SYLVANIA 9400 SYSTEM (SERIES No. 1)

PROGRAMMING EXERCISES

VOLUME I

Issued by the Educational Unit

of the

Product Support Organization

31 May 1960

SYLVANIA ELECTRONIC SYSTEMS
A Division of Sylvania Electric Products Inc.
PRODUCT SUPPORT ORGANIZATION
1210 V. F. W. Parkway, West Roxbury, Mass.

TABLE OF CONTENTS

Group		Page
1	Addition/Subtraction/Overflow	I-1
2	Fixed Point Multiplication/Division/Shifting/Overflow	I-1
3	Scaling	I-3
4	Looping/Comparison/Add Beta	I-4b
5	Sensing/Overflow/Review Concepts	I-7b
6	Indexing	I-9b
7	Repeat Combinations	I-10b
8	Floating Point Arithmetic Operations	I-12b
9	Cycling/Masking/Logical Operations	I-13
10	Subroutines/Floating Point/Double Precision	I-15
11	Input-Output/Magnetic Tape	I-16b
12	Input-Output/Paper Tape	I-19b
13	Output/Flexowriter	I-20b
14	Input-Output/Cards	I-20b
15	Output/Printer	I-22
16	Order Sequence Mode Operations	I-23
17	Trapping Mode	I-24
18	Real Time	I-25
19	Console	I-26
20	Review	I-26

INTRODUCTION

The exercises in these manuals are basically designed as a teaching guide for instructional purposes. They are to be used in conjunction with other classroom material and lectures by the instructors. With this in mind, the explanations relating to the uses of the computer operational codes in each problem are limited. The problems used exemplify to the fullest extent the uses and applications of the particular instructions, sometimes even at the expense of efficiency and practicality. It is expected that the student will apply his new found knowledge by rewriting the solutions wherever necessary using his own programming techniques and reaching his own degree of efficiency.

You will find that these manuals are quite easy to use and are conducive to learning. Where the statements of the problems are complex, flow charts are included for ease of understanding. The problems are divided into groups which represent teaching units. Octal notation is used in the first four groups of problems to represent computer memory locations and addressable registers. The beginning of the fifth group of problems has been chosen as the point at which the student is able to think of programming in more abstract terms. Here, symbolic programming is introduced and used throughout the remainder of the problems. All the symbols used in these exercise manuals correspond to those outlined in the programming reference manual and in the instructors' lectures. Just as English students find it easier to analyze literary works in terms of sentences and paragraphs, the programming student will find it easier to analyze programs in terms of logical groupings of lines of computer coding. For this reason you will find that the computer instructions in most of the problem solutions have been logically grouped and set off by means of ruled lines.

Any comments, corrections, or suggested problems should be directed to Mr. William Benson of the Educational Unit Staff.

Approved By: Samuel C. Hanna
Supervisor, Course Development
and Instruction Dept.
Sylvania Educational Unit
Product Support Organization

Date: 31 May 1960

GROUP 1 ADDITION/SUBTRACTION/OVERFLOW

1.1 Compute: $C(500) + C(501) + |C(502)|$

1.2 Compute: $-C(500) + |C(501)| - C(502) - |C(503)|$

1.3 Compute: $-|C(500)| + |C(501) - C(502)| - [C(503) + C(504)]$

Store in location 505 and the Q register.

1.4 Compute: $|C(200)| + C(201) - |C(202)| + C(203) - C(204) + |C(205)|$

Store in location 4000 and the B register

1.5 Compute: $x + (x - .1) - |x| + .2 + |x + .3|$. Set the overflow alarm but do not stop the computer whenever overflow takes place. x is a fraction in location 500. A fraction is entered at the leftmost part of a word.

GROUP 2 FIXED POINT MULTIPLICATION/DIVISION/SHIFTING/OVERFLOW

2.1 Compute: $C(500) \cdot C(501) \cdot C(502)$. Assume all numbers are fractions, and store the product in location 503. See the note in Problem 1.5 concerning fractions.

2.2 Compute: $\dfrac{C(500) \cdot C(501) \cdot C(502)}{C(503) - |C(504)|}$

Set the overflow alarm whenever overflow exists but do not stop the machine at that point. Assume all values are fractions, and that division will take place. Store the answer in location 506.

2.3 Compute: $8\left[\frac{-C(501)\cdot C(502)}{64}\right]\cdot C(503)$ Assume all values are fractions. Eliminate any possibility for overflow. Store Acc_{1-18} in location $(504)_{1-18}$ and Acc_{19-36} in location $(505)_{19-36}$.

Note: Remember that division by powers of two can be accomplished by shifting right and multiplication by powers of two can be accomplished by shifting left.

2.4 Compute: $8\left[\frac{x^2 + 2x^3}{y}\right] + .05$ x is a fraction in location 300 and y is a fraction in location 301. Use the proper beta bit configurations for overflow, so that five different effects may be shown for the five opportunities for overflow in this program. Store answer in location 305. Assume division will take place.

2.5 Compute: $\frac{2x^2 - 3y}{2z}$ where x, y, and z are fractions in locations 500, 501, and 502. Place the result in location 503. Assume that division will take place.

2.6 Compute: $$\frac{-\left|C(500)\cdot C(501)\right| + \frac{C(502)}{C(503)}\cdot\frac{C(504)}{128} - 8C(505)}{C(505)_{1-17} - C(506)_{20-36}}$$

Whenever overflow exists, control it so that no action takes place. Assume all locations contain fractional numbers and also that the denominator is treated as a fractional number. Store the answer in location 510 and the B register. Assume $C(505)_{18-36}$ are equal to zero.

GROUP 3. SCALING

3.1 Locate the binary point and determine how many leading zeros will be in the result of the following problems.

a. Multiply A x B where:
the binary point in A is between bit positions 21 and 22 and has 17 leading zeros;
the binary point in B is between bit positions 27 and 28 and has 20 leading zeros.

b. Divide C by D where:
the binary point in C is between bit positions 24 and 25;
the binary point in D is between bit positions 29 and 30.

3.2

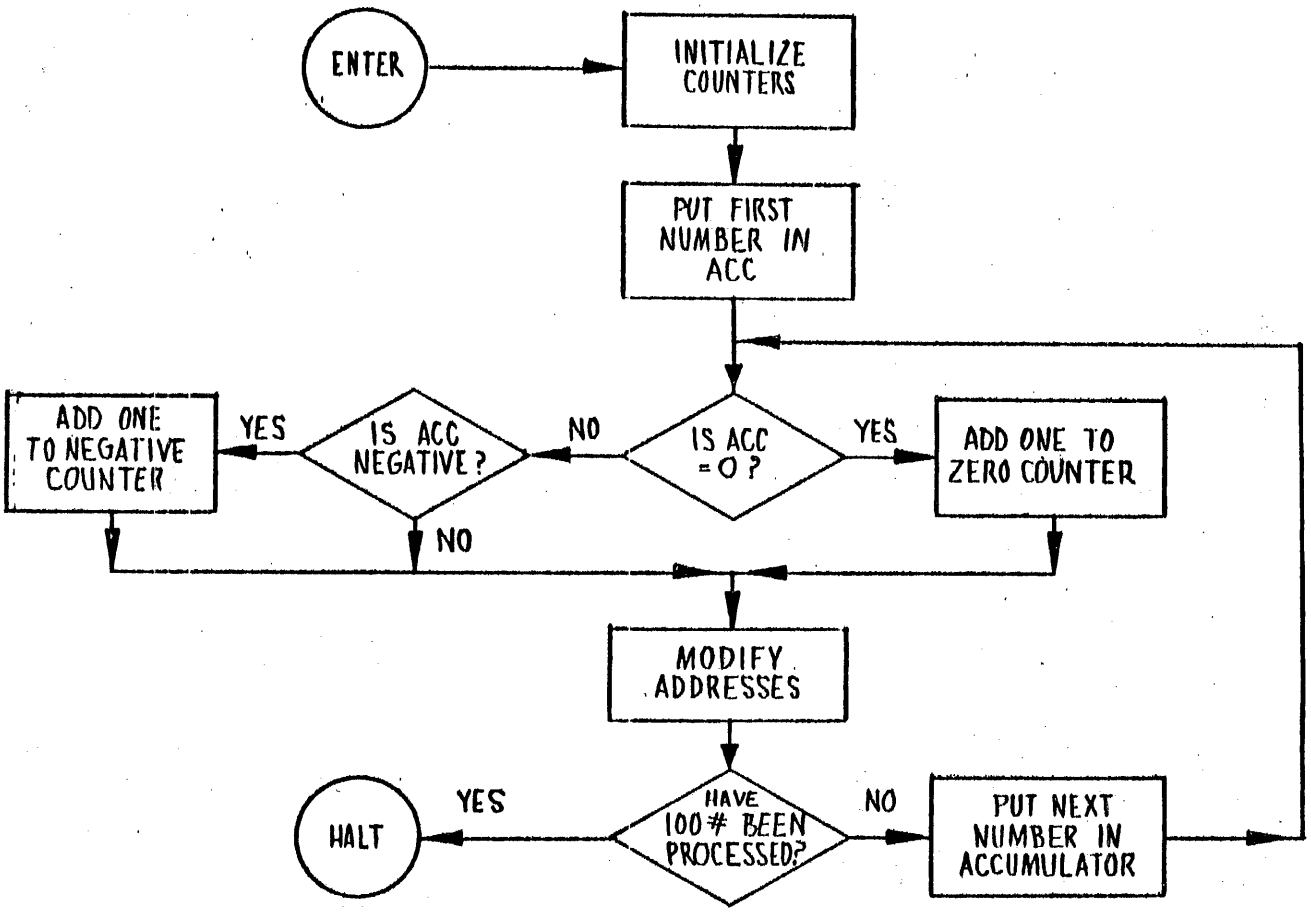
$y = 1/2(.9 + .6)$. .9 is in location 100 and .6 in location 101. Code so as to avoid overflow. Place y in location 102.

3.3 Find the product of the following items:

22.331
2.364135
31615.2
.1

Using the scaling feature of the DEC pseudo-op, enter these items into the computer in such a way that no bits are lost. Where should the binary point be considered to be in the product? Store the answer in location 500.

PROBLEM 4.2.1 - FLOW CHART



3.4 a. Compute: $z = xy$, where $|x| < 1000$ ($\bar{x}$ in location 204)
$|y| < 50$ ($\bar{y}$ in location 205)
$|z| < 4000$ ($\bar{z}$ in location 206)

b. Compute: $R = \frac{s \cdot t}{u \cdot v}$ where $|s| < 300$ ($\bar{s}$ in location 200)
$|t| < 800$ ($\bar{t}$ in location 201)
$|u| < 900$ ($\bar{u}$ in location 202)
$|v| < 100$ ($\bar{v}$ in location 203)
$|R| < 50$ ($\bar{R}$ in location 204)

Assume that once properly scaled, division will take place.

3.5

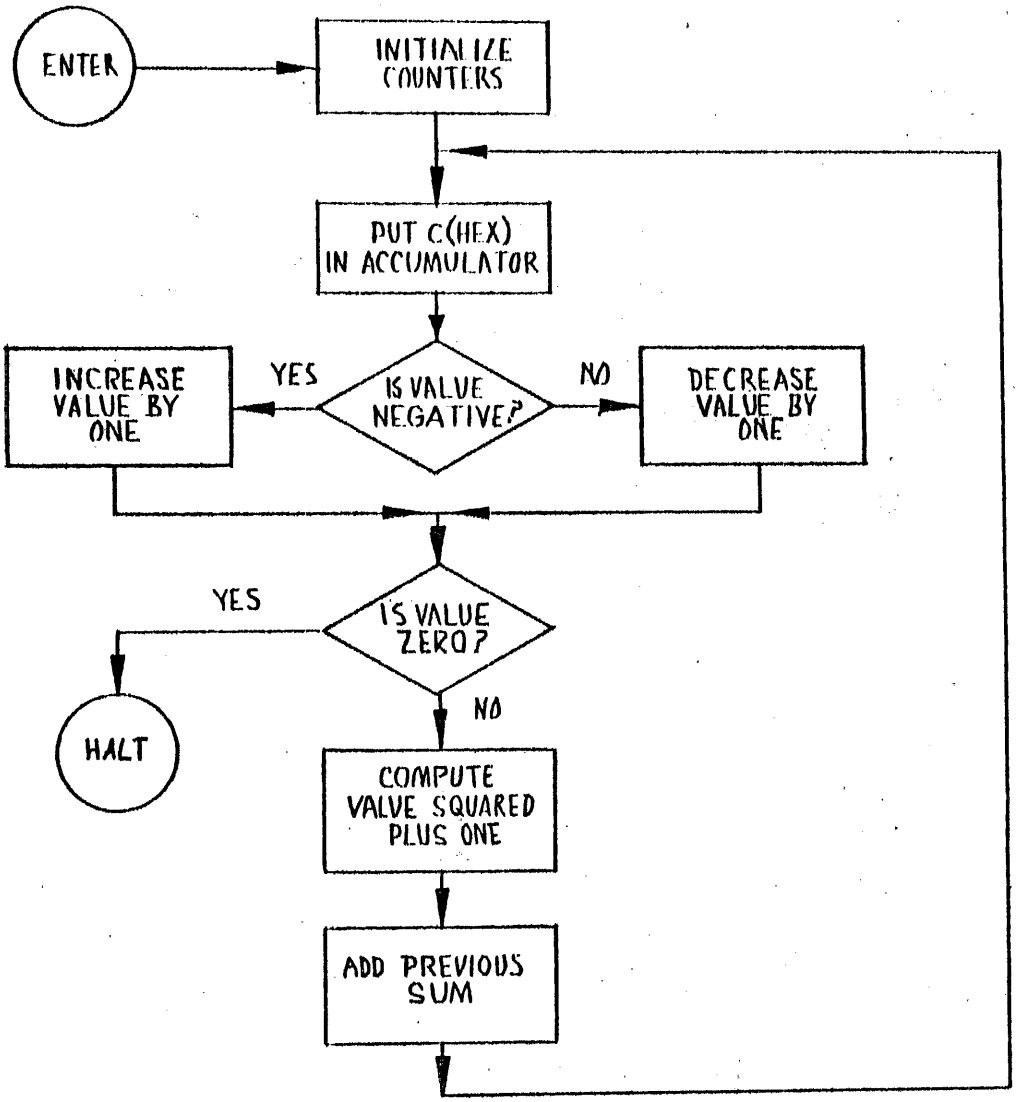
Compute x/y, where both values are unknown. x is in location 100 and y in location 101.

GROUP 4. LOOPING/COMPARISON/ADD BETA

4.1 Compute $x^2 + 1$ for each integral value of x from 5 to 15 inclusive. Store answer in TEMP to TEMP +10.

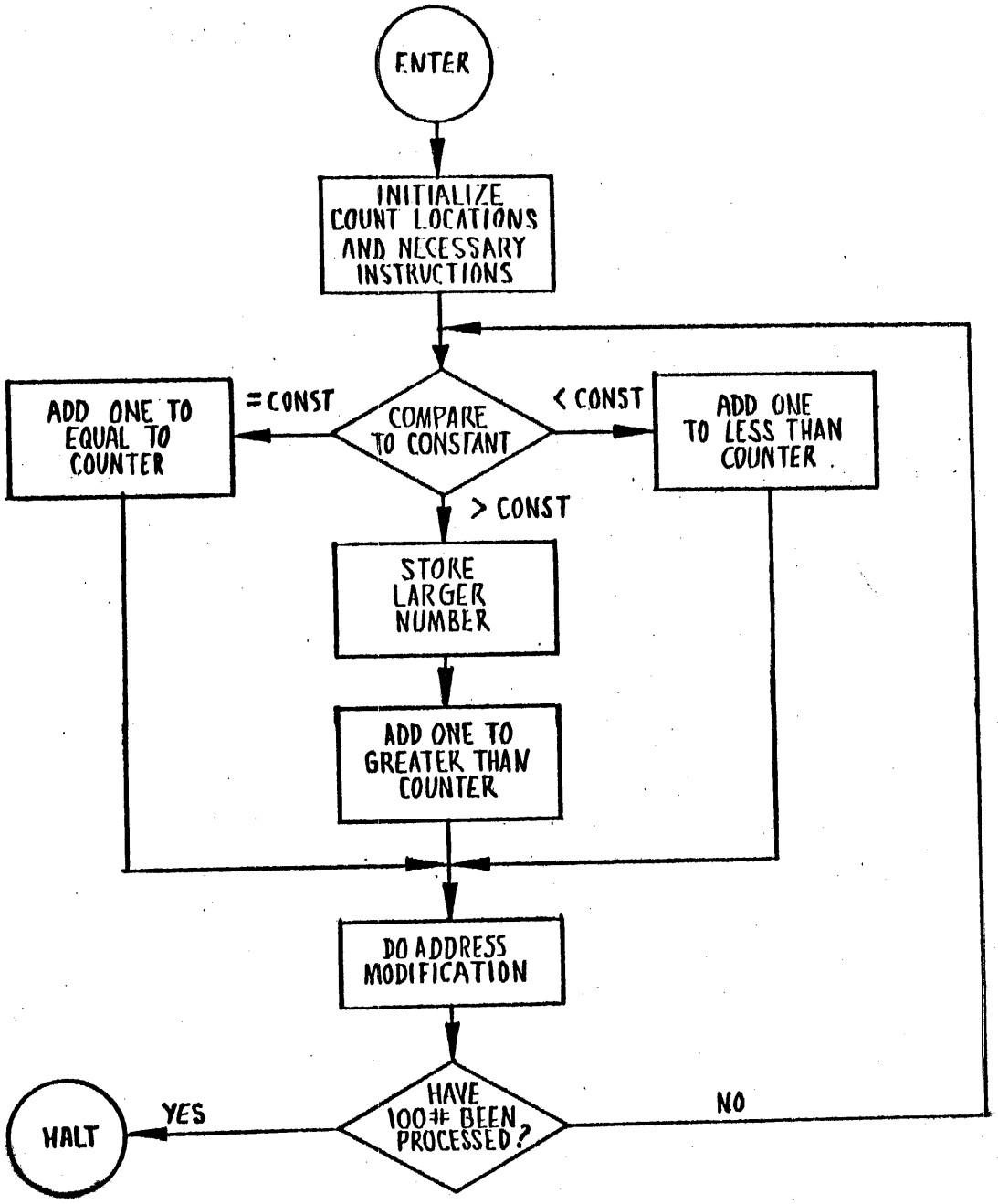
4.2.1 A block of 100 numbers begins at location NUM. Find out how many numbers are equal to zero and then, of the remaining numbers, how many are negative. Use locations ZCT and NCT for counters

PROBLEM 4.2.2 FLOW CHART



4.2.2 x is a non-zero integer stored in location HEX. Test to see if x is positive or negative. If positive, decrease it by 1; if negative, increase it by 1. Now compute $x^2 + 1$ for every integral value of x from the present value of x to, but not including, zero at which time halt the machine. Keep track of the number of times x is computed in location SUMA. Store the total sum in location SUMB

PROBLEM 4.5 - FLOW CHART



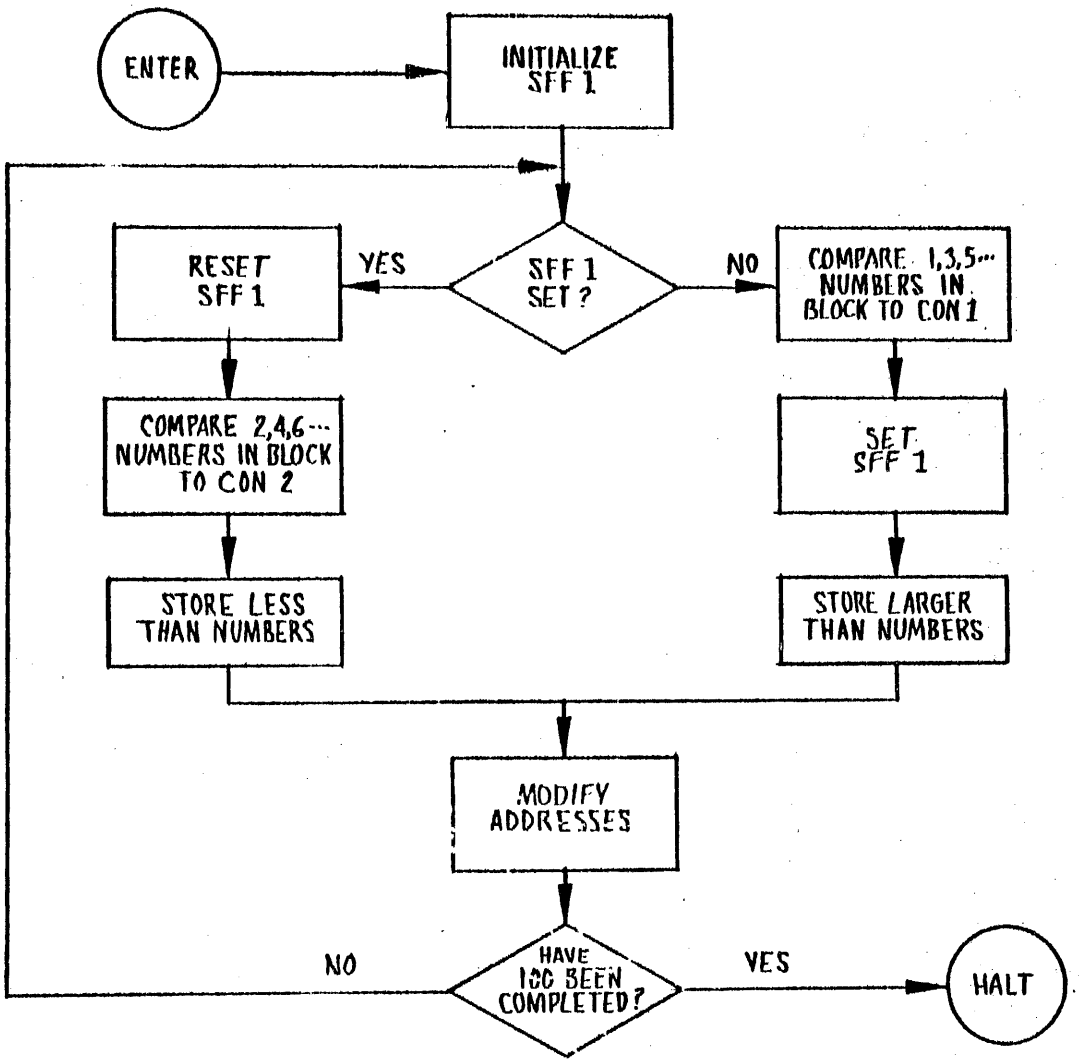
4.3 The Alcoholic Beverages Commission wants to know how many people in a town of 1000 are of drinking age now, and how many more will be next year so that they can estimate from the potential tax income how much raise to give the Commissioner.

The ages of the 1000 people are stored in sequential locations starting at location DATA. Store the number of those of drinking age now in location CT1, those of drinking age next year in location CT2.

4.4 Find the location of the largest number in locations BLOCK to BLOCK +99 inclusive, and store in location ANS. Flow chart the problem first.

4.5 Compare the numbers which are all positive in locations BLOCK to BLOCK + 99 to the value in location CONST. Move all the numbers which are larger than the value in location CONST to consecutive locations LARGE and following. Record in location ANS, ANS +1, ANS +2, respectively the number of values which are greater than, less than and equal to the contents of location CONST.

PROBLEM 5.3 - FLOW CHART



4.6 There are 100 numbers in locations BLOCK to BLOCK +99. Store the number of zeros in location SUM, and the difference between the number of positive and negative numbers in location DIFF.

4.7 Repeat problem 4.22 using Add Beta order.

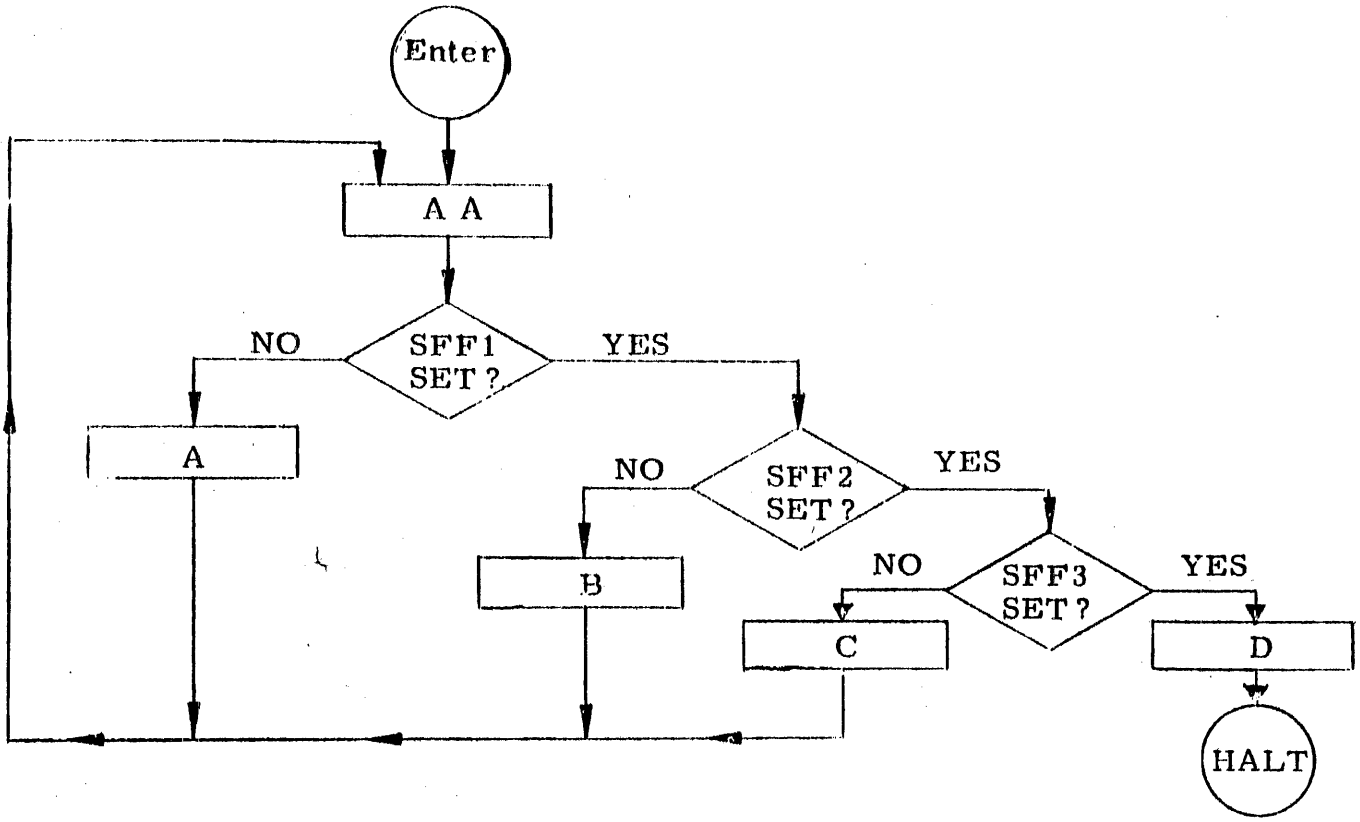
GROUP 5 SENSING/OVERFLOW/REVIEW CONCEPTS

5.1 If $x + y < 1$, compute x^2 and store in location SQURE unless $x < 0$, in which case compute y^2 and store in location SQARE. x is in PARA and y in PARA +1. If $x + y \geq 1$, stop. Assume x and y to be fractions.

5.2 Compute: C(BLOCK) + C(BLOCK +1) - |C(BLOCK +2)| + C(BLOCK +3). Put the result in location ANS. Check for overflow whenever possible. If there is an overflow, set the overflow alarm and add one to an overflow counter.

5.3 A group of data occupies locations BLOCK to BLOCK +99 (decimal locations 50 - 149). Compare locations BLOCK, BLOCK +2 ····· BLOCK +98 (every other location) to CON1 (in location 400) and put the data greater than this constant in location LARGE and following. Compare locations BLOCK +1, BLOCK +3 ···· BLOCK +99 (every other location) to CON2 (in location 401) and put the data less than this constant in location LESS and following. Use a general sense flip-flop as a switch. Start the program in location 500.

5.4 (Demonstration Only)

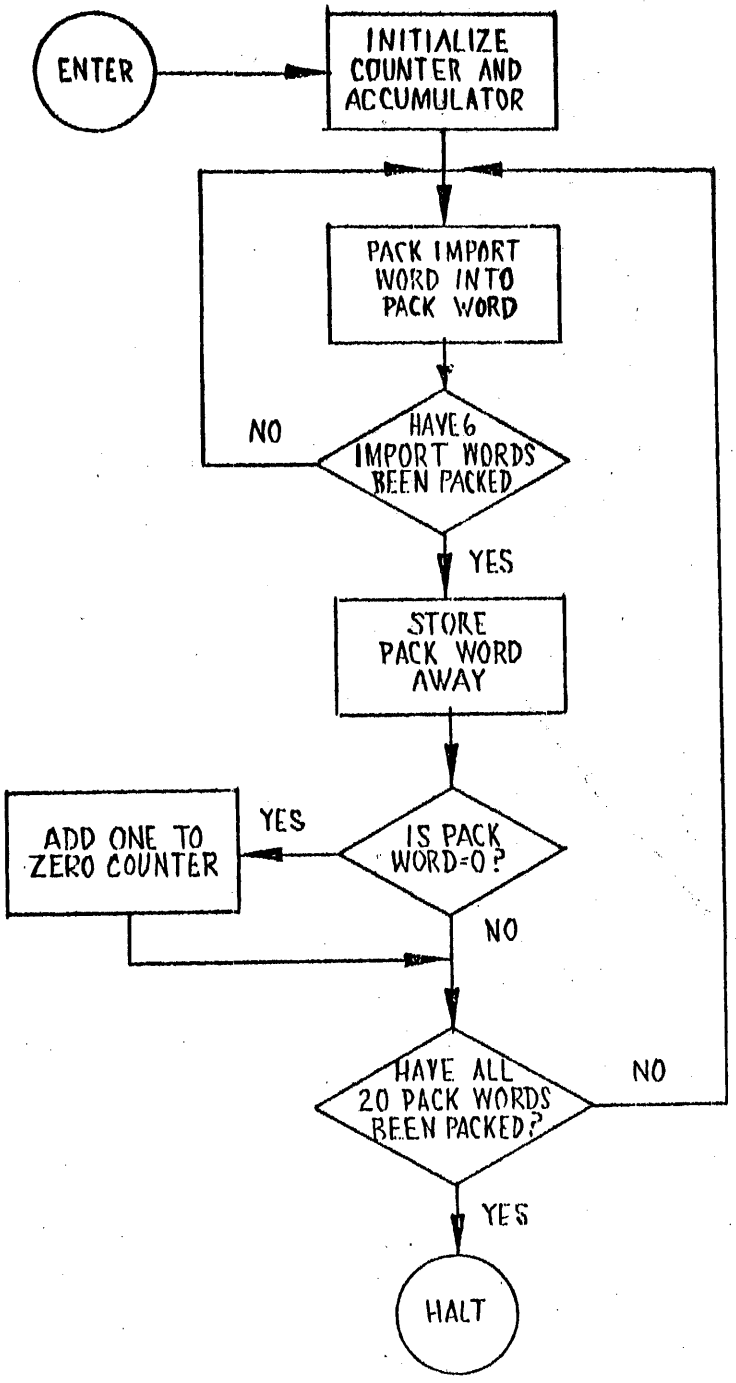


Alternate the program so that the first loop goes through A, the second through B, the third through A, the fourth through C, the fifth through A, the sixth through B, the seventh through A and the eighth through D. Assume no flip-flops are set initially. Be concerned with the sense instructions only.

5.5

Add the contents of every other location from DATA to DATA +199 inclusive. Keep a counter of the number of overflows in location OVC.

PROBLEM 6.7- FLOW CHART



GROUP 6 INDEXING

6.1 Add the contents of locations BLOCK to BLOCK +3000 inclusive. Ignore overflow. Store the sum in location ANS.

6.2 Move a block of 50 numbers starting in location DATA, to locations BLOCK to BLOCK +49.

6.3 Find out how many positive numbers are in a block of data from location SORT to SORT +75. Store the count in location ANS.

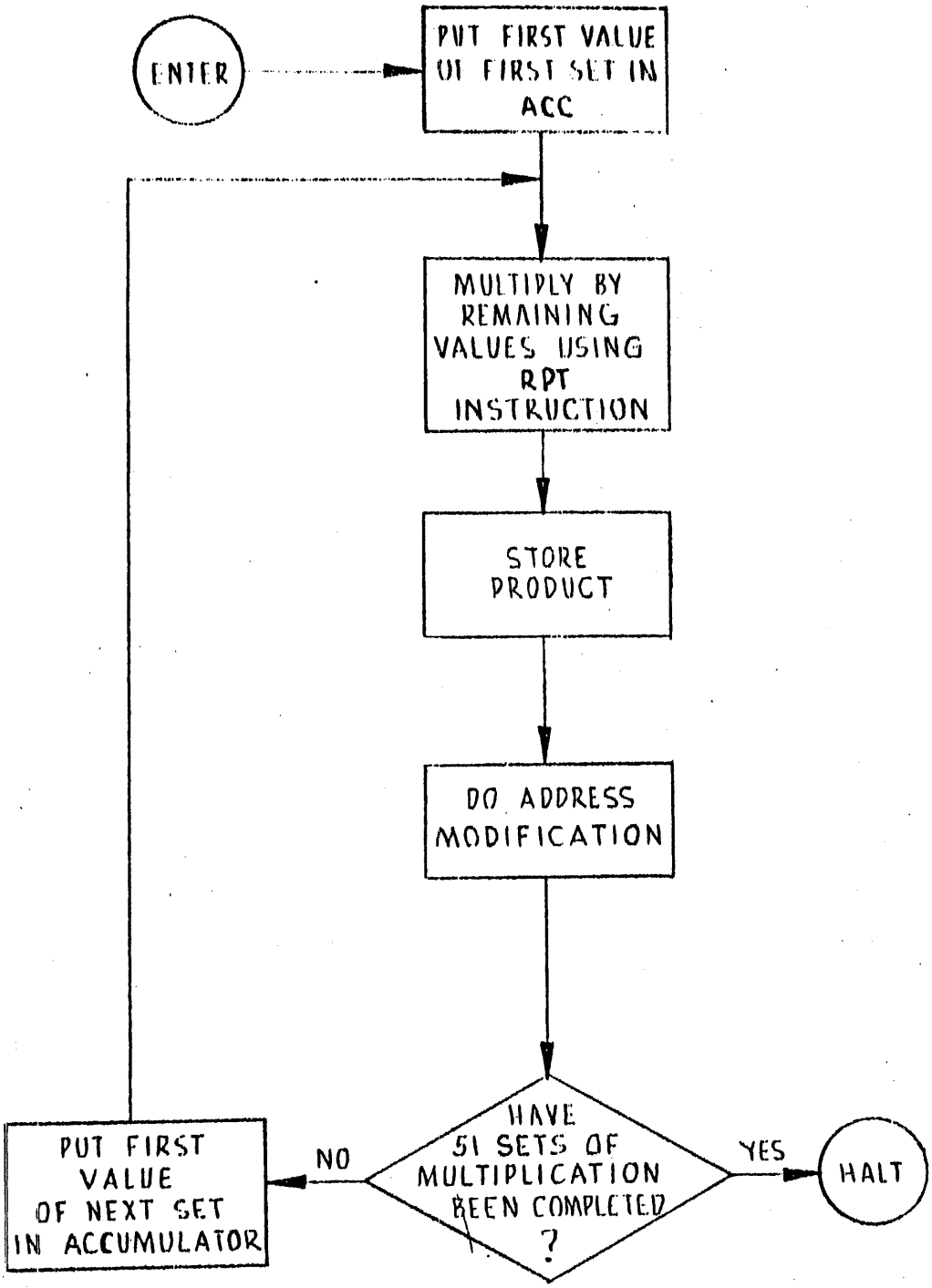
6.4 Write a program to zero all memory locations from TRASH to TRASH +80 inclusive.

6.5 A group of numbers are stored in BLOCK to BLOCK +30 inclusive. Multiply each number by a value in location CONST and store the individual products in consecutive locations PDT and following. Consider all numbers as fractions.

6.6 Assume x is a fractional value in location FRACT and n is an integral value > 0 in location INTEG. Compute x^n, using an index register, and leave the answer in the Accumulator.

6.7 The low order 6 bits of words from locations IMPORT to IMPORT +119 inclusive may contain important data. The remaining 30 bits of each of these locations are 0. Pack all this data into location PACK to PACK +19 inclusive. Keep a count of how many of these locations PACK to PACK +19 contain 0 in location COUNT. You may require all four index registers.

PROBLEM 7.22 — FLOW CHART:



GROUP 7 REPEAT COMBINATIONS

7.1 Add the contents of location SUM to SUM +10 inclusive, ignoring overflow. Store the sum in location ANS.

7.2

7.2.1 Compute q^2xyz. The factors are stored in every other location beginning with location FACT. Store product in location ANS and assume all numbers are fractions.

7.2.2 Consider the fractional values in locations VALUE to VALUE +1000. Compute:

C(VALUE) · C(VALUE +1).....C(VALUE + 1000) and store in location ANS.

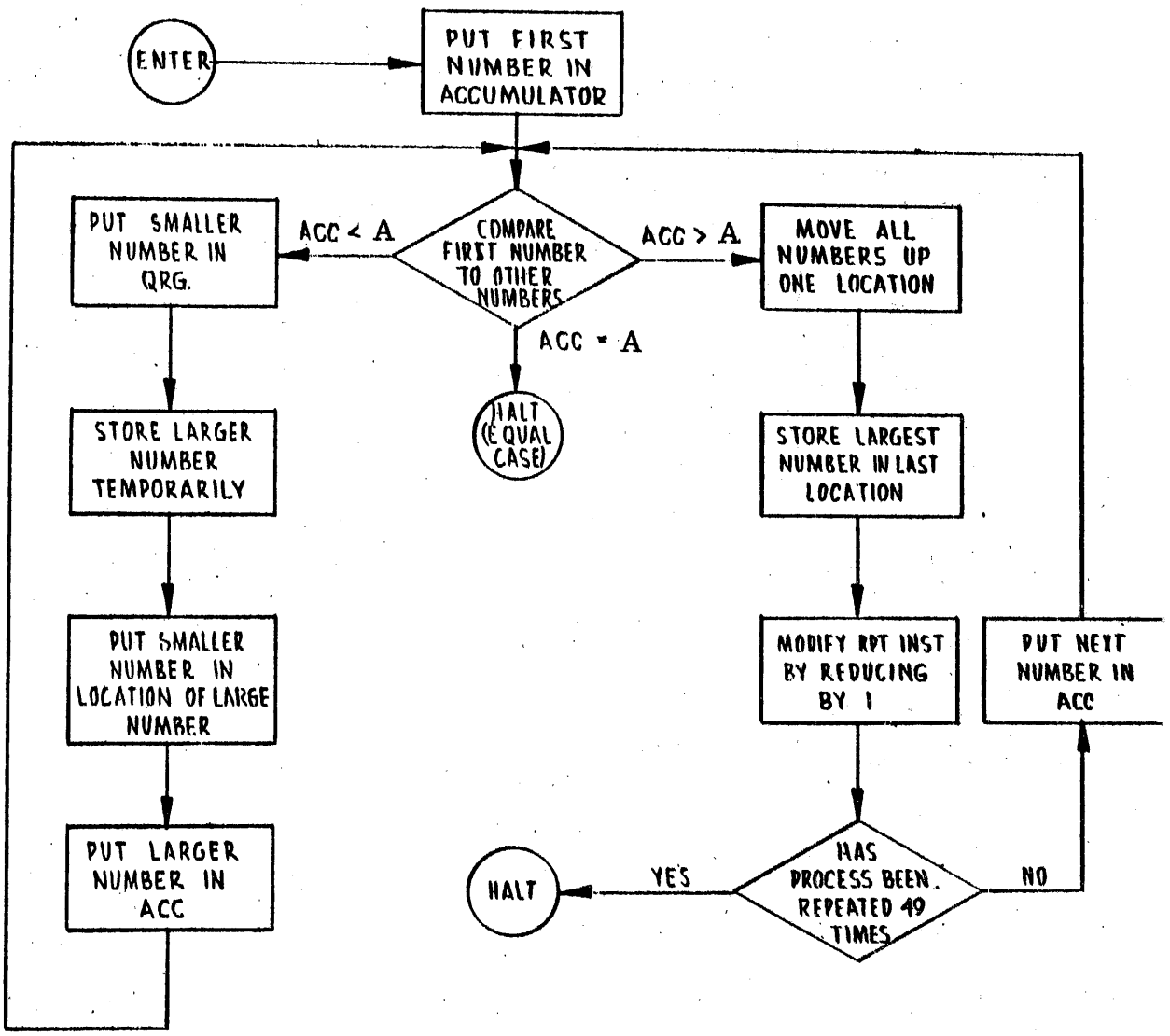
C(VALUE +1) · C(VALUE +2).....C(VALUE +1000) and store in location ANS +1.

C(VALUE +2) · C(VALUE +3).....C(VALUE +1000) and store in location ANS +2.

.
.
.
.
.
.
.

C(VALUE +50) · C(VALUE +51).....C(VALUE +1000) and store in location ANS +50.

PROBLEM 7.4.2—FLOW CHART



7.2.3 Compute: $C_1(X_1 + X_2 + \ldots + X_6) + C_2 (X_1X_2 \ldots X_6)$ and store the result in location ANS. The fractional values X_1 through X_6 are in locations HEX to HEX +5. The fractional values C_1 and C_2 are in locations SEE and SEE +1.

7.3

7.3.1 Move every other number in locations DATA to DATA +100 inclusive to every fourth location starting at MOVE.

7.3.2 Move a block of 50 numbers starting in location STORE to STORE +49 inclusive to every other location starting at BEGIN.

7.3.3 A block of 75 numbers begins at location A. Move the first 25 numbers to locations B and following, the second 25 numbers to locations C and following, and the last 25 numbers to locations starting at D. Set every other location to zero, from A to A +74, after completing the MOVE operations.

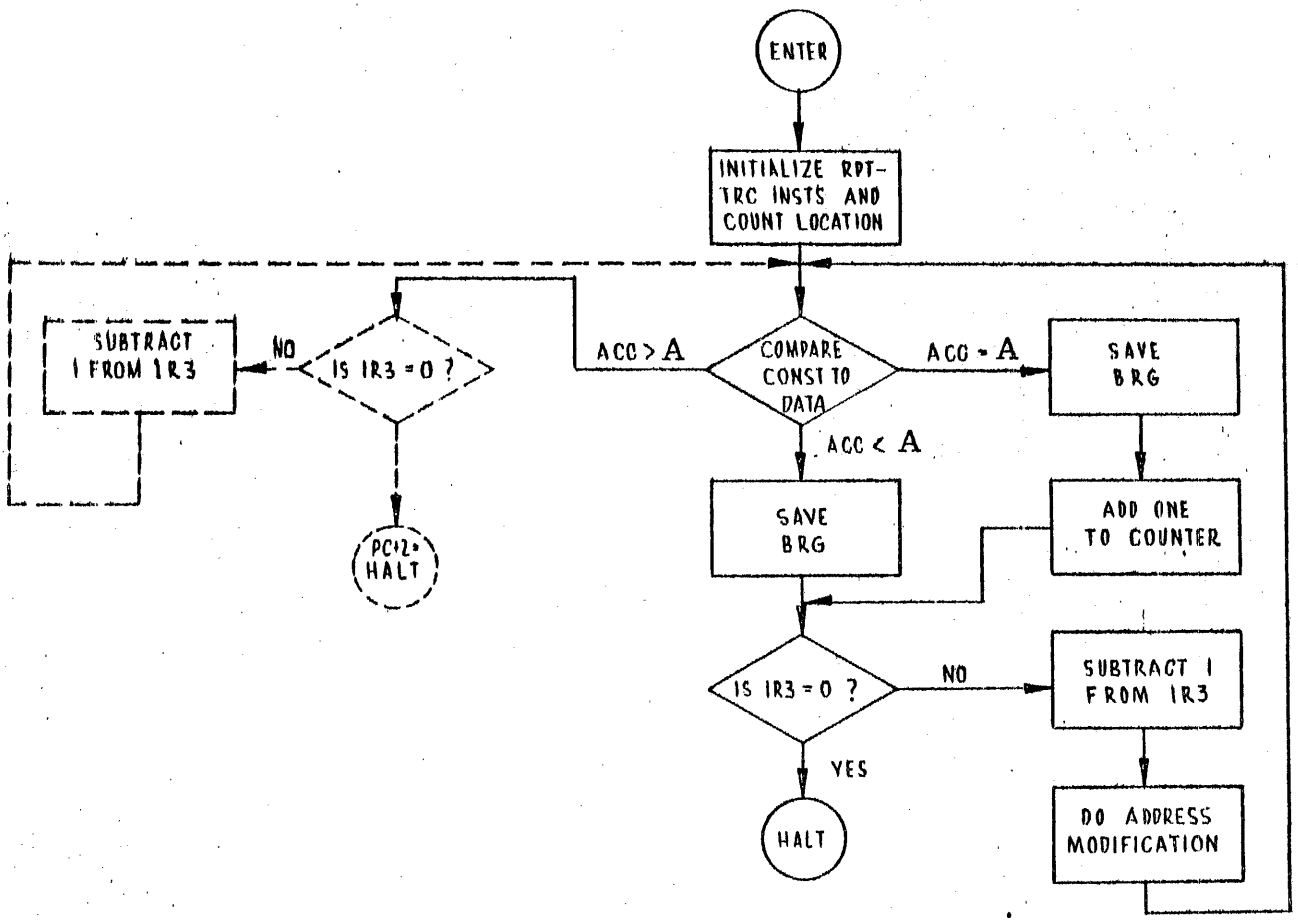
7.4

7.4.1 A block of 100 numbers starts in location FIELD. Compare these numbers with X in location CONST. If there are any numbers which are greater than or equal to X, set sense flip-flop 1 and halt. Otherwise set flip-flops 2 and 3 before halting.

7.4.2 (Demonstration)

Rearrange a block of 50 numbers starting at location A, so that they are in ascending order. If there are any numbers that are equal, stop the computer.

PROBLEM 7.4.3--FLOW CHART



- - - - - DASHED BOXES ARE DONE AUTOMATICALLY BY THE RPT- TRC INSTRUCTION SEQUENCE.

———— SOLID BOXES MUST BE PROGRAMMED.

7.4.3 Compare Z, a number in location CONST, with C(DATA) to C(DATA + 150) inclusive. Tabulate the number of values which are equal to Z, using RPT-TRC, and store in location COUNT.

GROUP 8 FLOATING POINT OPERATIONS

8.1 Find the normalized sum of the 200 floating point numbers in consecutive locations DATA and following. Store the sum in location SUM. Assume no overflow. (DATA = DECIMAL LOCATION 500)

8.2 Compute the following using floating point notation:

$$21.943 - \frac{(18.7)(23.591)^2}{(19.1)(.00246)} + 17.843\left[15.691 - 128\left(\frac{7.43}{14.111}\right)\right]$$

Store the answer in location ANS.

8.3 Place the normalized value X^n in location NORM, where $X(\neq 0)$ is a normalized floating point number stored in location FLOAT and $n \geq 0$ is a fixed point integer stored in the address portion of location FIX, and is unknown to the programmer. Assume no overflow.

GROUP 9. CYCLING/MASKING/LOGICAL OPERATIONS

9.1

9.1.1 Consider location EDIT to be halved and reverse halves. Bit 19 of the original word EDIT (before reversing) contains significant information. Place bit 1 in Q register without changing sign of Q register, and if it is 1, set sense flip-flop 1 before halting.

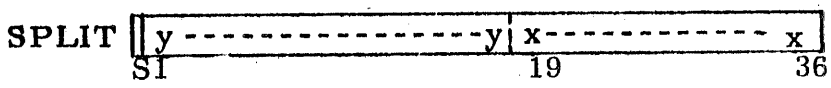
9.1.2 In order to satiate those who have a desire to solve puzzles, do the following problem assuming the Accumulator and Q register have no sign bits:

In the Accumulator and Q register are:

Rearrange the 9-bit groupings (labeled alphabetically) so that they appear in alphabetical order (according to their labels) in the Accumulator and Q register.

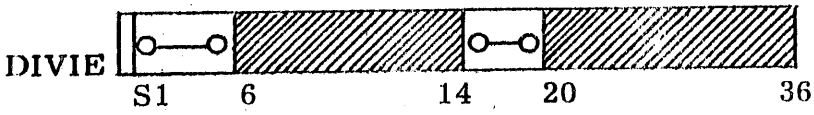
9.2

9.2.1 In location SPLIT are two values, x and y:



Place x and y in the low order bits of location HEX and WHY respectively.

9.2.2 In location DIVIE are two pieces of information:



Double each shaded area and place in location SHADE. If any overflow occurs into the unshaded area, let it remain, but set sense flip-flop 1.

9.2.3 Put the OP CODE of location INST, the I portion of location INST +1, the m portion of location INST +2, in their respective positions in location PACK. Allow the a portion of location PACK to remain unchanged.

9.3

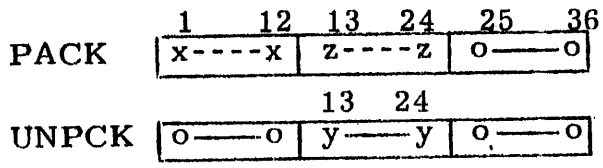
9.3.1 a) Put bits 12-36 of location HEX and bits 1-11 of location WHY into location PACK. Assume rest of HEX and WHY are equal to zero.

b) Isolate bits 12-23 of location MESS and store them in the same positions of location CLEAN.

9.3.2 Put $C(SEPER)_{12-22}$ into location $CLAMP_{12-22}$ without disturbing the remainder of location CLAMP. Assume rest of SEPER is zero.

9.3.3

Place the "Y" portion of location UNPCK into bits 25-36 of location PACK; also complement "Z" portion of location PACK.



GROUP 10 SUBROUTINES/ FLOATING POINT/ DOUBLE PRECISION

10.1 Compute: $x = \sqrt{2a - 1}$ where a is in location HAY, and the square root subroutine is at location SQRT. Store answer in location ANS. Assume the subroutine to be already present. Assume the entrance to subroutine requires parameter in Accumulator and answer in Accumulator after execution of subroutine.

10.2 Write a subroutine that will multiply a block of K integers, beginning at location NO, by a constant integral multiplier, stored at location M6. Store the products beginning at location P8. Start the subroutine at location SO. The following is given as that part of the main program which links the main program with the subroutine:

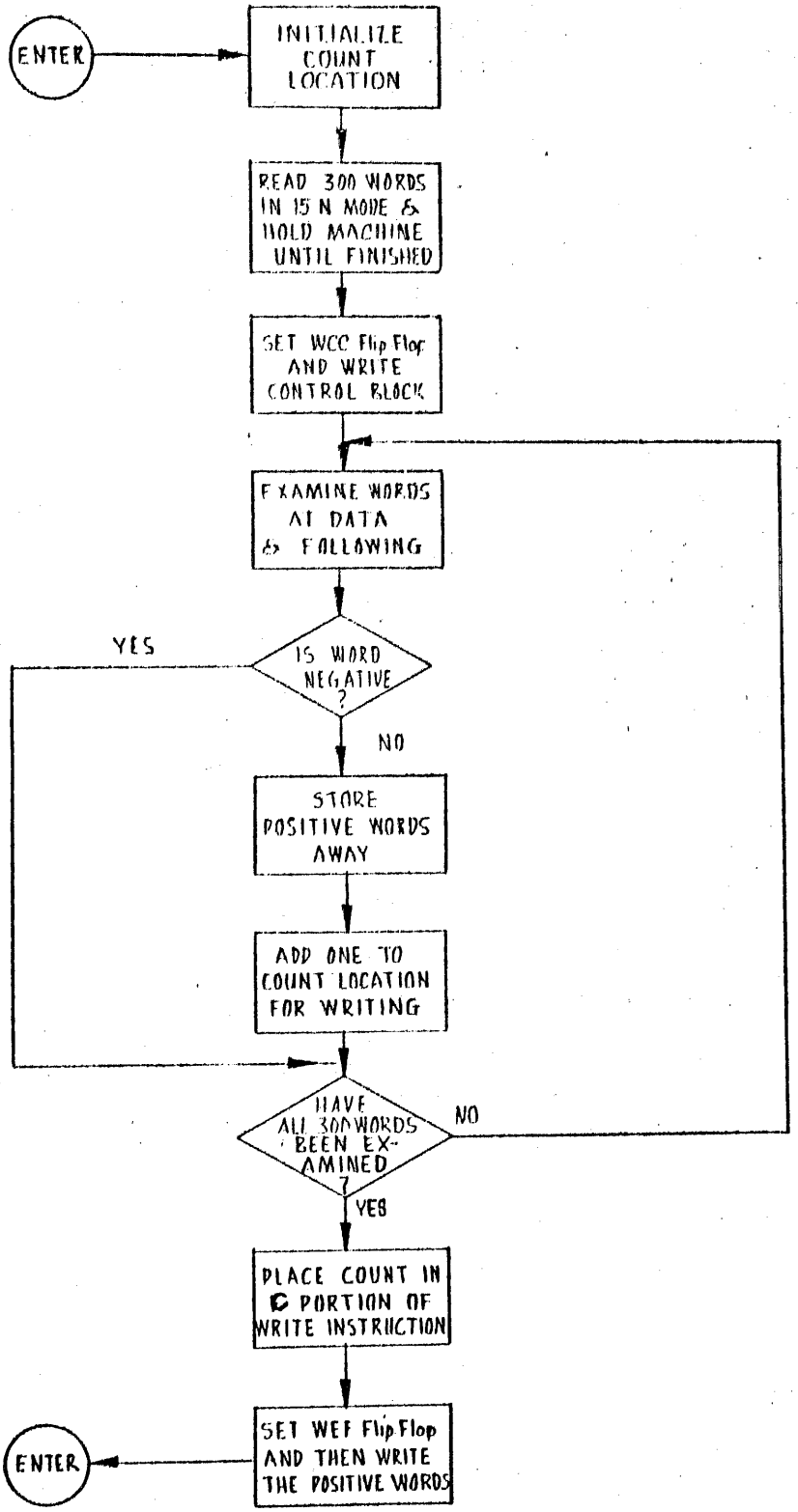
```
RO      TRL     SO
        HLT     NO      location of first integer
        HLT     K       number of integers
        HLT     P8      location of first answer
        HLT     M6      location of constant multiplier
(Reentry Point)
```

Assume that the product is less than 36 bits.

10.3 Write a subroutine that will develop a logical sum (LGA) for a variable length block of memory. Start the subroutine in location LOGSM. The following is given as part of the main program which links the main program with the subroutine (assume the main program will "housekeep" the necessary registers)

```
SUB     TRL     LOGSM
        HLT     N       (N = number of words in block)
        HLT     M       (M = location of first word in block)
        HLT     P       (P = location in main program where
                            logical sum should be stored).
```

PROBLEM 11.2 ——— FLOW CHART



10.4 Numerical examples using scientific notation:

Add: $.3298 \times 10^{20}$ $.3167 \times 10^{20}$

$.0032 \times 10^{20}$ $.4567 \times 10^{18}$

Multiply: (.000005)(15.1)

10.5 Assume all memory words and registers are of a 2 bit length. Number N1 (.3274) is stored in SPLIT and SPLIT +1. Number N2 (.3651) is stored in DIVIE and DIVIE +1. Add N1 + N2 and store answer in locations ANS, ANS +1, and ANS +2. All three locations will be necessary if final sum is more than 4 digit number.

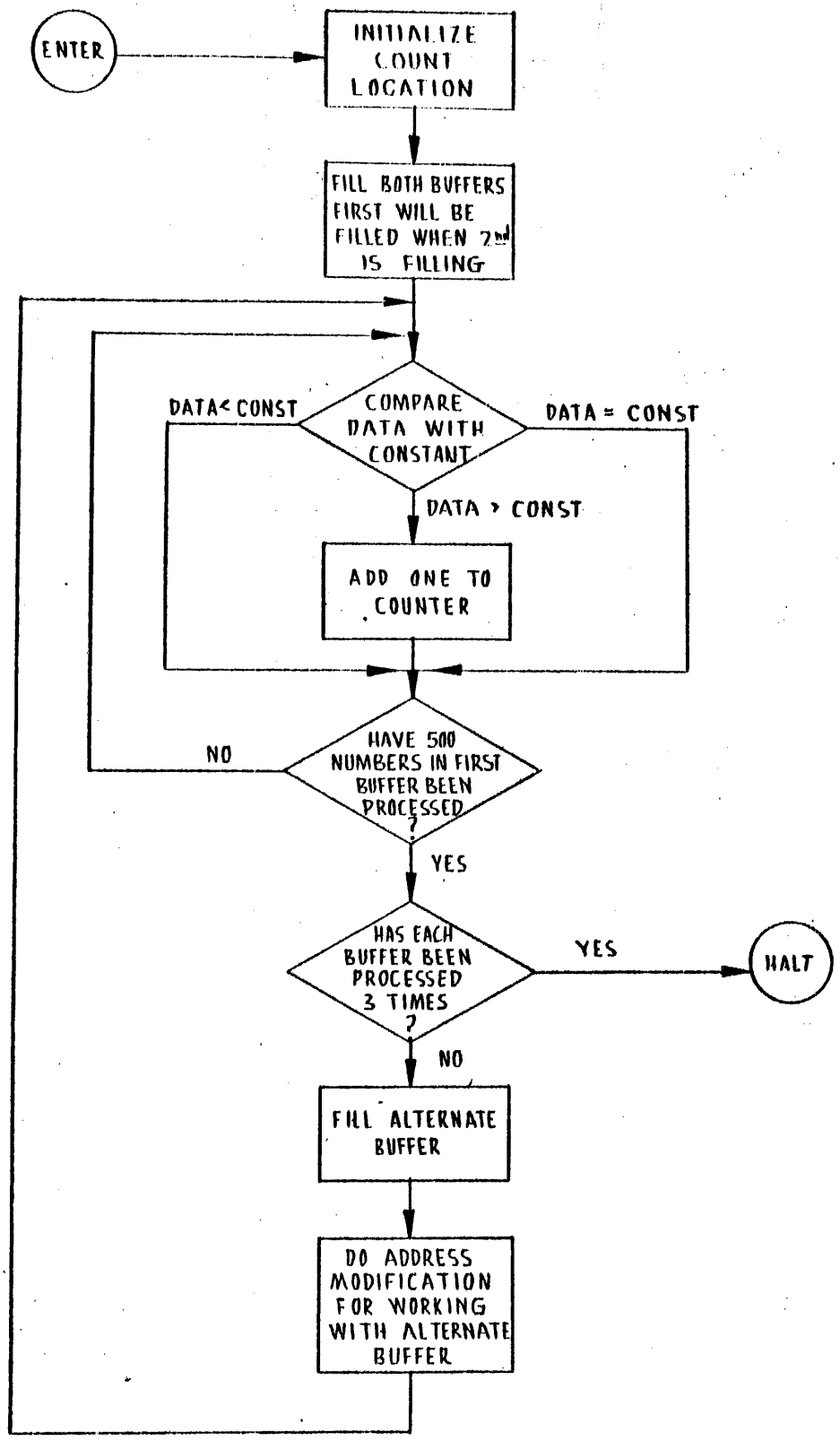
GROUP 11 INPUT-OUTPUT/MAGNETIC TAPE

11.1 Read 300 words with sign from magnetic tape 1 into location DATA and following. Write all the positive numbers consecutively on magnetic tape 2.

Caution: Unless all positive numbers are consecutive, they will not be stored consecutively on magnetic tape 2, because spaces will be left for negative numbers.

11.2 Read 300 words with sign from magnetic tape 1 into location DATA and following. Assume the words are assembled in blocks of fifty each. Write all the positive numbers consecutively on magnetic tape 2 as one block. Precede this block with another block of five words containing control characters located at CONTR and following. Allow the two blocks to constitute one file. Assume there is at least one positive number in the group.

PROBLEM 11.5 —— FLOW CHART



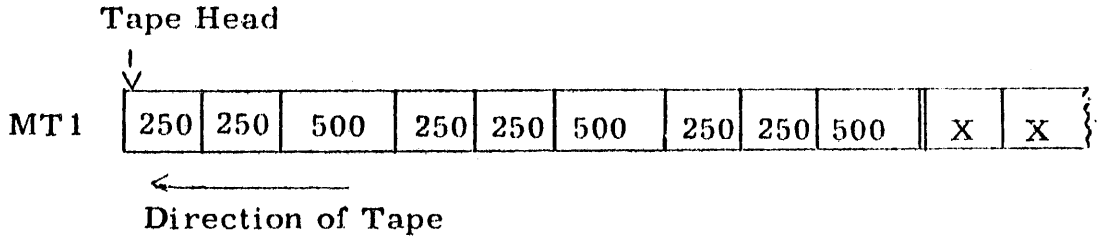
11.3

Write the information from location ANS to ANS + 299 on magnetic tape 1 so that it appears as six equally divided blocks, two blocks to a file. Write all information in the Interpret Sign Mode.

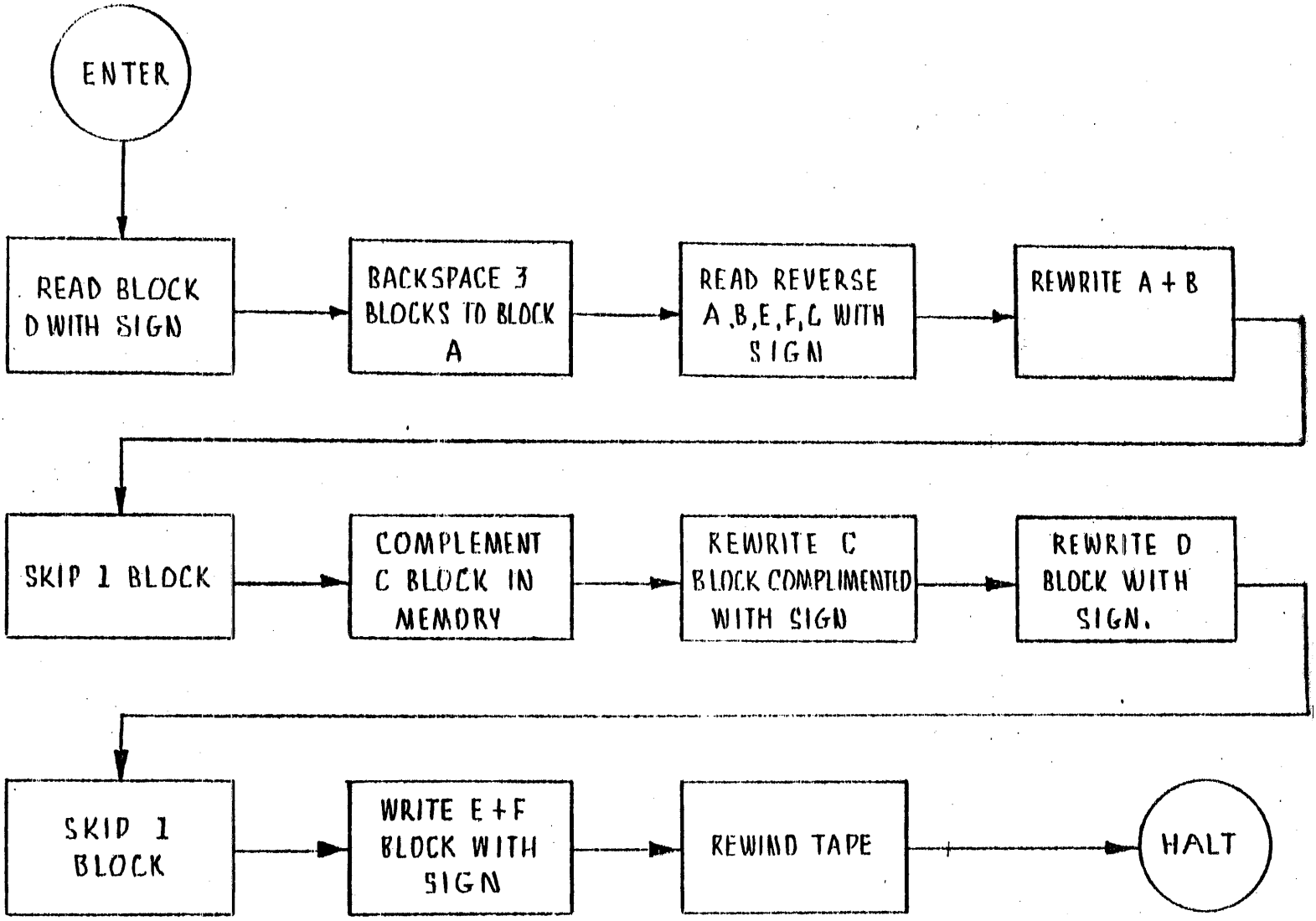
11.4 Assume that the order RAN DATA, MT1, 100 has been executed in a program. Name the different ways of holding up the remainder of the program until this Input order has been completed.

11.5 (Demonstration)

Two buffer areas of 500 random words each, located in BLOCK and following and DATA and following, are to be filled and processed alternately and filled from magnetic tape for further processing until each has been processed 3 times. The words are on tape in blocks of the arrangement shown on the following page. Examine all the numbers and keep a running count of all those greater than the contents of location SUM; store the count in location COUNT. Do not process X blocks.



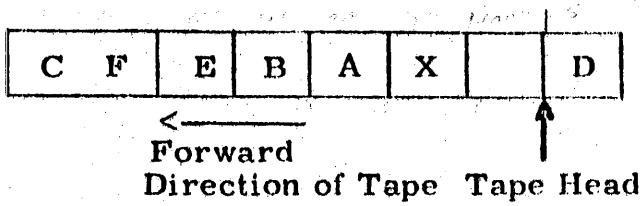
PROBLEM 11.7 - FLOW CHART



11.6

Read two files from magnetic tape 2 in the Interpret Sign mode. There is a maximum of 1000 words in each file.

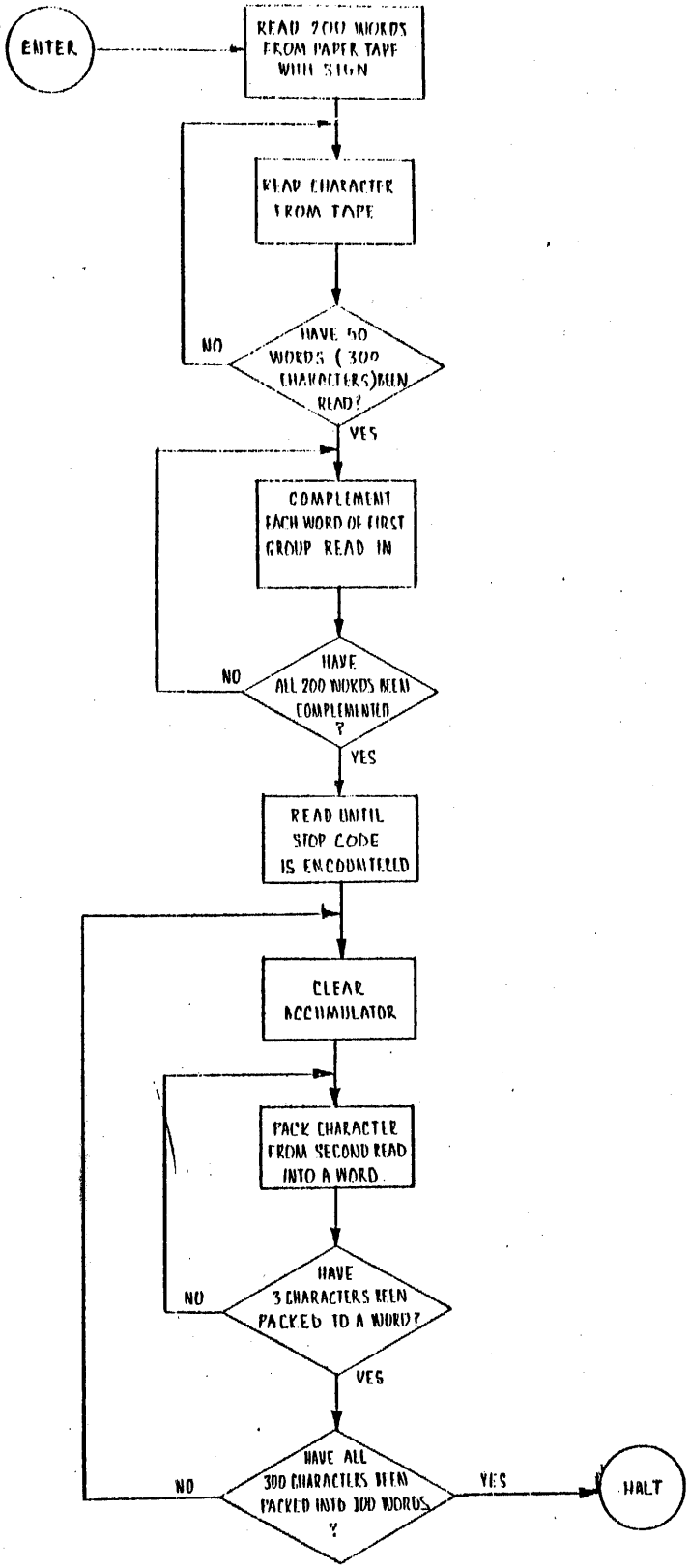
11.7 Rearrange the following magnetic tape blocks (each letter represents 50 words):



so that the same section of tape looks like this:

and then the characters of block C are complemented. Interpret all blocks with sign and rewind tape when finished.

PROBLEM 12.3 FLOW CHART



GROUP 12. INPUT-OUTPUT/PAPER TAPE

12.1 Read 100 non-zero words from paper tape in octal into locations DATA and following. Compute the square root of each of these numbers and write them out in octal on the same paper tape. Assume the entrance to the square root routine is at location SQRT and it is entered with the argument in the Accumulator and exits with the result in the Accumulator. Make the computation time simultaneous to processing time.

12.2 There exist 100 words starting at memory location BEGIN. Put the 100 words on paper tape and at the same time put all the positive numbers greater than the number in location CONST in location POSIT and following.

12.3 On paper tape is a group of at least 500 (and less than 1000) non-zero alphanumeric words, ending with a STOP CODE in the last word. Read the first 200 words into location FIRST and following, with sign, and then complement them; read the next 50 words in the character by character mode, and then put these characters, three to a word, in memory location THREE and following. Read the remainder, without sign, into memory location SAVE and following. Assume the first 200 words are in the Interpret Sign mode and the remainder in the non-Interpret Sign mode.

12.4 Assume there are 20 blocks of 50 words each, with sign, on magnetic tape 1, and 70 words in alphanumeric, with sign, on a paper tape.

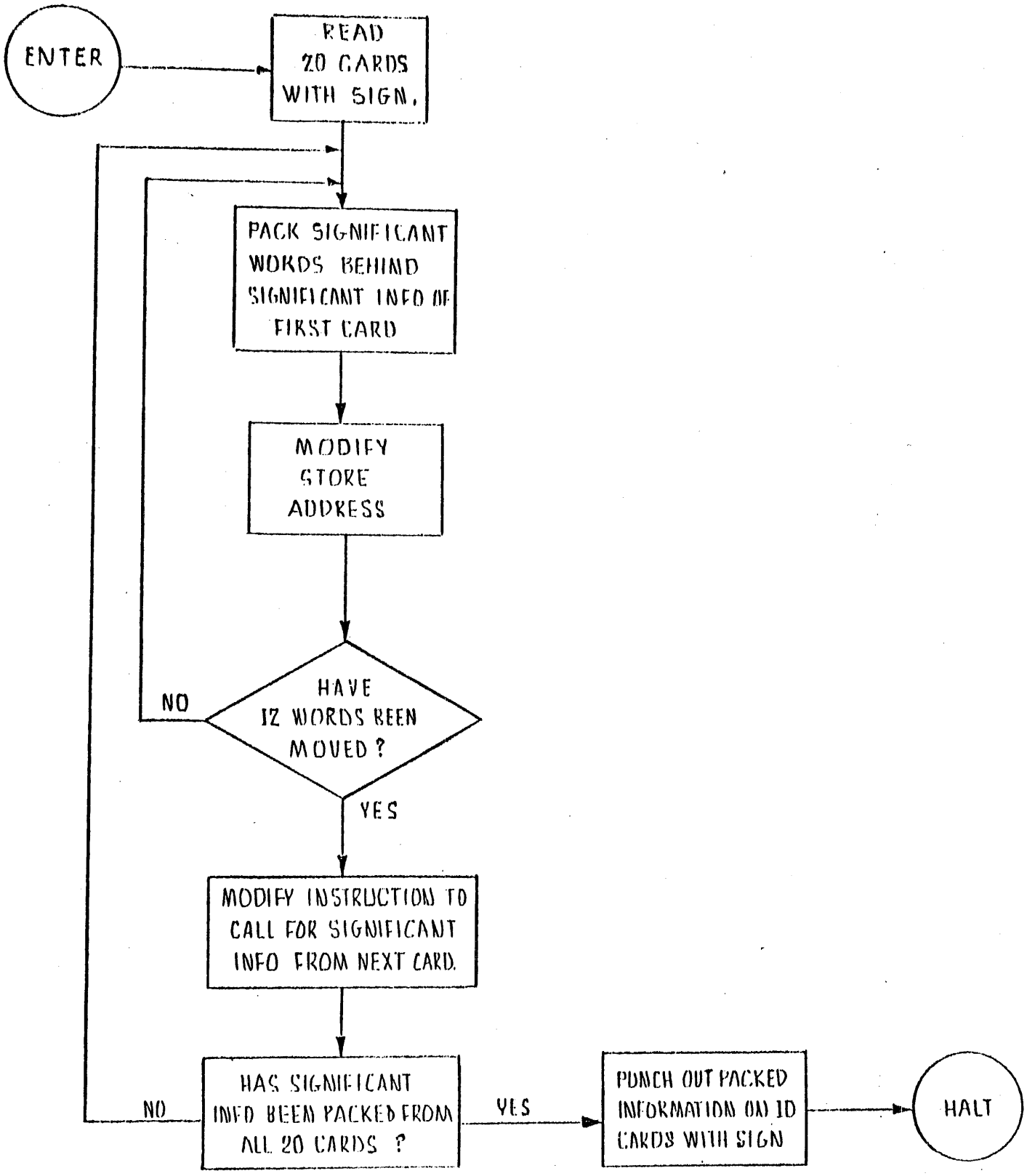
Read the first ten blocks into location DATA and following. Write blocks 16-20 in the same place on tape as blocks 11-15. Put contents of blocks 13-15 on paper tape with sign followed by a stop code. put out the 70 words from the first paper tape following this. Flow chart the problem first.

GROUP 13. OUTPUT/FLEXOWRITER

13.1 Write an octal dump of the first 4005 words of memory onto the Flexowriter using the following format on the printed page:

First Three Words	Second Three Words	Third Three Words
xxxxxxxxxxxxxxxxxx	xxxxxxxxxxxxxxxxxxxxx	xxxxxxxxxxxxxxxxxxx
THE SAME FORMAT AS ABOVE FOR NEXT 3996 WORDS.		
xxxxxxxxxxxxxxxxxx	xxxxxxxxxxxxxxxxxxxxx	xxxxxxxxxxxxxxxxxxx

PROBLEM 14.1 - FLOW CHART



14.1 There are 20 cards that have been punched alphanumeric in the interpret sign mode. Each card has significant information in the first 12 words only. Read these cards into locations xx and following. Punch new cards, 24 words per card, from location xx and following; each card is to contain only significant data.

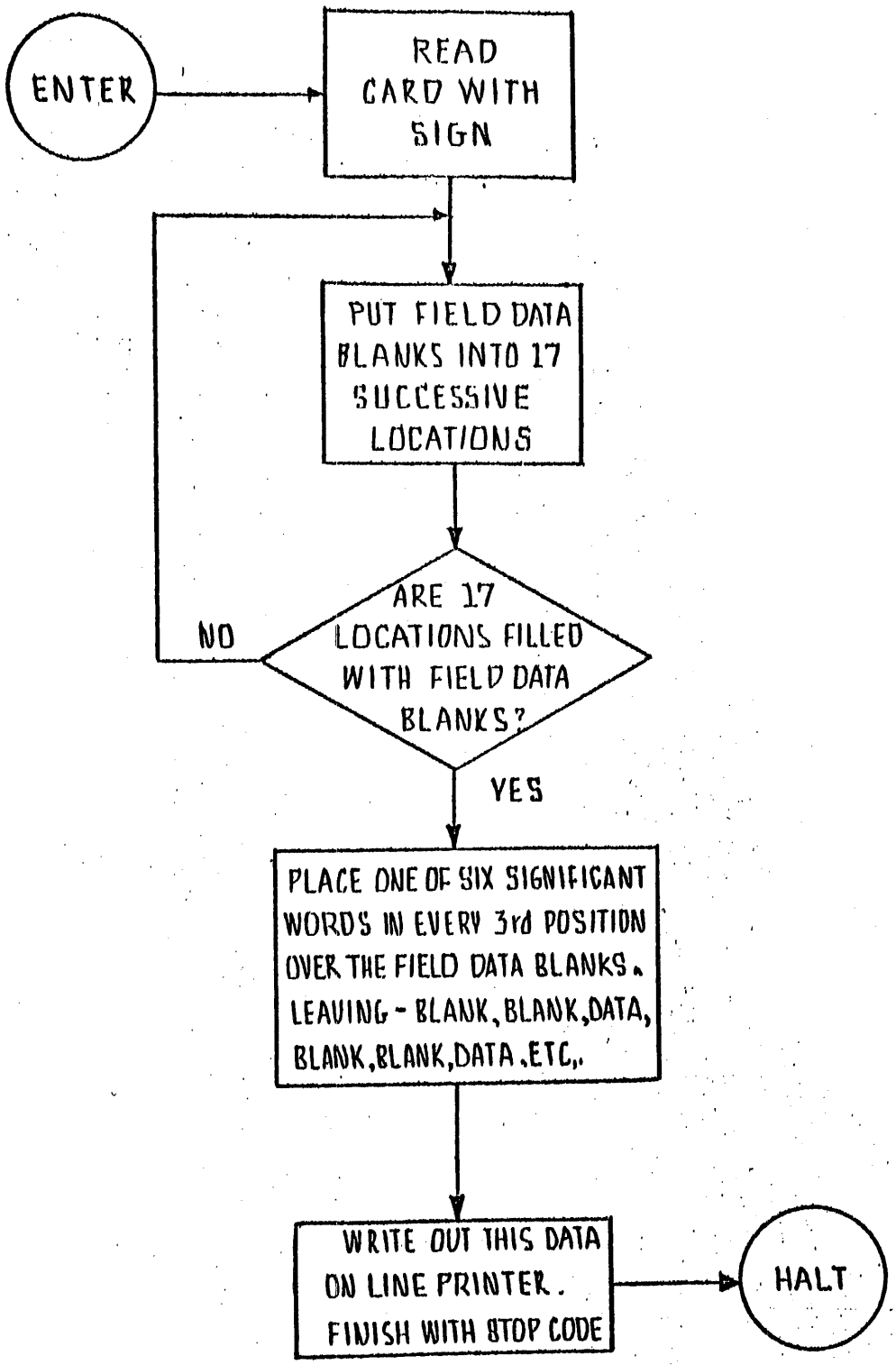
14.2 On MT1 in the ISN mode, there is a block of 300 words positioned for reading with the following format:

S1	Alphanumeric Project No.	18 19	Binary Dollars Allotted to Project	36

There are also 20 ISN cards positioned in the card reader with this same format.

Write out new ISN cards, in the same format, for those words on the cards which do not have a matching project number on the tape.

PROBLEM 15.1 - FLOW CHART



GROUP 15. OUTPUT/PRINTER

15.1 A card contains 6 words, punched alphanumerically in the interpret sign mode, starting in row 0. Read this card into memory, and print this information centered on one line, omitting the sign, and with 12 spaces between each group of 6 characters.

Assume the printer to be in automatic line feed mode. Assume that none of the characters are control symbols. Assume a straight 120 x 120 plugboard.

GROUP 16. ORDER SEQUENCE MODE OPERATIONS

16.1 Write out on Mag tape 4, five blocks of information in the interpret sign mode in alphanumeric, 50 words in each block in sequential locations starting with DATA (decimal loc. 1000). End the fifth block with an end of file mark. Interrupt the program after writing the second block, set sense flip-flop 2 and then return to normal sequence. Rewind the tape when finished. Use processor No. 2.

16.1.1 Write one block on MT2 in alphanumeric form in the Interpret Sign Mode 1000 words from the following locations:

250 words from location COUNT (decimal location 1000) and following.

500 words from location DATA (decimal location 1500) and following.

250 words from location SUM (decimal location 2500) and following.

When finished, stall the computer for four minutes (approx.) allowing the operator time to change the tape. Then write the same data on the new tape.

16.2 Write out on MT2 five blocks of alphanumeric information in the interpret sign mode from the following locations:

Block 1 - DATA (decimal location 500)

Block 2 - BLOCK (decimal location 1100)

Block 3 - NUMB (decimal location 1700)

Block 4 - RARE (decimal location 2300)

Block 5 - NEVER (decimal location 2900)

Make the number of words that you write in each block dependent upon the value entered (by manual control) in the Word Switch Register bits 1-9. If Sense flip-flop 1 is set, drop end of file marks when finished.

16.3 Read the first 150 words of each of three blocks of alphanumeric information on MT1 in the NISN mode as follows:

First 150 words of the first block into location COUNT (decimal location 1000)

First 150 words of the second block into location DATA (decimal location 1500)

First 150 words of the third block into location SUM (decimal location 2000)

Don't halt on a processor error. If an input parity error was caused, read the data once more. Keep repeating the reading until there is no parity error.

16.3.1 Read all the keys (less than 50) in from one file (< 255 blocks) on MT3. If the logical sum of all the keys is all ones, then backspace and read into location DATA the data between the third and fourth key.

16.3.2 This is a program to investigate the correctness of one phase of the order sequence mode. Write out one block of information on MT2 alphanumerically in the interpret sign mode in the following manner:

100 words from loc. MORE (decimal location 5000)

150 words from loc. LESS (decimal location 4000)

200 words from loc. LEAST (decimal location 3000)

Read the information back in and check it against original. If any data error, write "ERROR" out on Flexowriter. If any processor errors, read again until no errors take place.

16.4 In location KEY (decimal location 100) there are three 12 bit keys. They appear on MT2 in one file; KEY_{1-12} appears first, KEY_{13-24} appears second, and KEY_{25-36} appears third. Once a key is found, read the alphanumeric data in the ISN mode in the following manner:

Words after first key and before the next key - into location DATA

Words after second key and before the next key - into location SUM

Words after third key and before the next key - into location LOC

Write on the flexowriter the number of words you have read after each reading. (There are a maximum of 500 words between keys). If any key is not found then also write on the flexowriter "ERROR--SOME KEY WAS NOT FOUND".

16.5 Read five blocks from MT2 (alphanumeric in ISN mode), in such a manner that no stops will occur if there is a processor error. If a processor error was caused during the reading, set sense flip-flop 1.

If that error was an IPE (Input Parity) error, go back and read the five blocks again.

If no error existed, write "NO ERROR IN READ OPERATION" on the flexowriter and rewind MT2.

Assume the five blocks have a maximum of 50 words each.

16.6 Check every other block (< 500 words) in one file (< 255 blocks) on MT3 for zero words. Every time a zero word is examined add one to a counter. When finished write the counter out on the Flexowriter. Assume that a minimum of memory space is available to you.

16.7 Read alphanumeric Magnetic Tape 4 in the non-interpret sign mode the last twenty words of each of three forty word blocks. Check to see if these words are the same as the 20 words within the one set of keys on each corresponding block of MT3. If not, set SFF1, 2, or 3 depending on which block is not the same.

The significant key to search for is in location KEY_{25-36} (decimal loc. 100).

16.8 In location CT_{35-36} (decimal location 300) is a code number 1, 2 or 3 which was placed there as the result of some previous operation. Depending on the code, do the following output operation.

1; Write 50 words from DATA on MT2 (alphanumeric - ISN) followed by end of block mark

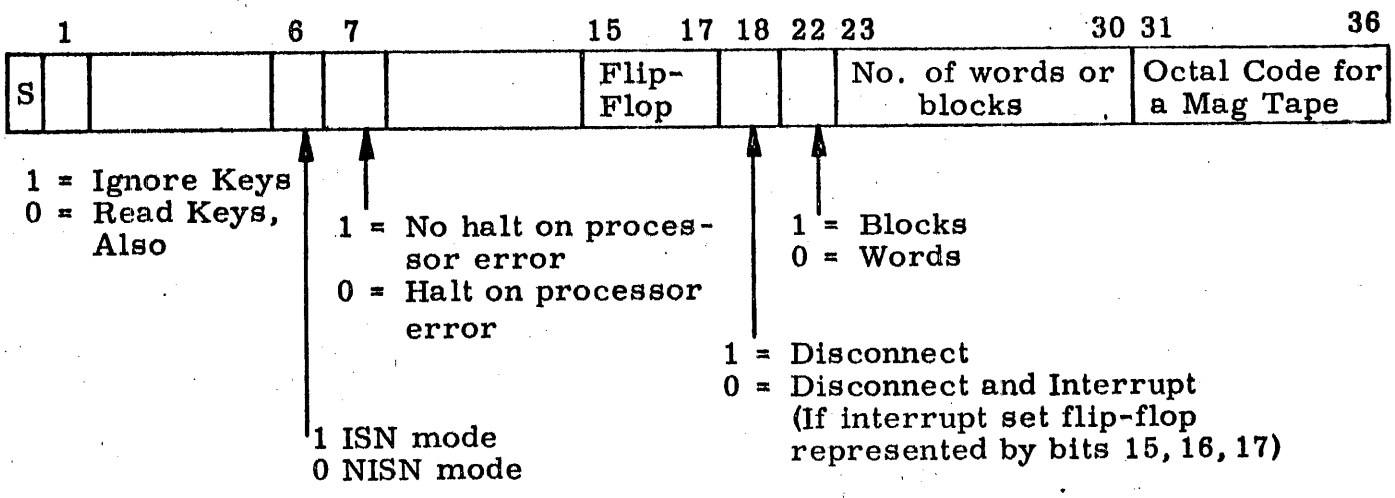
2; Write 100 words from SUM on MT2 (alphanumeric - ISN) followed by end of block mark

3; Write 75 words from DIFF on MT2 (alphanumeric - ISN) followed by end of file mark

Interrupt to zero out count when finished, then rewind the tape. Use processor No. 2.

16.9 Read one block (< 500 words) into loc. DATA and the first ten words of the second block into location SUM, including keys, non-interpret sign, from MT4. Do not allow the processor to stop if an alarm occurs. If it was an Input Parity Error, read again. If it was a Timing Read Error, erase the tape which was just read. Clear all alarms when finished.

16.9.1 In memory location CODE (decimal location 50) is a code word which indicates the following:



Read data from a magnetic tape according to the code word into location DATA and following.

GROUP 17. TRAPPING MODE

17.1 Write a master trapping mode program which resets the activity flip-flop which caused the trap and transfers to a location elsewhere in memory. Arrange it so that the first trap (CVB) transfers to decimal location 100, the second (DVB) to location 105, and each further trap to each subsequent fifth location. Assume only one processor is being used.

17.2 a) In order to check the correctness of the main program, trap every control transfer (except TRC) in locations PROG to PROG +6 inclusive. In the trapping program, print out on paper tape in alphanumeric the contents of a block of 100 words beginning with ANS, and read into ANS and following another group of words from paper tape in alphanumeric until a STOP CODE is encountered. Return control each time from the trapping program to the main program instruction after the transfer order that caused the trap.

b) How could you trap each transfer order (except TRC) and then, upon completion of the trap, actually execute the same transfer order?

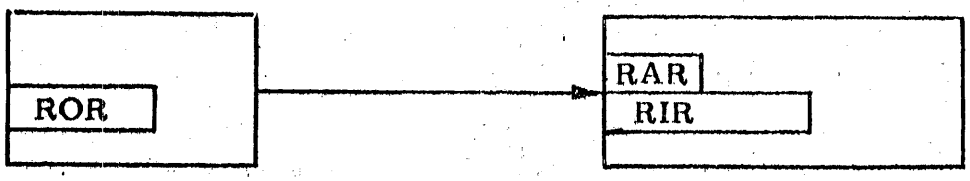
```
PROG    TRU     * +1, 0, 1          Sets TRA—>1
        SEN     DO, O, SFF1
        MOV       -----
        CLA       -----
        RPA       -----
        ADB       -----
        TRN     FIN
        TRU     * +1, 0, 3          Resets TRA —>0
```

17.3 Write information from memory (starting with decimal memory location 100 to decimal memory location 4095) in alphanumeric form in the ISN mode on MT1. If you run out of tape, continue on MT2. Rewind both tapes when finished. Assume only one processor is being used.

GROUP 18. REAL TIME

18.1 It is difficult to see an illustration which will exemplify the RIR register alone, since it is not addressable and the programmer has no control over it.

One example of the use of RIR is when two MOBIDICS are working in collaboration, one feeding information to the other. What is in the output register of one becomes the input register of the other.



The programmer can control only the ROR of MOBIDIC No. 1, and the RAR of MOBIDIC No. 2, which tells the address of where to put the information in RIR.

18.2 Keep track of 1000 words containing parity errors out of a total of 3000 words being read into location BLOCK and following. Double check to see if 1000 words with parity errors have actually been detected and, if not, set sense flip flop 1 and halt. Assume characters are arriving at intervals which allow sufficient programming time, and that the characters are being sent in the Interpret Sign Mode.

18.3 Transmit 500 words starting at location A in the Interpret sign mode. Every time a word is negative, add 1 to a counter in location COUNT. Assume the 501st word contains a control character. Allow it to interrupt the main program.

GROUP 19 CONSOLE

19.1 **Assume the last instruction executed in a particular program was also in the B register.**

a) **Display this on the console in two places.**

b) **Display the address of the next instruction in two places.**

```
19.2      ORG    200
START     LDX    0, 1, 51
          CLA    VALUE, 1
          RPT    948, 2, 1
          MLR    VALUE + 1, 1
          STR    ANS, 1
          TRX    START + 1, 1, 1
          HLT
          ORG    500
VALUE     BSS    1001
ANS       BSS    51
          END    START
```

Assume the data (VALUE to VALUE + 1000) is already entered into the computer. Execute this program by console control and then discuss how the results can be displayed.

GROUP 20 REVIEW

20.1 Add every seventh number from location DATA to DATA + 200 inclusive. Fill in the blank portions of the following instructions:

```
START     CLA    ZERO
          LDX    _____
          ADD    _____
          TRX    _____
```

^{7}

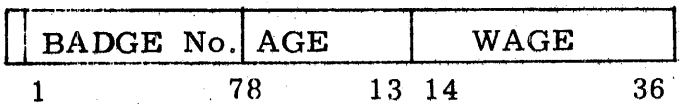
20.2 Assume bits SN, 1, 2, and 3 of the Q register contain ones, and the rest of the Accumulator and Q register contain zeros. After the following program, what bits in the Accumulator and Q register will contain ones?

```
SLL    36
CYS     2
CYL     1
SLL     1
```

20.3 The following program is to extract m from location DUMMY and place it in the Accumulator. Fill in the missing instruction in location START +

```
       OCT    1777000077777
START  CLA    * - 1
       _____________________
       LGM    DUMMY
```

20.4 An electronics firm retires its employees when they reach 65 years of age. The company keeps the records of each of its 200 employees according to badge number, in ascending order, on magnetic tape 1, in one block, the word formation being:



The information on magnetic tape is three years old. Count up the number of employees who have retired within that time, store in location SUM, and print each one's information word on paper tape. Use location COUNT to count up the number of employees to be retired next year. For simplification purposes, assume that for the past three years there has been no change in employee status other than those who have been retired.