HONEYWELL EDP

SOFTWARE MANUAL

MOD 1 (TR)

LIBRARY PROCESSORS C AND D

GENERAL SYSTEM: SERIES 200/OPERATING SYSTEM - MOD 1 (TAPE RESIDENT)

SUBJECT: Functional Characteristics and Operating Procedures for Library Processors C and D, Programs for Processing Macro Instructions and Inserting Macro Routines into the Symbolic Input Programs to Easycoder Assemblers C and D.

SPECIAL INSTRUCTIONS: This manual supersedes the software manual with the same title, dated February 25, 1966, and Addenda #1 and #2 thereto, dated July 6, 1966, and December 30, 1966, respectively.

DATE: May 5, 1967

FILE NO.: 123.1605.001C.2-051*

9676
8M
5567
Printed in U.S.A.

*Underscoring denotes Order Number.

FOREWORD

This software bulletin is a companion publication to the software bulletin entitled Easycoder Assemblers C and D, Order No. 041. The reader should be thoroughly familiar with the information presented in the above publication. The reader should also be familiar with the information presented in the Honeywell Series 200 Programmers' Reference Manual (Models 200/1200/2200/4200), Order No. 139 and the general operating characteristics of the applicable Series 200 equipment as explained in the various device manuals. For those concerned with a Model 120 system, the applicable reference manuals are the Series 200 Programmers' Reference Manual (Model 120), Order No. 141 and the Series 200 Equipment Operators' Manual (Model 120), Order No. 278.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

SECTION I
FUNCTIONS OF LIBRARY PROCESSORS C AND D

Library Processors C and D process symbolic card-image files that contain input to Easycoder Assembler C or D. Library Processor C is used prior to the assembly of a source program by Easycoder Assembler C; Library Processor D processes input to Easycoder Assembler D. The Library Processors insert macro routines which exist on a library symbolic program tape (SPT) into source programs in response to macro instructions (calls) within the source programs. In this process, each macro routine is specialized to perform the specific function desired. The only difference between Library Processors C and D is that Library Processor D accepts symbolic programs using the alternate card format which is also acceptable to Easycoder Assembler D. The alternate card format includes an 11-character location field. Library Processor C accepts only those symbolic programs using the standard (7-character location field) card format. Except where specifically noted in the text, all information in this manual pertains to both Library Processors.

A macro routine is a special form of an Easycoder program. This routine is considered generalized when it contains elements (parameters) whose values are missing and/or when only certain of the included instructions may be used in its execution. A parameter is a variable element within a macro routine which must be assigned an explicit value by a macro call in order to specialize the routine. In each macro routine, expressions called parameter designators are used to identify parameters of the routine. The routine becomes specialized when the missing parameter values are supplied and/or when the instructions which are not to be used are omitted.

A macro instruction (call) is an instruction written in a program, in-line, for the purpose of calling in a macro routine at that point. Included within the macro instruction is the list of desired parameter values for use in the macro routine being called.

Figure 1-1 illustrates the specialization function of Library Processor. The source program(s) includes macro instructions which call the macro routines from the library SPT. The source programs and macro routines are processed by Library Processor. The processing reproduces the source program(s), inserting the specialized macro routines to form a new deck which is now ready for assembly by Easycoder Assembler C or D. The concepts of "card" and "deck" are used in referring to card-image files, regardless of the medium on which the files exist. If a macro routine called by a macro instruction also includes macro instructions and their associated routines, the macro routines are called "nested." Library Processor

respecializes nested macro routines and replaces the old macro coding with the respecialized coding. All levels of nested macro routines are respecialized.

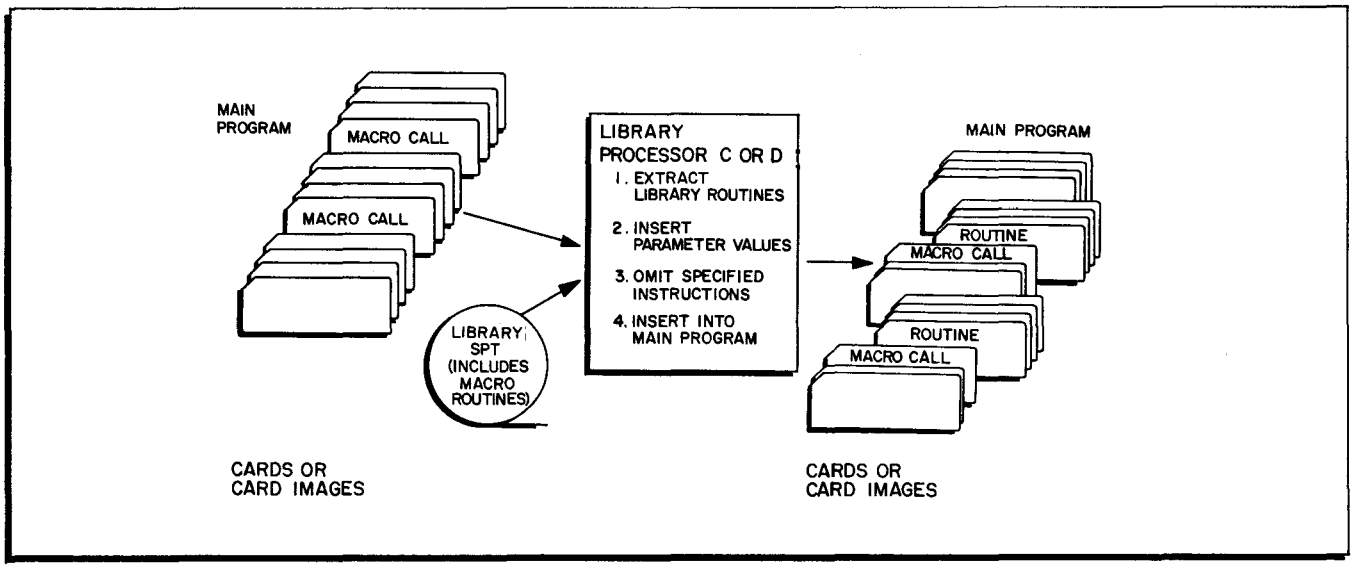


Figure 1-1. Specialization Function of Library Processor

Another function of Library Processor is the respecialization of all macro routines which are contained in a program on an input SPT for Easycoder Assembler C or D. This function is illustrated in Figure 1-2. Imbedded in the program on the input SPT are old macro instructions followed by their previously specialized routines. Library Processor is capable of respecializing the macro routines on the library SPT accordingly. Note that the input SPT is the source of previously specialized routines while the library SPT is the source of generalized macro routines. Physically, however, these two files may actually be on the same SPT. When Library Processor performs the respecialization function, the output is the respecialized macro routines, preceded by a correct director for Easycoder Assembler C or D. On a subsequent assembly run, the assembler may then replace the old macro routines in the program on the input SPT with the respecialized routines generated by Library Processor. Easycoder Assembler D automatically deletes old macro coding. In Easycoder Assembler C, the programmer must supply delete statements to eliminate old macro coding.

A third function of Library Processor is reproducing symbolic decks of entire programs on the library SPT.

Library Processor C or D can be run as a free-standing program or as an integral part of Easycoder Assembler C or D, respectively. The Equipment Configuration Descriptor (ECD) for Library Processor includes the file media fields required by the Easycoder Assemblers. When Library Processor receives an ECD which includes an equipment configuration for the

assembler as well as itself, it automatically calls Easycoder Assembler C or D without an intervening halt. When operating as an integrated system as described above, no console call card or ECD is required for the Easycoder Assembler. Figure 1-3 illustrates the integrated system of Library Processor and Easycoder Assembler.

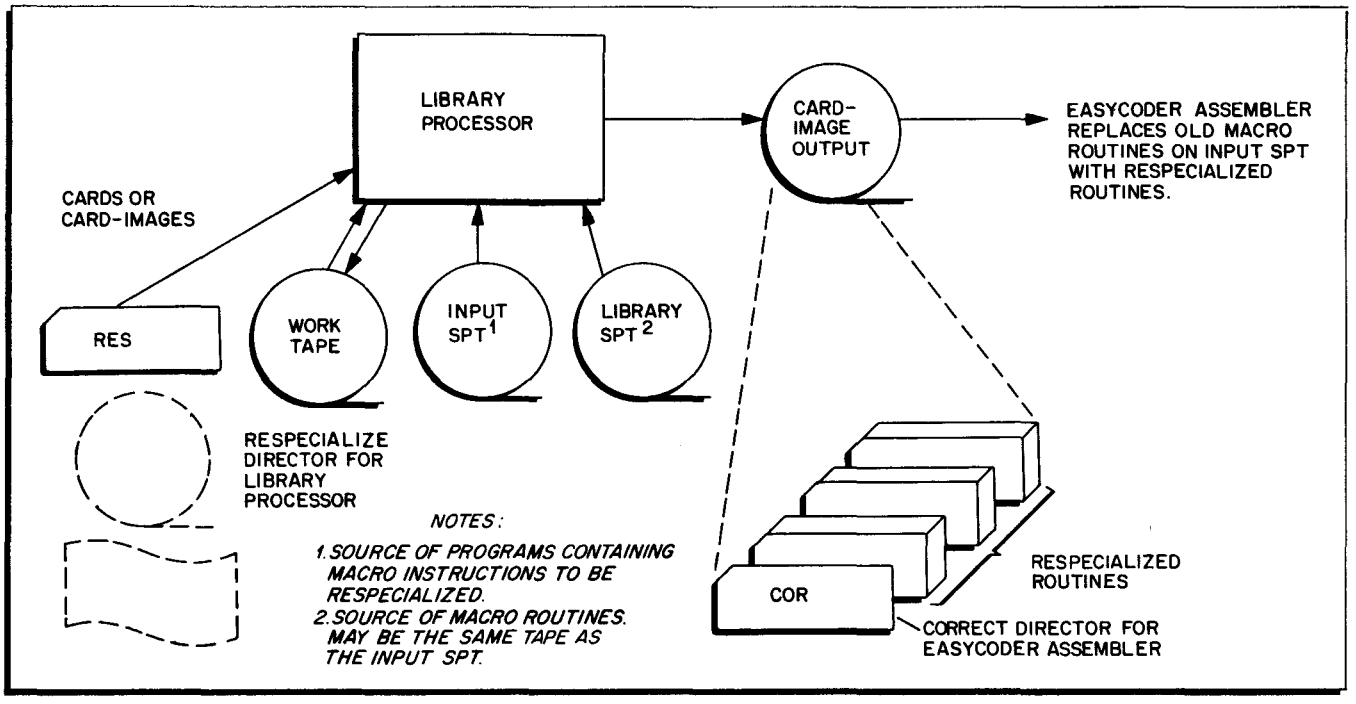


Figure 1-2. Respecialization Function of Library Processor

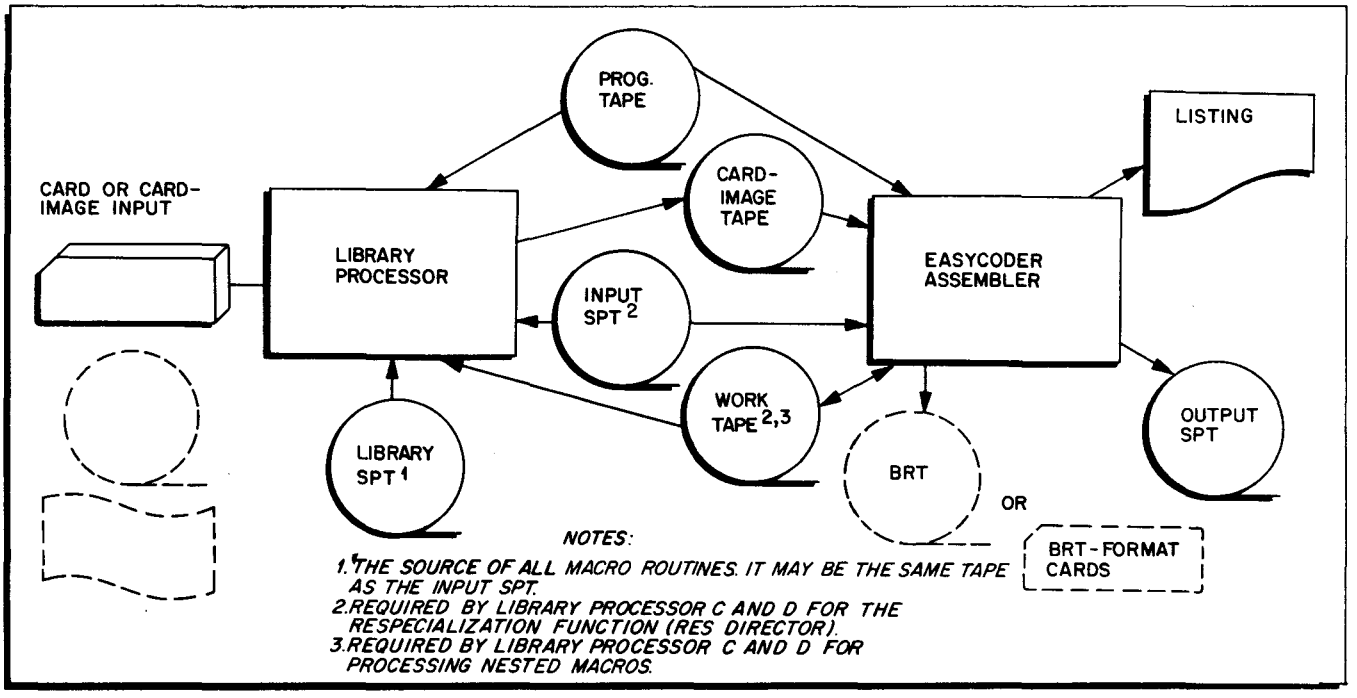


Figure 1-3. Integrated System of Library Processor and Easycoder Assembler

EQUIPMENT REQUIREMENTS

Library Processor C runs in 12,288 characters of memory; Library Processor D requires 16,384 characters. Additional memory may be used profitably to store a library SPT directory which contains entries for more than 175 programs. Both Library Processors require the Advanced Programming Instructions.

Table 1-1 lists the basic peripheral requirements for the Library Processors. In addition to these requirements, a work tape and an input SPT (if different from the library SPT) are required for the respecialization function. For automatic respecialization of nested macro routines, a work tape is required and the output device must be a card-image tape.

Table 1-1. Basic Peripheral Devices for Library Processor

File	Type of Equipment	Alternate Equipment
Program Loading Device	Type 204B Tape Unit	
Input Device[1]	Card Reader	Type 204B Tape Unit or Type 209 Paper Tape Reader
Library SPT	Type 204B Tape Unit	
Output Device[1]	Card Punch	Type 204B Tape Unit
Console Device	Control Panel	Type 220 Console Typewriter

[1] If the machine configuration contains a Type 214-2 or Type 224 Card Reader/Punch, either the input device or the output device must be a Type 204B Magnetic Tape Unit. The card reader/punch can be specified as the input device or as the output device, but not both.

SECTION II

WRITING A MACRO ROUTINE

Some routines are Honeywell-supplied (e.g., Input/Output Control programs) while others may be written by the user. This section explains how the user should write a generalized macro routine.

PARAMETER DESIGNATION

Parameter designators have the format pxy, where p is any alphanumeric character chosen by the programmer,[1] and x and y form the decimal parameter number from 00 to 63. A control instruction, Set Parameter Designator (SETP), is used to assign a value to p. SETP is written in the operation code field, and the desired value for p is written in column 21. The value of p may be changed at any time by writing another SETP instruction. If the SETP instruction is not used, the value of p is assumed to be octal 35, which prints as %. The following are possible assignments of p:

EASYCODER
CODING FORM

PROBLEM _____ PROGRAMMER _____ DATE ____ . ____ PAGE ____ OF ____

	CARD NUMBER	TYPE	MARK	LOCATION	OPERATION CODE	OPERANDS
	1 2 3 4 5	6	7	8 ... 14	15 ... 20	21 ... 62 63 ... 80
1					SETP @	
2					SETP *	
3						

Parameters are indicated by writing the currently assigned value of p, followed by a parameter number (xy) which the programmer assigns consecutively. When the routine is specialized by Library Processor, the parameter designator is replaced by the explicit value supplied in the calling macro instruction. For example, assume that parameter 03 is an index register number. An indexed address using that register with an augment of one would appear within the macro routine as 1+Xp03. When the routine is specialized, the parameter value (e.g., 5) replaces the designator p03, creating the address 1+X5. Parameter 00 (p00) is always used to indicate the tag (if any) written in the location field of the macro instruction.

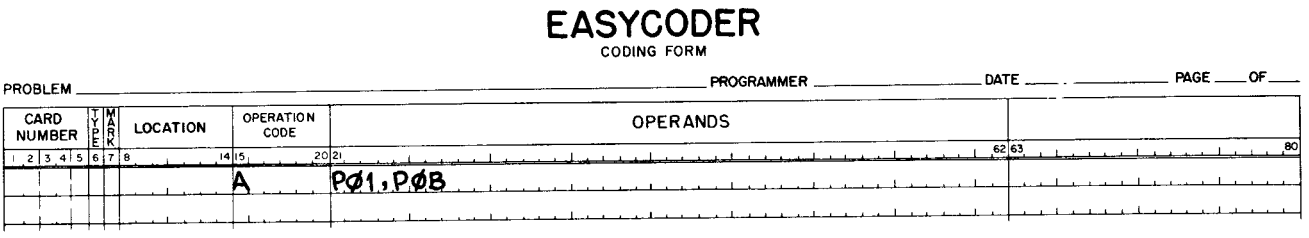
SELECTIVE OMISSION OF CODING

The programmer may desire that certain lines of coding be omitted from the macro

[1] Although there is no restriction on the characters that can be assigned to p, it is the responsibility of the programmer to insure that the resulting parameter designators do not duplicate the form of any other language element, such as a symbolic tag.

routine. The zone portion of y in the parameter designator may be overpunched with R(+) or X(-). An R(+) overpunch indicates that if the value for parameter xy is blank or omitted in the calling macro instruction, this line of coding is to be omitted from the routine. An X(-) overpunch indicates that if the parameter value is included in the macro instruction, this line of coding is omitted from the routine.

Suppose, for example, that a macro routine computes a hash total of a particular field on an optional basis. The field to be totaled is parameter 01, and the field to contain the total is parameter 02. The instruction in the routine to update the total would be coded as follows:

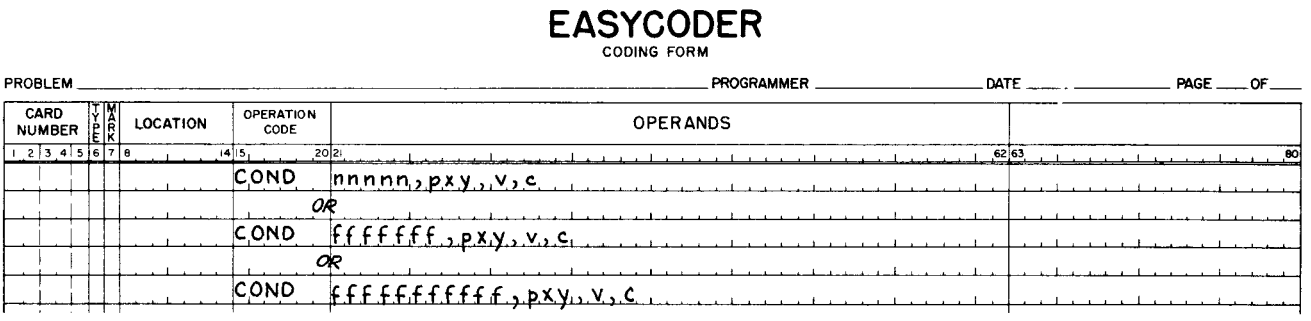


NOTE: B is the result of overpunching the 2 with an R (+).

If the parameter value for the field which is to contain the total is omitted from the call, this instruction is also omitted from the specialized routine by Library Processor. Instructions which do not explicitly address an optional parameter may also be selectively omitted by punching the parameter designator (with the appropriate overpunch) in the remarks portion of each such card.

Conditional Statements

Conditional (COND) control statements may also be used to omit lines of coding from the macro routine. The Conditional statement may have one of the following formats:



nnnnn - card number of the next statement not to be omitted when the condition is true

fffffff - (for standard card format) the location field of the next statement not to be omitted when the condition is true.

fffffffffff - (for alternate card format of Library Processor D only) the location field of the next statement not to be omitted when the condition is true

v - literal value against which parameter xy is tested

c - condition

The condition c is coded as follows:

c	Condition:
0	Never true
1	True if value of pxy > v
2	True if value of pxy = v
3	True if value of pxy ≥ v
4	True if value of pxy < v
5	True if value of pxy ≠ v
6	True if value of pxy ≤ v
7	Always true

The condition is tested by a binary Compare of the value of parameter xy (A address) against v (B address), followed by a Branch on Condition Test with a variant character of 4c (where c is interpreted as shown above). If c is true, all statements of the macro routine (including additional COND and SETP statements, if any) from this point up to, but not including, the designated card number or location field are omitted. All rules of the Compare instruction apply to the condition function.[1] A void parameter produces an equal result when compared with a field of up to 40 blanks.

EASYCODER
CODING FORM

PROBLEM _____ PROGRAMMER _____ DATE _____ PAGE ____ OF ____

CARD NUMBER	TYPE	MARK	LOCATION	OPERATION CODE	OPERANDS	
1 2 3 4 5	6	7	8 ... 14	15 ... 20	21 ... 62	63 ... 80
1				COND	ØØØ14, PØ2, KRAM, 1	
2						
3						

When the Conditional control statement illustrated in the example is processed, the value assigned to parameter 02 is compared with the literal value KRAM. If the parameter value is greater than KRAM (since c = 1), the following statements are omitted from the routine, up to statement 00014, which is included in the routine. If the parameter value is less than or equal to KRAM, no statements are omitted from the routine at this point.

Had the Conditional statement been coded as shown below, statements up to but not including the statement whose location field is JUMPΔΔΔ would be omitted when parameter 02 is greater than KRAM.

---

[1] The binary Compare reads the A- and B-address fields simultaneously from right to left starting at the end of the B address and ending with the word mark of the B address. (To avoid an equal comparison that may not be truly equal, an extra 0 character should begin the B-address field.) An example follows:

A address	B address (v values above)
nonstandard	(0) standard

(If the zero is present at the beginning of the B address, the comparison does not stop after reading "standard" in the two address fields but also compares the next character. When such possibilities do not exist, the additional character in the B address need not be used.)

EASYCODER
CODING FORM

PROBLEM _____ PROGRAMMER _____ DATE ____ PAGE ____ OF ____

CARD NUMBER	TYPE	MARK	LOCATION	OPERATION CODE	OPERANDS	
1 2 3 4 5	6	7	8 ... 14	15 ... 20	21 ... 62	63 ... 80
1				COND	JUMP ,,PØ2,KRAM,1	
2						
3						

TAG PREFIXES

Duplication of tags between the calling program and any of the macro routines called (or between two of the macro routines called) must be avoided. To avoid duplication, it is recommended that each macro routine be assigned a particular prefix and that each of its tags be preceded by this prefix. The length of the tag and its prefix must never exceed six characters in standard card format, or ten characters in alternate card format. The alternate format, described in Section IV, is acceptable to Library Processor D only. If the same macro routine is to be called more than once by a single main program, the tag prefix should be designated by means of a parameter to avoid duplication. Thus, the tag prefix will be different for each insertion of the routine.

ADDING MACRO ROUTINES TO THE SPT

A generalized macro routine is prepared in the same manner as any other program, except for the presence of parameter designators to indicate missing values. It is submitted to Easycoder Assembler C or D, complete with its own PROG and END cards, for addition to the SPT. Since there is no distinction between a macro routine and any other program on the SPT and since Library Processor searches the SPT by program name rather than by visibility, each macro routine must be assigned a program name which is unique within the entire SPT. Tape searching by Library Processor will be minimized if macro routines are stored together in one particular portion of the SPT.

SECTION III

USING THE LIBRARY PROCESSORS

WRITING A MACRO INSTRUCTION

The programmer writes a macro instruction at the point in his program where a macro routine is to be incorporated. The type field contains a C when all the parameter values for a particular macro routine do not fit on one card and require continuation cards to follow; otherwise, it contains an L. The location field may contain a symbolic tag which, when written, is always interpreted as the value of parameter p00. The operation code field contains the name of the desired macro routine (which is also the name on the PROG card for the routine); the operands field contains the parameter values, written in order by parameter number, starting with the value of parameter p01.

Continuation Cards

A continuation card is used when a macro call can not contain all the parameters for a particular macro routine. Although the first card of a multiple-card call is not a continuation card, it indicates, with a C in the type field, that a continuation card follows. Each continuation card, except the last, contains a C. The last continuation card contains an L in the type field.

Although macro instruction cards and continuation cards are reproduced by Library Processor, they are regarded as remarks cards by Easycoder Assembler C or D. COND, SETP, and END cards in a macro routine are not reproduced by Library Processor.

Omission of Parameters

A parameter value may contain any character except the comma. The comma is used to follow each parameter value, including the last. The comma also serves as a method of omitting a parameter value from the macro instruction. Each missing parameter value is indicated by its comma. However, any number of values may be omitted without their terminating commas if no further values are needed.

EXAMPLE: If a macro routine has ten parameters (p01 through p10) and the programmer wishes to omit values 3, 5, 6, and 8 - 10, he may code the instruction as follows.

EASYCODER
CODING FORM

PROBLEM _____ PROGRAMMER _____ DATE ___ . ___ PAGE ___ OF ___

CARD NUMBER	TYPE	MARK	LOCATION	OPERATION CODE	OPERANDS	
1 2 3 4 5	6	7	8 14	15 20	21 62	63 80
1	L		(TAG)	NAME	VAL1,VAL2,,VAL4,,,,VAL7,	
2						
3						

An alternative method of omitting parameter values is convenient for omitting several consecutive values when more values are to follow. Write the number of the next parameter not to be omitted in columns 15 and 16 of the next continuation card. Then write the actual value of this parameter in the operands field and continue as usual. To omit the first **n** values, do not write any values in the macro instruction card, and write the number of the first parameter whose value is not omitted in columns 15 and 16 of the first continuation card.

EXAMPLE: If a macro routine has parameters p01 through p10 and values 2 and 6-9 are to be omitted, the programmer may write the following.

EASYCODER
CODING FORM

PROBLEM _____ PROGRAMMER _____ DATE ___ . ___ PAGE ___ OF ___

	CARD NUMBER	TYPE	MARK	LOCATION	OPERATION CODE	OPERANDS
	1 2 3 4 5	6	7	8 ... 14	15, ... 20	21 ... 62
1		C		(TAG)	NAME	VAL1,,,VAL3,VAL4,VAL5,
2		L			1Ø	VAL1Ø,

To omit values 1-4, 6, 7, and 10, the programmer may write the following.

EASYCODER
CODING FORM

PROBLEM _____ PROGRAMMER _____ DATE ___ . ___ PAGE ___ OF ___

	CARD NUMBER	TYPE	MARK	LOCATION	OPERATION CODE	OPERANDS
	1 2 3 4 5	6	7	8 ... 14	15, ... 20	21 ... 62
1		C		(TAG)	NAME	
2		C			Ø5	VAL5,
3		L			Ø8	VAL8,VAL9,

The following example summarizes the complete relationship of the macro instruction, the generalized macro routine, and the macro routine after it has been specialized and incorporated into the main program. The macro routine is shown first in its generalized form as it exists on the library SPT.

EASYCODER
CODING FORM

PROBLEM _____ PROGRAMMER _____ DATE ___ . ___ PAGE ___ OF ___

	CARD NUMBER	TYPE	MARK	LOCATION	OPERATION CODE	OPERANDS
	1 2 3 4 5	6	7	8 ... 14	15, ... 20	21 ... 62
1	INS				PROG	PROB (Coded as required for Easycoder Assembler)
2					SETP	@
3				@ØØ	SCR	@Ø5ZEX+3,7Ø
4				@Ø5ZAD	A	@Ø1+X@Ø2,(@Ø4)
5					C	(@Ø4),@Ø5ZLM
6					BCT	@Ø5ZCN,45
7				@Ø5ZEX	B	ØØ
8				@Ø5ZCN	MCW	(@Ø4),@Ø3+15
9						
10						
11						
12						
13				@Ø5ZLM	DCW	@Ø6
14					END	

EASYCODER
CODING FORM

PROBLEM _____ PROGRAMMER _____ DATE _____ PAGE ____ OF ____

CARD NUMBER	TYPE	MARK	LOCATION	OPERATION CODE	OPERANDS	
			MAIN PROGRAM			
	L		TAGER	PROB	AUG,4,PRINT,SUM,MI,+1ØØ	Macro Instruction (call)
			TAGER	SCR	MIZEX+3,7Ø	
			MIZAD	A	AUG+X4,(SUM)	Specialized version of the same routine.
				C	(SUM),MIZLM	This is inserted directly after the macro
				BCT	MIZCN,45	call in the output produced by Library
			MIZEX	B	ØØ	Processor.
			MIZCN	MCW	(SUM),PRINT+15	
			MIZLM	DCW	+1ØØ	
			MAIN PROGRAM		(UNTIL THE NEXT CALL)	

Card Numbers

When a macro routine is inserted into a program, card numbers are assigned in either of two ways. The type is assigned according to the entry in the mark field (column 7) of the first macro instruction (call) card. It should be noted that when a macro call contains more than one card, each card must have a different number. The two mark field entries are specified and interpreted as follows:

Δ (blank): Generate card numbers by incrementing the card number of the macro call.

D: Generate card numbers by adding the card number of each card in the generalized macro routine to the card number of the macro call.

If the mark field is blank (Δ), card numbers are assigned to the specialized macro routine cards by incrementing by four (octal 04) the low-order character of the card number of the last macro instruction card (L type). Thus, on the assembly listing, the low-order character of the card number column may be non-numeric, as shown by the following example using a single card macro instruction with a card number of 00100.

Macro Instruction (Call)	Specialized Macro Routine	
	Program Listing	Octal Value of Low-Order Character
00100LΔ	00100*	00
	00104	04
	00108	10
	0010:	14
	0010+	20
	0010D	24
	0010H	30
	.	.
	.	.
	.	.
	0010(	74
	00110	00

If the mark field contains a D, card numbers are assigned to the specialized macro routine cards by decimally adding the card number of each card in the generalized macro routine to the card number of the last card (L type) of the macro instruction. Therefore, when D is used, the card numbers of the generalized macro routine must be decimal numerals and their values should be as low as practicable. The program listing of the resulting program will contain numeric card numbers unless there is an error in the macro instruction, in which case the resulting error lines will be numbered as if the mark field had been blank (Δ).

Since improper use of card numbers may cause the "symbolic cards out of sequence" error during assembly, the programmer should carefully allow for the card numbers assigned to the specialized macro routine when he assigns the card number to the line following the macro instruction (regardless of the method of card numbering). When unassigned card numbers are limited, it is advisable to use the first method of card numbering (blank in the mark field) to decrease the likelihood of introducing duplicate numbers in the specialized output.

Nested Macro Routines

As described in the Introduction, a nested macro routine is a macro instruction and specialized routine which is included in another macro routine. Library Processor is capable of respecializing all levels of nested macro routines and replacing the old nested routines with the respecialized routines. When the equipment configuration includes a work tape and a card-image output tape, Library Processor automatically respecializes nested macro routines. When nested macro routines are respecialized, the method of line number generation which is specified for the highest level routine is automatically applied to all lower level routines.

When nested macro routines are automatically respecialized, a lower level macro instruction can refer to parameters of the macro routine in which it is immediately contained. In its list of parameter values, the lower level macro instruction may use a parameter designator pxy

to request the value of parameter xy in the higher level routine. For example, the macro routine called by the macro instruction:

```
L       PROGA       TAG, 01, 5, LOC,
```

may contain another macro routine which is called by the instruction:

```
L       PROGB       X1, P03,
```

When the routine PROGB is respecialized, its second parameter assumes the value of the third parameter of the higher level routine PROGA, namely 5. Parameter values may be passed through any number of levels, provided that each lower level macro routine refers only to parameters of the next higher level routine.

Leaving Macro Instructions Unspecialized

In some instances, a programmer may wish to include certain macro instructions in input to Library Processor without having the associated macro routines specialized in that particular run. The specialization of such macro instructions may be bypassed by punching an X in column 80 of the first macro instruction card. The entire macro instruction, but not the associated specialized coding, appears in the Library Processor output.

RESPECIALIZING OLD MACRO ROUTINES

The Introduction describes the respecialization function of Library Processor, in which all old macro routines in a program on an Easycoder input SPT are respecialized so that the program may be subsequently updated by the Easycoder Assembler. The equipment configuration must include a work tape in addition to an input SPT. The macro routines called by the old macro instructions on the input SPT must be present on the library SPT. As previously noted, the input SPT and the library SPT may be the same physical tape.

To utilize the respecialization function, the programmer includes in the Library Processor input a Respecialize director, which is punched as follows:

Columns 1-3:	Contain RES.
Columns 15-20:	Contain PROGΔΔ .
Columns 21-26:	Contain the name of the program on the input SPT whose macro routines are to be respecialized.
Columns 27-80:	Are punched according to the requirements of a PROG card for the Easycoder Assembler (see Easycoder Assemblers C and D, Order Number 041).

Any corrections to the existing macro instructions on the input SPT may be placed immediately following the Respecialize director. The Library Processor merges any corrections with the old macro instructions in the program on the input SPT and respecializes the associated macro routines. Note that corrections to existing macro instructions are placed behind the Respecialize director only when it is necessary to change parameter values in some macro instructions at the

same time all the macro routines in the program are respecialized. The Respecialize director is not used to respecialize one macro routine with new parameter values when other macro routines in the same program should not be respecialized. Instead, Library Processor input should contain a Correct director for the assembler followed by the corrected macro instruction. Corrections to other (non-macro) instructions in a program on the input SPT may also follow a Respecialize director. In the Library Processor output, the Respecialize director is changed to a Correct director for Easycoder Assembler C or D. Following this director are any corrected instructions and the respecialized routines. On a subsequent assembly run, the old macro routine in the program on the input SPT is replaced by the respecialized routine and any corrections submitted with the original Respecialize director are made. Recall that Easycoder Assembler D automatically deletes old macro coding from the input SPT, while Easycoder Assembler C requires Delete cards when replacing an old macro routine.

USING LIBRARY PROCESSOR TO PRODUCE SYMBOLIC DECKS

In order to reproduce a complete program from the SPT, a special macro instruction is issued to Library Processor. This instruction consists of a card containing a P in the type field (column 6) and the name of the macro routine (program to be duplicated) in the op code field (columns 15-20). This single special macro instruction usually comprises a unique program in the input file, but it may alternatively be included in another program which also contains standard macro instructions and continuation cards (C and L cards). When included in another program, the P type of macro instruction must not be immediately preceded by a C card. When Library Processor reproduces a program as a result of the P type macro instruction, any Conditional statements and parameters in the program are not processed, the program line numbers are not renumbered according to the number of the macro instruction, any program END, COND, and SETP statements are reproduced, and the PROG statement is not reproduced. In addition, the P type macro instruction is itself eliminated from the Library Processor output.

SECTION IV
INPUT AND OUTPUT FILES

INPUT FILE

The input file for Library Processor comprises the optional equipment configuration descriptor and the input deck. The file is described in punched-card format. If the file appears on card-image tape or paper tape, the coding format remains the same (assuming normal use of control frames on paper tape).

Although it is described here, the console call card is not considered part of the input file. When used, it precedes the input file.

Console Call Card

The console call card, which is submitted to Tape Loader-Monitor C or Floating Tape Loader-Monitor C, initiates the loading of Library Processor C or D from the program tape (BRT). The program names of Library Processors C and D are AACLIB and AACLPR, respectively. The console call card is punched as follows:

Column(s)	Entry	Meaning
1-6	AACLIB or AACLPR	Program name of requested unit.
7-8	01	Segment name of requested unit.
9	d	Logical address of program tape (usually 0).
18	*	The asterisk in column 18 identifies the card as a Console Call card.

Equipment Configuration Descriptor

The methods of specifying the equipment configuration are explained in detail in Easycoder Assemblers C and D. These are:

Method 1 - Standard ECD number specified in the Loader-Monitor communication area.

Method 2 - Standard ECD number specified in input file.

Method 3 - Full equipment configuration descriptor specified in input file.

NOTE: Method 3 must be used when operating Library Processors C and D on a Model 120.

Under methods 2 and 3, an equipment configuration descriptor image precedes the input deck in the input file. This section defines the standard configurations for Library Processor and explains the procedure for preparing a full equipment configuration descriptor. The specific information in this section and the detailed treatment of the three methods in the

aforementioned publication provide complete instructions for specifying equipment configurations for Library Processor.

STANDARD EQUIPMENT CONFIGURATION NUMBERS

The following functional characteristics of each standard configuration are summarized in Table 4-1.

1. Input/output media.
2. Number of tapes required.
3. Processing capability: respecializing old macro routines (as a result of the Respecialize director), automatically respecializing nested macro routines.
4. Which assembly mode, if any, is automatically called by Library Processor.

Table 4-1. Functional Characteristics of Library Processor Standard Equipment Configurations

ECD No.	No. of Tapes	Auto/Ind[1]	RES[2]	NEST[3]	Input	Output	Assembly Mode
0	2	Ind	NO	NO	Card	Card	-----
1	2	Ind	NO	NO	Card	Card	-----
2	5	Auto	NO	YES	Card	Tape	Assemble Only
3	6	Auto	NO	YES	Card	Tape	Assemble & Select
4	7	Auto	YES	YES	Tape	Tape	Assemble, Update, Select
5	6	Auto	YES	YES	Card	Tape	Update & Assemble
6	4	Ind	NO	YES	Card	Tape	-----
7	7	Auto	YES	YES	Card	Tape	Assemble, Update, Select
8	5	Ind	YES	YES	Card	Tape	-----
9	6	Ind	YES	YES	Tape	Tape	-----

NOTES:

1. "Auto" means Easycoder Assembler C (or D) is automatically called after Library Processor C (or D). "Ind" means the Library Processor is free-standing.
2. "YES" means that the Respecialize director is acceptable in this configuration.
3. "YES" means that nested macro routines are automatically respecialized in this configuration.

The peripheral equipment used by each of the ten standard equipment configurations for Library Processor is listed in Table 4-2. The tape numbers are logical assignments. Tapes preceded by an asterisk (*) should be mounted in "Protect" status. The highest memory bank used by the standard configurations is bank 03_8, or 16K characters. The standard configurations employ the following peripheral addresses (octal):

Device	Peripheral Address
Tape Control Unit	00
Card Reader	41
Card Punch	01
Printer	02
Control Panel	00

Table 4-2. Peripheral Equipment for Library Processor Standard Configurations

ECD No.	Tape 0	Tape 1	Tape 2	Tape 3	Tape 4	Tape 5	Tape 6	Card Reader	Card Punch	Printer
0	*Program tape						*Library SPT	X	X	
1	*Program tape						*Library SPT	X	X	
2	*Program tape	Card-image output tape	Work[3] tape	Output SPT[1]			*Library SPT	X	X[1]	X[1]
3	*Program tape	Card-image output tape	Output BRT[1]	Output SPT[1]	Work[3] tape		*Library SPT	X	X[1]	X[1]
4	*Program tape	Card-image output tape	Output BRT[1] & Card-image input tape	Output SPT[1]	Work[2, 3] tape	*Input[2] SPT	*Library SPT		X[1]	X[1]
5	*Program tape	Card-image output tape	Work[2,3] tape	Output[1] SPT		*Input[2] SPT	*Library SPT	X	X[1]	X[1]
6	*Program tape	Card-image output tape			Work[3] tape		*Library SPT	X		
7	*Program tape	Card-image output tape	Output[1] BRT	Output[1] SPT	Work[2, 3] tape	*Input[2] SPT	*Library SPT	X	X[1]	X[1]
8	*Program tape	Card-image output tape			Work[2,3] tape	*Input[2] SPT	*Library SPT	X		

Table 4-2 (cont). Peripheral Equipment for Library Processor Standard Configurations

ECD No.	Tape 0	Tape 1	Tape 2	Tape 3	Tape 4	Tape 5	Tape 6	Card Reader	Card Punch	Printer
9	*Program tape	Card-image output tape	*Card-image input tape		Word[2, 3] tape	*Input[2] SPT	*Library SPT			

NOTES:

1. Used only by Easycoder Assembler C or D.
2. Required by Library Processor for the respecialization function.
3. Required by Library Processor for processing nested macro routines.

FULL EQUIPMENT CONFIGURATION DESCRIPTOR

If it should be necessary to specify an equipment configuration which is not included among the standard configurations for Library Processor, a full, user-constructed ECD image is required.[1]

The format of the full equipment configuration descriptor is described below. Note that columns 1 through 5 must be blank to distinguish this from an equipment configuration descriptor specifying a standard configuration.

Columns 1-5: Contain blanks.

Column 6: Contains E to designate equipment configuration descriptor.

Columns 8-10: Contain read/write channel assignments for card input, SPT input, and card-image output, respectively (normally read/write channels 1, 2, and 3 respectively). The coding for RWC assignments is as follows:

Column 8: X, 9 punch (51_8) for RWC1

Column 9: X, 8, 2 punch (52_8) for RWC2

Column 10: X, 8, 3 punch (53_8) for RWC3

NOTE: When Library Processor is run as a background program in conjunction with Interrupt Control D, one read/write channel must be reserved for the foreground program. For a processor with three RWC's, the same RWC code should appear in two columns. The third RWC must be free for the foreground program.

Columns 16-17: Contain 00 to designate the lowest memory bank (4K module) available (always 00 for the first bank used).

[1] If this full ECD is used frequently, one or more of the standard ECD's should be reassembled to reflect the needs of the installation.

Columns 19-20: Contain the number of the highest memory bank available (must be greater than or equal to 02).

Columns 21-80: Contain the file media fields, which designate the equipment configurations to be used. Columns 21 through 80 contain 20 of these file media fields made up of three columns each. These three columns, in turn, contain three characters which completely specify the device to be used. The contents of each of the fields are listed in Table 4-3.

Table 4-3. Format for Library Processor File Media Fields

Columns	Designate	First Character Device Type[1]	Second Character Peripheral Address[2]	Third Character Tape Drive[3]
21-23	Program tape	1	p	t
24-26	Console device	2 or 5	0 or p	0
27-29	Card reading device[4]	J, R, 1, or L	p	0 or t
30-32	Assembler listing device[5]	- or 1	p	0 or t
33-35	Assembler card output device[4]	Δ, O or K	Δ or p	Δ or 0
36-38	Input SPT[6]	1 or Δ	p or Δ	t or Δ
39-41	Input BRT[5]	1 or Δ	p or Δ	t or Δ
42-44	Output SPT[5]	1 or Δ	p or Δ	t or Δ
45-47	Output BRT[5]	1 or Δ	p or Δ	t or Δ
48-50	Work tape[7]	1 or Δ	p or Δ	t or Δ
51-53	Library SPT[8]	1	p	t
54-56	Library processor output device	K or S or 1	p	0 or t

NOTES:

1. The first character specifies the device type. The symbol for each device is as follows:

Character	Octal Code	Device Type
Δ	15	File Absent
0	00	Unspecified
1	01	Type 204B Magnetic Tape Unit
2	02	Control Panel
5	05	Type 220 Console
-	40	High-Speed Printer (Any Type)
J	41	Type 227 Card Reader
K	42	Type 227 Card Punch
L	43	Type 209 Paper Tape Reader
N	45	Type 223 or 224 Card Reader (also Type 214-2 when not attached to 120 integrated control)
O	46	Type 224 Card Punch (also Type 214-1 or 214-2 when not attached to 120 integrated control)
R	51	Type 123 Card Reader attached to 120 integrated control (or Type 214-2 Card Reader/Punch

Table 4-3 (cont). Format for Library Processor File Media Fields

Character	Octal Code	Device Type
R (cont)	51	attached to 120 integrated control and used only as a card reader)
S	62	Type 214-1 Card Punch attached to 120 integrated control (or Type 214-2 Card Reader/ Punch attached to 120 integrated control and used only as a card punch)

2. The second character specifies the peripheral address assignment and the input/output sector to which the peripheral control is connected. This is control character C_2 of the PDT instruction. The six bits of the character ($V_6V_5V_4V_3V_2V_1$) are used as follows:

   a. Bit V_6 is the I/O bit. For tape files, this bit is set by the Library Processor. For card and print files, this bit must be set correctly in the character as it is punched in the card.

   b. Bits V_5V_4 specify the I/O sector. These bits are 00 to indicate the first I/O sector and 10 to indicate the second I/O sector. I/O sector bits are meaningful only if the machine configuration includes Feature 1115 (Second Input/Output Sector for the Type 2201 Central Processor) and the Library Processor D program is being used.

   c. Bits $V_3V_2V_1$ specify the peripheral address assignment. If this character specifies an address in the second I/O sector, a second I/O sector read/write channel will be activated. This second I/O sector RWC is the counterpart of the first I/O sector RWC that appeared in the applicable read/write channel column of the ECD. (RWC's 1, 2, and 3 have as their second I/O sector counterparts RWC's 4, 5, and 6 respectively.) As an example, given a configuration containing Feature 1115, assume that in using Library Processor D a programmer decides to assign the input SPT to the second I/O sector. Bits V_5V_4 of the second character of the applicable file media field must contain 10 and bits $V_3V_2V_1$ must be a valid second I/O sector address. The RWC value that Library Processor D assigns to read the input SPT would then be modified to reflect a second I/O sector RWC, e.g., if RWC2 were specified, then RWC5 will be actuated.

3. The third character specifies the number of the tape drive to be used. This is the low-order octal digit of control character C_3 of the PDT machine instruction. For devices other than tape drives, this character is usually 0.

4. Either the card reading device field or the Library Processor output device field must specify a Type 204B Magnetic Tape Unit if the card reading and card punching equipment is a combination reader/punch (Type 214-2 or 224).

5. Required only for the assembler. These fields may be blank if Library Processor is run as a freestanding program. When Library Processor is run as a freestanding program, the output SPT field must be blank.

6. Required by Library Processor to process Respecialize directors. Note that the input SPT is not the source of generalized macro routines.

7. Required by Library Processor to process nested macro routines and/or Respecialize directors. Also required by the assembler.

8. This tape is the source of generalized macro routines. It may be the same tape as the input SPT.

Input Deck

While the equipment configuration descriptor may or may not appear as a physical part of the input file (it is not used when a standard ECD number resides in the Loader-Monitor communication area), the input deck will always be present and must contain a system specific header card, the assembly input, and an end-of-file card, in that order. These cards are shown in Figure 4-1 and described below.

SYSTEM SPECIFIC HEADER CARD

Columns 1-5: Contain 1HDR Δ

Columns 21-30: Contain EASYCODER Δ

ASSEMBLY INPUT

These cards include assembly directives and symbolic program cards (including macro instructions).

The alternate card format, which allows room for ten-character tags in the location field, is available in Easycoder Assembler D. Consequently, this alternate format is acceptable to Library Processor D. In the input to Library Processor D, the character A punched in column 75 of a PROG card specifies that all subsequent cards, up to but not including the next PROG card, are punched according to the alternate format. The alternate format is defined as follows:

Column	Contents
1-5	Line Number
6	Type
7	Mark
8-18	Location
19-24	Operation Code
25-80	Operands

PROG cards are always interpreted according to the standard format.

The PROG card does not specify the source format of macro routines called within the program. This is specified by a character in the program header of each macro routine, as it appears on the Library SPT. However, the PROG card does control the format of the specialized macro routine as it appears in the output of Library Processor D. It is, therefore, possible to include in a program using the alternate card format one or more macro instructions calling for macro routines written in the standard format. In this case, Library Processor D lengthens the location field by adding blanks and shifts the op code and operand fields four positions to the right in the card image. Any information that appeared in columns 77 through 80 of the original card image is lost. If a macro routine written in alternate format is called by a program written

in standard format, Library Processor D shortens by four characters the location field of each card image in the macro routine. Any information in columns 15 through 18 of the original card image is lost.

If the PROG card is a Respecialize director, the contents of column 75 also control the interpretation of the program on the input SPT. For example, if a director of the form

RES PROG ABCDEF

contains an A in column 75, Library Processor D assumes that the card images extracted from the SPT for the program ABCDEF are constructed according to the alternate format.

END-OF-FILE CARD

Columns 1-5: Contain 1EOFΔ

OUTPUT FILE

The output file consists of all the cards which made up the input deck. In order it contains:

1. The systems specific header card.
2. The input to the assembler with each macro instruction followed by the corresponding specialized macro routine.
3. The end-of-file card.

Two end-of-recorded-information cards (1ERIΔ in columns 1-5) are generated following the end-of-file card.

NOTE: When the output file is on tape an additional 80-character record precedes the system specific header record and has the following format:

Characters 1-5: 1HDRΔ

Characters 21-30: CARDIMAGES

Characters 41-44: 1A01

Characters 78-80: 11A

The remaining characters are preserved from the previous contents of the tape record.

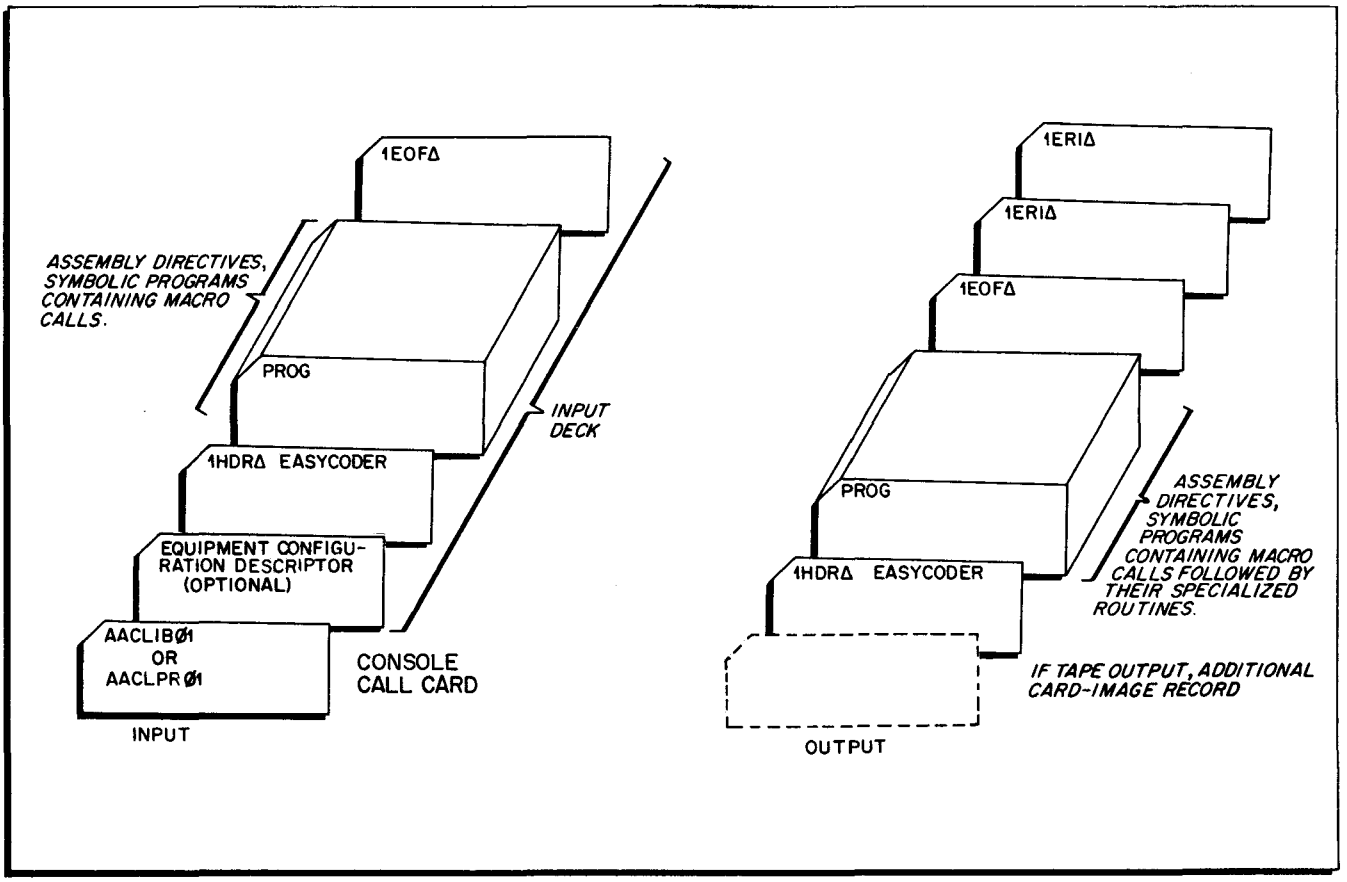


Figure 4-1. Library Processor Input and Output Deck Arrangements

SECTION V

OPERATING PROCEDURES

INITIAL SETUP PROCEDURES

1. Depress the STOP button on the control panel if the machine is not already stopped.
2. Place the cards to be read into the card reader. Initialize the reader and perform the necessary steps to begin reading cards.

   NOTE: If the card reader is not to be used (i.e., when the input deck is on a card-image tape or on paper tape and the contents of the console call card are to be entered from the control panel), omit this step and proceed to step 3.
3. If the input deck is on paper tape, mount the paper tape reel or insert the paper tape strip in the paper tape reader. Initialize the paper tape reader.

   NOTE: If the input deck is not on paper tape, omit this step and proceed to step 4.
4. Mount the program tape on the tape drive designated as logical 0. Rewind the program tape and set the PERMIT-PROTECT switch to PROTECT.
5. Prepare the remaining tapes in accordance with the designated usage as outlined in "Standard Equipment Configuration Numbers" in Section IV. If a full ECD is used, refer to the file media fields in columns 27 through 56 of the ECD card.

BOOTSTRAP PROCEDURES

The bulletins Tape Loader-Monitor C (Order No. 221) and Floating Tape Loader-Monitor C and Interrupt Control D (Order No. 005) outline the procedures for bringing into memory the Tape Loader-Monitor C and Floating Tape Loader-Monitor C, respectively. The aforementioned publications also contain console call procedures which should be used for loading and executing Library Processor.

ERROR CONDITIONS

For all halts, the B-address register should be displayed first to indicate the reason for the halt. In some cases, the A-address register contains supplementary information.

If Library Processor is using the control panel, the following actions occur at an error condition:

1. Library Processor halts.
2. Displaying the B address gives the reason for the halt (see Table 5-1).
3. If the run is to be continued, the RUN button is depressed.

If Library Processor is using the console typewriter, the following actions occur at an error condition:

1. The console warning bell rings.
2. The error message is typed (see Table 5-1).
3. The TYPE light is illuminated.
4. Library Processor stalls.
5. If the run is to be continued, the G key is depressed.

Table 5-1. Library Processor Error Conditions

Contents of B-Address Register	Console Typewriter Message	Cause and Action
07025	: NO E CARD␣	The card image that has been specified as the ECD does not contain an E in character 6. 1. If the ECD is to be entered through the input device specified by the ECD parameter in the loader communication area, ascertain that the desired ECD is in the input device and continue the run. 2. If the ECD is not to be entered through the input device specified by the ECD parameter in the loader communication area, the ECD field, locations 227_8 - 232_8, may be changed to a. Accept the ECD from a different device or b. Select one of the ten standard equipment configurations. When this has been done, continue the run by depressing RUN button or G key.
04007	: SPT DIRECTORY EXCEEDS MEMORY␣	This should occur only on a 12K memory configuration. Library Processor stores in memory the name of each program on the SPT over which it passes. On a 12K machine, it can store over 300 program names. Continue the run. Any library routines located on the SPT from this point on will not be processed by Library Processor.
04010	: NO SYSTEM HEADER␣	Either the director file or the SPT was not found. Mount the correct input and start the run over.

Table 5-1 (cont). Library Processor Error Conditions

Contents of B-Address Register	Console Typewriter Message	Cause and Action
0culd	: RD ER cu d␣	If "cu" is the address of a tape control, an uncorrectable read error has occurred on tape "d" of that control. Continue the run to retry the correction procedure. If this action is not effective, corrective measures such as cleaning the tape and re-starting the run, recreating the tape which caused the error, etc., should be performed.
0cull	: RD ER cu 1␣	If "cu" is the address of the card reader, a hole-count error has occurred. 1. Remove the cards from the input hopper of the card reader. 2. Run out the cards in the reader. 3. Place the cards from the runout hopper back into the card reader followed by the remaining input cards. Continue the run.
0cul2	: RD ER cu 2␣	If "cu" is the address of the card reader, a hole-count error has occurred. The same action is taken in this case as is outlined for a hole-count error above. NOTE: The first card in the runout hopper must be corrected to remove the illegal punch.
0cu2d	: WR ER cu d␣	If "cu" is the address of a tape control. an uncorrectable write error has occurred on tape d of that control. Continue the run to retry the correction procedure. If this action is not effective, corrective measures such as cleaning the tape and re-starting, repeating the run which created the tape, etc., should be performed.
0cu2l	: WR ER cu 1␣	If "cu" is the address of the card punch, a hole-count error has been detected. Continue the run to repunch the error cards.

Table 5-1 (cont). Library Processor Error Conditions

Contents of B-Address Register	Console Typewriter Message	Cause and Action
0cu10	: RD ER cu 0 ␣	If "cu" is the address of the paper tape reader control, a parity error has been detected on paper tape. Continue the run to ignore the error. (The operator may make note of which program is currently being processed.)

APPENDIX A

PAPER TAPE INPUT TO LIBRARY PROCESSOR

In general, paper tape processing closely parallels punched card processing. Paper tape data may be read directly into memory, interpreted by a programmed routine, and processed. Alternatively, paper tape data in any code may be translated into Series 200 code and edited prior to being written on magnetic tape. The data is then sorted and processed in a manner similar to card-generated input.

The Library Processors convert the frames punched in paper tape into Series 200 six-bit internal code by means of a translation routine and table. This table is easily changed to reflect the character sets of different paper tape perforators. The standard versions of Library Processors C and D accept paper tape input punched with six data channels. The data channels must be the six lowest-order channels on the paper tape; either 7/8-inch or 1-inch wide paper tape may be used. The six-bit values read from paper tape are translated to internal codes as shown in Table A-1, unless otherwise specified.

Table A-1. Translation of Paper Tape Code to Internal Code

Paper Tape Code (octal)	Internal Code (octal)	Paper Tape Code (octal)	Internal Code (octal)	Paper Tape Code (octal)	Internal Code (octal)
00	72	26	65	54	73
01	21	27	66	55	40
02	22	30	67	56	33
03	23	31	70	57	61
04	24	32	71	60	00
05	25	33	15	61	01
06	26	34	36 Delete line*	62	02
07	27	35	75 End field*	63	03
10	30	36	76 End reel/strip*	64	04
11	31	37	37 End line*	65	05
12	41	40	15	66	06
13	42	41	15	67	07
14	43	42	55	70	10
15	44	43	52	71	11
16	45	44	53	72	14
17	46	45	35	73	32
20	47	46	17	74	15
21	50	47	12	75	13
22	51	50	74	76	15
23	62	51	34	77	77 Ignore char.*
24	63	52	54		
25	64	53	20		

NOTE: All internal codes are word marked. In addition to the word mark, the control characters (those which have an asterisk) have an item mark.

PAPER TAPE INPUT OPTIONS

Library Processors C and D may be reassembled to accept the following options in paper tape input.

Parity Check

The constant tagged CINTY has a standard value of #1C77 (77 means no parity checking). If even parity checking is desired, this constant should be changed to #1C00; for odd parity, the constant should be changed to #1C04.

Six-Level Tape

To read six data channels (not including parity) using a single non-standard translation table, the 64-character table beginning at the location tagged CINTT must be changed. The revised translation table must not include a control frame indicator for "switch tables" (35_8 with an item mark), and the data channels must always be punched as the six low-order channels on the paper tape. Parity may be punched in either channel 7 or channel 8. (The tape may be 7/8-inch or 1-inch wide.)

Six-Level Tape with Two Translation Tables

It may at times be desired to read six data channels (not including parity) using two translation tables (so that a paper tape frame may have two different meanings depending on which translation table is being used). To accomplish this operation, the first (lower-case) translation table should be set up as a 64-character table beginning at the location tagged CINTT, and the second (upper case) translation table should be set up as a 64-character table beginning at the location tagged CINTR.

The data channels must always be punched as the six low-order channels on the paper tape, and parity may be punched in either the seventh or the eighth channel. The tape may be 7/8-inch or 1-inch wide.

Seven-Level Tape

To read seven data channels (not including parity) the constant tagged CINMK should be changed (from its standard value of #1C00) to #1C01. In addition, the 64-character translation table must be expanded to 128 characters. This latter operation should be performed by modifying the table beginning at location CINTT to contain the internal codes corresponding to paper tape frames between 000_8 and 077_8, and by inserting the internal codes corresponding to paper tape frames 100_8 and 177_8 starting at the location tagged CINTR. The resulting 128-character translation table must not include a control frame indicator for "switch tables" (35_8 with an item mark).

The data channels must always be punched as the seven low-order channels on the paper tape, and parity may be punched in column 8. The tape may be 7/8-inch or 1-inch wide.

Five-Level Tape

It may occasionally be desired to read five data channels (not including parity) using two translation tables (so that a paper tape frame may have two different meanings depending on which translation table is being used). To accomplish this operation, the first (lower-case) translation table should be set up as a 32-character table beginning at the location tagged CINTT, while the second (upper case) translation table should be set up as a 32-character table beginning at the location tagged CINTR. Following this, the 32-character table starting at CINTT should be repeated starting at location CINTT + 32, and the 32-character table starting at CINTR should be repeated starting at location CINTR + 32.

The data channels must always be punched as the five low-order channels on the paper tape, and parity may be punched in the sixth, seventh, or eighth channel. The tape may be 11/16-inch, 7/8-inch or 1-inch wide.

PREPARATION OF PAPER TAPE INPUT

Preparation of paper tape input must adhere to the standards specified in the Honeywell Information Bulletin, Easycoder Paper Tape Assembly and Loader Programs (DSI-395), as well as to the preparation considerations discussed below.

Positioning Paper Tape Input

A delete line character or a series of ignore characters should precede the input data at the beginning of a paper tape strip or reel. This convention allows the operator to position the paper tape in order to read the initial input data on tape. If only sprocket holes are present at the beginning (or end) of a paper tape strip or reel, then parity errors may occur when parity is being checked. The presence of only sprocket holes in the above positions will be detected as parity errors during even-parity checking for 11/16-inch or 7/8-inch tape and during odd-parity checking for 1-inch tape.

Field Definitions

Standard paper tape field definitions are equivalent to card fields beginning in card columns 1, 15, and 21.

Data immediately following a field termination character will be considered to be positioned at the beginning of the next field unless the field termination character is in the first column of the next field, in which case the field termination character will be ignored. Thus, if positioning to the next field is desired, then at least one character must precede the field termination character.

Beginning of Each Card Image

Each card image on paper tape is always considered to begin using the first (lower-case) translation table.

End of Paper Tape Strip or Reel

An end-of-strip control frame should be punched at the end of every reel or strip of paper tape in order to afford automatic runout of the paper tape. An end-of-strip frame punched on a paper tape strip causes the strip to be run directly out from the reading head. An end-of-strip frame on a reel of paper tape causes the paper tape to be unwound from the input reel and wound, in reverse order, onto the take-up reel. The operator should then mount the next strip or reel of input to the program, if additional input is required.

Honeywell

ELECTRONIC DATA PROCESSING

HONEYWELL EDP

TECHNICAL PUBLICATIONS REMARKS FORM

TITLE: MOD 1 (TR) LIBRARY PROCESSORS C AND D SOFTWARE MANUAL

DATED: MAY, 1967

FILE NO: 123.1605.001C.2-051

ERRORS NOTED IN PUBLICATION:

Fold

Cut Along Line

SUGGESTIONS FOR IMPROVEMENT TO PUBLICATION:

Fold

FROM: NAME _____ DATE _____

COMPANY _____

TITLE _____

ADDRESS _____

_____

FIRST CLASS
PERMIT NO. 39531
WELLESLEY HILLS
MASS.

BUSINESS REPLY MAIL

No postage stamp necessary if mailed in the United States

POSTAGE WILL BE PAID BY

HONEYWELL
ELECTRONIC DATA PROCESSING DIVISION
60 WALNUT STREET
WELLESLEY HILLS, MASS. 02181

ATT'N: TECHNICAL COMMUNICATIONS DEPARTMENT

Cut along this line

Honeywell
ELECTRONIC DATA PROCESSING