S I N G L E A D D R E S S A S S E M B L Y P R O G R A M

Date written
July, 1953

Written by
Ed Law

With suggestions from
Jack Strong
Charles Davis

Report written
March, 1954

Written by
Ed Law

With suggestions from
Jack Strong
Frank Wagner
Florence Anderson
Roger Mills

29-1013-0(NA)

CONTENTS

	Page
GENERAL DESCRIPTION	1
Objectives	1
Philosophy	1
Capacity	4
Speed	4
Components Used	5
Checks	5
Input	6
Output	6
Assembled Program Location	6
Electrostatic Storage Use during Assembly	7
Block Diagram	8
SYMBOLIC INPUT	10
Location	10
Operation	10
Alphabetic Operation Codes	11
Alphabetic Type Codes	11
Type	12
Address	12
Remarks	12
Examples	13
MACHINE ASSEMBLY	15
Symbolic Decimal Card Form	15
Card Order and Reader Board	15
Sense Switches	15
Programmed Stops and Sense Lights	16
Printer Board and Alteration Switches	17
Printer Board Wiring	18
Description of the Listing	19
Punch Panel	19
Binary Card Form and Loading	19
DETAIL LISTING PART I	21
Coding	21
Data	25
Temporary	25
DETAIL LISTING PART II	26
Coding	26
Data	32
Temporary	32

SINGLE ADDRESS ASSEMBLY PROGRAM

GENERAL DESCRIPTION

OBJECTIVES:

1 To provide a means of converting a program written in a symbolic decimal and alphabetic form to an absolute binary form.
2 To keep the symbolic form used as simple and direct as possible.
3 To avoid having to break any but the very largest program into blocks for assembly.
4 To produce a listing of the assembled program showing the symbolic and absolute forms as well as a description of the steps taken.
5 To punch binary cards which are ready for loading and execution. Each binary card to contain word count, initial address, check sum, and a signal as to whether or not it is the last card of a group.
6 To approach as nearly as possible the speed of continuous card reading and printing regardless of the size of the program being assembled.
7 To provide certain checks on the program being assembled.

PHILOSOPHY:

The reason for wanting to write a program in a symbolic form rather than in an absolute form is found in the programmer's desire for "elbow room" in developing the program. The analysis of even the best organized problem may be found in fault and changed after the program has been partially completed. The need for additional instructions in one part of a program often becomes apparent as another part is written. This sort of change is disastrous when the program has been written in absolute form.

More often than not, it is convenient to write the core of a program first, and then build out from that toward the beginning and end of the program. It is impossible in this case to know what absolute location each instruction will eventually occupy when the program is complete. But we do know that when the program is complete, successive instructions will occupy successive absolute locations. Hence we may adopt an arbitrary sequence of digits to stand for the location of the instruction. Call this the symbolic location. Then the only regularity demanded of the set of symbolic locations is that they be in order. No other restrictions need be placed on the symbolic locations used.

It is convenient, though not necessary, to adopt some convention as to the use of the sequence of digits making up the symbolic location. The coding usually falls into several logical sections. It is common practice to call them regions. The order of the instructions within a region is called the sequence number. Finally, at least one digit is needed for insertions between adjacent sequence numbers. Two decimal digits for region number gives the possibility of one hundred regions within a program. Up to one hundred instructions can be written in each region by using two more decimal digits for sequence number within a region. Finally, one decimal digit used for insertions gives the possibility of nine insertions between each instruction. Thus a total of five decimal digits used for symbolic location would seem ample for easy flexible programming.

Now since five decimal digits will convert to, at most, seventeen binary bits, a symbolic location will occupy one half word. This is important because the maximum size of the program which can be assembled depends directly on the space required for each symbolic location.

In general, the data required by a program fall into two classifications. There are the data required by the program every time it is executed. Call this permanent data. And there are the data which change each time the program is executed. Call this variable data. There will be data which fall in between the two classifications somewhere, but the following will still hold. In the interest of minimum loading time, the permanent data should probably be converted and punched on binary cards. This might also be done to variable data if it is going to be loaded more than once, but even if it is loaded from decimal cards, there should be as many numbers per card as possible. Either of these alternatives strongly suggests a compact data arrangement. This consists of a single area for a whole program with pieces of data assigned to consecutive locations within the area. The area itself can be assigned by the assembly program to the absolute locations immediately following the last instruction.

The need for freedom to make insertions in the instruction area all during the preparation of a program stems from the sequential way in which the machine executes the instructions. This is not the case with data, so no difficulty arises from assigning the data compactly within the data area.

A certain amount of erasable or temporary storage is also needed by the program during its execution. In general, a temporary location is used only briefly by the program and can be reused several times during the course of the program. As in the case of data, there is no programming objection to assigning the temporary locations compactly in a single temporary area. This facilitates an efficient use of a minimum number of temporary locations. The temporary area can be assigned by the assembly program to the absolute locations immediately following the data area.

The program has been broken, then, into three distinct areas in electrostatic storage--the instruction area, the data area, and the temporary area. Addresses referring to the instruction area will be

called symbolic type addresses. Those referring to the data area will be called data type, and those referring to the temporary area will be called temporary type. There is a fourth type of address which overlaps the other three. This is the absolute type by which reference is made directly to an absolute location anywhere in electrostatic storage or to an input-output unit, switch, light, etc.

Each of the four types of address requires different handling during the assembly process in order to correctly assign the corresponding absolute address. A symbolic address necessitates a search for the corresponding symbolic location. Since any instruction in a program may refer by way of its address to any other instruction in the program, it is necessary to have in electrostatic storage a file of all the symbolic locations used in the program being assembled. If each symbolic location requires only one half word in the file, the limit on the number of instructions which can be assembled at one time is equal to 4096 minus the number of half words required by the assembly program itself.

This file of symbolic locations can be built up by retaining in electrostatic storage each symbolic location as the instructions are read to one card. The remaining information on each card can be written on tape so that the cards will not have to be read again. When all of the instructions have been read and the file of symbolic locations is complete, the instructions can be read back from tape one at a time, and the symbolic type addresses used as the arguments in a search of the file of symbolic locations. The absolute file location in which the correct symbolic location is found is linearly related to the absolute program location. In fact, since the symbolic locations each occupy a half word in the file, the difference between the file origin and the assigned program origin is of course the difference between any file location and the corresponding program location. This difference can be stored and applied against each absolute file location which results from the search of the symbolic locations to yield the program location as assembly proceeds.

At the time the instructions have all been read into the machine and the file of symbolic locations completed, the number of instructions in the program being assembled can be calculated. Now, if the programmer has specified the absolute location at which his program should begin and the number of half words of data used in the program, the data area origin and the temporary area origin can be calculated. The data origin becomes the first even address following the last instruction, and the temporary origin becomes the first even address following the last data location. Then the absolute equivalent of a data type address or a temporary type address can be calculated by adding the relative address to the respective origin. If the number of half words of temporary storage is also specified, a check can be made to be certain the program is not located too high in memory.

The three values which the programmer has been asked to specify, the program origin, the number of half words of data, and the number of half words of temporary, are written in absolute as the addresses of the last three instructions.

When the tape is read back record by record and each instruction is assembled into absolute form, immediate printing saves both time and storage space. Tape reading and symbolic searching can be completed quite easily between continuous print cycles. Since the program listing should be in direct oder, the instructions have to come off the tape one by one in the same direct order. But it would be advantageous to avoid having to wait for the tape to rewind between writing and reading. This can be done if the instructions are written on the tape in reverse order and then the tape is read backwards. The cards can be sorted into reverse order as easily as into direct order. And the file of symbolic locations used can be set up starting at half word 4095 and working back. The last file location entered then becomes the file origin.

An alphabetic operation code is convenient to avoid having to memorize the absolute operation codes. It can be short and still be sufficiently mnemonic. It is desirable to have some kind of a check that the operation code written is allowable and that it has been key-punched correctly. The latter check demands some regularity in the codes; for example, that they all be two letter alphabetic codes. This is far from sufficient as a check, but it will catch a great many errors.

With regard to the first check, if the alphabetic codes are chosen so that the numeric underpunching is unique, the underpunching can be used as the argument in a table lookup operation to obtain the corresponding absolute code. Along with the absolute code, a test amount can be picked up to indicate that the underpunching is an allowable combination. This type of operation lookup also has the advantage of a readily accessible table if changes are desired. Additional operations for an interpretive scheme may be added for assembly.

The sign of an instruction is important when that instruction is to be modified, but otherwise it has a meaning which does not follow directly that of a full- or half-word reference. Many of the instructions are never modified, and when one is, the programmer's attention is directed to that instruction specifically. Consequently, the use of a 1 or a 2 instead of a sign in the symbolic instruction to mean one half word or two half words (a full word) is perhaps slightly more direct. This might not matter to the experienced programmer, but it may make a difference to the part-time man.

CAPACITY: Will assemble a program containing up to 3700 instructions.

SPEED: Requires approximately 850 milliseconds per instruction assembled. Most of this time is used in card reading and printing. Tape writing takes place between cards without slowing card reading. Then tape reading, assembly, and drum writing* take place between printing the assembled instructions one per line at full printing speed. Finally

*When the program being assembled contains more than 1850 instructions

drum reading* and binary punching take place. This speed can almost be doubled if a listing of the assembled program is not required. Putting sense switch #5 down eliminates printing.

COMPONENTS USED

Makes use of the card reader, all of electrostatic storage, tape #1 or tape #2 (under the control of sense switch #2), drum #4*, the printer if a listing of the assembled program is desired, and the punch.

CHECKS

The symbolic instructions are sequence checked as they are read. Any instruction out of sequence will result in a programmed stop.

The punch pattern on the cards is checked as the cards are read. Any punching which is contrary to the numeric-alphabetic pattern will result in a programmed stop.

The operation code is checked as the cards are read. Underpunching which does not match one of the standard codes will result in a programmed stop.

In these three cases, the offending card is the third one from the end if the cards are fed out. In the last two cases, running a corrected card followed by the remainder of the cards into the reader and pressing the start button will continue the process where it was interrupted. This is not possible in the case of the sequence error because the 701 tapes cannot be stepped back a unit record while they are in write status.

The number of instructions is checked during card reading, and a programmed stop will occur if an attempt is made to read in more than 3700 instructions.

A check sum is carried with each unit record on tape and will result in a programmed stop in the event of a tape error.

If printing is to take place, a check is made to be certain that the correct printer board is in. If ~~it is not~~, a programmed stop will occur. If the start button is pressed after the correct board has been put in, the assembly process will continue.

the correct board is not in the machine

A check is made to be certain that every symbolic address has a corresponding symbolic location. If this is not the case, a programmed stop will occur. If the start is pressed, the assembly process will continue with an indication being printed beside the instruction with the bad address. This address check is applied only to those instructions having symbolic addresses.

*When the program being assembled contains more than 1850 instructions

The total amount and location of electrostatic storage used for instructions, data, and temporary storage is checked to be certain that the high order end of electrostatic storage is not exceeded. If it is, a programmed stop will occur.

In all of the above stops, one of the sense lights (see section on sense lights) is turned on to help identify the cause of the stop.

INPUT

Input to the card reader consists of symbolic instructions punched one per card. See the section on machine assembly for the exact card columns. The principle reason for reading only one instruction per card is to maintain the ease of inserting or deleting single instructions.

The input must include three dummy instructions for the purpose of specifying three pieces of information about the program to be assembled. The three highest instructions must have as their address parts respectively the program origin, the number of half words of data, and finally the number of half words of temporary storage as shown in the following example:

```
99997   ST A1    2000     Program origin
99998   YE A1      48     Half words data
99999   YE A1      20     Half words temporary
```

OUTPUT

Output consists of both printing and punching. The program is listed one instruction per line in both symbolic decimal and absolute octal. Remarks are printed to help in following the course of a program. The absolute is printed in octal rather than decimal because octal is the form most useful in using the console, memory printouts, etc.

The assembled program is binary punched forty-four instructions per card. The nine-left row of each card is used for control information for that card. Columns 10-14 contain the full word count on that card. Columns 15-26 contain the address for the first instruction on that card to read into. Columns 33-44, when other than zero, signal that the card is the last of a group and give the address to which control is to be transferred when the card has finished reading. The 12-right row, or the last half row punched if the card is not full, contains the check sum, a simple sum of all the other half rows on the card, including the 9-left control word.

ASSEMBLED PROGRAM LOCATION:

There is no restriction on the area of electrostatic storage for which a program may be assembled. However, if a program origin is specified so high that instructions, data, and temporary will exceed the remainder of storage, an error stop will occur. (See the section on checks).

The instruction area begins at the program origin which has been specified by the programmer. The data origin is assigned by the assembly program to the first even location following the last instruction (the third of the three final dummy instructions). The origin of temporary storage is assigned by the assembly program to the first even location following the last data location. This has been determined by the programmer's specification of the number of half words of data.

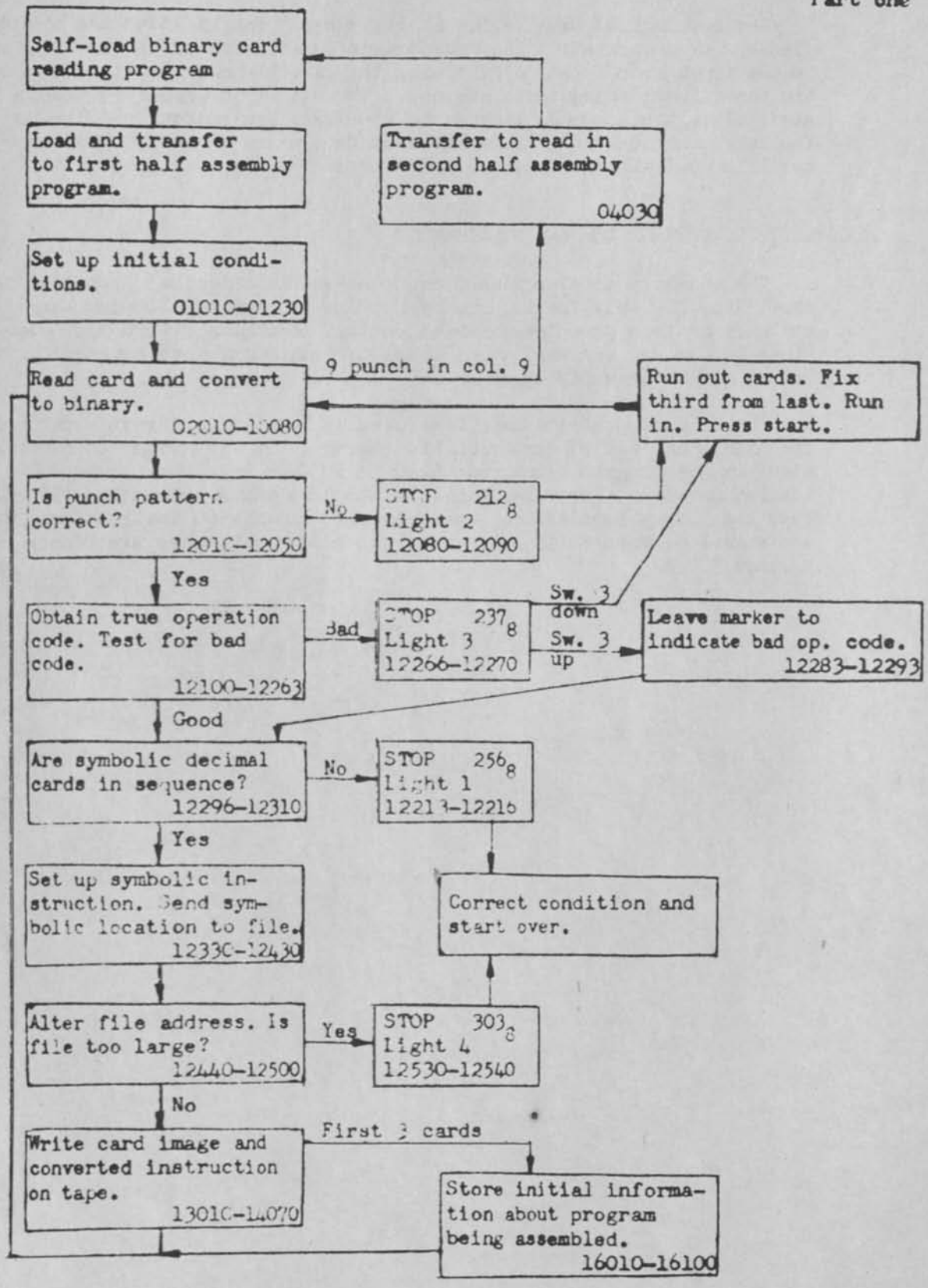
ELECTROSTATIC STORAGE USE DURING ASSEMBLY

The assembly program itself occupies absolute decimal locations 40 thru 395. The self-loading binary reading program for loading the assembly program occupies absolute decimal locations 0 thru 35. Locations 36 thru 39 are used as an absolute transition area between the two halves of the assembly program.

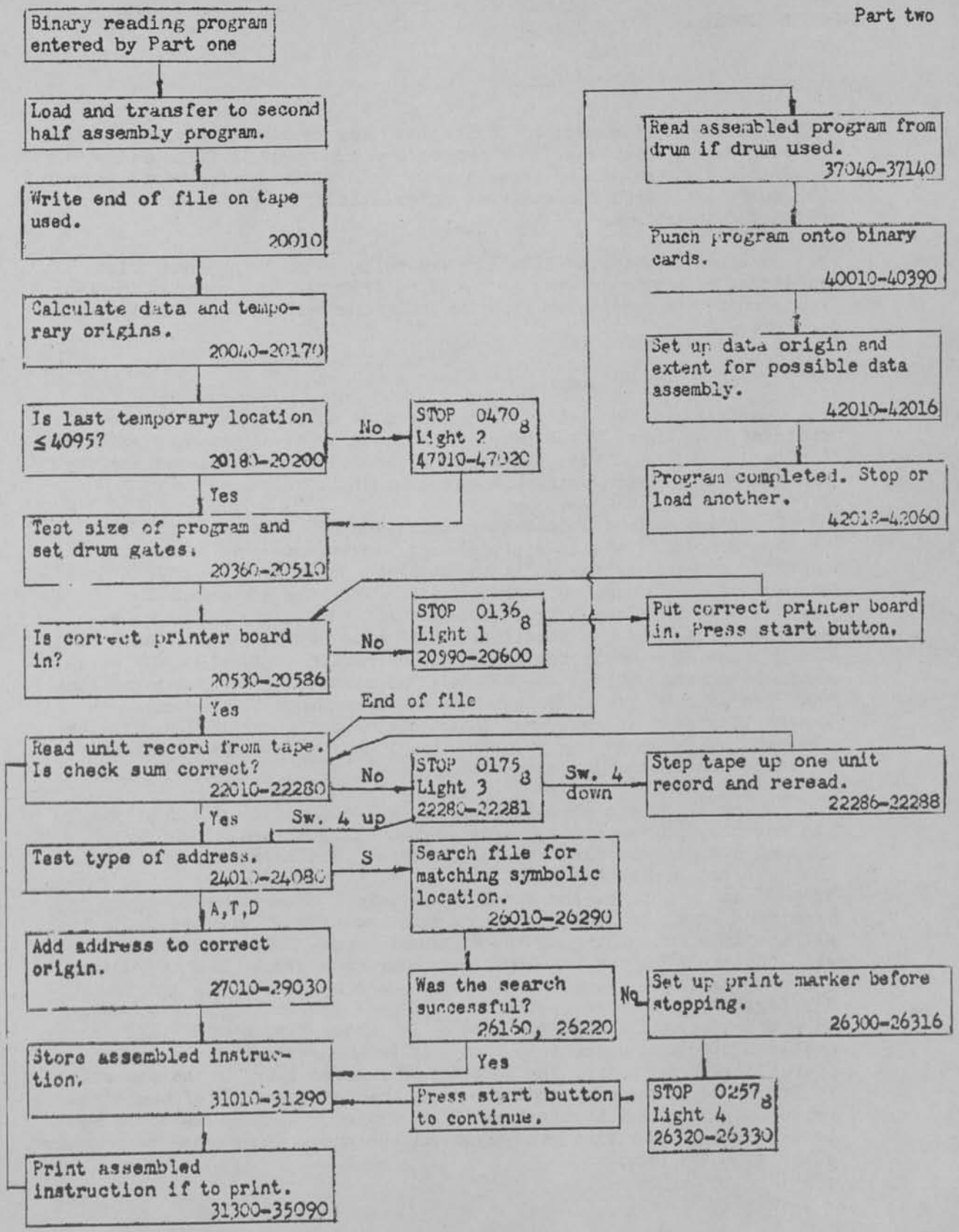
The file of symbolic locations used is located in direct order against the high order end of electrostatic storage. If the number of instructions in the program being assembled is 1850 or less, the assembled instructions are stored beginning in absolute decimal location 396 until they are binary punched. If there are more than 1850 instructions, they are stored on drum #4 in pairs after assembly until they are binary punched.

Block Diagram

Part one



Block Diagram



LOCATION

Consists of five decimal digits which may be used in any way desired as long as the locations for a program are monotonically increasing. A suggested division of the five is to use the first two for region number, the third and fourth for sequence number within the region, and the fifth for insertions.

Consecutive absolute locations beginning with the program origin specified by the programmer as the third from the last instruction are assigned to the successive symbolic locations during the assembly process.

OPERATION

Consists of a two-letter alphabetic code as shown in the attached table of operations. These have been chosen for their mnemonic value and for unique underpunching. The absolute operation codes are assigned by the assembly program during the execution of its first part.

There are two operations shown which are not standard. The first is ET for "Extract." This is a programming convenience only, since the assembly program assigns to it the absolute operation code of the Send Address order. ET must be used only with full-word addresses. The second is YE for "Your Entry." This is used in calling sequences when the address part is the specification of some absolute amount such as a magnitude. The reason for using this operation is found in the North American change routine. The YE operation part tells the change routine that the address part is an absolute amount and not to be changed as though it were an address when insertions or deletions of instructions are made elsewhere in the program. This in no way affects anyone who is not using the change routine.

Any or all of the alphabetic codes may be changed quite easily. The assembly program binary cards punched 0111201 thru 0111203 in columns 1 thru 7 are the data for the first part of the assembly program. The data origin for this first part is 354_8. The operations table runs from D1 11 thru D1 99. The absolute equivalent of any alphabetic operation code should be in the relative data location D1 xx where xx is the alphabetic underpunching of the alphabetic code. This location should also contain 2525_8 for the legitimate operation check. In other words, the entries in the operations table are each a half word in length. The first or high order twelve bits are used for the legitimate operation check amount, 2525_8. The last or low order five bits contain the absolute operation code. Note that this is the reverse of the usual operation-address order. The location of a given entry in the operations table can be computed by adding octally the underpunching of the alphabetic operation code to the base of the table, 354_8. Any new codes must be two-letter codes since the punch pattern check tests these card columns along with the rest.

ALPHABETIC OPERATION CODES

Stop and Transfer	ST
Unconditional Transfer	UT
OVerflow Transfer	OV
Plus Transfer	PT
Zero Transfer	ZT
SUbtract	SU
Reset and Subtract	RS
Subtract as if Plus	SP
No Instruction	NI
ADd	AD
Reset and Add	RA
Add as if Plus	AP
Send Result	SR
Send Address	SA
Send MQ	SQ
Load MQ	LQ
MultiplY	MY
Multiply and Round	MR
DiVide	DV
RouNd	RN
Long Left Shift	LL
Long Right Shift	LR
Accumulator Left Shift	AL
Accumulator Right Shift	AR
ReaD	RD
Read in Reverse	RR
WRite	WR
Write End of File	EF
ReWind	RW
Set Drum Address	DA
Sense and Skip or Control	SS
Copy and Skip	CS
ExtracT	ET
Your Entry	YE

ALPHABETIC TYPE CODES

Absolute	A
Symbolic	S
Temporary	T
Data	D

TYPE

Consists of a two-column description of the address on that instruction. The first column is alphabetic and tells whether the address is written in absolute or is a reference to the symbolic area, data storage area, or temporary storage area. The letters for these are A, S, D, or T respectively.

The second column is a one if the address refers to a half-word location and a two if the address refers to a full-word location. This column is the equivalent of the sign when the instruction is in absolute form.

An advantage of using this type column is that the address column is then left completely free of arbitrary restrictions on the choice of regions.

ADDRESS

Consists of five decimal digits. If the address is a reference to a symbolic location, the five decimal digits must be identical to the five decimal digits of the symbolic location referred to. An address with type "S" which does not have a corresponding symbolic location will result in an error stop.

If the address is a reference to a data or a temporary location, the absolute address will be obtained by adding the written address to the respective origin. The programmer usually assigns data or temporary locations compactly beginning with the respective origin, but this is not a requirement. When either or both of these areas have not been assigned compactly, special care should be taken that the specified number of half words of data or temporary covers both used and unused words in the area.

If the address is a reference to an absolute location, the type is "A" and the absolute address is exactly as written.

If the address is a modifiable address, it should be written as type "A" with brackets in the address columns. The brackets are then keypunched as a numeric X followed by four zeros. It is possible to include absolute, data, or temporary addresses within the brackets, but not symbolic addresses. A reference to full-word data location 2 (later to be modified) would be written D2 [2] and keypunched D2 x0002.

REMARKS

Anything which can be keypunched and for which there are characters on the 716 printer can be written in the remarks column. Thirty-five card columns are reserved for these remarks. They are written onto and

read from the tape without check sum. Only the left side of the card image is included in the tape check sum.

An adequate description of each instruction written in the remarks column as the program is prepared will prove very useful when the program is being tested or changed. These remarks on the listing of the assembled program will be of assistance to the coder during the checkout period.

EXAMPLES

The attached form ED-622-3 serves as an example of the form used at North American for symbolic programming and contains some isolated instructions from the assembly program to illustrate the manner in which the symbolic decimal input is written.

The first instruction has symbolic location 02020 and is a Read operation with absolute address 2048, the address of the card reader.

Instruction 02070 is a Store operation in temporary full word 56. Instruction 08040 is a Transfer on Overflow to symbolic half-word location 08050. Instruction 08250 is a Subtract operation with the address data half-word location 1.

Instruction 12180 is a Divide by data full-word location 6. Instruction 12240 is a Reset and Add operation with address being a modifiable half word. Instruction 22120 is a Copy operation with address being a modifiable full word. Both of these modifiable addresses are initially setup by the program itself and altered by instructions in the same loop.

Finally, instruction 26150 is a Subtract temporary half-word location 58.

SPEEDEX CODE SHEET

PROBLEM: EXAMPLES OF SYMBOLIC INPUT JOB NO. _____ PAGE _____ OF _____

PROGRAMMER: _____ CHECKED BY: _____ DECK NO. _____ DATE: _____

TRANSFER FROM	SYMBOLIC LOCATION	OP.	TYPE	ADDRESS	REMARKS (Limit to 35 columns)	BINARY MAGNITUDE
Do not key-punch	10 14	15	17	19 63	46 80	Do not key-punch
	0 2 0 2 0	R D	A 1	2 0 A 8	PREPARE TO READ CARD	
	0 2 0 7 0	S R	T 2	5 6	CLEAR CK SUM COUNTER	
	0 8 0 4 0	O V	S 1	0 8 0 5 0	RESET OVERFLOW INDICATOR	
	0 8 2 5 0	S U	D 1	1	ALTER DOUBLE SUM ADDRESSES	
	1 2 1 8 0	D V	D 2	6	DIVIDE BY TEN	
	1 2 2 4 0	R A	A 1	[ ]	OBTAIN TRUE OPERATION CODE	
	2 2 1 2 0	C S	A 2	[ ]	COPY LEFT WORD	
	2 6 1 5 0	S U	T 1	5 8	SUBTRACT NEW LOWER	

SYMBOLIC DECIMAL CARD FORM

See "SYMBOLIC INPUT" for more detailed description of symbolic instruction.

Card column	Contents
1-8	Indicative information if desired
9	Must be left blank
10-14	Symbolic location. Must have single numeric punch in each column.
15-16	Alphabetic operation code. Must have an alphabetic punch in both columns.
17	Alphabetic A, S, T, or D for type address. Must be alphabetic.
18	Numeric 1 or 2 for half or full-word address, respectively. Must be numeric.
19-23	Address. Must have a single numeric punch in each column. Numeric X in first column will give bracket. In this case, must not have digit punch also.
24-45	Should be left blank
46-80	Remarks. May be mixed alphabetic, numeric, or blank.

CARD ORDER AND READER BOARD

The symbolic decimal cards must be sorted into reverse order on columns 10 thru 14, the symbolic location. This puts the three program specification cards first. The binary cards for the assembly program are then split into two parts. Cards 01 100 01 and 01 112 01 thru 01 112 08 are put on the front of the reversed symbolic decimal deck. Cards 01 112 09 thru 01 112 19 are put on the back of the reversed decimal deck. The whole deck is then ready for the card reader.

A standard reader board reading card columns 9 thru 80 straight into calculate entries left and right will read both the binary and the decimal cards correctly.

SENSE SWITCHES

Switch #1: Not used by assembly program

Switch #2: Tape selection switch
- Up - Tape #1 used for intermediate storage
- Down - Tape #2 used for intermediate storage

Switch #3: Bad operation correction switch
- Up - Pressing the start button after a non-allowable operation stop will cause the assembly process to continue after a marker has been left indicating that the instruction on which the stop occurred has a bad operation code.
- Down - Pressing the start button after a non-allowable operation stop will cause the card on which the stop occurred to be reread. This assumes that the operator has run the cards under the brushes out, corrected the third from the last, and run it along with the following cards into the reader again.

Switch #4: Tape reading error reread switch
- Up - Pressing the start button after a tape check sum stop will cause the assembly process to continue after a marker has been left indicating that the instruction on which the stop occurred had a tape error.
- Down - The unit record on which the stop occurred will be reread if the start button is pressed. If the original stop was caused by a reading error, a second attempt to read the unit record may very well be successful.

Switch #5: Print switch
- Up - A complete listing of the program being assembled will be printed out at the rate of 150 instructions per minute.
- Down - No listing of the program being assembled will be printed.

Switch #6: Program following switch
- Up - When the assembly process has been completed and the binary cards punched, a program stop will occur.
- Down - When the assembly process has been completed and the binary cards punched, the programmed equivalent of the load button will occur for the purpose of reading in any self-loading program which follows the assembly program.

PROGRAMMED STOPS AND SENSE LIGHTS

For the purpose of being able to quickly ascertain the nature of a stop during the assembly process, one of the sense lights is turned on for each of the stops. The assembly process is broken into two parts, reading cards and printing or reading tape. The part being executed, plus one of the four lights will uniquely identify any one of the eight stops. The stops, associated lights, printing marker if there is one, and recommended course of action are shown below:

While reading cards:

Octal Loc.	Light	Marker	Meaning and Course of Action
0256	1	None	Decimal cards out of sequence. Put in sequence and restart from beginning.

Octal Loc.	Light	Marker	Meaning and Course of Action
0212	2	None	Punch pattern bad. Run cards out, fix third from last, run it and following cards into reader, and press start button.
0237	3	8	Alphabetic operation code bad. Continue or reread bad card. See description of sense switch #3 use.
0303	4	None	File too large. Attempting to assemble more than 3700 instructions. Break up into blocks to assemble.

While printing or reading tape*:

Octal Loc.	Light	Marker	Meaning and Course of Action
0136	1	None	Wrong printer board. Put correct board in and press start button.
0175	2	6	*Tape check sum error. Continue or reread unit record. See description of sense switch #4 use. Check punched binary card for error.
0470	3	None	*Upper limit of electrostatic storage exceeded. Reassign program origin and begin over again.
0257	4	7	*No such location in file for this symbolic address. Press start button to continue.

* In the event one of the last three error stops occurs while assembling without printing (sense switch #5 down), put sense switch #5 up before pressing the start button and let one or two lines print before putting it down again. This will give a record of the instruction being assembled on which the stop occurred.

PRINTER BOARD AND ALTERATION SWITCHES:

Alteration Switch #1

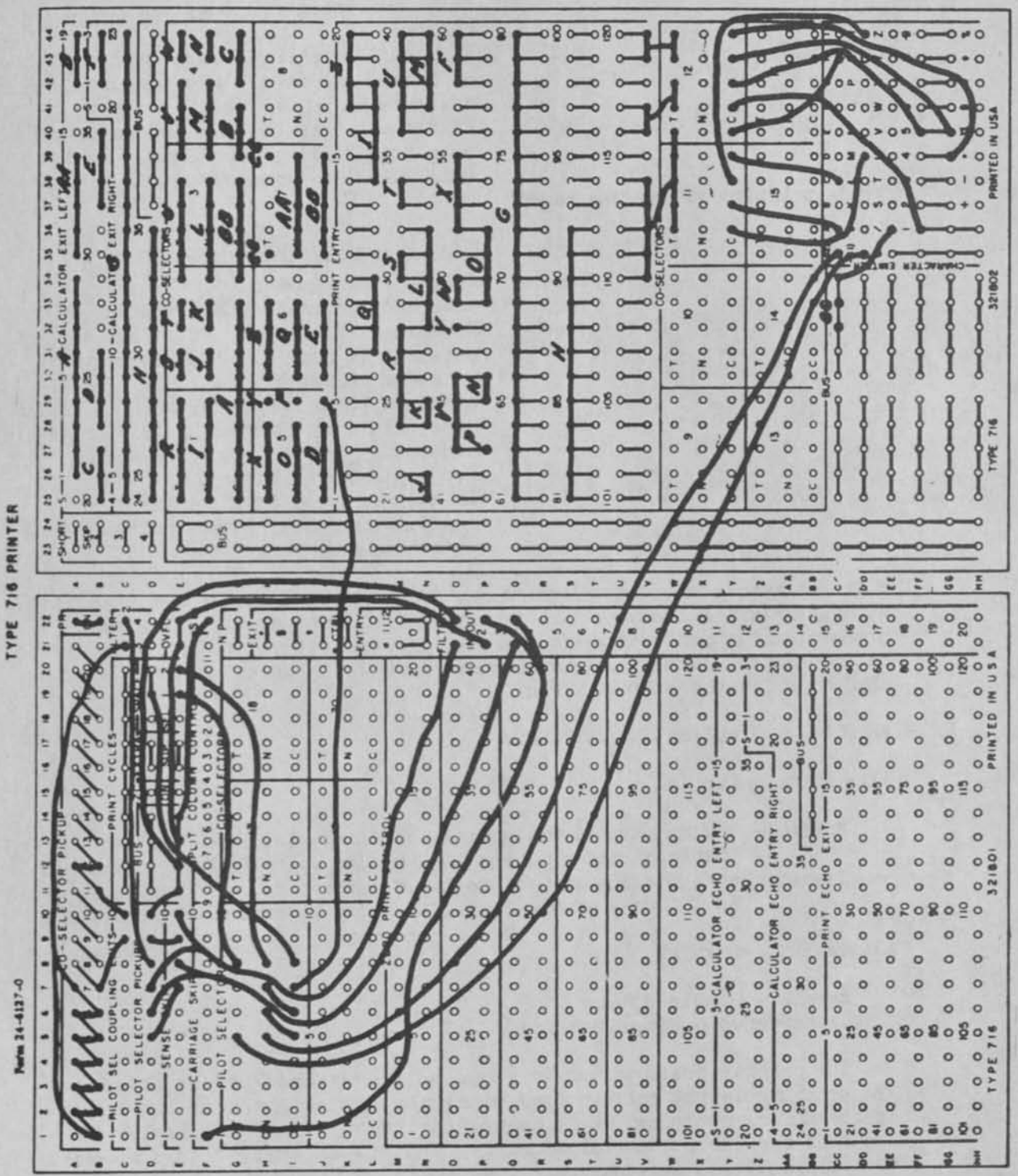
Normal - Spacing across the printed sheet is wider and more open. Somewhat easier to read. Occupies approximately 8 inches.
Transferred - Spacing across the printed sheet is narrower in order to fit on a narrow form. Occupies approximately 7 inches.

Alteration Switch #2

Normal - Double spacing
Transferred - Single spacing

The printer board wiring is shown on the attached wiring diagram. The printing is not echo checked since the critical part of the output is the binary punching. Zero control is not wired. The co-selectors are used for the horizontal spacing. The date is emitted thru a selector controlled by the carriage overflow impulse for printing on the first line of each sheet.

PRINTER BOARD WIRING



DESCRIPTION OF THE LISTING

The attached listing is the assembly program assembled by itself. Any program being assembled will list in this same form if sense switch #5 is up.

The symbolic decimal coding is printed to the left in exactly the same form as written. Brackets are printed as two slashes.

The absolute octal form of the program is printed in the center. Absolute octal was chosen rather than absolute decimal because of the necessity of using octal in connection with the console, memory printouts, etc. The idea of printing both was discarded because of its tendency to confuse.

The remarks or description of the instruction is printed to the right exactly as punched.

If a marker has been left by one of the error stops, it will print to the left of the symbolic decimal on the line with the instruction in error.

PUNCH PANEL

The punch panel is a standard seventy-two column board with calculate exits left and right being punched straight into card columns 9 thru 80.

BINARY CARD FORM AND LOADING

The binary cards punched by the assembly program are in the following form. The nine left row is used as a control word and contains (1) the number of full words on the card, excluding the control word itself and the check sum, in columns 10-14, (2) the address into which the nine right word on the card is to read in columns 15-26, (3) the signal as to whether or not the card is the last of a group, and if it is, to what location control is to transfer when the card has been read. The last is in columns 33-44. It is zero on all except those cards on which a transfer of control away from the reading program is desired, i.e., the last card of a program.

The last half row punched (the 12 right row in the case of a full card) is the check sum for that card. It is the simple sum of all the other half rows on the card with the overflows disregarded. The nine left row is included in the check sum.

The binary cards punched may be loaded for execution with a copy of the first card of the first part of the assembly program. This card is identified with an 01 100 01 in columns 1 thru 7. It is a self-loading binary card-reading program which occupies and makes use of absolute octal locations 0 thru 43. When already in storage it can be entered

for the purpose of reading more binary cards by transferring to absolute location 0006. A check sum error while reading in binary cards will cause a stop at octal location 0036. The difference between the card check sum and the calculated check sum will show in the accumulator at the time of the stop. The card in error should be reread by running it into the reader and transferring to 0006_8 or by reloading card 01 100 01 ahead of it.

Any binary data cards being loaded should precede the program cards since the program will be executed as soon as the last program card has been read.

This binary card form was chosen because it was felt that it would lend itself to the most compact punching and reading programs while at the same time maintaining the advantage that each card is an independent unit record.

Symbolic Decimal				Absolute Octal			Remarks
Loc.	Op.	Type	Addr.	Loc.	Op.	Addr.	
01010	RA	D1	00009	0050	+12	0365	SET STARTING ADDRESS FOR FILE
01020	SA	S1	12430	0051	+15	0272	X
01030	RA	S1	19001	0052	+12	0334	SET ENTRANCE TO INITIAL 3 REGION
01040	SA	S1	14070	0053	+15	0321	X
01050	RA	S1	19008	0054	+12	0343	SET SR ADDR IN INITIAL 3 REGION
01060	SA	S1	16020	0055	+15	0323	X
01080	RA	D1	00006	0056	+12	0362	CLEAR INFORMATION AREA
01090	SR	A2	00036	0057	-14	0044	X
01100	SR	A2	00038	0060	-14	0046	X
01110	RA	D1	00010	0061	+12	0366	SETUP FIRST VALUE FOR SEQUENCE CK
01130	SR	A1	00001	0062	+14	0001	X
01140	SS	A1	00070	0063	+36	0106	CHANGE TAPES IF 2 DOWN
01150	UT	S1	01230	0064	+01	0070	X
01160	RA	S1	19012	0065	+12	0347	X
01170	SA	S1	01230	0066	+15	0070	X
01180	SA	S1	12320	0067	+15	0257	X
01230	RW	A1	00256	0070	+34	0400	INSURE THAT TAPE IS REWOUND
02010	WR	A1	02052	0071	+32	4004	INSURE THAT MQ NOT IN USE
02020	RD	A1	02048	0072	+30	4000	PREPARE TO READ CARD
02030	RA	D1	00006	0073	+12	0362	CLEAR CARD CONVERSION COUNTERS
02040	SR	T2	00050	0074	-14	0602	X
02050	SR	T2	00052	0075	-14	0604	X
02060	SR	T2	00054	0076	-14	0606	X
02070	SR	T2	00056	0077	-14	0610	CLEAR CK SUM COUNTER
02080	RA	S1	19002	0100	+12	0335	RESET INITIAL COPY LOOP ADDRESS
02090	SA	S1	04010	0101	+15	0107	X
02095	SA	S1	04020	0102	+15	0110	X
02100	SA	S1	14010	0103	+15	0313	X
02110	RA	S1	19003	0104	+12	0336	RESET TRANSFER ADDRESS
02120	SA	S1	08180	0105	+15	0160	X
02130	SS	A1	00064	0106	+36	0100	TURN OFF SENSE LIGHTS
04010	CS	A2	/ /	0107	-37	0000	COPY LEFT WORD
04020	RS	A2	/ /	0110	-06	0000	TEST SIGN OF WORD JUST COPIED
04030	PT	A1	00006	0111	+03	0006	END OF FILE IF PLUS
04040	RA	S1	04010	0112	+12	0107	LEFT COPY ADDRESS
04050	SA	S1	05010	0113	+15	0133	ALTER LQ ADDRESS
04060	SA	S1	10010	0114	+15	0174	STORE LEFT WORD ADDRESSES
04070	SU	D1	00001	0115	+05	0355	FOR CK SUM
04080	SA	S1	10020	0116	+15	0175	X
04090	SU	D1	00001	0117	+05	0355	X
04100	SA	S1	04120	0120	+15	0121	ALTER RIGHT WORD COPY ADDRESS
04120	CS	A2	/ /	0121	-37	0000	COPY RIGHT WORD
04125	SU	D1	00002	0122	+05	0356	X
04130	SA	S1	04010	0123	+15	0107	ALTER LEFT WORD COPY ADDRESS
04135	SA	S1	04020	0124	+15	0110	X
04140	RS	S1	02040	0125	+06	0074	RESET ADDRESSES IN CARD
04150	SA	S1	08160	0126	+15	0156	CONVERSION LOOP
04160	SA	S1	08170	0127	+15	0157	X
04170	AD	D1	00001	0130	+11	0355	X
04180	SA	S1	08190	0131	+15	0161	X
04190	SA	S1	08200	0132	+15	0162	X
05010	LQ	A2	/ /	0133	-17	0000	SPACE LEFT ROW IMAGE

```
05020 LL A1 00005   0134 +24 0005  X
05030 AL A1 00001   0135 +26 0001  X
05040 LR A1 00006   0136 +25 0006  X
08010 RA S1 04120   0137 +12 0121  TEST FOR ZERO ROW
08020 SU S1 19004   0140 +05 0337  X
08030 ZT S1 08120   0141 +04 0152  X
08040 OV S1 08050   0142 +02 0143  RESET OVERFLOW INDICATOR
08050 RA D1 00008   0143 +12 0364  SET END OF GROUP INDICATOR
08060 LL A1 00001   0144 +24 0001  DIGIT INTO ACCUMULATOR
08070 OV S2 08150   0145 -02 0155  LOOP END IF LAST DIGIT
08080 SR T2 00048   0146 -14 0600  CONVERSION TO BINARY
08090 AL A1 00002   0147 +26 0002  X
08100 AD T2 00048   0150 -11 0600  X
08110 UT S1 08060   0151 +01 0144  RETURN FOR NEXT DIGIT
08120 RA S1 19005   0152 +12 0340  ALTER TRANSFER FORK ADDRESS
08130 SA S1 08180   0153 +15 0160  X
08140 UT S1 08040   0154 +01 0142  RETURN TO CONVERSION
08150 AL A1 00018   0155 +26 0022  POSITION CONVERTED GROUP
08160 AD A1 /   /   0156 +11 0000  ADD SINGLE SUM CONVERSION CNTRS
08170 SR A1 /   /   0157 +14 0000  X
08180 UT A1 /   /   0160 +01 0000  AVOID DBLE SUMS IF ROW 0 THRU 12
08190 AD A1 /   /   0161 +11 0000  ADD DOUBLE SUM CONVERSION CNTRS
08200 SR A1 /   /   0162 +14 0000  X
08210 RS S1 08170   0163 +06 0157  ALTER SINGLE SUM ADDRESSES
08220 SU D1 00002   0164 +05 0356  X
08230 SA S1 08160   0165 +15 0156  X
08240 SA S1 08170   0166 +15 0157  X
08250 SU D1 00001   0167 +05 0355  ALTER DOUBLE SUM ADDRESSES
08260 SA S1 08190   0170 +15 0161  X
08270 SA S1 08200   0171 +15 0162  X
08280 SU S1 02070   0172 +05 0077  LOOP END TEST
08290 PT S2 08040   0173 -03 0142  CONVERT NEXT GROUP
10010 RS A1 /   /   0174 +06 0000  MINUS LEFT HALF WORD
10020 SU A1 /   /   0175 +05 0000  MINUS RIGHT HALF WORD
10030 AR A1 00018   0176 +27 0022
10040 AD T2 00056   0177 -11 0610  PREVIOUS PARTIAL CK SUM
10050 SR T2 00056   0200 -14 0610  STORE NEW PARTIAL CK SUM
10060 RS S1 10010   0201 +06 0174  TEST FOR END OF CARD
10070 AD S1 19006   0202 +11 0341  X
10080 PT S1 04010   0203 +03 0107  X
12010 RA T1 00050   0204 +12 0602  ADD SINGLE SUM CONVERSION CNTRS
12020 AD T1 00052   0205 +11 0604  X
12030 AD T1 00054   0206 +11 0606  X
12040 SU D1 00003   0207 +05 0357  SUBTRACT CONVERSION TEST SUM
12050 ZT S2 12100   0210 -04 0213  TRANSFER IF CORRECT
12080 SS A1 00066   0211 +36 0102  PUNCH PATTERN ERROR STOP
12090 ST S1 02010   0212 +00 0071  X
12100 RA D1 00006   0213 +12 0362  CLEAR HIGH ORDER OF CONVERTED
12110 SR T1 00050   0214 +14 0602     SYMBOLIC OPERATION AND ADDR
12120 SR T1 00052   0215 +14 0604  X
12130 SR T1 00054   0216 +14 0606  X
12140 LQ T2 00052   0217 -17 0604  OBTAIN SYM OPER AS DIVIDEND
12150 DV D2 00006   0220 -22 0362  DIVIDE BY TEN
```

```
12160 SR A2 00002  0221 -14 0002  FIRST REMAINDER IS SYM SIGN
12170 RA D1 00006  0222 +12 0362  CLEAR ACCUMULATOR
12180 DV D2 00006  0223 -22 0362  DIVIDE BY TEN
12190 SR T2 00048  0224 -14 0600  SECOND REMAINDER IS TYPE
12200 SQ A2 00004  0225 -16 0004  CODED OPERATION AS FULL WORD
12210 RA A1 00005  0226 +12 0005  CODED OPERATION AS HALF WORD
12220 AD S1 19009  0227 +11 0344  ADD BASE OF ASSEMBLY DATA
12230 SA S1 12240  0230 +15 0231  ADDRESS OF TRUE OPERATION CODE
12240 RA A1 /   /  0231 +12 0000  OBTAIN TRUE OPERATION CODE
12245 LQ D1 00006  0232 +17 0362  CLEAR MQ
12250 LR A1 00023  0233 +25 0027  SHIFT OPERATION TO MQ
12260 SU D2 00004  0234 -05 0360  SUBTRACT OPERATION CK CONSTANT
12263 ZT S1 12296  0235 +04 0247  AVOID ERROR INDICATION IF OPERATION
12266 SS A1 00067  0236 +36 0103  ERROR INDICATION NO SUCH OPERATION
12270 ST S1 12273  0237 +00 0240  X
12273 SS A1 00071  0240 +36 0107  CONTINUE OR REREAD BAD OPERATION
12276 UT S1 12283  0241 +01 0243  CONTINUE
12280 UT S1 02020  0242 +01 0072  REREAD BAD OPERATION CORRECTED
12283 RA D1 00001  0243 +12 0355  LEAVE INDICATION OF BAD OPERATION
12286 AR A1 00017  0244 +27 0021  X
12290 AD T2 00004  0245 -11 0524  X
12293 SR T2 00004  0246 -14 0524  X
12296 RA T1 00051  0247 +12 0603  SEQUENCE CHECK OF LOCATIONS
12300 SU A1 00001  0250 +05 0001  X
12303 PT S1 12313  0251 +03 0255  SEQUENCE ERROR IF PLUS
12305 RA T1 00051  0252 +12 0603  REPLACE LAST WITH CURRENT LOCATION
12307 SR A1 00001  0253 +14 0001  X
12310 UT S1 12320  0254 +01 0257  CONTINUE
12313 SS A1 00065  0255 +36 0101  SEQUENCE ERROR
12316 ST S1 01010  0256 +00 0050  X
12320 WR A1 00256  0257 +32 0400  PREPARE TO WRITE ON TAPE
12330 LL A1 00035  0260 +24 0043  SHIFT OPERATION BACK TO ACCUM
12340 AD T1 00049  0261 +11 0601  ADD TYPE OF ADDRESS
12350 AD T2 00054  0262 -11 0606  ADD SYM ADDR TO OPER AND TYPE
12360 SR T2 00048  0263 -14 0600  STORE OP AND TYPE AND ADDR
12370 RA A2 00002  0264 -12 0002  SYMBOLIC SIGN
12380 AR A1 00001  0265 +27 0001  SHIFT FOR 1 OR 2 ODD OR EVEN
12390 ZT S1 12420  0266 +04 0271  AVOID MINUS ACTION
12400 RS T2 00048  0267 -06 0600  ATTACH MINUS SIGN IF NEGATIVE
12410 SR T2 00048  0270 -14 0600  X
12420 RA T1 00051  0271 +12 0603  SYMBOLIC LOCATION
12430 SR A1 /   /  0272 +14 0000  STORE IN FILE
12440 RA S1 12430  0273 +12 0272  FILE ADDRESS
12450 SA A1 00039  0274 +15 0047  LAST FILE ADDRESS LEFT HERE
12460 SA T1 00050  0275 +15 0602  FILE ADDRESS FOR TAPE WRITING
12470 SU D1 00001  0276 +05 0355  ALTER FILE ADDRESS
12480 SA S1 12430  0277 +15 0272  X
12490 SU S1 19013  0300 +05 0350  SUBTRACT FILE LOWER LIMIT
12500 PT S2 13010  0301 -03 0304  AVOID ERROR INDICATION
12530 SS A1 00068  0302 +36 0104  ERROR FILE TOO LARGE
12540 ST S1 01010  0303 +00 0050  X
13010 RS T1 00048  0304 +06 0600  COMPLETE CHECK SUM
13020 SU T1 00049  0305 +05 0601  X
```

```
13030 SU T1 00050   0306 +05 0602  X
13040 SU T1 00051   0307 +05 0603  X
13050 AR A1 00018   0310 +27 0022  X
13060 AD T2 00056   0311 -11 0610  X
13070 SR T2 00052   0312 -14 0604  X
14010 CS A2 /   /   0313 -37 0000  COPY WORD
14020 RA S1 14010   0314 +12 0313  ALTER COPY ADDRESS
14030 SU D1 00002   0315 +05 0356  X
14040 SA S1 14010   0316 +15 0313  X
14050 SU S1 19010   0317 +05 0345  CHECK FOR END OF LOOP
14060 PT S1 14010   0320 +03 0313  REMAIN IN LOOP IF PLUS
14070 UT A1 /   /   0321 +01 0000  FIRST THREE CARD GATE
16010 RA T1 00049   0322 +12 0601  PLACE INITIAL INFORMATION
16020 SR A1 /   /   0323 +14 0000  X
16030 RA S1 16020   0324 +12 0323  ALTER SR ADDRESS
16040 SU D1 00001   0325 +05 0355  X
16050 SA S1 16020   0326 +15 0323  X
16060 SU S1 19011   0327 +05 0346  TEST FOR LAST INITIAL SPEC
16070 PT S1 02010   0330 +03 0071  RETURN TO NEXT CARD IF PLUS
16080 RA S1 16100   0331 +12 0333  STOP USE OF REGION 16
16090 SA S1 14070   0332 +15 0321  X
16100 UT S2 02010   0333 -01 0071  RETURN TO READ NEXT CARD
19001 ST S1 16010   0334 +00 0322  ENTRANCE TO INITIAL 3 REGION
19002 ST T1 00000   0335 +00 0520  9 LEFT ROW IMAGE ADDRESS
19003 ST S1 08190   0336 +00 0161  DOUBLE SUM ADDITION INSTRUCTION
19004 CS T2 00038   0337 -37 0566  0 RIGHT ROW IMAGE ADDRESS
19005 ST S1 08210   0340 +00 0163  AVOID DOUBLE SUM TRANSFER ADDR
19006 RS T1 00044   0341 +06 0574  END OF CARD TEST ADDRESS
19007 CS T2 00048   0342 -37 0600  CARD IMAGE END TEST ADDRESS
19008 ST A1 00038   0343 +00 0046  FIRST SR ADDRESS IN INITIAL 3
19009 ST D1 00000   0344 +00 0354  BASE OF DATA REGION
19010 CS T2 00054   0345 -37 0606  TAPE WRITING LOOP END TEST
19011 SR A1 00036   0346 +14 0044  LAST INITIAL 3 TEST WORD
19012 RW A1 00257   0347 +34 0401  TAPE 2 ADDRESS
19013 SR A1 00396   0350 +14 0614  FILE LOWER LIMIT TEST
19997 ST A1 00040   0351 +00 0050  PROGRAM ORIGIN
19998 ST A1 00100   0352 +00 0144  NUMBER HALF WORDS DATA
19999 ST A1 00058   0353 +00 0072  NUMBER HALF WORDS TEMPORARY
```

DATA

Location	Contents	Binary Magnitude
D1 0	4096	17
D1 1	1	17
D1 2	2	17
D1 3	Conversion Test Sum	17
D2 4	Operation Test Constant	35
D1 6	0	17
D1 7	10	17
D1 8	8	17
D1 9	4095	17
D1 10	131071	17
D1 11 ⎫		
. ⎪		
. ⎬	Operation Table	
. ⎪		
D1 99 ⎭		

TEMPORARY

Location	Use
T2 0 ⎫	
. ⎪	
. ⎬	Card Image
. ⎪	
T2 46 ⎭	
T2 48	Conversion Working Storage Address Type Operation, Type, and Address
T1 50 ⎫	
. ⎪	
. ⎬	Conversion Summing Locations
. ⎪	
T1 55 ⎭	
T2 56	Partial Check Sum

```
20001 SS A1 00070   0050 +36 0106  CHANGE TAPES IF 2 DOWN
20002 UT S1 20010   0051 +01 0056  X
20003 RA S1 49010   0052 +12 0501  X
20004 SA S1 20010   0053 +15 0056  X
20005 SA S1 22010   0054 +15 0140  X
20006 SA S1 22286   0055 +15 0200  X
20010 EF A1 00256   0056 +33 0400  WRITE END OF FILE ON TAPE
20020 WR A1 00512   0057 +32 1000  PREPARE TO IDENTIFY PRINTER BOARD
20030 SS A1 00521   0060 +36 1011  X
20040 RA A1 00036   0061 +12 0044  PROGRAM LOCATION MINUS FILE LOCATIO
20050 SU A1 00039   0062 +05 0047  X
20060 SR T1 00054   0063 +14 0604  X
20080 AD D1 00003   0064 +11 0511  ADD 4096
20090 AD D1 00001   0065 +11 0507  ADD ONE TO CARRY IF ODD
20100 AR A1 00019   0066 +27 0023  FORCE LOW ORDER BIT TO ZERO
20110 AL A1 00019   0067 +26 0023  X
20120 SR T1 00055   0070 +14 0605  ORIGIN DATA STORAGE
20130 AD A1 00037   0071 +11 0045  ADD NUMBER HALF WORDS DATA
20140 AD D1 00001   0072 +11 0507  ADD ONE TO CARRY IF ODD
20150 AR A1 00019   0073 +27 0023  FORCE LOW ORDER BIT TO ZERO
20160 AL A1 00019   0074 +26 0023  X
20170 SR T1 00056   0075 +14 0606  ORIGIN TEMPORARY STORAGE
20180 AD A1 00038   0076 +11 0046  ADD NUMBER HALF WORDS TEMPORARY
20190 SU D1 00007   0077 +05 0515  SUBTRACT 4097
20200 PT S1 47010   0100 +03 0467  ELECTROSTATIC EXCEEDED IF PLUS
20310 RA A1 00039   0101 +12 0047  OBTAIN THE LOWER SEARCH EXTREME
20320 SU D1 00001   0102 +05 0507  X
20330 SR T1 00057   0103 +14 0607  X
20340 RA D1 00000   0104 +12 0506  SET FIRST DA ADDRESS TO ZERO
20350 SA S1 31230   0105 +15 0311  X
20360 RA T1 00055   0106 +12 0605  OBTAIN NUMBER HALF WORDS IN PROG
20370 SU A1 00036   0107 +05 0044  X
20380 SR T1 00059   0110 +14 0611  X
20390 SU D1 00006   0111 +05 0514  TEST SIZE OF PROGRAM
20400 PT S1 20450   0112 +03 0117  LARGE PROGRAM IF PLUS
20410 RA S1 31180   0113 +12 0305  SET DRUM GATES FOR SMALL PROGRAM
20420 SA S1 31150   0114 +15 0302  X
20430 RA S1 49009   0115 +12 0500  X
20440 UT S1 20480   0116 +01 0122  X
20450 RA S1 49007   0117 +12 0476  SET DRUM GATES FOR LARGE PROGRAM
20460 SA S1 31150   0120 +15 0302  X
20470 RA S1 49008   0121 +12 0477  X
20480 SA S1 37030   0122 +15 0370  X
20490 RA S1 31270   0123 +12 0315  SET DRUM READ TEST ADDRESS
20500 SU T1 00059   0124 +05 0611  X
20510 SA S1 49006   0125 +15 0475  X
20530 SS A1 00073   0126 +36 0111  SKIPS IF NO PRINTING DESIRED
20535 UT S2 20550   0127 -01 0131  TRANSFER IF PRINTING DESIRED
20540 UT S2 20610   0130 -01 0137  NO PRINTING DESIRED
20550 SS A1 00065   0131 +36 0101  TURN ON SENSE LITES
20584 MY A1 00000   0132 +20 0000  ALLOW TIME FOR SELECTOR PICKUP
20585 MY A1 00000   0133 +20 0000  X
```

```
20586 MY A1 00000  0134 +20 0000  X
20590 SS A1 00522  0135 +36 1012  IS CORRECT PRINTER BOARD IN
20600 ST S1 20030  0136 +00 0060  STOP IF NOT
20610 SS A1 00064  0137 +36 0100  TURN OFF SENSE LITES IF SO
22010 RR A1 00256  0140 +31 0400  PREPARE TO READ IN REVERSE
22020 CS T2 00052  0141 -37 0602  COPY TAPE CHECK SUM
22030 UT S1 22050  0142 +01 0144  AVOID END OF FILE OUT
22040 UT S2 37010  0143 -01 0366  END OF FILE OUT
22050 CS T2 00050  0144 -37 0600  COPY WORD
22060 RA S1 22050  0145 +12 0144  SET FIRST ADDRESS OF COPY LOOP
22070 AD D1 00002  0146 +11 0510  ALTER COPY ADDRESS
22080 SA S1 22120  0147 +15 0153  X
22090 SA S1 22130  0150 +15 0154  STORE ADDRESS FOR CK SUM
22100 SU D1 00001  0151 +05 0507  X
22110 SA S1 22140  0152 +15 0155  X
22120 CS A2 /   /  0153 -37 0000  COPY LEFT WORD
22130 RA A1 /   /  0154 +12 0000  LEFT WORD AGAINST CK SUM
22140 AD A1 /   /  0155 +11 0000  X
22150 AR A1 00018  0156 +27 0022  X
22160 AD T2 00052  0157 -11 0602  X
22170 SR T2 00052  0160 -14 0602  X
22180 RA S1 22120  0161 +12 0153  SET RIGHT COPY ADDRESS
22190 AD D1 00002  0162 +11 0510  X
22200 SA S1 22210  0163 +15 0164  X
22210 CS A2 /   /  0164 -37 0000  COPY RIGHT WORD
22220 UT S1 22070  0165 +01 0146  CONTINUE IN LOOP
22230 SR A1 00396  0166 +14 0614  INSTRUCTIONAL CONSTANT
22240 RA T1 00050  0167 +12 0600  COMPLETE CK SUM CHECK
22250 AD T1 00051  0170 +11 0601  X
22260 AR A1 00018  0171 +27 0022  X
22270 AD T2 00052  0172 -11 0602  X
22275 ZT S1 24010  0173 +04 0206  ZERO TRANSFER IF CORRECT
22280 SS A1 00067  0174 +36 0103  TURN ON LITE 3 FOR TAPE ERROR
22281 ST S1 22282  0175 +00 0176  ERROR STOP TAPE ERROR
22282 SS A1 00072  0176 +36 0110  REREAD OR GO ON
22284 UT S1 22290  0177 +01 0202  LEAVE MARKER AND GO ON
22286 RD A1 00256  0200 +30 0400  STEP BACK
22288 UT S1 20610  0201 +01 0137  REREAD
22290 RA D1 00001  0202 +12 0507  ERROR INDICATION TAPE ERROR
22300 AR A1 00015  0203 +27 0017  X
22303 AD T2 00012  0204 -11 0532  X
22306 SR T2 00012  0205 -14 0532  X
24010 RA S1 24030  0206 +12 0210  ORIGIN OF TRANSFER REGION
24020 AP T1 00048  0207 +13 0576  OPERATION AND TYPE
24030 SA S1 24040  0210 +15 0211  STORE TRANSFER FORK ADDRESS
24040 UT A1 /   /  0211 +01 0000  TRANSFER TO TYPE TRANSFER
24050 UT S2 29010  0212 -01 0264  TRANSFER FOR ABSOLUTE TYPE
24060 UT S2 26010  0213 -01 0216  TRANSFER FOR SYMBOLIC TYPE
24070 UT S2 27010  0214 -01 0260  TRANSFER FOR TEMPORARY TYPE
24080 UT S2 28010  0215 -01 0262  TRANSFER FOR DATA TYPE
26010 RA T1 00057  0216 +12 0607  OBTAIN LOWER EXTREME
26020 SR T1 00052  0217 +14 0602  STORE AS WORKING LOWER
26030 RA D1 00003  0220 +12 0511  OBTAIN UPPER EXTREME
```

```
26040 SR T1 00053  0221 +14 0603  STORE AS WORKING UPPER
26050 RA T1 00052  0222 +12 0602  BEGINNING OF SEARCH LOOP
26060 AD T1 00053  0223 +11 0603  SUM OF UPPER AND LOWER EXTREMES
26070 AR A1 00001  0224 +27 0001  MEAN OF UPPER AND LOWER EXTREMES
26080 SR T1 00058  0225 +14 0610  STORE MEAN
26090 SA S1 26100  0226 +15 0227  STORE MEAN AS ADDRESS
26100 RA A1 /   /  0227 +12 0000  CONTENTS MEAN LOCATION
26110 SU T1 00049  0230 +05 0577  SUBTRACT SYMBOLIC ADDRESS
26120 ZT S1 26260  0231 +04 0247  OUT OF LOOP IF SEARCH COMPLETED
26130 PT S1 26200  0232 +03 0241  AVOID MINUS ACTION
26140 RA T1 00052  0233 +12 0602  RESET ADD OLD LOWER
26150 SU T1 00058  0234 +05 0610  SUBTRACT NEW LOWER
26160 PT S1 26300  0235 +03 0252  ERROR NO SUCH SYMBOLIC ADDRESS
26170 RA T1 00058  0236 +12 0610  MEAN BECOMES NEW LOWER
26180 SR T1 00052  0237 +14 0602  STORE NEW LOWER
26190 UT S1 26060  0240 +01 0223  RETURN FOR NEXT TRIAL
26200 RA T1 00058  0241 +12 0610  RESET ADD THE MEAN
26210 SU T1 00053  0242 +05 0603  SUBTRACT OLD UPPER
26220 PT S1 26300  0243 +03 0252  ERROR NO SUCH SYMBOLIC ADDRESS
26230 RA T1 00058  0244 +12 0610  RESET ADD THE MEAN
26240 SR T1 00053  0245 +14 0603  STORE NEW UPPER
26250 UT S1 26050  0246 +01 0222  RETURN FOR NEXT TRIAL
26260 RA S1 26100  0247 +12 0227  RESET ADD FINAL SEARCH LOCATION
26270 AD T1 00054  0250 +11 0604  OBTAIN ACTUAL ADDRESS
26290 UT S2 29030  0251 -01 0266  END OF INSTRUCTION ASSEMBLY
26300 RA D1 00001  0252 +12 0507  NO SUCH SYMBOLIC ADDRESS
26310 AR A1 00016  0253 +27 0020  X
26313 AD T2 00008  0254 -11 0526  X
26316 SR T2 00008  0255 -14 0526  X
26320 SS A1 00068  0256 +36 0104  X
26330 ST S1 31010  0257 +00 0267  X
27010 RA T1 00056  0260 +12 0606  ORIGIN OF TEMPORARY
27020 UT S1 29020  0261 +01 0265  X
28010 RA T1 00055  0262 +12 0605  ORIGIN OF DATA
28020 UT S1 29020  0263 +01 0265  X
29010 RA D1 00000  0264 +12 0506  ZERO
29020 AD T1 00049  0265 +11 0577  ADD SYMBOLIC ADDRESS
29030 SA T1 00048  0266 +15 0576  STORE ACTUAL ADDRESS
31010 RA T1 00050  0267 +12 0600  FILE LOCATION
31020 AD T1 00054  0270 +11 0604  OBTAIN ACTUAL LOCATION
31030 SR T1 00050  0271 +14 0600  REPLACE FILE LOC WITH ACTUAL LOC
31040 RA T1 00048  0272 +12 0576  ACTUAL INSTRUCTION
31050 SR A1 00394  0273 +14 0612  STORE IN TEMPORARY LOCATION
31060 RA S1 31050  0274 +12 0273  ALTER TEMPORARY LOCATION BY 1
31070 AD D1 00001  0275 +11 0507  X
31080 SA S1 31050  0276 +15 0273  X
31081 SA S1 31083  0277 +15 0301  NEXT TEMPORARY LOCATION
31082 RA D1 00000  0300 +12 0506  ZERO
31083 SR A1 /   /  0301 +14 0000  CLEAR NEXT TEMPORARY LOCATION
31150 UT A1 /   /  0302 +01 0000  PRESET DRUM GATE
31160 RS S1 31050  0303 +06 0273  TEMPORARY LOCATION
31170 AD S1 22230  0304 +11 0166  TEMPORARY TEST LOCATION
31180 PT S1 31300  0305 +03 0320  TRANSFER IF PAIR NOT READY
```

```
31190 WR A1 00131   0306 +32 0203  PREPARE TO WRITE ON DRUM
31200 RA S1 31270   0307 +12 0315  RESET TEMPORARY PAIR ADDRESS
31220 SA S1 31050   0310 +15 0273  X
31230 DA A2 /   /   0311 -35 0000  SET DRUM ADDRESS
31240 RA S1 31230   0312 +12 0311  MODIFY DRUM ADDRESS
31250 SU D1 00002   0313 +05 0510  X
31260 SA S1 31230   0314 +15 0311  X
31270 CS A2 00394   0315 -37 0612  COPY WORD
31280 RA D1 00000   0316 +12 0506  RESET PAIR LOCATION TO ZERO
31290 SR A2 00394   0317 -14 0612  X
31300 SS A1 00073   0320 +36 0111  SKIPS IF NO PRINTING DESIRED
31305 UT S2 31320   0321 -01 0323  PRINTING DESIRED
31310 UT S1 20610   0322 +01 0137  NO PRINTING DESIRED
31320 WR A1 00512   0323 +32 1000  PREPARE TO WRITE PRINTER
31330 RA T1 00040   0324 +12 0566  11 LEFT ROW IMAGE
31340 AL A1 00011   0325 +26 0013  X
31350 ZT S1 33010   0326 +04 0330  TRANSFER IF NO X
31360 SS A1 00520   0327 +36 1010  PICKUP PRINT SELEC FOR PAREN
33010 RA D1 00002   0330 +12 0510  COLUMN INDICATOR
33020 SR T2 00052   0331 -14 0602  X
33025 RA T1 00048   0332 +12 0576  ASSEMBLED INSTRUCTION
33030 PT S1 33040   0333 +03 0335  AVOID PICKING SELECTOR IF PLUS
33035 SS A1 00518   0334 +36 1006  PICK FOR MINUS
33040 LR A1 00035   0335 +25 0043  SHIFT TO MQ
33050 RA T1 00050   0336 +12 0600  ACTUAL LOCATION
33060 AR A1 00017   0337 +27 0021  X
33070 LR A1 00013   0340 +25 0015  SHIFT INTO MQ
33080 LL A1 00003   0341 +24 0003  BEGIN CONVERSION LOOP
33090 AL A1 00020   0342 +26 0024  FOUR TIMES DIGIT EQUIV TO ADDR
33100 AD S1 49001   0343 +11 0471  ADD CARD IMAGE BASE
33110 SA S1 33160   0344 +15 0351  CORRECT ROW IMAGE ADDRESS
33120 SA S1 33170   0345 +15 0352  X
33130 RA T2 00052   0346 -12 0602  ALTER COLUMN INDICATOR
33140 AR A1 00001   0347 +27 0001  X
33150 SR T2 00052   0350 -14 0602  X
33160 AD A2 /   /   0351 -11 0000  ADD CORRECT ROW IMAGE
33170 SR A2 /   /   0352 -14 0000  STORE IN CORRECT ROW IMAGE
33180 AL A1 00027   0353 +26 0033  TEST FOR END OF CONVERSION
33190 ZT S1 33080   0354 +04 0341  REMAIN IN LOOP IF ZERO
35010 RA S1 49002   0355 +12 0472  SET UP FIRST COPY ADDRESS
35020 SA S1 35030   0356 +15 0357  X
35030 CS A2 /   /   0357 -37 0000  COPY WORD
35040 RA S1 35030   0360 +12 0357  ALTER COPY ADDRESS
35050 SU D1 00002   0361 +05 0510  X
35060 SA S1 35030   0362 +15 0357  X
35070 SU S1 49003   0363 +05 0473  TEST FOR END OF LOOP
35080 PT S1 35030   0364 +03 0357  REMAIN IN LOOP
35090 UT S1 20610   0365 +01 0137  TRANSFER TO NEXT TAPE READ
37010 WR A1 00512   0366 +32 1000  EJECT LAST PAGE IN PRINTER
37020 SS A1 00519   0367 +36 1007  EJECT LAST PAGE IN PRINTER
37030 UT A1 /   /   0370 +01 0000  PRESET DRUM GATE
37040 RA S1 49005   0371 +12 0474  WRITE LAST WORD ON DRUM
37050 SR S1 31300   0372 +14 0320  X
```

```
37060 UT S1 31190  0373 +01 0306  X
37070 RD A1 00131  0374 +30 0203  READ RECORD FROM DRUM
37080 DA A2 00000  0375 -35 0000  X
37090 CS A2 00394  0376 -37 0612  X
37100 RA S1 37090  0377 +12 0376  X
37110 SU D1 00002  0400 +05 0510  X
37120 SA S1 37090  0401 +15 0376  X
37130 SU S1 49006  0402 +05 0475  X
37140 PT S1 37090  0403 +03 0376  X
40010 RA D1 00005  0404 +12 0513  HALF AND FULL WORD COUNT FOR
40020 SR T1 00053  0405 +14 0603     FIRST CARD
40030 AL A1 00011  0406 +26 0013  X
40040 AD A1 00036  0407 +11 0044  1ST ADDRESS FOR 1ST CARD INTO
40050 SR T2 00000  0410 -14 0516  X
40060 RA T1 00059  0411 +12 0611  BEGINNING OF PUNCH LOOP
40070 ZT S2 42010  0412 -04 0456  OUT IF PUNCHING FINISHED
40080 WR A1 01024  0413 +32 2000  PREPARE TO PUNCH
40090 SU D1 00005  0414 +05 0513  SUBTRACT 44
40100 ZT S1 40120  0415 +04 0417  TO LAST CARD ACTION
40110 PT S1 40210  0416 +03 0431  AVOID LAST CARD ACTION
40120 RA A1 00036  0417 +12 0044  FIRST TO EXELX
40130 AR A1 00018  0420 +27 0022  X
40140 AD T1 00000  0421 +11 0516  FIRST TO READ INTO
40150 LR A1 00030  0422 +25 0036  X
40160 RA T1 00059  0423 +12 0611  HALF AND FULL WORD COUNT FOR LAST
40170 SR T1 00053  0424 +14 0603  X
40180 AR A1 00019  0425 +27 0023  X
40190 LR A1 00005  0426 +25 0005  X
40195 SQ T2 00000  0427 -16 0516  STORE CONTROL WORD FOR LAST CARD
40200 RA D1 00000  0430 +12 0506  CLEAR ACCUMULATOR
40210 SR T1 00059  0431 +14 0611  NUMBER HALF WORDS LEFT
40220 CS T2 00000  0432 -37 0516  COPY 0 LEFT ROW
40230 RA T2 00000  0433 -12 0516  START CK SUM
40240 SR T2 00002  0434 -14 0520  X
40250 RS S1 40290  0435 +06 0441  SETUP TEST LOOP WORD
40260 AD T1 00053  0436 +11 0603  X
40270 SR T1 00052  0437 +14 0602  X
40280 RA T2 00002  0440 -12 0520  PARTIAL CK SUM
40290 CS A2 00394  0441 -37 0612  COPY WORD
40300 AD A2 00394  0442 -11 0612  ADD TO CK SUM
40310 SR T2 00002  0443 -14 0520  X
40320 RA S1 40290  0444 +12 0441  ALTER LOOP ADDRESSES
40330 SU D1 00002  0445 +05 0510  X
40340 SA S1 40290  0446 +15 0441  X
40350 SA S1 40300  0447 +15 0442  X
40360 AD T1 00052  0450 +11 0602  TEST FOR END OF LOOP
40370 PT S1 40280  0451 +03 0440  REMAIN IN LOOP IF PLUS
40380 CS T2 00002  0452 -37 0520  COPY CK SUM
40383 RA T2 00000  0453 -12 0516  FIRST INTO FOR NEXT CARD
40386 AD T1 00053  0454 +11 0603  X
40390 UT S1 40050  0455 +01 0410  RETURN TO PUNCH NEXT CARD
42010 RA T1 00055  0456 +12 0605  SETUP DATA ORIGIN
42012 SR A1 04094  0457 +14 7776  X
```

```
42014 RA A1 00037  0460 +12 0045  SETUP NUMBER HALF WORDS DATA
42016 SR A1 04095  0461 +14 7777  X
42018 SS A1 00074  0462 +36 0112  SKIP IF ANOTHER PROGRAM READY
42020 ST A1 00006  0463 +00 0006  PROGRAM FINISH
42040 RD A1 02048  0464 +30 4000  EQUIVALANT OF LOAD BUTTON
42050 CS A2 00000  0465 -37 0000  X
42060 UT A1 00000  0466 +01 0000  X
47010 SS A1 00066  0467 +36 0102  FS CAPACITY EXCEEDED ERROR
47020 ST S1 20310  0470 +00 0101  X
49001 ST T2 00036  0471 -00 0562  0 LEFT ROW IMAGE ADDRESS
49002 ST T2 00000  0472 -00 0516  9 LEFT ROW IMAGE ADDRESS
49003 CS T2 00048  0473 -37 0576  CARD IMAGE END TEST ADDRESS
49005 UT S1 37070  0474 +01 0374  DRUM WRITE RETURN ADDRESS
49006 CS A2 /   /  0475 -37 0000  DRUM READ COPY LOOP TEST
49007 ST S1 31160  0476 +00 0303  DRUM GATE ADDRESSES
49008 ST S1 37040  0477 +00 0371  X
49009 ST S1 40010  0500 +00 0404  X
49010 ST A1 00257  0501 +00 0401  TAPE 2 ADDRESS
49997 ST A1 00040  0502 +00 0050  PROGRAM ORIGIN
49998 ST A1 00008  0503 +00 0010  NUMBER HALF WORDS DATA
49999 ST A1 00060  0504 +00 0074  NUMBER HALF WORDS TEMPORARY
```

DATA

Location	Contents	Binary Magnitude
D1 0	0	17
D1 1	1	17
D1 2	2	17
D1 3	4096	17
D1 4	256	17
D1 5	44	17
D1 6	Drum Use Test Amt.	17
D1 7	4097	17

TEMPORARY

Location	Use
T2 0 ⎫ . ⎪ . ⎬ . ⎪ T2 46 ⎭	Card Image
T2 48	Operation, Type, and Address
T2 50	File and Symbolic Locations
T2 52	Tape Check Sum Print Conversion Column Indicator
T1 52	Working Search Lower Extreme Punching Loop Test Word
T1 53	Working Search Upper Extreme Half Word Count for Card
T1 54	Program Origin minus File Origin
T1 55	Origin Data Area
T1 56	Origin Temporary Area
T1 57	File Lower extreme
T1 58	Working Search Mean
T1 59	Number Half Words to Punch