

Systems Reference Library

IBM 1410/7010 Operating System (1410-PR-155) System Monitor

This publication provides programmers and systems analysts with the basic principles and programming requirements for the System Monitor. The programming information given includes the control cards required, the linkage sequences to Monitor routines and data fields, the messages produced by the Monitor, and the console inquiries accepted.

The System Monitor provides control for the IBM 1410/7010 Operating System by furnishing programmed transition between runs and between jobs, relocating programs, and assigning input/output units. The System Monitor consists of the Resident and Transitional Monitors, 1410-SV-962, and the Linkage Loader, 1410-UT-963.

The reader is assumed to have a basic knowledge of the 1410 and/or 7010 Data Processing Systems and to have read the publication, *IBM 1410/7010 Operating System; Basic Concepts*, Form C28-0318.

Major Revision (November 1963)
This publication is a major revision of, and obsoletes, the publication, *IBM 1410/7010 Operating System; System Monitor*, Form C28-0319, and associated Technical Newsletter N28-1077.

Copies of this and other IBM publications can be obtained through IBM Branch Offices. Address comments concerning the contents of this publication to: IBM Corporation, Programming Systems Publications, Dept. D91, PO Box 390, Poughkeepsie, N. Y.

Contents

The System Monitor	5	Simple Search of System Tape Subroutine	24
Purpose of this Publication	5	Read System Disk Subroutine	. 34
Purpose of the System Monitor	5	Use of End-of-Program Routine	. 04.
Prerequisite and Related Literature	5	Normal End of Program	. აა
The state of the s	J	Unusual End of Program	. 35
		Special End of Program	ാര
System Monitor Concepts	6	opecial End of Hogiani	36
Use of the System Monitor	7		
Batch Processing	7	Information for Dependent Programs	37
Program Construction	7	Use of the Resident locs	37
Communication Symbols	8	No Halts	37
Linkage Symbols	8	Index Registers	37
System Symbols	8	Word Separator Characters	37
Listing of Communication Symbols	9	Priority Alert Mode	37
		Unit-Record Interrupts	37
Structure and Eurotions of the Contain Manites	10	/sız/ Field	37
Structure and Functions of the System Monitor		Reading from the Standard Input Unit	37
Structure of the System Monitor	10	Divide Overflow and Arithmetic Overflow Indicators	37
Functions of Resident Monitor		Two Highest Positions of Core Storage	38
Input/Output Control System	10	System Regeneration	38
Unit-Record Routines	10	_	
Assignment Routine	10	Use of Transitional Monitor	
End-of-Program Routine	10	von Card	39
Load Routine	10	JOB Card	39
Communication Region	11	MODE Card	39
Wait-Loop Routine	11	exeq Card	41
Console Inquiry Routine	11	COMT Card	41
Initialization Routine	11	ASGN Card	42
Functions of the Transitional Monitor	11	END Card	42
Control Card Interpretation Routine	11	DATE Card	42
Job Routine	11		
Functions of the Linkage Loader	11	Assignment of Input/Output Units	43
Input/Output Files	11	Physical Units	43
System Files	12	Symbolic Units	43
		Assignment Times for Symbolic Units	44
Use of Linkage Loader	1.4	Program References to Symbolic Units	45
Control Cards for the Linkage Loader	14	ASGN Card	45
CALL Card	15	Use of the ASGN Card for System Files	46
CALLN Card	15 19	Linkage Sequence for Assignment of Symbolic Units	46
CALLP Card		o i	10
PHASE Card	20 20	Massacra and Landelton	
BASEL Card		Messages and Inquiries	47
PRTCT Card	$\frac{21}{22}$	Memory Map	47
BASE2 Card		Monitor Diagnostic Messages	47
SNAP Card	24	Diagnostic Messages for Programmers	47
INPUT Card	24	Messages for Operators	49
TITLE Card	25	Initialization Messages	51
DISGO Card		Bootstrap Routine Messages	52
congo Card	25	Messages during Restart	53
CONGO Caru	26	Linkage Loader Diagnostic Messages	53
		Console Inquiries	54
Use of Resident Monitor Functions	27	Class A Messages	54
Use of Communication Region	27	Class B Messages	55
Contents of the Communication Region	27	Class C Messages	56
Modification of the Communication Region	27		
Use of Unit-Record Routines	27	Program Patching	57
Read Routine	27	Relocatable Object Cards	57
Print Routine	31	TITLE Card	57
Punch Routine	32	Termination Card (for Subprograms)	57
Use of Console Inquiry Routine	32	Load Card	
Use of Wait-Loop Routine	33	DEFIN Card	58
Use of Load Routine	33	BASEL Card	58
Read System Tape Subroutine	33	BASE2 Card	59
Backspace System Tape Subroutine	34	PRTCT Card	59 59
Rewind System Tape Subroutine	34	CALL Card	59 59
			UD

Random Load Card Clear Storage Card Set Word Mark or Record Mark Card Clear Word Mark Card	59 59	Example 1 Example 2 Restarting from a Checkpoint Immediate and Delayed Restarts Restart Conditions	62 64 64
Operation of the System Monitor Initialization of the System Monitor Functioning of the System Monitor	60	Index	66

Purpose of this Publication

This publication describes use of the System Monitor to control the 1410/7010 Operating System. Information is included about the functions and components of the System Monitor and its relationship to the Operating System. The publication also gives instructions for writing programs to run under control of the System Monitor.

Purpose of the System Monitor

The System Monitor controls the 1410/7010 Operating System. Some of the functions that it performs for programs within the Operating System are:

- 1. Assignment of input/output units.
- 2. Program loading, including relocation of programs, and linkage between programs and subroutines that were independently written and compiled.
- 3. Programmed transition from run to run and job to job.

The Operating System can provide, at the option of each installation, control facilities for IBM Tele-Processing® Systems. Detailed information concerning this type of application is contained in the publication IBM 1410/7010 Operating System; Tele-Processing Supervisor, Form C28-0321.

Prerequisite and Related Literature

The publication, *IBM 1410/7010 Operating System*; *Basic Concepts*, Form C28-0318, is a prerequisite. The reader is assumed to be familiar with both the terminology and concepts defined by that publication.

For a disk-oriented 1410 or 7010 Data Processing System, the publication, *IBM 1301 Disk Storage with IBM 1410 and 7010 Systems*, Form A22-6670, is a prerequisite.

Related literature includes the following publications, which contain detailed information about the other elements and functions of the 1410/7010 Operating System:

System Generation, Form C28-0352
Basic Input/Output Control System, Form C28-0322
Tele-Processing® Supervisor, Form C28-0321
Utility Programs, Form C28-0353
Generalized Tape Sorting Program, Form C28-0354
Random-Processing Scheduler, Form C28-0323
Autocoder, Form C28-0326
FORTRAN, Form C28-0328
COBOL, Form C28-0327
Operator's Guide, Form C28-0351

System Monitor Concepts

The System Monitor coordinates the functions of all Operating System components, including user-written programs operating under it. In this coordination, the System Monitor does the following:

- 1. It reads and analyzes control cards that describe functions to be performed.
 - 2. It assigns input/output units.
- 3. It reads programs from the card reader, tapes, or disk. These programs may be requested by control cards or by imbedded calls within programs.
- 4. It relocates these programs to unused areas of storage, provides communication among program elements, and loads the programs into storage.
- 5. It turns processing control over to the programs in storage and receives control from them when they are completed. These dependent programs may be user-written programs or IBM-provided components of the Operating System.
- 6. It handles all input/output operations for dependent programs.

To perform these control functions, a part of the System Monitor, the *Resident Monitor*, stays in lower core storage at all times. The Resident Monitor loads programs into core storage, handles input/output, and controls end-of-program actions.

The rest of the System Monitor and all other components of the Operating System are kept on tape or disk and are brought into core storage as required. The other parts of the System Monitor are:

- 1. Transitional Monitor, which analyzes control cards and assigns input/output units.
- 2. Linkage Loader, which relocates programs and provides communication among independently compiled program elements.

Programs compiled by the language processors are in relocatable format. Relocatable programs have been arbitrarily assigned locations by the language processors or the programmer. The Linkage Loader (1) relocates the addresses in the relocatable programs to core locations above the Resident Monitor, and above other programs that will be in storage below the program; or (2) relocates the addresses to installation-specified locations. After relocation, programs are in absolute format. The absolute addresses in these programs are the addresses the programs will use when they are executed.

The over-all operation of the System Monitor is shown in Figure 1.

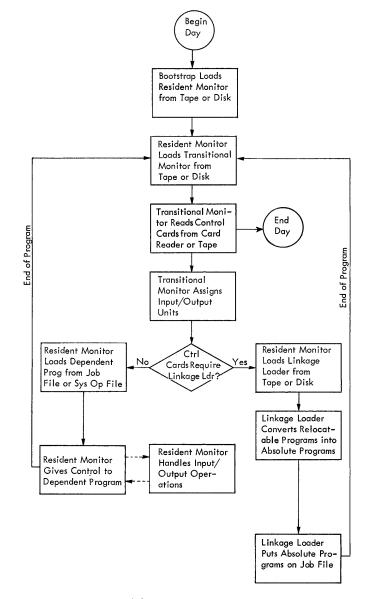


Figure 1. Operation of the System Monitor

Programs to be tested can be run under control of the System Monitor in the same way as fully tested programs. One or more Utility programs of the Operating System can be run at selected points during program execution and/or at the end of program to dump core storage, tapes, or disk.

Use of the System Monitor

Five types of control information are supplied by the installation to the System Monitor:

- 1. Monitor control cards. Monitor control cards are identified by Mon\$\$ in columns 6 through 10. These cards initiate program execution, assign input/output units to programs, and supply certain information to the System Monitor. They are described in the section, "Use of Transitional Monitor."
- 2. Linkage Loader control cards. These cards select the elements composing a program and control program relocation. They are described in the section, "Use of Linkage Loader."
- 3. Imbedded calls. These are requests coded in a dependent program to incorporate elements into the program. They are described in the section, "Use of Linkage Loader."
- 4. Linkage sequences. These are coding sequences within a dependent program that enable the program to use facilities of the Resident Monitor. Each sequence contains a branch instruction and may also contain one or more fields of control information. The branch instruction gives control to a routine in the Resident Monitor. These sequences are described in the section, "Use of Resident Monitor Functions."
- 5. Console inquiries. These are control messages typed on the console printer by the operator. They are described in the section, "Messages and Inquiries."

Batch Processing

A major advantage of the System Monitor is that it permits batching programs, that is, running programs in groups, with minimum operator intervention. Program batches are subdivided into:

- 1. Jobs
- 2. Runs

The day's work is divided into jobs. Each job consists of one or more program executions, called runs, plus control information. All programs in a job must have compatible input/output assignments.

The usual job is a request for a program in absolute format, residing on tape or disk, plus the necessary control information and input data.

However, almost limitless combinations of runs can be performed in a single job. For example, a job can include a compilation, relocation of all programs in the job, execution of programs to test them, use of one or more of the Utility Programs, and execution of one or more fully tested production programs.

Programs in source language may just be compiled, or may be compiled and executed (compile-and-go op-

eration) within one job. However, if errors are found in compilation, the request for execution of the compiled program will be automatically canceled.

Program Construction

For the Operating System, a program is constructed of one or more phases. Many programs will consist of only one phase. If a program is too large for storage, it must consist of several phases; the phases are executed in succession, with each phase partly or completely overlaying the preceding phase.

Each phase consists of one or more subprograms. A subprogram is the basic program unit with which the Linkage Loader performs its processing.

For the purposes of the Linkage Loader, a subprogram begins with a TITLE card and ends with a Termination card. The cards between the TITLE and Termination cards contain the relocatable statements produced by a language processor. (See the section, "Use of Linkage Loader," for a description of the TITLE card. Termination cards are described in the section, "Program Patching.")

A subprogram can be an entire, self-contained program, or it can be a closed subroutine to be executed in conjunction with other subprograms to form a phase.

The subprograms in a job may be in source language, requiring compilation, or in relocatable form. If a job contains more than one subprogram, it may contain a mixture of source and relocatable decks. In a job, a relocatable subprogram may accompany the control cards, or it may reside in a program library on tape or disk. In the latter case, the control cards must include a call for the subprogram, or another subprogram in the job must contain an imbedded call.

Subprograms are divided into two types: primary and secondary. A primary subprogram has an initial entry point specified by its Termination card. A primary subprogram may be a phase in itself or may form the nucleus for a group of subprograms to be made into a single, executable phase. A secondary subprogram does not have an initial entry point on its Termination card. Secondary subprograms may compose a complete phase or may be combined with a primary subprogram to form a phase.

In Autocoder, for example, the distinction between primary and secondary subprograms is based on the END or SPEND statements. If the programmer specifies an address in one of these statements, that address is used for the initial entry point of the subprogram, and the language processor informs the Linkage Loader that this is a primary subprogram. If the operand of the END or SPEND statement is blank, the language

processor informs the Linkage Loader that this is a secondary subprogram.

A program must contain at least one primary subprogram to establish the initial entry point for execution of the program. For a multiphase program, the first phase must include at least one primary subprogram. Succeeding phases may or may not contain primary subprograms. If the succeeding phase is of an executable nature, the preceding phase must contain the linkage to initiate its execution.

The initial entry point of a program phase is established by the Termination card of the first primary subprogram processed for that phase. The Linkage Loader ignores entry points specified in other Termination cards processed within that phase.

Communication Symbols

The System Monitor enables use of symbolic references, called *communication symbols*, so that a subprogram can refer to a routine in another separately compiled subprogram and to parts of the Resident Monitor.

The two types of communication symbols are:

- 1. Linkage symbols, which are used for communication between subprograms.
- 2. System symbols, which are used for communication from a subprogram to routines or data fields of the Resident Monitor.

Autocoder includes language statements that directly relate to the use of communication symbols. For the most part, COBOL and FORTRAN automatically generate these symbols as they are needed in a program.

Linkage Symbols

Linkage symbols are established by two types of cards: the TITLE card at the beginning of a subprogram, and Definition cards included in a subprogram. (Autocoder includes the DEFIN statement for the creation of Definition cards.)

Each time the Linkage Loader encounters one of these cards, it places into its symbol table the characters that have been declared as the linkage symbol and the relocated address assigned by the Linkage Loader to the symbol. For example, a TITLE card declares a subprogram name as a linkage symbol, and the origin point of the subprogram (after relocation) becomes the address assigned to that linkage symbol. This procedure enables a subsequent subprogram to refer to a previously processed subprogram by the linkage symbol that is the previous subprogram's name.

A linkage symbol can appear in a subprogram in either of two formats:

- 1. A symbol consisting of one to ten characters, such as a subprogram name. The first character must be alphabetic; the remaining characters can be alphabetic or numeric. No special characters can be used. If the symbol is less than ten characters, it is leftiustified, with trailing blanks in a ten-position field.
- 2. A five-character symbol consisting of four characters followed by a slash (ABCD/). The first character must be alphabetic; the next three characters can be alphabetic or numeric.

Reference to linkage symbols of the first format are made by DCWS and DCWF cards. COBOL and FORTRAN automatically generate these cards as needed. Autocoder includes language statements for the generation of these cards. When the Linkage Loader encounters a DCWS card, it puts into the subprogram a branch instruction containing the address assigned to the linkage symbol in the DCWS card. A DCWF card causes the Linkage Loader to put into the subprogram a five-character constant that is the address equated to the linkage symbol.

Note: Dows and Dowf cards can contain only oneto ten-character linkage symbols.

DCWs and DCWF cards are termed *imbedded* calls because they are requests for subprograms coded into a subprogram.

References to linkage symbols of the ABCD/ format can be made by using the symbol in the A-field or B-field of an instruction or by using the symbol as an address constant. For example, in the Autocoder language, the programmer can use the DEFIN statement to declare work/ as a linkage symbol that represents the address of a work area. The programmer can then use work/ in the operand of an instruction that refers to that work area, such as S+10,work/. Such an instruction can be included in any other subprogram processed by the Linkage Loader during the same run as the subprogram that declared the linkage symbol.

Note that linkage symbols are designed for use as references between separately compiled subprograms. Each of the language processors provides facilities for references within a subprogram. In Autocoder, these references (labels) are directly coded; the COBOL and FORTRAN processors generate references as required.

System Symbols

System symbols are defined by the System Monitor and require no declaration by subprograms. Subprograms need only refer to them. These symbols, given in the section, "Use of Resident Monitor Functions," refer to routines and data in the Resident Monitor.

All system symbols have the format /ABC/. These

symbols can appear in the operand of any instruction or as an address constant. The Linkage Loader converts all system symbols to actual addresses when it relocates the dependent programs.

Listing of Communication Symbols

The Linkage Loader produces a listing of all linkage symbols used during the processing of subprograms. Each symbol is listed with the core-storage address it represents. Thus, the listing shows the location of program elements after they have been relocated.

For all system and linkage symbols that are not defined, the Linkage Loader will:

- 1. Print all unresolved symbols. This list includes undefined communication symbols and all unfilled calls for subprograms.
- 2. Leave undefined communication symbols in the instruction in which they appeared.

The Linkage Loader will permit programs with undefined communication symbols or unresolved calls to be executed; however, these programs will fail if execution is attempted of an instruction containing an unresolved call or undefined communication symbol.

Structure and Functions of the System Monitor

Structure of the System Monitor

The System Monitor consists of three major elements:

- 1. Resident Monitor. This element contains the control routines required while running dependent programs and, therefore, remains in core storage at all times.
- 2. Transitional Monitor. This element performs functions related to the transition from run to run and job to job. It is brought into storage by the Resident Monitor when transitional functions are needed. One of its major functions is the analysis and processing of Monitor control cards.
- 3. Linkage Loader. This element performs the functions required to prepare relocatable programs for execution. It is brought into storage by the Resident Monitor when requested by Monitor control cards.

Functions of Resident Monitor

Input/Output Control System

The Input/Output Control System (10CS) is created during System Generation to meet the requirements of the installation. This 10CS, termed the *Resident IOCS*, is in the Resident Monitor. It contains the basic routines, such as those for error checking, that perform functions common to all input/output operations. (For detailed information concerning the Resident 10CS, see the publication, *Basic Input/Output Control System*.)

Unit-Record Routines

The Resident Monitor contains three routines to handle input/output operations for unit-record equipment. One of these routines is for card input (Read routine), another for card output (Punch routine), and the third for printer output (Print routine). Each of these routines has an alternate routine that performs the unit-record functions with magnetic tape. For each unit-record function, the installation can specify whether the function is to be performed with magnetic tape or unit-record equipment.

Assignment Routine

Facilities for assigning input/output units for program use are included in the Resident Monitor and the Transitional Monitor. The Resident Monitor contains tables of information about the installation's input/output equipment, and facilities for providing this

information as required during program execution. The Transitional Monitor analyzes control card information related to input/output assignment and coordinates this information with the Resident Monitor's Assignment routine.

End-of-Program Routine

The End-of-Program routine analyzes each end of program to determine the function to be performed next. If it is a normal end of program (successful completion of the program being executed), the Transitional Monitor is brought into storage to process the Monitor control cards that define the next run.

If the end of program is caused by program failure (unusual end of program), the End-of-Program routine preserves the contents of core storage on tape, if a tape was provided for that purpose. The Transitional Monitor then processes the Monitor control cards for the next run, if the job or program is being tested, or skips to the control cards for the next job.

A third type of end program, special end of program, is used by the Linkage Loader and the language processors to indicate that the contents of their output files are not valid.

Load Routine

The Load routine brings programs, which have been relocated and are ready for execution, into storage. These absolute-format programs are on the Job file produced by the Linkage Loader, or on the System Operating File. The Load routine loads from the file named in the Monitor control card that executes the program. (The files are discussed later in this section.)

The Load routine also searches for phases required during the execution of a job. For example, this routine is used by one phase of a multiphase program to locate the next phase and bring it into storage. Each program phase, at its completion, specifies to the Load routine which phase is to be located next. Note that the next program phase need not be the next in numerical sequence.

For example, Phase 1 of a program could issue a request for Phase 2 or any other phase, basing its choice on the result of the processing in Phase 1. Dependent programs may also use the subroutines of the Load routine to read, search, backspace, and rewind the file from which the program was loaded.

Note: The Load routine cannot load the two highest core positions. The Operating System places group

marks with word marks in the two highest core positions; these group marks with word marks must not be destroyed by dependent programs.

Communication Region

The Communication Region is a collection of fields that contain control information. Some fields contain information used for communication between various IBM programs within the Operating System; others contain information of value to the user's programs. Some fields contain data; other fields contain addresses.

For example, one Communication Region field contains the current date. This field can be addressed by any program. The Resident 10cs uses this field to determine the date for writing and checking tape labels. The user's programs can address this field for such purposes as creating a header record for printer output.

Communication Region fields can be modified only through the Resident Monitor's Modification routine, which is described in the section, "Use of Resident Monitor Functions."

Wait-Loop Routine

The Wait-Loop routine allows programs to suspend processing until the operator performs a specified function, such as changing forms on the printer. A program can write a message on the console printer, branch to the Wait-Loop routine, and resume processing after the operator signals the Resident Monitor that the function specified by the message has been completed.

Console Inquiry Routine

Facilities for responding to console inquiries are included in both the Resident Monitor and the Transitional Monitor. The Resident Monitor accepts console inquiries during a run, such as a signal from the operator to exit from the Wait-Loop routine. The Transitional Monitor accepts console inquiries between jobs, such as a signal to begin reading from the Alternate Input Unit.

The Console Inquiry routines (in both the Resident and Transitional Monitors) serve to communicate control information to the System Monitor. Programs operating under control of the System Monitor can also use the console printer for communication with the operator.

Initialization Routine

The Initialization routine performs housekeeping to prepare the System Monitor for work. This routine is at the top of the Resident Monitor area and is overlaid by the first dependent program brought into storage. The functions and use of the Initialization routine are explained in the publication, *Operator's Guide*.

Functions of the Transitional Monitor

Control Card Interpretation Routine

When the Transitional Monitor is brought into storage, one of its major functions is the interpretation of the Monitor control cards that define the next run or job. The Control Card Interpretation routine analyzes the Monitor control cards and, in accordance with their specifications, gives control to the System Monitor elements required to prepare and execute the next run.

Job Routine

The Job routine prepares the Resident Monitor for the next job. This preparation includes resetting various program switches, clearing certain areas in the Communication Region, and canceling certain input/output unit assignments.

Functions of the Linkage Loader

The Linkage Loader converts relocatable subprograms into absolute format and, in the process, resolves communication symbols into machine addresses.

Each Linkage Loader run produces a Job file, which contains program(s) in absolute format. Each program consists of all the subprograms to be executed as one program. A program on the Job file is brought into core storage by the Resident Monitor's Load routine when a Monitor control card requests execution of that program.

Input to the Linkage Loader is from the Standard Input Unit, the Go file, and/or the System Library file. (These input/output files are discussed in the following topic, "Input/Output Files.") When several subprograms are to be combined into a single program, the subprograms can be supplied to the Linkage Loader from any combination of the three input files, or they can all be from the same file.

Selection of the subprograms to be converted into a single program is determined by control cards or by imbedded calls in the subprograms being processed.

The Linkage Loader has the ability to incorporate patches or Snapshot requests into a previously compiled subprogram.

Input/Output Files

The input/output files for the Operating System and the dependent programs running under it are divided into four categories:

- 1. System Files. These input/output files, described below, are used by the System Monitor.
- 2. Work Files. These tape, disk, or unit-record files are used by Operating System programs and can be

used by installation programs. The symbolic names for these files are:

мw0 (мw-zero) through мw9

мwa through мwz

3. Reserve Files. These tape, disk, or unit-record files are reserved for use by installation programs. The symbolic names for these files are:

MRO through MRO MRA through MRZ

4. Tele-Processing System Files. These tape, disk, or unit-record files are used by dependent programs that operate under control of the Tele-Processing Supervisor. The symbolic names for these files are:

мт0 through мт9 мта through мтz

System Files

SYSTEM OPERATING FILE

The System Operating File contains Operating System programs in absolute format. This file contains the System Monitor and can contain the language processors, the Utility programs, and the Sort Definition program.

The System Operating File is created at System Generation and is constructed according to the program requirements of each installation. When the file is created, the user may add user-written programs, such as an inventory or payroll program.

The first element of the System Operating File is the Bootstrap routine, which initially loads the Resident Monitor into core storage. The following elements are IBM-supplied and installation components of the Operating System.

The System Operating File may be on tape or disk. The symbolic name for this file is sor.

SYSTEM LIBRARY FILE

An installation can have any number of libraries of relocatable programs, but only one is designated as the System Library file for a particular run of the Linkage Loader.

A library may be on tape or disk. If it is on tape, it can be on the same tape as the System Operating File or it can be on a separate reel. The System Operating File tape can contain several libraries. If a library is on a separate tape, this tape can contain only one library. The symbolic name for the System Library file is LIB.

Libraries are created through use of the System Generation maintenance programs. (See the System Generation publication.)

STANDARD INPUT UNIT

The Standard Input Unit contains the file of control information for the System Monitor. Source programs for the language processors, relocatable programs to be processed by the Linkage Loader, and data and control cards for dependent programs may also be placed in the Standard Input Unit.

The Standard Input Unit can be a card reader or a magnetic tape unit. The symbolic name for the Standard Input Unit is siu.

Note: The term *card* means one 80-character input record. This record can be read either from an actual card, in installations that use a card reader for the Standard Input Unit, or it can be a tape record of 80 characters. Tape files consisting of this type of record are said to be in *card-image format*.

ALTERNATE INPUT UNIT

At the option of each installation, capability for using the Alternate Input Unit may be specified at System Generation. Control information can then be submitted to the System Monitor through either the Alternate or the Standard Input Unit. This configuration permits a high-priority job in the Alternate Input Unit to break into the previously established sequence of jobs in the Standard Input Unit.

The Alternate Input Unit can be a card reader or a magnetic tape unit; it may be the same type of unit as the Standard Input Unit or it may be different. The type of unit for the Alternate Input Unit can be changed by a Monitor control card. The symbolic name for the Alternate Input Unit is AIU.

Requests for changing from the Standard Input Unit to the Alternate Input Unit can be made between jobs through a console inquiry.

STANDARD PRINT UNIT

The Standard Print Unit is used for all printer output from IBM programs within the Operating System. Diagnostic messages for the programmer are written on this unit. It can also be used by the installation's programs.

The Standard Print Unit can be either the IBM 1403 Printer or a magnetic tape unit. If it is a tape unit, it can be the same unit as the Standard Punch Unit, which is described below. The symbolic name for the Standard Print Unit is SPR.

The Standard Print Unit can be omitted from a system; however, omission means that the programmer will not receive diagnostic messages or language processor listings.

STANDARD PUNCH UNIT

The Standard Punch Unit is used for all punched-card output from IBM programs within the Operating System. It is also available for use by the installation's programs.

The Standard Punch Unit may be either the IBM 1402 Card Read Punch or a magnetic tape unit. If it is

a tape unit, it can be the same unit as the Standard Print Unit. The symbolic name for the Standard Punch Unit is spu. This unit can be omitted from a system.

When the Standard Punch and/or Standard Print Units are tape, the pow program can be used on an IBM 1401 to print and/or punch the tape records. When use of the pow program is specified at System Generation, the Transitional Monitor writes the pow program as the first record on the tape when the tape is opened, either during initialization of the system or at the direction of the operator. (The pow program is described in the *Operator's Guide* publication.)

JOB FILE

The Job file is the output file of the Linkage Loader. It contains programs in absolute format. Job file programs are loaded into storage by the Resident Monitor's Load routine.

The Job file must be assigned to the same type of device (tape or disk) as the System Operating File. The symbolic name for the Job file is MJB.

GO FILE

The Go file contains relocatable programs compiled by the language processors. The Go file may contain one or more relocatable subprograms that were output from one or more language processor runs in the same job. The Go file serves as one of the input sources for the Linkage Loader.

The Go file may be on tape or disk, as specified for each language processor during System Generation. The symbolic name for the Go file is MGO.

CORE IMAGE FILE

The Core Image file is used by the Resident Monitor to record the status of core storage at particular times during processing. This file is used by the Resident Monitor's End-of-Program routine each time an unusual end of program occurs. (The records written by the End-of-Program routine can later be used as input to the Utility program's Storage Print program, which edits and writes them on the Standard Print Unit.)

The Core Image file serves programs that use the IOCS macro-instruction for writing checkpoint records. Checkpoint records can be used to restart a program that was temporarily interrupted.

The Core Image file can also be used by dependent programs to log read errors. (See the *Basic Input/Output Control System* publication.)

Use of the Core Image file is optional, but if it is included in an installation, it must be assigned to a tape unit. The symbolic name of the Core Image file is MDM.

Use of Linkage Loader

The output from each of the language processors is in relocatable format to permit subprograms to be assigned to any available area of core storage. The Linkage Loader assigns a subprogram to a particular area of storage in accordance with three factors:

- 1. The size of the installation's Resident Monitor.
- 2. The size and relative location of any other subprograms to be loaded at the same time.
- 3. Control information specifying predetermined storage assignments.

Subprograms are *compiled* with address assignments relative to *Absolute Zero* (machine location 00000). That is, program instructions are compiled by the Autocoder and COBOL language processors, from location 00000 upward or, by the fortran processor, from location 00001 upward. The Autocoder processor can also compile from the origin point specified by an ord card.

Subprograms are relocated by the Linkage Loader relative to an address in core storage called Base Zero. Base Zero is the address of the location immediately above the last location occupied by the installation's Resident Monitor minus its Initialization routine. (For installations that permanently reserve an area above the Resident Monitor for programs within a Tele-Processing system, Base Zero is the location immediately above that reserved area.) Base Zero is, therefore, the lowest location available to dependent programs. Figure 2 illustrates the relationship of Absolute Zero and Base Zero.

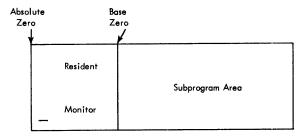


Figure 2. Relationship of Base Zero to Absolute Zero

When a subprogram is processed by the Linkage Loader, the subprogram's origin point is relocated in accordance with the installation's Base Zero. If, for example, the installation's Resident Monitor ended at location 10000, the Linkage Loader assigns an actual origin point of 10001 to the first subprogram it processes. Other addresses contained in the subprogram are relocated accordingly.

This type of relocation is termed *upward relocation*. Upward relocation is applied to the load addresses assigned by the language processor to locate instructions and areas in core storage and to the reference addresses in the A-fields and B-fields of the subprogram's instructions. Thus, the subprogram can be assigned to any area of core storage, yet interrelationship of the subprogram's elements is maintained.

Note: The Linkage Loader will not permit a dependent program to overlay the Resident Monitor. If a load address, after relocation, is below Base Zero, the Linkage Loader will stop processing the program and will return control to the Resident Monitor. The Resident Monitor will bring in the Transitional Monitor, which will resume processing with the next job.

Upward relocation is the most common type of relocation used for a subprogram. However, the Linkage Loader performs two other types of relocation within subprograms.

One of these is downward relocation. Where separately compiled subprograms are to share data areas, common data areas are used. These areas are located at the top of storage or at addresses specified by the programmer. Normally the specified address should be near the top of storage. The common data areas are addressed at the high-core position. When assigning addresses to common data areas, the language processors, except Autocoder, arbitrarily designate 99999 as the top of storage. The Linkage Loader then relocates the addresses downward to the actual top of storage or to the specified location.

The third type of relocation is *NO relocation*. The Linkage Loader does not relocate certain fixed addresses, such as index register addresses.

The value used by the Linkage Loader to perform upward relocation is called the upward relocation factor. For the first subprogram in core storage, the relocation factor is equal to Base Zero; for succeeding subprograms in the same phase, the relocation factor is equal to Base Zero plus the size of all preceding subprograms in that phase.

The subprograms that form a given phase can be in any order in core storage; a primary subprogram need not be the first. Note: During compilation, a character must be placed in the highest location of a subprogram. (That is, a subprogram cannot end with an area into which nothing is loaded, such as an area defined by an Autocoder da statement that does not set a group mark with word mark in its highest position.) Some character is necessary in the subprogram's highest location to enable the Linkage Loader to properly relocate any following subprograms.

This requirement is automatically satisfied by the COBOL and FORTRAN language processors. The requirement can be met for an Autocoder compilation by a statement that defines a one-character constant in the subprogram's highest location.

Control Cards for the Linkage Loader

Control cards for the Linkage Loader direct processing during one execution of the Linkage Loader, rather than directing the general flow of processing during a batch. Linkage Loader control cards are not Monitor control cards; that is, they do not contain the identification MON\$\$.

The Monitor control card that initiates execution of the Linkage Loader must immediately precede the control cards for the Linkage Loader. The Linkage Loader determines that it has come to the end of its control cards when it finds a card that is not in a format it recognizes. This card must be a Monitor control card.

Figure 3 lists the Linkage Loader control cards, summarizes the important information about them, and shows their formats.

CALL Card

The operand of the CALL card is the name of the subprogram to be processed by the Linkage Loader. This card is used to request a subprogram from the System Library file or the Go file. A subprogram in the Standard Input Unit does not require a CALL card; its TITLE card serves the same function. When the Linkage Loader encounters a CALL card, it adds the specified name to a list of requests. When the Linkage Loader is ready to begin processing the subprograms on this request list, it searches for them first in the Go file and then in the System Library file.

This search is performed by reading the names of the programs on the Go file and the System Library file and comparing them with the names on the request list. This technique minimizes the time required for searching.

Because the Linkage Loader uses this technique to minimize search time, the final sequence of subprograms in core storage is not necessarily the same as the CALL cards that requested the subprograms. (The CALLN and CALLP cards, described later, can be used to control the relative positioning of subprograms in core storage.)

For example, assume that the following subprograms are arranged sequentially on the System Library file: SP1, SP2, SP3, SP4, and SP5. Assume also that CALL cards were read from the Standard Input Unit in the order: CALL SP3, CALL SP1, CALL SP4. The Linkage Loader begins its search of the System Library file and locates the TITLE card for spl. It checks this name against the names on the request list and finds that there is a request for spl; therefore, it processes spl and places it on the Job file. The Linkage Loader then searches forward in the System Library file and locates the TITLE card for sp2. It checks the request list, finds no request for sp2, and continues the search. The TITLE card for sp3 is found in the Library and checked against the list. The Linkage Loader finds a request for sp3 and, therefore, processes it and places it on the Job file behind sp1. This procedure continues until all the subprograms on the request list have been located, processed, and placed on the Job file.

The final sequence of subprograms would be spl, sp3, sp4 – the order in which they were located on the Library. (The relative location of subprograms in storage is referred to as the memory map.) The relocation factor for the three subprograms was adjusted as follows: spl was given an origin point of Base Zero, and all upward relocation for spl was performed with a relocation factor equal to Base Zero; sp3 was given an origin point of Base Zero plus the size of spl, and all upward relocation for sp3 was performed with a relocation factor equal to Base Zero plus the size of spl; sp4 was given an origin point of Base Zero plus the combined size of spl and sp3, and all upward relocation for sp4 was performed with a relocation factor equal to Base Zero plus the combined size of spl and sp3.

IMBEDDED CALLS

In addition to requests for subprograms specified by CALL cards, Autocoder language DCWS, DCWF, and CALL cards are requests for subprograms. Because these cards are within subprograms, this type of request is termed an *imbedded call*.

When the Linkage Loader encounters an imbedded call, it checks its symbol table for the name of the called subprogram. If it finds the name, the subprogram has already been processed, since its TITLE card established the name as a linkage symbol. Then the Linkage Loader replaces the imbedded call, for a DCWs card, with a branch to the address assigned to

Location in		Format		Functions	Remarks		
Control Card Deck	6 15	15 16 Card 20 21 Op Type					
Beginning of first phase in program		PHASE	PROGMNAME1	Terminates the previous phase, be- gins a phase, causes all previous calls to be processed. Defines	Erases linkage symbols from Linkage Loader symbol table.		
Beginning of succeeding phase		PHASE		PROGMNAME1 as the name of the program.	Columns 60 through 70 must be blank.		
Beginning of succeeding phase (user-specified phase number)	005	PHASE			User specification of succeeding phase number, which must be in ascending order, need not be sequential, and may not be blanks or 999. Columns 60 through 70 must be blank.		
Immediately before BASE1 or PHASE card; anywhere within a subprogram except between TITLE and BASE1 cards		PRTCT PRTCT PRTCT PRTCT	LINKSYMBOL ABCD/ 15000	Keeps Linkage Loader from erasing linkage symbols with addresses equal to or higher than operand. Blank-operand card removes the protection.	Protection is changed by another PRTCT card and is effective only for the current Linkage Loader run.		
Immediately after TITLE card (if sub- program is on System Library file and is to		BASE1 BASE1 BASE1	LINKSYMBOL ABCD/ 15000	Replaces current relocation factor with address for the link-age symbol or with absolute address in operand.	Erases from Linkage Loader symbol table linkage symbols with a value equal to or higher than the BASE1 value but less than PRTCT value.		
be relocated to same point each time) or immediately after PHASE card or		BASE1	*+X00	Sets current relocation factor to next higher address that is a multiple of 100	BUT Jess Hight Kitch Value.		
PHASE and PRTCT cards		BASE1		Sets relocation factor to Base Zero.			
Within control cards for a phase or within a subprogram		CALL	SUBP GNAME2	Calls subprogram from System Library or Go file when all calls are collected.			
Within control cards for a phase		CALLN	SUBPGNAME2	Causes all previous calls to be processed, then brings in called subprogram from System Library or Go file.			
Within control cards for a phase		CALLP	SUBP GNAME2	Causes all previous calls to be processed, then brings in called subprogram from System Library or Go file and adds patches to called subprogram.	Patch cards must immediately follow CALLP card. (See "Program Patching" section for patch card formats.)		
Before subprogram affected is processed or, within a subprogram, after TITLE card, but before any cards re- ferring to common data		BASE2 BASE2 BASE2	LINKSYMBOL ABCD/ 15000	Sets upper limit of common data area to address for the linkage symbol or to absolute address in operand.	This downward relocation factor applies to all succeeding subprograms during the current Linkage Loader run or until the next BASE2 card.		
area		BASE2		Sets upper limit of common data area to top of core.			
First card of sub- program		TITLE	SUBPGNAME2 XXXXX YYYYY NOTE: Leave five blanks be- tween XXXXX and YYYYY.	Begins a subprogram. Defines SUBPGNAME2 as a linkage symbol. YYYYY = Size of common data area subprogram will use. XXXXX = Origin for subprogram expressed as an ab- solute address.	NOTE: This TITLE card format is in the relocatable object deck produced by the language processors; it is not the same format as the Autocoder symbolic TITLE card.		

Figure 3. Control Cards for Linkage Loader (Page 1 of 2)

Location in		Format		Functions	Remarks
Control Card Deck	6 15	16 Card 20 Type	21 Operand(s)		
Within a subprogram	ABCD/ LINKSYMBOL	DEFIN DEFIN	LINKSYMBOL 15000	Defines the linkage symbol in columns 6 through 15 for use by the Linkage Loader as an entry point or data field within a subprogram.	Can be coded directly in the Autocoder language. See the Autocoder publication for full details.
After subprogram requested but be- fore the Linkage Loader symbol table is erased and before the end of the phase	IDENT	SNAP	SUBP GNAME2 XXXXXYYYYY ZZZZZYYYYYY ZZZZZSS NOTE 1: Do not put commas between operands. NOTE 2: ss must be in columns 56 and 57. Therefore, if only one area is to be snapped, leave blanks for the second yyyyyzzzzz.	Requests Snapshots of storage during execution of the subprogram. Causes all previous calls to be processed. IDENT is any one to five characters to identify the Snapshot listing. xxxxx = Relocatable address of instruction before which the Snapshot is to be taken. yyyyy = Relocatable address of lower limit of area to be snapped. zzzzz = Relocatable address of upper limit of area to be snapped. ss = Length of instruction at xxxxx; if the instruction length is less than ten, it should be written with a zero in column 56.	Cannot be used in a compile-and-go operation. Instruction before which Snapshot is to be taken must be at least seven characters long, cannot be chained, and cannot be in an IOCS macroinstruction. NOTE: When relocating the addresses on the SNAP card, the Linkage Loader will add the value for the subprogram name linkage symbol to the relocatable addresses in the SNAP card. Therefore, if the subprogram was origined at other than 00000, subtract the origin value from the compiled addresses to get the addresses to be used on the SNAP card.
Anywhere within control cards for Linkage Loader		INPUT INPUT	Mxx SIU	Directs Linkage Loader to read control cards from a unit other than the one being read.	Mxx is the name of a currently assigned symbolic unit, such as MR1. End of file on the substitute unit causes a return to the Standard Input Unit, as does an INPUT card with the operand SIU. (See the section "Assignment of Input/Output Units.")
Before CALLN, CALLP, PHASE, SNAP, or Monitor control cards or end of file on Standard or Alternate Input Unit		DISGO		Go file will not be searched for subprograms.	Effective during current Linkage Loader run or until next CONGO card.
Anywhere after DISGO card		CONGO		Cancels DISGO card function.	

PROGMNAME1 is the linkage symbol that is the program name defined by the PHASE card. SUBPGNAME2 is the linkage symbol that is the subprogram name defined by the TITLE card. LINKSYMBOL is any linkage symbol.

All three of these linkage symbols consist of one to ten characters that are left-justified, with trailing blanks in a ten-position field.

ABCD/ is any five-character linkage symbol.

Figure 3. Control Cards for Linkage Loader (Page 2 of 2)

the linkage symbol or, for a DCWF card, with a five-character constant of that address.

If the called subprogram has not yet been processed, the Linkage Loader places the subprogram name on its request list. (The same request list is used for CALL card requests.) When the Linkage Loader later locates and processes the subprograms on this list, it supplies the previous imbedded calls with the branch addresses or constants.

Note 1: Autocoder provides the DEFIN statement for establishing linkage symbols to represent any point in a subprogram. (The linkage symbol established by the TITLE card of a subprogram represents the origin point of that subprogram.) Linkage symbols established by the DEFIN statement can also appear in DCWF and DCWS cards; these cards are treated by the Linkage Loader in the same manner as DCWF and DCWS cards that contain a subprogram name. However, note that a subprogram is located and processed only if it has been called by the name contained in its TITLE card. Therefore, if subprogram ANYNAME contains an entry point represented by the linkage symbol ABCDEFGHIJ, the card DCWS ABCDEFGHIJ cannot be completely processed unless a call is made for ANYNAME, either by a CALL card or an imbedded call.

Note 2: The linkage symbols established within a particular phase of a multiphase program can be referred to by subprograms within the same phase and succeeding phases, but they cannot be referred to in preceding phases. For example, Phase 1 cannot contain a DCWS card with the linkage symbol work/ if that symbol is not established by a TITLE card or Definition card until Phase 2.

Organization of programs should be planned with special care regarding imbedded calls. Imbedded calls, if not ordered properly, could cause additional searches of the System Library file and, therefore, be time consuming.

The following rules apply to calling programs:

- 1. Rules of precedence for memory maps are:
 - a. The first subprograms in core storage will be those read from the Standard Input Unit (in the order read).
 - b. The second group of subprograms in core storage will normally be those read from the Go file (in the order read).
 - c. The third group of subprograms in core storage will normally be those read from the System Library file (in the order read).
 - d. If the subprograms in items b and/or c contain imbedded calls, then item c will be repeated until all calls are completed. The Go file is not searched again.
- 2. Imbedded calls are considered to be for sub-

programs in the System Library file. If the subprogram requested by an imbedded call is on the Go file, it must also be requested by a CALL card that is not imbedded in a subprogram.

Note: The call card may be omitted if the requested subprogram follows the requesting subprogram on the Go file.

3. The Go file is reset to its beginning and searched if any subprogram names are on the request list when the Linkage Loader reads a CALLN, CALLP, PHASE, SNAP, or Monitor control card, or finds the end of file on the Standard or Alternate Input Unit.

Note: The disco card can be used to prevent this search of the Go file.

- 4. Subprograms placed in the Standard or Alternate Input Unit do not require a CALL card; their TITLE cards serve this function.
- 5. If a subprogram on the Go file is to replace a subprogram with the same name on the System Library file, and the subprogram is requested by an imbedded call, a CALL card should be used to guarantee loading the subprogram from the Go file.

EXAMPLE 1

This example illustrates the procedure for directing the Linkage Loader to process a subprogram from the System Library file. The subprogram (sp1) constitutes a self-contained, single-phase program that has no imbedded calls for other subprograms.

Only one control card is necessary: CALL spl. The Linkage Loader will locate spl on the Library, process it, and place it on the Job file.

EXAMPLE 2

This example illustrates the procedure for directing the Linkage Loader to construct a single-phase program from three subprograms: spl, sp2, and sp3. (One of the three must be a primary subprogram; the other two can be primary or secondary.) In this example, spl is on the System Library file, sp2 is in the Standard Input Unit, and sp3 is on the Go file. None of the subprograms contain imbedded calls.

The sequence of cards in the Standard Input Unit is as follows:

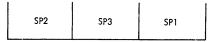
CALL SP1
TITLE card and relocatable object deck for SP2
CALL SP3

The Linkage Loader proceeds as follows:

- 1. CALL SP1: The name of SP1 is placed on the request list.
- 2. TITLE card and relocatable object deck for sp2: sp2 is processed and the name sp2 is added to the request list, with a flag indicating that sp2 has been processed.

3. CALL SP3: The name SP3 is added to the request list. The Linkage Loader resumes reading from the Standard Input Unit and finds no more Linkage Loader control cards. It begins searching the Go file, locates SP3, and processes it. Next, the Library is searched. SP1 is located and processed.

The memory map is shown below. The area between the outer vertical lines represents the area of core storage used for the subprograms processed.



EXAMPLE 3

This example illustrates the construction of a singlephase program from three subprograms, one of which is processed because of an imbedded call. sp1 is on the System Library file and contains an imbedded call for sp3; sp2 is on the Go file.

The sequence of cards in the Standard Input Unit is as follows:

CALL SP1

The Linkage Loader proceeds as follows:

- 1. CALL SP1: The name SP1 is placed on the request list.
- 2. CALL SP2: The name SP2 is placed on the request list. The Linkage Loader resumes reading from the Standard Input Unit and finds no more Linkage Loader control cards. It begins searching the Go file, locates SP2, and processes it. Next, the Library is searched. SP1 is located and processed. During the processing of SP1, the Linkage Loader finds the imbedded call for SP3. At that time, the name SP3 is placed on the request list. When the processing of SP1 is completed, the Linkage Loader checks the request list, finds the unsatisfied request, and resumes the search of the Library. SP3 is located and processed.

The memory map is as follows:



CALLN Card

The function of the CALLN card (Call Now) is essentially the same as the CALL card, except that the CALLN card directs the Linkage Loader to locate and process the specified subprogram immediately. This card can be used to control the memory map.

When the Linkage Loader reads a CALLN card, it does the following:

- 1. All subprograms specified by previous CALL cards are located and processed.
 - 2. Any imbedded calls resulting from processing

these subprograms are also processed (except imbedded calls for the subprogram named in the CALLN card).

- 3. Then the subprogram specified by the CALLN card is processed.
- 4. Any imbedded calls resulting from processing the subprogram specified by the CALLN card are added to the request list for subprograms (if the imbedded calls have not been satisfied by processing already performed).

The following rules apply to the use of the CALLN card:

- 1. Rules of precedence for memory maps:
 - a. All subprograms called before the subprogram named by the CALLN card will be the first subprograms in core storage. (The relative order of those subprograms is determined in accordance with the rules of precedence for the CALL card.)
 - b. The subprogram named by the CALLN card will be placed in core storage following the subprograms specified in item a, above.
- 2. The name of a subprogram on the Standard Input Unit should never be the operand of a CALLN card. If the preceding calls should be processed before a subprogram on the Standard Input Unit, the CALLN card should be used to call the last preceding subprogram.

EXAMPLE 1

This example illustrates the construction of a singlephase program from four subprograms that must be positioned in a certain order. Three subprograms are on the System Library file and the fourth is in the Standard Input Unit. None of them contain imbedded calls. The sequence of subprograms on the Library is sp1, sp2, sp3; sp3 must be positioned in storage after sp1 and before sp2 and sp4.

The sequence of cards in the Standard Input Unit is as follows:

CALL SP1
CALLN SP3
CALL SP2

TITLE card and relocatable object deck for sp4

The Linkage Loader proceeds as follows:

- 1. CALL SP1: The name SP1 is placed on the request list.
- 2. CALLN SP3: The name SP3 is stored, but not placed on the request list. The Linkage Loader begins searching for previously called subprograms. It locates SP1 on the Library and processes it. The Linkage Loader then determines that the request list has been satisfied. (SP1 was the only name on the list.) The name SP3 is moved to the request list. SP3 is then located and processed.

- 3. CALL SP2: The name SP2 is placed on the request list.
- 4. TITLE card and relocatable object deck for sp4: The Linkage Loader processes sp4. It resumes reading from the Standard Input Unit and finds no further Linkage Loader control cards. sp2 is then located and processed.

The memory map is as follows:

SP1	SP3	SP4	SP2
[

EXAMPLE 2

This example illustrates the construction of a singlephase program from three subprograms, one of which contains an imbedded call for a subprogram that must be placed between the other two subprograms.

The three subprograms are on the System Library file in the order sp1, sp2, sp3. sp3 must be positioned after sp1 and before sp2; sp1 contains an imbedded call for sp3.

The sequence of cards in the Standard Input Unit is as follows:

CALL SP1
CALL SP3
CALL SP2

The Linkage Loader proceeds as follows:

- 1. CALL SP1: The name SP1 is placed on the request list.
- 2. CALLN SP3: The name SP3 is stored but not placed on the request list. The Linkage Loader begins searching for previously called subprograms. It locates SP1 on the Library and processes it. During the processing of SP1, the Linkage Loader finds the imbedded call for SP3. The Linkage Loader will supply the imbedded call in SP1 with the address of SP3 when the TITLE card of SP3 is processed. The Linkage Loader then determines that there are no more names on the request list, moves SP3 to the request list, locates SP3, and processes it.
- 3. CALL SP2: The name SP2 is placed on the request list. The Linkage Loader resumes reading from the Standard Input Unit and finds no more Linkage Loader control cards. SP2 is then located and processed.

The memory map is as follows:



CALLP Card

The CALLP card (Call and Patch) functions in the same manner as the CALLN card. The CALLP card also directs the Linkage Loader to incorporate patches into the specified subprogram. The patches, in relocatable

machine-language form, must immediately follow the CALLP card in the Standard Input Unit. (See the section, "Program Patching," for patch card formats.)

PHASE Card

PHASE cards are used to create a multiphase program and to assign a name to that program. Each time the Linkage Loader encounters a PHASE card it performs the following functions:

- 1. All previous calls (from both control cards and imbedded calls) are processed. The Linkage Loader gives the resulting phase a termination record, which indicates the entry point of that phase if it included a primary subprogram. (This information is used by the Load routine of the Resident Monitor to initiate execution of the phase after it has been loaded into core storage.)
- 2. The Linkage Loader creates a header record for the next phase, consisting of the name and number of the phase.

The first Phase card for a multiphase program must specify the name of that program. Succeeding Phase cards for the same program must have blank operand fields. For the first Phase card, the Linkage Loader generates a header record, containing the name specified in the operand, and a phase number of 001. For succeeding Phase cards of the same multiphase program, the Linkage Loader generates a header record, containing the name specified by the first Phase card, and a phase number determined by incrementing a counter each time a Phase card with a blank operand is encountered. Therefore, the first Phase card with a blank operand indicates the beginning of the program's second phase, and the header record for this phase contains the phase number 002.

The programmer can specify the phase number, for any phases after the first phase, by putting a three-digit phase number (other than 999) in columns 6 through 8 of a blank-operand phase card. This phase number must be higher than the last phase number assigned; otherwise the Linkage Loader will ignore the user-assigned number and will assign the next sequential number to the phase. Following a user-assigned phase number, the next blank-operand phase card without a user-assigned number will be assigned a phase number one higher than the user-assigned phase number.

The following rules apply to the use of the PHASE

- 1. A PHASE card must immediately precede each group of Call cards (CALL, CALLN, CALLP) that specify the subprograms to be included in that phase.
- 2. If the first PHASE card is omitted, the name of the first subprogram that is processed is used by the Link-

age Loader as the name of the entire multiphase program. The header record of each phase will have this name. A single-phase program that consists of more than one subprogram should have a phase card if the program is relocated and executed in the same job. Because the subprograms are not necessarily processed in the order they are called, the phase card is needed to set the program name for the Monitor control card that executes the program.

- 3. A PHASE card with a program name in the operand field causes the Linkage Loader to reset the relocation factor to Base Zero and to erase from the Linkage Loader's symbol table any linkage symbols left from processing previous subprograms. (This erasure of symbols can be controlled by use of the PRTCT card, which is described later.)
- 4. A PHASE card with a blank operand field does not reset the relocation factor, nor does it cause the erasure of any symbols from the symbol table.
- 5. Columns 60 through 70 of the Phase card must be left blank. (These columns are used by the Linkage Loader. Their contents are defined in the *System Generation* publication.)

EXAMPLE

Because the PHASE card is most commonly used with the BASEL card, examples illustrating PHASE card use follow the BASEL card description.

BASE1 Card

The BASE1 card is used to control the relocation factor. This control of the relocation factor enables the programmer to direct the Linkage Loader to relocate a phase in such a manner that it will overlay the preceding phase at a predetermined point.

Three types of operands can be used in a BASE1 card:

1. Linkage Symbol: If a linkage symbol appears in the operand of a BASEL card, the Linkage Loader sets the relocation factor to the address equated with that symbol in the Linkage Loader's symbol table. For this reason, the linkage symbol must be defined by a subprogram processed before the one that includes the BASEL card. In most cases, the linkage symbol is declared by the TITLE card of a previous subprogram. Using a subprogram name as a BASEL operand enables one phase of a program to be relocated to the origin point of a subprogram in a previous phase even though, at the time the BASEl card was coded, the actual origin point of the previous subprogram had not yet been determined. When the phase that included the BASE1 card is loaded into core storage, it will overlay the previous phase, beginning at the origin point of the subprogram named in the operand of the BASE1 card.

Note: Because Autocoder provides the DEFIN state-

ment defining linkage symbols other than the origin point of a subprogram, Autocoder users can construct phases that overlay preceding phases at locations other than the beginning of a subprogram.

- 2. Actual Address: If the operand of a BASEl card contains an actual address, the Linkage Loader sets the relocation factor to that address. Care must be exercised in the use of this operand, because the final location of program elements that preceded a BASEl card containing an actual address cannot always be predicted.
- 3. *+X00: If the operand of a basel card contains *+X00 (asterisk-plus sign-X-zero-zero), the Linkage Loader sets the current relocation factor to the next highest address that is a multiple of 100. (If the relocation factor is already a multiple of 100, no adjustment is performed.) This operand is not necessarily related to a multiphase program. It is provided to control the relocation of subprograms that depend on certain areas being located at an even-hundred address. (For example, use of the Autocoder Clear Storage instruction can result in this requirement.)

Each time the Linkage Loader encounters a BASE1 card, it performs the following functions:

- 1. It sets the relocation factor to the address value specified by the operand of the BASEl card.
- 2. It erases from the symbol table all linkage symbols equated to addresses higher than the address value specified by the operand of the BASE1 card. (This erasure occurs only for BASE1 cards with a symbol or actual address in the operand. The *+X00 operand does not cause symbols to be erased.) The erasure of symbols can be controlled by the PRTCT card, which is described later.

The following rules apply to the use of the BASEL card:

- 1. A BASEL card, when placed in the Standard or Alternate Input Unit, should immediately follow a Phase card. A BASEL card will affect all subprograms that have not been processed, even if the calls for those subprograms preceded the BASEL card. (A PRICT card can be inserted between the Phase card and BASEL card.)
- 2. If a BASE1 card appears in a subprogram, it must appear immediately after the TITLE card. This location is required for subprograms that are on the System Library file and are to be relocated to the same point each time.
- 3. The address value of a linkage symbol used as the operand of a BASE1 card must have been defined during processing of a previous subprogram.

EXAMPLE 1

This example illustrates the construction of a twophase program from two subprograms. spl constitutes the first phase of the multiphase program, and sp2 constitutes the second phase, which is to overlay the first completely. Both subprograms are on the System Library file. Neither of the subprograms contains imbedded calls.

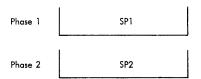
The sequence of cards in the Standard Input Unit is as follows:

PHASE SAM
CALL SP1
PHASE
BASE1 SP1
CALL SP2

The Linkage Loader proceeds as follows:

- 1. PHASE SAM: A header record is created. It contains the name SAM and the phase number 001.
- 2. CALL SP1: The name SP1 is placed on the request list.
- 3. Phase (blank operand): All previous calls are processed. (In this example, sp1 is the only previous call.) A termination record is generated for the first phase; a header record is generated for the second phase. It contains the name sam and the phase number 002.
- 4. BASE1 SP1: The relocation actor, which is not affected by a blank-operand PHASE card, is set to the address value esablished by the TITLE card of SP1. (All linkage symbols with an address equal to or higher than the TITLE card address are erased from the symbol table.)
- 5. CALL SP2: The name SP2 is placed on the request list. The Linkage Loader resumes reading from the Standard Input Unit and finds no more Linkage Loader control cards. SP2 is located and processed.

The memory maps are as follows:



EXAMPLE 2

This example illustrates the construction of a two-phase program from five subprograms. spl and sp2 constitute the first phase and sp3, sp4, and sp5 constitute the second phase, which is to overlay only sp2 of the first phase (leaving spl in storage). All the subprograms are on the System Library file. None of them contain imbedded calls.

The sequence of cards in the Standard Input Unit is as follows:

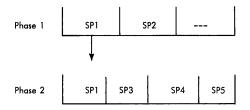
PHASE	MATILDA
CALL	sp1
CALLN	sp2
PHASE	

BASE1	sp2
CALL	sp3
CALL	sp4
CALL	sp5

The Linkage Loader proceeds as follows:

- 1. PHASE MATILDA: A header record is created. It contains the name MATILDA and the phase number 001.
- 2. CALL sp1: The name sp1 is placed on the request list.
- 3. CALLN SP2: In accordance with the procedure for CALLN cards, SP1 is located and processed, then SP2.
- 4. PHASE (blank operand): A termination record is generated for the first phase. It contains the entry point specified by the Termination card of whichever subprogram was the primary one. A header record is generated for the second phase. It contains the name MATILDA and the phase number 002.
- 5. BASE1 SP2: The relocation factor is set to the address value established by the TITLE card of SP2. (All linkage symbols with an address value equal to or higher than the TITLE card address are erased from the symbol table.)
- 6. CALL SP3: The name SP3 is placed on the request list.
- 7. CALL SP4: The name SP4 is placed on the request list.
- 8. CALL SP5: The name SP5 is placed on the request list. The Linkage Loader resumes reading from the Standard Input Unit and finds no further Linkage Loader control cards. SP3, SP4, and SP5 are located on the Library and processed.

The memory maps are as follows:



EXAMPLE 3

This example illustrates the construction of two separate multiphase programs during a single run of the Linkage Loader. All subprograms are on the System Library file, and none of them contain imbedded calls.

The sequence of cards in the Standard Input Unit is as follows:

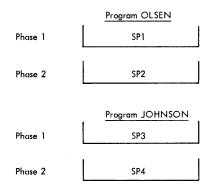
PHASE	OLSEN
CALL	sp1
PHASE	
BASE1	sp1
CALL	sp2
PHASE	JOHNSON

CALL SP3
PHASE
BASE1 SP3
CALL SP4

The Linkage Loader proceeds as follows:

- 1. PHASE OLSEN through CALL SP2: The Linkage Loader begins construction of a two-phase program in the manner illustrated by the preceding examples.
- 2. PHASE JOHNSON: The request list is checked for outstanding calls. The Linkage Loader finds the name sp2 on that list (from the CALL sp2 card), locates and processes sp2, and generates a termination record for the preceding phase. The relocation factor is then reset to Base Zero, and linkage symbols are erased from the symbol table. A header record is generated for the next phase. It contains the name JOHNSON and the phase number 001.
- 3. CALL SP3 through CALL SP4: The Linkage Loader constructs the second multiphase program.

The memory maps are as follows:



Note: The two programs are not necessarily related in any way. This example is designed to illustrate a means of directing the Linkage Loader to produce a Job file that is, in effect, a library of programs in absolute format.

PRTCT Card

The PRICT card is used to limit the erasure of linkage symbols from the symbol table. The Linkage Loader will retain in its symbol table all linkage symbols with addresses equal to or higher than the address value specified by the operand of the PRICT card. The operand of a PRICT card can be either a linkage symbol or an absolute address. This protection will be retained until it is changed by another PRICT card or by the initialization performed each time the Linkage Loader is brought into storage for execution.

The following rules apply to the use of the PRTCT card:

1. The PRTCT card must precede any control card that would cause erasure of symbols higher than the

address value specified by the operand of the PRTCT card.

2. The address value of a linkage symbol used as the operand of a PRTCT card must have been defined during the processing of a previous subprogram in the same execution of the Linkage Loader.

EXAMPLE

This example illustrates construction of a two-phase program in which one subprogram must be placed in upper storage and be used by other subprograms in both phases. All of the subprograms are on the System Library file, and none of them contain imbedded calls.

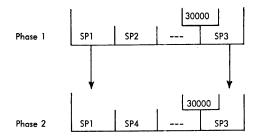
The sequence of cards in the Standard Input Unit is as follows:

PHASE	OTHELLO
CALL	sp1
CALLN	sp2
BASE1	30000
CALL	sp3
PHASE	
PRTCT	30000
BASEI	sp2
CALL	$\mathbf{sp4}$

The Linkage Loader proceeds as follows:

- 1. PHASE OTHELLO through CALLN SP2: SP1 and SP2 are processed as explained in previous examples.
- 2. BASE 1 30000: The relocation factor is set to 30000. (Note that all previous calls have been satisfied because of the CALLN card. Note also that if sp2 had contained an imbedded call for a subprogram that had not yet been processed, the new relocation factor would be applied to that subprogram.)
- 3. CALL SP3: The name SP3 is placed on the request list
- 4. Phase (blank operand): sp3 is located and processed. A termination record is generated for the first phase. A header record is generated for the second phase. It contains the name othello and the phase number 002.
- 5. PRICT 30000: The Linkage Loader notes that linkage symbols equated to an address value of 30000 and higher are not to be erased from the symbol table. (The operand of the PRICT card could also be sp3.)
- 6. BASE1 SP2: The relocation factor is reset to the address value established by the TITLE card of SP2. Linkage symbols equated to an address value higher than SP2 and lower than 30000 are erased from the symbol table.
- 7. CALL SP4: SP4 is placed on the request list. The Linkage Loader resumes reading from the Standard Input Unit and finds no more Linkage Loader control cards. SP4 is located and processed.

The memory maps are as follows:



BASE2 Card

The operand of the BASE2 card, which can be either a linkage symbol or an absolute address, sets the upper limit of a common data area.

Each time the Linkage Loader encounters a BASE2 card, it sets the downward relocation factor. This factor is then used for the adjustment of addresses that refer to a common data area.

The following rules apply to use of the BASE2 card:

- 1. The BASE2 card must be positioned in the Standard Input Unit so that it is read before the processing of any subprograms affected by it. The BASE2 card can also be placed within a subprogram, but before any references to a common data area.
- 2. More than one BASE2 card can be used, but rule 1 also applies to the additional cards.
- 3. If a linkage symbol is used for the operand of a BASE2 card, the address value of that symbol must have been defined during the processing of a previous subprogram.
- 4. The BASE2 limit will apply for all succeeding subprograms processed in the Linkage Loader run, unless changed by another BASE2 card.

EXAMPLE

This example illustrates the construction of a singlephase program from three subprograms, each of which refers to a common data area. All three subprograms are on the System Library file, and none of them contain imbedded calls.

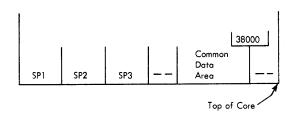
The sequence of cards in the Standard Input Unit is as follows:

BASE2 38000
CALL SP1
CALL SP2
CALL SP3

The Linkage Loader proceeds as follows:

- 1. BASE2 38000: The factor for downward relocation is set to 38000.
- 2. CALL SP1 through CALL SP3: These three subprograms are located and processed. During processing, all references to the common data area are adjusted in accordance with the factor set by the BASE2 card.

The memory map is as follows:



SNAP Card

The SNAP card directs the Linkage Loader to include the Snapshot program in a specified subprogram. (The Snapshot program is one of the Utility programs provided in the Operating System. See the *Utility Program* publication for a description of the Snapshot program.)

The operands used in this card are as follows: SUBPRGNAMEXXXXXYYYYYZZZZZYYYYYZZZZZSS

Where:

SUBPRGNAME is the one- to ten-character subprogram name, left-justified, with trailing blanks in a tenposition field.

xxxxx is the address of the instruction immediately before which a Snapshot is to be taken.

yyyyy is the address of the lower limit of an area for which a Snapshot is to be taken. This address, if not an even hundred, will be changed to an even hundred by the Snapshot program, which replaces the units and ten positions with zeros.

zzzzz is the address of the upper limit of an area for which a Snapshot is to be taken. This address, if not an address ending in 99, will be changed by the Snapshot program, which replaces the units and tens positions with 99.

ss is the length of the instruction at location xxxxx. If the instruction length is less than 10, it should be written as a two-digit number with a leading zero, e.g., 07.

NOTE 1: ss must be in card columns 56 and 57. Therefore, if only one area is to be snapped, leave blanks for the second yyyyyzzzzz.

Note 2: Do not put commas between operands.

The addresses used as operands are the relocatable addresses for a program compiled from 00000 upward. Normally these addresses can be obtained from the listing of the compilation. The Linkage Loader will add the value equated to the subprogram name linkage symbol to the addresses specified in the operand field of the SNAP card. However, if the subprogram was origined at other than 00000, the user must subtract the origin value in the Autocoder ord card or must subtract 1 for a fortran program from the compiled addresses to get the addresses to be used on the SNAP card.

Columns 6 through 10 of the SNAP card can be used

to establish a one- to five-character identification for the Snapshot(s) specified by the card. Any characters in those columns will be printed on the Snapshot listing.

The instruction at the point where the Snapshot is to be taken must be at least seven characters long, cannot be chained, and cannot be part of an rocs macro-instruction.

The following rules apply to use of the SNAP card:

- 1. A SNAP card must be positioned in the Standard Input Unit after the subprogram that it affects, or the call for that subprogram, but before the subprogram's name is erased from the Linkage Loader symbol table and before the end of the phase.
- 2. Any number of sNAP cards can be submitted between PHASE cards.
- 3. A SNAP card can contain either one or two sets of upper and lower limits for Snapshot areas.
- 4. A SNAP card forces all pending calls for sub-programs.
- 5. The Snapshot program must be in the Resident Monitor or in the library designated as the System Library file for this Linkage Loader run.

The SNAP card can be used only with a program that has been compiled. For a compile-and-go operation requiring Snapshot listings, use a Snapshot calling sequence in the source program. This sequence is given in the *Utility Programs* publication.

The Linkage Loader will replace the instruction before which the Snapshot is to be taken with a branch to a sequence that links to the Snapshot routine. The replaced instruction will appear in the linkage sequence, which is placed after the named subprogram.

INPUT Card

The INPUT card can be used to direct the Linkage Loader to read its control cards and relocatable object decks from a source other than the Standard Input Unit. The operand of the INPUT card is the name of the symbolic unit to be substituted for the Standard Input Unit.

The following rules apply to use of the INPUT card:

- 1. The name of the substituted symbolic unit must be one that is currently assigned to a physical unit. (See the section, "Assignment of Input/Output Units.")
- 2. End of file at the substitute unit causes the Linkage Loader to resume reading from the Standard Input Unit.
- 3. INPUT cards can be placed at any point in the control card deck for the Linkage Loader.
- 4. The INPUT card can contain the operand: SIU. In this case, the Linkage Loader resumes reading from the Standard Input Unit. Such a card would be used in a substitute unit, after an INPUT card in the Standard

Input Unit had directed the Linkage Loader to begin reading from that substitute unit.

Note: The term *Standard Input Unit*, as used in the above description, refers to the unit currently used by the Monitor as its source of control information. This source could be either the Standard or Alternate Input Unit.

TITLE Card

The TITLE card is the first card of each subprogram. It defines the name of the subprogram as a linkage symbol. It is compiled by the Autocoder, FORTRAN, and COBOL language processors. The format for the TITLE card is:

TITLE SUBPRGNAMEYYYYY XXXXX

Where SUBPGRNAME is the one- to ten-character subprogram name. (If less than ten characters, it is left-justified with trailing blanks in a ten-position field.)

yyyyy is the base to which the subprogram was compiled. This is normally 00000 or blank for Autocoder and cobol programs. (If blank 00000 is assumed by the Linkage Loader.) It is 00001 for FORTRAN programs.

xxxxx is the total number of core locations in the common data area required by *this* subprogram.

Note: Five blanks separate yyyyy and xxxxx.

The value expressed by yyyyy, which is the low origin point for the subprogram, is subtracted from the current upward relocation value to produce the relocation factor for this subprogram.

The value expressed by xxxxx, the common data area size, is added to each relocated load address to determine if core storage has been exceeded. If storage is exceeded, the Linkage Loader prints the message core exceeded on the Standard Print Unit. This is only a warning message; the Linkage Loader will put the subprogram that exceeded storage on the Job file and, if requested by an exeq card, the subprogram will be executed.

After the Linkage Loader reads a TITLE card, it will read the next card in order to determine if it is a BASEL card. If so, the BASEL card will be processed before the TITLE card

The TITLE cards generated by the COBOL and FORTRAN processors contain additional operands, which are not used by the Linkage Loader.

DISGO Card

The pisco card specifies that the Go file is not to be searched for called subprograms. This card should be used to save search time when all called subprograms are on the System Library file.

After the Linkage Loader reads the DISCO card, it will not search the Go file (1) for the rest of the current Linkage Loader run, or (2) until a congo card is read.

The DISCO card is needed because the CALLN, CALLP, PHASE, SNAP, and Monitor control cards, and the end of file on the Standard or Alternate Input Unit, cause the Linkage Loader to process all calls on its request list. For this processing, the Linkage Loader rewinds the Go file and searches it from the beginning.

When a disco card is in effect, the Go file must

remain mounted, ready, and assigned by an ASGN card. (For details about the ASGN card, see the "Assignment of Input/Output Units" section.)

CONGO Card

The congo card cancels the effect of the DISGO card. Thus, in the next search after the congo card, the Go file will be searched.

The congo card is required only if a disco card has been used because, if a disco card is not used, the Go file is searched once in each search.

Use of Resident Monitor Functions

Resident Monitor functions can be used through linkage sequences and system symbol references coded into a source program.

The linkage sequences are branch instructions and, usually, one or more fields of control information. The branch instruction gives control to a routine in the Resident Monitor. The linkage sequences are listed in Figure 4.

Macro-instructions to generate most of the linkage sequences are included in the Autocoder Macro Library. These macro-instructions are given in this section after the corresponding linkage sequence descriptions.

The system symbols refer to areas of the Communication Region, so that all dependent programs can use certain information stored in the Resident Monitor. System symbols referring to areas in the Communication Region are listed in Figure 5.

Use of Communication Region

Contents of the Communication Region

Information in the Communication Region can be used by any dependent program at any time. However, the information can be modified only through the linkage sequence described below.

Some Communication Region fields contain data; others contain an address constant that gives the location of an area or data.

Each area of the Communication Region contains a word mark in its high-order position only.

Modification of the Communication Region

gram)

To modify a field of the Communication Region, the dependent program must contain a linkage sequence constructed as follows:

BXPA /MCR/ /xxx/ DCW (low-order address of information to be DCW placed in Communication Region field (address of dependent program routine DCW to be given control if field cannot be modified) (Next sequential instruction in dependent pro-

An example of the need for a dependent program to modify a Communication Region field is modification of the /IPI/ field during one program to set a switch for a later program in the same job. The following Autocoder macro-instruction can be used to generate the linkage sequence for modification of a Communication Region field.

ANYLABEL MONOP MODIFY,XXX,YYYYY,ZZZZZ must appear as shown.

MODIFY operand:

the system symbol, without slashes, xxx operand: for the Communication Region

field to be modified.

the low-order address of the inforyyyyy operand: mation to be placed in the Communication Region field.

zzzzz operand:

optional. This is the address to which control is to be given if the Communication Region field cannot be modified. If this operand is omitted, control will be returned to the next sequential instruction after the macro-instruction.

Use of Unit-Record Routines

The three unit-record routines in the Resident Monitor can be used by dependent programs. One of these routines reads from the Standard or Alternate Input Unit, another writes on the Standard Print Unit, and the third writes on the Standard Punch Unit.

All three routines perform their functions with unblocked records, in Move mode. If tape is used for any of the routines, the unit-record functions are performed in odd parity, and the records are unblocked.

Read Routine

The Read routine reads 80-character records from the Standard or Alternate Input Unit. (In constructing the linkage sequence to the Read routine, the programmer makes no distinction between reading from the Standard or Alternate Input Unit.)

The high-order address of the input area last filled by the Read routine is placed in the /CRD/ field in the Communication Region. /CRD/ must be used to retrieve information from an input area. The input areas are in the Resident Monitor. Word marks must not be set or cleared in the input areas.

Resident Monitor Functions	Linkage Sequence in Autocoder Language	Remarks
Read Routine: Reads 80-character records from Standard Input Unit and Alternate Input Unit	B /RSI/ B (address of end-of-file routine in dependent program) BBE USERERROR,/MCS/,n NSI Where the contents of n must be selected by the user.	/MCS/ contains the Channel Status Character constructed by the Resident IOCS from any machine indicators turned on by read errors. The Channel Status Character is described in the Basic Input/Output Control System publication.
Print Routine: Writes 132–character records on Standard Print Unit	One output area: BXPA /PRT/ DCW #5 DCW (high-order address of output area) NSI Two output areas: BXPA /PRT/ DCW #5 DCW (high-order address of output area) DC (high-order address of other output area) NSI	Output areas defined by dependent program, 134 positions long 1. First position. Carriage-control character. (See Figure 6.) 2. Positions 2 through 133. Record to be written on Standard Print Unit. (If tape, first character is also written.) 3. Position 134. Group mark with word mark.
Punch Routine: Punches 80–character records on Standard Punch Unit	One output area: BXPA /PCH/ DCW #5 DCW (high-order address of output area) NSI Two output areas: BXPA /PCH/ DCW #5 DCW (high-order address of one output area) DC (high-order address of order output area) NSI	Output areas defined by dependent program, 81 positions long: 1. Positions 1 through 80. Record to be punched by Standard Punch Unit. 2. Position 81. Group mark with word mark.
Console Inquiry Routine: Clears /MCI/ to blank, informs operator of need for information, enters waiting loop until operator has typed input, and returns control to NSI.	BXPA /MCR/ DCW /MCI/ DCW (address of a blank) DCW (address of dependent pro- gram's error routine) IOCTL TYPE, MESSA GE BCE *-11, /MCI/ NSI	High-order position of input message area is designated by /RIQ/. Operator identifies input message with \$3x. x is placed in /MCI/. MESSAGE is the label for the console printer message, coded in the dependent program, that informs the operator of the need for the \$3x message.
Wait-Loop Routine	B /WAT/ NSI	Console inquiry message, 550, causes the Resident Monitor to exit from the Wait-Loop routine and return control to NSI.
Load Routine: Locates and loads a phase on the Job file or System Operating File (as specified on the EXEQ card for the program).	Locate, load, and execute phase: B /LOD/ DCW +nnn Locate and load phase: B /LOD/ DCW -nnn NSI nnn = phase number of a phase of the program being executed.	
Subroutines of Load Routine: 1. Backspace System Tape: Backspaces records on the Job file or System Operating File (as specified on the EXEQ card for the program).	ZA (number of records to be back- spaced, minus one),X13 B /ZBS/ NSI	
 Read System Tape: Reads one record from the Job file or System Operating File (as specified on the EXEQ card for the program). 	BXPA /ZRR/ DCW (@L@ for Load mode or @M@ for Move mode) DCW (input area address) BBE USSERERROR,/ZRS/,n NSI Where the contents of n must be selected	The fourth instruction is optional. It will give control to a dependent program error routine (USERERROR). /ZRS/ contains the Channel Status Character constructed by the Resident IOCS from any machine indicators turned on by read errors.
 Rewind System Tape: Closes and opens, with rewind and label checking if applicable, the Job file or System Operating File (as specified on the EXEQ card for the program). 	by the user. B /ZRW/ NSI	
 Simple Search of System Tape: Locates the requested phase of the program currently being executed. 	B /ZSS/ DCW (three-position unsigned phase number) DCW (address of dependent program routine to be given control if phase is not found).	

Figure 4. Linkage Sequences for Use of Resident Monitor Functions (Page 1 of 2)

Resident Monitor Functions	Linkage Sequence in Autocoder Language	Remarks
 Read System Disk: Reads one record in load mode from the Job file or System Operating File (as specified on the EXEQ card for the program) into an input area. 	Read Specific Track: BPXA /ZRR/ DCW @N@ BBE USERERROR,/ZRS/,n NSI Read Next Track: BXPA /ZRR/ DCW (four-character track address) BBE USERERROR,/ZRS/,n NSI Where the contents of n must be selected by the user.	Input area defined by Resident Monitor: High-order position is /MBF/. Low-order data position (one position left of terminating group mark with word mark) is /MBX/. USERERROR is the label of an error routine in the dependent program. /ZRS/ contains the Channel Status Character constructed by the Resident IOCS from any machine indicators turned on by read errors.
End-of-Program Routine	Normal End of Program: B /BOP/ Unusual End of Program: B /UEP/ Special End of Program: BXPA /MCR/ DCW /CGO/ DCW (address of an A bit) DCW (address of dependent program routine to be given control if field cannot be modified) NSI NOTE: NSI is usually B /EOP/.	
Modification of Communication Region Routine	BXPA /MCR/ DCW /XXX/ DCW (low-order address of information to be placed in field /XXX/) DCW (address of dependent program routine to be given control if field cannot be modified)	

 $\ensuremath{\mathsf{NSI}}$ is the next sequential instruction in the dependent program.

Figure 4. Linkage Sequences for Use of Resident Monitor Functions (Page 2 of 2)

System Symbol	Contents of Field	Contents Type	Length	Modi- fiable	System Symbol	Contents of Field	Contents Type	Length	Modi- fiable
/AMS/	Actual machine size minus two (highest lo- cation that can be loaded by the Load routine). For example, for a machine with	Address Constant,		No	/PHN/	Phase number of program phase currently be- ing executed. (This area is set to 001 at each Monitor EXEQ card.)	Data	3	No
	80,000 positions of core storage, this field would contain 79997. This field is filled in at System Generation.				/PNM/	Name of program currently being executed. (The name is left-justified in the area.)	Data	10	Yes
/CRD/	Contains the high-order address of the input area last filled by the Resident Monitor's Read routine.	Address Constant	5	No	/SIZ/	Highest location loaded for the current dependent program (including any previous phases of it). This field is set by the Resident Monitor's Load routine.	Address Constant	5	Yes
/DAT/	Date (first two positions for the year, followed by three positions for the day). This area is set by a Monitor control card during initialization of the Resident Monitor. (See the section, "Use of Transitional Monitor.")	Data	5	No	/TPB/	Lowest location of the area reserved for programs under control of the Tele-Processing Supervisor. (Tele-Processing Systems are described in the Tele-Processing Supervisor publication.) This	Address Constant	5	No
/IPI/	Interprogram information. (This area can	used to store control information be— ten runs within a job. For example, a gram can set switches in this area that the tested by the next program exe—		location is also the top of the Resident Monitor minus the Initialization routine.					
	tween runs within a job. For example, a program can set switches in this area that can be tested by the next program exe-				/MCI/	Set by console inquiry \$3x, where x is placed in /MCI/. It is set to blank by End-of-Program routine.	Data	1.	Yes
	cuted.) This area is set to blanks between each job.				/MGA/	Contains track address of last track on disk Go file.	Disk Address	4	Yes
/LIN/	Number of lines per page for printer out- put. This field is set at System Genera- tion. (This field can be reset by any program that wishes to establish a con- stant that can be compared to a program counter for its printer output. However, any program that resets this field must set it back to the original value before end of program.)	Data	2	Yes	/CGO/	Go and other indicators. The A bit ON indicates that the contents of the Go and Job files are not valid. The B bit ON indicates that a language processor should put its output on the Go file. The B bit OFF indicates that the Go file contents are not valid. These indicators are set by the language processors, Linkage Loader, Monitor	Data	1	Yes
/ORG/	Contains lowest location of storage area available to dependent programs. (This field, which is set at System Generation, is used by the Linkage Loader to determine the value of Base Zero.) This location is at the top of the Resident Monitor minus the Initialization routine or at the top of the area reserved for programs under control of the Tele-Processing Supervisor, if such an area exists.	Address Constant	5	No			MODE control card, and dependent programs. Both bits are turned OFF between jobs.		

Figure 5. System Symbols for Communication Region Fields

The Read routine checks each input record to determine whether it is a Monitor control card. If it is, the routine branches to the dependent program's end-of-file address specified in the linkage sequence. Therefore, input records for a dependent program must not contain MON\$\$ in columns 6 through 10.

The Resident 10cs checks for all read errors.

The linkage sequence for use of the Read routine is as follows:

B /RSI/

B (address of end-of-file routine in dependent program)

BBE USERERROR,/MCS/,n

(Next sequential instruction in the dependent program)

If the Read routine signals an end-of-file condition, it returns control to the branch instruction containing the address of the dependent program's end-of-file routine. Otherwise, control is returned to the third instruction in the linkage sequence. (The third instruction, as shown, is optional.)

A program using the Read routine can also check for read errors by including the third instruction, as shown. This instruction interrogates /MCS/. /MCS/ contains the Channel Status Character constructed by the Resident 10CS from any machine indicators turned on by read errors. The Channel Status Character is described in the Basic Input/Output Control System publication.

USERERROR in the third instruction is the address to which control is to be given if an error occurred. The contents of n, which must be selected by the user, will be compared to /MCS/ to determine if control should be given to USERERROR.

The following Autocoder macro-instruction can be used to generate the linkage sequence to the Resident Monitor's Read routine.

ANYLABEL STDIO READ, AREA, EOFADDR, USERERROR, n

READ operand:

must appear as shown.

AREA operand:

the high-order address of an 81position field, which includes a group mark with word mark at the end, into which the record is to be moved. This area must be defined by the dependent program.

EOFADDR operand:

the address in the dependent program to which control is to be given when end of file occurs.

usererror operand:

optional. The address in the dependent program to which control is to be given if an error occurs.

n operand:

optional. The character to which /MCS/ will be compared to determine if control should be given to USER-ERROR. The user must select the contents of n.

Print Routine

The Print routine writes 132-character records on the Standard Print Unit. The Print routine can be addressed by two types of linkage sequences. One requests a print operation using one-area logic; the other requests a print operation using two-area logic. Both linkage sequences include the address of the output area(s) containing the record(s) to be written on the Standard Print Unit.

Note: Only one record will be written per entry to this routine.

Each output area must be defined by the dependent program, and each must be 134 positions long. The first position of the area is used to specify the carriage-control character. (See Figure 6 for the carriage-control characters.) Positions 2 though 133 of the output area contain the record to be written on the Standard Print Unit. (If the Standard Print Unit is tape, the first position of the output area is also written) Position 134 of the output area must contain a group mark with word mark.

The two types of linkage sequences for using the Print routine are as follows.

One Output Area:

BXPA /PRT/ DCW #5

Ocw (high-order address of the output area)
(Next sequential instruction in the dependent program)

Two Output Areas:

BXPA /PRT/ DCW #5

(high-order address of one output area)

d	IMMEDIATE SKIP TO	d	SKIP AFTER PRINT TO	d SPACE
1	Channel 1	A	Channel 1	blank 1 Space
2	Channel 2	В	Channel 2	K 2 Spaces
3	Channel 3	С	Channel 3	L 3 Spaces
4	Channel 4	D	Channel 4	
5	Channel 5	E	Channel 5	
6	Channel 6	F	Channel 6	SPACE AFTER
7	Channel 7	G	Channel 7	PRINT
8	Channel 8	н	Channel 8	/ 1 Space
9	Channel 9	1	Channel 9	S 2 Spaces
0	Channel 10	?	Channel 10	T 3 Spaces
#	Channel 11	•	Channel 11	
@	Channel 12	П	Channel 12	

Figure 6. Carriage-Control Characters for Print Routine

ocw (high-order address of the other output area)

(Next sequential instruction in the dependent program)

Note: The DCW of five blanks (immediately below the branch instruction) is used for internal functions of the Print routine. This field must be included in every linkage sequence to the Print routine.

By testing for a word mark in the position immediately following the pow that contains the address of the output area, the Print routine determines which type of linkage sequence is being used. If a word mark is present, it is assumed to be the word mark over the operation code of the next instruction in the dependent program. If no word mark is present in that position, as with the pc, the Print routine assumes the presence of an address for a second output area.

Note: If only one output area is specified, the position following the address of the output area must contain a word mark.

If two output areas are specified, the Print routine first uses the output area originally specified by the DCW. After completion of each print operation, the routine rotates the two addresses in the linkage sequence. Before the first print operation, the linkage sequence would appear as follows:

BXPA /PRT/
DCW #5
DCW XXXXX
DC YYYYY

After the first print operation, the linkage sequence would appear as follows:

 BXPA
 /PRT/

 DCW
 #5

 DCW
 yyyyy

 DC
 xxxxx

The next time the Print routine is addressed by this linkage sequence, it will use the output area represented by yyyyy. After that print operation is completed, the two addresses are rotated again.

If the printed output is to start on a new page, the first data record must have a "1" as the carriage-control character. If the last data record of the printed output is to be the last line printed on a page, one additional record of blanks, with a "1" as the carriage-control character, should follow the last data record.

The Resident rocs checks for all print errors (or write-tape errors, if tape is used), except for wrong-length records.

The following Autocoder macro-instruction can be used to generate the linkage sequence to the Print routine:

ANYLABEL STDIO PRINT, AREA 1, AREA 2 PRINT operand: must appear as shown.

AREA1 operand: the high-order address of a 134-

position field from which the record is to be printed. The first position of the field specifies the carriage-control character (Figure 6). Positions 2 through 133 contain the record to be printed. Position 134 contains a group mark with word mark.

area2 operand:

optional. This operand specifies a second output area with the same format as AREA1.

Punch Routine

The information concerning the Print routine also applies to the Punch routine, with the following exceptions:

- 1. The Punch routine writes 80-character records on the Standard Punch Unit. The output area(s) for these records must be 81 positions long. (Position 81 must contain a group mark with word mark.)
- 2. The system symbol used in the BXPA instruction is /PCH/.

The following Autocoder macro-instruction can be used to generate the linkage sequence to the Punch routine:

ANYLABEL STDIO PUNCH, AREA1, AREA2

PUNCH operand: must appear as shown.

AREA1 operand: the high-order address of an 81-position field from which

the record is to be punched. Position 81 must contain a group mark with word mark. optional. This operand specifies a second output area with

the same format as AREA1.

The Resident rocs checks for all punch errors (or write-tape errors, if tape is used), except wrong-length record.

Use of Console Inquiry Routine

AREA2 operand:

Through the Console Inquiry routine, a dependent program can request input from the console printer, enter a waiting loop while the operator types in the information, and then retrieve the input.

The linkage sequence for console input is:

BXPA /MCR/
DCW /MCI/
DCW (address of a blank)
DCW (address of dependent program's error routine)
IOCTL TYPE,MESSAGE
BCE *-11,/MCI/

(Next sequential instruction in the dependent program. This instruction should move the console input from /RIQ/ to a field in the dependent program.)

MESSAGE is the label for a console printer message coded in the dependent program. This message is typed to request that the operator enter a \$3x message. Then the sequence enters a waiting loop that is broken when the \$3x message is entered.

The typed-in message begins with \$3x. The x is moved to a one-character field designated /MCI/. /MCI/ is cleared to blank by the Console Inquiry routine, before it requests the input, and by the End-of-Program routine.

The high-order position of the console input area is designated /RIQ/. The input area length is defined at System Generation; it can be five positions minimum to 20 positions maximum. In a Tele-Processing system, the input message area must be 20 positions long. Wrong-length-record errors are ignored by the Console Inquiry routine.

The input area is not cleared by the Console Inquiry routine. Because a following console message might overlay the \$3x message, the dependent program should immediately move the message from the input message area and compare the \$3x in the message to the x in /MCI/ to verify that the message retrieved is the correct one.

Use of Wait-Loop Routine

When a dependent program requires that processing be suspended until the machine operator performs a function (such as changing forms in the printer), the program can write a message on the console printer and then enter the Wait-Loop routine. When the operator has performed the function specified by the program's message, he breaks the waiting loop by means of console inquiry \$50. The Resident Monitor then returns control to the dependent program at the instruction immediately following the branch to the Wait-Loop routine.

The entry point to the Wait-Loop routine is represented by the system symbol /wat/. In Autocoder, the instruction B /wat/ is used to enter the routine. COBOL generates this branch from the stop (literal) statement. FORTRAN generates this branch from the PAUSE statement.

In a Tele-Processing system, the system can receive interrupts during the waiting loop.

Use of Load Routine

The Load routine in the Resident Monitor can be used by dependent programs to locate and load phases of a multiphase program.

The linkage sequence to request locating, loading, and execution of a specific phase is as follows:

B /LOD/ DCW +nnn

When the phase number, nnn, contains a plus sign over the units position, the Load routine initiates execution of the phase at its entry point.

The linkage sequence to request only locating and loading of a specific phase is as follows:

B /LOD/
DCW —nnn
(Next sequential instruction in dependent program)

When the phase number, nnn, contains a minus sign over the units position, the Load routine returns control to the next sequential instruction following the linkage sequence that issued the request for the phase.

The following considerations should be observed in using linkage sequences to request program phases:

- 1. Requests can be issued only for phases of the program being executed. (The name of the program being executed is contained in the /PNM/ field of the Communication Region.)
- 2. The number of the phase being executed is contained in the /PHN/ field of the Communication Region. Using this field, the dependent program can construct a request for a specific phase in relation to the phase being executed.

- 3. If the requested phase cannot be located, control is given to the Resident Monitor's End-of-Program routine at the unusual-end-of-program entry point. The requested phase must be on the System Operating File or Job file, as specified in the exeq card for the program. (Details concerning end-of-program procedures are in the topic, "Use of End-of-Program Routine." Details on the exeq card are in the section, "Use of Transitional Monitor.")
- 4. As a phase is brought into storage, it overlays all information in the previous phase that falls inside the new phase's lowest and highest loading addresses. Therefore, information to be communicated from one phase to the next must reside below the lowest address into which information is loaded for a succeeding phase, or above the highest address.

For a tape-oriented Resident Monitor, dependent programs can use subroutines in the Load routine to read, backspace, rewind, and search the Job file or System Operating File, whichever was specified in the Monitor exeq control card for the program. For a diskoriented Resident Monitor, dependent programs can use a subroutine in the Load routine to read the Job file or System Operating File, whichever was specified in the exeq card for the program.

Read System Tape Subroutine

This subroutine reads the next record from the Job file or System Operating File into the area designated in the linkage sequence. The input area must be defined by the dependent program. Wrong-length-record errors are not checked. If the subroutine detects the end of file, the file will be rewound and opened with label checking, if labels exist, and the first data record after the label will be read into the designated area.

The linkage sequence for the Read System Tape subroutine is:

The fourth instruction is optional. It will give control to a dependent program error routine labeled USERERROR if an error occurred. The user must select the contents of n to match the error conditions to be checked. /ZRS/ contains the Channel Status Character

constructed by the Resident 10cs from any machine indicators turned on by read errors.

program)

The following Autocoder macro-instruction can be used to generate the linkage sequence to the Read System Tape subroutine:

ANYLABEL SYSIO READ, X, AREA, USERERROR, n

READ operand:

must appear as shown.

x operand:

must be an L for Load mode or an M for Move

mode.

AREA operand:

the high-order address of the area into which the record is to be read. This area must be defined by the dependent program.

USERERROR operand: optional. This is the ad-

dress in the dependent program to which control is to be given if an error occurred.

n operand:

optional. This is the character used for comparison to determine if an error occurred. The user must select the contents of n.

Backspace System Tape Subroutine

This subroutine backspaces one or more records, as specified in the linkage sequence, on the Job file or System Operating File. The subroutine will backspace over tape labels, treating them as data records.

The linkage sequence for the Backspace System Tape subroutine is:

za (number of records to be backspaced, minus one), X13

B /ZBS/

(Next sequential instruction in the dependent program)

The following Autocoder macro-instruction can be used to generate the linkage sequence to the Backspace System Tape subroutine:

ANYLABEL SYSIO BSP,X

BSP operand: must appear as shown.

x operand:

the number of records to be back-

spaced, minus one.

Rewind System Tape Subroutine

This subroutine closes and opens, with rewind and label checking if applicable, the Job file or System Operating File, whichever was specified in the EXEQ card for the program.

The linkage sequence for the Rewind System Tape subroutine is:

2 /7237/

(Next sequential instruction in the dependent program)

The following Autocoder macro-instruction can be used to generate the linkage sequence to the Rewind System Tape subroutine:

ANYLABEL SYSIO RWD

RWD operand: must appear as shown.

Simple Search of System Tape Subroutine

This subroutine locates but does not load the requested phase of the program being executed. It will not search for phases of other programs. If the phase is not located, control is transferred to the not-found address specified in the linkage sequence.

The linkage sequence for the Simple Search of System Tape subroutine is:

B /ZSS/

DCW DCW (three-position unsigned phase number)
(address of dependent program routine
to be given control if phase is not
found)

(Next sequential instruction in dependent program)

The following Autocoder macro-instruction can be used to generate the linkage sequence to the Simple Search of System Tape subroutine:

ANYLABEL SYSIO SEARCH operand:

SEARCH,nnn,NOTFNADDR

nnn operand:

must appear as shown. the three-digit unsigned

number of the phase to be found.

NOTENADDR operand:

optional. This is the address in the dependent program to which control is to be given if the phase is not found. If omitted, control will be returned to the instruction after the macro-instruction.

Read System Disk Subroutine

This subroutine reads a record from the System Operating File or Job file in a disk-oriented Operating System. The subroutine reads in Load mode. The input area is defined by the Resident Monitor; the high-order position is /MBF/ and the low-order position (one position to the left of the terminating group mark with word mark) is /MBX/.

The linkage sequence for the Read System Disk subroutine for a specific track (to be read from the System Operating File or Job file, whichever is specified in the EXEQ card) is: BXPA /ZRR/

pcw (four-character track address)

BBE USERERROR,/ZRS/,n

(Next sequential instruction in dependent program)

The linkage sequence to read the next track from the System Operating File or Job file, whichever is specified in the exeq card, is:

bxpa /zrr/ dcw @n@

BBE USERERROR,/ZRS/,n

(Next sequential instruction in dependent program)

USERERROR is the label of an error routine in the dependent program. /ZRS/ contains the Channel Status Character constructed by the Resident 10cs from any machine indicators turned on by read errors. The user must select the contents of n to match the error conditions to be checked.

The following Autocoder macro-instruction can be used to generate the linkage sequence to the Read System Disk subroutine. The forms of the macro-instruction to read a specific or the next track are:

ANYLABEL SYSIO READ,,nnnn,USERERROR,n ANYLABEL SYSIO READ,,,USERERROR,n

READ operand:

must appear as shown.

Commas:

must appear as shown.

nnnn operand:

f present, this operand specifies the four-digit track address of the reccord to be read. If omitted, the next sequential track will be read. (Omission must be indicated by a comma, as shown in the second form above.)

USERERROR operand:

optional. This is the address in the dependent program to which control is to be given if an error occurred.

n operand:

optional. This is the character used for comparisons to determine if an error occurred. The user must select the contents of n. If a comma or commercial at sign (@) is used for this operand, write it as follows:

For a comma: @,@ For the at sign:@@

Use of End-of-Program Routine

Dependent programs can end execution in one of three ways: normal end of program, unusual end of program, or special end of program. The results of the three end-of-program types are summarized below.

END OF PROGRAM	NOT TEST MODE	TEST MODE
Normal entry point: /EOP/	Continue processing with the next Monitor control card.	Continue processing with the next Monitor control card.
Special entry point: /EOP/; A bit of /CGO/ ON	Skip to next JOB card.	Continue processing with the next Monitor control card. However, Linkage Loader will not be loaded and executed and programs on the Job file will not be executed, if requested.
Unusual entry point: /UEP/	Write contents of storage on Core Image file, if this file was specified at System Generation. Skip to next JOB card.	Write contents of storage on Core Image file, if this file was specified at System Generation. Continue process- ing with next Mon- itor control card.

The mode is determined by the MODE card (described in the section, "Use of Transitional Monitor").

Note: As the Transitional Monitor skips to the next JOB card, it will process Monitor control cards directed to the Tele-Processing system and will process a Monitor END card.

Normal End of Program

When a dependent program completes its required functions, it signals end of program by branching to the Resident Monitor at the entry point represented by the system symbol /EOP/. This linkage sequence in Autocoder language is B /EOP/. The Resident Monitor brings the Transitional Monitor into storage to read the Standard or Alternate Input Unit for the next Monitor control card.

Note: The next Monitor control card can be a control card indicating the beginning of the next job or requesting the performance of another run within the current job (such as the execution of a Utility program). Therefore, within one job, the Resident Monitor can receive any number of end-of-program indications.

Unusual End of Program

An unusual end of program can be signaled to the Resident Monitor in one of two ways: by the dependent

program, itself, or by the console inquiry, \$10, from the operator. The linkage sequence to the unusual-end-of-program portion of the End-of-Program routine is:

B /UEP/.

When the Resident Monitor receives an unusualend-of-program indication, it preserves the contents of storage on the Core Image file (if the installation includes such a file). The Resident Monitor then checks to determine if the dependent program is being tested, as indicated by a MODE card. If the TEST mode is not on, the Resident Monitor cancels any subsequent portions of the current job by bringing the Transitional Monitor into storage to read the Standard or Alternate Input Unit until it finds a Monitor control card that indicates: the beginning of the next job (job card), the end of the current batch (Monitor END card), or a card directed to the Tele-Processing system. Thus, if the program is not being tested, execution of remaining programs within this job will not be performed. As an example, the Core Image file will not be printed, even though a storage print was requested by a Monitor control card requesting execution of the Utility programs.

If the dependent program is being tested, the Transitional Monitor is brought into storage to read the Standard or Alternate Input Unit for the next Monitor control card. If, for example, a storage print is to be taken for the program, a Monitor control card that requests execution of the Utility programs would be placed at this point in the deck.

Note: Because core storage is written on the Core Image file only for an unusual end of program, and because the Core Image file serves as input to the Storage Print program of the Utility programs, a dependent program being tested should end with a branch to /UEP/, rather than /EOP/. This ensures that a Storage Print can always be obtained for a program being tested, whether it comes to a successful conclusion or not.

Special End of Program

The special end of program is the same as unusual end of program, except that the Go and Job file contents are indicated as invalid and the contents of storage are not written on the Core Image file. Because the Go and Job file contents are indicated as invalid, the Linkage Loader and any program to be loaded from the Job file will not be executed if requested, whether the TEST mode is ON OF OFF. (See the "Use of Transitional Monitor" section for a description of the MODE card.)

To obtain a special end of program, the dependent program must have the following coding sequence to set the A bit of the /cco/ field on. This sequence must be in the coding before the program branches to /EOP/.

BXPA /MCR/
DCW /CCO/
DCW (address of an A bit)
DCW (address of dependent program routine to be given control if field cannot be modified)

(Next sequential instruction in the dependent program. Usually this instruction is B /EOP/.)

Information for Dependent Programs

The following rules are programming requirements that must be met by all dependent programs.

Use of the Resident IOCS

All programs within the Operating System — those supplied by IBM and those that are user-written — use the Resident IOCS. The Autocoder, COBOL, and FORTRAN processors automatically generate the necessary linkages to the Resident IOCS, in accordance with the input/output statements of each source language. (The Autocoder language statements for input/output operations are contained in the Basic Input/Output Control System publication.)

No Halts

Dependent programs must not use programmed halts. At end of program, the programs must branch to the Resident Monitor's End-of-Program routine.

Index Registers

The System Monitor sets a word mark in the highorder position of each index register. Group marks must not be placed in that position. Dependent programs must not clear or set word marks in the index registers.

Each time the Transitional Monitor is brought into storage by the Resident Monitor to process Monitor control cards, it clears all index registers, but leaves the word mark in the high-order position of each register.

Various elements of the System Monitor require index registers 14 and 15. Therefore, X14 and X15 must not be used by any interruptable instruction sequence in a dependent program. If used by a non-interruptable sequence, X14 and X15 must not be left negative.

Index register 13 is used for subroutine linkage by all programs within the Operating System. (For example, the first instruction in the Resident Monitor's Print routine is Store B-Address Register into X13. The Print routine then uses the contents of index register 13 for reference to the control information in the linkage sequence and for returning to the dependent program at the instruction following the linkage sequence.)

Index register 13 should be used for linkage to all subroutines introduced into the Operating System by the installation. Because of the convention of using index register 13 for subroutine linkage, IBM subroutines do not save and restore the contents of this reg-

ister. Subroutines introduced by the installation also do not need to save and restore its contents. However, if a dependent program uses index register 13 for purposes other than subroutine linkage, it should save the contents before issuing a linkage sequence to a subroutine and should restore the contents of the register after control is returned from the subroutine.

If index registers 13, 14, and 15 are used by installation programs, they must never be left negative.

Word Separator Characters

A word mark should never be set over a word separator character. (A word mark over a word separator character would be placed on the Core Image file as three word separator characters, but would be read from the file into core storage as a word separator character with a word mark over the following character.)

Priority Alert Mode

Dependent programs should not exit from Priority Alert mode, except as specified in the section, "Use of Resident Monitor Functions," when the dependent program is addressing the Resident Monitor (such as a branch to the Print routine).

Unit-Record Interrupts

A dependent program cannot be initiated by means of the Priority Select On-Off Switch on the console.

/SIZ/ Field

A dependent program should have a character in the highest core-storage position it will use. This character enables the Load routine to properly set the /srz/ field of the Communication Region.

Reading from the Standard Input Unit

Dependent programs must use the Read routine of the Resident Monitor to perform any read operations for the Standard (or Alternate) Input Unit.

Divide Overflow and Arithmetic Overflow Indicators

Each time the Transitional Monitor is brought into storage by the Resident Monitor to process Monitor control cards, it turns off the divide overflow and arithmetic overflow indicators.

Two Highest Positions of Core Storage

The Operating System places group marks with word marks in the two highest positions of core storage. These group marks with word marks must not be destroyed by dependent programs.

System Regeneration

If the installation regenerates the Operating System and if the Resident Monitor in the new System Operating File was changed from that in the previous System Operating File, the Job files produced by the previous Operating System cannot be used with the new system.

Monitor control cards in the Standard (or Alternate) Input Unit direct operation of the System Monitor. These cards are read and analyzed by the Transitional Monitor. The Monitor control cards are listed in Figure 7.

At System Generation, the installation can specify that the System Monitor write each Monitor control card on the console printer and/or the Standard Print Unit. The installation can also elect at System Generation to record all Job cards on the Standard Punch Unit to help separate punched output.

Figure 7 shows the format for Monitor control cards. The standard Autocoder rules apply to the operands of Monitor control cards:

- 1. Operands must be separated by a comma.
- 2. Operands cannot contain blanks.
- 3. An intentionally omitted operand must be indicated by placing its trailing comma adjacent to the preceding comma, except where the omitted operand is the last one.

JOB Card

The job card indicates the beginning of a job. The operand is the name assigned to the job by the programmer.

Upon reading a joв card, the Resident Monitor closes out the previous job as follows:

- 1. It completes any remaining rocs requests.
- 2. It clears Communication Region fields: size of dependent program field (/siz/), interprogram information field (/ipi/), phase number field (/phn/), program name field (/pnm/), and other fields used internally.
- 3. It turns off the indicators in the /cco/ field set for the operands of the MODE card (described below).
- 4. For tape Standard Print and Punch Units, it closes, rewinds, and unloads the tapes, if requested, to permit off-line processing after the previous job is completed. (These indicators are set by \$B2 and \$B3 console inquiries, which are described in the section, "Messages and Inquiries.")
- 5. It cancels all Reserve file assignments and all assignments of *alternate* physical units. (See the "Assignment of Input/Output Units" section.)

The Resident Monitor spaces the Standard Print Unit to the next page and prints on the Standard Print Unit and the console printer one line, which gives:

1. An "S" or "A" to indicate whether the JOB card

was read from the Standard or the Alternate Input Unit.

- 2. A two-character sequence number that indicates the number of job cards that have been read from that unit since the beginning of the batch.
 - 3. The contents of the Job card.

The JOB card contents and information will be written on the Standard Print Unit and the console printer whether or not other Monitor control cards are also printed and/or typed. An additional feature, which can be specified at System Generation, is the punching of JOB cards on the Standard Punch Unit.

The System Monitor does not use the operand(s) of Job cards. However, an installation-written module to use the operand(s) can replace the Transitional Monitor's IBACCOUNT module.

The IBACCOUNT module, as supplied in the Transitional Monitor, is a dummy module. After the JoB card is printed and all other JoB card functions are completed, the Transitional Monitor branches to this module, which, as supplied, will immediately return control to the Transitional Monitor. The installation can put into this module accounting routines or any other types of routines desired. To replace the dummy module with an installation module, see information concerning generation of the Transitional Monitor in the System Generation publication.

MODE Card

This card describes the type of operation wanted in a job. The operands for the card are:

- 1. GO
- 2. TEST
- 3. sg

The co operand may appear on a card with TEST and/or sc operands. The co operand sets the B bit of the /cco/ field on. The B bit is turned off by the next JOB card or the next MODE card without a co operand.

The co operand is used to specify compile-and-go operation. If it is specified, the output from following compilations will be written on the Go file for subsequent execution within the job.

If the Go file being used was created any time before the last Linkage Loader run, the go mode must not be in effect. If the go mode is in effect when a previously created Go file is used, the Linkage Loader will tape-mark the Go file at the point where the tape is when the Linkage Loader execution begins.

ocation in Control Card Deck	6	Card 5 16 Type 20	21 Operand(s) 72	Functions	Remarks
before first JOB ard during nitialization	MON\$\$	DATE	yyddd yy=Last two num- bers in year ddd=Three-digit number for day of the year	Sets Communication Region field /DAT/.	Must be given in daily information cards during initialization of the System Monitor.
First card in a job	MON\$\$	JOB	ANY NAME AND INFORMATION	Closes out previous job and begins new job.	Clears /SIZ/, /IPI/, /PHN/, /PNM/. Turns off /CGO/ indi- cators for MODE card operands.
Before EXEQ card or program that vill use symbolic unit	MONSS	ASGN	ABC,XX,YY,ZZ ABC=Symbolic name for input/ output file XX,YY,ZZ=As- signment symbols for any number of physical units	Assigns the symbolic unit in the first operand to the physical units represented by the assignment symbols in the second and following operands. The second operand is the base unit and the additional operands are alternate units.	See the section, "Assignment of Input/Output Units," for full details.
	MON\$\$ MON\$\$	ASGN ASGN	ABC, XYZ LIB, SOF	Assigns the symbolic unit in the first operand to the physical units for the symbolic unit in the second operand.	Limited to Work files and Reserve files, and to the combination: System Library file and System Operating File.
	MON\$\$	ASGN	ABC	Cancels the assignment previously made for the symbolic unit in the operand.	
Sefore EXEQ card for program to which MODE card applies	MONSS MONSS MONSS MONSS MONSS MONSS	MODE MODE MODE MODE MODE	GO,TEST GO,SG GO TEST SG	GO operand specifies that lan- guage processors write on Go file for compile-and-go opera- tion. TEST operand specifies that rest of job is to be executed if pro- gram comes to an unusual end. SG operand specifies that System Generation functions are to be performed. (See the System Generation publication.)	MODE card without GO turns GO indicator off. NOTE: If a previously created Go file is being used, do not use MODI GO. TEST indicator, once turned on, remains on until next job. SG indicator, once turned on, remains on until System Monitor is initialized.
After JOB card	MON\$\$	EXEQ	NAME1, MJB,XXX	Initiates search for program in first operand on System Operating File (SOF) or Job file (MJB) specified in second operand, then causes it to be loaded and executed.	
			card of program NAME1 for IBM-provic LINKLOAD (Linkag AUTOCODER) COBOL FORTRAN SORTDEFINE UTILITIES SGI	(language processors) (Sort Definition program) (Utility programs) tem Generation programs) (Tele-Processing system programs) (Disk Library Loader program) here	
Anywhere	MONSS	COMT	ANY INFORMATIO TO BE TYPED AND PRINTED	N Information in columns 21 through 72 is typed on console printer and printed on Standard Print Unit.	Information is typed and printed, whether other Monitor control cards are similarly recorded or not
At end of cards in Alternate Input Unit and Standard Input Unit	MON\$\$	END		In Alternate Input Unit, causes Resident Monitor to resume reading from Standard Input Unit, after giving operator opportunity to close Standard Print Unit, Stand- ard Punch Unit, and Core Image file. In Standard Input Unit, signifies end of current series of jobs and	If Standard and Alternate Input Units are on tape, END card causes tape to be rewound and unloaded.

Figure 7. Monitor Control Cards

The TEST operand specifies that following programs are being tested. At unusual end of program, the Resident Monitor's End-of-Program routine preserves the contents of storage on the Core Image file, if this file was specified at System Generation. Then, if TEST mode was specified, the Transitional Monitor, when brought into storage, reads the Standard or Alternate Input Unit for any Monitor control cards. Thus, a card requesting execution of the Utility programs to print the Core Image file can be read. If TEST mode had not been specified, the Resident Monitor would cancel any remaining portion of the current job.

If a Utility programs storage print is wanted when a program under test successfully reaches end of program, the program must branch to /UEP/.

The TEST indicator, when turned on by a MODE card, stays on until the next job or until System Monitor initialization.

The sc operand turns the System Generation indicator on. This operand specifies that a System Generator file or System Operating File is being generated, or that System Generation maintenance procedures are being performed. (See the System Generation publication.)

EXEQ Card

The EXEQ card initiates a search for the named program and causes it to be loaded and executed. Before the program is loaded, storage from the locations specified in the /ORG/ field to the top of core minus two (/AMS/) will be cleared.

The program to be executed is named in the first operand. This program may be an installation program, the Linkage Loader, or the other IBM-provided programs. The name to be used for an installation program is defined by the first PHASE card of the program.

The second operand specifies the file containing the program. This operand can be either sor for the System Operating File or MJB for the Job file. If the second operand is omitted, the program is assumed to be on the osr. (Omission must be indicated by a comma, if additional operands are used.) The Resident Monitor's Load routine loads the named program from the file specified.

The third operand, which is optional, changes the Resident Monitor's Read routine to read from a substitute unit instead of the Standard or Alternate Input Unit. After the program is executed, the Read routine automatically resumes reading from the Standard Input Unit. This operand cannot be used if the exeq card is in the Alternate Input Unit, because the Read routine treats the specified unit as a temporary Alternate Input Unit. This operand can be used only if the facility for reading an Alternate Input Unit was incorpo-

rated into the Resident Monitor at System Generation.

This operand must be the symbolic name for a Work file, Reserve file, the Standard Input Unit, or the Alternate Input Unit.

For an installation program, the third operand can specify the location of input data. (No Monitor control cards should appear in this input data.)

For the language processors, the third operand indicates that the source statements do not immediately follow the EXEQ card. This operand specifies the unit containing the source statements.

For the Linkage Loader, the third operand indicates the location of the Linkage Loader control cards.

The EXEQ LINKLOAD card can have a fourth operand to specify the System Library file for this Linkage Loader run. In a tape-oriented system, the Linkage Loader uses two factors to determine the Library to be used:

- 1. The fourth operand of the EXEQ card. If blank, the Linkage Loader assumes that the Library is the IBM Library (named IBMLIBR) on the System Operating File.
- 2. The ASON card. If absent, the Linkage Loader assumes that the Library named is on the System Operating File.

In a disk-oriented system, the Library name must be specified in the fourth operand of the EXEQ card and an ASGN card for the Library must be present.

Installation programs can specify fourth and succeeding operands for use of the program named in the EXEO card.

When the EXEQ card contains more than three operands, the address of the high-order character of the fourth operand is placed in card columns 6 through 10 in the card input area, overlaying MON\$\$. No word mark is set in column 6. The /CRD/ field contains the high-order address of the card input area until the next read operation performed by the Read routine.

An EXEQ LINKLOAD card can have an S in column 1 to specify that the memory map should not be printed. However, the S in column 1 will not suppress the printing of diagnostic messages. (See the "Messages and Inquiries" section.)

COMT Card

The COMT (Comments) card is used to record information on the console printer and the Standard Print Unit. All information in columns 21 through 72 of the COMT card is typed and printed, whether other Monitor control cards are similarly recorded. The COMT card is convenient to:

- 1. Give additional instructions to the operator.
- 2. Identify a program's output in greater detail than the 10B card permits.

ASGN Card

The ASGN card is used to assign input/output units. The functions of the ASGN card are described in detail in the section, "Assignment of Input/Output Units."

END Card

The END card must be the last card in the Standard Input Unit and the Alternate Input Unit.

When this card is read at the Standard or Alternate Input Unit, the Resident Monitor does the following:

- 1. Closes the unit with a rewind and unload, if the unit is tape.
- 2. Types two messages: END SIU (or END AIU) and 20501 ENTER B MESSAGES. All class B console inquiries will be accepted except \$B1, which will be ignored if the END card was on the Standard Input Unit. (For an END card on the Alternate Input Unit, see item 4.) Thus, the operator can close the Standard Print Unit, the Standard Punch Unit, and/or the Core Image file. During this time, the System Monitor is in a waiting loop and Tele-Processing system interrupts may occur. (See the "Messages and Inquiries" section for details on console inquiries.)
- 3. Remains in the waiting loop until the \$BX inquiry is entered. Then, the Transitional Monitor resumes reading from the Standard Input Unit, which must be on the same physical unit used in the previous batch. If the END card was on the Standard Input Unit, the

Transitional Monitor will ignore all cards until it reaches the first JOB card. Then it will begin processing. If the END card was on the Alternate Input Unit, the Transitional Monitor resumes processing with the next Monitor control card on the Standard Input Unit.

4. If the END card was on the Alternate Input Unit, a \$B1 inquiry (before the \$BX inquiry) will make the Transitional Monitor resume reading from the Alternate Input Unit.

Two counters are used to determine the number of JOB cards read since the beginning of a batch. One counter is associated with JOB cards read from the Standard Input Unit, the other with JOB cards read from the Alternate Input Unit. Each time an END card is read from one of the input units, the counter associated with that unit is set to zero.

DATE Card

The DATE card contains the date in its operand. The first two positions contain the last two numbers in the year, and the next three positions contain the three-digit number for the day of the year.

This card is required in the daily information, which is before the first JOB card, during initialization of the Resident Monitor. (See the *Operator's Guide* publication for details on initialization.)

The operand is placed in the /DAT/ field of the Communication Region.

Assignment of Input/Output Units

Input/output units are referenced as:

- 1. PHYSICAL UNITS. *Physical unit* designates all of, or sections of, a physical device. Each physical unit is assigned a two-character *assignment symbol* at System Generation. The assignment symbols are selected by the installation.
- 2. SYMBOLIC UNITS. Symbolic unit designates a file to be read from or written on a physical unit. Symbolic units are referred to by the symbolic names given in this section.

The symbolic unit names are used in the SYMUNIT entry of the IOCS Define the File statement in programs (see the *Basic Input/Output Control System* publication) and in the first operand of the Monitor ASCN card. The symbolic unit files are assigned to one or more physical units by the ASCN card and IBLOOKM linkage sequence. The assignments are made by assignment routines in the Resident and Transitional Monitors.

To enable an installation to make the most efficient use of input/output units for a series of jobs, program-references to input/output units are made symbolically. Also, the programmer is freed from concern as to which specific input/output units will be available when his program is executed.

A physical unit is one of the following:

- 1. A reader, punch, printer, or other unit-record device.
 - 2. A tape unit.
- 3. A set of consecutive tracks in a 1301 Disk Storage module.

Note: The 1402 Card Read Punch must be defined for each function. (The read function is specified as one physical unit, the punch function as another.)

The assignment symbols specified for physical units during System Generation are placed into a table in the Resident Monitor. During operation of the system, this table represents the input/output units that are available for use by dependent programs and by the System Monitor.

This table can be changed – in terms of additions and deletions – only by another System Generation. The operator, however, can indicate through \$B5 and \$B6 inquiries that a physical unit is not currently available for use, or that it is again available. (See the "Messages and Inquiries" section for details on console inquiries.)

Note: Definition of Tele-Processing devices is also performed during System Generation, but in a differ-

ent manner. They are defined by specifying the related elements of the Tele-Processing Supervisor. (See the *Tele-Processing Supervisor* publication.) The definitions of Tele-Processing devices are not placed in the Resident Monitor's table. However, physical units used by programs operating within the Tele-Processing system (other than Tele-Processing devices) are given assignment symbols in the same manner as physical units for other programs, and their assignment symbols are also placed in the Resident Monitor's table.

Physical Units

The following example illustrates one possible set of assignment symbols to represent physical units. Since any scheme of two-character symbols can be incorporated into the System Monitor, each installation can designate the set of assignment symbols that best suits its needs.

Channel 1 Tape Units: A0 (A-zero) through A9 Channel 2 Tape Units: B0 through B9 Unit-Record Equipment:

1402 Card Read Punch (read function): C1 1402 Card Read Punch (punch function): C2 1403 Printer: C3

Tracks of Disk Storage (Channel 1, Module 0):

0000-0079: D1 0120-0159: D2 7000-7999: D3

(Note that although the tracks that constitute each physical unit must be consecutive, separate physical units need not be consecutive or contiguous.)

Tracks of Disk Storage (Channel 1, Module 0):

4000-8999: E1 9315-9386: E2

(Note that track assignments need not begin and end at cylinders.)

Tracks of Disk Storage (Channel 2, Module 0): 0000-9999: F1

Note: Physical unit track definitions may overlap each other.

Symbolic Units

The symbolic names to be used for symbolic units are defined during System Generation, when they are placed in the Resident Monitor table. During system operation, this table is used in conjunction with the table of physical units to assign input/output units for program use.

Symbolic units are divided into four groups, in accordance with the type of program that uses the symbolic unit:

- 1. System files are used by the System Monitor.
- 2. Work files are used by IBM-provided components of the Operating System.
 - 3. Reserve files are used by installation programs.
- 4. Tele-Processing system files are used by installation programs that operate under control of the Tele-Processing Supervisor.

Note: Work files can also be used by installation programs, but care must be exercised to ensure that conflicts do not arise between assignments for ibmprovided programs and assignments for installation programs.

The symbolic names for the symbolic units are listed below. For each System file, the symbolic name is predetermined; for the other three groups of symbolic units, the symbolic names are assigned from the specific sets by the Operating System at System Generation. System files:

```
SOF

    System Operating File

       - Standard Input Unit
SIU
       - Standard Print Unit
SPR
       - Standard Punch Unit
SPU
AIU
       - Alternate Input Unit
       - System Library file
LIB

 Job file

MJB
       -Go file
MGO
       - Core Image file
MDM
       - TP Library file (for Tele-Processing sys-
MLT
       - Temporary Storage file (for Tele-Process-
MDT
           ing system)
```

Work files:

MW0 (MW-zero) through MW9 MWA through MWZ

Reserve files:

MRO through MR9

MRA through MRZ

Tele-Processing system files:

мт0 through мт9 мта through мтz

Assignment Times for Symbolic Units

The symbolic units are divided into five categories with respect to the times that they can (or must) be assigned to physical units:

1. Symbolic units that are assigned during System Generation, initialization, or reinitialization:

SOF (System Operating File) SIU (Standard Input Unit)

Note: The device type for the System Operating

File cannot be changed at initialization or reinitialization.

2. Symbolic units that are assigned during initialization or reinitialization of the System Monitor and cannot be reassigned, except by initialization or reinitialization procedures:

SPR (Standard Print Unit) SPU (Standard Punch Unit) MDM (Core Image file)

Note: Assignment of these units is part of the daily information supplied to the System Monitor during initialization and reinitialization procedures. These units must be assigned if facilities for them were incorporated into the Resident Monitor at System Generation.

3. Symbolic units that can be assigned during initialization or reinitialization procedures and can also be assigned or reassigned between each run within a job:

```
LIB (System Library file)
AIU (Alternate Input Unit)
MWX (All Work files)
```

Note: The Alternate Input Unit cannot be assigned by a control card that is read from the Alternate Input Unit. The System Library file should not be assigned if it is on the System Operating File. Three Work files must be assigned for the Autocoder and COBOL language processors; the physical units, if tape drives, for two of these files should be on one channel and the physical unit for the third file should be on the other channel, for maximum processor efficiency. The physical units, if tape, for the two Work files that must be assigned for the FORTRAN processor should be on different channels. (See the *Operator's Guide* publication for full details.)

4. Symbolic units that are assigned between each job:

```
мјв (Job file)
мсо (Go file)
мкх (All Reserve files)
```

Note: The System Monitor cancels the assignment of these units at each job card. These files can be assigned (or reassigned) between each run within a job. However, care should be taken in reassigning the Job and Go files to ensure that the Linkage Loader and language processors have the proper input/output files.

5. Symbolic units that can be assigned at System Generation or at other times when the Tele-Processing system is not open:

```
MLT (TP Library file)
MDT (Temporary Storage file)
MTX (All Tele-Processing system files)
```

Note: The assignment of these units is canceled when the total System Monitor is initialized or reinitialized. Between initializations (or reinitializations)

of the total System Monitor, these assignments are maintained, regardless of the number of times the Tele-Processing system is opened and closed. The assignments can, however, be changed by ASGN cards each time the Tele-Processing system is closed.

Program References to Symbolic Units

After the installation has established the set of symbolic names for the symbolic units, these names are used in installation programs to designate the symbolic units for a program's files. For example, in Autocoder, the symbolic name is used in the symunit entry of the Define the File statement. COBOL and FORTRAN similarly provide means by which the programmer identifies symbolic units for a program.

The Linkage Loader converts the symbolic names for the symbolic units into machine addresses that enable the Resident and Transitional Monitors to assign input/output units in accordance with the ASGN cards.

ASGN Card

The Ason card, which is a Monitor control card, directs the System Monitor to assign one or more physical units to a particular symbolic unit.

As shown in Figure 8, the first operand of an ASGN card is the symbolic name for the symbolic unit. The second operand is the assignment symbol for the first physical unit (the base unit) to be assigned to that symbolic unit. Additional operands can be used to specify alternate units.

In Figure 8, A1 is the base unit; A2 and A3 are the alternate units. When end of unit is reached for A1, the next input/output operation for MR1 will be performed with the physical unit represented by A2. (For tape, end of unit is determined by reaching end of reel; for disk, end of unit is determined by reaching the end of the set of tracks that constitute one physical unit. The use of alternate units does not apply to unit-record equipment.)

At the end of unit for A2, the next input/output operation for MR1 will be performed with the physical unit represented by A3. The end of unit for A3 is considered end of file for the symbolic unit MR1, if MR1 represents a disk file. However, if MR1 represents a tape file, continuous cycling of the physical units is performed. Thus, A1 would be the fourth physical unit used. In such a case, end of file is determined by reel counts, trailer labels, or end-of-reel conditions (for unlabeled tapes.).

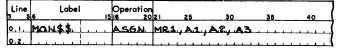


Figure 8. Format of the ASGN Card.

A physical unit can be assigned to more than one symbolic unit at the same time. For example, a job involving a compile-and-go operation could use a tape unit as a Work file for the language processor and then use the same unit as a Reserve file for the program that was just compiled. The ASGN cards for the two symbolic units can both be placed immediately behind the job card for this job. The System Monitor makes the assignments as the ASGN cards are read, but since assignments are not used until a program initiates an input/output operation through the locs, no conflict occurs because of the dual assignment. Figure 9 illustrates the use of two ASGN cards to assign the same physical unit to different symbolic units.

Line 3 5	Label	Operation	21	25	30	35	40
0,1,	MON \$ \$	 ASGN	MW4	S.A.			
	MONSS.	 ASGN	MRZ	AZ			1 1 1
0,3,	l	 					

Figure 9. Assignment of Two Symbolic Units to the Same Physical Unit

Note that all symbolic units that share a base unit also share any alternate units specified for that base unit. The physical units are shared in the same order. A base unit assigned to one symbolic unit cannot be assigned to another symbolic unit as an alternate unit. (For example, these two assignments cannot exist simultaneously: MR1,A1,A2 and MR2,A2.) However, if an alternate unit must be assigned as the base unit of another symbolic unit later in the job (or vice versa), the initial assignment must be canceled.

CANCELLATION OF ASSIGNMENT

Symbolic unit assignments are canceled by an ASGN card containing only the first operand. For example, the ASGN card in Figure 10 cancels the assignment for the Reserve file MR1. This cancellation permits the physical units previously assigned to MR1 to be used for another file.

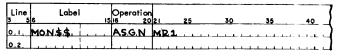


Figure 10. Cancellation of Assignment

The ASGN card in Figure 10 and the ASGN card used to assign MRl's physical units to the next file should both be placed immediately before the EXEQ card for the program that uses the next file. (The card that can-

cels the assignment to MRI must precede the card that makes the new assignment of the physical units.)

PLACEMENT OF THE ASGN CARD

The ASGN cards are submitted to the System Monitor through the Standard (or Alternate) Input Unit. The position of an ASGN card in the control card deck is determined by the category of symbolic unit named in the asgn card.

ASGN cards for symbolic units that are assigned during initialization or reinitialization are in the daily information in the Standard Input Unit.

ASGN cards for symbolic units that are assigned only for the duration of a job must follow the related JOB card and precede the EXEQ card for the program that uses the files.

ASGN cards for symbolic units that are used by the Tele-Processing system must precede the control card or the console inquiry that opens the Tele-Processing system.

Use of the ASGN Card for System Files

The physical units assigned to certain System files cannot be assigned to any other symbolic units during the time they are assigned to the System files. These System files are:

sof (System Operating file) SIU (Standard Input Unit) SPR (Standard Print Unit) SPU (Standard Punch Unit) LIB (System Library file) MDM (Core Image file)

In addition, physical units assigned to symbolic units used by the Tele-Processing system cannot be assigned to any other symbolic units when the Tele-Processing system is ready to receive input from Tele-Processing devices.

The physical units assigned to the System Operating File and the Standard Input Unit can be reassigned to other symbolic units only through System Generation, initialization, or reinitialization. The physical units assigned to the Standard Print Unit, the Standard Punch Unit, and the Core Image file can be reassigned to other symbolic units only through initialization or reinitialization. The physical unit assigned to the System Library file (if this file is not part of the System Operating File) can be reassigned to other symbolic units either through initialization and reinitialization or through cancellation of the Library's assignment by an ASGN card containing LIB as the only operand or containing the operands sof, LIB.

For compile-and-go operations, the Go file must not be assigned to a physical unit used for the language processor's Work files.

Linkage Sequence for Assignment of Symbolic Units

Through the IBLOOKM linkage sequence, a dependent program during its execution can set the symbolic unit selection for any Reserve file, Work file, or Tele-Processing system file. Using this sequence, the dependent program is written to operate with any symbolic unit. It is stored in absolute format in the System Operating File. If this sequence were not used, the program would have to be processed by the Linkage Loader each time to resolve the one symbolic unit entry that would change with each execution.

The linkage sequence is:

DCWS **IBLOOKM** @sym@ DCW DCW FILE

> (address of dependent program instruction to which control is to be given if symbolic unit requested is not a Reserve, Work or Tele-Processing system file or does not exist)

(Next sequential instruction in dependent program)

Where sym = The symbolic name of the symbolic unit to be used by the dependent program.

> FILE = The label used in the Define the File Header Line for the corresponding file

Note: The iblookm module must be on the currently assigned System Library file.

To use this sequence, a control card can be read during program execution to supply the information for the sym operand. Such a program could be a Utility program written to print any tape file.

The following Autocoder macro-instruction can be used to generate the above linkage sequence:

RESOLVE operand:

ANYLABEL MONOP RESOLVE, SYM, FILE, NOMODADDR must appear as shown.

sym operand: is the symbolic name for the symbolic unit.

FILE operand:

is the label for the Define the File Header Line for the corresponding file.

NOMODADDR operand: optional. This is the ad-

dress of the dependent program instruction to which control is to be given if the symbolic unit requested is not admissible. Ιf omitted. control will be returned to the next sequential instruction following the macro-instruction.

During operation, the System Monitor types information and diagnostic messages on the console printer and/or prints them on the Standard Print Unit. If an installation does not specify a Standard Print Unit at System Generation, it will not receive certain information and messages that are printed only on the Standard Print Unit.

The System Monitor can also receive inquiries through the console printer.

Memory Map

The Linkage Loader will write information on the Standard Print Unit to provide a memory map of the program being relocated. This memory map information will be interspersed with Linkage Loader diagnostic messages.

The column headings printed at the top of the memory map are:

MESSAGE CARD NAME VALUE SYMBOL VALUE SYMBOL VALUE SYMBOL VALUE SYMBOL VALUE SYMBOL VALUE

All Linkage Loader error messages will begin in the MESSAGE column.

The CARD column contains the name of the following five key Linkage Loader control cards in the order in which they are processed. All information about each card will be printed on one line.

PHASE Card: The word PHASE will appear in the CARD column. The program name will appear in the NAME column for a PHASE card with an operand; for a blank-operand PHASE card, the NAME column will be blank. The upward relocation factor for the phase will be printed in the leftmost VALUE column. Note that if a BASE1 card is processed after the PHASE card, but be-

fore any load cards, the VALUE column of the PHASE card line will not be accurate.

BASE1 Card: The word base1 will appear in the card column. The leftmost value column will contain the new upward relocation factor.

BASE2 Card: The word BASE2 will appear in the CARD column. The leftmost VALUE column will contain the new downward relocation factor.

PRTCT Card: The word PRTCT will appear in the CARD column. The leftmost VALUE column will contain the new protection value for the Linkage Loader symbol table.

TITLE Card: The word TITLE will appear in the CARD column. The subprogram name will appear in the NAME column. The leftmost VALUE column will contain the absolute address at which the first character in the subprogram will be loaded. To the right of the leftmost VALUE column will be printed REL FACTOR XXXXX, where XXXXX is the relocation factor actually used in relocating the subprogram.

The linkage symbols used, and their corresponding absolute addresses, will be printed in the five sets of SYMBOL and VALUE columns in the right half of the memory map.

Monitor Diagnostic Messages

The following messages are typed on the console printer and/or printed on the Standard Print Unit. These messages are:

- 1. Diagnostic messages for programmers
- 2. Messages for operators
- 3. Initialization messages
- 4. Bootstrap routine messages
- 5. Messages during restart

Diagnostic Messages for Programmers					
MESSAGE	EXPLANATION	LOCATION	ACTION		
UEPX UEP1 UEP2 UEP321274 UEP938918 UEP9	Unusual end of program. Execution of program has been terminated or the Transitional Monitor has determined that the remainder of the job cannot be successfully executed. If $x = 1$: Dependent program requested a phase that could not be found. If $x = 2$: Dependent program requested a phase to be loaded and executed. Phase was loaded, but has no entry point.	printer	None by operator.		

EXPLANATION

LOCATION

ACTION

If x = 3: Dependent program attempted input/output with a symbolic unit for which no physical unit was assigned. Or, a file table in the dependent program contains a symunit field with an address that does not correspond to any entry in the Resident Monitor's table of symbolic units.

Note: If x = 3, the message contains the address of the units position of file table field before the symunit field.

If x = 9: A branch to the /UEP/ entry point of the End-of-Program routine was made from outside the Resident Monitor. When a dependent program branches to /UEP/, the message will contain the contents of the B-Address Register at the time of the branch. If the address is absent, i.e., UEP9 message, the Transitional Monitor entered the Endof-Program routine at the /UEP/ point and the status of core storage was not written on the Core Image file.

00501 sofDIRECTORY NOT FOUND

00502 мјв DIRECTORY NOT FOUND

00503 INVALID DIRECTORY RECORD

10520 CGO-LL

10521 ссо-мјв

10522 1st op inv 10522 2nd op inv 10522 3RD OP INV

10523 no aiu

10524 AIU ACTIVE

10525 chg ttttt

Transitional Monitor is unable to find a directory for the System Operating

Transitional Monitor is unable to find a directory for the disk Job file. The and/or Standard directory was inadvertently destroyed, an Print Unit error was made in assignment, or the Linkage Loader is not functioning prop-

Transitional Monitor has located what should be the directory for the disk System Operating File, but record read was Print Unit not a valid directory record.

A language processor has canceled the Go file and execution of the Linkage Loader is, therefore, being bypassed.

The contents of the Job file are not valid; therefore, the exec card for a and/or Standard program on the Job file is being bypassed. Print Unit

An exec card with an invalid operand

has just been read and is being bypassed. An exec card with a third operand has just been read but cannot be processed because facilities for using an

Alternate Input Unit were not specified

at System Generation. An exeo card with a third operand has just been read from the Alternate and/or Standard Input Unit and is, therefore, being by- Print Unit passed.

The symbolic unit in the third operand of the EXEQ card just read cannot be and Standard Print found in the Resident Monitor's assign- Unit ment table. Therefore, the Transitional Monitor cannot effect a change to that unit. The exec card is bypassed.

ttttt = The address of the Transitional Monitor routine that tried to effect the change.

Console printer and/or Standard Print Unit

Console printer

Console printer and/or Standard

Console printer and/or Standard Print Unit

Console printer

Console printer and/or Standard Print Unit

Console printer and/or Standard Print Unit

Console printer

Console printer

None by operator. (System Operating File must be regenerated.)

Reinitialization is necessary, if the batch is to be continued. The situation should first be analyzed to determine the exact cause of error. Regeneration of the System Operating File may be required.

None by operator. (System Operating File must be regenerated.)

None by operator.

MESSAGE	EXPLANATION	LOCATION	ACTION
10526 CHG UNASGN	The symbolic unit in the third operand of the exeq card just read is not currently assigned to a physical unit. The exeq card is bypassed. (This message possibly could occur if parts of the Resident Monitor were destroyed, breaking the assignment for the Standard and/or Alternate Input Unit.)		None by operator.
10527 CHG INV	The symbolic unit in the third operand of the EXEQ card just read is currently assigned to a disk physical unit, which is not permissible. The EXEQ card is bypassed.	Console printer and Standard Print Unit	None by operator.
10529 go dsk exc	During compilation, the disk physical unit assigned to the Go file has been exceeded. The language processor has canceled the Go file and returned control to the System Monitor.	Console printer	None by operator.
10530 NOT FOUND	Tape System Operating File: The program named in the exeq card cannot be found on the System Operating File, if sof specified on card, or on the Job file, if MJB specified on card. The exeq card is bypassed. Disk System Operating File: The program named in the exeq card cannot be found in the directory specified by the card. The exeq card is bypassed.		None by operator. (The System Operating File or Job file should be analyzed to determine the reason for the failure, in a tape system. The directory for the System Operating File or Job file should be analyzed to determine the reason for the failure, in a disk system.)
10531 NOT FOUND IN SOF DIRECTORY	The program named in the EXEQ card is not listed in the tape System Operating File directory. The EXEQ card is bypassed.	and/or Standard	None by operator. (The tape System Operating File directory should be analyzed to determine the reason for the failure.)
10532 INCOMPATIBILITY IN BLOCK COUNTS	Using block counts to search the System Operating File, the Transitional Monitor is unable to locate the desired program. One correction attempt will be made and, if not successful, the System Operating File will be rewound and sequentially searched.		None by operator. Note: The frequent appearance of this message indicates one or more of the following conditions: 1. Block counts in directory are wrong. 2. Block counts in phase headers are wrong. 3. Backspace often fails on the tape unit. Correction of the trouble will improve search time for the System Operating File.
10535 mjb invalid loc	In searching the Job file, the Simple Search of System Tape subroutine has read one record from the file; this record is not a phase header record. The two possible causes are: 1. The physical unit assigned to the Job file does not contain the Job file. 2. The Job file tape has been moved to an incorrect position.	and/or Standard Print Unit	None by operator. The System Monitor will give control to the End-of-Program routine at the /UEP/ entry point.

Messages for Operators

MESSAGE	EXPLANATION	LOCATION	ACTION
20501 ENTER B MESSAGES	Transitional Monitor is ready to accept class B messages from operator.	Console printer	1. Press INQUIRY REQUEST. 2. Enter message. (Class A messages are also acceptable at this time.)

3. Press INQUIRY RELEASE.

Note: This procedure is repeated for each message. The last message must be \$BX to cause normal processing to resume.

MESSAGE	EXPLANATION	LOCATION	ACTION
20502 SUB FOR SS	An ASCN card has been read for a physical unit that is currently unavailable.	33113010	1. Press inquiry request. 2. Enter class C message: \$C1ss. 3. Press inquiry release. Note: The \$B6ss message cannot be used at this time to remove the unavailable indication from the unit named in the 20502 sub for ss message.
20503 INV AIU END SITUATION	END card or end of file has been encountered on the Alternate Input Unit, but a return to the Standard Input Unit cannot be effected. (Possible cause: contents of Standard Input Unit input area have been destroyed.) Transitional Monitor has entered a dead-end waiting loop.	Console printer	Reinitialization is necessary if batch is to be continued.
20504 INV B INQ	Invalid console inquiry has been made while Transitional Monitor is ready to accept class A or B messages. Monitor remains in waiting loop.	Console printer	 Press inquiry request. Enter valid message. (If none, enter \$BX.) Press inquiry release.
ASGN OF INVALID XNN	ASCN card operand is invalid. System Monitor rejects assignment and enters End-of-Program routine at /UEP/ entry point. nn = Number of ASCN card operand that was invalid. x = (blank): Either the symbolic unit or the physical unit could not be found in the Resident Monitor's assignment table. x = 4: Sequence rules for sharing physical units were violated. (For example, if "Mw2,A1,A2" is a current assignment, then "Mw5,A2,A3" cannot be made.) x = 8: Rules for sharing physical units were violated. (For example, the physical unit used by the Standard Input Unit cannot also be assigned to another symbolic unit.) x = T: Attempt was made to assign a Tele-Processing system was open.	Console printer	Correct ason card.
END AIU	END card or end of file has occurred on the Alternate Input Unit.	Console printer	None by operator.
END SIU	END card or end of file has occurred on the Standard Input Unit.	Console printer	None by operator.
INV A MSG	An invalid console inquiry was entered when only class A messages can be accepted. The message has been ignored and processing resumed.	Console printer	Repeat the console inquiry, if a valid class A message was intended. Otherwise, wait until "20501 ENTER B MESSAGES" appears on the console printer before repeating the console inquiry.
SWITCH TO NEXT SIU W/O INITIALIZATION	The System Monitor has begun reading the new Standard Input Unit batch in accordance with instructions from the console. (This message occurs when a new batch has been placed on the Standard Input Unit, and the waiting loop following the end of the previous Standard Input Unit has been broken by the \$BX inquiry.)	Console printer	None by operator.

Console

Repeat console inquiry.

Console inquiry was made while a Conso previous inquiry was being processed; printer error occurred during a console read; or console inquiry was canceled.

10104 REP INQ

MESSAGE EXPLANATION LOCATION ACTION

Console

10103 CPT nnx Records have been written on the Core Image file, because an unusual end printer

of program occurred or the current program took a checkpoint.

nn = Number of such records since the last initialization of the system.

x = C: Checkpoint.

x = D: Contents of core storage writ-ten on Core Image file following an unusual end of program.

None by operator.

Initialization Messages

CHAR

For details on initialization and reinitialization, see the Operator's Guide publication.

valid initialization status character. (That printer

is, it is not A, B, C, or D.)

MESSAGE	EXPLANATION	LOCATION	ACTION
DATE NOT ENTERED	First JOB card has been read, but no DATE card has been processed.	Console printer	 Press inquiry request. Enter five-character date. Press inquiry release.
ENTER DATE, I/O ASSIGN- MENTS, JOB NUMBERS	Initialization routine is ready to accept initialization information. (This message is given only during reinitialization without rewind.)	Console printer	 Press inquiry request. Enter five-character date. Press inquiry release. Wait for next message, which should be "io or jn".
ENTER JOB NUMBERS	Initialization routine is ready to accept job numbers for the Standard or Alternate Print Unit. (This message is given only during reinitialization with rewind.)	Console printer	1. Press inquiry request. 2. Enter two-character job numbers (Standard Print Unit first). If the Alternate Input Unit is not included in an installation or was not assigned when the need for reinitialization arose, enter two blank characters for the Alternate Input Unit job count. (A total of four characters must be entered.) 3. Press inquiry release.
ENTER RST NUMBER	Initialization routine is ready to accept the number of the checkpoint that is to be used for the restart.	Console printer	 Press inquiry request. Enter three-character checkpoint number. Press inquiry release.
ENTER SOF/SIU ASGN SYMB	Initialization routine is ready to accept the assignment symbols for the System Operating File and the Standard Input Unit. (This message is given only when a word mark has been entered into lo- cation 00000.)	Console printer	1. Press Inquiry request. 2. Enter assignment symbols for System Operating File and Standard Input Unit. Both must be entered even if only one assignment is being changed. System Operating File symbol must precede Standard Input Unit symbol. 3. Press Inquiry release.
EOF SIU/AIU IN INIT	Initialization routine has encountered an end-of-file condition on the Standard or Alternate Input Unit.	Console printer	Start over with the proper input on the unit.
INIT ASGN EXCEED TABLE	Initialization routine for disk-oriented system has been given more initial assignments than can be stored in the table it is building for reinitialization. (Because the table will accommodate up to 150 assignments, this message is not normally expected.)	Console printer	Start over with fewer ASGN cards preceding the first JOB card.
(ASGN card image) invalid ASSIGNMENT	Initialization routine has read an ASCN card that cannot be processed. (The ASCN card could be constructed internally from an assignment entered through the console printer.)	Console printer	None. (This message is given to supply a reference for the message "10 or IN RE ENTER XXX", which should immediately follow.)
INV INITIAL	Character at location 00000 is not a	Console	Start over with a valid initialization

status character.

ESS	

EXPLANATION

LOCATION

ACTION

IO OR IN

Initialization routine is ready to accept input/output assignments or job numbers. (This message is given only during reinitialization without rewind.)

The first time this message appears during reinitialization, it indicates the date has just been processed. Each successive time the message appears, it indicates that the assignment just entered through the console printer has been processed; the Initialization routine is ready to accept the next assignment or, if all assignments have been given, the job numbers for the Standard and Alternate Input

Console printer

1. Press inquiry request.

2. Enter a five-character assignment (symbolic unit name followed by assignment symbol) or, if all assignments have been entered, enter two-character job numbers (Standard Input Unit first). If the Alternate Input Unit is not included in an installation or was not assigned at the time the need for reinitialization arose, enter two blank characters for the Alternate Input Unit job number.

3. Press inquiry release.

Note: An inquiry request and in-QUIRY RELEASE must precede and follow each five-character assignment entry and the four-character job numbers entry.

IO OR JN RE ENTER XXX

This message is given to permit the operator to correct an invalid assignment printer or to cancel the assignment by entering the job numbers for the Standard and Alternate Input Units.

Console

1. Press inquiry request.

2. Enter five-character assignment or two two-character job numbers.

2. Enter current assignments for both

the System Operating File and Standard

Input Unit. Enter the two-character as-

signment symbol for the System Operating File first, immediately followed by symbol for Standard Input Unit. 3. Press inquiry release.

3. Press inquiry release.

1. Press inquiry request.

MUST ENTER SOF/SIU ASGN SYMB

System Operating File is on a unit different from the one specified at System printer Generation, but operator did not enter a word mark at location 00000 to request a change of assignment.

Console

SHARE VIOLA-TION MONITOR

COMPILATION

At System Generation, the rules concerning the assignment of System files printer were violated. This violation is detected during the first initialization of the System Operating File containing the invalid assignments.

Console

None. (A new System Operating File must be generated to correct the invalid assignment(s).)

JOB SEARCH OUT OF STEP

Something has caused the Initialization routine to search past the job count printer entered. (This message is given only during reinitialization with rewind.)

Console

Begin reinitialization again.

Bootstrap Routine Messages

MESSAGE

EXPLANATION

LOCATION

ACTION

BOOTSTRAP ERROR 01

Bootstrap routine, in a tape system, has found a header record, rather than printer an execute record, after the Resident Monitor record on the System Operating File. The Bootstrap routine has entered a waiting loop.

Console

Begin bootstrap again.

BOOTSTRAP ERROR 02

Tape read error, other than wronglength record, persists after re-reading printer nine times. The Bootstrap routine has entered a waiting loop.

Console

Begin bootstrap again.

BOOTSTRAP ERROR 03

A tape read error occurred while the Bootstrap routine was trying to read a printer header record. The Bootstrap routine has entered a waiting loop.

Console

Begin bootstrap again.

BOOTSTRAP ERROR 04

The first record after the Bootstrap routine is not a header record. The Boot- printer strap routine has entered a waiting loop.

Console

Begin bootstrap again.

MESSAGE	EXPLANATION	LOCATION	ACTION
bootstrap error 05	The instruction loaded by the Bootstrap routine into location 00000 is not valid because it does not contain a non-overlap character in location 00002. The Bootstrap routine has entered a waiting loop.		Begin bootstrap again.
BOOTSTRAP ERROR 06	Disk read error persists after re-reading five times. The Bootstrap routine has entered a waiting loop.		Begin bootstrap again.
BOOTSTRAP ERROR 07	The Bootstrap routine has found an unidentifiable record while loading the Resident Monitor. The Bootstrap routine has entered a waiting loop.		Begin bootstrap again.
00438	The instruction loaded by the Bootstrap routine into location 00000 is not valid because it does not contain a valid non-overlap character for any channel in location 00002. The computer has halted.	Console printer	Begin bootstrap again.
00202	A tape read error, other than wrong- length record, occurred while the Boot- strap 2 routine was being read. The com- puter has halted.	Console printer	Begin bootstrap again.

Messages During Restart

MESSAGE	EXPLANATION	LOCATION	ACTION
10533 siu u/r	Because the Standard Input Unit is a card reader, no attempt will be made by the Restart program to reposition the Standard Input Unit during restart.		None by operator.
10534 aiu u/r	Because the Alternate Input Unit is a card reader, no attempt will be made by the Restart program to reposition the Alternate Input Unit during restart.	Console printer	None by operator.
10528 SPU-SPR SAME UNIT	Because the Standard Punch Unit and Standard Print Unit share the same tape unit, the Restart program will make no attempt to reposition this unit during restart.	Console printer	None by operator.

Linkage Loader Diagnostic Messages

The following is a list of diagnostic messages printed by the Linkage Loader on the Standard Print Unit.

MESSAGE	MEANING	ACTION
SOF LIBRARY NAME UNEQUAL TO EXEQ CARD	The name of the relocatable library specified in the fourth operand of the EXEQ LINKLOAD card cannot be found on the System Operating File.	The Linkage Loader will end execution with a special end of program.
/LIB/ HEADER NAME UN- EQUAL TO EXEQ CARD	The physical unit assigned to LIB does not contain the library named in the fourth operand of the EXEQ LINKLOAD card.	The Linkage Loader will end execution with a special end of program.
INPUT CARD INVALID	The operand of the INPUT card cannot be identified by the Linkage Loader.	The Linkage Loader will end execution with a special end of program.
***** BAD INPUT CARD (followed by contents of card)	The card type in column 72 or the card name in column 16 is in error. (The <i>input card</i> can be any card, not just the INPUT control card.)	The Linkage Loader will end execution with a special end of program.

MESSAGE

MEANING

***** BAD INDICATOR (followed by contents of card) is

***** CORE EXCEEDED (followed by contents of card)

**** SYMBOL UNDEFINED (followed by contents of card)

**** TABLE OVERFLOW

***** SNAP IGNORED (followed by contents of SNAP card)

bad LOAD ADDR (followed by contents of card)

***** JOB FILE NOT CREATED

***** UNRESOLVED ENTRIES
NONE

******UNRESOLVED ENTRIES (list of unresolved linkage symbols and calls) One of the relocation indicators on the card

The relocated load address plus the size of the common data area is greater than the machine's highest addressable location.

The symbol in the operand (for a control card) or used as the load address (for a load card) has not been defined.

The Linkage Loader symbol table has been exceeded.

This message indicates that the subprogram named in the SNAP card has not been processed. The message can also indicate that columns 56 and 57 of the SNAP card were blank.

After relocation, the load address in the printed card is in the Resident Monitor area.

This message indicates that (1) the Linkage Loader put no output on the Job file, (2) the Linkage Loader read other subprogram cards before reading a Phase or TITLE card, or (3) the first Phase card read did not contain an operand.

All linkage symbols and calls were resolved.

All linkage symbols and calls were resolved except the ones listed. The format of the list of entries is:

XXXXXLINKSYMBOL

or

XXXXXLINKSYMBOL

Where xxxxx = location where definition of symbol or call was needed

LINKSYMBOL = linkage symbol that was undefined

The plus zone over the low-order position of xxxxx indicates that the entry was an imbedded call.

If xxxxx contains blanks, the subprogram call was made in a Linkage Loader control card.

ACTION

The Linkage Loader will end execution with a special end of program.

If the sc mode is on, this is a warning message; Linkage Loader processing is continued. If the sc mode is off, the Linkage Loader will end execution with a special end of program.

This is a warning message. Linkage Loader processing is continued.

The Linkage Loader will end execution with a special end of program.

Note: The user should reorder the program deck so that the number of forward references of linkage symbols is reduced.

The Linkage Loader ignores the Snapshot request and continues processing.

The Linkage Loader will end execution with a special end of program.

The Linkage Loader will end execution with a special end of program.

The Linkage Loader continues processing.

The Linkage Loader continues processing.

Console Inquiries

Console inquiries are used to communicate control information to various portions of the System Monitor. The messages that can be entered through the console are divided into three classes, as follows:

- A. Messages that can be accepted by the Resident Monitor during the execution of a dependent program.
- B. Messages that can be accepted by the Transitional Monitor between jobs.
- C. Messages that are communicated to the Transitional Monitor in reply to a request from the Transitional Monitor for specific control information.

The messages are communicated to the System Monitor by:

- 1. Depressing the Inquiry Request key.
- 2. Typing in the message.
- 3. Depressing the Inquiry Release key.

Class A Messages

The following messages can be accepted during execution of a dependent program. Unless otherwise stated, the entire message consists of the three characters listed.

MESSAGE	MEANING	ENTRY TIME	REMARKS
\$10	This message causes intermediate entry to the /UEP/ point of the Resident Monitor's End-of-Program routine. This message cannot be used for programs under control of the Tele-Processing Supervisor.	ecution of a depend-	The operator can also branch to the /UEP/ entry by pressing the Computer Reset key and then Start key. However, this procedure resets certain machine indicators. Therefore, records on the Core Image file may not reflect the status of storage at the time the dependent program failed. Note: The Reset and Start key procedure should not be used in a Tele-Processing system.
\$20	This message requests the Transitional Monitor to notify the operator when it can accept class B messages.		
\$3x (any input wanted)	This message causes the Resident Monitor to place x in /MCI/. The entire message is put in the input area whose high-order position is designated by /RIQ/.	In answer to user- specified message or any time during exe- cution of a dependent program	This message is used to provide console input to the dependent program.
\$50	This message causes the Resident Monitor to exit from its Wait-Loop routine and return control to the dependent program.	Normally after a user-specified message	For a fortran program, the user message will be pause.
\$70	This message signals the Resident Monitor to initiate immediate restart procedures. (See the <i>Operator's Guide</i> publication.)	Any time during execution of a program that is requesting checkpoints and can be restarted	
\$8x	This message, which has three forms $(x = A, R, \text{ or } C)$, is described in the Operator's Guide publication.	In reply to a 20101, 30101, or 30102 message from the Resident IOCS	
\$90b (any input of up to 16 characters)	This message is used to communicate information to the Tele-Processing system. (See the <i>Tele-Processing Supervisor</i> publication.)	Any time	

Class B Messages

The following messages can be accepted by the Transitional Monitor. When class B messages can be accepted, the Transitional Monitor notifies the operator by the console message: 20501 ENTER B MESSAGES. Class A messages can also be accepted at this time. This message is written if the operator previously entered the

\$20 inquiry, and each time an END card is read from the Standard Input Unit or Alternate Input Unit. The Transitional Monitor enters a waiting loop after writing this message.

Unless otherwise stated, each of the following messages consists entirely of the characters listed.

MESSAGE	MEANING
\$B1	This message causes the Resident Monitor's Read routine to be altered to read from the Alternate Input Unit.
\$B2	This message causes the Transitional Monitor to close, rewind, and unload the Standard Print Unit tape.
\$B3	This message causes the Transitional Monitor to close, rewind, and unload the Standard Punch Unit tape.

EMARKS

Never after 20501 enter B messages following an end siu message.

After the tape is unloaded, the operator mounts another tape on the unit.

After the tape is unloaded, the operator mounts another tape on the unit.

Note: This message is used only if the Standard Punch Unit is not assigned to the same tape as the Standard Print Unit. If both units use the same tape, then the \$B2 message is used for the closing function.

MESSAGE \$B4
\$B5ss
\$B6ss

\$BX

for use.

This message indicates to the Transitional Monitor that the physical unit represented by assignment symbol ss is now available for use. The Transitional Monitor removes the unavailable indication it had set in response to a \$B5ss message for this physical unit.

This message causes the Transitional Moni-

This message indicates to the Transitional Monitor that the physical unit represented by assignment symbol ss is currently unavailable

tor to close, rewind, and unload the Core Image

This message indicates that the operator has no more class B messages at this time. It causes the Transitional Monitor to exit from the waiting loop it had entered to allow console inquiries. This message must be given to enable the Transitional Monitor to resume processing.

REMARKS

After the tape is unloaded, the operator mounts another tape on the unit.

After the \$BX message is entered, the Transitional Monitor completes functions related to previous class B messages. For example, the complete procedure for use of a \$B2 message is:

MEANING

file tape.

- 1. The Transitional Monitor types out 20501 ENTER B MESSAGES to notify the operator it is ready to accept class B messages and then enters a waiting loop.
 - 2. The operator enters the \$B2 message.
- 3. The Transitional Monitor performs the closing functions for the Standard Print Unit and returns to the waiting loop.
- 4. The operator mounts another tape for the Standard Print and enters the \$BX message.
- 5. The Transitional Monitor opens the file for the new tape and resumes other processing related to preparation for the next job.

When the \$BX message is entered in answer to a 20501 ENTER B MESSAGES following an END SIU message, the Transitional Monitor resumes reading and processing from the Standard Input Unit and without initialization. Therefore, it will ignore any daily information, such as a DATE card, and will skip to the next JOB card.

Class C Messages

The following message is given in reply to a message from the Transitional Monitor.

MESSAGE

MEANING

\$C1ss

This message specifies that the assignment symbol ss is to be substituted for the one contained in the ASGN card. The substituted physical unit must be currently available.

Note: The \$B6ss message cannot be used at this time to indicate that the unit specified on the ason card is now available; it must have been entered before the ason card was read.

ENTER IN REPLY TO

20502 SUB FOR SS (This message indicates that an ASGN card specified an assignment symbol, ss, for a physical unit that is not available.)

Relocatable subprograms on the System Library file or Go file can be corrected or changed by placing patch cards in the Standard or Alternate Input Unit after a CALLP card. The patching is performed by the Linkage Loader immediately after it processes the called subprogram.

Considerations for patching are:

- 1. Calls imbedded in a subprogram by DCWS, DCWF, and CALL statements cannot be negated through patching.
- 2. The functions of DEFIN, BASE1, BASE2, and PRTCT cannot be overriden through patching.
- 3. An instruction that contains a linkage symbol for an address can be patched only if the symbol in the original instruction has been resolved before the patch is processed.

Patch cards must be punched in the formats for relocatable object cards as described below.

All cards in a subprogram must have a card type punched in column 72. The card types are summarized in Figure 11.

Number in Column 72	Card Type
GOIGHHI 72	Curu Type
0	Load card with absolute location or linkage symbol as load address
1	Load card with upward relocation factor to be applied to load address
2	Load card with downward relocation factor to be applied to load address
3	Termination card for primary subprogram
4	DEFIN, BASE1, BASE2, PRTCT, or CALL card
5	TITLE card
8	Set word mark or record mark card
9	Termination card for secondary subprogram
w	Clear word mark card
Y	Clear storage between limits card
z	Random load card

Figure 11. Card-Type Summary

Column 71 and down of certain cards contain relocation indicators, which give to the Linkage Loader relocation information and characteristics of instruc-

tions and data in the card. The relocation indicators are summarized in Figure 12.

Note 1: Constants and instructions may not be on the same card.

Note 2: A linkage symbol, when used as an address, must indicate no relocation for that section of the instruction.

Relocatable Object Cards

TITLE Card

The format for the TITLE card is:

Column	Contents
16-20	TITLE
21-30	(1- to 10-character subprogram name, left-justified, with trailing blanks)
31-35	(Origin point to which subprogram was compiled. Usually 00000 or blanks; for FORTRAN programs, this field is 00001.)
41-45	(Total number of storage locations required by this
72	subprogram for a common data area)

Termination Card (for Subprograms)

The format for the subprogram Termination card is:

Column	Contents
1	E
2	Word separator character (0-5-8 punch)
3-7	(Entry point for primary subprogram, blanks for secondary subprogram)
72	3 (for primary subprogram) or 9 (for secondary subprogram)

The entry point address in columns 3 through 7 is the compiled address. It will be relocated by the Linkage Loader, using the upward relocation factor for the subprogram.

Load Card

The format for the load card is:

Coumn	Contents
1	Word separator character (0-5-8 punch)
2-6	(Load address: Low-core address of the area into which the characters in columns 13 and up are to be loaded)
7	Word separator character (0-5-8 punch)
8-10	Zeros
11-12	(Length in core storage. Number of characters to be loaded from this card.)
13-up	(Instructions or constants to be loaded into storage, left-justified)
	Note 1: To load a word mark into storage, punch a word separator character in the column pre- ceding the character with which the word mark is to be associated. These word separator char-

Column	Contents acters are not included in the count columns 11 and 12. To enter a word separator character into storage, punch word separator characters in two adjacent columns. Add only "1" to the count.
	Note 2: Instructions, down calls, and down calls may appear on the same card, but cannot appear on a card containing constants.
71-down	(Relocation indicators. The indicator punched in column 71 applies to the first instruction or constant in the card, the indicator in column 70 applies to the second instruction or constant, etc. Relocation indicators are given in Figure 12.
72	 0 (for load card with absolute location or ABCD/linkage symbol as load address. If the load address is a linkage symbol, the symbol must have been defined before the Linkage Loader reads this card.) 1 (for load card with upward relocation factor to be applied to numeric load address.) 2 (for load card with downward relocation factor to be applied to numeric load address.)

DCWS and DCWF calls may be included in the instructions in columns 13 and up. To include these calls, punch a word separator character, followed by the 1-to 10-character subprogram name, left-justified, with trailing blanks in a 10-position field. The factor to be added into the count in columns 11 and 12 is seven for a DCWF.

Linkage symbols appearing as an address are given a NO relocation indicator.

DEFIN Card

The format for the DEFIN card is:

1110 10111	
Column	Contents
6-15	(5-character or 1- to 10-character linkage symbol, left-justified, with trailing blanks)
16-20	DEFIN
21-30	(5-character numeric address or 1- to 10-character linkage symbol, left-justified, with trailing blanks. For a numeric address, index register tags in the tens and hundreds positions are permissible. If a linkage symbol is used, its value must have been defined before the Linkage Loader reads this card.)
71	Blank (Relocate upward) P (Relocate downward) Q (No relocation) Note: These relocation indicators are for the address in columns 21 through 30. The Q relocation indicator is required if a linkage symbol is used in columns 21 through 30.
72	4

BASE1 Card

The format for the BASEl card is:

1110 1011.		
Column	Contents	
16-20	BASEl	
21-30	(Left-justified linkage symbol, absolute address,	,
	+X00, or blank)	
79	4	

If a linkage symbol is used, it must have been defined before the Linkage Loader reads this card.

	Without d-Modifier		With d-Modifier			
Description of Relocation	Indicator	Card Code for Indicator	Length in Core Storage	Indicator	Card Code for Indicator	Length in Core Storage
No address	9	9	1	1	12-9	2
No address X-control operation, B-address upward	ĹĹ	11-1	10			
X-control operation, B-address downward	K	11-2	10			
X-control operation, no relocation for B-address	1 1	11-3	10			
X-control operation, no B-address	M	11-4	5		İ	ì
Single address, upward	@	4-8	6		12-4-8	7
Single address, downward		5-8	6	1 7	12-5-8	7
	>	6-8	6		12-6-8	7
Single address, no relocation	Ó	0	11	9	12-0	12
Two addresses, no relocation	i	1	lii	À	12-1	12
Two addresses, only B-address upward, no relocation for A-address	'	'	1 ''			
Two addresses, only B-address downward, no relocation	2	2	11	В	12-2	12
for A-address	_	_				
Two addresses, only A-address upward, no relocation for	3	3	11	C	12-3	12
B-address						İ
Two addresses, both upward	4	4	11	D	12-4	12
Two addresses, A-address upward, B-address downward	5	5	11	E	12-5	12
Two addresses, A-address opward, B-address downward Two addresses, only A-address downward, no reloca-	6	6	11	F	12-6	12
tion for B-address						
Two addresses, A-address downward, B-address upward	7	7	11	l G	12-7	12
	8	8	1 11	і н	12-8	12
Two addresses, both downward Address constant, upward	N	11-5	5			ļ
·	P	11-7	5			1
Address constant, downward DCWS	s	0-2	7			1
	Q	11-8	5			
Address constant, no relocation Constant	blank	none				- 1
DCWF	T	0-3	5		1	1

Figure 12. Relocation Indicators

BASE2 Card

The format for the BASE2 card is:

Column Contents 16-20 BASE2

21-30 (Left-justified linkage symbol, absolute address, or blank)

72 4

If a linkage symbol is used, it must have been defined before the Linkage Loader reads this card.

PRTCT Card

The format for the PRTCT card is:

Column Contents
16-20 PRTCT
21-30 (Left-justified linkage symbol, absolute address, or blank)
72 4

If a linkage symbol is used, it must have been defined before the Linkage Loader reads this card.

CALL Card

The format for the CALL card is:

Column Contents
16-20 CALL
21-30 (1- to 10-character subprogram name, left-justified, with trailing blanks)
72 4

Random Load Card

The format of the Random Load card is:

Cotumn	Contents
1	Word separator character (0-5-8 punch)
2-6	(Load address: low-order address of area into
	which the characters in columns 7 through 11
	are to be loaded)
7-11	(Five characters to be loaded in Move mode, be-
	ginning at the location specified in columns 2
	through 6. These five characters are considered
	to be an address and will thus be relocated.
	Loading the five characters will not affect any
	word marks set by previous load cards.)
12-22	(Same as columns 1 through 11)
23-33	(Same as columns 1 through 11)
34-44	(Same as columns 1 through 11)
45-55	(Same as columns 1 through 11)
67	(Relocation indicator for columns 45 through 55)
68	(Relocation indicator for columns 34 through 44)
69	(Relocation indicator for columns 23 through 33)
70	(Relocation indicator for columns 12 through 22)
71	(Relocation indicator for columns 1 through 11)
72	\mathbf{Z}

The relocation indicators to be used in columns 67 through 71 are:

6		
RELOCATION INDICATOR	LOAD	FIVE CHARACTERS
INDICATOR	ADDRESS	TO BE LOADED
0	No relocation	No relocation
1	No relocation	Relocate upward
2	No relocation	Relocate downward
3	Relocate upward	No relocation
4	Relocate upward	Relocate upward
5 .	Relocate upward	Relocate downward
6	Relocate downward	No relocation
7	Relocate downward	Relocate upward
8	Relocate downward	Relocate downward

These relocation indicators are also given in Figure 12.

If word marks are wanted over the five characters to be loaded, they must be set before the Random Load card is loaded into storage for program execution.

If less than five fields are needed, the relocation indicator corresponding to the first unused field must be left blank.

The load address and/or the five characters to be loaded may be an ABCD/ format linkage symbol. The corresponding relocation indicator must specify NO relocation.

Clear Storage Card

The format for the Clear Storage card is:

Column	Contents
1	Word separator character (0-5-8 punch)
2-6	(Lowest address of area to be cleared)
7	Word separator character (0-5-8 punch)
8-12	(Highest address of area to be cleared)
71	N (Relocation indicator when the addresses in col-
	umns 2 through 6 and 8 through 12 are to be relocated upward)
	P (Relocation indicator when the addresses are to
	be relocated downward)
72	Y

Set Word Mark or Record Mark Card

The format for the Set Word Mark or Record Mark card is:

Column	Contents
1	Word separator character (0-5-8 punch)
2-6	(Address of first word mark or record mark to be set)
7	Word separator character (0-5-8 punch)
8-12	(Increment factor, in storage locations, used to determine the location of the second, and succeeding, word marks or record marks to be set)
13	Word separator character (0-5-8 punch)
14-18	(Number of word marks or record marks to be set)
20	Record Mark (0-2-8 punch). (If this card is to set record marks.)
	Blank (If this card is to set word marks.)
71	N (Relocation indicator, when the address in col- umns 2 through 6 is to be relocated upward)
	P (Relocation indicator, when the address in col-
	umns 2 through 6 is to be relocated downward)
72	8

Clear Word Mark Card

The format for the Clear Word Mark card is:

Column	Contents
1	Word separator character (0-5-8 punch)
2-6	(Address of first word mark to be cleared)
7	Word separator character (0-5-8 punch)
8-12	(Increment factor, in storage locations, used to de-
	termine the location of the second, and succeed-
	ing, word marks to be cleared)
13	Word separator character (0-5-8 punch)
14-18	(Number of word marks to be cleared)
71	N (Relocation indicator when the address in col-
	umns 2 through 6 is to be relocated upward)
	P (Relocation indicator when the address in col-
	umns 2 through 6 is to be relocated downward)
72	W

Operation of the System Monitor

Operation of the System Monitor is divided into two categories:

- 1. *Initialization* to prepare the Resident Monitor for work.
- 2. Functioning of the System Monitor to control the Operating System.

The System Monitor also enables restarting dependent programs from checkpoints.

Initialization of the System Monitor

Initialization of the System Monitor begins with bringing the Resident and Transitional Monitors into core storage from the System Operating File. They are brought into storage by the Bootstrap routine, which is the first element on the System Operating File. The Bootstrap routine is also used if reinitialization of the System Monitor becomes necessary because of an emergency. (For example, a dependent program might fail in such a way that portions of the Resident Monitor are destroyed.)

The Initialization routine is brought into storage with the Resident Monitor. It is given control to perform the housekeeping needed to prepare the Resident Monitor. For this housekeeping, the installation supplies the Resident Monitor with *daily information* control cards. These cards are the Monitor DATE card and ASGN cards for certain files used by the System Monitor.

Reinitialization procedures include facilities for repositioning the Standard or Alternate Input Unit, if tape, to the next job specified by the operator.

For full details on initialization and reinitialization, see the *Operator's Guide* publication.

Functioning of the System Monitor

Example 1

To illustrate functioning of the System Monitor, assume that a program is to be compiled, relocated, and executed. Control cards for such a program are shown in Figure 13. These cards are in a card reader Standard Input Unit.

Both the Resident Monitor and Transitional Monitor are left in storage by initialization. During initialization, the Transitional Monitor had just read and processed the daily information cards on the Standard Input Unit. When the Transitional Monitor reads the first JOB card, Monitor functioning begins.

Following the JOB EXAMPLE card are COMT, ASGN, and MODE cards. Note that for the language processors the MODE GO card is placed before the EXEQ cards. The Transitional Monitor processes these cards, then reads the first exeq autocoder card. It locates the Autocoder language processor on the System Operating File and branches to the Resident Monitor's Load routine with a request to load the Autocoder processor. A Clear Storage routine from the System Operating File is loaded immediately above the end of the Resident Monitor. The Clear Storage routine clears all of storage above the Resident Monitor, thus destroying the Transitional Monitor, except for the first 99 positions above the Resident Monitor and the last two positions at the top of core storage. After storage is cleared, the Load routine brings in the Autocoder processor.

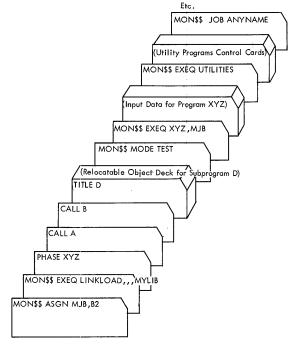
The Autocoder processor reads its source statements from the Standard Input Unit. It uses three Work files, assigned by the daily information, as temporary storage during the compilation. The processor places its output (relocatable card-image records) on the Go file, because MODE GO was specified, and also punches out a relocatable object deck on the Standard Punch Unit. Use of the tape Go file eliminates the operator intervention that would have been required with only punched output.

When the Autocoder processor is finished, it gives control to the Resident Monitor's End-of-Program routine at the /EOP/ entry point. The Resident Monitor brings in the Transitional Monitor. The Transitional Monitor resumes reading the Standard Input Unit. The next card is the second EXEQ AUTOCODER card. For this card, the System Monitor repeats the procedure performed for the first EXEQ AUTOCODER card.

The Autocoder processor must be executed twice, because the source statements for the two subprograms are on different files.

In its second execution, the Autocoder processor places its output on the Go file after the output for subprogram A.

When the Autocoder processor is finished, it returns control to the Resident Monitor, which brings in the Transitional Monitor. The Transitional Monitor reads the Standard Input Unit, assigning the Job file (мјв) to physical unit в2.



NOTE; Subprogram B Source Statements are on Tape MR1

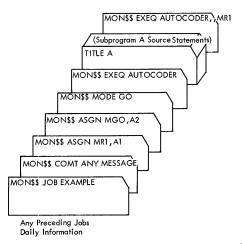


Figure 13. Control Card Deck for First Example

Then the Transitional Monitor, for the EXEQ LINK-LOAD card, repeats the procedure performed for the language processor's EXEQ cards to bring the Linkage Loader into storage.

In this example, the Linkage Loader reads the CALL cards for subprograms A and B and puts the calls on the request list. Then it reads and processes subprogram D. The Linkage Loader continues reading and finds a Monitor control card. The Monitor control card forces the Linkage Loader to read in and process all subprograms on the request list. Thus, the Linkage Loader calls in and processes the newly compiled subprograms A and B from the Go file. Subprogram A contains an imbedded call for subprogram D, which has already been processed. Subprogram B contains

an imbedded call for subprogram C, which is on a Library named MYLIB on the System Operating File. The fourth operand of the EXEQ LINKLOAD card designates MYLIB as the System Library file for this Linkage Loader run. The Linkage Loader, when it processes subprogram B, puts subprogram C on the request list. Because no more Linkage Loader control cards exist, the Linkage Loader then calls in subprogram C from the Library.

The memory map for these subprograms, which form program xyz, will be:



Note that A was before B on the Go file, because A was compiled before B.

The Linkage Loader puts program xyz, in absolute format, on the Job file. If the installation wants to keep using this program in absolute format, the operator should save the Job file tape reel after the program is executed. This reel can later be mounted on a physical unit that has been assigned to the Job file.

When the Linkage Loader is finished, it returns control to the Resident Monitor, which brings in the Transitional Monitor. The Transitional Monitor reads the MODE card. Because two of the subprograms are newly compiled, MODE TEST is specified before the program is executed. The last subprogram to be executed ends with B /UEP/ in order to return control to the Resident Monitor's End-of-Program routine at the /UEP/ entry point.

When the Transitional Monitor reads the EXEQ XYZ card, it repeats the procedure used to bring in the language processors and Linkage Loader, except that it loads phase 001, named xyz, from the Job file. Note that dependent programs, whether IBM-provided or user-written, are given the same processing by the System Monitor.

In this example, subprogram B was the primary subprogram. Therefore, processing control is initially given to the entry point for subprogram B.

The program, during execution, reads input data from the Standard Input Unit through the Resident Monitor's Read routine.

If the program fails, the operator gives control to the Resident Monitor's End-of-Program routine through the /UEP/ entry point. If the program executes successfully, it also gives control to the /UEP/ entry point. In either case, the End-of-Program routine will write the contents of storage on the Core Image file, and the Resident Monitor will bring in the Transitional Monitor.

The Transitional Monitor, because the unusual end of program occurred in Test mode, will read the Standard Input Unit for any Monitor control cards, rather than skipping to the next Job card. If the program fails before all input data cards are read, the Transitional Monitor will skip past them and resume processing when it finds the first Monitor control card. Thus, the Transitional Monitor will read the EXEQ UTILITIES card. It will bring the Utility programs into storage from the System Operating File in the same way it brought in the other dependent programs.

When the Utility programs are finished, they will return control to the Resident Monitor, which will bring in the Transitional Monitor. The Transitional Monitor, upon resuming reading from the Standard Input Unit, will read the JOB ANYNAME card. It will close out the

previous job, named EXAMPLE, and reset the Resident Monitor for the new job.

The above example shows just one of the many job combinations that are possible. For example, the subprograms compiled could form a user-written language processor. This new processor could be relocated by the Linkage Loader and executed to compile a program. The newly compiled program could then be relocated and executed.

The example showed execution of only one program. The job could have included execution of many programs.

Example 2

The following example shows the complete control card deck for a batch. The machine configuration for this example is shown in Figure 14. Figure 15 illustrates the control card deck for this example. Figure 16 presents a chart of input/output assignments as they are made from the ASGN cards in the control card deck.

The following comments give information concerning the makeup of the control card deck and explain the changing input/output assignments summarized in Figure 16.

CARDS 1-8

These cards constitute the daily information for initialization of the System Monitor. In addition to supplying the current date, these cards assign input/output units

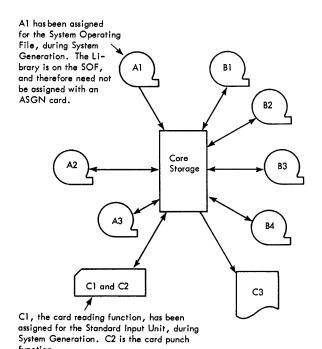


Figure 14. Machine Configuration for Second Example

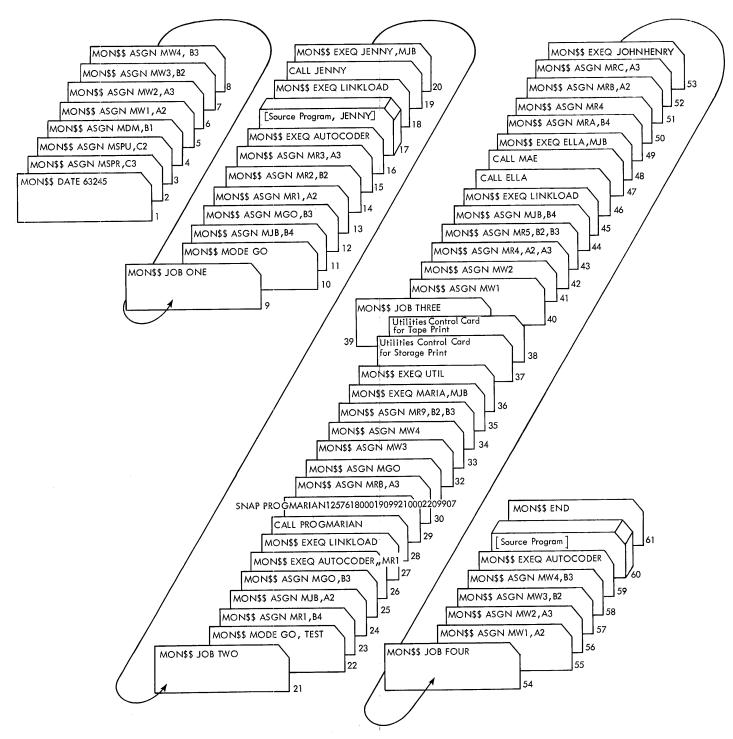


Figure 15. Sample Control Card Deck for Second Example

Units	Initial – ization	First . Job	Second Job	Third Job	Fourth Job
Al	SOF	SOF	SOF	SOF	SOF
A2	MW1	MW1 and MR1	MW1 and MJB	1. MW1 cancelled 2. MR4 (base unit) 3. MR4 cancelled 4. MRB	MW1
A3	MW2	MW2 and MR3	MW2 and MR8	1. MW2 cancelled 2. MR4 (alternate unit) 3. MR4 cancelled 4. MRC	MW2
В1	MDM	MDM	MDM	MDM	MDM
B2	MW3	MW3 and MR2	1. MW3 2. MW3 cancelled 3. MR9 (base unit)	MR5 (base unit)	WW3
B3	MW4	MW4 and MGO	1. MR4 and MGO 2. MR4 and MGO cancelled 3. MR9 (alternate unit)	MR5 (alternate unit)	MW4
B4	un- assigned	MJB	MW1 (source program tape for Autocoder)	MJB and MRA	unas- signed
Cl	SIU	SIU	SIU	SIU	SIU
C2	SPU	SPU	SPU	SPU	SPU
СЗ	SPR	SPR	SPR	SPR	SPR

Figure 16. Assignments of Input/Output Units for Second Example

for the Standard Print Unit, the Standard Punch Unit, the Core Image file, and the language processors' Work files.

CARDS 9-20

These cards define the first job of the batch. The job consists of a compile-and-go operation for a single subprogram.

CARDS 21-38

These cards define the second job. This job consists of a compile-and-go operation for a single subprogram to be tested. The Snapshot program is to be incorporated into the subprogram. The Storage Print and Tape Print programs are to be executed after execution of the subprogram. Note that for the compilation, the source statements are to be mounted on tape physical unit B4. Also note that the test program will use tape physical units B2 and B3 as base and alternate Reserve files. Since B2 and B3 have been previously assigned as base units only, the previous assignments must be canceled.

CARDS 39-53

These cards define the third job. This job consists of the execution of the Linkage Loader to combine two subprograms from the Library, the execution of the resulting program, and then the execution of one of the installation's programs contained on the System Operating File.

Cards 40 and 41 cancel the assignment of two Work files to permit A2 and A3 to be used as base and alternate Reserve files. Note that the assignment made by card 43 is permissible because the last effective assignment for B2 and B3 was for a Reserve file, and that assignment was automatically canceled between jobs. Card 50 cancels the assignment made by card 42, permitting A2 and A3 to be used as base units for the next program within the job.

CARDS 54-60

These cards define the fourth and last job in the batch. This job consists of a single compilation. It is included in the sample batch to illustrate the need for recording the changing input/output assignments, so that canceled assignments for symbolic units can be restored when those symbolic units are again required. In this job, the four Work files assigned during initialization must be assigned again, since during the course of the preceding jobs, all the Work file assignments have been canceled (cards 32, 33, 40, and 41).

Restarting from a Checkpoint

The System Monitor provides facilities to restart certain programs that were discontinued because of certain error conditions recognized by the operator or because a program of higher priority, to be loaded from the Alternate Input Unit, required the machine. Also, the processing of some programs requires so much time that the processing is performed in installments.

Note: Only programs that request checkpoints on the Core Image file can use the restart facilities of the System Monitor.

The operator discontinues programs (that have taken checkpoints and will be restarted) by using the \$10 or \$3x console inquiry. The \$3x inquiry can be used only if the program recognizes it as a request to discontinue at some convenient point (checkpoints were taken previously) or as a request to take a checkpoint and then discontinue.

Restarting is described in detail in the *Operator's Guide* publication.

Immediate and Delayed Restarts

The Resident IOCS provides a routine for writing checkpoint records on the Core Image file. (See the *Basic* Input/Output Control System publication for details concerning the creation of checkpoint records.) These records consist of the information in core storage at the time the checkpoint is taken. From these records the System Monitor can restart a program in one of two ways:

Immediate Restart: An immediate restart can be effected by entering the \$70 message through the console printer. This message causes the Resident Monitor to bring the Transitional Monitor into storage. The Transitional Monitor then executes a Restart routine, which locates the most recent checkpoint on the Core Image file and initiates a restart of the dependent program from that checkpoint. The major functions performed for the immediate restart are as follows:

- 1. Reloading the dependent program into core storage.
- 2. Restoring areas of the Resident Monitor related to the status of the dependent program at the time the checkpoint was taken.
- 3. Repositioning some tape files that were open at the time the checkpoint was taken.
- 4. Resuming execution of the dependent program at the point at which the checkpoint was taken.

Delayed Restart: A delayed restart is effected by initialization of the System Monitor. At the time of an initialization for the purpose of a delayed restart, the operator supplies control information specifying the

particular checkpoint from which the dependent program is to be restarted. (Note the difference from the immediate restart, which uses only the most recent checkpoint.)

The functions performed for the dependent program during a delayed restart are essentially the same as those performed during an immediate restart. Care must be exercised, however, that the configuration at the time the checkpoint was taken is duplicated for the restart procedures.

Restart Conditions

The following considerations apply to programs using the checkpoint and restart facilities of the System Monitor:

- 1. No repositioning can be performed for unitrecord equipment.
- 2. Programs that update records in disk storage cannot be effectively restarted, since there is no way for the restart procedure to restore disk records that have been changed.
- 3. Tape files that are closed at the time of a checkpoint cannot be repositioned during a restart from that checkpoint. (This is no problem if the tape file was rewound when it was closed.)
- 4. Programs operating under control of the Tele-Processing Supervisor cannot use the checkpoint and restart facilities.

Index

Absolute format (see format)		Communication symbol	8, 11
Absolute Zero	14	(also see linkage symbol and system symbols	;)
Addresses		Undefined	9
Load	14, 25, 53, 58	Compilation (see language processors)	
Reference	14	Compile-and-go operation	. 7, 15, 25, 39, 46
Relocatable (see format, relocatable)		COMT card (see Monitor control cards)	
AIU (see System files)		conco card (see Linkage Loader control cards)
Alternate Input Unit (see System files)		Console inquiries	7, 12, 47, 50, 54
Alternate unit (see physical unit)		Class A messages	50, 54
'AMS/ field (see Communication Region)		\$10	
Arithmetic overflow indicator	37	\$20	55
ASGN card (see Monitor control cards)		\$3x (any input wanted)	27, 32, 55, 64
Assignment, cancellation	39, 45	\$50	27, 33, 55
Assignment routines	43	\$70	
Assignment symbols	39, 43, 51, 55	\$8x	
Autocoder (see language processors)		\$90b (any input up to 16 characters)	55
Backspace System Tape subroutine	34	Class B messages	42, 50, 54
(see also linkage sequences)		\$B1	
BASE1 card (see Linkage Loader control cards)		\$B2	
BASE2 card (see Linkage Loader control cards)		\$B3	
Base unit (see physical unit)		\$B4	
Base Zero	14, 21, 27	\$B5ss	43, 56
(see also Communication Region, /ORG/ field)	, ,	\$B6ss	43, 50, 56
Batch processing	7, 42, 62	\$BX	
Bootstrap routine	12, 52, 60	Class C messages	
CALL card (see Linkage Loader control cards)		\$C1ss	50, 50
CALLN card (see Linkage Loader control cards)		Console Inquiry routine	11, 32
CALLP card (see Linkage Loader control cards)		(see also linkage sequences)	20 20 41 47 5
Calls	15, 20, 54	Console printer	32, 39, 41, 47, 35
CALL, CALLN, and CALLP cards	, ,	Constants	37, 30
(see Linkage Loader control cards)		Control Card Interpretation routine	11
Imbedded 7, 11	, 15, 19, 54, 57	Core Image file (see System files)	
Unresolved		/CRD/ field (see Communication Region) Cycling of physical units	11
Cancellation of assignments (see assignment, cance	ellation)		
Card (defined)		Daily information	42, 44, 46, 56, 60
Card-image format (see format)		Data, input	
Card type	57	DATE card (see Monitor control cards)	
Carriage-control character	27, 30	/DAT/ field (see Communication Region)	
/cco/field (see Communication Region)		DCWF card	8, 15, 57, 58
Channel Status Character	27, 30, 33, 35	DCWs card	8, 14, 57, 58
Checkpoint13	, 51, 55, 60, 64	defin card	
Class A messages (see console inquiries)		(see Linkage Loader control cards)	
Class B messages (see console inquiries)		Define the File statement	43, 45, 46
Class C messages (see console inquiries)	• .	Definition cards	
Clear Storage card (see Linkage Loader control ca		(see DEFIN card under Linkage Loader contr	rol cards)
Clear Storage routine		Dependent programs 6, 10, 27	, 37, 41, 54, 62, 64
Clear Word Mark card (see Linkage Loader contr	ol cards)	Diagnostic messages (see messages)	
COBOL (see language processors)	14 04 24 20	DISGO card (see Linkage Loader control cards)	
Common data area		Divide overflow indicator	
Communication Region		DSKLIBLDR (see Monitor control cards, exeq ca	rd)
/AMS/ field			
/cgo/ field		END card (see Monitor control cards) End of file15	18 25 20 45 50
/CRD/ field	27,41	End of program	97 37 4
/DAT/ field		End-of-Program routine 10, 13, 27, 32	35 37 49 61 6
/IPI/ field /LIN/ field		(see also linkage sequences)	, 55, 51, 12 , 51, 0
/LIN/ neid /MCI/ field		Normal end of program	3
/MCA/ field		Special end of program	35. 36. 5
/MGA/ field		Unusual end of program	. 35, 39, 47, 49, 5
(see also Base Zero)	21, 11	End of reel	4
/PHN/ field	27, 33, 39	End of unit	4
/PNM/ field		END statement	
/siz/ field		Entry point	15, 18, 20, 5
/TPB/ field		/EOP/ (see system symbols)	. , ,

Errors, input/output (see Resident 10cs and Channel Status Character)	Set Word Mark or Record Mark card 59 Termination card 58
Exceeding storage 25	TITLE card
EXEQ card (see Monitor control cards)	Linkage sequences
Format	to Backspace System Tape subroutine
Absolute 6, 11, 46	to Console Inquiry routine
Card-image 12	to End-of-Program routine 27, 35
of cards (see individual cards)	to IBLOOKM routine
Relocatable 6, 11, 24, 57	to Load routine
FORTRAN (see language processors)	to Modification of Communication Region routine
Go file (see System files)	to Punch routine 27, 30 to Punch routine 27, 31
go mode (see Monitor control cards, MODE card)	to Read routine 27, 31
Halts, programmed	to Read System Disk subroutine 27, 34
Header record 20	to Read System Tape subroutine 27, 33
Highest positions of core storage (see storage)	to Rewind System Tape subroutine
IBACCOUNT module 39	to Simple Search of System Tape subroutine 27, 34
(see also linkage sequences)	to Wait-Loop routine
(see also linkage sequences) IBMLIBR	Linkage symbol 8, 15, 21, 23, 47, 54, 57
Imbedded calls (see calls)	LINKLOAD (see Monitor control cards, exeq card)
Index registers	Listings, language processor
Initial entry point of subprograms	Load card
Initialization of Resident Monitor 27, 39, 41, 44, 50, 56, 60, 65	(see also Linkage Loader control cards)
Initialization routine	Load routine
Initialization status character	(see also linkage sequences)
INPUT card (see Linkage Loader control cards)	/LOD/ (see system symbols)
Input/Output Control System (see Resident 10cs)	/MBF/ (see system symbols)
Input/Output files	/MBX/ (see system symbols)
and Tele-Processing system files)	/MCI/ field (see Communication Region)
Installation programs (see dependent programs)	/MCR/ (see system symbols)
Instructions 57, 58	/MCs/ (see system symbols)
/IPI/ field (see Communication Region)	MDM (see System files)
iocs (see Resident iocs)	MDT (see System files)
Job	Memory map
(see also Monitor control cards, job card)	Initialization 51
joв card (see Monitor control cards)	BOOTSTRAP ERROR 01 52
Job counters (see job numbers)	BOOTSTRAP ERROR 02 52
Job file (see System files)	BOOTSTRAP ERROR 03
Job numbers 42, 51, 52 Job routine 11	BOOTSTRAP ERROR 04
	BOOTSTRAP ERROR 05
Labels, tape	BOOTSTRAP ERROR 06
Autocoder	BOOTSTRAP ERROR 07 53 DATE NOT ENTERED 51
COBOL 8, 13, 25, 33	DATE NOT ENTERED
FORTRAN 8, 14, 24, 33, 55, 58	ENTER DATE, 1/O ASSIGNMENTS, JOB NUMBERS 51
LIB (see System files)	ENTER RST NUMBERS 51
Library (see System Library file under System files)	ENTER SOF/SIU ASGN SYMB
/LIN/ field (see Communication Region)	EOF SIU/AIU IN INIT
Linkage Loader	INIT ASGN EXCEED TABLE
General 6, 10, 12, 27, 36, 39, 41, 44, 48	(ASGN card image) INVALID ASSIGNMENT
Functions 11, 45, 47, 57, 61	INV INITIAL CHAR
Linkage Loader control cards 7, 15, 41 Codable by programmers	io or in 52
BASE1 card	IO OR JN RE ENTER XXX 52 JOB SEARCH OUT OF STEP 52
BASE2 card	MUST ENTER SOF/SIU ASGN SYMB 52
CALL card	SHARE VIOLATION, MONITOR COMPILATION 52
CALLN card	00202 53
CALLP card	00438 53
congo card	10528 SPU-SPR SAME UNIT
DEFIN card	10533 siu u/r 53
DISGO card 15, 18, 25	10534 AIU U/R
INPUT card 15, 25, 53	Linkage Loader
PHASE card	INPUT CARD INVALID 53
SNAP card	/LIB/ HEADER NAME UNEQUAL TO EXEQ CARD 53
Produced by language processors	SOF LIBRARY NAME UNEQUAL TO EXEQ CARD
Clear Storage card	***** BAD INPUT CARD (card contents) 53
Clear Word Mark card 59	***** BAD LOAD ADDR (card contents) 54
Load card 58	***** CORE EXCEEDED (card contents)
Random Load card 59	***** JOB FILE NOT CREATED

	54	Alternate
0111202 011221 11122 (111121)	54	Base 45
TIPLE O'LLIE TO !!	54	/PNM/ field (see Communication Region)
OTTELDOD TED TETTELD	54	Pow program 13
(list of unresolved linkage symbols and calls)	_,	Printer, console (see console printer)
	54	Print routine
Atobiaciit and Atamoracian incomment	47	(see also linkage sequences) Priority Alert mode 37
	50	Priority Alert mode 37 Priority Select On-Off Switch 37
END AIU	50	
END SIU		/PRT/ (see system symbols)
1147 11 1/250	50	PRICT card (see Linkage Loader control cards)
SWITCH TO REAL SEC W/O MINERAL SECTION	50	Punch routine
ODIA	47	(see also linkage sequences)
00001 001 pinterona not receive	48	Random Load card (see Linkage Loader control cards)
0000= 1.1,0 = 1.11=	48	Read error logging
00000 21111222 2121221 2122 2122	48	Read routine
20100 011 11111	51	(see also linkage sequences)
101011111	50	Read System Disk subroutine
100=0 000 22 111111111111111111111111111	48	(see also linkage sequences)
10021 000 11.72	48	Read System Tape subroutine
10022 201 01 1111	48	(see also linkage sequences)
100=1 =110 01 =111	48	Record mark 59
10022 010 01 1111	48	Reel counts
20020 110 1110	48	Reference address (see addresses)
	48	Reinitialization of Resident Monitor 44, 46, 49, 51, 61
10020 0110 0000	48	Relocatable format (see format)
100-0 0111-011	49	Relocatable subprograms (see format)
	49	Relocation 6, 14
10020 00 2011 2110	49	Downward
	49	No
10001 1101 100112 111 001	49	Upward 14
	49	Relocation factor
10000 11132 1111111111111111111111111111	49	Upward
20501 ENTER B MESSAGES		Downward 15, 24, 48
20502 SUB FOR SS 50,		Request list, Linkage Loader
	50	Relocation indicators
	50	Reserve files (MRX) 12, 39, 41, 44, 46
/MGA/ field (see Communication Region)		Resident 10cs 10, 27, 30, 31, 33, 35, 37, 39, 43, 55, 64
мсо (see System files)		Resident Monitor
мјв (see System files)		General 6, 10, 12, 25, 27, 37, 39, 41, 43, 60, 65
MLT (see System files)		Functions 10, 39, 42, 54
MODE card (see Monitor control cards)	07	Use of
Modification of Communication Region routine 11,	27	(see also linkage sequences and Communication Region)
(see also linkage sequences)	20	Restart
Monitor control cards		Rewind System Tape subroutine
Asgn card		(see also linkage sequences)
COMT card		/RIQ/ (see system symbols)
DATE card		/RSI/ (see system symbols)
END card		Run 7
EXEQ card		Set Word Mark or Record Mark card
Job card		(see Linkage Loader control cards)
MODE card	34	sc1 (see Monitor control cards, exec card)
MON\$\$ (see Monitor control cards)		sc2 (see Monitor control cards, exeq card)
MRX (see Reserve files)		sc mode (see Monitor control cards, MODE card)
MTX (see Tele-Processing system files)	22	Simple Search of System Tape subroutine
Multiphase programs 8, 10, 18, 20, 23,	33	(see also linkage sequences)
MWX (see Work files)		siu (see System files)
	20	/siz/ field (see Communication Region)
	27	SNAP card (see Linkage Loader control cards)
Normal end of program (see End-of-Program routine)		Snapshots
NSI (see next sequential instruction)		(see also Linkage Loader control cards, snap card)
/ORG/ field (see Communication Region)		sor (see System files)
Origin point of subprogram	48	SORTDEFINE (see Monitor control cards, exeq card)
Patches 11, 15, 20,		Sort Definition program 12
/PCH/ (see system symbols)		Source statements 41
Phase	47	Special end of program (see End-of-Program routine)
PHASE card (see Linkage Loader control cards)	•	spend statement
Phase header record	4 9	SPR (see System files)
Phase number 20, 27,	33	spu (see System files)
/PHN/ field (see Communication Region)		Standard Input Unit (see System files)
		Standard input Ont (see System files)
Physical unit	56	Standard Input Ont (see System files) Standard Print Unit (see System files)

Standard Punch Unit (see System files)	
Storage Two highest positions of core storage	
(see also Communication Region, /AMS/ field)	,
Subprogram	
Primary	
Secondary	Te
Symbolic names for symbolic swite 10.00	Te
(or input/output files)	Te
Symbolic unit	Te
Symbol table, Linkage Loader	16
40 47 40	Te
System Clas	Te.
Alternate Input Unit (Aug.) 19 97 35 27 20 41 44 46	
40 E1 EE E7 00	TES
Coro Imago 610 (1971) 12 26 20 41 44 46 49 51 55 64	TIT
Go fle (MGO) 11 13 15 18 25 27 26 20 44 48 57	To
Tob Blo (xem) 10 12 15 02 05 07 22 26 20 41 44 40 54	TP
Standard Input Unit (cm) 11 15 19 05 07 25 27	/TI
30 41 44 46 50 55 60	TP
Standard Print Unit (cpp) 19 97 30 41 44 46 59 55	TР
Standard Punch Unit (SPU) 12, 26, 39, 42, 44, 46, 53, 55	TPI
System I ibuses Cl. () 11 17 10 01 00 07 00	TPI
System Library file (LIB)	Tra
41, 44, 46, 53, 57	(
System Operating File (sor) 10, 12, 27, 33, 38, 41,]
44, 46, 48, 51, 60	1
Temporary Storage file (MDT)	/ui
1P 11Drary me (ML1) 44, 40	Un
System Generation 12, 27, 32, 38, 41, 43, 46, 52	Un
System Generator file 41	011
System Library file (see System files)	Un
System Monitor 6, 12	UT
System Operating File (see System files)	Uti
System symbols	
(see also Communication Region)	(
/EOP/ 27, 35, 36	Wa
/LOD/	(
/MD1/	/w
/MBX/ 27, 34	Wo
/MCR/ 27. 32. 36	Wo
/MCs/ 27, 29	W
	/ze
· · · · · · · · · · · · · · · · · · ·	/ZF
	ZF
	/ZF
	zr Zs
/ 021 / 21, 33, 41, 46	ZS

/WAT/	47, 33
/zbs/	27, 34
/ZRR/ 27, 3	33, 3 5
/zrs/ 27, 3	33, 35
/zrw/	27. 34
/zss/	
Tele-Processing system 12, 27, 32, 33, 35, 42, 46, 50,	
Tele-Processing system files (MTX)	16, 50
Temporary Storage file (see System files)	10, 00
Termination cards	. 7.8
(see also Linkage Loader control cards)	, .
Termination record	. 20
Testing programs	
TEST mode (see Monitor control cards, MODE card)	-,
TITLE card (see Linkage Loader control cards)	
Top of core storage (see storage)	
TPADLIBGEN (see Monitor control cards, exec card)	
/TPB/ field (see Communication Region)	
TPATLIBGEN (see Monitor control cards, exec card)	
TP Library file (see System files)	
TPRDLIBGEN (see Monitor control cards, exeq card)	
TPRTLIBGEN (see Monitor control cards, exeq card)	
Transitional Monitor	
General	30, 65
Functions 10, 37, 4	1, 54
Use of	39, 60
	39, 60
/UEP/ (see system symbols)	,
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines	. 37 10, 27
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines	. 37 10, 27
/UEP/ (see system symbols) Unit-record interrupts	. 37 10, 27
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines (also see Read routine, Punch routine, and Print routine Unusual end of program (see End-of-Program routine) UTILITIES (see Monitor control cards, EXEQ card)	. 37 l0, 27
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines (also see Read routine, Punch routine, and Print routine Unusual end of program (see End-of-Program routine)	. 37 l0, 27
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines (also see Read routine, Punch routine, and Print routine Unusual end of program (see End-of-Program routine) UTILITIES (see Monitor control cards, EXEQ card)	. 37 l0, 27
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines (also see Read routine, Punch routine, and Print routine) Unusual end of program (see End-of-Program routine) UTILITIES (see Monitor control cards, EXEQ card) Utility programs (also see Snapshots)	. 37 10, 27)
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines (also see Read routine, Punch routine, and Print routine Unusual end of program (see End-of-Program routine) UTILITIES (see Monitor control cards, EXEQ card) Utility programs 12, 36, 4	. 37 10, 27)
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines	. 37 10, 27)
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines (also see Read routine, Punch routine, and Print routine) Unusual end of program (see End-of-Program routine) UTILITIES (see Monitor control cards, exeq card) Utility programs (also see Snapshots) Wait-Loop routine (see also linkage sequences) /WAT/ (see system symbols) Word mark 27, 31, 37, 4	. 37 (0, 27) (2, 63 (33, 55
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines (also see Read routine, Punch routine, and Print routine) Unusual end of program (see End-of-Program routine) UTILITIES (see Monitor control cards, exeq card) Utility programs (also see Snapshots) Wait-Loop routine (see also linkage sequences) /WAT/ (see system symbols) Word mark 27, 31, 37, 4	. 37 (0, 27) (2, 63 (33, 55
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines (also see Read routine, Punch routine, and Print routine) Unusual end of program (see End-of-Program routine) UTILITIES (see Monitor control cards, exeq card) Utility programs	. 37 (0, 27) (12, 63) (13, 55) (11, 58) (17, 58)
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines (also see Read routine, Punch routine, and Print routine) Unusual end of program (see End-of-Program routine) UTILITIES (see Monitor control cards, exeq card) Utility programs	. 37 (0, 27) (12, 63) (13, 55) (11, 58) (17, 58)
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines (also see Read routine, Punch routine, and Print routine) Unusual end of program (see End-of-Program routine) UTILITIES (see Monitor control cards, exeq card) Utility programs (also see Snapshots) Wait-Loop routine (see also linkage sequences) /WAT/ (see system symbols) Word mark 27, 31, 37, 4 Word separator character Work files (Mwx) 11, 39, 41, 4 /ZBS/ (see system symbols)	. 37 (0, 27) (12, 63) (13, 55) (11, 58) (17, 58)
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines	. 37 (0, 27) (12, 63) (13, 55) (11, 58) (17, 58)
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines	. 37 (0, 27) (12, 63) (13, 55) (11, 58) (17, 58)
/UEP/ (see system symbols) Unit-record interrupts Unit-Record routines	. 37 (0, 27) (12, 63) (13, 55) (11, 58) (17, 58)

Reader's Comments

IBM 1410/7010 Operating System (1410-PR-155) System Monitor

Form C28-0319-1

From		
Name		
Address		
Your comments regarding the completeness, clarity, and accumulated will help us improve future editions. Please check the appropryour comments, and mail.		
	YES	NO
Does this publication meet the needs of you and your staff?		
Is this publication clearly written?		
Is the material properly arranged?	***************************************	
If the answer to any of these questions is "NO," be sure to elaborate.		
How can we improve this publication?	Please answer	below.
Suggested Addition (Page , Timing Chart, Drawing, Pro	ocedure, etc.)	
Suggested Deletion (Page)		
Error (Page)		
COMMENTS:		

FOLD

FOLD

BUSINESS REPLY MAIL NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

IBM CORPORATION
P.O. BOX 390
POUGHKEEPSIE, N.Y.

ATTN: PROGRAMMING SYSTEMS PUBLICATIONS
DEPARTMENT D9I

FIRST CLASS PERMIT NO. 81

POUGHKEEPSIE, N. Y

oLD -

FOLD

IBM

International Business Machines Corporation Data Processing Division 112 East Post Road, White Plains, N. Y. 10601 Printed in U.S.A.