USEful Notes

Number 1

15 October 1956

SUBJECT: A Minimum Service Routine Library for the 1103A. (RR)

Bioctal paper-tapes of this library are available upon request to:

LEO KENNEDY
REMINGTON RAND UNIVAC
1902 W. MINNEHAHA AVE.
ST. PAUL W4, MINNESOTA

UNIVAC SCIENTIFIC 1103A

MINIMUM

SERVICE ROUTINE LIBRARY

Entry	Service Routine	Storage
70000	-----	
70001	Bioctal Loading Routine	75170-75356
70002	Flex Code Loading Routine	74520-74730
70003	-----	
70004	-----	
70005	Flex Dump	73760-74247
70006	Bioctal Dump	74253-74501
70007	-----	
70010	-----	
70011	-----	
70012	-----	
70013	Changed Word Post Mortem	75370-75552
70014	-----	
70015	-----	
70016	Single Breakpoint Stop	75560-75573
70017	Automatic Sampler	73255-73750
70020	-----	
70021	-----	
70036	Common Exit	

PROGRAMMING AND OPERATION CONVENTIONS

I. Drum Image of HSS:

Drum cells 76000-77777 are reserved for the image of 00000-01777 of HSS. This image is used by most service routines as temporary storage for part of HSS while the service routine operates from HSS. The programmer is advised not to load into the image as this may result in incorrect loading of HSS. The programmer may use this part of drum storage as a temporary pool or work space during the operation of his program, but in so doing deprives himself of the use of Changed Word Post-Mortem.

II. Drum Storage for the Service Library

Drum cells 70000-75700 are reserved for the Service Library and are not, in general, available for program use. Loading programs into the range 70000-70037 deprives the programmer of all facilities of the Service Library, while loading into the range 70040-75777 *may* deprive him of only part of the Service Library.

COMMENTS ON USE OF SERVICE LIBRARY

I. Paper Tape Preparation

a) Bioctal tapes should have two 7th level punches at the very end of the tape.

b) Flex code (absolute) program tapes should have at least one 7th level punch at the very end.

c) Flex dump tapes are suitable for reloading via Flexie. Be sure that a 7th level punch is present at the end of the tape.

II. Loading Routines "Transfer Control" Option

Both loading routines have a "transfer control" option. The following procedure will effect the transfer for either load routine.

(1) Set program tape in reader

(2) MASTER CLEAR

(3) Set the computer on MAIN PULSE Ø.

(4) Manually insert the following into PCR

37 70036 70001 (2)

(5) Set PAK = program start

(6) START.

PROGRAM ENTRIES TO SERVICE ROUTINES

The block of cells 70000-70037 is reserved for entries to the service routines. Cell 70036 is reserved as the common exit from those service routines which by their nature admit program entry and exit. For example, the use of the Bioctal Loading Routine as a subroutine would be effected by the instruction 37 70036 70001. All required parameter words must be placed in the appropriate registers before entry is made to the particular service routine by a Return Jump instruction. For example, the use of the Bioctal Dump would be effected by the following sequence:

```
            n:  11 (x) 31000
          n+1:  37 70036 70006
          n+2:  -- ----- -----
where, say (x) = 00 00001 01777
```

CAUTION: Since the service routines each have only <u>one</u> entry, any inadvertent (or not) loading in the range 70000-70037 deprives one of <u>all</u> the service routines.

SERVICE ROUTINES

Bioctal Loading Routine

The routine will load anywhere. Loading into HSS and 76000-77777 can result in incorrect loading of HSS. A sum check is made whenever the input tape contains an insert to 75202, followed by a double precision check sum and a check address of 75204. Note: cells 75202 and 75203 will not be loaded with the sum.

Operating Instructions:

(1) Set PAK = 70001; START.

(2) Computer halts on 56 00000 70001 after completing read in. START to load another tape.

(3) Two consecutive seven-level punches in the trailer should be present. If these punches are not present, the following procedure may be used: FORCE STOP after the paper tape has passed through the reader, MASTER CLEAR, START at 00032. The last block of information read in is then stored in its proper location.

(4) Errors

(a) Machine prints "t" and halts. The loading routine is not in HSS correctly and must be restored. START causes another transfer to HSS. If the check fails again, reload the service library onto MD.

(b) Machine prints "c" and halts. A check address has failed. STARTing ignores this error and routine proceeds as though error had not occurred. A check address failure should not be ignored as it is very likely that the paper tape is in error.

(c) Machine prints "m" and halts. Check sum has failed to agree with computed sum of data read in. START to ignore this error and continue loading.

Flex Code Loading Routine

This routine is designed to load Flex Code tape prepared on a Flexowriter in the conventional fashion for translating to bioctal. It operated in the same fashion as the bioctal loading Routine. A sum check is made whenever the input tape contains an insert to 75202, 75203. (See above).

Operating Instructions:

(1) Set PAK = 70002; START

(2) Computer halts on 56 00000 70002 after completing read in. START to load another tape.

(3) At least one seven-level punch should be present in the trailer to stop the routine. If this punch is not present, the following procedure may be used.

(a) FORCE STOP after the paper tape has passed through the reader.

(b) MASTER CLEAR; set PAK = 00025; START.

(4) Errors:

(a) Machine prints "t" and halts. The loading routine is not in HSS correctly and must be restored. START causes another transfer to HSS. If the check fails again, reload the service library onto MD.

(b) Machine prints "c" and halts. A check address has failed. A START ignores this error and routine proceeds as if no error had occurred.

(c) Machine prints "m" and halts. A check sum has failed to agree with computed sum of data read in. START to ignore the error.

Flex Dump

This routine dumps the contents of consecutive storage cells on punched paper tape only. Automatic page editing is provided and every eighth address is given. The punched tape is suitable for re-loading via the Flex Code loading routine. A check sum is punched out at the end of the dump. (A_L), (A_R), (Q) are not restored or punched out. HSS is restored.

Operating Instructions:

(1) Enter in Q_u the address of the first cell to be dumped.

Enter in Q_v the address of the last cell to be dumped.

If a seven-level punch stop code is desired at the end of the dump set Q_{35}=1.

(2) Turn ON the High Speed Punch.

(3) Set PAK = 70005; START.

(4) The machine halts on 56 00000 70005 providing a re-entry for another dump.

(5) Errors:

(a) Machine prints "t" and halts. The dump routine is not in HSS correctly. START causes another transfer to HSS. If the check fails again, reload the service library onto MD.

(b) Machine prints "p" and halts. An illegal parameter word has been set up in Q and is displayed there. Clear Q manually and insert correct parameter; START

(6) This routine dumps only one tape of storage at a time, either HSS or drum. 76000 to 76314 is used as an image region for 00000 to 00314.

[Bioctal Dump]

This routine will dump onto paper tape in bioctal form the contents of any specified number up to 7777_8 of consecutive storage cells in HSS or the drum except 76000-77777. A check sum is automatically punched at the end of the dump. A double seven-level punch at the end of the tape is optional.

Operating Instructions:

(1) Enter in Q_u the address of the first cell to be dumped.
Enter in Q_v the address of the last cell to be dumped.
If a double seven-level stop code is to be punched following this dump, set $Q_{35} = 1$.

(2) Turn High Speed Punch ON.

(3) Set PAK = 70006; START.

(4) The stop at the end of the dump, 56 00000 70006, provides a re=entry for another dump. The contents of A and Q are not retained. HSS is restored at the end of the routine.

Changed Word Post Mortem

This routine is designed to compare the contents of 00000 to 01777 of HSS with its image at 76000-77777. The image contains (unless disturbed) the original contents of HSS as read into the computer. Those words in HSS which have been changed by the execution of the program are the only ones reported out.

Operating Instructions:

(1) Turn High Speed Punch ON.

(2) Set PAK = 70013; START.

(3) Compare halts on 56 00000 70013.

The following will be punched in Flex Code.

a) (Q)

b) (A_R)

c) (A_L)

d) Any changed word according to the following:

HSS word   Image word   HSS address

At the end of the routing, (A), (Q) and HSS are restored.

(4) This routine uses the cells 74740-75137 as temporary storage for part of HSS while the routine operates.

Single Breakpoint Stop

This routine permits one to select a single address of a program which one can run on high speed and stop before executing the instruction at that address. One may then sample the results of computation to date or step through several instructions. Restriction: the breakpoint instruction must be one which is not modified by the program.

Operating Instructions:

(1) enter in Q_v the breakpoint address.

enter in A_v the entry address for the program.

(2) Set PAK = 70016; START. The program will be executed up to the breakpoint at which time the computer will halt on 56 00000 70016, providing a re-entry for another breakpoint stop.

[Automatic Sampler (Sam-0)]

This routine provides for the printing or punching (in octal or decimal) the contents of any selected cells at selected check points. Output is suppressed for the first N_p times through the check point and after $(N_p + N_s)$ times. The program which is being sampled is executed normally between check points. It is not necessary to provide for sampling while writing the program. The programmer stores in any available block of memory a list of information regarding check points, cells to be sampled, or scales.

A) Operating Instructions:

(1) Read in the program to be sampled. This is the unmodified problem program.

(2) Read in the "Sampling List Tape". See below for description of this tape.

(3) Set PAK = 70017 START. The routine sets up check points and transfers control to α which is contained in the sampling list.

B) Sampling List Tape (Flex or Bioctal).

This tape loads cell 73643 and the sampling list which contains a number of sublists, one for each check point. Each sublist contains all information necessary for sampling at one check point. This information may be stored in any convenient set of consecutive HSS or MD cells except 00100 through 00167.

The sampling list tape format is as follows:

		73643	XX	L_o	L_f
Fixed Storage					
one complete sublist	Check point address	L_o	00	00000	c.p.
	Index Word	L_1	00	N	N_s
	Parameter words	L_2	0a	M_p	s
		.	.	.	.
		.	.	.	.
		.	0a	M	s
	End word	.	70	00000	00000
	End word	.	70	00000	00000
		.	00	00000	c.p.
		.	00	N_p	N_s
		.	0a	M	s
		.	.	.	.
		.	.	.	.
		.	70	00000	00000
		L_f	70	00000	α

(1) Fixed storage - The word XX L_o L_f read into 73643, gives the address of the first cell (L_o) and the last cell (L_f) of the sampling list. Printing or punching is specified by XX, 61 for printing and 63 for punching.

(2) Check point address - The word 00 00000 c.p. gives the check point address, c.p. Sampling occurs before execution of the instruction at c.p.

(3) Index word - The word 00 N_p N_s gives two 5-octal-digit numbers, N_p, signifying the number of times the check point is to be passed before sampling starts, and N_s, the number of times sampling is to occur at the check point.

(4) Parameter words - These are of the form Oa M s where the first octal digit is always zero. The second octal digit, a, takes on the values

0 for octal output

1 for decimal output.

If a > 1, the parameter word is ignored. The u-portion of each parameter word contains the octal address, M, of a cell whose contents are to be sampled. If M = 32001, (A_L) is sampled. If M is not a machine address the parameter word is ignored. The v-portion of each parameter word contains the binary scale factor, s, of the contents of M. $0 \leq s < 70$. If $s \geq 70$, "2 small" is printed.

(5) End words - The last two words of each sublist are of the form 70 00000 00000 with the exception of the second end word of the last sublist (L_f), which is 70 00000 α. SAM-0 jumps to α after setting up check points on a 70017 start.

C) Output

Shown below is an example of sampler output where the check point address was 00303.

```
00303
00075      12 34567 12340
00076      77 03124 65432
00100       1.23456789017
00101    -321.098632812
00102    993059913.000
00103       0.43210987653
32001      14 00000 00000
31000      37 37373 73737
```

D) Restrictions

(1) The word initially stored at a check point must be an

instruction; it must not be a repeat command or a repeated instruction and it may not be written into or out of at any time during the course of the program.

(2) The Sampling List Tape must not load into cells 01777 or 02000, i.e., it must be on one side or the other of this point.

APPENDIX

Paper Tape Loading Routines

Introduction

Paper tape is described as being divided into rows and columns: a single column of positions across the width of a tape is called a frame. Frames are divided parallel to the length of the tape into seven levels. Six of these levels are used primarily to represent information to be placed in computer storage, while the seventh level is used to represent loading directions. A hole punched in any of the six data levels of the tape represents a one, while the absence of a hole represents a zero.

Words punched onto paper tape are of three kinds: enter data words, insert address words, and check address words. Enter data words are those which contain information to be stored internally by the computer, while insert address words and check address words are used only for loading purposes The insert address contains the address at which the first data word of a block of consecutive data words is to be stored, while the check address contains the address plus one at which the last data word has been stored.

For a check sum of data on the tape, the following four words should appear on the tape after the data to which the sum applies:

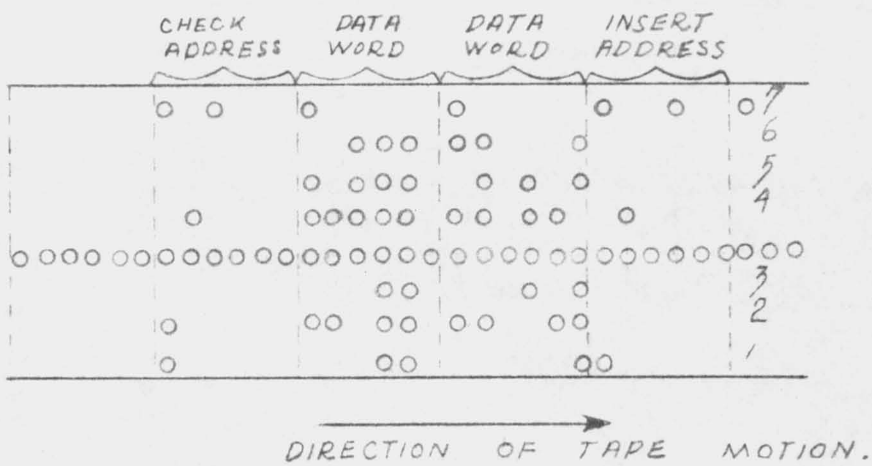
1. Insert address 75202
2. High order of 36 bits of check sum
3. Low order of 36 bits of check sum
4. Check address 75204

The check sum must be the sum of all the data on the tape following the preceding check sum. The check sum will not be loaded into 75202 and 75203. These words will be left undisturbed. Since a check sum test is performed whenever a check address of 75204 is encountered, 75204 should not be used for any other check address.

Bioctal Loading Routine

Words containing 12 octal digits each are punched onto bi-octal tape two digits to a frame. Thus, six frames of tape are necessary to represent one bi-octal coded word. When the bi-octal loading routine is used, the three kinds of words described above are distinguished by the fames in which seventh level holes are located. Each enter data word must have a seventh level hole in the sixth frame. Insert address words and check address words also have seventh level-holes in the sixth frame, but in addition, the insert address has one in the third frame and the check address has one in the fourth frame. Furthermore, should there be a gap on the tape between the two blocks, a seventh level hole must be punched in the frame directly preceding the first frame of the next insert address word on the tape. This also applies to the very first block of data on the tape. The tape should always begin with an unpunched leader of about 10 inches, and should end with an unpunched trailer of about the same length. A seven level punch in any two consecutive frames of the trailer directs the Bioctal Loader routine to store the data read in thus far, transfer the location 76000-77777 to 00000-01777, and come to a programmed stop.

The following is a diagram of a tape bearing the two data words 671234007252 and 00777701232, which are to be stored at address 01001 and 01002 respectively.



Flex Code Loading Routine

Input tapes using this loader are usually prepared directly from the electric typewriter (Flexowriter) in coded form, one character to a frame. In order to obtain a correct loading format, it is important that periods be used only in standard positions.

Each data word must consist of twelve octal digits between two periods. These twelve digits are grouped as follows: two digits followed by a space, five digits followed by a space, and the last five digits followed by a period and carriage return. Insert and check addresses also consist of twelve digits but have periods and digit groupings that are peculiar to each. An insert address reading from left to right is composed of a period, six zero digits, a period, a zero, and a five digit address followed by a period and carriage return. A check address is grouped from left to right as follows: seven zero digits, the most significant digit of a five digit address, a period and the remaining four digits of the address followed by a period and carriage return.

Example:

Insert address	.000000.076050.	(address of first data word)
Data word	00 00400 00500.	
Check address	00000007.6051.	(address following last data word)

Incorrect word format will result in that word not being interpreted correctly, or not being stored in its proper location. Errors of this type will usually show up as check address failures.

One seven level punch should be used at the end of the tape after the last check address to stop the reader and complete the routine. There should be no other seven level punch since any such punch will halt the reader regardless of its position on the tape.

USEful Note Number 2 1 November 1956

SUBJECT: APL Complex Arithmetic Package
(Not a standard USE routine)

This subroutine converts the 1103A temporarily to a machine with three-address logic, separate storage for instructions and data, working on complex floating point numbers. At each entry it can handle interpretive programs containing up to 512 instructions and up to 512 data.

The data are in the form Z = x + iy, where x and y are standard 1103AF floating point numbers and are stored in two successive machine storage cells.

The instructions are in the form

op L ABC,

where op is a 2-octal-digit operation code, L is an address modification index of 1 octal digit, and A, B, C are each 3 octal digits indicating instruction addresses, data addresses, or integers, according to the operation code. Each instruction therefore consists of 12 octal digits and can be stored in one machine storage cell.

If u is the actual machine address of the first cell used for data storage and v is the actual machine address of the first cell used for instruction storage, then the relation between machine addresses and pseudo-addresses is as follows:

Data		Instructions	
u + 0 }	000	v + 0	000
u + 1 }		v + 1	001
u + 2 }	001	v + 2	002
u + 3 }		etc.	
u + 4 }	002		
u + 5 }			
etc.			

The instruction stored at instruction address 000 (i.e. at machine address v) is the first one executed when the complex arithmetic package is called in by writing

```
RJ R L016
00 u v
NI
```

<u>The pseudo-instructions</u>

00 0 000 000 000	Leave the interpretive system and execute NI in basic machine language.
01 L A B C	Perform the operation indicated by the code A upon the contents of data address B and store the result at data address C. (See Table 1)
02 L A B C	Add the contents of data address A to the contents of data address B and store in data address C.
03 L A B C	Subtract the contents of data address B from the contents of data address A and store in data address C.
04 L A B C	Multiply the contents of data address A by the contents of data address B and store in data address C.
05 L A B C	Multiply the contents of data address A by the contents of data address B, reverse the sign of the product, and store in data address C.
06 L A B C	Divide the contents of data address A by the contents of data address B and store in data address C.
07 L A B C	Set the contents of data addresses A, B, and C equal to zero.
10 L A B C	Here A is an integer. Transfer A consecutive data from the block starting at data address B to the block starting at data address C. In case the two blocks overlap, no datum is overwritten until after it has been transferred.
11 L A B C	Here A is an integer. Transfer A consecutive instructions from the block starting at instruction address B to the block starting at instruction address C. In case the two blocks overlap, no instruction is overwritten until after it has been transferred.
12 L 000 000 C	Transfer control to instruction address C.
13 L A B C	Put the three octal digits A in the C-address portion of the instruction at address B and transfer control to instruction address C.
14 L A B C	Put the three octal digits A in that portion of the instruction at address C which is indicated (see Table 2) by the code B.

15 L	A	B	C	Add the integer A to that portion of the instruction at address C which is indicated (see Table 2) by the code B.
16 L	A	B	C	Subtract the integer A from that portion of the instruction at address C which is indicated (see Table 2) by the code B.
17 L	A	B	C	Here A and B are integers and C is an instruction address. Add one to A. If then $A < B$ jump to instruction C. If, however, $A \geqslant B$ then subtract B from all those addresses (in all instructions from the C-th to the one preceding this one) which were modified as a result of an L-code different from zero. Then set A=000 and take next instruction.
20 L	A	B	C	If the contents of data address A are less in absolute value than the contents of data address B then transfer control to instruction address C; otherwise take next instruction.
21 L	A	B	C	If the real part of the contents of data address A is negative, transfer control to instruction address B; otherwise transfer control to instruction address C.
22 L	A	B	C	If the imaginary part of the contents of data address A is negative, transfer control to instruction address B; otherwise transfer control to instruction address C.

The L-code

The L-code of three bits in each instruction permits any of the addresses in that instruction to be increased by one each time the instruction is carried out, as described in Table 3. This increase of addresses is done after the operation is carried out but before going on to the next instruction.

The L-code for each instruction, together with the loop instruction 17LABC, provides a very simple and surprisingly versatile method for coding loops. The following almost trivial example may help to indicate how the coding might go in more complicated and more interesting cases. It shows, in particular, that loops within loops can be handled with a minimum of step and reset operations.

Example $y = \sum_{i=1}^{5} a_i \sum_{j=1}^{i} b_{ij} x_j$

Data address

100-104	$x_1 - x_5$
105	b_{11}
106-107	$b_{21} - b_{22}$
110-112	$b_{31} - b_{33}$
113-116	$b_{41} - b_{44}$
117-123	$b_{51} - b_{55}$
124-130	$a_1 - a_5$
131	$b_{ij} x_j$
132	$\sum b_{ij} x_j$
133	$a_i \sum b_{ij} x_j$
134	$\sum a_i \sum b_{ij} x_j = y$

Instruction address	op	L	A	B	C	comment
100	14	0	105	100	103	reset b_{ij}
101	07	0	134	134	134	clear 134
102	07	0	132	132	132	clear 132
103	04	6	105	100	131	$b_{ij} x_j \rightarrow 131$
104	02	0	131	132	132	$\sum b_{ij} x_j \rightarrow 132$
105	17	2	000	001	103	loop
106	04	4	124	132	133	$a_i \sum b_{ij} x_j \rightarrow 133$
107	02	0	133	134	134	$\sum a_i \sum b_{ij} x_j \rightarrow 134$
110	15	4	001	100	103	step b_{ij}
111	17	0	000	005	102	loop

Relation to USE program

The subroutine here described is being coded in a form somewhat different from the USE standard form for local convenience. It will be noticed, for example, that the parameter word 00 u v is put after the return jump used to call in the subroutine rather than, for instance, in the accumulator; this can of course easily be changed if desired. The other departures from standard (such as use of local sub-subroutines for mathematical functions, etc.) are in much the same category now that the standard compiler permits subroutines to call in other subroutines. In short, the conversion to standard USE form could easily be made if the organization wanted it done.

The logical portion of the subroutine, which interprets the operation codes and does the housekeeping, has purposely been kept separate from the mathematical portion, which actually does the addition, subtraction, etc. This was done so that the same logical portion could serve for different arithmetics, such as real floating point double precision, real stated point double precision, complex double precision, matrix arithmetic, etc., as the required mathematical coding for the several operations becomes available. The advantage of a common logic for these various kinds of arithmetics needs no insistence.

If USE decides not to include this among the standard routines, the coding in present form will be made available (when checked out) to any individual members upon request. In any event, APL will welcome suggestions and criticisms.

Acknowledgement

The debt owed by the present routine to the interpretive system devised by Bell Telephone Laboratories for the IBM 650 (see IBM Tech. Newsletter No. 11) will be obvious to those who are familiar with that system, and is hereby brought to the attention of those who are not.

Table 1: Operations performed by 01LABC

A	Operation performed on contents of B
000	absolute value
001	exponential
002	logarithm (principal value)
003	conversion rectangular to polar form
004	conversion polar to rectangular form
005	multiply by -1.

Table 2: B-codes for operations 14, 15, 16

B (octal)	Portions of instruction C which are modified
000	none
001	C address
010	B address
011	B and C addresses
100	A address
101	A and C addresses
110	A and B addresses
111	A, B and C addresses.

Table 3: L-codes

L (octal)	L (binary)	Addresses affected
0	000	none
1	001	C
2	010	B
3	011	B, C
4	100	A
5	101	A, C
6	110	A, B
7	111	A, B, C

USEful Notes

Number 3

6 November 1956

<u>Subject:</u> Preliminary Information on the

Content of Registers

Execution Times

for the 1103A Floating Point Instructions (RR)

PRELIMINARY INFORMATION

ON THE

CONTENT OF REGISTERS OF THE

1103A FLOATING POINT INSTRUCTIONS

Date: 15 October 1956

Prepared by: P. Warburton

Issued by: Systems Analysis Dept.
Systems Group of Univac Scientific Applications

October 12, 1956

The layout of the "Floating Point Content of Registers" is not the same as that of the fixed point instructions. There are more conditions affecting the final content of A. First, has the NEFF been set or cleared by instruction 05? Second, what is the relative size of (u) and (v)? For these reasons, only the Pack and Unpack commands are in the usual format. Since the arithmetic Floating Point commands do not change (u) and (v), $(u)_f$ and $(v)_f$ are not included in the Contents of Registers of operations 64, 65, 66, 67, 01, and 02.

The binary point of floating point numbers is usually between the twenty-eighth and the twenty-seventh place. After the arithmetic pseudo-normalizing process, the mantissa is in A_L, and the binary point is between A_{63} and A_{62}. It may or may not be normalized. The position of the most significant bit (MSB) indicates what has occurred. If normalized, the MSB will be in A_{62}.

The value of the significant bits depends upon whether rounding has occurred. Rounding in effect adds an extra bit to the value of (a) at A_{35} (unless the addition of the rounding bit carries into A_{62}, in which case the final left shift is omitted and the rounding bit remains added to the value of A_{34}.

The value of $(Q)_f$ will be either (1) the normalized rounded, and packed result (NRP), or (2) the pseudo-normalized result (PN).

NOTE: If A or Q is the v-address of any floating point command other than the pack or unpack command (A) or (Q) will be destroyed by the Unpack (u) sequence before the unpack (v) sequence is reached.

Instruction: Floating Add (FAuv) | **Operation: 64**

Function: Form in Q the normalized rounded and packed floating point sum of (u) and (v).

NE FF	Arithmetic Conditions		$(A)_f$ MSB	$(A)_f$ Value of significant bits	$(A)_f$ Round	$(Q)_f$ Norm	$(Q)_f$ Value
0	$(u) \geq (v)$		A_{62}	$(u_m).2^{(u_c) - (v_c)} + (v_m)$	yes	NRP	(u) + (v)
	$(u) < (v)$		A_{62}	$(v_m).2^{(v_c) - (u_c)} + (u_m)$	yes	NRP	(u) + (v)
1	$(u) \geq (v)$	$(u_c)-(v_c) \geq 2$	A_{61}	$(u_m).2^{(u_c) - (v_c)} + (v_m)$	no	PN	(u) + (v)
		$(u_c)-(v_c) < 2$	$A_{61}-A_{33}$	$(u_m).2^{(u_c) - (v_c)} + (v_m)$	no	PN	(u) + (v)
	$(u) < (v)$	$(v_c)-(u_c) \geq 2$	A_{61}	$(v_m).2^{(v_c) - (u_c)} + (u_m)$	no	PN	(u) + (v)
		$(v_c)-(u_c) < 2$	$A_{61}-A_{33}$	$(v_m).2^{(v_c) - (u_c)} + (u_m)$	no	PN	(u) + (v)

Instruction: Floating Subtract (FSuv)	Operation: 65

Function: Form in Q the normalized, rounded and packed floating point difference of (u) and (v).

NE FF	Arithmetic Conditions		$(A)_f$ MSB	Value of significant bits	Round	$(Q)_f$ Norm	Value
0	$(u) \geq (v)$		A_{62}	$(u_m).2^{(u_c)-(v_c)}-(v_m)$	yes	NRP	(u) - (v)
	$(u) < (v)$		A_{62}	$(v_m).2^{(v_c)-(u_c)}-(u_m)$	yes	NRP	(u) - (v)
1	$(u) \geq (v)$	$(u_c)-(v_c) \geq 2$	A_{61}	$(u_m).2^{(u_c)-(v_c)}-(v_m)$	no	PN	(u) - (v)
		$(u_c)-(v_c) < 2$	$A_{61}-A_{33}$	$(u_m).2^{(u_c)-(v_c)}-(v_m)$	no	PN	(u) - (v)
	$(u) < (v)$	$(v_c)-(u_c) \geq 2$	A_{61}	$(v_m).2^{(v_c)-(u_c)}-(u_m)$	no	PN	(u) - (v)
		$(v_c)-(u_c) < 2$	$A_{61}-A_{33}$	$(v_m).2^{(v_c)-(u_c)}-(u_m)$	no	PN	(u) - (v)

Instruction: Floating Point Multiply (MPuv)	Operation: 66
Function: Form in Q the normalized rounded and packed floating point product of (u) and (v).	

NE FF	Arithmetic Condition	$(A)_f$ MSB	$(A)_f$ Value of significant bits	$(A)_f$ Round	$(Q)_f$ Norm	$(Q)_f$ Value
0		A_{62}	$(u_m).(v_m)$	yes	NRP	(u).(v)
1	$(u_m).(v_m) \geq \frac{1}{2}$	A_{61}	$(u_m).(v_m)$	no	PN	(u).(v)
	$(u_m).(v_m) < \frac{1}{2}$	A_{60}	$(u_m).(v_m)$	no	PN	(u).(v)

Instruction: Floating Point Divide (FDuv)	Operation: 67

Function: Form in Q the normalized, rounded and packed floating point quotient of $(u) \div (v)$

NE FF	Arithmetic Condition	$(A)_f$			$(Q)_f$	
		MSB	Value of significant bits	Round	Norm	Value
0		A_{62}	$(u_m) \div (v_m)$	yes	NRP	$(u) \div (v)$
1	$(u_m) \div (v_m) \geq 1$	A_{61}	$(u_m) \div (v_m)$	no	PN	$(u) \div (v)$
	$(u_m) \div (v_m) < 1$	A_{60}	$(u_m) \div (v_m)$	no	PN	$(u) \div (v)$

Instruction: Floating Point Polynomial Multiply (FPuv)	Operation: 01

Function: Form in Q the sum of (v) and the product of $(Q)_i \cdot (u)$

(NE FF should be cleared for the execution of this instruction. If it is not the product mantissa will be rounded not with one, but with (A_L).)

NE FF	Arithmetic Condition	$(A)_f$			$(Q)_f$	
		MSB	Value of mantissa	Round	Norm	Value
0	$(Q)\ (u) \geq (v)$	A_{62}	$(Qu)_m \cdot 2^{(Qu)c-(vc)} + (v_m)$	yes	NRP	$(Q)_i(u)+(v)$
	$(Q)\ (u) < (v)$	A_{62}	$(v_m) \cdot 2^{(vc)-(Qu)c} + (Qu)_m$	yes		

Instruction: Floating Point Inner Product (FIuv)	Operation: 02

Function: Form in Q the normalized, rounded and packed sum of (Q_i) and the product of (u) and (v).

(NE FF should be cleared for the execution of this instruction; if it is not, the product mantissa will be rounded, not with <u>one</u>, but with (A_L).)

NE FF	Arithmetic Condition	$(A)_f$			$(Q)_f$	
		MSB	Value of significant bits	Round	Norm	Value
0	$(Q)_i \geq (u)\ (v)$	A_{62}	$(Qm)_i \cdot 2^{(Qc)_i-(uv)_c}+(uv)_m$	yes	NRP	$(Q)_i+(u)(v)$
	$(Q) < (u)\ (v)$	A_{62}	$(uv_m) \cdot 2^{(uv)_c-(Qc)_i}+(Qm)_i$	yes	NRP	$(Q)_i+(u)(v)$

Instruction: Floating Point Unpack (UPuv)	Operation: 03

Function: Unpack (u) replacing (u) with (u_m) and replacing (v_c) with (u_c) or its complement if (u) is negative. The characteristic portion of $(u)_f$ contains sign bits. The sign and mantissa bits of $(v)_f$ are cleared to zero.

Storage Class		Contents of Register & Storage Position After Operation					
		$(MC)_f$ or $(MD)_f$		$(A)_f$			$(Q)_f$
u	v	u	v	MSB	Value of bits	Round	
MC or MD	MD or MC	(u_m)	(u_c)		No change		No change
	A	(u_m)	——	A_{34}	(u_c)	no	No change
	Q	(u_m)	——		No change		(u_c)
A	MD or MC	——	(u_c)	A_{26}	$(A_m)_i$	no	No change
	A	——	——	A_{34}	$(A_c)_i$	no	No change
	Q	——	——	A_{26}	$(A_m)_i$	no	$(A_c)_i$
Q	MD or MC	——	(u_c)		No change		$(Q_m)_i$
	A	——	——	A_{34}	$(Q_c)_i$	no	$(Q_m)_i$
	Q	——	——		No change		$(Q_c)_i$

Instruction: Normalize, Round, & Pack (NPuv)	Operation: 04

Function: Replace (u) with the normalized rounded packed floating point number obtained from the possibly unnormalized mantissa in $(u)_i$ and the biased characteristic in $(v)_c$.

It is assumed that $(u)_i$ has the binary point between u_{27} and u_{26} ($(u)_i$ is scaled 2^{-27}).

Storage Class		Contents of Registers & Storage Position After Operation					
		$(MC)_c$ or $(MD)_f$		$(A)_f$			$(Q)_f$
u	v	u	v	MSB	Value of bits	Round	
MC or MD	MD or MC	$NRP(u)+(v_c)$	No change	A_{62}	$(u_m)_f$	yes	No change
	A	$NRP(u)+(v_c)$	——	A_{62}	$(u_m)_f$	yes	No change
	Q	$NRP(u)+(v_c)$	——	A_{62}	$(u_m)_f$	yes	No change
A	MC						
	MC	——	No change	A_{34}	$NRP(A_R)_i+(v_c)$	yes	No change
	A	——	——	A_{34}	$NRP(A_R)_i+(A_{Rc})_i$	yes	No change
	Q	——	——	A_{34}	$NRP(A_R)_i+(Q_c)$	yes	No change
Q	MD or MC	——	No change	A_{62}	$(u_m)_f$	yes	$NRP(Q)_i+(v_c)$
	A	——	——	A_{62}	$(u_m)_f$	yes	$NRP(Q)_i+(A_{Rc})_i$
	Q	——	——	A_{62}	$(u_m)_f$	yes	$NRP(Q)_i+(Q_c)_i$

Instruction: Floating Point Normalize Exit (NEj)	Operation: 05

Function: If J=0 clear the normalize exit flip-flop (designated NFF); if j-1 set NFF to 1

(a) The results of setting NFF to 1 is set forth in the "Contents of Registers"

(b) When NFF is set to 1, it will remain set until cleared by another NEj - instruction

(c) NFF must be cleared for FP, FI, and NP instructions

PRELIMINARY INFORMATION

ON THE

EXECUTION TIME OF THE

1103A FLOATING POINT INSTRUCTIONS

Date: 15 October 1956

Prepared by: F. Warburton

Issued by: Systems Analysis Dept.
Systems Group of Univac Scientific Applications

SUMMARY OF EXECUTION TIMES

		Max.	Min.
Floating add and subtract	$(N \leq 1)$	300 μsec.	144 μsec.
$(N) = (u_c) - (v_c)$	$(N \geq 2)$	236	148
Floating multiply		380	162
Floating divide		654	648
Floating polynomial multiply		619	262
Floating inner product		637	280
Floating unpack		52	54
Floating normalize pack		180	144
Normalize exit		20	20

All times given include magnetic core reference time. If (u) is A, subtract 6 μsec.; if Q, subtract 4 μsec. All cases are for NE FF set to zero. If NE FF = 1, set K = 0 and subtract 12 μsec. All cases include rounding. If the full number of normalizing shifts are made (35 for addition and subtraction, 2 for multiplication and division), the mantissa is zero and rounding is omitted. For this case, subtract 14 μsec.

The following symbols are used in the formulas.

$N = (u_c) - (v_c)$ for operations 64, 65, 66, and 67

$N = (Qu)_c - (v_c)$ for operation 01

$N = (uv)_c - (Q_c)$ for operation 02

N_9 is the sign of this difference (corresponding to S_9 at the

points when $(S) = (u_c) - (v_c)$

or $(S) = (Qu)_c - (v_c)$

or $(S) = (uv)_c - (Q_c)$

K is the number of normalizing shifts (the number of shifts necessary to put the MSB of the arithmetic result of (u_m) and (v_m) in A_{61} -- maximum of 35 for addition and subtraction, 2 for multiplication and division).

Floating point add and subtract (FAuv and FSuv)

MP 0 (Unpack (u) sequence)	$7 + 2(u_{35})$			
1 (Unpack (v) sequence)	$7 + 2(v_{35})$			
2 (Initial alignment)	1	$+ 3(N_9)$		
3 (Final alignment)	4	$+ 2(N_9)$	$+ 2\|N\|$	
5 (NRP sequence)	12			$+ (34 - \|N\|) + 2K$
6 &7 (Set up NI)	7			

Total (in clock pulses) $38+2(u_{35})+2(v_{35})+5(N_9)+2|N|+(34-|N|)+2K$

Total (in μsec.) $76+4(u_{35})+4(v_{35})+10(N_9)+4|N|+(68-2|N|)+4K$

Case I N=0 Max.	Case I N=0 Min.	Case II N=1 Max.	Case II N=1 Min.
76	76	76	76
4	0	4	0
4	0	4	0
10	0	10	0
0	0	4	4
68	68	66	66
136	0	136	0
298 μ sec.	144 μ sec.	300 μ sec.	146 μ sec.

Case III N=2 Max.	Case III N=2 Min.	Case IV N=34 Max.	Case IV N=34 Min.
76	76	76	76
4	0	4	0
4	0	4	0
10	0	10	0
8	8	136	136
64	64	0	0
0	0	0	0
166 μ sec.	148 μ sec.	230 μ sec.	212 μ sec.

$N \leq 1$, Maximum time: 300 μ sec.
Minimum time: 144 μ sec.

$N \geq 2$, Maximum time: 236 μ sec.
Minimum time: 148 μ sec.

Note: In cases I and II, maximum addition times are 4 μ sec. less than shown. (If (u) and (v) are both positive, K = 0.)

Floating point multiply (FMuv)

MP 0 (Unpack (u) sequence)	$7 + (u_{35})$
1 (Unpack (v) sequence)	7
*2 (Multiply sequence)	$40 + \sum_{i=1}^{26} (u_{35} \oplus u_i) + 2(u_{35} \oplus u_0)$
5 (NRP sequence)	$20 + 2k$
6 & 7 (Set up NI)	7
Total (clock pulses)	$81 + (u_{35}) + 4\sum_{i=1}^{26} (u_{35} \oplus u_i) + 2(u_{35} \oplus u_0) + 2k$
Total (μsec.)	$162 + 2(u_{35}) + 8\sum_{i=1}^{26} (u_{35} \oplus u_i) + 4(u_{35} \oplus u_0) + 4k$

Maximum time: 380 μ sec.

Minimum time: 162 μ sec.

*If (u) is negative, the complement of (u) is sent to Q. Therefore, Q_{35-27} is always zero, and (Q_{26-0}) may be the complement of (u_{26-0}).

Floating point divide (FDuv)

MP 0 (Unpack (u) sequence)	$7 + (u_{35})$
1 (Unpack (v) sequence)	7
2 (Initial shift of (u)	36
3 (Divide sequence)	222
4 (Q $\rightarrow$ A sequence)	6
5 (NRP sequence)	$39 + 2k$
6 & 7 (Set up NI)	7
Total (clock pulses)	$324 + (u_{35}) + 2k$
Total (μsec.)	$648 + 2(u_{35}) + 4k$

Maximum time: 654 μ sec.

Minimum time: 648 μ sec.

Floating point polynomial multiply (FPuv)

MP	0	(Unpack (Q) sequence)	$5 + (Q_{35})$
	1	(Unpack (u) sequence)	7
	2	(Multiply sequence)	$40 + 4\sum_{i=1}^{26} (Q_{35} \oplus u_i) + 2(Q_{35} \oplus u_0)$
	(3)	(NRP (Q) · (u) sequence)	14
	(4)	Unpack (v) sequence	$7 + (v_{35})$
	3^1	(Initial alignment)	$1 + 3(N_9)$
	4^1	(Final alignment)	$4 + 2(N_9) + 2\|N\|$
	5	(NPP sequence)	$12 + (34 - \|N\|) + 2K$
	6&7	(Set up NI)	7

Total (clock pulses): $97+(Q_{35})+4\sum_{i=1}^{26}(Q_{35}\oplus u_i)+2(Q_{35}\oplus u_0)+5(N_9)+2|N|+(34-|N|)+2K$

Total (μ sec.): $194+2(Q_{35})+8\sum_{i=1}^{26}(Q_{35}\oplus u_i)+4(Q_{35}\oplus u_0)+5(N_9)+4|N|+(68-2|N|)+4K$

Maximum and minimum times depend upon the value of N as well as (Q). (See the four cases given under Floating add and subtract.) Taking the largest maximum (N=1) and the smallest minimum (N = 0), the maximum and minimum time are:

Maximum time (N = 1): 619 μ sec.

Minimum time (N = 0): 262 μ sec.

Floating point inner product (FIuv)

MP	0	($Q \rightarrow F_4$ sequence)	7
	1	(Unpack (u) sequence)	$7 + (u_{35})$
	(2)	(Unpack (v) sequence)	7
	(3)	(Multiply sequence)	$40 + 4\sum_{i=1}^{26} (u_{35} \oplus u_i) + 2(u_{35} \oplus u_0)$
	(4)	(NRP (u) · (v) sequence)	14
	2^1	(Unpack (F_4) sequence)	7
	3^1	(Initial alignment)	$1 + 3(N_9)$
	4^1	(Final alignment)	$4 + 2(N_9) + 2\lvert N\rvert$
	5	(NRP sequence)	$12 + (34 - \lvert N\rvert) + 2K$
	6&7		7

Total (clock pulses): $106+(u_{35})+4\sum_{i=1}^{26}(u_{35}\oplus u_i)+2(u_{35}\oplus u_0)+5(N_9)+2\lvert N\rvert+(34-\lvert N\rvert)+2K$

Total (μ sec.): $212+2(u_{35})+8\sum_{i=1}^{26}(u_{35}\oplus u_i)+4(u_{35}\oplus u_0)+10(N_9)+4\lvert N\rvert+(68-2\lvert N\rvert)+4K$

Maximum and minimum times depend upon the value of N as well as (u).

(See the four cases given under Floating add and subtract.)

Taking the largest maximum (N = 1) and the smallest minimum (N = 0), maximum and minimum times are:

Maximum (N = 1): 637 μ sec.

Minimum (N = 0): 280 μ sec.

Floating point unpack (UPuv)

MP	0	(Unpack (u) sequence)	7 + (u_{35})
	1	($u_m \rightarrow$ m sequence)	5
	2	($u_c \rightarrow$ s sequence)	1
	5	($u_c \rightarrow$ v sequence)	5
	6&7	(Set up NI)	8
Total (clock pulse)			26 + (u_{35})
Total (μ sec.)			52 + $2(u_{35})$

Floating point normalize pack (NPuv)

MP	0	(Read (v) sequence)	7
	1	($v_c \rightarrow$ c sequence)	1
	2	(Read (u) sequence)	7
	3	($u \rightarrow$ A sequence)	3
	4	(NRP sequence)	39 + 2K
	5	(Write in (u) sequence)	7
	6&7		8
Total (clock pulses)			72 + 2K
Total (μ sec.)			144 + 4K $(9 > K \geq 0)$

Maximum time: 180 μ sec.

Minimum time: 144 μ sec.

Floating point normalize exit (NE_j-)

MP	0	(Clear x)	1
	1	(Set NE FF	1
	5	------	1
	6&7		7
Total (clock pulses)			10
Total (μ sec.)			20

USEful Note Number 4 November 10, 1956

SUBJECT: Revision of the Card Package Routine (RR)

By the following modifications the Card Package Routine (CV37), which was written for the 1103, has been made useable on the 1103A.

A and Q Addresses

Accumulator and Q-register addresses have been modified to the standard address of each for the 1103A as follows:

A - From 20000 to 32000
Q - From 10000 to 31000

IOB Master Selection Bit

The IOB Master Selection Bit for the Card Unit has been changed from IOB6 for the 1103 to IOB35 for the 1103A. This change has been accomplished by modification of the instruction in which the Master Bit is introduced. This modification for the Card Read Routine and the Card Punch Routine now reads:

Read Routine -	72163	00161	01013	31	01211	00036
Punch Routine-	72361	00357	01007	31	01163	00036

Check Sum

A revision of the check sum for the transfer of the Card Package Routine to the ~~card~~ CORE has been made because of the above mentioned revisions throughout the routine. Thus the contents of the drum address 72313 now reads:

72313 00311 01143 23 72111 72133

CV-37

CARD PACKAGE ` IC 001

The Card Package Routine punches out on bioctal tape either of the card routines described below. The card routine punched out is modified according to a specified ES operating address (address of the first word of the card routine as it is stored for use in ES). In addition, the bioctal tape is punched with a specified insert address (storage address) which may or may not be the same as the ES operating address.

The Card Package sums itself after transfer to ES. If an improper sum is obtained, "SUM" is typed out by the supervisory typewriter and the 1103 stops. The routine tests the control word-- if it is not suitable, a new control word is asked for by typing out "set q".

OPERATING INSTRUCTIONS

1) Put 1103 in test mode.
2) Set PAK = 72000 and start.
3) a. If "SUM" types out, reread tape and rerun.
   b. If "set q" types out, set up control word in (Q).
4) Control word·

   (Q) = XO MMMMM mmmmm

   X = 1 for card read routine.
   X = 2 for card punch routine.
   M = desired insert address.
   00000 ≤ M ≤ 01777
   40000 ≤ M ≤ 77777
   m = desired ES operating address.
   00000 ≤ m ≤ 01541
5) Turn on paper tape punch and start.

If X ≠ 1 or 2, or if M or m is not in the proper range, the routine asks for a corrected control word.

Either card routine requires 235 octal words of ES in which to operate, constants and temporary storage included.

The Card Package does not use the constant pool. ES is used and not restored.

Drum address: 72000 - 72531

CARD READ ROUTINE

This subroutine causes the Bull Reproducer to go through a read cycle. The decimal information from the Bull Reproducer is converted to binary and scaled according to a given scaling factor. The results are then stored in specified ES memory locations.

CARD PUNCH ROUTINE

This suboutine converts specified binary numbers into decimal and sends equivalent coded-decimal information to the Bull Reproducer and causes it to go through a punch cycle.

These card routines require the following information:

1) Binary scaling.
2) Decimal scaling.
3) Locations of fields on the card.
4) Zero suppression (punch only).

This information is supplied to the card routine in a standard form called a parameter word. One parameter word is required for each card field.

A field consists of a number of consecutive card columns. The last column of a field is reserved for the sign of the decimal number stored in that field. An 11-punch signifies a negative number, no punch (blank column) signifies a positive number. A combination 12, 3 and 8 punch in one column represents a decimal point.

Fields need not be adjacent-- there may be unused columns, punched or unpunched, between them-- nor need they be alike in size.

Either card routine is entered from line y as follows:

y)	37 mmmmm mmmmm	(to card routine)
y+1)	AB PPPPP DDDDD	(control word)
y+2)	Next Instruction	

m represents the beginning address (ES operating address) of the card routine to be used.

The control word is described below.

The 37 command records in m the address of the control word. The routine is then entered at m. After finishing its operation, the card routine exits to y+2, the line following the control word.

CONTROL WORD

The control word controls the operation of the card routines. Its composition is as follows:

AB PPPPP DDDDD

A, the first octal digit, controls positioning of cards in the read and punch channels of the Bull Reproducer.

A = 1	Pick a read card from the read hopper.
A = 2	Pick a punch card from the punch hopper.

B, the second octal digit, controls the operation to be performed.

B = 1	Read a card.
B = 2	Punch a card.

P is the address of the first parameter word.

D is the address of the first data word.

P and D both must be ES addresses.

The table on page 5 lists the combinations of operations that may be performed by the card routines.

PARAMETER WORD

A parameter word consists of twelve octal digits divided into six groups of two each:

FF SS BB LL RR ZZ

FF: Flag for final parameter word.
FF = 77 octal for final word.
FF = 00 otherwise.

SS: Binary scaling factor. (number of bits to the right of the binary point)

BB: Number of blank or unused columns to the left of the field.

LL: Number of digit positions to the left of the decimal point.

RR: Number of remaining columns in the field, exclusive of sign. (number of decimal digits to the right of the decimal point plus one for the decimal point)
RR = 00 indicates no decimal point and no decimal fraction.

ZZ: Flag for zero suppression.
ZZ = 77 octal for zero suppression.
ZZ = 00 for no zero suppression. These two digits are decoded by the punch routine only. Only zeros in the integer part are suppressed. A zero immediately preceeding the decimal point is not suppressed.

Range of the parameters:

decimal	octal
00 ≤ SS ≤ 35	00 ≤ SS ≤ 43
00 ≤ BB ≤ 63	00 ≤ BB ≤ 77
00 ≤ LL ≤ 10	00 ≤ LL ≤ 12
00 ≤ RR ≤ 11	00 ≤ RR ≤ 13
01 ≤ LL+RR ≤ 11	01 ≤ LL+RR ≤ 13

The parameter words, one for each field, must be stored consecutively starting at some ES memory location P. There must be an equal number of consecutive words starting with some ES memory location D, reserved for storing the results of the read routine, or filled with data for the punch routine.

Punching takes place at the third card station in the punch channel, therefore two punch cards must be advanced before punching can take place. This can be done manually, or either card routine can be used to position the cards as follows:

```
37 mmmmm mmmmm    ( to card routine )
20 00000 00000    ( pick punch card )
37 mmmmm mmmmm    ( to card routine )
20 00000 00000    ( pick punch card )
```

Reading takes place at the second card station in the read channel-- one read card must be advanced before reading takes place. This also may be done manually, or may be done by either card routine:

```
37 mmmmm mmmmm    ( to card routine )
10 00000 00000    ( pick  read card )
```

The card just advanced will not feed further unless another order to pick a card is given-- both pick and read orders must be given to read this card.

It should be noted that once a card enters either the read or punch channel it continues to advance one card station each time the Bull Reproducer is cycled.

Numbers are rounded to the desired number of decimal digits before punching takes place. A divide check error stop results if an insufficient number of card columns is allowed for the integer portion of a field.

In case of a card machine failure or an accidental stop in the middle of a card cycle, the current card may be reread or punched again: reposition the cards, set PAK = 00000, and start.

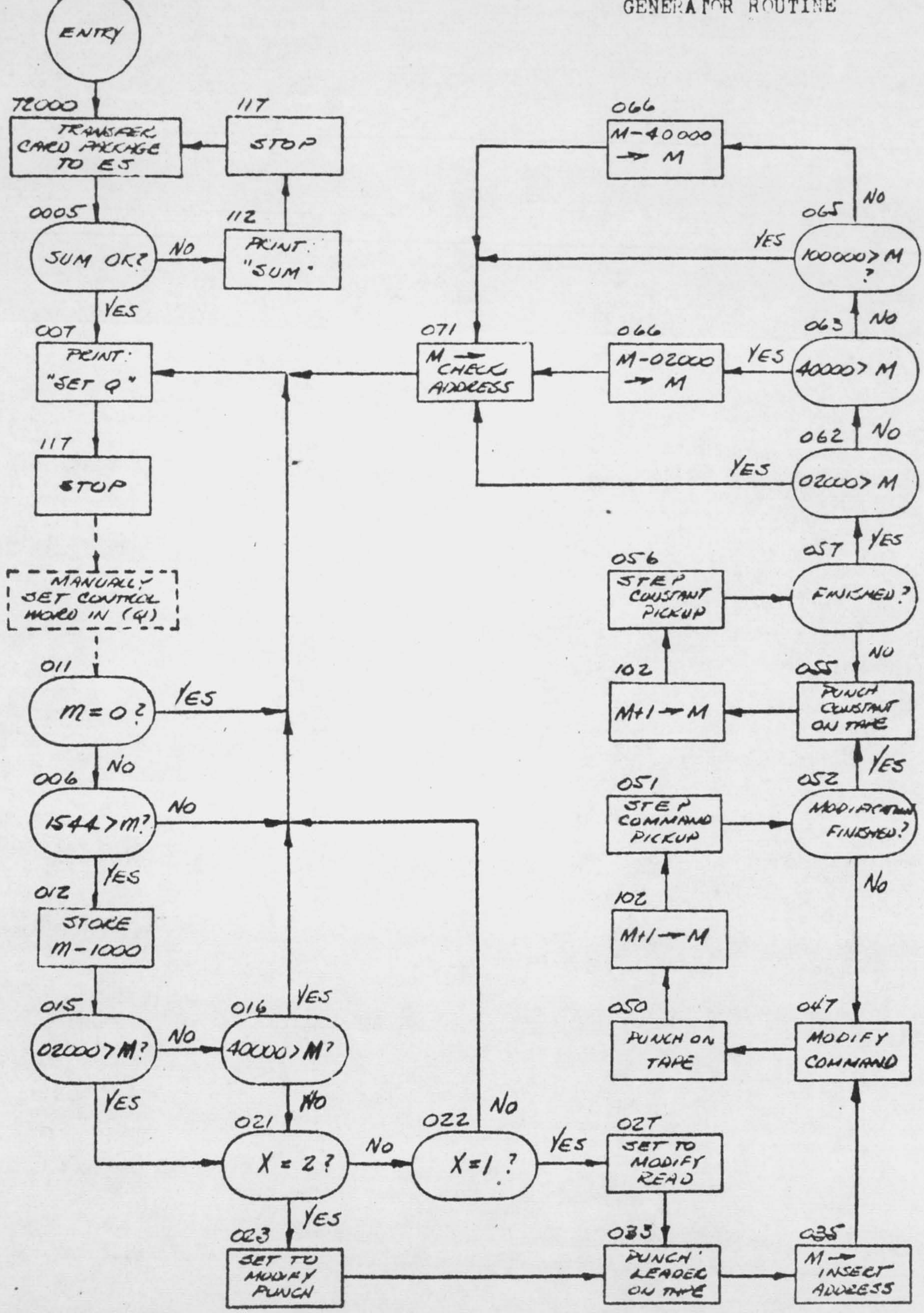
CARD ROUTINE OPERATIONS

A B	OPERATION
0 0	Cycle Bull Reproducer. Cards already in either channel are advanced one card station. This operation also is performed with all of the following operations.
0 2	Punch a card.
0 3	Punch a card (when used with the punch routine-- do not use with the read routine).
1 0	Pick a card from the read hopper. This card is not in the read channel until the next card is picked from the read hopper.
1 1	Pick a card from the read hopper and read a card.
1 2	Pick a card from the read hopper and punch a card.
1 3	Pick a card from the read hopper and either read or punch a card. See note below.
2 0	Pick a card from the punch hopper. This card is not in the punch channel until the next card is picked from the punch hopper.
2 2	Pick a card from the punch hopper and punch a card.
2 3	Pick a card from the punch hopper and punch a card (when used with the punch routine-- do not use with the read routine).
3 0	Pick a card from both hoppers.
3 1	Pick a card from both hoppers and read a card.
3 2	Pick a card from both hoppers and punch a card.
3 3	Pick a card from both hoppers and either read or punch a card. See note below.

Note: both routines can pick either or both read and punch cards. However, the read routine cannot be used to punch a card and the punch routine cannot be used to read a card. If both operatio

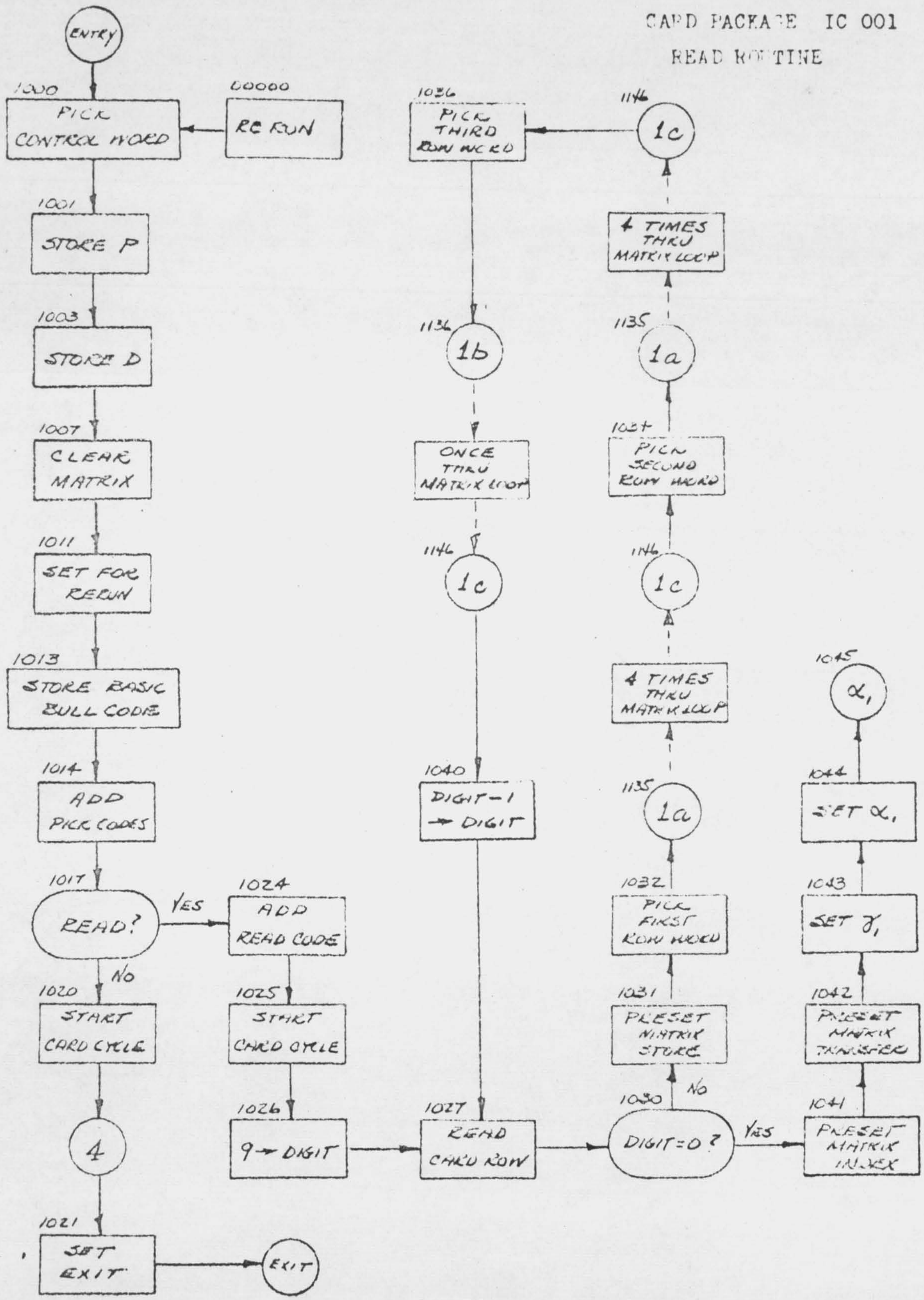
CARD PACKAGE IC001

GENERATOR ROUTINE

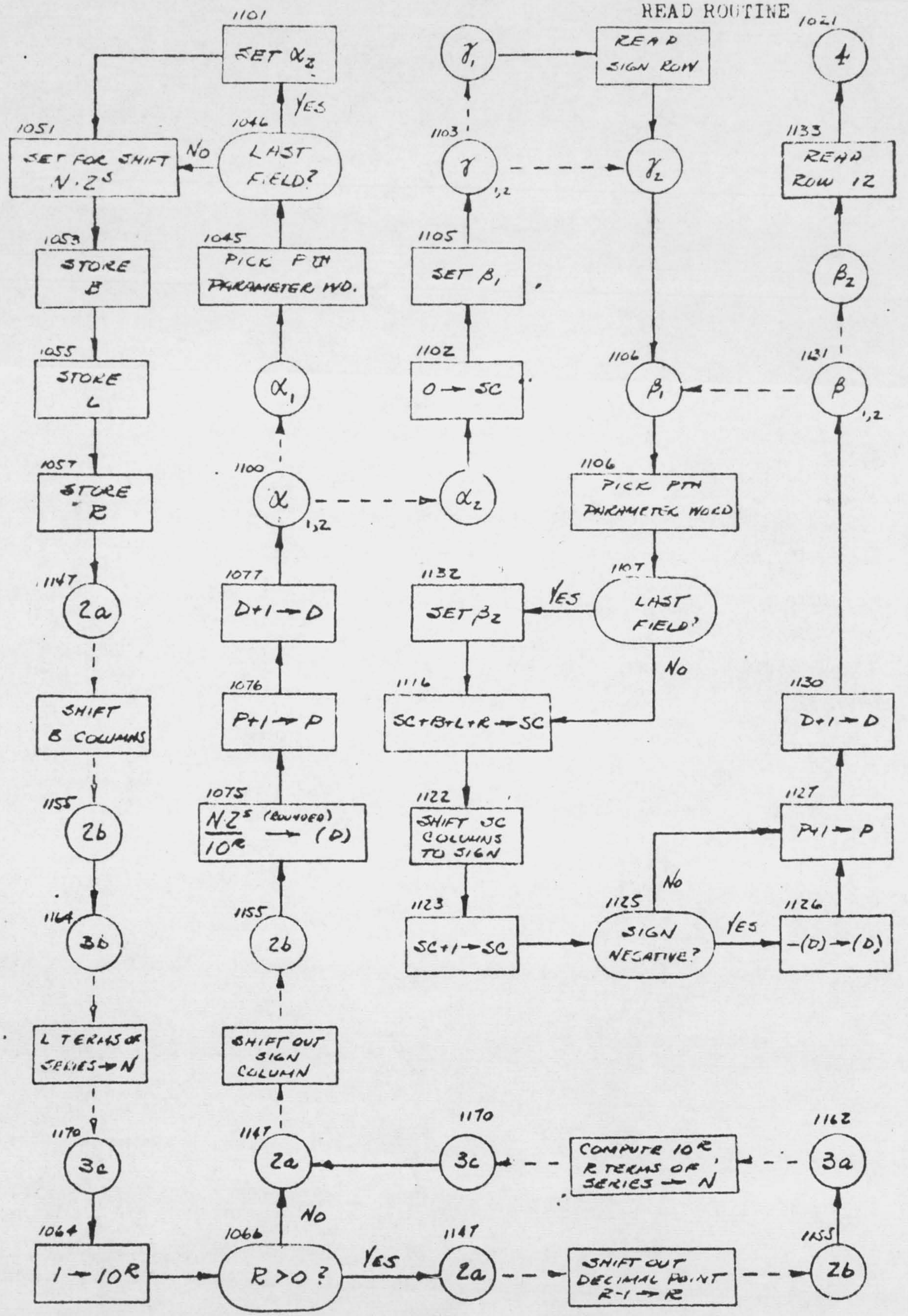


CARD PACKAGE IC 001

READ ROUTINE



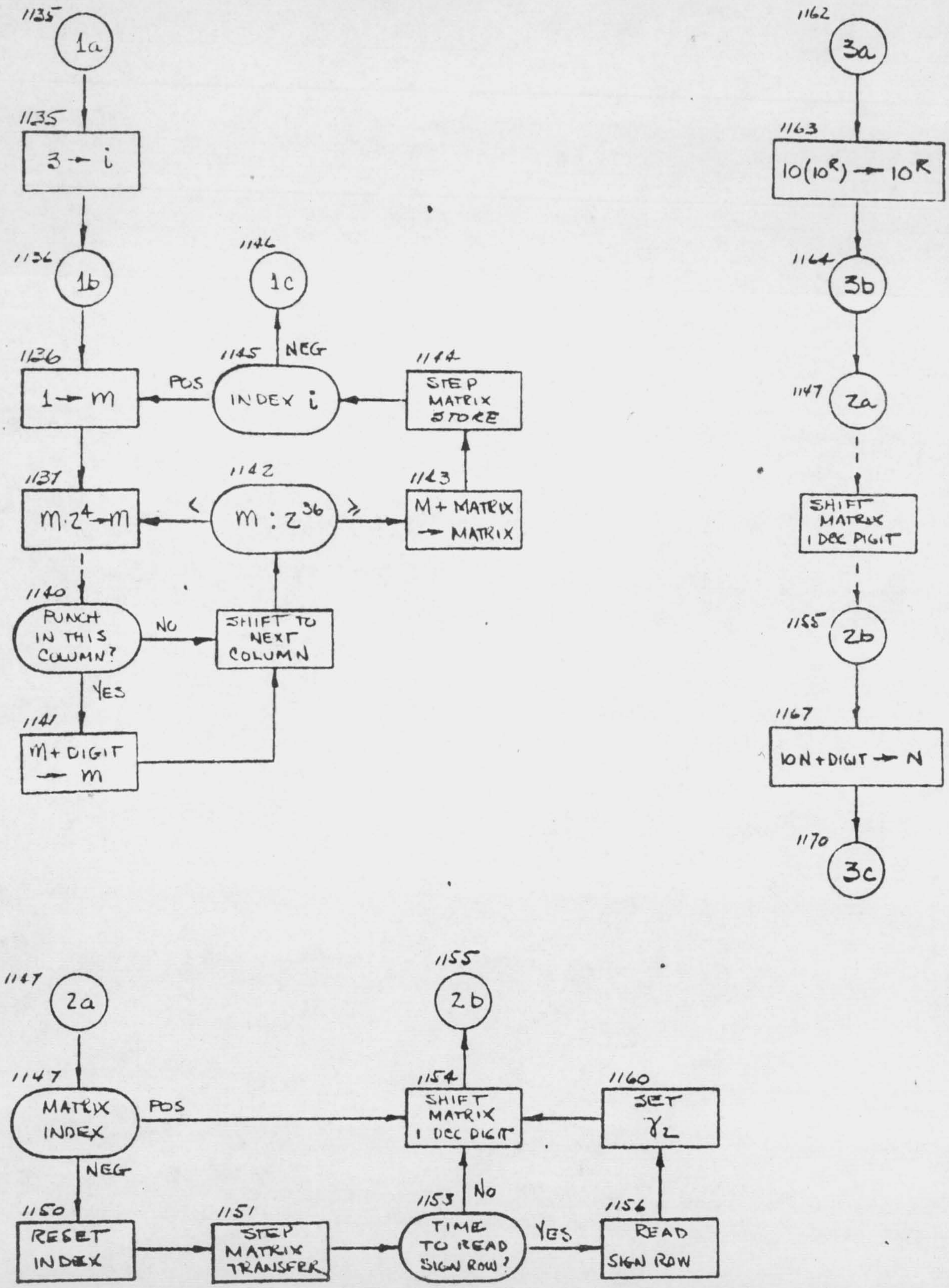
CARD PACKAGE IC 001
READ ROUTINE



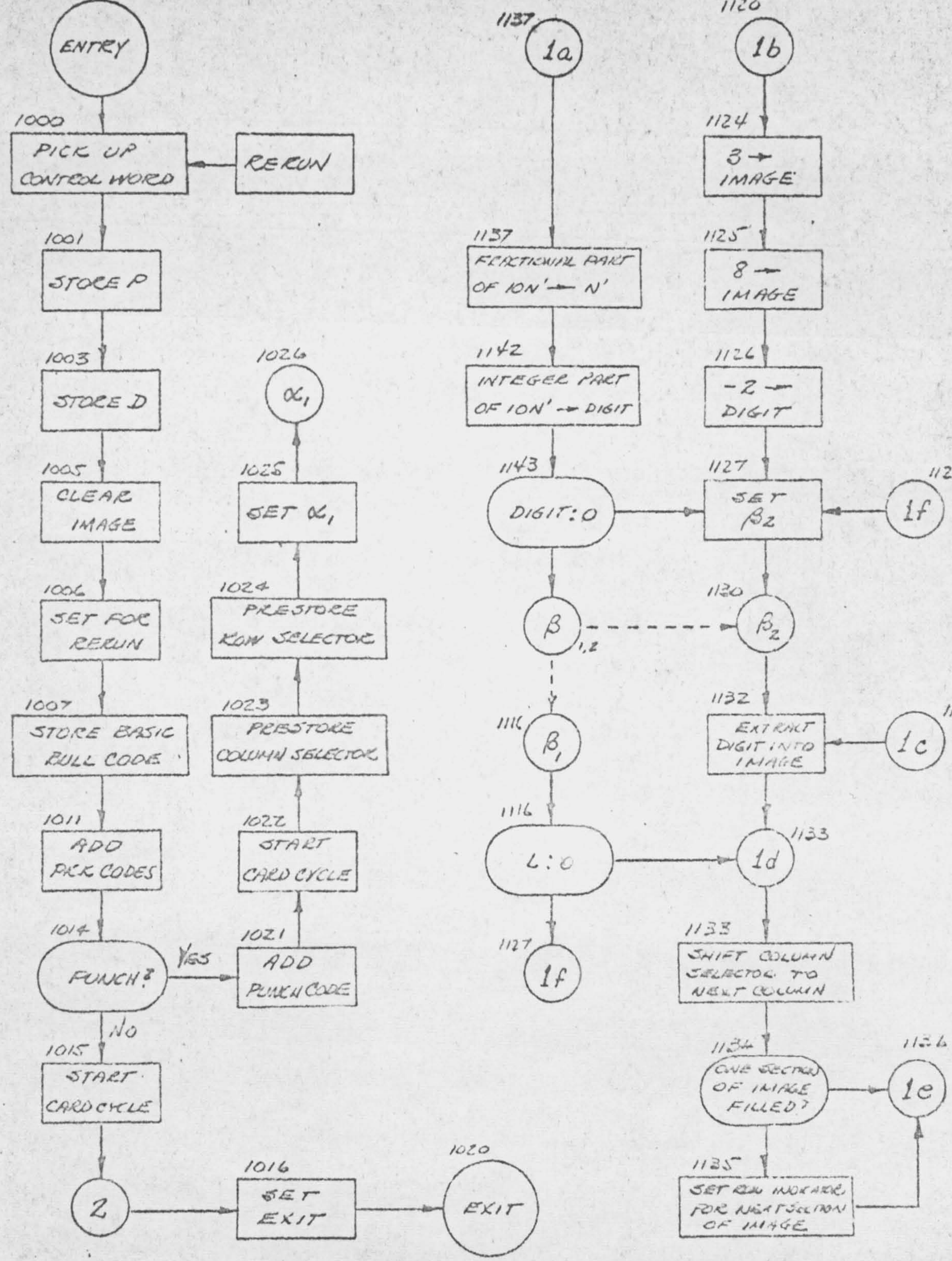
PX 71900-4-37

CARD PACKAGE IC 001

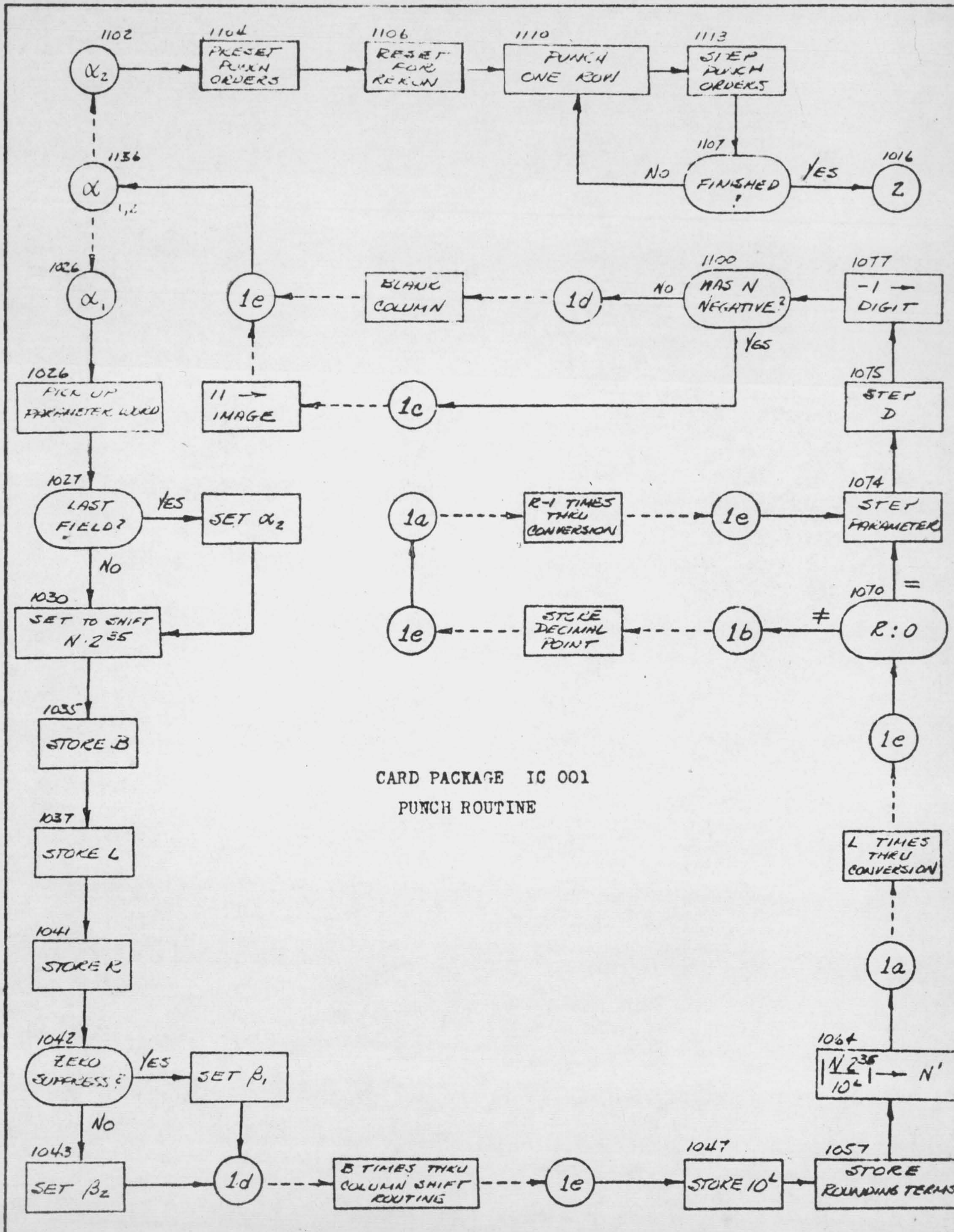
READ ROUTINE



CARD PACKAGE IC 001
PUNCH ROUTINE



X 71900-4-37



CARD PACKAGE IC 001

PUNCH ROUTINE

PX 71900-4-37

CV-37

CARD PACKAGE IC 001

GENERATOR ROUTINE

72000		75 30530 00001	TRANSFER CARD PACKAGE
72001		11 72002 00000	TO ES.
72002	00000	45 00350 00001	
72003	00001	23 10000 20000	0 → (A), (Q)
72004	00002	75 20530 00004	MEMORY SUM → (A)
72005	00003	32 00000 00000	
72006	00004	11 20000 20000	(R) → (A)
72007	00005	47 00112 00007	SUM OK ?
72010	00006	42 00121 00012	1544 > m ?
72011	00007	37 00117 00111	PRINT: "SET Q"
72012	00010	51 00122 20000	m → (A)
72013	00011	47 00006 00111	m = 0 ?
72014	00012	36 00346 01000	STORE m -1000
72015	00013	51 00123 20000	M → (A)
72016	00014	54 20000 00071	SR15 (A)
72017	00015	42 00127 00017	02000 > M ?
72020	00016	42 00130 00111	40000 > M ?
72021	00017	11 20000 01001	STORE M
72022	00020	44 00021 00021	SL 1 (Q)
72023	00021	44 00023 00022	PUNCH ?
72024	00022	44 00027 00111	READ ?
72025	00023	15 00000 00043	SET
72026	00024	15 00026 00054	FOR
72027	00025	15 00035 00120	PUNCH
72030	00026	45 00135 00032	
72031	00027	15 00072 00043	SET
72032	00030	15 00100 00054	FOR
72033	00031	15 00000 00120	READ
72034	00032	11 00272 01002	SET INDEX
72035	00033	75 20200 00041	PUNCH LEADER AND 7TH
72036	00034	63 47601 00001	LEVEL HOLE ON BIOCTAL TAPE
72037	00035	63 00530 10000	PUNCH
72040	00036	55 01001 00006	INSERT
72041	00037	41 01003 00035	ADDRESS
72042	00040	63 10000 10000	ON
72043	00041	11 00272 01003	BIOCTAL
72044	00042	41 01002 00036	TAPE

GENERATOR ROUTINE

```
72045  00043  11 30000 10000   COMMAND TO BE MODIFIED → (Q)
72046  00044  51 00124 01002   EXTRACT MULTIPLIER
72047  00045  54 01002 00077   SR9
72050  00046  11 10000 20000   COMMAND → (A)
72051  00047  72 01002 01000   ADD MODIFICATION
72052  00050  37 00103 00104   PUNCH MODIFIED COMMAND ON TAPE
72053  00051  21 00043 00143   STEP COMMAND PICKUP
72054  00052  42 00120 00043   MODIFICATION FINISHED ?
72055  00053  11 00141 01003   SET INDEX
72056  00054  31 30000 00000   PICK UP CONSTANT
72057  00055  37 00103 00104   PUNCH CONSTANT ON TAPE
72060  00056  21 00054 00143   STEP CONSTANT PICKUP
72061  00057  41 01003 00054   FINISHED ?
72062  00060  31 01001 00000   CHECK ADDRESS → (A)
72063  00061  11 00127 01002   02000 → S
72064  00062  42 00127 00067   02000 > CHECK ADDRESS ?
72065  00063  42 00130 00066   40000 > CHECK ADDRESS ?
72066  00064  11 00130 01002   40000 → S
72067  00065  42 00143 00067   100000 > CHECK ADDRESS ?
72070  00066  36 01002 01001   CHECK ADDRESS -S → CHECK ADDRESS
72071  00067  11 00272 01002
72072  00070  11 00135 01003
72073  00071  55 01001 00006   PUNCH
72074  00072  63 00146 10000     CHECK
72075  00073  41 01002 00071       ADDRESS
72076  00074  55 01001 00006         ON
72077  00075  63 10000 10000           BIOCTAL
72100  00076  41 01003 00071             TAPE
72101  00077  75 00200 00007   PUNCH TRAILER, RETURN
72102  00100  63 00131 00001   TO BEGINNING.
72103  00101  63 10000 10000           TAPE
72104  00102  21 01001 00135
72105  00103  45 00000 30000
72106  00104  55 20000 00000             PUNCH
72107  00105  55 10000 00006
72110  00106  43 10000 00101
72111  00107  63 00000 10000
72112  00110  45 00000 00105               ROUTINE.
```

X 71900-4-37

GENERATOR ROUTINE

72113	00111	11	00126	00125		
72114	00112	54	00125	00044		
72115	00113	43	20000	00117	PRINT	
72116	00114	54	20000	00006	ROUTINE	
72117	00115	61	00000	20000		
72120	00116	45	00000	00113		
72121	00117	56	00000	72000		
72122	00120	11	30000	10000	COMPARAND	
72123	00121	00	00000	01544	ES LIMIT	
72124	00122	00	00000	77777	EXTRACTOR	
72125	00123	00	77777	00000	EXTRACTOR	
72126	00124	00	01000	01000	EXTRACTOR	
72127	00125	45	47243	40757	" SUM "	
72130	00126	45	24200	10435	" SET Q "	
72131	00127	00	00000	02000	ES LIMIT	
72132	00130	00	00000	40000	DRUM LIMIT	
72133	00131	00	00000	00011	THESE	
72134	00132	00	00001	00001	FIRST	
72135	00133	00	00000	00017	ELEVEN	
72136	00134	00	00000	00044	CONSTANTS	
72137	00135	00	00000	00001	ARE	THESE
72140	00136	00	00000	00003	USED	NINE
72141	00137	00	00000	00005	BY	CONSTANTS
72142	00140	00	00000	00010	THE	ARE
72143	00141	00	00000	00012	CARD	USED
72144	00142	00	00000	00077	READ	BY
72145	00143	00	00001	00000	ROUTINE	THE
72146	00144	00	00000	00014		PUNCH
72147	00145	40	00000	00000		ROUTINE

CARD PACKAGE IC 001

READ ROUTINE

72150	00146	01000	71	01206	30000	CONTROL WORD ⟶ (A)
72151	00147	01001	15	20000	01045	PRESET PARAMETER
72152	00150	01002	15	20000	01106	PICKUP COMMANDS
72153	00151	01003	16	20000	01075	PRESET DATA
72154	00152	01004	16	20000	01126	STORAGE COMMANDS
72155	00153	01005	55	20000	00017	PRESET DATA
72156	00154	01006	15	10000	01126	PICKUP COMMAND
72157	00155	01007	75	20011	01011	CLEAR
72160	00156	01010	23	01223	20000	MATRIX
72161	00157	01011	16	01200	00000	SET (00000) FOR RERUN
72162	00160	01012	55	10000	00030	
72163	00161	01013	31	01211	00001	BASIC BULL CODE ⟶ (A)
72164	00162	01014	52	01207	20000	ADD PICK CODES, IF PRESENT
72165	00163	01015	54	20000	00002	SL z (A)
72166	00164	01016	55	10000	00002	SL z (Q)
72167	00165	01017	44	01024	01020	READ ?
72170	00166	01020	17	00000	20000	START CARD CYCLE
72171	00167	01021	21	01000	01206	SET
72172	00170	01022	16	20000	01023	EXIT
72173	00171	01023	45	00000	30000	EXIT OF READ ROUTINE
72174	00172	01024	35	01206	20000	ADD READ CODE
72175	00173	01025	17	00000	20000	START CARD CYCLE
72176	00174	01026	11	01202	01124	9 ⟶ DIGIT
72177	00175	01027	37	01174	01171	READ CARD ROW
72200	00176	01030	47	01031	01041	BEGIN CONVERSION WHEN DIGIT =
72201	00177	01031	11	01176	01143	PRESET MATRIX STORE
72202	00200	01032	55	01220	00000	ROW WORD 1 ⟶ (Q)
72203	00201	01033	37	01146	01135	4 TIMES THRU MATRIX LOOP
72204	00202	01034	55	01221	00000	ROW WORD 2 ⟶ (Q)
72205	00203	01035	37	01146	01135	4 TIMES THRU MATRIX LOOP
72206	00204	01036	55	01222	00034	ROW WORD 3 ⟶ (Q)
72207	00205	01037	37	01146	01136	ONCE THRU MATRIX LOOP
72210	00206	01040	41	01124	01027	DIGIT -1 ⟶ DIGIT, REPEAT
72211	00207	01041	11	01202	01143	PRESET MATRIX INDEX
72212	00210	01042	15	01010	01152	PRESET MATRIX TRANSFER
72213	00211	01043	16	01027	01103	SET SIGN READ SWITCH
72214	00212	01044	37	01100	01045	SET α_1

X 71900-4-37

READ ROUTINE

72215	00213	01045	55	30000	00000	PARAMETER WORD → (Q)
72216	00214	01046	44	01101	01047	LAST FIELD?
72217	00215	01047	55	10000	00013	SL_{11} (Q)
72220	00216	01050	51	01213	20000	S → A
72221	00217	01051	16	20000	01073	SET SHIFT
72222	00220	01052	55	10000	00006	SL_6 (Q)
72223	00221	01053	51	01213	01216	STORE B
72224	00222	01054	55	10000	00006	SL_6 (Q)
72225	00223	01055	51	01213	01217	STORE L
72226	00224	01056	55	10000	00006	SL_6 (Q)
72227	00225	01057	51	01213	01215	STORE R
72230	00226	01060	37	01155	01061	SHIFT MATRIX
72231	00227	01061	41	01216	01147	B DECIMAL DIGITS
72232	00230	01062	37	01170	01063	L TERMS OF SERIES
72233	00231	01063	41	01217	01164	CONVERSION → N
72234	00232	01064	11	01206	01217	1 → 10^R
72235	00233	01065	16	01201	01155	IF R > 0, SHIFT OUT
72236	00234	01066	41	01215	01147	DECIMAL POINT, R-1 → R
72237	00235	01067	37	01170	01070	COMPUTE 10^R, R TERMS OF
72240	00236	01070	41	01215	01162	SERIES ADDED TO N
72241	00237	01071	37	01155	01147	SHIFT OUT SIGN POSITION
72242	00240	01072	54	01217	10107	$10^R \cdot 2^{-1}$ → (Q)
72243	00241	01073	31	01216	30000	$N \cdot 2^S$ → (A)
72244	00242	01074	32	10000	00000	$N \cdot 2^S + 10^R \cdot 2^{-1}$ → (A)
72245	00243	01075	73	01217	30000	$N \cdot 2^S/10^R$ ROUNDED → (D)
72246	00244	01076	21	01045	01214	STEP PARAMETER
72247	00245	01077	21	01075	01206	STEP D
72250	00246	01100	45	00000	30000	α SWITCH
72251	00247	01101	37	01100	01047	SET α_2
72252	00250	01102	23	01216	20000	SET SHIFT COUNT = 0
72253	00251	01103	37	01174	30000	SIGN READ SWITCH
72254	00252	01104	55	01222	00034	SL 28 SIGN WORD 3
72255	00253	01105	37	01131	01106	SET β_1
72256	00254	01106	55	30000	00000	PARAMETER WORD → (Q)
72257	00255	01107	44	01132	01110	LAST FIELD?
72260	00256	01110	55	10000	00021	SL_{17} (Q)
72261	00257	01111	31	01216	00000	SHIFT COUNT → (A)
72262	00260	01112	52	01213	01216	SC + B → A

CV-37

READ ROUTINE

(Page Rev. 2/22/55)

72263	00261	01113	55	10000	00006	SL_6 (Q)
72264	00262	01114	52	01213	01216	SC + B + L → (A)
72265	00263	01115	55	10000	00006	SL_6 (Q)
72266	00264	01116	52	01213	01216	SC + B + L + R → SC
72267	00265	01117	73	01205	10000	SC/36 → (Q) REM → (R)
72270	00266	01120	55	10000	00017	SL_{15} (Q)
72271	00267	01121	35	10000	20000	$(Q) \cdot 2^{15}$ + REM → (A)
72272	00270	01122	35	01175	01124	SET SHIFT COMMAND
72273	00271	01123	21	01216	01206	SC + 1 → SHIFT COUNT
72274	00272	01124	[00	00000	00002]	SHIFT SIGN BIT → A_{71}
72275	00273	01125	46	01126	01127	A_{71} = 1 ?
72276	00274	01126	13	30000	30000	−(D) → (D)
72277	00275	01127	21	01106	01214	STEP PARAMETER
72300	00276	01130	21	01126	01203	STEP D
72301	00277	01131	45	00000	30000	β SWITCH
72302	00300	01132	37	01131	01110	SET β_Z
72303	00301	01133	37	01174	01171	READ ROW 12
72304	00302	01134	45	00000	01021	EXIT
72305	00303	01135	11	01207	01216	SET INDEX = 3
72306	00304	01136	11	01206	01217	1 → M
72307	00305	01137	54	01217	00004	SL_4 M
72310	00306	01140	44	01141	01142	Q35 = 1
72311	00307	01141	35	01124	01217	DIGIT + M → M
72312	00310	01142	43	20000	01137	FULL MATRIX WORD IN (R) ?
72313	00311	01143	[37	23120	75225]	MATRIX + M → MATRIX
72314	00312	01144	21	01143	01214	STEP STORE COMMAND
72315	00313	01145	41	01216	01136	FINISHED MATRIX LOOP ?
72316	00314	01146	45	00000	30000	EXIT MATRIX LOOP
72317	00315	01147	41	01143	01154	USED UP MATRIX INDEX ?
72320	00316	01150	11	01211	01143	RESET INDEX
72321	00317	01151	21	01152	01214	STEP MATRIX TRANSFER
72322	00320	01152	11	30000	01223	TRANSFER MATRIX WORD
72323	00321	01153	43	01177	01156	TIME TO READ SIGN ROW ?
72324	00322	01154	55	01223	00004	SL_4 MATRIX WORD
72325	00323	01155	45	00000	30000	EXIT OF SHIFT ROUTINE
72326	00324	01156	37	01174	01171	READ SIGN ROW
72327	00325	01157	55	01222	00034	SL_{28} SIGN WORD 3
72330	00326	01160	37	01103	01154	SET SIGN READ SWITCH TO SKIP

FX 71900-4-37

CV-37

READ ROUTINE

72331	00327	01161	45	00000	01105	
72332	00330	01162	71	01212	01217	$10 \cdot 10^R$
72333	00331	01163	11	20000	01217	$\longrightarrow 10^R$
72334	00332	01164	37	01155	01147	SHIFT MATRIX 1 DECIMAL DIGIT
72335	00333	01165	31	01216	00002	4N ⟶ (A)
72336	00334	01166	32	01216	00001	2 (4N + N) ⟶ (A)
72337	00335	01167	52	01204	01216	10N + DIGIT ⟶ N
72340	00336	01170	45	00000	30000	EXIT OF SERIES ROUTINE
72341	00337	01171	76	00000	01222	READ ONE
72342	00340	01172	76	10000	01220	ROW FROM
72343	00341	01173	76	10000	01221	PUNCHED CARD
72344	00342	01174	45	00000	30000	ROW READ EXIT
72345	00343	01175	31	01220	00044	PRESET
72346	00344	01176	21	01223	01217	PRESET
72347	00345	01177	11	01230	01223	COMPARAND
72350	00346	01200	00	00000	01000	CONSTANT
72351	00347	01201	00	00000	01067	CONSTANT
		1202	00	00000	00011	9 DECIMAL
		1203	00	00001	00001	U AND V ADVANCE
		1204	00	00000	00017	4 BIT EXTRACTOR
		1205	00	00000	00044	36 DECIMAL
		1206	00	00000	00001	1
		1207	00	00000	00003	3
		1210	00	00000	00005	5
		1211	00	00000	00010	8
		1212	00	00000	00012	10
		1213	00	00000	00077	TWO OCTAL DIGIT EXTRACTOR
		1214	00	00001	00000	U ADVANCE
		1215				R
		1216				B. N. SC. INDEX
		1217				L. M. 10
		1220				ROW WORD 1
		1221				ROW WORD 2
		1222				ROW WORD 3
		1223				MATRIX WORD 1
		1224				MATRIX WORD 2
		1225				MATRIX WORD 3
		1226				MATRIX WORD 4

CV-3 7

READ ROUTINE

1227	MATRIX WORD 5
1230	MATRIX WORD 6
1231	MATRIX WORD 7
1232	MATRIX WORD 8
1233	MATRIX WORD 9

CV-37

CARD PACKAGE IC 001

PUNCH ROUTINE

72352	00350	01000	71	01160	30000	CONTROL WORD → (A)
72353	00351	01001	15	20000	01026	SET PARAMETER PICKUP
72354	00352	01002	55	20000	00017	SL$_{15}$ CONTROL WORD
72355	00353	01003	15	10000	01060	SET DATA PICKUP
72356	00354	01004	75	20044	01006	CLEAR
72357	00355	01005	23	01171	20000	IMAGE
72360	00356	01006	16	01156	00000	SET (00000) FOR RERUN
72361	00357	01007	31	01163	00001	BASIC BULL CODE → (A)
72362	00360	01010	55	10000	00030	SL$_{24}$ CONTROL WORD
72363	00361	01011	52	01161	20000	EXTRACT PICK CODES
72364	00362	01012	54	20000	00002	SL$_2$ (A)
72365	00363	01013	44	01014	01014	SL$_1$ CONTROL WORD
72366	00364	01014	44	01021	01015	PUNCH ?
72367	00365	01015	17	00000	20000	NO START CARD CYCLE.
72370	00366	01016	21	01000	01160	SET
72371	00367	01017	16	20000	01020	EXIT
72372	00370	01020	45	00000	30000	EXIT, SWITCH
72373	00371	01021	32	01137	00000	YES, ADD PUNCH CODE.
72374	00372	01022	17	00000	20000	START CARD CYCLE
72375	00373	01023	11	01170	01110	PRESTORE COLUMN SELECTOR
72376	00374	01024	11	01154	01111	PRESTORE ROW SELECTOR
72377	00375	01025	37	01020	01026	SET α_1
72400	00376	01026	55	30000	00000	PARAMETER WORD → (Q)
72401	00377	01027	44	01101	01030	LAST PARAMETER WORD ?
72402	00400	01030	55	10000	00013	SET
72403	00401	01031	51	01165	20000	UP
72404	00402	01032	13	20000	20000	SHIFT
72405	00403	01033	35	01155	01061	ORDER
72406	00404	01034	55	10000	00006	SL$_6$ (Q)
72407	00405	01035	51	01165	01132	STORE B
72410	00406	01036	55	10000	00006	SL$_6$ (Q)
72411	00407	01037	51	01165	01125	STORE L
72412	00410	01040	55	10000	00006	SL$_6$ (Q)
72413	00411	01041	51	01165	01153	STORE R
72414	00412	01042	44	01115	01043	ZERO SUPPRESSION ?
72415	00413	01043	16	01127	01143	SET FOR NO ZERO SUPPRESSION

FX 71900-4-37

CV-37

PUNCH ROUTINE

72416	00414	01044	37	01136	01045	SHIFT COLUMN SELECTOR
72417	00415	01045	41	01132	01133	BB COLUMNS
72420	00416	01046	31	01125	00017	$L \cdot 2^{15} \rightarrow (A)$
72421	00417	01047	37	01146	01144	$10^{L} \rightarrow (A)$
72422	00420	01050	11	20000	01112	STORE 10^{L}
72423	00421	01051	31	01153	00017	$R \cdot 2^{15} \rightarrow (A)$
72424	00422	01052	37	01146	01144	$10^{R-1} \rightarrow (Q)$ OR $1 \rightarrow (Q)$ IF $R = 0$
72425	00423	01053	16	01156	00000	RESET (00000) FOR RERUN
72426	00424	01054	31	01166	00023	$2^{34} \rightarrow (A)$
72427	00425	01055	73	10000	10000	$2^{34}/10^{R-1} \rightarrow$ ROUNDING TERM
72430	00426	01056	31	01112	00107	$10^{L} \cdot 2^{-1} \rightarrow$ DIVIDE ROUND
72431	00427	01057	35	10000	10000	STORE ROUNDING TERMS
72432	00430	01060	12	30000	01132	STORE $N \cdot 2^{S}$
72433	00431	01061	00	30000	30000	$N \cdot 2^{35} \rightarrow (A)$
72434	00432	01062	35	10000	20000	ADD ROUNDING TERMS
72435	00433	01063	73	01112	20000	$N \cdot 2^{35}/10^{L} \rightarrow (A)$
72436	00434	01064	35	20000	01112	$N \cdot 2^{36}/10^{L} \rightarrow N'$
72437	00435	01065	37	01136	01066	L TIMES THRU
72440	00436	01066	41	01125	01137	CONVERSION LOOP
72441	00437	01067	16	01157	01136	SET EXIT IN IMAGE ROUTINE
72442	00440	01070	41	01153	01120	STORE DECIMAL POINT IF $R > 0$
72443	00441	01071	41	01153	01137	$R - 1$ TIMES THRU CONVERSION
72444	00442	01072	15	01060	01073	SET NEXT INSTRUCTION
72445	00443	01073	55	30000	00000	$N \cdot 2^{S} \rightarrow (Q)$
72446	00444	01074	21	01026	01166	STEP PARAMETER
72447	00445	01075	21	01060	01166	STEP D
72450	00446	01076	16	01020	01136	SET EXIT
72451	00447	01077	13	01160	20000	$-1 \rightarrow (A)$
72452	00450	01100	44	01130	01133	N NEGATIVE ?
72453	00451	01101	37	01020	01030	SET α_2
72454	00452	01102	75	20014	01104	SHIFT L_8 RIGHT THIRD OF
72455	00453	01103	55	01221	00010	CARD IMAGE
72456	00454	01104	75	30003	01106	SET UP
72457	00455	01105	11	01150	01110	PUNCH ORDERS
72460	00456	01106	16	01156	00000	RESET (00000) FOR RERUN
72461	00457	01107	43	01151	01016	ALL 12 ROWS PUNCHED ?

CV-37

PUNCH ROUTINE

72462	00460	01110	00	30000	30000	PUNCH
72463	00461	01111	00	30000	30000	ONE
72464	00462	01112	00	30000	30000	ROW
72465	00463	01113	75	20003	01106	STEP
72466	00464	01114	23	01110	01160	PUNCH ORDERS
72467	00465	01115	37	01143	01044	SET FOR ZERO SUPPRESSION
72470	00466	01116	43	01125	01127	IF L = 0, NO ZERO SUPPRESSION
72471	00467	01117	45	00000	01133	
72472	00470	01120	31	01161	00000	3 ⟶ (A)
72473	00471	01121	35	01111	01124	SELECT ROW 3
72474	00472	01122	35	01162	01125	SELECT ROW 8
72475	00473	01123	55	01110	00000	COLUMN SELECTOR ⟶ (Q)
72476	00474	01124	00	30000	30000	EXTRACT 3
72477	00475	01125	00	30000	30000	EXTRACT 8
72500	00476	01126	33	01160	00001	SELECT ROW 12
72501	00477	01127	37	01143	01130	SET NO ZERO SUPPRESSION
72502	00500	01130	35	01111	01132	ADD ROW SELECTOR TO (A)
72503	00501	01131	55	01110	00000	COLUMN SELECTOR ⟶ (Q)
72504	00502	01132	00	30000	30000	EXTRACT DIGIT
72505	00503	01133	55	01110	00043	SR, COLUMN SELECTOR
72506	00504	01134	44	01135	01136	1/3 IMAGE FILLED ?
72507	00505	01135	21	01111	01167	YES, STEP ROW SELECTOR
72510	00506	01136	45	00000	30000	EXIT OF IMAGE ROUTINE
72511	00507	01137	31	01112	00002	4 N' ⟶ (A)
72512	00510	01140	32	01112	00001	10N' ⟶ (A)
72513	00511	01141	11	20000	01112	FRACTIONAL PART ⟶ N'
72514	00512	01142	34	20000	00044	INTEGER PART ⟶ (R)
72515	00513	01143	47	01127	30000	THIS DIGIT = 0 ?
72516	00514	01144	15	20000	01146	SET REPEAT ORDER
72517	00515	01145	54	01160	10000	1 ⟶ (A), (Q)
72520	00516	01146	75	30000	30000	FORM 10^{x} IN (A)
72521	00517	01147	71	20000	01164	AND 10^{x-1} IN (Q)
72522	00520	01150	77	00000	01234	PRESET
72523	00521	01151	77	10000	01204	PRESET
72524	00522	01152	77	10000	01220	PRESET
72525	00523	01153	00	30000	30000	
72526	00524	01154	53	10000	01173	PRESET
72527	00525	01155	31	01132	00043	PRESET

PX 71900-4-37

CV-37

PUNCH ROUTINE

72530	00526	01156	00 00000 01000	
72531	00527	01157	00 00000 01071	
		1160	00 00000 00001	1
		1161	00 00000 00003	3
		1162	00 00000 00005	5
		1163	00 00000 00010	8
		1164	00 00000 00012	10
		1165	00 00000 00077	TWO OCTAL DIGIT EXTRACTOR
		1166	00 00001 00000	U ADVANCE
		1167	00 00000 00014	12
		1170	40 00000 00000	PRESET FOR COLUMN SELECTOR
		1171		CARD IMAGE COLS 1-36 ROW 12
		1172		ROW 11
		1173		ROW 0
		1174		ROW 1
		1175		ROW 2
		1176		ROW 3
		1177		ROW 4
		1200		ROW 5
		1201		ROW 6
		1202		ROW 7
		1203		ROW 8
		1204		ROW 9
		1205		CARD IMAGE COLS 37-72 ROW 1
		1206		ROW 1
		1207		ROW 0
		1210		ROW 1
		1211		ROW 2
		1212		ROW 3
		1213		ROW 4
		1214		ROW 5
		1215		ROW 6
		1216		ROW 7
		1217		ROW 8
		1220		ROW

PX 71900-4-37

CV-37
PUNCH ROUTINE

1221	CARD IMAGE COLS 73-80	ROW 12
1222		ROW 11
1223		ROW 0
1224		ROW 1
1225		ROW 2
1226		ROW 3
1227		ROW 4
1230		ROW 5
1231		ROW 6
1232		ROW 7
1233		ROW 8
1234		ROW 9

USEful Note #5
SUBJECT: Double Precision Add, Multiply (RR)

12 December 1956
Talmadge

HEADING

In order to complete the following routines as library subroutines in the USE format, the following heading should precede each routine. This heading will add 0.04 MS to each subroutine.

Loc	op	u	v	Remarks
entry	MJ	0	start	entry line
error	RJ	diag + 2	diag	error exit
exit	MJ	0	fill	normal exit
b_1	fill	0	0	input data
b_2	fill	0	0	
c_1	fill	0	0	
c_2	fill	0	0	
d_1	fill	0	0	output data
d_2	fill	0	0	
start				subroutine
.				
.				
.				

DOUBLE PRECISION ADD (METHOD 1)

Want $b + c = d$ $b = b_1 + b_2$ ditto c and d

Assume each word contains 35 bits preceded by a sign bit in one's compliment form.

Loc	op	u	v	Remarks
start	TP	b_1	A	add (0.24 MS)
	AT	c_1	A	
	LA	A	35	
	AT	b_2	A	
	AT	c_2	A	
unpack	LT1	0	d_2	unpack (0.17 MS)
	LT	1	d_1	
	TP	mask	Q	
	QS	d_1	d_2	
	MJ	0	exit	
mask	40	0	0	

Total time = 0.41 MS

This routine ignores the fact that overflow may occur into bit 71 of the accumulator.

DOUBLE PRECISION ADD (METHOD 2)

Want $b + c = d$ $b = b_1 + b_2$ ditto c and d (one's compliment form)

All words with subscript 2 contain 36 data bits and all words with subscript 1 contain 35 data bits preceded by a sign bit. Note: all numbers (b, c, and d) are in 1's compliment form.

Loc	op	u	v	Remarks
start	SP	b_1	0	add 0.20 MS
	SA	c_1	36	
	SA	b_2	0	
	SA	c_2	0	
	LT1	0	d_2	unpack (0.08 MS)
	LT	0	d_1	
	MJ	0	exit	

Total time = 0.28 MS

This routine ignores the fact that overflow may occur into bit 72 of the accumulator. To test for overflow is not difficult nor lengthy but is omitted here. If the operations of multiply or divide are to be used also, this method of packing numbers for addition complicates these routines unduly and is not significantly faster for addition than method 1.

DOUBLE PRECISION MULTIPLY (METHOD 1)

Want $b \cdot c = d \qquad d = b_1 \cdot c_1 + b_1 \cdot c_2 + b_2 \cdot c_1 + b_2 \cdot c_2$

Assume 35 bits and a sign bit in words with subscript 2 and 34 bits and two sign bits in words with subscript of 1 in one's compliment form. (Note: no change in method 1 add is necessary with these restrictions.)

Loc	op	u	v	Remarks
start	MP	b_2	c_2	round
	LT	1	temp	
	MP	Q	c_1	mult.
	MA	b_1	c_2	
	AT	temp	A	
	SJ	neg	pos	
neg	SS	round	0	round
	MJ	0	pos + 1	
pos	SA	round	0	
	LT	1	temp	
	MP	Q	c_1	mult.
	AT	temp	A	
unpack	LT1	0	d_2	
	LT	1	d_1	
	TP	round	Q	unpack
	QS	d_1	d_2	
	MJ	-	exit	
round	40	0	0	

Max. error ± 1/2 in the last place kept

average time of round = 0.39 MS

average time of multiply	=	0.95 MS
average time of unpack	=	0.17 MS
average total time	=	1.52 MS

(Times, assume 0.27 MS for MP)

Note: this routine assumes only 69 places as input and output, (in general overflow occurs if more bits are assumed).

DOUBLE PRECISION MULTIPLY (METHOD 2)

Want $b \cdot c = d$

(See method 1 of multiply for scaling on b, c, and d)

$d = b_1 \cdot c_1 + b_1 \cdot c_2 + b_2 + c_1$

Loc	op	u	v	Remarks
start	MP	b_2	c_1	Form $(b_2 \cdot c_1) + (b_1 \cdot c_2)$
	MA	b_1	c_2	
	LT	1	temp	scale down
	MP	Q	c_1	Form d
	AT	temp	A	
unpack				See method 1 for unpacking routine

Average time = 0.88 MS (assumes 0.27 MS for MP)

Average total time = 1.05 MS (including unpacking)

Max. error ± 1 in the last place kept

The main difference in these two methods of multiply is in the accuracy obtained. It should be noted that the maximum difference in the error between the two methods is only one place.

Further note that no overflow can occur as the result of unpacking (as in the add routines) in these two multiply routines.

Talmadge

11 December 1956

USEful Note #6

Subject: Preliminary Information- General Tape Handler (RR)

PRELIMINARY INFORMATION--GENERAL TAPE HANDLER

The General Tape Handler is a routine to facilitate the use of the 1103A magnetic tape system. The routine provides built in checks which prevent the initiation of erroneous tape operations, and an error entry which makes use of information on the 'last' tape operation initiated correctly, to recover from faults occurring while tape is in operation. A parity check is made for each block read, and blocks in which errors occur are automatically re-read at different bias levels.

Initiation of tape operation

The tape handler is in the USE subroutine form with one parameter word which specifies the type of operation, etc. The parameter word is placed in location GTH+3, and routine is entered with a return jump.

Form of parameter word

R M T NN SS VVVVV

R - octal digit which specifies operation
- 1 rewind
- 2 rewind/interlock
- 3 move forward
- 4 move backward
- 5 read forward
- 6 read backward
- 7 write

M - blockette spacing (for write only)
M = 1 - 0" blockette spacing
= 2 - 0.1" blockette spacing
= 4 - 1.0" blockette spacing

T - block spacing and density (for write only)

T = 0	1" block space	128 lines/in. density
= 1	1" block space	50 lines/in. density
= 2	2.4" block space	128 lines/in. density
= 3	2.4" block space	50 lines/in. density

NN - number of blocks to be written (write only)

MTNN - number of blocks to be moved or read

SS - number of servo on which operation is to be performed

VVVVV - High Speed Storage address for read and write

1) For write first word is taken from location specified by VVVVV. Succeeding words are taken from ascending storage locations.

2) For read forward first word is read into location specified by VVVVV and succeeding words are read into ascending storage locations.

3) For read backward first word is read into location VVVVV + 120n-1. (where n is number of blocks to be read) Succeeding words are read into descending storage locations, and the last word to be read is placed in VVVVV.

Sentinel blocks

The routine recognizes two types of sentinel blocks; lead and final. A complete block of all ƶ words is recognized as a final sentinel. A block whose first and last blockettes are all Ƶ's is recognized as a lead sentinel. The other four blockettes may also contain all ƶ words, but at least one word must be a non - ƶ word.

Errors detected by the routine

Upon detection of many errors, a code word is placed in a diagnostic routine (not a part of this routine), followed by a return jump to the diagnostic routine.

Errors which are treated this way are:
(1) No operation specified (R = 0)
(2) No blockette space designation
(3) Read which would clobber the routine if executed
(4) Read which would clobber F_1 to F_5, or would try to read words into locations outside of high speed storage
(5) Write, which takes words from locations outside of H.S.S.

If a parity error occurs, and attempts to reread the block at different bias levels are unsuccessful, the routine prints out p rf for read forward, or p rb for read backward and stops with PAK set to re-enter the reread routine.

If a final sentinel block is reached on a read or move forward n blocks before the nth block is read, tape is stopped, repositioned to first word of the sentinel block. The routine prints out s rf or s mf and stops with PAK set to exit from the routine. If the sentinel block is the nth block to be read, tape is repositioned to first word of block, but no print out is made and computer is not stopped.

If a lead sentinel is reached on a read or move backward n blocks before the nth block is read, tape is stopped, s rb or s mb is printed and computer stops with PAK set to exit from routine. Tape is not repositioned to last word of sentinel block, since the lead sentinel may contain information necessary to identify tape on next read forward.

If the lead sentinel block is the nth block tape is stopped but computer is not stopped.

Error entry

Most faults occuring while the tapes are in operation are recoverable by re-entering the routine from the error entry.

The one error which may not be recoverable is a >720 error in the last block to be read or moved. In this case computer coasts for ~80 m.s. before the fault stop. If < 6 extra lines are detected, a second tape operation cannot be initiated during this 80 m.s. coasting time since the instruction which would normally stop the tape has hung up in TCR without giving an IOB resume. If program is not altered during this coasting time, and other tape operations have not been started, recovery can be made by re-entering at error entry.

Indications of >720 error in last block

MT B fault indicator illuminated. Sproket Error indicator in tape control cabinet illuminated. PCR contains something other than an ER. (If PCR contains an ER, the error was either a <720 fault or a >720 fault in some block other than the last block to be read.) If less than six extra lines were detected TCR contains stop bits (6000). If more than six extra lines were detected the contents of TCR cannot be predicted, and another tape operation may have been initiated and a second fault may have been generated when the computer stopped.

NOTE: Copies of the GTH flow-charts are available upon request.

Talmadge

26 December 1956

USEful Note, (ML1) Number 7

The following write-up contains a description of three non-standard 1103A routines to be used as preliminary acceptance tests.

1. NAME: ML TP1 Acceptance Test, Tape
Systems Development Group - 1 October 1956
Lockheed Missile Systems Division

METHOD: One hundred and twenty pseudo-random numbers are generated and stored in core. These numbers are then written onto tape, read back, and shifted circulary in sequence. The process is repeated 120 times and the resulting block is compared with an undisturbed image of the original block.

OPERATION: 1. Set PAK to 00001.

2. Set low order of Q-register with Uniservo number if test is to be run using only one Uniservo.

3. Set M.J. switches.

	1	2	3
on	Test One Uniservo	Rewinds	-
off	Test All Uniservos	No Rewinds	-

4. Set MS switches.

	1	2	3
on	Stop and Continue	-	-
off	No Stop Repeat Test	-	-

5. Depress Start

The following indications will appear on the typewriter:

OUTPUT: HUZZAH - Test Successful

BADTPE - Program fails on any bias.

TAPE n NORMAL - Bias failure

TAPE n HIGH - Bias failure

TAPE n LOW - Bias failure

STOPS: (PAK) = 00001 Intermediate or final stop, depressing start button continues test with rewinding of Uniservos.

(PAK) = 00003 Intermediate or final stop, depressing start button continues test with no rewinding of Uniservos.

TIMING: Approximately three minutes for one test.

2. NAME: ML MD1, Magnetic Drum
Systems Development Group, 1 October 1956
Missile Systems Division

METHOD: Four thousand and ninety-six pseudo-random numbers are generated in two stops and stored as working storage on a logical drum. The same numbers are stored as amirror image on another logical drum. The working storage is then rotated between drum and core 4,096 times with an offset of one. The final rotation is then compared against the mirror image. The program tests sequentially drum 5 working - drum 7 mirror, drum 6 working - drum 7 mirror, drum 7 working - drum 5 mirror, and finally drum 4 upper as working - drum 5 upper as mirror. It should be noted that the program is divided into four independent parts and thus a start may be initiated at any of the four divisions.

OPERATION: 1. Set F1 to 40001.

2. Set PAK to 40000, (40110, 40216, 40324, optional).

3. Depress Start Button.

STOPS; (PAK) = 40401 Final stop

(PAK) = 40454 Intermediate stop if error occurred while checking 2048 numbers.

OUTPUT: Typewriter is utilized for monitoring.

1. START w-m - Indicates which test is being executed by noting the working and mirror drums.

2. OK1 - Indicates first 2048 numbers check

3. OK2 - Indicates second 2048 numbers check.

ERROR ROUTINE: 1. Obtain as print out in octal of all errors occurring in checking 2048 random numbers using the following format:

Location	Working Word	Mirror Word
XXXXX	XXXXXXXXXXXX	XXXXXXXXXXXX

2. Optional Start

a. Set high order position of Q-register to one for

repeat of test using same working and mirror storage
- Depress Start

b. Depressing Start continues test sequentially.

TIME: Approximately two hours for a complete test, 35 minutes each for the first three parts.

3. NAME: ML MD2, Leap Frog
Systems Development Group, 1 October 1956
Lockheed Missile Systems Division

PURPOSE: This program is intended to test the arithmetic circuits, memory retention, transfer of information from core to drum to core, and the interpretation circuits of the 1103A.

METHOD: The program prepares a copy of itself which is transmitted ahead to a new core location. During this process the arithmetic and interpretation circuits are checked. After transmission, the core to drum to core exchange is tested except when the program overlaps F1.

OPERATION: 1. Set PAK to 46100.

2. Set MS switches.

	1	2	3
on	Stop, 4096 leaps	Stop, 1 leap	-
off	Continue	Continue	-

3. Depress Start Button

STOPS: 1. Final Stop - Either a failure in the arithmetic section or a check sum failure from the core to drum to core transfer.

2. Intermediate Stop - Depends upon MS switch setting - depressing Start Button continues test.

3. Any Other Stop - Failure of core memory.

OUTPUT: None

TIMING: Approximately 80 minutes for 4096 leaps.

Date	LOCKHEED AIRCRAFT CORPORATION MISSILE SYSTEMS DIVISION	Model	Page
Approved	Title ML TP 1	Report No.	

STORAGE	EXEC.	OP	U	V	COMMENTS
		45	00000	00001	MACHINE F1
00001	00001	75	10005	00003	REWIND
00002	00002	17	00000	00131	TAPES
00003	00003	11	31000	00136	SAVE UNIT NUMBER
00004	00004	11	00150	00142	SET INDEX TO 119
00005	00005	16	00036	00010	SET STORE
00006	00006	71	00137	00140	GENERATE
00007	00007	11	32000	31000	AND
00010	00010	22	00026	01000	STORE
00011	00011	51	00141	00137	RANDOM
00012	00012	21	00010	00145	NUMBERS
00013	00013	41	00142	00006	
00014	00014	75	30170	00016	STORE AT
00015	00015	11	01000	02000	CORE IMAGE
00016	00016	11	00150	00142	SET NUMBER OF TIMES INDEX
00017	00017	11	00147	00143	SET TAPE INDEX
00020	00020	17	00000	00151	SET NORMAL BIAS
00021	00021	11	00153	00152	AND BIAS STATUS
00022	00022	45	10000	00027	TEST FOR ONE TAPE OR ALL TAPES
00023	00023	41	00143	00025	FIND UNISERVO NUMBER
00024	00024	35	00147	00143	RESET INDEX
00025	00025	31	32000	00014	POSITION
00026	00026	45	00000	00031	UNISERVO NUMBER
00027	00027	11	00136	32000	UNISERVO NUMBER TO A
00030	00030	34	00145	00014	LESS ONE
00031	00031	35	00173	00176	ESTABLISH WRITE CODE
00032	00032	16	32000	00175	SET READ CODE
00033	00033	16	32000	00174	SET MOVE CODE
00034	00034	17	00000	00176	WRITE
00035	00035	75	10170	00037	ONE
00036	00036	77	10000	01000	BLOCK
00037	00037	17	00000	00174	MOVE BACK
00040	00040	11	00146	00144	SET BIAS INDEX
00041	00041	17	00000	00175	READ ONE
00042	00042	75	10170	00044	BLOCK
00043	00043	76	10000	01001	INTO
00044	00044	76	00000	32000	BUFFER
00045	00045	11	01170	01000	CIRCULATE
00046	00046	47	00047	00107	TEST FOR PARTY ERROR
00047	00047	61	00000	00162	CARRIAGE RETURN
00050	00050	41	00144	00055	TEST BIAS
00051	00051	11	00157	31000	PRINT WORD
00052	00052	37	00106	00101	BADTPE
00053	00053	61	00000	00162	CARRIAGE RETURN
00054	00054	57	00000	00001	STOP
00055	00055	31	32000	00017	SHIFT BIAS NUMBER
00056	00056	35	00021	00073	SET FOR NEXT STATUS
00057	00057	11	00156	31000	PRINT WORD
00060	00060	37	00106	00101	TAPE
00061	00061	61	00000	00172	SHIFT DOWN
00062	00062	55	00175	31025	POSITION AND
00063	00063	31	31000		OBTAIN

FORM # MSD 8520

Date	LOCKHEED AIRCRAFT CORPORATION MISSILE SYSTEMS DIVISION	Model	Page
Approved	Title ML TP 1	Report No.	

STORAGE	EXEC.	OP	U	V	COMMENTS
00064	00064	22	00003	32000	UNISERVO NUMBER
00065	00065	35	00053	00066	PRINT
00066	00066				NUMBER
00067	00067	75	00004	00071	FOUR
00070	00070	61	00000	00170	SPACES
00071	00071	11	00152	31000	PRINT
00072	00072	37	00106	00101	STATUS WORD
00073	00073				NEW STATUS
00074	00074	31	00144	00014	SET UP
00075	00075	35	00151	00066	AND
00076	00076	17	00000	00066	CHANGE BIAS
00077	00077	17	00000	00174	BACKSPACE
00100	00100	45	00000	00041	REREAD
00101	00101	61	00000	00171	SHIFT UP
00102	00102	11	00147	00066	SET INDEX
00103	00103	55	31000	00006	POSITION
00104	00104	61	00000	31000	AND TYPE
00105	00105	41	00066	00103	TEST END
00106	00106	45	00000		BACK
00107	00107	41	00142	00020	TEST TIMES
00110	00110	75	30170	00112	SUBTRACT
00111	00111	23	01000	02000	ORIGINAL
00112	00112	75	30170	00114	OBTAIN
00113	00113	12	01000	01000	MAGNITUDES
00114	00114	23	32000	32000	ZERO ACCUMULATOR
00115	00115	75	20170	00121	TEST
00116	00116	42	01000	00117	FOR ERROR
00117	00117	11	00161	31000	ERROR WORD TO Q
00120	00120	45	00000	00122	
00121	00121	11	00160	31000	NO ERROR WORD
00122	00122	61	00000	00162	CARRIAGE RETURN
00123	00123	37	00106	00101	TYPE WORD
00124	00124	61	00000	00162	
00125	00125	11	00136	31000	UNISERVO NUMBER TO Q
00126	00126	45	20000	00130	REWIND OPTION
00127	00127	56	10000	00003	STOP AND OR CONTINUE NO REWIND
00130	00130	56	10000	00001	STOP AND OR CONTINUE γ REWIND
00131	00131	2	00200	10000	REWIND CODES
00132	00132	2	00200	20000	
00133	00133	2	00200	30000	
00134	00134	2	00200	40000	
00135	00135	2	00200	50000	
00136	00136				UNISERVO NUMBER
00137	00137	15	44755	54415	CONSTANT
00140	00140		2	30455	POWER OF FIVE
00141	00141	37	77777	77777	MASK
00142	00142				TIMES INDEX
00143	00143				TAPE INDEX
00144	00144				BIAS INDEX
00145	00145			00001	1
00146	00146			00003	3
00147	00147			00005	5

FORM # MSD 8520

STORAGE	EXEC.	OP	U	V		COMMENTS
00150	00150			00167	119	
00151	00151	2	00001	50000	NORMAL BIAS	
00152	00152				BIAS STATUS	
00153	00153	6	03120	73011	NORMAL	
00154	00154	11	03310	40404	LOW	
00155	00155	5	14130	50404	HIGH	
00156	00156	1	30152	00404	TAPE	
00157	00157	23	30220	11520	BADTPE	
00160	00160	5	34212	13005	HUZZAH	
00161	00161	6	03130	30322	NOGOOD	
00162	00162			00045		
00163	00163			00052	FLEX 1	
00164	00164			00074	FLEX 2	
00165	00165			00070	FLEX 3	
00166	00166			00064	FLEX 4	
00167	00167			00062	FLEX 5	
00170	00170			00004	SPACE	
00171	00171			00047	SHIFT UP	
00172	00172			00057	SHIFT DOWN	
00173	00173	2	00606	10001	TAPE CONSTANT	
00174	00174	2	00614		MOVE BACKWARD	
00175	00175	2	00602		READ FORWARD	
00176	00176	2	00606		WRITE	

LOCKHEED AIRCRAFT CORPORATION
MISSILE SYSTEMS DIVISION

Date | Model | Page
Approved | Title: ML MD 1 | Report No.

STORAGE	EXEC.	OP	U	V	COMMENTS
40000	40000	45	00000	40002	DRUM START
40001	40001	45	00000		FI JUMP
40002	40002	75	30032		BRING RANDOM NUMBER
40003	40003	11	40004		PROGRAM TO CORE
40004	00000	71	00026	00024	COMPUTE R SUB I
40005	00001	11	32000	31003	R SUB I TO Q
40006	00002	22	00024	02000	STORE RANDOM NUMBER
40007	00003	51	00025	00024	SET R SUB I+1
40010	00004	21	00002	00027	MODIFY STORE ADDRESS
40011	00005	41	00030		LOOP TEST
40012	00006	75	34000	00010	STORE WORKING NUMBERS AT LOWER
40013	00007	11	02000	50000	PART OF DRUM 5-CALLED 5L
40014	00010	75	34000	00012	STORE MIRROR AT LOWER
40015	00011	11	02000	70000	PART OF DRUM 7 - CALLED 7L
40016	00012	71	00026	00024	COMPUTE SECOND
40017	00013	11	32000	31036	SET OF RANDOM
40020	00014	22	00024	02000	NUMBERS
40021	00015	51	00025	00024	
40022	00016	21	00014	00027	MODIFY
40023	00017	41	00031	00012	LOOP TEST
40024	00020	75	34000	00022	STORE AT UPPER PART
40025	00021	11	02000	54000	OF DRUM 5 - CALLED 5U
40026	00022	75	34000	40037	STORE AT UPPER PART
40027	00023	11	02000	74000	OF DRUM 7 - CALLED 7U
40030	00024	15	44755	54415	R SUB ZERO
40031	00025	37	77777	77777	
40032	00026		2	30455	5 TO THE SEVENTH
40033	00027			00001	
40034	00030			03777	INDEX
40035	00031			03777	INDEX
40036	40036		7	00005	INDICATIVE WORK 5 - MIRROR 7
40037	40037	37	40433	40406	RJ TO PRINT
40040	40040	11	40402	40403	SET NUMBER OF ROTATIONS INDEX
40041	40041	75	37777	40043	DRUM 5 TO CORE AND
40042	40042	11	50000	00001	LAST WORD
40043	40043	11	57777		ON 5 TO ZERO
40044	40044	75	37777	40046	FILL DRUM 5
40045	40045	11	00000	50000	FROM CORE
40046	40046	11	07777	57777	
40047	40047	41	40403	40041	LOOP TEST FOR ROTATING
40050	40050	75	34000	40052	DRUM 7 LOWER TO
40051	40051	11	70000	04000	UPPER CORE
40052	40052	75	34000	40054	MAKE LOWER CORE
40053	40053	27	00000	04000	ZERO BY CC
40054	40054	23	32000	32000	ZERO ACCUM
40055	40055	75	34000	40057	SET SIGN OF
40056	40056	12	00000		LOWER CORE +
40057	40057	75	24000	40061	TEST FOR
40060	40060	42	00000	40475	ERROR
40061	40061	37	40451	40450	TEST FOR PREVIOUS ERROR
40062	40062	61	00000	40134	TYPE
40063	40063	61	00000	40155	OUT

STORAGE	EXEC.	OP	U	V	COMMENTS
40064	40064	61	00000	40005	THE
40065	40065	61	00000	40017	SUCCESS
40066	40066	61	00000	40050	OK1
40067	40067	75	34000	40071	DRUM 5 UPPER
40070	40070	11	54000		TO LOWER CORE
40071	40071	75	34000	40073	DRUM TO 7 UPPER
40072	40072	11	74000	04000	TO UPPER CORE
40073	40073	75	34000	40075	MAKE LOWER
40074	40074	27	00000	04000	CORE ZERO
40075	40075	75	34000	40077	SET SIGN OF
40076	40076	12	00000		LOWER CORE 7
40077	40077	23	32000	32000	SET ACCUMULATOR ZERO
40100	40100	75	24000	40102	TEST CORE FOR
40101	40101	42	00000	40475	GREATER THAN ZERO
40102	40102	37	40451	40450	TEST FOR PREVIOUS ERROR
40103	40103	61	00000	40134	TYPE
40104	40104	61	00000	40155	OUT
40105	40105	61	00000	40005	THE
40106	40106	61	00000	40017	SUCCESS
40107	40107	61	00000	40442	OK2
40110	40110	75	30032		START THE TESTING OF THE
40111	40111	11	40112		NEXT SECTION OF DRUM
40112	00000	71	00026	00024	COMPUTE
40113	00001	11	32000	31000	RANDOM
40114	00002	22	00024	02000	NUMBERS
40115	00003	51	00025	00024	
40116	00004	21	00002	00027	MODIFY STORE
40117	00005	41	00030		LOOP TEST
40120	00006	75	34000	00010	STORE FROM CORE TO
40121	00007	11	02000	60000	DRUM 6 LOWER
40122	00010	75	34000	00012	STORE MIRROR TO
40123	00011	11	02000	70000	DRUM 7 LOWER
40124	00012	71	00026	00024	COMPUTE SECOND
40125	00013	11	32000	31000	SET OF RANDOM
40126	00014	22	00024	02000	NUMBERS
40127	00015	51	00025	00024	R SUB I TO CORE
40130	00016	51	00014	00027	MODIFY STORE
40131	00017	41	00031	00012	LOOP TEST
40132	00020	75	34000	00022	STORE FROM CORD
40133	00021	11	02000	64000	TO 6 UPPER
40134	00022	75	34000	40145	STORE MIRROR TO
40135	00023	11	02000	74000	DRUM 7 UPPER
40136	00024	15	44755	54415	R SUB ZERO
40137	00025	37	77777	77777	
40140	00026		2	30455	5 TO THE SEVENTH
40141	00027			00001	
40142	00030			03777	INDEX
40143	00031			03777	INDEX
40144	40144		7	00006	INDICATIVE WORK 6 - MIRROR 7
40145	40145	37	40433	40406	RJ TO PRINT
40146	40146	11	40402	40403	SET ROTATION INDEX
40147	40147	75	37777	40151	ROTATE RANDOM

Date	LOCKHEED AIRCRAFT CORPORATION MISSILE SYSTEMS DIVISION	Model	Page
Approved	Title ML MD 1	Report No.	

STORAGE	EXEC.	OP	U	V	COMMENTS
40150	40150	11	60000	00001	NUMBERS FROM
40151	40151	11	67777		DRUM 6
40152	40152	75	37777	40154	REPLACE DRUM
40153	40153	11	00000	60000	FROM CORE WITH
40154	40154	11	07777	67777	OFF SET
40155	40155	41	40403	40147	TEST INDEX
40156	40156	75	34000	40160	DRUM 7 LOWER
40157	40157	11	70000	04000	TO UPPER CORE
40160	40160	75	34000	40162	MAKE LOWER
40161	40161	27	00000	04000	CORE ZERO
40162	40162	23	32000	32000	CLEAR ACCUMULATOR
40163	40163	75	34000	40165	SET SIGN OF
40164	40164	12	00000		LOWER CORE γ
40165	40165	75	24000	40167	TEST CORE FOR
40166	40166	42	00000	40475	GREATER THAN ZERO
40167	40167	37	40451	40450	TEST FOR PREVIOUS ERROR
40170	40170	61	00000	40134	TYPE
40171	40171	61	00000	40155	OUT
40172	40172	61	00000	40005	THE
40173	40173	61	00000	40017	SUCCESS
40174	40174	61	00000	40050	OK1
40175	40175	75	34000	40177	SECOND PART OF TEST
40176	40176	11	64000		6U TO LOWER CORE
40177	40177	75	34000	40201	DRUM 7 UPPER
40200	40200	11	74000	04000	TO UPPER CORE
40201	40201	75	34000	40203	MAKE LOWER
40202	40202	27	00000	04000	CORE ZERO
40203	40203	75	34000	40205	SET SIGN OF
40204	40204	12	00000		LOWER COREγ
40205	40205	23	32000	32000	CLEAR ACCUMULATOR
40206	40206	75	24000	40210	TEST FOR
40207	40207	42	00000	40475	GREATER THAN ZERO
40210	40210	37	40451	40450	TEST FOR PREVIOUS ERROR
40211	40211	61	00000	40134	TYPE
40212	40212	61	00000	40155	OUT
40213	40213	61	00000	40005	THE
40214	40214	61	00000	40017	SUCCESS
40215	40215	61	00000	40442	OK2
40216	40216	75	30032		START THE TESTING OF THE
40217	40217	11	40220		NEXT SECTION OF DRUM
40220	00000	71	00026	00024	COMPUTE
40221	00001	11	32000	31000	RANDOM
40222	00002	22	00024	02000	NUMBERS
40223	00003	51	00025	00024	
40224	00004	21	00002	00027	MODIFY STORE
40225	00005	41	00030		LOOP TEST
40226	00006	75	34000	00010	STORE FROM CORE
40227	00007	11	02000	70000	TO DRUM 7 LOWER
40230	00010	75	34000	00012	STORE MIRROR AT
40231	00011	11	02000	50000	DRUM 5 LOWER
40232	00012	71	00026	00024	COMPUTE SECOND SET OF
40233	00013	11	32000	31000	RANDOM NUMBERS

Date	LOCKHEED AIRCRAFT CORPORATION MISSILE SYSTEMS DIVISION	Model	Page
Approved	Title ML MD 1	Report No.	

STORAGE	EXEC.	OP	U	V	COMMENTS
40234	00014	22	00024	02000	
40235	00015	51	00025	00024	
40236	00016	21	00014	00027	MODIFY STORE
40237	00017	41	00031	00012	LOOP TEST
40240	00020	75	34000	00022	STORE FROM CORE
40241	00021	11	02000	74000	TO 7 UPPER
40242	00022	75	34000	40253	STORE MIRROR AT
40243	00023	11	02000	54000	DRUM 5 UPPER
40244	00024	15	44755	54415	
40245	00025	37	77777	77777	
40246	00026		2	30455	
40247	00027			00001	
40250	00030			03777	INDEX
40251	00031			03777	INDEX
40252	40252		5	00007	IDENTIFICATION
40253	40253	37	40433	40406	RJ TO PRINT
40254	40254	11	40402	40403	SET ROTATION INDEX
40255	40255	75	37777	40257	ROTATE
40256	40256	11	70000	00001	THE WORKING
40257	40257	11	77777		STORAGE
40260	40260	75	37777	40262	
40261	40261	11	00000	70000	
40262	40262	11	07777	77777	
40263	40263	41	40403	40255	LOOP TEST
40264	40264	75	34000	40266	DRUM 5L TO
40265	40265	11	50000	04000	UPPER CORE
40266	40266	75	34000	40270	MAKE LOWER CORE
40267	40267	27	00000	04000	ZERO
40270	40270	23	32000	32000	CLEAR A
40271	40271	75	34000	40273	SET SIGN OF
40272	40272	12	00000		LOWER CORE γ
40273	40273	75	24000	40275	TEST FOR CORE
40274	40274	42	00000	40475	GREATER THAN ZERO
40275	40275	37	40451	40450	TEST FOR PREVIOUS ERROR
40276	40276	61	00000	40134	TYPE
40277	40277	61	00000	40155	OUT
40300	40300	61	00000	40005	THE
40301	40301	61	00000	40017	SUCCESS
40302	40302	61	00000	40050	OK1
40303	40303	75	34000	40305	DRUM 7 UPPER
40304	40304	11	74000		TO LOWER CORE
40305	40305	75	34000	40307	DRUM 5 UPPER
40306	40306	11	54000	04000	TO UPPER CORE
40307	40307	75	34000	40311	MAKE
40310	40310	27	00000	04000	CORE ZERO
40311	40311	75	34000	40313	SET SIGN OF
40312	40312	12	00000		LOWER CORE γ
40313	40313	23	32000	32000	CLEAR ACCUMULATOR
40314	40314	75	24000	40316	TEST FOR CORE
40315	40315	42	00000	40475	GREATER THAN ZERO
40316	40316	37	40451	40450	TEST FOR PREVIOUS ERROR
40317	40317	61	00000	40134	TYPE

Date	LOCKHEED AIRCRAFT CORPORATION MISSILE SYSTEMS DIVISION	Model	Page
Approved	Title ML MD 1	Report No.	

STORAGE	EXEC.	OP	U	V	COMMENTS
40320	40320	61	00000	40155	OUT
40321	40321	61	00000	40005	THE
40322	40322	61	00000	40017	SUCCESS
40323	40323	61	00000	40442	OK2
40324	40324	75	30017		START TESTING THE NEXT
40325	40325	11	40326		SECTION OF DRUM
40326	00000	71	00014	00012	COMPUTE THE
40327	00001	11	32000	31000	RANDOM NUMBERS
40330	00002	22	00024	02000	
40331	00003	51	00013	00012	
40332	00004	21	00002	00015	MODIFY STORE
40333	00005	41	00016		LOOP TEST
40334	00006	75	34000	00010	STORE WORKING NUMBERS
40335	00007	11	02000	44000	IN DRUM A UPPER
40336	00010	75	34000	40346	STORE MIRROR IN
40337	00011	11	02000	54000	DRUM 5 UPPER
40340	00012	15	44755	54415	CONSTANT
40341	00013	37	77777	77777	CONSTANT
40342	00014		2	30455	CONSTANT
40343	00015			00001	CONSTANT
40344	00016			03777	INDEX
40345	40345		5	00004	
40346	40346	37	40433	40406	RJ TO PRINT
40347	40347	11	40034	40403	SET ROTATION INDEX
40350	40350	75	33777	40352	ROTATE THE
40351	40351	11	44000	04001	NUMBERS
40352	40352	11	47777	04000	BETWEEN
40353	40353	75	33777	40355	CORE AND
40354	40354	11	04000	44000	DRUM 4
40355	40355	11	07777	47777	UPPER
40356	40356	41	40403	40350	LOOP TEST
40357	40357	75	34000	40361	DRUM 4 UPPER
40360	40360	11	44000		TO LOWER CORE
40361	40361	75	34000	40363	DRUM 5 UPPER
40362	40362	11	54000	04000	TO LOWER CORE
40363	40363	75	34000	40365	MAKE LOWER
40364	40364	27	00000	04000	CORE ZERO
40365	40365	23	32000	32000	CLEAR ACCUMULATOR
40366	40366	75	34000	40370	SET SIGN OF
40367	40367	12	00000		LOWER CORE
40370	40370	75	24000	40372	TEST FOR
40371	40371	42	00000	40475	GREATER THAN ZERO
40372	40372	37	40451	40450	TEST FOR PREVIOUS ERROR
40373	40373	61	00000	40134	TYPE
40374	40374	61	00000	40155	OUT
40375	40375	61	00000	40005	THE SUCCESS
40376	40376	61	00000	40017	OK
40377	40377	75	00007	40401	SEVEN
40400	40400	61	00000	40134	BLANK LINES
40401	40401	57	00000		SUCCESS STOP-
40402	40402			07777	BASIC ROTATION CONSTANT
40403	40403				ROTATION INDEX

FORM #
MSD 8520

Date	LOCKHEED AIRCRAFT CORPORATION MISSILE SYSTEMS DIVISION	Model	Page
Approved	Title ML MD 1	Report No.	

STORAGE	EXEC.	OP	U	V	COMMENTS
40404	40404			00001	ERROR TRIGGER
40405	40405				ERROR INDEX
40406	40406	61	00000	40134	CARRIAGE RETURN
40407	40407	11	40434	31000	SET V MASK
40410	40410	51	40433	32000	PICK UP Y?1
40411	40411	34	40435	00017	COMPUTE Y-1
40412	40412	35	40436	40413	EST TRANSMIT
40413	40413				CONTENTS Y-1 TO A
40414	40414	35	40437	40430	SET UP
40415	40415	31	32000	00025	INDICATIVE FOR
40416	40416	22	00000	32000	TYPEWRITER OUTPUT
40417	40417	35	40437	40432	
40420	40420	45	00000	40421	FILL IN
40421	40421	61	00000	40155	USE UPPER CASE
40422	40422	61	00000	40471	S
40423	40423	61	00000	40404	T
40424	40424	61	00000	40414	A
40425	40425	61	00000	40326	R
40426	40426	61	00000	40404	T
40427	40427	61	00000	40426	SPACE
40430	40430				WORKING INDICATIVE
40431	40431	61	00000	40426	SPACE
40432	40432				MIRROR INDICATIVE
40433	40433	45	00000		EXIT
40434	40434			77777	MASK
40435	40435			00002	CONSTANT
40436	40436	11	00000	32000	DUMMY 1
40437	40437	61	00000	40440	DUMMY 2
40440	40440			00037	FLEX
40441	40441			00052	FLEX 1
40442	40442			00074	FLEX 2
40443	40443			00070	FLEX 3
40444	40444			00064	FLEX 4
40445	40445			00062	FLEX 5
40446	40446			00066	FLEX 6
40447	40447			00072	FLEX 7
40450	40450	41	40405	40452	TEST FOR ERROR
40451	40451	45	00000		NORMAL EXIT
40452	40452	11	40031	31000	SET Q POSITIVE
40453	40453	56	00000	40454	INTERMEDIATE STOP
40454	40454	44	40465	40455	TEST FOR REPEAT
40455	40455	11	40474	32000	NO REPEAT
40456	40456	42	40451	40461	
40457	40457	31	40464		
40460	40460	45	00000	40462	ENTER 5
40461	40461	31	40345	00001	
40462	40462	35	40451	40463	
40463	40463				ENTER 4
40464	40464			00005	CONSTANT
40465	40465	11	40451	32000	PEPEAT DRUM TEST
40466	40466	42	40472	40002	FOR 5 WORK 7 MIRROR OR
40467	40467	42	40473	40110	FOR 6 WORK 7 MIRROR OR

FORM #
MSD 8520

Date	LOCKHEED AIRCRAFT CORPORATION MISSILE SYSTEMS DIVISION	Model	Page
Approved	Title ML MD 1	Report No.	

STORAGE	EXEC.	OP	U	V	COMMENTS
40470	40470	42	40474	40216	FOR 7 WORK 5 MIRROR OR
40471	40471	45	00000	40324	FOR 4 WORK 5 MIRROR
40472	40472	45	00000	40160	TEST WORD FOR 5-7
40473	40473	45	00000	40260	TEST WORD FOR 6-7
40474	40474	45	00000	40360	TEST WORD FOR 7-5
40475	40475	16	40001	40626	SET UP PHONEY REPEAT
40476	40476	75	30003	40500	SET INIATIAL
40477	40477	11	40622	40617	CONDITIONS
40500	40500	55	31000	02017	J AND N-R TO Q AND A
40501	40501	13	32000	32000	NEGATE
40502	40502	35	40617	32000	R IN U ADDRESS
40503	40503	15	31000	40617	SET REPEAT
40504	40504	11	40634	31000	MASK TO Q
40505	40505	51	32000	31000	R TO Q AND A
40506	40506	35	40620	40620	SET TJ INSTRUCTION
40507	40507	11	40626	32000	SUCCESS EXIT TO A
40510	40510	43	40534	40520	TEST FOR
40511	40511	43	40535	40522	PROPER HALF
40512	40512	43	40536	40524	DRUM TO
40513	40513	43	40537	40526	COMPUTE TRUE
40514	40514	43	40540	40530	LOCATION
40515	40515	43	40541	40532	
40516	40516	11	40542	40635	SET 4400
40517	40517	45	00000	40551	
40520	40520	11	40543	40635	SET 5
40521	40521	45	00000	40551	
40522	40522	11	40544	40635	SET 5400
40523	40523	45	00000	40551	
40524	40524	11	40545	40635	SET 6
40525	40525	45	00000	40551	
40526	40526	11	40546	40635	SET 6400
40527	40527	45	00000	40551	
40530	40530	11	40547	40635	SET 7
40531	40531	45	00000	40551	
40532	40532	11	40550	40635	SET 7400
40533	40533	45	00000	40551	
40534	40534	45	00000	40061	DRUMS 5 AND 7
40535	40535	45	00000	40102	DRUMS 5 AND 7
40536	40536	45	00000	40167	DRUMS 6 AND 7
40537	40537	45	00000	40210	DRUMS 6 AND 7
40540	40540	45	00000	40275	DRUMS 7 AND 5
40541	40541	45	00000	40316	DRUMS 7 AND 5
40542	40542		44000	54000	WORKING 4U MIRROR 5U
40543	40543		50000	70000	WORKING 5L MIRROR 7L
40544	40544		54000	74000	WORKING 5U MIRROR 7U
40545	40545		60000	70000	WORKING 6L MIRROR 7L
40546	40546		64000	74000	WORKING 6U MIRROR 7U
40547	40547		70000	50000	WORKING 7L MIRROR 5L
40550	40550		74000	54000	
40551	40551	31	40620		OBTAIN LOCATION
40552	40552	34	40623		OF ERROR
40553	40553	15	32000	40621	

Date	LOCKHEED AIRCRAFT CORPORATION MISSILE SYSTEMS DIVISION	Model	Page
Approved	Title ML MD 1	Report No.	

STORAGE	EXEC.	OP	U	V	COMMENTS
40554	40554	31	40635		
40555	40555	32	40621		SET WORKING
40556	40556	15	32000	40563	ADDRESS
40557	40557	54	32000	00017	
40560	40560	32	40621		SET MIRROR
40561	40561	15	32000	40564	ADDRESS
40563	40562	22	00033	40636	SET LOCATION
40563	40563	11	00000	40627	PICK WORKING WORD
40564	40564	11	00000	40630	PICK MIRROR WORD
40565	40565	61	00000	40134	CARRIAGE RETURN
40566	40566	61	00000	40155	UPPER CASE
40567	40567	61	00000	40426	SPACE
40570	40570	61	00000	40426	SPACE
40571	40571	11	40631	40635	SET INDEX OF 4
40572	40572	11	40252	31000	DIGIT MASK TO Q
40573	40573	55	40636	00003	PRINT
40574	40574	51	40252	32000	THE
40575	40575	35	40437	40576	LOCATION
40576	40576				
40577	40577	41	40635	40573	
40600	40600	61	00000	40426	SPACE
40601	40601	61	00000	40426	SPACE
40602	40602	11	40632	40635	SET INDEX TO 11
40603	40603	55	40627	00003	PRINT
40604	40604	51	40252	32000	THE
40605	40605	35	40437	40606	WORKING
40606	40606				WORD
40607	40607	41	40635	40603	
40610	40610	61	00000	40426	SPACE
40611	40611	11	40632	40635	SET INDEX TO 11
40612	40612	55	40630	00003	PRINT
40613	40613	51	40252	32000	THE
40614	40614	35	40437	40615	MIRROR
40615	40615				WORD
40616	40616	45	00000	40637	JUMP TO PATCH
40617	40617				REPEAT SET UP AFTER ERROR
40620	40620				TEST CORE INSTRUCTION
40621	40621				LOCATION COUNTER
40622	40622	75	24000	40625	CONSTANT
40623	40623	42	00000	40500	CONSTANT
40624	40624				
40625	40625	11	40404	40405	SET ERROR TRIGGER
40626	40626	45	00000		SUCCESS EXIT
40627	40627				WORKING WORD FOR PRINT
40630	40630				MIRROR WORD FOR PRINT
40631	40631			00004	INDEX CONSTANT
40632	40632			00013	INDEX CONSTANT
40633	40633		1		
40634	40634		77777		MASK
40635	40635				LOC OF WORKING AND MIRROR
40636	40636				LOCATIONS
40637	40637	41	40635	40612	TEST LOOP

FORM #
MSD 8520

Date	LOCKHEED AIRCRAFT CORPORATION MISSILE SYSTEMS DIVISION	Model	Page
Approved	Title ML MD 2	Report No.	

STORAGE	EXEC.	OP	U	V	COMMENTS
46100	46100	11	46101		SET F1
46101	46101	45	00000	46102	
46102	46102	75	30160	00101	PROGRAM TO CORE
46103	46103	11	46104	00100	AT 1
46104	00100		100	00261	MOVE CONTROL WORD
46105	00101	15	00100	00103	WORD FOR MOVE
46106	00102	16	00100	00151	SET TRANSMIT
46107	00103	11	00000	00254	CURRENT WORD TO BUFFER 1
46110	00104	11	00126	00255	BUMP CONSTANT TO BUFFER 2
46111	00105	11	00127	31000	CONTROL MASK
46112	00106	51	00241	00240	ACTION MODIFICATION
46113	00107	21	00240	00132	SET ACTION
46114	00110	15	00106	00111	SHIFT CURRENT
46115	00111	55	00000	00003	CONTROL WORD
46116	00112	11	00136	31000	OP-U-V MASK TO Q
46117	00113	46	00236	00236	TO ACTION LOOP
46120	00114	11	00136	31000	OP-U-V MASK TO Q
46121	00115	21	00106	00133	BUMP CONTROL WORD
46122	00116	51	00106	00106	ADD MODULO 10000
46123	00117	45	00000	00105	WORD ON NEW CONTROL
46124	00120	55	00433	00017	RESTORE CONTROL WORD
46125	00121	13	00232	32000	MIN 1 TO A
46126	00122	47	00170	00170	CONTINUE
46127	00123		7777	07777	U-V MASK
46130	00124	15	00167	00256	CHANGE TO UΥ
46131	00125	47	00144	00144	TO MOVE
46132	00126		161	00161	BUMP CONSTANT
46133	00127			00034	CONTROL MASK
46134	00130	16	00133	00255	SET FOR U ADDRESS ONLY
46135	00131	44	00144	00144	TO MOVE
46136	00132	45	00000	00114	DUMMY JUMP
46137	00133		1		U ADVANCE AND TALLY TEST WORD
46140	00134	16	00167	00256	CHANGE TO VΥ
46141	00135	47	00144	00144	TO MOVE
46142	00136	77	07777	07777	OP-U-V MASK
46143	00137		1	00001	BUMP CONTINUE CONST
46144	00140	15	00127	00255	SET V ADVANCE ONLY
46145	00141	44	00144	00144	TO MOVE
46146	00142	17	77777	77777	ARITHMETIC
46147	00143	20	00000		CONSTANTS
46150	00144	11	00256	31000	MASK TO Q
46151	00145	51	00254	32000	U-V WORD
46152	00146	35	00255	00256	BUMP UV
46153	00147	53	00256	00254	ADD MOVE 10000 OR 40000
46154	00150	37	00166	00154	TO ARITHMETIC
46155	00151	11	00254		MOVE TO NEW LOCATION
46156	00152	51	00100	00100	ADD MODULO 10000
46157	00153	46	00101	00101	MOVE NEXT WORD
46160	00154	71	00142	00254	WORD TIMES 2 SCALED 34-1
46161	00155	72	00232	00254	ΥWD SCALED 1 EQUALS 2 SCALED 34
46162	00156	73	00143	00254	12 SCALED 34 EQUALS WORD
46163	00157	52	00254	32000	WORD IF REMAINDER EQUALS ZERO

FORM #
MSD 8520

STORAGE	EXEC.	OP	U	V	COMMENTS
46164	00160	22	00044	00254	WORD
46165	00161	22	10044	00254	WORD
46166	00162	27	31000	00254	Q ZERO IF WORD
46167	00163	21	00100	00137	BUMP MOVE CONTROL
46170	00164	12	31000	32000	ZERO TO A
46171	00165	11	00136	31000	OP U-V MASK TO Q
46172	00166	47	00167		BACK TO STORE
46173	00167	57	37777	37777	ERROR STOP
46174	00170	33	32000		1Y2 SCALED 72 2 SCALED 36
46175	00171	54	32000	00044	ALL ONES IN A RIGHT
46176	00172	31	32000		CLEAR A LEFT
46177	00173	74	32000	00320	1 TO V NEXT
46200	00174	31	32000		CLEAR A LEFT
46201	00175	36	00143	32000	A RIGHT 2 SCALED 34 SHOULD
46202	00176	43	00142	00200	EQUAL 2 SCALED 34-1
46203	00177	37	00167	00167	IF NOT ERROR
46204	00200	23	00233	00152	TEST LOCATION OF FIRST WORD
46205	00201	46	00262	00202	NEXT IF LOC IS GREATER THAN 7440
46206	00202	23	00152	00253	LOC OF FIRST WORD -1
46207	00203	46	00262	00204	NEXT IF FIRST WORD EQUALS ZERO
46210	00204	21	00415	00232	TALLY
46211	00205	42	00133	00210	IF EQUAL 4096
46212	00206	11	00416	00415	RESET TALLY
46213	00207	56	10000	00211	STOP ON 1 IF EQUAL TO 4096
46214	00210	56	20000	00211	STOP ON 2
46215	00211	11	00117		RESTORE
46216	00212	75	00040	00214	BANG ONE
46217	00213	11	00323	00323	STORAGE
46220	00214	75	30161	00216	NEXT COPY
46221	00215	11	00261	62571	TO DRUM
46222	00216	55	00133	00001	AFTER 20 TIMES
46223	00217	44	00220	00214	CONTINUE
46228	00220	23	32000	32000	ZERO ACCUMULATOR
46225	00221	75	20161	00223	FORM
46226	00222	32	00261		CHECK SUM
46227	00223	75	30161	00225	BACK FROM
46230	00224	11	62571	00261	DRUM
46231	00225	75	20161	00227	SUBTRACT
46232	00226	34	00261		DRUM COPY
46233	00227	12	32000	32000	MAGNITUDE
46234	00230	41	32000	00167	ERROR IF
46235	00231	45	00000	00262	NOT ZERO
46236	00232			00001	V ADVANCER
46237	00233	51	07440	00100	TEST CONSTANT
46240	00234			70000	TALLY
46241	00235			70000	TALLY RESET
46242	00236	51	00240	00240	ADDRESS MODULO 10000
46243	00237	11	00123	00256	UV MASK TO BUFFER 3
46244	00240				ACTION WORD
46245	00241	33	31733	15433	CONTROL WORD 1
46246	00242	73	25573	73031	CONTROL WORD 2
46247	00243	73	27733	77037	CONTROL WORD 3

Date	LOCKHEED AIRCRAFT CORPORATION MISSILE SYSTEMS DIVISION	Model	Page
Approved	Title ML MD 2	Report No.	

STORAGE	EXEC.	OP	U	V	COMMENTS
46250	00244	75	57333	33033	CONTROL WORD 4
46251	00245	31	66673	55433	CONTROL WORD 5
46252	00246	77	67573	33037	CONTROL WORD 6
46253	00247	33	32656	72433	CONTROL WORD 7
46254	00250	73	65652	57421	CONTROL WORD 8
46255	00251	76	77733	77426	CONTROL WORD 9
46256	00252	77	77777	44037	CONTROL WORD 10
46257	00253	51	00001		CONTROL WORD TEST
46260	00254				CURRENT WORD
46261	00255				BUMP
46262	00256				MASK

FORM #
MSD 8520

Date	LOCKHEED AIRCRAFT CORPORATION MISSILE SYSTEMS DIVISION	Model	Page
Approved	Title ML MD 1	Report No.	

STORAGE	EXEC.	OP	U	V	COMMENTS
40640	40640	23	32000	32000	CLEAR ACCUMULATOR
40641	40641	45	00000	40617	GO TO REPEAT SET UP

FORM #
MSD 8520

USEful Note # 8 (Revised)

Talmadge

REMINGTON RAND UNIVAC

ST. PAUL DEPARTMENT--INFORMATION SCIENCE

18 December 1956
(Rev. 18 Feb. 1957)

1103 TO 1103A CONVERSION ROUTINE

I. TYPE: Service routine or subroutine.

II. STATUS: Code checked and machine checked by Bill Wallace.

III. PURPOSE: This routine changes A and Q machine addresses from 20000 and 10000 to 32000 and 31000 respectively, and detects magnetic tape and external function instructions. Various options are provided for print out of those addresses where an A or Q reference is modified, (indicating also u or v portion) and punching the converted program in bioctal or flex code on paper tape.

IV. USAGE:

A. STORAGE REQUIRED: The program is coded in RECO form and it is therefore possible to operate the program from a location providing 320 octal drum address and 2000 additional octal drum addresses for a HSS image region. Such a location of the program and image region is done by assigning the desired starting addresses to regions BB and IR respectively, of the reco tape (see coding) all other regions being in HSS, and hence remaining the same. The regional assignment can be on a separate tape from the main program reco tape, but this tape should have END. c.r. at the end. (See RECO write-up.)

In addition to the RECO tapes, a bioctal tape of the program is available where the program is stored at 66000-66320, with the image region 76000-77777.

B. INPUT-OUTPUT: Output is a punched tape in bioctal or flex code of the changed program if desired. This option makes use of tape output routines which are not integral parts of this conversion routine. These output routines (Bioctal or Flex Dumps) are included in the Serial 9 1103 Service Library or may be found in the Minimum Service Routine Library for 1103A (See USEful Note #1). Also the following is printed out as the conversion routine is operating: (This is also optional.) u aaaaa or vaaaaa, where aaaaa is the address where an A or Q reference has been modified and u or v shows whether the u or v address of the instruction has been modified. Also, TAPE is printed out when an 1103 magnetic tape instruction is encountered, and EF and address when an external function command occurs.

C. OPERATING INSTRUCTIONS:

(1) Used as a service routine proceed as follows: (the term "program" here refers to a program to be converted.)

a) Master clear, MD start
b) Set PAK to 66000, (or bb)

c) Insert in Q_u the first address of the program
d) Insert in Q_v the last address of the program
e) Insert in v address of A_R the address of the last instruction of the program, or the last address of the program wherein one wishes to have A and Q addresses modified.
f) Insert in v address of A_L the following codes for the various options:

00000	bioctal punch of converted program and print out of addresses where modification occurs.
00001	same as above but no print out.
00002	flex code punch of converted program and print out.
00003	same as above, but no print out.
00004	print out, but no punch of converted program.
00005	no print out and no punch of converted program.

A 56 0 66010 (bb10) stop occurs if a gross error is made in the set-up, e.g. transposition of Q_u and Q_v.

(2) Used as a subroutine, proceed as follows:

a) Program the transfer of parameter as listed above to the A and Q registers.
b) Execute the instruction:
RJ bb2 bb
c) The options are selected in the same manner as previously shown.

(3)

a) The use of this conversion routine assumes that the program to be converted is stored either all in core storage or all in drum storage.
b) The conversion routine is coded for operation on either an 1103A, or on the 1103 (Serial 9) at RRU, St. Paul. If this conversion routine is to be used on an 1103A, provision must be made for the output routine mentioned in IV-B above.

V. CODING

A. Regions

```
re bb66000        re ua124
re ir76000        re va137
re ff30000        re up155
re cr0            re pr162
re cb33           re tp173
re cd54           re ef200
re cf71           re cs205
re kk75           re dd212
re mm112          re tt310
```

B. Program

bb0		45	0	bb10	Entrance
1		56	0	bb10	Error stop
2		45	0	(ff)	Subroutine exit
3		0	0	0	Storage first address
4		0	0	0	Storage last address
5		0	0	0	Storage initial A_r
6		0	0	0	Storage initial Q_r
7		45	0	ff	Constant
10		tp	Q	bb6	
11		lt	10000	bb5	
12		lt	00000	A	
13		tp	0	ir	
14		tp	bb	0	
15		rp	31777	bb17	
16		tp	1	ir1	Store HSS
17		rp	30400	cb	To start of core program
20		tp	bb1	cr1	Conclusion of program
21		rp	31777	bb23	
22		tp	ir1	1	Restore HSS
23		tp	ir	0	
24		tp	bb6	Q	Restore Q for dump
25		tp	bb7	A	
26		ej	bb2	bb31	Test, subr. or serv?
27		rj	70036	(70006)	No, subr.
30		45	0	bb2	To exit
31		rj	70036	(70006)	
32		56	00000	bb	
33	cb0	ej	dd43	cd	No punch
34	1	ej	dd64	cd2	Punch flex
35	2	ej	dd65	cd5	No print, no punch
36	3	ej	dd66	cd13	No print, punch flex
37	4	ej	dd60	cd6	No print, punch bioctal
40	5	tp	cr5	A	Last address

41	6	tj	dd	cf	HSS?
42	7	qt	ddl	cr4	Store first address
43	10	lq	q	25	
44	11	qt	ddl	cr3	Store last address
45	12	tv	cr3	mm	Set up transfer
46	13	la	cr3	al7	of Modified Contents
47	14	tu	A	kk	Set up first address
50	15	ra	cr4	dd60	to be modified
51	16	st	cr3	tt	No. of words
52	17	ij	tt	kk	
53	20	45	0	bbl	Error
54	cd0	tv	dd67	mm4	
55	1	45	0	cb5	No punch
56	2	rs	bb27	dd60	
57	3	rs	bb31	dd60	
60	4	45	0	cb5	Flex punch
61	5	rj	cdl	cd	
62	6	tv	up4	ual0	
63	7	tv	up4	ual2	
64	10	tv	val5	va7	
65	11	tp	val5	vall	No print
66	12	45	0	cb5	
67	13	rj	cd4	cd2	
70	14	45	0	cd6	
71	cf0	ra	cr5	dd3	Add 76000 to V
72	1	ra	Q	dd2	Add 76000 to U and V
73	2	tp	cs	pr	Arrange to print core address
74	3	45	0	cb7	
75	kk0	tp	(ff)	Q	
76	1	tp	Q	ttl	
77	2	qt	dd5	tt3	Mask operation code
100	3	tp	tt3	A	
101	4	ej	ddl0	ef	External function
102	5	ej	ddll	mm	Final stop
103	6	ej	ddl2	mm	Interpret
104	7	rp	20014	kkll	Commands where
105	10	ej	ddl3	tp	V only to be modified
					_ and tape commands
106	11	rp	20004	kkl3	Split instruction,
107	12	ej	dd27	mml0	Modify U only
110	13	rj	ua6	ua	Modify U
111	14	rj	va5	va	Modify V
112	mm0	tp	ttl	(ttl)	Transfer modified
					Content
113	1	tp	kk	A	
114	2	lt	25	A	Obtain current
115	3	st	dd6	A	Address
116	4	ej	cr5	bb21	Test, end of
117	5	ra	kk	dd7	Modifiable address

```
120   6     ra   mm     dd60    Add 1
121   7     45   0      cb17
122   10    rj   ua6    ua      Modify U only
123   11    45   0      mm

124   ua0   tp   tt1    Q
125   1     lq   Q      25
126   2     qt   dd33   tt4     Mask 1st octal digit
127   3     tp   tt4    A
130   4     ej   dd34   ua7     Q?
131   5     ej   dd35   ua11    A?
132   6     45   0      ff
133   7     ra   tt1    dd36    Add 21000
134   10    45   0      up      To print
135   11    ra   tt1    dd37    Add 12000
136   12    45   0      up

137   va0   tp   tt1    Q
140   1     qt   dd33   tt4
141   2     tp   tt4    A
142   3     ej   dd34   va6     Q?
143   4     ej   dd35   va10    A?
144   5     45   0      ff
145   6     ra   tt1    dd40    Add 21000
146   7     45   0      va11    To print
147   10    ra   tt1    dd41    Add 12000
150   11    pr   0      dd42    Carriage return
151   12    pr   0      dd43    Space
152   13    pr   0      dd44    "V"
153   14    rj   pr10   pr
154   15    45   0      va5

155   up0   pr   0      dd42    Carriage return
156   1     pr   0      dd45    "U"
157   2     pr   0      dd43    Space
160   3     rj   pr10   pr
161   4     45   0      ua6

162   pr0   tp   kk     Q
163   1     lq   Q      6
164   2     tp   dd43   tt2     Index
165   3     lq   Q      3
166   4     qt   dd46   A
167   5     at   dd47   pr6     Print digit
170   6     (pr  0      ff)
171   7     ij   tt2    pr3
172   10    45   0      ff

173   tp0   rp   20004  kk14    Test for tape
174   1     ej   dd20   tp2       Instructions
```

175	2	pr	0	dd42	Carriage return
176	3	rp	10004	mm	
177	4	pr	0	dd60	Print "tape"
200	ef0	pr	0	dd42	Carriage return
201	1	rp	10005	ef3	
202	2	pr	0	dd70	Print "EF"
203	3	rj	pr10	pr	Print address
204	4	45	0	kk14	To V address modification
205	cs0	rj	cs4	cs1	
206	1	tu	kk	tt5	
207	2	rs	tt5	dd4	Subtract 76000 from V
210	3	tp	tt5	Q	
211	4	45	0	ff	
212	dd0	0	0	02000	
213	1	0	0	77777	
214	2	0	76000	76000	
215	3	0	0	76000	
216	4	0	76000	0	
217	5	77	0	0	
220	6	0	11	0	
221	7	0	1	0	
222	10	ef	0	0	
223	11	fs	0	0	
224	12	ip	0	0	
225	13	lt	0	0	
226	14	45	0	0	
227	15	56	0	0	
230	16	pr	0	0	
231	17	pu	0	0	
232	20	rm	0	0	
233	21	wm	0	0	
234	22	am	0	0	
235	23	bm	0	0	
236	24	rp	0	0	
237	25	er	0	0	
240	26	ew	0	0	
241	27	sp	0	0	
242	30	sa	0	0	
243	31	sn	0	0	
244	32	ss	0	0	
245	33	0	0	70000	
246	34	0	0	10000	
247	35	0	0	20000	
250	36	0	21000	0	
251	37	0	12000	0	
252	40	0	0	21000	

253	41	0	0	12000	
254	42	0	0	45	Carriage return
255	43	0	0	4	Space
256	44	0	0	17	V
257	45	0	0	34	U
260	46	0	0	7	
261	47	61	0	dd50	
262	50	0	0	37	Flex code 0
263	51	0	0	52	1
264	52	0	0	74	2
265	53	0	0	70	3
266	54	0	0	64	4
267	55	0	0	62	5
270	56	0	0	66	6
271	57	0	0	72	7
272	60	0	0	1	flex code t
273	61	0	0	30	flex code a
274	62	0	0	15	flex code p
275	63	0	0	20	flex code e
276	64	0	0	2	
277	65	0	0	5	
300	66	0	0	3	
301	67	0	0	bb32	
302	70	0	0	47	Shift up
303	71	0	0	20	"E"
304	72	0	0	26	"F"
305	73	0	0	57	Shift down
306	74	0	0	4	
307	75	0	0	0	Not used
310	tt0	0	0	0	Temporary storage
311	1	0	0	0	
312	2	0	0	0	
313	3	0	0	0	
314	4	0	0	0	
315	5	0	0	0	

Talmadge

DIGITAL COMPUTER BRANCH
SIMULATION AND COMPUTATION DIVISION
Holloman Air Development Center
Holloman Air Force Base, New Mexico

RULES FOR SYMBOLIC CODING

For the KRA-1103 A Computer

For programs to be translated by

TRANS-USE-1

Prepared by

ROBERT G. TANTZEN

14 AUGUST 1956

Revised 11 December 1956

A. Introduction.

The 1103A computer works internally in the binary number system. Therefore, all information (numbers and instructions) must eventually have a binary machine representation. This is usually given in octal notation, which can be considered a shorthand writing of binary numbers. Writing programs in this absolute fashion is rather tedious work. Besides taking a long time, it is liable to coding and clerical errors. It is possible to let the computer do part of the tedious coding work. A step in this direction is the regional coding scheme. Another step forward is called symbolic coding, a system which has been adopted as the standard coding procedure by USE.

Symbolic coding replaces absolute numbers by names or symbols. As these names can be chosen to suggest certain ideas, a program in symbolic form is much more easily written and interpreted than the same program in absolute numbers. For example suppose we have to compute

$$y=(kx^2+ mx + n) \cdot \frac{1}{P}$$

Using suggestive symbols for addresses and nemonic codes for the operations, the symbolic program would be:

```
START  MP  K  TEMP
       AT  M  A
       MP  A  TEMP
       SA  N  0
       DV  P  TEMP + 1
```

Trans-USE-1 will accept this symbolic code, assign correct numbers for the nemonic commands, addresses for the other symbols, and thereby translate it into its absolute form which, e.g., could be:

```
00200    710030001000
00201    350033332000
```

00202   713200001000

00203   320033400000

00204   730035001001

The rules set up for using this symbolic coding to be translated by Trans-USE-1 are explained in detail in the rest of the paper.

B. Symbol

A symbol consists of from 1 to 6 characters, at least one of which must be alphabetic. The characters permitted are all letters A-Z and all numbers 0-9. The following symbols have special meanings and cannot be used in the location column.

1. The symbol "A" represents the octal number 32000 and refers to the accumulator.
2. The symbol "Q" represents the octal number 31000 and refers to the quotient register.
3. The symbol "D" represents the octal number 40000 and is used for references to the drum.
4. The symbol "FILL" in the operation or either address field indicates that that portion of the command is to be supplied by the program. In the u or v address fields, "FILL" will be replaced by the illegal address 30000. When used as an operation symbol, "FILL" will be replaced by 00.

C. Location

The location column of a line of coding may be blank or may consist of a single symbol. These symbols are used for reference purpose only and do not, in any way, determine the order in which the instructions finally appear in the memory. The sequence in which the instructions are written determines the order in which they are to be executed. The special symbols A, D, Q, and FILL may not be used as location symbols.

D. Operation

1. The two-character nemonic abbreviations as prepared by Remington Rand are used for basic machine operation codes. Thus TP stands for 11, AT for 35, etc. In instructions requiring a "j" as part of the u-address the number "j" ($0 \leq j \leq 7$) is appended to the nemonic operation abbreviation as a third character; so it will appear in the operation field rather than in the u-field. If j = 0, it may be omitted. To make coding easier, the number j may be replaced by a more suggestive character, e. g., ERB means ER1 indicating a reading from 10B. The complete list of these operation symbols is stated below:

LT= LTO = LTL,	LT1 = LTR
PU = PUO = PU6,	PU1 = PU7 (7 for 7th level)
ER = ERO = ERA	RP 1 = RPV
ER1 =ERB	RP2 = RPU
EW = EWO = EWA	RP3 = RPB (B for both)
EW1 = EWB	NE = NEO

2. Trans-USE-1 will also accept 2 or 3 digit octal operation codes, whether or not these represent legal operations.

3. The operation symbols END, BREAK, RESERV, refer to pseudo-operations, see under G.

4. The operation symbols "B" and "X" are used to write absolute constants, see under F.

E. Addresses

1. Reference to a line of coding is made by putting in the u- or v-field the location symbol of that line to be referenced.

2. The programmer might wish to write an address not as a symbol but as an absolute number. Also, several instructions require a number in one of the addresses, e. g., SP u k or RP jnv. In these cases the coder will write a number in the address field. Trans-USE-1 will

consider numbers to be decimal, unless the letter B is attached at the end, in which case they are considered octal. For example, 16 is the decimal number 16 and 32000 B is the octal number 32000 or the address of the accumulator, normally written as "A".

3. In certain cases it is convenient to express an address as a combination of location symbols and numbers. It is legal to specify the sum or the difference of any two.

A few examples might be helpful. LOOP2 and LOOP3 are different symbols, their addresses are not related to each other. LOOP2 + 5 designates the 5th line of coding after LOOP2. LOOP2 - 3 is accordingly the 3rd line preceding LOOP2. LOOP2 - COMP will be translated into the difference of the absolute addresses assigned to the two symbols.

F. Constants.

Constants and numbers are always full words. Therefore, there is no separation of a word into operation, u- and v-fields. The machine internally considers all numbers to be integers. USE has established conventions for writing octal anddecimal integers. On the coding sheet the letter "X" or "B" has to be written in the operation column to specify decimal or octal integers respectively. The u- and v-fields are considered as a unit, into which the desired number is entered. The number will be considered as a constant only if a B or X appears in the operation column. A sign may precede the digits. If omitted the constant is considered positive.

A few examples and their absolute octal equivalents follow:

```
X          100
00 00000 00144

B          100
00 00000 00100
```

```
X           - 5
77 77777 77772

B        2 00001
00 00002 00001
```

Incidentally, the last constant could also be written by leaving the operation blank, write 2 in the u-field and 1 in the v-field. See under E2 and D2.

G. Pseudo-Instructions

These are instructions to the translation routine and do not form part of the translated program.

1. "BREAK" (BREAK ---) Symbolic programs may be longer than 500 lines of coding, which is the capacity of one Unityper 200 ft. tape. Programs up to this size are typed on one single magnetic tape. Longer programs can be handled in segments of approximately 500 lines each. When reaching the end of a tape, the typist simply types a line of coding consisting of the word "BREAK" in the operation column, leaving all other columns blank (see example under L1). When encountering the BREAK operation, the computer will stop and wait for the next reel to be put on the same tape unit. Pushing the start button on the console will then cause the translation to continue. When reading from cards, BREAK may be used if desired.

2. "RESERV": (RESERV N N) This operation reserves, in memory, a block of N temporary cells. The symbol appearing in the location column will be assigned the current value of the address counter. This counter will then be incremented by the amount specified in the v-field. To make the program acceptable to the USE-compiler, both of the u and the v-field must contain the number of cells to be reserved. The coding lines containing a RESERV operation will occupy one cell each in the translated program. Therefore, they have to be at the end of a program directly preceding the "END" line. The end of a program will

then look something like this:

LOC	OP	u	v	remarks
C16	0	1	1	address, modify
C17	0	24B	0	20 in u
TEMP1				sin t
IMAGE	RESERV	36	36	card image
BLOCK	RESERV	120	120	tape information

3. "END": (END--* The pseudo-instruction END is on the last line of coding. There must not be a symbol in the location column, the u- and v-fields are ignored. "END" signals the end of the program to be translated.

H. Coding Sheet.

The coding sheet is divided into 5 columns. The maximum permissable number of characters for each column is as follows:

Location	6
Operation	6
u-address	13
v-address	13
Remarks	18

I. General Rules of Coding

1. Any symbol may appear only once in the location column of a program. If it appears more often, the first (lowest) address will be assigned to that symbol.

2. References to subroutines and service routines are made by using an established symbol for each. If, for instance, a subroutine symbol is MATRIX, which could mean a matrix-multiply routine, the jump to the subroutine would be

RJ MATRIX + 2 MATRIX.

Where to supply operands and to acquire results may be found in a table of subroutines, which will be available at each installation.

See also USE-standards for subroutines.

3. The normal use of the repeat instruction (RP jn w) requires a jump command in cell 00000 (the w is placed in the v portion). The programmer should make sure that it is there.

4. Free symbols: Usually the programmer does not care what addresses are being assigned to symbols of temporaries. He then does not write them in a location column. These so-called free symbols will be assigned successive addresses at the end of the program in the order in which they appear on the coding sheet.

5. Symbols may consist of an arbitrary mixture of letters and numbers. Some characters therefore, may cause some uncertainty as to which one is meant. The character "0" may be the letter "O" or the number zero. To avoid confusion it is good policy to choose only symbols which imply an unambiguous meaning. A "0" standing alone is always a zero. Symbols like CO or P03 should be avoided. A general rule to go by is that 0 is always a zero unless included between letters. In handwriting care should be taken to distinguish between 2 and Z, 1 and I, 5 and S.

6. If all or part of an instruction will be computed and supplied by the program, the symbol FILL should be written in the corresponding columns. This is a safety device. If the program does not provide the information for some reason or other, the computer will stop with an MCT or SCC fault indication. See also under B4.

7. Never assume any temporary cell to initially contain certain information. Sometimes the whole memory will be cleared before reading-in a program. Then the temporaries would be zero. However, it is safer not to take this for granted.

J. Arrangement of a Program

Programs to be translated and assembled consist of three sections which must be written in the following order.

1. Section 1, n_1 words, containing all words subject to address modification, usually commands and relative constants.

2. Section 2, n_2 words, containing all absolute constants. This section also includes those temporaries which are assigned locations by placing their symbols in a location column.

3. Section 3, n_3 words, consisting of the number of temporaries used, not including those already accounted for in section 2.

The total memory space a program needs is $n_1 + n_2 + n_3$ cells. All information constituting a program is contained in sections 1 and 2. Therefore, the translation extends over $n_1 + n_2$ words only.

K. Parameters

For checking and control purposes each program must be preceded by 6 lines of coding containing the following parameters:

1. ITA: Initial translation address which shall be assigned by the translation to the first line of the body of the program.

2. n_1: The number of words subject to address modification (see under J).

3. n_2: The number of absolute constants and, possibly, certain temporaries (see under J).

4. n_3: The number of temporary locations used (see under J).

5. p: Is used only in subroutines, and is the number of operands or control data needed by a subroutine.

6. r: Is used only in subroutines, and is the number of results yielded by a subroutine.

In programs other than subroutines the values of p and r are irrelevant, but they must not be omitted. In addition to the above 6 parameters, a 6 character identification tag for the program, the first 2 characters identifying the installation (for Holloman, HO), must appear in the comments field of the first line of coding. The above mentioned parameters may be given as constants using either the B or X operation. So the first 6 lines of coding will look like this:

LOC	OP	u	v	Remarks
	B or X	ITA		Identification
	B or X	n_1		Title of Program
	B or X	n_2		Title of Program
	B or X	n_3		Other information,
	B or X	p		as: date, coder,
	B or X	r		project number, etc.

L. Typing a symbolic program

For input to the computer a symbolic program must either be typed on magnetic tape with the Unityper, or key-punched on IBM cards.

1. Magnetic tape.

After inserting a reel of tape into the Unityper and loading the leader, the program is typed starting with the first line of coding (containing ITA). The fields are separated by commas, even if they contain no information. Spaces may be typed anywhere; they are ignored by the translation program except in the remarks column. After a line of coding has been typed the rest of the blockette is filled with zeros (tabulator key). An inter-blockette space is then generated with the trip key. Each coding line will thus occupy one line on the Unityper paper and correspondingly one blockette on tape.

To give an idea of how a program looks on paper, here are a few examples:

```
TRANS, TP, FILL, BT, COMMAND TO BT
MODIF, RA, TRANS, C18, MODIFY + 20
,TP, BT + 1, A, OPERATION TO A
,D, 31415926, PI SCALED + 7
,BREAK,,,
```

Note that each line normally has 4 commas. Only for constants (B and X operation symbols) are there 3, because the u and v fields are considered as a unit.

A program might be too long for one reel of tape. Then the typist must type the BREAK instruction on the end of that tape before continuing on a new one. See under G1.

2. IBM - Cards.

The five fields of the coding sheet are assigned certain columns on the cards. The rest of the card is used for information generated by Trans-USE-1; in this way cards punched out by the translation program, containing both the symbolic and absolute program for side by side listing, may be reused for input. The field allocation is as follows:

Card Columns	1 - 6	location
	7 - 12	operation
	13 - 25	u-address, constants
	26 - 38	v-address,
	39 - 56	remarks
	57 - 60	absolute octal address
	61 - 72	absolute octal word
	73 - 75	decimal sequence number
	76	insert digit
	77 - 80	identification tag, last 4 characters.

Besides punching the information from the coding sheet, the key punch operator has to punch sequence numbers. These (positive) numbers must be in ascending order. This enables the translation routine to check on the correct sequence of the cards read. The sequence numbers on the first card may be any, including zero. The sequence check may be bypassed by setting a switch on the console of the computer.

M. How to make Changes in a Program

1. Corrections: Replace the incorrect cards by correct ones.

2. Deletions: Simply take out the unwanted cards.

3. Insertions: New cards may be inserted any place in the program. They should contain the sequence number of the card after which they are inserted and also an insert digit, which will be regarded in the sequence test as a decimal fraction added to the integral sequence number.

After each change the program must be retranslated. Care should be taken to correct n_1 and n_2 when making changes.

N. Size limit of a program

Trans-USE-1 has been coded for the "minimum 1103-A", which contains 4096 words of core memory. There is, however, no fixed limit on the size of a symbolic program, as far as the translation goes. The only restrictions is the number of different symbols, which cannot exceed 245. This number does not include the special fixed symbols and the subroutine symbols. For the assembly the size of a program is directly determined by the core capacity. The following two conditions must both be satisfied:

1) The assembled program must not go beyond address 07200. For a program without subroutines the last line of coding should not go beyond that point. If subroutines are used, the last word of the last subroutine must stay under this limit.

2) The temporary region used for execution of the program should not exceed core capacity (address 07777).

As the lowest initial assembly address IAA is 00014, a complete program can have up to 3700 words, and 4083 words including the temporaries. Larger programs can be handled, if they can be broken up into independent sections.

INDEX

CARD CHARACTERS USED WITH 1103-A AND 407.

CHARACTER	CARD HOLES
Blank	None
+	12
-	11
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
A	12 1
B	12 2
C	12 3
D	12 4
E	12 5
F	12 6
G	12 7
H	12 8
I	12 9
J	11 1
K	11 2
L	11 3
M	11 4
N	11 5
O	11 6
P	11 7
Q	11 8
R	11 9
/	0 1
S	0 2
T	0 3
U	0 4
V	0 5
W	0 6
X	0 7
Y	0 8
Z	0 9

CHARACTER	CARD HOLES
Space	None
= (equals)	8 3
? (question)	8 4
. (period)	12 8 3
) (Par. close)	12 8 4
° (degree)	11 8 3
| (absolute)	11 8 4
, (comma)	0 8 3
((Par. open)	0 8 4

OPERATING INSTRUCTIONS

FOR

TRANS-USE-1

AN 1103-A SERVICE ROUTINE

DATE: 1 Dec 56
Prepared by: Robert G. Tantzen
Issued by: Holloman Air Development Center
Computers Division

A. Normal Operating Procedure.

1. Make sure Trans-Use-1 is loaded (on 60,000 drum).

2. Make sure the subroutine library is loaded (if the programs to be translated refer to same).

3. Put symbolic program tape on TU1 (tape unit logical1) or put symbolic cards in reader, do not cycle Bull. Place blank cards in punch. Set Bull for 3 fields.

4. Put tapes on TU2, TU3, and TU4. TU3 and TU4 must be rewound.

5. If input from tape, turn MJ1 on.

6. Optional: If check on sequence numbers from input cards shall be bypassed, turn MJ2 on. (Applies for card input only).

7. If assembly is to be indirect, turn MJ3 on.

8. Set PAK=60,000 and start.

Stops:	PAK=60,121	Alarm print. Look at typewriter print-out. Refer to list of alarm prints below and decide what to do.
	PAK=60,002	Typewriter has printed a "t" indicating the end of current translation. If another translation is wanted, go back to 5. If assembly is wanted combine with 9.
	PAK=04412	MJ1 on: Tape with symbolic program is used up, a "BREAK" is found. Put next symbolic tape on TU1 and start.
		MJ1 off: A "BREAK" card is read. Put more imput cards in reader, cycle Bull once and start.

REMARKS: Should Bull mispick a read card, start all over. Should Bull mispick a punch card, resume computation at 04660. All cards except the first 6 will be repunched. This procedure good only if less than 120 cards have been punched so far.

9. Recheck choice for direct or indirect assembly. If change is wanted, set MJ3 now, and reset PAK to 60002.

10. Direct run: Start
Indirect run: Put "Assembly Control Card" in reader, do not cycle Bull. Ascertain that TU4 is rewound, and start.

Stops: PAK=60121 Alarm print. Look at typewriter print-out. Refer to list of alarm prints below and decide what to do. Alarm print with tagword <u>"assemb" Indicates successful end of assembly.</u> To execute first assembled program, start. Programs 2, 3, 4, of this assembly can be started at 00011, 00012, 00013 respectively. If another assembly is wanted, combine with 11.

11. Put tape with translation(s) on TU3, another tape on TU4, then go to 9.

B. <u>List of Alarm Prints</u>.

Each alarm print consists of a 6-character tagword followed by the contents of A and Q. The computer stops with PAK=60121. To resume computation, start.
An explanation of the alarm prints follows, specified by the different tagwords printed.

"seq no" Occurs on card input only.
Fault: The last card read has a sequence number not larger than the previous one. A and Q hold the new and old sequence numbers in excess 3.

Remedy 1: If proper card sequence does not matter, start.
Remedy 2: Get deck of cards in proper order and go back to 3 and rewind TU3. If previous translations are already recorded on TU3, a new tape should be taken.

"parity" A=1. Occurs during translation only.
Fault? A parity error is found while reading TU2 or TU1.
Remedy: Determine from console which tape unit failed. Try another tape or another tape unit.

"length" Fault: The length of the program being translated does not agree with the sum of the parameters N_1 & N_2.
A contains the difference between actual length and $N_1 + N_2$, Q contains the actual length, both in octal.
Remedy: Start correct parameters or program later.

"limit" Fault: The symbolic program contains more than 245 different location symbols. The 245th symbol is in cell 06525, it may be found also from the last cards read.
Procedure: Go off the computer. Reduce number of symbols in program.

"trm tp" A=1.
Fault: A parity error is found while reading translated program from TU3 during assembly.
Remedy: Repeat assembly with another bias setting or with tape on another tape unit.

"ass tp" A=1

Fault: A parity error is found while reading back the assembled program from TU4 for verifying.
Remedy: Try another bias, another tape, or another tape unit for TU4. If MJ3 is off, start. Assembly is repeated. If MJ3 is on, go to 11.

A=5

Fault: Assembled program was recorded incorrectly on TU4.
Remedy: Start. A new recording will be attempted. Do not change tape or tape unit. Change of bias setting is permissable.

"excess" Fault: The program to be assembled, including subroutines, exceeds core memory. There are two cases:

A=7200: The last word of the complete program will have an address greater than 07200.

A=7777 The temporary region required by the program exceeds core capacity.

Remedy: Repeat translation or assembly with lower ITA or IAA. Do not continueoperation, because assembly routine will be clobbered.

C. Format of "assembly control card"

Only field I is used, fields II and III are ingored. Field I is again divided into 6 smaller fields, 6 columns each, which must contain the following information:

Field	Columns	Contents
1	1-6	new name for program to be assembled. (program tag)
2	7-12	initial assembly address (IAA) in octal, up to 5 octal digits. Address may be anywhere in the field.
3	13-18	name of first program to be assembled.
4	19-24	names of other programs to be assembled together with the first one. Must be left blank if not used.
5	25-30	
6	31-36	

D. Incorporation of subroutines.

All subroutines to be used by a symbolic program must be in standard USE-format, i.e.,

a. They must be stored on the drum.
b. They must have the 5 parameters, N_1, N_2, N_3, P, R, stored directly in front of themselves.
c. They must be coded relative to 01000.
d. The normal exit must be two lines later than the entry line.
e. The subroutine must be self-contained.

In the symbolic program a subroutine is referenced by its name or symbol. The translation will replace it by an octal symbol, a number between 30400 and 30777. A subroutine referencing another one will do so by using this octal symbol.

For each subroutine we have now 3 items:

1. Name or symbol, to be stored in DS-region
2. Octal symbol, to be stored in DA-region
3. Drum location, to be stored in DL-region

How to put these into the Trans-Use-1 routine may be best explained by an example. Assume we have the following 4 subroutines.

Name	Drum Location	Highest Reference	Cells to be reserved for octal symbol
SS1	40200	SS1 + 3	4
SS2	43000	SS2 + 11	12
SS3	40500	SS3 + 2	3
SS4	40600	SS4 + 4	5

The number of cells in the last column is determined by the number of controls of the subroutine. It must be at least 3, because the normal exit is the 3rd line always. The regions DS, DL and DL have the drum addresses 61756, 62045, and 60210 respectively. The cells to be filled for our example are therefore:

Address	Contents	remarks
61756	00 0000656504	name of 1st subroutine, SS1
61757	00 0000656505	SS2
61760	00 0000656506	SS3 (DS)
61761	00 0000656507	SS4
62045	00 0000030400	Octal symbol of SS1
62046	00 0000030404	" " SS2
62047	00 0000030420	" " SS3 (DA)
62050	00 0000030423	" " SS4
62051	00 0000030430	Octal symbol of future subr.
60210	00 0000040200	Locations of subroutines in library (DL)
60211	00 0000043000	
60212	00 0000040500	
60213	00 0000040600	

Note that the next free octal symbol must be filled in the DA-region. Up to 49 subroutines can be handled.

10 Dec 56 R. Tanlyn

Talmadge

USEful Note # 9 | 28 January 1957

SUBJECT: APL Floating Point to Stated Point Conversion Subroutine

This subroutine converts a decimal floating point number in excess 3 form in two 1103A words to a stated point number in excess 3 form in two 1103A words.

The format of the decimal floating point number is as follows:

First 1103A word	$\underline{\triangle}\ M_1\ M_2\ M_3\ M_4\ M_5$
Second 1103A word	$M_6\ M_7\ M_8\ \pm\ C_1\ C_2$

where

a. the sign of the mantissa is represented by the symbol △(octal 01) if positive,
b. the sign of the mantissa is represented by the symbol –(octal 02) if negative,
c. the eight digit mantissa is equal to or greater than 0.1 but less then 1.0,
d. the sign of the characteristic is represented by the symbol +(octal 63) if positive, and
e. the sign of the characteristic is represented by the symbol –(octal 02) if negative.

This subroutine will convert to stated point numbers only those floating point numbers in which $c_1 = 0$. If $c_1 \neq 0$, the floating point number is transferred to the output without conversion. When conversion occurs the format of the stated point number is as follows:

Decimal Characteristic	First 1103A Word	Second 1103A Word
-9	$\underline{\triangle}$ .0 0 0 0	0 0 0 0 0 $\overline{M_1}$
-8	$\underline{\triangle}$ .0 0 0 0	0 0 0 0 M_1 $\overline{M_2}$
-7	$\underline{\triangle}$ .0 0 0 0	0 0 0 M_1 M_2 $\overline{M_3}$
-6	$\underline{\triangle}$ .0 0 0 0	0 0 M_1 M_2 M_3 $\overline{M_4}$
-5	$\underline{\triangle}$ .0 0 0 0	0 M_1 M_2 M_3 M_4 $\overline{M_5}$
-4	$\underline{\triangle}$ .0 0 0 0	M_1 M_2 M_3 M_4 M_5 $\overline{M_6}$
-3	$\underline{\triangle}$ .0 0 0 M_1	M_2 M_3 M_4 M_5 M_6 $\overline{M_7}$
-2	$\underline{\triangle}$ .0 0 M_1 M_2	M_3 M_4 M_5 M_6 M_7 M_8
-1	$\underline{\triangle}$ .0 M_1 M_2 M_3	M_4 M_5 M_6 M_7 M_8 0
0	$\underline{\triangle}$ .M_1 M_2 M_3 M_4	M_5 M_6 M_7 M_8 0 0

Decimal Characteristic	First 1103A Word	Second 1103A Word
+1	$\triangle\ M_1 \cdot M_2\ M_3\ M_4$	$M_5\ M_6\ M_7\ M_8\ 0\ 0$
+2	$\triangle\ M_1\ M_2 \cdot M_3\ M_4$	$M_5\ M_6\ M_7\ M_8\ 0\ 0$
+3	$\triangle\ M_1\ M_2\ M_3 \cdot M_4$	$M_5\ M_6\ M_7\ M_8\ 0\ 0$
+4	$\triangle\ M_1\ M_2\ M_3\ M_4 \cdot$	$M_5\ M_6\ M_7\ M_8\ 0\ 0$
+5	$\triangle\ M_1\ M_2\ M_3\ M_4\ M_5$	$\cdot M_6\ M_7\ M_8\ 0\ 0$
+6	$\triangle\ M_1\ M_2\ M_3\ M_4\ M_5$	$M_6 \cdot M_7\ M_8\ 0\ 0$
+7	$\triangle\ M_1\ M_2\ M_3\ M_4\ M_5$	$M_6\ M_7 \cdot M_8\ 0\ 0$
+8	$\triangle\ M_1\ M_2\ M_3\ M_4\ M_5$	$M_6\ M_7\ M_8 \cdot 0\ 0$
+9	$\triangle\ M_1\ M_2\ M_3\ M_4\ M_5$	$M_6\ M_7\ M_8\ 0 \cdot 0$

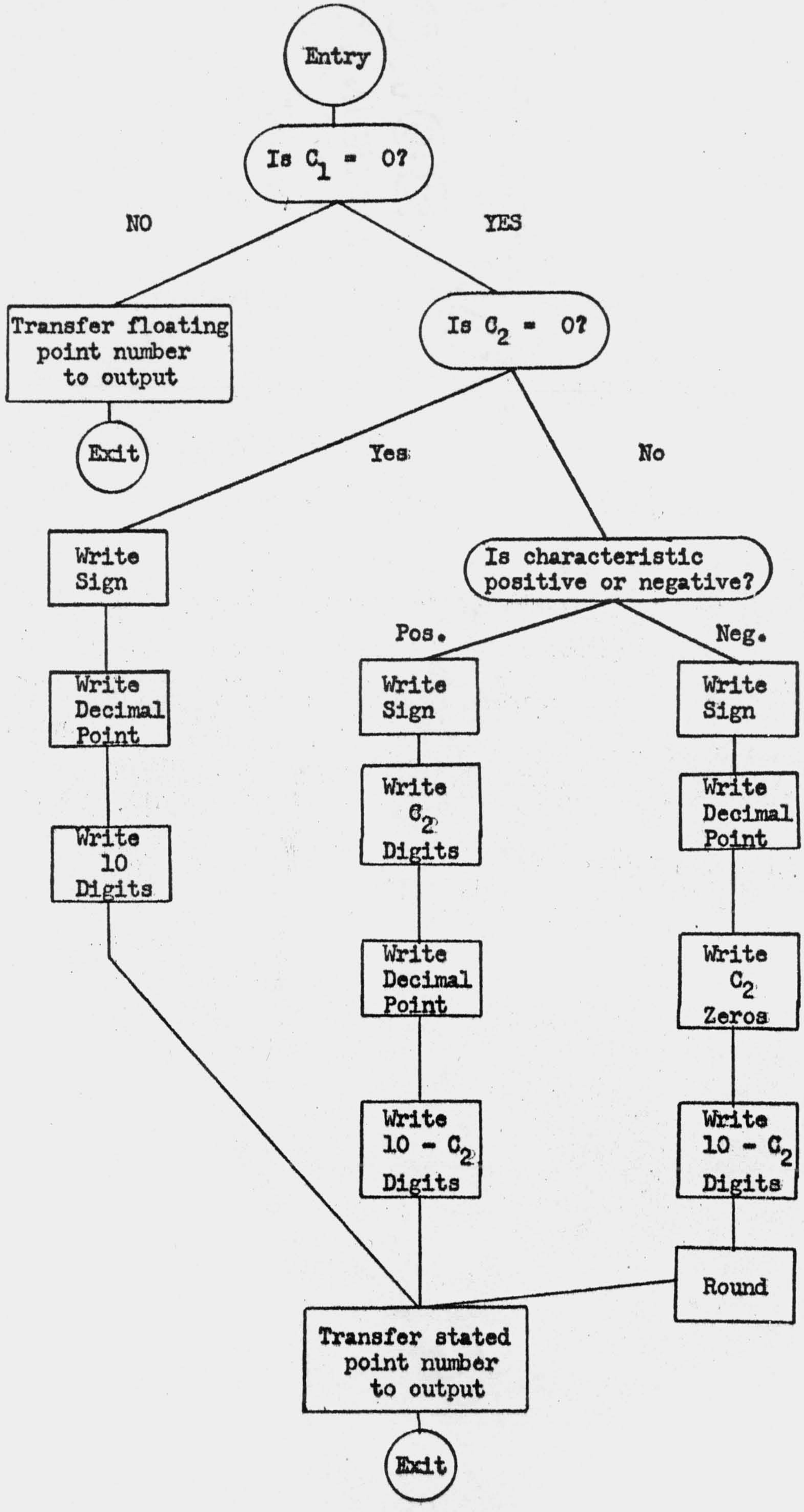
where

a. the sign of the number is represented by the symbol △ (octal 01) if positive,

b. the sign of the number is represented by the symbol – (octal 02) if negative, and

c. the bar over the last digit indicates that the number has been rounded.

A general flow chart and the coding (not machine checked) in the APL format is enclosed.


Entry
Is C1 = 0?
NO
YES
Transfer floating point number to output
Exit
Is C2 = 0?
Yes
No
Write Sign
Write Decimal Point
Write 10 Digits
Is characteristic positive or negative?
Pos.
Neg.
Write Sign
Write C2 Digits
Write Decimal Point
Write 10 - C2 Digits
Write Sign
Write Decimal Point
Write C2 Zeros
Write 10 - C2 Digits
Round
Transfer stated point number to output
Exit

T00	00	F	F	excess 3 floating
T01	00	F	F	point input
T02	00	F	F	stated point
T03	00	F	F	output
P001	LQ	U001	31006	C_1 to A
2	QT	T01	A	
3	EJ	U002	P007	Is $C_1 = 0$?
4	TP	T00	T02	Floating point
5	TP	T01	T03	number to output
6	MJ	Z	Exit	
7	TP	U003	Q	Store characteristic
8	QT	T01	U014	
9	QS	U013	T01	Place zeros in char. part of floating pt. no.
10	LQ	U001	31014	Store sign of
1	QT	U014	U015	characteristic
2	TP	U004	Q	Store C_2
3	QS	U013	U014	
4	RS	U014	U011	Subtract excess 3 from C_2
5	TP	U005	A	$9 - C_2$
6	ST	U014	U016	
7	TP	U013	U017	Clear storage
8	TP	U013	U018	
9	TP	U014	A	C_2 to A
20	ZJ	P021	P041	Is $C_2 = 0$?
1	ST	U000	U019	$C_2 - 1$
2	TP	U015	A	Is char. pos.?
3	EJ	U006	P034	

P024	RJ	P056	P047	Write sign
5	RJ	P062	P060	Write point
6	RJ	P059	P057	Write C_2 Zeros
7	IJ	U019	P026	
8	RJ	P056	P047	Write 10 - C_2 digits
9	IJ	U016	P028	
30	RJ	P078	P069	Round
1	TP	U017	T02	Stated point number
2	TP	U018	T03	to output
3	MJ	Z	Exit	
4	RJ	P056	P047	Write sign
5	RJ	P056	P047	Write C_2 digits
6	IJ	U019	P035	
7	RJ	P062	P060	Write point
8	RJ	P056	P047	Write 10 - C_2 digits
9	IJ	U016	P038	
40	MJ	Z	P031	
1	TP	U005	U016	Set index
2	RJ	P056	P047	Write sign
3	RJ	P062	P060	Write point
4	RJ	P056	P047	Write 10 digits
5	IJ	U016	P044	
6	MJ	Z	P031	
7	TP	U001	Q	Mask to Q
8	RJ	P068	P063	Shift new word
9	SP	T00	00044	
50	SA	T01	Z	

```
P051   LA   A      00006   } Shift old word
   2   LT   Z      T00     }
   3   TP   A      T01     }
   4   QT   T01    A         Read digit
   5   AT   U018   U018      Write digit
   6   MJ   Z      F
   7   RJ   P068   P063      Shift new word
   8   RA   U018   U011      Write zero
   9   MJ   Z      F
  60   RJ   P068   P063      Shift new word
   1   RA   U018   U007      Write point
   2   MJ   Z      F
   3   SP   U017   00044   }
   4   SA   U018   Z       }
   5   LA   A      00006   } Shift new word
   6   LT   Z      U017    }
   7   TP   A      U018    }
   8   MJ   Z      F
   9   TP   T00    A       } Is rounding
  70   TJ   U008   P078    }   required?
   1   TP   U000   U019      Set rounding digit
   2   TP   U012   U015      Set index
   3   TP   U009   U016      Set 9 tester
   4   TP   U001   Q         Mask to Q
   5   QT   U018   A         Digit to A
   6   EJ   U016   P079      Is digit = 9?
   7   RA   U018   U019      Add one
```

P078	MJ	Z	F	
9	QS	U013	U018	Make 9 = 0
80	LQ	Q	00006	Shift mask
1	LA	U019	00006	Shift 9 tester
2	LA	U016	00006	Shift rounding digit
3	IJ	U015	P075	Repeat 5 more times
4	QT	U017	A	Digit to A
5	EJ	U009	P088	Is digit = 9?
6	RA	U017	U000	Add one
7	MJ	Z	P078	
8	QS	U013	U017	Make 9 = 0
9	RA	U017	U010	Add one to next digit
90	MJ	Z	P078	

U000	00	Z	00001	Modifier
U001	00	Z	00077	Mask
U002	00	Z	00300	Excess 3 zero
U003	00	00007	77777	Mask
U004	00	00007	77700	Mask
U005	00	Z	00011	Decimal 9
U006	00	00006	30000	Excess 3, +
U007	00	Z	00022	Excess 3, .
U008	10	Z	Z	Escess 3, 5
U009	00	Z	00014	Excess 3, 9
U010	00	Z	00100	Carry
U011	00	Z	00003	Excess 3
U012	00	Z	00005	Index
U013	Z			Zero
U014	Z			Temps
U015	Z			
U016	Z			
U017	Z			
U018	Z			
U019	Z			

Talmadge

USEful Note #10 10 April 1957

SUBJECT: Specifications of D.T.M.B. Omnibus Tape Handling Routine for the Univac I Computer.

CONTRIBUTOR: RR

UNIVAC MEMO
No. 114

DATE: June 7 1956
DTMB SERVICE ROUTINE
F. E. Holberton

Navy Department
David Taylor Model Basin
Applied Mathematics Laboratory

D.T.M.B. OMNIBUS*

Programmed by N.Y.U.
Reprogrammed and extended by S.E. Kenady and F. E. Holberton DTMB

The D.T.M.B. Omnibus is a general utility service system, designed to incorporate into a single routine those functions frequently required in preparation and correction of information stored on tape. All subroutines are performed using the standard UNIVAC C-10 code.

* Supercedes all other Omnibus writings.

TABLE OF CONTENTS Page Number

DESCRIPTION The D.T.M.B. Omnibus will perform 18 independent functions from control words manually typed at the computer, or previously prepared on a control tape. Some of the more frequently used functions are:

COPY BLOCKS   CORRECT SINGLE OR CONSECUTIVE WORDS IN BLOCKS

COMPARE BLOCKS - FORWARD OR BACKWARD   MEMORY CLEAR AND INITIAL READ

SKIP BLOCKS - FORWARD OR BACKWARD   SEARCH BLOCKS, THEN COPY

REWIND TAPES   WORD SEARCH   COPY BLOCKS WHILE SEARCHING

SAMPLE BLOCKS OR BLOCKETTES   SELECT AND MERGE DATA BY WORDS

VERIFY TWO UNITYPED TAPES   HIGH SPEED PRINTER COEDIT

DIGIT CHANGE BY EXTRACTION PATTERN   STOP

Functions which produce selected output, such as discrepancies in comparison or sampled data, may be printed on the Supervisory Control Printer or written on tape. Information, if written on tape, is prepared for printing on the High Speed Printer using a Memory Dump board (10 words across with word spacing).

Many functions can be completely described by a single control word of 12 digits. The first character in the control word specifies the function the Omnibus routine is to perform. The interpretation of the remaining 11 digits is dependent upon the function performed. Wherever possible the format of the remaining 11 digits has been standardized. The standard control word format is as follows:

digit	
1	Control function
2	Input Servo number
3	Output Servo number
4	Supervisory control or Servo number for discrepancies during comparison
5	Specification for rewind of input servo
6	Specification for rewind of output servo
7	Write density for output servo
8	Optional comparison of input with output tape(combine with digit 4)
9 - 12	Number of blocks

Some functions require additional information to be supplied in order to perform the operation. The additional information is in the nature of a sub-function and/or a quantity. Functions and sub-functions are interpreted by the Omnibus routine to perform certain operations, but quantities are information supplied as word substitutions.

eg.		
	Function	Copy with corrections
	Sub-function	Block and word number of correction
	Quantity	(Word to be substituted for existing word)
	Sub-function	End of corrections

CONTROL

1. Manual. Control words may be typed manually at the keyboard after the routine prints CONTROL WD. If a function requires additional information, appropriate words will be printed on the S.C. Printer before any request for a type in is made. At the conclusion of any function except the Initial Read and Stop functions, the routine will ask for another CONTROL WD.

2. Control Tape. Control words may be previously prepared on a tape in the order of their desired execution. The last valid control function on a Control Tape must be either an Initial Read or a Stop function. The Control Tape may be mounted on any UNISERVO except minus (-). Set the Conditional Transfer Breakpoint corresponding to the Control Tape UNISERVO number. Clear C, and input ready light if set, and actuate the start bar. The routine will stop on the Breakpoint set. FORCE TRANSFER on the Breakpoint will cause the routine to print TAPE CONTROL and proceed to read the Control Tape and perform the functions specified on the Control Tape. Tape (u), if used in any of the control functions, and the Control Tape will be automatically rewound with interlock when interpreting a Stop or an Initial Read function on the Control Tape. No Rewind control functions are required to rewind the Control Tape or Tape (u).

TO START THE CONTROL TAPE AT OTHER THAN THE BEGINNING, SET BKPT. 6, after selecting the Tape Control Breakpoint. The first word on the Control Tape will be printed on the S.C. Printer and the computer will stop on BKPT. 6. FORCE TRANSFER on BKPT. 6 will cause the following print out to appear: BLK AND WD OF CONTR WD. Type in the block and word number of the location containing the desired starting control word in the following form: 00bbbb 0000wd. Release BKPT. 6. The Control Tape will be rewound without interlock and read forward to the block and word specified by the type in. The Control Tape will be processed starting with the specific control word located.

TO CHANGE A CONTROL WORD ON THE CONTROL TAPE SET BKPT. 5. The computer will stop after printing each control word about to be processed. If a FORCE TRANSFER on BKPT. 5 is performed, the routine will print out TYPE NEW WD and set up a type in. Type in a control word to replace the last control word printed. BKPT. 5 cannot be used to change an incorrect sub-control function.

6

OPERATING INSTRUCTIONS

Omnibus Instruction Tape - Any servo, if using DTMB Service Routine Locator which automatically changes all servo "1S" instructions to "Δ" servo number. THE INITIAL TAPE SELECTOR BUTTON MUST REMAIN SET TO THE SERVICE ROUTINE TAPE NUMBER AT ALL TIMES. The following control words will cause the Service Tape to be read to locate the necessary subroutines: H, K, L, Q, V, and W. At the conclusion of the subroutines, or a clear C operation, the Omnibus routine is read back into the memory. The Service Routine Tape is rewound without interlock after any reading of the Omnibus routine or its externally stored subroutines.

Block Subdivider Button - If tape output is used (a non zero digit typed as digit four of a control word) see notes on Tape u. No print out to instruct the operator of this condition is made. The H. S. P. Codedit routine (H) will instruct the operator to set the appropriate Block Subdivider Button and Stop.

Breakpoints - A few breakpoints may be used during the processing of certain control functions.

BKPT 5, BKPT 6, and Control Tape number - see Control Tape
BKPT 1 - See Search, and Copy While Searching
BKPT 9 - See Word Changer
BKPT 3 - See H. S. P. Codedit
BKPT 0, 1, comma, and Tape number - See Verify
BKPT 5 - See Line Merge
BKPT 7 - See Copy with Corrections

Notes to the Operator.

The Initial Read Function (I) is the only control word which destroys the Omnibus Routine in the memory.

The Omnibus Routine may be used for both control tape and manual option. It should not be necessary to Initial Read the Omnibus Routine into the memory between consecutive users, no matter which option is used.

The Clear C operation will reset the Omnibus Routine to the original conditions, with the exception of the block counter and the location storing the presence of tape u during any set of control words. The presence of tape u is cleared only by a Stop function. A Clear C operation may be performed at any time, except during the reading of the Service Tape.

Clear C to set the Breakpoint option for the Control Tape, if the last word printed was "CONTROL WD." Clear the Input Ready signal before proceeding. The Control Tape Breakpoints are sequenced for execution before the print out of "CONTROL WD".

The "R" control word will not rewind tape u as long as it is remebered in the memory. The Stop function resets this memory location to zero and rewinds tape u with interlock. If it is necessary to rewind tape u because of computer trouble, Clear C to bring up the CONTROL WD and type in the "R" function with tape u specified as the 4th zone alphabetic equivalent of the numeric tape number or rewind the tape manually.
Any control word which makes use of tape u will complete the last partial block stored in the memory and write the data on tape u at the conclusion of the control function.
If it is necessary to determine the block counter after the inability to read a designated number of blocks - Clear C and bring up the CONTROL WD. Type in 500901 000000 as the control word. Memory location 901 will contain the number of blocks read correctly, and copied, if specified by the process.

Storage locations within Omnibus for Specific Information

Block Counter	901
Block Limit	902
Control Word	903
Input Block (tape s)	500-599 and output for Z, X, W Q and K
Second Input Block (tape s2)	560-619 for C, C, and V
Control Tape Input	620-679
Output Tape Option (tape u)	700-759

RERUN

Clear C, and Input Ready if the neon is lighted. Register I is normally empty at the completion of any cortrol word except the Initial Read Function.

The Clear C operation will reset the Omnibus Routine to the original conditions. USE ONLY IN CASE OF ERROR OR TO PROCEED THROUGH CONTROL TAPE BREAKPOINT OPTION.

When the computer stops on a normal Stop Instruction DO NOT CLEAR C OR rI. The computer has been stopped after printing specific instructions to the operator on the S. C. Printer or from a stop control word or an Initial Read control word.

The actuation of the Start Bar after a Stop control function will send control to 000.

The actuation of the Start Bar after an Initial Read control word will transfer the contents of rI to 000-059 and send control to 000.

Do not Clear C if an error has been made during the reading of the Omnibus Routine into the memory from the Service Tape. The Clear C operation may be momentarily non operative when searching for certain control function subroutines.

OMNIBUS SUPERVISORY CONTROL PRINT OUTS REQUIRING NO ACTION

Reason	Function	Type of Control	Print Out
End of Subroutine	K Q W X Z	Both	ΔΔENDΔCOPYΔΔ
End of comparison subroutine	C D W X Z	Both	ΔΔENDΔCHECKΔ
End of H. S. P. Cododit	H	Both	ΔENDΔCODEDIT
End of Line Merge	L	Both	ΔENDΔMERGEΔΔ
End of Verify	V	Both	≠ENDΔVERIFYΔ
No print outs designate end of functions	B F I R S 5 9	Both	
Tape number n contains xxxx blocks of edited output	H	Both	TnΔBLKSΔxxxx
Word on which search is made was not found on tape	K	Both	NOΔSUCHΔWORD
Block and word number of searched word found	K Q	Both	ΔBxxxxΔWxxΔΔ
Total number of blocks merged	L	Both	MRGΔBLKSxxxx
Result of forcing BKPT s to select word	V	Both	≠SELECTΔsΔΔΔ
Result of not forcing BKPT s to select word	V	Both	≠SELECTΔNONE
Result of forcing BKPT s to advance word	V	Both	≠STEPΔΔΔsΔΔΔ
Result of not forcing BKPT s to advance word	V	Both	≠STEPΔNONEΔΔ
Result of forcing BKPT 0 to print last output word	V	Both	≠PRIORΔWORDΔ
Result of setting up skip inst. after BKPT(.)to omit choices previously printed	V	Both	≠NOTΔDONEΔΔΔ
If more than 47 consecutive subcontrol words for the Word Changer are listed on the control tape, the last subcontrol word used will be printed and subsequent subcontrol words skipped until the "Z" subcontrol word is reached	W	Tape	LASTITEMUSED
Printed after BKPT 9 is forced to omit changing a specific word	W	Manual	ΔSKIPPEDΔΔΔΔ
Result of forcing BKPT option for control tape		Tape	TAPEΔCONTROL
All control words are printed when operating from Tape Control Option. Subcontrol words are not printed, except Line Merge subcontrol words.			

OMNIBUS SUPERVISORY CONTROL PRINT OUTS REQUIRING ACTION

Action	Function	Type of Control	Print Out
Type in control word.	ALL	Manual	ΔCONTROLΔWDΔ
Wrong form of control word, type in replacement.	ALL	Both	REJECTΔWORDΔ
Type in subcontrol word for location of correction	Q X	Manual	ΔBLKΔANDΔWDΔ
Type in subcontrol word for Line Merge	L	Manual	TYPEΔCONTROL
Type in subcontrol word for Word Changer	W	Manual	CCCOLDEEENEW
Type in Date for H. S. P. Cododit page	H	Manual	ΔTYPEΔDATEΔΔ
Type in Heading of data being edited	H	Manual	TYPEΔHEADING
Type in subcontrol word for H.S.P. Cododit	H	Manual	CIOLLISSSBBB
Subcontrol word wrong of listed out of block sequence. Type in replacement for subcontrol word.	L Q W X	Both	CHECKΔORDERΔ
Type in QUANTITY on which search is made	K Q	Manual	ΔWHATΔWORDΔΔ
Type in QUANTITY, used after "Ax" subcontrol when x ≠ 0 or Z	L	Manual	WHATΔFILLΔWD
Type in as many words as specified by n in subcontrol word "T"	L	Manual	TYPEINΔWORDS
Last output block in Line Merge is not full, type in a Quantity to be used to fill to the end of the block.	L	Both	BLKΔNOTΔFULL WHATΔFILLΔWD
Type in word to replace compared words in Verify after force transfer on Bkpt 1.	V	Both	≠TYPEΔWORDΔΔ
Verify. The last word on the input tape is printed out. Occurs when omissions or duplications of verified data produce a last partial output block. Type in a QUANTITY to be used to fill the last output block.	V	Both	≠LASTΔVALIDΔ WORDΔΔΔΔΔΔΔΔ ≠TYPEΔFILLΔΔ
Set S. C. Printer for 3 words. Actuate start bar. Occurs when discrepancies are found and S. C. Printer option is specified.	C D W X Z	Both	SETΔSCΔ3ΔWDS
Set S. C. Printer to normal. Actuate start bar.	V	Both	SETΔSCΔNORML
Set block subdivider listed and actuate start bar.	H	Both	SETΔB.S.D.xΔ
Mount a blank on output servo, previous tape is full.	H	Both	MT.ΔNEWΔTAPE

From forcing transfer on BKPT 5 to change control word on control tape. Type in replacement control word.	Tape	TYPE△NEW△WD△
From forcing transfer on BKPT 6 to start a control tape at other than beginning. Type in block and word of control tape starting word in form 000bbb0000wd	Tape	△BLK△AND△WD△ OF△CONTR△WD△

NOTES ON THE USE OF THE OPTIONAL TAPE OUTPUT - TAPE u

Six control words can make use of the optional tape u output. These are C, D, S, W, X and Z. The information written on tape u is prepared for printing on the High Speed Printer using a memory dump board. If the data contains no multi-line or fast feed symbols, it can be printed on NORMAL for best results. The C, D, W, X, and Z control words use three words in the blockette for: the old word, block and word number, and the new word. The related servo numbers are listed in the heading. The remaining words in the blockette are filled with spaces.

The sample routine (S) will make use of one, two, six or seven words in the blockette, depending upon the choice of digits 3 and 6 in the S control function. Sampling a block without listing the block number will print one word, or six words if sampling blockettes. If the block number is printed, two or seven words of the blockette will be used when sampling blocks and blockettes respectively.

At the completion of any control function using tape u, the partial block in the memory is filled with spaces and written on tape u, so that the pertinent information will be on tape when entering a new control sequence.

The blocks written on tape u are written at 100/in. density and are not counted, so that care must be exercised by the user not to exceed 1250 blocks.

The Omnibus routine does not remind the operator to set the Block Subdivider Button when specifying tape u. When the fourth digit of the above control words is not zero, the appropriate Subdivider Button must be set.

It has been assumed that the same servo number will be used for tape u, by the programmer on any sequence of operations requesting optional output. If it is desired to vary the servo number for tape u during a sequence of operations, it must be remembered that only the last tape number used will contain a Printer Stop at the end of the tape and be automatically rewound when a Stop function is given. The other tapes will not contain a Printer Stop symbol, but can be rewound with the R control word after a new tape u number has been initiated.

Tape u will not respond to the R control function even when called for. This safeguards any accumulated output from being destroyed by an operator error in typing an incorrect tape rewind control function.

If it is necessary to restart the process and rewind tape u, type in a Rewind control function and specify the servo u number by the 4th zone alphabetic equivalent of the servo number or rewind tape u manually.

Never use tape u for the output for the H. S. P. Codedit. The H. S. P. Codedit requires a special plugboard and rewinds both input and output tapes before starting the codedit, which would destroy any previous data on tape u.

13

COPY WITHOUT CORRECTIONS

$Z\ s\ t\ u\ r_s\ r_t\ d\ c\ b\ b\ b\ b$

COPY [Z] from tape (s) to tape (t) (bbbb) blocks at (d) density.
d = 0 for 100/in., d ≠ 0 for 20/in. density.

If digit c is greater than zero, compare tapes (s) and (t) word by word reading in the backward direction. No discrepancies should occur. If the two tapes differ, the words which differ, together with the block and word number of each discrepancy will be either written on tape or printed on the Supervisory Control Printer as specified by digit u.

u = 0 Print on S. C. Printer
u ≠ 0 Write on Tape (u) (u) must be 1 to 9 or -. Set Block Subdivider u.

The r digits control the rewind options for the input (r_s) and output (r_t) tapes.

r = 0 Do not rewind tape. Reposition tape bbbb blocks forward if comparison has taken place.
r = 6 Rewind tape without interlock.
r = 8 Rewind tape with interlock.

COPY WITH CORRECTIONS

$X\ s\ t\ u\ r_s\ r_t\ d\ c\ b\ b\ b\ b$

COPY WITH CORRECTIONS [X] from tape (s) to tape (t) a total of (bbbb) blocks at (d) density. d = 0 for 100/in., d ≠ 0 for 20/in. density. A lsit of corrections together with the block and word number of the corrections, terminated by a word of all Z's will follow the control function.

If digit c is greater than zero tapes (s) and (t) are compared word by word reading in the backward direction. The words which differ, together with the block and word number of each discrepancy will be written on tape or printed on the Supervisory Control Printer as specified by digit u.

u = 0 Print on S. C. Printer
u ≠ 0 Write on tape (u) (u) must be 1 to 9 or -. Set Block Subdivider u.

The r digits control the rewind options for the input (r_s) and output (r_t) tapes.

r = 0 Do not rewind tapes. Reposition tape bbbb blocks forward if comparison has taken place.
r = 6 Rewind tape without interlock.
r = 8 Rewind tape with interlock.

Sub-control words

n n x x x x 0 0 0 0 y y	n n	Number of consecutive corrections listed under the Sub-control word nn = 00 or 01 for a single word correction. Consecutive corrections must be within the block number listed.
	x x x x	Block number
	y y	Starting word number for corrections
Z Z Z Z Z Z Z Z Z Z Z Z		End of corrections.

When using the nn digits to specify a consecutive list of corrections, follow the subcontrol word with nn QUANTITIES to be supplied as corrections. The block and word number represent the location of the first correction. When operating from manual control, the first old word will be printed on the S. C. Printer, and nn consecutive "input ready" signals will be set up without printing the consecutive old words.

NOTE: All subcontrol words must be listed in ascending order by block number.

When a subcontrol word from a Control Tape specified a consecutive list of corrections and the nn digits, describing the number of corrections, have been typed incorrectly, the routine may attempt to decode a QUANTITY as a subcontrol word and produce one of two kinds of errors.

1. Adder Alpha

The routine has attempted to add digits 1 and 2 to 11 and 12 of a Quantity instead of a subcontrol word. Digits 1 and 2 are added to 11 and 12 of a subcontrol word to determine if nn + yy is > 60.

Remedy: Set BKPT 7

SCI_{CR} 000000 U00224

Proceed as for CHECK ORDER Remedy.

2. CHECK ORDER printed, followed by a QUANTITY rather than a subcontrol word.

Remedy: If the word printed out after CHECK ORDER is not a subcontrol word, but a QUANTITY, and the QUANTITY is either the 2nd consecutive correction word when nn was typed as 00, or the nn + 1st consecutive correction where nn was less than the number of corrections, the following insertions may be made:

Set BKPT 7

Type in a new subcontrol word nnxxxx0000yy

Where yy is the location of the printed QUANTITY, nn is the number of corrections to be made, and xxxx is the same block number as the previous subcontrol word.

FORCE TRANSFER on BKPT 7 to cause the routine to use the type in as an insertion, rather than a replacement. A word of all "Z's" may not be typed as an insertion when using BKPT 7. See Control Tape BKPT 6 option.

COMPARE FORWARD $C\ s_1\ s_2\ u\ r_{s1}\ r_{s2}\ 0\ 0\ b\ b\ b\ b$

COMPARE [C] tapes (s_1) and (s_2) reading in the forward direction (bbbb) blocks. If any differences occur between tapes (s_1) and (s_2) record the word from tape (s_1), the block and word number, and the word from tape (s_2) on the Supervisory Control Printer if digit $u = 0$. If digit $u \neq 0$ write discrepancies on tape (u). $u \neq 0$ must be 1 to 9 or –. Set Block Subdivider u. The r digits control the rewind option for tapes (s_1) and (s_2).

r = 0 Do not rewind tape.
r = 6 Rewind tape without interlock.
r = 8 Rewind tape with interlock.

COMPARE BACKWARD $D\ s_1\ s_2\ u\ r_{s1}\ r_{s2}\ 0\ 0\ b\ b\ b\ b$

Same as COMPARE FORWARD except tapes are read and compared in the backward direction. All discrepancies are listed in a descending order with the first block read in backward as block bbbb and the last block read as 0001. The tapes are not repositioned forward if rewind is not specified.

SKIP FORWARD F s 0 0 0 0 0 0 b b b b

SKIP FORWARD [F] (read in the forward direction) on tape (s) (bbbb) blocks.

SKIP BACKWARD B s 0 0 0 0 0 0 b b b b

SKIP BACKWARD [B] (read in the backward direction) on tape (s) (bbbb) blocks.

REWIND TAPES WITHOUT INTERLOCK R s s s s s s s s s s 0

REWIND [R] the tapes specified by the digits s. Digits (s) must be 0, 1 to 9 or –. Any number of tapes may be rewound with a single control word. Fill the unused digit positions with zero. It is not necessary to type the servo numbers in any special order and zeros may be interspersed.

If tape u has been written on at any time during the use of the Omnibus routine, the "R" control function will omit the rewinding of this tape even though called for. This safe guards the information on tape u from receiving a Printer Stop block which would destroy the first block on the tape if allowed to rewind. The Printer Stop is automatically supplied when a STOP OR INITIAL READ control function is executed, and tape u is rewound with interlock.

NOTE: It is never necessary to rewind the control tape (if used) or tape u. This operation is performed automatically at the execution of a STOP or INITIAL READ function.

MEMORY CLEAR, INITIAL READ I s a 0 0 0 0 0 0 0 0 0

Clear the memory to zero if digit a = 0. If digit a ≠ 0, clear the memory to Stop Instructions with the address of line number inserted in the stop instruction. e.g. 900060 900060
900999 900999

Read one block from Tape (s) into rI. The computer will Stop. Actuation of the start bar will cause (rI) to go to 000-059 and control will be transferred to line 000. The Omnibus Routine has been destroyed.

If tape (u) has been specified during any previous Omnibus Control function, the Initial Read function will cause a Printer Stop block to be written on Tape (u) and Tape (u) will be rewound with interlock.

If a control tape has been used, the Initial Read Function will cause the control tape to be automatically rewound with interlock.

The last control word for either manual or control tape option should be an Initial Read or a Stop function.

STOP 9 9 9 9 9 9 9 9 9 9 9 9

The control tape, if used, will be rewound with interlock. Tape (u), if previously specified, will receive a Printer Stop block and be rewound with interlock. The computer will be stopped. The Omnibus routine is reset to the initial conditions and the storage for control tape and tape(u) are cleared.

Actuation of the start bar will send control to 000 (same as a Clear C operation).

The last control word for either manual or control tape option should be a Stop or an Initial Read function.

SKIP - NO OPERATION 0 0 0 0 0 0 0 0 0 0 0 0

Omnibus will perform a skip operation during a tape control option only. It is used to delete control words from a control tape if certain functions are not to be performed, so that retyping of a control tape is not necessary.

SAMPLE

S s k u r_s x w w b b b b

SAMPLE [S] one word in each block or blockette, the location specified by digits (ww), from tape (s) for a total number of bbbb blocks. If digit x = 0, sample one word per block. If digit .. sample one word per blockette. The sampled words will be printed on the S. C. Printer if digit u = 0 and written on tape (u) if u ≠ 0. Digit u must be 0, 1 to 9 or minus (-).

If digit k = 0 the block number of each sampled word will be omitted in the print out.

If digit k ≠ 0 the block number will appear for each new block sampled.

When the output is prepared for tape (u), one, two, six, or seven words will appear in each H. S. Printer blockette, depending upon the specifications for digits k and x.

Sampling by blocks will produce either one, or two words per output line.

Sampling by blockettes will produce either six or seven words per output line. When samplingblockettes, digits (ww) must lie between 00 and 09, which specifies the word position within the blockette, to sample correctly.

The r digit controls the rewind option for the input tape s.

r = 0 Do not rewind Tape s.
r = 6 Rewind Tape s without interlock.
r = 8 Rewind Tape s with interlock.

PRINTOUT OR BLOCKWRITE

5 0 0 m m m 0 0 0 0 0 0

The "5" control word is primarily provided to aid the operator in determining the block counter number, without manually setting up the Static Register, when a tape cannot be read to its specified block limit.

Clear C and type in control word 500901 000000 will cause this instruction word to be executed and the block counter in 901 will contain the number of the last correct block read (and written if during copy).

The control word may be used to write a block of zeros on a tape if typed as 5n0700000000 provided no output tape u option has been performed which uses 700-759 as the output block, and the H, V or W functions have not been performed since the last memory clear operation.

The second instruction in the word may be used for a limited selection. e.g. 00, 10, 50 or 5n instruction.

SEARCH, THEN COPY WITH CORRECTIONS $\quad$ Q s t 0 r_s r_t w w b b b b

Follow the control word with a 12 digit QUANTITY on which the search is made. The search is made on all 12 digits.

ww > 59	All words are searched
ww ≤ 59	Only one word, as specified, in block is searched
bbbb	Number of blocks to be copied with corrections, including the block containing the searched word.

SEARCH [Q] on tape (s) until the 12 digit QUANTITY supplied is equal to the searched word in location (ww), then copy with corrections from tape (s) to tape (t) (bbbb) blocks, starting with the block containing the searched word. Follow the QUANTITY with the sub-control words for COPY WITH CORRECTIONS. Tape (t) is written at 100/in. pulse density.

The r digits control the rewind options for the input (r_s) and output (r_t) tapes.

r = 0 Do not rewind tape.
r = 6 Rewind tape without interlock.
r = 8 Rewind tape with interlock.

The block and word number of the searched QUANTITY will be printed on the S. C. Printer.

To SEARCH WITHOUT COPY $\quad$ Q s 0 0 r_s 0 w w 0 0 0 0 $\quad$ set the block number equal to zero. The block and word number of the searched QUANTITY will be printed on the S. C. Printer. Follow the QUANTITY specified for the search with the next control word.

If it is desired to search after the initial agreement of the QUANTITY and the searched data set BKPT 1 at the beginning of the search process. The block and word number will be printed and the computer will stop on BKPT 1. FORCE TRANSFER to continue the search process. This procedure may be continued as often as desired. If the word is never found on thetape, the tape will either stop on a two block read or if a blank tape beyond the data, will cause the tape to read off the end.

COPY - NO CORRECTIONS, WHILE SEARCHING   K s t 0 r_s r_t w w b b b b

Follow the control word with a 12 digit QUANTITY on which the search is made. The search is made on all 12 digits.

	ww > 59	All words are searched.
	ww ≤ 59	Only one word, as specified, in each block is searched.
bbbb		Upper limit of blocks copied while searching. If bbbb is reached without finding the search word, the routine prints out NO SUCH WORD and proceeds to the next control word.

COPY [K] from tape (s) to tape (t) while searching for agreement between the QUANTITY and the searched word in position (ww). Copy to and including the block containing the searched QUANTITY. Tape (t) is written at 100/in. pulse density.

The r digits control the rewind options for the input (r_s) and output (r_t) tapes.

r = 0 Do not rewind tape.
r = 6 Rewind tape without interlock.
r = 8 Rewind tape with interlock.

The block and word number of the searched QUANTITY will be printed on the S. C. Printer.

If it is desired to copy and search after the initial agreement of the QUANTITY and the searched data, set BKPT 1 at the beginning of the search process. The block and word number will be printed and the computer will stop on BKPT 1. FORCE TRANSFER tocontinue the search process. This procedure may be continued as aften as desired. If the word is never found on the tape when the upper limit block count (bbbb) is reached, the routine will print out NO SUCH WORD, perform the rewind option, and proceed to the next control word.

20

H. S. P. CODEDIT H 0 0 0 0 0 0 0 0 0 0 0

The Omnibus Routine will call in the DTMB H. S. P. CODEDIT, which has been modified to be used as part of Omnibus, if desired. The control words performed by the CODEDIT are described in UNIVAC MEMO 111.

The control words associated with the CODEDIT must not exceed 60 words (not including the control function word H00000 000000), but may be split between blocks on a control tape.

At the completion of the CODEDIT, the Omnibus Routine is automatically called back and subsequent words on the control tape, if used, are performed.

If it is desired to type the control words for the CODEDIT in a separate block so that they may also be used by the ANALYZER routine for future processing, fill the block containing the "H" function with zero and compile the CODEDIT control words in the block following. The CODEDIT routine, if called in by an Omnibus control tape, will consider the first non zero word following the "H" function to be the date (the first CODEDIT control word).

The H. S. P. CODEDIT may be called in by a manual option from Omnibus but perform a control tape CODEDIT by setting BKPT 3 and FORCING TRANSFER when the CODEDIT has been located.

When performing a CODEDIT from an Omnibus control tape, the control words are assumed to be on the same control tape.

Do not use tape (u) as the output for the CODEDIT, because the input and output tapes are rewound at the beginning of the CODEDIT and any accumulated data on tape (u) will be destroyed. Information accumulated on Tape (u) from previous Omnibus control words will be printed using a memory dump plugboard and the output from the CODEDIT requires a special board.

Rerun: Clear C

When the CODEDIT has been called in from a manual control Omnibus, the Clear C operation will start the CODEDIT routine over again.

When the CODEDIT has been called in from an Omnibus control tape, the Clear C operation will recall the Omnibus routine and perform the general Omnibus Clear C function.

21

LINE MERGE - COPY BY WORDS

L 0 t 0 0 r 0 0 0 0 0 0

The LINE MERGE routine permits information to be copied from many tapes to a single output tape on a word by word basis instead of a block by block basis. The control word specifies only the output servo number (t) and the output rewind option (r_t). r_t = 0, 6 or 8, do not rewind, rewind without interlock, or rewind with interlock, respectively. The remaining controls are performed by the subcontrol words which follow the control word.

The subcontrol words permit three options for copying data.

1. Copy from tapes (other than the control tape).
2. Type in data at the console, or transfer data from the control tape.
3. Fill a specified number of words, or to the end of the current output block, with a specific QUANTITY.

Sub control functions

M BLK WD s blk wd	-	Copy [M] from tape (s) starting with block and word number BLK WD to and including block and word number blk wd.
Ax0000 00nnnn	-	Add [A] (nnnn) words of the same pulse combination to the output tape. If x = 0, words of zero will be written. If x = Z, words of all Z's will be written. If x = X, the QUANTITY specified by the word will be written on the output tape.
Ax0000 000000	-	Fill to the end of the current output block with the QUANTITY specified by the x digit. If the previous block had been completely filled prior to the execution of the subcontrol word, a complete block of fill digits will be written each time the subcontrol word is given.
T00000 00nnnn	-	Transfer [T] the next nnnn words from the control tape, or type in nnnn words at the console (if on manual option) to the output tape.
ZZZZZZ ZZZZZZ	-	End of Line Merge.

Failure to complete the information in the last output block when all Z's is supplied as a subcontrol word, will cause the routine to print out BLOCK NOT FULL and ask for a type in of a 12 digit QUANTITY which will be used to fill to the end of the current output block. The number of blocks on the output tape will be printed on the S. C. Printer.

No provision is made for rewinding the input tapes within the Line Merge subroutine, but may be supplied as a "R" control word after the word of all Z's.

The format of the "M" subcontrol word limits the input block number to three digits or 999 blocks.

When the Line Merge Control word is specified, the routine assumes that all 10 servos on the computer are (figuratively speaking) about to read block one. The routine remembers the number of blocks read from all servos, and will read the tape in the forward or backward direction to locate the starting block number, depending upon the previous operation on any specific servo (s). Block and word numbers are always specified in relation to their physical location on the tape and not related to the position of the reading head at any time except at the very beginning when each tape number, regardless of previous motion, is assumed to be at the block beginning for counting purposes.

Information may be copied any number of times from tapes and no restrictions are placed on switching from one input servo to another and back again.

The starting block and word number must be equal to or less than the ending block and word number for any single "M" subcontrol word. The routine assumes a minimum of one word will be copied when supplying the "M" subcontrol word. Failure to meet this requirement will cause the routine to print out CHECK ORDER and call for a type in of a subcontrol word to substituted for the error one.

If it is known in advance that a Line Merge subcontrol word on a control tape is in error set BKPT 5. Each subcontrol word (on the Line Merge only) is printed on the S. C. Printer before it is executed. The computer will stop on BKPT 5 after printing each subcontrol word. FORCE TRANSFER on BKPT 5 after the incorrect subcontrol word is printed. The routine will print out TYPE CONTROL. Type in a subcontrol word to replace the error. The Line Merge routine is the only routine which permits the correction of a subcontrol word in connection with BKPT 5. BKPT 5 is normally reserved for control word correction only.

23

WORD CHANGER

W s t u r_s r_t d c b b b b

The Word Changer routine permits substitution of digits in the operation parts, the address parts of an instruction word, or in specified digit positions in a full word. Sub control functions specify the mode of examination, the digit quantities to be searched on, the extractor to be used in the comparison and the quantity to be substituted when equality is found.

The word changer routine may be used as a Code search operation, searching on as many as 47 different addresses simultaneously. The routine may be used to perform a variety of changes, such as changing all servo numbers, supplying a minus to all D, X, A or S instructions or changing a group of addresses when constants have been moved in the coding without listing each change independently with an X control word.

The complete list of subcontrol function words is stored in the memory at the beginning of the process. Each word on the input tape is compared against the list and appropriate changes are made when agreement is found.

Copy (bbbb) blocks from tape (s) to tape (t) making the designated changes described by the subcontrol list. Write on tapes (t) at pulse density (d). If d = 0, write at 100/in, if d ≠ 0, write at 20/in. density.

If digit c <u>is greater than zero</u>, compare tapes (s) and (t) in the backward direction, printing the differences on the S. C. Printer if digit u = 0, or on tape (u) if u ≠ 0. If c is = 0, no comparison will take place.

The r digits control the rewind options for the input (r_s) and output (r_t) tapes.

r = 0 Do not rewind tape. Reposition tape bbbb blocks forward if comparison has taken place.
r = 6 Rewind tape without interlock.
r = 8 Rewind tape with interlock.

Sub-control words - For 1/2 word operation.

CCC OLD EEE NEW

CCC = OPR if digits 1 - 3 and 7 - 9 (operation part of an instruction word) are to be examined.
CCC = ARD if digits 4 - 6 and 10 - 12 (address part of an instruction word) are to be examined.
OLD = Three digit quantity (of operation or address) on which the search is made.

EEE = Extractor pattern used to determine the equality of the old pattern and the insertion of the NEW. If all

NEW = Three digit quantity which is used (in conjunction with EEE) to replace the OLD quantity.

ZZZ ZZZ ZZZ ZZZ   End of list for modification.

Sub-control words - For full word operation

CCC OLD EEE NEW

CCC =	FULL△WORD△△△	12 digits for full word control.
OLD =	( )	12 digits for old word.
EEE =	( )	12 digit extractor
NEW =	( )	12 digits for new word.

Comparisons and substitutions are made by the extractor pattern of a full word. If all digits of the OLD word are to be compared, the extractor must be all ones.

ZZZ ZZZ ZZZ ZZZ   End of list for modification.

A maximum of 47 sub-control function words (OPR, ADR, or FULL WORD) may be operated on at any time. Both full and 1/2 word controls may appear in the list.

Each sub-control set is separated into a 10 word item and stored in the computer. 47 such items, not including ZZZ ZZZ ZZZ ZZZ, may be used in a single list. If the number of subcontrol functions on a control tape exceeds this number the computer will print out LAST ITEM USED and the last subcontrol function used will be printed on the S. C. Printer. The control tape is then searched for the word of all "%'s" and all intervening subcontrol words are not used.

The sub-control function is determined by examination of only the 2nd digit in CCC.   OPR, ADR, FUL

On manual control, each old word, block and word number, and substituted word is printed on the S. C. Printer. If it is desired not to make a specific cahnge, BKPT 9 should be set when approaching the specific block number. The routine will stop after printing the old word, and block and word number. FORCING TRANSFER on BKPT 9 will cause the routine to print SKIPPED and omit making the specific change on the output tape. This option is not available from control tape option, because the S. C. Printout of each change is omitted.

VERIFY TAPE s_1 AND TAPE s_2 V s_1 s_2 t $r_{s_1 s_2}$ r_t d 0 b b b b

CORRECT OUTPUT TO TAPE t

The VERIFY Routine compares two input tapes and makes a third tape of corrected data. Provisions have been made to take care of duplications, or omissions on either tape and be able to get them back in step for comparison.

In order to correctly process any discrepancies which may be found on the input tapes, it is assumed that the hard copy which was used in the original tape preparation is available to the operator during the verify process.

Each input tape (s_1) and (s_2) must contain the same number of blocks (bbbb). The output tape (t) density is specified by digit (d). If d = 0, write at 100/in. density. If d ≠ 0 write at 20/in. density. No automatic comparison of the output tape with an input tape is possible.

Tapes (s_1) and (s_2) are rewound if ($r_{s_1 s_2}$) = 6 or 8 and not rewound if = 0.

Tape (t) is rewound if (r_t) = 6 or 8 and not rewound if = 0.

Tapes (s_1) and (s_2) may not be mounted on servo minus (-) because Conditional Transfer Breakpoints, corresponding to the servo numbers s_1 and s_2 are used when discrepancies are found. To eliminate any confusion during the processing, tapes s_1 and s_2 should be mounted on any servo number from 2 to 9. Breakpoint 1 is used for a special function.

Whenever a discrepancy occurs between tapes (s_1) and (s_2) the word on tape (s_1) the block and word number (related to the destination in the output block) and the word on tape (s_2) will be printed on the S. C. Printer. The computer will stop.

The following options are available by FORCE TRANSFER BKPT. OPTION:

1. Print the last correct word on the output tape. Used when the two words are completely different to determine if an omission or duplication has occurred. BKPT 0 must remain set until a selection is made for subsequent operations because the routine will revert back to the BKPT 0.	F.T. BKPT 0
2. Then- Select either one of the two input words. FORCING TRANSFER on the BKPT corresponding to the tape number will cause the corresponding word to be selected for the output word.	F.T. BKPT = to the servo number of the tape desired for selection.
3. Or - Type in a new word at the S. C. Console.	F.T. BKPT 1

4. Or - Select none. Actuate the start bar without forcing any BKPTS.

The advancing of the input words for the next comparison is independent of the selection for the output. Either, both or neither may be stepped by forcing the breakpoint or breakpoints associated with the tape numbers. The selection precedes the advancing of the word in computer operations.

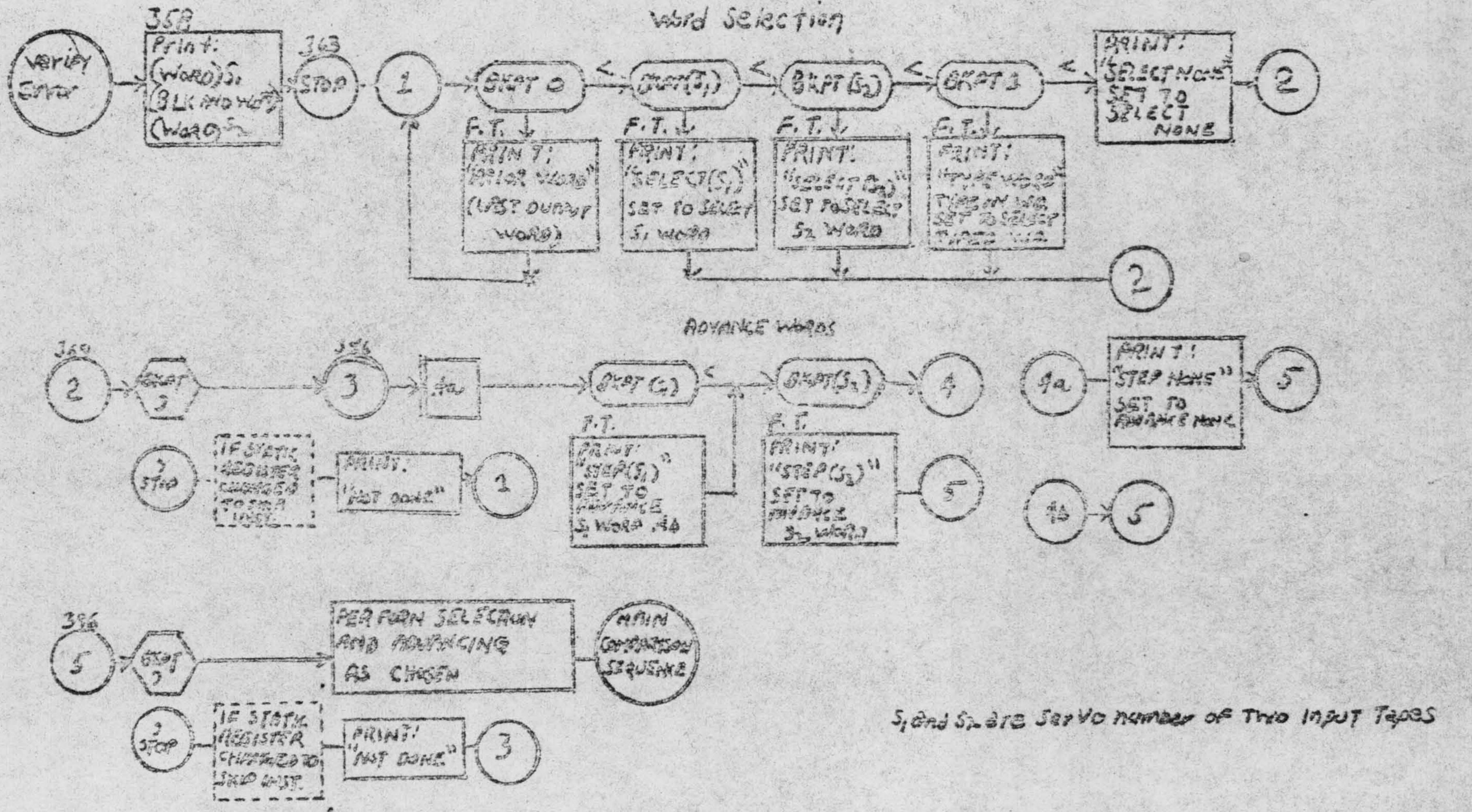
A complete record of all error words, and choices is recorded on the S. C. Printer. The routine will print out SELECT TAPE ( ) of a word chosen and STEP TAPE ( ) when a word is advanced. If BKPT COMMA (,) is not used, both Breakpoints referring to (s_1) and (s_2) should remain set at all times.

By setting BKPT COMMA (,) the computer will stop after printing each choice, but before the operation is actually performed. If the operator has forced the wrong BKPT., it is possible to correct his mistake. Set the correct BKPT and replace the instruction currently in the Static Register by a skip instruction. Actuation of the Start Bar will cause the routine to print NOT DONE and stop on the BKPT set and give the operator another chance to correct the error choice.

When omissions of data occur on both tapes at different locations, it is possible to complete the verify sequence with more words than were originally on the two input tapes and the last output block will be partially filled. Under this condition the routine will print out LAST VALID WORD and the contents of the last word. It will then print TYPE FILL and call for a type in of the QUANTITY to be used to fill the last output block. When the number of blocks on the output tape is not the same as the original input tapes, the new block count will be printed on the S. C. Printer.

The following flow chart is provided to clarify the Breakpoint procedure required to handle errors on tapes to be verified.

Verify Error Sequence with Breakpoint Options



S_1 and S_2 are Servo number of Two Input Tapes

EXAMPLE OF OMNIBUS CONTROL TAPE

This control tape will combine parts of two old instruction tapes, change a few words and cododit the new tape produced. It will verify two unityped data tapes and add additional data from a previous data tape to the verified tape. It will sample the data tape, sending the block number and the second word of each blockette to the output on servo 9. Block subdividers for 8 and 9 are depressed. Tapes are mounted as follows:

Servo No. 1	Blank for instructions
" " 2	" " data
" " 3	Old instructions to be removed after rewind with interlock. Replace with tape to be verified.
" " 4	Old instructions (no ring)
" " 5	Old data tape
" " 6	Control tape (F.T. BKPT 6)
" " 7	Tape to be verified
" " 8	Blank for cododit
" " 9	" " tape option output
" " -	Omnibus routine

Word No.	Word	Purpose
00	Z41069 010002	Copy instructions without corections from servo 4 to servo 1. Compare, rewind 4 without interlock, reposition 1.
01	X31930 010005	Copy 5 blocks from servo 3 to servo 1.
02	020001 000058	Correct 3 successive words beginning
03	320380 B00380	in block 1 word 58. Compare and write
04	L00152 Q00181	the discrepancies on servo 9. Rewind
05	000002 000000	servo 3 with interlock. Reposition 1.
06	000000 U00178	
07	ZZZZZZ ZZZZZZ	
08	L01000 000000	Copy from block 3 word 20 through
09	M00320 401006	block 10 word 06 from servo 4 to
10	T00000 000002	servo 1. Transfer 2 words from the
11	900000 U00163	control tape to the output for servo
12	END△CO DING△△	1. Fill the rest of the block with
13	A00000 000000	zeros.
14	ZZZZZZ ZZZZZZ	
15	B10000 000014	Read backward on servo 1, 14 blocks
16	R40000 000000	Rewind servo 4.

17 18 19 20 21 22 23 24 25	W14966 001014 ADD500 111400 OPR540 111530 OPR640 111630 FULL∆W ORD∆∆∆ FEB.∆1 6∆1956 111111 111111 JUNE∆1 6∆1956 ZZZZZZ ZZZZZZ	Change all of the references to memory location 500 to 400. Change all54 and 64 orders to 53 and 63 orders. Update the tape from FEB. 16, 1956 to JUNE 16, 1956
26 27 28 29	H00000 000000 JUNE∆1 6∆1956 BASIC∆ CODE∆1 Z48000 000014	CODEDIT 14 blocks on servo 4 with output on servo 8.
30	V37260 000045	Compare the two tapes on servos 3 and 7 and write a verified tape on servo 2. Rewind 3 and 7 but leave 2 positioned for additional data.
31 32 33	Q52000 590015 000000 15CD01 ZZZZZZ ZZZZZZ	Search on tape 5 for a block containing 000000 15CD01 in word 59. Copy 15 blocks to tape 2 with no corrections.
34	D52060 000015	Compare 15 blocks backward on servos 5 and 2. Rewind servo 5.
35	F20000 000015	Reposition servo 2
36 37	K52060 590200 000000 16AC01	Copy from tape 5 to tape 2 through a block containing 000000 16AC01 in word 59. The I.D. should be found within 200 blocks. Rewind tape 5.
38	520500 620000	Copy the last block again and rewind tape 2.
39	S21961 020261	Sample the second word of each blockette on servo 2. Include the block number in the output which will go on tape 9. Rewind 2.
40	999999 999999	Rewind tapes 9 and 6. Stop.

Note: Since the programmer was not sure there were 200 blocks to be copied from servo 5 to servo 2 using the K function, he may wish to set breakpoint 5 during the K function. After S21961 090261 prints on SCP, he may force transfer on breakpoint 5 and adjust the block limit (261) by typing in a new control word.

DAVID TAYLOR MODEL BASIN OMNIBUS

OPERATION	CONTROL	SERVO IN	SERVO OUT	DIFF.	REWIND IN	REWIND OUT	DENS.	COMP.	BLOCKS
Digit Position	1	2	3	4	5	6	7	8	9 - 12
COPY WITHOUT CORRECTIONS	Z	s	t	0 SCP u T_u	0 6 } s 8	0 6 } t 8	=0(5n) ≠0(7n)	=0 no >0 yes	b b b b
WITH CORRECTIONS	X	s	t	"	"	"	"	"	b b b b
WORD CHANGER - INST., ADDRESS, or FT. WD.	W	s	t	"	"	"	"	"	b b b b
VERIFY Ts_1 and Ts_2 CORRECT OUTPUT to Tt	V	s_1	s_2	t	" (s_1, s_2)	" t	"	0	b b b b
COMPARE Ts_1 to Ts_2 FORWARD READ	C	s_1	s_2	0 SCP u T_u	" (s_1)	" (s_2)	0	0	b b b b
BACKWARD READ	D	s_1	s_2	"	"	"	0	0	b b b b
SEARCH, THEN COPY WITH CORRECTIONS	[illegible]	s	t 0	0	" (s)	" (t)	word No.		b b b b 0 0 0 0
COPY - NO CORRECTIONS WHILE SEARCHING	[illegible]	s	t	0	"	"	word No.		b b b b (limit)
SEARCH - 1 WORD in BLOCKS or BUCKETTES		s	=0 no ≠0 blkNo	0 SCP u T_u	"	=0 blk ≠0 blk	word No.		b b b b
SKIP ON T_s FORWARD	F	[illegible]	0	[illegible]	0	0	0	0	b b b b
BACKWARD	B	[illegible]	0	0	0	0	0	0	b b b b
REWIND TAPES & RESET INTERLOCK	R	s		s	s	s	s	s	s s 0 0
MASTER CLEAR, INITIAL READ T_s	1	s	=0 zero ≠0 stop (clear)	0	0	0	0	0	0 0 0 0
LOAD HERE COPY (g words)	[illegible]	0	t	0	0	0 2 } t 8	0	0	8 0 0 0
PRINTOUT or [illegible] ([illegible])	5	[illegible]	0	m	m	m	00, 10, 50, or 5n		[illegible]
STOP	7	9	9	9	9	9	9	9	9 9 9 9
NO OPERATION	0	0	0	0	0	0	0	0	0 0 0 0
H.S.R. CORRECT	[illegible]	[illegible]	[illegible]	[illegible]	0	0	0	0	0 0 0 0

SUMMARY OF SUB-CONTROL WORDS

COPY WITH CORRECTIONS (X s t u $r_s r_t$ d c b b b b)

Sub-control words

n n x x x x 0 0 0 0 y y	n n	Number of consecutive corrections listed under the sub-control word. nn=00 or 01 for a single word correction. Consecutive corrections must be within block number listed.
	x x x x	Block number.
	y y	Starting word number for corrections.
Z Z Z Z Z Z Z Z Z Z Z Z		End of corrections.

WORD CHANGER (W s t u $r_s r_t$ d c b b b b)

Sub-control words - For 1/2 word operation.

CCC OLD EEE NEW	CCC	OPR if digits 1-3 and 7-9 (operation part of instruction word) is to be examined.
	CCC	ADR if digits 4-6 and 10-12 (address part of instruction word) is to be examined.
	OLD	Three digit quantity (of operation or address) on which the search is made.
	EEE	Extractor pattern used to determine the equality of the old pattern and the insertion of the NEW. If all three digits are to be examined EEE=111.
	NEW	Three digit quantity which is used (in conjunction with EEE) to replace the old pattern.
ZZZ ZZZ ZZZ ZZZ		End of list for modification.

Sub-control words - For full word operation.

CCC OLD EEE NEW	CCC	FULL∆WORD∆∆∆12 digits for full word control.
	OLD	( )12 digits of old word.
	EEE	( )12 digits extractor.
	NEW	( )12 digits for new word.
		Comparisons and substitutions are made by extractor pattern on full word. If all digits of OLD are to be compared, extractor must be all ones.
ZZZ ZZZ ZZZ		End of list for modification.

A maximum of 47 sub-control function words (OPR, ADR, of FULL WORD) may be operated on at any time. Both full and 1/2 word controls may appear in the same list. (Each sub-control set is separated into 10 word items and 47 such items, not including ZZZ ZZZ ZZZ ZZZ, may be used in a single list.

The sub-control function is determined by examination of only the 2nd digit in CCC.

O<u>P</u>R, A<u>D</u>R, F<u>U</u>L

SEARCH, THEN COPY WITH CORRECTIONS (Q s t 0 $r_s r_t$ w w b b b b)

w w > 59	All words are searched.
w w ≤ 59	Only word specified is searched.
b b b b	Number of blocks to be copied with corrections.

Follow the control word with a 12 digit QUANTITY on which the search is made. The search is made on all 12 digits. To continue search after initial agreement of QUANTITY and searched data, SET BKPT 1, FORCE TRANSFER.

Search, then copy - bbbb is the number of blocks to copy starting with the block containing the searched word. Follow the QUANTITY with the sub-control words for COPY WITH CORRECTIONS.
Search, without copy - bbbb is zero. Follow the QUANTITY with the next control word.

COPY - NO CORRECTIONS WHILE SEARCHING (K s t 0 $r_s r_t$ w w b b b b)

w w > 59	All words are searched.
w w ≤ 59	Only word specified is searched
b b b b	Upper limit of blocks copied while searching. If bbbb is reached without finding the search word, an appropriate printout will occur and the operation is terminated.

Follow the control word with a 12 digit QUANTITY on which the search is made. The search is made on all 12 digits.
To continue copy and search after initial agreement of QUANTITY and searched data, SET BKPT 1, FORCE TRANSFER.
Copy - to and including the block containing the search word.

LINE MERGE (COPY by WORDS) (L 0 t 0 0 r_t 0 0 0 0 0 0)
Sub-control words

M BLK WD s blk wd	M	Control Digit - Copy.
	BLK	Starting block number.
	WD	Starting word number within block BLK.
	s	Input servo number.
	blk	Ending block number.
	wd	Last word in block blk copied.
A x 0 0 0 0 0 0 0 n n n n	A	Control digit - Add.
	x	x=0 Add nnnn words of zeros to output.
		x=Z " " " " all Z's to output.
		x=X " " " " the contents of which is specified by the QUANTITY which directly follows the sub-control word, to the output.
A x 0 0 0 0 0 0 0 0 0 0		Fill to the end of the current output block with the quantity specified by x.

63

T 0 0 0 0 0 0 0 n n n n	T	Control digit - Transfer.
	n n n n	Transfer the next nnnn words from the control tape or manual typein to the output block and tape.
Z Z Z Z Z Z Z Z Z Z Z Z		End of Line Merge. Failure to complete the information in the last output block will cause the computer to print out "BLOCK NOT FULL" and ask for a typein of the QUANTITY to be used as a fill.

When the LINE MERGE control word is specified, the routine assumes that all 10 servos on the computer are (figuratively speaking) about to read block 1. The routine remembers the number of blocks read from ALL servos. The tapes will read in the forward or backward direction to locate the starting block number, depending upon previous operation on servo 6. Information may be merged from many input tapes to a single output tape. No restrictions are placed on the sequence of data merged, except - the starting block and word number must be equal to or less than the ending block and word number for any single "M" sub-control word. Information is copied with the tape moving in the forward direction.

16 July 1956

Talmadge

USEful Note #11 29 April 1957

SUBJECT: Octal Card Load and Octal Dump

CONTRIBUTOR: HO

1. Identification HOSR24, OCTAL CARD LOAD
Robert G. Tantzen, 12 Apr 57
Holloman Air Development Center
1103A-Service Routine

2. Purpose
To read any number of octal cards and store their contents on core or drum.

3. Method
a. This is a service routine with manual and program entry. It bootstraps itself into core, and restores core upon exit.
b. Each card is handled as a unit. The words are assembled in temporaries and then block-transferred to their destination.
c. Checks are made to assure that:
  1. The card contains the identification punch
  2. The address is punched correctly
  3. Each word is punched correctly
d. Under all circumstances the core is completely restored, including 00000, which need not have an MJ.

4. Usage
a. Program entry is effected with the instruction
  RJ HOSR24+2, or, in abs. 37 44002 44000.
  Manual entry is done by starting at 40002.
b. The routine exits upon finding a card without identification punch. On manual operation then PAK = 40002.
After a successful exit (A)=0,(Q)=1.
c. Space required (on drum) 88 cells. (HO-library:44000-44130).
d. Error indications
Address: A correct address must have 5 octal digits. If not 5, or if an 8 or 9 appears, the typewriter prints "a". Card is not loaded, routine exits.

Words: Each word field on the card must have 12 octal digits punched, or be completely blank. If less than 12 digits, or 8's and 9's appear, typewriter prints "w". Card is not loaded, routine exits, computer stops with IO-fault. If a column is double punched, the higher digit will be read.
e. Input cards
The first card must be in reading position. If the routine is to be used repeatedly, the individual sections to be loaded must be seperated by one blank card. Cards in each section may be in any order. Place 3 empty cards at end of whole card deck.

f. Card Format

Column:	1-12	first word in octal
	13-24	2nd word
	25-36	3rd word
	37-48	4th
	49-60	5th
	61-72	6th word
	73-77	insert address, 5 digits
	80	a 5, identification punch

The address belongs to the first word, the others go into consecutive cells. If less than 6 words are needed, the fields not used must be left blank. There must always be a first word. Whenever a blank word field is detected, the information found so far is stored and the routine reads the next card.

5. Restrictions
   a. Attempt to load into cells exceeding the core capacity will cause an SCC-fault.
   b. Cells 77000-77200B are used for image purposes and should not be loaded, because they will be block-transferred back to 00000-00200B.

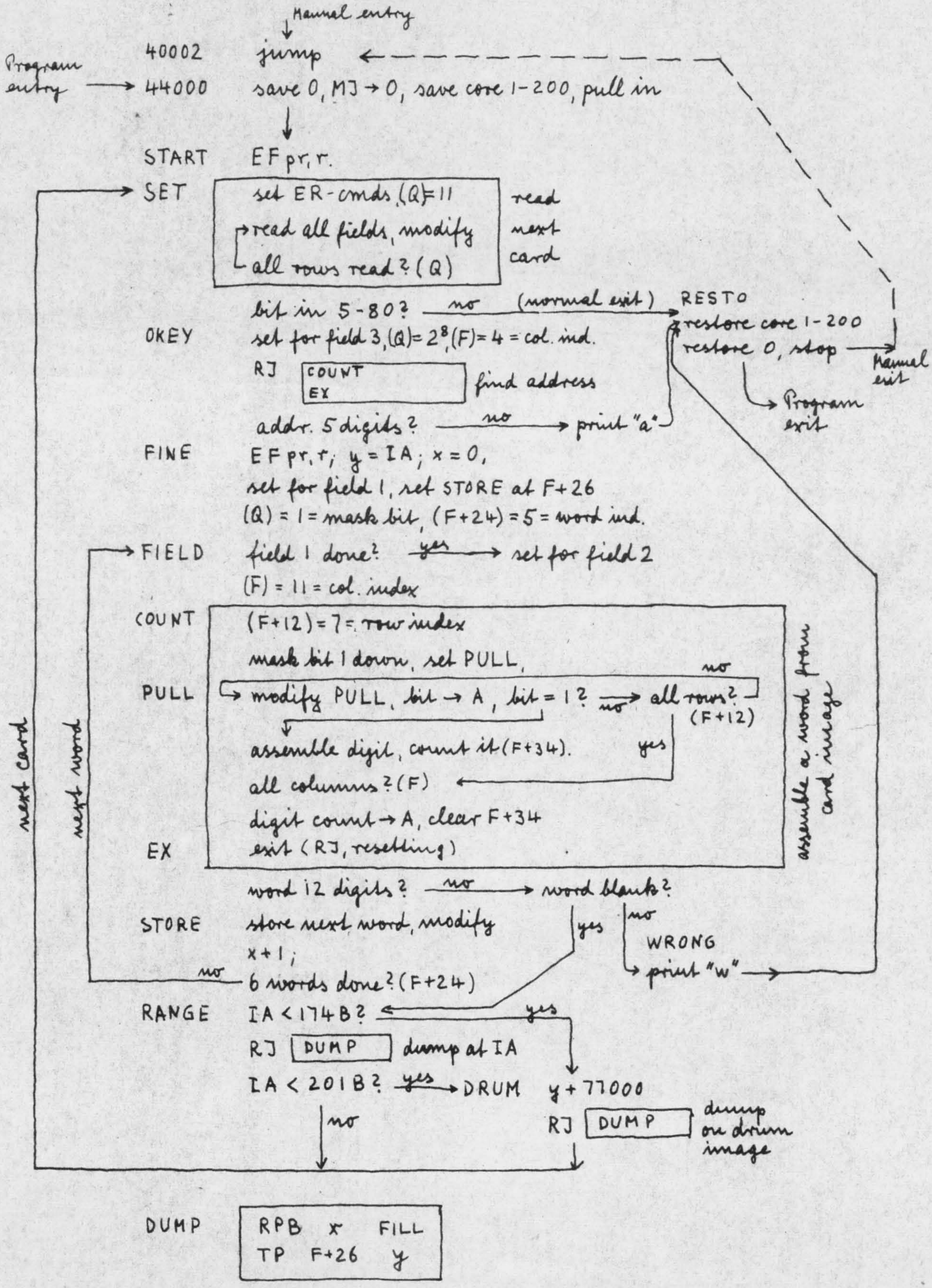
6. Coding Information
   a. Timing: Cards are processed at full Bull speed, 120 cards/min.
   b. Status: Routine is completely checked out.

LOC	OP	U-ADDR		V-ADDR		REMARKS
	B		4			HOSR24
	X		79			OCTAL CARD LOAD
	X		11			APRIL 57
	X		35			MEDLEY, TANTZEN
	X					SERVICE ROUTINE W
	X					MAN AND PROG ENTRY
	MJ			44004B		ENTRANCE
	TP	77000B				RESET 000
	RJ	44002B		44003B		PROGRAM EXIT
	MS			40002B		MANUAL EXIT
	TP			77000B		SAVE 0000
	TP	44000B				MJ TO 000
	RPB	200B		44010B		SAVE
	TP	1		77001B		CORE
	RPB	117B		START		PULL IN
	TP	44012B		START		ROUTINE
START	EF	ADDR+	1	M+	4	PR,R
SET	TV	FIELD-	1	READ		SET
	TV	FIELD+	1	READ+	1	READ
	TV	COUNT		READ+	2	COMMANDS
	TP	M		Q		ROW INDEX = 11
READ	ERA	F+	25	FILL		READ
	ERB			FILL		A
	ERB			FILL		ROW
	RPU	3		READ+	5	MODIFY FOR
	RA	READ		C		NEXT ROW
	IJ	Q		READ		12 ROWS? NO
	TN	A		Q		Q = 1
	QT	F+	28	A		5 COL 80 TO A
	ZJ	OKEY		RESTO		5 COL 80? YES,NO
OKEY	TU	READ+	4	COUNT+	2	SET FOR FIELD 3
	LQ	Q		8		BIT 8 UP
	TP	C+	2	F		COL INDEX = 4
	RJ	EX		COUNT		DECODE ADDRESS
ADDR	EJ	C+	3	FINE		ADDR CORRECT? YES
	PR	F+	13	ROW+	2	PRINT -A-
RESTO	RPB	200B		44001B		RESTORE CORE
	TP	77001B		1		AND EXIT
FINE	EF			M+	4	PR,R
	TV	F+	35	DUMP+	1	SET DUMPADDR = IA
	TU	WRONG		DUMP		SET DUMP WORDS
	TU	FINE+	4	COUNT+	2	SET FOR FIELD 1
	TV	K+	2	STORE		SET STORE AT F+26
	TP	C		Q		SET MASK BIT = 1
	TP	C+	3	F+	24	WORD INDEX = 5
FIELD	EJ	C+	1	F2		SET FIELD 2 ? YES
	TP	M		F		COL INDEX = 11
COUNT	TP	C+	4	F+	12	ROW INDEX = 7

Octal Card Load

Flow Chart

HOSR24



HDRSD,

12 April 57, [illegible]

1. Identification HOSR14, OCTAL CARD DUMP
Paul D. Medley, 15 April 1957
Holloman Air Development Center
1103A-Service Routine

2. Purpose
To dump the contents of any number of consecutive cells on octal cards.

3. Method

a. This is a service routine with manual and program entry. It bootstraps itself into core, and restores core upon exit, including cell 00000.

b. The words to be dumped are transferred into temporary storage and punched out 6 words per card. This enables the Bull to operate at maximum speed.

4. Usage

a. Program entry is effected by:

LOC	OP	U	V	REMARKS
y-2	TP	FA	Q	First addr to Q
y-1	TP	LA	A	Last addr to A
y	RJ	HOSR14+2	HOSR14	
y+1	NORMAL RETURN			

at y+1 A and Q = 0

b. Manual entry is effected by:

Set first address in Q_u
Set last address in $A_{(R)(u)}$
Start at 40003

c. The routine exits when the region has been punched. For manual entry PAK = 40003, A and Q = 0.

d. Space required (on drum) 108 cells. (HO-library 44131 -44327).

e. Card format:

Column:	1-12	first word in octal
	13-24	second word
	25-36	third word
	37-48	fourth word
	39-60	fifth word
	61-72	sixth word

73-77 address, 5 digits
80 a 5 for identification

The address is associated with the first word, the other words are punched from consecutive cells. If less than 6 words are on a card the remaining columns for words are left blank. A zero word is punched all zeros.

f. Cell 00000 may be dumped.

g. A MJ for cell 00000 is not required.

h. Cards punched with this routine may be reloaded with HOSR24.

i. The Bull need not be cycled before punching and the channels are cleared after punching.

5. Restrictions

a. Any attempt to dump an illegal address causes a SCC fault, or produces meaningless cards.

b. Dump of A and Q is not permissable.

c. Cells 76000B-77777B should not be dumped because they are used as core image. If they are dumped the cards have image addresses for information that was in 00000-01777B before entry.

6. Coding Information

a. Timing: Cards are punched at maximum Bull speed except for a 1/4 sec. delay after each 128 cards.

b. Status: This routine has been completely machine checked.

8

LOC	OP	U-ADDR		V-ADDR		REMARKS
	LQ	Q		35		BIT 1 RIGHT
	TU	FILL		PULL		SET FIELD ADDRESS
	RA	PULL		M+	3	ADVANCE PULL CMD
PULL	QT	FILL		A+	31B	IMAGE BIT TO A
	ZJ	PULL+	2	ROW		BIT IN ROW? YES,NO
	SP	F+	35	3		ASSEMBLE NEXT
	AT	F+	12	F+	35	OCTAL DIGIT
	RA	F+	34	C		COUNT 1 DIGIT
	MJ			ROW+	1	JUMP
ROW	IJ	F+	12	PULL-	1	8 ROWS TRIED? NO
	IJ	F		COUNT		ALL COLS? NO
	TP	F+	34	A+	30B	DIGIT COUNT TO A
	LTL			F+	34	CLEAR DIGIT COUNT
EX	RJ	EX		EX+	1	ON ADDR GO TO ADDR
	EJ	M+	1	STORE		WORD CORRECT? YES
	ZJ	WRONG		RANGE		WRONG OR BLNK WORD
STORE	TP	F+	35	FILL		STORE NEXT WORD
	RA	STORE		C		ADVANCE STORE
	RA	DUMP		M+	3	ADVANCE WORD COUNT
	IJ	F+	24	FIELD		6 WORDS READ? NO
RANGE	TP	DUMP+	1	A		TP TO A
	TJ	K		DRUM		IA UNDER 174B?YES
	RJ	DUMP		DUMP		DUMP AT IA
	TJ	K+	1	DRUM		IA UNDER 2 1B? YES
	MJ			SET		GO READ NEXT CARD
DRUM	RA	DUMP+	1	M+	2	IA + 7700 B
	RJ	DUMP		DUMP		DUMP ON DRUM
	MJ			SET		GO READ NEXT CARD
DUMP	RPB			FILL		DUMP INFORMATION
	TP	F+	26	FILL		INTO MEMORY
F2	TU	START		COUNT+	2	SET FOR FIELD 2
	MJ			FIELD+	1	JUMP
WRONG	PR	30000B		PULL		PRINT -W-
	MJ			RESTO		GO OUT
K	TP	F+	26	174B		TEST
	TP	F+	26	201B		DUMMIES
		F+	1	F+	26	
C				1		ONE
				2		TWO
				4		FOUR
				5		FIVE
				7		SEVEN
M				11		
				12		TWELVE
				77000B		FIRST IMAGE ADDR
		1				U-MOD
	40			5		PR,R CONST
F	RESERV	36		36		135
	END					201 9 APR 57

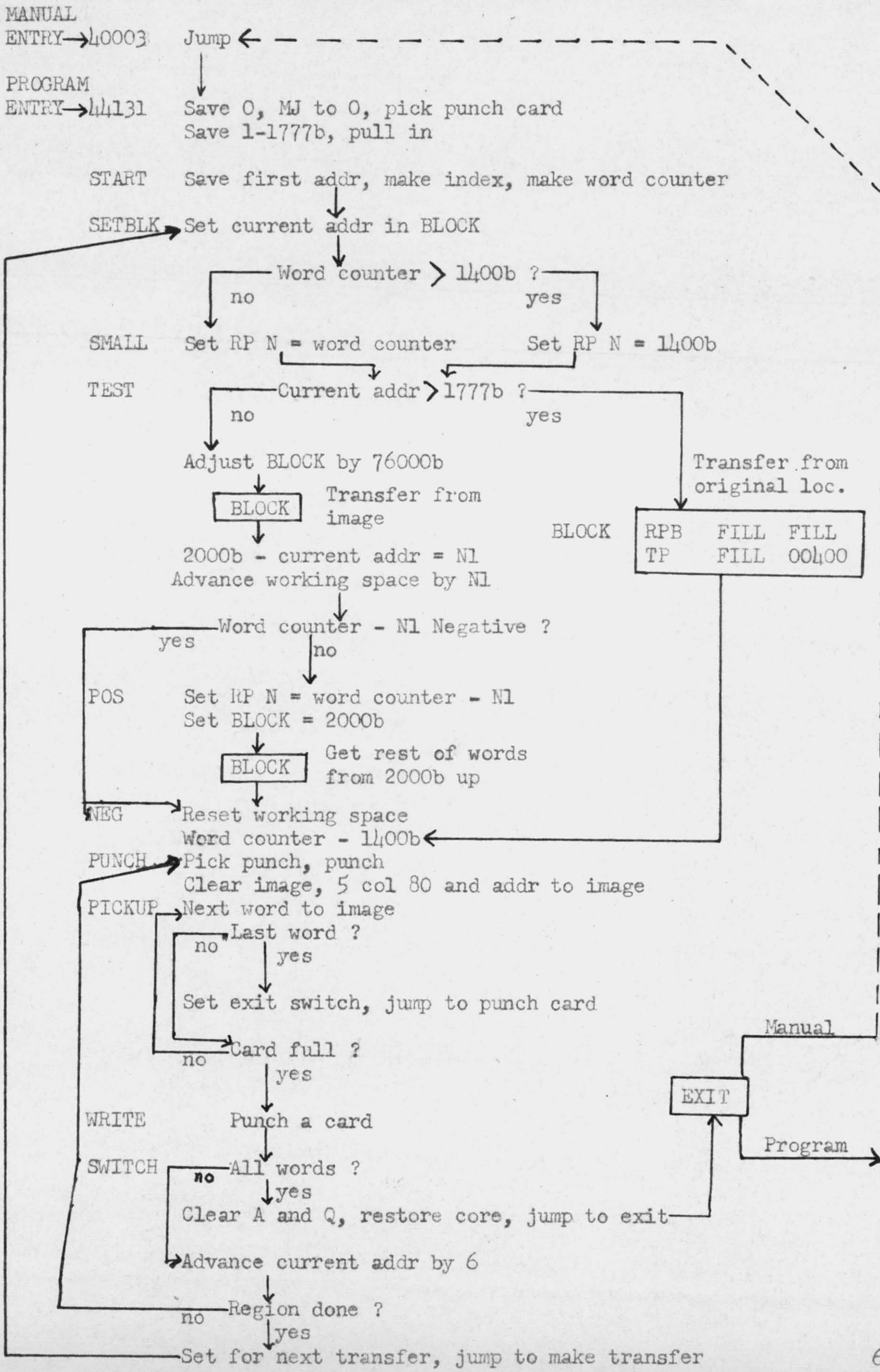
LOC	OP	U-ADDR		V-ADDR		REMARKS
	B		15			HOSR14
	X		1 1			OCTAL CARD DUMP
	X		19			DUMPS ALL ADDRS
	X		42			12 WORD P S
	X					MEDLEY,TANTZEN
	X					REVISED 4,57
	MJ			44135B		ENTRANCE
	TP	76000B				RESET 0
	RJ	44133B		44134B		PROGRAM EXIT
	MS			40003B		MANUAL EXIT
	EF			44276B		PICK PUNCH
	TP			76000B		SAVE 0
	TP	44131B				MJ TO 0
	RPB	1777B		44142B		SAVE CORE
	TP	1		76001B		1 TO 1777B
	RPB	110		START		PULL IN
	TP	44144B		START		PROGRAM
START	TP	Q		T		SAVE FIRST ADDR
	ST	Q		A		LA-FA
	LTL	21		A		STORE
	TP	A		T+	1	WORD INDEX
	SA	C1		15		MAKE WORD COUNT
	TP	A		T+	2	SAVE WC
SETBLK	TU	T		BLOCK+	1	SET CA
	TJ	C1400		SMALL		WC UNDER 1400? YES
	TP	K		A		MAKE X FOR RP
	AT	C1400		BLOCK		= 1400
	MJ			TEST-	1	
SMALL	AT	K		BLOCK		MAKE X OFRP =WC
	TP	C1777		A		REGION END TO A
TEST	TJ	T		GO		CA OVER 1777 ? YES
	RA	BLOCK+	1	C76000		ADJUSTBLOCK PICKUP
	RJ	BLOCK		BLOCK		MAKE TRANSFER
	TP	C2000		A		REGION END TO A
	ST	T		Q		2000-CA TO Q
	LA	A		57		N1 TO AV
	AT	TK		BLOCK+	1	SET WS
	TP	T+	2	A		N TO A
	ST	Q		A		N-N1 = X
	SJ	NEG		POS		N-N1 NEG. YES, NO
POS	AT	K		BLOCK		X TO RP
	RJ	BLOCK		BLOCK		MAKE TRANSFER
NEG	TP	TKS		BLOCK+	1	RESET WS
	MJ			PUNCH-	1	
GO	RJ	BLOCK		BLOCK		TRANS IF OVER 1777
	RS	T+	2	C1400		WC-1400
PUNCH	EF			PPP		PICK PUNCH, PUNCH

LOC	OP	U-ADDR		V-ADDR		REMARKS
	RPV	36		AFT		CLEAR
	TP	C0		FIELD		IMAGE
AFT	TP	C1		FIELD+	31	5 COL 80 TO IMAGE
	TP	C400		T+	3	PLACE ADDR BIT
	LQ	T		Q+	21	CA TO QV
	TP	C4		T+	4	DIGIT INDEX =4
	TU	K4		K1		SET FLD III ADDR
	TP	C1		FIELD+	1	SET ADDR TEST
	TP	Q		T+	5	WORD TO SHIFTER
SHIFT	LQ	T+	3	35		BIT 1 RIGHT
	LQ	T+	5	3		POSIT NEXT DIG
	QT	C7		A		DIGIT TO A
	AT	K1		PLACE		MAKE CMD
PLACE	FILL	FILL		FILL		CC FILL T 3
	IJ	T+	4	SHIFT		WORD FINISHED ? NO
	IJ	FIELD+	1	ADDR		JUMP ONLY ON ADDR
	IJ	T+	1	MORE		LAST WORD ? NO
	RJ	SWITCH		WRITE-	3	GO PUNCH LAST WORD
	RS	Q		A		CLEAR A AND Q
	RP	3		BLOCK-	2	CLEAR
	EF			SB		BULL
	RPB	1777B		44132B		RESTORE
	TP	76001B		1		CORE
BLOCK	FILL	FILL		FILL		RPB FILL FILL
	TP	FILL		400B		BLOCK TRANSFER
MORE	IJ	T+	6	PICKUP		FIELD DONE ? NO
	TU	K3		K1		SET FLD II
	IJ	T+	7	PICKUP-	1	FLDII DONE ? NO
	TP	C8		T+	6	ROW INDEX =11
	RPB	3		WRITE		SET EWS
	TV	K2		WRITE		CMDS
WRITE	EWA			FILL		WRITE
	EWB			FILL		A
	EWB			FILL		ROW
	RPU	3		WRITE+	5	MODIFY FOR FMR
	RS	WRITE		C1		NEXT ROW
	IJ	T+	6	WRITE		ALL ROWS ? NO
SWITCH	RJ	SWITCH		SWITCH+	1	ALL WORDS SWITCH
	RA	T		C60		ADV CA BY 6
	TP	PICKUP		A		REGION
	TJ	BKB		PUNCH		FINISHED ? NO
	TP	T+	2	A		SET FOR NEXT
	TU	CU		PICKUP		RESET PICKUP
	MJ			SETBLK		GO FOR NEXT TRANS
ADDR	TU	K2		K1		SET FLDI ADDR
	TP	C1		T+	7	SET GET FLD II
	TP	C1		T+	3	SET IMAGE BIT

LOC	OP	U-ADDR		V-ADDR		REMARKS
	TP	C9		T+	6	WORD INDEX =2
PICKUP	LQ	400B		Q+	15	POSITION WORD
	TP	C8		T+	4	DIGIT INDEX =11
	RA	PICKUP		C100		MODIFY PICKUP
	MJ			SHIFT-	1	GO MAKE IMAGE
TK	TP	2000B		400B		TEST+6
TKS	TP	.FILL		400B		NEG
BKB	LQ	2000B		Q+	15	SWITCH+3
K	RP	FILL		FILL		BLOCK TRANS CON
K1	CC	FILL		T+	3	PLACE
K2		FIELD+	2	FIELD+	35	
K3		FIELD+	14	FIELD+	11	
K4		FIELD+	26	FIELD+	23	
PP	40			10B		
PPP	40			12B		PPP
SB	40					
C1				1		
C4				4		
C8				11		
C100	0	1				
C9				2		
C0						
C60	0	6				
C400				400B		
CU	0	400B				
C7	0	7				
C1400		1400B				
C1777		1777B				
C76000		76000B				
C2000		2000B				
T	RESERV	8		8		
FIELD	RESERV	36		36		
	END					PUNCHED BY WELCH

OCTAL CARD DUMP
HOSR14

FLOW CHART



Talmadge

USEful Note #13 1 July 1957

SUBJECT: Parity Error Routine

CONTRIBUTOR: HO

Useful Note No.

1. Identification

HOSP11, PARITY ERROR ROUTINE
Robert G. Tantzen, 11 June 1957
Holloman Air Development Center
1103A Service Routine

2. Purpose

To recover from parity errors when reading magnetic tape, in fixed block mode, without computer stop.

3. Method

This is a service routine with program entry only. It can handle all cases where reading is done with a 120 times repeated ERB. So the main program may read forward or backward, free run or one block only. Re-reading is tried first on high, then on low bias. If one or the other attempt was successful, main program continues, the bias being reset to normal.

4. Usage

To use this routine, a calling sequence of three commands has to be inserted in the main program; this is two more than needed normally. A typical main program with calling sequence is given:

LOC	OP	U	V	REMARKS
KICK	EF	0	K	Start read
	⋮			
	RPV	120	READ+1	
READ	ERB	0	XXX	Read 1 block
	ERA	0	A	
	ZJ	BAD	GOOD	Parity error? yes, no
BAD	TP	KICK	HOSP11+3	Place controls
	TP	READ	HOSP11+4	
	RJ	HOSP11+2	HOSP11	Go to parity error routine
GOOD				Block read correctly

If both attemps to re-read are unsuccessful, typewriter prints P TUX, where X is the tape unit number. The computer comes to a PS stop.

5. Restrictions

a. Main program cannot use cells 07773-07777, and 77702-77777(image).

b. Contents of Q is preserved, the main program may not read tape information into Q.

c. Space needed in drum library = 62 cells. (HO-library 43610-43705)

HOSP11 -

Flow Chart — Parity Error Routine

from main program

↓

Save core 7702-7777 at 77702-77777

Pull in routine

ENT — Restore control cells on drum

Set bias high, place READ-Cmd.

SET — Pick and save EF-constant from main progr.

Reading free? (no → Place exit address)

Set EXIT to give no final EF

Place exit address

Save (Q)

Make move and read constants

Move tape 1 block, 35 ms delay

READ — Read 1 block and stop

Parity error? (no → GOOD)

↓ yes

Set bias low, 35 ms delay

Was this first attempt? (yes → Move tape 1 block, 35 ms delay)

Print PTU on typewriter

Make print cmd

UNIT — Print TU-number

Restore core 7702-7777

Program Stop.

GOOD — Restore Q

Restore core 7702-7772

Set bias normal

35 ms delay

EXIT — jump out / give original EF jump out

↓

back to main program

```
          B            7702                   HOSP11                7674  00 00000 07702   001   SP11
          X              61                   REREAD MAGN TAPE      7675  00 00000 00075   002   SP11
          X               1                   UPON PARITY ERROR     7676  00 00000 00001   003   SP11
          X               0                   SERVICE ROUTINE       7677  00 00000 00000   004   SP11
          X               0                   28 MAY 57             7700  00 00000 00000   005   SP11
          X               0                   ROBERT G TANTZEN      7701  00 00000 00000   006   SP11
BIAS      HJ      0            43610B+    5   ENTRY                 7702  45 00000 43615   007   SP11
          PS      0                  0        ALARM STOP            7703  57 00000 00000   008   SP11
CORE      0       0                  0        CONTROL, EXIT ADDR    7704  00 00000 00000   009   SP11
                                                  EF-CMD            7705  00 00000 00000   010   SP11
                                                  ER-CMD            7706  00 00000 00000   011   SP11
          RPB   76B            43610B+    7   SAVE                  7707  75 30076 43617   012   SP11
          TP   BIAS            77702B          CORE                 7710  11 07702 77702   013   SP11
          RPB   76B               ENT         PULL IN               7711  75 30076 07730   014   SP11
          TP 43610B              BIAS          ROUTINE              7712  11 43610 07702   015   SP11
HIGH      02      1            70000B         HIGH                  7713  02 00001 70000   016   SP11
 LOW      02      1            60000B         LOW BIAS CONST        7714  02 00001 60000   017   SP11
NORM      02      1            50000B         NORM                  7715  02 00001 50000   018   SP11
          2     20B               52B         FLEX 1, TEST NUMD     7716  02 00020 00052   019   SP11
          2     10B            70074B              2                7717  02 00010 70074   020   SP11
                                  70B              3                7720  00 00000 00070   021   SP11
                                  64B             4                 7721  00 00000 00064   022   SP11
                                  62B              5                7722  00 00000 00062   023   SP11
                                  66B              6                7723  00 00000 00066   024   SP11
                                  72B              7                7724  00 00000 00072   025   SP11
   K      PR      0               LOW         PRINT DUMMY           7725  61 00000 07714   026   SP11
REDE           602B                                                 7726  00 00602 00000   027   SP11
MOVE            14B                 1                               7727  00 00014 00001   028   SP11
 ENT      RPU     3              ENT+     2   CLEAR                 7730  75 20003 07732   029   SP11
          RS 43610B+     2          A          CONTROL CELLS        7731  23 43612 32000   030   SP11
          EF      0            43610B+    9   SET HIGH BIAS         7732  17 00000 43621   031   SP11
          TP   CORE+     2       READ         PLACE READ CMD        7733  11 07706 07752   032   SP11
          TV   CORE+     1        SET         GET                   7734  16 07705 07735   033   SP11
 SET      CC   EXIT+     2       FILL          EF CONST             7735  27 07777 30000   034   SP11
          TJ   NORM+     1        SET+    3   READ FREE? YES        7736  42 07716 07740   035   SP11
```

```
        TP    BIAS          EXIT         PREVENT FINAL EF      7737  11 07702 07775  036  SP11
        TV    CORE          EXIT         SET EXIT ADDR         7740  16 07704 07775  037  SP11
        TP       Q          CORE         SAVE Q                7741  11 31000 07704  038  SP11
        TP    NORM+     2     Q+    15B  MASK TO Q             7742  11 07717 31015  039  SP11
        QT       A            Q+     4   SELECTED BITS TO Q    7743  51 32000 31004  040  SP11
        CC    MOVE            Q+     1   MAKE MOVE CONST       7744  27 07727 31001  041  SP11
        CC    REDE            Q+    34B       READ             7745  27 07726 31034  042  SP11
        EF       0          MOVE         MOVE 1 BLOCK          7746  17 00000 07727  043  SP11
        TP       D             D         DELAY                 7747  11 40000 40000  044  SP11
KICK.   EF       0          REDE         READ                  7750  17 00000 07726  045  SP11
        RPV    120          READ+    1    ONE                  7751  75 10170 07753  046  SP11
READ     0    FILL          FILL              BLOCK            7752  00 30000 30000  047  SP11
        ERA      0             A                               7753  76 00000 32000  048  SP11
        ZJ     BAD          GOOD         PARITY ERR, YES,NO    7754  47 07755 07770  049  SP11
BAD     EF       0           LOW         SET LOW BIAS          7755  17 00000 07714  050  SP11
        TP       D             D         DELAY                 7756  11 40000 40000  051  SP11
        RJ     BAD+     2   KICK-    2   TRY LOW? YES          7757  37 07757 07746  052  SP11
        RPV      4          BAD+     5   PRINT                 7760  75 10004 07762  053  SP11
        PR       0          SET+     5        P TU             7761  61 00000 07742  054  SP11
        SP    MOVE            21         GET TU NUMBER         7762  31 07727 00025  055  SP11
        SP       A            39              FOR PRINTING     7763  31 32000 00047  056  SP11
        AT       K          UNIT                               7764  35 07725 07765  057  SP11
UNIT     0    FILL          FILL         PRINT IT              7765  00 30000 30000  058  SP11
        RPB    76B        43610B+    1   RESTORE CORE          7766  75 30076 43611  059  SP11
        TP  77702B          BIAS          AND GO TO STOP       7767  11 77702 07702  060  SP11
GOOD    TP    CORE             Q         RESTORE Q             7770  11 07704 31000  061  SP11
        RPB    71B          GOOD+    3    AND CORE             7771  75 30071 07773  062  SP11
        TP  77702B          BIAS              EXCEPT LAST 5    7772  11 77702 07702  063  SP11
        EF       0          NORM         SET BIAS NORMAL       7773  17 00000 07715  064  SP11
        TP       D             D         DELAY                 7774  11 40000 40000  065  SP11
EXIT    RP       1          FILL         EXIT WITH OR          7775  75 00001 30000  066  SP11
        EF       0.         EXIT+    2    WITHOUT EF           7776  17 00000 07777  067  SP11
                                         TEMP EF-CONST         7777  00 00000 00000  068  SP11
        END                                                             3 05027  069  SP11
```

Talmadge

USEful Note #12 20 May 1957

SUBJECT: Boeing 1103A Service Routine Library

CONTRIBUTOR: BA

TABLE OF CONTENTS

BOEING 1103A SERVICE ROUTINE LIBRARY

GENERAL DESCRIPTION

During the course of program checkout the programmer often has need of information which is not normally supplied as a direct result of the operation of the program. The information required may be in a variety of forms and include such items as octal instructions, intermediate results in floating or stated point, contents of registers, or statements in English. It is expected that the programmers will provide for all foreseeable difficulties by programming linkages to standard subroutines to provide the required information. The diagnostic can be used in this fashion to provide listings of instructions in octal or direct statements regarding difficulties encountered. Output subroutines can be used to provide listings of intermediate results.

Because of the extremely large number of contingencies which might arise it is often impractical (if not impossible) for the programmer to provide for all such. This is particularly true of machine malfunctions. This Service Routine Library is made available to provide for such contingencies. Included herein are the service routines most often required to provide the programmer with the information necessary to diagnose his difficulty and to assist the operator in the diagnosis of machine malfunctions. It should be noted that most of these routines are a simple rework of pre-existing routines to fit on the dead space of the Boeing 1103A computer. The routines were adapted from the Central Exchange Newsletters published by

GENERAL DESCRIPTION

Remington Rand or were taken from the customer engineers service routine library. Any or all of the service routines may be used by the programmer provided the operator is given a written set of instructions describing the routine to be used and how it is to be used.

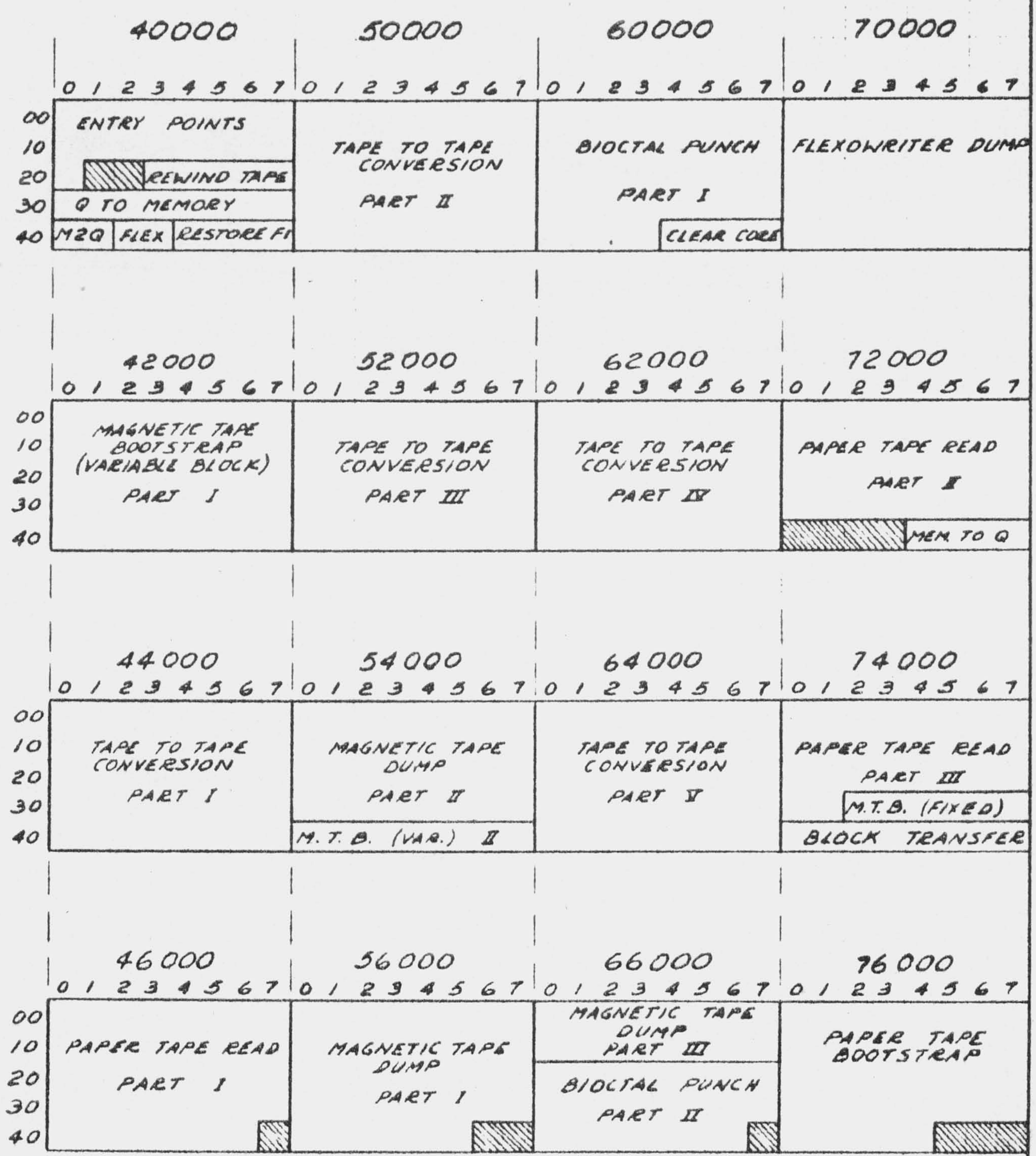
The routines included in the library are packed with a 4 word interlace on the dead space of the drum where they cannot be inadvertently destroyed. Because of the limited space available the output of the routines is of the simplest form (octal instructions) and checking is kept to a minimum. Except as otherwise noted the routines are written to operate in the first part of core which is normally reserved for bootstrapping operations, the diagnostic and the tape read or write subprogram. The general procedure to initiate the use of a particular service routine is as follows:

1. Set Drum to ABNORMAL.
2. Set PAK equal to entry point of desired routine.
3. Press START key. Computer will stop at location which indicates successful transfer into core (or successful execution of the routine in the case of a routine which operates from the dead space).
4. Set Drum to NORMAL.
5. Enter parameters on console.
6. Turn on auxiliary equipment required (if any).
7. Press START key.

GENERAL DESCRIPTION

A map of the drum dead space can be found on page 1.0. The remainder of the document contains a complete description of each service routine and its computer code in octal.

1103A SERVICE ROUTINES
DEAD SPACE DRUM MAP



CALC	D. Cook	4-57	REVISED	DATE	BOEING 1103A SERVICE ROUTINE LIBRARY
CHECK	L. Cook				D2-1381
APR					
APR					BOEING AIRPLANE COMPANY SEATTLE 24, WASHINGTON

MAGNETIC TAPE BOOTSTRAP (VARIABLE BLOCK)

PURPOSE: Read the first block from magnetic tape on servo number 1 into magnetic core beginning with location $_8(00240)$.

ENTRY:

1. Set Drum to ABNORMAL.
2. Set PAK = 40000.
3. Turn Uniservo #1 on.
4. Press START key. Computer will stop with PAK = 00240.
5. Set Drum to NORMAL.
6. Press START key to proceed.

STORAGE ASSIGNMENT:

1. This routine occupies "dead space" locations 42000 thru 42047 and 54040 thru 54047.
2. This routine works from MC locations 00000 thru 00044.

LIMITATIONS:

1. This routine assumes that the first block on magnetic tape was written in the variable block mode with an 18 bit check sum as the last word of the block.
2. This routine does not check for parity errors.

MAGNETIC TAPE BOOTSTRAP (VARIABLE BLOCK)

COMPUTER STOPS:

In the event that the reading of information from tape is incorrect as indicated by a check sum failure, this routine will attempt to re-read the information at high and low bias. In the event that the reading still fails to check the computer will stop with PAK = 00240 and MCT = 00034. In addition, the "low bias" status will be indicated. To ignore the check sum failure, set Drum to NORMAL and press the START key. To try reading again do a MASTER CLEAR and press the START key.

A successful reading of information from tape is indicated by the computer stopped with PAK = 00240 and MCT = 00035.

40000	45	00000	42000
42000	17	00000	42047
42001	75	30045	00003
42002	11	42003	00000
42003	45	00000	00003
42004	45	00000	00000
42005	37	00000	00000
42006	17	00000	00043
42007	23	00045	00045
42010	16	00035	00014
42011	76	00000	32000
42012	47	00010	00013
42013	43	00036	00013
42014	43	00037	00025
42015	45	00000	00032
42016	76	10000	31000
42017	11	31000	00240
42020	11	31000	00047
42021	51	00040	32000
42022	55	31000	31022
42023	52	00040	32000
42024	22	10000	00046
42025	35	00045	00045
42026	21	00014	00036
42027	45	00000	00006

42030	17	00000	00041
42031	31	00045	00000
42032	34	00046	00000
42033	43	00047	00035
42034	45	00000	00033
42035	17	00000	00041
42036	17	00000	00042
42037	37	00034	54040
42040	56	00000	00240
42041	00	00000	00001
42042	00	00000	00002
42043	00	00007	77777
42044	02	00600	00000
42045	02	00014	10001
42046	02	00062	10000
42047	02	00200	10000

54040	17	00000	54046
54041	37	00034	00003
54042	17	00000	42007
54043	37	00034	00003
54044	56	00000	00240
54045	02	00062	10000
54046	02	00001	60000
54047	02	00001	70000

PAPER TAPE BOOTSTRAP

PURPOSE: Read a bi-octal paper tape into the addressable memory.

ENTRY:

1. Set Drum to ABNORMAL.
2. Set PAK = 40001.
3. Press START key. Computer will stop with PAK = 00006.
4. Set Drum to NORMAL.
5. Turn on Ferranti Reader.
6. Press START key.

STORAGE ASSIGNMENT:

1. This routine occupies "dead space" locations 76000 thru 76044.
2. This routine works from MC locations 00000 thru 00041.

LIMITATIONS:

1. Information cannot be read from paper tape into MC locations 00000 thru 00041.
2. The following information must be punched in the tape in the order indicated.
    a. Single seventh level punch.
    b. Insert address (six frames with seventh level punch on third and sixth frames.

LIMITATIONS: (contd.)

c. Bi-octal computer words (six frames each with seventh level punch on sixth frame only - routine assumes frame immediately preceding first frame of word, i.e., last frame of preceding word, contained a seventh level punch).

d. Optional check address (a computer word with seventh level punch on fourth and sixth frames whose value is equal to the insert address plus the number of words read).

3. This program does not recognize the double seventh level punches used to indicate an end of tape and will continue to read tape until a FORCE STOP is executed from the console.

4. The reading of information destined for magnetic core only can be accelerated slightly by setting switch MJ3 which suppresses the Ferranti stop after every sixth frame.

COMPUTER STOPS:

A check address failure is indicated by the execution of a PROGRAM STOP(57) command.

40001	45	00000	76000
76000	75	30042	76002
76001	11	76003	00000
76002	56	00000	00006
76003	45	00000	76002
76004	45	30000	00003
76005	17	00000	00040
76006	11	00035	40000
76007	21	00003	00037
76010	45	30000	00007
76011	17	00000	00041
76012	76	00000	31000
76013	31	00035	00006
76014	52	00027	00035
76015	31	00036	00001
76016	52	00030	31000
76017	51	00030	00036
76020	43	00032	00001
76021	43	00031	00021
76022	43	00033	00023
76023	45	00000	00007
76024	16	00035	00003
76025	45	00000	00007
76026	11	00003	32000
76027	36	00034	32000

76030	43	00035	00007
76031	57	07070	70707
76032	00	00000	00077
76033	00	00000	17700
76034	00	00000	11100
76035	00	00000	10100
76036	00	00000	10500
76037	11	00035	00000
76040	00	00000	00000
76041	00	00000	00000
76042	00	00000	00001
76043	10	00001	00000
76044	10	00002	00000

PAPER TAPE READ

PURPOSE: Read a bi-octal paper tape into the addressable memory.

ENTRY:

1. Set Drum to ABNORMAL.
2. Set PAK = 40002.
3. Press START key. Computer will stop with PAK = 00011.
4. Set Drum to NORMAL.
5. Turn on Ferranti Reader.
6. Turn on Flexowriter.
7. Press START key.

STORAGE ASSIGNMENT:

1. This routine occupies "dead space" locations 46000 thru 46046 72000 thru 72037 and 74000 thru 74031.
2. This routine works from core locations 00000 thru 00137 and uses locations 00140 thru 01777 as erasable storage.

LIMITATIONS:

1. Information cannot be read from paper tape into MC locations 00000 thru 01777.
2. The following information must be punched in the tape in the order indicated.
   a. Single seventh level punch.
   b. Insert Address (six frames with seventh level punch on third and sixth frames).

LIMITATIONS: (contd.)

c. Bi-octal computer words (six frames each with seventh level punch on sixth frame only - routine assumes frame immediately preceding first frame of word, i.e., last frame of preceding word, contained a seventh level punch).

d. Optional check address (a computer word with seventh level punches on fourth and sixth frames whose value is equal to the insert address plus the number of words read).

e. End of tape (2 seventh level punches in successive frames) following the last block on the tape.

3. A check sum may be used to check the reading of information. The check sum is formed by performing a repeated SA command and consists of two computer words. In the paper tape it must be preceded by an insert address of $_8(00272)$ and followed by a check address of $_8(00274)$. Where used it must be punched following the block of information and preceding the insert address for the next block.

4. A block of information punched in a tape without a check sum cannot be followed by a block for which a check sum is supplied. Each check sum encountered is used to check the reading of all information following the last check sum encountered or the beginning of the tape.

PAPER TAPE READ

COMPUTER STOPS:

1. A check address failure is indicated by the letter "C" typed on the flexowriter and PAK = 00040. To resume reading, press START key. To reread tape, set PAK = 00011.

2. A check sum failure is indicated by the letter "M" typed on the flexowriter and PAK = 00011. Press START key to resume reading or reread tape.

3. An end of tape is indicated by a computer stop with PAK = 45000.

40002	45	00000	46000
46000	75	30040	46002
46001	11	46007	00000
46002	75	30040	46004
46003	11	72000	00040
46004	75	30032	46006
46005	11	74000	00100
46006	56	00000	00011
46007	45	00000	46002
46010	00	00000	00000
46011	00	00000	00000
46012	10	00002	00000
46013	10	00001	00000
46014	45	10000	00267
46015	45	00000	00011
46016	16	00174	00025
46017	16	00175	00064
46020	23	00131	00131
46021	23	00132	00132
46022	11	00112	00053
46023	17	00000	00003
46024	76	00000	31001
46025	31	00133	00006
46026	52	00113	00133
46027	31	00114	00001

72012	56	00000	00011
72013	11	00133	00300
72014	21	00053	00124
72015	31	00131	00044
72016	32	00132	00000
72017	32	00133	00000
72020	11	32000	00132
72021	22	00000	00131
72022	11	00062	00062
72023	11	00053	32000
72024	43	00125	00066
72025	45	00000	00015
72026	17	00000	00004
72027	31	00053	00000
72030	34	00112	00017
72031	35	00126	00075
72032	11	00115	32000
72033	42	00127	00103
72034	16	32000	00076
72035	75	30000	00077
72036	11	00300	00000
72037	11	00114	32000
74000	43	00117	00105
74001	43	00120	00107

```
74002  56 00000 45000
74003  21 32000 00130
74004  45 00000 00074
74005  21 00115 00111
74006  45 00400 00013
74007  16 00133 00115
74010  45 00000 00013
74011  00 00000 01500
74012  11 00133 00300
74013  00 00000 00077
74014  00 00000 00000
74015  00 00000 00000
74016  00 00000 17700
74017  00 00000 10100
74020  00 00000 11100
74021  00 00000 10500
74022  00 00000 00300
74023  00 00000 00274
74024  00 00000 00001
74025  11 00133 02000
74026  75 30000 00077
74027  00 00000 02000
74030  00 00000 00000
74031  00 00000 00000
```

FLEXOWRITER DUMP

PURPOSE: Print small blocks of instructions from the addressable memory in octal on the flexowriter. Since this is a relatively slow process, dumping with this program should be restricted to few instructions. In general, blocks of 8 instructions or more should be dumped with the octal dump (see next routine).

ENTRY:

1. Set Drum to ABNORMAL.
2. Set PAK = 40003.
3. Press START key. Computer will stop with PAK = 00020.
4. Set Drum to NORMAL.
5. Set location of first instruction to print in u-address of Q and number of instructions to print in v-address of Q.
6. Turn flexowriter on.
7. Press START key.

STORAGE ASSIGNMENT:

1. This routine occupies "dead space" locations 40042, 40043 and 70000 thru 70047.
2. This routine works from core locations 00000 thru 00047.

LIMITATIONS:

This routine will not print more than $_8(1000) = {}_{10}(512)$ words.

```
40003  45 00000 40042


40042  75 30050 00047
40043  11 70000 00000


70000  45 00000 00047
70001  00 00000 00004
70002  00 00001 00001
70003  00 00000 00007
70004  00 00000 00045
70005  00 00000 00057
70006  00 00000 00037
70007  00 00000 00052
70010  00 00000 00074
70011  00 00000 00070
70012  00 00000 00064
70013  00 00000 00062
70014  00 00000 00066
70015  00 00000 00072
70016  61 00000 00006
70017  00 00000 00000
70020  15 31000 00027
70021  16 31000 00017
70022  61 00000 00005
70023  45 00000 00044
```

70024	55	00027	31006
70025	37	00041	00032
70026	61	00000	00001
70027	11	00000	31000
70030	16	00002	00050
70031	37	00041	00033
70032	11	00001	00050
70033	61	00000	00001
70034	55	31000	00003
70035	51	00003	32000
70036	35	00016	00037
70037	61	00000	00006
70040	41	00050	00034
70041	37	00041	00042
70042	37	00041	00032
70043	21	00027	00002
70044	61	00000	00004
70045	61	00000	00004
70046	41	00017	00024
70047	56	00000	00020

MAGNETIC TAPE DUMP

PURPOSE: Write instructions on magnetic tape in a form suitable for printing in octal on the High Speed Printer. Each page of instructions to be printed is preceded by a heading containing a fast feed symbol and a blank line. The instructions are printed 6 per line and 58 lines per page. A blank line containing a printer stop code follows the last instruction to print.

ENTRY:

1. Set Drum to ABNORMAL.
2. Set PAK = 40004.
3. Press START key. Computer will stop with PAK = 00051.
4. Set Drum to NORMAL.
5. Set location of first instruction to print in u-address of Q and number of instructions to print in v-address of Q.
6. Turn Uniservo #2 on.
7. Press START key.

STORAGE ASSIGNMENT:

1. This routine occupies "dead space" locations 56000 thru 56045, 54000 thru 54037 and 66000 thru 66017.
2. This routine works from MC locations 00050 thru 00210.

LIMITATIONS:

The magnetic tape is written in variable block form one blockette of 20 words per record.

40004	45	00000	56000
56000	75	30040	56002
56001	11	56006	00050
56002	75	30040	56004
56003	11	54000	00110
56004	75	30020	00140
56005	11	66000	00150
56006	00	00000	00000
56007	11	31000	00050
56010	15	00050	00102
56011	11	00141	00157
56012	16	00050	00157
56013	41	00157	00057
56014	45	00000	00133
56015	17	00000	00155
56016	11	00145	00160
56017	77	10000	00150
56020	75	10003	00064
56021	77	10000	00151
56022	41	00160	00062
56023	45	00000	00164
56024	17	00000	00155
56025	75	00024	00071
56026	77	10000	00141
56027	11	00147	00161

56030	17	00000	00156
56031	16	00075	00114
56032	75	10024	00076
56033	11	00141	00172
56034	55	00102	31006
56035	11	31000	00170
56036	37	00116	00105
56037	11	00145	00160
56040	11	30000	00170
56041	11	00143	00162
56042	37	00116	00106
56043	11	00163	00162
56044	11	00141	00171
56045	55	00170	00003

54000	31	00171	00006
54001	52	00146	32000
54002	35	00144	00171
54003	41	00162	00107
54004	11	00171	30000
54005	21	00114	00143
54006	37	00116	00117
54007	37	00116	00105
54010	21	00102	00142
54011	41	00160	00123

54012	45	00000	00124
54013	41	00157	00102
54014	17	00000	00155
54015	75	10024	00127
54016	77	10000	00172
54017	17	00000	00156
54020	41	00161	00132
54021	45	00000	00055
54022	41	00157	00073
54023	17	00000	00155
54024	75	00024	00136
54025	77	10000	00154
54026	17	00000	00156
54027	37	00137	00140
54030	56	00000	00051
54031	00	00000	00000
54032	00	00001	00000
54033	00	00000	00001
54034	00	00000	00003
54035	00	00000	00005
54036	00	00000	00007
54037	00	00000	00071

66000	37	00465	12600
66001	00	00000	05152

66002	00	00006	70000
66003	00	00007	00000
66004	60	00000	00000
66005	02	00066	20000
66006	02	00600	20000
66007	00	00000	00000
66010	00	00000	00000
66011	00	00000	00000
66012	00	00000	00000
66013	00	00000	00004
66014	77	10000	00141
66015	17	00000	00156
66016	45	00000	00066
66017	00	00000	00000

Q REGISTER TO MEMORY

PURPOSE: Transfer a word from the Q register to the addressable memory. Following the successful transfer, the word is displayed in the accumulator for checking.

ENTRY:

1. Set Drum to ABNORMAL.
2. Set PAK = 40005.
3. Press START key. Computer will stop with PAK = 00001.
4. Set Drum to NORMAL.
5. Set location to store word in u-address <u>and</u> v-address of A right.
6. Enter word to store in Q register.
7. Press START key.
8. Word transferred will appear in A for checking.

STORAGE ASSIGNMENT:

1. This routine occupies "dead space" locations 40030 thru 40037.
2. This routine works from MC locations 00000 thru 00005.

LIMITATIONS: None

```
40005  45 00000 40030


40030  75 30006 00005
40031  11 40032 00000
40032  45 00000 00005
40033  16 32000 00003
40034  15 32000 00004
40035  11 31000 00000
40036  11 00000 32000
40037  56 00000 00001
```

MEMORY TO Q REGISTER

PURPOSE: Display a word in the addressable memory for visual inspection. The word to be displayed will appear in the Q register.

ENTRY:

1. Set Drum to ABNORMAL.
2. Set PAK = 40006.
3. Press START key. Computer will stop with PAK = 00001.
4. Set Drum to NORMAL.
5. Set location of word to be displayed in the u-address of A right.
6. Press START key.
7. Word to be displayed will appear in the Q register.

STORAGE ASSIGNMENT:

1. This routine occupies "dead space" locations 40040, 40041 and 72044 thru 72047.
2. This routine works from MC locations 00000 thru 00003.

LIMITATIONS: None

```
40006  45 00000 40040

40040  75 30004 00003
40041  11 72044 00000

72044  45 00000 00003
72045  15 32000 00002
72046  11 00000 31000
72047  56 00000 00001
```

SET MEMORY TO ZERO

PURPOSE: Set all of magnetic core memory to zero. This routine works from the drum dead space.

ENTRY:

1. Set Drum to ABNORMAL.
2. Set PAK = 40007.
3. Press START key. After clearing all of core memory, computer will stop with PAK = 40000.

STORAGE ASSIGNMENT:

This routine occupies "dead space" locations 60044 thru 60047 and works from these locations.

LIMITATIONS:

1. This routine assumes F1 (location 00000) contains a Manual Jump command prior to entering.
2. F1 is set to zero by this routine and must be restored before proceeding.

```
40007  45 00000 60044


60044  75 37777 60046
60045  23 00001 00001
60046  23 00000 00000
60047  56 00000 40000
```

REWIND MAGNETIC TAPE.

PURPOSE: Rewind a magnetic tape from the console. This routine works from the drum dead space.

ENTRY:

1. Set Drum to ABNORMAL.
2. Set PAK = 40010.
3. Press START key. Computer will stop with PAK = 40025 and (C) = 02 00200 00000.
4. Enter number of servo to be rewound in C_{14} thru C_{12}.
5. Press START key to rewind tape.

STORAGE ASSIGNMENT:

This routine occupies "dead space" locations 40023 thru 40027 and works from these locations.

LIMITATIONS:

The servo number entered in C must be one of the logically assigned servos and must be ready.

COMPUTER STOPS: None.

```
40010  45 00000 40026


40024  02 00200 00000

40025  17 00000 31000

40026  11 40024 31000

40027  56 00000 40025
```

RESTORE F1

PURPOSE: Restore the contents of F1 (location 00000) to a Manual Jump command. The v-address portion of F1 will not be altered.

ENTRY:

1. Set Drum to ABNORMAL.
2. Set PAK = 40011.
3. Press START key. Computer will stop with PAK = 00000. The original contents of F1 will be displayed in [illegible].
4. Set Drum to NORMAL.
5. Press START key to transfer to F1 and continue.

STORAGE ASSIGNMENT:

This routine occupies "dead space" locations 40044 thru 40047 and works from these locations.

LIMITATIONS: None.

COMPUTER STOPS: None.

40011 45 00000 40044

40044 11 00000 31000

40045 11 40000 00000

40046 16 31000 00000

40047 56 00000 00000

BLOCK TRANSFER

PURPOSE: Transfer a block of consecutive words from one location to another within the addressable memory of the computer.

ENTRY:

1. Set Drum to ABNORMAL.
2. Set PAK = 40012.
3. Press START key. Computer will stop with PAK = 00002.
4. Set Drum to NORMAL.
5. Set present starting address of block to be transferred in u-address of Q.
6. Set desired starting location of block in v-address of Q.
7. Set number of words to be transferred in u-address of A.
8. Press START key.

STORAGE ASSIGNMENT:

1. This routine occupies "dead space" locations 74040 thru 74047.
2. This routine works from core locations 00000 thru 00005.

LIMITATIONS:

All words to be transferred must be contained within the addressable memory both before and after transferring.

COMPUTER STOPS: None.

```
40012  75 30010 00006
40013  11 74040 00000

74040  45 00000 00007
74041  75 30000 00007
74042  35 00001 00005
74043  15 31000 00006
74044  16 31000 00006
74045  75 30000 00007
74046  11 00000 00000
74047  56 00000 00002
```

PAPER TAPE PUNCH (BI-OCTAL)

PURPOSE: Punch a block of addressable memory into paper tape in bi-octal form. Insert and check addresses and a check sum are provided automatically but may be suppressed at the operator's option.

ENTRY:

1. Set Drum to ABNORMAL.
2. Set PAK = 40014.
3. Press START key. Computer will stop with PAK = 00006.
4. Set Drum to NORMAL.
5. Enter location of first word to punch in the u-address of Q and the number of words to be punched in the v-address of Q.
6. Set MJ1 to suppress punching of insert and check addresses and check sum.
7. Set MJ2 to suppress punching check sum only.
8. Turn high speed punch on.
9. Press START key.

STORAGE ASSIGNMENT

1. This routine occupies "dead space" locations 60000 thru 60043 and 66020 thru 66046.
2. This routine works from core locations 00000 thru 00065 and uses locations 00272 and 00273 as erasable storage for the check sum.

LIMITATIONS: None

COMPUTER STOPS: None

```
40014  45 00000 60000


60000  75 30040 60002
60001  11 60004 00000
60002  75 30026 00065
60003  11 66020 00040
60004  45 00000 60002
60005  00 00000 77777
60006  00 00000 00000
60007  00 00000 00000
60010  00 00272 00002
60011  75 20000 00060
60012  75 20100 00010
60013  63 47701 00003
60014  45 10000 00021
60015  75 20003 00013
60016  63 47776 00003
60017  31 31000 00055
60020  63 00000 32000
60021  54 32000 00006
60022  63 00000 32000
60023  54 32000 00006
60024  63 10000 32000
60025  16 31000 00002
60026  31 00002 00017
60027  35 00005 00056
```

60030	15	31000	00057
60031	45	00000	00041
60032	15	31000	00027
60033	31	00000	00052
60034	63	00000	32000
60035	54	32000	00006
60036	37	00032	00033
60037	37	00032	00030
60040	37	00032	00030
60041	37	00032	00030
60042	37	00032	00030
60043	63	10000	32000

66020	21	31000	00001
66021	41	00002	00026
66022	45	10000	00064
66023	75	00003	00045
66024	63	00000	00003
66025	31	31000	00055
66026	63	10000	32000
66027	54	32000	00006
66030	63	00000	32000
66031	54	32000	00006
66032	63	10000	32000
66033	37	00053	00054

66034	45	20000	00064
66035	31	00003	00000
66036	75	20000	00060
66037	32	00000	00000
66040	22	00000	00272
66041	22	10000	00273
66042	11	00004	31000
66043	37	00053	00011
66044	37	00064	00065
66045	56	00000	00006
66046	00	00400	00000

MAGNETIC TAPE BOOTSTRAP (FIXED BLOCK)

PURPOSE: Read the first block of 120 words from tape on servo #6 into MC locations 00001 thru $_8$(00170). This routine works from the drum dead space and was designed to bootstrap the customer engineers diagnostic routines.

ENTRY:

1. Set Drum to ABNORMAL.
2. Rewind tape on servo number 6.
3. Set PAK = 40016.
4. Press START key. After reading first block from tape 6 into core, computer will stop with PAK = 00116.
5. Set Drum to NORMAL.
6. Press START key to proceed.

STORAGE ASSIGNMENT:

This routine occupies "dead space" locations 74032 thru 74037 and works from these locations.

LIMITATIONS:

1. This routine assumes that the first block on tape 6 was written in fixed block form without blockette spaces.
2. This routine assumes F1 (location 00000) contains a Manual Jump (45) command. If F1 has been destroyed, restore F1 and start over.

```
40016  45 00000 74032


74032  17 00000 74037
74033  75 10170 74035
74034  76 10000 00001
74035  76 00000 32000
74036  56 00000 00116
74037  02 00602 60000
```

TAPE TO TAPE CONVERSION

PURPOSE: Convert a magnetic tape containing binary information to a magnetic tape of a form suitable for printing the information on the High Speed Printer as octal instructions. The information is read from the tape to be converted into magnetic core beginning with location $_8(01002)$. The record number is stored in location $_8(01000)$ and the number of parity errors detected in reading that record is stored in location $_8(01001)$. This information is written on the tape on servo #2 in XS3 code with 6 words per line and one blank line between each record. In addition, a trail is printed on the flexowriter to indicate the condition of each record as it is read from the tape.

ENTRY:

1. Set Drum to ABNORMAL.
2. Set PAK = 40020.
3. Press START key. Computer will stop with PAK = 00100.
4. Set Drum to NORMAL.
5. Set the servo number (n) containing the tape to be converted in Q_{14} thru Q_{12}.
6. Turn on servos number n and number 2.
7. Turn on flexowriter.
8. Set Switch MJ1 to suppress rewinding tape n before conversion.
9. Press START key.

TAPE TO TAPE CONVERSION

STORAGE ASSIGNMENT:

1. This routine occupies "dead space" locations 44000 thru 44047, 50000 thru 50047, 52000 thru 52047, 62000 thru 62047, and 64000 thru 64047.
2. This routine works from core locations 00000 thru 00423.

LIMITATIONS:

1. This program will not read records from tape containing more than ${}_{10}(3582) = {}_{8}(6776)$ words.
2. Since servo #2 is used for output this program will not convert a tape on servo #2.

COMPUTER STOPS: None.

```
40020  45 00000 64000


64000  75 30050 64002
64001  11 44000 00000
64002  75 30050 64004
64003  11 50000 00050
64004  75 30050 64006
64005  11 52000 00120
64006  75 30050 64010
64007  11 62000 00170
64010  75 30036 00140
64011  11 64012 00240


44000  45 00000 64002
44001  00 00000 00000
44002  00 00000 00001
44003  00 00000 00002
44004  00 00000 00003
44005  00 00000 00004
44006  00 00000 00005
44007  00 00000 00006
44010  00 00000 00007
44011  02 00200 00000
44012  02 00062 00000
44013  02 00600 00000
```

44014	00	00001	00000
44015	00	00000	00071
44016	00	00000	00077
44017	00	00000	00013
44020	00	00000	07777
44021	00	00000	00045
44022	00	00000	00037
44023	00	00000	00052
44024	00	00000	00074
44025	00	00000	00070
44026	00	00000	00064
44027	00	00000	00062
44030	00	00000	00066
44031	00	00000	00072
44032	00	00000	00001
44033	47	12571	62204
44034	00	00000	00002
44035	04	15301	21401
44036	25	45000	00000
44037	00	00000	00001
44040	04	03364	50000
44041	00	00000	00002
44042	04	07032	20466
44043	45	00000	00000

```
44044  00 00000 00003
44045  04 20062 20403
44046  26 04013 01520
44047  45 00000 00000

50000  60 00000 00000
50001  02 00066 20000
50002  02 00600 20000
50003  00 00032 00032
50004  00 00034 00034
50005  00 00037 00037
50006  00 00041 00041
50007  00 00044 00044
50010  61 00000 00022
50011  76 10000 01002
50012  00 00000 00000
50013  00 00000 00000
50014  00 00000 00000
50015  00 00000 00000
50016  37 00465 12600
50017  00 00000 05152
50020  00 00006 70000
50021  00 00007 00000
50022  16 00231 00250
50023  45 00000 00232
```

50024	00	00000	00000
50025	00	00000	00000
50026	00	00000	00000
50027	00	00000	00000
50030	75	10003	00274
50031	16	31000	00011
50032	17	00000	00011
50033	61	00000	00021
50034	11	00001	01000
50035	15	00053	00155
50036	15	00053	00166
50037	37	00170	00155
50040	21	01000	00002
50041	11	00017	00062
50042	11	01000	31000
50043	55	31000	31003
50044	51	00010	32000
50045	47	00121	00116
50046	41	00062	00113
50047	55	31000	31003
52000	51	00010	32000
52001	35	00060	00122
52002	61	00000	00022
52003	41	00062	00117

```
52004  17 00000 00012
52005  16 00061 00141
52006  11 00001 01001
52007  76 00000 32000
52010  43 00001 00141
52011  43 00002 00144
52012  43 00003 00146
52013  43 00004 00172
52016  43 00005 00174
52015  43 00006 00140
52016  43 00007 00140
52017  43 00010 00140
52020  56 00000 00100
52021  76 10000 01002
52022  21 00141 00002
52023  45 00000 00127
52024  21 01001 00002
52025  45 00000 00141
52026  11 01001 32000
52027  47 00150 00152
52030  15 00054 00155
52031  45 00000 00153
52032  15 00055 00155
52033  17 00000 00013
```

52034	15	00155	00166
52035	11	00037	00062
52036	45	00000	00165
52037	11	00006	00063
52040	55	31000	31006
52041	51	00016	32000
52042	47	00163	00164
52043	61	00000	31000
52044	41	00063	00160
52045	21	00166	00014
52046	11	00040	31000
52047	41	00062	00157

62000	37	00170	00171
62001	45	00000	00204
62002	15	00056	00155
62003	45	00000	00153
62004	15	00057	00155
62005	37	00170	00154
62006	17	00000	00011
62007	17	00000	00051
62010	75	00024	00202
62011	77	10000	00050
62012	17	00000	00052
62013	56	00000	00100

```
62014  31 00141 00000
62015  34 00061 00000
62016  35 00003 00062
62017  15 00110 00236
62020  45 00000 00264
62021  41 00062 00213
62022  45 00000 00267
62023  17 00000 00051
62024  11 00006 00063
62025  77 10000 00066
62026  75 10003 00220
62027  77 10000 00067
62030  41 00063 00216
62031  77 10000 00001
62032  17 00000 00052
62033  17 00000 00051
62034  75 00024 00226
62035  77 10000 00001
62036  11 00015 00064
62037  17 00000 00052
62040  75 10024 00072
62041  11 00001 00400
62042  55 00236 31006
62043  11 31000 00074
```

62044	37	00252	00241
62045	11	00006	00063
62046	11	01000	00076
62047	11	00002	00065
64012	37	00252	00242
64013	11	00005	00065
64014	11	00001	00075
64015	55	00074	00003
64016	31	00075	00006
64017	52	00010	32000
64020	35	00006	00075
64021	41	00065	00243
64022	11	00075	00400
64023	21	00250	00002
64024	37	00252	00253
64025	37	00252	00241
64026	21	00236	00014
64027	41	00063	00257
64030	45	00000	00260
64031	41	00062	00236
64032	17	00000	00051
64033	75	10024	00263
64034	77	10000	00400
64035	17	00000	00052

64036	41	00064	00266
64037	45	00000	00211
64040	41	00062	00230
64041	17	00000	00051
64042	75	00024	00272
64043	77	10000	00001
64044	17	00000	00052
64045	45	00000	00105
64046	45	10000	00103
64047	45	00000	00102

Talmadge

USEful Note # 16

2 October 1957

SUBJECT: Multiple Precision Floating Point Routine

CONTRIBUTOR: Remington Rand

A MULTIPLE PRECISION FLOATING POINT ROUTINE

FOR THE

UNIVAC SCIENTIFIC COMPUTER

by
T. C. Tollefson

General Description

The multiple precision floating point routine, described on page 1 through 4, was coded for use with the Univac Scientific 1103A Computer. The coding is on pages 5 through 9 in both relative (RECO) and absolute addresses.

The routine performs arithmetic operations on floating point numbers of the order of 62 bits by use of stated point operations. Four arithmetic operations are provided by the routine; they are addition, subtraction, multiplication and division. Any one of the four operations may be performed by entering into the proper subroutine by means of the Interpret instruction. The Interpret instruction indicates the operation to be performed as well as the storage locations of the two operands involved.

Each of the two operands occupies two storage cells. The capacity of each cell or storage location is 36 bits and therefore each operand is 72 bits in length. For each operand, 62 bits are allocated for the mantissa M, 8 bits for the (biased) characteristic C, and 2 bits, S_1 and S_2, for the sign. The 72 bits are arranged as shown below. The arrangement is true of both the operand and the result.

Number of Bits

	1	8	27
Cell 1	S_1	C	M_1

	1	35
Cell 2	S_2	M_2

The higher order 27 bits (M_1) of the total 62 bits which comprise the mantissa are always in cell 1 and the lower order 35 bits (M_2) are always in cell 2. The mantissa contained in the two cells is always normalized and therefore the first significant bit occurs as the left-most bit of the higher order 27 bits, M_1. The binary point is assumed to be immediately to the left of this bit (between C and M_1).

Any number N used by the routine must be in the form $M \cdot 2^C$ and must satisfy one of the following conditions

(1) $N = 0$ or

(2) $2^{-129} < |N| \leq 2^{127}$

The characteristic C is biased which allows the number N to vary over the values stated in (2) above. If K represents the normal exponent of the number, then C, the biased characteristic, is given by $C = K + 128$. The value of the mantissa M, located in M_1 and M_2, may range in value as defined by the expression $\frac{1}{2} \leq M < 1$.

The sign bits S_1 and S_2 must be in agreement, that is, they must both be either zero or one. If they are "1's", M is a negative number and C and M are in one's complement form.

S_1, S_2, C, M_1 and M_2 are necessarily equal to zero whenever N equals zero.

The order of precision of this routine is 62 bits. The results are similarily 62 bits with an occasional round-off error at the 62nd bit.

The Interpret instruction 14 OP U'V' is utilized to initiate any one of the arithmetic operations. The operation code, OP, used for any specific operation is analogous to the regular machine codes for standard stated point operations. The code for these four arithmetic operations is listed below.

Add:	AT (35)
Subtract:	ST (36)
Multiply	MP (74)
Divide:	DV (73)

The U' and V' addresses refer to the first of two consecutive storage locations at which each operand is stored. Thus, if the first operand is stored at u_1 and u_2, and the second operand at v_1 and v_2; $U' = u_1$ and $V' = v_1$.

The result of the operation performed is properly packed and normalized and is found in the A and Q registers. The result containing the higher order 27 bits and the characteristic, is in double extension form in A. At the conclusion of the operation (i.e., control transferred to F_1), the initial operands are found undisturbed in their original locations U', and U' + 1, and V', V' + 1.

The coding of the routine, pages 5 through 9 , is in two forms; the address in the left-hand column is relative to 1000, and the right-hand code is in RECO form.

The routine is divided into six subroutines, ready for RECO assembly. The regional assignments are given below.

SA: addition and subtraction
MP: multiplication
DV: division
DC: decoder and unpacking
RP: rounding and packing
CS: constants and temporary storage

A jump instruction at F_2 transfers control to 1000 (DCO), which initiates decoding, i.e., determines which arithmetic operation is to be performed, and unpacks the normalized, packed operands. DCO is the absolute location of the first instruction of the decoding section. A jump is then made to the desired subroutine (SA, MP, DV) to perform the arithmetic operation. The rounding and packing subroutine (RP) performs a rounding operation and truncation at the 63rd bit of the mantissa and normalizes and packs the results which are then stored in A and Q. Control is now transferred to F_1, which contains a jump instruction to the address following the initiating Interpret instruction.

Use of the Routine

If the six subroutines discussed above are to be RECO assembled, the individual subroutine regions must be labeled SA, MP, DV, DC, RP and CS. These may be placed, individually, anywhere in the memory that is desired; or, the entire routine, coded relative to 1000, may be address-modified to be placed as one package anywhere in the memory.

The routine does not set up F_1 or F_2; these must be preset by the user and should be set up as follows:

F_1: MJ 0 [0]

F_2: MJ 0 1000 (or DCO)

The Interpret instruction modifies the v address of F_1; F_1 and F_2 are not modified in any way by the routine.

If we let U' and V' represent the storage of the two operands, these each being separated into two locations, u_1, u_2 and v_1, v_2 it must be remembered that u_1 and u_2 must be consecutive as are v_1 and v_2. U' and V' need not be consecutive.

Example:

Compute (a - b): a is stored at 00500 and 00501, and b at 02100 and 02101. The proper Interpret instruction would be 1436 0500 2100. Note: Drum addresses cannot be used.

Alarm and Excess Considerations

The characteristic C, equal to K + 128 as stated above, cannot exceed certain limits. Results which yield biased characteristics greater than 255 or less than zero cannot be expressed; consequently, alarms and/or procedures have been incorporated in the routine to indicate when these limits have been exceeded.

If the characteristic of the result is $C \leq 0$ or $C > 255$, zeros are entered in the result locations and control transferred to F_1. If instead of having zeros entered into the result and control transferred to F_1, it is desired to have an alarm or error exit under these extremes, two instructions, 1203 (RP52) and 1212 (RP61), must be changed. These two instructions normally read

RP52 : MJ 0 RP55

RP61 : SJ RP55 RP53

These must be altered to read

RP52 : MJ 0 C

RP61 : SJ C RP53,

where C is any specified (by the user) cell to jump to in case of the above extreme condition. <u>Note</u>: If it is convenient to have results equal to zero in case the characteristic C becomes less than zero and have an alarm only for $C > 255$, RP52 is the only cell that need be altered.

If an attempt is made to divide by zero the operation is by-passed, control is transferred back to F_1, but the computer is halted by the instruction DV51, which normally reads MS 0 0. This instruction may be altered to provide any needed divide-error indication or exit.

Since no check is made on the validity (correctness of format) of the input operands, use of incorrect operands will result in either an error indication or nonsensical answers.

		Absolute Address			Relative Address			Remarks
1000	11	01117	31000	TP	CS 36	Q	DCO	Decode S. R. Enter
1001	53	00000	32000	QS	0	A		
1002	34	01102	00017	SS	CS 21	17		
1003	15	32000	01007	TU	A	DC 7		
1004	15	32000	01015	TU	A	DC 15		
1005	15	32000	01033	TU	A	DC 33		
1006	11	01115	31000	TP	CS 34	Q		
1007	53	00000	32000	QS	0	A		
1010	54	32000	00003	LA	A	3		Determine Location
1011	15	32000	01022	TU	A	DC 22		of Operands
1012	32	01104	00000	SA	CS 23	0		
1013	15	32000	01023	TU	A	DC 23		
1014	11	01116	31000	TP	CS 35	Q		
1015	53	00000	32000	QS	0	A		
1016	54	32000	00017	LA	A	17		
1017	15	32000	01024	TU	A	DC 24		
1020	32	01104	00000	SA	CS 23	0		
1021	15	32000	01025	TU	A	DC 25		
1022	11	00000	01061	TP	0	CS 0		
1023	11	00000	01062	TP	0	CS 1		
1024	11	00000	01063	TP	0	CS 2		Move Operands to
1025	11	00000	01064	TP	0	CS 3		Temporary Storage
1026	11	01061	01065	TP	CS 0	CS 4		
1027	11	01062	01066	TP	CS 1	CS 5		
1030	45	00000	01031	MJ	0	DC 31		
1031	45	00000	01040	MJ	0	DC 40		
1032	11	01120	31000	TP	CS 37	Q		
1033	53	00000	32000	QS	0	A		
1034	43	01122	01220	EJ	CS 41	SA 2		Code Test
1035	43	01123	01216	EJ	CS 42	SA 0		
1036	43	01124	01313	EJ	CS 43	MP 0		
1037	45	00000	01340	MJ	0	DV 0		
1040	11	01061	32000	TP	CS 0	A		
1041	22	00011	01075	LT	00011	CS 14		
1042	46	01043	01044	SJ	DC 43	DC 44		
1043	13	01075	01075	TN	CS 14	CS 14		
1044	22	10033	01067	LT	10033	CS 6		
1045	11	01060	31000	TP	DC 60	Q		
1046	53	01061	01067	QS	CS 0	CS 6		
1047	22	31000	01061	LT	31000	CS 0		
1050	11	01063	32000	TP	CS 2	A		Mask Out Mantissas
1051	22	00011	01076	LT	00011	CS 15		and Characteristics
1052	46	01053	01054	SJ	DC 53	DC 54		
1053	13	01076	01076	TN	CS 15	CS 15		
1054	22	10033	01067	LT	10033	CS 6		
1055	53	01063	01067	QS	CS 2	CS 6		
1056	22	31000	01063	LT	31000	CS 2		
1057	45	00000	01032	MJ	0	DC 32		
1060	00	77777	77777	00	77777	77777	DC 60	
1061	00	00000	00000	0	0	0	CS 0	
1062	00	00000	00000	0	0	0		
1063	00	00000	00000	0	0	0		
1064	00	00000	00000	0	0	0		
1065	00	00000	00000	0	0	0		

1066	00	00000	00000	0	0	0	
1067	00	00000	00000	0	0	0	
1070	00	00000	00000	0	0	0	Temporary Storage
1071	00	00000	00000	0	0	0	
1072	00	00000	00000	0	0	0	
1073	00	00000	00000	0	0	0	
1074	00	00000	00000	0	0	0	
1075	00	00000	00000	0	0	0	
1076	00	00000	00000	0	0	0	
1077	00	00000	00000	0	0	0	
1100	00	00000	00000	0	0	0	
1101	00	00000	00000	0	0	0	
1102	00	00000	00001	00	00000	00001	
1103	00	00000	00400	00	00000	00400	
1104	00	00001	00000	00	00001	00000	
1105	00	00000	00034	00	00000	00034	
1106	00	00000	00044	00	00000	00044	
1107	00	00000	00076	00	00000	00076	
1110	00	00000	00100	00	00000	00100	
1111	00	00000	00144	00	00000	00144	
1112	00	00000	00200	00	00000	00200	
1113	00	00000	00276	00	00000	00276	Constants
1114	00	00000	00400	00	00000	00400	
1115	00	00777	70000	00	00777	70000	
1116	00	00000	07777	00	00000	07777	
1117	00	00000	77777	00	00000	77777	
1120	00	77000	00000	00	77000	00000	
1121	37	70000	00000	37	70000	00000	
1122	00	35000	00000	00	35000	00000	
1123	00	36000	00000	00	36000	00000	
1124	00	71000	00000	00	71000	00000	
1125	00	00000	01157	0	0	RP 26	
1126	00	00000	01176	0	0	RP 45	Set-Up Constants
1127	00	00000	01133	0	0	RP 2	
1130	00	00000	00043	00	00000	00043	CS 47
1131	16	01125	01156	TV	CS 44	RP 25	RP 0. Add Round and Pack Enter
1132	45	00000	01134	MJ	0	RP 3	
1133	16	01126	01156	TV	CS 45	RP 25	Multiply Round and Pack Enter
1134	47	01135	01206	ZJ	RP 4	RP 55	Zero Test
1135	74	32000	01073	SF	A	CS 12	
1136	54	32000	00044	LA	A	44	
1137	46	01142	01140	SJ	RP 11	RP 7	
1140	32	01103	00000	SA	CS 22	0	
1141	46	01147	01144	SJ	RP 16	RP 13	Rounding
1142	36	01103	32000	ST	CS 22	A	
1143	46	01144	01147	SJ	RP 13	RP 16	
1144	11	01101	01074	TP	CS 20	CS 13	
1145	54	32000	00001	LA	A	1	
1146	45	00000	01150	MJ	0	RP 17	
1147	13	01102	01074	TN	CS 21	CS 13	
1150	22	00033	01070	LT	00033	CS 7	
1151	22	10001	32000	LT	10001	A	
1152	22	00043	01067	LT	00043	CS 6	
1153	22	00000	32000	LT	00000	A	Truncation
1154	27	01070	01101	CC	CS 7	CS 20	
1155	22	00043	01070	LT	00043	CS 7	
1156	45	00000	00000	MJ	0	0	

1157	31	01073	00000	SP	CS 12	0	
1160	35	01074	32000	AT	CS 13	A	
1161	42	01106	01165	TJ	CS 25	RP 34	
1162	32	01075	00000	SA	CS 14	0	
1163	34	01111	00000	SS	CS 30	0	Add Characteristic
1164	45	00000	01211	MJ	0	RP 51	
1165	35	01075	32000	AT	CS 14	A	
1166	36	01105	32000	ST	CS 24	A	
1167	45	00000	01202	MJ	0	RP 51	
1170	11	01067	32000	TP	CS 6	A	
1171	46	01172	01173	SJ	RP 41	RP 42	
1172	13	01077	01077	TN	CS 16	CS 16	Packing
1173	11	01121	31000	TP	CS 40	Q	
1174	53	01077	01067	QS	CS 16	CS 6	
1175	45	00000	01213	MJ	0	RP 62	
1176	31	01075	00000	SP	CS 14	0	
1177	32	01076	00000	SA	CS 15	0	Multiply Characteristic
1200	34	01112	00000	SS	CS 31	0	
1201	32	01074	00000	SA	CS 13	0	
1202	42	01114	01211	TJ	CS 33	RP 60	Char. Overflow Test
1203	45	00000	01206	MJ	0	RP 55	
1204	22	10033	01077	LT	10033	CS 16	
1205	45	00000	01170	MJ	0	RP 37	
1206	11	01101	01067	TP	CS 20	CS 6	Put Zeros in Answer
1207	11	01101	01070	TP	CS 20	CS 7	
1210	45	00000	01213	MJ	0	RP 62	
1211	11	32000	32000	TP	A	A	Char. Overflow Test
1212	46	01206	01204	SJ	RP 55	RP 53	
1213	11	01067	32000	TP	CS 6	A	Move Answer to A
1214	11	01070	31000	TP	CS 7	Q	Move Answer to Q
1215	45	00000	00000	MJ	0	0	RP 64
1216	13	01063	01063	TN	CS 2	CS 2	SA 0. Subtract S.R. Enter
1217	13	01064	01064	TN	CS 3	CS 3	
1220	11	01075	32000	TP	CS 14	A	Add S.R. Enter
1221	36	01076	32000	ST	CS 15	A	
1222	46	01223	01235	SJ	SA 5	SA 17	
1223	11	01061	31000	TP	CS 0	Q	
1224	11	01063	01061	TP	CS 2	CS 0	
1225	11	31000	01063	TP	Q	CS 2	
1226	11	01062	31000	TP	CS 1	Q	Reverse Operands So
1227	11	01064	01062	TP	CS 3	CS 1	One With Larger
1230	11	31000	01064	TP	Q	CS 3	Characteristic is First,
1231	11	01075	31000	TP	CS 14	Q	i.e.,
1232	11	01076	01075	TP	CS 15	CS 14	$\lvert Cx - Cy \rvert > 0$
1233	11	31000	01076	TP	Q	CS 15	
1234	13	32000	32000	TN	A	A	
1235	11	32000	01100	TP	A	CS 17	
1236	42	01107	01250	TJ	CS 26	SA 32	
1237	11	01062	01070	TP	CS 1	CS 7	
1240	11	01061	32000	TP	CS 0	A	
1241	46	01242	01243	SJ	SA 24	SA 25	If
1242	13	01075	01075	TN	CS 14	CS 14	$62 \leq \lvert Cx - Cy \rvert$
1243	11	01061	01067	TP	CS 0	CS 6	Compute Sum
1244	11	01121	31000	TP	CS 40	Q	
1245	54	01075	00033	LA	CS 14	33	
1246	53	01075	01067	QS	CS 14	CS 6	
1247	45	00000	01213	MJ	0	RP 62	

1250	11	01100	32000	TP	CS 17	A		
1251	43	01106	01264	EJ	CS 25	SA 46		
1252	42	01106	01264	TJ	CS 25	SA 46		
1253	11	01106	32000	TP	CS 25	A		
1254	35	01130	32000	AT	CS 47	A		If
1255	36	01100	32000	ST	CS 17	A		$37 \leq \|Cx - Cy\| \leq 61$
1256	16	32000	01260	TV	A	SA 42		Compute Sum
1257	11	01063	32000	TP	CS 2	A		
1260	54	32000	00000	LA	A	0		
1261	22	00000	01070	LT	00000	CS 7		
1262	23	01067	01067	RS	CS 6	CS 6		
1263	45	00000	01304	MJ	0	SA 66		
1264	11	01106	32000	TP	CS 25	A		
1265	36	01100	32000	ST	CS 17	A		
1266	16	32000	01272	TV	A	SA 54		
1267	35	01106	32000	AT	CS 25	A		
1270	16	32000	01277	TV	A	SA 61		
1271	11	01063	32000	TP	CS 2	A		
1272	54	32000	00000	LA	A	0		If
1273	22	00000	01067	LT	00000	CS 6		$0 \leq \|Cx - Cy\| \leq 36$
1274	54	01063	20043	LA	CS 2	A 43		Compute Sum
1275	35	01064	32000	AT	CS 3	A		
1276	54	32000	00001	LA	A	1		
1277	54	32000	00000	LA	A	0		
1300	22	31000	31000	LT	31000	Q		
1301	11	01063	32000	TP	CS 2	A		
1302	27	31000	01101	CC	Q	CS 20		
1303	22	00043	01070	LT	00043	CS 7		
1304	11	01061	32000	TP	CS 0	A		
1305	35	01067	32000	AT	CS 6	A		
1306	54	32000	00043	LA	A	43		
1307	35	01062	32000	AT	CS 1	A		Construct Sum in A
1310	35	01070	32000	AT	CS 7	A		
1311	54	32000	00001	LA	A	1		
1312	45	00000	01131	MJ	0	RP 0	SA 74	
1313	11	01061	32000	TP	CS 0	A	MP 0	
1314	47	01315	01206	ZJ	MP 2	RP 55		
1315	11	01063	32000	TP	CS 2	A		Zero Test
1316	47	01317	01206	ZJ	MP 4	RP 55		
1317	16	01127	01337	TV	CS 46	MP 24		
1320	11	01061	01065	TP	CS 0	CS 4		
1321	11	01062	01066	TP	CS 1	CS 5		
1322	71	01066	01064	MP	CS 5	CS 3		
1323	22	00000	32000	LT	00000	A		
1324	22	00013	01067	LT	00013	CS 6		
1325	71	01065	01064	MP	CS 4	CS 3		$x_1y_2 + x_2y_1$
1326	72	01063	01066	MA	CS 2	CS 5		
1327	22	00001	01070	LT	00001	CS 7		
1330	55	32000	00043	LQ	A	43		
1331	22	00012	01071	LT	00012	CS 10		
1332	71	01065	01063	MP	CS 4	CS 2	x_1y_1	
1333	35	01070	32000	AT	CS 7	A		
1334	54	32000	00011	LA	A	11		xy
1335	35	01071	32000	AT	CS 10	A		
1336	35	01067	32000	AT	CS 6	A		
1337	45	00000	00000	MJ	0	0	MP 24	

1340	11	01061	32000	TP	CS 0	A	DV 0
1341	47	01342	01206	ZJ	DV 2	RP 55	Zero Test
1342	11	01063	32000	TP	CS 2	A	
1343	47	01344	01411	ZJ	DV 4	DV 51	
1344	54	01103	20064	LA	CS 22	A 64	
1345	73	01063	01065	DV	CS 2	CS 4	$1/y_1$
1346	22	31000	01066	LT	31000	CS 5	
1347	47	01350	01356	ZJ	DV 10	DV 16	
1350	11	01065	32000	TP	CS 4	A	
1351	46	01352	01353	SJ	DV 12	DV 13	Remainder
1352	13	01066	32000	TN	CS 5	A	
1353	11	01066	32000	TP	CS 5	A	
1354	54	32000	00034	LA	A	34	
1355	73	01063	01066	DV	CS 2	CS 5	
1356	71	01065	01065	MP	CS 4	CS 4	
1357	22	00002	32000	LT	00002	A	$y_1^{-2}y_2$
1360	71	32000	01064	MP	A	CS 3	
1361	22	00002	01067	LT	00002	CS 6	
1362	71	01067	01065	MP	CS 6	CS 4	
1363	22	00000	32000	LT	00000	A	
1364	71	32000	01064	MP	A	CS 3	$y_1^{-3}y_2^2$
1365	22	00000	32000	LT	00000	A	
1366	22	00016	31000	LT	00016	Q	
1367	54	01065	20033	LA	CS 4	A 33	
1370	36	01067	32000	ST	CS 6	A	
1371	54	32000	00001	LA	A	1	
1372	35	31000	32000	AT	Q	A	
1373	35	01066	32000	AT	CS 5	A	
1374	74	32000	01070	SF	A	CS 7	Construct 1/y in A
1375	22	00034	01063	LT	00034	CS 2	
1376	22	10000	31000	LT	10000	Q	
1377	22	00000	32000	LT	00000	A	
1400	27	31000	01101	CC	Q	CS 20	
1401	22	00043	01064	LT	00043	CS 3	
1402	37	01337	01320	RJ	MP 24	MP 5	Compute xy
1403	37	01156	01134	RJ	RP 25	RP 3	Round and Truncate
1404	31	01075	00000	SP	CS 14	0	
1405	34	01076	00000	SS	CS 15	0	Divide Characteristic
1406	32	01112	00000	SA	CS 31	0	
1407	32	01074	00000	SA	CS 13	0	
1410	45	00000	01202	MJ	0	RP 51	Pack Answers
1411	56	00000	00000	MS	0	0	DV 51 Alarm Stop