X59-1N

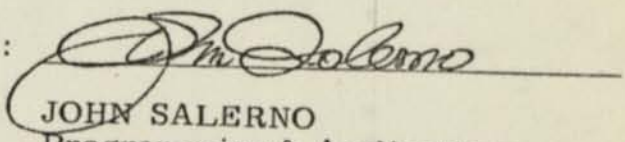
MOBIDIC
A

INPUT-OUTPUT SYSTEM
AND
MICRO FLOW CHARTS

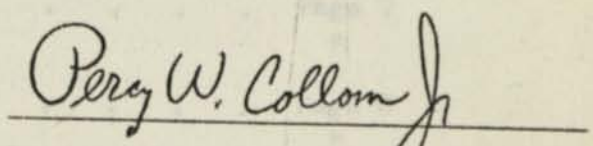
TITLE: SPECIFICATIONS FOR STANDARD FIELDATA TAPE$

DATE: Revised July 31, 1959

PURPOSE: To specify conventions to be required for use on all FIELDATA magnetic tapes. The initial sort and merge routines written for MOBIDIC will utilize these standards. Other conventions may be added at a later date.

Prepared by: 

JOHN SALERNO
Programming & Applications

Approved by: 

PERCY W. COLLOM, JR. Captain
Chf, Programming & Applications

TABLE OF CONTENTS

PROGRAMMER'S DATA FORMAT STANDARDS

The format and contents of these specifications are based on the following references:

1. Sylvania Electric Products, MOBIDIC PROGRAMS - ISSUE NO. 1: "SPECIFICATIONS FOR TAPES BEING USED IN SORTS AND MERGES", 21 November 1958.
2. Carnegie Institute of Technology, Letter, Dr. A. Perlis to Captain P. W. Collom, 14 January 1959, and comments made 9 April 1959.
3. I. B. M. Memo: "FIELDATA MAGNETIC TAPE SPECIFICATIONS", 12 January 1959.
4. Philco, Document: "FIELDATA MAGNETIC TAPE SPECIFICATIONS".
5. Sylvania Electric Products, MOBIDIC PROGRAMS - ISSUE NO. 4: "REVISED SPECIFICATIONS FOR TAPES BEING USED IN SORTS AND MERGES", 9 February 1959.
6. Sylvania Electric Products, MOBIDIC PROGRAMS - ISSUE NO. 8: "FORMAT OF SPECIFICATION BLOCK AND BLOCK LABEL", 16 March 1959.
7. Discussion, FIELDATA Magnetic Tape Standards Meeting, USASRDL, Fort Monmouth, New Jersey, 18 March 1959.
8. Written comments from "Multiple Task Study" Group, 10 April 1959.
9. USASRDL, "SPECIFICATIONS FOR STANDARD FIELDATA TAPES", extract from "FIELDATA Equipment Intercommunication Characteristics", 1 April 1959.
10. USASRDL, "REVISED SPECIFICATIONS FOR STANDARD FIELDATA TAPES", 21 April 1959
11. Sylvania Electric Products,"REVISED SPECIFICATIONS FOR STANDARD FIELDATA TAPES", July 6, 1959.

1. INTRODUCTION

The following magnetic tape format specifications are FIELDATA standards. These standards have been established for the FIELDATA family of equipments so that interchangeability of tapes between machines may be maintained. The conventions have been prepared after consideration of all factors, including recommendations from Sylvania, IBM, Philco and others.

Two forms are established:

A. A high reliability form for normal use.

B. A relaxed form for use with off-line equipment which produces or uses magnetic tapes.

It should be emphasized that these are tape specifications rather than program specifications. The sort and merge programs being written will use these tape standards as a basis for further definition of the program specifications.

The exact format of spec blocks and trailer blocks will be specified at a later time. Every attempt has been made to allow maximum program flexibility throughout the FIELDATA family of machines and programs, within the limitations of requirements.

2. DEFINITIONS

2.1 Word: One FIELDATA Computer word consisting of 36 bits plus a sign bit (plus a parity bit not available to the programmer.)

2.2 FILE: All the information of a specified type (e.g., personnel data, stock numbers) needed for a given problem or part of the problem. It may include more than one tape, be only one tape long, or be just a part of one tape. A set of Items.

2.3 TAPE: One reel of magnetic tape up to 3600 feet long.

2.4 BLOCK: A group of words contained within a pair of beginning and ending block marks.

2.5 SPEC BLOCK: The block which essentially identifies and defines a file (or tape). It appears either at the beginning of the file (or tape) or on a separate tape.

2.6 TRAILER BLOCK: The block which contains file and/or tape statistics, as well as information on end of file, end of tape, etc. It appears either at the end of the file (or tape) or on a separate tape.

2.7 BLOCK LABEL: The first word of each data block. It specifies either the number of complete items within the block or an indication that the block contains a split item, and may contain additional information.

2.8 SENTINEL: A code word or bits within a word used to indicate a condition within a block or file or tape.

2.9 TRAILER SENTINEL: A sentinel used to signal that the current block is either a file or tape trailer.

2.10 ITEM: A unit of information consisting of specific information elements known as fields (e.g., an item for one man might consist of name, rank, and serial number.) An item consists of one or more fields.

2.11 FIEID: A specific information element (e.g., name, rank, or serial number.)

2.12 ITEM KFY: The particular fields that are used to specify the criteria for a sort or merge.

2.13 ITEM LABEL: The first word of each variable size item. Specifies the number of words within a variable item and may contain other information.

2.14 SHORT ITEM: An item containing less than 256 words.

2.15 LONG ITEM: An item containing 256 or more words.

2.16 DATA BLOCK: A set of items, either of fixed length or variable length but not both. A data block is neither a specification block nor a trailer block.

3. BLOCK SIZE

3.1 NUMBER OF WORDS:

(A) FIXED ITEM SIZES:

If a file is composed of long, fixed size items, each item must be written as one or more blocks.

In general, for short fixed size items, a block contains more than one item. Data blocks should normally be made as large as possible (up to 511 words) by putting the maximum number of items into one block. This is illustrated in the following table:

ITEM SIZE	Max. No. of Items/Data Block
Over 255	1
171 - 255	2
128 - 170	3
. . .	. . .
3	170
2	255
1	510 (See 5.3,5.4)

(B) VARIABLE ITEM SIZES:

Blocks may contain varying numbers of variable length items. The length of the block is variable but limited to a maximum of 511 words.

3.2 FILE CONTENTS: All items within any one file will be of the same type: all fixed length or all variable length items.

3.3 FIXED LENGTH ITEMS: All data blocks, except possibly the last, will be the same size within any file which contains only fixed length items. Items of size greater than 510 words will be evenly distributed over an economical number of data blocks. In case of an "odd" number of words, even distribution will be accomplished by padding with a sufficient number of dummy words to permit even division.

3.4 VARIABLE LENGTH ITEMS: More than one short, variable length item may be put into one data block. Long variable length items will start at the beginning of a new data block.

3.5 SPLIT ITEMS: A single item may be contained in more than one data block; but a data block may not contain both part of a split item and all or part of another item. No item which is less than 511 words may be split across two or more blocks.

4. TAPE FORMAT FORMS

Two basic forms are permissible under these standards, as follows:

4.1 HIGH RELIABILITY FORM:

Whenever possible, this high reliability form will be used for information on magnetic tapes. This applies particularly to the input tapes and files processed by the sort and merge routines, and to all tapes prepared by computers for possible further processing by these sort-merge routines. The principal features of the high reliability form are listed below. These features will be later described in separate paragraphs.

(1) Tape Spec Block

(2) File Spec Block

(3) A block label on every block

(4) An item label on every variable length item.

(5) File Trailer Block

(6) End of File Sentinel (Hardware or programmed or both)

(7) Tape Trailer Block

(8) End of Tape Sentinel (Hardware or Programmed or both)

4.2 RELAXED FORM:

The relaxed form may be used whenever it is desired to process information which has not been produced by a computer. This applies to areas such as card-to-tape conversion, recorded output from analog-to-digital conversion, etc. In general, when it is neither possible nor convenient to add information of any type to the data itself, utilization of the relaxed form will require only the addition of spec (header) and trailer blocks. The use of the relaxed form is normally limited to files of fixed length items. However, variable length items may be padded out to fixed length in order to use this relaxed form. This, for example, a file on 80-commumn cards can be processed in the relaxed form after adding zeros to pad all cards to 80 characters. The principal features of the relaxed form are:

(1) Tape Spec Block, possibly on separate tape

(2) File Spec Block, " " " "

(3) A File Trailer Block, " " " "

(4) A Tape Trailer Block, possibly on separate tape

(5) Indicators for End of File and End of Tape Determination

(6) Parameters identifying location of (1) - (5)

5. IDENTIFICATION MARKERS

5.1 TAPE SPEC BLOCK: Associated with each tape will be a tape spec block. The tape spec block will be the first block on the tape in the high reliability form and will be written in the non-interpret sign mode. The tape spec block will be preserved and/or updated by all routines, and is limited to a maximum size of 511 words, as are all other blocks. When the relaxed tape format is used, the tape specification block may be an input from either magnetic tape, punched paper tape, or punched cards.

(A) LABEL PART:

Every tape spec block will contain a label part.

The label part will be prepared in a specified format suitable for printing on a FIELDATA typewriter without editing. The label part of the tape spec block will contain the following information:

(1) Security Classification (including password for tape)

(2) Installation Code

(3) Tape Identification

(4) Date/Time Group

(5) Reel Number (mmm of nnn)

(6) Available sentinels (block label, true block count, mechanized EOT)

(7) Indication of type of spec block (i.e. Tape)

(8) Name, location, and number of files on this tape reel

(B) PARAMETER PART:

Every tape spec block will contain a parameter part. The parameter part will be prepared in a specified format suitable for printing on a FIELDATA typewriter without editing. The parameter part of the tape specification block will contain the following information:

(1) Tape Size (Number of Blocks in this Tape), plus Indicator for Estimated or True.

(2) Blank area for use by Sort-Merge programs.

(C) OPTIONAL REMARKS PART:

Every tape spec block will have provision for an optional remarks part. The optional remarks part is completely variable in format, and its size is subject only to the overall block size limitation of 511 words. The optional remarks part of the tape specification block is not normally referenced in programming, and may be wholly or partially unused. Information such as the following may be included in the optional remarks part of the tape spec block:

(1) Locator Routine for Files

(2) Maximum Size of Blocks on this Tape.

5.2 FILE SPEC BLOCK: Associated with each file will be a file spec block. The file spec block will be written in the

5.3 BLOCK LABEL: When the high reliability format is used, the first word of every data block will be a block label.

The 36 bits are used as follows: 6 bits for a sentinel, 9 bits to specify the number of words per block, 9 bits to specify number of items per block or number of blocks required for item, 1 as an indicator bit, and 11 as a hash total for the block.

The sentinel area in the block label of any and all data blocks must not contain the End of File or the End of Tape configurations shown in the respective spec blocks. However, any other configurations may be used as indicators for other purposes.

The use of block labels is shown diagramatically in Fig. I. The bit arrangement of the block label is shown in Fig. II.

5.4 ITEM LABEL: Item labels will be used only for variable length items, and therefore only when the high reliability format is used. The first word of every variable length item will be an item label.

The 36 bits are used as follows: 2 for a sentinel, 4 to show block spread (i.e. How many blocks are required for the item), 6 bits are presently reserved, 12 bits indicate the number of words in the item, and 12 bits are a hash total for the item.

The use of item labels is shown in Fig. I. The bit arrangement of the item label is shown in Fig. III.

5.5 TRAILER SENTINEL (= EOF, EOT Sentinels)

(A) Programmed Sentinels

The trailer sentinel for either EOF or EOT appears in the left hand six (6) bits of the block label of the trailer. Any FIELDATA character which is printable by the Flexowriter may be used. The configuration of these 6 bits may differ on different tapes or files.

In the case where the file trailer block is the next to last block, then it contains a fixed indicator to show this.

(B) Mechanized Sentinels:

The mechanized EOF mark should be at the end of the File Trailer block. The mechanized EOT mark may appear either before or after the Tape Trailer block, depending on the amount of information on the Tape. The use of sentinels and trailer blocks is illustrated in Figure IV.

5.6 FILE TRAILER BLOCK: Associated with each file is a file trailer block. If a file is continued into another tape, the file trailer will appear only at the end of the file. The file trailer block will be preserved and/or updated by all processing runs. It will be in the same mode as the data and follows immediately after the last data block of the file.

(A) DESCRIPTOR PARTS:

Every file trailer block will contain a descriptor part. The descriptor part will be in a specified format suitable for printing the a FIEIDATA typewriter without editing. The descriptor part of the file trailer block will contain the following information:

(1) Indication of type of trailer (i.e. Tape or File)

(2) File Identification (Same as in File Spec Block)

(3) Data/Time Group

(4) Block Count (Total number of data blocks in the file).

(5) Item Count (Total number of data items in the file).

(6) Block Sub-Count (Number of data blocks in the portion of the file on this tape).

(7) Item Sub-Count (Number of data items in the portion of the file on this tape).

(8) A hash total of word N of each item in a file. (Word N designation appears in the file spec block.)

(B) OPTIONAL REMARKS PART:

Every file trailer block will have provision for an optional remarks part. The optional remarks part is completely variable in format; and its size is subject only to the overall maximum block size limitation of 511 words.

5.7 TAPE TRAILER BLOCK: Associated with every tape is a tape trailer block, which will be written in the same mode as the block preceeding it. The tape trailer block will always be the last valid block on the tape, is completely variable in size subject to the overall maximum block size limitation of 511 words, and will be preserved and/or updated by all processing runs.

(A) DESCRIPTOR PART:

Every tape trailer block will contain a descriptor part. The descriptor part will be in a specified format suitable for printing on a FIELDATA typewriter without editing. The descriptor part of the tape trailer block will contain the following information:

(1) Indication of type of trailer (i.e. Tape or File)

(2) Tape Identification

(3) Date/Time Group

(4) Number of both partial and complete files on this tape.

(5) Total number of blocks on this tape (including all spec, data and trailer blocks, and this block).

(6) Total number of data items on this tape.

(7) Is the last file on this tape continued into another tape? (Yes or No)

(8) If last file on this tape is continued into another tape, the number of data blocks from this incomplete file on this tape.

(9) If last file on this tape is continued into another tape, the number of items from this incomplete file on this tape.

(B) OPTIONAL REMARKS PART:

Every tape trailer block will have provision for an optional remarks part. The total number of words is limited only by the restriction that the maximum block size is 511 words.

6. GENERAL SENTINEL: A word consisting of all 1's may never be used as a block or item label.

7. METHOD OF COUNTING: All block and word counts include the block or word in which they are contained. In particular, the block counts include spec and trailer blocks, and word counts include block and item labels.

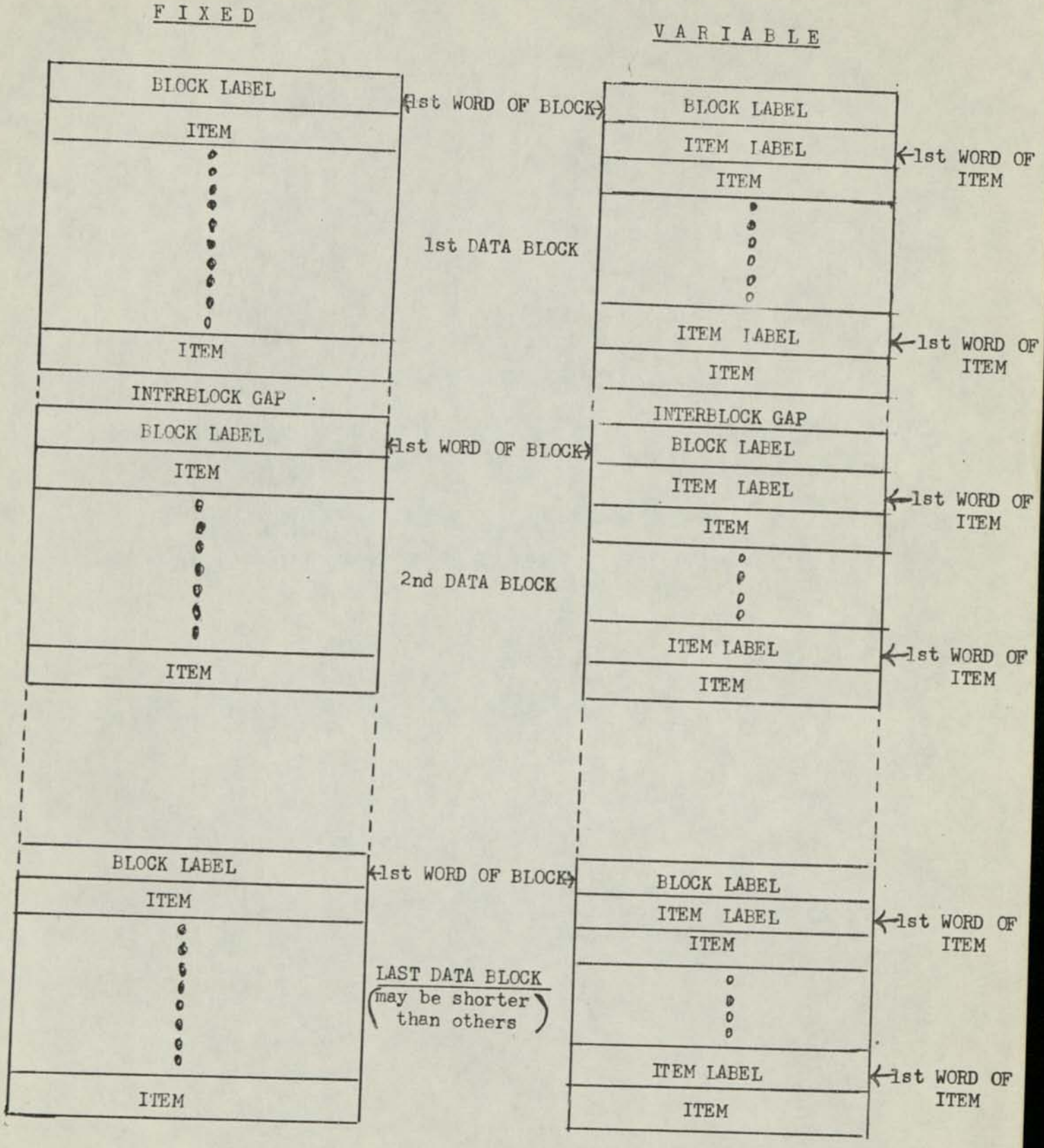
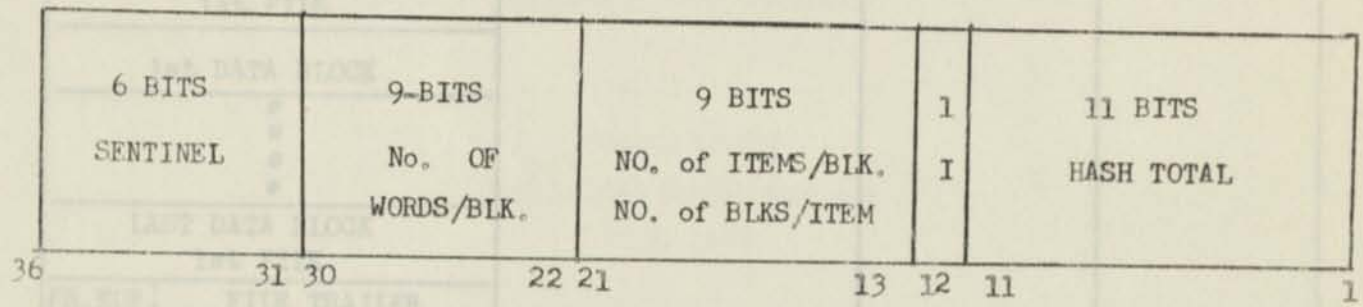


FIG. I

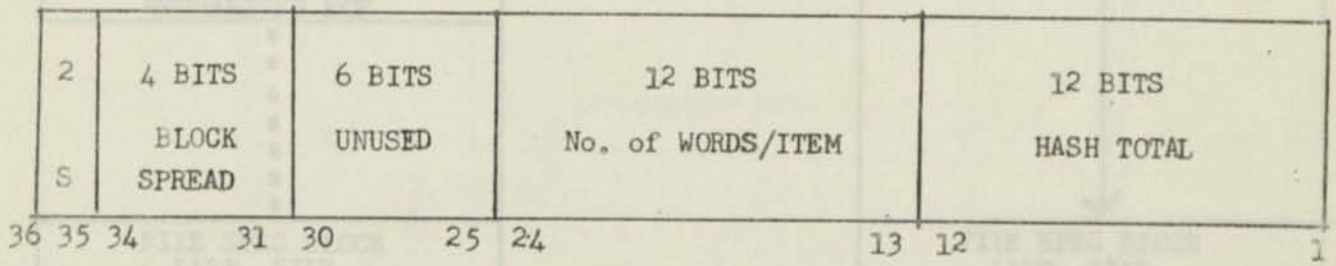
BLOCK, ITEM LABELS



I = INDICATOR BIT

FIG II

BLOCK LABEL



S = 2 Bit Sentinel

FIG III

ITEM LABEL

(A)

ONE TAPE FILES

(B)

MULTI TAPE FILE

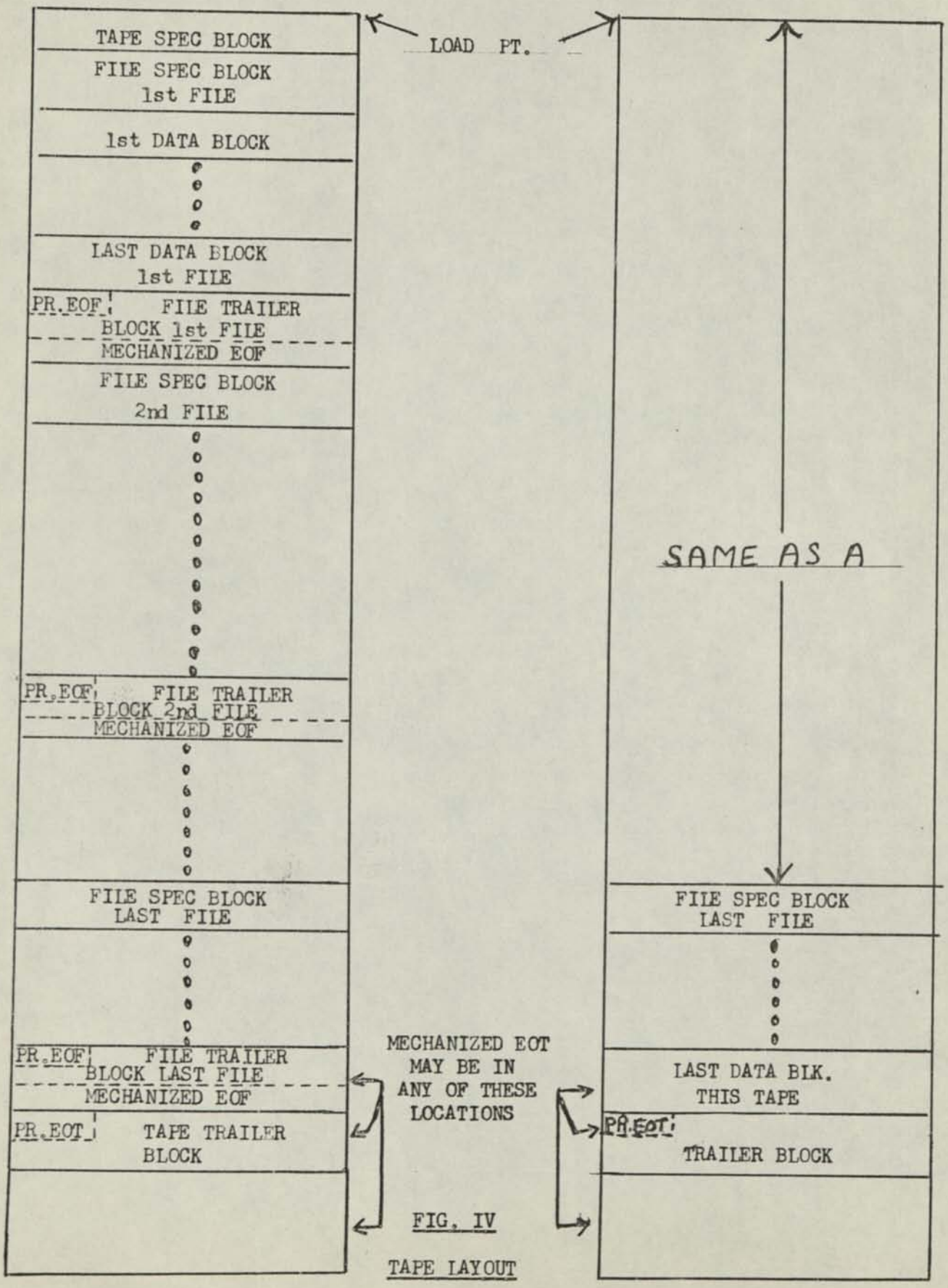


FIG. IV

TAPE LAYOUT

MOBIDIC A
INPUT-OUTPUT SYSTEM
AND
MICRO FLOW CHARTS

Signal Corps
Technical Requirements
SCL 1866

Contract No. DA-36-039-SC-78111

Submitted to:
U. S. ARMY
Signal Research and Development Laboratories
Fort Monmouth, New Jersey

December 1959

SYLVANIA ELECTRONIC SYSTEMS
A Division of Sylvania Electric Products Inc.
DATA SYSTEMS OPERATIONS
189 B Street - Needham 94, Massachusetts

TABLE OF CONTENTS

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONTINUED)

LIST OF ILLUSTRATIONS

SECTION I

INTRODUCTION

This document has been prepared under the assumption that the user is familiar with the MOBIDIC Preliminary Programming Manual.

The Micro Flow Charts are designed to answer reasonable questions which a programmer might ask about the Input-Output System. In a few cases, slight liberties have been taken with the actual mechanization in order to make the Micro Flow Charts more meaningful to the programmer.

The MOBIDIC Input-Output System is the means by which information to be processed and related instructions are transferred from or to the various devices. This system is also used to report or store the results obtained from the information and the instructions.

Two types of devices can be used for the input on MOBIDIC A -- either the paper tape reader or the magnetic tape device. The paper tape can be prepared on off-line equipment whereas the magnetic tape must be prepared and read through the use of a program in MOBIDIC A.

Output may be obtained by the use of either paper tape, or printing on the flexowriter. Intermediate output may be on magnetic tape.

For the type of problem that involves the handling of large quantities of data the use of magnetic tape is recommended because of its superior speed and large storage capabilities. There are also many problems where the use of the slower paper-tape devices will either be unavoidable or not restrictive. (Not restrictive in that the speed of the device will have very little effect on the problem.)

SECTION II

CONVERTER SELECTION AND TRANSFER

2.1 TRANSFERRING INFORMATION

During the process of transferring information or data to and from MOBIDIC A and its various In-Out devices several factors have to be considered.

1. In-Out devices, e.g., magnetic tapes, operate asynchronously with respect to MOBIDIC.
2. The rate of information flow differs widely for the several types of In-Out devices.
3. The format of the transferred information may vary.
4. A parity bit for each character transferred must be generated or checked.

Communication between the central processor and the various In-Out devices depends on the In-Out converters (2 in MOBIDIC A). These are the synchronizing and counting units which also partly interpret the data, add, delete, or alter bits (or characters) as required by the individual instructions and the In-Out device addressed.

2.2 TIMING FUNCTION

When the In-Out instruction is decoded, the central processor must determine if a converter is free and if the In-Out device specified by the instruction is available. If not, the basic cycle of the central processor is interrupted in Timing Function 8 (TF8). This is effected by setting λ to ONE, a function known as "sticking λ". The timing function continues to repeat itself until λ is set to ZERO ("unstuck"). If either the two converters or the specified device is busy, TF8 is repeated while the converters make the memory accesses required by the In-Out instructions already being processed. The instructions already in operation continue to be processed in this manner until both the device specified by the new In-Out instruction and a converter are available. At this time, the central computer selects the lowest numbered free converter and transfers the In-Out instruction to its associated Converter Instruction Register (CIS).

2.3 INSTRUCTION TRANSFER TO CIS

At the time the instruction is transferred to the CIS, certain additional information from the central processor is also sent to the selected converter. The state of both the No Halt on Converter Error Flip-Flop (NHC), and the Interpret Sign Flip-Flop (ISN) is transferred to the converter flip-flops (NH and IS) respectively. The flip-flops (NHC and ISN) in the central processor are immediately reset after the transfer of information. Both the NHC and the ISN may be set manually on the console but will be reset internally by the processor after the transfer of the instruction to the converter. The console switches remain unchanged and will therefore not always indicate the state of the flip-flops. Both flip-flops are addressable by the program.

When the In-Out instruction is decoded, the type of the specified device is determined and its classification is transferred to the converter along with the instruction.

The In-Out devices for MOBIDIC A are classified as follows:

1. Input type -- Paper Tape Reader
2. Output type -- Flexowriter, Paper Tape Punch
3. Input or Output type -- Magnetic Tape

When the converter determines the device type, a check is made by the converter to determine if the In-Out instruction is "legal". If it is not legal, the converter's In-Out Alarm is set. Future action by the central processor at this point depends on the state of the NH flip-flop and is discussed in the section on "Alarm and Halt Conditions".

If the instruction is legitimate, the converter assumes control of the instruction and causes the specified device to be connected to the converter through the Device Switching Unit (DSU). (The DSU controls the flow of information between the device and the converter.) The device then begins operating in the specified reading or writing mode.

Each device has connected to it one DSU for each of the converters. The converter exercises overall control as well as control of the flow between the

central processor and the DSU. The operation of the converter proceeds independently of the central processor except when a memory access is required. Each access to memory by the converter interrupts the basic cycle of the central processor for eight (8) micro-seconds.

When the converter completes the In-Out instruction, the device is then disconnected and the converter made available. (Figure 2-1 shows a block diagram of this system.)

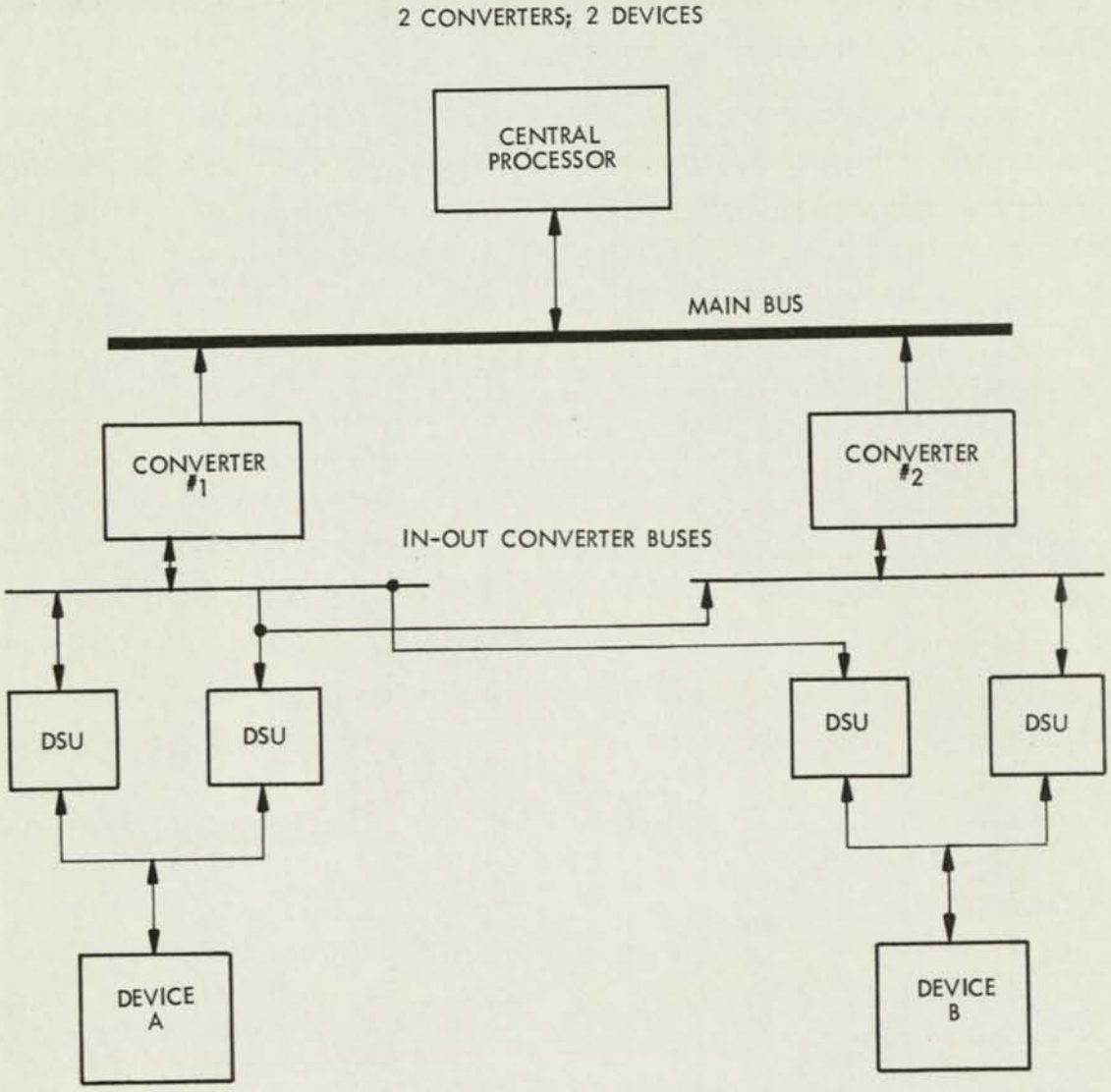


Figure 2-1. General Diagram MOBIDIC Input-Output System

SECTION III

MOBIDIC CHARACTER ARRANGEMENTS AND BIT CONFIGURATIONS

3.1 DEFINITION: "CHARACTER"

The term "Character" may apply to several different bit configurations in the MOBIDIC system depending on the machine word's location and the mode of operation then in use. The number of characters in a word will also vary for the several different modes.

3.2 BIT CONFIGURATIONS

3.2.1 6-Bit

A word in a MOBIDIC memory is composed of 36 binary bits plus one sign bit. The Alphanumeric In-Out instruction interprets the 36-data bits as six 6-bit characters. The remaining sign bit is expanded to a 6-bit character if the ISN is set, or is ignored if the ISN is reset. This means that a word may consist of either 6 or 7 characters on magnetic tape or paper tape depending on the state of the ISN flip-flop.

3.2.2 8-Bit

On MOBIDIC In-Out devices it is convenient to represent each character in an 8-bit configuration. These 8 bits are made up of:

1. A 6-bit information code.
2. A control bit in the 7th position indicating to the converter whether the character is a data character (1) or a control character (0).
3. A parity bit which is generated or checked by the converters.

Words are arranged on the device medium with the sign character, if interpreted, first. This is followed by the character occupying the most significant position in the word and continuing on to the least significant position.

3.3 WORD GROUPINGS

On the MOBIDIC magnetic tape, words are grouped into blocks consisting of from one word to the full length of the tape (3600 ft. maximum). In a normal operation the maximum block size is 511 words. To produce blocks of any greater length necessitates order modification of the converter. Blocks of information are separated by an "interblock gap" formed by the placement of a pair of "start of block" marks before a new block of data and a pair of "end of block" marks after each data block is recorded. This gap is necessary to allow the tape to accelerate to its full speed before the data is read or recorded. The interblock gap has a tolerance that allows for MOBIDIC's rewrite feature. The block marks are the control characters used by the converter during the execution of instructions involving magnetic-tape. (Figures 3-1 and 3-2 show the character size at various parts of the system and the magnetic tape format respectively.)

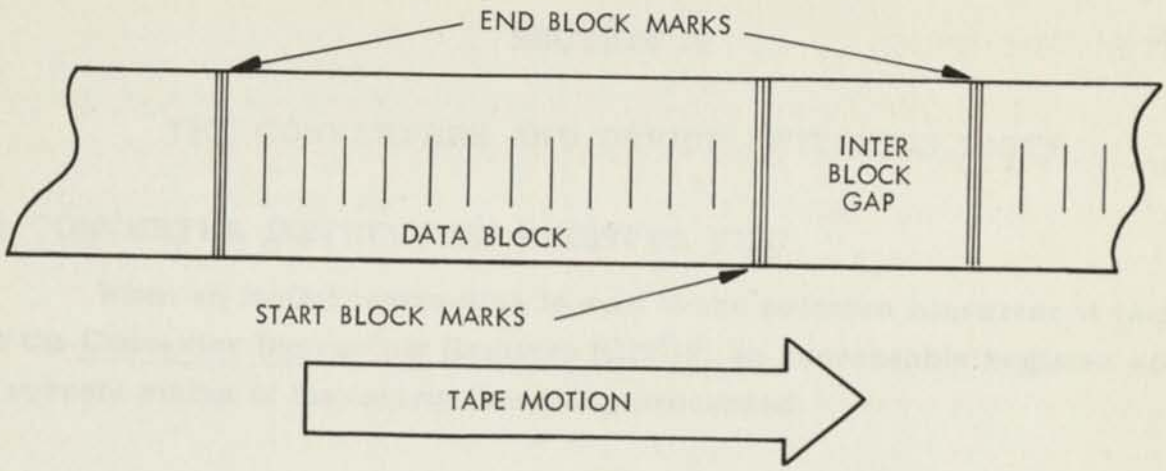


Figure 3-1. Magnetic Tape Configuration

OP CODE	bits/char. CENTRAL PROCESSOR	CHARACTERS PER WORD		bits/character AT THE DEVICE
WAN	6 1SN BIT/WD	ISN = 1 ISN = 0	7 6	8
WWA	6 1SN BIT/WD	ISN = 1 ISN = 0	7 6	8
RAN	6 1SN BIT/WD	ISN = 1 ISN = 0	7 6	8
RAV	6 1SN BIT/WD	ISN = 1 ISN = 0	7 6	8
WOK	3 1SN BIT/WD	13		8
ROK	3 1SN BIT/WD	13		8

Figure 3-2. Character Sizes in MOBIDIC System

SECTION IV

THE CONVERTERS AND DEVICE SWITCHING UNITS

4.1 CONVERTER INSTRUCTION REGISTER (CIS)

When an In-Out instruction is sent to the selected converter it is placed into the Converter Instruction Register (CIS^i)*, an addressable register containing the current status of the instruction being processed.

4.1.1 Address Counter

Bits 1-15 (a) of the CIS specify the memory address in the central processor to or from which the data is being transferred. This is the Address Counter (ADC) of the CIS and will be incremented by one during each memory access. The ADC will not be incremented if it contains a register address.

4.1.2 Device Address Register

Bits 16-21 (j) of the CIS specify the device being addressed by the instruction. This is the Device Address Register (DAR) and its contents are not changed during the processing of an In-Out instruction.

4.1.3 Word Block Flip-Flop and Word Block Counter

Bits 22-30 (k) indicate the number of words or blocks to be processed during the operation. The exact mode of operation depends on the interplay of the Word Block Flip-Flop (WB), Bit 30, and the Word Block Counter (WBC), Bits 22-29. Generally, k is decreased by one, depending on the In-Out instruction, after a word or block is processed by the converter. The exact interplay of WB and WBC may be found in any instruction write-up and on the following micro flow charts.

* i corresponds to the specific converter selected for this instruction.

4.1.4 Instruction Register

Bits 31-36, the Instruction Register (ISR), specify the type of In-Out instruction being processed. (This is the part of the CIS that controls the device specified by the DAR.) (Figure 4-1 shows the breakdown of the CIS.)

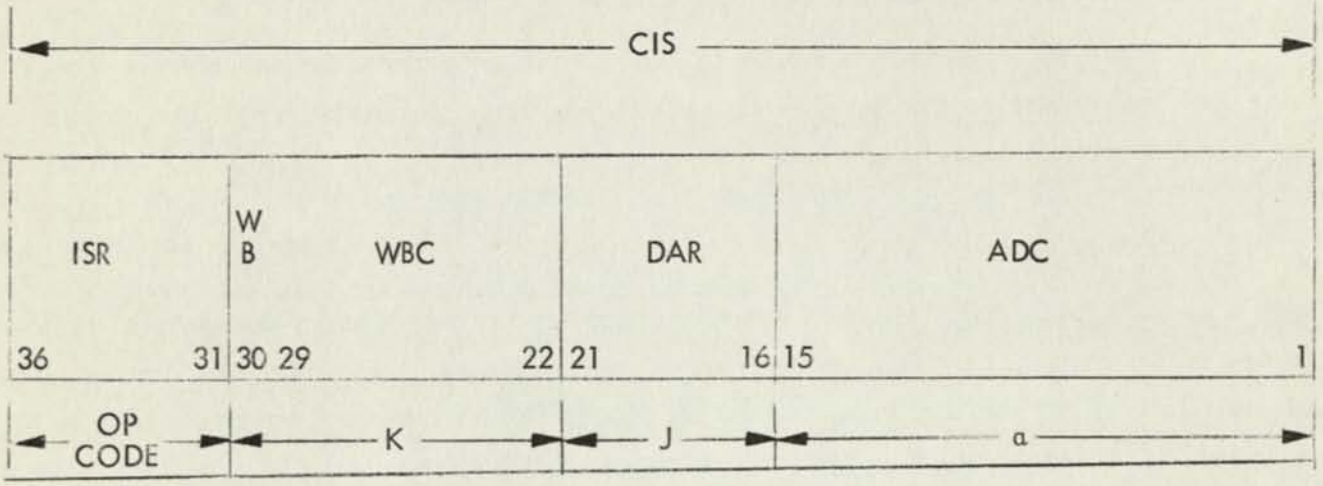


Figure 4-1. Converter Instruction Register

As information is passed through the converter, it is partially interpreted; bits or whole characters are added, deleted or altered as required. Parity is generated or checked by the converter.

4.2 BUFFER REGISTER

Each converter has a 37-bit Buffer Register (BFR) where the characters are assembled into a full machine word before being transferred to memory during a read instruction and broken down into characters to be sent to the device during a write instruction.

4.2.1 Busy Bit

When the BFR is ready to receive or send a word, it sets its Busy Bit (BB^i) to ONE. This BB is sensed during TF8 in the Basic Cycle and at certain places in the longer operations such as Multiply, Divide, etc.* No more than 22 micro-seconds can pass before the Busy Bits are sensed.

* After the access this BB^i is reset to zero.

4.2.2 Priority System

If more than one BB is set during this interval, the sequence in which they are acknowledged depends on the priority system as indicated in flow chart No. 5. Briefly, the priority is: (a) the real time system; (b) any magnetic tapes with the lowest numbered converter first; and (c) other Input or Output devices in converter order.

4.2.3 8-Bit Register (CIR)

Associated with the BFR is an 8-bit register (CIR). The CIR with the three other 8-bit registers (BSR), (BXR), and (TAR) form the path over which information must pass between the converter and the device. (See Figure 4-2.) The action of the converter during a write instruction is described to indicate how the various components of the CIS react as characters are sent to the output device.

For a write instruction, a word is transferred from memory to the BFR when the BB^i for that converter indicates that the BFR is ready to receive a word. The sign, if interpreted, is shifted to the CIR and expanded to the 6-bit code for either a Fieldata "ONE" or "ZERO", before the data bits are processed. The word is then broken down into characters by the transfer of one character from the high order position of the BFR to the CIR. (Its size depends on the particular instruction.) Each character is moved in sequence to this high order position by shifting the BFR left, the number of shifts being equal to the number of bits in a character for this order. The CIR expands the character to 8-bits by generating a parity bit and adding the control bit. For an octal character of 3 bits the CIR forms the 6-bit pattern for the Fieldata character corresponding to the 3-bit octal number. From the CIR the character is sent to the BSR, and the CIR is cleared. The next character in the BFR is then sent to the CIR where it is expanded. This process continues until the BFR is empty whereupon the BB is set. This causes the central processor to place the next word from memory into the BFR and the BB will be cleared. This word is now operated on in a similar fashion to the one already described. The characters from the BFR proceed through the BSR and BXR to the TAR as fast as characters are cleared out of the register ahead. From the TAR they are sent to the device by the DSU, at the required rate. This rate is controlled by the DSU connected

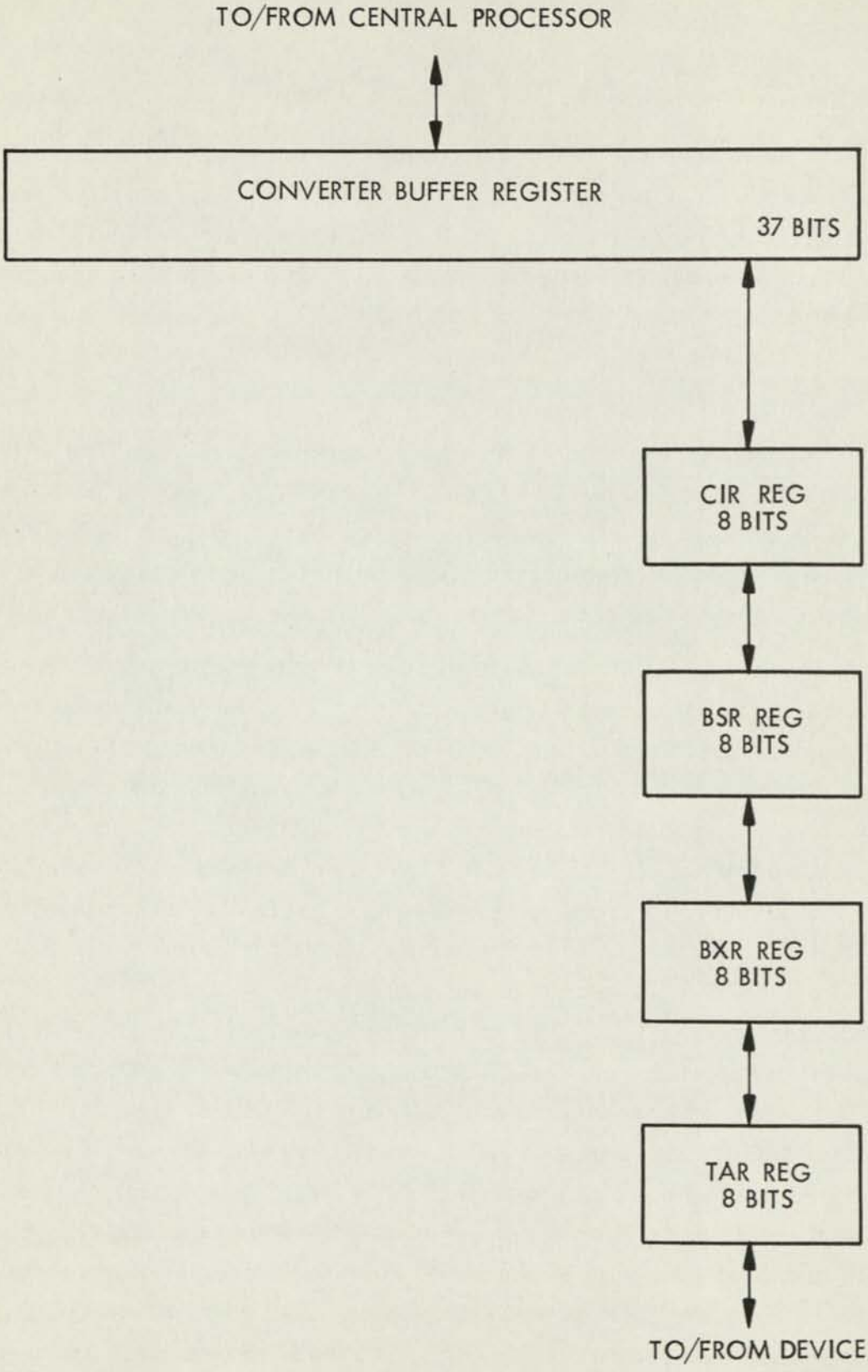


Figure 4-2. Buffer Registers

between the converter and the device. When all the required words have been transmitted, the device is disconnected and the converter made available. For Magnetic Tapes, at the completion of the order, End Block marks are generated in the CIR and transmitted to tape before the tape is disconnected.

For a read instruction, the flow of information is in the opposite direction to that described above. The characters arrive at the TAR from the device and are sent through the BXR and BSR to the CIR as soon as the previous register is cleared. In the CIR, parity is checked and the Interpret Parity Error (IPE) alarm set if an error is detected. (See Alarm section for details.) From the CIR the character is sent to the low order positions of the BFR while the data already in the BFR is shifted left to accommodate this character. For alphanumeric data, the BFR data is shifted left six places just before the 6-bit character is received from the CIR. For Octal data the BFR data is shifted left 3 places and only the low order 3 bits of the CIR are transferred to the BFR.

4.2.3.1 Interpret Sign Error (ISE)

During the operation of an Octal order, if the CIR detects a data character other than a Fieldata 0-7 the Timing Read Error (TRE) alarm is set. An integral number of alphanumeric words, between octal words is allowed as long as the octal words are complete. Since the converter knows which character is the sign, it examines this character to determine whether or not it is a Fieldata 1 or 0. If it is neither, the Interpret Sign Error (ISE) alarm is set. When a full word is assembled in the BFR, the BB is set and the word is transferred from the BFR to memory. The BFR now begins to assemble the next word. The above process continues until the order is complete and then the device and converter are disconnected.

4.2.3.2 N Counter

The converter controls the number of characters in the word being assembled in the BFR through the N counter. This counts the number of characters that have been shifted to the BFR, and therefore knows when the word has been completely assembled. The N counter also tells the converter which character should be the sign so that this character can be checked in the CIR. Depending on the

instruction this counter counts 6 (alphanumeric with no sign), 7 (alphanumeric with a sign) or 13 (octal with a sign) characters. When the operation is Read Reverse in the interpret sign mode, the converter checks the last character for a legal sign (when N = 7) which is the last character of a word to be read.

4.2.3.3 Extra 8-Bit Registers (BXR and BSR)

The extra 8-bit registers, BXR and BSR, are necessary to maintain the steady flow of data to or from the fastest device, Magnetic Tape. The worst possible case would be, in tape reading, the early arrival of the characters at the TAR and late acknowledgment of the "Busy Bits" by the central computer. Since the data comes from magnetic tape during a read operation with no definite synchronization with respect to the rest of the system, the character itself generates a synchronizing signal when it reaches the TAR. This tells the DSU that a character is now complete in the TAR and ready to be transferred to the BXR.

For a write operation the flow is in the opposite direction and the DSU sees that a character is placed on tape within the proper time interval.

SECTION V

ALARM AND HALT CONDITIONS

5.1 IN-OUT ALARM FLIP-FLOPS (IOA)

There are six In-Out alarm flip-flops for each converter. The In-Out Alarm flip-flop (IOA) is set whenever any of the other five alarm conditions occur. Each IOA is addressable and may be reset either manually or under program control.

5.1.1 Device Alarm Flip-Flop (DVA)

The Device Alarm flip-flop (DVA) is used to indicate various failures of the device. The type of failure is not specified by the alarm. Device power failure, Magnetic Tape Breakage or reaching the physical end of tape will all set the DVA. The converter is halted immediately and the central processor stops after completing any other In-Out instruction being processed by the other converters. Any internal instruction being processed is completed before the halt takes effect.

5.1.2 No Halt on Converter Error Flip-Flop (NHC)

The remaining four alarms are affected by the state of the No Halt on Converter Error Flip-Flop (NHC) at the time the instruction is transferred to the converter. NH is the flip-flop in the converter that shows the state of the NHC. If NH is reset, the action of the converter depends on the type of alarm, but the central processor is halted.

5.1.3 Improper Order Alarm Flip-Flop (IMO)

As soon as an In-Out instruction is sent to the CIS, it is checked to determine whether or not it is a legal order. Read Instructions that specify an output device or in some cases a ZERO (0) in the WBC, are illegal. The Programming Manual explains what an improper order would be for any individual instruction. If the order is inconsistent the Improper Order Alarm flip-flop (IMO) is set. If the NH is set, the error is ignored and the order causing the illegality will

effectively have been skipped over; i. e., the converter and device are not connected. It is important to note that the order is only checked for consistency at the time of the instruction transfer to the CIS and any modification of the CIS by the programmer may produce an improper order that will not be detected in the normal fashion.

5. 1. 4 Interpret Parity Error Flip-Flop (IPE) and Interpret Sign Error Flip-Flop (ISE)

If a parity error is detected in the CIR during a read operation the Interpret Parity Error flip-flop (IPE) will be set. If a character that should be a sign is found to be something other than a Fieldata one or zero, the Interpret Sign Error flip-flop (ISE) will be set. If the NH is set, either or both of these alarms will not affect the converter operation. The affected character is sent to the BFR in the normal fashion and the operation continues uninterrupted. If NH is not set, the central computer stops subject to the usual restrictions for a normal halt.

5. 1. 5 Timing Read Error Flip-Flop (TRE)

If any block marks occur out of place or an illegitimate character is detected during a Read Octal operation, the Timing Read Error flip-flop (TRE) is set. For an octal instruction a legitimate character must be the Fieldata code for a number (0-7). Certain timing errors that may occur in the DSU will also set this alarm. If the NH is set the converter ceases to transmit data and, in the case of magnetic tape, searches for an end block mark. The converter is disconnected but the central processor is not stopped. If the NH is reset, the converter still searches for an end of block mark while the central computer executes a normal halt.

5. 2 DISPLAY OF ALARMS

Only the IOA alarms are displayed on the console. The other In-Out alarms are displayed in the converter itself but a panel must be lifted to inspect them. There is no programming method of determining which of the five alarms set the IOA since they are not addressable.* The same failure may in some cases set more than one alarm.

*Resetting the IOA will reset all five individual alarms.

SECTION VI

ORDER MODIFICATION

6.1 CIS MODIFICATION

A MOBIDIC In-Out instruction may be modified after the operation has begun because the CIS is addressable. This CIS^i* may be modified by a LOD instruction, a CLA, STR sequence, or any other instruction sequence that affects the contents of an addressable register.

Since the CIS^i monitors the in-out operation of the converter and its timing is independent of the central processor's basic cycle, great care must be taken when any modification is attempted. The state of the CIS^i may change before the desired modification is accomplished and this possibility must be expected. The NH and IS flip-flops, as well as the device type flip-flops, are not addressable and cannot be modified. The DAR cannot be altered in any way. Attempts to do so are blocked. It is possible, however, to alter the contents of the ISR, WB, WBC, and ADC during an in-out operation.

6.2 WBC MODIFICATION

If the WBC is changed during the in-out operation, it has the expected effect of extending or abbreviating the number of words or blocks processed by the converter. A gain or loss of a word is possible, if the timing problem is ignored. If the WBC is set to zero, the order may be arbitrarily terminated and further order modification may upset the word structure. Example 1 shows how a correct WBC modification may be programmed and explains the timing problem in greater detail.

The WB flip-flop which controls the mode of operation may be changed during an instruction. The change in mode involved depends on the operation and timing at the time of modification. A read instruction in the block by block mode can be changed to reading words by making WB a 0 instead of a 1.

* i refers to the particular converter involved

6.3 ADC MODIFICATION

Altering the ADC has the expected effect of changing the memory address involved. If timing is ignored, the same location may be addressed twice or a location may not be loaded as intended. ADC modification is also shown and explained in Example 2 following this section.

6.4 ISR MODIFICATION

Successful modification of the ISR is possible in special cases. The device must be magnetic tape and the WBC equal to zero. If the ISR is changed before the WBC is zero, confusion may result possibly with the converter remaining selected but doing nothing, since the old order can not be properly terminated and the normal starting sequence is by-passed for the new one.

In the case of magnetic tapes, changes in the ISR or the WBC, after the WBC has reached zero, results in the new operation being correctly performed. Changes from reading to writing, or to and from the reverse mode on tape (BSP & RRV) are all possible, since, before disconnecting the magnetic tape, the DSU senses the CIS to determine if the order has been modified. If it has, the DSU initiates the normal start sequence and the instruction is executed properly. If the order has not been modified, the tape unit and converter are disconnected normally.

If an order involving an input or output type device is modified when WBC = 0 it is expected that it will be ignored since the device is disconnected almost immediately after the WBC reaches zero.

The following examples show how most of the mentioned modifications may be accomplished, and the timing difficulties in each case.

SECTION VII

MODIFYING A "WRITE" INSTRUCTION

Since the converter sends a character to magnetic tape in a definite time interval, the time available for testing and converter modification depends on the number of characters in the word. If the ISN is set, there are 7 characters per word and the time available for testing is 156 microseconds. If the ISN is not set, there are only 133 microseconds for testing and modification. *

The ADC is incremented by one at the time of the memory access, but the WBC is decreased only after all the characters have been shifted out of the BFR. This interval between the memory access and decrementing of the WBC is the time when the testing and modification must occur. The converter's contents at any time are determined by the equations listed below:

Original Instruction: WAN A, MT_j, B_k

After the Nth word has been shifted from buffer and N + 1 word is in buffer WAN (A + N + 1), MT, (B - N)

When the Nth word is in the BFR: WAN (A + N), MT B - (N - 1)

7.1 EXAMPLE 1: "WRITE" MODIFICATION WITH WBC

Write a block of 1000 words onto tape in the non ISN mode from locations 1000 to 1999.

```
START   SNR     * + 1, 0, ISN
        WAN     1000, MT, 510
        .
        .
        .
        CLA     K           CONST TO A Register
GO      TRC     CNV         compare with converter
```

*This is the time between "busy bits" so that the 8 microsecond access time must be taken from it.

```
        TRU   GO                Test again
        TRU   * + 1             Go Modify
        LOD   NEWOP, 0, CNV     New Instr. to CNV
        HLT                     STOP
K       WAN   1501, MT, 10
NEWOP   WAN   1501, MT, 500
        END   START
```

Worst Case: If memory access is made just after TRC so that a second comparison will have to be made before the converter's change to desired value is discovered.

Timing:

8+16+22+18+8 = 72 microseconds

Mem Acc+TRU+TRC+LOD+Mem. Acc. of other converter = Time

This falls well within the 133 microsecond time limit and the instruction will be accurately modified. In the worst case, it took 54 microseconds to discover that a change is needed. This means that any modification that takes 79 [133-54 = 79] microseconds or less could be used in place of the LOD Instruction. It must be remembered that when the converter is being modified, the word must be fully specified or part of the CIS may be destroyed.

As long as the testing and modifying is accomplished in less than the time used by MOBIDIC to send a word to tape it will be successful.

7.2 EXAMPLE 2. "WRITE" MODIFICATION WITH ADC

Write a block of 500 words on magnetic tape in the non-ISN Mode. The first 250 words begin in 1000 and the second 250 start in 2000.

```
START   SNR   *+1, 0 ISN
        WAN   1000, MT, 500
        CLA   K                 CONST       A REG
```

```
GO      TRC   CNV          Compare with converter
        TRU   GO
        TRU   *+1
        LOD   NEWOP, 0, CNV
        HLT
K       WAN   1250, MT, 251
NEWOP   WAN   2000, MT, 251
        END   START
```

The worst case and timing for this example is the same as Example 1.

The constants have been changed to ensure that the converter ADC is changed as soon as the word in 1249 has been sent to the converter.

7.3 EXAMPLE 3: "WRITE" MODIFICATION WBC AND ADC

Write a block of 1000 words onto magnetic tape in the non-ISN mode. The first 500 words start in 1000 and the second 500 start in 2000.

```
START   SNR   *+1, 0, ISN
        WAN   1000, MT, 510
        .
        .
        .
        .
        CLA   K
GO      TRC   CNV
        TRU   GO
        TRU   *+1
        LOD   NEWOP, 0, CNV
```

```
        HLT
K       WAN   1500, MT, 11
NEWOP   WAN   2000, MT, 501
        END   START
```

The timing is the same as the previous example. The constants are the only changes.

From these examples it can be seen that changing the constants can force this typical loop to perform many types of order modification.

SECTION VIII

MODIFYING A "READ" INSTRUCTION

The converter must accept the characters as they come from magnetic tape even with a possible variance in the tape speed. This means a reduction or expansion of 10% in the time available for testing and modification. The 10% reduction, since it cuts down on the available modification time, is the case to be considered. Instead of 133 microseconds in the non-ISN mode only 120 microseconds are available to the programmer. In the ISN mode, only 140 micro-seconds, instead of 156 are available.

The WBC is decreased by one as soon as a complete word is assembled in the BFR. The ADC is then incremented during the memory access that transfers this word to memory. The value of the converter before the WBC and ADC have been changed is the value to be used in testing. This value at any point in the operation can be found from the equations listed below.

Original Instruction: RAN A, MT, B

After N words have been read into memory the converter has:

RAN(A + N), MT, (B - N)

8.1 EXAMPLE 4: "READ" MODIFICATION WBC

Read 500 words in the non-ISN mode from magnetic tape into memory starting at location 1000.

```
START    SNR     *+1, 0, ISN
         RAN     1000, MT, 255
         .
         .
         .
         .
```

```
                CLA     K
GO              TRC     CNV
                TRU     GO
                TRU     *+1
                LOD     NEWOP,0,CNV
                HLT
K               RAN     1250,MT,5
NEWOP           RAN     1250,MT,250
                END     START
```

The timing here is the same as in Examples 1 - 3.

The worst case again is 72 micro-seconds which falls well within the limit of 120 micro-seconds. Note, however, any method of modification instead of LOD must be completed in 66(120 - 54) or less micro-seconds. To be sure, no more than 60 micro-seconds should be used.

By changing the constants, the above example can be used to perform a wide variety of reading order modifications. The following example gives the new constants to be applied to the sample test loop of the previous examples.

8.2 EXAMPLE 5: "READ" MODIFICATION WBC AND ADC

Read 500 words in non-ISN mode from magnetic tape into memory. The first 250 go into 1000 to 1499 and the second 250 on to 2000 to 2499.

```
1st. instr.     RAN     1000,MT,255
K               RAN     1250,MT,5
NEWOP           RAN     2000,MT,250
```

To successfully modify the converter, a program must be able to examine, test, and modify the converter within the time limit. This time limit is the time between words for the specified mode and device.

For this reason it is simpler to modify the converter for instructions involving Input or Output type devices than those involving magnetic tape.

The times for the devices involved are as follows:

	Paper Tape Punch-60		Paper Tape Punch-100		Paper Tape Reader		Flexowriter	
Instruction	WAN*		WAN		RAN**		WAN	
Characters/Second	60		100		270		10	
Words/Second	$\frac{(IS)'}{60/6}$ (10)	$\frac{IS}{60/7}$ (8.7)	$\frac{(IS)'}{100/6}$ (16.7)	$\frac{(IS)}{100/7}$ (14.3)	$\frac{(IS)'}{270/6}$ (45)	$\frac{IS}{270/7}$ (38.6)	$\frac{(IS)'}{10/6}$ (1.7)	$\frac{IS}{10/7}$ (1.4)

Modifying the Instruction Register (ISR) or modification after the WBC is zero.

The ISR can be changed successfully only when magnetic tape is involved and the WBC is equal to zero. In the case of magnetic tape the converter is examined before the tape is completely disconnected and if the order has been modified or the WBC given some new value other than zero the new order is properly implemented. Changes from reading to writing, from the forward to reverse mode are acceptable only when made after WBC reaches zero.

Timing here is not a problem since once the WBC is found to be zero there is ample time before the tape is disconnected to perform the modification.

* Write Alphanumeric
**Read Alphanumeric

8.3 EXAMPLE 6: "ORDER" MODIFICATION AFTER WBC IS ZERO

Write two blocks of 510 words each onto magnetic tape in the non-ISN mode. The first block starts from 1000 and the second from 2000.

```
START     SNR     *+1, 0, ISN
          WAN     1000, MT, 510
          .
          .
          .
          .
          CLA     K
GO        TRC     CNV
          TRU     GO
          HLT
          LOD     NEWOP, 0, CNV
          HLT
K         WAN     1511, MT, 0
NEWOP     WAN     2000, MT, 510
          END     START
```

The interblock gap is of normal size even though passed over with the tape moving.

SECTION IX

CONVERTER ACTION DURING THE RAN INSTRUCTION

9.1 THETA FLIP-FLOP: θ_1 AND θ_2

After the transfer of the instruction to the selected CIS^i, the central processor continues with the basic cycle. The processing of the in-out instruction by the converter now proceeds independently of the basic cycle.

The converter has its own timing function, controlled by the combined state of the Theta flip-flops θ_1 and θ_2. There are four possible combinations of these flip-flops and these are used to subdivide the converter's timing function. Since the flip-flops are controlled by the converter, the length of any particular subdivision is variable and depends on the conditions of the instruction being processed.

9.1.1 Subdivisions

The four subdivisions are:

	θ_1	θ_2
1.	0	0
2.	1	0
3.	1	1
4.	0	1

A change from one subdivision to another is only possible when the state of one of the flip-flops is changed during the subdivision being executed. The action of the converter for a RAN instruction (Flow Chart No. 6) is described on the following pages.

9.1.1.1 $\theta_1 = 0$; $\theta_2 = 0$

a) The CIR is cleared.

b) CF is set to 0 if a Block Start Mark (BLS) has reached the CIR.

c) If BSR is non zero (contains either a control or data character) then go to (d), otherwise back to (a).

d) Contents of BSR are transferred to CIR (For RAN the first time this can happen is when the BLS is read). BSR is then cleared.

e) If CF is one go to (a). (CF is one when the instruction is transferred to CIS.) If CF has been set to 0 go to (f).

f) CF is 0 and the BSR has been transferred to the CIR so that CIR contains a data character, now the θ_1 flip-flop is set to 1.

1) This is the subdivision where BLS marks are recognized and after one is received the converter waits until a character is in the CIR before going on.

9.1.1.2 $\theta_1 = 1$; $\theta_2 = 0$

a) If the character in the CIR is the first character of a word the converter checks to be sure that it is a legal sign if IS = 1 or if it is a BLE mark. If ISN was set (1) and the character is neither a Legal sign or a BLE mark the Interpret Sign Error (ISE) flip-flop is set and Flow Chart No. 5 should be examined to determine the next step.

If ISN is reset and the character is not a BLE mark N is set to 1 (so only 6 characters will be included in the word).

b) If the busy bit (BB) has been set but not yet acknowledged the converter waits until the memory access has been made and the busy bit cleared.

c) θ_2 flip-flop is set to 1.

This timing subdivision checks the first character of a word, recognizes End Block Marks (BLE) and waits until the busy bit is cleared before going on. If the character is not the first one, nothing happens and step (c) is performed.

9.1.1.3 $\theta_1 = 1$; $\theta_2 = 1$

a) The BFR is shifted left 3 places (1 = SLB*).

b) If N is not 0 or the character in the CIR is not a BLE mark, the N counter is incremented by 1.

c) The BFR is shifted left 3 places (1 = SLB*).

d) Now the converter determines if the busy bit (BB) should be set and if the WBC should be decremented according to the table below.

device	WB	N	IS	BLE	BB Set	WBC Decremented
MT	0	7	1/0	No	Yes	Yes
MT	1	7	1/0	No	Yes	No
MT	1	0	1/0	Yes	No	Yes
Input	0	2	0		Yes	No
Input	1	7	0		Yes	Yes (until WBC = 0)
Input	0/1	7	1		Yes	Yes

e) If a parity error is detected in the CIR, the Interpret Parity Error flip-flop (IPE) is set. Flow Chart No. 5 should be examined to see the next action.

f) If the device is an Input Type, N=7 and a stop code is received when WB = 1 and IS = 0 then this subdivision is continued until N = 7 and a full word (filled by zeros) can be assembled in the BFR register.

g) θ_1 is reset to 0.

9.1.1.4 $\theta_1 = 0$; $\theta_2 = 1$

a) If the instruction has been completed, END level is set to 1 (completed state depends on the particular mode involved at this time).

b) If the character in the CIR is not a BLE mark it is transferred into the low character position of the BFR.

c) If N = 0 and the character is a BLE mark, the CF flip-flop is set (1 = CF).

d) If N = 7 it is reset to 0.

e) If a BLS mark is in the CIR or a BLE mark has been received when $N \neq 0$ then the Timing Read Error is set (TRE) and the CH flip-flop is set.

9.2 PROCESSING A RAN ORDER

The following chart shows portions of the converters action while processing a RAN order for 200 words. These words are spread out over two blocks on Magnetic Tape.

CF	C(BSR)	C(CIR)	C(BFR)	WB	WBC	ADC	BB	$\theta_1\theta_2$	N	Comment
1	0	0	0000000	0	200	1000	0	00	0	Start
1	BLS	0	"	"	"	"	"	"	"	
1	0	BLS	"	"	"	"	"	"	"	
0	0	0	"	"	"	"	"	"	"	
"	SA	0	"	"	"	"	"	"	"	
"	A_1	S_N	"	"	"	"	"	"	"	
"	A_1	S_N	"	"	"	"	"	10	"	
"	A_1	S_N	"	"	"	"	"	11	1	
"	A_1	0	000000 S_N	"	"	"	"	01	"	
"	B_1	A_1	"	"	"	"	"	00	"	
"	"	"	"	"	"	"	"	10	"	
"	B_1	A_1	00000 S_N 0	"	"	"	"	11	2	
"	B_1	0	00000 S_N A_1	"	"	"	"	01	2	
"	C_1	B_1	"	"	"	"	"	00	"	
"	"	"	"	"	"	"	"	10	"	
"	C_1	B_1	0000 S_N A_1 0	"	"	"	"	11	3	
"	C_1	0	0000 S_N A_1 B_1	"	"	"	"	01	3	
"	D_1	C_1	"	"	"	"	"	00	"	
"	"	"	"	"	"	"	"	10	"	
"	"	"	000 S_N A_1 B_1 0	"	"	"	"	11	4	
"	D_1	0	000 S_N A_1 B_1 C_1	"	"	"	"	01	"	
"	E_1	D_1	"	"	"	"	"	00	"	
"	"	"	"	"	"	"	"	10	"	
"	E_1	D_1	00 S_N A_1 B_1 C_1 0	"	"	"	"	11	5	
"	E_1	0	00 S_N A_1 B_1 C_1 D_1	"	"	"	"	01	"	
"	F_1	E_1	"	"	"	"	"	00	"	
"	"	"	"	"	"	"	"	10	"	
"	"	"	0 S_N A_1 B_1 C_1 D_1 0	"	"	"	"	11	6	
"	F_1	0	S_N A_1 B_1 C_1 D_1 E_1	"	"	"	"	01	"	
"	S_N	F_1	"	"	"	"	"	00	"	
"	"	"	"	"	"	"	"	10	"	
"	"	"	S_N A_1 B_1 C_1 D_1 E_1 0	"	199	"	1	11	7	BB Set
"	S_N	0	S_N A_1 B_1 C_1 D_1 E_1 F_1	"	"	"	1	01	0	
"	A_2	S_N	"	"	"	"	1	00	0	
"	A_2	S_N	0000000	"	199	1001	0	10	"	Memory Access
"	A_2	S_N	"	"	"	"	"	11	1	
"	A_2	0	000000 S_N	"	"	"	"	01	"	
			The following words are assembled in the same way as the word shown above.							

Figure 9-1. Processing a RAN Order

CF	C(BSR)	C(CIR)	C(BFR)	WB	WBC	ADC	BB	$\theta_1\theta_2$	N	Comment
0	BLE	F_N	$S_N A_N B_N C_N D_N E_N 0$	0	100	1000+N	1	11	7	After N-1 words are in memory
0	BLE	0	$S_N A_N B_N C_N D_N E_N F_N$	0	100	1000+N	1	01	0	Memory Access
0	0	BLE	0000000	"	"	"	1	00	0	
0	"	"	0000000	"	"	1001+N	0	10	0	
0	"	"	"	"	"	1001+N	"	11	0	
1	"	"	"	"	100	1001+N	"	01	"	CF = 1 now wait
"	BLS	"	"	"	"	"	"	00	"	till inter block cap
"	0	BLS	"	"	"	"	"	00	"	has been passed
0	0	0	"	"	"	"	"	00	"	over
0	S_M	0	"	"	"	"	"	00	"	
"	A_1	S_M	"	"	"	"	"	00	"	
"	"	"	"	"	"	"	"	10	"	
"	"	"	"	"	"	"	"	11	1	
"	A_1	0	000000 S_M	"	"	"	"	01	"	
Now words are again assembled as on the previous page until WBC = 0										
0	S_X	F_M	$S_M A_M B_M C_M D_M E_M 0$	0	0	1000+M	1	11	7	WBC = 0
"	"	0	$S_M A_M B_M C_M D_M E_M F_M$	0	0	1001+M	1	01	0	END = 1 here
0	A_X	S_X	0000000	0	0	"	0	00		Memory Access
Now END level disconnects converter and the device is disconnected as soon as a BLE is detected.										

Figure 9-1. Processing a RAN Order (Cont'd)

APPENDIX A

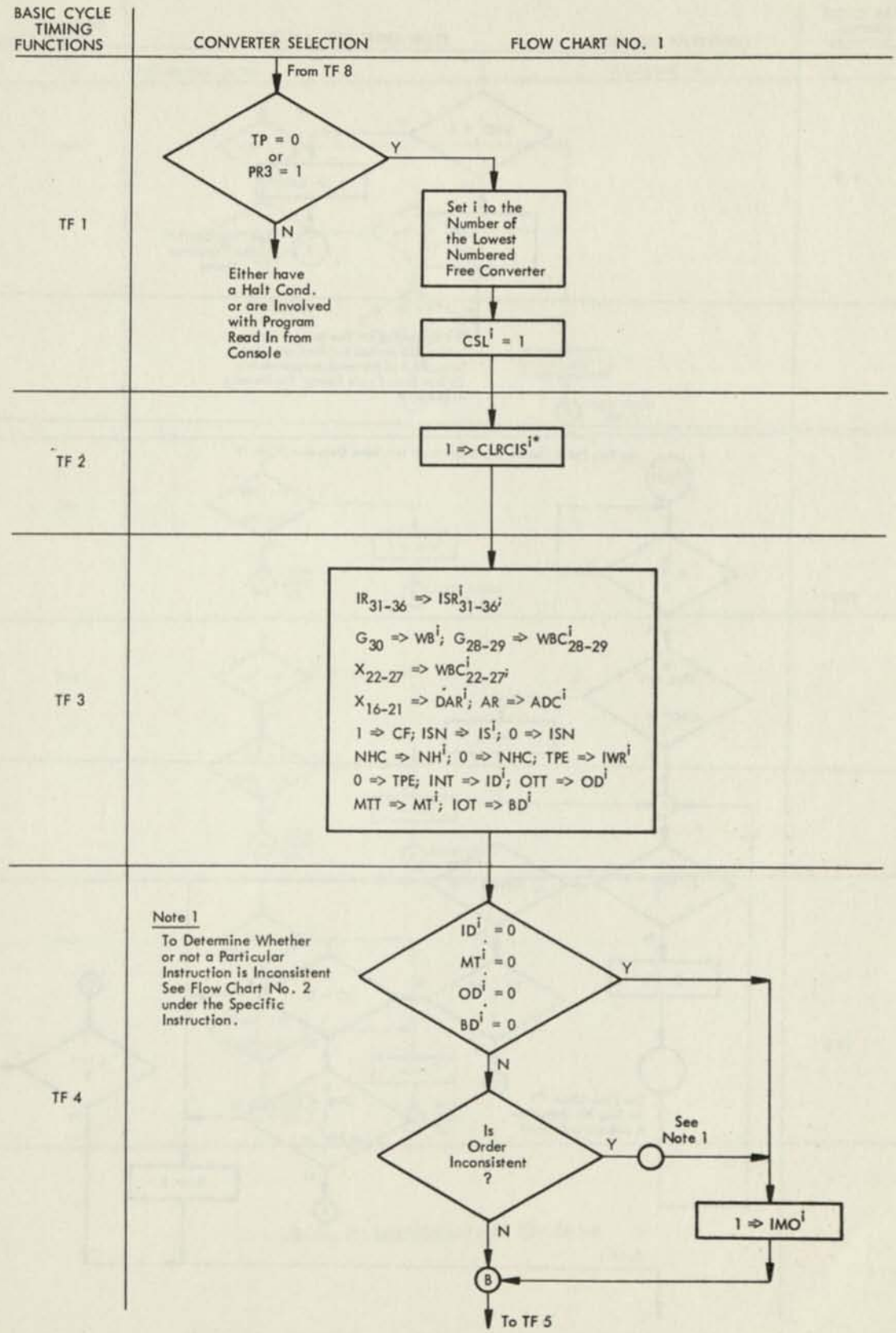
MICRO FLOW CHARTS

A.1 CONVENTIONS USED ON MICRO FLOW CHARTS

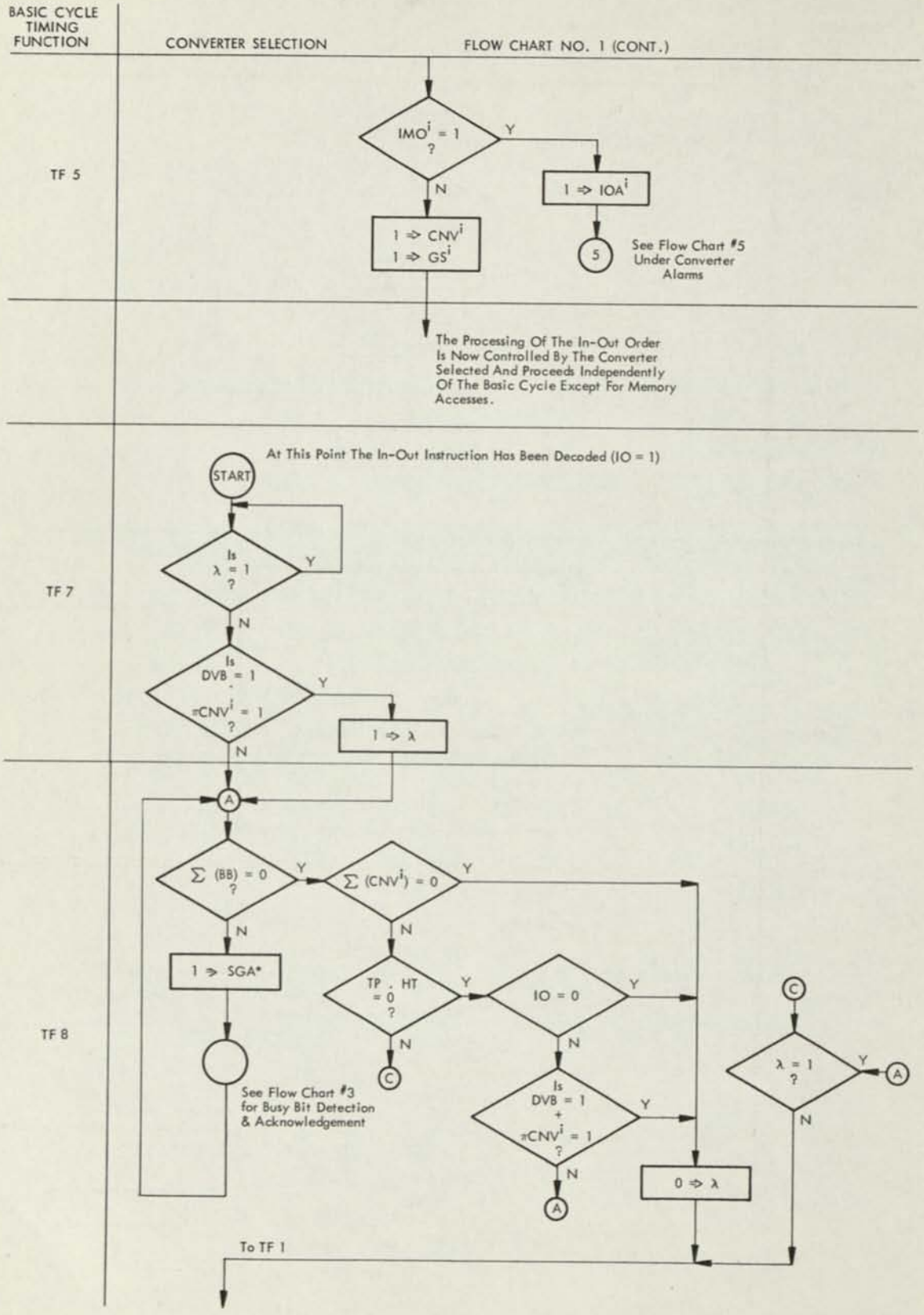
+	=	logical or
·	=	logical and
=>	=	transfer of information

A.2 MICRO FLOW CHARTS

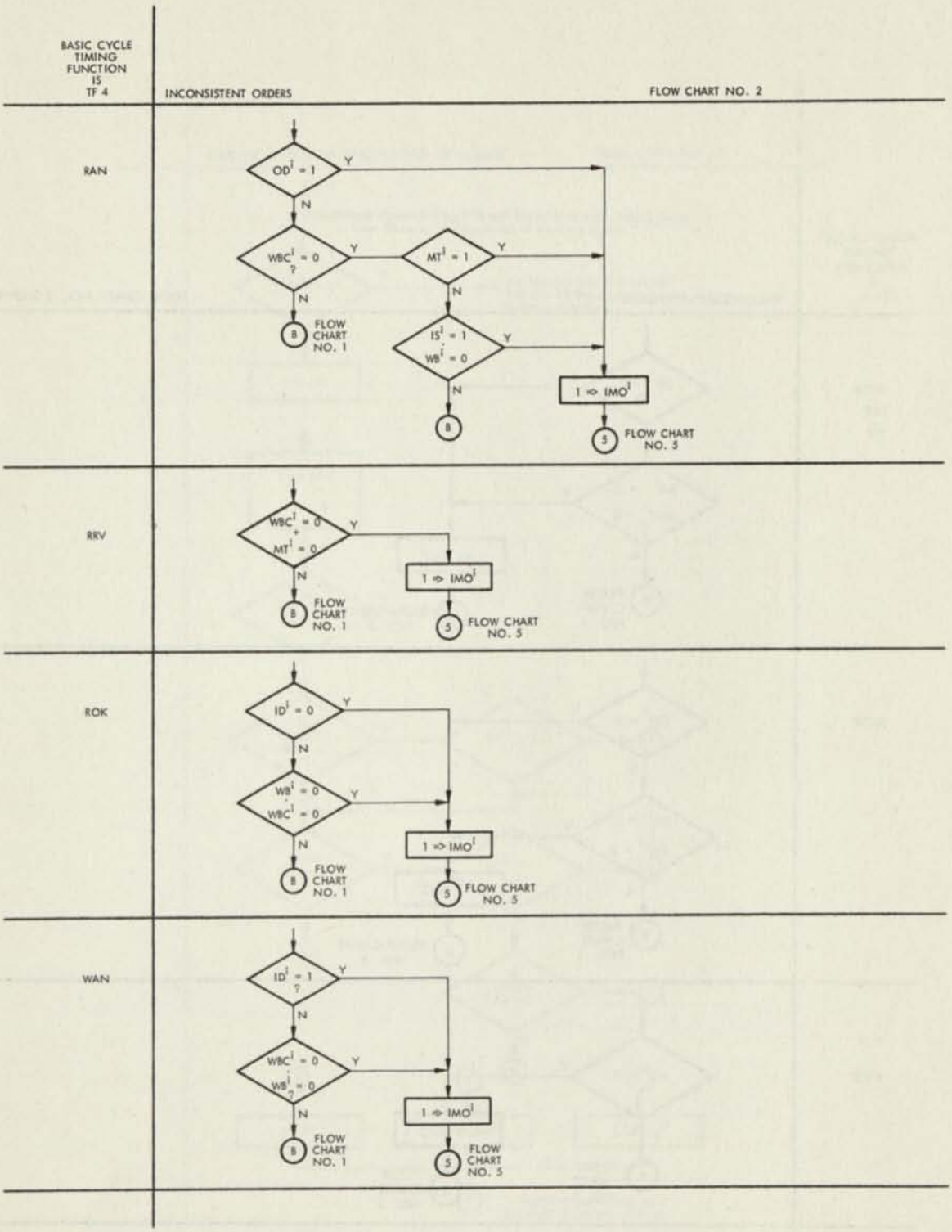
No.	Title
1.	Converter Selection
2.	Inconsistent Orders
3.	Busy Bit Detection and Acknowledgement
4.	In-Out Memory Cycle
5.	Converter Stop Logic
6.	Read Alphanumeric
7.	Read Reverse
8.	Read Octal
9.	Write Alphanumeric
10.	Rewrite Alphanumeric
11.	Write Octal
12.	Skip
13.	Backspace



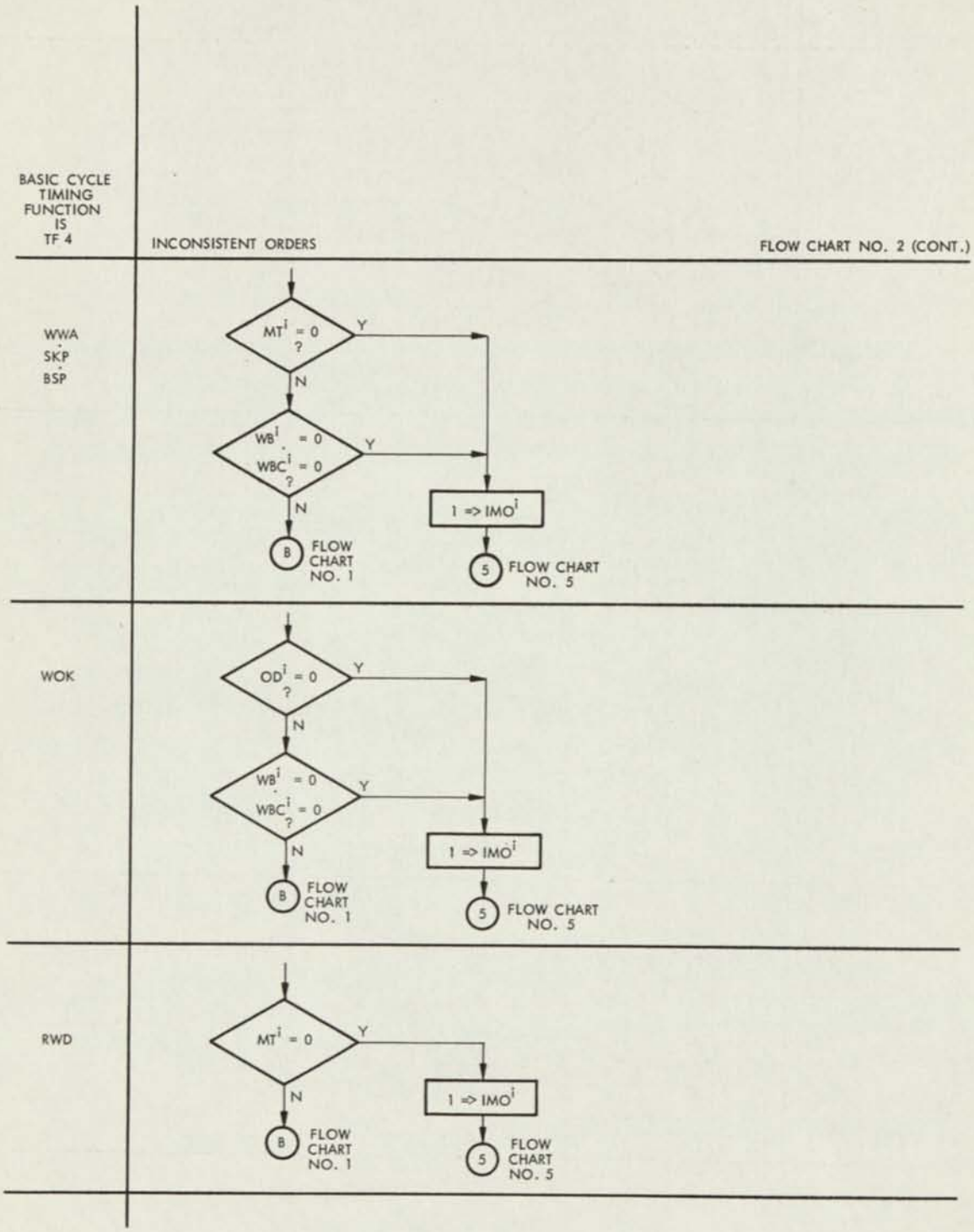
A. 2. 1 Converter Selection



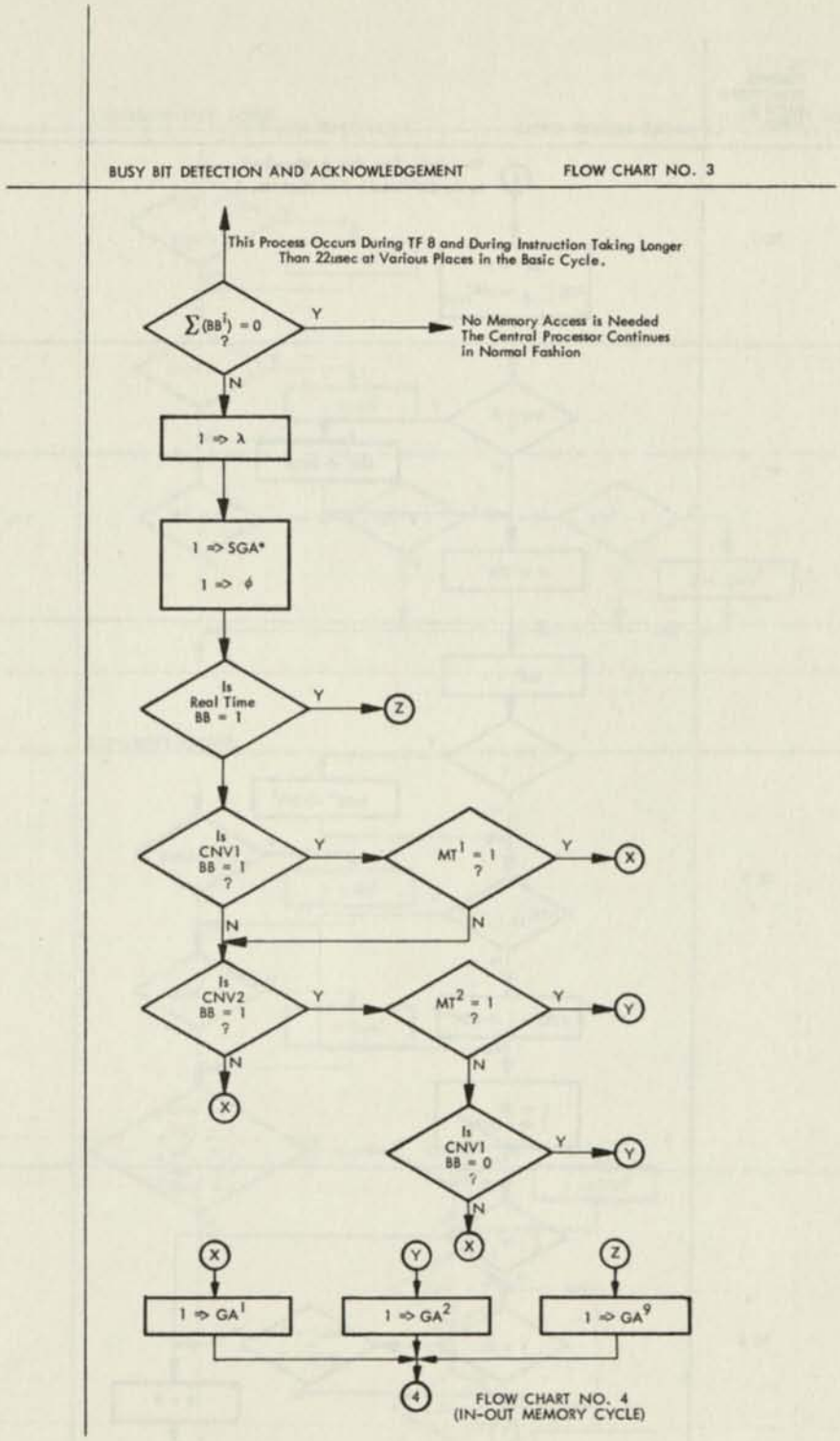
A.2.1 Converter Selection (Continued)



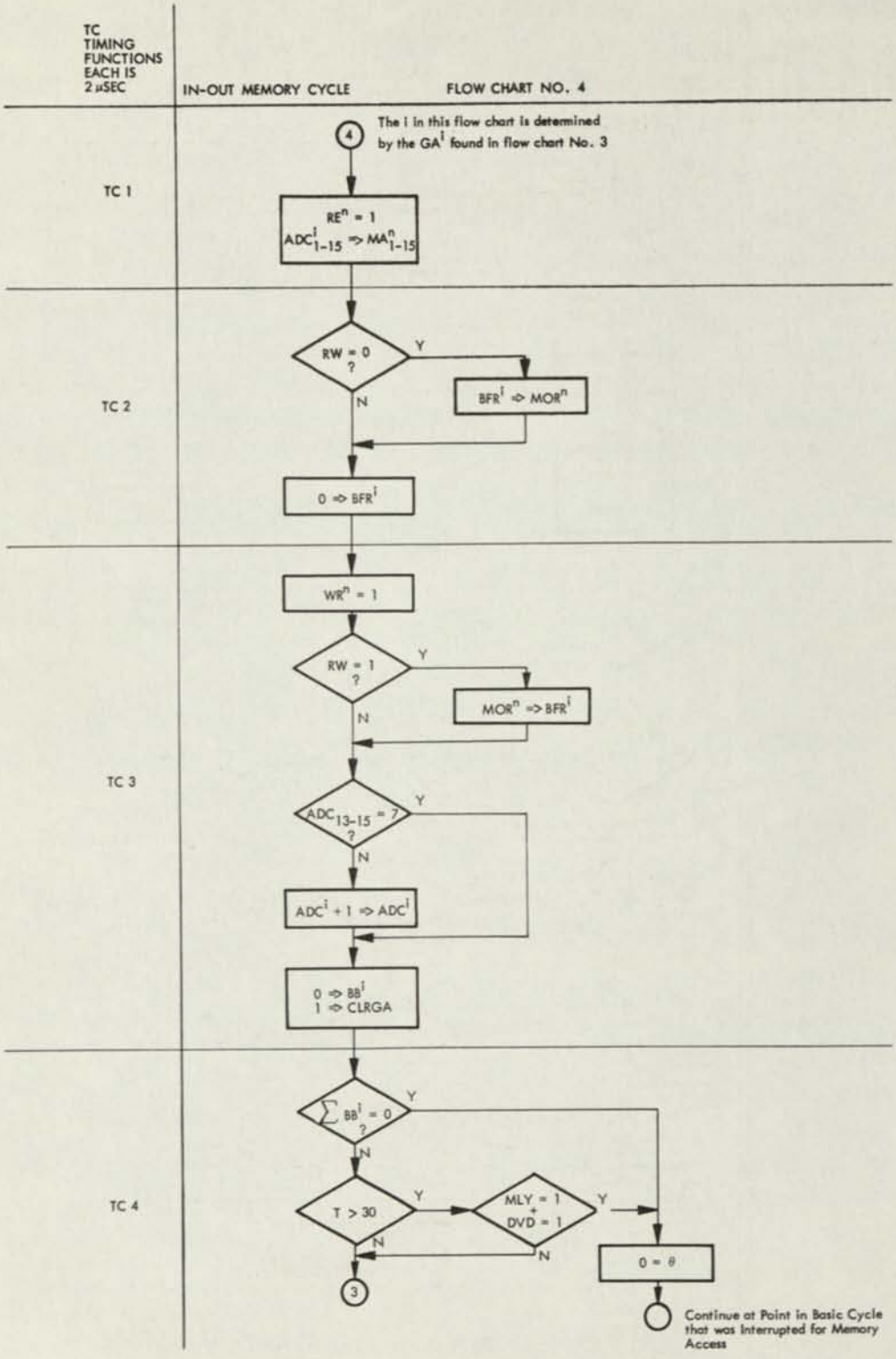
A. 2. 2 Inconsistent Orders



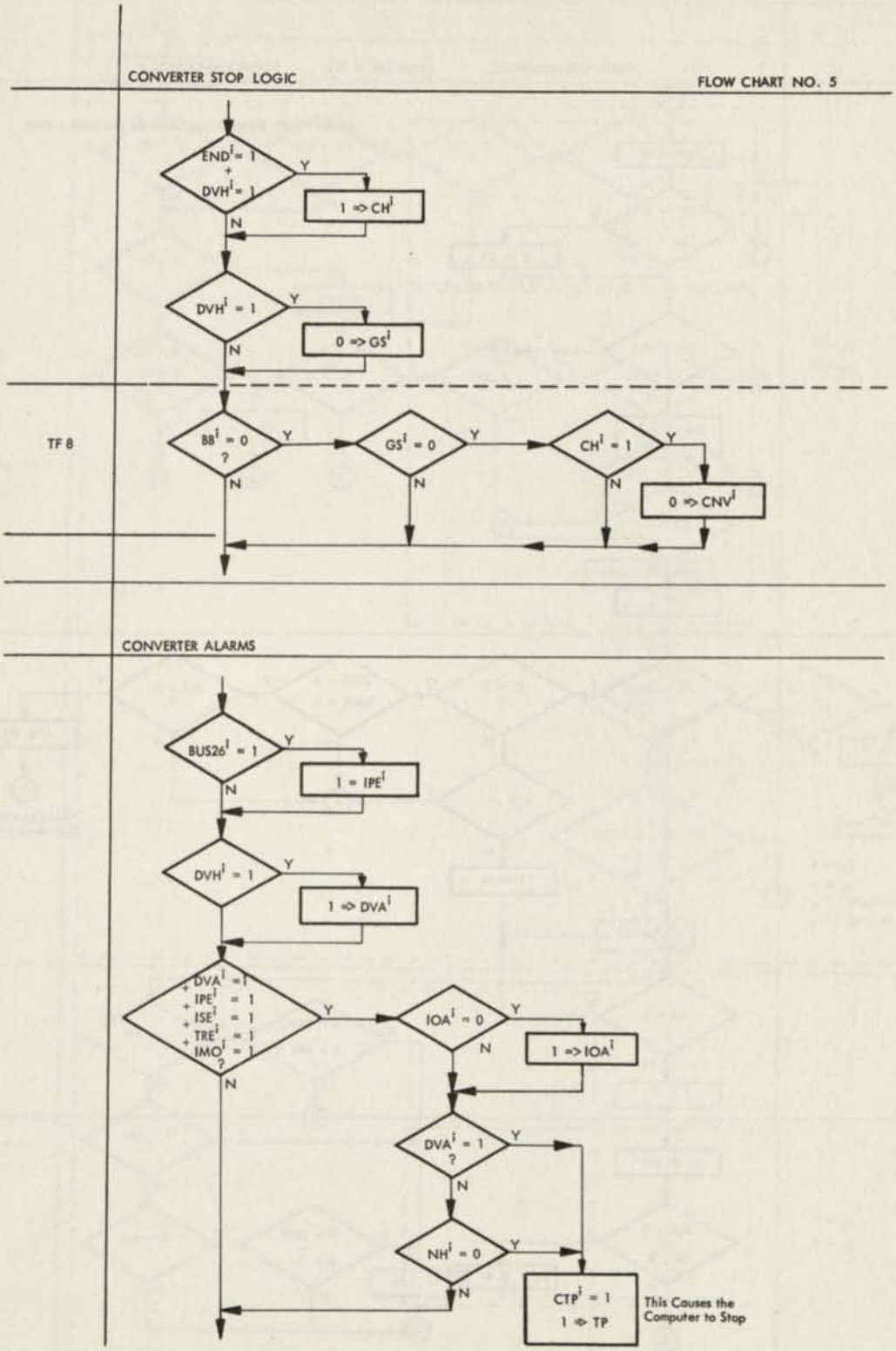
A.2.2 Inconsistent Orders (Continued)



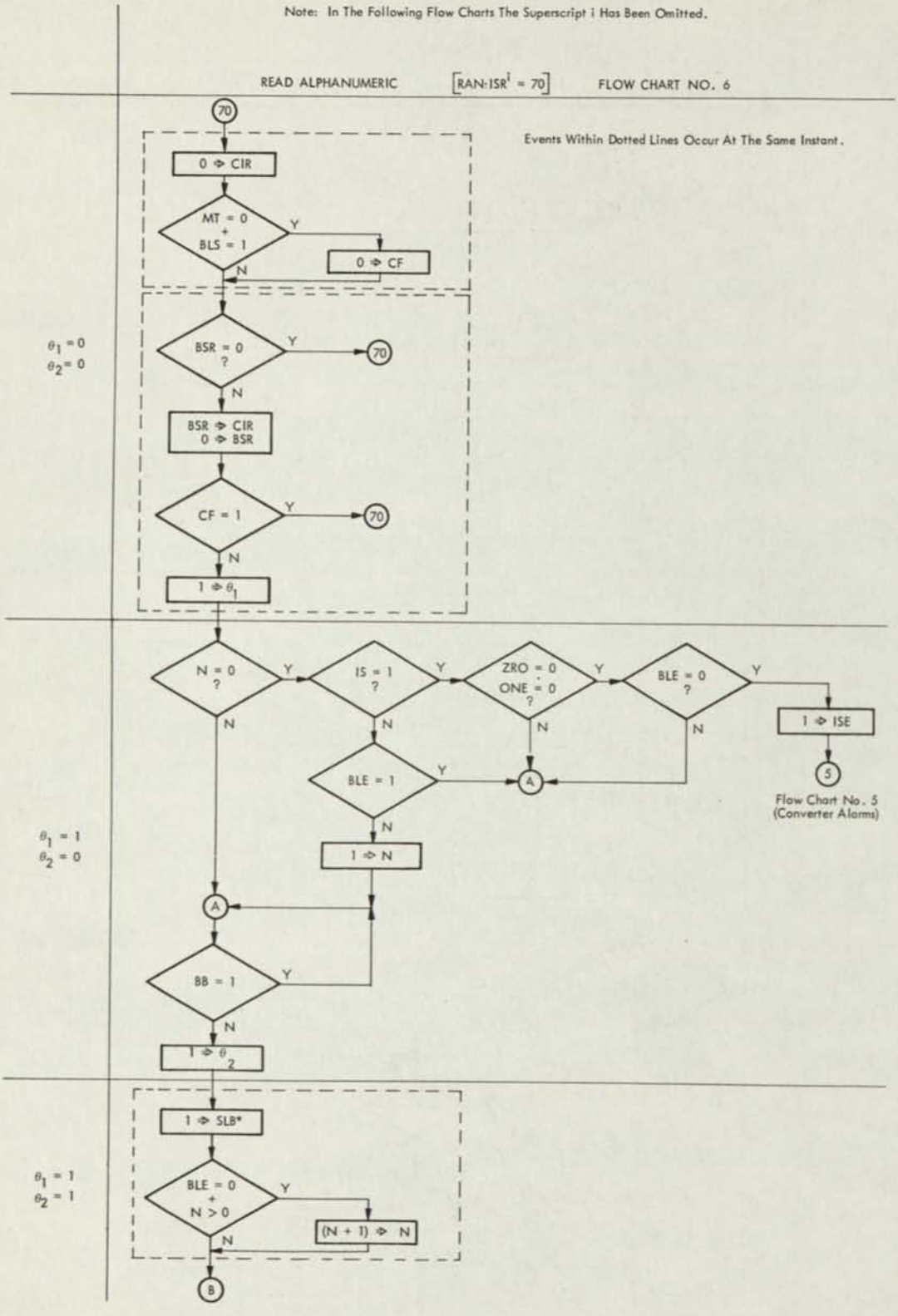
A. 2. 3 Busy Bit Detection and Acknowledgement



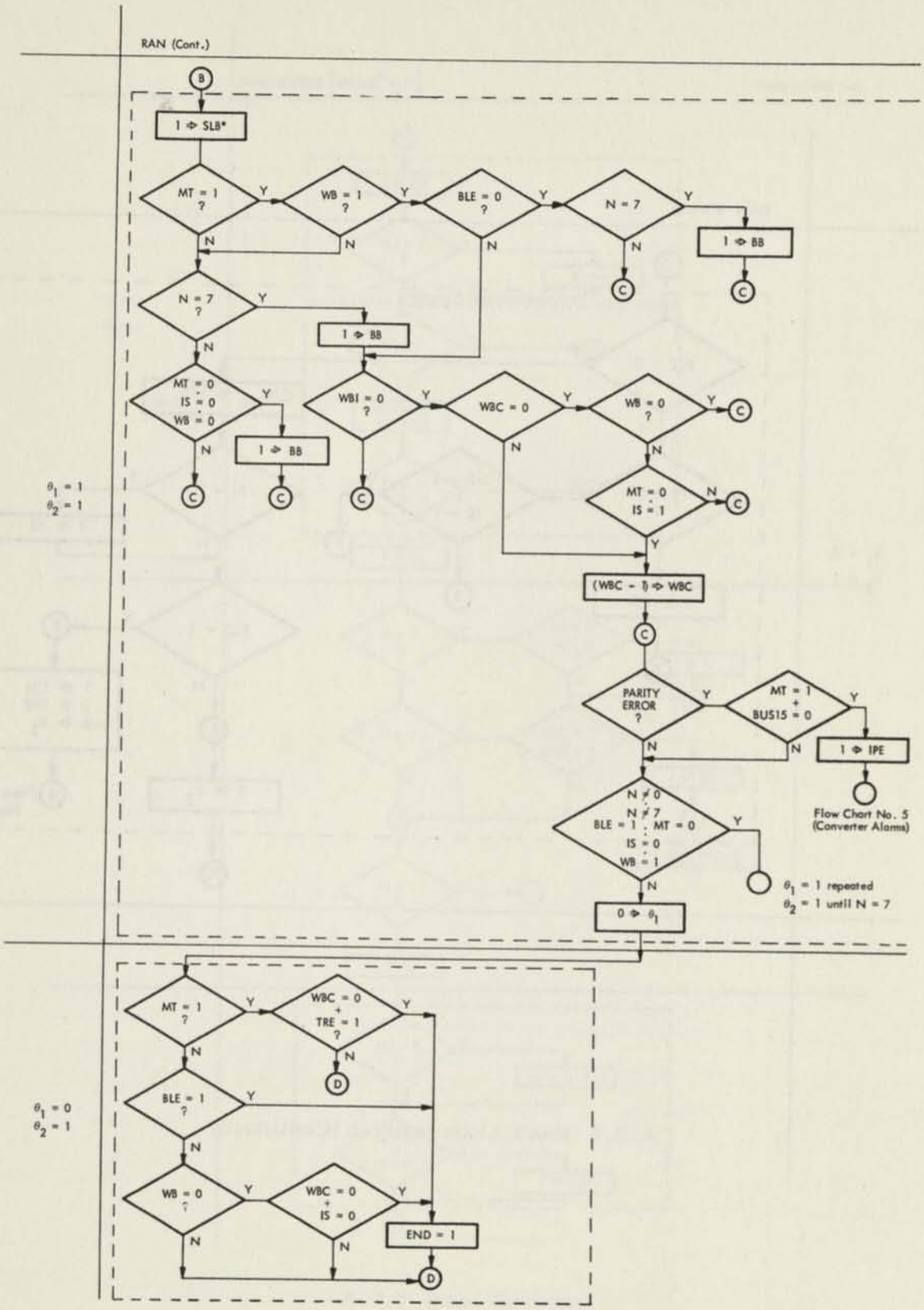
A. 2. 4 In-Out Memory Cycle



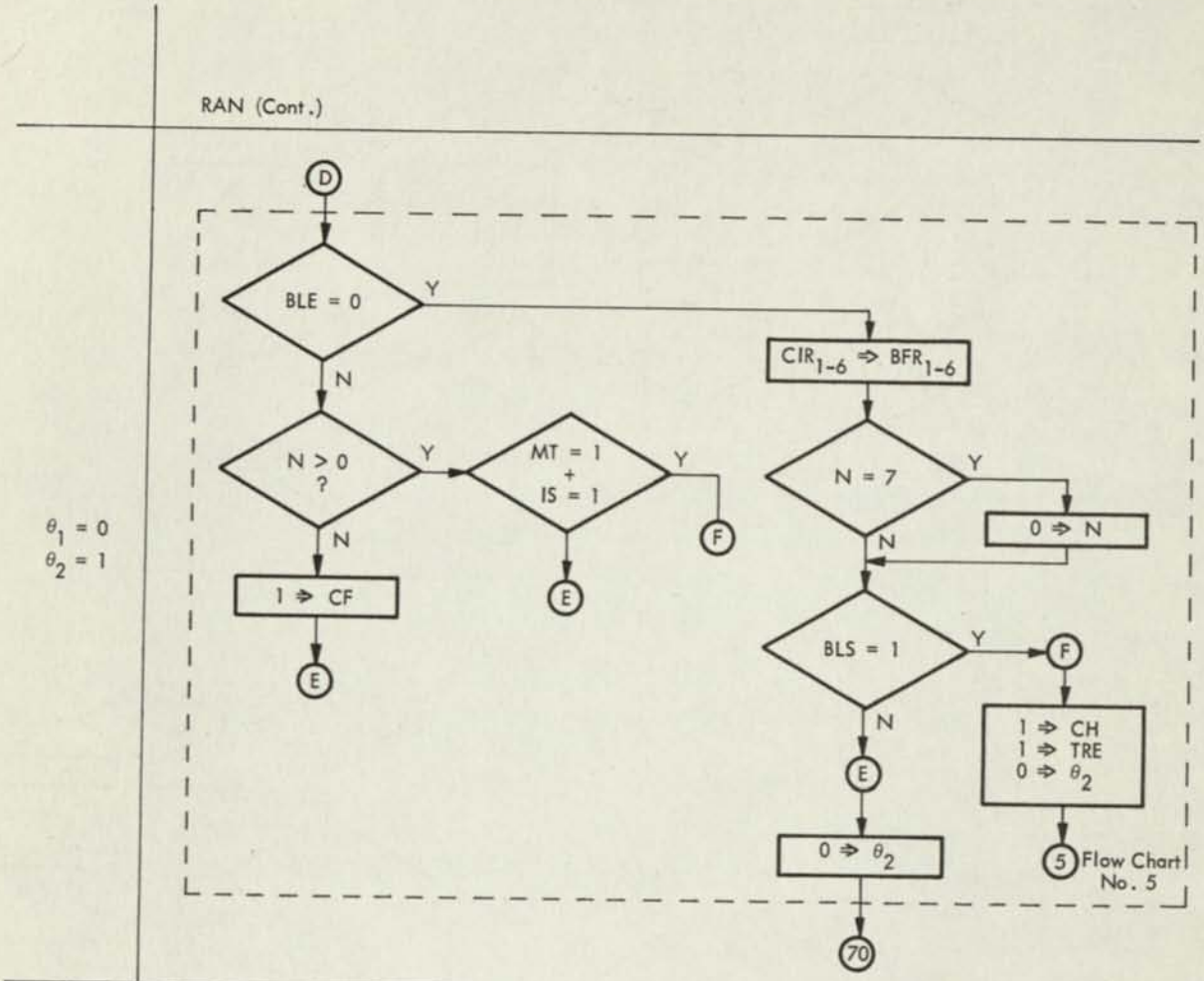
A. 2. 5 Converter Stop Logic



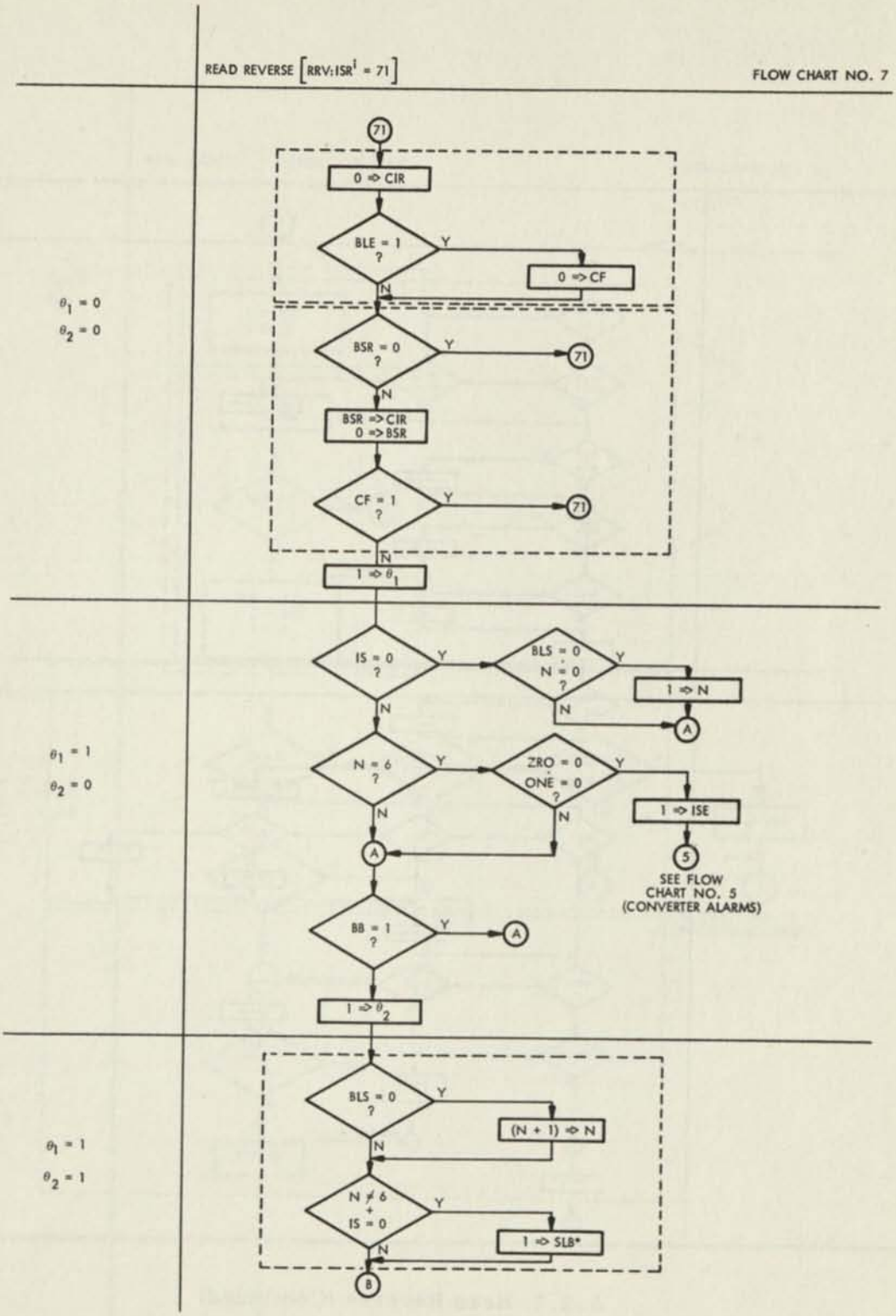
A.2.6 Read Alphanumeric



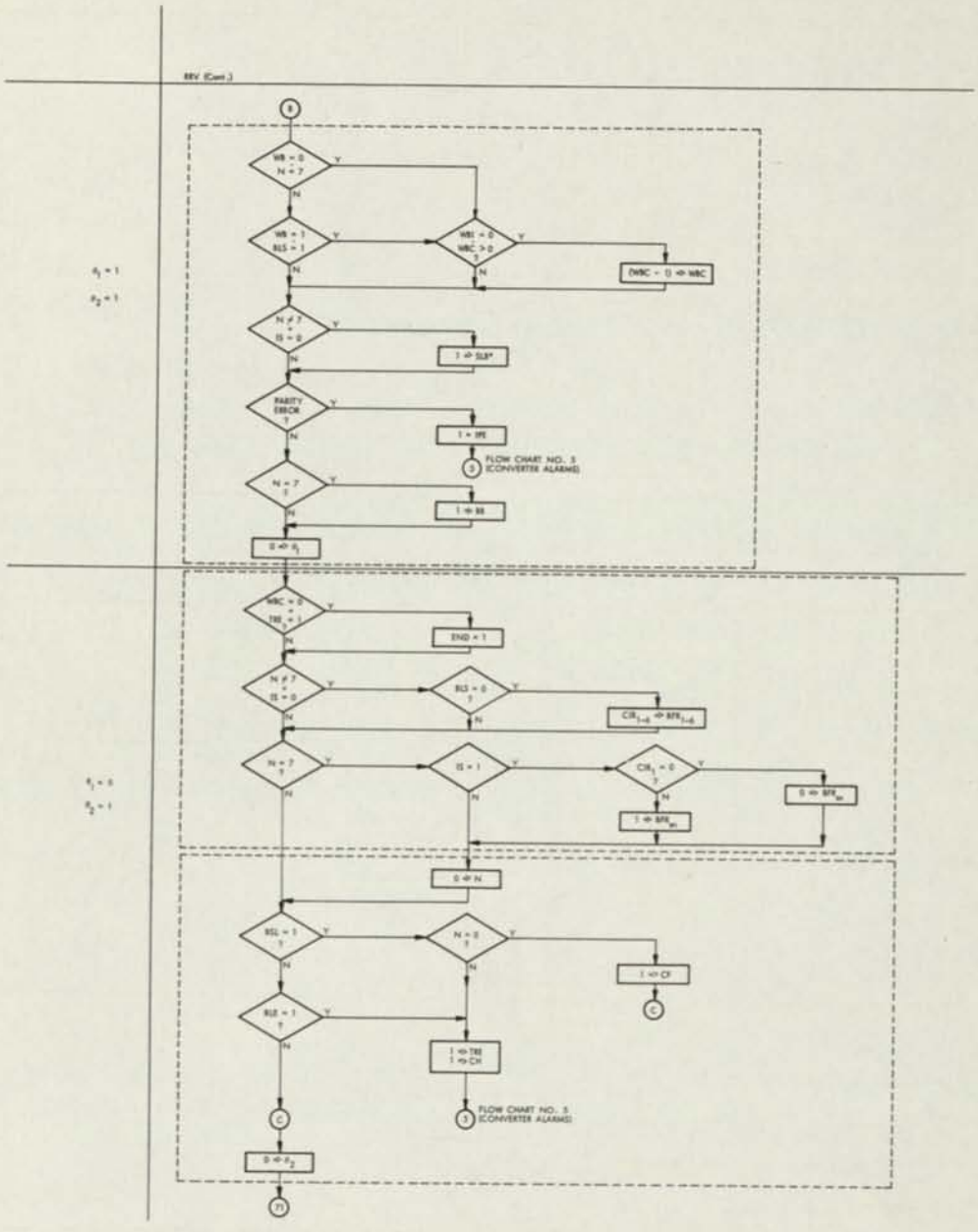
A.2.6 Read Alphanumeric (Continued)



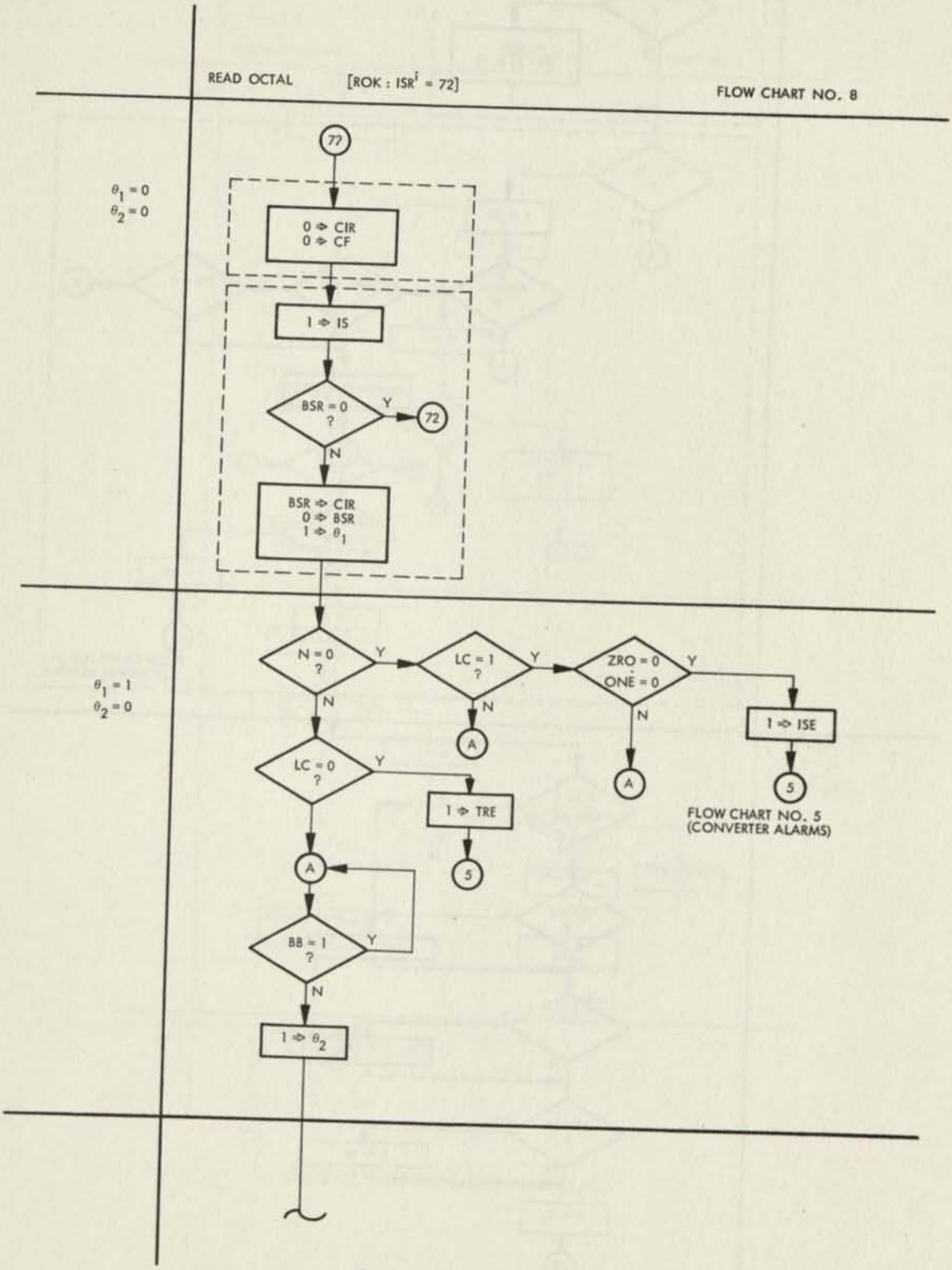
A.2.6 Read Alphanumeric (Continued)



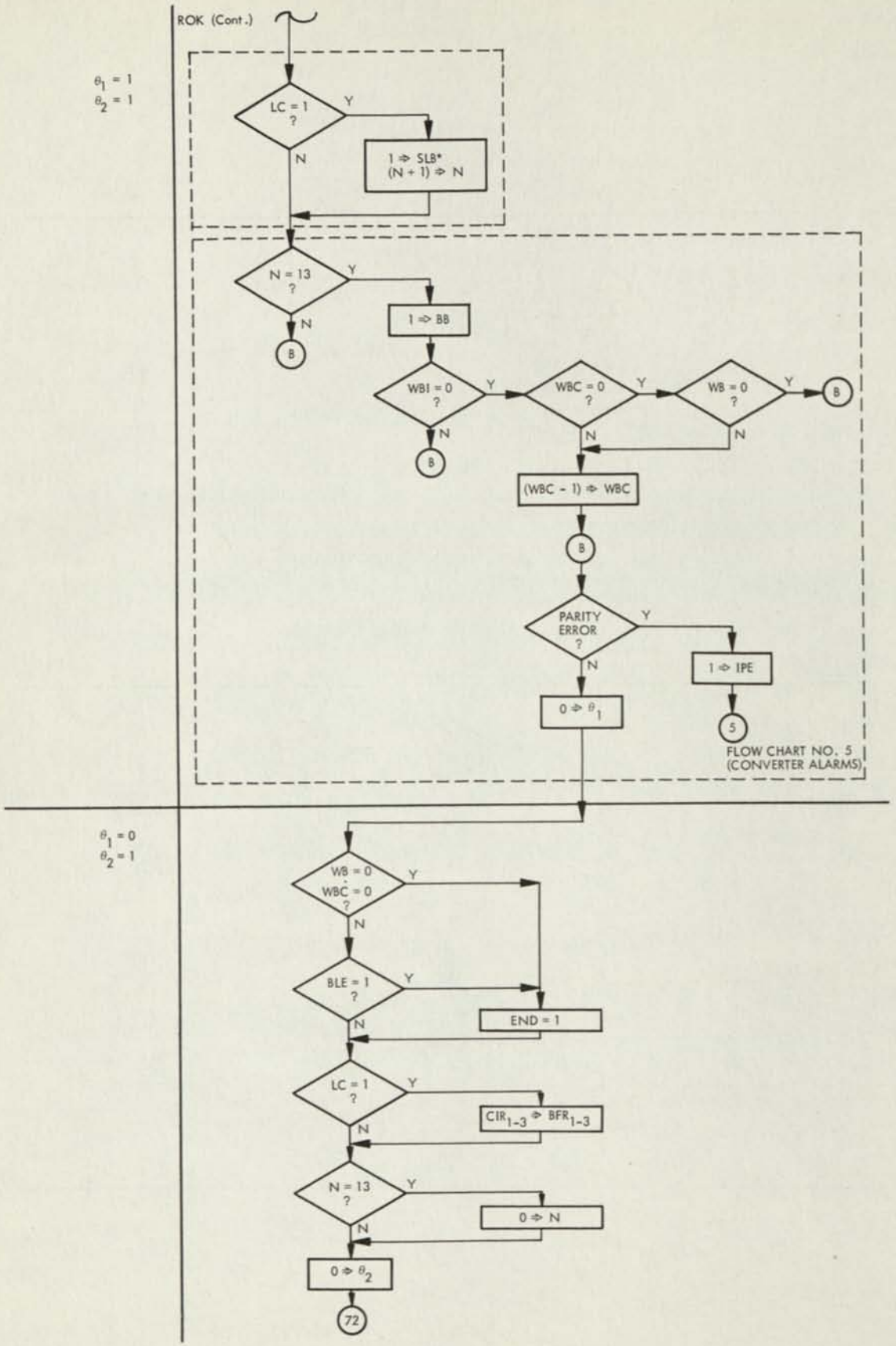
A. 2. 7 Read Reverse



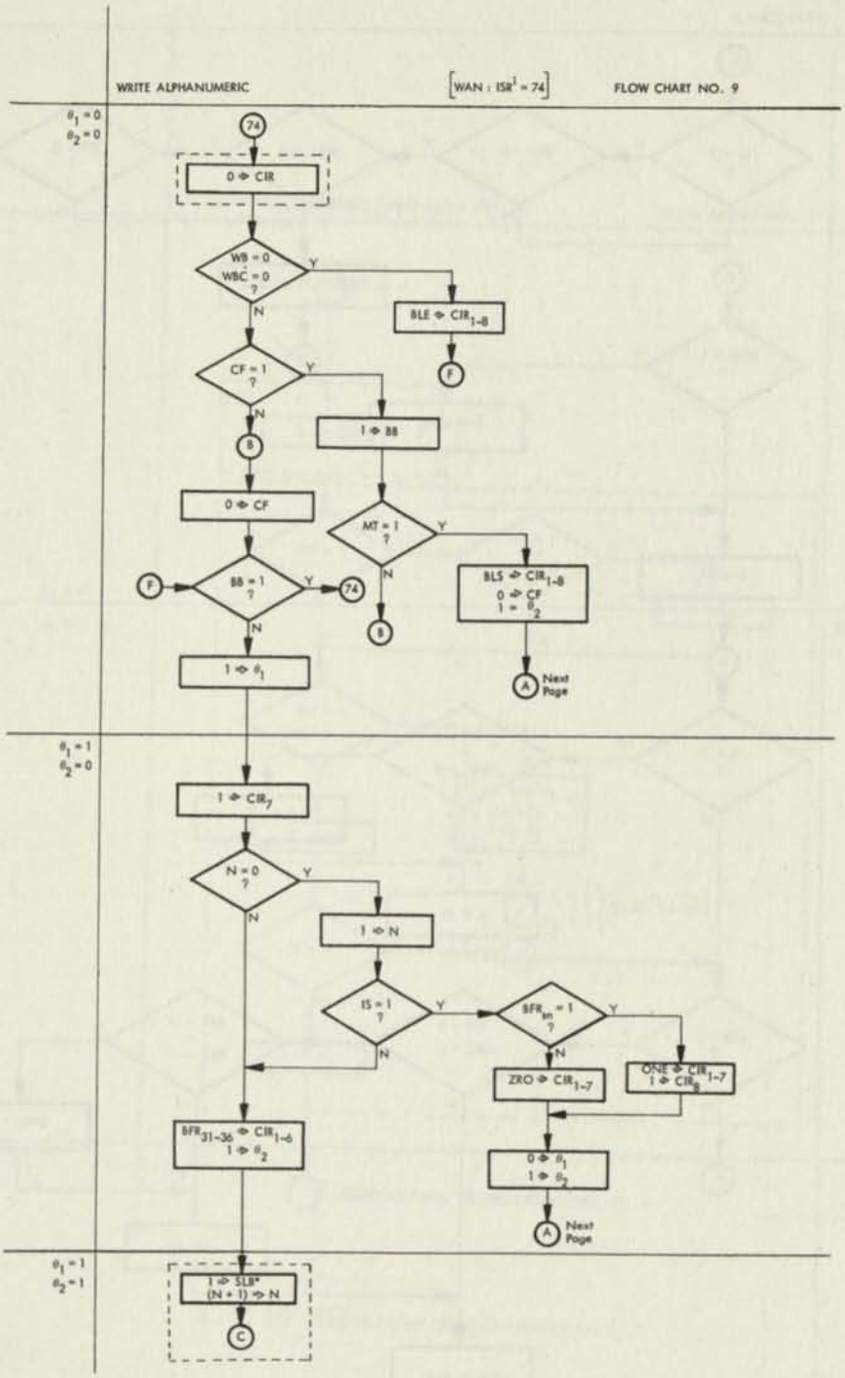
A.2.7 Read Reverse (Continued)



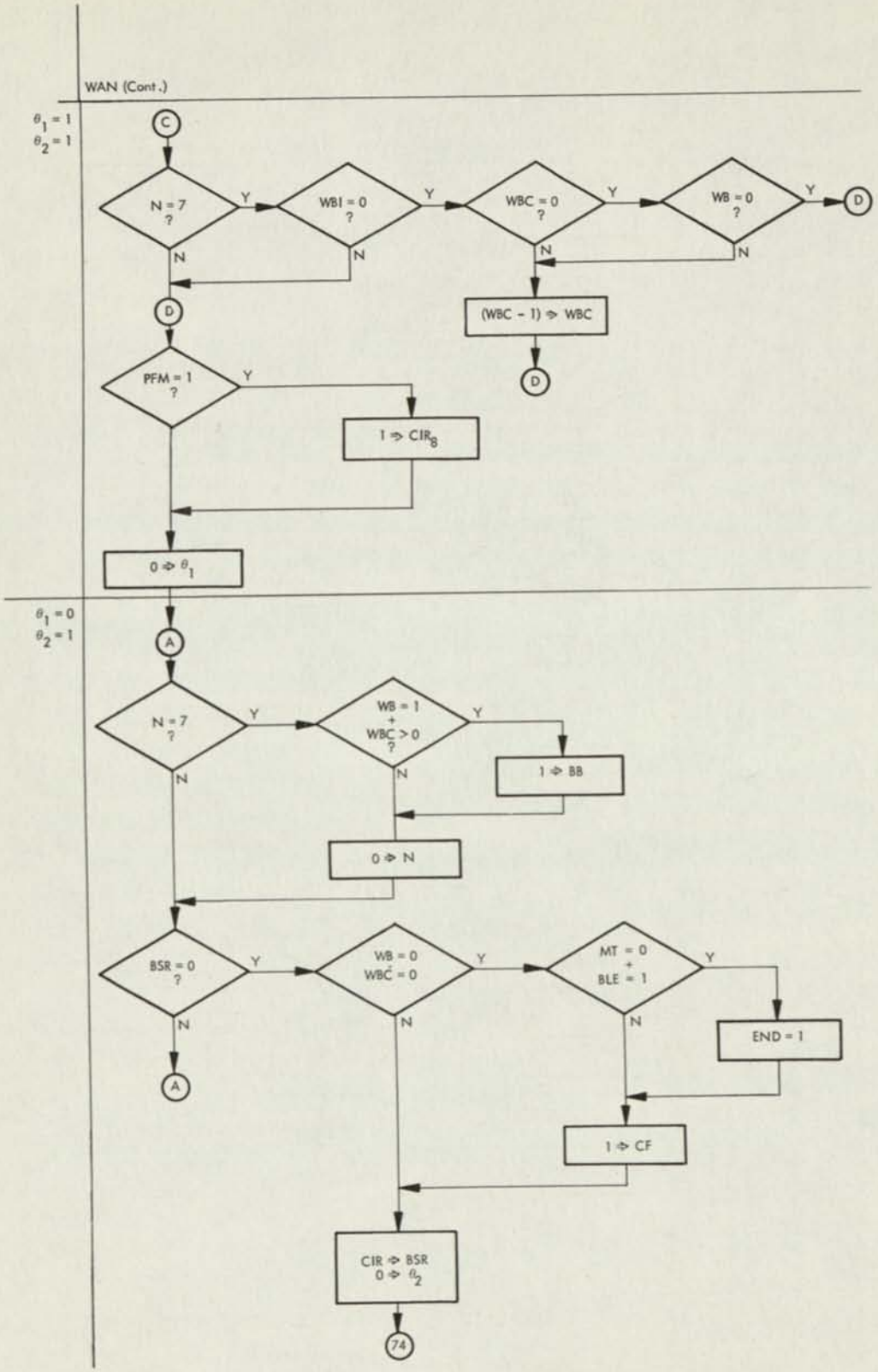
A. 2. 8 Read Octal



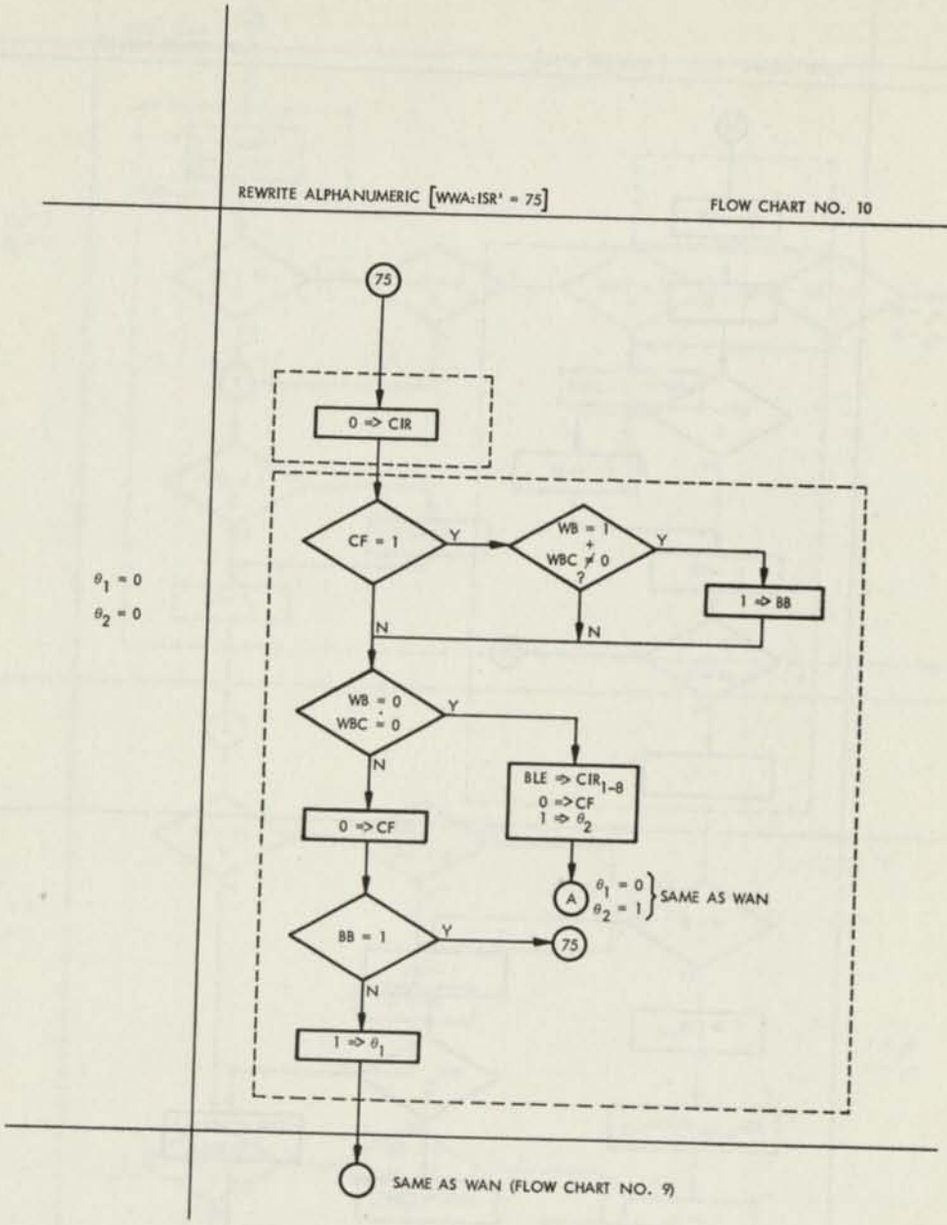
A.2.8 Read Octal (Continued)



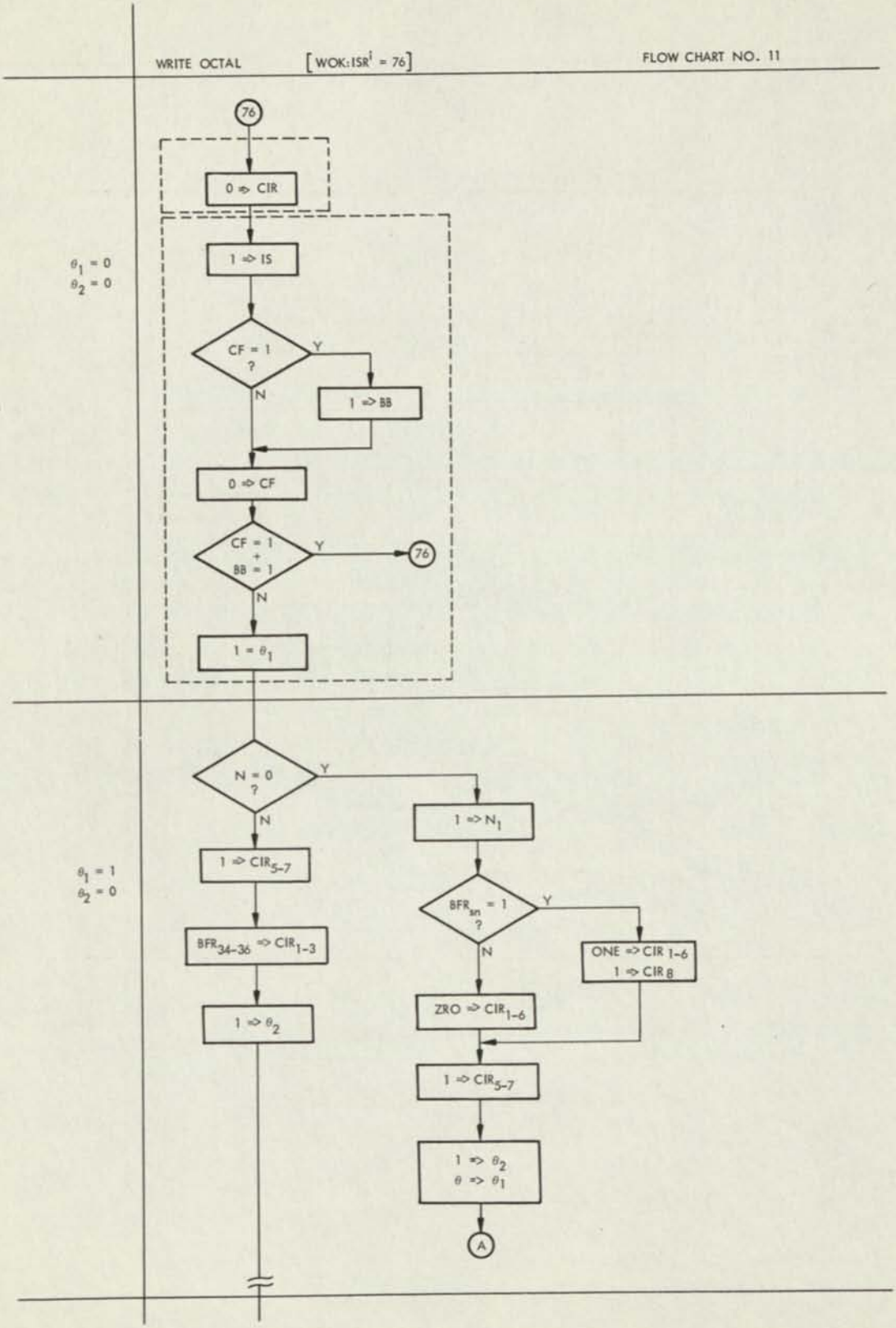
A. 2. 9 Write Alphanumeric



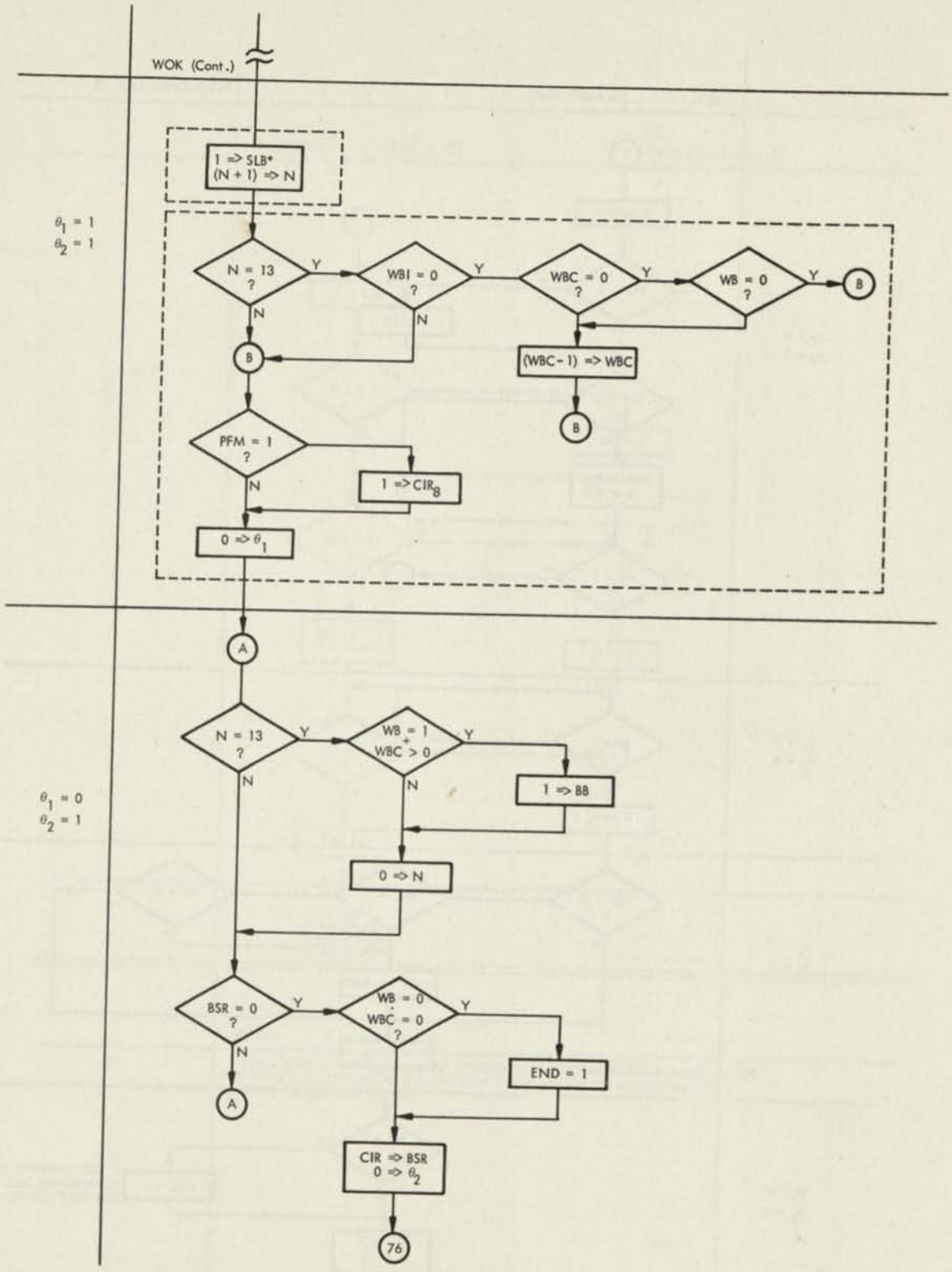
A.2.9 Write Alphanumeric (Continued)



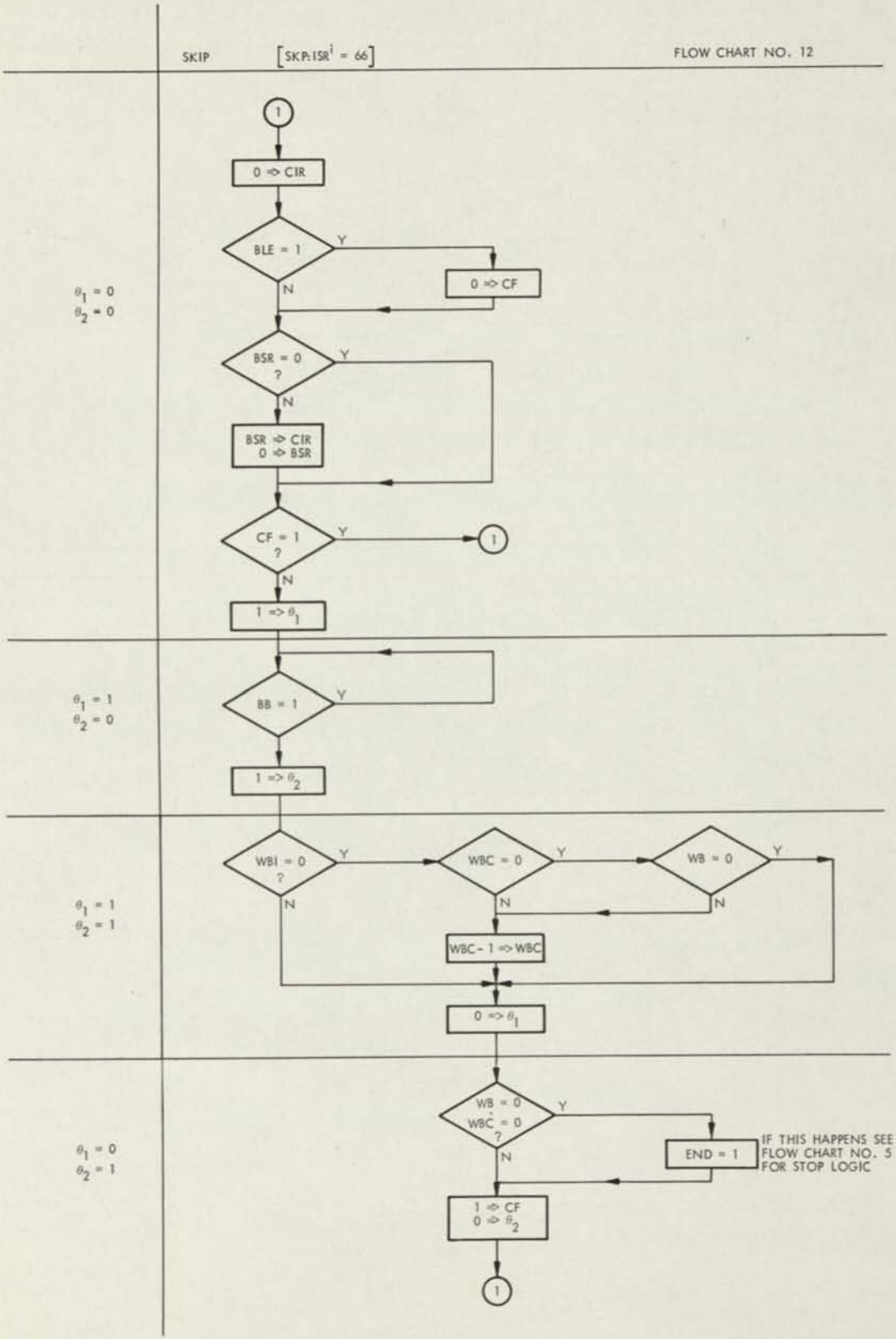
A. 2. 10 Rewrite Alphanumeric



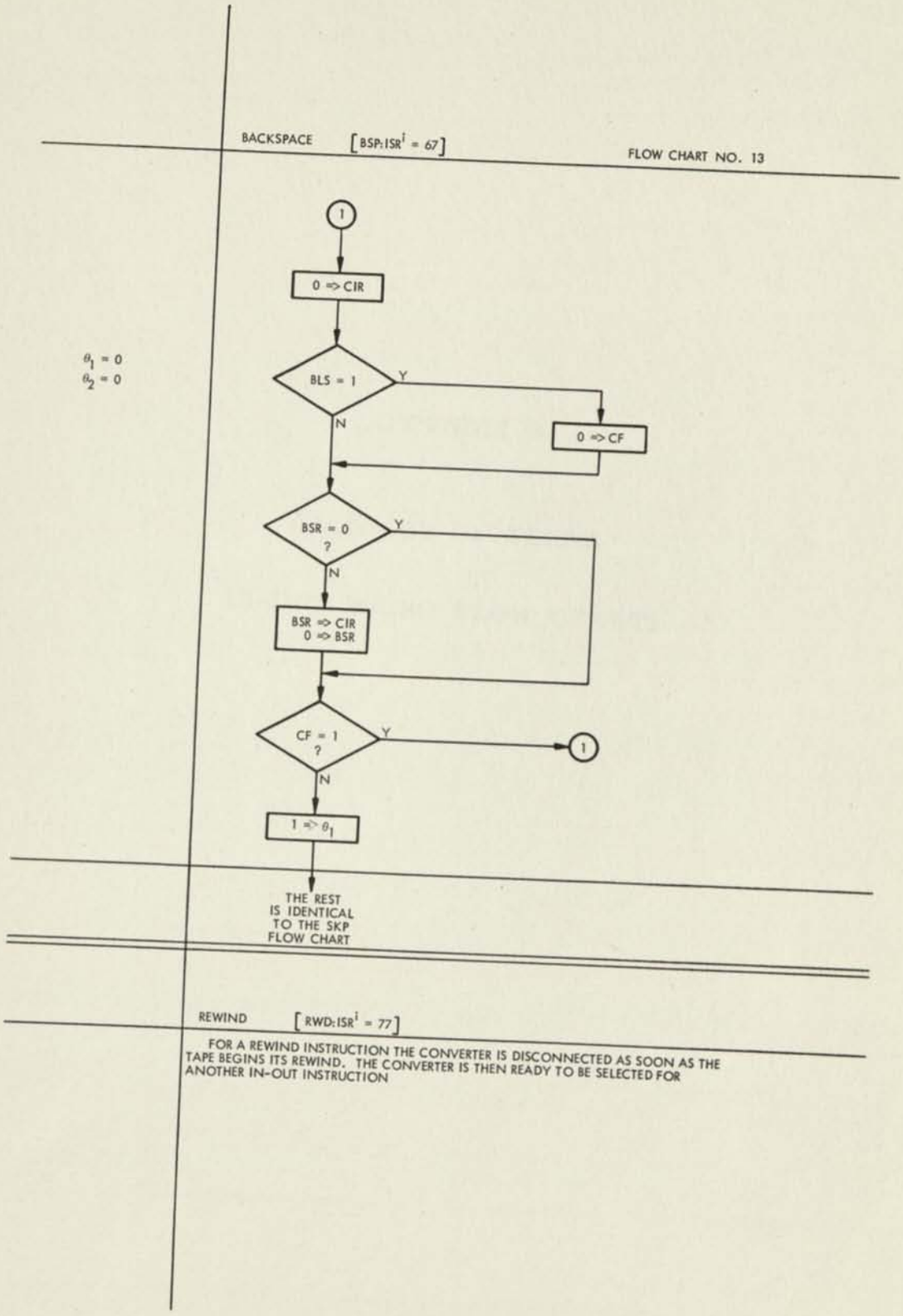
A. 2. 11 Write Octal



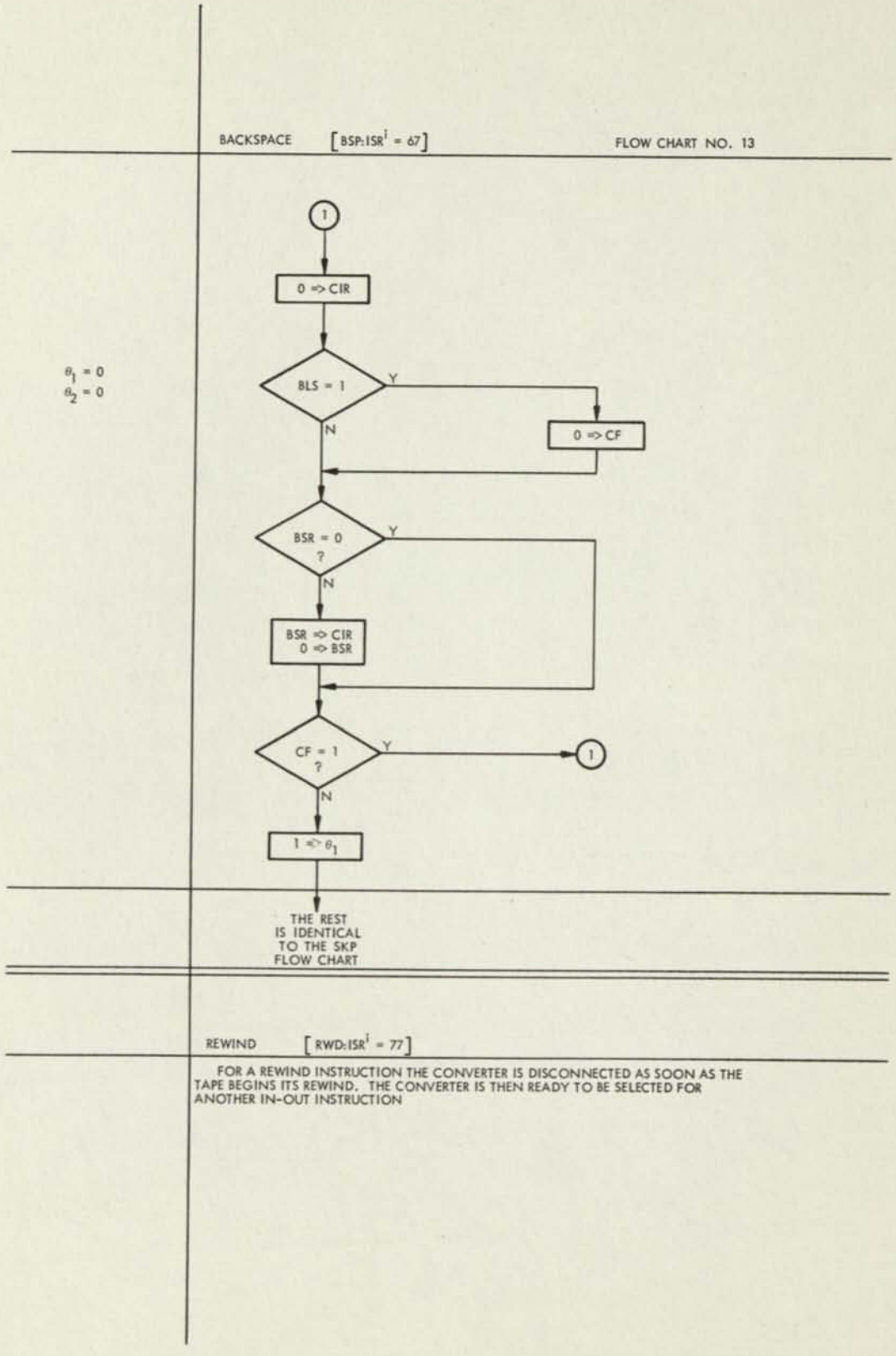
A.2.11 Write Octal (Continued)


SKIP
[SKP:ISR^i = 66]
FLOW CHART NO. 12
θ1 = 0
θ2 = 0
0 ⇒ CIR
BLE = 1
Y
N
0 ⇒ CF
BSR = 0 ?
Y
N
BSR ⇒ CIR
0 ⇒ BSR
CF = 1 ?
Y
1
N
1 ⇒ θ1
θ1 = 1
θ2 = 0
BB = 1
Y
1 ⇒ θ2
θ1 = 1
θ2 = 1
WBI = 0 ?
Y
N
WBC = 0
Y
N
WB = 0
Y
N
WBC - 1 ⇒ WBC
0 ⇒ θ1
θ1 = 0
θ2 = 1
WB = 0
WBC = 0 ?
Y
N
END = 1
IF THIS HAPPENS SEE FLOW CHART NO. 5 FOR STOP LOGIC
1 ⇒ CF
0 ⇒ θ2
1

A. 2. 12 Skip



A. 2. 13 Backspace



A. 2. 13 Backspace

APPENDIX B

GLOSSARY OF TERMS

IN-OUT MICRO FLOW CHARTS

1. ADC — 15 bit address counter in converter i.
2. AR — Address Register
3. BB — The busy bit flip-flop of converter i. When equal to one the converter is ready for memory access.
4. BFR — 37 bit Buffer Register in converter i.
5. BLE — Level equal to one whenever CIR contains the code for an end of block mark (10001001).
6. BLS — Level equal to one (high) whenever CIR contains the code for a start of block mark (10100100).
7. BSP — Converter instruction register output when instruction is "Backspace".
8. BSR — An eight bit character buffer register.
9. BXR — An eight bit character buffer register.
10. CF — A control flip-flop in converter i; used in conjunction with block marks for in-out orders involving magnetic tape.
11. CH — A control flip-flop in converter i. Used in conjunction with the converter stop logic. When set halts the operation of the converter.
12. CIA — Level which when high (1) clears In-Out Alarm flip-flop in converter i.
13. CIR — An eight bit character level in converter i.
14. CLRC1* — Clear pulse which generates necessary level to clear all flip-flops in converter i.
15. CLRGAS — Pulse that clears all GA flip-flops.
16. CNV — Control flip-flop which equals one state when converter i is busy.
17. πCNV — Will be equal to one if all the in-out converters are in use.

    Boolean expression $(CNV^1)\ (CNV^2)\ (.....)\ (CNV^n)$
18. $\sum$ CNV — Will be equal to 1 if any of the in-out converters are in use.

19. CSL — Converter select level; i determines which converter will be used to execute the in-out order being processed.

20. CTP — A converter i control level, when equal to one the TP flip-flop will normally be set to one.

21. CVB — A level which is equal to one when converter i is busy.

22. DAR — 6 bit address register in converter i, corresponds to j portion of in-out instruction, bits 16-21.

23. DVA — Device Alarm flip-flop in converter i.

24. DVB — Control level which is high whenever the x register holds the address of an In-Out device already in use.

25. DVH — Device Halt flip-flop.

26. END — A control level originating in converter i when the in-out order is complete.

27. GA — A flip-flop which equals one whenever converter i has access to IN-OUT Memory Cycle.

28. $\sum$GA — Equals one when any GA is equal to one.

29. GS — Control flip-flop when a one indicates that converter i may be released to the converter selector circuit.

30. HT — A control level equal one whenever IR holds the code for the HLT instruction.

31. ID — Flip-flop that is set to one in converter i when device selected is an input-type and not magnetic tape.

32. IMO — Improper order Alarm flip-flop in converter i.

33. IO — A control level equal to one whenever IR holds the operation code for an in-out instruction.

34. IOA — In-Out Alarm flip-flop in converter i.

35. IPE — In-Out Parity Error flip-flop in converter i.

36. IR — 6 bit Instruction Register.

37. IS — Interpret Sign flip-flop in converter i.

38. ISE — Sign Error flip-flop in converter i.

39. ISN — Interpret Sign Flip-flop.

40. ISR — 6 bit Instruction Register in converter i.

41. IWR — Tape Erase flip-flop in converter i.

42. LC — A control level equal one whenever CIR holds a legitimate character for ROK order.

43. MA — Memory Address Register of memory No. i.

44. MO — In-out Register of memory No. i.

45. MOR — Is equal to one when either MO or REG is equal to one.

46. MT — Control level equal to one whenever the device in DAR specifies a magnetic tape unit.

47. NHP — Ignore halt on parity error flip-flop in central machine.

48. NH — Ignore Halt on In-Out Alarm flip-flop in converter i.

49. NHC — Non Halt on converter error in central processor.

50. OD — Flip-flop equal to one when the DAR specifies an output type device.

51. RAN — The ISR decoder output which equals a one when the converter i op. code is Rood Alphanumeric.

52. RE — Read pulse which reads into MOR what is in the address found in MA.

53. REG — Any addressable register.

54. RH — Control flip-flop which turns tape on (1) or off (0).

55. ROK — ISR decoder output which equals one when converter i contains "Read Octal".

56. RR — Control level equal to one when ISR contains the op. code for either RRV or RAN.

57. RRV — ISR decoder output which equals one when converter i contains "Read Reverse".

58. RW — Read Write Control in converter i. RW^i = 0 for reading operations and 1 for write operations.

59. RWD — ISR decoder output which equals one when converter i contains "rewind" instruction.

60. SKP — ISR decoder output which equals one when converter i contains "Skip" instruction.

61. SGA — Pulse required to set selected GA flip-flop for the converter In-Out memory cycle.

62. SLB* — A shift pulse that shifts the BFR left three places.

63. TAR — 8 bit buffer character register connecting the In-Out and the BXR register.

64. TC(1-4) — In-Out Memory cycle timing function.

65. TP — Stop flip-flop.

66. TRE — Flip-flop set to one when there is a timing read error in converter or Device Switching Unit.

67. TPE — Tape erase flip-flop in central processor.

68. WAN — ISR decoder output equal to one when converter i contains the "Write Alphanumeric" instruction.

69. WB — Word or Block control flip-flop in converter i WB^i = 0 words are specified, 1 and blocks are specified.

70. WBC — Word Block counter in converter i.

71. WBI — A level which is equal to one when a transfer into WBC is about to take place.

72. WOK — ISR decoder output equal to one whenever converter i contains "Write Octal" instruction.

73. WR — Write Pulse which writes what is in MOR into the address specified by MA. This pulse must have been preceded by RE.

74. WW — A control level equals one when instruction in converter i is either WAN or WWA.

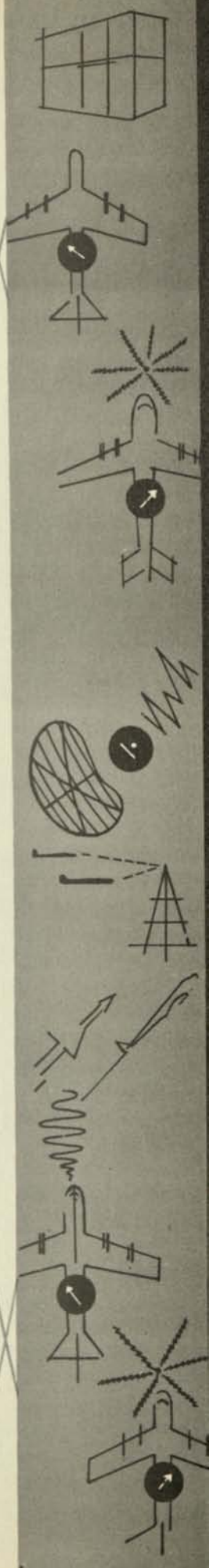
75. WWA — ISR decoder output which equal to one when converter i contains a "Re-write Alphanumeric" instruction.

76. ϕ — Flip-flop which must be one to enable an In-Out memory cycle. Starts the counter.

77. λ — A control flip-flop which interrupts the normal basic cycle and holds the processor in one timing function.

78. θ^{i}_{n} — A flip-flop which controls the variable timing of the converter cycle.

79. CLRCIS* — The pulse line that clears the converter instruction register when a one.



SYLVANIA ELECTRONIC SYSTEMS

A DIVISION OF SYLVANIA ELECTRIC PRODUCTS INC.

63 SECOND AVENUE, WALTHAM, MASS.

A59-30N

MOBIDIC

A

INTERNAL INSTRUCTION MICRO FLOW CHARTS

SYLVANIA ELECTRONIC SYSTEMS
A DIVISION OF SYLVANIA ELECTRIC PRODUCTS INC.

DATA SYSTEMS OPERATIONS
189 B STREET, NEEDHAM 94, MASS.

MOBIDIC A

INTERNAL INSTRUCTION

MICRO FLOW CHARTS

Signal Corps
Technical Requirements
SCL 1866

Contract No. DA-36-039-SC-78111

Submitted to:

U.S. ARMY
Signal Research and Development Laboratories
Fort Monmouth, New Jersey

1 July 1959

SYLVANIA ELECTRONIC SYSTEMS
A Division of Sylvania Electric Products Inc.
DATA SYSTEMS OPERATIONS
189 B STREET - NEEDHAM 94, MASS.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

OPERATION CODE ABBREVIATIONS AND INDEX TO FLOW CHARTS

OPERATION CODE ABBREVIATIONS AND INDEX TO FLOW CHARTS (CONT'D)

PREFACE

This document has been prepared under the assumption that the user is familiar with the MOBIDIC preliminary programming manual.

The Micro Flow Charts are designed to answer reasonable questions which a programmer might ask about unusual or subtle machine functions; e.g., the difference between adding +5 to -5 and -5 to +5. In a few cases, slight liberties have been taken with the actual mechanization in order to make the Micro Flow Charts more meaningful to the programmer.

The written material preceding the flow charts has been organized in three categories. Sections I and II contain basic information which is needed in order to read and understand the flow charts. Sections III, IV, and V contain information which is useful to the programmer but which cannot conveniently appear in the flow charts. Section VI presents detailed information on machine timing. The reader may therefore select material on his own level of interest.

Sylvania wishes to acknowledge the contribution of Miss D. Davidson in the preparation of these micro flow charts.

I. THE BASIC CYCLE

The time taken to execute one complete instruction is called a machine cycle. The time required for one machine cycle is normally 16 microseconds. During the execution of a machine cycle, there are actually two operations being performed simultaneously. They can be divided into what are called basic cycle operations and special operations. The basic cycle operations are those which will be executed during every machine cycle for all instructions; the special operations are those which will be executed only for the particular order being interpreted.

Each machine cycle is subdivided into 8 periods, called Timing Functions (TF's) each of which is normally two microseconds long. For easy reference, they are numbered TF1 through TF8. For instructions requiring more than 16 microseconds, such as multiplication and division, any of the 8 timing functions can be extended in increments of 2 microseconds by a concept called "sticking lambda" (λ). This concept is described in detail on page 14.

During the execution of each instruction which is not a control transfer, the address of the next instruction in sequence is determined; that is, the program counter (PC) is stepped to the next memory location.

The beginning of the basic cycle, usually considered to be at TF4, does not coincide with the "beginning" of an instruction execution, which is usually considered to be at TF1. The instruction is picked up at TF4; from TF5 through TF8, the instruction is decoded, and the operand, if one is designated, is read out of memory at TF8. The instruction is then executed from TF1 through TF8. However, on this cycle, a new instruction is again being picked up by the basic cycle at TF4. See Figure 1.

When an instruction is first extracted from memory, the address specified by the instruction (bits 1-15) is stored in the address register (AR) and then transferred to the Memory Address Register. The next twelve bits of the instruction word (bits 16-27) are stored in the X register, and bits 28-30 are stored in the G register. From the X and G registers the information is transferred to an index register or to another Memory Address Register or to an input-output converter, depending on the particular instruction given.

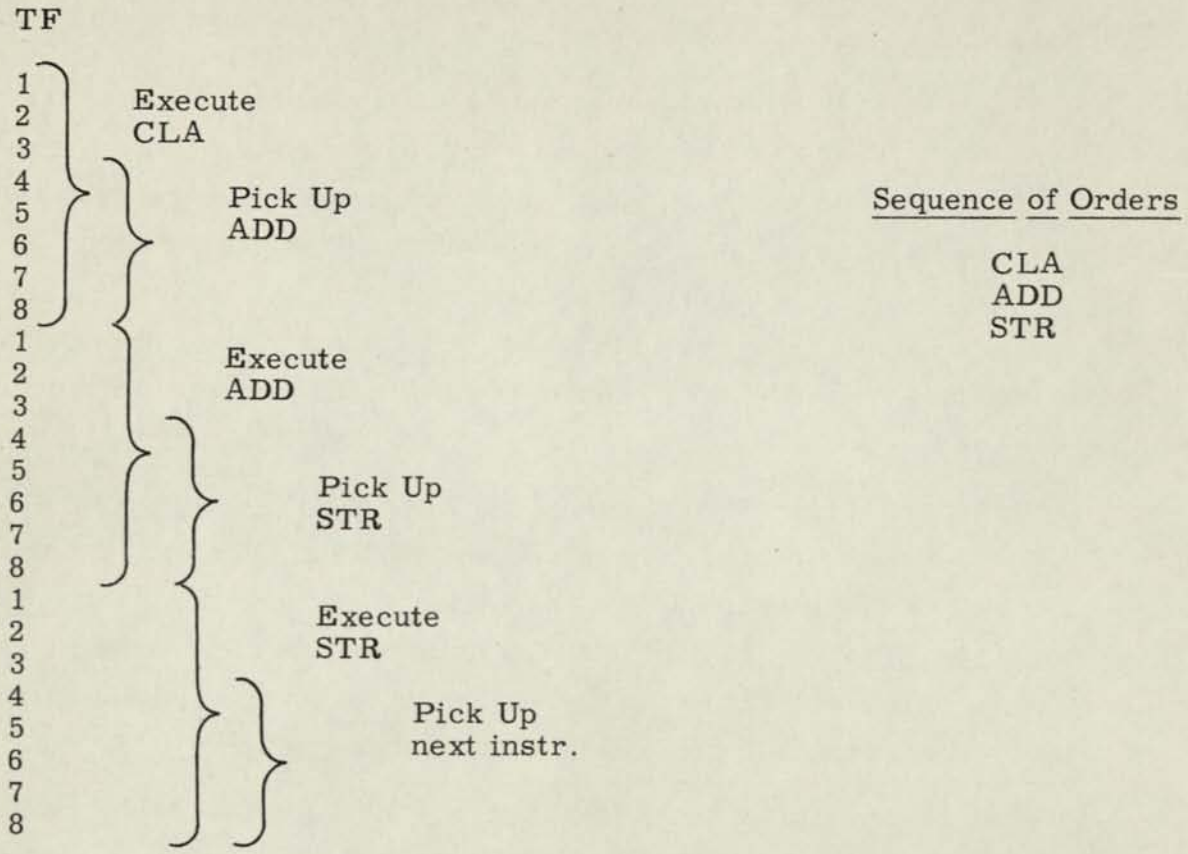


Figure 1. Coincidence of Basic Cycle and Instruction Cycle.

The last six bits (31-36), which specify the operation code, are transferred to the Decoder register to be interpreted. Figure 2 shows the breakdown of an instruction into its various components before interpretation.

When reading the micro flow charts, the reader is cautioned concerning the following fact: As the reader is examining the micro flow chart of a particular instruction, he should also consult the Basic Cycle micro flow chart for functions which are being performed there simultaneously. This is the underlying philosophy on which these flow charts are based; used independently they have no meaning. For example, if Timing Function 3 is being investigated in the ADD instruction, Timing Function 3 of the Basic Cycle must also be examined to determine what operations are occurring at the same time. See Figure 3.

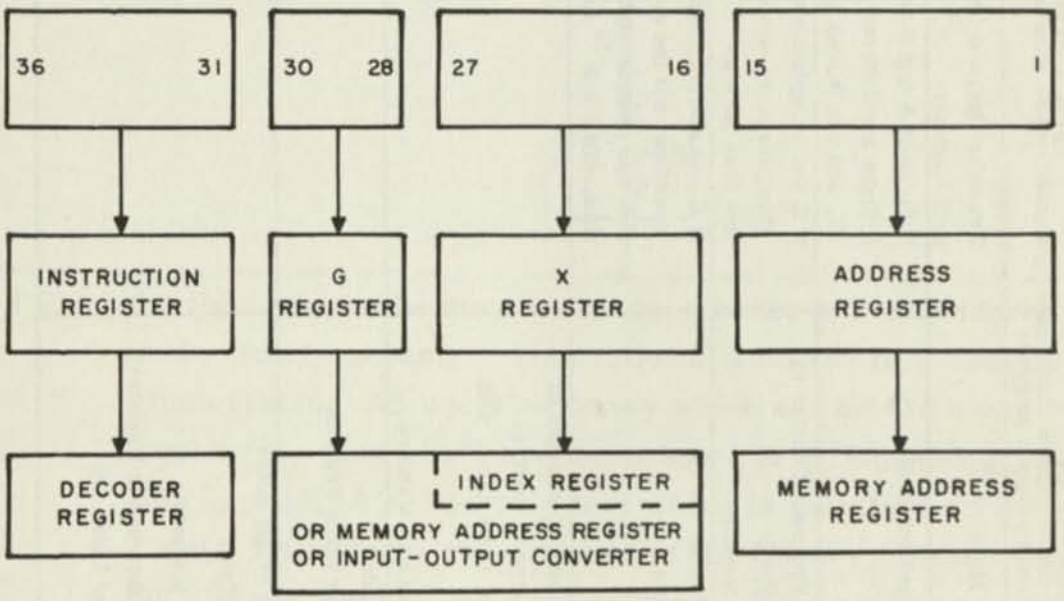


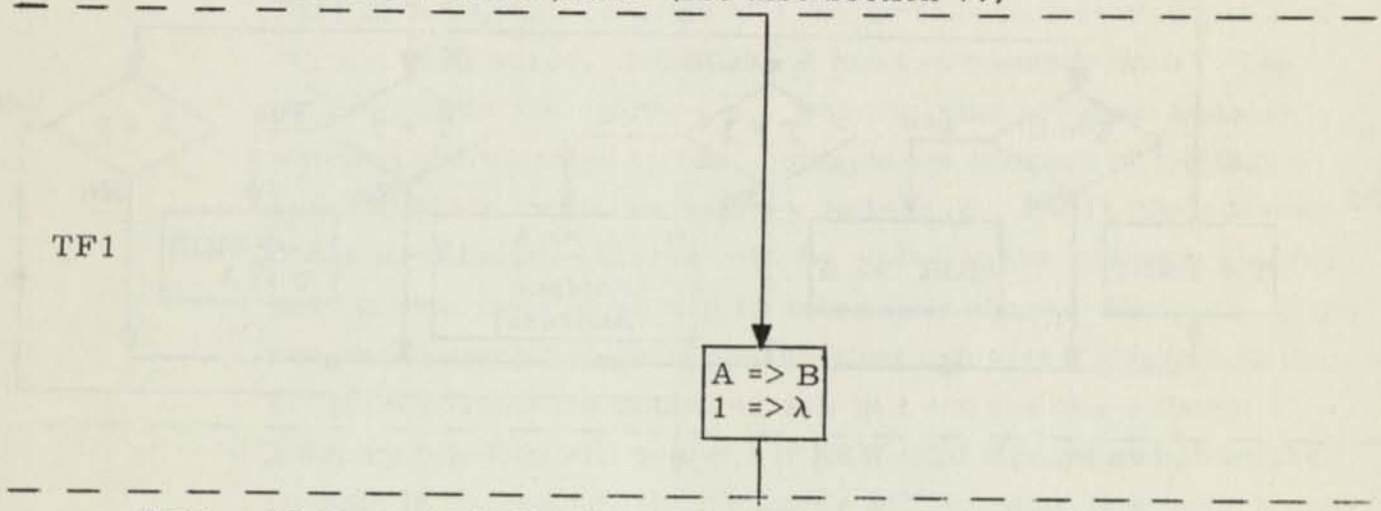
Figure 2. Decomposition of Machine Instruction

Operational / Timing Function	BASIC CYCLE OPERATIONS	SPECIAL OPERATIONS FOR ADD MECHANIZATION
TF-1	1. Operand arrives at Memory Output Register	
TF-2	1. Rewrite Operand into Memory	1. Transfer Operand from Memory Output Register to B Register
TF-3	1. (Used for in-out instructions only)	1. If Signs of A and B are alike, add B to A; If Signs are different, form one's complement of B, and add B plus 1×2^{-36} to A
TF-4	1. Transfer Contents of Program Counter to Memory Address Register 2. Send a Read Pulse to Memory	1. If B was complemented and there was no overflow, form the one's complement of A and reverse its sign
TF-5	1. Next Instruction arrives at Memory Output Register	1. If B was not complemented and there was an overflow, set the overflow alarm flip-flop 2. If B was complemented and there was no overflow, add 1×2^{-36} to A
TF-6	1. Transfer appropriate bits of contents of Memory Output Register to Address Register, X Register, G Register, and Instruction Register 2. Rewrite the Instruction into Memory	
TF-7	1. Step the Program Counter 2. Add the Contents of the Index Register specified by γ to AR	
TF-8	1. Transfer the Contents of AR to the Memory Address Register 2. Send a Read Pulse to Memory 3. Transfer contents of Instruction Register to Decoder - Decoder Lines are energized	

Figure 3. Basic Cycle in Coincidence with Add Instruction

II. EXPLANATORY EXAMPLE - REPLACE ADDRESS (RPA)

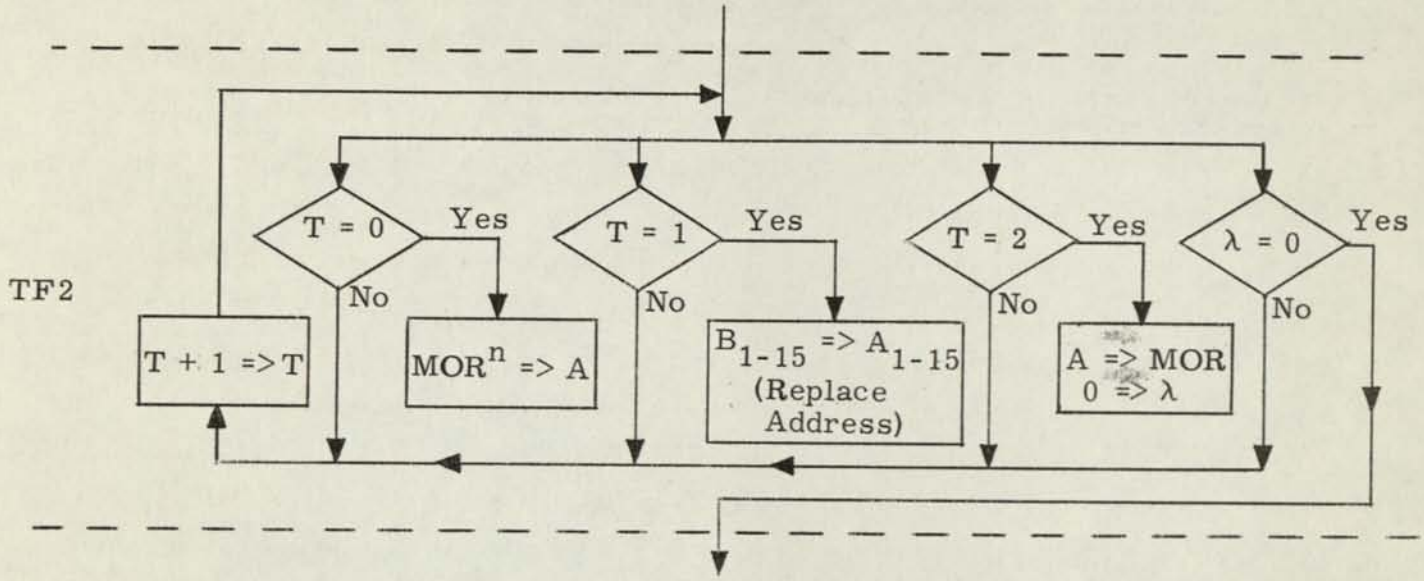
TF1 - The contents of the Accumulator are transferred to the B-Register. This destroys the previous contents of the B-Register. The λ flip-flop is set. This causes a delay in TF2; that is, TF2 will take more than 2 μsec. (See also Section V.)



TF2 - All the tests in the diamond-shaped boxes are considered to be made simultaneously. This follows from the flow chart convention which states: All decision boxes which are on the same horizontal level on the flow chart are considered to be happening in parallel. (See Appendix A.) The only test which is satisfied at this point is T = 0. Therefore, the contents of the Memory Output Register (MOR^n) are sent to the Accumulator. This could be the contents of a core memory location, or of a register, whichever is denoted by **a** of the instruction. The T-counter is then increased and the tests are made again. This time T = 1; thus the low-order 15 bits of the B-Register (actually the contents of **a**) are transferred to the low-order 15 bits of the Accumulator. This accomplishes the "Replace Address".

The next time around the loop, T = 2 is satisfied and the contents of the Accumulator are sent to MOR^n. This information is then available to be written back into memory. (See Basic Cycle TF2). λ is reset to zero.

On the next pass through the loop, $\lambda = 0$ is satisfied and the flow continues to TF3.



TF3 - No Action

TF4 - No Action

TF5 - The contents of the B-Register are sent back to the Accumulator.

TF6 - No Action

TF7 - No Action

TF8 - No Action

III. INSTRUCTION ERROR CONTROLS

a. Non Existent Memory Address

MOBIDIC has a maximum capacity of seven memory units. Each memory unit consists of a magnetic core matrix capable of storing 4096 words. MOBIDIC A has two memory units. The memory units are cyclic; i.e., whenever the program counter, which is incremented by one, contains the address of the last location of the available memory units (e.g., 8191), the address of the first location (zero), will be placed in the program counter and the next instruction will be taken from there. However, if the program attempts to put a non-existent memory address into the program counter the computer will halt and the non-existent memory flip-flop will be set. If KEW (Kill Error) switch on the console is depressed, the computer will not stop; however, a non-existent memory address which is in the Program Counter will now be incremented resulting in another non-existent memory address. The machine will continue to run but of course no orders will be executed.

Consideration must be given to an instruction that is indexed which results in a non-existent memory address. The following three cases are considered, assuming a MOBIDIC with two memory units (8192 words) and with Index Register 1 containing 1001_8.

The following rule should be followed in determining what happens to a non-existent memory address: If the address of an instruction is indexed so as to change the three high order bits of AR, (AR_{13-15}), this effective address will be interpreted modulo 2^{12}. If AR_{13-15} are not changed, the address is treated as a non-existent register address. (See next page).

1) If the address of an instruction to be indexed is a valid machine location such as CLA 17235_8, 1, the resulting effective address will be 20236_8, a non-existent memory address. However this address will be reduced modulo the size of the memory unit, i.e., $20236_8 - 10000_8 = 10236_8 - 10000_8 = 00236_8$. Therefore the instruction that will be executed will be CLA 00236_8.

2) If the address of an instruction to be indexed is a non-existent memory address such as CLA 27235_8, 1, the resulting effective address will be 30236_8, a non-existent memory address. However this address will also be reduced modulo the size of the memory unit, i.e., $30236_8 - 10000_8 = 20236_8 - 10000_8 = 10236_8 - 10000_8 = 00236_8$. Therefore the instruction that will be executed will be CLA 00236_8.

3) If the address of an instruction to be indexed is a non-existent memory address such as CLA 20235, 1, the resulting effective address will be 21236_8, a non-existent memory address. This address will not be modified and will be treated as a non-existent register address. See the next section.

b. Non-Existent Register Address

There are two conditions that must be considered when a program refers to a non-existent register. If the instruction that is referring to the non-existent register is a data transfer type, i.e., STORE, LOAD, MOVE, the actual transfer of data is not accomplished. The data is not destroyed in its original location, but since there is no register to receive the data, no data transfer is done (effectively a no-op).

If, however, a non-existent register is called for, i.e., CLA 70000, the contents of that address is considered to be zero. That is, if the alpha portion of an instruction which requires a memory access (ADD, LOD, SBB) is a non-existent register address, then the Memory Output Register (MOR^n) would contain zero.

c. Non-Existent Instruction

When the binary configuration for a non-existent instruction is transferred to the Decoder register, the machine will halt unless the KEW switch on the console is depressed. If the KEW switch is depressed, the instruction is ignored (effectively a no-op).

d. Non-Existent Flip-Flop

If a non-existent flip-flop is specified in a sense instruction, the computer will operate as though the flip-flop were in the "zero" state. The three possible cases are as follows:

1) Sense	SEN α_1, γ_1, β_1	The next instruction is taken in sequence
2) Sense & Set	SNS α_2, γ_2, β_2	The next instruction is taken from α_2
3) Sense & Reset	SNR α_3, γ_3, β_3	The next instruction is taken in sequence

e. Non-Existent Index Register

1) TRX. If a non-existent index register is specified by the γ bits of the TRX instruction, the index registers will not be altered and control is transferred to the location specified by α of the TRX instruction.

2) LDX. If a non-existent index register is specified by the γ bits of the LDX instruction, the index registers will not be altered and the instruction may be considered a no-op.

3) ADB, SBB. If a non-existent index register is specified by the γ bits of either ADB or SBB, the result, A_{1-12}, of the addition or subtraction is not placed in any of the index registers.

4) TRL. If the γ bits of the TRL instruction specify a non-existent index register the order is executed as before, except that β is not placed in any of the index registers.

IV. TRAPPING MODE CONTROL

When the computer is operating in the trapping mode, a transfer to memory location zero will occur for every transfer and sense order in the program except TRC. The trapping mode flip-flop (TRA) is not addressable. It is controlled only through use of the two low order bits of β (β_{16} and β_{17}) of the TRU instruction. When an instruction is "trapped", control is transferred to location zero, the contents of the program counter are sent to the B register, and the trapping mode flip-flop (TRA) is reset.

The TRU instruction can be made immune to the trapping mode flip-flop through use of the same β bits. For additional information on trapping mode, reference is made to the MOBIDIC Programming Manual.

It will be noted from the Basic Cycle that trapping mode is implemented in TF7 by sending a TRU to the Instruction Register whenever a "trappable" order is detected.

V. REPEAT MECHANISM

If a REPEAT (RPT) instruction is given, the instruction immediately following the RPT is repeated the number of times specified by the α of the RPT. The alpha portion of the instruction being repeated is incremented on each execution (after the first) by the β portion of the RPT order. For example:

```
RPT  10, 0, 2
ADD  100
```

The sequence of instructions which would be executed is

```
RPT  10, 0, 2          10 => I^3,  2 => I^4
ADD  100
ADD  102
ADD  104
 .
 .
 .
ADD  120
```

The ADD instruction would be executed 11 times in all. If the RPT had been indexed (RPT 10, γ, 2), then the number of times the ADD would be repeated would be $10 + C(I^{\gamma})$ times. Notice that Index Registers 3 and 4 are always used by the RPT mechanism.

If the instruction being repeated were a MOV order, the contents of Index Register 2 would be added to the $\gamma\beta$ portion of the MOV order on each execution after the first.

See the MOBIDIC Preliminary Programming Manual for a complete discussion of RPT - TRC.

The operation of the repeat control in the Basic Cycle is illustrated by the flow chart on page 104. Note that on TF6 if the repeat is in operation, the new instruction is not initiated; also, this is the point at which Index Register 3 is reduced by one. This Index Register tells the computer whether or not the repeat condition is satisfied. If I^3 were zero at the next test in TF2, the RPT control would be discontinued.

If the repeat is not in operation normal indexing of the instruction occurs. The program counter is also incremented. If the repeat is in operation normal indexing does not occur; instead the contents of I^4 are added to the contents of the Address Register (AR) (the address portion of the instruction being repeated).

VI. DETAILED TIMING INFORMATION

Every machine cycle is subdivided into eight periods or Timing Functions. Each timing function is normally 2 microseconds long and is split into a p-pulse and a t-pulse each 1 microsecond apart. These p and t-pulses have not been shown on the accompanying flow charts; however, an explanation of these pulses is needed to fully understand the machine mechanization. The functions connected with the p-pulse are executed first, followed by the operations connected with the t-pulse. Every p-pulse must be followed by a t-pulse.

a. Description of Lambda (λ)

Lambda is a flip-flop, not under program control, which is set or reset depending upon whether certain conditions of the operation are met. The use of lambda, as described below, can extend a timing function longer than the normal two microseconds.

When lambda is "stuck" (set to 1; i.e. $1 \Rightarrow \lambda$) on a t-pulse, the following timing function is repeated until λ is "unstuck" (reset λ; i.e., $0 \Rightarrow \lambda$). When λ is stuck on a p-pulse, the timing function of the p-pulse is repeated until λ is unstuck.

When λ is unstuck on a t-pulse, the timing function of the t-pulse is repeated one more time after which the next timing function is executed. If λ is unstuck on a p-pulse, the following t-pulse is normally executed after which the next timing function is executed. A sample of λ control timing is shown in Figure 4.

b. T Counter

The T-counter is an add-one or subtract-one counter. The T-counter is cleared to zero by the Basic Cycle in TF8. It is used as a shift control counter for all shift operations in the arithmetic unit. It is also used as an auxiliary timer whenever λ is stuck at one of the timing functions. (Many of the detailed logical steps needed for the implementation of certain instructions depend on the state of the T-counter.)

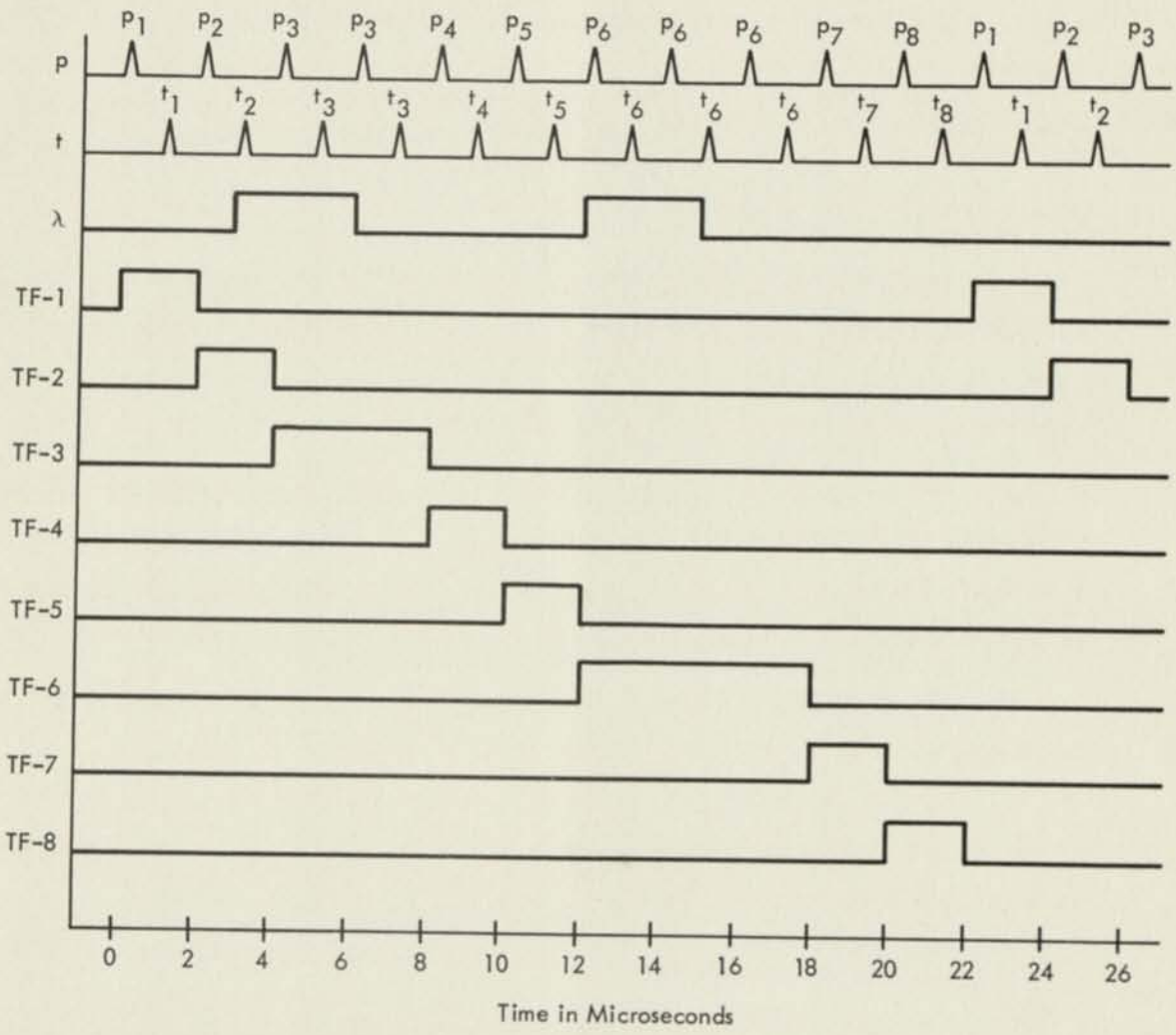
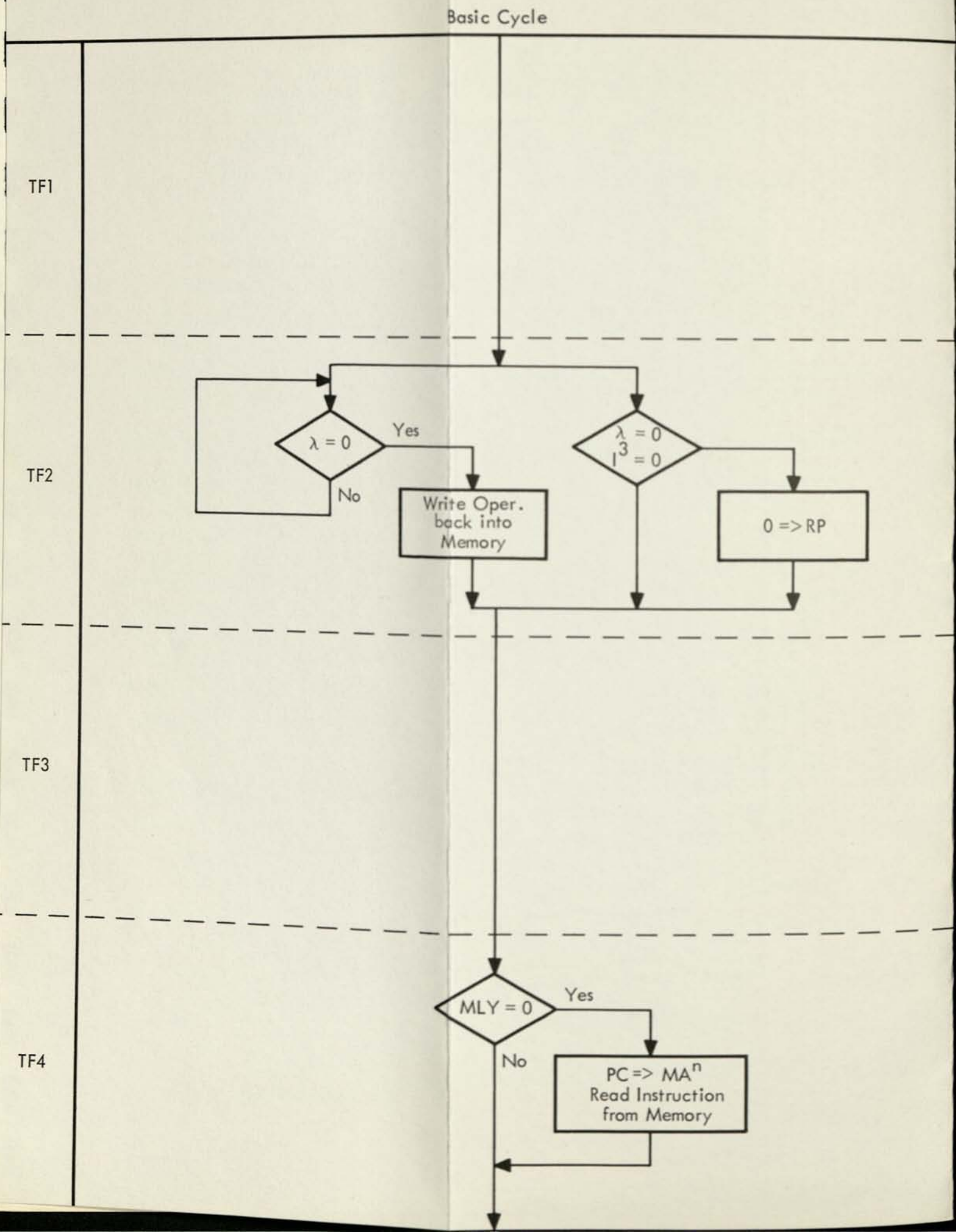


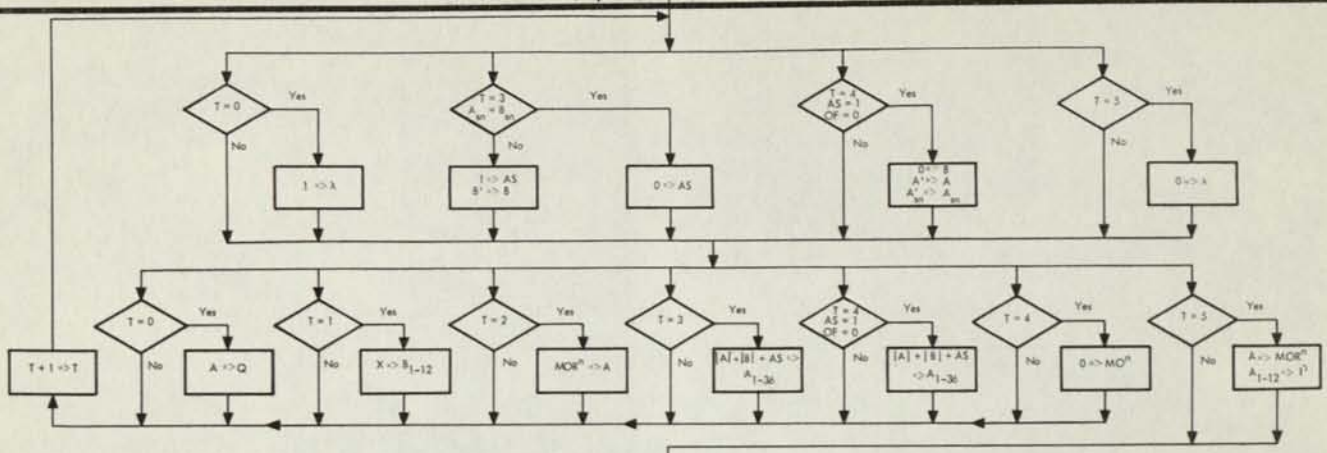
Figure 4. Control Timing with Lambda Flip-Flop


Basic Cycle
TF1
TF2
TF3
TF4
λ = 0
Yes
No
λ = 0
I3 = 0
Write Oper. back into Memory
0 => RP
MLY = 0
Yes
No
PC => MAn
Read Instruction from Memory

MICRO FLOW CHARTS

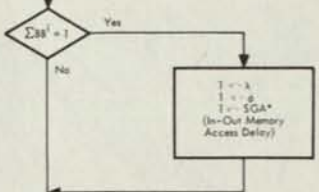
ADB 24 $C(A) \Rightarrow Q,\ \beta + C(\alpha) \Rightarrow A,\ \alpha,\ I^{\gamma}$ 26μs.

TF1

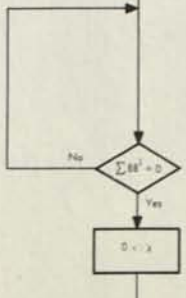


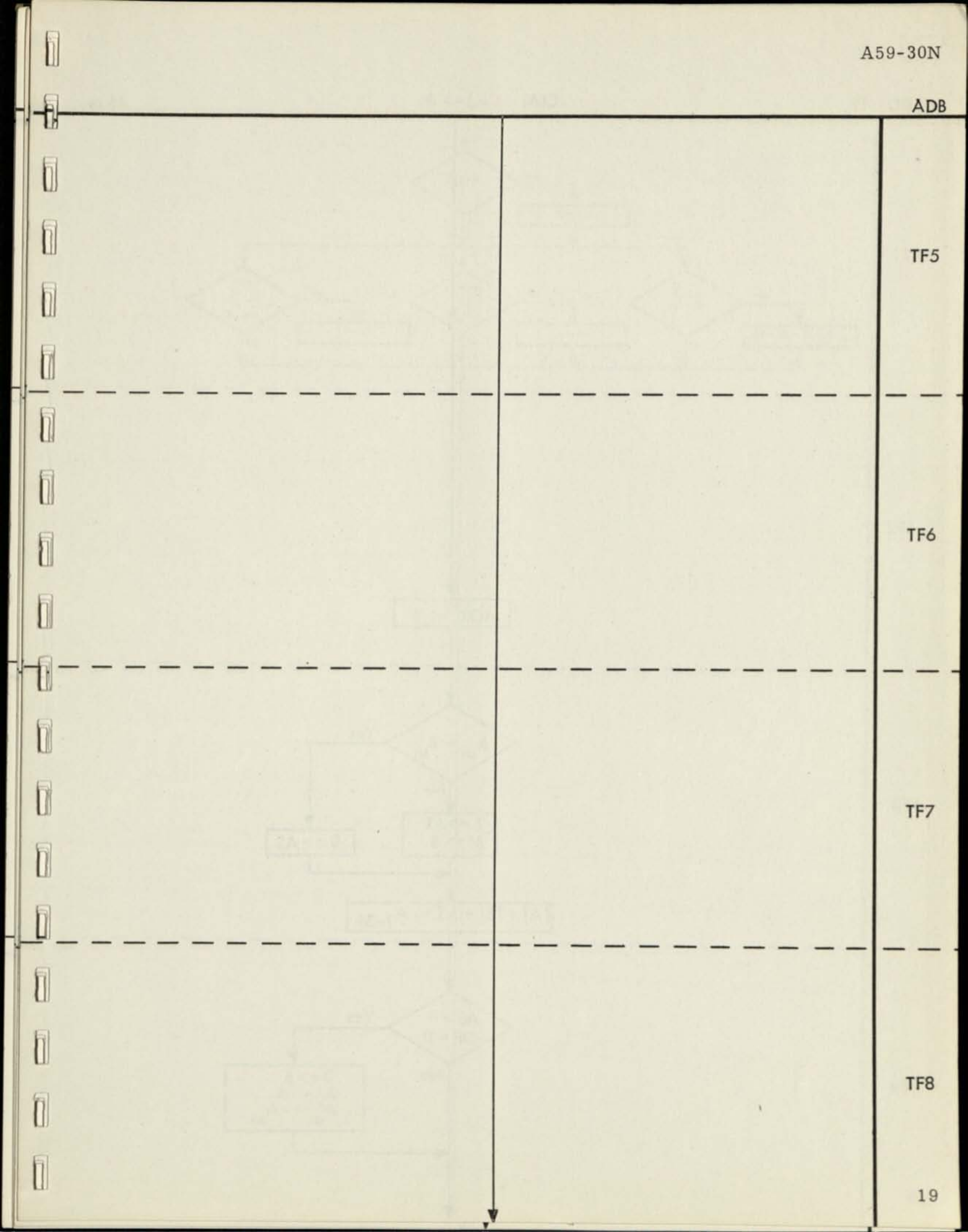
TF2

TF3



TF4

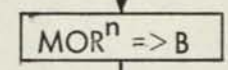



ADB
TF5
TF6
TF7
TF8

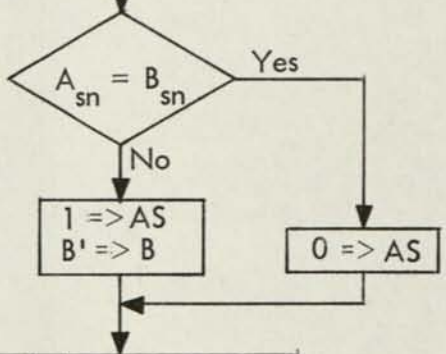
ADD 12 C(A) + C(a) => A 16 μs.

TF1

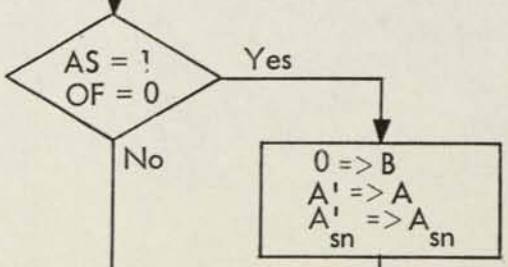
TF2



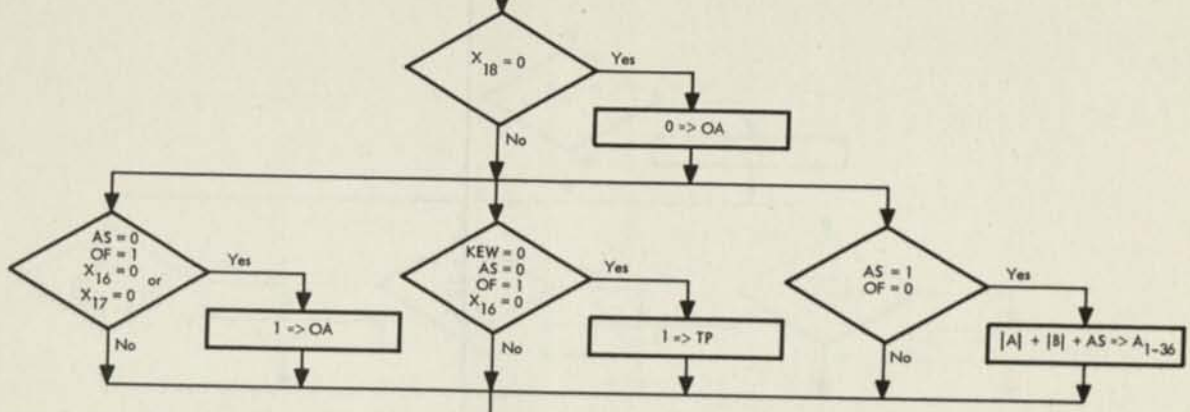
TF3



TF4



ADD



TF5

TF6

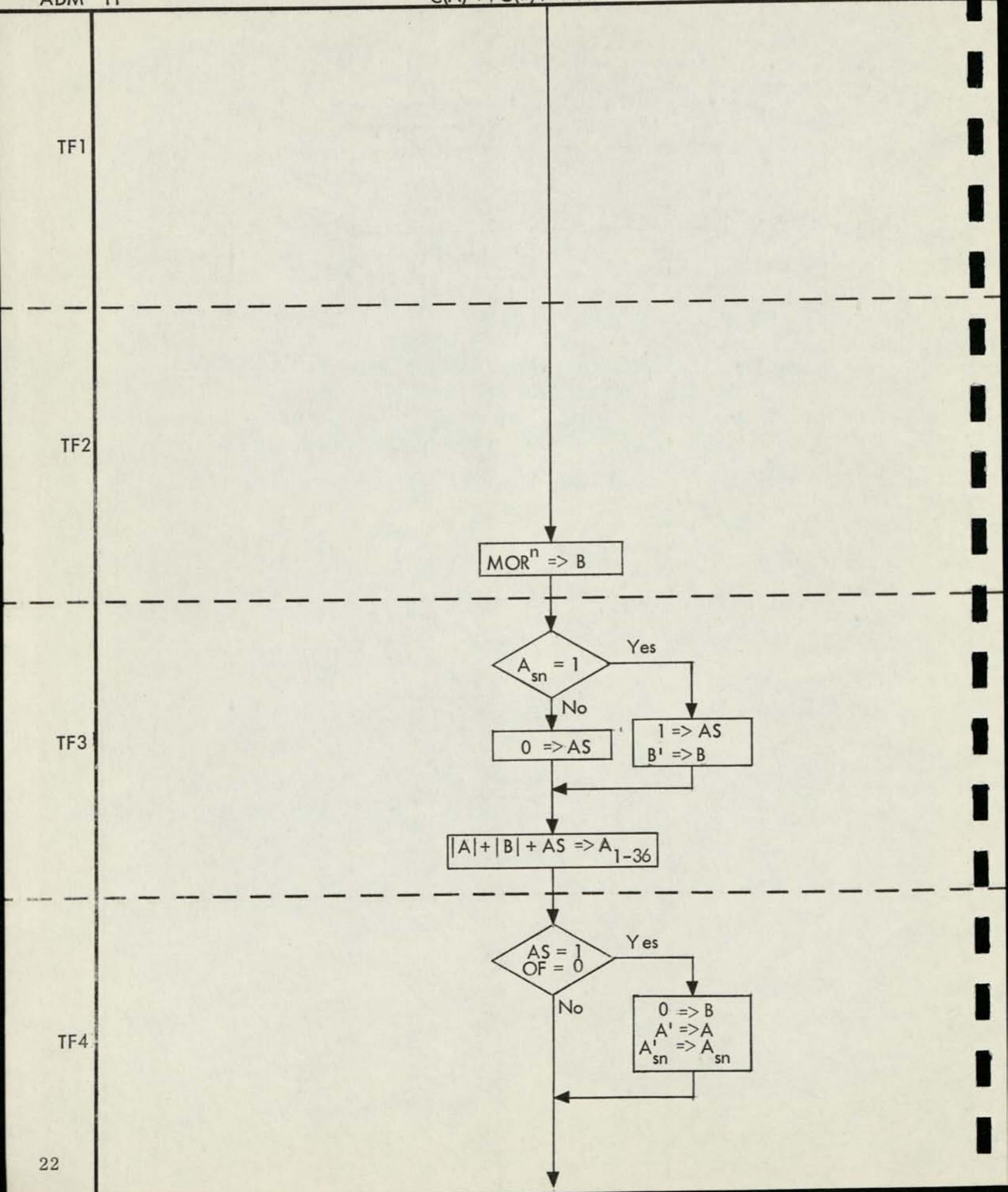
TF7

TF8

ADM 11 — C(A) + | C(a) | => A — 16 μs.

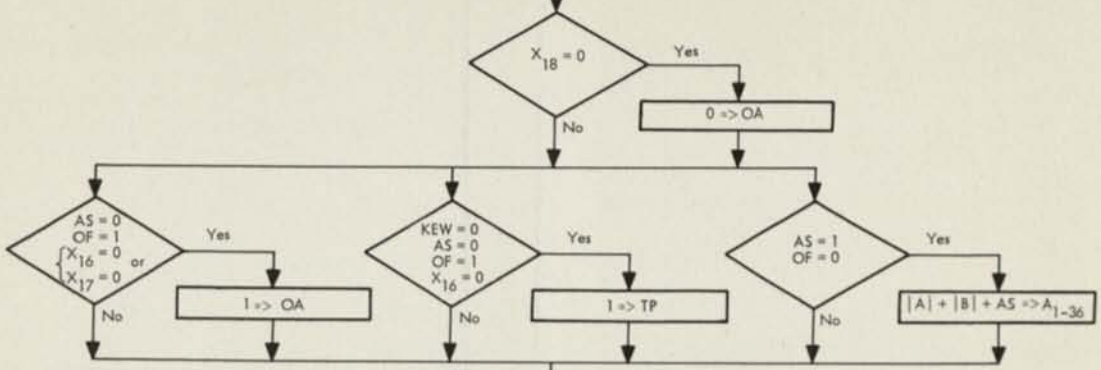
TF1

TF2



TF3

TF4



TF5

TF6

TF7

TF8

CAM 11

$|C(a)| \Rightarrow A$

16 μs.

TF1

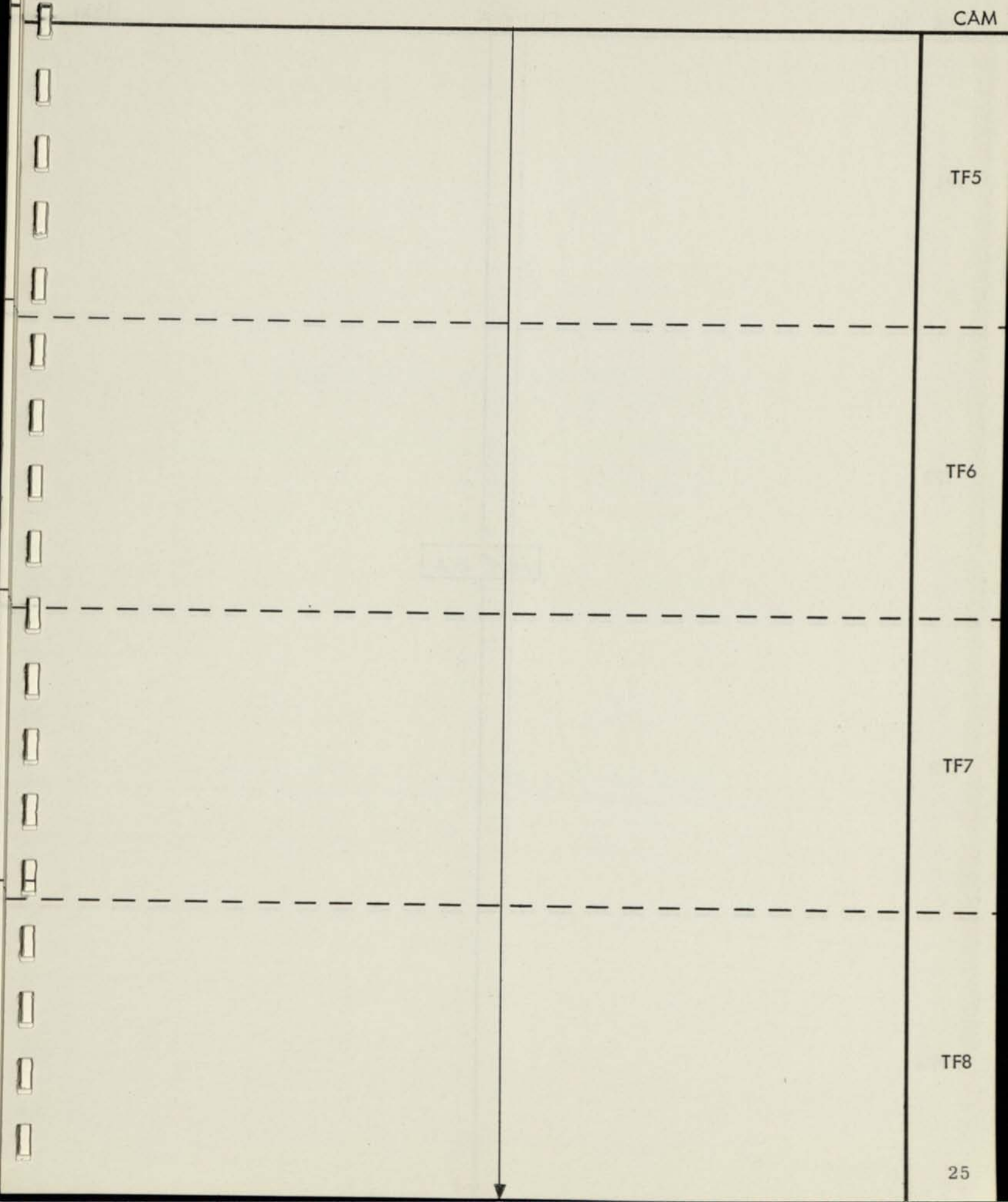
TF2

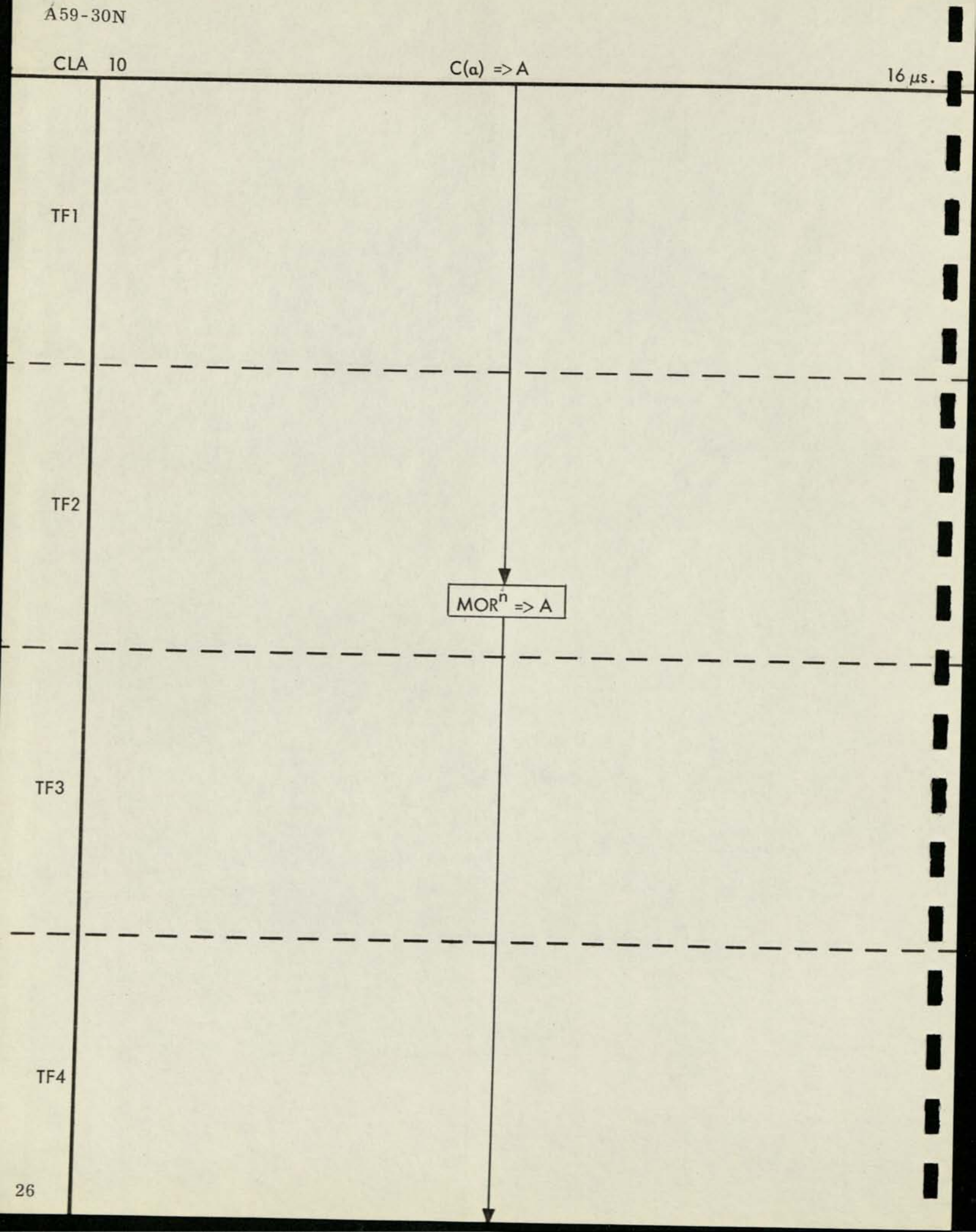
$MOR^n \Rightarrow A$

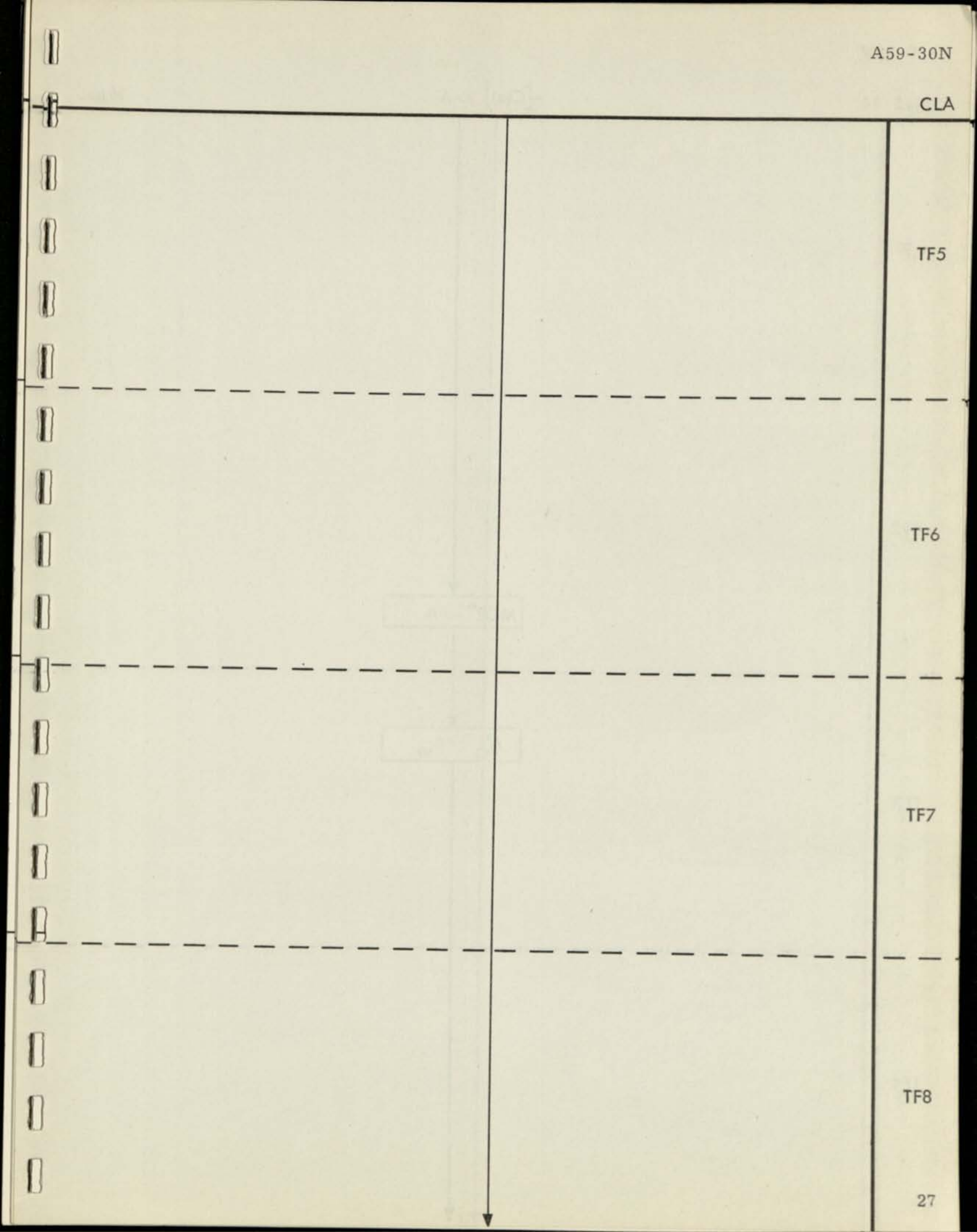
TF3

$0 \Rightarrow A_{sn}$

TF4


CAM
TF5
TF6
TF7
TF8


CLA 10
C(a) => A
16 μs.
TF1
TF2
MOR^n => A
TF3
TF4


CLA
TF5
TF6
TF7
TF8

CLS 14

$-[C(a)] \Rightarrow A$

16 µs.

TF1

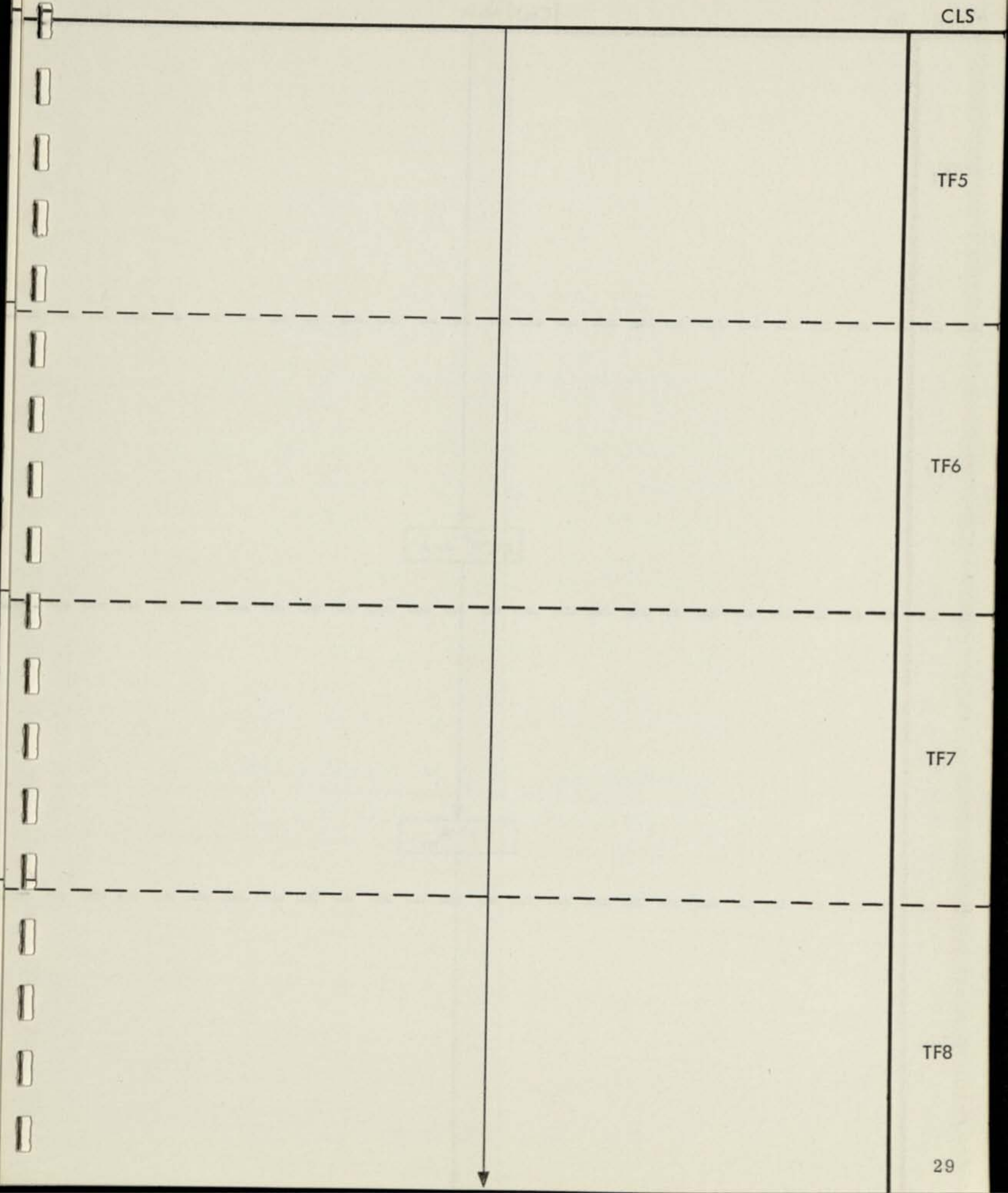
TF2

$MOR^n \Rightarrow A$

$A'_{sn} \Rightarrow A_{sn}$

TF3

TF4


CLS
TF5
TF6
TF7
TF8

CSM 15

$-|C(a)| \Rightarrow A$

16μs.

TF1

TF2

$MOR^n \Rightarrow A$

TF3

$1 \Rightarrow A_{sn}$

TF4

CSM

TF5

TF6

TF7

TF8

CYL 35

C(A) and C(Q) Cycle Left;
$Q_{sn} \Rightarrow A_1$, $A_{sn} \Rightarrow Q_1$

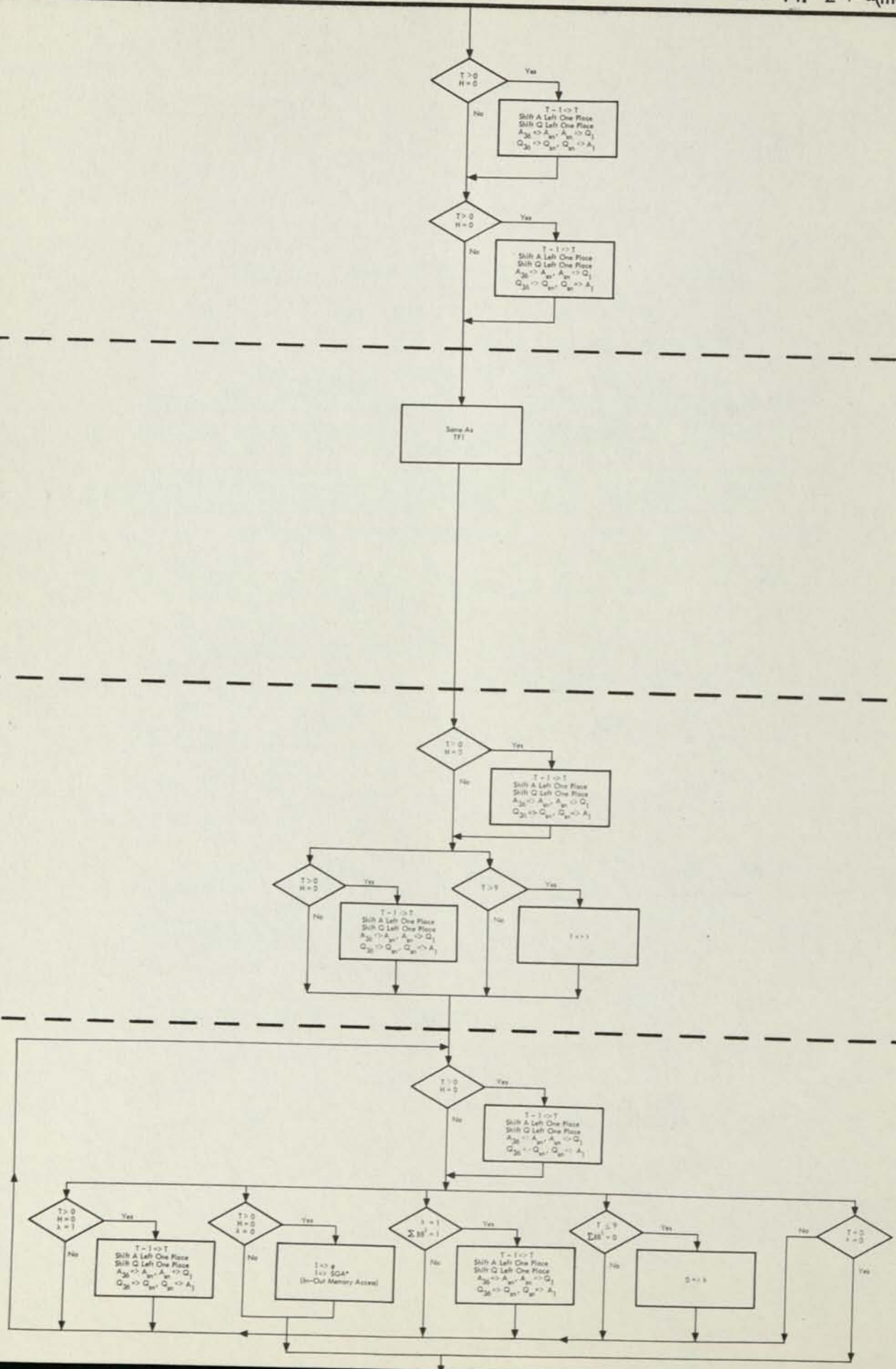
$\alpha \leq 14$: $16\,\mu s$.
$\alpha > 14$: $2 + \alpha \pmod 2 \mu s$.

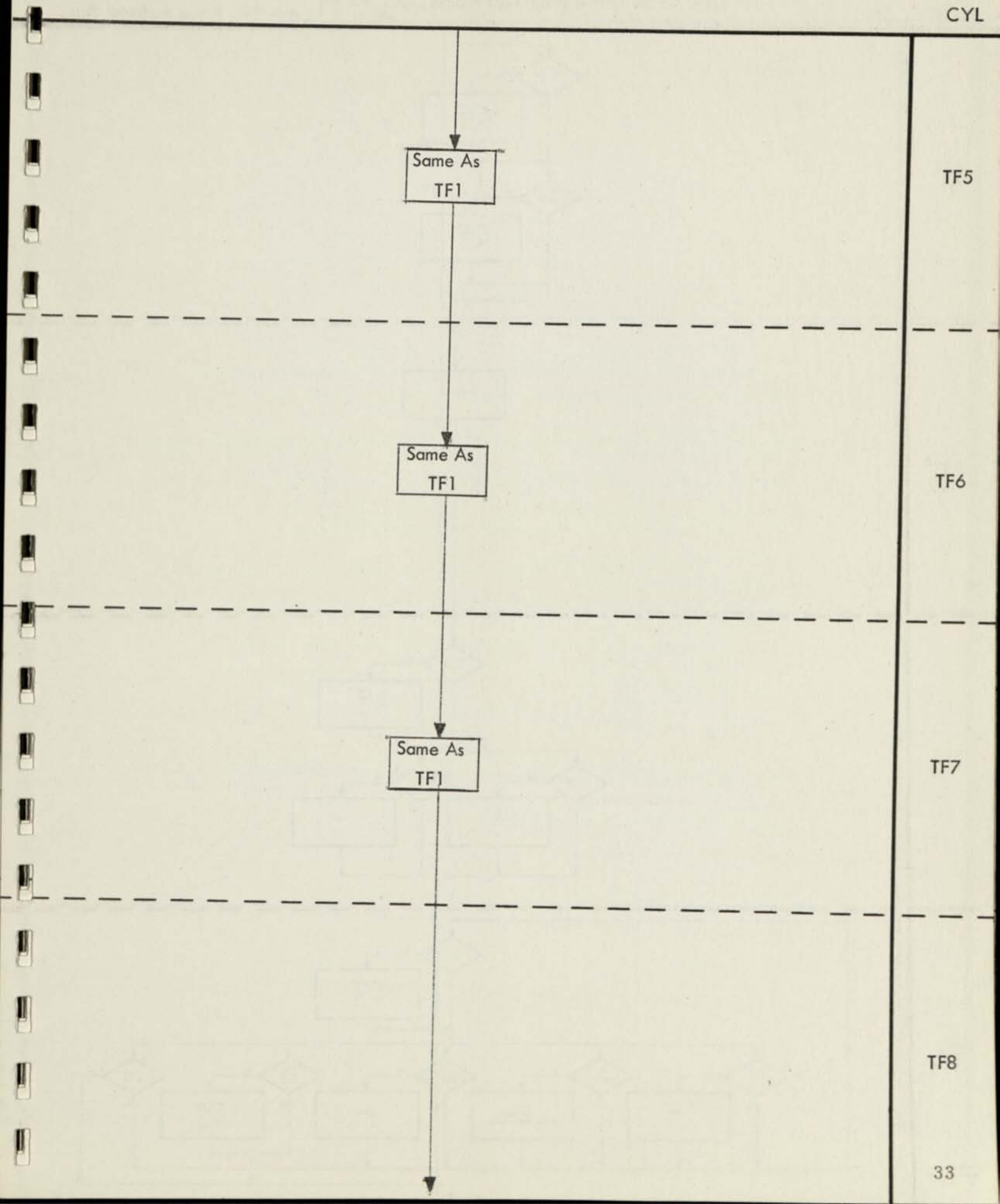
TF1

TF2

TF3

TF4




CYL
Same As TF1
TF5
Same As TF1
TF6
Same As TF1
TF7
TF8

CYS 34

C(A) Cycle Left a (mod 128) Places, $A_{36} \Rightarrow A_1$

$a \leq 14$: $16\mu s$.

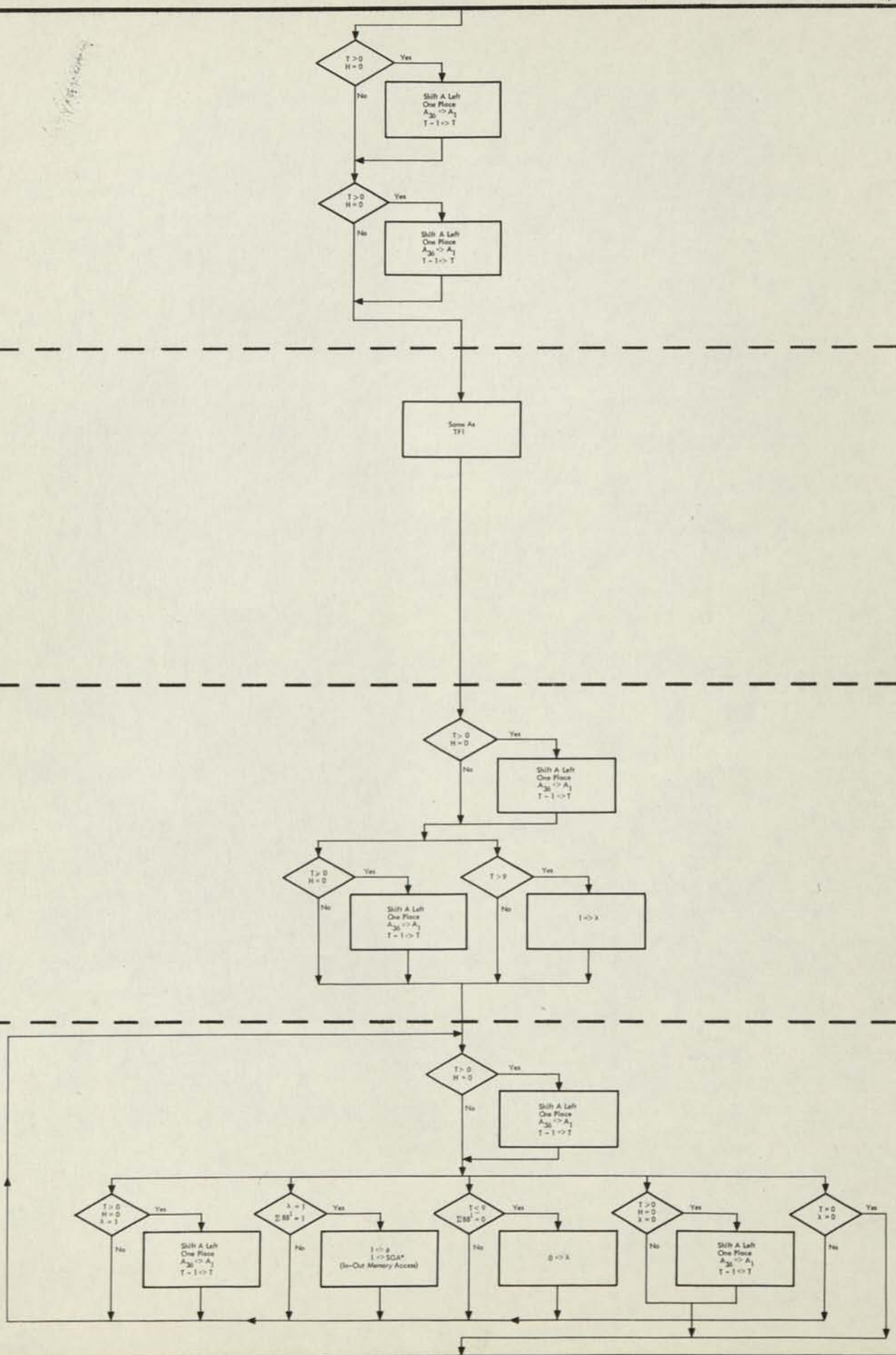
$a > 14$: $2 + a + a \pmod 2 \mu s$.

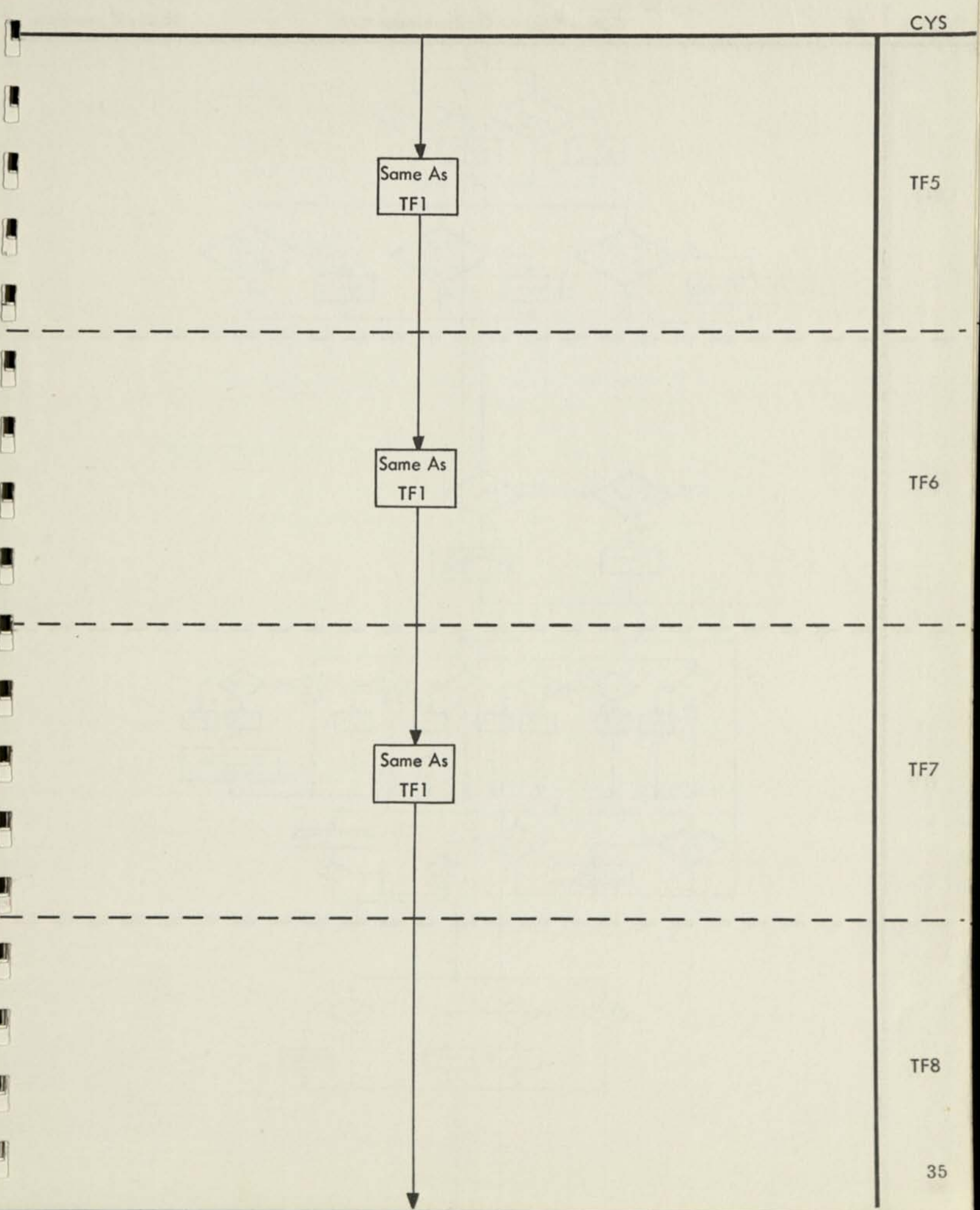
TF1

TF2

TF3

TF4




CYS
Same As TF1
TF5
Same As TF1
TF6
Same As TF1
TF7
TF8

DVD 22

C(A) ÷ C(a) => Q; Remainder =>A

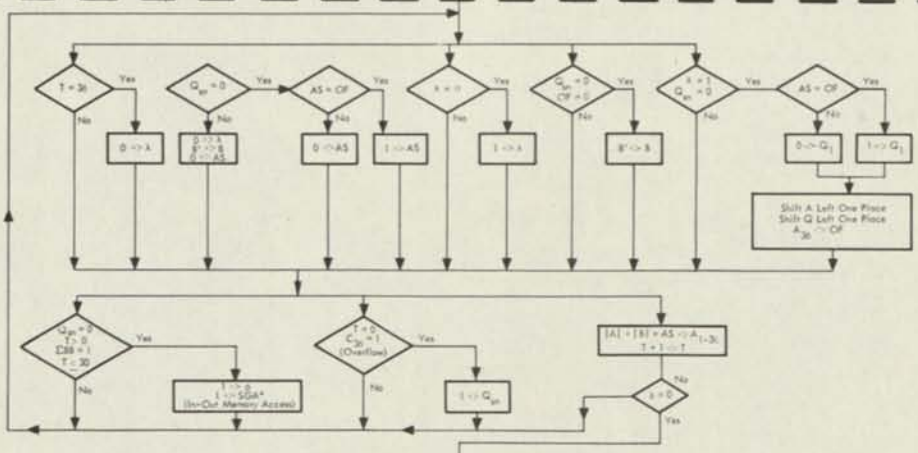
88 μs.

18 μs. if overflow

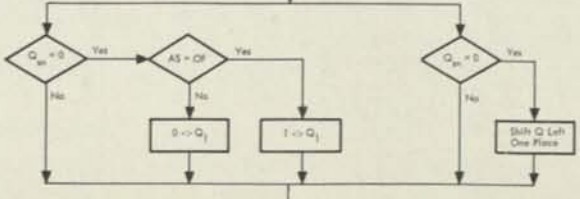
TF1

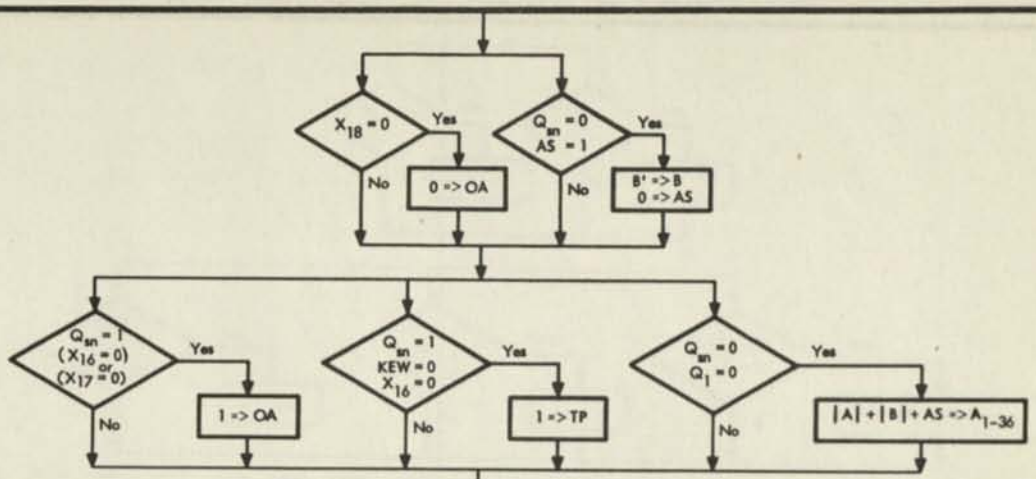
TF2

TF3

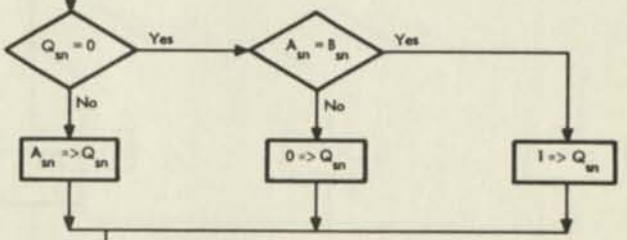


TF4





TF5



TF6

TF7

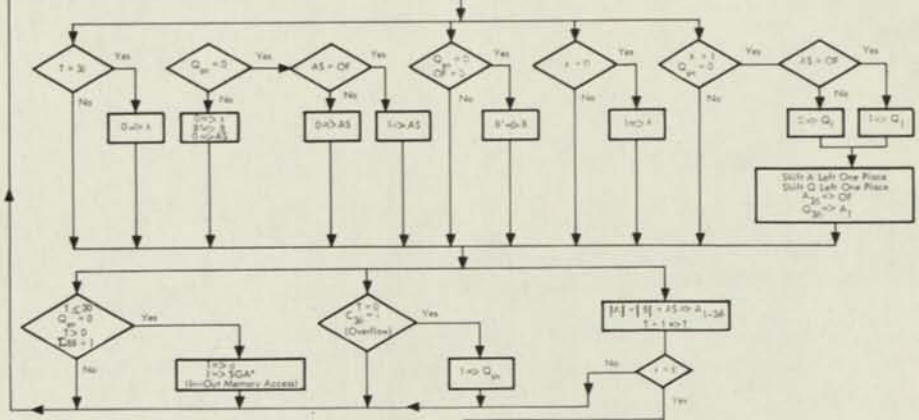
TF8

DVL — $C(A + Q) \div C(a) \Rightarrow Q$; Remainder $\Rightarrow A$ — 88 μs. / 18 μs. if overflow

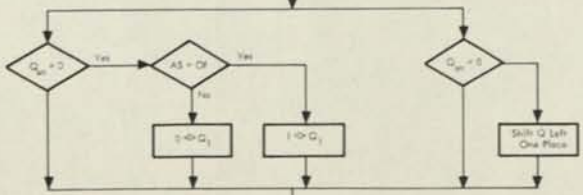
TF1

TF2

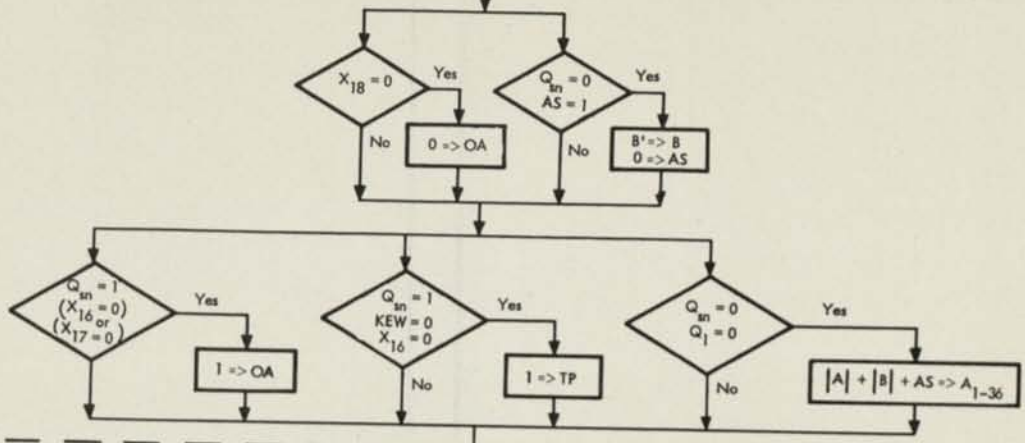
TF3



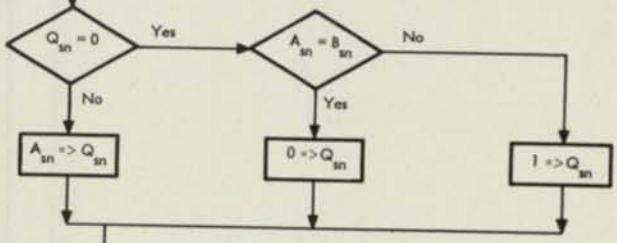
TF4



DVL



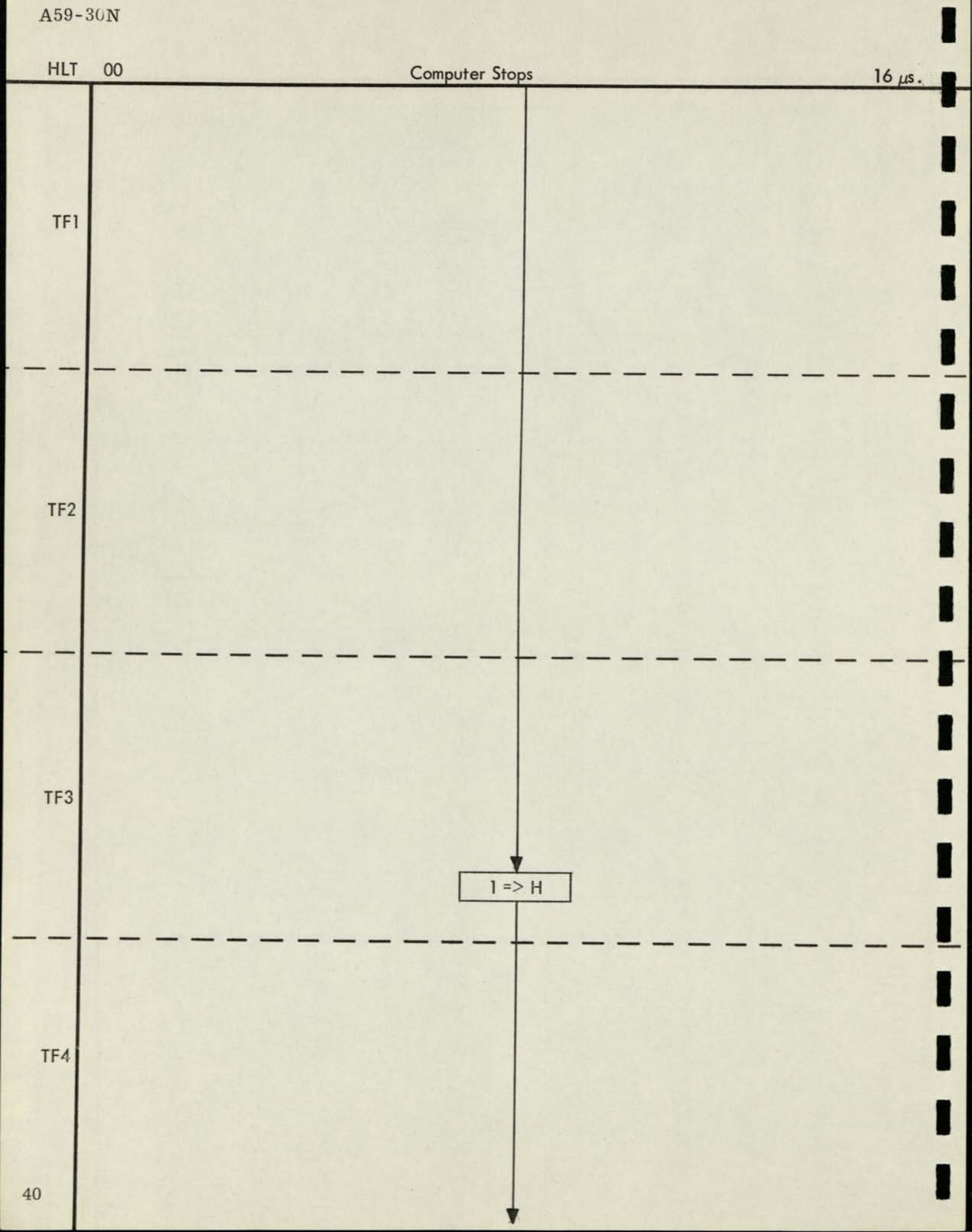
TF5

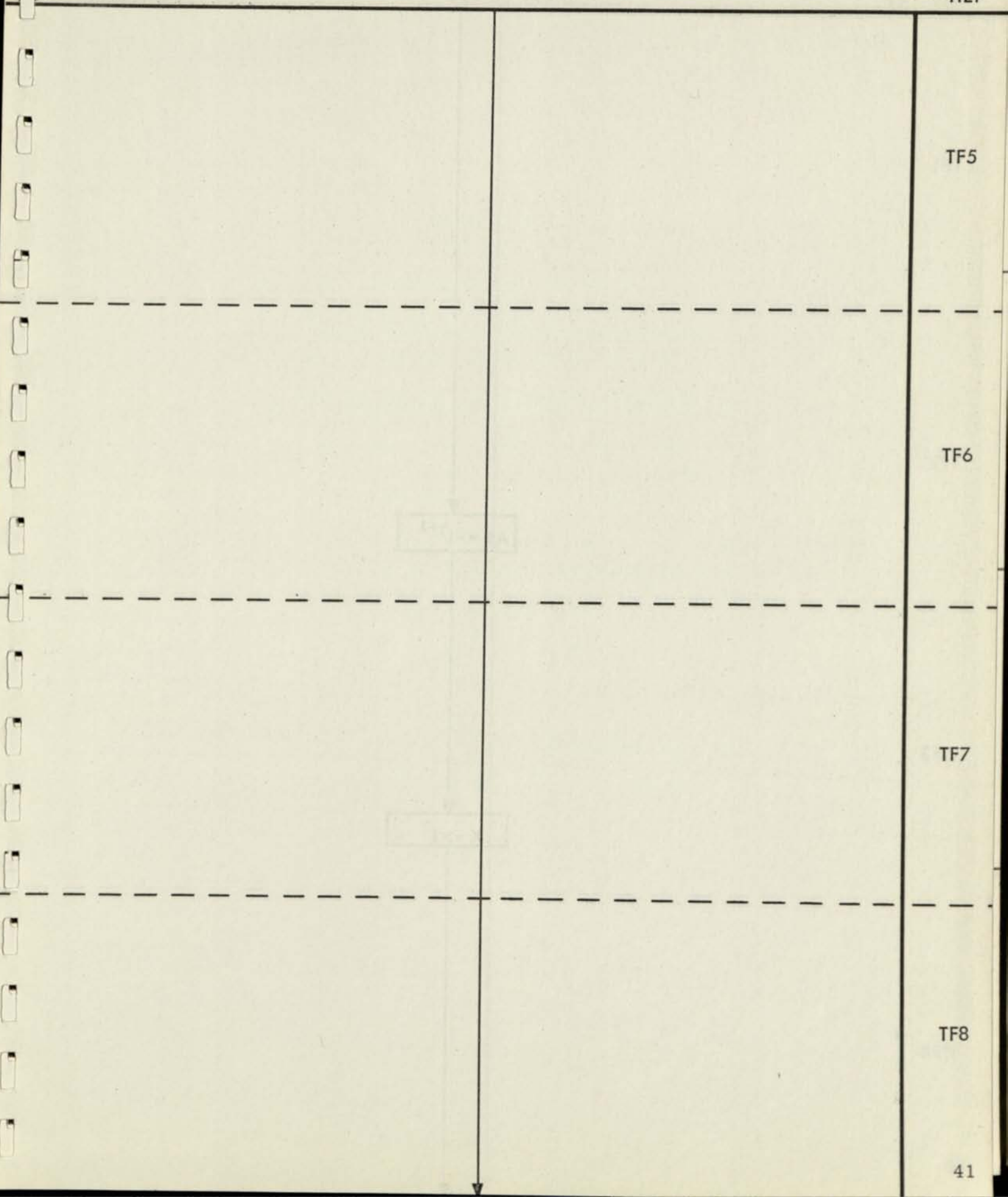


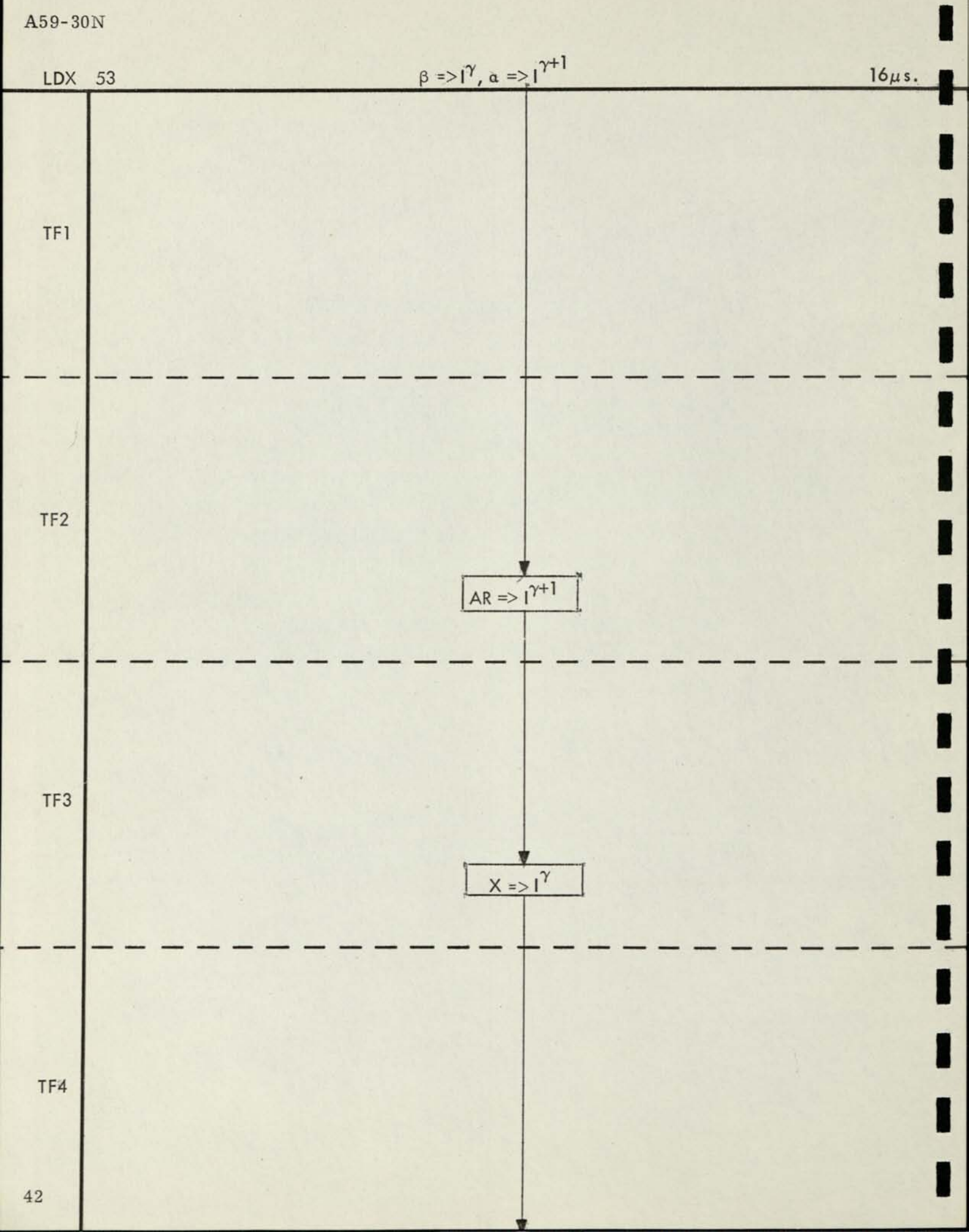
TF6

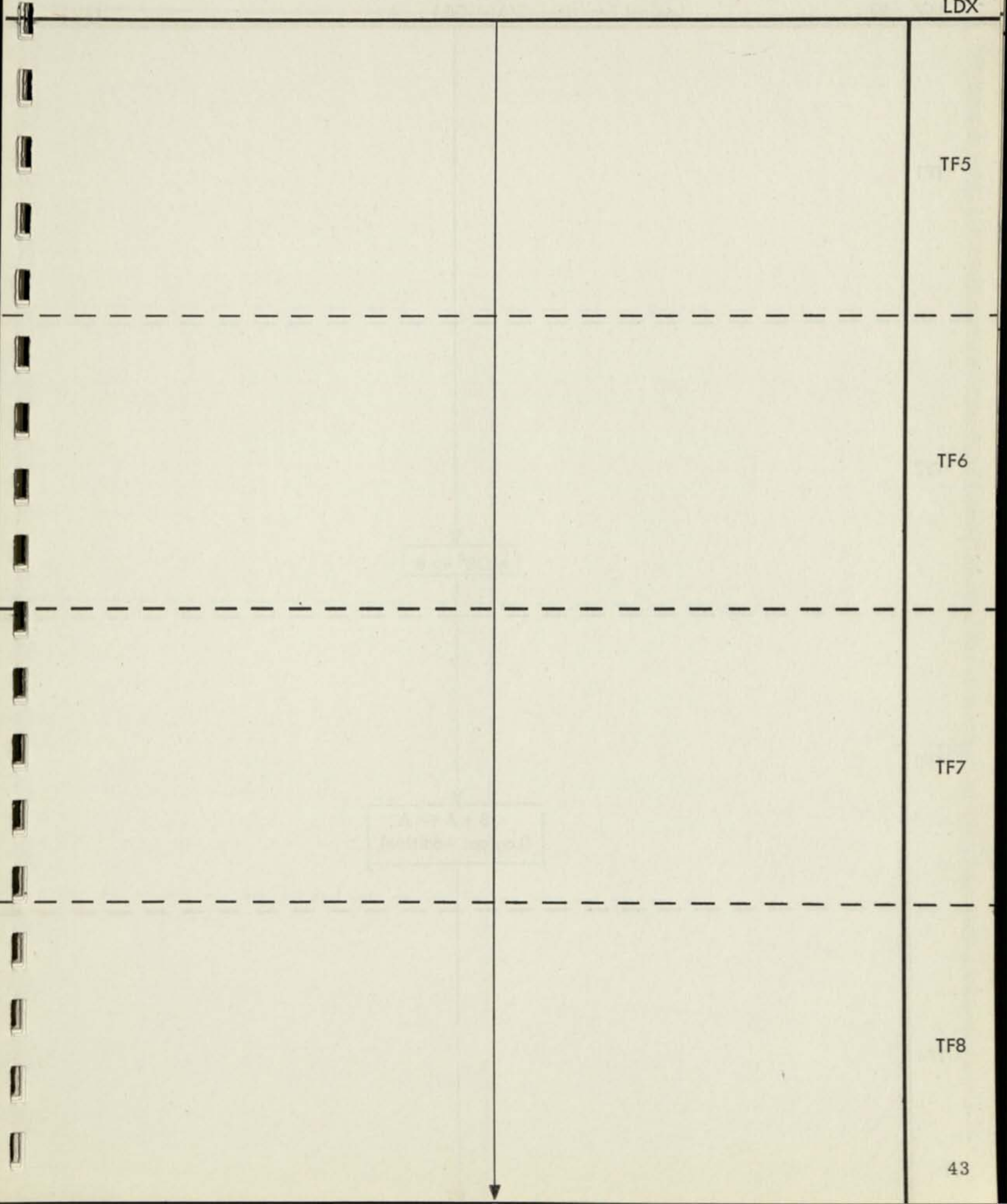
TF7

TF8


HLT 00
Computer Stops
16 μs.
TF1
TF2
TF3
TF4
1 => H


HLT
TF5
TF6
TF7
TF8


LDX 53
β => I^γ, α => I^γ+1
16μs.
TF1
TF2
AR => I^γ+1
TF3
X => I^γ
TF4


LDX
TF5
TF6
TF7
TF8

LGA 03 Logical Function: C(A) + C(α) => A 16 μs.

TF1

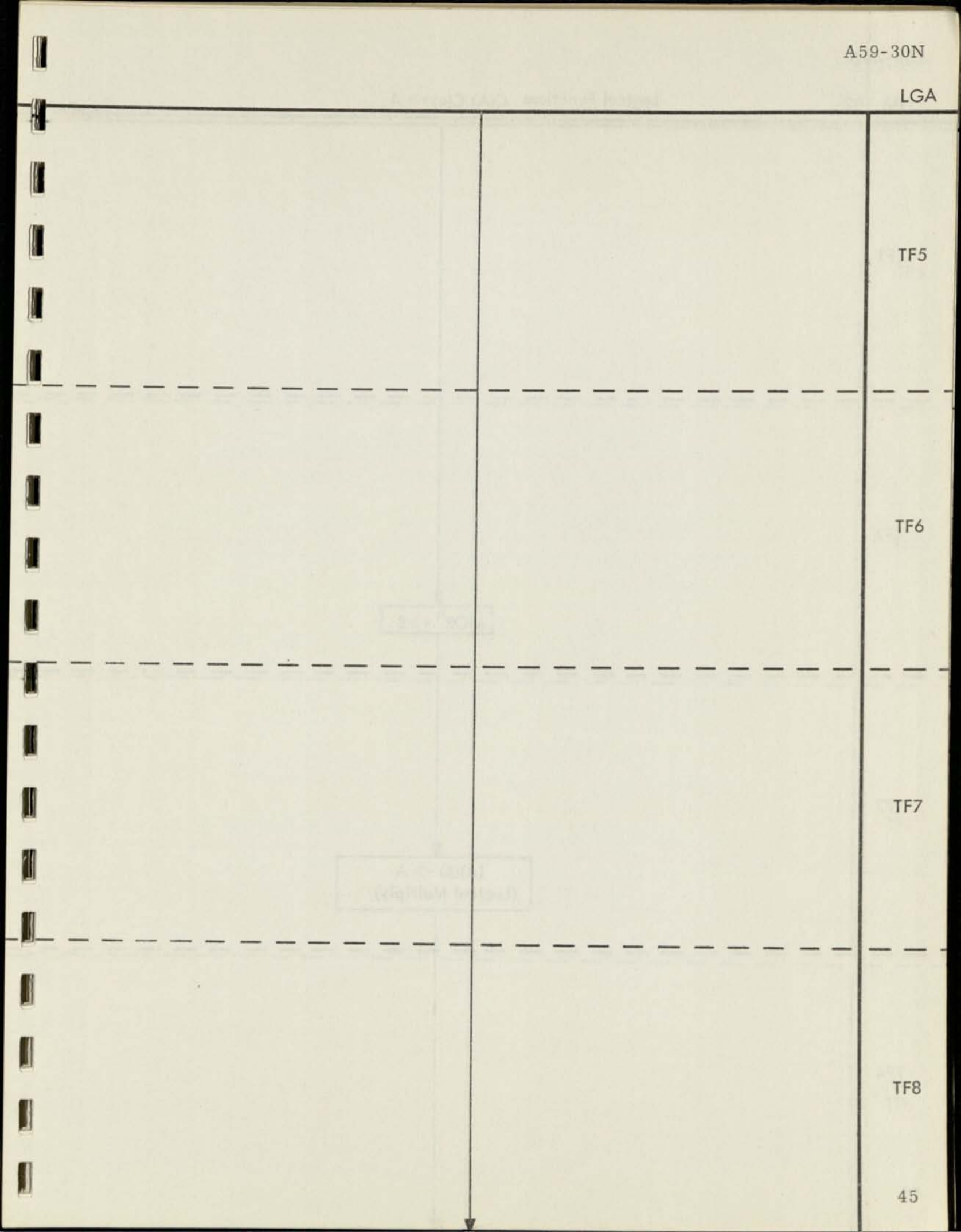
TF2

MOR^n => B

TF3

B + A => A
(Logical Addition)

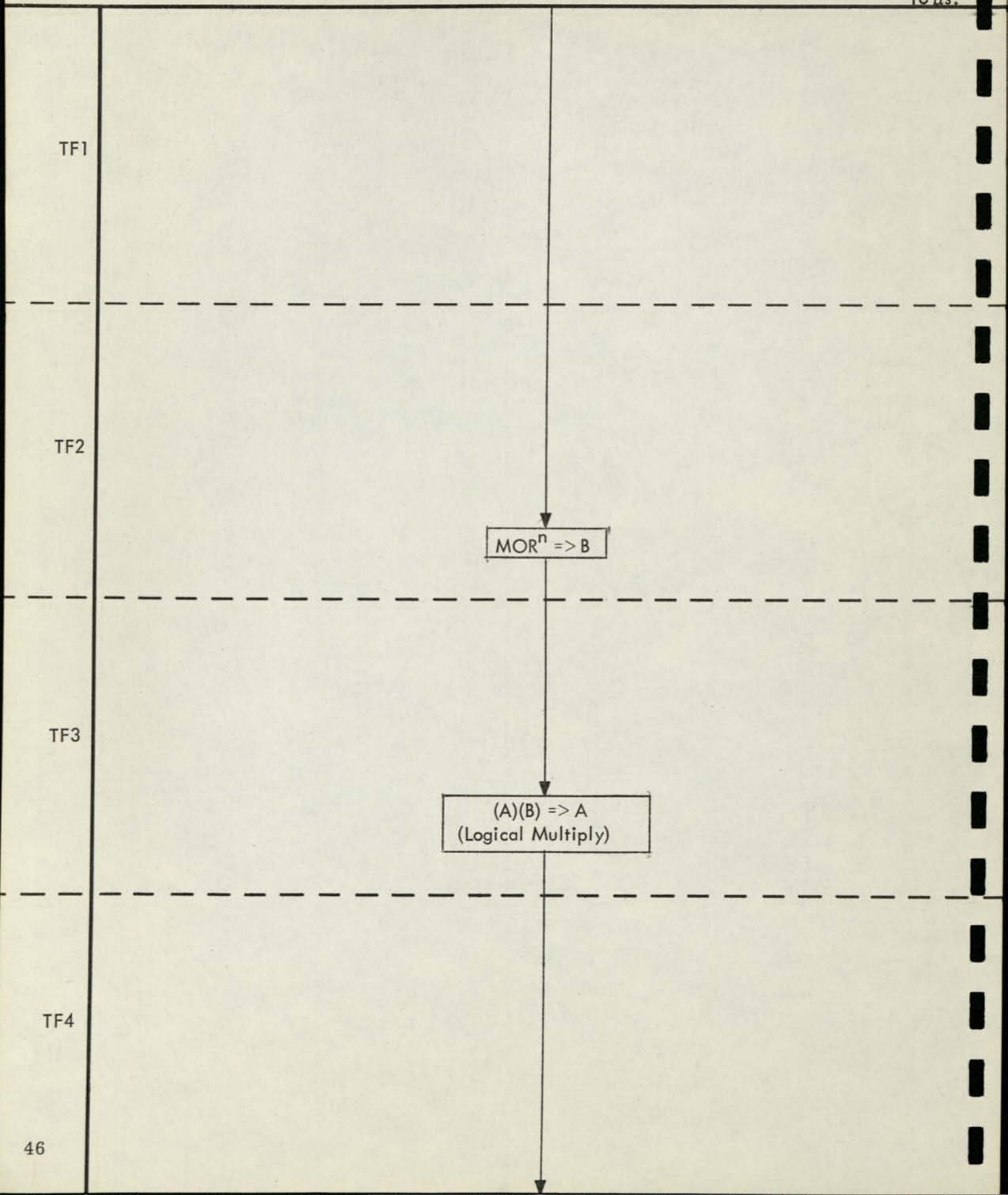
TF4


LGA
TF5
TF6
TF7
TF8
45

LGM 02

Logical Function: C(A) C(α) => A

16 μs.



LGM

TF5

TF6

TF7

TF8

LGN

$C(a)'_{sn,\ 1-36} \Rightarrow A_{sn,\ 1-36}$

16 μs.

TF1

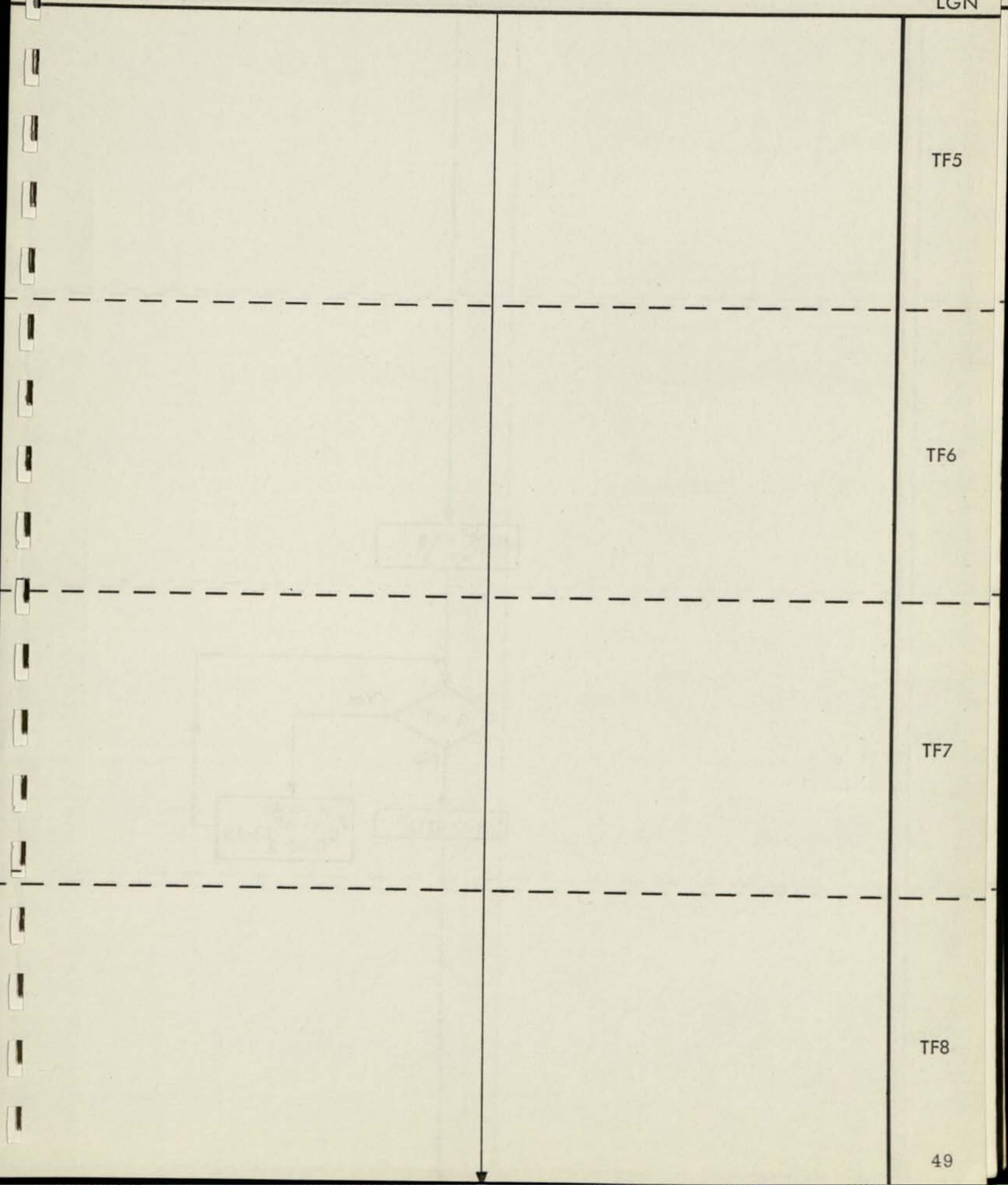
TF2

$MOR^n \Rightarrow A$

TF3

$A' \Rightarrow A$

TF4


LGN
TF5
TF6
TF7
TF8

LOD 51

$C(a) \Rightarrow Reg^n$

18 μs.

TF1

TF2

$MOR^n \Rightarrow B$
$1 \Rightarrow \lambda$

TF3

$\lambda = 1$

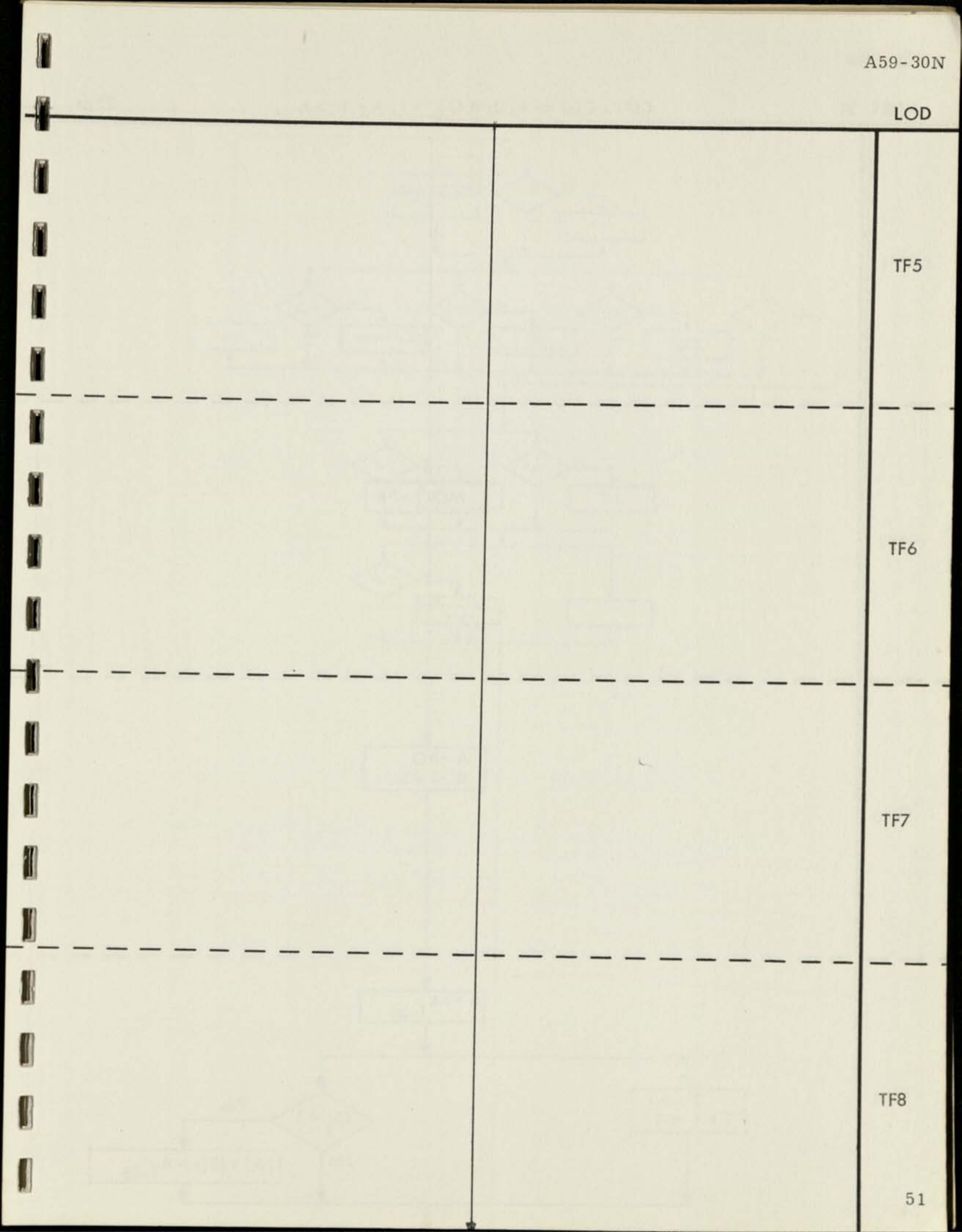
Yes

No

$B \Rightarrow REG^n$

$X \Rightarrow AR$
$7_8 \Rightarrow AR_{13-15}$
$0 \Rightarrow \lambda$

TF4


LOD
TF5
TF6
TF7
TF8

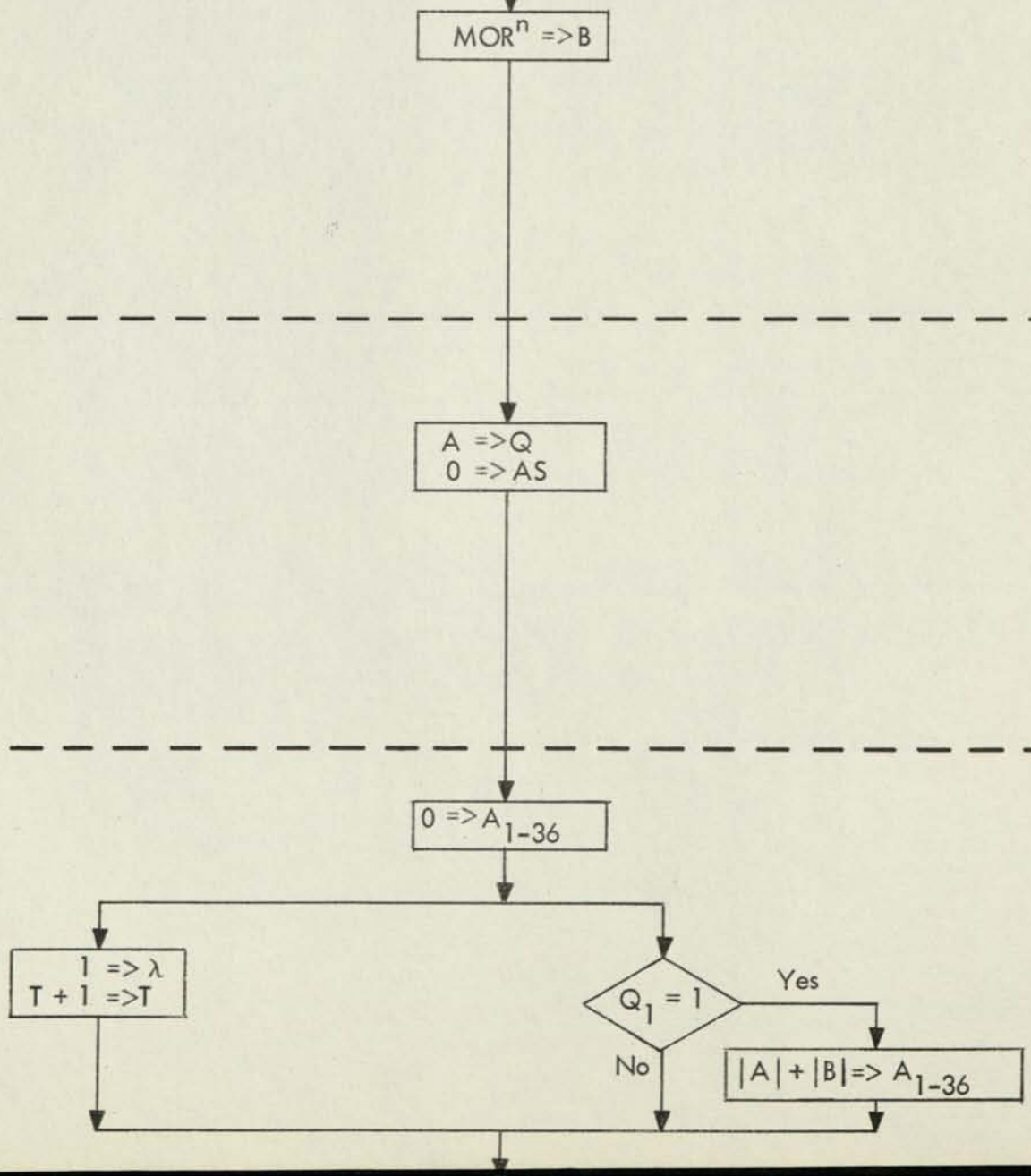
MLR 21 $C(A) \times C(a) \Rightarrow A, Q$; If $Q_{36} = 1$, $A + 1 \Rightarrow A$ 21 μs.

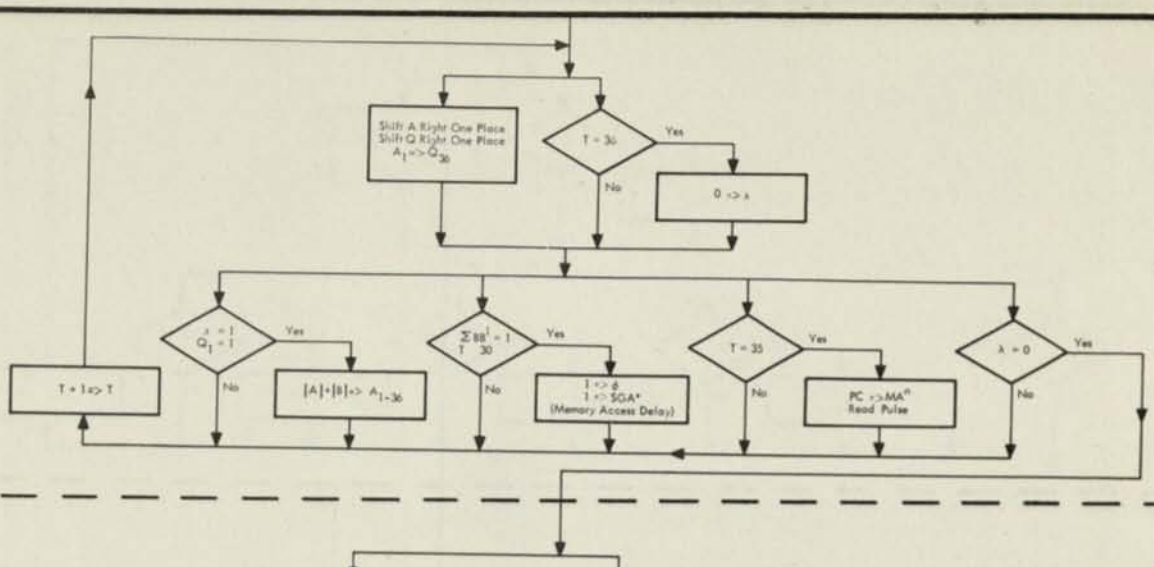
TF1

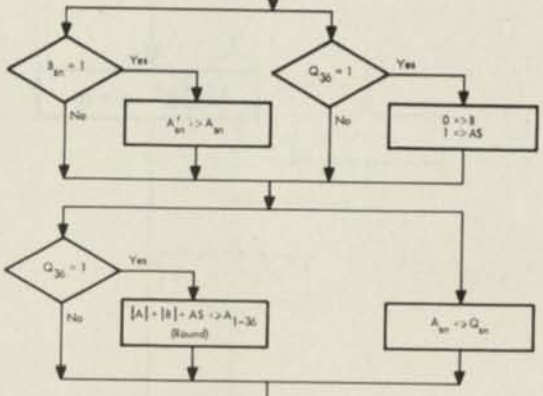
TF2

TF3

TF4







TF5

TF6

TF7

TF8

MLY 20 — C(A) x C(a) => A, Q — 86 μs.

TF1

TF2

MOR^{n} => B

TF3

A => Q
0 => AS

TF4

0 => A_{1-36}

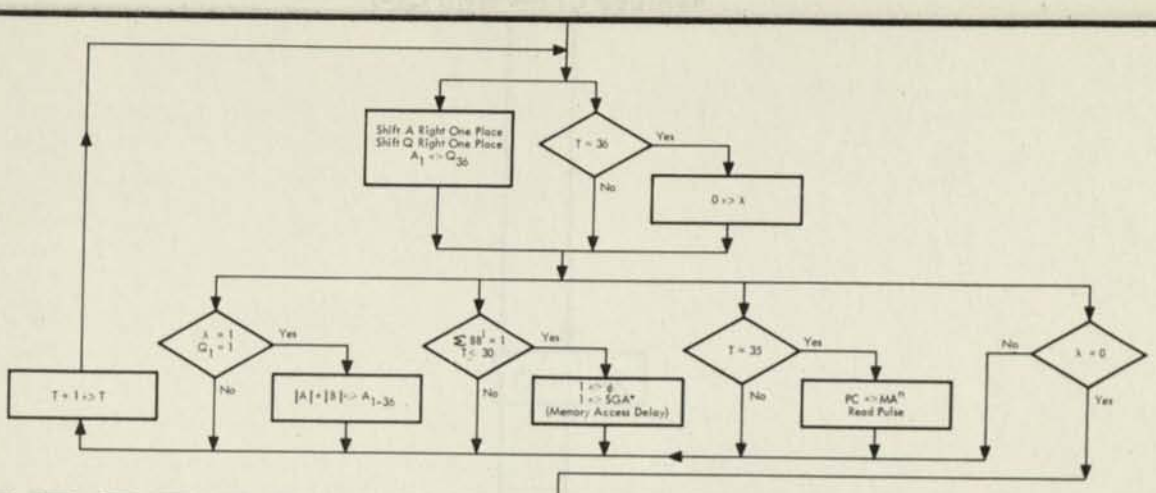
1 => λ
T + 1 => T

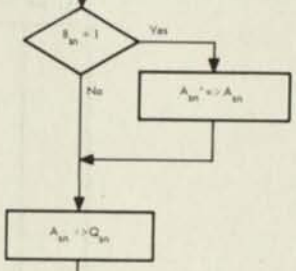
Q_1 = 1

Yes

No

|A|+|B| => A_{1-36}





TF5

TF6

TF7

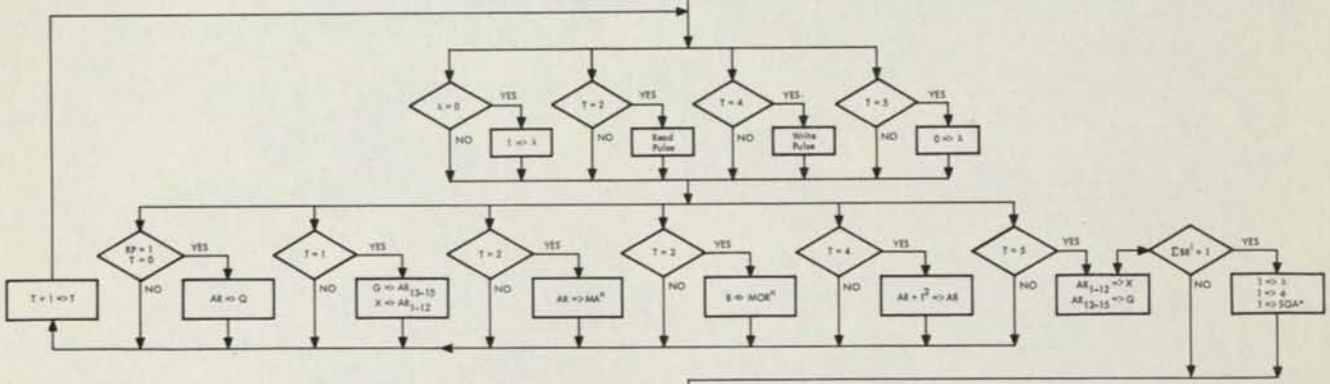
TF8

MOV 52 Replace C(γβ) With C(α) 26 μs.

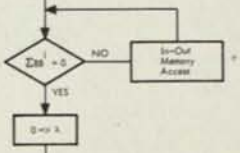
TF1

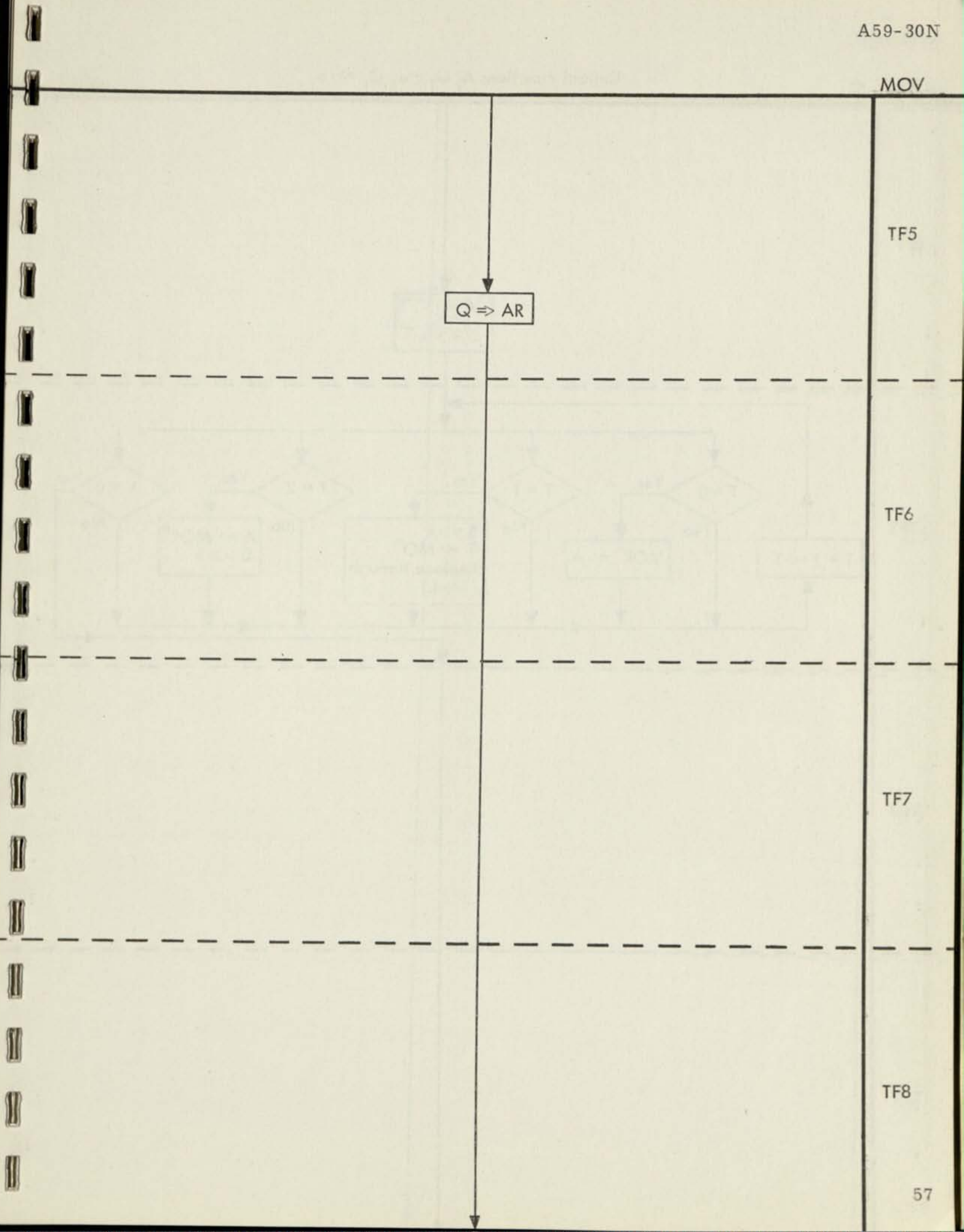
TF2

TF3



TF4




MOV
TF5
Q => AR
TF6
TF7
TF8

MSK 55

Logical Function: $A_i\ Q_i + a_i\ Q_i' \Rightarrow a_i$

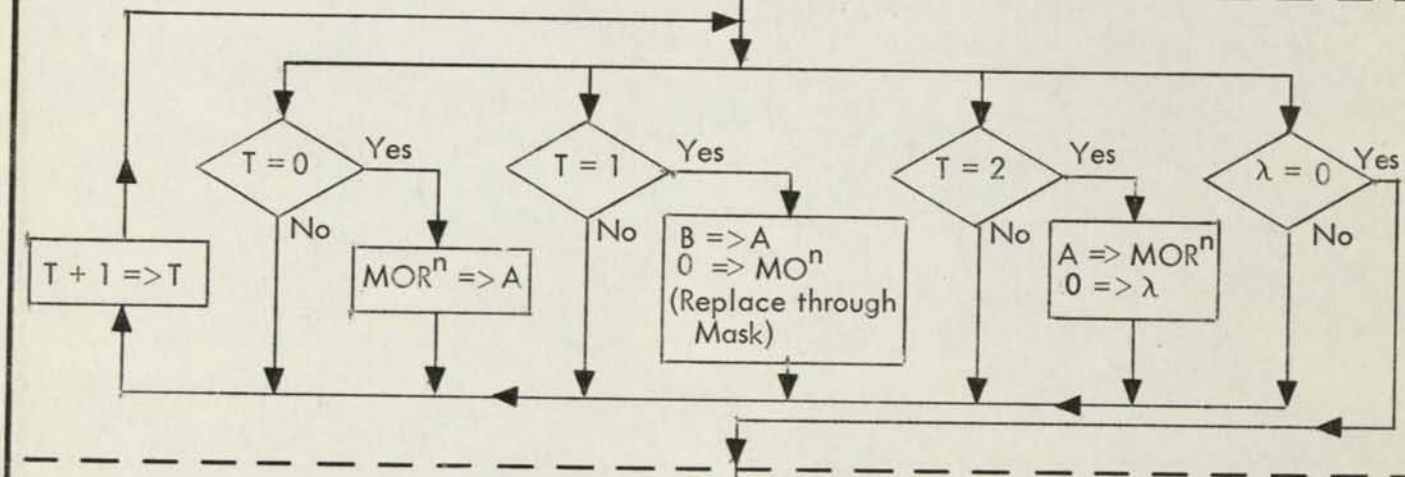
22 μs.

TF1

TF2

TF3

TF4



MSK

TF5

B => A

TF6

TF7

TF8

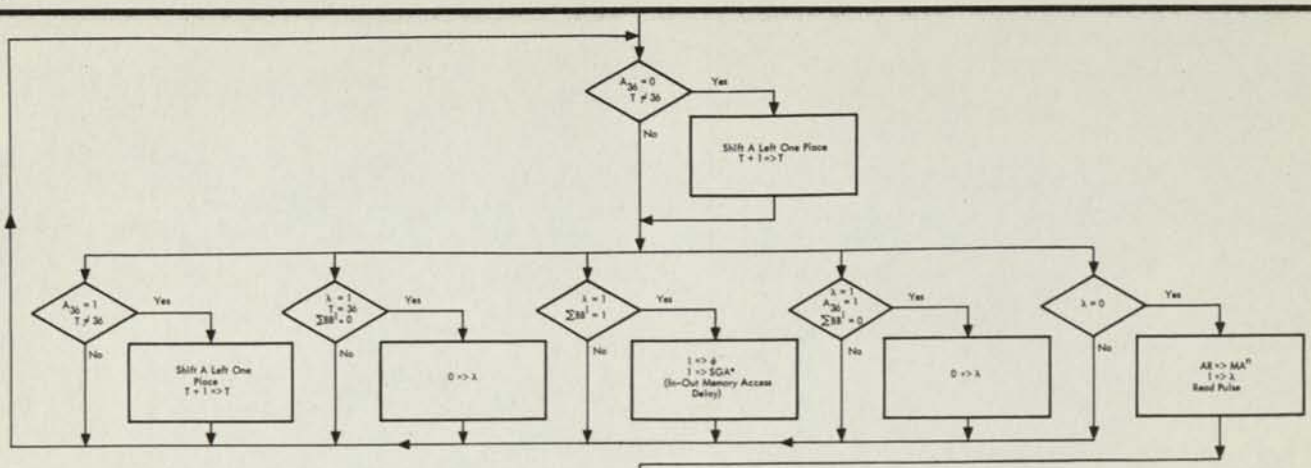
NRM 37

C(A) is Shifted Left Until $A_{36} = 1$; no. of shifts $\Rightarrow$ a

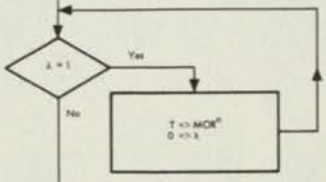
$18 + n - n(\text{mod } 2)\mu s.$

n = no. of Shifts

TF1



TF2



TF3

TF4

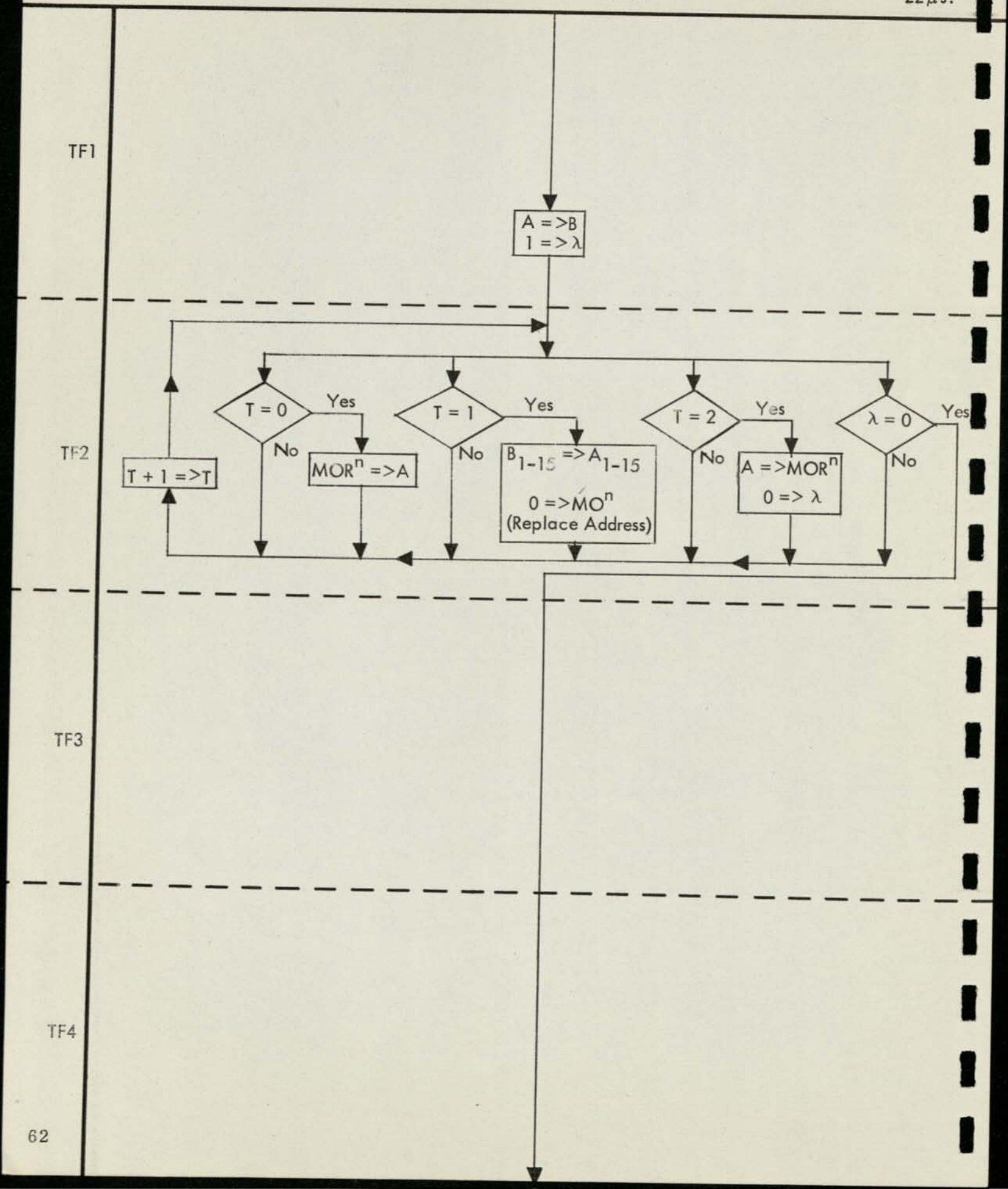
TF5

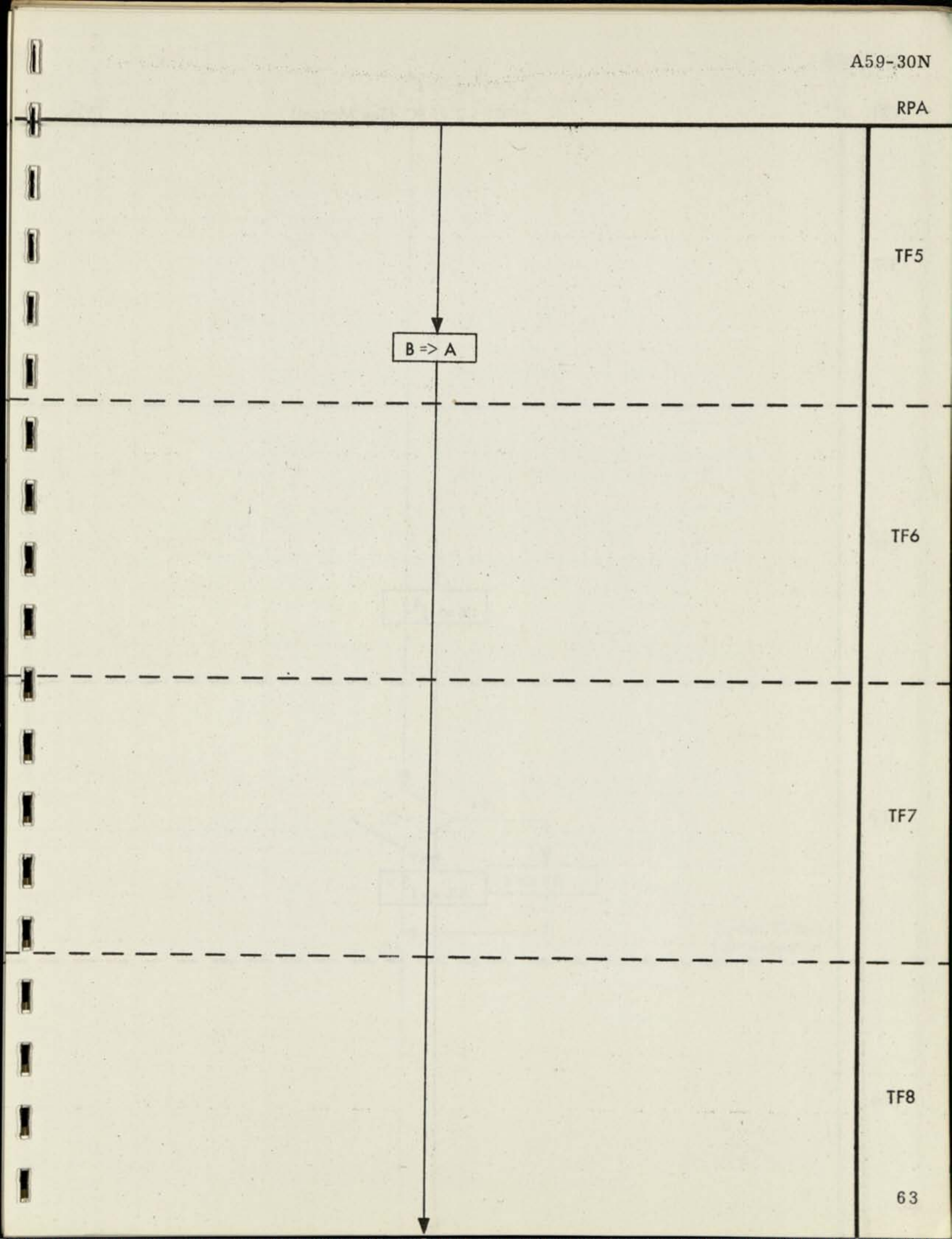
TF6

TF7

TF8

RPA 54 — Replace $C(a)_{1-15}$ with $C(A)_{1-15}$ — 22 μs.




RPA
TF5
B => A
TF6
TF7
TF8

RPT 01

$\alpha \Rightarrow I^3; \beta \Rightarrow I^4$

C(PC) + 2 => PC (See Manual)

16 μs.

TF1

TF2

$X \Rightarrow I^4$

TF3

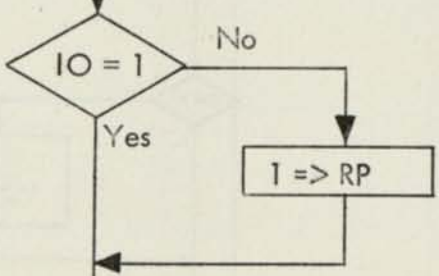
$AR \Rightarrow I^3$

TF4

RPT

TF5

TF6



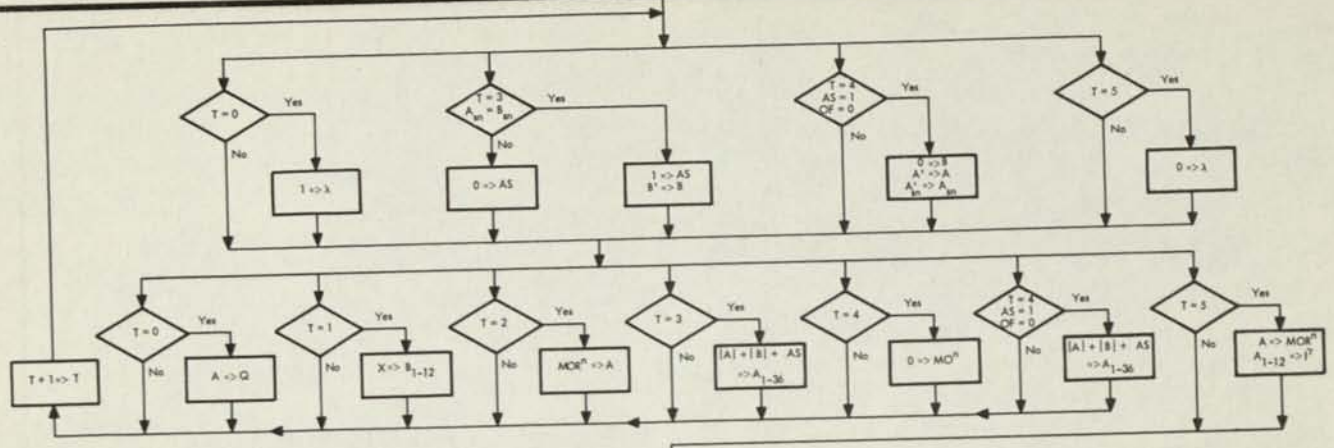
TF7

In-out Orders
Not Repeated

TF8

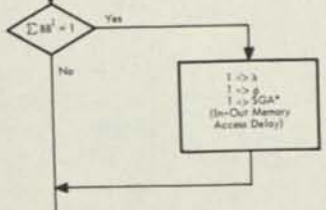
SBB 25 $C(A) \Rightarrow Q,\ C(a) - B \Rightarrow A,\ a,\ I^{\gamma}$ 26 μs.

TF1

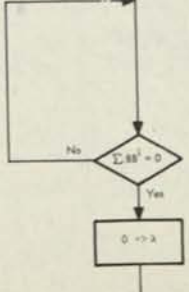


TF2

TF3



TF4



SBB

TF5

TF6

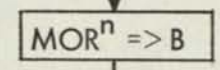
TF7

TF8

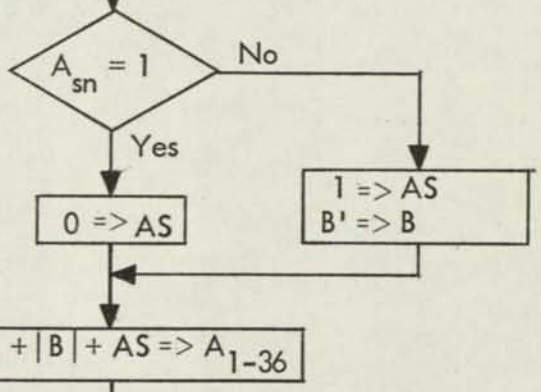
SBM 17 — C(A) - |C(a)| => A — 16 μs.

TF1

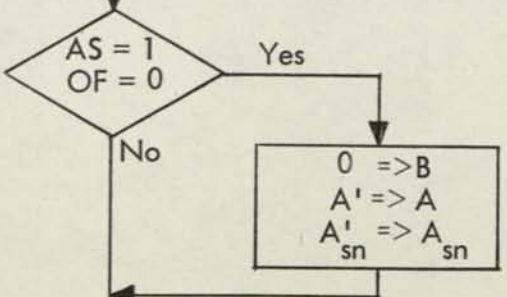
TF2

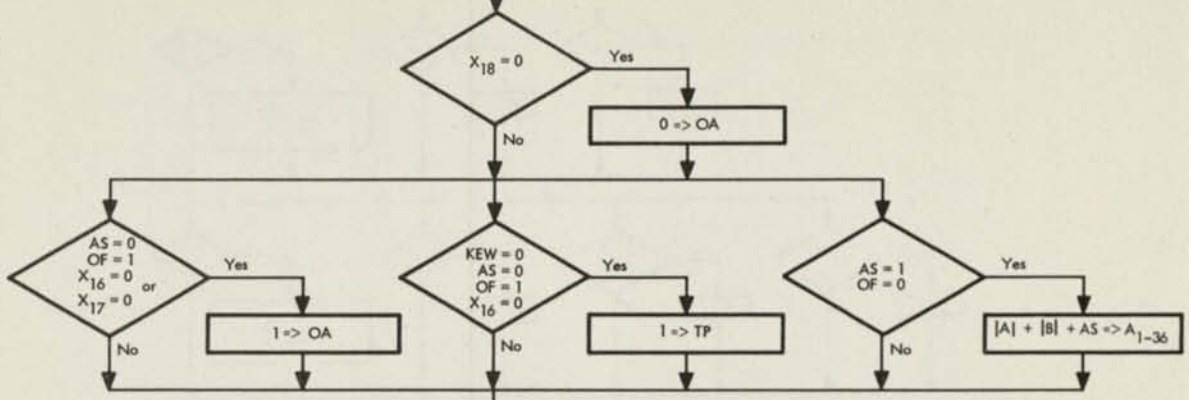


TF3



TF4





TF5

TF6

TF7

TF8

SEN 05

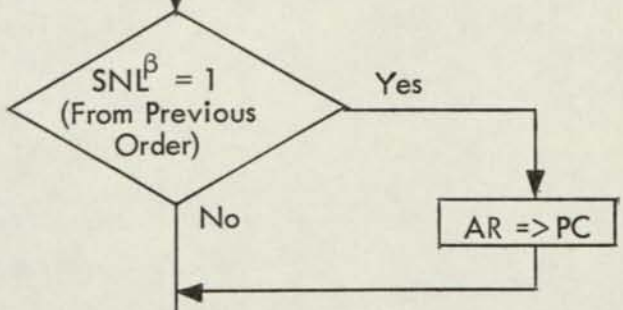
If C(β) = 1, α => PC;
If C(β) = 0, PC + 1 => PC

16μs.

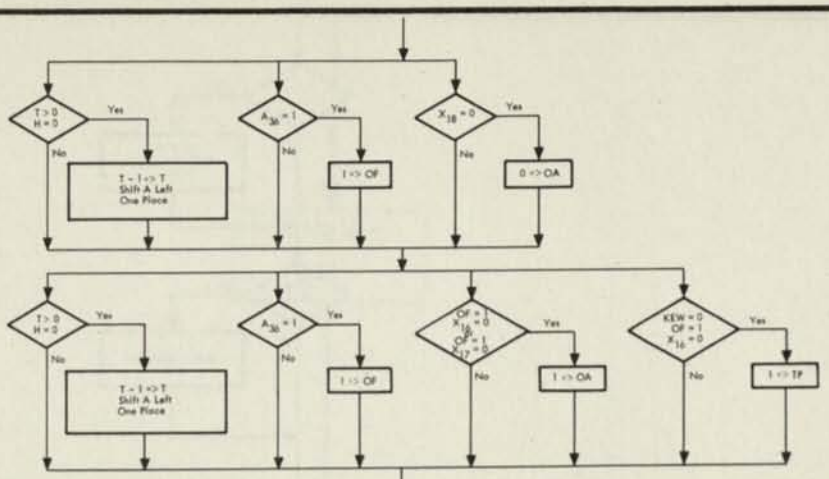
TF1

TF2

TF3



TF4



TF5

TF6

TF7

TF8

SHR 32

C(A) is Shifted a(mod 128) Places Right

$a \leq 14$ 16 μs.

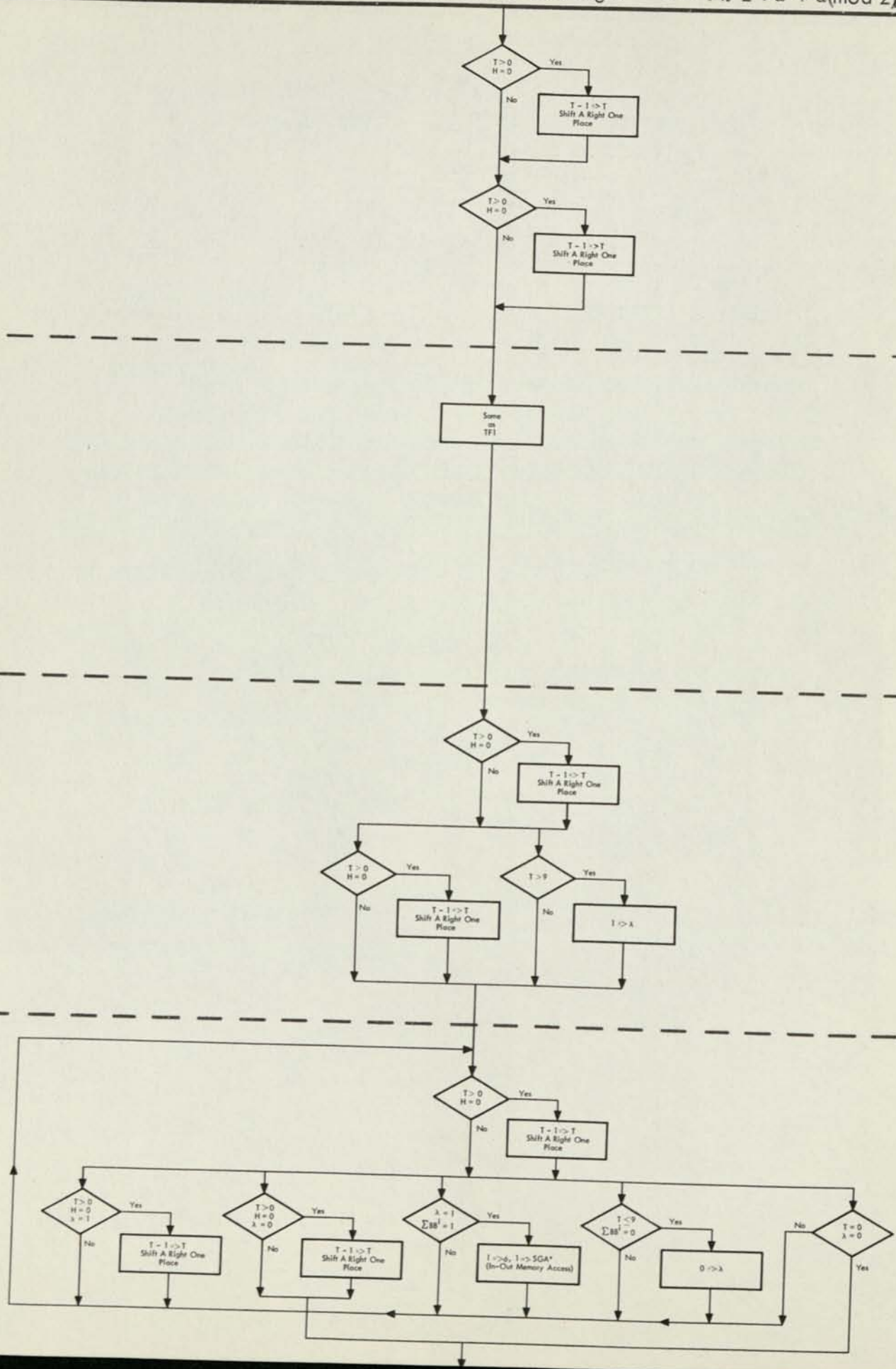
$a > 14$: 2 + a + a(mod 2) μs.

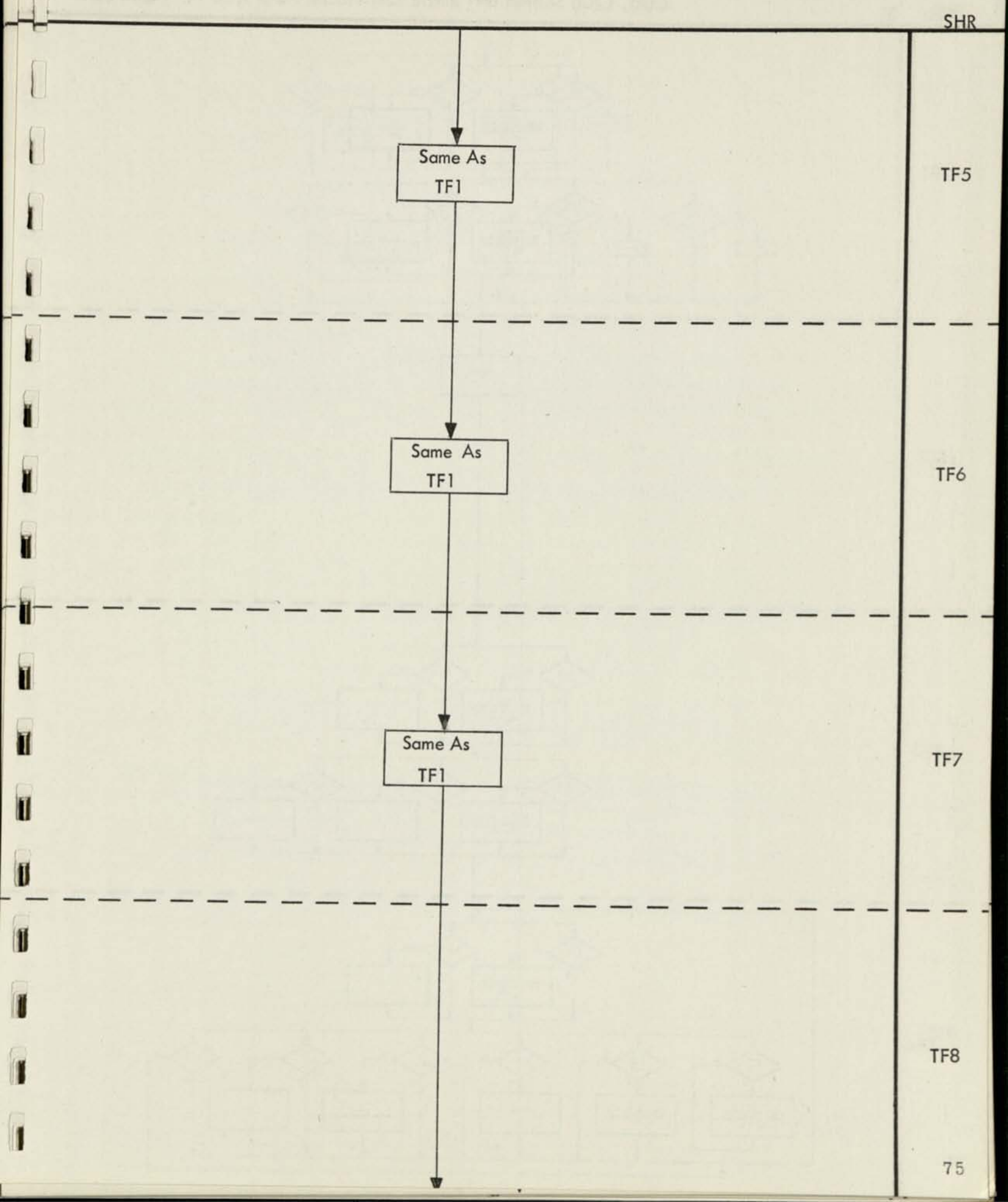
TF1

TF2

TF3

TF4




SHR
Same As TF1
TF5
Same As TF1
TF6
Same As TF1
TF7
TF8

SLL 31

C(A), C(Q) Shifted Left a(mod 128) Places

$a \leq 9$: 16 μs.

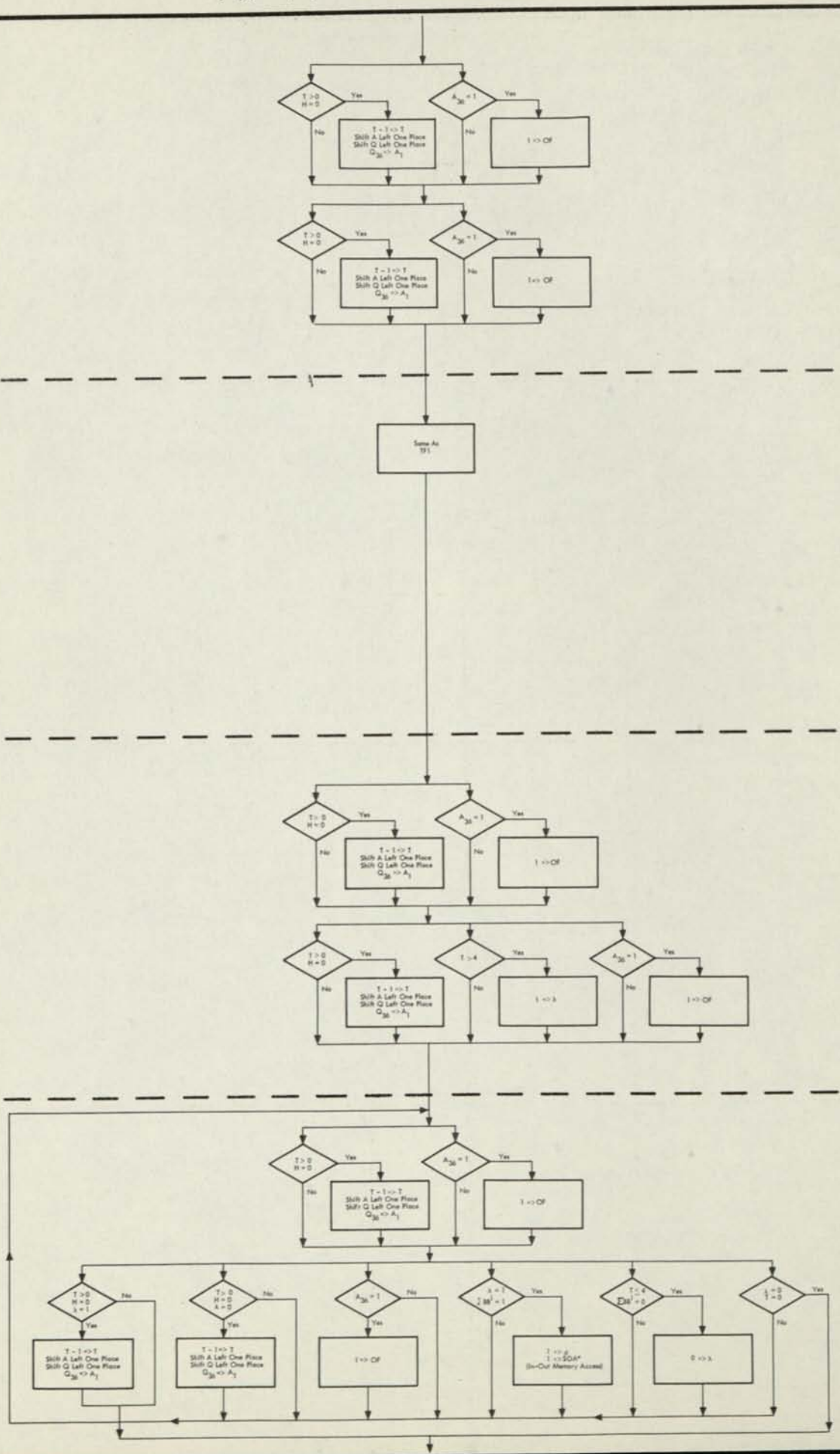
$a > 9$: $8 + a - a(\text{mo } 2)$ μs.

TF1

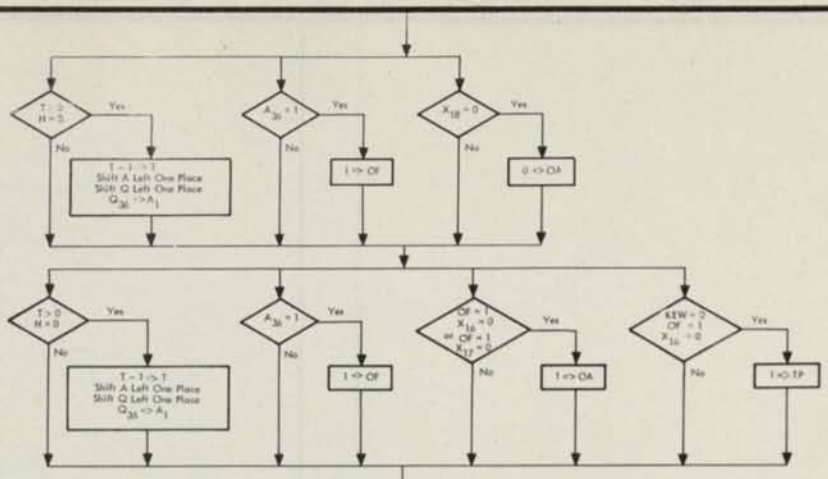
TF2

TF3

TF4



SLL



TF5

TF6

TF7

TF8

A59-30N

If C(β) = 1, 0 => C(β), α => PC;

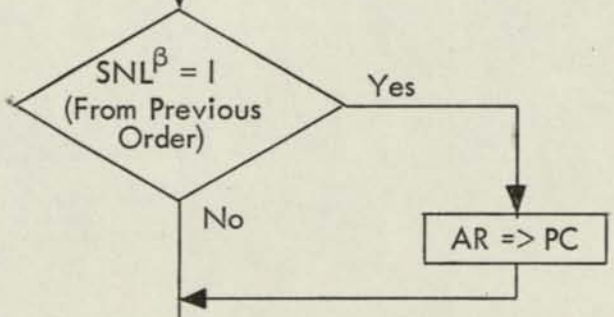
SNR 07 If C(β) = 0, PC + 1 => PC 16 μs.

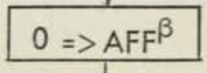
TF1

TF2

TF3

TF4





SNR

TF5

TF6

TF7

TF8

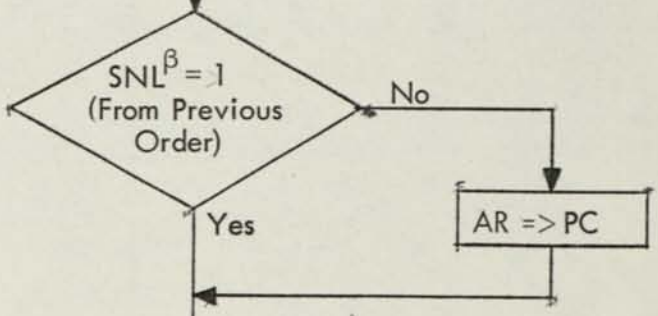
If C(β) = 0, 1 => C(β), α => PC;

SNS 06 If C(β) = 1, PC + 1 => PC 16 μs.

TF1

TF2

TF3



TF4

1 => AFF^β

SNS

TF5

TF6

TF7

TF8

SRL 33

C(A), C(Q) Shifted Right a(mod 128) Places

$a \leq 14$: 16 μs.

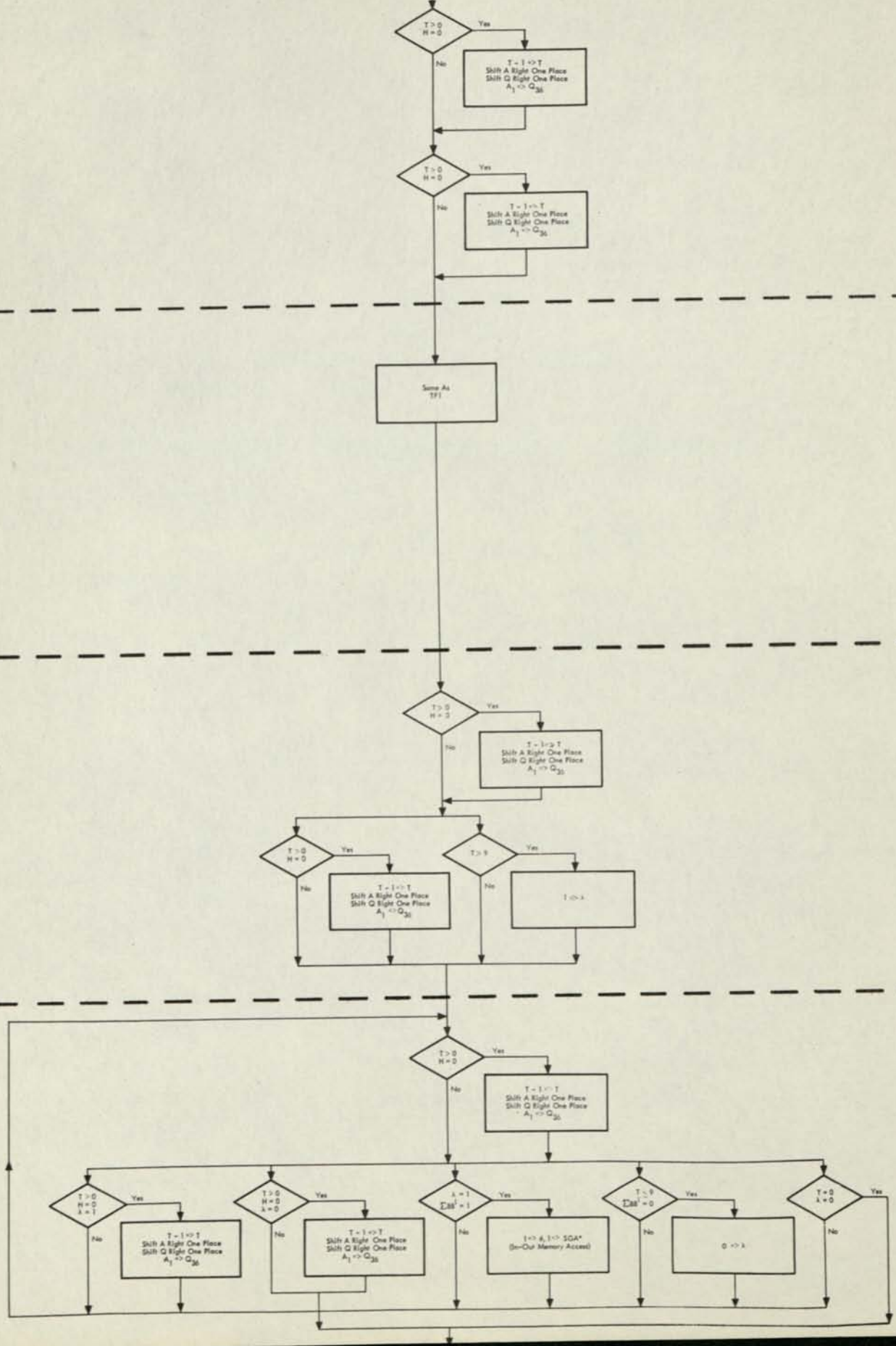
$a > 14$: $2 + a + a(\text{mod } 2)$ μs.

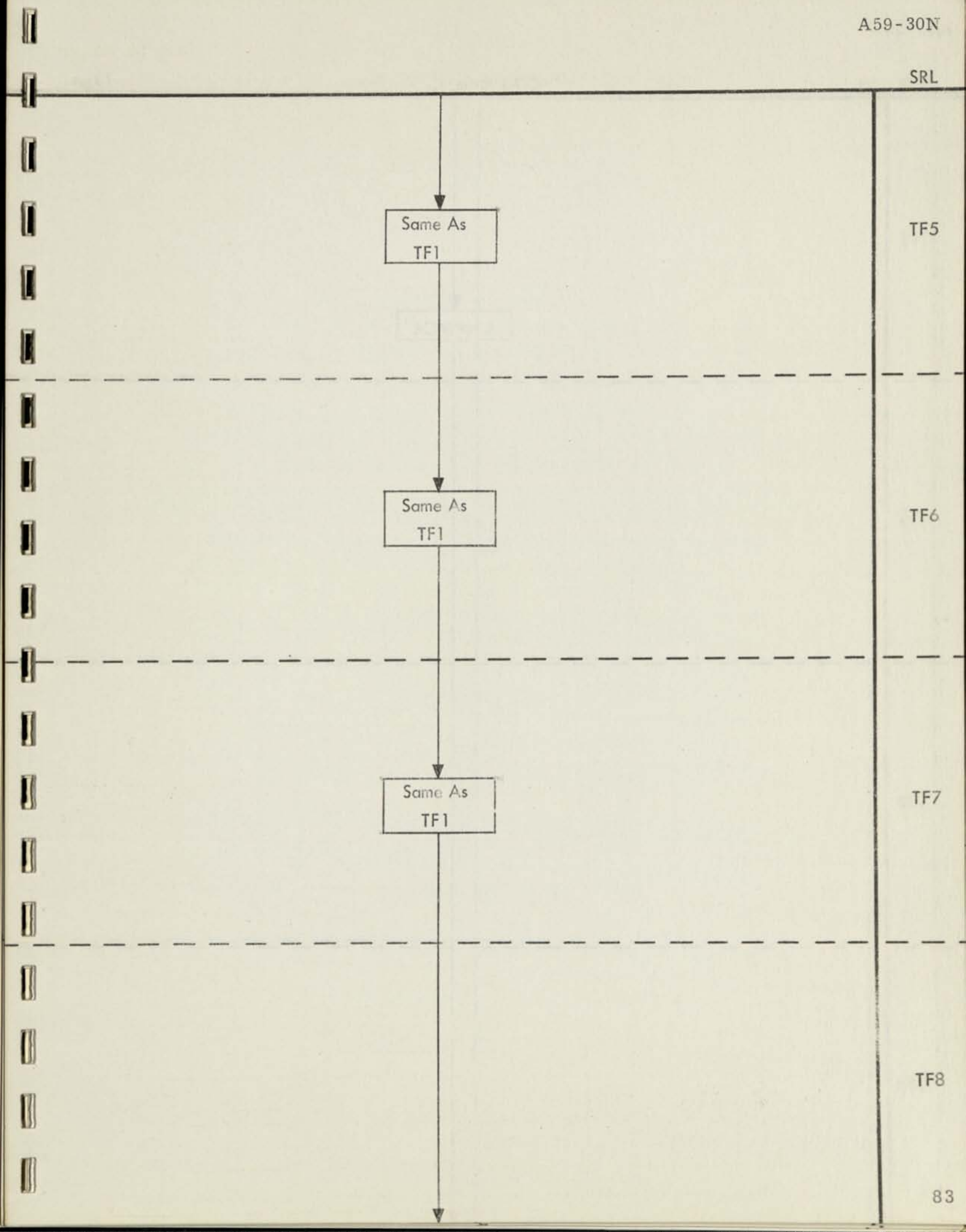
TF1

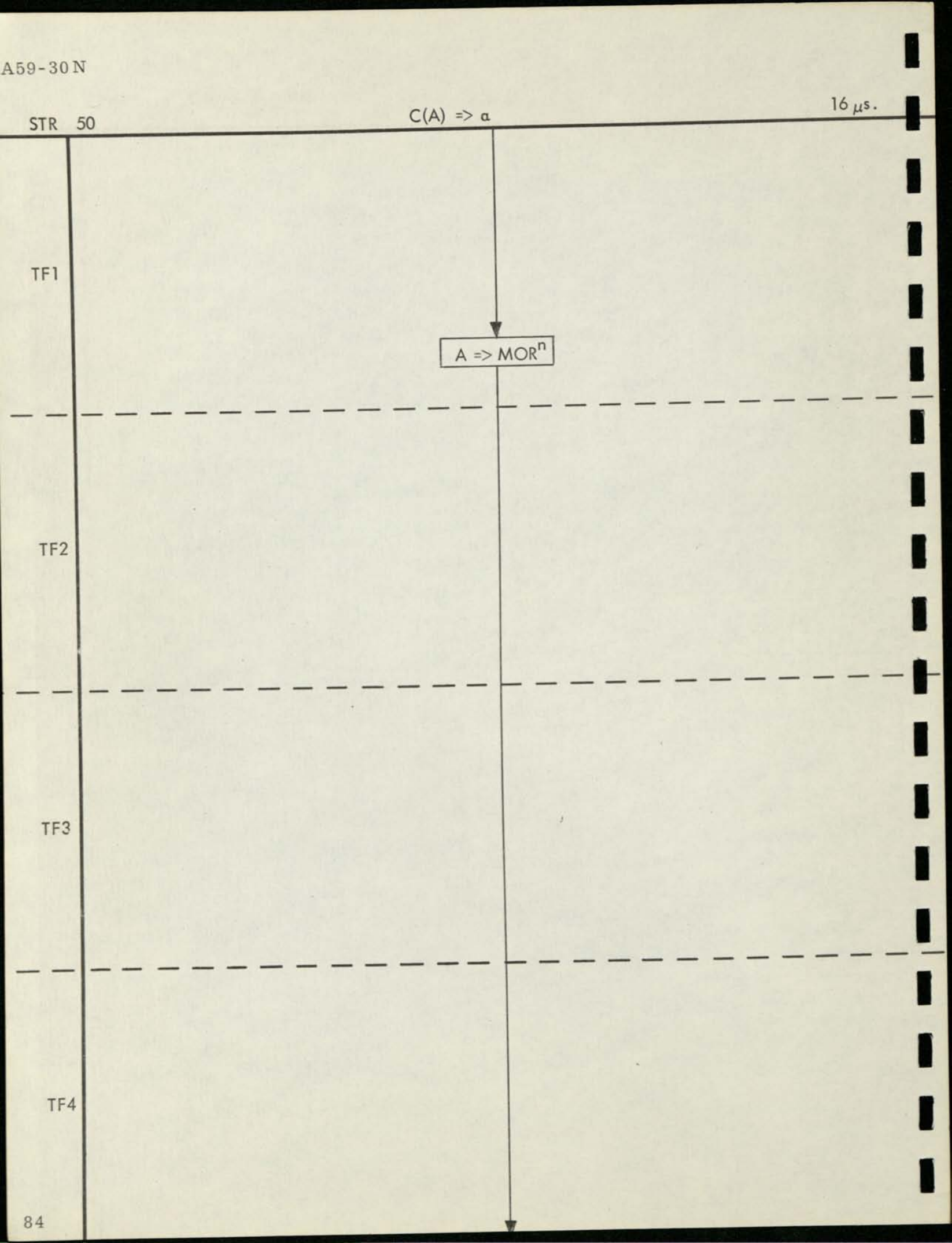
TF2

TF3

TF4




SRL
Same As TF1
TF5
Same As TF1
TF6
Same As TF1
TF7
TF8


STR 50
C(A) => a
16 μs.
TF1
A => MOR^n
TF2
TF3
TF4

STR

TF5

TF6

TF7

TF8

SUB 16 C(A) - C(a) => A 16μs.

TF1

TF2

MOR^n => B

TF3

A_sn = B_sn — Yes → 1 => AS; B' => B

No

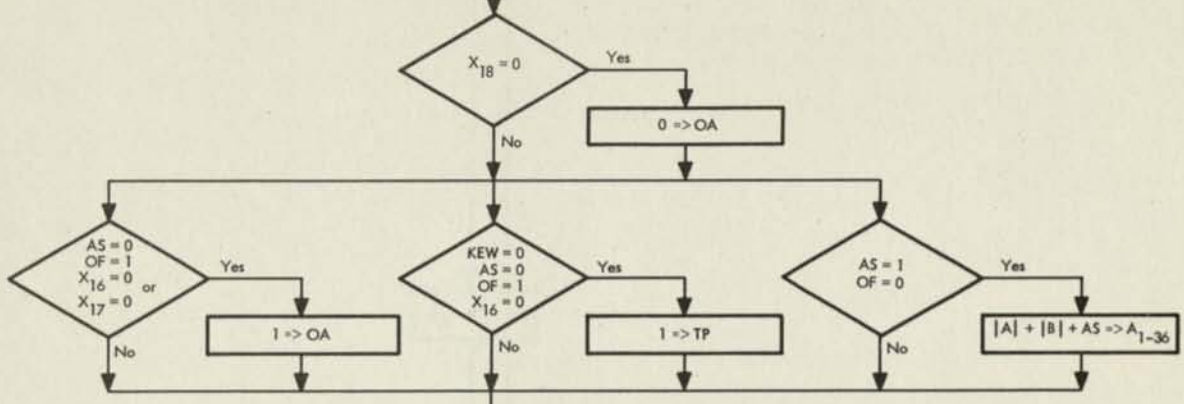
0 => AS

|A| + |B| + AS => A_1-36

TF4

AS = 1 OF = 0 — Yes → 0 => B; A' => A; A'_sn => A_sn

No



TF5

TF6

TF7

TF8

TRC 47

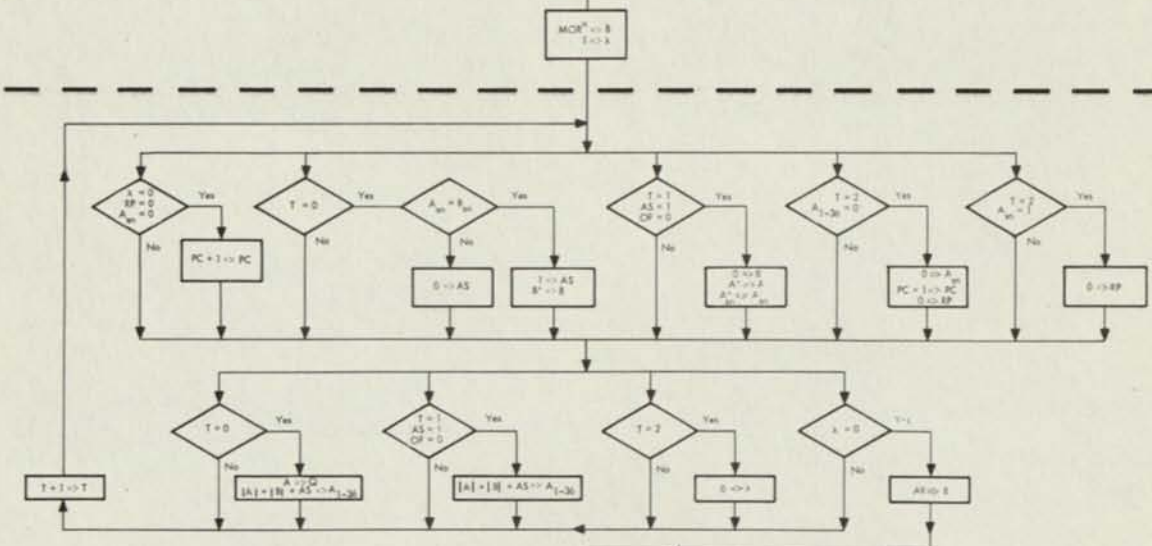
If C(A) < C(α), PC + 1 => PC;
C(A) > C(α), PC + 2 => PC; C(A) = C(α), PC + 3 => PC

22 μs.

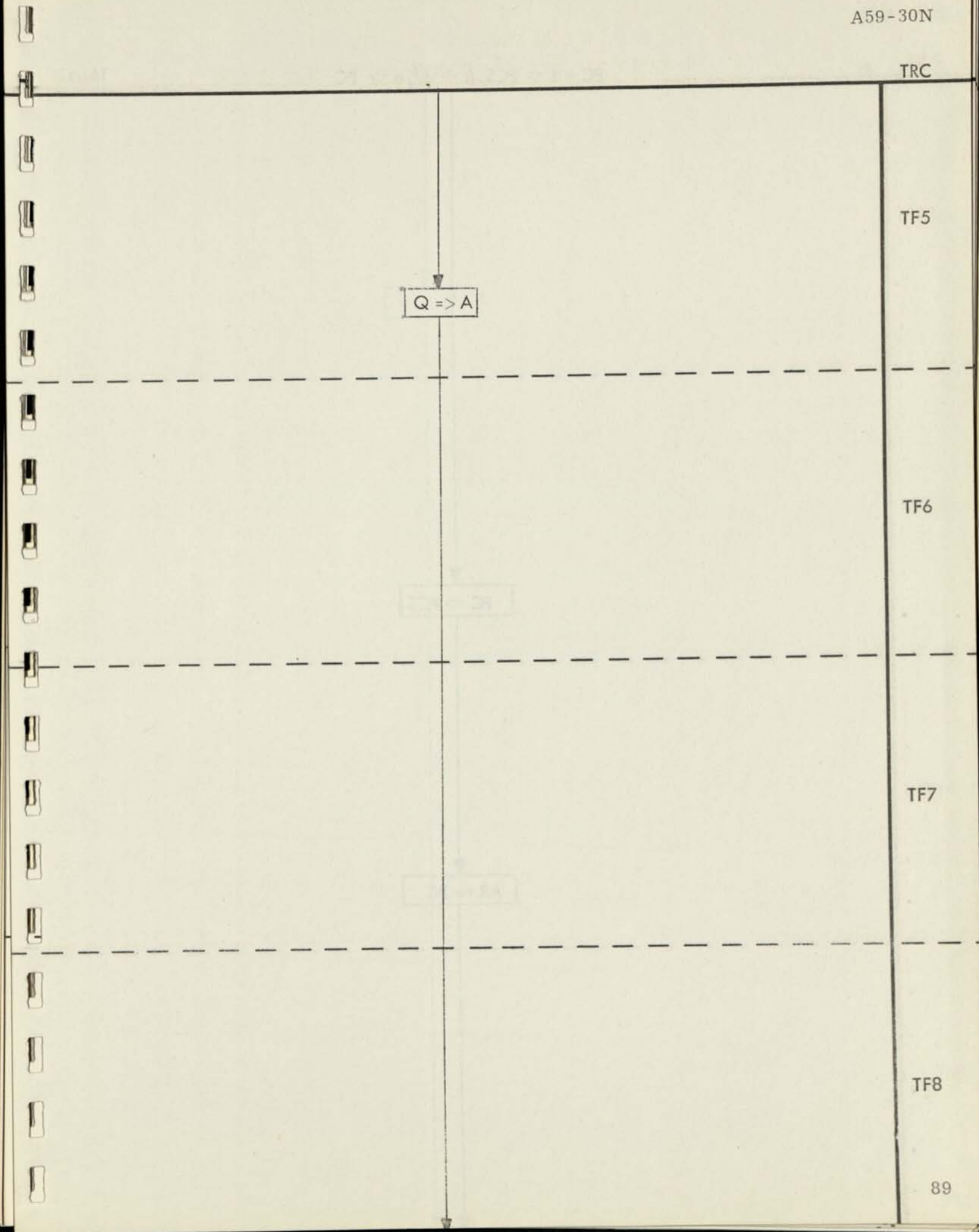
TF1

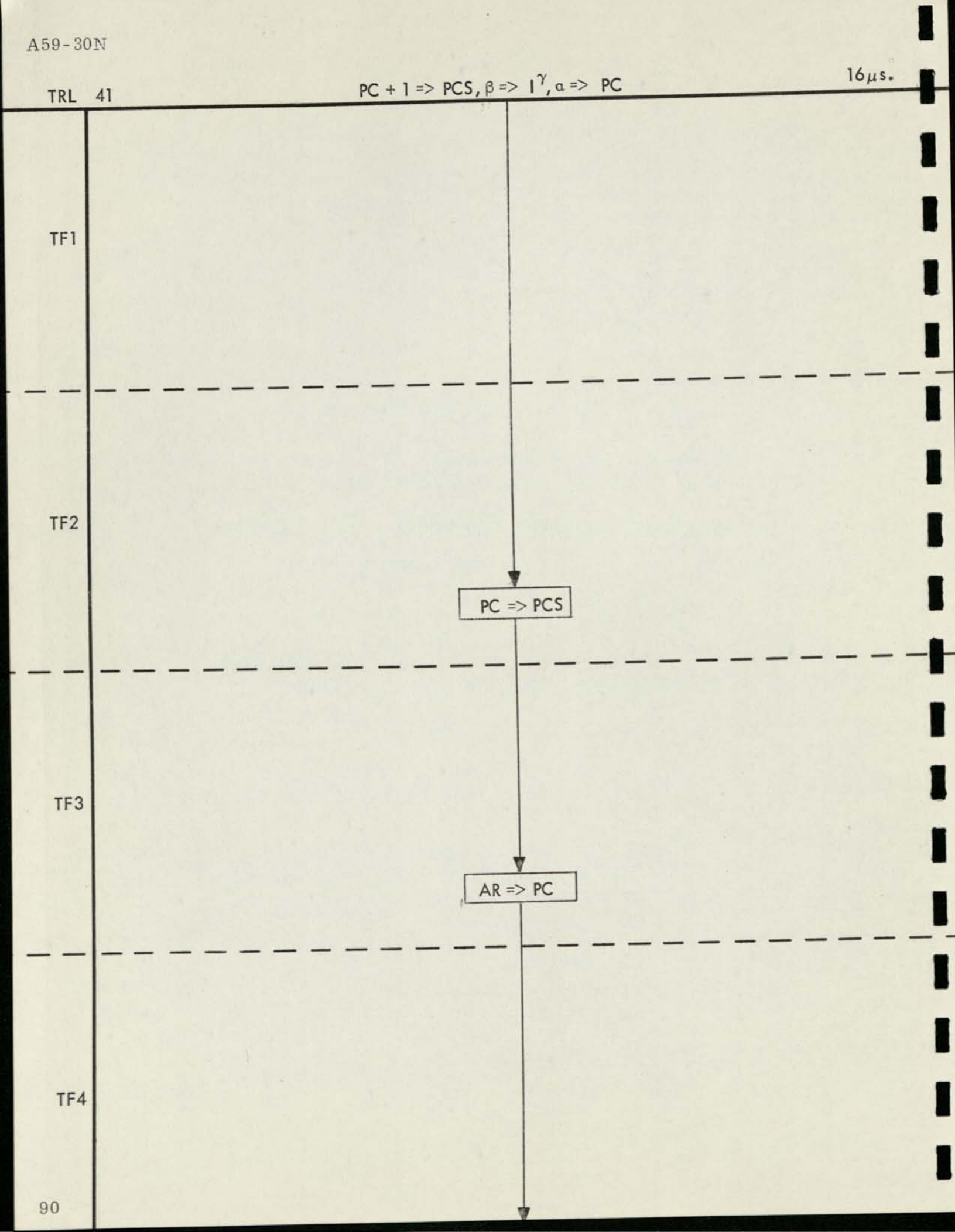
TF2

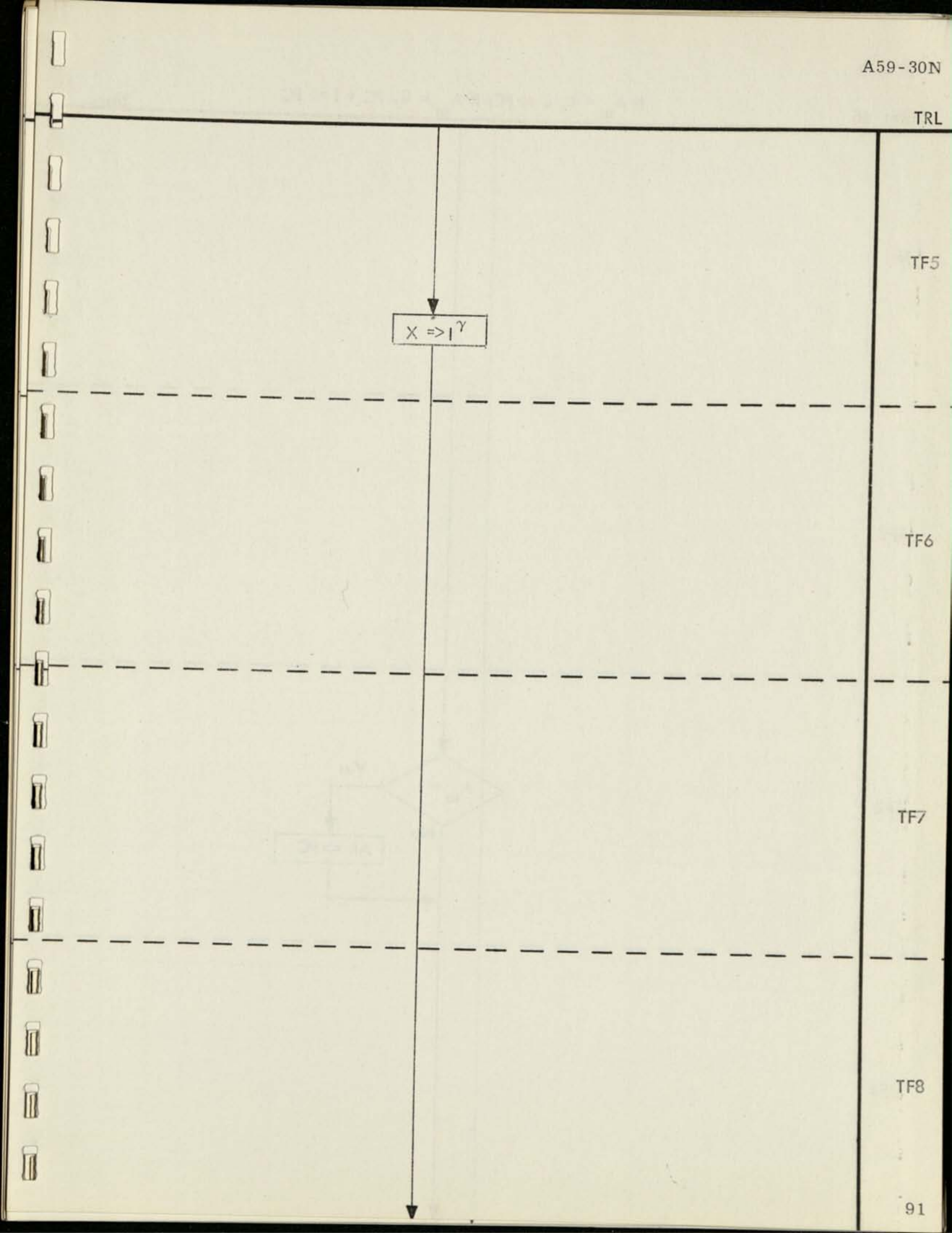
TF3



TF4


TRC
TF5
Q => A
TF6
TF7
TF8


TRL 41
PC + 1 => PCS, β => I^γ, α => PC
16μs.
TF1
TF2
PC => PCS
TF3
AR => PC
TF4


TRL
TF5
X =>1 γ
TF6
TF7
TF8

TRN 46

If A_{sn} = 1, a => PC; If A_{sn} = 0, PC + 1 => PC

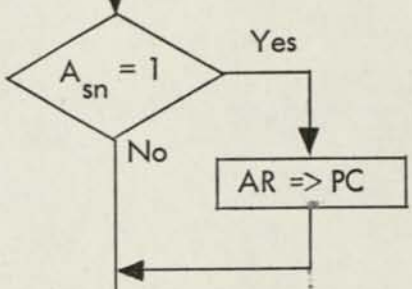
16μs.

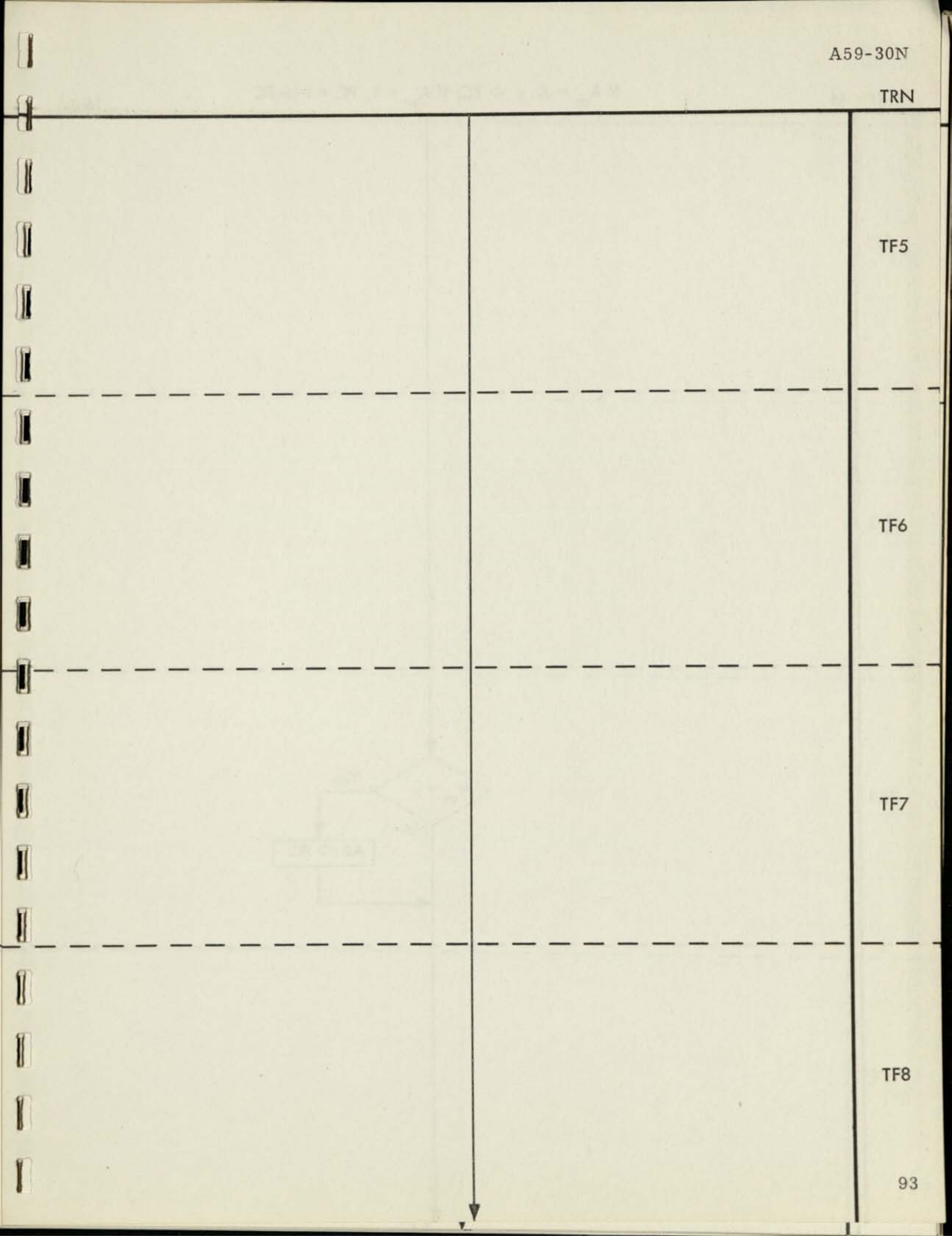
TF1

TF2

TF3

TF4




TRN
TF5
TF6
TF7
TF8

TRP 44

If A_{sn} = 0, a => PC; If A_{sn} = 1, PC + 1 => PC

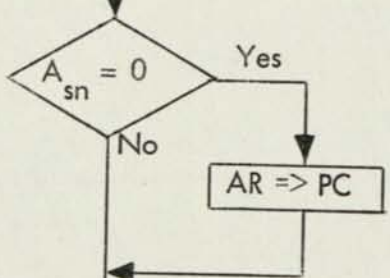
16 μs.

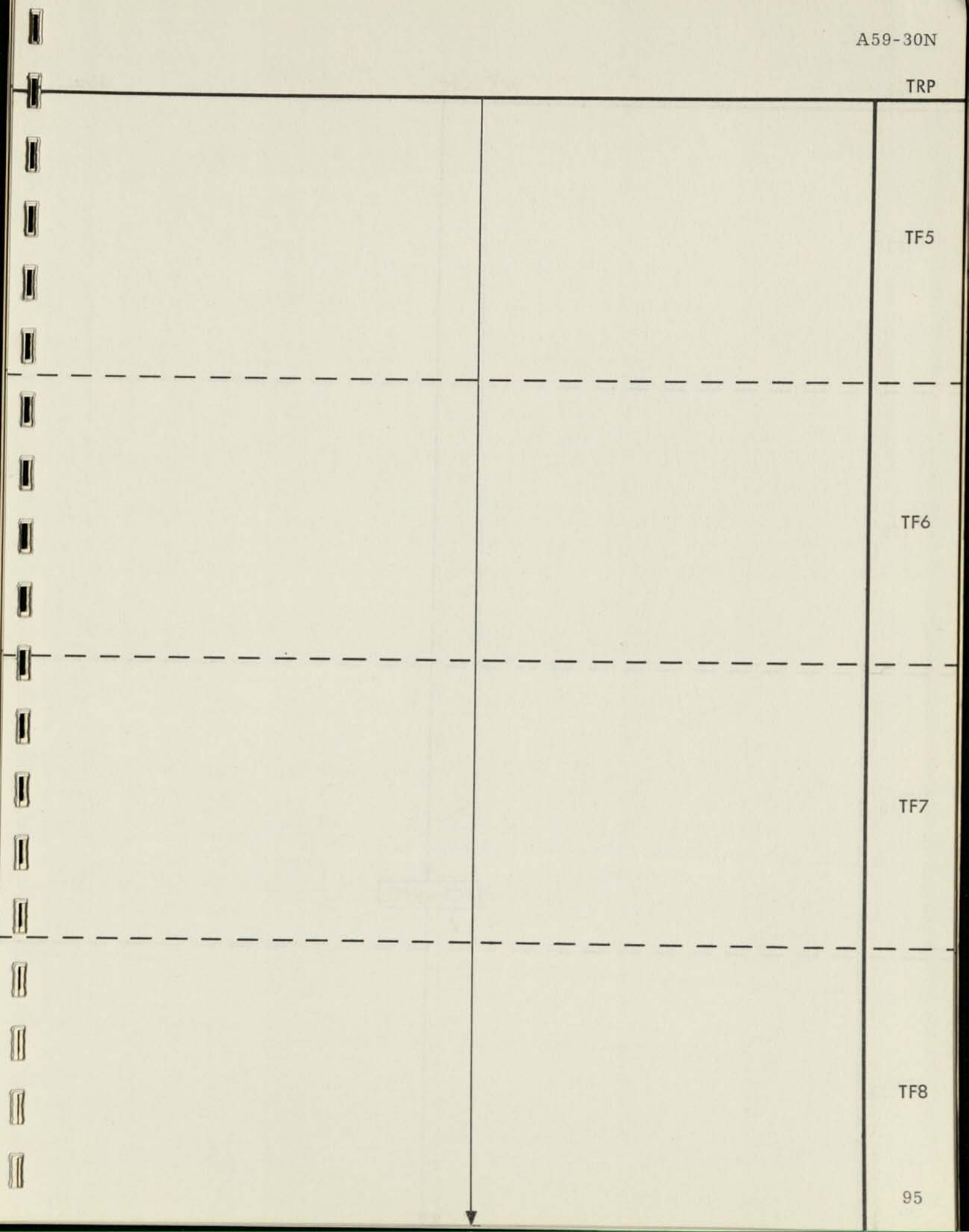
TF1

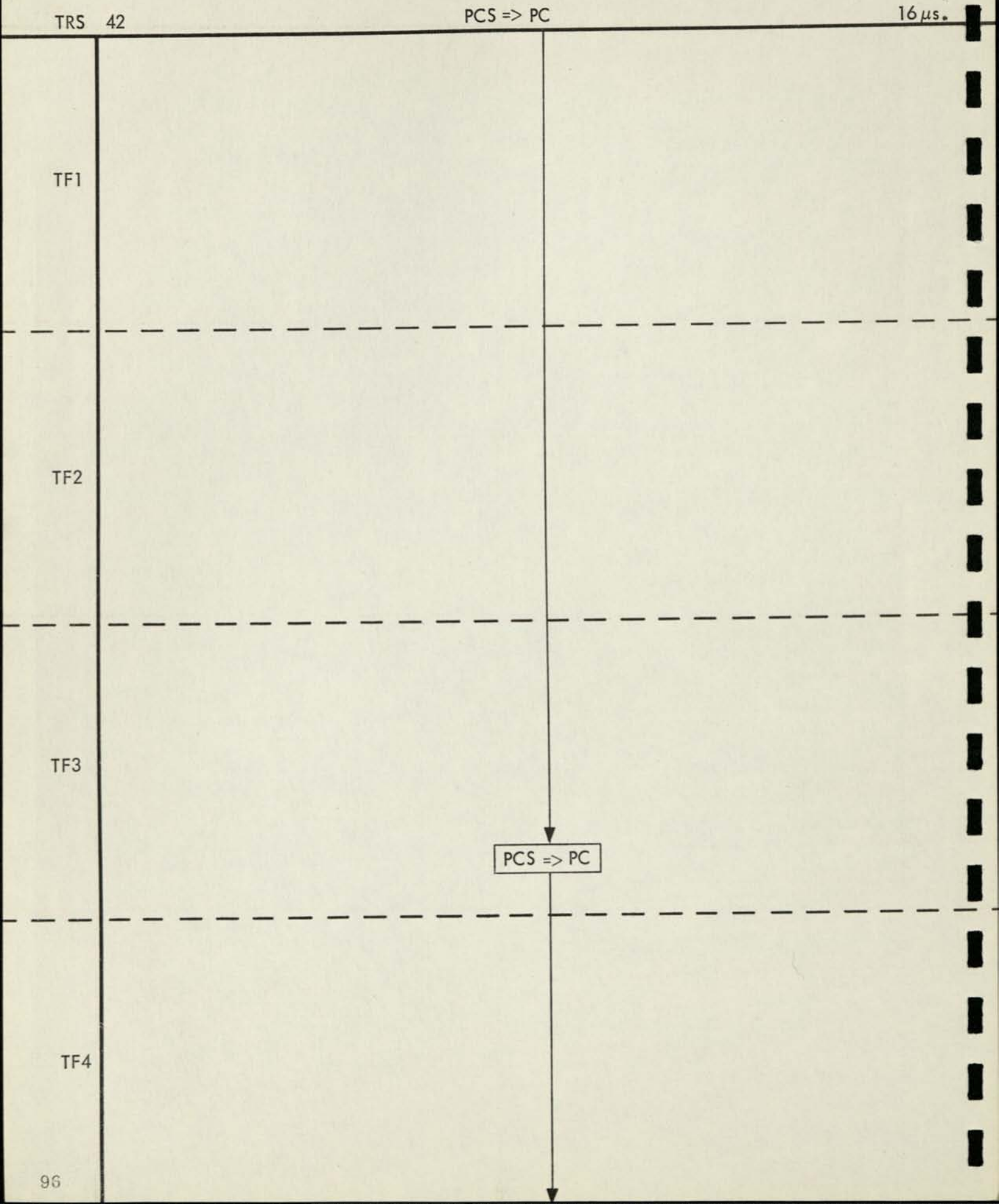
TF2

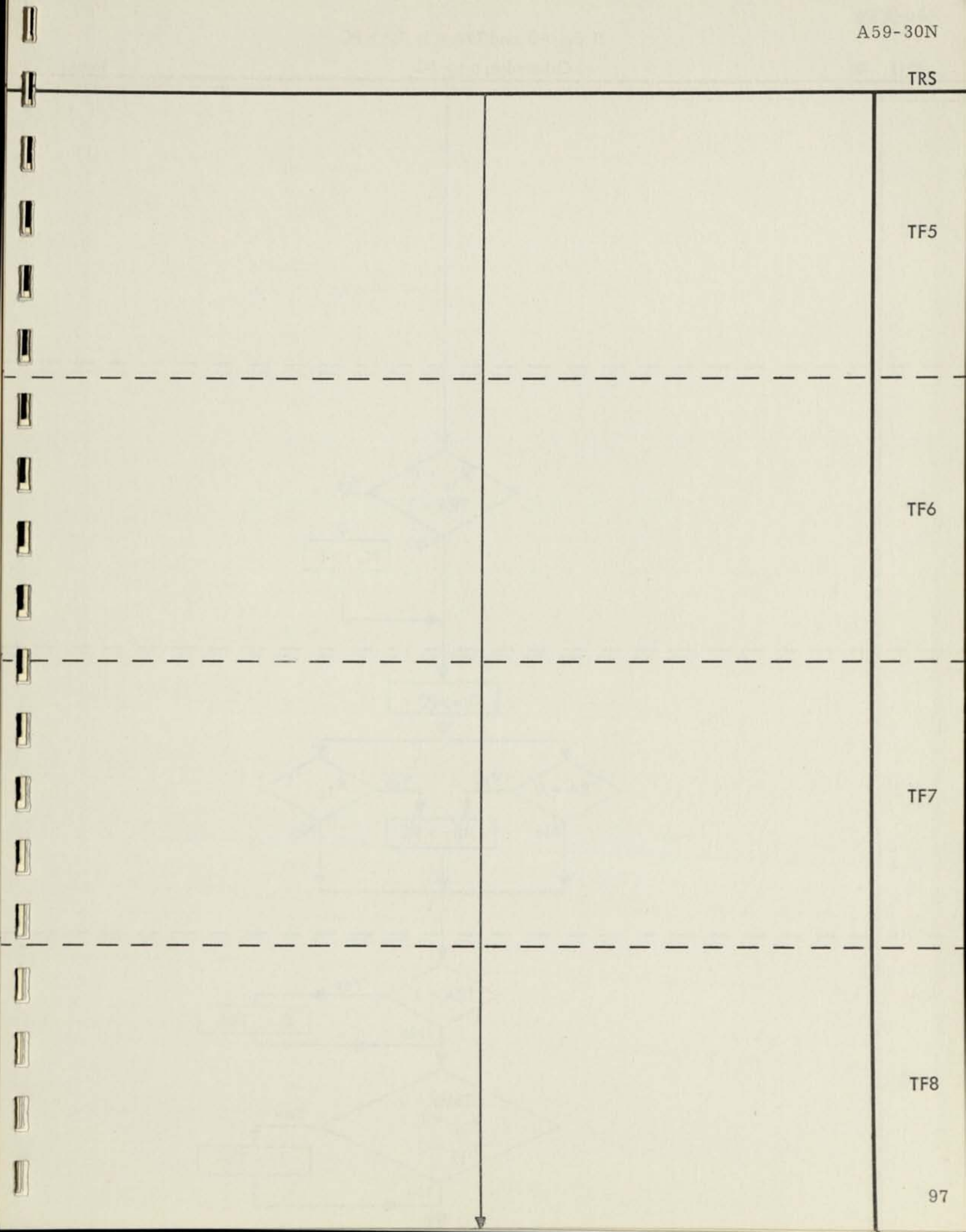
TF3

TF4




TRP
TF5
TF6
TF7
TF8


TRS 42
PCS => PC
16 μs.
TF1
TF2
TF3
TF4
PCS => PC


TRS
TF5
TF6
TF7
TF8

If β_{16} =0 and TRA = 1; 0 => PC
Otherwise; α => PC

TRU 40

16 μs.

TF1

TF2

X_{16} = 0
TRA = 1

Yes

No

PC => B

0 => PC

TF3

TRA = 0

Yes

No

AR => PC

Yes

X_{16} = 1

No

TF4

TRA = 1

Yes

0 => TRA

No

CTMW = 0
X_{16} = 1
X_{17} = 0

Yes

1 => TRA

No

TRU

TF5

TF6

TF7

TF8

TRX 43

If $C(I^{\gamma+1} \neq 0)$, $\alpha \Rightarrow PC$, $I^{\gamma} + \beta \Rightarrow I^{\gamma}$
If $C(I^{\gamma+1} = 0)$, $PC + 1 \Rightarrow PC$

22 μs.

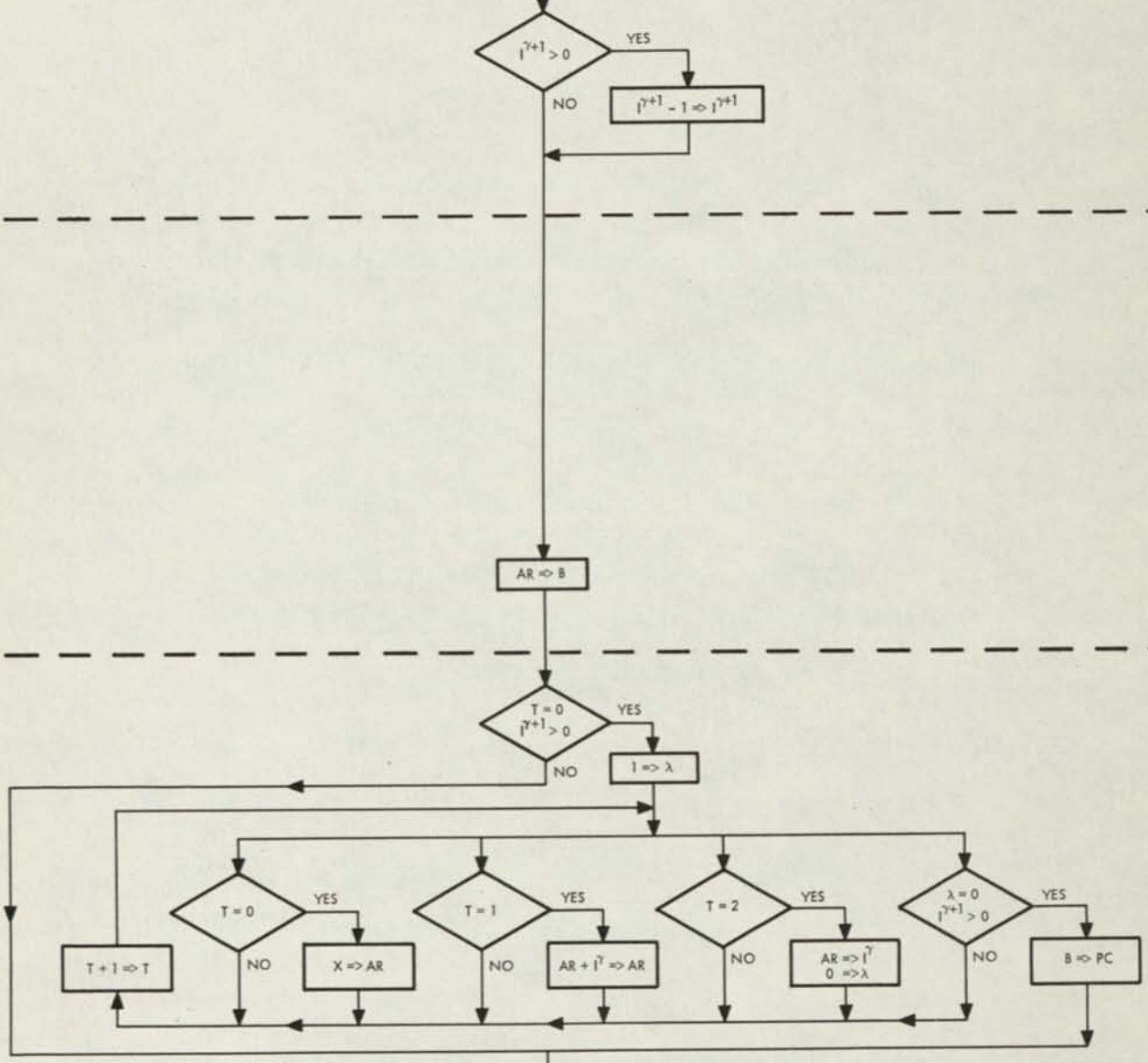
TF1

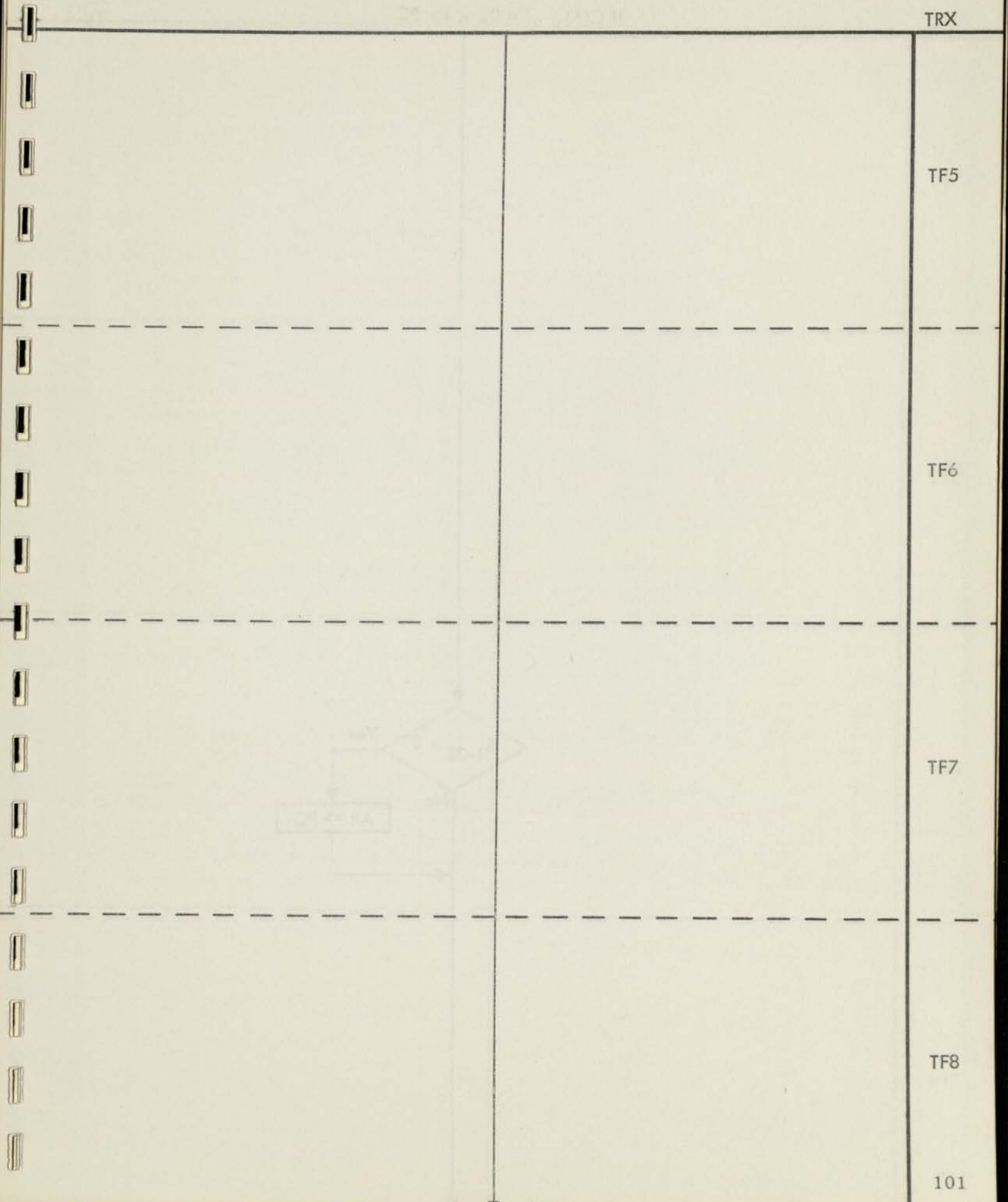
TF2

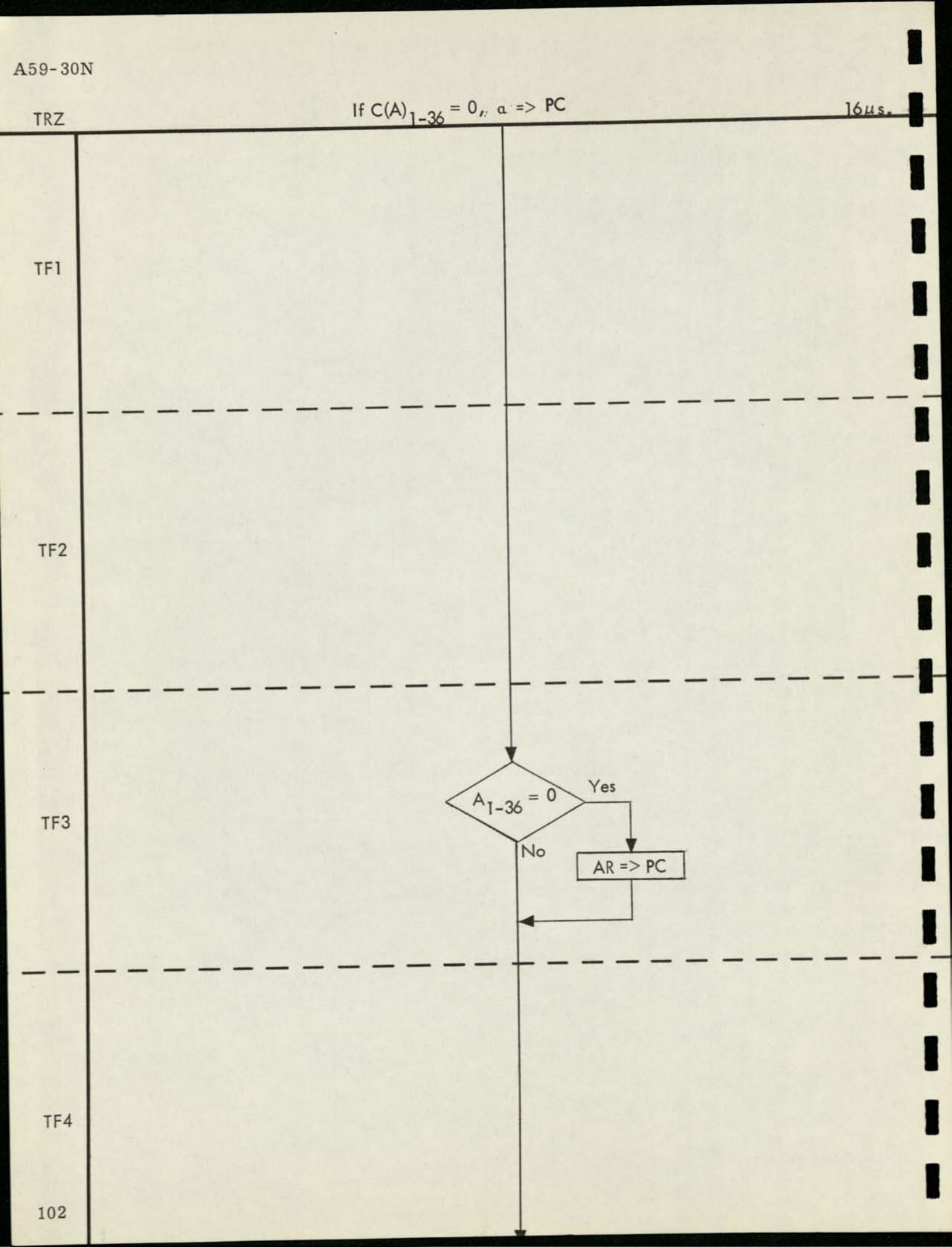
TF3

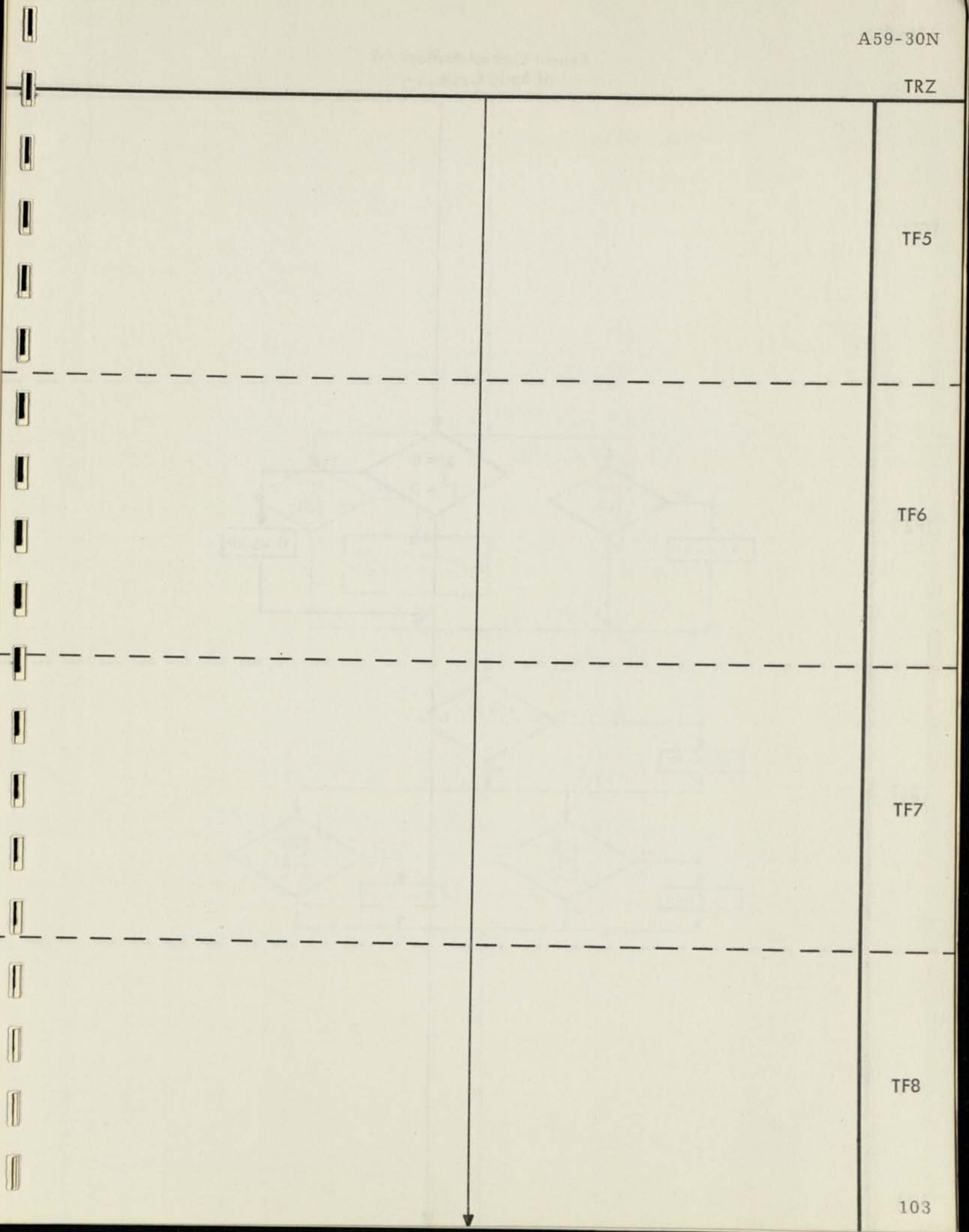
TF4

100




TRX
TF5
TF6
TF7
TF8


TRZ
If C(A)1-36 = 0, a => PC
16μs.
TF1
TF2
TF3
TF4
102
A1-36 = 0
Yes
No
AR => PC


TRZ
TF5
TF6
TF7
TF8

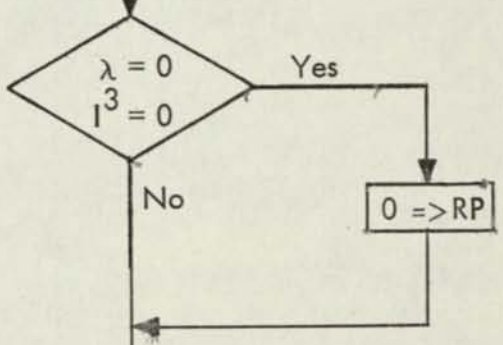
Repeat Control Portion
of Basic Cycle

TF1

TF2

TF3

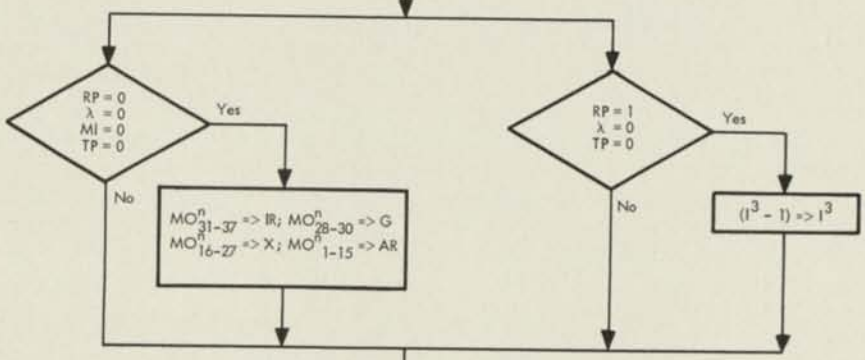
TF4



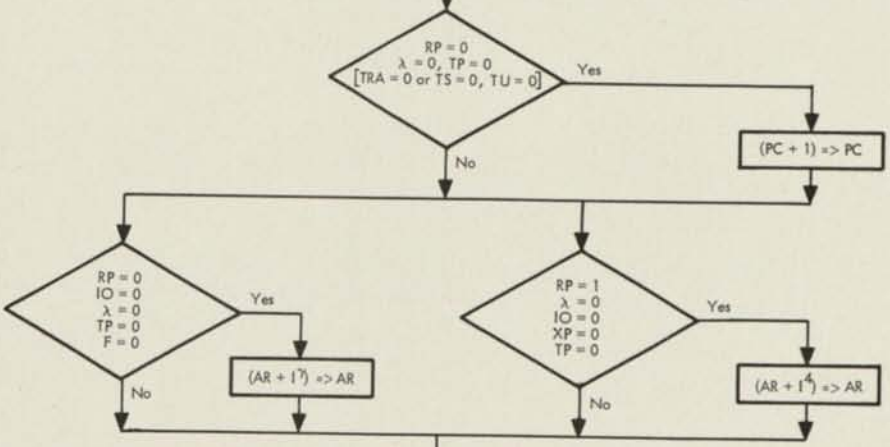
Repeat Control Portion Of Basic Cycle

TF5

TF6



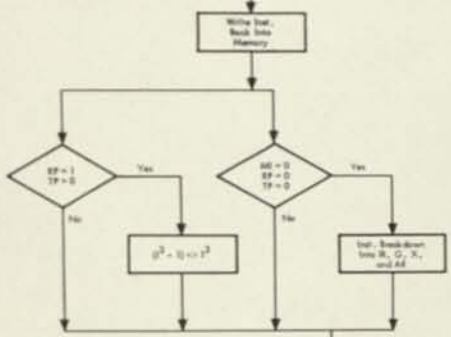
TF7



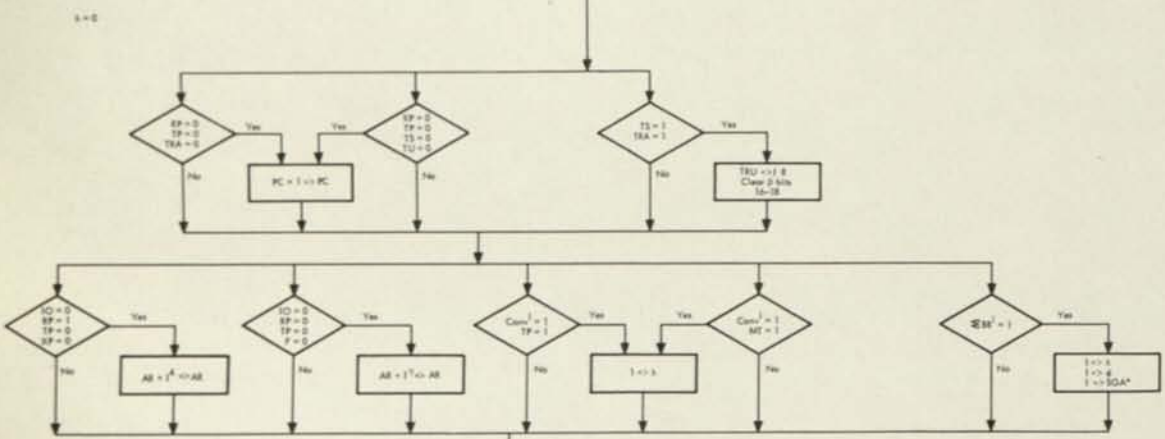
TF8

TF5

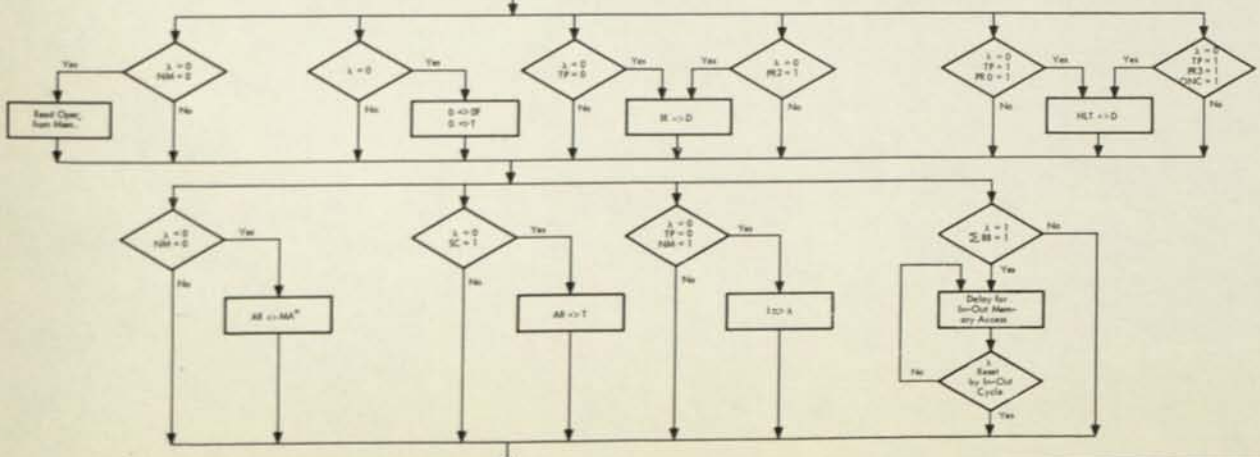
TF6



TF7



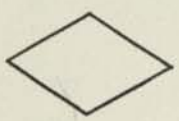
TF8



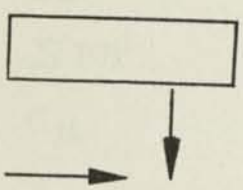
APPENDIX A

FLOW CHART SYMBOLS AND CONVENTIONS

=>	Indicates transfer of information
+	Logical addition (= "or")
( )( )	Logical multiplication (= "and")
'	Logical negation (if A = 1, A' = 0)
Σ	Logical sum, i.e., sum of sequence of "or" s
Y_{i-k}	i-k indicates bits of the register referred to by Y
$\oplus$	Exclusive "or" (A $\oplus$ B = either A or B but not both)



Decision box - Question implied



Action box

Path through chart

Parallel occurrences are indicated by multiple vertical lines off one horizontal line. (Decisions and actions are taking place at the same instant on all Vertical lines.) All decision boxes, which are on the same horizontal level on the flow chart are considered to be happening in parallel.

Timing functions which are left blank indicate No Action. Exit at TF4 assumes continuation to TF5. Exit at TF8 assumes continuation to next instruction.

A' => A implies A'_{sn} => A_{sn}

B' => B does not imply B'_{sn} => B_{sn}

APPENDIX B

FLOW CHART ABBREVIATIONS

A_{sn}	Sign bit of the A register.
A	A register.
AFF^{β}	Addressable flip-flop specified by β of a SEN, SNS, SNR instruction.
AR	Address register (15 bits).
AS	Add-Subtract control flip-flop. When used in the expression $\|A\| + \|B\| + AS \Rightarrow A$, AS is considered a low-order bit, i.e., $AS \times 2^{-36}$.
AU	Arithmetic Unit.
B	B Register.
B_{sn}	Sign bit of B register.
BB^i	The busy bit flip-flop of the i^{th} converter.
$\sum BB^i$	The Boolean expression $(BB^1 + BB^2 + \cdots + BB^n)$.
C_{36}	Carry output from 36^{th} stage of A register. C_{36} will indicate whether or not an overflow condition will occur before the addition of $\|A\| + \|B\| + AS$ is actually performed.
CNV^i	Control flip-flop which is in the "one" state whenever converter i is busy.
$(\sum CNV^i)'$	Boolean expression $(CNV^1 + CNV^2 + \cdots + CNV^n)'$.
$(\prod CNV^i)'$	Boolean expression $\left[(CNV^1)(CNV^2)(\ ) \cdots (CNV^n)\right]'$.
CTMW	Clear trapping mode switch.
D	Decoder register.
F	A control level which is on whenever IR holds the code for one of the following instructions: MOV, LDX, TRL, TRX, ADB, SBB.
G	G register (3 bits).

APPENDIX B CONTINUED

H	Halt flip-flop. When equal to 1, the machine is in halt condition.
I^{γ}	Index register indicated by γ, i.e., I^3 is Index Register 3.
IO	A control level which is on whenever IR holds the code for an IN-OUT instruction.
IR	Instruction register (6 bits).
ISN	Interpret sign flip-flop.
KEW	Inhibit error halt switch on console.
MA^n	Memory address register of memory n. (See Read Pulse and Write Pulse)
MI	Manual Instruction Control level from Console.
MO^n	In-Out register of memory n. (See Read Pulse and Write Pulse)
MOR^n	Either MO^n or REG^n.
NM	A control level which is on whenever IR holds the code for NRM instruction.
OA	Overflow alarm.
OF	Overflow bit of A register. (Not an alarm)
ONC	One-cycle operation flip-flop.
p-level	A level generated at a 500 k.c. rate in the timer.
PC	Program counter.
PRO	A timing level which controls program read-in.
PR2	A timing level which controls program read-in.
Q	Multiplier-quotient register.
Q_{sn}	Q-register sign bit.
Read Pulse	When a read pulse is given, the contents of the memory cell whose address is in MA^n is put into MOR^n.
REG^n	Any addressable register.

APPENDIX B CONTINUED

RP	REPEAT control flip-flop.
SC	A control level which is on whenever IR holds the code for one of the following instructions: SHL, SLL, SHR, SRL, CYS, CYL.
SFF^n	Sense flip-flop n (n = 1 to 16).
SGA*	A pulse which initiates the memory access cycle whenever converter i has access to memory.
SNL	A sense flip-flop or a device whose address is β.
T	T-Counter.
t-level	A level generated at a 500 KC rate in the timer.
TP	Stop flip-flop. Initiates halt condition.
TRA	Trapping mode flip-flop.
TS	A control level which is on whenever IR holds the code for one of the following instructions: TRL, TRN, TRP, TRS, TRU, TRX, TRZ, SEN, SNR, SNS.
TU	A level which is on when the instruction register contains TRU order.
Write Pulse	When a write pulse is given the content of MOR^n is put into the memory cell whose address is given by MA^n.
X	X-register (12 bits).
XP	Timing flip-flop for halt circuit. A control flip-flop which is set to "one" whenever an operation is initiated which is not program read-in.
ZT	Zero test control line of AU. (A level which is on whenever the magnitude of the A register is zero.)
ϕ	Flip-flop which enables in-out memory cycle.
λ	A control flip-flop. Normal sequencing of timing functions is interrupted whenever λ is in the "one" state. (See accompanying writeup.)

Note: Subscripts affixed to register abbreviations indicate particular bits of the register unless otherwise noted; i.e., MOR_{1-15} refers to bits 1 through 15 of MOR. Superscripts affixed to an abbreviation refer to a memory unless otherwise noted; i.e., MO^n refers to the memory output register of memory n.