IBM Customer Engineering

Instruction–Reference

7090 Data Processing System

System Fundamentals
7100 Central Processing Unit
7151 Console Control Unit
7606 Multiplexor

IBM® Customer Engineering Instruction-Reference

7090 Data Processing System

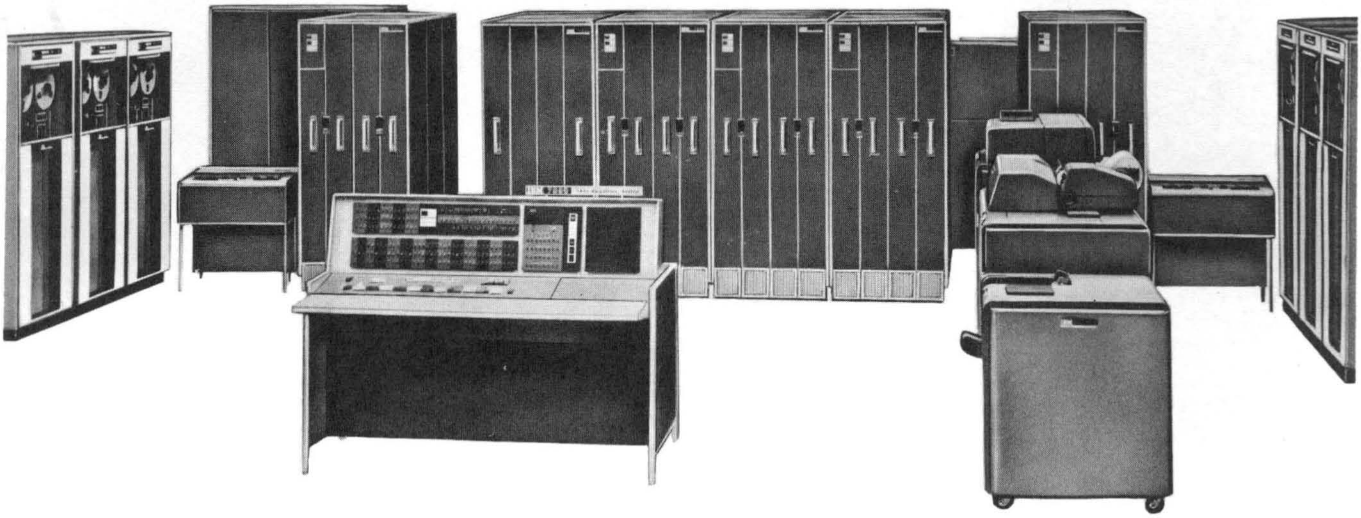
System Fundamentals

7100 Central Processing Unit

7151 Console Control Unit

7606 Multiplexor

CONTENTS



IBM 7090 DATA PROCESSING SYSTEM

MAJOR REVISION--September, 1961
This edition, Form 223-6895-1, obsoletes Form 223-6895
and all earlier editions. This edition has been revised
throughout and should be reviewed completely.

Address comments concerning this manual to:
IBM Corporation
CE Manuals, Department 296
P.O. Box 390
Poughkeepsie, New York

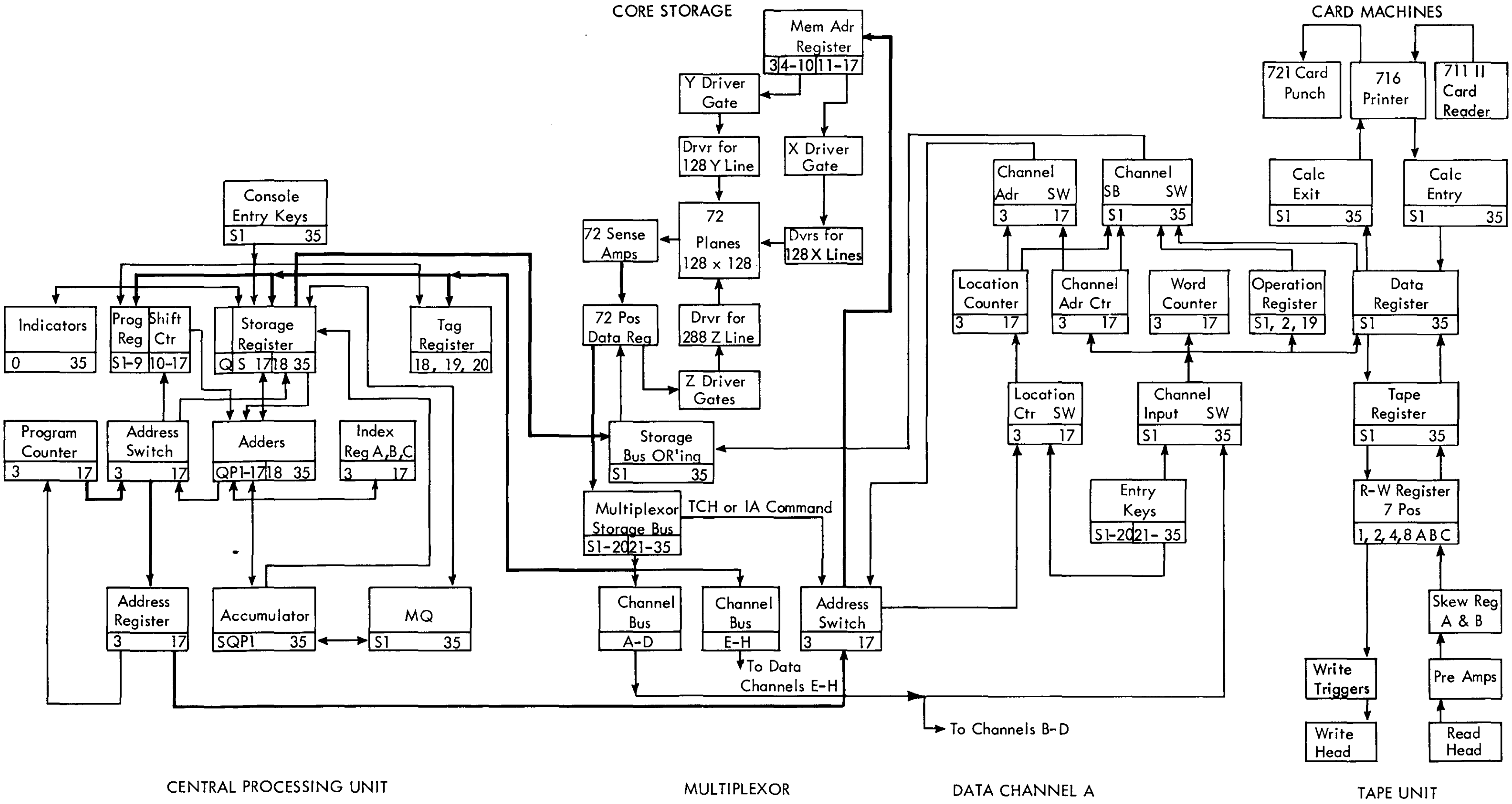


FIGURE 2.5-2. 7090 SYSTEM COMPONENTS -- REGISTERS, SWITCHING FUNCTIONS, AND DATA PATHS

1.0.00 INTRODUCTION TO THE IBM 7090 DATA PROCESSING SYSTEM

The IBM 7090 Data Processing System is a solid-state digital computer that is approximately six times faster than its vacuum tube predecessor, the IBM 709. Along with increased speed, the 7090 is also more versatile. It can perform more than 200 distinct operations.

The 7090 is used in such fields as airplane, rocket, and missile design, atomic research, weather, and missile tracking. Using this computer, calculations in these fields can be done in much more detail and with greater accuracy than ever before. Much of the need for experimental testing is also eliminated; rapid, detailed calculations during design can foresee many troubles that formerly would be indicated only by testing an assembly.

The IBM 7090 is also advantageous and economically feasible for business paper work such as payroll, billing, and sales analysis. Management, with the aid of such a computer, can base business decisions on more up-to-date information than was ever before possible.

1.1.00 GENERAL SYSTEM OPERATION

A computer system may be made to add, subtract, multiply or divide. According to its make-up, a system may also print, read cards, punch cards, read or write magnetic tape, or perform many logic operations. Shifting numbers right or left, setting the algebraic sign, and comparing the size of two numbers are examples of logic operations.

To solve a problem, a sequence of arithmetic and logic operations and the required data are needed. A computer does one operation at a time. When one operation is complete, the next one in sequence begins. Executing operations continues until the sequence has run its course and the final answer to the problem has been computed. The sequence of operations for solving a problem is called a program.

1.2.00 FUNCTIONAL PARTS OF A COMPUTER SYSTEM

The normal make-up of a computer system consists of five functional parts:

1. Input
2. Storage for data and instructions
3. Arithmetic for actual computing
4. Control of all sections
5. Output

These parts work together to accept data and instructions into the system, compute the solution to the problem, and send the solution back out of the system. This all takes place under the direction of the program.

The input section of a computer system accepts information from any outside source and places it in the storage section. This information may come from punched cards, magnetic tape, or manually operated keys. The information may be instructions, data (numbers for arithmetic calculations), or alphabetic characters for printing page headings, comments, and so forth.

The storage unit accepts and stores information that comes into the system through the input section. When any portion of the information in storage is needed, that portion is located and sent out to the section that requested it. All information in the system is at one time or another, in storage; therefore, computer speed depends on storage speed. The storage scheme of most computer systems today is random access--any portion of information can be located directly without searching other locations.

The arithmetic section is the calculating section of the computer system. Here, portions of information, either instructions or data, can be transformed, combined, or altered.

The control section directs the other sections. It tells them what to do and when to do it. Instructions come into the control section from storage. The control section also controls itself in that it keeps account of the instruction it is using and the one that it will use next.

The output section takes calculated information from storage and presents it to an outside user. Commonly used forms of output are: information on magnetic tape, punched cards, printed reports, or indicator lights.

1.3.00 7090 SYSTEM MAKE-UP

The 7090 system includes all five of the sections previously mentioned. Figure 1.3-1 shows the general grouping of these sections in the 7090 system; arrows indicate the general flow of information. Although the functional sections can be neatly separated, in practical application it is more likely that some will be combined and others separated. Input and output are combined with a portion of control in a data channel, and arithmetic with another portion of control in the central processing unit (CPU). Storage is the only functional section that is a separate machine unit. The multiplexor controls the routing of information into and out of storage.

The arrangement shown in Figure 1.3-1 allows input-output to operate somewhat independently, sharing storage with CPU. The highest order of controls is in the CPU, where control is delegated to the lower order controls in the data channel and multiplexor.

A representative 7090 system appears in Figure 1.3-2; the physical grouping of 7090 functions is shown, with machine types. The IBM 7100 Central Processing Unit is contained in two cabinets, or frames, CPU1 and CPU2. The 7151 Console Control Unit provides manual controls for the system operated as a whole. The IBM 7606 Multiplexor and IBM 7302 Core Storage correspond to the same units shown in Figure 1.3-1. The number of machines available for 7090 input-output (I-O) operations is variable, but only seven types are ordinarily used:

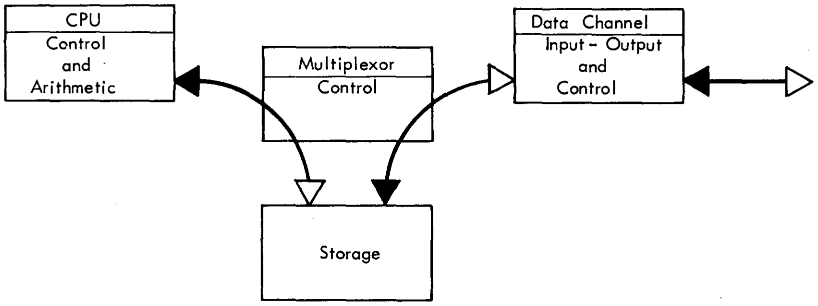


FIGURE 1.3-1. 7090 SYSTEM FUNCTIONAL ARRANGEMENT

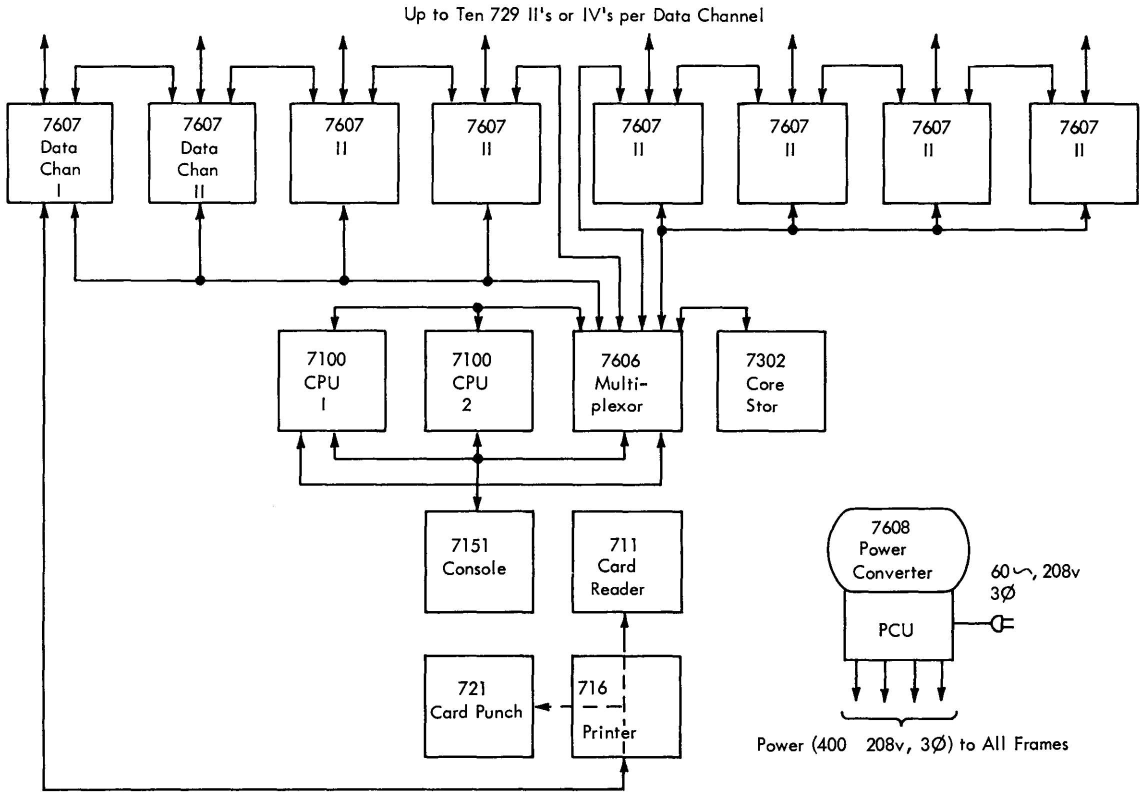
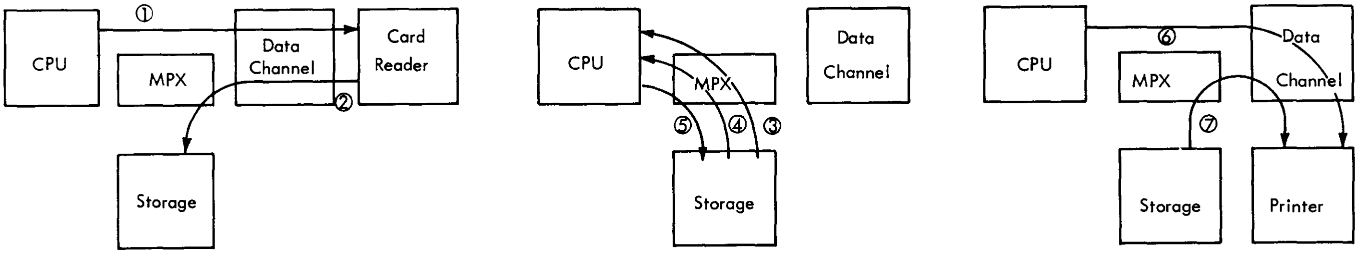


FIGURE 1.3-2. BLOCK REPRESENTATION OF 7090 SYSTEM



Run Card Reader and Read A and B

Add A and B, Store Sum C

Start the Printer and Print C

FIGURE 1.4-1. ADD A AND B, PRINT C

7607 Data Channel Model 1
716 Printer
721 Card Punch
711 Card Reader

7607 Data Channel Model 2
729 II Magnetic Tape Unit
729 IV Magnetic Tape Unit

The reader, punch, and printer can be used only with the 7607 Data Channel Model 1. The 729 II and 729 IV tape units can be intermixed on either model data channel.

The IBM 7100 CPU is the control center of the 7090 system. In addition, all arithmetic and logic circuits are located in the CPU. The CPU receives information from storage, decodes it, and performs the necessary operation. Even though I-O is an independently functioning section, its operation must be initiated from the CPU.

The IBM 729 II and 729 IV Magnetic Tape Units write information on magnetic tape or read information from magnetic tape. The two models perform indentical functions, but the 729 IV moves tape at a higher rate than the 729 II.

The IBM 711 Card Reader reads information from punched cards at 250 cards per minute.

The IBM 716 Printer prints information from core storage at 150 lines per minute. The typewheel echo pulses are available to the 7090 system, where they may or may not be used to check the accuracy of printing.

The IBM 721 Card Punch punches information from core storage at 100 cards per minute.

The IBM 7607 Data Channel Models 1 and 2 control the flow of information between the I-O units and core storage. A 7607 Model 1 can control any combination of ten 729 II and 729 IV tape units and up to one each of reader, punch, and printer. The printer must be present if either a reader or punch are to be used. A 7607 Model 2 can control ten tape units, but neither card machines nor printer. The 7090 system may include up to eight data channels.

Each data channel can be regarded as a subsystem, with its own manual control console and indicator panel. Once a data channel is set in operation by an instruction in the CPU program, it can call in its own instructions (called commands in channel operations). These commands make up what is known as an I-O program. This program controls the operation of the I-O unit that is selected and also provides information to or receives information from that I-O unit. Information received from an I-O unit is placed in core storage, or information is taken from core storage to be supplied to a selected I-O unit.

The CPU handles instructions that select a particular data channel and the I-O device on that channel. The CPU is also responsible for supplying the channel with its first command. This first command can be the first of a series of commands (I-O program) that will sustain the selected channel and device in operation independently of the CPU. When this I-O program has run its course, the selected device stops and the operation is complete.

It is possible for a 7090 with the full eight data channels to have eight I-O programs and the CPU program in operation simultaneously --each independent of the others and all sharing core storage.

The IBM 7302 Core Storage is a fast, random-access storage unit. A unit of information can be read into (or out of) any one of its 32,768 storage locations in 2.18 microseconds. Read-out is spoken of as being nondestructive in that the information remains intact in core storage after read-out. Storage serves both the CPU and data channels. The only restriction is that no two units can be using storage at exactly the same time. If a data channel calls for storage while CPU is using storage, the channel waits until CPU permits storage priority to pass to the channel.

The IBM 7606 Multiplexor is a time sharing and switching device. It provides a path to and from storage for the CPU and data channels. The multiplexor also performs certain anticipatory, or look-ahead functions associated with data channel operations.

The IBM 7151 CPU Console Control Unit provides the means to manually control the system and to display, in indicator lights, the contents of various registers, or any one of the storage locations. Several registers are continually displayed. The console also houses the CE test panel and the marginal voltage check panel.

1.4.00 7090 GENERAL LOGIC

The 7090 system operation can be compared to a 407-514 summary punch operation. The card feed in the 407 is input; storage, arithmetic, and control are in the 407; the 407 print wheels and 514 punch are output. The 514 punches a card only when instructed to do so by the 407 and information to be punched must be in storage and wired to the 514. The 407 waits for a signal from the 514 before proceeding to another operation. Similar requirements for control exist in the 7090 system. Input waits in a ready status until called for by the control section; the control section manipulates information in a predetermined manner and sequence; output operates when called for by the control section. Again--as in the accounting machine, summary punch system--the control section is continually informed of the progress of events in all sections so that another operation can be initiated as soon as the current operation is complete.

An example of information flow and control in the 7090, is the sequence taken by the system in solving the following problem:

Given:	The quantities A and B punched in a card.
Problem:	Add B to A and print the result, C.

(A+B = C, print C)

1. Cause the card reader to run and feed the card.
2. Read the information from the card and place it in storage.
3. Bring A from storage into the arithmetic section.
4. Bring B from storage and add it to A.
5. Place the result, C, in storage.
6. Cause the printer to run.
7. Bring C from storage and send it to the print wheels for printing.

The program for this problem is made up of instructions and data channel commands that control the 7090 to perform the operations outlined. These instructions and commands progress step-by-step and the control section recognizes each in turn when the previous operation is complete.

Because the control section of the system operates at electronic speed and the card reader and printer are relatively slow, it is apparent that much communication is necessary between the mechanical functions and the control section. The control section must wait until A and B have been read from the card before adding them together. The sum C can be developed and stored at electronic speed, but must wait in storage until the printer is ready to receive it for printing.

Figure 1.4-1 shows the progression of the steps in the foregoing program. Not shown in Figure 1.4-1 are the instructions and commands that initiate each of the operations depicted. These are also in storage along with the factors A and B and constitute the stored program. As each instruction ends its operation, the next instruction is called out and initiates its operation. In this manner, the 7090 system progresses through the solution according to the program in storage.

1.4.01 The Stored Program

The program previously described used CPU instructions and data channel commands from storage. These instructions controlled other information going to and from storage. But how did the instructions and commands get into storage? In the beginning, the author of the program decides what instructions and commands to use and in what order to use them. There are several ways to get these instructions and commands into storage without a stored program already in the system. Among these ways are direct manual entry from the operator's keys on the IBM 7151 Console Control Unit*, entry from cards or tape caused by a command manually entered into a data channel from the IBM 7617 Data Channel Console*, and forced automatic operation of a card reader or tape unit caused by depression of a load key on the CPU console.

Whatever the means of entry, both the stored program instructions and the data to be processed must be previously translated into machine language in the form of binary words. A binary word, as used in the 7090, consists of 36 binary positions. The binary number system is explained in Appendix A of IBM 7090 Data Processing System Reference Manual, Form A22-6528. Also explained there are processes for conversion of decimal numbers to binary numbers, and binary numbers to decimal numbers. A third system (called octal) provides a quicker and less cumbersome means of expressing binary numbers.

1.4.02 Exercises

The following equivalent numbers in binary, octal, and decimal provide an opportunity to practice conversion from one number system to another. Do the conversions and check your results against those given. (Decimal and octal fractions are rounded to three places and binary fractions are rounded to nine places.)

* The CPU console (IBM 7151) is a separate machine type, and there is only one on each 7090 system. Each 7607 Data Channel has a manual console of its own (IBM 7617).

Whole Numbers

1.	5_{10}	=	5_8	=	101_2
2.	1125_{10}	=	2145_8	=	10 001 100 101_2
3.	177_8	=	1 111 111_2	=	127_{10}
4.	1325_8	=	1 011 010 101_2	=	725_{10}
5.	7777_8	=	111 111 111 111_2	=	4095_{10}
6.	1 101 010_2	=	152_8	=	106_{10}
7.	1 111 111_2	=	177_8	=	127_{10}
8.	1 010 101_2	=	125_8	=	85_{10}
9.	4096_{10}	=	10000_8	=	1 000 000 000 000_2
10.	3333_{10}	=	6405_8	=	110 100 000 101_2

Fractions

1.	0.7_{10}	=	0.546_8	=	0.101 100 110_2
2.	0.25_{10}	=	0.2_8	=	0.01_2
3.	0.33_{10}	=	0.251_8	=	0.010 101 001_2
4.	0.145_{10}	=	0.112_8	=	0.001 001 01_2
5.	0.915_{10}	=	0.724_8	=	0.111 010 1_2
6.	0.4_8	=	0.1_2	=	0.5_{10}
7.	0.57_8	=	0.101 111_2	=	0.734_{10}
8.	0.715_8	=	0.111 001 101_2	=	0.9_{10}
9.	0.101_2	=	0.5_8	=	0.625_{10}
10.	0.101 010_2	=	0.52_8	=	0.656_{10}

Improper Fractions

1.	17.05_{10}	=	21.031_8	=	10 001.000 011 01_2
2.	40.96_{10}	=	50.753_8	=	101 000.111 101 011_2
3.	17.05_8	=	1 111.000 101_2	=	15.078_{10}
4.	77.77_8	=	111 111 . 111 111_2	=	63.984_{10}
5.	11.11_2	=	3.3_8	=	3.375_{10}
6.	10.01_2	=	2.2_8	=	2.25_{10}

2.0.00 COMPUTER OPERATIONS

A more detailed study of 7090 operations requires additional information about the system. This section of the manual deals with the designation of each word location in storage, the 36-position 7090 binary word, and the fundamental components of system logic. Again, these will be applied to the problem, A+B = C, print C. In this description of the problem, the importance of the central processing unit (CPU) becomes more apparent.

2.1.00 STORAGE WORD DESIGNATION

The storage unit in the 7090 system contains 32,768 locations. Each of these word locations is made up of 36 binary positions. Ordinarily, all 36 positions of any one word are moved to or from storage at one time. This is known as parallel data transmission and allows the moving of a large numeric factor each time that storage reads in or out.

Think of storage as a large double square of pigeon-holes as shown in Figure 2.1-1. Each pigeon-hole has the capacity of a full 36-bit word. When a word is to be stored in or taken from any location, this location must be designated. Obviously, there must be 32,768 of these designations, or addresses, one for each location. Because all operations in the 7090 system are done in binary notation, there are 15 binary positions used for addressing storage locations. Only 15 are required because storage addresses are expressed in octal numbers ranging from 00000 to 77777, rather than from 00001 to 100000. An address of all zeros designates the first word location in storage. Every storage operation, whether directed by the CPU or a data channel, must be addressed.

2.2.00 THE 7090 WORD

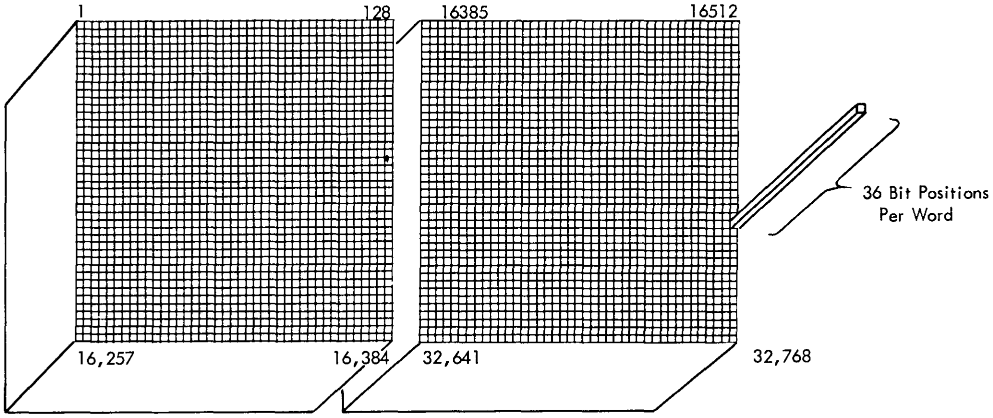
A 7090 word may be a numeric quantity, a CPU instruction, or a data channel command. In all cases, the word is a full 36 positions. The logic format of a word differs according to its use.

2.2.01 Numeric Quantity (Data) Word

Numbers, usually referred to as data, normally appear in true form in 35 of the 36 binary positions in the 7090 word. The remaining position is the algebraic sign of the number (Figure 2.2-1). A binary bit, one, in the sign position is a negative sign. No binary bit, zero, in the sign position of the word is a positive sign. The sign position and the 35 numeric magnitude positions make up the format of the data word.

2.2.02 CPU Instruction Word

The CPU instruction word (Figure 2.2-2) has a different logic format. The CPU instruction word gains its distinctive format because it is called for by CPU at a time



2 Sections, 16,384 Word Locations Per Section

FIGURE 2.1-1. STORAGE LOCATIONS

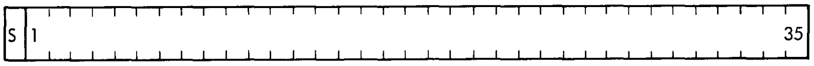


FIGURE 2.2-1. DATA WORD

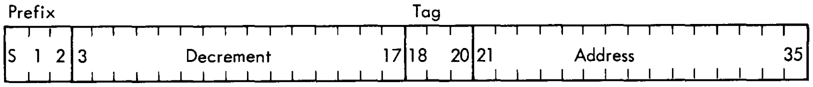


FIGURE 2.2-2. INSTRUCTION WORD

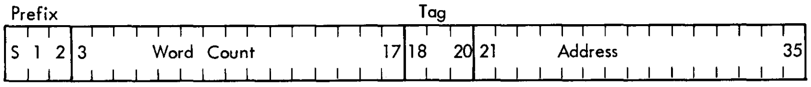


FIGURE 2.2-3. COMMAND WORD

when one operation is about finished and an instruction is needed. The author of the program must provide the proper instruction word in storage and must build his program so that the CPU has the address for that instruction when it is needed.

Each field of the instruction word has its particular significance because the CPU brings the word in as an instruction. The sign position is always a part of the operation code. Along with the sign, either the remainer of the prefix field or the decrement field dictates what operation is to be done. If there is nothing in the remainder of the prefix, the sign and decrement contain the operation code. If either of the two remaining positions of the prefix contains a bit, the prefix contains the entire operation code and the decrement field is used for another purpose.

The address field usually contains the address of a data word in storage. This data word is brought into the CPU as a part of whatever arithmetic or logic function is called for by the operation code. Thus, the instruction not only dictates the operation to be performed, but also specifies the address of the data to be used. In some instances, the address field is a part of the operation code. When this is the case, the address field is not used to address data in storage.

The tag field is a means of causing the CPU to calculate a storage address that is different from the address field of the instruction.

2.2.03 Data Channel Command Word

Similar in format and application to the CPU instruction word, the data channel command word gains its special significance by being called out of storage by the control function in the data channel. This, as in CPU, occurs when one operation is finished and the data channel needs to be directed what to do next. The two major differences are that:

1. The prefix is always the operation code in the data channel.
2. Positions 3-17 are a word counter. Whether the operation is reading or writing, once the command is in channel, the word count becomes one less each time a word is handled. Thus, when the word count becomes zero, the channel must ask for a new command.

Data channel operations always start by unit selection from a CPU instruction. Independent operation continues on the same unit as long as successive commands require it. Once the succession of commands is broken by a command to disconnect, that unit and the channel that was operating have completed their assignment. This is all planned in advance by the author of the program, and commands are stored where the data channel will call for them at the proper time.

2.3.00 FUNDAMENTAL COMPONENTS

The fundamental components described in this section are the operating features (Figure 2.3-1) used in the operation A+B = C, print C. For the most part, these are groups of binary storage positions where information may be placed outside of the storage unit itself. These groups are commonly called registers.

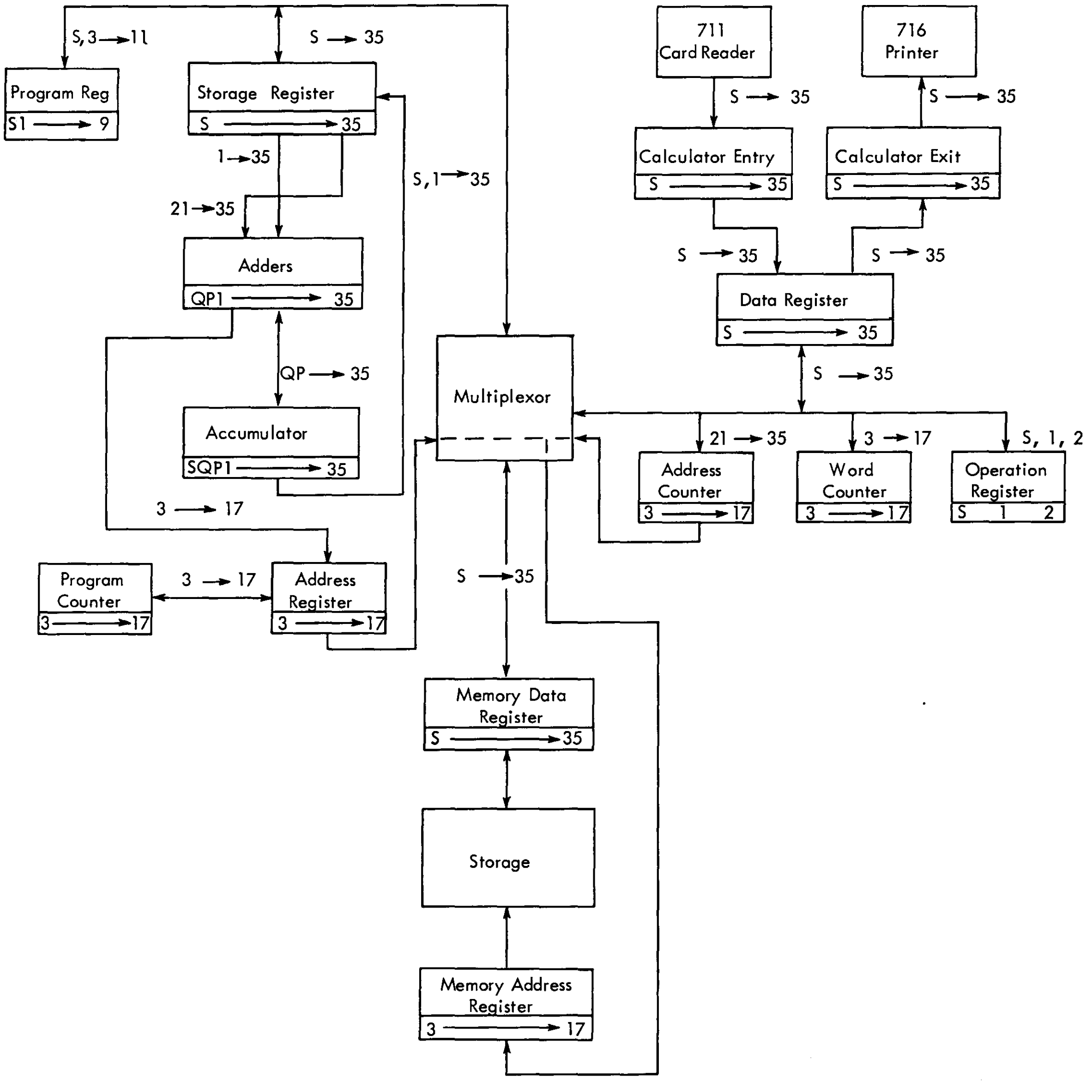


FIGURE 2.3-1. FUNDAMENTAL SYSTEM COMPONENTS USED IN A + B = C, PRINT C

Besides performing a temporary storage function, the registers described also offer the ability to break the 36-position word into lesser groups. You will see that particularly the operation code and the address are extracted from the whole instruction word and placed in separate registers. You will also see the use of the address field of the instruction word as a part of the operation code.

Note that the descriptions in this section are elementary and only as complete as is required for the problem A+B = C, print C. Other registers and switching functions will be described in later sections.

The calculator entry and calculator exit handle information coming from or going to the card machines (reader, punch, or printer). There is no storage here, but information waits at these points until the data register is ready to receive the information from calculator entry or the card punch is ready to receive the information from calculator exit.

The data register is the receiving and distributing point for data passing to and from the data channel. Data words are received here from input units for forwarding to storage. Data words also come from storage to this register for forwarding to output units.

The channel address counter, word counter, and operation register (with the command location counter) are the operating controls in the data channel. Because the location counter is not needed in A+B = C, print C, it has been omitted from Figure 2.3-1.

The channel address counter holds the storage address for the data word being currently handled in the data channel. As a register, this counter receives a beginning address from the address field of a command coming into a channel. As each word is handled, the channel address counter advances one count. Thus, words are stored from channel input in consecutive ascending locations in storage. Likewise, words withdrawn from storage for output come from consecutive ascending locations.

The word counter, as a register, receives a word count from a command coming into a data channel. As a counter, it is reduced by one each time a word is handled. The data channel recognizes zero in the word counter as a signal that the command has been completely executed.

The operation register receives the prefix portion of a command word coming into a data channel. This register is decoded to dictate what operation is to be performed.

The memory address register receives, through the multiplexor, the address of a storage word location to be entered or read out. This address may originate in the CPU or in the data channel.

The memory data register receives a word either from a storage location, or (through the multiplexor) from the CPU or the data channel. The latter case is for the purpose of storing the word at the addressed location; the former is a read-out to the CPU or data channel from the addressed location.

The circuits in the multiplexor are actually switching circuits rather than registers. These switching functions allow either the CPU or the data channel to use storage for purposes of addressing and data transmission. The multiplexor also signals storage whether the operation is to be stored or read out, governed by control from the CPU or data channel.

The storage register accepts a word from storage for use by the CPU, or from the CPU for forwarding to the memory data register. This is true for both instructions and data. The storage register also plays a part in certain arithmetic and logic operations that are not used in A+B = C, print C.

The address register and program counter are the addressing registers of the CPU. The address register, in every CPU operation involving storage, contains the storage address. This address is sent to the memory address register when CPU is using storage.

The program counter keeps account of the progress of the program. This counter normally advances by one after each instruction has been called for. Thus, the normal course of events is to call for instructions from ascending addresses in storage. When one operation is complete, the current reading in the program counter goes to the address register for forwarding to the memory address register. The author of the program is responsible for having the next instruction stored at this address.

The adders and accumulator are the main arithmetic components of the entire system. Although the adders are not registers and have no capacity for holding numbers, they play such a prominent part in almost every CPU operation that they cannot be considered as an ordinary switching circuit.

The adders perform a merging function, having the capacity for operating on an entire word at one time. Numbers from two sources can be added together and carries simultaneously added in. Two full words can be sent to the adders, and the complete answer can be taken almost immediately to a register. In addition, the adders provide the only means of entry to the accumulator and the only direct route from the address field of the storage register to the address register.

The major use of the accumulator is to receive the output of the adders. The name accumulator, although not completely appropriate for a noncounting, nonadding register, is applicable in that this register receives and holds sums from the adders.

The program register receives and holds the operation code of each CPU instruction. Decoding the configuration of binary bits in the positions of this register establishes the controls for the operation to be done. Each time that an operation is ended and a new one is to begin, the old operation code is cleared from the program register and the new code is set in. Thus, the controls for the operation completed are dropped and the controls for the new operation are brought into effect.

2.4.00 A+B = C, PRINT C

The following explanation of 7090's operation in performing A+B = C, print C is more detailed than the general sequence in Section 1.4.1. Review that section before proceeding.

Assume that the necessary instructions and commands are in storage before the program starts. Figure 2.4-1 shows the locations and binary bit configurations of the words in storage when the program begins. The following discussion follows these words sequentially as the program progresses to completion.

Before starting the program, the card with A and B punched in it is placed in the card reader and the card reader start key is depressed. This action causes the card to feed down to the reading brushes and the reader to go into ready status. It is now possible for the CPU to direct a data channel (let us say channel A) to run the card reader and read the card.

The operation of the CPU starts with the depression of the start key on the CPU console. First, the CPU must have an instruction for the very first operation. Storage must have an address for this instruction word to locate it for the CPU. The program counter (PC), initially zeros, contains this address, which goes to the address register (AR) and then to the memory address register (MAR). The word in storage is now located by its address (00000), set into the memory data register (MDR), and sent to the storage register (SR). At the same time that this first instruction goes to the SR, the operation code goes to the program register (PR). Because positions 1 and 2 of the word contain no bits, S(sign) and 3 through 11 constitute the operation code, going to S and 1 through 9 of the PR.

The function of the PC for the first instruction is finished when the PR receives the operation code. It then advances one to 00001 in preparation for calling in the next instruction when the first has completed its operation.

The function of the first instruction is to select the card reader on channel A. Decoding PR (S-9) indicates only a read select, so CPU looks to the SR (23-35) for a channel and unit selection. Here, the reader on channel A is indicated. Channel A acknowledges the selection of its reader if the reader is ready to run, and the CPU is free to go on to another instruction.

The program counter (PC) now reads 00001, so that when the CPU requests a word from storage, the second instruction comes to the SR and PR. This occurs in the same manner described for the first instruction, and the PC advances to 00002.

Decoding the new bit configuration in PR (S-9) at this time indicates that the channel registers are to be reset and a command is to be received. The CPU has the address of the command in SR (21-35), so this address goes through adders 3-17 to the AR and MAR. At the same time, PR decoding in the CPU causes the word at this address, 00012 (Figure 2.4-1), to go to channel A rather than to the CPU. Channel A receives the operation code of the word (S, 1, 2) in its operation register, the word count (3-17) in its word counter (WC), and the first storage address (21-35) in its channel address counter (CAC).

At this point in the program, all operations have occurred at electronic speed within a few microseconds. Channel A has its reader selected for a reading operation, a disconnect-type operation code, a word count of two, and a beginning address where the first of the two words is to be stored. The reader feeds and reads the card, and channel A stores the first word at 00014 according to the CAC. The WC decreases to 1 and the

	S	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	
00000-	0	0	0	1	1	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	1	0	0	0	1	Read Select Channel A Card Reader
00001-	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	Reset Channel A Registers and Load a Command From Location 00012
00002-	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	While Channel A is In Operation, Take the Next Instruction From Location 00002
00003-	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	Clear the AC and Enter the Contents of Location 00014 (A)
00004-	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	Add to the Contents of the AC From Location 00015 (B)
00005-	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	Store the Contents of the AC at Location 00011 (C)
00006-	0	0	0	1	1	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	0	0	1	0	Write Select Channel A Printer
00007-	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	Reset Channel A Registers and Load a Command From Location 00013
00010-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Halt CPU Operation. When Started Again, Take the Next Instruction From Location 00000
00011-																																					This Location is Reserved for C.
00012-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	Run the Selected I/O Unit for Only Two Words, Starting at Location 00014 (Read A and B)
00013-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	Run the Selected I/O Unit for Only One Word, Located at 00011 (Print C)

FIGURE 2.4-1. PROGRAM WORDS FOR A + B = C, PRINT C

CAC steps to 00015 when the first word, factor A, has been sent to storage. When the second word, factor B, is read, the data channel sends it to storage location 00015 as indicated by the CAC. The word count now goes to zero as the WC is reduced again. With a disconnect operation code, the work of the data channel is done and the CPU is notified of this.

What has CPU been doing during the relatively long period of mechanical operation of the card reader? It cannot be allowed to go on until factors A and B have been read in and stored.

When channel A acknowledged receipt of its command, the CPU sent to storage for the word at 00002. The PC to AR to MAR caused the word at 00002 to come from storage to SR and PR. Decoding the bits in PR sent the CPU to the address in SR (21-35) for a new instruction and indicated that nothing more was to be done except that this address was also to be put in the PC. This operation was indicated to be conditional, however, and was to be done only if channel A was in operation. Channel A was in operation, so the following events occurred while channel A read in factors A and B and stored them.

1. PC advanced to 00003.
2. SR (21-35), through adders 3-17, went to AR and then to MAR, each time being 00002.
3. AR contents of 00002 replaced PC contents of 00003.
4. Storage sent back the same word originally called for as the third instruction.
5. The same operation code was decoded and the entire operation repeated, so long as channel A remained in operation.

Eventually, channel A completes reading in and storing factors A and B. This time, the CPU does not send for 00002 again, but allows the PC to retain 00003 and sends this to AR and MAR. Now the word at 00003 comes to SR and PR. Decoding PR causes the accumulator to be cleared. The address field, SR (21-35), of the instruction goes through adders 3-17 to AR and then to MAR. This time, the CPU is not finished with its operation and is not looking for a new instruction. Instead, factor A comes from storage location 00014 to the SR. No portion of this word enters the PR and the clear and add operation code is retained there. Factor A goes from the SR through the adders to the accumulator (AC). There is no offsetting of the address field this time, because the SR contains a data word rather than an instruction word.

With the placement of factor A in the accumulator, the clear and add operation is complete, so the CPU sends to storage for a new instruction. The PC has advanced to 00004, and this address through AR and MAR causes storage to send the contents of 00004 to SR and PR. Decoding the new operation code in the PR, the CPU sends SR (21-35) through adders 3-17 to the AR and MAR. This address of 00015 is the address of factor B, which is to be added to the contents of the AC. The PC advances to 00005. Storage sends factor B to the SR in the same manner that it sent factor A previously. Here, again, the CPU has not yet completed its operation, and the word is treated as a data word. From the SR, factor B goes to the adders. At the same time, factor A goes to the adders from the AC, due to the operation code in the PR. The sum of factors A and B out of the adders goes to the AC, replacing factor A there. The CPU has now completed the add instruction, using the address of factor B.

The address of 00005 in the PC, through the AR and MAR, brings the next instruction to the SR and PR. Decoding the PR, the CPU operation is store and the address-- SR (21-35)--at which to store is 00011. The PC advances to 00006. Sum C in the AC, because of decoding store, goes to the SR after SR (21-35) goes to the AR and MAR. Then storage accepts SR to the memory data register (MDR) and stores it at address 00011. The operation is now complete and sum C is stored for printing.

The only remaining task for the CPU is to tell channel A to start the printer and to give channel A the address of C. The next instruction, coming from 00006 as dictated by the PC, decodes as write select. The PC advances to 00007. Again, as in read select, the channel and unit selection is in the address field of the instruction. The bit configuration there indicates the printer on channel A. Channel A acknowledges the selection if the printer is ready. This frees the CPU to ask for another instruction, this time from 00007. The instruction, decoded in the PR, directs the resetting of channel A registers and the loading of a command from the address found in the address field of the instruction. In this case, SR (21-35) has the address of the command. The address goes to adders 3-17, then to the AR and MAR. Storage delivers the contents of this address, 00013, to channel A instead of the CPU because of controls set up by the PR.

By this time, the PC had advanced to 00010, so when channel A has received its command, the CPU receives the contents of 00010 as an instruction. Decoding a PR of zeros causes the CPU to stop. The operation code of zeros is called halt and transfer. If the CPU should be restarted, the address field of 00010, which is all zeros, would go to the PC and the program would restart at 00000.

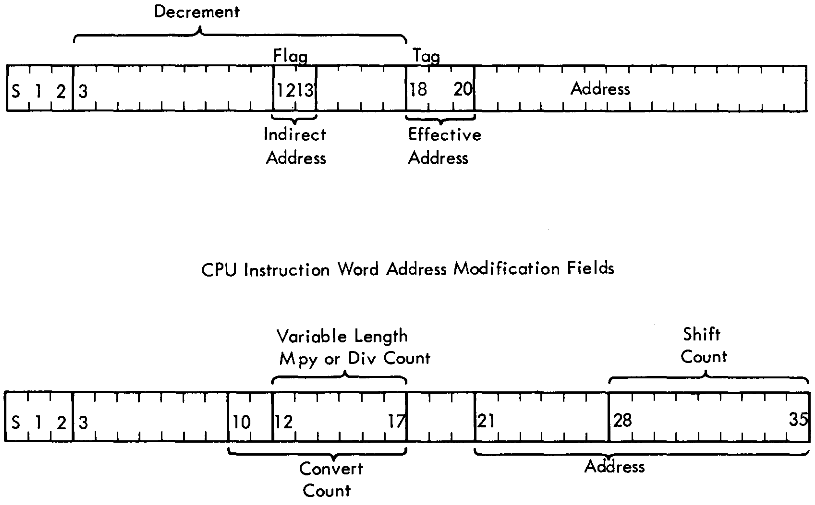
The role of the CPU in A+B = C, print C is now ended. However, channel A still has to print C. The printer is started and channel has a disconnect command in the operation register, a word count of one, and the address of C--00011--in the CAC. These all came from storage location 00013 as the CPU executed the instruction at 00007. Channel sends 00011 from the CAC to the MAR. Sum C comes back to the DR, and the WC steps to zero. As soon as the printer is ready to receive C, the word goes from the DR to the calculator exit and is printed. Channel A is also finished now, and A+B = C, print C has been accomplished.

If you have followed the program for A+B = C, print C, carefully, referring to Figure 2.3-1, you should now have a general idea of the operation of the 7090 system.

2.5.00 OTHER COMPONENTS, INSTRUCTIONS, AND COMMANDS

The preceding explanation is simplified, and many components and functions are omitted. Before continuing with a study of this manual, study the instructions and commands described in IBM 7090 Data Processing System Reference Manual, Form A22-6528.

Figure 2.5-1 shows two other possible configurations of the CPU instruction word: Figure 2.5-2 is a more complete representation of system components. These figures will prove helpful in studying the 7090 Reference Manual.



CPU Instruction Word Address Modification Fields

CPU Instruction Word Count Fields

FIGURE 2.5-1. CPU INSTRUCTION WORD VARIATIONS

Following the study of the instructions and commands used in the 7090 system, study the operation of the individual current switching component circuits that make up the system. These are explained in Transistor Components Circuits, IBM Customer Engineering Manual of Instruction, Form 223-6889.

These books, together with the information previously presented in this manual, comprise the information needed to continue the study of this manual beyond this point.

The following information in this manual, although generally in outline form, is quite specific and requires the use of 7090 systems diagrams to provide maximum benefit.

3.0.00 CPU INTERNAL FUNCTIONS

3.1.00 FUNCTIONAL COMPONENTS

Figure 3.1-1 shows the basic registers and switching units of the 7100 CPU. Because the registers and switching units have different requirements, they are made up of various types of basic components which are explained below.

Shift Cell

A shift cell allows simultaneous read-in and read-out operations. Shift cells are used to shift data to adjacent positions within a register, or to read out old data and read in new data simultaneously.

The basic shift cell is shown in Figure 3.1-2A. Two triggers have multiple inputs to turn on the selected register position. The control lines "set" and "hold" are generated by the CP set pulse. Note that "hold" falls prior to the rise of "set" and rises again prior to the fall of the set pulse. Refer to Figure 3.1-2B.

The numbers on the timing chart in Figure 3.1-2B correspond to the test points of Figure 3.1-2A. With the input line active, the signal at test point 2 becomes active.

Test point 2 conditions the +TO with its output, test point 3, conditioning one leg of the +TA. The set pulse conditions the other leg of the +TA. The output of this trigger, test point 4, remains active for the duration of the set pulse. Test point 4 conditions the lower +TO, so that test point 5 becomes active and conditions one leg of the +TA. Because "hold" becomes active prior to the fall of the "set," the lower trigger latches and retains the information.

The registers that contain shift cells are:

1. Storage register (SR)
2. Accumulator register (AC)
3. Multiplier-quotient register (MQ)
4. Index register (XR)

Triggers

A trigger acts as an information storage unit. Information can be read into a trigger, and retained there until the trigger is reset.

The registers that contain triggers are:

1. Program register (PR)
2. Address register (AR)
3. Sense indicators (SI) (This register actually uses two triggers for each position to permit complement operations; it should not be confused with shift cell registers.)

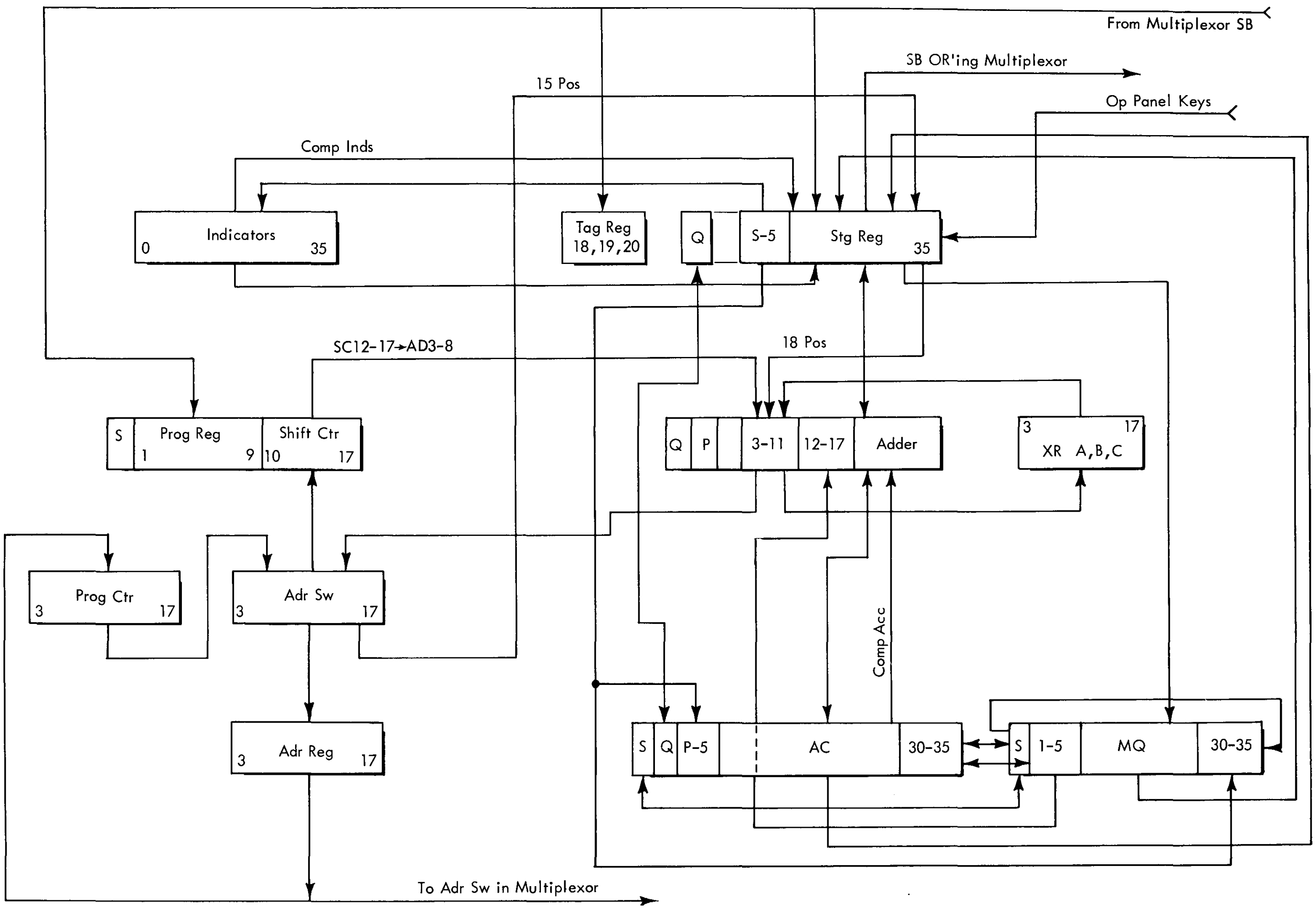


FIGURE 3.1-1. 7090 CENTRAL PROCESSING UNIT

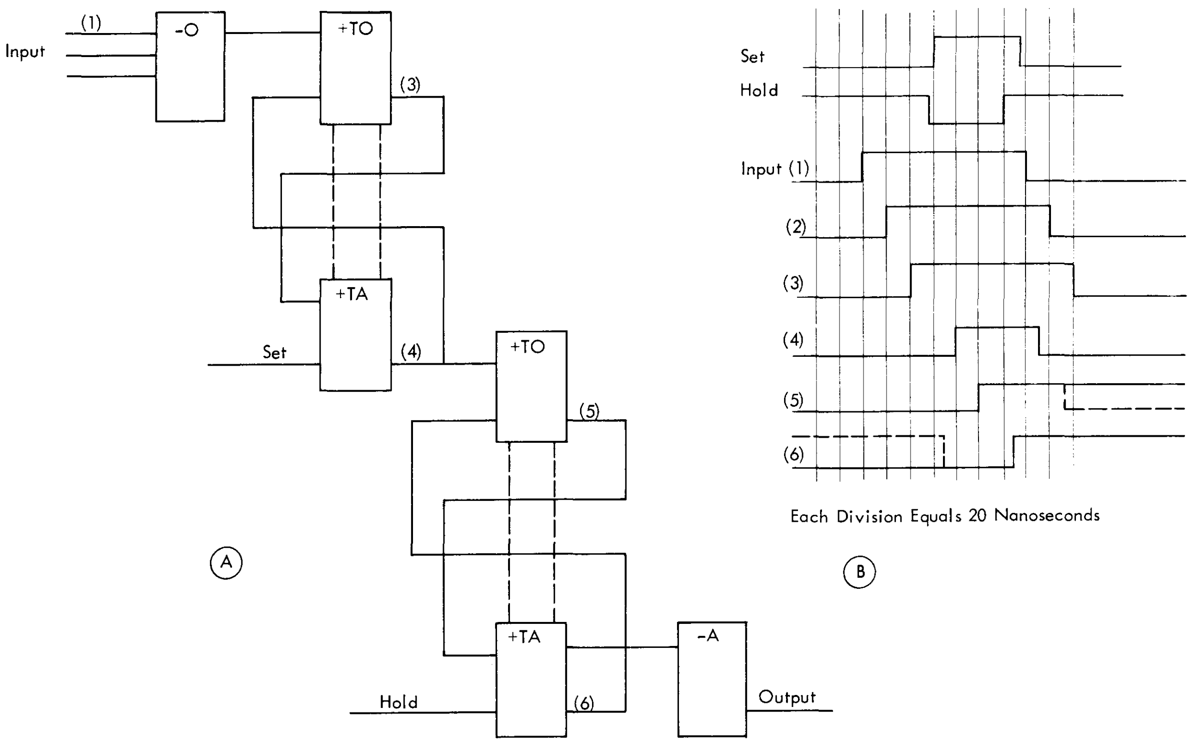


FIGURE 3.1-2. SHIFT CELL

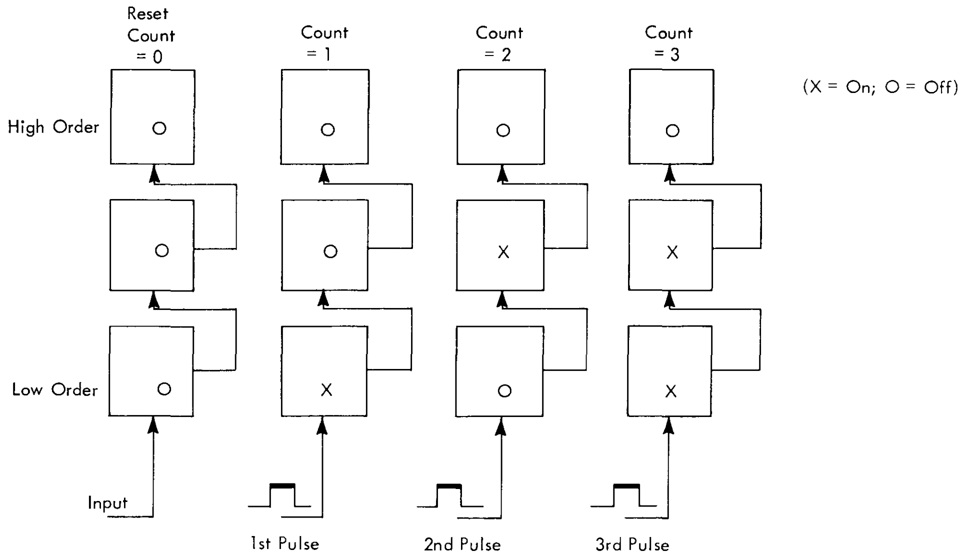


FIGURE 3.1-3. COUNT-UP COUNTER

Switching Units

Switching circuits are comprised of AND and OR logic blocks. They cannot hold information, and their output is conditioned only as long as the input is conditioned.

The switching units are:

1. Address switches (AS)
2. Adders (AD) (See Section 3.1.11 for arithmetic adding applications.)

Binary Counters

A binary counter is a group of binary trigger positions designed to count the number of input pulses to the low-order position. There are two types of binary counters:

1. Count-Up Counter: Each position, as it goes off, reverses the status (on-off) of the next higher position (Figure 3.1-3).
2. Count-Down Counter: Each position, as it goes on, reverses the status of the next higher position (Figure 3.1-4).

Each counter position has three inputs (Figure 3.1-5):

1. Set Pulse: Presets counter to some initial status.
2. Step Pulse: Advances (count up) or decreases (count down) a preset number.
3. Reset Line: Resets counter positions to all zeros.

Speed-Up Counting

In a counter position for a large number, normal circuitry (where one position feeds the next higher position) can cause a considerable delay. An example of this delay is shown in Figure 3.1-6. When each position goes off, it feeds the next higher position and reverses its status. Because of the high speed of the system, this delay must be overcome to insure reliable operation. The following system is used to reduce the delay.

The outputs of several positions are AND'ed, and if all these positions are impulsed, their AND changes the next higher position, as shown in Figure 3.1-7. The stepping of a low-order counter position may step all positions above it. Therefore, circuits are attached to the counter unit to speed up counter operation. Figures 3.1-7A and 3.1-7B show the logic of the speed-up circuits used with the shift counter and program counter.

7090 Counters

1. The program counter (PC) (instruction counter) is a count-up type (Systems, Volume 5, 03.00.05.1-3.05.09.1). See Figure 3.1-7A.

2. The shift counter (SC) is a count-down type (Systems, Volume 5, 03.04.14.1-.03.04.19.1). See Figure 3.1-7B.

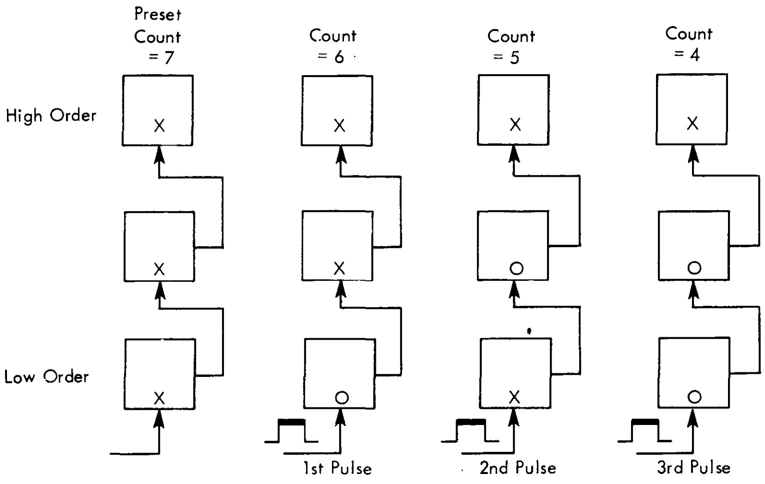


FIGURE 3.1-4. COUNT DOWN COUNTER

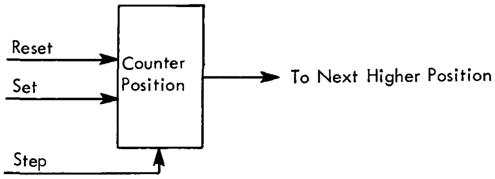


FIGURE 3.1-5. BINARY COUNTER INPUTS

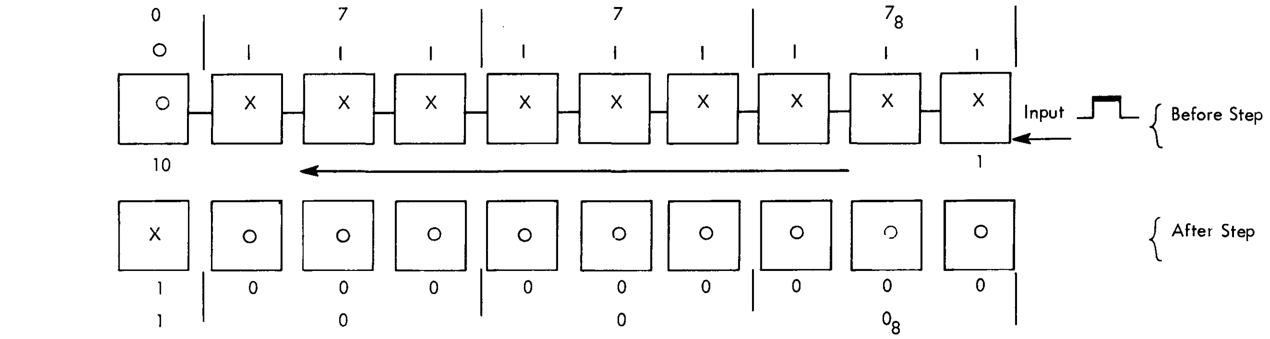


FIGURE 3.1-6. NORMAL COUNTER RIPPLE DELAY

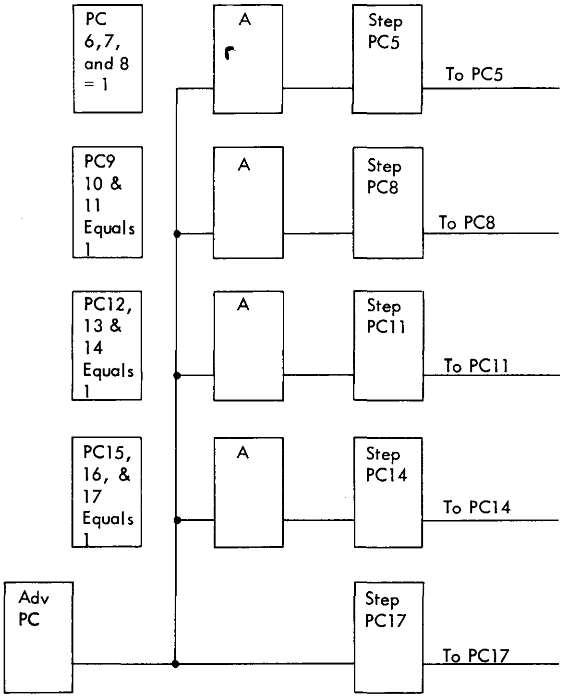


FIGURE 3.1-7A. PROGRAM COUNTER STEP GENERATOR

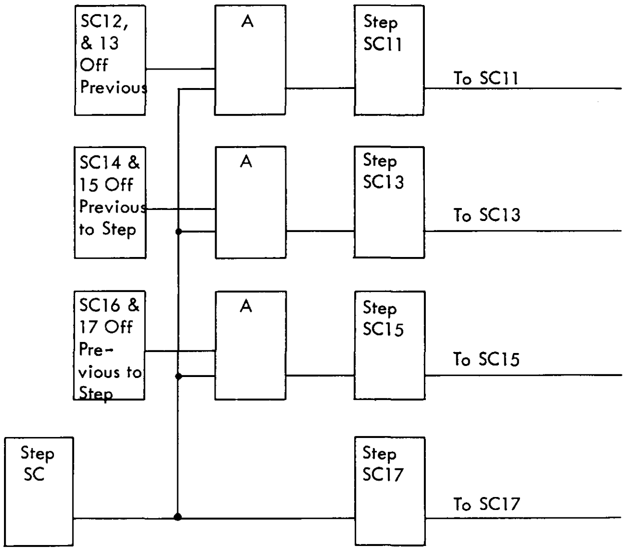


FIGURE 3.1-7B. SHIFT COUNTER STEP CONTROL

3.1.01 Storage Register (SR) (Systems 2.01.00.1-2.01.37.1)

The storage register is a 37-position shift cell register and the register positions are labeled (S), (1-35), and (Q). All instructions and data coming into or leaving the CPU go through the storage register, positions (S), (1-35). Storage register positions (S), (1-35), and (Q) are also used for holding factors and partial results during the execution of many instructions.

3.1.02 Accumulator Register (AC) (Systems 2.03.00.1-2.03.37.1)

The accumulator is a 38-position shift cell register used in nearly every arithmetic operation. The register positions are labeled (S), (Q), (P), (1-35). Positions (S), (1-35) accommodate the word in standard operations. Positions (Q) and (P) are used as overflow positions because the sum of two 35-position numbers can be greater than 35 positions. Postion P also holds the S bit of a word during logical operations. As the name implies, the accumulator is the unit into which results accumulate; it does not perform the addition.

3.1.03 Multiplier-Quotient Register (MQ) (Systems 2.04.00.1-2.04.35.1)

The multiplier-quotient register is a 36-position shift cell register with the positions labeled (S), (1-35). This register has several uses. During a multiply operation it holds the multiplier; after the multiply operation it holds the least significant half of the product. During a divide operation, it holds the least significant half of the dividend; after the divide operation it holds the quotient. In several floating point operations, the MQ holds the least significant 35 bits of the result.

3.1.04 Sense Indicator Register (SI) (Systems 2.06.00.1-2.06.35.1)

The sense indicator register is a 36-position trigger register, labeled (0-35). Each position of the register contains two triggers. One of the triggers is used to retain information in the register, and the other trigger is used as a remembering device during an invert operation. The sense indicator register is controlled completely by the program and is not used by the computer as a part of its arithmetic operations. It can be used as a set of switches which are set and tested by the program to check the progress or direction of the program. The SI register may also be used to store words or parts of words temporarily and, in this way, it is useful for altering and testing words. The SI register inputs and outputs go directly to the storage register.

3.1.05 Index Registers (XR) (Systems 2.07.01.1-2.07.15.1)

There are three 15-position shift cell registers called index registers A, B, and C. The register positions are labeled (3-17). The three registers are identical in operation and are used for instruction address modification. They are activated by the tag positions of an instruction and can be used singly or in parallel or combinations. They modify an address by adding the complement of their contents to the address; in effect, the address is reduced by the contents of the index register.

There are many instructions which operate on the index registers, thereby making these registers useful programming tools for counting, word alteration, program loop control and so forth.

3.1.06 Program Register (PR) (Systems 3.04.00.1-3.04.03.1)

The program register is a ten-position trigger register whose positions are labeled (S), (1-9). The program register receives the operation code of the instruction directly from the storage bus and holds the operation code throughout the execution of the instruction. The output of the program register is decoded to initiate and control the CPU so it will execute the instruction. Positions (1-5) contain the primary operation part of the operation code; positions (6-9) contain the secondary part.

3.1.07 Shift Counter (SC) (Systems 3.04.14.1-3.04.19.1)

The shift counter (SC) is an eight-position, count-down counter. The SC counts the number of shifts taken during a shift operation and is also used as part of the decoder for operations that have a primary operation code of 76. See Section 5.00.00.

Figure 3.1-8B is a reproduction of an ALD page without the reset lines, but with special terms applied to blocks or combinations of blocks.

Because the SC is a step-down counter, a number must be set in the counter at the start of the operation. The address switches are gated to set the SC. Considering a shift of seven, the "+P AS to SC" line to positions 15, 16, and 17 will be active. This condition causes one trigger (A or B) in each position, to latch on, thereby indicating a 1 in the three positions.

With "AS 17 to SC 17" active, the +OR circuits at 3F and 3H are conditioned, indicating that trigger B will latch on and trigger A will be in a preset condition. The in-phase output of the +OR at 3F conditions the A trigger of position 16, along with "AS 16 to SC 16." The out-of-phase line of the +OR at 3F deconditions trigger B. Now positions 16 and 17 have a bit. Position 15 has both triggers preset by "AS 15 to SC 15." Because the SC is not being stepped at this time, trigger B of position 15 will latch on.

The SC is divided into groups of two for lookahead operations. The outputs of each group are AND'ed together to determine if the higher order positions are to be stepped. The high-order position of each group have a line, "-N SC set gate," to prevent an error if a number other than seven is set in the SC. Suppose a 1 is in the AS; this sets trigger B of position 17. The in-phase output of the +OR at 3F will condition one leg of the +AND at 4D, as this in-phase output did previously. Because position 16 trigger B is off, the out-of-phase output of the +OR at 3B is plus. With trigger B of position 16 off (not turned on yet because of circuit delays), the out-of-phase output of trigger B is also plus. Normally, this output would condition the preset AND circuit at 4C and the A trigger of position 16 would turn on. The turning on of the A trigger of position 16 is prevented by "-N SC set gate," which deconditions the circuit at 4C whenever the AS is gated to the SC.

A 70 ns pulse, generated by the CP set pulse, steps the SC. When in automatic, the SC is stepped every clock pluse until reaching zero. The "+N step SC 17" and "-N step SC 17" are generated by the same pulse and will be active at the same time. These two lines condition triggers A and B of position 17. Because trigger B of position 17 is on, the minus step pulse to the +AND at 4F will turn off. Because trigger B of position 16 is in the preset condition, it will latch on with the out-of-phase output of position 17 going positive. Position 15 will not be affected because the step control circuits are not conditioned. Both triggers A and B of position 17 are off, with trigger A preset. Both positions 15 and 16 have trigger B on and the SC value is now six.

The next step pulse (2nd) latches on the A trigger of position 17. The in-phase output of the +AND at 4H conditions the+OR at 3F, presetting trigger B of position 17. The out-of-phase output of the +OR at 3F goes minus and causes trigger B of position 16 to turn off. Because the A trigger was not preset, both triggers in position 16 will be off at the end of the step pulse. After the step pulse, however, trigger B will be preset. Trigger A of position 17 can stay on only for the duration of the step pulse. When the step pulse ends, the 68 uuf capacitor at 2H keeps the output of the preset AND circuit active long enough to allow trigger A to latch on. At the end of this step pulse (2nd), the SC contains a 5.

The next step pulse turns off position 17, and only trigger B of SC 15 is on. The SC is now 4.

Step control circuits now allow positions 15 and 17 to receive the next step pulse. This step pulse will turn off position 15 by deconditioning the AND circuit that corresponds to 4F in Figure 3.1-8B (Systems 03.04.16.1 for actual circuit). The B trigger of position 17 and the A trigger of position 16 are latched on. The SC will continue to step down until it reaches zero. Figure 3.1-8A is a timing chart that shows the stepping of the SC. The solid lines indicate the latched condition, while the dotted lines indicate a preset condition.

3.1.08 Program Counter (PC) (Systems 3.05.00.1-3.05.07.1)

The program counter (PC) is a 16-position count-up counter. The high-order position, PC2, is used to signal that all addresses in core have been set to zero, on the clear operation only. The remaining positions, 3 through 17, indicate the location of the next instruction to be executed by the CPU when it is operating sequentially.

Figure 3.1-9B is a reproduction of an ALD page without the reset lines, and with special terms applied to blocks or combinations of blocks.

The step of the PC is generated on Systems 03.05.08.1. Two lines, "-N STP PC 17" and "N minus to STP PC 17," are generated by +-N advance PC CNTR." These two lines are active whenever PC is stepped.

With the PC set to zero, a step pulse will advance the PC to one. To do so, the step pulse AND's with the out-of-phase output of trigger B of position 17 (now off), to condition the AND circuit at 4H. The +A at 4H presets trigger A of PC17.

Trigger A remains preset until the step pulse falls (becomes positive),and then trigger A latches on. A capacitor holds the preset condition until the latching action takes place.

With trigger A latched on, its output conditions the +OR at 3G as well as the +OR at 3H, signaling a 1 in PC17. With the +OR at 3G conditioned, its in-phase output presets trigger B of PC17. The in-phase output of 3G also presets the +AND at 4D, which in turn presets trigger A of position 16.

The next step pulse will reset trigger A and latch trigger B of PC17. After the step pulse, trigger B of PC17 goes off. With the PC17 trigger off, its out-of-phase output becomes plus, conditioning the +AND at 4E, which latches trigger A of PC16. The +OR at 3C is conditioned, presetting trigger B of PC16. The output of the +OR at 3C also

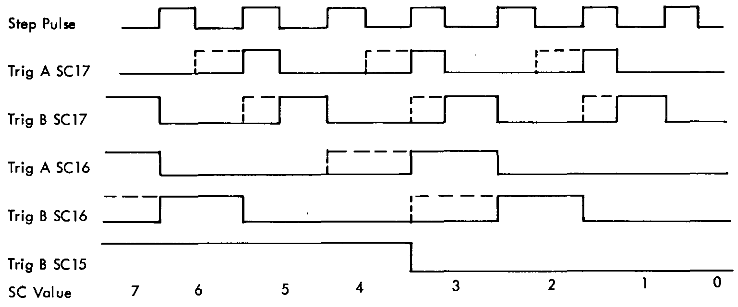


FIGURE 3.1-8A. SHIFT COUNTER

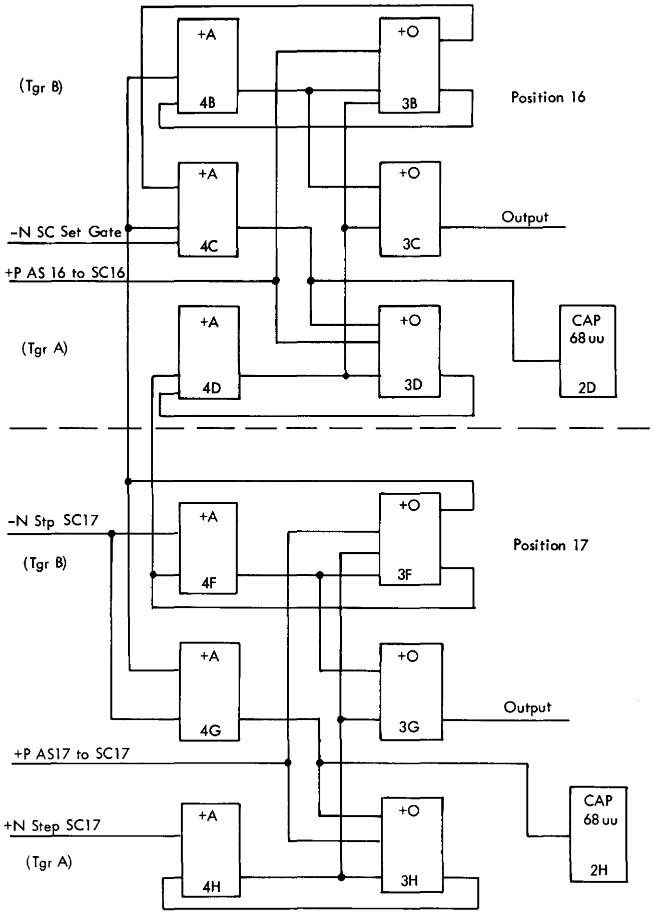


FIGURE 3.1-8B. SHIFT COUNTER POSITIONS 16 AND 17

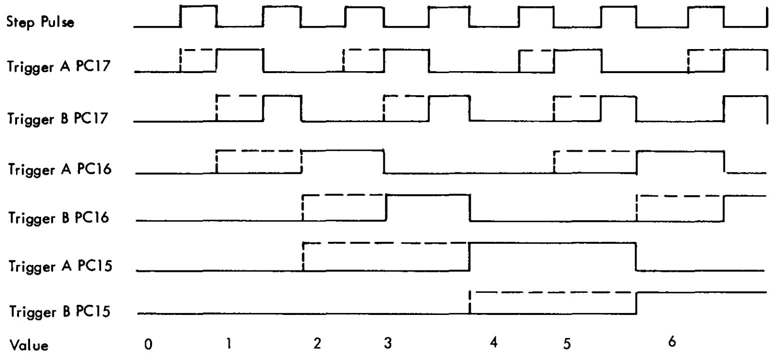


FIGURE 3.1-9A. STEPPING OF THE PROGRAM COUNTER

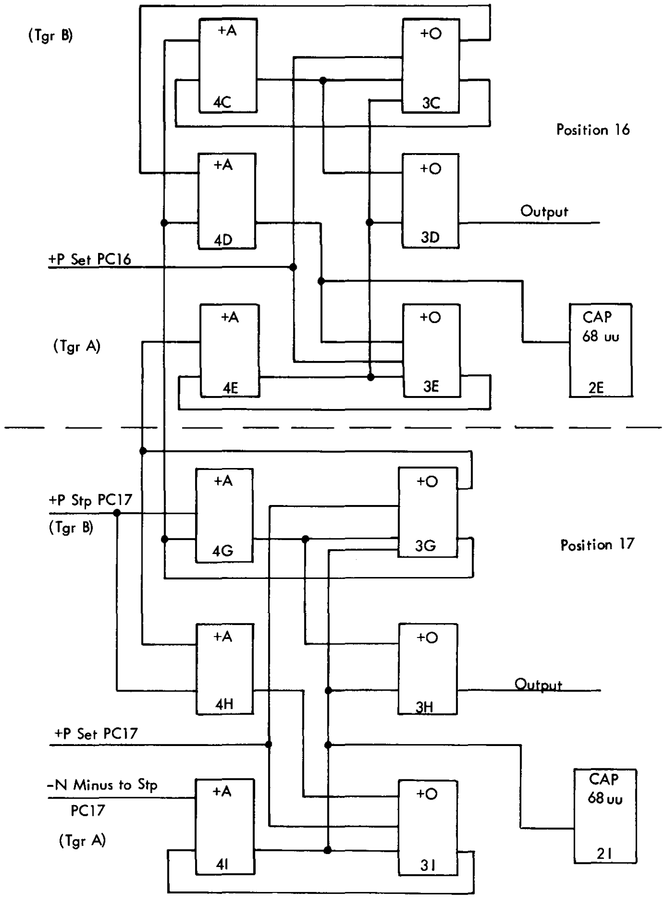


FIGURE 3.1-9B. PROGRAM COUNTER POSITIONS 16 AND 17

presets trigger A of PC15 through the +AND circuit at 4H on Systems 03.05.06.1. This +AND circuit corresponds to the preset AND circuit at 4H in Figure 3.1-9B. At this point, both triggers of PC17 are turned off, trigger A of PC16 is on, and trigger B is preset. Trigger A of PC15 is also preset.

With the "next step PC17," position 17 will turn on in the same manner as before. The different positions will turn on and off as the PC counts up. Figure 3.1-9A is a timing chart showing the stepping of the PC with six pulses. The solid lines indicate the actual condition of a trigger; the dotted lines, a preset condition.

The PC is divided into groups of three, with the outputs of the different groups used, as a lookahead circuit. Each of these groups of three acts like positions 15, 16, and 17, just discussed; however, they will not be stepped as often.

3.1.09 Address Register (AR) (Systems 3.06.00.1-3.06.04.1)

The address register is a 15-position register whose positions are labeled (3-17). The address register gets the desired core storage address before a word is brought out of core storage. The address goes from the address register to the address selection components in core storage.

3.1.10 Address Switches (AS) (Systems 3.06.06.1-3.06.10.1)

The address switch is a 15-position gating circuit with positions labeled (3-17). The switches receive information from the program counter or the adders and this information can be gated through the switches to the address register, shift counter, or storage register.

3.1.11 Tag Registers (Systems 2.08.01.1)

The tag register is a three-position trigger register numbered 18, 19, and 20. It is fed directly from the multiplexor storage buses and is used to hold the tag portion of an instruction during the instruction execution.

3.1.12 Adders (AD) (Systems 2.02.01.1-2.02.37.1)

The adders are transistor switching circuits used to combine binary numbers or words into a binary sum. There are 37 separate adders (Q), (P), (1-35) in the adder unit capable of adding two binary words.

The adders do the arithmetic calculation of the system. Basically all the system will do is add. Subtraction is done by adding a complement number. Multiplication is accomplished by a series of additions and shifts. Division is accomplished by a series of complement additions and shifts.

Figure 3.1-8 shows a basic adder circuit. The adder circuit is designed to use three inputs at one time with one of the three inputs being a carry from the next low order position. The adder produces two outputs, a sum and a carry. The adders are designed to follow the example of binary addition illustrated in Figure 3.1-9.

Some adders have several inputs because the adders have multiple uses. However, only two of the outside inputs and the carry in can be active at the same time. The adding function of these multipurpose adders is the same as shown in the basic adder circuit, Figure 3.1-8.

Adder Unit and Look Ahead Circuits (LAC)

The basic principle behind connecting individual adders together to make an adder unit is to take the carry out of one adder and connect to the carry in of the next high order position adder. This would mean that, if all adder positions contained a one and a one was added to the low order position, a carry would have to ripple through the adder unit from the low order position to the high order position. As the carry ripples through the adders the logic circuits introduce a delay into this action. The 7090 is too fast to accept these conditions. To overcome this slow ripple condition in the adders, the 7090 adders have associated circuits that speed up the carry operation. These circuits determine how many adder positions the carry must go through, then send the carry almost immediately to a higher order adder a few positions from the initial carry impulse. These carry speed-up circuits are called look ahead circuits (LAC). The logic of the operation of the look ahead circuits is shown in Figure 3.1-10 and Figure 3.1-11. Notice that for look ahead purposes the adders are grouped into five groups of five adders each and two groups of six adders. The output of the first stage LAC feeds the inputs of the second stage LAC.

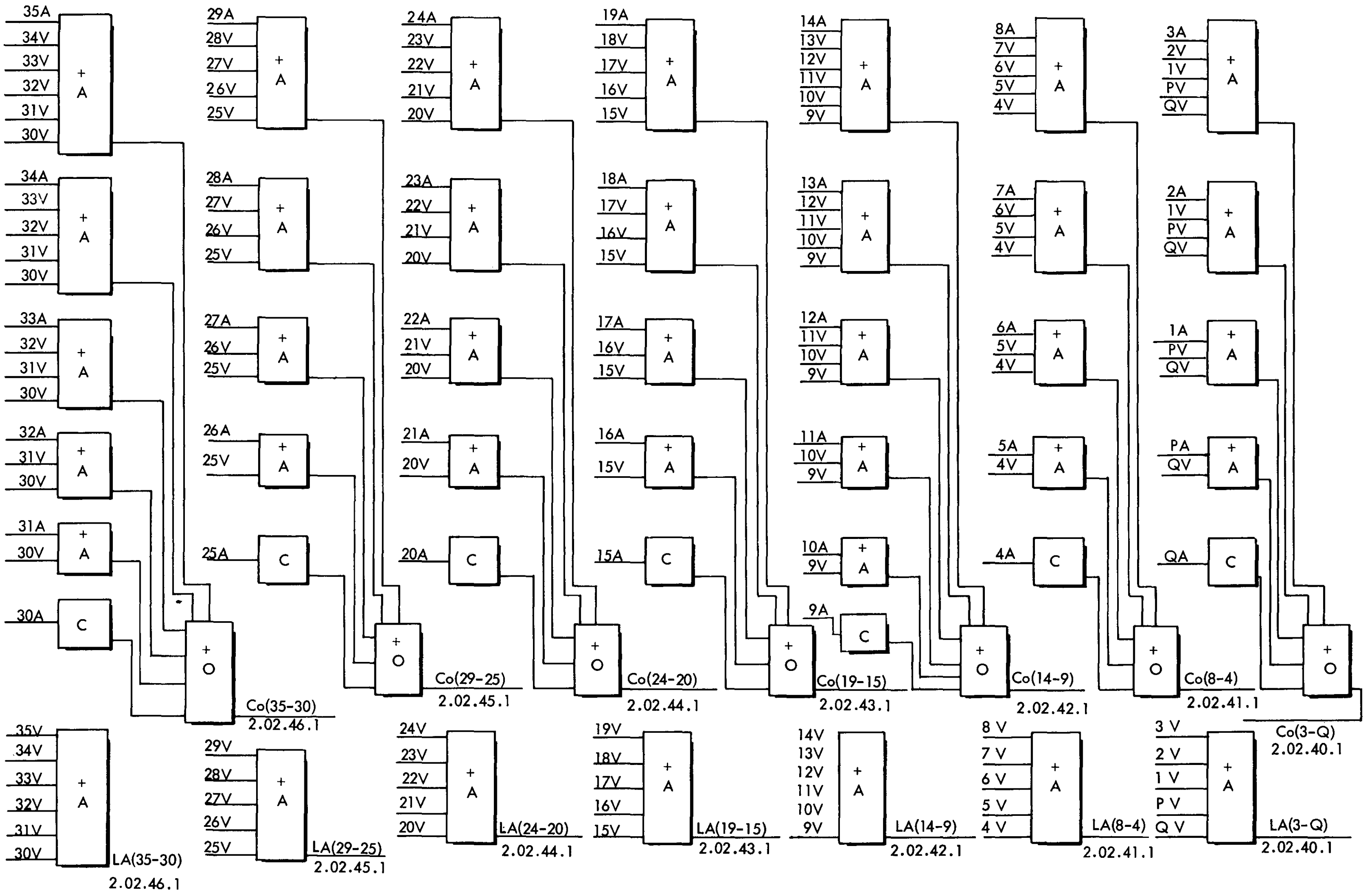
All adder positions, except the Q position, have two LAC outputs. One is the OR LAC and the other is the AND LAC. The OR LAC gives a usable output to the first stage LAC when a bit is sent to either input 1 or input 2 or both. The AND LAC gives a usable output to the first stage LAC when bits are sent to both inputs 1 and 2.

There are two outputs from the first stage LAC. One output is the carry out LAC and the other output is active when all adders in the group are receiving at least one input. This latter output is simply called look-ahead. The carry out (CO) LAC is active when any combination of bits in that particular group of adders would cause a carry out of the high order adder of the group.

The second stage LAC uses the outputs of the first stage LAC, along with a line to indicate a carry in or a bit to adder 35. These LAC initiate a carry into the low order adder of the next higher group.

Following through the first and second stage LAC, note that if adder 35 has an AND LAC output and adders 34-30 have OR LAC outputs, a carry in would be sent to adder 29. If all adders have an OR LAC output, these outputs would cause a carry in to be sent to adders 29, 24, 19, 14, 8, and 3 almost simultaneously if a carry into adder 35 occurred.

Note that a carry has to ripple through a maximum of 6 adders rather than through 37 because of the look ahead circuits, and the carry operation through the adders has been greatly speeded up.



A = 2 Bits (AND LAC) V = At Least 1 Bit (OR LAC) Co = Carry Out LA = Look Ahead

FIGURE 3.1-10. FIRST STAGE LOOK AHEAD CIRCUITS

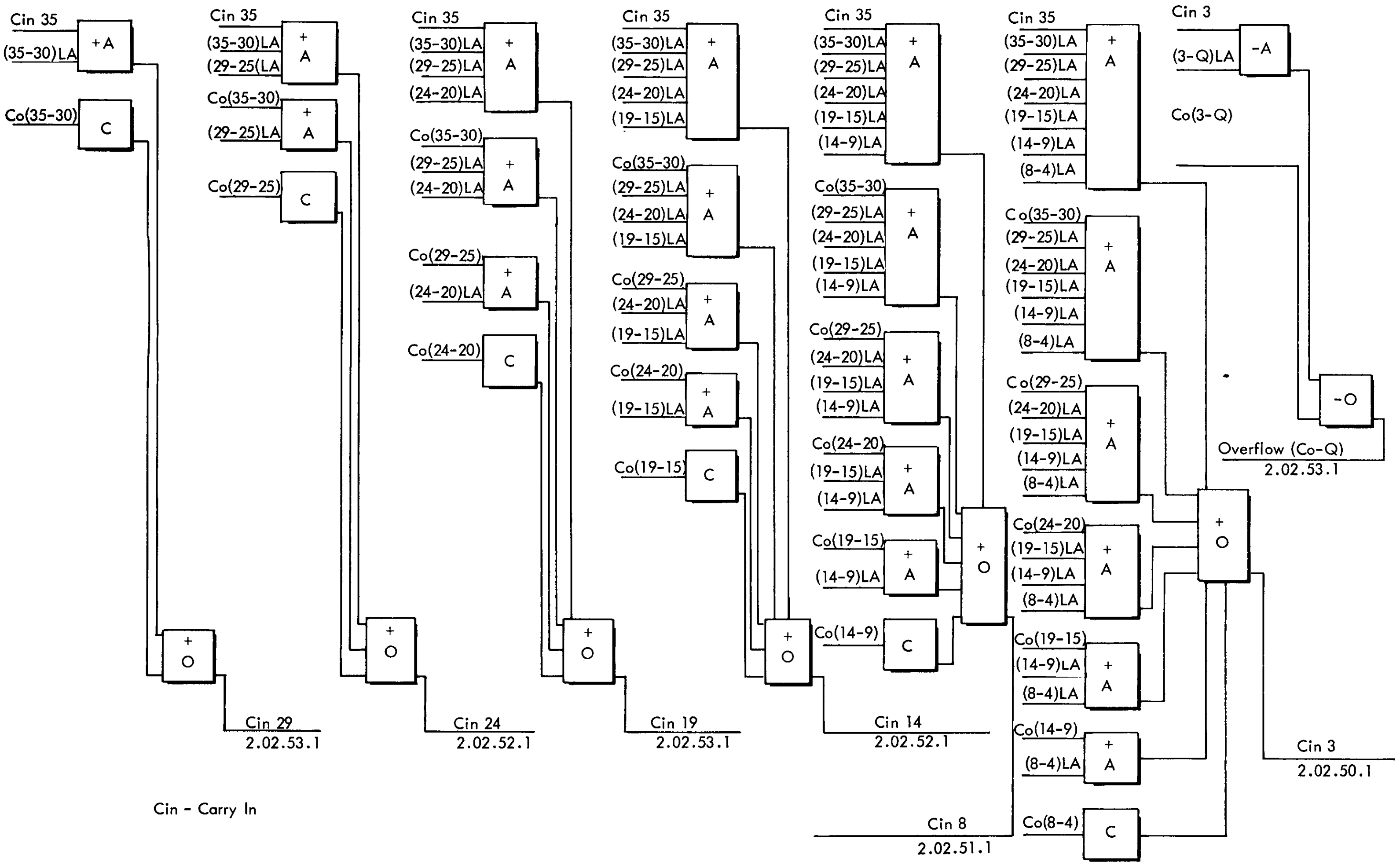


FIGURE 3.1-11. SECOND STAGE LOOK AHEAD CIRCUITS

Carry From Adder 9 to Adder 8

A special circuit can prevent or allow a carry from adder 9 to adder 8. The reason for this is that a floating-point word has two separate parts, a characteristic (positions 1-8) and a fraction (positions 9-35). During some floating-point operations, a carry out of the fraction must not be allowed to gate to the characteristic.

3.2.00 INSTRUCTION DECODING AND PROCESSING

The output of the program counter is gated to the address register through the address switches. The output of the address register is then used to locate the address of the instruction in core storage. The output of core storage is sent to the storage register and the operation code is also sent to the program register. The output of the program register then is decoded to instruct the system.

3.2.01 Operation Decoders

The output of the program register feeds two decoders. The outputs of positions 1-5 feed the primary operation decoders (POD), and the outputs of positions 6-9 feed the secondary operation decoders (SOD). The primary operation is used to establish the basic execution control routing. The secondary operation decoder is used only for the sense, or primary operation 76 instructions.

Some similar instructions have the same primary operation code. Direct outputs of the program register or storage register are sent to individual machine circuits to cause the machine to operate according to the minor differences among these similar instructions.

3.2.02 Control Circuits

The control circuits cause the machine to operate according to the decoding of the instruction. Specific control circuits cause data to be moved and functions to be performed so the machine gives the desired results of the instruction.

3.2.03 Pulses

The control circuits require a great number of pulses and gates. These pulses and gates, obtained by mixing clock pulses from the multiplexor clock, function in the 7090 as the circuit breakers function in electro-mechanical card machines. They are used to establish proper sequence of operation in the system.

3.3.00 BASIC CYCLE

The basic cycle in the CPU is 2.1818 microseconds. It is divided into 12 equal times of the 181.8 milli-microseconds each. The 12 times are sent to the CPU from the multiplexor.

There are three different types of cycles used by the CPU. These are:

1. Instruction	I cycle
2. Execution	E cycle
3. Logic	L cycle

Every instruction has an I cycle. This cycle brings the instruction from core storage to the CPU, where the instruction is decoded. Some instructions require only the I cycle for their execution; most instructions require additional cycles.

E cycles are used when information is to be moved between core storage and the CPU to execute an instruction.

L cycles are used to execute an instruction when information from core storage is not needed.

Most instructions use only one E or L cycle for their execution, but some instructions require multiple E or L cycles. For example: a convert instruction requires several E cycles; a multiply instruction requires several L cycles.

4.0.00 IBM 7606 MULTIPLEXOR

The IBM 7607 Multiplexor is best described as the unit of the 7090 system that passes all data to or from core storage. These data may be CPU instructions, data channel commands, data to be processed by the CPU, data that has been processed by the CPU, or input-output words. Data that go to core storage come to the multiplexor from the CPU and data channels.

Multiplexing is the forming of parallel paths from a serial path when data are going to core storage. When data are coming from core storage, multiplexing is the forming of a serial path from parallel paths.

In addition to its function of multiplexing, the multiplexor contains the clock for the CPU and data channels, and a matrix for zero testing data coming from core storage. The multiplexor also contains look-ahead circuits that are used in conjunction with the data channel operations. The CPU clock and CPU multiplexing circuits are described in this manual; the remainder of multiplexor functions are described in Customer Engineering Manual of Instruction, IBM 7607 Data Channel, Form 223-6898.

Two of the four standard modular system (SMS) sliding gates in the multiplexor frame are reserved for special features. The remaining two SMS sliding gates house the multiplexor circuitry.

Data flow from core storage to the multiplexor storage bus, then to either the CPU or any one of the data channels. See Figure 4.0-1. The address field (21-35) of the data coming from core storage also flows to the multiplexor address switches. The flow through the multiplexor address switches is controlled by the look-ahead circuits in the multiplexor.

Data going to core storage flow through the multiplexor storage bus OR'ing. This bus is a group of OR circuits that multiplex data going to core storage. It is completely independent of the multiplexor storage bus.

The multiplexor address switches receive addresses from the CPU, data channels, and the multiplexor storage bus. These switches multiplex the address sent to core storage and the location counter switches in each data channel.

4.1.00 MULTIPLEXOR FUNCTIONAL UNITS

The multiplexor contains the multiplexor clock, storage bus, storage bus OR'ing, and the address switches.

4.1.01 Multiplexor Clock

The multiplexor clock (Figure 4.1-1) supplies timing pulses to the 7090 system. The clock consists of a 12-stage ring. The clock is started by turning on the zero clock trigger with a start clock pulse. The A0(D1) clock pulse is obtained by gating the last half of the zero clock trigger pulse with the negative portion of the even ring drive pulse. The rise of the A0(D1) pulse turns on one clock trigger. The A1(D1) pulse is obtained

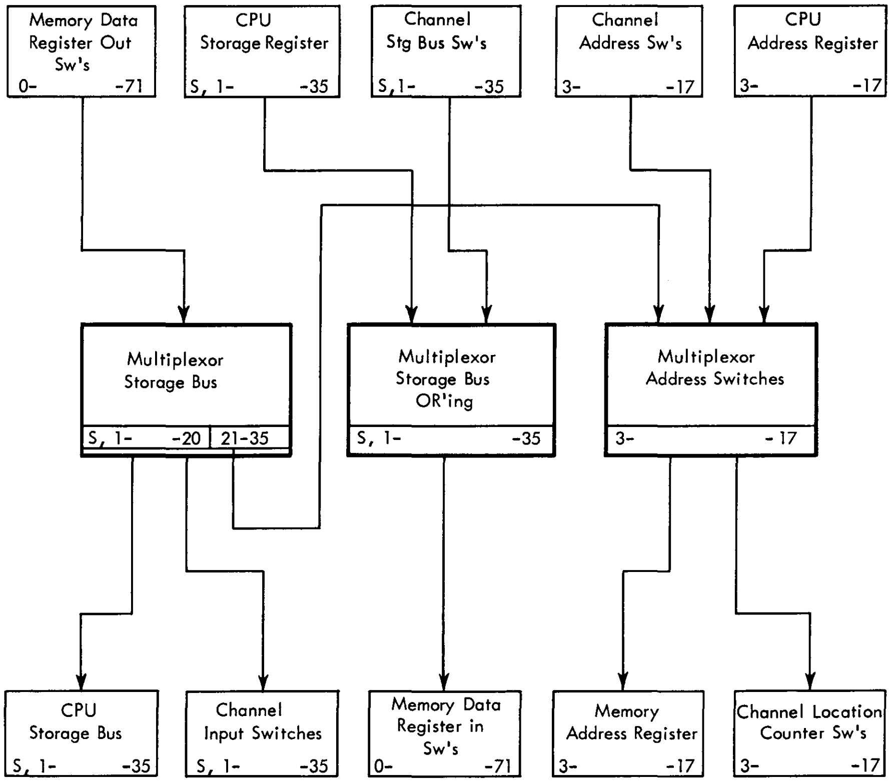


FIGURE 4.0-1. FLOW TO AND FROM THE MULTIPLEXOR

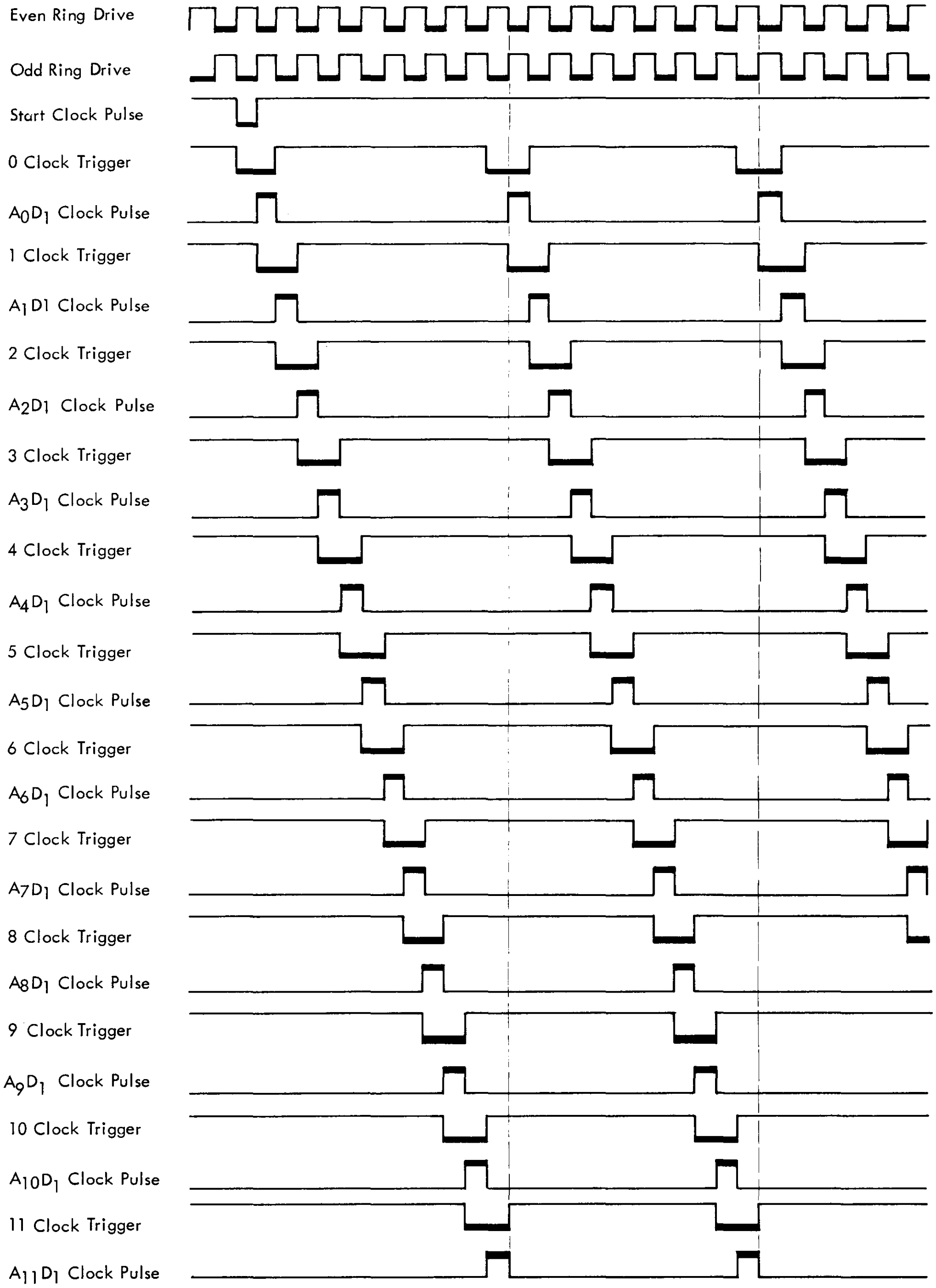


FIGURE 4.1-1. MULTIPLEXOR CLOCK

by gating the last half of the 1 clock trigger pulse with the negative portion of the odd ring drive pulse. The rise of the A1(D1) pulse turns on two clock trigger, and the turning on of the two clock trigger turns off zero clock trigger. This sequence continues through 11 clock trigger. The rise of the A11(D1) pulse turns zero clock trigger on again. The following chart shows the controls significant to each stage of the ring.

Clock Tgr	Turned on by	Turned off by	Duration	Output	Gated Output
0	A11(D1)	2 clock tgr	A11(D2)	A11(D2)	A0(D1)
1	A0(D1)	3 clock tgr	A0(D2)	A0(D2)	A1(D1)
2	A1(D1)	4 clock tgr	A1(D2)	A1(D2)	A2(D1)
3	A2(D1)	5 clock tgr	A2(D2)	A2(D2)	A3(D1)
4	A3(D1)	6 clock tgr	A3(D2)	A3(D2)	A4(D1)
5	A4(D1)	7 clock tgr	A4(D2)	A4(D2)	A5(D1)
6	A5(D1)	8 clock tgr	A5(D2)	A5(D2)	A6(D1)
7	A6(D1)	9 clock tgr	A6(D2)	A6(D2)	A7(D1)
8	A7(D1)	10 clock tgr	A7(D2)	A7(D2)	A8(D1)
9	A8(D1)	11 clock tgr	A8(D2)	A8(D2)	A9(D1)
10	A9(D1)	0 clock tgr	A9(D2)	A9(D2)	A10(D1)
11	A10(D1)	1 clock tgr	A10(D2)	A10(D2)	A11(D1)

Note that the pulse width of each clock trigger is twice that of an individual clock pulse. This slower switching of the clock triggers provides for increased reliability in the operation of the clock.

The multiplexor clock pulse distribution enables the individual clock pulses to be distributed to the CPU and data channels. Because of inherent delays in logic blocks, clock pulses distributed to the CPU arrive about one clock pulse late. For this reason, those clock pulses distributed from the multiplexor to the CPU are labeled one higher than the actual clock pulse. Therefore, an A0(D1) pulse going to the CPU would be labeled A1(D1). This pulse leaves the multiplexor at A0 time but, when it arrives at the CPU, the A1(D1) pulse is rising at the multiplexor. It is important to notice that A1(D1) pulses at the CPU and multiplexor now are in coincidence, although developed from different clock triggers. This provides for continuity in the timing relationship between the CPU and multiplexor.

4.1.02 Multiplexor Storage Bus

The multiplexor storage bus (Figure 4.1-2) routes all data from core storage to either the data channels or the CPU. Seventy-two lines from core storage feed the bus, 36 of which lines carry data at a given time. Lines 0-35 are logically OR'ed with lines 36-71, respectively.

The bus feeds a group of 36 AND circuits that act as inputs to the CPU. The bus also feeds both banks of the data channel by way of the channel buses.

In a multiplexor storage bus test, positions 1-35 of the multiplexor storage bus feed a matrix, which tests positions 1-35 and 3-17 for a zero condition. The zero test on positions 3-17 is used in conjunction with the multiplexor look-ahead circuits. Zero testing positions 1-35 provide for minimum execution time of various CPU instructions.

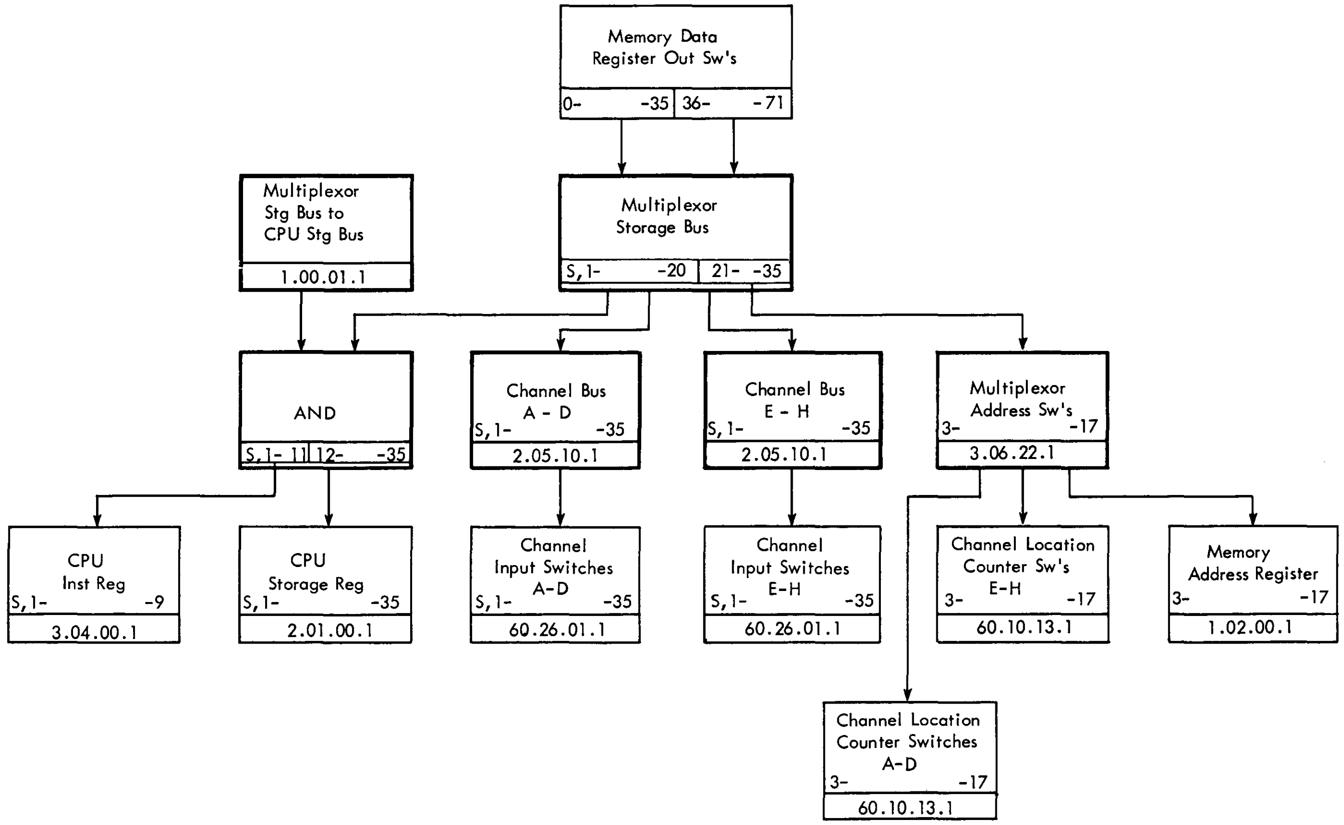


FIGURE 4.1-2. FLOW FROM MULTIPLEXOR TO CPU AND DATA CHANNELS

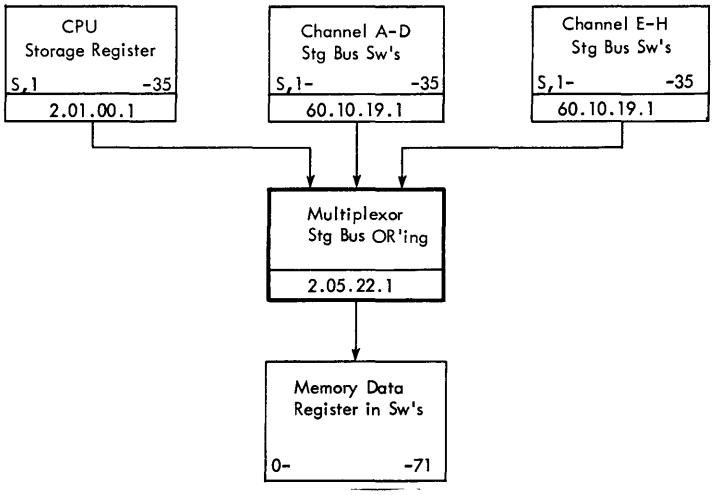


FIGURE 4.1-3. FLOW FROM CPU AND DATA CHANNELS TO MULTIPLEXOR

4.1.03 Multiplexor Storage Bus OR'ing

All data going to core storage are routed through the multiplexor storage bus OR'ing circuits. See Figure 4.1-3. These circuits consist of OR circuits that multiplex data coming from either bank of the data channels or the CPU. Positions S, 1-35 of the CPU storage register are logically OR'ed with positions S, 1-35 of both banks of channel storage bus switches. This provides the proper isolation between the three sets of inputs to the bus, and allows for the proper matching of the output.

It is important to notice that data being routed through these circuits are gated at the CPU storage register or the channel storage bus switches.

The output of the OR circuits consists of 36 (S, 1-35) lines, powered and matched, to route data to core storage.

4.1.04 Multiplexor Address Switches

The address, where data are taken to or from, must be switched through the multiplexor address switches. These switches determine whether the address going to core storage is from a data channel, the CPU, the multiplexor storage bus, or is a forced address due to a channel trap. The switches also provide isolation between the various inputs. The switches have three outputs, all of which are active simultaneously. Two of the outputs feed both banks of the data channels and terminate at the location counter switches. The third output feeds the memory address register in core storage.

4.2.00 DATA FLOW AND CONTROL

Data flow through the multiplexor and its associated control circuitry in the CPU can best be described by examining the following paths:

CPU to core storage
Core storage to CPU

4.2.01 CPU to Core Storage

Data that flow from the CPU to core storage (Figure 4.1-3) are routed from the storage register through the multiplexor storage bus OR'ing circuits to the memory data register. Data flow is controlled by gating at the output of the storage register. The data are gated during an E cycle in which CPU control circuitry calls for store control.

The address at which the data are being stored is switched through the multiplexor address switches. The CPU address register output is gated through these switches as long as the B time trigger is not on. The address is sampled at the memory address register at a given time.

The data flow from the multiplexor to core storage on 36 data lines. At core storage, the data must be set to either 0-35 or 36-71 of the memory data register. The address set to the memory address register specifies which half of the memory data register is set.

4.2.02 Core Storage to CPU

When in automatic status, data flow from core storage to the CPU (Figure 4.1-2) always occurs during I cycles and may occur during E cycles. In either case, the data are routed to the multiplexor storage bus where they are gated to the CPU. Data transmitted to the CPU during I cycles are in the format of instructions, whereas data transmitted to the CPU during E cycles may be in any format.

Notice that the bus unconditionally feeds both banks of data channels. However, data are gated through the channel input switches only during B cycles or the E cycle of a reset and load or load channel instruction. The address field (21-35) of the bus also feeds the multiplexor address switches. These positions are gated through the switches only during a look-ahead address control operation.

I Cycle Flow

At I6(D3), data on the multiplexor storage bus are gated to the CPU. CPU storage bus positions S, 1-35 are set to the storage register at I7(D1), while positions S, 1-11 are routed to the instruction register. If the CPU storage bus positions 1 or 2 hold a one at I8(D1), bus positions S, 1, and 2 are set to instruction register positions S, 8, and 9, respectively. If the CPU storage bus positions 1 or 2 do not hold a one at I8(D1), bus positions S, 3-11 are set to instruction register positions S, 1-9 respectively.

E Cycle Flow

During all E cycles in which store control is not up, the multiplexor storage bus is gated to the CPU at E6(D3). At E7(D1), positions S, 1-35 of the CPU storage bus are set to the storage register.

5.0.00 CPU DATA FLOW AND TIMING

5.1.00 I CYCLE

I cycle operation is much the same for all instructions. E and L cycle operations vary depending on the instruction.

I cycle operation is shown in Figure 5.1-1. The end operation trigger (END OP TGR) is turned on during the last cycle of an operation and for an initial starting condition. This causes the I time trigger to be turned on. Also with the end operation trigger on, the address of the instruction is sent to core storage. The instruction is then taken from that address in core storage and sent to the CPU on the storage bus (SB). The instruction is put into the storage register (SR) and the operation code is sent to the program register (PR). Note the difference between instructions with a bit in positions 1 and 2 and those without a bit in 1 or 2. Few instructions have a bit in positions 1 or 2.

When the instruction is decoded the computer will decide what kind of a cycle is to be taken. Most instructions requiring an L cycle have operation codes 0-177 or from 700 on. Instructions with operation codes between these limits usually require an E cycle. The exceptions to these general L and E time call rules are in the MF systems pages.

Some instructions require only one cycle; therefore, the end operation trigger is turned on during I time. This forces "go to I time" on Systems 8.00.12.1. "Go to I time" will turn on the master I time trigger and will prevent turning on of the master E or L time triggers.

The program counter is advanced to give the location of the next sequential instruction in core storage. The address portion of the instruction is sent to the address register. For primary operations 76 it is sent to the shift counter. The address in the address register is used to locate a particular address in storage if the next cycle is an E cycle. Part of the address is sent to the shift counter on primary operations 76. The shift counter is used on these operations along with the program register to instruct the system.

5.2.00 INDIRECT ADDRESSING

Indirect addressing is a programming device which permits changing an instruction address. Bits in positions 12 and 13 of an instruction are signals for indirect addressing. This causes an E cycle, during which the word located at the original instruction address is read out of storage and its address portion is substituted for the instruction address. The instruction then operates on the new address. Only indexable instructions may be indirectly addressed. The only change in operation of an instruction caused by indirect addressing is the insertion of an E cycle. As with direct address instructions, indexing is automatic.

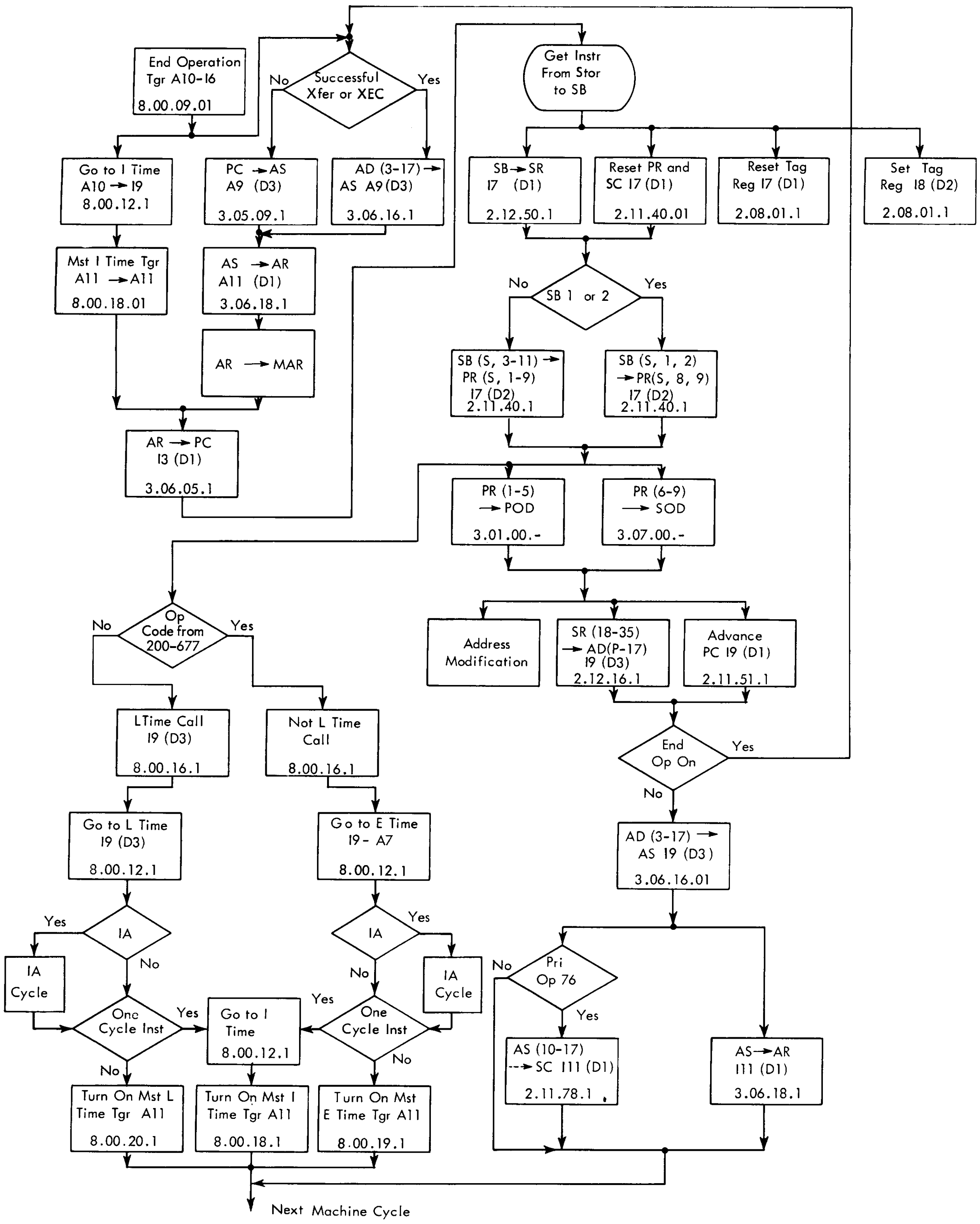

End Operation Tgr A10-16 8.00.09.01
Successful Xfer or XEC
No
Yes
Go to I Time A10 → I9 8.00.12.1
PC → AS A9 (D3) 3.05.09.1
AD (3-17) → AS A9(D3) 3.06.16.1
Mst I Time Tgr A11 → A11 8.00.18.01
AS → AR A11 (D1) 3.06.18.1
AR → MAR
AR → PC I3 (D1) 3.06.05.1
Get Instr From Stor to SB
SB → SR I7 (D1) 2.12.50.1
Reset PR and SC I7 (D1) 2.11.40.01
Reset Tag Reg I7 (D1) 2.08.01.1
Set Tag Reg I8 (D2) 2.08.01.1
SB 1 or 2
No
Yes
SB (S, 3-11) → PR (S, 1-9) I7 (D2) 2.11.40.1
SB (S, 1, 2) → PR(S, 8, 9) I7 (D2) 2.11.40.1
PR (1-5) → POD 3.01.00.-
PR (6-9) → SOD 3.07.00.-
Address Modification
SR (18-35) → AD(P-17) I9 (D3) 2.12.16.1
Advance PC I9 (D1) 2.11.51.1
End Op On
Yes
No
AD (3-17) → AS I9 (D3) 3.06.16.01
Op Code from 200-677
No
Yes
LTime Call I9 (D3) 8.00.16.1
Not L Time Call 8.00.16.1
Go to L Time I9 (D3) 8.00.12.1
Go to E Time I9 - A7 8.00.12.1
IA
Yes
No
IA Cycle
One Cycle Inst
Yes
No
Go to I Time 8.00.12.1
IA
Yes
No
IA Cycle
One Cycle Inst
Yes
No
Turn On Mst L Time Tgr A11 8.00.20.1
Turn On Mst I Time Tgr A11 8.00.18.1
Turn On Mst E Time Tgr A11 8.00.19.1
Pri Op 76
No
Yes
AS (10-17) → SC I11 (D1) 2.11.78.1
AS → AR I11 (D1) 3.06.18.1
Next Machine Cycle

FIGURE 5.1-1. INSTRUCTION CYCLE

Indirect addressing controls are shown in Figure 5.2-1. Once an indirect addressing condition is recognized, the IA control trigger is turned on to block normal operations and an E cycle is initiated. When the E cycle is completed, the trigger is turned off, and normal execution is resumed.

5.3.00 INSTRUCTIONS

5.3.01 Word Transmission Instructions*

Word transmission instructions are necessary to move information into and out of the CPU. Factors must be brought in from core storage and results must be returned. Instructions are available to move parts of words or whole words to or from core storage.

Store STO +0601(I, E) Figure 5.3-1

This instruction moves a full word to core storage. The contents of the AC(S, 1-35) replace the contents of storage location X. The AC is unchanged. The store prefix, decrement, tag, address, and MF store control lines are activated so a full word can be put into core storage.

Store Logical Word SLW +0602 (I, E) Figure 5.3-1

This instruction replaces the contents of storage location X with the contents of the AC(P-35). The AC is unchanged. Execution of this instruction is identical to that of store except for the routing of AC(P) to the storage register sign position. AC(P-35) is gated to SR(S-35) on Systems 2.12.02.1.

Store MQ STQ -0600 (I, E) Figure 5.3-1

The contents of core storage location X are replaced by the contents of the MQ(S, 1-35) register. The contents of the MQ are unchanged. This instruction operates similarly to store, except that the word sent to the SR comes from the MQ rather than from the AC.

Store Zero STZ +0600(I, E) Figure 5.3-1

This instruction causes zeros to be placed in all positions of storage location X. The operation of this instruction is similar to store, except that nothing is put on the SB. Therefore, when the storage bus is gated to core storage, zeros are put into that location of storage.

Store Prefix STP +0630(I, E) Figure 5.3-2

This instruction places the contents of AC positions (P, 1 and 2) into core storage location X, positions (S, 1 and 2). The contents of the AC and storage positions (3 through 35) are unchanged. MF store prefix and MF store control are activated to cause core storage to store what is on SB(S, 1 and 2). To get the information to the storage bus, it must be moved from the accumulator to the SR.

*See Store Location and Trap, and Figure 5.3-34.

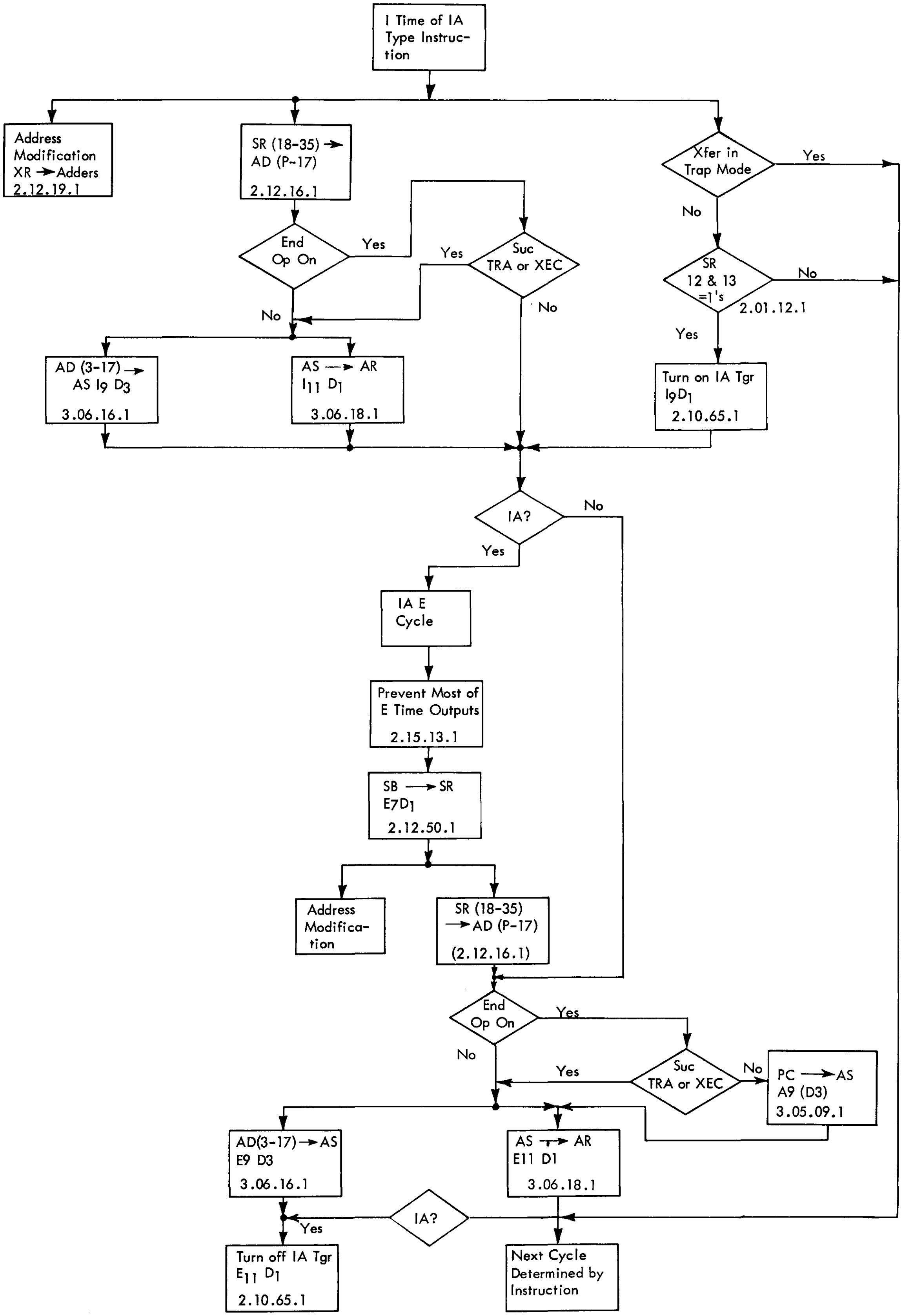

I Time of IA Type Instruction
Address Modification XR → Adders 2.12.19.1
SR (18-35) → AD (P-17) 2.12.16.1
Xfer in Trap Mode
Yes
No
End Op On
Yes
Suc TRA or XEC
Yes
No
No
SR 12 & 13 =1's
2.01.12.1
No
Yes
AD (3-17) → AS I9 D3 3.06.16.1
AS → AR I11 D1 3.06.18.1
Turn on IA Tgr I9D1 2.10.65.1
IA?
No
Yes
IA E Cycle
Prevent Most of E Time Outputs 2.15.13.1
SB → SR E7D1 2.12.50.1
Address Modification
SR (18-35) → AD (P-17) (2.12.16.1)
End Op On
Yes
No
Yes
Suc TRA or XEC
No
PC → AS A9 (D3) 3.05.09.1
AD(3-17) → AS E9 D3 3.06.16.1
AS → AR E11 D1 3.06.18.1
IA?
Yes
Turn off IA Tgr E11 D1 2.10.65.1
Next Cycle Determined by Instruction

FIGURE 5.2-1. INDIRECT ADDRESSING

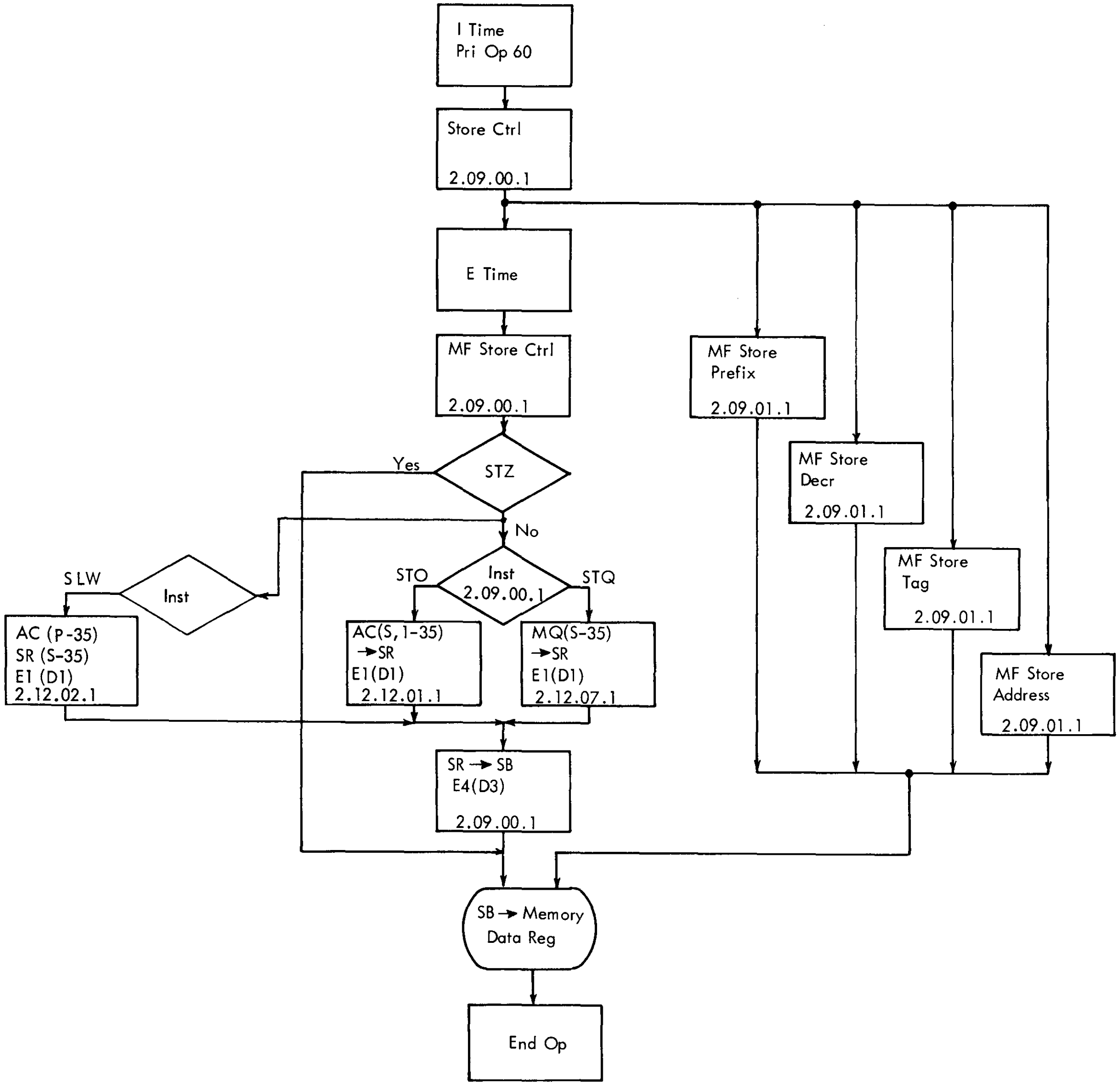


FIGURE 5.3-1. STO + 0601; STQ - 0600; STZ + 0600; SLW + 602

Store Decrement STD +0622 (I, E) Figure 5.3-2

The contents of AC(3-17) replace the contents of positions (3-17) of core storage location X. The remaining storage positions are unchanged, and the AC is unchanged. This instruction operates similarly to store prefix, except that MF store decrement rather than MF store prefix is made active.

Store Tag STT +0625 (I, E) Figure 5.3-2

The contents of AC(18-20) replace the contents of positions (18-20) of core storage location X. The AC and the remaining positions of core storage are unchanged. This instruction operates similarly to store prefix, except that MF store tag rather than MF store prefix is made active.

Store Address STA +0621 (I, E) Figure 5.3-2

The contents of AC(21-35) replace the contents of positions (21-35) of core storage location X. The remaining positions of storage and the contents of the AC are unchanged. This instruction also operates similarly to store prefix, except that MF store address rather than MF store prefix is made active.

Store Left Half MQ SLQ -0620 (I, E) Figure 5.3-3

The contents of positions (S-17) of the MQ replace the contents of positions (S-17) of storage location X. The remaining storage positions and the contents of the MQ are unchanged. MF store prefix, MF store decrement, and MF store control lines are activated so SB(S-17) can be read into storage. The word is sent from the MQ to the SR so it can be put on the SB.

Store Instruction Location Counter STL -0625 (I, E) Figure 5.3-4

The contents of the program counter, which contains the location of the STL instruction plus one, replace the contents of positions (21-35) of storage location X. The contents of the PC and the remaining positions of storage are unchanged. The MF store address and MF store control lines are activated so the address portion of the storage word can be changed. The PC contents are sent to core storage on the SB. The contents of the PC are sent to the SR through the AS and the output of the SR feeds the SB.

Load MQ LDQ +0560 (I, E) Figure 5.3-5

The contents of the MQ are replaced by the contents of storage location X. The storage word is put into the SR, and the output of the SR is sent to the MQ.

Exchange AC and MQ XCA +0131 (I) Figure 5.3-6

The contents of the AC(S, 1-35) are interchanged with the contents of the MQ(S, 1-35). AC positions Q and P are reset. The SR is used for temporary storage while routing the MQ to the AC.

Exchange Logical Accumulator and MQ XCL -0130 (I) Figure 5.3-7

This instruction interchanges the contents of AC(P-35) and the contents of the MQ (S-35). Positions (S) and (Q) of the AC are cleared. Execution is identical to that of XCA except for the handling of AC positions (S) and (P). The contents of the MQ are put into the SR. The contents of the SR(S, 1-35) and the AC (P, 1-35) are interchanged, putting the original MQ in the AC. Gating the output of the SR(S, 1-35) to the MQ(S, 1-35) places the original AC contents in the MQ. AC positions (Q) and (S) are cleared because of normal shift cell operations.

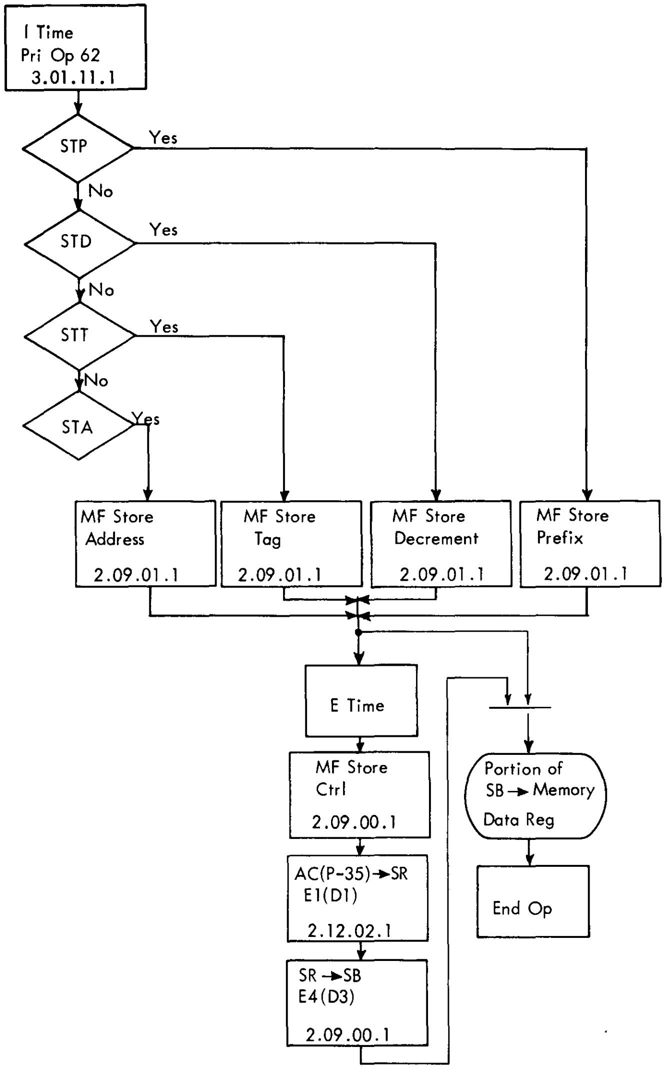


FIGURE 5.3-2. STP +0630; STD +0622; STT +0625; STA +0621

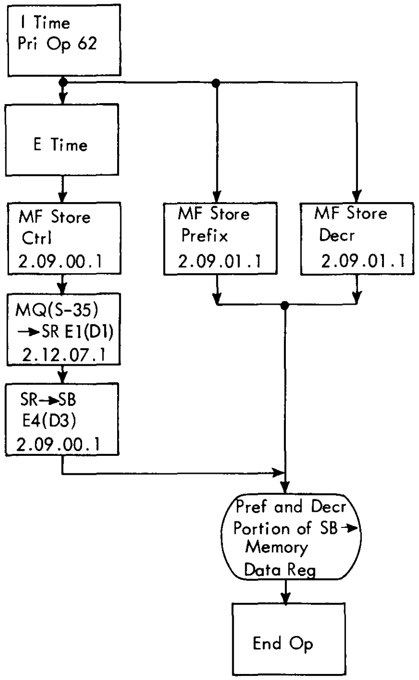


FIGURE 5.3-3. SLQ - 0620

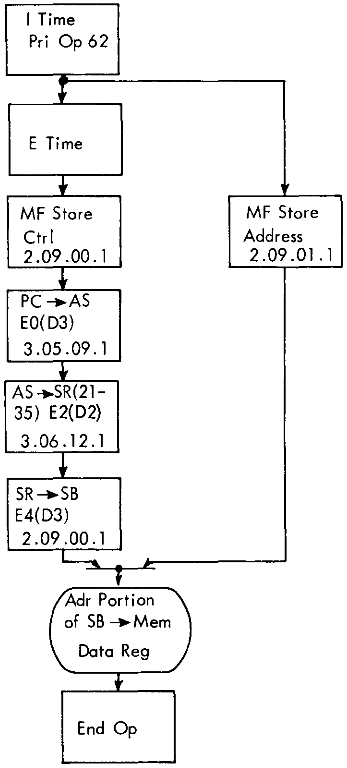


FIGURE 5.3-4. STL -0625

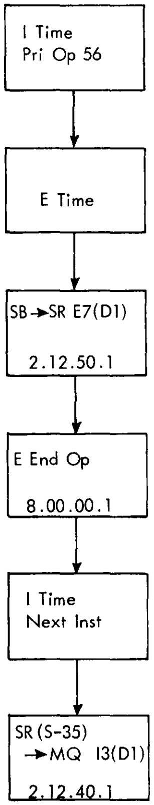


FIGURE 5.3-5. LDQ +0560

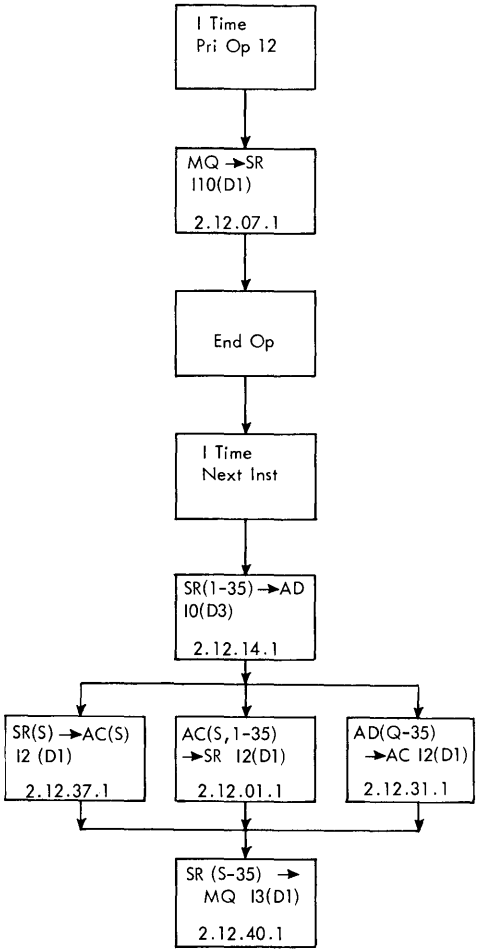


FIGURE 5.3-6. XCA +0131

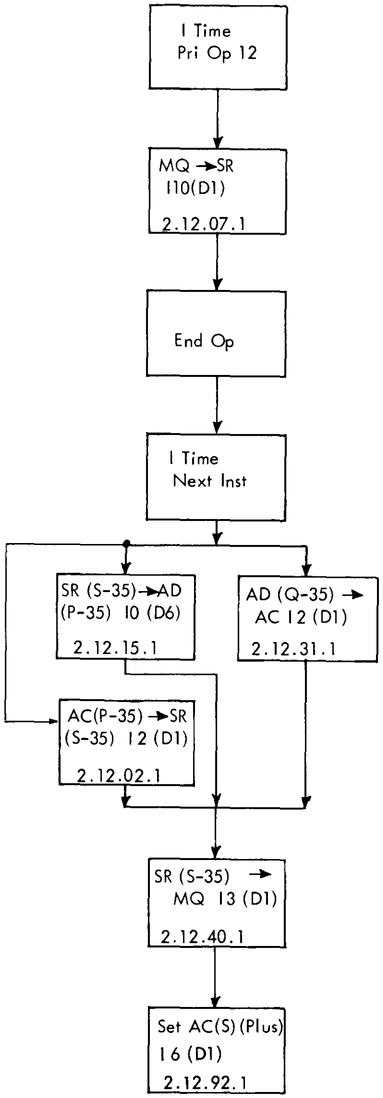


FIGURE 5.3-7. XCL -0130

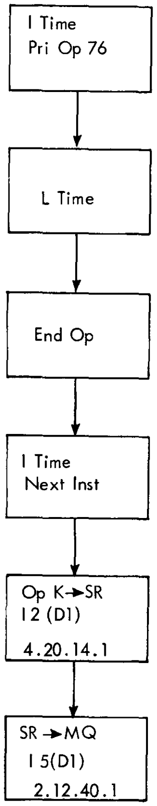


FIGURE 5.3-8. ENK +0760...0004

Enter Keys ENK +0760...0004 (I, L) Figure 5.3-8

The word represented in console keys (S, 1-35) replaces the contents of the MQ. This is a primary operation 76 instruction. The word in the keys is put into the SR and the output of the SR is gated into the MQ.

5.3.02 Shifting Instructions

Shift instructions are used to align words, or for fast multiplication or division by a power of 2. Shifting moves the bits of the AC or MQ, or both, to the right or left within the registers. Bits shifted out of the end of a register are lost and bits shifted away from either end of a register are replaced by zeros. Because the shift counter receives only the last eight positions of the instruction address, the maximum number of shifts possible is 255_{10} ($11\ 111\ 111_2$).

The number of shifts to be taken is indicated by the address portion of the shift instruction. This address is gated to the shift counter at I11(D1). Shifting starts at the next L1 time and continues to shift at the rate of one position of shift for each clock pulse until the shift counter has been reduced to zero.

The cyclic makeup of a shifting instruction is an I time followed with as many L times as required to complete the shifts designated by the address portion of the instruction. Eleven shifts may be completed during the first L time, and twelve shifts may be completed during all succeeding L times. The "end operation" condition is signaled by having the shift counter at seven or less at any L10 time of a shift instruction. This means that up to five shifts may be made in the I time of the following instruction.

Shifting to the left is the same as multiplying by a power of 2; shifting to the right reduces or divides by a power of 2. The number of shifts is equal to the exponent.

Accumulator Left Shift ALS +0767 (I, L...) Figure 5.3-9

This instruction causes the contents of the AC (Q-35)* to be shifted left a number of places equal to the eight low order positions of the address. Zeros replace any bits shifted away from position (35). Bits shifted past Q are lost. A "1" shifted into the P position turns on the AC overflow indicator.

Long Left Shift LLS +0763 (I, L...) Figure 5.3-10

For this instruction the contents of the MQ and AC (except the sign positions) are shifted left the number of places designated by the eight low order positions of the address. The MQ (1) position is shifted to the AC (35) position. The AC sign is set to agree with the MQ sign. Bits shifted past Q are lost and bits shifted away from MQ(35) are replaced by zeros. Bits shifted into P cause the AC overflow indicator to be turned on.

Logical Left Shift LGL -0763 (I, L...) Figure 5.3-10

This instruction shifts the contents of AC(Q-35) and MQ(S-35) left the number of places designated by the address. Bits shifted from MQ(1) enter MQ(S), and from MQ (S) enter AC(35). Bits shifted into AC(P) cause the AC overflow indicator to be turned on. Bits shifted past AC(Q) are lost and bits shifted away from MQ(35) are replaced by zeros. The operation of LGL is similar to LLS except for handling of the MQ sign.

* This text section uses (Q - 35) to represent (Q, P, 1 - 35).

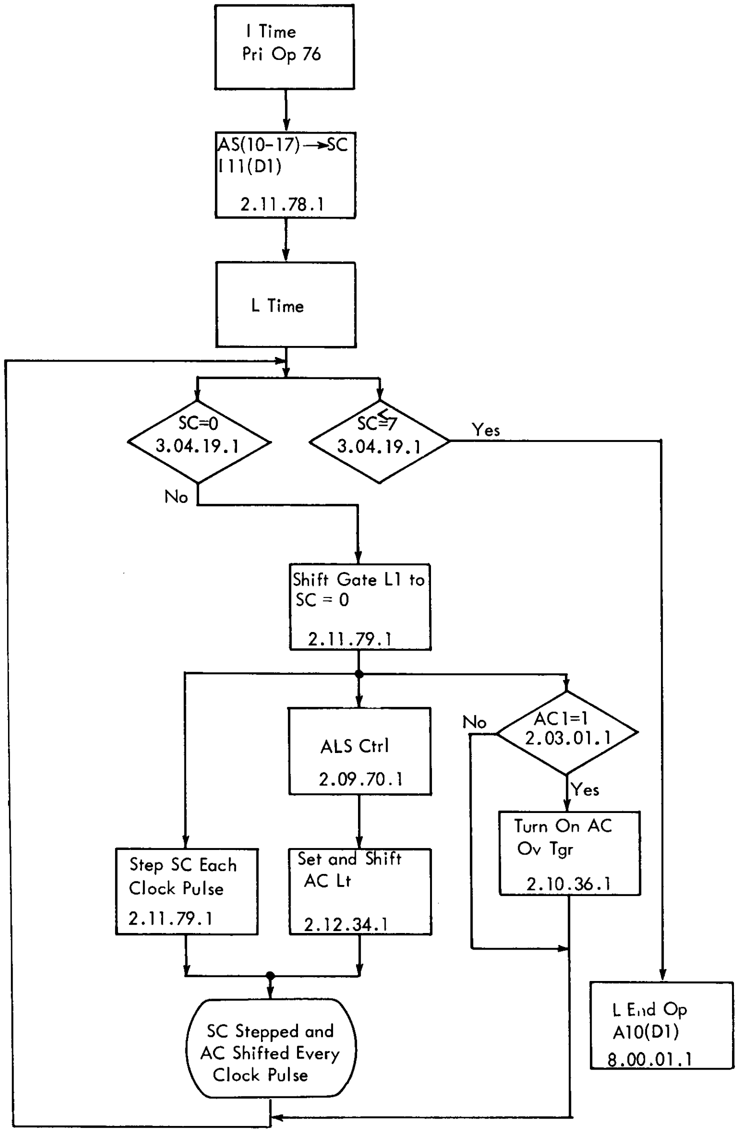


FIGURE 5.3-9. ALS +0767

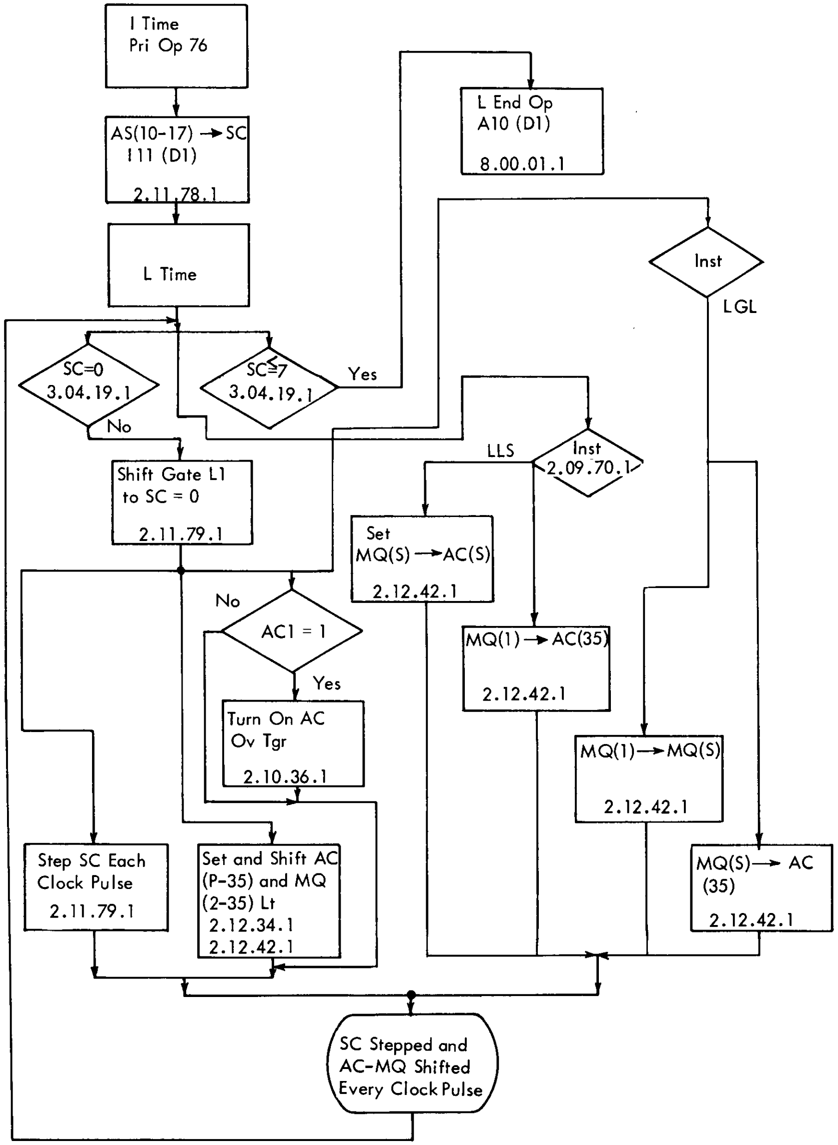


FIGURE 5.3-10. LLS +0763; LGL -0763

Accumulator Right Shift ARS +0771 (I, L...) Figure 5.3-11

This instruction causes the contents of AC (Q-35) to be shifted right the number of places indicated by the address. Bits shifted away from Q are replaced by zeros; bits shifted past position (35) are lost. ARS is similar to ALS except for direction of the shifting.

Long Right Shift LRS +0765 (I, L...) Figure 5.3-12

The contents of the MQ and AC (except the sign positions) are shifted right the number of places indicated by the address. The MQ sign is set to agree with the AC sign. Bits shifted away from Q are replaced by zeros and shifted past AC(35) enter MQ(1). Bits shifted past MQ(35) are lost.

Logical Right Shift LGR -0765 (I, L...) Figure 5.3-12

This instruction shifts the contents of the AC(Q-35) and MQ(S-35) right the number of places designated by the address. Bits shifted out of AC(35) are entered into MQ(S) and from MQ(S) to MQ(1). Bits shifted past MQ(35) are lost. The operation of LGR is the same as LRS except for the handling of the MQ sign.

Rotate MQ Left RQL -0773 (I, L...) Figure 5.3-13

This instruction shifts the contents of the MQ, including the sign, left the number of places designated by the address. Bits shifted out of MQ(1) enter the sign position and from the sign position enter MQ(35).

5.3.03 Fixed-Point Arithmetic Instructions

Fixed-point arithmetic instructions are the basic calculating instructions. Addition is the foundation of all of these instructions, and parts of the ADD instruction will be found in all instructions. All arithmetic instructions use the accumulator.

Clear and Add CLA +0500 (I, E) Figure 5.3-14

The instruction clears the AC (S, Q-35) and loads positions (S, 1-35) with the contents of the storage location indicated by the address. The word that is to be put into the AC is brought from core storage to the SR during the E cycle. In the I cycle of the next instruction, positions (1-35) are gated from the SR through the AD to the AC. The sign position is gated directly from the SR to the AC.

Clear and Subtract CLS +0502 (I, E) Figure 5.3-14

This instruction replaces the contents of the AC with the word from the core storage location indicated by the address. The sign of the word is reversed. Execution of this instruction is the same as the CLA except that the sign of the word is inverted during the E cycle after the word has entered the SR.

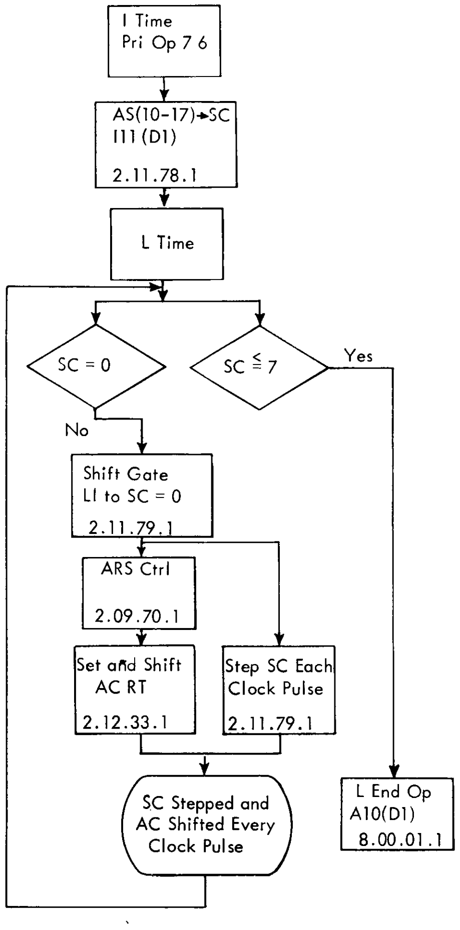


FIGURE 5.3-11. ARS +0771

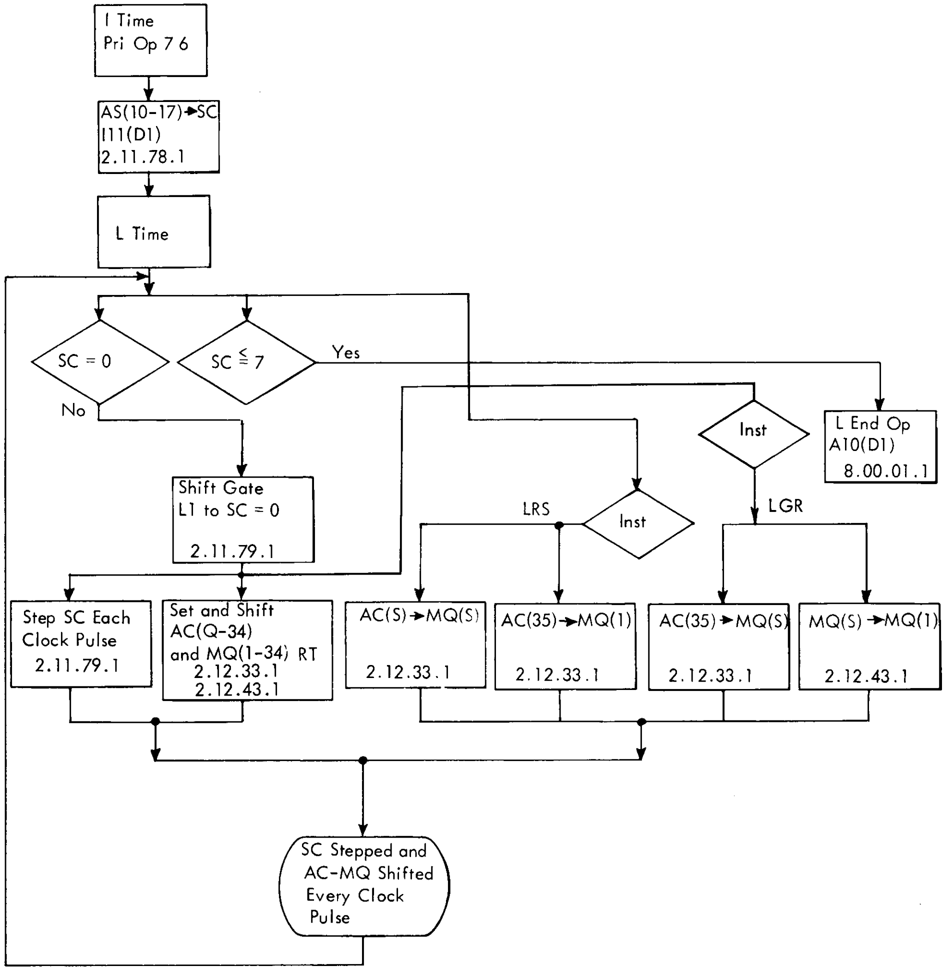


FIGURE 5.3-12. LRS +0765; LGR -0765

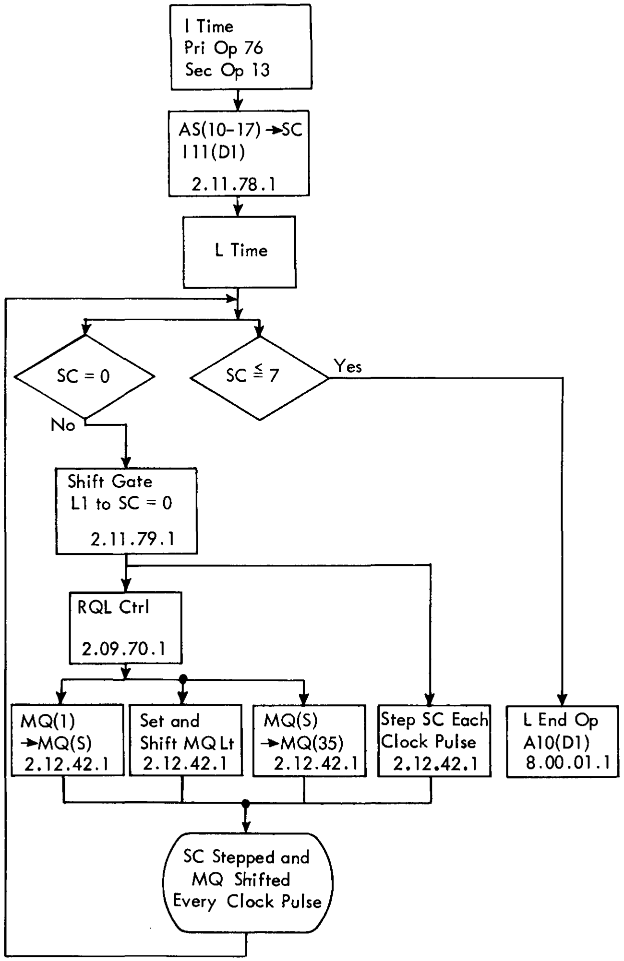


FIGURE 5.3-13. RQL - 0773

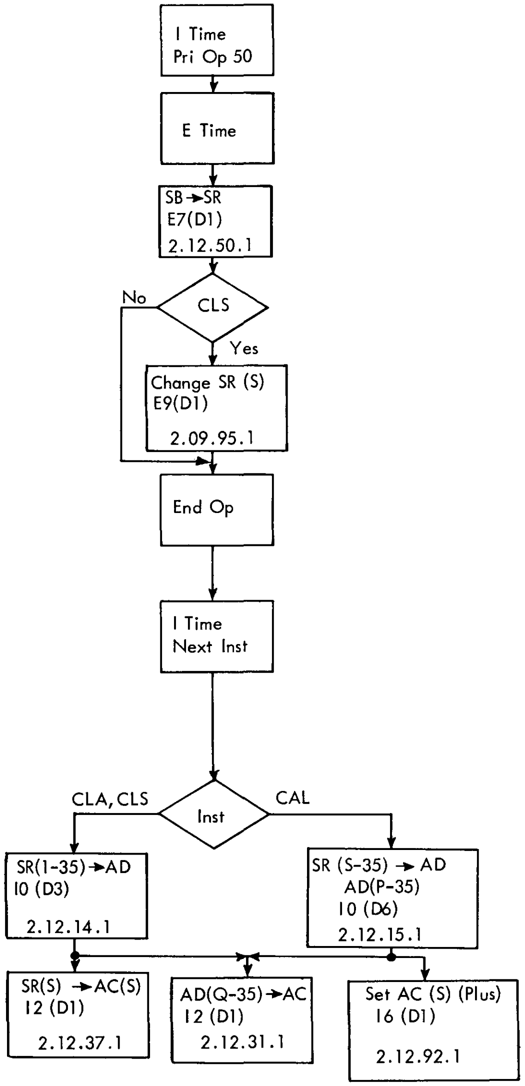


FIGURE 5.3-14. CLA +0500; CLS +0502; CAL - 0500

Clear and Add Logical Word CAL -0500 (I, E) Figure 5.3-14

This instruction clears the AC(S, Q, P, 1-35) and enters the word stored at the location indicated by the address into AC (P, 1-35). This instruction operates the same as CLA except that the (S) of the word is sent to AC(P). The sign of the word in the AC is positive.

Add ADD +0400 (I, E) Figure 5.3-15

This instruction causes the word, stored at the location indicated by the address, to be added algebraically to the contents of the AC. The resulting sum or difference replaces the AC factor.

The following rules of addition are used during the execution of the add instruction:

1. Accumulator and storage register signs alike:
   a. Add true accumulator factor to the storage register factor.
   b. The accumulator sign is unchanged.
2. Accumulator and storage register signs unlike:
   a. Add 1's complement of the accumulator factor to the storage register factor.
      (1) If no end-carry results, complement the accumulator factor and leave the accumulator sign unchanged.
      (2) If an end-carry results, add one to the result and change the accumulator sign.

Except for bringing a word from storage to the SR, most of the execution of this instruction is accomplished during the I cycle of the next instruction. The contents of the AC or the 1's complement of the AC and the contents of the SR are added in the adders. Whether to use true AC or complemented AC is determined by the comparison between the AC and SR signs. Complement addition is used to obtain the difference between the contents of the SR and the contents of the AC.

The difference between the SR and AC contents can be a complement number or a true number. The result will be in complement form if the AC is larger than the SR factor. A true number will result if the AC factor is smaller than the SR factor. During the addition a carry out of AD position Q indicates that the AC factor is smallerand no Q carry indicates that the AC factor is larger. To remember the carry, a Q carry trigger is turned on by a carry out of AD position (Q).

If the result of the complement addition is a true number it is one less than it should be because the 1's complement rather than the 2's complement was used in the addition. Therefore, a one is added to the result in the AC to get the correct difference. If the result of the addition is a complement number, it must be re-complemented to get the correct true number. The sign of the result in the AC is set the same as the sign of the largest original factor, as determined by the status of the Q carry.

<u>Example 1</u>

Signs Alike

-7 + (-6) = -13

-0111 SR(7)

<u>-0110</u> AC(6)

-1101 Result in AC (13)

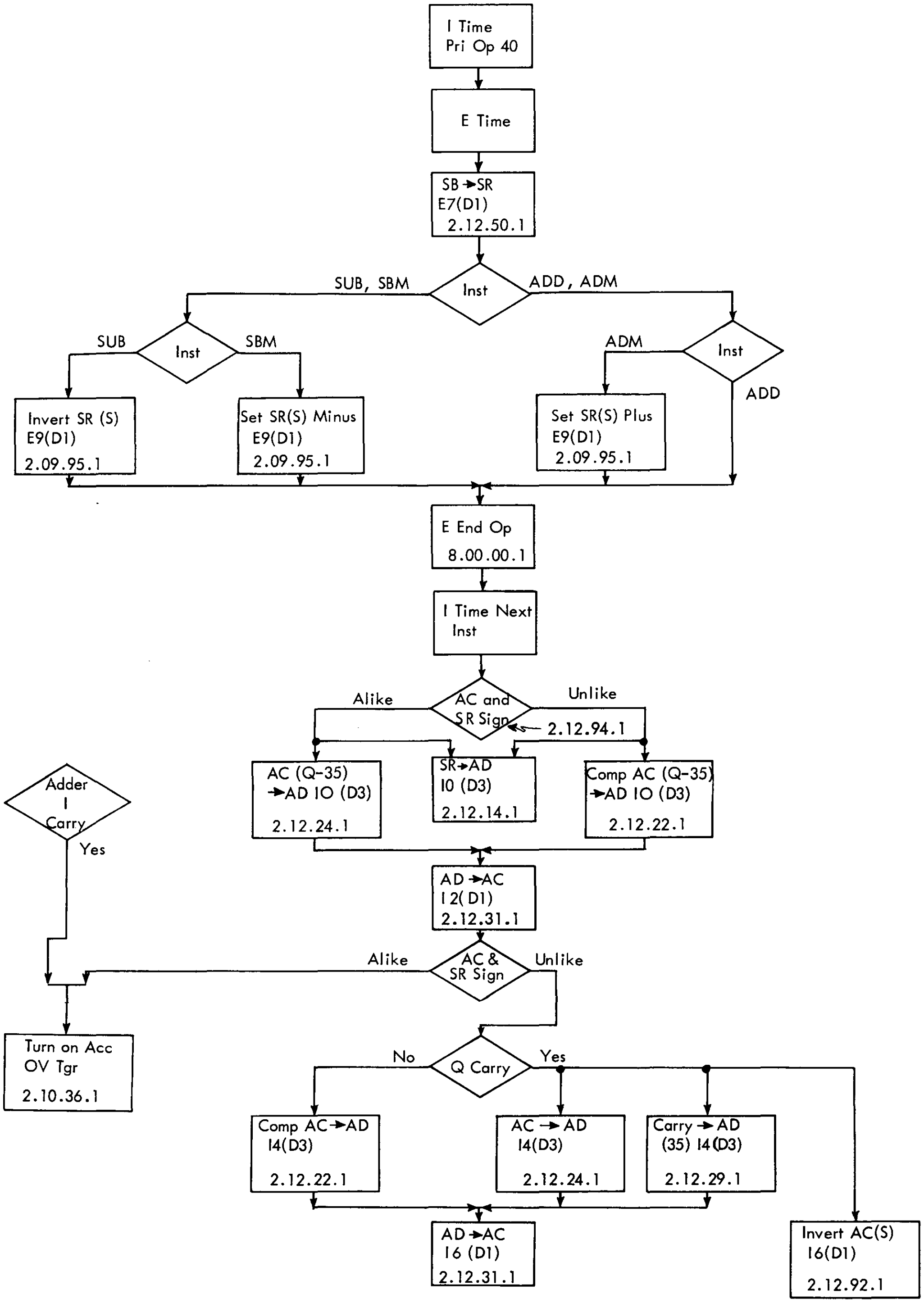


FIGURE 5.3-15. ADD +0400; ADM +0401; SUB +0402; SBM -0400

Example 2

Signs Unlike, AC Smaller

```
    7 + (-6) = +1
+0111  SR(7)
-1001  1's comp of AC(6)
-0000  Q carry
    1  Add one
-0001  Result in AC
+0001  Change sign
```

Example 3

Signs Unlike, AC Greater

```
    6 + (-7) = -1
+0110  SR(6)
-1000  1's comp of AC(7)
-1110  No Q carry, Result in AC
-0001  Comp AC
```

Add Magnitude   ADM +0401 (I, E)   Figure 5.3-15

For this instruction, the word at the storage location indicated by the address is to be considered positive. This positive word is added, algebraically, to the contents of the AC. ADM operates the same as ADD, except that the sign of the word is set positive after it has entered the SR.

Subtract   SUB +0402 (I, E)   Figure 5.3-15

This instruction algebraically subtracts the word at the storage location indicated by the address from the contents of the AC. The execution of SUB is similar to ADD, except that the sign of the word in the SR is inverted.

Subtract Magnitude   SBM -0400 (I, E)   Figure 5.3-15

For this instruction the sign of the storage word indicated by the address is considered negative. This negative word is added algebraically to the contents of the AC. The execution of SBM is the same as ADD, except that the sign of the SR word is forced negative.

Add and Carry Logical Word   ACL +0361 (I, E)   Figure 5.3-16

This instruction adds the 36-bit logical word stored at the location indicated by the address to the contents of the AC(P-35). AC(S) is ignored; AC(Q) is unchanged. A carry out of AD(P) is added to position 35 in the adders. The operation of this instruction is the same as ADD, except for the handling of the (S), (Q), and (P) positions.

Multiply   MPY +0200 (Min I, E)   Figure 5.3-17
(Max I, E, 12L)

This instruction multiplies the word stored at the location indicated by the address by the contents of the MQ. The 35 most significant bits of the 70-bit product replace the AC(1-35), and the 35 least significant bits replace the MQ(1-35). Positions (Q) and (P) of the accumulator are cleared, and the signs of the AC and MQ are set to the algebraic sign of the product.

```
00  0110  Multiplicand (SR)
    1011  Multiplier (MQ)
    0110  Partial Product
   0110     "       "
  0000      "       "
 0110       "       "
 1000010  Product (AC and MQ)
```

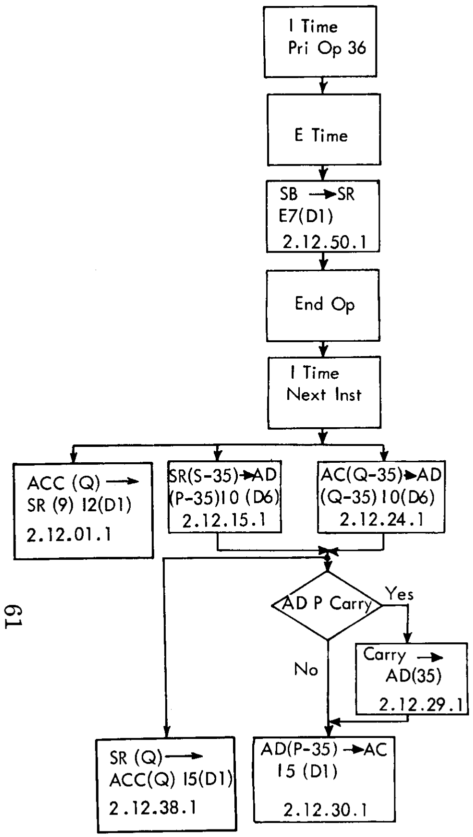


FIGURE 5.3-16. ACL +0361

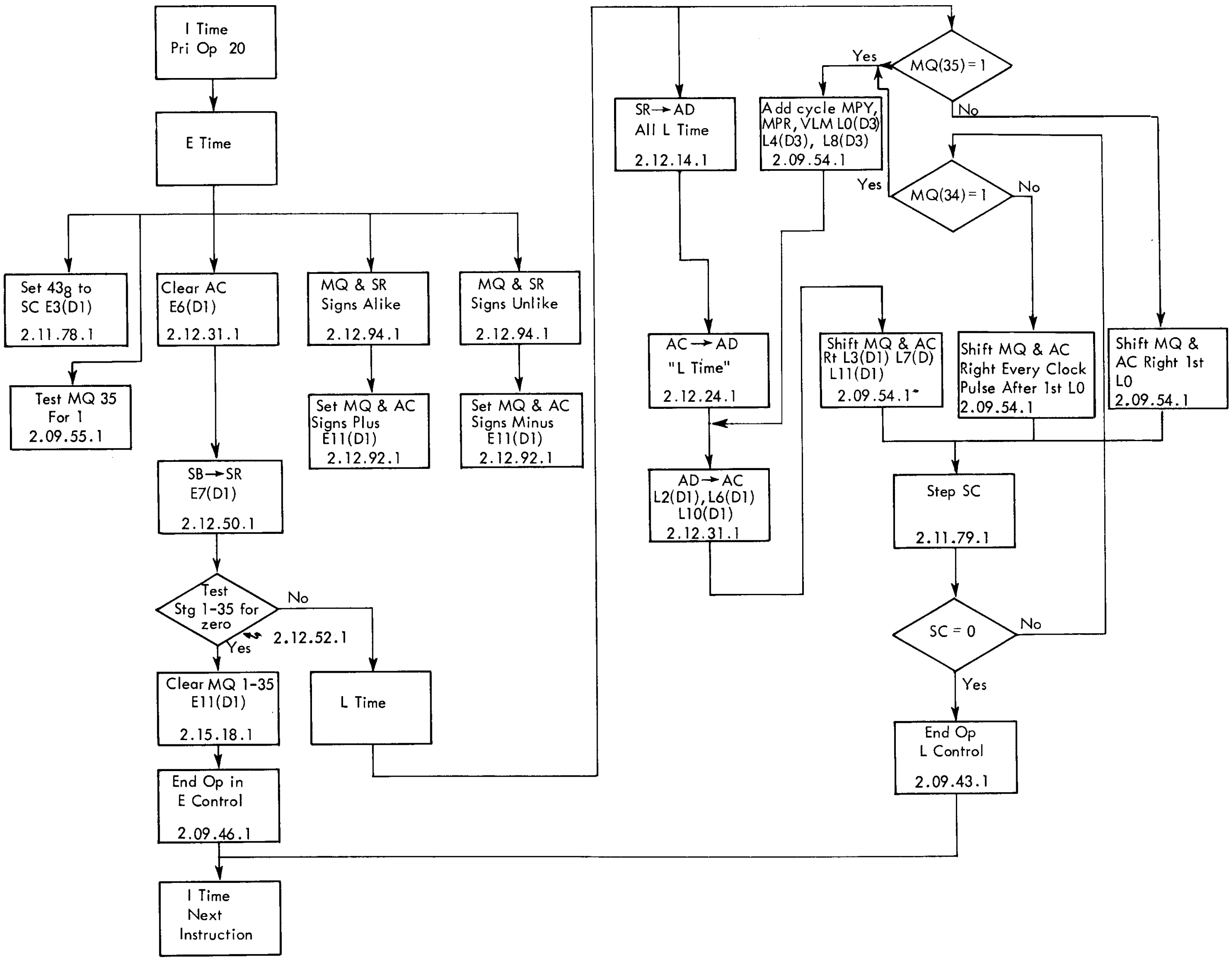


FIGURE 5.3-17. MULTIPLY +0200

In the 7090, multiplication follows this procedure closely. Starting at the right of the multiplier in the MQ, only one position of the multiplier is scanned at a time. In the 7090 there is no means of remembering partial products so they may be added atter each position of the multiplier has been used. Therefore, if the multiplier position being scanned is a one, the multiplicand is added to any previous partial product that is in the AC. In the 7090, multiplication is done as follows:

```
   0110   Multiplicand (SR)
   1011   Multiplier (MQ)
   ----
   0000   Start with zeros in AC
  +0110   Right MQ position scanned
  -----
   0110   Partial Product
 +0110    Next MQ position scanned
 ------
  10010   Partial Product
 0000     Next MQ positions scanned
-------
 010010   Partial Product
+0110     Next MQ position scanned
-------
1000010   Product (AC and MQ)
```

In this example, each time the multiplicand is added to the previous partial product, the multiplicand must be moved to the left to correctly line up the bit positions. In the 7090, the partial product is moved to the right rather than the multiplicand moving to the left. The end result is the same.

Multiplication in the 7090 breaks to two major procedures:

1. Addition of SR and AC contents.
2. Shifting the contents of the AC and MQ right.

Figure 5.3-17 shows a MPY operation in the 7090. During the E cycle, in addition to bringing the multiplicand to the SR, the AC is cleared and the sign is set. This allows the MPY to start with the AC containing zeros. Also during the E cycle the word coming from storage is tested to see if all positions contain a zero. If the multiplicand is zero the product is to be zero. To save machine time, the MQ is cleared and the instruction ends operation; 43_8 (35_{10}) is put into the SC. The SC will be used to indicate when all bits of the MQ have been tested. Also, because the storage word was not zero, the computer is put into L time for multiplication.

Now MQ(35) is tested for a bit. If there is a bit in MQ(35), the SR and AC contents are added and the result is put into the AC. If MQ(35) had no bit, this addition would not have been done because the sum of zero and a number equals the same number. The AC and MQ are shifted right to put the next position of the multiplier in MQ(35) to be tested, and put the partial product in correct alignment with the multiplicand. This type of shift is called a slow-speed shift. High-speed shifting (shifting once for each clock time) will occur if MQ(34) contains zero. This is done to speed the multiplication operation. Each time the AC and MQ are shifted, the shift counter is stepped and, as long as the shift counter does not equal zero, MQ(34) is once more tested and the process starts again. When the SC has been stepped down to zero, the computer signals that multiplication is complete. Because all positions of the multiplier have been scanned and all additions have been completed, MPY then ends operation.

Note that three additions can be done during each cycle of MPY. The maximum number of shifts during an L cycle can be 12, depending on positions (34) and (35) of the MQ. Maximum number of machine cycles required for MPY is 14; minimum is two.

Multiply and Round MPR -0200 (Min I, E) Figure 5.3-18
(Max I, E, 12L)

This instruction executes a multiplication followed by rounding the AC contents. The multiplication is identical to MPY; the rounding is accomplished by adding one to AC (35) if MQ(1) contains a one.

Variable-Length Multiply VLM +0204 (Min I, E) Figure 5.3-19
(Max I, E, 12L)

This instruction operates much the same as MPY, except that the number of multiplier positions to be tested is specified by a number in the decrement portion of the instruction. The difference between MPY and VLM occurs in the E cycle. A count from the decrement portion of the instruction is put into the SC rather than 43_8. Usually this count will be less than 43_8. A count of 60_8 or greater will cause an I/A cycle, and the count field (12-17) will OR with 12-17 of the IA word.

Divide or Halt DVH + 0220 (Min I, E, L) Figure 5.3-20
(Max I, E, 12L)

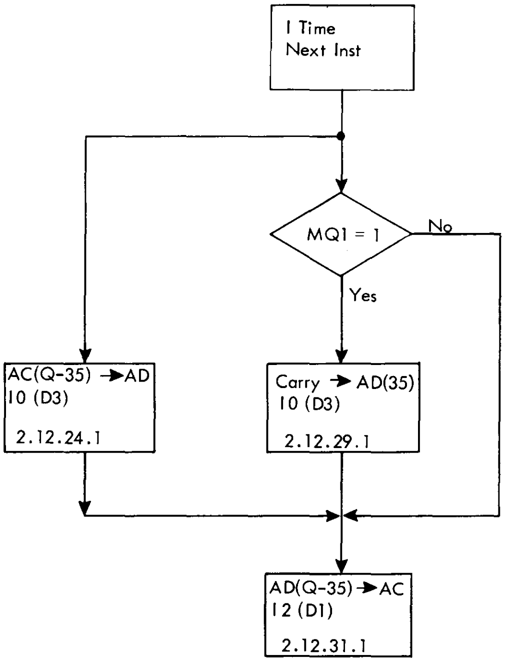
This instruction divides the contents of the AC and MQ, taken together as the dividend, by the word stored at the location specified by the address. A 35-position quotient is developed in the MQ, and the remainder of the dividend is left in the AC. The sign of the MQ is set to the algebraic sign of the quotient, as determined by the SR and AC signs. The sign of the remainder remains the same as the sign of the dividend. The size of the registers restricts the size of the factors to be divided. If the AC portion of the dividend were greater than or equal to the divisor, the quotient would be too large for the MQ. In this case, division cannot take place, and the computer is stopped with the divide check indicator on.

The following demonstrates binary division:

```
             0110     Quotient (MQ)
     1011 ⟌1000010   Dividend (AC and MQ)
           1011
          01011
           1011
           00000
            0000
             0000     Remainder (AC)
```

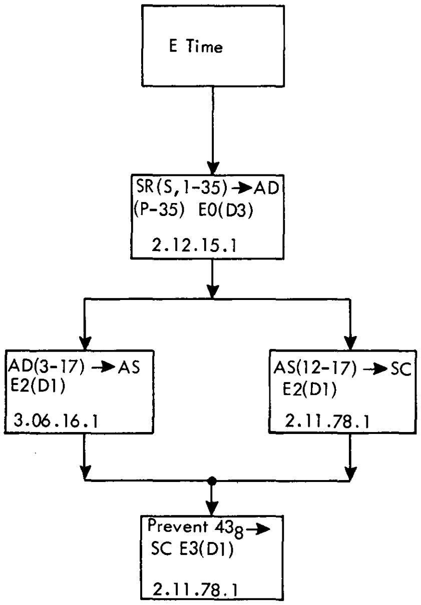
Note that the divisor will go once or not at all into the high order positions of the dividend. Therefore, it is only necessary to determine if the divisor is equal to or smaller than these positions of the dividend. If the divisor is equal to or smaller than the selected positions of the dividend, a one is put in the quotient and the divisor is subtracted from that portion of the dividend. If the divisor is larger than the selected portion of the dividend, a zero remains in the quotient. Another position of the dividend is now taken into account, and the procedure starts again. This continues until all positions in the dividend have been used.

In the 7090, the SR and AC are complement added to determine if a reduction of the high order positions of the dividend is possible. If the reduction is possible, these



Operates the Same as MPY, Except that the Above is also Done During the I Cycle after MPY.

FIGURE 5.3-18. MPR - 0200



Operates the Same as MPY, Except that the Above is also Done During the E Cycle.

FIGURE 5.3-19. VLM + 0204

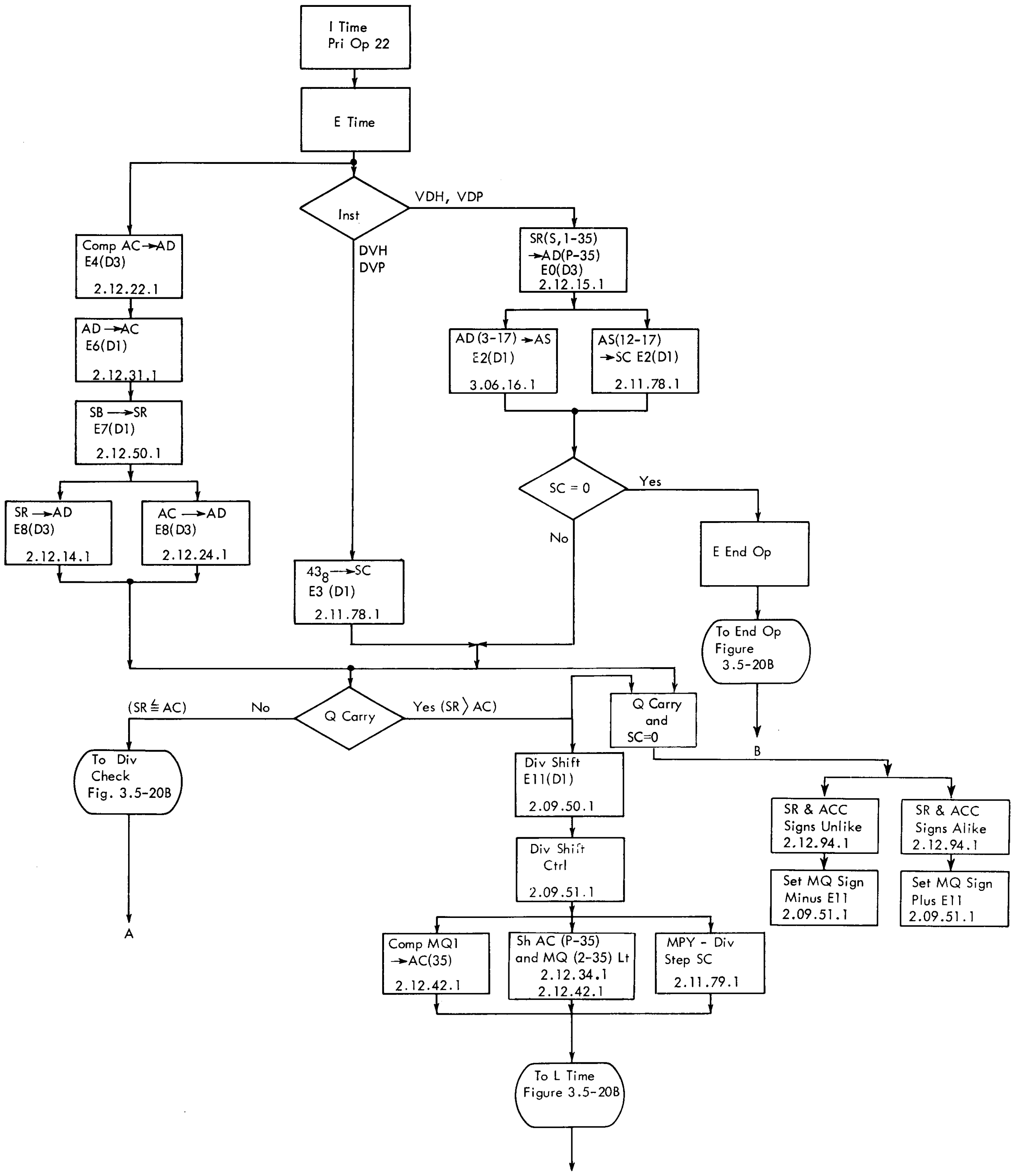


FIGURE 5.3-20A. DVH +0220; DVP +0221; VDH +0224; VDP + 0225

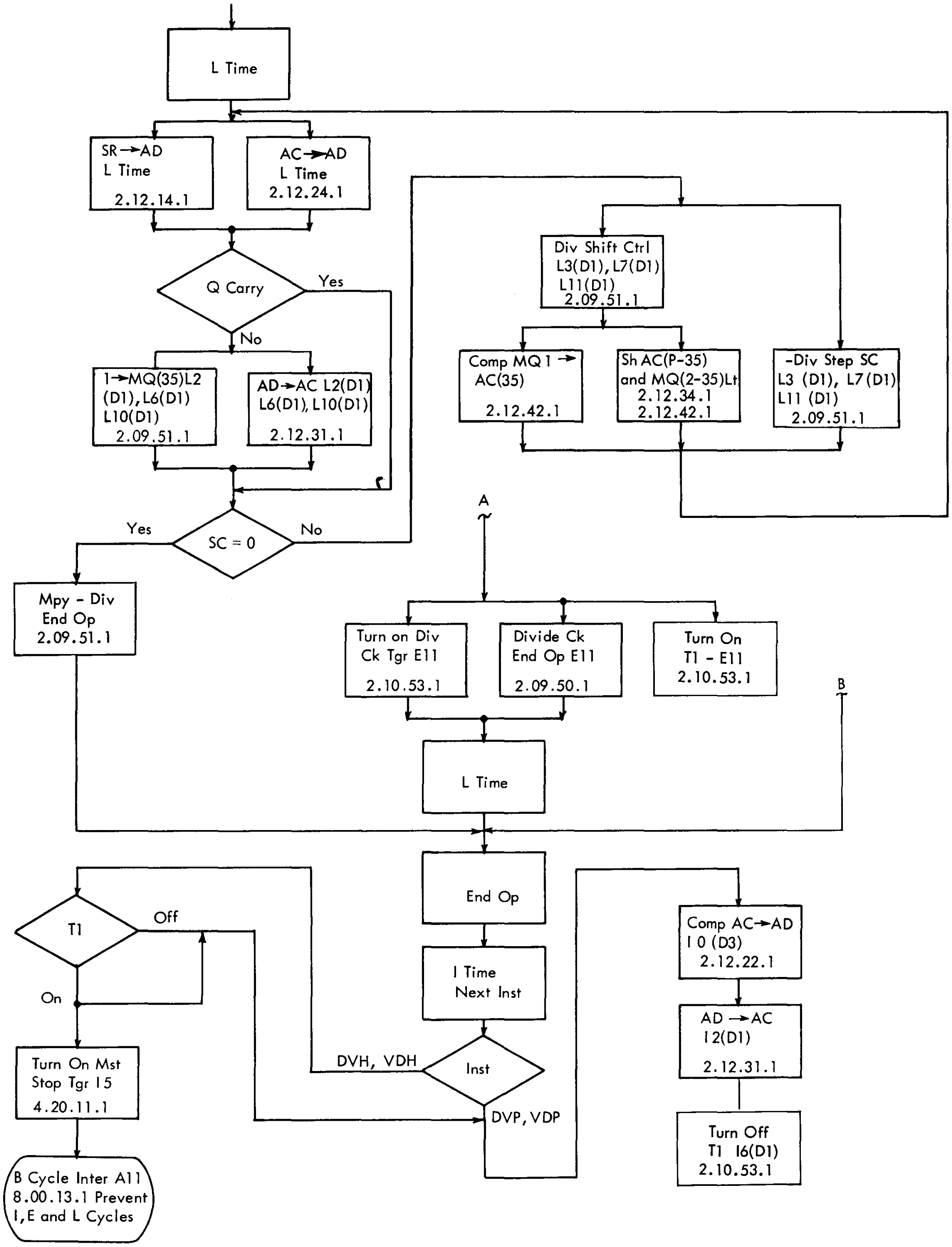


FIGURE 5.3-20B. DVH + 0220; DVP + 0221; VDH + 0224; VDP + 0225

positions of the dividend are reduced by the amount of the divisor and the difference is put into the AC. If a reduction is not possible the AC remains the same. A successful reduction causes a one to be put into MQ(35). After the reduction time, the AC and MQ are shifted left to bring the next position of the dividend into the AC and another reduction is tried. This operation continues until all positions of the MQ portion of the dividend have been moved to the AC.

Figure 5.3-20 shows the sequence of a DVH operation. Because subtraction in the 7090 is accomplished through complement addition, the AC contents are put into complement form early in the E cycle. This complement and the contents of the SR are added to determine if the quotient will be small enough for the MQ register. A Q carry indicates that division is possible and a no Q carry indicates that division is not possible. Also, during the E cycle, 43_8 is put into the SC and this will be used to indicate when all dividend positions have been used.

When there is no Q carry as a result of the E cycle test, the divide check trigger is turned on and the computer is signalled to divide check end operation. An L cycle occurs before actual end operation because the divide check end operation signal comes too late in the E cycle. During the I cycle of the next instruction, the master stop trigger is turned on and this causes B cycle interrupt to be activated. B cycle interrupt prevents I, E, and L cycles.

A Q carry as the result of the E cycle test allows normal divide operation to take place. Toward the end of the E cycle the AC and MQ are shifted left, the SC is stepped and the MQ sign is set. This brings the next position of the dividend into the AC. Note that while shifting from MQ(1) to AC(35), the information is complemented because a complement number is used in the AC.

The computer then goes into L cycles. A reduction is attempted and a 1 is put into MQ(35) if the reduction is successful. Again there is a left shift of the AC and MQ and the SC is stepped. This attempted reduction and shifting procedure continues until the SC equals zero. When the SC=0, divide end operation is actuated.

During the I cycle of the next instruction the AC is again complemented to make the remainder a true number.

Divide or Proceed DVP +0221 (Min I, E, L) Figure 5.3-20
(Max I, E, 12L)

The execution of this instruction is identical to DVH except that the computer is not stopped for the divide check violation, but proceeds to the next instruction. The divide check trigger is not allowed to turn on the master stop trigger during the I cycle of the next instruction.

Variable-Length Divide or Halt VDH +0224 (Min I, E) Figure 5.3-20
(Max I, E, 12L)

This instruction operates the same as DVH, except that the number of reductions to be taken is specified by the count in positions (12-17) of the instruction. The number of positions in the quotient is equal to the count and will be contained in the low-order positions of the MQ. The count should be restricted to a number between 0 and 43_8. A zero count ends operation in E time and prevents the shift at the end of the E cycle. A count of 43_8 will give the same result as DVH. A count greater than 43_8 causes part of the quotient to be shifted into the AC, where it can be altered. A count of 60_8 or greater will cause an I/A cycle, and the count field (12-17) will OR with 12-17 of the IA word.

Variable-Length Divide or Proceed	VDP +0225 (Min I, E) Max I, E, 12L)	Figure 5.3-20

The execution of this instruction is the same as VDH, except that the computer will not stop for a divide check, but will proceed to the next instruction.

Round	RND +0760...0010 (I, L)	Figure 5.3-21

This instruction examines the contents of MQ(1) and if it contains a one, the magnitude of the AC is increased by one. If MQ(1) contains a zero, the AC is unchanged. The MQ is not changed in either case. AC overflow is possible. This is a primary operation 76 instruction. The contents of the AC are sent to the AD and, if MQ(1) contains a one, a one is sent to AD(35) and the output of the AD replaces the contents of the AC.

Clear Magnitude	CLM +0760...0000 (I, L)	Figure 5.3-22

This instruction puts zeros in AC(Q-35). CLM is a primary operation 76 instruction. The operation is accomplished by gating the AD to the AC, with nothing in the adders. The AC(S) remains unchanged.

Complement Magnitude	COM +0760...0006 (I, L)	Figure 5.3-22

The contents of the AC(Q-35) are complemented. Positions containing ones are changed to zeros and positions containing zeros are changed to ones. This instruction is executed by complementing the AC to the AD and replacing the contents of the AC with this complement.

5.3.04 Floating-Point Arithmetic Instructions

The range of numbers anticipated during a calculation may be extremely large, extremely small or, in some cases, unpredictable. Such situations make fixed-point arithmetic difficult to work with for two reasons:

1. The size of the number is limited by the size of the register (35 binary bits or 10 decimal digits).
2. The programmer must keep track of the point in all numbers throughout the calculation.

To meet the needs of large numbers and to automatically keep track of the point, an alternative set of arithmetic instructions, called floating-point arithmetic instructions, are available.

Floating-point arithmetic is merely arithmetic dealing with numbers in exponential form. The numbers 5.6×10^3 or 56000×10^{-4} have a familiar form. The numbers are made of three parts; a fraction (5.6 or 56000), an exponent (3 or -4), and a base (10).

Floating-point numbers in binary are similar to decimal floating-point numbers. The major difference is the base. Numbers in the 7090 use 2 as a base, because it is a binary computer. The other difference is one of terms. Instead of a decimal point, we will call it a binary point.

The following chart gives a comparison of fixed-point binary numbers and floating-point binary.

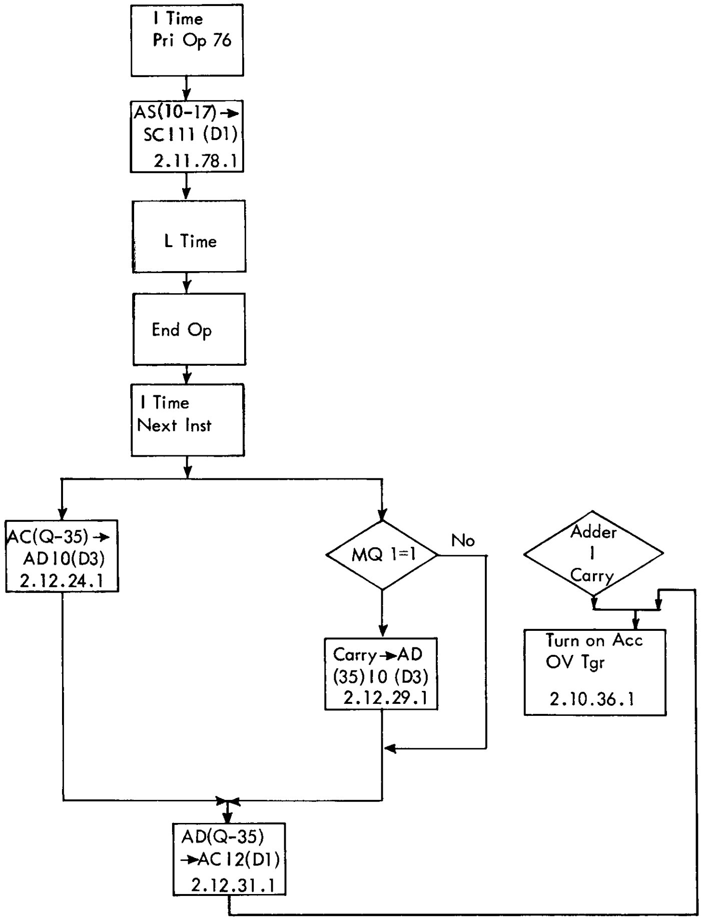


FIGURE 5.3-21. RND +0760...0010

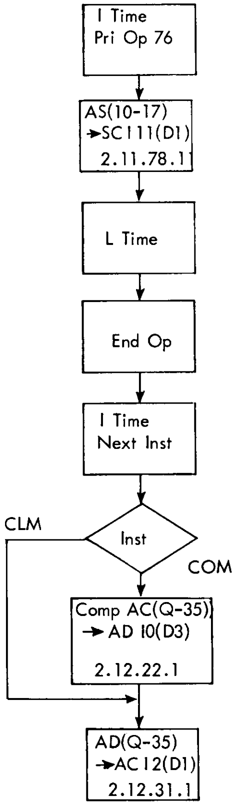


FIGURE 5.3-22. CLM +0760...0000; COM +0760...0006

Fixed-Point Binary	Floating Point
(4) 000100	$.1 \times 2^{011}$
(11) 001001	$.1001 \times 2^{100}$

Because the 7090 works in binary, all floating-point numbers will be to the base 2. Therefore, to represent a floating-point number in the computer, there is no need to carry the base along with the number. This cuts our need to representing the fraction and the exponent. The exponent is represented in positions (1-8) of the word and is now called the characteristic. The fraction is contained in positions (9-35). The binary point is to the left of the 9 bit. The sign position is used to sign the fraction. Word layout takes this format:

```
S   1 ----------- 8   .9---------------------35

      Characteristic            Fraction
```

The value of the number in the characteristic field signifies the exponent and its sign. The characteristic is derived by adding 200_8 to the exponent. If the characteristic is 200_8 the exponent is zero. If the number is 201 to 377, the exponent is positive. If it is 0 to 177, the exponent is negative. The following chart gives examples of exponential numbers and their floating point representation:

Exponential		Floating Point	
Binary	S	1 - 8	9 - 35
$+.1 \times 2^{011}$	+	10000011	10000----0
$-.01 \times 2^{001}$	-	10000001	0100-----0
$+.1 \times 2^{-011}$	+	01111101	1000-----0

Normal and Unnormal Forms

A floating-point number is said to be in normal form when the digit immediately to the right of the point is a significant bit (1). If the number is a zero, it is said to be in unnormal form. The exception to this rule is a normal zero: a normal zero is a floating-point number whose characteristic and fraction are both zero.

To go along with the two types of numbers, the instructions are also divided into two categories, normal and unnormal. The difference in computer operation is that the normal instructions always attempt to produce a normal answer and the unnormal instructions do not.

Arithmetic of Floating Point

Addition of floating-point numbers is done by adding the fractions of floating-point numbers which have equal characteristics. The characteristics are set equal preceding the addition by placing the number with the smallest characteristic in the AC. The fraction is then shifted right, and for each right shift one is added to the characteristic. When the characteristic of the AC equals the characteristic of the SR, shifting stops, and the fractions of the AC and SR are added. Bits shifted out of AC(35) enter MQ(9). The sum appears in the AC and forms the most significant part of the answer. The least significant part is the bits that were shifted into the MQ. The MQ characteristic is set 27_{10} less than the AC characteristic to complete an unnormalized floating add. If it were a normalizing instruction, a check would be made to see if a 1 were in AC position nine. If AC(9) does contain a 1, the operation would be complete; if not, the AC would shift left until a one did appear in position nine. Shifting increases the number, so to keep it the same, the characteristic is reduced by the number of left shifts taken. Floating-point subtraction works the same except that the fractions are subtracted.

Floating-point divide is accomplished by dividing the fraction of the dividend by the fraction of the divisor and subtracting the characteristics. During the subtraction of the characteristics, the 200_8 that is added to all exponents is lost. Therefore, before the answer is final, 200_8 must be added to the quotient characteristic.

Floating multiply is accomplished by multiplying the fraction in the SR by the fraction in the MQ. The exponents in multiply are added, so in a floating multiply, the computer adds the characteristics. Because 200_8 had been added to each exponent originally, the characteristic is increased by 400_8 after the addition. Before the answer is final, 200_8 must be subtracted from the characteristic. The most significant part of the product is in the AC and the least significant part in the MQ.

Sign control is as follows:

Multiplication and Division	Signs of factors alike; answer plus Signs of factors unlike; answer minus
Addition	Answer always has sign of the largest factor
Subtraction	After sign of SR is inverted, answer always has sign of the largest factor

Floating-Point Tally Counter

The floating-point add, multiply, and divide type instructions require an I, an E, and several L cycles. Execution of the instruction is accomplished during the L cycles. To differentiate between the different types of L cycles, there is a tally counter. This is a five stage counter whose output is added with L time to direct the various phases in the execution of the instruction. When the required operations are complete for first step L time, the tally counter is stepped to produce second step L time. This causes the computer to go into the next phase in the execution of the instruction. The operation continues to the next step, and so on. All five steps of the tally counter are not used for every instruction; each instruction uses as many as it requires.

Floating Add FAD +0300 (Min I, E, 4L) Figure 5.3-23
(Max I, E, 13L)

This instruction adds, algebraically, the floating-point number stored at the location indicated by the address and the floating-point number contained in the accumulator. The most significant portion of the result appears as a normalized floating-point number in the accumulator. The least significant portion of the result appears in the MQ as a floating-point number with a characteristic 27_{10} less than the characteristic of the number in the accumulator. The signs of the AC and MQ are set to the sign of the larger factor. If both the resulting MQ and AC fractions are zero, the registers will be reset to contain normal zeros with the signs corresponding to the original factor having the smaller characteristic. If the characteristics were equal, the resulting signs will correspond to the original AC sign.

The result in the AC is always normalized, whether the original factors were normal or not. No attempt is made to normalize the MQ.

First Step L Time. As first step L time is entered, one of the floating-point words to be added is in the AC and the other is in the SR. The objectives of 1st step L time are to:

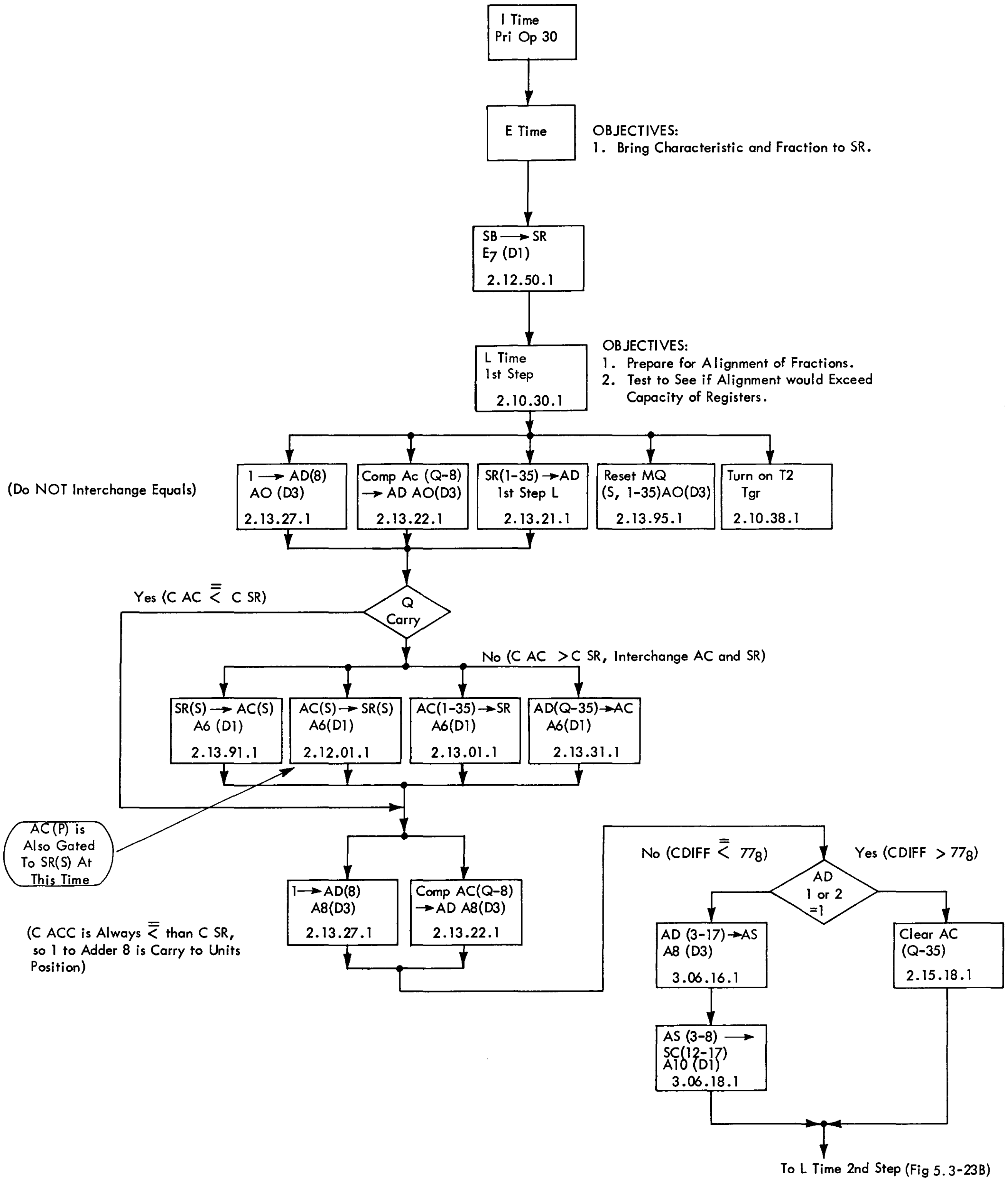
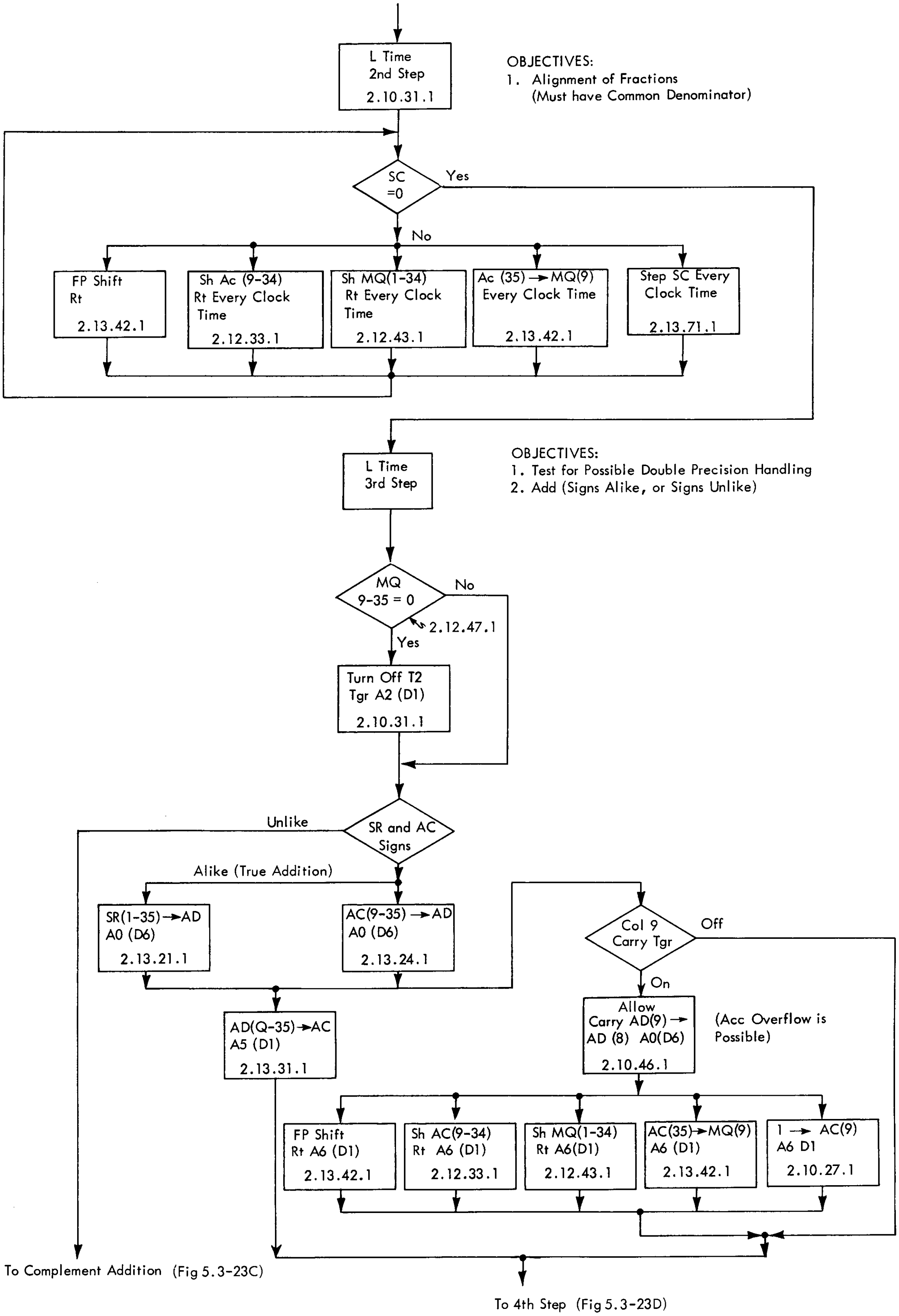


FIGURE 5.3-23A. FAD +0300



OBJECTIVES:
1. Alignment of Fractions
(Must have Common Denominator)

OBJECTIVES:
1. Test for Possible Double Precision Handling
2. Add (Signs Alike, or Signs Unlike)

FIGURE 5.3-23B. FAD +0300

COMPLEMENT ADDITION

1. 9 Carry, Add 1 to Units Position, Change the Sign of Acc (SR was > than Acc).
2. No 9 Carry, Re-complement and Leave the Sign. (SR was $\overline{\lessgtr}$ than Acc)

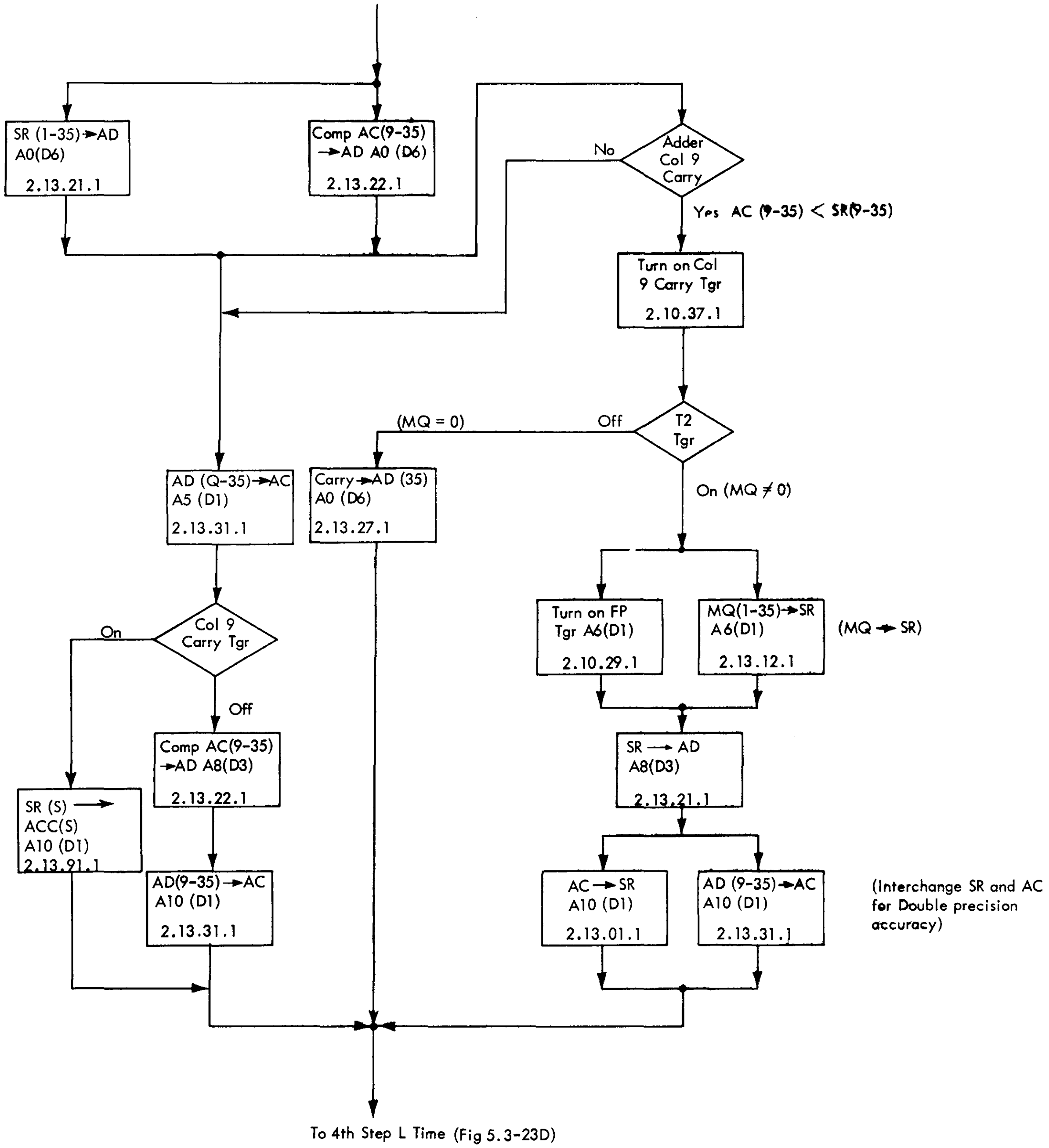

SR (1-35)→AD A0(D6) 2.13.21.1
Comp AC(9-35) →AD A0 (D6) 2.13.22.1
No
Adder Col 9 Carry
Yes AC (9-35) < SR(9-35)
Turn on Col 9 Carry Tgr 2.10.37.1
(MQ = 0)
Off
T2 Tgr
On (MQ ≠ 0)
AD (Q-35)→AC A5 (D1) 2.13.31.1
Carry→AD (35) A0 (D6) 2.13.27.1
On
Col 9 Carry Tgr
Off
Comp AC(9-35) →AD A8(D3) 2.13.22.1
SR (S) → ACC(S) A10 (D1) 2.13.91.1
AD(9-35)→AC A10 (D1) 2.13.31.1
Turn on FP Tgr A6(D1) 2.10.29.1
MQ(1-35)→SR A6(D1) 2.13.12.1
(MQ → SR)
SR → AD A8(D3) 2.13.21.1
AC→SR A10 (D1) 2.13.01.1
AD (9-35)→AC A10 (D1) 2.13.31.1
(Interchange SR and AC for Double precision accuracy)
To 4th Step L Time (Fig 5.3-23D)

FIGURE 5.3-23C. FAD +0300

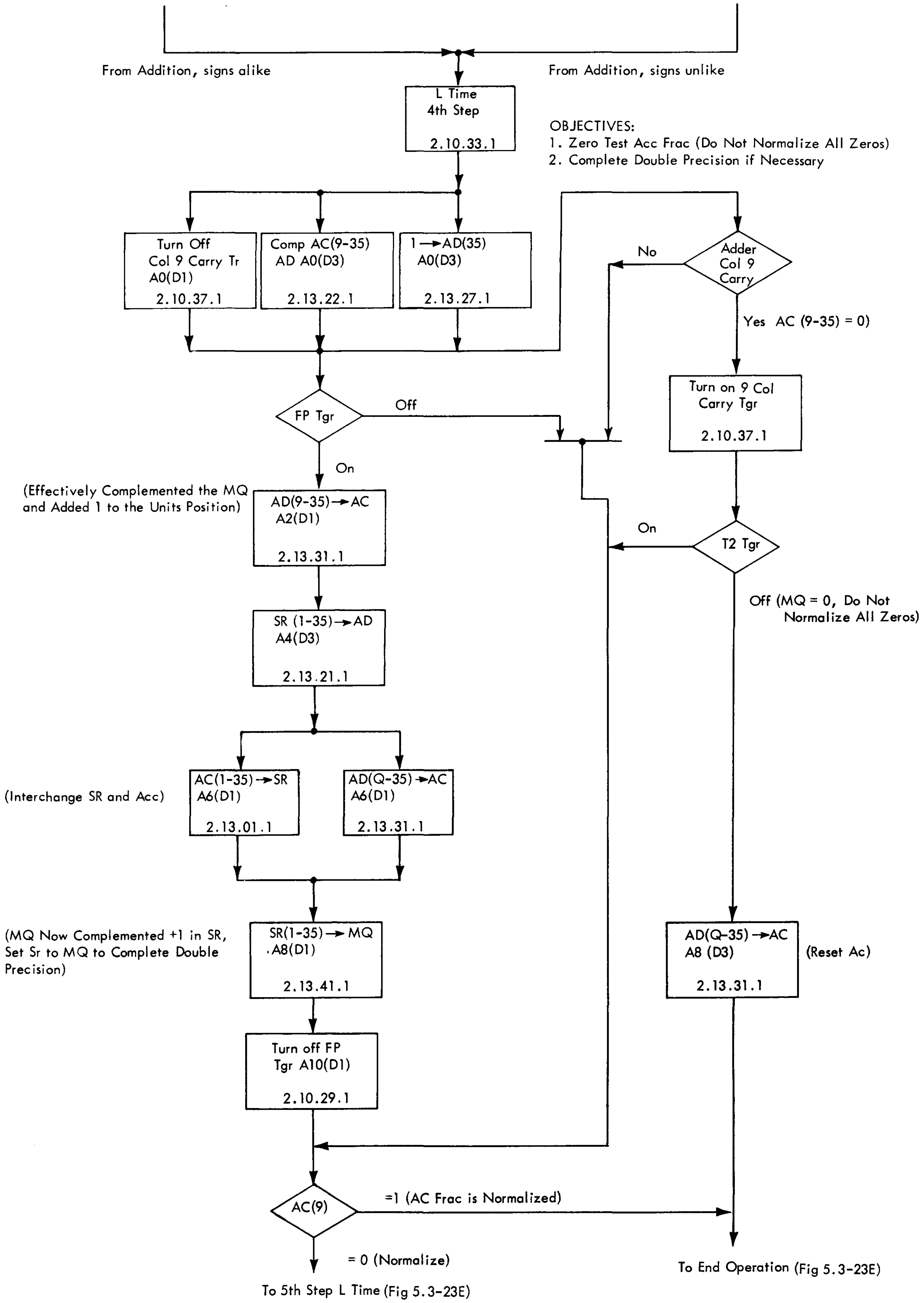

From Addition, signs alike
From Addition, signs unlike
L Time 4th Step 2.10.33.1
OBJECTIVES:
1. Zero Test Acc Frac (Do Not Normalize All Zeros)
2. Complete Double Precision if Necessary
Turn Off Col 9 Carry Tr A0(D1) 2.10.37.1
Comp AC(9-35) AD A0(D3) 2.13.22.1
1 → AD(35) A0(D3) 2.13.27.1
Adder Col 9 Carry
No
Yes AC (9-35) = 0)
Turn on 9 Col Carry Tgr 2.10.37.1
FP Tgr
Off
On
(Effectively Complemented the MQ and Added 1 to the Units Position)
AD(9-35) → AC A2(D1) 2.13.31.1
T2 Tgr
On
Off (MQ = 0, Do Not Normalize All Zeros)
SR (1-35) → AD A4(D3) 2.13.21.1
(Interchange SR and Acc)
AC(1-35) → SR A6(D1) 2.13.01.1
AD(Q-35) → AC A6(D1) 2.13.31.1
(MQ Now Complemented +1 in SR, Set Sr to MQ to Complete Double Precision)
SR(1-35) → MQ .A8(D1) 2.13.41.1
AD(Q-35) → AC A8 (D3) 2.13.31.1
(Reset Ac)
Turn off FP Tgr A10(D1) 2.10.29.1
AC(9)
=1 (AC Frac is Normalized)
= 0 (Normalize)
To 5th Step L Time (Fig 5.3-23E)
To End Operation (Fig 5.3-23E)

FIGURE 5.3-23D. FAD +0300

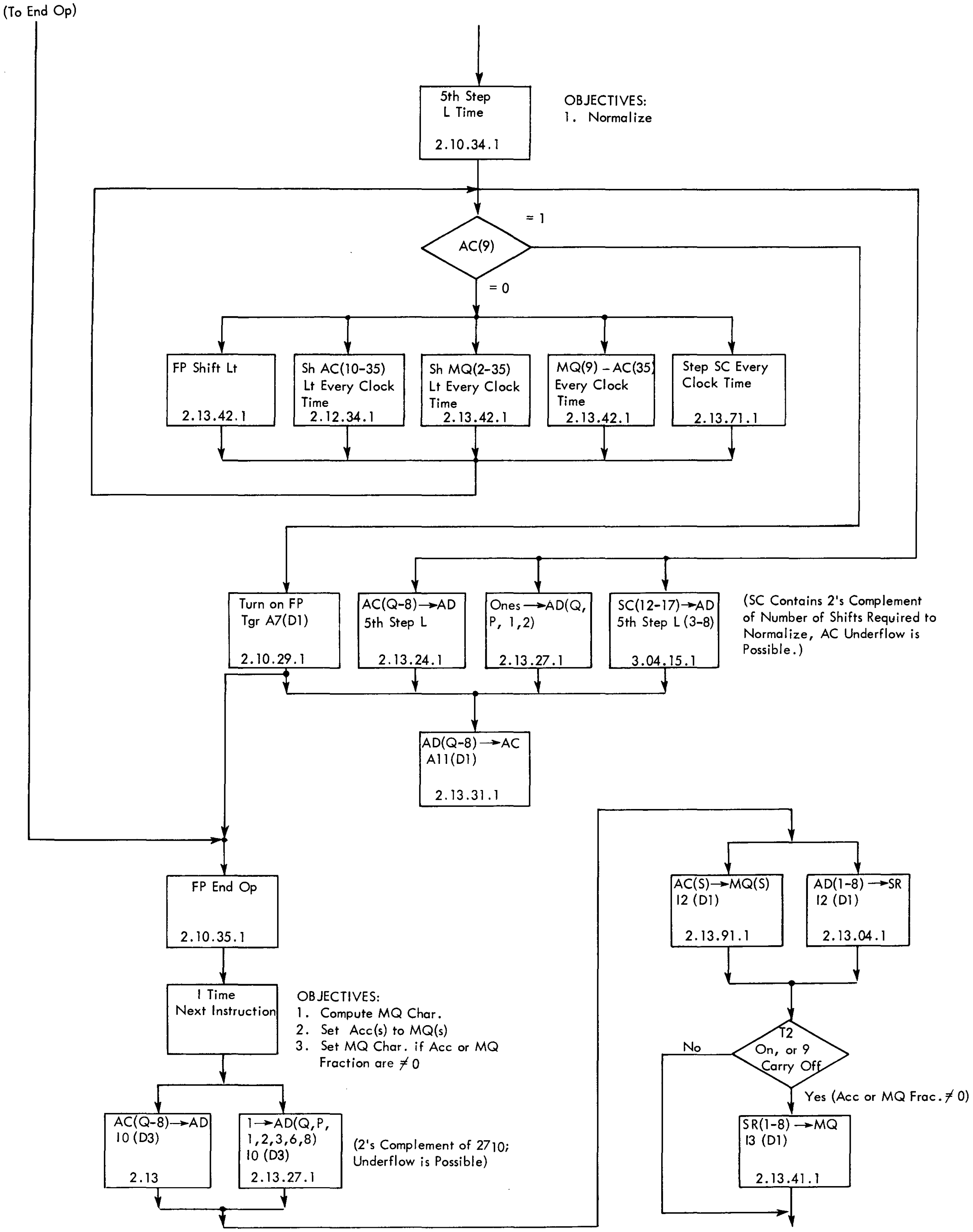


FIGURE 5.3-23E. FAD +0300

1. Put the word with the smallest characteristic in the AC.
2. Determine the difference between AC and SR characteristics.
   a. If the difference is less than 100_8, put the difference in the SC.
   b. If the difference is greater than 77_8, clear the AC.

To determine which characteristic is smaller, the 2's complement of the AC characteristic is added to the SR characteristic. Also, during this time the MQ is reset and T2 trigger is turned on. These are the initial starting conditions for T2 and the MQ. Trigger T2 will be used later, during the execution of the instruction, if the MQ contains a zero or not.

A Q carry, because of the complement addition of the AC characteristic to the SR characteristic, means that the SR characteristic is larger. No Q carry means that the AC characteristic is larger. Therefore, the word in the SR is moved to the AC and the word in the AC is moved to the storage register.

Again, the 2's complement of the AC characteristic is added to the SR characteristic to determine the exact difference between the characteristics. A characteristic difference of less than 100_8 causes the difference to be put into the SC and a difference greater than 77_8 causes the AC to be cleared. The AC is cleared for a difference greater than 77_8 because, due to this difference, the information eventually would be shifted out of the AC and MQ anyway. Shifting the MQ and AC fractions right makes possible setting the characteristics of the SR to the characteristic of the AC. It takes 66_8 shifts to move a bit from AC fraction position (9) out through MQ fraction position (35). The characteristic difference is checked for 77_8 rather than 66_8 because this is easier to do in the computer. The AC is cleared rather than let shifting take place because machine time is saved.

Second Step L Time. Second step L time is used to shift the AC and MQ fractions right a number of places equal to the difference between the AC and MQ characteristics. The principle here is the same as moving the point to the left and increasing the exponent by one for each position that the point is moved. The changing of the characteristic is done later. However, the fraction is adjusted for this change during second step. The AC and MQ fractions are shifted right until the SC is stepped down to zero.

Third Step L Time. Third step L time occurs when the SC equals 0. The objectives of the third step are to:

1. Set the AC characteristic equal to the SR characteristic.
2. Add, algebraically, the AC fraction to the SR fraction.
3. Make AC sign equal to the algebraic sign of the result of the addition.

The AC characteristic is set equal to the SR characteristic as a result of the addition because the AC(9-35) or the 1's complement of AC(9-35) is sent to the adders with SR (1-35). Therefore, when AD(Q-35) are gated back to the AC, the correct characteristic is set into the AC along with the result of the addition.

When binary fractions with like signs are added, there is a possibility of a carry out of the high order fraction position. This is the same as a carry across a decimal point. The 7090 has no register position to hold this carry (1), therefore, the fraction and characteristic of the sum must be adjusted. This is done by moving the AC and MQ fractions one position to the right, putting the one in the high order position of the AC fraction, and increasing the characteristic by one. The characteristic is automatically increased by one during the addition because the carry out of AD(9) goes to the AD(8).

When signs are unlike, a complement addition occurs. A column 9 carry resulting from this addition means the AC fraction was smaller than the SR fraction and the result placed in the AC is a true number. Assuming the MQ to be zero, this true number is 1 less than the sum, because the 1's complement of the AC fraction was used for addition. Therefore, a one is added to the AC fraction. If the MQ contains something other than zero, the result of the addition is 1 minus MQ fraction less than it should be. Therefore, one is not added to the AC fraction, but the MQ is subtracted from this one. This is done by, in effect, complementing the MQ. Because no provisions are in the circuits to complement the MQ , the MQ contents are moved to the AC through the SR. The result of the addition is moved to the SR when the MQ contents are brought into the AC.

Fourth Step L Time. Fourth step L time is used to:

1. Complement the AC fraction.
   a. AC contains original MQ fraction if the MQ was not zero.
   b. If original MQ fraction was zero, it was not moved to the AC; therefore, this complementing is used to test the AC fraction for a zero result of the addition.
2. If the MQ and AC are zero, clear the AC to give a zero characteristic.
3. End operation if normalizing is not needed.
   a. If AC and MQ are zero.
   b. If AC(9) contains a one.

After complementing the AC fraction with the FP trigger on (MQ was not zero), the complement is sent to the MQ and the result of the addition is moved from the SR back to the AC. Again available circuits must be used to get the complement to the MQ. Therefore, the AC is gated to the SR and the SR is gated to the MQ.

Fifth Step L Time. Fifth step L time is used for normalizing the result and ending operation. Normalizing is done by (1) shifting the AC and MQ fractions left until AC(9) receives a one, (2) counting the number of shifts, and (3) subtracting the number of shifts from the characteristic. The SC is used to count the number of shifts. Since the SC is a count down counter which starts from zero, the 2's complement of the number of shifts will be in the SC when shifting is stopped. Adding this complement to the characteristic of the AC gives the correct adjusted characteristic.

I Time. I time of the next instruction sets the MQ(S) and characteristic. The AC(S) is gated to the MQ(S) so both signs are the same.

For an MQ and AC fraction that is not zero, the characteristic of the MQ is set 27_{10} (33_8) less than the AC characteristic. This is done by adding the 2's complement of 33_8 to the AC characteristic and putting the result into the MQ characteristic.

Unnormalized Floating Add UFA -0300 (Min I, E, 3L)
(Max I, E, 8L)

This instruction algebraically adds two floating-point numbers in the same manner as FAD. The result, however, is not normalized. The sequence of operation for UFA is the same as FAD with these exceptions:

1. For like SR and AC signs, UFA ends operation at the end of the third step because a normalizing step is not needed.
2. UFA ends operation at the end of the third step if AC and SR signs are unlike and there is no column 9 carry as a result of the addition. This is because the correct unnormalized result is in the AC and MQ at the end of the 3rd step.

Floating Add Magnitude FAM +0304 (Min I, E, 4L)
(Max I, E, 13L)

This instruction algebraically adds the magnitude of the floating-point number stored at the location designated by the address to the floating-point number in the accumulator. The sequence of operations is identical to that of FAD except that the SR sign is set positive during the E cycle. This control line is on Systems 2.09.95.1.

Unnormalized Add Magnitude UAM -0304 (Min I, E, 3L)
(Max I, E, 8L)

This instruction operates the same as FAD, with the exceptions pointed out in UFA and FAM.

Floating Subtract FSB +0302 (Min I, E, 4L)
(Max I, E, 13L)

This instruction algebraically subtracts the floating point number stored at the location indicated by the address from the floating-point number in the accumulator. FSB operates the same as FAD, except that the sign of the word in the SR is inverted during the E cycle. This is shown on Systems 2.09.95.1.

Unnormalized Floating Subtract UFS -0302 (Min I, E, 3L)
(Max I, E, 7L)

This instruction algebraically subtracts two floating-point numbers without normalizing the result. Execution is the same as FAD with the exceptions noted in UFA and FSB.

Floating Subtract Magnitude FSM +0306 (Min I, E, 4L)
(Max I, E, 13L)

This instruction algebraically subtracts the magnitude of the floating-point number stored at the location indicated by the address from the floating-point number in the accumulator. Execution of this instruction is the same as FAD, except that the sign of the word in the SR is forced minus during the E cycle. This is shown on Systems 2.09.95.1.

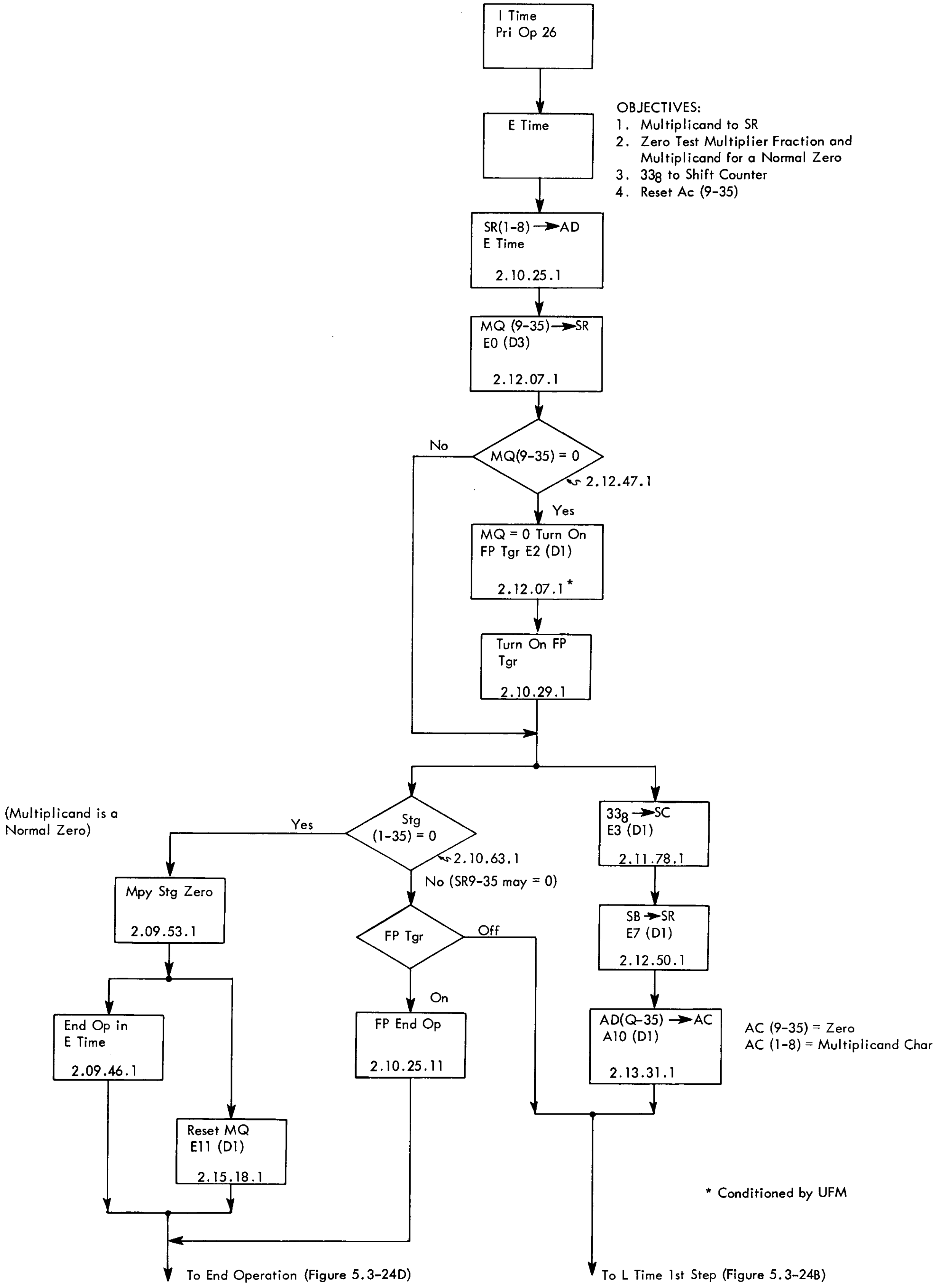
Unnormalized Subtract Magnitude USM -0306 (Min I, E, 3L)
(Max I, E, 9L)

This instruction operates the same as FSM, with the exceptions explained under UFA.

Floating Multiply FMP +0260 (Min I, E) Figure 5.3-24
(Max I, E, 11L)

This instruction multiplies the floating-point number stored at the location designated by the address by the floating-point number stored in the MQ. The product appears as two floating-point numbers: the most significant part in the accumulator, and the least significant part in the MQ. The signs of both registers are set to the algebraic sign of the product. If the multiplicand is zero, the product will be two normal zeros with proper algebraic signs. If the AC fraction or the multiplier fraction is zero, the AC is reset to a normal zero and no characteristic is assigned to the MQ. If the AC fraction is not zero, the MQ is assigned a characteristic 27_{10} less than the AC characteristic.

Remember that when multiplying numbers using exponents, the fractions are multiplied and the exponents are added. Multiplying the fractions is done much the same as in a MPY operation, by adding and shifting. Adding the characteristic is not enough

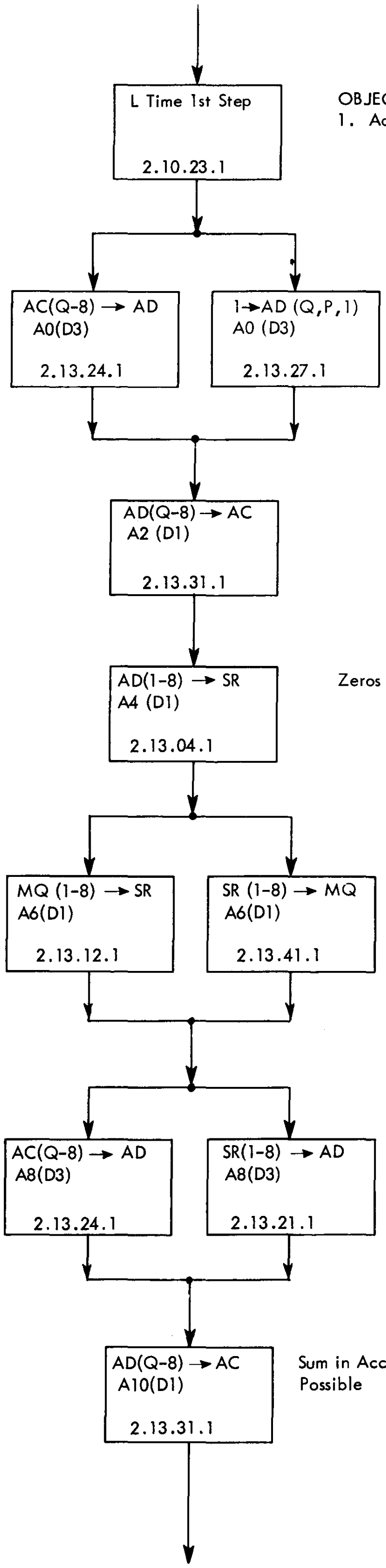


OBJECTIVES:
1. Multiplicand to SR
2. Zero Test Multiplier Fraction and Multiplicand for a Normal Zero
3. 33_8 to Shift Counter
4. Reset Ac (9-35)

AC (9-35) = Zero
AC (1-8) = Multiplicand Char

* Conditioned by UFM

FIGURE 5.3-24A. UFM -0260; FMP +0260



OBJECTIVES:
1. Adjust Characteristics

Subtract Base 200_8 From Multiplicand Char

Zeros to SR (1-8)

Multiplier Char to SR, Zeros to MQ(1-8)

Addition of Multiplicand and Multiplier Characteristics

Sum in Accumulator, Overflow or Underflow is Possible

To L Time 2nd Step (Fig 5.3-24C)

FIGURE 5.3-24B. UFM - 0260; FMP + 0260

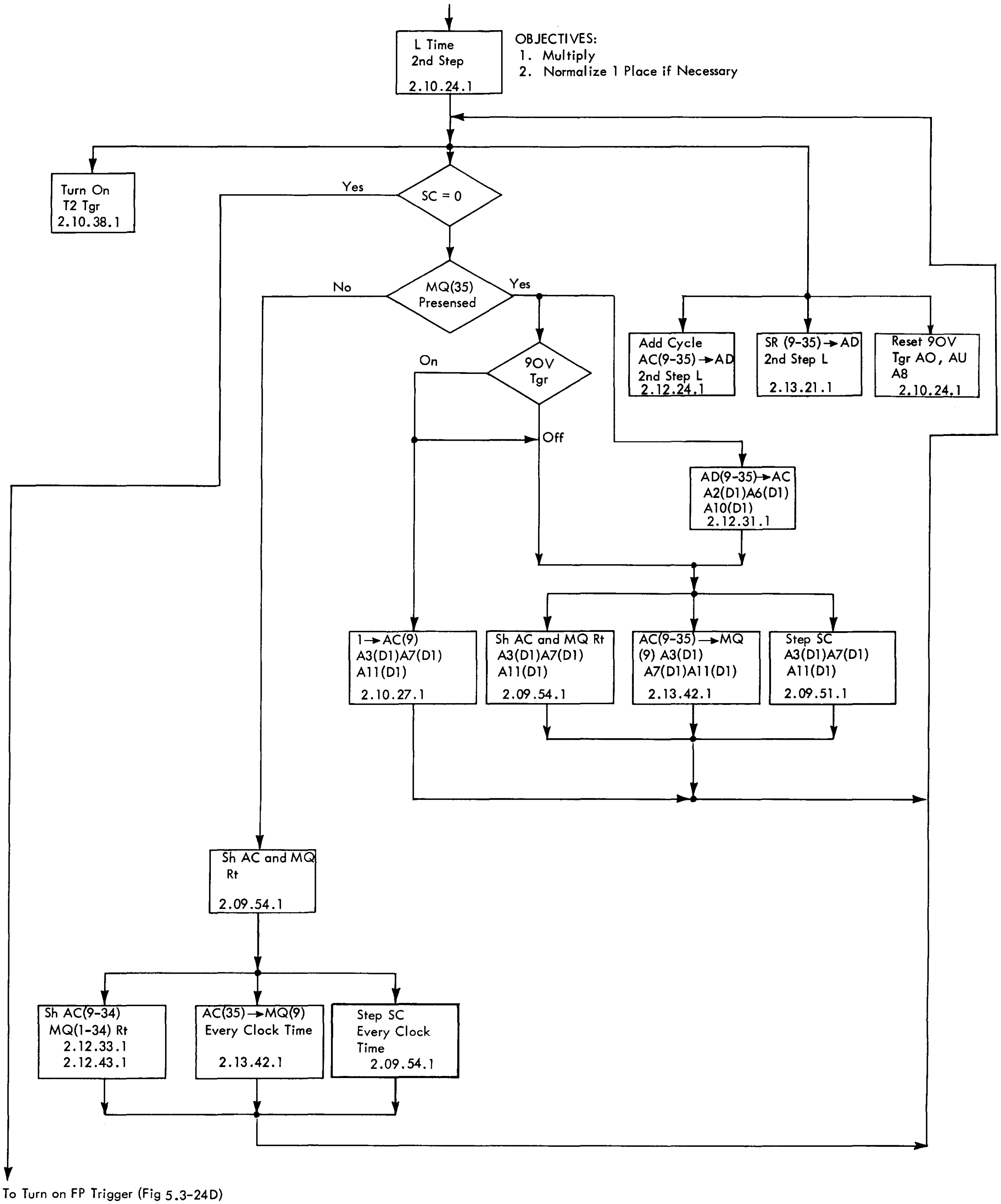
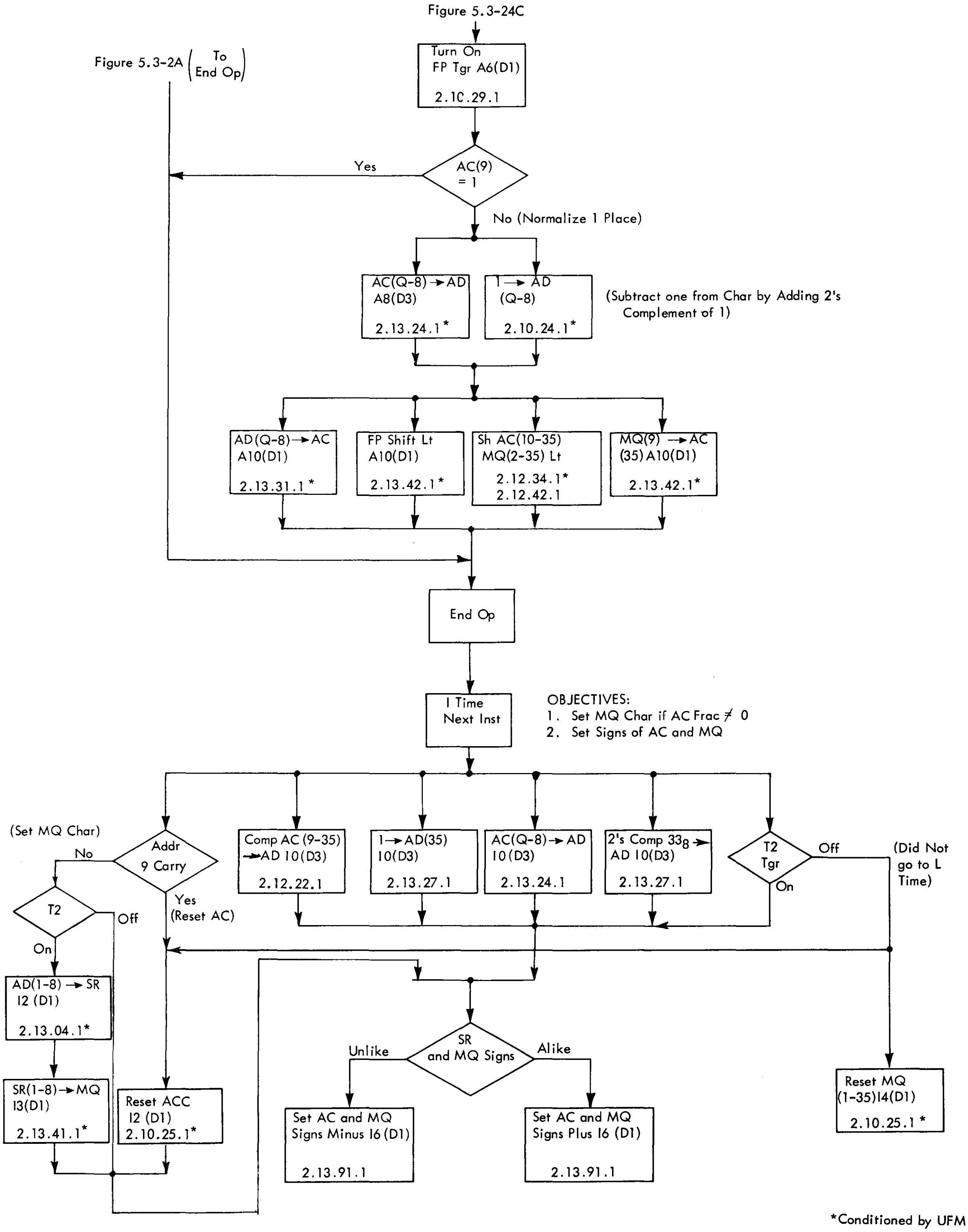


FIGURE 5.3-24C. UFM - 0260; FMP + 0260



*Conditioned by UFM

FIGURE 5.3-24D. UFM -0260; FMP + 0260

to produce the proper characteristic of the product. Because both of the original characteristics are the exponent plus 200_8, adding these characteristics would produce a resultant characteristic 200_8 too large. Therefore, the addition of the characteristics must be adjusted by 200_8.

The execution of this instruction is accomplished by I, E, and L cycles. The L cycles are divided into first step L time and second step L time.

The E cycle, in addition to bringing the word from storage, has the following objectives:

1. For a zero MQ fraction
   a. Put zeros in the characteristic of the SR and the MQ
   b. End operation at the end of the E cycle
2. For a zero storage fraction
   a. Reset the MQ
   b. End operation at the end of the E cycle
3. Set 33_8 in the SC (used to count the number of shifts)
4. Reset the AC

If either the MQ or the storage fraction is zero, the product must also be zero. Therefore, for this zero condition the MQ(1-35) and the AC(Q-35) contain zeros at the end of the E cycle. This is a normal zero product. For this zero condition end operation comes at the end of E time because no further multiplication is necessary.

To check the MQ fraction for zero, MQ(9-35) is gated to the SR. The set and hold of the SR are not operated; the information does not go to the SR but to the zero checking circuits on Systems 2.12.47.1. The FP trigger is used to remember the MQ zero condition and to cause end operation at the end of the E cycle.

First step L time is used to get the initial characteristic of the final product. This characteristic may be changed at the end of FMP if normalizing is required.

The characteristic of the SR is reduced by 200_8 so the characteristic of the product will be only 200_8 greater than the exponent. The multiplicand characteristic minus 200_8 is put into the AC, and the multiplier characteristic is put into the SR. The two characteristics are added to produce the AC characteristic for the product. During this time the MQ characteristic is set to zero.

Second step L time is used to do the fraction multiplication and to normalize the product. The multiplication is done the same as in MPY, by adding and shifting. For no bits in MQ(34-35) fast shifting occurs, otherwise only three shifts per cycle take place. If there is a carry out of AD (9) during the add cycle, a one is put into AC (9) when the AC and MQ are shifted right.

When 33_8 shifts are complete, the fraction multiplication is accomplished and the shift counter equals zero. The shift counter going to zero turns on the FP trigger, and the trigger is used to instruct the system that the multiplication part of the instruction is finished.

If AC(9) is zero, a single normalizing step is performed. One step is the maximum needed if two normal numbers were used for the multiplication.

During I time of the next instruction, the AC fraction is checked for zero by adding one to the 1's complement of the fraction. If the fraction is zero, the addition will cause a column 9 carry. For a zero AC fraction, the whole AC is set to zero and the characteristic of the MQ is left at zero. If the AC fraction is not zero, there is no column 9 carry and the MQ characteristic is set 27_{10} less than the AC characteristic. During I time of the next instruction, the AC and MQ signs are set to the algebraic sign of the product.

Unnormalized Floating Multiply UFM -0260 (Min I, E)
(Max I, E, 11L)

This instruction multiplies the floating-point number stored at the location indicated by the address by the floating-point number in the MQ. This instruction operates the same as FMP except that the result is not normalized or tested. The minus PR (S) prevents FP normalizing gate (Systems 2.10.24.1). The PR (S) not being plus prevents the reset of the AC during the following I cycle (Systems 2.10.25.1). The minus PR (S) prevents turning on the FP trigger if MQ = 0, Systems 02.12.07.0.

Floating Round FRN +0760...0011(I, L) Figure 5.3-25

This instruction examines the contents of MQ(9) and if MQ(9) contains a bit, a one is added to the contents of the AC. A carry out of AC(9) increases the characteristic by one, causes the fraction to be shifted right, and a one to be placed in AC(9).

Floating Divide or Halt FDH +0240(Min I, E, L) Figure 5.3-26
(Max I, E, 12L)

This instruction divides the floating-point number placed in the AC by the floating-point number stored at location X. The result is a floating-point quotient in the MQ and the floating-point remainder of the dividend in the AC. The sign of the MQ is the algebraic sign of the quotient. The sign of the AC is the sign of the dividend. The characteristic of the MQ is equal to the characteristic of the ACC minus the characteristic of the SR, increased by 200_8. The characteristic of the remainder is 27_{10} less than the original dividend characteristic.

If two normalized numbers are used as divisor and dividend fractions, the dividend fraction cannot be twice as large as the divisor fraction. Therefore, the 7090 is designated to halt only when the dividend fraction is at least twice as large as the divisor fraction. The divide check indicator is turned on to cause the halt. The contents of the MQ will be normal zero and the contents of the ACC will remain unchanged.

A zero dividend fraction causes the division to be skipped, but does not halt the computer. A normal zero quotient, plus remainder, results. If the initial factors are normal floating point numbers, the quotient is also normal.

As in any division using exponents, the fractions are divided and the exponent of the divisor is subtracted from the exponent of the dividend. The result is the correct quotient factor and exponent. Division of the fractions is done in the same manner as DVH, by shifting left and attempting reduction. When reduction is successful, a bit is put into MQ(35). Only 27_{10} reductions are attempted in FDH because the fraction is contained in 27_{10} positions.

FDH uses an E cycle and several L cycles. The L cycles are divided into three types (first, second, and third step) through use of the tally counter.

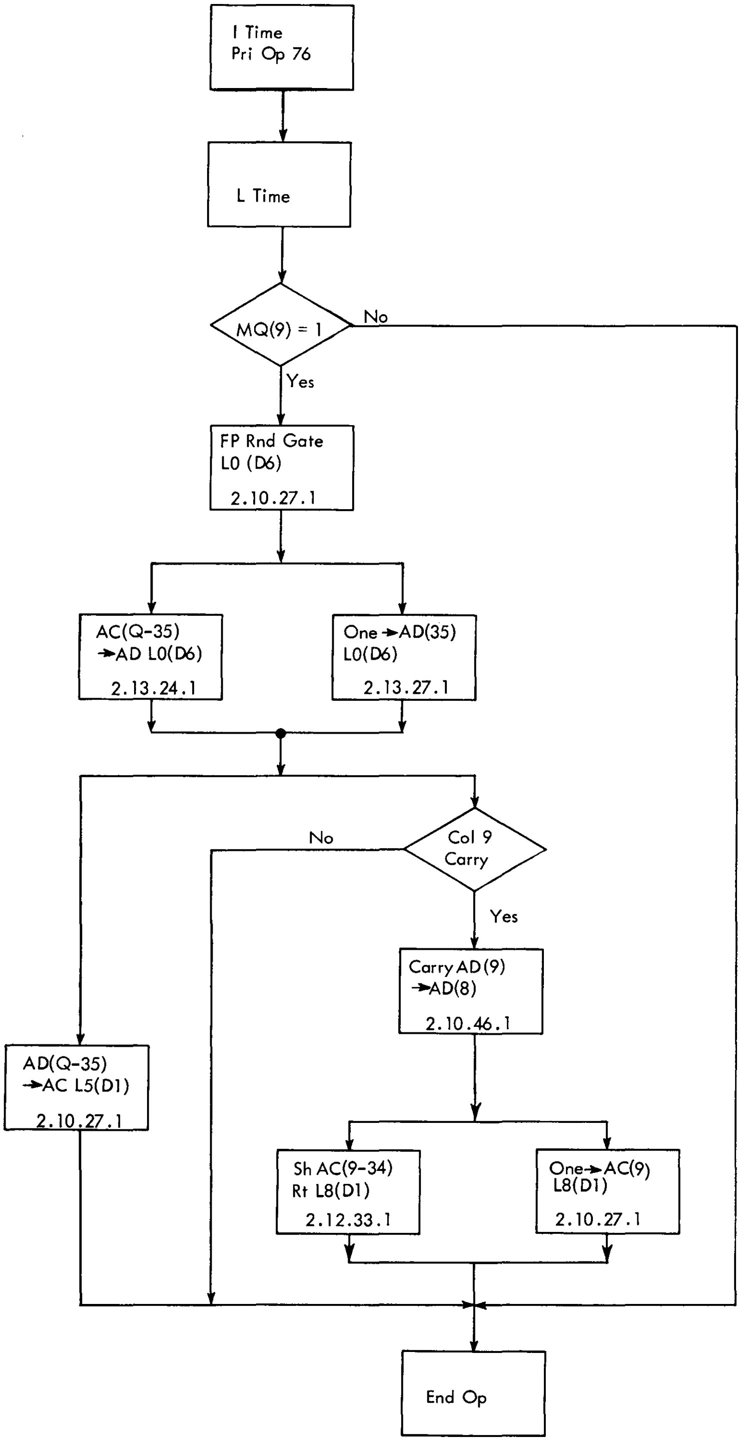


FIGURE 5.3-25. FRN +0760...0011

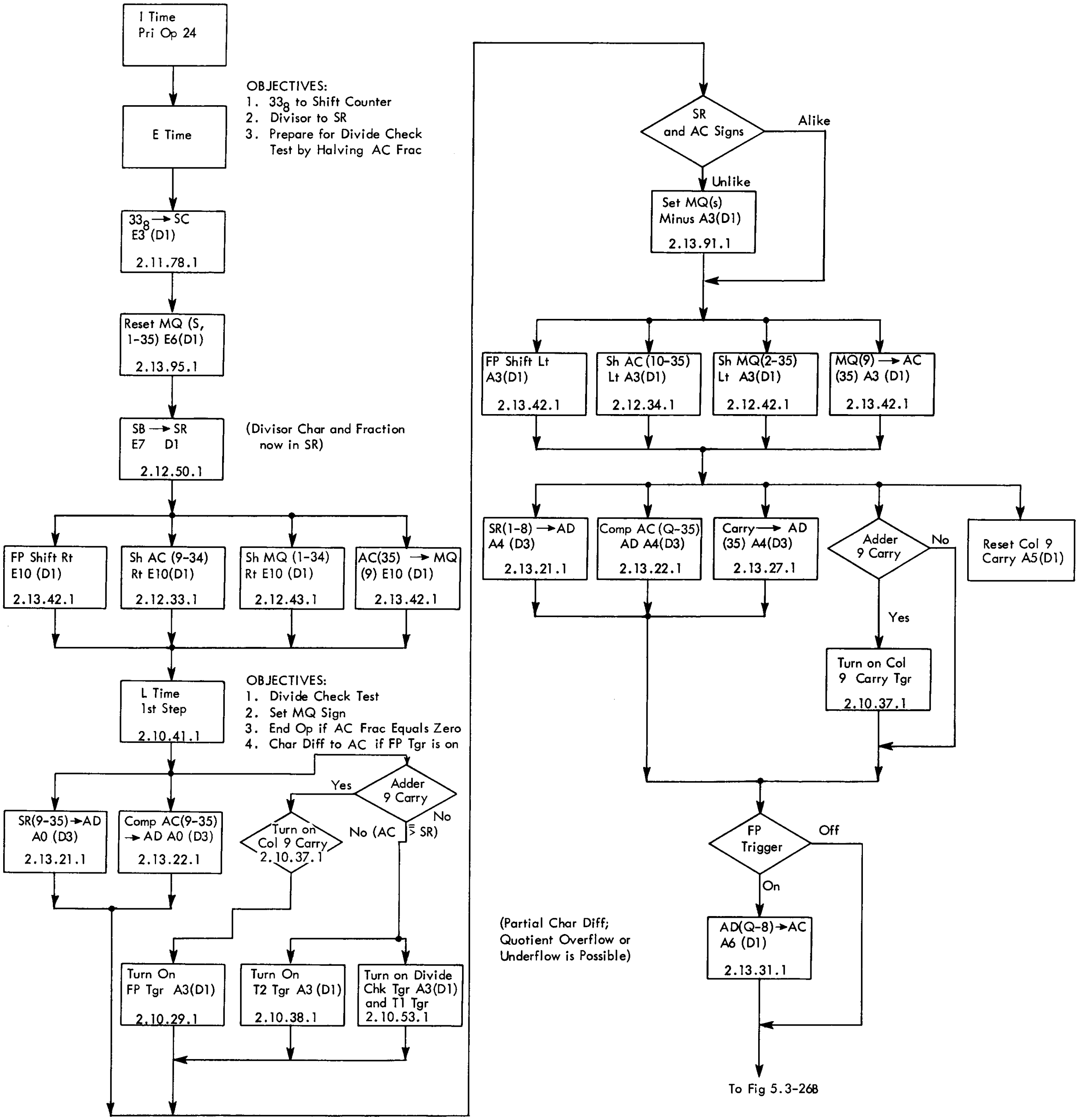
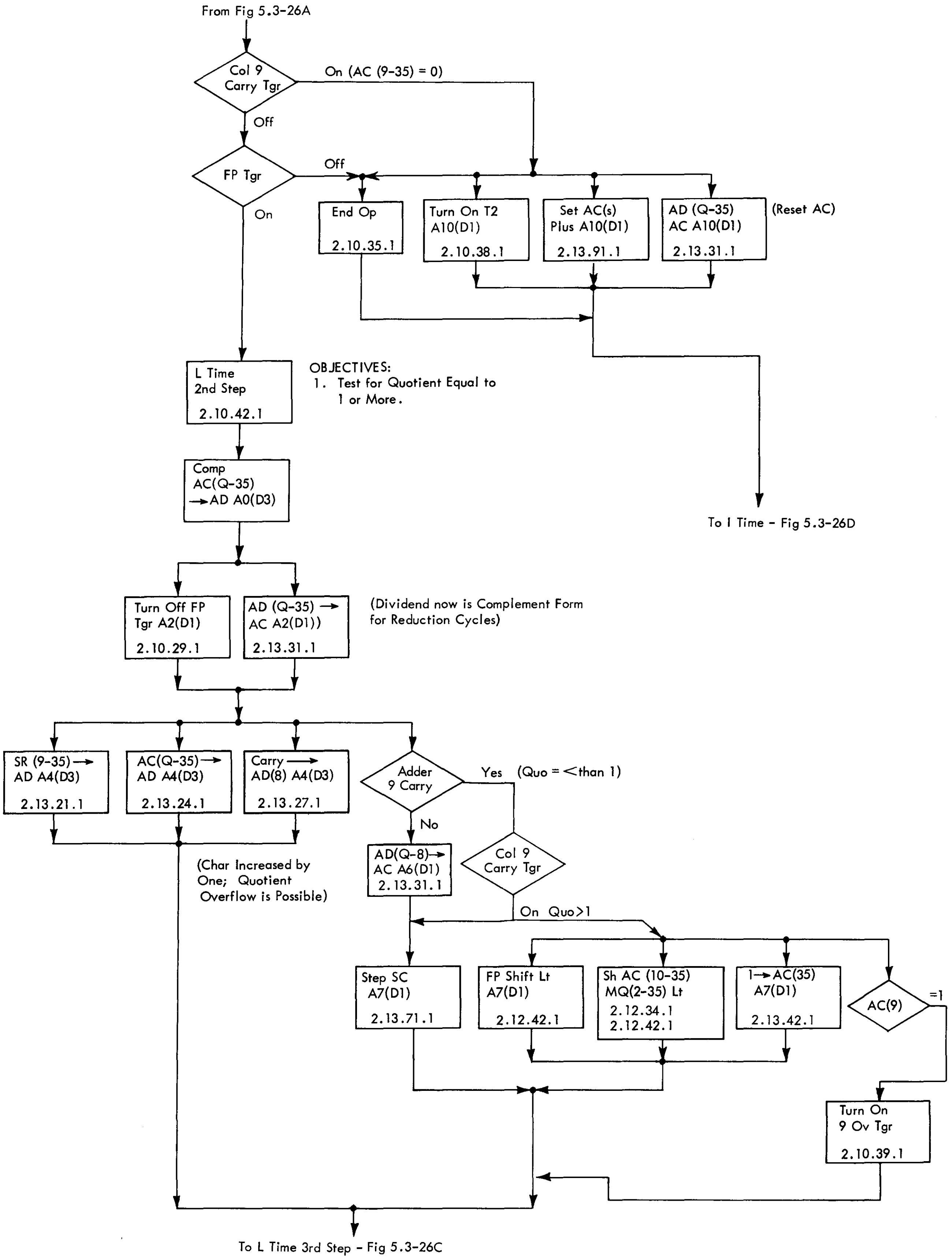


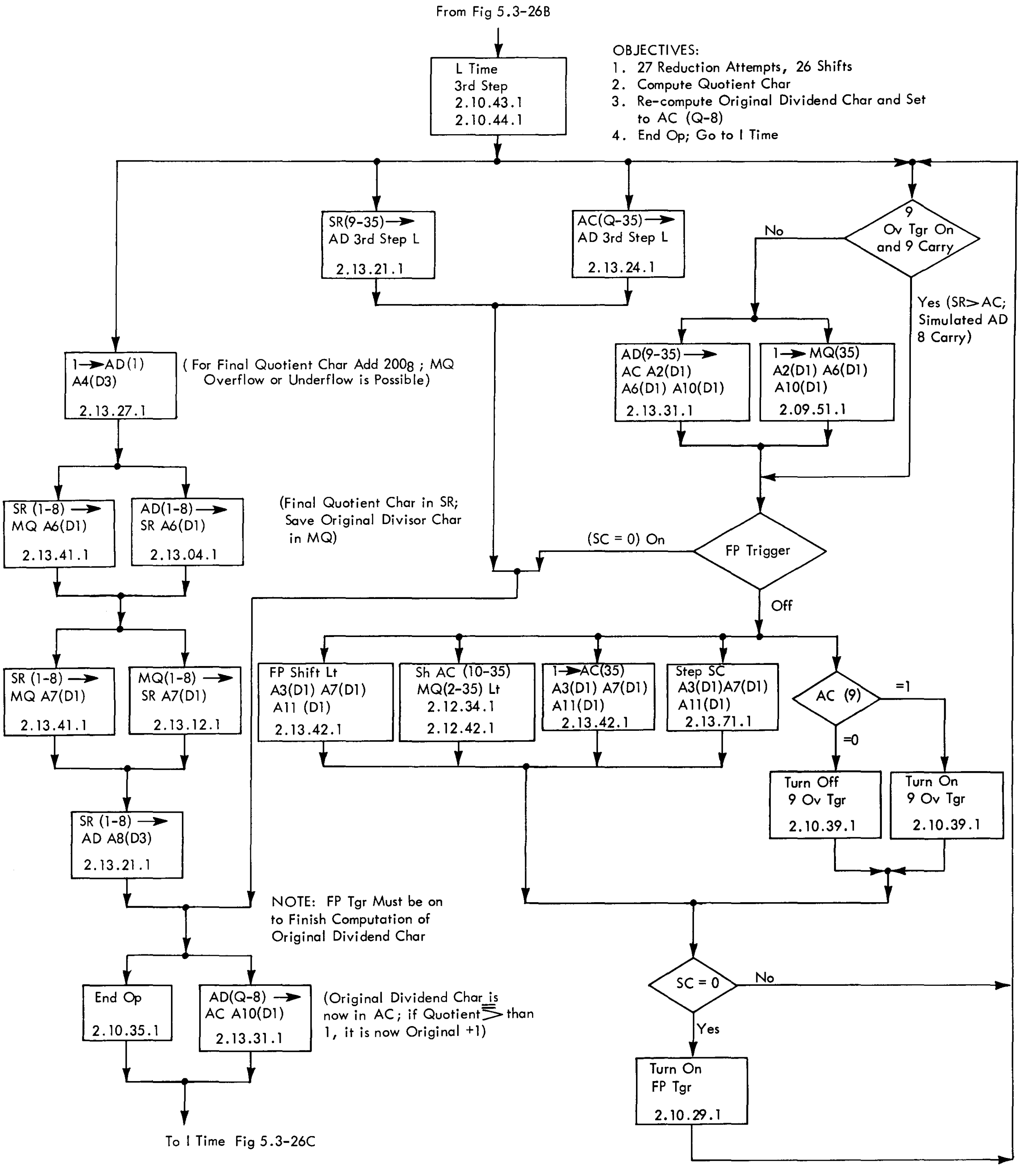
FIGURE 5.3-26A. FDH + 0240


From Fig 5.3-26A
Col 9 Carry Tgr
On (AC (9-35) = 0)
Off
FP Tgr
Off
On
End Op 2.10.35.1
Turn On T2 A10(D1) 2.10.38.1
Set AC(s) Plus A10(D1) 2.13.91.1
AD (Q-35) AC A10(D1) 2.13.31.1
(Reset AC)
L Time 2nd Step 2.10.42.1
To I Time - Fig 5.3-26D
Comp AC(Q-35) →AD A0(D3)
Turn Off FP Tgr A2(D1) 2.10.29.1
AD (Q-35) → AC A2(D1)) 2.13.31.1
(Dividend now is Complement Form for Reduction Cycles)
SR (9-35)→ AD A4(D3) 2.13.21.1
AC(Q-35)→ AD A4(D3) 2.13.24.1
Carry → AD(8) A4(D3) 2.13.27.1
Adder 9 Carry
Yes (Quo = <than 1)
No
AD(Q-8)→ AC A6(D1) 2.13.31.1
Col 9 Carry Tgr
On Quo>1
(Char Increased by One; Quotient Overflow is Possible)
Step SC A7(D1) 2.13.71.1
FP Shift Lt A7(D1) 2.12.42.1
Sh AC (10-35) MQ(2-35) Lt 2.12.34.1 2.12.42.1
1→AC(35) A7(D1) 2.13.42.1
AC(9)
=1
Turn On 9 Ov Tgr 2.10.39.1
To L Time 3rd Step - Fig 5.3-26C

OBJECTIVES:

1. Test for Quotient Equal to 1 or More.

FIGURE 5.3-26B. FDH + 0240

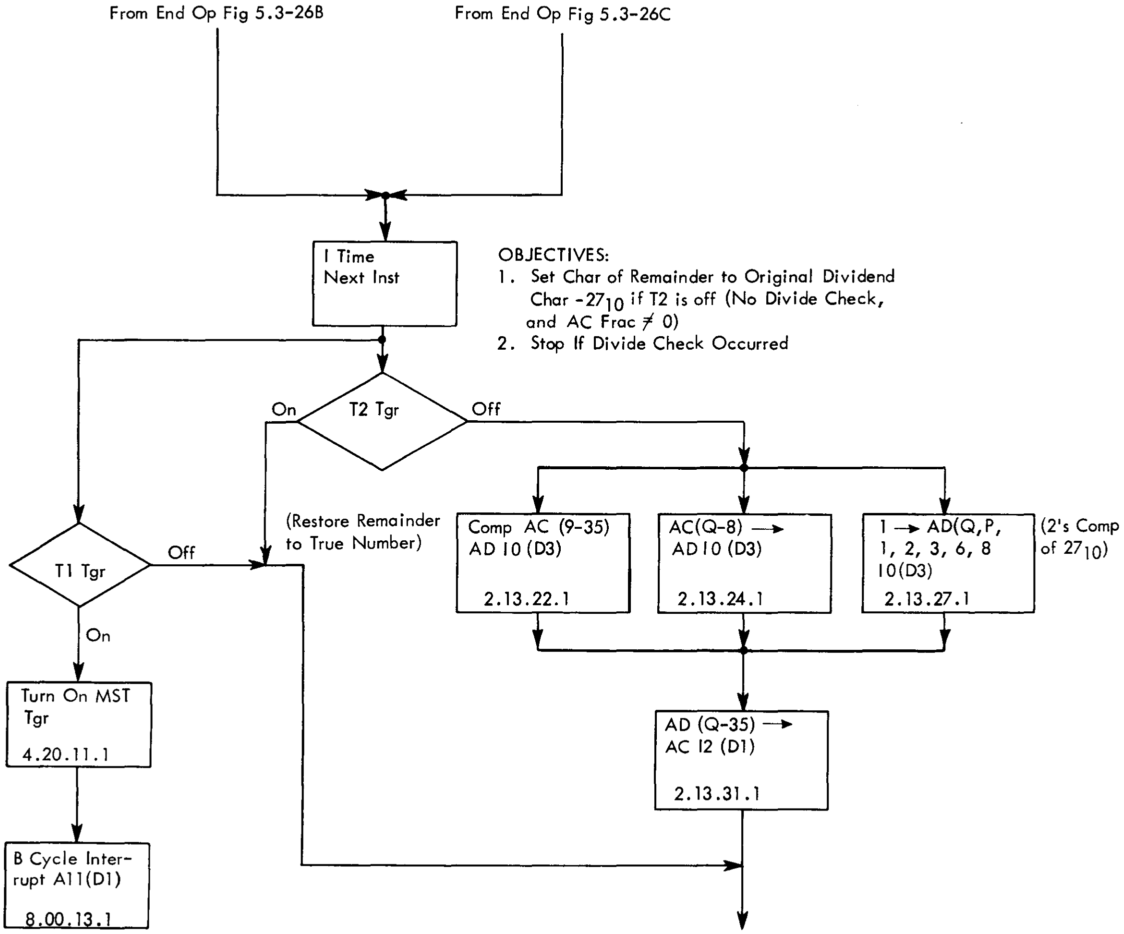

From Fig 5.3-26B
L Time 3rd Step 2.10.43.1 2.10.44.1
SR(9-35) → AD 3rd Step L 2.13.21.1
AC(Q-35) → AD 3rd Step L 2.13.24.1
9 Ov Tgr On and 9 Carry
No
Yes (SR>AC; Simulated AD 8 Carry)
1 → AD(1) A4(D3) 2.13.27.1
(For Final Quotient Char Add 200$_8$; MQ Overflow or Underflow is Possible)
AD(9-35) → AC A2(D1) A6(D1) A10(D1) 2.13.31.1
1 → MQ(35) A2(D1) A6(D1) A10(D1) 2.09.51.1
SR (1-8) → MQ A6(D1) 2.13.41.1
AD(1-8) → SR A6(D1) 2.13.04.1
(Final Quotient Char in SR; Save Original Divisor Char in MQ)
(SC = 0) On
FP Trigger
Off
SR (1-8) → MQ A7(D1) 2.13.41.1
MQ(1-8) → SR A7(D1) 2.13.12.1
FP Shift Lt A3(D1) A7(D1) A11 (D1) 2.13.42.1
Sh AC (10-35) MQ(2-35) Lt 2.12.34.1 2.12.42.1
1 → AC(35) A3(D1) A7(D1) A11(D1) 2.13.42.1
Step SC A3(D1)A7(D1) A11(D1) 2.13.71.1
AC (9)
=1
=0
Turn Off 9 Ov Tgr 2.10.39.1
Turn On 9 Ov Tgr 2.10.39.1
SR (1-8) → AD A8(D3) 2.13.21.1
SC = 0
No
Yes
End Op 2.10.35.1
AD(Q-8) → AC A10(D1) 2.13.31.1
Turn On FP Tgr 2.10.29.1
To I Time Fig 5.3-26C

OBJECTIVES:
1. 27 Reduction Attempts, 26 Shifts
2. Compute Quotient Char
3. Re-compute Original Dividend Char and Set to AC (Q-8)
4. End Op; Go to I Time

NOTE: FP Tgr Must be on to Finish Computation of Original Dividend Char

(Original Dividend Char is now in AC; if Quotient ≧ than 1, it is now Original +1)

FIGURE 5.3-26C. FDH + 0240


From End Op Fig 5.3-26B
From End Op Fig 5.3-26C
I Time Next Inst
On
T2 Tgr
Off
(Restore Remainder to True Number)
Comp AC (9-35) AD 10 (D3) 2.13.22.1
AC(Q-8) → AD 10 (D3) 2.13.24.1
1 → AD(Q,P, 1, 2, 3, 6, 8 10(D3) 2.13.27.1
(2's Comp of 27₁₀)
T1 Tgr
Off
On
Turn On MST Tgr 4.20.11.1
B Cycle Interrupt A11(D1) 8.00.13.1
AD (Q-35) → AC 12 (D1) 2.13.31.1

OBJECTIVES:

1. Set Char of Remainder to Original Dividend Char -27_{10} if T2 is off (No Divide Check, and AC Frac ≠ 0)
2. Stop If Divide Check Occurred

FIGURE 5.3-26D. FDH + 0240

The E cycle is used to:

1. Reset the MQ
2. Bring the divisor from storage
3. Set 33_8 (27_{10}) into the SC (used to count the number of shifts during divide)
4. Divide dividend fraction by 2 by shifting the fraction right one place (to be used for comparison of dividend fraction to divisor fraction)

First step L time is used to:

1. Determine if dividend fraction is more or less than two times the divisor fraction
2. Set the MQ(S) to the algebraic sign of the quotient
3. Shift dividend left one place to put dividend back to initial position
4. Check AC fraction for zero
5. Get the difference between divisor and dividend characteristics
6. End operation if the AC fraction is zero or if the dividend fraction is twice as large or larger than the divisor fraction
7. Reset the whole AC if the AC fraction is zero

Second step L time is used to:

1. Increase the characteristic difference by one if dividend fraction is equal to or greater than the divisor fraction
2. Shift the dividend fraction left one place if the dividend fraction is less than the divisor fraction
3. Step the shift counter down to 26_{10}--the desired number of shifts to move a bit from MQ(35) to MQ(9).

Third step L time is used to do the actual division. The objectives of third step L time are to:

1. Attempt to reduce the AC fraction by the amount of the divisor fraction
2. If the reduction is successful put a 1 in MQ(35)
3. Shift AC and MQ left one place, step shift counter, and put a 1 in AC(35)
4. Repeat steps 1, 2, and 3 until the SC equals zero
5. Compute the characteristic of the quotient and put it into the MQ
6. End operation when the shifting and reductions are complete

I time of the next instruction is used to:

1. If there was no divide check and the dividend fraction was not zero:
   a. AC fraction is re-complemented to get true number
   b. Characteristic of the remainder is set to 27_{10} less than the original characteristic of the dividend
2. Stop the machine if there was a divide check

Floating Divide or Proceed FDP +0241 (Min I, E, L)
(Max I, E, 12L)

This instruction operates the same as FDH, except that a divide check does not halt the computer.

5.3.05 Transfer Instructions

Transfer instructions are used to alter the sequence of instructions. The conditional transfers allow automatic testing of problem conditions without stopping the computer. The transfer instructions greatly reduce program length by allowing program loops and subroutine operation

Transfer	TRA +0020 (I)	Figure 5.3-27

The transfer instruction causes the computer to take its next instruction from location X and resets the instruction counter to X. The address portion of the transfer instruction is substituted for the instruction counter when setting the address register to locate the next instruction. The instruction counter is then set to this new address, and the instruction sequence continues from the new location.

Transfer on MQ Plus	TQP +0162(I)	Figure 5.3-28

If the sign of the MQ register is positive, a transfer will be taken to storage location X. If the MQ sign is minus, the computer proceeds to the next instruction in sequence.

Transfer on Plus	TPL +0120 (I)

If the sign of the accumulator is plus, a transfer is taken to storage location X. If the sign is minus, the computer proceeds to the next instruction in sequence. TPL is executed in the same manner as TQP, except that the AC sign rather than the MQ sign is tested. See Systems 2.10.08.1.

Transfer on Minus	TMI -0120 (I)

If the sign of the AC is minus, a transfer is taken to storage location X. If the AC sign is plus, the computer proceeds to the next instruction in sequence. The execution of this instruction is the same as TQP, except that the AC sign rather than the MQ sign is tested. See Systems 2.10.08.1.

Transfer on Overflow	TOV +0140 (I)

If the AC overflow trigger is on as a result of a previous operation, a transfer is taken to storage location X, and the overflow trigger is turned off. If the overflow trigger is off, the computer proceeds to the next instruction in sequence. Execution of TOV is much like TQP, except that the overflow trigger rather than the MQ sign is tested. See Systems 2.10.08.1. The AC overflow trigger is turned off on Systems 2.10.36.1.

Transfer on No Overflow	TNO -0140 (I)

If the AC overflow trigger is off, the next instruction is taken from storage location X. If the overflow trigger is on, no transfer is taken, and the overflow trigger is turned off. Operation of this instruction is the same as TQP, except that the overflow trigger rather than the MQ sign is tested.

Transfer on Quotient Overflow	TQO +0161 (I)

If the quotient overflow is on because of a previous operation, a transfer is taken to storage location X, and the quotient overflow trigger is turned off. If the quotient overflow trigger is off, the computer proceeds to the next 'instruction in sequence. This is a 704 compatibility instruction.

Transfer on Zero	TZE +0100 (I, L)	Figure 5.3-29

If the contents of the AC (including the overflow positions) are zero, a transfer is taken to storage location X. If the contents are not zero, the computer proceeds to the next instruction in sequence. In either case the contents of the AC are not changed.

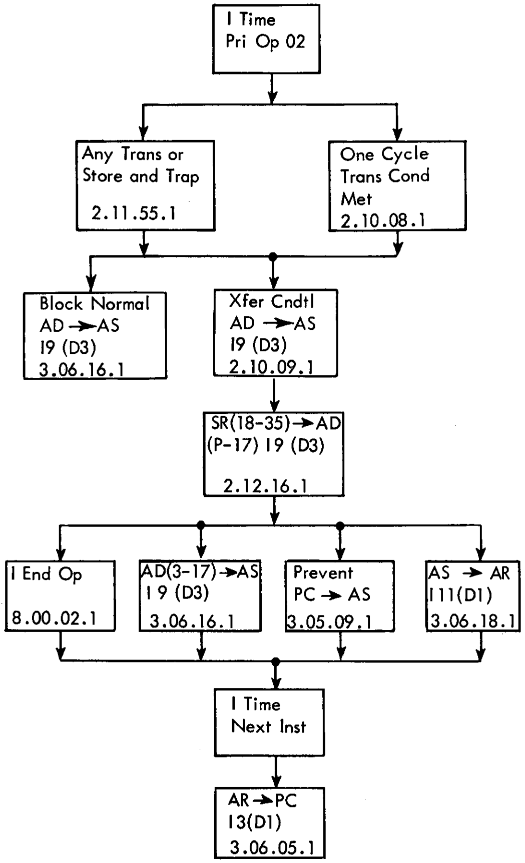


FIGURE 5.3-27. TRA + 0020

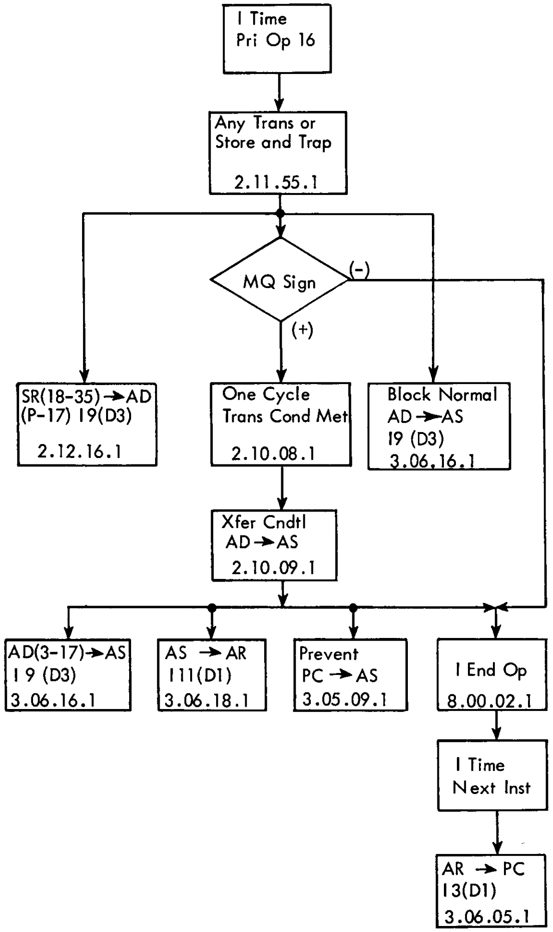


FIGURE 5.3-28. TQP + 0162

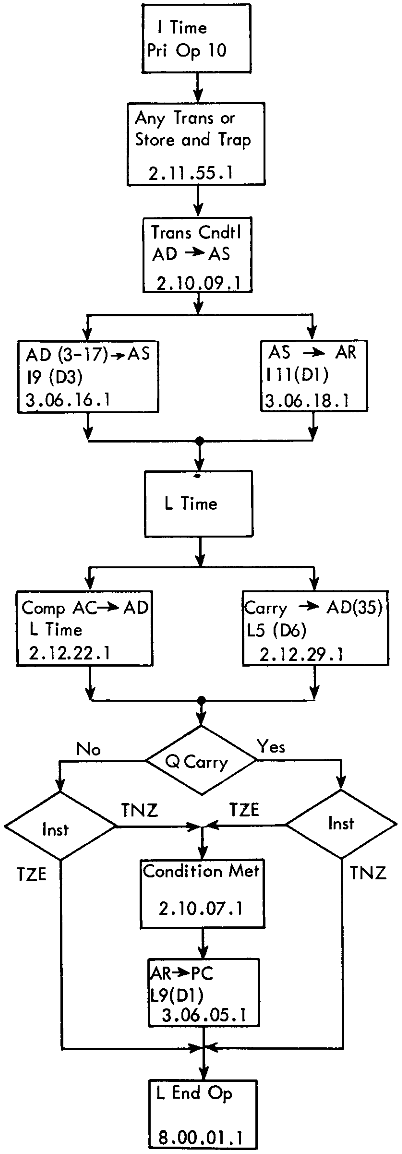


FIGURE 5.3-29. TZE + 0100; TNZ - 0100

To test for a zero condition, the accumulator is complemented to the adders, and a one is added to position 35. If a zero condition exists, a carry results which ripples through all of the adders and turns on the Q carry trigger. The Q carry trigger is used to condition the transfer circuits.

Transfer on No Zero TNZ -0100 (I, L) Figure 5.3-29

If the contents of the AC are not zero, the next instruction is taken from storage location X. If the contents are zero, the computer proceeds to the next instruction in sequence.

Transfer on Low MQ TLQ +0040 (I, L) Figure 5.3-30

An algebraic comparison is made between the MQ and the accumulator contents. If the MQ contents are less than the accumulator contents, an instruction transfer is made to storage location X. If the MQ is greater or equal, no transfer is taken. For this instruction a +0 is considered to be larger than a -0. The contents of both registers are left unchanged.

The operation is performed by adding the contents of the MQ to the complemented accumulator contents. The Q carry trigger is then matched with the register signs to determine whether the conditions for transfer have been met. To prevent transfers when the factors are equal, a one is added to AC(35) to produce a Q carry when the AC factor is plus.

The following table illustrates comparisons which might be made, along with the desired result:

No. in AC	No. in MQ	Q Carry	Transfer
-7	-6	No	No
-6	-6	No	No
-0	-6	Yes	Yes
-0	-0	No	No
+0	-0	Yes	Yes
-0	+0	No	No
-6	+6	No	No
+0	+6	Yes	No
+6	+6	Yes	No
+7	+6	No	Yes

Transfer on Channel in Operation TCO +XXXX (I, L) Figure 5.3-31

This instruction causes a transfer to storage location X if the particular data channel is in use. Operation codes +0060 through +0067 are used to check data channels A through H, respectively.

Transfer on Channel Not in Operation TCN -XXXX (I, L) Figure 5.3-31

TCN causes a transfer to storage location X if the data channel is not in operation. Operation codes -0060 through -0067 are used to select data channels A through H, respectively.

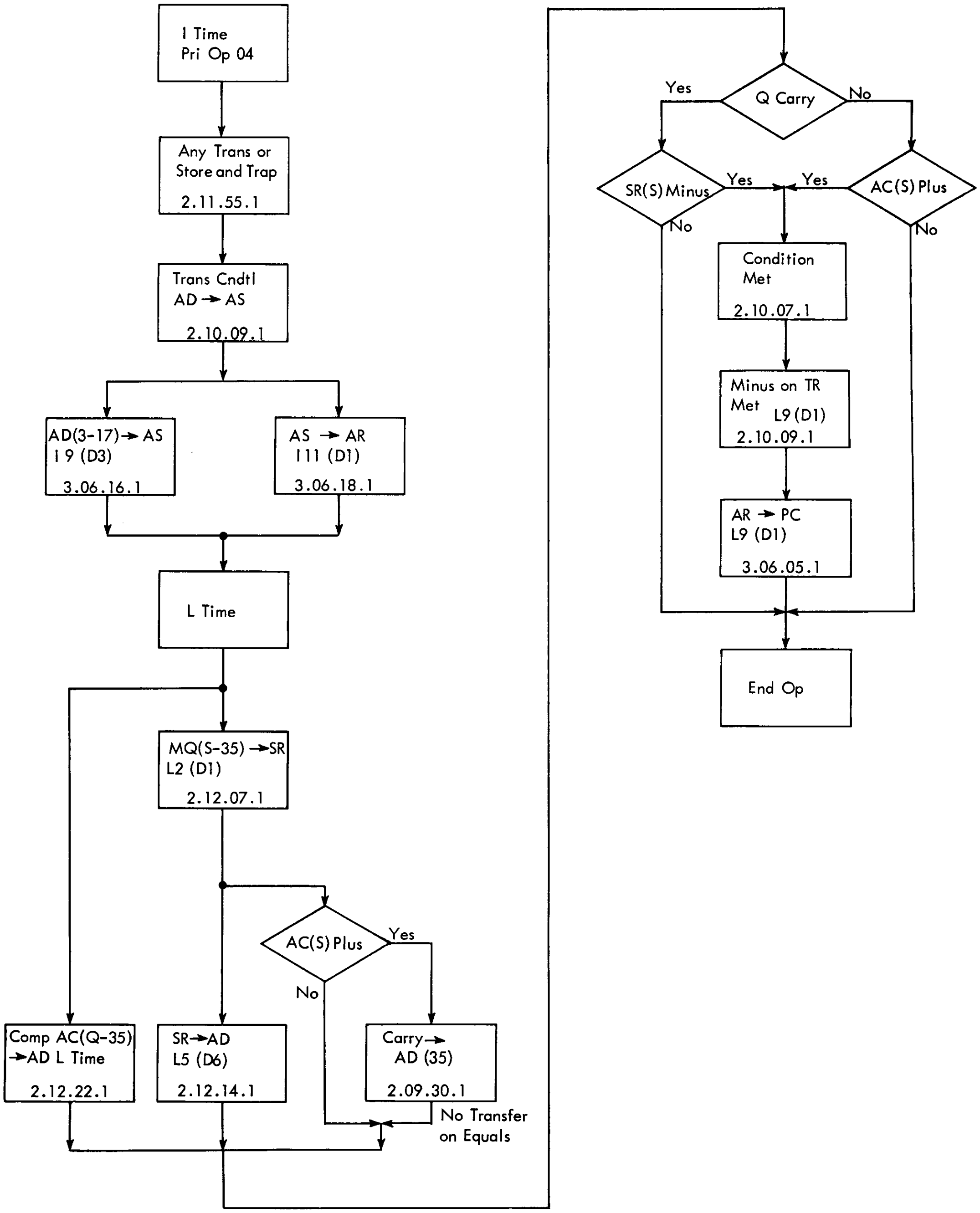


FIGURE 5.3-30. TLQ + 0040

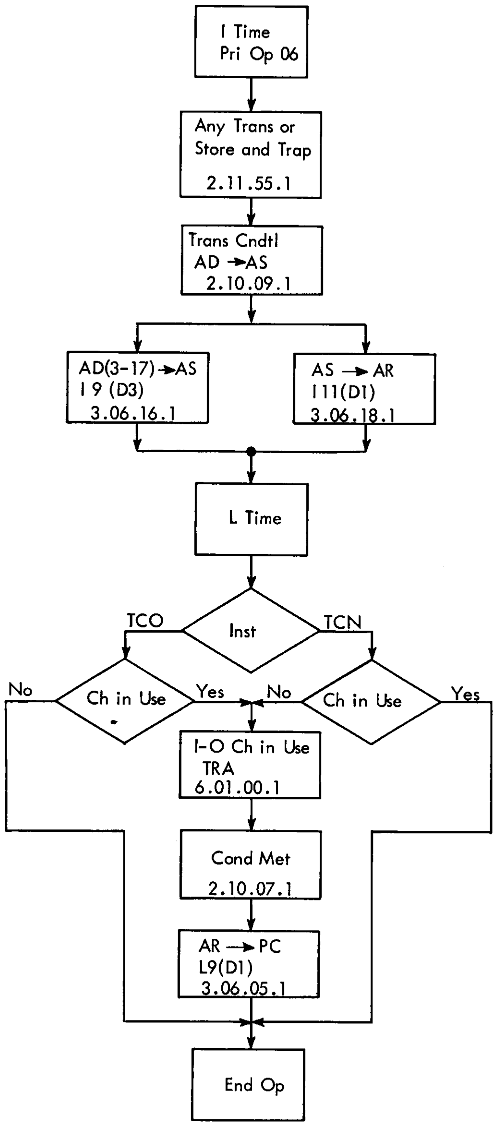


FIGURE 5.3-31. TCO + 0060; + 0061; + 0062; Etc.
TCN – 0060; – 0061; – 0062; Etc.

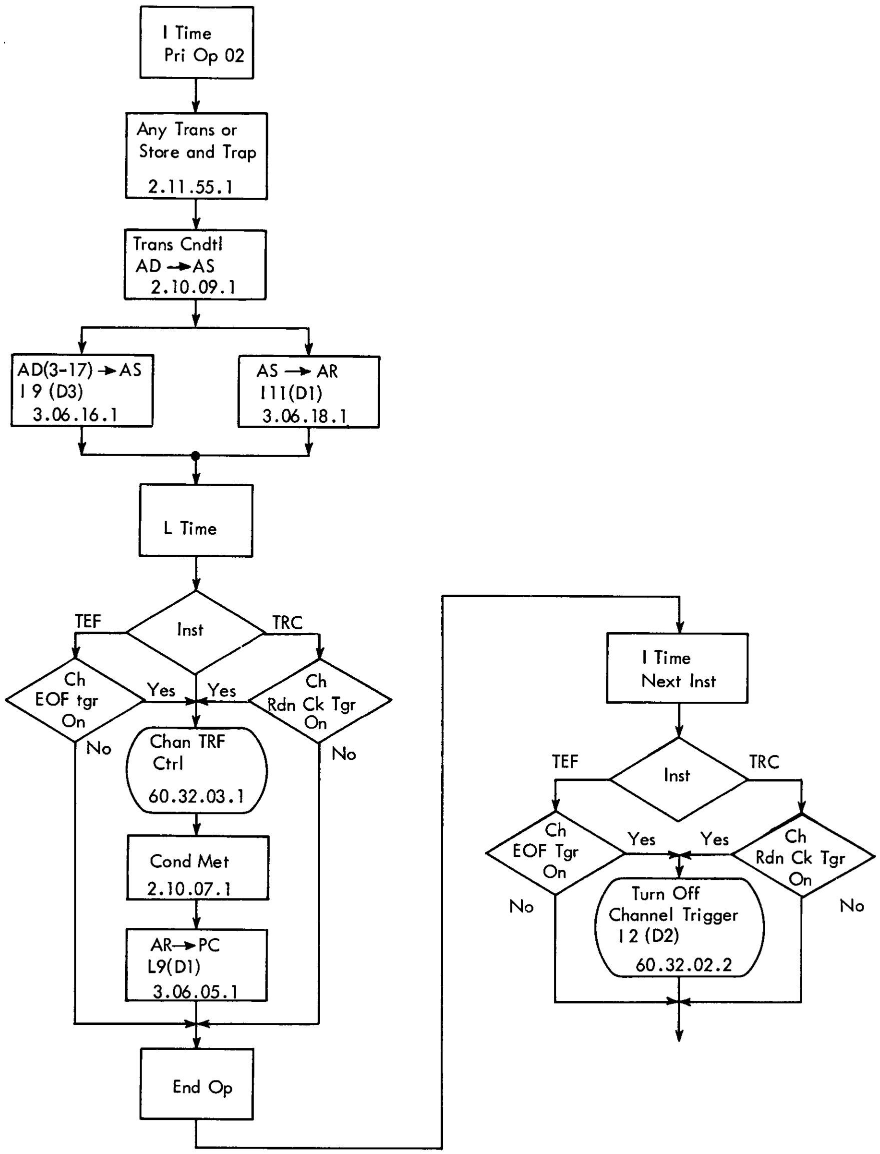


FIGURE 5.3-32. TRC + 0022; -0022; + 0024; Etc.
TEF + 0030; -0030; + 0031; Etc.

Transfer on Data Channel Redundancy Check TRC ±XXXX(I, L) Figure 5.3-32

If the data channel redundancy check trigger is on, a transfer is taken to location X and the trigger is turned off. If the trigger is off, the computer takes the next sequential instruction. Operation codes +0022, -0022, +0024, -0024, +0026, -0026, +0027, and -0027 are used to select data channels A through H, respectively.

Transfer on Data Channel End of'File TEF ±XXXX(I, L) Figure 5.3-32

If the data channel end-of-file trigger is on, a transfer is taken to storage location X and the trigger is turned off. If the trigger is off, the computer takes the next instruction in sequence. Operation codes +0030, -0030, +0031, -0031, +0032, -0032, +0033, and -0033 are used to select data channels A through H, respectively.

5.3.06 Trap Mode Instructions

The 7090 can be operated in either of two modes, normal or trapping. Entrance to trapping mode is gained by executing the ETM instruction. Exit is accomplished by using the LTM instruction or the clear or reset keys on the console. Trapping mode affects the operation of transfer instructions except TTR. In trapping mode, the location of each transfer instruction is stored in the address portion of location 0000. Successful transfers are not executed; instead, an instruction transfer, or trap, is taken to location 0001. One instruction, trap transfer, is immune to trapping mode. The locating of successful transfers and the trap to a common check point make trapping mode useful for debugging program flow. Unsuccessful transfers have their location stored in the address portion of 0000, but do not trap to 0001.

Enter Trapping Mode ETM +0760...0007 (I, L)

This instruction places the computer in trapping mode by turning on the trap mode trigger on Systems 2.10.53.1. The computer remains in trapping mode until a LTM instruction is executed or the clear or reset buttons are depressed. ETM is a primary operation 76 instruction and requires an I and an L cycle.

TRA in Trapping Mode TRA +0020 (I, E)

The effect of trapping mode on a transfer is shown in Figure 5.3-33.

Leave Trapping Mode LTM -0760...0007 (I, L)

This instruction returns the computer to normal mode by turning off the trap mode trigger on Systems 2.10.53.1. This is a primary operation 76 instruction, and an I and an L cycle are required.

Trap Transfer TTR +0021 (I)

This is the only instruction which provides an instruction transfer to location X regardless of the operating mode of the computer. The TTR instruction nullifies the transfer blocking circuits of trapping mode (Systems 2.10.53.1) and operates like TRA.

Store Location and Trap STR -1000 (I, E) Figure 5.3-34

This instruction stores its location plus one in the address portion of storage loca-

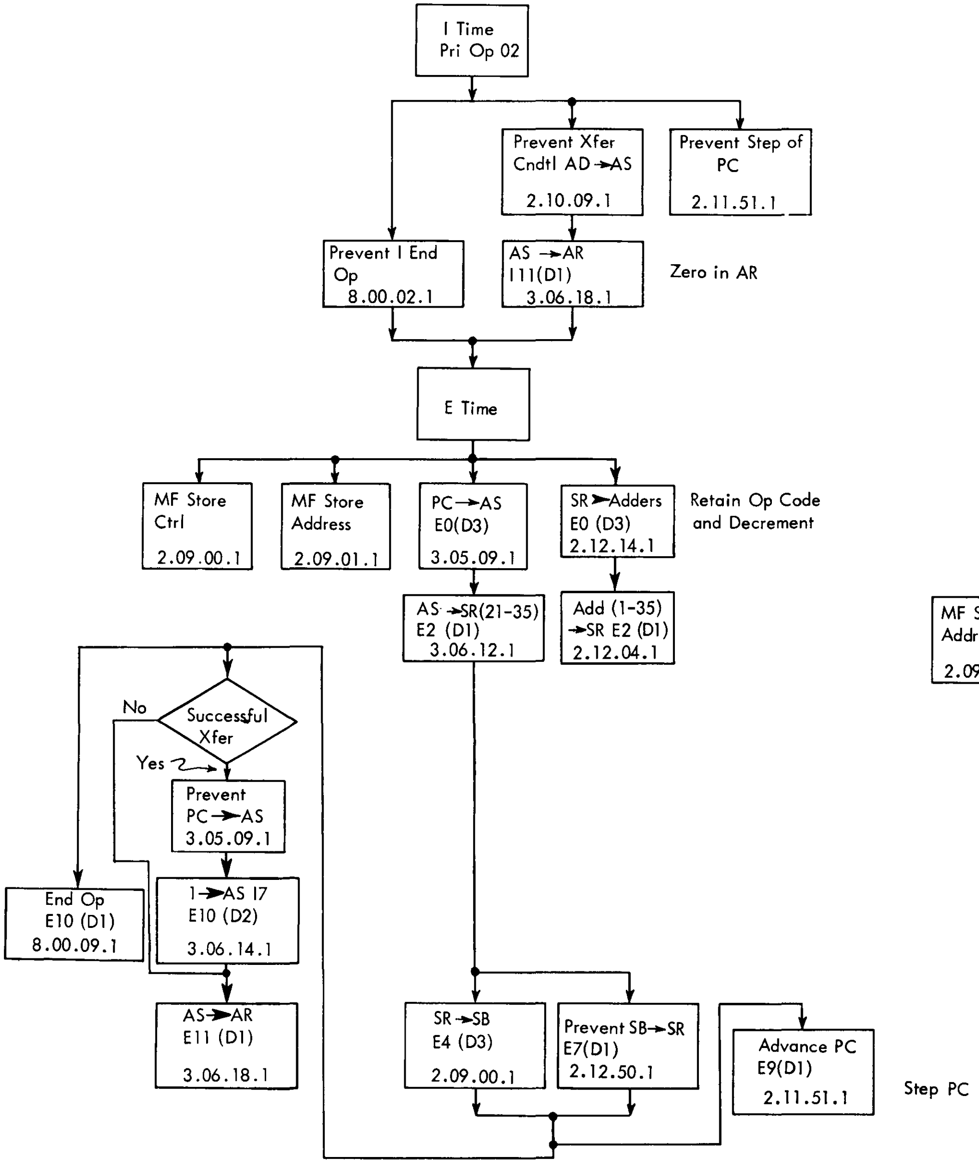


FIGURE 5.3-33. TRA +0020 TRAP MODE

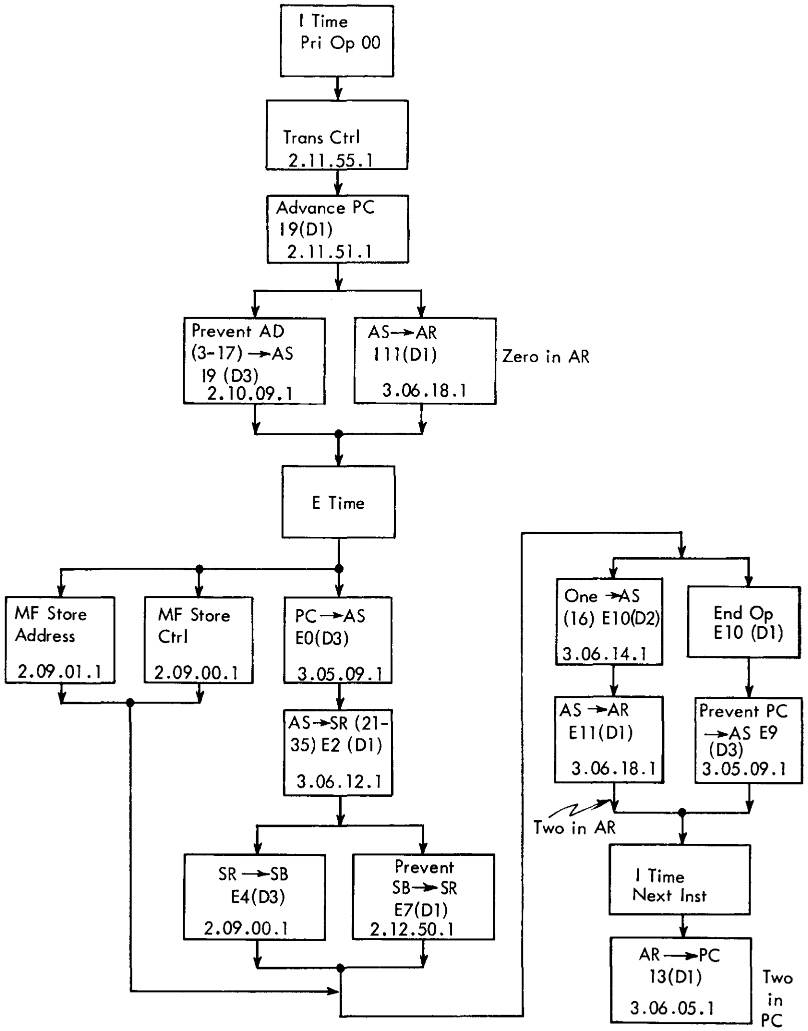


FIGURE 5.3-34. STR - 1000

tion 0000. It then traps or transfers the computer to location 0002 where the instruction sequence is resumed. STR does not place the computer in trapping mode.

5.3.07 Skip Instructions

The skip instructions allow the programmer to alter the program to meet special conditions without stopping the computer. These instructions are similar to the conditional transfer instructions but, instead of transferring, they cause one or two instructions to be skipped. Skipping is accomplished by supplying an extra advance pulse to the instruction counter.

Most skip instructions cause the computer to skip when the condition being tested is met. The exceptions are the error testing and I-O testing instructions, which cause skipping when the condition being tested is not met; e.g., when there are no errors. This exception (skip on no error) allows a straight-line program until an error is detected. To process an error, an instruction transferring to an error subroutine usually follows the test instruction. Another exception, the CAS instruction, has three possible results: it can fail to skip for AC greater, skip once for equal, or skip twice for AC less.

P Bit Test PBT -0760...0001(I, L) Figure 5.3-35

A bit in accumulator (P) position causes the computer to skip one instruction. If there is no bit in (P), the computer takes the next instruction in sequence.

Low-Order Bit Test LBT +0760...0001(I, L) Figure 5.3-35

A bit in accumulator (35) causes the computer to skip one instruction. If there is no bit in (35), the computer takes the next instruction in sequence.

Storage Zero Test ZET +0520(I, E) Figure 5.3-36

If the contents of storage location X, except the sign, are zero, the computer will skip one instruction. If storage is not zero, the computer proceeds to the next instruction in sequence. Storage is unchanged. The information from storage on the SB is tested for zero as shown on Systems 2.12.52.1.

Storage Non-Zero Test NZT -0520(I, E) Figure 5.3-36

If positions (1-35) of storage location X are not zero, the computer skips the next instruction. If the contents of storage location X are zero, no skip is taken. Storage is unchanged.

Compare Accumulator with Storage CAS +0340(I, E, L) Figure 5.3-37

The accumulator is compared with the word at storage location X. Comparison is accomplished by taking an algebraic difference. If the accumulator is greater than the word in storage, no skip is taken. If the accumulator is equal to the word in storage, one instruction is skipped. If the accumulator is less than the word in storage, the next two instructions are skipped. Neither the accumulator nor the word in storage is changed.

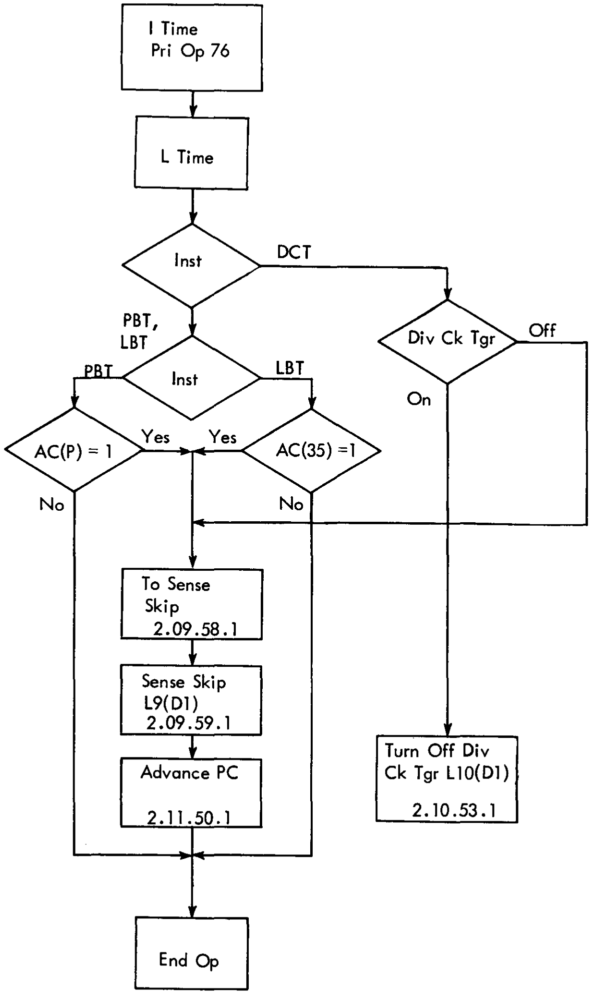


FIGURE 5.3-35. PBT -0760...0001; LBT +0760...0001; DCT +0760...0012

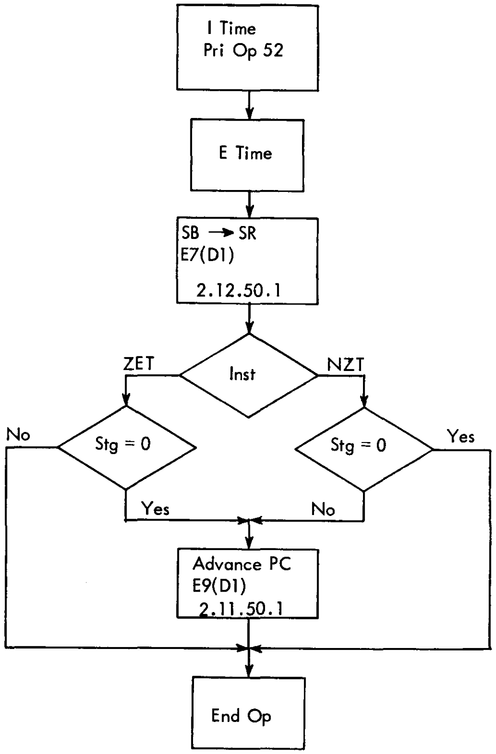


FIGURE 5.3-36. ZET +0520; NZT -0520

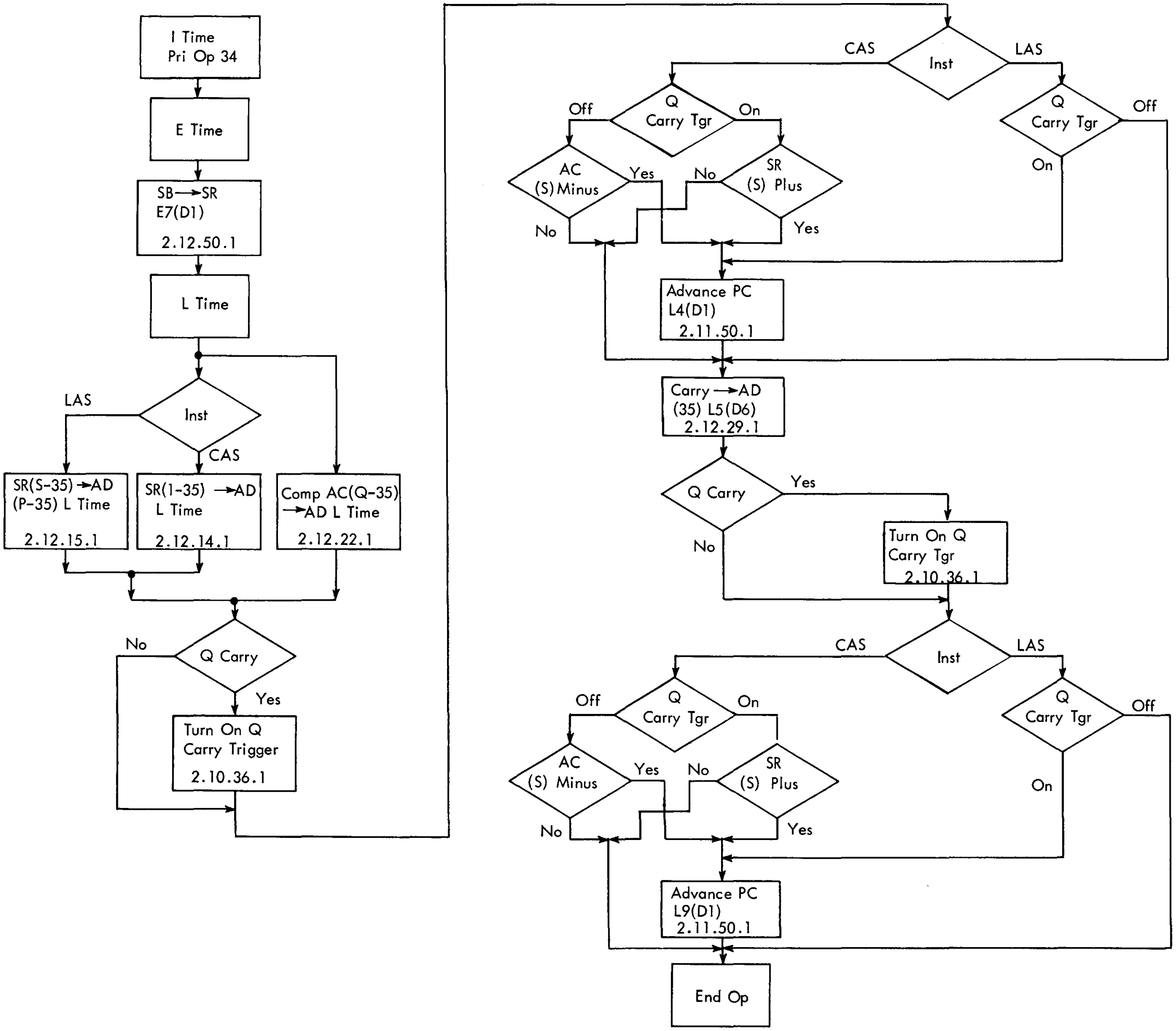


FIGURE 5.3-37. CAS +0340; LAS - 0340

To execute this instruction, an E cycle is required to bring the word in storage to the storage register; then an L cycle is used for two comparisons in the adders. For the first comparison, the SR and the complement of the AC are fed to the adders; the Q carry and sign conditions are matched to condition the first possible skip. The second comparison is made after a one has been added to the difference, to differentiate words of equal magnitude.

The following illustrate some of the possible combinations:

Number in	Number in	Q Carry Tgr		
Accumulator	Storage	1st Comp	2nd Comp	Result
-6	-7	On	On	Next Instruction
-6	-6	Off	On	Skip 1 Instruction
-6	-4	Off	Off	Skip 2 Instructions
-6	+6	Off	On	Skip 2 Instructions
-0	+0	Off	On	Skip 2 Instructions
+0	-0	Off	On	Next Instruction
+6	-6	Off	On	Next Instruction
+6	+4	Off	Off	Next Instruction
+6	+6	Off	On	Skip 1 Instruction
+6	+7	On	On	Skip 2 Instructions

Logical Compare Accumulator with Storage LAS -0340(I, E, L) Figure 5.3-37

This instruction compares the contents of the AC(P, 1-35) with the logical word (S, 1-35) stored at location X. The sign of the AC is disregarded; the contents of the AC and storage are unchanged.

If the contents of the AC are greater than the contents of storage location X, the computer takes the next instruction in sequence. If the AC equals storage, the computer skips one instruction. If the contents of the AC are less than the contents of storage, the computer will skip the next two instructions.

This instruction is executed the same as CAS, except that the signs are not used for an algebraic comparison and AC(P) is compared against SR(S).

Plus Sense PSE +0760...XXXX(I, L)

This instruction provides a means to test the status of any of the six sense switches, to turn on or off the four console sense lights, and to permit the transmission of an impulse to or from the exit or entry hubs of either the printer or punch.

The address portion of the instruction determines whether a light, switch, printer, or card punch is being sensed; further, it determines which light, switch, or hub is being sensed. The octal addresses for the different sense instructions are:

Address	Instruction
0140	Turn off all sense lights (Figure 5.3-38)
0141-0144	Turn on sense light 1, 2, 3, or 4, respectively (Figure 5.3-39)

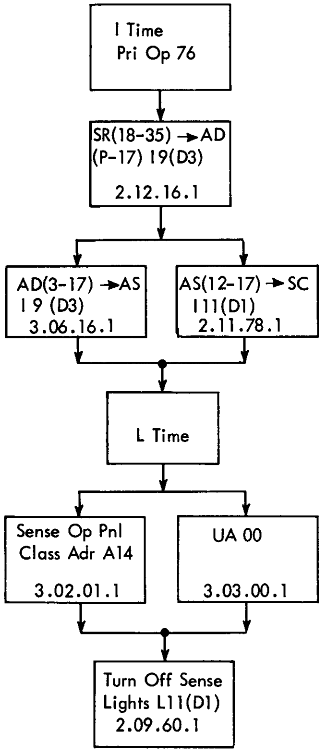


FIGURE 5.3-38. PSE +0760...0140

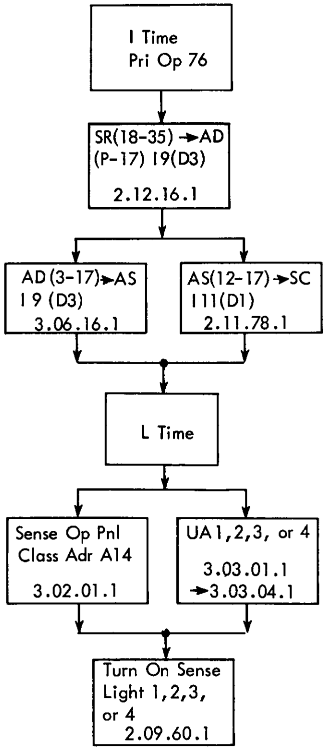


FIGURE 5.3-39. PSE +0760...0141, 0142, 0143, 0144

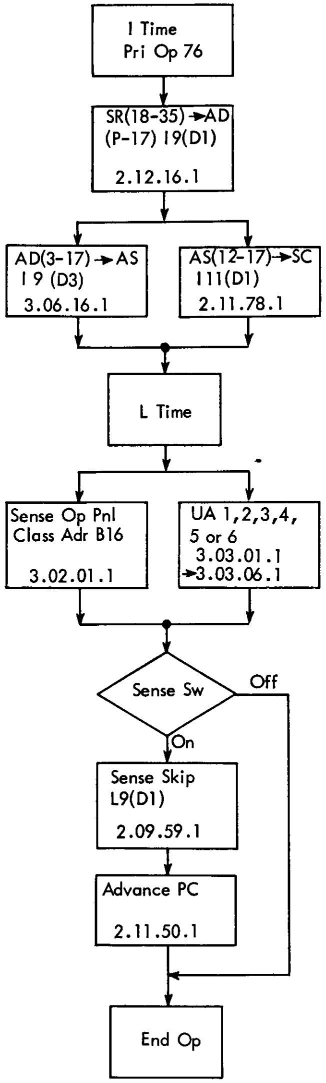


FIGURE 5.3-40. PSE +0760...0161, 0162, ETC.

Address	Instruction
0161- 0166	If the corresponding sense switch is down (on), the computer skips the next instruction. If the sense switch is up (off), the computer takes the next instruction in sequence (Figure 5.3-40).
1341- 1342 2341- 2342 3341- 3342 4341- 4342 5341- 5342 6341- 6342 7341- 7342 10341-10342	The computer causes an impulse to appear at the specified exit hub of the control panel of the card punch attached to Data Channel A, B, C, D, E, F, G, or H respectively (Figure 5.3-41).
1360 2360 3360 4360 5360 6360 7360 10360	If an impulse is present at the entry hub of the control panel of the printer attached to Data Channel A, B, C, D, E, F, G, or H respectively, the computer skips the next instruction. If there is no impulse, the computer takes the next instruction in sequence (Figure 5.3-42).
1361- 1372 2361- 2372 3361- 3372 4361- 4372 5361- 5372 6361- 6372 7361- 7372 10361-10372	The computer causes an impulse to appear at the specified exit hub of the control panel of the printer attached to Data Channel, A, B, C, D, E, F, G, or H, respectively (Figure 5.3-43).

Minus Sense MSE -0760...XXXX(I, L) Figure 5.3-44

If the sense light, on the operator's console, corresponding to the address portion of the instruction is on, this light is turned off and the computer skips the next instruction. If the sense light is off the computer takes the next instruction in sequence. Addresses 0141-0144 correspond to sense lights 1-4, respectively.

Input-Output Check Test IOT +0760...0005(I, L) Figure 5.3-45

If the I-O check trigger is on, the indicator is turned off and the computer takes the next instruction in sequence. If the I-O check indicator is off, the computer skips the next instruction.

Divide Check Test DCT +0760...0012 (I, L) Figure 5.3-35

This instruction examines the status of the divide check trigger. If the trigger is off, the next instruction is skipped. If the trigger is on, it is turned off and the computer takes the next instruction in sequence.

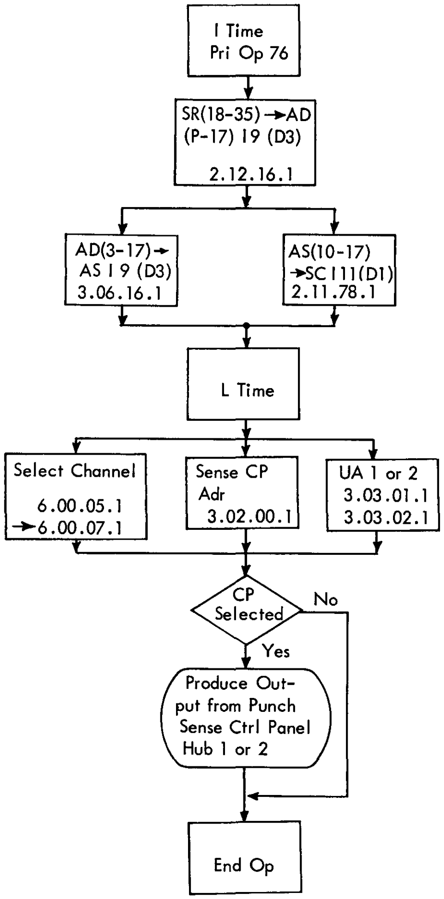


FIGURE 5.3-41. PSE +0760...1341, 2342, 3341, ETC.

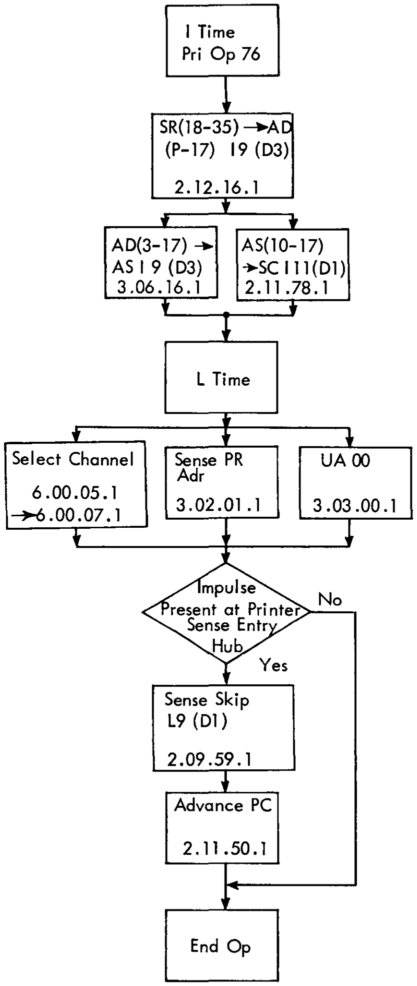


FIGURE 5.3-42. PSE +0760...1360, 2360, ETC.

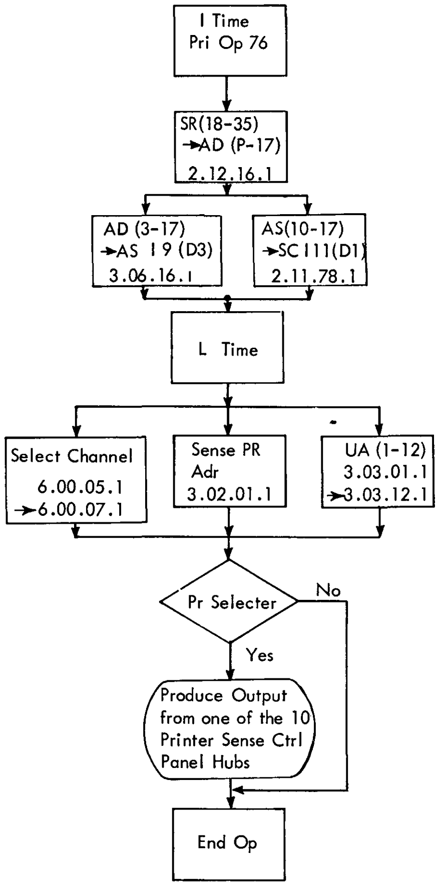


FIGURE 5.3-43. PSE +0760...1361, 1372, 2361, 2362, ETC.

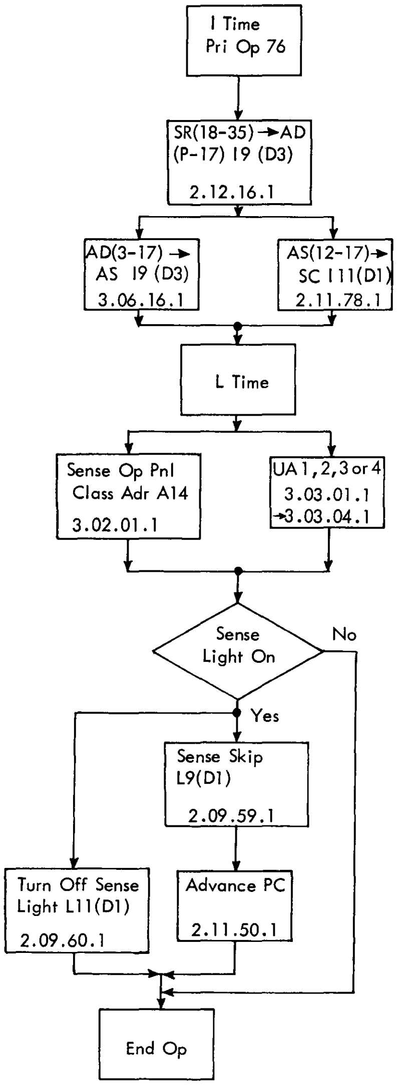


FIGURE 5.3-44. MSE -0760...0141, 0142, 0143, 0144

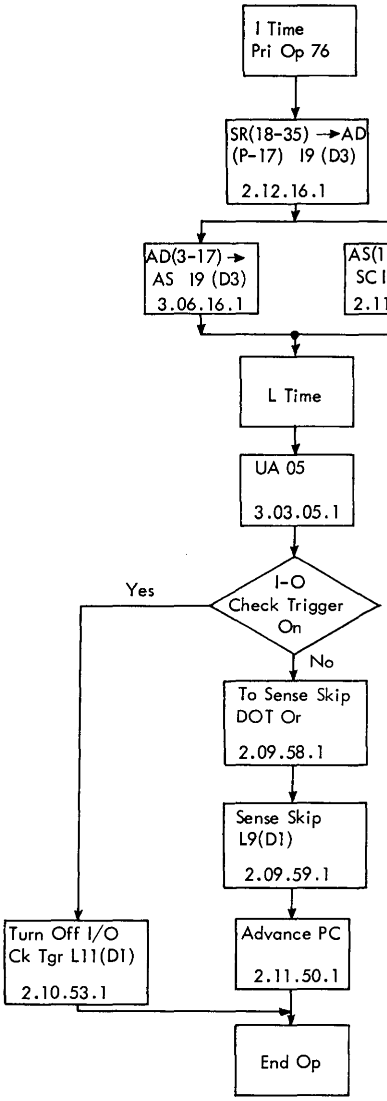


FIGURE 5.3-45. IOT +0760...0005

Beginning-of-Tape Test BTT +0760...XXXX (I, L) Figure 5.3-46

This instruction tests the status of the beginning-of-tape indicator in a particular data channel. Address 1000-10000 selects data channels A-H respectively. If the beginning-of-tape indicator for the selected channel has been turned on by a previous instruction, the indicator is turned off and the computer takes the next instruction in sequence. If the indicator is already off, the computer skips the next instruction.

End-of-Tape Test ETT -0760...XXXX (I, L) Figure 5.3-46

This instruction uses address 1000-10000 to select the data channel in which the end-of-tape indicator is to be tested. If the indicator is on, it is turned off and the computer takes the next instruction in sequence. If the indicator is off, the computer skips the next instruction.

5.3.08 Control Instructions

Control instructions are provided so the programmer may change the problem conditions or service the computer.

Halt and Proceed HPR +0420 (I, L) Figure 5.3-47

This instruction causes the computer to stop at the end of I time. When the start button is depressed, the computer proceeds to the next instruction in sequence. When halted, the PC contains the address of the next sequential instruction.

Halt and Transfer HTR +0000 (I, L) Figure 5.3-48

This instruction causes the computer to stop at the end of I time by turning on the master stop trigger at I_{10}(D1). Depressing the start button causes the computer to proceed in L time of a transfer operation and the computer takes an instruction transfer to location X. When halted, the PC contains the address of the HTR instruction.

No Operation NOP +0761 (I, L)

The NOP instruction performs no active function, but is used to reserve space for other instructions. Since this instruction has a primary operation 76, an I and an L cycle are required. The only function of this instruction is to turn on the end operation trigger to allow the computer to proceed to I time of the next sequential instruction. SOD O1 on Systems 3.07.01.1 causes L END OP on Systems 8.00.09.1.

Execute XEC +0522 (I) Figure 5.3-49

This instruction causes the computer to perform the instruction at location X. The program counter is not altered; therefore, after the instruction at location X has been executed, the computer proceeds to the next sequential instruction (instruction in next position beyond the XEC instruction). XEC prevents AR to PC on Systems 3.06.05.1.

If the instruction at location X is an unconditional transfer or a successful conditional transfer, the computer does not proceed to the next instruction following the XEC, but proceeds to the address specified by the transfer. Another exception occurs when the instruction at location X is a skip type, and the conditions are met for this skip. The PC is stepped and the skip is in relation to the location of the XEC instruction.

Set Sign Plus SSP +0760...0003 (I, L) Figure 5.3-50

This instruction places a zero (a plus) in the accumulator sign position. Positions (Q-35) of the accumulator are unchanged.

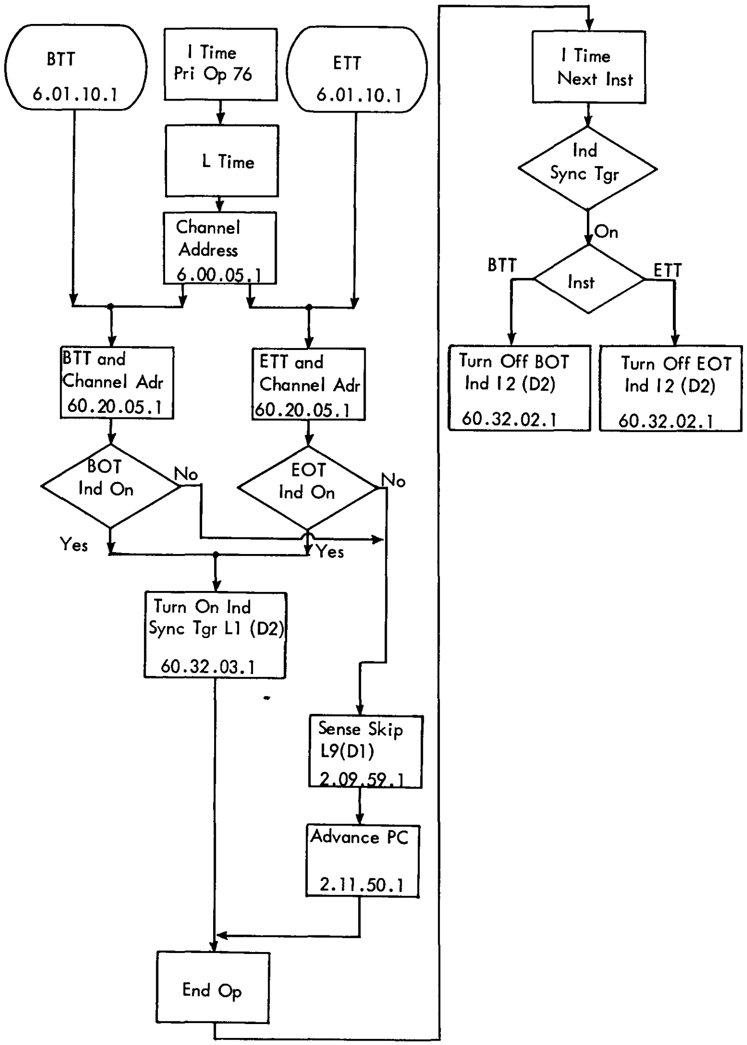


FIGURE 5.3-46. BTT +0760, ETT -0760

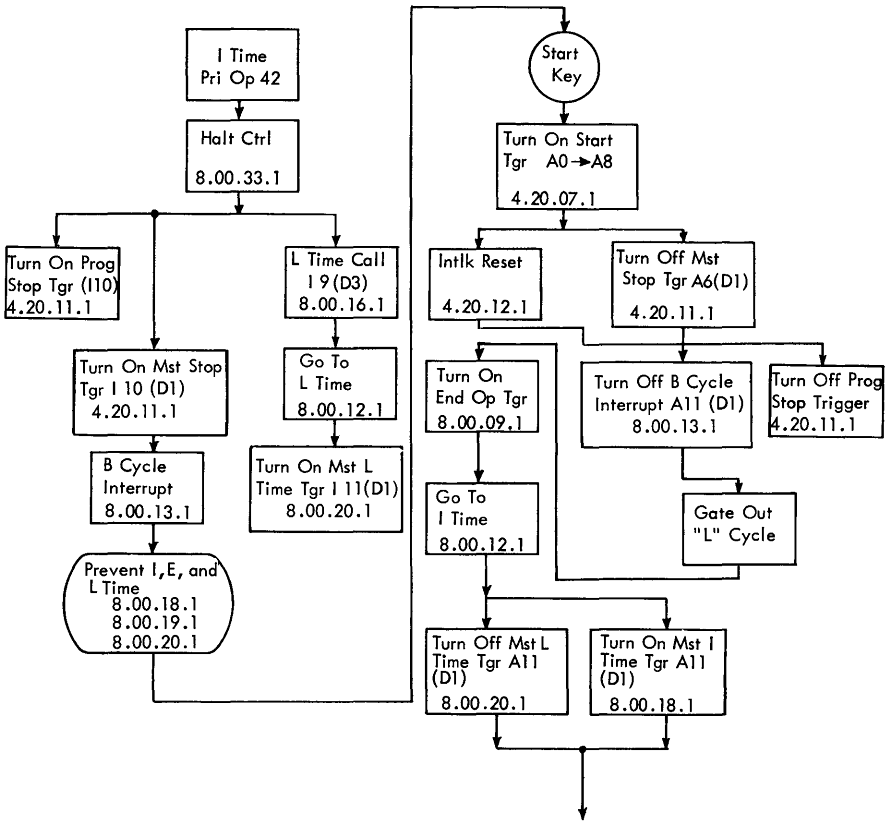


FIGURE 5.3-47. HPR +0420

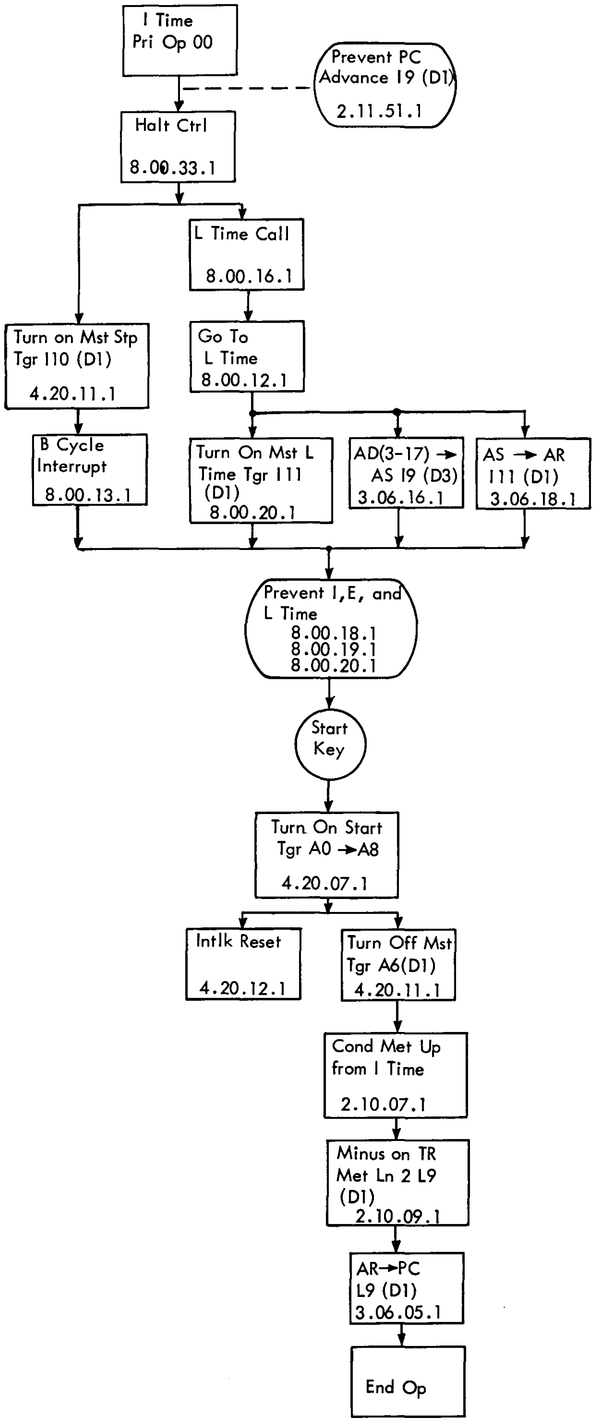


FIGURE 5.3-48. HTR + 0000

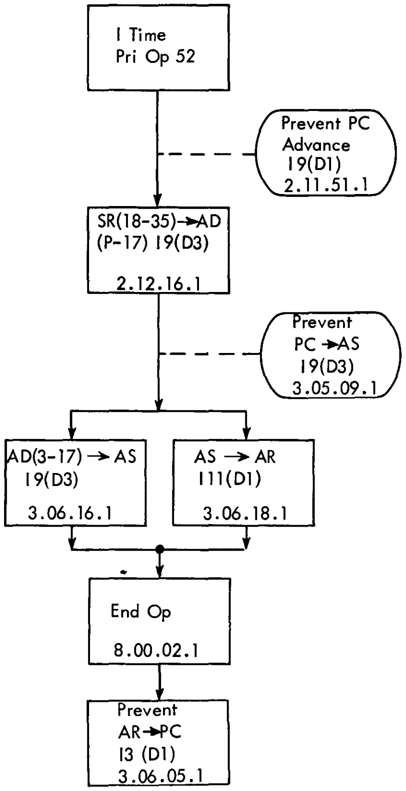


FIGURE 5.3-49. XEC + 0522

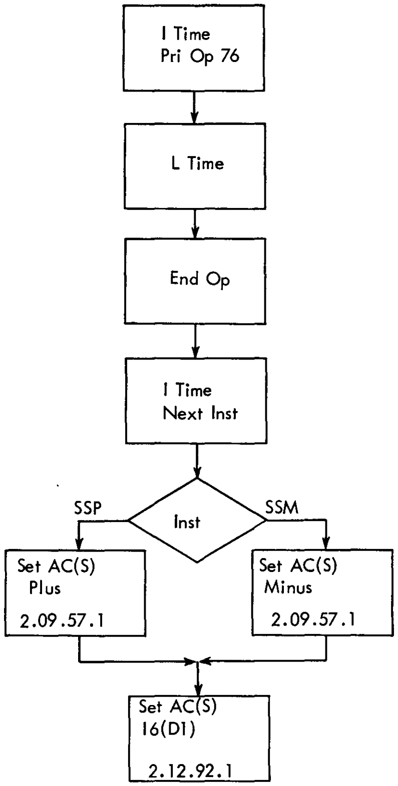


FIGURE 5.3-50. SSM - 0760...0003; SSP + 0760...0003

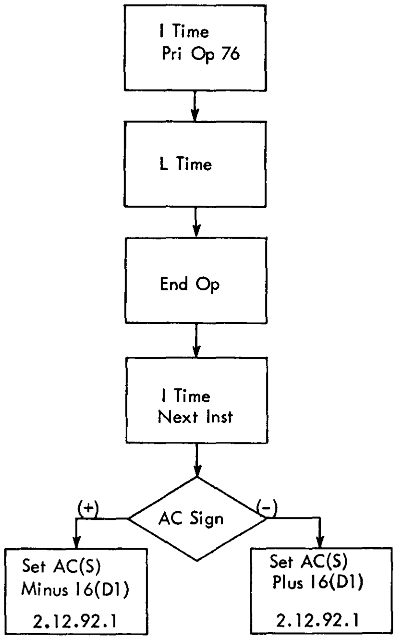


FIGURE 5.3-51. CHS +0760...0002

Set Sign Minus SSM -0760...0003(I, L) Figure 5.3-50

This instruction places a one (a minus) in the accumulator sign position. Positions (Q-35) of the accumulator are unchanged.

Change Sign CHS +0760...0002(I, L) Figure 5.3-51

This instruction complements the sign position of the accumulator. A one is replaced by a zero and a zero is changed to a one. Positions (Q-35) of the accumulator are unchanged.

5.3.09 Sense Indicator Instructions

The sense indicator instructions are a group of instructions which operate on the sense indicator register. These instructions enable the computer to set and test the indicators under program control.

Load Indicators LDI +0441(I, E) Figure 5.3-52

The contents of storage location X (S,1-35) are placed in indicator positons (0-35). The contents of storage are unchanged.

Store Indicators STI +0604(I, E)

The contents of indicator positions (0-35) replace the contents of storage location X. The indicators are unchanged. Execution of this instruction is the same as STO (Figure 3.5-1, Section 3.5.01) except that SI(0-35) is taken to the SR rather than AC (S,1-35). SI(0-35) to the SR(S,1-35) is shown on Systems 2.12.13.1.

OR Storage to Indicators OSI +0442(I, E) Figure 5.3-52

This instruction places the logical OR of the word at storage location X and the contents of the indicators in the sense indicator register. Storage is unchanged.

Invert Indicators from Storage IIS +0440(I, E) Figure 5.3-52

This instruction inverts the positions of the SI register which have corresponding "1" bits in the word at storage location X.

Reset Indicators from Storage RIS +0445(I, E) Figure 5.3-53

This instruction utilizes the word at storage location X to reset the sense indicator register position. A "1" bit in any position of the word causes the corresponding sense indicator trigger to be turned off. Indicator positions which correspond to "0" bits are unchanged. Storage is unchanged.

Set Indicators of Right Half SIR +0055(I) Figure 5.3-54

For this instruction, the control field (18-35) of the instruction is OR'ed with the right half of the sense indicator register. A "1" bit in either the control field or the indicator places a "1" bit in the corresponding sense indicator. Because the only

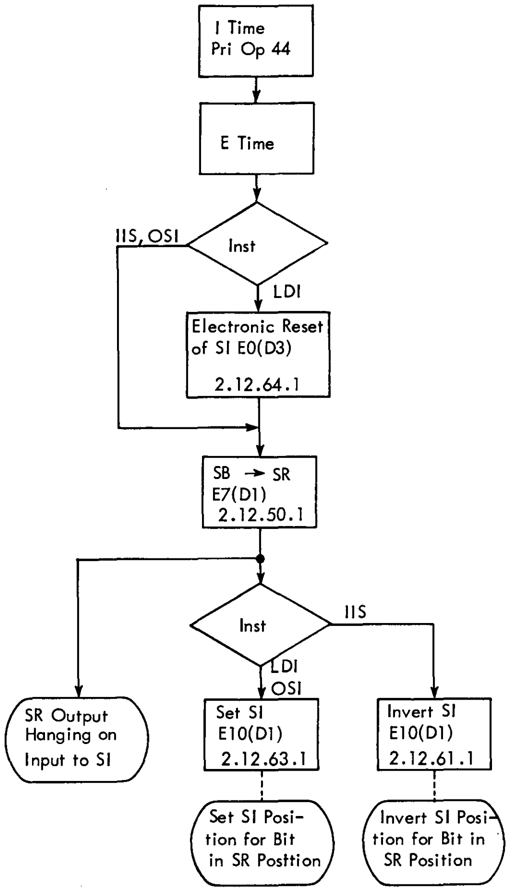

I Time
Pri Op 44
E Time
IIS, OSI
Inst
LDI
Electronic Reset of SI E0(D3)
2.12.64.1
SB → SR
E7(D1)
2.12.50.1
Inst
IIS
LDI
OSI
SR Output Hanging on Input to SI
Set SI
E10(D1)
2.12.63.1
Invert SI
E10(D1)
2.12.61.1
Set SI Position for Bit in SR Posttion
Invert SI Position for Bit in SR Position

FIGURE 5.3-52. LDI +0441; OSI +0442; IIS +0440

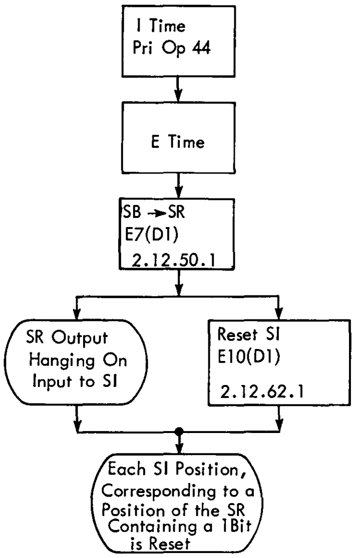

I Time
Pri Op 44
E Time
SB → SR
E7(D1)
2.12.50.1
SR Output Hanging On Input to SI
Reset SI
E10(D1)
2.12.62.1
Each SI Position, Corresponding to a Position of the SR Containing a 1Bit is Reset

FIGURE 5.3-53. RIS +0445

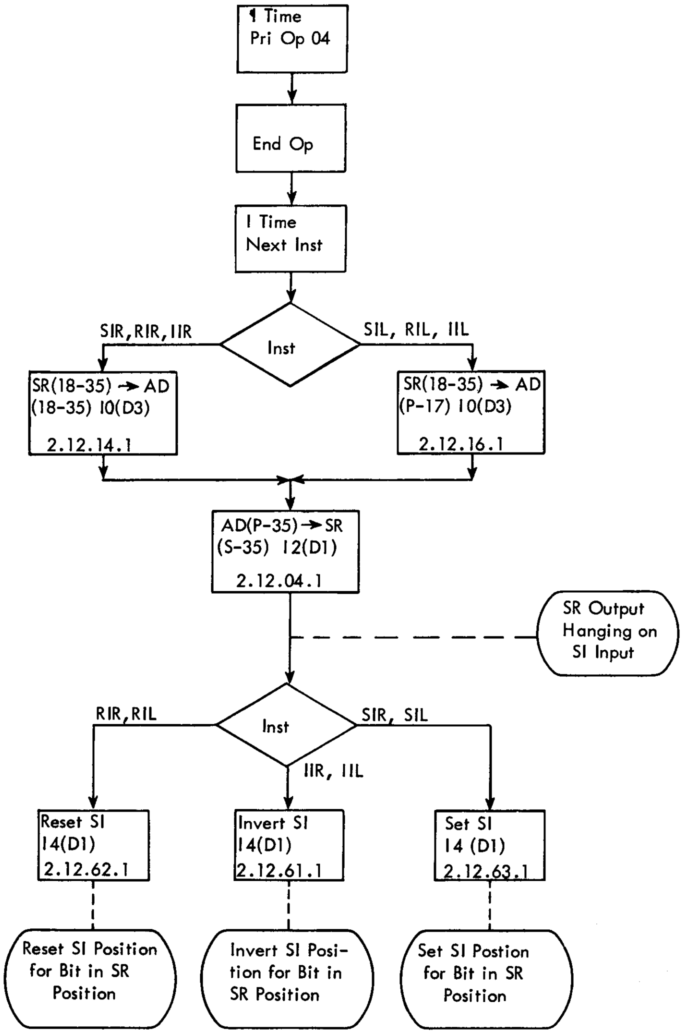


FIGURE 5.3-54 SIR + 0055; SIL - 0055; RIR + 0057; RIL - 0057; IIR + 0051; IIL - 0051

input to the sense indicator register is from the SR, the control field must be placed in the right half of the SR, and the left half of the SR must be cleared before setting the indicators. This is accomplished by gating only the right half of the SR to the adders and then gating adders (P, 1-35) to SR(S, 1-35).

Set Indicators of Left Half SIL -0055 (I) Figure 5.3-54

This instruction OR's the control field (18-35) of the instruction with the left half of the sense indicator register. Execution is the same as SIR except that the control field must be switched to the left half of the SR, and the right half of the SR must be cleared before the indicators can be set. This is done by routing SR(18-35) to AD(P-17).

Reset Indicators of Right Half RIR +0057 (I) Figure 5.3-54

This instruction causes the reset of the positions of the right half of the sense indicator register which correspond to ones in the control field (18-35) of the instruction. Indicator positions corresponding to zeros are unchanged. Execution of this instruction is identical to SIR, except that the reset-indicators input is used instead of set indicators.

Reset Indicators of Left Half RIL -0057 (I) Figure 5.3-54

For this instruction, the control field of the instruction is used as a reset mask for the left half of the indicator register. A one in the control field resets the corresponding indicator position. Execution of this instruction is similar to that of SIL except that the reset-indicators pulse is used.

Invert Indicators of Right Half IIR +0051 (I) Figure 5.3-54

This instruction inverts positions of the right half of the indicator register which correspond to ones in the control field of the instruction. Indicator positions corresponding to zeros in the control field are unchanged. Execution is identical to that of SIR except that the invert-indicators pulse to the indicator input is used.

Invert Indicators of Left Half IIL -0051 (I) Figure 5.3-54

This instruction inverts positions of the left half of the indicator register which correspond to ones in the control field (18-35) of the instruction. Indicator positions corresponding to zeros in the control field are unchanged. Execution of this instruction is identical to that of SIL, except that an invert-indicators pulse is used.

Place Indicator in Accumulator PIA -0046 (I) Figure 5.3-55

The contents of sense indicators (0-35) are placed in positions (P, 1-35) of the accumulator. The sense indicators are unchanged. AC positions Q and S are cleared.

Place Accumulator in Indicators PAI +0044 (I) Figure 5.3-55

The contents of the accumulator (P, 1-35) are placed in the sense indicator register. The accumulator is unchanged.

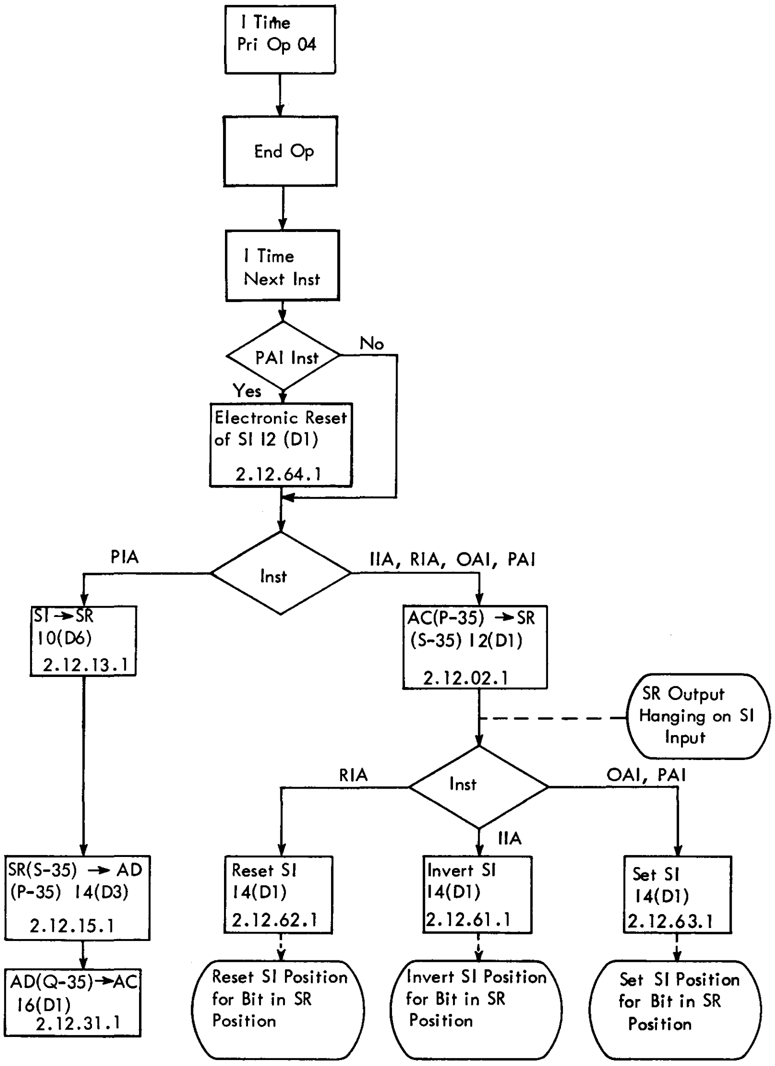


FIGURE 5.3-55. PAI +0044; PIA - 0046; OAI +0043;
RIA - 0042; IIA +0041

OR Accumulator to Indicators   OAI +0043 (I)   Figure 5.3-55

The logical OR of the contents of the accumulator (P,1-35) and the indicators is placed in the sense indicator register. If a position in either register contains a one, a one will be placed in that position of the indicator. The accumulator is unchanged. Execution is accomplished using the sequence of PAI but omitting the indicator reset.

Reset Indicators from Accumulator   RIA -0042 (I)   Figure 5.3-55

The positions of the sense indicator register which correspond to the positions of the AC(P,1-35) having "1" bits are reset to zero. Indicator positions which correspond to "0" bits are unchanged. The AC is unchanged.

Invert Indicators from Accumulator   IIA +0041 (I)   Figure 5.3-55

This instruction inverts any sense indicator position for which there is a corresponding "1" bit in the accumulator (P,1-35).

Transfer if Indicators On   TIO +0042 (I, L)   Figure 5.3-56

If all of the ones in the AC are matched by ones in the indicator register, an instruction transfer is taken to location X. AC positions containing zeros are not compared with indicator positions. If all of the ones are not matched, the computer takes the next instruction in sequence. To execute this instruction, the complement of the AC is OR'ed with the indicator. If the ones match, the result will be all ones. The result of the OR is stored in the SR and fed to the adder. A carry to adder (35) will ripple down the adder and carry out of the AD(P). The adder (P) carry is used to condition the transfer.

Transfer if Indicators Off   TIF +0046 (I, L)   Figure 5.3-56

If all of the ones in the AC are matched by zeros in the indicator register, an instruction transfer is taken to location X. AC positions containing zeros are not compared with indicator positions. If all of the ones are not matched, the computer takes the next instruction in sequence. Execution of this instruction is identical to that of TIO, except that the complement of the indicators is OR'ed to the complement of the AC. Test procedure remains the same.

On Test for Indicators   ONT +0446 (I, E, 2L)   Figure 5.3-57

If the ones contained in the word stored at location X are matched by ones in the corresponding indicator register positions, the next instruction will be skipped. If all of the ones are not matched, the computer will take the next instruction in sequence. Positions of storage location X which contain zeros are not compared. Execution of this instruction requires an E cycle to obtain the test word from storage and two L cycles to complete the test. The test is accomplished by moving the test word to the accumulator, where it can be complemented. The complement is OR'ed with the indicators and returned to the SR. The result of the OR will be all ones if the ones in the test word match the indicator. To test for all ones a carry is added to the OR which will result in an adder (P) carry to condition the advance instruction counter. The contents of the accumulator are saved and restored to normal during execution of the instruction.

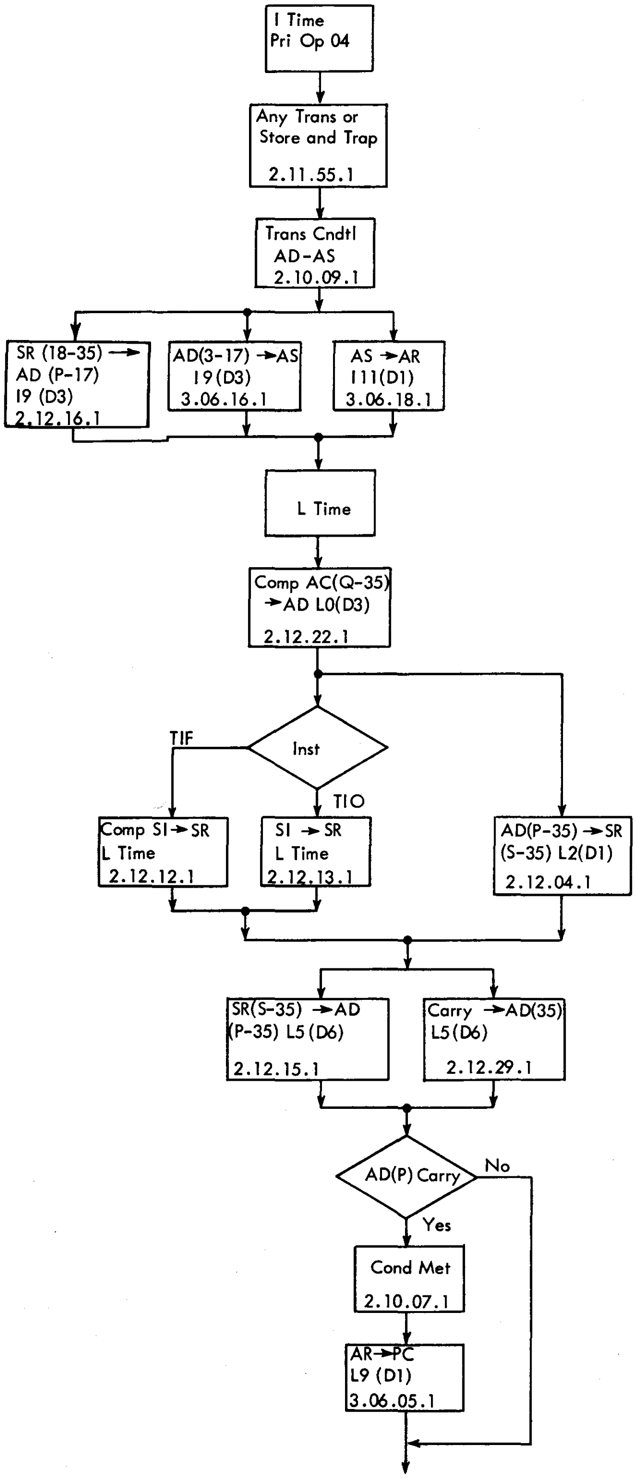


FIGURE 5.3-56. TIO + 0042; TIF + 0046

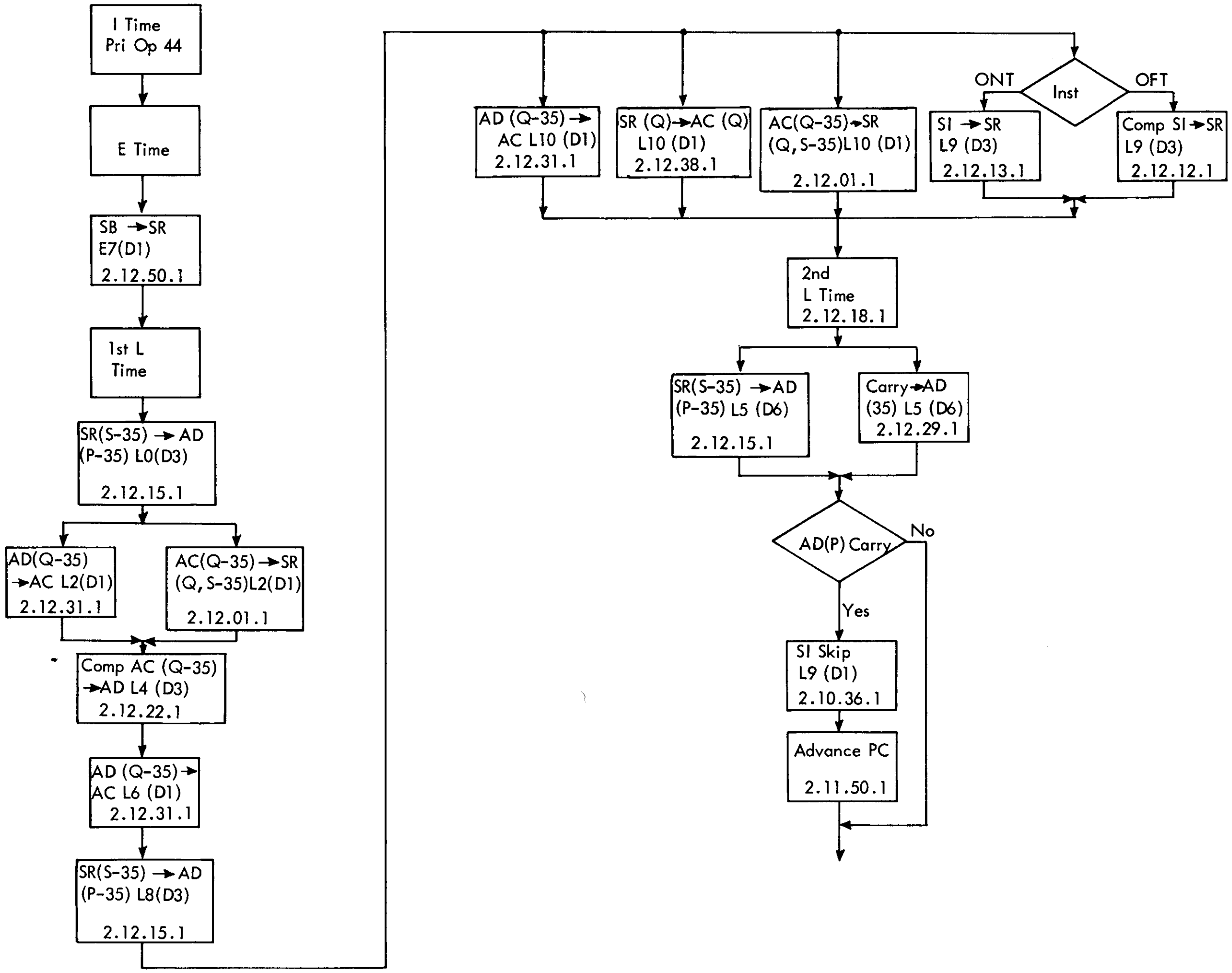


FIGURE 5.3-57. ONT +0446, OFT +0444

Off Test for Indicators OFT +0444 (I, E, L, L) Figure 5.3-57

If the ones contained in the word stored at location X are matched by zeros in the corresponding indicator positions, the computer will skip one instruction. If all of the ones are not matched by zeros, the computer will take the next instruction in sequence. Positions in the test word from storage which contain zeros are not compared. Execution of this instruction is identical to that of ONT, except that the complement of the indicator is used for the OR operation instead of the true indicator.

Right-Half Indicators, On Test RNT +0056 (I, L, L) Figure 5.3-58

This instruction matches the ones in the control field (18-35) of the instruction against the corresponding positions of the right half of the indicator register. If all of the ones are matched by ones, the computer skips one instruction. If all are not matched, the computer takes the next instruction in sequence. Positions of the control field containing zeros are not compared. Execution of this instruction is the same as ONT except that only positions (18-35) of SR are used.

Left-Half Indicators, On Test LNT -0056 (I, L, L) Figure 5.3-58

If the ones in the control field of the instruction are matched by ones in the corresponding positions of the left half of the indicator register, the computer will skip the next instruction. If all ones are not matched, the computer takes the next instruction in sequence. Execution of this instruction is the same as ONT, except that the SR (18-35) is compared against SI(0-17).

Right-Half Indicators, Off Test RFT +0054 (I, L, L) Figure 5.3-58

If the ones in the control field of this instruction are matched by zeros in the right half of the indicator register, the computer will skip one instruction. If all ones are not matched by zeros, the computer will take the next instruction in sequence. Positions of the control field containing zeros are not compared. This instruction is executed the same as OFT, except that only positions (18-35) of the SR are used.

Left-Half Indicators, Off Test LFT -0054 (I, L, L) Figure 5.3-58

If the ones in the control field of this instruction are matched by zeros in the left half of the indicator register, the computer will skip one instruction. If all are not matched, the computer will take the next instruction in sequence. Positions of the control field containing zeros are not compared. The control field and the indicator register are unchanged. Execution of this instruction is the same as OFT, except that SR(18-35) is compared with SI(0-17).

5.3.10 Index Transmission Instructions

These are the instructions which operate on or with the index registers. They are used to load or modify the index registers, to alter instructions from the index registers, or to control program flow.

There are three shift cell index registers containing 15 positions each. The tag bit positions (18-20) of the instruction control which index registers are to be used. Any

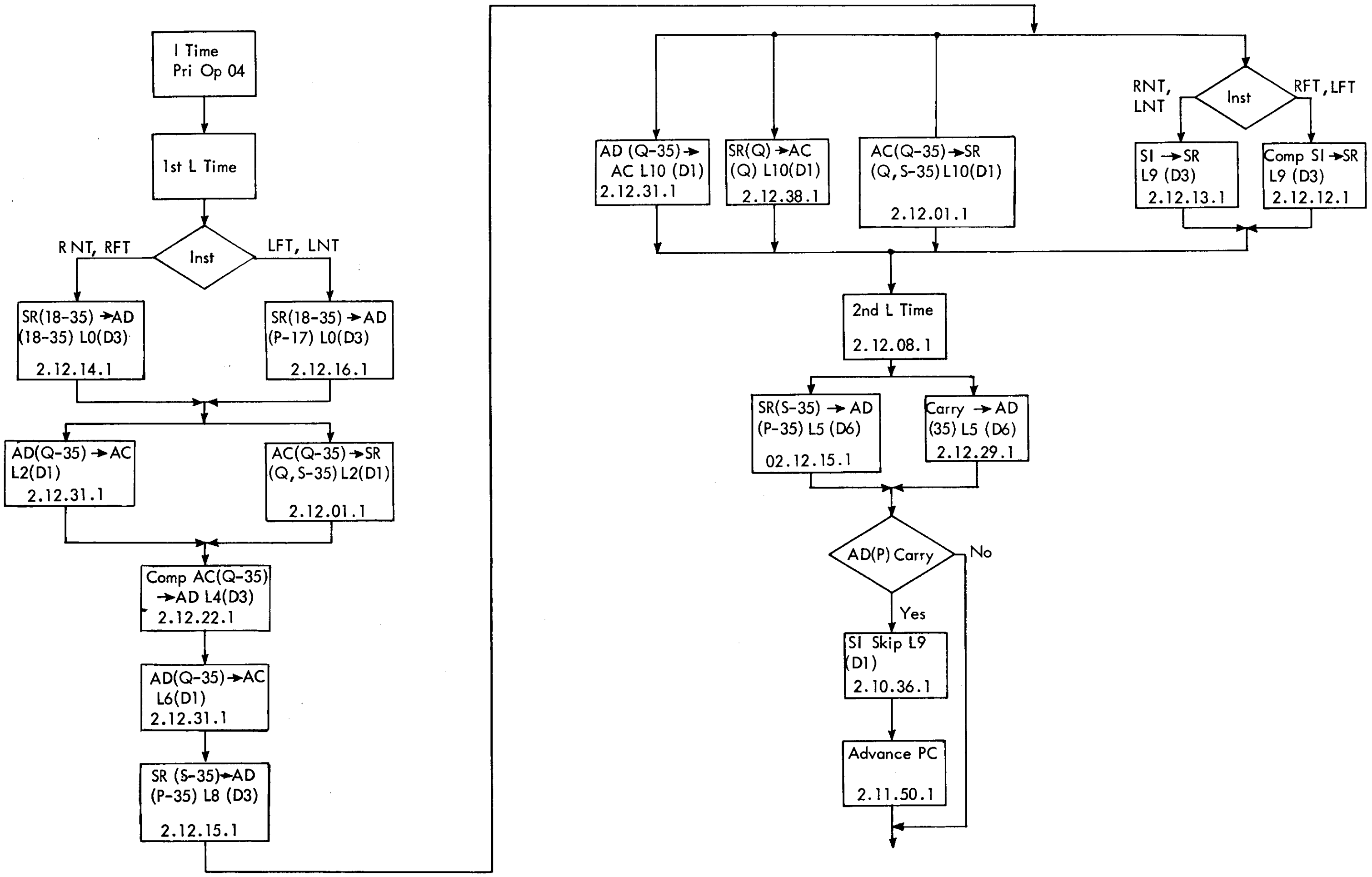


FIGURE 5.3-58. RNT +0056; LNT - 0056; RFT +0054; LFT - 0054

index register or combination of index registers may be selected. The tag bits are, retained by the tag register on Systems 2.08.01.1. SB(18-20) outputs are used to turn on the tag register triggers during I time of the instruction. The output of the index register, when gated to the adders, is always complemented. To make this a 2's complement, a carry is gated to adder (17) at the same time the XR is gated to the AD.

Many instructions are indexable, as controlled by their tag positions, allowing the address portion of the instruction to be modified. Appendix A shows which instructions are indexable.

Transfer with Index Incremented TXI +1000 (I, L) Figure 5.3-59

This transfer adds its decrement to the specified index register and causes an unconditional transfer to location X. The index register outputs are always complemented to the adders. Thus, to add the decrement, the index register must be cycled through the adders so they contain a complement before the addition.

Transfer on Index TIX +2000 (I, L) Figure 5.3-60

If the number in the specified index register is greater than the decrement, the contents of the index register will be reduced by the amount of the decrement, and an instruction transfer will be taken to storage location X. When the number in the index register is equal to or less than the decrement, no reduction is made and the computer takes the next instruction in sequence. The comparison between the index register contents and the decrement is made by gating the decrement and the 2's complement of the index register contents to the adders. No carry from adder (3) indicates that the index register contents are greater and that the transfer and index register reduction are to be made. A carry from adder (3) will, therefore, block the transfer and index register reduction.

Transfer on No Index TNX -2000 (I, L) Figure 5.3-60

If the number in the specified index register is greater than the decrement, the contents of the index register will be reduced by the amount of the decrement, and the computer will proceed to the next instruction in sequence. When the number in the index register is equal to or less than the decrement, no reduction is made, but an instruction transfer will be taken to storage location X.

The sequence of operations for this instruction is like that of TIX, except that the conditional transfer circuits are activated with the adder (3) carry trigger on.

Transfer on Index High TXH +3000 (I, L) Figure 5.3-60

If the number in the specified index register is greater than the decrement, an instruction transfer is taken to storage location X. If the number in the index register is less than or equal to the decrement, the computer takes the next instruction in sequence. Execution of this instruction is identical to that of TIX. Because no reduction is to be made, the adders are not routed to the index registers.

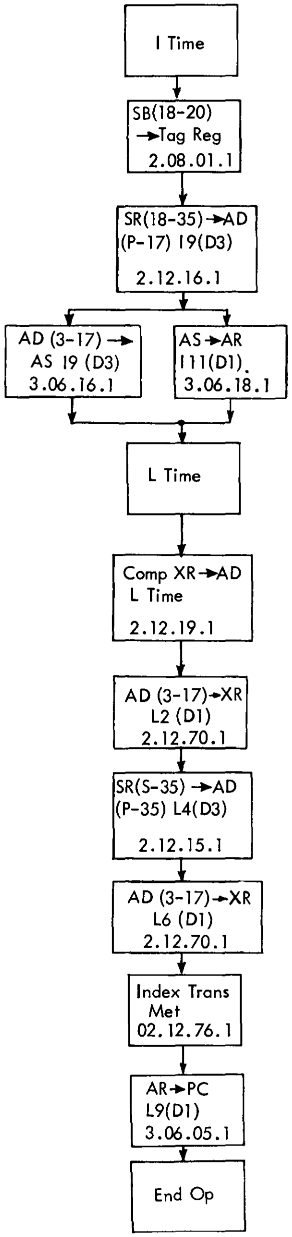

I Time
SB(18-20) →Tag Reg 2.08.01.1
SR(18-35)→AD (P-17) 19(D3) 2.12.16.1
AD (3-17) → AS 19 (D3) 3.06.16.1
AS→AR 111(D1). 3.06.18.1
L Time
Comp XR→AD L Time 2.12.19.1
AD (3-17)→XR L2 (D1) 2.12.70.1
SR(S-35) →AD (P-35) L4(D3) 2.12.15.1
AD (3-17)→XR L6 (D1) 2.12.70.1
Index Trans Met 02.12.76.1
AR→PC L9(D1) 3.06.05.1
End Op

FIGURE 5.3-59. TXI +1000

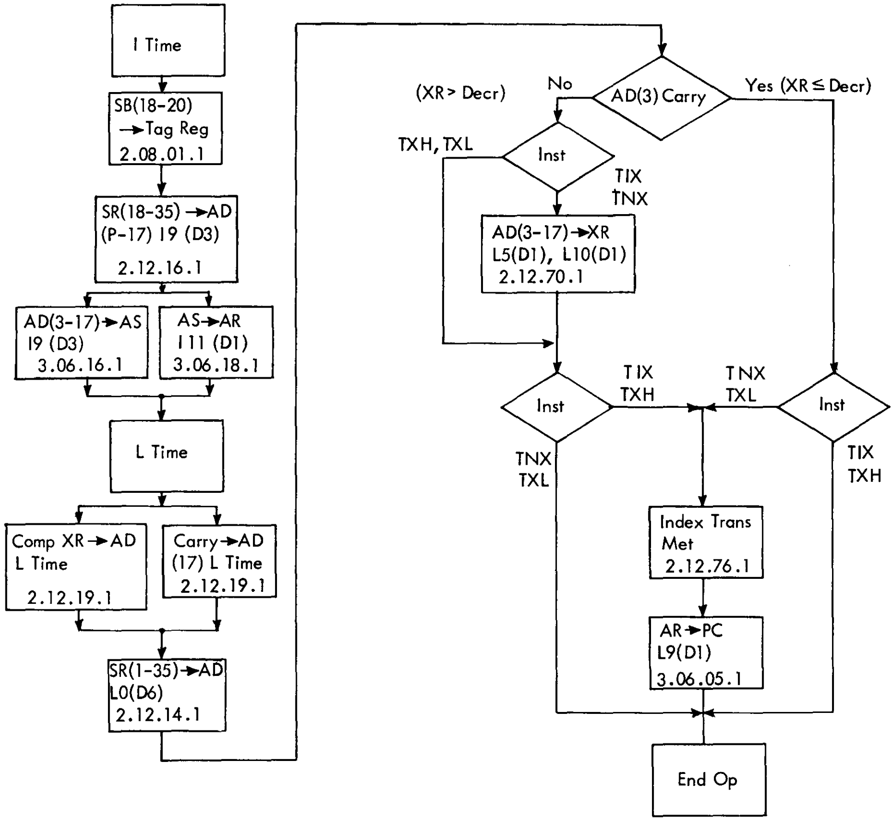

I Time
SB(18-20) →Tag Reg 2.08.01.1
SR(18-35)→AD (P-17) 19 (D3) 2.12.16.1
AD(3-17)→AS 19 (D3) 3.06.16.1
AS→AR 111 (D1) 3.06.18.1
L Time
Comp XR→AD L Time 2.12.19.1
Carry→AD (17) L Time 2.12.19.1
SR(1-35)→AD L0(D6) 2.12.14.1
AD(3) Carry
(XR> Decr)
No
Yes (XR≤ Decr)
Inst
TXH, TXL
TIX
TNX
AD(3-17)→XR L5(D1), L10(D1) 2.12.70.1
Inst
TIX
TXH
TNX
TXL
Inst
TNX
TXL
TIX
TXH
Index Trans Met 2.12.76.1
AR→PC L9(D1) 3.06.05.1
End Op

FIGURE 5.3-60. TIX +2000; TNX - 2000; TXH +3000; TXL - 3000

Transfer on Index Low or Equal TXL -3000 (I, L) Figure 5.3-60

If the number in the specified index register is greater than the decrement, the computer takes the next instruction in sequence. If the number in the index register is equal to or less than the decrement, an instruction transfer will be taken to storage location X. Execution of this instruction is identical to that of TNX except that, because no reduction is required, the adders will not be routed to the index register.

Transfer and Set Index TSX +0074 (I, L) Figure 5.3-61

This instruction places the 2's complement of the instruction counter (the location of the TSX instruction) in the specified index register and causes an instruction transfer to storage location X. Execution of this instruction requires, in addition to the normal transfer controls, the routing of the instruction counter to the index register. This involves use of the address switch, storage register, and adders. To obtain the required 2 's complement, the index register contents must also be cycled through the adders after receiving the contents of the instruction counter.

Place Address in Index PAX +0734 (I) Figure 5.3-62

This instruction places the contents of AC(21-35) in the specified index register in true form. To shift the address field to the decrement positions of the adders, AC (21-35) is routed through the SR to AD(P-17).

Place Decrement in Index PDX -0734 (I) Figure 5.3-62

This instruction places the true contents of AC(3-17) in the specified index register. Because no shift is required to execute this instruction, AC(3-17) is sent to AD(3-17) through the SR.

Place Complement of Address in Index PAC +0737 (I) Figure 5.3-62

This instruction places the 2's complement of AC(21-35) in the specified index register. The AC is unchanged. The complementing is accomplished by putting the address in the XR and then cycling the XR through the adders.

Place Complement of Decrement in Index PDC -0737 (I) Figure 5.3-62

This instruction loads the 2's complement of AC(3-17) in the specified index register. The AC is unchanged. The complementing is accomplished in the same manner as in PAC.

Place Index in Address PXA +0754 (I) Figure 5.3-63

This instruction places the true contents of the specified index register in AC(21-35). Positions (S, Q, P-20) of the AC are cleared. To obtain the true contents of the index register for executing this instruction, the index register contents are cycled through the adders. The index register contents are then rerouted to the adders and routed to the storage register by way of the address switch to shift the index register contents to the address field.

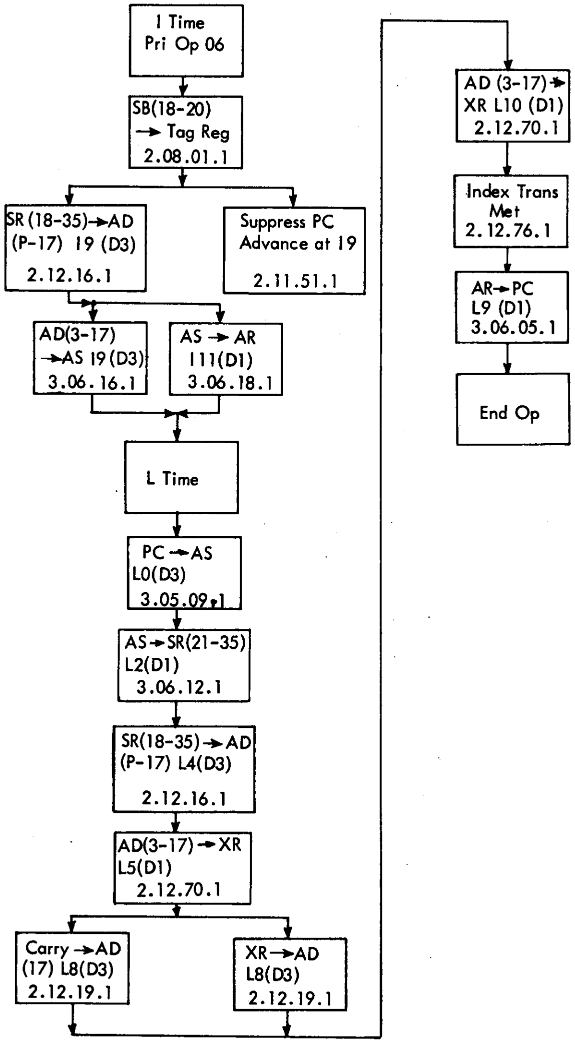


FIGURE 5.3-61. TSX +0074

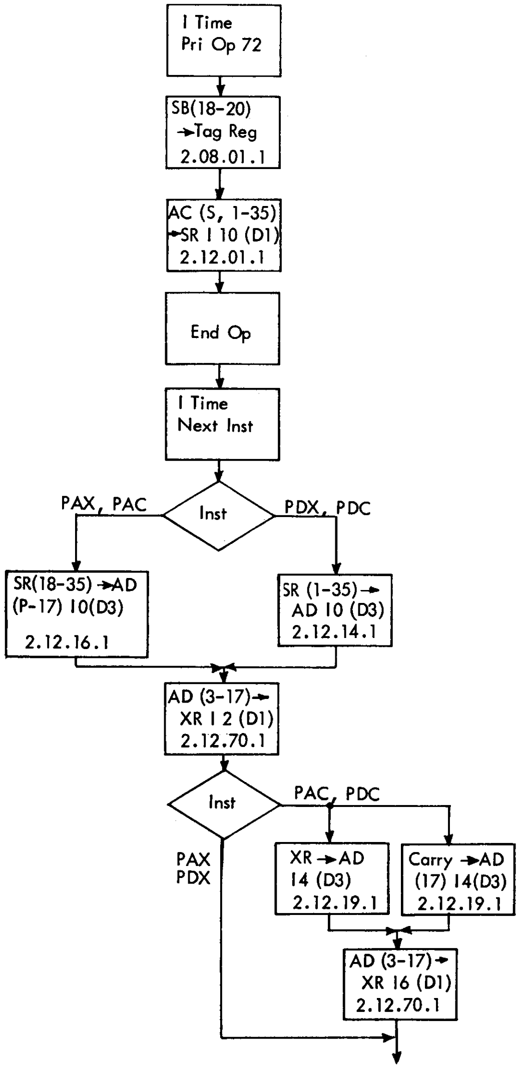


FIGURE 5.3-62. PAX +0734; PDX -0734; PAC +0737; PDC -0737

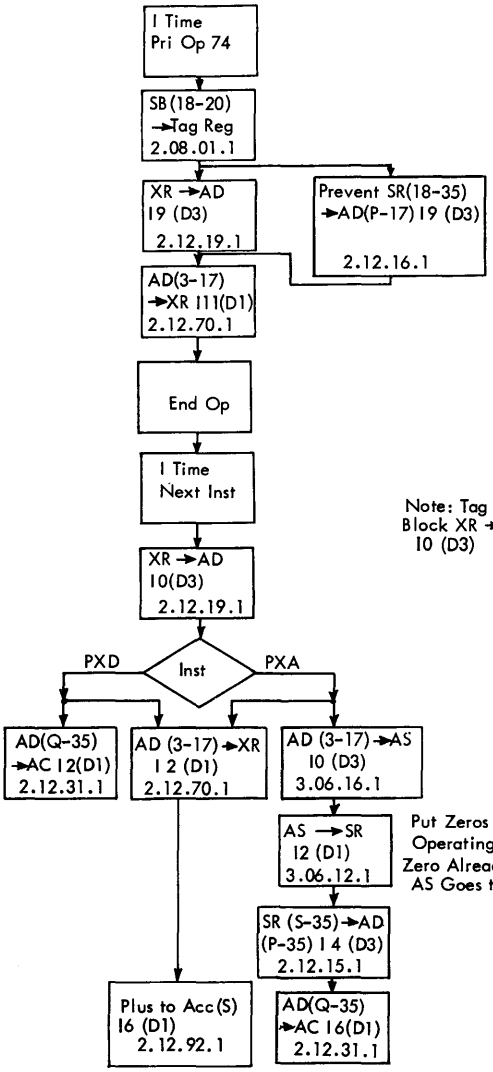


FIGURE 5.3-63. PXA +0754; PXD - 0754

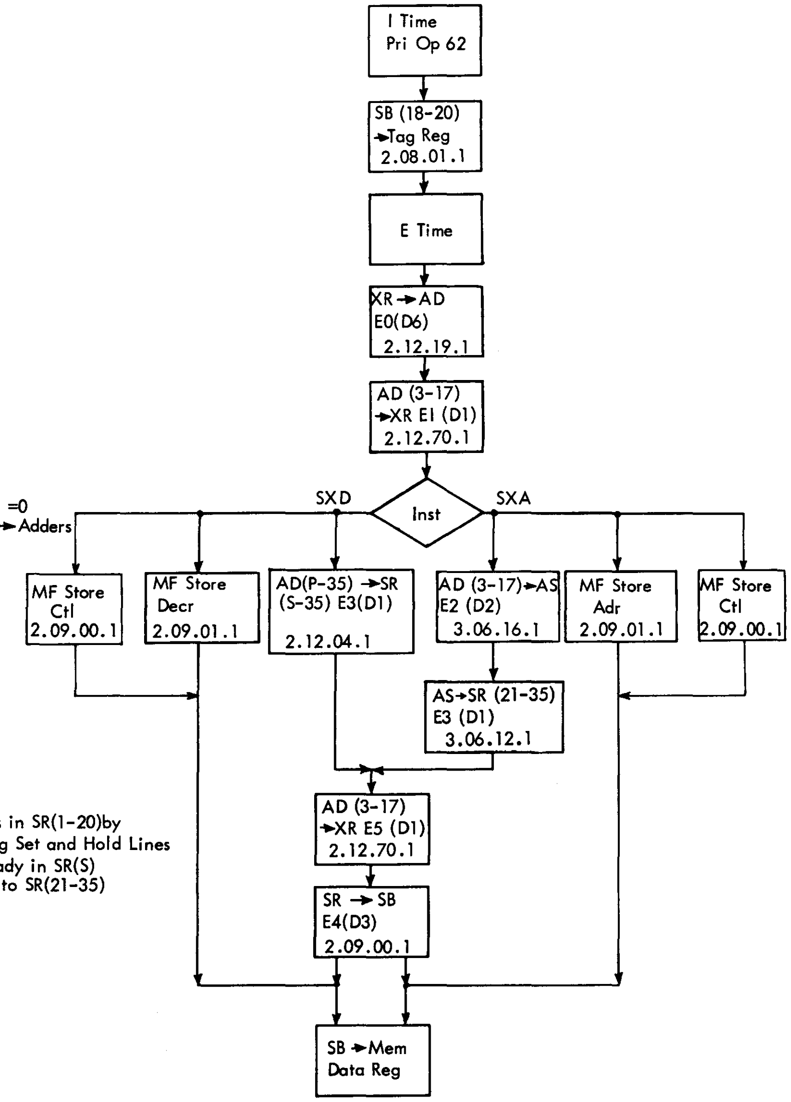


FIGURE 5.3-64. SXA +0634; SXD - 0634

Place Index in Decrement PXD -0754 (I) Figure 5.3-63

This instruction places the true contents of the specified index register in positions (3-17) of the AC. The remainder of the AC is cleared. Execution of this instruction is similar to PXA. However, because no shift is required, the index register contents can be routed directly from the adders to the AC.

Store Index in Address SXA +0634 (I, E) Figure 5.3-64

This instruction stores the true contents of the specified index register in the address field (21-35) of storage location X. The remaining positions of location X are unchanged. The logic of the execution of this instruction is similar to PXA. However, an E cycle is needed to put the information in core storage.

Store Index in Decrement SXD -0634 (I, E) Figure 5.3-64

This instruction stores the true contents of the specified index register in the decrement (3-17) of storage location X. The remaining positions of location X are unchanged. Execution of this instruction is similar to SXA but, because there is no shift required, the output of the adders goes directly to the SR.

Address to Index True AXT +0774 (I) Figure 5.3-65

This instruction loads the specified index register with the address field (21-35) in true form.

Address to Index Complemented AXC -0774 (I) Figure 5.3-65

This instruction loads the specified index register with the 2's complement of the address portion (21-35) of the instruction. Execution of AXC is similar to that of AXT except that the index register contents must be cycled through the adders during the following I cycle to obtain the 2's complement.

Load Complement of Address in Index LAC +0535 (I, E) Figure 5.3-66

This instruction loads the 2's complement of the address (21-35) of storage location X into the specified index register. Execution of this instruction requires an E cycle to obtain the word from storage. The address portion is transferred to the index register by way of the adders at the end of the cycle,and, to obtain the 2's complement, the index register is cycled during the following I cycle.

Load Complement of Decrement in Index LDC -0535 (I,E) Figure 5.3-66

This instruction loads the 2's complement of the decrement field (3-17) of storage location X into the specified index register. Execution of this instruction is identical to that of LAC except that the SR decrement, instead of the address, goes to the adders.

Load Index from Address LXA +0534 (I,E) Figure 5.3-66

This instruction loads the specified index register with the contents of the address field (21-35) of storage location X in true form. Execution of this instruction is identical to that of LAC, except that the complementing operation is not required.

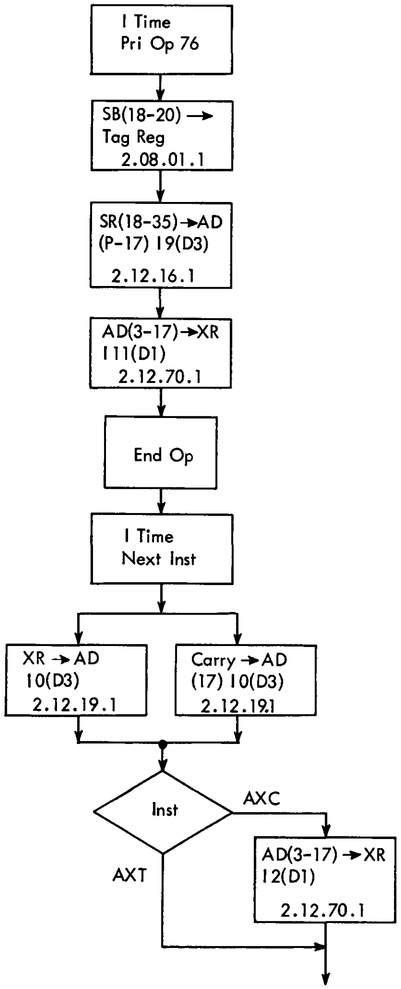


FIGURE 5.3-65. AXT +0774; AXC - 0774

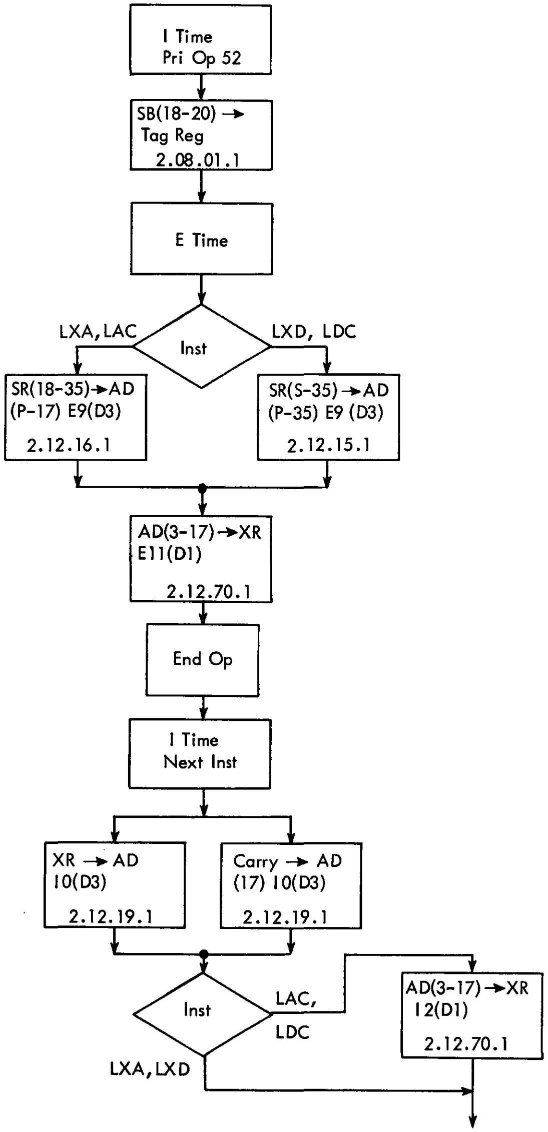


FIGURE 5.3-66. LXA +0534; LXD -0534; LAC +0535; LDC -0535

Load Index from Decrement LXD -0534 (I, E) Figure 5.3-66

This instruction loads the specified index register with the contents of the decrement field (3-17) of storage location X in true form. Execution of LXD is identical to that of LDC, except that the complementing operation is not used.

5.3.11 AND and OR Instructions

The AND and OR instructions are used to produce logical combinations of bits. These are useful for masking or matching words.

OR to Storage ORS -0602 (I, E) Figure 5.3-67

This instruction stores the logical OR of the word stored at location X and the AC (P, 1-35) in location X. The logical OR is obtained by matching the two words and storing ones in all positions which have ones in either word. For this instruction, the OR is developed in the memory data register during the E cycle. The words from core storage and the SB both go the memory data register and the result in the memory data register is put back into core storage.

OR to Accumulator ORA -0501 (I, E) Figure 5.3-68

This instruction places the logical OR of the word stored at location X and AC(P-35) in the accumulator. The OR is produced by matching the two words and placing ones in all positions of the AC which have ones in either word. The OR is developed in the SR by gating both the storage and AC words into the SR at the same time. S and Q remain unchanged.

AND to Accumulator ANA -0320 (I, E, L) Figure 5.3-69

This instruction places the logical AND of the word stored at location X and the AC (P, 1-35) in the accumulator (P, 1-35). The logical AND is obtained by matching the two words and placing ones in all positions of the accumulator which have corresponding ones in both the AC and storage. This instruction is executed by OR'ing the complement of both words and then complementing this sum.

AND to Storage ANS +0320 (I, E, L, E) Figure 5.3-69

This instruction stores the logical AND of the word stored at location X and the contents of the AC(P, 1-35) in location X. The logical AND is obtained by matching the two words and placing ones in all positions of storage location X which have corresponding ones in both the AC and storage. The contents of the AC(S, Q, P, 1-35) are unchanged. The AND is accomplished the same for this instruction as for ANA. An additional E cycle is required to put the AND in storage. The complement of the original AC (P, 1-35) is returned to the AC and re-complemented to restore the contents of AC(P, 1-35). During the execution, the original AC(Q) is saved in SR(Q) and then returned to AC(Q).

Exclusive OR to Accumulator ERA +0322 (I, E, L) Figure 5.3-70

This instruction matches AC(P, 1-35) with the logical word stored at location X. Zeros replace all positions of the AC which match the corresponding positions of the logical word. Accumulator positions (S) and (Q) are cleared.

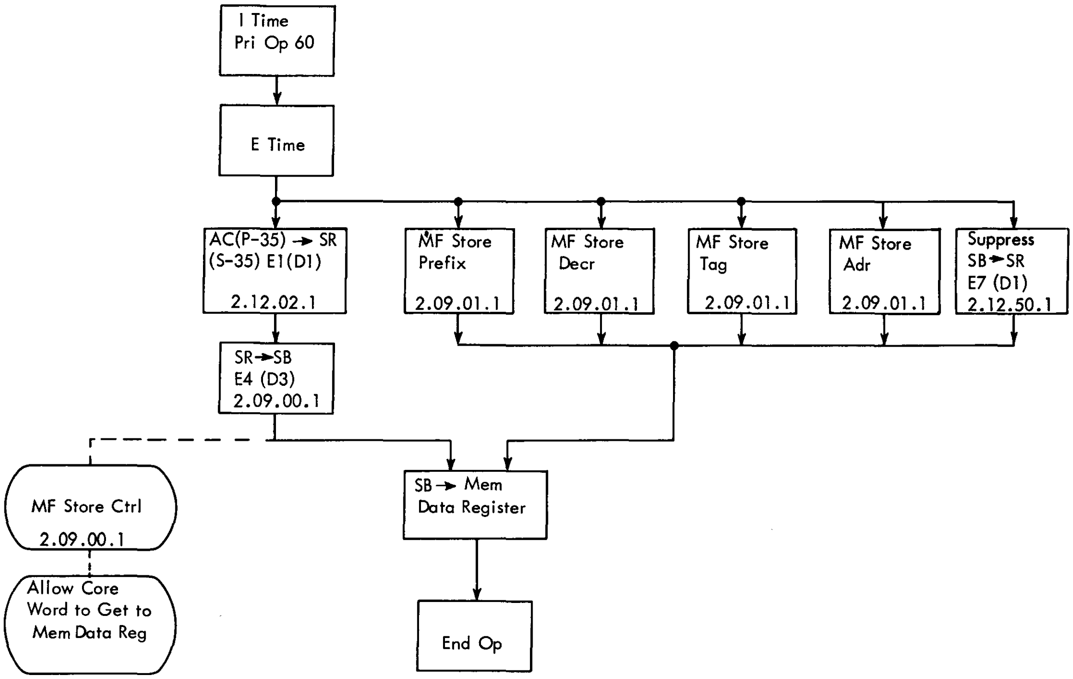


FIGURE 5.3-67. ORS - 0602

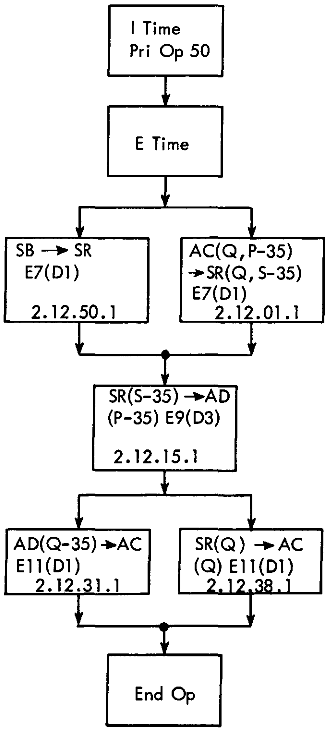


FIGURE 5.3-68. ORA - 0501

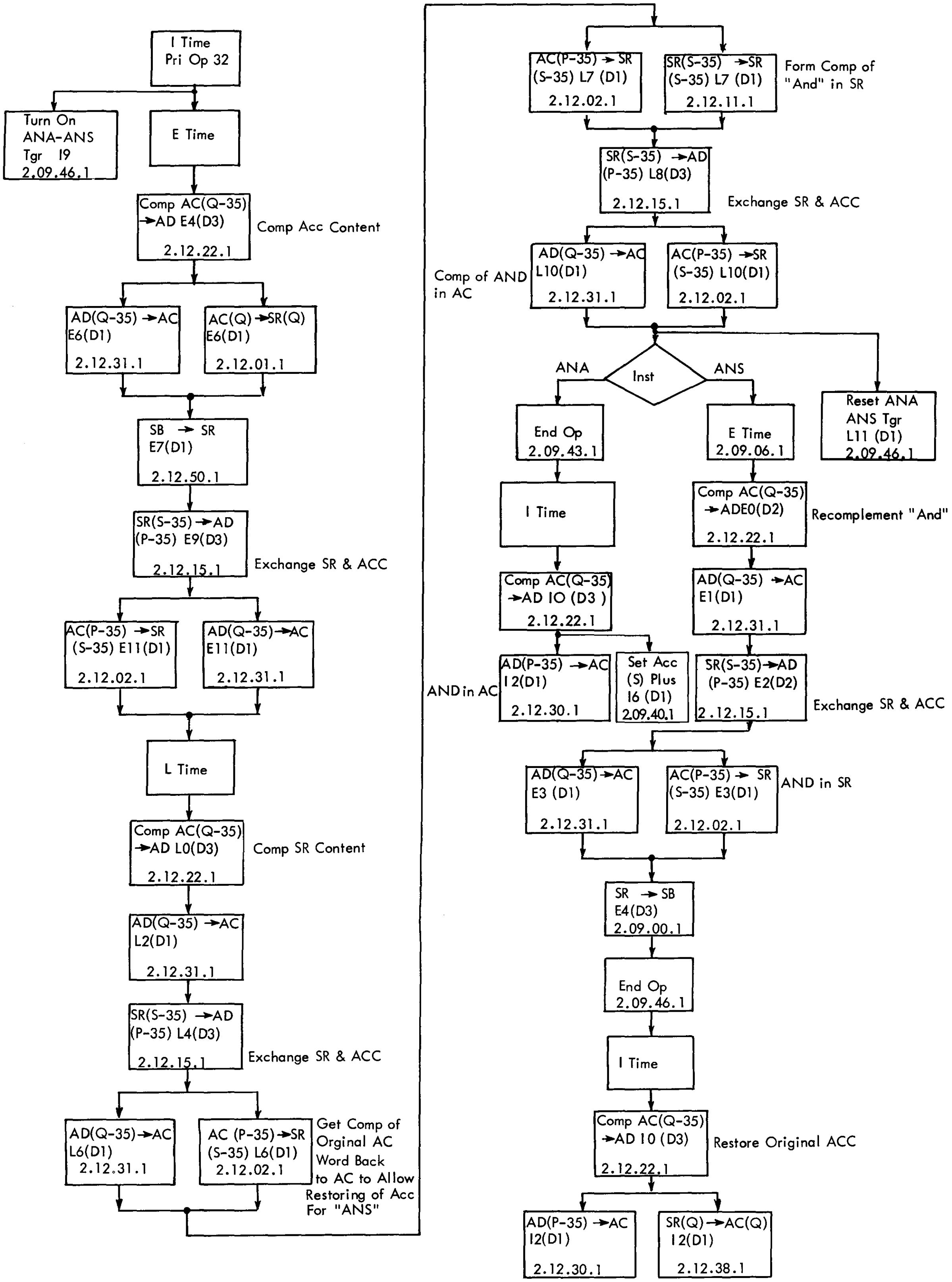


FIGURE 5.3-69. ANA - 0320; ANS +0320

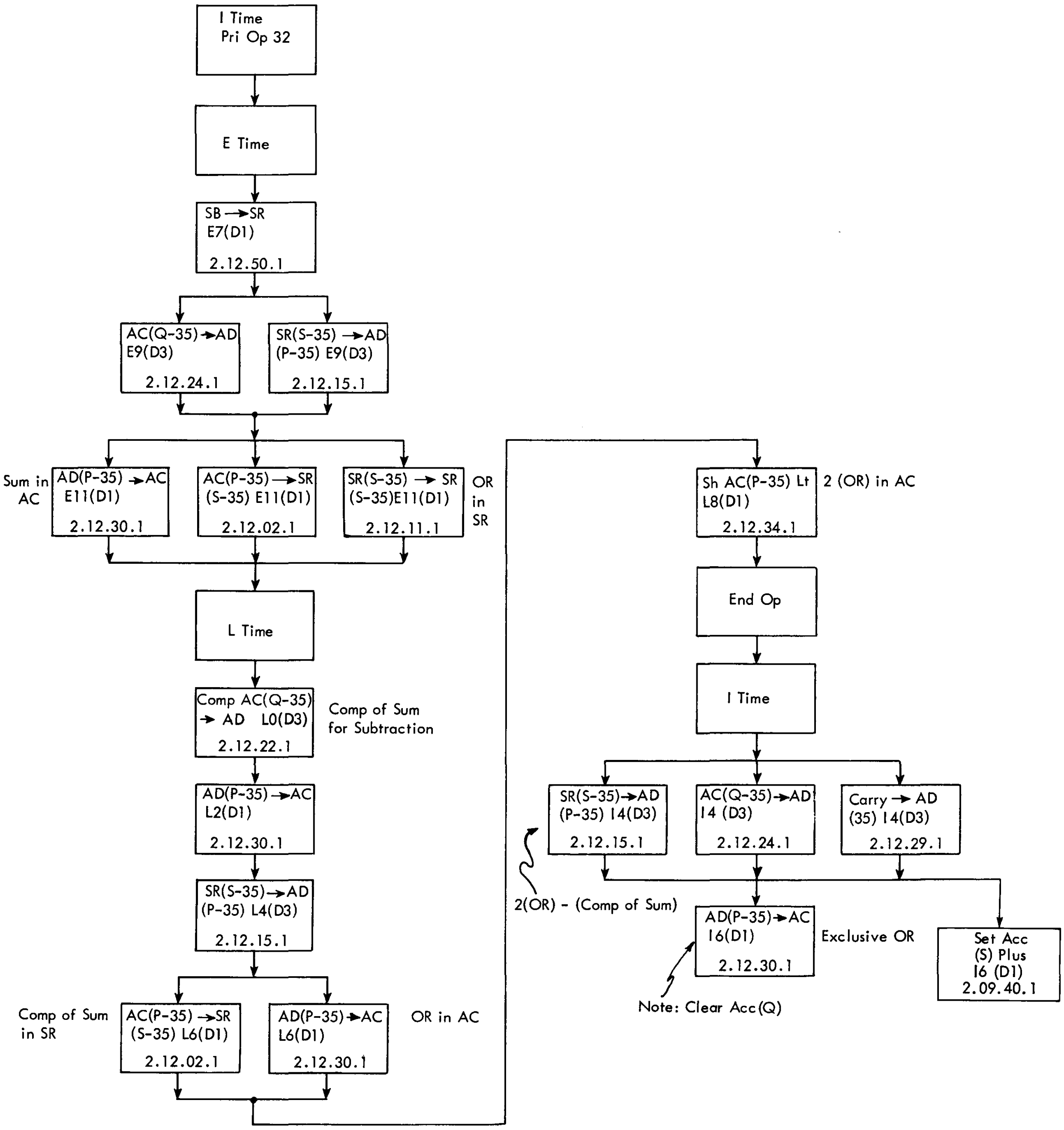


FIGURE 5.3-70. ERA +0322

Logically speaking, the EXCLUSIVE OR is the result of OR'ing bits from either one word or the other, but not both. The computer, which can only AND and OR, makes use of the following logical equation to develop the EXCLUSIVE OR:

EXCLUSIVE OR = 2 (A or B) - (A + B)

The derivation of the above equation can be shown with the following adder table:

Factor A	0	0	0	1	1
Factor B	0	0	1	0	1
A + B (carry not blocked)	0	1	0	0	0
Carry to be blocked	0	0	0	1	0
A + B (carry blocked)	0	0	1	1	0
EXCLUSIVE OR (A + B carry blocked)	0	0	1	1	0
A or B	0	0	1	1	1
2 (A + B)	1	0	0	0	0
2 (A or B)	0	1	1	1	0

From the table it can be seen that the EXCLUSIVE OR is equal to the sum output of the adders with all carries blocked. The carries cannot be blocked, but the EXCLUSIVE OR can be obtained by subtracting an amount equal to the blocked carries from the sum of two words:

1. EXCLUSIVE OR = (A + B) - blocked carries
   The blocked carries can be simulated by subtracting twice the OR from twice the sum of two words:
2. Blocked carries = 2 (A + B) - 2 (A or B) = 10000 - 01110 = 00010
   Substituting the equation 2 in equation 1:
3. EXCLUSIVE OR = (A + B) -[2 (A + B) - 2 (A or B)] = 01000 - 00010 = 00110
   And simplifying equation 3:
4. EXCLUSIVE OR = 2 (A or B) - (A + B) = 01110 - 01000 = 00110

5.3.12 Convert Instructions

The three convert instructions can materially reduce the time required for many "housekeeping" and table-look-up routines. They can be used for number conversions, for preparing print fields, and even for adding numbers in systems other than binary.

The convert instructions, like variable-length instructions, include a count field as well as the operation code, address and tag bit. The convert instructions cause a series of references to be made (usually six) and the address specifies the starting location of the first storage table. The register (accumulator or MQ) from which the reference is controlled is considered to be made up of six 6-bit groups. The first of these groups is added to the instruction address to give the location of the first storage reference. The word stored at this location must contain, in addition to its conversion information, the starting location of the next storage table. The convert-by-replacement instructions shift the controlling register six places, clearing the six places on the opposite end of the register. Positions(S-5)* of the table word are entered into the six cleared positions, and the next group of six bits is in position to add to the starting location of the next table. The process continues until up to six references have been made. The controlling register is gradually replaced by six-bit entries from the storage tables. When the number of references specified by the count has been made, the conversion is complete.

The tag of the instruction has a unique function for the convert instructions. Positions (18) and (19) are not used, but a bit in (20) will cause the storage table starting location contained in the last reference word to be stored in index register A.

Convert by Replacement from Accumulator CVR +0114 (Min I, L) Figure 5.3.71
(Max I, L, 6E)

This instruction treats the contents of the AC(P, 1-35) as six 6-bit representations. The instruction replaces a number of these representations, equal to the count of the instruction, with the contents of positions (S-5) of a like number of words from storage. These words are found by adding AC(30-35) (the first six-bit group) to SR(21-35) (initially the instruction address) and directing storage to this modified address. The word thus found is brought to the SR; the AC is shifted right six places; SR(S-5) replaces AC(P, 1-5). SR(21-35) adds to the next six-bit group in AC(30-35) to locate the next word in storage. The process is then repeated. After the required number of replacements, the address portion of the last storage word can be stored in index register A by including a "tag" bit in position (20) of the instruction.

The following illustrates the use and operation of CVR:

Direct Addition of BCD Numbers:

A	+	B	=	C
134589	+	691593	=	826182

Table required in storage for this example:

Storage Location	Contents	
	(S-5)	(21-35)
1000	0	1000
1001	1	1000
1002	2	1000
1003	3	1000
1004	4	1000
1005	5	1000
1006	6	1000
1007	7	1000
1008	8	1000
1009	9	1000

Storage Location	Contents	
	(S-5)	(21-35)
1010	0	1001
1011	1	1001
1012	2	1001
1013	3	1001
1014	4	1001
1015	5	1001
1016	6	1001
1017	7	1001
1018	8	1001
1019	9	1001

Instructions required for operation:

```
CAL  A            (First BCD word)
ADD  B            (Second BCD word)
CVR  6, 0, 1000   (Convert sum to BCD word)
```

* This text section uses (S-5) to represent (S, 1-5).

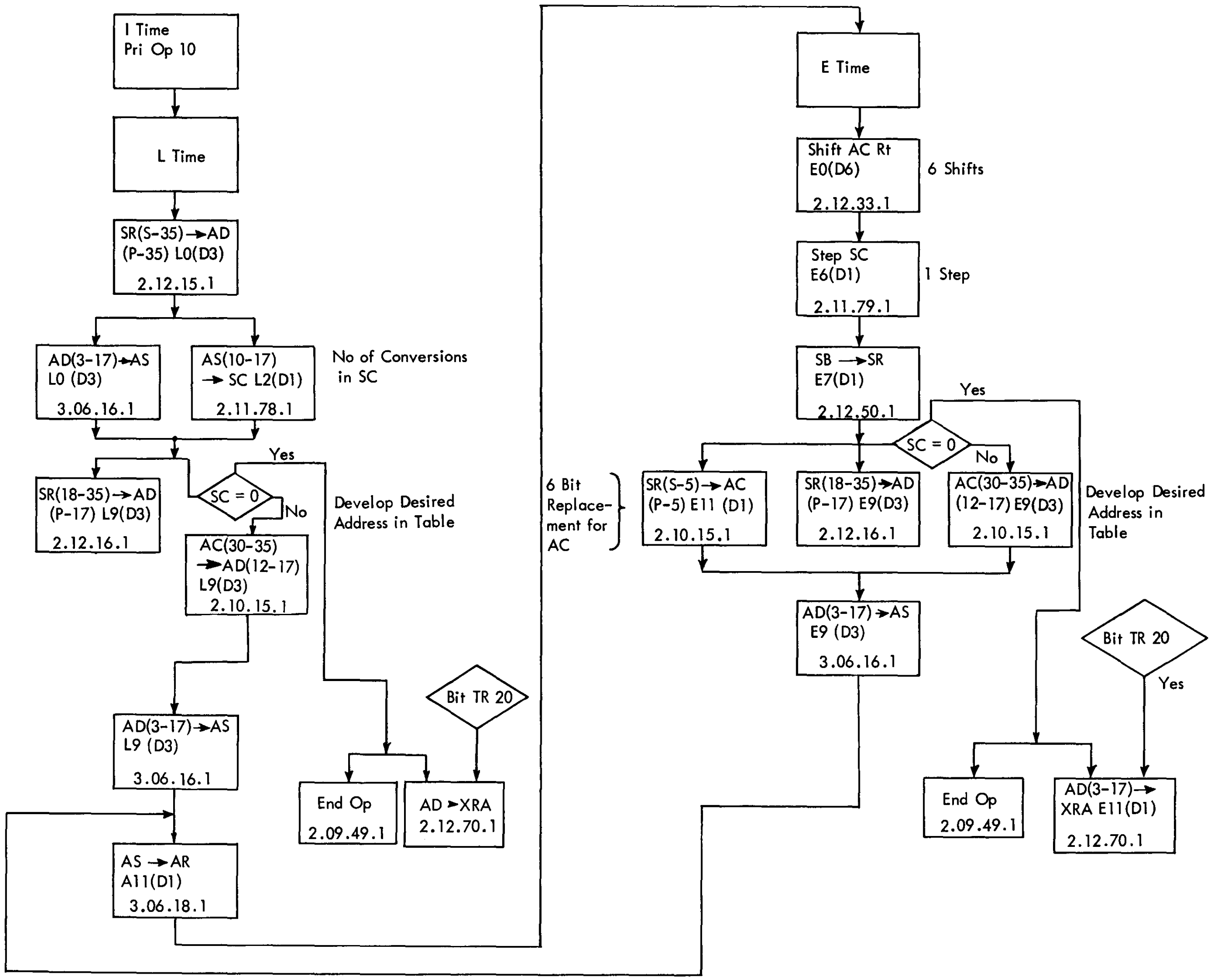
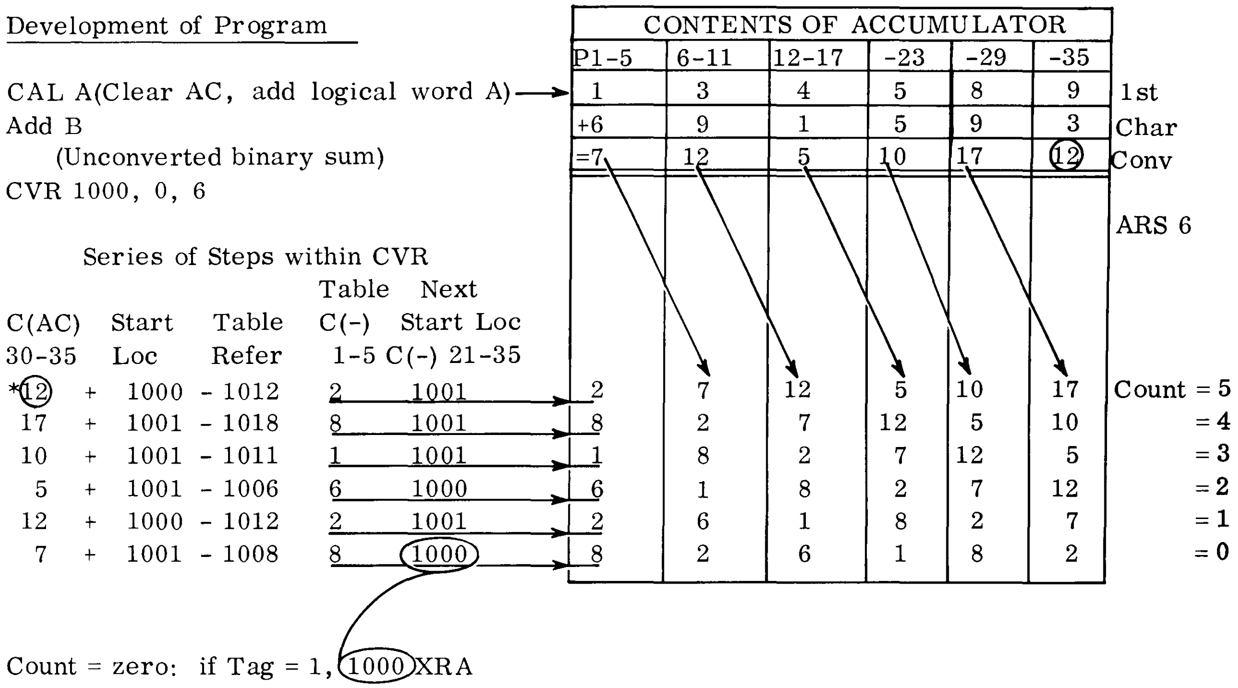


FIGURE 5.3-71. CVR +0114



Development of Program

CAL A(Clear AC, add logical word A)
Add B
(Unconverted binary sum)
CVR 1000, 0, 6

CONTENTS OF ACCUMULATOR						
P1-5	6-11	12-17	-23	-29	-35	
1	3	4	5	8	9	1st
+6	9	1	5	9	3	Char
=7	12	5	10	17	12	Conv
						ARS 6
2	7	12	5	10	17	Count = 5
8	2	7	12	5	10	=4
1	8	2	7	12	5	=3
6	1	8	2	7	12	=2
2	6	1	8	2	7	=1
8	2	6	1	8	2	=0

Series of Steps within CVR

C(AC) 30-35		Start Loc	Table Refer	Table C(-) 1-5	Next Start Loc C(-) 21-35
*12	+	1000	- 1012	2	1001
17	+	1001	- 1018	8	1001
10	+	1001	- 1011	1	1001
5	+	1001	- 1006	6	1000
12	+	1000	- 1012	2	1001
7	+	1001	- 1008	8	1000

Count = zero: if Tag = 1, 1000 XRA
SLW C (Store converted BCD sum as logical word)

The execution of CVR requires one L cycle to set the count in the shift counter and to calculate the initial table location. The instruction is completed in as many E cycles as specified by the count.

Convert by Replacement from MQ CRQ -0154 (Min I, L) Figure 5.3-72
(Max I, L, 6E)

This instruction operates on the MQ, considered to be composed of six 6-bit representations. The instruction replaces a number of these representations equal to the count of the instruction, with the contents of positions (S-5) of a like number of words from storage. The location of the first of these storage words is found by adding the contents of MQ(S-5) to SR(21-35) (intially the instruction address portion). The word stored at this modified location is brought to the SR, and the MQ is shifted left six places. Positions (S-5) of the stored word are placed in MQ(30-35), and the location of the next storage word is computed by again adding MQ(S-5) to SR(21-35) which is now the address portion of the previous storage word. The process continues until the required number of replacements have been made. At this time the presence of a tag bit in position 20 of the instruction will cause the address portion of the final storage word to be stored in index register A.

The following illustrates the use and operation of CRQ:

Prepare a BCD number for printing be replacing leading zeros with blanks:

Convert BCD number 007109 to BL BL 7109
Instructions required for this operation:

```
LDQ        A      (BCD number in storage)
CRQ 6,0, 2000     (Convert from MQ)
STQ        B      (Store converted number)
```

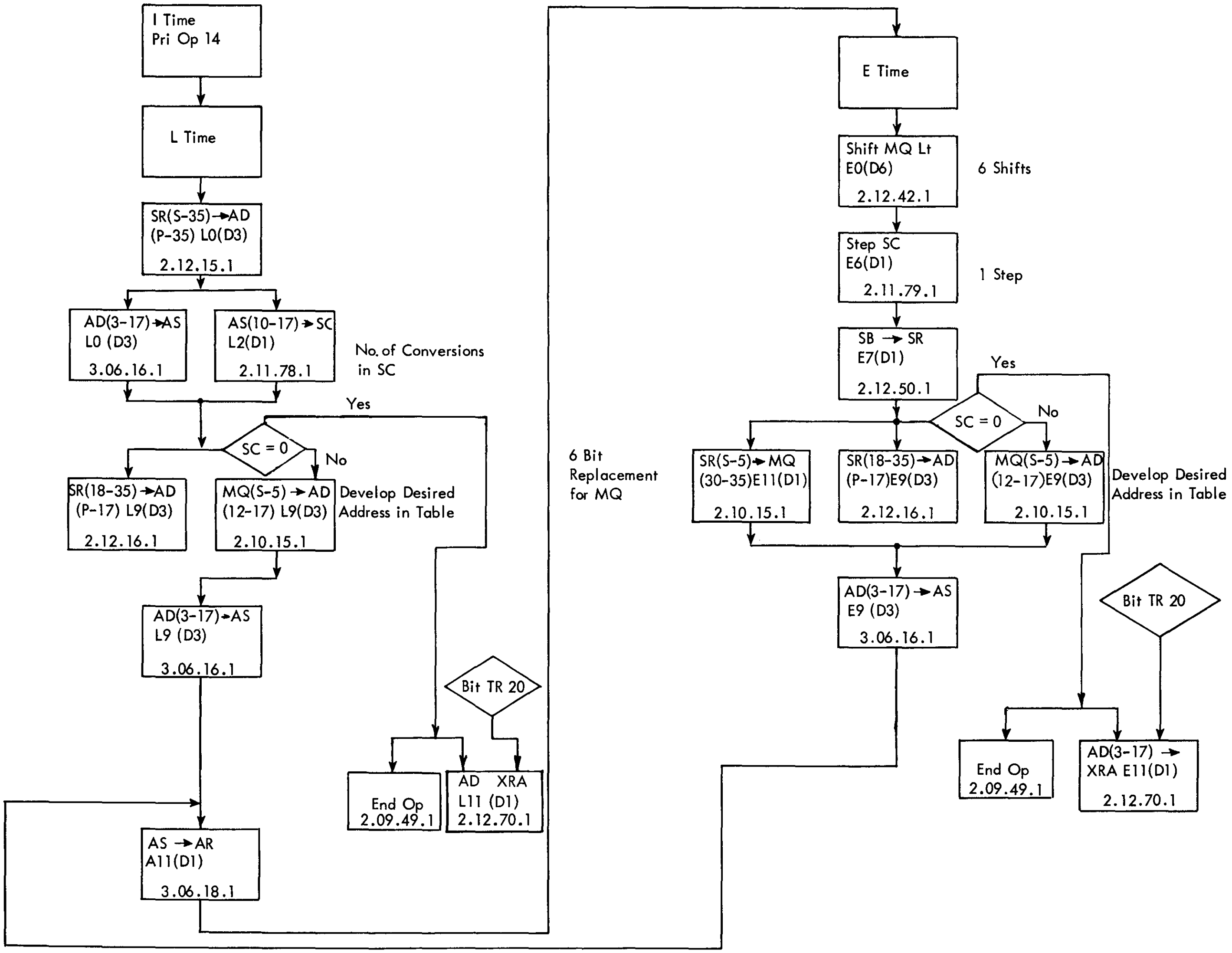


FIGURE 5.3-72. CRQ - 0154

Storage table required for CRQ (in decimal):

Storage Location	Contents	
	(S-5)	(21-35)
2000	BL	2000
2001	1	2010
2002	2	2010
2003	3	2010
2004	4	2010
2005	5	2010
2006	6	2010
2007	7	2010
2008	8	2010
2009	9	2010

Storage Location	Contents	
	(S-5)	(21-35)
2010	0	2010
2011	1	2010
2012	2	2010
2013	3	2010
2014	4	2010
2015	5	2010
2016	6	2010
2017	7	2010
2018	8	2010
2019	9	2010

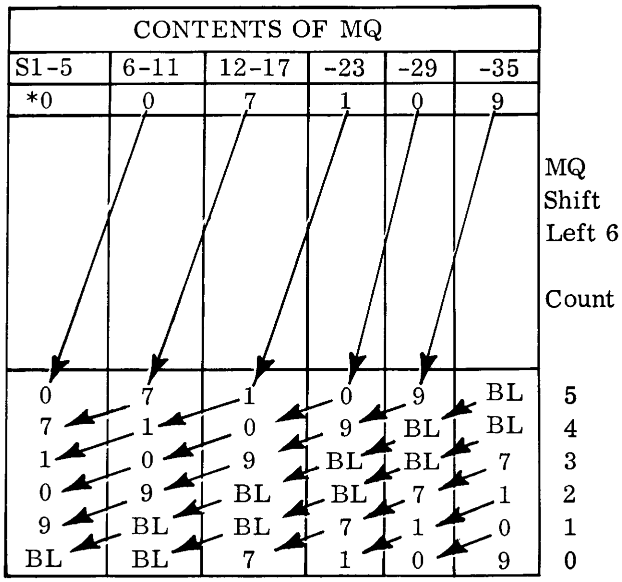
Development of Program

```
LDQ A                  1st
                       Char
CRQ 2000, 0, 6         Conv
```

Series of Steps within CRQ

C(MQ)s 1-5		Start Loc	Table Refer	Table C(-)s 1-5	Next Starting Loc C(-) 21-35
*0	+	2000 =	2000	BL	2000
0	+	2000 =	2000	BL	2000
7	+	2000 =	2007	7	2010
1	+	2010 =	2011	1	2010
0	+	2010 =	2010	0	2010
9	+	2010 =	2019	9	2010

Count = 0; if Tag = 1, 2010 → XRA



CONTENTS OF MQ						
S1-5	6-11	12-17	-23	-29	-35	
*0	0	7	1	0	9	
						MQ Shift Left 6
						Count
0	7	1	0	9	BL	5
7	1	0	9	BL	BL	4
1	0	9	BL	BL	7	3
0	9	BL	BL	7	1	2
9	BL	BL	7	1	0	1
BL	BL	7	1	0	9	0

The execution of CRQ is accomplished in one L cycle and a number of E cycles equal to the count.

Convert by Addition from MQ CAQ -0114(Min I, L) (Max I, L, 6E) Figure 5.3-73

This instruction treats the MQ as six 6-bit representations, as does CRQ. This instruction does not replace any of these representations, but uses them to locate a number of storage words equal to the count of the instruction. The storage words are located at the address developed by adding MQ(S-5) to SR(21-35) (initially the instruction address) and are brought to the SR. The storage words are then added to the contents of the accumulator. The MQ is not replaced, but is merely rotated left six places to allow the next six-bit representation to be added to the address portion of the previous storage word. The process continues until a number of additions (to the AC) equal to the count have been made. The operation is then complete. The address portion of the last storage word used can be stored in index registers for future reference by adding a tag bit in position 20 of the instruction.

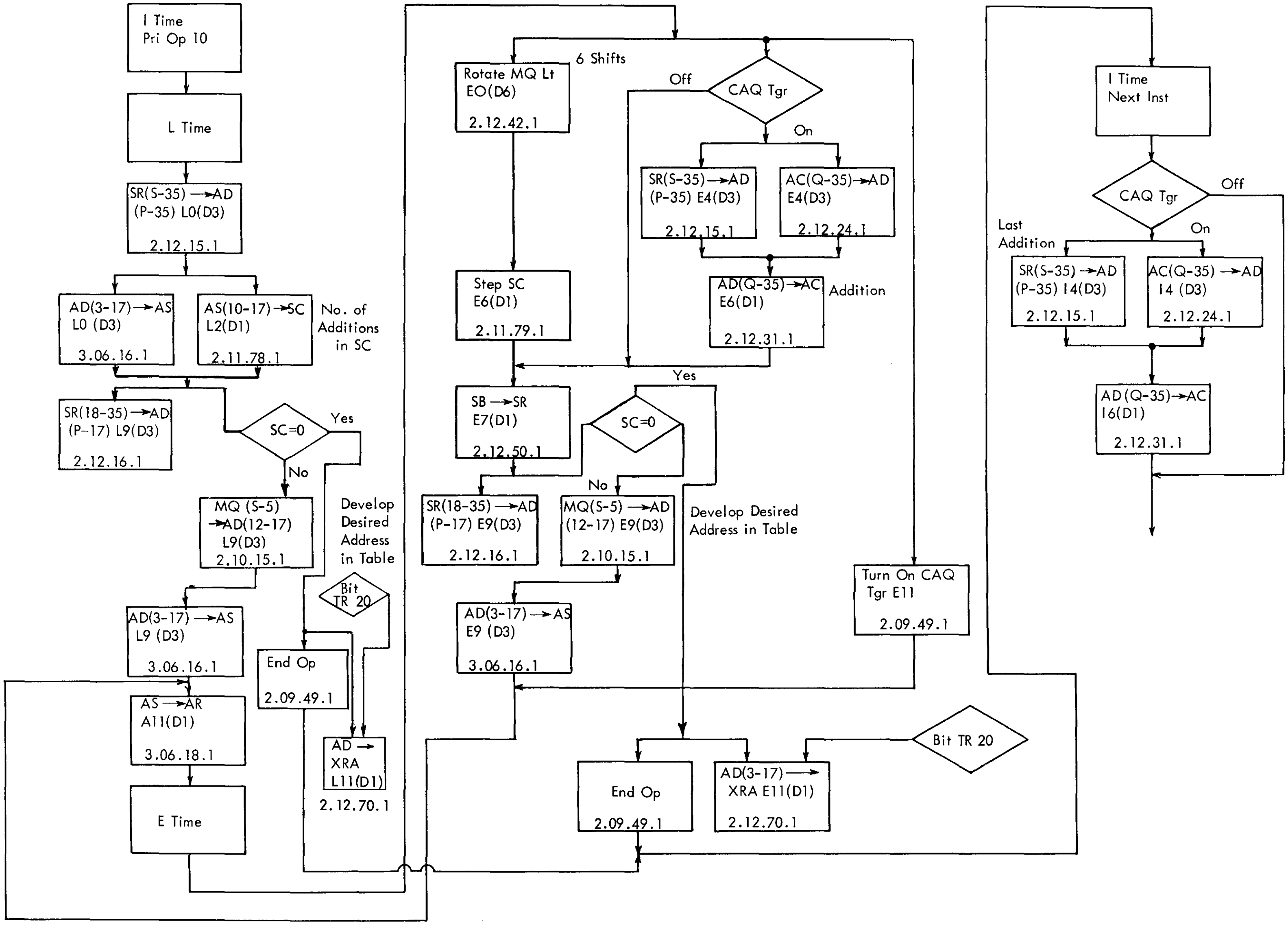


FIGURE 5.3-73. CAQ - 0114

To illustrate the operation and use of CAQ, a program which will convert BCD to binary follows.

Convert BCD Word to Binary

709542 (BCD) converted to 2,551,646 (Octal)

Instructions required for this operation:

LDQ	A	(BCD word)
CLM		(Clear AC)
CAQ 6, 0, 3000		(Convert to binary)
ARS 16_{10}		(Position result in AC)
SLW	B	(Store result)

Storage table required for CAQ:

Storage Location	Contents	
	(S-19)	(21-35)
Decimal	Octal	Decimal
3000	0	3100
3001	303,240	3100
3002	606,500	3100
3003	1,111,740	3100
3004	1,415,200	3100
3005	1,720,440	3100
3006	2,223,700	3100
3007	2,527,140	3100
3008	3,032,400	3100
3009	3,335,640	3100
3100	0	3200
3101	23,420	3200
3102	47,040	3200
3103	72,460	3200
3104	116,100	3200
3105	141,520	3200
3106	165,140	3200
3107	210,560	3200
3108	234,200	3200
3109	257,620	3200
3200	0	3300
3201	1,750	3300
3202	3,720	3300
3203	5,670	3300
3204	7,640	3300
3205	11,610	3300
3206	13,560	3300
3207	15,530	3300
3208	17,500	3300
3209	21,450	3300

Storage Location	Contents	
	(S-19)	(21-35)
Decimal	Octal	Decimal
3300	0	3400
3301	144	3400
3302	310	3400
3303	454	3400
3304	620	3400
3305	764	3400
3306	1,130	3400
3307	1,274	3400
3308	1,440	3400
3309	1,604	3400
3400	0	3500
3401	12	3500
3402	24	3500
3403	36	3500
3404	50	3500
3405	62	3500
3406	74	3500
3407	106	3500
3408	120	3500
3409	132	3500
3500	0	
3501	1	
3502	2	
3503	3	
3504	4	
3505	5	
3506	6	
3507	7	
3508	10	
3509	11	

Development of Program

Binary Equivalent

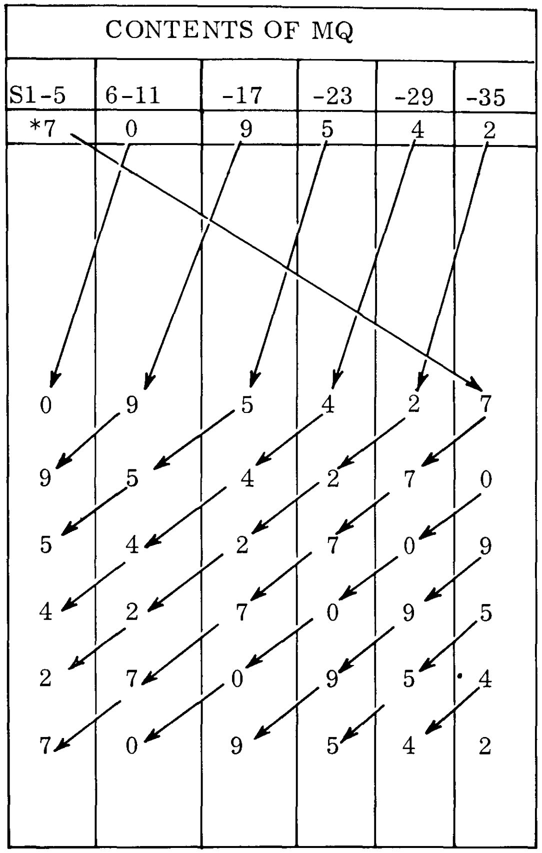
BCD Word

```
LDQ BCD WORD
CLM
CAQ 3000, 0, 6
```

Series of Steps within CAQ

C(MQ) S, 1-5		Start Loc	Table Refer	Table C(-)S, 1-19	Next Start Loc C(-) 21-35)		CONTENTS OF ACCUMULATOR P, 1-19	21-35	Count
							XX	XX	
							00	00	
								RQL 6	
7	+	3000 =	3007	2,527,140	3100	+	2,527,140	3100	5
0	+	3100 =	3100	0	3200	+	0	3200	4
						=	2,527,140	6300	
9	+	3200 =	3209	21,450	3300	+	21,450	3300	3
						=	2,550,610	9600	
5	+	3300 =	3305	764	3400	+	764	3400	2
						=	2,551,574	13000	
4	+	3400 =	3404	50	3500	+	50	3500	1
						=	2,551,644	16500	
2	+	3500 =	3502	2	-	+	2	-	0
						=	2,551,646	16500	

ARS 16 (Binary Equivalent) → 2,551,646



CONTENTS OF MQ

S1-5	6-11	-17	-23	-29	-35
*7	0	9	5	4	2
0	9	5	4	2	7
9	5	4	2	7	0
5	4	2	7	0	9
4	2	7	0	9	5
2	7	0	9	5	4
7	0	9	5	4	2

SLW_

5.3.13 Floating Point Trap (Figure 5.3-74)

To simplify the programming associated with floating-point overflow or underflow, a unique system of spill indication is used. When an underflow or overflow occurs in either the accumulator or MQ during a floating-point operation, an instruction trap is taken to location 0010_8, after ending operation on the executer instruction. In addition, the location of the floating-point instruction plus one is stored in the address portion of location 0000, and a spill identification code is stored in the decrement portion of location 0000.

The 7090 floating-point trap mode trigger (Systems 2.10.71.1) is reset on, putting the CPU in a floating-point trap mode status normally. To leave this condition, a LFTM (leave floating-trap mode) instruction must be executed. The mode of operation may then be re-entered by resetting the CPU or executing an EFTM (enter floating trap mode) instruction.

The spill identification code is produced as follows:

1. Position 14 is set to one if overflow or underflow occurs during a floating-point divide operation.
2. Position 15 is set to one if overflow occurs in either the accumulator or MQ.
3. Position 16 is set to one if overflow or underflow occurs in the accumulator.
4. Position 17 is set to one if overflow or underflow occurs in the MQ.

This results in the spill identification codes for the various conditions as follows:

Floating-Point			Positions				Octal
Operation	Accumulator	MQ	14	15	16	17	Code
Add, Subtract		Underflow	0	0	0	1	01
Multiply,	Underflow	Underflow	0	0	1	1	03
Round	Overflow		0	1	1	0	06
	Overflow	Overflow	0	1	1	1	07
Divide		Underflow	1	0	0	1	11
	Underflow		1	0	1	0	12
	Underflow	Underflow	1	0	1	1	13
		Overflow	1	1	0	1	15

The code stored in the decrement may be loaded into an index register and an indexed transfer instruction may be used to lead to the proper corrective routine.

Underflow and overflow of the floating-point characteristic are detected by examining positions P and Q of the adders and the accumulator. Overflow is identified by a bit in (P) only. Underflow is identified by bits in (P) and (Q). MQ overflow or underflow must be recognized as the MQ characteristic is being computed in the adders. ACC overflow to underflow may be recognized at any time after the final characteristic has been assigned.

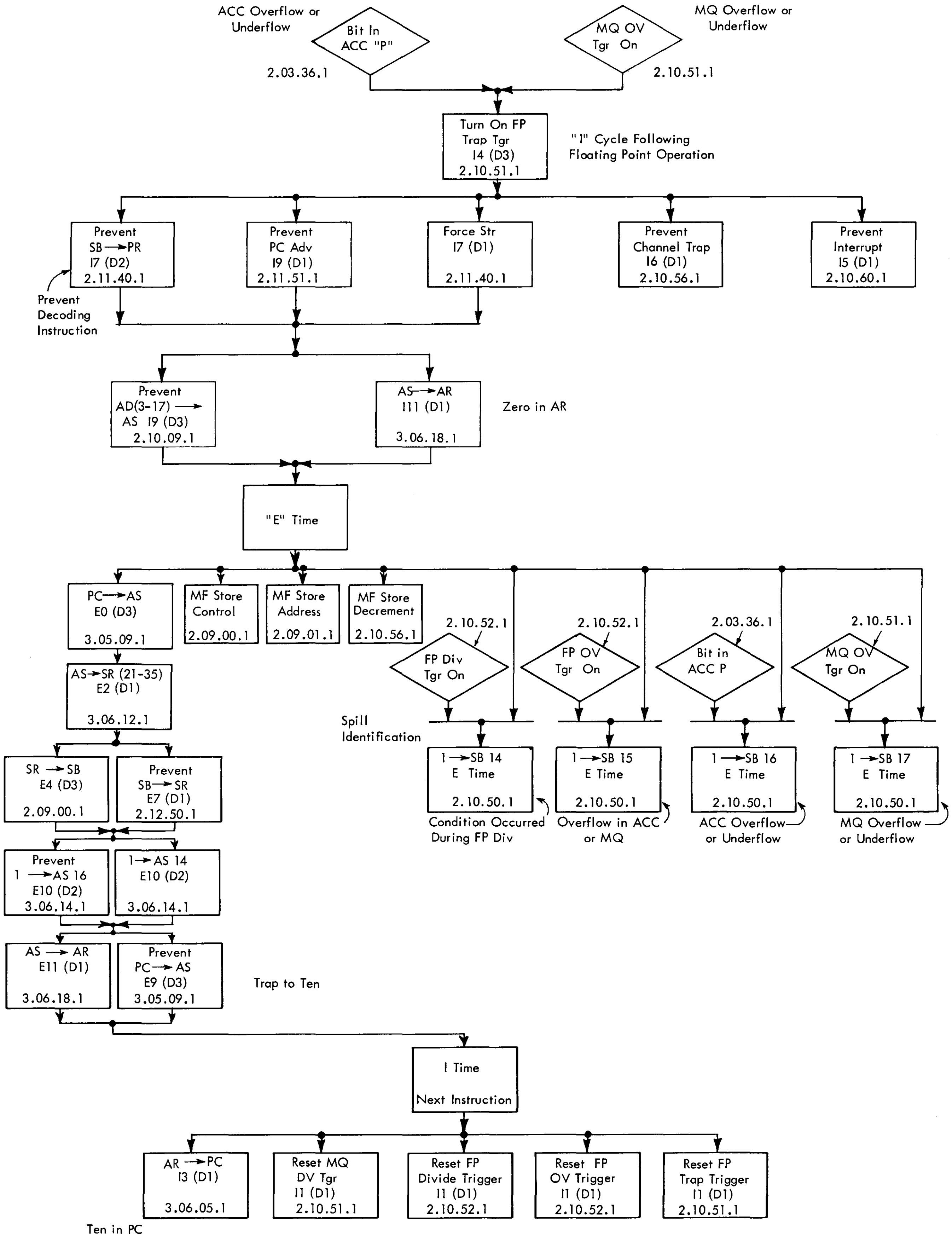

ACC Overflow or Underflow
Bit In ACC "P"
2.03.36.1
MQ Overflow or Underflow
MQ OV Tgr On
2.10.51.1
Turn On FP Trap Tgr 14 (D3) 2.10.51.1
"I" Cycle Following Floating Point Operation
Prevent SB→PR I7 (D2) 2.11.40.1
Prevent Decoding Instruction
Prevent PC Adv I9 (D1) 2.11.51.1
Force Str I7 (D1) 2.11.40.1
Prevent Channel Trap I6 (D1) 2.10.56.1
Prevent Interrupt I5 (D1) 2.10.60.1
Prevent AD(3-17) → AS I9 (D3) 2.10.09.1
AS→AR I11 (D1) 3.06.18.1
Zero in AR
"E" Time
PC→AS E0 (D3) 3.05.09.1
MF Store Control 2.09.00.1
MF Store Address 2.09.01.1
MF Store Decrement 2.10.56.1
AS→SR (21-35) E2 (D1) 3.06.12.1
SR → SB E4 (D3) 2.09.00.1
Prevent SB→SR E7 (D1) 2.12.50.1
Prevent 1 →AS 16 E10 (D2) 3.06.14.1
1→AS 14 E10 (D2) 3.06.14.1
AS → AR E11 (D1) 3.06.18.1
Prevent PC→AS E9 (D3) 3.05.09.1
Trap to Ten
2.10.52.1
FP Div Tgr On
2.10.52.1
FP OV Tgr On
2.03.36.1
Bit in ACC P
2.10.51.1
MQ OV Tgr On
Spill Identification
1 →SB 14 E Time 2.10.50.1
Condition Occurred During FP Div
1 →SB 15 E Time 2.10.50.1
Overflow in ACC or MQ
1 →SB 16 E Time 2.10.50.1
ACC Overflow or Underflow
1 →SB 17 E Time 2.10.50.1
MQ Overflow or Underflow
I Time Next Instruction
AR →PC I3 (D1) 3.06.05.1
Ten in PC
Reset MQ DV Tgr I1 (D1) 2.10.51.1
Reset FP Divide Trigger I1 (D1) 2.10.52.1
Reset FP OV Trigger I1 (D1) 2.10.52.1
Reset FP Trap Trigger I1 (D1) 2.10.51.1

FIGURE 5.3-74. FLOATING POINT TRAP MODE

The recognition circuits are shown on Systems 2.10.50.1. To set position 14 of the decrement, the floating-point divide occurred trigger is used. To set position 15, the floating-point overflow trigger is turned on by a bit in (P) only. To set position 16, a bit in ACC (P) is all that is required. To set position 17, the MQ overflow trigger is turned on by an adder (P) output while the MQ characteristic is being computed.

The controls for floating-point trap are shown on Systems 2.10.51.1. An overflow or underflow causes the floating-point trap trigger to be turned on. This trigger initiates a pseudo STR instruction by blocking the instruction following the floating-point instruction during which the condition occurred and setting IR sign (-) and a one in IR (9). Address 0000 is set into the address register, store decrement and store address controls are activated, and an E cycle is initiated. The program counter and the four spill code lines are routed to the storage bus and stored in location 0000. The address register is then set to 0010_8 to locate the next instruction to be executed.

6.0.00 IBM 7151 CONSOLE CONTROL UNIT

The 7151 Console Control Unit is a separate unit that may be placed at any convenient location within certain cable length restrictions. It provides manual and semiautomatic control over the system. The console consists of three panels: an operator's panel, a customer engineer's test panel, and a marginal check panel. With proper use, the console can be one of the customer engineer's most powerful tools. The time spent learning how to use it effectively is returned many times when diagnosing system errors.

This section introduces the keys, lamps, switches, and test facilities, their function, and any associated logic.

During the progress of a program, the operator may want some amount of control. For example, at a given point in a calculation, the computer is given the instruction to halt. The operator then can make a visual check of the information developed thus far in the program. At this point, one of several alternate manual steps may be performed, depending on the data observed. For this operation, the automatic-manual switch is set to MANUAL. With the machine in this state, the operator may enter and execute an instruction, interrogate any locations in storage for a visual check of the information stored, or load data from the operator's panel keys. After the desired manipulations have been made, the machine is returned to automatic status; the start key is depressed and the program continues.

The start and stop of the machine are under control of the master stop trigger. This trigger in turn controls "B cycle interrupt," which gates the I, E, and L cycles.

The keys, switches, and lamps on the console provide a means to:

1. Start or stop the machine
2. Step through a program at reduced speed
3. Check the status of the CPU
4. Display or revise the contents of storage
5. Alter the program

In addition, the customer engineer has facilities for several testing features which include:

1. Auxiliary start and reset key
2. I-O interlocks
3. Continuous execution of an instruction
4. Power jacks for test equipment
5. B cycle control

The console (Figure 6.0-1) is divided into three sections; an operator's panel, a customer engineer's test panel, and a bias panel. Figure 6.0-2 designates system page locations for the keys and indicators located on the console.

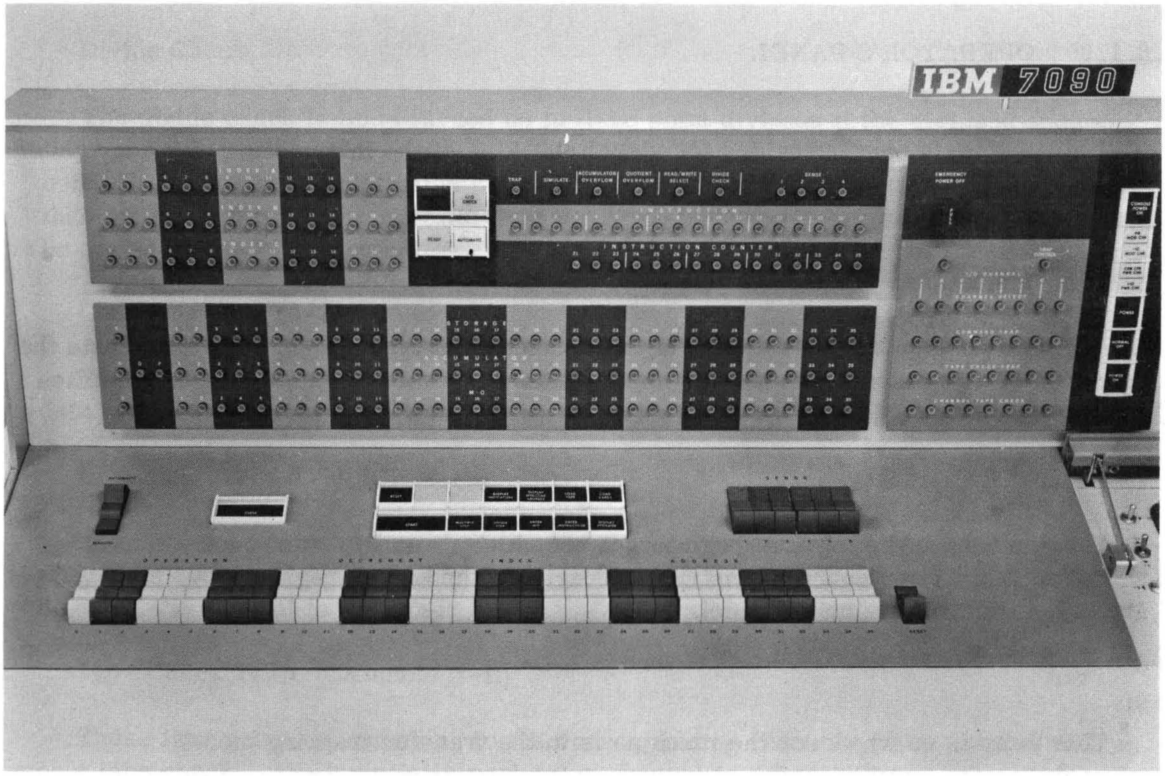


FIGURE 6.0-1. IBM 7151 CONSOLE

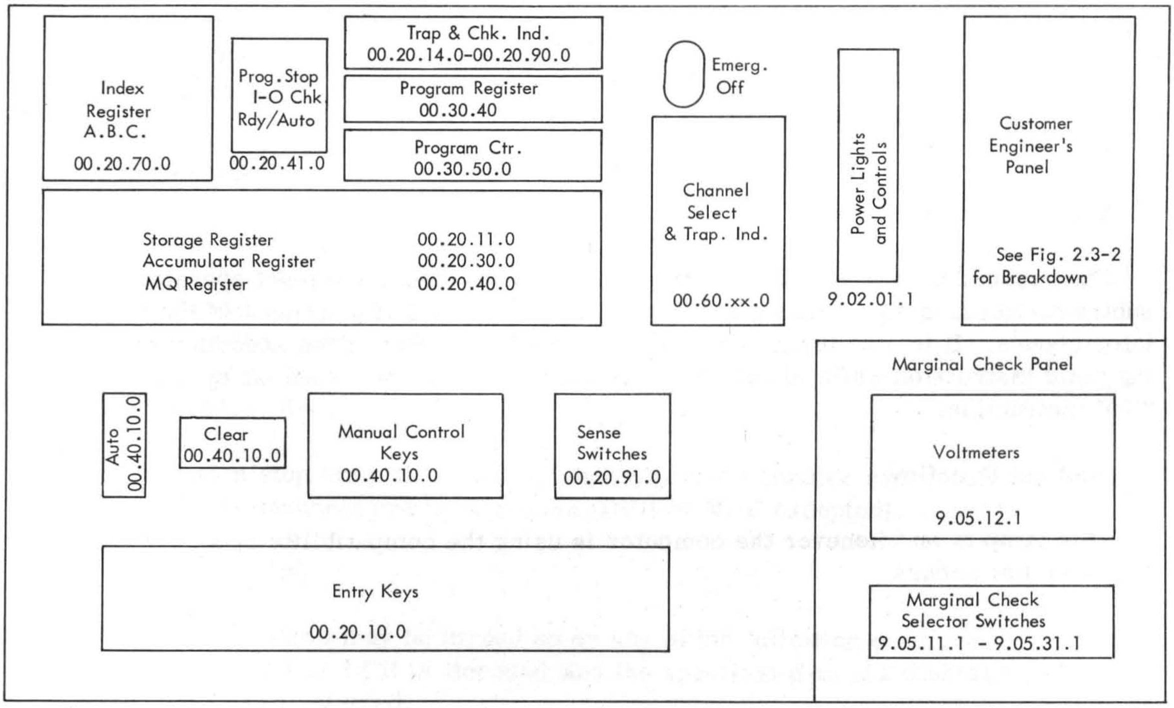


FIGURE 6.0-2. BLOCK DIAGRAM OF CONSOLE

6.1.00 OPERATOR'S PANEL

The operator's panel provides for visual checking of the information in the computer and for manual control of the computer's functions. It is also a station from which power may be applied to or removed from the system.

6.1.01 Indicators

All indicators on the console are of the incandescent type. When used to indicate the condition of a register, a lamp being on signifies a 1, while a lamp being off signifies a 0.

Internal Registers

The contents of the internal registers (accumulator, multiplexor-quotient, storage register, instruction counter, instruction register, and index registers A, B, and C) are displayed directly on the panel.

Trap

This lamp is on whenever the machine is in the transfer trapping mode.

Simulate

The simulate lamp is on when the 7090 is operating in any of the following modes associated with the 704, 709, or 7090 compatibility program:

1. I-O select and sense trap mode
2. Copy and locate drum address trap mode
3. Storage nullification mode

Accumulator Overflow

The accumulator overflow lamp is on any time during a fixed point operation (add, subtract, and so on) or shifting operation that a carry out of position 1 of the accumulator occurs. It is also turned on by a bit in position P during the execution of a floating point instruction while in compatibility mode. It may be turned off by the TNO or TOV instruction.

Quotient Overflow

This lamp is on whenever the computer is using the compatibility program and an MQ overflow occurs.

Read-Write Select

The read-write (R-W) select lamp is on whenever the channel-in-use trigger is on in any data channel.

Divide Check

The divide check lamp is turned on in fixed point division if the dividend accumulator (AC) is greater than or equal to the divisor (storage register). In floating-point divide the lamp is on if the magnitude of the fraction of the dividend is greater than or equal to twice the magnitude of the fraction of the divisor. The indicator may be tested by the DCT instruction.

Channel Select (A-H)

The channel select (A-H) lamps, one for each channel, are on according to the data channel that is in operation. They are off when the channel is not in operation.

Command Word Trap (Channels A-H)

These lamps are turned on according to the corresponding channel that is enabled to trap on command word or end of file. The lamp for a particular channel is turned off when the channel is disabled.

Tape Check Trap (Channels A-H)

These lamps are on according to the corresponding channel that is enabled to trap on a tape check. The lamp for a particular channel is off when the channel is disabled.

Channel Tape Check

These lamps, one for each channel, are on if any error is detected while writing or reading. The lights may be turned off by the execution of a Transfer on Redundancy Check instruction.

Trap Control

This lamp is on when the channel is not executing a channel trap. It is off when any channel enters a trap condition. While the light is off, no channel traps may be executed. Channel traps may be executed only when the light is on at the same time as any of the enabled lamps.

Program Stop (Red)

The program stop lamp is turned on whenever the computer executes a halt instruction and no data channels are in operation (DVH or VDH excepted).

I-O Check (White)

The I-O check lamp may be turned on by any of the following conditions:

1. If a RCH or LCH is decoded and the specified data channel has not been selected.
2. If, when writing, a data channel data register has not been loaded with a word from storage by the time its contents are to be sent to the output unit.

3. If, when reading, a data channel data register has not transmitted its contents to storage by the time new information is to be loaded into it from an output unit.

The I-O check lamp may be turned off by the execution of an IOT.

Read Light (White)

The ready lamp comes on after power comes up and remains on except when the machine is in automatic status, the continuous enter instruction switch is on, the I-O interlock switch is in manual, or if any B-time control switch is on.

Automatic (Yellow)

The automatic lamp is on whenever the computer is executing instructions in the automatic mode or whenever a data channel is in operation.

Console Power-On (Red)

The console power-on lamp is on whenever power is applied to the console.

Central Components Power Check (Yellow)

The central components power check comes on whenever a fuse or circuit breaker opens in CPU frame 1 or 2, multiplexor, or core storage. It also lights when core storage has improper oil temperature or low oil pressure.

I-O Power Check (Yellow)

The I-O power check lamp is turned on whenever a fuse, an open thermal, or airflow failure is sensed in a data channel.

Power (Red)

The power indicator is turned on whenever DC power is up in core storage.

Marginal Check +6 (Yellow)

This indicator is on whenever the +6 supply marginal check variable autotransformer is not in the home position.

Marginal Check -12 (Yellow)

This indicator is on whenever the -12 supply marginal check variable autotransformer is not in the home position.

6.1.02 Manual Controls

Figure 6.1-1 shows the keys and switches on the operator's panel that provide for starting and stopping the machine and initiating computer functions. All keys are of the spring-returned variety with the exception of the auto-manual key, entry keys, sense switches, and emergency-off key.

Power-On

With the system in normal-off status, pressing the power-on key will commence the power-on sequence. Ready status (power-on) will be reached in about 20 seconds. As power comes on, a clear operation is performed, resetting all registers and triggers, and setting memory to all zeros.

Normal-Off

The normal-off key will initiate the following:

1. Immediate removal of 60-cycle power from the MG set, MG blower, and all frame blowers except memory.
2. Immediate removal of 400-cycle power from the 30-60 volt memory power supply.
3. After 5 seconds, removal of 400-cycle power from the standard memory supply.
4. After 3 minutes, removal of power from the memory blowers.

The dropping of the 60-cycle power removes the input to the MG, but the MG still rotates at about full speed for longer than 5 seconds, allowing the memory power to sequence down.

Emergency-Off

When the emergency-off switch is pulled, all power is immediately removed from the 7090 system, except the voltage to HR 24 and HR 30 points in the power control unit. The emergency-off switch is to be used only in emergencies, because of possible damage to circuits.

Resets

The reset circuits control the resetting of various components in the system. There are three types of resets which may be initiated from the panel.

> An interlock reset is initiated by the load keys, clear key, and power-on key. It causes the resetting of all registers (except sense indicator), panel light (except power on and ready), all channels that are in operation and their associated registers.
>
> An operator's reset is initiated by the reset key. It performs all the functions of an interlock reset plus a bias reset for CPU triggers.
>
> A power-on reset occurs when power is applied to the system. This reset accomplishes an interlock reset, an operator's reset, plus the resetting of the clock and setting core storage to all zeros. The clear key also initiates a power-on reset if the machine is in automatic status.

Automatic-Manual Key

The auto-manual switch controls the rest of the switches on the console. If a program is running in automatic status, and the switch is put in the manual position, the program will stop after it completes the operation it is performing. If an I-O program is running, I-O will complete the operation before stopping. The auto-manual switch will not affect the program stop status.

Auto.

Man

CLEAR

RESET	SPARE	SPARE	Disp Ind	Disp Eff Adr	Load Tape	Load Cards
START		Mult Step	Sing Step	Enter MQ	Enter Inst	Disp Stg

S ENTRY KEYS 35

Key Reset

FIGURE 6.1-1. OPERATOR'S PANEL

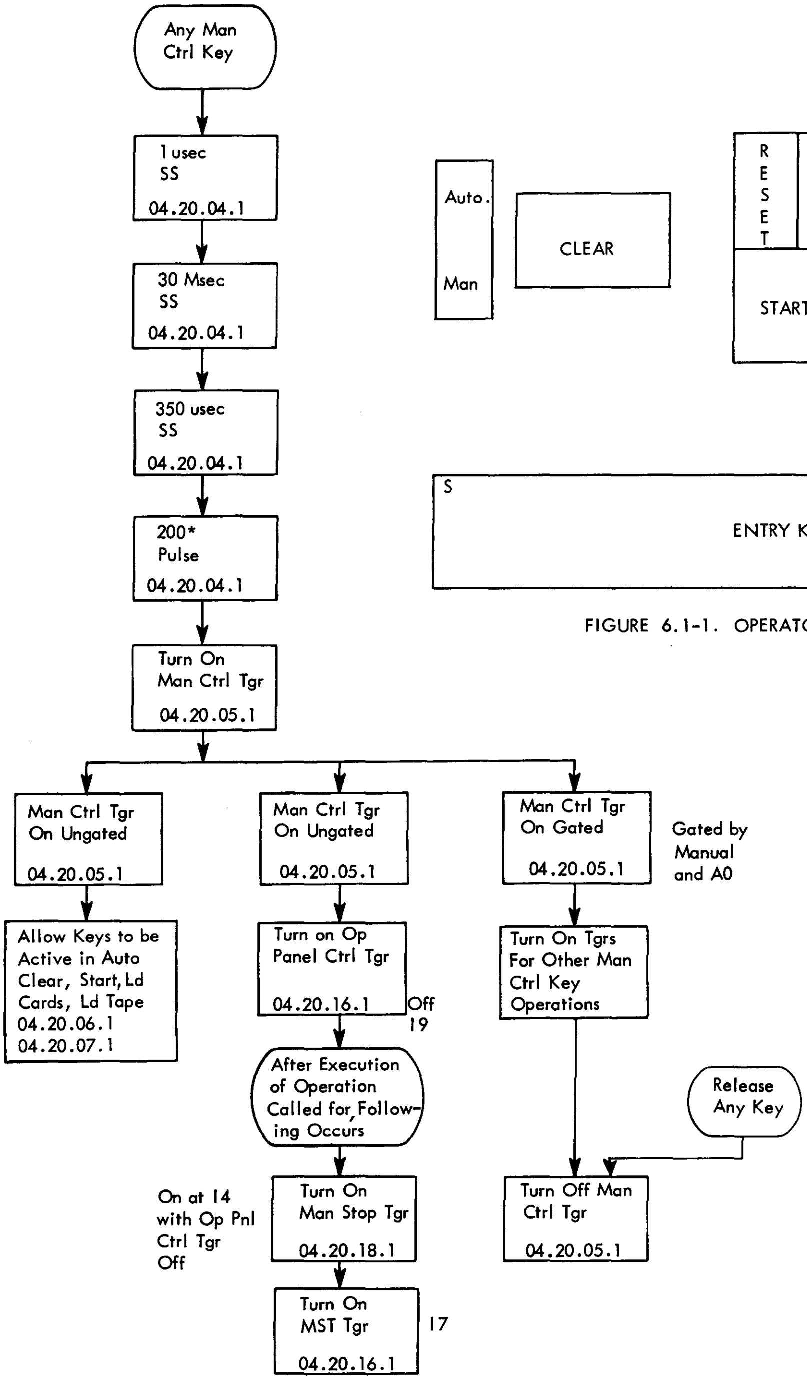

Any Man Ctrl Key
1 usec SS 04.20.04.1
30 Msec SS 04.20.04.1
350 usec SS 04.20.04.1
200* Pulse 04.20.04.1
Turn On Man Ctrl Tgr 04.20.05.1
Man Ctrl Tgr On Ungated 04.20.05.1
Allow Keys to be Active in Auto Clear, Start, Ld Cards, Ld Tape 04.20.06.1 04.20.07.1
Man Ctrl Tgr On Ungated 04.20.05.1
Turn on Op Panel Ctrl Tgr 04.20.16.1
Off 19
After Execution of Operation Called for, Following Occurs
On at 14 with Op Pnl Ctrl Tgr Off
Turn On Man Stop Tgr 04.20.18.1
Turn On MST Tgr 04.20.16.1
17
Man Ctrl Tgr On Gated 04.20.05.1
Gated by Manual and A0
Turn On Tgrs For Other Man Ctrl Key Operations
Release Any Key
Turn Off Man Ctrl Tgr 04.20.05.1

FIGURE 6.1-2. MANUAL CONTROL KEY

Entry Keys

There are 36 entry keys on the operator's panel: S, 1-35. Depressing a key sets a 1 in that position; leaving a key normal sets a 0 in that position. Information set in the entry keys may be entered into storage, executed, or used for a storage inquiry address. The entry keys may all be reset to 0 by depressing the reset key to the right of position 35.

6.1.03 Manual Control Keys

When any manual control key is pressed, a series of single-shots and triggers are set. Separating the key from the usable signal prevents false indications from noise generated by the key (Figure 6.1-2). Three single-shots are fired in sequence: a 1-usec, a 20-ms and a 350-usec. The 350-usec single-shot is taken through a delay network to develop a 200-ns pulse that turns on the manual control trigger. By this time, the switch has settled down and the trigger to perform the desired operation turns on, resetting the manual control trigger.

Start Key (Figure 6.1-3)

If the CPU is in automatic and ready status, pressing the start key initiates machine operation by turning off the master stop trigger and the program stop trigger. The master stop trigger off conditions "not B cycle interrupt," allowing the computer to proceed. If the system is in manual status, pressing the start key turns off the program stop trigger.

Clear Key (Figure 6.1-4)

The clear key is only operative if the computer is in automatic status. Pressing the clear key:

1. Fires a 1-usec single-shot to reset the clock and all channel registers of channels that are not in manual status.
2. Resets CPU interlocks and registers.
3. Conditions circuits which allow zeros to be written into all storage locations.

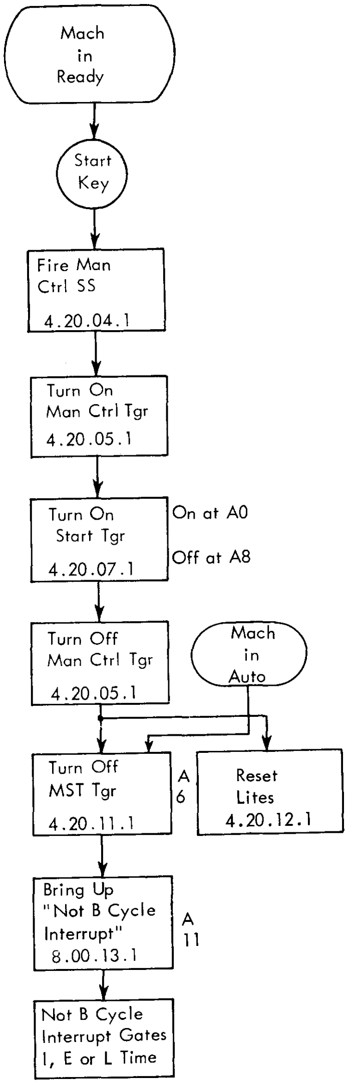
The program counter controls the stepping through memory, with position 2 of the PC indicating when all addresses have been zeroed. The turn-on of PC2 trigger will turn on the master stop trigger at E10.

Display Storage (Figure 6.1-5)

Pressing the display storage key causes the address portion of the operator's panel keys to be sensed, to determine the address in storage to be displayed. All 36-bit positions of the address will be displayed in the storage register. This is accomplished by: (1) turning off the master stop trigger, (2) bringing the operator's panel keys to the storage register in I time, and (3) suppressing "storage bus to storage register." The address portion of the SR is routed through the adders and address switches, to the address register. During E time, the SB is gated to the SR, which contains the 36 bits of the desired address. If a tag or indirect addressing is specified in the operator's keys, the contents of the effective address or the I-A address will be displayed.

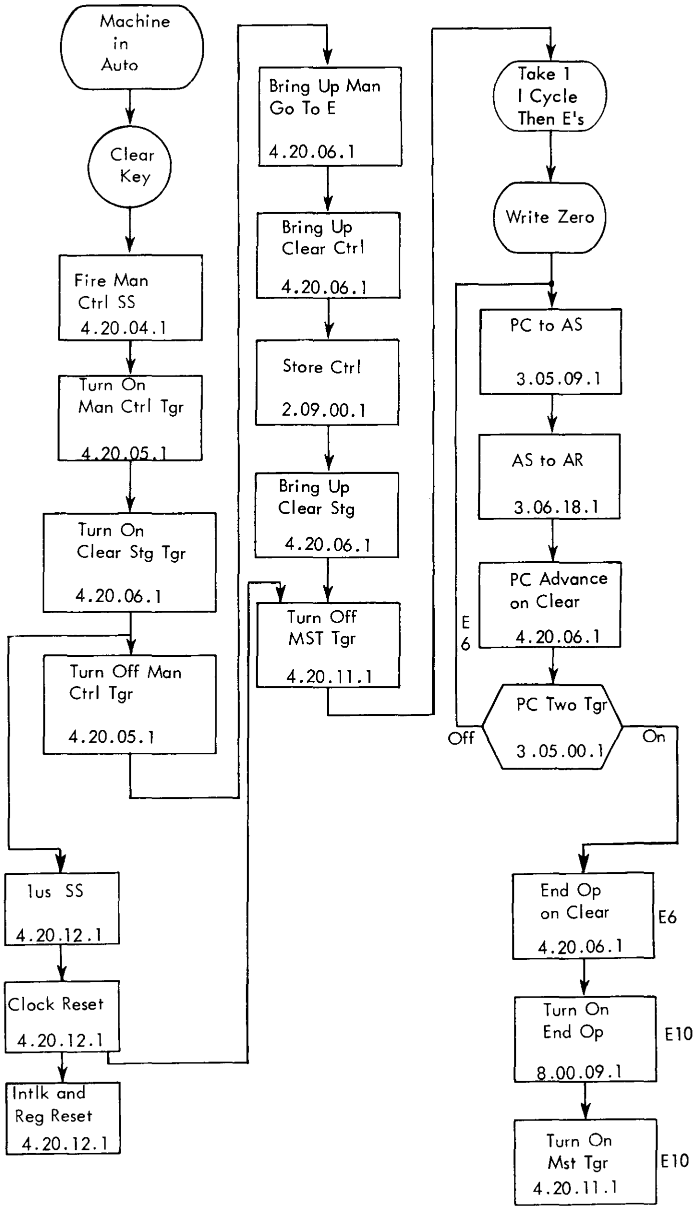
Display Indicators

The display indicators key gates the true value of the sense indicators to the storage for display (Figure 6.1-6).



Mach in Ready
Start Key
Fire Man Ctrl SS 4.20.04.1
Turn On Man Ctrl Tgr 4.20.05.1
Turn On Start Tgr 4.20.07.1
On at A0
Off at A8
Turn Off Man Ctrl Tgr 4.20.05.1
Mach in Auto
Turn Off MST Tgr 4.20.11.1
A 6
Reset Lites 4.20.12.1
Bring Up "Not B Cycle Interrupt" 8.00.13.1
A 11
Not B Cycle Interrupt Gates I, E or L Time

FIGURE 6.1-3. START



Machine in Auto
Clear Key
Fire Man Ctrl SS 4.20.04.1
Turn On Man Ctrl Tgr 4.20.05.1
Turn On Clear Stg Tgr 4.20.06.1
Turn Off Man Ctrl Tgr 4.20.05.1
Ius SS 4.20.12.1
Clock Reset 4.20.12.1
Intlk and Reg Reset 4.20.12.1
Bring Up Man Go To E 4.20.06.1
Bring Up Clear Ctrl 4.20.06.1
Store Ctrl 2.09.00.1
Bring Up Clear Stg 4.20.06.1
Turn Off MST Tgr 4.20.11.1
Take 1 I Cycle Then E's
Write Zero
PC to AS 3.05.09.1
AS to AR 3.06.18.1
PC Advance on Clear 4.20.06.1
E 6
PC Two Tgr 3.05.00.1
Off
On
End Op on Clear 4.20.06.1
E6
Turn On End Op 8.00.09.1
E10
Turn On Mst Tgr 4.20.11.1
E10

FIGURE 6.1-4. CLEAR STORAGE

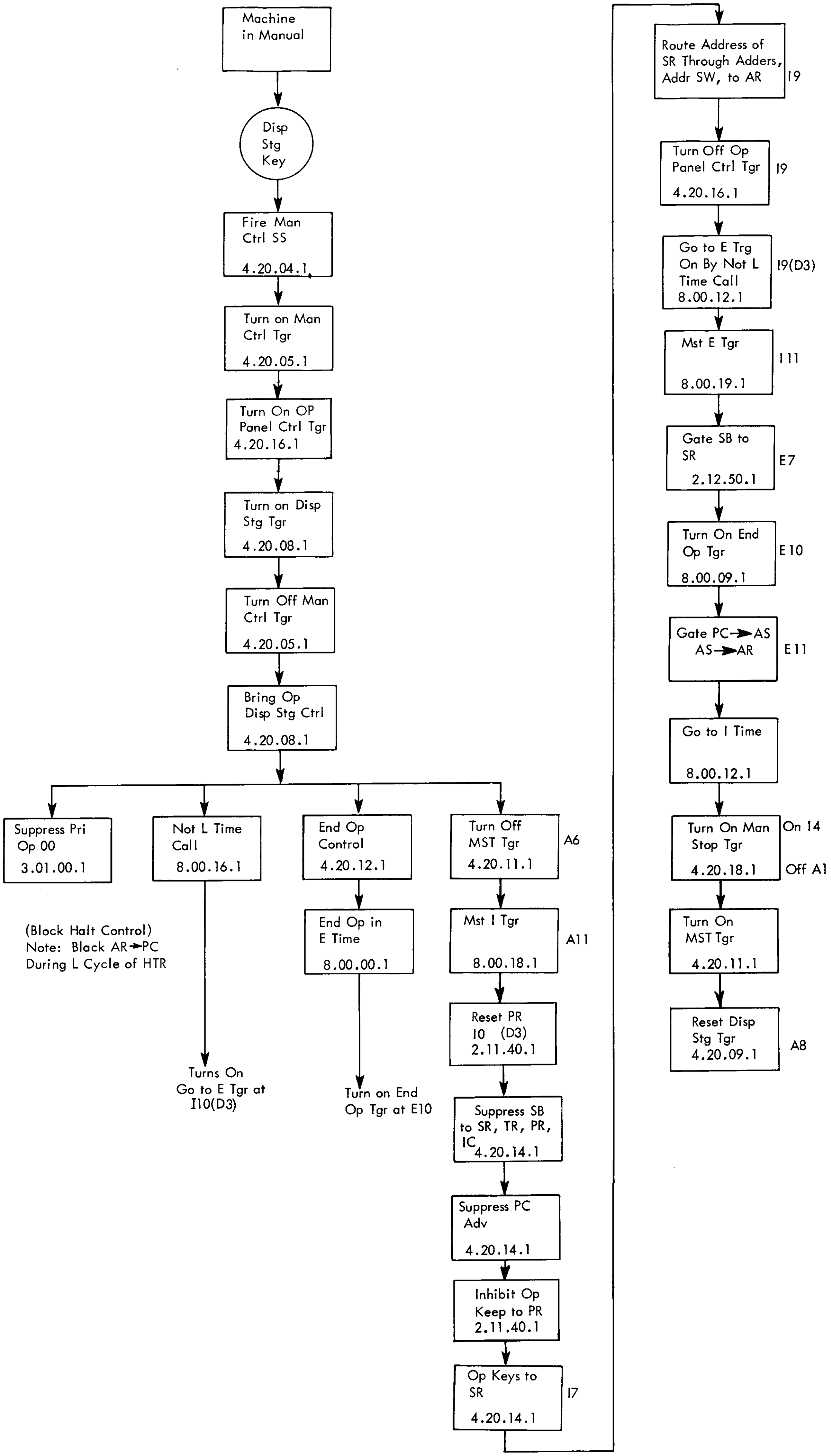

Machine in Manual
Disp Stg Key
Fire Man Ctrl SS 4.20.04.1
Turn on Man Ctrl Tgr 4.20.05.1
Turn On OP Panel Ctrl Tgr 4.20.16.1
Turn on Disp Stg Tgr 4.20.08.1
Turn Off Man Ctrl Tgr 4.20.05.1
Bring Op Disp Stg Ctrl 4.20.08.1
Suppress Pri Op 00 3.01.00.1
(Block Halt Control)
Note: Black AR→PC During L Cycle of HTR
Not L Time Call 8.00.16.1
Turns On Go to E Tgr at I10(D3)
End Op Control 4.20.12.1
End Op in E Time 8.00.00.1
Turn on End Op Tgr at E10
Turn Off MST Tgr 4.20.11.1
A6
Mst I Tgr 8.00.18.1
A11
Reset PR 10 (D3) 2.11.40.1
Suppress SB to SR, TR, PR, IC 4.20.14.1
Suppress PC Adv 4.20.14.1
Inhibit Op Keep to PR 2.11.40.1
Op Keys to SR 4.20.14.1
I7
Route Address of SR Through Adders, Addr SW, to AR
I9
Turn Off Op Panel Ctrl Tgr 4.20.16.1
I9
Go to E Trg On By Not L Time Call 8.00.12.1
I9(D3)
Mst E Tgr 8.00.19.1
I11
Gate SB to SR 2.12.50.1
E7
Turn On End Op Tgr 8.00.09.1
E10
Gate PC→AS AS→AR
E11
Go to I Time 8.00.12.1
Turn On Man Stop Tgr 4.20.18.1
On I4
Off A1
Turn On MST Tgr 4.20.11.1
Reset Disp Stg Tgr 4.20.09.1
A8

FIGURE 6.1-5. DISPLAY STORAGE

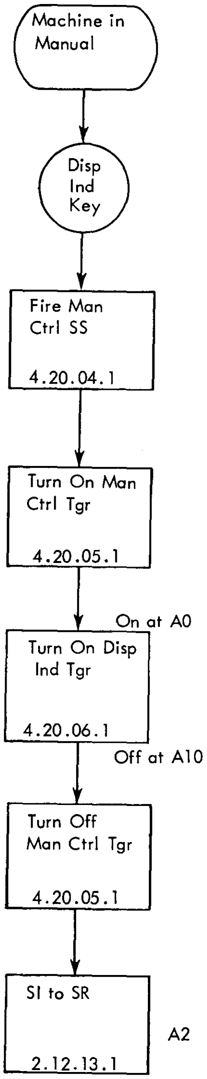

Machine in Manual
Disp Ind Key
Fire Man Ctrl SS 4.20.04.1
Turn On Man Ctrl Tgr 4.20.05.1
On at A0
Turn On Disp Ind Tgr 4.20.06.1
Off at A10
Turn Off Man Ctrl Tgr 4.20.05.1
SI to SR 2.12.13.1
A2

FIGURE 6.1-6. DISPLAY INDICATORS

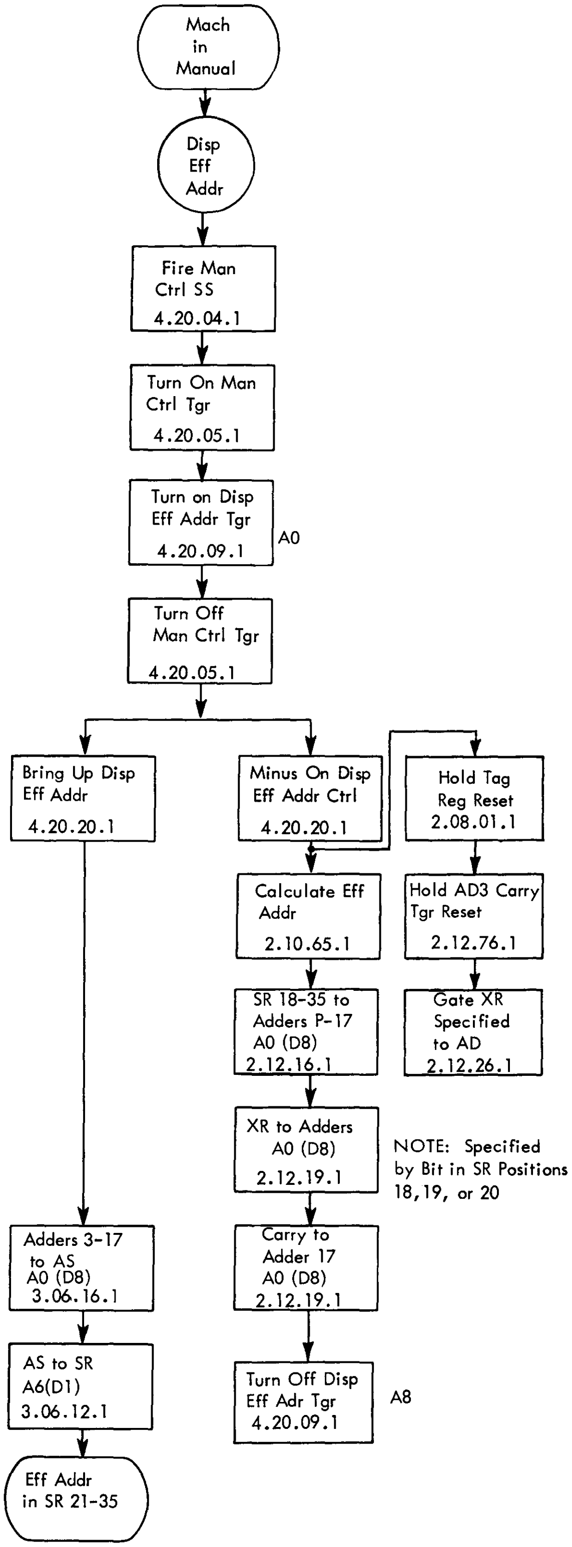

Mach in Manual
Disp Eff Addr
Fire Man Ctrl SS 4.20.04.1
Turn On Man Ctrl Tgr 4.20.05.1
Turn on Disp Eff Addr Tgr 4.20.09.1
A0
Turn Off Man Ctrl Tgr 4.20.05.1
Bring Up Disp Eff Addr 4.20.20.1
Minus On Disp Eff Addr Ctrl 4.20.20.1
Hold Tag Reg Reset 2.08.01.1
Calculate Eff Addr 2.10.65.1
Hold AD3 Carry Tgr Reset 2.12.76.1
SR 18-35 to Adders P-17 A0 (D8) 2.12.16.1
Gate XR Specified to AD 2.12.26.1
XR to Adders A0 (D8) 2.12.19.1
NOTE: Specified by Bit in SR Positions 18,19, or 20
Adders 3-17 to AS A0 (D8) 3.06.16.1
Carry to Adder 17 A0 (D8) 2.12.19.1
AS to SR A6(D1) 3.06.12.1
Turn Off Disp Eff Adr Tgr 4.20.09.1
A8
Eff Addr in SR 21-35

FIGURE 6.1-7. DISPLAY EFFECTIVE ADDRESS

Display Effective Address (Figure 6.1-7)

The display effective address key initiates the calculation of the effective address of the instruction in the storage register. The actual address from the storage register is combined with the two's complement of the specified index register to produce the effective address. The calculated address is then placed in the address portion of the storage register. Positions 1 through 20 of the storage register are set to zero. The effective address may be calculateđ only once because the positions 18, 19, and 20 of the storage register have been set to zero, an indication of no index register.

Single Step (Figure 6.1-8)

Pressing the single-step key results in executing the instruction whose address is in the instruction counter prior to key depression. The instruction counter will be advanced, or altered, under control of the instruction executed, once for each time the key is pressed. If an I-O operation is executed, the machine will continue to execute instructions at high speed until the end of the I-O operation. If the continuous enter instruction switch is on, the single-step key is pressed, and the 7090 is in manual status, the instruction set in the OP keys will be executed once. The single-step key is effective only if the system is in manual status, and not in program stop status.

Multiple Step (Figure 6.1-9)

This key is effective only in manual status with the program stop trigger on. The multiple-step key causes the repetition of single-step operations. The rate of operation is under control of a toggle switch located on the customer engineer test panel. The operator has the choice of low speed with a delay of 104 milliseconds between each instruction, or high speed with a delay of 24 milliseconds between each instruction. The program will continue to run as long as the multiple step key is pressed, or until a program halt is decoded.

Enter MQ Key

The enter MQ key provides a means for loading the MQ register from the operator's entry keys. The information may then be loaded into storage by placing the instruction STQ along with the desired address in the entry keys and depressing the enter instruction key. The enter MQ key is effective only when in manual status. See Figure 6.1-10.

Enter Instruction (Figure 6.1-11)

With the machine in manual status, the enter instruction key executes completely and correctly any legitimate instruction entered in the operator's panel entry keys. The contents of the instruction counter remain unchanged when an instruction is executed (except for a transfer or a skip type of instruction). The key is effective only in manual status.

Load Cards (Figure 6.1-12)

The load cards key causes a reset of the instruction counter, address register, program stop trigger, simulate, and all channels not in manual status. Pressing the load cards key then causes a select of the card reader on channel A, reads the first three words, and proceeds to storage locations zero for the next command word. This is accomplished by bringing up "load ctrl," which fires a 1-usec single-shot. The auto load trigger goes on in the CPU, selecting the card reader on channel A. The word counter is set to three, and indicator S is turned on in channel. Three words from the first

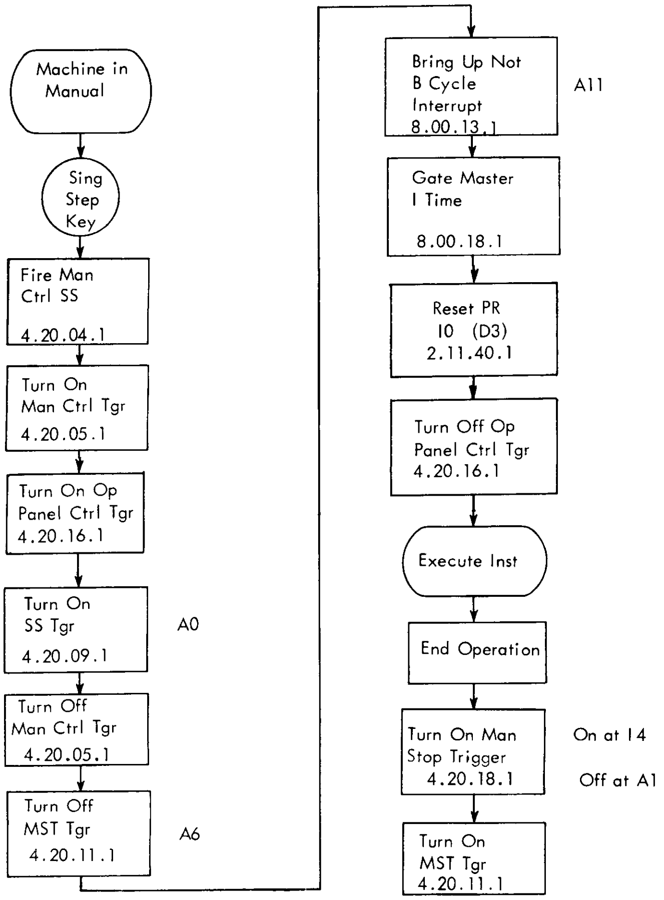

Machine in Manual
Sing Step Key
Fire Man Ctrl SS 4.20.04.1
Turn On Man Ctrl Tgr 4.20.05.1
Turn On Op Panel Ctrl Tgr 4.20.16.1
Turn On SS Tgr 4.20.09.1
A0
Turn Off Man Ctrl Tgr 4.20.05.1
Turn Off MST Tgr 4.20.11.1
A6
Bring Up Not B Cycle Interrupt 8.00.13.1
A11
Gate Master I Time 8.00.18.1
Reset PR 10 (D3) 2.11.40.1
Turn Off Op Panel Ctrl Tgr 4.20.16.1
Execute Inst
End Operation
Turn On Man Stop Trigger 4.20.18.1
On at I 4
Off at A1
Turn On MST Tgr 4.20.11.1

FIGURE 6.1-8. SINGLE STEP KEY

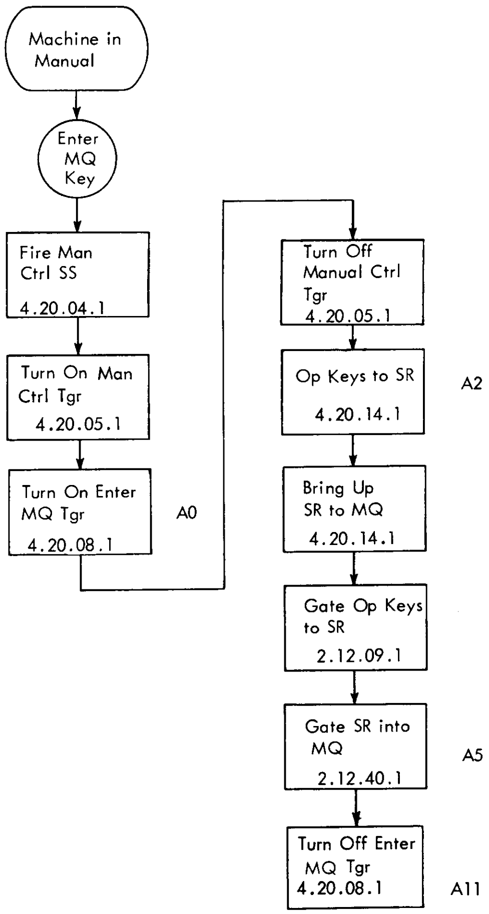

Machine in Manual
Enter MQ Key
Fire Man Ctrl SS 4.20.04.1
Turn On Man Ctrl Tgr 4.20.05.1
Turn On Enter MQ Tgr 4.20.08.1
A0
Turn Off Manual Ctrl Tgr 4.20.05.1
Op Keys to SR 4.20.14.1
A2
Bring Up SR to MQ 4.20.14.1
Gate Op Keys to SR 2.12.09.1
Gate SR into MQ 2.12.40.1
A5
Turn Off Enter MQ Tgr 4.20.08.1
A11

FIGURE 6.1-10. ENTER MQ

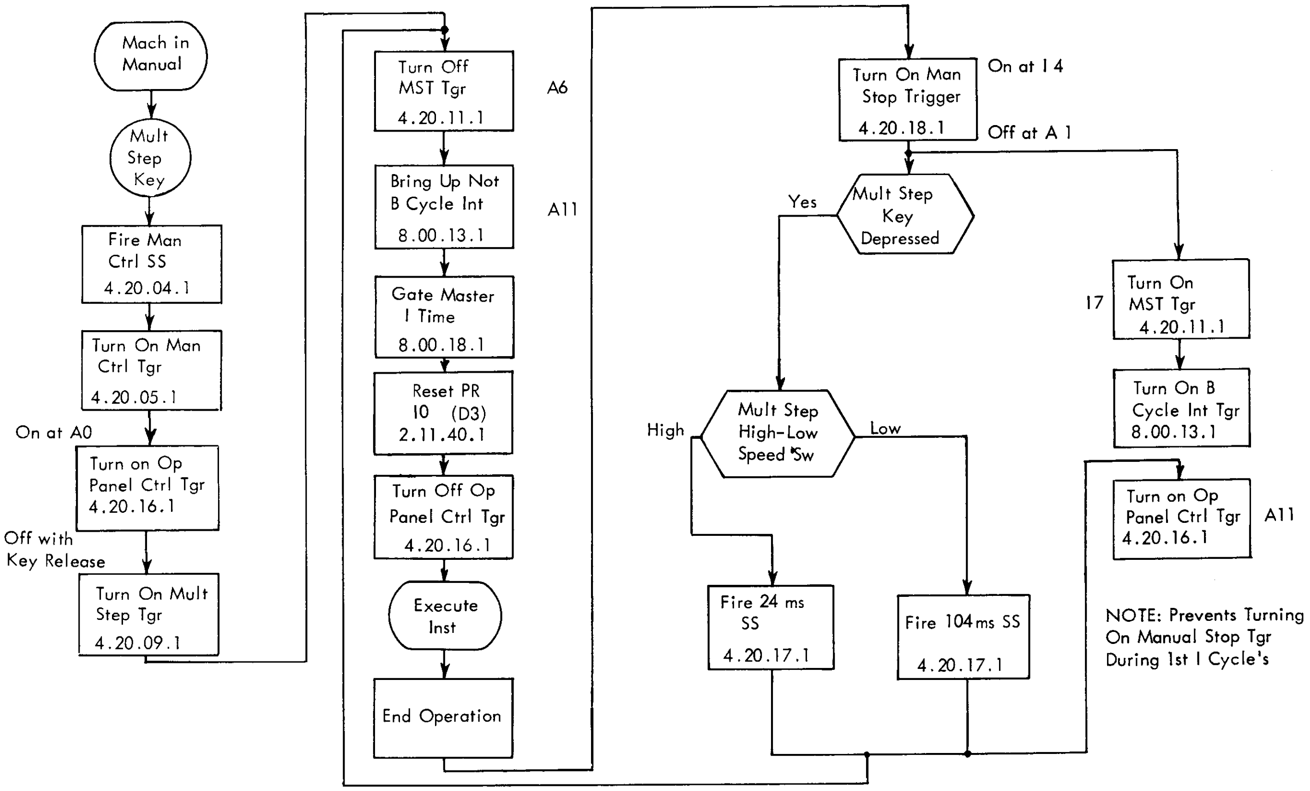

Mach in Manual
Mult Step Key
Fire Man Ctrl SS 4.20.04.1
Turn On Man Ctrl Tgr 4.20.05.1
On at A0
Turn on Op Panel Ctrl Tgr 4.20.16.1
Off with Key Release
Turn On Mult Step Tgr 4.20.09.1
Turn Off MST Tgr 4.20.11.1
A6
Bring Up Not B Cycle Int 8.00.13.1
A11
Gate Master I Time 8.00.18.1
Reset PR 10 (D3) 2.11.40.1
Turn Off Op Panel Ctrl Tgr 4.20.16.1
Execute Inst
End Operation
Turn On Man Stop Trigger 4.20.18.1
On at I 4
Off at A 1
Mult Step Key Depressed
Yes
Mult Step High-Low Speed Sw
High
Low
Fire 24 ms SS 4.20.17.1
Fire 104ms SS 4.20.17.1
17
Turn On MST Tgr 4.20.11.1
Turn On B Cycle Int Tgr 8.00.13.1
Turn on Op Panel Ctrl Tgr 4.20.16.1
A11
NOTE: Prevents Turning On Manual Stop Tgr During 1st I Cycle's

FIGURE 6.1-9. MULTIPLE STEP KEY

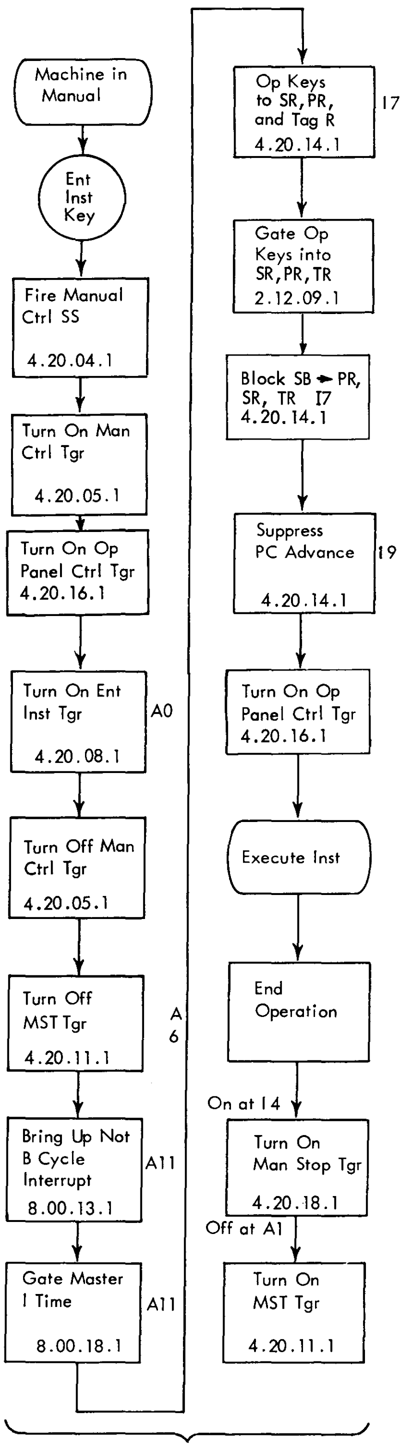

Machine in Manual
Ent Inst Key
Fire Manual Ctrl SS 4.20.04.1
Turn On Man Ctrl Tgr 4.20.05.1
Turn On Op Panel Ctrl Tgr 4.20.16.1
Turn On Ent Inst Tgr 4.20.08.1
A0
Turn Off Man Ctrl Tgr 4.20.05.1
Turn Off MST Tgr 4.20.11.1
A 6
Bring Up Not B Cycle Interrupt 8.00.13.1
A11
Gate Master I Time 8.00.18.1
A11
Op Keys to SR, PR, and Tag R 4.20.14.1
17
Gate Op Keys into SR, PR, TR 2.12.09.1
Block SB → PR, SR, TR I7 4.20.14.1
Suppress PC Advance 4.20.14.1
19
Turn On Op Panel Ctrl Tgr 4.20.16.1
Execute Inst
End Operation
On at I4
Turn On Man Stop Tgr 4.20.18.1
Off at A1
Turn On MST Tgr 4.20.11.1

FIGURE 6.1-11. ENTER INSTRUCTION

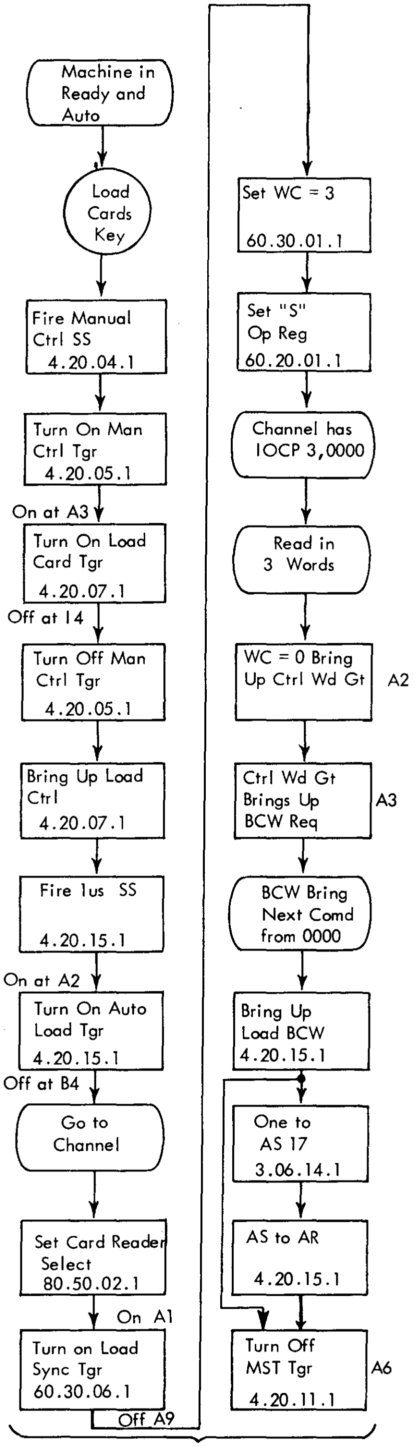

Machine in Ready and Auto
Load Cards Key
Fire Manual Ctrl SS 4.20.04.1
Turn On Man Ctrl Tgr 4.20.05.1
On at A3
Turn On Load Card Tgr 4.20.07.1
Off at I4
Turn Off Man Ctrl Tgr 4.20.05.1
Bring Up Load Ctrl 4.20.07.1
Fire 1us SS 4.20.15.1
On at A2
Turn On Auto Load Tgr 4.20.15.1
Off at B4
Go to Channel
Set Card Reader Select 80.50.02.1
On A1
Turn on Load Sync Tgr 60.30.06.1
Off A9
Set WC = 3 60.30.01.1
Set "S" Op Reg 60.20.01.1
Channel has IOCP 3,0000
Read in 3 Words
WC = 0 Bring Up Ctrl Wd Gt
A2
Ctrl Wd Gt Brings Up BCW Req
A3
BCW Bring Next Comd from 0000
Bring Up Load BCW 4.20.15.1
One to AS 17 3.06.14.1
AS to AR 4.20.15.1
Turn Off MST Tgr 4.20.11.1
A6

FIGURE 6.1-12. LOAD CARDS

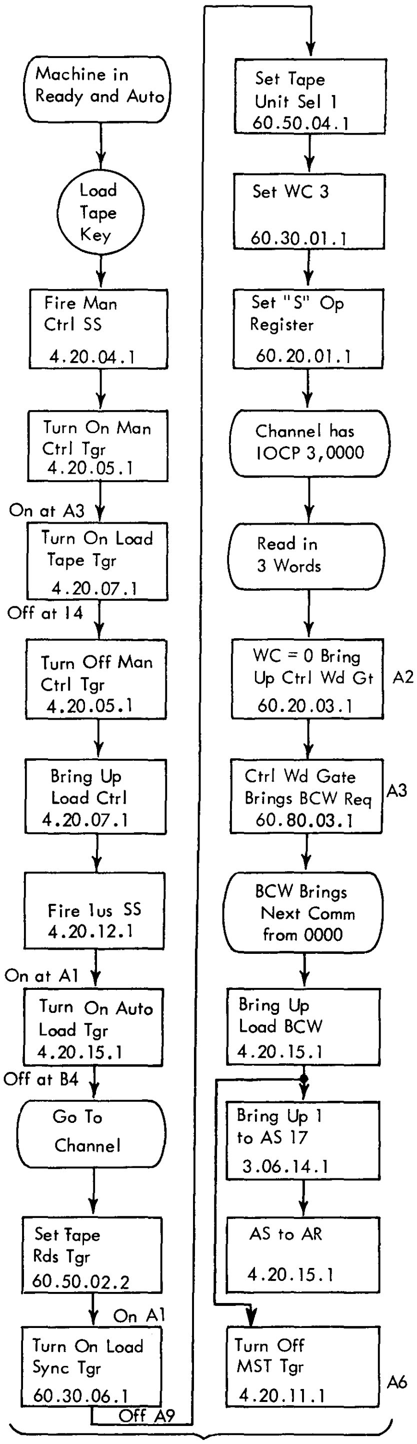

Machine in Ready and Auto
Load Tape Key
Fire Man Ctrl SS 4.20.04.1
Turn On Man Ctrl Tgr 4.20.05.1
On at A3
Turn On Load Tape Tgr 4.20.07.1
Off at I4
Turn Off Man Ctrl Tgr 4.20.05.1
Bring Up Load Ctrl 4.20.07.1
Fire 1us SS 4.20.12.1
On at A1
Turn On Auto Load Tgr 4.20.15.1
Off at B4
Go To Channel
Set Tape Rds Tgr 60.50.02.2
On A1
Turn On Load Sync Tgr 60.30.06.1
Off A9
Set Tape Unit Sel 1 60.50.04.1
Set WC 3 60.30.01.1
Set "S" Op Register 60.20.01.1
Channel has IOCP 3,0000
Read in 3 Words
WC = 0 Bring Up Ctrl Wd Gt 60.20.03.1
A2
Ctrl Wd Gate Brings BCW Req 60.80.03.1
A3
BCW Brings Next Comm from 0000
Bring Up Load BCW 4.20.15.1
Bring Up 1 to AS 17 3.06.14.1
AS to AR 4.20.15.1
Turn Off MST Tgr 4.20.11.1
A6

FIGURE 6.1-13. LOAD TAPE

card enter storage location zero, one, and two. The word entered in address zero should be a control word. When the word counter goes to zero, the channel asks for another control word from location zero. The CPU gets its next instruction from location one.

Load Tape Key (Figure 6.1-13)

The load tape key works the same as the load cards key, causing a reset to the CPU and all channels not in manual status. The load tape key also causes a read select of tape unit 1 on channel A, IOCP word count of 3. The next command for the channel will come from location zero, and the next instruction for CPU will come from location 00001.

Sense Control

There are two types of sense controls on the operator's panel: sense lights and sense switches. The conditions of these sense devices may be checked by machine instructions and used to control program flow.

The four sense lamps on the panel may be turned on or off by instructions in the main program and then checked by the sense instructions. The condition of the lamp determines if the program steps are to be skipped.

The six sense switches may be set on or off from the operator's panel. The condition of the switch may then be checked by sense instructions in the program to determine whether to skip the following program step.

6.2.00 CUSTOMER ENGINEER'S TEST PANEL

In addition to the indicators and manual controls on the operator's panel, the customer engineer has at his disposal the indicators and controls on the customer engineer's test panel.

The indicators provide a means for checking the address register contents, the tally counter, various test triggers, and the cycle time. The switches and jacks provide means to continually execute any instruction, control I-O operation, control B time, and step through instructions cycle by cycle.

Figure 6.2-1 shows the layout of indicators and controls for this panel and designates system page locations.

6.2.01 Indicators

Indicators on the customer engineer's test panel are of the incandescent type. Indicators connected to test triggers are on when the trigger is on and off when the trigger is off. Indicators concerned with the registers and counters signify a 1 when they are on and a 0 when off.

Address Register

The address register contains the address portion of the instruction under execution.

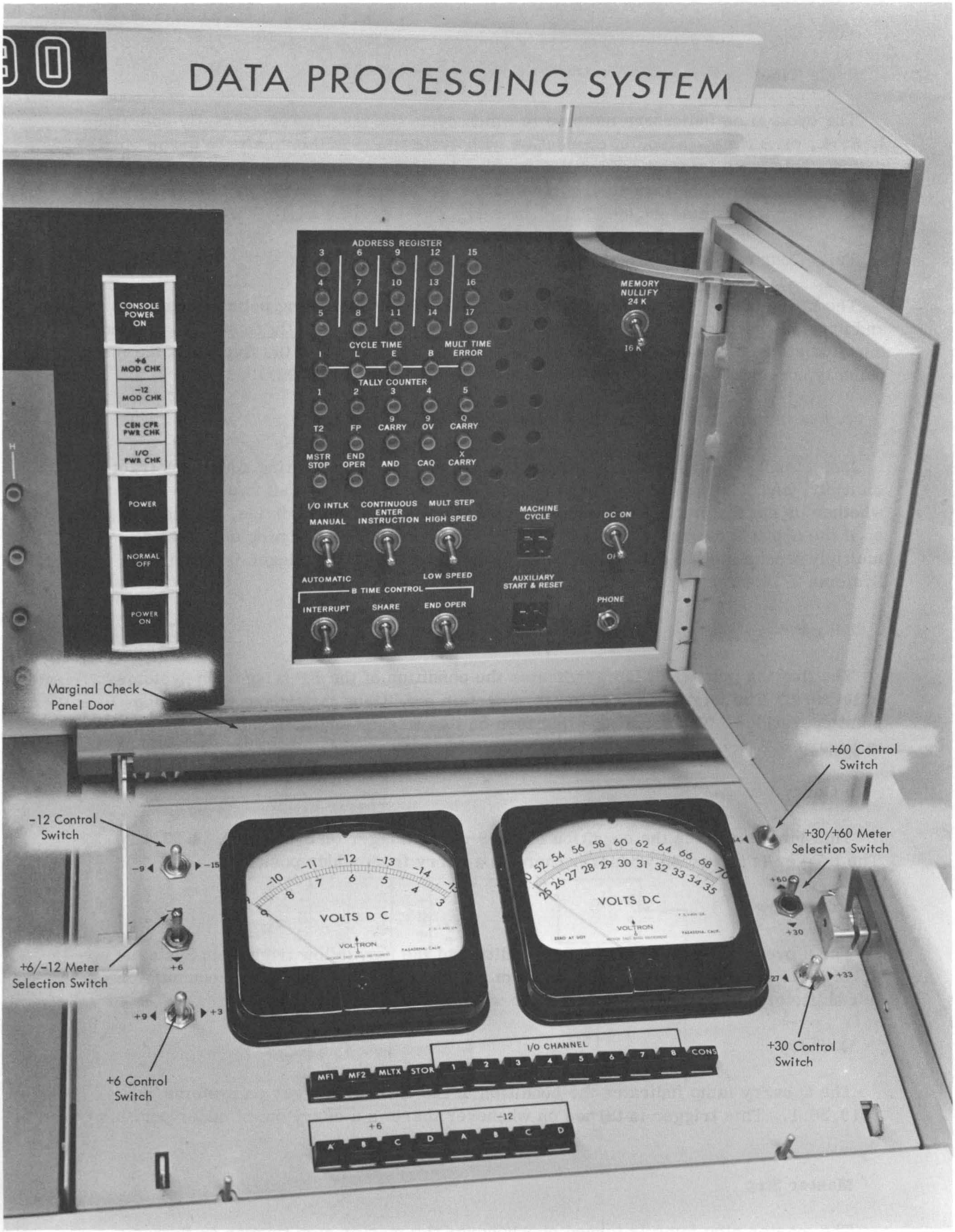


FIGURE 6.2-1. CUSTOMER ENGINEER TEST PANEL

Cycle Time

The cycle time indicators indicate in which cycle the machine is currently operating: I, E, L, or B time. Also, in connection with cycle time, is the "multiple error" indicator. This lamp is on whenever the machine is trying to perform two different cycles at the same time other than an L cycle and a B cycle, which is a normal share operation. Refer to Systems 08.00.17.1

Tally Counter

The tally counter differentiates between the L cycles of floating point instructions and provides gating for their different operational steps. The tally counter is divided into five steps. The lamps on the test panel indicate in which of the five steps the machine is currently operating. Refer to Systems 02.10.21.1 and 02.10.22.1.

T-2

Indicator T-2 signifies the condition of the T-2 trigger in floating add or subtract, multiply, and floating divide. During floating add or subtract, T-2 is used to indicate whether or not the multiplier quotient equals 0. During floating divide, T-2 is turned on if the quotient is greater than 2. T-2 is turned on at the beginning of a floating multiply second step to gate second step operations. The T-2 trigger is found on Systems 2.10.38.1.

FP

The floating point (FP) lamp indicates the condition of the FP trigger on Systems 2.10.29.1. The trigger is used to store certain conditions throughout the floating point operations. The conditions that turn on this trigger and indicator are shown in Figure 6.2-2.

9 Carry

This lamp indicates the condition of the 9 carry trigger on Systems 2.10.37.1. The trigger is turned on whenever there is a carry from adder position 9.

9 Overflow

The 9 overflow lamp signifies the condition of the 9 overflow trigger on Systems 2.10.39.1. The trigger and lamp come on whenever AC 9=1 during an accumulator left shift, or during a multiply add cycle when there is an adder 9 carry.

Q Carry

The Q carry lamp indicates the condition of the Q carry trigger on Systems 2.10.36.1. This trigger is turned on whenever there is a carry out of adder position Q.

Master Stop

The master stop lamp is turned on whenever the master stop trigger on Systems 4.20.11.1 is turned on.

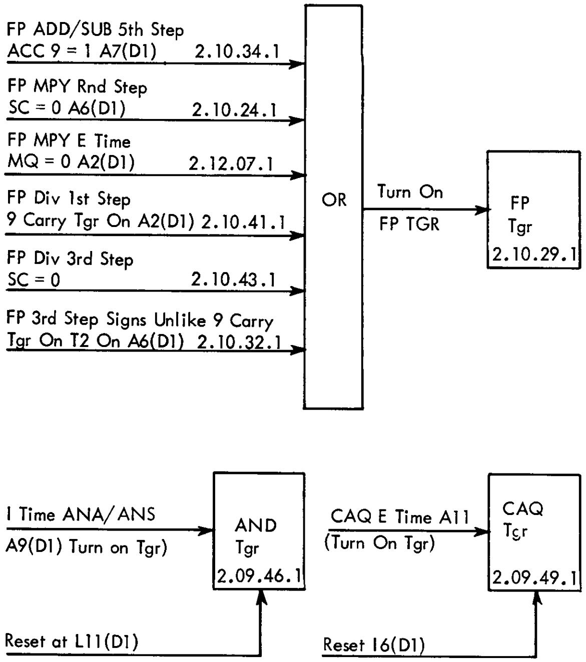


FIGURE 6.2-2. AND, CAQ, AND FP TRIGGERS

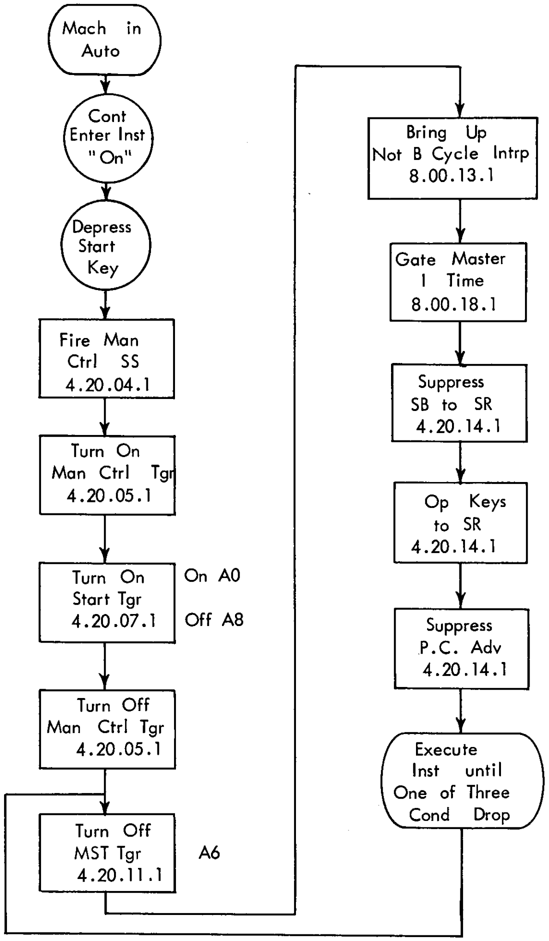


FIGURE 6.2-3. CONTINUOUS ENTER INSTRUCTION

End Operation

The end operation indicator is turned on whenever the end operation trigger on Systems 8.00.09.1 is turned on.

AND

The AND lamp indicates the condition of the AND trigger on Systems 2.09.46.1. It can be used to distinguish between the first and second E cycles of an ANS operation. It is on during the first E cycle of an ANS or ANA and off during the second E cycle of an ANS. See Figure 6.2-2.

CAQ

The CAQ indicator signifies the condition of the CAQ trigger on Systems 2.09.49.1. It denotes the difference between the first and succeeding E cycles of a CAQ instruction. The trigger is off for the first E cycle and on for the remaining E cycles and I time of next instruction. See Figure 6.2-2.

X Carry

The X carry lamp indicates the condition of the X carry trigger on Systems 2.12.76.1. The trigger is on when the machine is not in memory nullification mode and a carry occurs from adder position 3. If the machine is in memory nullification mode, a carry from adder position 4, when in 16K mode, or a carry from adder position 5, when in 24K mode, turns on the trigger.

6.2.02 Switches

I-O Interlock Switch

The I-O interlock switch is effective only when the system is in manual status. It has two positions: automatic and manual. With both the automatic-manual and I-O interlock switches set in the manual position, the computer will stop after executing each instruction. With the I-O interlock switch set to automatic, the machine will not stop if an I-O device is in operation. The normal setting (automatic) of the I-O interlock switch allows the computer to continue at high speed after I-O selection, to allow normal operation of the I-O device. The machine will stop after each instruction, providing no I-O device or data channel is in use.

Continuous Enter Instruction (Figure 6.2-3)

The continuous enter instruction switch is effective in automatic or manual status. With this switch on, all instructions are obtained from the operator's panel entry keys, not memory. The instruction counter is not used and does not advance. The IC contents will not be altered unless a skip, trap, or transfer results from the instruction in the entry keys. If the system is in automatic status, the instruction will be executed at normal operating speeds; if in manual status, the speed will be under control of multi-step or single-step operations.

Multiple Step High and Low Speed

The following explanation is with reference to Section 3.12.00. With this switch in the high position, the multiple step key on the operator's panel performs program steps, one after another, every 24 ms. With the switch in the low position, program steps are performed every 104 ms. The high-speed position is advantageous when stepping through index loops.

B Cycle Controls

These switches allow the machine to operate normally when they are in the down position. In the up position, they modify normal operation as follows:

Interrupt:	Prevents the 7606 from obtaining a B cycle by interrupting a CPU instruction.
Share:	Prevents the 7606 access to core storage during CPU L cycles.
End Operation:	Prevents the 7606 from taking a B cycle when a CPU instruction ends operation.

The switches can be used if the customer engineer wants to test the end operation procedure of obtaining a B cycle. The interrupt and share switches would then be put in the up position. This prevents a B cycle from being started in any manner except end operation. Any of these switches in the up position turns off the ready light.

Machine Cycle Key (Figure 6.2-4)

The machine cycle receptacle on the customer engineer's test panel accepts the machine cycle key. This key is used to sequentially step through the basic machine cycles I, E, and L. When the machine cycle key is inserted into the customer engineer's panel and pressed, the machine cycle trigger and machine cycle gate trigger are turned on at A-0. The machine cycle trigger turns off the MST and allows for the proper cycle time, after which the MST is turned on again. The machine cycle gate does two things. First, it allows only three shifts per cycle on a shift operation, and forces a wait until the shift counter equals zero before ending operation in L time. Second, it is used to hold up "manual I time cntl" to insure that the machine will use a full I cycle.

"Manual I time cntl" is AND'ed with the output of the master I time trigger (Systems 08.00.18.1). Thus, if "manual I time cntl" drops, I time is finished. Usually, turning on the MST at I7 causes "manual I time cntl" to fall and stop I time at I7. This allows the machine to finish with an instruction before the following instruction is brought in. Normally, the go to I time trigger (Systems 08.00.12.1) is turned off at I9, but because there is no I9, it remains on. The all pulse is blocked from resetting the master I time trigger (Systems 08.00.08.1) by "minus on not go to I time." Therefore, if we start again, the MST is turned off at A6, bringing up "manual I time cntl" and starting the cycle at I6.

To insure getting a full I cycle when using the machine cycle key, the machine cycle gate trigger holds up "manual I time cntl" in the OR circuit on Systems 04.20.10.1.

When the key is not in use, a plug (sent with the system) that shorts pins 1 and 3 must be inserted.

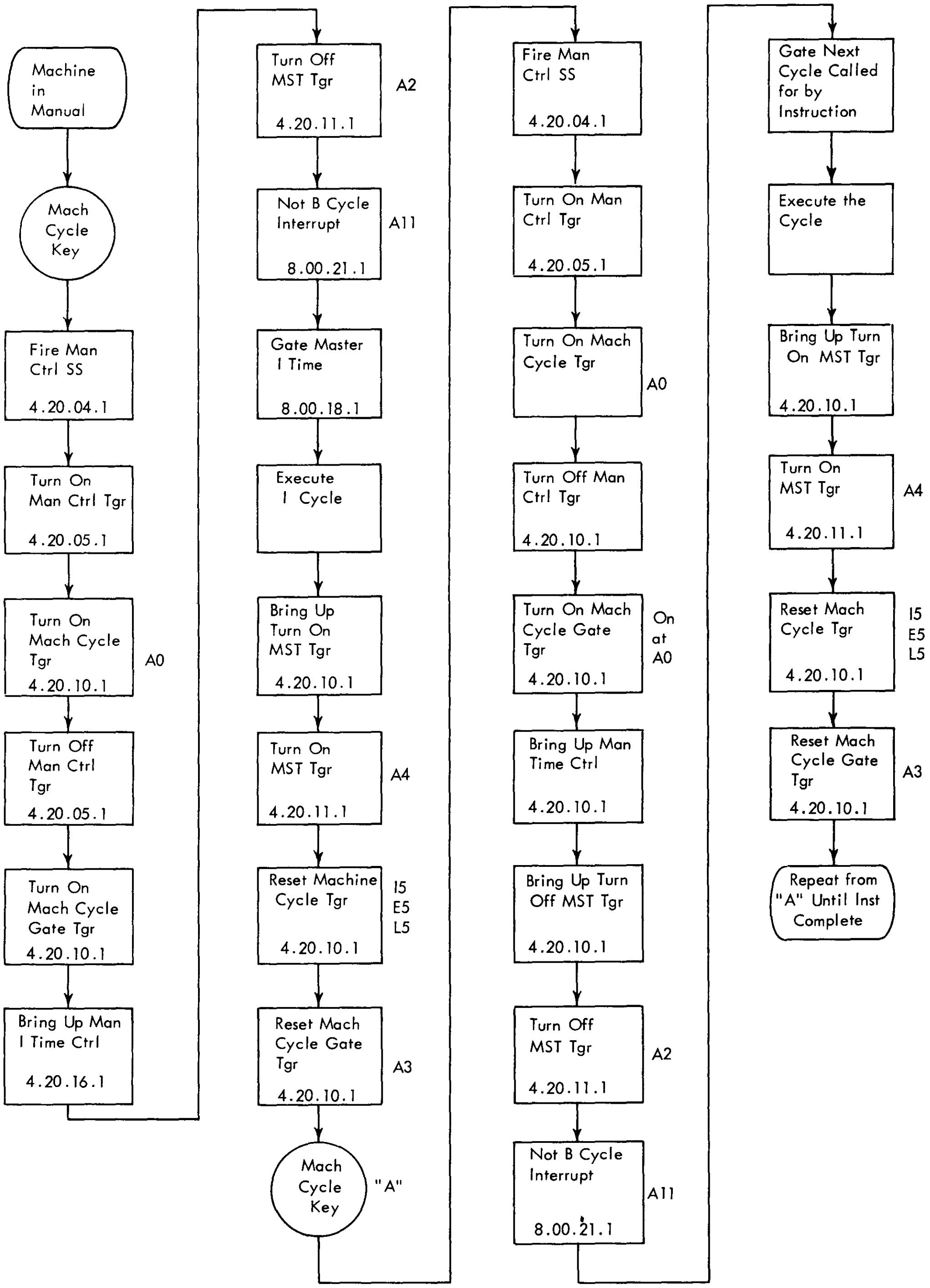

Machine in Manual
Mach Cycle Key
Fire Man Ctrl SS 4.20.04.1
Turn On Man Ctrl Tgr 4.20.05.1
Turn On Mach Cycle Tgr 4.20.10.1
A0
Turn Off Man Ctrl Tgr 4.20.05.1
Turn On Mach Cycle Gate Tgr 4.20.10.1
Bring Up Man I Time Ctrl 4.20.16.1
Turn Off MST Tgr 4.20.11.1
A2
Not B Cycle Interrupt 8.00.21.1
A11
Gate Master I Time 8.00.18.1
Execute I Cycle
Bring Up Turn On MST Tgr 4.20.10.1
Turn On MST Tgr 4.20.11.1
A4
Reset Machine Cycle Tgr 4.20.10.1
I5
E5
L5
Reset Mach Cycle Gate Tgr 4.20.10.1
A3
Mach Cycle Key
"A"
Fire Man Ctrl SS 4.20.04.1
Turn On Man Ctrl Tgr 4.20.05.1
Turn On Mach Cycle Tgr
A0
Turn Off Man Ctrl Tgr 4.20.10.1
Turn On Mach Cycle Gate Tgr 4.20.10.1
On at A0
Bring Up Man Time Ctrl 4.20.10.1
Bring Up Turn Off MST Tgr 4.20.10.1
Turn Off MST Tgr 4.20.11.1
A2
Not B Cycle Interrupt 8.00.21.1
A11
Gate Next Cycle Called for by Instruction
Execute the Cycle
Bring Up Turn On MST Tgr 4.20.10.1
Turn On MST Tgr 4.20.11.1
A4
Reset Mach Cycle Tgr 4.20.10.1
I5
E5
L5
Reset Mach Cycle Gate Tgr 4.20.10.1
A3
Repeat from "A" Until Inst Complete

FIGURE 6.2-4. MACHINE CYCLE KEY

Auxiliary Start and Reset

This jack accepts the auxiliary start and reset key. The key is on a long cable, giving the customer engineer a means for starting and resetting the machine at points other than the console.

Phone Jack

The phone jack, in conjunction with the phone jacks on the other units in the system, provides a means of communication between customer engineers working at different units.

16K/24K Mode Switch

This switch is used with the compatibility package. When it is in the 16K position, 16K core storage positions are available to the 704 program; in the 24K position, 8K core storage positions are available to the 704 program.

DC On

The DC on switch controls the 400-cycle power supplied only to the 7151. Putting the DC on switch in the off position will immediately remove all power to the console, except the convenience outlets and the reset motor (operator's keys). All voltages should be normal about ten seconds after turning this switch to the on position.

6.3.00 MARGINAL CHECK

The system biasing network can be useful to the customer engineer. The controls are located on the marginal check console on the 7151. Any single gate or combination of gates in any single module, or combination of modules, can be biased at the same time. The only exception is the 7302 memory. This module is under control of only one key, A, and the whole module will be biased when this key is pressed.

Each module has a key for selecting it, and each gate has a key with key A for gate A, B for gate B,etc.

When biasing, all gates must be selected prior to varying the voltage. After the voltage has been varied, another gate cannot be selected until returning to normal voltage. If gate A is being biased and it is decided to bias gate B instead, gate A MC relay must be dropped out; therefore it will be necessary to take the MC voltage to normal before selecting gate B. If this is not done, gate A MC relay will have a hold through the B MC relay points. This means that both gates A and B would be biased instead of gate B only, as desired.

It is also possible to vary the +30v and +60v in the 7302. These controls are also located in the 7151, on the MC panel.

When varying the different voltages, there are two meters to monitor the amount of voltage being varied. One meter is for the +30 and +60 in the 7302, and one meter, for the +6 and -12 for the rest of the system. The meters indicate what the voltage is if the MC relays are picked and the points are properly adjusted. A periodic check of the voltages, while biasing, should be made.

7.0.00 REFERENCE INFORMATION

7.1.00 CONDENSED LOGIC

7.1.01 Adders (Systems 02.02.01.1-02.02.37.1)

The adders consist of transistor switching circuits for combining binary numbers into a binary sum. The 7090 can only add, therefore control lines are needed to develop true and complement conditions.

ACC to ADD	02.12.24.1
SR to ADD	02.12.17.1
XR to ADD	02.12.19.1
COMP ACC to ADD	02.12.23.1

Adders are used for many different types of operations and some controls will affect only certain adders. The adders have three types of outputs: sum, carry, and lookahead. Section 3 covers the theory of the adder outputs.

7.1.02 Address Register (Systems 03.06.00.1-03.06.04.1)

The address register is a 15-position register of triggers. It is labeled 3 through 17. The address register signals the desired address to the core control circuits (Figure 7.1-1). Control lines for the address register are:

Address Switches to AR	03.06.05.1
AR to PC	03.06.05.1

7.1.03 Program Register (Systems 03.04.00.1-03.04.03.1)

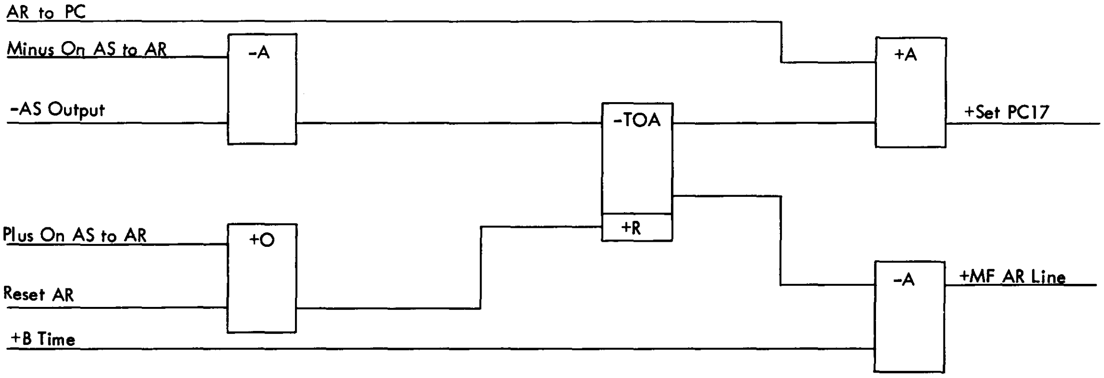
The program register is made up of ten positions, and is used for decoding the instructions during I time. Lines controlling the input and output of the program register (Figure 7.1-2) are:

SR to PR	02.11.40.1
OPK to PR	02.12.09.1
SB to PR	02.05.10.1
Reset PR	02.11.40.1

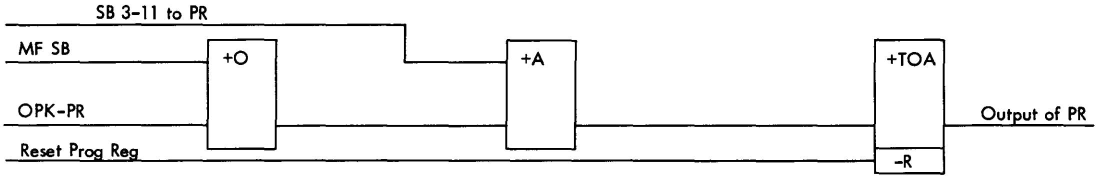
7.1.04 Sense Indicators (Systems 02.06.00.1-02.06.35.1)

The sense indicator register is a 36-position register of triggers. There are two triggers to each position (Figure 7.1-3). Lines controlling the sense indicators are:

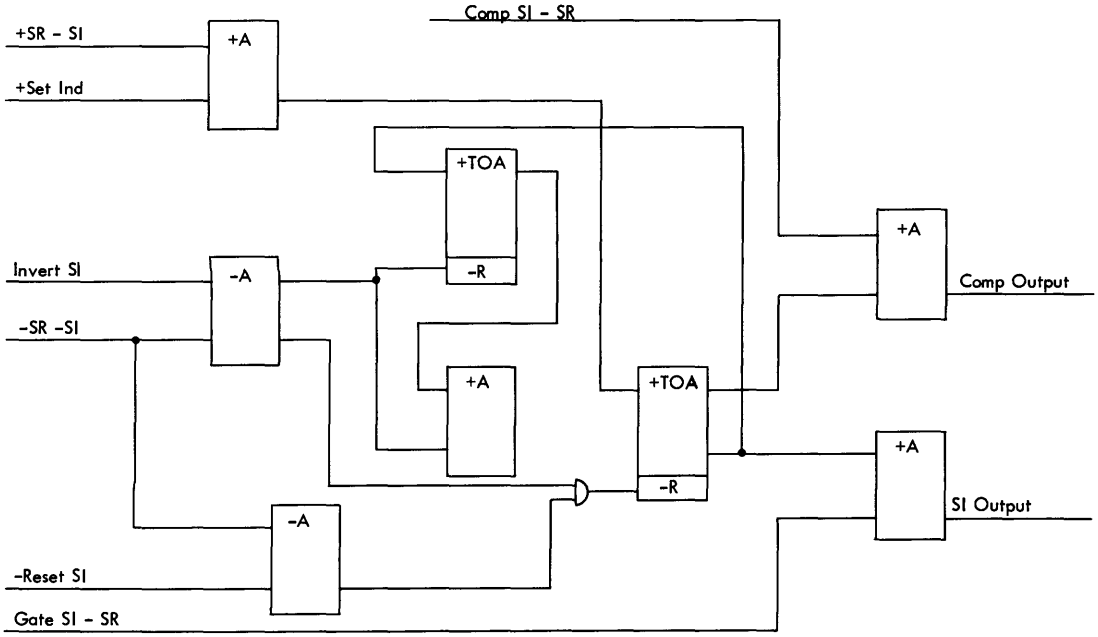
Invert SI	02.12.61.1
Set IND	02.12.63.1
COMP SI to SR	02.12.12.1
SI to SR	02.12.62.1



7.1-1. ADDRESS REGISTER



7.1-2. PROGRAM REGISTER



7.1-3. SENSE INDICATOR

The "invert SI" has no effect on the SI position if there is no bit in the corresponding position of the storage register. With a bit in the SR and the invert line active, the information in the SI will be inverted.

7.1.05 Shift Counter (Systems 03.04.14.1-03.04.19.1)

The shift counter is an 18-position counter of triggers. It is a step-down counter when used for shifting. The shift counter is also used as part of the program register for certain instructions. During a shifting operation, the counter will be stepped every clock pulse (Figure 3.1-7A). Lines controlling the shift counter are:

AS to SC	02.11.78.1
Step SC	02.11.79.1

7.1.06 Program Counter (Systems 03.05.00.1-03.05.07.1)

The program counter is a 16-position counter of triggers. These positions are labeled 2 through 17 (Figure 3.1-7B). Lines controlling the program counter are:

ADV PROG CTR	02.11.51.1
PC to AS	03.05.09.1

The program counter is a count-up counter, and can sequentially step through a program.

7.1.07 Accumulator (Systems 02.03.00.1-02.03.37.1)

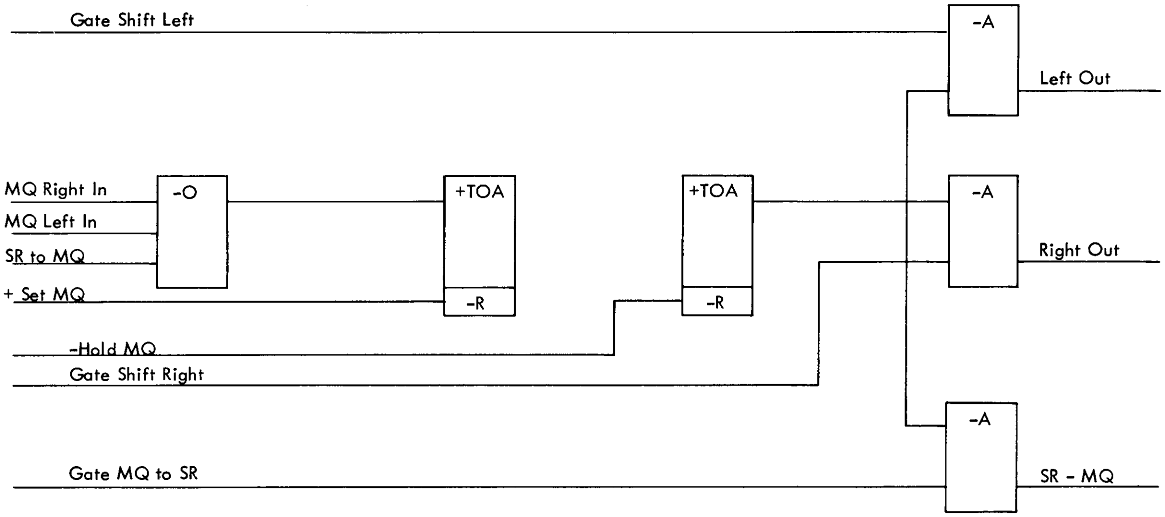
The accumulator is a 38-position register of shift cells. Some trouble may result from the "CP set" being in error. This is true for all registers made up of shift cells. Do not vary the "CP set" until all other possibilities have been checked. Refer to Section 7 for adjustment of the "CP set." Lines controlling the accumulator are shown in Figure 7.1-5.

Shift Right	02.12.36.1
Shift Left	02.12.35.1
SR to ACC	02.10.15.1
Adder to AC	02.12.32.1
Set ACC	02.05.03.1
Hold ACC	02.05.03.1

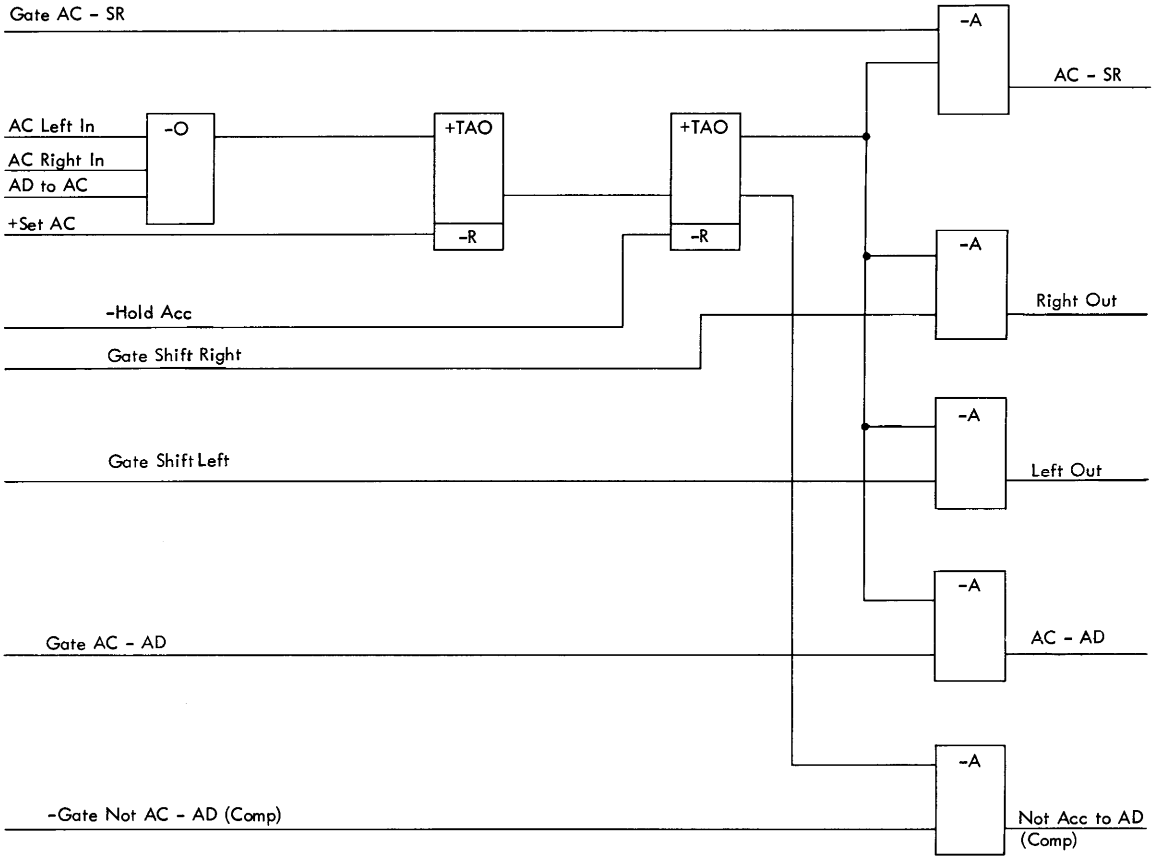
7.1.08 Multiplier Quotient (Systems 02.04.00.1-02.04.35.1)

The MQ is a 36-position register of shift cells. The MQ is used on multiply and divide operations, although some other operations may use it (Figure 7.1-4). Lines that control the MQ are:

SR to MQ	02.12.40.1
Shift MQ Left	02.12.44.1
Shift MQ Right	02.12.45.1
Set MQ	02.05.04 and
Hold MQ	02.05.05.1



7.1-4. MQ POSITION



7.1-5. ACCUMULATOR REGISTER

7.1.09 Index Registers (Systems 02.07.01.1-02.07.15.1)

The index registers are three identical 15-position registers using shift cells. When the registers are read out, the information comes out in complement form (Figure 7.1-6). Lines controlling the index register are:

ADD to XR	02.12.70.1
Gate XRA to ADD	02.12.26.1
Gate XRB to ADD	02.12.26.1
Gate XRC to ADD	02.12.26.1
Set XR	02.12.72.1 and
Hold XR	02.12.73.1

7.1.10 Storage Register (Systems 02.01.00.1-02.01.37.1)

The storage register is a 36-position register of shift cells. The storage register has eight inputs with the following controlling lines (Figure 7.1-7).

ACC to SR	02.12.03.1
ADD to SR	02.12.05.1
MQ to SR	02.12.08.1
OP Keys to SR	02.12.09.1
SR to SR	02.12.10.1
IND to SR	02.12.13.1
Address Switch to SR	03.06.12.1
SB to SR	02.12.50.1
Set SRR	02.05.00.1
Hold SR	02.05.00.1

7.2.00 SERVICE AIDS

7.2.01 One Card Programs

Programs other than the normal diagnostic programs may be used to troubleshoot the machine. For instance, in diagnosing troubles which appear only when the customer's program is being run, the program may be altered slightly to isolate the trouble. Then, if the program is one used regularly, it becomes a good diagnostic in its altered form. In cases of this kind, pass along the information to allow newly released diagnostic programs to incorporate it.

Short programs that you write yourself are also handy when working on a machine. Punch these programs on cards and save the time used to enter them in storage with the keys. Make each card self-loading, if possible, and there will be less cards to keep track of. Keep a file of these one-card programs and add to it. A few samples of one-card programs are given below.

Display Storage

This program causes display of the contents of sequential storage locations, starting with the location entered in the operator's panel keys. The card is self-loading and halts on its first pass to allow the operator's panel keys to be set. After the start key

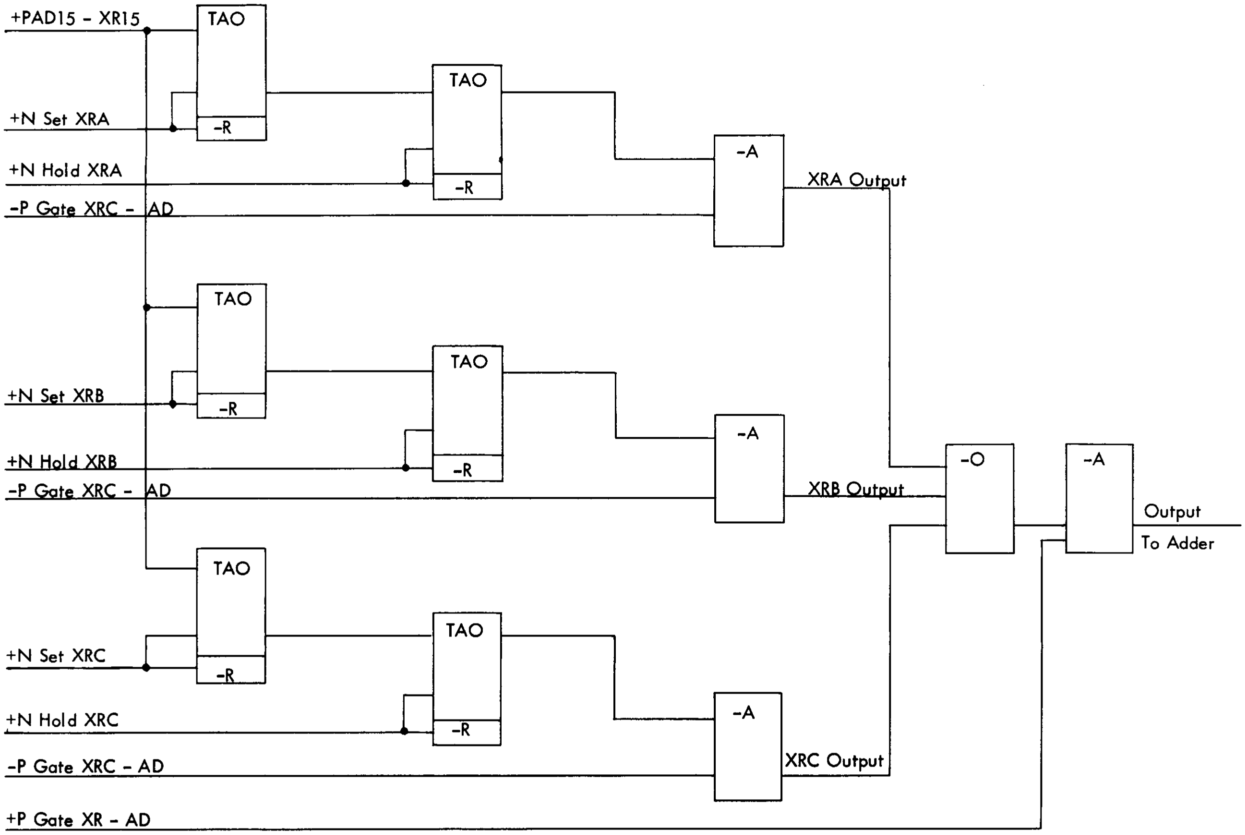


FIGURE 7.1-6. INDEX REGISTER

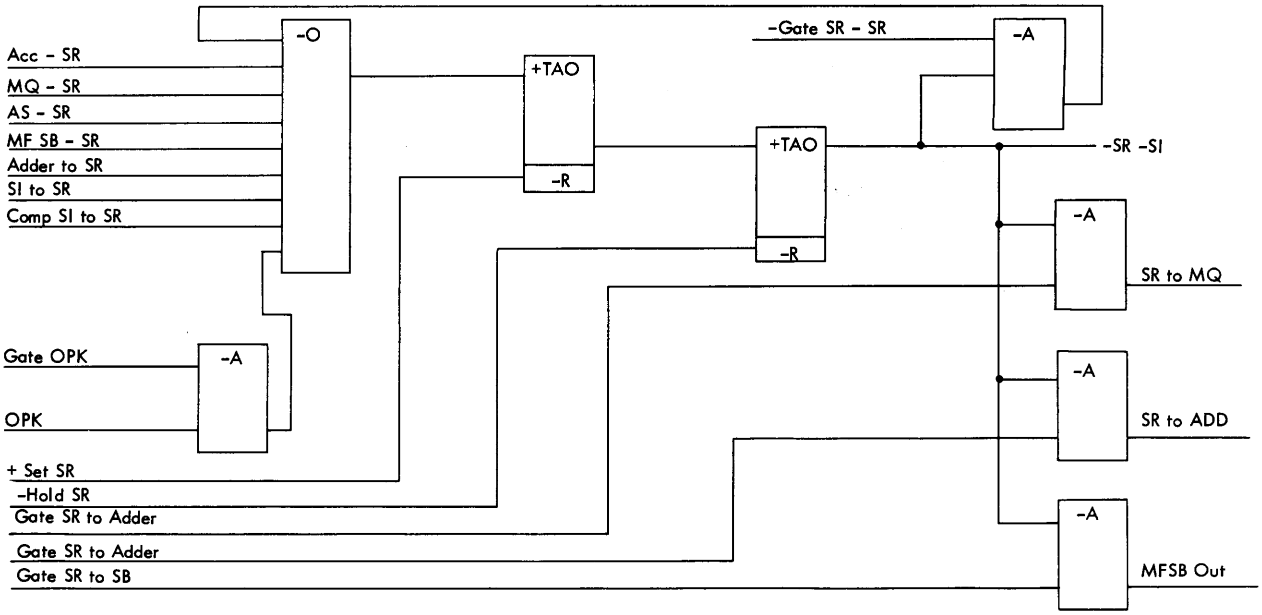


FIGURE 7.1-7. STORAGE REGISTER

is pressed, the program halts again. At this time, the number of the storage location is seen in the accumulator and the contents of the storage location are seen in the MQ. Each time the start key is pressed, the number and contents of the next sequential storage location are displayed. If it is desired to display storage at a new location, press the reset key, place the new address in the operator's panel keys, and start.

Instructions to be punched in the card are:

	Op	Decr	Tag	Adr	
0	IOCD	12		00000	Read 12 words starting at 0.
1	HPR				Halt to set keys.
2					
3	NOP*				Location of this instruction becomes 0000.
4	NOP*				
5	ENK				Keys to MQ.
6	XCA				MQ to Acc.
7	STA			00005	Address of Acc to 00005.
10	LDQ			00000	Load MQ from address in Acc.
11	HTR			00007	MQ contains word whose address is in Acc.
12	ADD			00011	Step address in Acc by one.
13	TRA			00004	Repeat.
14	HTR			00001	

* These instructions are used so this program will not be destroyed even if a diagnostic program using a trapping mode is later run on the machine.

Store Address

The following program fills each memory location with its corresponding address. The program itself only takes 12 positions in upper memory. It is designed so that if sense switch 1 is left up, addresses are stored in all memory locations from 00000 to 77773 before the program destroys itself. If sense switch 1 is placed down before the card is loaded, the program stops to allow the operator to enter an address on the keys. When the start key is pressed, the machine stores addresses in sequential locations, starting at the address placed in the operator's panel keys.

Instructions to be punched in the card are:

	Op	Decr	Tag	Adr	
0	IOCD	14		77764	
1	TCOA			00001	
2	TRA			77764	
3	PSE			00161	Location of this instruction becomes 77764.
4	TRA			77772	
5	HPR				Halt to enter address on keys.
6	ENK				
7	XCA				
10	TRA			77773	
11	CLM				

```
      Op      Decr    Tag     Adr
12    STA                     77774
13    STO                     00000
14    ADD                     77777
15    TRA                     77773
16    HTR                     00001
```

Search for Location of a Word

This program looks through storage until it finds the location of any 36-bit word entered on the operator's panel keys. When the memory location of the word is found, the program halts and the location may be read from the address portion of the MQ. If the start key is pushed again, the program continues to search sequential memory locations. The complete contents of storage, starting from location 20, are searched.

```
      Op      Decr    Tag     Adr
 0    IOCD    15              00003
 1    HPR                                 Halt to set keys.
 2    ENK
 3    STQ                     00017
 4    CLA                     00020
 5    CAS                     00017
 6    TRA                     00010
 7    TRA                     00014
10    CLA                     00004
11    ADD                     00016
12    STA                     00004
13    TRA                     00004
14    LDQ                     00004
15    HTR                     00010       Read location from address part of MQ
16    HTR                     00001
17    HTR                     00000
```

Search for Location of an Operation Code

This program finds the location of any operation code entered on keys S, 1-11. Keys corresponding to bit positions 12-35 must be left up.

```
      Op      Decr    Tag     Adr
 0    IOCD    16              00003
 1    HPR                     00001       Halt to set keys.
 2    ENK
 3    STQ                     00020
 4    CLA                     00021
 5    ARS                     00030
 6    ALS                     00030
 7    CAS                     00020
10    TRA                     00012
11    TRA                     00016
12    CLA                     00004
13    ADD                     00001
14    STA                     00004
```

	Op	Decr	Tag	Adr	
15	TRA			00004	
16	LDQ			00004	
17	HTR			00012	Read location from address part of MQ
20	HTR			00000	

This program may be easily modified to allow it to search memory for a specific address rather than for an operation code. The only change necessary is to enter instruction ALS 00025 in location 5 and instruction ARS 00025 in location 6.

7.2.02 Voltage

Proper sequence for bringing up power is:

1. Press power on reset key on 7618.
2. Reset MG CB1 on 7608.
3. Press power on switch on 7618 or 7151.

Note: Failure to follow this sequence may cause a blown fuse in the 48-volt supply. High resistant fuses or bad fuse clips may cause a voltage drop across the fuse large enough to cause intermittent system trouble. A voltage drop of 0.1 volt across a fuse indicates a defective fuse, fuse holder, or connection.

It is possible to have a power supply circuit breaker in an SMS frame partially open without mechanically tripping the breaker. This may cause one phase to be lost to the SMS frame power supply. A quick check for this condition is to observe the 400-cycle volt meter on the 7618. This condition (loss of one phase in some power supply) will usually be indicated by a larger spread in the voltage reading of the three phases. The normal readings should be recorded for future reference. This procedure is not 100 per cent effective, but will detect a large percentage of single-phased power supplies.

When installing engineering changes, a periodic check of resistance between -6 and ground will catch wiring errors, where the shield is tied to the wrong pin. Although normal resistance is low when using a low scale on the ohmmeter, a change of resistance will be indicated if an error is made.

Blower Motors

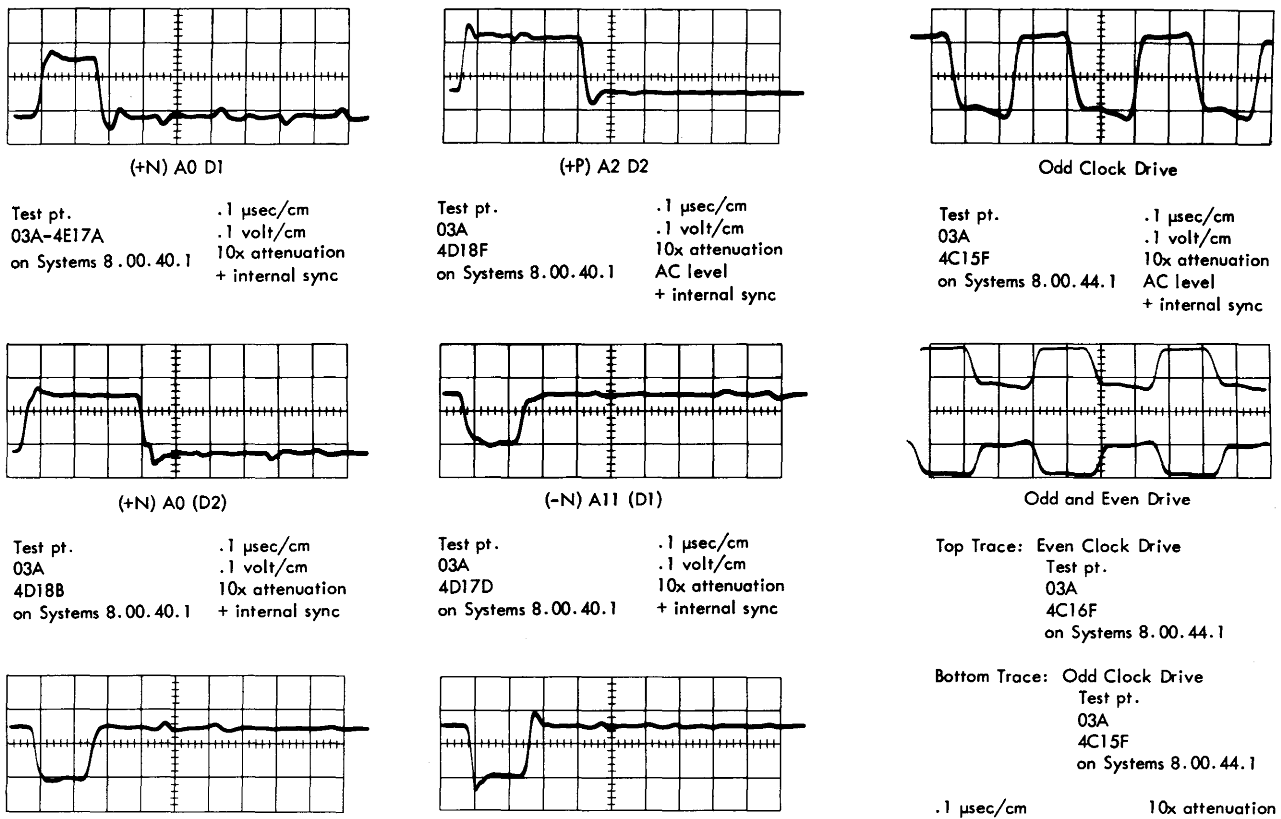
All blower motors, P/N 598593, in the 7090 system are 1/30 horsepower, with exception of memory, which uses a 1/8 horsepower motor, P/N 598594. These blower motors all rotate clockwise when viewed from the hinged end of the gate.

7.2.03 Adjustment of C Pulse Set (Figure 7.2-1)

General operation of a shift cell is described in Section 3. The setting and timing of the C pulse set line is very important.

1. Make sure that the oscilloscope being used has probe leads of the same length.

2. Synchronize the scope on +N pulse at 02A2C15F on Systems 02.05.11.1. This pulse is an A1D1 pulse. Check the following pulses; they should be 182 + 15 - 45 nanoseconds in duration, and should repeat themselves every 2.18 microseconds.



CLOCK

(+N) A0 D1

Test pt.
03A-4E17A
on Systems 8 . 00. 40. 1

. 1 μsec/cm
. 1 volt/cm
10x attenuation
+ internal sync

(+P) A2 D2

Test pt.
03A
4D18F
on Systems 8.00.40.1

. 1 μsec/cm
. 1 volt/cm
10x attenuation
AC level
+ internal sync

(+N) A0 (D2)

Test pt.
03A
4D18B
on Systems 8.00.40.1

. 1 μsec/cm
. 1 volt/cm
10x attenuation
+ internal sync

(-N) A11 (D1)

Test pt.
03A
4D17D
on Systems 8.00.40.1

. 1 μsec/cm
. 1 volt/cm
10x attenuation
+ internal sync

(-N) A1 D1

Test pt.
03A
4F18G
on Systems 8.00.40.1

. 1 μsec/cm
. 1 volt/cm
10x attenuation
AC level
+ internal sync

Clock Trigger 1

Test pt.
03A
4E17B
on Systems 8.00.40.1

. 1 μsec/cm
. 1 volt/cm
10x attenuation
+ internal sync

OSCILLATOR

Odd Clock Drive

Test pt.
03A
4C15F
on Systems 8.00.44.1

. 1 μsec/cm
. 1 volt/cm
10x attenuation
AC level
+ internal sync

Odd and Even Drive

Top Trace: Even Clock Drive
Test pt.
03A
4C16F
on Systems 8.00.44.1

Bottom Trace: Odd Clock Drive
Test pt.
03A
4C15F
on Systems 8.00.44.1

. 1 μsec/cm
. 1 volt/cm

10x attenuation
+ internal sync

FIGURE 7.2-1. MULTIPLEXOR PULSE WAVE FORMS

+N	A2D1	02A2C13B
+N	A3D1	02A2D20C
+P	A4D1	02A3F13B

The time between the A2D1 rising through ground potential, and the A3D1 pulse rising through ground potential, must be 182 (-0, +40) nanoseconds. This is also true of A3D1 and A4D1. The time between A2D1 and A4D1 must be 356 (-0, +40) nanoseconds. The distance between the leading edges of any two D1 pulses at the point of distribution to logic blocks should be within 50 nanoseconds of multiples of 182 nanoseconds.

3. After checking the clock pulses, sync on I time (02A2D03D) with a continuous enter instruction of LDQ (+0560). Check the following MQ test points, with the C pulse set being wholly contained in the gate for each set of points. This is accomplished by varying the delay line located in 03B3. Remember that the pulses are negative going pulses.

CP Set	Gate	Systems
02A2H18C	02A2H18E	02.05.04.1
02A2H18E	02A2H18F	02.05.05.1
02A2H25C	02A2H25D	02.05.04.1
02A2H25E	02A2H25F	02.05.04.1

4. While still synchronized on I time, change the instruction (still under continuous enter instruction) from LDQ to CLA (+0500). Check the following points that the C pulse set is wholly contained within the gate for each point given. Vary the same delay line used for the MQ positions.

02A2J18C	02A2J18D	02.05.02.1
02A2J18E	02A2J18F	02.05.02.1
A2A2J23C	02A2J23D	02.05.02.1
02A2J23E	02A2J23F	02.05.02.1

The setting of the delay line should meet conditions set in both 3 and 4, the CP set contained within all B gates. This will adjust the set and hold pulses for the shift cells.

7.3.00 OPERATOR'S PANEL

7.3.01 CONSOLE INDICATORS

Indicators	Systems	Driver Indicator - Systems
ACC Indicators	00.20.30.0	2.03.00.1 to 2.03.37.1
ACC Overflow	00.20.14.0	02.10.36.1
Address Reg (3-17)	00.30.60.0	3.06.00.1to 3.06.04.1
AND	00.20.19.0	2.09.46.1
Automatic	00.20.41.0	4.20.13.1
B Time	00.80.00.0	8.00.21.1
CAQ	00.20.19.0	2.09.49.1
Channel Selected (A-H)	00.60.00.0	6.00.00.1 to 6.00.01.1
Channel Tape Check (A-H)	00.60.01.0	60.32.02.2
Central Word Trap (A-H)	00.60.03.0	60.38.03.1
Divide Check	00.20.41.1	2.10.53.1
E Time	00.80.00.0	8.00.19.1

Indicators	Systems	Driver Indicator - Systems
End OP	00.20.19.0	8.00.09.1
FP	00.20.19.0	2.10.29.1
I Time	00.80.00.0	8.00.18.0
I-O Check	00.20.41.0	2.10.53.1
Index Reg A-B-C	00.20.70.0	2.70.01.1 to 2.07.15.1
L Time	00.80.00.0	8.00.20.1
Master Stop	00.20.19.0	4.20.11.1
MQ Indicators	00.20.40.0	2.04.00.1 to 2.04.35.1
MQ Overflow	00.20.14.0	02.10.51.1
Multiple Time Error	00.80.00.0	8.00.17.1
Program Counter (3-17)	00.30.40.0	3.05.00.1 to 3.05.07.1
Program Register (S, 1-9)	00.30.50.0	3.04.00.1 to 3.04.03.1
Program Counter (3-17)	00.30.40.0	3.05.00.1 to 3.05.07.1
Q Carry	00.20.19.0	2.10.36.1
R-W Select	00.20.14.0	6.00.02.1
Ready	00.20.41.0	4.20.13.1
Sense Lights (1,2,3,4)	00.20.90.0	2.09.60.1
Shift Counter (10-17)	00.30.40.0	3.04.14.1 to 3.04.17.1
Simulate	00.20.14.0	2.10.70.1
SR Indicators	00.20.11.0	2.01.00.1 to 2.01.35.1
T2	00.20.19.0	2.10.38.1
Tally Counter (1-5)	00.20.17.0	2.10.21.1 to 2.10.22.1
Tape Check Trap (A-H)	00.60.02.0	60.38.03.1
Trap	00.20.14.0	2.10.53.1
Trap Control Indicators	00.60.02.0	2.10.56.1
X Carry (AD-3)	00.20.19.0	2.12.76.1
9 Carry	00.20.19.0	2.10.37.1
9 Overflow	02.20.19.0	2.10.39.1

7.3.02 Indicator Lights

Indicator lights are small incandescent lamps rated at 10 volts, 15 ma. Power is brought in through a two-pin base. One pin is connected to the yellow input signal cable, while the other is returned to -12 volts (violet wire).

Removal and Replacement

Some machines use special clips to hold the indicator lights; others use regular sockets. Open the back panel to make sure of which you have. Regular sockets present no problem; merely pull out the old light and push in a new one to replace it. If special clips are used, it may be necessary to remove the voltage leads before taking out the light. If this is the case, flip the console DC-off switch to knock down power before removing the leads. Then use a plier, wrench, or special tool, if it is available, to loosen the clip and remove the light. These indicator lights are somewhat delicate, so use care when replacing them.

Refer to Section 7.3.01, Service Aids, for a list of indicator lights and corresponding Systems pages.

7.3.03 Unitized Assembly Lights and Keys

Unitized assemblies use telephone-type lights. Some operate on 48 volts and some on 6 volts. Be sure to use the correct replacement for each bulb.

Removal and Replacement

To remove the plastic cover over the light or key assembly, press in on one side of the cover and work it out of its holder. The light bulb is removed by grasping it on the end and pulling straight out. To replace, reverse the procedure.

7.3.04 Switches and Keys

To gain access to the switches and keys on the operator's panel, the panel must be tilted up. To do this, three latches must be released. Release the latches by inserting a screw driver in each latch hole, one after the other, and pressing in. The latch holes are located across the front, underneath the shelf.

Adjustment

It may become necessary to adjust the switches so that the reset motor will return all of them to the OFF position at the same time. The switches can be adjusted up or down and in or out. Usually best results will be obtained if the bottom nut is lowered as close to the base of the switch as is possible. In this way the switch lever is brought as deeply as possible into the plastic rocker.

Removal and Replacement

To remove a switch, it may be necessary to loosen the bar upon which all the switches are mounted. This bar is held by two screws at either end. Removing the bar will not affect the plastic rockers, for they are held by a single shaft which is independent of the switches.

7.3.05 Plastic Rockers

Plastic rockers are used to actuate toggle switches mounted underneath the panel. They may require replacement if they become chipped or broken.

Removal and Replacement

Removal of one or more plastic rockers requires the removal of the assembly upon which all the toggle switches are mounted. Two nuts at either end are removed to take off the assembly. The shaft on which the rockers are strung can then be pulled out and any or all of the rockers can be taken off. New rockers may then be slipped on the shaft and the shaft put back in its proper place on the panel.

7.3.06 Reset Motor

There are two 115-volt reset motors in the console, one whose shaft rotates 60 rpm clockwise, and the other 60 rpm counterclockwise. A manufacturer's part number is stamped on each motor. The number ending in the letter J identifies the right-hand motor. The left-hand motor is stamped with a number ending in the letter K.

Service Checks

If the motor will not operate, check the plug connection and also check for a blown fuse. The fuse is located under the right-hand motor mounting bracket.

Adjustment

A slight adjustment of the relative' position of the motors can be made by loosening the mounting screws, moving the motor, and retightening the screws.

Removal and Replacement

The right-hand motor may be removed by taking out the mounting screws and mounting bracket, and then slipping the shaft out of the rubber coupling sleeve after the set screw is loosened.

Removing and replacing the left-hand motor is somewhat more involved. There is little or no room to move the motor to the left, out of the coupling sleeve; therefore, the coupling sleeve must be moved to the right, away from the motor. To do this, the right-hand motor must be taken out, together with the reset bar and its three mounting brackets. Use the following step-by-step procedure:

1. Loosen all set screws in the coupling sleeves on both motors.
2. Take out the screws holding the right-hand motor mounting bracket to the next bracket.
3. Take out the screws holding the right-hand motor to the mounting bracket.
4. Unplug and remove the motor. Remove the right-hand coupling sleeve from the reset bar.
5. Take off the reset bar's three mounting brackets and remove the entire assembly. Take the coupling sleeve off the left-hand reset motor shaft.
6. Now the screws securing the left motor to its mounting bracket and the bracket to the next bracket can be removed.
7. Unplug and lift out the motor.

To replace the motor, reverse the procedure. Before closing the panel, be sure the reset bar is positioned so that its blade faces the operator and points downward.

7.4.00 CE PANEL

7.4.01 Indicator Lights

The indicator lights used in the CE panel are identical to those used in the operator 's panel. For information concerning these lights, consult Section 7.3.02.

7.4.02 Switches and Receptacles

Switches used on the CE panel are double-throw toggle switches. In addition to these, there are two Jones plug receptacles into which external keys on cables can be connected. If the keys are not connected, pins 3 and 4 of the receptacle must be connected together. This is generally done by means of a special shorting plug.

Removal and Replacement

Access to the switches and receptacles is obtained by opening the rear panel. The Jones plug receptacles are removed by taking off the back retaining plate which is secured by three nuts. Rubber washers are used to hold the plug retaining tabs in place.

For safety reasons, if the console DC-off switch is to be removed, the wall switch must be opened or the emergency OFF switch must be pulled first before the wires are disconnected.

7.5.00 MARGINAL CHECK PANEL

The marginal check panel contains marginal check (MC) switches, motor Variac* control switches, and two voltmeters. Normally the panel is covered. The cover is opened by tripping a lever located on the right side of the Console, under the shelf. Provision for marginal checking the Console is placed in the MC panel. However, this feature will not be used unless transistor cards are built into the unit.

7.5.01 MC Switches

Service Check

If an MC switch fails to make contact, voltages will not be varied in the frame or gate chosen. However, the marginal check voltmeter will indicate just as if everything were operating correctly. Therefore, if trouble is suspected with any of the switches, check the voltages on the frame or gate in question with a voltmeter to see if they actually change as they should. Sometimes one can determine by listening whether the marginal check relays in the power supply have been picked when the MC switches have closed.

Adjustment

No adjustment is provided on these switches. Usually, if a switch is bad, the entire assembly must be replaced.

Removal and Replacement

In order to remove the MC switches, the front curved panel of the console must be removed. To do this, first remove the front base panel by pulling out. This exposes the bottom row of screws on the curved panel. Lifting up the entry key panel exposes the top row of screws. Take out all these screws and remove the panel. Now look underneath the marginal check panel. There is a protective shield located under the panel and beneath it are two MC switch assemblies mounted by a screw on either end to mounting pedestals. Once these screws are removed, the entire assembly can be dropped. Throw the console DC off before starting.

7.5.02 MC Meters

These two meters monitor the variable voltages used for biasing. The MC meter is an ac voltmeter and measures only the amount a voltage is being biased if the MC relay for the module has been picked. If there is any doubt as to the accuracy of a bias limit, the voltage should be checked at the actual gate being biased.

*Trademark of General Radio Company

7.6.00 TAILGATE

The tailgate area contains all the signal and power connectors that enter and leave the console. It is located in gate H of the console frame. See Figure 7.6-1.

7.6.01 Signal Connectors

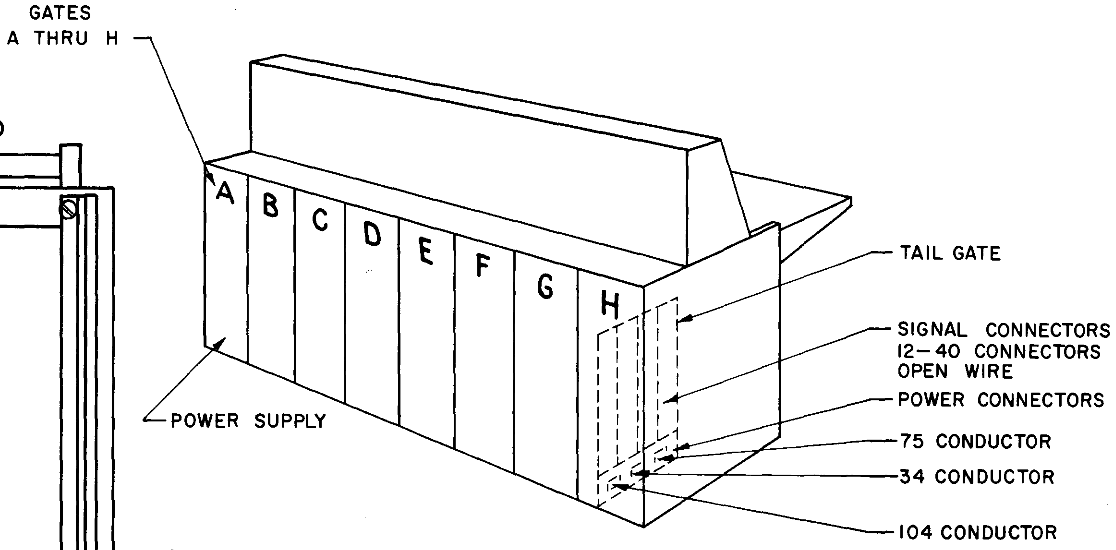
The tailgate contains twelve signal connectors, sometimes called "biscuits," each of which holds 40 conductors.

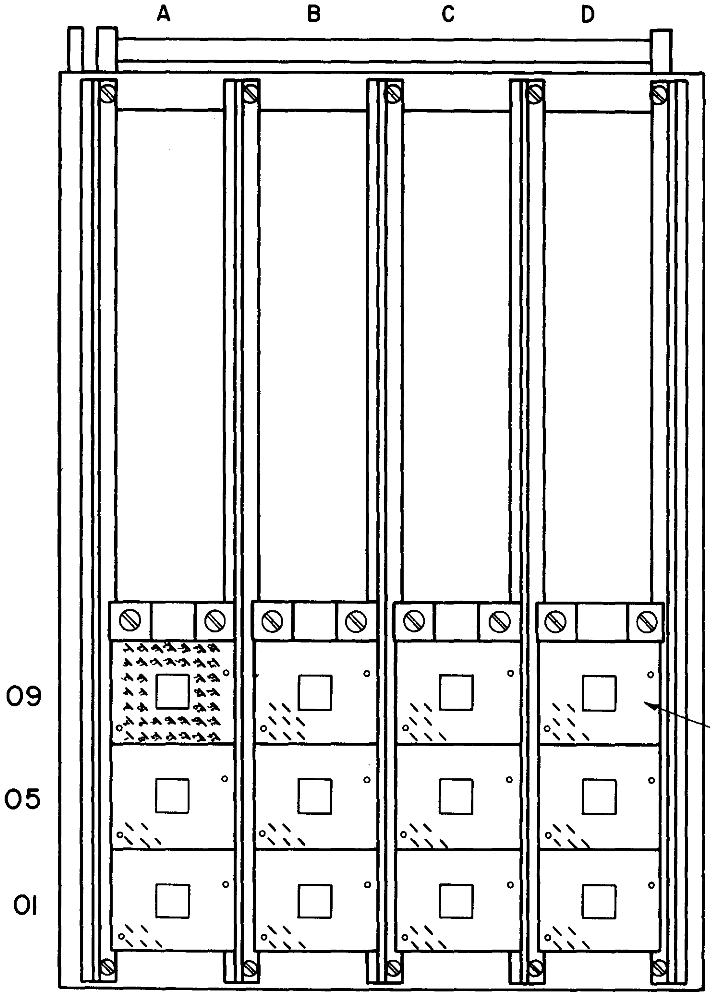
7.6.02 Power Connectors

Three Winchester male power connectors are mounted at the base of the tailgate assembly, at an angle to the vertical signal connector portion. They consist of a 104-conductor, a 34-conductor, and a 75-conductor connection. The 104-conductor cable contains the marginal checking control lines, the 34-conductor cable contains the 400-cycle lines from the power distribution frame (PDF), and the 75-conductor cable contains the 60-cycle power and other miscellaneous control lines to and from the PDF.

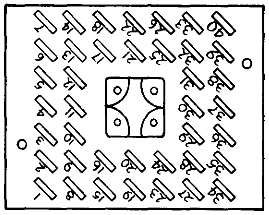
Removal and Replacement

Be sure the marginal check voltages are all at normal before removing the 104-conductor cable. The three-power connectors can be removed from the console in any order. When replacing cables, hold the connector with its female lug on the left side as it faces the tailgate.





TAIL GATE (FRONT VIEW)



ENLARGED VIEW OF BISCUIT

FRAME	GATE	ROW	COLUMN	PIN
08	H	09	D	1—40

EXAMPLE

CABLE CONNECTIONS			
CONSOLE TO	M F 1	M F 2	MULTI-PLEXOR
08H01A	01E21F		
08H05A	01F37F		
08H09A	01E21C		
08H01B	01E21D		
08H05B	01F45C		
08H09B	01F37G		
08H01C	01E21G		
08H05C	01E21E		
08H09C			03F37G
08H01D		02E25C	
08H05D		02E25D	
08H09D		02F33G	

FIGURE 7.6-1. CONSOLE GATES AND TAILGATE

Appendix A

Alphabetic Listing and Index of 7090 Instructions

OPERATION CODE		INSTRUCTION	INDEXABLE	INDIRECTLY ADDRESSABLE	PAGE
ALPHA	OCTAL				
ACL	0361	Add and Carry Logical Word	X	X	60
ADD	0400	Add	X	X	58
ADM	0401	Add Magnitude	X	X	60
ALS	0767	Accumulator Left Shift	X		52
ANA	—0320	AND to Accumulator	X	X	128
ANS	0320	AND to Storage	X	X	128
ARS	0771	Accumulator Right Shift	X		55
AXC	—0774	Address to Index Complemented			126
AXT	0774	Address to Index True			126
BSF	—0764	Backspace File	X		*
BSR	0764	Backspace Record	X		*
BTT	0760..xxxx	Beginning of Tape Test	X		107
CAL	—0500	Clear and Add Logical Word	X	X	58
CAQ	—0114	Convert by Addition from MQ			137
CAS	0340	Compare AC with storage	X	X	99
CHS	0760..0002	Change Sign	X		111
CLA	0500	Clear and Add	X	X	55
CLM	0760..0000	Clear Magnitude	X		68
CLS	0502	Clear and Subtract	X	X	55
COM	0760..0006	Complement Magnitude	X		68
CRQ	—0154	Convert by Replacement from MQ			135
CVR	0114	Convert by Replacement from AC			133
DCT	0760..0012	Divide Check Test	X		104
DVH	0220	Divide or Halt	X	X	63
DVP	0221	Divide or Proceed	X	X	67
ECTM	—0760..0006	Enter Copy Trap Mode	X		*
EFTM	—0760..0002	Enter Floating Trap Mode	X		*
ENB	0564	Enable from Y	X	X	*
ENK	0760..0004	Enter Keys	X	X	52
ERA	0322	Exclusive OR to Accumulator	X	X	128
ESNT	—0021	Enter Storage Null. and Transfer	X	X	*
ESTM	—0760..0005	Enter Select Trap Mode	X		*
ETM	0760..0007	Enter Trapping Mode	X		97
ETT	—0760..xxxx	End of Tape Test	X		107
FAD	0300	Floating Add	X	X	71
FAM	0304	Floating Add Magnitude	X	X	79
FDH	0240	Floating Divide or Halt	X	X	85
FDP	0241	Floating Divide or Proceed	X	X	91
FMP	0260	Floating Multiply	X	X	79
FRN	0760..0011	Floating Round	X		85
FSB	0302	Floating Subtract	X	X	79
FSM	0306	Floating Subtract Magnitude	X	X	79
HPR	0420	Halt and Proceed			107
HTR	0000	Halt and Transfer	X	X	107
IIA	0041	Invert Indicators from AC			116
IIL	—0051	Invert Indicators of Left Half			114
IIR	0051	Invert Indicators of Right Half			114
IIS	0440	Invert Indicators from Storage	X	X	111
IOT	0760..0005	Input-Output Check Test	X	X	104
LAC	0535	Load Complement of Address in Index			126
LAS	—0340	Logical Compare Accumulator with Storage	X	X	102
LBT	0760..0001	Low- Order Bit Test	X		99
LCHA	0544	Load Channel A	X	X	*
LCHB	—0544	Load Channel B	X	X	*
LCHC	0545	Load Channel C	X	X	*
LCHD	—0545	Load Channel D	X	X	*
LCHE	0546	Load Channel E	X	X	*
LCHF	—0546	Load Channel F	X	X	*
LCHG	0547	Load Channel G	X	X	*
LCHH	—0547	Load Channel H	X	X	*
LDC	—0535	Load Complement of Decrement in XR			126
LDI	0441	Load Indicators	X	X	111
LDQ	0560	Load MQ	X	X	50
LFT	—0054	Left Half Indicators, Off Test			119
LFTM	—0760..0004	Leave Floating Trap Mode	X		*
LGL	—0763	Logical Left Shift	X		52
LGR	—0765	Logical Right Shift	X		55
LLS	0763	Long Left Shift	X		52
LNT	—0056	Left Half Indicators, On Test			119
LRS	0765	Long Right Shift	X		55
LSNM	—0760..0010	Leave Storage Nullification Mode	X		*
LTM	—0760..0007	Leave Trapping Mode	X		97
LXA	0534	Load Index from Address			126
LXD	—0534	Load Index from Decrement			128
MPR	—0200	Multiply and Round	X	X	63
MPY	0200	Multiply	X	X	60
MSE	—0760	Minus Sense	X		104
NOP	0761	No Operation			107
NZT	—0520	Storage Not-Zero Test	X	X	99
OAI	0043	OR Accumulator to Indicators			116
OFT	0444	Off Test for Indicators	X	X	119
ONT	0446	On Test for Indicators	X	X	116
ORA	—0501	OR to Accumulator	X	X	128
ORS	—0602	OR to Storage	X	X	128
OSI	0442	OR Storage to Indicators	X	X	111
PAC	0737	Place Complement of Address in XR			123
PAI	0044	Place Accumulator in Indicators			114
PAX	0734	Place Address in XR			123
PBT	—0760..0001	P-bit Test	X		99
PDC	—0737	Place Complement of Decrement in XR			123
PDX	—0734	Place Decrement in Index			123
PIA	—0046	Place Indicator in Accumulator			114
PSE	0760	Plus Sense	X		102
PXA	0754	Place Index in Address			123
PXD	—0754	Place Index in Decrement			126
RCHA	0540	Reset and Load Channel A	X	X	*
RCHB	—0540	Reset and Load Channel B	X	X	*
RCHC	0541	Reset and Load Channel C	X	X	*
RCHD	—0541	Reset and Load Channel D	X	X	*
RCHE	0542	Reset and Load Channel E	X	X	*
RCHF	—0542	Reset and Load Channel F	X	X	*
RCHG	0543	Reset and Load Channel G	X	X	*
RCHH	—0543	Reset and Load Channel H	X	X	*
RCT	0760..0014	Restore Channel Traps	X		*
RDCA	0760..1352	Reset Data Channel A			*
RDCB	0760..2352	Reset Data Channel B			*
RDCC	0760..3352	Reset Data Channel C			*
RDCD	0760..4352	Reset Data Channel D			*
RDCE	0760..5352	Reset Data Channel E			*
RDCF	0760..6352	Reset Data Channel F			*
RDCG	0760..7352	Reset Data Channel G			*
RDCH	0760.10352	Reset Data Channel H			*
RDS	0762	Read Select	X		*
REW	0772	Rewind	X		*
RFT	0054	Right Half Indicators, Off Test			119
RIA	—0042	Reset Indicators from Accumulator			116
RIL	—0057	Reset Indicators of Left Half			114
RIR	0057	Reset Indicators of Right Half			114
RIS	0445	Reset Indicators from Storage	X	X	111
RND	0760..0010	Round	X		68
RNT	0056	Right Half Indicators, On Test			119
RQL	—0773	Rotate MQ Left	X		55

* Description not included in text.

OPERATION CODE		INSTRUCTION	INDEXABLE	INDIRECTLY ADDRESSABLE	PAGE
ALPHA	OCTAL				
RUN	—0772	Rewind and Unload	X		*
SBM	—0400	Subtract Magnitude	X	X	60
SCHA	0640	Store Channel A	X	X	*
SCHB	—0640	Store Channel B	X	X	*
SCHC	0641	Store Channel C	X	X	*
SCHD	—0641	Store Channel D	X	X	*
SCHE	0642	Store Channel E	X	X	*
SCHF	—0642	Store Channel F	X	X	*
SCHG	0643	Store Channel G	X	X	*
SCHH	—0643	Store Channel H	X	X	*
SDN	0776	Set Density			*
SIL	—0055	Set Indicator of Left Half			114
SIR	0055	Set Indicator of Right Half			111
SLQ	—0620	Store Left Half MQ	X	X	50
SLW	0602	Store Logical Word	X	X	47
SSM	—0760..0003	Set Sign Minus	X		111
SSP	0760..0003	Set Sign Plus	X		107
STA	0621	Store Address	X	X	50
STD	0622	Store Decrement	X	X	50
STI	0604	Store Indicators	X	X	111
STL	—0625	Store Instruction Location Counter	X	X	50
STO	0601	Store	X	X	47
STP	0630	Store Prefix	X	X	47
STQ	—0600	Store MQ	X	X	47
STR	—1000	Store Location and Trap			97
STT	0625	Store Tag	X	X	50
STZ	0600	Store Zero	X	X	47
SUB	0402	Subtract	X	X	60
SXA	0634	Store Index in Address			126
SXD	—0634	Store Index in Decrement			126
TCNA	—0060	Transfer on DSC A Not in Operation	X	X	94
TCNB	—0061	Transfer on DSC B Not in Operation	X	X	94
TCNC	—0062	Transfer on DSC C Not in Operation	X	X	94
TCND	—0063	Transfer on DSC D Not in Operation	X	X	94
TCNE	—0064	Transfer on DSC E Not in Operation	X	X	94
TCNF	—0065	Transfer on DSC F Not in Operation	X	X	94
TCNG	—0066*	Transfer on DSC G Not in Operation	X	X	94
TCNH	—0067*	Transfer on DSC H Not in Operation	X	X	94
TCOA	0060	Transfer on DSC A in Operation	X	X	94
TCOB	0061	Transfer on DSC B in Operation	X	X	94
TCOC	0062	Transfer on DSC C in Operation	X	X	94
TCOD	0063	Transfer on DSC D in Operation	X	X	94
TCOE	0064	Transfer on DSC E in Operation	X	X	94
TCOF	0065	Transfer on DSC F in Operation	X	X	94
TCOG	0066	Transfer on DSC G in Operation	X	X	94
TCOH	0067	Transfer on DSC H in Operation	X	X	94
TEFA	0030	Transfer on DSC A End of File	X	X	97
TEFB	—0030	Transfer on DSC B End of File	X	X	97
TEFC	0031	Transfer on DSC C End of File	X	X	97
TEFD	—0031	Transfer on DSC D End of File	X	X	97
TEFE	0032	Transfer on DSC E End of File	X	X	97
TEFF	—0032	Transfer on DSC F End of File	X	X	97
TEFG	0033	Transfer on DSC G End of File	X	X	97
TEFH	—0033	Transfer on DSC H End of File	X	X	97
TIF	0046	Transfer if Indicators Off	X	X	116
TIO	0042	Transfer if Indicators On	X	X	116
TIX	2000	Transfer on Index			121
TLQ	0040	Transfer on Low MQ	X	X	94
TMI	—0120	Transfer on Minus	X	X	92
TNO	—0140	Transfer on No Overflow	X	X	92
TNX	—2000	Transfer on No Index			121
TNZ	—0100	Transfer on No Zero	X	X	94
TOV	0140	Transfer on Overflow	X	X	92
TPL	0120	Transfer on Plus	X	X	92
TQO	0161	Transfer on Quotient Overflow	X	X	92
TQP	0162	Transfer on MQ Plus	X	X	92
TRA	0020	Transfer	X	X	92
TRCA	0022	Transfer on DSC A Redundancy Check	X	X	97
TRCB	—0022	Transfer on DSC B Redundancy Check	X	X	97
TRCC	0024	Transfer on DSC C Redundancy Check	X	X	97
TRCD	—0224	Transfer on DSC D Redundancy Check	X	X	97
TRCE	0026	Transfer on DSC E Redundancy Check	X	X	97
TRCF	—0026	Transfer on DSC F Redundancy Check	X	X	97
TRCG	0027	Transfer on DSC G Redundancy Check	X	X	97
TRCH	—0027	Transfer on DSC H Redundancy Check	X	X	97
TSX	0074	Transfer and Set Index			123
TTR	0021	Trap Transfer	X	X	97
TXH	3000	Transfer on Index High			121
TXI	1000	Transfer with XR Incremented			121
TXL	—3000	Transfer on XR Low or Equal			123
TZE	0100	Transfer on Zero	X	X	92
UAM	—0304	Unnormalized Add Magnitude	X	X	79
UFA	—0300	Unnormalized Floating Add	X	X	78
UFM	—0260	Unnormalized Floating Multiply	X	X	85
UFS	—0302	Unnormalized Floating Subtract	X	X	79
USM	—0306	Unnormalized Subtract Magnitude	X	X	79
VDH	0224	Variable Length Divide or Halt	X		67
VDP	0225	Variable Length Divide or Proceed	X		68
VLM	0204	Variable Length Multiply	X		63
WEF	0770	Write End of File	X		*
WRS	0766	Write Select	X		*
XCA	0131	Exchange AC and MQ			50
XCL	—0130	Exchange Logical AC and MQ			50
XEC	0522	Execute	X	X	107
ZET	0520	Storage Zero Test	X	X	99

* Description not included in text.

APPENDIX B

SYSTEMS REFERENCES

Operation Code	Systems	Operation Code	Systems	Operation Code	Systems	Operation Code	Systems
ACL	02.10.02.1	LCHD	06.00.06.1	RDCF	60.65.04.1	TCOD	06.01.01.1
ADD	02.09.91.1	LCHE	06.00.06.1	RDCG	60.65.04.1	TCOE	06.01.02.1
ADM	02.09.95.1	LCHF	06.00.06.1	RDCH	60.65.04.1	TCOF	06.01.02.1
ALS	02.09.70.1	LCHG	06.00.07.1	RDS	06.01.12.1	TCOG	06.01.03.1
ANA	02.09.46.1	LCHH	06.00.07.1	REW	06.01.12.2	TCOH	06.01.03.1
ANS	02.09.46.1	LDC	02.12.16.1	RFT	02.12.67.1	TEFA	06.01.00.1
ARS	02.09.70.1	LDI	02.12.68.1	RIA	02.12.67.1	TEFB	06.01.00.1
AXC	02.15.62.1	LDQ	02.12.40.1	RIL	02.12.62.1	TEFC	06.01.01.1
AXT	02.15.62.1	LFT	02.12.67.1	RIR	02.12.62.1	TEFD	06.01.01.1
BSF	06.01.12.1	LFTM	02.10.71.1	RIS	02.12.62.1	TEFE	06.01.02.1
BSR	06.01.12.1	LGL	02.09.70.1	RND	02.09.57.1	TEFF	06.01.02.1
BTT	02.10.80.1	LGR	02.09.70.1	RNT	02.12.58.1	TEFG	06.01.03.1
CAL	02.10.02.1	LLS	02.09.70.1	RQL	02.09.70.1	TEFH	06.01.03.1
CAQ	02.09.49.1	LNT	02.12.68.1	RUN		TIF	02.10.57.1
CAS	02.09.43.1	LRS	02.09.70.1	SBM	02.09.95.1	TIO	02.10.57.1
CHS	02.09.57.1	LSNM	02.10.70.1	SCHA	06.00.05.1	TIX	02.15.62.1
CLA	02.10.02.1	LTM	02.10.53.1	SCHB	06.00.05.1	TLQ	02.09.30.1
CLM	02.09.57.1	LXA	02.12.70.1	SCHC	06.00.06.1	TMI	02.10.08.1
CLS	02.10.02.1	LXD	02.12.16.1	SCHD	06.00.06.1	TNO	02.10.08.1
COM	02.09.57.1	MPR	02.09.53.1	SCHE	06.00.06.1	TNX	02.15.62.1
CRQ	02.09.49.1	MPY	02.09.53.1	SCHF	06.00.06.1	TNZ	02.09.30.1
CVR	02.09.49.1	MSE	02.09.58.1	SCHG	06.00.07.1	TOV	02.10.08.1
DCT	02.09.58.1	NOP	08.00.01.1	SCHH	06.00.07.1	TPL	02.10.08.1
DVH	02.09.50.1	NZT	02.11.50.1	SDN	60.50.02.3	TQO	02.10.08.1
DVP	02.09.50.1	OAI	02.12.67.1	SIL	02.12.63.1	TQP	02.10.08.1
ECTM	02.10.70.1	OFT	02.12.68.1	SIR	02.12.63.1	TRA	02.10.08.1
EFTM	02.10.71.1	ONT	02.12.68.1	SLQ	02.09.00.1	TRCA	06.01.00.1
ENB	02.10.61.1	ORA	02.10.02.1	SLW	02.09.00.1	TRCB	06.01.00.1
ENK	04.20.14.1	ORS	02.09.00.1	SSM	02.09.57.1	TRCC	06.01.01.1
ERA	02.09.40.1	OSI	02.12.68.1	SSP	02.09.57.1	TRCD	06.01.01.1
ESNT	02.10.70.1	PAC	02.12.70.1	STA	02.09.01.1	TRCE	06.01.02.1
ESTM	02.10.70.1	PAI	02.12.67.1	STD	02.09.01.1	TRCF	06.01.02.1
ETM	02.10.53.1	PAX	02.12.70.1	STI	02.09.00.1	TRCG	06.01.03.1
ETT	02.10.80.1	PBT	02.09.58.1	STL	02.09.01.1	TRCH	06.01.03.1
FAD	02.10.30.1	PDC	02.12.70.1	STO	02.09.00.1	TSX	02.15.62.1
FAM	02.09.95.1	PDX	02.12.16.1	STP	02.09.01.1	TTR	02.10.08.1
FDH	02.10.40.1	PIA	02.12.67.1	STQ	02.09.00.1	TXH	02.15.62.1
FDP	02.10.40.1	PSE	02.09.58.1	STR	02.09.00.1	TXI	02.15.62.1
FMP	02.10.23.1	PXA	02.12.70.1	STT	02.09.01.1	TXL	02.15.62.1
FRN	02.10.27.1	PXD	02.12.70.1	STZ	02.09.00.1	TZE	02.09.30.1
FSB	02.09.95.1	RCHA	06.00.05.1	SUB	02.09.95.1	UAM	02.09.95.1
FSM	02.09.95.1	RCHB	06.00.05.1	SXA	02.09.01.1	UFA	03.01.10.1
HPR	03.01.06.1	RCHC	06.00.06.1	SXD	02.09.01.1	UFM	02.10.24.1
HTR	03.01.06.1	RCHD	06.00.06.1	TCNA	06.01.00.1	UFS	02.09.95.1
IIA	02.12.67.1	RCHE	06.00.06.1	TCNB	06.01.00.1	USM	02.09.95.1
IIL	02.12.61.1	RCHF	06.00.06.1	TCNC	06.01.01.1	VDH	02.09.50.1
IIR	02.12.61.1	RCHG	06.00.07.1	TCND	06.01.01.1	VDP	02.09.50.1
IIS	02.12.61.1	RCHH	06.00.07.1	TCNE	06.01.02.1	VLM	02.09.53.1
IOT	02.09.58.1	RCT	02.10.61.1	TCNF	06.01.02.1	WEF	06.01.12.2
LAC	02.12.70.1	RDCA	60.65.04.1	TCNG	06.01.03.1	WRS	06.01.12.2
LAS	02.09.43.1	RDCB	60.65.04.1	TCNH	06.01.03.1	XCA	02.10.06.1
LBT	02.09.58.1	RDCC	60.65.04.1	TCOA	06.01.00.1	XCL	02.10.06.1
LCHA	06.00.05.1	RDCD	60.65.04.1	TCOB	06.01.00.1	XEC	03.06.05.1
LCHB	06.00.05.1	RDCE	60.65.04.1	TCOC	06.01.01.1	ZET	02.11.50.1
LCHC	06.00.06.1						

CUT ALONG LINE

COMMENT SHEET

IBM 7090 (7100, 7151, 7606)

CUSTOMER ENGINEERING INSTRUCTION-REFERENCE, FORM 223-6895-1

FROM

NAME ____

OFFICE NO. ____

CHECK ONE OF THE COMMENTS AND EXPLAIN IN THE SPACE PROVIDED

□ SUGGESTED ADDITION (PAGE , TIMING CHART, DRAWING, PROCEDURE, ETC.)

□ SUGGESTED DELETION (PAGE)

□ ERROR (PAGE)

EXPLANATION

NO POSTAGE NECESSARY IF MAILED IN U.S.A.

FOLD ON TWO LINES, STAPLE, AND MAIL

STAPLE STAPLE

FOLD FOLD

FIRST CLASS
PERMIT NO. 81

POUGHKEEPSIE, N. Y.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

POSTAGE WILL BE PAID BY

IBM CORPORATION
P. O. BOX 390
POUGHKEEPSIE, N. Y.

ATTN: CE MANUALS, DEPARTMENT 296

FOLD FOLD

IBM
International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, New York