Sylvania

===
SEARCH PROGRAMS FOR MOBIDIC B
Davidson, Pizzo, Sammet
Sylvania Electronic Systems & U.S. Army
January 1959

Presented to The Computer Museum
by Prof W F Luebbert, Dartmouth Coll.

SUBJECT: SEARCH PROGRAMS FOR MOBIDIC B

DATE: 21 January 1959

TO: J. Sammet

FROM: D. Davidson, R. Pizzo

Attached is a comprehensive discussion of the MOBIDIC B Search Program problem. It includes related information on MOBIDIC B, the Data Retrieval Unit, and Organization of File Information. Possible solution programs, which have been checked in detail, together with timing information is submitted herewith. Suggested improvements in the coding, which are contingent on the logical design, are also included.

Prepared by D. Davidson
D. Davidson

R. Pizzo
R. Pizzo

Approved by J. Sammet
J. Sammet

CONTENTS

LIST OF ILLUSTRATIONS

LIST OF APPENDIXES

1. INTRODUCTION TO MOBIDIC B

The MOBIDIC B computer differs from MOBIDIC A in several respects. The following is a brief description of these differences.

1.1 Mechanization of Orders

MOBIDIC B performs all the orders of the MOBIDIC A order code, but also performs orders unique to MOBIDIC B. Not all of the orders are mechanized, that is, built into the hardware. When the MOBIDIC B decodes an order which is not mechanized, it transfers control to a stored subroutine which executes the order, and then transfers control back to the program. The "subroutined" orders take longer than the mechanized orders, the time depending on the complexity of the required subroutine. A list of the mechanized orders and their execution times is given in Appendix A.

1.2 Duplex Operation

The MOBIDIC B has two distinct processing units which may operate simultaneously but independently of each other. These processors have access to the same input-output devices and therefore may communicate with each other through magnetic tape. There are two standard MOBIDIC in-out converters which operate in the MOBIDIC B system.

1.3 Mass Memory Unit

In addition to the standard family of MOBIDIC in-out devices, the MOBIDIC B system includes a mass memory unit which is also treated as an in-out device. The mass memory unit consists of a 50-million-bit disc memory. Each disc is separated into numbered tracks. The information in the mass memory is addressable via these track numbers. The control circuits associated with this device present

data to and from the converter busses in a manner similar to magnetic tape data. Information, therefore, can be read and recorded in the mass memory with the same instructions used for magnetic tape.

1.4 Data Retrieval Unit

In order to do open format* searching, an additional piece of hardware, the Data Retrieval Unit (DRU) is required. The DRU monitors the in-out bus from the mass memory or the magnetic tapes to the converter, and selects information for storage in special memory locations. See the discussion of the Open Format Search Program for further description of the operation of the DRU.

2. ORGANIZATION OF INFORMATION IN FILES

2.1 Open Format

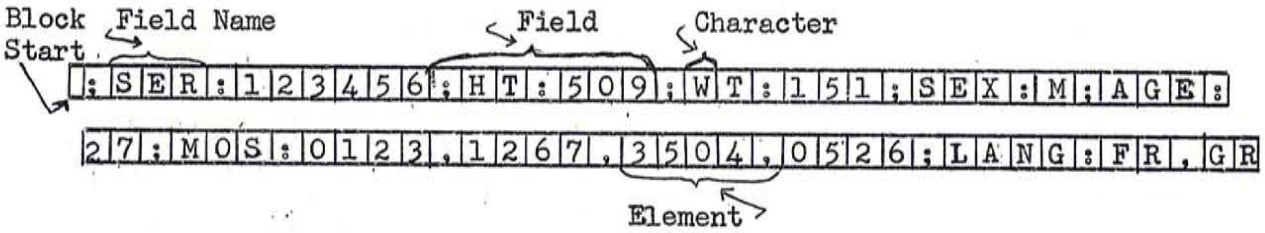
The information which is stored on the magnetic tape or mass memory may be of different types, e.g., personnel information, intelligence information, logistics, or any other data. It is often desired to extract specific information from the file. In order to do this, the data must be organized in some consistent format. The format described below is only one possible scheme; however, it is the arrangement which must be used for MOBIDIC B Open Format Search programs.

The information in the storage unit (mass memory or tapes) is organized in a format specified by the following rules: A tape may consist of one or more files of information. These files are divided into sections called Blocks and are set off by Beginning-of-Block Marks and End-of-Block Marks. Each block contains one,

*See page 20

and only one, complete "set" of information. This "set" is here referred to as a record,* Each record may have any number of fields, but every record in one particular file of information must contain the same number of fields, and these fields must be in the same order within each record. Each field may consist of up to eight elements. The number of elements may vary from field to field. An element may consist of from 1 to 6 alpha-numeric characters, each character being 6 bits in length. (This limits the largest element to the size of one machine word, 36 bits.) Each element must be of uniform length in the field.

Sample Record:



This record has seven fields. Each field has associated with it a field designator or field name, e.g. HT. Notice that the field and elements are separated by special punctuation. This punctuation is the signalling device used to assist the data retrieval unit in selecting the desired information.

2.2 Closed Format

A closed format file is distinguished from an open format file by the following:

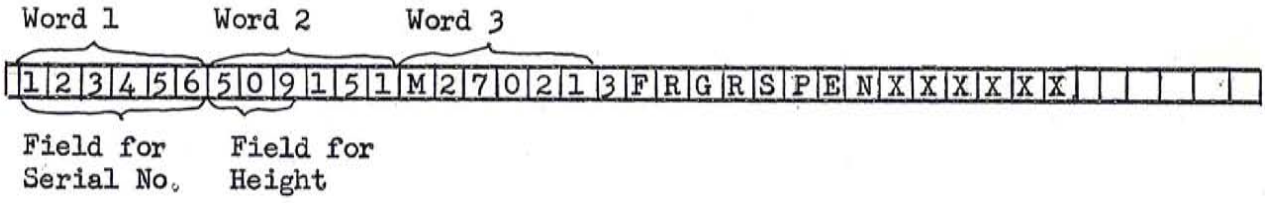
a) The closed format file has one and only one element per field.

---

*Record and block may appear to be synonymous; however, they are entirely different concepts. A block is a physical division of information on a tape or mass memory. It may extend from 1 to n words (where n is the capacity of the tape). A record is a division of information within a file. In other systems any number of records may be contained in one block; the systems described above requires that there be only one record per block.

b) There are no field names or punctuation in the closed format. (There is no necessity for punctuation since the record shape is entirely defined by the format, i.e., closed format records are all exactly the same length in the file.)

Sample Record - Fields corresponding to those in the open format example are indicated.



3. GENERAL CONCEPTS OF SEARCH PROGRAM

The purpose of a search program is to examine particular data and to indicate whether it meets specified criteria. For instance, if the information on mass memory or magnetic tape were a personnel file, we might wish to know the names of all men who were between 5' 9" and 6' 2", who weighed under 200 pounds, who could speak French and German, and who could drive a motor scooter. We could also say that provided he did not speak French and German, we would accept him if he spoke Italian and was an expert on Gorgonzola cheese. We could then reduce this search criterion to the following logical form:

$ABCD + ADC'EF$

where

A - Height between 5' 9" and 6' 2"

B - Weight under 200 pounds

C - Speaks German and French

D - Drives motor scooter

E - Speaks Italian

F - Expert on Gorgonzola cheese

and where plus (+) denotes logical addition (=or), juxtaposition denotes logical multiplication (=and), apostrophe (') denotes logical negation (=not).

The problem now is to write a program which will search for these criteria.* Since we have set up a strict format for the information, the program for any particular search will vary only in the set of search criteria provided. It therefore becomes feasible to develop a program to generate search programs. We do not discuss this "generator" here; we rather assume its existence and consider the type of program it will generate.

The main consideration in the search program is timing. The aim is to accomplish the search while reading from the tape or mass memory at full speed. It is also necessary that the programs be compatible for both tapes and mass memory. The reading speed of the tape is faster, but because the interblock gap on the magnetic tape is great, the block-by-block reading speed of the mass memory is faster. It then follows that the program time is limited by the reading speed of the mass memory.

The general solution method of the MOBIDIC B search programs is as follows: Requests for retrieval of data are entered into either of the processors through one of the standard input devices, such as the paper tape reader. These interrogations must specify (a) the data to be searched, (b) criteria for selection, and (c) the manner of disposition of the selected data. Upon receipt of such an interrogation, a program is formed by the search generator program. The required data is

*The programs shown here are the same as or based on those shown by R. Little in memo N377.42-019-58. See Reference (1).

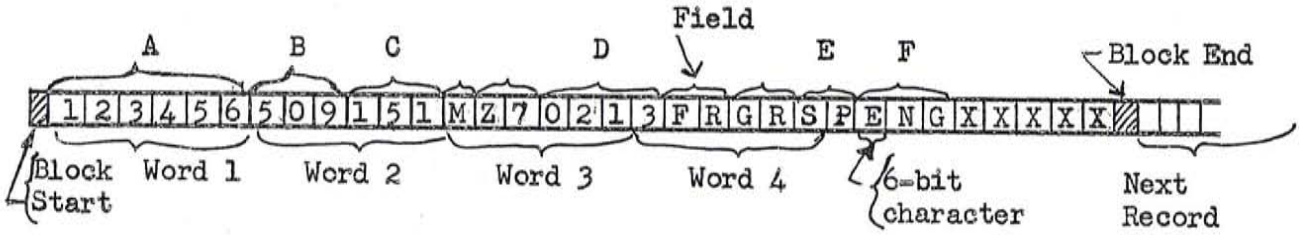
extracted from tapes or mass memory and is then examined at high speeds. Data is selected according to the specified criteria; then the desired action, such as print-out, storage on an output tape, further processing, etc., may be taken.

4. CLOSED FORMAT SEARCH PROGRAM

4.1 General Principles of Program

The closed format search program is a special case of the open format search program; however, an understanding of the closed format search will greatly facilitate comprehension of the open format search.

In the program discussed below, we assume that the files are closed format. The following is a sample record in closed format.



Note that a field may be divided between two words, or may be entirely contained in one word, and that more than one field or parts of fields may occur in one machine word.

If the field were divided between two words, it would be necessary for the program to assemble the field into one word. The generator program would set up the necessary instructions to accomplish this using the given search specifications. Since there is only one element per field in the closed format, the maximum length of the field is 36 bits.

In the closed format, any field may be specified entirely by the location in the record of its first character, and its length. This specification is called the "field address".

4.1.1 Specifications of the Closed Format Search

Up to six field criteria (labeled A, B, C, D, E, F,) may be specified for the search. Each of these is given in terms of the following quantities.

a) Field Address: The field address specifies the location of the first character of the field and the length of the field.

b) Logic: A 36 bit mask must be given for each field. This mask serves a dual purpose. It eliminates the "junk" after the field has been assembled, and it allows one to select only a portion of the field for examination. Bits of the field which correspond to zeros in the mask are ignored; bits of the field which correspond to ones in the mask are considered.

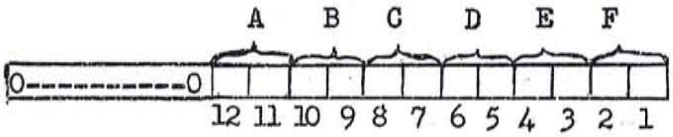
c) Upper and Lower Bounds: The upper and lower bounds define the limits between which the field must fall to satisfy the individual field criteria. A search may be made for a specific value by specifying equal upper and lower bounds.

d) Logical Combinations: A logical sum of up to 6 search criteria, T_i, may be specified. Each criterion may be composed of 6 field specifications* or their inverses. In addition, an "m of n" criterion, T_7 (based on the T_i in use for this problem) may be specified. This means

*See discussion of search specification, Section 3, p 4.

that if m out of n of the conditions in the criterion are satisfied, then the criterion is satisfied.* When specifying T_7, both the criterion and the number m must be given. The number m cannot be larger than the total number of different fields specified in T_1 to T_6.

Each of the T_i which specify the search function is represented by one machine word. The 12 low-order bits of the word are significant; the high order bits must be zeros. These bits are set up in the following manner.



$i = 1,2,\ldots,6$

The two bits corresponding to each of the field criteria (A, B, C...) in each term of the logical sum are coded with the following meaning.

10 = yes = x (where x is the field criterion)

01 = no = x'

00 = don't care

For each record which is examined, a "Tag" word is set up which corresponds in make-up to the T_i. In this way the selected data from each record may be completely described by 12 bits, and may be conviently compared to each T_i.

---

*If the m of n criterion were AB'C'D and m = 3, then if the record satisfied the logical sum AB'C + AB'D + B'C'D + AC'D, then the m of n criterion would be satisfied.

4.1.2 Search Program Procedure

Each element or field to be processed is assembled into one computer word (36 bits) and modified by the mask given in the search specification. This is done to eliminate portions of the field that are not of interest. The assembled element is then compared with the upper and lower bounds (UB_i and LB_i) associated with the field. If a given element falls within the bounds, a binary-coded "yes" is stored in the proper position of a "Tag" word. This "Tag" word is used to facilitate the coding of the program. If an element does not lie with the bounds, a binary-coded "no" is stored in the Tag word. When all the fields have been interrogated, and the Tag word set up, it is compared with each T_i (T_1 to T_6) given by the search specification. If the Tag word is found to be equal to any one of the criteria, the record fits the search specification and may be written on the output tape if required. If an "m of n" criterion has been specified the count of coincidences with the m of n criterion is checked. If the count is greater than or equal to m, then the record meets the specification. Only records which do not meet one of the other criteria is checked against the m of n criterion. Records that do not fit the specification are left in memory to be replaced by a new record.

A generalized program for the closed format search is given in the next section. A sample program using

specific search criteria is given in Appendix B. A macro flow chart of the closed format search program is given in Figure 1. A micro flow chart is given in Figure 2.

4.1.3 Program

The closed format search program is written relative to X, the location of its first instruction. Two record input areas, A_1 and A_2, are specified in memory so that processing of one can be accomplished while the other is being filled by the input device (magnetic tape or mass memory). The program is initiated by giving two RAN orders in succession. This procedure will hold up computation until the first record has filled area A_1 and the reading of the search record has commenced. The blank addresses are filled in by the generator program.

X	RAN	A_1
X + 1	RAN	A_2

As soon as input area A_1 has been filled, processing begins. The first field is assembled in the accumulator into one machine word.

X + 2	CLA	Pick up word containing first part of field.
X + 3	SHL	Shift left so that first character of field occupies bit positions 31-36.

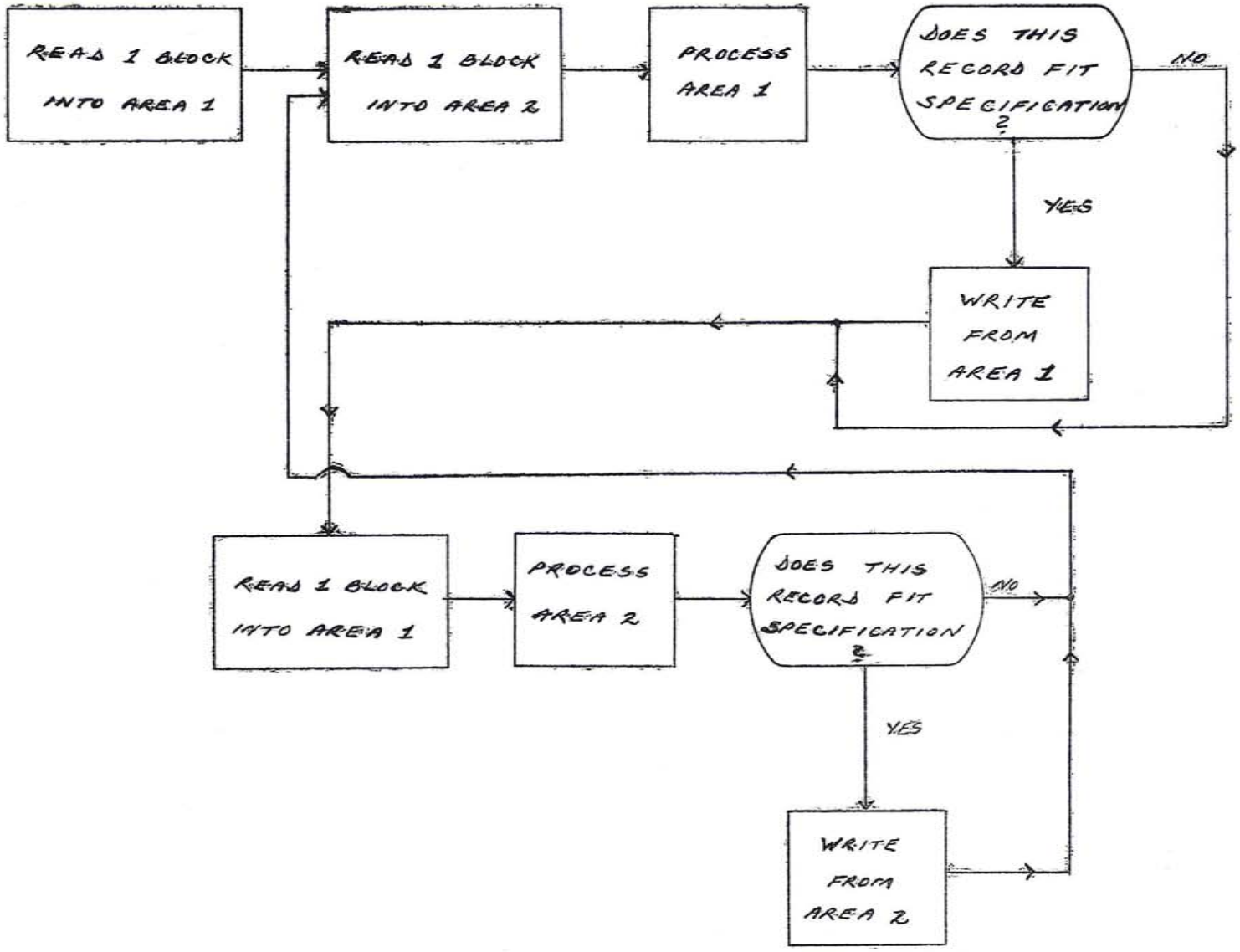


FIGURE 1

GENERAL DIAGRAM OF SEARCH PROGRAMS

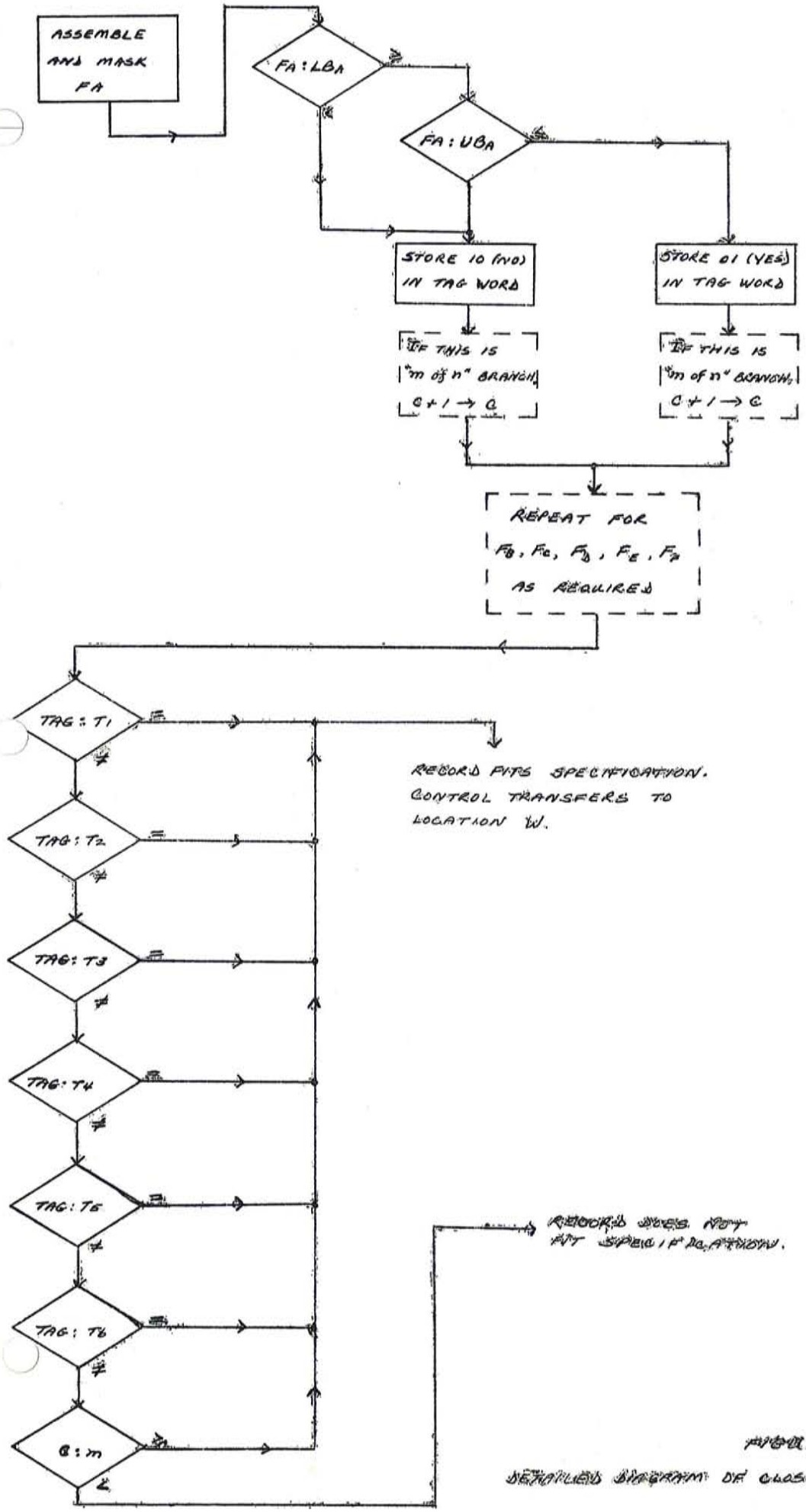


FIGURE 2

DETAILED DIAGRAM OF CLOSED FORMAT SEARCH PROGRAM

X + 4	STR	Comon	Store partial field in temporary storage.
X + 5	CLA		Second part of field in accumulator.
X + 6	SHR		Line up second part of field with first part of field. (Number of right shifts = 36 minus number of left shifts.)
X + 7	LGA	Comon	"OR" first part of field from temporary storage to form complete field.
X + 8	LGM	Mask	Modify field with mask given in specifications to remove "garbage", if any, at low order end of word, and to eliminate portions of field we are not interested in.

The field is now completely assembled in the accumulator. Notice that if the field is completely located within one word in the record, instructions X + 2 through X + 8 can be replaced by

X + 2	CLA	Pick up word containing field.
X + 3	LGM	Modify field with mask.

Notice that in the latter case it is not necessary to shift the field since the bounds with which it is to be compared can be positioned to coincide properly.

The assembled field is compared with the upper and lower bounds associated with it. The conditional transfers of this part of the program are set up by the search assembler (generator) to provide a "yes" or "no" for fields lying within the bounds or outside the bounds as determined by the search specification.

X + 9	SUB	LB	Subtract lower bound associated with the field.
X + 10	TRN	Z	A transfer of control at this point indicates the field is less than the lower bound.
X + 11	SUB	R	If the field was shown by X + 10 to be greater than or equal to the lower bound, a number R is subtracted. (R = UB - LB + 1).
X + 12	TRP	Z	A transfer of control indicates the field is greater than the upper bound associated with it. Control goes to X + 13 if and only if the field lies within the bounds.

At this point the operation of the program depends upon the search specifications. For example, a transfer to Z may require that we store "yes" in the tag and count one on the m of n counter, or we may require to store "no" in the tag and count one on the m of n counter.

Rather than show the programs for the four possible branches, a typical pair of branches is programmed below.

```
X + 13    BSLA    TAG, 2, 1     Stores "Yes" in tag.
```

The BSLA instruction places the tag in the accumulator, shifts it two places to the left, inserts 01 (the binary code for "yes"), and stores the modified tag back in memory. The binary codes for "yes" and "no" to form the "TAG" word are complimentary to the "yes" and "no" conditions for the search criteria words (T_i), so that when the "TAG" and the T_i are "anded" together, only cases that fit the search criteria will go to zero.

The program starting at X + 14 examines the other fields of the search specifications and sets up the corresponding tag for each field.

```
Z + 1     BSLA    TAG, 2, 2     Store "no" in tag.
Z + 1     BTRX    X + 14, 1     Subtract 1 from Index Register 1 which was initialized to m + 1 by the search assembly program (generator), and return control to main sequence. This instruction serves to count down on m for the m of n criterion.
```

When all fields have been examined and the tag has been formed, it must be compared with each of the criteria.

Y	LGM	T_1	"and" tag with first criteria mask.
Y + 1	TRZ	W	If logical product of mask and tag is zero, the record fits given specifications. If not, examine next criterion, if any.
Y + 2	CLA	TAG	Place tag in accumulator.
Y + 3	LGM	T_2	"and" tag with second criterion mask.
Y + 4	TRZ	W	

This process is repeated for each of up to six criteria. A transfer to W at any point indicates the record fits the specifications. The disposition of the record at location W is arbitrary, see Figure 2. If no transfer to W occurs, the record does not meet any of the primary criteria. If an m of n criteria has been specified, it must be tested for satisfaction.

Y + 5	BTRX	V, 1	Index Register 1 was reduced by one each time a specification of the m of n criterion was met.
W	TRU		If this count was equal to m, the program would transfer to W (next instruction in sequence because I.R.1 is zero). Control would transfer to V if I.R.1 was not zero.

If the record does not fit the specification, a test for end of file is made and control returns to the search program for processing input area A_2.

V	LOD	Set IR1 to m+1.
V + 1	BTRX	Test for end of file. If end of file condition is not met, control is transferred to program that will process input area A_2.

4.2 Timing

The amount of time required for searching a record depends upon the following variables:

a) Number of fields to be examined.

b) Number of fields lying in two machine words as opposed to fields lying entirely in one machine word.

c) Number of logical combinations of criteria which satisfy the search.

The longest search occurs when there are six fields to be examined in each record; each field occupies two machine words, and the search specification consists of six logical combinations plus an m of n criterion.

For the closed format search, the maximum possible processing time in micro seconds is given by the formula:

$$70\ F_1 + 270\ F_2 + 226\ F + (38\ F + 78)M + \lceil 70 + 104(C-1) \rceil + 64$$

where

F_1 = Number of fields contained completely in one word.

F_2 = Number of fields divided between two words.

$F = F_1 + F_2$ = Total Fields.

C = Number of primary criteria.

$$M = \begin{cases} 1 \text{ if m of n criterion is specified.} \\ 0 \text{ if m of n criterion is not specified.} \end{cases}$$

Each term in the equation is obtained from the appropriate part of the program. The matching is shown below and the derivation of the coefficients can be checked by referring to Appendix A.

$70\ F_1$	CLA, LGM (X + 2 and X + 3)
$270\ F_2$	Assemble Field (X + 2 through X + 8)
$226\ F$	Test for Bounds (X + 9 through X + 12 plus X + 13, or Z + 1 and Z + 2)
$(38F + 78)M$	1 BTRX for each field (U). 1 BTRX to test for end of file. 1 LOD to set index register to m±1 (V and V + 1)
$\lceil 70 + 104(C-1) \rceil$	LGM, TRZ plus CLA, LGM, TRZ (Y through Y + 4)
64	RAN (X + 1) and TRU (W)

Simplifying the above equation,

$$T = 296\ F_1 + 496\ F_2 + (38\ F + 78)M + 104(C-1) + 134$$

The time available for closed format search processing while reading from 28 KC zone of the Mass Memory (worst case) is .178W m sec. where W is the number of words in a record.

See Section 7, Figures 5 and 6, for graphs of closed format timing.

4.3 Suggested Changes to Improve Time of Search Program

4.3.1 Mechanization of Shift Left Long

If SLL were mechanized, the sequence shown below could be used in place of steps X + 2 through X + 6 in the closed format search program. This would give a saving of 15.2% in the overall program (94 μsec per field) and a saving of 3 locations per field.

```
X + 2    CLA    WORD
         LOD    WORD + 1,,Q
         SLL    **
```

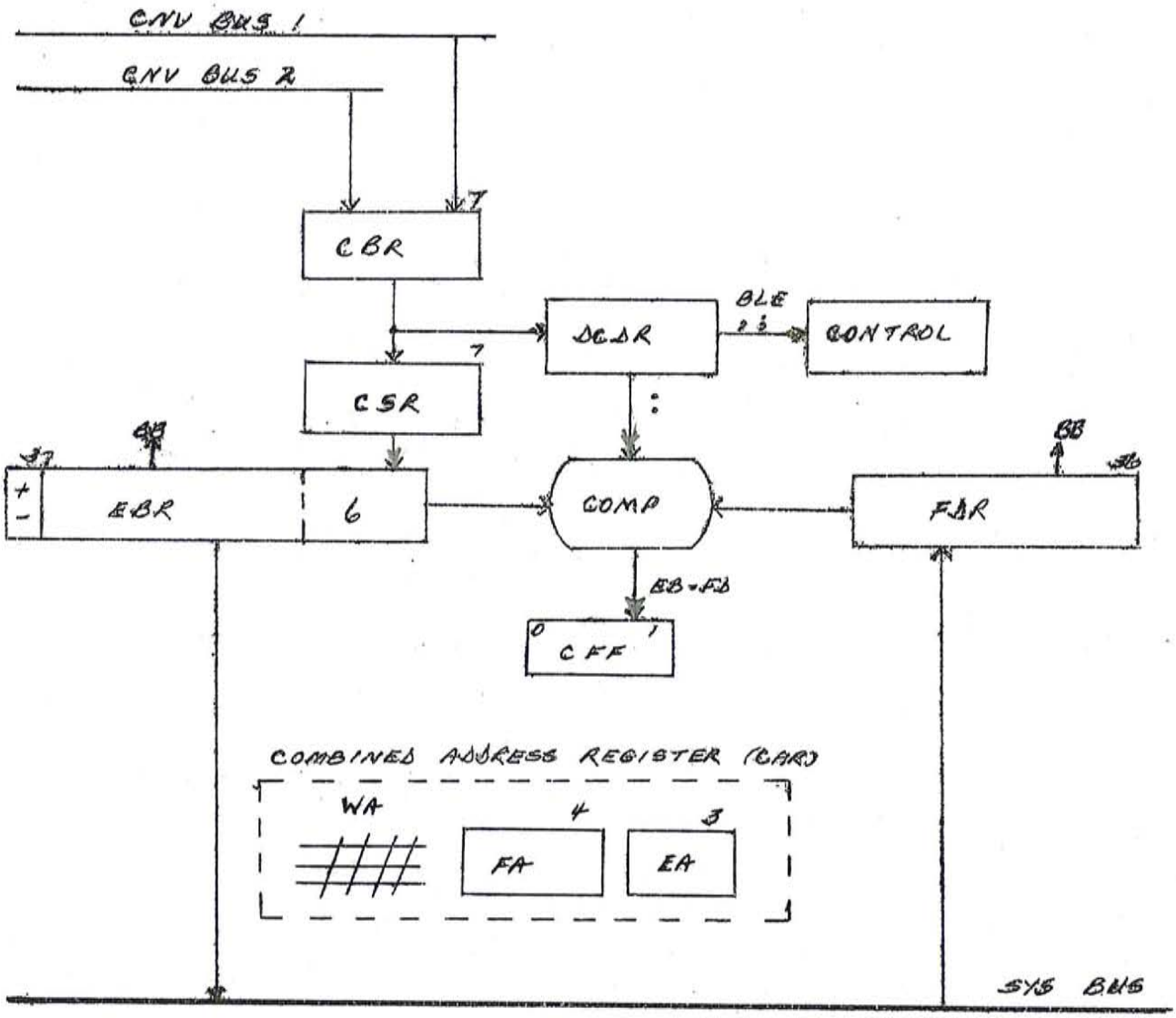
4.3.2 Use of BTRX Order

Note that the method of handling the count for the m of n criterion differs from that of memo No. N377.42-019-58. By using the BTRX instruction both to count down on "m" and to test for satisfaction of the m of n criterion, a saving of 100 μsec. and 4 locations is accomplished. This change requires no modification in the mechanization and was therefore included in the present search program.

5. DESCRIPTION OF DATA RETRIEVAL UNIT

The operation of the DRU is described by the flow chart of Figure 3. To initiate operation of the DRU, the programmer must give a BSCH order. This sends an RAN instruction to the converter and starts a magnetic tape or the mass memory. Before giving the BSCH instruction, the Mass Memory Address Register, MAR, must be loaded with the track number and block number of the first record.

Prior to the beginning of the search, the first field designator will be loaded into the Field Designator Register, FDR. The search then commences. Information being transferred over the converter busses is also loaded into the Character Buffer Register, CBR. The information from CBR is then transferred and shifted in the Element Buffer Register, EBR, to assemble full words. Upon the appearance of the symbol signifying end-of-field designator, which we have assumed to be a colon, a comparison is made between the contents of the EBR and FDR. If these two registers do not match, the desired field designator has not yet been located; the steps mentioned above are repeated for each successive field designator appearing on the record. When finally, a field designator appears in EBR which matches the contents of FDR, the desired



CAR - COMBINED ADDRESS REGISTER
CBR - CHARACTER BUFFER REGISTER
EBR - ELEMENT BUFFER REGISTER
FDR - FIELD DESIGNATOR REGISTER
DCDR - SYMBOL DECODER
CFF - COMPARISON FLIP-FLOP

FA - FIELD ADDRESS COUNTER
EA - ELEMENT ADDRESS COUNTER
WA - WIRED-IN ADDRESS COUNTER
COMP - COMPARISON CIRCUIT

FIGURE 2

DATA RETRIEVAL UNIT

field has been located. Now, as the succeeding characters arrive in EBR, they are shifted left again as before until either a comma or a semi-colon appears in CBR. A comma or semi-colon detected by the symbol decoder will cause the contents of EBR to be transferred to memory at an address specified by the Element Address, EA, and Field Address, FA, counters. These two counters together constitute the address of one of the 48 buffer locations in memory reserved for storage of up to 6 fields per record and up to 8 elements per field. If a semi-colon had appeared in CBR, this would have indicated the last element of a field. Accordingly, the sign bit of the EB Register is made negative and the last element in the field is stored having a negative sign. Following the transfer of the contents of EBR to the processor memory, the FA Register is indexed; and the next field designator is loaded from the processor memories to the FD Register. This sequence of operations continues until all elements of all selected fields have been loaded to the processor memory. The DRU signal is terminated by the appearance of a block end character in the CB Register.

For a more detailed description of the DRU, see the USASEL Mobile Digital Computer, MOBIDIC B, First Quarterly Report, Reference (2).

6. OPEN FORMAT SEARCH PROGRAM

The open format search program is written relative to X, the location of its first instruction. Two record input areas, A_1 and A_2, are specified in memory so that processing of one area can proceed while the other is being filled with new data. The program is initiated by giving two BSCH orders in succession. The BSCH instruction (MOBIDIC B Search) starts the Data Retriever Unit (DRU) and inserts the order "RAN" into the converter. This causes one entire record to be read into the memory location specified by the α of the BSCH instruction; it also causes the specified elements of the record to be extracted and placed into special predetermined memory.

6.1 Program

A flow chart of the open format search program appears in Figure 4.

X	BSCH	A_1	
X + 1	BSCH	A_2	
X + 2	CLA		Place element in Accumulator. The location of the first element of the field is predetermined.
X + 3	TRN	J + 1	Transfer if this is last element of field.
X + 4	LGM	Mask	Reduce field if desired.
X + 5	SUB	LB	Subtract lower bound associated with field.
X + 6	TRN	X + 11	A transfer of control at this point indicates the element is less than the lower bound.
X + 7	SUB	R	R = UB - LB + 1 If the element was shown by X + 6 to be greater than or equal to the lower bound, subtract R to check on the upper bound.
X + 8	TRP	X + 11	A transfer to X + 11 at this point indicates the element is greater than the upper bound. Control goes to X + 9 if and only if the element lies within the bounds.

A transfer to X + 11 would indicate that the element examined did not lie within the bounds. Unlike the closed format search program, we cannot at this time place a binary coded "NO" in the "TAG" word, since we have not yet interrogated the last element of the field. Once a "yes" condition is obtained it is not necessary to examine the remaining elements in the field and we would transfer to Location L where the elements of the next field would be examined.

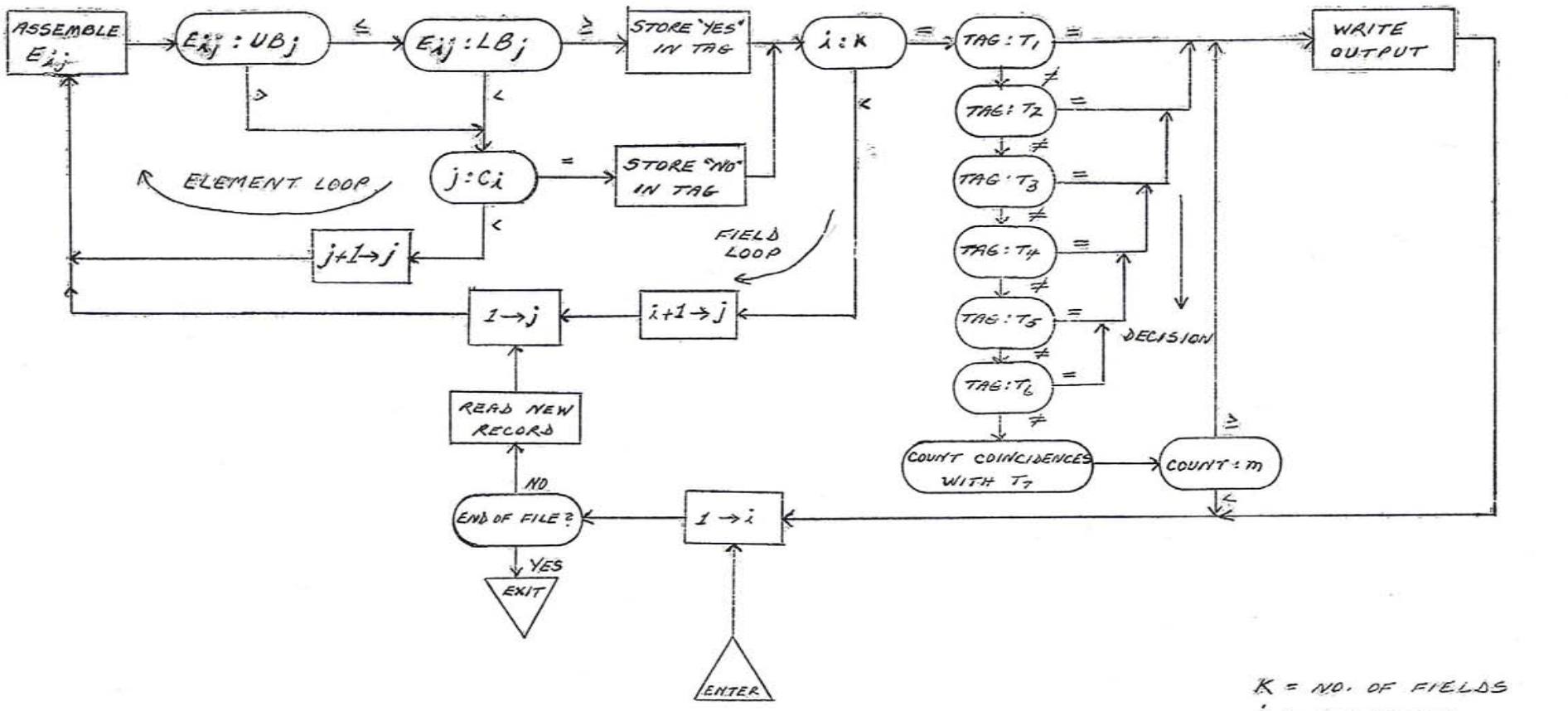


FIGURE 4

SEARCH PROGRAM (OPEN FORMAT RETRIEVAL)

X + 9	BSLA	TAG, 2, 1	Store "YES" in "TAG" word.
X + 10	BTRX	L, 1	Subtract 1 from Index Register which was initialized to m + 1 (by "generator") and transfer control to a program (location L) that examines the elements of the second field. If no m of n criterion were specified, this would be replaced by 2 TRU instructions.
X + 11	CLA		Place second element of first field in Accumulator.

The program starting at X + 11 continues to examine this field until an element is found which lies between the bounds. This set of instructions is identical with instructions X + 2 through X + 10. In order to save time, a loop is not used.

A transfer to location J + 1 at instruction X + 3 would indicate that the last element of the field under consideration is to be interrogated. The following is the set of instructions that would accomplish this.

J	CLA		Place last element of field in Accumulator.
J + 1	LGM	Mask	Modify with mask associated with field.
J + 2	SUB	LB	Compare element with bounds.
J + 3	TRN	JNO	
J + 4	SUB	R	

```
J + 5   TRP    JNO
J + 6   BTRX   X + 1, 1      Count m on m of n counter.
J + 7   BSLA   TAG, 2, 1     Store "YES" in "TAG" word.
L                            Begin program to examine second field.
```

If a transfer to JNO occurs at this time, it indicates that the elements of the field did not meet the specifications and thus a binary coded "NO" may be put into the "TAG" word. When we have done this, we are ready to examine the second field. The address of the first element of the second field is known and the coding is identical with the above program.

```
JNO     BSLA   TAG, 2, 2     Store "NO" in "TAG" word.
TRU     L
```

When all fields have been examined and the "TAG" word formed, it must be compared with each of the criteria (T_i). This portion of the program is identical to the closed format search program and the reader is referred to Page 16, Instruction Y, (Closed Format Program).

6.2 Timing

For the open format search program, the maximum processing time in micro seconds is given by the formula described below.

$$256(E-F) + 222\,F + 74\,F + 104(C-1) + 70 + (38F + 38)M + 28$$

$$\therefore\ T = 256\,E + 40\,F + 104(C-1) + 38(F + 1)M + 98$$

where

F = Number of fields to be searched.

E = Total number of elements in all fields searched.

C = Number of primary criteria.

$$M = \begin{cases} 1 \text{ if m of n criteria is specified.} \\ 0 \text{ if m of n criteria is not specified.} \end{cases}$$

Identification of terms in relation to search program:

256(E - F)	Element loop for all but last element. (X + 2 through X + 8)
222 F	Last element for each field. (J through J + 5)
74 F	To establish TAG and transfer. (JANO)
104(C - 1)	Criteria test for T_2 - T_6.
70	T_1 criteria test.
(38 F + 38)M	Test m of n condition. BTRX per field and BTRX to test of m of n.
28	BSCH instruction to start search.

The time available for the open format search processing while reading from the 28 KC Zone of Mass Memory (worst case) is derived from the following:

$$T = .216\ W - .038(F + E + W)\ \text{m sec.}$$

where

.216 m sec = Time required to read one word from Mass Memory.

.038 m sec = Time necessary for one memory access. (One memory access is needed for each field designator, each element, and every word in the record.)

W = Number of words in a record.

Simplifying the above equation,

$$T = .178\ W - .038(F + E)\ \text{m sec.}$$

See Section 7, Figure 7, for graph of open format timing.

7. TIMING GRAPHS

The following charts show capabilities of the closed and open format search programs while reading from the 28 KC zone of mass memory. Note that only one case where the m of n criteria was specified is shown on the graphs.

Closed Format
All Fields in One Word

28Kc Zone of Mass Memory

W = words/record

R = Equivalent Read Time for record

C = Number of Criteria

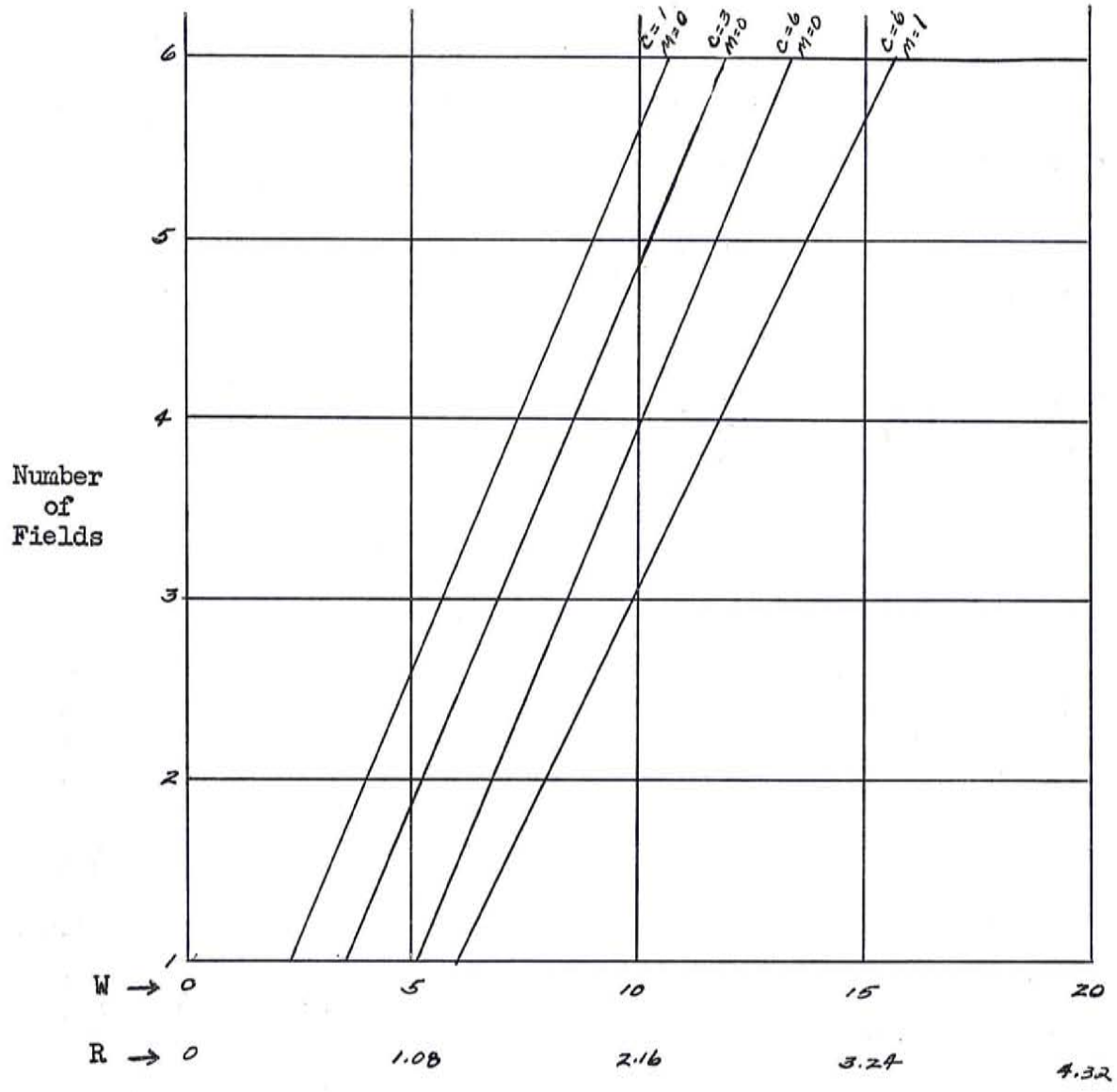


Figure 5 - Timing for Closed Format Search; Representation of Smallest Record Required to Remain Input-Limited

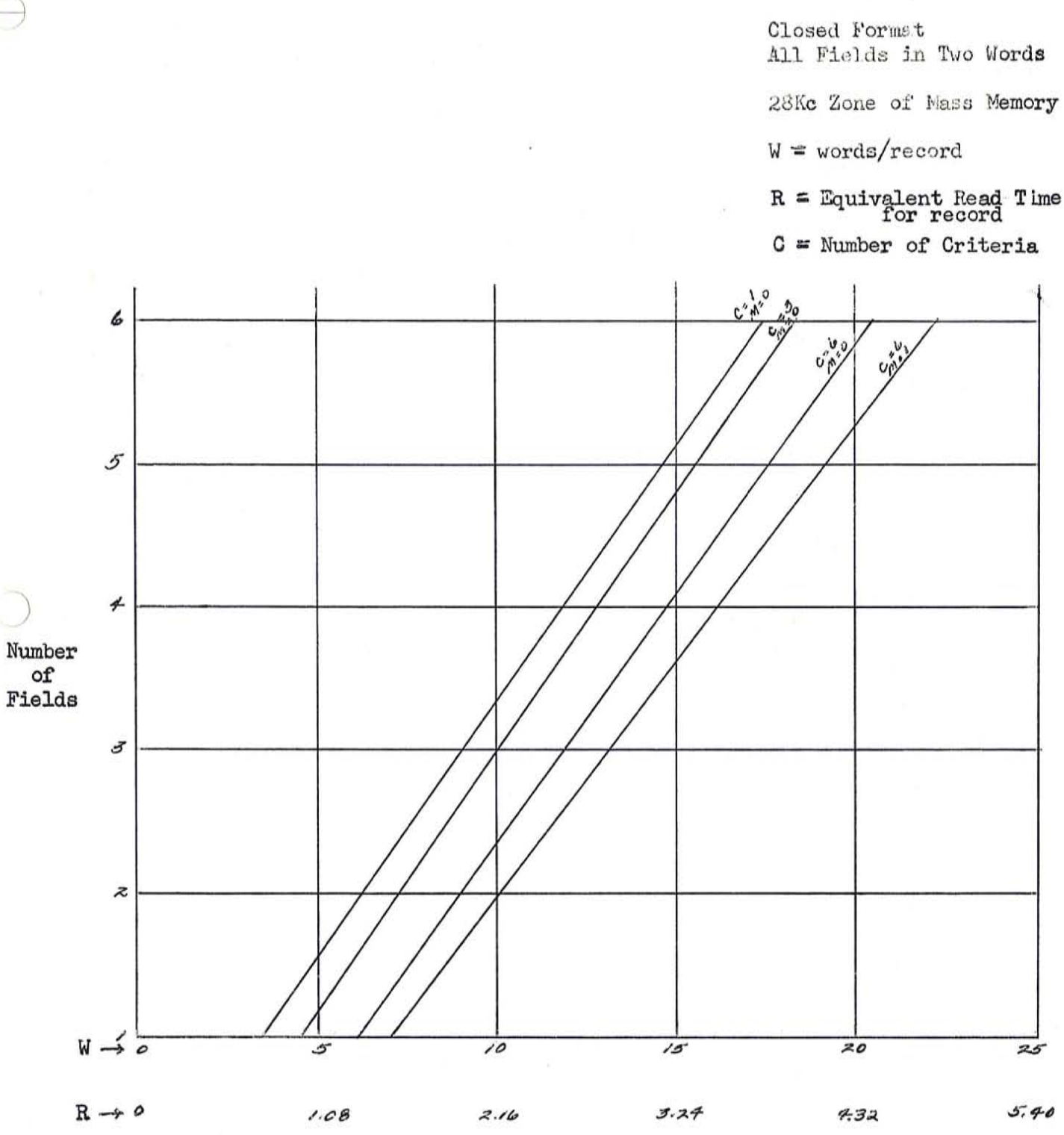


Figure 6 - Timing for Closed Format Search & Representation of Smallest Record Required to Remain Input-Limited

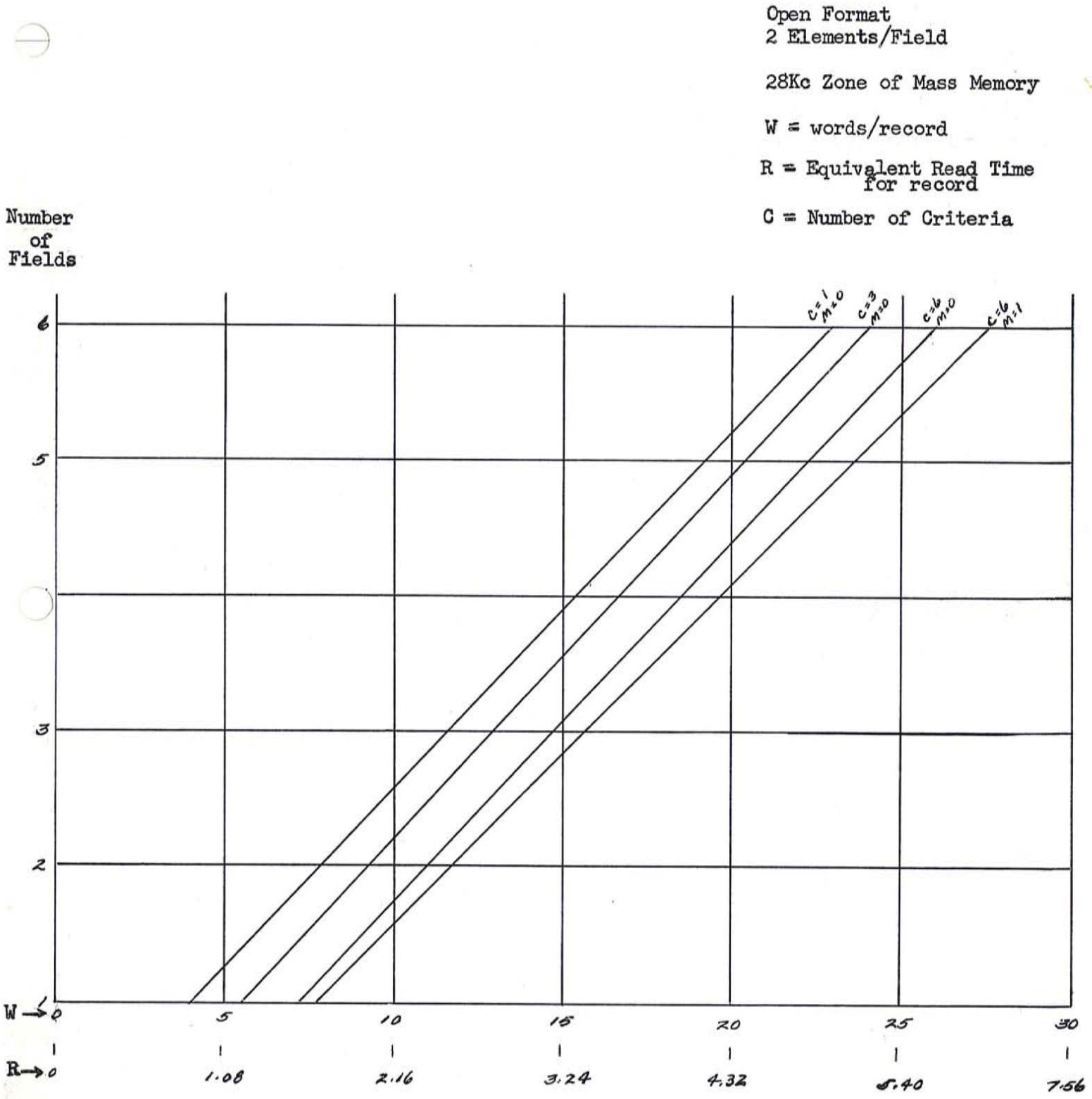


Figure 7 - Timing for Open Format Search : Representation of Smallest Record Required to Remain Input-Limited

APPENDIX A

MOBIDIC B Mechanized Orders

OP	ORDER	TIME	OP	ORDER	TIME
00	HALT	24 μsec	50	STR	34 μsec
02	LGM	36	51	LOD	36/40
03	LGA	36	52	MOV	36/40
04	LGN	34	60	BSCH*	28
10	CLA	34	61	BTRX*	38
11	CAM	34	62	BSLA*	30-38
12	ADD	42	05	SEN	26
13	ADM	42	06	SNS	26
14	CLS	34	07	SNR	26
15	CSM	34	66	SKP	20
16	SUB	42	67	BSP	28
17	SBM	42	70	RAN	28
30	SHL	30-66	71	RRV	28
32	SHR	30-66	72	ROK	28
40	TRU	36	74	WAN	28
44	TRP	26/34	75	WWA	28
45	TRZ	26/34	76	WOK	28
46	TRN	26/34	77	RWD	28

---

* BSCH - Starts the DRU and inserts the order "RAN" into the converter. Bits 30-1 are used as in the order "RAN".

BTRX - Subtracts one from index register specified by gamma (γ). If I_γ is not zero, insert α into the program counter. If I_γ is zero, go on to the next instruction.

BSLA - Alpha (α) specifies the operand. Shifts operand left the number of places specified by Bits 21-19; logically adds Bits 27-22 to the least significant bits of the operand. Bits 16-18 control overflow.

APPENDIX B

Sample Problem: MOBIDIC B Closed Format Search Program

Search Specification: $\overbrace{A\ B'\ C'\ D}^{T_1} + \overbrace{B\ C\ D'\ F'}^{T_2} + \overbrace{A'\ C\ D\ E}^{T_3} + \overbrace{C\ D\ E'\ F'}^{T_4}$

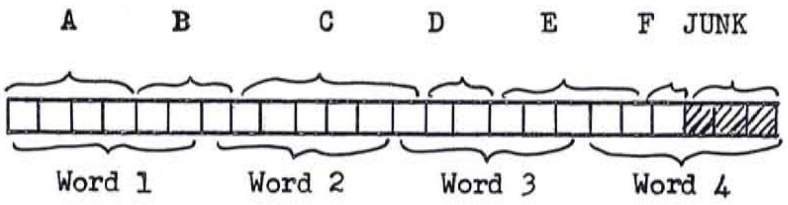
m of n criterion: m = 4, A'B'CDE'

The complete search specification is:

$$\overbrace{A\ B'\ C'\ D}^{T_1} + \overbrace{B\ C\ D'\ F'}^{T_2} + \overbrace{A'\ C\ D\ E}^{T_3} + \overbrace{C\ D\ E'\ F'}^{T_4} + \overbrace{A'\ B'\ C\ D + A'\ B'\ C\ E}^{T_7}$$

$$+ \overbrace{A'\ C\ D\ E' + B'\ C\ D\ E' + A'\ B'\ D\ E'}^{T_7}$$

	Field Designation	UB	IB	MASK
A	1, 4			77 77 77 77 00 00$_8$
B	5, 3			77 77 77 00 00 00$_8$
C	8, 6			77 77 77 77 77 77$_8$
D	14, 2			00 77 77 00 00 00$_8$
E	16, 5			77 77 77 77 77 00$_8$
F	21, 1			00 00 77 00 00 00$_8$



(Each division specifies one character.)

The mask for the TAG is coded with the following meaning:

01 = "yes" = X

10 = "no" = X'

00 = Don't Care

The mask for the criterion (T_1 - T_6) is coded complementary to the TAG so that on "anding" the TAG and the criterion, only the cases that fit the criterion will go to zero.

10 = "yes" = X

01 = "no" = X

00 = Don't Care

All initialization, positioning of the masks and the upper and lower bounds for most efficient programming is assumed to have been done by the generator program.

LOC	OP	Variable	Comments
	RAN	A_1	Read information into input.
	RAN	A_2	areas A_1 and A_2.
A	CLA	Word 1	Pick up word containing Field A.
	LGM	Mask A	Modify Field A to remove "garbage".
	SUB	LBA	Subtract lower bound.
	TRN	NA	Transfer to NA indicates field<LBA.
	SUB	RA	RA = UB - LB + 1.
	TRP	NA	Transfer to NA indicates field>UBA.
	BSLA	TAG, 2, 1	Store "yes" in TAG.
B	CLA	Word 1	Pick up word containing Field B.
	SHL	24	Shift 1st character of Field B to position 31-36.
	STR	TEMP	Temporary storage for first part of field.
	CLA	Word 2	Pick up word containing 2nd part of field.
	SHR	12	Number of right shifts = 36 - number of left shifts.
	LGA	TEMP	"or" first part of field from temporary storage to form complete field.

LOC	OP	Variable	Comments
	LGM	Mask B	Modify field with mask given in search specifications to remove "garbage", if any, from low order end of word.
	SUB	LBB	
	TRN	NB	
	SUB	RB	RB = UBB - LBB + 1
	TRP	NB	
	BSLA	TAG, 2, 1	Store "yes" in TAG.
C	CLA	Word 2	
	SHL	6	
	STR	TEMP	
	CLA	Word 3	
	SHR	30	
	LGA	TEMP	
	LGM	Mask C	
	SUB	LBC	
	TRN	NC	
	SUB	RC	RC = UBC - LBC + 1.
	TRP	NC	
	BTRX	* + 1, 1	Tally m
	BSLA	TAG, 2, 1	Store "yes" in TAG.
D	CLA	Word 3	
	LGM	Mask D	
	SUB	LBD	
	TRN	ND	
	SUB	RD	RD = UBD - LBD + 1
	TRP	ND	

```
LOC     OP      Variable        Comments
        BTRX    * + 1, 1        Tally m
        BSLA    TAG, 2, 1       Store "yes" in TAG
E       CLA     Word 3
        SHL     18
        STR     TEMP
        CLA     Word 4
        SHR     18
        LGA     TEMP
        LGM     Mask E
        SUB     LBE
        TRN     NE
        SUB     RE              RE = UBE - LBE + 1
        TRP     NE
        BSLA    TAG, 2, 1       Store "yes" in TAG
F       CLA     Word 4
        LGM     Mask F
        SUB     LBF
        TRN     NF
        SUB     RF              RF = UBF - LBF + 1
        TRP     NF
        BSLA    TAG, 2, 1       Store "yes" in TAG
TEST    LGM     T1              "and" tag with first criterion mask
        TRZ     Fit             Logical product = 0 if record fits specifica-
                                tions and control is transferred to location
                                Fit
        CLA     TAG             Place TAG in accumulator
        LGM     T2
```

LOC	OP	Variable	Comments
	TRZ	Fit	
	CLA	TAG	
	LGM	T3	
	TRZ	Fit	
	CLA	TAG	
	LGM	T4	
	TRZ	Fit	
	BTRX	NOFit Exit, 1	If index register equals 1 or 0, record fits m of n criterion and next instruction is taken in sequence. If index register is greater than 1, record does not fit m of n criterion and control is transferred to location NOFit.
Fit	TRU	———	
NA	BTRX	NA + 1, 1	Tally m. (Index Register initially set to M + 1)
	BSLA	TAG, 2, 2	Store "no" in TAG
	TRU	B	Return control to B
NB	BTRX	NB + 1, 1	Tally m
	BSLA	TAG, 2, 2	Store "no" in TAG
	TRU	C	
NC	BSLA	TAG, 2, 2	Store "no" in TAG
	TRU	D	
ND	BSLA	TAG, 2, 2	Store "no" in TAG
	TRU	E	
NE	BTRX	NE + 1, 1	Tally m
	BSLA	TAG, 2, 3	Store "no" in TAG
	TRU	F	

```
LOC     OP      Variable        Comments
NF      BSLA    TAG, 2, 2       Store "no" in TAG
        TRU     TEST

MASK A  OCT     77 77 77 77 00 00
MASK B  OCT     77 77 77 00 00 00
MASK C  OCT     77 77 77 77 77 77
MASK D  OCT     00 77 77 00 00 00
MASK E  OCT     77 77 77 77 77 00
MASK F  OCT     00 00 77 00 00 00
TEMP
TAG
```

			Binary Representation of low order bits
T1	OCT	00 00 00 00 45 40	10 01 01 10 00 00
T2	OCT	00 00 00 00 12 21	00 10 10 01 00 01
T3	OCT	00 00 00 00 22 50	01 00 10 10 10 00
T4	OCT	00 00 00 00 02 45	00 00 10 10 01 01

```
LBA                     Lower bound associated with:  Field A
LBB                                                   Field B
LBC                                                   Field C
LBD                                                   Field D
LBE                                                   Field E
LBF                                                   Field F

RA                      (Upper bound - lower bound + 1)for:  Field A
RB                                                           Field B
RC                                                           Field C
RD                                                           Field D
RE                                                           Field E
RF                                                           Field F
```

REFERENCES

1. Memo Number N377.42-019-08, MOBIDIC B Search Programs, R. Little, Sylvania Electronic Systems, Needham, Mass., 8 October 1958.

2. USASEL Mobile Digital Computer, MOBIDIC B, First Quarterly Progress Report, 1 July 1958 to 30 September 1958, Contract #DA36-039SC-78050, Sylvania Electronic Systems, Needham, Mass.

3. Proposal for MOBIDIC B Data Processing System, 21 April 1958, Sylvania Electric Products, Inc., Waltham, Mass.

DISTRIBUTION LIST

1 & 3 P & S Files
5. M. Abbott
6. E. Armstrong
7. A. Ashley
8. G. Bates
9. W. Benson
10. M. Berberian
11. R. Bryant
12. S. Butcher (4)
16. M. Cerier (5)
21. S. Chao
22. E. Cohler
23. R. Conners
24. M. Crystal
25. E. Cuba
26. D. Davidson
27. K. Drew
28. V. Ellins
29. C. Engberg
30. H. Graham
31. R. Gregory
32. H. Hajko
33. F. Hajjar
34. C. Hammer
35. R. Handy
36. M. Haven
37. W. Humphrey
38. E. Jervis
39. J. Kearns
40. M. King
41. R. Kingsbury
42. J. Lee
43. R. Little
44. J. Moriarty
45. R. Pizzo
46. V. Reeves
47. G. Salton
48. J. Sammet (5)
53. R. Sanderson
54. J. Seymour
55. G. Sokol
56. J. Terzian (15)
71. H. Ullman
72. N. Zachary