VOLUME I

Technical Manual
for
DPS-2402
COMPUTER

1 February 1966

WESTINGHOUSE ELECTRIC CORPORATION

SURFACE DIVISION

P. O. Box 1897 Baltimore, Md. 21203

MDE 6195

LIST OF SECTIONS

TABLE OF CONTENTS

SECTION 1 - GENERAL INFORMATION

SECTION 1 - GENERAL INFORMATION

SECTION 2 - INSTALLATION

SECTION 3 - OPERATOR'S SECTION

SECTION 3-OPERATOR'S SECTION (Cont.)

SECTION 4-PRINCIPLES OF OPERATION

SECTION 4-PRINCIPLES OF OPERATION (Cont.)

SECTION 4-PRINCIPLES OF OPERATION (Cont.)

SECTION 4-PRINCIPLES OF OPERATION (Cont.)

SECTION 5-TROUBLESHOOTING

LIST OF ILLUSTRATIONS

SECTION 1 - GENERAL INFORMATION

SECTION 3 - OPERATORS SECTION

SECTION 4 - PRINCIPLES OF OPERATION

SECTION 4 - PRINCIPLES OF OPERATION

SECTION 4 - PRINCIPLES OF OPERATION(Cont.)

SECTION 4- PRINCIPLES OF OPERATION(Cont.)

SECTION 5-TROUBLESHOOTING

I. GENERAL DESCRIPTION

This section contains DPS-2402 Computer specifications, a general block diagram discussion and a detailed description of the repertoire of instructions.

1-1. SPECIFICATIONS

a. General Features. - The following summarizes the technical characteristics of the Westinghouse DPS-2402 Computer:

(1) Type: *General-purpose, stored program, integrated circuit, medium scale, binary, parallel.

i Memory Section:
- *1.0 microsecond access time
- *2.0 microsecond read-write cycle time
- *Parity Check Feature
- *24-bit word length, parallel transfer (plus 1 parity bit)
- *4,096, 8,192 or 16,384 words expandable to 32,768 words.
- *Coincident current mode.

ii Arithmetic Section:
- *Organization: 24 bit, parallel, 2's complement, binary, fixed-point
- *Registers - Two arithmetic (A and Q)
  - One transient (D)
  - One shift count (N)
- *Instruction Execution Time (Including procurement of instruction and operand)
- *Add Time 4.0 μsec
- *Multiply Time 12.0 μsec
- *Divide Time 22.0 μsec

iii Control Section:
- *Word length: 24 bits
- *Single address instructions: one instruction per word.
- *Addressable index registers: 3 (B-boxes)
- *Parallel operation
- *Multiple-level indirect address capability with indexing of primary and indirect addresses.
- *38 basic instructions, or a total of 74 instructions by using modifiers.

iv Input/Output Section:
- *Eight (8) channels (expandable to 16)
- *Any channel may be used in fully buffered, express, or external function mode.
- *Both input and output buffers may be enabled on any channel simultaneously.
- *Buffer mode operation, once initiated proceeds without program intervention.
- *Internal and external interrupts on each channel.
- *Real-time clock, set by program, with interrupt on zero count.

(2) Physical Characteristics:
- *Size: 72 x 28.75 x 25 inches
- *Weight: 750 pounds
- *Power Consumption: 1200 watts 120 volt, single phase, 60 cycle.

(3) Environmental Requirements:
- *Non-volatile core memory with power failure lockup feature which saves the contents of operational registers. The mechanical design of the DPS-2402 was performed using the requirements of the following specifications as a guide.
- *General: MIL-E-16400E
- *Vibration: MIL-STD-167, Type I
- *Shock: MIL-S-901 or MIL-E-5400, paragraph 3.2.21.6
- *RF Interference: MIL-I-16910A
- *Temperature: 0°-50°C Room Ambient
- *Humidity: To 95%

b. Machine Structure. - Figure 1-1-A is a photograph of the Basic Westinghouse DPS-2402 Computer System and Figure 1-1-B is a functional system block diagram. The DPS-2402 Computer is organized into four main units, the arithmetic, control, input-output, and the memory units.

(1) Arithmetic Unit. - The arithmetic unit contains control and numerical registers, a high-speed adder, and logical gating necessary for high-speed execution of the arithmetic and logical commands. The registers required by the unit include two addressable registers (A and Q), one transient register (D), a shift count register (N), and two control registers (AC and AF).

(2) Control Unit. - The control unit interprets commands and directs the cycles of operation which execute the commands. The control unit includes two working registers (I and P), the core memory parity logic, the function code and designator translators, and the sequence generators. The fault and parity interrupts are processed by the control unit.

(3) Input-Output. - The input-output unit consist of a data exchange register (X), a register for updating the buffer control word (Z), input amplifiers, level converters, output drivers, priority gating, buffer control, external function control, and interrupt control.

(4) Memory Unit. - The DPS-2402 Computer contains a magnetic core memory operating in a coincident-current mode for storage. The memory has a capacity of 16,384 words of 24 bits (plus one parity bit) each and requires 2 microseconds for one complete read-write cycle.

c. Symbolism for Registers

A	Accumulator (A-Register)
(A)	Content of the accumulator
B_b	Specified Index Register (location) i. e., B1, B2, or B3.
I	Instruction Register
P	Program Register
Q	Multiplier-Quotient Register
S	Storage Address Register
M	Memory Register
D	Arithmetic Transient Register
N	Shift Count Register - Counter
X	Input-Output Transfer Register
Z	Buffer Control & Real Time Clock Register

d. Register Descriptions

(1) A-Register (24 Bits). - This accumulator, referred to as the A-register, is the principle arithmetic register. Inputs to this register are derived from the adder selection gates which provide parallel addition and shifting capability. With the exception of multiplication, all arithmetic operations call for one operand to be in this register prior to execution of that instruction.

After addition or subtraction, the A-register contains the sum or difference; after a multiplication, the most significant half of the product remains in the accumulator. Additionally, this register contains both the remainder after the execution of a divide and the number on which logical operations are performed.

The contents of the accumulator may be shifted either left or right, closed or open, as described by the shift operations.

(2) Q-Register (24 Bits). - This register serves as the multiplier-quotient register. Prior to multiplication, the register contains the multiplier. During multiplication, the multiplier is shifted right, two positions at a time. The three least-significant bits are examined during each shift to determine if the multiplicand should be applied to the partial product. At the same time, the significant part of the partial product is shifted right into the Q-register. At the completion of the high-speed multiplication process, facilitated by the two-bit look-ahead feature, the least significant half of the product is found in the Q-register.

The Q-register contains the least significant half of the dividend prior to the division process. This register is used to assemble and hold the quotient, and the sign of the quotient is found in the most significant bit position.

Shifting of the Q-register contents is similar to that of the A-register. There are cases in which the A and Q register are shifted as a single 48-bit register.

(3) D-Register (24 Bits). - The D-register is an intermediate register which contains the operand from memory (Y) while the sum, difference, product, or quotient is being formed. This register also contains the B-modifier while it is being added to the address.

A second function of this register is that it serves to transfer data for instruction words to and from the memory unit and all of the arithmetic and control unit registers.

(4) N-Register (6 Bits). - This register is implemented in the form of a counter and is used to control the shifting during multiplication, division, and shift command execution.

To readily facilitate a programmed floating point, a "scale factor" instruction is included

Figure 1-1-A. Basic Westinghouse DPS-2402 Computer

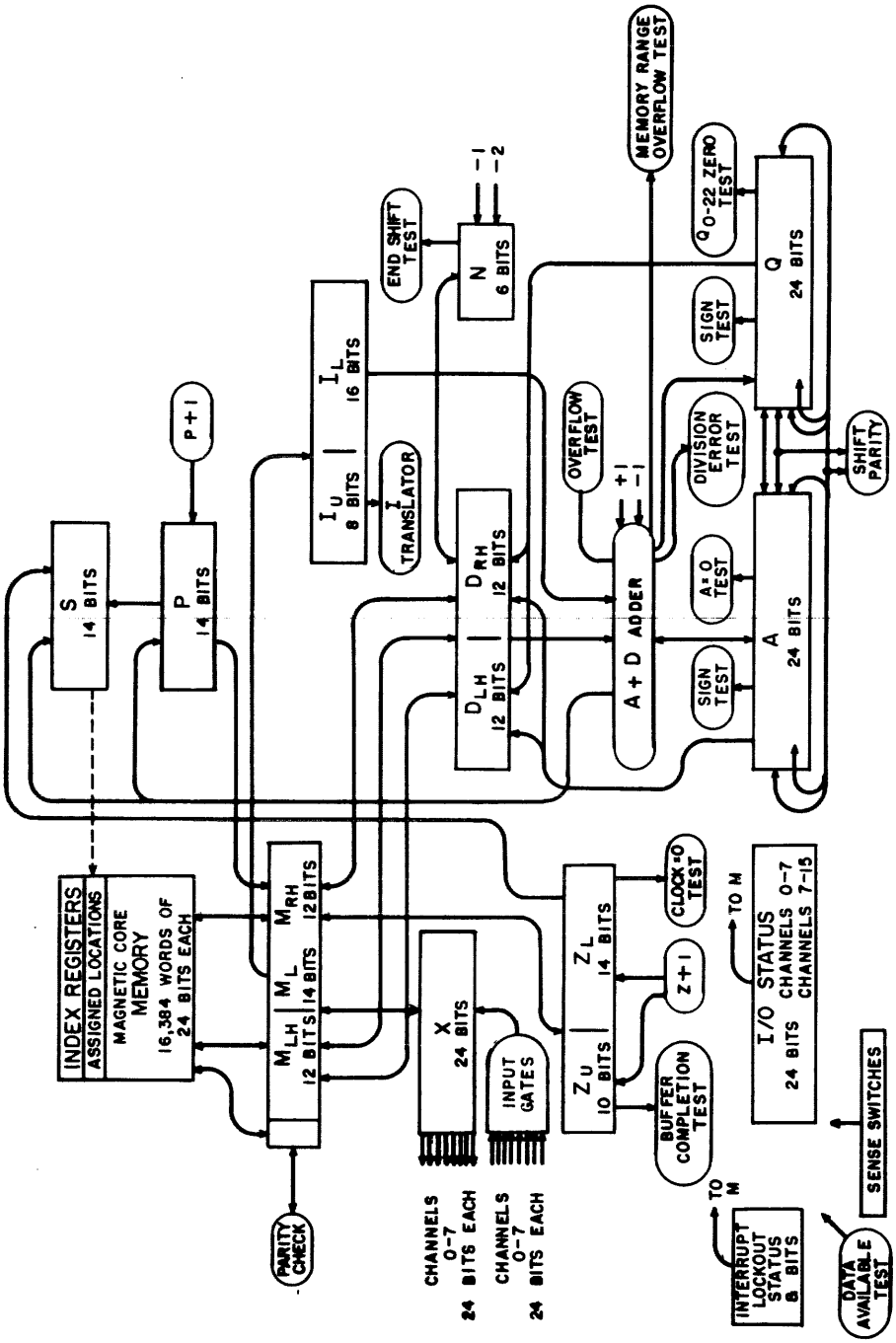


Figure 1-1-B. Westinghouse DPS-2402 Computer Block Diagram

in the repertoire. After execution of this instruction, the N-register contains the number of positions shifted in the scaling operation. The N-register is not addressable; therefore, the scale factor count is stored in location Y.

(5) I-Register (24 Bits). - The I-register stores the current instruction word and is the source for the interpretor. This register acquires the instruction from memory through the M-register. In addition to serving as an input to the function code and designator translators, this register also inputs to the adder circuit to provide modification of the operand address.

(6) P-Register (14 Bits). - The address of the next instruction is stored in the P-register. This register is loaded through the adder from the I-register and is incremented by 1, automatically, as the instructions are sequentially executed. Execution of a skip instruction causes the P-register to be incremented by an additional 1. The parallel loading of this register is accomplished by the execution of a jump instruction. The contents of the P-register may be stored allowing a jump and set return instruction.

(7) S-Register (14 Bits). - The S-register holds a storage address during memory references. The address is received from the input-output or control units at the beginning of a storage access period. The contents of the S-register then are decoded by the memory address selection system.

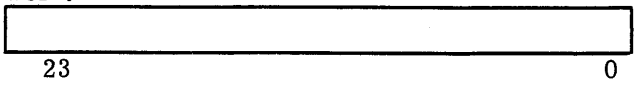
(8) M-Register (25 Bits). - The M-register serves as buffer for storage references. At the beginning of each memory access period, the M-register is cleared. During the read-access period, the contents of the desired memory location are sensed, destroyed, and loaded into the M-register. The contents of M then are written back into memory; the M-register controls the inhibit circuits used to restore the information in the memory location. During the write-access period, the contents of the desired memory location are sensed by the memory sense circuits, but not loaded into the M-register. The M-register is loaded from the input-output or control units; then the contents of M are written into the desired memory location. Twenty-four bits are transmitted between M and D or Z. The 25th bit is a parity check bit.

(9) X-Register (24 Bits). - The X-register is an exchange register which functions to transfer data and commands between the core memory and peripheral equipment.

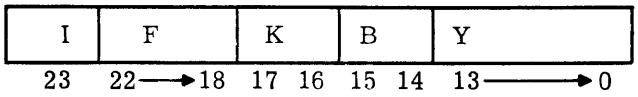
(10) Z-Register (24 Bits). - The Z-register is a 24-bit register which functions to increment a count and an address in buffer mode control and the contents of the real time clock location.

e. Word Formats. - The DPS-2402 has five word formats. One is used frequently and may be regarded as basic. The remaining formats are less common and will be explained later.

The 24 bit positions of the DPS-2402 Computer word are numbered 0 to 23 from right to left. Each instruction is contained in a single 24-bit word.



(1) The Basic Instruction Format. - The basic instruction format has an I, F, K, B, and Y field, as shown below.



The I field is used to specify indirect addressing on certain instructions.

The F field specifies the basic function or operation to be performed by the instruction.

The K field contains a designator which modifies or qualifies the instruction in the F field.

The B field is used to specify a B-index which will modify the address given in the Y field by a fixed amount.

The Y field specifies the address of the operand to which the function in the F-field applies.

(2) B Designator. - Three memory locations; 00001, 00002, and 00003, are used as B index registers. These locations may be addressed normally, as well as being used as B index registers.

Instructions whose function specifies an operation on the index registers are not susceptible to relative addressing. All other instructions with a B field are performed as follows:

If B = 0, no indexing occurs.

If B = 1, the bits 0 to 13 of location 1 are obtained from memory and added to bits 0 to 13 (the Y-field) of the instruction. If the number added to the Y-field is too large, and a bit needs to be carried into bit 14 position, the memory range overflow indicator will lightup. Bits 0 to 13 of the sum are used as the instruction address.

If B = 2 or B = 3 the operation is similar to the case B=1, using locations 2 and 3, respectively. B-Indexing requires one additional memory reference and is performed before indirect addressing.

If an indirect address specifies an index register, one additional memory reference is required, and indexing is affected, before consideration is given the K field option.

(3) Indirect Addressing. - For those instructions which permit indirect addressing (f = 10 through 37), a logical "1" in bit position 23 will specify the indirect option.

Indirect addressing may be explained by the following examples: Unless otherwise noted, all numbers of the examples are octal.

The direct "Enter A" instruction (21000350) will place the contents of memory address 0350 in the A register. The indirect Enter A (61000350) will obtain a location (address) at 0350. Assume the content of 0350 to be 00000150. The computer will place the contents of 150 in the A register.

If the instruction has specified indexing by a b register, this operation would be performed prior to the indirect address cycle. For example if B_1 = 00000005, the indirect Enter A instruction 61040350 would obtain the address at 350 + 5 = 355. Assume the content of address 355 is 00007776. The operand to be entered into the A register would be obtained from the contents of address 7776.

If the contents of address 355 had specified a b register, a second indexing process would have occurred. For example if address 355 would have contained 00047776, the operand would have been obtained from address 7776 + 5 = 10003. Thus the content of address 10003 would be entered into the A register.

(4) Multi-Level Indirect Addressing. - The preceding examples all assumed that bit 23 of the location specified by the y field was a logical "0", the machine will continue performing indirect cycles or index and indirect cycles until a logical "0" in the address bit 23 is found. This logical "0" specifies the effective operand address. For example, consider the execution of the instruction 61041000 where the contents of the following locations are:

B_1 (00001) = 00000001
B_2 (00002) = 00000120
B_3 (00003) = 00000056
(01001) = 46742050
(2126) = 43103520
(3640) = 36010250
(10250) = 25432102

The machine will first index the y field of the instruction by B_1 and will obtain the content of address 01001 which is another indirect (bit 23 set). The machine will obtain the next level location after indexing by B_3, or 2050 + 0056 = 2126; address 2126 also contains an indirect type address (bit 23 set) and calls for indexing by B_2. Therefore, the address 3520 + 0120 = 3640 would be obtained. This is a "direct" type address (bit 23 clear) and no indexing is specified. Therefore, the operand is finally found at address 10250 and the final content of the A register is 25432102 upon completion of the execution. Each indirect operation requires two microseconds.

There is no limit to the number of indirect operations that can be accomplished. As long as the indirect process is in progress, the computer cannot be stopped by depressing the STOP switch and all Input/Output operations are suspended including real time clock updating and interrupt processing.

(5) K Designator. - The K designators specify which bits of the 24 bit operand word will be used. Four options are available. The options apply after the B and I fields have had their effect on the Y address.

For K = 0 the entire 24 bit word located at the Y address is used as an operand.

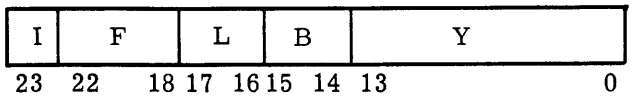
For K = 1 the right half (RH) of the word at Y is used as the operand. When the operand is acquired, bits 0 to 10 of Y become bits 0 to 10 of the operand, and bit 11 of Y is replicated as bits 11 to 23 of the operand. When a store is performed, bits 0 to 11 of the operand become bits 0 to 11 of Y. The left half of the word at Y is not changed.

For K = 2 the left half (LH) of the word at Y is used as the operand. When the operand is acquired, bits 12 to 22 of Y become bits 0 to 10 of the operand and bit 23 of Y is replicated as bits 11 to 23 of the operand. When a store is performed, bits 0 to 11 of the operand become bits 12 to 23 of Y. The right half of the word at Y is not changed.

For K = 3 the 14 low order bits (Y) of the instruction word itself are used as the operand, i. e. the Y field is used as the operand. No operand acquisition from memory is required, since the bits of the Y field become bits 0 to 13 of the operand, and a zero is replicated as bits 14 to 23 of the operand. When a store is performed the Y-field is used as an address and bits 0 to 13 of the operand replace bits 0 to 13 of the word at the memory address indicated by the Y field leaving bits 14 to 23 of the word at Y unchanged.

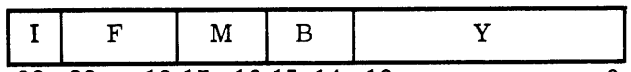
(6) Additional Instruction Formats. - The remaining instruction formats for the DPS-2402 are shown below.

Conditional skip instruction format

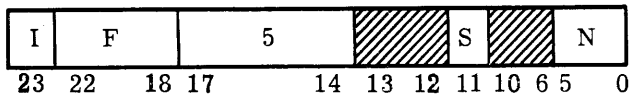


Input/Output instruction format

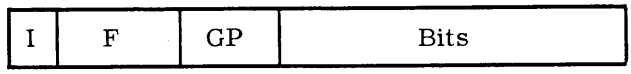
Jump instruction format



Shift instruction format



Sense switch instruction format



(7) L Designator. - Some skip and jump instructions use an L designator in place of the K designator. In this case, the operand always refers to a full word. The F designator specifies what is to be tested, and the L designator specifies the case for the abnormal sequence.

For L = 0 the skip or jump occurs when the sign of the operand is plus, (or zero).

For L = 1 the skip or jump occurs when the sign of the operand is minus.

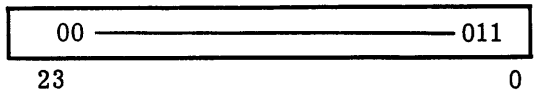
For L = 2 the skip or jump occurs when the operand is zero.

For L = 3 the skip or jump occurs when the operand is non-zero.

Special word formats for shift and input-output are described concurrent with the instructions. Other instructions requiring the I and/or K field for special designations are described concurrent with the function described concurrent with the function description.

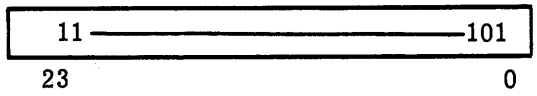
(8) M Designator. - The M designator interpretation of bits 16 and 17 depends on the particular instruction.

(9) Numeric. - Numeric data are stored in 2's complement form. This applies to both full and half words. Full-word positive numbers have a zero in bit position 23, while negative numbers have a 1 in this bit position (e.g., $+3_{10}$ appears in storage as



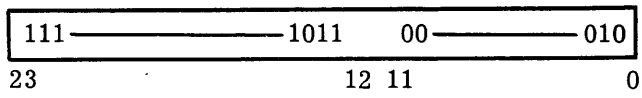
;

-3_{10} appears in storage as



).

Half-word numbers similarly have the sign bit as the left most bit of the appropriate half. For example, with the left half-5 and the right half +2 the word in storage appears as



The range of full-word numbers as integer and fraction is given below. Note that with 2's complement notation, zero is unique as a positive number and a negative sign (1 in bit position 23) with 23 trailing zeros representing the largest negative number.

Integer		Binary	Fractional	
8388607	$2^{23}-1$	011——11	1.0 -2^{-23}	.99999988
		23 0		
1	1	00——01	2^{-23}	.00000012
0	0	00——00	0	.0
-1	-1	11——11	-2^{-23}	-.00000012
-8388608	-2^{23}	10——00	-1.0	-1.0

f. Interrupt Definition and Sources. - The term interrupt refers to the process of causing the computer to execute an instruction out of the programmed sequence. In the interrupt mode, the instruction address is not specified by the P-Register but is instead obtained from a fixed location corresponding to the particular interrupt. The instruction at this fixed interrupt location in most cases would be a Jump and Set Return which stores the point of departure from the program when the interrupt occured and jumps to

the proper subroutine. The interrupt may be ignored by placing a No Operation instruction in the fixed location. It should be noted that when an instruction is acquired from a fixed location, the normal incrementing of the P-Register does not take place.

An interrupt is processed after the completion of the current instruction if it is the highest priority interrupt waiting to be processed. Interrupt priorities are briefly described below.

(1) Fault Interrupt. - If an instruction is executed which has bits 18 to 23 all zero (i. e., a FAULT instruction), and the FAULT REACTION switch is in the "no half" position, the instruction in location 00020 is executed in the interrupt mode.

(2) Input/Output Parity Error Interrupt. - If the computer detects a parity error while reading from memory during the input or output (buffered or express) process and the I/O PARITY REACTION switch is in the "no halt" position, the instruction in location 00021 is executed in the interrupt mode.

(3) Program Parity Error Interrupt. - If the computer detects a parity error while reading instructions or operands from memory and the PROGRAM PARITY ERROR REACTION switch is in the "no halt" position, the instruction in location 00022 is executed in the interrupt mode.

(4) Manual Interrupt. - If the computer is running and the MANUAL INTERRUPT switch on the OPERATION panel is depressed, the instruction in location 00023 is executed in the interrupt mode.

(5) Power Failure Interrupt. - Loss of power is sensed 100 microseconds before voltages drop below safe operating levels. Upon sensing imminent power failure, the instruction in location 00026 is executed in the interrupt mode.

(6) Real Time Clock Interrupt. - This interrupt is described under topic 8, "Real Time Clock".

(7) Input/Output Channel Interrupts. - The input/output control provides for both internal (buffer completed) interrupt and external interrupt processing on all channels. See section 4-5 for details and priorities.

(8) Real Time Clock. - When the computer is running and the REAL TIME CLOCK switch is in the ON position, the contents of location 00024 is automatically incremented every 5 milliseconds. The incrementing process does not interrupt the execution of a program, but does require two memory cycles for each update.

An interrupt will occur immediately after the clock has been updated from all 1's to all 0's. This interrupt forces the instruction in location 00025 to be executed in interrupt mode. Since a full 24-bit word is used for the clock location, its 0 to 0 period is slightly less than 24 hours.

The programmer may use this interrupt feature to generate any accurate real time interval by pre-setting the contents of location 00024.

g. Assigned Core Memory Special Locations and Sense Switches

OCTAL	
00000	Master Clear - Start Location
00001	Index Register #1
00002	Index Register #2
00003	Index Register #3
to 00004 – 00017	Reserved for Index Register Expansion
00020	Fault Interrupt Location
00021	Parity Interrupt Location, I/O
00022	Parity Interrupt Location, Program
00023	Manual Interrupt Location
00024	Real Time Clock
00025	Clock = 0 Interrupt Location
00026	Power Failure Interrupt Location
0030	Input Buffer Status, Channels 0 - 7
0031	Output Buffer Status, Channels 0 - 7
0032	Input Buffer Status, Channels 8 - 15
0033	Output Buffer Status, Channels 8-15
to 0040 – 0057	Input Buffer Control Word for Channels 0 through 15
to 0060 – 0077	Output Buffer Control Word for Channels 0 through 15
to 0100 – 0117	Input Buffer Complete Interrupt Locations for Channels 0 through 15
to 0120 – 0137	Output Buffer Complete Interrupt Locations for Channels 0 through 15
to 0140 – 157	External Interrupt Status Word Address for Channels 0 through 15
to 160 – 177	Input External Interrupt Locations for Channels 0 through 15
to 0200 – 0217	Output External Interrupt Locations for Channels 0 through 15

h. Detailed Description of Instructions. - Instructions are divided functionally into eight categories: data transfer, arithmetic, logical, shift, increment, conditional skip, jump, and input-output. The assembler, descriptive name, and field designation are presented for each instruction. Timing is given as the number of

Sense Switch Assignments

Bit	Manual Switch	Input Data Available	Output Data Lines Available	Special
	GP = 0	GP = 1	GP = 2	GP = 3
0	Switch 0	Channel 0	Channel 0	Arithmetic Overflow
1	Switch 1 (S1)	Channel 1	Channel 1	Division Error
2	Switch 2 (S2)	Channel 2	Channel 2	Shift Parity
3	Switch 3	Channel 3	Channel 3	Sign A
4	Switch 4	Channel 4	Channel 4	Sign of Q
5	Switch 5	Channel 5	Channel 5	Contents of A = 0
6	Switch 6	Channel 6	Channel 6	Memory Range Overflow
7	Switch 7	Channel 7	Channel 7	Master Lockout
8	Switch 8	Channel 8	Channel 8	Program Flag Light (set)
9	Switch 9	Channel 9	Channel 9	
10	Switch 10	Channel 10	Channel 10	
11	Switch 11	Channel 11	Channel 11	
12	Clock - ON, OFF	Channel 12	Channel 12	Not Assigned
13	Parity Reaction	Channel 13	Channel 13	
14	Parity Reaction I/O	Channel 14	Channel 14	
15	Fault Reaction	Channel 15	Channel 15	

memory accesses to acquire the instruction and operand and to perform the instruction without indexing or indirect addressing. One memory access is added for each indirect addressing operation and one memory access for each address modification by an index register. The first instruction in a category is described in more detail than its various options. A composite list of all instructions appears in Figure 1-1-C.

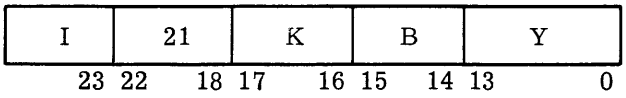
(1) Address Indexing. - For those instructions which allow indexing, the specification of a B modifier will cause the contents of the designated B-Index location to be added to the direct or indirect Y field. The resulting sum serves as the operand address or the new indirect address. Each indirect address level can indicate a different or no B modifier.

Timing: 1 cycle for direct addressing
1 cycle per level for indirect addressing

(2) Multiple-Level Indirect Addressing. - Execution of an indirect instruction is identical to the execution of its direct equivalent once the effective address has been attained. The indirect cycle can run indefinitely and is stopped by the acquisition of any direct instruction code. The indirect address at every level is modified by the contents of B specified at that level.

(3) Enter Class Instructions (DATA TRANSFER)

ENA
Enter A



With K = 0, the contents of location Y replace the contents of the A register. The contents of location Y remain unchanged. Indexing and indirect addressing apply in all combinations. No other registers are changed and the next instruction is taken in sequence. Timing: 2 cycles.

Bit Number							Function Mnemonic		Definition	Assembly Language Mnemonics							
23	22 18	17 16	15 14	13 0			PAGE			DESIGNATOR FOR BITS 16, 17							
0	0 0						FAULT	63	FAULT	FAULT							
1	0 0	CHANNEL		Y			EXI	66	(CHANNEL) → Y & input ack.	EXI							
0	0 1	S		S		N	SHF	41	Shift	SCA SCAQ RBAQ TLY SHR SHR Q SHR AQ SHR C SHR QC SHR AQC LGR LGR Q LGR AQ LGR C LGR QC LGR AQC SHL SHL Q SHL AQ SHL C SHL QC SHL AQC LGL LGL Q LGL AQ LGL C LGL QC LGL AQC							
1	0 1	CHANNEL		Y			EXO	66	(Y) → CHANNEL & output ack.	EXO							
0	0 2	NOT ASSIGNED															
1	0 2	NOT ASSIGNED															
0	0 3	L	B	Y			JDB	50	(B) - 1 → B, cond. jump	0 JDB P		1 JDB M		2 JDB Z		3 JDB N	
1	0 3	K	B	Y			RAO	53	(Y) + 1 → Y	0 ADD1		1 ADD1 RH		2 ADD1 LH		3 ADD1 Y	
0	0 4	L	B	Y			SSK	55	(Y), cond. skip	0 SSK P		1 SSK M		2 SSK Z		3 SSK N	
1	0 4	M	B	Y			CIS	72	Change interrupt status	0 CIR		1 SAL		2 ELS		3 SLS	
0	0 5	L	B	Y			SSH	55	(Y) shifted left 1, cond. skip	0 SSH P		1 SSH M		2 SSH Z		3 SSH N	
1	0 5	CHANNEL					CBM	68		BIN, BINI, BOT, BOTI, BOMT, BOMTI, BIMT, BIMTI, SOBS, SIBS							
0	0 6	L	B	Y			RSO	52	(Y) - 1 → Y, cond. skip	0 SUB1 P		1 SUB1 M		2 SUB1 Z		3 SUB1 N	
1	0 6	CHANNEL		Y			EXF	66	(Y) → CHANNEL & EXP signal	EXF							
0	0 7	GP		BITS			SSW	56	Senseline Test	0 SSW		1 IDA		2 ODA		3 SOF, SDE, SOP, SAM, SQM, SAZ, SMO, SMS, HEY	
1	0 7	M	B	Y			NXN	65 32 63		0 NEG		1 XAQ		2 JHT		3 NOP	
I	1 0	K	B	Y (Y)			AND AND	38	(A) ∧ (Y) → A	0 AND		1 AND RH		2 AND LH		3 AND Y	*
I	1 1	K	B	Y (Y)			IOR IOR	38	(A) ∨ (Y) → A	0 IOR		1 IOR RH		2 IOR LH		3 IOR Y	*
I	1 2	K	B	Y (Y)			SCL SCL	39	Clear (A) for 1's of (Y)	0 SCL		1 SCL RH		2 SCL LH		3 SCL Y	*
I	1 3	K	B	Y (Y)			EOR EOR	39	(A) ⊕ (Y) → A	0 EOR		1 EOR RH		2 EOR LH		3 EOR Y	*
I	1 4	K	B	Y (Y)			SBT SBT	40	(Y) → A for 1's of (Q)	0 SBT		1 SBT RH		2 SBT LH		3 SBT Y	*
I	1 5	L	B	Y (Y)			SCP SCP	56	(A) - [(Q) ∧ (Y)], cond skip	0 SCP P		1 SCP M		2 SCP Z		3 SCP N	
I	1 6	K	B	Y (Y)			CMP CMP	53	if (A) < (Y) skip	0 CMP		1 CMP RH		2 CMP LH		3 CMP Y	*
I	1 7	L	B	Y (Y)			CYS CYS	54	(A) - (Y), cond. skip	0 CYS P		1 CYS M		2 CYS Z		3 CYS N	
I	2 0	K	B	Y (Y)			ENQ ENQ	29	Enter Q, (Y) → (Q)	0 ENQ		1 ENQ RH		2 ENQ LH		3 ENQ Y	*
I	2 1	K	B	Y (Y)			ENA ENA	28	Enter A, (Y) → A	0 ENA		1 ENA RH		2 ENA LH		3 ENA Y	*
I	2 2	K	B‡	Y (Y)			ENB ENB	30	Enter B, (Y) → B	0 ENB		1 ENB RH		2 ENB LH		3 ENB Y	*
I	2 3	K	B	Y (Y)			RAD RAD	37	(A) + (Y) → Y	0 RAD		1 RAD RH		2 RAD LH		3 RAD Y	
I	2 4	K	B	Y (Y)			ADD ADD	33	(A) + (Y) → A	0 ADD		1 ADD RH		2 ADD LH		3 ADD Y	*
I	2 5	K	B	Y (Y)			SUB SUB	34	(A) - (Y) → A	0 SUB		1 SUB RH		2 SUB LH		3 SUB Y	*
I	2 6	K	B	Y (Y)			MUL MUL	34	(Q) x (Y) → AQ	0 MUL		1 MUL RH		2 MUL LH		3 MUL Y	*
I	2 7	K	B	Y (Y)			DIV DIV	35	(AQ) ÷ (Y) → Q, A	0 DIV		1 DIV RH		2 DIV LH		3 DIV Y	*
I	3 0	K	B	Y (Y)			STQ STQ	31	Store Q, (Q) → Y	0 STQ		1 STQ RH		2 STQ LH		3 STQ Y	
I	3 1	K	B	Y (Y)			STA STA	30	Store A, (A) → Y	0 STA		1 STA RH		2 STA LH		3 STA Y	
I	3 2	K	B‡	Y (Y)			STB STB	32 64	Store B, (B) → Y Store Zero → Y	0 STB B=0 STZ		1 STB RH B=0 STZ RH		2 STB LH B=0 STZ LH		3 STB Y B=0 STZ Y	
I	3 3	K	B	Y (Y)			RSB RSB	37	(A) - (Y) → Y	0 RSB		1 RSB RH		2 RSB LH		3 RSB Y	
I	3 4	K	B	(Y)			XEQ INOP	65 63	Execute (Y)	0 XEQ		1 NOT ASSIGNED		2 NOT ASSIGNED		3 INOP	
I	3 5	M	B	Y (Y)			JPA JPA	60	(A), cond. jump	0 JMP AP		1 JMP AM		2 JMP AZ		3 JMP QP	
I	3 6	M	B	Y (Y)			JPM JPM	60	misc. cond. jump	0 JMP		1 JMP S1		2 JMP S2		3 JMP CML Not Indexable	
I	3 7	M	B	Y (Y)			JSR JSR	62	(P) → Y, jump to (Y) + 1	0 JSR		1 JSR AM		2 JSR AZ		3 JSR SML Not Indexable	

L designator condition

P: L = 0 ( ) = + M: L = 1 ( ) = - Z: L = 2 ( ) = 0 N: L = 3 ( ) ≠ 0

I designator — To specify indirect addressing in the assembly language enclose address field in parenthesis

I = 0 no direct addressing I = 1 indirect addressing

K designator

blank:	K = 0	USE FULL WORD
RH:	K = 1	USE RIGHT HALF
LH:	K = 2	USE LEFT HALF
Y:	K = 3	USE Y FIELD

B designator

blank:	B = 0	NO INDEXING
,1:	B = 1	Y + (B1) → Y
,2:	B = 2	Y + (B2) → Y
,3:	B = 3	Y + (B3) → Y

S designator

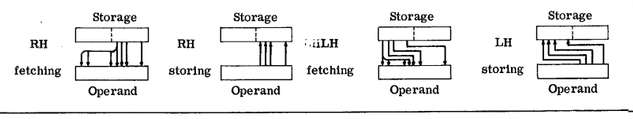
b_{17}	b_{16}		b_{15}		b_{14}		b_{11}	
0	0	see SCA and RBAQ	1	circular	0	logical	0	left
1	0	use A	0	linear	1	arithmetic	1	right
0	1	use Q						
1	1	use AQ	N is 6 bit shift count.					

GP DESIGNATOR

SEE OPTIONS TO RIGHT

M DESIGNATOR

Interpretation dependent on particular instruction.



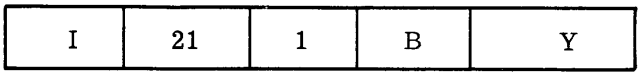
* The low order 14 bits of the instruction with 10 leading zeros are used as the operand providing B = 0. If B ≠ 0, the low order 14 bits of the designated index register are used as the operand.

\+ The low order 14 bits of the location called out by the address are used as the operand. The high order 10 bits are zero after fetching and are not disturbed while storing.

\+ The B field designates the index register to which the function applies unless indirect addressing is specified. Indirect addressing uses the B field to designate indexing and uses the B field of the indirect address word as part of the function.

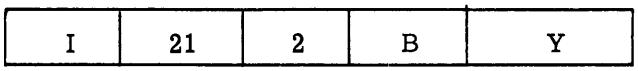
Figure 1-1-C. Composite List of all Instructions

ENA RH
Enter A from Right Half



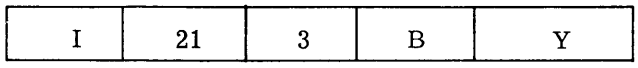
With K = 1, the contents of bits 0 to 10 of location Y replace the contents of bits 0 to 10 of A. The contents of bit 11 (sign) of location Y is replicated in bits 11 to 23 of A. Thus the sign and integer magnitude of the right half are represented in 2's complement form as a full 24-bit word in the A register. The contents of location Y remain unchanged. Indexing and indirect addressing apply in all combinations. The RH option applies to the effective address after indexing and indirect addressing. No other registers are changed and the next instruction is taken in sequence. Timing: 2 cycles.

ENA LH
Enter A from Left Half



With K = 2, contents of bits 12 to 22 of location Y replace the contents of bits 0 to 10 of A. The contents of bit 23 (sign) of location Y is replicated in bits 11 to 23 of A. See ENA RH for further details.

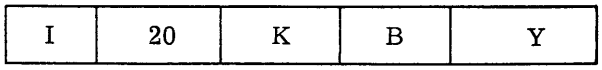
ENA Y
Enter A with Bits of Y



With K = 3, bits 0 to 13 of this instruction (i. e., the Y field itself) with 10 leading zero bits replace the contents of the A register. Thus the 14 bit Y address field of the ENA instruction is expanded into a full word positive integer in A. If an index register is specified, bits 0 to 13 of the index register with 10 leading zeros replace the contents of the A register. If indirect addressing is specified the net result is to place the effective address in bits 0 to 13 of A and zero bits 14 to 23 of A. No other registers are changed and the next instruction is taken in sequence.

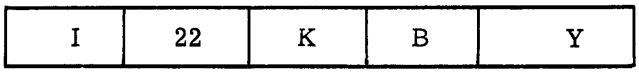
Timing: 2 cycles.

ENQ
Enter Q



The contents of location Y replace the contents of the Q register. The contents of Y remain unchanged. Indexing and indirect addressing apply in all combinations. No other registers are changed and the next instruction is taken in sequence. Timing: 2 cycles. The K options: right half, left half and full word apply as described under ENA.

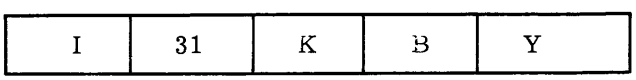
ENB
Enter B



The contents of location Y replace the contents of the memory location specified by the B field. The Y field is not modified by the specified index register. If indirect addressing is called for, it proceeds as usual with Y being modified by the B index register designated in the instruction. The index register designated in the contents of location Y is then loaded with the contents of the unindexed indirect address. Note: location 00000 may be loaded even though it is not considered an index register. No other registers are changed, and the next instruction is taken in sequence. Memory range overflow is possible if B $\neq$ 0 and indirect addressing is used. Timing: 3 cycles.

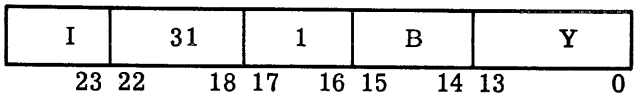
The K = 1 and K = 2 options apply as described under ENA. When the K = 3, Y, option is used, bits 0 to 13 of this instruction replace 0 to 13 of the specified index register. Bits 14 to 23 of the index register remain unchanged.

STA
Store A



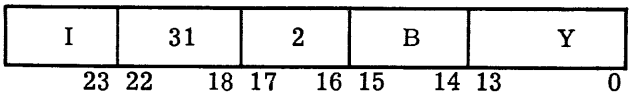
When K = 0, the contents of A replace the contents of location Y. The A register remains unchanged. Indexing and indirect addressing apply in all combinations. No registers are changed, and the next instruction is taken in sequence. Timing: 2 cycles.

STA RH
Store **A** in Right Half



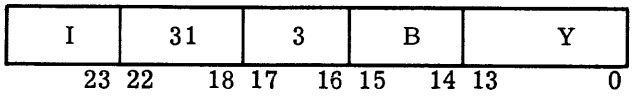
When K = 1, the contents of bits 0 to 10 of A replace the contents of bits 0 to 10 of location Y. The contents of bit 11 (the same as bit 23, the sign, if the half word has not overflowed) of A replace the contents of bit 11 of location Y. Bits 12 to 23 of location Y remain unchanged. See STA for further details.

STA LH
Store A in Left Half



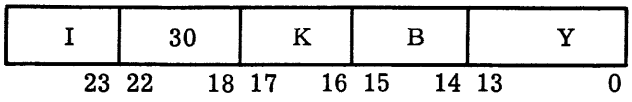
When K = 2, the contents of bits 0 to 10 of A replace the contents of bits 12 to 22 of location Y. The contents of bit 11 of the A register replace the contents of bit 23 of location Y. Bits 0 to 11 of location Y remain unchanged. See STA for further details.

STA Y
Store A in Address



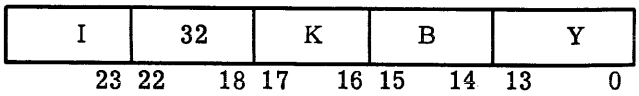
When, K = 3, the contents of bits 0 to 13 of the A register replace the contents of bits 0 to 13 of location Y. Bits 14 to 23 of location Y remain unchanged. See STA for further details.

STQ
Store Q



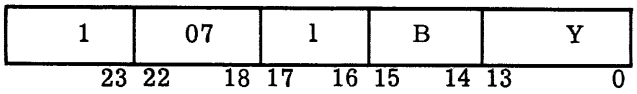
The contents of Q replace the contents of location Y. The Q register remains unchanged. Indexing and indirect addressing apply in all combinations. No registers are changed and the next instruction is taken in sequence. Timing: 2 cycles. The K options apply as described in STA.

STB
Store B



The contents of the specified B-index register replace the contents of location Y. The Y field of the instruction is not modified by the specified index register. If indirect addressing is called for, it proceeds as usual with Y being modified by the B-index register designated. When an effective address has been reached the contents of the B-index register specified are then stored at the unindexed address. The case where B = 0 is described under STZ, the store zero instruction. No registers are changed and the next instruction is taken in sequence. Memory range overflow is possible if B $\neq$ 0 and indirect addressing is used. Timing: 3 cycles. The K options apply as described under STA.

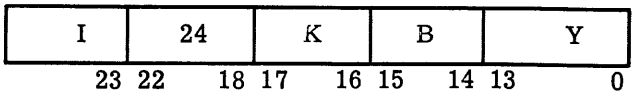
XAQ
Exchange A and Q and Jump



The contents of the A and Q registers are interchanged. No memory locations are changed. The next instruction is taken from location Y. B-Indexing can be used to modify the location from which the next instruction is taken. Indirect addressing is not available. Timing: 2 cycles.

(4) Arithmetic Instructions

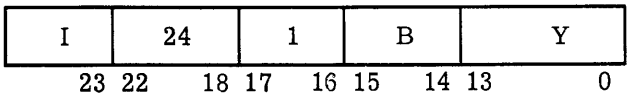
ADD
Add



When K = 0, the contents of location Y are added to the contents of the A register. Both numbers are treated as full word 2's complement numbers. The result is a full word 2's complement number in the A register. The Q register and the contents of location Y remain unchanged. B-Indexing and indirect addressing apply in all combinations. Overflow is

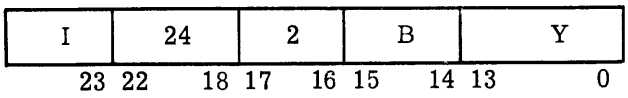
possible. The next instruction is taken in sequence. Timing: 2 cycles.

ADD RH
Add Right Half



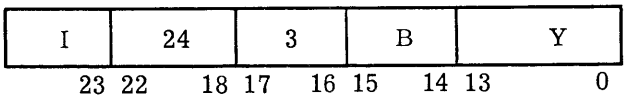
When K = 1 the contents of bits 0 to 10 of location Y are used as bits 0 to 10 of the operand. The contents of bit 11 (sign) of location Y are replicated in bits 11 to 23 of the operand. The resulting full word 2's complement operand is added to the contents of the A register. (See ADD for further details.)

ADD LH
Add Left Half



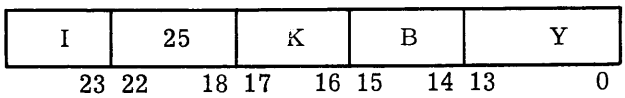
When K = 2, the contents of bits 12 to 22 of location Y are used as bits 0 to 11 of the operand. The contents of bit 23 (sign) of location Y is repliacted in bits 11 to 23 of the operand. The resulting full word 2's complement operand is added to the contents of the A register. (See ADD for further details).

ADD Y
Add Bits of Y



When K = 3, bits 0 to 13 of this instruction (i. e. , the Y field itself) with 10 leading zero bits are used as a full word 2's complement operand providing B = 0. If indexing is specified and indirect addressing is not used, bits 0 to 13 of the specified index registers are used as the operand. (See ADD for further details.)

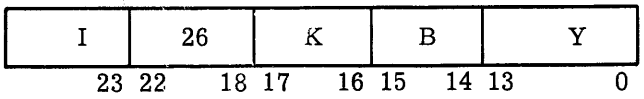
SUB
Subtract



The contents of location Y are subtracted from the contents of the A register. Both numbers are treated as full word 2's complement numbers. The result is a full word 2's complement number. The Q register and the contents of location Y remain unchanged. B-Indexing and indirect addressing apply in all combinations. Overflow is possible. The next input is taken in sequence. Timing: 2 cycles.

The K options apply as described under ADD RH, ADD LH, and ADD Y, respectively, with the operand being subtracted from the contents of the A register.

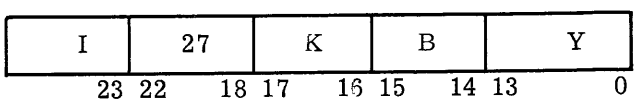
MUL
Multiply



The contents of location Y are multiplied by the contents of the Q register. Before multiplication the A register is cleared. The operand and contents of the Q register are treated as full word 2's complement numbers. The high order part of the product is retained in the A register, and the low order part of the product is retained in the Q register. Bit 23 of the A register is the sign of the product as is bit 23 of the Q register. The only exceptions to this rule are when the contents of A are negative and the magnitude of A is zero (and thus positive), or the contents of Q are negative and the magnitude of A is zero (and thus positive). Therefore, the Q register may be stored as the signed product of two integers with less than 23 significant bits, or the A register may be stored as the signed product of two fractions with less than 23 significant bits without loss of accuracy, etc. In general, the binary point of the product is n bits to the left of bit 0 where n is computed as the sum of the number of bits the binary points are to the left of bit 0 in the multiplier and multiplicand. Indexing and indirect addressing apply in all combinations. The overflow indicator is not changed by this instruction. The next instruction is taken in sequence. Timing: 6 cycles.

The K options apply as described under ADD RH, ADD LH, and ADD Y, respectively, with the operand being multiplied by the contents of the Q register.

DIV
Divide



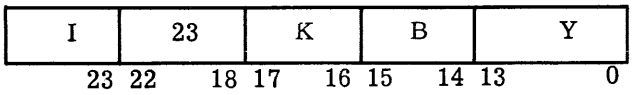
The contents of the A and Q register, treated as a 46-bit dividend, are divided by the

contents of location Y. It is assumed A and Q are full word 2's complement numbers with the same sign. The operand is also treated as a full word 2's complement number. The signed quotient which is retained in the Q register is in the 2's complement form. The remainder is retained in the A register in the 2's complement form with the same sign as Q. The only exception to this rule is when the quotient is negative and the remainder is zero, in which case, the sign of the A register is plus. The division error indicator is turned on if significant bits of quotient would be lost (i. e. , if the magnitude in the A register is greater than or equal to the magnitude in location Y). If a division error is detected, the A and Q registers remain unchanged. The Q register may be stored as the integer quotient of two integers or the fractional quotient of two integers or the fractional quotient of two fractions, etc. In general, the binary point for the quotient is n bits to the left of bit 0 when n is computed as the number of bits the binary point of the dividend is to the left of bit 0 minus the number of bits the binary point of the divisor is to the left of bit 0. The binary point of the remainder corresponds to the binary point of the dividend.

B-Indexing and indirect addressing apply in all combinations. The overflow indicator is not changed by this instruction. The next instruction is taken in sequence. Timing: 11 cycles.

The K options apply as described under ADD RH, ADD LH, and ADD Y, respectively, with the operand being the divisor.

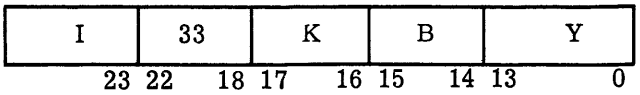
RAD
Replace ADD



Same as ADD except the sum replaces the contents of location Y and the contents of the A register remain unchanged. Timing: 3 cycles.

Two K options right half and left half apply as described for ADD RH and ADD LH with the same bits being replaced by the sum that were used as the operand. When the K = 3, Y, option is used, bits 0 to 13 of location Y are used as the operand. Bits 0 to 13 of the sum of the operand and the A register replace bits 0 to 13 of the location from which the operand was acquired. Use of the B index registers apply for all options.

RSB
Replace Subtract

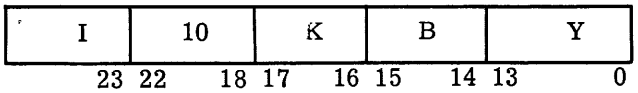


Same as SUB, except the difference replaces the contents of location Y, and the contents of the A register remain unchanged. Timing: 3 cycles.

The K options apply as described for RAD.

(5) Logical Instructions. - In combination and with appropriate masks, the following instructions can be used to obtain all 16 logical functions (bit by bit) of two words.

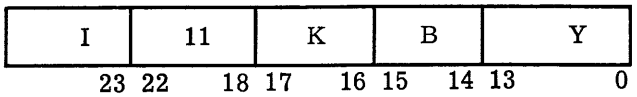
AND
Logical AND



The contents of location Y and the contents of the A register are examined bit by bit for all 24 bits. Whenever both have a 1 in a given bit position, a 1 is left in that position of the A regiter, otherwise a 0 replaces the contents of the A register in that position. The contents of the Q register and location Y remain unchanged. B-indexing and indirect addressing apply in all combinations. The next instruction is taken in sequence. Timing: 2 cycles.

The K options apply as described under ADD RH, ADD LH, and ADD Y respectively for acquiring the operand from memory.

IOR
Logical Inclusive
OR

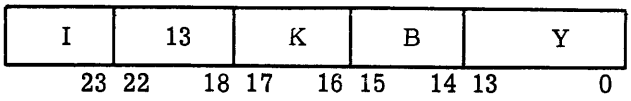


The contents of location Y and the contents of the A register are examined bit by bit for all 24 bits. Whenever either or both have a 1 in a given bit position a 1 replaces the contents of the A register in that position, otherwise a 0 replaces the contents of the A register in that position. The contents of the Q register and location Y remain unchanged. B-Indexing and indirect addressing apply in all combinations.

The next instruction is taken in sequence. Timing: 2 cycles.

The K options apply as described under ADD RH, ADD LH, and ADD Y, respectively, for acquiring the operand.

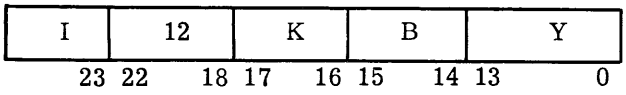
EOR
Logical Exclusive
OR



The contents of location Y and the contents of the A register are examined bit by bit for all 24 bits. Whenever both have a 0 or both have a 1 in a given bit position a 0 replaces the contents of that position in the A register, otherwise, a 1 replaces the contents of that position in the A register. The contents of the Q register and location Y remain unchanged. B-Indexing and indirect addressing in any combination is allowed. The next instruction is taken in sequence. Timing: 2 cycles.

The K options apply as described under ADD RH, ADD LH, and ADD Y, respectively, for acquiring the operand.

SCL
Selective Clear

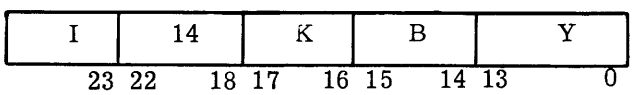


The contents of location Y and the contents of the A register are examined bit by bit for all 24 bits. Whenever there is a 1 in a given bit position of the contents of location Y a zero is placed in the corresponding position of the A register; otherwise the A register is not changed for that position. The contents of the Q register and location Y remain unchanged.

B-Indexing and indirect addressing apply in all combinations. The next instructions are taken in sequence. Timing: 2 cycles.

The K options apply as described under ADD RH, ADD LH, and ADD Y, respectively for acquiring the operand.

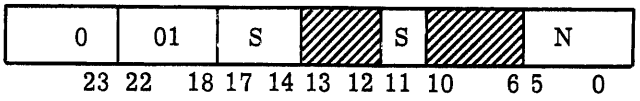
SBT
Substitute



The Q register is used as a 24-bit mask. For each bit position in the Q register which is a 1, the contents of the corresponding bit of location Y replace the bit of the A register for that position. The A register is unchanged in those positions where the Q register has zeros. The contents of the Q register and location Y remain unchanged. B-Indexing and indirect addressing in any combination is allowed. The next instruction is taken in sequence. Timing: 2 cycles.

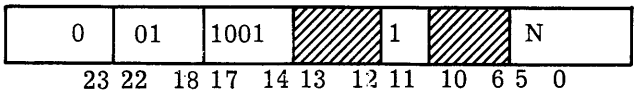
The K options apply as described under ADD RH, ADD LH, and ADD Y, respectively, for acquiring the operand.

(6) Shift Instructions. - The basic format for the shift instruction is



The five bits of the S designator are used to specify the 24 possible shift instructions and four special instructions. For shift instructions, the 6 bits of N specify the number of places to shift. The S designator in position 11 controls the direction of the shift, i.e. right or left shift. Thus a right-half (K = 1) store instruction into a shift instruction location can set the count and direction of the shift instruction. Neither indirect addressing nor B-indexing is allowed. Timing is proportional to the number of positions shifted; 1.3 + N/6 cycles.

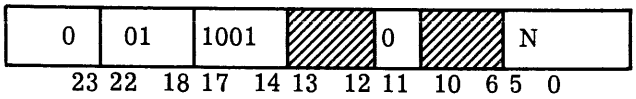
SHR
Shift Right



The contents of the A register are shifted right the number of positions specified by N. Bits shifted out of bit position 0 are lost. Bit 23 (the sign) of A is not shifted, and positions vacated on the left are filled from bit 23. Thus when the contents of the A register are considered as a 2's complement number, the effect of the shift is to divide by 2^N while retaining the

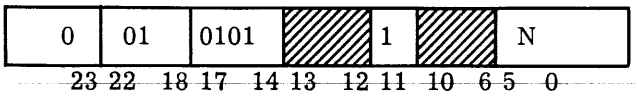
2's complement form. The Q register is unchanged. The next instruction is taken in sequence. Timing: 1 to 5 cycles.

SHL
Shift Left



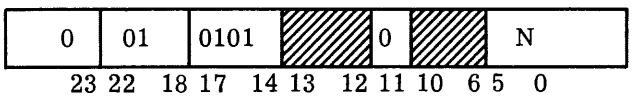
The contents of the A register are shifted left the number of positions specified by N. Bits shifted out of position 22 are lost and Bit 23 (the sign) of A is not shifted. Positions vacated on the right are filled with zeros. The overflow indicator is turned on if a significant bit leaves position 22 when the contents of the A register are considered as a 2's complement number. The effect of the shift is to multiply by 2^N while retaining the 2's complement form. The Q register is unchanged. The next instruction is taken in sequence. Timing: 1 to 5 cycles.

SHR Q
Shift Q Right



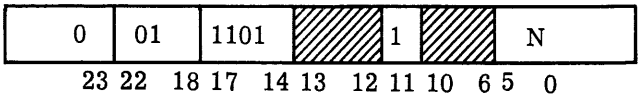
The contents of the Q register are shifted right the number of positions specified by N. Bits shifted out of bit position 0 are lost. Bit 23 (the sign) of Q is not shifted, and positions vacated on the left are filled from bit 23. Thus, when the contents of the Q register are considered as a 2's complement number, the effect of the shift is to divide by 2^N while retaining the 2's complement form. The A register is unchanged. The next instruction is taken in sequence. Timing: 1 to 5 cycles.

SHL Q
Shift Q Left



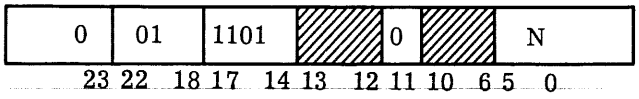
The contents of the Q register are shifted left the number of positions specified by N. Bits shifted out of bit position 22 are lost. Bit 23 (the sign) of Q is not shifted. Positions vacated on the right are filled with zeros. The overflow indicator is turned on if a significant bit leaves position 22 when the contents of the Q register are considered as a 2's complement number. The effect of the shift is to multiply by 2^N while retaining the 2's complement form. Timing: 1 to 5 cycles.

SHR AQ
Shift A and Q Right



The contents of the A and Q register are shifted right together the number of positions specified by N. Bits shifted out of position 0 of the Q register are lost. Bits shifted out of position 0 to the A register enter position 22 of the Q register. Bit 23 of the A and Q registers are not shifted. Positions vacated on the left of the A register are filled from bit 23 of the A register. Thus when the contents of the AQ register are considered as a double length 2's complement number, the effect of the shift is to divide the double length number by 2^N while retaining the 2's complement form. The next instruction is taken in sequence. Timing: 1 to 9 cycles.

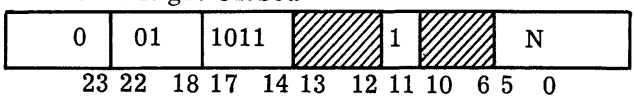
SHL AQ
Shift A and Q Left



The contents of the A and Q register are shifted left together the number of positions specified by N. Bits shifted out of position 22 of the A register are lost. Bits shifted out of position 22 of the Q register enter position 0 of the A register. Positions vacated at the right of the Q register are filled with zeros.

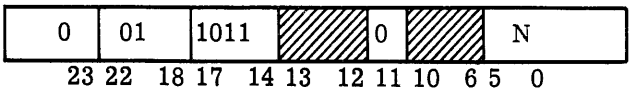
The overflow indicator is turned on if a significant bit leaves position 22 when the contents of the AQ register are considered as a double length 2's complement number. The effect of the shift is to multiply the double length number by 2^N while retaining the 2's complement form. The next instruction is taken in sequence. Timing: 1 to 9 cycles.

SHR C
Shift Right Closed



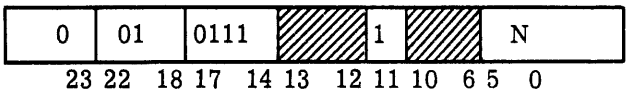
The contents of the A register are shifted right the number of positions specified by N. Bits shifted out of position 0 enter position 22. Position 23 is not shifted. The Q register remains unchanged and no overflow is possible. The next instruction is taken in sequence. Timing: 1 to 5 cycles.

SHL C
Shift Left Closed



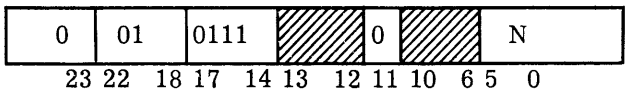
The contents of the A register are shifted left the number of positions specified by N. Bits shifted out of position 22 enter position 0. Position 23 is not shifted. The Q register remains unchanged and no overflow is possible. The next instruction is taken in sequence. Timing: 1 to 5 cycles.

SHR QC
Shift Q Right Closed



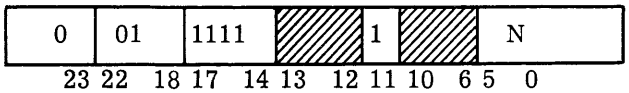
The contents of the Q register are shifted right the number of positions specified by N. Bits shifted out of position 0 enter position 22. Position 23 is not shifted. The A register remains unchanged and no overflow is possible. The next instruction is taken in sequence. Timing: 1 to 5 cycles.

SHL QC
Shift Q Left Closed



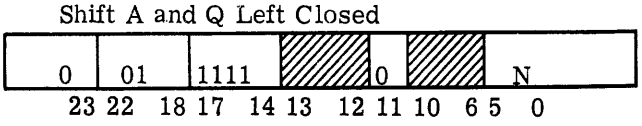
The contents of the Q register are shifted left the number of positions specified by N. Bits shifted out of position 22 enter position 0. Position 23 is not shifted. The A register remains unchanged and no overflow is possible. The next instruction is taken in sequence. Timing: 1 to 5 cycles.

SHR AQC
Shift A and Q Right Closed



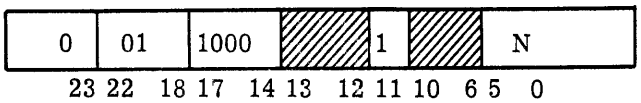
The contents of the A and Q are shifted right together the number of positions specified by N. Bits shifted out of position 0 of the A register enter position 22 of the Q register. Bits shifted out of position 0 of the Q register enter position 22 of the A register. Position 23 of A and Q registers is not shifted. No overflow is possible. The next instruction is taken in sequence. Timing: 1 to 9 cycles.

SHL AQC
Shift A and Q Left Closed



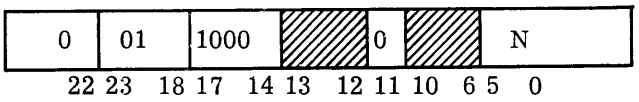
The contents of the A and Q registers are shifted left together the number of positions specified by N. Bits shifted out of position 22 of the A register enter position 0 of the Q register. Bits shifted out of position 22 of the Q register enter position 0 of the A register. Position 23 of A and Q registers is not shifted. No overflow is possible. The next instruction is taken in sequence. Timing: 1 to 9 cycles.

LGR
Logical Right Shift



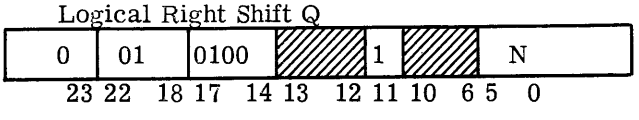
The contents of the A register are shifted right the number of positions indicated by N. Bits shifted out of position 0 are lost. Positions vacated on the left are filled with zeros. The Q register is not changed. Parity is accumulated. No overflow is possible. The next instruction is taken in sequence. Timing: 1 to 5 cycles.

LGL
Logical Left Shift



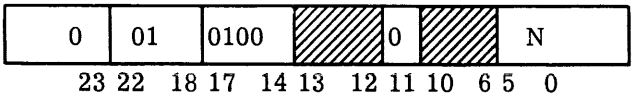
The contents of the A register are shifted left the number of positions indicated by N. Bits shifted out of position 23 are lost. Positions vacated on the right are filled with zeros. The Q register is not changed. Parity is accumulated. No overflow is possible. The next instruction is taken in sequence. Timing: 1 to 5 cycles.

LGR Q
Logical Right Shift Q



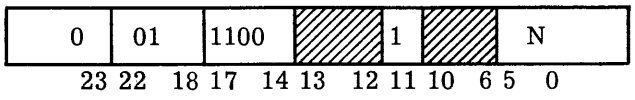
The contents of the Q register are shifted right the number of positions indicated by N. Bits shifted out of position 0 are lost. Positions vacated on the left are filled with zeros. The A register is not changed. Parity is accumulated. No overflow is possible. The next instruction is taken in sequence. Timing: 1 to 5 cycles.

LGL Q
Logical Left Shift Q



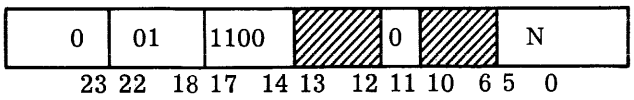
The contents of the Q register are shifted left the number of positions indicated by N. Bits shifted out of position 23 are lost. Positions vacated on the right are filled with zeros. The A register is not changed. Parity is accumulated. No overflow is possible. The next instruction is taken in sequence. Timing: 1 to 5 cycles.

LGR AQ
Logical Right Shift A and Q



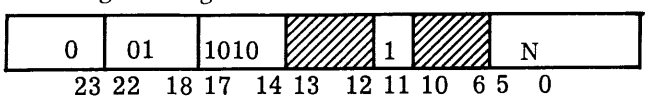
The contents of the A and Q registers are shifted right the number of positions indicated by N. Bits shifted out of position 0 of the A register enter position 23 of the Q register. Bits shifted out of position 0 of the Q register are lost. Positions vacated on the left are filled with zeros. Parity is accumulated. No overflow is possible. The next instruction is taken in sequence. Timing: 1 to 9 cycles.

LGL AQ
Logical Left Shift A and Q



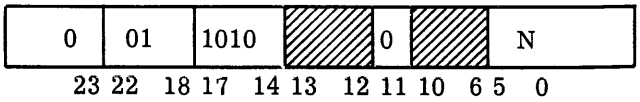
The contents of the A and Q registers are shifted left the number of positions indicated by N. Bits shifted out of position 23 of the Q register enter position 0 of the A register. Bits shifted out of position 23 of the A register are lost. Positions vacated on the right are filled with zeros. Parity is accumulated. No overflow is possible. The next instruction is taken in sequence. Timing: 1 to 9 cycles.

LGR C
Logical Right Shift Closed



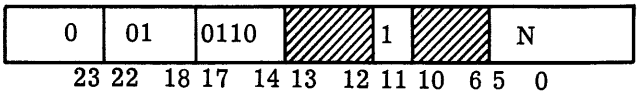
Same as LGR except bits shifted out of position 0 of the A register enter position 23 of the A register and parity is not accumulated.

LGL C
Logical Left Shift Closed



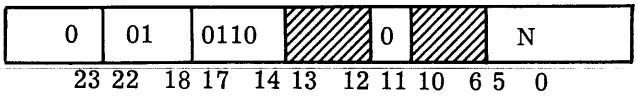
Same as LGL except bits shifted out of position 23 of the A register enter position 0 of the A register and parity is not accumulated.

LGR QC
Logical Right Shift Q Closed



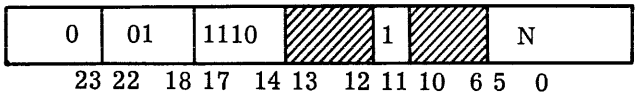
Same as LGR Q except bits shifted out of position 0 of the Q register enter position 23 of the Q register and parity is not accumulated.

LGL QC
Logical Left Shift Q Closed



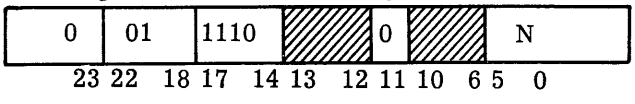
Same as LGL Q except bits shifted out of position 23 of the Q register enter position 0 of the register and parity is not accumulated.

LGR AQC
Logical Right Shift A and Q Closed



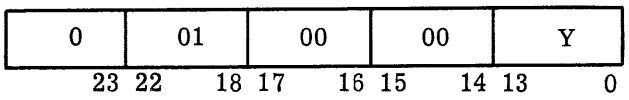
Same as LGR AQ except bits shifted out of position 0 of the Q register enter position 23 of the A register and parity is not accumulated.

LGL AQC
Logical Left Shift A and Q Closed



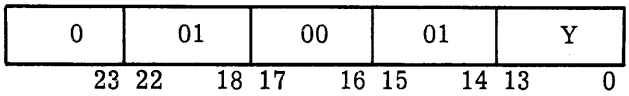
Same as LGL AQ except bits shifted out of position 23 of the A register enter position 0 of the Q register and parity is not accumulated.

SCA
Scale A Register



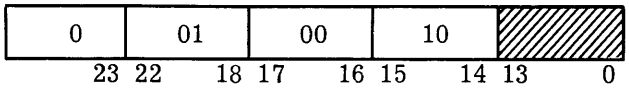
The contents of the A register are shifted left until the most significant bit is in position 22. Positions vacated on the right are filled with zeros. The sign (position 23) is not shifted. One is added to the shift counter for each shift, and the final contents of the shift counter are stored in location Y as a right-half integer. Scaling the number zero results in a shift count of 63 being stored. Scaling a number which is already in normalized form results in a shift count of zero being stored. To restore a scaled number, it is entered into the A register and a SHR instruction is executed with the scale count in its Y field. The Q register is not changed. No overflow is possible. Parity is not accumulated. The next instruction is taken in sequence. Timing: 1 to 7 cycles (except scaling zero: 13 cycles).

SCAQ
Scale A and Q Registers



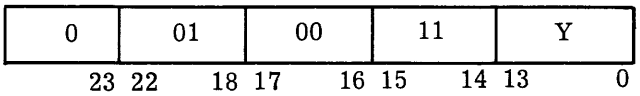
Same as SCA except A and Q are both shifted left with bits shifted out of position 22 of the Q register entering position 0 of the A register. Position 23 of the Q register is not shifted except if bits 0 to 22 of the Q register would result in all zeros in which case bit 23 of the Q register is forced to zero. The inverse operation is SHR AQ. Timing: 1 to 11 cycles (except scaling zero: 13 cycles).

RBAQ
Reverse Bits in A and in Q



The bits in the A register and Q register are reversed such that the contents of position 0 and position 23 are interchanged, the contents of position 1 and position 22 are interchanged, etc. No overflow is possible. Parity is not accumulated. The next instruction is taken in sequence. Timing: 6 cycles.

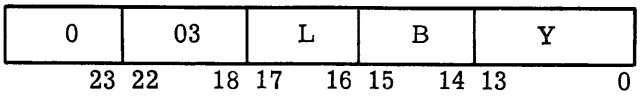
TLY
(Optional)
Tally 1's in the A Register



The number of 1's in the A register are counted and the result is stored in location Y as a right half integer. The contents of the A and Q registers remain unchanged by this instruction. The status of the overflow indicator is not changed. The next instruction is taken in sequence. Timing: 11 cycles.

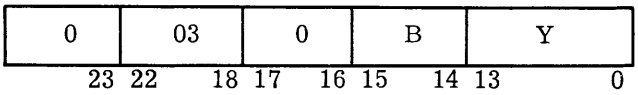
(7) Increment or Decrement Instructions

JDB
The basic format for the jump and decrement index register instruction is



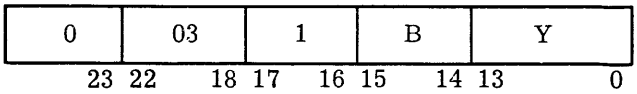
An integer 1 is subtracted from the contents of the designated index register. The result replaces the contents of that index register. The result also is used as an operand for a conditional jump according to the L designator as described below. Y is neither indexable nor indirectly addressable. The contents of register, indicators, and other memory locations remain unchanged. Timing: 3 cycles.

JDB P



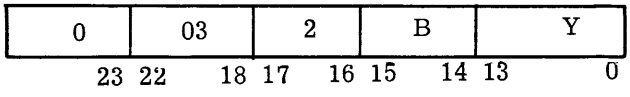
If the resulting operand has a sign which is plus, the next instruction is taken from location Y, otherwise the next instruction is taken in sequence.

JDB M



If the resulting operand has a sign which is minus, the next instruction is taken from location Y; otherwise the next instruction is taken in sequence.

JDB Z

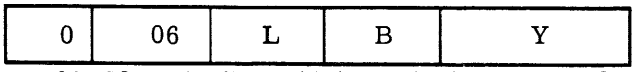


If the resulting operand is zero, the next instruction is taken from location Y; otherwise the next instruction is taken in sequence.

JDB N

If the resulting operand is non-zero, the next instruction is taken from location Y; otherwise the next instruction is taken in sequence.

SUB1
Subtract 1 from Storage and Skip



An integer 1 is subtracted from the contents of location Y as a full word. The result is used for an operand and placed back in the same location. Indexing applies. Indirect addressing does not apply. Overflow is possible. The contents of registers and other memory locations are not affected. Timing: 3 cycles.

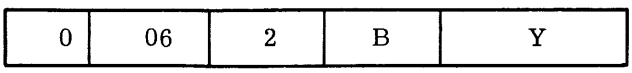
SUB1 P

If the resulting operand has a sign which is plus, the next instruction is skipped and execution proceeds from the following instruction; otherwise, the next instruction is taken in sequence.

SUB1 M

If the resulting operand has a sign which is minus, the next instruction is skipped and execution proceeds from the following instruction; otherwise, the next instruction is taken in sequence.

SUB1 Z

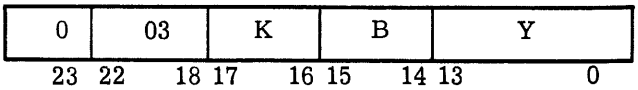


If the resulting operand is zero, the next instruction is skipped and execution proceeds from the following instruction; otherwise the next instruction is taken in sequence.

SUB1 N

If the resulting operand is non-zero, the next instruction is skipped and execution proceeds from the following instruction; otherwise the next instruction is taken in sequence.

ADD1
Add 1 to Storage

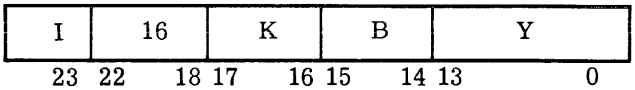


An integer 1 is added to the contents of location Y. The result is placed back in the same location. Indexing applies. Indirect addressing does not apply. Overflow is possible. The contents of registers and other memory locations are not affected. Timing: 3 cycles.

The right half, left half and full word options apply as described under STA RH, STA LH, and STA Y with the same portion fetched and incremented as is stored back.

(8) Skip Instructions

CMP
Compare



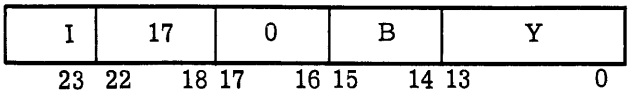
The contents of location Y are compared with the contents of the A register. If the contents of the A register are algebraically smaller, the the next instruction in sequence is skipped and the following instruction is executed; otherwise the next instruction in sequence is executed. Indexing and indirect addressing apply in all combinations. The contents of location Y and the A register remain unchanged. Timing: 2 cycles.

The right half, left half and full word options apply as described under ADD RH, ADD LH, and ADD Y, respectively.

CYS
Compare with Y and Skip

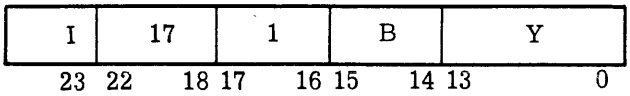
The contents of location Y are subtracted from the contents of the A register to form an operand. If the skip condition specified by the L designator is met, the next instruction in sequence is skipped and the following instruction is executed; otherwise the next instruction in sequence is executed. Indexing and indirect addressing apply in all combinations. The contents of location Y and the A register remain unchanged. Timing: 3 cycles. The options are as follows:

CYS P



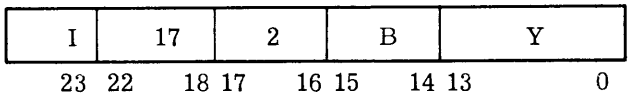
Skip if the sign of the operand is plus.

CYS M



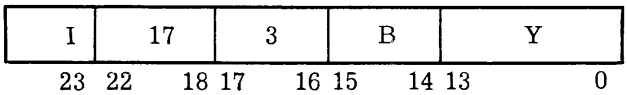
Skip if the sign of the operand is minus.

CYS Z



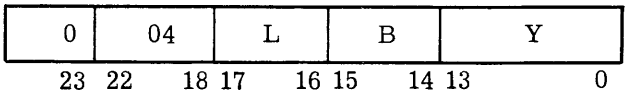
Skip if the operand is zero.

CYS N



Skip if the operand is non-zero.

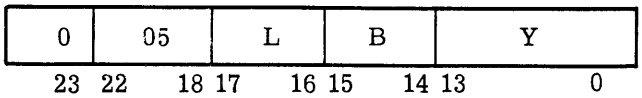
SSK
Storage Skip



The contents of location Y are examined as an operand. If the condition specified by the L designator is met, the next instruction in sequence is skipped and the following instruction is executed; otherwise the next instruction in sequence is executed. Indexing applies but indirect addressing is not available. The contents of location Y remain unchanged. Timing: 2 cycles.

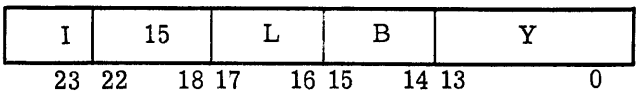
The P, M, A, and N options apply as described under CYS P, CYS M, CYS Z, and CYS N, respectively.

SSH
Storage Shift Skip



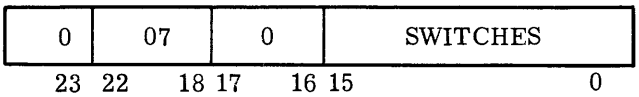
Same as SSK except the contents of location Y are shifted logically left closed one position then replaced. The test is made on the resulting shifted operand.

SCP
Selective Compare



The Q register is assumed to contain a mask. The logical AND of the contents of the Q register and the contents of location Y are formed. The result is subtracted from the contents of the A register to form an operand. Use of index registers and indirect addressing apply in all combinations. From this point the execution is the same as for CYS. Timing: 2 cycles.

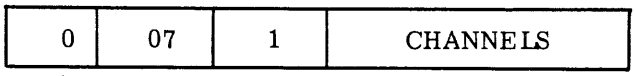
SSW
Sense Switch



The settings of the 16 sense switches on the programmers panel are sensed. The bit-by-bit AND of the sense switches and the SWITCHES bits are formed, and the result is combined by a logical OR to form the skip condition. If the skip condition is 1, the next instruction in sequence is skipped, and execution proceeds with the following instruction; otherwise the next instruction is executed in sequence. Neither indexing nor indirect addressing apply.

A switch in the up position is "ON" and has value 1. A switch in the down position is "OFF" and has value 0. The skip occurs when any of the switches are "ON" that are specified by 1's in the SWITCH field. No registers, indicators, or memory locations are changed by this instruction. Timing: 1 cycle.

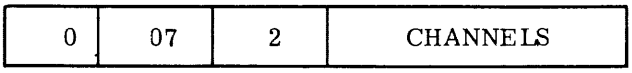
IDA
Input Data Available



The next instruction is skipped and execution proceeds from the following instruction if there is input data available on any channel where there is a 1 in the CHANNELS field; otherwise the next instruction is taken in sequence. No registers, indicators, or memory locations are changed by this instruction. Timing: 1 cycle.

> Note: The results of this instruction are unreliable if a buffer input mode is established on any of the channels being tested. In the assembly language, when designating one channel, the option field contains the decimal channel number to be tested. When designating more than one channel, an octal qualifier followed by octal digits are placed in the address field.

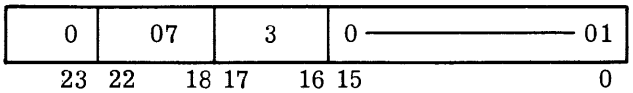
ODA
Output Data Lines Available



The next instruction is skipped and execution proceeds from the following instruction if the output lines are available on any channel where there is a 1 in the CHANNELS field. See description of IDA instruction for further details.

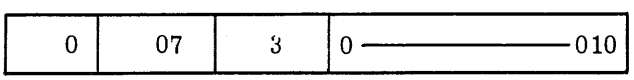
Indexing and indirect addressing do not apply to the following instructions. Timing: 1 cycle.

SOF
Skip on Overflow



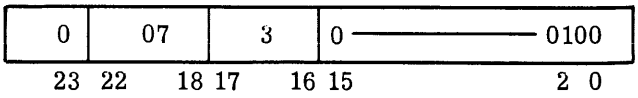
Skip the next instruction if the overflow indicator is "ON". The execution of this instruction turns the indicator "OFF".

SDE
Skip on Division Error



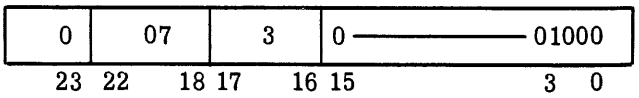
Skip the next instruction if the division error indicator is "ON". The execution of this instruction turns the indicator "OFF".

SOP
Skip on Odd Parity Developed by Shifting



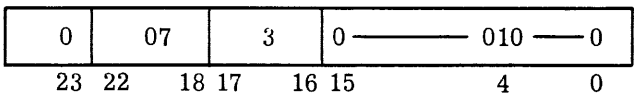
Skip the next instruction if the parity developed by open logical shifting is odd. The execution of this instruction sets the indicator to even.

SAM
Skip if Sign of A is Minus



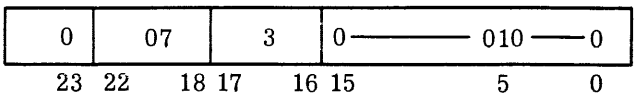
Skip the next instruction if the sign bit of the A register is minus. The contents of the A register are not changed.

SQM
Skip if Sign of Q is Minus



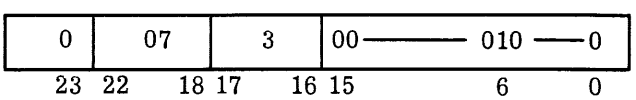
Skip the next instruction if the sign bit of the Q register is minus. The contents of the Q register remain unchanged.

SAZ
Skip if Contents of A is Zero



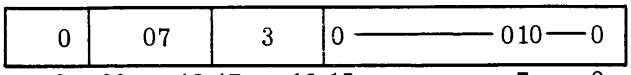
Skip the next instruction if the contents of the A register is zero. The contents of the A register remain unchanged.

SMO
Skip if Memory Range Overflow



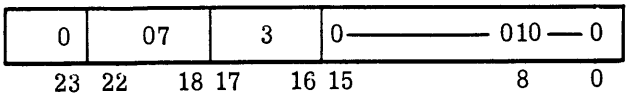
Skip the next instruction if the memory range overflow indicator is "ON". The execution of this instruction turns the indicator "OFF".

SMS
Skip if Master Lockout not Set



Skip the next instruction if the master lockout is not in effect. The execution of this instruction does not change the state of the master lockout.

HEY
Turn on Program Flag Light



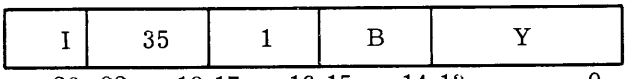
The Program Flag light on the operation panel is turned on to gain the operators attention. The light can only be turned off by depressing the light.

(9) Jump Instructions (JMP, JSR)

JMP AP
Jump on A Plus

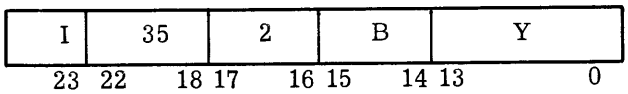
If the sign of the A register is plus, the next instruction is taken from location Y and execution proceeds from there; otherwise the next instruction is executed in sequence. Indexing and indirect addressing apply in all combinations. Timing: 1 cycle.

JMP AM
Jump on A Minus



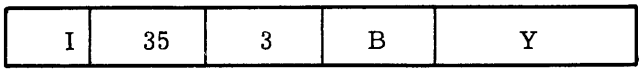
If the sign of the A register is minus the next instruction is taken from location Y and execution proceeds from there; otherwise the next instruction is executed in sequence. Indexing and indirect addressing apply in all combinations. Timing: 1 cycle.

JMP AZ
Jump on A Zero



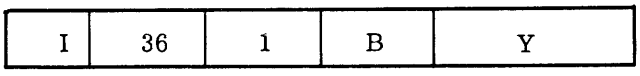
If the contents of the A register is zero the next instruction is taken from location Y and execution proceeds from there; otherwise the next instruction is executed in sequence. Indexing and indirect addressing apply in all combinations. Timing: 1 cycle.

JMP QP
Jump on Q Position



If the sign of the Q register is plus, the next instruction is taken from location Y and execution proceeds from there. Otherwise the next instruction is executed in sequence. Timing: 1 cycle.

JMP S1
Jump on Switch 1 on

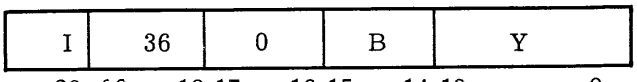


If sense switch 1 is up (in the 'on' position) the next instruction is taken from location Y and execution proceeds from there; otherwise the next instruction is executed in sequence. Indexing and indirect addressing apply in all combinations. Timing: 1 cycle.

JMP S2
Jump on Switch 2 on

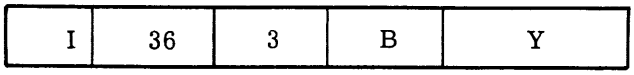
If sense switch 2 is up (in the "on" position) the next instruction is taken from location Y and execution proceeds from there; otherwise the next instruction is executed in sequence. Indexing and indirect addressing apply in all combinations. Timing: 1 cycle.

JMP
Jump



Unconditional jump. The next instruction is taken from location Y and execution proceeds from there. Indexing and indirect addressing apply in all combinations. Timing: 1 cycle.

JMP CML
Jump and Clear Master Lockout

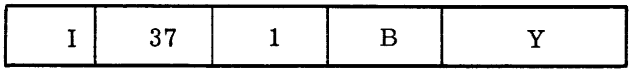


The master lockout is reset. See Input-Output Instructions for further details. The next instruction is taken from location Y and execution proceeds from there. The B field is ignored for this option. Indirect addressing applies. Timing: 1 cycle.

JSR
Jump and Set Return

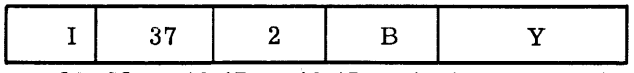
The location of this instruction plus 1 is stored in bits 0 to 13 of location Y. The remainder of that location is not changed. The next instruction is taken from location Y plus 1 and execution proceeds from there. Use of index register and indirect addressing apply in all combinations. No registers, indicators, or other memory locations are changed. Timing: 2 cycles.

JSR AM
Jump and Set Return if A Minus



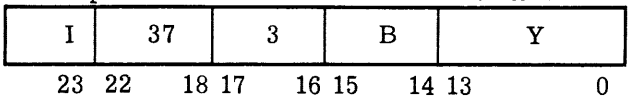
If the sign of the A register is minus, execution proceeds the same as JSR; otherwise the next instruction is executed in sequence. Timing: 1 or 2 cycles.

JSR AZ
Jump and Set Return if A Zero



If the contents of the A register are zero, execution proceeds the same as JSR; otherwise the next instruction is executed in sequence. Timing: 1 or 2 cycles.

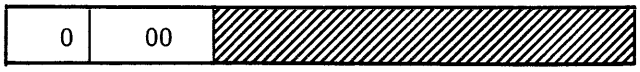
JSR SML
Jump and Set Return Set Master Lockout



The master lockout is turned on and execution proceeds the same as JSR. The B field is ignored for this option. Timing: 2 cycles.

(10) Misc. Instructions

FAULT
Fault



If sense switch No. 15 is in the HALT position, the sequence clock and real time clock are turned off. All computation and input-output processing is suspended and the FAULT indicator is lighted. The machine must be manually reset to reinstate operation. If sense switch 15 is up the machine does not stop and a FAULT interrupt occurs.

JHT
Jump and Halt

The sequence clock and real time clock are turned off. All computation and input-output processing is suspended and the RUN indicator is turned off. Computation resumes with a jump to location Y when the RUN button is depressed. Relative addressing via indexing is applicable. The jump address is displayed in the P register when the stop occurs.

NOP
No Operation

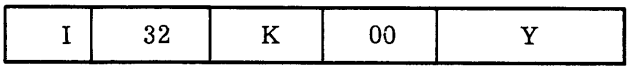
No registers, or memory locations are changed. The computer takes the instruction in sequence. The B and Y fields may be used but do not affect the execution of the instruction. Timing: 1 cycle.

INOP
Indirect NO Operation

This instruction is intended for use in connection with indefinite indirect addressing operations. When executed as a direct instruction,

no registers or memory locations are changed, however, a normal operand acquisition cycle is executed. If an index register is specified, an additional cycle time is lost. Timing: 2 cycles.

STZ
Store Zero



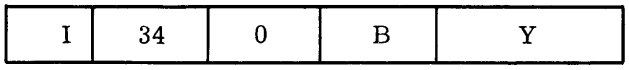
A full word of zeros is stored at location Y. Indirect addressing is applicable but no indexing is allowed. Timing: 2 cycles.

The options RH, LH, and Y apply as described under STA RH, STA LH, and STA Y, respectively, to specify the portion of a word to be zeroed.

NEG
Negate and Jump

The contents of the A register are negated (i.e., replaced by its 2's complement). Since 0 and the maximum negative number, -2^{23}, are their own 2's complement, the A register is unchanged for these. No overflow is possible. Indirect addressing is not applicable. Indexing is applicable. The next instruction is taken from location Y. Timing: 2 cycles.

XEQ
Execute

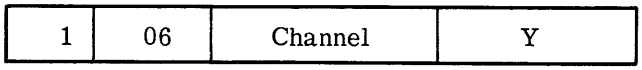


The instruction at location Y is executed. The program then returns to the next instruction in normal sequence. Indexing and indirect addressing apply. Timing: 1 cycle + normal timing of instruction executed.

(11) Input-Output Instructions

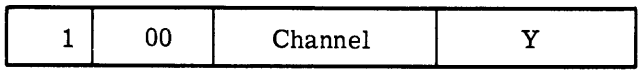
Details of input-output channels are contained in the next section. The mnemonics and operation of input-output instructions are given below. The timing indicated is for the instruction execution and does not include input-output data transfer unless specifically stated. Timing requirements for input-output devices are covered in section 4-5.

EXF
External Function



The contents of location Y are placed on the 24 output lines of the indicated channel in parallel. At the same time the external function control signal is sent to the indicated channel. Neither indirect addressing nor indexing is allowed. This instruction may be executed while a buffer is established. Timing: 2 cycles.

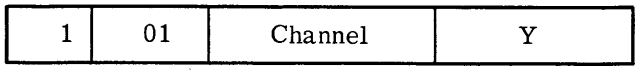
EXI
Express Input



The 24 input lines of the indicated channel are sensed in parallel and the resulting word is stored in location Y. Upon completion of the sensing, an input acknowledge signal is sent to the indicated channel. Timing: 2 cycles.

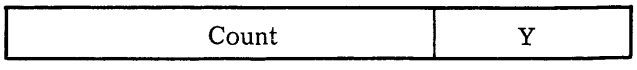
> Note: Depending on the input-output device there may be restrictions on the use of this instruction. See Input-Output Instructions and EXF instruction for further description.

EXO
Express Output



The contents of location Y are placed on the 24 output lines of the indicated channel in parallel. At the same time, the output acknowledge signal is sent to the indicated channel. Timing: 2 cycles.

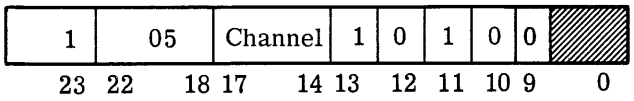
BCW
Buffer Control Word



This is not a machine instruction but rather a data word which specifies the first address of the buffer and the number of words to be transferred. This buffer control word must always follow an input or output buffer instruction. The 2's complement of the count must be placed in bits 23

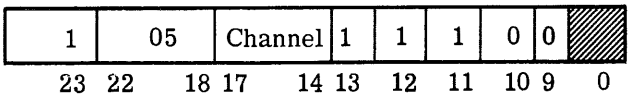
through 14, thus, the maximum count is 1024. The address (Y) portion (0 through 13) calls out the first location used for input-output. The successive locations involved in input-output proceed from the address in the Y field to the address in the Y field plus the count. The buffer will be terminated automatically after the count reaches zero. Upon termination, the special BCW address contains the following: Bits 23 to 14 are zero and the Y field equals the last location used for input or output.

BIN
Establish Buffer Input Mode



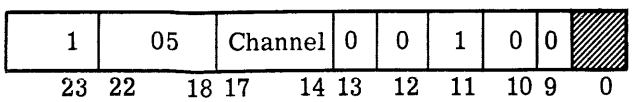
The contents of the next sequential location, the buffer control word (BCW), is placed in the buffer control word location as indicated in Assigned Core Memory Locations. The designated channel is then established in buffer input mode. No interrupt will occur when the buffer count has reached zero but further input request signals will be ignored. Neither indirect addressing nor indexing is applicable. No registers, indicators, or memory locations are changed, and no information is transmitted to or received from the indicated channel. The next location in sequence is not used as an instruction, and execution proceeds with the instruction in the following location. The input device on the indicated channel is activated in the appropriate mode by an external function (EXF) instruction. Timing: 3 cycles.

BINI
Establish Buffer Input Mode, Interrupt



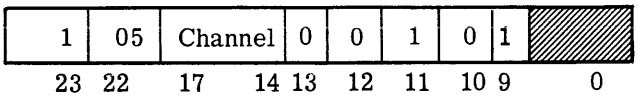
Same as BIN except an interrupt occurs forcing the instruction in the Buffer Finished Interrupt location of the designated channel to be executed when the buffer control word count has reached zero.

BOT
Establish Buffer Output Mode



The contents of the next sequential location, the buffer control word (BCW), is placed in the buffer control word location as indicated in Assigned Core Memory Locations. The designated channel is then established in buffer input mode. No interrupt will occur when the buffer count has reached zero, but further output request signals will be ignored. Neither indirect addressing nor indexing is applicable. No registers, indicators, or memory locations are changed, and no information is transmitted or received from the indicated channel. The next location in sequence is not used as an instruction and execution proceeds with the instruction in the following location. The output device on the indicated channel is activated in the appropriate mode by an external function (EXF) instruction. Timing: 3 cycles.

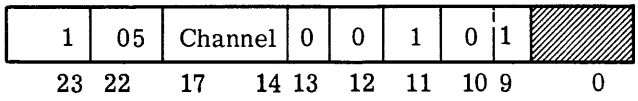
BOTB
Establish Block Buffer Output Mode



The contents of the next sequential location, the Buffer Control Word, is placed in the buffer control location as specified in the Assigned Memory Locations. The designated channel is then established in the block mode.

The buffer control word location is shipped and execution proceeds from the next succeeding address. Timing: 3 cycles.

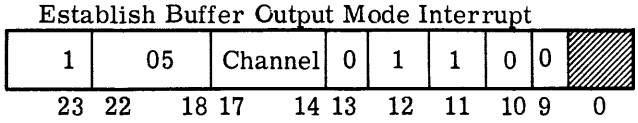
BINB
Establish Block Buffer Input Mode



The contents of the next sequential location, the Buffer Control Word, is placed in the buffer control location as specified in the Assigned Memory Locations. The designated channel is then established in the block mode.

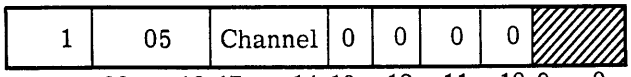
The buffer control word location is skipped and execution proceeds from the next succeeding address. Timing: 3 cycles.

BOTI
Establish Buffer Output Mode Interrupt



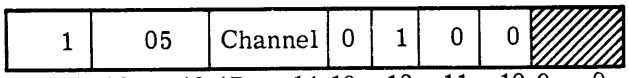
Same as BOT, except an interrupt occurs forcing the instruction in the buffer finished interrupt location to be executed.

BOMT
Output Buffer Mode Termination



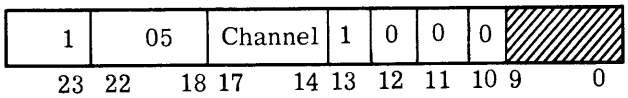
The output buffer on the designated channel is terminated without an interrupt occurring. The next instruction is skipped and execution proceeds from the following location. The buffer control word in the fixed location is not changed. Timing: 2 cycles.

BOMTI
Output Buffer Mode Terminate and Interrupt



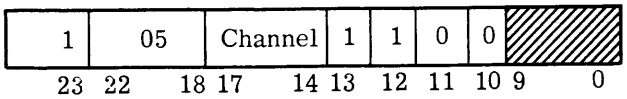
The output buffer mode on the designated channel is terminated and an interrupt results. A return after the interrupt will skip the next instruction. The buffer control word in the fixed location is not changed. Timing: 2 cycles.

BIMT
Input Buffer Mode Termination



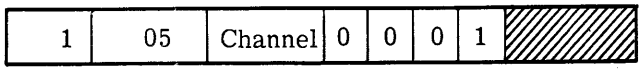
The input buffer on the designated channel is terminated without an interrupt occurring. The next instruction is skipped and execution proceeds from the following location. The buffer control word in the fixed location is not changed. Timing: 2 cycles.

BIMTI
Input Buffer Mode Terminate and Interrupt



The input buffer on the designated channel is terminated and an interrupt results. A return after the interrupt will skip the next instruction. The buffer control word in the fixed location is not changed. Timing: 2 cycles.

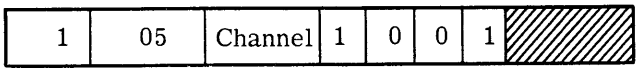
SOBS
Store Output Buffer Status



The status of eight output channels is stored in a fixed location as indicated in Figure 1-1-D. If any output channel 0 to 7 is specified the status of all is stored in location 00031. If any output channel 8 to 15 is specified the status of all is stored in location 00033. The first (upper) bit is a 1 if the block buffer mode is set (middle).

The second (middle) bit is 1 if the channel is to interrupt upon completion of the buffer (IR). The third bit is 1 if the buffer mode is active (BA). The next instruction is taken in sequence. Neither indirect addressing nor indexing is applicable. No registers, indicators or other memory locations are changed. Timing: 2 cycles.

SIBS
Store Input Buffer Status

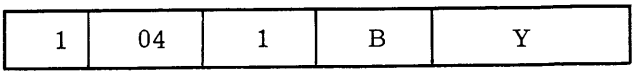


The status of eight input channels is stored in a fixed location as indicated in Figure 1-1-D. If any input channel 0 to 7 is specified the status of all is stored in fixed location 00030. If any input channel 8 to 15 is specified the status of all is stored in location 00032. See SOBS for further description.

CIR
Clear Interrupt Request

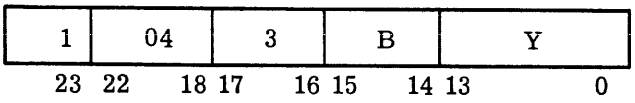
The interrupt request state is cleared on all channels indicated by a 1 in the contents of location Y. See Input-Output Instructions, for further description. Indexing is allowed but indirect addressing is not applicable. Timing: 2 cycles.

SAL
Set Additional Lockouts



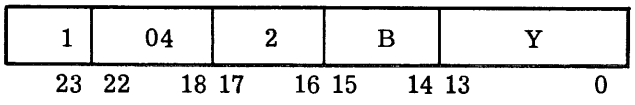
For each bit in location Y that is a 1 the lockout state is set. Previously locked out channels remain locked out. See CIR instruction for further description.

SLS
Store Lockout Status



The state of the lockouts on all channels is stored in location Y. Bit positions of channels in lockout status are set to 1 and all other bits are set to 0. See CIR instruction for further description.

ELS
Enter Lockout Status



The channels corresponding to the bits of location Y are set to lockout status for 1's and reset for 0's. See CIR instruction for further description.

i. Writing and Interpreting Instructions in Machine Code. - As the various designators of the instructions do not fall conveniently into octal groupings it is often necessary to make a mental calculation when writing or interpreting a machine coded program. For convenience, these programs are written in octal form, i.e., each instruction is eight octal digits. Figures 1-1-E and 1-1-F have been compiled to assist in interpreting the various designators directly from the machine code. In figure 1-1-E all of the various instructions are listed with the corresponding 6 bit function code. The various options referred to in the "interpretation" column (S, L, M, J, K, L', L and T) correspond to tables shown on Figure 1-1-F.

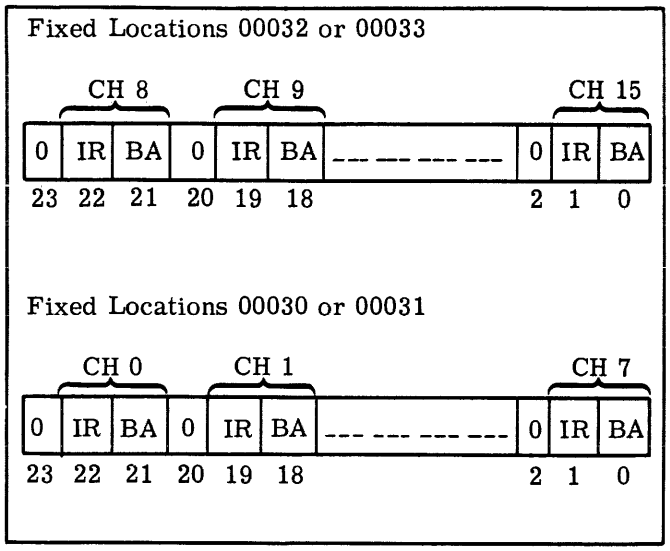


Figure 1-1-D. Buffer Status Word

j. Operator and Maintenance Panels. - The 2402 Computer has two front panels. The operator panel (see Fig. 1-1-G) at the top of the cabinet is always visible. The maintenance panel is located below the operator panel. See Figure 1-1-H. A complete set of displays and controls are available on these panels to allow convenient operation and maintenance.

Section 3 of this manual describes the function of the switches on the operation panel while Section 5 explains the maintenance switches and indicators.

ff	Interpretation
(01)	Shift as spec. by S
02	Not assigned: NOP
(03)	B^b-1→B^b; Jump as spec. by L
(04)	Sense(Y); Skip as spec. by L
(05)	Shift(Y) left one, skip spec. by L
(06)	(Y)-1→Y; skip as spec. by L
(07)	Skip next instr. as spec. by K'
10	L[(Y)(A)]→A; $(Q)_i = (Q)_f$
11	Set A_n for $Y_n = 1$
12	Clear A_n for $Y_n = 1$
13	Clear A_n for Logical Sum (A) & (Y) = 0
14	$Y_n \rightarrow A_n$ for $Q_n = 1$
(15)	(A)-L[(Y)(Q)]; Skip as spec. by L
16	If (A)< (Y) skip
(17)	(A)-(Y); skip as spec. by L
20	(Y)→Q; $(A)_i = (A)_f$ NOTE 1
21	(Y)→A; $(Q)_i = (Q)_f$ NOTE 1
(22)	(Y)→B^b
23	(A) + (Y) → Y
24	(A) + (Y) →A
25	(A) - (Y) →A
26	(Q)X(Y) →AQ
27	(AQ)/(Y)→Q; Rem. → A
30	(Q) →Y
31	(A) →Y
32	(B^b) →Y if b=0, clear Y
33	(A)-(Y) →Y
(34)	Execute instruction at address Y if K=0, NOP if K=3
(35)	Sense(A); Jump to Y as spec. by L
(36)	Jump to Y as spec. by J
(37)	(P)→Y, Jump to Y+1 spec. by R

ff	Interpretation
(40)	INPUT Ch. C → Y, send ACK
(41)	OUTPUT (Y) → Ch. C, send ACK
42	Not assigned: NOP
43	(Y) + 1 → Y
(44)	Set Interrupt Status as spec. by L'
(45)	Buffer as spec. by T on Ch. C
(46)	(Y)→OUTPUT LINES; Set EXF contr. line
(47)	Oper. as spec. by J'; Jmp to Y
50	Indirect 10
51	Indirect 11
52	Indirect 12
53	Indirect 13
54	Indirect 14
(55)	Indirect 15
56	Indirect 16
(57)	Indirect 17
60	Indirect 20
61	Indirect 21
(62)	Indirect 22
63	Indirect 23
64	Indirect 24
65	Indirect 25
66	Indirect 26
67	Indirect 27
70	Indirect 30
71	Indirect 31
72	Indirect 32
73	Indirect 33
(74)	Indirect 34
(75)	Indirect 35
(76)	Indirect 36
(77)	Indirect 37

Legend
NOTE 1: For K=3 and B≠0 the address portion of B_b is the operand.
(Q): contents of Q
(A): contents of A
(Y): contents of address Y
L[(Y)(Q)]: Logical product of $\bar{Y}$&Q
A_n: selected bits of A
B^b: B register spec. by the b designator
→: Transfer of Data
A: A register
Y: lower 14 bits of the instruction word
(AQ): contents of A and Q registers as one 48 bit register
Ch. C: channel spec. by the C designator
NOP; No operation
EXF; External Function Control Line
ACK; Acknowledge
Rem: Remainder
( ): Instrs which have special interpretation of bits 16&17 (non-standard K field)
⌒: Instrs which do not permit normal indexing

Figure 1-1-E. Westinghouse DPS-2402 Computer Repertoire of Instructions

LEGEND	NORMAL K & B DESIGNATORS		C DESIGNATOR
X indicates octal digit to position the designator (XXX) indicates bits within an octal digit	K (operand source, dest.) XX(00X) Full 24 bit word XX(01X) Right Half XX(10X) Left Half XX(11X) Use Y field*	B (index modifier) XX(XX0)(0XX) No Indexing XX(XX0)(1XX) Use B1 XX(XX1)(0XX) Use B2 XX(XX1)(1XX) Use B3	XXC(CXX)

* Not permitted for store or enter B instrs

BIT POS	L TABLE	L' TABLE**	J TABLE	J' TABLE	R TABLE	BIT POS
XX(00X)	Jmp/skip if +	CLR INT Request on Y	Unconditional	Complement (A)	Unconditional	XX(00X)
XX(01X)	Jmp/skip if -	Set Lockouts on Y	If SS #1 set	Interchange (A)&(Q)	Jmp if (A) -	XX(01X)
XX(10X)	Jmp/skip if = 0	Set Lockout Status Y	If SS #2 set	Uncond jmp, half at Y	Jmp if (A) = 0	XX(10X)
XX(11X)	Jmp/skip if ±0	Store Lockout Status at Y	Uncond Clear Mstr Lockout	NO OPeration	Uncond, set Master Lockout	XX(11X)

** Bit Nos. are chan. numbers

SPECIAL INSTR'S

S TABLE (01 Instruction)

BIT POS	INTERPRETATION	BIT POS	INTERPRETATION	BIT POS	INTERPRETATION
010(0XX)	Scale A, count →Y	013(0XX)(1XX)	Logical Right Shift Q closed	015(1XX)(0XX)	Arith Left Shift A closed
010(1XX)	Scale AQ, count →Y	013(1XX)(0XX)	Arith Left Shift Q closed	015(1XX)(1XX)	Arith Right Shift A closed
011(0XX)	Reverse Bits in A&Q	013(1XX)(1XX)	Arith Right Shift Q closed	016(0XX)(0XX)	Logical Left Shift AQ open
011(1XX)	No of 1's in A →Y	014(0XX)(0XX)	Logical Left Shift A open	016(0XX)(1XX)	Logical Right Shift AQ open
012(0XX)(0XX)	Logical Left Shft Q open	014(0XX)(1XX)	Logical Right Shift A open	016(1XX)(0XX)	Arith Left Shift AQ open
012(0XX)(1XX)	Logical Right Shft Q open	014(1XX)(0XX)	Arith Left Shift A open	016(1XX)(1XX)	Arith Right Shift AQ open
012(1XX)(0XX)	Arith Left Shft Q open	014(1XX)(1XX)	Arith Right Shft A open	017(0XX)(0XX)	Logical Left Shft AQ closed
012(1XX)(1XX)	Arith Right Shft Q open	015(0XX)(0XX)	Logical Left Shft A closed	017(0XX)(1XX)	Logical Right Shft AQ closed
013(0XX)(0XX)	Logical Left Shft Q closed	015(0XX)(1XX)	Logical Right Shft A closed	017(1XX)(0XX)	Arith Left Shft AQ closed
				017(1XX)(1XX)	Arith Right Shft AQ closed

Figure 1-1-F. Option Tables (Sheet 1 of 2)

T TABLE (INSTR 45)	
45C(C00)(000)	Terminate Output Buffer
45C(C10)(000)	Terminate Input Buffer
45C(C01)(001)	Terminate Output Buffer, Interrupt
45C(C11)(000)	Terminate Input Buffer, Interrupt
45C(C00)(010)	Store Output Buffer Status at 0033
45C(C10)(010)	Store Input Buffer Status at 0032
45C(C00)(100)	Estab. Output Buffer
45C(C10)(100)	Estab. Input Buffer
45C(C01)(100)	Estab. Output Buffer with monitor
45C(C11)(100)	Estab. Input Buffer with monitor
45C(C00)(101)	Estab. Output Buffer, Block mode
45C(C10)(101)	Estab. Input Buffer, Block mode
45C(C01)(101)	Estab. Output Buffer/w mon., Block mode
45C(C11)(101)	Estab. Input Buffer w/mon., Block mode

K' TABLE (INSTR 07)	
07(00X)	Skip if Y_n = any "on" sense switch
07(01X)	Skip if Input Data Request on any ch. spec. by Y_n
07(10X)	Skip if Output Data Request on any ch. spec. by Y_n
07(11X)	Skip if Y equals

0001 and OVERFLOW INDICATOR ON
0002 and DIVISION ERROR INDICATOR ON
0004 and PARITY INDICATOR ON
0010 and (A) negative
0020 and (Q) negative
0040 and (A) = 0
0100 and MEMORY OVERFLOW INDICATOR ON
0400 and MASTER LOCKOUT NOT SET
1000 - Do not skip; TURN ON PROG. FLAG

Figure 1-1-F. Option Tables (Sheet 2 of 2)

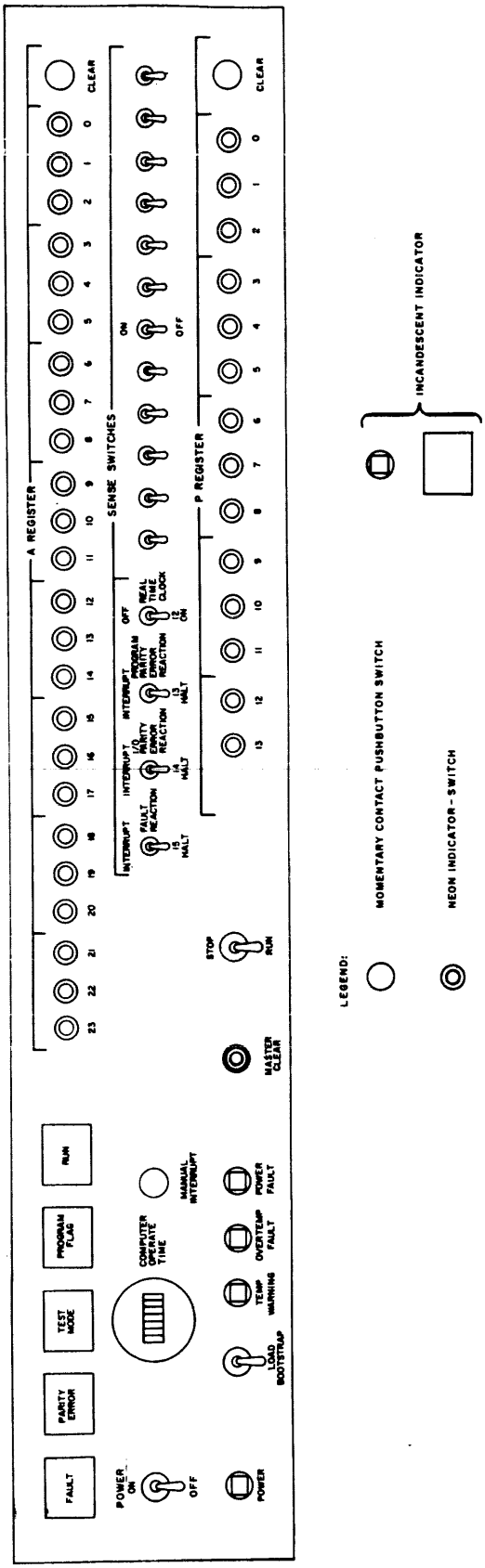


Figure 1-1-G. Operator's Panel

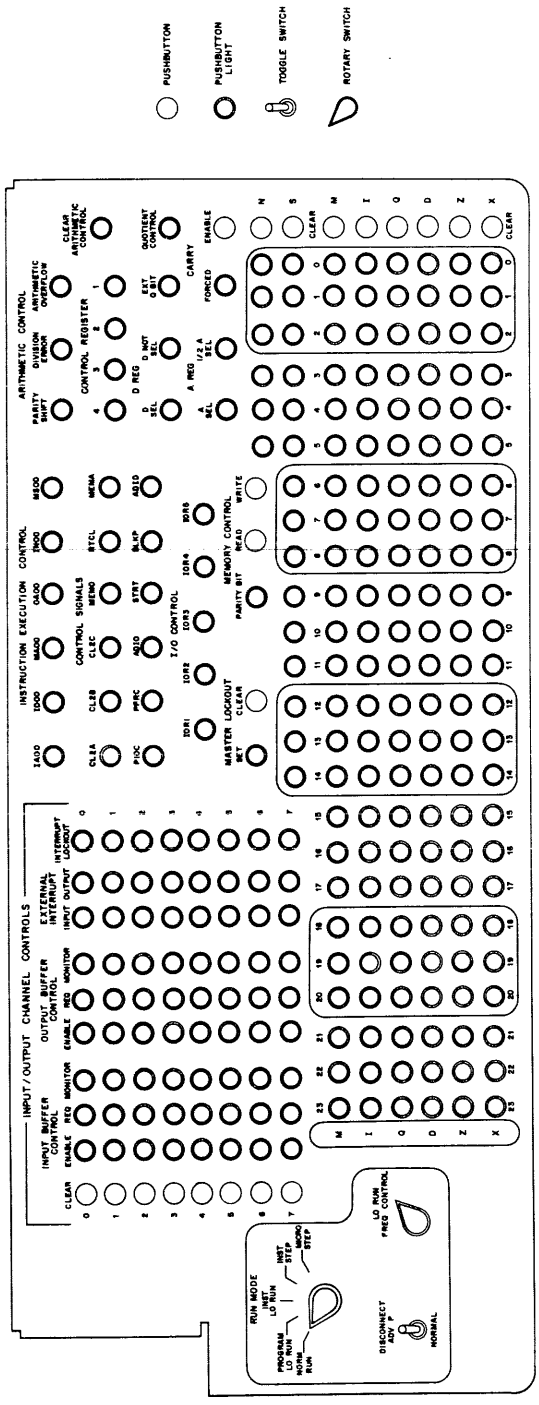


Figure 1-1-H. Maintenance Panel

2. INSTALLATION

This section contains unpacking, installation and cabling information for the computer.

2-1. UNPACKING PROCEDURE

The computer is shipped in a single* container ( ) with all modules in place and ready for operation.

Important: Lift computer only by the two lifting pads provided on the upper portion of the computer.

The computer, after being unpacked, should be visually examined for damage. Report any damages immediately to the shipping company. Open the computer door, unbolt and swing out the module chassis (A1 and A2). Check for any printed circuits modules which may have been loosened during shipping.

2-2. MOUNTING.

The computer should be placed on level, even flooring of at least 170 lbs/sq ft loading capacity. A mounting bracket is provided on the rear of the computer cabinet. Input/output and power cables are located on the side of the cabinet.

2-3. REMOTE OPERATION CONSOLE CONNECTION.

The remote operation console is connected by three cables to the computer. These cables, labelled J1, J2 and J3 are connected to the back of the computer just above the mounting bracket.

2-4. POWER INPUT CONNECTION.

A three wire (115 VAC, AC return and ground) power input is required. The AC line should be protected by an external circuit breaker of 15 ampere capacity.

Before connecting the power cable to the computer, insure that the POWER switch on the operation panel is in the OFF position.

2-5. INSTALLATION PROCEDURE.

The following steps should be performed in sequence to install the computer:

a. Perform a visual inspection.

b. Ensure that all printed circuit cards are firmly in place.

c. Connect the remote operation console to the computer.

d. Connect the power cable.

e. Switch the COMPUTER POWER switch to the ON position.

f. Master clear the computer

g. Visually inspect the operator & mainteance panel. All indicator lights should be off except STRT, IAOO and CL2A.

2-6. OPERATING CHECKS.

Load and run the diagnostic program outlined in Section 10.

* When a remote operation console is applicable, it is shipped in a separate container.

3. OPERATOR'S SECTION

3-1. FUNCTION.

The operator's section describes the operation of each switch and indicator light located on the DPS 2402 computer's front panels. The switches and indicator lights constitutes a complete set of displays and controls to allow convenient operation and maintenance.

The operator's section also describes the start-up procedure, the instruction execution sequence and the bootstrap load sequence.

3-2. OPERATION PANEL.

The operation panel is located at the top of the DPS 2402 computer cabinet, and it is always visible. As an optional feature, the operation panel may be remotely located.

WARNING

The pushbutton-lights, switches, and knobs on the panel are always active; therefore, any action taken at the panel applies immediately, even if the computer is running. A good operating procedure is to always stop the machine before using any control except sense switches.

a. Operation Panel Description. - A picture of the Operation Panel is shown in Figure 3-2-A

FAULT

The FAULT LIGHT is illuminated when the program executes an instruction with 6 or more high-order bits which are zero.

The FAULT light is extinguished by depressing the light.

PARITY ERROR

The PARITY ERROR light is illuminated when a word read from core storage has an incorrect parity.

The PARITY ERROR light is extinguished by depressing the light.

TEST MODE

The TEST MODE light remains illuminated as long as certain switches on the maintenance panel are in test, rather than normal, status.

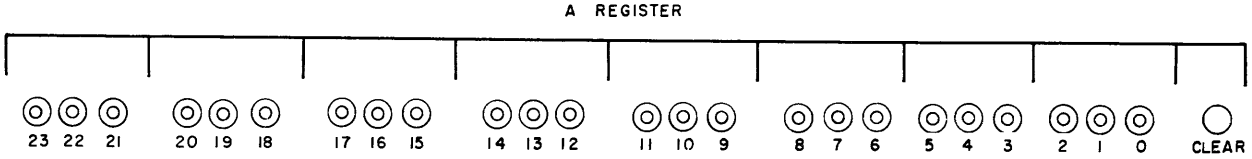
PROGRAM FLAG

The PROGRAM FLAG light is illuminated by executing the Program Flag, HEY, instruction.

The PROGRAM FLAG light is extinguished by depressing the light.

RUN

The RUN light is illuminated when the RUN switch is depressed to the RUN position.



The status of the A REGISTER is continuously displayed by 24 pushbutton lights. Depressing a pushbutton causes that bit of the A REGISTER to be set to a 1 which illuminates the light. The A REGISTER may be cleared and its indicator lights extinguished at any time by depressing the CLEAR pushbutton to the right of the register display.

The POWER switch is a two-position toggle switch. This switch controls the distribution of power to the cabinet. Up is the power ON position, and down is the power OFF position.

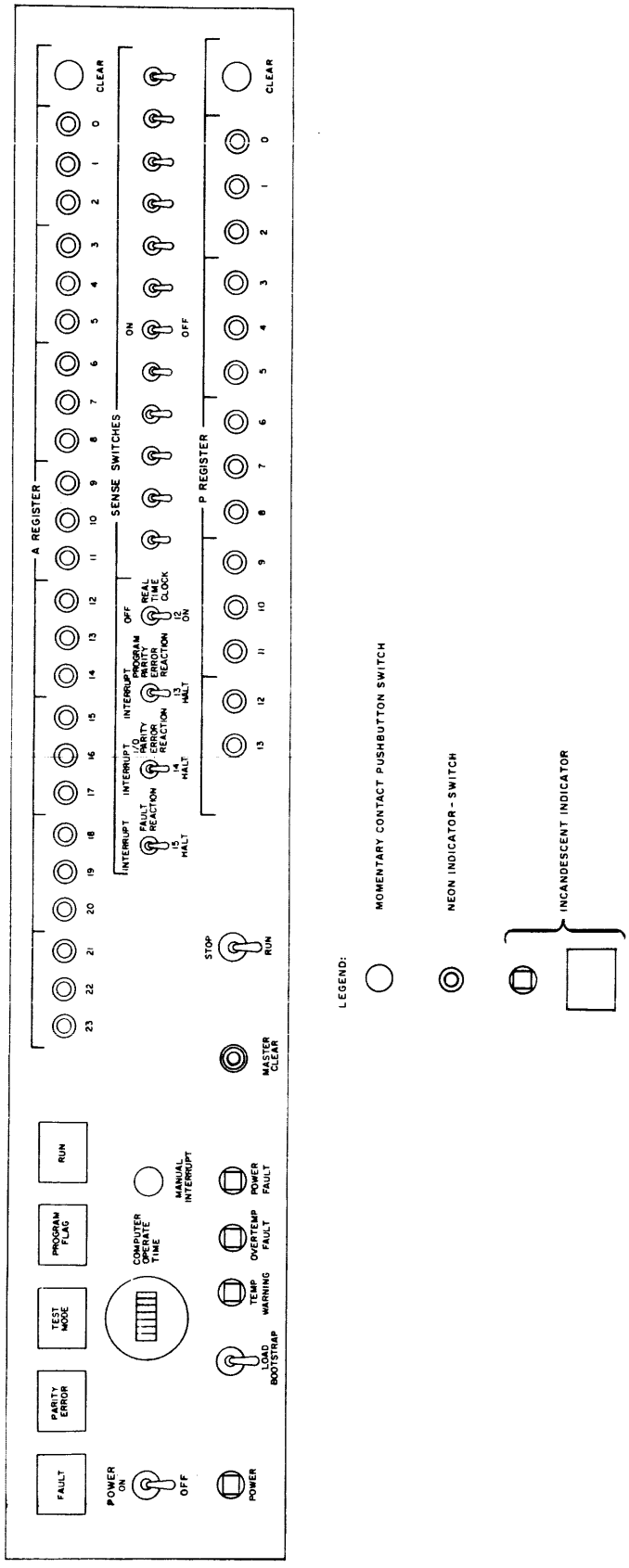
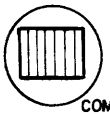


Figure 3-2-A. **Operator's Panel**



The COMPUTER OPERATE TIME meter reads the number of hours the d-c power has been ON.

Depressing the MANUAL INTERRUPT pushbutton while the computer is running causes the instruction in location 00023 to be processed in interrupt mode. Depressing the MANUAL INTERRUPT pushbutton when the computer is not running may cause an interrupt to occur when the RUN switch is depressed.

The POWER light is illuminated whenever the power switch is ON.

The LOAD BOOTSTRAP switch is a spring-loaded pushbutton indicator switch. When depressed the bootstrap sequence is enabled.

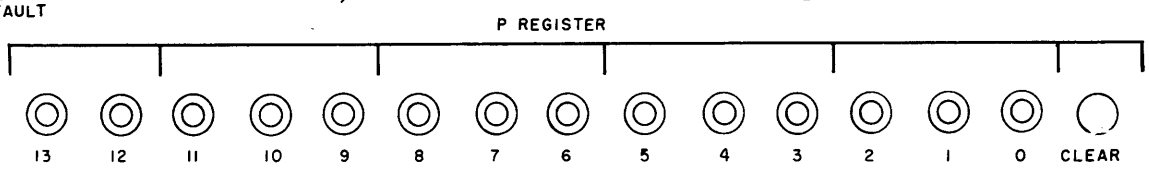
The TEMP WARNING light is illuminated whenever the cabinet temperature approaches the upper operating limits (65°C).

The OVERTEMP FAULT light when illuminated, indicates power has been shut OFF due to a cabinet temperature which exceeds the upper operating limit (70°C).

The POWER FAULT light when illuminated indicates power from the source has exceeded the allowable operating tolerances or else, a d-c power failure has occurred.

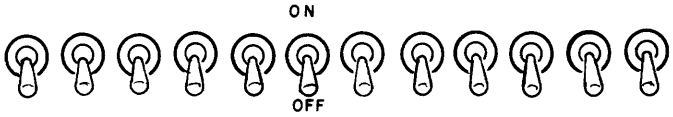
The MASTER CLEAR switch is a spring-loaded switch. When depressed, all registers are cleared and most of the flip-flops are set to the normal or inactive state. Core memory is not cleared or changed.

The RUN switch is a three-position spring-loaded to neutral toggle switch. When power is ON, depressing the RUN switch to the run position causes the computer to begin executing the instruction at the location indicated in the P-REGISTER. Raising the RUN switch to the stop position causes the computer to stop when the processing of the current instruction is completed.

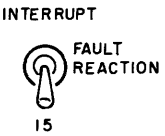


The status of the P REGISTER is continuously displayed by 14 combination pushbutton lights. Depressing a pushbutton causes that bit of the P REGISTER to be set to a 1 and illuminates the light. The entire P REGISTER is cleared and its indicator lights extinguished by depressing the CLEAR pushbutton to the right of the register display.

SENSE SWITCHES



The SENSE SWITCHES are two-position toggle switches. A switch in the up position is ON and will cause a skip when tested by a sense switch, SSW, instruction. A switch in the down position is OFF and does not cause a skip when tested.



SENSE SWITCH No. 15 is wired as a FAULT REACTION switch. If the switch is down, in HALT position, the computer stops when a FAULT instruction is executed. If the switch is up, in INTERRUPT position, the computer does not stop, and the instruction at location 00020 is processed in interrupt mode.

SENSE SWITCH No. 14 is wired as an I/O PARITY ERROR REACTION switch. If the switch is down, in HALT position, the

computer stops when a PARITY ERROR occurs while reading from core storage during input-output. If the switch is up in INTERRUPT position, the computer does not stop, instead the instruction at location 00021 is processed in interrupt mode.

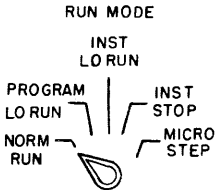
SENSE SWITCH No. 13 is wired as a PROGRAM PARITY ERROR REACTION switch. If the switch is down, in HALT position, the computer stops when a PARITY ERROR occurs while reading from core storage (except during input-output). If the switch is up, in INTERRUPT position, the computer does not stop, instead the instruction at location 00022 is processed in interrupt mode.

SENSE SWITCH No. 12 is wired as a REAL TIME CLOCK switch. If the switch is ON (down), the core location 00024 is incremented by one each 5 milliseconds. If the switch is OFF (up), no incrementation takes place.

3-3. MAINTENANCE PANEL.

The maintenance panel is located just below the operation panel and has a removable cover.

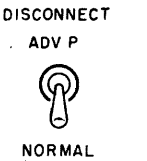
a. Maintenance Panel Description. - Figure 3-2-B shows the Westinghouse DPS-2402 Computer maintenance panel.



The RUN MODE selector switch allows the selection of five modes of execution. This allows maintenance personnel to perform most malfunction locations from the panel with the aid of diagnostic programs accompaning the computer. The RUN MODE selector should not be turned while the computer is running (Depress STOP switch on operation panel before turning selector). The TEST MODE light is illuminated when the selector is in a position other than NORMAL RUN.

1. The NORMAL RUN position is for normal operation.
2. The PROGRAM LO RUN position allows normal operation; except, when a halt instruction, JHT, is executed the computer stops for a given period of time. The duration of the stop is controlled by the LO RUN FREQUENCY CONTROL.
3. The INST LO RUN position allows normal operation, except that, after the completion of the execution of each instruction the computer stops for a period of time. The duration of the stop is controlled by the LO RUN FREQUENCY CONTROL.
4. The INST STEP position allows normal execution of one instruction for each depression of the RUN switch on the operation panel.
5. The MICRO STEP position allows execution of one micro step for each depression of the RUN switch on the operation panel. All instructions except those with a store cycle can be micro-stepped while maintaining the correct execution.

The LO RUN FREQ CONTROL allows the time the computer stops in RUN MODES 2 and 3 to be varied from 0.1 second to 2 seconds. Counter-clockwise rotation increases the delay.



The DISCONNECT ADV P switch is a two-position toggle switch. In the NORMAL position, the P-register is incremented during the execution of instructions, other than jump or skip types. In the DISCONNECT ADV P position, the advance of the P-register is suppressed and thus the same instruction is repeatedly executed for instructions other than jump or skip types. When in the DISCONNECT ADV P position, the TEST MODE light on the operation panel is illuminated.

The MASTER LOCKOUT status can be set by depressing the

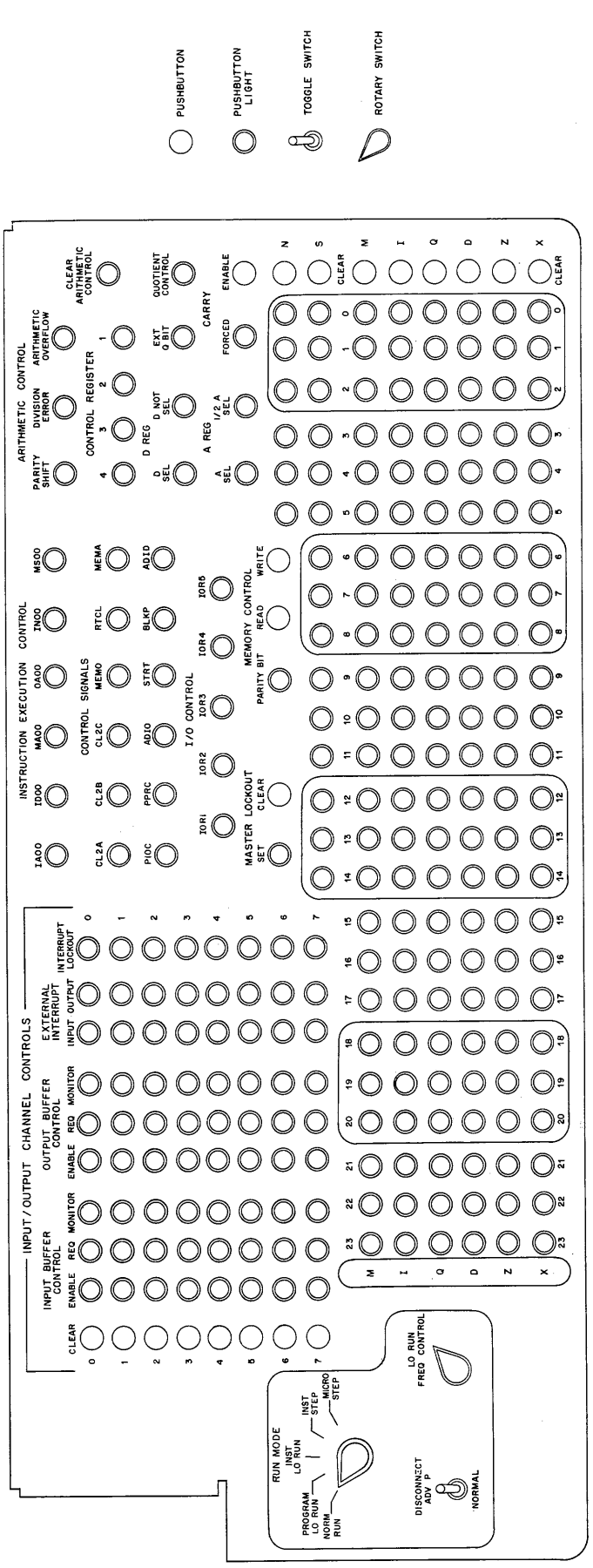


Figure 3-2-B. Maintenance Panel

MASTER LOCKOUT SET

CLEAR

pushbutton-light labeled SET and can be cleared by depressing the pushbutton-light labeled CLEAR. The SET pushbutton-light is illuminated when the MASTER LOCKOUT is in effect.

(1) Instruction Execution Control

IA00

An illuminated IA00 light indicates an instruction acquisition is in progress.

ID00

An illuminated ID00 light indicates indirect addressing is in progress.

MA00

An illuminated MA00 light indicates an address modification is in progress.

OA00

An illuminated OA00 light indicates an operand acqusition is in progress.

IN00

An illuminated IN00 light indicates an intermediate execution is in progress (not required by most instructions).

MS00

An illuminated MS00 light indicates a memory store is in progress.

(2) Control Signals

CL2A CL2B CL2C

The 3.0-megacycle clock is divided by 6 with a 3 stage counter to yield a 2 microsecond time interval. CL2A is illuminated at the beginning of each 2-microsecond cycle and remains illuminated for 1 microsecond. CL2B is illuminated 2/3 of a microsecond after CL2A, and CL2C is illuminated 2/3 of a microsecond after CL2B.

MEMO RTCL MEMA

The MEMO light is illuminated when the MEMORY RANGE OVERFLOW INDICATOR is set.

The RTCL light is illuminated when the real time clock is locked out by program.

The MEMA light is illuminated when memory access is available to input-output.

PIOC PPRC ADID

The PIOC light is illuminated while in the interrupt mode due to an input-output parity error.

The PPRC light is illuminated while in the interrupt mode due to a program parity error.

The ADI0 light is illuminated while transferring I_{0-13} to the S or P registers.

The BLKP light is illuminated during the IA cycle, when in interrupt mode, to suppress the advancing of the P-register.

STRT BLKP ADIC

The STRT light is illuminated when memory is either available for input-output or available for fetching the next instruction. The ADID light is illuminated when indexing.

(3) I/O Control

IOR1

The IOR1 light is illuminated during the fetching of a buffer control word on the Real Time Clock; when calling up the fixed location for storing the Interrupt Status Word while servicing on external interrupt; when selecting a channel during either an EXI, EXO, on EXF instruction; when fetching the Buffer Control word to be transferred to the buffer control register, and when establishing a buffer on the special high speed channel.

IOR3

The IOR3 light is illuminated when transferring data during an input or output buffer on a 24 bit channel; when storing the Interrupt Status Word on a 24 bit channel; when transferring data on a 24 bit channel during either a EXO, EXI or EXF instruction.

The IOR4 light is illuminated when storing the Buffer Control

IOR4

Word on the Real Time Clock; when storing the contents of the buffer control register; when doing a terminate buffer instruction on a special high speed channel; when calling the fixed location for the interrupt location; when doing an external interrupt; and when acknowledging a normal input or output buffer.

READ

Depressing the READ pushbutton causes the contents of the location specified by the S-register to be fetched into the M-register.

WRITE

Depressing the WRITE pushbutton causes the contents of the M-register to be stored at the location called out by the contents of the S-register. The PARITY BIT pushbutton light is illuminated when the last stored word has an even nunber of ONES.

(4) I/O Channel Control

CLEAR

CLEAR CHANNEL PUSHBUTTON - depressing a CLEAR channel pushbutton resets all channel control flip-flops and extinguishes all indicator lights on the respective channel.

ENABLE

BUFFER ENABLE - depressing the ENABLE pushbutton light causes the indicator light to illuminate and sets the enable buffer flip-flop. The illuminated light indicates a buffer is established on the channel.

REQ

DATA REQUEST - depressing the REQ pushbutton light causes the indicator light to illuminate and sets the data request flip-flop. The illuminated light indicates there is data available on the line for the input channel and the peripheral device is ready for data on the output channel.

MONITOR

MONITOR - depressing the MONITOR pushbutton light causes the indicator light to illuminate and sets the monitor flip-flop. The illuminated light indicates that an interrupt will occur when the buffer is terminated.

INTERRUPT REQUEST

INPUT OUTPUT

INTERRUPT REQUEST - depressing a Interrupt REQUEST pushbutton light causes the indicator light to illuminate and sets the interrupt request state. The illuminated light indicates the peripheral device wants to interrupt the computer. If it is an input interrupt request the input data lines contain an interrupt status word.

INTERRUPT LOCKOUT - depressing the INTERRUPT LOCKOUT pushbutton light causes the indicator light to illuminate and sets the interrupt lockout flip-flop. The illuminated light indicates no interrupts can occur on that channel.

(5) Arithmetic Control

ARITHMETIC CONTROL

PARITY SHIFT DIVISION ERROR ARITH OVERFLOW

The PARITY SHIFT light is illuminated whenever the PARITY SHIFT indicator is set. The DIVISION ERROR light is illuminated whenever the DIVISION ERROR indicator is set. The ARITHMETIC OVERFLOW light is illuminated whenever the ARITHMETIC OVERFLOW indicator is set.

CONTROL REGISTER

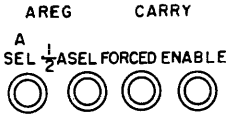
4 3 2 1

The four pushbutton-lights of the ARITHMETIC CONTROL register are illuminated as described below: (1 is illuminated, 0 is extinguished).

PHASE	CONTROL REGISTER 4321	
A	0000	During all one-cycle manipulations
B	1000	During scale A
L	0100	During division
P	1100	During division
C	0010	During shifting
E	1010	During scale AQ

PHASE	CONTROL REGISTER 4321	
M	0110	During division
N	1110	During division
D	0001	During division and negate A
Q	1001	During division
R	0101	During division
S	1101	During division
F	0011	During reverse bits of A and Q
K	1011	During multiply
G	0111	During reverse bits of A and Q
J	1111	During reverse bits of A and Q

is the extra bit of the Q-register used in multiplication. The QUOTIENT CONTROL pushbutton when illuminated indicates that the partial remainder has equalled zero during the division process.

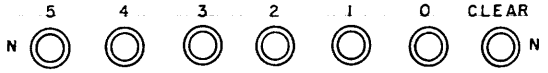


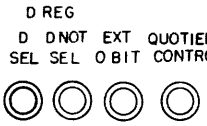
The A SEL pushbutton light is illuminated when the (A) is selected to be gated into the adder.

The 1/2A SEL pushbutton light is illuminated when the contents of the A-register, shifted right one bit are gated into the adder.

The FORCED pushbutton-light is illuminated when a carry is forced into the low-order stage of the adder.

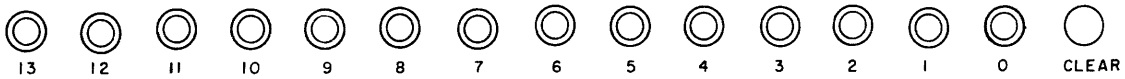
The ENABLE pushbotton-light is illuminated during airhtmetic operations requiring use of the adder carry circuits.



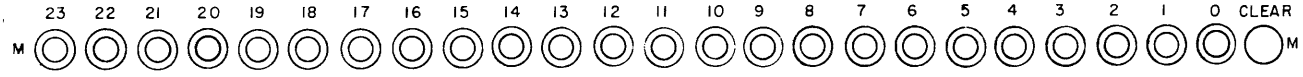


The D SEL pushbutton light is illuminated when (D) is selected to be gated into the adder. The D NOT SEL light is illuminated when the complement of (D) is selected to be gated into the adder. The EXT Q BIT pushbotton light illuminates the light. The register can be cleared to all zeros by depressing the CLEAR pushbutton.

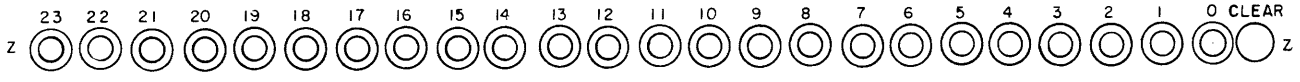
The N register is continuously displayed by 6 pushbutton-lights. Depressing a pushbutton causes that bit of the N register to be set which



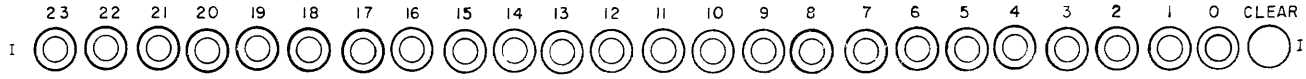
The S register is continuously displayed by 14 pushbutton-lights. Depressing a pushbutton causes that bit of the S register to be set which illuminates the light. The register can be cleared to all zeros by depressing the CLEAR pushbutton.



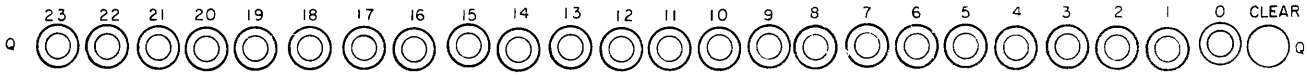
The M register is continuously displayed by 24 pushbutton-lights. Depressing a pushbutton causes that bit of the register to be set which illuminates the light. The register can be cleared to all zeros by depressing the CLEAR pushbutton



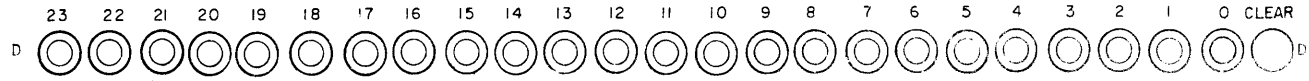
The Z register is continuously displayed by 24 pushbutton-lights. Depressing a pushbutton causes that bit of the Z register to be set which illuminates the light. The register can be cleared to all zeros by depressing the CLEAR pushbutton.



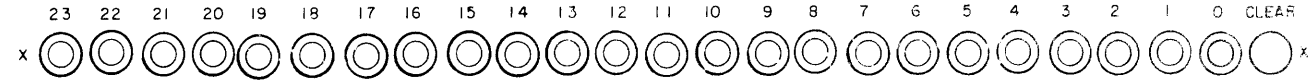
The I register is continuously displayed by 24 pushbutton-lights. Depressing a pushbutton causes that bit of the I register to be set which illuminates the light. The register can be cleared to all zeros by depressing the CLEAR pushbutton.



The Q register is continuously displayed by 24 pushbutton-lights. Depressing a pushbutton causes that bit of the Q register to be set which illuminates the light. The register can be cleared to all zeros by depressing the CLEAR pushbutton.



The D register is continuously displayed by 24 pushbutton-lights. Depressing a pushbutton causes that bit of the D register to be set which illuminates the light. The register can be cleared to all zeros by depressing the CLEAR pushbutton.



The X register is continuously displayed by 24 pushbutton-lights. Depressing a pushbutton caused that bit of the X register to be set which illuminates the light. The register can be cleared to all zeros by depressing the CLEAR pushbutton.

3-4. FUSE PANEL.

The fuse panel is located at the lower right corner of the cabinet and shares the maintenance panels removable cover.

3-5. OPERATION

a. Start-Up Procedure. - The following procedure should be used to turn the computer ON:

Step 1. Place the COMPUTER POWER switch on the Operating Panel to the ON position.

Step 2. Ensure that all switches are in the desired operating position.

Step 3. Depress the MASTER CLEAR switch.

Step 4. If the program to be run is in memory, set the starting address into the P register. If the program is not in memory, it may be loaded by use of a load routine or the bootstrap load sequence. The bootstrap load sequence is referenced by depressing the LOAD BOOTSTRAP switch on the Operating Panel. (For details see Bootstrap Load Sequence Section 3-4(2).

Step 5. Depress the RUN switch.

The computer is now in the run condition and the green RUN indicator is illuminated. The instructions are executed consecutively. If an instruction is not a jump or skip type, the P register is incremented by one for each instruction. Thus, the normal arrangement of instructions is placed in ascending sequence in core storage.

b. Bootstrap Load Sequence. - The COMPUTE POWER switches must be in the ON position. No error or warning indicators should be on. Status of the machine is cleared via depressing the MASTER CLEAR switch. The power must be ON for the input device on channel zero, and the device should be readied for use manually. The LOAD BOOTSTRAP switch is depressed and released; then the RUN switch is depressed to

initiate the bootstrap load sequence. The following procedure should be used to initiate the bootstrap load sequence.

Step 1. Place the COMPUTER POWER switch on the operating panel to the ON position.

Step 2. Ensure that all switches are in the desired operation position.

Step 3. Depress the MASTER CLEAR SWITCH.

Step 4. Ensure the paper tape reader is loaded and ready for loading the computer in accordance with the paper tape readers operating instructions.

Step 5. Depress and then release the LOAD BOOTSTRAP switch.

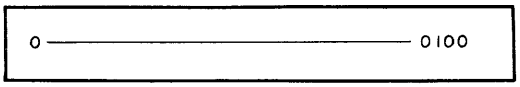
Step 6. Depress the run switch.

The LOAD BOOTSTRAP light indicator is illuminated and the computer is operating in the following load bootstrap sequence (described for channel 0).

(1) An unconditional jump to 00000_8.

(2) A buffer control word is stored in location 00040. This word has a count of 1023 and an address of 00041_8.

(3) The external function signal is applied to channel 0 and the word

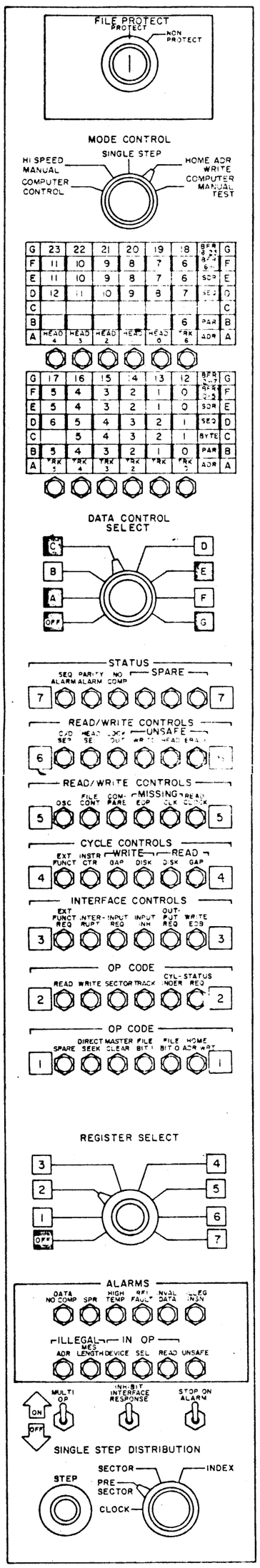


is applied to the output lines of channel 0.

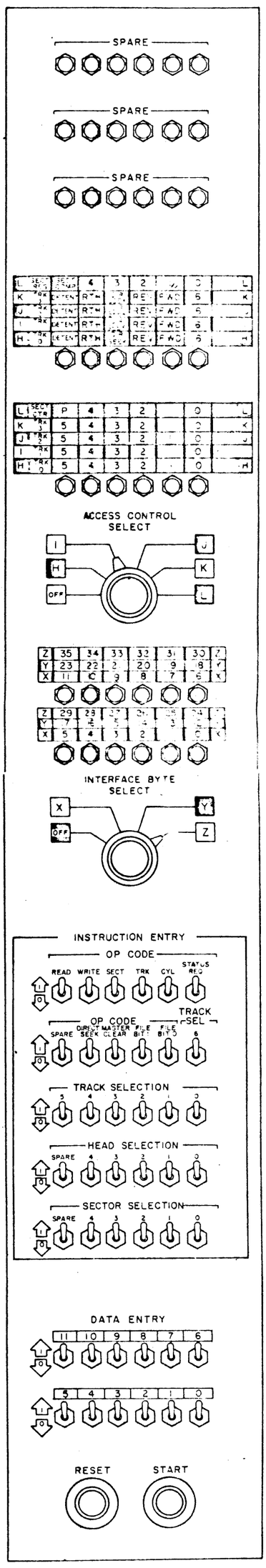
(4) A buffer input mode with interrupt upon termination is established for channel 0.

(5) Execution started at location 0 when the RUN switch was depressed.

(6) Execution proceeds via an interrupt to locating 00060 or 00120, depending on the type of bootstrap program being loaded. It is recommended that the real time clock be "OFF" and all reaction switches be set to "HALT" during the bootstrap load sequence.



SUBPANEL B5



SUBPANEL C5

Figure 3-1. Operator's Control Panel

N.B Fig 4-16 has cover removed.

COVER ON

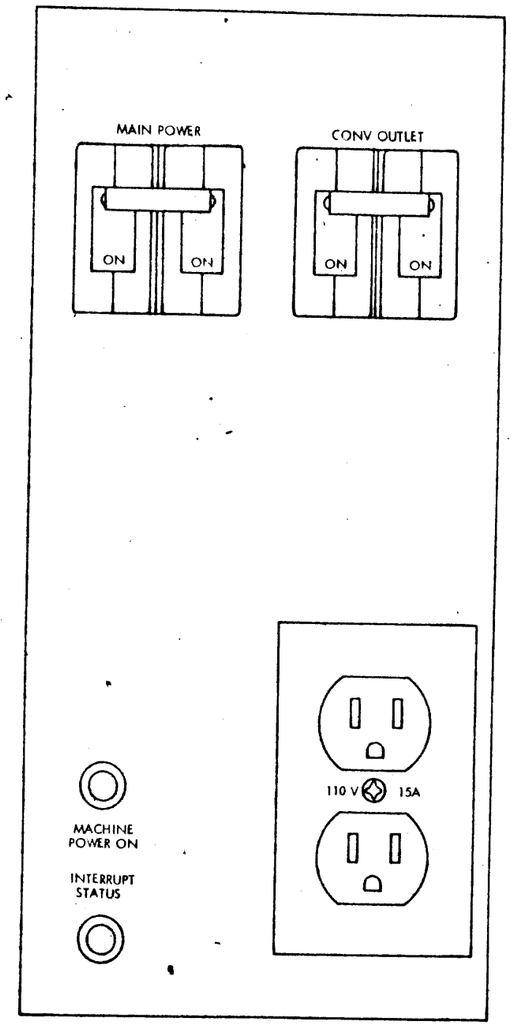


Figure 3-3. A-C Power Control Panel

4. PRINCIPLES OF OPERATION

This section describes the theory of operation of the DPS-2402 computer which includes operation of the various registers, translators, and sequential logic employed by the four major computer sections.

4-1. GENERAL INFORMATION.

This subsection contains general principles of operation of the DPS-2402 computer. The basic logic employed, data flow and interdependence of the major computer sections are discussed in general terms.

a. Description. - The computer consists of four sections: Memory, Control, Arithmetic, and Input/Output.

(1) Memory Section. - The memory section employs a medium capacity, magnetic core storage medium. The DPS-2402 Computer Memory is a high speed, random access, destructive readout, non-volatile storage system; i.e., its speed is compatible with the computer, its data words may be accessed in any order, and it will retain information when the power is removed.

(a) Registers. - The memory section contains the following registers:

M Register - a twenty-five bit register which serves as a memory buffer register. At the beginning of each access period, the M-register is cleared. During the read-access period, the contents of the desired memory location are loaded into the M-register. The contents of M then are written back into memory to compensate for the destructive readout phenomenon. The M-register controls the inhibit circuits to restore the information in the memory.

S Register - a memory address translator register. The S-register holds the storage address during memory references. The address is received from the input-output or control units at the beginning of a storage access period. The contents of the S-register then are decoded by the memory selection system.

(b) Auxiliary Circuits. - The circuits required for memory section are:

S Translator, and Drive Line Selection - circuits which operate from information contained in the S register and are used to select a particular X and Y drive lines for each address.

Wired Memory (Bootstrap) - a four instruction wired memory is used to provide a non-destructible load routine. This routine is referenced by the LOAD BOOTSTRAP switch on the Operators' Panel.

(c) Capacity. - The memory section contains 16,384 locations for storage of 25-bit words (24 bits for data or instructions and one for parity). Each 24-bit data word can, by proper programming, be treated as two twelve-bit words and referenced separately.

(d) Timing. - The time required for one complete memory reference or the basic cycle time (read-write cycle) is two microseconds, although information is available in the M register for use by the computer after about one microsecond. Timing is accomplished by a delay line. Initiation of a memory cycle may be accomplished by signals from the Control Section or Input/Output section, or by manual means from the maintenance panel.

(2) Control Section. - The control section of the computer is comprised of registers, modifying circuits, designator and timing circuits which function together to execute instructions and perform operations.

(a) Registers. - The following registers are located in the control section:

I Register - a 24-bit program control register. The I-register stores the current instruction word during its execution. This register acquires the instruction from memory through the M-register. In addition to serving as an input to the function code and designator translators, this register also inputs to the adder circuit, lower 14 bits, to provide modification of the operand address.

P Register - a 14-bit program address register. The address of the next instruction is stored in the P-register. This register is loaded through the adder from the I-register and also is incremented by 1, automatically, as the instructions are sequentially executed. Execution of a skip instruction causes the P-register to be incremented by an additional 1 if the respective conditions are met. The content of this register is changed only by manual means or by the execution of a jump instruction. A capability of storing the contents of the P-register also is included as a means of implementing the "jump and set return" (JSR) instruction.

(b) Modifying Circuits. - The following modifier circuits are located in the control section:

P Adder - an incrementing circuit for the program control register I translator, an instruction interpretation device.

(c) Timing Circuits. - The timing circuits of the control section are as follows. Each of these timing sequence flip flops may be set for two microseconds, and then cleared. Only one sequence flip flop may be in the set state at any given time.

IA (Instruction Acquisition) - a single flip flop in the major timing sequence which is set during the acquisition of the instruction from memory.

ID (Indirect Addressing) - a single flip flop in the major timing sequence which is set during indirect addressing.

MA (Modify Address) - single flip flop in the major timing sequence which is set during address modification (indexing).

OA (Operand Acquisition) - a single flip flop in the major timing sequence which is set during the process of obtaining the operand from memory.

IN (Intermediate) - a single flip flop in the major timing sequence which indicates that an intermediate arithmetic operation is taking place.

MS (Memory Store) - a single flip flop in the major timing sequence which is set when information from a working register is being stored in memory.

(3) Arithmetic Section. - The arithmetic section of the computer consists of the A, D, and Q data registers; two control registers, AC and AF; a 6-bit counter, N; and a 24-bit parallel adder. The adder combines the contents of the A and D registers. The purpose and function of the arithmetic section is to perform addition, subtraction, multiplication, division, scaling and shifting operations in addition to various operations in support of the control section as dictated by the instruction being executed.

(a) Registers. - The following registers are included in the arithmetic section.

A-Register - a 24-bit addressable accumulator. This accumulator, referred to as the A-register, is the principle arithmetic register. Inputs to this register are derived from the adder selection gates which provide parallel addition and shifting capability. With the exception of multiplication, all arithmetic operations call for one operand to be in this register prior to execution of that instruction.

After addition or subtraction, the A-register contains the sum or difference; after a multiplication, the most significant half of the product remains in the accumulator. Additionally, this register contains both the remainder after the execution of a divide and the number on which logical operations are performed.

The contents of the accumulator may be shifted either left or right, closed or open, as described by the shift operations.

Q-Register - a 24-bit addressable register. This register serves as the multiplier-quotient register. Prior to multiplication, the register contains the multiplier. During multiplication, the multiplier is shifted right, two positions at a time. The three least-significant bits are examined during each shift to determine if the multiplicand should be applied to the partial product. At the same time, the significant part of the partial product is shifted right into the

Q-register. At the completion of the high-speed multiplication process, facilitated by the two-bit look-ahead feature, the least significant half of the product is found in the Q-register.

The Q-register contains the least significant half of the dividend prior to the division process. This register is used to assemble and hold the quotient, and the sign of the quotient is found in the most significant bit position. Shifting of the Q-register contents is similar to that of the A-register. There are cases in which the A and Q register are shifted as a single 48-bit register.

D-Register - a 24-bit non-addressable register. The D-register is an intermediate register which contains the operand from memory (Y) while the sum, difference, product, or quotient is being formed. This register also contains the B-index modifier while it is being added to the address.

A second function of this register is that it serves to transfer data for instruction words to and from the memory unit and all of the arithmetic and control unit registers.

N-Register - a 6-bit non-addressable shift control register. This register is implemented in the form of a counter and is used to control the shifting during multiplication, division, and shift command execution.

To readily facilitate a programmed floating point, a "scale factor" instruction is included in the repertoire.

AC-Register - a four-bit arithmetic control register which generates the various "phases" used during execution of the various arithmetic instructions.

AF-Register - the AF-register is a series of flip flops which operate independently of one another to control the selection of adder inputs and to indicate the existence of arithmetic error conditions.

(4) Input/Output Section. - The input/output section serves as the communications link between the other sections of the computer and the external equipment or additional computers. All such communication is accomplished in a 24-bit parallel mode. The computer is provided with five input and five output channels. The I/O section possesses its own control functions which can operate essentially independent (asynchronous) of the primary instruction control. Special memory addresses are reserved for I/O operation, i.e., interrupt addresses for each channel, buffer status address, buffer control per channel, real time clock, etc.

(a) Registers. - The Input/Output Section contains two working registers, a status register, and associated modifying and gating logic.

X-Register - the 24-bit X-register is an input/output data buffer register which serves to compensate for timing differences between the computer and peripheral equipment. The X-register serves all channels. Channels are assigned for 24-bit parallel data transfer where one memory location is used for each transfer.

Z-Register - the 24-bit Z-register is used to update the buffer control word, during buffer operations, and to increment the real time clock address contents as called for in the RTC (update) interrupt mode.

I/O Status - the I/O Status Register is a static register which holds the current status of the Input/Output Section. This information includes the following for each channel:

Monitor Set
Channel Lockout Set
Input and/or Output Buffer Active

(b) Input Data Gated Amplifiers. - Data from the peripheral equipment is placed on the computer's input lines. This data is then gated directly into the X-register by special gated amplifiers. Refer to section 6 for an electronic description of these circuits.

(c) Input/Output Timing. - The Input/Output Section runs asynchronously or independently from the main program timing. This I/O timing which must share memory with the control section has a separate timing chain (sequencer) which governs the I/O operation. Refer to paragraph 4-5b of this section for a detailed discussion of Input/Output timing.

(5) Summary of Computer. - Data is transmitted from external equipment or computers into the computer and from the computer to the external equipment or computers through the input/output section. All data entering the computer is transferred into memory. Likewise, data transmitted from the computer through the input/output section must have been previously stored in memory. Data is transmitted from core storage to the appropriate output channel or to the arithmetic section as called for by the computer program. All arithmetic processes such as addition, subtraction, multiplication, and division are performed by the arithmetic section. The control section generates the necessary commands and timing pulses in order to assure the orderly flow of information within the computer and the subsequent execution of instructions. The core storage medium provides ready access by the control and Input/Output Sections to data, codes and instructions as well as providing storage for arithmetic results.

(6) Functional Schematics. - The functional schematics in Section 9 show the logical functions of the computer which comprise the primary instrument for describing the theory of operation as well as being the working document for corrective maintenance. This section will refer constantly to Section 9. The functional schematics in Section 9 are designated according to the section of the computer they describe. A letter suffix describes the particular computer section.

Figure 9A - Arithmetic Section Figure
Figure 9C - Control Section Figure
Figure 9I - Input/Output Section Figure
Figure 9M - Memory Section Figure

To locate a particular logic element on a given figure, a grid system is used. The grid in the vertical direction is divided into four areas labeled A through D bottom to top of page. The horizontal grid areas are eight sections labeled 1 through 8, right to left.

For example, the designation "Figure 9A4-3C" refers to Section 9 - Arithmetic - sheet 4 - coordinate 3C. (Refer to Figure 4-1-A.)

Card interconnections are specified as shown on Figure 4-1-B. The example shows a card interconnection between card 2A1015 pin 31 to pin 42 of card 2A1014.

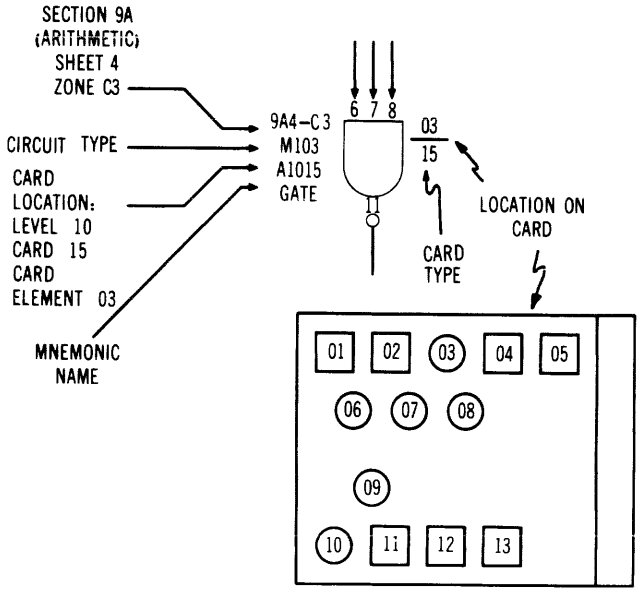


Figure 4-1-A. Logic Symbology

The element locations on a card are determined by positioning the card as shown on Figure 4-1-K and counting the elements left to right as shown.

(7) Basic Circuits and Symbology. - This paragraph reviews the basic circuits and symbology used by the 2402 Computer.

(a) Basic Circuit. - The basic circuit of the 2402 Computer is the NAND circuit (see Figure 4-1-C). This circuit can be described as a logical AND function followed by a logic inverter. Its function can be considered fundamental in that all sequential and combinational logic functions can be performed entirely by NAND gates.

The basic circuit is packaged as a silicon integrated circuit Westinghouse type WM-201. It is encapsulated in a 12-pin unit which contains two independent NAND circuits on a single silicon chip. Operating temperature range for this circuit is -67°F to +257°F.

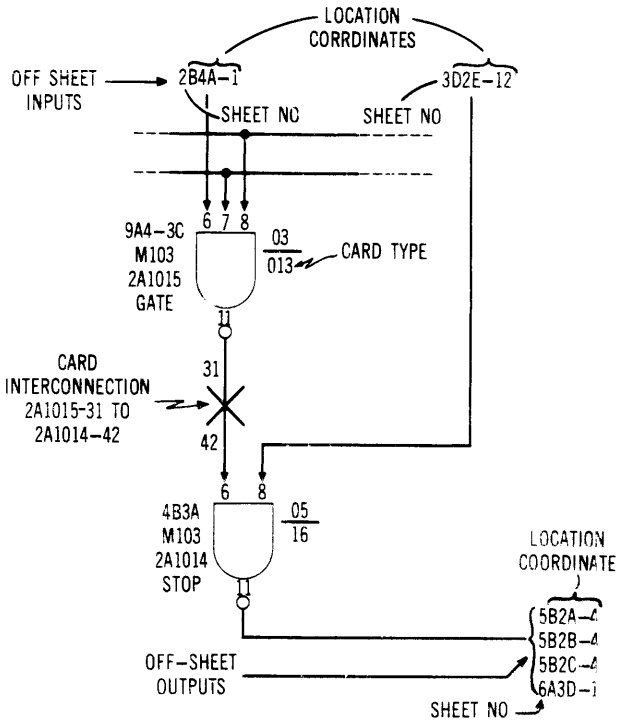


Figure 4-1-B. Card Interconnection

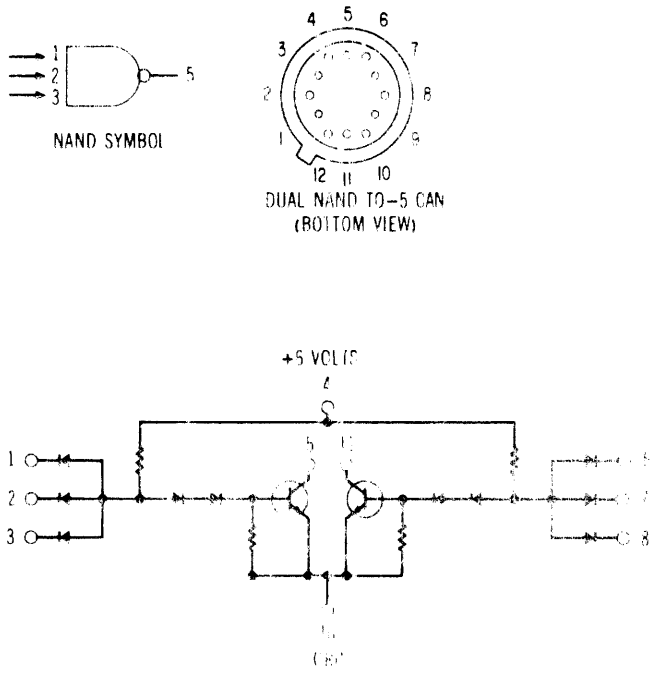


Figure 4-1-C. Dual NAND Circuit

The NAND circuit has one to three inputs and its output may drive up to 10 other NAND circuits, i.e., the "fan out" of the circuit is ten at a temperature of 25°C. Switching time (rise and fall times) for this circuit range from 33 to 54 nanoseconds under one-gate load.

The NAND circuit is an extremely versatile building block from which many logic functions may be constructed. NAND gates can be used to perform either "AND"ing or "OR"ing functions.

1. "OR" Logic Function. - The basic NAND circuit lends itself to the "OR"ing of input logical zeros. From a logical analysis it is apparent that a zero on any or all inputs will produce a "1" output. Zero volts is Logical "0" while +6 volts or an open circuit equals a logical "1". The NAND gate shown below is a current device, i.e., when a ground path is provided, current flows and when the path is switched open, current does not flow and the gate "switches". A ground (logical "0") on any one or all of the input diodes will provide a current path through the diode from the +6 volt source resulting in a 6 volt drop across R_1. Base to emitter voltage decreases below the cutoff potential causing the NPN transistor to cut off. The output (D) will increase to +6V by action of the pull up circuit R_3.

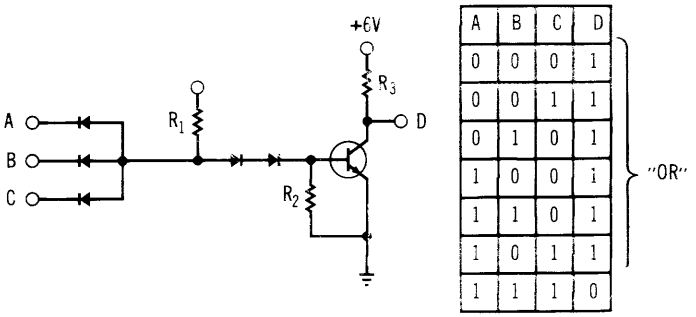


Figure 4-1-D. OR Logic Function

2. "AND" Function. - The NAND circuit can also be used in the logical "AND" capacity. Whereas logical "0"'s were ORed logical "ones" are logically ANDed. A logical one on all input terminals is the only condition which will produce a logical zero at the output terminal.

The truth table for the NAND element is shown in Figure 4-1-E.

A	B	C	D	
0	0	0	1	
0	0	1	1	
0	1	0	1	
1	0	0	1	
1	1	0	1	
1	0	1	1	
1	1	1	0	"AND"

Figure 4-1-E. AND Logic Function

3. Storage Function. - Two NAND circuits may be connected to form a bistable device, and therefore may be used to store a binary digit (bit) of information. The device thus formed is known as a "Reset-Set" (RS) flip flop.

a. RS Flip Flop. - The Set-Reset type flip flop may be logically formed by two NAND gates by connecting them as shown in Figure 4-1-F. Each NAND gate of the RS flip flop is assigned a name; one is arbitrarily called the "SET side" and the other the "RESET side". Their outputs will always be opposite, i.e., when the SET side is outputting a logical "1" the RESET side is outputting a logical "0".

When the SET side is outputting a logical "1" the flip flop is said to be "set". On the other hand, when the SET side is outputting a logical "0" the flip flop is said to be "reset" or "in the clear state". A logical "0" on the proper input is required to change the state of the flip flop. Zeros appearing simultaneously on both set and reset inputs may or may not change the state of the flip flop; i.e., the flip flop assumes an indeterminate state. This feature is characteristic of the RS flip flop.

Consider as an example the case where the flip-flop is initially reset (i.e., a logical "1" at the output of the RESET side - output F). The SET side (output E) is a logical "0" which holds the reset input at a logical "0". Zeros at the other reset inputs (C&D) will have no effect on the state of the flip flop. Inputs A and B are held at a logical "1". The stable "reset" condition is thus maintained. A logical "0" on either input of the SET side (A or B) causes the flip flop to change state to the "set" condition. The foregoing logical analysis can be confirmed by an electronic analysis utilizing Figure 4-1-G.

b. J-K Flip Flop. - In contrast to the RS flip flop previously discussed, the JK flip flop is actuated by a logical one instead of a logical zero. The set and set enable inputs are "ANDed" as are the reset and reset enable inputs. Also, unlike the RS flip flop, the JK requires a transient (negative going) pulse to set or reset the flip flop via the normal set or reset. (See Figure 4-1-H.) Assume that, when power is initially applied, the flip flop assumes the reset state. Transistor Q_1 is cut off and Q_2 conducting. The base of Q_2 (V_{b2}) is held positive with respect to the emitter by the current path from the (+12 volt supply) $-R_7-CR_{10}-CR_{11}-R_6$-gnd.

The base of Q_1 (V_{b1}) is held at ground potential by the current path from (+12 supply) $-R_2-CR_4-Q_2$-gnd. The voltage drop across R_2 = 12 volts. The collector voltage of transistor Q_1 (V_{c1}) is +6 volts by action of pull up resistor R_4.

The flip flop will now change states (be set) via the normal set and set enable input. Assume the set input (CR_{14}) in a constant +6V. Now assume a +6 volt pulse appears at the set enable input (CR_{13}). As the level of the pulse goes from 0 volts to +6 volts, capacitor C2 charges. This charging has no effect on the circuit other than to increase V_{b2} thereby driving Q2 into even greater conduction.

As the set enable pulse returns to 0 volts, capacitor C2 discharges producing a negative voltage spike to the base of transistor Q2. V_{b2} becomes negative with respect to the emitter of Q2 and cuts off. V_{c2} is raised to +6 volts as the drop across R5 goes from 6 volts to 0 volts. The base potential of transistor Q1 is raised to some positive voltage by the current path from (+12 volt supply) $-R_2-CR_6-CR_7-R_3$-gnd, causing transistor Q1 to conduct. V_{c1} goes to ground potential causing the base potential of transistor Q2 to approach 0 volts. Q2 is held cut off by this action, i.e., the current path from the (+12 volt supply) $-R_7-CR_9-Q_1$-gnd. The state of the flip flop is now changed to the "set" condition.

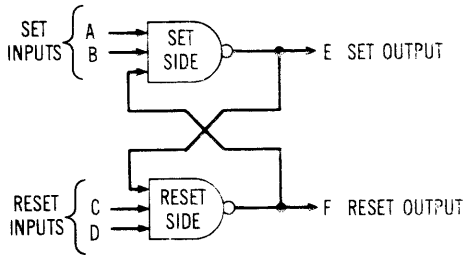


Figure 4-1-F. R-S Flip Flop (Logic Connections)

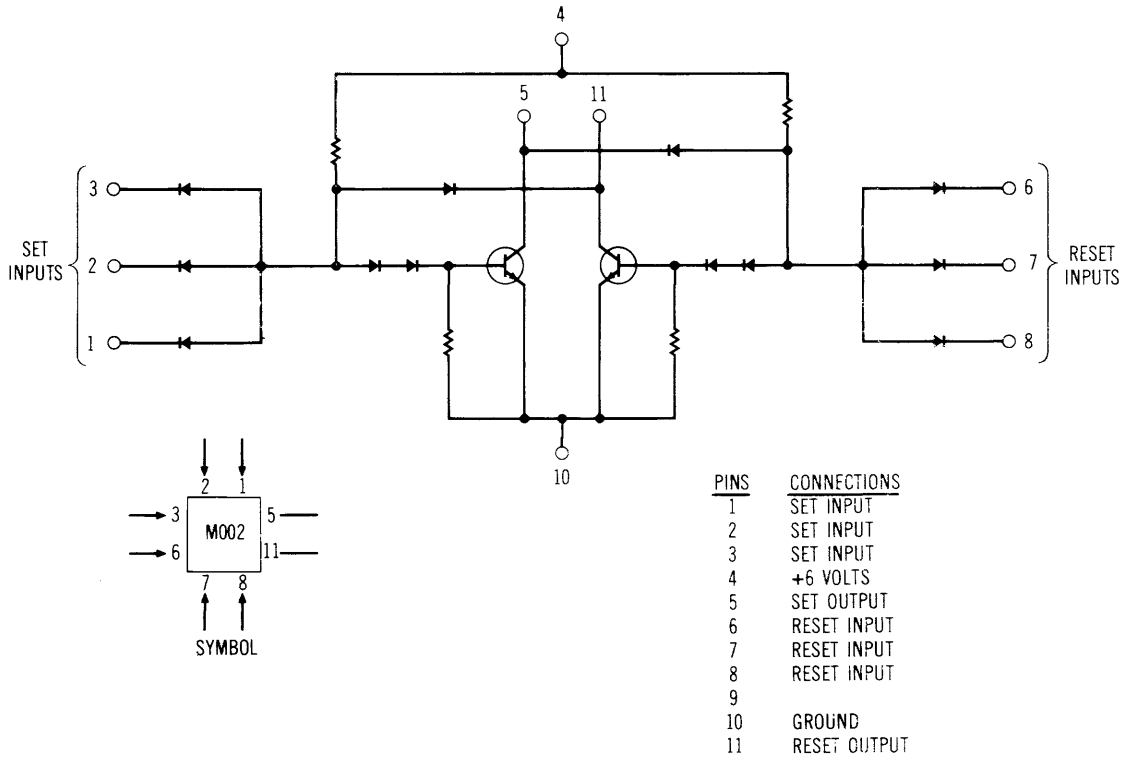


Figure 4-1-G. R-S Flip Flop (Microelectric Device)

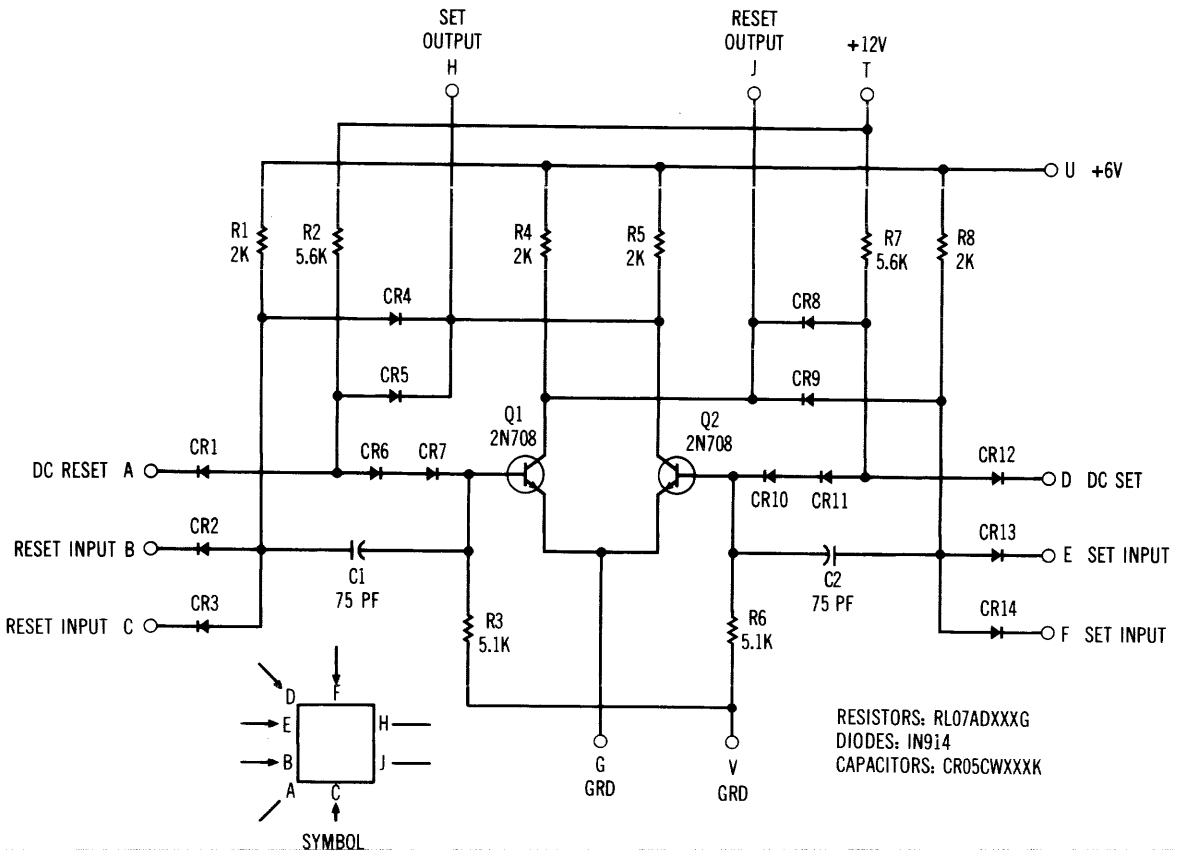


Figure 4-1-H. M001 J-K Flip Flop

Consider next the case when set and reset inputs, with their enables, are present simultaneously. Assume that the flip flop is initially in the set condition. A constant +6 volt signal is applied to the set input and the reset input. The reset enable and set enable inputs receive the same +6 volt pulse. Now the "steering" diodes CR9 and CR4 come into use. With the flip flop in the set condition CR9 will be forward biased to ground (V_{c1}) is at ground in the "set" state) and will eliminate the charging potential from the set and set enable input (which would normally charge C2). On the other hand, the diode CR4 is providing a clamp to the +6 volts (V_{c2}) at the collector of Q2 and will allow capacitor C1 to charge. In this way, only the reset signals were "honored" and the flip flop will change to the reset condition in the normal manner. Had the flip flop been initially reset, the set inputs would have been honored.

Consider the set and reset overrides which, up to this point, have been ignored. These overrides are zero volt levels which control the flip flop in the same manner as the RS flip flop. Referring to Figure 4-1-I, only the override circuit is shown.

Assume that the flip flop is in the reset condition. A zero volt level at the set override will force the base of Q2 to zero volts, causing Q2 to cut off. V_{c1} will increase to +6 volts, causing the base of Q2 to increase to about +6 volts. Q2 now conducts and the flip flop has changed to the set state. Note that zero volts appearing at both the set and reset overrides simultaneously will produce an indeterminate state in the same manner simultaneous set and reset signals do in the RS flip flop.

The J-K flip flop has five possible inputs: set, set enable, reset, reset enable, and override reset. The set and the set enable inputs form an "AND" gate input to the set side of the flip flop. The reset and reset enable inputs form an "AND" gate input to the reset side of the flip flop. The enable inputs need not be connected in order for the flip flop to operate.

With the enable inputs connected, the flip flop assumes the set state when a logical "1" signal appears at the set input and the set enable is at a logical "1".

When signals appear simultaneously at the "set" and "reset" inputs, the flip flop

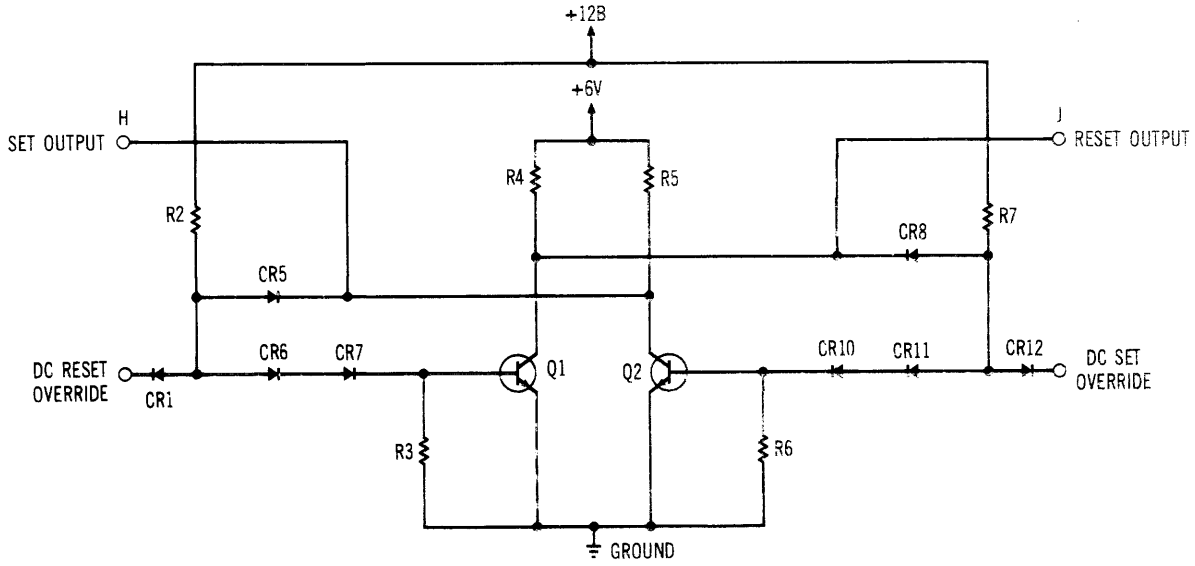


Figure 4-1-I. J-K Flip Flop Analysis

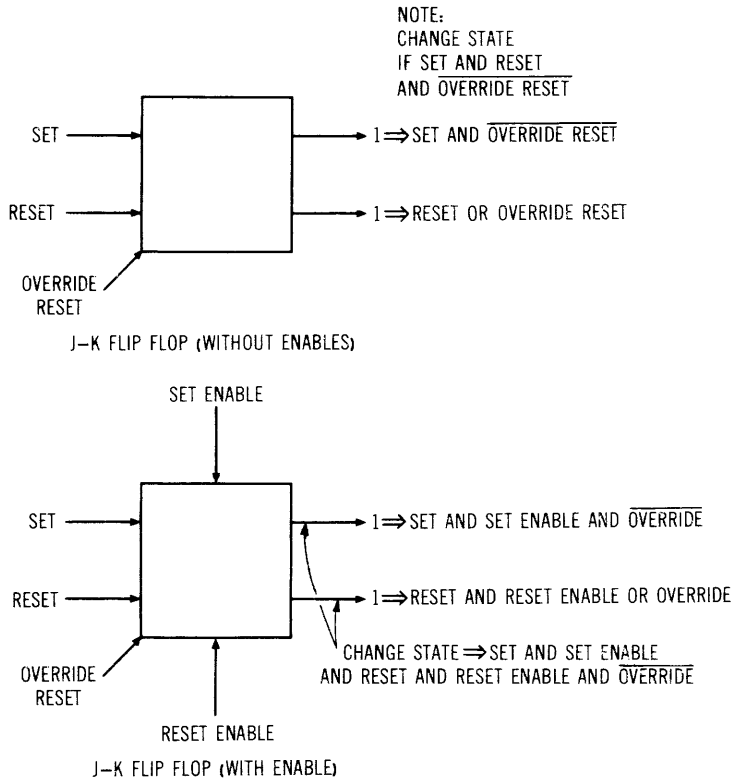


Figure 4-1-J. J-K Logic Symbols

reverses its state. The override reset input resets the flip flop regardless of the signals at the other inputs. The override signals are logical "0" (0 volts).

An integrated circuit JK flip flop (M215) is also used in the computer. This circuit operates in a similar manner to the D-T-L M001 circuit described above. The schematic of the M215 circuit is shown in Section 6.

(8) Time Delay Circuits

(a) Monostable Multivibrator (One Shot). - The "one shot" (Figure 4-1-K) has two possible input triggers, either will cause the element to change state from reset to set. Either the leading edge of a negative input pulse (lower left hand input) or the trailing edge of a negative input pulse (upper left hand input) will trigger the circuit.

The normal (unactuated) state of the one shot is zero. When actuated, the set output assumes the "1" state and the reset output assumes the "zero" state for a duration of time determined by the time constant of the device. The schematic of the M010 circuit is shown in Section 6.

(b) Delay Line. - A transmission line delay permits a pulse to be delayed in increments which are picked off the delay line at various points. Figure 4-1-L shows the symbol for a transmission line delay and the various increments of delay that can be attained. The delay line is useful when a certain sequence of events must be performed.

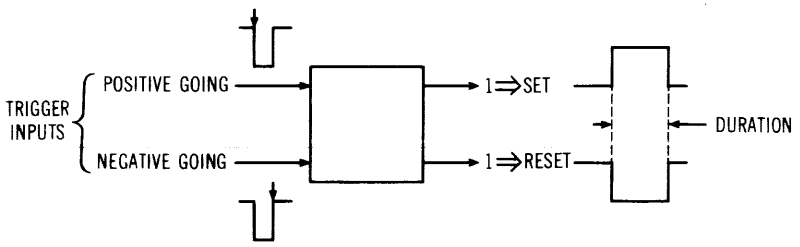


Figure 4-1-K. One Shot Multivibrator (M010)

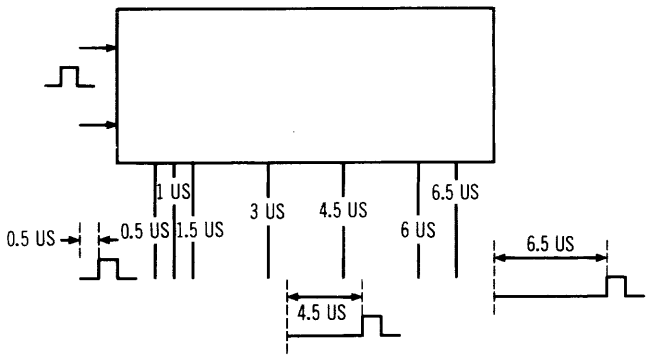


Figure 4-1-L. Delay Line Symbol

(9) Clearing Logic

(a) Master Clear. - The function of the MASTER CLEAR switch on the operators' console is to clear all working registers and establish a salient state of control flip flops in the entire computer.

The MASTER CLEAR is a momentary contact pushbutton switch shown on Figure 9C12-D7. When the switch is depressed, the logical "0" from gate MASC is distributed to the DC reset side input of all register flip flops throughout the computer, except a select few which are set by this action. For example the STRT flip flop shown on Figure 9C7-4D is set by the master clear switch. Refer to Figure 9C12-D7 for a list of master clear destination.

(b) Manual Clear. - All working registers, control flip flops and timing flip flops can be cleared individually or as individual units by means of switches on the maintenance and operator consoles. The manual clear switches (momentary contact) for all working registers are located to the right of the register indicator and, when depressed, generate a logical "0" which is sent to the DC reset side input of all flip flops in the respective register.

(10) Set Logic. - Each flip flop in the computer for which there is a neon indicator on the maintenance or operator panel may be manually set by depressing the (neon) pushbutton switch. The switch/indicator driver circuit is shown in Figure 4-1-K. The switch (pin 5) is connected to the set input for RS flip flops and the DC (over ride) set in the case of JK flip flops. Thus the flip flop will be sent when the pushbutton is depressed. The set output of the flip flop is connected to pin 1 of the circuit shown in Figure 4-1-M. When the flip flop is reset, a ground on this pin will hold the base of the transistor at a slightly negative voltage with respect to the emitter. The PNP transistor will conduct and prevent a sufficient voltage difference across the neon indicator to illuminate it. However, when the flip flop is set, the transistor is cut off by the positive base voltage and causes the neon to be illuminated.

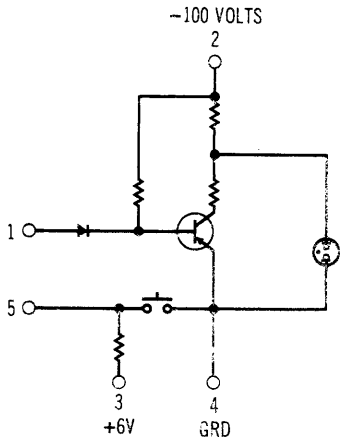


Figure 4-1-M. Neon Switch-Indicator

b. Computer Power Supplies

(1) Requirements. - The computer supplies provide the following voltages:

Voltage	Current	Application	% Reg.	Ripple
+6	25 amp	logic	.05	1 m. v.
+12	12 amp	logic	.05	1 m. v.
-12	6 amp	Logic & I/O drivers	.05	1 m. v.
-12	3 amp	Memory	.05	1 m. v.
+25	12 amp	Memory	.05	1 m. v.
-47	12 amp	Memory	.05	1 m. v.
-100	3/4 amp	neon indicators	(nonreg)	5 volts

These voltages are provided by replaceable sealed power units which contain built-in short circuit protection. They are not externally fused. Information on the power supplies may be found in Section 7, "Parts List". Once the circuit breaker in the power supplies has tripped, the breaker will reset automatically when input power is removed. Each power supply voltage is adjustable over its working range by screw driver adjustment on the supplies.

Figure 4-1-L shows the primary power wiring from the maniframe terminal boards into the power supplies.

Power to the cards normally utilize the following pins

Pin	Voltage
1	ground
47	ground
44	+6
45	+12
46	-12

Special voltages required by the memory section are brought in on various pins as shown on the card logic diagrams (Figure 9L).

(2) Power Failure Detection. - The computer has the ability to sense a power failure and immediately interrupt the main program. The purpose of this function is to transfer to core memory the contents of all principal data registers within the 200 microseconds between initial power failure detection and the time at which the logic voltages are below operating levels. After storing the contents of these registers, and after storing a jump instruction at address 00000000, the logic generates a master clear signal. If the power is restored within about 300 milliseconds after detection the logic will initiate an automatic recovery and the computer will restart beginning the program at adress 00000000. Address 00000000 will contain the beginning address of a routine to restore the contents of the working registers and jump to the point in the main program that was being executed when the power interrupt occurred.

Refer to Figure 4-1-N. The input AC power to the DC power supplies is sampled each cycle by two Schmitt Trigger circuits. These circuits have the characteristic that their outputs will remain DC levels as long as the AC voltage is continuous. When interruption occurs in the AC voltage, a pulsating DC signal will cause flip flop T to set. (Refer to Section 6, "Repair" for an electronic illustration of this detection circuit.) With flip flop T (FF01) set, gate GAT is fully enables and triggers the 150 μs one shot. Latching relay No 1 is in the position shown. Latching relay No 2 is in the opposite position from that shown because the COMPUTER POWER switch is in the "ON" position. While the 150 μs one shot is outputting a logical "0" (150 μs) the power interrupt (PINT) signal is sent to the I/O interrupt logic (Fig. 9I4). The trailing edge of PINT triggers one shot $\overline{\text{GOT}}$ which then outputs a logical "0" for 150 milliseconds. During this period, latching relay No 1 switches position and de-energizes relay K901 which removes power to the computer and turns on the Power Failure Indicator light. Also gate MASC is disabled (pin 7 = logical "0") and generates the computer master clear signal ($\overline{\text{MC}}$) to figure 9C12.

The trailing edge of $\overline{\text{GOT}}$ triggers a second 150 millisecond one shot $\overline{\text{GIT}}$ which outputs a logical "0" for 150 μs. Note that the Master clear signal is continuously generated during this entire period by the logical "0" into pin 7 of gate MASC from latching relay No 1.

Following the second 150 millisecond delay, the reset enable (pin 8) of Flip flop FF01 (T) is a logical "1". At this point the Schmitt trigger outputs are sampled to determine if power has been restored. If it has, flip flop FF01 is reset and the automatic start sequence begins. The gate at 5/122 will be fully enabled for one Schmitt trigger pulse width and will trigger one shot $\overline{\text{GOT}}$. Note that PINT is also generated at this time but is of no consequence. Latching relay No 1 will switch to the position shown during the $\overline{\text{GOT}}$ signal. Again the master clear signal is produced during $\overline{\text{GOT}}$ and subsequently during $\overline{\text{GIT}}$.

Following $\overline{\text{GIT}}$, gate MASC is enabled (outputs a logical "0"). The gate at 7/123 is fully enabled at this time and triggers the 2μs one shot which sends the auto start (AUST) signal to figure 9C12 to set the RUNN flip flop. Latching relay No 2 prevents sending AUST if the power switch is in the OFF position by grounding the signal. Figure 4-1-N (sheet 3) shows the timing relationships for the power fail/auto start sequence.

c. Master Clock. - The master timer-phase generator is considered part of the control section. However, because the clock pulses are

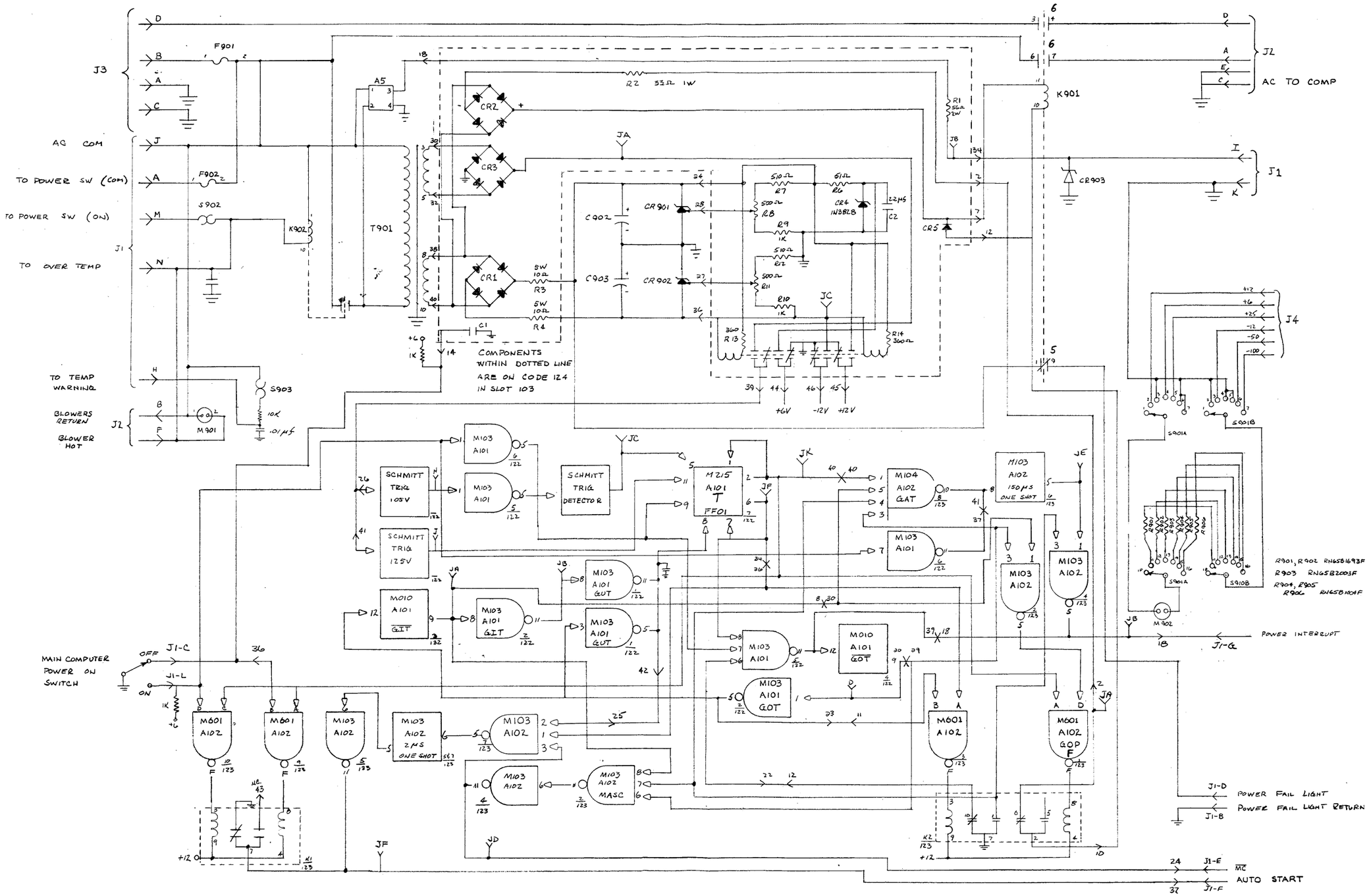


Figure 4-1-N. Power Failure Detection Logic (Sheet 1 of 3)

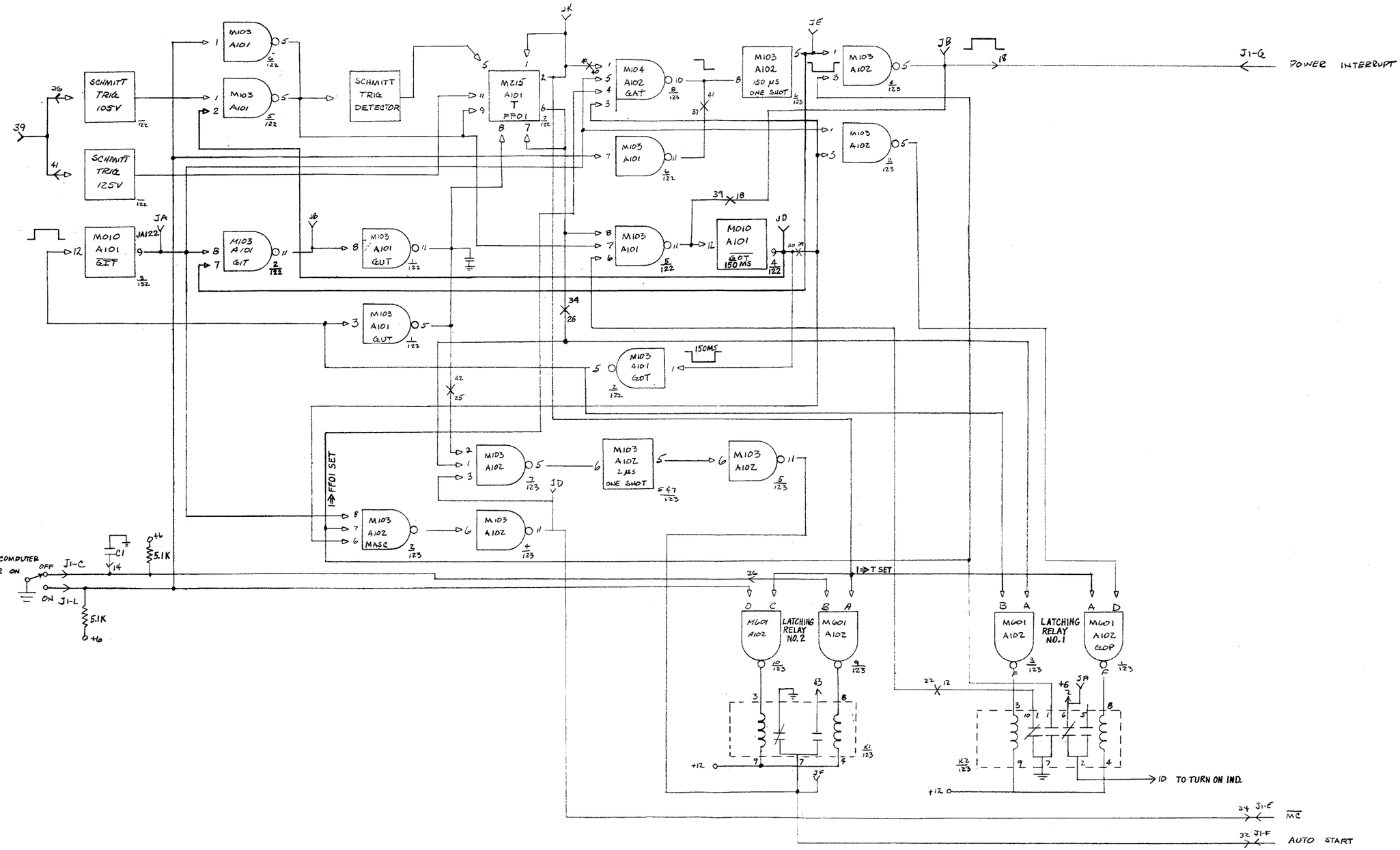


Figure 4-1-N. Power Failure Detection Logic (Sheet 2 of 3)

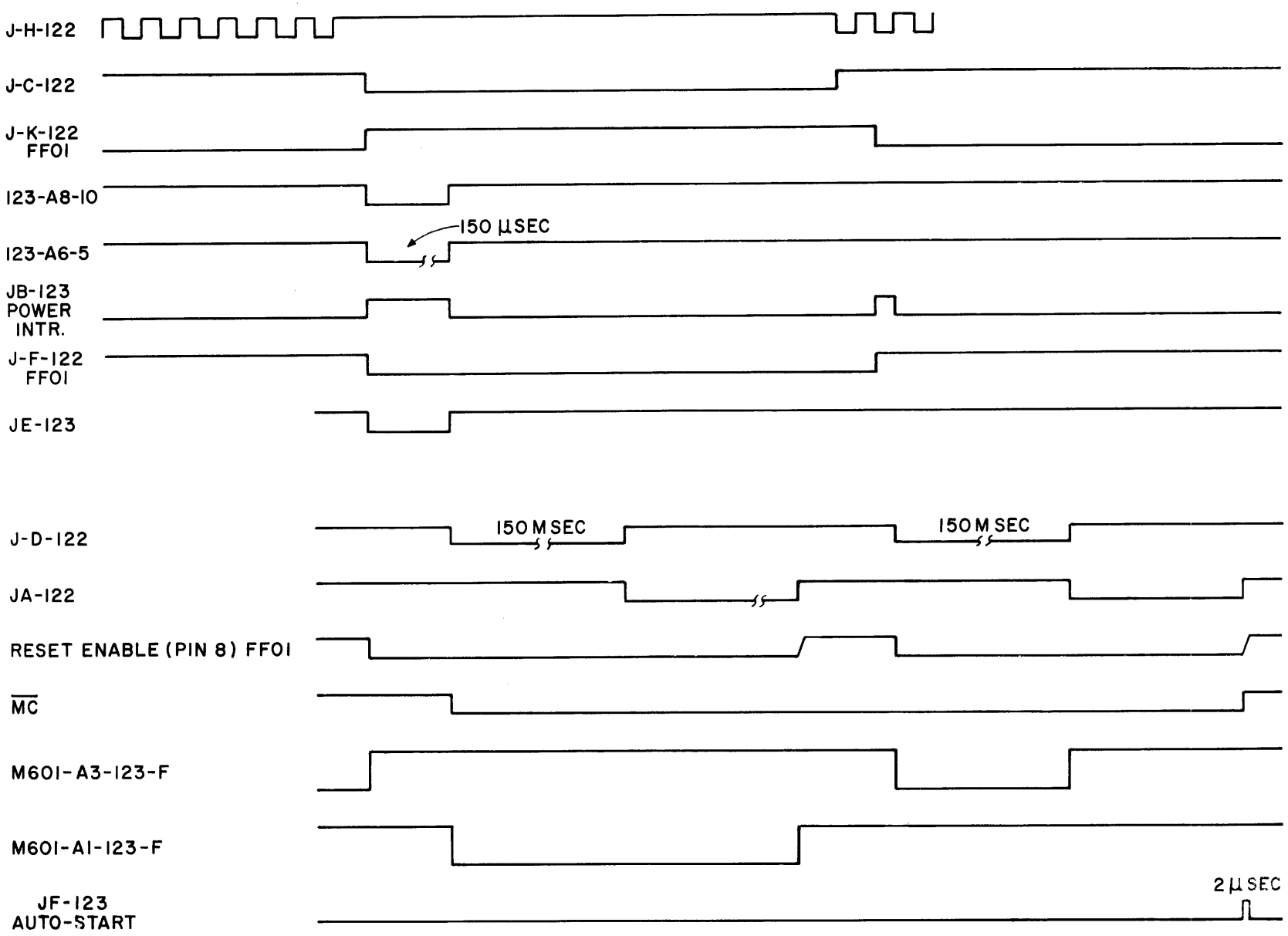


Figure 4-1-N. Power Failure Detection Logic (Sheet 3 of 3)

used for timing by all sections of the computer, it is fitting that the master clock logic preceed the theory of operation of the computer sections.

The master clock is shown in Figure 9C12. Eleven clocking pulses are generated from a basic 3 megacycle crystal controlled sine wave oscillator source. These pulses are generated, shaped and used to control the flow of information within the computer. Two modes of operation are possible: "High Speed: and "Step".

(1) High Speed Operation. - The basic clock oscillator module is described in Section 7.

Refer to Figure 9C12-C7. The output of the 3-megacycle sine wave oscillator (CLOS) is constantly applied to inputs F and C of flip flop SING and to gates C055 and C056.

The RUNN (Figure 9C12-B3) flip flop, when set, allows clock pulses to be generated as shown in Figure 4-1-A. The RUNN flip flop is set for High Speed operation in the following manner.

Assume that the computer has been MASTER CLEARED and flip flops SING, SINT, and RUNN are placed in the "clear" condition. With RUNN cleared, flip flops CL2B and CL2C are cleared and CL2A is set. This ensures that the clock will always start at the sampe place. With SINT cleared, the clock oscillator (CLOS) signal has not effect on SING as the logical "0" from the set side of SINT is placed on inputs B and E of SING.

Flip flop RUNT (Figure 9C12-A8) is normally set (run switch in the neutral position).

The entire sequence of events for generating the clock phases during high speed run is as follows:

NOTE

This sequence of events include a summary of the preceding discussion on setting the RUNN flip flop.

1. Run Mode switch is placed in the "NORMAL RUN" position (position 1).

2. The operator depressed the RUN switch.

3. Flip flop RUNT is thus reset and a logical "0" is generated which sets SINT.

4. SINT being set allows SING to be set and cleared by CLOS, the clock oscillator, at the oscillator's rate of 3 mc.

5. As SING sets for the first time output from SING sets RUNN.

6. SING being set enables gate CO56. CLOS can now pass through CO56. And, with RUNN set, CLOK is generated from the shaped CLOS signal. When RUNN is set, CLOK allows pulses from CLOS to pass through. The stream of CLOK pulses causes a stepping action of flip flops CL2A, CL2B and CL2C. The initial state of these flip flops (which is a result of RUNN being cleared) is CL2A set, CL2B and CL2C both cleared. Therefore, the first CLOK pulse only generates CL01 as CL01 is the only gate fully enabled at this point. (Figure 9C12-D1).

On the trailing edge of the first CLOK pulse, flip flop CL2B will set. The next CLOK pulse will generate CL#1. The process continues to produce the timing pulse sequence shown in Figure 4-1-O until either a programmed stop occurs or the operator activates the STOP switch. In either case the machine does not stop until after the current instruction has been executed. The end of execution of the current instruction is signified by the signal COGY (Figure 9C7-C7).

Consider the process of stopping the generation of phases by depressing the STOP switch. This switch and logic is shown on Figure 9C12-7D. The logical "0" from the STOP switch is inverted by the single input gate STOP (9C12-7B); the resulting logical "1" is used to partically enable the lower CHAT gate. When SYNC and COGY (which indicates the completion of the current instruction) appear the gate CHAT is fully enabled and the RUNN flip flop is cleared, thus further phase generation is prevented.

(2) Micro-Step Operation. - In the microstep mode, the clock logic will produce a single clock pulse each time the RUN switch is depressed.

The following sequence of events illustrates how a single clock pulse is produced (See Figure 4-1-P)

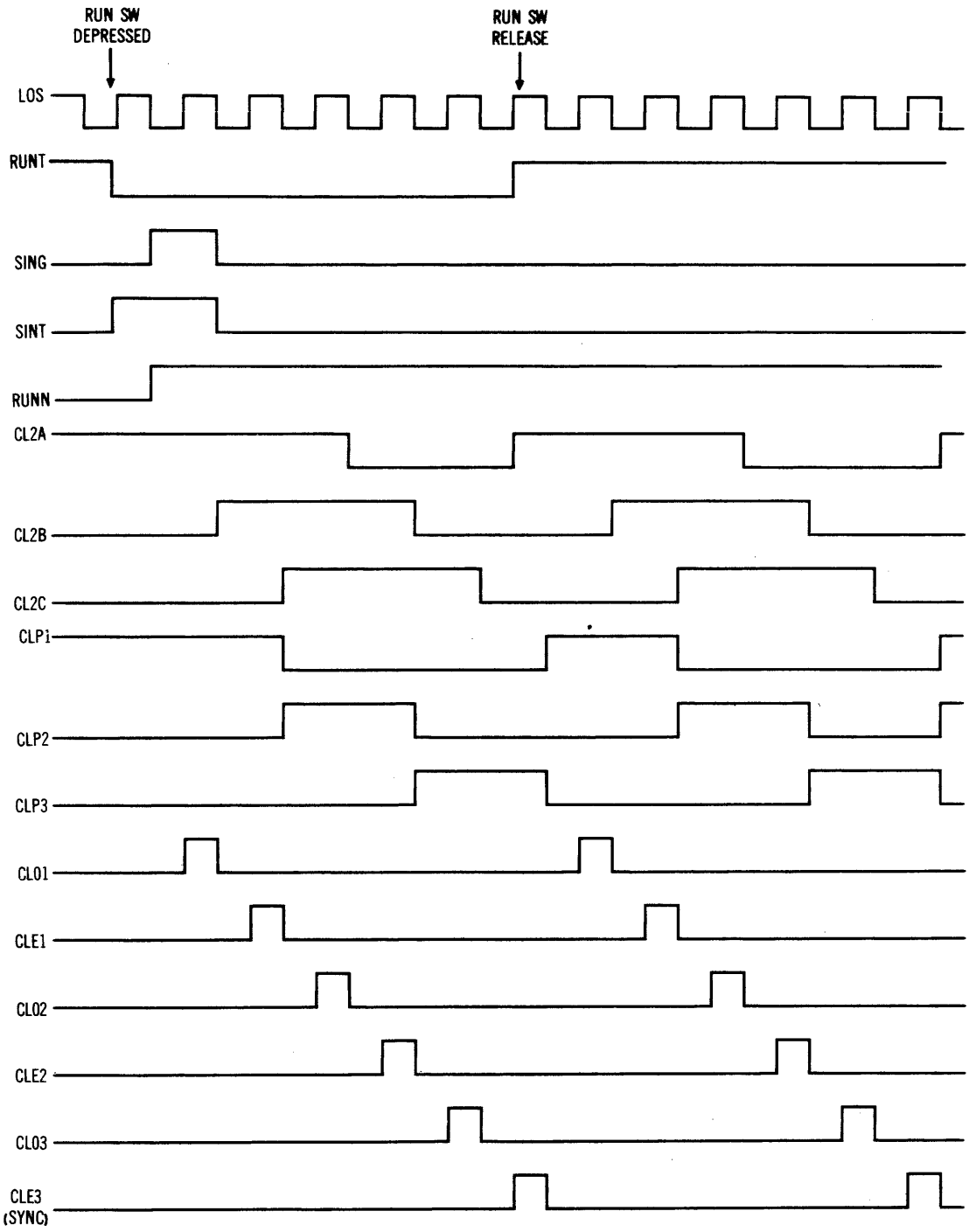


Figure 4-1-O. Clock Waveforms, Free Running Operation

Assume that a master clear has just been accomplished and SING, SINT, RUNN, CL2B and CL2C have been cleared; while RUNT, and CL2A are set

1. The operator depressed the RUN switch, clearing flip flop RUNT (Figure 9C12-B8).

2. As RUNT is cleared, SINT is set. SING is allowed to set and clear with CLOS.

3. On the trailing edge of the next CLOS pulse, RUNN is set because SINT is cleared.

4. SINT will clear on the next trailing edge of CLOS; but, in the meantime, one pulse has been generated at the output of gate C055. The upper CLOK gate has a constant logical "1" on input pin A (because of the output of CO56) therefore, the logical state of the CLOK gate is goverened by input C from CO55.

5. The CLOK pulse will cause the stepping action of the flip flops CL2A, CL2B and CL2C as described in the previous discussion of the free running operation. Thus a different CL clock pulse is generated for each depression of the RUN switch.

6. As this run mode will not be used except for maintenance pruposes, a yellow TEST light on the operator console is illuminated when the RUN switch is in the step mode position (Figure 9C12-8B).

(3) Instruction Step. - Instruction Step Mode of Operation, depressing the RUN switch generates a burst of pulses sufficient to execute one complete instruction at the normal high speed rate. The RUN MODE switch is placed in the INSTRUCTION STEP position (switch position Number 4). Refer to Figure 4-1-Q. Generation of CLOK signals proceeds at the normal rate until the signal COGY, from the signal section 9C-7C, droes to a logical "0" and signifies that the current instruction has been completed. At this time gate CHAT (Figure 9C12-B6) is enabled (the uppermost CHAT gate) and the resulting logical "0" clears the RUNN flip flop and prevents further phase generation by the clock logic.

A single instruction will be executed each time the run switch is depressed. The TEST light is illuminated on the operating console.

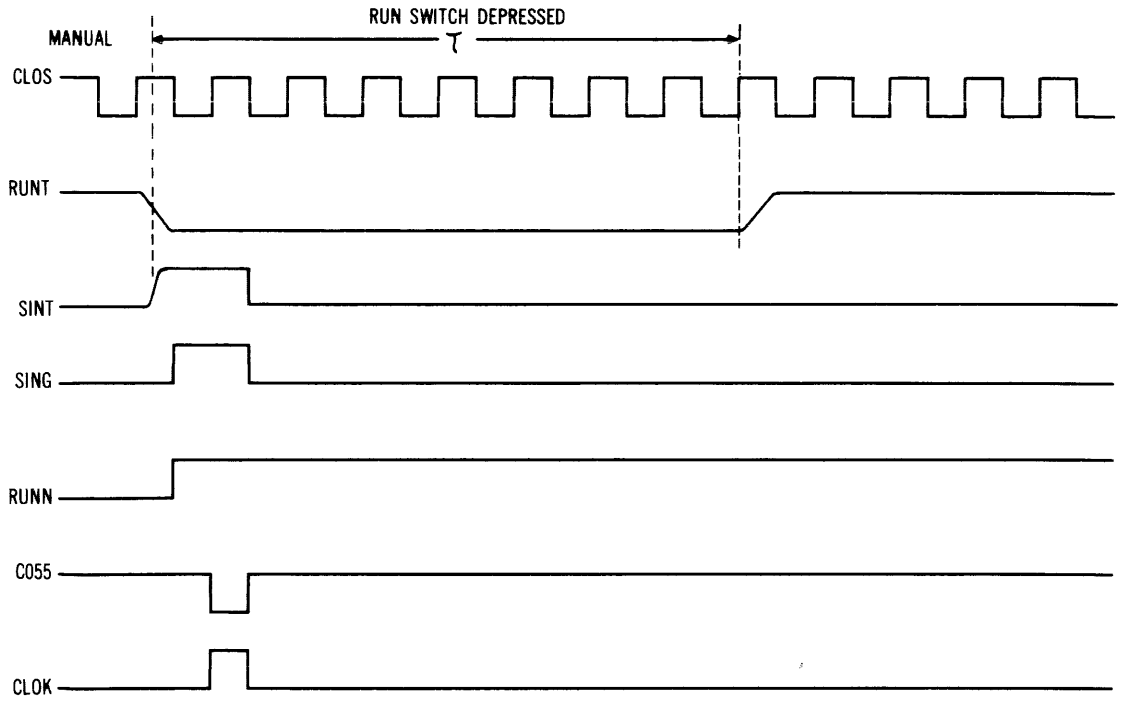


Figure 4-1-P. Microstep Operation

(4) Instruction LO RUN. - In this mode, one instruction will be executed at high speed when a pulse is received from a variable (1-200 cps) low speed oscillator. Operation is similar to Instruction Step mode with the exception that the low speed oscillator replaces the RUN switch as the initiating device.

The low speed oscillator is simply a variable electronic delay (one-shot oscillator). Three one-shot oscillators OST1, OST2, and OST3 are connected in series to provide the desired range of delay. Refer to Figures 4-1-E and 9C12-C7.

With the RUN MODE switch in the INSTR LO RUN or PROGRAM LO RUN (positions 2 or 3) gate LORS (9C12-7C) is partially enabled (pins 7 and 8) and requires only a logical "1" on pin 6 to set SINT. This logical "1" will result after the delay one shots have produced the desired delay.

Assume that the RUN MODE switch is in the INSTR LO RUN position (position no. 3). The TEST INDICATOR light on the operators' panel will be illuminated and the following sequence will result:

1. The operator depresses the RUN switch. RUNT is cleared, SINT and SING set and RUNN is set in the same manner as is accomplished in the previously discussed run modes. (See Figure 4-1-E)

2. The CLOK pulses will step the flip flops CL2A, CL2B and CL2C and produce clock pulses in the same manner as the INSTRUCTION STEP mode.

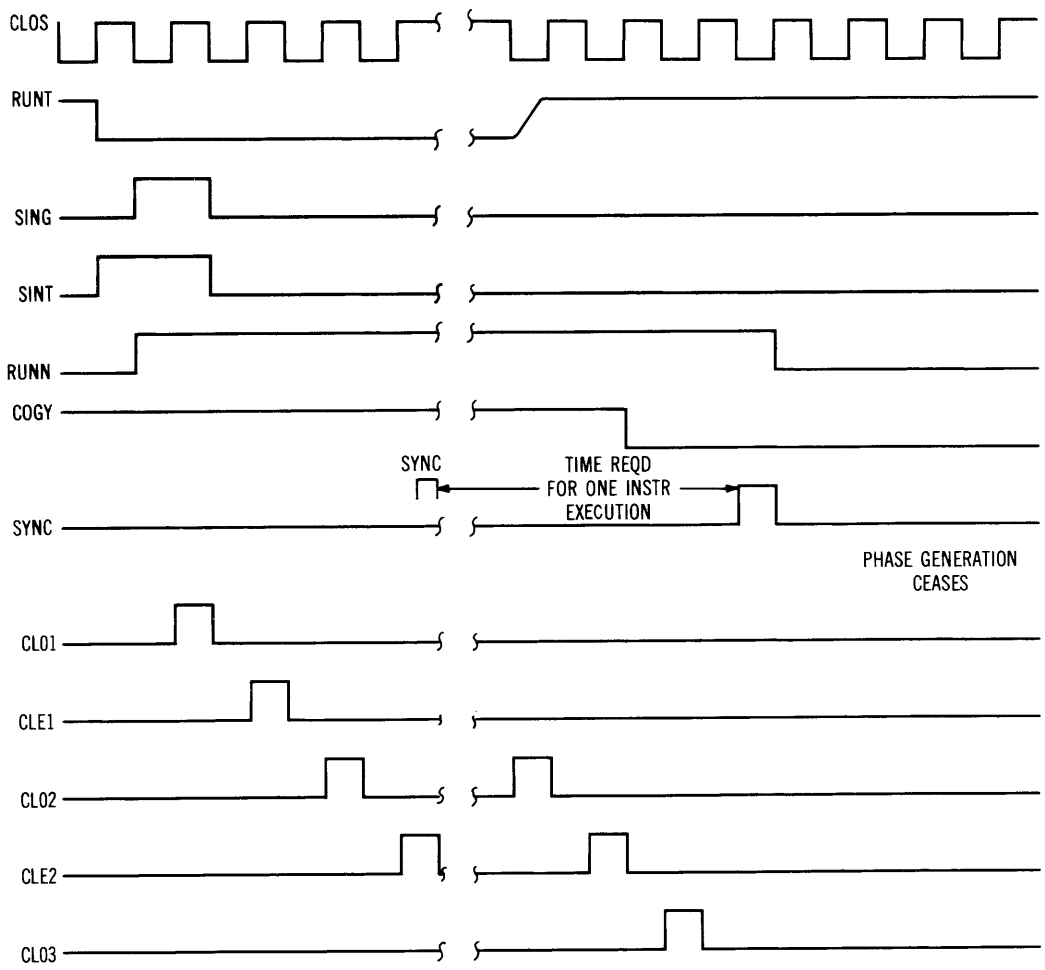


Figure 4-1-Q. Instruction Step Operation

3. At the end of the instruction, COGY will be produced and the RUNN flip flop will be cleared by the upper CHAT gate (9C12-B6) during CLE3 (sync) time.

4. When the RUNN flip flop is cleared, the logical "1" from the reset side (pin 11) is fed to gate LFG1 (9C12-7C) and inverted. The resulting logical "0" is fed into one shot OST1. (Refer to Section 6 for a description of the variable one-shot module). The output of OST1 enters OST2 and thence to OST3. The output of OST3 is inverted by LFG4 and from there to gate LORS. The logical "0" from the fully enabled LORS is used to set SINT via the DC set input and the clock produces a burst of clock pulses to execute the next instruction.

5. The INSTR LO RUN process will continue until either the LO RUN FREQUENCY control is switched to the OFF position or the operator holds the STOP switch until the one shots have completed a cycle. Both actions stop the LO RUN process at LORS (Figure 9C12-B7).

(5) Program LO RUN Operation (See Figure 4-1-R). - When the RUN MODE switch is in the PROG LO RUN position 2, the computer will recover from programmed STOP consitions by a pulse from the low speed oscillator. The TEST light is illuminated on the operators' panel and the following operations will occur:

1. The operator depresses the RUN switch. RUNT is cleared, SINT, SING and RUNN set as shown in Figure 4-1-E.

2. A programmed stop will produce a logical "0" from gate CHAT (Figure 9C1-3B) which will clear the RUNN flip flop.

3. As in the discussion on the INSTR LO RUN; operation, when the RUNN flip flop is cleared the resulting "1" from pin 11 of RUNN is inverted by gate LFG1 (9C12-7C) and fed into the three one shot delay OST1, OST2 and OST3.

4. The output of the delay one-shots will enable LORS and set flip flop SINT to restart the generation of clock pulses by means of SING and RUNN flip flops as before.

5. The program LO RUN operation can be stopped in the same manner as the INSTR LO RUN, ie., switch the LO RUN frequency control to the OFF position or hold the RUN/STOP switch to the STOP position.

4.2 MEMORY SECTION OPERATION

The Memory section of the computer is a current-operated, magnetic core memory which uses the magnetic properties of ferrite cores to store binary information. See paragraph 4-2(7) for a review of magnetic core theory.

a. General Information. - The computer memory is high speed, random access and non-volatile. These terms are defined as follows:

high speed - The speed of operation is compatible with the other computer sections. Two microseconds is required for a complete read-write cycle. However, the core information is available in the M-register for use by the other computer sections after about one microsecond.

random-access - Data may be referenced in a non-sequential manner, i. e., the same amount of time is required to access any address in memory.

non-volatile - The memory system retains its data when power is removed from the computer.

The computer memory has a capacity of 16,384 locations (expandable to 32,768) for the storage of 24-bit words (plus a parity check bit). Each storage location is assigned an address (000000000_8 through 777777777_8). A 24-bit word in storage may be divided into two 12 bit segments, designated LH and RH by proper selection of a program k-designator. (See Section 4-3, "Control Section Operation".)

The memory section of the computer is physically located on chassis 2 and is functionally divided into four segments (see Figure 4-2-A):

The Core Stack - the ferrite storage device for the computer

Address Selection Circuits - selection of any particular address at random.

Data Control and Timing - regulation of the flow of information into and out of the memory section

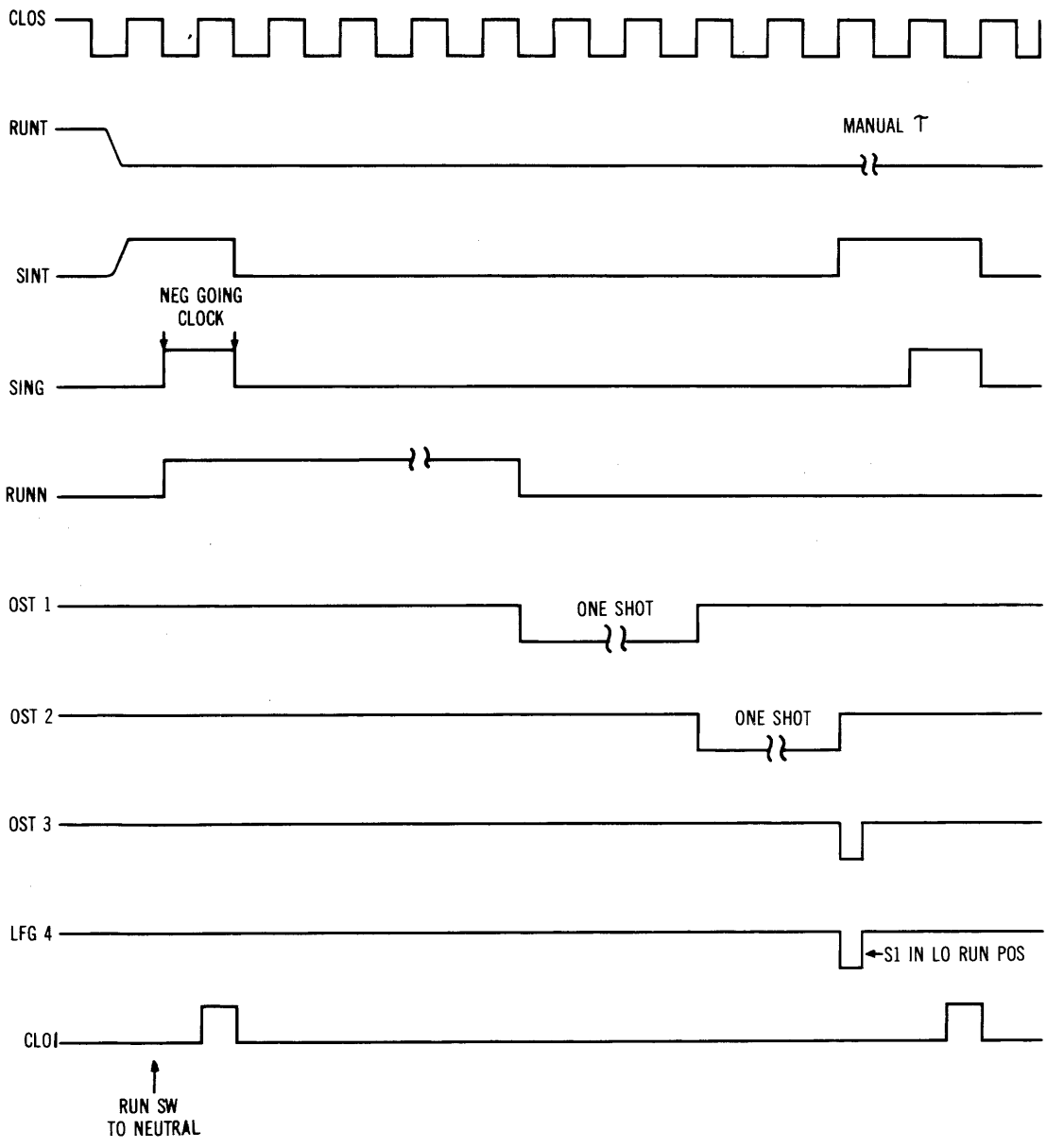


Figure 4-1-R. Lo-Run Operation

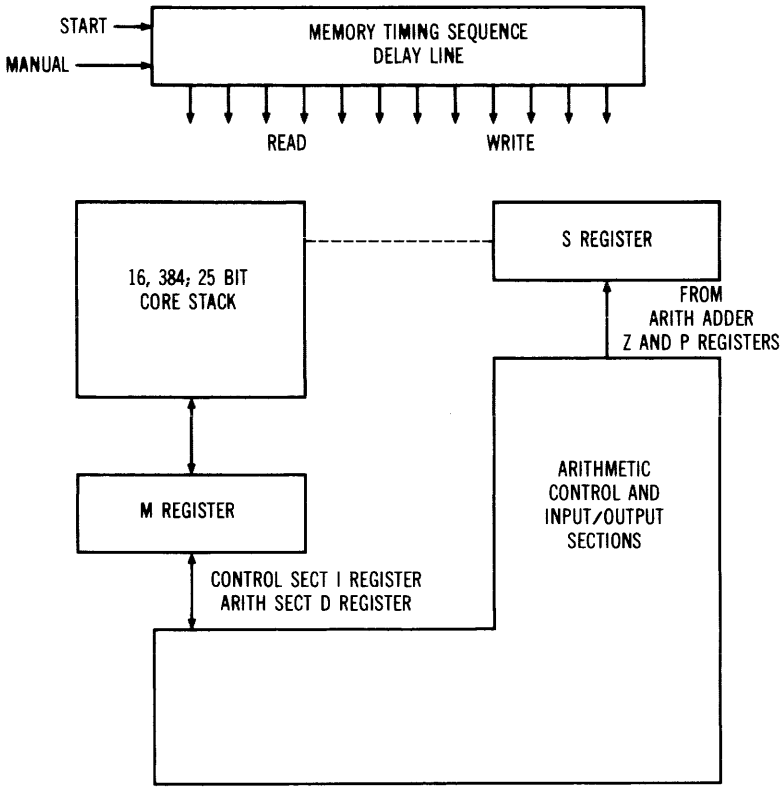


Figure 4-2-A. Memory Section Simplified Block Diagram

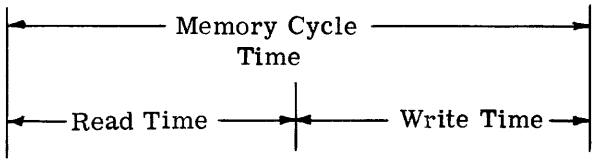
Wired Memory - provision for a permanent wired boot-strap load routine in a five word auxiliary memory.

The memory section contains two registers: a memory buffer registor M and an address register S. The M-register serves a twofold purpose: (1) to act as a timing buffer between the memory section and the rest of the computer and, (2) to hold the content of a memory location until it can be replaced in the same address of memory. This replacement is necessary because the content of a location is destroyed as it is read (destructive readout). For this reason, a "write" function necessarily follows the "read" operation in the memory cycle.

The data transmission into or out of the selected storage location (register) is channelled through the M register. The Control and Input/Output sections of the computer have independent access to the storage registers through the use of the S register and translator, the M-register, and the Memory Timing sequence.

All timing relationships in the memory section are established by the Memory Timing circuits (delay line) which may be initiated manually or by the control or I/O sections.

The same time sequence is employed whether the computer is to read information from memory or to write new information into memory. Each of these operations is accomplished by having a read and write time occur during the memory cycle. The time required for this timing sequence (read and write time) is referred to as the memory cycle time and is two microseconds in the DPS-2402 computer.



This timing sequence occurs for a read or write operation

Two accesses of memory are accomplished each memory cycle.

The S register is used to hold the address which is to be referenced during the memory cycle. It is one register, but can better be thought of as being divided into two sections since it must contain two addresses which function as one. One part of the address designates the selection of the X drive line and the other one, the Y line. This register is usually made up of flip-flops since a steady-state output and both the set and reset outputs are needed by the decoder.

When a specific storage location in memory is referenced, the S register contains a 14-bit address word that specifies one of the storage locations.

The cores are assembled in square matrices on a memory board. Figure 4-2-B illustrates a simplified memory board containing a 4 x 4 array of cores. The actual bit plane used in the computer is a 128 x 128 array. See Figure 4-2-C. The bit plane is divided into four quadrants as shown.

Selectable conductors (X and Y drive lines) thread through each core in each row and column. Any one of the magnetic cores in the matrix may be selected (addressed) by pulsing a given row and column of cores. The following sequence illustrates how a specific core is selected. A coincident half-amplitude current pulse is generated in the selected row and column. The core at the intersection of the row and column (selected core) receives a net full-amplitude current pulse. For example, assume core A on Figure 4-2-B has been selected or addressed. It receives a net full-amplitude (H_m) pulse when coincident half-amplitude current pulses are impressed on its intersecting drive lines. All other cores in the same row or in the same column as core A receive half-amplitude current pulses; e. g., cores B, C, D, G, and E in Figure 4-2-B. These cores are half-selected. The remaining cores, neither half-selected nor fully selected, are referred to as unselected cores; in the above example core F (in Figure 4-2-B) is representative of an unselected core.

The binary information ("0" or "1") stored in a core is determined by the polarity of its remanent magnetization. The information is extracted from a selected core when two coincident half-amplitude read current pulses, one pulse on each of the two intersecting drive lines, attempt to switch the magnetization of the core. The read pulse polarity is such that if the core holds a "1", the magnetization is reversed. This reversal of the core's magnetization induces voltage in the output line (sense winding) which threads through every core in the plane (See Figure 4-2-D). Sense amplifiers detect the induced voltage and interprets it as a "1". When the selected core is in the "0" state, an insignificant voltage is induced in the sense winding when the read pulses are applied. The small induced voltage indicates no flux reversal. The sense amplifier, unable to detect a voltage, interprets the results as a logical "0".

When operating on a selected core, a read pulse is followed by a write pulse of the opposite polarity on the drive lines. For this reason the drive lines are also called read/write lines or R/W lines. When a core contains a "1", a full-amplitude read pulse switches the core to a "0" (destructive readout). The following full-amplitude write pulse switches the core back to a "1". If a logical "0" is to be written in the selected core, an inhibit pulse is applied simultaneously with the Write pulse. The Inhibit pulse is applied in the read direction and is equal to a half-amplitude read pulse. The net effect on a selected core by applying a full-amplitude write pulse coincident with an inhibit pulse, is a half-amplitude write pulse. This pulse is insufficient to switch the core from a "0" to a "1". For a discussion of core theory, refer to paragraph 4-2b(7).

The operation described in the previous topic for the selection of core is applied, with refinements, to the selection of a word from any specified address in memory.

Figure 4-2-E shows an example of a complete 3 bit memory system. The S register contains the address to referenced; the upper half of which specifies the X drive line and the lower half the Y drive line in all bit planes. The cores are driven in the "read" direction and those which are switched from "1" to "0" output a signal to the sense amplifiers for the particular bit. The sense amplifiers, in turn, set the corresponding bit of the M-register. The cores all contain "0"'s after reading.

During the "write" portion of the cycle, the write current writes a "1" in each core of the address for which there is no inhibit. An inhibit is generated by a zero bit in M. Thus the original contents are re-written in memory.

New information can be written into memory by placing the new data in the M-register and then inhibiting the sense amplifier outputs during the read portion of a read/write cycle.

(1) Memory Stack. - The storage elements of the Memory section are contained in memory stacks. Each stack contains all of the 16, 384 addresses required for the storage of 24-bit words.

(2) Memory Board. - The memory board is the basic unit of the memory stack. It has 16, 384 magnetic cores that are at the intersection of horizontal and vertical conductors. See Figure

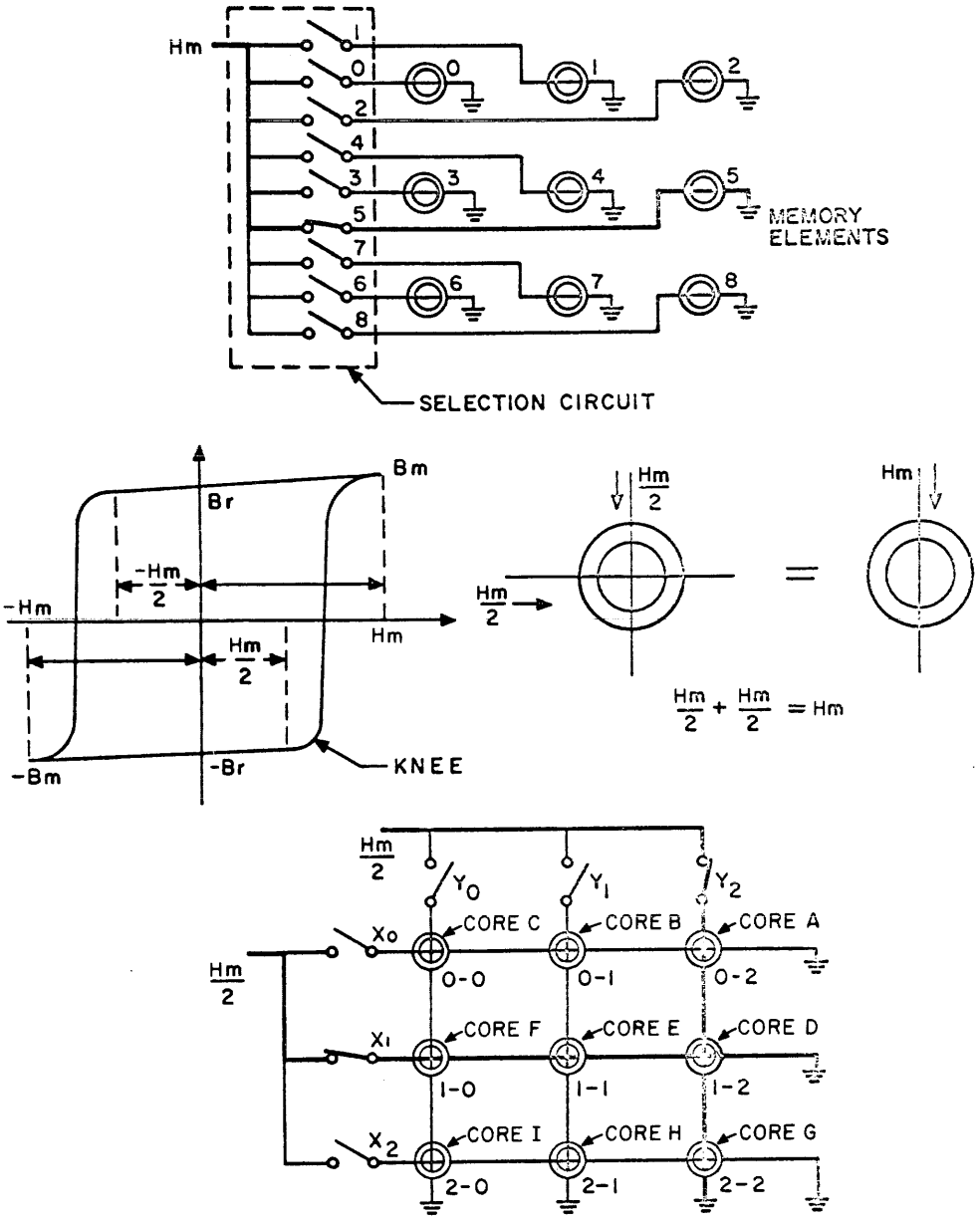


Figure 4-2-B. Memory Selection

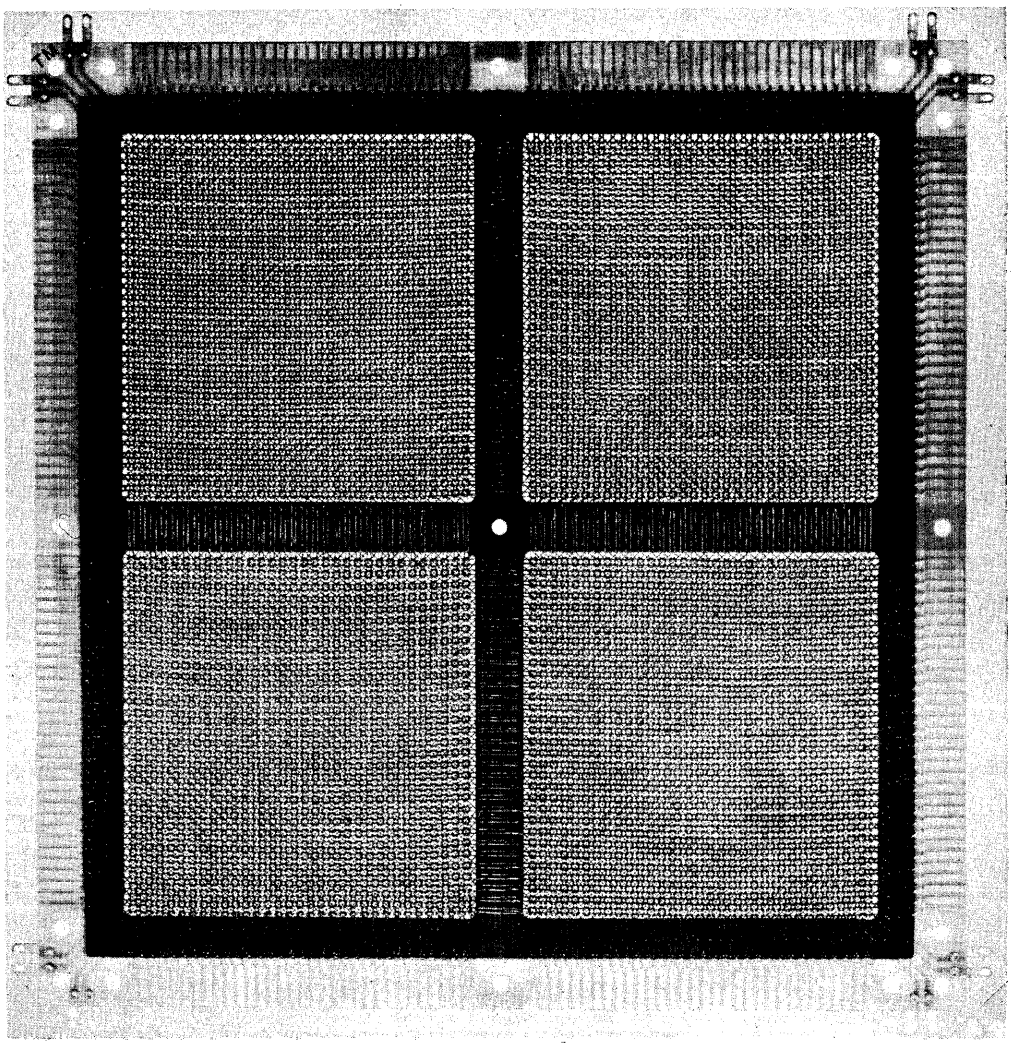


Figure 4-2-C. 128 by 128 Array, Top View

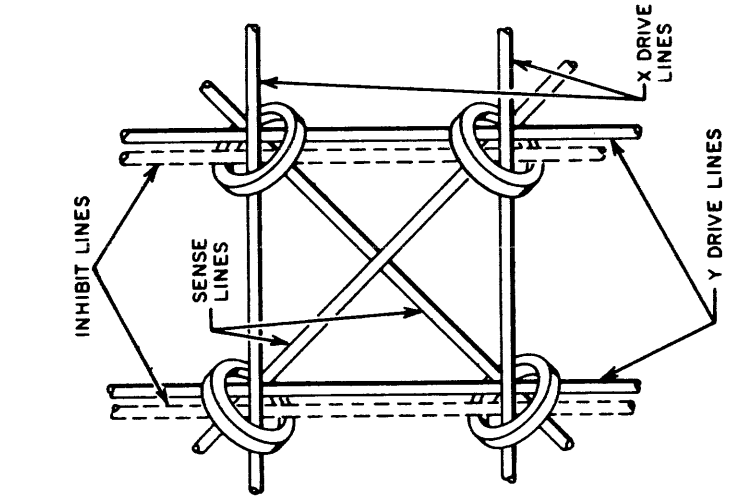
4-2-D for a simplified view of drive line intersection.

The two basic problems of address selection are: (1) selection of an X and Y drive line in each stack, and (2) selection of the inhibit line that is in the same quadrant as the selected drive lines. All of the three selections ar are made by outputs from the S translator.

A memory board is required to store all the possible address for one bit. The vertical conductors defining cores along the horizontal (X) axis are called X drive lines. The horizontal conductors defining cores along the vertical ((Y) axis are called Y drive lines (128 x 128 array).

The X and Y drive lines terminate at tabs along the edges of the board. (Refer to Figure 4-2-E for a simplified view of drive line connections.) A drive line connected to a tab on one edge of a board terminates on a tab at the opposite edge. A Memory board has four inhibit windings and four sense windings that are brought out to solder terminals at the four corners of the board.

A memory board is divided into four quadrants numbered from the lower left, as shown on Figure 4-2-C. A quadrant on an actual board contains a 64 x 64 array of cores (See Figure 4-2-F). Each quadrant has its own sense winding. An inhibit winding is threaded vertically in quadrants 1 and 2, and horizontally in quadrants 3 and 4. This method of threading the inhibit lines equalizes the loading effect on the drive lines by the inhibit lines. A sense line is also threaded through all of the cores in a quadrant. The sense line is oriented within a core so that maximum voltage is induced in the sense line when a core is switched from a "1" to a "0". Refer to Figure 4-2-G, views B and C, for typical inhibit line and sense line orientation respectively.



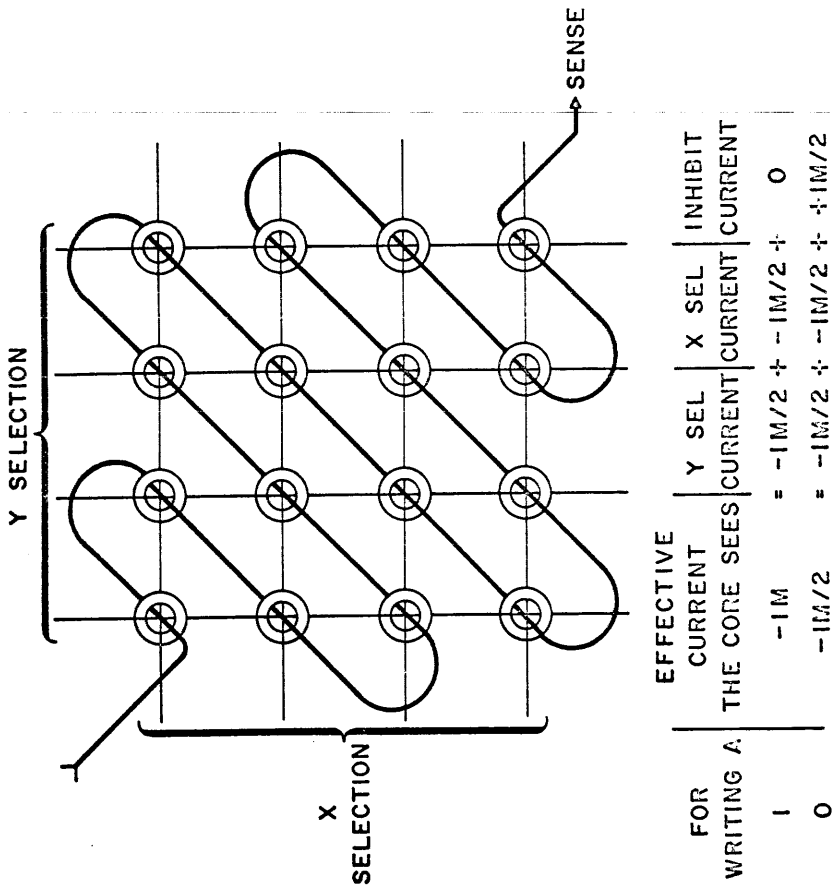


Figure 4-2-D. Sensing and Inhibiting Functions

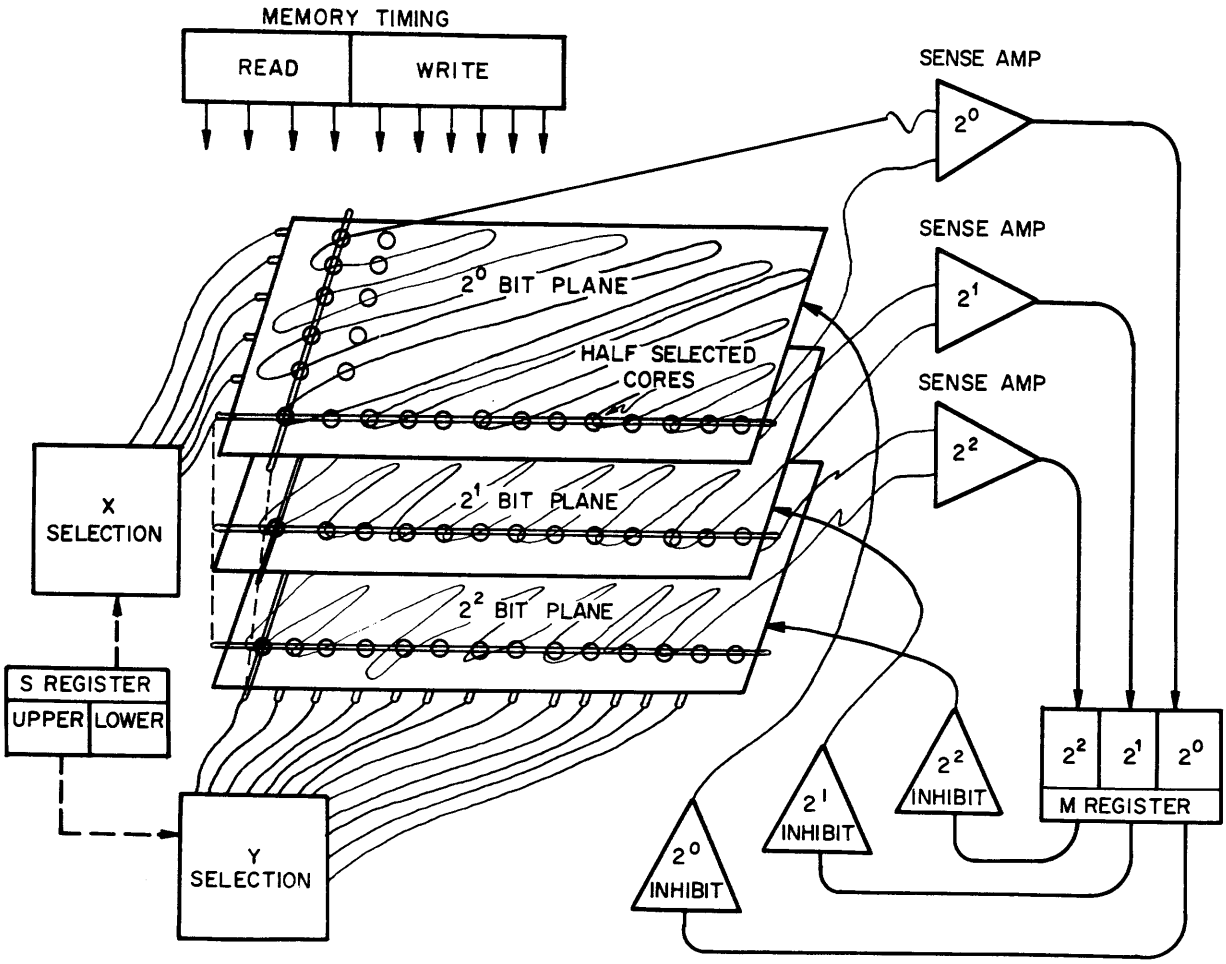


Figure 4-2-E. Three-Bit Memory

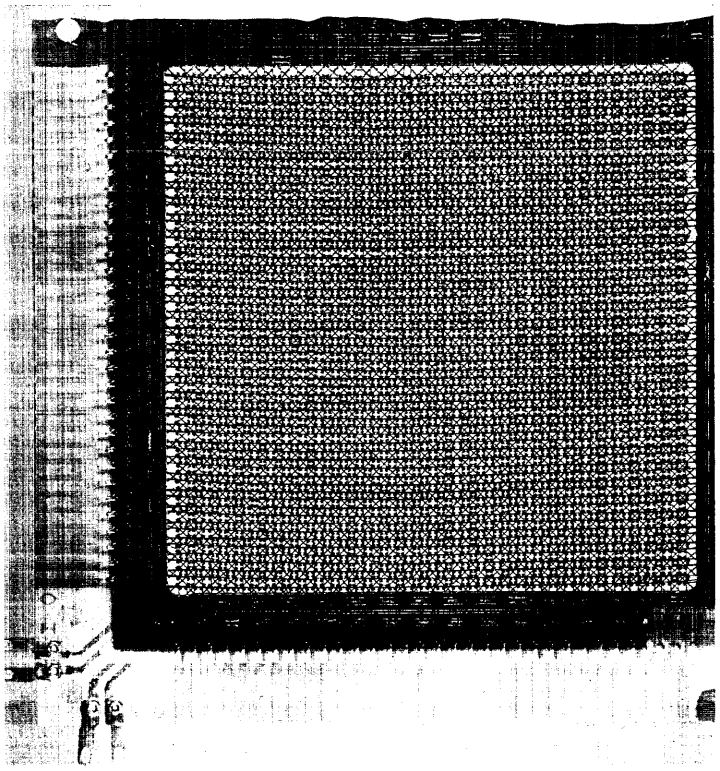
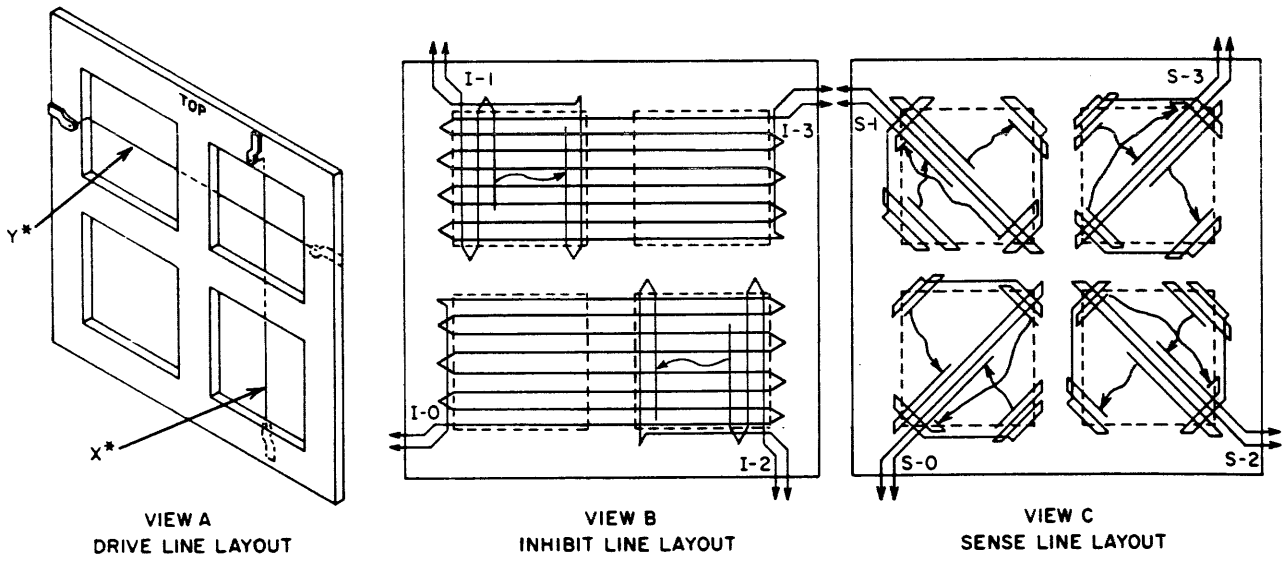


Figure 4-2-F. Quadrant 64 by 64 Array, Top View



* DRIVE LINES EXTEND FROM TABS ON THE FRONT SIDE OF THE CENTER BOARD TO THE CORRESPONDING TAB ON THE BACK SIDE AT THE OPPOSITE EDGE

INDICATES CONTINUATION OF PATTERN.

Figure 4-2-G. Sense Line Orientation

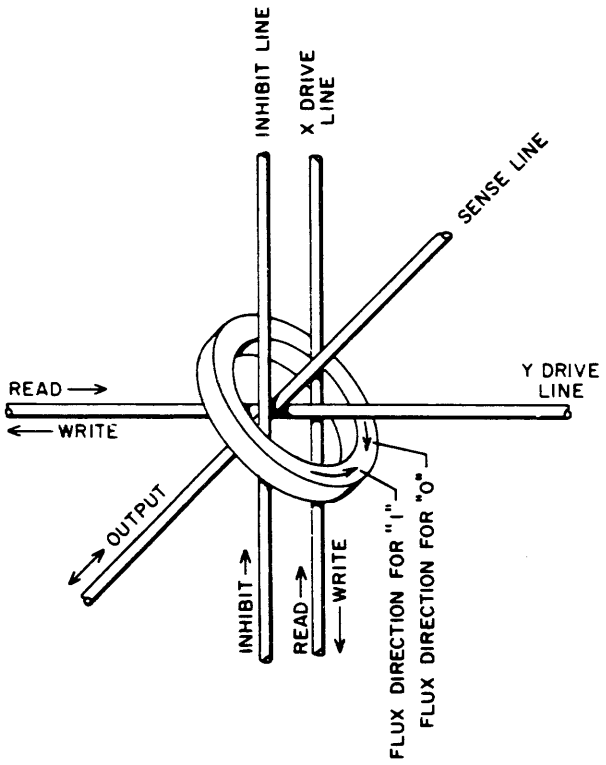


Figure 4-2-H. A Typical Ferrite Core, Inhibit Line X-Oriented

b. DPS-2402 Memory Section Detailed Analysis. - The following topic describes in detail the operation of the DPS-2402 Memory Section.

(1) S-Register. - Any computer function requiring access to memory must first transmit the address of the Memory register to be referenced into the S-register. During the memroy reference, outputs from the S-translator enable the proper X and Y drive lines and the proper inhibit lines. The S-register and translator are shown in Figures 9M2, 9M3, 9M4, and 9M5. The S-register is comprised of RS flip flops and, therefore, either a clear signal must preceed data transfer or special transfer logic is required. The S register is cleared by the master clear (Figure 9M1-A8) signal via SRRS, or MRRS; it can, also, be cleared manually by individual bit indicator lights via SRCL or by manual clear (MRRS) from the manual "read" or "write" switch operation.

The S register communicates with the P register, the arithmetic adder, and the Input/ Output Z register through the use of a unique method of data transfer. The following discussion considers the transfer of a single, representative, bit (P 10) from the P register to the S-register (Figure 4-2-I). Assume that P10 is equal to a logical "1" and that the required TRPS signal is present. (Figures 9C16 and 9C17 shows the transfer logic for all 13 bits.) Under this condition signal ST10 is a logical "0" which causes the output of gate SC10 to be a logical "1". When the signal DCLS (P-S), from the control section, arrives at the inputs of gates SC10 and SS10, gate SC10 is not affected because of the logical "0" from ST10, hence SC10 outputs a logical "1". However, gate SS10 is fully enabled, and the resulting logical "0" sets the S register bit.

In the case where P10 is a logical "0" ST10 remains at a logical "1" which fully enables SC10 during the DCLS commend. At this time SC10 disables SS10 and the S register bit is cleared.

Note that gate SC10 is used in the "or" capacity and allows data to be transfer from the I/O Section (Z-register), by the ENZA enable, in the same manner as previously discussed for the P register.

In addition, the S-register may be manually set via the indicator switches on the maintenance panel. When one of these switches is depressed a signal ground is applied to the set input (pin 8) of the selected register flip-flop.

(2) Address Translation. - The address translator is shown in Figures 9M2, through 9M3. The function of this translator is to select a unique X and Y drive line for each value (address) placed in the S-register.

The problem is refined to selecting one of 128 X-drive lines and one of 128 Y-drive lines. Since the selection of the X and Y lines are identical, only the Y selection is discussed. Figure 4-2-J provides a block diagram presentation of this implementation. The Y-selection translator divides the seven S bits, required for Y selection, into a group of four bits (0-3) and a group of three bits (4-6). The group of four is translated directly to select one of sixteen (16) switches which enables a column of eight selection transformers. The group of three is indirectly translated to provide the selection of one of eight sets of primaries on a load sharing matrix. The selected set of primaries induces a voltage in the corresponding secondary which feeds a row of sixteen selection transformers. The unique Y drive line thus selected is the one driven by the selection transformer located at the intersection of the selected column and the selected row.

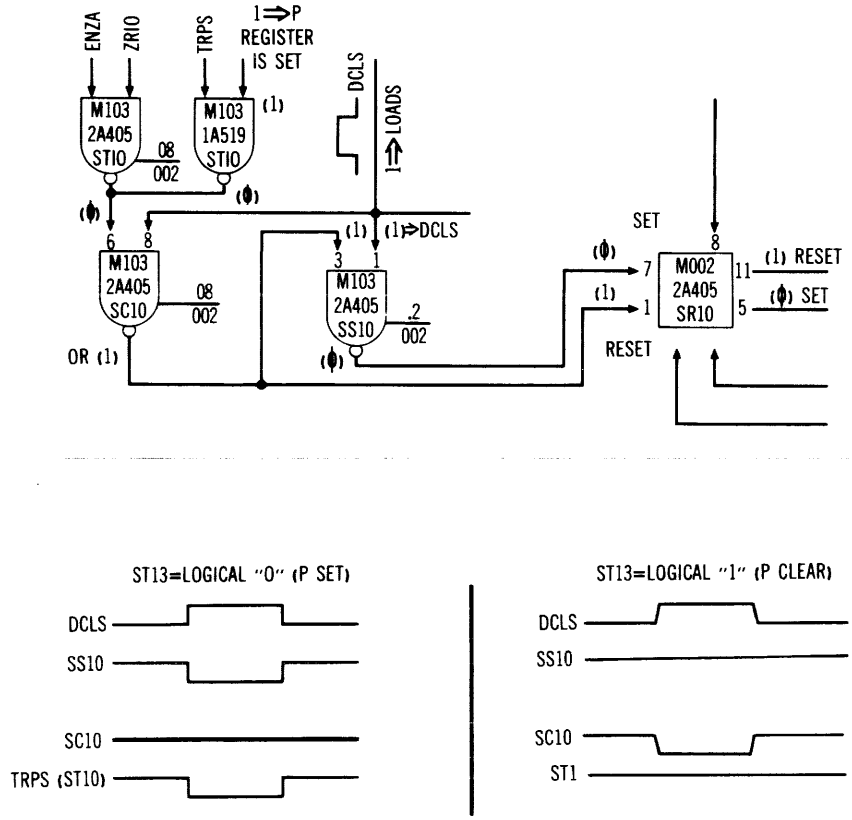


Figure 4-2-I. Transfer Logic

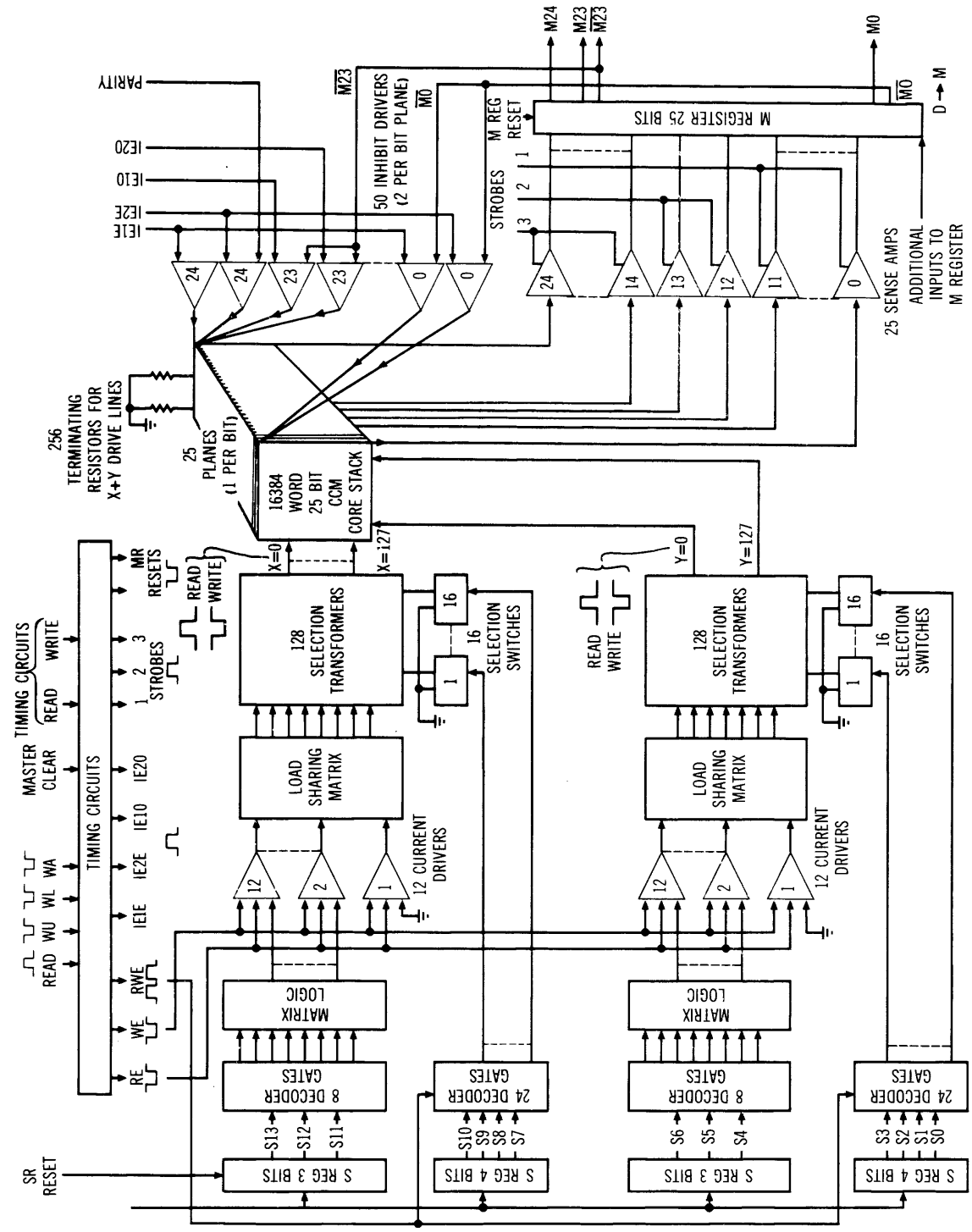


Figure 4-2-J. 2402 Memory Unit 2-μsec Cycle Time

There are 128 selection transformers which make up a 16 x 8 matrix (i. e. , 16 columns of eight transformers and eight rows of 16 transformers). Thus the required 1:128 (16.8) selection capability is obtained.

The logic shown in Figure 9M2-A illustrates how the 1:16 selection is obtained from the S register bits 0-3. Sixteen YS (YS01, YS02, . . . , YS16) signals are possible. However, only one of the sixteen can be a logical "0" at any given time, and only then if the enabling signal, RWEN, is a logical "1". For example, if RWEN is a logical "0", all sixteen YS signals are a logical "1"; if RWEN is a logical "1" the selected YS signal is a logical "0" and the other fifteen YS signals are a logical "1". For each of the sixteen possible values which these four bits (0-3) can exhibit, a different YS signal is enabled. For example, a value of 0000_2 enables YS01 and a value of 1000_2 enables YS09.

Each YS signal enables a particular column of selection transformers (Figure 9M7) by placing a logical "0" at the input of the respective switch. A logical "0" at the input of one of these 16 switches causes a logical "0" output which enables the respective column of selection transformers by grounding the center tap of each transformer.

Figures 9M3, 9M6, and 9M7 illustrate how the three S-register bits (4, 5, and 6) are translated to select one of eight rows of selection transformers. The initial translation is accomplished by a group of gates which allow five of eleven YC signals (YC02 through YC12) to go to a logical "0" for each value which bits 4, 5, and 6 exhibit (Figures 9M3 and 4-2-K). For example, if bits 4, 5, and 6 are 101 respectively (or equal to 5_8) YC02, YC05, YC07, YC08, and YC09 are a logical "0"; the other six YC signals remain a logical "1".

Each of the eleven YC signals go to a separate current driver (Figure 9M6-CD). A twelfth current driver has a constant ground (i. e. , a logical "0") at its J-input. The function of these twelve current drivers is to generate current pulses which then pass through primaries on the load sharing matrix.

During the read/write cycle the six current drivers with a logical "0" on their J input, generate current pulses during the read portion of the cycle. The other six current drivers have a logical "1" on their J input and, therefore, generate current pulses during the write portion of the cycle.

		3 BIT REGISTER VALUE (OCTAL)							
		0	1	2	3	4	5	6	7
LINES SELECTED	YC02	X	X				X		
	YC03	X				X			X
	YC04				X			X	
	YC05			X			X		X
	YC06		X			X		X	X
	YC07	X			X		X	X	X
	YC08			X		X	X	X	
	YC09		X		X	X	X		
	YC10	X		X	X	X			
	YC11		X	X	X				X
	YC12	X	X	X				X	

Figure 4-2-K. 3-Bit Selection

Figure 4-2-L is a schematic of the current driver used in the memory section. The current driver is turned on when transistor W conducts. Transistor W conducts only during the read or the write portion of the read/write cycles since a -12 volt enabling signal is needed to turn on the transistor. If a logical "0" is at input J, the -12 volt read signal indirectly turns W on; under this condition the -12 volt write pulse has no effect on W. If a logical "1" is at input J, the -12 volt write signal indirectly turns on W; under this condition the -12 volt read pulse has no effect on W. Since the input at J is either a logical "1" or a logical "0" during the entire read/write cycle, a given current driver is turned on (i. e. , transistor W) only for read or for write not for both.

Eight transformers make up the lead sharing matrix (Figure 9M6). Each transformer has 12 primary windings (one per current driver) and one secondary winding. The primaries of each transformer are so connected to the current drivers that 6 primaries receive current in one direction while the other 6 receive current in the opposite direction. This concept is illustrated by the dots shown on Figure 9M6-6A. Each dot indicates the polarity of the respective primary.

The polarities of all primary windings in the load sharing matrix are permanently arranged so that only one of the eight transformers can induce a voltage in its secondary during any

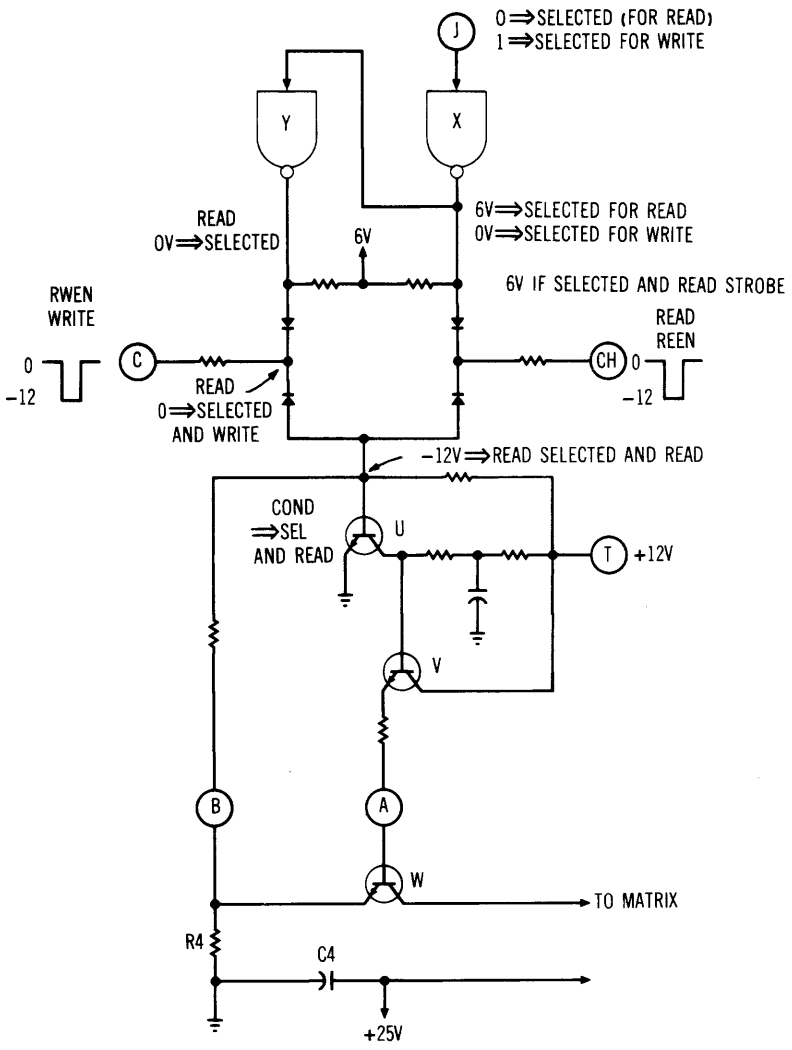


Figure 4-2-L. Current Driver

given read/write cycle. This selection of one transformer out of eight is possible since the polarities of a given transformer add algebraically. To further illustrate the point consider a read/write cycle. During the read portion of the cycle 6 primaries of each transformer receive current from the current drivers. In the selected transformer the current in all 6 primaries is in the same direction (i. e. , of the same polarity). Since polarities add algebraically there is a net current which induces a voltage in the respective secondary. In the seven non-selected transformers, 3 of the 6 primaries receiving current receive it in one direction while the other 3 receive it in the opposite direction. As a result there is no net current and consequently no voltage induced in the secondary windings. During the write portion of the read/write cycle, the 6 primaries of each transformer receiving current are the ones that did not receive current during the read portion. The transformer selected during the write portion of the cycle is the same one that was selected during the read portion. The transformer selection process is the same as it was for read.

As far as the load sharing matrix is concerned the difference between read and write is

in the direction in which the 6 selected primaries gate current through the selected transformer. For read the net current is gated in one direction; for write it is gated in the opposite direction. Therefore, the polarity of the voltage induced in the selected transformer's secondary winding is in one direction for read and in the opposite direction for write. This is how the current is reversed in the cores for writing into memory.

The translation of the 3 S-register bits 4, 5, and 6 is completed by the 12 current drivers and the load sharing matrix. For each load sharing matrix transformer selected, a corresponding row of selection transformers is selected (Figure 9M7). There are 8 rows of selection transformers; each row is fed by the secondary winding of the corresponding load sharing matrix transformer.

During any given read/write cycle only one of the 128 selection transformers has an induced voltage at its input diodes and, at the same time, has its center tap grounded. This unique selection transformer is located by intersecting the enabled column and the enabled row of selection transformers. The current output of this selected transformer and a similar one for the X-drive line are used to pulse the selected core of each bit plane by the coincident current principle.

(3) M Register. - The M-register, shown in Figures 9M10 through 9M16, functions as a buffer between the memory section and the arithmetic, control and I/O sections of the computer. All information entering or leaving the memory passes through this register. The M-register is a 24 bit register that operates at the bit plane level, i.e., each stage of the M-register corresponds to a specific bit plane in memory.

The M-register can function to divide the 24 bit word into upper (LH), lower (RH), and address (2^{12}, 2^{13}) portions. The means by which this is accomplished is discussed in paragraph 4-2(6).

(a) M-Register Inputs. - Each stage of the M register may receive data from the core stack via the sense amplifiers, the wired memory, the D register, the -Z register, the P-register (lower 14 bits only), and the I/O Status Register.

Data transferred from the core stack is initially picked up by the sense windings of the various bit planes. Each bit plane has 4 sense windings, one per quadrant. These 4 sense windings provide the inputs to the respective preamplifier (Figure 9M10-4A). The preamplifier's output is passed through an isolation transformer and becomes an input to a sense amplifier. When the sense amplifier is strobed the data from the respective bit plane is clocked into the M-register.

(b) M-Register Outputs. - In response to the proper command the M-register outputs data to the I-register (control), the D-register (arithmetic), the Z-register (input/output), the X-register (input/output), the inhibit current drivers (memory) and the parity "tree" (control section).

(4) Memory Timing. - Memory timing is governed by a transmission-type delay line. Once started the complete read-write memory timing cycle must run its course without interruption. A complete cycle requires about two microseconds. The "read" portion which requires about half of this period always preceeds the "write" portion. Figure 4-2-M shows a timing relationship of all major memory commands.

Initiation of the memory timing cycle can be accomplished manually and by computer control.

(a) Manual Memory Reference. - Refer to Figure 4-2-N. The operator may initiate a manual read or write operation by depressing the READ or WRITE test switch on the maintenance panel. Assume the READ switch is depressed. The RS flip-flop RETS is set by the ground applied to input pin 1. Gate MRTR (9M1-C7) is partially enabled (pin 6) by the RETS logical "1" level.

The logical "0" from the reset side of RETS sets flip flop RWTP which was initially cleared by MASC. The signal at input pin E of RWTP is a logical "1" (because flip flop WRTS is cleared at this time) and therefore has no effect on the flip-flop's state. When flip flop RWTP sets, gate MRTR is fully enabled and the MRRS signal which clears the entire M register is generated the M register must be cleared prior to entering data into it since the associated input logic does not permit the transfer of logical "0"'s.

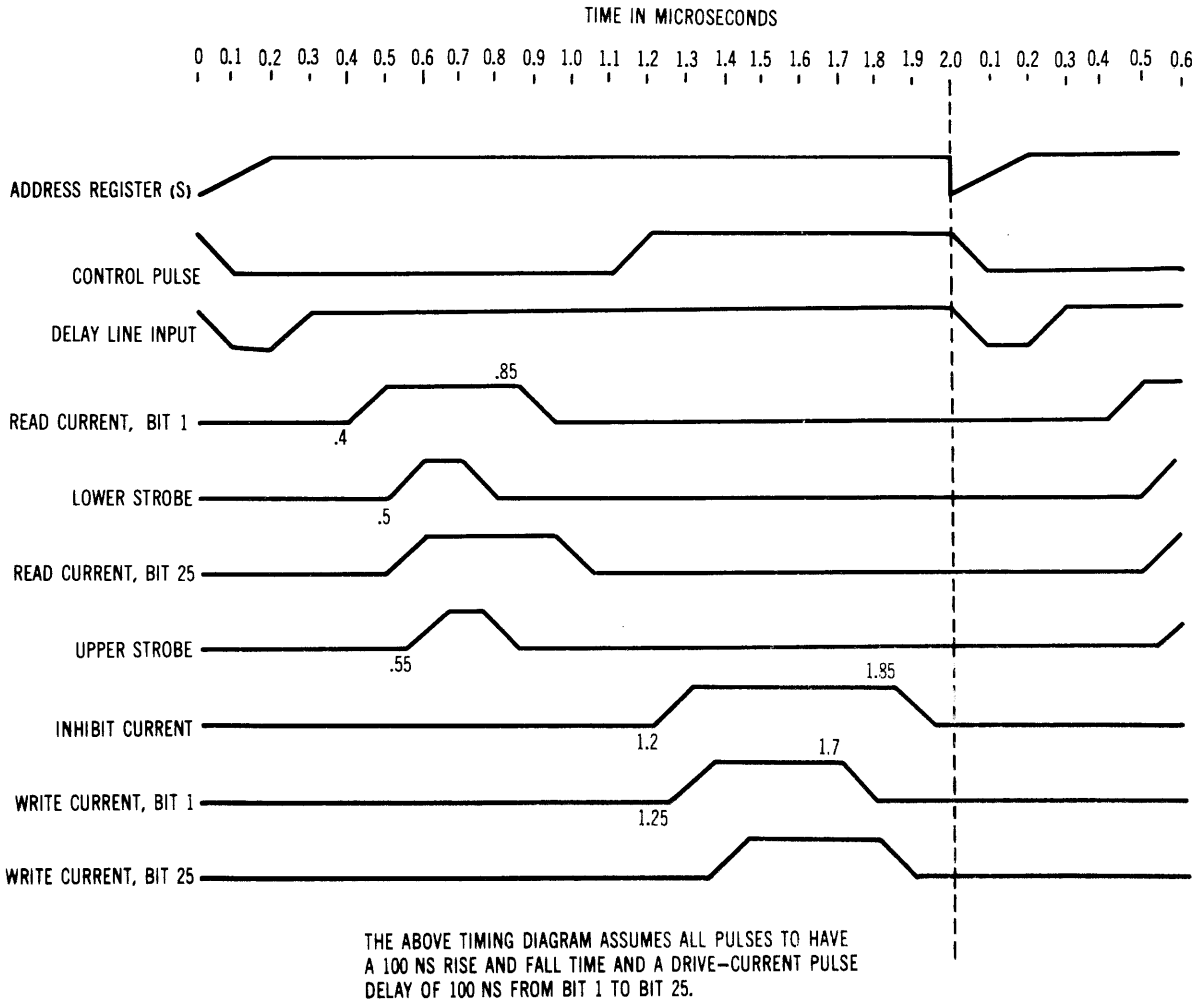


Figure 4-2-M. 2402 Memory Timing Diagram

The logical "1" from the set side of flip flop RWTP is inverted by RDTS, OR'ed with RDIN (which equals a logical "1") and inverted again by ILSE. The resulting logical "1" is combined with a DLEN signal. The DLEN signal indicates, by a logical "1" that a memory cycle is not in progress (Figure 9M1). If a memory cycle is not in progress gate DLDR is fully enabled and a logical "0" is entered into the delay line which starts the memory cycle. The pulse travels down the delay line and 150 ns later, sets the DLEN flip flop which blocks any succeeding memory cycle requests and clips the delay line pulse to a width of 150 ns. At the 200 ns point, a signal resets flip flop RWTP via gate RUTH and thereby allows future manual requests to be honored.

(b) Memory Reference by the Computer. - The Computer Control and I/O timing circuits reference the memory by a 2-microsecond READ signal which combines with CL01 at gate RDIN (9M1-D7). The resulting RDIN Signal starts the delay line in the same manner as discussed in paragraph 4-2(6a).

(c) Read Portion of the Memory Cycle. - Once the delay line has been initiated, the 150 ns pulse sets flip flop DLEN, clears RWTP after 200 ns, and sets flip flop RERS. The output of RERS enables the read pulse REEN which triggers the current drivers shown in Figure 9M6. RERS, in conjunction with DLEN, produces the RWEN signal (Figure 9M1-B4) which enable the translation gates associated with the group of 4 bits (0-3) from the S-Register (Figure 9M2). Flip flop RERS is cleared (85 ns) marking the end of the read time period.

As the cores are switched by the read current a voltage is induced in the sense windings which thread through the bit plane. These signals are strobed in the sense amplifiers by the strobe signals STB1, STB2 and STB3 (9M1-A6). STB1 clocks data into M-register bits 0 through 11; STB2 clocks data into bits 12 and 13; and STB3 clocks data into bits 14 through 24. All three strobe signals are triggered by the delay line DL1 providing one of the CCW (CCW L, CCW A, and (CCW U) inhibits does not exist.

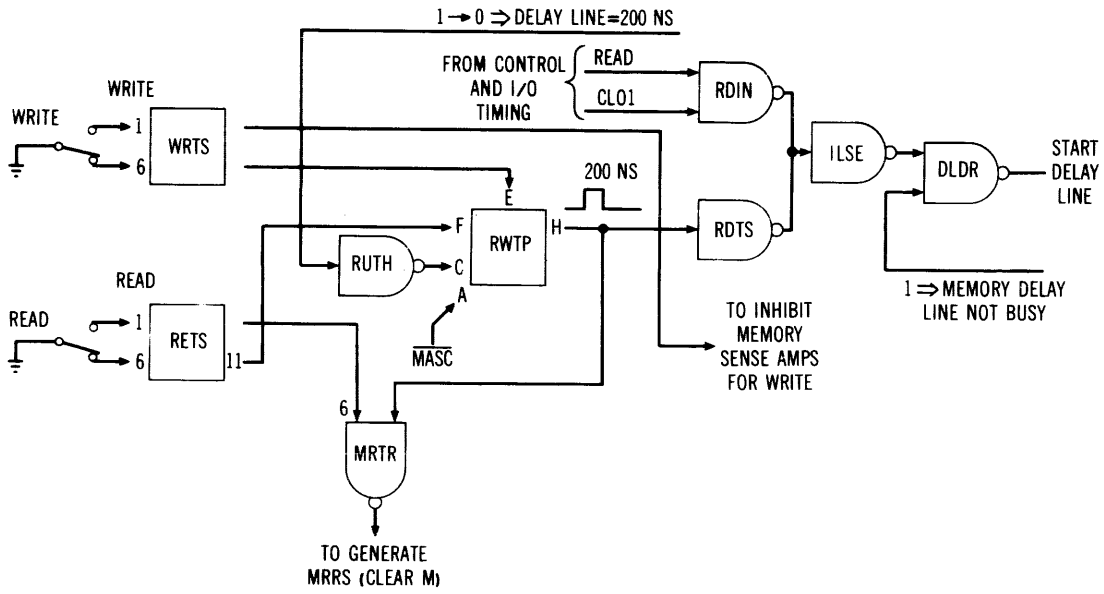


Figure 4-2-N. Manual Memory Reference (Read-Write Test Switch Logic)

The only time a CCW inhibit can exist is during a memory store operation. Its sole purpose is to prevent data from being relocked into the M-register which currently holds the data to be stored. If a complete 24-bit word is to be stored, all three inhibits exist during the read portion of the read/write cycle. As a result the three strobes are blocked; unwanted data in not clocked into the M-register, and the selected memory address is cleared.

The three different CCW signals (and therefore the three strobe signals) provide for the storing of partial words in memory. When a partial word is stored the following events occur.

(1) The partial word is entered in the M-register.

(2) The entire 25 bits of the desired memory address is cleared. Clearing occurs during the read portion of the read-write cycle.

(3) The contents of that portion of the memory address to receive the partial word is prevented (inhibited by the appropriate CCW signal) from being clocked into the M-register. The rest of the contents of the selected memory address is clocked into the M-register.

(4) The entire 25 bits of the M-register is now written into memory at the selected address.

When an instruction calls for a partial word to be read out of memory, the system accomodates by reading the entire word (all 24 bits) into the M-register and then rewriting the entire word back in memory. The partial word is obtained by selectively gating it out of the M-register.

(d) Write Portion of the Memory Cycle. - About half way through the delay line (120 ns point) the write time begins. Flip flop IERS is set (Figure 9M1), and inhibit drivers are activated for those bits of M which are in the clear condition. The function of the inhibit drivers is to cancel the effect of the coincident write currents for those bit planes that are to have a logical "0" written into them. These bit planes, of course, are selected on the basis of the

ADDRESS RANGE	EVEN M BITS		ODD M BITS	
	IEE1	IEE2	IE01	IE02
X' = two lower bits arbitrary, upper bit zero				
000X'X00 - 0000X'X17	X		X	
0000X'X20 - 0000X'X37	X			X
0000X'X40 - 0000X'X57	X		X	
0000X'X60 - 0000X'X77	X			X
00004X00 - 00004X17		X	X	
00004X20 - 00004X37		X		X
00004X40 - 00004X57		X	X	
00004X60 - 00004X77		X		X
0001X'X00 - 0001X'X17	X		X	
0001X'X20 - 0001X'X37	X			X
0001X'X40 - 0001X'X57	X		X	
0001X'X60 - 0001X'X77	X			X
.				
.				
.				

Figure 4-2-O. Inhibit Selection

individual bit values in the M register. The inhibit driver selection gates (9M1-C1) are selected on the basis of content of the S register bits 2^4 and 2^{11}. This is done so that the inhibit task can be divided between two driver circuits. Thus, the length of the line (and the associated propogation delay) for each driver can be reduced by one half.

The output of IE01 and IE02 are used to select a set of inhibit drivers for the odd M register bits. While IE03 and IE04 are used to select the inhibit drivers for the even bits of the M register. Figure 4-2-O illustrates the pattern of inhibit driver sharing for a range of addresses. After the inhibit current drivers are selected for the bits of M which are reset, the 150ns pulse in the delay line has proceeded to the 135ns point where it sets flip flop WERS (9M1-B2). This flip flop, when set, generates WREN which enables the selected X and Y current drivers in the write direction.

Thus, those selected bit cores of an address which do not have inhibits present receive a write current pulse and are set to a logical "1" state. Those bit cores of the address for which inhibits are present contain logical "0". Remember, the destructive nature of the read process always clears all bit cores of the selected address and thus will remain logical "0" unless pulsed by a "write" current.

The memory cycle terminates with the clearing of DLEN and IERS at the 170 ns point and clearing of WERS at the 190 ns period of time. When DLEN is cleared, the delay line honors the next request for a memory access.

(5) Core Theory. - A typical magnetic core is shown in Figure 4-2-O. Its outside diameter is 0.050 inches; its inside diameter is 0.030 inches; and its thickness is 0.015 inches. The ferrite core is magnetized by the field produced by a current flowing in a wire that is threaded through the core. It retains a large amount of this induced flux when the current is removed. Flux lines are clockwise or counterclockwise around the core, depending upon the direction of the current. These two unique states are designated "0" or "1", depending on the orientation of the core with respect to the wiring. The time required to switch the core from one state to the other is approximately 1.2 microseconds with the current pulse used (approximately 400 milliamperes for two microseconds).

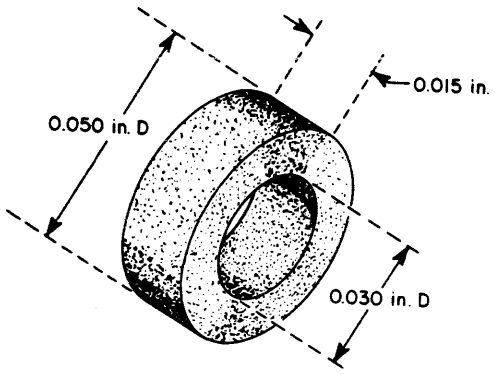


Figure 4-2-P. A Magnetic Core

The state of magnetization of a core is shown on the hysteresis diagram (Figure 4-2-Q) which plots magnetic flux density in gauss (B) as a function of the field (induced by the current) in oersteds (H).

The diagram shown in Figure 4-2-Q assumes that the core is already in some state of magnetization, such as A. If a current flow, with a direction (+) that produces a field (H) of a given magnitude (m), (i. e. , $+H_m$) is applied to the Drive line, the flux density increases to saturation as indicated by $+B_s$. If the current is removed, the flux density retained by the core drops slightly to the level indicated by $+B_r$ (remanence) or the residual flux density. This state is arbitrarily designated as the "0" state. Another pulse of $+H_m$ would now merely shift the core to $+B_s$ again, and after the pulse is removed, the core would return to $+B_r$.

When a current pulse of the same magnitude, but in the opposite direction ($-H_m$), is applied to the drive line, the flux density shifts along the curve to $-B_s$ causing a reversal of the flux density in the core. When the current pulse is removed, the residual flux density is at $-B_r$, the state that is arbitrarily designated as the "1" state.

Any change in the flux of a core induces a voltage in all windings passing through the core. Hence, the induced voltage on the sense winding is sampled to see if the core switches, with $+H_m$ applied, from $-B_r$ to $+B_s$. If a large induced voltage is sensed (over 50 millivolts), the core was in the "1" state and had been switched from

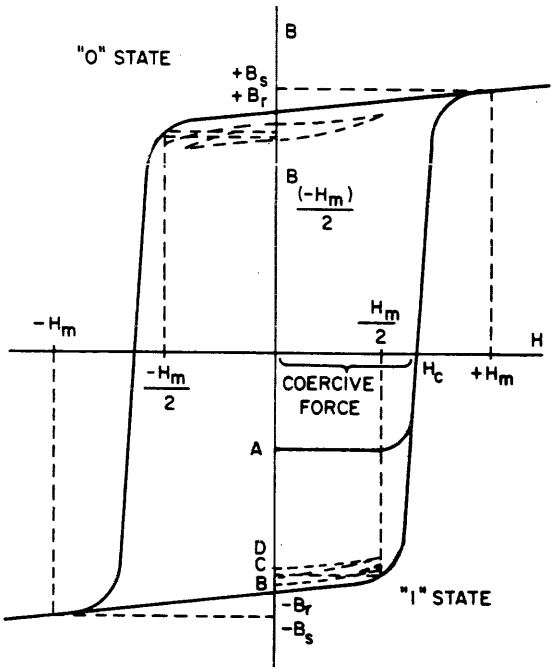


Figure 4-2-Q. Idealized Hystersis Loop of a Ferrite Core

$-B_r$ to $+B_s$. Because the content of the core is determined in this manner, the Current pulse corresponding to $+H_m$ is called a Read pulse. A memory utilizing this type of magnetic core storage element is referred to as a Destructive Readout memory. Therefore, a restore function is necessary to return the core to its original state.

When a core is in the "0" state is is possible by applying a pulse corresponding to $-H_m$, to enter a "1" into the core or by not applying the pulse, leave a "0". Because the data circuits and addressing circuits are separated, the $-H_m$ is applied unconditionally. When entering a "0" the field is partially cancelled by an inhibit pulse on the Inhibit line. An inhibit pulse is the equivalent of 1/2 ($+H_m$), a half-read pulse. A write pulse of net half-amplitude, 1/2 ($-H_m$) is not sufficient to switch the core to the "1" state.

The operation of memory depends on the ability of each core to distinguish between two current levels on its read/write windings (R/W drive lines). Each core in a plane is linked by four windings. Two of these windings, the X and Y drive lines, determine the address of the core. To operate on the selected core, half-amplitude pulses are applied to each selected drive line so the core at the intersection of the two selected drive lines is the only core that receives a net field, or full amplitude pulse, of H_m. All other cores on the two selected drive lines receive

only the half-pulse field associated with one drive line.

As seen in Figure 4-2-Q, the coercive force, H_c, is the field required to switch the core. The drive currents have been selected so that $\frac{Hm}{2}$ is less than H_c and, as a result, is insufficient to switch the core. But the sum of the two drive currents, $\frac{Hm}{2} + \frac{Hm}{2} = H_m$ is greater than H_c and switches the core in just over one microsecond.

When a core receives a half-current pulse, the field induces a change in the flux density of the core. Assuming that the core is in the "1" state at $-B_r$, a half-read pulse causes the flux to shift along the hysteresis loop to the point limited by $\frac{Hm}{2}$ and then return to a slightly lower remanent value, such as point B. Since the core is operating on a slightly smaller loop, further half-pulses again reduce the remanent flux. This effect soon reaches a limit as at point D. When the core is in the "0" state ($+B_r$), half-write pulses produce a similar effect.

The shift in flux, caused by half-current pulse, induces a small voltage on the sense winding. The amplitude of this voltage is a function of the squareness of the hystersis loop. The squareness ratio, R_S, is defined as the ratio of the flux density value at $\frac{Hm}{2}$ to that at $+H_m$. Values of R_S range from a practical limit of 0.7 to an ideal limit of 1.0. A typical value for the cores is about 0.9. This and other operating margins, such as slant, H_C, and B_S are ensured by grading. A core with a low R_S, or a greater slant, has a greater shift in flux for a given half-current pulse. The small voltage induced on the sense winding as the flux level travels along the "knee" of the hystersis loop to a slightly lower remnent value is a significant source of noise that tends to obscure the desired signal. This effect can be paritially eliminated by strobing the core output during the maximum "1" time (see Figure 4-2-R).

(6) Summary. - The operation of the memory section, described in the previous paragraphs, illustrated the methods used to store information for future use and the methods used to readout stored information for current use. The Memory section provides the storage area for routines that are to be executed by the computer, for data that is to be transmitted to external equipment, and for data that is sent to the computer from external equipment.

The Memory section provides signals in a fixed time relationship to accomplish its address selection and translation, its read/write functions, and its inhibit function.

The control, and I/O sections of the computer have independent access to memory, each section being able to initiate memory through its own control functions.

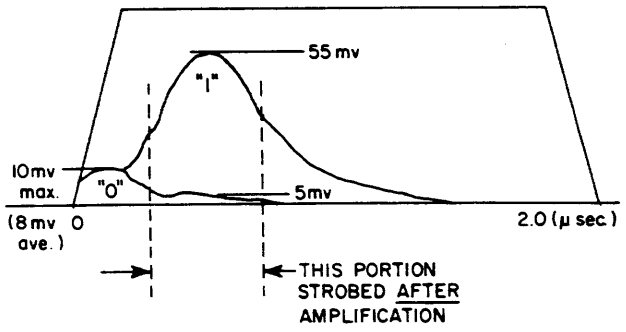


Figure 4-2-R. A Typical Magnetic Core Output

4-3. CONTROL SECTION OPERATION.

a. General. -The control section governs all computer operations except certain Input/Output functions and detailed execution of certain arithmetic operations.

Figure 4-3-A shows a block diagram of the control section.

(1) Control Section Registers. - Registers contained in the control section consist of a program register (P) and an instruction register (I). In addition, three index modifier (B) registers, which utilize memory locations, are used by the control section for address modification.

The program register (P) holds the next instruction address that is to be executed by the computer. Built into the register is a facility for incrementing by one which provides for sequentially executing a list of instructions (program).

The function of the I register is to hold the instruction while it is being executed.

(2) Timing Control. - A major sequence timer, consisting of six flip flops, provides overall timing control. These flip flops are IA00, ID00, MA00, OA00, IN00 and MS00. Each are normally set for two microseconds and then cleared. No more than one of these flip flops can be in the set state at any given time.

IA00 - The Instruction Acquisition flip flop is set at the beginning of every instruction. The instruction in the P-register address is acquired from memory and placed in the I register for interpretation. The sequence flip flops which are set during the next 2 microsecond period is dependent entirely on the particular instruction in the I register. Only those sequences flip flops required to execute the instruction are set depending on the I register content after the IA00 cycle.

ID00 - The Indirect Addressing sequence flip flop is set when the instruction in the I register is an indirect type and bit 23 is set. As a result of the indirect operation, the contents of the Y field address replaces the Y, B, and I fields of the current instruction in the I register.

MA00 - The Index Modifier sequence flip flop is set when the b designator of the instruction word in the I register is non zero. The indexing operation causes the content of the specified B register to be added to the Y field of the instruction.

OA00 - The Operand Acquisition sequence flip flop is set when the operand is being read from memory and placed in the D register.

IN00 - The Intermediate sequence flip flop is set for certain instructions which require an extra delay time period for operation.

MS00 - The Memory Store sequence flip flop is set when a memory store is being performed. The contents of the D-register are transferred to M and read into core storage.

(3) Instruction Word Translator (I Translator). - A decoder (translator) provides for interpretation of the I register in order to determine the particular instruction being executed.

(4) Parity Check Logic. - The contents of the M register are examined during each memory read operation and compared against a parity bit in order to determine if a parity detectable error occurred in Memory.

(5) Sequencer Control. - The sequencer control governs the starting and the holding up of the sequencer. The major sequences are delayed until a long arithmetic (shift, scale, multiply, etc.) operation is completed and while the IO section has control of memory.

(6) The Auto Load (Bootstrap) Logic. - This logic is used only at the beginning of a program loading operation to begin the program loading process. The auto load executes an initial wired program when initiated by depressing the AUTO LOAD switch on the operator panel.

(7) Instruction Execution Control. - The necessary major commands required to execute most of the instructions originate from this subsection. The logic combines timing signals with instruction (I register) translations to form the command enables required for computer operation (Figure 9T).

(8) Fixed Location Gates. - Several of the control section operations require that a fixed value be placed into the S-register. For example, b modification or indexing requires that the address of the designated B-register be placed in the S-register in order that the B-register's content may be taken from memory and placed in the D-register where it can be added to I_Y.

b. Detailed Description of Control Section Operation.

(1) Components

(a) P-Register. - The P-register, a fourteen bit program register, is shown in figure 9C16 and 9C17. It's flip flops are of the J-K Type. The P register serves as a counter for

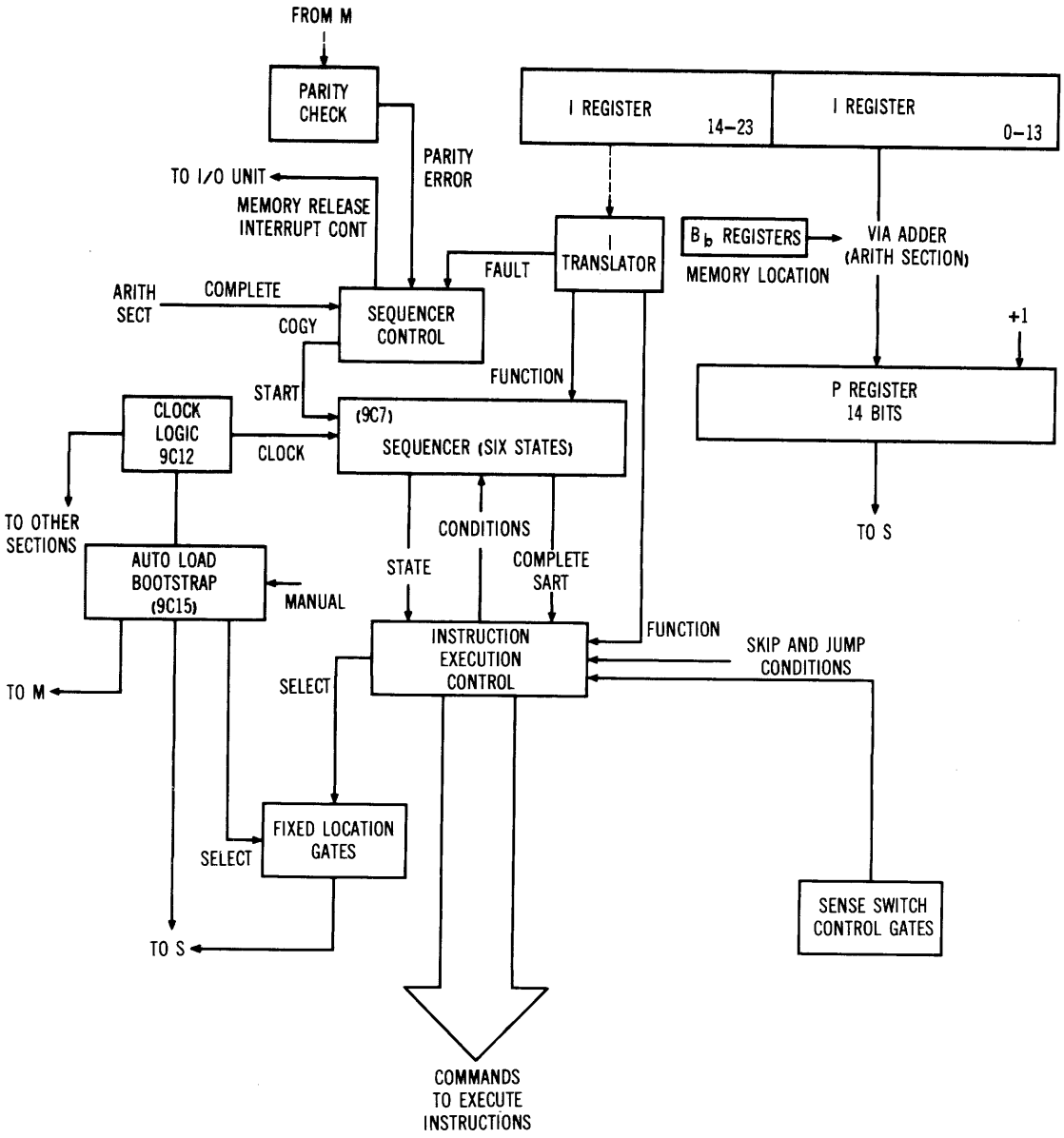


Figure 4-3-A. Control Unit Block Diagram

incrementing the program address by one, thereby allowing the computer to sequentially execute a list of instructions. The feature that permits J-K flip flops to operate in an AC rather than DC manner makes possible the simple ripple-carry incrementing circuit used in the P register. Incrementing the P-register's contents is accomplished by the pulse INCP. Each INCP pulse changes the state of PR00; every second INCP pulse changes the state of PR01; every fourth INCP pulse changes the state of PR02; every eighth INCP pulse changes the state of PR03 and etc. (see Figure 4-3-B). In order to observe how incrementation occurs refer to figure 9C16 and assume the P register is initially cleared. The first INCP pulse (trailing edge) changes the state of PR00; inputs C and F are floating and are equivalent to a logical "1". No other flip flop is effected by this pulse since inputs C and F of all even numbered flip flops are a logical "0" and the odd numbered flip flops do not receive the INCP pulse. After PR00 is set, inputs C and F of PR01 are a logical "1" and gate PRAA is partially enabled (pin 7). The content of the P register is now 000----001_2. The second INCP pulse changes the state of PR00 to the reset condition. As the voltage of pin H of PR00 drops, flip flop PR01 changes its state to the set condition. No other flip flop is effected. The contents of the P-register is now 00----010_2. Gate PRAA is not yet fully enabled because pin 7 has returned to logical "0" by virtue of PR00 being in the clear state.

The next pulse again changes the state of PR00 to the set condition but has no effect on any other flip flop. The content of the P-register is now 000----011_2. At this time, gate PRAA is fully enabled and outputs a logical "0" to PRAB which inverts the signal and places a logical "1" on inputs C and F of flip flop PR02. The next (fourth) INCP pulse sets PR02 resets PR00 which inturn resets PR01. No higher stage is effected and the contents of the P-register is now 000---0100_2. The fifth INCP pulse changes only PR00 and the content of the P-register changes to 00---0101_2. The same incrementing process continues as INCP pulse are received.

Data may be transferred to the P-register from the outputs of the adder (BSi) only. See Figure 9C16. The data transfer is accomplished by first clearing the P-register and then setting those P-register flip flops which correspond to a logical "1" in the respective adder bits.

Figure 4-3-C is a cleaned up version of Figure 9C16-B1 which illustrates how the P-register can be cleared. The RESP signal is used to clear the P-register prior to loading data into P. The logic which generates RESP is shown in Figure 9C5-C3.

Since the P-register is cleared prior to transferring data to it, only those bits which are a logical "1" need to be transferred. Consequently only those stages receiving a logical "1" have their input gating fully enabled by the transfer signal LOSP (Figure 9C16-D). The output of each fully enabled gate is a logical "0". Thus, a ground is placed on the respective flip flop's reset output (i. e., the nonconducting transistor's collector). Grounding the reset output switches the flip flop to the set condition and completes the data transfer.

To further illustrate how data is transferred to the P-register, consider the case where a logical "1" is to be transferred from the least significant bit of the adder to bit one of the P-register. REST is forced to a logical "0" by RESP. As REST goes to ground level the P-register is cleared.

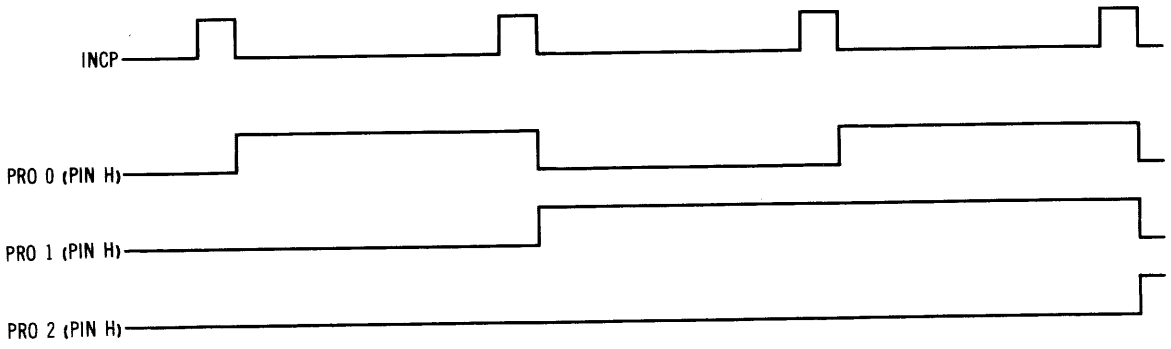


Figure 4-3-B. Incrementing P-Register From 0 - 4

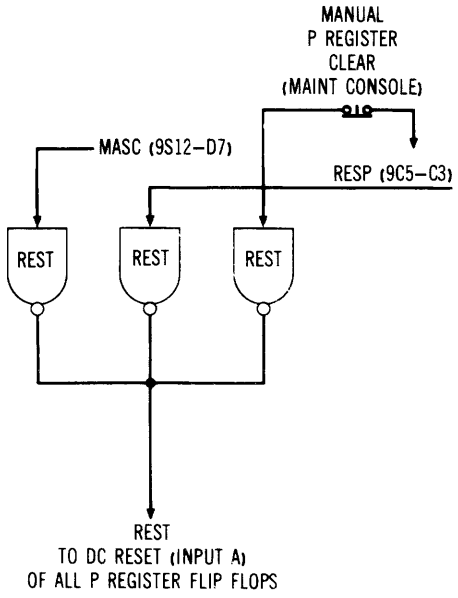


Figure 4-3-C. P-Register Clearing Logic

The P-register transfer signal LOSP, in conjunction with BS00, fully enables gate 16D1B. The output of gate 16D1B is a logical "0" which grounds pin J of flip flop PR00. A ground at pin J (reset output) switches PR00 to the set condition and the required data transfer has been accomplished.

(b) I Register. - The I register (Figure 9C9) is a 24 bit register which holds the instruction while it is being translated and executed. The I-register consists of RS flip flops, and the sole input to it is from the Memory Buffer Register (M).

The I-register is loaded by the series of gates shown in the upper left hand corner of Figure 9C9. The I-register is cleared on clock CLE1 (or clock CL01 for enable 1A00) and data is transferred from M on clock CLE2. Control transfer signals IA00 and ID00 separately initiate a clear load sequence. For example, assume the IA00 enable is present. During clock CL01, gates DIRS and DIRG are both enabled and the entire register is cleared by the resulting logical "0" which grounds the reset inputs (pin 7) of all flip flops. During phase CLE2, gates DRIV and DROV (Figure 9C9-7C) are enabled and, in turn provide the necessary gating signals to the M register's input gates.

Assume next that the ID00 enable is present. During CLE1, gate DIRS is enabled which clears the lower 16 bits (0-15) of the I register. The 2^{23} bit (IR23) is unconditionally cleared by ID00 via gate IR23 (Figure 9C9-8C) by pulling down the set output (collector) voltage of IR23. IR23 is not loaded during the ID00 enable, and bits 2^{16} through 2^{22} are not disturbed but retain their original contents.

The next clock CLE2 fully enables the gate DRIV (Figure 9C9-D8) and loads bits 2^{0} through 2^{15} from the M-register.

Output connections from the I-register includes lines to the arithmetic adder and the Instruction translator.

The manual clear logic for the I-register is shown on Figure 9C9-8B.

(2) Instruction (I) Translator (Decoder). - The function of the instruction translator is to determine which instruction is in the I-register and generate the necessary command enables for proper execution of the instruction.

Translation of the I-register contents is accomplished in two steps. Decoder A (Figure 9C10) decodes the various K and b designator bits of the instruction word and partially translates the function codes (most significant six bits of the instruction). Decoder B (Figure 9C11) completes the translation of the function codes.

The outputs of Decoder A are designated as indicated by the following examples:

CS4X or 0⇒S = 4x (logical 0 when S = 4x)

Indicates logical "0" when selected

Indicate selected lines for the six bits are octal 4 for first digit and any octal number for second digit

Indicates the function code bits

DBE1 or 1⇒B = 1 (logical one when B = 1)

Indicates logical "1" when selected

Indicates the value of B which is the selected value

Indicates the designator B

RS06 or 1⇒S = x6 (logical "1" when S = x6)

Indicates function translator

Indicates right octal digit is 6 left most octal digit

Decoder B (Figure 9C11) symbology is illustrated by the following example:

CDIV or 0⇒DIVIDE (function code = 27_8)

Indicates logical "0" when selected

Instruction code which selects the gate

DSSW or 1⇒SSW (function code = 07_8)

Indicates logical "1" when selected

Instruction code which selects this gate

(3) Parity Test Logic. - The parity test logic function is to detect errors in memory. During WRITE the word in the 24-bit M-register content is examined and its parity determined by the parity logic. A parity bit is generated on the basis of this determination and stored in memory along with the original data word. The parity is generated such that the parity of the combined parity bit and original word is "even" parity.

During the READ process, the content of the entire 25-bit M-register is checked for odd parity. If odd parity is detected, an error has occurred and a parity error interrupt is generated. The exact type of parity interrupt depends on whether an I/O operation is in progress. If so, an I/O parity error results. Otherwise a program parity error is generated. The parity logic is shown in Figure 9C14.

Three stages, all utilizing the same basic parity element (see Figure 4-3-D), are used to determine M-register parity. The basic parity element consists of five NAND gates, one as an inverter and four of which serve as the odd parity determination. Alternatives for three inputs designated A, B, and C which make the three odd parity are:

1. $A\ B\ C$
2. $\overline{A}\ \overline{B}\ C$
3. $A\ \overline{B}\ \overline{C}$
4. $\overline{A}\ B\ \overline{C}$

The resultant parity of each element is inverted and both true and false inputs fed into one of the three inputs of the next parity stage in succession. (See Figure 4-3-E) The output of the final element is indicative of M-register (24 data bits) parity. This parity information is utilized as shown in Figure 4-3-F during "read" to determine parity errors. The parity is checked against the parity bit 2^{24} and the resultant parity is odd if no parity error occurred.

This same logic structure is used during "write" to generate the proper value of the parity bit written into memory.

(4) Automatic Load Control (Bootstrap). - The "wired memory" of the computer consists

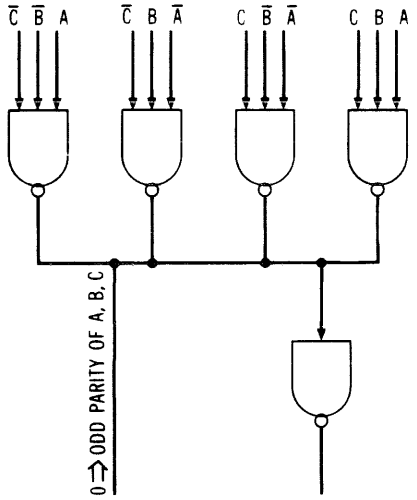


Figure 4-3-D. Basic Parity Element (Three Inputs Checked)

of four pseudo instructions used to initiate the loading of programs. The present wired memory is set up to load a punched paper tape on channel 0 in the assembled (full 24 bit) mode. These pseudo instructions are:

Step 1. Enter 36000000 in address 00000

2. Enter 0000004 in X register (External function code). Establish Input buffer with interrupt on channel 0.

3. Send External Function on channel 0 to call for READ ASSEMBLED MODE. Write 00000040 in address 40 (buffer control word).

The above functions are performed in three discrete steps which are accomplished in sequence and governed by a two-flip flop timing chain (Figure 9C15-3D).

At the beginning of the load process the paper tape is positioned in the reader with the starting point just under the read head. The computer is master cleared. Flip flop LOAD (Figure 9C15-4D) and the timing flip flops I and II are all cleared by master clear. Depressesing the bootstrap load switch-indicator sets the LOAD flip flop which, in turn, sets the first timing flipflop (I). The bootstrap load indicator is illuminated on the operating panel, and the main control timing is inhibited since IA00 is held in the clear condition by the LOAD flip flop (Figure 9C15-1D). Nothing else can happen until clock pulses are generated; i. e., the computer is placed in the run condition.

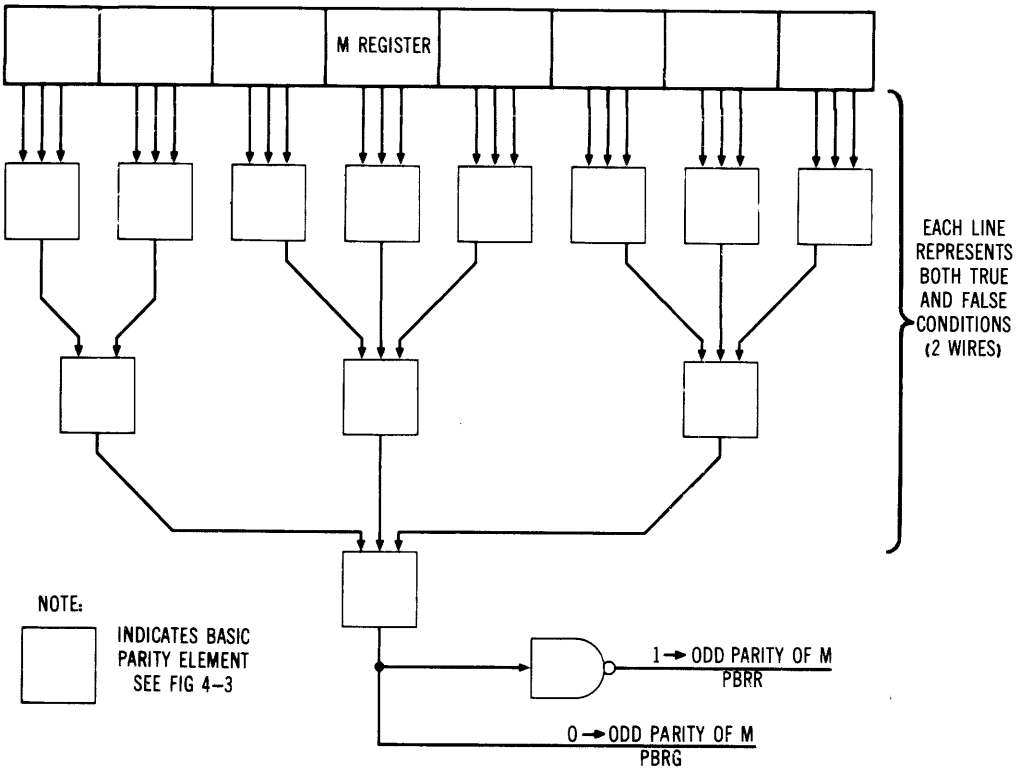


Figure 4-3-E. Parity Tree

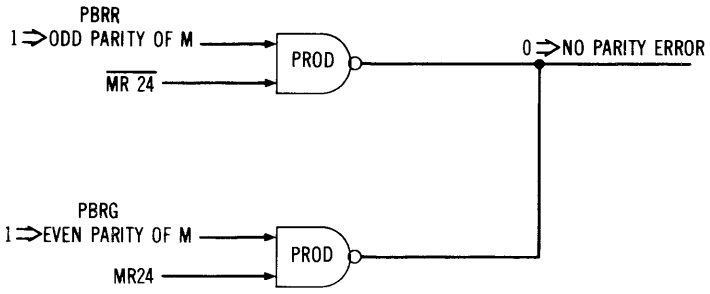


Figure 4-3-F. Parity Check Logic (Used For Read Only)

When the operator depresses the RUN switch CL2A pulses are generated and the following operations in the bootstrap load sequence are initiated: Refer to Figure 4-3-G in addition to the subsequent referenced Figures.

Step 1. The first CL2A pulse (a logical "1" enables gate C054 (Figure 9C15-3C) which sets the value 36000000 into the M-register (Note: The M-and S-registers are initially cleared). At the same time CCWU and CCWL are generated for the purpose of inhibiting the read amplifiers so that 36000000 can be written into memory at address 00000000.

Step 2. The second CL2A pulse sets X-register bit 2^2. The results is an external function code equal to 0004. A read/write cycle is initiated and the contents of M (placed there during step 1) are written into memory address 00000000. An input buffer is established on channel 0 by setting the INPUT ACTIVE and MONITOR

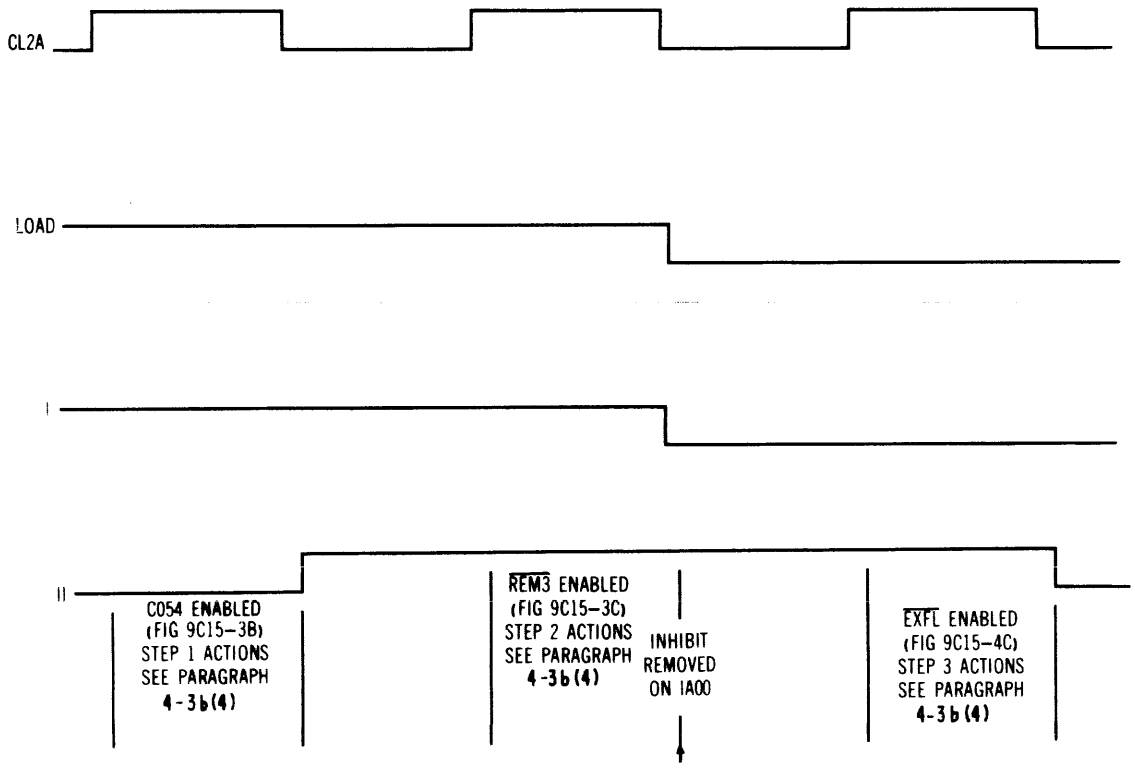


Figure 4-3-G. Automatic Load Control Timing (Bootstrap)

flip flops in the I/O status register. In addition, the inhibit on IA00 is removed.

Step 3. The third CL2A pulse generates an External function control signal which is sent to the paper tape equipment on channel 0 to initiate a READ ASSEMBLY MODE. The code placed in X, during step 2, is now interpreted by the paper tape equipment and the reader begins reading the tape. The value 00000040 is placed in both the M- and S-registers. Next, the value 00000040 is written into memory address 00000040. The I/O timing chain, which produces the necessary timing signals, is initiated by command $\overline{CH00}$ (Figure 9C15-1B). The read inhibits CCWU and CCWL are also generated as part of the normal memory store operation.

(5) Skip Conditional Control Gates. - The SSW (SENSE SWITCH SKIP) instruction which has numerous alternative skip conditions is utilized by the control gates shown in Figure 9C13.

Skips are accomplished by incrementing the P-register a second time during the execution of the SSW instruction. The second incrementing command, INCP, is generated for the SSW (f=07) instruction when CMSW is a logical "0". CMSW is shown in the lower left hand corner of Figure 9C13. INCP is generated by the logic shown in Figure 9C6-7B. When a SSW instruction is in the I-register and its k bits (2^{16} and 2^{17}) are equal to zero, a skip is accomplished if <u>any</u> sense switch is on and the corresponding bit in the I register is a logical "1". (The sense switches as they are shown in Figure 9C13-3D, 4D, 5D, 6D and 7D, are in the on position). If, in any instance, a bit of I corresponds to an on sense switch the output of the respective gate drops to a logical "0" and the CMSW and INCP commands are generated.

For the case where the k bits equal 01_2, the skip condition is satisfied if there is an INPUT DATA REQUEST (a logical "1") on any channel and a logical "1" in the corresponding bit in I. This condition is checked by the series of gates shown in the upper right hand corner of Figure 9C13. IPXX signifies an Input data request (IDR). If the skip conditions are satisfied the respective gate is enabled, its output (CMSW) drops to a logical "0" which generates INCP.

In a like manner, output data request are compared with corresponding bits of I if the k bits of I equal 10_2. The logic which accomplishes this function is shown in the lower right hand corner of Figure 9C13.

If the k bits are equal to 11_2, the skip condition is dependent on the SSW instruction's Y field and the status of various control and data signals. The skip condition is satisfied, for k equal to 11_2, when the following conditions are met.

Y Field equals	
0001_8	and the overflow indicator is on.
0002_8	and the division error indicator is on.
0004_8	and the parity indicator is on.
0010_8	and (A) is negative.
0020_8	and (Q) is negative.
0040_8	and (A) is equal to zero.
0100_8	and the memory overflow indicator is on.
0400_8	and the master lockout flip flop not set.

The logic gates shown in the lower central portion of Figure 9C13 checks for the above listed conditions. When one of these gates is fully enabled its output (CMSW) is a logical "0" and the INCP command is generated.

One addition possibility exists for k equal to 11_2. In this case if the Y field of the SSW instruction is a 1000_2 the program flag flip flop HEY (Figure 9C13-4A) is set. When HEY is set the computer does <u>not</u> do a skip, instead the program flag indicator is illuminated. The program flag flip flop can only be reset by a manual operated reset switch.

(6) Instruction Execution Sequence Timing and Control. - The major control section timing and control logic is shown in Figure 9C7. Instruction execution sequencing is accomplished by six timing flip flops: IA00, ID00, MA00, OA00, IN00 and MS00. Each of these flip flops is set for two microseconds (set and cleared normally by CLE3 SYN). Each instruction requires that IA00 be set for 2 microseconds at the beginning of execution in order to acquire the instruction; however, the sequence of the other timing flip flops depends on the instruction being executed (see Figure 4-3-H). Figure 4-3-I shows which flip flops are set for the various instructions when indirect addressing is <u>not</u> called for.

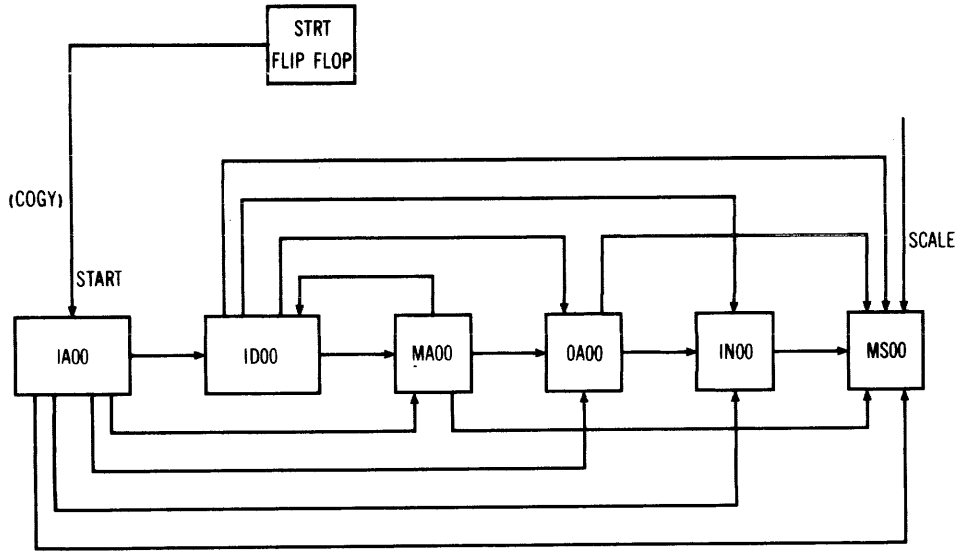


Figure 4-3-H. Instruction Execution Timing Control

The chief functions accomplished when each of the timing flip flops is set are as follows:

IA00 - Instruction Acquisition - Transfer (P) to S and read the referenced instruction into the I and D registers. Increment the P register by 1.

MA00 - Address Modification - Obtain the contents of the B register specified by the B field of the instruction. Add this value to the least significant 14 bits of the I register and transmit the sum to the S register.

ID00 - Indirect Addressing - Reference memory and transfer bit 2^{23} of M to bit position 2^{23} of I. In addition, transfer bits 2^0 - 2^{15} of M to the corresponding I register bit positions. This permits additional indirect operation and modification if the I field of the new instruction still has 2^{23} bit set. Hence a ID-MA-ID-MA can be accomplished unitl a non-indirect type instruction is referenced (see Figure 4-3-H).

OA00 - Operand Acquisition - Obtain the operand from the correct memory location and places it in the D register.

IN00 - Intermediate - Delay the major sequence for certain short arithmetic operations.

MS 00 - Memory Store - Transfer the D register's contents to the M register; generate the necessary read inhibits (CCWU, CCWL) and write the contents of M into core.

(a) Sequence Timing Operation.

1. Initial Start. - Master clear sets flip flop STRT (Figure 9C7-5C) and clears all the timing flip flops. The STRT flip flop, with its set output (H), partially enables COGY (pin 1). The other two COGY inputs are both a logical "1" when the I register is cleared, i. e., no scale, multiply divide or shift instruction and assuming I/O is not requesting memory (IORM = logical "0"). The resultant COGY, a logical "0", is inverted and used as an input to flip flop IA00. When the RUN switch is depressed clock pulses are generated (see paragraph 4-1B). The first SYNC (CLE3) pulse sets IA00 and the computer acquires the first instruction.

2. Sequence Enabling. - The IA00 flip flop in the set state, partially enables the other timing flip flops' input gates. The instruction in the I register determines which flip flop will be set.

Note that the IA00 flip flop remains set until the next SYNC pulse, at which time it is

INSTRUCTION	B DES.	K DES.	COND. MET	IA CYCLE	MA CYCLE	OA CYCLE	IN CYCLE	MS CYCLE
ADD	0	X	X	*		*		
SUB	0	X	X	*		*		
MUL	0	X	X	*		*		
DIV	0	X	X	*		*		
ENA	0	X	X	*		*		
IOR	0	X	X	*		*		
EOR	0	X	X	*		*		
SCL	0	X	X	*		*		
SBT	0	X	X	*		*		
AND	0	X	X	*		*		
CMP	0	X	X	*		*		
ADD	1	X	X	*	*	*		
SUB	1	X	X	*	*	*		
MUL	1	X	X	*	*	*		
DIV	1	X	X	*	*	*		
ENA	1	X	X	*	*	*		
ENQ	1	X	X	*	*	*		
IOR	1	X	X	*	*	*		
EOR	1	X	X	*	*	*		
SCL	1	X	X	*	*	*		
SBT	1	X	X	*	*	*		
AND	1	X	X	*	*	*		
CMP	1	X	X	*	*	*		
STA	0	X	X	*				*
STA	1	X	X	*	*			*
STQ	0	X	X	*				*
STQ	1	X	X	*	*			*
STB	0	X	X	*				*
STB	1	X	X	*		*		*
ENB	X	=3	X	*				*
ENB	X	≠3	X	*		*		*
RAD	0	X	X	*		*		*
RSB	0	X	X	*		*		*
RAO	0	X	X	*		*		*
RSO	0	X	X	*		*		*
RAD	1	X	X	*	*	*		*
RSB	1	X	X	*	*	*		*
RAO	1	X	X	*	*	*		*
RSO	1	X	X	*	*	*		*
JSR	X	X	0	*				
JSR	0	X	1	*				*
JSR	1	=3	X	*				*
JSR	1	≠3	1	*	*			*
JMP	0	X	X	*				
JMP	1	X	0	*				
JMP	1	=3	X	*				
JMP	1	≠3	1	*	*			

Figure 4-3-I. Sequence of Instruction Cycles
(Sheet 1 of 2)

INSTRUCTION	B DES.	K DES.	COND. MET	IA CYCLE	MA CYCLE	OA CYCLE	IN CYCLE	MS CYCLE
JPA	0	X	X	*				
JPA	1	X	0	*				
JPA	1	X	1	*	*			
JDB	X	X	X	*	*			*
SSW	X	X	X	*				
SSH	0	X	X	*		*	*	*
SSH	1	X	X	*	*	*	*	*
SCP	0	X	X	*		*	*	
SCP	1	X	X	*	*	*	*	
SSK	0	X	X	*		*		
SSK	1	X	X	*	*	*		
SHF ($\overline{SCA+SCAQ}$)	X	X	X	*			*	
SHF(K=0)(B=3)				*				
SHF(SCA+SCAQ)	X	X	X	*			*	*
CYS	0	X	X	*		*		
CYS	1	X	X	*	*	*		
NXN	X	≠3	X	*	*			
NXN	X	=3	X	*				
CIS	0	≠3	X	*		*		
CIS	0	=3	X	*				*
CIS	1	≠3	X	*	*	*		
CIS	1	=3	X	*	*			*
EXO	X	X	X	*		*		
EXF	X	X	X	*		*		
EXI	X	X	X	*				*
NTA + NAS + FAULT→FLT	X	X	X	*				
	IR10	IR11						
CBM	0	0	X	*		*		
CBM	0	1	X	*		*		*
CBM	1	X	X	*				*

Figure 4-3-I. Sequence of Instruction Cycles
(Sheet 2 of 2)

cleared and another timing flip flop is selected. For those instructions which do not require the setting of another timing flip flop, IA00 is re-selected. Gate $\overline{IA00}$ (Figure 9C7-7D) when enabled and clocked by CL2A reselects IA00 by grounding reset output of IA00.

The conditions for enabling each timing flip flop are evaluated as indicated in Figure 4-3-J. Sequence enabling logic is shown in Figure 9C8. These enabling gates select the timing flip flop to be set during the next two microsecond period. For example, when SMAI (Figure 9C8-7B) is a logical "1" flip flop MA00 is selected. SMAI is a logical "1" when any of its inputs is a logical "0". The various inputs are a logical "0" under the following conditions:

Pin 1. Gate C070 (Figure 9C8-8B) outputs a logical "0" if all of its sixteen inputs are a logical "1". This condition exists only when the B designator does not equal zero and the I translator does not contain one of the following instructions:

CBM
FLT
EXI
EXF
EXO
NXN
SHF
SSW
JPA
JPM
JSR
ENB
STB
NTA
NAS

Note that all of the above instructions have a normal B designator interpretation which permits indexing.

Also, OR'ed with this pin 1 of input SMAI are three additional C070 gates (Figure 9C8-6D). Two of these are for jump (35 and 36) instructions with the jump condition satisfied (DMJM or DMJA = logical "1") and B≠0. The third C070 gate output is a logical "0" when the NXN instruction is in the I translator and k≠3.

Pin 4. Gate C068 (Figure 9C8-7D) checks for the normal B interpretation of the arithmetic conditions on the JSR instruction. Therefore, pin 4 is a logical "0" when B≠0, f=37, K≠3, and the arithmetic jump condition is satisfied.

Pin 3. Input pin 3 of SMAI is logical "0" when the JDB instruction is in the I translator.

All of the above logical "0"'s are OR'ed by gate SMAI. When any of the SMAI inputs is equal to a logical "0", the output of SMAI is a logical "1"; thus, an index modification is called for and the MA00 flip flop is set after the time period IA00 (if indirect addressing is not indicated).

The other sequence enables shown on Figure 9C8 are determined in a similar manner.

3. Enabling the ID00 Cycle. - The ID00 flip flop may be set from two sources, IA00 and MA00. When an instruction, with a normal b interpretation, is placed in the I register it is examined first to determine if the b designator is non zero. If the b designator is non-zero, a MA cycle is entered from IA00. If no index modification is called for (b designator equal to zero)or after the required modification is performed, the instruction word is examined to determine if indirect addressing is necessary.

FROM	TO	CONDITION
IA00	ID00	INDIRECT I register bit 23 and 21 or 22 set and B designator=0
IA00	MA00	No INDIRECT and SMAI (see Figure 9C8-7B)
IA00	OA00	No INDIRECT and SINI (see Figure 9C8-6B)
IA00	IN00	No INDIRECT and SINI (see Figure 9C8-5B)
IA00	MS00	No INDIRECT and SMSI (see Figure 9C8-3B)
ID00	MA00	INDIRECT and B designator≠0
ID00	OA00	INDIRECT and SOAI (see Figure 9C8-6B)
ID00	IN00	INDIRECT and SINI (see Figure 9C8-5B)
ID00	MS00	INDIRECT and SMSI (see Figure 9C8-3B)
MA00	ID00	INDIRECT
MA00	OA00	No INDIRECT and SOAM (see Figure 9C8-5B)
MA00	MS00	No INDIRECT and SMS2 (see Figure 9C8-3B)
OA00	IN00	SINO (see Figure 9C8-3B)
OA00	MS00	SMS3 (see Figure 9C8-4B)
IN00	MS00	SMS4 (see Figure 9C8-4B)
any	IA00	$\overline{SART}$ (see Figure 9C8-1D)

Figure 4-3-J. Sequence Enable Condition

The logic which performs these decisions is located in the upper left hand corner of Figure 9C7. Four gates are involved: CODE, CODU, INOR and INDR. These gates translate bits of the I register and determine if bit 23 is set and also if the instruction is one of the group which permits indirect addressing. Either IR21 or 22 must be set along with IR23 in order for the indirect operation to be called for. This excludes instructions 00 through 07 and 40 through 47.

4. Sensing the End of an Instruction. - The end of a current instruction is sensed by a group of $\overline{\text{SART}}$ gates shown in the upper right hand corner of Figure 9C7. The output $\overline{\text{SART}}$ is a logical "0" when ever a timing flip flop is set and no enables are present from any succeeding timing flip flop. For example, the left most $\overline{\text{SART}}$ gate (Figure 9C7-2D) is fully enabled during IA00 time when neither CODA (ID00 enable), COBY (MA00 enable), COBU (OA00 enable), COBE (IN00 enable) nor COBA (MS00 enable) is selected (logical "0"). The adjacent $\overline{\text{SART}}$ gate is fully enabled during the ID00 time when neither COBY, COBU, COBE nor COBA is selected.

When $\overline{\text{SART}}$ is generated, by any of the $\overline{\text{SART}}$ gates, the resulting logical "0" is inverted and utilized to set the STRT flip flop (when SYNC appears) and to partially enable (pin 2) COGY (Figure 9C7-7C). The other enable to COGY (pin 3) is also a logical "1" unless IORM (I/O requesting memory) or a long arithmetic operation is in progress (as indicated by CODY, BAHE, and BBOO). Signal COGY, when at logical "0", indicates that another IA00 time can start and enables the IA00 flip flop which is then set at SYNC. COGY is also sent to the clock control logic (Figure 9C12) to clear the RUNN flip flop and stop the computer at the end of an instruction if either a program or manual machine STOP is called for.

5. Major Commands Issued During IA00. - When the IA00 flip flop (Figure 9C7-7B) is set, signal $\overline{\text{IA00}}$ becomes a logical "0" and signals IA01 and IA00 become logical "1"'s. These signals are used to provide the various enables and commands which acquire an instruction from memory and places it in the I register.

Normal IA00 Timing. - The following signals are generated during the normal IA00 time:

TRPS	Transfer P to S
RESM	Reset M
RESI	Reset I
READ	Read to Memory Timing
TRMI	Transfer M to I
LMDU	Transfer M_L to D_L
LMDL	Transfer M_U to D_U
CLDI	Clock D
INCP	Increment P
ADIO	$I_Y + 0$

The above commands, along with some of lesser importance, primarily serve to obtain the next instruction and place it in the I register. The instruction is also transferred to the D register where it is used primarily for arithmetic instruction interpretation.

Refer to 9E for the exact time relationship of these signals.

Interrupt IA00 Timing. - When any control or I/O interrupt occurs, the next instruction of the program to be executed is taken from a special fixed location unique to that particular interrupt. The fixed interrupt location is placed directly in the S register. Obviously, the normal IA00 operation would destroy this input, therefore the normal P to S transfer is inhibited. Also, the content of the P register should not be incremented during the interrupt IA00 sequence. In order to prevent both of these operations during IA00 time, flip flop BLKP (Figure 9C7-4D) is set. BLKP inhibits TRPS generation by placing a logical "0" on pin 3 of gate KUHN (Figure 9C1-8C). BLKP, also, inhibits INCP generation by a logical "0" on pin 1 of gate 6C (Figure 9C6).

Any interrupt causes the BLKP flip flop (Figure 9C7-4D) to be set following the current instruction's execution cycle. Gate COJU (Figure 9C7-4D) is fully enabled at SYNC time when COGY and INTR occurs. COGY signifies the end of an instruction, and $\overline{\text{INTR}}$ signifies an interrupt condition exists.

This logic permits interrupts to be honored only after an instruction is executed. BLKP is then cleared by clock CLE2 during IA00 time.

The interrupt signal INTR (Figure 9C8-3B) which sets flip flop BLKP and initiates an IA00 interrupt is generated by several means. Any I/O type interrupt (Power failure, Manual, etc.) sets flip flop PRIN in the I/O Section (Figure 9I1-6A). This (PRIN) signal is inverted and will generate INTR unless an I/O type instruction is being executed or a jump is in progress. Other conditions which generate INTR are derived directly from the control section: $\overline{\text{FLTI}}$ when a function code of 00_8 is translated from the

I register; and parity errors $\overline{PRPI}$ (program parity) and $\overline{IORI}$ (I/O parity error).

6. Major Commands Issued During MA00. - Refer to Figure 9C7-5B. The MA00 flip flop when set indicates an address indexing is called for and will issue the following major commands:

$\overline{READ}$	Start memory cycle
$\overline{RESM}$	Reset M
$\overline{LMDU}$, $\overline{LMDL}$	Transfer M to D
CLDI	Clock D
TRBS	Fixed location for b to S
ADID	Add $I_Y + (B_b)$

The above operation reads from memory the designated B location's content and places it in the D register. The necessary commands are then issued to add the lower 14 bits (address portion) of the I register to the contents of the B register location (in D). A timing relationship of the above commands is shown in Figure 9E.

7. Major Commands Issued During ID00. - Refer to Figures 9C7-6B and 9E10. The ID00 flip flop when set, indicates that an indirectable instruction is in the I register and that bit 2^{23} is set. Under these conditions the following major commands are issued.

READ	Start memory cycle
RESI	Reset I_Y, I_b and 2^{23} only
LMDU, LMDL	Transfer M to D
CLDI	Clock D
ADIO	Enable I_Y + 0 to adder
LOSS	Load adder output to S
TRMI	Y and B fields and 2^{23} only

The above series of operations transfer the location specified by the Y field of the instruction to the S register. A memory READ brings the contents of this specified location to the M register. The Y & B fields and the indirect address bit, 2^{23}, are now transferred from the M register to the I register. Bit 2^{23} of I is again checked to determine if another indirect operation is required. If it is the next sequence is another ID00 providing the new b designator is equal to zero. If the new b designator is not equal to zero an MA00 sequence is done and then the ID00 sequence is done. The indirect addressing cycle may continue indefinitely (N level indirecting) as long as the content of the indirect location has bit 2^{23} set.

An instruction indirecting to itself (i. e. $\frac{\text{address}}{01000}$ $\frac{\text{instruction}}{76001000}$) continues indefinitely. The end of the instruction can never be attained! Thus, the computer connot be stopped and no I/O operations can be performed during this condition.

8. Major Commands Issued During OA00. - Refer to Figure 9C7-4B. The OA00 flip flop is set during the operand acquisition process, and the following major commands are issued:

READ	Start memory cycle
$\overline{RESM}$	Reset M
LOSS	Transfer adder output to S
LDMU (if K=0)	Transfer M_u to D_u
LDML(if K=0 or K=1)	Transfer M_L to D_L
LDMX(if K=2)	Transfer $M_u \rightarrow D_L$
EXDO(if K=3)	Place zeros in upper 10 bits of D
CLDI	Clock D
EXDS(if $\overline{LMDU}$)	Extend sign bit to upper 12 bits of D

The output of the adder at the beginning of the OA00 cycle is the operand address. This location is gated to S and then a memory READ is performed. After the READ, the M register contains the operand. Depending on the K-bits, in the I-register, the operand may be transferred to the D-register in full or in part, or it may not be transferred at all. If K equals 0, the full word (operand) is transferred; if K equals 1 the lower half of the word is transferred; and if K equals 2 the upper half of the word is transferred. In addition, when K equals 1 of 2 the sign bit of the operand is extended (EXDS).

If the K-bits, in the I-register, equals 3, a word transfer to the D-register is not made since the operand is already in the D-register. Since the operand only utilizes the lower 14 bits of the D-register, a K of 3 causes the upper 10 bits of D to be cleared. The operand is placed in D in accordance with the b designator of the instruction in the I-register. For a b equal to zero the operand is part of the instruction in the I-register and was transferred to D at the same time it was transferred to I. For a b not equal to zero the operand is the contents of the designator B box and was transferred to D during the previous MA00 cycle.

9. Major Commands Issued During MS00. - Refer to Figure 9C7-1A. The MS00 flip flop is set during a memory store. The major commands

issued during the MS00 sequence include:

READ	Start memory cycle
RESM	Reset the M register
LOSS	Adder output to S
LDMU (k=0)	Transfer D_u to M_u
LDML (k=0, 1 or 3)	Transfer D_L to M_L
LDMX (k=2)	Transfer D_u to M_L
LDMA (k=3)	Transfer D_A to M_A (address bits)
CCWL (k=0 or 1)	Inhibit read lower 12 bits
CCWU (k=0 or 2)	Inhibit read upper 12 bits
CCWA (k=3)	Inhibit read address bits

The above commands store in memory the contents of the D-register at a location specified by S. The upper, lower, or address portions of D may be stored in accordance with the k-designator of the stored instruction. The input gates to those M-register bit positions which contain data to be stored are inhibited during the read portion of the memory cycle. The information in M is then stored during the write portion of the memory cycle. The exact timing relationship of these commands is shown in Figure 9E9.

(b) Summary of Commands Issued by The Control Section. - The majority of command enables issued from the control section are generated by the logic shown in Figure 9C1 through 9C6, "Instruction Execution Control A through F" respectively. All major command enables have been assigned names for identification purposes.

4-4 ARITHMETIC SECTION OPERATION.

a. General Information. - The function of the arithmetic section is to perform the numerical manipulations called for by the list of instructions. The general philosophy of the DPS-2402 arithmetic section is that it is essentially an independent unit with its own registers, controls, and timing circuits. Dependence on the control section lies solely in a start command for a particular operation (i. e. add, subtract, multiply, divide, etc.) and clock pulses. The major internal arithmetic control commands are listed in Figure 4-4-A. Prior to the start command, the control section has insured that the data (operand), if required by the instruction, has been placed in the D-register and all is ready for the arithmetic operation. After the arithmetic operation has been initiated, it runs virtually (except for 1.5 mc CLEA clock pulses) independent of commands from the control section. In some cases, the control section becomes dependent upon the arithmetic section, as control will cease operation until the arithmetic section completes a lengthy function such as multiply, divide, or scale.

b. Arithmetic Section Components. - Figure 4-4-B shows a block diagram of the Arithmetic Section. The Arithmetic Section consists of three data registers, AA, AD, and AQ; two control registers AC and AF, one counter-register AN. A single 24 bit parallel additive adder is capable of adding the contents of the A-register (directly and shifted right one place) and D-register or the contents of the D-register and the least significant 14 bits of the I-register. Inputs to the adder may be the complement of the D-register contents; therefore, when (D), contents of the D-register, is added to (A) a subtraction function is performed. The adder output (BS_i) is capable of being gated to the A-register directly or the A-register shifted, the Q-register, as well as the P- or the S-registers. The Arithmetic Section also consists of numerous translators and control logic as might be expected. The Arithmetic Section logic diagrams are shown in figures 9A0 through 9A53.

During the instruction acquisition (IA00) cycle of the control section, the instruction word is always transferred to the D-register as well as the I-register. This permits immediate access to the instruction by the arithmetic section; therefore, upon receipt of the start command, from the Control Section, the arithmetic unit decodes the arithmetic instruction, enables the various arithmetic control flip flops and transfers the shift count to the N register for those instructions where a shift is required. The shift count is located in the least significant 6 bits of the introduction word.

During the address modification (MA) cycle the content of the B-register (location), specified by the b designator, is entered in the D-register. The adder adds this quantity to the lower 14 bits (Y field) of the instruction and the sum (modified address) is gated to the S-register (or P-register in the case of an indexed jumped instruction with the jump condition satisfied).

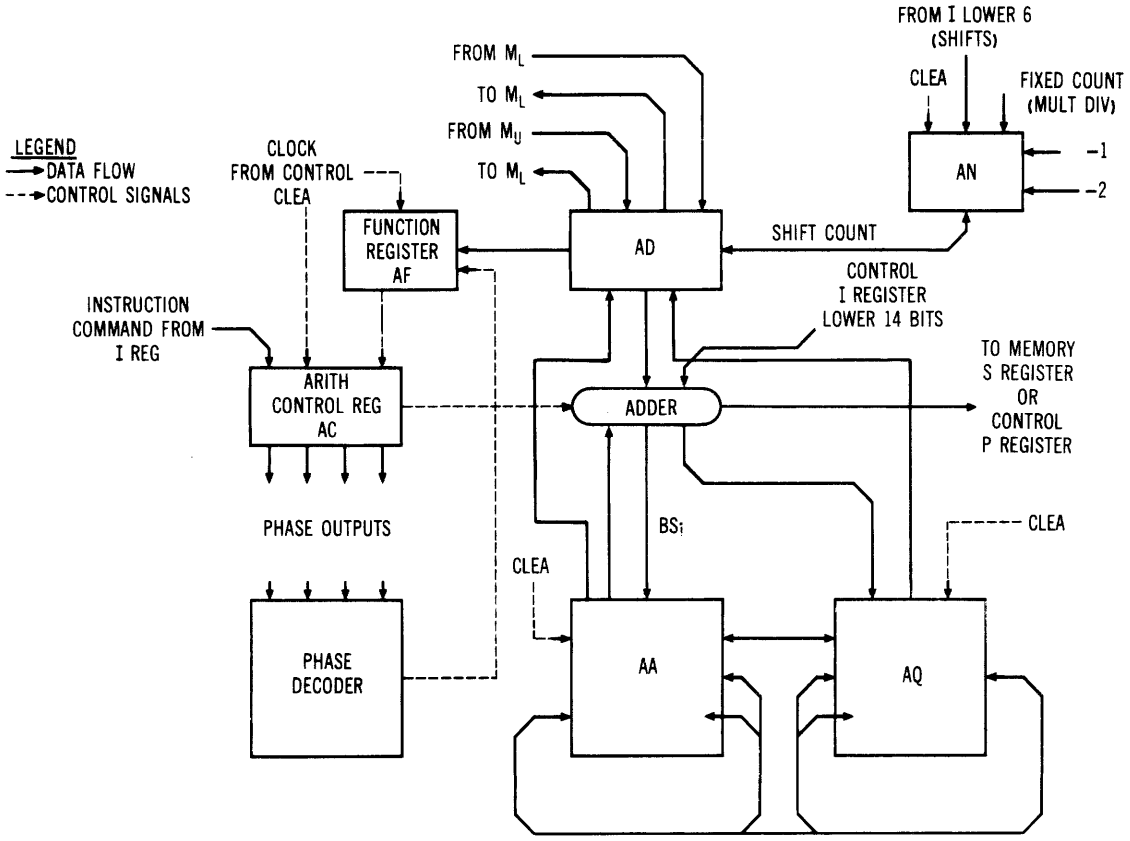


Figure 4-4-B. Arithmetic Section Block Diagram

(1) A-Register and Adder. - The A-register is the principal arithmetic register. After execution of an addition or a subtraction instruction, the sum or difference is found in the A-register. As previously indicated, the adder always adds. When a subtraction is called for, the one's complement of (D) is gated to the adder; in addition, a "forced" carry is gated into the first stage of the adder. This results in the two's complement of (D) being added to (A). After multiplication, the most significant half of the product is held by the A-register. The remainder is found in A after a divide instruction.

(a) A-Register Logic. - The A-register is shown on Figures 9A40 through 9A47. J-K flip flops are used and are clocked by clocks AX00 and AX23 (Figure 9A10-2C) for the set side and AY00 and AY23 (Figure 9A9-2B) for the reset side.

One source of inputs to the A-register is from the Adder. Variable transfer is available and it provides for a great deal of versatitity. Depending upon the control signal, adder bits (BS_i) may be transferred directly to corresponding A-register (AA_i) flip flops or may be shifted left or right, singly or doubly by appropriate commands (see Figure 4-4-C).

AL01	Single Left Shift ($BS_i \longrightarrow AA_{i-1}$)
AM02	Double Left Side ($BS_i \longrightarrow AA_{i-2}$)
AS00	Double Right Shift ($BS_i \longrightarrow AA_{i+2}$)
AR00, BERD	Single Right Shift ($BS_i \longrightarrow AA_{i+1}$)
BERB, BERA	

Inputs to the A-register from the D-register may be directed by the transfer command AU02 via gate A of Figure 4-4-C, providing the corresponding bit of Q is set. This operation is recognized as that used by the Substitute (SBT) Instruction. If Q is a logical "0", gate B prevents setting A by holding input F of the A-register flip flop at logical "0".

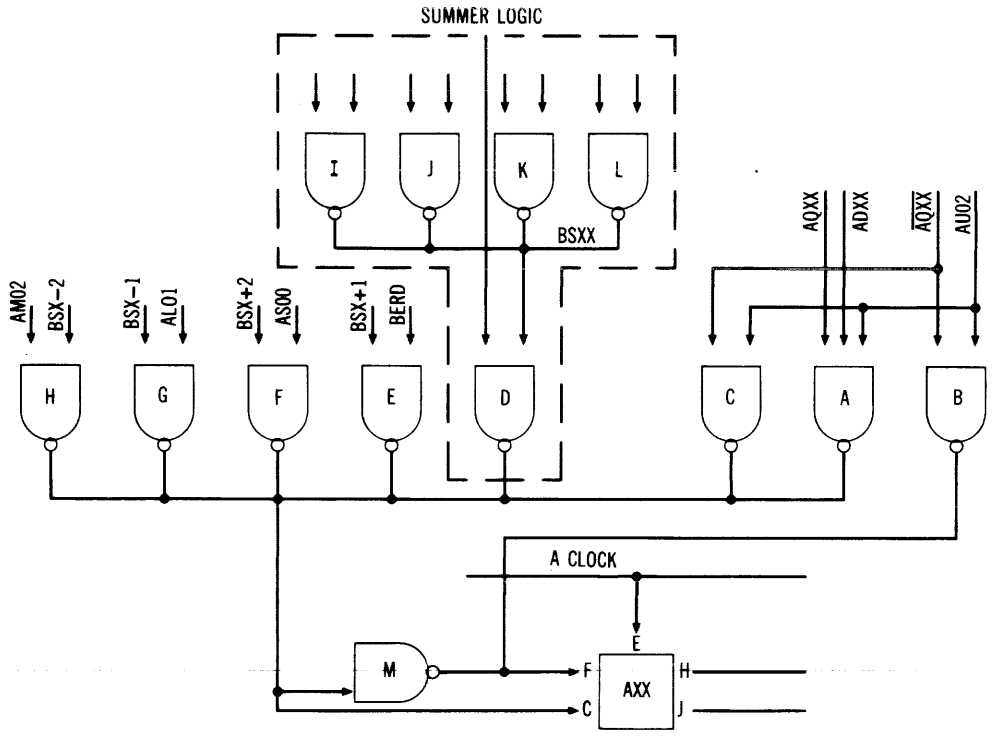


Figure 4-4-C. A-Register Transfer Logic (Input)

Gates D, I, J, K, and L of Figure 4-4-C are logic associated with the adder and are discussed in detail in paragraph 4-4b(1)(b).

Gate E (Figure 4-4-C) is the adder output of the next highest stage and the command BERD (shift right one place).

Gate F (Figure 4-4-C) is the adder output from two stages to the left of the stage in question and is enabled by the command AS00.

Gate G (Figure 4-4-C) utilizes the adder output from one stage to the right and is enabled by AL01 (shift left one place).

Gate H (Figure 4-4-C) is used to shift left two places and therefore utilizes AM02 and the adder output from two stages to the right of the stage in question.

(b) Adder. - The Arithmetic Adder, a 24-bit parallel additive device may be logically divided into two functional portions: the "summing" portion and the "carry" portion. The various gates for these two functions are labeled BS_i and BC_i respectively, where i indicates any bit position 00 through 23.

1 Sum. - The logical sum (BS_i) is formed (see Figure 4-4-D) for each bit by logic shown on the A-register. Five gates are required for this function. Four to form the boolean expression BS_i shown in Figure 4-4-D, and a single input NAND to invert this quantity to BS_i and to gate the sum to the A-register upon command BL00.

The truth table for BS_i is shown in Figure 4-4-E. To check the adder logic on Figure 4-4-D assume that the bit of D is a logical "1", the bit of A is logical "0", and that no carry C exists from the previous stage. For this instance, the quantity BS_i should be a logical "1" (from the truth table) and the AA_i flip flop should be set when BL00 command is issued.

TRDA	Transfer (D) → (A), '1' ⇒ ENA
ADDX	1 ⇒ ADD, RAD
SUBO	1 ⇒ JDB, SUB 1 (RSO)
SUBT	1 ⇒ SUB, RSB, CMP, CYS, SCP
ADDO	1 ⇒ RAO
ANDQ	1 ⇒ SCP
CSAL	1 ⇒ SSH
SDAX	1 ⇒ NXN
EXOR	1 ⇒ EOR
INHX	1 ⇒ SCL
INOR	1 ⇒ IOR
SSUB	1 ⇒ SBT
TRDQ	Transfer (D) → (Q), 1 ⇒ ENQ, NXN (K = 1 = XAQ)
DIVX	1 ⇒ DIV
ANDA	1 ⇒ AND
ADID	1 ⇒ JDB, MA01 ie: INDEXING
AD10	1 ⇒ Transferring Io - 13 → P or S?
NEGA	1 ⇒ NXN (K 0 ie: NEG)
CLDI	1 ⇒
EXDO	1 ⇒ K = 3 (ADD + SUB + DIV + MUL + ENA + ENQ + SBT + EOR + SCL + IOR + AND + CMP + ENB)
TRQD	1 ⇒ NXN (K = 1 ie: XAQ) , STQ
TRAD	1 ⇒ RAD , RSB , STA
EXSH	1 ⇒ SHF K = 0
RVBA	1 ⇒ RBAQ
MULT	1 ⇒ MUL
SDAX	1 ⇒ (NXN. K = 1 ie: XAQ) , (JDB + RAO + RSB + RAD + RSO) , SSH
SCAQ	1 ⇒ SHF. B = 1

Figure 4-4-A. Major Arithmetic Control Signals (Sheet 1 of 2)

'D' REGISTER

Enabling Logic	bit 0 BE00 - bit 23 BE23
AN25	Enable 'N' reg → D bits 0-5
BM50	Enable Q → D
BM00	Enable A → D
AP50	Enable Q & D
AX25	Clock AD00 - AD13
AX39	Clock AD14 - AD23

'Q' REGISTER

Enabling Logic	AQ00 BE75 - AQ23 BE98
AM50	Enable Double Left Shift
AL50	Enable Single Left Shift
AS50	Enable Double Right Shift
AR50	Enable Single Right Shift
BL50	Enable BS → Q
AX50	Clock AQ00 - AQ22
AX73	Clock AQ23

'A' REGISTER

Enabling Logic	AA00 BE50 - AA23 BE73
AM02	Double Left Shift
AL01	Single Left Shift
AS00	Double Right Shift
AR00, BERD, BERB, BERA	Single Right Shift
BN24	Force Carry on AA00
BL00	Enable BS → A
AX00	Clock AA00 - AA22 Set Clock
AY00	AA00 - AA22 Reset Clock
AX23 AY23	Clock AA23
BS00 thru BS23	Summer Outputs

ADDER

BT23	Enable D
BT50	Enable $\overline{D}$
BU00	Enable AA00 - AA23
BU50	Enable AA01 - AA23
BF00	Simulated All 1's in A register
BJ00	Enables IR00 - IR13
BN00	Carry Enable
BN24	Forced Enable
BC00 thru BC23	Carry

'F' REGISTER

CLEA	(AX99) Clock
AF00	Division Error
AF01	Arithmetic Overflow
AF02 thru AF09	Various
AQ500	Phantom Bit (External Q Bit) (For divide and multiply)

'N' REGISTER

CLEA	Clock
BG00	Enable Decrement

Figure 4-4-A. Major Arithmetic Control Signals (Sheet 2 of 2)

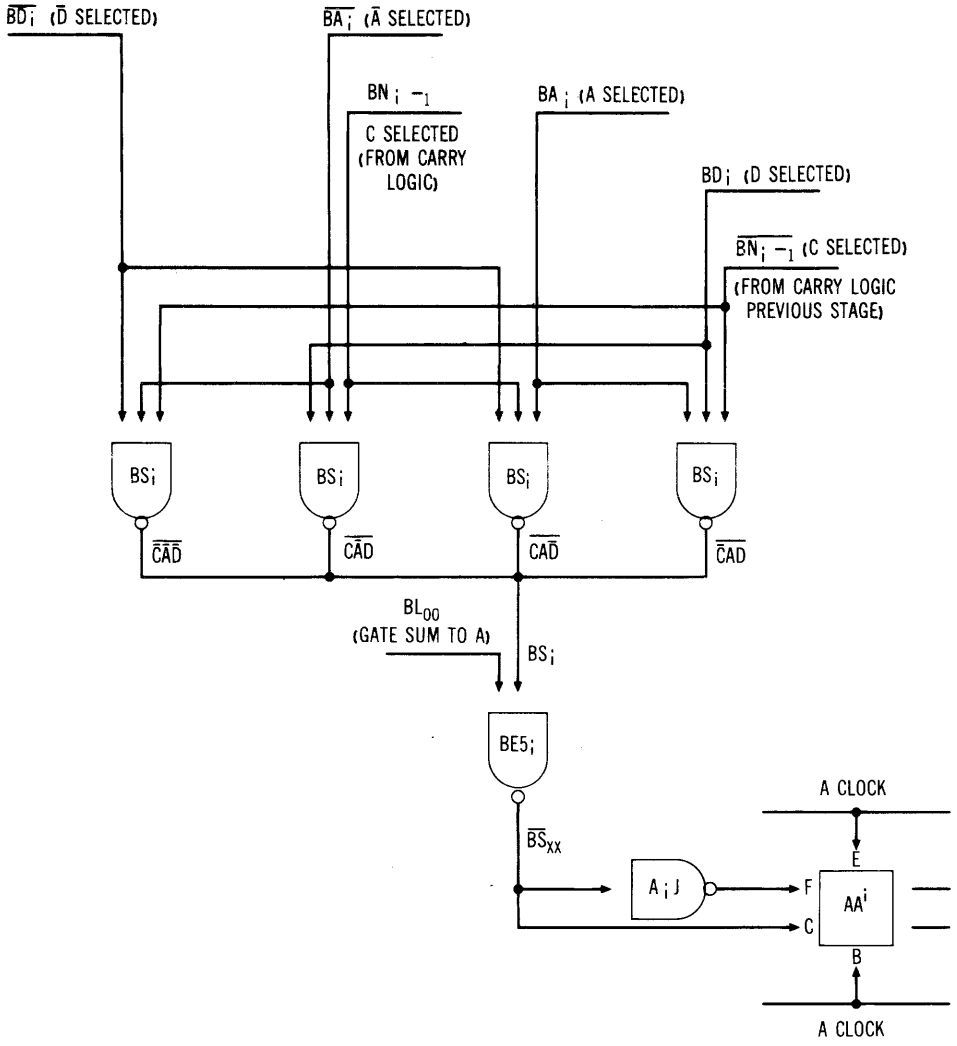


Figure 4-4-D. ADDER Logic

Carry From Previous Stage C	A	D	Sum BS_i	Carry To Next Stage C_o
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

(a) TRUTH TABLE – FULL ADD

$$\overline{BS_i} = \overline{C}\,\overline{A}\,\overline{D} + C\,\overline{A}\,D + C\,A\,\overline{D} + \overline{C}\,A\,D$$

$$C_o = C\,(A + D) + AD$$

(b) LOGICAL EQUATIONS – FULL ADD

Figure 4-4-E. Full Adder Truth Table and Logic Equations

Referring to Figure 4-4-D, for this condition none of the four BS_i gates are enabled and BS_i is a logical "1". When command BL_i becomes logical "1", gate $BE5_i$ is enabled, resetting AA_i with the resulting logical "0". The logical "1" from gate A_iJ combines with the A clock signal (pulsed) to set AA_i flip flop.

Consider a case where the summer output is a logical "0". Gate $BE5_i$ is not enabled even when BL00 is present. The A clock signal will clear the AA_i flip flop.

Each combination of the truth table may be confirmed in like manner.

2 Carry. - The requirement that a complete addition must be accomplished in less than one microsecond with 40 nanosecond switches places the requirement that the carry logic be accomplished by parallel-ripple means. From a circuit standpoint, a ripple (serial) carry is the simplest, while the complete parallel is logically the most complicated. The parallel-serial carry used in the DPS-2402 is a compromise between logic complexity and speed. Figure 4-4-F shows a block diagram of the complete adder. Carrys are generated in parallel within a section (intrasection carries) and serially between the four sections (intersection carries).

In order to obtain the desired parallel carry generation, the carry for an individual stage, namely $C_i = C_{i-1}(A_i + D_i) + A_i D_i$, is expanded by substitution for C_{i-1} as shown in Figure 4-4-G. As can be seen from these logic equations, implementation of the carry out of the section becomes more difficult as the number of adder stages within the section increases. Figures 9A32 through 9A39 show the complete carry logic gates which implement the logic equations of Figure 4-4-G. Two sheets of logic are devoted to each section. For example, Figure 9A32 contains the carry logic for the lower three carry stages, while Figure 9A33 shows the two higher order stages of Section I (see Figure 4-4-F). The carry is formed spontaneously within a section; i.e., to all summer intrasection gates and the intersection carry to the next section. In this way high adder speed is obtained. Figure 4-4-G illustrates the logic complexity introduced by increasing the number of stages.

3 Composite Adder. - Figure 4-4-H shows the complete addition process for stage 08. The summer output formed by the BS08 series gates from bits of A, D and carry are available for gating into the A-register or the S-register.

The carry (BC08) is formed from the lower order A and D bits and the incoming section carry. This carry is used by the summer for bit 09 to form the sum for that bit. Note that information from A_{08} and D_{08} also contribute to formation of the carry for higher order stages of the section and ultimately to the intersection carry to the next higher section.

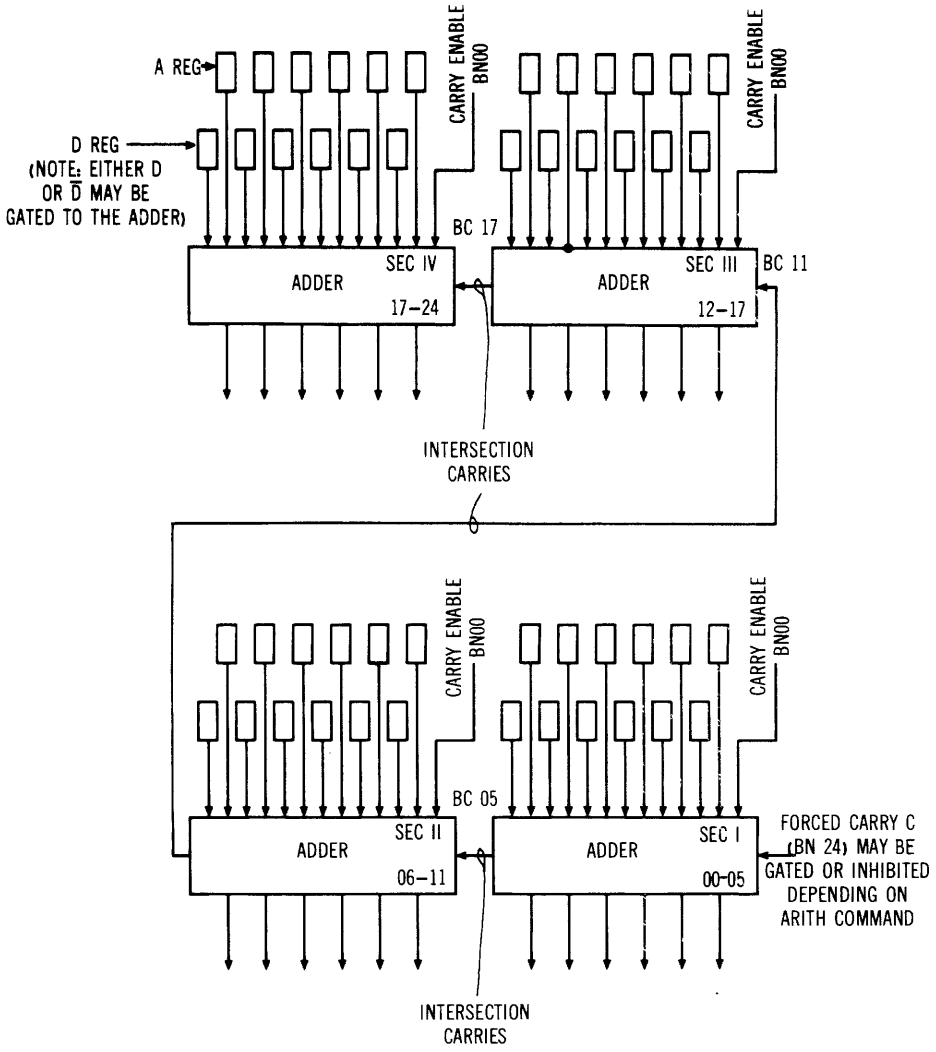


Figure 4-4-F. Arithmetic Adder Block Diagram

C_I = Carry into the section (intersection carry)

Intrasection Carries To Adder Bits Within A Section

$$C_0 = A_0 D_0 + (A_0 + D_0)\ C$$

$$C_1 = A_1 D_1 + (A_1 + D_1)\ C_0 = A_1 D_1 + (A_1 + D_1)\ A_0 D_0 + (A_0 + D_0)\ (A_1 + D_1)\ C$$

$$C_2 = A_2 D_2 + (A_2 + D_2)\ (A_1 D_1) + (A_2 + D_2)\ (A_1 + D_1)\ (A_0 D_0) + (A_2 + D_2)\ (A_1 + D_1)\ (A_0 + D_0)\ C$$

$$C_3 = A_3 D_3 + (A_3 + D_3)\ A_2 D_2 + (A_3 + D_3)\ (A_2 + D_2)\ (A_1 D_1) + (A_3 + D_3)\ (A_2 + D_2)\ (A_1 + D_1)\ A_0 D_0 + (A_3 + D_3)\ (A_2 + D_2)\ (A_1 + D_1)\ (A_0 + D_0)\ C$$

$$C_4 = A_4 D_4 + (A_4 + D_4)\ A_3 D_3 + (A_4 + D_4)\ (A_3 + D_3)\ (A_2 D_2) + (A_4 + D_4)\ (A_3 + D_3)\ (A_2 + D_2)\ A_1 D_1 + (A_4 + D_4)\ (A_3 + D_3)\ (A_2 + D_2)\ (A_1 + D_1)\ (A_0 + D_0)\ A_0 D_0 + (A_4 + D_4)\ (A_3 + D_3)\ (A_2 + D_2)\ (A_1 + D_1)\ (A_0 + D_0)\ C$$

$$C_5\ A_5 D_5 + (A_5 + D_5)\ A_4 D_4 + (A_5 + D_5)\ (A_4 + D_4)\ (A_3 + D_3)\ A_2 D_2 + (A_5 + D_5)\ (A_4 + D_4)\ (A_3 + D_3)\ (A_2 + D_2)\ (A_1 D_1) + (A_5 + D_5)\ (A_4 + D_4)\ (A_3 + D_3)\ (A_2 + D_2)\ (A_1 + D_1)\ (A_0 D_0) + (A_5 + D_5)\ (A_4 + D_4)\ (A_3 + D_3)\ (A_2 + D_2)\ (A_1 + D_1)\ (A_0 + D_0)\ C$$

C_5 Carry out of section (intersection carry)

Figure 4-4-G. Boolean Equation For Parallel Carry Logic (Identical For Each Section)

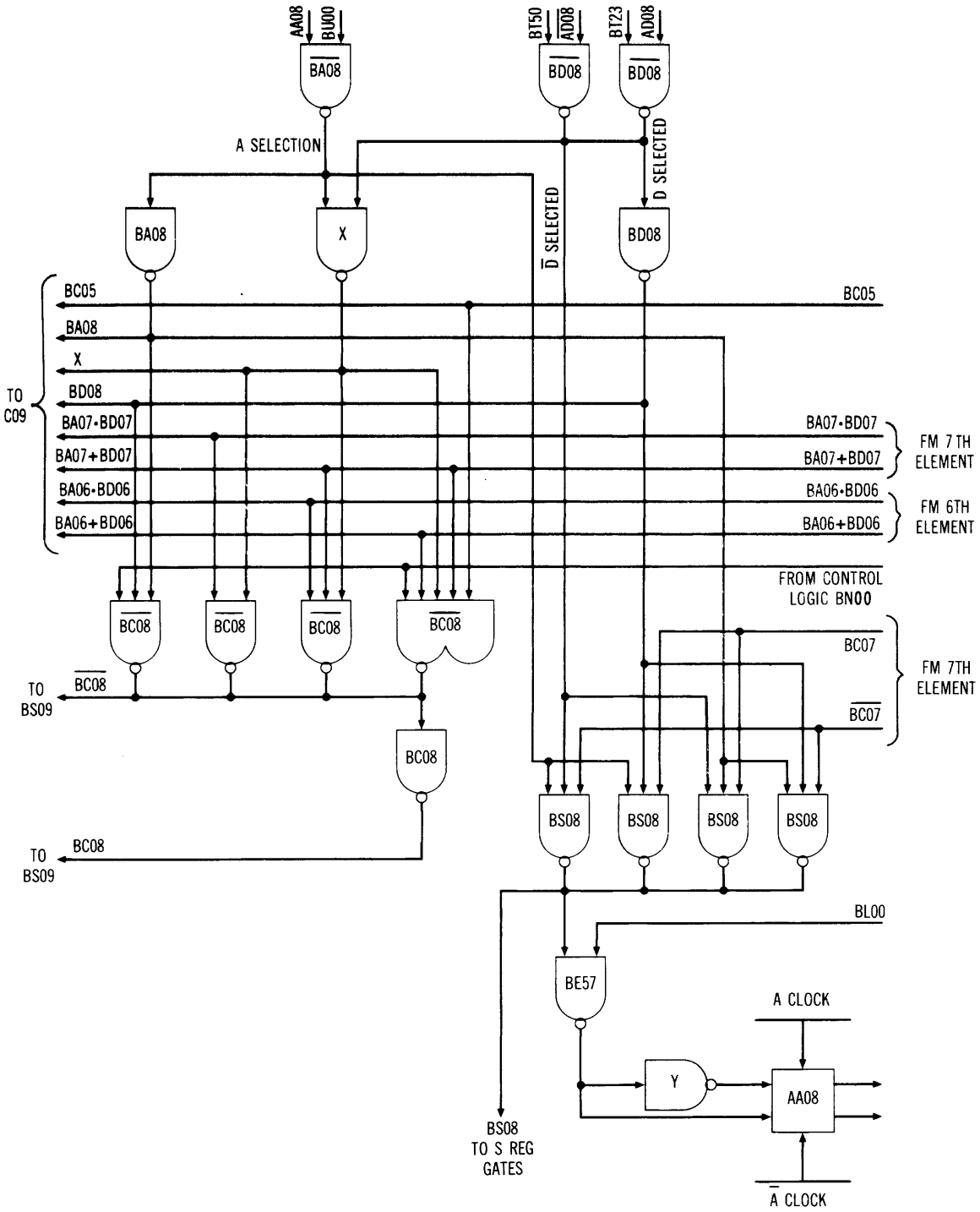


Figure 4-4-H. Sample Adder Element

(2) Q-Register. - The Q-register shown on Figures 9A28 through 9A32 serves as a secondary addressable arithmetic register. It is 24 bits in length and is comprised entirely of J-K flip flops. The set and reset sides of the Q-register are clocked by AX50 (Figure 9A11-2B). Figure 4-4-I shows a typical stage of the Q register.

(a) Inputs. - The Q-register may be loaded from the adder and manually from the indicator switches (DC set override). The source of these inputs and the associated arithmetic commands are:

Source	Command
Adder output BSS	BSi (Gate Adder to Q)
Next higher stage of Q Q_{i+1}	ARi (Shift Q right one place)
2nd higher stage of Q Q_{i+2}	ASi (Shift Q right two places)
Next lower stage of Q Q_{i-1}	ALi (Shift Q left one place)
2nd lower stage of Q Q_{i-2}	AMi (Shift Q left two places)

Another input source for the Q-register is the A-register, since contents of A may be loaded serially into Q during single or double shifts.

(b) Q-Register Outputs. - The Q-register contents may be transferred to the D-register by command BM50 and the D-register clock or serially to the A-register for AQ shifts (single or double shifts).

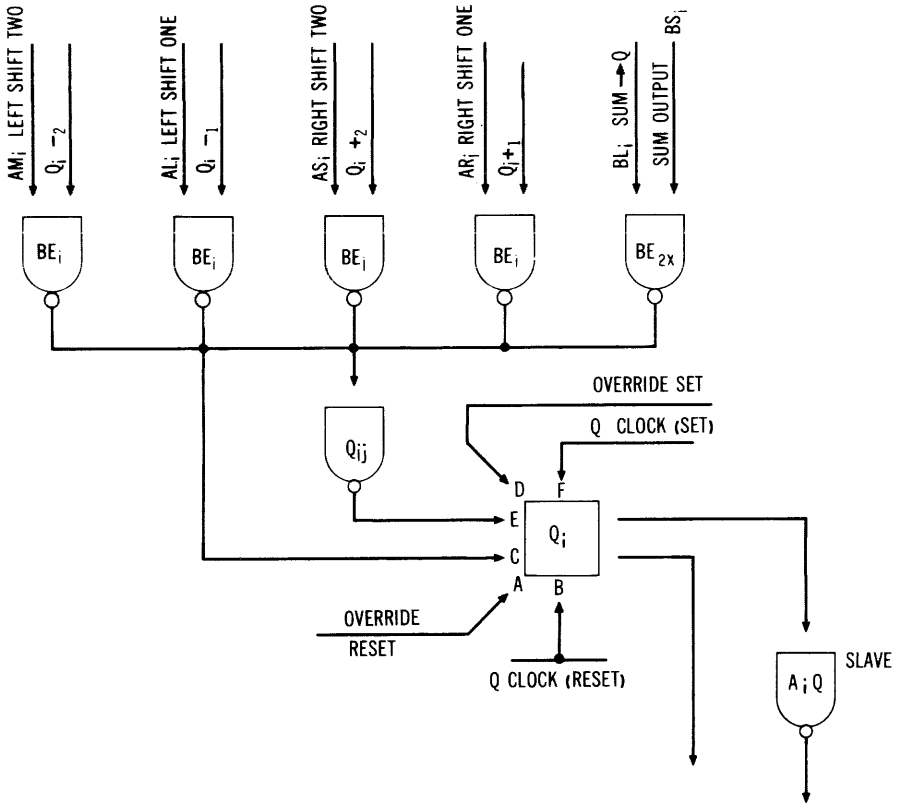


Figure 4-4-I. Q-Register Stage

(3) D-Register. - The 24-bit D-register shown on Figures 9A15 through 9A18 is the buffer register for the arithmetic section. The D-register is comprised of J-K flip flops and are clocked on the set and reset sides by AX24 and AX39 (Figure 9A10-6B). All arithmetic information and results pass through this register. In addition, the D-register serves as an instruction interpretation function for arithmetic instructions. During the Instruction Acquisition cycle, the Instruction word is gated to D as well as the I-register. If the instruction is interpreted by the I translator to be an arithmetic instruction, control is transferred to the arithmetic section and the instruction is immediately available in D for access by the arithmetic control section.

(a) Inputs. - The D-register is loaded from the Q-register, from the A-register and from the original content of D taken with logical product of Q. This latter path is utilized during execution of the SBT instruction.

The lower six bits of D may be loaded from N with proper command enables. This path is utilized during the scale instructions to permit storing the scale count.

The D-register may also be loaded from the M-register. The transfer of a logical "1" is achieved by driving BEi to logical "0". The logic for this operation is found on Figures 9M1 through 9M5.

Figure 4-4-J shows a typical stage of the D-register with the aforementioned transfer gates.

(b) Outputs. - The contents of the D-register may be transferred to the M-register, N-register (lower 6 bits only) or adder.

(4) N-Register. - The N-register is a 6-bit register-counter; it is shown on Figure 9A3 and is used to control the shifting operations during multiplication, division and shift execution. The N-register contains the scale count for a SCALE instruction, and is not directly addressable by the main program.

(a) Inputs. - The N-register may receive data from the D-register during shift instructions (EXSF) via gates BAKY (Figure 9A3-3D) and BALA. A fixed number is placed in the N-register for multiply and divide.

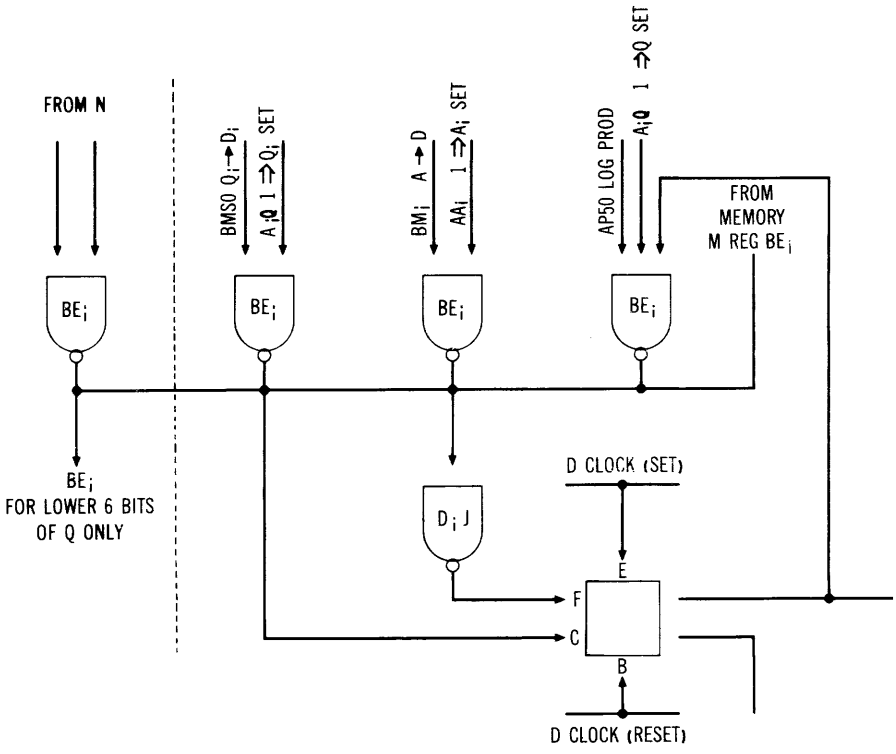


Figure 4-4-J. D-Register Stage

(b) Outputs. - The N-register may transfer information to the D-register (lower 6 bits). This path is used when storing the scale count after a scale operation.

(c) Decrementing the N-Register. - The contents of the N-register may be decremented by two on signal from arithmetic control. The N-register may also be decremented by three (the 2 least significant bit of N reset) when N = 0.00011_2.

Refer to Figures 9A3 and 4-4-K. All N-register flip flops are clocked by $\overline{AX99}$, the 1.5 mc arithmetic clock. Each even clock pulse of the Master clock produces the $\overline{AX99}$ pulse. The decrement by two command is BG00. Figure 4-4-L shows the decrementing process when this signal (BG00) is a constant logical "1". Gates A, B, C and D of Figure 4-4-K serve as "carry" or "borrow" paths and are enabled as shown in Figure 4-4-L .

(d) Clearing the N-Register. - The N-register is cleared manually by the DC reset signal AV75 (Figure 9A48-5C). It is also automatically cleared by the normal process of decrementing to a zero count.

(5) Arithmetic Function Register (AF). - This register functions to issue and hold commands to the arithmetic section as called for by the function code. The AF-register flip flops are set and cleared selectively as required by the instruction being executed, timing from the Arithmetic Control (AC)-register, and CLEA clock pulses.

(a) Inputs. - The flip flops of the AF-register are set and cleared selectively as the arithmetic instruction is executed.

(b) Outputs. - Each flip flop of the AF-register controls a specific function of the arithmetic section. These are listed in Figure 4-4-M.

AF01 - The arithmetic overflow flip flop is set during an ADD if the sign bits of A and D are identical and the carry bit from the 22nd bit position is of opposite logical value. For example, if both A_{23} and D_{23} are a logical "0" and BC_{22} equals a logical "1" AF01 will be set. Figure 4-4-O shows the logic which tests this condition during ADD. During a SUBTRACT operation AF01 is set if BC_{22} and the sign bit of D are identical and the sign bit of A is of opposite logical value (Figure 9A12-1C and 2C).

(6) The Arithmetic Control (AC)-Register. - The AC-register is a four-bit sequencing register which controls the issuance of commands to the AF-register and to the various gating and control logic within the arithmetic section. Each combination of the AC-register flip flops is called a "phase". Figure 4-4-N shows a list of these phases and those instructions in which they are used.

Refer to Figure 9A1. All AC-register flip flops are J-K type and are clocked by arithmetic clock (CLEA) pulses.

c. Arithmetic Instruction Execution. - This topic describes the step by step execution of the major arithmetic instructions. The pre-arithmetic role of the control section is not discussed. It will be assumed that the instruction is in the I-register and the operand is in the D-register in the case of ADD, SUBTRACT, MULTIPLY, DIVIDE and that the D-register contains the Instruction word in the case of SHIFTS, REVERSE BITS IN A & Q and SCALE.

(1) ADD. - The control section places the operand in the D-register and generates the command ADDX (Figure 9C4-7C). This ADDX command generates the following enables:

1) Select D-enable (Figure 9A8-7C) - allows the contents of the D-register to be gated to the adder.

2) Select A-enable (Figure 9A8-5C) - allows the contents of the A-register to be gated to the adder.

3) Carry-enable (Figure 9A8-3C) - allows the carry from each stage to be propagated to the succeeding stages.

4) A-register set and reset clocks (Figure 9A10-2C) - causes the output of the adder to be loaded into the A-register.

5) A-register input gating (Figure 9A11-8C) - allows the adder outputs (BS_i) to be gated to the A-register.

6) Adder overflow test (Figure 9A12-3C) - allows the sensing of an arithmetic overflow condition.

See paragraph 4-4b(5)(b) for a discussion of conditions which set the arithmetic overflow flip flop.

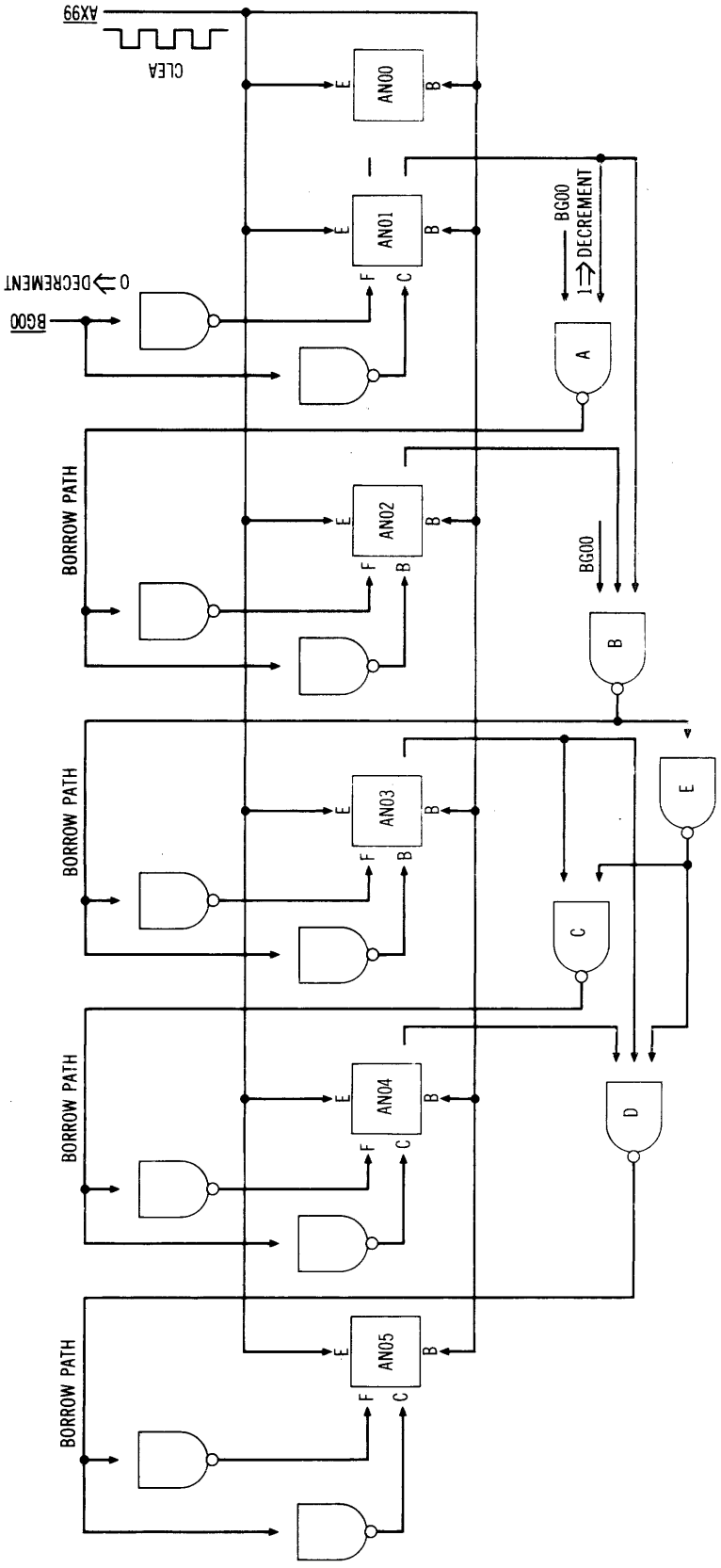


Figure 4-4-K. N-Register Counter Logic

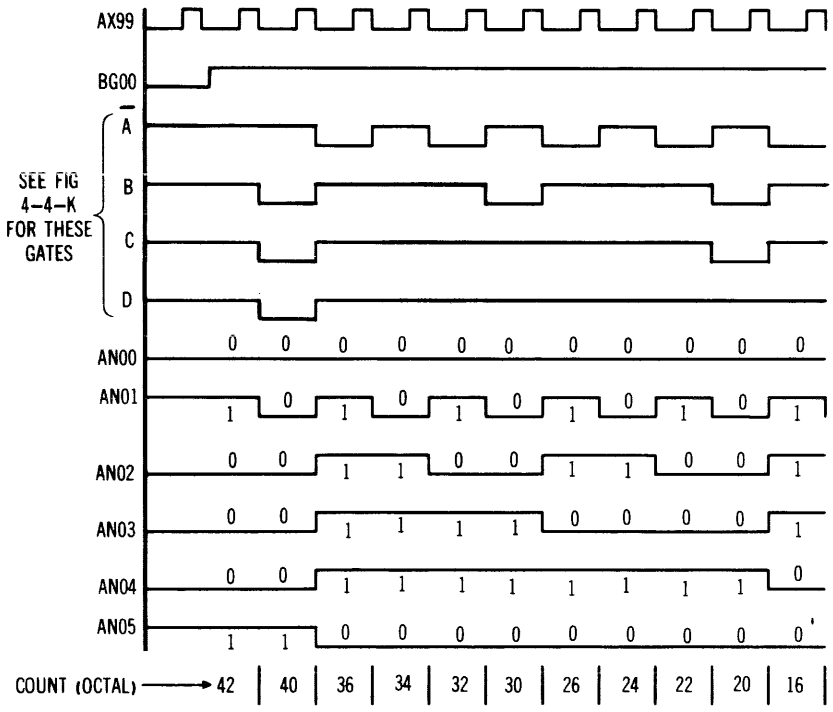


Figure 4-4 L. Decrementing Process

F/F	Name	Figure No.	
AF00	ARITHMETIC OVERFLOW	9A12	The adder overflows during add or subtract (see Figure 4-4-O).
AF01	DIVISION ERROR	9A12	
AF04	D SEL	9A4	The D-register is to be gated into the adder.
AF05	D NOT SEL	9A5	The ones complement of the D-register is gated into the adder.
AF06	A SEL	9A6	The A-register is to be gated into the adder.
AF07	A SHIFTED RIGHT ONE	9A7	The contents of the A-register shifted right one place are gated to the adder.
AF08	FORCED CARRY	9A7	A carry is forced into the low order adder stage.
AF09	CARRY ENABLE	9A7	ENABLES the adder carry circuits.
AQ50	EXT Q BIT		Extra bit of the Q-register used for multiplication and division.

Figure 4-4-M. AF-Register and Extended Q Bit (AQ50)

Phase	4321		Phase	4321	
A	0000	During all one-cycle manipulations	Q	1001	During division
B	1000	During scale A	R	0101	During division
L	0100	During division	S	1101	During division
P	1100	During division	F	0011	During reverse bits of A and Q
C	0010	During shifting			
E	1010	During scale AQ	K	1011	During multiply
M	0110	During division	G	0111	During reverse bits of A and Q
N	1110	During division			
D	0001	During division and negate A	J	1111	During reverse bits of A and Q

Figure 4-4-N. AC-Register

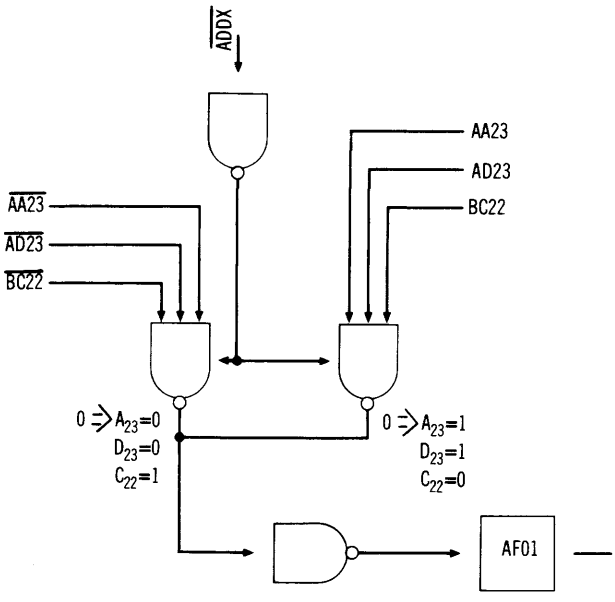


Figure 4-4-O. Arithmetic Overflow

ADD is a single cycle manipulation, i.e., the entire process takes place during a single CLEA (AX99) clock pulse. The AC-register remains at 0000 during the operation and the Arithmetic Function Register, AF, is not changed unless an adder overflow occurs.

The sum is clocked into the A-register on the trailing edge of CLEA and the ADD instruction is complete.

(2) SUBTRACT. - The control section places the command SUBT on the control lines (Figure 9C4-8C). This command, as shown in Figure 9A8-6C, generates the select $\overline{D}$ (the one's complement of D) enable which allows the complement of (D) to be gated to both the summer and the carry logic. Forced carry (BN24) is enabled which, when added to the complement of (D), produces the two's complement of (D). This addition is performed in the 2° stage of the adder during SUBT. The (A) is gated to the adder (Figure 9A8-5C) by virtue of the select A enable. The SUBT command also generates the carry enable (Figure 9A8-3B). Both A-register clocks are enabled (Figure 9A10-2C) so that the output of the adder (BS_i) can be loaded into A.

In addition, the arithmetic overflow condition is checked (Figure 9A12-1C).

As in the ADD instruction, SUBTRACT is a one-cycle manipulation, accomplished during one CLEA pulse, the trailing edge of which gates the result into the A-register. The AC-registers remains at 0000 and the AF-register is unchanged unless an overflow occurs.

(3) ENTER A. - This instruction performs no manipulation on the number and therefore is not considered an arithmetic instruction. Nevertheless, the data must pass through the adder to the A-register during the OA sequence time. This command, $\overline{\text{TRDA}}$, is generated as shown in Figure 9C4-7C and is used as illustrated on Figure 9A10-2C to generate a logical "0" out of gates BAVE and BAVU (Figure 9A10-1B). BAVU is used to generate AX00 (Figure 9A10-2C), the A-register clock (set side). BAVE is used at gate AY00 (Figure 9A9-2B) to generate the A-register clock (reset side).

The $\overline{\text{TRDA}}$ signal, from the control section, is also sent to generate BT23 (Figure 9A8-7C). The VT23 enable then allows (D) to be gated to the adder. Since neither the SELECT A ENABLE or the CARRY ENABLE is produced, the adder adds zero to (D); i.e., the adder passes the (D) directly into the A-register without altering its value. One CLEA cycle is required, and the AC- and AF-registers do not change.

(4) ENTER Q (ENQ). - The Enter Q instruction is performed in a similar manner to the Enter A discussed in the preceding paragraph. $\overline{\text{TRDQ}}$ is generated during the OA cycle (Figure 9C3-3C). This command generates AV50, AX50, AX73 (Figure 9A11-4B), and the Q-register clock (set and reset); also BT23 (Figure 9A8-8C), Select D enable, is generated.

One CLEA pulse is required and the AC- and AF-registers do not change value.

(5) SUBSTITUTE (SBT). - The substitute is similar to the ENTER A (ENA) instruction with the important exception that only those bits of D are transferred to the A-register for which Q-bits are logical "1". The command issued from the control section is SSUB (Figure 9C3-1B). This command generates $\overline{\text{BT23}}$ (Figure 9A8-7C), the D-enable to the adder.

The $\overline{\text{SSUB}}$ command also generates AU02 (Figure 9A5-2C) which is transmitted to a series of three gates on Figure 9A40-1C to selectively enable the A-register for those bits of the Q-register that are set. (Refer to Figure 4-4-C.) These gates are shown on this figure as A, B, and C. When AU02 is present, the output of gates A and C will be a logical "0" providing the respective bits of D and Q are set. This will permit a logical "1" out of gate M and the flip flop will be set. However, if the Q-bit is reset, the output of gate M will be held at logical "0" by gate B.

Also the $\overline{\text{SSUB}}$ command will generate BAVE and BAVU (Figure 9A10-2B) which will, in turn, generate AX00 and AY00 the A register clocks (set and reset sides).

(6) LOGICAL INCLUSIVE OR (IOR). - The command from the control section is $\overline{\text{INOR}}$ (Figure 9C3-2B). This command is used by the arithmetic section to generate BAVE (Figure 9A10-1B), which, in turn, generates AX00 and AX23, the A-register clock (set side).

$\overline{\text{INOR}}$ also generates the select D enable and the select A enable (Figure 9A8-7A and Figure 9A8-5C respectively). The adder carry enable is not generated. The instruction IOR, when executed, will set a given A-register bit when the respective bit of either A or D contains a logical "1". Even though three possible cases exist, only one case needs to be considered by the logic. This case is the one where the A-bit is reset and the D-bit is set.

In this situation, the output of the summer (Figure 9A40-2C for bit 2°) is a logical "1" which will enable the respective bit of the A-register to be set. In the case where both A- and D-bits are set, the A-register bit will not be reset, even though the BS_i output is a logical "0", since the reset side of the A-register receives no clocks.

(7) LOGICAL EXCLUSIVE OR (EOR). - The exclusive or command ($\overline{\text{EXOR}}$) is generated in the control section as illustrated on Figure 9C3-1B. As in the inclusive or (IOR) instruction, select D is enabled (Figure 9A8-7C), select A is enabled (Figure 9A8-8C), and both BAVE and BAVU (set and reset side of the A-clock) are generated (Figure 9A10-2B). Since the reset side of the clock is generated, the A-register bit will be reset for the case where both A and D bits are logical "1"'s.

Note that the shift, scale, multiply, reverse bits in A & Q, tally and divide instructions are all multiple cycle instructions; therefore,

they normally require more than one CLEA clock pulse for execution.

(8) SHIFTS. - During a SHIFT instruction, the arithmetic section begins its activity upon receipt of the command $\overline{EXSH}$ from the control section. The $\overline{EXSH}$ command is generated by the CLPI clock during the IN00 sequence (Figure 9C3-6C). At this time the SHIFT instruction is in both the I- and O-registers.

Upon the issuance of the command $\overline{EXSH}$ the set input of flip flop AC02 (Figure 9A1-3C) is enabled. On the trailing edge of the next arithmetic clock (CLEA) pulse AC02 is set and the system enters phase C (Figure 4-4-N). The $\overline{EXSH}$ command is also transmitted to a series of gates, shown in Figure 9A7-2C, where it is used to provide the following enables:

1) Reset Enable for AF09 (Figure 9A7-5C) - Resetting AF09 ensures the inactivity of the adder's CARRY ENABLE.

2) Reset Enable for AF05 (Figure 9A5-8C) - Resetting AF05 ensures the inactivity of the adder's SELECT $\overline{D}$ ENABLE.

3) Reset Enable for AF04 (Figure 9A4-8C) - Resetting AF09 ensures the inactivity of the adder's SELECT D ENABLE.

4) Reset Enable for AF07 (Figure 9A7-8B) - Resetting AF07 ensures the inactivity of the adder's SELECT 1/2 A ENABLE.

The command $\overline{EXSH}$ is inverted, as shown in Figure 9A6-3C and 2C to generate the EXSH and BATU signals. These signals provide the necessary enables for loading the shift count (lower 6 bits of the D register) into the N-register. In addition, EXSH partially enables the setting of AF06 (Figure 9A6-5C).

All of the above functions are enabled during the CLPI clock. On the trailing edge of the first CLEA pulse, the enabled functions are performed; that is, the shift count is clocked into the N-register; AC02 (phase C) is set, and the arithmetic function (AF) register is clocked (see Figure 4-4-P and 4-4-Q). AF04, AF05, AF07, and AF09 are cleared; AF06 is set if (A) is to be shifted. The reset output of AF06 generates the SELECT A ENABLE (Figure 9A8-5C).

The AC02 flip flop causes the phase decoder (translator) shown on Figure 9A2 to output a phase "C" condition (see Figure 4-4-N). With phase "C" active, a logical "1" occurs at the outputs of gates AC52, AC72 and AC82 (Figure 9A2-6C). The AC52 signal, in conjunction with the AD17 flip flop, enables the A-register clocks AY00, AY23, AX00, and AX23 (Figures 9A9 and 9A10). If AD16 is a logical "1", AC52 enables the Q-register clocks AX50 and AX73 (Figure 9A11).

The AC72 and AC82 signals, in conjunction with D-register bits 11, 14, 15, 16, and 17 are utilized to determine direction and type of shift (open, closed, arithmetic or logical) to be accomplished. Figures 9A20 through 9A26 show the logic required to decode the contents of the D-register and set up the required enables to carry out the specified shift. The following chart illustrates the interpretation placed upon the D-register's contents by the decoding logic.

D-Register Bit Position	Value (Binary)	Shift Logic Interpretation
11	1	right shift
11	0	left shift
14	1	normal shift
14	0	logical shift
15	1	closed shift
15	0	open shift
16	1	shift Q-register
17	1	shift A-register

The following is a brief description of the shift signals generated by the shift decoding logic.

Figure 9A19: BG00 - enables the N-register's decrementing logic, generated by BB09 during phase C and (N) > 1.

AM02 - enables a double (2 places) left shift of the A-register's contents.

AM50 - enables a double left shift of the Q-register's contents.

AS00 and BERE - enables a double right shift of the A-register's contents.

AR00 and BERB - enables a single (1 place) right shift of the A-register's contents; generated by BB03 during phase C and (N) = 0.

Figure 9A20: BE72 - enables data to be clocked into AA22 (A-register bit 22).

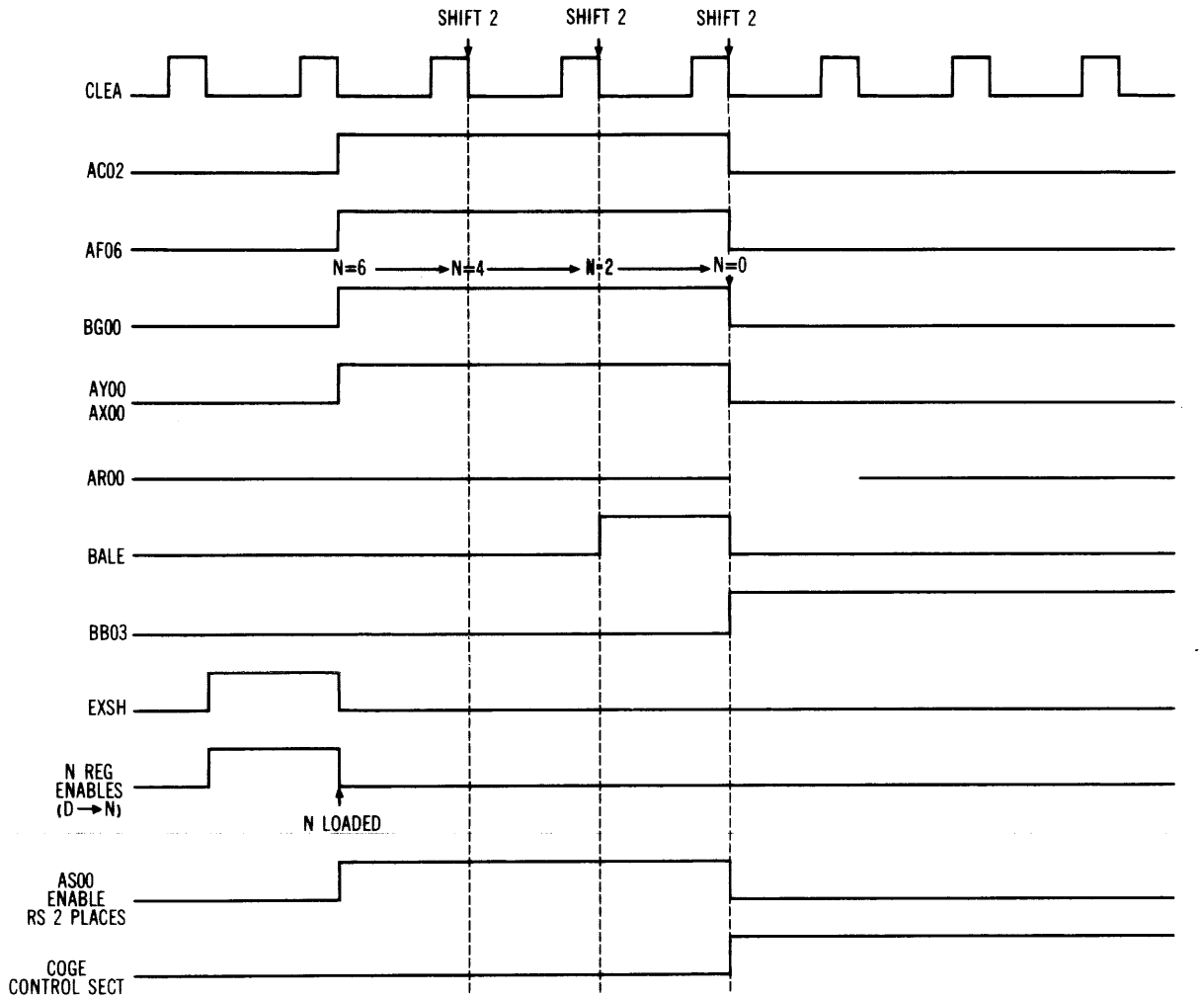


Figure 4-4-P. Right Shift A-Timing (N Count = 6)

Each gate is enabled for a specific type of shift.

AR22 - enables the shifting of data from AA21 to AA22 (shift A left 1 place).

BE73 - enables data to be clocked into AA23.

Figure 9A21: AS50 and BE97 - enables a double right shift of the Q-register's contents.

AR50 - enables a single right shift of the Q-register's contents.

AS22 - enables the shifting of data from AA23 to AA22.

BE71 - enables data to be clocked into AA21.

AL00 - enables the shifting of data from AQ23 to AA00; generated by AL01 (Figure 9A14-3B) which enables a single left shift of the A-register's contents.

AS21 - enables the shifting of data from AA23 to AA21.

Figure 9A22: BE50 - enables data to be clocked into AA00. Each gate is enabled for a specific type of shift.

Figure 9A23: AM00 - enables data to be shifted from AQ23 and AQ22 to AA01 and AA00 respectively.

BE51 - enables data to be clocked into AA01. Each gate is enabled for a specific type of shift.

Figure 9A24: BE98 - enables data to be clocked into AQ23. Each gate is enabled for a specific type of shift.

BE96 - enables data to be clocked into AQ21. Each gate is enabled for a specific type of shift.

Figure 9A25: BE75 - enables data to be clocked into AQ00. Each gate is enabled for a specific type of shift.

Figure 9A26: BE76 - enables data to be clocked into AQ01. Each gate is enabled for a specific type of shift.

BE97 - enables data to be clocked into AQ22. Each gate is enabled for a specific type of shift.

Figure 9A27: BEVU - resets AF06 and terminates phase C (resets AC02) on the last required shift.

BB03 - enables the single shift gates; requires a phase C condition and (N) equal to zero.

BB01 - signals the control section that all double shifts have been completed.

BB09 - enables the double shift gates.

The function of the N-register during a shift instruction is to keep track of the number of shifts remaining after each clock (Figure 9A3). Each clock shifts the respective data 2 places and decrements the N-register by 2 providing (N) is greater than 1. If an even number of shifts are called for, all shifts are double shifts and N is decremented by two until a count of zero is obtained. The clock which shifts the data the last two places also terminates phase C. Figure 4-4-P is a timing diagram illustrating the shift sequence for an even number of shifts.

When an odd number of shifts is called for, all shifts are double shifts and the N-counter is decremented by two until a count of three is obtained. The three-count indicates the respective data still needs to be shifted three places. On the next clock, the data is shifted two places and the N-register is decremented three places. Although the N-register is now at a zero count a data shift of one place is still required. Under this condition, phase C and (N) equal to zero, the double shift enable, BB09, (Figure 9A27) is terminated and the single shift enable, BB03, is generated. The next clock pulse shifts the data one place and terminates phase C. Figure 4-4-Q is a timing diagram illustrating the shift and N decrementing sequences for an odd number of shifts.

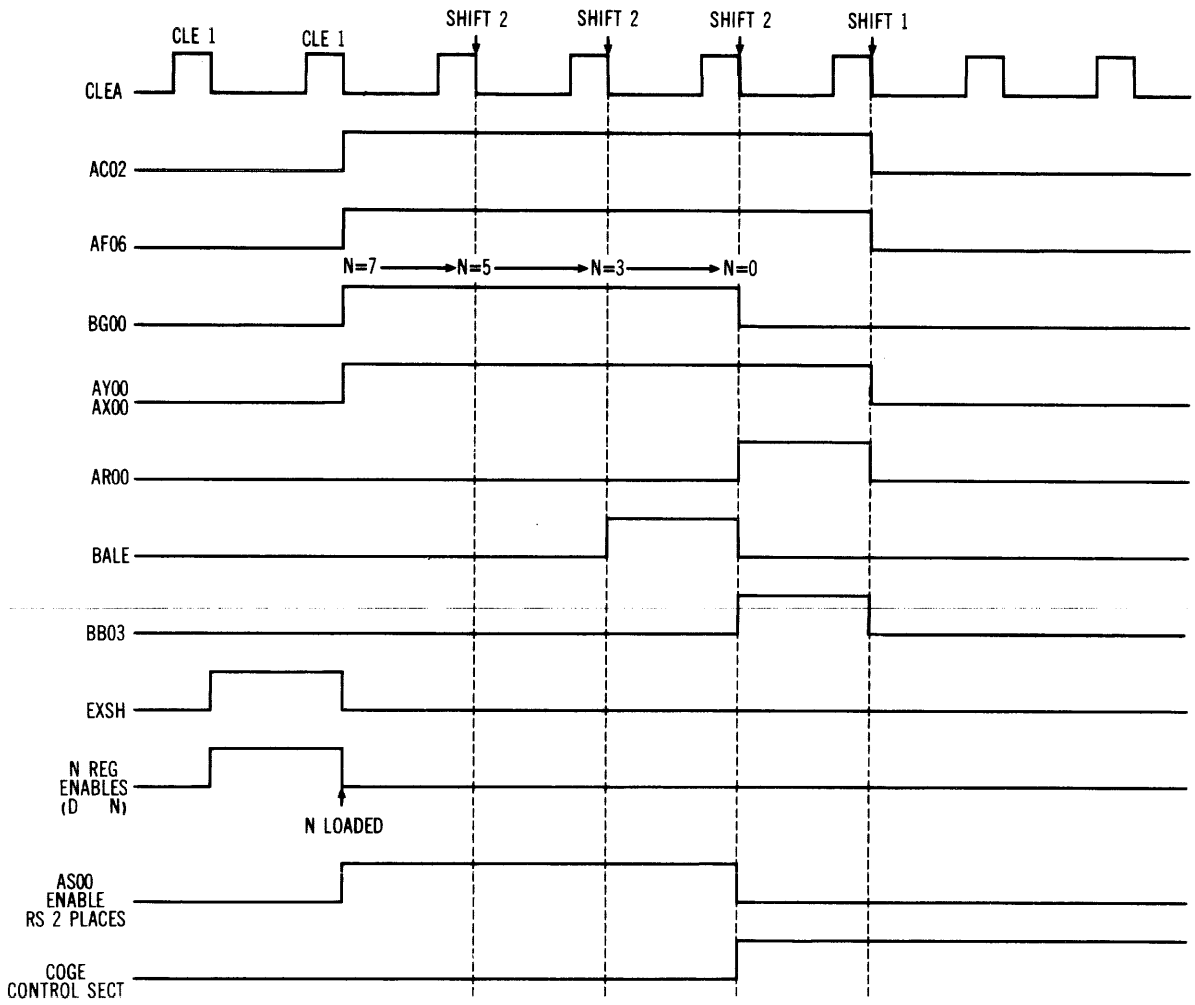


Figure 4-4-Q. Right Shift A-Timing (N Count = 7)

(9) Scale. - The scale instruction when interpreted by the control section, during the INOO sequence, transmits to the arithmetic unit the commands $\overline{SCAX}$ or $\overline{SCAQ}$ (Figure 9C6-3C) for scale-A or scale-AQ respectively. During the execution of a SCALE instruction the control section is held up, as in shifts, for an indefinite period of time (see Figure 4-4-R).

The $\overline{SCAX}$ signal accomplishes the following:

(a) Enables the setting of AC04 via BABY (Figure 9A1-7C).

(b) Enables the setting of AF06 via BEJY (Figure 9A6-5B).

(c) Enables the setting of AN05, AN04, AN03, AN00 via BAHU (Figure 9A3-7D).

(d) Enables the setting of AN02 and AN01 via AB02 (Figure 9A1-7B).

(e) Enables the clearing of AF09, AF07, AF05, and AF04.

On the trailing edge of the first CLEA clock pulse, the above flip flops are clocked to the state indicated by the respective enables. The combination of AC04 set and the remaining AC flip flops cleared signifies a phase-B condition (AC24, Figure 9A2-7C). AC24 provides the following functions:

(f) enable for clearing AN01 (decrementing via BALB (Figure 9A3-2D)

(g) a partial enable for a single shift left signal AL01 via BARE (Figure 9A3-2D). The other enable is a single shift required condition (Figure 9A14-7D).

(i) enable for clearing A select flip flop AF06 when AA21 and AA22 are of opposite state than sign bit AA23 (scale complete).

(j) Generates the A-register clocks AX00 and AY00 if SCALING is not complete (Figure 9A10-2C and 9A9-2B) respectively.

(k) Generates BG00 (decrement N-register by two) if at least two more scaling shifts are required; i. e., for (AA23) $(\overline{AA22})$ $(\overline{AA21})$ or (AA23) (AA22) (AA21).

(l) Generates the AM02 signal which shifts the A-register left 2 places (Figure 9A19-3B).

(m) If an AQ-scale is called for AX50 and AV50, the Q-register clocks and the shifting gates are also enables.

With the trailing edge of each CLEA pulse, a two-place left shift is performed and the N register-counter is decremented by two. (Refer to Figure 4-4-S.) When the gate BARF (Figure 9A19-2C) outputs a logical "1" two shifts are no longer required to bring the most significant bit of the number into bit position 22 of the A-register; instead either a single left shift or no shift is required. BARF is conditioned on $(\overline{AA23})$ (AA22) (AA21) or (AA23) $(\overline{AA22})$ $(\overline{AA21})$. When BARF is a logical "1", BG00 is dropped to a logical "0" (Figure 9A19-2B) and the N-register decrementing by two ceases. Also AM02 becomes a logical "1" which stops the shifting left by two of the A-register. Flip flop AF06 is cleared (Figure 9A6-7B) as is AC04 (Figure 9A11-5C). If an additional left shift is required, to place the most significant bit in AA22, BARE is generated (Figure 9A3-2D). BARE decrements the N-register by 1 and generates AL01 (Figure 9A14-3B). AL01 shifts the contents of the A-register left one position.

After the SCALE operation is completed, the complement of (N) is gated to the least significant six bits of Q by AN25 (Figure 9A11-5B). The control section via COHE (Figure 9C7-6C) enables the MS00 cycle to store the scale count.

(10) Multiply

(a) General Philosophy. - Binary multiplication is accomplished in a computer by a series of shifts and additions. According to the concept of place value, a left shift of one place results in multiplication by the radix of the number. Similarly a right shift of one place divides the number by the radix. Therefore, in a binary system, shifting left one place results in doubling the initial number. For example, if the binary number 0101 is doubled, the result is 1010 (decimal 5 and 10 respectively). The multiplier is examined from one end to the other, bit by bit; if the bit is a 1, an add and shift is performed; if a zero, only a shift is performed.

The process of binary multiplication can be best shown by an example. Multiply the

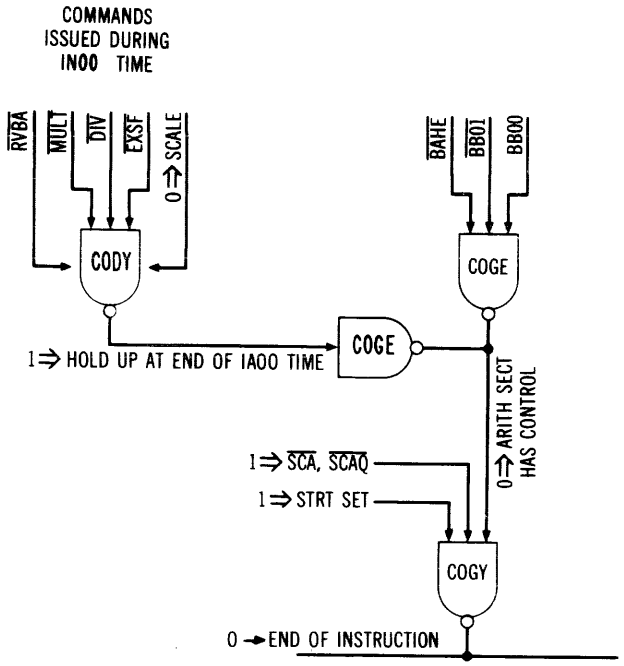


Figure 4-4-R. Control Section Hold-up for Multiply, Divide, Shifts and Scale

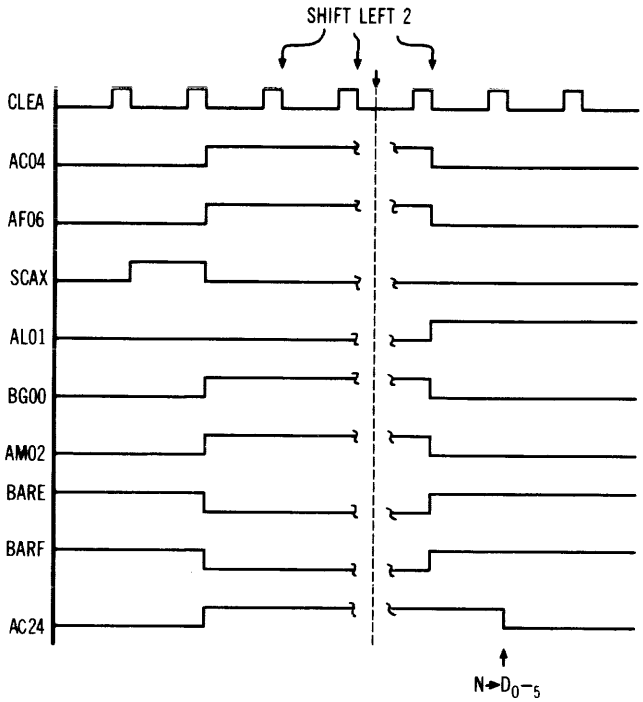


Figure 4-4-S. Scale A (Even Number of Shifts Required to Bring MSB into AA22)

decimal number 5 times decimal 7, written in binary form thus:

```
  111
x101
```

The first multiplication is accomplished as normally is done $111_2 \times 1_2$ to obtain the partial product 111_2.

```
  111
x101
 111 first partial product
```

Next partial product is shifted to the left one place (doubled) though not added to the first partial product because the multiplier is zero in the 2^1 stage.

```
  111
  101
 0111 first partial product
 1110 shifted first partial product
```

The 2^2 stage of the multiplier contains a "one" so a shift and add is performed to obtain the second and final product.

```
  111
x101
 0111 first partial product
 1110 shifted first partial product (not
      added to first partial product)
11100 second shift first partial product
100011 final product
```

Thus, the multiplication of binary numbers conforms to the following rule:

Each stage of the multiplier is considered separately and results in a left shift of the multiplicand equal to the power of the multiplier stage. If the coefficient (1000) of the stage is zero, no addition is performed: if a one, addition is performed to produce a partial product, the process proceeds from least to most significant multiplier bit.

Multiplying two numbers with negative or unlike signs may be accomplished by two's complement arithmetic without sign correction.

Recall that the 2's complement is formed by reversing the zero and ones and then adding a one to the right hand bit. The multiplication of a positive multiplier with a negative multiplicand will be used in the following example.

1 10 011	Multiplicand (2's compl) = -15_8
0 01 011	Multiplier = $+13_8$
1 11111 10 011	right hand bit x multiplicand
1 11111 00 110	next x multiplicand and shift left
1 11110 11 001	—— add

1 10 011		
0 01 011		
1 11111 10 011	*(1)	2^0
1 11111 00 110	*(1)	2^1 shift & add
1 11110 11 001	*	Sum (partial product)
1 11110 01 100	*(0)	2^2 (Shift Only) do not add
1 11100 11 000*	(1)	2^3 (Add & Shift)
1 11011 10 011		Sum (final product)

The exact effect of shifting the partial products left would be obtained if the multiplicand were shifted to the right. Most computers, including the DPS 2402, shift the multiplicand to the right.

(b) Multiplication in the DPS-2402 Computer. - Multiplication in the DPS-2402 computer follows a similar scheme of shift and add with several important variations: (1) A two bit look-ahead provides for faster multiplication by effectively handing two bits of the multiplier at a time, and (2) formation of the partial products are obtained by shifting and addition or subtraction, depending on the two multiplier bits in the Q-register.

The 2402 multiplication algorithm is implemented by the adder and four registers: AN, AA, AQ and AD as shown in Figure 4-4-T. In addition, the AC-and AF-control registers are employed.

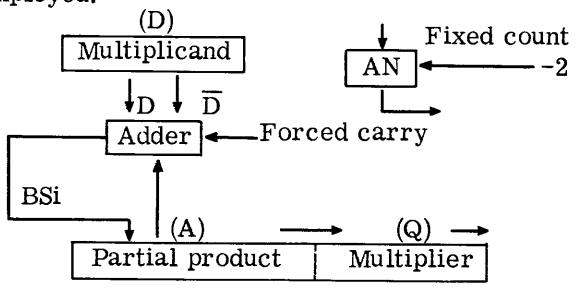


Figure 4-4-T. Adder and Four Registers

The A-register contains the partial product as the multiplication proceeds. The N-register counts down from a fixed count (26_8) and thereby ensures that the multiply will be performed in the specified number of operations.

Initially, the control section places the operand (multiplicand) in the D-register and generates the command MULT (Figure 9C3-2B) during the 0A cycle. The control section can not enable another IA cycle due to the inhibit CODY (Figure 9C7-7A and Figure 4-4-R) until the multiply is complete.

The Q-register initially contains the multiplicand (refer to Figure 9AO). While the command MULT is present, the Q-register bits 00 and 01 are sampled and, based on the table in Figure 9AO, specific AF flip flops are enabled. In addition, the A-register's reset clocks (Figure 9A10-2C) are enabled.

Finally, the AC-register flip flops AC01, AC02, and AC04 are enabled (Figure 9A1-7C). When these AC flip flops are set the arithmetic section enters phase K (AC61).

The first CLEA pulse (trailing edge) accomplishes the enabled operations. These are summarized in the multiply setup as follows:

1. Phase K entered in the AC-register
2. 26_8 (fixed count for multiply) entered into the N-register
3. AF-register set up according to initial Q register content
4. A-register is cleared

Figure 4-4-U shows a complete timing operation for a multiplication example. The numbers used are multiplier (Q)i = 01677035_8 and the multiplicand = 11335471_8. CLEA pulses numbered one through 13 clock the various operations. The N-register counts down (decrements) by two after each CLEA pulse. The arithmetic function register (AF) flip flops are enabled prior to each CLEA pulse by bits 01 through 03 of the Q-register (see Figure 9AO for a table of criteria for setting the AF flip flops. The adder performs as called for by the AF-registers current content.

Each CLEA pulse logically shifts the Q-register's content two places (open). Bits 22 and 23 of Q are set from the adder outputs BS00

and BS01 respectively. The output of the adder is shifted either one or two places to the right before entering the A-register. The AF-register's content determines the number of shifts (see Figure 9AO).

Consider the following example.

1. Multiply Setup: N = 010 110

D
001.001.011.011.101.100.111.001
A
000.000.000.000.000.000.000.000
Q
000.001.110.111.111.000.011.101

From Figure 9AO, AF04 (Select D), AF06 (Select A), and AF09 (Enable Carry) flip flops are set at the trailing edge of pulse 1 (Figure 4-4-U). With phase K enabled ($\overline{AC61}$) gate GEDA (Figure 9A19-5C) generates a logical "0" (until the last operation; N = 0). This command gates the output of the summer two stages to the right and thereby result in a "logical open right shift AQ" after each operation.

2. Following trailing edge of pulse 2 (Figure 4-4-U) the following is contained in the A-, Q-, and D- registers.

A N = 010 100
000.010.010.110.111.011.001.110
Q
010.000.011.101.111.110.000.111

Bits shifted out of AQ00 enter the Q extended bit (AQ50). AF05 (Select D), AF07 (Forced Carry), AF09 (Carry Enable) and AF06 (Select A) flip flops are set from Q 01 through 03 enables. The D register remains unchanged during the entire multiply operation.

AF04 (Select D), AF07 (1/2A select) and AF09 (Carry Enable) are enabled to be set on the trailing edge of pulse 3 from AQ01 through AQ03 translation. (See Figure 9AO.)

3. The output of the summer at this point is:

110 110 100 100 010 011 000 110	$\overline{D}$
000 010 010 110 111 011 001 110	A
110 100 110 010 101 000 001 000	Logical sum
001 100 001 001 100 110 011 101	Carry row
111.000.111.011.001.110.010.101	BSi

This quantity (BSi) is shifted right two places. Bit AA01 goes to bit AQ23 and bit AA00 goes to Bit AQ22. The Q-register is shifted right logically

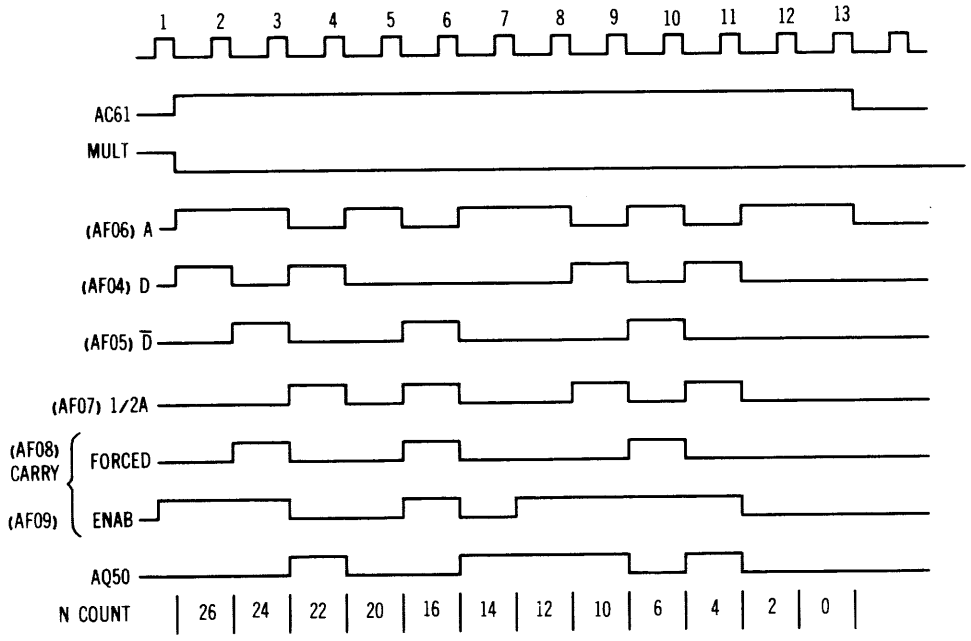


Figure 4-4-U. Multiply Timing for Example
$(Q)_i = 01677035_8$
$(Y) = 11335471_8$

two places (open ended). Bit AQ01 goes to the extension flip flop AQ50. Bits AA23 and AA22 do not change state.

The content of the A- and Q-registers after the CLEA pulse No. 3 (Figure 4-4-U) is:

N = 010 010

A

111.110.001.110.110.011.100.101

Q

010.100.000.111.011.111.100.001

The Arithmetic Function (AF) flip flops that were enabled from bits 01-03 of the Q-register before pulse 3 trailing edge are also set at this time. These are AF04 (Select D), AF09 (Carry Enable), and AF07 (1/2 A select). The N-register decrements to 22_8. (See Figure 4-4-U.)

The enables are now present from Q bits 01-03 to allow setting AF06 (Select A) on the trailing edge of pulse 4.

4. When the trailing edge of pulse 4 arrives, the actions called for by the AF flip flops are performed. Also the AF-register changes state according to the enables present, namely AF06 (Select A) set; all other AF's reset.

The Arithmetic operations called for are D+1/2A. The summer outputs are, therefore:

001001011011101100111001	D
111111000111011001110010	1/2A
110110011100110101001011	Half add
111110111111100111	Carry row
001000100011000110101011	BSi

This number is shifted right one place before being set into the A register and, as always, the Q-register is shifted right two places. The final content of the A- and Q-registers after pulse 4 is therefore:

A N = 010 000

000.100.010.001.100.011.010.101

Q

110.101.000.001.110.111.111.000

5. As indicated by Figure 9AO the Q-bits 01-03 now enable AF05 (D Select), AF07 (1/2 A select), AF08 (Forced Carry), and AF09 (Carry Enable). These flip flops are set on the trailing edge of pulse no. 5.

The output of the adder prior to pulse no. 5 is (because only AF06 set (A+0)):

000.100.010.001.100.011.010.101	A
000 000 000 000 000 000 000 000	O
000.100.010.001.100.011.010.101	BSi

On the trailing edge of pulse 5 the A-register is loaded from the summer output two places to the left, i. e., right shift two. Q is also shifted two places to the right.

A N = 001 110

000.001.000.100.011.000.110.101

Q

011.101.010.000.011.101.111.110

Pulse no. 5 also sets the AF flip flops previously enables; i. e., AF05, AF07, AF08, and AF09

6. The Q bits 01-03 enable the setting of AF06 (A select) and also enables the clearing of all other AF flip flops. This is accomplished on the trailing edge of pulse no. 6 (see Figure 4-4-U).

The adder outputs prior to pulse no. 6 (because AF05, AF07, AF08 and AF09 are set) is:

110110100100010011000110	$\overline{D}$
000000100010001100011010	1/2 A
110110000110011111011100	Half add
000001000000000000001111 1	Carry row
110111000110011111100001	BSi

At pulse no. 6 (trailing edge) the adder bits are shifted one place to the right and entered into the A-register. Q is shifted right two places.

A N = 001 100

111.011.100.011.001.111.110.000

Q

110.111.010.100.000.111.011.111

On the trailing edge of pulse no. 6, AF06 (A select) is set and all other AF flip flops cleared.

7. Q bits 01-03 enable the setting of AF06 and AF09 and the clear all other AF flip flops. This is accomplished on the trailing edge of CLEA pulse no. 7 (Figure 4-4-U). The adder outputs prior to CLEA pulse no. 7 are (with only AF06 set) A+0, or:

111.011.100.011.001.111.110.000 BSi

At CLEA pulse no. 7, this output is shifted right two and entered into the A-register. Q is shifted right two places.

A N = 001 010
111.110.111.000.110.011.111.100
Q
001.101.110.101.000.001.110.111

After pulse no. 7 AF06 and AF09 are set. All other AF flip flops cleared.

Q bits 01-03 enable the setting of AF04 (Select D), AF07 (1/2A select), and AF09 (carry enable). These are set on CLEA pulse no. 8.

8. At CLEA, pulse no. 8, the quantity zero is added to the A-register and BSi shifted right 2 places. Q is also shifted two places.

A N = 001 000
111.111.101.110.001.100.111.111
Q
000.011.011.101.010.000.011.101

Also on pulse no. 8, AF04, AF08, AF07 and AF09 are set. All other AF flip flops are cleared. Q bits 01-03 (figure 9AO) enable AF06, AF05, AF07, and AF09 which are set on CLEA pulse no. 9.

9. The adder output at pulse no. 9 is:

001.001.011.011.101.100.111.001	D
111 111 110 111 000 110 011 111	1/2A
110 110 101 100 101 010 100 110	Half add
111 111 111 110 011 001 111 111	Carry row
001 001 010 010 110 011 011 001	BSi

This adder output is shifted right one place and entered into the A register at the trailing edge of pulse No. 9. Q is shifted right 2 places.

A N = 000 110
000.100.101.001.011.001.101.100
Q
010.000.110.111.010.100.000.111

Also AF05, AF06, AF07, and AF09 are set. The Q bits 01-03 are such as to enable (see Figure 9AO) the setting AF04, AF07, and AF09 at the trailing edge of pulse 10.

10. The adder output during pulse no. 10 is:

110110100100010011000110	$\overline{D}$
000100101001011001101100	A
110010001101001010101010	Half add
001001000000100110011	Carry row
111011001101101100110011	BSi

The adder output is shifted two places right and entered into the A-register. Q is shifted right two places.

A N = 000 100
111.110.110.011.011.011.001.100
Q
110.100.001.101.110.101.000.001

Also AF04 and AF07 are set. The Q bits 01-03 provide enables for setting AF06 after the trailing edge of pulse 11.

11. The adder output at pulse no. 11 is:

001001011011101100111001	D
111111011001101101100110	1/2A
110110000010000001011111	Half add
111110110111011011	Carry row
001000110101011010011110	BSi

This adder output is shifted right one place and entered into A. Q is shifted right two places.

A N = 000 010
000.100.011.010.101.101.001.111
Q
101.101.000.011.011.101.010.000

Also AF06 is set and all other AF flip flops are cleared. With the above value in the Q-register, AF06 is again enabled and is set following pulse no. 12.

12. The adder output at pulse no. 12 is:
(A+0)

000100011010101101001111 BSi

This output is shifted right 2 places and gated into A. Q is shifted right two places.

A N = 000 000
000.001.000.110.101.011.010.011
Q
111.011.010.000.110.111.010.100

With the N register equal to zero the output of the adder at the trailing edge of pulse 13, is A+0 or

000001000110101011010011

For this example, the adder output is shifted one place to the right and entered into A. Q is shifted right two places. The AC and AF-registers are cleared. The final result of the multiply operation is:

A
000.000.100.011.010.101.101.001
Q
011.110.110.100.001.101.110.101

(11) NEGATE A. - This instruction complements the content of the A-register i.e., replace the content of the A-register with its 2's complement value.

The instruction translator transmits to the arithmetic section the command $\overline{\text{NEGA}}$ = logical "0". (Figure 9C1-3C). This command generates the following enables in the arithmetic section:

(a) Enable setting AC01 via BAGA (Figure 9A1-1C)

(b) Enable reseting AF04 via BERX (Figure 9A4-8C)

(c) Enable setting AF05 and AF08 via BALK (Figure 9A5-4B)

(d) Enable setting AF09 via BERF (Figure 9A7-4C)

(e) Enable the D register clock via AX25 (Figure 9A10-5B)

(f) Enable an A → D transfer via BMi (Figure 9A11-6B)

The trailing edge of the first CLEA clock sets the enabled flip flops and transfers the contents of the A-register to the D-register. With flip flop AC01 being set the phase decoder registers a phase D ($\overline{\text{AC71}}$, Figure 9A2-5C) condition. The command $\overline{\text{AC71}}$ generates the command BLOO (Figure 9A11-7B) which gates the adder output to the A register.

When the second CLEA clock pulse arrives (Figure 4-4-V), the entire operation is simultaneously accomplished and the control flip flop (AC01), and AF05, and AF09 are cleared. Specifically, the two's complement of the D register is transferred to the A register. The two's complement of D is the one's complement of D plus the forces carry. Therefore, (A) is not gated into the adder; the two quantities added are 1 (forced carry) and the contents of the D register.

(12) Reverse Bits in A and Q. - This instruction reverses the bits in the A- and Q-registers such that the content of position 0 is shifted to position 23 and from position 23 to position 0.

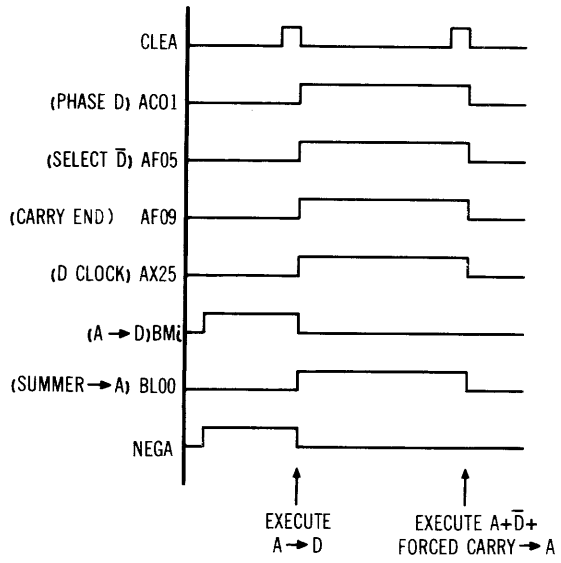


Figure 4-4-V. Negate the A-Register

All other positions of the registers are likewise interchanged.

The control section generates the command $\overline{\text{RVBA}}$ (Figure 9C3-5C) during the IN00 cycle. The arithmetic section uses this command to provide the following enables:

1. Enable setting N to 30_8 via BAVA (Figure 9A6-2C)
2. Enable setting AF06 (Select A)
3. Enable setting AC01 and AC02 (Phase F) (Figure 9A1)
4. Holds up Control section (Figure 9C7-7D and Figure 4-4-R)

The first CLEA pulse (trailing edge) accomplishes the above operations (see Figure 4-4-W).

With phase F in the AC-register, the following enables are provided:

1. BG00 to count down N via Figure 9A19-2B
2. AM02 (shift A left two places) if N≠0 via Figure 9A27-4C
3. AS50 (shift Q right two places) if N≠0 via Figure 9A27-4C
4. BS23 Q22 vis Figure 9A26-6C
5. Q00 A01 via Figure 9A23-4C

A and Q clocks are enabled by BB05 (Figure 9A27-4C). Each CLEA pulse causes the A register to shift right and the Q-register to shift left by two utilizing the paths shown in Figure 4-4-X. The 12 required shifts places the original AA23 in AQ00.

The N counter decrements to zero as detected by gates BB05 and BB06 (Figure 9A27-4C). At this time the bits are reversed in order but in the wrong register, i.e., the original bits of A are in an inverted order in the Q-register and the original bits of Q are in an inverted order in the A-register. The actions that

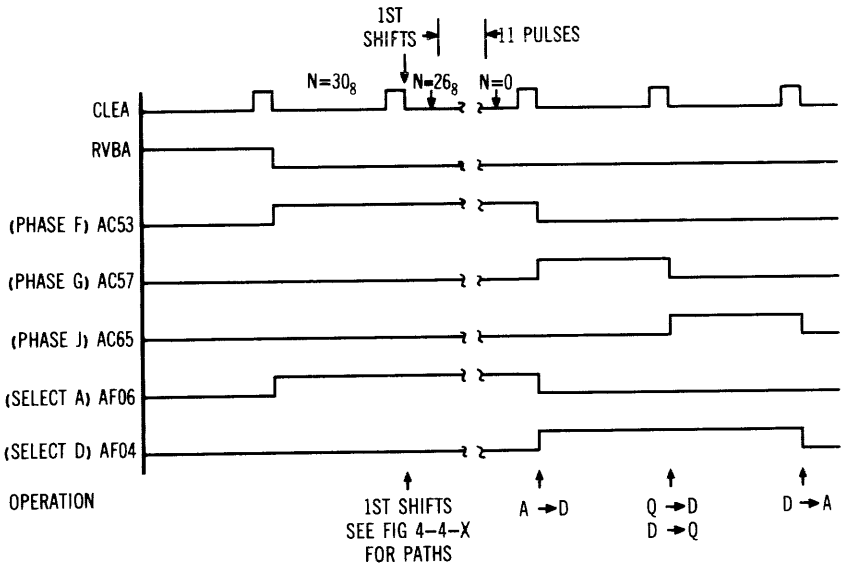


Figure 4-4-W. $\overline{\text{RVBA}}$ Timing

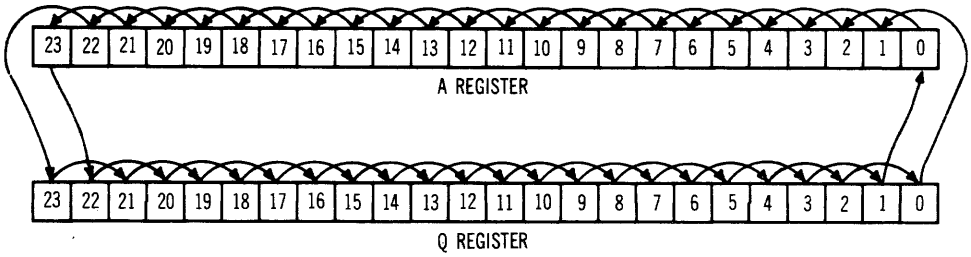


Figure 4-4-X. Shifting Paths (Phase F)

follow, therefore, are designed to switch the contents of the A- and Q- registers.

With the N counter equal to zero, the shifting ceases (Figure 9A27-4C) and AF06 is cleared (Figure 9A52-4C). With phase F and N=0 gate BB07 is enabled (Figure 9A27-5C). This gate enables the setting AC03 (phase G). This gate also enables the D register's set and reset clocks (Figure 9A10-5B) and the D select flip flop AF04. In addition, gate BB07 provides the A→D transfer enable (Figure 9A11-6B). The next CLEA clock transfers (A) to D and sets AF04 and AC03. With phase G in the AC register, the following enables are issued by $\overline{AC79}$ (Figure 9A2-7C):

1. Q to D transfer via Figure 9A9-4B
2. D to Q transfer via Figure 9A14-4B
3. A and Q register's set and reset clocks (Figures 9A10 & 11)
4. The setting of AC04 (phase J)

As shown on Figure 4-4-W, the contents of the D- and Q- registers are interchanged and phase J is entered. The phase J command AC65 (Figure 9A2-4C) produces enables to clear the AC-register, clock the A-register (Figure 9A10-2C) and gate the adder output to A (Figure 9A11-7B). The D select flip flop AF04 is still set, therefore AC65 also enables the clearing of AF04 (Figure 9A4-6A). The final CLEA pulse completes the instruction with the D to A transfer. The AC-register is cleared (return to phase A) releasing the control section "hold up" thus permitting the next IA cycle to begin.

(13) Divide. - The divide instruction divides the combined contents of the A- and Q-registers by the divisor which is placed in the D-register by the control section during the OA cycle. The quotient, after the instruction is executed, is found in the Q-register and the remainder in the A-register.

(a) General Philosophy of Computer Division. - As might be expected, computer division is the inverse of multiplication. Where the multiplication process utilized a series of right shifts and additions, the division operation uses left shifts and magnitude subtractions. In many ways, computer division is similar to the well known pencil and paper process. There are several important differences, however. The first performs the arithmetic; the second cycle enters the quotient bit and performs a single left arithmetic shift of the Q-register. The adder outputs are shifted left one place and entered in the A register.

First examine the division process for positive numbers as illustrated by Figure 4-4-Y.

1. The divisor is subtracted from the A register. If the difference is positive a "1" is placed in the least significant bit of Q as both A and Q area shifted arithmetically one place to the left.
2. Following the first subtraction, the next operation is based upon whether the adder output (BSi) from the previous operation is a positive or negative number. If positive; i. e. , the original A register content was greater in magnitude than the divisor, a divide overflow occurs in which case the quotient is greater than the maximum Q register capacity. The divider error indicator is set and the divide aborted. If the adder output is negative after this first subtrac tion, a "0" is placed in the least significant bit of Q and A and Q are shifted left one place. The next operation in this case is a subtraction. After the first subtraction, if the

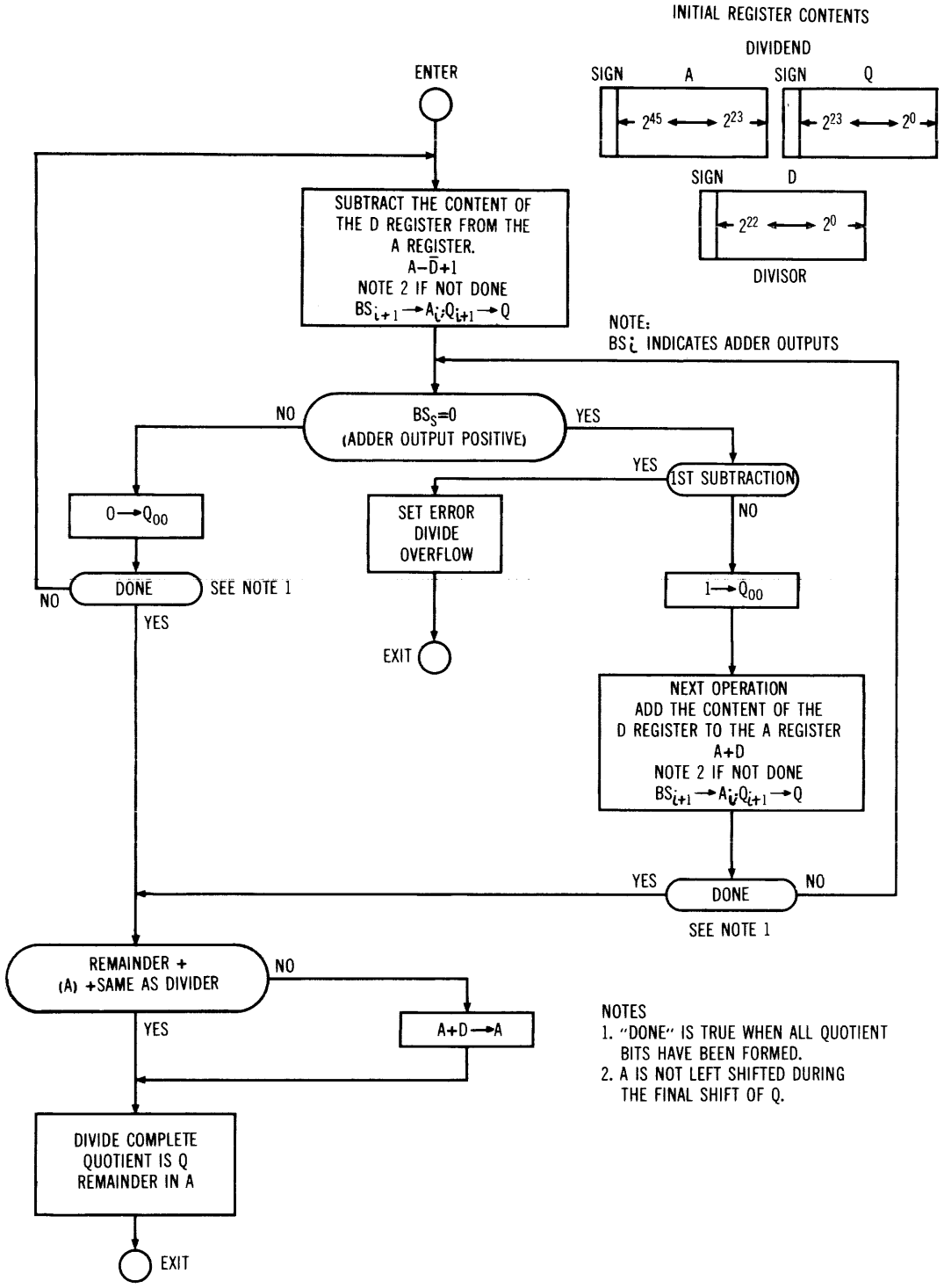


Figure 4-4-Y. Divide Flow (For Positive Numbers Only)

adder output is positive, the least significant bit of Q is set to "1" as the left shift is performed and the next operation is an addition. On the other hand, if the adder output is negative, Q is reset as A and Q are shifted left one place the next operation in this case is subtraction.

3. After all quotient bits have been formed in this manner, the division process is done except for "restoring" the remainder. For positive numbers no restoring of the quotient, (Q), is necessary. The remainder, (A), should be positive. If it is not, the divisor is added to it to form the final remainder in A.

Consider the following example:

EXAMPLE 1: Pencil and paper method

(Two positive numbers with remainder = 0)

```
                 101.
        0110. ⌈011110.
              0110
              ------
              000110
                 110
                 ---
                 000 = Remainder
```

In order to illustrate how the computer performs this operation, assume four-bit registers hold these numbers (the same rules apply to 24-bit numbers). Enter the divisor into D = 0110, the dividend into A = 0110 and Q = 0110. Note that the extreme left hand bit of both quantities is reserved for the sign bit. All register contents A, D and Q are positive in the example and therefore have a maximum magnitude of +7.

Initially (D) = 0110 (A) = (0011) (Q) = (0110)

(D) $\lceil$(A) (Q)

0110. $\lceil$ (0011) (0110).

Refer to Figure 4-4-Y. In the computer, the first operation is a subtraction of (D) from (A). Of course in the DPS-2402, this is actually (A) + $(\overline{D})$ + 1. Hence

```
(0011) (0110) (AQ)   Note: Q is not subtracted.
 1001    D
 ----
 1010  half add
 0110  carry row
 ----  forced
 1101  BSi adder output ⟶ A(i+1)
```

Observe that the sign bit of our four bit example adder has become negative. A sign change occurred because the initial content of the A-register was positive and smaller than the multiplier. No divide overflow will occur. The Q-register is open shifted left one place arithmetically. The most significant magnitude bit is shifted to the least significant bit of A. A "0" is entered into the least significant bit position of Q because the adder output was negative. The new contents of AQ is 10110100.

Also, a consequence of the negative adder output is that an addition is enabled for the next operation (A+D). Thus:

```
   A      Q
(1011) (0100)  (AQ)
 0110          D
 ----
 1101  half add
  110  carry row
 ----
 0001  BSi ⟶ A(i+1)
```

This time the adder sign bit is positive. Therefore a "1" is entered into the least significant bit of Q as the adder output is gated to A_{i+1} and the Q-register is shifted left one place arithmetically. Because of the positive adder output, a third subtraction is performed. The content of the AQ-registers following the shifting is:

```
0011  0001

   A      Q
(0011) (0001)  (AQ)
 1001
 ----
 1010  half add
 0110  carry row
 ----  forced
 1101  BSi ⟶ A(i+1)
```

After this subtraction the adder sign is negative. Therefore, a "0" is placed in the least significant bit of the Q-register as the AQ left shift is accomplished. An add is indicated for the next operation. The contents of the A- and Q-registers after this operation is:

```
1010  0010
```

The add operation is accomplished

```
1010  0010
0110
----
1100  half add
  11  carry row
----
0000
```

The A-register is not shifted and the remainder 0000 is correct. Since the adder output is positive the Q-register bit 0 is set to a logical "1" and the Q-register is left shifted one place to form the quotient 0101.

A summary of the foregoing operation is shown in Figure 4-4-Z. The quotient and remainder required no corrections for this example. For some combinations of signed numbers, quotient and remainder corrections are necessary after the shift and arithmetic operations have been completed. These criteria are discussed in the next topic.

(b) General Analysis of Division by the DPS-2402 Computer. - The DPS-2402 computer is capable of dividing both positive and negative numbers. When divisor and dividend are of unlike sign, a negative quotient is developed and the remainder, if any, is also negative (i. e. , appear in 2's complement form).

The division process is essentially divided into two major operations: (1) shifting-arithmetic step, and (2) correction of quotient and remainder. Each of these may be broken down into steps (phases):

	Phase	General Function
Shifting-arithmetic steps	A	Initial Setup for the divide and first arithmetic step-shift
	L	Check for Divide Error
	M	Arithmetic-shifting steps
Correction of quotient and remainder	N P D	Correct remainder if necessary
	Q R D	Correct quotient if necessary
	A	Complete divide, exit

Refer to Figure 4-4-AA for a detailed flow diagram of each of these phases.

The rules for the arithmetic-shifting step operation are identical for all combinations of positive and negative numbers. They may be simply stated as follows:

1. The step-arithmetic operations add or subtract A and D.

2. The type of arithmetic operation is always chosen so that the magnitude of the partial remainder in the A-register will be diminished. For example, if (A) is a positive number and (D) is negative, an add is called for. If (A) is negative and (D) is negative, a subtract is performed.

3. The adder output (BSi) is shifted one place left and gated to the A register. If the sign of the adder output BS23 is the same as the divisor set AQ50; otherwise reset it. Then the Q register and extension bit are shifted left one place arithmetically; i. e. , Q23 is not changed and Q22 goes to A00.

This method, known as the "non-restoring" method, develops quotient bits in the Q register as the shifting proceeds and partial remainders in the A register. When 23 quotient bits have been formed, the terminal phase is entered to determine if correction of quotient and/or remainder is necessary and perform the necessary corrections. Following this, phase A is entered and the arithmetic section (via CODY - Figure 9C7) allows the control section to enter the next IA00 cycle.

(c) Detailed Analysis of DPS-2402 Division. - The divide operation begins, as far as the arithmetic section is concerned, with the command DIVX (Figure 9C3-2B) from the control section. D holds the divisor and AQ contains the dividend.

Referring to Figure 4-4-AA, and Figure 9A0, it is seen that a decision is made for the first operation, i. e. , add or subtract based on the signs of the divisor and the dividend.

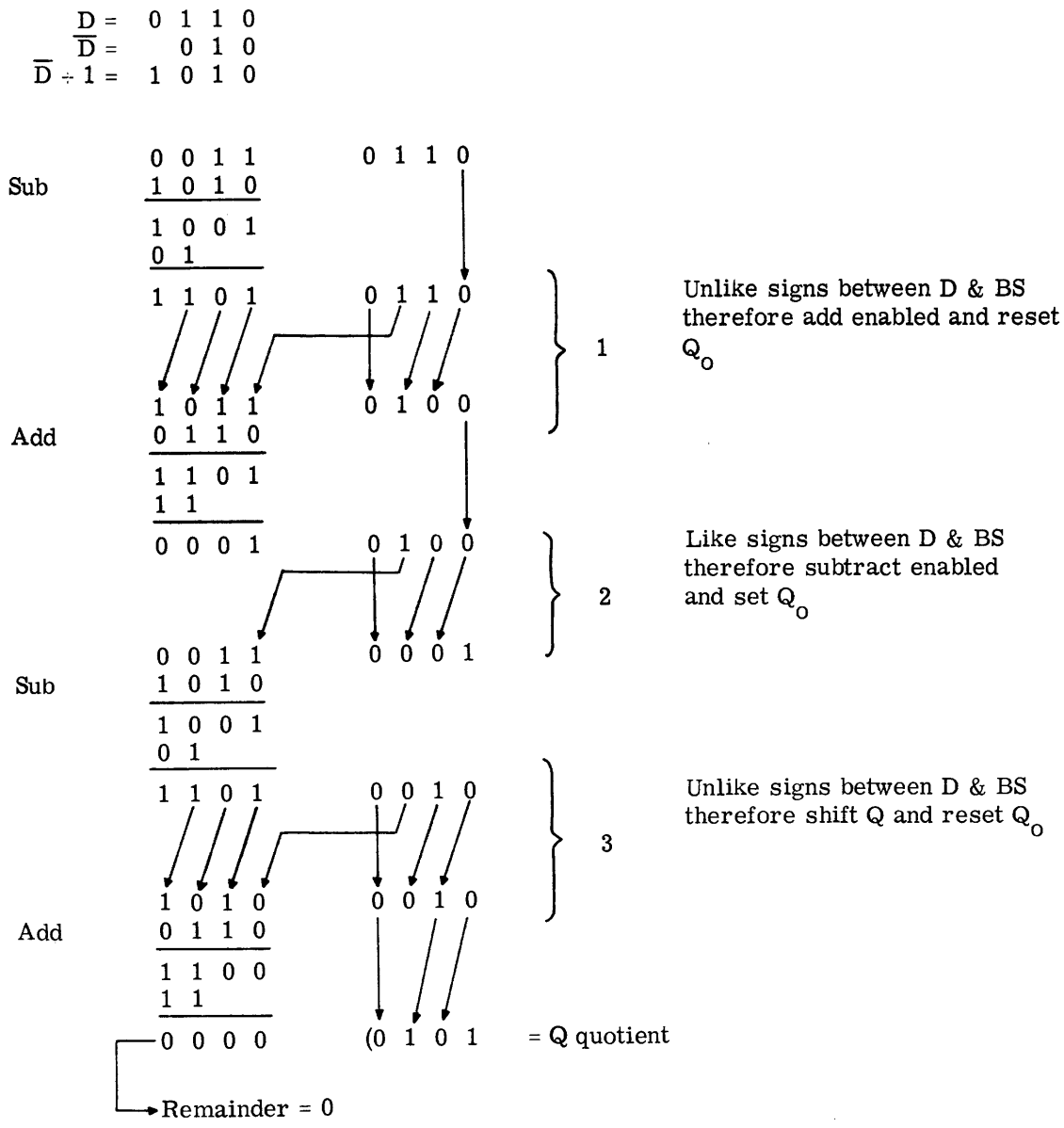


Figure 4-4-Z. Pos Quotient (No Remainder)

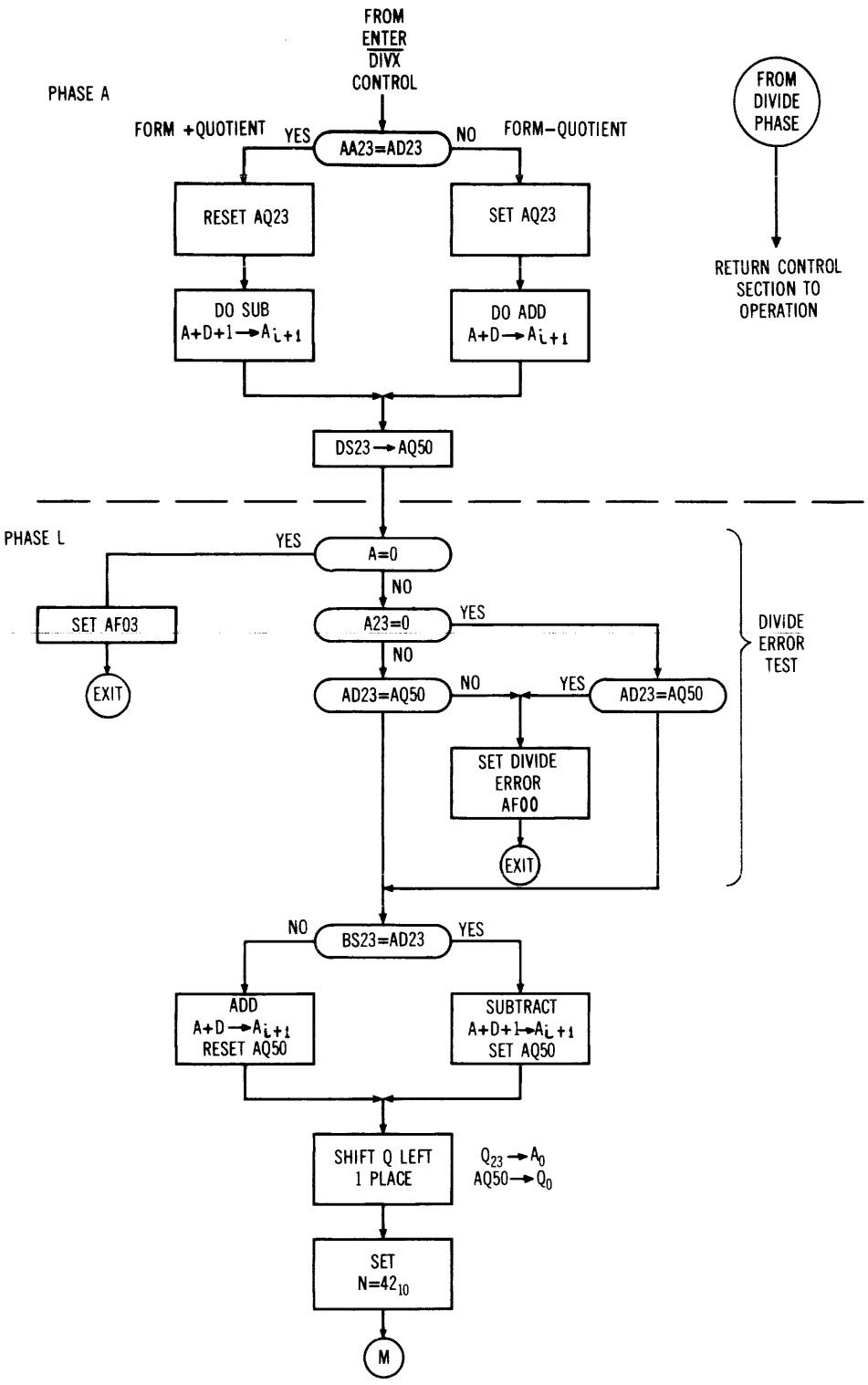


Figure 4-4-AA. Divide Flow (Sheet 1 of 3)

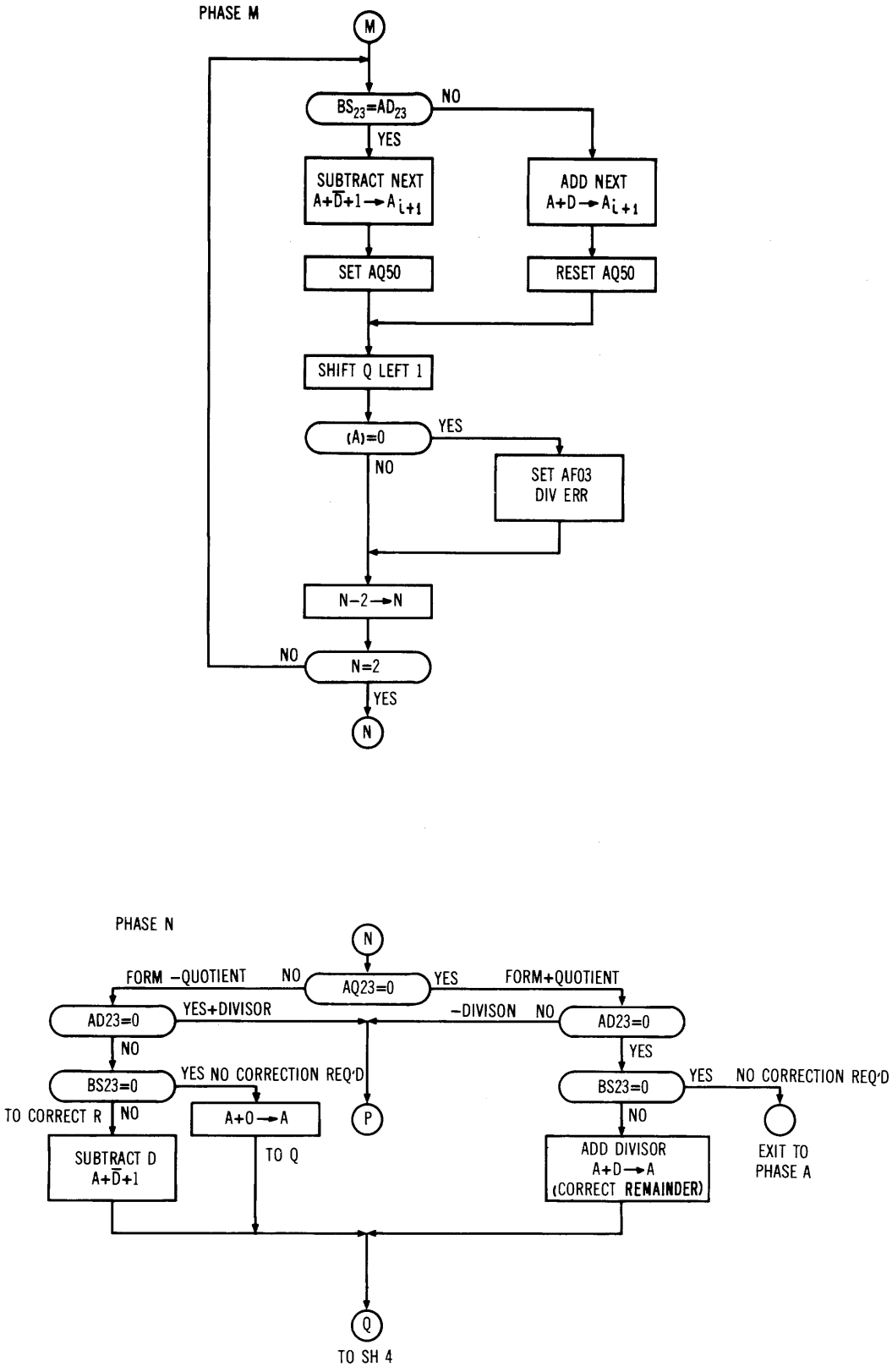


Figure 4-4-AA. Divide Flow (Sheet 2 of 3)

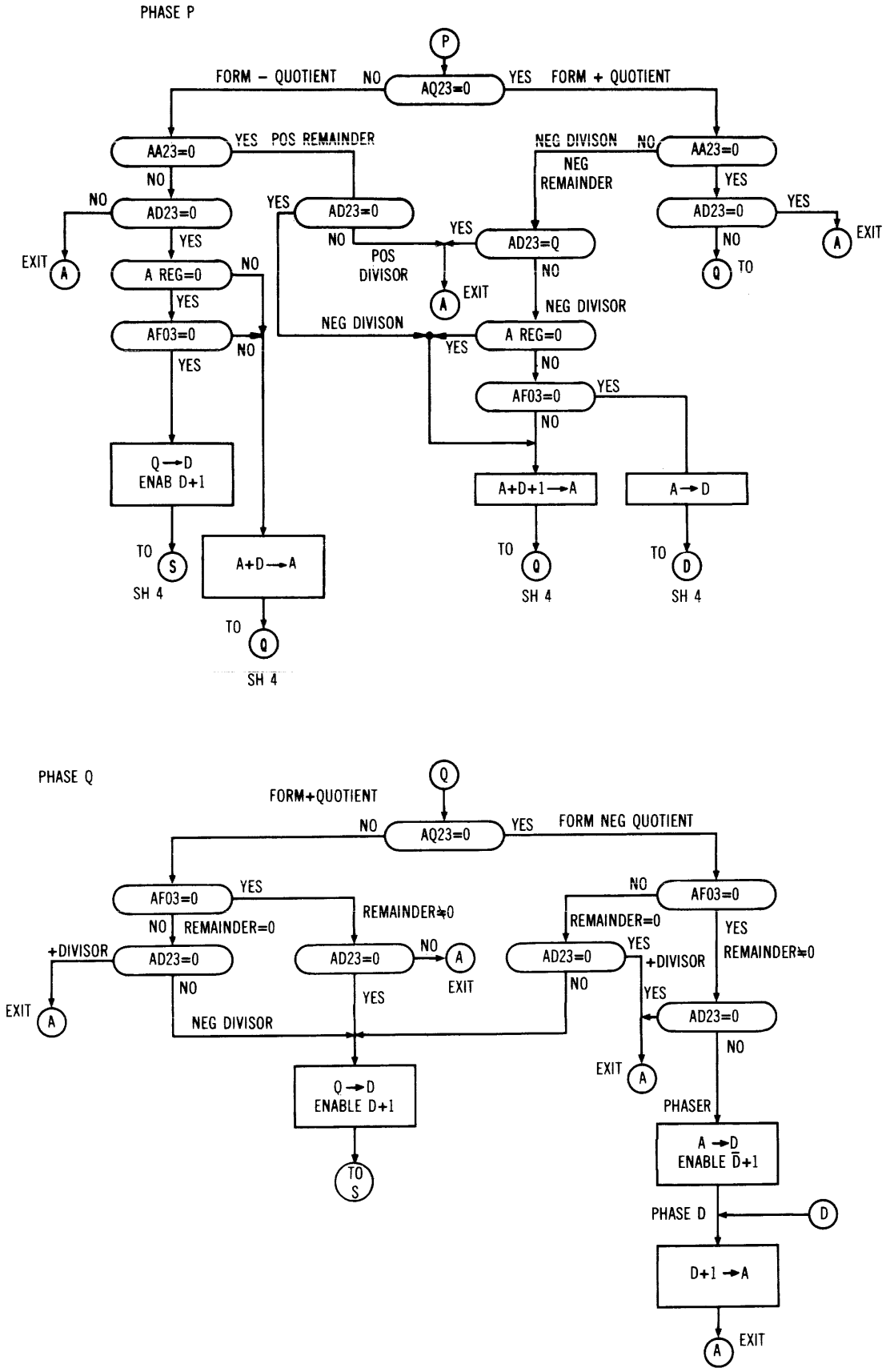


Figure 4-4-AA. Divide Flow (Sheet 3 of 3)

The $\overline{\text{DIVX}}$ signal generates BAXY (Figure 9A6-1C); BAXY, in turn, enables entry into phase L (Figure 9A1-5C) and enables the setting of the D select (AF04) flip flop (Figure 9A4-5B) if an add is in order for the first operation. If a subtract is required by the signs of A and D, BALN (Figure 9A4-7B) which clears D select (AF04) and sets "D NOT" select (AF05) is enabled.

An additional function of BAXY (Figure 9A6-1C) is to set AQ23, the sign bit of the Q register to a logical "1" if a negative quotient results, i. e. , unlike signs of dividend and divisor. This operation is performed by a gate at Figure 9A24-6C. The $\overline{\text{DIVX}}$ signal, in addition, generates the adder carry enable (Figure 9A8-3C), select A enable (Figure 9A8-5C), and the A register's clocks (Figure 9A10-2C).

Signal AL01 (Figure 9A14-3B) is enabled from DIVX to gate the output of the adder left one place to the A register stages. The DIVX signal holds up the control section via CODY (Figure 9C7) until the divide instruction is complete.

The first CLEA pulse (trailing edge) sets the flip flops previously enabled and essentially performs the first arithmetic shift-step.

The arithmetic section enters Phase L (AC54) shown on Figure 9A2-6C.

At this time the DIVIDE ERROR flip flop AF00 is enabled if the magnitude of A exceeds the magnitude of D; i. e. , a divide overflow will occur and the quotient will overflow the Q register. See Figure 4-4-AA, sheet 1.

The signals AC54 and AC74 (phase L) enable the setting of the N register to 57_8 (Figure 9A3). This signal (AC54) also enables the logic for entering phase N (Figure 9A1-7C) for gating bits AQ22 to AA00 (Figure 9A22-6C) and AQ50 to AQ00 (Figure 9A25-5C). The second CLEA pulse caused the arithmetic operation to enter phase M, begin continuous arithmetic-shift steps, and to decrement the N register by two as called for by $\overline{\text{AC56}}$ (phase M Figure 9A2-1C).

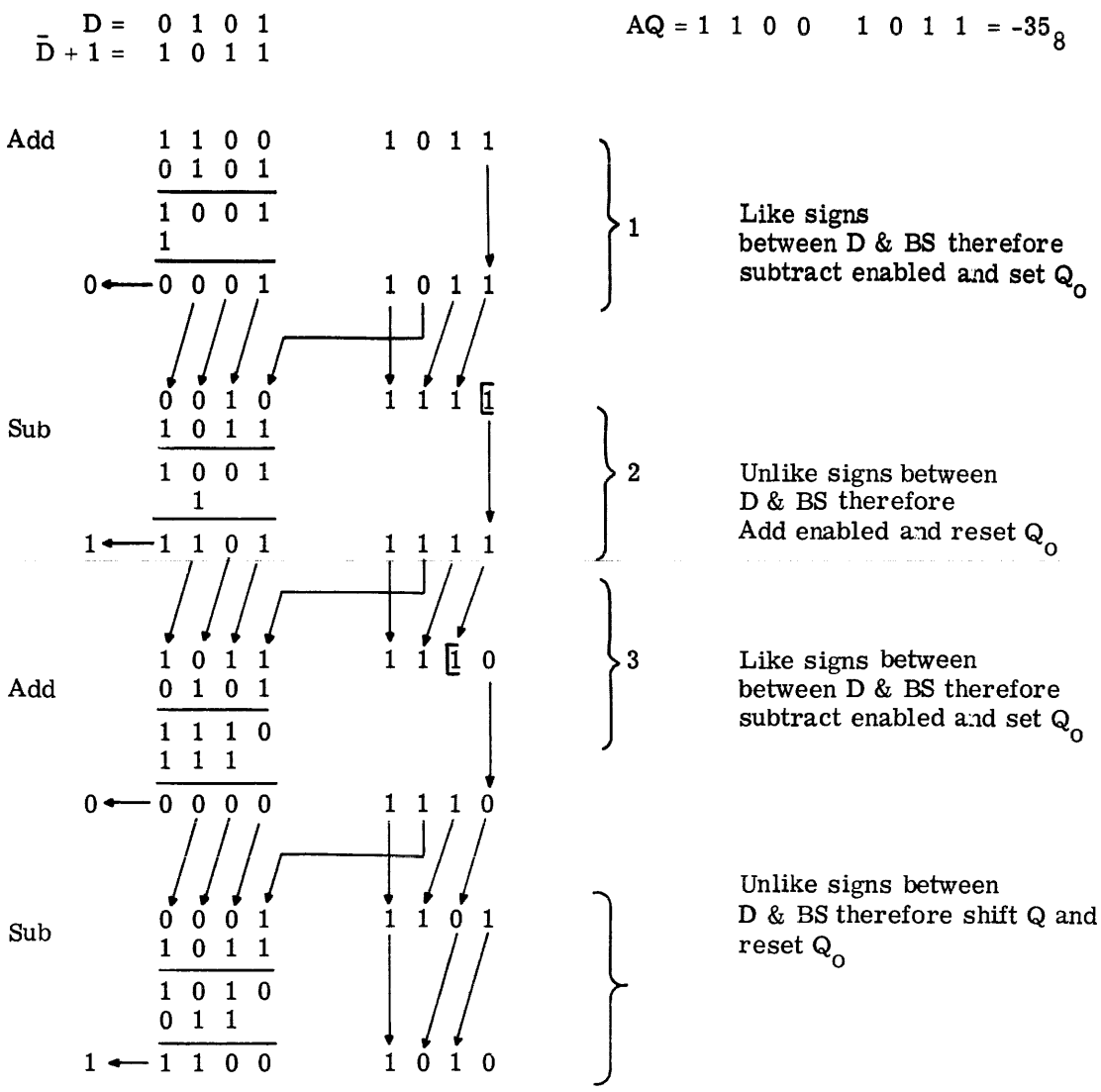
Refer to Figure 4-4-AA, sheet 2. AC56 also partially enables a gate at Figure 9A1-7C which enters the arithmetic section into phase N when BALE (Figure 9A27-7C) is a logical "1" (N=2 or 3). During phase M, the 23 quotient bits are all formed and then phase N is entered. Phase N ($\overline{\text{AC64}}$) (Figure 9A2-4C) checks the remainder to determine if correction is required. These remainder conditions, shown on Figure 4-4-AA, sheet 2, are determined by the logic shown on Figures 9A4, 5, 6 and 7.

Remainder correction (complementation) is required following phase M if a negative quotient is being developed and, the divisor is positive (AQ23=1 and AD23=0) and the remainder (BS23) is positive (Figure 4-4-AB); or, if a positive quotient is being developed and a negative remainder appears with the divisor positive.

In the event the divisor is negative, when a negative remainder is formed with a positive quotient, the remainder is correct only if the A-register (remainder) is equal to zero. See Figure 4-4-AC for this case; four bit registers are used for this example.

The conditions for which the quotient must be corrected are checked during phase Q. Refer to Figure 4-4-AA, sheet 4 for the conditions that are examined to determine if Q required correction. The correction to be applied to the quotient is the addition of 1 to the magnitude of the quotient, i. e. , a negative 1 is added to a negative quotient and a positive 1 is added to a positive quotient.

The remaining phases of the divide operation are entered into as determined by logic on Figure 9A1 based upon conditions shown in Figure 4-4-AA, sheet 4.



Remainder correction not necessary

1 1 0 0

Remainder = - 4

Quotient correction (Q + 1)

Q = 1 0 1 0
0 0 0 1
1 0 1 1

Q = 1 0 1 1 = -5_8

Figure 4-4-AB. Negative Quotient with Remainder (Pos Division)

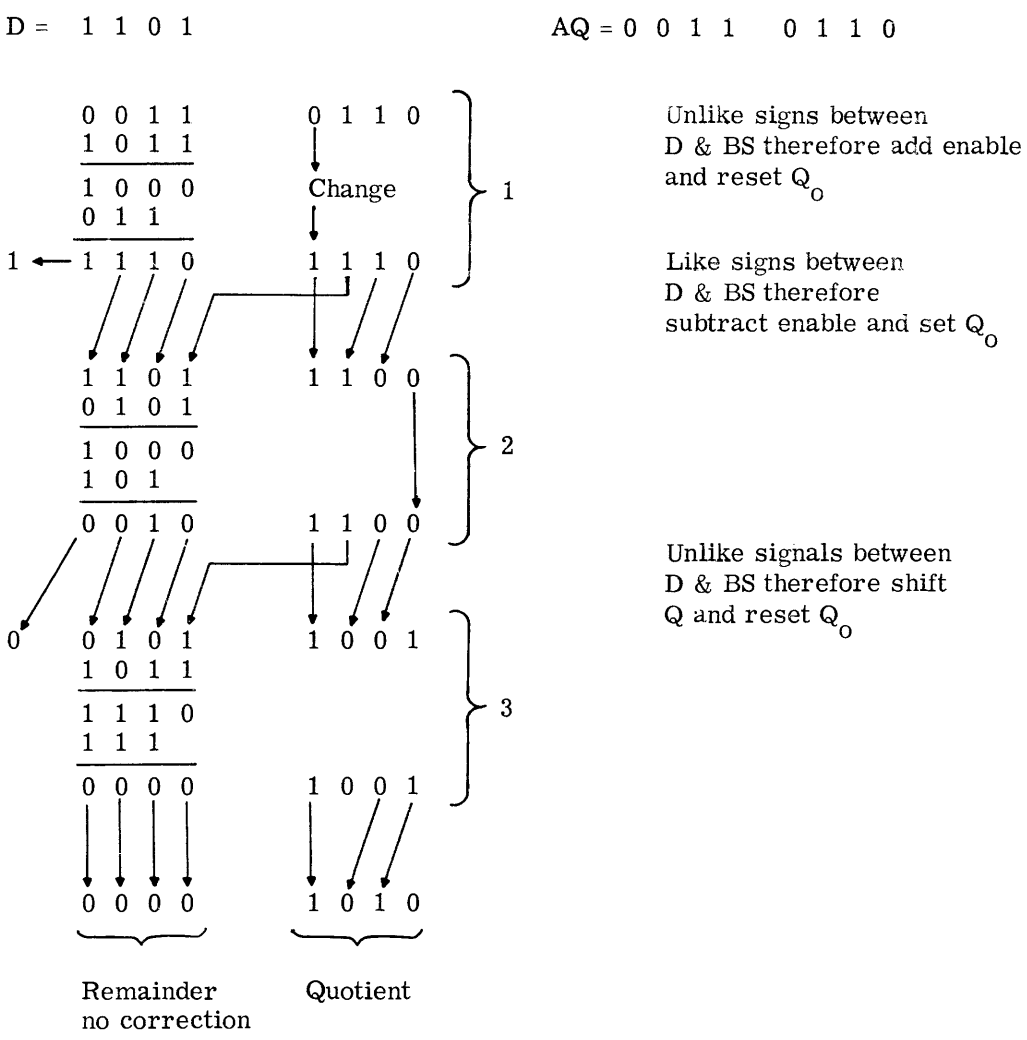


Figure 4-4-AC. Quotient Negative

4-5. Input-Output Section Operation.

a. General Input/Output Philosophy. - Modern high speed computers operate many times faster than most peripheral equipments, i. e., typewriters, Magnetic Tape Units, Paper Tape equipments. Because of this vast speed differential, it is necessary to provide a method of timing compensation, which usually takes the form of buffering registers. These registers, hold the information until it can be accepted by the receiving equipment. Normally, both the computer and the peripheral equipments each contain at least one buffer register.

The resulting timing incompatibility dictates that a strict procedure be adhered to during the communication process. This precedure is governed by a set of control lines which accompany the data or codes on both output and input channels.

(1) Output Channels. - Figure 4-5-A is a block diagram of a computer with a typical output channel which utilizes parallel data transfer methods.

There are four control lines and data lines on the output channel. The functional operation of the control lines is as follows:

OUTPUT DATA REQUEST (ODR) - The output data request always originates in the peripheral equipment and is sent to the computer when the the peripheral equipment is ready to receive DATA from the computer.

OUTPUT ACKNOWLEDGE (OUTACK) - The output acknowledge signal originates in the computer and is used to inform the peripheral equipment that there is DATA on the data lines.

EXTERNAL FUNCTION (EXFCT) - The externnal function signal originates in the computer and indicates to the peripheral equipment that there is an EXTERNAL FUNCTION CODE on the data lines. The external function is a command to the peripheral equipment instructing it to set up for a particular operation; i. e., master clear, buffer register, turn power off, ect. Each function is a unique code which is interpreted by the peripheral device. The external function signal is generated as the result of an EXF instruction being executed by the computer.

OUTPUT INTERRUPT (OUTINT) - The output interrupt originates in the peripheral equipment and is interpreted, by the computer that a predetermined condition in the peripheral equipment exists. For example, a power failure of a peripheral device; in such case output data is sent to a different channel or some other remedial action is initiated as determined by the program.

When a function word is transmitted from the computer, it is accompanied by a signal on the External Function line. The External Function signal being present instead of the output acknowledge identifies the word transmission as a function word. When a transmission is performed, the presence of the output acknowledge and the absense of the External Function signal signifys to the peripheral equipment that the word on the lines is data and not a function code. In other words, each data word is accompanied by an Output Acknowledge signal from the computer which identifies the word transmission as a data word and gates the data bits into the peripheral buffer register. There must be either an External Function signal or an Output Acknowledge signal accompanying each transmission from the computer.

An additional important difference exists with regard to these two signals. The Output Acknowledge signal is the computers' response to an Output Data Request (ODR). Therefore no OUTACK can be sent unless a prior ODR had been received. On the other hand, the External Function signal is

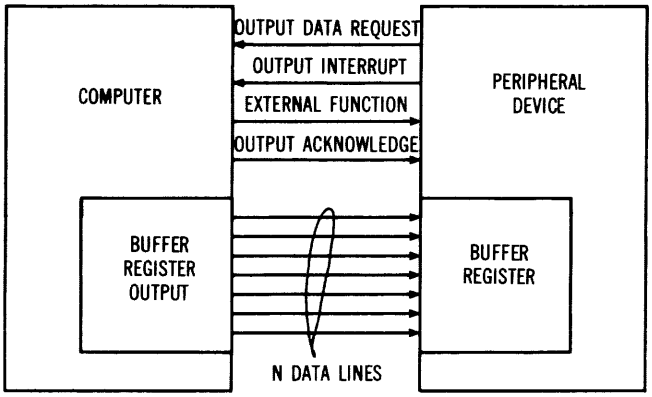


Figure 4-5-A. Output Channel Block Diagram

a command from the computer to the peripheral equipment and required no previous action by the peripheral device.

Data may be transferred in or out of the computer in two basic modes: NORMAL and BLOCK BUFFER. The procedure for transmitting data from the computer to peripheral equipment (OUTPUT CASE) in the NORMAL mode is as follows. It is assumed that the peripheral equipment is a high speed punch and that it is not, at present, set up to receive information, i. e. , no ODR signal being sent to the computer. An ODR must be present before data can be transmitted from computer to peripheral equipment.

1st. - The computer initiates an Output Buffer mode by executing one of the Output Buffer instructions, which specifies a given channel and the location in computer memory of the specified number of output words.

2nd. - The computer executes the EXTERNAL FUNCTION INSTRUCTION and places the External Function code on the data lines of the output cable.

3rd.- The computer sets the External Function line.

4th. - The Paper Tape Equipment detects the active External Function line.

5th. - The Paper Tape Equipment samples the lines of the output cable and loads the code into its buffer register.

6th. - The computer drops the External Function line and the External Function code on the data lines.

7th. - The punch control circuit sets the Output Data Request line to indicate that it is in a condition to accept data.

8th. - The computer detects the Output Data Request signal at its own convenience.

9th. - The computer places information on the lines of the output cable.

10th. - The computer sets the Output Acknowledge line to indicate that the data lines are ready for sampling.

11th. - The punch control circuit detects the Output Acknowledge signal.

12th. - The punch control circuits drops the Output Data Request signal on receipt of the Acknowledge signal.

13th. - The high-speed punch samples the lines of the output cable.

14th. - The computer drops the Output Acknowledge signal and the data lines.

15th. - The high speed punch makes a cycle and punches the data in its buffer register onto the tape. When the punch is ready to receive the next data word, it raises the ODR line signal and steps 7 through 15 above are repeated.

The cycles continue until the number of words specified by the buffer instruction are transferred at which time the computer terminates the sending of data and the OUTPUT ACK signals.

From the foregoing, it is observed that the data transfer rate is governed by the speed at which the peripheral equipment can accept the data. This is true unless the peripheral equipment can accept data faster than it can be supplied by the computer, which is rarely the case in practice.

(a) Output Channel Timing (NORMAL MODE).- In the transmission of data between the computer and peripheral equipment, time must be allowed for the data lines to stabilize before being sampled. An inherent time delay is added by the computer between the loading of the output register and the energizing of the Acknowledge signal. An external input device places the data word on the input lines immediately upon activating the Input Data Request signal. Because of the variation in signal rise times and adverse tolerance buildup, it is not possible for the computer to reach recognition state before the data lines have settled from a signal standpoint. Thus, the timing requirement is made to protect against faulty transmission during signal transmission during signal transition time.

1. Output Data Timing. - The peripheral equipment must set the Output Data Request line, indicating that the device is in a condition to accept data from the computer. This is necessary because data is available to the peripheral buffer register for a fixed period of several microseconds and the buffer input circuits must be in a position to accept the data in the time allowed. The time which may elapse between the request and the data being placed on the output lines is not fixed, but may vary, depending on the computer program, the priority of the particular channel, and the data rates of other peripheral equipment in communication with the computer. The computer sets the Output Acknowledge line after placing the data word in the Output buffer register. The Output Acknowledge signal is present for a fixed period of time and then is automatically dropped by the computer. Thus, the peripheral input circuits must sample the data

lines within this time period. Many peripheral equipments must be capable of detecting and Output Acknowledge which may exist in a stable "1" state for as little as 0.5 microsecond, allowing for a maximum permissible rise time of 0.5 microsecond. The Output Data Request signal is dropped after the punch control circuit senses the Output Acknowledge. Output timing relationships are shown in Figure 4-5-B.

2. External Function Timing. - The computer places the External Function code on the data lines of the output cable and sets the External Function signal. The External Function line is set in the DPS-2402 computer for a period of about two microseconds; then the signal is dropped, as is the data on the lines. As in the case of input and output transmissions, the paper tape equipment must be capable of detecting, as an External Function, a signal which may exist in the stable "1" state for this period. Figure 4-5-C illustrates these timing relationships.

(2) Input Channels. - Figure 4-5-D shows a block diagram of a typical input channel. Three control lines and n data lines are used to transmit, by parallel transfer, data from the peripheral equipment to the computer. The functional assignment of the control lines are as follows:

INPUT DATA REQUEST (IDR) - The input data request always originates in the peripheral equipment. It indicates that input data is available on the n data lines for input to the computer.

INPUT ACKNOWLEDGE (INACK) - The input acknowledge is sent to the peripheral equipment by the computer and indicates that the computer has sampled the input data on the n data lines. The peripheral equipment normally uses this signal to drop the IDR and data.

INPUT INTERRUPT (ININT) - The Input Interrupt is a signal sent by the peripheral equipment to the computer which is accompanied by a unique code on the data lines and is used to inform the computer of one of many possible conditions existing in the peripheral equipment. The computer samples the interrupt code on the data lines and sends an Input Acknowledge to the peripheral equipment. Upon receipt of the INACK, the peripheral equipment normally drops the interrupt code and the interrupt control signal. The computer analyzes the code by means of a programmed routine to determine the reason for the interrupt (power failure,

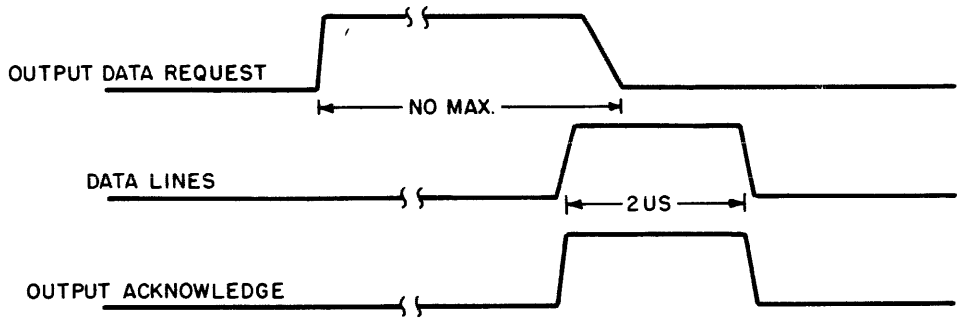


Figure 4-5-B. Output Timing (Data)

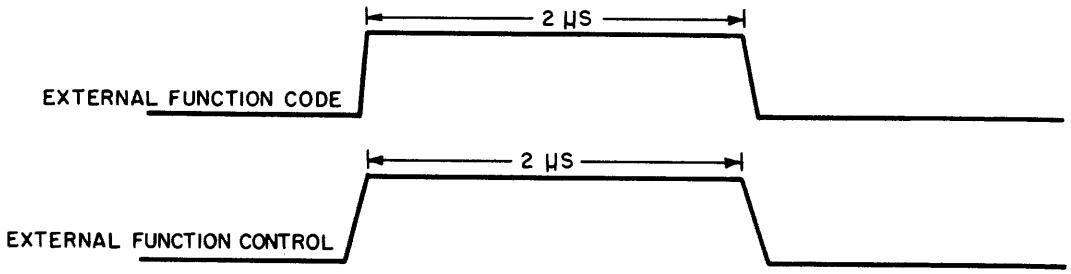


Figure 4-5-C. External Function Timing

message complete, etc.) and take appropriate action.

The following sequence serves to illustrate the operation of the control signals on the input channel in the NORMAL mode. For purposes of this example, assume that the peripheral device is a photoelectric reader. Assume the reader is, at present, not set up to input data (no IDR signal present) and an external function is required to start reading a punched tape.

1st. - The computer initiates an Input Buffer mode, by executing one of the input buffer instructions for a given channel and a given number of word transmissions.

2nd. - The computer transmits an external function code, as a result of the External Function (46) instruction, to select the reader as an input device.

3rd. - The computer sets the EXTERNAL FUNCTION control line.

4th. - The photoelectric reader logic detects the external function signal.

5th. - The reader logic gates the EXFCT code into the buffer register.

6th. - The computer drops the EXTERNAL FUNCTION control signal and the EXTERNAL FUNCTION code.

7th. - The photoelectric reader reads the first frame of tape and places a data word on the lines of the input cable.

8th. - The reader control circuit sets the Input Data Request line to indicate that a data word is ready for transmission.

9th. - The computer detects the Input Data Request signal.

10th. - The computer samples the 24 data lines at its own convenience.

11th. - The computer sets the Input Acknowledge line to indicate that the data lines have been sampled.

12th. - The reader control circuits detect the Input Acknowledge signal.

13th. - The reader control circuit drops the Input Data Request line and removes the data from the 24 lines of the input cable.

Steps 7 through 13 are repeated for each word transferred. Again, data transfer proceeds at a rate governed by the peripheral equipment up to peripheral speeds greater than computer speed. Because most peripheral equipment is very slow with respect to computer speed, the input and output function must run independently (except for sharing memory) of the control section. The execution of instructions proceeds at near normal high speed rates and the input output buffer proceeds at a rate governed by the peripheral equipment.

(a) Input Channel Timing

1. Input Data Timing. - The Input Data Request signal indicates to the computer that a data word has been placed on the lines of the input cable by the photoelectric reader. The data lines must remain stable as long as the Input Data Request signal is up, and the request signal must be maintained until the computer indicates its acceptance of the data word by an Input Acknowledge signal. Since the computer samples the 24 data lines at its own convenience, the time during which the request signal and the data lines must remain active is not fixed. The Input Acknowledge signal indicates to the reader control circuits that the input data lines have been sampled. The Input

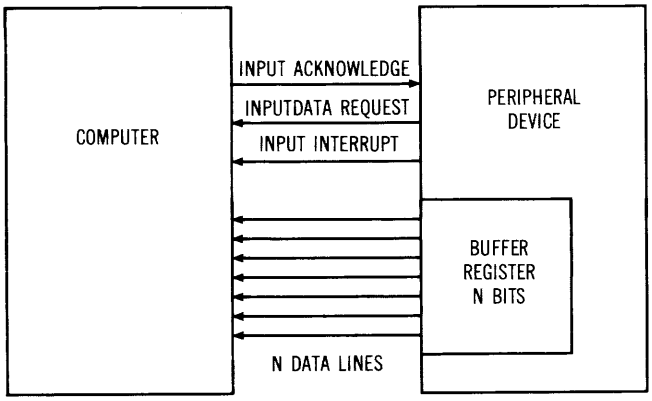


Figure 4-5-D. Input Channel Block Diagram

Acknowledge signal is set for a fixed period of about two microseconds. The reader control circuits must be capable of detecting, as an Input Acknowledge, a signal which may exist in a stable "1" state for as little as 2 microseconds. On sensing the Input Acknowledge, the Input Data Request signal is dropped to the "0" state. Input timing relationships are shown in Figure 4-5-E.

2. Input Interrupt Timing. - Input Interrupt Timing is identical to the input data timing shown in Figure 4-5-E with the exception that the control line utilized is the Input Interrupt line instead of the IDR line and the informations on the data lines is the interrupt code in place of input data.

(3) Block Buffer Mode. - When the BLOCK BUFFER mode is called for by the buffer instruction, a single ODR from the peripheral equipment transfers the complete block of memory locations at the maximum data transfer rate. The two-microsecond Acknowledge signal accompanies each transmission during the entire block transfer operation. The data transfer ceases only after the number of words specified by the buffer control word has been transferred. This mode may be used when the peripheral equipment is capable of receiving or transmitting data at the computer rate (one word each 4 μs).

The control section ceases all operations during the BLOCK BUFFER mode and, therefore, no instructions are executed while this mode is in operation. Likewise, no interrupts can be honored during this mode.

(4) Summary. - The basic signals required for the operation of peripheral equipment, exclusive of data transmissions, are : Interrupt request, Input Data request, Output Data request, Input Acknowledge, Output Acknowledge, and External Function signals. The request signals are always generated by the peripheral equipment and are used to inform the computer that data transmission is desired. These action "requests" are DC levels which must remain in a stable state until honored by the computer. The Acknowledge signals and the External Function signals are commands to the peripheral equipment generated by the computer and are present for a limited period of time. The input circuits of the peripheral equipment must be capable of recognizing these signals within the established time limits and take appropriate action.

b. Input-Output Operation in the DPS-2402 Computer.

(1) General. - Figure 4-5-F shows a block diagram of the DPS-2402 Computer Input-Output Section.

The computer has 5 input and 5 output channels for interequipment communication. Each channel has associated with it two cables through which the interchange of information between computer and peripheral equipment takes place. Each cable carries 24 data lines and four control lines (see Figure 4-5-G).

The input connector consists of 24 incoming data lines, an input request line, an input acknowledge line, and an interrupt line. The output connector consists of 24 outgoing data lines, an output request line, an output acknowledge line, an external function line, and an interrupt line.

The input request state is set by a continuous signal on the input request line supplied by a peripheral device. This state can be tested by a conditional skip instruction, input data available, IDA. If an input buffer is established the input request is serviced and acknowledged in accordance to its priority. The request line must be dropped for a minimum of 1 microsecond in order to establish a new request state.

The output request state is set by a continuous signal on the output request line supplied by a peripheral device. This state can be tested by a conditional skip instruction, output data request, ODA. If an output buffer is established the

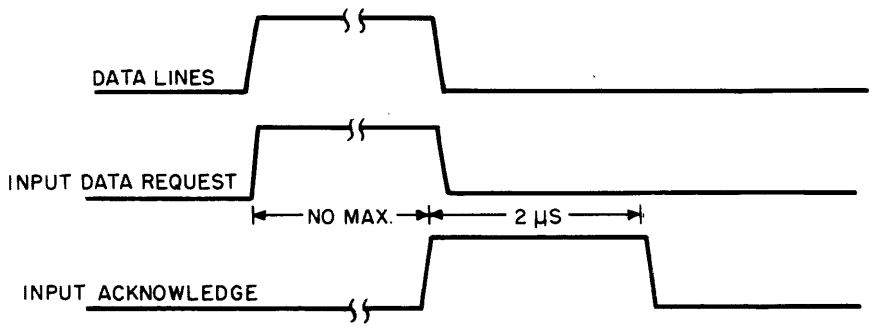


Figure 4-5-E. Input Timing

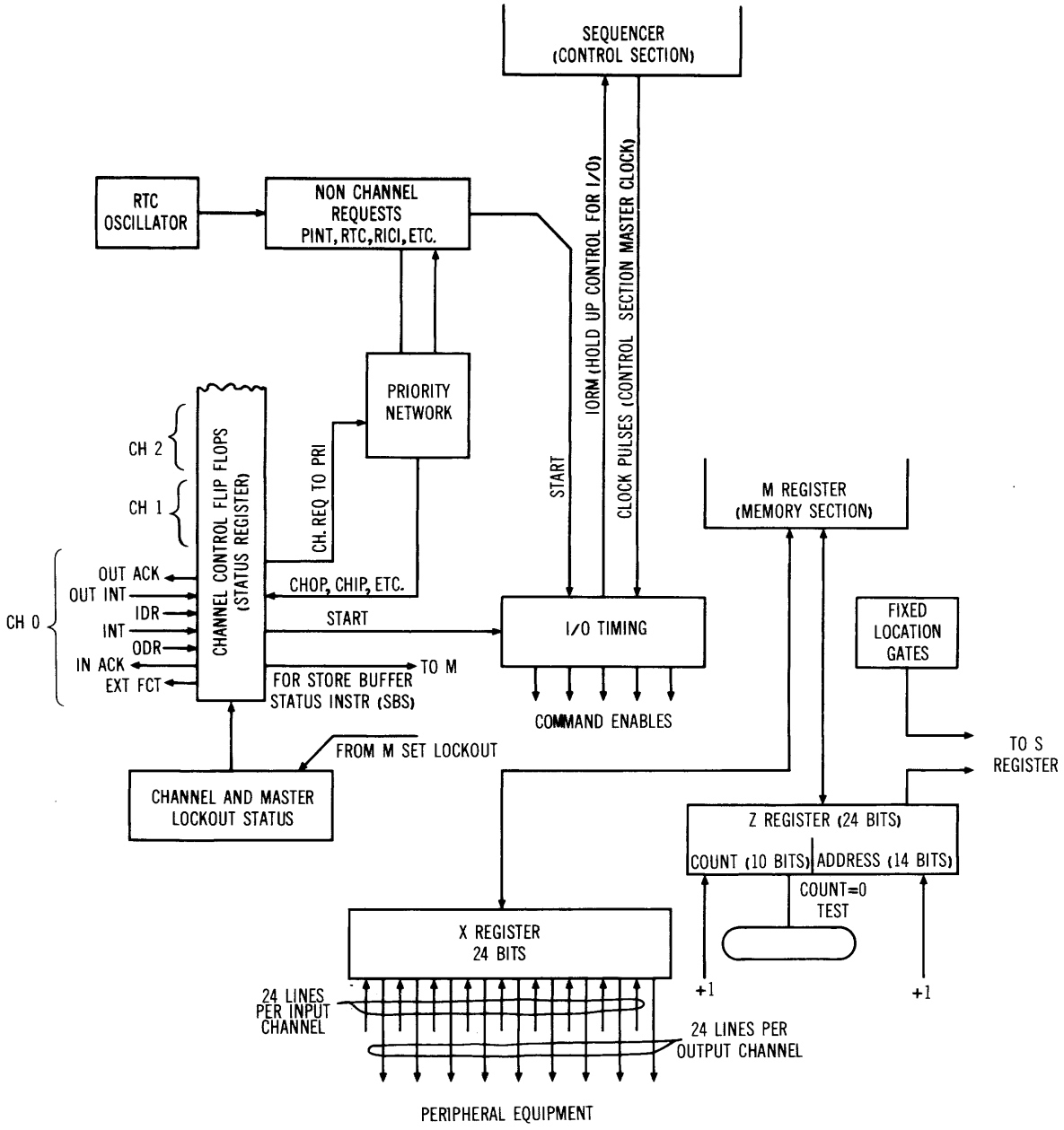


Figure 4-5-F. Input/Output Block Diagram

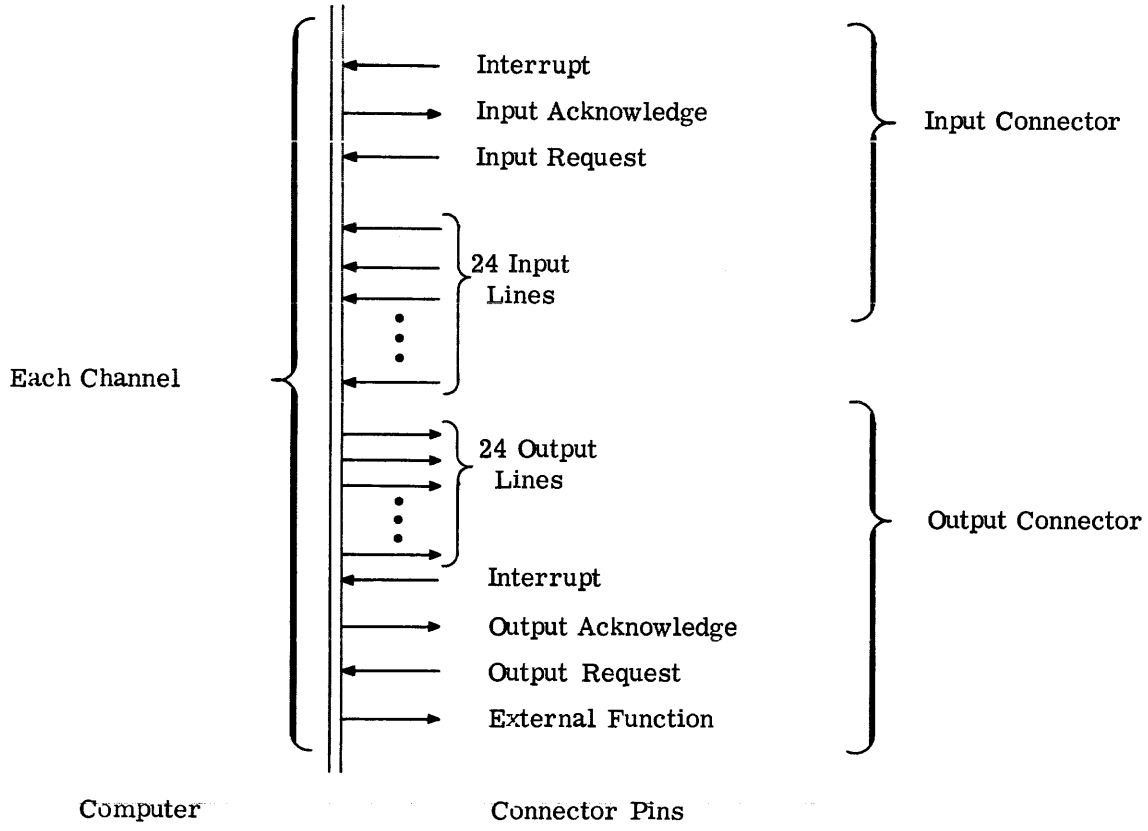


Figure 4-5-G. Channel Interface

servicing is as described for input request.

The input external interrupt state is set by a continuous signal on the input interrupt line supplied by the peripheral device. When the interrupt is serviced the 24-bit status word is stored in an address specified by the contents of fixed locations 00140 to 00157. According to Channel 0 the program is then forced to execute one of the instructions at locations 00160 through 00177. The interrupt state cannot be set again until the request line has dropped for a minimum of 1 microsecond.

The output external interrupt state is set by a continuous signal on the output interrupt line supplied by the peripheral device. When the interrupt is serviced the interrupt is acknowledge and the program is forced to execute one of the instructions at locations 00200 through 00217 according to the specified channel.

If the external interrupts on any given channel have been locked out, and it is desirable to clear the interrupt states without servicing them, a CIR instruction may be executed.

A channel is set to the input buffer active state by the execution of either establish buffer input mode instruction, BIN or BINI. This state is cleared when the buffer control word count is exhausted, and an input internal interrupt state is set if previously specified by BINI. This internal interrupt forces the program to execute one of the instructions at locations 00100 through 00117 according to the specified channel. The input buffer active state can also be cleared with or without an interrupt by the execution of the proper terminate buffer input mode instruction, BIMT or BIMTI.

The output buffer active state is identical to the input buffer state except the interrupt locations are 0120 through 137, and the instructions involved are BOT, BOTI, BOMT, and BOMTI.

Both the input and output buffer active states may exist on any channel simultaneously.

An interrupt lockout state is established, modified or stored by execution of the instructions: enter lockout status (ELS), set additional lockouts (SAL), and store lockout status (SLS).

(a) Acknowledge Signals. - An input acknowledge signal is generated when an external interrupt is processed, when input data is stored by an express input instruction, and when input

data is transmitted to core by a channel in input buffer mode.

An output acknowledge signal is generated when output data is placed on the output lines by either express or buffered output.

(b) External Function Signals. - An external function, EXF, instruction is provided which places a signal on the external function line of a channel and places a 24-bit word from memory on the channel output lines. Interpretation of this word varies with the type of peripheral device. For a channel with many peripheral devices, part of the word would be interpreted by a multiplexer, appropriate routing paths established, and the remainder of the word used to establish the desired state of a particular device.

(c) Express Mode. - Upon execution of an express output instruction, EXO, a word from memory is sent to the designated channel. The output acknowledge line is signalled concurrent with data for 1 microsecond. The peripheral device must be ready to accept the output in that 1-microsecond period. If there is a possibility the channel is not ready to accept the output, the output request state may be tested by an output data available, ODA, instruction.

Upon execution of an express input instruction, EXI, a word from the designated channel is stored in memory. The input acknowledge line is signalled for 1 microsecond immediately after data transfer has been completed. The channel is assumed ready to supply the input data. If there is a possibility the channel is not ready, the input request state may be tested by an input data available, IDA, instruction.

(d) Buffer Mode. - Information is transferred in a 24-bit parallel mode. The 24-bit input word is gated into the computer through a group of input amplifiers and fed directly into the X-register. The computer output word is formed in the X-register and is placed on the lines of the active output cable by a group of line drivers. The X-register thus serves as a timing (buffer) register between the computer and the peripheral equipments.

Data may be transferred into or out of the computer in either high (BLOCK BUFFER) or low speed (NORMAL) modes. When the high speed mode is called for by the CBM instruction,

The entire block of data is transferred requiring only <u>one</u> data request from the peripheral equipment.

the control section is locked out by IORN (Figure 9I1-8A) until the buffer terminates.

On the other hand, the normal mode accomplishes one word transfers. Once an input buffer is established, the occurrence of an input request from a peripheral device causes the following sequence of events: the input request state is set; when processing of the current instruction and higher priority input-output is complete, a memory access is made to fetch the buffer control word; one (1) is added to the count, and 1 is added to the address of the buffer control word; the result is stored back in the same buffer control word location; and the output acknowledge signal is sent concurrently with the output for 1 microsecond.

Upon incrementing either the input or output buffer control word count from all 1's to all 0's, the internal interrupt state is set and an instruction from the prescribed fixed location is executed providing the buffer was established to interrupt on buffer termination.

The time sequences for signals and data flow are shown in Figures 4-5-H and 4-5-I for the case where a sequence of ADD instructions are being executed by a program and only one channel is being used.

(e) Hardware Interrupt and Buffer Priorities. - Interrupt and request lines are sampled only at the completion of an instruction. Since it is possible for two or more interrupts or request signals to arrive during the execution of a single instruction, hardware priorities are necessary to allow only one interrupt or request to be serviced at a time. All interrupts released from master lockout or channel lockouts are considered to have arrived simultaneously and are also subject to these priorities; it should be noted that instructions are available to lock out an interrupt signal so that the order of processing interrupt subroutines need not be the same order as for hardware priorities. Following an interrupt or request being processed or serviced, one normal instruction is executed before the next sampling. This feature prevents a runaway peripheral device from locking out an I/O control program. Also, no sampling takes place immediately after a JMP CML or JSR SML instruction.

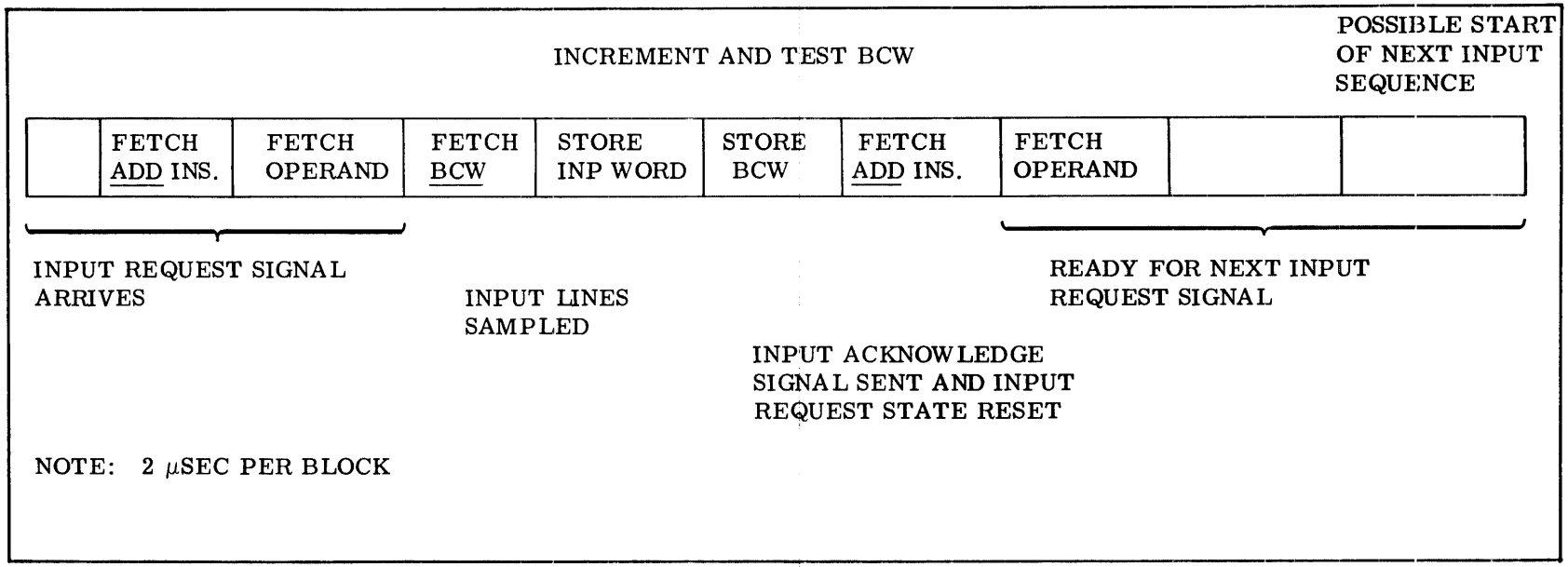


Figure 4-5-H. Buffer Input Timing and Signals

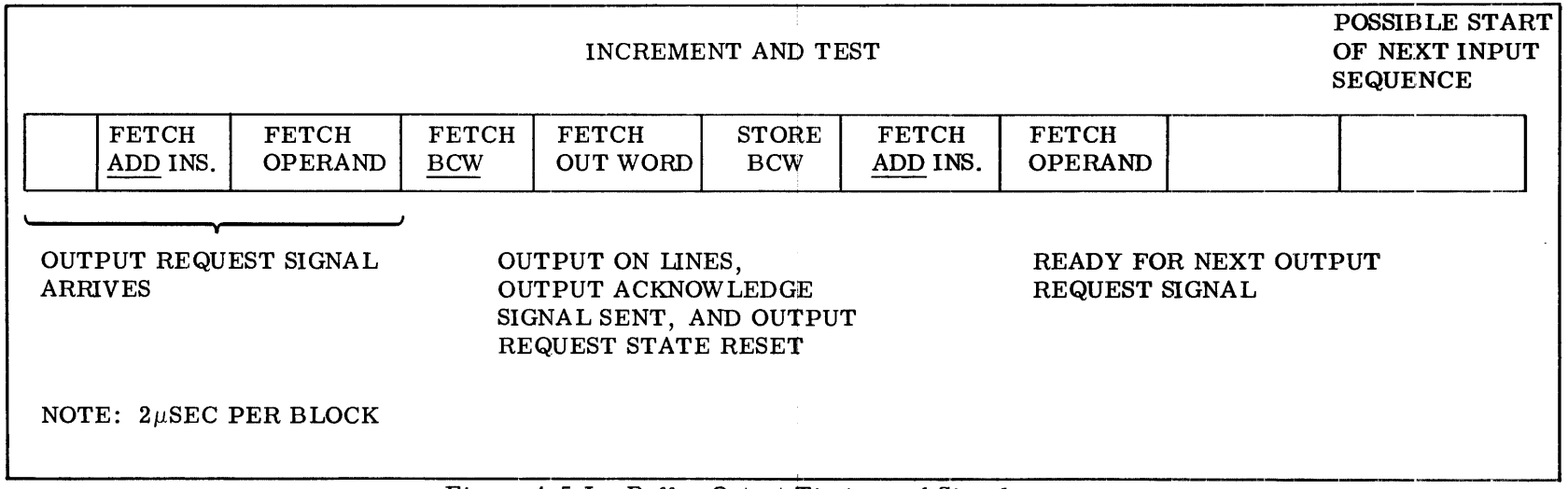


Figure 4-5-I. Buffer Output Timing and Signals

Table of Priorities for Interrupts and Input-Output Functions

Priority Order	Description
1	Program Parity Error Interrupt
2	Input-Output Parity Error Interrupt
3	Fault Interrupt
4	Power Failure Interrupt
5	Real Time Clock Accumulation
6	Real Time Clock = 2^{24}-1 Interrupt
7	Manual Interrupt
8	Channel 0 - Input External Interrupt - Output External Interrupt - Input Buffer - Output Buffer - Input Internal Interrupt (Buffer Complete) - Output Internal Interrupt (Buffer Complete)
9	Channel 1 - Input External Interrupt - Output External Interrupt - Input Buffer - Output Buffer - Input Internal Interrupt (Buffer Complete) - Output Internal Interrupt (Buffer Complete)
10	Channel 2 - Input External Interrupt - Output External Interrupt - Input Buffer - Output Buffer - Input Internal Interrupt (Buffer Complete) - Output Internal Interrupt (Buffer Complete)
11	Channel 3 - Input External Interrupt - Output External Interrupt - Input Buffer - Output Buffer - Input Internal Interrupt (Buffer Complete) - Output Internal Interrupt (Buffer Complete)
12	Channel 4 - Input External Interrupt - Output External Interrupt - Input Buffer - Output Buffer - Input Internal Interrupt (Buffer Complete) - Output Internal Interrupt (Buffer Complete)
13	Channel 5 - Input External Interrupt - Output External Interrupt - Input Buffer - Output Buffer - Input Internal Interrupt (Buffer Complete) - Output Internal Interrupt (Buffer Complete)
14	Channel 6 - Input External Interrupt - Output External Interrupt - Input Buffer - Output Buffer - Input Internal Interrupt (Buffer Complete) - Output Internal Interrupt (Buffer Complete)
15	Channel 7 - Input External Interrupt - Output External Interrupt - Input Buffer - Output Buffer - Input Internal Interrupt (Buffer Complete) - Output Internal Interrupt (Buffer Complete)

(f) Interrupt Lockouts. - If an interrupt lockout is established and an interrupt signal arrives, the interrupt signal is ignored. Only when the interrupt lockout is removed can the interrupt be processed according to the priority specified in Hardware Interrupt priorities.

1. Channel Interrupt Lockouts. - Each channel has an interrupt lockout which, when set, prevents processing of input and output buffer finished interrupts and external interrupts. The lockout status on all channels can be set by the enter lockout status instruction, ELS. If some channels are already locked out, a set additional lockouts, (SAL) instruction will place more channels in lockout status without affecting those already locked out. The status of the lockouts on all channels may be determined by storing the lockout status, SLS, instruction and examining the lockout bits.

If an interrupt has been locked out and no processing of the interrupt is desired, the clear interrupt request, CIR, instruction can be used.

2. Master Interrupt Lockout. - When the master interrupt lockout is activated, all external and internal channel interrupts are prevented while all other interrupts are allowed. The real time clock is still incremented and buffer the input and output continues.

The master lockout is set by execution of the "jump and set return set master lockout" instruction (JSR SML) is cleared by the "jump and set return clear master lockout" instruction. The master lockout is used to prevent other channel interrupts from being honored before the execution of the current subroutine is completed. By ending the subroutine with a jump and clear master lockout, interrupts may be honored.

(g) General Input-Output Timing Considerations. - Peripheral equipment should normally be designed to accept an external function, (i. e., control signal at any time), however, data transfers must be geared to the rate of the external device. Synchronization of data transfers is automatic when accomplished in the buffer mode; however, synchronization is a responsibility of the programmer when data transfers are made under program control. To assist the programmer in this respect, two instructions (IDA and ODA) are available.

Prior to executing input or output instructions or establishing a buffer mode, an external function code should be outputted to the peripheral device to set it in the proper state for data transfers.

External function code definition and other related information can be found in the reference manual describing the particular peripheral equipment involved.

When high bit-transfer rate input-output devices are being used, a number of hardware timing characteristics influence the coding of an input-output executive. The following tables give worst case figures on various characteristics.

The maximum input-output bit transfer rate is 6×10^6 bits per second. This may be obtained on input by a sequence of express input, EXI, instructions or by the Block buffer mode. One 24-bit data word is input per instruction executed. Two microseconds are required to store the data, and two microseconds are required for the acknowledge. Thus, every 4.0 microseconds 24 bits of data can be inputed. Similarly, a sequence of express output, EXO, or Block Buffer instructions can output a 24 bit word every 4.0 microseconds.

Operating under express input or output, the computer expects the peripheral equipment to be ready for the input or output and does not wait for request signals. This express function is useful where very high speed peripheral equipment is being used, and where blocks of data are relatively small, since one instruction per input-output word is needed. A slower form of express input-output uses the following sequence to bring in a block of N words into locations, L, L + 1, . . . , L + N-1.

```
ENB      = N, 1
EXI      L          4 microseconds
ADD 1    *-1        6 microseconds
JDB N    *-2, 1     6 microseconds
```

Express input-output does not allow the computer to wait for a peripheral device to request an input-output transfer.

Buffered input-output makes use of circuitry within the computer to increment the buffer address, to increment the count of the number of data words transferred, and to automatically terminate the buffer when the count is exhausted. The current buffer address and count are stored in core as a buffer control word.

(2) Registers and Control Logic

(a) X-Register and Transfer Logic. - The X-register, shown on Figures 9I8 and 9I9, is a 24-bit register composed of molecular (M215) J-K flip flops. It is cleared by the $\overline{\text{REXR}}$ signal generated by the X-register clear switch located on the maintenance panel to the right of the register indicators, or by the master clear logic (Figure 9I2-3A). The X-register is set manually by the indicator switches at the maintenance console or by DC signals from the transfer gates (Figure 9I2). The X-register does not use the AC features of the J-K flip flops but acts as a register of RS flip flops that are first cleared, then loaded in a DC manner by the transfer gates shown on Figure 9I5. The only input to the X-register is from the M-register. The transfer gates transfer the contents of M to the X-register on the command $\overline{\text{0MR3}}$ (Figure 9I2-7D) and CLP3. Note that X is cleared during CLP1 (Figure 9I2-B4).

The X-register transfers its contents to the M-register utilizing a set of gates shown on the logic schematics directly under the X-register flip flops. The command for this transfer (X to M) is TRXM (Figure 9I2-5A).

(b) Z-Register, Transfer and Incrementing Logic. - The Z-register is a 24 bit register reserved for incrementing the buffer control word (count and address portions) and the real time clock location. The Z-register communicates with the M-register and the S-register (lower 14 bits only) in the memory section. Like the X-register, the Z-register is effectively a RS flip flop register as far as transfer logic is concerned. Only the DC override signals are used during Z loading. The Z-register is first cleared $\overline{\text{REZR}}$ (Figure 9I2-3A) during CLP1 clock and loaded from M by means of the transfer gates command ENMZ during IOR1 (Figure 9I5-6D) and CLP3 clock.

Outputs from the Z-register to the M-register are effected by the gates shown under the Z-register flip flops. The command for this transfer (Z to M) is TRZM (Figure 9I2-5A) on clock CLE2.

The Z-register incrementing may be accomplished in two modes. The first of these is to increment the entire 24-bit Z-register by 1. This mode is used for updating the real time clock.

The second mode is incrementing by 1 the upper 10 bits of the Z-register and the lower 14 bits as two separate registers. This mode is used to increment the buffer control word count and address portions. The difference in incrementing mode is accomplished by a single gate on the carry path from bit-13 to bit-14 (Figure 9I7-7D) of the Z-register.

The command INCZ (Figure 9I2-5A) causes the Z-register to increment by 1 in either of the two modes. RTCP, shown on Figure 9I-7D, results in incrementing the entire Z-register.

The INCZ pulse at CLE3 clock time clocks all Z-register flip flops for the incrementing process. Flip flop ZR00 changes state on the trailing edge of each INCZ pulse. Flip flop ZR01 changes state after two INCZ pulses, flip flop ZR02 after four, etc. (See Figure 4-5-J.)

The carry paths 022R, 042R, etc. propagate the carry to higher order stages of the Z-register. The carry path is broken between the 13th and 14th bits of Z unless a TRCP (real time clock processed) signal is at a logical "1". If this signal is not present, the incrementing process treats the Z-register upper ten bits as a separate register; thus ZR13 and ZR00 both change state on each INCZ pulse and each portion increments by one.

(c) Channel Status Register (Channel Control). - Each channel has a set of control flip flops (see Figures 9I11 through 9I13). These flip flops retain the channel status until the I/O section can take specific action. The output control flip flops; OODR, OOIR, OBEO and OIEO are shown on the right hand portion of the channel control schematics:

OODR - Sets when the peripheral equipment on the channel sends an output data request and the previous ODR signal has been dropped, i.e., the flip flop just above OODR must be in the clear state when the ODR arrives on the output channel control line. Cleared after the request is processed.

OOIR - Sets when the peripheral equipment sends an output Interrupt Request OIR on the control line and the previous OIR has been dropped by the peripheral equipment. Cleared after the interrupt is processed.

OBE0 - Sets when an output buffer instruction is executed for the given channel. Cleared when the buffer is terminated.

OIE0 - Sets when an output buffer with interrupt (monitor) is executed on the given channel. Cleared after the monitor interrupt is processed unless the buffer is terminated prematurely by an instruction, at which time, it is cleared without processing an interrupt.

The channel control flip flops associated with the input function are located on the left half of the channel control schematics. These are OIDR, OIIR, IBE0 and IIE0 and serve the following functions:

OIDR - Sets when the peripheral equipment sends an input Data Request (IDR) to the computer and the previous IDR has been dropped by the peripheral equipment. Cleared when the request is processed.

OIIR - Sets when the peripheral equipment on the input channel sends an Input Interrupt Request (ININT) to the computer and the previous Input Interrupt signal has been dropped by the peripheral equipment. Cleared when the request is processed.

IBE0 - Sets when the computer executes an input buffer instruction on the given channel. Cleared when the buffer is terminated.

IIE0 - Sets when the computer executes an input buffer instruction with interrupt (monitor) for the given channel. Cleared after the monitor interrupt is processed unless the buffer is terminated prematurely by an instruction. In this case the IIE0 flip flop is cleared directly without processing an interrupt.

(d) Priority Logic. - The priority logic network is shown in Figures 9I3 and 9I4. Requests entering the priority logic are of two types: non-channel and channel. Non-channel requests are

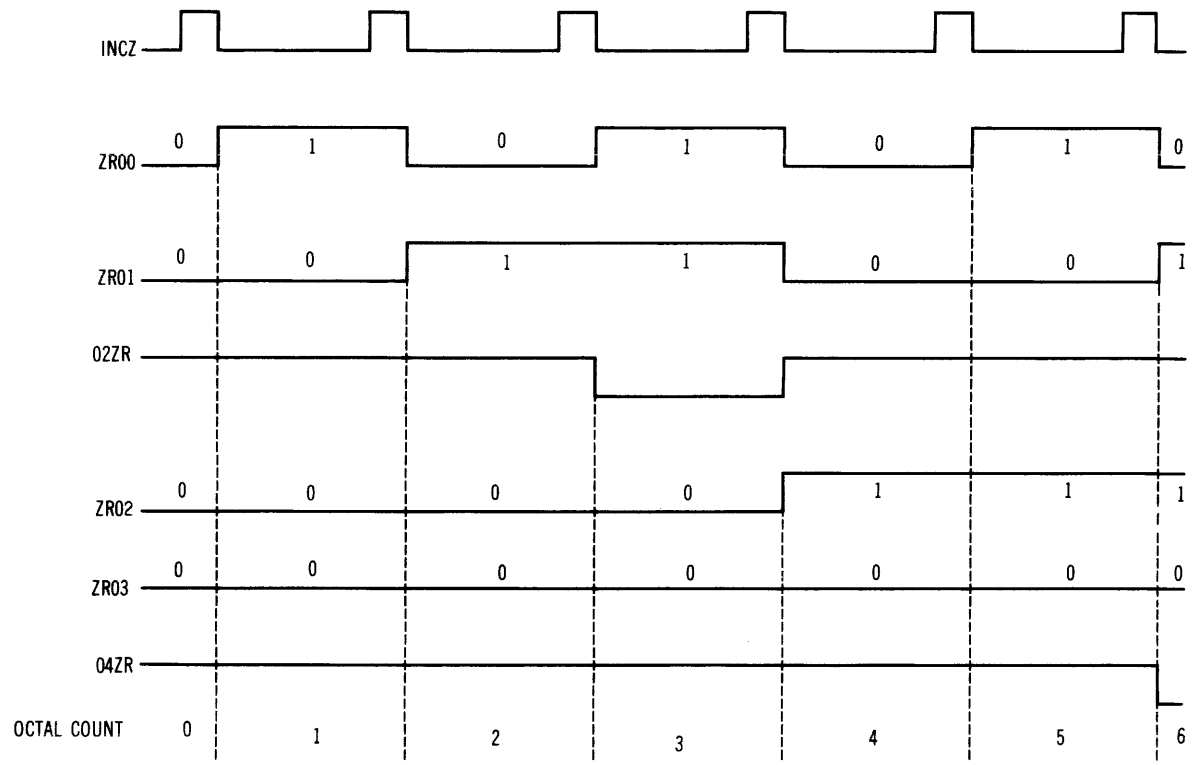


Figure 4-5-J. Z-Register Incrementing from Zero to Six

those which do not have a channel associated with them (Real Time Clock, Power Failure, Manual). Non-channel interrupts enjoy the highest priority. The order of priority of non-channel interrupts is:

Accomplished by the Control Section	Program Parity Error Interrupt Input/Output Parity Error Interrupt Fault Interrupt
Accomplished by the I/O Section	Power Failure Interrupt Real Time Clock Update Real Time Clock Overflow Manual Interrupt

Only those interrupt requests considered by the I/O section are discussed here. Refer to Paragraph 4.3 for a discussion of the control section interrupt types.

After non-channel interrupt requests are considered, the next level of priority is by channel number. Channel 0 has the highest priority and so forth. And finally, an order of priority exists within each channel for request types. This order of priority from highest to lowest, is:

Input External Interrupt (INPINT)
Output External Interrupt (OUTINT)
Input Buffer
Output Buffer
Input Monitor Interrupt
Output Monitor Interrupt

To illustrate the priority operation, consider the following examples. If an Output Monitor Interrupt on channel 1 and an Input External Interrupt on channel 2 occurred simultaneously, the Output External Interrupt is processed first because channel priority is determined first and any request type on channel 1 takes precedence over any request type on channel 2. If an input buffer request and output buffer request occurred simultaneously on channel 0 the input buffer request takes precedence.

The priority operation may be summarized as follows: First, the priority logic checks to see if any non-channel interrupt is present. If so, which of the non-channel requests has the highest priority. If no non-channel request is present, the channels are checked to determine the highest channel requesting I/0 action. Once the highest requesting channel is found, a check is made of the types of requests on that particular channel to determine the highest request type.

Refer to Figure 914. The Power Failure Interrupt (PINT) is the highest non-channel I/O interrupt. This signal enables gate $\overline{\text{PINP}}$ (Figure 914-1B) if the I-register does not contain an EXF, EXI, EXD or CBM instruction ($\overline{\text{BLBM}}$ = logical "1"). The gate $\overline{\text{PINP}}$ outputs a logical "0" and thus disables all other request gates, namely $\overline{\text{RTCP}}$ (Real Time Clock) MINP (manual interrupt) and all channel request gates; CHP0, CHP1, CHP2, etc.

In like manner, the requests entering the logic shown on Figure 914 appear in order of their priority from right to left on the figure; each one disabling all request gates to the left of it. After one of the channels is selected, the CH (channel no.) P signal is used with that of the status register to allow the request types for that channel to enter the request type priority gates shown on Figure 914. Here, final selection is made of the one request that is processed during the next I/O $\overline{\text{operation}}$ (OBMS, IBMS, etc.).

(e) Timing and Control Logic. - Figure 9I1 shows the primary timing and control logic for the Input/Output Section. Three timing flip flops: IOR1, IOR3, and IOR4 are used to sequence the various I/O operations.

The IORM flip flop (Figure 9I1-8A) is set when a buffer request or RTC update request is honored and holds up the control section at the beginning of an IA00 sequence so that the I/O section may use memory. A signal from the IORM flip flop also starts the I/O timing sequence.

The PRIN flip flop is set when any interrupt is processed so that the control section can execute an interrupt IA00 sequence instead of a normal IA00 sequence. PRIN sets the BLKP flip flop (Figure 9C7) in the control section to accomplish this function. The series of flip flops located in zones 1 through 3 of Figure 9I1 retain the fact that a non-channel I/O interrupt has occurred and holds the request until it can be honored by priority.

(3) Execution of I/O-Type Instructions. - When an I/O type instruction is in the I register, all requests to the priority network are blocked by BLBM (Figure 914-1D). The instructions which are considered I/O instructions are:

EXF (f = 46)
CBM (f = 45)
EXI (f = 40)
EXO (f = 41)

Blocking all requests when the above instructions

are being executed is required because all of the above effect directly the Input/Output Section.

(a) EXF Instruction. - The external function (EXF) instruction in the I register generates $\overline{\text{P10C}}$ (Figure 9I1-4C) during IA00 and CL03 clock. $\overline{\text{P10C}}$ sets flip flops IOR1 and IORM to hold up the control section prior to the next IA00 sequence.

With IOR1 flip flop set a logical "1" is transferred from gate $\overline{\text{IDLE}}$ (Figure 9I1-3A). This signal is used to initiate a memory timing sequence by generating a READ command.

EXI0 (Figure 9I1-3C) is used by the control unit to set the AD10 flip flop (Gates I_L to S). After the READ, the external function code is in the M-register. With IOR1 set, this register content is gated to Z (Figure 9I2-5D). This operation does not contribute to the execution of the instruction but is not prevented, i. e. , a don't-care condition. The next CLE3 clock pulse causes IOR3 to set and IOR1 to clear. With IOR3 set, the TRMX signal is generated (Figure 9I2) which loads the external function code into the X-register.

The next synch (CLE3) pulse clears IOR3 and sets IOR4. The EXF control signal on the designated channel remains on the output cable for two microseconds and drops as IOR4 is reset. IORM is reset via IOBA (Figure 9I1-8C) on the CLO3 preceding the IOR4 reset, thus allowing the IA00 flip flop in the control section to set on this same synch pulse.

(b) Buffer Instructions. - Execution of a buffer instruction performs two operations :

(1) Store the Buffer Control word which follows the Buffer instruction in the special memory location reserved for that type of buffer

(2) Set the appropriate flip flops in the channel control logic for the channel specified.

The above operations are performed by the control section. When the CBM instruction is in the I-register, the control section enters an OA cycle with a P⟶S (the buffer control word address). The buffer control word is read into the M-register. The P-register is incremented and an MS cycle begins. The MS cycle begins with the special fixed location for the channel and buffer type being placed in S (see Figure 9I3). The succeeding write operation writes the BCW into the special address. In order to illustrate setting the buffer status flip flops consider an output buffer instruction on channel 0 with monitor. BLBM prevents a channel request from being honored. CH0P (channel 0 priority) is generated by gates shown on Figure 9I12 which decode the channel designator of the I-register. Bit 11 of the I-register is set indicating a buffer-type instruction, which generates SBUF (Figure 9I2) Signals SBUF and CH0P combine as shown on Figure 9I11-3C with the reset side of bit 13 of the I-register (indicating output mode select). The resulting logical "0" sets flip flop OBE0 (output buffer enabled, channel 0). Bit 12 of the I register generates SINT (Figure -2B) which combines with CH0P and $\overline{\text{IR13}}$ to set OIE0, (Figure 9I11-4C) the output monitor flip flop.

(4) Buffer Mode Processing. - When a buffer request is sent by the peripheral equipment, it is received by the channel control logic (Figure 9I11 for channel 0). Assume an ODR occurs on channel 0; If the IO section is not busy the signal IDLE is a logical "1" (Figure 9I1-3A). IDLE combines with the CLO1 clock (Figure 9I2 -1C) to generate REI0 (reset I0). On clock CLE1 REI0 generates SAI0 (Sample I0). The first signal resets all request flip flops; the second signal samples all request lines, and allows the appropriate channel control flip flops to set.

For example, the channel 0 output data request signal, ODR0, is allowed to set the enabling flip flop (located above the OODR flip flop, Figure 9I11-1B). This enabling flip flop ensures that the OODR signal is dropped and raised again for each request recognized. Assuming the enabling flip flop is in the reset state, the incoming request resets the enabling flip flop and, at the same time, sets the OODR flip flop. If an output buffer instruction on channel 0 had been previously executed; i. e. , OBE0 flip flop set; gate $\overline{\text{OBA0}}$ is enabled. This signal goes to the priority network gate (Figure 9I6) to determine if any higher interrupt type is requesting service (see Figure 9I3). If no higher channel request is present, the function priority logic (Figure 9I13 determines the highest function request on channel 0. Assuming that the only request present is the output buffer, signal $\overline{\text{OBMS}}$ is generated (Figure 9I3-2C) and is used to generate IOAT (Figure 9I1-7B). During CL03 if no power interrupt (BIOR) has occurred to block normal input-output, IO is not busy (IDLE) and a CIS instruction is not in progress; therefore the IORM flip flop is set. The set IORM flip flop prevents the control section from

entering an IA00 cycle and sets IOR1 (Figure 9I1-5C) on the next SYNCH (CLE3) pulse via IOAV (Figure 9I1-5C). With IOR1 set, the IO section enters the active (non-idle) state. A signal from $\overline{\text{IDLE}}$ initiates a memory cycle READ operation. Signal $\overline{\text{OBMS}}$ enters from gate IOBJ (Figure 9I3-7C) into the S-register the value (30 + the channel number) which is the output buffer control word location. The buffer control word is thus obtained from memory and placed in the M-register. During CLP3 the command M to Z is issued from the transfer gates (SZi) shown on Figure 9I5.

IOR1 also enables gate IOAK (Figure 9I1-3C) to set IOR3 on the next SYNCH pulse. With IOR3 set command, ENZA is issued (Figure 9I5-2B) to gate Z_L to S. Also, a READ command is sent to memory from $\overline{\text{IDLE}}$ which begins the memory cycle. Thus the output data is read from core memory into the M-register. The Z-register is incremented by command INCZ (Figure 9I5-5B). The X-register is cleared by REXR (Figure 9I2-4B) and then the contents of the M-register are gated, via OMR3 (Figure 9I2-6B), to the X-register.

A signal from IOR3 also enables the setting of IOR4 which is then set during the next SYNCH pulse. Again, $\overline{\text{IDLE}}$ generates a READ command to memory. The buffer control address control logic (Figure 9I3-7A) is gated by $\overline{\text{EXR4}}$ which is generated during IOR4 by $\overline{\text{EXI0}}$ (Figure 9I2-7D). The inhibits CCWU and CCWL are also generated at this point and sent to memory so that the buffer control word may be written into the buffer control word address. The command Z to M is generated during IOR4 by gate TRZM (Figure 9I2-4A) to gate BCW into the M-registers. The acknowledge control signal is also generated during IOR4 by gate $\overline{\text{ACKN}}$ (Figure 9I2-4A). This acknowledge is present for the entire IOR4 cycle (about 2μs) and then dropped.

The IORM flip flop is reset via IOBA (Figure 9I1-8C); if not block mode has been selected. If a block buffer mode had been called for, a BLOC signal (Figure 9I1-8C) prevents the control section from starting unitl the entire block has been transferred. Flip Flop IORM is not reset by IOBA until BCE1 is a logical "1" (buffer count equal 1). Until the count equals 0001, the I0 operation alternates between IOR3 and IOR4. As before, IOR3 increments Z, transfers Z_L to S, reads data to M, transfers M to X, and steps to IOR4. IOR4 sends the acknowledge and stores the incremented buffer control word. The cycle returns to IOR4 via IOAK (Figure 9I1-3C). Thus one word is transferred each 4 microseconds (250 kc) during the block mode.

Only one Output Data Request is required to start the block mode transfer and it will continue transferring data until the buffer terminates. The Input/Output section, therefore, is locked in a "wired loop" and cannot be interrupted during this period even by a power failure or real time clock interrupt. An acknowledge accompanies each transfer.

(5) Interrupt Mode Processing. - When the interrupt processing logic accepts an interrupt request, the special interrupt address is placed in the S-register and a signal is sent to the control section to initiate an interrupt IA00 sequence. See paragraph 4-3b(6)(a) for a discussion of the interrupt IA00 sequence function.

Figure 9I1-2C contains the "source" flip flops for all non-channel type interrupts.

(a) Power Failure Interrupt. - When a power failure interrupt is sensed (POFA) by the power controller, flip flop PINT (Figure 9I1-1D) is set during SAI0 time. The logical "1" from the set output of flip flop PINT combines with IOLS which is at logical "1" if no interrupt is being processed at the present time (BLKP flip flop cleared). The resulting logical "0" sets flip flop BIOR (Figure 9I1-1A) which blocks any other I0 requests. Power failure interrupts enjoy the highest priority of any I0 type interrupts.

Signal PINT is sent to channel priority (Figure 9I4-1B) where, if no I0 type instruction is being executed, $\overline{\text{PINP}}$ is generated. Signal $\overline{\text{PINP}}$ sets up the fixed location gates IOBI, IOBL, and IOBS which partially enable ST04, ST02, and ST01, respectively. Signal $\overline{\text{PINP}}$ sets flip flop PRIN (Figure 9I1-7A) via the IOAP-IOAR-IOAO-IOAN gates. A signal from PRIN is sent to the control section to set the BLKP flip flop via gate INTR (Figure 9C3-2A).

(b) Manual Interrupt. - The function of the manual interrupt processing is identical with that for the power interrupt described in (a) above except that a different fixed address is placed in the S-register.

(c) External Interrupt (Input). - Processing an external input interrupt requires that the code on the data lines be stored at a special address and an input acknowledge sent to the interrupt source; that is, the peripheral equipment

which is sending the external interrupt to the computer.

For example, the peripheral equipment on channel 0 sends an external interrupt. This request signal, IIR0, (Figure 9I11-6C) sets an enabling flip flop and flip flop OIIR on the channel control logic when the SAI0 (sample I0) pulse is present.

A signal is sent from the set side of this flip flop to gate $\overline{\text{IEAO}}$ (Figure 9I10-7B) which is fully enabled (output a logical "0") if the master lockout flip flop (IL00) is not set. Assuming the master lockout flip flop is cleared, the signal IBA0 is sent to the priority network to determine if any higher priority requests exist at this time. If there is not a higher priority request, signal CH0P (channel 0 priority) is sent back from priority to channel 0 control (Figure 9I10-8D) in order to allow the request to be cleared.

Signals also present at this time from the I0 channel priority logic (Figure 9I-13) are CH0A (channel 0 active) and $\overline{\text{IE00}}$ (Input external interrupt, channel 0). Signal $\overline{\text{IE00}}$ produces IEMS (Input external interrupt mode select) from logic on Figure 9I4-1C. IEMS, in turn, is sent to Figure 9I1-7C to set first the IORM, and then the PRIN flip flop. With the PRIN flip flop set, the next IA cycle will cause the main program to jump to the interrupt address specified by the fixed location gates (Figure 9I3 for this interrupt. The IORM flip flop will "hold up" the control section and substitute the I0 sequence. The I0 timing will produce the necessary commands to store the external function code in the address contained in a special address reserved for the particular channel.

During IOR1, the fixed location gates for the status word are gated to the S-register (Figure 9I3-7C) and the special location contents are read into the M-register and transferred to the Z-register.

During IOR2, the address portion of the Z-register is gated to the S-register and the content of the data lines is gated to the X-register and thence to the M-register. CCWL and CCWU inhibits are sent to memory control. The contents of M (External interrupt code) is read into the memory at the location contained in the special address.

During IOR4, the input acknowledge is sent to the peripheral device and a signal is sent to gate IOAR (Figure 9I1-6C) which combines with IEMS and CL03 to set the PRIN flip flops and an additional signal to IOBA (Figure 9I1-8C) to clear the IORM flip flop. With PRIN set and IORM clear, the control sequence will begin in the interrupt mode (BLKP set) and read the instruction at an address specified by the interrupt (IDLS) fixed location gates (Figure 9I3-5C).

(d) External Interrupt (Output). - Processing of an external interrupt (output) is similar to the interrupt portion of the external interrupt (input) in that only PRIN is set, causing the next instruction to be taken from a fixed interrupt location. There is no requirement to store a code because there is none associated with the output external interrupt. Therefore, no I0 cycle is required and IORM remains cleared.

(e) Real Time Clock Updating. - The real time clock provides for the incrementing of the contents of a special memory address (0024) at a rate governed by a crystal controlled oscillator (RTC0) shown on Figure 9I1-2D. If sense switch 12 is in the ON position, flip flop RTCF (Figure 9I1-2C) will set and clear at the RTC oscillator rate. With RTCF set, flip flop RTCR (real time clock request) will set and will set flip flop RTSA (real time clock service active). This signal will enter the priority network, and if honored, will generate $\overline{\text{RTCP}}$ (Figure 9I2-2A). Signal RTSA will set IORM via IOAT (Figure 9I1-7B). IORM will, in turn, start the I0 sequence by setting IOR1 on the trailing edge of the SYNC pulse. IORM will also "hold up" the control section while the I0 cycle is in progress.

During IOR1, the fixed location 0024 will be forced into the S-register by the fixed location gates (Figure 9I3-6C). Note that the RTCP signal is used for this purpose. The content of the real time clock address is read into M and transferred to the Z-register.

During the IOR2, the real time clock contents are incremented (full 24-bit word). The overflow condition is checked and flip flop RTIA (Figure 9I1-2A) is set if the Z-register goes from all ones to all zeros.

During IOR4, the updated clock count in Z is transferred to M and stored in address 0024. Flip flop IORM is also cleared thereby allowing the control section to resume.

(f) Real Time Clock Interrupt. - If a real time clock overflow had been sensed on the last clock update, flip flop RTIA would be set

(Figure 9I1-2A). This interrupt request signal goes to priority, and, when honored, emerges as RTCP and RTCI. Flip flop PRIN is set by IOAP (Figure 9I1-6C) from signal RTCI and the fixed location gates (Figure 9I3) loaded with (0025). Thus, the next instruction will be taken from this address during the interrupt IA00 cycle.

5. TROUBLESHOOTING

This section contains instructions and procedures for isolating computer malfunctions.

5-1. INTRODUCTION AND GENERAL INFORMATION.

All inspections and tests outlined in this section should be performed prior to placing the computer in general operation.

Inspections and test should be performed before service and repair to determine the extent of repair required after repair to make certain that repairs are made correctly. A record should be kept of failures, operating time, repair time, failure analysis, and failure causes. If a failure occurs while performing tests found in this section, the cause of failure must be determined so that necessary corrective action can be taken. Troubleshooting consists of visual inspection, component testing, and operational testing or demonstrating to provide statistical samples of reliability. The results provide information to determine whether reliability meets requirements, whether reuse is feasible, or whether repair is required.

Section 10 contains the maintenance routines which may be used to ensure that the computer is operating properly (preventive measure) or to assist in localizing malfunctions (corrective measure).

5-2. TEST EQUIPMENT.

The test equipment and tools required for inspecting and testing the computer are as follows:

Special Tools

Westinghouse DPS-2402 Card Tester
DPS-2402 Card Extractor

Standard Equipment

Multimeter AN/PSM-4 series (or equivalent)
Tektronix model 945 (Preamp type MC)
Tektronix model 545 (Preamp type CA)
Voltmeter, Weston #433 (or equivalent)

5-3. TEST CONDITIONS.

The computer must be cooled by air circulating through the equipment at a circulating temperature not greater than 21°C (70°F). Relative humidity must be less than 60 percent.

5-4. VISUAL INSPECTIONS.

The computer should be visually inspected for signs of damage.

5-5. ELECTRICAL TESTS.

Electrical tests consist of checking the voltage output of the computer power supplies. Voltage requirements and test points are outlined in figures 5-5A and 5-5-B.

5-6. POWER CONTROL ASSEMBLY.

The power control assembly provides control of power for distribution to the computer and contains circuits and indicators for monitoring computer operating conditions.

The circuit breakers in the individual power supplies will not be demonstrated because of the possibility of damage to the computer.

a. Computer Power Switch. — Placing the computer power switch on the operator's panel to the ON position will energize all power supplies and will supply power to the blower motors.

b. Time Totalizing Meter. — The time totalizing meter (M1) accumulatively records the time that power is on for distribution to the computer. The range of the meter is 0 to 9999.99 hours and is not resettable.

c. Overtemperature Indicator. — The overtemperature indicator on the operator's panel is illuminated when the internal temperature of the computer exceeds 65°C.

d. Temperature Fault. — The temperature fault indicator on the operator's panel is illuminated when the computer internal temperature exceeds 70°C. Power is removed from the computer.

	ITB1	ITB2	ITB3	ITB4	ITB5	ITB6	ITB7	ITB8
1	+25	-47	NC	-47	+25	+40	NC	+100
2	+6	+6	+6	+6	+6	+6	+6	+6
3	+12	+12	+12	+12	+12	+12	+12	+12
4	-12	-12	-12	-12	-12	-12	-12	-12
5	gnd	gnd	gnd	gnd	gnd	gnd	gnd	gnd ret for 100

	2TB1	2TB2	2TB3	2TB4	2TB5	2TB6	2TB7	2TB8
1	+40	+25	-47	NC	-47	+25	+40	NC
2	+6	+6	+6	+6	+6	+6	+6	+6
3	+12	+12	+12	+12	+12	+12	+12	+12
4	-12	-12	-12	-12	-12	-12	-12	-12
5	gnd	gnd	gnd	gnd	gnd	gnd	gnd	gnd

Figure 5-5-A. DC Voltage at Chassis Bus

A5	+6V
A6	-12V
A7	+12V
A8	+25V
A9	+47V

Figure 5-5-B. Power Supply Assemblies

5-7. CONSOLE CONTROLS AND INDICATORS.

The operator and maintenance panels allow the operator or maintenance technician to control the computer. A list of the various controls and indicators and a brief description of their function is found in Section 3.

5-8. TESTING THE MASTER CLOCK.

The master clock pulse generation logic is capable of operating in five modes as specified by the RUN MODE select switch.

a. Testing NORMAL RUN Condition

(1) Place the RUN MODE select switch in the NORMAL RUN position.

(2) Place the ADVANCE P switch in the DISCONNECT ADVANCE P position.

(3) Master clear the computer.

(4) Store the instruction 47600000 in address 00000.

(5) Depress the RUN switch. Clock pulses will be generated at the normal high speed rate.

(6) Externally synchronize the oscilloscope with test jack 1A408-JH.

(7) Set up.

(8) Observe clock waveforms and relative positions, noting voltage levels, overshoot, rise and fall times, etc. Figure 5-8-A shows the ideal waveform pattern.

b. Testing the MICROSTEP Operation

(1) Follow steps two through 4 of 5-8a above.

(2) Set the RUN MODE select switch to MICROSTEP.

(3) Observe the CL2A, CL2B, and CL2C indicators on the maintenance panel and depress the RUN switch. The following sequence should be observed each time RUN is depressed:

Initial	CL01	CL2A set, CL2B clear, CL2C clear
1st	CLE1	CL2A set, CL2B set, CL2C clear
2nd	CL02	CL2A set, CL2B set, CL2C set
3rd	CLE2	CL2A clear, CL2B set, CL2C set
4th	CL03	CL2A clear, CL2B clear, CL2C set
5th	CLE3	CL2A clear, CL2B clear, CL2C clear
6th	CL01	CL2A set, CL2B clear, CL2C clear

One clock pulse as shown should be generated each time the RUN switch is depressed. There is no way to cause the microstep function to repeat at a rate sufficient for observation. However, the "Ready" light feature can be used to detect the presence of a single pulse.

c. Testing the INSTRUCTION STEP Operation

Each time the RUN switch is depressed, a single instruction should be executed by the computer. To test this condition, proceed as follows:

(1) Master clear the computer.

(2) Place the RUN Mode select switch in the INST STEP position.

(3) Store 24600001 in address 00000.

(4) Store 36000000 in address 00002.

(5) Depress the RUN switch. The A register should count up by two each cycle of the two instruction loop. The P register will go from 0 to 1 and back to 0 each time RUN is depressed.

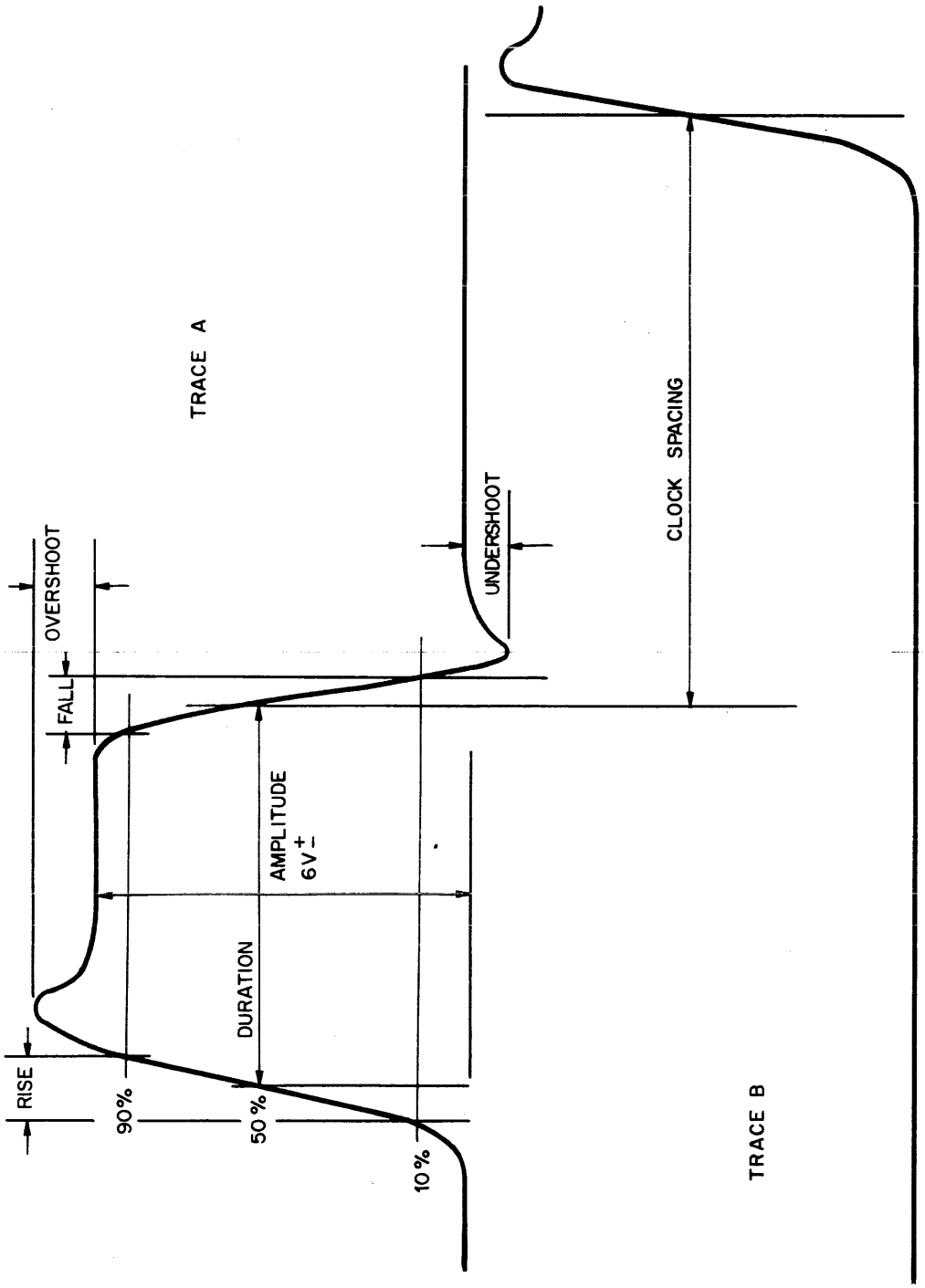


Figure 5-8-A. Clock Waveforms - Normal Run Condition