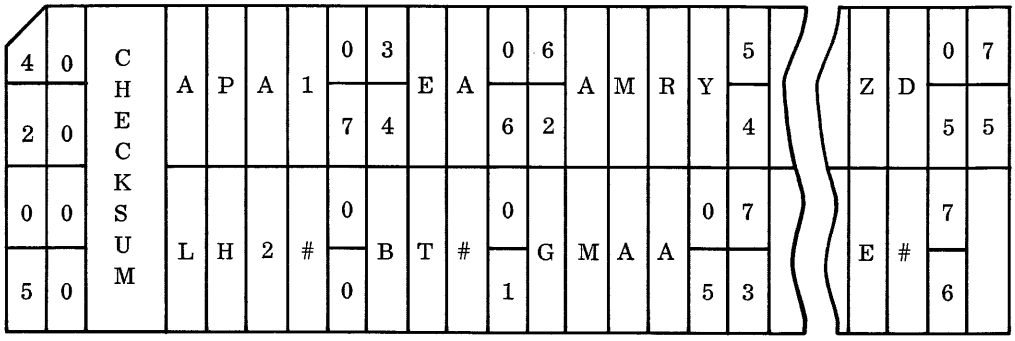
An entry point name card notifies the loader that the listed locations may be referenced by programs other than the one currently being loaded.



w	42_8; word count for EPT card
a	Sequence number, right justified in the field when deck contains more than one EPT card. First EPT card has sequence number 0; cards must be sequential in the deck.
columns 5-80	Variable length fields, each containing an entry point name in BCD and its address.
entry point name	An entry point name may be 1-8 alphanumeric characters; when it is fewer than 8 characters, 72_8 terminates the field.
entry point address	The address field for each entry point name begins after the last character of the name or after 72_8 and is 18 bits long.

The address, with any leading zeros, occupies the least significant 15 bits of the field. If bits 18-16 are 0, the address is a word address; if bits 18-16 are not 0, the loader converts the address to a character address. Entry point name and address fields may not be continued from one card to another.

Example:

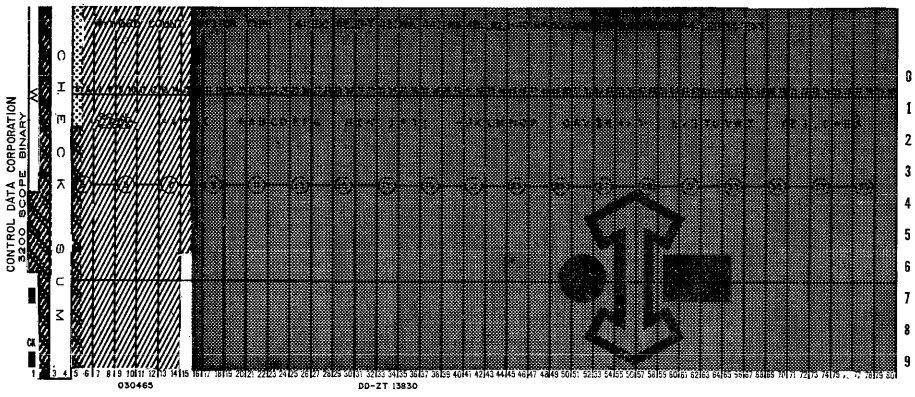


Entry Point Name	Address
ALPHA21	70034
BETA	60162
GAMMARAY	55473
⋮	⋮
ZED	57675

9.2.3 RIF CARD

Each relocatable information card provides the loader with 1 to 32 relocatable storage words and the means for computing absolute addresses of these words at load time.

Program or data relocatable address indicator:



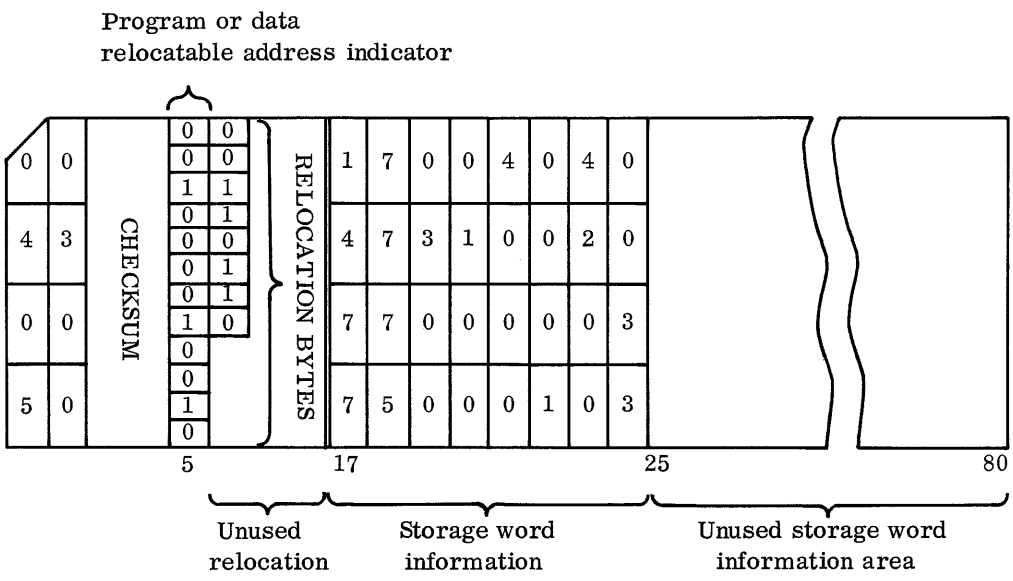
w	$1\text{-}40_8$, number of words on card
a	Relocatable address of information stored in columns 17-18
column 5, rows 12, 11, 0, 1	This byte is 2 if the contents of the A field is a program relocatable address; it is 4 if the field contains a data relocatable address
columns 5-15	Up to 32 4-bit relocation bytes

Modification of storage word information by relocation bytes:

Relocation Byte	Significance
x000	Unused; constitutes an error
x001	No modification (absolute address)
x010	Subprogram increment
x011	Common block increment
x100	Data block increment
x101	Subprogram decrement
x110	Common block decrement
x111	Data block decrement

columns 17-80	Up to 32 storage words; the address portion of the first of these words is modified by the first relocation byte in the relocation field; the second word is modified by the second relocation byte, etc. The first word is then stored at the relocation address in the A field; the second word is stored at relocatable address (A) + 1, etc.

Example:



A	Relocation Byte	Word
00300	1 0001	1 14777775
	2 0010	2 03000100
	3 0011	3 40000001
	4 0110	4 42000033

If the subprogram to which this RIF card pertains begins at:

60000

DATA at: 60500

COMMON at: 60600

The results are:

(60300) = 14777775

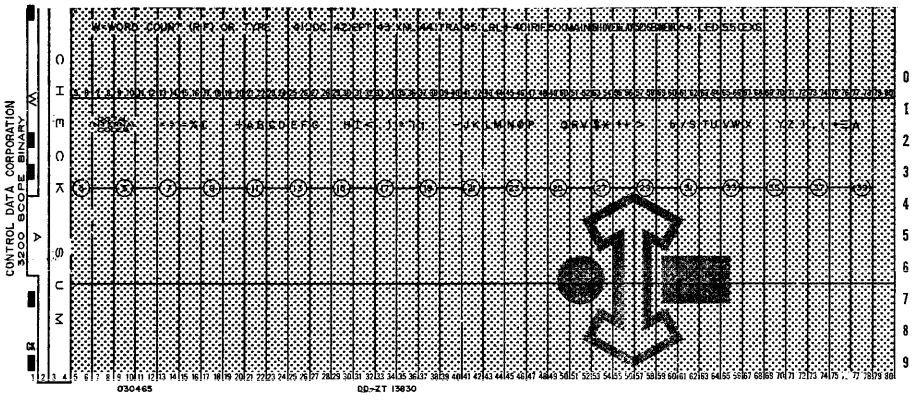
(60301) = 03060100

(60302) = 40060601

(60303) = 42060545

9.2.4 XNL CARD

An external name card lists locations referred to in the subprogram which are entry points of other subprograms.



w	43_8 word count for XNL card.
a	Sequence number, right justified in field when deck contains more than one XNL card. First XNL card has sequence number 0; cards must be sequential in deck.
columns 5-80	Variable length fields, each containing an external name and the last address in the subprogram at which the external name is referenced.
external name	An external name may be 1-8 alphanumeric characters; when it has fewer than 8 characters, it is terminated by a 72_8.
external name address	The address field for each external name begins after the last character of the name or after 72_8 and is 18 bits long. The address, with any leading zeros, occupies the low-order 15 bits of the field. The most significant 3 bits are zero for a word address and nonzero for a character address.

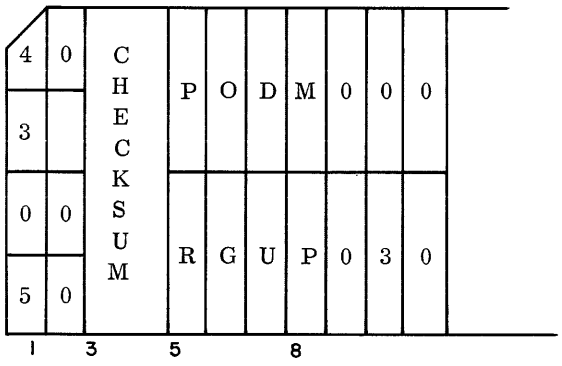
If the symbol is declared external but is not otherwise referenced in the subprogram, the address on the XNL card is 77777_8.

The address given on the XNL card may specify another location in the subprogram where reference is made to the external name. A series of reference addresses is called an external string. The low-order 15 bits of the last entry in the string contain 77777_8. External strings may run in any order through the subprogram.

All external references are to subprogram relocatable word addresses. However, the external reference may be made by either a word- or character-type instruction. All entries in a string are either from word-type instructions or from character-type instructions.

XNL cards may not be in the first position in the subprogram deck. An external string may refer to previously encountered external symbols only after the relocatable information has been loaded for them.

Example:



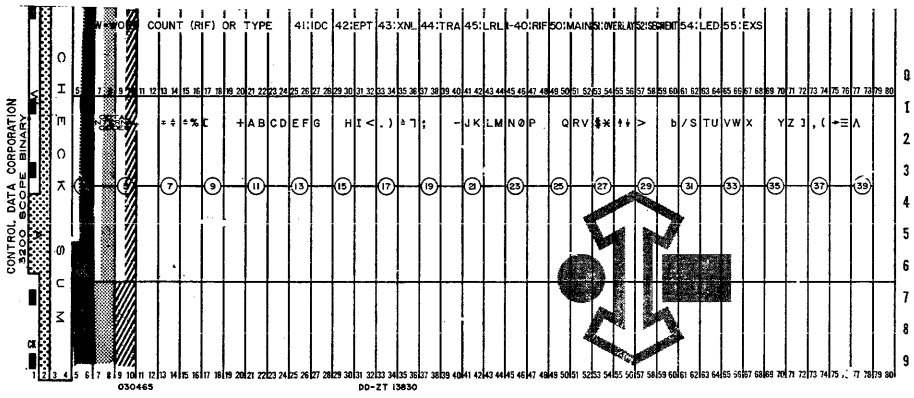
The external name PROGDUMP was last referenced in the subprogram at relocatable address 00300.

9.2.5 LRL CARD

A local reference list (LRL) card establishes a reference string when a program refers to a term before that term is defined. It is produced only by the assemblers and compilers that list it as binary output.

The location given in cia is the last relocatable address at which the symbol is referenced. The address portion of this instruction contains the relocatable address at which the symbol was previously referenced. This string continues until the location first referencing the symbol is reached. The address portion of this word contains 77777_8.

If the contents of n do not equal the number of references to the symbol, an SL (string loop) error (appendix H) is indicated.

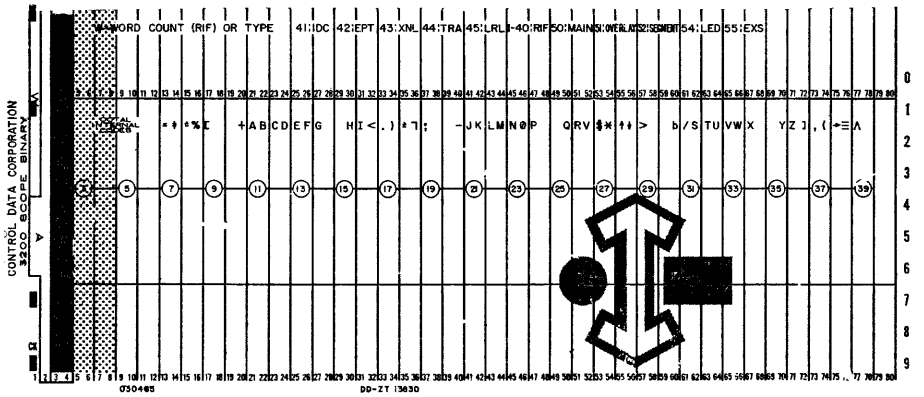


w	45_8 word count for LRL card
a	Word/character flag; a=0 word address string a≠0 character address string
rac	17-bit field in rows 5-9 of column 5 and all of column 6; contains the relocatable address at which the referenced term is defined. The upper 2 bits are nonzero if that address is a character address; if it is a word address, the upper 2 bits are zero. Positions in column 5 to the left of the rac field are zero filled.
cia	15-bit field in rows 7-9 of column 7 and all of column 8; contains the relocatable address of the last reference to the term in the subprogram.
n	15-bit field in rows 7-9 of column 9 and all of column 10; contains octal digits indicating the number of times the term was referenced.

9.2.6 TRA CARD

The transfer card (TRA) indicating the end of the subprogram appears at the end of the subprogram deck. A deck may contain multiple TRA cards, but no more than two TRA cards may contain transfer point symbols. When Real-Time SCOPE gives control to the loaded program, control goes to the most recently encountered transfer point.

When loading from the library, the first TRA card encountered terminates loading.



w	44_8, word count for TRA card
a	Not used
checksum	Sum of all checksums on the other cards in the binary deck
columns 5-12	Transfer symbol expressed in Hollerith. The transfer point symbol must appear as an entry point name in the loaded program deck or in a library subprogram.

9.3 INSERTED CARD FORMATS

The programmer may prepare the following cards and insert them into binary decks; loader equipment declaration (LED), external symbol (EXS), and loader declaration (ELD), and overlay control cards. The overlay cards are discussed in chapter 10.

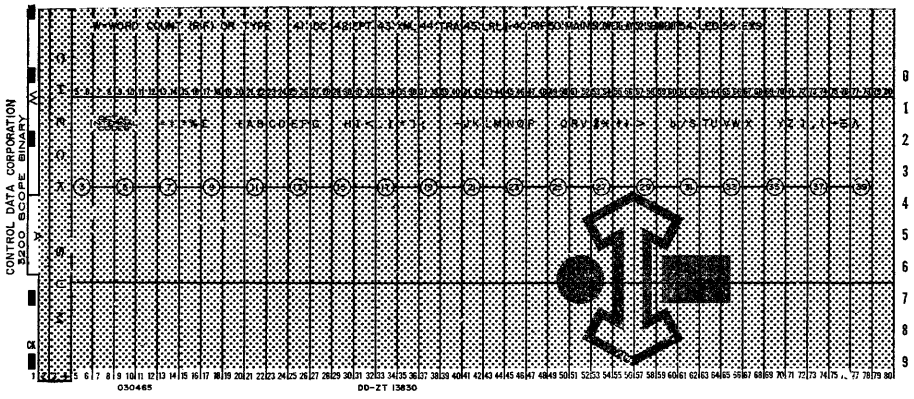
LED and EXS cards have identification codes of 50_8 or greater and contain 7, 9 punches in column 1. Other columns contain symbolic information in Hollerith code.

9.3.1 LED CARD

A loader equipment declaration card assigns logical units to specific hardware units or to hardware of a particular type. If a logical unit named on an LED card has been assigned previously, the LED declaration is ignored. Units assigned by LED cards cannot be declared by EQUIP statements within the same run. A card may contain one or more hardware declarations.

Units assigned by LED cards are logged on CTO and the operator is asked: READY?

Priority programs may not contain LED cards.



w	54_8, word count for LED card	
columns 2-80	uuhh uuhheeuu	Parameters appear in Hollerith code, occupying 4 or 8 columns.
	uu	Logical unit, 01-56
	hh	Hardware type, $01\text{-}37_8$ (section 2.4)
	c	Channel number, 0-7
	e	Equipment, 0-7
	uu	Unit designator, $00\text{-}77_8$

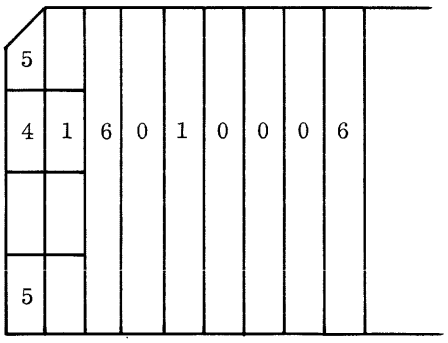
Parameters may be blank, if so Real-Time SCOPE assigns an available unit of the hardware type specified.

Declarations are separated by commas and embedded blanks are ignored. Any number of LED cards may appear in a program deck.

If the logical unit indicated in a declaration has been previously assigned to a physical unit, the declaration is ignored.

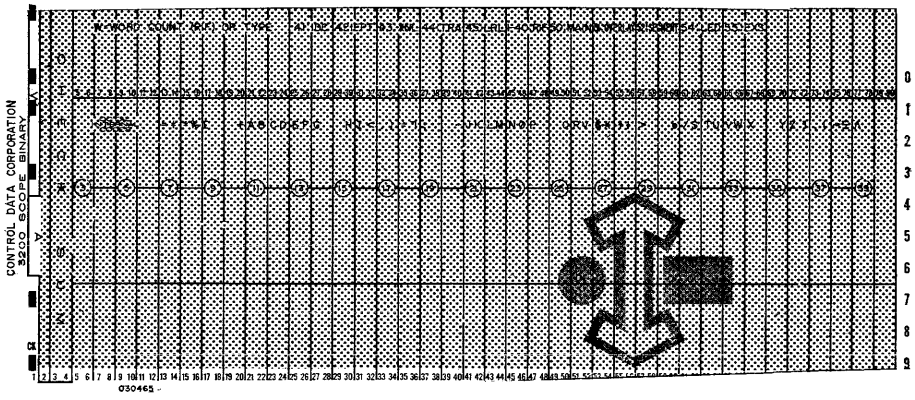
Example:

This card assigns logical unit 16 to the magnetic tape designated by the hardware code C0E0U06.



9.3.2 EXS CARD

An external symbol card declares symbols not listed on an XNL card or equates one or more external symbols to a single entry point.



w	55_8, word count for EXS card
columns 2-80	Contain Hollerith information. To declare symbols external, the form is: external symbol, external symbol,...,external symbol To equate symbols to an entry point, the contents are: external symbol, external symbol,...,external symbol=entry point

When EXS declares SNAPSHOT external, a SNAP control card (section 8.2.17) must provide linkage. When EXS declares other subroutines external, the source program must provide linkage information.

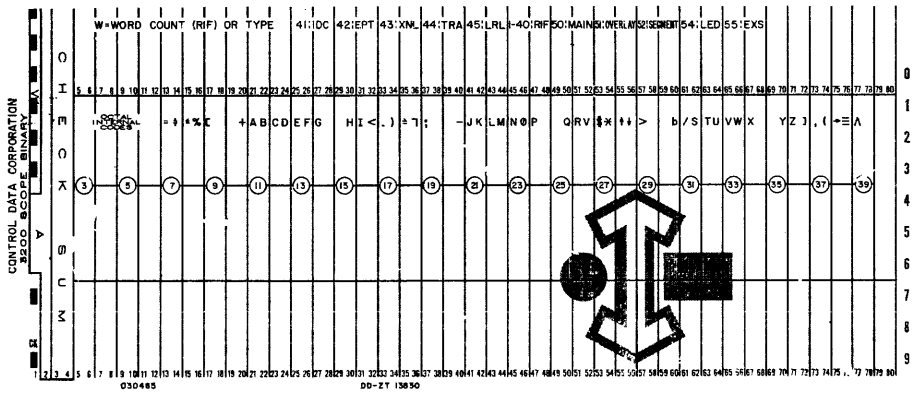
If EXS is used to equate symbols to an entry point, only the program containing the entry point is loaded.

EXS declarations override later EPT declarations. If a subsequent entry point name is identical to an external symbol in an EXS declaration, a duplicate symbol error results.

9.3.3
ELD CARD

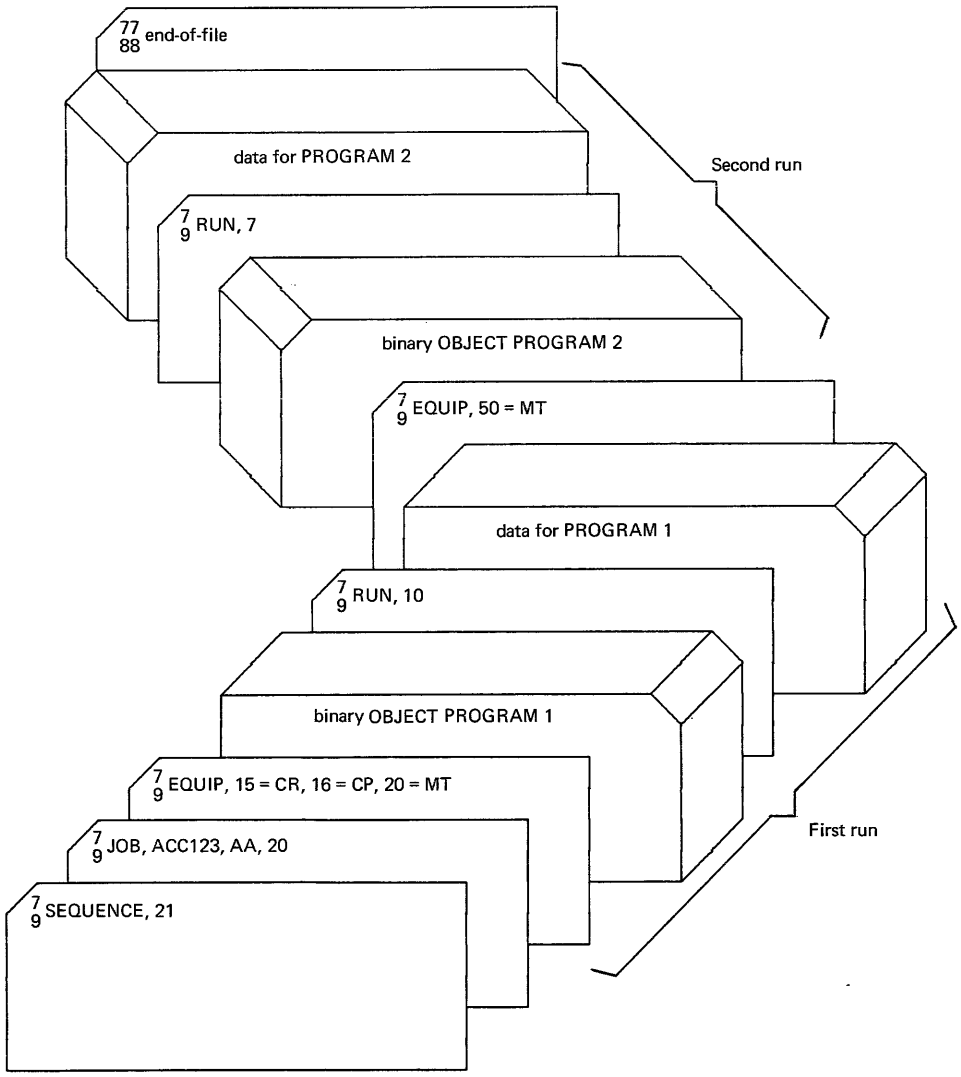
An end loader declaration card has a word count (w) of 77_8. No other columns are punched.

This card is inserted at the end of a binary deck to be loaded from INP when the programmer wants loading terminated but is unable to follow the deck with a Real-Time SCOPE control card or an end-of-file card. For example, this option permits a batch user program to call the loader to load a relocatable binary deck. Calls to the loader are discussed in appendix D.



Load and Execute Two Binary Decks

A job can include one or more runs. Programmer units must be defined prior to their use; scratch units must be defined prior to their use in each run.

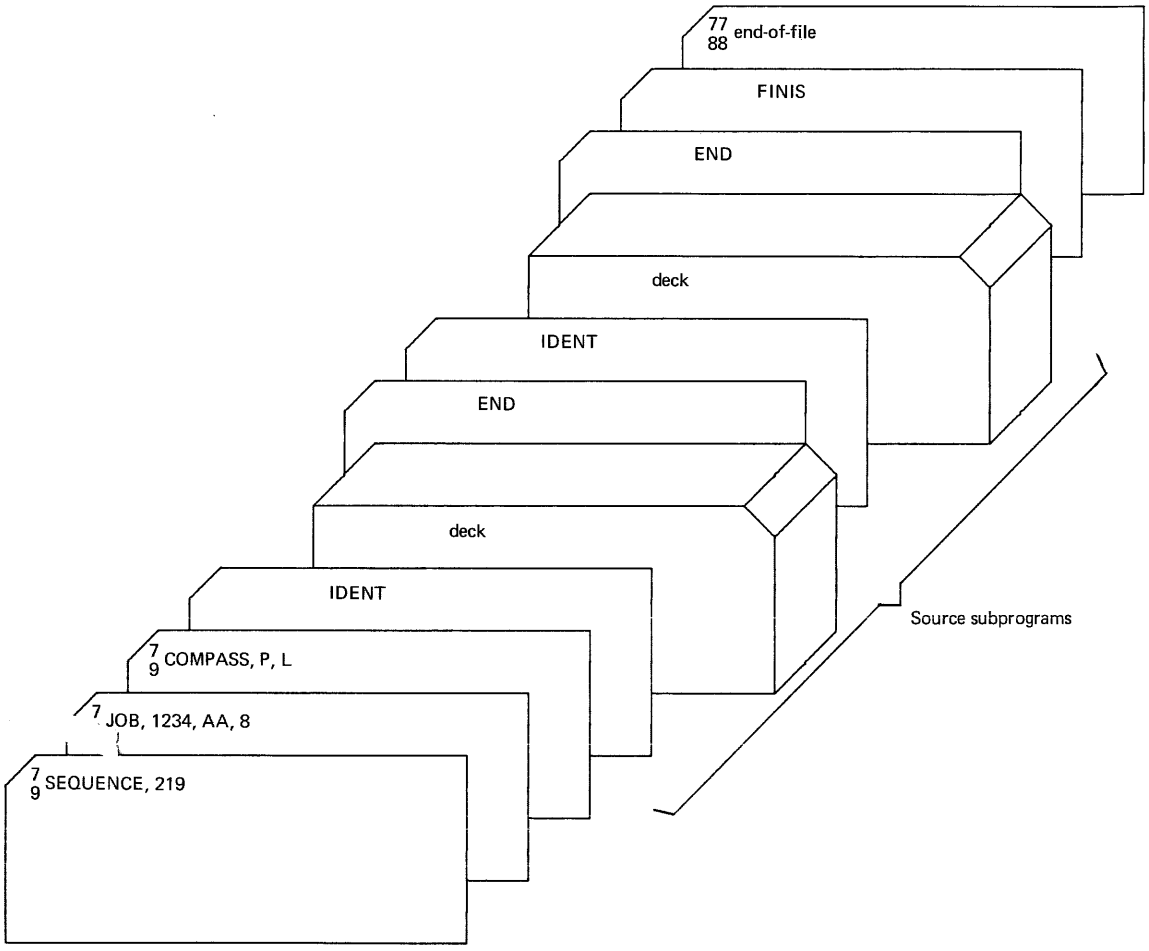


Logical unit 15 is assigned to the card reader, 16 to the card punch, and 20 to magnetic tape for the duration of the job.

Binary object decks on INP are automatically loaded.

The RUN cards specify execution of the loaded programs for each run.

COMPASS Assembly; Punch and List Only



COMPASS is called from library.

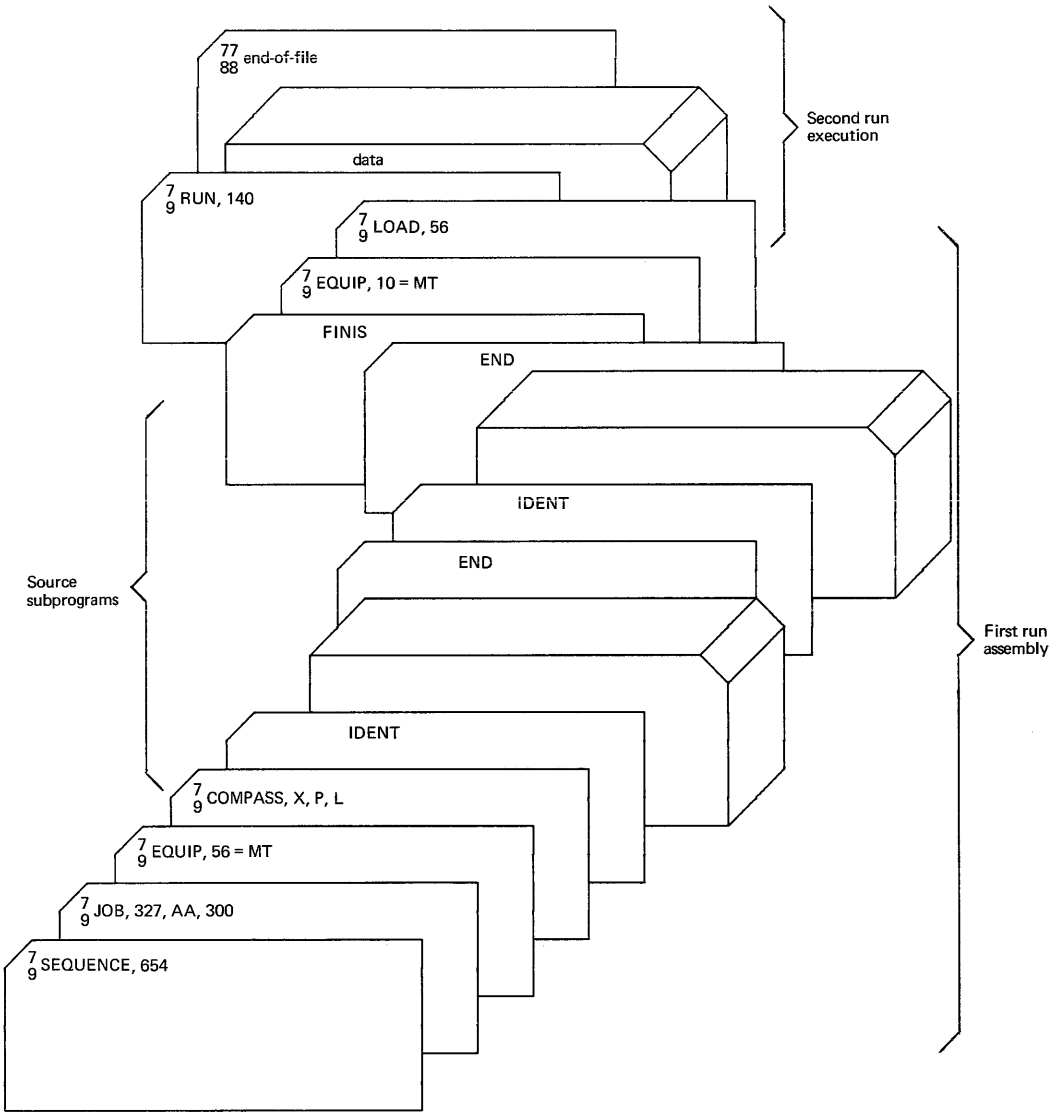
Source input is standard input unit (60).

No binary object program is written on LGO.

Relocatable binary deck is punched on standard punch unit (62).

There is no execution of object program.

COMPASS Assembly; Load and Execute Newly Generated Decks



Source input is standard input unit (60).

Binary object program is written on load and go (LGO); end-of-file mark is written after last subprogram.

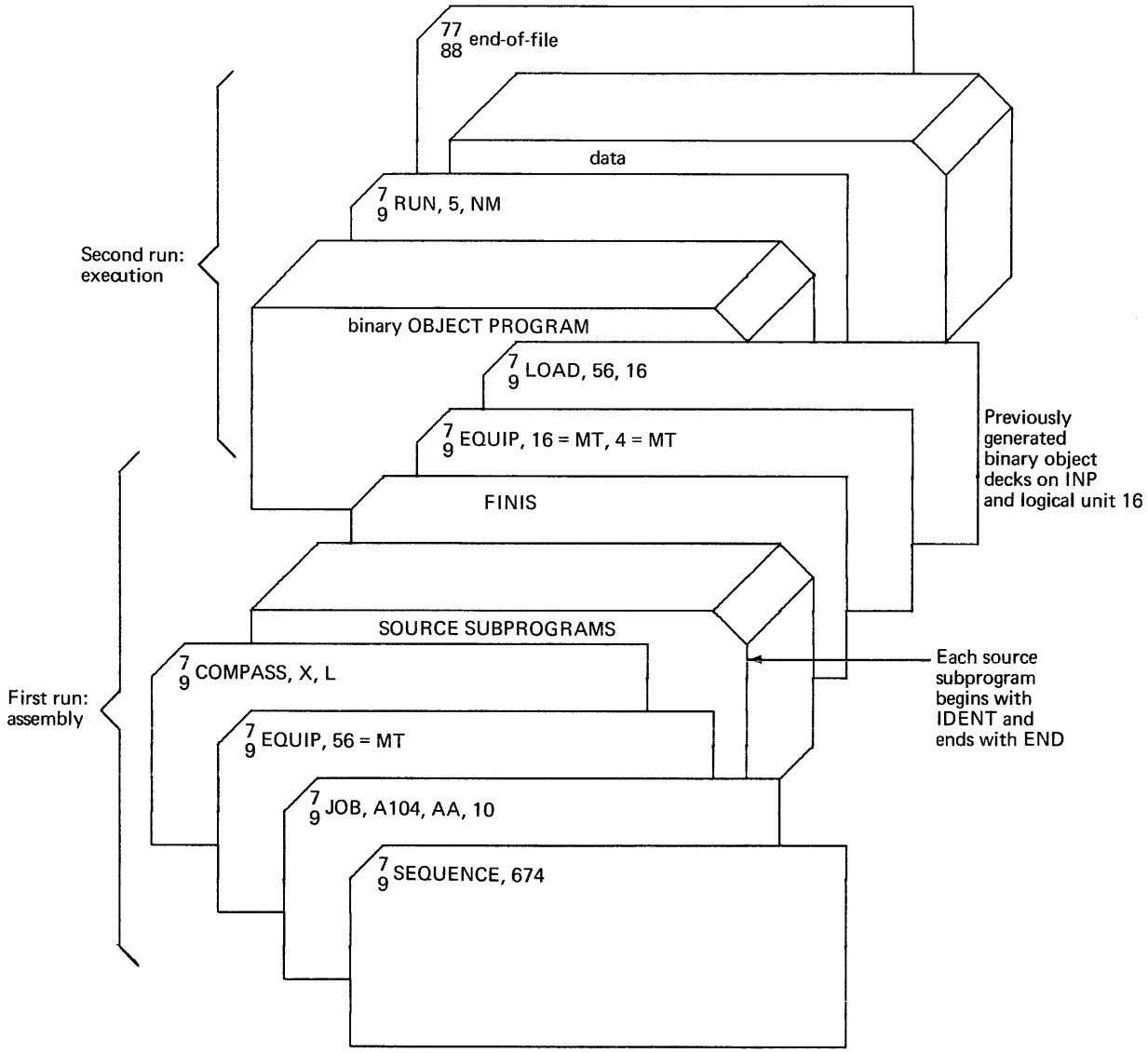
Binary deck is punched on standard punch unit 62.

Source and object programs are listed on standard output unit (61).

The binary programs are loaded from 56.

The RUN card specifies execution of loaded programs.

COMPASS Assembly; Load and Execute New and Previously Generated Decks



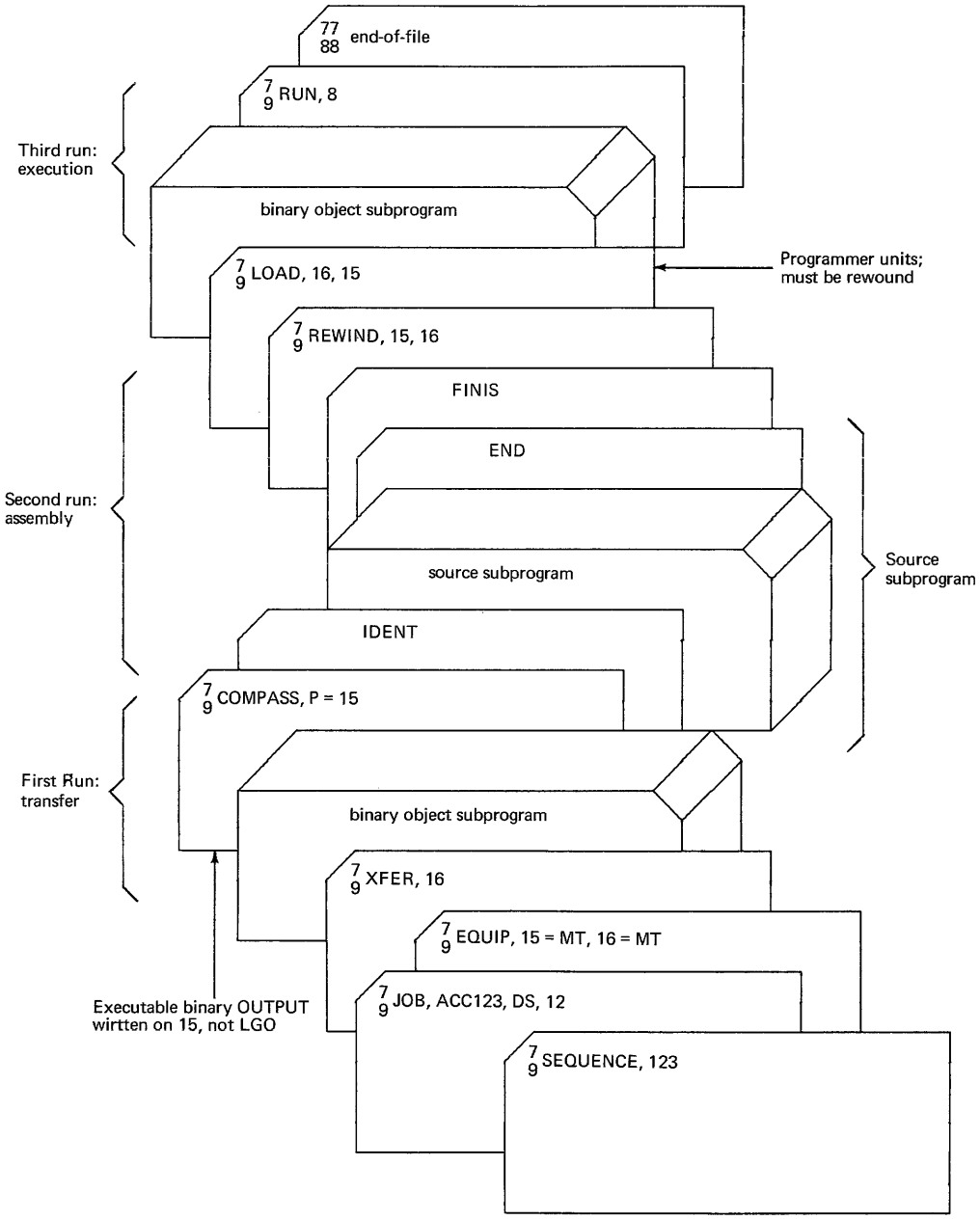
COMPASS is called from LIB to assemble source subprograms on INP.

Executable binary output is written on the load-and-go device (56).

The object programs are loaded from 56, unit 16, and from INP.

The RUN card specifies execution of the loaded object programs.

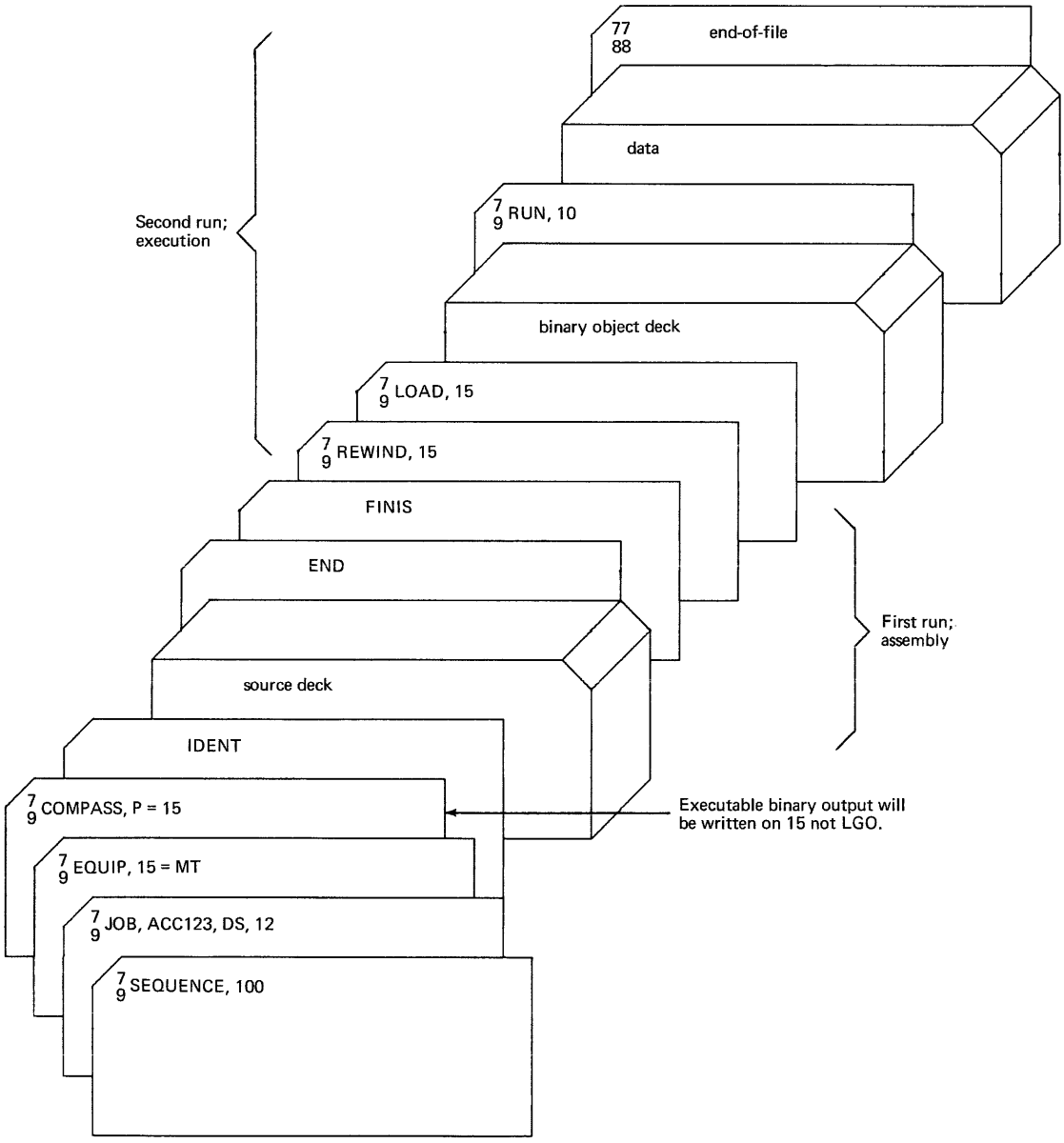
Transfer Binary Object Subprograms to Tape; Assemble COMPASS Subprograms; Load These and Another Binary Object Deck; Execute



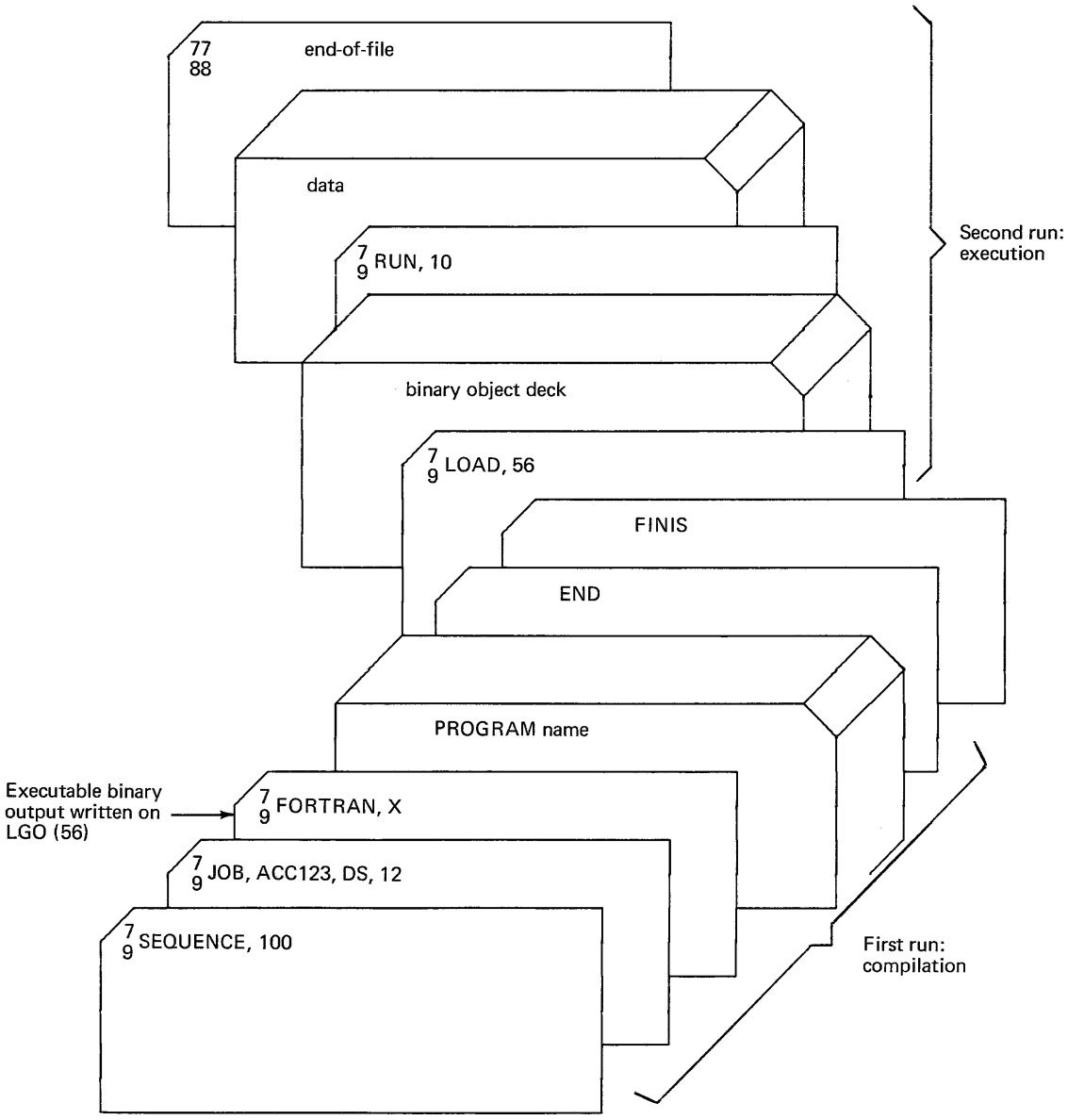
Because a XFER card causes u (if unassigned) to be assigned to an available magnetic tape, it can precede EQUIP cards. A XFER deck is a separate run.

Units 15 and 16 are rewound in the order specified on the cards.

Assemble and Execute



Compile and Execute



OVERLAYS 10

A batch or priority program that is larger than available core may be converted to an overlay program. In Real-Time SCOPE, an overlay program is a hierarchy of independently assembled subprograms loaded into available memory. The subprograms communicate through data areas. The controlling subprogram is called the main element of the program.

The main program resides in core during execution of all the related subprograms. The other elements are called overlay elements (or overlays) and segments. The main program sequentially calls overlays into core.

The main program:

- Must declare EXECOVR an external name
- Must call at least one overlay
- Must refer only to addresses defined within itself or resident
- Must be assembled or compiled independently

An overlay element:

- Must refer only to addresses defined within itself, in resident, or declared as entry points in the main program
- Must declare EXECOVR external if the overlay calls a segment
- Must be assembled and compiled independently

Each overlay is loaded immediately below the main element and destroys some or all of a preceding overlay. An overlay may perform operations on data produced by a preceding overlay.

Each segment element is called sequentially by its controlling overlay. A segment is loaded immediately below the overlay to which it is subordinate; hence, loading of a segment destroys some or all of a preceding segment in core.

A segment:

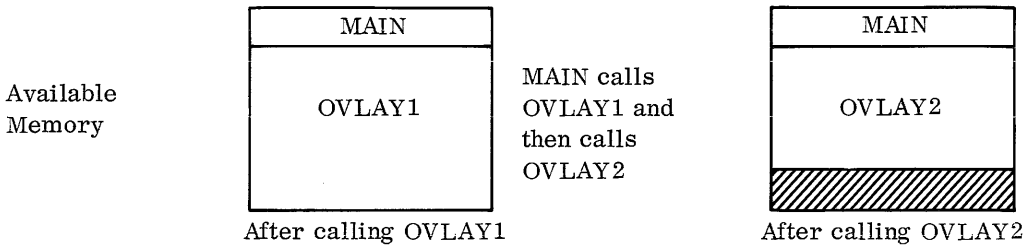
- Must be assembled or compiled independently
- May reference addresses in resident, entry points defined in the main program, or entry points defined in the overlay to which the referencing segment is subordinate

The user identifies the elements of the overlay program to the Real-Time SCOPE loader through the MAIN, OVERLAY, and SEGMENT control cards. At least one overlay card must follow a main card.

The main element may call a maximum of 99 overlays. Each overlay may call a maximum of 99 segments. During execution, the main program, one overlay, and one segment may reside concurrently in core.

Each element of the program may be followed by OCC cards when the element is loaded from INP. Programs on the library may be composed of main, overlay, and segment elements, but may not be followed by OCC cards; an OCC card on the library results in a card sequence error.

Program with Two Overlays:



Program with Three Segmented Overlays:

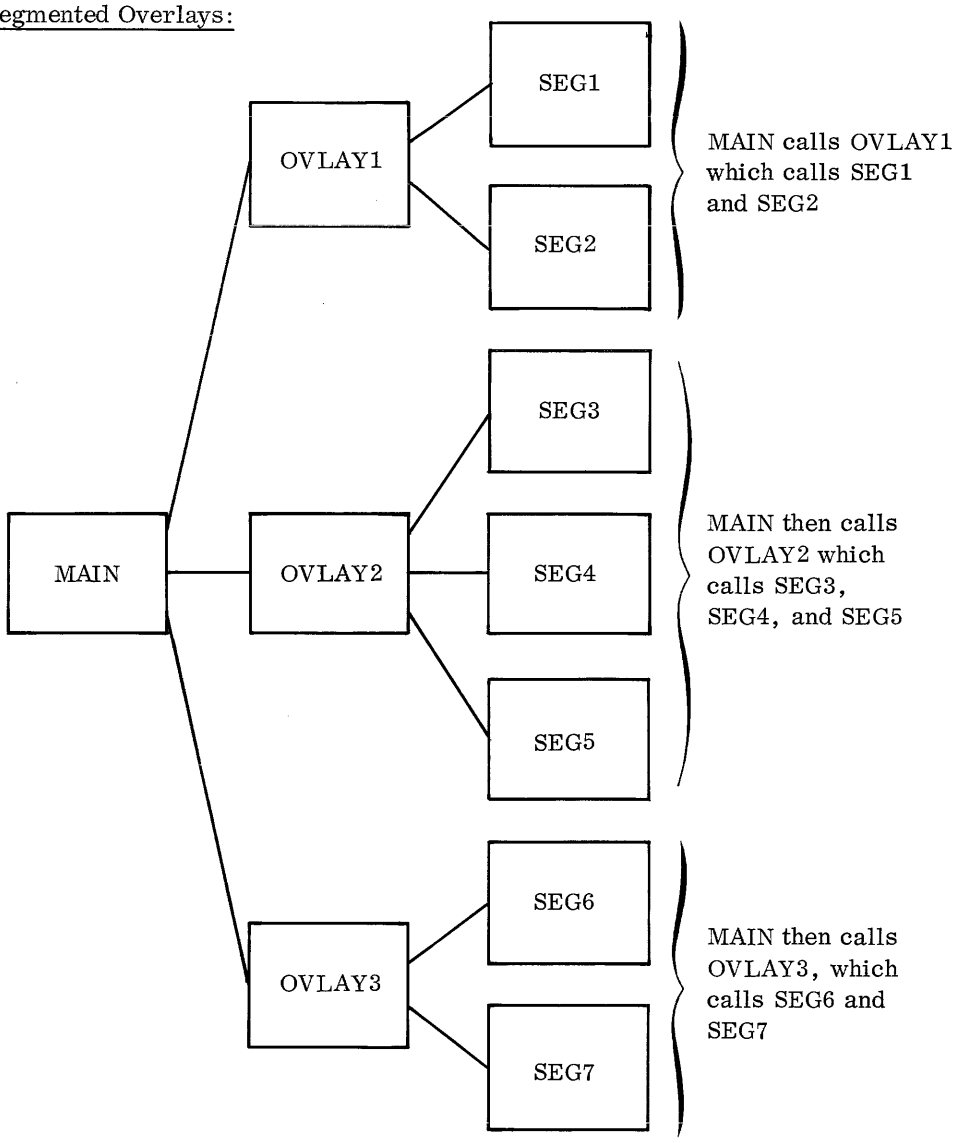


Figure 10.1 Overlays and Segments

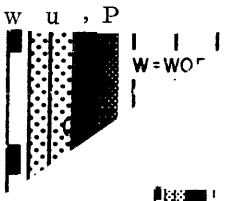
10.1 OVERLAY CONTROL CARDS

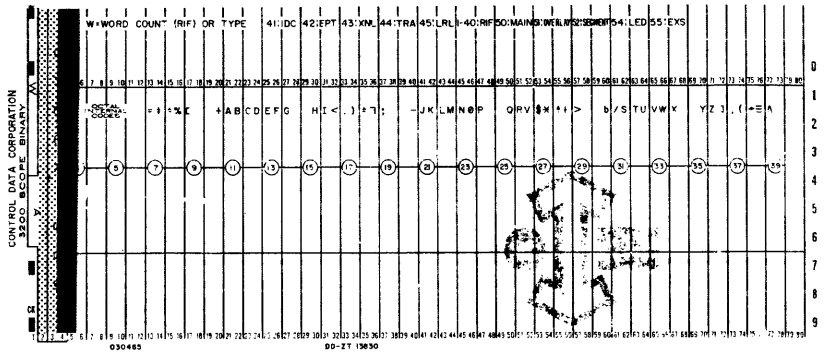
An appropriate MAIN, OVERLAY, or SEGMENT control card must precede each element of the program. The programmer prepares and inserts an overlay control card for each overlay and segment. The user may suppress the memory map of overlays or segments by means of overlay control card parameters.

Control card parameters specify the tape (logical unit 1-55) on which main, overlay, and segment elements are to be written and from which they are called for execution. No other logical unit is legal.

10.1.1 MAIN CARD

A MAIN card must be the first card encountered by the loader when overlays are used. The binary deck (IDC-TRA) following this card and preceding the next overlay control card constitutes the main program.



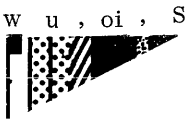


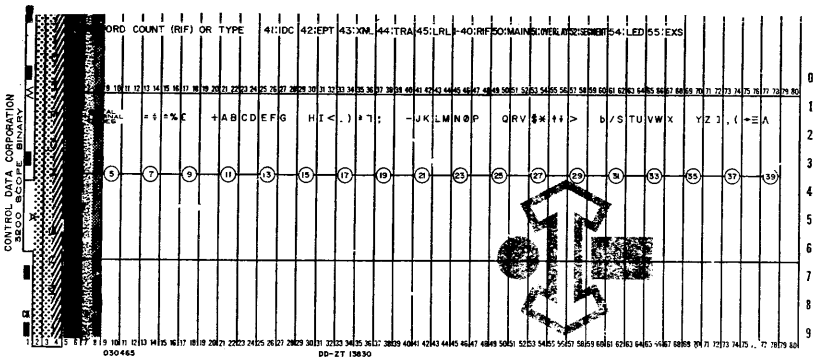
Columns	Field	Use	Contents
1	w	MAIN card identification	50_8
2,3	u	Logical unit used to store this element	Hollerith decimal digits 1-55, designating an open tape on which the main element and an EOF are to be written. If 0,00, blank, or comma, the main program will not be written on a tape.

Columns	Field	Use	Contents
4	,	Field separator	Comma (Hollerith) or blank
5	P	Octal correction indicator	P (Hollerith) If an OCC is detected and P is not declared, a card sequence error (CS) is indicated.

10.1.2 OVERLAY CARD

The appearance of this card on INP signals the beginning of an overlay element. The overlay consists of all subprograms between this card and the next overlay control card, or between this card and the first following Real-Time SCOPE control card if no overlay control card intervenes.



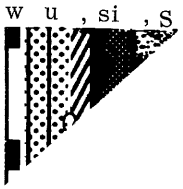


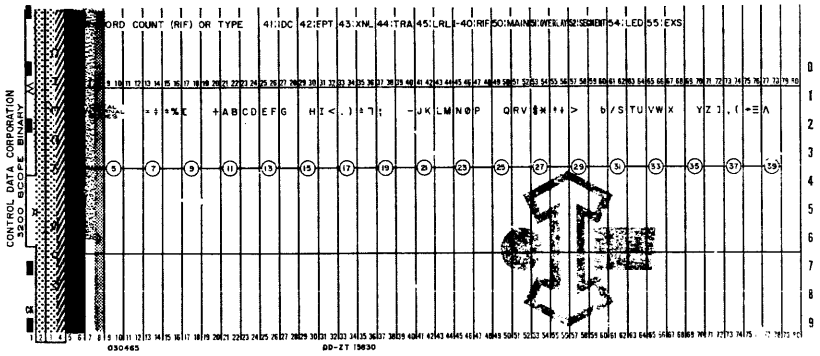
Columns	Field	Use	Contents
1	w	OVERLAY card identification	51_8
2,3	u	Logical unit used to store this element	Hollerith decimal digits 1-55, designating an open tape unit
4	,	Field separation	Comma (Hollerith)
5,6	oi	Overlay identification	$01\text{-}99_{10}$ (Hollerith)
7	,	Field separator; if missing end of data	Comma (Hollerith) or blank
8,9	S	Overlay map suppression	S (Hollerith); optional

10.1.3 SEGMENT CARD

This card signals the beginning of a segment that is subordinate to the overlay element defined by the most recently encountered overlay card. The segment consists of all subprograms between this card and the next overlay control card on INP, or between this card and the first following Real-Time SCOPE control card if no overlay control card intervenes.

If a SEGMENT card is prior to the first OVERLAY card, RTS processes the segment as an overlay element and writes a card sequence error message on OUT (appendix H).

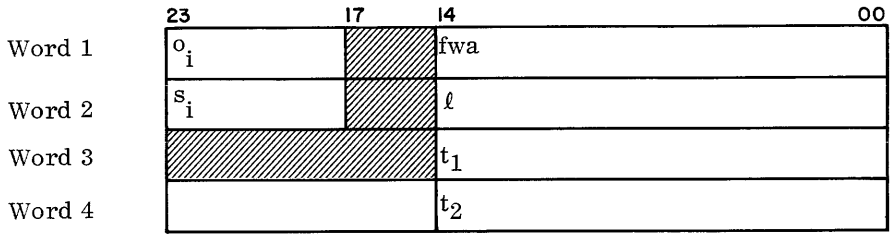




Columns	Field	Use	Contents
1	w	SEGMENT card identification	52_8
2,3	u	Logical unit used to store this element	Hollerith decimal digits 1-55, designating an open tape unit
4	,	Field separation	Comma (Hollerith)
5,6	si	Segment identification	$01\text{-}99_{10}$ (Hollerith)
7	,	Field separation or end of data	Comma (Hollerith) or blank
8	S	Segment map suppression	S (Hollerith); optional

10.2 OVERLAY IDENTIFICATION

The Real-Time SCOPE loader prepares overlays and segments as directed by overlay control cards and stores these elements on the designated logical unit. On completion of this processing, a five-word identifying record precedes each program element on tape.



t_1	Primary (most recently encountered) transfer address
t_2	Secondary transfer address
o	Overlay identification, $001\text{-}143_8$
s	Overlay o segment identification $00\text{-}143_8$
fwa	First word address for record
ℓ	Length of record in words

o_i and s_i are both 000 if the record is the main program.

The record consists of absolute binary words read into fwa through fwa + ℓ -1. The last information record on each overlay or segment tape is followed by an end-of-file.

10.3 MAPPING OF OVERLAYS

When an overlay or a segment is prepared, a map of the memory is produced on the standard output unit, unless suppression is indicated by an S in the third field on the overlay or segment card. The map shows:

First word address of each subprogram comprising overlay or segment

All entry points defined within the overlay or segment

Overlay or segment extension, if any, to contain corrections

Each load map for an overlay and segment begins a new page. The following heading appears on the first line of each page.

" OVLAY o_i SEG s_i TAPE u

o_i Identification number (decimal) of this overlay or of the last preceding overlay if this is a segment map

s_i Identification number (decimal) of this segment or 00 if this is an overlay map

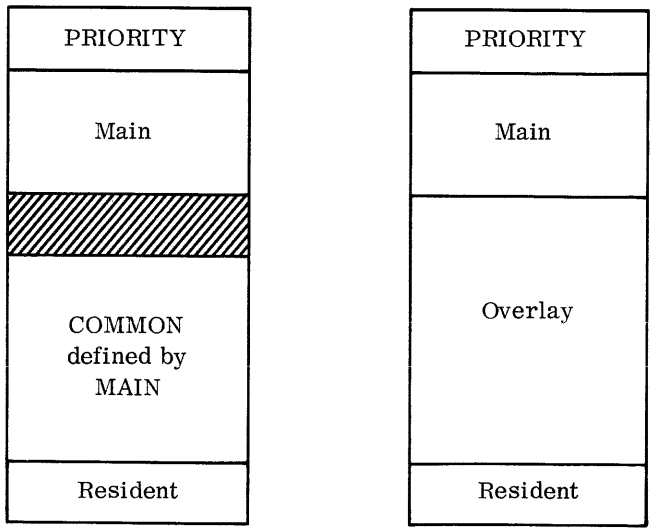
u Logical unit (decimal) of the tape on which this element was written. Illegal tape designations produce a card format error message and u appears as 00.

Similarly, illegal values of o_i or s_i produce an error message, and the heading line of the corresponding map shows the illegal values as 00.

10.4 COMMON

Each element of a batch program may define a common block. The length of common is the greatest length defined by any program element in core unless some element of the program is loaded into the common area.

An element of the program that does not require common need not define it. It is possible that an overlay element that does not define common may be loaded in an area of common used by some other element, as shown in the following diagram.

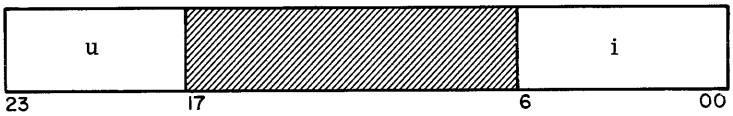


10.5 DATA

Each element of a batch or priority program prepared in overlays may define a data block and load data into this block. When a program element defines a data block, only the defining element and elements subordinate to the defining element may use that data block. Only the defining element may preset the area. Thus, if the main element defines a data block, the main element may load data into the area; main, all overlays, and all segments may refer to this data. If an overlay defines a data block, the defining overlay may preset the area, and only that overlay and its subordinate segments may refer to this data. If a segment defines a data block, only the defining segment may preset or refer to the data.

10.6 OVERLAY EXECUTION

To call a subordinate element, the main element or an overlay places the following in the A register:



- u Logical unit on which overlay or segment is stored
- i Identifying number of overlay (o_i) or segment (s_i) $001\text{-}143_8$

When the parameters are in A, the next instruction is:

```
RTJ  EXECOVR
```

EXECOVR locates the subordinate element, loads it and passes control to it. When return is made from a segment, EXECOVR passes control back to the overlay which last called EXECOVR. When return is made from an overlay, EXECOVR passes control to the main program.

When EXECOVR cannot locate a requested element or cannot read the file without error, the job terminates abnormally.

EXECOVR must be declared as an external symbol in the main program and in each overlay that calls a segment.

PROGRAM DEBUGGING 11

Real-Time SCOPE provides the following debugging aids for use at source language or object deck level.

- Memory allocation print (MAP)
- SNAP control card for dumping selected portions of memory during program execution (section 8.2.17)
- Source language calls for system memory dumps
- Recovery dumps on abnormal termination of a job
- Octal correction routine modification or extension of loaded programs (section 8.2.18)

11.1 MEMORY ALLOCATION PRINT

When it processes a RUN statement, Real-Time SCOPE automatically produces a map of memory allocated to a loaded program. The user may suppress the memory map by specifying NM on a RUN card. Real-Time SCOPE obtains information for the map from the loader symbol table.

Heading	Category
SUBP	Name of subprogram from IDC card and absolute address of first location of subprogram
ENTR	Entry point symbol from EPT card and absolute address of each entry point declared in any subprogram loaded for the run
COMM	Absolute addresses of first and last locations in common area
DATA	Absolute address of first location in the data area
EXTA	Absolute address of first location in program extension area
MEMORY	Defines lower batch core limit
MEMORYE	Defines upper batch core limit

Example:

```
SUBP
    54676 Q8QERROR     70134 DUMMY

ENTR
    54702 Q8QERROR     70136 DUMMY     75410 START

COMM
    03517 06555

DATA
    76474

EXTA
    NONE
```

In the SUBP and ENTR maps, the absolute address precedes the subprogram name. The word NONE is printed when no information accompanies a heading. Each MAP begins a new page of print.

11.2 SYSTEM DUMP ROUTINE

The system dump routine produces an octal printout of the console register contents and, optionally:

An octal dump of the contents of the register file

Selected portions of core memory in octal, BCD, or decimal floating-point formats

The dump routine may be called in three ways:

RTS control statement SNAP produces a return jump to SNAPSHOT, a dump routine entry point

COMPASS calling sequence to PROGDUMP, a dump routine entry point

FORTRAN call to FORTDUMP, a dump routine entry point

The dump routine produces the following first line on OUT.

ident LOC(p) A(a) Q(q) B1(b_1) B2(b_2) B3(b_3) IMR(imr)

ident 0-4 BCD characters identifying the dump

( ) Octal contents of the designated registers

If the user selects the register file option, the routine prints the subheading REGISTER FILE, followed by an octal printout of the contents of the 64 registers.

11.3 PROGDUMP AND FORTDUMP

COMPASS and FORTRAN programs can call the system dump routine to print the contents of variables in octal, character, or decimal floating-point modes.

PROGDUMP and FORTDUMP produce an octal printout of console register contents and, optionally, the register file.

The selected core memory dump consists of eight-word data lines printed in the designated mode and preceded by the absolute octal address of the first word of the line. When the dump routine detects a line in which all words are identical to the last word of the preceding line, the line is suppressed and the word GAP is substituted.

11.3.1 COMPASS PROGDUMP

Calling sequence to obtain a dump with COMPASS:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
p	RTJ	PROGDUMP	
p+1		b	
p+2	m	e	
p+3	BCD	1,id	
p+4	*		

b Beginning address of the dumped region

m Mode in which the dump of selected core memory will occur:

Octal Code	Mode
1	Octal
2	BCD character
3	Floating-point
4	Register file
5	Octal; register file
6	BCD character; register file
7	Floating-point; register file

e Ending addres of dumped region

id Dump identifier of 0-4 BCD characters

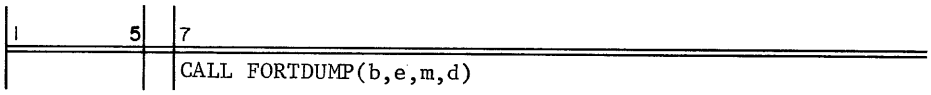
* Control returns at this instruction

Caution:

PROGDUMP enables an interrupt and should not be called during an interrupt subroutine. PROGDUMP must be declared as an external symbol in the subprogram which contains the calling sequence.

11.3.2 FORTRAN FORTDUMP

FORTRAN statement to call the system dump routine:



b Simple or subscripted variable of first word to be dumped

e Simple or subscripted variable of last word to be dumped

m Mode; octal constant with same meaning as COMPASS PROGDUMP, or a variable identifier for the location of the constant

d Hollerith or internal BCD constant or variable identifier giving the location of 4 BCD characters that identify the dump

FORTDUMP need not be declared as an external symbol in the source program.

11.3.3 SAMPLES OF PROGDUMP AND FORTDUMP

To dump the area from BUFY to BUFY+24:

COMPASS Calling Sequence

```
RTJ     PROGDUMP
        BUFY
 1      BUFY+24
BCD     1,PRG1
 .       .
 .       .
 .       .
```

FORTRAN Call

CALL FORTDUMP (BUFY(1),BUFY(25),1,4HPRG1)

The area from BUFY(1) to BUFY(25) is dumped in octal and the dump identified by PRG1.

11.4 RECOVERY DUMP

When a condition causing abnormal termination occurs, Real-Time SCOPE enters the abnormal routine. Unless the user has specified ND on the JOB card (for batch programs) or on the BACK card (for priority programs), ABNORMAL will cause a postexecution dump of console conditions, the register file, and core memory of the terminated program.

The format of the dump is as follows:

LOC(p) A(a) Q(q) B1(b_1) B2(b_2) B3(b_3) IC($p00005_{11-00}$) IA($p00004_{14-00}$)
TC($p00011_{05-00}$) TA($p00010_{14-00}$) IMR(imr)

REGISTER FILE

```
00 (contents)
 .
 .
 .
10 ...
 .
 .
 .
20 ...
 .
 .
 .
70 ...
```

MEMORY

(contents of all relevant user locations except portions of common; 8 words per line)

IC($p00005_{11-00}$)	Interrupt condition in lower 12 bits of 00005. (See Computer System Reference Manual Pub. No. 60157000).
IA($p00005_{14-00}$)	Interrupt address; upon interrupt recognition, address of current unexecuted instruction in lower 15 bits of 00004.
TC($p00011_{05-00}$)	Trapped condition; upper 6 bits of the instruction stored in the lower 6 bits of 00011. If ($p00011_{05-00}$) are 0, termination was due to an abnormal interrupt rather than a trapped instruction.
TA($p00010_{14-00}$)	Trapped address; address of instruction succeeding the trapped instruction, stored in lower 15 bits of 00010. If trapped code is 00, this is the address of ABNORMAL.

APPENDIX SECTION

NOTES ON PRIORITY JOBS A

Characteristics required of priority programs are summarized in this appendix.

- EQUIP statements must be for units using resident drivers only.
- LED cards may not appear in priority binary decks.
- Manual interrupt messages for priority must begin with =.
- Standard system units, such as LIB and INP, can be used only for loading a priority program. The program itself may not reference logical units 50-57 and 60-63.
- No time limit can be imposed on a priority program by Real-Time SCOPE.
- User-driven real-time equipment can not be listed in the AET.
- A user should not attempt to load compilers or assemblers as priority programs or priority programs as batch jobs.
- When the same library entry point is declared external to batch and priority, one copy is loaded for batch and one for priority.
- Real-Time SCOPE prevents direct communication between batch and priority programs; each remains independent.
- Priority programs may not use common.
- Priority and batch programming may not share units, such as MT or CP.
- A priority program can submit batch jobs (appendix B).
- Real-Time equipment should not share a channel with equipment which is not real-time.
- A user must select, process, and clear all interrupts when equipment is user-driven, not system I/O driven.
- Before calling CIO, a real-time program must determine whether it obtained control by interrupting CIO execution. Otherwise it will use CIO on a re-entrant basis (section 5.4).
- If a priority program does not select interrupts it can become dormant: until it is terminated by Real-Time SCOPE or the operator, a new priority program can not be loaded.
- Priority programs cannot select internal interrupt conditions.
- Priority programs must not execute FDP or BCD instructions if hardware features are not present.
- Priority programs should not use block control instructions such as MOVE, SEARCH, etc. or register file locations; their use may cause conflicts with batch processing.

RTS PRIORITY PROGRAM CONTROL STATEMENTS:

On INP:

7/9 SEQUENCE

7/9 BACK

7/9 EQUIP(1)

7/9 TRAIN(2)

7/9 LOAD,p_1,p_2,p_3(3)

binary deck(4)

7/9 RUN

7/9 JOB

.
.
.

continue with batch
program control cards

7/9 SEQUENCE

7/9 BACK

7/9 EQUIP(1)

7/9 TRAIN(2)

7/9 library name

7/9 JOB

.
.
.

continue with batch
program control cards

NOTES: (1) EQUIP cards may occur between the BACK card and the card causing priority program loading. These cards cause a logical unit assignment to be made for priority program use.

(2) TRAIN cards may occur between the BACK card and the card causing priority program loading. They cause a specified train to be mounted on the 512 printer.

(3) If no parameters are present, the priority program is loaded from the INP unit. Otherwise the parameters specify the priority program logical units from which the program is to be loaded. This card may be omitted; then the first IDC card on INP causes loading of the following deck from the INP unit.

(4) A binary deck is found on the units specified by the parameters on the LOAD card. If no parameters exist or if the LOAD card is not present, this deck is on the INP unit.

On CFO:

The operator gains CFO control at SEQUENCE time by setting SELECT JUMP 6. EQUIP and TRAIN statements may occur anywhere between the BACK and CALL statements. These statements cause a logical unit assignment to be made for the priority program.

BACK

EQUIP

TRAIN

CALL,63,library name

CONTROL STATEMENTS THAT MAY NOT APPEAR WITH PRIORITY PROGRAMS:

OCC	REWIND
SNAP	PAUS
UNLOAD	

SYSTEM FLAGS PERTINENT TO PRIORITY PROGRAMS:

BKLDFLG

Definition: Priority Load Flag

Location: SCICRECI (See RTS Installation Handbook)

Description: Lower 15 bits of word defined.
When set $\neq 0$, a BACK card has been encountered and a priority program is being loaded. This flag is referenced throughout EXEC for various tests on the type of program currently being loaded.

Preload initially sets the flag when a BACK card or statement is encountered and clears it at next inter-job cycle or ABNORMAL if termination is executed.

BKRDYFLG

Definition: Priority Ready Flag

Location: SCICRECI

Description: Lower 15 bits of word defined.
When set $\neq 0$, a priority program currently resides in core. It is referenced to determine if certain priority actions are legal. It is set by IRP after the loader has completed loading a priority program and is cleared by RLSBACK and/or ABNORMAL when the current priority program is terminated.

BKRUNFLG

Definition: Priority Run Flag

Location: SCICRECI

Description: Lower 15 bits of word defined.
When set $\neq 0$, the priority program is in execution. This flag must always be set during priority program execution for proper I/O references through CIC, CIO, and MANUAL INTERRUPT.

BLOWMEM

Definition: Priority Lower Memory Limit

Location: SCICRECI

Description: Lower 15 bits of word defined.
Always contains value of lower limit. Value is set by postload after a priority program has been loaded and reset by RLSBACK on termination of priority program.

BNJ.STAT

Definition: Priority Next Job Status

Location: SCICRECI

Description: Holds status code of a stack submitted.

BUPMEM

Definition: Priority Upper Memory Limit

Location: SCICRECI

Description: Lower 15 bits of word defined.
Always contains the value of upper limit; set by installation at system generation time.

PRIORITY-SUBMITTED BATCH JOBS B

The subroutine BNJ. allows the priority program to submit a batch job stack. When calling BNJ., the priority program temporarily reassigns at least one of the standard units INP, OUT, or PUN. These units must be ready for processing when the job is submitted; that is, logical units must be equipped.

The subroutine accepts a parameter string that changes one or all standard unit designations. The changes are substituted between jobs and remain changed until an ENDSCOPE card is encountered on redesignated INP. At ENDSCOPE time the original units are restored and processing continues.

To call BNJ., the user must declare BNJ. an external symbol. The user must not call BNJ. before processing is complete on a preceding priority submitted batch job.

The calling sequence to BNJ. is:

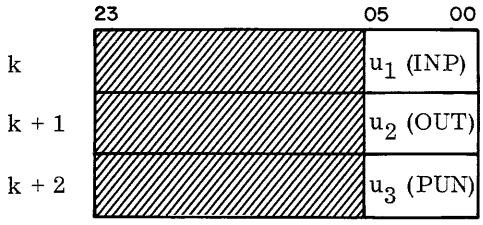
LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
p	ENA	k	
p+1	ENQ	c	
p+2	RTJ	BNJ.	

Parameters include:

k — Address of a three-word list of standard unit designations and other parameters.

c=0 — Normal job processing resumes from interrupted INP.

≠0 — Call returns address; after processing new INP, control returns to priority program at specified return address.



u_i — Word k contains the priority logical unit (u_1) to be assigned as the new INP; word k+1 contains the logical unit to be assigned as OUT; and word k+2 contains the logical unit to be assigned as PUN. If a standard unit is not to be reassigned, the parameter in the corresponding word must be zero. PUN and OUT may be assigned to the same logical unit (u_2=u_3)

If c is 0 in the call, RTS does not return to the priority program after completion of the priority-submitted job stack. RTS resumes processing from the original INP. If there are no batch jobs on the original INP, RTS executes an idle loop, allowing the priority program to run until the priority program submits a new batch stack. Entrance to the priority subroutine specified by parameter c may be interrupted by a priority interrupt.

If the priority program submits one or more batch job stacks in immediate succession on a redesignated INP, RTS is unable to process batch jobs from the original standard input unit without operator intervention. To give priority to the batch jobs on the original INP, the operator presses MANUAL INTERRUPT, types B/P, and presses MANUAL INTERRUPT. RTS completes the current priority-submitted batch job stack and then resumes processing from the original INP. RTS then does not accept additional priority-submitted batch jobs until the operator enters the manual interrupt message P/P. After receiving the message, P/P, RTS again gives processing priority to the priority-submitted batch job stack. Appendix F describes manual interrupt procedures.

LOADER CALLING SEQUENCES C

A batch user can call RDCKF1 to load an absolute record from the library and can call LOADER to load relocatable programs from specified units. When either loader is called, its symbolic name (RDCKF1 or LOADER) must be declared external.

ABSOLUTE

To load an absolute record from the library:

Place the name of the program (CKREC), left justified and with blank fill, into AQ.

Enter B_3 with the number of words to be read; use -0 if the entire record is to be read.

Execute an RTJ RDCKF1.

On return:

(A) = 0 indicates the absolute program was not available.

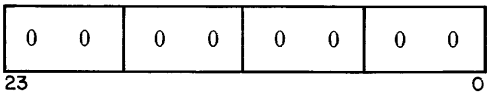
(A) $\neq$ 0 indicates the first word address of the program; entry point address may vary; see description of particular absolute program.

RELOCATABLE

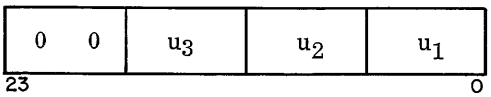
To load a relocatable program, a copy of the loader must have been loaded by RDCKF1. To call the loader to load from specified units until an end-of-file or to the end of a program deck on INP:

Clear the A register

Specify up to three logical units in the Q register as shown below; u_i can be 01-56; 00 designations are ignored. The units must have been equipped (section 2.4).



A register



Q register

Execute an RTJ LOADER instruction.

After an end-of-file is sensed on units u_1, u_2, and u_3, loading continues from INP. To load only from INP, (Q) = 00000000. When loading from INP, reading an end-of-file card causes abnormal job termination. In place of the EOF card, an ELD card (section 8.3.3) can be used. When the end of a program is reached, library routines are called, linkages are established, and control returns to the calling program.

When the loader returns control to the user through the RTJ, the A and Q registers contain the relocated address of the last transfer address and the next-to-the last address, respectively. When more than two transfer point symbols have been read, only the last two are saved and returned to the caller. A loading error is indicated.

$(A)_{0-14}$	Relocated address of last transfer address in the loaded program
$(Q)_{0-14}$	Next-to-the-last address, if there is one
$(Q)_{23-18}$	Number of errors detected during loading

Paper Tape Station

Operation Performed	Permissible Function Code	Calling Sequence	Macro Name	Logical Unit (u)						
				1-57	58 CFO	59 CTO	60 INP	61 OUT	62 PUN	63 LIB
Read 1 record	01	data transfer								
Write 1 record	02	data transfer		yes						
Write EOF	11		WEOF							
	13	status	STATUS	yes						
	14	format	FORMAT							

EDITED STATUS

Paper Tape Station

Bit	Octal Mask	Name	Bit Set	Significance
00	000001	Ready	0 1	Ready
01	000002	Busy	0 1	Busy
02	000004	Not used	0 1	
03	000010	EOF read	0 1	End-of-file read
04	000020	Not used	0 1	
05	000040	Tape supply	0 1	Low
06	000100	Not used	0 1	
07	000200	Not used	0 1	
08	000400	Not used	0 1	
09	001000	End-of-operation	0 1	Operation completed; cleared when new operation initialized
10	002000	Not used	0 1	
11	004000	Binary conversion	0 1	Binary conversion requested
12	010000	Not used	0 1	
13	020000	Station status	0 1	Reader last used
14	040000	Not used	0 1	
15	100000	Channel parity	0 1	Channel parity error
16	200000	Not used	0 1	

Plotter

Operation Performed	Permissible Function Code	Calling Sequence	Macro Name	Logical Unit (u)						
				1-57	58 CFO	59 CTO	60 INP	61 OUT	62 PUN	63 LIB
Plot	02	data transfer	WRITES	yes						
	13	status	STATUS	yes						
	14	format	FORMAT							

EDITED STATUS

Plotter

Bit	Octal Mask	Name	Bit Set	Significance
00	000001	Ready	0 1	Plotter ready
01	000002	Busy	0 1	Plotter busy
02	000004	Not used	0 1	
03	000010	Not used	0 1	
04	000020	Not used	0 1	
05	000040	Manual stop	0 1	Manual stop depressed
06	000100	Not used	0 1	
07	000200	Not used	0 1	
08	000400	Not used	0 1	
09	001000	End-of-operation	0 1	Operation completed; cleared when new operation initialized
10	002000	Not used	0 1	
11	004000	Assembly mode	0 1	Assembly mode
12	010000	Not used	0 1	
13	020000	Not used	0 1	
14	040000	Not used	0 1	
15	100000	Channel parity	0 1	Channel parity error
16	200000	Not used	0 1	

OPTICAL CHARACTER READER

I/O functions for the optical character reader (OCR) differ in certain particulars from the standard CIO requests. The five types of functions for the OCR are:

1. Data transfer:

    - Read from OCR into CONTROL DATA® 3195 Page Reader Controller buffer
    - Read from 3195 buffer into core

2. Mirror control:

    - Read current mirror status (mirror status is the current horizontal position of the mirror)
    - Position mirror and/or advance page

3. Other control functions:

    - Zero mirror
    - Locate line
    - Advance counter 1, 2, or 3
    - Clear counter 1, 2, or 3
    - Stop read
    - Primary or secondary sort
    - Alarm 1 or 2
    - Mark document

4. Status:

    The OCR function is identical to the standard CIO status function and uses the same calling sequence (section 5.4.1)

5. Format:

    The OCR function is identical to the standard CIO format function and uses the same calling sequence (section 5.4.3)

Legal functions, format codes for the format calling sequence, and status bits for the OCR driver are described in the OCR driver table.

OCR Parameters

The following parameters appear in OCR calling sequences and macros:

u — Logical unit number of OCR

jump — RTJ or UJP; in the macros, zero or blank = UJP

raddr — Reject address

i — Interrupt selections:

- 0 — No interrupt
- 1 — Interrupt on abnormal end-of-operation
- 2,3 — Interrupt on end-of-operation, normal or abnormal

Other parameters are defined as they appear.

OCR Data Transfer

To read from OCR into 3195 buffer:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
p	RTJ	CIO	Call CIO
p+1	02	u,i	
p+2	jump	raddr	} Function code and parameters
p+3	m	p_1,0	
p+4		p_2	
p+5	normal return if i = 0	iaddr	
p+6	normal return if i ≠ 0		Continue program

m — Mode change

- 00 — No change
- 40 — Scan 3 character heights, alphanumeric
- 41 — Scan 3, alphabetic
- 42 — Scan 3, numeric
- 43 — Scan 3, mark sense (read zeros and filled zeros only)
- 44 — Scan 2 character heights, alphanumeric
- 45 — Scan 2, alphabetic
- 46 — Scan 2, numeric
- 47 — Scan 2, mark sense (read zeros and filled zeros only)

p_1 Initial mirror coordinate of data block; three octal digits, $0 \leq p_1 \leq 377_8$

p_2 Final mirror coordinate of data block; three octal digits, $p_1 < p_2 \leq 377_8$

The OCR macro for this function is:

```
LOCATION | OPERATION, MODIFIERS  ADDRESS FIELD                  COMMENTS
1        8| 10                   20                            41
         | READOCR              (u,icoor,fcoor,m,s,raddr,jump,i,iaddr)
```

icoor Location containing initial mirror coordinate of the data block

fcoor Location containing final mirror coordinate of the data block

m Mode change

	0	No change
	ANM	Alphanumeric
	APH	Alphabetic
	NUM	Numeric
	MKS	Mark sense
s	2	Scan 2 character heights
	other symbol	Scan 3 character heights

In processing this function, the OCR driver determines the position of the mirror. If the mirror is to the right of the requested initial mirror coordinate, the driver sets the mirror position error bit in the status entry and rejects the request. If the mirror is to the left of the requested initial coordinate, the read operation occurs, starting at the requested initial coordinate. The function code is changed from 02 to 04 to prevent a channel from being declared busy.

To read from 3195 buffer into core:

```
LOCATION | OPERATION, MODIFIERS  ADDRESS FIELD  COMMENTS
1        8| 10                   20             41
p        | RTJ                  CIO            Call CIO
p+1      | 01                   u,i
p+2      | jump                 raddr
p+3      | 00                   fadd,1         Function code
p+4      | 00                   1              and parameters
p+5      | normal               iaddr
         | return
         | if i = 0
p+6      | normal return if i ≠ 0              Continue program
```

fwaddr Address of first word in user's buffer

n Octal digits specifying number of words to read

Character addressing is not allowed since data input is in 12-bit bytes.

The OCR macro for this function is:

```
LOCATION | OPERATION, MODIFIERS | ADDRESS FIELD                | COMMENTS
1      8 | 10                   | 20                          | 41
         | INPUT                | (u,fwaddr,r,raddr,jump,p,i,iaddr)
```

Parameters are as described previously.

OCR Mirror

To read current mirror status:

```
LOCATION | OPERATION, MODIFIERS | ADDRESS FIELD | COMMENTS
1      8 | 10                   | 20           | 41
p        | RTJ                  | CIO          | Call CIO
p+1      | 01                   | u,i          |
p+2      | jump                 | raddr        |
p+3      | 00                   | fwaddr,0     |   Function code
p+4      | 00                   | n            |   and parameters
p+5      | normal               | iaddr        |
         | return               |              |
         | if i = 0             |              |
p+6      | normal return if i ≠ 0              | Continue program
```

fadd Address of location in which mirror status is to be stored

The OCR macro for this function is:

```
LOCATION | OPERATION, MODIFIERS | ADDRESS FIELD          | COMMENTS
1      8 | 10                   | 20                    | 41
         | MIRSTAT              | (u,mstat,raddr,jump)  |
```

mstat Word address of location where mirror status is to be stored

To position mirror and/or advance page:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
p	RTJ	CIO	Call CIO
p+1	02	u,i	
p+2	jump	raddr	Function code and parameters
p+3		p_1,1	
p+4		p_2	
p+5	normal return if i = 0	iaddr	
p+6	normal return if i ≠ 0		Continue program

p_1 Four octal digits of the form Dxxx

D = 2 Forward mirror movement

3 Backward mirror movement

xxx Desired mirror coordinate; $0 \leq xxx \leq 377_8$

p_2 Four octal digits of the form xxx

xxx Number of lines to advance page

The OCR macro for this function is:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	POSITION	(u,nl,mc,raddr,jump,i,iaddr)	

nl Location containing the number of lines to advance; $0 \leq (nl) \leq 177_8$

mc Location containing the desired mirror coordinate; $0 \leq (mc) \leq 377_8$

When positioning the mirror for a read request, the user should place the mirror 10 to 20 coordinates to the left of the initial coordinate of the data block. This position compensates for document drift, hardware tolerances, and acceleration time.

If the direction specified for motion of the mirror is not the direction in which the mirror must move to reach the desired coordinate, the OCR driver rejects the request. The mirror position error bit is set in the status word for this type of reject.

When the request format is legal, the driver changes the function code to 04 to prevent a channel from being declared busy.

Other OCR Control Functions

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
p	RTJ	CIO	Call CIO
p+1	02	u,i	
p+2	jump	raddr	Function code and parameters
p+3		p_1,2	
p+4		0	
p+5	normal return if i = 0	iaddr	
p+6	normal return if i ≠ 0		Continue program

p_1 Hardware control function code

Code	Meaning	
05	Zero mirror	These codes generate an external interrupt
07	Line locate	
47	Stop read	
50	Primary sort	
51	Secondary sort	
57	Mark document	
30	Advance counter 1	These codes do not generate an external interrupt. In the calling sequence, i should be zero and iaddr should be blank. The normal return is to p+5.
31	Clear counter 1	
32	Advance counter 2	
33	Clear counter 2	
34	Advance counter 3	
35	Clear counter 3	
52	Alarm 1	
53	Alarm 2	

The OCR driver rejects a request for an illegal hardware function. For legal requests which generate external interrupt, the driver changes the dummy function code 02 to 04 to prevent a channel from being declared busy. For legal requests for functions which do not generate external interrupt, 02 is changed to 14 (FORMAT) so that neither the channel nor the equipment is declared busy.

OCR macros for these functions are as follows:

Control functions which generate external interrupt:

```
LOCATION  OPERATION, MODIFIERS  ADDRESS FIELD                COMMENTS
1       8 10                    20                           41
          ZMIRR                 (u,raddr,jump,i,iaddr)
          LINELOC               (u,raddr,jump,i,iaddr)
          MARK                  (u,raddr,jump,i,iaddr)
          STPREAD               (u,raddr,jump,i,iaddr)
```

ZMIRR	Zero mirror
LINELOC	Line locate
MARK	Mark document
STPREAD	Stop read

```
LOCATION  OPERATION, MODIFIERS  ADDRESS FIELD                COMMENTS
1       8 10                    20                           41
          SORT                  (u,S,raddr,jump,i,iaddr
```

S	When present, eject page to secondary hopper; when omitted, eject page to primary hopper

Control functions which do not generate external interrupt:

```
LOCATION  OPERATION, MODIFIERS  ADDRESS FIELD                COMMENTS
1       8 10                    20                           41
          ADVANCC               (u,cn,raddr,jump)
          CLEARC                (u,cn,raddr,jump)
```

ADVANCC	Advance counter	
CLEARC	Clear counter	
cn	1, 2, or 3	Advance counter 1, 2, or 3
	Other symbol	Advance counter 1

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	ALARM	(u,a,raddr,jump)	

a	1 or blank	Alarm 1
	2	Alarm 2
	Other symbol	Alarm 1

OCR Status and Format Calls

Calling sequences for status and format requests for the OCR are identical to the sequences for other peripheral equipment (section 5.4).

The OCR macros for these functions are:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	STATUS	(u,d)	

d	1	Dynamic status request
	0	Static status request

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	FRMTOCR	(u,m,s,raddr,jump)	

m	Mode change:	
	0	No change
	ANM	Alphanumeric
	APH	Alphabetic
	NUM	Numeric
	MKS	Mark sense (read only zeros or filled zeros)
s	2	Scan 2 character heights
	other symbol	Scan 3 character heights

OCR Reject and Abort

The normal CIO reject conditions and abort conditions apply in calls for the OCR. The following conditions result in reject by the OCR driver.

- Function code in p+1 is not 01, 02, 13, or 14. The driver returns control to CIO with (A) = 0.
- Mirror position error occurs; mirror position error status bit is set to 1.
- Request contains an illegal hardware function code.

CIO passes the reject to the user program.

OPTICAL CHARACTER READER

Operation Performed	Permissible Function Code	Calling Sequence	Macro Name	Logical Unit (u)						
				1-57	58 CFO	59 CTO	60 INP	61 OUT	62 PUN	63 LIB
Read from OCR to 3195 buffer	02†	data transfer	READ OCR	yes						
Read from 3195 buffer into core	01	data transfer	INPUT	yes						
Read present mirror status	01	mirror control	MIRSTAT	yes						
Position mirror and/or advance Page	02†	mirror control	POSITION	yes						
Select hard-ware function 1	02†	other control	1	yes						
Check unit Status	13	status	STATUS	yes						
Select Format 2	14	format	FORMAT OCR	yes						

B=batch P=priority

† Dummy function code

1. Hardware function codes and macro names:

	Code	Meaning	Macro Name
Generate external interrupt	05	Zero mirror	ZMIRR
	07	Line locate	LINELOC
	47	Stop read	STPREAD
	50	Primary sort	SORT
	51	Secondary sort	SORT
	57	Mark document	MARK
Do not generate external interrupt	30	Advance counter 1	ADVANCC
	31	Clear counter 1	CLEARC
	32	Advance counter 2	ADVANCC
	33	Clear counter 2	CLEARC
	34	Advance counter 3	ADVANCC
	35	Clear counter 3	CLEARC
	52	Alarm 1	ALARM
	53	Alarm 2	ALARM

2. Format codes:

00	Scan 2, alphanumeric
01	Scan 3, alphabetic
02	Scan 3, numeric
03	Scan 3, mark sense
04	Scan 2, alphanumeric
05	Scan 2, alphabetic
06	Scan 2, numeric
07	Scan 2, mark sense

EDITED STATUS

Optical Character Reader

Bit	Octal Mask	Name	Bit Set	Significance
00	000001	Ready	0 1	OCR ready
01	000002	Busy	0 1	OCR busy
02	000004	Mirror Position error	0 1	Mirror position error
03	000010	Buffer full	0 1	Buffer full
04	000020	Mirror far reverse	0 1	Mirror far reverse
05	000040	Mirror far forward	0 1	Mirror far forward
06	000300	Mode	00 01	Alphanumeric Alphabetic
07			10 11	Numeric Mark sense
08	000400	Reject	0 1	Lost data
09	001000	End-of-operation	0 1	Operation completed; cleared when new operation initialized
10	002000	Line Locate error	0 1	Line locator missing
11	004000	Not used	0 1	
12	010000	Compare	0 1	Requested mirror coordinate detected
13	020000	Scan	0 1	Scan 3 Scan 2
14	040000	Not used	0 1	
15	100000	Channel parity	0 1	Channel parity error
16	200000	Not used	0 1	

INSTALLATION ACCOUNTING E

The RTS system:

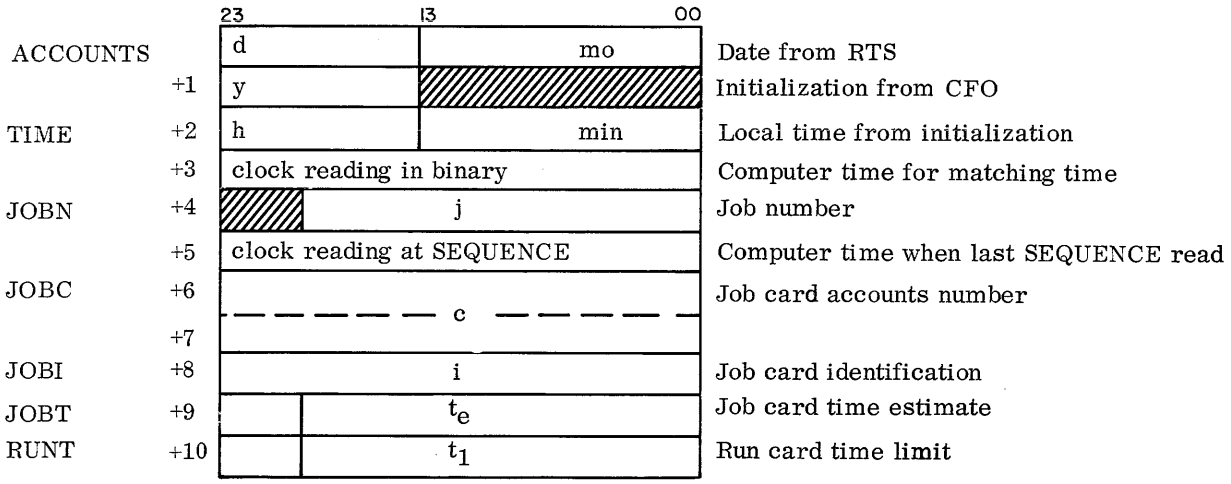
Maintains a table of accounting information

Provides for linkage with the installation-provided accounting routine

Provides for logical unit 57 (ACC) to be used for accounting

The accounts table shown below contains information only for batch jobs.

ACCOUNTS TABLE



Upon detecting each of the control statements SEQUENCE, JOB, ENDSCOPE, ENDREEL, and RUN, the accounting routine does the following:

SEQUENCE: Executes a return jump to the accounting routine to process the accounting record for the preceding batch job. If there is no accounting routine, it continues with the next job.

JOB: Places account number (c), programmer identification (i), and time estimates (t_e) in ACCOUNTS.

ENDSCOPE: Executes a return jump to the accounting routine to process the accounting record for the preceding batch jobs and closes the accounts file prior to unloading ACC.

ENDREEL: Executes a return jump to the accounting routine to record job stack end.

RUN: Places the run time limit or zero in ACCOUNTS.

Each time CIC (chapter 6) prepares to transfer control to the priority program, it executes a return jump to the accounting routine so that it can simulate turning off the clock. Each time the priority program relinquishes control to the batch, CIC executes a return jump to the accounting routine so that it can simulate turning on the clock again.

BATCH JOB TIMING

In addition to maintaining the accounts table, the basic accounting routine may, at the user's option, begin timing when a JOB card in a batch job is encountered (see Installation Handbook). Timing ends when a SEQUENCE or ENDSCOPE card is encountered. Interruption of the batch job by a priority job stops timing so that priority time is not included in the accounting. A message (appendix H) is written on CTO and OUT at the completion of the batch job. Provision is made for addition of a system unit for output from the accounting routine.

MANUAL INTERRUPT PROCEDURES AND OPERATOR STATEMENTS

F

MANUAL INTERRUPT PROCEDURES

A manual interrupt routine in RTS permits the operator to direct messages typed on the CFO to the priority program, the batch program, the error recovery programs, or the operating system. The first character typed indicates which program is to receive the message:

/	batch processing	'	batch error recovery
=	priority processing	!	priority error recovery
Other	operating system		

Before a batch or priority program can receive a message from CFO through a manual interrupt, it must have specified an address to which the manual interrupt routine will pass control. The locations to be initialized are declared as system entry points. They are:

MIFORADD	Batch programs
MIBKADD	Priority programs

Each time a batch or priority message is processed, its corresponding address (MIFORADD or MIBKADD) is cleared.

When RTS detects a batch or priority message, it notes the appropriate manual interrupt address and clears the location. It then places a blank in the first character of the manual interrupt buffer (replaces it with 60_8) before executing a return jump to the manual interrupt address. A system message is not altered before being transferred to its processor.

Program usage of the console typewriter for comments to operator should be through CIO. A message should be in standard format (appendix H) and should terminate with a carriage return. The operator procedure for entering manual interrupt messages is:

1. Press: MANUAL INTERRUPT
2. Type appropriate prefix:

/	Batch processing
=	Priority processing
None	System message
'	Batch error recovery
!	Priority error recovery

3. Type message. If a typing error occurs, press REPEAT and repeat the message correctly.
4. Press: MANUAL INTERRUPT

RTS routes the message to the appropriate program or processes the message if it is a message to the monitor.

In two cases, RTS will not process the message but will respond with a diagnostic (Appendix H). When the message format is incorrect, the operator should start with step 1 of the manual interrupt procedures and repeat the message in correct form. When the message is to a batch or priority program which is not prepared to receive a manual interrupt message, the operator should terminate the job and return it to the programmer for corrective action.

STATEMENTS FROM OPERATOR

In addition, if SELECT JUMP 6 is set before RTS processes a SEQUENCE card, RTS processes the card. In either case, the TYPE LOAD light signals that the system is ready to process operator statements. The operator enters the statement on the CFO and presses FINISH. If there are no detectable errors in the operator statements, RTS processes the statement and waits for the next operator statement. The operator types a period (.) to signal the end of the statements. When the operator presses FINISH after typing the period, RTS resumes processing from INP.

At autoload and initialization, the operator may gain control by typing OPER and pressing FINISH in response to the RTS requested CFO. The RTS Operator's Manual describes operator procedures in greater detail.

Statement	Meaning
AET	Type out entire contents of AET on CTO
AET,a	Type AET entry a on CTO
AET,a,UP	Designate as operable AET entry a
AET,a,DOWN	Designate as inoperable AET entry a
AET,a,RES	Reserve for satellite AET entry a
AET,a,FREE	Free from satellite reservation AET entry a
AET,a,entry	Replace with entry AET entry a
BACK	Identify priority program to RTS
CALL,library name, $p_1,\ldots,p_n$	Load program named from LIB
CALL,u, entry name, $p_1,\ldots,p_n$	Load background program named from LIB or u
EQUIP,u_1=hhCcEeUuu,...	Equip logical units by designating specific physical unit, hardware type, or another previously equipped logical unit (u=1-63)
EQUIP,u_1=hh_1,u_2=hh_2,...	
EQUIP,u_1=u_2,...,u_m=u_n	

ENDSCOPE	End batch processing
SEQUENCE SEQUENCE,j	Alter normal batch job-processing sequence by skipping to logged j, taking j designated by operator, taking next job on INP, or repeating last job
REWIND,u_1,...,u_n	Rewind logical unit u_i
UNLOAD,u_1,...,u_n	Unload logical units u_i

ERROR RECOVERY G

Error recovery routines provide standardized I/O procedures for recovery from errors (appendix H) on magnetic tape, card reader, card punch, and printer. Modular hardware procedures, easily modified to reflect hardware changes and customer needs, implement efficient recovery from detected errors. Batch and priority processing programs may each have a copy of error recovery in core; when a batch or priority program using error recovery is loaded, its copy of error recovery routines is also loaded.

Error recovery notifies the operator of error type, equipment involved, and suggested intervention when recovery is unsuccessful. The operator must:

Press:	MANUAL INTERRUPT
Type:	! (priority) or ' (batch)
Type:	R (retry) or A (abandon) or D (unit down)
Press:	MANUAL INTERRUPT

On retry (R), recovery is again attempted. No further recovery for the particular error is attempted on abandon (A). No recovery is possible when a unit is down (D). In the latter two cases, the user program is informed of the type of error and of the operator's decision; the using program determines the appropriate procedure.

When detected on a reject from the CIO call, all not ready conditions are corrected by recovery routines; they are ignored on completion of data transfer. Errors affecting data transfer are ignored until the operation is completed either normally or abnormally.

Error recovery routines are not re-entrant. If the program includes interrupts, error recovery routines must not be re-entered by the interrupt routines.

REJECT ANALYSIS AND RECOVERY (RAAR)

Reject recovery routines provide recovery for all not ready conditions on magnetic tape units, card readers, card punches, and printers. RAAR, the highest level routine, analyzes a reject to CIO and reissues the call whenever possible. If the equipment is not ready, RAAR can call lower level routines NRC or NWR for recovery. RAAR processes the not ready condition and types an action

diagnostic (appendix H). RAAR exits to the first word address of the CIO calling sequence to reissue the CIO call. If the operator confirms that recovery is impossible or if the reject is due to an illegal function, RAAR exits to RAARREJ+1, a routine and entry point provided by the user program.

Sample calling sequence for RAAR:

```
LOCATION | OPERATION, MODIFIERS | ADDRESS FIELD | COMMENTS
       8 |                   10 |           20 | 41
---------+----------------------+--------------+-----------------
p        | RTJ                  | CIO          | Call CIO
p+1      | f                    | u,i          |
p+2      | RTJ                  | RAAR         |
p+3      | m,C                  | fwaddr       |  Parameters
p+4      | c                    | n            |
p+5      | normal               | iaddr        |
         | return               |              |
         | if i = 0             |              |
p+6      | normal               |              |
         | return               |              | Continue program
         | if i ≠ 0             |              |
```

Parameters include:

f	Input/output function code (see function for each driver, section 5.4)
u	Logical unit number
i	Interrupt selection
	0 No interrupt
	1 Interrupt only on abnormal end-of-operation
	2,3 Interrupt at end-of-operation, normal or abnormal
RAAR	Analyzes a reject; control returns at p+2 in calling sequence
m	Mode selection (see mode for each driver, section 5.4)
fwaddr	Address of first word or character in buffer for data transmission
n	Number of buffered words or characters to be transmitted
iaddr	When interrupt occurs, control transfers to interrupt address specified
,C	Required after mode selection when character addressing is selected
c	Character addressing flag
	40_8 Character addressing selected
	0 or blank Word addressing selected

Input parameters:

The user must provide entry points for RAARREJ and PROGNAME.

Example:

```
LOCATION    OPERATION, MODIFIERS  ADDRESS FIELD      COMMENTS
1        8  10                    20                 41
           ENTRY                 PROGNAME           Provides program entry
PROGNAME   BCD                   2,routine name
    :        :                        :
RAARREJ    ENTRY                 RAARREJ            Provides reject routine
p+1        NOP
           reject
           processing
    :        :
```

Parameters to RAARREJ supplied by RAAR:

(A) = 0 Recovery not possible (decided by operator)
≠ 0 Contains function code

(Q) Word 2 of UST, as passed by reject from CIO

B registers, as passed by reject from CIO

RAARREJ contains first word address of CIO call.

Not Ready Condition (NRC)

RAAR or the user program calls NRC to inform the operator that a peripheral equipment is not ready. If the operator responds with R, NRC transfers control to the CIO calling sequence to reissue the call. If the operator replies with D or A, NRC returns to p+3 of the calling sequence for irrecoverable return.

No Write Ring (NWR)

RAAR or the user program calls NWR to inform the operator that the write ring is missing. If the operator corrects the condition and replies with R, NWR transfers control to the CIO calling sequence to reissue the call. If the operator replies with either A or D, NWR returns to p+3 of the calling sequence for irrecoverable return.

Calling sequence for NRC and NWR:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	ENA	fwaddr	Call CIO
p	RTJ	routine name	Call NRC or NWR
p+1	0	fwa	
p+2	RTJ	reject return	
p+3	irrecoverable		
	return		

Macro for NRC and NWR:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	LOWREJ	(rn,fwaddr,raddr,fwa)	

Parameters include:

rn	Routine name (NRC or NWR)
fwaddr	First word address of CIO call
raddr	Reject return address
fwa	First word address of 2 word program name

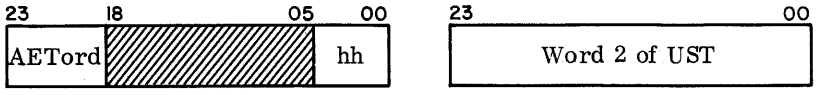
Input Parameters:

(A) First word address of CIO call

Output parameters:

B registers are restored.

Irrecoverable return



A register Q register

23-18	AET ordinal
17-05	Unused
04-00	Hardware type code

Reject return

(A) = 1 Illegal call
2 Illegal LUN

STATUS CHECK AND RECOVERY (SCAR)

Data transfer recovery routines provide recovery for errors within data transfer operations. SCAR, the highest level routine, checks for completion of operation, determines device type, and provides recovery routine. The intermediate level routine applies only to magnetic tape units; it combines all routines for data transfer operations. For the lowest level routines, the programmer determines device type, input/output functions, and error detection before using the routine. The latter two levels are discussed under each device.

SCAR has two entry points (SCAR and SCARNM) for status check and recovery. It checks for completion of a data transfer operation and waits if the operation is not complete. If the status check indicates error recovery is unnecessary, SCAR exits to normal return at p+5. If error recovery is necessary, SCAR determines equipment type and calls the appropriate subroutine. When recovery is accomplished, SCAR returns at p+5.

SCAR changes mode in magnetic tape READ operations before a parity error is declared irrecoverable; SCARNM does not. A pseudo checksumming feature repositions the tape on write errors. When SCAR is used for magnetic tape operations, it must be used for all data transfer. SCAR cannot be called to check status more than once; if it is, the pseudo check is invalid and recovery from a write error is impossible. To insure WRITE operation tape recovery with the checksum feature, the user must:

1. Always use SCAR or SCARNM.
2. Never call SCAR or SCARNM more than once for a single operation.

Calling sequence for SCAR:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	ENA	fwaddr	Call CIO
	ENQ	fwap	
p	RTJ	SCAR or SCARNM	Call SCAR or SCARNM
p+1	n	fwa	Parameters
p+2	RTJ	raddr	
p+3	RTJ	edaddr	
p+4	RTJ	iraddr	
p+5	normal or recovered return		

Macro for SCAR:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	SCAR or SCARNM	(fwap,fwaddr,fwa,raddr,edaddr,iraddr,n)	

fwap	First word address of previous punch (for punch only)
fwaddr	First word address of CIO call
fwa	First word address of two-word program name
raddr	Address of reject routine
edaddr	Address of equipment down routine
iraddr	Address of irrecoverable routine
n	Minimum characters allowed in tape record

Input parameters:

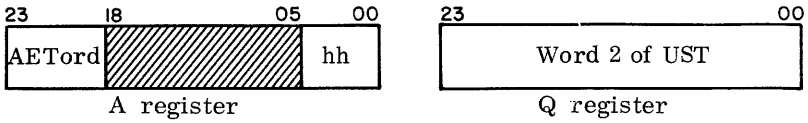
(A) First word address of CIO call (may be a dummy call)

(Q) First word address of previous punch CIO call; used for punch only

Output parameters:

B registers are restored; SCAR entry point SCARUST1 may be used to obtain the appropriate UST words 1 and 2 for all returns.

Irrecoverable return



23-18	AET ordinal
17-05	Unused
04-00	Hardware type code

Reject return

(A) = 1 Illegal call
2 Illegal LUN

Equipment Determination Routine

RAAR, SCAR and user programs can use WHATKIND to determine which intermediate level routine to call. WHATKIND determines equipment type code and the AET ordinal for any given logical unit number.

Calling sequence for WHATKIND:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	LACH	caddr	
p	RTJ	WHATKIND	Call WHATKIND
p+1	return		

Macro for WHATKIND:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	WHATKIND	(caddr)	

caddr Character address of logical unit number

Input parameters:

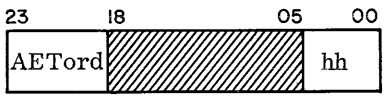
(A) Logical unit number

Output parameters:

Q and B registers are not used.

(A) = 0 Error; logical unit number not 1-63

or



A register

23-18 AET ordinal
17-05 Unused
04-00 Hardware type code

DATA ERROR RECOVERY

All data error recovery routines callable by the user program and/or by SCAR use the same calling sequence. If called by the user program, the operator must determine equipment and error type before the user program calls MTDER/MTDERNM, MTLDACRR, MTRPR, MTWPR, CRDER, CPDER, or PRCPR.

Calling sequence:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	ENA	fwaddr	Call CIO
	ENQ	fwap	
p	RTJ	subroutine name	Call subroutine
p+1	n	fwa	Parameters
p+2	RTJ	raddr	
p+3	RTJ	edaddr	
p+4	RTJ	iraddr	
p+5	recovered		

Macro:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	LOWDATE	(rn,fwaddr,fwap,fwa,raddr,edaddr,iraddr,n)	

Parameters include:

rn	Subroutine name (PRCPR, CRDER, CPDER, MTDER, MTLDACPR, MTWPR, MTRPR)
fwaddr	First word address of CIO call
fwap	First word address of previous punch call (for punch only)
fwa	First word address of two-word program name
raddr	Reject address
edaddr	Downed equipment address
iraddr	Irrecoverable address
n	Minimum characters allowed in tape record

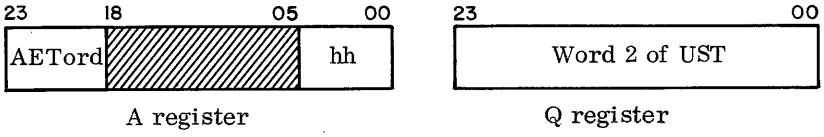
Input parameters:

(A)	First word address of CIO call
(Q)	First word address of previous CIO punch call (for punch only)

Output parameters:

B registers are unused or restored.

Returns p+3, p+4, p+5



23-18	AET ordinal
17-05	Unused
04-00	Hardware type code

Reject return (p+2)

(A) = 1 Illegal call
2 Illegal LUN

Routines for Magnetic Tape

MTDER and MTDERNM — Magnetic Tape Data Error Recovery/No Mode Change: SCAR or the user program can call this routine to provide recovery and operator communication for channel parity errors, for lost data errors, and for parity errors in READ, READB, and WRITE operations. MTDER changes mode in READ parity recovery before declaring irrecoverable; MTDERNM does not. The routine can call MTLDCPRR, MTRPRR, and MTWPRR.

MTLDACPR — Magnetic Tape Lost Data and Channel Parity Recovery: The user program can call this to provide recovery and operator communication for lost data errors and for channel parity errors in READ, READB, and WRITE operations.

MTRPR or MTRPRNM — Magnetic Tape Read Parity Recovery/No Mode Change: The user program can call this to provide recovery and operator communication for parity errors in READ and READB operations. When MTRPR is the entry point, the routine will change the mode if necessary for recovery; MTRDRNM does not change mode.

MTWPR — Magnetic Tape Write Parity Recovery: The user program can call MTWPR to provide recovery and operator communication for parity errors in a WRITE operation.

NRD — Noise Record Detection: SCAR or the user program can call NRD to detect noise records in READ and READB operations. It uses the lowest allowable record length parameter in Q, the address of the CIO call, and the buffer control register character address (bcr) from the UST to determine if the record just read was noise. It uses its own calling sequence.

Calling sequence for NRD:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	ENA	fwaddr	Call CIO
	ENQ	n	
p	RTJ	NRD	Call NRD
p+1	RTJ	raddr	Incorrect call or
p+2	RTJ		noise record
p+3	irrecovera	ble	
	error		

Macro for NRD:

LOCATION	OPERATION, MODIFIERS	ADDRESS FIELD	COMMENTS
1 8	10	20	41
	NRD	(fwaddr,n,raddr,nraddr)	

Parameters include:

fwaddr	First word address of CIO call
raddr	Reject address
nraddr	Noise record address
n	Minimum characters allowed in record

Input parameters:

(A) First word address of data transfer CIO call; may be a dummy call

(Q) Minimum characters allowed in record

Output parameters:

B registers are restored.

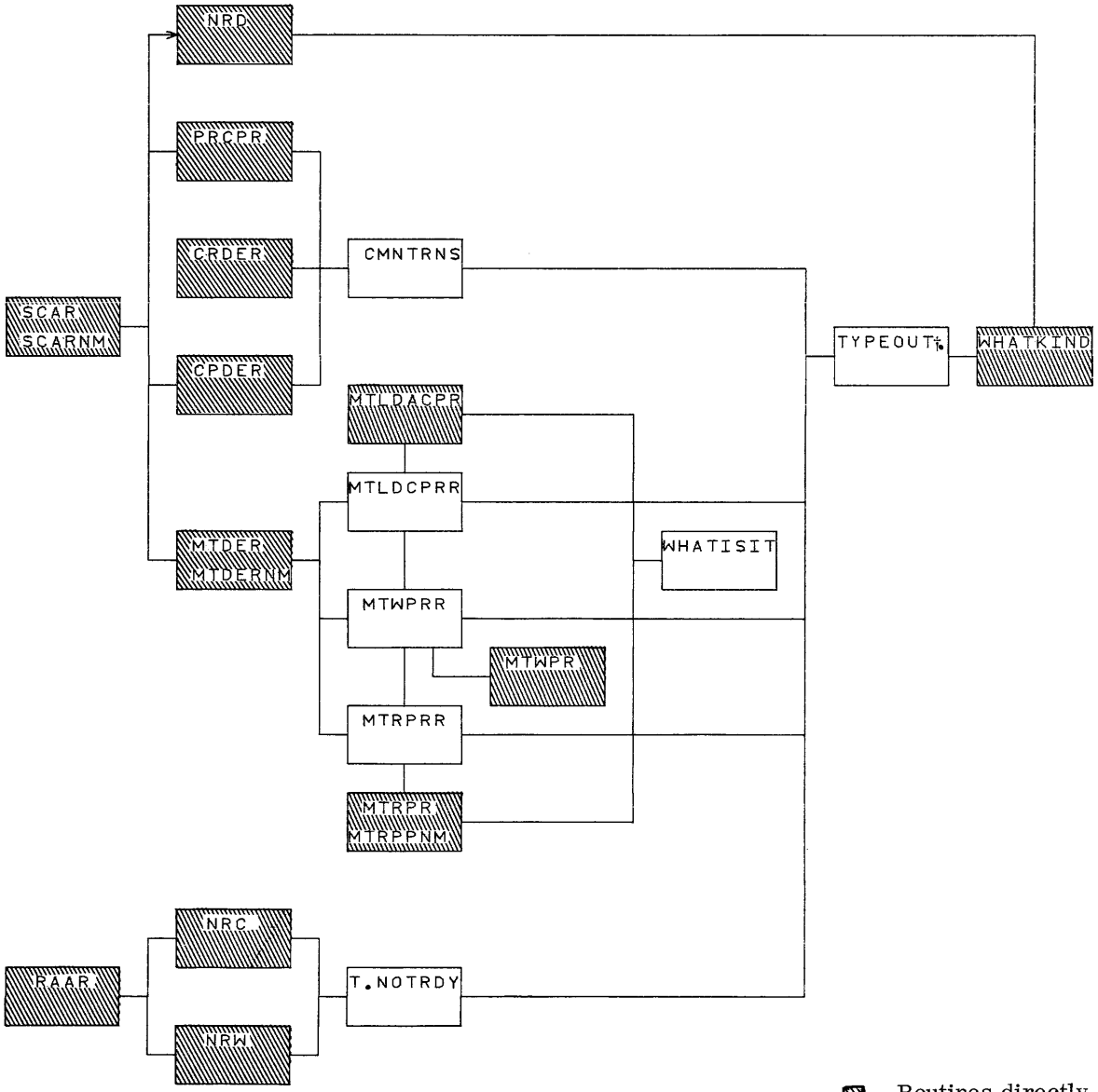
Noise record return and no-noise return

(AQ) UST entry

Reject return

(A) = 1 Illegal call
2 Illegal LUN

Routines are used only in the direction of the arrows, which indicate dependencies (e.g., SCAR calls MTDER but MTDER cannot call SCAR).



† Waits for operator response

▨ Routines directly callable by user program

□ Internal routines

Magnetic Tape Error	Operator Action	Hardware Action
Tape Parity READ	None if recovered; if not recovered, type: R (retry) or A (abandon) or D (unit down)	1. Executes a backspace. 2. Executes a READ; checks for error. 3. Repeats steps 1 and 2 up to three more times if error persists. 4. Backspaces three records to ensure that error record has passed under tape cleaner; checks for load point after each backspace. 5. Executes up to three READs (number determined by presence of load point), nonstop if possible. Transmits record contents only during last read. 6. Checks for READ error on last transmission. 7. Repeats steps 4 through 6 up to four times if error persists. 8. Declares irrecoverable if error persists.
READB		1. Executes a backspace.† 2. Executes a READB; checks for error. 3. Repeats steps 1 and 2 up to three more times if error persists. 4. Backspaces three records to ensure that error record has passed under tape cleaner; checks for end-of-file and end-of-tape after each backspace. 5. Executes up to three READBs (number determined by presence of end-of-file or end-of-tape), nonstop if possible. Transmits record contents only during last read. 6. Checks for READB error on last transmission. 7. Repeats steps 4 through 6 up to four times if error persists. 8. Declares irrecoverable if error persists.

† A backspace in READB moves tape forward.

Magnetic Tape Error	Operator Action	Hardware Action
WRITE	None if recovered; if not recovered, type: R (retry) or A (abandon) or D (unit down)	1. Executes two backspaces. 2. Executes dummy READ forward; checks for error. 3. If no error, goes to step 6. 4. Executes READ recovery. 5. If error persists, repeats steps 1 through 4 up to three more times; declares error irrecoverable. 6. Executes n† skip bad spot commands and rechecks. 7. If error, declares WRITE irrecoverable with bad erase. 8. Executes WRITE; checks for error. 9. If error persists, checks for EOT. 10. If EOT, declares error irrecoverable with EOT status. 11. Increases and tests erase count n †n=10 Declares irrecoverable with maximum skip count status ≠10 Repeats process until recovered or n = 10
Lost Data		1. Executes a backspace. 2. Repeats function. 3. If error persists, repeats steps 1 and 2 up to four times. 4. Declares error irrecoverable if it persists.
Channel parity		1. Executes a backspace. 2. Repeats function. 3. If error persists, repeats steps 1 and 2 up to four times. 4. Declares error irrecoverable if it persists.

† n Up to 10 erases of 6 inches each

Routines for Card Reader

CRDER — Card Reader Data Error Recovery: SCAR or the user program can call this routine to provide recovery and operator communication for PREREAD, COMPARE, and channel parity errors. A slipped card condition causes a diagnostic (appendix H).

Card Reader Error	Operator Intervention	Hardware Action
PREREAD or COMPARE	1. Place last card in output tray in first position of tray.	
	2. Press: READY Press: MANUAL INTERRUPT Type: R Press: MANUAL INTERRUPT	
		3. READ is repeated and checked for error.
	4. If error persists, declare irrecoverable.	
Channel Parity buffered	1. Place last two cards in output tray in first two positions of input tray.	
	2. Press: READY Press: RELOAD MEMORY Press: MANUAL INTERRUPT Type: R Press: MANUAL INTERRUPT	
		3. READ is repeated and checked for error.
	4. If error persists, declare irrecoverable.	
unbuffered	1. Place last card in output tray in first position of input tray.	
	2. Press: READY Press: MANUAL INTERRUPT Type: R Press: MANUAL INTERRUPT	
		3. READ is repeated and checked for error.
	4. If error persists, declare irrecoverable.	

Routines for Card Punch

CPDER — Card punch Data Error Recovery: SCAR or the user program can call this routine to provide recovery and operator communication for COMPARE and channel parity errors.

Card Punch Error	Operator Intervention	Hardware Action
COMPARE		1. Repunches last card; erroneous card automatically goes to output tray.
	2. Remove last 2 cards	
	3. Type: ' (priority) or ! (batch) Type: R (retry)	
		4. Repunches erroneous card.
	5. If error persists; declare irrecoverable.	
Channel Parity	1. Remove last card	
	2. Type: ' (priority) or ! (batch) Type: R (retry)	
		3. Repunches card and checks for error.

Routines for Printer

PRCPR — Printer Channel Parity Recovery: SCAR or the user program can call this routine to provide recovery and operator communications for channel parity errors. NRC detects out-of-paper conditions.

Printer Error	Operator Intervention	Hardware Action
Channel Parity		1. Repeats call and checks for error up to four times.
	2. None if recovered; if not recovered, type: R (retry) or A (abandon) or D (unit down)	

DIAGNOSTICS H

All RTS system and I/O messages are in standard format. They are listed in numerical order. The standard format is:

Type Source Number Message Unit

Type: A Action message; operator action necessary
D Destructive; fatal error occurred
I Informative; verifies system initiation or completion of action or indicates occurrence of a nonfatal error

Source: Four-digit code indicating task initiating message

Number: Three-digit number identifying message

Message: Description of error type, action initiated or completed, or operator action required

Unit: Identification of output unit for diagnostic

In response to I/O messages, the operator must type:

R Retry
or
A Abandon
or
D Unit down

! Priority (if program identification is required)
or
' Batch (if program identification is required)

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
I	SYS	008	(ILL MI MSG)	CTO	A MANUAL INTERRUPT message is unrecognizable. Either a prefix (=, /, !, ≠) is missing, the message is misspelled, or the target program is not set up to receive the message.	1. Press MANUAL INTERRUPT 2. Type correct message 3. Press MANUAL INTERRUPT If the correct message is rejected, terminate the program and return it to the programmer for corrections.
A	SYS	011	error ERR. ON LIB. LD PE XP	CTO	Error when reading absolute records from LIB. Ten attempts were made. Lost data Tape parity error Transmission parity error	1. Press MANUAL INTERRUPT 2. Type TRY to retry recovery GO to search for another copy of absolute program 3. Press MANUAL INTERRUPT Otherwise: 1. Mount backup copy of library 2. Press MANUAL INTERRUPT 3. Type NEW 4. Press MANUAL INTERRUPT

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
D	SYS	013	BATCH OV	CTO	Loading of the current system or batch absolute program would destroy the priority program. The system cycles until the current priority program is released or until the operator terminates the batch program.	
A	SYS	014	LUN NOT RDY unit	CTO	Specified logical unit is neither ready nor busy.	1. Ready specified logical unit 2. Press MANUAL INTERRUPT 3. Type GO 4. Press MANUAL INTERRUPT
D	SYS	015	NO RHT DEF unit	CTO OUT	Specified logical unit has an illegal zero RHT entry. Either the resident tables are out of order or the system was destroyed.	Autoload; resubmit job
D	SYS	016	NO DRIV ADD unit	CTO OUT	Specified logical unit has a zero driver address entry in the AET. Either the resident tables are out of order or the system was destroyed.	Autoload; resubmit job
D	SYS	017	NON-RES DRIV unit	CTO OUT	Specified system unit has a nonresident driver. Either the AET entry is incorrect or the system was destroyed.	Autoload; resubmit job

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
A	SYS	019	REPEAT	CTO	The last statement from CFO was not recognizable to the monitor.	Repeat statement in correct form.
I	SYS	020	CALL FINISHED	CTO	The last CALL statement from CFO was accepted; the specified program was successfully loaded.	Operator has control to continue.
D	SYS	021	CALL ABORTED	CTO	The last CALL statement from CFO was accepted; the specified program was not successfully executed.	Operator has control and may retry the call.
A	SYS	022	RELEASE PRIORITY?	CTO	A request was made to load a new priority program but one already resides in core.	To release current priority program: 1. Type YES 2. Press FINISH To continue the current program and bypass the new one: Press FINISH
D	SYS	023	CANNOT DRIVE unit	CTO	Error condition exists on specified logical unit. If error occurs in a batch program, monitor loops to allow priority execution.	Attempt recovery. 1. Press MANUAL INTERRUPT 2. Type GO 3. Press MANUAL INTERRUPT If this fails, resubmit job.
I	SYS	024	PRIORITY X ONLY	CTO OUT	Destructive batch job errors were encountered.	Batch processing is suspended; priority processing continues. Autoload after priority execution is complete.
D	SYS	025	AET ERR RE-AUTOLOAD	CTO	An AET entry is missing, either the tables were destroyed or there is a machine error.	Autoload

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
A	SYS	026	ENABLE WRITE unit FINISH GO	CTO	The specified logical unit needs a write ring before processing can continue.	1. Place write ring on specified unit. 2. Press FINISH if FINISH appears in message. Otherwise: 3. Press MANUAL INTERRUPT 4. Type GO 5. Press MANUAL INTERRUPT
I	SYS	027	INP ERR	CTO	Read error on INP	Processing continues. The card image is accepted as read. Retry job.
A	SYS	028	LOAD NEW unit FINISH GO	CTO		1. Mount new tape reel on specified logical unit 2. Press FINISH if FINISH appears in message. Otherwise: 3. Press MANUAL INTERRUPT 4. Type GO 5. Press MANUAL INTERRUPT
D	SYS	029	MCC ERR	CTO OUT	A monitor control card is in illegal format or is unrecognizable to the system.	Current job terminates. Correct the illegal card; resubmit job.
D	SYS	030	SEQ ERR	CTO OUT	A SEQUENCE, ENDSCOPE, EOF or ENDREEL is missing or misplaced in the job stack.	Job terminated. 1. Correct error 2. Resubmit job
D	SYS	031	UNCNVRT	CTO OUT	An illegal combination of bits was encountered in Hollerith to BCD conversion.	Job terminates.
D	SYS	032	EQA ERR unit	CTO OUT	Equipment assignment error on specified logical unit. If unit is blank, the logical unit is out of range.	Job terminates. 1. Correct error 2. Resubmit job
D	SYS	033	EOT unit	CTO OUT	Illegal end-of-tape condition on specified logical unit.	Job terminates. 1. Correct error 2. Resubmit job

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
D	SYS	034	LOADING DELETED	CTO OUT	Errors encountered in generating load-and-go file.	Job terminates abnormally.
D	SYS	036	RUN ABORTED	CTO OUT	This diagnostic appears when the system detects errors which prohibit execution of a program.	Job terminates
D	SYS	037	LDR ERRS no.	CTO OUT	Specified number of loader errors were found.	Execution is inhibited.
D	SYS	038	PCC	CTO OUT	Priority control card error.	Job terminates
D	SYS	039	AT B/P/S	CTO OUT	Abnormal termination by a program through a jump to ABNORMAL.	Job terminates
D	SYS	040	B/P (xt) execution address	CTO	CIO detected on error identified by x and t codes.	
			xt			
			IC		Illegal call to CIO	
			LU		Logical unit	
			UU		Undefined unit	
			x			
			C		Attempt to connect unit	
			D		Data transfer operation on unit	
			P		Request for operation on protective unit	
			S		Select operation on unit	
			t			
			1		Illegal request on system file	
			2		Illegal request on PUN	
			3		Illegal request on OUT	
			4		Illegal request on INP	
			5		Illegal request on CFO/CTO	
			6		End-of-file error on INP	
			E		External reject; equipment reserved or busy	
			P		Transmission parity error; probably hardware malfunction	
			Q		Unknown reject equipment or unit possibly misdialed	

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
I	SYS	041	UI $^{B}_{P}$ no.	CTO	An interrupt occurred. The corresponding central interrupt table does not contain a user procedure address.	Control returns to the interrupted subprogram.
			0ℓc		External interrupt ℓ line 0-7 c channel designator 0-7	
			10c		I/O channel interrupt c channel designator 0-7	
			110		Real-time clock interrupt	
			111		Arithmetic overflow fault	
			112		Divide fault	
			113		Exponent overflow fault	
			114		BCD fault	
			115		Search/move interrupt	
			116		Manual interrupt	
			117		Associated processor interrupt	
D	SYS	042	$^{B}_{P}$ INT STACK OV	CTO	Batch or priority interrupt stack table overflowed.	Job terminates
I	SYS	043	PRIORITY OFF	CTO	Current priority program was terminated either by the operator or by priority program.	Job terminates
A	SYS	044	$^{Q}_{E}$R(C no. E no. U no.)		One of the following rejects occurred during I/O processing:	Type R to retry or ready A to abandon, results in halt; if GO is pressed RTS assumes condition is corrected and retries.
			Q		No response reject on channel	
			E		External reject active on channel	
A	SYS	045	LI C no. E no. U no.	CTO	An equipment interrupt was lost.	To simulate interrupt and continue Press FINISH

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
I	SYS	046	UI(C no. E no.)	CTO	An unexpected interrupt occurred on specified channel and equipment.	Informative message
I	SYS	047	PE(C no.)	CTO	Transmission parity occurred on specified channel. If message appears once, that transmission error was detected at time of equipment interrupt. If message appears twice, that transmission error was detected at both channel and equipment interrupt time.	
I	SYS	048	(OPER. CONTROL)	CTO		Operator has control to type in control statements.
D	SYS	049	IQ RE-ENTERED	CTO	The user returned to the central interrupt control routine because he had been processing an interrupt; no interrupt exists.	Job aborts
A	SYS	060	MOUNT TRAIN id ON LU no.	CTO	id External identifier which matches the table entry in PRELOAD and POSTLOAD. It is called out on the control card or on the operator statement TRAIN.	1. Mount specified 512 train on specified LU no. 2. Press MANUAL INTERRUPT 3. Type GO Type NO if specified TRAIN is unavailable 4. Press MANUAL INTERRUPT
I	SYS	061	TRAIN id MOUNTED	OUT CTO	This message appears if the operator typed GO in response to message SYS 060. The 512 printer train id was mounted. id External identifier	
D	SYS	062	TRAIN id UNAVAILABLE	OUT CTO	This message appears when the operator typed NO to message SYS 060 signifying that 512 printer train id cannot be mounted by the operator. id External identifier	System continues with next job.

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
D	SYS	063	TRAIN id NOT IN TABLE	CTO OUT	Id on the control card or on the operator statement TRAIN does not match any entries in the PRELOAD/POSTLOAD tables. id External identifier	System continues with the next job.
I	SYS	100	LOADER I/O REJECT UNIT no.	CTO OUT	Loader encountered an irrecoverable I/O reject. The standard error recovery package was unable to correct the problem.	
A	SYS	101	READY LUN no. GO	CTO	The specified unit is not busy and not ready.	1. Ready unit 2. Press MANUAL INTERRUPT 3. Type GO 4. Press MANUAL INTERRUPT
A	SYS	102	RELOAD LAST CARD. GO	CTO	A PREREAD/COMPARE error or channel parity error occurred.	If card reader is ready, a channel parity error occurred. On buffered card reader: 1. Move last 2 cards 2. Press RELOAD memory On unbuffered card reader, move last card If card reader is not ready a PREREAD/COMPARE error occurred. 1. Move 1 card 2. Ready unit
D	SYS	103	OVPRO ILLEGAL DEVICE UNIT no.	CTO OUT	The unit designated is not magnetic tape.	Job terminates
I	SYS	0145	subr PC	CTO OUT	Common was declared for priority program.	Loading terminates
I	SYS	0146	subr CF wdct hollerith field subr CF wdct subp subr CF subp	CTO OUT	Card format error. May indicate illegal character, illegal separation character, or unconvertible Hollerith bit pattern. Particularly suspect are TRA, EXS, and LED cards. May also indicate incorrect or illegal punches in subfield or overlay control cards such as MAIN, OVERLAY, SEGMENT.	

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE		RESULT/ACTION
I	SYS	0147	subr CK wdct addr	CTO OUT	Checksum error. If the word count (wdct)=44, error may be subprogram checksum error.		
I	SYS	0150	subr CS wdct hollerith field	CTO OUT			
			subr CS wdct addr	CTO OUT	Card sequence error.		
					Card type	Error	
			00		undetermined	Nonloader card read from unit other than INP	
			01-40		RIF	No IDC card is in front of loader cards which follow TRA	
			41		IDC	Card does not follow a TRA card and is not first IDC card	
			42		EPT	No IDC card in front of loader cards which follow TRA	
			43		XNL	Card out of sequence	
			44		TRA	Zero length subprogram	
			45		LRL	Card out of sequence	
			46-47		unrecognized		
			50		MAIN	1. More than one MAIN card 2. MAIN not first OVERLAY control card	
					(continued)		

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE		RESULT/ACTION
			(continued)				
					Card type	Error	
			51		OVERLAY	1. OVERLAY control card within subprogram deck 2. MAIN not first OVERLAY control card	
			52		SEGMENT	1. Contiguous OVERLAY control cards (no subprogram decks intervened) 2. MAIN not first OVERLAY control card 3. SEGMENT card not preceded by OVERLAY card; SEGMENT card in error is treated as if it were OVERLAY; execution is inhibited	
			56-76		unrecognized		
			77		ELD	ELD card read on unit other than INP	
I	SYS	0151	subr DB wdct addr	CTO OUT	Data block exceeds previously defined length. wdct parameters are listed in I SYS 0150.		
I	SYS	0152	subr DS subp	CTO OUT	An entry point symbol was encountered more than once.		
I	SYS	0153	subr EP	CTO OUT	LED card encountered in priority program.		Loading terminates

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
I	SYS	0154	subr EOF unit	CTO OUT	End-of-file read on INP.	Loading terminates
I	SYS	0155	subr HR unit	CTO OUT	Hardware reject from mass storage unit.	
I	SYS	0156	subr I/O unit	CTO OUT	Irrecoverable error in I/O operation	If it occurred while writing overlays, loading terminates.
I	SYS	0157	subr MS subp	CTO OUT	Missing subprogram. Entry point name to correspond with library name statement is not in DRS. Occurs only on load library subprogram call.	
I	SYS	0160	subr OV subp	CTO OUT	Loader symbol table overflowed into program area.	Loading terminates
I	SYS	0161	subr RL wdct addr	CTO OUT	One of the following caused this relocation factor error: 1. Load address relocation byte is not 0010(2) or 0100(2) 2. Load address relocation byte is 0100(2) and the data area is undefined 3. Data area is part of OVERLAY element already on tape 4. Relocation byte is zero	
I	SYS	0162	subr SL subp	CTO OUT	String of addresses for external symbol caused a loop.	
I	SYS	0163	subr TR no.	CTO OUT	Either no TRA card contains a transfer symbol or more than two TRA cards contain transfer symbols.	
I	SYS	0164	subr UD subp	CTO OUT	Declared external name is not a defined entry point in loaded subprograms, or file two of LIB, or in permanent portion of loader symbol table.	
I	SYS	0166	subr LX subp	CTO OUT	Symbols on an EXS card are equated to define a loop.	

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
I	SYS	0174	subr OV wdct addr	CTO OUT	Storage required to load subprogram exceeds available memory. wdct parameters are listed with message I SYS 0150.	Loading terminates
I	SYS	0175	subr OV	CTO OUT	There is insufficient memory for the temporary storage which is required by the loader to load from the library.	
I	SYS	200	UNIT no. IRRECOVERABLE I/O ERROR	CTO OUT	Irrecoverable I/O error on specified unit.	
A	SYS	202	NO SEQUENCE J	CTO	Operator requested system to search for SEQUENCE no. J, but the system cannot locate the card.	Operator has control.
D	SYS	203	UNIT no. DOWNED	OUT	Specified unit downed because of irrecoverable error.	
I	SYS	204	RESUBMIT LAST JOB	CTO	This message appears after message number 200.	Retry last job.
A	SYS	205	REASSIGN INP OUT IF DESIRED PUN	CTO	Irrecoverable error on INP OUT PUN	To reassign INP OUT PUN 1. Type AET ordinal when INP OUT appears on CTO PUN 2. Press FINISH If no reassignment desired, press FINISH
D	SYS	211	RDUMP IRRECOVERABLE ERROR ON UNIT 61	CTO	Standard error recovery could not recover from an error on unit 61.	Job terminates
D	SYS	212	RDUMP ILLEGAL DEVICE FOR 61	CTO	A device was illegally assigned to 61 for RDUMP.	Job terminates
D	SYS	213	RDUMP CANNOT DUMP	CTO	Irrecoverable conditions on unit 61 inhibit output.	Job terminates

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
A	SYS	214	READY TAPE unit	CTO	The dialed magnetic tape unit is not ready.	1. Press MANUAL INTERRUPT 2. Type GO if correctable NO if not correctable 3. Press MANUAL INTERRUPT
I	SYS	300	X length	OUT	An OCC card incorrectly define an extension area. length 5-digit octal length of the defined extension area.	
I	SYS	301	PN COL. no.	OUT	Subprogram name on either OCC or on SNAP card is unidentified.	
I	SYS	302	BS COL. no.	OUT	DATA or COMMON block referenced on SNAP card is unidentified.	
I	SYS	303	AD COL. no.	OUT	Address or location field on SNAP card begins with illegal character.	
I	SYS	304	8F COL. no.	OUT	Octal field contains a non-octal character.	
I	SYS	305	XA COL. no.	OUT	Program extension area undefined or too small.	
I	SYS	306	WR COL. no.	OUT	Wraparound. Address exceeds upper limits of available memory.	
I	SYS	307	IM	OUT	Illegal mode field on SNAP card.	
I	SYS	308	AN COL. no.	OUT	Antecedent error on OCC card; * relocation factor and no subprograms previously defined; or + load address before loading area defined.	
I	SYS	309	RL COL. no.	OUT	Relocation portion of a correction contains an illegal character.	

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
I	SYS	310	RG	OUT	Beginning address of area to be snapped is larger than ending address.	
I	SYS	311	OV	OUT	Overflow of memory will occur if SNAPSHOT is loaded.	
D	SYS	321	***NOS	CTO OUT	A SNAP statement was encountered, but the SNAPSHOT routine is not in memory; probably haven't declared SNAPSHOT as external.	Run terminates
D	SYS	323	POSTLOAD I/O REJECT UNIT no.	CTO OUT	POSTLOAD encountered an irrecoverable I/O reject. Either the standard error recovery package cannot recover or a system error occurred which resulted in an incorrect call to the standard error recovery routines.	Job terminates
I	SYS	400	JOB job ELAPSED TIME hrs HRS mins MINS secs SECS	CTO OUT	Accounting information message output after each batch job.	

I/O MESSAGES

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
A	I/O	001	B/P LUN no. c e u REPEAT	CTO	Reply to message pertaining to specified logical unit c e u not R, A, or D.	Repeat reply
A	I/O	002	B/P LUN no. c e u CRFF progname	CTO	Feed failure occurred on card reader assigned to specified logical unit c e u.	Do not press RELOAD MEMORY on buffered card reader. 1. Take corrective action 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT
A	I/O	003	B/P LUN no. c e u CRHE progname	CTO	Hopper empty on card reader assigned to specified logical unit c e u.	1. Take corrective action 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT
A	I/O	004	B/P LUN no. c e u CPFF progname	CTO	Feed failure occurred on card punch assigned to specified logical unit c e u.	1. Take corrective action 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT
A	I/O	005	B/P LUN no. c e u PROP progname	CTO	Printer assigned to specified logical unit c e u out of paper.	1. Take necessary corrective action 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
A	I/O	006	B P LUN no. c e u UNNR progname	CTO	Not ready condition exists on unit assigned to specified logical unit c e u. It is not feed failure, hopper empty, or out of paper.	1. Take necessary corrective action 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT
A	I/O	007	B P LUN no. c e u WRMT progname	CTO	Write ring missing on magnetic tape reel on unit assigned to specified logical unit c e u.	1. Take necessary corrective action 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT
A	I/O	008	B P LUN no. c e u MTWP progname	CTO	Error recovery routine could not recover from WRITE parity error on unit assigned to specified logical unit c e u.	1. If necessary, take corrective action 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT
A	I/O	009	B P LUN no. c e u MTRP progname	CTO	Recovery routine could not recover from READ parity error on magnetic tape unit assigned to specified logical unit c e u.	1. If necessary, take corrective action 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT
A	I/O	010	B P LUN no. c e u MTWL progname	CTO	Recovery routine could not recover from WRITE lost data error on magnetic tape unit assigned to specified logical unit c e u.	1. If necessary, take corrective action 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
A	I/O	011	B/P LUN no. c e u MTRL progname	CTO	Error recovery routine could not recover from READ lost data error on magnetic tape unit assigned to specified logical unit c e u.	1. If necessary, take corrective action 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT
A	I/O	012	B/P LUN no. c e u MTWC progname	CTO	Error recovery routine could not recover from channel parity error in output buffer to magnetic tape unit assigned to specified logical unit c e u.	1. If necessary, take corrective action 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT
A	I/O	013	B/P LUN no. c e u MTRC progname	CTO	Error recovery routine could not recover from channel parity error in input buffer from magnetic tape unit assigned to specified logical unit c e u.	1. If necessary, take corrective action 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT
A	I/O	014	B/P LUN no. c e u CRCE progname	CTO	Preread or compare error occurred on card reader assigned to specified logical unit c e u.	1. Move last card from output bin to first card in input bin 2. Ready unit 3. Press MANUAL INTERRUPT 4. Type ' for batch ! for priority 5. Type R, A, or D 6. Press MANUAL INTERRUPT

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
A	I/O	015	B/P LUN no. c e u CRRC progname	CTO	Error recovery routine could not recover from channel parity error in input buffer from card reader assigned to specified logical unit c e u.	1. Buffered card reader: Move last two cards from output bin to first two cards in input bin. Unbuffered card reader: Move last card from output bin to first card in input bin. 2. Press RELOAD MEMORY if buffered 3. Ready unit 4. Press MANUAL INTERRUPT 5. Type ' for batch ! for priority 6. Type R, A, or D 7. Press MANUAL INTERRUPT
A	I/O	016	B/P LUN no. c e u PRWC progname	CTO	Error recovery routine could not recover from channel parity error in output buffer to printer assigned to specified logical unit c e u.	1. Take necessary corrective action 2. To reprint line: a. Go to printer b. Press STOP c. Press START 3. Press MANUAL INTERRUPT 4. Type ' for batch ! for priority 5. Type R, A, or D 6. Press MANUAL INTERRUPT
A	I/O	017	B/P LUN no. c e u CPCE progname	CTO	Compare error occurred on card punch assigned to specified logical unit c e u.	1. Remove and dispose of last two cards in output bin. 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
A	I/O	018	$\frac{B}{P}$ LUN no. c e u CPWC progname	CTO	Channel parity error occurred on card punch assigned to specified logical unit c e u.	1. Remove and dispose of last card in output bin 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT
A	I/O	019	$\frac{B}{P}$ LUN no. c e u CRSC progname	CTO	A card (or cards) passed the read station without being read; there is no way to determine the number of cards not read.	1. Take necessary corrective action 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT
A	I/O	020	$\frac{B}{P}$ LUN no. c e u MTEP progname	CTO	A parity error was detected on an ERASE magnetic tape operation. Recommended reply is D since no action can guarantee recovery; reply R ignores parity error on ERASE; reply A abandons recovery but logical recovery by the using program may occur. When job is complete repair or replace tape.	1. Press MANUAL INTERRUPT 2. Type ' for batch ! for priority 3. Type R, A, or D (see significance column) 4. Press MANUAL INTERRUPT
A	I/O	021	$\frac{B}{P}$ LUN no. c e u MTCK progname	CTO	Recovery routine is unable to reposition correctly while attempting recovery on magnetic tape WRITE parity error.	1. If necessary, take corrective action 2. Press MANUAL INTERRUPT 3. Type ' for batch ! for priority 4. Type R, A, or D 5. Press MANUAL INTERRUPT

TYPE	SOURCE	NO.	MESSAGE	UNIT	SIGNIFICANCE	RESULT/ACTION
A	I/O	022	B/P LUN no. C no. E no. U no. PRCF progname	CTO	A nonprintable character was encountered and recovery was attempted four times. Normal conditions indicate that the image memory is not loaded for output for this program.	1. Press MANUAL INTERRUPT 2. Type ' for batch ! for priority 3. Type R, A, or D 4. Press MANUAL INTERRUPT
A	I/O	023	B/P LUN no. C no. E no. U no. PRNP progname	CTO	A print character is not on the print chain. Recovery was tried four times. Either the print chain does not match image memory or a hardware error occurred.	1. Press MANUAL INTERRUPT 2. Type ' for batch ! for priority 3. Type R, A, or D 4. Press MANUAL INTERRUPT

INDEX

ABBREVIATIONS AND ACRONYMS

The list of abbreviations does not include COMPASS pseudo instructions, machine language instructions, programmer control cards, operator control statements, or diagnostics.

ACC	Standard accounting unit
AET	Available equipment table
ALGOL	Algorithmic language compiler
BCD	Binary coded decimal
BNJ.	Priority next job routine
BRHT	Priority running hardware table
BSIPP	Background simultaneous peripheral processor
CFO	Comment-from-operator unit; console typewriter
CIC	Central interrupt control routine
CIO	Central input/output routine
CIT	Central interrupt table
COBOL	Common business oriented language
COMPASS	Comprehensive assembly system
COSY	Compressed symbolic library program
CP	Card punch
CPU	Central processing unit
CR	Card reader
CST	Channel status table
CTO	Comment-to-operator unit; console typewriter
DINT.	Disable interrupt routine
EINT.	Enable interrupt routine
ELD	End-loader-declaration card
EOF	End-of-file
EOO	End-of-operation interrupt
EOT	End-of-tape mark
EPT	Loader entry point name card

EST	Equipment status table
fca	First character address
FORTRAN	Formula translation compiler
fwa	First word address
iaddr	Interrupt address
IC	Interrupt condition or code
IDC	Subprogram identification card
IMR	Interrupt mask register
INP	Standard input unit; typically a card reader
I/O	Input/output
lca	Last character address
LED	Loader equipment declaration card
LGO	Load-and-go unit
LIB	Standard library unit
LRL	Local reference list loader card
LUN	Logical unit number
lwa	Last word address
MAP	Memory allocation print
MT	Magnetic tape
ND	No dump
NM	No map
OCR	Optical character reader
OUT	Standard output unit; typically a printer
PERT	Programmed evaluation and review technique
PL	Plotter
PR	Printer
PRELIB	Program to prepare library
PT	Paper tape station (reader and punch)
PUN	Standard punch unit
RHT	Running hardware table
RHTD	Running hardware table duplicate
RIF	Relocatable information loader card

RIO	Routine to protect CIO from reentry by real-time program
RNI	Read next instruction
SCOPE	Program for supervisory control of program execution
SL	Satellite controller
ta	Trapped address
tc	Trapped condition
TP	Paper tape punch
TR	Paper tape reader
TRA	Transfer address loader card
TS	Channel typewriter
TY	Console typewriter
UST	Unit status table
XNL	External name loader card

GLOSSARY

ABNORMAL DUMP
: A dump occurring immediately following abnormal termination of a program.

ABORT
: To terminate a program when a condition (hardware or software) exists from which the program or computer cannot recover.

ABSOLUTE BINARY PROGRAM
: A program that must be loaded according to specific logical addresses.

ABSOLUTE CODE
: A code using absolute operators and addresses; i.e., a code using machine language.

ALLOCATE
: To reserve an amount of some resource in a computing system for a specific purpose (usually refers to a data storage medium).

ALPHANUMERIC
: Pertaining to the character set that contains alphabetic letters, numerical digits, and special characters which are usually machine processable.

ASSEMBLE
: To prepare an object language program from a symbolic language program by substituting machine operation codes for symbolic operation codes and absolute or relocatable addresses for symbolic addresses.

ASSEMBLER
: A computer program that generates machine instructions from symbolic input data through translating symbolic-operation coding into computer operating instructions, assigning locations in storage for successive instructions, or computing absolute addresses from symbolic addresses. An assembler generates machine instructions from symbolic codes and produces, as output, nearly the same number of instructions or constants as were defined in the input.

AUTOLOAD
: To place the resident routines of the operating system in core storage.

BACKGROUND
Replaced with priority.

BATCH
In RTS, an object program running in a stacked job manner. Shares the central processing unit with the priority program when a priority program is present and executes only when the priority program is not in control of the processor. Batch interrupts have lowest priority in the interrupt processing priority scheme.

BINARY
A characteristic property, or condition, having two alternatives; a numbering system based on 2 rather than 10 and using only 0 and 1.

BLOCK
Consecutive machine words or characters considered or transferred as a unit, particularly applicable to input and output.

BLOCKING
Combining two or more numbers (records) into one block to reduce the number of physical operations.

BLOCK LENGTH
Number of records, words, or characters in one block.

BUFFER
A magnetic core buffer external to core memory to compensate for speed differences between peripheral devices and the processor. Operations can then be overlapped with all devices operating simultaneously at rated speeds. Buffering eliminates the need for more expensive, multiple I/O channels, and reduces programming having complex I/O timing considerations.

BUFFERING
Overlapping execution of one or more I/O routines with the execution of the program that called them.

CARD COLUMN
A vertical line of punching positions on a card.

CARD IMAGE
A representation in storage of the holes punched in a card such that holes are represented by ones and unpunched spaces are represented by zeros. In machine language, a duplication of the data on a punched card.

CARD ROW
A horizontal line of punching positions on a card.

COMMON
An area of memory that may be shared between batch subprograms. Common may not be preset with data. Priority programs may not have a common area.

COMPILER

A program which translates a programming language such as FORTRAN or COBOL into an assembly language and, often, into machine language. A compiler may generate many machine instructions for a single symbolic statement.

CONTROL CARD, CONTROL STATEMENT

An instruction recognized by the operating system.

DATA AREA

An area of memory which may be preset with data at load time and shared between subprograms. Both batch and priority programs may have data areas.

DECK

A collection of punched cards that has a definite service or purpose, structured to represent a processing unit to the operating system.

DRIVER

A program that controls the use of a peripheral device.

DUMP

To copy the contents of all or part of a storage device, usually from internal storage into external storage; the process of performing the copy, or the resulting document.

END-OF-FILE

Information designating the termination point of data or of a program.

END-OF-FILE INDICATOR

A signal supplied by an input or output unit that makes an end-of-file condition known to the routine or operator controlling the device.

EQUIPMENT

An interface between a data channel and a unit; a channel controller.

ERROR

Any deviation of a computed or a measured quantity from the theoretically correct value.

EXECUTE

To carry out an instruction or perform a routine.

EXECUTION

The process whereby the instructions contained in a program direct the activities of the central processing unit.

EXTERNAL INTERRUPT

An interrupt occurring as a result of conditions within peripheral devices or their immediate interfaces. Interrupts occurring as a result of conditions within a data channel are classified external or internal in keeping with specifications set forth in individual hardware system reference manuals.

FAULT

1. A physical condition that causes a device, a component, or an element to fail to perform in a required manner; e.g., a short circuit, a broken wire, an intermittent connection; synonymous with malfunction.
2. An operation whose results exceed the capacity of one or more registers and which is detected by the hardware.

FIELD

In a record, a specified area used for a particular category of data; e.g., a group of card columns used to represent a wage rate or a set of bit locations in a computer word used to express the address of the operand.

FILE

A collection of related records treated as a unit; e.g., in inventory control, one line of an invoice forms an item, a complete invoice forms a record, and the complete set of such records forms a file.

FLAG

1. Any of various types of indicators used for identification; e.g., a wordmark.
2. A character or bit that signals the occurrence of some condition, such as the end of a word.
3. An indicator (program or hardware initiated) used frequently to tell some later part of a program that some condition occurred earlier.
4. To generate a flag (1,2,3).

FLOW

A general term to indicate a sequence of events.

FOREGROUND

Replaced by batch.

INITIALIZE

To set counters, switches, and addresses to zero or some other starting value at the beginning of or at prescribed points in a program.

INPUT

Information or data transferred from an external storage device into computer memory.

INPUT/OUTPUT

The bidirectional transmission of information between computer memory and peripheral devices.

INPUT/OUTPUT SYSTEM

The portion of the monitor that handles I/O; includes CIO, CIC, and I/O drivers.

INTERNAL INTERRUPT

An interrupt occurring as a result of conditions within computer mainframe or immediate interfaces.

INTERRUPT

1. A break in the normal flow of a system or routines such that the flow can be resumed from that point at a later time. An interrupt is usually caused by a hardware-generated signal.
2. To cause an interrupt.

JOB

A deck consisting of control cards and possibly also program and data decks; presented serially in a job stack to RTS through the standard input unit.

LIBRARY

An organized collection of standard, checked-out programs, routines, and subroutines which can be used to solve many types of problems. The entirety of the operating system is also called the library.

LINKAGE

The interconnections between subprograms or between a main routine and closed subroutines; e.g., the entry into a closed routine and the exit back to the main routine.

LOAD-AND-GO TAPE

The RTS tape designated by the logical unit 56 when RTS is initiated. This is automatically positioned to its origin when the user requests loading from the tape. When loading is complete, RTS again positions the load-and-go tape to its origin to make it available for other output. Unless the user specifies otherwise, assembly and compilation output is written on the load-and-go tape.

LOADING

The process of transferring a program from external devices to storage. In RTS the relocatable loader transfers a relocatable program to the first sequential available positions in core; the absolute loader RDCKF1 transfers programs which must be loaded into specific locations.

LOCATION

A position in storage where one computer word can be stored and which is usually identified by an address.

LOGICAL UNIT

A number that can be equated to any one of a variety of peripheral units.

MACRO INSTRUCTION

An instruction in a source language that is equivalent to a specified sequence of machine instructions. Usually, a symbolic mnemonic type instruction that a programmer can write in a source program to call for library or special routines.

MAIN

An element of a program prepared in overlays. The main element typically is a controlling program which calls overlay elements into core in succession.

MULTIPROGRAMMING

In RTS, a technique for processing two programs simultaneously by overlapping or interleaving their execution. In RTS, multiprogramming is achieved by allowing the priority program to gain control of the processor periodically through interrupts.

OBJECT LANGUAGE

The language that is the output of a given translation process; i.e., the language into which an assembler or compiler translates a source language.

OPERATING SYSTEM

An organized collection of programmed techniques and procedures for operating a computer.

ORDINAL

The location of an entry in a table.

ORIGIN

1. The absolute address of the beginning of a program or block.
2. In relative coding the absolute address to which addresses in a region are referenced.

OVERLAY

In overlay processing, an element called by the main element.

OVERLAY PROCESSING

A technique for processing a program whose total storage requirements for instructions exceed available memory. The user divides the program into elements which are stored on magnetic tapes and brought into core at different points of processing. An element of an overlay program, when brought into core memory, may occupy the same storage locations as another element which executed previously.

PARAMETER

1. A variable that is given a constant value for a specific purpose or process.
2. A quantity in a routine which specifies a machine configuration, subroutines to be called, or other operating conditions.

PRIMARY ENTRY POINT

An entry point named on a TRA card in a binary deck.

PRIORITY

A scheme for determining that a routine or job is to be executed before another. In RTS priority distinctions are applicable in the following:

1. Multiprogramming. The priority program may gain control of the processor from the batch program through interrupts; the batch program receives control of the processor only when the priority program relinquishes control.

2. Interrupts. Real-time program interrupts have highest priority processing under RTS; i.e., when an interrupt generated by a real-time program occurs, RTS immediately gives control to the real-time interrupt processing routine. Non-real-time priority programs have next highest priority; RTS gives control to the non-real-time priority program when an interrupt generated by that program occurs, except when this interrupt occurs during RTS input/output processing. In this case, RTS waits for the input/output routine to complete execution and then gives control to the non-real-time priority program. Batch program interrupts have the lowest priority. The priority program, real-time or non-real-time, may gain control of the processor through an interrupt after a batch interrupt has occurred. RTS waits until the priority program has relinquished control of the processor before routing control to the batch program interrupt routine.

3. Job stack processing. Under RTS, the priority program may submit a batch job stack, and this stack has processing priority over all batch jobs except the one currently in execution. RTS waits for completion of this job and then initiates processing of the priority-submitted batch job stack. The operator, however, has the option of altering this priority. Through the manual interrupt message B/P, he may direct RTS to bypass priority-submitted batch job stacks and continue processing batch jobs from the standard input unit. The manual interrupt message P/P restores the processing priority of the priority-submitted batch job stack.

PRIORITY PROGRAM

A specially prepared program requiring control for discrete intervals or that is I/O bound; resides in core during batch runs.

PROCESSOR

A device capable of receiving data, manipulating it, and supplying results.

PROGRAM

1. The precise sequence of coded instructions necessary to solve a problem.

2. To plan the procedures for solving a problem. This may involve, among other things, analyzing the problem, preparing a flow diagram, providing details, developing and testing subroutines, allocating storage, specifying input and output formats, and incorporating a computer run into a complete data processing system.

READ

To transfer information, usually from an input device, to internal storage.

REAL-TIME

Pertaining to a program for which time requirements are particularly stringent; that is, the data transfer must keep up with a physical process within a time period of seconds or less.

RECORD

1. A collection of related items of data treated as a unit.

2. To put data into a storage device.

RE-ENTRANT
Capable of being called into use while in use.

RE-ENTRANT CONDITION
In programs and routines that are not re-entrant, indicates the error condition that arises when a routine or program executing with one set of values or data receives a second set of values or data.

RELOCATABLE BINARY SUBPROGRAM
A program that can be contiguously loaded with the aid of a loader program into available logical memory.

RESIDENT, CORE MEMORY
That part of the system residing in core memory at all times.

RETURN
To transfer control back to a point in a program or subprogram from which a call was issued.

ROUTINE
A set of instructions arranged in a sequence such that the computer performs a desired task.

SEGMENT
An element of a program prepared in overlays. A segment element is called into core by an overlay element.

SNAPSHOT DUMP
A selective dynamic dump performed at various points in a machine run.

SOURCE LANGUAGE
Input language for a given translation process.

STATUS
A state or condition of hardware or task; e.g., busy or not busy.

SUBPROGRAM
A part of a larger program which can be converted into machine language independently.

SUBROUTINE
1. A portion of a routine that causes a computer to carry out a well-defined mathematical or logical operation.
2. A routine arranged so that control may be transferred to it from a master routine and so that, at the conclusion of the subroutine, control reverts to the master routine. Such a subroutine is usually called a closed subroutine.

SYSTEM FILES
The entirety of the operating system as it appears on the system device; sometimes called the library.

TIME-SHARING

The capability of a computing system to accommodate more than one user during the same interval of time without apparent restriction by the existence of other users. In time-sharing, a given device is used in rapid succession by a number of other devices or various units of a system are used by different users or programs. See also multiprogramming.

TRAPPED INSTRUCTION

1. An instruction that is executed by a software routine if the necessary hardware is lacking or if the central processor is not in the required state.
2. An instruction whose execution is blocked.

UNIT

A peripheral device capable of storing, receiving, transmitting, or interpreting data; connected to an equipment.

UNLOAD

To remove a tape from ready status by rewinding beyond the load point; the tape is then no longer under control of the computer.

UPDATE

1. To modify a file with current information according to a specified procedure.
2. To modify an instruction so that its operand address is changed by a stated amount each time the instruction is performed.

USER INTERRUPT

An interrupt selected by the user program through a CIO call.

USER PROGRAM

An object program loaded and entered under RTS control; includes batch and priority programs and library routines.

UTILITY ROUTINE

A routine in general support of the operation of a computer; e.g., an input/output, diagnostic, tracing, or monitoring routine.

WRITE

To transfer information, usually from internal storage, to an output device.

CONTROL DATA
CORPORATION

COMMENT AND EVALUATION SHEET

31/32/33/35 REAL-TIME SCOPE
Reference Manual

Pub. No. 60172500A March 1969

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 REFERENCE.

FROM NAME: _____

BUSINESS ADDRESS: _____

_____

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
Software Documentation
4201 North Lexington Avenue
St. Paul, Minnesota 55112

MD 248

FOLD

FOLD

STAPLE

STAPLE

► ►CUT OUT FOR USE AS LOOSE-LEAF BINDER TITLE TAB

31/32/33/3500 REAL TIME SCOPE REFERENCE MANUAL

CONTROL DATA
CORPORATION

CORPORATE HEADQUARTERS, 8100 34th AVE. SO., MINNEAPOLIS, MINN. 55440
SALES OFFICES AND SERVICE CENTERS IN MAJOR CITIES THROUGHOUT THE WORLD

Pub. No. 60172500

Litho in U.S.A.