intercom 1000

DOUBLE PRECISION

TECHNICAL MANUAL

THE BENDIX CORPORATION COMPUTER DIVISION
5630 Arbor Vitae Street, Los Angeles 45, California

APPLICATIONS SECTION

Project No. 83

TITLE: INTERCOM 1000

TYPE: Interpretive Routine

EQUIPMENT AFFECTED: G-15D

SUBROUTINES USED: Refer to contents of this project

PREPARED BY: Don E. Hassell

APPROVED BY: T. Yamashita

Date: 1 May 1959 **Revision:**

TABLE OF CONTENTS

PART I

TABLE OF CONTENTS (Cont'd)

TABLE OF CONTENTS (Cont'd)

INTERCOM 1000

DOUBLE PRECISION

CHECK SUM

.3121959	Loader I
-.7v7u575	Line I
-.4wx8v55	Line 2
.xx13506	Line 3
.9w7wv36	Line 6
.z5x1zz3	Line 7
-.0z8v9y3	Line 8
-.7y08v58	Line 0
-.1347192	Line 4
-.1000601	Loader II
.1000600	Selective Print
.1001601	Formater

1. STRUCTURE

1.1 BASIC PACKAGE

The Intercom 1000 double precision package is contained within G-15 memory lines 00 through 08. Line 00 contains the basic logic for the conversion and execution of commands. Line 01 is used by the index register operations and output commands. Line 02 and 03 are used by the floating point input-output conversion routine and line 07 contains the constants used by these routines. Line 04 contains the floating point arithmetic operation.

Words 78-u7 of line 05 are used for temporary storage and execution of the typewriter input sequence. There are no commands premanently stored in line 05. Words 00-74 of line 05 are used when the Selective Listing Subroutine is in operation, but if the Selective Listing Subroutine is not to be used, line 05 may be used for the execution of appendix routine or words 00-74 may be used for storing data or Intercom commands. This expands the interpretive memory by 75 words. The use of this line for the execution of appendix subroutine is explained in more detail on page 33.

Parts of the above-mentioned lines and line 06 and 08 are utilized by the other operations available in Intercom.

1.2 INTERPRETIVE MEMORY

Words 00 through 99 of lines 09 through 18 comprise the interpretive memory. Appendix routines share this memory. Generally, the addition of any appendix routine reduces the interpretive memory by 100 words. More details concerning appendix subroutines are given in Applications Section Project Number 84.

Words u0 and u1 of line 09 through 18 are not useable as command storage but may be used for storage of data providing storage positions for ten numbers.

The Intercom address is the same as the G-15 address. For example, the Intercom address 1063 corresponds to G-15 location 1063.

1.3 INDEX REGISTERS

Index registers 1 through 9 are located in lines 09 through 17, respectively. Index 10 is stored in line 18, however, since the command format provides for seven digits only, this index register must be referred to as index register u. A command naming this index register would be written u OPCHWD.

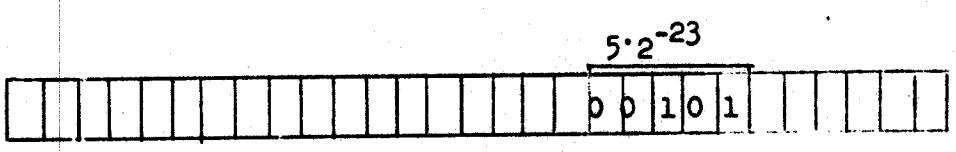
A set of index registers is composed of six components, three for control of the indexing of WD and three for control of the indexing of CH. Bwd is in word u4, Bch is in word u5, Dwd is in word u2, Dch is in word u6, Lwd is in word u3 and Lch is in word u7.

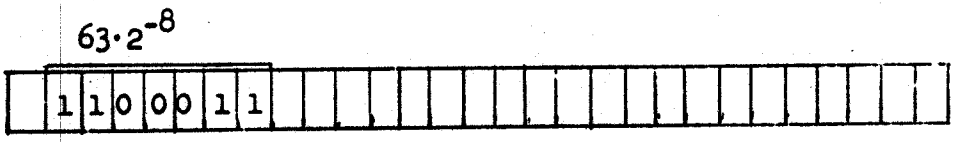
The use of an appendix routine that utilizes words u2 through u7 destroys the set of index registers contained within the line in which the appendix routine is stored.

Index registers are set to a binary number by the "Set Index Register" commands. The channel setting is stored in the index register as a two digit binary integer scaled by 2^{-23}. For example, the execution of the command 1740500 sets the sexadecimal number "00000u0" in the Dch register of set one.

The word setting is stored in the index register as a two digit binary integer scaled by 2^{-8}. For example, the execution of the command 2720099 sets the sexadecimal number "6300000" in the Lwd register of set two.

The following is the binary representation of the sexadecimal numbers given in the example of the two preceding paragraphs as they would appear in memory.





1.4 THE "A" REGISTER

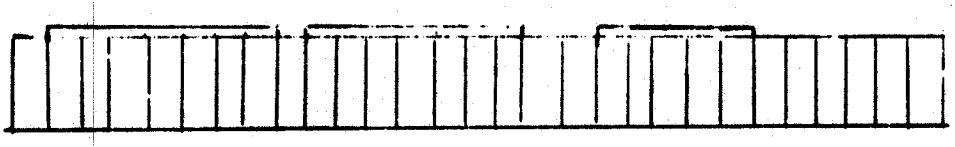
The "A" register, addressed 2100, occupies words 00 and 01 of short line 21.

1.5 COMMANDS

Intercom commands are stored in memory in decimal form. A simple conversion takes place between type-in and storage; namely, K OP CHWD is converted to KWDOPCH. All commands appear in memory in this converted form.

During the process of interpretation and execution, the command is converted to binary and placed in 21.03 in the form shown below.

The significance of each bit after conversion is as follows:



T_{29}	Zero
T_{28}-T_{22}	Word location in channel
T_{21}	Zero
T_{20}-T_{14}	Operation Code (OP). (This is the first command of any appendix subroutine and usually the first command of an Intercom operation sequence.)
T_{13}-T_{12}	Zero
T_{11}-T_{7}	Channel after conversion to sexadecimal.
T_{6}-T_{2}	Zero
T_{1}	Zero

Note the absence of K in the converted command. The contents of the index register K are added to the command during conversion; therefore, the value of K does not need to be retained.

Operations that require data pick-up have a dummy added to the converted command. The modified converted command is obeyed in AR, which has the effect of picking up data into ID and transferring to the first command of some subroutine.

1.6 NUMBERS

Numbers may be entered into or typed from the memory of the computer in fixed or floating point decimal notation; however, all numbers are carried internally as floating point binary numbers. They occupy an even word and the next consecutive higher odd-numbered location. The sign is carried in the least significant bit (T_1) of the even word.

The characteristic (b) or power of 2 by which the mantissa (B) is multiplied are the bits T_2-T_9 of the even word in excess -128 notation. The value of these numbers may be expressed as follows:

$$1/2 \leq |B| < 1 \quad \text{and} \quad 1 \leq b \leq 255$$

To elaborate B is carried in normalized form, that is, the most significant bit of all non-zero numbers is a one in the T_{29} bit of the odd word.

Example:

50.500000000000 = $1/2 \times 2^0$ which is 80000000000080
51.100000000000 = 1×2^1 which is 80000000000081
-52.100000000000 = $10/16 \times 2^4$ which is -u000000000084
53.100000000000 = $100/128 \times 2^7$ which is w800000000087
Exception zero = 00000000000001

2. BASIC LOGIC (Refer to Figure 1, Page 6)

The Intercom 1000 system has two modes of operation--manual and automatic. The basic difference between the modes of operation is the source of input of commands. In the manual mode all commands are entered by the operator through the typewriter.

In the automatic mode of operation commands are picked up from memory in their proper order of execution by a device known hereafter as the command counter (cc). The cc is composed of the address of the commands to be executed.

Upon execution of OP code 69 (compute automatically) a patch switch is set in line 00 word 01 which takes control from the operator and gives it to the cc. Also, in the same sequence the address of 69ADDR is placed in cc giving it a beginning location for a sequence of commands. Thereafter, the execution of all commands will end with some modification of the address held in cc. Normally this address is incremented by one so that the next command in sequence is picked up, however, transfer operations are the exception. Upon execution of a transfer operation a new address is established in cc to form a new sequence of operations.

The remaining logic of intercom may be discussed independent of the mode of operation, that is, regardless of the mode of operation the interpretation and execution of commands is the same.

Referring to block 3 of the general flow diagram, (figure 1) note the command conversion from decimal to binary. During this conversion all commands that name index registers cause the overflow to be set, thus providing a convenient test to determine if a command is ready for execution or must it be changed by the contents of an index register.

Commands that name an index register may or may not require indexing. To elaborate, index register commands (OP 70, 71...77) should not have their address incremented. These commands are executed immediately.

Command with OP $<$ 70 that are modified by an index register and commands that do not name an index register funnel back to the point on the flow diagram where the OP code is tested to determine if it equals 49. All commands with this OP are executed immediately.

From this point the command may be further categorized into groups that possess features in common. For example, all arithmetic commands are treated as one category.

Special operations which include commands with OP codes within $49 <$ OP < 70 are in another category. These commands are not so critical as far as time is concerned, therefore, they are executed in line 19.

Another category of commands contain all output operations. A final category is transfer operation which requires the pickup of no data but may be conditional upon the contents of the A register. By grouping these, coding is convenient and time of execution is cut to a minimum.

Finally, execution of all operations, except transfers, end by transferring to 00.01 which directs control to the operator if in manual or to the cc if in automatic.

BASIC LOGIC

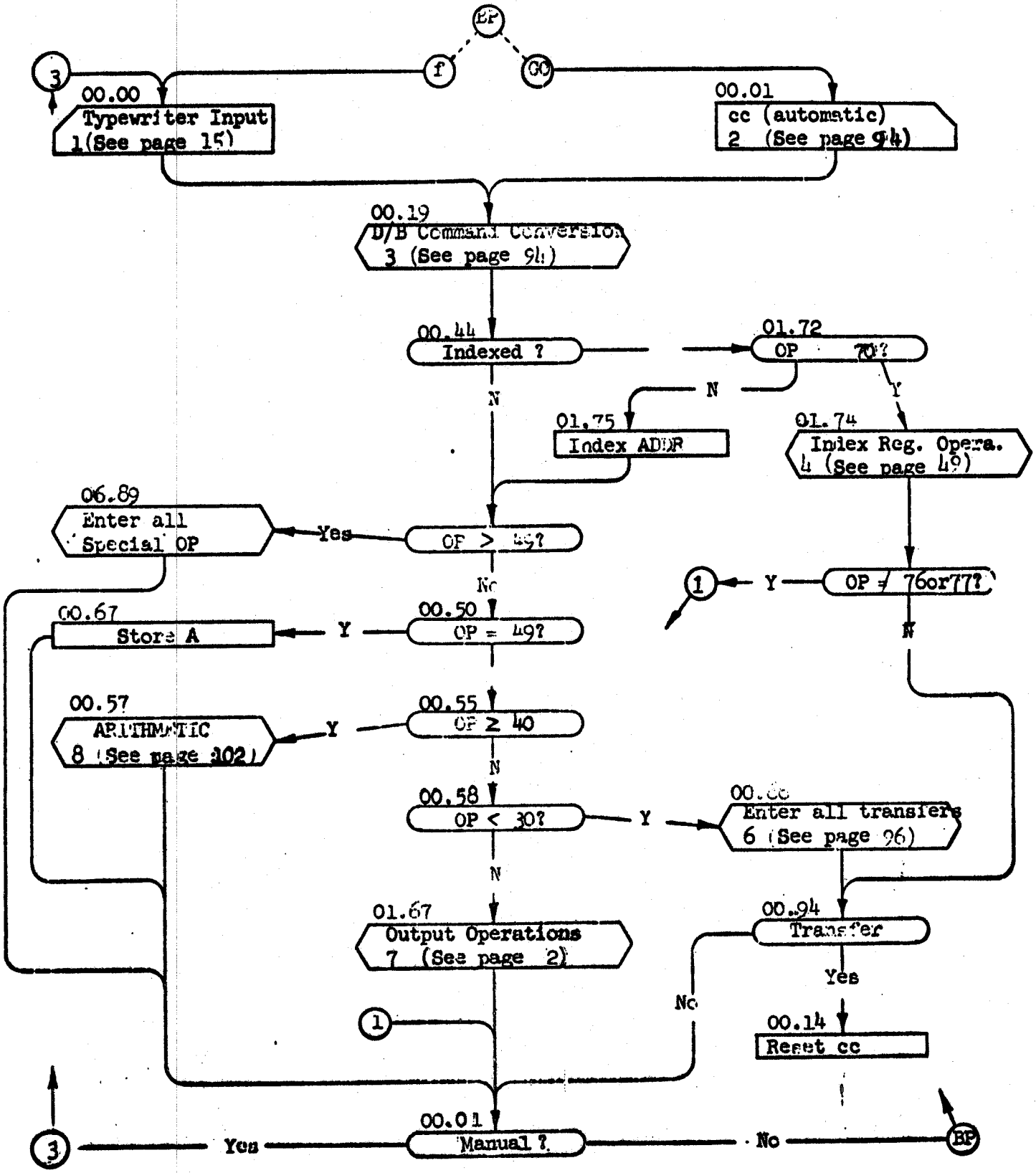


FIGURE I

3. COMMAND COUNTER

The command counter is the controlling element of automatic execution of programs. It holds the address of the last command executed at all times. Consequently, by adding one to the Command Counter, the address of the next command to be obeyed is formed. Therefore, the execution of every command, except transfer commands, must ultimately transfer to a position in Intercom where the Command Counter can be picked up and incremented by one. Command 01 of line 00 initiates this process by transferring the command counter (22.03) into PN. From there it is incremented by one, copied into AR, then copied back in its permanent storage location. The Command Counter, plus a dummy command, is obeyed in AR, effecting the pick-up of the appropriate command from memory. From this point, the command enters the decimal-to-binary conversion and, finally, the interpretation routine.

Transfer of control operations bypass the incrementing of the command counter and replace it with the address specified in the transfer command. This has the effect of establishing a new location in the memory as the position at which to proceed and obtain subsequent commands. Note, therefore, if the computer is in "Manual" the effect of a transfer command will always be to execute at least one command from the designated position in memory.

4. CONVERSION ROUTINES

The floating point input-output subroutines are stored in lines 02 and 03 with constants stored in line 07. These routines are coded to work as a three line package and may be extracted from the basic package if desired. The method used for the conversions is expressed mathematically as follows:

4.1 FLOATING BINARY CONVERSION TO FLOATING DECIMAL*

To find D, d from B, b such that $N = B \cdot 2^{b-128} = D \cdot 10^{d-50}$; where b, d are integers; $1/2 \leq |B| < 1$ and $10^{-1} \leq |D| < 1$ if $N \neq 0$; and $1 \leq b \leq 255$.

Method:

(1) Determine $d' = P + 50$, where P is an integral multiple of 8 such that $10^{P-8} < 2^{b-128} < 10^{P}$.

*See Figure II, Page 9

$$0 \le \frac{P}{8} + 4 < 5 + (\frac{b}{8} - 16) \log 2 < \frac{P}{8} + 5.$$

Let I denote the integral part of $\left[\frac{b}{8} \log 2 + (5-16 \log 2)\right]$; then $P = 8I - 32$ and $d' = 8I + 18$.

(NOTE: Program constants involving log 2 are approximations which give $I = 4$ when $b = 128$.)

(2) Compute $B' = (10^{-P} \cdot 2^{Q})\ B$ and $b' = b - (Q + 128)$, where $2^{Q} < 10^{P} < 2^{Q+1}$. If $P = 0$, then $B' = B$ and $b' = b - 128$.

(Ten values of $10^{-P} \cdot 2^{Q}$ are stored as a table in 07.72-91. Each occupies the 48 most significant bits of a double precision number whose least significant 9 bits contain $Q + 128$.)

$N = B' \cdot 2^{b'} \cdot 10^{d'-50}$, where $1/4 < |B'| < 1$ and $10^{-8} < 2^{b'} \le 1$.

(3) Let p denote the integral part of $|b'| \log 2$; then $10^{p} < 2^{-b'} < 10^{p+1}$, where $0 \le p \le 7$. Let the integer q be defined by $2^{q-1} < 10^{p} < 2^{q}$.

Compute $D' = (10^{p} \cdot 2^{-q})\ B' \cdot 2^{b'+q}$, where $-3 \le b' + q \le 0$.

If $p = 0$, then $D' = B' \cdot 2^{b'}$.

(A table in 07.u0-u6 stores seven values of $10^{p} \cdot 2^{-q} + q \cdot 2^{-28}$.)

Now $N = D' \cdot 10^{(d'-p)-50}$, where $\frac{1}{40} < |D'| < 1$.

(4) If $|D'| \ge 10^{-1}$, then $D = D'$ and $d = d' - p$;

If $|D'| < 10^{-1}$, then $D = 10\ D'$ and $d = d' - p - 1$.

INTERCOM 1000 D

Floating Binary Conversion to Floating Decimal

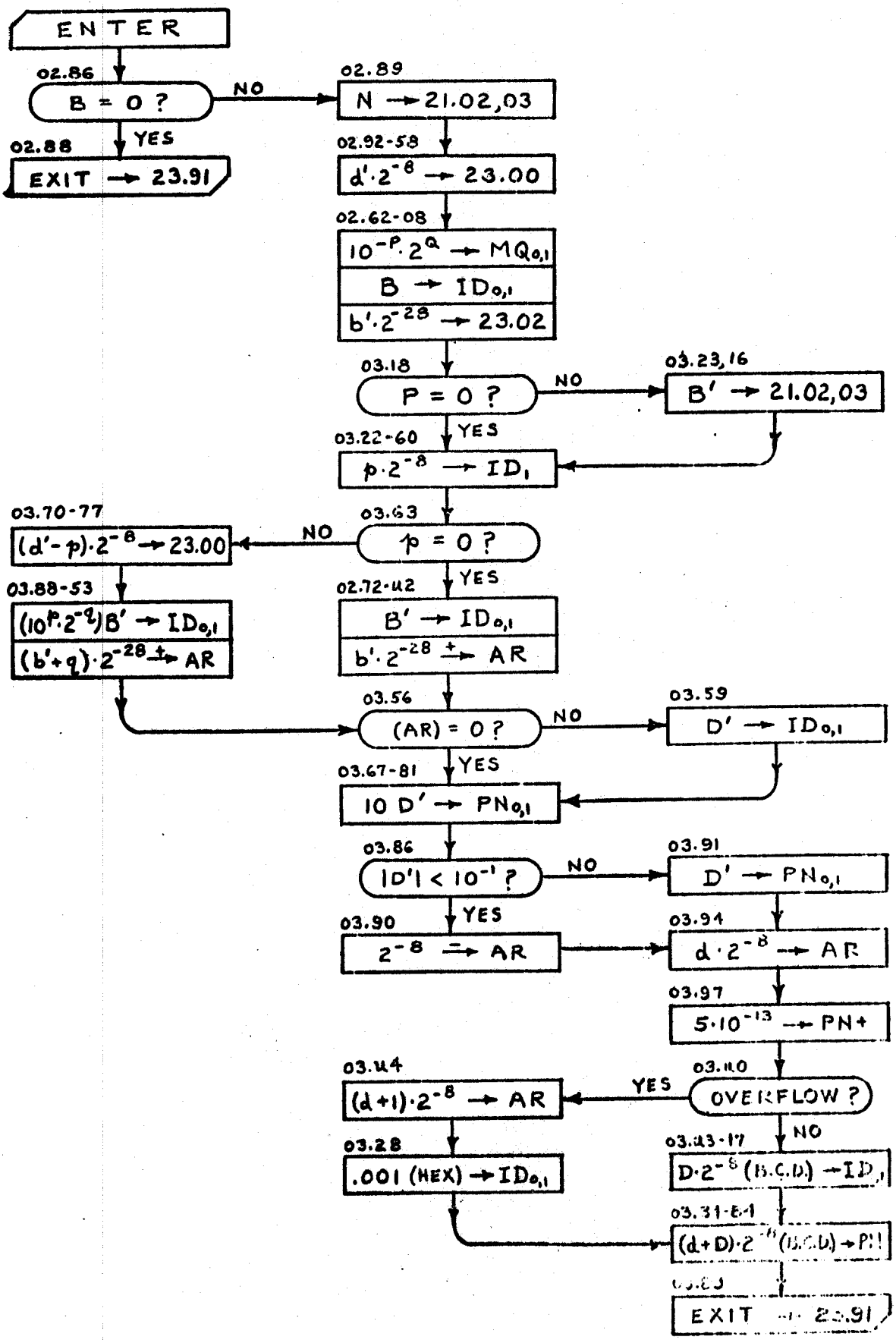


FIGURE II

4.2 FLOATING DECIMAL TO FLOATING BINARY

Given: Binary-coded decimal $N^* = d_1 \cdot 2^{-4} + d_0 \cdot 2^{-8} + 2^{-8} \sum_{i=1}^{12} 2^{-4i} D_{ij}$

where $10d_1 + d_0 = d$, $\sum_{i=1}^{12} 10^{-i} D_i = D$, $D_1 \neq 0$, and

$D \cdot 10^{d-50} = N$.

Find: Binary B, b such that $N = B \cdot 2^{b-128}$, where $1/2 \leq |B| < 1$

and b is an integer 1 b 255 if 2^{-128} N 2^{127}; i.e., if

$(.293873587706)\ (10^{-38}) \leq |N| \leq (.170141183460)\ (10^{39})$.

Set B = 0 if $|N| < 2^{-128}$, or indicate error if $|N| \geq 2^{127}$.

Method:

(1) Extract alternate digits from N^* and multiply by $10 \cdot 2^{-4} = 5 \cdot 2^{-3}$, giving $N^1 = (d = D') \cdot 2^{-8}$, where $D^1 = \sum_{j=1}^{6} (10D_{2j-1} + D_{2j}) \cdot 2^{-8j}$.

If $D' = 0$, then $B = 0$; otherwise conversion of D^1 to binary D is completed. $(d-10) \cdot 2^{-8}$ is formed; if negative, then $B = 0$.

(2) Let I denote the integral part of $\frac{d-10}{8}$. If $d \geq 90$, $|N| \geq 2^{127}$; therefore, either $0 \leq I \leq 9$ or an error is indicated.

Let $P = 40-8I$; $p = d-10-8I$. $0 \leq p \leq 7$, since $I \leq \frac{d-10}{8} < I + 1$.

Let Q and q be integers determined by $2^Q < 10^P < 2^{Q+1}$ and $2^{q-1} < 10^P < 2^q$.

Then $N = D \cdot 10^{d-50} = D \cdot 10^{-P+p} = \frac{D}{2} (10^{-P} \cdot 2^Q)\ (10^P \cdot 2^{-q}) \cdot 2^{-Q+q+1}$.

(3) The same tables of $10^{-P} \cdot 2^Q$ and $10^P \cdot 2^{-q}$ are used here as in conversion from floating binary to floating decimal. Since the least significant nine bits of each entry in the table of $10^P \cdot 2^Q$ contain Q + 128, let $B' = \frac{D}{2} (10^{-P} \cdot 2^Q)(10^P \cdot 2^{-q})$; $b^1 = 257-(Q + 128) + q$. $\frac{1}{60} < |B^1| < 1/2$.

(4) Before normalizing B^1 with adjustment of b^1 to give B and b, 2^{-54} is added to B', providing correctly rounded conversion of integers.

4.3 FLOATING DECIMAL TO FIXED DECIMAL (Refer to Page 53)

The output modes initiated by op codes 33 and 38 cause a number, x, (contents of ADDR) to be placed in ID and a return command to be placed in 23.03. Control is then transferred to the binary to decimal floating point conversion routine.

After conversion is completed control is transferred to line 01 where a format, contents of 07.07 and 07.08 for OP 38 or the contents of 07.06 and 07.07 for OP = 33, is placed in temporary storage. Control is transferred to the "Floating to fixed point" conversion routine.

In this conversion the binary representation of the characteristic E of X is compared with a constant H. The constant H is stored in 06.42 as a binary integer scaled 2^{-8} representing fifty plus the largest number of digits to be typed to the left of the decimal place. Intercom 1000 is issued with H = 57 (50 + 7) expressed as "3900000" in hexadecimal notation indicating that numbers are typed with 7 decimal places before the decimal place.

If $E < H$, the mantissa of X is placed in ID and shifted H-E digits. ID is copied into PN and exit is made to the typeout sequence (See page 16.6).

If $E > H$, X is too large to be typed under the control of the fixed point format, therefore, control is transferred to the appropriate floating point output sequence.

The value of H may be changed to provide for more or less digits before the decimal point. An accommodating format should be placed in the location mentioned in paragraph three of this page.

4.4 COMMAND CONVERSION (Refer to Page 94)

The decimal form of the command, taken from memory or the type-in sequence, is transferred to both sides of the PN register in order that the special extract command (3.23.31) can be used to separate the digits of the command. The conversion from decimal to hexadecimal is accomplished with four commands. As stored in memory, the commands have their OP, CH and Wd numbers one hexadecimal digit to the right.

If the tens portion of a number (OP, CH or Wd) is referred to as t and the units portion as u, the number may be expressed as t + u/16. What is required of the conversion is to put it in the form of 10t + u which is the hexadecimal equivalent to tu.

Starting with t + u/16, the units portion of three numbers is extracted and added to the original form of the command in AR producing the sum t + u/8. The next command shifts the entire contents of the accumulator two binary digits to the left which multiplies the number by four. As a result, the accumulator now contains 4t + u/2. At this point, the tens portion of the three numbers is added to the accumulator producing 5t + u/2. One additional bit shift to the left produces 10t + u, which is the hexadecimal form of the respective numbers in the command. Note that during the conversion the numbers changed positions so that they are now in the correct position with the exception of the source (CH) number which is one binary digit to the right. To provide the additional shift of CH, the converted number is placed in PN and CH extracted into ID and then ID is added into AR. This has the effect of shifting it one binary position to the left and now all portions of the command are in the proper position. Note that 1/2 the real hexadecimal value of CH is left in ID. This convenient location is utilized later to form the address of the store command. (See Page95)

5. TYPEWRITER INPUT (Refer to Figure 3, Page 15)

All input from the typewriter goes through a common input routine. The handling of the input by Intercom is controlled by program switches set by the command initiating input and the operator.

The control mechanism (operated from line 05) gates typing and then, when typing has been completed, checks (19.00) to see if anything has been sent there by a slash. Line 19 contains line 06 during input. Word 00 of line 06 is zero unless input is to be obeyed. If (19.00) is not clear, (23.00) is executed. If (19.00) is clear, a second location is tested for zero to determine if input is a command or a number. The results of the second test switch determine which conversion routine to enter before the input is stored.

The same control mechanism is used for the input of "fixed point numbers" except in this case operation is from line 19 instead of line 05. By operating from line 19, the slash has a different meaning. When a slash or decimal point is typed with the input, the command line is precessed four words and (23.00) is placed in 19.00. Location 23.00 contains the number of digits before the decimal point; therefore, when typing is completed (19.00) is used to determine the size of the exponent to be attached to the input.

If no slashes are typed, the input control operates as if it were in line 05, and the input is treated as if it were floating point decimal.

The typing of two slashes causes line 19 to be precessed 8 words. This is used as a control to initiate the execution of input.

5.1 NORMALIZATION OF INPUT

Before typing is gated for the input of numbers, a flag* is placed in short line 23 to provide an indication of the number of digits typed by the operator. It also provides a convenient indication of a full fourteen digit input. If the input is less than fourteen digits the leading zero's and the flag are shifted off introducing the required number of trailing zero's.

5.2 INPUT SEQUENCE

After conversion of numbers to binary floating point, control is transferred to the floating point output conversion which reconverts the number , then control is transferred to the typeout sequence which end with a "return to Mark" taking control to a looping zero test of line 19. Two drum cycles after line 19 typeout precesses all the non-zero information out of line 19; ready is set, stopping typeout, and exit is made to the "store input sequence."

Once input has been classified as a command, it must go through a simple conversion routine before it is stored or executed. After the command is converted from KOPCHWD to KWDOPCH, (19.00) is tested again. If (19.00) is not zero the converted command is transferred to the command interpretation logic where it is executed.

*Words 01-03 of line 23 are set to zero and the constant 0000004 is placed in location 23.00.

If (19.00) is zero the store sequence is entered. The input (number or command) is copied into line 23 and a store command is formed and obeyed.

Immediately following the typeout and storing of numbers or commands a test is executed to determine whether operation is automatic or manual. If in automatic, the normal exit to 00.01 is executed. If in manual, the address of the store command is incremented by one for command input or by two for number input. During the incrementation the increment is made 156 (hexadecimal 9W) too large so that if the decimal equivalent passes from 99 to 00 module 100 an end-around carry in AR will result causing an overflow. This is tested and either the excess 156 is subtracted backout or the hundreds digit is incremented.

TYPEWRITER INPUT

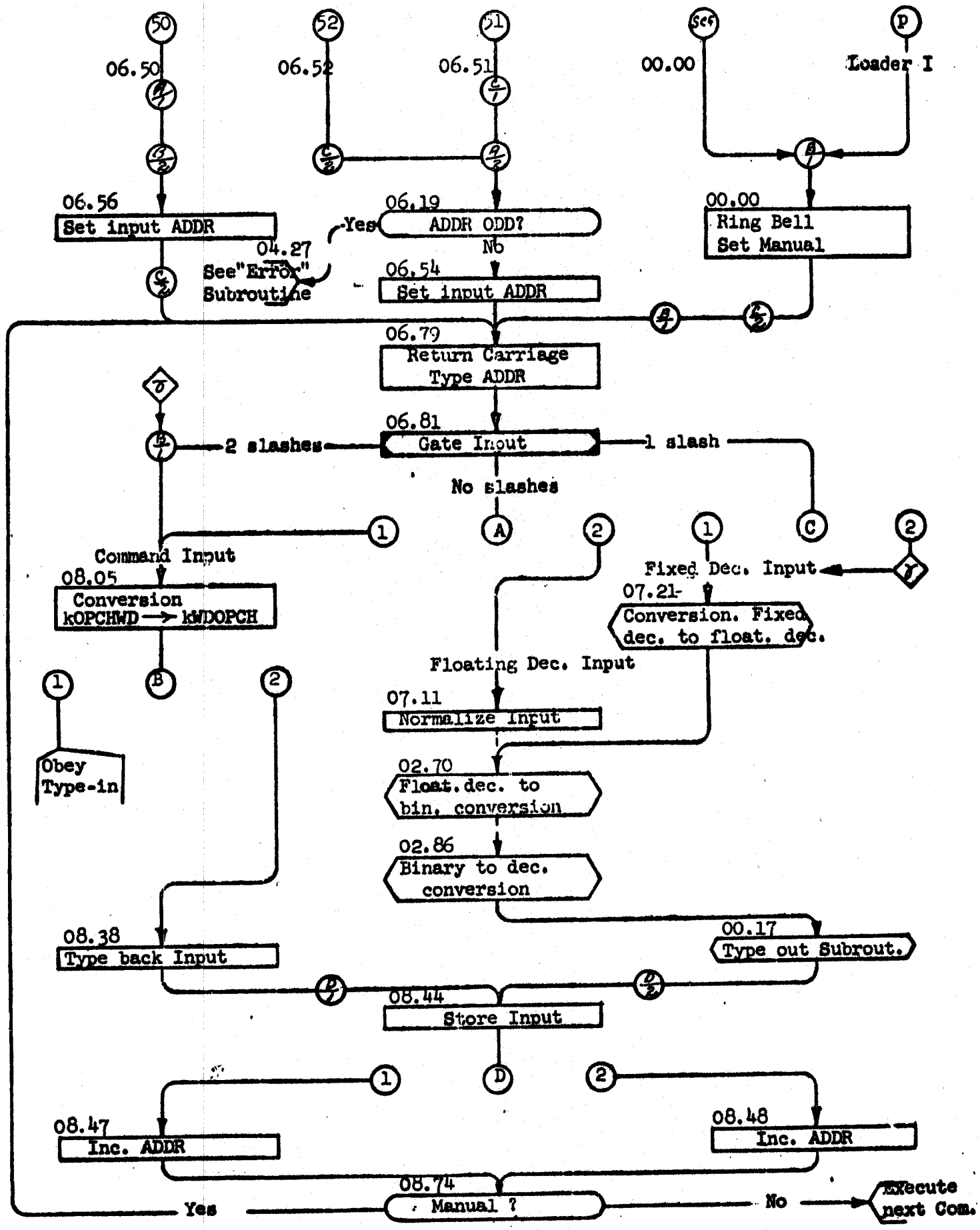


FIGURE III

6. TYPE-OUT SEQUENCE (Refer to Page 98)

The type-out sequence begins with a test for Ready. As soon as Ready is set a "type line 19" command is executed. During the next drum cycle a format previously stored in $MQ_{0,1}$ is transferred to 02.02-03 and the sign of the output is placed in 19.u7. During the next drum cycle no function is performed. The delay is needed to allow time for the sign type-out before the clearing of line 19. This follows during the next drum cycle. Finally, the absolute value of the output which has been stored in PN previously is transferred to 19.u6-u7. The sequence is ended by a return transfer to some word between u4 and 01, or word 01, of line 00.

During the number input mode the sequence returns to word u4 of line 00. This location begins a sequence of commands which repeatedly tests line 19 for zero. When line 19 becomes zero, ready is set. This suppresses the type-out of trailing zeros.

7. ERROR INDICATIONS (Refer to Page 75)

The error indicator sequence transfers an AR type-out subroutine into line 23 and a format of the form "wait, period, end" into 03.03. The constant (zzzzz00) is copied into AR. A mark transfer command is executed to transfer control to line 23. In line 23 a return to mark is executed after AR is typed under the format named above, and the bell rings. Typing AR under the control of this format precesses AR one digit (4 bits). Upon exit from the type-out routine AR is tested for zero. If AR is not zero, again control is transferred to line 23 without reloading AR. Consequently, after 5 periods are typed and the bell has rung 5 times, AR will become zero as a result of precession and the sequence halts.

Upon termination of the halt state, by placing computer switch to GO or BP, control is transferred to the "Type Location of Last Command Executed" sequence which results in the type-out of the address held in cc. This is the location of the last command executed.

8. LOADER I (Refer to Figure 4, Page 18)

The basic package loader operates in lines 23 and 04. After executing commands to turn off the DA-1* and reset the overflow, it controls the loading of the basic package and sets the extractors -.zzzzz00 and -.zzzzzzz into 20.00 and 20.01, respectively. Index registers, Interpretive Memory and appendix routine are not disturbed.

* The turn-off DA-1 command may initiate a CA-2 operation.

Loading of all the long lines is under the control of a loop which loads the lines in the order 01,02, 03, 06, 07, 08, 00 and 04. The loop provides for reversing and re-reading of tape if the channel sum is not correct. Upon termination of loading, the above extractors are set in line 20, line 04 is transferred from line 19 and control is set to the manual mode.

The check sum of each line is contained in the loader, therefore, if any changes are made to the basic package the loader must be changed also. In the following list the column on the left is the word in the loader that contains the check sum of the line listed in the right column in the corresponding position:

Word 61	Line 00
Word 49	Line 01
Word 51	Line 02
Word 53	Line 03
Word 63	Line 04
Word 55	Line 06
Word 57	Line 07
Word 59	Line 08

LOADER I

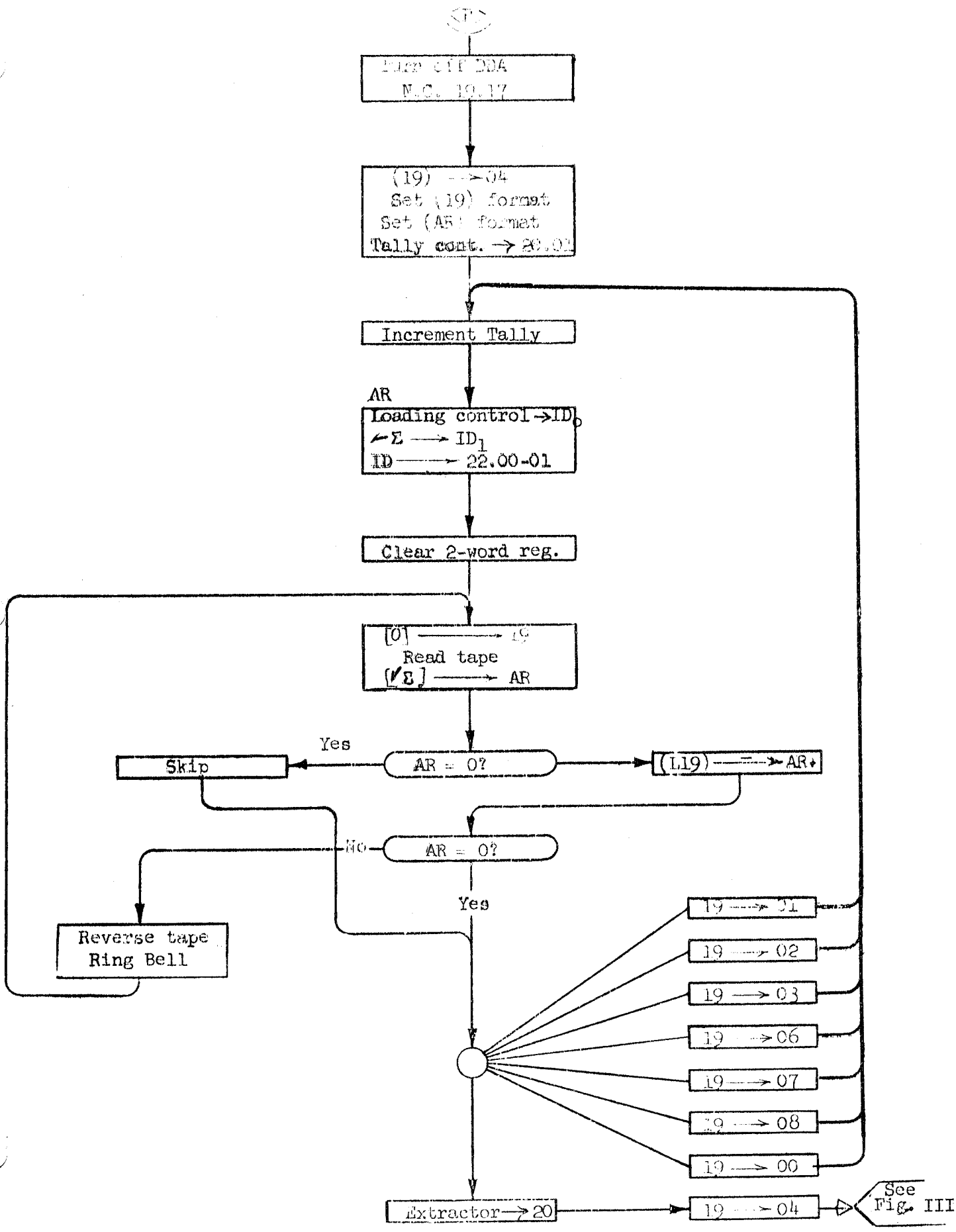


FIGURE IV

9. ARITHMETIC OPERATIONS

Referring to Figure 1 note that all arithmetic commands are placed in a common block for execution. Upon entry to this block control is transferred to line 04, the address is tested for an odd number. If the address is even, a dummy is added to the command and an exit command is placed in 23.03. The modified command is executed in AR effecting the transfer of an operand (the contentsof the specified address) into ID. The next command obeyed is the first command of the appropriate arithmetic operation.

If the above test determines the address is odd, exit is made to the error sequence.

9.1 CLEAR AND ADD k42CHWD

The contents of ID are copied into the A register and control is transferred to 23.03.

9.2 ADD (Refer to Page 102) k43CHWD

The number in ID is added to the contents of A. The mathematical expression of this addition follows.

Given two numbers which are to be added:

$$A = \pm A_m \cdot 2^{(A_E - 128)}$$

$$X = \pm X_m \cdot 2^{(X_E - 128)}$$

when $A_E \leq X_E$

$$A + X =$$

$$\pm A_m x 2^{(A_E - 128)} + \pm X_m x 2^{(X_E - 128)} =$$

$$\left(\left[(\pm A_m x 2^{(A_E - X_E - 1)} \right] + \left[\pm X_m \cdot 2^{-1} \right] \right) x 2^{(X_E - 127)} =$$

$$\pm T_m \cdot 2^{(T_E - 128)} = T \qquad \text{and } 0 \leq T_m < 1$$

The initial contents of A is labeled A and the operand originally in ID is X. When $A_E < X_E$ their positions are reversed. T is denoting the total or sum.

First A_E is compared to X_E to determine whether it is necessary to interchange X for A. Then the difference between A_E and X_E is tested to determine if the difference is greater than 2^{-48}. A difference this great indicates that the large number is not affected by the addition of the smaller number, therefore, the larger number is placed in in the A register. This test provides a one drum cycle execution time for the addition of numbers with a large difference in value or the sum of two numbers, one of which is equal to zero.

If $|A_E - X_E| < 2^{-48}$ X_m is copied into 21.02-03 (X register) and A_m is copied into ID for scaling. A scale factor which equals $(A_E - X_E - 1)$ is formed in AR. This scale factor controls the denormalizing of A by controlling the shift of A in ID. The denormalized value of A is stored in the A register. At this point X_E (larger E) is copied into AR and X_m is copied into ID. ID is shifted one bit scaling X_m times 2^{-1} and incrementing AR thereby producing a candidate for T_E in AR. The scaled mantissa in ID is added to the denormalized mantissa held in A to form a sum in PN. The absolute value of the sum is transferred to MQ and the sign is set in IP. At this point it is necessary to normalize the results. This could reduce the value held in AR, therefore, (AR) is replaced by its negative complement, then exit is made to the normalize routine which is common to add, multiply, and divide.

9.3 NORMALIZE (Refer to Page 104)

Upon entry the number held in MQ is normalized. The incrementation of AR during the normalize process decrements the absolute value of the characteristic. Letting d equal the number of bits the mantissa is shifted,

$$\pm T_m x 2^{(T_E - 128)} = (T_m x 2^d) \; x 2^{(T_E - 128 - d)}$$

however, the quantity $(d > (T_E - 128)$ is defined as an underflow. This condition can be determined by testing the sign of AR. If AR is positive a floating point zero is copied into A and exit is made to 23.03. If AR is still negative, a test must be made to determine if $(T_E - 128 - d) > 2^{255}$ or if an overflow exists. This test is made by adding zzzzz00 to the uncomplement value held in AR. This addition would cause a sign change in AR in an overflow exit.

If AR changed sign the arithmetic operation just performed generated a number greater than 89.170141183460 thereby generating an overflow. This condition causes exit to be made to the error indicator sequence.

If an overflow or underflow condition does not exist MQ or T_m and the sign of T is copied into A and finally T_m and T_E are combined in the A register to form the proper result of the arithmetic operation and exit is made to 23.03.

9.4 CLEAR AND SUBTRACT (Refer to Page 102) k40CHWD

A negative number is transferred to MQ effecting a change of the sign held in IP. This reverses the sign of the operand held in ID. The contents of ID are then copied into A.

9.5 SUBTRACT (Refer to Page 19.9.2) k41CHWD

The sign of the operand is changed as explained in the operation above and then the add sequence is entered.

9.6 CLEAR AND ADD ABSOLUTE VALUE k45CHWD

The number in ID is copied into the A register using an odd characteristic. This ignores the sign held in IP, thereby effecting a copy of the absolute value into A.

9.7 MULTIPLY (Refer to Page 104) k44CHWD

A test is made to determine if either the multiplicand (A) or the multiplier (x) is zero. If so, a floating zero is placed in A before exit is made. Otherwise, the operation is summarized as follows:

$$A \cdot X = (A_m x 2^{(A_E - 128)}) \times (X_m x 2^{(X_E - 128)}) =$$

$$A_m x X_m x 2^{(A_E + X_E - 256)} = T_m x 2^{(T_E - 128)} = T$$

Where $T_E = A_E + X_E - 128$

The product T_m is copied from PN into MQ and its sign is held in IP. The negative complement of T_E is copied into AR and exit is made to the normalize sequence.

9.8 DIVIDE (Refer to Page 105) k48CHWD

Before execution of the divide operation, A is tested for zero to allow a fast exit. If A is not zero, X is tested for zero to prevent division by zero. If x is zero, exit is made to the error indicator. If x and A are not zero, the following expression is evaluated.

$$\frac{A}{X} = \pm \frac{A_m 2^{(AE - 128)}}{X_m 2^{(XE - 128)}} \left[\frac{A_m}{X_m} x 2^{-1}\right] x\ 2^{(AE - X_E + 1)} =$$

$$T_E x 2^{(T_E - 128)} = T$$

The routine produces $T_m . 2^{-1}$ in MQ as a result of the division. T_E is copied into the AR in its negative complemented form and exit is made to the normalize sequence.

9.9 STORE (Refer to Page 95) k49CHWD

Referring to "command conversion" the source held in ID is shifted four bits to form the destination channel. This is added to the WD held in 21.03 to form a store command. The contents of A are placed in ID and the store command is executed causing ID to be copied into the designated position in memory.

10. TRANSFER OPERATIONS (Refer to Page 96)

10.1 TRANSFER CONTROL k29CHWD

The address held in the cc is changed to the address of the subject command. This has the effect of establishing the entry for a new sequence of commands.

10.2 TRANSFER IF ACCUMULATOR = 0 k23CHWD

The odd side (most significant side) of the A register is tested. If A is in zero the "Transfer Control" sequence is entered. If not, exit in normal.

10.3 TRANSFER IF ACCUMULATOR < 0 k22CHWD

Minus the least significant side of A and the sign is copied into AR and the sign of AR is tested. If it is positive, entry is made to the transfer sequence. If not, exit in normal.

10.4 TRANSFER IF ACCUMULATOR ≥ 0 k20CHWD

The sign of A is copied into AR and the "Transfer if Accumulator<0" sequence is entered.

10.5 MARK PLACE AND TRANSFER I k26CHWD

The contents of cc are placed in location 06.u3 and the routine enters the transfer control sequence.

10.6 MARK PLACE AND TRANSFER II k28CHWD

The contents of the cc are placed in location 00.u3 and the routine enters the transfer control sequence.

10.7 RETURN TO MARK PLACE I 160000

The cc is replaced by the contents of 06.u3 then control is transferred to 00.01 (refer to Figure 1).

10.8 RETURN TO MARKED PLACE II 180000

The cc is replaced by the contents of 00.u3 then the normal exit is made, namely, control is transferred to 00.01 where the cc is incremented and obeyed.

11. INPUT-OUTPUT OPERATIONS

11.1 TYPE COMMAND FROM MEMORY k35CHWD

The contents of the address designated are typed from AR as they appear in memory, namely, KWDOPCH. The format controlling this output is used during the "Permit Command Type-in" operation, therefore, it is necessary that it end with a tab.

11.2 NUMBER TYPE-OUT OPERATIONS (Refer to Pages 52, 7.4.1,11.4.3)

The execution of OP codes 32, 33, 34 and 38 begins with the setting of a return command in location 23.03, then the operations funnel into a common sequence of commands, namely, the floating-binary to floating-decimal conversion routine explained on page 7.4.1. Upon completion of conversion the pre-set return command is obeyed effecting the proper exit. To complete the execution of OP codes 32 and 34 the proper format (locations 08.u4-u5 for OP 32 or locations 08.u5.u6 for OP 34) is copied into MQ and the entry to the type-out sequence is executed. The type-out sequence ends by returning control to word 01 of line 00.

The execution of OP codes 33 and 38 require a conversion of the floating-decimal number to a fixed-decimal number. Upon completion of this conversion the proper format (explained on page 11.4.3) is set into MQ and, as in the paragraph above, the type-out sequence is entered.

11.3 POSITION TYPEWRITER PAPER (Refer to Pages 51 and 97) k30CHWD

The WD portion of the address is extracted into AR as a negative number scaled 2^{-28}. This value of AR is used to control the shift of a bit in ID. After the AR controlled shift of ID, line 19 is cleared, ID is stored in 19.u6-u7 and a type line 19 command is executed controlled by a "carriage return-end code" format. Each carriage return shifts the bit in ID one bit position. If the shift does not clear 19, the end code is interpreted as a reload and the cycle is repeated. While the carriage returns are being executed a "tab-end code" format is being established and CH is replacing the previous function of WD. When all required tabs are executed the contents of ID (now the control of tabs) are placed in line 19. Once the tab control is placed in 19 and output is executed exit is made to 00.01.

11.4 TYPE TABULATING NUMBER (Refer to Page 106) k31CHWD

The incremented address of the "Type Tabulating Number" command is converted to a four digit decimal number and typed from AR followed by a tab.

The conversion of the number from binary to decimal is done by a series of subtraction of 10 from WD, accumulating the decimal equivalent of the number of times ten will divide into WD. The final subtraction leaves a remainder which is equal to the decimal units digit of WD. The same loop is entered to convert CH and finally CHWD is typed from AR as a four digit number followed by a tab.

By setting CHWD equal zero, an index register (K) in question may be typed by the execution of this command. This is particularly useful during debugging operations.

11.5 READ PUNCHED TAPE (Refer to Page 74) k55CHWD

The "Read Punch Tape", like all input-output operations, must wait for other input-output operations to terminate before execution. When execution is possible, line 19 is cleared and a read paper tape operation is initiated. During read-in a command is formed in AR using the incremented address of the subject command. When read-in is completed the command in AR is executed at word time u7, effecting the transfer of words 00 up to word WD of line 19 to line CH.

11.6 PUNCH PAPER TAPE (Refer to Pages 52, 107) k39CHWD

As soon as the computer is in a state to accept an output command, line 19 is cleared and words 00 through WD-1 are copied from line CH to line 19. Words 19.00-03 are tested for zero. If zero, a non-zero number is placed in word 00; then, as in the case where 19.00-03 was not zero, the tape leader punch sequence is entered. A complete description of the method used is explained in detail in Technical Applications Memorandum No. 33. Finally, line 19 is cleared of leading zero's using the "type AR - set ready" method of precessing line 19 and the rest of line 19 is punched on tape.

As soon as precessing is completed, control is transferred back to 00.01 so that computation can continue during punch out.

12. SPECIAL COMMANDS

All the operations listed under the subject heading were not programmed for maximum speed. It was determined that these operations would not be executed as part of programs very often, therefore, these commands were written to be executed out of line 19. This means that they must wait for the completion of any input or output operation before their execution.

12.1 RING BELL 630000

Self explanatory

12.2 HALT AND PERMIT MANUAL OPERATION 670000

A zero is placed in word 01 of line zero. Hereafter normal exits from operation return to manual.

12.3 PERFORM SUBROUTINE k08CHWD

When the input-output circuitry is in a ready state, line CH is transferred to line 19 and entered at word WD. Upon completion of execution, control is restored to Intercom by transferring control back to 00.01.

A companion OP code (02) is available that operates faster, in that it does not require the input-output circuitry to be tested; that is, computation of subroutines perform by OP 02 may continue during input-output operation. Subroutines to be executed by code 02 must be programmed to operate out of line 05. Note, the execution of OP 02 destroys the Selective List Routine, which may be in Line 05.

12.4 TYPE LOCATION OF LAST COMMAND EXECUTED 060000

Referring to the definition of cc, it can be shown that this device holds the location of the last command executed in the automatic mode. This location is extracted into 21.03 and the "Type Tabulating Number" routine is entered resulting in the desired type-out.

13. INDEX REGISTER OPERATION (Refer to Pages 50,51)

13.1 SET INDEX REGISTER

Each of the six set index register commands places a command or switch in PN. The T number of the command in PN corresponds to the location of a particular register being set, then entry is made to a common sequence of commands.

In the common sequence of commands, the address portion of the command being obeyed is extracted and placed in MQ. The command is formed to store the index register setting by adding the K value to the destination of the command previously placed in PN. This store command is used to transfer the proper setting to the appropriate register.

13.2 INCREMENT INDEX REGISTERS

After the setting of initial conditions both index register incrementation commands utilize the same sequence of operations.

The contents of the B, D and L registers are transferred from memory to a short line. The contents of B and D are added together to form a new value of B. If the new B is less than or equal to the value of L the value of B is stored and the transfer sequences entered.

If B is greater than L the incremented B value is stored and the normal exit is executed.

13.3 DECREMENTING

The "Increment Register" commands may be used for decrementing if the D register is made negative before the increment command is executed.

To negate a D register it is necessary to execute a "Clear and Subtract" and a "Store" command whose addresses are the address of the D register.

With the D register set to a negative value the execution of the increment command initiates the following sequences of events:

1. The number in D is subtracted from the number in B and the results are placed in B.

NOTE: The new contents of B must never be allowed to become negative.

2. If the new contents of B are greater than the number in L, the next command in sequence is obeyed.

3. If the new contents of B are less than or equal to the number in L, control is transferred to the address shown in the increment command.

Example a:

Index register B_{wd} of set 4 contains 14; D_{wd} of set 4 contains the number 2; L_{wd} of the same set contains 4.

After D_{wd} is set to a negative value by the execution of the commands 0 40 12 u2 and 0 49 12 u2, the command 4 76 10 13, stored in location 09.26, places 12(14-2) in B_{wd}; as this is greater than the number 4 in L_{wd}, the next command in sequence is obeyed.

Example b:

L_{wd} contains 16; other conditions as in Example a. After execution of 4 76 10 13, stored in 09.26, B_{wd} contains 12; as this is less than the number 16 stored in L_{wd}, control is transferred to the command in 10.13.

13.4 REPLACING

The floating point arithmetic commands, "Clear and Add" and "Store", are used for replacing the contents of one index register with the contents of another.

The following is an illustration of how these commands are used to replace registers of set one. To replace registers in sets 2, 3 - 9, substitute 10, 11 - 17, respectively, for 09 in the address of the commands.

The execution of the commands:

1. 4209u2 and 4909u4 replaces the contents of B_{wd} and B_{ch} with the contents of D_{wd} and L_{wd}.

2. 4209u2 and 4909u5 replaces the contents of B_{ch} with L_{wd}.

3. 4209u2 and 4909u6 replaces the contents of D_{ch} and L_{ch} with the contents of D_{wd} and L_{wd}.

4. 4209u2 and 4909u7 replaces the contents of L_{ch} with the contents of L_{wd}.

5. 4209u4 and 4909u2 replaces the contents of D_{wd} and L_{wd} with the contents of B_{wd} and B_{ch}.

6. 4209u4 and 4909u3 replaces the contents of L_{wd} with the contents of B_{ch}.

7. 4209u4 and 4909u6 replaces the contents of D_{ch} and L_{ch} with the contents of B_{wd} and B_{ch}.

8. 4209u4 and 4909u7 replaces the contents of L_{ch} with the contents of B_{ch}.

9. 4209u6 and 4909u2 replaces the contents of D_{wd} and L_{wd} with the contents of D_{ch} and L_{ch}.

10. 4209u6 and 4909u3 replaces the contents of L_{wd} with the contents of L_{ch}.

11. 4209u6 and 4909u4 replaces the contents of B_{wd} and B_{ch} with the contents of D_{ch} and L_{ch}.

12. 4209u6 and 4909u5 replaces the contents of B_{ch} with the contents of L_{ch}.

13.5 OTHER OPERATIONS (Refer to Page 50)

Also available in Intercom 1000 double precision are two operations very useful for replacing index registers. These operations are by-products of the system and are not available in the single precision package.

13.5.1 CLEAR AND ADD INDEX REGISTER** k78000D*

The number in register D of set k is copied into a special accumulator (Index Register Accumulator). Its machine location is 22.02.

Example:

The execution of 3780001 copies L_{wd} of set three into index register accumulator.

13.5.2 STORE INDEX REGISTER ACCUMULATOR** k79000D*

The number in the index register accumulator is stored in register D of set k.

In order for this operation to function properly 02.83 (6701708) must be changed to y601708.

Example:

The execution of 4790002 copies the index register accumulator into B_{wd} of set four.

*The value of D is given below:

D	Register
0	D_{wd}
1	L_{wd}
2	B_{wd}
3	B_{ch}
4	D_{ch}
5	L_{ch}

**The two operations above must be executed as a pair of commands, that is, OP 79 should be executed immediately following OP 78.

13.6 DO NOTHING (Refer to Page 99) 0 00 0000

An OP 00 is interpreted as a ring bell command. Its execution time is three drum cycles.

13.7 BLOCK COPY (Refer to Page 54) k81CHWD

The contents of words u2-u7 and 00-(WD-1) of channel CH are copied into the corresponding words of channel (08+k).

Example:

The execution of 2 81 11 u2 transfers words u2-u7, and 00-u1 of channel 11 into words u2-u7 and 00-u1 of channel 10(08+02=10).

The execution of 3 81 29 u2 transfers zeros (source 29) into every word, including index registers, of channel 11 (8+3).

The execution of 5 81 29 00 transfers zeros (source 29) into every register of set 5.

14. TIMING

The automatic mode of operation begins its cycle at word 01. Command pick-up is executed at word 16. Command conversion begins at word 19. If the overflow is set during the conversion of the command, the contents of a B register must be added to the converted command. The addition of a B register adds a drum revolution to the execution time.

If the overflow is not set, or after the B register has been added to the converted command, a series of tests are made to determine if data is to be picked up. Commands that require data pick-up require an additional drum revolution for execution. The execution of the command subroutine ends at word u5.

Operation in the Selective Print Mode adds an undetermined number of drum revolutions to the execution time.

The following is the quotation of the time required to execute the Intercom 1000 commands. The execution time includes command pick-up, data pick-up, command conversion and interpretation and the execution of the command subroutine. Time is quoted in drum revolutions.

This information may be used to make time estimates for program running time.

Command	Execution Time	
	Address Incremented?	
	No	Yes
Clear and Add	3	4
Clear and Subtract	3	4
Clear and Add Absolute Value	3	4
Add	5	6
Subtract	5	6
Multiply	5	6
Divide	5	6
Inverse Divide	5	6
Store	3	4
Transfer	2	3
Transfer on Zero	2	3
Transfer on Negative	2	3
Transfer on Positive	2	3
Set Index Register	2	-
Increment Index Register	3	-
Mark Place	2	3
Return to Mark	2	-
Floating Point Output	8	9
Fixed Point Output	11	12

The time for the execution of any arithmetic command that addresses the A register may be reduced one drum revolution if the address of A is written as 21.76 instead of 21.00

One drum revolution is required for the pick-up of data or commands; therefore, it is not possible to have badly accessed commands or data.

15. SELECTIVE LISTING

The listing subroutine is stored by Loader II in words 00-75 of line 05. Upon read-in of this routine by Loader II, two patches are placed in line 06, words 61 and 62, to provide a transfer to line 05 upon execution of OP codes 61 and 62.

Upon execution OP 61 (Set Selective Print), a patch is placed in line 00, line 23 is cleared and type-in is gated for the input of First Selector and Second Selector. When typing is terminated, Second Selector is picked up out of 23.00 and converted to the form common to all commands stored in memory, namely $D_1D_2D_3D_4D_5D_6D_7$ is converted to $D_1D_6D_7D_2D_3D_4D_5$. Finally, the First Selector is picked up out of 23.01 and converted, as was the Second Extractor, and stored in line 05. Control is then transferred to word 01 of line 00.

Word 01 of line 00 is a patch switch which determines whether operation is automatic or manual. When in manual, this word is equal to zero, therefore the next command in sequence is 00.00 (Ring Bell) which starts each manual command input cycle.

When the command is executed to "Compute Automatically", a patch (03.04.0.22.26-) is placed in 00.01 and a command counter is formed in 22.03. Thereafter, the execution of all commands, except transfers, end by transferring to this switch. Note the patch switch contains a B.P., hence, when operating in the automatic mode, or while listing, any interruption of execution should start by placing the compute switch in the B.P. position. When computation halts, the lights on the computer should read source 22, destination 26. At this point place compute switch in off position, enable switch in on position and type "f". This returns control to manual operation.

Failure to halt computation at B.P. may result in mass destruction of Intercom. This is always the case when using the list routine. The reason being that the list routine sets several patches in the basic control lines and these patches are all removed if computation is allowed to proceed to the B.P.

The patches mentioned above allow the list routine to use the conversion and type-out operation available in Intercom, thereby minimizing the logic required for the list sequence.

When the selectors have been set by the execution of OP = 61 and the command is given to compute automatically, each command is stored in its binary form and its decimal form before it is executed. Using the Second Selector certain specified digits of the command to be executed are extracted and compared with the First Selector to determine whether the command is to be listed. If so, certain patches are set in line 00 to bring control back to the list routine after execution of the command being tested.

After execution of the command control is transferred back to line 05. In line 05 the cc is used to form the information necessary to type out the location. Following the location type out KOP is extracted from the decimal form of the command and typed out. The incremented address portion of the command is converted to decimal and typed. Finally, the contents of the A register are converted to floating decimal and typed. After the last type-out patches are removed from line 00 and the cycle is repeated.

Execution of OP 62 removes all the switches and returns Intercom to the normal mode of operation.

16. SPECIFICATIONS FOR WRITING APPENDIX SUBROUTINES

16.1 EXECUTION

Subroutines may be executed from line 19 or words 75-u7 of line 05. Words 00-74 of line 05 are used as permanent storage of the "Selective Print" routine. However, it would be an advantage to execute subroutine from all of line 05, thereby eliminating the necessity for the execution of routine in the input-output buffer (line 19). An OP code (02) is available for this purpose. Note, however, that execution of OP 02 will destroy the selective print subroutine. To prevent this destruction during list operations, a test is built into the selective print routine that halts operation when an OP 02 is encountered.

The obvious advantage in the use of OP 02 is that execution of commands may continue during input-output operations. This is not true for OP 08.

It is advised that programs be written and checked out using OP 08. Then, when check out is completed, reposition the subroutines to operate out of line 05 or replace the subroutines with those from appendix package II and replace OP codes 08 with OP codes 02.

16.2 STORAGE

Any line of interpretive memory (line 09, 10-18) may be used for the storage of machine language subroutines. Lines 09-17 contain index register 1-9 in words u2-u7. Therefore, a subroutine that uses these words should be stored in line 18 if possible, otherwise the storage of the subroutine will destroy a set of index registers.

16.3 SHORT LINES USED

Upon entering a subroutine the short lines will contain the following:

	00	01	02	03
Line 20	-zzzzz00	-zzzzzzz		007z000
Line 21		- A -		Converted * Command
Line 22				cc **
Line 23				

Words 20.00, 01, 22.03 must be returned to their original state at the end of appendix subroutine if any changes have been made to their contents during the operation.

*The "converted command" is the machine language representation of the incremented Intercom command.

**Command Counter

16.4 ENTRY

Any word may be used as the entry to the appendix subroutine. The WD number of the command to execute the subroutine is the decimal location of the first machine language command of the subroutine. The CH number is the line in which the subroutine is stored.

16.5 DATA INPUT

The A register is in 21.00-01. Other information will be found in the short lines as indicated.

16.6 EXIT

Upon exit the overflow should be left in a reset condition and the short lines should be restored as indicated previously. Finally control is transferred to word 01 of line 00 to return control to Intercom.

16.7 DATA OUTPUT

The result of any arithmetic operation should be placed in the A register. A zero exit from a routine should place a floating zero in the A register.

16.8 CHECK SUMS

Any subroutine to be attached to the appendix package or the basic package must have a number equal to its check sum stored in word u7.

17. ARITHMETIC SUBRCUTINES IN BASIC PACKAGE

The subroutines within the Intercom basic package may be used by an appendix routine. The specifications for the use of the floating point operations follow.

It is assumed that the extractors "-zzzzz00" and "-zzzzzzz" are stored in 20.00 and 20.01 upon entering the floating point operations.

17.1 ARITHMETIC

Storage: Normal loading of Intercom places the arithmetic routines in line 04. Certain constants used by the routine are stored in other lines of the basic package.

Entry:		
	Clear and Add	04.78
	Clear and Subtract	04.76
	Clear and Add Absolute Value	04.81
	Add	04.79
	Subtract	04.77
	Multiply	04.80
	Divide	04.84
	Inverse Divide	04.83

Data Input: For Clear and Add, Clear and Subtract, and Clear and Add Absolute Value, place operand in ID, sign in IP.

Other operations require numbers in both ID and the A register 21.00.01. In divide operations the denominator is placed in $ID_{0,1}$.

All the operations require a return command in 23.03 and the constant "-zzzzz00" is 20.02. The Return Command is obeyed at word time u7.

Data Output: Result in 21.00.01 (A register).

Exit: (Normal) Obey return command in 23.03 at word time u7.

(Error) The command in 04.26 transfers to the error indication sequence.

Short Line Used: Extractors 20.00-01-02
Temporary Storage 21.00-03 and 23.03.
Also 22.00-01 during divide operations.

17.2 FLOATING POINT DECIMAL-TO-BINARY CONVERSION SUBROUTINE

Entry: 02.70

Data Input: Floating Point Decimal Number in MQ as a 14 digit number, sign in IP, Exit command in 23.03.

Data Output: 22.00-01 = Floating Binary Number
PN, IP = Reconverted Floating Decimal Number
AR = $d \cdot 2^{-8}$ in AR, where d = binary form of decimal exponent.

Exits: (Normal) Command in 3.89 transfers to 23.03

(Zero) Command in 02.88 transfers to 23.03

(Error) Command in 03.78 transfers to 04.26

(Error Exit if input number $\geq 89.17014 \ldots\ldots = 2^{127} - 2^{78}$)

Short Lines Used: Extractors in 20.02-03 and 20.00-01.
Results of decimal-to-binary conversion stored in 22.00-01
Return Command in 23.03
Temporary storage in 21.02-03, 23.00-02

*Execution time: (Convert and Reconvert) 15 revolutions (Maximum)

*Execution time can be reduced 6 revolutions by replacing 02.86 with the constant (5858000). After exit, 02.86 must be restored to its original value of (485833v). With this change made, the reconverted number can no longer be found in PN and the exit (normal) will be the same as exit (zero).

17.3 FLOATING BINARY-TO-FLOATING DECIMAL CONVERSION SUBROUTINE

Entry: 01.41

Data Input: Floating Binary Number into ID, Sign in IP.
Return Command in AR.

Data Output: Floating Decimal Number in PN, Sign in IP.
$d \cdot 2^{-8}$ in AR where d = binary form of decimal exponent.

Exit: From 23.03 at word 91.

Short Lines Used: Extractors in 20.02-03 and 20.00-01.
Return Command 23.03.
Temporary Storage in 21.02-03, 23,00-02.

18. FRACTION SELECTOR

The fraction selector (formater) is a subroutine attached to the basic package and the appendix package. It provides an easy way of changing the format that controls the typewriter output sequence initiated by OP 33 or OP 38.

The typing of "D-" while operating the memory preparation routine initiates the read-in of the fraction selector into line 19. Control is transferred to 00 of line 19 and D is extracted into AR and shifted to the "next command" position. This is obeyed in AR effecting the transfer of control to a word in line 19 (word D). The command in word D copies the desired format into word 06, 07 and 08 of line 07. A number C is placed in word 09 of line 07. C is used to set up an extractor to pull out the desired digits for typeout eliminating the necessity for "wait" codes in the format.

After the format and C are placed in line 07, a "return to mark" is executed. Entry is made to Intercom after the tape is reversed four blocks and read forward one block restoring line four.

This same routine may be used as a subroutine and executed under control of Intercom commands. The operation is the same as described above except exit is made to Intercom, skipping the tape reading operations.

APPENDIX TO PART I

The attached is the list of changes between the contents of the tapes distributed to all users in August, 1958, and the contents of the coding sheets presented in this manual.

	Coding Sheets	August Tapes
8.01	.9w00000	.0000000
8.03	-.8w00001	.8000002
8.07	.000008w	.0000000
8.61	.0000010	-.8024019
8.62	.0000200	.0000000
8.64	.0000008	.0000000
8.90	.y64y51w	.y60wx1w
8.u1	.8066013	.6v6v3zz
8.u2	-.23232vz	.8067013
8.u7	.177927v	.177u27v
7.06	.0000188	.00001z1
7.08	.0000088	.00001x1
7.09	.0000024	.0000000
7.38	.282wyzz	.282uyzz
7.42	.0000000	.2w2w799
7.44	.3333uvz	.2z2zuvz
7.59	.084735z	.0000000
7.60	.0000C01	.0000080
7.61	-.80240:9	.084735z
7.64	.006v008	.0000000
7.65	.703v37z	.703x37z
6.00	.002x23z	.6v3z23z
6.18	.9819u9w	.951929w
6.23	.0002C00	.0006000
6.32	.0000000	.801301w
6.34	.xx666vz	.66666vz
6.39	.y1676vz	.67676vz
6.46	.z192000	.ww80000
6.53	.3839v15	.0000000
6.71	.0600040	-.02001y0
4.06	-.8831358	-.8813358
4.11	-.0y137v8	-.721373z
4.19	-.723173z	.153zzxx
4.21	-.82252y2	-.82212y2
4.33	-.v033z9w	.v033wxx
4.35	.872191w	.25250xw
4.49	.v43zzxx	.0000000
4.50	-.v436vzv	-.v43635v
4.51	.v53839v	.35382xz
4.54	.vw690z5	.4y4y2zz
4.56	-.67676vz	.802115z
4.57	.802115z	.3v2301z
4.84	.x85z000	-.x85u2v6
4.90	-.xw5z2x5	-.xw5z7v8
4.95	.y16433v	.y1642vv
4.96	.62363v5	-.y4693v5
4.u0	.646823z	.69692zz
4.u1	.y96u2vv	.y76u33v
4.u4	.681u23z	.686u23z
4.u6	.81363v5	.6u1u23z
4.u7	-.16172vu	-.94172xu

	Coding Sheets	August Tapes
3.05	.223031z	.222931z
3.38	-.uu3933u	-.u839338
3.41	.0w4u37z	-.uy30358
3.61	-.w05351y	.0w4u37z
3.78	-.50452vz	-.506u2vz
3.83	-.8429358	.6701708
2.75	-.wy6v754	-.wy53754
2.83	.6701708	-.880u31u
2.u7	-.9w1y31u	.0000000
1.00	.020711x	.0404702
1.07	.0u0u6xx	.0ul16xx
1.08	.0ul10z7	-.2929yvz
1.10	.0w0w3vz	-.xx302vz
1.12	.9214x1x	.8y6v796
1.13	-.45452vz	-.9013716
1.17	.9v1wz3w	.131451x
1.19	-.1618716	.u829z9w
1.20	.1617v99	.18183vz
1.21	.975y8xy	.971u8xy
1.23	.021v31z	.9v1wz3w
1.24	.9819z9w	.1ulvx1x
1.25	.uu2v4xx	-.45452vz
1.26	-.2929yvz	.1w1xvxw
1.27	.9x2433u	.2u2u2vz
1.29	-.2040334	.u0400xx
1.32	.22290xw	.22280xw
1.33	.23290xw	.23280xw
1.34	.24290xw	.24280xw
1.36	.2628yzz	.0000000
1.38	.28290xw	.28280xw
1.39	.u4500xw	.v25051w
1.40	.2u2u2vz	.uv2xv57
1.41	.uv2xv57	.2v2v4xx
1.47	.313029z	.5x5x2vz
1.54	.383uvzw	.3838vzw
1.56	.w0630zx	.3u3u336
1.60	.3z3z119	.3z3z039
1.61	.501x35z	.0000010
1.62	-.w2693xu	.0000200
1.64	-.w43y6x8	.w35y396
1.68	-.w65x335	-.y06233u
1.71	-.xx302vz	.0000000
1.73	.240835z	.241735z
1.75	.xy6111w	.wx462x9
1.77	.831u103	.8308103
1.78	.x26v796	.x05xv5w
1.79	.0047105	.000u105
1.82	.545v85y	.545v87y
1.89	.5v6111x	.5v5w11w
1.92	.5y38v5x	.xz61vxx
1.93	.y162yzw	.5z6311x
1.98	-.373x059	-.y4696x8
1.u2	.08134z8	.080x4z8
1.u3	.09134z8	.090x4z8
1.u6	.0000uvw	.0007uvw

	Coding Sheets	August Tapes
.00	.002x23z	.6v3z23z
.05	.8i3339w	.071513z
.11	.680129z	.0w143v3
.15	.806901z	.8v0v39w
.18	.146v13z	.yv6v333
.20	.930z000	-.yu66753
.21	.151427v	.950z39w
.41	.y17v0yu	.0000000
.45	.813z3u0	.0000000
.51	.34663v3	.0000000
.60	.3y3y0zw	.3y3y11w
.70	.486uvzw	.4848vzw
.72	.490123z	.4z4z6vz
.79	.yv05333	.0000000
.u2	-.yu0v753	.680129z
.u4	.9415000	.686839z
.u6	.wy4z6vz	.0000000
.u7	.1863702	.1819702

PART II

CODING SHEETS

LOADER I

Bendix Computer — Los Angeles 45, California

Page 1 of 4.

G-15D

PROGRAM PROBLEM: Loader I (Refer to page 16.8)

Prepared by D. Hassell

Date:

Line 04

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99
u0	u1	u2	u3
u4	u5	u6	

L	P	T or L_k	N	C	S	D	BP	NOTES
00		01	01	1	19	31		Stop DDA (See page 16)
61		04	17	6	21	31		N C 19.17
17	U	18	04	0	19	04		Loader ——▷ Line 04
04	U	09	09	0	19	23		
05								00000X0
06								0000034
07								800000X
08								0000110
09		02	10	4	23	02		Set standard line 19
10		00	11	4	23	02		Format
11		02	12	4	19	03		Set standard AR format
02								
03								
12	U	17	18	0	19	20		Set up dummy sw.
13								0200000
14								ZZZZZZZ
15		46	23	4	04	25		
16								-0
18		22	22	4	21	31		N.C. 04.22
22		23	24	0	20	28		(20.03) ——+——▷ AR
24		25	26	0	20	29		(20.01) ——+——▷ AR+
26		27	28	0	28	20		(AR) ——▷ 20.03
28		30	30	0	31	31		
AR		46+i	23	4	04	25		Pick up loader ∀ Σ ——▷ ID

FORM BCD 108-5.0

Bendix Computer

Los Angeles 45, California

Page 2 of 4.

G-15D

Prepared by D. Hassell

Date:

PROGRAM PROBLEM: Loader I — Intercom 1000 D.P.

Line 04

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99
U0	U1	U2	U3
U4	U5	U6	

L	P	T or L_k	N	C	S	D	BP	NOTES
23		24	27	4	25	22		ID ⟶ 22.00-01
27		28	29	0	22	28		22.00 $\xrightarrow{+}$ AR
29		30	30	3	20	29		(ZZZZZZZ) $\xrightarrow{-}$ AR+
30		32	33	0	28	27		(AR) ≠ 0?
								Yes
34		37	37	0	23	31		Clear registers
37	U	38	38.	0	25	79		Clear line 19
38		41	41	0	15	31		Read tape
41		45	35	0	22	28		(22.03) ⟶ AR
35		35	35	0	28	31		Wait
36		37	31	0	28	27		Sum check?
								No
31		32	42	0	28	28		
								Yes
32	U	33	40	3	19	29		Σ 19 $\xrightarrow{-}$ AR+
40		41	42	0	28	27		Σ OK?
								Yes
42		44	21	0	22	28		
21		20	20	0	29	31		
AR	U	23	22	0	19	XX		
								Enter when LΣ is incorrect
43		44	44	0	06	31		Reverse tape
44		44	44	0	28	31		Wait
45		46	37	0	17	31		Ring Bell

FORM BCD 108-5.0

Los Angeles 45, California

Page 3 of 4.

G-15 D

Prepared by D. Hassell

Date:

PROGRAM PROBLEM: Loader I Intercom 1000 D.P.

Line 04

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99
u0	u1	u2	u3
u4	u5	u6	

L	P	T or Lk	N	C	S	D	BP	NOTES
68								-X173445 Balancer
46								-Y3Y7Z03 Balancer
47								3121915 LΣ
48	U	23	22	0	19	01		Load line 01
49								-7Y7V575 Line 01 ΔΣ
50	U	23	22	0	19	02		Load line 02
51								-4WX8U55 Line 02 ΔΣ
52	U	23	22	0	19	03		Load line 03
53								XX13506 Line 03 ΔΣ
54	U	23	22	0	19	06		Load line 06
55								9W7WV36 Line 06 ΔΣ
56	U	23	22	0	19	07		Load line 07
57								Z5X1ZZ3 Line 07 ΔΣ
58	U	23	22	0	19	08		Load line 08
59								-0Z8V943 line 08 ΔΣ
60	U	23	22	0	19	00		Load line 00
61								-7Y08V58 Line 00 ΔΣ
62		64	56	4	04	20		Set up extractor
63								-1347192 Line 04 ΔΣ
64								-ZZZZZ00
65								-ZZZZZZZ
66	U	67	34	0	19	04		Load line 4 return control to 00.01
33		36	20	0	20	28		
20		22	22	0	31	31		Obey AR

FORM BCD 108--5.0

Bendix Computer

MEMORY ALLOCATION CHART

Problem { Intercom 1000 Double Precision Page 4 of 4
Loader I Date

Prepared by D. Hassell

Check Sum .3121959 Line 04

U7	U6	U5	U4
U3	U2	U1	U0
99	98	97	96
95	94	93	92
91	90	89	88
87	86	85	84
83	82	81	80
79	78	77	76
75	74	73	72
71	70	69	68 -x173445
67	66 4322264	65 -zzzzzzz	64 -zzzzz00
63 -1347192	62 -w042094	61 -7y08v58	60 1716260
59 - z8v9y3	58 1716268	57 z5xlzz3	56 1716267
55 9w7wv36	54 1716266	53 xx13506	52 1716263
51 -4wx8v55	50 1716262	49 -7v7u575	48 1716261
47 3121959	46 -y3y7z03	45 2y2523z	44 2w2w39z
43 2w2w0xz	42 uw152xw	41 ux232xw	40 2u2u39v
39	38 29291zz	37 2626333	36 261z39v
35 232339z	34 25252zz	33 u41429w	32 2128y7x
31 212u39w	30 u02139v	29 1z1yy9x	28 1y1y3zz
27 1x1x2xw	26 1w1w394	25 9x142xw	24 1u1u29x
23 -1u1v336	22 181829w	21 14143vz	20 16163zz
19	18 -16162vz	17 1204264	16
15 -uy17099	14 zzzzzzz	13 200000	12 1112274
11 -820w263	10 -800v2y2	09 -820u2y2	08 110
07 800000x	06 34	05 x0	04 909277
03 -8w00000	02 1000000	01 - 411uvz	00 10167z

FOUR-WORD LINES

LINE	3	2	1	0

FORM BCD 108-15.1

INCREMENT ADDRESS

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: INCREMENT ADDRESS

Prepared by D. Hassell

Date: 10/30/58

Line 01

L	P	T or L_K	N	C	S	D	BP	NOTES
								Enter all commands with $K \neq 0$
49	.	50.	51	1	25	30		Recombine K → PN_0
51	u.	54.	54	2	26	24		PN_0 → MQ_1
54	.	55.	58	2	31	28		|OP| → AR
58	.	59.	60	3	06	29		46 (70 in dec.) $\cdot 2^{-16}$ → AR+
60	u.	63.	63	0	08	25		$1 \cdot 2^{-19}$ → ID_0 $1 \cdot 2^{-24}$ → ID_1
63	.	08.	72	0	24	31		$PN_0 = K \cdot 2^{-23}$ $PN_1 = K \cdot 2^{-28}$
72	.	74.	74	0	22	31		OP ≥ 70 ?
74	.	75.	76	0	08	29		dummy filler → AR +
76	.	78.	78	0	31	31		Interpret OP ≥ 70
AR	u	80	11 ± OP	2	30	28		|Address of IC| → AR See next page
06	u.	00.	00	2	21	28		IC → AR
00	.	01.	07	0	08	29		Excess 28 → AR
07	u.	10.	10	1	22	29		B_k (WD & CH) → AR
10	.	12.	12	0	29	31		WD > 99?
								If yes, see error loop
12	.	18.	20	3	08	29		Excess 28 → AR NO
20	.	21.	23	2	28	25		Incremented value → ID_1
23	.	02.	27	0	24	31		Shift 1 bit to divide by 2
27	.	29.	36	0	25	26		ID → PN
36	.	38.	40	3	23	31		Incremented source CH → ID_1
40	.	42.	42	0	21	31		NC 00.42 page 00/1
								Enter if WD > 99
13	.	69.	69	4	21	31		Exit to error loop NC 04.69 page 04/1

Necessary to make "store" command function properly.

FORM BCD 108-17.0

Los Angeles 45, California

G-15 D

Prepared by D. Hassell

Page 2 of 7

Date: 10/30/58

PROGRAM PROBLEM: INDEX REGISTER OPERATIONS

Line 01

L	P	T or L_K	N	C	S	D	OP	NOTES
75	.94.	97	0	08	28			Dummy register pickup command
97	.98.	99	2	26	29			$K \cdot 2^{-23} \xrightarrow{+} AR_+$
99	.u1.	u1	0	31	31			
AR	u4	u6	4	08 + K	22			$B_k \longrightarrow 22.0,1$ see page 01/1
81	.83.	91	2	06	30			Dummy B_{wd} set command $\xrightarrow{+}$ PN OP = 70
91	u.94.	94	1	28	24			Register setting $\xrightarrow{+} MQ_{1,0}$
94	.96.	96	0	21	31			NC·0·96 page 00/6
82	.83.	91	2	02	30			Dummy D_{wd} set command $\xrightarrow{+}$ PN OP = 71
								Enter "Set L_{wd} register"
83	.85.	91	2	06	30			Dummy L_{wd} set command $\xrightarrow{+}$ PN OP = 72
								Enter "Set B_{ch} register"
84	.85.	91	2	08	30			Dummy B_{ch} set command $\xrightarrow{+}$ PN OP = 73
								Enter "Set D_{ch} register"
85	.87.	91	2	06	30			Dummy C_{ch} set command $\xrightarrow{+}$ PN_1 OP = 74
								Enter "Set L_{ch} register"
86	.87.	91	2	08	30			Dummy L_{ch} set command $\xrightarrow{+}$ PN_1 OP = 75
								Enter "Increment B_{wd}"
87	.88.	97	2	08	28			Dummy k pickup command $\xrightarrow{+}$ AR OP = 76
								Enter "Increment B_{ch}"
88	.89.	97	2	08	28			Dummy k pickup command $\xrightarrow{+}$ AR OP = 77
								Enter "CA R_k" "IRA" (See page 29.13.5.1)
89	.90.	97	0	08	29			Dummy pick up command AR OP = 78
								Enter Store IRA R_k (See page 29.13.5.2)
90	.91.	65	2	30	30			Construct Store OP = 79
65	.66.	82	0	22	28			Command

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D

Prepared by D. Hassell

Page 3 of 7

Date: 10/30/58

PROGRAM PROBLEM: POSITION TYPEWRITER PAPER; TYPE TABULATING NUMBER

Line 01

L	P	T or Lk	N	C	S	D	BP	NOTES
								Increment B_{wd} (cont'd)
01		.03.	05	2	06	30		Build B_{wd} Store Command
05		.08.	09	0	23	28		B_{wd} ⟶ AR
09		.10.	11	1	23	29		D $\xrightarrow{+}$ AR +
11	u.	14.	14	1	28	24		New B $\xrightarrow{+}$ $MQ_{0,1}$
14		.15.	16	1	23	28		L $\xrightarrow{+}$ AR
16		.17.	18	3	24	29		B_{wd} $\xrightarrow{-}$ AR +
18		.21.	11	0	22	31		B ≤ L?
								Enter to store new B and transfer
21		.23.	94	2	06	30		Set to GO to 00.03 Yes
								Enter to store new B and NO transfer
22		.96.	96	0	21	31		NC 00.96 page 00/6 No
								Enter "Position Typewriter Paper"
30		.33.	42	0	23	31		Clear Registers OP = 30
42		.43.	48	0	30	28		Address = No. of tabs and C.R. ⟶ AR
48		.50.	53	0	28	26		Number of tabs and C.R. ⟶ PN_0
53	u.	72.	u4	4	26	30		Shift C.R.'s ⟶ PN_1
u4	w.	u6.	71	0	21	31		NC 00.71 page 00/4
								Enter "Type Tabulating Number"
31		.35.	79	0	30	28		Output ⟶ AR OP = 31
79	u.	00.	71	0	08	05		Type out routine ⟶ Line 05
71	w.	93.	48	4	21	31		Mark 93 NC 04.48 page 04/5
								Exit to Binary-to-Decimal Conversion

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D

Prepared by D. Hassell

Page 4 of 7

Date: 10/30/58

PROGRAM PROBLEM: TYPE & TAB; TYPE & C. R.; & OTHER OUTPUT OPERATIONS

Line 01

L	P	Tag	N	C	S	D	BP	NOTES
								Enter "Type and Tab" floating point number
32	.33.41		0	06	28			RC ⟶ AR OP = 32
41	.43.45		2	26	23			RC ⟶ 23.03; Zero ⟶ AR
45	.48.52		5	20	24			Conversion Extractors ⟶ MQ
52	.54.57		5	24	20			Extractors ⟶ 20.02-3
57	.86.86		2	21	31			Exit to Output Conversion NC 02.86 page 02/1
								Enter "Type and Carriage Return" floating point number
34	.35.41		0	06	28			RC ⟶ AR OP = 34
								Enter "Type and Tab" fixed point number
33	.34.41		0	06	28			RC ⟶ AR OP = 33
								Enter "Type and Carriage Return" fixed point number
38	.39.41		0	06	28			RC ⟶ AR OP = 38
								Enter "Punch Paper Tape"
39	86.86		0	06	28			Leader punch tally ⟶+ AR OP = 39
80	u.00.04		0	08	05			Punch routine ⟶ 05
04	.30.30		4	21	31			NC 04.30 page 04/6
								Increment B_{ch} (cont'd)
02	.05.06		0	23	28			B_{ch} ⟶ AR
06	.07.09		2	06	30			Build B_{ch} Store Command
								Enter all 29 ≤ OP ≥ 40
67	.69.70		0	08	28			Dummy data pickup command ⟶ AR
70	.71.76		0	21	29			Address ⟶+ AR

FORM BCD 3-17-0

Bendix Computer — Los Angeles 45, California

G-15D

Prepared by D. Hassell

Page 5 of 7

Date: 10/30/58

PROGRAM PROBLEM: FLOATING DECIMAL-TO-FIXED DECIMAL CONVERSION

Line 01

L	P	T or L_K	N	C	S	D	BP	NOTES
								Enter floating decimal-to-fixed decimal conversion
u2	u.	08.	19	1	07	24		Format ⟶ MQ "TT entry"
19		20.	24	5	24	22		Store Format
24		24.	25	3	28	28		2^{-8} $\xrightarrow{-}$ AR
25		42.	43	1	06	29		$39 \cdot 2^{-8}$ (=57 dec.) $\xrightarrow{+}$ AR
43		46.	46	0	22	31		Output exceed limit of format?
								If "Yes", see 01.47 below
46		71.	73	1	28	25		57 (Exp.) = ΔE_1 ⟶ ID_1 [No]
73		86.	08	0	26	31		$\Delta E \cdot 2^{-26} = 4 \Delta E \cdot 2^{-28}$ ⟶ ID_1
08		09.	17	0	07	23		Extractor shift control (ESC) ⟶ 23.01
17		27.	28	3	25	28		$4 \Delta E \cdot 2^{-28}$ $\xrightarrow{-}$ AR
28		34.	37	5	26	24		Decimal output ⟶ MQ
37		16.	55	0	24	31		Shift off decimal exponent
55		64.	66	5	24	25		Mantissa of output $\xrightarrow{+}$ ID
66		67.	68	0	28	27		Exponent = 57?
								If no mantissa is shifted, see 01.69 below
68		70.	93	4	25	21		Output ⟶ 21:2·3 [Yes]
93		97.	98	3	23	28		ESC $\xrightarrow{-}$ AR
98	u.	55.	61	4	02	25		Extractor ⟶ ID — see 01.08 above; form digit extractor
61		80.	29	0	26	31		Shift controlled by AR
29		30.	64	4	25	20		Extractors ⟶ 20·2·3
64		68.	62	5	22	24		Format ⟶ MQ
62		66.	u5	4	30	26		Extract output ⟶ PN
u5		17.	17	0	20	31		Exit to Typeout routine
69		v6.	68	0	26	31		Shift to align decimal point

FORM BCD 08-17.0

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: PERFORM SUBROUTINE

Prepared by D. Hassell

Page 6 of 7

Date: 10/30/58

Line 01

L	P	T or Lk	N	C	S	D	BP	NOTES
.47		.49.	.48	0	20	31		Exit to type output in floating point
.u3		u.09.	.9	1	07	24		"TR" format → MQ
								"Perform Subroutine" (cont'd)
.95		.95.	.95	0	28	31		Wait for ready
.96		.98.	.u0	0	08	28		Build command to load subroutine
.u0		u.u1.	.u1	0	06	19		(Line 06) → Line 19
.u1		.u7.	.u7	6	21	31		NC 06.u7 . page 06/4
								Enter "Type Command from Memory"
.35		u.44.	.44	0	08	23		4-digit output routine → 23 OP = 35
.44		.48.	.50	0	06	23		
.50		.58.	.59	0	06	28		Build memory pickup command. NC AR
.59		.63.	.76	0	30	29		
AK		XX	77	0	XX	28		Memory → AR
.77		.03.	.26	0	08	03		AR Format → 03.03
.26		.41.	.41	7	21	31		NC 08.41 page 08/1
								0A (IR_k) (cont'd)
.78		.82.	.u7	1	28	22		Store IR_k IRA
.u7		.21.	.21	0	21	31		NC 00.01 page 00/1
								Enter "Block Transfer" routine
.92		.93.	.56	2	26	29		$K \cdot 2^{-28} \xrightarrow{+}$ AR OP = 81 (See page 29.13.7)
.56		.64.	.99	0	07	29		dummy transfer command → AR
.03		.w400000						
.15		.7z00000						

FORM BCD 108-17.0

MEMORY ALLOCATION CHART

Bendix Computer

Problem: Intercom 1000 double precision — Page 7 of 7

Date

Prepared by D. Hassell

Check Sum -.7v7u575 — Line 01

U7 15152vz	U6 uvw	U5 111129z	U4 yu472vz
U3 9134z8	U2 8134z8	U1 -6v6vuvz	U0 65650x3
99 65653zz	98 -373x059	97 6363v5x	96 y26411w
95 5z5z39z	94 60602vz	93 y162yzw	92 5y38v5x
91 5y5y798	90 5w41vxy	89 5v6111x	88 5u6191w
87 596191w	86 585v91y	85 x75v8xy	84 565v91y
83 x55v8xy	82 545v85y	81 x35v8xy	80 4105
79 47105	78 x26v796	77 831u103	76 4y4y3zz
75 xy6111w	74 4w4w11x	73 240835z	72 4u4u2xz
71 -xx302vz	70 484w2vx	69 744435z	68 -w65x335
67 w54611w	66 444439v	65 43522xw	64 -w43y6x8
63 84831z	62 -w2693xu	61 501x35z	60 3z3z119
59 vz4w3xx	58 3w3wwxx	57 5656uvz	56 w0630zx
55 -w042719	54 383uvzw	53 -436835y	52 -v639714
51 3636v58	50 vu3v0xw	49 333373y	48 v23539u
47 313029z	46 w749799	45 -v034698	44 v0320x7
43 2y2y2xz	42 2w303xw	41 uv2xv57	40 2u2u2vz
39 u4500xw	38 28290xw	37 103731z	36 2628yzz
35 2w2w117	34 24290xw	33 23290xw	32 22290xw
31 u34z3xw	30 212u2zz	29 -2040334	28 -u225758
27 9x2433u	26 -2929yvz	25 uu2v4xx	24 9819z9w
23 21v31z	22 60602vz	21 975y8xy	20 1617v99
19 -1618716	18 15152xz	17 9v1wz3w	16 1212z1x
15 7z00000	14 10106zw	13 -45452vz	12 9214x1x
11 y0y798	10 w0w3vz	09 v0v6zx	08 u110z7
07 u0u6xx	06 8098xy	05 88092zw	04 -1y1y2vz
03 w400000	02 85062zw	01 83058xy	00 20711x

FOUR-WORD LINES

LINE	3	2	1	0

FORM BCD 108-15.1

DECIMAL-TO-BINARY BINARY-TO-DECIMAL

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: DECIMAL-TO-BINARY BINARY-TO-DECIMAL

Prepared by S. H. Lewis

Page 1 of 6

Date: 10/30/58

Line 02

L	P	T or L_K	N	C	S	D	BP	NOTES
								Enter Floating Decimal-to-Binary conversion
70	.72.	75	5	20	26			Set up Extractors
75	.78.	83	5	26	20			
83	.08.	10	4	24	26			Output ⟶ $PN_{0,1}$
10	.12.	18	4	07	24			$5 \cdot 2^{-3} + 5^2 \cdot 2^{-9} + 5^4 \cdot 2^{-21} + 5^4 \cdot 2^{-33}$ = [-.uw93893, 1000000] ⟶ $MQ_{0,1}$
18	.21.	27	3	23	31			Extract [z0z0z0z, lylylyO]
27	.06.	38	0	24	31			Multiply
38	.40.	40	3	23	31			Extract [zz00000] χ_8 (hex) ID_1
40	.42.	46	4	26	27			Input = 0?
46	.49.	59	0	23	31			Clear Registers (Yes)
59	.60.	82	0	07	25			Floating Zero
82	.84.	86	4	25	22			Enter here from 03.79
86	.87.	88	0	25	27			Begin Binary-to-Decimal conversion
								Here if output zero
88	.90.	91	7	21	31			Exit to RC in 23.03 zero exit
								Begin Binary-to-Decimal conversion
89	.90.	92	4	25	21			$ID_{0,1}$ ⟶ 21.02-3 (χ)
92	.94.	99	0	30	28			(30.02) = b ⟶ AR
99	u.12.	14	0	28	29			2^{20}(AR) ⟶ AR
14	.15.	17	0	28	24			$b \cdot 2^{-8}$ = (AR) ⟶ MQ_1
17	.18.	25	6	07	25			$2^{-2} \log_2$ = [.1344135] CVA ⟶ ID_1
25	.16.	42	0	24	31			Multiply
42	.43.	48	0	07	30			$2^{-7}(5^{-16}\log 2)$ = [.005xz65] ⟶ PN +
48	.50.	50	3	23	31			Extract [.1y00000]

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D

Prepared by S. H. Lewis

Page 2 of 6

Date: 10/30/58

PROGRAM PROBLEM: DECIMAL-TO-BINARY BINARY-TO-DECIMAL

Line 02

L	P	T or LK	N	C	S	D	BP	NOTES
.50	.	51.	52	0	25	28		ID ⟶ AR
.52	u.	55.	55	0	28	29		2^2(AR) ⟶ AR
.55	.	56.	58	0	07	29		[.1200000] ⟶ AR
.58	.	60.	62	0	28	23		b'' = (AR) ⟶ 23.00
.62	.	63.	66	0	25	28		ID_1 ⟶ AR
.66	.	67.	68	0	07	29		Command ⟶ AR
.68	.	70.	70	0	31	31		
07.67 AR		72	97	4	07	26		$10^{P'}$ ⟶ $PN_{0,1}$
.97	.	99.	u1	3	23	31		Extract [.0C001zz]
.u1	.	u2.	u3	0	30	28		b'' = (30.02) ⟶ AR
.u3	.	u4.	u5	3	25	29		(ID_0) ⟶ AR
.u5	u.	00.	04	4	26	24		$10^{P'}$ = ($PN_{0,1}$) ⟶ $MQ_{0,1}$
.04	.	06.	08	4	21	25		B = (31.02-03) ⟶ $ID_{0,1}$
.08	.	10.	13	1	28	23		$b' \cdot 2^{-28}$ = (AR) ⟶ 23.02
.13	.	15.	18	3	21	31		NC 03.18 page 03/2
								Dec.-Binary (Cont'd)
.47	.	49.	51	1	25	28		$d \cdot 2^{-8}$ = (ID) $\xrightarrow{+}$ AR
.51	w.	56.	71	7	23	31		Extract [-.00zz00z, y01zy00]
.71	.	12.	87	0	24	31		Multiply finishes conversion
.87	w.	90.	93	7	23	31		Extract [.007zzz0, 0000000] to a binary integer
.93	.	24.	28	0	24	31		Multiply
.28	.	31.	31	3	23	31		Extract [-.000zzzz, zzy0000]
.31	.	24.	60	0	24	31		Multiply
.60	.	62.	67	5	07	25		$10^{12.2-56}$ = [9442:000, 0000y8X]
.67	.	68.	69	3	07	29		[Ou00000] ⟶ AR

FORM BCD 108-17.0

Los Angeles 45, California

Page 3 of 6

G-15D

Prepared by _____

Date: 10/30/58

PROGRAM PROBLEM: DECIMAL-TO-BINARY BINARY-TO-DECIMAL

Line 02

L	P	T or LR	N	C	S	D	CP	NOTES
69	.	4.76	1	25	31			Divide Convert to a binary fraction
76	.	78.78	0	22	31			AR Test
78	.	79.80	1	28	26			AR $\xrightarrow{+}$ PN_1
80	.	82.85	3	23	31			Extract [2800000]
85	.	86.96	4	24	21			MQ $\longrightarrow$ 21.02-03
96	.	97.09	1	26	23			PN $\longrightarrow$ 23.01
09	.	04.26	0	24	31			Shift 2 bits Multiplier table pick up T
26	.	28.32	3	07	29			.5000000 $\xrightarrow{-}$ AR
32	.	34.35	0	22	31			(AR) < 0?
35	.	37.78	3	21	31			N.C. from error loop goes to 04.26
36	.	37.45	0	07	28			Command $\longrightarrow$ AR
45	.	47.68	3	25	29			$I D_1$ $\xrightarrow{-}$ AR
AR	904X	95	4	07	26			10^{P1} $\longrightarrow$ $PN_{0,1}$
79	.	80.46	0	28	28			
95	w.	98.u0	3	23	31			Extract (.00001zz) Stored X_E ID_0
u0	.	u2.u6	3	25	28			ID_0 $\xrightarrow{-}$ AR ID_0 AR
u6	.	04.05	0	07	29			.0000101
05	.	07.08	3	21	31			N. C. 03.08
72	.	74.77	4	31	25			(31.02-03) $\longrightarrow$ $ID_{0,1}$ Here from 03.69
77	.	78.u2	1	23	28			b' = (23.02) $\xrightarrow{+}$ AR
u2	.	u4.56	3	21	31			N.C. 03.56

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: CONSTANTS

Prepared by D. Hassell

Date: 10/30/58

Line 02

L	P	T or LK	N	C	S	D	BP	NOTES
.53								-.00000zz — See 03.52 Page 03/4
.54								.zzzzzzz
.56								.y01zy00 — See 02.51 Page 02/2
.57								-.00zz00z
.61								.0700000 — See 03.60 Page 03/2
.63								-.000000z — See 03.62 Page 03/4
.64								.zzzzzzz
.65								.0z00000 — See 03.64 Page 03/5
.73								-.0000000 — See 03.72 Page 03/4
.74								.zzzzzzz
.81								.7800000 — See 02.80 Page 03/3
.84								.1zzzzzz — See 03.82 Page 03/4
.91								.007zzz0 — See 02.87 Page 02/2
.94								.01zzzzz — See 03.92 Page 03/5
.98								.00001zz — See 02.97 Page 02/2
.u4								.001zzzz — Sec 03.u2 Page 03/5
.41								.0000000
.90								.0000000 — Significant Zero's
.u								
183	u	u3	01	1	24	08		See 01.82

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D

Prepared by D. Hassell

Page 5 · 6

Date: 10/30/58

PROGRAM PROBLEM: CONSTANTS

Line 02

L	P	T or LR	N	C	S	D	BP	NOTES
.00								.0000110
.01								.00000x0 Line 19 Format (Variable)
.02								.1000000
.03								.8w00001
.06								.0001zzz See 03.04 Page 03/5
.07								.7z003y0
.11								-.0zzzzzz See 03.10 Page 03/4
.12								.zzzzzzz
.15								-.zzzzzzz See 03.14 Page 03/5
.16								.zzzzy00
.19								.z0z0z0z See 02.18 Page 02/1
.20								.1y1y1y0
.21								-.00zzzzz See 03.20 Page 03/4
.22								.zzzzzzz
.23								.0z0z0z0 See 00.22 Page 00/1
.24								.10z0z0z See 00.22 Page 00/1
.29								-.000zzzz See 02.28 Page 02/6
.30								.zzy0000
.33								-.000zzzz See 03.32 Page 03/4
.34								.zzzzzzz
.37								.00001z0 See 00.36 Page 00/1
.39								.zz00000 See 07.38 Page 07/2
40								- 0 -
.43								-.0000zzz See 03.42 Page 03/4
.44								.zzzzzzz
.49								.1y00000 See 02.48 Page 02/1

FORM BCD 108-17.0

Bendix Computer

MEMORY ALLOCATION CHART

Problem { Intercom 1000 double precision Page 6 of 6
Date

Prepared by S. H. Lewis Check Sum -.4wx8v55 Line 02

U7	U6	U5	U4
-9w1y31u	84050zx	- 4358	1zzzz
U3	U2	U1	U0
6969z3x	6838yvz	67673xw	y66uz3w
99	98	97	96
w0y39x	1zz	6365yzz	6209757
95	94	93	92
y264yzz	1zzzzz	181w31z	xy633xw
91	90	89	88
7zzz0		-5w5w335	-5u5vyvz
87	86	85	84
-xu5xyzz	585833v	-5860315	1zzzzzz
83	82	81	80
6701708	-x456336	7800000	5255yzz
79	78	77	76
512y39w	505079u	4z666zw	4y4y2xz
75	74	73	72
-wy6v754	zzzzzzz	-0000000	-wu4x3z9
71	70	69	68
w5731z	-w84v69u	724w73z	46463zz
67	66	65	64
4545wzx	44440zx	z00000	zzzzzzz
63	62	61	60
- z	404233w	700000	-vy434z9
59	58	57	56
3x520z9	vw3y397	- zz00z	y01zy00
55	54	53	52
393u0zx	zzzzzzz	- zz	373739x
51	50	49	48
-v847yzz	343433w	1y00000	3232yzz
47	46	45	44
v13373w	313v2zz	uz44z3x	zzzzzzz
43	42	41	40
- zzz	2w300zy		-uu2y35v
39	38	37	36
zz00000	2828yzz	1z0	262x0zw
35	34	33	32
254yyvz	zzzzzzz	- zzzz	22232xz
31	30	29	28
183w31z	zzy0000	- zzzz	1z1zyzz
27	26	25	24
62631z	9w20wzx	102u31z	10z0z0z
23	22	21	20
z0z0z0	zzzzzzz	- zzzzz	1y1y1y0
19	18	17	16
z0z0z0z	151vyzz	-14198z9	zzzzy00
15	14	13	12
-zzzzzzz	1011398	z12yvz	zzzzzzz
11	10	09	08
- zzzzzz	-8w120z8	41u31z	8u0x797
07	06	05	04
7z003y0	1zzz	708yvz	-86082v9
03	02	01	00
8030000	1z1	8030000	1z1

FOUR-WORD LINES

LINE	3	2	1	0

FORM BCD 108-15.1

DECIMAL-BINARY BINARY-DECIMAL CONVERSION

Bendix Computer — Los Angeles 45, California

G-15D — Prepared by S. H. Lewis — Page 1 of 7 — Date: 10/30/58

PROGRAM PROBLEM: Decimal-Binary Binary-Decimal Conversion — Line 03

L	P	T or L_K	N	C	S	D	BP	NOTES
.08	u.	11.	11	.4	26.	24		Dec. - Binary (Cont'd)
.11		.14.	21	.4	21.	25		(21.02-03) ⟶ $ID_{0,1}$
.21		.22.	26	.1	28.	21		(AR) ⟶ 21.02
.26		.27.	30	.0	24.	27		(MQ_1) ⟶ Test
								If ≠ see 03.31 page 03/2
.30		.33.	38	.0	23.	27		23.01 ⟶ Test
								If ≠ see 03.39 page 03/2
.38		.42.	57	.4	25.	26		$ID_{0,1}$ ⟶ $PN_{0,1}$
.57		.58.	61	.3	21.	28		21.02 $\xrightarrow{-}$ AR
.61		.64.	83	.5	08.	30		Round off $\xrightarrow{+}$ $PN_{0,1}$ | here also from 03.54
.82		.04.	41	.4	26.	24		PN ⟶ $MQ_{0,1}$
.41		.12.	74	.0	27.	31		Normalize
.74		.76.	98	.0	22.	31		AR < 0?
								If yes, see 03.99 below
.98		.u0.	46	.2	21.	31		[+] Zero exit page 02/1
.99		.u0.	u1	.1	28.	28		[-] AR $\xrightarrow{+}$ AR
.u1		.u3.	19	.4	24.	25		MQ_1 ⟶ ID_1.
.19		.22.	27	.0	24.	21		MQ_0 ⟶ 21.02
.27		.32.	33	.2	20.	29		|(20.00)| $\xrightarrow{+}$ AR +
.33		.34.	37	.0	27.	25		(27.02) ⟶ ID_0
.37		.39.	78	.0	22.	31		AR < 0?
								If yes, see 03.79 page 03/2
.78		.80.	69	.4	21.	31		Error exit see 04.26 page 04/1

FORM BCD 108-37.0

Bendix Computer — Los Angeles 45, California

G-15D — Prepared by H. Lewis — Page 2 of 7

PROGRAM PROBLEM: Decimal-Binary Binary-Decimal Conversion — Date: 10/30/58 — Line 03

L	P	T or LK	N	C	S	D	BP	NOTES
79	.81.	82		2	21	31		Normal exit → 02.82
39	.40.	87		0	07	28		/≠ 0/ Command → AR
87	.89.	96		0	23	29		23.01 → AR
96	.98.	98		0	31	31		Obey AR
AR		99	u7	0	07	28		10^{P2} → AR
u7	.21.	25		0	28	24		AR → MQ_1
05	.34.	48		0	24	31		Multiply
48	.50.	51		2	27	28		|(27.02)| → AR
51	.52.	54		3	28	28		(AR) → AR
54	.58.	61		3	21	29		(21.02) → AR see page 03/1
31	.96.	24		0	24	31		Multiply
24	.26.	30		4	26	25		$(PN_{0,1})$ → $ID_{0,1}$
								Bin-Dec Conversion (here from 02.13)
18	.19.	22		0	24	27		MQ_1 → Test
								If ≠ see 03.23 page 03/3
22	.26.	29		6	23	25		b' = (23.02) CVA→ ID_1
29	.12.	43		0	24	31		Shift
43	.44.	46		6	25	24		$\|b'\| \cdot 2^{-5} = (ID_0)$ → MQ_1
46	.47.	49		0	07	25		$2^{-3} \log_{10} 2 = [.09u209u]$ → ID_1
49	.10.	60		0	24	31		Multiply
60	.62.	63		3	23	31		Extract [.0700000]
63	.65.	69		0	25	27		ID_1 → Test
								See following page

FORM BCD 88-17.0

Los Angeles 45, California

G-15 D

Prepared by H. Lewis

Page 3 of 7

Date: 10/30/58

PROGRAM PROBLEM: Binary-Decimal Decimal-Binary Conversion

Line 03

L	P	T or LR	N	C	S	D	BP	NOTES
69	.71.	72	2	21	31			N.C. 02.72
70	.72.	73	0	23	28			≠0 d' = (23.00) → AR
73	.75.	77	3	25	29			(ID_1) $\xrightarrow{-}$ AR
77	.80.	88	0	28	23			d' = (AR) → 23.00
88	.89.	93	0	25	28			ID_1 $\xrightarrow{+}$ AR
93	.94.	96	0	07	29			Command $\xrightarrow{+}$ AR
(07.94) AR		99	00	0	07	28		(10^{P2}) → AR
23	.96.	16	0	24	31			Multiply
16	.18.	22	4	26	21			$(PN_{0,1})$ → 21.02-03
00	.02.	06	4	31	25			(31.02-03) → $ID_{0,1}$
06	.07.	09	0	28	24			(AR) → MQ_1
09	.10.	12	0	23	21			b' = (23.02) $\xrightarrow{+}$ 21.02
12	.14.	15	2	27	28			|(27.02)| → AR
15	.34.	50	0	24	31			Multiply
50	u.53.	53	4	26	25			$PN_{0,1}$ → $ID_{0,1}$
53	.54.	56	1	21	29			b' = (21.02) $\xrightarrow{+}$ AR
56	.57.	58	0	28	27			AR = 0? Here from 02.u2
								If ≠ 0, see 03.59 page 03/6
58	.59.	67	0	28	28			Skip
67	.68.	71	4	25	26			$ID_{0,1}$ → $PN_{0,1}$
71	u.76.	76	6	26	30			$\|(PN_{0,1})\|$ $\xrightarrow{+}$ $PN_{0,1}$ (4 wts)
76	.78.	81	6	25	30			$\|(ID_{0,1})\|$ $\xrightarrow{+}$ $PN_{0,1}$
81	.82.	86	6	26	30			$\|(PN_{0,1})\|$ $\xrightarrow{+}$ $PN_{0,1}$

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D — Prepared by H. Lewis — Page 4 of 7 — Date: 10/30/58

PROGRAM PROBLEM: Decimal-Binary Binary-Decimal Conversion — Line 03

L	P	T or LK	N	C	S	D	BP	NOTES
.86	.	90.	90	0	29	31		Overflow ⟶ Test
								If set see 03.91 page 03/5
.90	.	92.	94	3	07	28		[.0100000] $\xrightarrow{-}$ AR
.94	.	96.	97	0	23	29		$d \cdot 2^{-8} = (23.00)$ ⟶ AR +
.97	.	98.	u0	4	07	30		$[5 \cdot 10^{-13}]$ $\xrightarrow{+}$ $PN_{0,1}$
.u0	.	u3.	u3	0	29	31		Overflow?
								If set , see 03.u4 page 03/5
.u3	.	u4.	u6	4	26	25		$(PN_{0,1})$ ⟶ $ID_{0,1}$
.u6	u.	01.	01	4	07	24		Telescope Multiplier ⟶ $MQ_{0,1}$
.01	.	06.	10	0	24	31		Multiply
.10	.	13.	13	3	23	31		Extract [-0zzzzzz, zzzzzzz]
.13	.	06.	20	0	24	31		Multiply
.20	.	23.	25	3	23	31		Extract [-00zzzzz, zzzzzzz]
.25	.	06.	32	0	24	31		Multiply
.32	.	35.	35	3	23	31		Extract [-000zzzz, zzzzzzz]
.35	.	06.	42	0	24	31		Multiply
.42	.	45.	45	3	23	31		Extract [-0000zzz, zzzzzzz]
.45	.	06.	52	0	24	31		Multiply
.52	.	55.	55	3	23	31		Extract [-00000zz, zzzzzzz]
.55	.	06.	62	0	24	31		Multiply
.62	.	65.	65	3	23	31		Extract [-000000z, zzzzzzz]
.65	.	06.	72	0	24	31		Multiply
.72	.	75.	75	3	23	31		Extract [-0000000, zzzzzzz]
.75	.	06.	82	0	24	31		Multiply
.82	w.	84.	85	3	23	31		Extract [1zzzzzz] 0

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D — Prepared by H. Lewis — Page 5 of 7 — Date: 10/30/58

PROGRAM PROBLEM: Decimal-Binary Binary-Decimal Conversion — Line 03

L	P	T or Lk	N	C	S	D	BP	NOTES
.85	.	06.	92	.0	24	31		Multiply
.92	w.	94.	95	.3	23	31		Extract [.01zzzzz] 0
.95	.	06.	u2	.0	24	31		Multiply
.u2	w.	u4.	u5	.3	23	31		Extract [.001zzzz] 0
.u5	.	06.	04	.0	24	31		Multiply
.04	w.	06.	07	.3	23	31		Extract [.0001zzz] 0
.07	.	06.	14	.0	24	31		Multiply
.14	.	17.	17	.3	23	31		Extract [-.zzzzzzz, zzzzy00]
.17	.	16.	34	.0	24	31		Shift (no increment)
.34	.	35.	36	.0	28	24		$d \cdot 2^{-8} = (AR) \longrightarrow MQ_1$
.36	.	38.	40	.4	25	21		
.40	.	41.	44	.0	07	25		$10^{-1} =$ [.1999u00] $\longrightarrow ID_1$
.44	w.	46.	47	.0	23	31		Clear even side of 2-word register
.47	.	16.	64	.0	24	31		Multiply
.64	.	66.	66	.3	23	31		Extract [.0z00000]
.66	.	67.	68	.0	28	26		$d \cdot 2^{-8} = (AR) \longrightarrow PN_1$
.68	u.	80.	80	.4	25	30		$\lvert (ID_{0,1}) \rvert \xrightarrow{+} PN_{0,1}$ (11 wts)
.80	.	82.	84	.0	21	25		$21.02 \longrightarrow ID_0$
.84	.	86.	89	.6	21	30		$\lvert (21.02-03) \rvert \longrightarrow PN_{0,1}$
.89	.	91.	91	.7	21	31		Exit Normal *
								from 03.u0
.u4	.	02.	28	.0	07	29		[.0100000] $\xrightarrow{+}$ AR
.28	.	32.	34	.5	07	25		[.0010000, 0000000]
								from 03.86
.91	.	92.	94	.4	25	26		$(ID_{0,1}) \longrightarrow PN_{0,1}$

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D
PROGRAM PROBLEM: Decimal-Binary Binary-Decimal Conversion

Prepared by H. Lewis

Page 6 of 7
Date: 10/30/58
Line 03

L	P	T or LR	N	C	S	D	BP	NOTES
59		.06.	67	0	26	31		Shift from 03.56
								01/2
03								format (variable)
02								1000000 End code for format

Bendix Computer

MEMORY ALLOCATION CHART

Problem { Intercom 1000 double precision } Page 7 of 7

Date

Prepared by D. Hassell

Check Sum .xx13506 Line 03

U7	U6	U5	U4
9519398	- 1010z8	60431z	821w0zx
U3	U2	U1	U0
-6u6u359	y869yzz	-y713319	67673vz
99	98	97	96
656579w	642yuvz	-64640zy	62623zz
95	94	93	92
66631z	y0612zx	5z600zx	xy5zyzz
91	90	89	88
-5y5y33u	xw5ywzw	-5v5vyvz	5u5x33w
87	86	85	84
x9602zx	5u5u3vz	65w31z	-x659uvy
83	82	81	80
-8429358	x455yzz	-5456v5y	x2542v9
79	78	77	76
5152uvz	-50452vz	x058397	-wy51v3y
75	74	73	72
65231z	4w622xz	wv4xz3x	4v4vyzz
71	70	69	68
-4w4wv5y	w8492zw	4748uvz	-505033y
67	66	65	64
-464733u	444439u	64831z	4242yzz
63	62	61	60
w14533v	4141yzz	-w05351y	3y3zyzz
59	58	57	56
64335z	3w4339w	3v3xyvw	3u3u39v
55	54	53	52
63y31z	vu3xyvx	37386vx	3737yzz
51	50	49	48
3536z9w	-3535359	u3w31z	v233v7w
47	46	45	44
104031z	30310z9	63431z	uy2z2zz
43	42	41	40
-2y2yv38	2x2xyzz	w4u37z	2u2w0z9
39	38	37	36
29570zw	-uu3933u	274y2xz	-u628335
35	34	33	32
62u31z	2424398	2325379	2323yzz
31	30	29	28
601831z	u1262zv	w2v31z	-u0224z9
27	26	25	24
u021u9x	1w1y31v	62031z	-9u1y359
23	22	21	20
601031z	-9u1xuz9	171u795	1719yzz
19	18	17	16
961v315	141631v	102231z	-9216355
15	14	13	12
223231z	1111yzz	61431z	8y0zv7w
11	10	09	08
-8y152v9	x0xyzz	v0w2z5	- v0v358
07	06	05	04
60y31z	809398	223031z	8607yzz
03	02	01	00
w400	1000000	60u31z	-82063z9

FOUR-WORD LINES

LINE	3	2	1	0

FORM BCD 108-15.1

TYPE-IN ROUTINE

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: Type-in Routine

Prepared by D. Hassell

Page 1 of 9

Date: 10/30/58

Line 06

L	P	T or LK	N	C	S	D	BP	NOTES
								Enter all special commands and input commands
.89	.	00.	77	0	06	27		Special Command?
								If "NO", see 06.78 below
.77	.	79.	44	0	31	28		OP ⟶ AR (Yes)
.44	.	46.	46	0	31	31		Interpret
AR		60	OP	0	00	00		See OP below
.50	.	53.	56	0	08	06		Set switch for Command Input OP = 50
.56	.	59.	66	2	30	28		Address ⟶ AR
.66	.	71.	79	1	28	06		Store address ⟶ 06.71
.79	u.	00.	04	0	06	05		Input Routine ⟶ Line 05 (Enter input loop)
.04	.	03.	05	0	07	03		CR format ⟶ 03.03
.05	.	10.	10	0	08	31		Return Carriage
.10	.	10.	10	0	28	31		Wait for Ready
.11	.	71.	72	0	06	21		Destination Address ⟶ 21.03
.72	w.	u2.	46	4	21	31		NC 04.46 (four digit Binary-Decimal conversion)
								Enter here if in "Manual"
.78	u.	00.	15	0	06	05		Typewriter Input ⟶ Line 05
.15	.	16.	20	0	29	23		Clear 23
.20	.	81.	81	5	21	31		N.C. 05.81 See 06.81
								Return from four-digit Binary-Decimal output
.u2	u.	u3.	u5	0	06	19		Type-in Routine ⟶ Line 19
.u5	.	71.	90	0	06	28		Destination Address ⟶ AR
.90	u.	95.	86	0	29	23		Clear 23
.86	.	98.	98	0	22	31		Number Input?
								If "YES", see 06.99 on following page

FORM BCD 108-17.0

Behdix Computer — Los Angeles 45, California

G-15D
PROGRAM PROBLEM: INPUT ROUTINE

Prepared by D. Hassell

Date: 10/30/58
Line 06

L	P	T or LK	N	C	S	D	BP	NOTES
.98		.53.	80	.0	.06	.27		Fixed point number input?
								If "YES", execute input routine in Line 19
.80		.81.	81	.6	.21	.31		/Yes/
								Enter type in cycle
.81		.83.	u0	.0	.12	.31		Gate Input /No/
u0		.u0.	u0	.0	.28	.31		Wait for Ready
u1		.00.	91	.0	.19	.27		Obey Input?
								If "YES", see 06.92 below
.91		.94.	94	.0	.22	.31		Command?
								Enter here for command input
.94	w.	.38.	u6	.5	.21	.31		Mark 38. N.C. 06.u6 Page 06/6
								Enter Floating Point Input
.95		.09.	09	.6	.21	.31		N.C. 06.09 Page 06/6
								Enter here to obey input
.92		.92.	92	.0	.28	.31		Wait for Ready
.93	w.	.12.	u6	.5	.21	.31		Mark 12. N.C. 06.u6 Page 06/6
								Enter to set normalizing constant for number input
.99		.16.	98	.0	.06	.23		[.0000004] (Normalizing Constant) ⟶ 23.00
								Enter after typein of Fixed Point Number (1 slash)
.96		.u0.	u0	.0	.28	.31		Wait for Ready. Go to 06.97
.97		.00.	17 (13)	.4	.19	.24		Input to left of decimal point ⟶ MQ
.13	u.	.18.	21	.0	.07	.19		Line 07 ⟶ Line 19 N.C. 07.21 Page 07/1
								Enter "Compute Automatically"
.69		.01.	02	.0	.06	.00		Set "Automatic" switch. OP = 69
.02		.03.	06	.2	.30	.28		C.C. address ⟶ AR

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D — Prepared by D. Hassell — Page 3 of 7 — Date: 10/30/58

PROGRAM PROBLEM: INPUT ROUTINE — Line 06

L	P	T or LK	N	C	S	D	BP	NOTES
06	.	09.	09	0	21	31		N.C. 00.09 Page 00/4
								Enter "Permit Type in of Fixed Point Data"
51	.	53.	49	0	29	06		Set switch OP = 51
49	.	59.	19	0	08	20		Odd Address Extractor → 20.03
19	.	23.	26	0	31	27		Address Odd?
								If "YES" see 06.27 below
26	.	39.	54	0	08	20		Restore extractor
54	.	55.	66	3	30	28		Double Precision Address → AR
27	.	44.	44	4	21	31		See Error Loop
								Enter "Permit Type in of Floating Point Data"
52	.	53.	49	0	08	06		Set switch OP = 52
								Enter "Read Punched Paper Tape"
55	.	08.	64	0	26	31		Form Destination CH OP = 55
64	.	65.	70	2	25	28		Destination CH +→ AR
70	.	73.	74	0	06	29		Command +→ AR
74	u.	00.	15	0	08	05		Read Routine Line 05 N.C. 08.15
								Enter "Halt and Permit Manual Operation"
67	.	01.	82	0	29	00		Set for "Manual Operation" OP = 67
82	.	01.	01	0	21	31		Return to Manual
								Enter "B. P. Halt"
68	.	70.	82	0	16	31		Halt OP = 68
								Enter "Ring Bell"
63	.	62.	82	0	17	31		Ring Bell OP = 63

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: ERROR LOOP

Prepared by D. Hassell

Page 4 of 7

Date: 10/30/58

Line 06

L	P	T or LK	N	C	S	D	BP	NOTES
.61	.	61.	61	.0	.06	.06		Do nothing OP = 61
61a		02	02	5	21	31		Set Selective Print
.62	.	62.	62	.0	.06	.06		Do nothing OP = 62
62a		00	00	5	21	31		Remove Selective Print
.31	.	33.	30	.0	.29	.31		Reset Overflow Enter Error Loop
.30	u.	44.	45	.0	.08	.23		Output Routine ⟶ 23
.45	.	46.	47	.0	.06	.28		Error Type-out Format
.47	.	03.	12	.0	.28	.03		⟶ 03.03
.12	.	37.	14	.0	.06	.28		R. C. ⟶ 23.00
.14	.	16.	18	.0	.28	.23		
.18	.	24.	25	.2	.20	.28		Tally in AR
.25	.	25.	29	.0	.17	.31		Ring Bell
.29	.	33.	33	.7	.21	.31		N. C. 23.03 = 08.41
								Type Period and Tab the Return to 06.43 below
.37	.	22.	22	.6	.21	.31		R. C. (obey from Line 23)
.22	.	23.	24	.0	.28	.27		(AR) = 0?
								If (AR) ≠ 0
.24	.	24.	43	.0	.16	.31		Halt
.43	.	78.	78	.0	.21	.31		Transfer to Typeout of Last Command Obeyed
								Enter if "2 slashes" are typed with fixed point input
.84	.	92.	92	.5	.21	.31		See 06.92 Page 06/1
								Here from 01.ul Page 01/6
.u7	.	23.	28	.2	.30	.21		Address ⟶ 21.03
.08	.	15.	17	.0	.01	.20		[7z00000] ⟶ 20.03

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: CONSTANTS

Prepared by D. Hassell

Page 5 of 7

Date: 10/30/58

Line 06

L	P	T or LK	N	C	S	D	OP	NOTES
.17	.	19.	21	.	0	31	25	Wd → ID
.21	.	16.	38	.	0	24	31	Shift Wd to OP position
.38	.	39.	40	.	0	30	29	CH +→ AR
.40	.	41.	44	.	0	25	29	OP +→ AR
.46 AR	. u	47	OP	0	CH	19		N.C. 19.OP
.48	.	01.	01	.	0	21	31	See 01.44 Page 01/6
.01	.	03.	04	.	0	22	26	- See 01.a Page 00/1
.03	.	u4.	01	.	1	24	08	See 01.01 Page 01/3
.07	.	u5.	01	.	1	24	08	See 01.07 Page 01/4
.83	.	u4.	01	.	1	24	08	See 01.81 Page 01/2
.85	.	u3.	01	.	1	24	08	See 01.83 Page 01/2
.87	.	u6.	01	.	1	24	08	See 01.85 Page 01/2
.88	.	00.	19	.	0	00	28	
.33	.	93.	93	.	0	21	31	See 01.32 Page 01/4
.32								

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D
PROGRAM PROBLEM: CONSTANTS

Prepared by D. Hassell

Page 6 of 9
Date: 10/30/58
Line 06

L	P	T or LK	N	C	S	D	BP	NOTES
.u6	u.	u7.	u4.	0.	08.	19		Enter Command
.u4	.	05.	05.	6.	20.	31		Conversion
.09	u	.10.	11.	0.	07.	19		Here from typein of floating point
EMPTY	. SPACES							= 28, 32, 41
.34	w.	93.	u2.	1.	21.	31		See 01.33 Page 01/4
.u3	(					)		Storage of "MARK I"
.35	.	97.	97.	0.	21.	31		See 01.34 Page 01/4
.36	u.	20.	w7.	0.	00.	19		See 04.35 Page 04/6
.39	w.	97.	u3.	1.	21.	31		See 01.38 Page 01/4
.53	(	.	.	.	.	)		Fixed Point or Command Switch
.58	.	00.	77.	0.	00.	28		See 01.50 Page 01/6
.73	u.	20.	v3.	0.	19.	00		See 06.70 Page 06/3
.76	u.	47.	00.	0.	00.	05		See 00.75 Page 00/4
.00	(					)		Manual or Special Command Switch
.16								.0000004 Normalizing Constant
.23								.0002000 See 01.21 Page 01/3
.42								.3900000 See 01.25 Page 01/5
.46								.z192000 Wait-sign-period-end-end
.57								.0u00000 See 04.53 Page 04/5
.59								.0046000 See 01.58 Page 01/1
.60								.000u000 See 00.58 Page 00/3
.65								.1000000 See 04.64 Page 04/5
.71	(					)		... Hold Address of Input
.75								.7z00000 See 08.72 Page 08/1

FORM BCD 108-17.0

MEMORY ALLOCATION CHART

Problem { Intercom 1000 double precision Page 7 of 7
Date

Prepared by D. Hassell Check Sum .9w7wv36 Line 06

U7	U6	U5	U4
971wvx5	6v68113	w75u0xw	- 505u9z
U3	U2	U1	U0
	67690x3	805v27v	646439z
99	98	97	96
90620x7	v5500xv	-8011278	646439z
95	94	93	92
- 909uvz	-u66u6vz	-8w6u6vz	5w5w39z
91	90	89	88
5y5y2xz	5z563v7	804x0xv	801301w
87	86	85	84
yu01708	62622xz	y701708	-5w5w6vz
83	82	81	80
y801708	1012vz	536419z	-5151uvz
79	78	77	76
40w5	z0w5	wz2w3zw	2z00005
75	74	73	72
7z00000	z105	1471260	-y62y2vz
71	70	69	68
600040	w94u0xx	81020w0	465221z
67	66	65	64
81523u0	w74z786	1000000	4246v3w
63	62	61	60
3y5223z	vy3y0w6	vx3x0w6	u000
59	58	57	56
46000	804x01w	u00000	vv42vxw
55	54	53	52
84035z	3842zxw	3839v15	3631106
51	50	49	48
v5313u6	v538106	vv13114	1012vz
47	46	45	44
830w383	z192000	2z2z0xw	2y2y3zz
43	42	41	40
4y4y2vz	3900000		2u2w33x
39	38	37	36
y1676vz	28283xx	-1616uvz	147z013
35	34	33	32
61612vz	xx666vz	5x5x2vz	
31	30	29	28
211y3vz	2w2x117	-2121yvz	
27	26	25	24
-68682vz	u736114	191x23z	182v21z
23	22	21	20
2000	181839v	102631z	-51516vz
19	18	17	16
971u3zv	9819u9w	93153z9	4
15	14	13	12
11143v7	9012397	12150z3	u50y0xw
11	10	09	08
w7480x5	u0u39z	u0v0z3	8z11034
07	06	05	04
y901708	9092vz	u0u11z	83050y3
03	02	01	00
y801708	406vxw	83842xu	2x23z

FOUR-WORD LINES

LINE	3	2	1	0

FORM BCD 108-15.1

FIXED POINT DECIMAL-TO-FLOATING POINT BINARY CONVERSION

Bendix Computer — Los Angeles 45, California

Page 1 of 6

G-15D — Prepared by D. Hassell — Date: 10/30/58

PROGRAM PROBLEM: Fixed Point Decimal-to-Floating Point Binary Conversion — Line 07

L	P	T or LK	N	C	S	D	BP	NOTES
								Fixed point decimal-to-floating point binary conversion
.21		.22	.57	.3	.07	.28		$32 \cdot 2^{-26}$ ($20 \cdot 2^{-26}$ in hex) + 36(=54·dec.)·2^{-28} $\xrightarrow{-}$ AR
.57		v2	.52	.0	.27	.31		Normalize controlled by AR (locate dec. pt.)
.52		.55	.58	.1	.28	.21		(AR) $\xrightarrow{+}$ 21.03
.58		.60	.65	.4	.23	.24		Input $\longrightarrow$ MQ
.65		v2	.59	.0	.27	.31		Shift off leading zeros
.59		.08	.71	.0	.26	.31		Shift off normalizing constant
.71		.96	.97	.0	.29	.28		Clear AR
.97		.14	.16	.4	.24	.27		Input = 0?
								If ≠ 0, see 07.17 below
.16		.19	.24	.0	.23	.31		Clear Register
.24		.27	.29	.0	.07	.23		R.C. $\longrightarrow$ 23.03
.29		.02	.35	.0	.26	.31		
.35		.70	.70	.2	.21	.31		N.C. Floating Point Decimal-Binary conversion
.17		v6	.95	.0	.27	.31		Normalize accumulating tally in AR
.95		.10	.15	.0	.07	.20		0000003 $\longrightarrow$ 20.02
.15		.18	.20	.2	.28	.21		(AR) (Tally) $\longrightarrow$ 21.02
.20		.22	.25	.3	.31	.28		Odd number of shifts $\xrightarrow{-}$ AR
.25		.28	.31	.0	.23	.25		Sign $\longrightarrow$ IP
.31		.32	.34	.5	.24	.25		MQ $\longrightarrow$ ID
.34		.36	.45	.3	.07	.29		0000008 $\xrightarrow{-}$ AR
.45		.22	.70	.0	.26	.31		Normalize to the Decimal Digit
.70		.95	.96	.2	.21	.28		Binary exponent into AR
.96		u6	.01	.3	.30	.29		Zero positions $\xrightarrow{+}$ AR

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D

Prepared by D. Hassell

Page 2 of 5

Date: 10/30/58

PROGRAM PROBLEM: FLOATING POINT INPUT (NORMALIZE)

Line 07

L	P	T or L/K	N	C	S	D	CP	NOTES
.01	u.	20.	26	.2	.28	.29		Shift 18 bits (AR = Exp.·2^{-8}
.26		.27.	46	.0	.07	.23		RC ⟶ 23.03
.46		.50.	53	.4	.25	.24		Dec. Mantissa ⟶ MQ
.53	u.	28.	70	.0	.02	.19		See 02.70-27
								Enter from 02.27
.38		.40.	44	.3	.23	.31		Binary Exp. ⟶ ID
.44		.51.	51	.2	.21	.31		N.C. 02.51
								Enter Floating Point Input (cont'd)
.11		.14.	14	.0	.23	.31		Clear 2-word registers
.14		.16.	19	.4	.23	.24		Input ⟶ MQ
.19		.22.	23	.0	.23	.27		Input = 14 digits?
								If "YES", see 07.24 Page 07/1
.23		.v2.	93	.0	.27	.31		Shift off insignificant zeros /No/
.93		.06.	24	.0	.26	.31		Shift off normalizing constant
.27	w.	u4.	93	.0	.21	.31		R.C. for type-in sequence
.00								.xu00000 See 03.u6 Page 03/4 & 3
.02								.0100000 See 03.u4 Page 03/5
.03								.4400000 See 00.71 Page 00/4
.04								.0000101 See 02.u6 Page 02/3
.05								.9x1301w See 00.04 Page 00/1
.06								Format positions for Fixed Point Output
.07								
.08								

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D
PROGRAM PROBLEM: CONSTANTS

Prepared by D. Hassell

Page 3 of 6
Date: 10/30/58
Line 07

L	P	T or L_K	N	C	S	D	BP	NOTES
.09	(						)	Extractor Shift Control — See 01.80 Page 01/5
.10								.0000003 — See 07.95 Page 07/1
.12								.1000000 $-5^2 \cdot 2^{-3} + 5^2 \cdot 2^{-9} + 5^4 \cdot 2^{-21} + 5^4 \cdot 2^{-33}$ See 02.10 Page 02/1
.13								-vuw93893
.18								.1344135 $2^{-2}\log^2$ See 02.17 Page 02/1
.22								.00000zy — See 07.21 Page 07/1
.28								.5000000 — See 02.26 Page 02/3
~~.32~~								~~.0000000~~
.33								.0010000 — See 03.28 Page 03/
.36								.0000008 — See 07.34 Page 07/1
.37		.90.	95.	4.	07.	26		See 02.36 Page 02/3
.40		.99.	u7.	0.	07.	28		See 03.39 Page 03/2
.41								.1999u00 10^{-1} See 03.40 Page 03/
.43								.005xz65 $2^{-7}(5 - 16\log^2)$ See 02.42 Page 02/1
.47								.09u209u $-2^{-3}\log_{10}{}^2$ See 03.46 Page 03/2
.48								.0400000 49.50 — See 04.47 Page 04/6
.51								8000000 — See 00.35 Page 00/4
.56								.1200000 — See 02.55 Page 02/2
.60								.0000001 — See 02.59 Page 02/1
.61		..	00.	36.	4.	00.	25	See 00.60 Page 00/2
.62								.94u2000 $10^{12} \cdot 2^{-56}$ See 02.60 Page 02/2
.63								.0000y8x
.64	u	00	u7	0	00	03		See 01.56 Page 01/6
.67		..	72.	97.	4.	07.	26	See 02.66 Page 02/2
.68								.0u00000 — See 02.67 Page 02/2
.72								.056y215 $21 \cdot 2^{-57}$

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D
PROGRAM PROBLEM: CONSTANTS

Prepared by D. Hassell

Date: 10/30/58
Line 07

L	P	T or Lk	N	C	S	D	BP		NOTES
.73								-.9xw5uxu	= $10^{32} \cdot 2^{-107}$
.74								.x99xw30	$48 \cdot 2^{-57}$
.75								-.x3w21vw	$10^{24} \cdot 2^{-80}$
.76								.y08004u	$74 \cdot 2^{-57}$
.77								.8y1vw9v	$10^{16} \cdot 2^{-54}$
78								.0000065	$101 \cdot 2^{-57}$
.79								vyvw200	$10^{8} \cdot 2^{-27}$
.80								.0000080	$128 \cdot 2^{-57}$ = floating point zero
.81								.0000000	
.82								.308w49u	$154 \cdot 2^{-57}$
.83								uvww771	$10^{-8} \cdot 2^{26}$
.84								x889wv5	$181 \cdot 2^{-57}$
.85								-.y69594v	$10^{-16} \cdot 2^{53}$
.86								u88yuwz	$207 \cdot 2^{-57}$
.87								-.9uvy14w	$10^{-24} \cdot 2^{79}$
.88								.u8u74yu	$234 \cdot 2^{-57}$
.89								-.wzv11yu	$10^{-32} \cdot 2^{106}$
.90								.7757v04	$260 \cdot 2^{-57}$
.91								-.8v61313	$10^{-40} \cdot 2^{132}$
.92								.0100000	See 03.92 Page 03/4
.94		99	00	0	07	28		.y3000zw	See 03.93 Page 03/3
.98								.001197u	$5 \cdot 10^{-13}$ See 03.98 Page 03/4
.99								0000000	
.u0								.u000004	10^{P2}
.u1								.w800007	

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: CONSTANTS

Prepared by D. Hassell

Page 5 of 6

Date: 10/30/58

Line 07

L	P	T or L_R	N	C	S	D	OP	NOTES
.u2								.zu000Cu
u3								.9w4000y
.u4								.w350011
.u5								.z424014 — 10^{p2} (Cont'd)
.u6								.9896818
.u7								.v6xv6xv — See 03.u6 Page 03/4

FORM B60 103-17.0

Bendix Computer

MEMORY ALLOCATION CHART

Problem _____ Page 6 of 6

Date _____

Prepared by _____

Check Sum .z5x1zz3 Line 07

U7	U6	U5	U4
v6xv6xv	9896818	z424014	w350011
U3	U2	U1	U0
9w4000y	zu0000u	w800007	u000004
99	98	97	96
-0-	1197u	-8y1031v	yu01zxx
95	94	93	92
8u0z0z4	y3000zw	61835z	100000
91	90	89	88
-8v61313	7757v04	-wzv11yu	u8u74yu
87	86	85	84
-9uvy14w	u88yuwz	-y69594v	x889wv5
83	82	81	80
uvww771	308w49u	-0-	80
79	78	77	76
vyvw200	65	-8y1vw9v	y08004u
75	74	73	72
-x3w21vw	x99xw30	-9xw5uxu	56y215
71	70	69	68
y0613vw	xz60uvw		u00000
67	66	65	64
-w8610zu		703v37z	6v008
63	62	61	60
y8x	94u2000	-8024019	1
59	58	57	56
84735z	-vw412z8	703437z	1200000
55	54	53	52
		1w46053	v73u795
51	50	49	48
8000000			400000
47	46	45	44
9u209u	-v235338	164635z	3333uvz
43	42	41	40
5xz65		1999u00	y36v0zw
39	38	37	36
	282wyzz	-xu5z0zu	8
35	34	33	32
4646uvz	u42xwzx	10000	-0-
31	30	29	28
-2222719		22335z	5000000
27	26	25	24
y85x2vz	1w2y0z7	9w1z2z9	9v1x0z7
23	22	21	20
705x37z	zy	1739wzw	9619zzw
19	18	17	16
96172zv	1344135	745z37z	13182zz
15	14	13	12
9214v95	-90132z8	-uw93893	1000000
11	10	09	08
y0y2zz	3	24	88
07	06	05	04
8000006	188	9x1301w	101
03	02	01	00
4400000	100000	141uv9x	xu00000

FOUR-WORD LINES

LINE	3	2	1	0

FORM BCD 108-15.1

COMMAND CONVERSION

Los Angeles 45, California

G-15 D
PROGRAM PROBLEM : Command Conversion

Prepared by D. Hassell

Page 1 of 6
Date: 10/30/58
Line 08

L	P	T or LK	N	C	S	D	BP	NOTES
								Enter Command Conversion
.05		.09	.09	.0	.23	.31		Clear Registers
.09		.12	.15	.0	.23	.28		Input [K OP CH WD] → AR
.15		.16	.17	.0	.28	.24		[K OP CH WD] → MQ_0
.17		.32	.53	.0	.24	.31		[(K OP C) -2^{-29}] → MQ_1 [H WD 00 00] → MQ_0
.53	u.	.56	.57	.2	.24	.21		K OP CH WD → 21.02, [H WD 00 00] → 21.03
.57		.10	.24	.0	.26	.31		0 OK OP CH → MQ_1
.24		.26	.28	.4	.08	.20		[z000000] → 20.02, [0zz0000] → 20.03
.28		.29	.30	.0	.24	.28		[0 OK OP CH] → AR
.30		.34	.35	.4	.27	.28		(27.02) = [K OK OP CH] → AR, (27.03) = [K WD OP CH] → AR
.35		.37	.36	.6	.20	.31		Return to Mark
								Enter to Store Command typed in — Mark Set by OP = 50
.38		.03	.16	.0	.08	.03		Format → 03.03
.16	u.	.21	.20	.0	.28	.23		Converted input → 23
.20		.22	.41	.0	.21	.28		K OP CH WD → AR (Type In) → AR
.41		.41	.41	.0	.28	.31		Wait for Ready
.42		.44	.43	.0	.08	.31		Type Back Command — Used during all AR type out
.43		.43	.43	.0	.28	.31		Wait for Ready
.44		.71	.72	.0	.06	.21		Store Command → AR
.72		.75	.76	.0	.06	.20		[7z00000] → 20.03
.76		.79	.11	.0	.30	.25		CH → AR
.11		.10	.22	.0	.26	.31		Shift to Destination Position
.22		.23	.46	.0	.31	.28		WD → AR
.46		.29	.31	.0	.25	.29		CH·2^{-28} $\xrightarrow{+}$ AR
.31		.33	.55	.0	.08	.29		Command $\xrightarrow{+}$ AR

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D
PROGRAM PROBLEM: TYPE-IN SEQUENCE

Prepared by D. Hassell

Page 2 of 6
Date: 10/30/58
Line 08

L	P	T or LK	N	C	S	D	BP	NOTES
.55		.57.	57	.0	.31.	31		Obey
.33		.00. wd	45	.4 .0	.23.	00 CH		number input / command input } to storage
.45		.47.	47	.0	.22.	31		Command input?
								If "NO" See 08.48 below
.47		.50.	52	.0	.08.	28		[9x00000] Address Incrementer (Yes)
.52		.71.	73	.2	.06.	29		Address +→ AR
.73		.75.	77	.0	.29.	31		Increment CH?
								If "YES" See 08.78 below
.77		.18.	21	.3	.08.	29		Reduce by excess 28 (No)
.21		.23.	29	.0	.26.	29		Add sign (+ if command, - if numbers)
.29		.71.	74	.0	.28.	06		Restore Incremented Command
.74		.01.	36	.0	.00.	27		Manual?
								If "NO" See 08.37 below
.36	u.	.37.	79	.0	.06.	19		(Line 06) → 19 NC 06.79 (Yes)
.37		.01.	01	.0	.21.	31		(Continue) computing automatic
.78		.79.	21	.0	.08.	29		Increment CH
.48		.51.	52	.0	.08.	28		[9y00000] Increment by 2 see 52 above
								Enter to Precess before Punching OP-39 (cont.)
.u3	u.	.00.	82	.0	.19.	27		19.u4 - u7 = 0?
								If "NO" See 08.83 Page 08/3
.82		.84.	u0	.0	.08.	31		Pair with u0' following "YES"

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: LINE 19 PRECESSION

Prepared by D. Hassell

Page 3 of 6

Date: 10/30/58

Line 08

L	P	T or LK	N	C	S	D	BP	NOTES
.u0	.	u2.	99	0	00	31		Precess four words
99	u.	u4.	82	0	19	27		19·u0-u3 = 0?
								If = 0, see 08.82 on preceding page
83	.	85.	93	0	10	31		Punch [NOT]
93	.	01.	01	0	21	31		Go to C. C. pick up
[75	u.	80.	81	2	30	28	]	See 01.74 Page 01/1
81	.	83.	84	0	00	20		[7z00000] → 20.03 "Read Punch Tape" (cont'd)
84	.	87.	92	0	31	29		W_d →(+) AR
92	u.	93.	86	0	29	19		Clear Line 19
86	.	88.	96	0	15	31		
96	.	96.	96	0	28	31		Wait for ready
97	.	u7.	u7	0	31	31		Store Input
90	.	u2.	78	1	08	28	*	See 01.89 Page 01/2
12	.	19.	19	0	21	31		
32	u.	38.	34	1	22	25		Number input (cont'd)
34	u.	39.	44	1	25	23		prepare to store input NC 08.44 page 08/1
40	.	46.	45	5	20	31		
07								000008W = 140 (dec) = 7 inch leader tally
85	.	u5.	01	1	24	08		See 01.84 Page 01/2
87	.	u7.	01	1	24	08		See 01.86 Page 01/2
89	u.	00.	02	0	08	23		See 01.88 Page 01/2

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

Prepared by D. Hassell

Date: 3/13/59

G-15D

PROGRAM PROBLEM: PUNCH ROUTINE (Cont.): CONSTANTS

Line 08

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99
U0	U1	U2	U3
U4	U5	U6	

L	P	Tr Lk	N	C	S	D	BP	NOTES
			PUNCH ROUTINE CONTINUED					
u7	u	03 23	u1 w1	0	19	27		(19.00-03) = 0?
u1		00	u2	0	00	19		Yes Non-zero → 19.00
u2		35	35	4	21	31		No NC 04.35 Page 4/6
01								9w00000 See 01.00 Page 01/1
03								-.8w00001 See 08.38 Page 08/1
06								.7z003y0
08								.0000081 See 04.07 Page 04/4
10		00	19	0	00	28		See 00.09 Page 00/4
13								.1w00000 See 00.12 Page 00/1
14								-.8000020 See 00.13 Page 00/3
18								.9w00000 See 00.12 Page 01/1
19								.0000030 See 04.18 Page 04/1
23								.0000031 See 04.22 Page 04/2
25								.9w00000
26								.z000000 See 08.24 Page 08/1
27								.0zz0000
39								.007z000 See 00.38 Page 00/2
49								.0031000 See 00.67 Page 00/2
50								See 08.47 Page 08/6 .9x00000 See 01.39 Page 01/4
51								.9y00000 See 08.48 Page 08/2
54								.0009000 See 00.53 Page 00/2
58								-.zzzzz00 See 00.57 Page 00/2
59								.0100000
61								.0000010 See 01.60 Page 01/1

FORM BCD 108-5.0

Bendix Computer — Los Angeles 45, California

G-15D
PROGRAM PROBLEM: CONSTANTS

Prepared by D. Hassell

Page 5 of 6
Date: 10/30/58
Line 08-

L	P	T or LK	N	C	S	D	BP	NOTES
.62								* .0000200 — See 00.60 Page 01/1
.64								* .0000008 — See 03.61 Page 03/1
.65								
.67								.7z003y0 — See 00.59 Page 00/4
.68		.00.	01.	4.	25.	00		See 00.67 page 00/2
.69		.00.	00.	4.	00.	25		See 01.67 Page 01/4
.71		.21.	21.	0.	21.	31		See 04.68 Page 04/1
.80	u.	85.	72.	3.	21.	28		See 00.76 Page 00/3
.88	u.	u5.	01.	0.	08.	23		See 01.87 Page 01/2
.91								.000w400 — See 04.74 Page 04/6
.94		.u4.	u6.	4.	08.	22		See 01.75 Page 01/2
.95								.0000080 — See 04.94 Page 04/4
.98	u.	47.	00.	0.	00.	19		See 01.96 Page 01/6
.u4								.00001z1 — floating point format
.u5								8030000
.u6								.00001x1
EMPTY	SPACES							00, 02, 04, 56, 60, 63, 66, 70.
.79								.0000010-
90	u	10	78	0	08	21		01.89 page 01/2

FORM BCD 108-17.0

Bendix Computer

MEMORY ALLOCATION CHART

Problem _____ Page 6 of 6

Date _____

Prepared by _____

Check Sum -.0z8v9y3 Line _____

U7 177927v	U6 1x1	U5 8030000	U4 1z1
U3 5227v	U2 -23232vz	UI 8066013	UO 666301z
99 685227v	98 2z00013	97 6v6v3zz	96 606039z
95 80	94 -y86u116	93 1012vz	92 5x563v3
91 w400	90 y64y51w	89 2117	88 6901117
87 yv01708	86 58601zz	85 y901708	84 x75w3zx
83 555x15z	82 546411z	81 x354014	80 5548yvw
79 - 20	78 501511x	77 9215x1x	76 wz0v3x9
75 5051vxw	74 812401v	73 4v4x3vz	72 wv4w0x4
71 15152vz	70	69 -8000019	68 -8001320
67 7z003y0	66	65	64 8
63	62 200	61 10	60
59 100000	58 -zzzzz00	57 u1835z	56
55 39393zz	54 9000	53 3839v15	52 w7498xx
51 9y00000	50 9x00000	49 31000	48 v33411w
47 v23411w	46 9x1z33x	45 2z2z2xz	44 w7480x5
43 2v2v39z	42 2w2v11z	41 292939z	40 -2y2x69z
39 7z000	38 8310103	37 1012vz	36 254z0x3
35 -2524u9z	34 272w737	33 802x2y0	32 26226x9
31 u13711x	30 -u22337w	29 w74u386	28 1y1y31w
27 zz0000	26 z000000	25 9w00000	24 -9u1w114
23 31	22 182y3zw	21 971x35x	20 96292vw
19 30	18 9w00000	17 203531z	16 1514397
15 1111398	14 -8000020	13 1w00000	12 13132vz
11 u1635z	10 801301w	09 8w0z2zw	08 81
07 8w	06 7z003y0	05 9092zz	04
03 -8w00001	02	01 9w00000	00

FOUR-WORD LINES

LINE	3	2	1	0

FORM BCD 108-15.1

CONTROL AND COMMAND CONVERSION

Los Angeles 45, California

G-15D PROGRAM PROBLEM: INTERCOM 1000 D.P. (Control and Command Conversion)

Prepared by D. Hassell

Date: 10/30/58

Line 00

L	P	T or LK	N	C	S	D	BP	NOTES
00		00	45	0	17	31		Ring Bell <scf> entry; end of Manual cycle
45		01	63	0	29	00		Set for Manual operation
63		00	85	0	00	06		Set switch to obey type-in
85		85	85	0	28	31		Wait for ready
86	u	87	89	0	06	19		Type-in routine (line 06) into line 19
89		89	89	6	21	31		N.C. 06.89 (see page 06/1)
01m			-0-					Go to 00 above (manual)
01a		03	04	0	22	26		Enter C.C. interpretation. C.C. $\longrightarrow$ PN_1 (Automatic) BP
04		05	06	0	07	28		$[9x130lw] \xrightarrow{+}$ AR
06		08	08	3	23	31		C.C. into ID. (Extractor = 7z003y0)
08		09	10	2	25	29		Increment C.C.
10		12	12	0	29	31		Increment channel?
								YES, see 00.13 page 00/3 NO
12		13	14	3	08	29		$[1w00000] \xrightarrow{-}$ AR (subtract excess 28 from address of C.C
14		15	16	0	28	22		Store incremented C.C.
16		18	18	0	31	31		Interpret C.C.
AR		xx	19	0	xx	28		INTERCOM command (dec. format = K WD OP CH) $\longrightarrow$ AR
								ENTER COMMAND CONVERSION
19		20	22	4	28	26		AR (K WD OP CH) $\longrightarrow$ $PN_{0,1}$
22		25	25	3	23	31		(0 DO oO AO) $\longrightarrow$ ID_1; $[n\ OD\ OP\ OH] \longrightarrow ID_0$ (n = 1 when K is odd)
25		26	27	2	25	29		double units digits (D P H) or $[t + (u/16)\cdot 2]$
27	u	30	30	2	28	29		$AR\cdot 2^2 = 4t + 8u/16$
30		31	32	2	25	29		double units digits (W O C) = 5t + u/2
32		33	34	2	28	29		AR·2 (sets overflow if K ≠ 0) = 10t + u
34		35	36	2	28	26		INTERCOM command (Binary) $\longrightarrow$ PN_1
36		38	38	3	23	31		1/2 source-channel $\longrightarrow$ ID_1 (extractor = 0000lz0)

COMMAND CONVERSION

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D
PROGRAM PROBLEM: INTERCOM 1000 D. P. CONTROL

Prepared by D. Hassell

Page 2 of 7
Date: 10/30/58
Line 00

L	P	T or LK	N	C	S	D	BP	NOTES
.38		.39	.40	.0	.08	.20		007z000 ⟶ 20.03
.40		.41	.42	.2	.25	.29		2 x 1/2 S = source channel
.42		.43	.44	.0	.28	.21		Converted Command ⟶ 21.03
.44		.46	.46	.0	.29	.31		K ≠ 0? if yes see 00.47/3
.46		.47	.48	.2	.31	.28		|OP| ⟶ AR
.48		.49	.50	.3	.08	.29		$31 \cdot 2^{-16}$ (hex for 49) $\xrightarrow{-}$ AR
.50		.52	.52	.0	.22	.31		OP ≥ 49
								Enter all OP ≥ 49
.52		.63	.67	.0	.28	.27		OP = 49
								Enter "Store" (OP = 49)
.67		.68	.73	.0	.08	.28		Store dummy ⟶ AR OP=49
.73		.08	.82	.0	.24	.31		Shift source to destination
.82		.83	.84	.0	.00	.20		Extractor (7z00000) ⟶ 20.03
.84		.85	.87	.1	.25	.29		Destination Channel $\xrightarrow{+}$ AR
.87		.88	.91	.4	.21	.25		A ⟶ ID
.91		.95	.99	.0	.31	.29		Destination word $\xrightarrow{+}$ AR
.99		.u1	.u1	.0	.31	.31		
AR		xx	01	4	25	xx		Obey AR
.53		.54	.55	.0	.08	.29		$9 \cdot 2^{-16}$ $\xrightarrow{+}$ AR
.55		.57	.57	.0	.22	.31		49 < OP > 39
								Enter all arithmetic operations
.57		.58	.60	.4	.08	.20		Extractors ⟶ 20.02-03
.60		.61	.62	.0	.07	.28		Dummy operand pickup command ⟶ AR
.62		.63	.64	.0	.21	.29		Intercom command (binary) $\xrightarrow{+}$ AR

FORM BCD 108-17.8

Los Angeles 45, California

G-15D

PROGRAM PROBLEM: INTERCOM 1000 D. P. (CONTROL & TRANSFER OPS)

Prepared by D. Hassell

Date: 10/30/58

Line 00

L	P	T or LK	N	C	S	D	BP	NOTES
.64		66	66	.4	21	.31		NC see 04.66 page 04/1
								Enter all OP < 40
.58		.60	.61	.0	06	29		$(u \cdot 2^{-16}) \xrightarrow{+}$ AR
.61		.65	.65	.0	22	31		OP > 29?
								Enter 40 > OP > 29
.65		.67	.67	.1	21	31		NC see 01.67 page 01/4
								Enter OP < 30 Enter all transfers
.66		.67	.69	.2	31	28		OP → AR
.69		.71	.76	.0	22	26		C. C. → PN
.76		.80	.81	.0	08	29		Dummy Command → AR 5548Y V8 = (08.80)
.81		.83	.83	.0	31	31		Interpret Intercom Command
AR	u	85	72 + OP	3	21	28		(sign of A) $\xrightarrow{-}$ AR NC see (OP + 72)
.88		.u3	.01	.0	06	22		Mark I → CC OP = 16
								Enter 49 < OP < 70
.68		.00	.85	.0	29	06		Set switch to store input
								Enter from Command Conversion when K ≠ 0
.47		.49	.49	.1	21	31		NC 01.49 page 01/1
								Enter to Increment C.C. channel
.13		.14	.14	.0	08	29		$(-.8000020) \xrightarrow{+}$ AR
								Enter "Return to Mark II"
.90		.u3	.01	.0	00	22		Mark II → C.C. OP. =18
								Enter transfer positive
.92		.93	.94	.3	28	28		- (minus sign of A) → AR OP = 20
.94		.96	.03	.0	22	31		Transfer? OP = 22
								Yes see 00.03 page 00/4 No see 00.04 page 00/1

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D
PROGRAM PROBLEM: INTERCOM 1000 D. P. (CONTROL & TRANSFER OPS.)

Prepared by D. Hassell

Page 4 of 7
Date: 10/30/58
Line 00

L	P	T or LK	N	C	S	D	BP	NOTES
.03		.07	.09	.2	.30	.28		New Address of C.C. ⟶ AR
.09		.10	.14	.0	.08	.29		Dummy command pickup $\xrightarrow{+}$ AR
								Enter "Transfer Zero"
.95		.97	.03	.0	.21	.27		A = 0? "NO", see 00.04 page 00/1 "YES", see 03 above OP = 23
								Enter "Mark Transfer I"
.98		.u3	.03	.0	.22	.06		Store Mark I (C.C.) OP = 26
								Enter "Mark Transfer II"
.u0		.u3	.03	.0	.22	.00		Store Mark II (C.C.) OP = 28
								Enter "Transfer Unconditional"
.u1		.u3	.03	.0	.00	.00		See 00.03 above OP = 29
								Enter "Type Location of Last Command Executed"
.78		.79	.69	.0	.22	.21		C.C. ⟶ 21.03 OP = 06
.59		.67	.70	.0	.08	.20		7z003y0 ⟶ 20.03
.70		.71	.U6	.2	.31	.28		|C.C. Address| ⟶ AR
.U6	W	.78	.79	.1	.21	.31		N.C. see 01.79 page 01/3
								Enter "Perform Subroutine" (Line 05)
.74	u	.75	.75	.0	.06	.05		Line 06 ⟶ 05 OP = 02
.75		.76	.77	.0	.06	.28		Dummy ⟶ AR
.77		.u7	.u7	.5	.21	.31		N.C. 05.u7 page 06/4
								Enter "Perform Subroutine" (Line 19)
.80		.95	.95	.1	.21	.31		N.C. 01.95 page 01/6 OP = 08
								Enter "Position Typewriter Carriage" output conversion
.71		.03	.31	.0	.07	.03		C.R. Format ⟶ 03.03
.31		.33	.35	.3	.26	.28		(No. of positions) $\xrightarrow{-}$ AR
.35		.51	.39	.0	.07	.25		Position Control ⟶ ID_1

FORM BCD 08-17.0

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: INTERCOM 1000 D. P. (CONTROL & TYPEOUT)

Prepared by D. Hassell

Page 5 of 7

Date: 10/30/58

Line 00

L	P	T or L_R	N	C	S	D	BP	NOTES
39	u.	70.	54	4	26	30		Shift number of C.R. positions
54	.	55.	28	0	28	27		Any Tabs? C.R.'s?
28	.	30.	01	4	26	27		Through positioning paper? No
								YES, see 00.01 page 00/1 NO, see 00.02 below.
.02		03.	31	0	01	03		Tab format → 03
								Enter to convert position control
29	.	43.	49	1	00	29		Extra bit complement $\xrightarrow{+}$ AR
49	.	46.	23	0	26	31		Shift position control bit
23	.	23.	23	0	28	31		Wait for ready
24	.	03.	37	0	03	02		Format → 02
37	u.	38.	56	0	29	19		Clear Line 19
56	.	u6.	33	4	25	19		Position control → 19
33	.	35.	28	0	09	31		Position Paper
								Enter Output Routine
17	.	17.	17	0	28	31		Wait for Ready
18	.	20.	u7	0	09	31		Execute Output
u7	u.	24.	99	1	24	02		Format → 02.02-03
79	.	u7.	05	0	25	19		Sign → 19.u7
05	.	04.	51	0	28	28		Delay
51	u.	52.	u2	0	29	19		Clear Line 19
u2	.	u6.	11	5	26	19		Output → 19.u6-u7
11	.	u4.	01	0	20	31		Return to Mark
								Between u3 and 01 or to 01

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D
PROGRAM PROBLEM: CONTROL (MISC.)

Prepared by D. Hassell

Page 6 of 7
Date: 3/13/59
Line 00

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99
u0	u1	u2	u3
u4	u5	u6	

L	P	T or L_k	N	C	S	D	BP	NOTES
		MARK SET BY TYPE-IN SEQUENCE						
u4		20	21	0	00	00		
21	u	22	20	0	19	27		No Thru typing significant digits
20		19	15	0	00	00		Yes last Delay to allow typing of digit
15	w	00	u5	0	00	31		Set Ready
u5	u	u6	26	0	08	19		Line 08 ——▷ Line 19
26		32	32	6	21	31		NC 19.32
				ENTER TO SET FORMAT FOR OP = 32				
93	u	u6	17	1	08	24		T T format ——▷ MQ
				ENTER TO SET FORMAT FOR OP = 34				
97	u	u7	17	1	08	24		T R format ——▷ MQ
				ENTER FROM INDEX REGISTER COMMANDS				
96		97	99	2	26	28		K Store command ——▷ AR
				(MISC.)				
43			0000001					See 00.29 page 00/5
83			7Z00000					See 00.82 page 00/1
41								Available
u3								Mark transfer II storage
07		10	14	0	22	28		Used Sel. Pr. Routine
72		73	01	0	17	31		OP = 00 Useful to skip empty space
								See page 29.13.6

Bendix Computer

MEMORY ALLOCATION CHART

Problem { Intercom 1000 double precision } Page 7 of 7

Date 3/13/59

Prepared by D. Hassell

Check Sum -.7y08v58 Line 00

U7	U6	U5	U4
1863702	wy4z6vz	6ulul13	9415000
U3	U2	U1	U0
	-yu0v753	y703000	y7032w0
99	98	97	96
65653zz	y7032w6	6v11518	6263v5w
95	94	93	92
y1032vv	60032xz	6u11518	5y5yz9w
91	90	89	88
xz633zx	y701016	-5959uvz	y7010x6
87	86	85	84
-5u5v2v9	57590x3	555539z	565773x
83	82	81	80
7z00000	5454014	53533zz	5z5z6vz
79	78	77	76
yv05333	503v2x5	-6v6v6vz	x05111x
75	74	73	72
4x4x0xw	4v4v0w5	85231z	490123z
71	70	69	68
831z0y3	486uvzw	w74w2xu	80553u6
67	66	65	64
454911w	4445vzw	43436vz	-42422vz
63	62	61	60
8055006	40402vx	41412xz	3y3y0zw
59	58	57	56
w346114	vw3x0xx	-3w3w114	-yu21333
55	54	53	52
39392xz	381w39v	373711x	vz4339v
51	50	49	48
34663v3	34342xz	741735z	3232x1x
47	46	45	44
31316vz	3030vzw	813z3u0	2y2y3vz
43	42	41	40
1	2w2w395	y17v0yu	2u2uv3x
39	38	37	36
-463635y	2828114	26383v3	2626yzz
35	34	33	32
v3270z9	2424v9u	231w13z	2222v9x
31	30	29	28
u123z5w	2020v3x	uv3141x	-9y0135v
27	26	25	24
1y1yv9x	-2020uvz	1v1vv3x	8325062
23	22	21	20
171739z	1919yzz	151427v	930z000
19	18	17	16
-161639u	146v13z	111139z	12123zz
15	14	13	12
806901z	1010396	z0y11x	y0yx1x
11	10	09	08
680129z	w0w3vz	v0y11x	u0uv3x
07	06	05	04
8v0y2xw	808yzz	843339w	6060zw
03	02	01	00
8709vxw	41z023		2x23z

FOUR-WORD LINES

LINE	3	2	1	0

FORM BCD 108-15.1

ARITHMETIC

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: 1000 D. P. "CA" "CS" "SU" "AD"

Prepared by H. Lewis & D. Hassell

Page 1 of 7

Date:

Line 04

L	P	T or LK	N	C	S	D	SP	NOTES
								Entry for all arithmetic "OP" codes
.66		.67.	68	.0	.31	.27		Odd address?
								If "YES", see 04.69 below
.68		.71.	72	.0	.08	.23		R.C. ⟶ 23.03
.72		.74.	74	.0	.31	.31		Interpret Command
AR		xx	OP + 36	4	xx	25		
.69		.69.	u4	.0	.17	.31		Ring Bell. Enter if address odd.
.u4		.u4.	26	.0	.17	.31		Ring Bell. Overflow
.26		.26.	26	.0	.28	.31		Wait for Ready
.27	u.	28.	28	.0	.06	.19		Error Loop ⟶ 19
.28		.31.	31	.6	.21	.31		N.C. 06.31 Page 06/4
								Enter "Clear and Subtract"
.76		.77.	78	.0	.20	.24		Reverse Sign OP = 40
								Enter "Clear and Add"
.78		.80.	u5	.4	.25	.21		Operand ⟶ A OP = 42
.u5		.u7.	u7	.7	.21	.31		Normal Exit
								Enter "Subtract"
.77		.78.	79	.0	.20	.24		Reverse sign OP = 41
								Enter "Add"
.79		.80.	82	.0	.30	.28		E_a ⟶ AR OP = 43
.82		.86.	92	.4	.25	.21		$(ID_{0,1})$ = X ⟶ 21.02-03
.92		.94.	98	.3	.30	.29		E_x ⟶ AR
.98		.01.	01	.0	.22	.31		$E_x > E_a$?
								If "YES", see 04.02 Page 04/03

FORM BCD 108-17.8

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: 1000 D. P. Add and Subtract (cont.)

Prepared by H. Lewis & D. Hassell

Page 2 of 7

Date: 10/30/58

Line 04

L	P	T or LK	N	C	S	D	BP	NOTES
.01		.02.	04	.4	.21	.25		$\chi \longrightarrow ID_{0,1}$
.04		.05.	12	.3	.28	.28		Negate (AR) = $(E_a - E_x)$ -
.12		.14.	18	.0	.31	.25		$E_x \longrightarrow ID_0$
.18		.19.	20	.0	.08	.29		$[0000030] \longrightarrow$ AR
.20		.22.	22	.0	.22	.31		$(A - \chi) > 2^{-48}$
								If "YES", see 04.23 Page 04/3
.22		.23.	25	.3	.08	.29		$[0000031] \xrightarrow{-}$ AR (No)
.25		.98.	13	.0	.26	.31		Denormalize controlled by AR
.13		.14.	16	.4	.25	.21		Denormalize X X' $\longrightarrow$ 21.02-03
.16		.20.	24	.4	.31	.25		$M_a \longrightarrow ID_{0,1}$
.24		.28.	29	.0	.30	.28		$E_a \longrightarrow$ AR
.29		.02.	32	.0	.26	.31		$M_a \cdot 2^{-1}$ & $1 \cdot 2^{-28} \xrightarrow{\pm}$ AR
.32		.34.	36	.4	.25	.26		$M_a \cdot 2^{-1} \longrightarrow PN_{0,1}$.
.36		.38.	40	.4	.26	.26		PN' $\longrightarrow$ PN
.40		.42.	45	.5	.21	.30		$\chi' \xrightarrow{+}$ PN
.45		.48.	50	.5	.26	.21		$\chi' + \Lambda = \Sigma \longrightarrow$ 21.00-01
.50		.52.	54	.6	.31	.27		Denormalized $\Sigma = 0$?
.54		.60.	15	.0	.07	.21		Floating point zero $\longrightarrow$ A (Yes)
.55		.56.	58	.4	.21	.24		$\Sigma \longrightarrow MQ_{0,1}$ (No)
.58		.60.	61	.0	.21	.25		Sign $\longrightarrow$ IP
.61		.62.	63	.3	.28	.28		Larger E + $1 \cdot 2^{-28}$ = (AR) $\xrightarrow{-}$ AR

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: 1000 D. P. Normalize & Multiply

Prepared by H. Lewis & D. Hassell

Page 3 of 7

Date: 10/30/58

Line 04

L	P	T or LK	N	C	S	D	BP	NOTES
								Enter "Normalize" Routine
.63	.	u0.	85	.0	.27	.31		Normalize controlled by (AR)
.85	.	86.	87	.1	.28	.28		(AR) ⟶ AR
.87	.	88.	91	.4	.24	.21		Normalized $\Sigma = M_\Sigma$ ⟶ 21.00-01
.91	.	96.	96	.0	.22	.31		Underflow?
								If "NO", see 04.97 below
.96	.	97.	5[illegible]	.0	.29	.21		Floating Zero ⟶ A (Yes)
.97	.	98.	99	.2	.20	.29		[zzzzz00] $\xrightarrow{+}$ AR (No)
.99	.	u0.	u2	.0	.27	.21		$M_\Sigma + E_\Sigma$ ⟶ 21.00-01 = A
.u2	.	u4.	u4	.0	.22	.31		Overflow? if "No" see normal exit
								If "YES", see Error Loop
								Enter when $(A-X) > 2^{-48}$
.23	.	u0.	u5	.4	.21	.25		Larger Number ⟶ A
								Add (Cont.)
.02	.	06.	08	.5	.21	.24		
.08	.	12.	14	.4	.31	.25		Interchange X for A
.14	.	16.	18	.5	.24	.21		
								Enter "Multiply"
.80	.	81.	88	.0	.21	.27		A = 0? OP = 44
.88	.	92.	u5	.4	.21	.25		21.00-01 ⟶ $ID_{0,1}$ & Exit (Yes)
.89	.	91.	93	.0	.25	.27		X = 0? (No)

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: 1000 D. P. Multiply (cont.) & Divide

Prepared by H. Lewis & D. Hassell

Page 4 of 7

Date: 10/30/58

Line 04

L	P	T or LK	N	C	S	D	CP	NOTES
.93		.96.	u5	.4	.25	.21		Zero = $ID_{0,1}$ → A Exit (Yes)
.94		.95.	u3	.0	.08	.28		[0000080] → AR (No)
.u3		.u4.	05	.4	.31	.24		M_A → $MQ_{0,1}$
.05		.06.	10	.0	.25	.21		χ → 21.02-03
.10		.14.	15	.3	.30	.29		E_χ → AR
.15		.98.	06	.0	.24	.31		Multiply χ · A
.06		.08.	49	.4	.26	.24		χ · A = P → MQ
.49		.52.	63	.3	.30	.29		E_A → AR N.C. 04.63 Page 04/3
								Enter Divide
.84		.88.	95	.0	.00	.00		Skip OP = 48
.95		.97.	u0	.0	.25	.27		χ = 0?
.u0		.u0.	u4	.0	.17	.31		See Error Loop (Yes)
.u1		.u5.	u6	.0	.21	.27		A = 0? (No)
.u6		.01.	54	.0	.29	.21		Floating Zero → A Exit (Yes)
.u7	u.	.22.	23	.4	.21	.26		A → PN (No)
.03		.06.	07	.0	.25	.21		Sign & E_χ → 21.02
.07		.08.	09	.3	.08	.28		[0000081] → AR
.09		.10.	11	.0	.30	.29		E_χ → AR
.11		.12.	19	.5	.29	.24		Clear MQ
.19		.v4.	49	.5	.25	.31		Divide

FORM BCD 18-17.0

Bendix Computer — Los Angeles 45, California

Page 5 of 7

G-15D

Prepared by H. Lewis & D. Hassell — Date: 10/30/58

PROGRAM PROBLEM: 1000 D. P. /CA/, DI^{-1}, four digit output conversion. Line 04

L	P	T or L_K	N	C	S	D	BP	NOTES
								Enter "Inverse Divide"
.83		.84.	86	.4	25	.22		Interchange χ & A — OP = 47
.86		.88.	90	.4	21	.25		
.90		.92.	95	.4	22	.21		see preceding page
								Enter Clear and Add Absolute Value
.81		.84.	88	.5	25	.21		/ID/ ⟶ A and Exit
								Enter four-digit Binary-Decimal Conversion
.46		.47.	48	.2	21	.28		Output ⟶ AR
.48		.51.	53	.0	23	.31		Clear two-word registers
.53		.57.	59	.3	06	.29		$u \cdot 2^{-16}$ $\xrightarrow{-}$ AR
.59		.64.	64	.0	22	.31		WD $<$ $u \cdot 2^{-16}$?
.64		.65.	53	.1	06	.30		$1 \cdot 2^{-4}$ $\xrightarrow{+}$ PN_1 — No
.65		.57.	60	.1	06	.29		$u \cdot 2^{-16}$ $\xrightarrow{+}$ AR — Yes
.60		.61.	62	.1	26	.29		PN_1 $\xrightarrow{+}$ AR
.62	u.	.65.	70	.0	25	.27		through converting?
								If "YES", see 04.71 Page 04/6
.70		.73.	75	.0	28	.25		Units and tens digit ⟶ ID — No
.75		.16.	41	.0	24	.31		Shift eight bits
.41	u.	.57.	38	.0	28	.29		Replace WD with CH
.38		.52.	52	.0	29	.31		Reset Overflow
								If not set, see 04.53 above
.52		.54.	53	.0	04	.25		Non-Zero ⟶ ID_0

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D

Prepared by D. Hassell

Page 6 of 7

Date: 3/14/59

PROGRAM PROBLEM: FOUR-DIGIT OUTPUT (Cont.): LEADER PUNCH

Line 04

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99
u0	u1	u2	u3
u4	u5	u6	

L	P	T or L_k	N	C	S	D	BP	NOTES
71		73	74	0	25	29		Units & Tens. + L AR+
74		91	00	0	08	23		Four-digit format D 23.03
00		03	39	0	23	03		Set format
39	u	44	44	0	08	23		Type-out Routine - L 23.00-03
44		49	49	7	20	31		N.C. After typing return to 08.41
					PUNCH (Cont.)			Set format & Pickup Output
30		30	30	0	28	31		Wait for ready
31	u	32	47	0	29	19		
47	u	52	73	0	07	23		Format D 23
73		00	21	4	23	02		Set format L 02.00-03
21		02	37	4	23	02		
37		39	67	2	30	29		(Address) - + L AR+
67		97	97	5	21	31		NC See 08.07 page 08/3
					ENTER TO PUNCH			LEADER
35		07	33	2	08	28		$(8w \cdot 2^{-28})$ L AR
33		48	51	7	28	28		$(1 \cdot 2^{-28})$ - D AR+
51		53	56	0	28	27		
								$\neq 0$
57	w	00	33	0	10	31		
56		u3	u3	5	21	31		NC 08.u3 page 08/2
34		01	01	0	21	31		Used by Londer II
		EMPTY SPACES						17, 42, 43

FORM BCD 108-5.0

MEMORY ALLOCATION CHART

Problem { Intercom 1000 double precision Page ___ of ___

Date ___

Prepared by D. Hassell

Check Sum -.1347192 Line 04

U7	U6	U5	U4
-16172vu	81363v5	-6v6vyvz	681u23z
U3	U2	U1	U0
-6u053z8	68682xz	y96u2vv	646823z
99	98	97	96
6566375	1012xz	6363u9x	62363v5
95	94	93	92
y16433v	606711w	-y069335	xy62zxx
91	90	89	88
60602xz	-xw5z2x5	xv5x33v	-xw692v9
87	86	85	84
-5u5v315	-x85u2v9	575779w	x85z000
83	82	81	80
-5656336	-x65w335	-x458735	52582vy
79	78	77	76
51523xw	-x069335	4z4z298	4y4y298
75	74	73	72
102931z	xv00117	-80152y2	4u4u3zz
71	70	69	68
w94u33x	w94v399	456823z	w748117
67	66	65	64
-61616vz	44443zv	v93w4xx	42354xy
63	62	61	60
645537z	414633v	3z3zz9w	3y3y75x
59	58	57	56
40402xz	vw3x2v9	802115z	-67676vz
55	54	53	52
-3u3u2v8	vw690z5	v93vwxx	v635099
51	50	49	48
v53839v	-v436vzv	v43zzxx	33352zz
47	46	45	44
34490z7	3030uvw	-v032755	-3131y9z
43	42	41	40
		392639x	-uu2x6vy
39	38	37	36
2w2w117	34343vz	u743vxx	-u62835u
35	34	33	32
872191w	1012vz	-v033z9w	-u22433u
31	30	29	28
202z3v3	1y1y39z	22035z	-1z1zuvz
27	26	25	24
1w1w0x3	1u1u39z	620x35z	9w1x3xw
23	22	21	20
-y4692v9	1819x1x	-82252y2	16162xz
19	18	17	16
-723173z	141411x	-721v73z	-94183z9
15	14	13	12
620631z	-9012715	-1010335	8y123z9
11	10	09	08
- y137v8	8y0zzxx	v0v3xx	-8w0y3z9
07	06	05	04
909x1w	-8831358	70u335	60wz9w
03	02	01	00
8607335	-86086v8	- 4042v9	83272y3

FOUR-WORD LINES

LINE	3	2	1	0

FORM BCD 108-16.1

LOADER II

Bendix Computer — Los Angeles 45, California

G-15D
PROGRAM PROBLEM: Loader II Intercom 1000 D.F.

Prepared by D. Hansell

Page 1 of 6
Date:
Line 04

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99
U0	U1	U2	U3
U4	U5	U6	

L	P	T or Lk	N	C	S	D	BP	NOTES
								"Manual Entry" <P>
00	U	01	01	0	19	04		Loader II ——▷Line 04
01		U6	U6	4	21	31		N C 04.U6
U6	U	01	02	2	04	04		Precess 04.U7 ——▷04.00
U7		02	02	0	02	00		
02	U	07	07	0	29	23		0 ——▷23
07		07	07	0	28	31		Wait for ready
08		12	13	0	04	23		Flag ——▷23.00
12								0008000
13		15	15	0	12	31		Gate input
15		15	15	0	28	31		Wait for ready
16		20	21	0	23	28		Input ——▷AR
21		23	24	0	28	21		(AR) ——▷21.03
24		21	26	0	23	27		Input ≠ 5 digits
								Yes
26		35	39	0	04	20		
35								-000000Z
39		03	61	0	31	28		(31.03) ——▷AR
61		60	70	0	28	27		Input = 0?
								Return to Manual
70		72	92	0	06	31		Yes Reverse tape
92		92	92	0	28	31		Wait
93		94	88	0	06	31		Reverse tape
88		88	88	0	28	31		Wait
89		98	98	0	15	31		Read (Line 04)

FORM BCD 108-5.0

Bendix Computer — Los Angeles 45, California

G-15D
PROGRAM PROBLEM: Loader II

Prepared by D. Hassell

Page 2 of 6
Date:
Line 04

0	1	2	3	L	P	T or L_k	N	C	S	D	BP	NOTES
4	5	6	7									
8	9	10	11	98		98	98	0	28	31		Wait for ready
12	13	14	15	99		U2	U0	0	29	31		Reset Overflow
16	17	18	19									
20	21	22	23	U0	U	W	34	0	19	04		Restore line 04
24	25	26	27	U1	U	U2	34	0	19	04		N C .00.01
28	29	30	31									Input ≠ 0
32	33	34	35	71		69	74	0	22	31		Input ≠ D-
36	37	38	39									Yes
40	41	42	43	74		76	73	3	04	29		
44	45	46	47	76								0000001
48	49	50	51	73		74	78	0	28	27		Input = 1?
52	53	54	55									Yes
56	57	58	59	78		80	80	0	15	31		Read
60	61	62	63	80		80	80	0	28	31		Wait
64	65	66	67	81		U7	05	1	19	28		$L\Sigma \xrightarrow{+} AR$
68	69	70	71	05	U	06	06	3	19	29		Σ 19 $\xrightarrow{-}$ AR+
72	73	74	75	06		07	10	0	28	27		Read OK
76	77	78	79									Yes
80	81	82	83	10		00	00	6	21	31		N C 19.00
84	85	86	87									Bad Read No
88	89	90	91	11		11	14	0	17	31		Ring Bell
92	93	94	95	14		16	77	0	06	31		Reverse
96	97	98	99	77		77	77	0	28	31		See 78 above
U0	U1	U2	U3									
U4	U5	U6										

FORM

Bendix Computer — Los Angeles 45, California

Page 3 of 6

Prepared by D. Hassell

Date:

G-15D

PROGRAM PROBLEM: Loader II

Line 04

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99
U0	U1	U2	U3
U4	U5	U6	

L	P	T or L_k	N	C	S	D	BP	NOTES
79		76	82	3	04	29		
82		83	84	0	28	27		D = 2
								Yes
84		86	91	0	04	21		
86	U	00	20	0	29	08		
91		95	97	0	04	28		No. of lines to be cleared
95	U	00	00	0	00	10		
97		99	72	0	28	21		(AR) ⟶ 21.03
76		74	94	0	21	29		(21.02) ⟶ AR
94		99	99	0	31	31		N C AR
AR	U	00	20	0	29	08+N		
20		23	25	0	21	28		
25		76	87	3	04	29		
87		95	96	0	28	29		Through clearing?
								No See 97 above
96		00	00	0	00	00		Go to 0
								Enter When D = 3
85		90	91	0	04	21		
90	U	00	20	0	29	08		
								Enter when input = D-
75		77	77	0	15	31		Read tape
								See 04.77 above

FORM

Bendix Computer — Los Angeles 45, California

Page 4 of 6.

Prepared by D. Hassell — Date: _____

G-15D

PROGRAM PROBLEM: Loader II — Line 04

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99
U0	U1	U2	U3
U4	U5	U6	

L	P	T_{Pr} L_k	N	C	S	D	BP	NOTES
27		31	32	0	04	20		
31								00ZZZZZ
32		35	36	0	31	28		Input ——▷AR
36	U	37	37	0	04	19		Line 04 ——▷line 19
37	W	59	04	6	21	31		N C 04.04 mark 59
04	U	58	0 5	0	08	19		Command input conv. ——▷19
		Convert (FCHWD) ——▷						(WDOKCH)
59	U	60	18	0	04	19		Line 04——▷19
18	U	43	19	0	00	19		D/B conv.——▷19
								Convert input to binary
44		47	48	3	25	29		ADDR - CH ——▷AR
48		49	50	3	25	29		
50		51	52	0	28	21		WD ——▷21.03
52		54	54	4	21	31		N C 04.54
54		55	56	2	31	28		ON ——▷AR
56		57	58	0	04	29		1 ——▷AR
57	U	00	01	0	00	00		
58		62	63	0	28	21		AR ——▷21.06
63		65	65	0	15	31		Read
65		57	66	3	04	29		# -1 ——▷AR+
66		66	66	0	28	31		Wait
67		68	62	0	28	27		AR = 07
								Yes
62		U7	09	1	19	28		$-\Sigma$ ——▷AR

FORM BCD

Bendix Computer — Los Angeles 45, California

G-15D
PROGRAM PROBLEM: Loader II

Prepared by D. Hassell

Page 5 of 6
Date:
Line 04

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99
U0	U1	U2	U3
U4	U5	U6	

L	P	T or L_R	N	C	S	D	BP	NOTES
09	U	10	22	3	19	29		Σ 19 ——=——▷AR+
22		23	29	0	28	27		Read correctly?
								Yes
29		08	38	0	24	31		Multi (shift)
38		39	40	0	30	28		(WD ——▷AR
40		41	42	0	04	29		
41	U	20	65	0	19	00		
42		43	U5	0	25	29		CH——+——▷AR+
U5		U7	U7	0	31	31		
AR	U	20+ WD	65	0	19	CH		
								Enter Reverse Loop
45		46	47	0	21	28		⚡=
47		49	68	0	06	31		◁
68		68	68	0	28	31		
69		57	60	3	04	29		
60		61	46	0	28	27		⚡
								When AR = 0 see page
46		00	00	0	00	00		Go to 00
55								}
64								} Balancer
83								}
								Read incorrectly!
30		29	45	0	17	31		

Empty spaces 03, 17, 19, 23, 28, 33, 34, 43, 49, 51, 53, U2, U3, U4

FORM 800

MEMORY ALLOCATION CHART

Bendix Computer

Problem: Intercom 1000 double precision Page 6 of 6

Date

Prepared by D. Hassell

Check Sum -.1000601 Line 04

U7 8202000	U6 102884	U5 6v6v3zz	U4
U3	U2	U1 6622264	U0 6522264
99 66643vz	98 626239z	97 y348395	96 8000000
95 u	94 63633zz	93 5y580xz	92 5w5w39z
91 xz6109w	90 64143u8	89 62621zz	88 585839z
87 xz6039v	86 143u8	85 xu5v095	84 x65v095
83 -79v78yu	82 545439v	81 yv0567w	80 505039z
79 ww52w9x	78 50501zz	77 4x4x39z	76 1
75 4x4x1zz	74 ww49w9x	73 4v4y39v	72 wu5y2vx
71 454u2xz	70 485w0xz	69 v93ww9x	68 444439z
67 453y39v	66 424239z	65 v942w9x	64 z000000
63 41411zz	62 yv0967w	61 vw4639v	60 3y2y39v
59 3w12093	58 vy3z395	57 1000	56 3u3u09x
55 -yzzz9zz	54 3838vzw	53	52 -36362vz
51	50 3434395	49	48 3232z3x
47 31440xz	46 8000000	45 2z2z2vw	44 uz30z3x
43	42 2w6933x	41 1441260	40 2u2u09x
39 833x3zw	38 28283xw	37 -vv04uvz	36 2525093
35 - z	34	33	32 u3243zw
31 zzzzz	30 1x2x23z	29 82631z	28
27 9z20094	26 u327094	25 ww57w9x	24 951u2zv
23	22 181x39v	21 9718395	20 97192vw
19	18 2v13013	17	16 94152zw
15 z0z39z	14 104x0xz	13 z0z19z	12 8000
11 v0y23z	10 - uvz	09 u16y7x	08 8w0x097
07 70739z	06 80u39v	05 606y7x	04 3u05113
03	02 7073v7	01 -6u6u2vz	00 101264

FOUR-WORD LINES

LINE	3	2	1	0

FORM BCD 108-15.1

SELECTIVE PRINT SUBROUTINE

Bendix Computer — Los Angeles 45, California

G-15D

PROGRAM PROBLEM: Selective Print Subroutine

Prepared by D. Hassell

Page 1 of 6.

Date:

Line 05

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99
u0	u1	u2	u3
u4	u5	u6	

L	P	Tr Lk	N	C	S	D	BP	NOTES
								Enter to Load Sel. Pr.
00	U	01	01	0	19	05		(Line 19) ——▷ line 05
01		75	75	5	21	31		N C 05.75
75	U	78	78	1	05	25		
76		00	00	5	21	31		
77		02	06	5	21	31		
78		61	79	1	25	06		Patch to form OP = 61
79		62	80	1	25	06		Patch to form OP = 62
80		82	81	4	05	25		
82								
83		03	04	0	22	26	-	
81		00	84	4	25	05		Replace 00 - 01
84		86	85	0	06	31		Reverse tape 1 block
85		85	85	0	28	31		
86		00	00	4	21	31		Return to loader I
								Enter from OP = 61
02		07	08	0	05	23		
07		21	21	5	21	31		Set patch^2 ——▷ 05.19
08		19	20	0	23	00		
20	U	26	26	0	29	23		(0) ——▷ 23
26		28	28	0	12	31		Gate input
28		28	28	0	28	31		Wait for ready
29	U	74	74	0	05	19		Line 05 ——▷ 19
74	U	58	58	0	08	19		(KOPCHWD) Conversion ——▷ 19

FORM RCD

Bendix Computer — Los Angeles 45, California

G-15 D

PROGRAM PROBLEM: Selective Print Subroutine

Prepared by D. Hassell

Page 2 of 6

Date:

Line 05

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99
U0	U1	U2	U3
U4	U5	U6	

L	P	T or Lk	N	C	S	D	BP	NOTES
58	W	63	05	6	21	31		N C 08.05
								Convert Code 2
63		67	68	0	28	05		Code 2 ——▷ 05.67
67								Code 2
68		69	70	0	23	28		}
70		72	73	0	28	23		} Set up code 1 for conv.
73	W	60	05	6	21	31		}
60		71	37	0	28	05		
71								Code 1
								N C 08.37 NC 00.01
								Enter from Patch 1
21		22	24	4	28	21		KWDOPCH ——▷ 21.02 - 03
24		66	04	4	05	20		Code 2 ——▷ 20.03
66								000ZX00
04		07	09	0	31	28		Command Z ——▷ AR
09		71	10	3	05	29		Code 1 ——▷ AR
10		11	12	0	28	27		Command = Code?
								Yes
12		14	36	0	31	27		OP Code = 02?
								Yes
36		35	49	0	16	31		Halt
								≠0 No List
13		42	50	0	05	00		Replace patch 2
42		43	44	0	28	21		Patch 2

FORM BCD 108-5.0

Los Angeles 45, California

Page 3 of 6.

G-15 D

Prepared by D. Hassell

Date:

PROGRAM PROBLEM: Selective Print Subroutine

Line 05

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99
U0	U1	U2	U3
U4	U5	U6	

L	P	T or L_k	N	C	S	D	BP	NOTES
50		51	52	0	21	28		KWDOPCH ——▷AR
52		55	56	0	22	05		CC
55								CC
56		59	64	0	21	05		KWDOPCH ——▷05.59
59								KWDOPCH
64		66	16	4	28	26		KWDOPCH ——▷PNo,1
16		22	22	0	21	31		N C 00.22
								Enter to set for list
37		38	39	0	05	28		Patch 3 AR
38		69	69	5	21	31		Patch 3
39		42	50	0	28	00		Patch 3 ——▷00.42
								See 05.50 above
								Enter from patch 3
69		11	17	2	28	05		Store converted com.
11								Converted command
17		22	23	0	05	23		Set
22		27	27	5	21	31		Patch #4 Patch #4
23		14	25	0	23	00		——▷00.04
25		27	30	0	28	21		(AR) ——▷21.03
30		44	44	0	21	31		N C 00.44
								Enter from Patch #4
27		31	32	0	28	22		
32		33	34	0	05	28		Patch #5 ——▷AR
33		31	31	5	21	31		Patch #5
34		01	15	0	28	00		AR ——▷00.01

FORM BCD 108-5.0

Los Angeles 45, California

Page 4 of 6.

Prepared by D. Hassell

Date:

G-15D

PROGRAM PROBLEM: Selective Print Subroutine

Line 05

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99
u0	u1	u2	u3
u4	u5	u6	

L	P	T or L_k	N	C	S	D	BP	NOTES
15		55	57	0	05	21		CC ⟶ 21.03
57		59	59	0	21	31		Exit to type loc.
								Return from typing loc.
31		03	35	0	05	03		
03								1Z80400 format
35		59	72	0	05	28		KWDOPCH ⟶ AR
72		74	61	0	08	31		Type AR = KOP
61		61	61	0	28	31		Wait for ready
62		11	40	0	05	21		Converted com ⟶ 21.5
40		41	43	0	05	28		Patch #6 ⟶ AL
41		45	45	5	21	31		Patch #6
43		01	44	0	28	00		AR ⟶ 00.01
44		47	06	0	31	28		
06		79	79	1	21	31		Type ADDR
								Return after typing ADDR
45		48	51	4	21	25		A ⟶ ID
51		53	54	0	05	28		Patch #7 ⟶ AR
53	W	07	05	5	21	31		Patch #7
54		01	46	0	05	00		Restore patch #7
46		34	34	1	21	31		
05		01	48	0	05	00		
01		03	04	0	22	26	-	Previously ⟶ 05.83
65		67	66	0	20	31		Return to mark ⟶ 00
								Return to 00.07

FORM BCD 108-5.0

Bendix Computer — Los Angeles 45, California

Page 5 of 6.

G-15D

Prepared by D. Hassell

Date:

PROGRAM PROBLEM: Selective Print Subroutine

Line 05

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	9'
U0	U1	U2	U3
U4	U5	U6	

L	P	T or L_k	N	C	S	D	BP	NOTES
82	W	01	18	5	21	31		
18		19	49	0	05	00		Replace patch
19		20	22	4	28	26		
49		42	48	0	05	00		
42		43	44	0	28	21		
48		14	65	0	05	00		
14		15	16	0	28	22		
								Used to return to Loader II
87		87	87	0	28	31		Wait
88		90	90	0	06	31		Reverse
90		90	90	0	28	31		Wait
91		93	92	0	15	31		Read
92		92	92	0	28	31		Wait
93	U	94	94	0	19	04		
U6								Balancer
U7								L Σ

FORM BCD

MEMORY ALLOCATION CHART

Problem { Intercom 1000 double precision Page 6 of 6
Date

Prepared by D. Hassell Check Sum .1000600 Line 05

U7	U6	U5	U4
1000600	-10133zw		
U3	U2	U1	U0
99	98	97	96
95	94	93	92
		5y5y264	5w5w39z
91	90	89	88
5x5w1zz	5u5u39z		5u5u0xz
87	86	85	84
575739z	- 2vz	555539z	56550xz
83	82	81	80
83842xu	-81126vz	-8054325	-x2510v9
79	78	77	76
vy50726	vx4z726	- 2026vz	- 6vz
75	74	73	72
4y4y4v9	3v3u113	-vw05uvz	4u3x11z
71	70	69	68
	w849397	8v11v85	46462zw
67	66	65	64
	zx00	434229z	-w2103u
63	62	61	60
w344385	8v280v5	3x3x39z	w725385
59	58	57	56
	-vz05uvz	3v3v2vz	vv402u5
55	54	53	52
	812y380	-87056vz	v7382w5
51	50	49	48
v5360vw	34342vw	uu300u0	8y410u0
47	46	45	44
	22226vz	-v0332v9	uz063zw
43	42	41	40
812w380	2w2w395	-2x2x6vz	2u2v0vw
39	38	37	36
uu32380	-45456vz	27270vw	233121z
35	34	33	32
vv480vw	810z380	-1z1z6vz	22220vw
31	30	29	28
83230u3	2w2w2vz	4u4u0v3	1w1w39z
27	26	25	24
9z20396	1w1w19z	9v1y395	-w2040v4
23	22	21	20
8y192y0	-1v1v6vz	-1818395	1u1u3v7
19	18	17	16
-161639u	14310u0	96170v7	16162vz
15	14	13	12
v7390v5	1010396	uu320u0	8y243zv
11	10	09	08
	w0w39v	w70uwvx	93142y0
07	06	05	04
-15156vz	4z4z6vz	81300u0	87093zw
03	02	01	00
1z80400	87080v7	-4v4v6yz	101265

FOUR-WORD LINES

LINE	3	2	1	0

FORM BCD 108-15.1

FORMATER

Bendix Computer — Los Angeles 45, California

Page 1 of 5

G-15D

Prepared by D. Hassell

Date:

PROGRAM PROBLEM: FORMATER (INTERCOM 1000 D.P.)

Line

L	P	T or LK	N	C	S	D	BP	NOTES
								ENTER FROM LOADER II
00		03	09	2	31	28		
09	u	22	23	0	28	29		Shift 12 bits
23		25	25	0	31	31		
AR		00	0D	0	00	00		
								Enter when D=1
01	u	42	70	0	19	23		
38								6200000
39								8000006
40								2200000
41								000000w
								Enter when D=2
02	u	46	70	0	19	23		
42								0w40000
43								8000006
44								0440000
45								0000010
								Enter when D=3
03	u	50	70	0	19	23		
46								0188000
47								8000006
48								0088000
49								0000014

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

G-15D

Prepared by D. Hassell

Page 2 of 5

Date:

PROGRAM PROBLEM: FORMATER (INTERCOM 1000 D.P.)

Line

L	P	T or LK	N	C	S	D	BP	NOTES
								ENTER WHEN D = 4
04	u	54	70	0	19	23		Format & K 23
50								0031000
51								8000006
52								0011000
53								0000018
								ENTER WHEN D.= 5
05	u	58	70	0	19	23		Format & K 23
54								0006200
55								8000006
56								0002200
57								000001w
								ENTER WHEN D = 6
06	u	62	70	0	19	23		Format & K 23
58								0000w40
59								8000006
60								0000440
61								0000020
								ENTER WHEN D = 7
07	u	66	70	0	19	23		Format & D 23
62								0000188
63								8000006
64								0000088
65								0000024

FORM BCD 108-17.0

Los Angeles 45, California

Page 3 of 5

G-15D

PROGRAM PROBLEM: FORMATER (INTERCOM 1000 D.P.)

Prepared by D. Hassell

Date:

Line

L	P	T or LK	N	C	S	D	BP	NOTES
								ENTER WHEN D = 8
08	u	70	70	0	19	23		
66								y200000
67								-8000007
68								u200000
69								0000008
70		06	71	4	23	07		Set format
71		08	72	4	23	07		
72		73	74	0	19	28		Return Command AR
73		10	13	6	20	31		
74		08	08	0	31	31		Obey AR
AR		10	13	6	20	31		Return to Mark 13 or 13
								RETURN TO C.C.
13		01	01	0	21	31		N.C. 00.01
								ENTER TO REWIND LOADER II and RTM
11		81	82	0	00	00		Do nothing
82	u	00	12	0	19	05		19.83-u7 05.83-u7
12		83	83	5	21	31		N.C. 05.83
83		85	85	0	06	31		Rewind Formater
85		85	85	0	28	31		Wait
86		88	88	0	06	31		Rewind Sel. Pr.
88		88	88	0	28	31		Wait
89		70	70	4	21	31		N.C. Word 70 of Loader II

FORM BCD 108-17.0

Bendix Computer — Los Angeles 45, California

Page 4 of 5

G-15D

Prepared by D. Hassell

Date:

PROGRAM PROBLEM: FORMATER CHANGES FOR OP = 02 ≠ 08

Line

L	P	T or LK	N	C	S	D	SP	NOTES
01	u	42	70	0	05	23		
02	u	46	70	0	05	23		
03	u	50	70	0	05	23		
04	u	54	70	0	05	23		
05	u	58	70	0	05	23		
06	u	62	70	0	05	23		
07	u	66	70	0	05	23		
08	u	70	70	0	05	23		
72		73	74	0	05	28		
73		10	13	5	20	31		
11		81	83	5	21	31		

FORM BCD 104-17.0

Bendix Computer

MEMORY ALLOCATION CHART

Problem _____ Page 5 of 5

Date _____

Prepared by _____

Check Sum 1001601 Line _____

U7	U6	U5	U4
1001601	-5x1w963		
U3	U2	U1	U0
99	98	97	96
95	94	93	92
91	90	89 -46462vz	88 585839z
87	86 58580xz	85 555539z	84
83 55550xz	82 w265	81	80
79	78	77	76
75	74 8083zz	73 - u0xu9z	72 4u4u27w
71 -88482y7	70 -86472y7	69 8	68 u200000
67 -8000007	66 y200000	65 24	64 88
63 8000006	62 188	61 20	60 440
59 8000006	58 w40	57 1w	56 2200
55 8000006	54 6200	53 18	52 11000
51 8000006	50 31000	49 14	48 88000
47 8000006	46 188000	45 10	44 440000
43 8000006	42 w40000	41 w	40 2200000
39 8000006	38 6200000	37	36
35	34	33	32
31	30	29	28
27	26	25	24
23 19193zz	22	21	20
19	18	17	16
15	14	13 1012vz	12 -53536vz
11 x152000	10	09 161739x	08 4646277
07 4246277	06 3y46277	05 3u46277	04 3646277
03 3246277	02 2y46277	01 2u46277	00 8309vzw

FOUR-WORD LINES

LINE	3	2	1	0

FORM BCD 108-15.1