

VOLUME I OF II

Laboratory Maintenance Instructions

SATURN V LAUNCH VEHICLE DIGITAL COMPUTER

Simplex Models

NASA Part No. 50M35010

IBM Part No. 6109030

(International Business Machines Corporation)

Contract NAS 8-11561

CHANGE
NOTICE

**LATEST CHANGED PAGES SUPERSEDE
THE SAME PAGES OF PREVIOUS DATE**

Insert changed pages into basic
publication. Destroy superseded pages.

VOLUME I

GENERAL DESCRIPTION AND THEORY

TECHNICAL LIBRARY
BELLCOMM, INC.
955 L'Enfant Plaza North
Washington, D. C. 20024
B.W.

30 NOVEMBER 1964

CHANGED 4 JANUARY 1965

LIST OF EFFECTIVE PAGES

INSERT LATEST CHANGED PAGES. DESTROY SUPERSEDED PAGES.

TOTAL NUMBER OF PAGES IN VOLUME I IS 226 CONSISTING OF THE FOLLOWING:

Page No.	Issue	Page No.	Issue
*Title	4 Jan 65	* 10	4 Jan 65
*A	4 Jan 65	Glossary-	
*i	4 Jan 65	1 thru 13	Original
ii Blank	Original	14 Blank	Original
iii thru iv	Original	Index-	
*v	4 Jan 65	* 1	4 Jan 65
vi Blank	Original	2	Original
vii	Original	* 3	4 Jan 65
viii Blank	Original	4 Blank	Original
ix	Original		
x Blank	Original		
1-1	Original		
*1-2	4 Jan 65		
1-3 thru 1-4	Original		
*1-5 thru 1-7	4 Jan 65		
1-8	Original		
2-1 thru 2-14	Original		
*2-15	4 Jan 65		
2-16 thru 2-33	Original		
2-34 Blank	Original		
*2-35	4 Jan 65		
2-36 thru 2-51	Original		
*2-52 thru 2-53	4 Jan 65		
2-54	Original		
*2-55	4 Jan 65		
2-56	Original		
*2-57	4 Jan 65		
2-58 thru 2-70	Original		
*2-71	4 Jan 65		
2-72 thru 2-108	Original		
*2-109	4 Jan 65		
2-110 thru 2-121	Original		
*2-122	4 Jan 65		
2-123 thru 2-143	Original		
*2-144	4 Jan 65		
2-145 thru 2-159	Original		
*2-160 thru 2-162	4 Jan 65		
*2-163 thru			
2-177 Added	4 Jan 65		
*2-178 Blank			
Added	4 Jan 65		
Logic Symbols-			
1 thru 9	Original		

*The asterisk indicates pages changed, added, or deleted by the current change.

TABLE OF CONTENTS

Section	Title	Page
	LIST OF ENGINEERING CHANGES	vii
	LIST OF RELATED MANUALS	ix
I	INTRODUCTION AND DESCRIPTION	1-1
	1-1. Introduction	1-1
	1-8. Structural Description	1-2
	1-17. Electrical Description	1-6
II	THEORY OF OPERATION	2-1
	2-1. <u>Scope and Purpose</u>	2-1
	2-4. <u>Computer Organization</u>	2-1
	2-6. <u>Functional Organization</u>	2-1
	2-47. Word Organization	2-10
	2-52. Timing Organization	2-11
	2-57. Instruction Organization	2-13
	2-61. <u>Computer Logic Circuits</u>	2-13
	2-62. <u>Timing Element</u>	2-13
	2-88. Memory Element	2-32
	2-140. Memory Control Element	2-59
	2-192. Data Control Element	2-85
	2-209. Program Control Element	2-91
	2-271. Arithmetic Element	2-117
	2-303. Multiply-Divide Element	2-131
	2-418. Delay Lines	2-174
	2-423. Voting Element	2-175
	2-430. Computer Operations	2-176
	Logic Symbols	Logic Symbols-1
	Glossary	Glossary-1
	Index	Index-1



LIST OF ILLUSTRATIONS

Figure	Title	Page
1-1	Computer Assemblies	1-3
1-2	Logic Printed Circuit Cables	1-4
1-3	Page Assembly	1-4
1-4	Reference Designator Arrangement	1-5
1-5	Page Assembly Location Guide	1-7
1-6	Voltage/Function Listing	1-8
1-7	Voter Circuits, Block Diagram	1-8
1-8	Channel Module Arrangement, Block Diagram	1-8
2-1	Computer Functional Block Diagram	2-3
2-2	Data Word Layout	2-10
2-3	Instruction Word Layout	2-11
2-4	Computer Timing Organization	2-12
2-5	Operation Code Map	2-14
2-6	Selecting An Instruction	2-15
2-7	Operand Address	2-15
2-8	List of Instructions (5 Sheets)	2-16
2-9	PIO Addresses (2 Sheets)	2-21
2-10	Clock Generator Block Diagram	2-23
2-11	Buffer-Amplifier	2-24
2-12	Timing Logic	2-25
2-13	Clock Drivers	2-27
2-14	Clock Generator Timing	2-28
2-15	Timing Gate Generator Block Diagram	2-28
2-16	Control Circuit	2-28
2-17	Shift Register	2-30
2-18	Phase Generator	2-31
2-19	TBC Latch	2-31
2-20	Computer Timing	2-33
2-21	Memory Element Block Diagram	2-35
2-22	Ferrite Core Characteristics	2-36
2-23	6 × 6 Core Plane	2-37
2-24	Core Array Windings	2-39
2-25	Memory Module Block Diagram	2-41
2-26	Toroidal Core Array	2-42
2-27	Drive Line Assignments	2-43
2-28	Memory Clock Drivers	2-44
2-29	Memory Timing Diagram	2-46
2-30	EI Driver Logic Symbol	2-47
2-31	Paired EI Drivers	2-47
2-32	Y Drive Line Selection	2-49
2-33	X Drive Line Selection	2-50
2-34	Timing Pulse Assignments	2-51
2-35	Inhibit Driver And Winding	2-52

LIST OF ILLUSTRATIONS (cont)

Figure	Title	Page
2-36	Memory Sense Amplifier and Winding	2-53
2-37	Error Detector Logic Symbol	2-54
2-38	Temperature Controlled Voltage (TCV) Regulator Logic Symbol	2-55
2-39	Memory Address Decoding	2-57
2-40	Memory Addressing Diagram	2-58
2-41	Memory Buffer Register, Typical Latches	2-60
2-42	Memory Buffer Register Timing Circuits	2-61
2-43	Memory Address Block Diagram	2-62
2-44	Lo X Decoder	2-63
2-45	Lo Y Decoder	2-64
2-46	Hi X Decoder	2-65
2-47	Hi Y Decoder	2-66
2-48	Syllable Select Latches	2-68
2-49	Memory Sync Timing Block Diagram	2-69
2-50	Memory Sync Timing Circuit	2-71
2-51	Memory Read Sync Timing Diagram	2-72
2-52	Memory Store Sync Timing Diagram	2-73
2-53	Memory Mode and Module Select Block Diagram	2-75
2-54	Memory Module Registers	2-76
2-55	Memory Module Select Gates	2-78
2-56	Memory Module Select Control Circuit	2-79
2-57	Exclusive Or (XOR) Circuits	2-80
2-58	A Memory Parity Check Circuit	2-81
2-59	Memory Error Monitor Circuit	2-82
2-60	Memory Buffer Select Latches	2-84
2-61	Transfer Register (2 Sheets)	2-86
2-62	Transfer Register Simplified Block and Timing	2-89
2-63	Parity Counter	2-91
2-64	Parity Counter Timing	2-92
2-65	Address Register and Transfer Address Latch	2-94
2-66	Address Register Timing	2-95
2-67	Sector Registers and Control Latches	2-96
2-68	Data and Instruction Sector Register Timing	2-98
2-69	Operation Code Register	2-99
2-70	Operation Code Register Timing	2-100
2-71	Operation Decoders (2 Sheets)	2-102
2-72	Interrupt Control	2-106
2-73	Start-Stop Control	2-107
2-74	HOP Constant Serializer	2-109
2-75	Automatic HOP Save Circuit Timing	2-111
2-76	Delay Circuits	2-112
2-77	STMD Latch	2-113
2-78	Instruction Counter (2 Sheets)	2-114
2-79	Instruction Counter Timing	2-116

LIST OF ILLUSTRATIONS (cont)

Figure	Title	Page
2-80	Arithmetic Element Block Diagram	2-118
2-81	Arithmetic Element (2 Sheets)	2-120
2-82	Arithmetic Element General Timing	2-122
2-83	ZER Latch UTR-UACCO Latch Timing	2-124
2-84	A/S Circuit, Simplified	2-126
2-85	A/S Circuit Truth Table	2-127
2-86	LSD-Shift Timing	2-130
2-87	MSD-Shift Block Diagram	2-132
2-88	MSD-Shift Timing	2-133
2-89	Table of Delta Values	2-138
2-90	Table of Multiplier Bit Effective Values	2-139
2-91	Multiply Circuits, Block Diagram	2-142
2-92	Command Circuit	2-143
2-93	Multiply-Divide Timing Latches	2-144
2-94	Phase Counter	2-145
2-95	Multiplicand Register, Block Diagram	2-146
2-96	Multiplier and Product-Quotient Registers, Block Diagram	2-147
2-97	Partial Product Register, Block Diagram	2-147
2-98	Multiplicand-Divisor Register	2-148
2-99	Multiplier Quotient and Product-Quotient Registers	2-149
2-100	Partial Product-Remainder Register (4 Sheets)	2-151
2-101	Multiply Timing Diagram	2-155
2-102	Multiply Computation Cycle Flow Diagram	2-159
2-103	Sample Restoring Divide Problem	2-162
2-104	Sample Nonrestoring Divide Problem	2-163
2-105	Nonrestoring Divide with Signal Shortcut	2-165
2-106	Truth Table for Second Quotient Bit	2-166
2-107	Divide Circuit, Block Diagram	2-167
2-108	Divisor Register, Block Diagram	2-170
2-109	Quotient and Product - Quotient Registers	2-171
2-110	Remainder Register, Block Diagram	2-172
2-111	Delay Line, Simplified Diagram	2-174
2-112	Voter Circuits	2-177



LIST OF ENGINEERING CHANGES

The following list of Engineering Changes applies to the Breadboard II model of the computer. Each letter or pair of letters represents a different Engineering Change.

EC Number

- 66109 EM, ER, GE, GM, GZ, HD, KR, KZ, LD through LH, LJ, LM
- 66123 HK, KB, LD, NP, PK, PL, PM, SF, SW, WM
- 66124 A through H, J through N, P, R, S, T, V, W, Z
AA, AB, AD through AH, AJ through AN, AP,
AR, AS, AT, AV, AW, BA through BH, BJ through
BN, BP, BR, BS, BT, BV, BW, BZ, CA, CC
through CH, CJ through CN, CP, CR, CS, CT,
CV, CW, CZ, DA through DH, DJ through DN,
DR, DS, DV, DW, DZ, EA, EC through EH, EJ
through EN, EP, ER, ES, ET, EV, EW, EZ,
FA through FH, FJ through FM, FP, FR, FS,
FT, FV, FW, FZ, GA through GH, GJ through
GK, GM, GN, GP, GR, GS, ST, GV, GW, GZ,
HA through HH, HJ, HK, HL, HN, HP, HR, HT,
HV, HW, HZ, JA through HG, JJ through JM,
JP, JR, JS, JT, JV, JW, JZ, KB, KC, KD, KF,
KG, KJ, KL, KM, KN, KR, KS, KT, KV, KW,
LA
- 66125 C, F, K, Z, AH, AK, AN, BB, BC, BF, BG,
BM, BS, BT, CA, CG, CR, CT, DA, DV
- 66126 K, AF, AM, BF, CG, DH
- 66127 AD, AF, AH, AK



LIST OF RELATED MANUALS

Manual Title

Technical Manual, Laboratory Maintenance Instructions, Launch
Vehicle Data Adapter (Simplex Models)

Technical Manual, Checkout Procedures, for Saturn Launch Vehicle
Digital Computer and Data Adapter

Technical Manual, Laboratory Maintenance Instructions, Saturn V
Test Equipment, Vol. I, Vol. II, Vol III

2

2

2

SECTION I

INTRODUCTION AND DESCRIPTION

1-1. INTRODUCTION.

1-2. **PURPOSE OF MANUAL.** This manual contains the laboratory maintenance instructions for breadboard model II of the Launch Vehicle Digital Computer (LVDC), NASA part number 50M35010, IBM part number 6109030, manufactured by International Business Machines Corporation, Federal Systems Division, Rockville, Maryland, under contract number NAS 8-11561.

1-3. The Launch Vehicle Digital Computer, breadboard II Model is hereafter referred to as the computer.

1-4. This manual is divided into two volumes with descriptive material and supporting diagrams in volume I and procedural material and reference diagrams in volume II. The index and all appendices are included with volume I. Volume I may be used separately for training; volume I and volume II should be used together for maintenance. This arrangement increases the handling convenience of the manual.

1-5. Within volume I frequently referred to diagrams are printed only on one side so that they may be detached and referenced while reading the text. Included in the binder's pocket is a plastic-covered timing diagram. This diagram is a duplicate of the referenced timing diagram and is supplied as a special aid to the reader.

1-6. **PURPOSE OF EQUIPMENT.** The computer performs four functions in the Saturn V Launch Vehicle: pre-launch checkout, guidance from launch to parking orbit, orbital checkout, and back-up guidance from parking orbit to lunar transfer trajectory. The breadboard II model is provided for evaluation only, and will not be used in flight.

1-7. **BASIC CHARACTERISTICS.** The basic characteristics of the computer are as follows:

- **TYPE** — binary, serial, general purpose, stored program.
- **ACCURACY** — Add/Subtract: 2^{-25} ; Multiply/Divide: 2^{-23} .
- **SPEED** — 512,000 bits per second serial; Add: 82 usec; Multiply: 328 usec; Divide: 656 usec.
- **MEMORY** — Ferrite core, random access, 4,095 locations of 28 bits each memory; two self-contained memory modules.
- **ARITHMETIC** — Fixed point, 2's complement.
- **PROGRAMMING** — 18 instruction codes consisting of 10 arithmetic instructions, 6 program control instructions, 1 input/output instruction, and 1 store instruction.
- **ELECTRONICS** — Current-switching diode logic (positive), micro-miniature packaging.

- WORD MAKE-UP — Data word: Sign + 25 magnitude bits; Instruction word: 4 operation code +9 operand address bits.
- TIMING — 4-clock system, clock repetition rate: 2.048 MC; bit time: 1.953 microseconds; phase time: 14 bit times; computer cycle: 3 phase times.
- INPUT/OUTPUT — Program controlled, serial.

1-8. STRUCTURAL DESCRIPTION.

1-9. GENERAL. The computer (figure 1-1) measures approximately 29-1/2"×12-1/2"×10-1/2" and weighs approximately 72-1/2 pounds. Electrical inputs and outputs are supplied through eight 55-pin connectors numbered J1 through J8. Directly beneath connector J8 is a time indicator which records the total operating hours of the computer.

1-10. The connectors are located in the front side of the computer. Connectors J5 through J8 and the time indicator are adjacent to the right side. Proceeding clockwise, the remaining sides are bottom, left and top, respectively. Orientation may be quickly established by remembering that the time indicator is located in the bottom right corner of the front side.

1-11. The computer consists of two major sections, the logic section and the memory section. The two sections are electrically interconnected by four pluggable printed circuit cables.

1-12. LOGIC SECTION. The logic section consists of a frame assembly, five panel assemblies, and a number of page assemblies or "pages". The pages slide into mounting grooves in the frame assembly and are electrically interconnected by the panel assemblies. The panel assemblies, which are mounted on the back of the frame assembly, contain sockets into which the pages are plugged. The panel assemblies are interconnected by printed circuit cables mounted on terminal blocks. (See figure 1-2.) (The printed circuit cables and terminal blocks are mounted on the rear of the back panels and cannot be seen in figure 1-1.)

1-13. The panel assemblies are connected to the eight interface connectors by means of individual wires from the connector pins to the terminal block on the panel assembly.

1-14. The page assembly (figure 1-3) has logic components located on both sides, called A and B. Thirty-six test points, 18 on each side, are available at the top of each page. The test points are found in groups of nine on either side of the "through" pins also found at the top of the page. The "through" pins provide interconnection between the logic circuits on either side of the page.

1-15. MEMORY SECTION. The memory modules are self-contained assemblies with memory timing, drive, inhibit, and sensing circuits arranged around the core array. The modules are mounted on the memory plate assembly and electrically connected to the memory plate assembly by means of a pluggable harness. The printed circuit cables mentioned earlier then connect the memory to the logic via the memory plate assembly.

1-16. REFERENCE DESIGNATORS. Figure 1-4 illustrates the reference designators that have been assigned to the major assemblies, connectors, and terminal blocks. In the logic section, the combination of a panel assembly and the attaching pages comprises a "channel." Figure 1-5 illustrates where the pages (by part number and logic function)

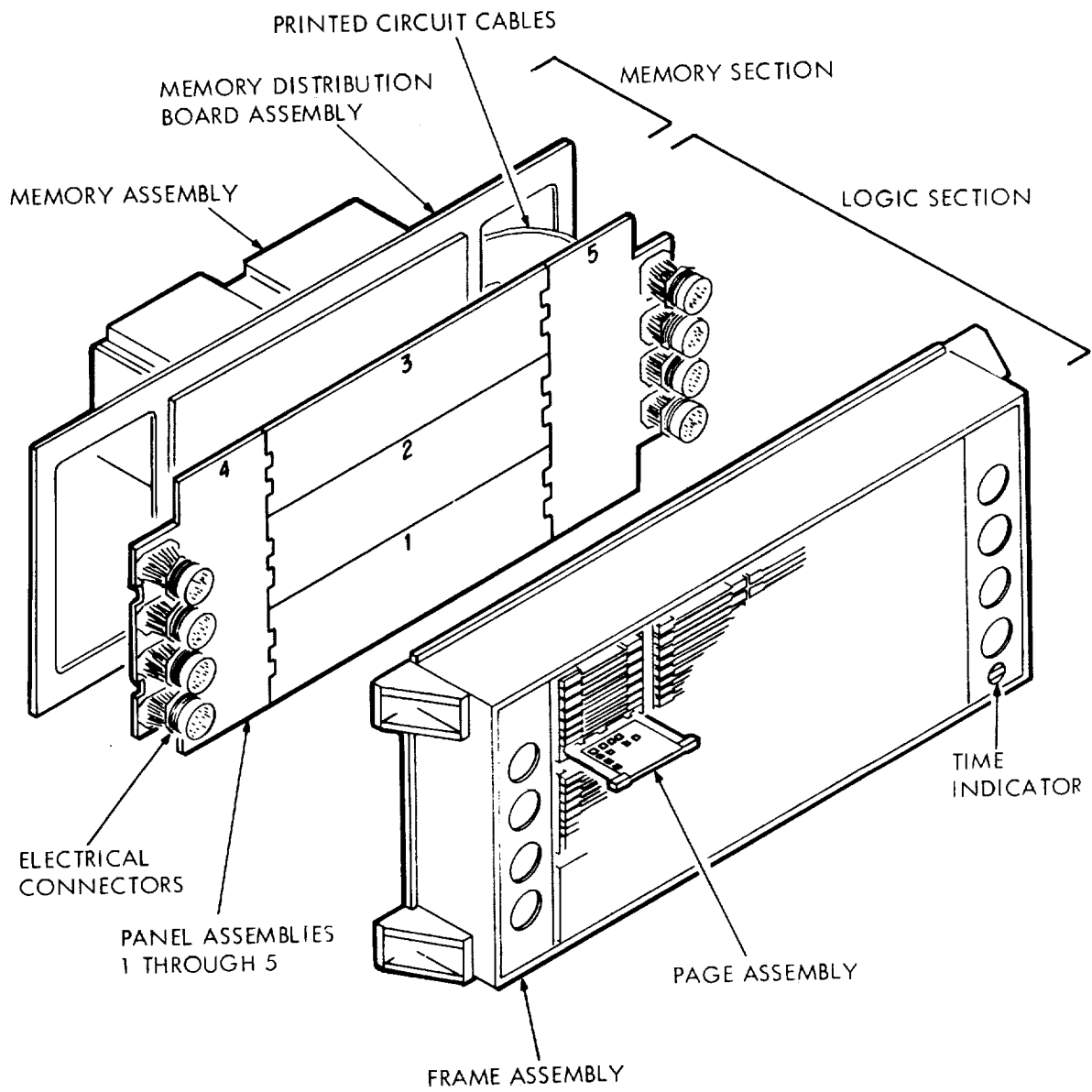


Figure 1-1. Computer Assemblies

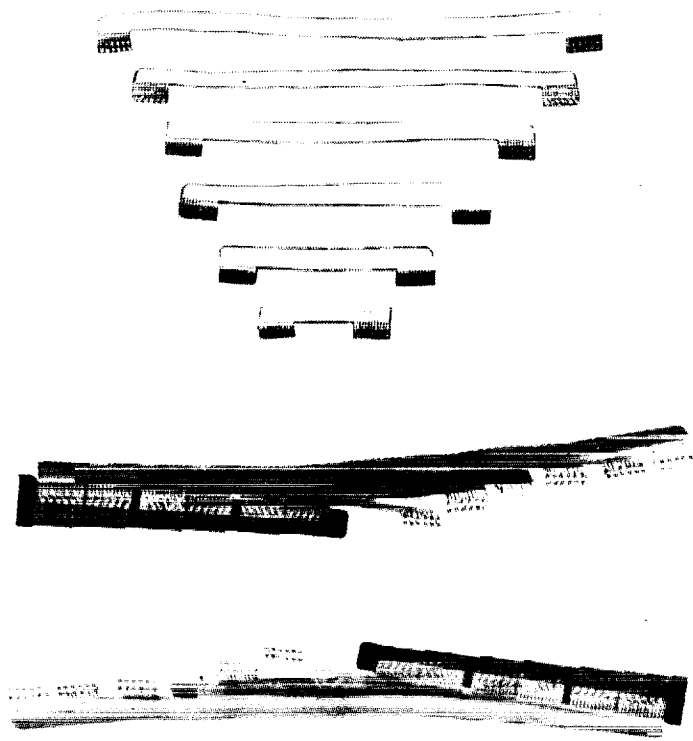


Figure 1-2. Logic Printed Circuit Cables

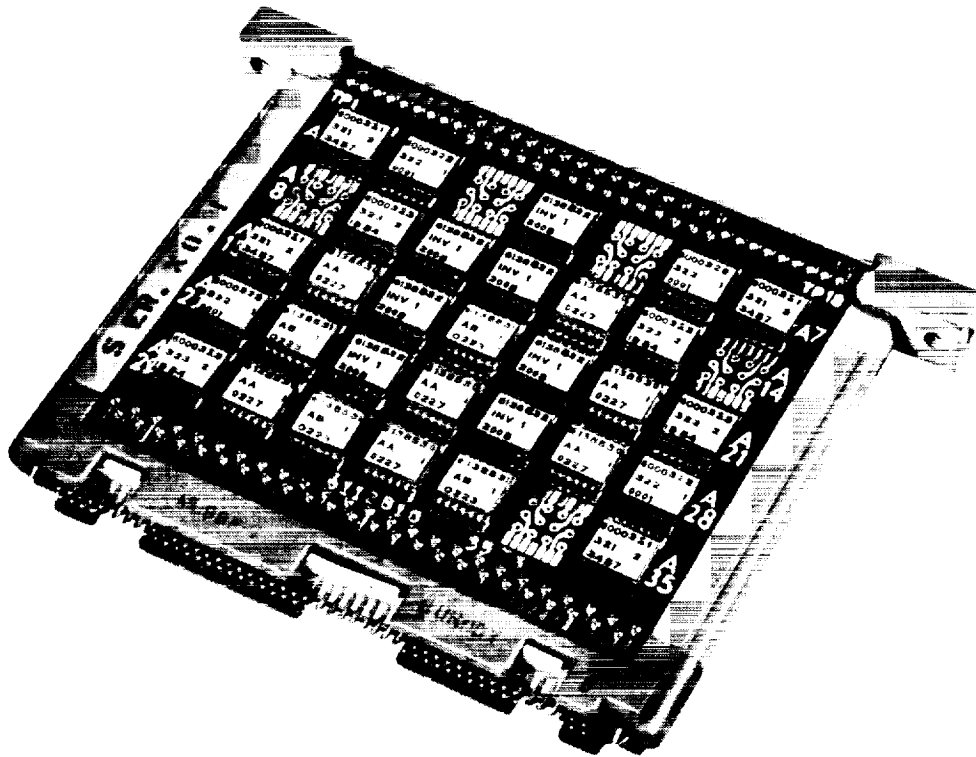


Figure 1-3. Page Assembly

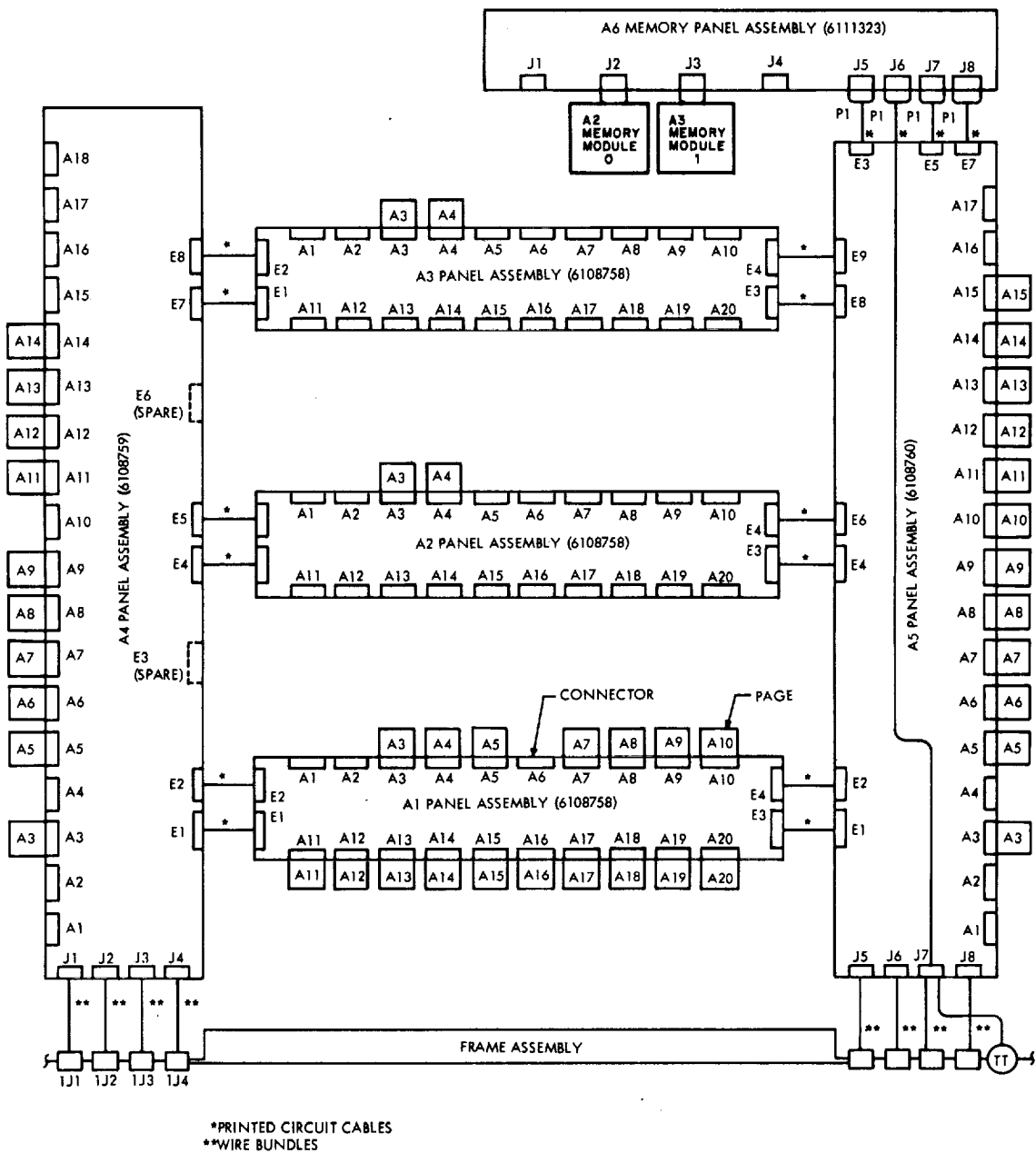


Figure 1-4. Reference Designator Arrangement

Changed 4 January 1965

have been connected to the panel assemblies. Signals are prefixed with their reference designators. Thus, if signal ZER were located on the A side of the page in the A10 position of channel 1, its full title would be A1A10A-ZER.

1-17. ELECTRICAL DESCRIPTION.

1-18. VOLTAGE DISTRIBUTION. All DC power for the computer is supplied by the data adapter. The voltages and their functions are listed in figure 1-6. Figure 3-2 contains the input connector-pin locations for these inputs from the data adapter.

1-19. TRIPLE MODULAR REDUNDANCY (TMR). A failure of virtually any circuit in the computer, even a momentary failure, could conceivably cause intolerable errors in computation. To prevent such errors, the computer circuits are redundant in the form of three identical sets, or channels, of logic. Signals from the three channels are fed to "voter" circuits which sense the majority input. Any differing signal is thereby "out-voted" by two unanimous signals (figure 1-7).

1-20. Each channel is divided into seven aggregates of circuits called modules (figure 1-8). Each module has an average of 15 outputs which are voted. Errors from any one module are outvoted before they can be passed on to succeeding modules. Thus, it is possible to have numerous failures without affecting operation, provided that two identical failures do not occur simultaneously.

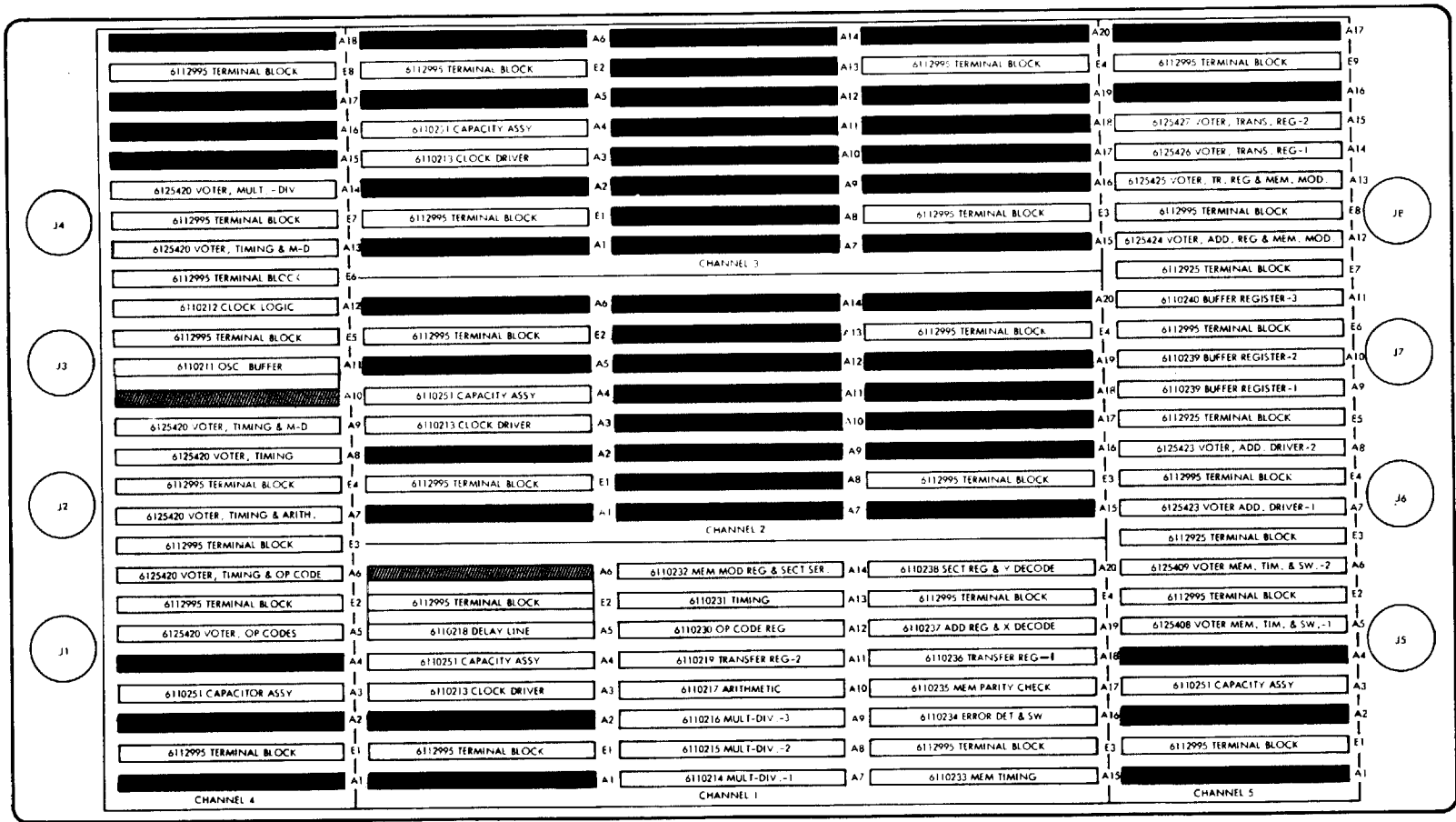
NOTE

The breadboard II model contains only one channel of logic. Provision is made to hold the missing-channel inputs of the voters to a "tie vote", i. e. , one "0" and one "1". The existing channel signal then breaks the tie, and the output of the voter is always equivalent to the existing channel input.

1-21. DISAGREEMENT DETECTORS. Since the voter outputs mask errors by providing correct outputs, a circuit called a disagreement detector is provided. This circuit provides an output when one of the inputs to a voter differs from the other two.

NOTE

No disagreement detectors are provided with the breadboard II model.



LEGEND

- SPARE LOCATION
- SPARE LOCATION, NOT USABLE BECAUSE OF COMPONENT OVERHANG

Figure 1-5. Page Assembly Location Guide

Voltage	Function
V1 01 thru V1 24	+6 volts applied to logic
V1 MEM1 thru V1 MEM3	+6 volts applied to memory
V3 MEM1, V3 MEM2	-3 volts applied to memory
V3 01 thru V3 10	-3 volts applied to logic
V4 M1 thru V4 M7	+6 volts used for module switching
V5 01, V5 02	+12 volts applied to logic
V5 MEM1, V5 MEM2	+12 volts applied to memory
V5 M1 thru V5 M7	+12 volts used for module switching
V20 01, V20 02	+20 volts applied to logic
V20 AM1, V20 AM2	+20 volts applied to even memory
V20 BM1, V20 BM2	+20 volts applied to odd memory

Figure 1-6. Voltage/Function Listing

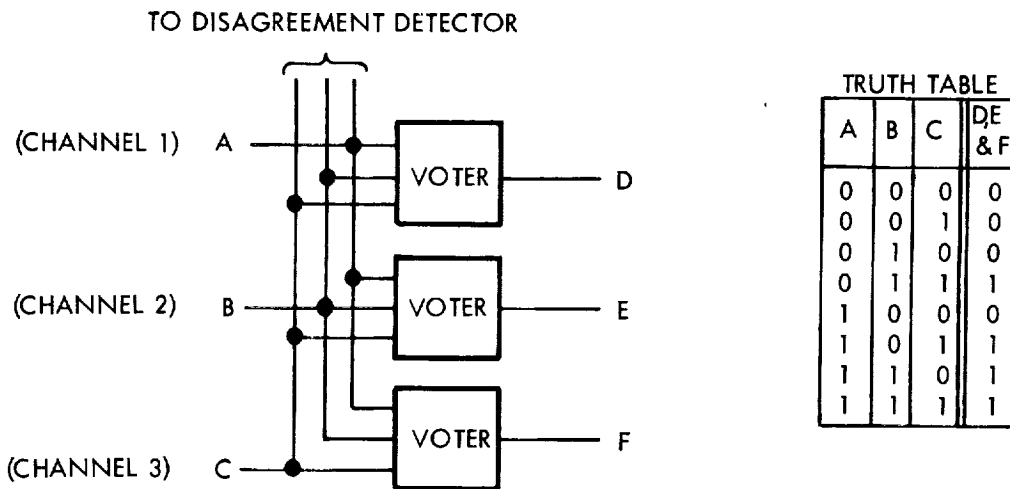


Figure 1-7. Voter Circuits, Block Diagram

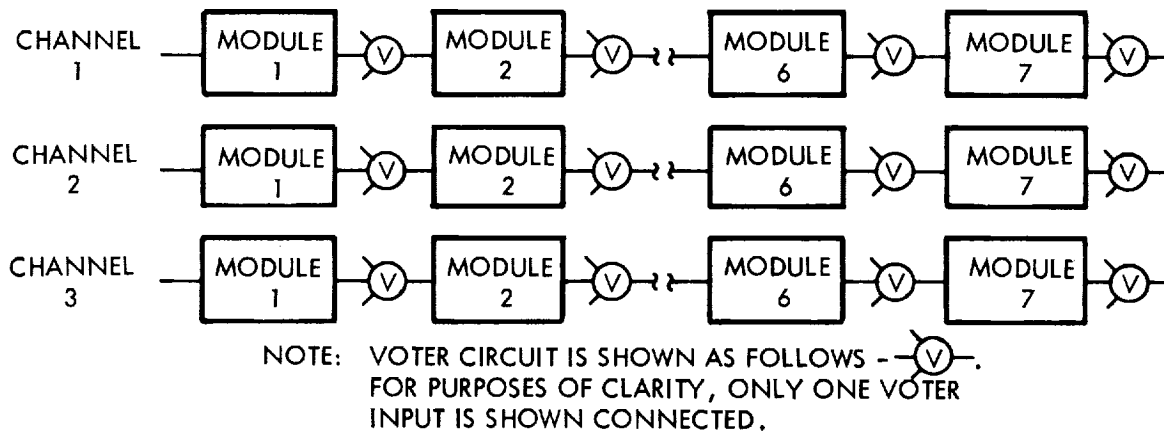


Figure 1-8. Channel - Module Arrangement, Block Diagram

SECTION II

THEORY OF OPERATION

2-1. SCOPE AND PURPOSE.

2-2. This section describes the operation of the computer. Descriptions contained herein are provided to enable maintenance personnel to understand the operation of the computer at the basic logic-block level.

2-3. First, the general organization of the computer is presented to provide a framework into which details may be fitted. Next, the logic circuits are divided into "functional elements" with each element being described in detail. Finally, the functional elements are tied together by detailed descriptions of all the computer operations.

2-4. COMPUTER ORGANIZATION.

2-5. The following paragraphs describe the organization of the computer from the aspects of its functional organization, its word layout and timing, and the instructions it performs.

2-6. FUNCTIONAL ORGANIZATION

2-7. The LVDC is a general purpose computer which, under control of a stored program, processes data using fixed point, two's complement arithmetic. Data is processed serially in two arithmetic sections which can, if desired, operate concurrently. Addition, subtraction, and logical extractions are performed in one arithmetic section while multiplication and division are performed in the other.

2-8. The principal storage device is a random access ferrite-core memory with separate controls for data and instruction addressing. The memory can be operated in either a simplex or duplex mode. In duplex operation, memory modules are operated in pairs with the same data being stored in each module. Readout errors in one module are corrected by using data from its mate to restore the defective location. In simplex operation, each module contains different data which doubles the capacity of the memory. However, simplex operation decreases the reliability of the computer since the ability to correct readout errors is sacrificed. The memory operating mode is program controlled. Temporary storage is provided by static registers composed of latches and by shift registers composed of delay lines and latches.

2-9. The computer is composed of the following eight functional elements:

- | | |
|--------------------|---------------------|
| (1) Timing | (5) Program Control |
| (2) Memory | (6) Arithmetic |
| (3) Memory Control | (7) Multiply-Divide |
| (4) Data Control | (8) Voting |

Figure 2-1 is a block diagram of the computer showing the major parts of each functional element (except the Voting Element) and how they are interconnected. The heavy lines on the diagram show major paths of information flow.

2-10. The Timing Element develops three levels of standard timing signals which synchronize and control logic operations in the remaining seven elements. Direct outputs of the Timing Element are used in every functional element except the Memory Element which generates its own timing signals.

2-11. Memory Element. The Memory Element and the Memory Control Element operate together to store the programmed instructions and the constants and data required by the program. The Memory Element is composed of up to eight memory module assemblies, each of which contains a ferrite-core array and all the circuits necessary to transfer data to and from it. The Memory Control Element determines when a transfer will occur and selects both the direction of transfer and the memory location to be exercised. Buffer storage for data enroute between the memory and the central computer is also located in the Memory Control Element.

2-12. In addition to controlling data transfers, the Memory Control Element determines the mode (simplex or duplex) in which the memory will operate. In duplex operation, the memory modules are divided into two groups, effectively forming two separate memories with identical contents. The even numbered modules make up the "A" memory and the odd numbered modules make up the "B" memory. Each module in the A memory is paired with a module in the B memory which contains the same information. The Memory Control Element selects both modules of a pair and provides separate buffer storage for each memory. The two memories are operated simultaneously but the computer uses information from only one memory. The Memory Select and Error Monitor circuit determines which memory will be used and changes the selection when an error is detected in the active memory. The Memory Select and Error Monitor circuit continuously samples the drive currents in each memory and checks the parity of each word transferred out of both memories. In simplex operation, each module contains different information and only one module is selected at a time. The Memory Select and Error Monitor circuit then makes the memory selection follow the selected module so that the B memory is active when an odd numbered module is selected and the converse. In simplex operation the ability to correct errors is sacrificed.

2-13. Data and constants are stored in the memory in two segments called syllables. Thus, the memory is described as being divided into syllables. One syllable of data is transferred to or from the memory at a time; consequently, two memory cycles are necessary to transfer a complete data word. Instructions are only half the length of data words and thus are stored one per syllable. The syllable selection for instructions is stored in the Syllable Select circuit of the Memory Control Element. Timing signals determine whether the stored selection will be used or the selection will be sequenced to read both syllables of a data word.

2-14. When a memory operation is required, the Memory Control Element provides a sync impulse to the Memory Element. Memory Mode and Module Select circuits direct the impulse to two memory modules for duplex operation or one module for simplex. Once impulsed, each memory module generates all the timing signals and drive currents required to complete its cycle.

2-15. The Memory Clock Drivers produce either read or store timing pulses, depending on a control signal from the Memory Timing and Sync Select circuits. The Memory Address Decoders reduce the address of the desired memory location to selection signals for the X and Y memory address drivers of that location in each syllable of the memory.

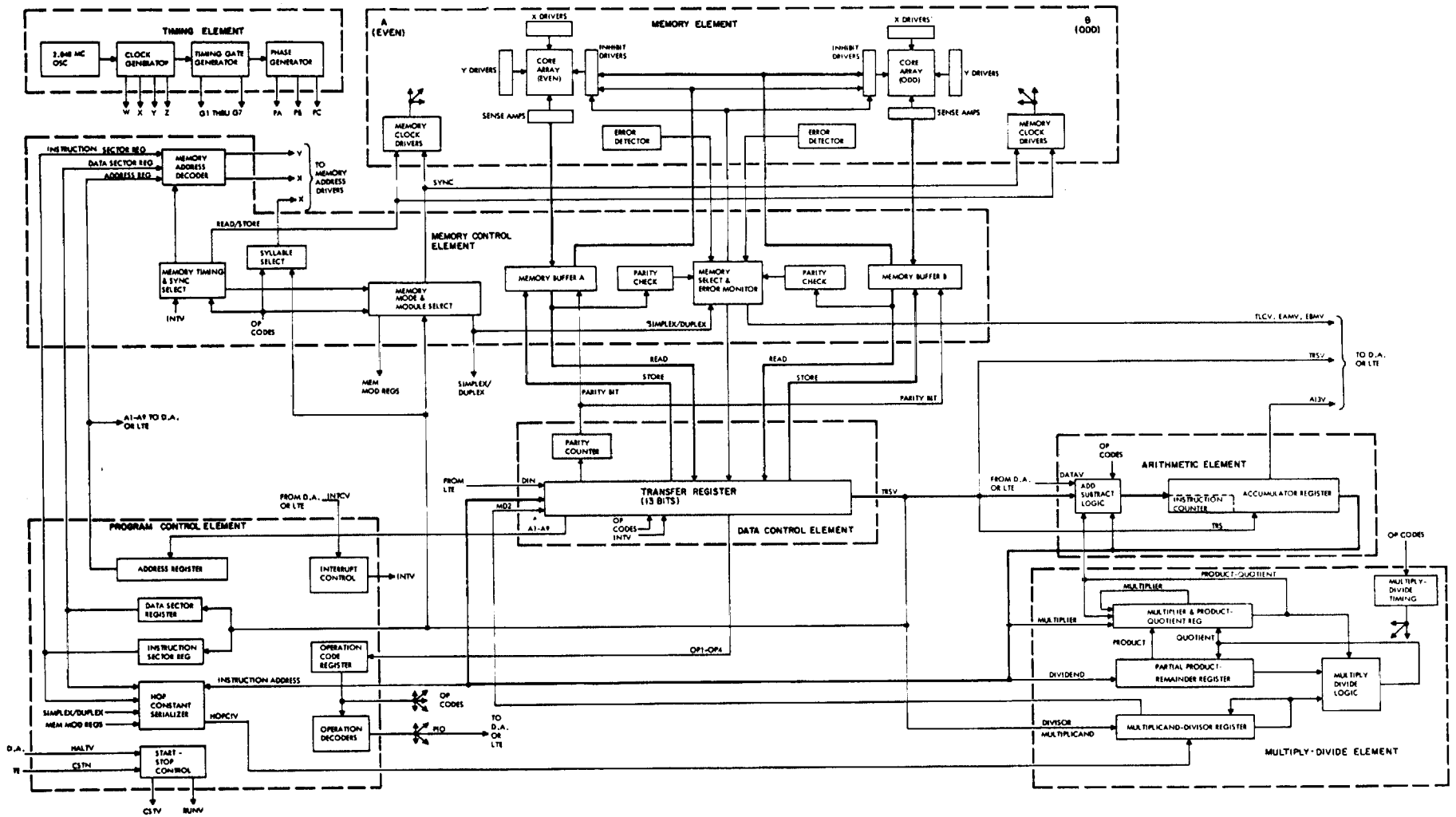


Figure 2-1. Computer Functional Block Diagram

The Syllable Select circuit determines which syllable will be used. Syllable and address selection signals are applied to all the modules in the memory. However, only the module (or modules for duplex) which receive a sync impulse will generate the memory clocks which cause the module to cycle. When the memory clocks occur, data is transferred in parallel between the Memory Buffer Registers (A and B) and the selected memory location. All data entering or leaving the memory passes through the Memory Buffer Registers enroute to or from the Data Control Element.

2-16. Data Control Element. Data words and constants enter the Transfer Register from the memory in two syllables. Each syllable enters the Transfer Register in parallel and is then serialized and distributed to the Arithmetic Element, the Multiply-Divide Element or the data adapter. Some special constants are also distributed to the Program Control and Memory Control Elements in serial form.

2-17. Data words which are bound for the memory enter the Transfer Register in serial form; the Transfer Register then divides them into syllables and forwards each syllable to the Memory Buffer Registers in parallel. At the Memory Buffer Registers, a "parity bit" is added to each syllable before it is stored in the memory. The Parity Counter monitors the number of "1's" entering the Transfer Register and assigns odd parity to the Memory Buffer Registers by controlling the parity bit.

2-18. Another item which enters the Transfer Register serially is the contents of the Instruction Counter. The Instruction Counter controls the normal sequencing of the program through the memory. Just before a new instruction is required, the contents of the Instruction Counter are shifted into the Transfer Register and transferred in parallel to the Address Register in the Program Control Element. The Address Register then selects the memory location of the new instruction.

2-19. When the new instruction is read, it is transferred from the Memory Buffer Registers into the Transfer Register. The instruction is then separated into an operation code and an operand address; the two parts are transferred in parallel to the Program Control Element. The operation code is loaded into the Operation Code Register and the operand address into the Address Register. The operand address of some instructions constitutes a special constant which, in addition to the parallel transfer, is shifted out of the Transfer Register in serial form.

2-20. Program Control Element. The Program Control Element stores and decodes the programmed instructions and in addition, controls sequencing the program through the memory. The Operation Code Register and the Address Register store their respective parts of the instructions. Outputs of the Operation Code Register, and those of the Operation Decoders, control the Arithmetic and Multiply-Divide Elements, the Data and Memory Control Elements and parts of the Program Control Element to perform the commanded operations.

2-21. The Address Register selects the memory location of the data to be used in the commanded operation. When the operation is complete, the Address Register is reloaded with the contents of the Instruction Counter to address the next instruction. The Address Register is augmented by the Data Module Register and the Instruction Module Register (located in the Memory Control Element) and by the Data Sector Register and the Instruction Sector Register in the Program Control Element.

2-22. The module and sector registers preselect an area of the memory within which the program must operate. As implied by the names of the registers, two selections are made, one for data and one for instructions (the same area may be selected for both).



The Address Register then selects individual locations within the preselected areas from which data and instructions will be read. Timing signals discriminate between instruction and data addressing. The contents of the module and sector registers are changed upon command of the program.

2-23. It was noted previously that the Program Control Element controls sequencing the program through the memory. Actually, the program controls its own sequencing through the use of instructions which command the Program Control Element to change the contents of the module and sector registers (discussed in the preceding paragraph) and the Instruction Counter.

NOTE

Although the Instruction Counter is logically part of the Program Control Element, it is instrumented in the Arithmetic Element.

The Instruction Counter stores the address of the next instruction to be operated. Each time the computer performs an instruction, the instruction address is incremented by one to develop the address of the following instruction. In this manner, the program steps sequentially through the area of memory selected by the Instruction Module Register and the Instruction Sector Register.

2-24. The sequential stepping continues until the program issues an instruction to alter the sequence or select a different area of the memory to read instructions from. Area selection is changed by shifting a "HOP constant" out of the Transfer Register which reloads the module and sector registers. The same constant also reloads the Instruction Counter to select the starting location of the next instruction sequence. The Instruction Counter then begins stepping sequentially through the newly selected memory area beginning at the specified starting point.

2-25. The Instruction Counter can also be reloaded without affecting the module and sector registers. This feature permits repeating short program loops and enables the computer to make logical choices. When commanded, the computer can examine the contents of the Accumulator Register for certain conditions and alter the program sequence if the condition exists. If the condition is not met, sequential stepping continues.

2-26. The Program Control Element also contains two other circuits which can alter the normal program sequencing, the Start-Stop Control and the Interrupt Control circuits. The Start-Stop Control circuit provides the means to externally control starting and stopping the computer and to single-step through its program one instruction at a time. Application of the HALTV signal results in clearing certain registers to an initial condition and directs the computer to the first instruction of its program. During power application, the HALTV signal remains on until power is fully applied to prevent spurious instructions from being operated. The CSTN signal is manipulated by the test equipment to step through the program one instruction at a time, on command.

2-27. The Interrupt Control circuit enables external equipment to break off normal sequencing of the program and direct it to process data of a higher priority. The Interrupt Control circuit allows the instruction in progress at the moment of the interruption to finish, then forces execution of a command to reload the Instruction Counter and the module and sector registers.

2-28. The first few instructions of the new program sequence store the contents of important registers in order to preserve the conditions existing in the computer at the moment of interruption. Once the priority data has been processed, the computer can restore these conditions and resume operation where it left off. The very first item to be stored must be a "HOP constant" defining the location at which the interrupt occurred. This constant is produced by the HOP Constant Serializer and is used to redirect the program to the point of interruption.

2-29. Arithmetic and Multiply-Divide Elements. The Arithmetic and Multiply-Divide Elements encompass the machine's total computing ability. The Arithmetic Element performs addition, subtraction, shifts and logical extractions, and the Multiply-Divide Element performs the iterative processes of multiplication and division. The two elements are independent and can, if desired, be operated concurrently.

2-30. Once a multiply or divide operation has been started, it runs automatically to completion and the next few instructions of the program are operated during its progress. During this time the Arithmetic Element is available for concurrent use. There is, however, a program option for multiplication which stops the program until the product of the multiply is available. When using this option the product is placed in both the Product-Quotient Register and in the Accumulator Register.

2-31. The Accumulator Register in the Arithmetic Element provides one of the operands for all the arithmetic instructions and the memory provides the other. For multiplication and division, the accumulator contents and the operand from memory are simply transferred into shift registers in the Multiply-Divide Element for temporary storage during the iterative process. The remaining arithmetic operations combine the two operands in the Add-Subtract Logic and place the result in the Accumulator Register. The result can then be sensed to control logical decisions or it can be transferred to the memory, the Multiply-Divide Element or the data adapter. The result remains unchanged in the Accumulator Register after sensing or transfer.

2-32. The Multiply-Divide Logic senses the operands each iteration and forms a partial product or quotient. The final result is circulated in the Product-Quotient Register until another multiply or divide operation is initiated, or until the computer is commanded to store data in the Product-Quotient Register. Unlike the Accumulator Register which provides data to several destinations, the Product-Quotient Register has only one outlet for its contents. All data leaving the Product-Quotient Register must pass through the Accumulator Register. The Product-Quotient Register contents are accessible to all arithmetic instructions except multiply and divide.

2-33. The Voting Element comprises the voters and disagreement detectors located between modules of the computer and between the central computer and the memory. The voters resolve differences between the outputs of triple redundant circuits before the outputs are passed on to succeeding circuits. Thus the computer can sustain multiple failures (but not identical failures occurring simultaneously) and still function reliably. The disagreement detectors monitor inputs to the voters and provide outputs which indicate when a failure has occurred. Since failures are self-correcting, there are no failure indications for noncatastrophic malfunctions in the computer logic except those from the disagreement detectors. The Voting Element is not shown in figure 2-1 because it is widely disseminated through and between the functional elements.

2-34. Typical Operation. Assume that the computer program has been initiated and is running smoothly. The area of the memory in which the program will operate has been selected and instructions are coming from syllable zero. The memory is operating in the duplex mode with memory A selected. For descriptive purposes, typical operation

begins with reading an instruction from the memory. The address of the instruction is shifted out of the Instruction Counter and applied to the Transfer Register and the Add-Subtract Logic in the Arithmetic Element. The Add-Subtract Logic increments the count by one and reinserts it into the Accumulator Register where it is circulated. The instruction address is shifted into the Transfer Register and then transferred in parallel to the Address Register. The next events occur in the Memory Control Element.

2-35. Timing inputs indicate to the Syllable Select circuit and the Memory Timing and Sync Select circuit that an instruction is due to be read. The Syllable Select circuit selects syllable zero (the stored syllable selection for instructions) in all memory modules. The Memory Timing and Sync Select circuit performs three functions at this time. It conditions the Memory Address Decoders to decode the contents of the Instruction Sector Register rather than the Data Sector Register; it conditions the Memory Clock Drivers of all memory modules to produce read pulses; and it delivers a sync impulse to the Memory Mode and Module Select circuit. The Memory Mode and Module Select circuit routes the sync impulse to the Memory Clock Drivers of the two (duplex operation) memory modules selected by the Instruction Module Register.

2-36. Upon being impulsed, the Memory Clock Drivers produce read timing pulses which enable the selected X and Y Drivers to transfer the instruction out of the addressed memory location. The Memory Sense Amplifiers transmit the instruction to the Memory Buffer Registers. The Memory Buffer Registers store the instruction for parity checking and to enable the Inhibit Drivers during the restore memory cycle (explained later). If the parity of either Memory Buffer Register is incorrect or if a drive current failure occurred during the read operation, the Memory Select and Error Monitor circuit provides an error indication to the data adapter. Upon sensing an error, the Memory Select and Error Monitor circuit also places both sets of Inhibit Drivers under control of the Memory Buffer Register containing correct information and if the error is in the selected memory, changes the memory selection. The instruction is then transferred from the selected Memory Buffer Register to the Transfer Register.

2-37. Reading a ferrite-core memory destroys the information in the memory. Consequently, a read operation must always be followed by a store (or restore) operation to preserve the contents of the memory. The restore operation is accomplished after the instruction has been transferred to the Transfer Register. The Memory Timing and Sync Select circuit changes the conditioning level to the Memory Clock Drivers so that they produce store timing pulses, then delivers a second sync impulse to the Memory Mode and Module Select circuit. There have been no changes in address or memory module selection, therefore the same memory location is exercised again. However, on this memory cycle, data is transferred from the Memory Buffer Registers to the memory through the Inhibit Drivers.

2-38. Almost immediately upon entering the Transfer Register, the instruction is separated into an operation code and an operand address. The operation code defines what operation will be performed and the operand address gives the location in memory of the data to be used in the operation. The operation code is transferred in parallel to the Operation Code Register and the operand address to the Address Register. The remainder of the operation is dependent upon the code transferred to the Operation Code Register.

The code can specify any of three general types of operations: 1) those which require data from memory, 2) those which place data into the memory and 3) those which do not use the memory.

2-39. Assume first that the operation code requires data to be read from the memory. The Syllable Select circuit selects syllable zero, the first syllable of the data word. The Memory Timing and Sync Select circuit conditions the Memory Address Decoders to combine and decode the contents of the Address Register and the Data Sector Register; it then initiates a read cycle in the memory which transfers the first half of the data word to the Memory Buffer Registers. While in the Memory Buffer Registers, the half-data word is parity checked, then transferred to the Transfer Register and restored in the memory. The Transfer Register serializes the data by shifting it to the TRS output. The TRS output makes the data available to the Arithmetic and Multiply-Divide Elements, the Program and Memory Control Elements and the data adapter. Outputs from the Operation Code Register and the Operation Decoders determine which element will accept it and how it will be used.

2-40. As the first half of the data word nears the end of the Transfer Register, the Syllable Select circuit selects syllable 1 and the Memory Timing and Sync Select circuit initiates another read cycle in the memory. This read cycle transfers the second half of the data word into the Memory Buffer Registers where it is parity checked. Then, just in time to fall in behind the last bit of the first half data-word, the second half of the data word is transferred into the Transfer Register. The Transfer Register continues shifting and the two halves of the data word appear as a continuous serial output on the TRS line.

NOTE

To obtain the result of a multiply or divide operation, the operand address specifies the Product-Quotient Register. When the Product-Quotient Register is addressed, the memory is inhibited and the product or quotient is shifted directly into the Accumulator Register without passing through the Transfer Register.

2-41. When providing a data output, the Transfer Register is synchronized with the Accumulator Register. Thus, the output of the Transfer Register and the contents of the Accumulator Register can be combined bit-for-bit in the Add-Subtract Logic. The results of these operations are placed in the Accumulator Register for recirculation. As the last bit of the accumulator contents emerges from the Accumulator Register, timing signals end the arithmetic processes in the Add-Subtract Logic and the contents of the Instruction Counter begin to emerge from the Accumulator Register. The Add-Subtract Logic increments the instruction count by one and reinserts it in the Accumulator Register immediately behind the result of the arithmetic operation. Simultaneously, the instruction count is shifted into the Transfer Register to address the next instruction.

2-42. When the operation code specifies that data is to be transferred into the memory, the basic operations are reversed. Instead of transferring two syllables of data to the Transfer Register which serializes them into a continuous unit, a continuous unit of serial data is shifted into the Transfer Register which divides it into syllables. The syllables are then transferred to the memory in parallel. Shortly after the instruction is loaded into the Operation Code and Address Registers the data word to be stored begins to emerge from the Accumulator Register. Outputs from the Operation Code Register

and the Operation Decoders condition the Transfer Register to shift in step with the Accumulator Register and accept its output. As the data word enters the first position of the Transfer Register, the Parity Counter monitors the number of "1's" it contains. At this point, the focus moves to the Memory Control Element.

2-43. Timing signals indicate to the Memory Control Element that a data word is due to be transferred to or from the memory. The operation code specifies that data is going to the memory by causing the Memory Timing and Sync Select circuit to switch the Read/Store signal to Store. The Syllable Select circuit selects syllable zero as the first syllable to be transferred. The Memory Timing and Sync Select circuit conditions the Memory Address Decoders to enable the X and Y Drivers of the memory location addressed by the Address Register and the Data Sector Register. Just before the last bits of the syllable enter the Transfer Register, the Memory Timing and Sync Select circuit delivers a sync impulse to the Memory Mode and Module Select circuit. The Memory Mode and Module Select circuit routes the impulse to the Memory Clock Drivers of the selected data modules. The Memory Clock Drivers produce read timing pulses, but the data transferred out of the memory is not sensed because a store, not read, operation has been specified. Since reading the memory is a destructive process, the addressed memory location is left cleared.

2-44. Simultaneous with clearing the memory location, the last bit of the syllable enters the Transfer Register and the syllable is transferred to the Memory Buffer Registers. The Transfer Register continues shifting and the second syllable of data starts to file into the Transfer Register. At the Memory Buffer Registers, a "parity bit" is added to the syllable entering the memory. The parity bit is controlled by the Parity Count circuit so that each Memory Buffer Register contains an odd number of "1's." (Bear in mind that the Parity Count circuit monitors the number of "1's" in the syllable as it enters the Transfer Register.) The Memory Timing and Sync Select circuit then impulses the memory again and the first syllable of data enters the memory through the Inhibit Drivers. The second syllable of data enters the memory in the same manner as the first and like the first includes its own parity bit.

NOTE

It was noted previously that data could be stored in the Product-Quotient Register. When this operation is specified, the memory is inhibited and the accumulator contents are shifted directly into the Product-Quotient Register in one intact unit.

2-45. Operations which do not use the memory either shift the contents of the Accumulator Register or use the operand address as a constant for controlling the program. Shifting the contents of the Accumulator Register is accomplished by shortening or lengthening its circulation loop. Before the first bits of the accumulator contents begin to emerge, outputs from the Operation Code Register and the Operation Decoders, along with control bits from the Address Register, alter the recirculation path of the Accumulator Register to affect the shift. When the last bit has been shifted, timing signals restore the loop to normal to prevent shifting the contents of the Instruction Counter. The Instruction Counter is then incremented and shifted into the Transfer Register to address the next instruction.

2-46. Among the operations which do not use the memory are those which make logical decisions based on the contents of the Accumulator Register. As noted previously, logical decisions are made by altering the contents of the Instruction Counter if a specified condition exists in the Accumulator Register. If the condition exists, the operand address of the instruction replaces the existing contents of the Instruction Counter and the next instruction begins the new sequence. Logical decisions are implemented by continuously sampling the accumulator contents part way through the register for the conditions upon which decisions are based. Thus, when a logical decision is initiated, most of the accumulator contents have already been sampled and the computer need not wait until the complete contents emerge from the register to make the decision. The contents of the Instruction Counter immediately follow the accumulator contents in the Accumulator Register. Consequently, when the last data bit has been sensed and the decision made, the contents of the Instruction Counter are still only part way through the Accumulator Register. If the condition is met, the Accumulator Register is broken just behind the last bit of the accumulator contents and the operand address is shifted into the Accumulator Register from the Transfer Register, replacing the existing instruction count.

NOTE

Since the memory is not being used, the Transfer Register is not needed to distribute data from it, and therefore is not cleared after the instruction is distributed to the Program Control Element.

If the condition is not met, the Accumulator Register is not broken and the existing contents of the Instruction Counter are retained to address the next instruction.

2-47. WORD ORGANIZATION.

2-48. The computer uses a 28-bit word consisting of two 14-bit syllables. Each syllable includes 13 data bits and one parity bit (figure 2-2) and each syllable is stored separately in the memory. The parity bits are used only to check the accuracy of transfers to and from the memory which leaves a data word 26 bits in length.

BIT POSITION	SYLLABLE 0													SYLLABLE 1															
	1	2	3	4	5	6	7	8	9	10	11	12	13	PAR	1	2	3	4	5	6	7	8	9	10	11	12	13	PAR	
NUMERIC DESIGNATION	13	14	15	16	17	18	19	20	21	22	23	24	25		S	1	2	3	4	5	6	7	8	9	10	11	12		
NON-NUMERIC DESIGNATION	14	15	16	17	18	19	20	21	22	23	24	25	26			1	2	3	4	5	6	7	8	9	10	11	12	13	

Figure 2-2. Data Word Layout

2-49. The use of the data word is at the discretion of the programmer. Each of the 26 bits may be used as an indicator to show the presence or absence of some condition, or the word may be used as a binary number for arithmetic computations. Binary numbers are represented by a sign bit and 25 magnitude bits; negative numbers are shown in two's complement form.

2-50. This manual uses two methods for designating bit positions in the data word. Where the word represents a binary number, the bits are designated Sign and 1 through 25, proceeding toward the least significant digit (LSD). When used in this context, bits

1 through 25 are called "magnitude bits." When referencing the data word not as a binary number, the bits are designated 1 through 26; bit 1 corresponds to Sign and bit 26 corresponds to bit 25 of the numeric designations.

2-51. The computer word is also used to store the instructions of the computer program. Instructions are 13 bits in length with one instruction stored in each syllable of the computer word. An instruction consists of a 4-bit operation code (OP1-OP4) and a 9-bit address (A9, A1-A8). Figure 2-3 shows how the instructions are placed in the computer words.

BIT POSITION	SYLLABLE 0													SYLLABLE 1														
	1	2	3	4	5	6	7	8	9	10	11	12	13	PAR	1	2	3	4	5	6	7	8	9	10	11	12	13	PAR
DESIGNATION	A	A	A	A	A	A	A	A	A	O	O	O	O		A	A	A	A	A	A	A	A	A	O	O	O	O	
	8	7	6	5	4	3	2	1	9	4	3	2	1		8	7	6	5	4	3	2	1	9	4	3	2	1	
USE	OPERAND ADDRESS									OPERATION CODE					OPERAND ADDRESS									OPERATION CODE				

Figure 2-3. Instruction Word Layout

2-52. TIMING ORGANIZATION.

2-53. The basic unit of computer timing is called the "computer cycle." The duration of a computer cycle is approximately 82 microseconds; this is the time required for the computer to read, decode and operate its basic instructions. The signals which define a computer cycle originate in the Phase Generator. A computer cycle corresponds to a full cycle of the Phase Generator which produces three equal-length timing signals called: phase A (PA), phase B (PB) and phase C (PC). Thus, a computer cycle is defined by three "phase times."

2-54. The Phase Generator separates the computer cycle into "instruction time" and "operation time." As the names imply, instruction time is the period in which an instruction is read and decoded, and operation time is the period in which the operation commanded by the instruction is performed. Generally, instruction time occurs during phase A and operation time during phase times B and C.

2-55. Notice the correlation between the organization of timing and word layout. The computer cycle is defined by three phase times and involves three syllables of information: one syllable for an instruction and two syllables of data for an operand. The correlation also extends downward one step. A syllable consists of 14 bits (13 data, 1 parity) and each phase time is divided into 14 equal segments called "list times." A bit time is the time each bit of data is stored in one position of a serial storage device before moving on to the next.

2-56. To facilitate several logic operations during each bit time, the bit times are subdivided into four equal parts called clocks. The clocks are identified by the letters W, X, Y and Z and occur in alphabetical sequence. Figure 2-4 shows the relationships of the various levels of timing and gives the time duration of each level. In this manual, timing is given by a letter-number-letter abbreviation of the phase time, bit time and clock time in descending order of time duration. For example, an event resulting from the W clock of bit-time 11 during phase A is described as happening at A-11-W time. If one level of timing does not apply or has been previously established, that level may be dropped from the abbreviation; i. e., a circuit which operates independent of phase timing may produce an output at every 6-Z time.

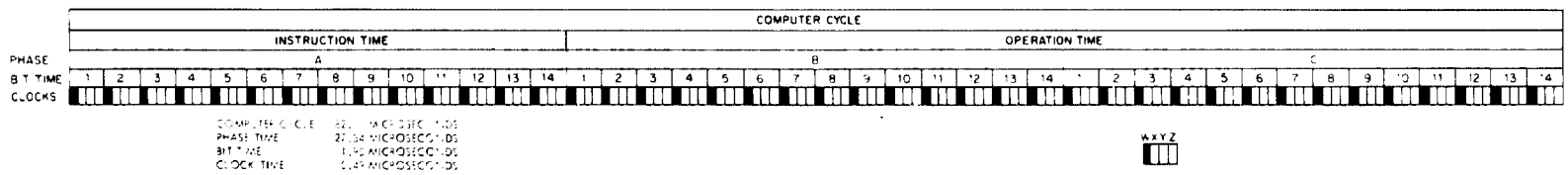


Figure 2-4. Computer Timing Organization

2-57. INSTRUCTION ORGANIZATION.

2-58. The computer uses a complement of 18 single address instructions which are composed of a 4-bit operation code and a 9-bit operand address. The 4-bit operation code can select one of 16 different instructions to be executed; this range is extended to 18 by grouping three instructions under one operation code, then using bits A8 and A9 of the address to discriminate between them. Bits A8 and A9 serve no other function for the instructions which are grouped. Figure 2-5 is a map of the operation codes showing the names of the instructions assigned to each code. Figure 2-6 shows how addition of each bit of the operation code narrows the field of selection by de-selecting half the remaining area of the map. A list of the 18 instructions available is given in figure 2-8 with the operation code and a brief description of each.

2-59. The 9-bit operand address, figure 2-7, permits selection of 512 memory addresses for use as operands or data storage locations. The memory is divided into a number of sectors, each containing 256 addresses. Address bits A1 through A8 select one of the addresses within a sector. Bit A9 determines whether the address will be in a sector previously selected by the program or in a special sector called "residual memory." Consequently, bit A9 is called the residual bit; residual memory is selected when A9 is a "1."

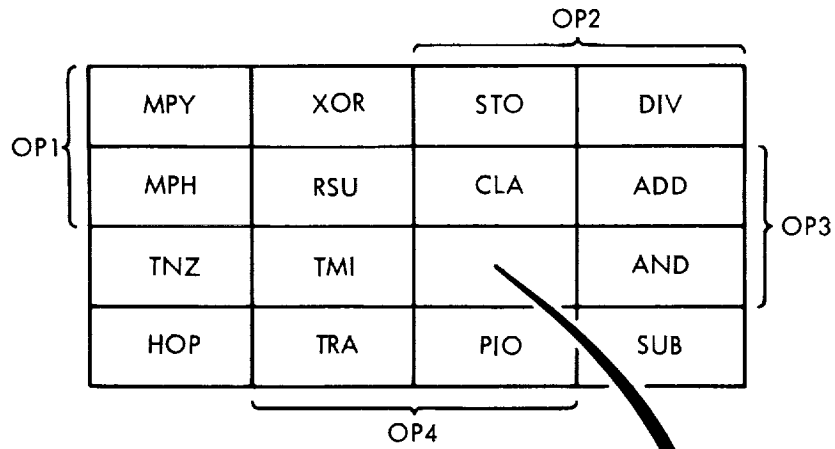
2-60. Instructions which do not require that data be read from the memory frequently use the operand address for special purposes. These special purposes are pointed out in the List of Instructions, figure 2-8, where they occur. For example the shift instructions (SHF) always manipulates the contents of the accumulator and thus requires no operand address; this frees the operand address for use as shift control. One instruction which makes extensive use of its operand address is PIO. A complete breakdown of the PIO addresses is given in figure 2-9.

2-61. COMPUTER LOGIC CIRCUITS.

2-62. TIMING ELEMENT.

2-63. The timing element contains all the circuits which generate timing signals for the operation of computer logic. (Memory timing circuits are described under the memory control element.) Basically the timing element consists of an oscillator and three cascaded frequency dividers: the clock generator (which includes the oscillator), the timing-gate generator, and the phase generator. The oscillator produces the basic timing signal, from which all other timing signals are derived. The clock generator divides its oscillator output frequency by four to produce a sequence of four clock signals which recur each bit time. The clock signals divide each bit time into four parts, enabling several logic operations to be performed on each serial data bit. The signals used to identify bit times are developed by the timing-gate generator. The timing-gate generator divides the clock frequency by seven, producing seven timing gates which, with their complements, can be combined to identify fourteen bit times. The phase generator then divides the output frequency of the timing-gate generator by three to define the three phases of the computer's operation cycle. Several additional signals, embodying various combinations of phase, timing gate, and clock signals are generated by special timing circuits.

2-64. **CLOCK GENERATOR.** The clock generator consists of four parts: an oscillator, a buffer-amplifier, timing logic and clock drivers (figure 2-10). (All parts except the oscillator are triple redundant; therefore, the following description applies to all channels.) The clock generator produces a repetitive sequence of four clock pulses (W, X,



INSTRUCTION	A8	A9
CDS	X	0
SHF	0	1
EXM	1	1

Figure 2-5. Operation Code Map

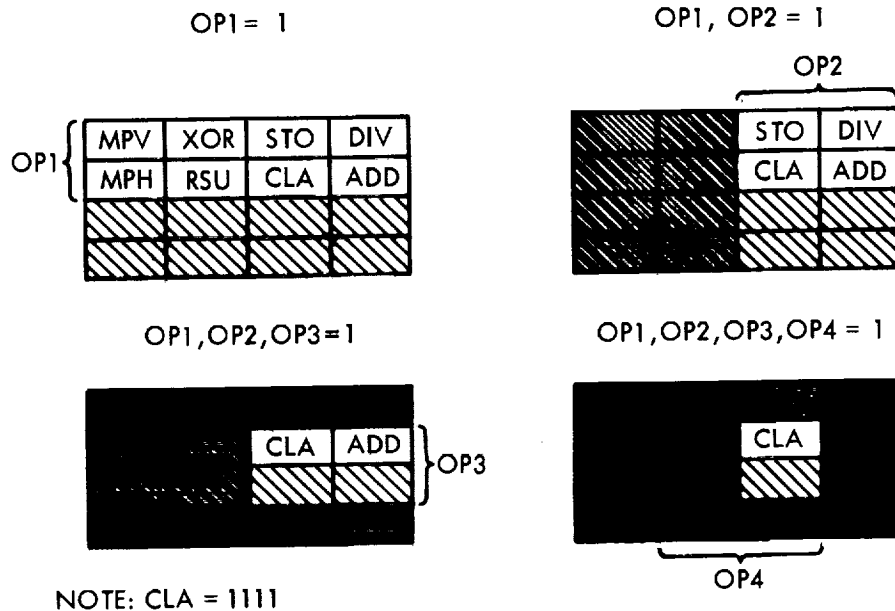


Figure 2-6. Selecting An Instruction

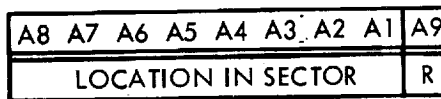


Figure 2-7. Operand Address Layout

Y, and Z) which are used to time logic operations in the computer. A special clock called BON is generated for driving the delay lines. The logic clocks occur at a repetition rate of 512 KC and the delay-line clock at 2.048 MC.

2-65. The oscillator produces a 2.048 MC sine wave which is used, ultimately, to generate all other computer timing signals. The sine wave is amplified and shaped in the buffer-amplifier which produces a complementary pair of square wave outputs, BO and BON. The BON signal is fed to the delay lines and the BO signal to the timing logic where it is frequency divided to generate signals which separate the four clocks.

2-66. The timing logic produces three pairs of signals to separate the clocks. One of the signal pairs is at one-half the frequency of BO; the remaining two pairs, 90 degrees apart, are at one-fourth the frequency of BO. Combinations of these signals are applied to the clock drivers to produce the clocks in the proper sequence. Each clock driver produces ten outputs; a clock signal for use in the data adapter (WD), a "not clock" (WN), and eight isolated outputs for the actual clock signal (W1 through W8).

2-67. Oscillator. A modified Pierce oscillator is used to generate the 2.048 MC sine wave which is the basic timing signal for the clock generator. The oscillator contains only six components and draws all its operating current from the buffer amplifier. The low component count and absence of a supply-voltage input greatly enhance the oscillator's reliability.

INSTRUCTION	OPERATION CODE				DESCRIPTION
	4	3	2	1	
HOP	0	0	0	0	Transfers program to memory location specified by HOP constant and controls simplex/duplex operation of memory. Operand address specifies memory location of HOP constant used to load the registers shown in HOP constant format (below). Full HOP constant MUST be specified each time. First instruction following HOP comes from new location.
MPY	0	0	0	1	Multiplies contents of memory location specified in operand address by contents of accumulator. Uses 24 high order bits of each operand to form 26-bit product. Product of multiplicand and 12 low order bits of accumulator is available by addressing 775 with second instruction following MPY; final product is available to fourth instruction following MPY. Program continues while MPY is in progress; concurrent use of accumulator is permitted.
SUB	0	0	1	0	Subtracts contents of memory location specified in operand address from contents of accumulator. Places remainder in accumulator.
DIV	0	0	1	1	Divides contents of memory location specified by operand address into contents of accumulator. The 24-bit quotient is available to the 8th instruction following divide by addressing the P-Q register (775). Program continues while DIV is in progress; concurrent use of accumulator is permitted.

Figure 2-8. List of Instructions (Sheet 1 of 5)

INSTRUCTION	OPERATION CODE	DESCRIPTION
	4 3 2 1	
TNZ	0 1 0 0	Conditional transfer. Transfers operand address (A1-A8) to instruction counter and A9 to syllable select <u>if accumulator contents are not zero</u> . Next instruction comes from new syllable and address. If accumulator is zero, perform next instruction in sequence.
MPH	0 1 0 1	Multiplies contents of memory location specified in operand address by contents of accumulator. Uses 24 high-order bits of each operand to form 26-bit product. Holds up program until multiplication is complete. Product is available from accumulator or P-Q register with instruction following MPH.
AND	0 1 1 0	AND's contents of memory location specified in operand address with contents of accumulator. Result is placed in accumulator.
ADD	0 1 1 1	Adds contents of memory location specified in operand address to contents of accumulator. Sum is placed in accumulator.
TRA	1 0 0 0	Unconditional transfer. Transfers operand address (A1-A8) to instruction counter and A9 to syllable select. Next instruction comes from new syllable and address.
XOR	1 0 0 1	Exclusive - OR contents of memory location specified in operand address with contents of accumulator, bit-for-bit. When accumulator bit and bit from memory are different a "1" is placed in the corresponding bit of the accumulator; if accumulator bit and memory bit are alike, a "0" is placed in the accumulator.
PIO	1 0 1 0	Process input or output. Operand address specifies input or output and gives address of data source and destination. See figure 2-9 for listing of addresses.
STO	1 0 1 1	Contents of accumulator is stored in memory location specified by operand address. Contents of accumulator is unchanged. Data can be stored in P-Q register by using address 775. A STO instruction with operand address 776 or 777 causes contents of multiplicand register to be stored in memory location 776 or 777 as specified. These addresses are used for HOP-save feature.

Figure 2-8. List of Instructions (Sheet 2)

INSTRUCTION	OPERATION CODE	DESCRIPTION										
	4 3 2 1											
TMI	1 1 0 0	Conditional transfer. Transfer operand address (A1-A8) to instruction counter and A9 to syllable select <u>if accumulator sign is minus</u> . Next instruction comes from new syllable and address. If accumulator sign is plus, perform next instruction in sequence.										
RSU	1 1 0 1	Reverse subtract. Contents of accumulator are subtracted from contents of memory location specified by operand address. Remainder is placed in accumulator.										
SHF	1 1 1 0 A8 = 0 A9 = 1	Contents of accumulator are shifted MSD or LSD a maximum of two bit positions as specified by operand address. Shift control codes are: <table style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Address Bit</th> <th>Shift</th> </tr> </thead> <tbody> <tr> <td>A1 = 1</td> <td>LSD 1</td> </tr> <tr> <td>A2 = 1</td> <td>LSD 2</td> </tr> <tr> <td>A5 = 1</td> <td>MSD 1</td> </tr> <tr> <td>A6 = 1</td> <td>MSD 2</td> </tr> </tbody> </table> <p>Clears accumulator if all shift control bits are "0's".</p>	Address Bit	Shift	A1 = 1	LSD 1	A2 = 1	LSD 2	A5 = 1	MSD 1	A6 = 1	MSD 2
Address Bit	Shift											
A1 = 1	LSD 1											
A2 = 1	LSD 2											
A5 = 1	MSD 1											
A6 = 1	MSD 2											
CDS	1 1 1 0 A9 = 0	Change data sector. Operand address is used as a constant to load registers indicated below.										

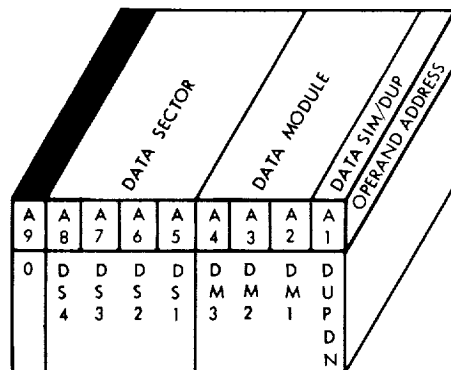


Figure 2-8. List of Instructions (Sheet 3)

INSTRUCTION	OPERATION CODE 4 3 2 1	DESCRIPTION																																																	
EXM	1 1 1 0 A8 = 1 A9 = 1	<p>Execute modified. Operand address selects one of 8 instructions in residual memory to be the next instruction executed and modifies its operand address as shown below prior to execution.</p> <div style="text-align: center;"> <table border="1" data-bbox="909 630 1169 724"> <tr><th colspan="8">EXM OPERAND ADDRESS</th></tr> <tr><th>A</th><th>A</th><th>A</th><th>A</th><th>A</th><th>A</th><th>A</th><th>A</th></tr> <tr><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2 1</td></tr> </table> </div> <p data-bbox="706 766 893 882">PART OF EXM OPERATION CODE, A9 SELECTS RESIDUAL MEMORY FOR NEXT INSTRUCTION</p> <p data-bbox="1218 745 1429 798">REPLACE BITS A1 & A2 IN NEXT INSTRUCTION</p> <p data-bbox="1218 808 1461 850">ORd WITH BITS A3 & A4 OF NEXT INSTRUCTION</p> <p data-bbox="1218 861 1412 903">SYLLABLE SELECT FOR NEXT INSTRUCTION</p> <p data-bbox="1218 913 1453 955">SELECT 1 OF 4 ADDRESSES FOR NEXT INSTRUCTION</p> <table data-bbox="1201 976 1453 1113"> <tr> <td>A7</td> <td>A6</td> <td>ADD</td> <td>A5</td> <td>SYL</td> </tr> <tr> <td>0</td> <td>0</td> <td>600</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>640</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>700</td> <td></td> <td></td> </tr> <tr> <td>1</td> <td>1</td> <td>740</td> <td></td> <td></td> </tr> </table>	EXM OPERAND ADDRESS								A	A	A	A	A	A	A	A	9	8	7	6	5	4	3	2 1	A7	A6	ADD	A5	SYL	0	0	600	0	0	0	1	640	1	1	1	0	700			1	1	740		
EXM OPERAND ADDRESS																																																			
A	A	A	A	A	A	A	A																																												
9	8	7	6	5	4	3	2 1																																												
A7	A6	ADD	A5	SYL																																															
0	0	600	0	0																																															
0	1	640	1	1																																															
1	0	700																																																	
1	1	740																																																	

Figure 2-8. List of Instructions (Sheet 4)

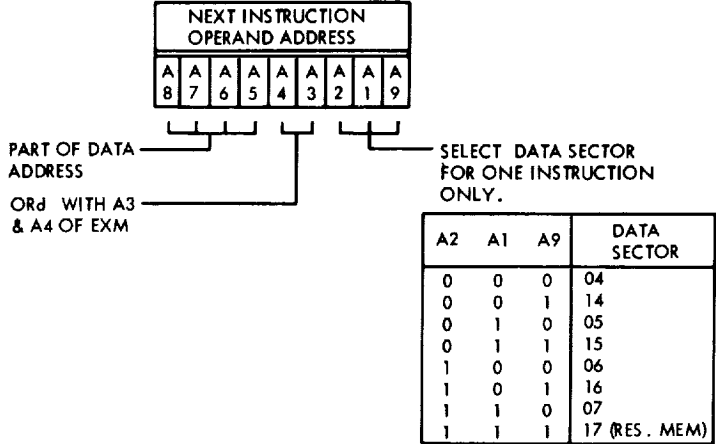
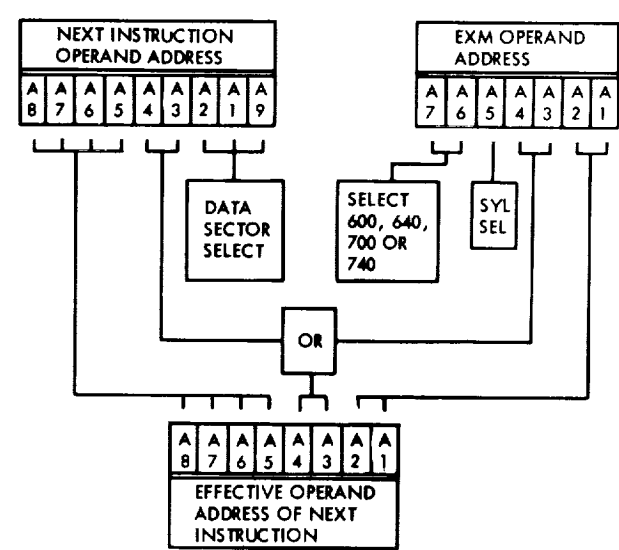
INSTRUCTION	OPERATION CODE 4 3 2 1	DESCRIPTION
EXM (cont)	1 1 1 0 A8 = 1 A9 = 1	<div style="text-align: center;">  <p style="text-align: center;">EXM operation is diagrammed below.</p>  </div> <p style="text-align: center;">Contents of memory location specified by the operand address are transferred to cleared accumulator.</p>
CLA	1 1 1 1	Contents of memory location specified by the operand address are transferred to cleared accumulator.

Figure 2-8. List of Instructions (Sheet 5)

GROUP SELECTION AND FUNCTIONS										
Group	Data Source	Data Destination	A9	A8	A7	A6	A5	A4	A3	A2 A1
1 & 3	LVDC Accumulator	LVDA Telemetry Registers	X	0	┌ Address ─┐				0	X
1 & 3A	LVDC Main Memory	LVDA Telemetry Registers	0	1					0	X
1 & 3B	LVDC Residual Memory	LVDA Telemetry Registers	1	1					0	X
2	LVDC Accumulator	LVDA Output Registers	X	0					1	0
2A	LVDC Main Memory	LVDA Output Registers	0	1					1	0
2B	LVDC Residual Memory	LVDA Output Registers	1	1					1	0
4	LVDA Peripheral Inputs and Errors	LVDC Accumulator	X	0					1	1
5	LVDA Resolver Processor Inputs	LVDC Accumulator	X	1		└───┘			1	1
GROUP ADDRESSES										
Group 2		Group 4		Group 5			A7	A6	A5	A4 A3
Mode Register				Spare No. 6			0	0	0	0 0
Discrete Output Register (Reset)							0	0	0	0 1
Discrete Output Register (Set)				Computer COD Counter Start			0	0	0	1 1
Internal Control Register (Set)		Error Monitor Register		Fine Gimbal No. 1			0	0	1	0 0
Internal Control Register (Reset)							0	0	1	0 1
Interrupt Register Reset				Coarse Gimbal No. 3			0	0	1	1 0
Switch Selector Register (Load)				Computer COD Counter Start			0	0	1	1 1
Orbital Checkout		Command Receiver or RCA-110		Coarse Gimbal No. 1			0	1	0	0 0
Switch Selector & Discrete Output Registers (Read)		Discrete Input Spares		Horizon Seeker No. 1			0	1	0	0 1
				Spare No. 3			0	1	0	1 1

Figure 2-9. PIO Addresses (Sheet 1 of 2)

GROUP ADDRESSES								
Group 2	Group 4	Group 5	A7	A6	A5	A4	A3	
Switch Selector Interrupt Counter	Telemetry Scanner		0	1	1	0	0	
COD Error (Read)			0	1	1	0	1	
Inhibit Interrupt	Switch Selector	Spare No. 4	0	1	1	1	0	
Minor Loop Timed Interrupt Counter			0	1	1	1	1	
	Real Time	Fine Gimbal No. 4	1	0	0	0	0	
	Accelerometer Processor X		Spare No. 1	1	0	0	0	1
	Accelerometer Processor Z		1	0	0	1	0	
	Accelerometer Processor Y	Horizon Seeker No. 3	1	0	1	0	0	
		Horizon Seeker No. 2	1	0	1	0	1	
		Coarse Gimbal No. 4	1	0	1	1	0	
		Spare No. 5	1	0	1	1	1	
Ladder No. 1		Coarse Gimbal No. 2	1	1	0	0	0	
		Fine Gimbal No. 3	1	1	0	0	1	
		Spare No. 2	1	1	0	1	0	
Ladder No. 2		Fine Gimbal No. 2	1	1	1	0	0	
		Horizon Seeker No. 4	1	1	1	0	1	
			1	1	1	1	0	
			1	1	1	1	1	
Ladder No. 3								
Ladder No. 4								
Ladder No. 5								

Figure 2-9. PIO Addresses (Sheet 2)

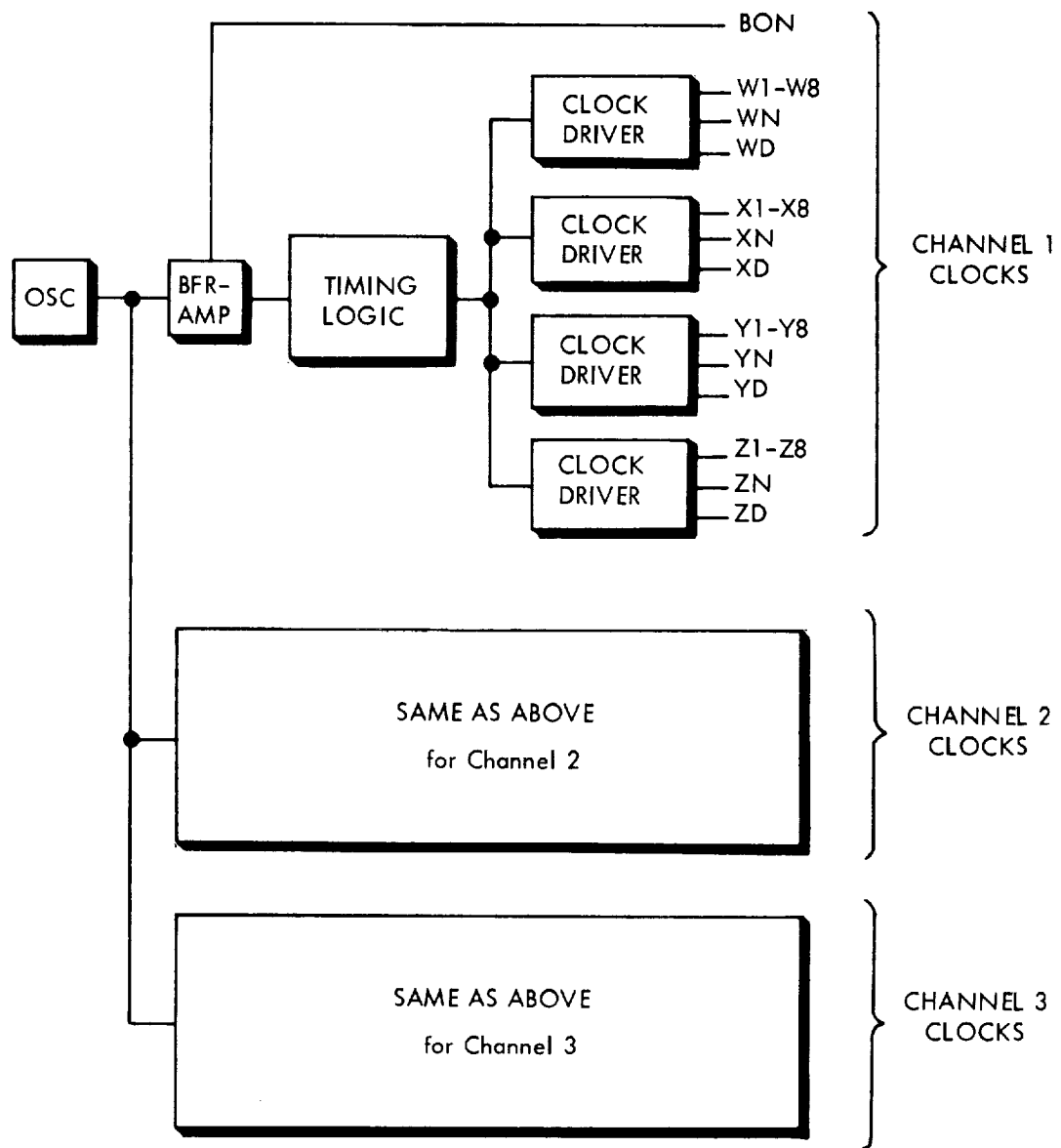


Figure 2-10. Clock Generator Block Diagram

2-68. Buffer-Amplifier. The output from the oscillator feeds the buffer-shaper circuit (BFR SHP) in each section of the buffer-amplifier (figure 2-11). The buffer-shaper converts the oscillator sine wave to a square wave and isolates the oscillator. The output from the buffer-shaper is fed through two inverter amplifiers in series (IA1 and IA2). These inverters provide level shifting and isolate the buffer-shaper from the load. The output of IA2 is essentially the inverse of the buffer-amplifier output, BO. The voter inverter (VI) operates as a high speed inverter to produce the BO output. The buffer power-amplifier (BFR PA) provides the fast transition times and special levels necessary for BON to drive the delay lines.

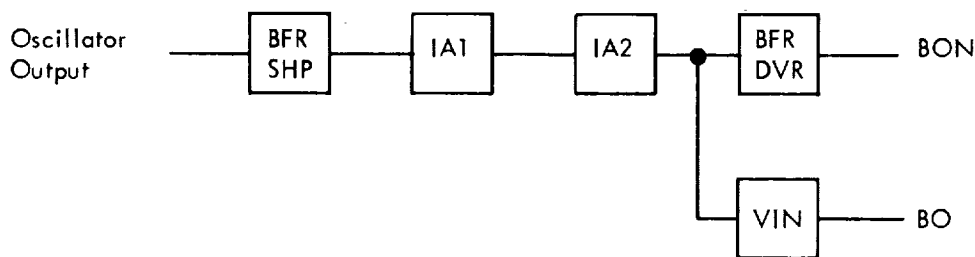


Figure 2-11. Buffer-Amplifier

2-69. Timing Logic. The timing logic consists of two pairs of latches (figure 2-12) which frequency divide the BO output of the buffer-amplifier by two and by four. The first pair is comprised of the S and P latches and the second pair is comprised of the R and Q latches which are described in a subsequent paragraph.

2-70. The S and P latches form a ring which divides the BO frequency by two. The S latch changes its state each time BO is a "1"; the P latch changes state when BO is a "0". (Details of driving the P latch are given in a subsequent paragraph.) The set and reset outputs of the S latch are fed through voters to the corresponding inputs of the P latch so that, when BO is a "0", the P latch is driven to the same state as the S latch. To complete the ring, the P-latch outputs are returned to the S-latch inputs but cross connected so that the P-latch set and reset outputs feed the S-latch reset and set inputs, respectively. Thus, when BO is a "1", the S latch is driven to the state opposite that of the P latch. Now it can be shown how the P latch changes state when BO becomes a "0".

2-71. Since the state of the S latch is opposite that of the P latch, both P-latch input gates are disabled. For example, if the S latch were set and the P latch reset, SVN would disable one gate and P would disable the other. Thus, when BO changes to a "0", the P latch is driven to a double "1" output condition. The double "1" output then permits the set gate to open and present a "1" drive to the set input of the P latch. The "1" drive over-rides the double zero drive and sets the P latch. When BO becomes a "1" again, the S latch is set opposite the P latch and so the process continues with each latch changing state every complete cycle of BO. Outputs of the P latch are fed through emitter followers before being used by the clock driver; S-latch outputs are used only at the P latch.

2-72. The R and Q latches make up a ring which is identical to the S-P ring except that both latches are under control of the P latch and both change state when BO is a "1". Both R-latch inputs are gated by the set output of the P latch and both Q-latch inputs are gated by the P-latch reset output. Thus, the R latch can change state only when the P latch is set and the Q latch can change state only when the P latch is reset. This action divides the P-latch output frequency by two, reducing the R- and Q-latch output frequency to one-fourth that of the BO output. The R and Q latches change state on opposite levels of a symmetrical signal; thus, their outputs are 90 degrees apart. Outputs of the R and Q latches are fed through emitter followers before being used by the clock drivers. The timing logic also contains emitter followers which boost the power of BO before it is sent to the clock drivers. Power amplification is indicated by adding the letter P to the signal name.

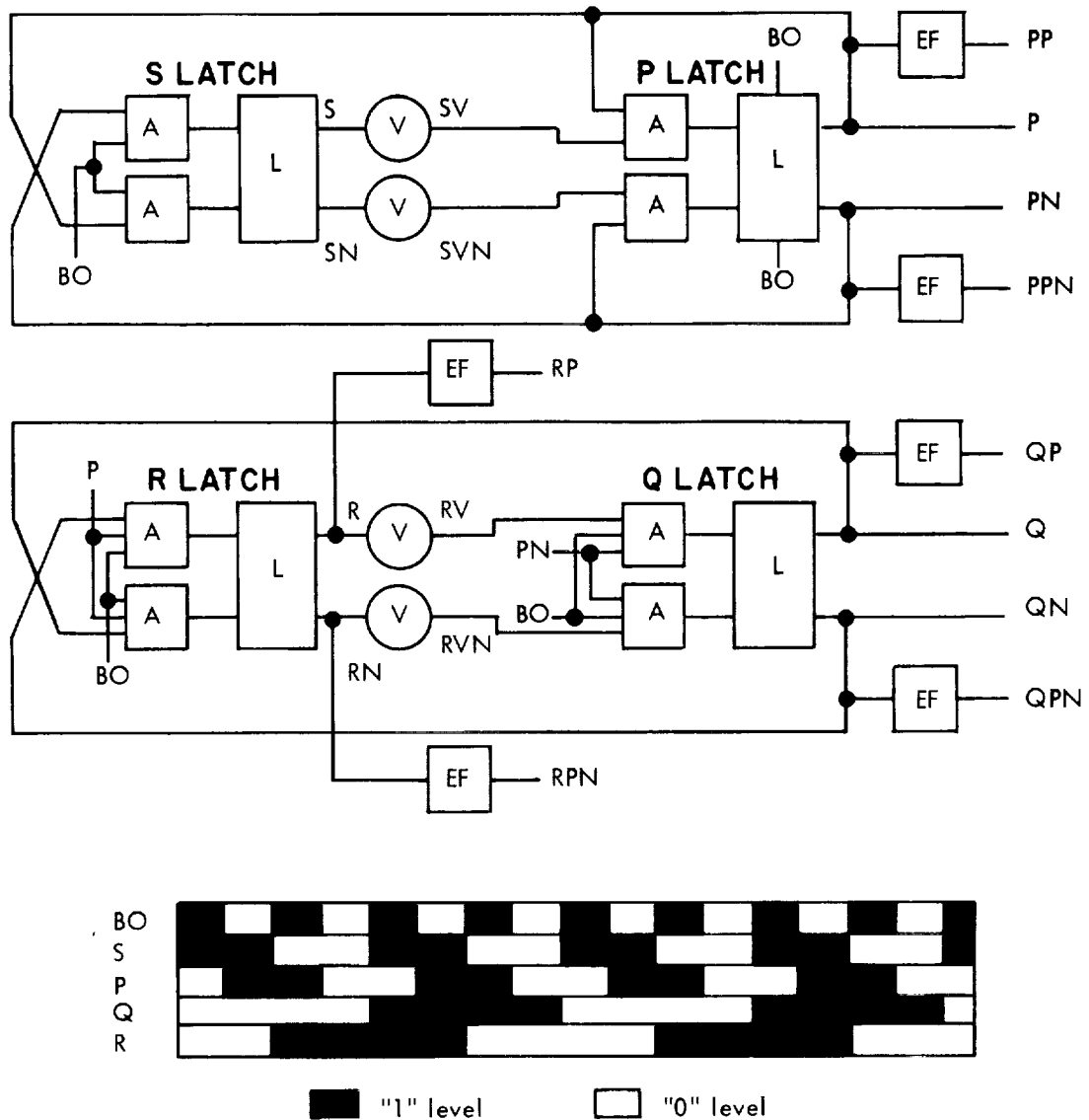


Figure 2-12. Timing Logic

2-73. Since the clocks are used to time logic operations which must then be voted on, the three clock generators must all produce the same clock at the same time. This is accomplished by synchronizing the three sections of timing logic. Each two-latch ring is synchronized with its counterpart in the other two sections of timing logic by the voter circuits between the two latches of the ring. For example, the S-P-latch rings are synchronized in the following manner.

2-74. Assume that, on application of power, two of the S latches come on in the set state and one in the reset state. Since the output of a voter represents the majority of its inputs, the voter in the out-of-sync ring indicates that the S latch in that ring is set when it is not. When BO next goes to a "0", the P latch is set in the normal manner. (Should the P latch already be set, the set input-gate would open and over-ride the double zero

drive to hold the latch set.) In the remaining two rings the same process occurs which results in all three P latches being set. When BO switches to a "1", the two S latches which were set are reset and the third S latch remains reset. Thus, the S-P-latch rings are completely synchronized in one full cycle of the BO signal. The Q-R-latch rings are synchronized in precisely the same manner. Once the P latches are synchronized, the Q and R latches require one full cycle of the P-latch outputs to come in step.

2-75. Clock Drivers. Each redundant section of the clock generator contains four clock drivers (figure 2-13), one for each of the clocks (W, X, Y and Z). Each clock driver consists of a nonsaturating inverter (NSI) with four clock gates (CG), a series of complementary emitter followers (CEF) and an inverter (INV). At each clock driver, the four clock-gate outputs feed the nonsaturating inverter input in common; the nonsaturating inverter then inverts its input and feeds a series of complementary emitter followers to produce the clock signal. A "1" on the common nonsaturating inverter input takes precedence over "0's"; therefore, any clock gate which is enabled holds the clock signal at "0". Thus, the clock-driver output is a "1" when the clock gates all produce "0's" in coincidence.

2-76. Signals generated by the timing logic and feedback signals from the other clock drivers are applied to the clock gates in the combinations necessary to produce the clocks in the proper sequence (figure 2-14). The feedback signals also prevent overlap from one clock to the next. At each clock driver, the last signal switched to complete the coincidence of "0's" is the feedback signal from the preceding clock. Thus, no clock can become a "1" until after the preceding clock has been switched to "0".

2-77. The first clock gate enabled to break the coincidence of "0's" is the one conditioned by BOP and the P latch. The P latch changes state when BO is a "0". Therefore, the clock gate is partially enabled before BOP becomes a "1". Since the R and Q latches change state when BO becomes a "1", circuit delays prevent their outputs from enabling a clock gate before BOP. Thus the clocks are controlled by the basic oscillator output and displacement between the clocks of the three redundant sections is held to a minimum.

2-78. TIMING GATE GENERATOR. The timing gate generator consists of a control circuit and a seven-bit modified shift register (figure 2-15). The shift register shifts its contents serially one place each bit time and various combinations of its outputs are sampled in parallel to identify bit times. Position one of the shift register is gated such that initially, it will force the register to all "1's" or all "0's" within seven bit times (gating not shown in figure 2-15). Voters at the outputs of each latch in the register synchronize the three timing gate generators (for TMR computers) in the same manner as the clock generator timing logic. (See paragraph 2-74.)

2-79. The register shifts toward position seven which is returned to position one through the complementing circuit. Thus, the shift register continuously cycles a series of seven "0's" followed by seven "1's", repeating itself every fourteen bit times. The rate at which the shift register is cycled is determined by the control circuit which is a binary counter driven by the Y clock.

2-80. Control Circuit. The control circuit consists of a pair of latches interconnected to form a binary counter (figure 2-16). The AB latch stores the configuration of the A latch during W clock and steers the A latch to its complement state on the following Y clock. Thus the A latch is set and reset during alternate bit times. The control circuits in the three channels are synchronized by the voters at the outputs of the A latches. The synchronization takes place in the same manner as synchronization of the clock generator timing logic. (See paragraph 2-74.)

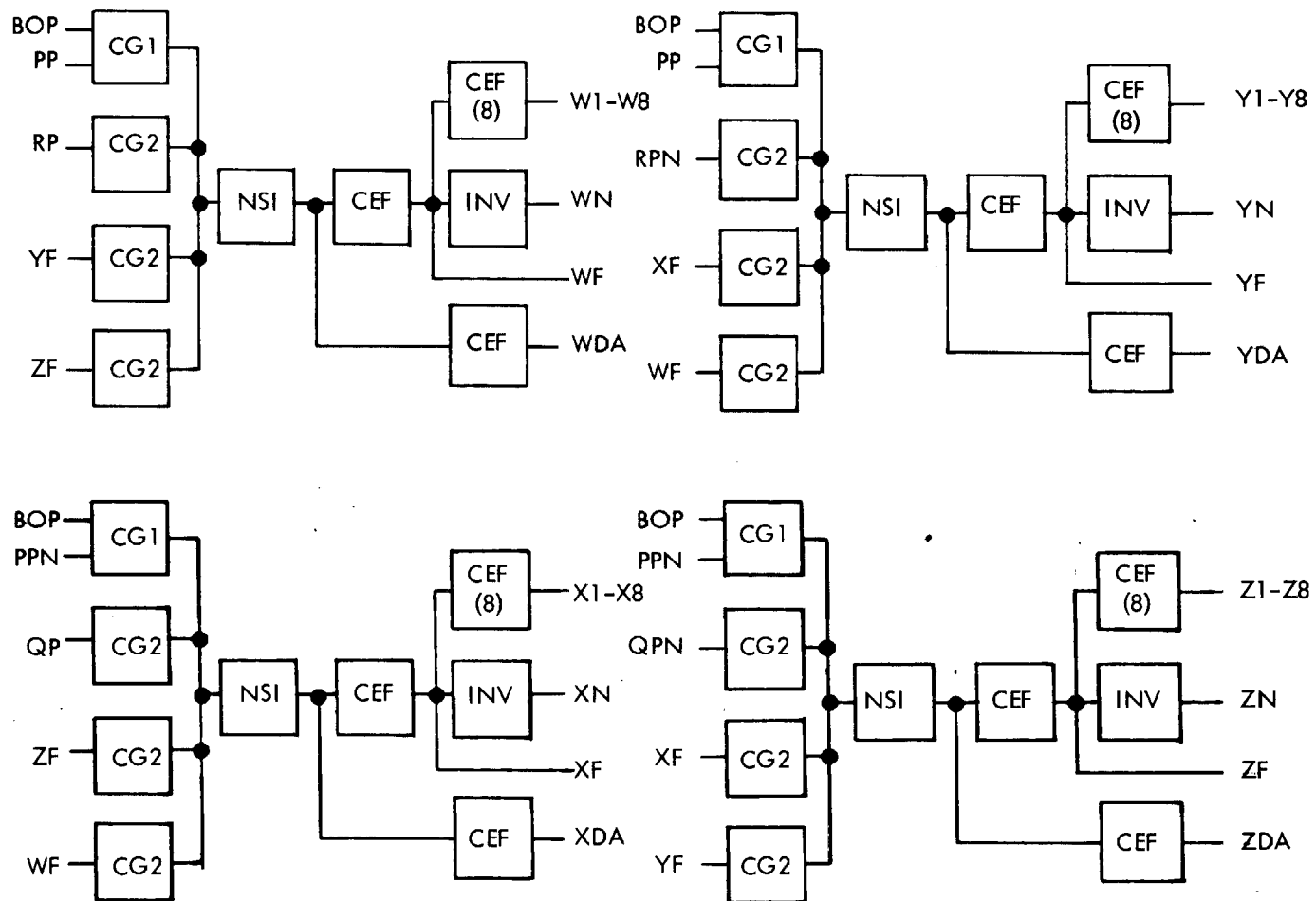


Figure 2-13. Clock Drivers



Figure 2-14. Clock Generator Timing

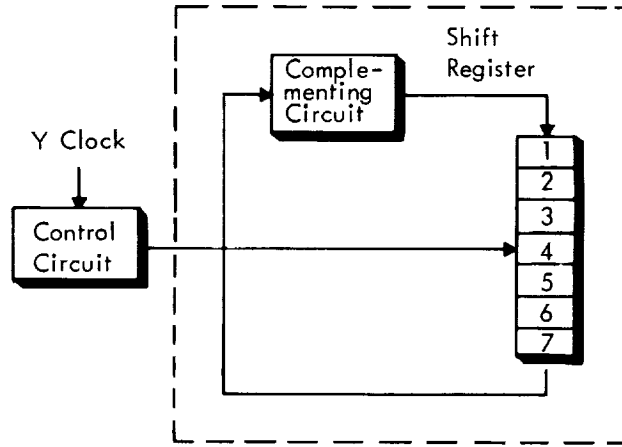


Figure 2-15. Timing Gate Generator Block Diagram

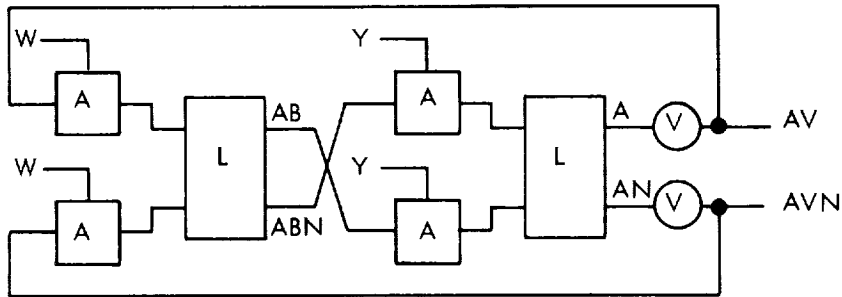


Figure 2-16. Control Circuit

2-81. Shift Register. The shift register consists of seven latches connected in series (figure 2-17). As long as any "1's" exist in latches G2 through G7, gate A1 cannot open. Ultimately, any "1's" in the shift register are shifted into latch G7 to open gate A2 and reset latch G1. Latch G1 remains reset, cycling "0's" into the shift register until latches G1 through G7 are all reset. Then gate A1 opens, setting latch G1 and effectively cycling "1's" into the register. Latch G1 remains set as long as "0's" are being cycled out of latch G7. The complementing circuit previously mentioned is actually latch G1 with its driving gates, A1 and A2.

2-82. In this shift register, "rippling" is prevented by the control circuit which gates inputs to the shift register latches. Adjacent latches are gated set and reset during opposite halves of the control circuit cycle. Even numbered latches are set during the time the A latch is set and odd numbered latches are set when the A latch is reset. Consequently, neither a set nor a reset condition can be propagated through successive latches on any one shift.

2-83. Identification of Bit Times. The outputs of the timing gate generator are diagrammed in figure 2-20. Bit 1 time is identified by the coincidence of signals G1V and G2VN; however, this identification is valid only during the Y and Z clocks, since G1V is in transition during the W and X clocks. During the W and X clocks, bit 1 time must be identified by the coincidence of signals G2VN, G7VN and AVN. Therefore, three signals are needed to identify bit times during W and X clocks, whereas only two signals are needed during Y and Z clocks. Bit times are decoded at the point where they are used. The combinations of signals used to identify bit times are as follows:

<u>Bit Time</u>	<u>W or X Clock</u>	<u>Y or Z Clock</u>
*1	G7VN, G2VN, AVN	G1V, G2VN
2	G1V, G3VN, AV	G2V, G3VN
3	G2V, G4VN, AVN	G3V, G4VN
4	G3V, G5VN, AV	G4V, G5VN
5	G4V, G6VN, AVN	G5V, G6VN
6	G5V, G7VN, AV	G6V, G7VN
7	G6V, G1V, AVN	G7V, G1V
8	G7V, G2V, AV	G1VN, G2V
9	G1VN, G3V, AVN	G2VN, G3V
10	G2VN, G4V, AV	G3VN, G4V
11	G3VN, G5V, AVN	G4VN, G5V
12	G4VN, G6V, AV	G5VN, G6V
13	G5VN, G7V, AVN	G6VN, G7V
14	G6VN, G1VN, AV	G7VN, G1VN

*See paragraph 2-87 for special considerations

2-84. PHASE GENERATOR. The phase generator is a three-step ring counter formed from three latches, PA, PB and PC (figure 2-18). The ring stabilizes itself with one latch set and two reset; the set condition is then stepped around the ring in alphabetical sequence each bit 1 time. At W clock of each bit 1 time, the next latch in the sequence is set; at the following Y clock, the preceding latch is reset. During a W clock when a latch is being set, the reset output of the latch for the previous phase time prevents the set condition from "rippling" into the third latch of the ring.

2-85. The "anti-ripple" signals also help the ring to reach its normal configuration. When two latches in the ring are set at the same time, the "anti-ripple" signals prevent the third latch from being set at W clock of the next bit 1 time. One latch is then reset at Y clock of each bit 1 time until the normal configuration is reached.

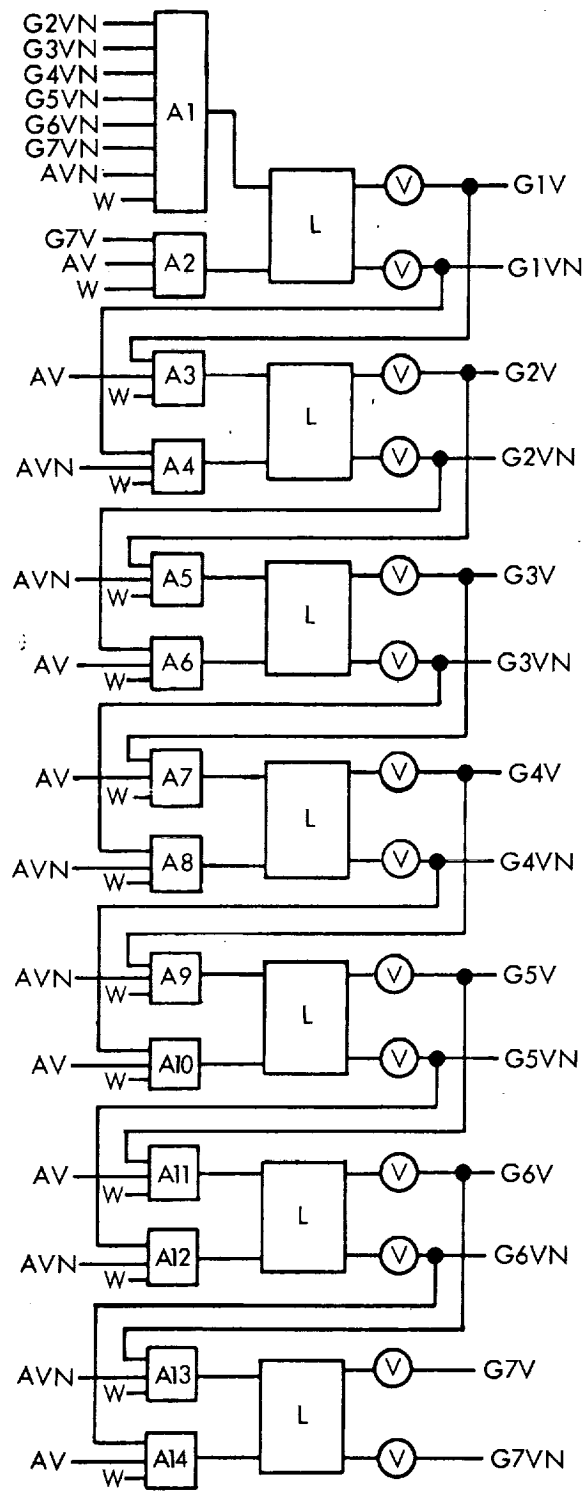


Figure 2-17. Shift Register

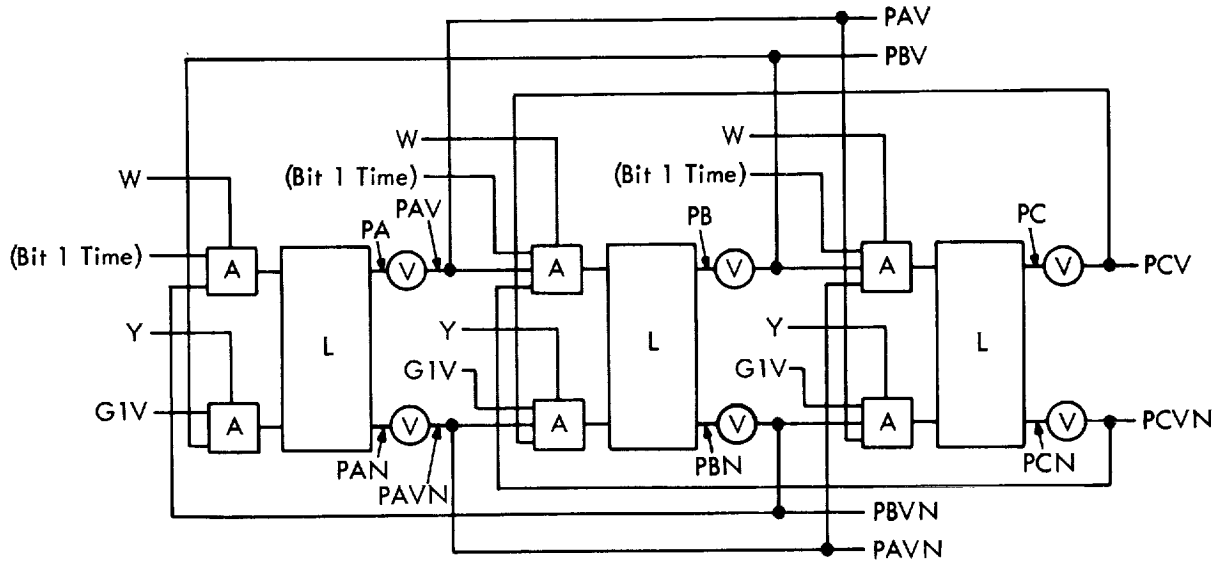


Figure 2-18. Phase Generator

2-86. The outputs of the phase generator are diagrammed in figure 2-20. The outputs of the ring counter correspond to the phase times except during bit 1 time where transition time and overlap occur. To properly identify phase time during any bit 1 time, it is necessary to sense the coincidence of timing signals as follows:

	Y or Z Clock		W or X Clock
Phase A	$\left. \begin{array}{l} \text{PAV, PBVN} \\ \text{PBV, PCVN} \\ \text{PCV, PAVN} \end{array} \right\} \text{G1V, G2VN}$	and	$\left. \begin{array}{l} \text{PCV, PBVN} \\ \text{PAV, PCVN} \\ \text{PBV, PAVN} \end{array} \right\} \text{G7VN, AVN}$
Phase B			
Phase C			

2-87. SPECIAL TIMING. The TBC latch (figure 2-19) generates a special timing signal which is used in the add-subtract and multiply-divide elements. The TBC latch is set at B-3-Y time and reset at the following A-1-Y time.

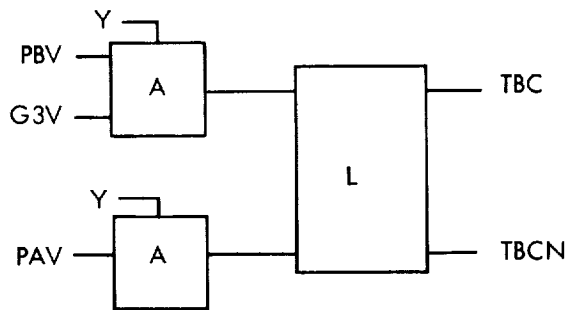


Figure 2-19. TBC Latch

2-88. MEMORY ELEMENT.

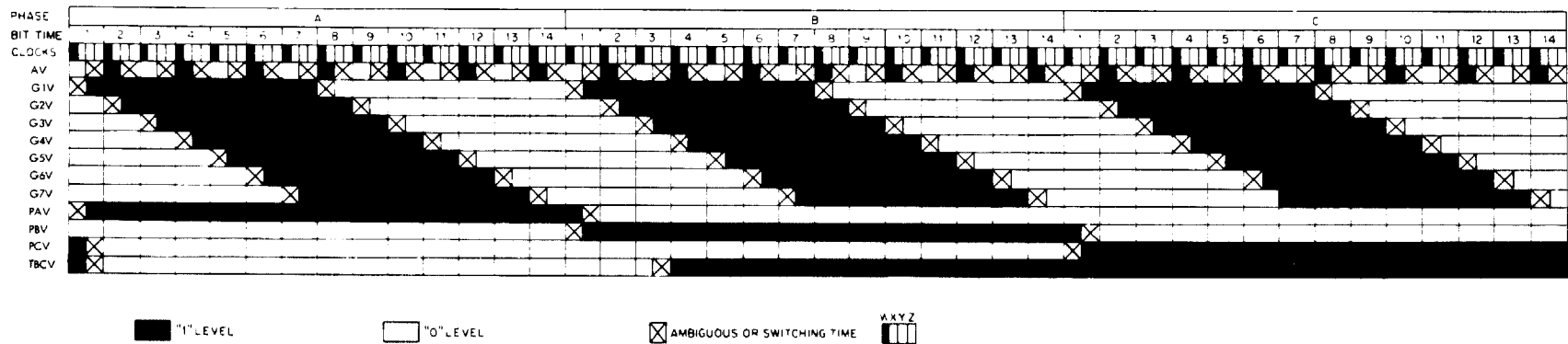
2-89. The Memory Element, figure 2-21, provides 4,096 locations of primary storage in each of up to eight memory modules. The Memory Element may be operated in either a simplex or duplex mode, as determined by the Memory Control Element. Simplex operation takes full advantage of the available capacity by storing different information in each mode. Duplex operation halves the capacity of the memory but increases its reliability through redundant storage. In duplex operation, the modules function in pairs with each module in a pair containing identical information. Errors detected in one module are corrected with the good information from its mate. The even numbered modules are grouped together to form one independent memory while the odd numbered modules form another memory with identical contents. Module pairs are split between the two memories, being formed of one odd and one even numbered module.

2-90. Each memory module operates independently on command from the Memory Control Element. The modules are each divided into sixteen sectors, one of which is designated the "residual" sector. Each sector contains 256 locations, or addresses. Sector selection is identical for each module and address selection is the same for every sector, except the residual sector. Address, sector and module selections are controlled separately in the Memory Control Element. As far as "addressing" in the usual sense is concerned, the Memory Control Element provides only sector and individual location addressing signals; they are applied to every module in the Memory Element and select the same address in each module. The Memory Control Element then commands only the selected module (two modules in duplex operation) to operate.

NOTE

In the accompanying diagrams, the second character of signal names applicable to the memory identifies the module or group of modules to which the signals apply. The small letter "m" indicates that a separate signal is developed for each module in which the module number (0 through 7) replaces the "m." The small letter "n" identifies duplex signals for which one signal is developed for all the even numbered modules and another for the odd numbered modules; in these signals the "n" is replaced by a "1" for odd modules and a "2" for even modules.

2-91. Toroidal ferrite cores are the storage medium, utilizing coincident current addressing and destructive readout techniques. A restore cycle follows every read operation to preserve the contents of the memory. Each memory location stores a 28-bit word which is divided into two 14-bit syllables. The Memory Element transfers one syllable at a time which necessitates four operations (two read cycles and two restore cycles) to transfer a complete word. The computer has random access to stored information within pre-selected sectors.



NOTE

This figure is located at the end of the Timing Element description and should be referenced throughout the description. For the reader's convenience, a slide rule version of this figure is located in the pocket inside the front cover. The reader will also find this slide rule a handy reference tool in the descriptions which follow the Timing Element.

Figure 2-20. Computer Timing



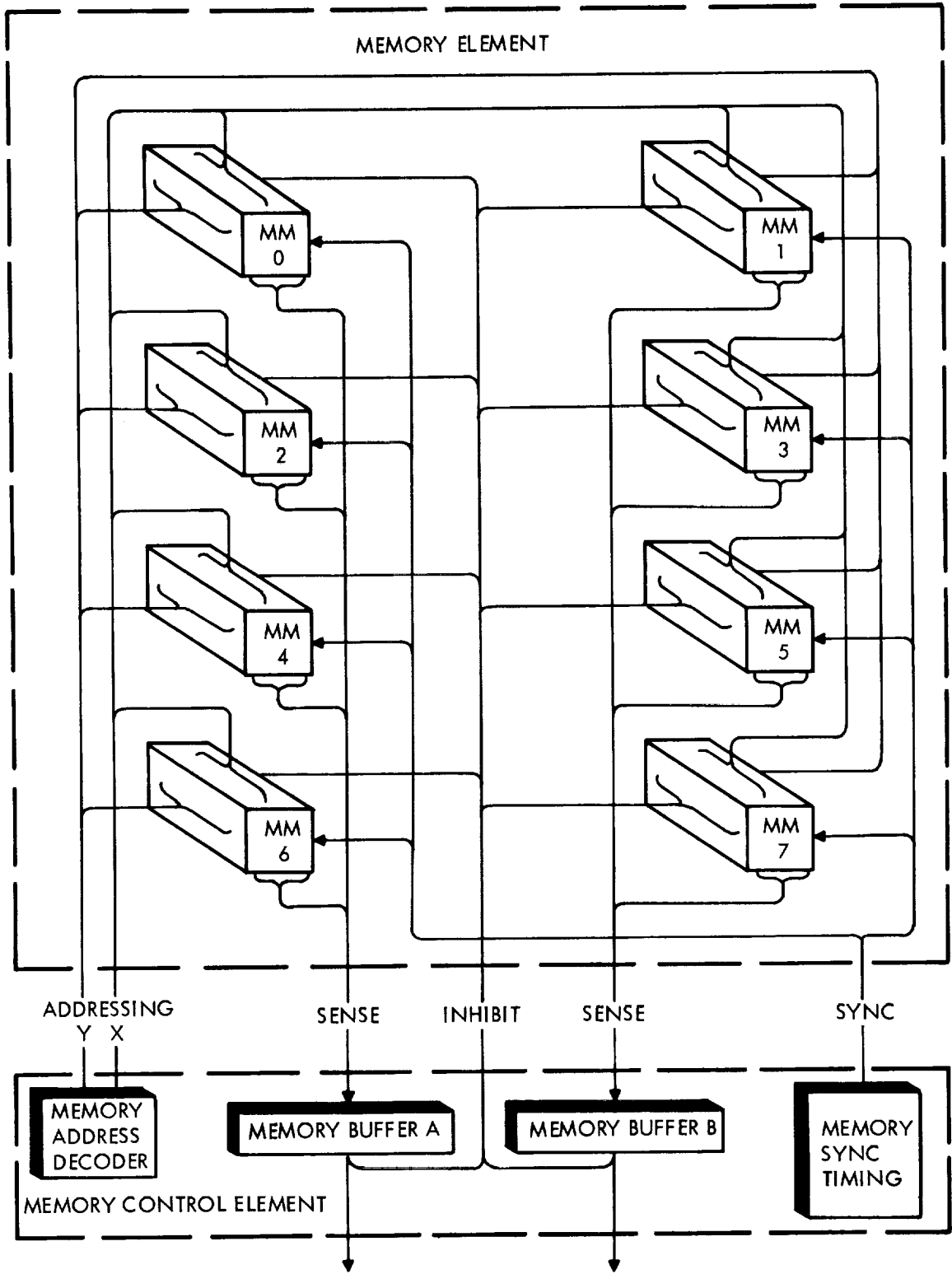


Figure 2-21. Memory Element Block Diagram

Changed 4 January 1965

2-92. CORE MEMORY FUNDAMENTALS. Figure 2-22 illustrates the properties of a ferrite core. As shown in the figure, the core can be magnetized in either of two directions. By establishing that a core "contains" a "1" when magnetized in one direction and a "0" when magnetized in the opposite direction, the core can be used to store one "bit" of a binary number. The core is magnetized by passing a d-c current of $I/2$ through the X and Y drive lines in coincidence.

NOTE

I is the current necessary to drive the core into magnetic saturation. The directions of the currents in X and in Y must be additive to operate the core.

The direction of magnetic flux about the core can be reversed by passing the same currents through the drive lines in the opposite direction. Reversing a core in this manner is called "switching" it. A sense winding, represented by S in the figure, also passes through the core and lies within its magnetic field. When a core is switched, the resulting collapse and expansion of its magnetic field causes a pulse of voltage to be induced in the sense winding. Thus, if a core containing a "1" is switched to a "0", the resulting voltage pulse in the sense winding can be amplified and used to set a latch, indicating the core contained a "1". If the core already contained a "0" it would not switch and no voltage pulse would appear in the sense winding to set the latch.

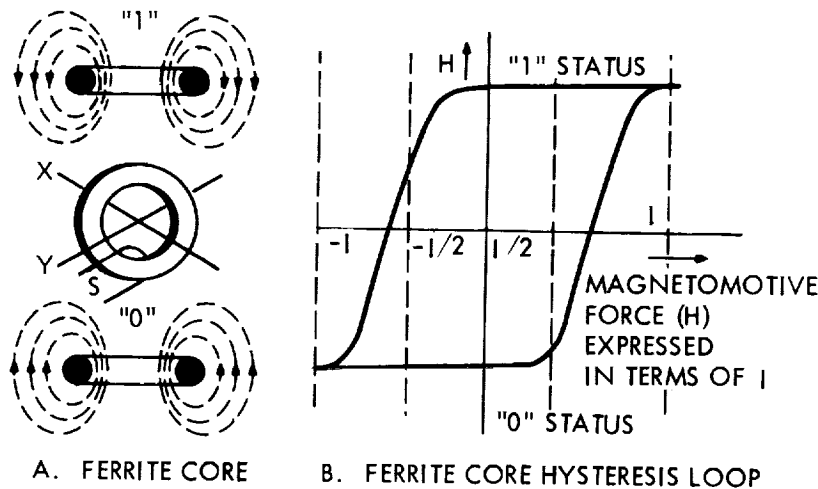


Figure 2-22. Ferrite Core Characteristics

2-93. This is the method used for reading information from a core memory. It is called "destructive readout" because after the core is read, it no longer contains the "1" which was sensed from it. The latch into which the "1" was read is a buffer between the core and the circuits which use the stored information. The "buffer latch" stores the "1" until it can be transferred into the computer and until the core can be switched back to its original state. The process of driving a core to a "0" and sensing whether or not it produced an output is called a "read cycle". Switching the core back to its original state is called a "store cycle".

2-94. The preceding paragraphs dealt with the operation of a single core. However, a single core can store only one bit of information, while a core memory must store many bits and select each bit on command. This ability can be achieved by arranging cores in rows and columns as shown in figure 2-23. An X drive line passes through all the cores in a column and a Y drive line through all the cores in a row. The cores are held in place by the wires passing through them; the wires are fastened to a frame and the unit is called a plane. The cores are positioned on the drive lines in a pattern which puts the magnetic fields of adjacent cores 90 degrees apart; this eliminates interference between cores. The sense winding runs parallel to the Y drive lines and passes through all the cores in the plane. The sense winding is wound in a pattern which results in cancellation of "half-select" noise (discussed later).

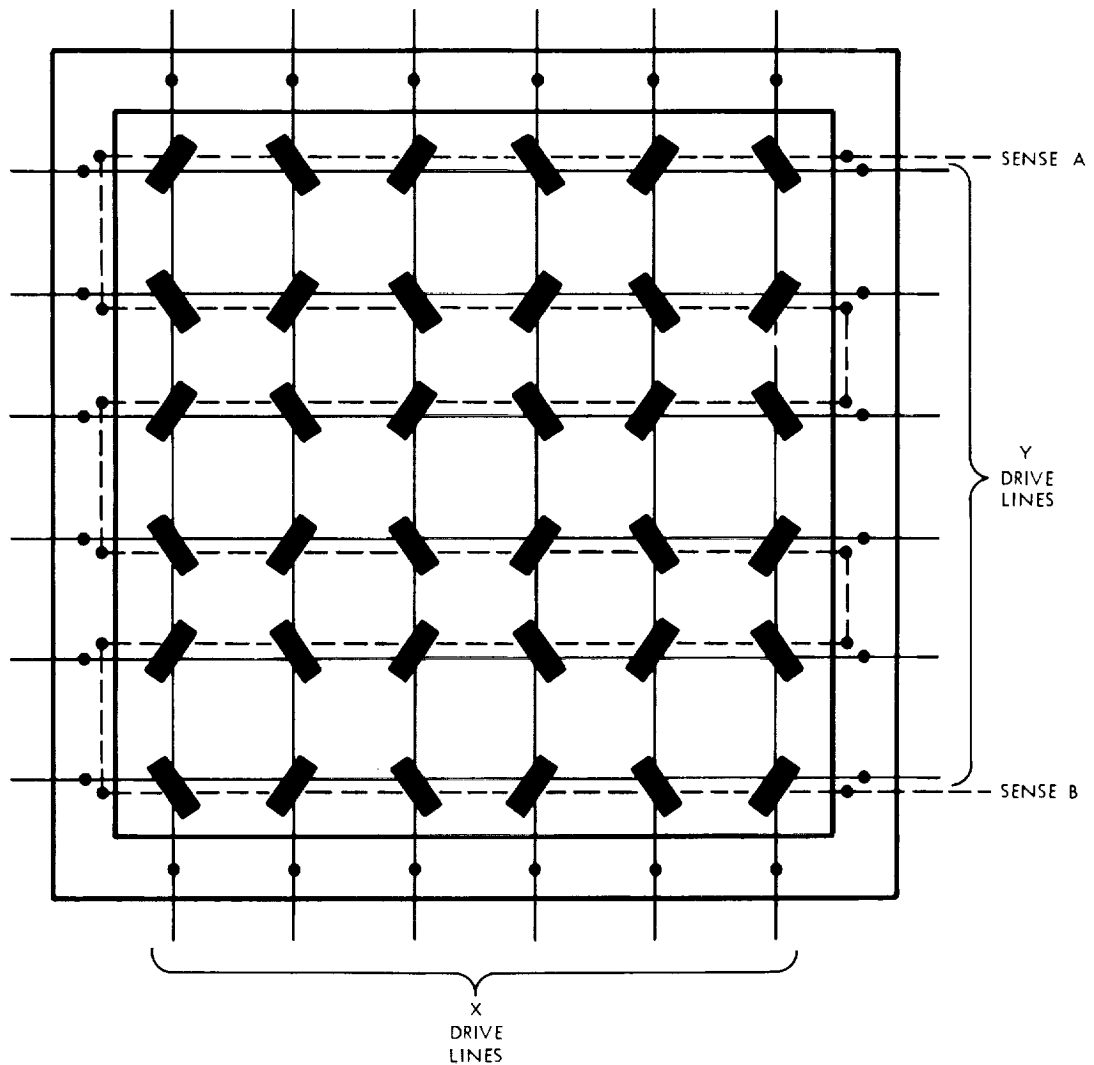


Figure 2-23. 6 x 6 Core Plane

2-95. If the plane is viewed as the first quadrant of a Cartesian Coordinate System, the X and Y drive lines represent units along the X and Y axes. The location of each core can then be defined by the X and Y drive lines which pass through it. The circuits which operate the plane are controlled so that only one X and one Y drive line carry current at any given time. Only the core at the intersection of the selected drive lines receives the coincident current necessary to switch it. Hence, this method of selecting memory locations is called "coincident-current addressing". When a memory location is selected by coincident-current addressing, one core along each selected drive line receives the full current necessary to select it. However, there are a number of other cores along each of the selected lines which receive one-half the current of the selected core; these cores are called "half-selects". Although half-select cores remain virtually at magnetic saturation, those containing "1's" will transmit small voltage spikes (called half-select noise) into the sense winding. The sense winding is wired so that half-select noise from one core cancels that from another. Thus half-select noise cannot mask a "0" in the selected core.

2-96. The arrangement of cores into planes facilitates storing and addressing many bits of information, but still falls short of practicality, because only one bit can be addressed at a time. Computers operate using "words" which comprise a number of bits, thus the memory must be expanded to store a complete word in each location. The expansion is accomplished by providing a separate plane for each bit in the word. The planes are stacked one above the other as shown in figure 2-24, to form an "array". Each X drive line is linked to the corresponding lines of the other planes in the array to form one continuous drive line. This drive line passes through the same locations in every plane of the array. The Y drive lines are similarly interconnected. Thus, when a pair of drive lines are energized, a core is selected in each and every plane at the corresponding intersection of the X and Y drive lines.

2-97. Each plane has a separate sense winding and buffer latch to allow the bits to be sensed individually. But one encounters a problem in attempting to restore the information back into the memory. As presented up to this point, energizing a pair of drive lines switches all the cores in the addressed location to the same state. There is (thus far) no provision for storing numbers which contain both "1's" and "0's". Therefore, a fourth winding called an "inhibit" winding is wound through each plane to allow bits to be stored individually. The inhibit winding runs parallel to the X drive line and carries the same current as the drive line but operates only during store cycles. During a store cycle, the drive lines attempt to switch every core in the addressed location to "1's". However, any buffer storage latches containing "0's" energize the inhibit windings in the corresponding planes. Inhibit current opposes the current in the X drive line and cancels its effect. Thus, the addressed cores in the inhibited planes feel only the half-select current of the Y drive line and do not switch. Since the cores in the addressed location were all switched to "0's" when they were read, any inhibited cores will simply remain in the "0" state.

2-98. Associated with the array are a number of special circuits which are required to operate it. Included in these circuits are the sense amplifiers, some special timing circuits and the circuits which provide power to the drive lines and inhibit windings. These circuits vary widely from system to system and therefore are not included in the description of Core Memory Fundamentals. Detailed operation of the circuits used in this system are described in subsequent paragraphs.

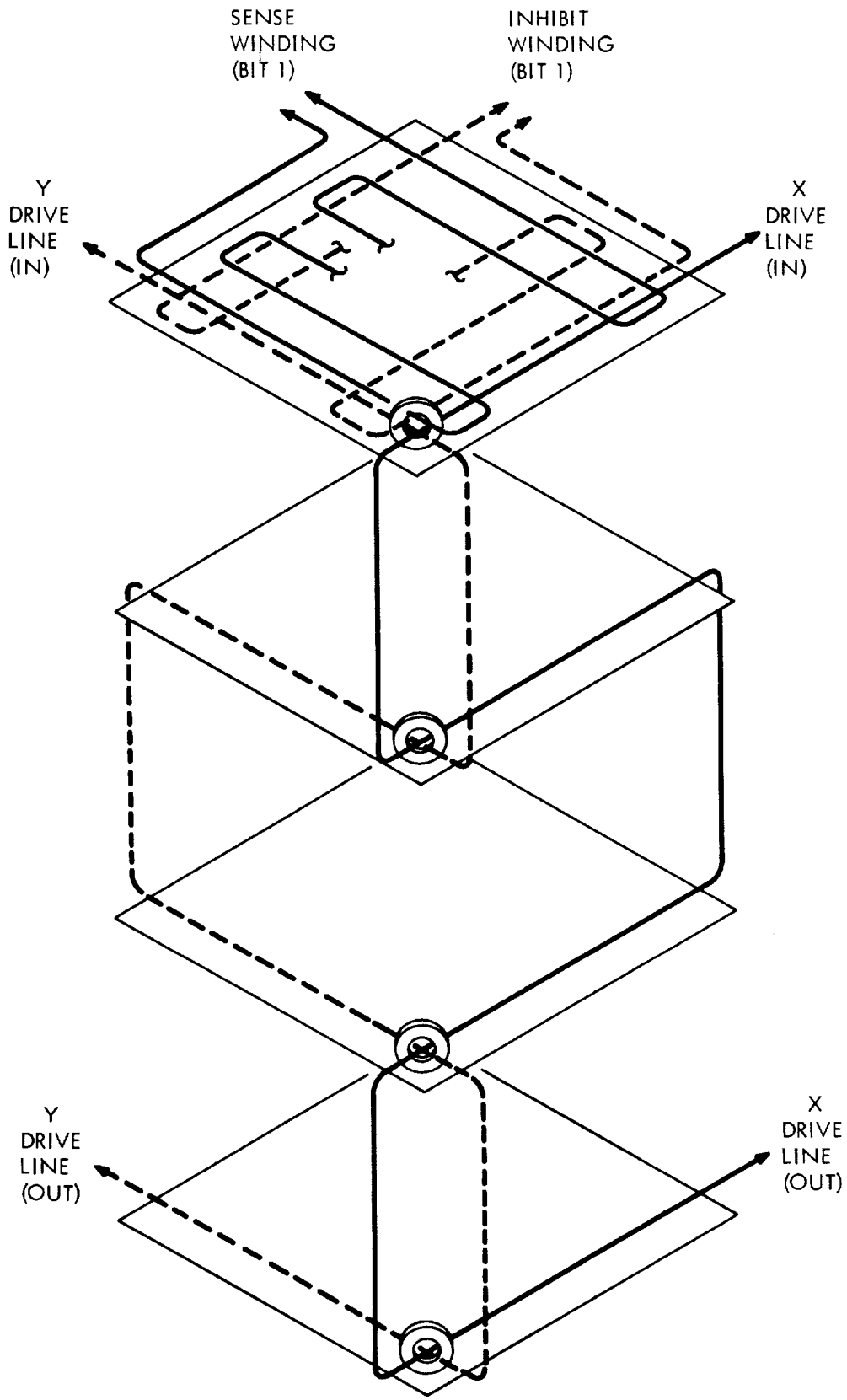


Figure 2-24. Core Array Windings

2-99. **MEMORY MODULES.** The Memory Element is composed of up to eight integral and asynchronous memory modules, which operate under control of the Memory Control Element. The modules are termed "integral" because each module is a complete memory in itself. Included in each module are a core array and all the circuits required to transfer information to and from it. Asynchronous refers to the fact that the timing signals which carry a module through a memory cycle are developed within the module and are not in synchronism with any others in the computer. When information is to be transferred to or from the Memory Element, the computer delivers a "sync" impulse to the selected module (or modules), waits a reasonable time for the module to perform the transfer, then continues assuming that the transfer is complete. During the memory cycle, the Memory Control Element provides signals to the Memory Element which control addressing and determine the direction of transfer.

2-100. Figure 2-25 is a block diagram of a memory module showing all the circuits it comprises. The Memory Clock Drivers (upper left) receive the sync impulse from the computer and convert it to a series of read or store timing pulses. The timing pulses turn the Memory Address Drivers and Inhibit Drivers on and off and strobe the Sense Amplifiers at the proper time. There are two memory address drivers for each drive line in the array. Addressing signals from the Memory Control Element select one X and one Y drive line by conditioning two X and two Y memory address drivers. When the timing pulses occur, the conditioned memory address drivers pass coincident current pulses through the selected drive lines to switch the cores in the addressed location. Diode matrices provide isolation between drive lines. A pair of Error Detectors continuously sense X and Y half-select current to indicate addressing errors to error monitor circuits in the Memory Control Element. A Temperature Controlled Voltage (TCV) Regulator adjusts the memory address drivers and inhibit drivers to produce the optimum current output for the prevailing temperature in the array. A temperature sensing element attached to the array develops the inputs for the TCV regulator.

2-101. Static control signals from the Memory Control Element determine whether information will be read or stored by conditioning the memory clock drivers to produce the appropriate series of timing pulses. If information is to be read, a series of read pulses is generated causing the memory address drivers to switch all the addressed cores to "0's". Read timing includes a strobe pulse for the sense amplifiers which is delayed from the drive current to allow for the inductive reactance of the windings through the array. If the purpose of the read cycle is to clear the addressed location so that new information may be stored in it, the strobe pulse is inhibited and the sense amplifiers produce no outputs.

2-102. If a store cycle is required, the timing pulses produced by the memory clock drivers reverse the polarity of the memory address driver outputs. The memory address drivers then attempt to drive all the addressed cores to "1's"; however, the store pulses are also routed to the inhibit drivers. Any inhibit drivers conditioned by "0's" in the memory buffer register will produce an output opposing the X drive current. Inhibited cores feel only the Y half-select current and therefore, remain in the "0" state.

2-103. **Core Array.** A memory module is built around a toroidal core array with a capacity for storing 4,095 28-bit computer words. The core array, figure 2-26, is a stack of 14 memory planes 64 cores wide and 128 cores long. Each plane stores two bits of information, one bit from each of the two syllables into which the computer word is divided. The array is electrically split into two separate 64 x 64 arrays, one for each syllable. Thus the array itself is described as being divided into syllables and the two halves of the array are called "Syllable 0" and "Syllable 1". Each syllable contains 4,095 14-bit locations which are defined by the X and Y drive lines passing through them.

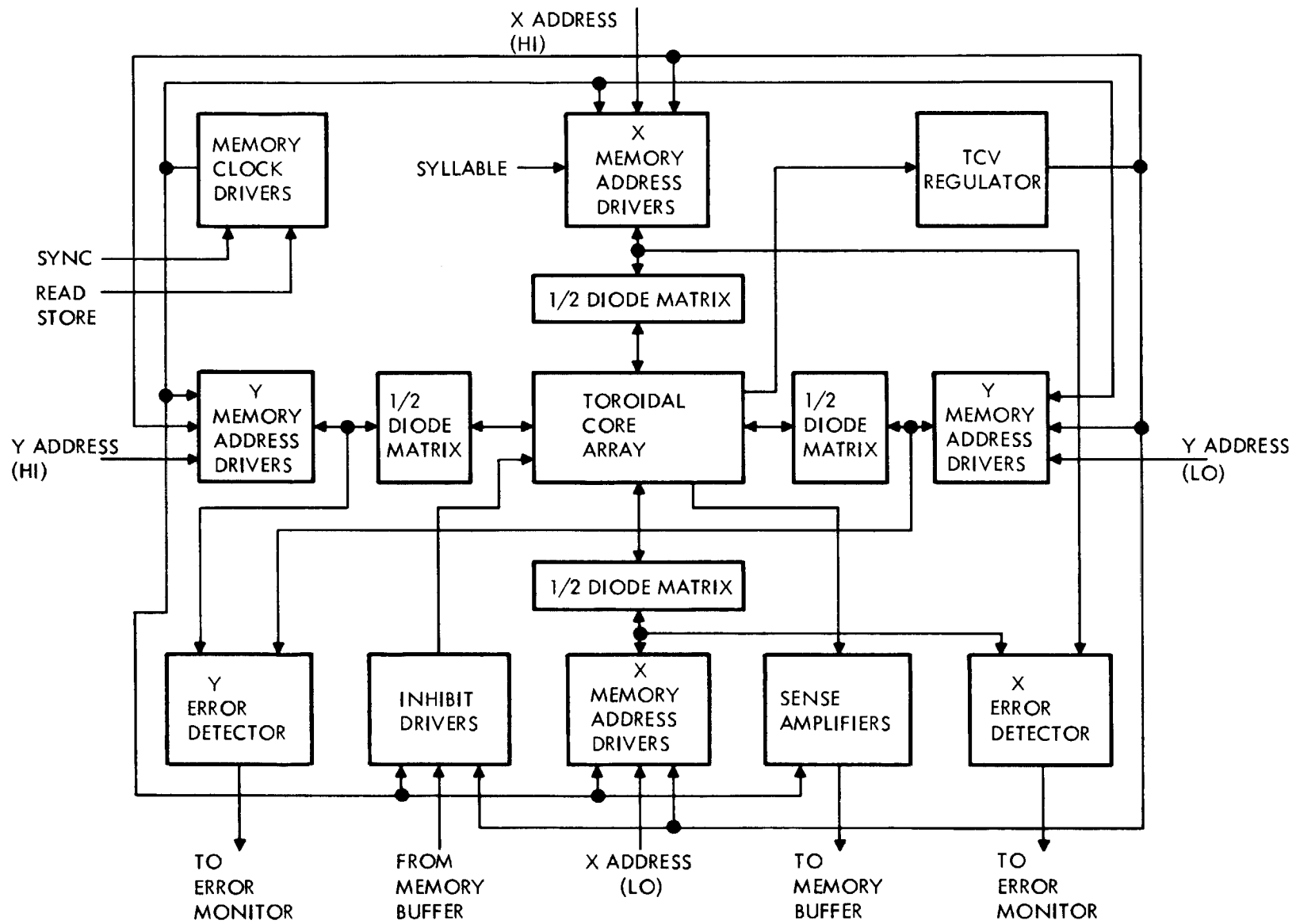


Figure 2-25. Memory Module Block Diagram

NOTE

The normal complement of addresses for a 64^2 array is 4,096, not 4,095. In this computer, a circulating shift register in the Multiply-Divide Element has been assigned one of the memory addresses. Consequently, no cores exist in the corresponding location of the array.

The Y drive lines are connected between sides B and D of the planes and are numbered octally as shown in figure 2-27. The X drive lines are connected between sides A and C of the planes and are also assigned octal numbers. There is a separate set of X drive lines for each syllable, thus syllable selection becomes a part of the memory addressing scheme. Also wound through each plane are a sense winding and an inhibit winding. The sense winding runs parallel to the Y drive lines and the inhibit winding parallel to the X drive lines.

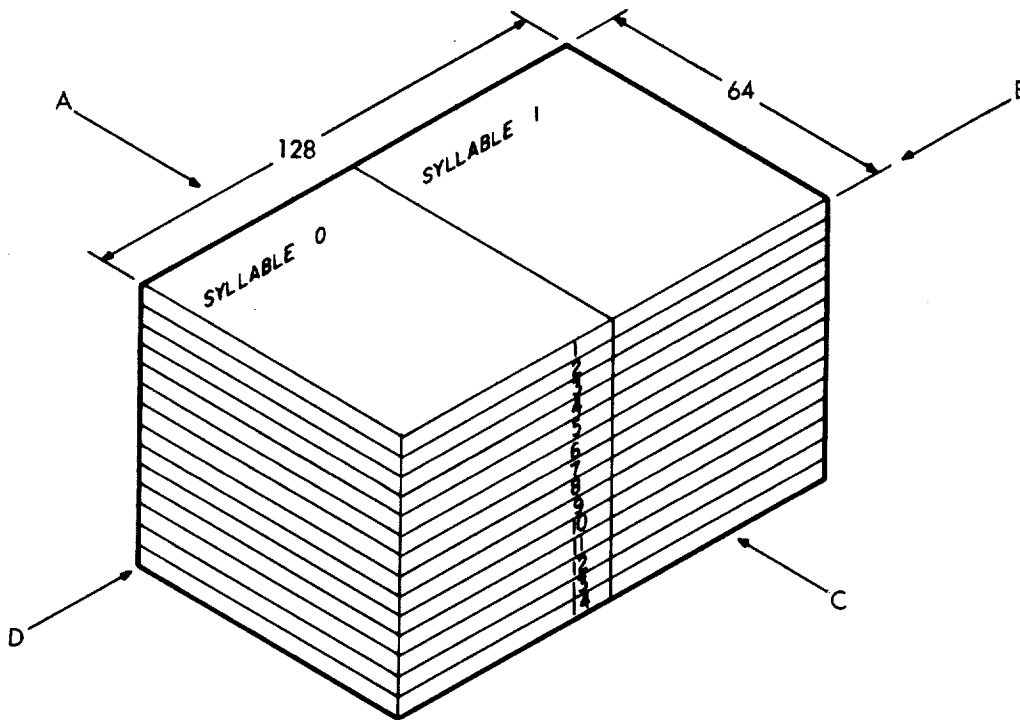
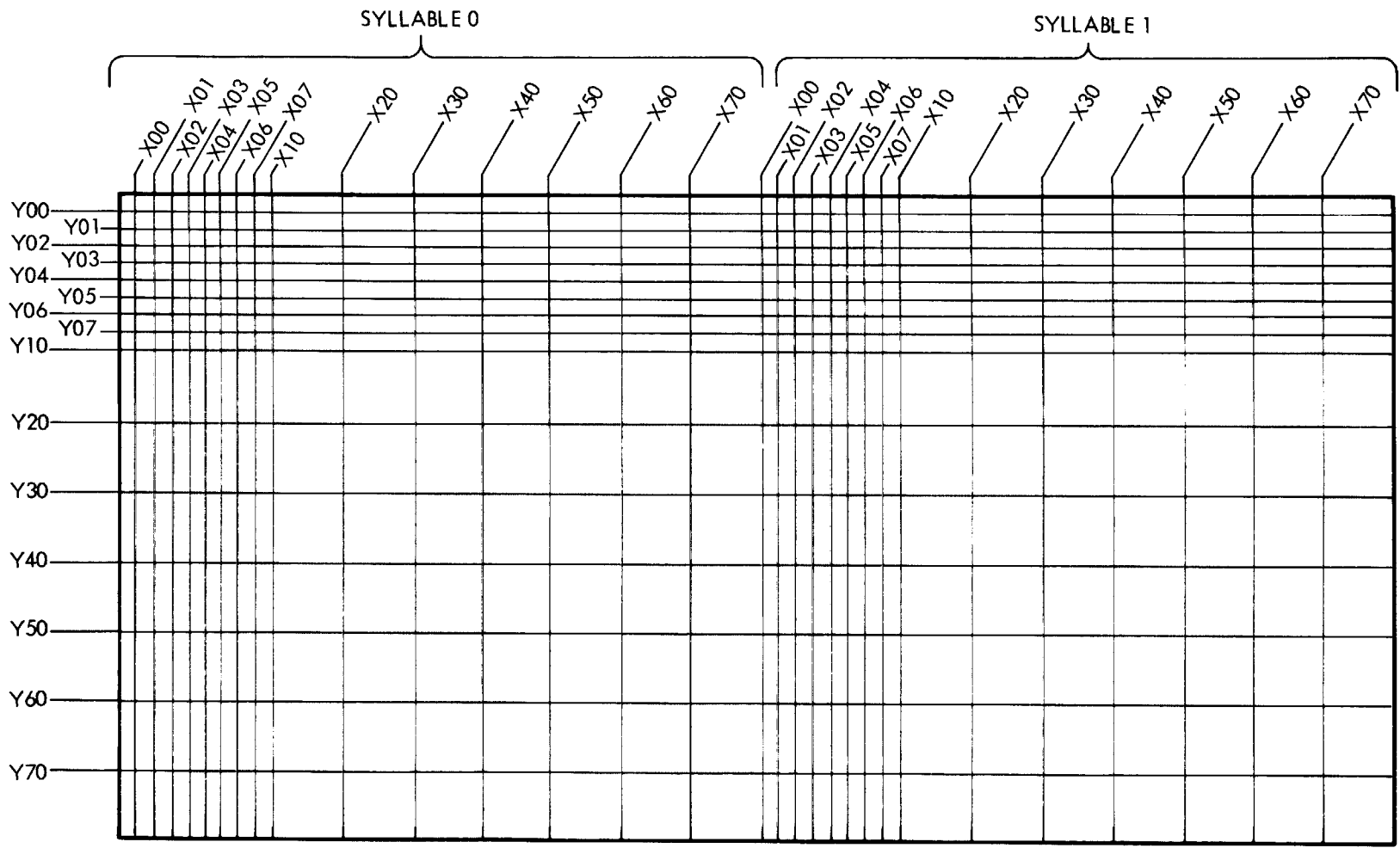


Figure 2-26. Toroidal Core Array

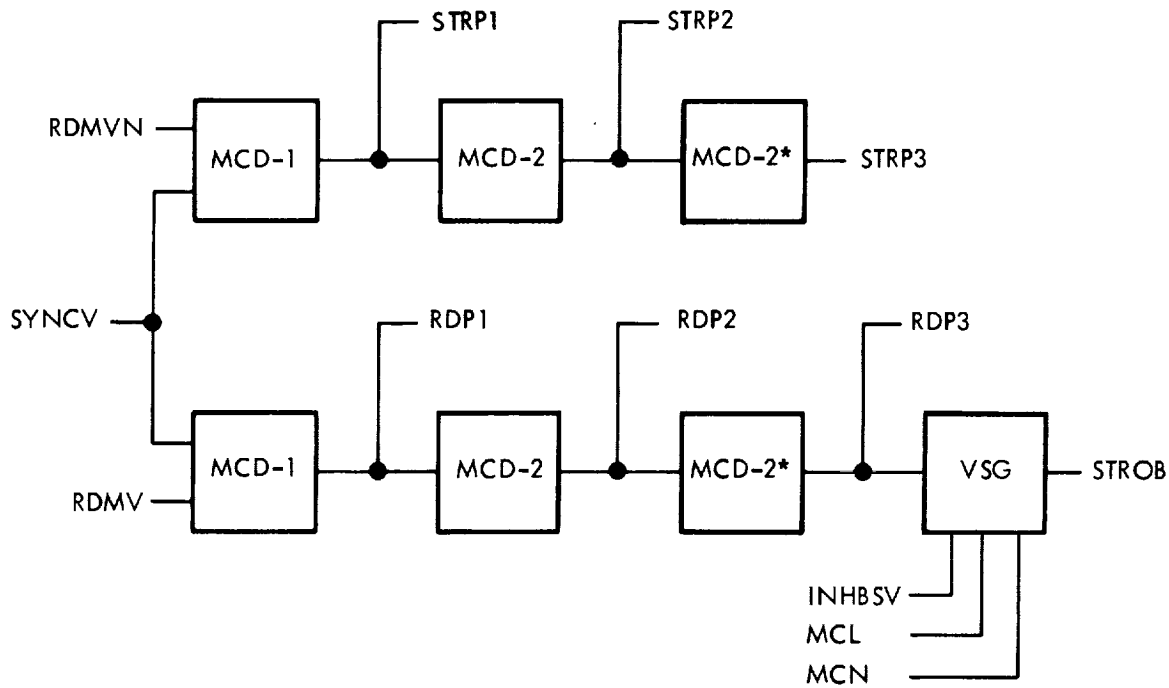
2-104. Short pins protruding from all sides of the planes frames facilitate connections to the planes and interconnections between planes. The sense and inhibit windings are connected to pins at the corners of each plane. The drive lines enter plane 1 and are then jumpered down to plane 2; after passing through plane 2, they are jumpered down to plane 3, etc. - weaving their way down through the array to exit at plane 14. To



2-43 Figure 2-27. Drive Line Assignments

reduce congestion at the points where the drive lines are connected to the driving circuits, even numbered drive lines enter and exit on one side (end) of the array and odd numbered lines enter and exit the opposite side (end). Thus the drive currents in adjacent lines flow in opposite directions. The inhibit windings then enter the planes at one end and weave their way back and forth across the planes like the shuttle in a loom. Consequently, inhibit current reverses direction with every pass across the plane so as to always oppose the X drive current during store cycles. Drive current for both X and Y is provided by the memory address drivers; the direction of the drive current is controlled by the memory clock drivers.

2-105. Memory Clock Drivers. The Memory Clock Drivers generate the timing pulses which gate information in and out of the array. During read cycles, the memory clock drivers gate read current pulses out of the memory address drivers and strobe the memory sense amplifiers at the proper time. During store cycles, the memory clock drivers gate pulses of store current out of the memory address drivers and turn selected inhibit drivers on and off. Included in the memory clock drivers are six memory clock driver circuits and a variable delay strobe gate. These circuits are connected in two chains, figure 2-28, one of which produces a series of read timing pulses and the other a store pulse sequence. A common sync-impulse initiates the operation of both chains, but only one chain operates at a time. The Read Memory (RDM) latch in the Memory Control Element determines which chain will be activated by each sync impulse.



NOTE: SEE FIGURE 10-36 FOR DETAILED LOGIC

Figure 2-28. Memory Clock Drivers

2-106. Operation of the two chains is identical except for the variable delay strobe gate which applies only to the read chain. Figure 2-29 shows the timing of both a read cycle and a strobe cycle and includes the control signals for reference. A memory cycle is initiated when the fall of SYNCV triggers the MCD-1 at the input to the selected chain. Each circuit then triggers the next in the chain which results in the outputs shown in the timing diagram. The Read Pulses (RDP1, 2, 3) control the timing and pulse width of the "read current" by timing the memory address drivers on and off. The variable delay strobe gate produces a delayed output called STROB which enables the memory sense amplifiers at the proper time to sense the outputs of the cores. The STROB output is normally a "1" level and is momentarily switched to a "0" by the positive transition of RDP3. The negative output pulse is approximately 460 nanoseconds wide with the leading edge delayed from the leading edge of RDP3 by approximately 400 nanoseconds. When the outputs of the cores are not to be sensed, i. e., when clearing a memory location to change its contents, the INH. Bit Strobe (INHBSV) signal is enabled. When INHBSV is a "1", the variable delay strobe gate produces no output; consequently, the memory sense amplifiers are not enabled and do not sense the core outputs. The delay time of the strobe pulse (STROB) can be varied during testing by two signals, Marginal Check Normal (MCN) and Marginal Check Late (MCL). These signals are manipulated by the test equipment to produce the following effects on strobe timing:

MCN	MCL	STROB
0	0	40 nanoseconds early
1	0	Normal
1	1	50 nanoseconds late

2-107. If a store cycle is required, the memory clock drivers produce Store Pulses (STRP1, 2, 3) rather than read pulses. Like the read pulses, the store pulses time the memory address drivers on and off, but the store pulses reverse the polarity of the memory address drivers so that "store current" is produced instead of read current. The store pulses also gate the inhibit drivers on and off so that cores which are to contain "0's" will not be switched to "1's".

2-108. Memory Address Drivers. The Memory Address Drivers develop the X and Y drive currents and provide a means for selecting individual drive lines. Forty composite circuits called EI drivers, figure 2-30, are divided into two groups, twenty-four for X and sixteen for Y, to implement a coincident current addressing scheme. Each group drives through a diode matrix to maintain isolation between drive lines. The memory address decoders (Memory Control Element) develop the selection signals which condition two X and two Y EI drivers at a time. Then, when enabled by the memory clock drivers, the selected EI drivers produce coincident pulses of either read or store current in one X and one Y drive line.

2-109. Each EI driver includes a voltage source (E) and a constant current source (I). Inverse selection signals applied to inputs A and B condition both sections to produce outputs when clocked. However, only one section is clocked at a time. The E clock and I clock inputs are the timing pulses developed in the memory clock drivers; one section receives store pulses and the other read pulses. The addressing scheme determines the type clock to be applied to each section. The section receiving a clock input produces a dual output if the selection signals at A and B are "0's".

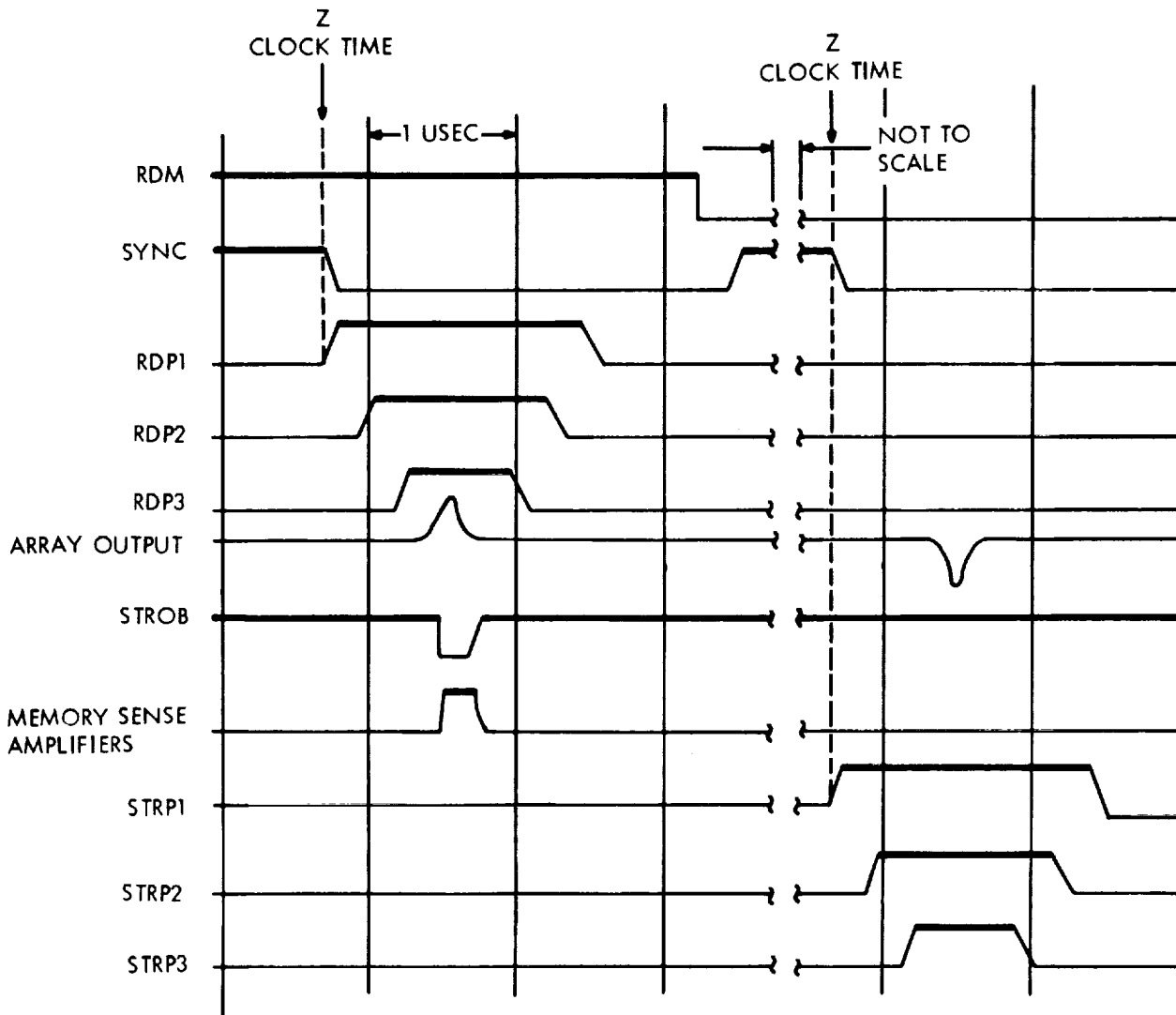


Figure 2-29. Memory Timing Diagram

2-110. The E section produces a positive level of approximately 17 VDC on both its outputs for the duration of the clock input. The E output is applied to one end of the array through a diode matrix; the EDE output is isolated from the E output and allows the operation of the voltage source to be monitored by an error detector circuit. The I section produces a current pulse of approximately 220 milliamperes which, like the voltage outputs, is timed by the clock input. The current pulse is applied to the same end of the array as the voltage output and, through the isolated output EDI, to the same error detector circuit.

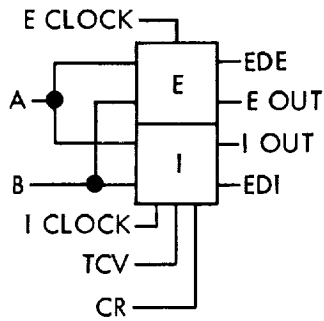


Figure 2-30. EI Driver Logic Symbol

2-111. The I section of each EI driver bears two signals, TCV and CRX or CRY, which are not associated with the E sections. The TCV input adjusts the output of the constant current source to compensate for temperature variations in the array. A rising TCV input indicates a decrease in the temperature of the array and increases the current output at the rate of 1.33 milliamperes per degree (C) of temperature change. The CRX input connects a common current regulating resistor to the constant current source of all the EI drivers serving X drive lines. A separate resistor connected by CRY is common to all the Y EI drivers.

2-112. As shown in figure 2-31, the EI drivers operate in pairs. Read pulses control the voltage source of the EI driver at one end of the drive line and the constant current source at the other end; store pulses control the remaining sections of the driver pair. When the read pulses occur, the E section provides a positive voltage source which draws current through the drive line from the constant current source. Store pulses place the enabled voltage source at the opposite end of the drive line and therefore reverse the direction of current flow.

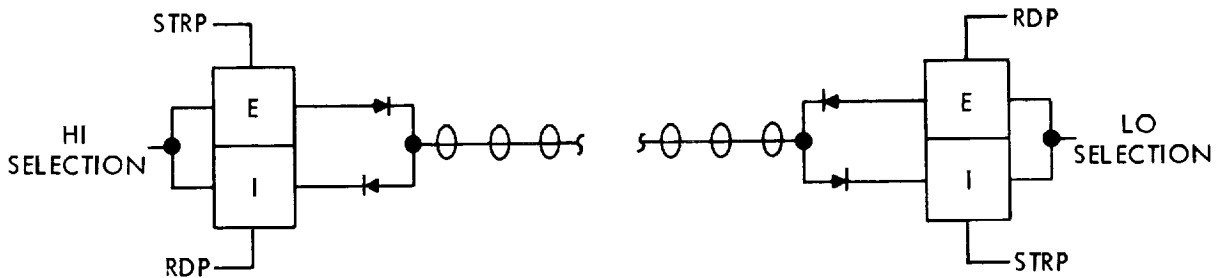


Figure 2-31. Paired EI Drivers

2-113. The fact two EI drivers must be conditioned in order to provide power to one drive line is used to advantage in instrumenting the selection of addresses. Looking back at figure 2-27 notice that both the X and the Y drive lines are assigned two digit octal numbers ranging from 00 through 77. There are two sets of X drive lines, therefore Y address selection, being the simpler, is described first. The memory address decoders provide sixteen signals for use in selecting Y drive lines; eight of these signals represent all the possible values (0 through 7) for the high order bit of the drive line numbers, the remaining eight signals represent the low order bit. Combinations of one high order and one low order signal can select all of the Y drive lines just as the high and low order bits can be combined to number them. The EI drivers lend themselves readily to such a selection scheme.

2-114. An EI driver is provided for each selection signal, eight high order and eight low order; one signal conditions each driver. The high order drivers select one end of the drive lines and the low order drivers select the other end. Figure 2-32 illustrates this selection technique and a sample section of the diode matrix which maintains isolation between the drive lines. The outputs of each high order EI driver are applied through the diode matrix to a block of eight consecutive drive lines; eight blocks of eight lines each are formed. The opposite end of each line a block is connected (again through the diode matrix) to the outputs of a separate low order driver. Each low order driver selects one line in each of the eight blocks. Thus, any high driver can be paired with any low order driver (or the converse) to select any Y drive line. The selection signals must be controlled so that one and only one high order and one and only one low order signal are enabled for each memory cycle. The error detectors monitor both outputs of each EI driver to detect any departure from this requirement.

2-115. The X drive lines are selected in precisely the same manner as the Y drive lines except that there is a separate group of high order drivers for each syllable. This accounts for the fact that eight more EI drivers are required for X than for Y. Figure 2-33 depicts the X drive line selection scheme in simplified form. The X diode matrix is identical to the Y diode matrix except it is doubled in size to account for the two syllables. Note that the high order X EI drivers require two selection inputs while all the others require only one. The extra input determines syllable selection. The high order drivers in each syllable receive a common syllable selection signal.

2-116. The preceding paragraphs described the manner in which one X and one Y drive line are selected to address a memory location with coincident current pulses. In addition, the read and store timing pulses must be assigned to the proper combinations of E and I sections of the EI drivers so that read pulses always produce read current and store pulses always produce store current. Since the cores produce usable outputs when switched in either direction, the function assigned to each polarity is completely arbitrary. In practice, the I sections of the high order drivers and the E sections of the low order drivers are enabled by read pulses; the E sections of the high order drivers and the I sections of the low order drivers are then enabled by store pulses. Figure 2-34 shows which timing pulses are assigned to each set of EI drivers. Note that the voltage sources are turned on before the constant current sources and are turned off after the drive current has been terminated. This timing arrangement minimizes the effect of drive line inductance on the drive current rise time. Assigning the timing pulses to specific EI drivers establishes the polarity of the read and store functions. Since the memory sense amplifiers are not polarity sensitive, the only polarity remaining to be reckoned with is that of the inhibit drivers.

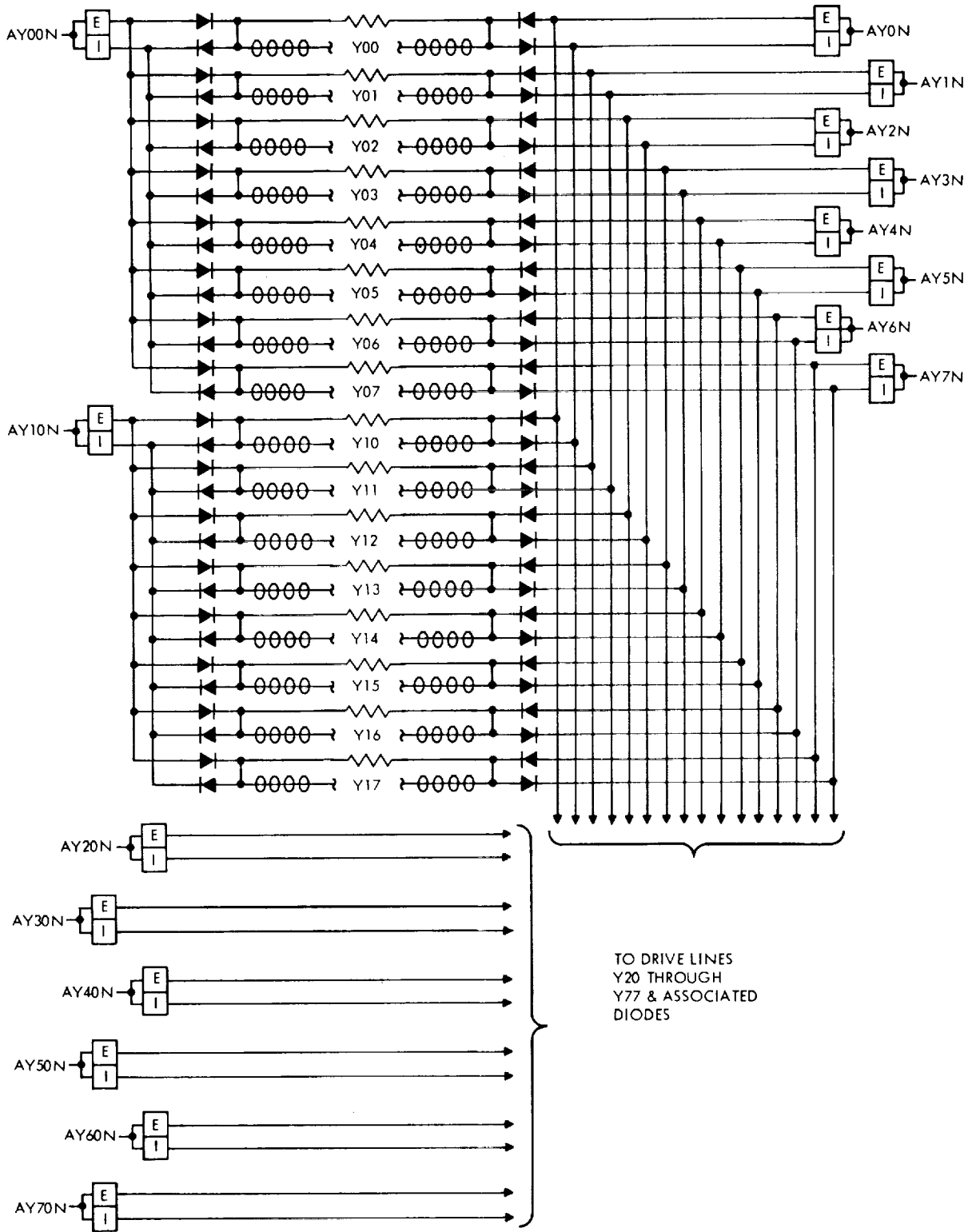


Figure 2-32. Y Drive Line Selection

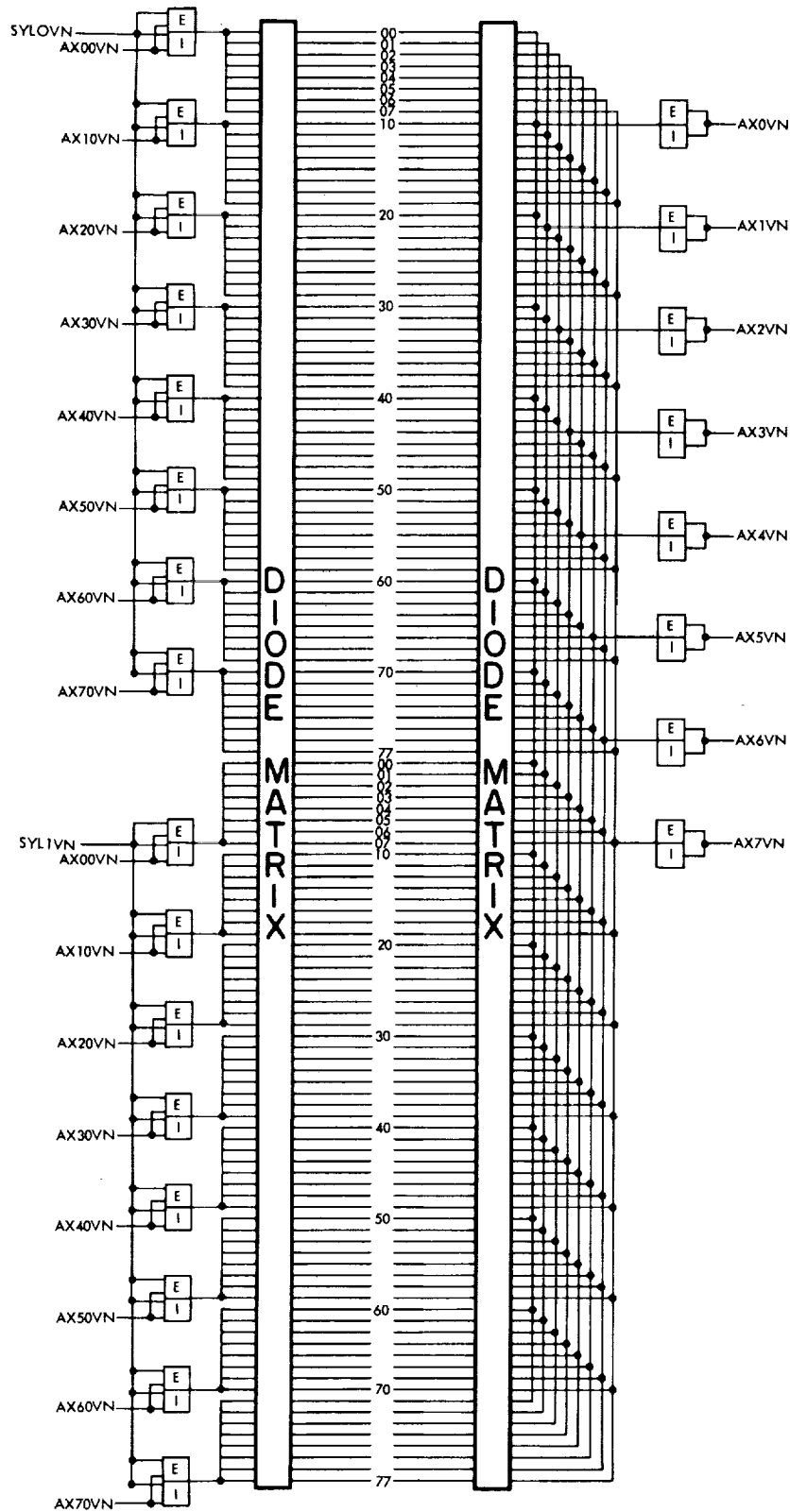


Figure 2-33. X Drive Line Selection

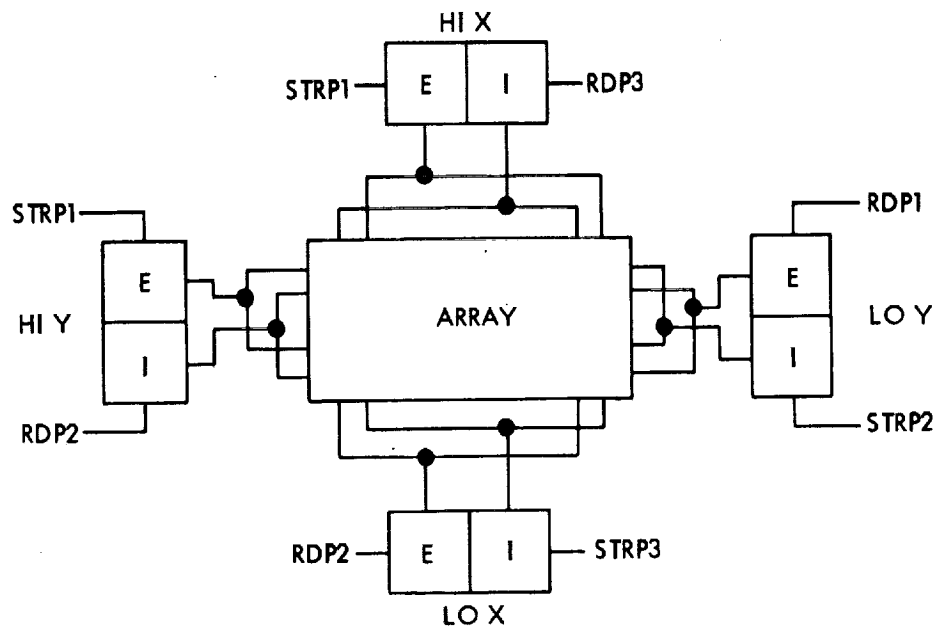


Figure 2-34. Timing Pulse Assignments

2-117. **Inhibit Drivers.** The Inhibit Drivers facilitate storing each bit of an addressed memory location separately, by preventing cores which are to contain "0's" from being switched to "1's". Fourteen inhibit drivers such as the one shown in figure 2-35 are provided to drive the fourteen planes of the array. During store cycles, inhibit drivers conditioned by "0's" in the memory buffer registers develop current pulses equal in magnitude but opposing the X half-select current. The memory clock drivers time the inhibit drivers on and off so that inhibit current overlaps the X drive current. Thus, addressed cores in inhibited planes feel only the Y half-select current and are not switched. The inhibit drivers of each memory module can be conditioned by either the A or the B memory buffer register; this provides a path over which good information from one memory can be routed to a failing memory during duplex operation to correct errors.

2-118. The inhibit driver circuit resembles the constant current source in the EI driver, but with a more complex input stage to provide greater flexibility of control. Two pairs of signals from the Memory Control Element control each inhibit driver. Each pair of control signals consists of an input from the A or B memory buffer and a gating signal from the buffer select latches. The inhibit drivers of even numbered memory modules are gated by the Buffer Register A On (BRAO) latch and odd numbered modules by the BRBO latch. An input from the A memory buffer completes one pair and an input from the B memory buffer completes the other pair at each inhibit driver. In the even numbered modules, the A memory buffer input is gated by BRAOV and the input from the B memory buffer by BRAOVN; in the odd numbered modules the A memory buffer input is gated by BRBOVN and the B input by BRBOV. The inhibit driver assumes that it will be turned on, then if either memory buffer output is a "1" and that memory buffer is "on", the inhibit driver is disabled.

2-119. Two store pulses, STRP1 and STRP2, are required to turn an inhibit driver on. The STRP1 pulse clocks the two AND circuits in the input stage to prevent them from drawing current during non-memory cycle times. The STRP2 pulse actually turns the inhibit driver on and off. Since the low order X EI drivers are controlled by STRP3, inhibit current overlaps the X drive current. Unlike the drive current, inhibit current

always flows in the same direction making a switchable voltage source unnecessary. The inhibit windings are therefore returned directly to the +20 VDC supply. The inhibit drivers also differ from the EI drivers in that each inhibit driver contains its own current regulating resistor. A single resistor can be commoned for the X (and another for the Y) EI drivers because only one current source is enabled at a time. The inhibit drivers, on the other hand, may be enabled simultaneously in multiples of up to thirteen and require separate resistors to withstand the added current flow.

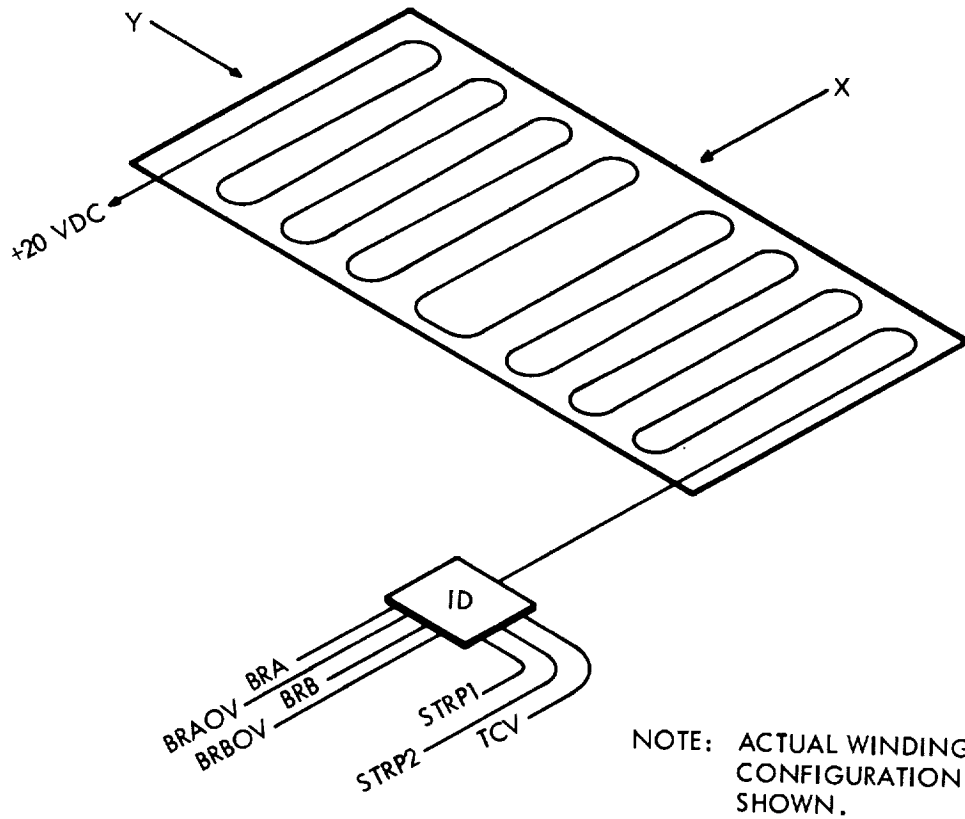


Figure 2-35. Inhibit Driver and Winding

2-120. To appreciate the impact of the inhibit driver control scheme on overall memory operation, one needs some insight to the working of the BRAOV and BRBOV signals. The BRAO and BRBO latches are described under the Memory Control Element, however for the moment some general information will suffice. During simplex operation, memory modules are selected singularly and only one memory buffer is used, the other remains cleared. However, once BRAO and BRBO are set, they remain set until an error is sensed in the corresponding memory. If an error is detected in a selected module, the "buffer register on" signal for the appropriate memory is disabled before the module is cycled again to restore the information. However, the "buffer register on" signal for the opposite memory may still be enabled since error sensing is not performed when the memory is not selected. Thus, when the restore cycle occurs, the inhibit drivers will sense the cleared memory buffer and force all "0's" into the error location, including the parity bit. If both BRAO and BRBO are disabled, the inhibit drivers will ignore both memory buffers which gives the same result. When the error location is read again it will cause a parity error because its contents have even parity and odd parity is required. This feature identifies error locations even for intermittent failures.

2-121. In duplex operation where modules are selected in pairs, both BRAOV and BRBOV are enabled and both memory buffers are used. Each memory buffer controls the inhibit drivers in one memory, A or B, gated by BRAO or BRBO. When an error is detected in one memory, the appropriate latch is reset as in simplex operation; the reset side of the latch then gates the information from the good memory buffer into the inhibit drivers of the failing module to correct the error location. An error in one module during duplex operation does not constitute a failure for the computer, therefore there is no need to identify the error location.

2-122. Memory Sense Amplifiers. The Memory Sense Amplifiers discriminate between "1's" and "0's" at the outputs of the memory plane sense windings. A separate memory sense amplifier, figure 2-36, is provided for each of the fourteen memory planes; thus, each bit of a memory location is sampled individually. Upon sensing a "1" in the sense winding, the memory sense amplifier delivers a timed output to set the memory buffer latch corresponding to the bit which was sensed.

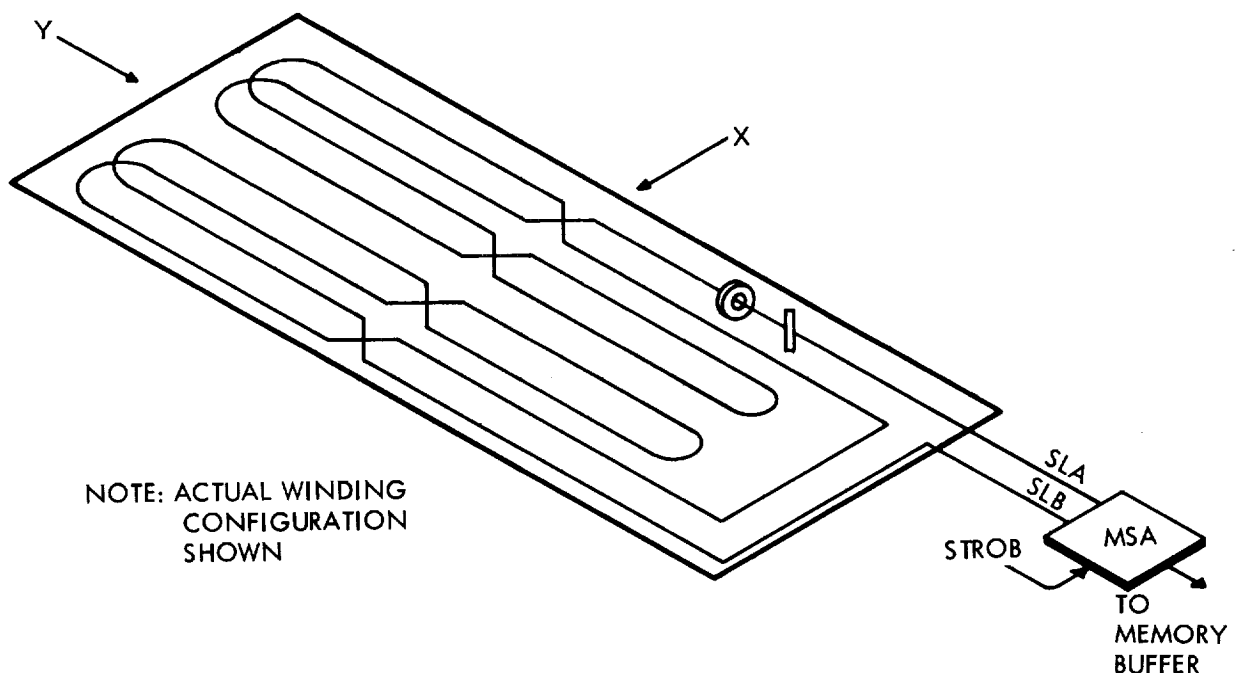


Figure 2-36. Memory Sense Amplifier and Windings

2-123. The input stage to the memory sense amplifier is a differential amplifier, thus the circuit senses both positive and negative "1" outputs from the sense winding equally well. The dual polarity "1's" result from the 90 degree separation of adjacent cores on the sense winding. The outputs of the differential amplifier are buffered and coupled to an output-switching stage which is controlled by the input from the variable delay strobe gate, STROB.

2-124. The output-switching stage contains a special latching circuit with an emitter follower output. If a "1" is sensed by the differential amplifier, the latch will be set when the STROB input goes down. The emitter follower then produces a "1" output until STROB comes back up, disrupting the latch circuit. Thus, the pulse width of the memory sense amplifier output is determined by the width of the strobe pulse and is not dependent upon the pulse width of the core outputs.

2-125. Error Detectors. The Error Detectors monitor the operation of the memory address drivers to provide an auxiliary indication of memory performance during read cycles, when parity checking is performed, and during store cycles when parity checking is not possible. Each memory module contains separate error detectors for the X and Y EI drivers. These error detectors provide outputs to the Memory Control Element which, along with parity checking circuits, are used in controlling selection of the memory buffer registers. The error detectors indicate the following categories of errors:

- 1) Multiple address selection
- 2) Partial address selection
- 3) Spurious address selection between memory cycles

2-126. Each error detector circuit, figure 2-37, consists of a current summing network, an error output circuit and a special input circuit for monitoring the current source regulator resistors. The EI driver voltage sources provide special monitoring outputs which are commoned at the input to the summing network. During normal operation, since only one voltage source is selected at a time, the output level of the summing network is insufficient to trigger the error output circuit. If a failure results in the selection of two or more voltage sources simultaneously, the output level of the summing network triggers the error output circuit indicating that parallel memory drive current paths are present.

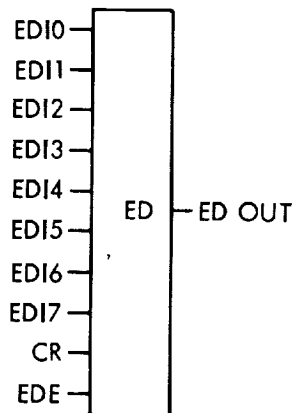


Figure 2-37. Error Detector Logic Symbol

2-127. The EI driver constant current sources are monitored in a manner similar to the voltage sources. Since the Y-coordinate high-order and low-order drivers are never clocked simultaneously, the monitor outputs of one high-order and one low-order driver are commoned. In the X-coordinate, the sensing is provided in a similar manner except that two groups of high-order drivers are used for syllable selection. In this case, the monitor output from each driver with the same address in both syllables is commoned with a low-order driver. The eight groups of commoned outputs for each coordinate are used to drive eight groups of current amplifiers commoned in a summing network. During normal operation, since only one current source is selected per coordinate, the output level of the summing network is insufficient to trigger the error output circuit. If a failure results in the simultaneous selection of two or more current sources, multiple selection of current amplifiers at the summing network triggers the error output circuit, indicating that parallel memory drive current paths are present.

2-128. To detect missing half-select currents, the outputs of current regulator resistors CRX and CRY are monitored. During normal operation the voltage pulse across each regulator resistor is sufficient to trigger the special input circuit whose output is connected to the EI drive voltage-source summing network. A malfunction that results in a loss of voltage across the current regulator resistor causes the threshold level of the summing network to rise in a manner similar to that caused by an EI driver voltage source failure.

2-129. During normal operation, the error detector produces a "1" output pulse which occurs during memory addressing. If the error detector output is a "0" during memory address time or a "1" between memory cycles, an error is propagated. The output of the error detector is clocked into the memory select and error monitor circuit in the Memory Control Element.

2-130. Temperature Controlled Voltage (TCV) Regulator. The switching characteristics of ferrite cores are subject to change with variations in temperature. The TCV regulator, figure 2-38, continuously adjusts the current output of the EI drivers and inhibit drivers to the optimum value for the temperature in the array. Linear temperature compensation is provided over the range of 10 to 70 degrees centigrade.

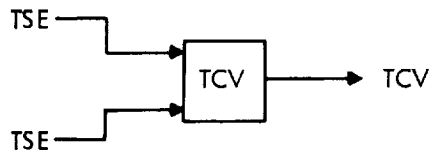


Figure 2-38. Temperature Controlled Voltage (TCV) Regulator Logic Symbol

2-131. The TCV regulator consists of a current source which drives a Temperature Sense Element (TSE) mounted on the array, and a buffer amplifier to reflect the voltage drop across the sense element. Rising temperatures in the array increase the resistance of the TSE. The TCV output varies inversely with the temperature change, ranging from approximately +6 VDC at 10 degrees (C) to +4 VDC at 70 degrees (C).

2-132. MEMORY ADDRESSING. The preceding paragraphs have described the role each circuit plays in storing and retrieving information from the memory. But these circuits have limited use without the ability to specify at random the location of desired pieces of information in the memory. Random access to the memory's contents facilitates variation in the programmed instruction sequence and processing similar data with a common program subroutine. The system for specifying the location of information in the memory is the addressing scheme.

NOTE

Memory addressing must not, in this computer, be confused with the X and Y drive line selection which is merely the implementation of memory addressing. Drive line selection signals are developed by decoding memory addresses.

2-133. The packaging method used in the Memory Element (separate, individually controlled modules) results in an unusual four part memory addressing scheme. A complete memory address specifies a memory module, a syllable within the module, a sector within the syllable and a location within the sector. The syllable selection applies to all modules; the sector address applies to both syllables of all modules; and the location address applies to all sectors of both syllables of all modules. But, this is not yet a true picture of the memory addressing scheme, because modules are not "addressed" in the usual sense. Instead, a sector address and a location address are combined in the memory address decoders to develop X-Y coordinate selection signals. These signals, along with syllable selection signals, are applied to the memory address drivers of every module. The module is addressed by routing the sync impulse which initiates a memory cycle to only the selected module. Module sync selection is described under the Memory Control Element and the remainder of this discussion is limited to correlating the sector and location addresses to the X-Y coordinates of the array.

2-134. Two 4-bit registers in the Program Control Element store separate sector addresses for instructions and data words. The instruction sector register is used during instruction time and the data sector register during operation time. The content of these registers is changed at intervals in the program by special instructions located at appropriate points in the instruction sequence. The sector registers augment a single 9-bit address register which selects individual locations within the selected data or instruction sector. The address register is reloaded twice per computer cycle, once at the beginning of instruction time and again at the beginning of operation time.

2-135. The memory address decoders combine the contents of the sector and address registers in four 3-bit groups to form octal selection signals for the EI drivers. Figure 2-39 shows how the two addresses are combined and which set of EI drivers is selected by each 3-bit group. In the Lo X and Lo Y groups the octal values 0 through 7 select EI drivers 0 through 7, respectively; in the Hi X and Hi Y groups, the same values select EI drivers 00 through 70, respectively.

REGISTER	SECTOR ADDRESS				LOCATION ADDRESS							
ADDRESS REG.	A9 (R)				A8	A7	A6	A5	A4	A3	A2	A1
SECTOR REG (IS/DS)	3	2	1	4								
EI DRIVERS	Hi Y				Hi X		Lo Y			Lo X		

Figure 2-39. Memory Address Decoding

2-136. Notice that bit A9 of the address register overlaps the entire sector register in the figure. Bit A9 is a special purpose bit which provides the option of addressing data for one instruction from the residual sector instead of the pre-selected data sector. Because of its purpose, bit A9 is sometimes called the "residual bit". A "1" in bit A9 appears to the memory address decoders as though the data sector register contained all "1's"; this over-rides the register's actual contents. The A9 option is NOT available for addressing instructions; however, instructions can be read from the residual sector by selecting it with the instruction sector register. Similarly, the residual sector can be selected by the data sector register, but this defeats the purpose of the A9 bit.

2-137. Figure 2-40 illustrates the division of a memory array into sectors. Both syllables of the array are identically divided because syllable selection is independently controlled and constitutes a separate facet of memory addressing. The sectors in each syllable are assigned the octal addresses 0 through 17; sector 17 is designated the residual sector. Each sector is 8 x 32 cores in width and length comprising 256 individual locations. Sector 07 is expanded in the figure to show the addresses of the locations within the sector. Every sector, except the residual sector, is assigned the same set of addresses, 0 through 377. The locations in the residual sector are assigned addresses 400 through 777.

2-138. The addresses assigned to the residual sector seem to contradict figure 2-39. But, in the figure, bit A9 is excluded from the Location Address in order to show its correlation to the sector register. Thus, addresses 400 through 777 exceed the capacity of the bits which specify the location address. However, each instruction contains a 9-bit operand address and it is convenient to consider all nine bits in describing data addresses. Thus, the addresses in the residual sector become 400 through 777.

2-139. The selection of addressed sectors and of locations within the sectors is accomplished by allowing specific portions of the memory address to control groups of EI drivers. Figure 2-39 gives the correlation between the memory address and the EI drivers, figure 2-40 relates the EI drivers to sectors and addresses in the array. As shown in figure 2-39, the sector address controls selection of the Hi Y EI drivers and limits the range of the Hi X EI driver selection. (If bit 4 of the sector address is a "0", Hi X drivers above 30 cannot be selected, etc.) Thus, the three low order bits of the sector address select identical pairs of sectors along the Y coordinate in each syllable. The syllable selection eliminates one pair of the sectors and bit 4 of the sector address completes the sector selection by determining whether the Hi X EI driver will be selected from the upper or the lower half of the range.

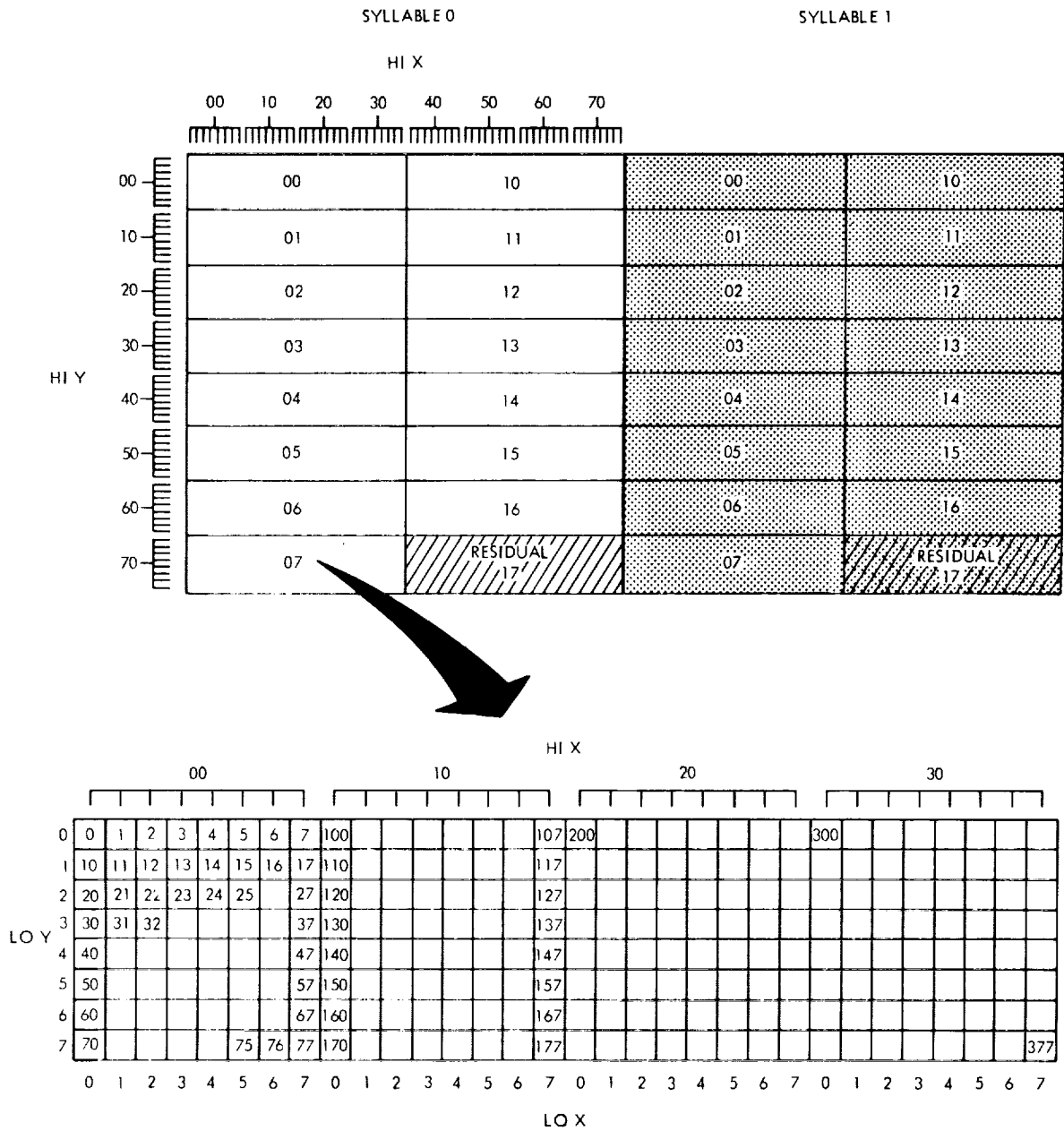


Figure 2-40. Memory Addressing Diagram

2-140. MEMORY CONTROL ELEMENT.

2-141. The memory control element provides the timing, decoding, memory selection and temporary storage operations which are required to properly operate the memory element. The memory control element consists of two buffer registers, a timing and sync selection circuit, address decoding circuits, syllable selection circuits, a memory mode and module selection circuit, and an error monitor and memory select circuit.

2-142. BUFFER REGISTERS. The computer memory readout is destructive, i. e., as data is read from the memory, the applicable memory bits are all driven to "0". The buffer registers hold the data read from memory long enough so that it can be read back in, thereby making the memory readout effectively non-destructive.

2-143. In addition to saving the memory from progressive obliteration, the memory buffer registers feed memory data to the transfer register, and receive data to be stored in memory from the transfer register. The transfer register assumes the role of a shift register when it is receiving data to be loaded into memory. To function as a shift register, the transfer register makes use of multiple clocks. Therefore, special timing is required to lift data from the transfer register as soon as it has been loaded, and before it starts another shift cycle. This timing is provided by the buffer register timing circuits which will be discussed later.

2-144. Two buffer registers are provided; one for even-numbered memory elements, and one for odd-numbered memory elements. Each buffer register is composed of 14 latches - those in the odd memory buffer are labeled BRB- ; those in the even memory buffer are labeled BRA-.

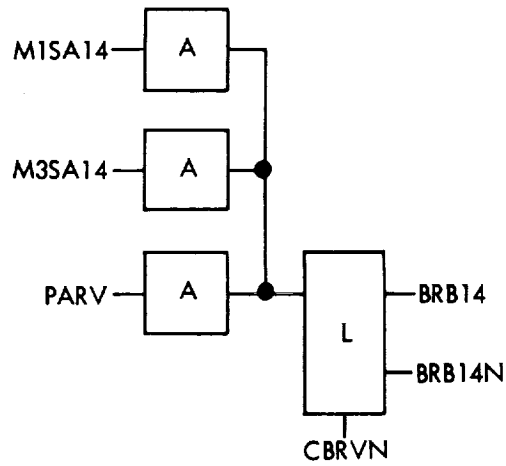
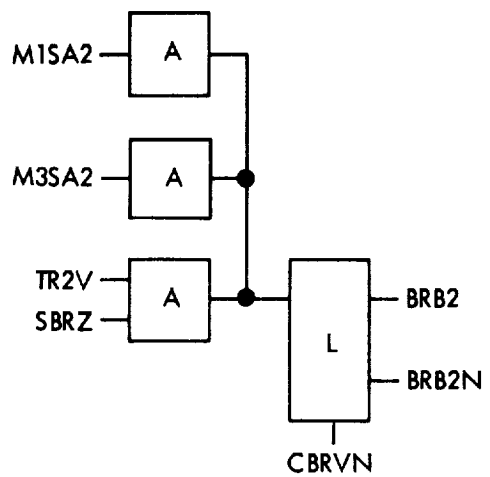
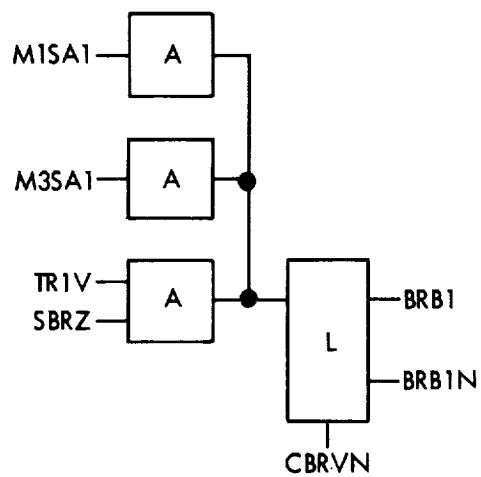
2-145. Figure 2-41 shows latches BRB1, 2 and 14 from the odd memory buffer. The top two AND gates in each of these latches receive the outputs of the appropriate memory sense amplifiers (M1SA1, etc.). These are the gates which set data into the buffer registers during readout.

2-146. The bottom AND gate in each latch (except BRB14) is fed by the corresponding latch in the transfer register (TR1V, etc.). The transfer register input is ANDed with a timing signal from the buffer register timing circuits.

2-147. Latch BRB14 is fed from a circuit called the parity check circuit which will be discussed later. All the buffer register latches are reset by CBRVN, another signal from the buffer register timing circuits.

2-148. The buffer register timing circuits consist of three latches and a gate-inverter circuit (figure 2-42). Latch SBRX is set (on a store command) at the end of phase B (specifically, at phase B, bit 14X time), and is also set at phase C, bit 13X time. A data word coming into the shift register is handled in two 13-bit segments from bit 2 to bit 14 of phase B, and from bit 1 to bit 13 of phase C. (Bit 14 of each segment is supplied by the parity check circuit.)

2-149. Latch SBRX (Set Buffer Register at X time) feeds latch SBR \bar{Y} which, in turn, feeds latch SBRZ. Latches SBR \bar{Y} and SBRZ provide the clock timing required to match the buffer register to the transfer register.

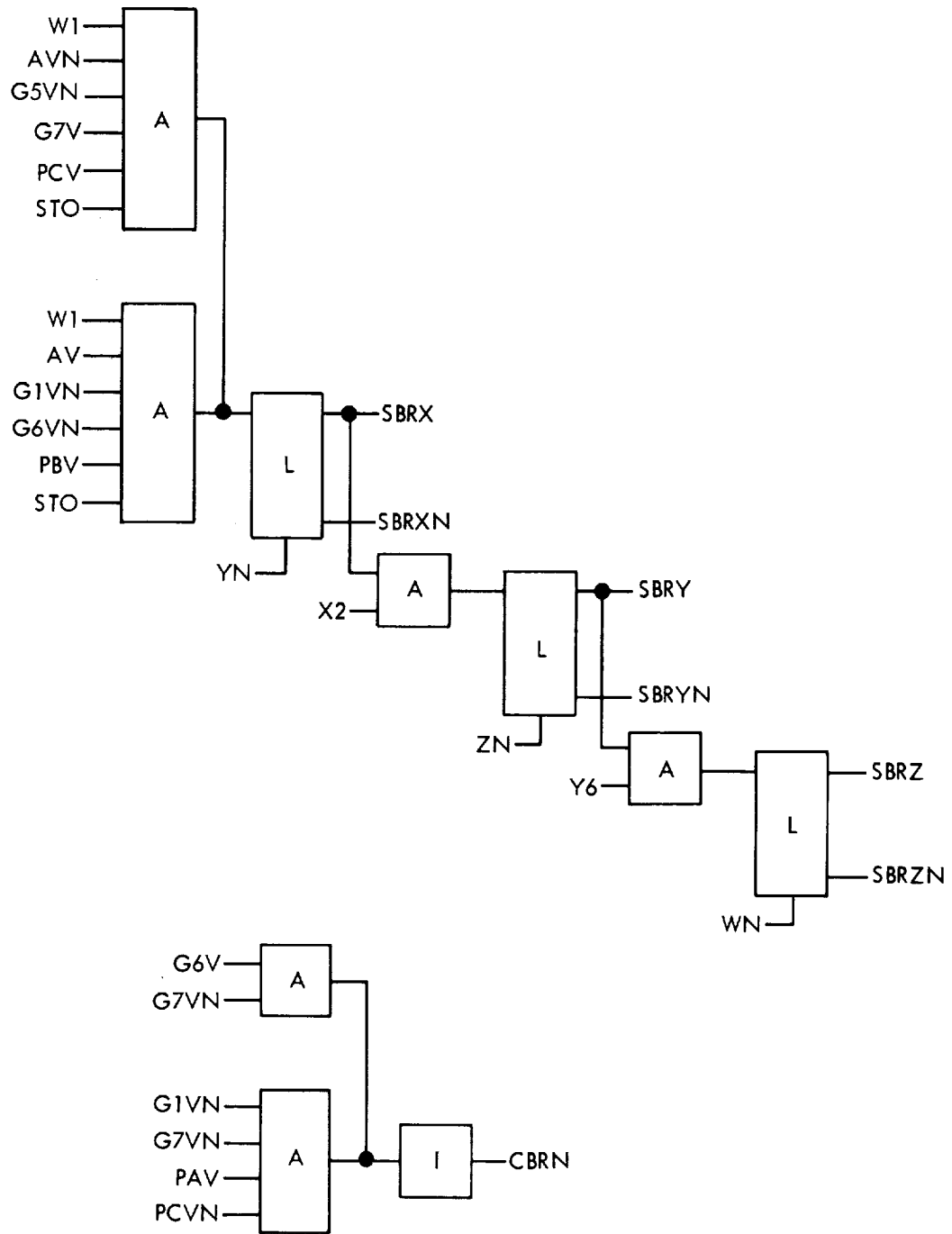


Note: See figure 10-32 for detailed logic.

Figure 2-41. Memory Buffer Register, Typical Latches

2-150. The gate inverter provides a zero-drive pulse (CBRVN) to reset (clear) all the buffer registers at every bit 6 time and at phase A, bit 14 time. These reset times were selected as being optimal for clearing the buffer registers after all read in and read out cycles.

2-151. MEMORY ADDRESS DECODERS. (Figure 2-43) The Memory Address Decoders process memory addressing signals from the Program Control and Data Control Elements into conditioning levels for the memory address drivers in the Memory Element. The memory address decoders consist of four sections, Lo X, Lo Y, Hi X and Hi Y. These decoders reduce the sector and location address of the desired memory location to a 2-bit octal selection code for each coordinate of the memory array. Figure 2-39 shows which parts of the addresses are used in forming each of the codes.



Note: See figure 10-7 for detailed logic.

Figure 2-42. Memory Buffer Register Timing Circuits

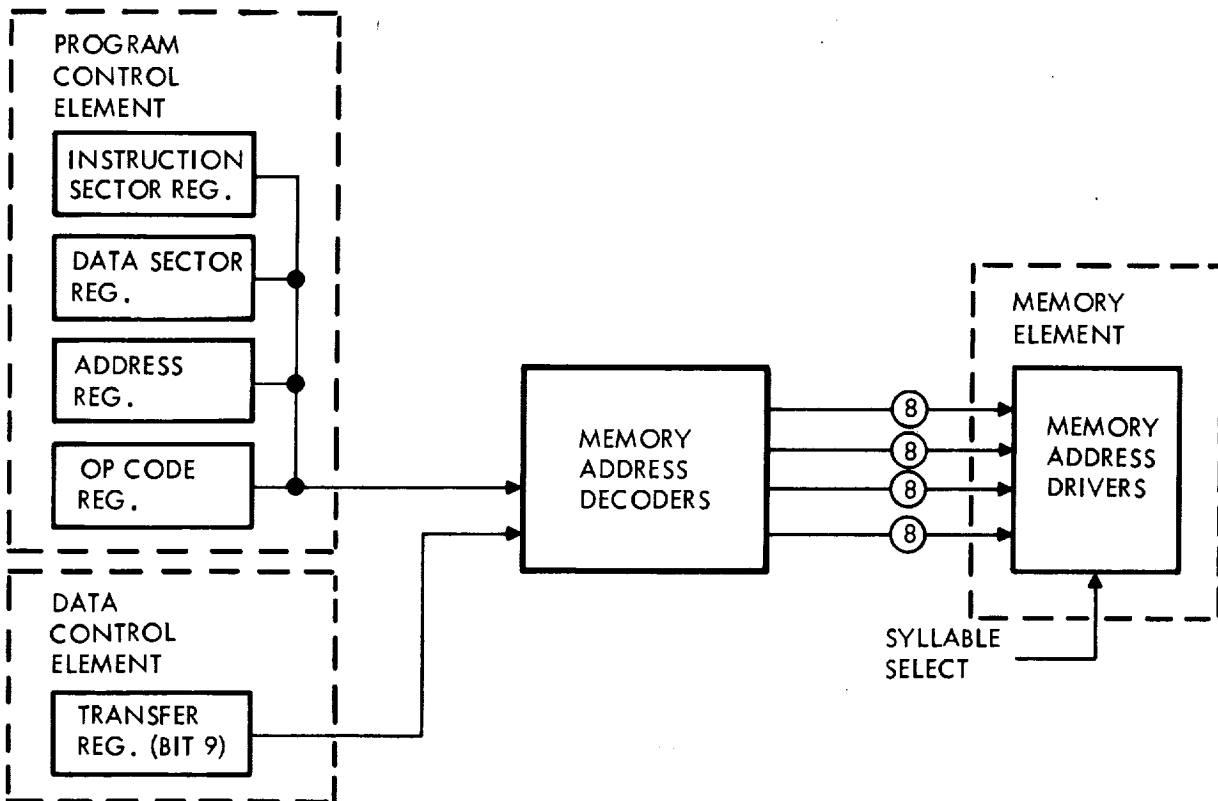
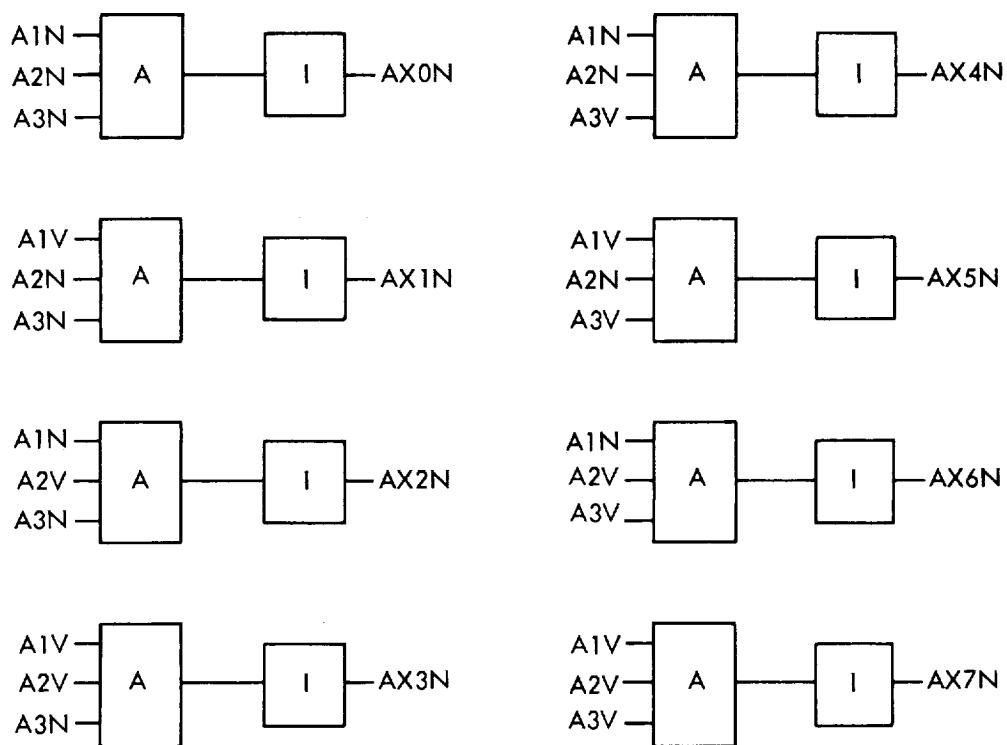


Figure 2-43. Memory Addressing Block Diagram

2-152. Lo X and Lo Y Decoders. (Figures 2-44 and 2-45) The Lo X and Lo Y Decoders each combine three binary bits from the address register to form the low-order digits of the octal selection codes. Both decoders include an AND-Inverter for each of the eight values the three address register bits can represent. When the inputs to an AND-Inverter are all '1's', the inverter produces a '0' output which conditions the appropriate EI driver of each memory module in the Memory Element. Only one AND-Inverter can produce a '0' at a time, each of the remaining seven AND circuits is disabled by at least one '0' input.

2-153. Hi X Decoder. (Figure 2-46) The Hi X Decoder combines bits A7 and A8 of the address register with the most-significant bit of the sector address to form the high-order bit of the X octal selection code. The sector address bit is determined by the instruction sector register during instruction time and by the data sector register or bit A9 of the operand address during operation time. The Hi X decoder consists of eight AND-Inverters, similar to those in the Lo X and Lo Y decoders, and the S4 latch which represents the sector address bit.

2-154. The S4 latch is reloaded each time a new address is transferred into the address register. The Transfer Address (TA) signal gates instruction addresses into the address register at bit-time A-7, and operand addresses at bit-time A-13, of each computer cycle. Concurrent with each address transfer, TA gates the high-order bit of the appropriate sector address into the S4 latch.



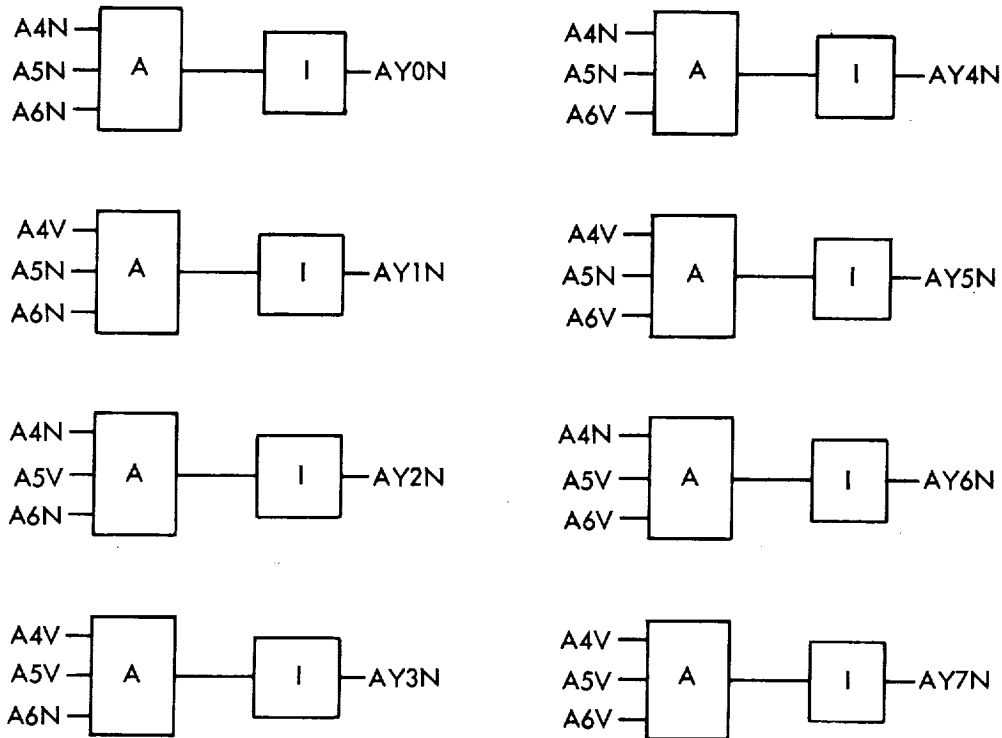
NOTE: SEE FIGURE 10-16 FOR DETAILED LOGIC

Figure 2-44. Low X Decoder

2-155. Inputs IS4 and DS4 are the high-order bits of the instruction and data sector registers, respectively. Input IS4 is sensed at A-7 time (TA and G1V) concurrent with the instruction address transfer, and DS4 and TR9V are sensed at A-13 time (TA and G6VN) with the operand address transfer. Since it is sensed during the "transfer address" signal, bit A9 must be sensed from the transfer register rather than from the address register. If A9 is a "1", the S4 latch is set irrespective of the data sector register contents. Note that with the S4 latch set, only outputs AX40N, AX50N, AX60N and AX70N can be enabled; these outputs select the X coordinates of the residual sector.

2-156. During operation of the instruction following an EXM command, special considerations govern selection of the data sector. For that instruction, bits A1, A2 and A9 of the operand address select the data sector, the content of the data sector register is ignored. The EXMDN signal is applied to gate A2 of the S4 latch to prevent it from sensing DS4 for the data sector address during the instruction immediately following an EXM command. Bit A9 of the operand address alone determines the sector address bit.

2-157. Hi Y Decoder. The Hi Y Decoder develops the high-order bit of the Y-coordinate octal selection code from the three least-significant bits of the sector address. Like the Hi X decoder, it selects inputs from the instruction sector register during instruction time and inputs from the data sector register during operation time. For EXM operations, the Hi Y decoder rejects both the instruction and data sector registers to implement more of the special sector addressing features of this instruction. The Hi Y decoder consists of the Phase A Delayed (PAD) latch and the nine AND-Inverters shown in figure 2-47.



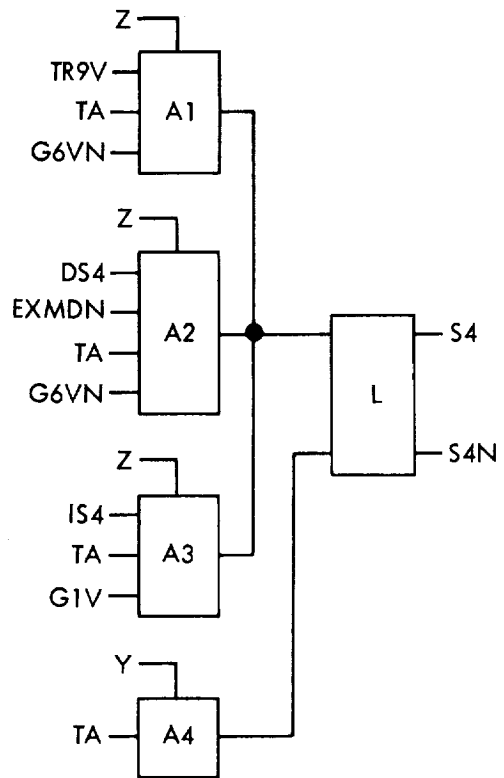
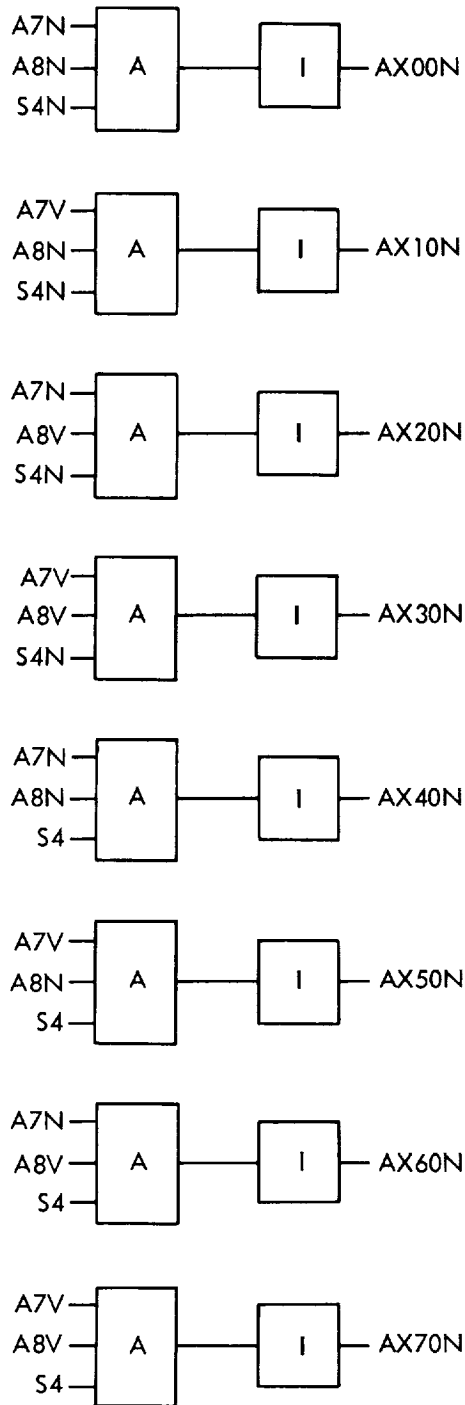
NOTE: SEE FIGURE 10-16 FOR DETAILED LOGIC

Figure 2-45. Lo Y Decoder

2-158. The PAD latch provides the timing signals which discriminate between instruction time and operation time for the Hi Y decoder. The PAD latch is set from A-7-Z time to A-13-Z time except during EXM operations (discussed later); this period spans the time during which instruction addressing signals must remain conditioned. The set output of the PAD latch gates inputs from the instruction sector register into the decoder, and the output of the ninth AND-Inverter, A9PADN, gates inputs from the data sector register into the decoder. During most instructions, bit A9 and EXMD are both "0's"; under these conditions, A9PADN rises to a "1" gating inputs from the data sector register into the decoder as soon as the PAD latch is reset at A-13-Z time.

2-159. If the A9 option is used in an operand address to select the residual sector, A9PADN is disabled and the normal inputs to the decoder are blocked. However, a special gate at the input to the AY70N inverter ANDs bit A9 with PADN and EXMDN. Thus, if it is not instruction addressing time (PADN) and if no EXM operation is underway (EXMDN), A9 forces selection of the AY70N output, the Y coordinate of the residual sector.

2-160. The Hi Y decoder also implements two of the special sector addressing features of EXM operation. First, the instruction following EXM is read from one of four locations in residual memory. Second, the data sector addressed with the instruction following EXM is selected by bits A1, A2 and A9 of the operand address via the Data Sector Modified (DS1M, DS2M) latches. Selection of the residual sector during instruction addressing time is forced in the following manner. The EXMVN signal prevents the PAD latch from being set at A-7-Z time following an EXM instruction; this, in turn, prevents



NOTE: SEE FIGURE 10-16 FOR DETAILED LOGIC.

Figure 2-46. High X Decoder

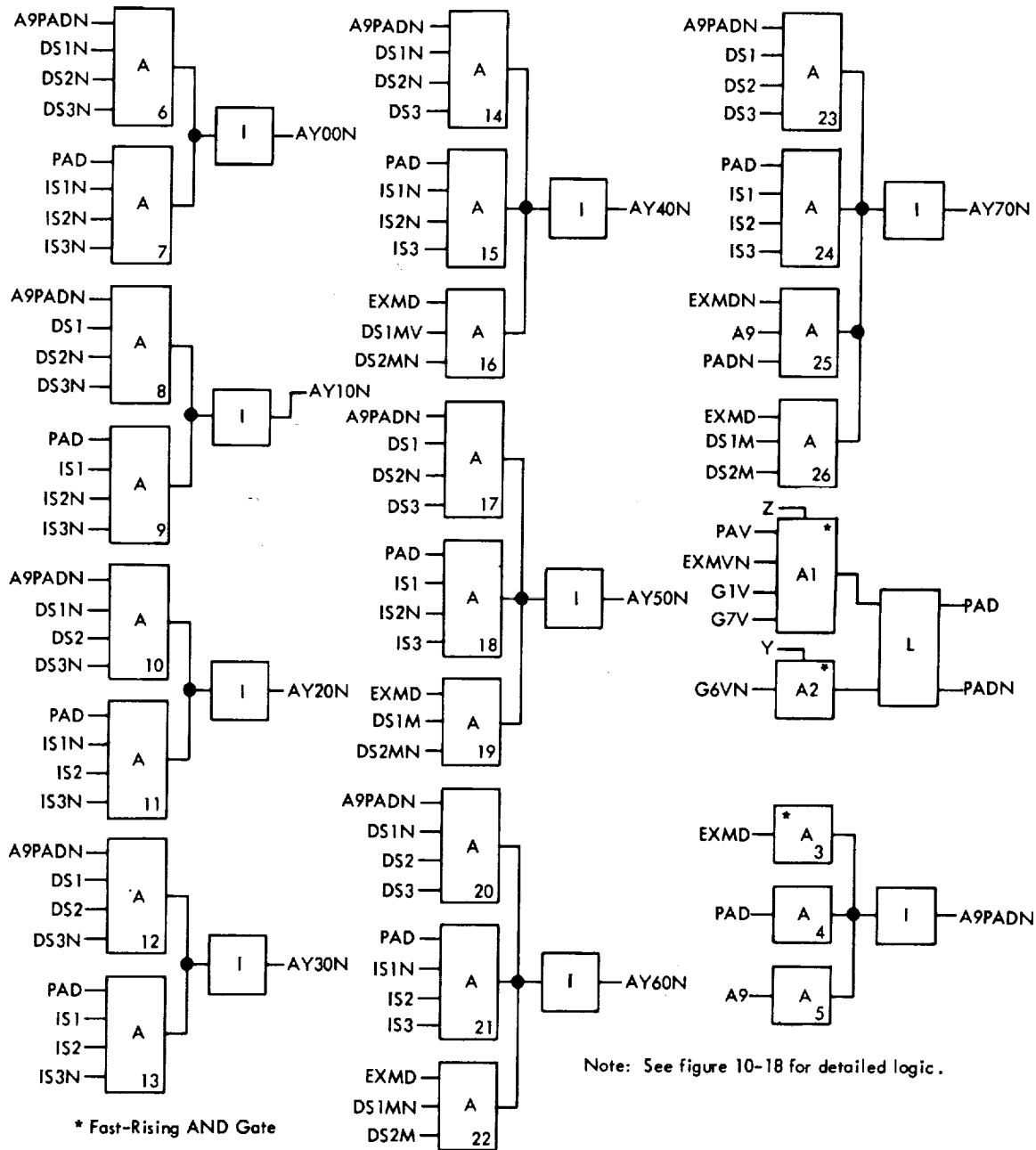


Figure 2-47. High Y Decoder

sensing inputs IS1, IS2 and IS3 for the instruction sector address. The A9 bit of the EXM instruction is retained in the address register which disables the A9PADN signal; this prevents sensing the data sector register outputs. With PAD and A9PADN both disabled, the conditions are the same as when using the A9 option to select the residual memory, except the conditions are now present during instruction addressing time. Selection of the residual sector is completed at the Hi X decoder. The S4 latch is set because a "1" in bit A9 is part of the EXM instruction code. The TA signal which resets the S4 latch prior to sensing IS4 for the next instruction address is inhibited, leaving the S4 latch set. With the S4 latch set, only the Hi X coordinates of the residual sector can be selected.

2-161. During operation time of the instruction following EXM, the DS1M and DS2M latches select the data sector from sectors 04, 05, 06, 07, 14, 15, 16 and 17. The selection is limited to these sectors by preventing the Hi Y decoder from decoding the values 00 through 30 (see figure 2-40). Bit A9 of the operand address controls the Hi X selection directly as previously described.

2-162. To prevent decoding the values 00 through 30, the EXMD signal is applied to the A9PADN inverter. Since the PAD latch was not set during instruction time, both PAD and A9PADN are "0's". This prevents sensing the output of either sector register, thus all the normal inputs to the decoder are blocked. However, the EXMD signal also gates the outputs of the DS1M and DS2M latches into the AY40N through AY70N inverters, enabling these outputs to be selected. Since inverters AY00N through AY30N have no corresponding inputs, they remain blocked and the selection is limited as desired. Note that at the AY70N inverter the EXMDN signal prevents the A9 bit from over-riding the selection made by DS1M and DS2M.

2-163. SYLLABLE SELECT. The Syllable Select circuit determines which syllable of the memory will be addressed with each memory cycle. Included in the syllable select circuit are the Syllable Delayed (SLD) latch and the SYLC1 and SYL0N-SYL1N latches, figure 2-48. The SLD latch performs two syllable select functions. During HOP and transfer operations, it senses the instruction syllable select bit; during EXM operations, it stores the syllable selection for the instruction following EXM. The SYLC1 latch stores the syllable selection for instructions sensed by the SLD latch. The SYL0N-SYL1N latch gates out the instruction syllable selection during instruction time and sequences the selection from syllable zero to syllable one to address both halves of the data word.

2-164. The SLD latch senses instruction syllable selection from two points in the transfer register, TR4 and TR13. The TR13 input is gated at B-9-Z time to sense the syllable select bit of either a HOP or transfer instruction. The SLD latch is reset at B-8-Z time, just prior to sensing, so that a false input from TR4 will not spoil the selection. If the bit is sensed as a "1", the SLD latch is set. The UTR inputs to the SYLC1 latch determine whether or not the new selection will be used. If the selection is valid, UTR will become a "1" at B-10-Y time, gating a double line transfer from the SLD latch into the SYLC1 latch at the following X clock time. The SYLC1 latch cannot then be changed until another HOP or transfer operation is commanded.

2-165. Operation of the SLD latch differs only slightly for EXM operations. Bit A5 of the EXM operand address specifies the syllable from which the next instruction will be read; this bit is sensed by the SLD latch from latch TR4 of the transfer register at A14-Z time. The EXMVN signal prevents sensing TR13, and more important, inhibits re-setting SLD for one computer cycle. At the following instruction time, the SYL0N-SYL1N latch gates out the syllable selection of the SLD latch in place of that from the SYLC1 latch. Consequently, the instruction syllable selection is altered for the one instruction following an EXM.

NOTE: SEE FIGURE 10-13 FOR DETAILED LOGIC

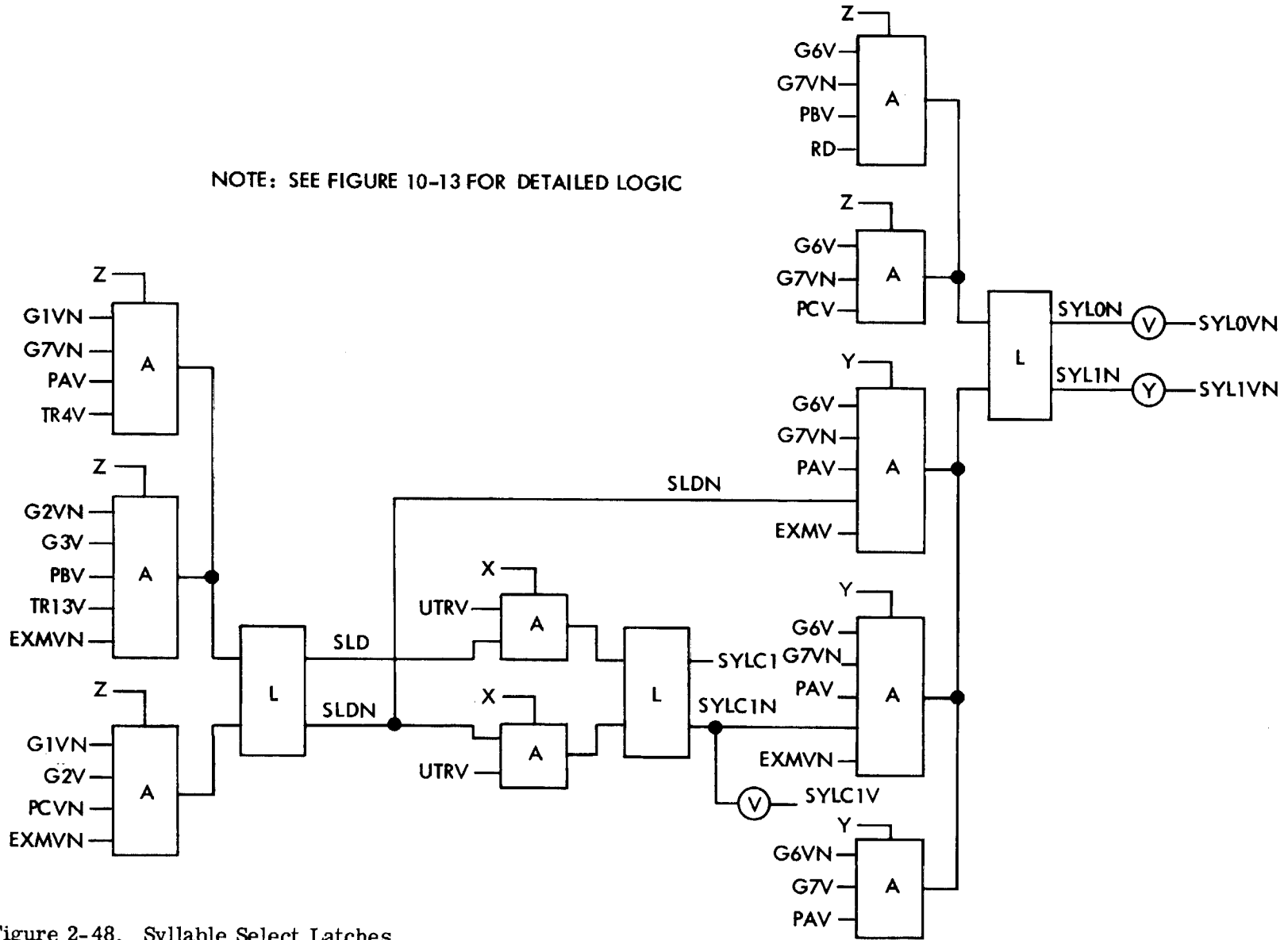


Figure 2-48. Syllable Select Latches

2-166. The outputs of the SYLON-SYL1N latch are inverse selection signals (syllable zero is selected by a "0" on the SYLON signal, etc.) for the Hi X memory address drivers in the Memory Element. The latch is set to select syllable zero at A-13-Y time to address the first syllable of the data word. If data are being read from the memory (RD) SYL1N is driven to a "0" at B-6-Z time to select the second syllable of data. If data are being stored into the memory, syllable one is not addressed until C-6-Z time; this accounts for the difference in timing between read and store operations (see figures 2-51 and 2-52). At the following A-6-Y time, the instruction syllable selection is made. The latch is already set to select syllable one and is merely reset if syllable zero is to be selected. The instruction syllable is controlled by the SYLC1 latch or by the SLD latch is an EXM operation is underway.

2-167. MEMORY SYNC TIMING. (Figure 2-49) The Memory Sync Timing circuit determines when the Memory Element will be operated and whether information will be transferred to the Memory Element, or from the Memory Element to the central computer. When a memory operation is required, the memory sync timing circuit applies a control signal to the Memory Element to establish the direction of transfer, then issues a sync impulse to start the transfer. The sync impulse is routed through the memory mode and module select circuit to the selected memory module(s). Included in the memory sync timing circuit are the Memory Operate (MOP) gate, the Read Memory (RDM) latch, the Read (RD) gate, the INHiBit Strobe (INHBS) gate and the SINK and SYNC latches. The SINK latch under control of the MOP, INHBS and RD gates produces a series of timed outputs which, after buffering through the SYNC latch, become the "start memory" signals for the memory modules. The RD gate enables certain of the SINK outputs which are peculiar to read operations; the INHBS gate performs a corresponding function for store operations. The MOP gate decodes the conditions under which the memory is not operated e. g. operation time of instructions which require no data, and inhibits the SINK latch when such a condition exists.

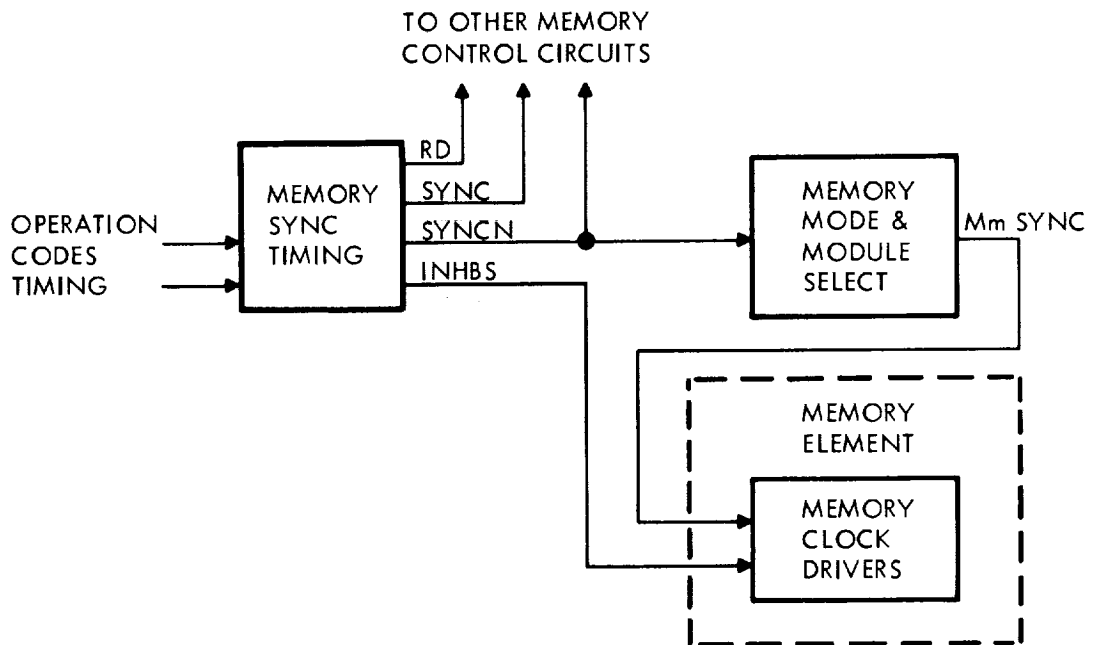


Figure 2-49. Memory Sync Timing Block Diagram

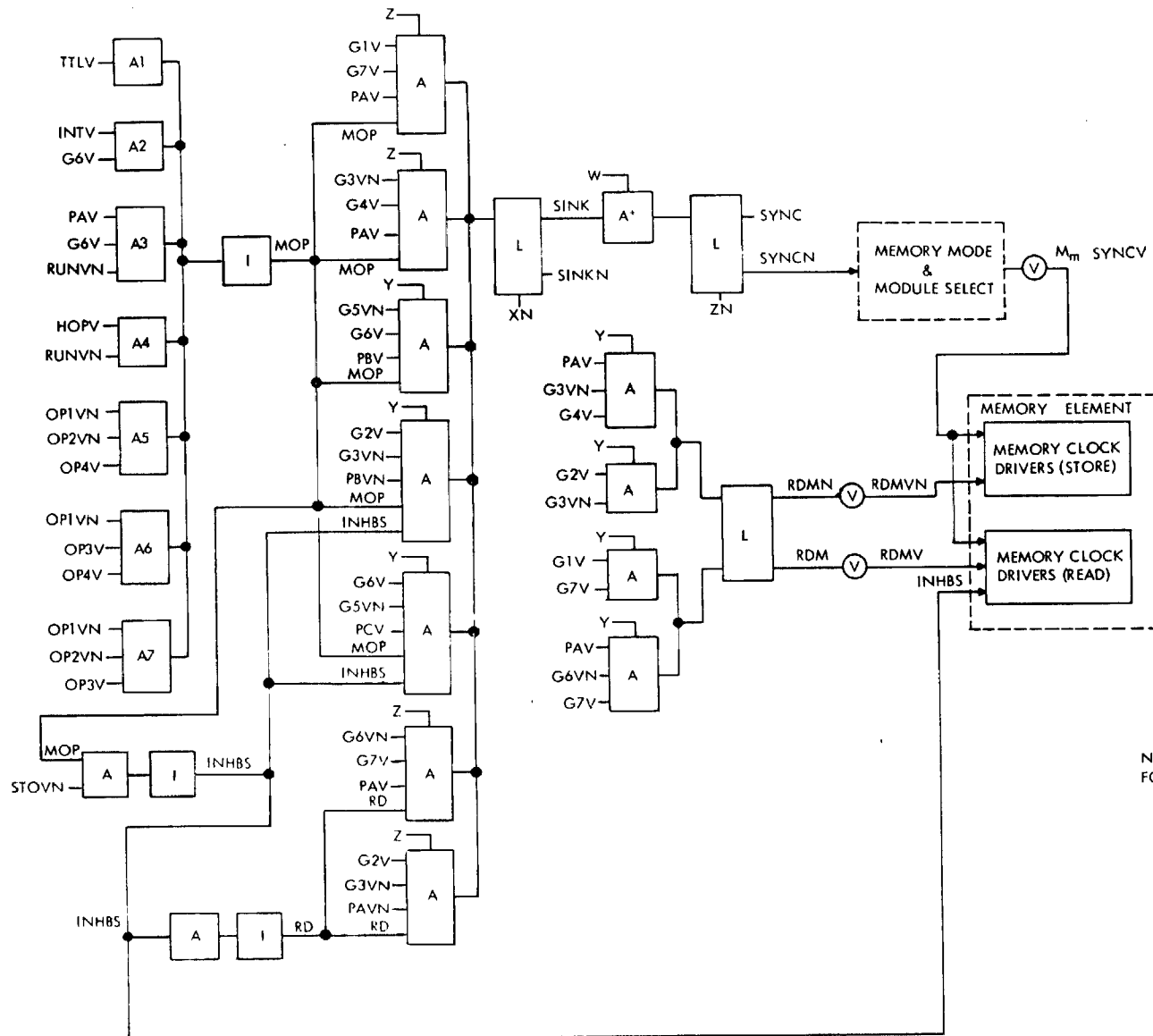
2-168. Figure 2-50 illustrates the functioning parts of the memory sync timing logic and, in phantom, the circuits they feed. Note first that MOP gates every set input to the SINK latch; it is applied directly to all of the ANDs except two and is present (after double inversion) at those two in the RD signal. Each of the input ANDs to the MOP inverter represent some condition which prevents the memory from cycling. The TTLV input identifies selection of memory address 775; since this address specifies the PQ register in the Multiply-Divide Element, therefore the memory is not cycled. Input gate A2 prevents reading the memory during the instruction time following recognition of an interrupt; this results in the HOP instruction which diverts the program to the interrupt processing subroutine. Input gate A3 senses that the computer is halted and inhibits the memory to force the first instruction of the program, HOP 000. Once the HOP instruction has been forced, gate A4 prevents all further memory operation until the computer is allowed to run. Gates A5, A6 and A7 decode the following instructions which do not require data from the memory:

CDS	TMI
EXM	TNZ
SHF	TRA

2-169. Each input gate for the SINK latch represents a different time (or in some cases times) when an impulse must be developed for the memory. The SINK latch is reset at every X clock-time by the XN zero-drive input and is set with the Y or Z clock of the bit times selected by the input gates. Figures 2-51 and 2-52 are timing diagrams which show the outputs of the memory sync timing circuit for reading information from the memory and for storing into the memory. Both figures include the outputs of the memory clock drivers for convenience. Each time a transfer to or from the memory is affected, the SINK latch issues two outputs; the first output initiates a read cycle in the memory, the second initiates a store cycle. During read operations the read cycle transfers information out of the memory and the store cycle restores the information to the memory location. During store operations the read cycle produces no output and consequently, clears the memory location by destructive readout. The store cycle then transfers new information into the memory. The RDM latch is timed to run in step with the SINK latch so that the memory clock drivers are conditioned to produce read and store timing pulses on the appropriate alternate SYNC outputs.

2-170. By comparing the two timing diagrams it is apparent that some of the times at which SYNC outputs occur are common to both read and store operations, while others are peculiar to one operation or the other. The SYNC outputs which occur at C-13 time and at A-3 time are peculiar to store operations. When a store instruction is in progress, the STOVN signal causes INHBS to be a "1"; this conditions the two gates which produce the outputs peculiar to store and conditions the RD signal. With the RD signal a "0", the two gates producing the outputs peculiar to read operation are disabled. During read operation the functions are reversed. The SYNC latch provides an inverse output to the memory mode and module select circuit of the proper time duration to change the inputs to the memory modules.

2-171. MEMORY MODE AND MODULE SELECT CIRCUIT. The Memory Mode and Module Select Circuit selects either individual memory modules (simplex) or pairs of memory modules (duplex) under direction of signals from the Program Control Element. Separate selection of data and instruction modules is provided.



NOTE: SEE FIGURE 10-13 FOR DETAILED LOGIC.

Figure 2-50. Memory Sync Timing Circuit

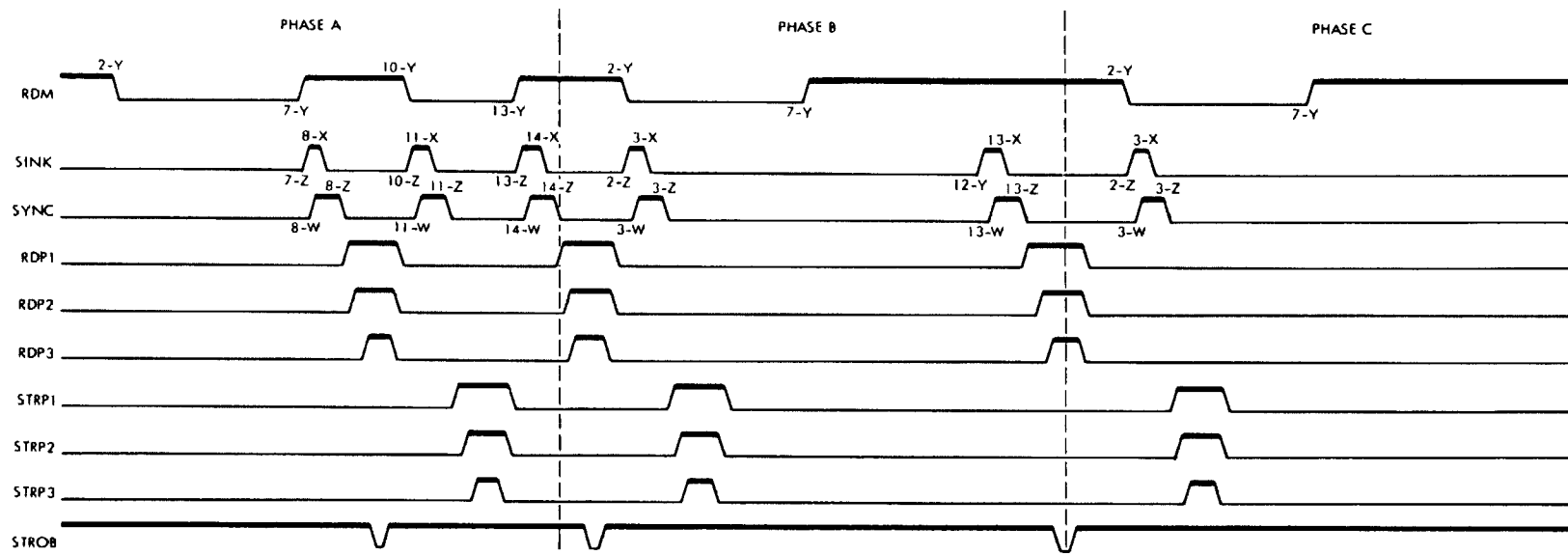


Figure 2-51. Memory Read Sync Timing Diagram

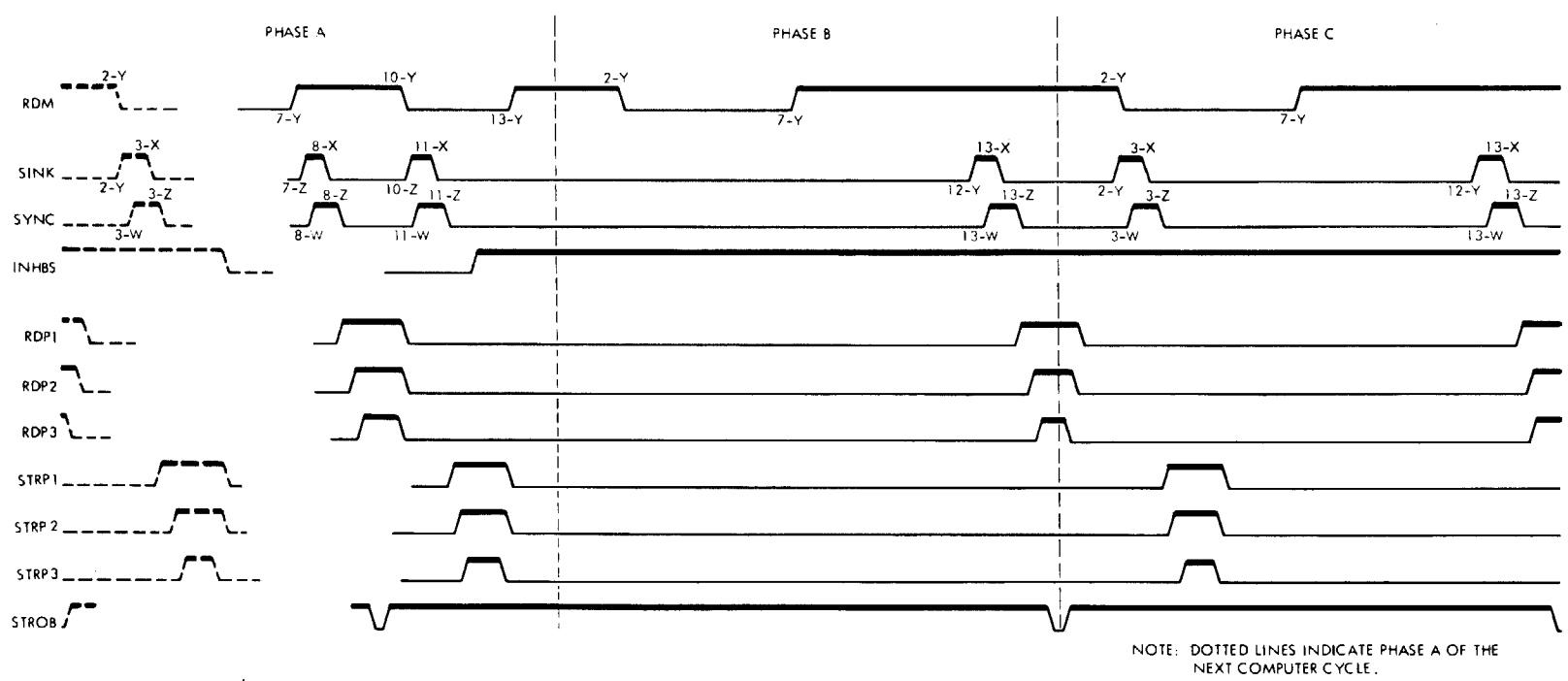


Figure 2-52. Memory Store Sync Timing Diagram

2-172. The memory mode and module select circuit consists of the Memory Module Registers, the Memory Module Select Control Circuit and the Memory Module Select Gates (figure 2-53). The memory module registers, under command of the Program Control Element, decode certain bits of the HOP constant and the CDS operand address to determine what modules will be selected for data and instructions. The Memory Module Select Control Circuit provides the proper timing to select either data or instructions from the modules selected in the memory module registers. In addition, the memory module select control circuit provides proper signals for simplex or duplex operation. Outputs from both the memory module registers and the memory module select control circuit are fed to the Memory Module Select gates which decode them to select the appropriate modules.

2-173. Memory Module Registers. (See figure 2-54.) These are two memory module registers, one for data (DM), and one for instructions (IM). The data module register monitors the HOP constant and the CDS operand address (CDSV) under command of the program control element (DSS). Briefly, the data module register senses four bits, 1 for mode and 3 for module selection. The mode bit is sensed first, then bits 1, 2 and 3 of the module selection code.

2-174. At bit time six, the mode bit is sensed (figure 2-54) and, if a "1", drives tratch output DUPDN to a "0", indicating duplex operation. If the mode bit is a "0", the tratch will be driven to DM1 = "0", which for the moment, establishes only that the mode is not duplex (DUPDN = "1"). At bit time seven, the first module selection bit is sensed. If it is a "0" the tratch remains set to DM1 = "0" or more pertinently, DM0 = "1". If the first module selection bit is a "1", the tratch is driven to DM0 = "0" (DM1 = "1"). To summarize, if a duplex mode is called for, DUPDN will be a "0" and both DM0 and DM1 will be "1's". Otherwise the tratch will indicate the status of the low order bit, DM0 or DM1.

NOTE

A command for duplex operation could be overridden by a "1" in the first module selection bit. Therefore, if duplex operation is required, the first module selection bit must be a "0".

2-175. Latches DM2 and DM3 (figure 2-54) sense the remaining two module selection bits in straightforward fashion. Both latches are cleared one bit time before their respective module selection bits are sensed. If the bits are "1's", their respective latches will be set.

2-176. The operation of the instruction module register is the same as the data module register, except that the bits appear and are sensed in the following order: bit 2, bit 3, mode bit and bit 1.

2-177. Memory Module Select Control Circuit. (See figure 2-56) Instructions are read from memory essentially during phase A and data is read during phases B and C. The timing which discriminates between instruction and data module selection is provided by two latches, the read instruction latch (REI) and the read data latch (RED), both part of the memory module selection circuit.

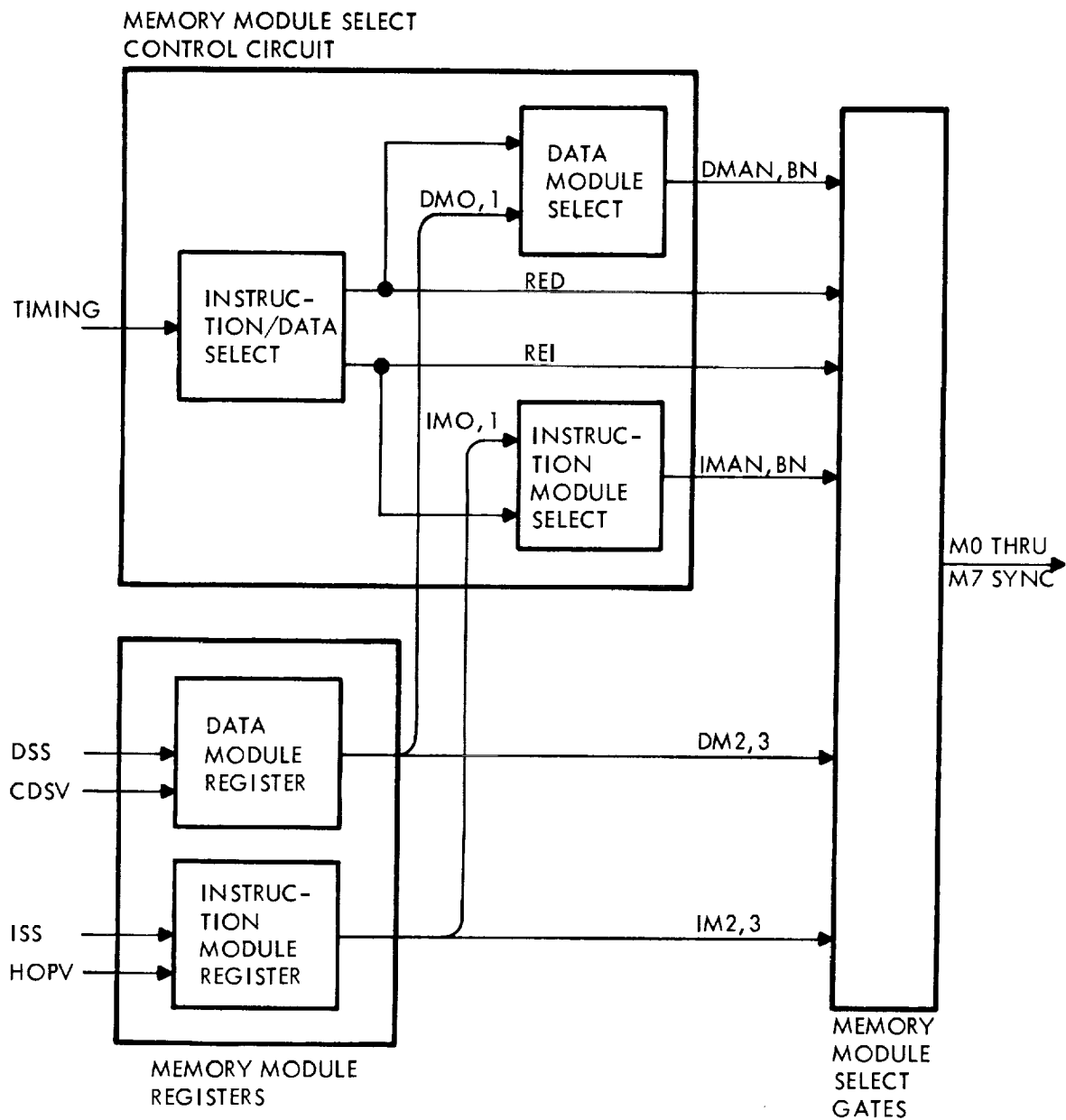


Figure 2-53. Memory Mode and Module Select Block Diagram

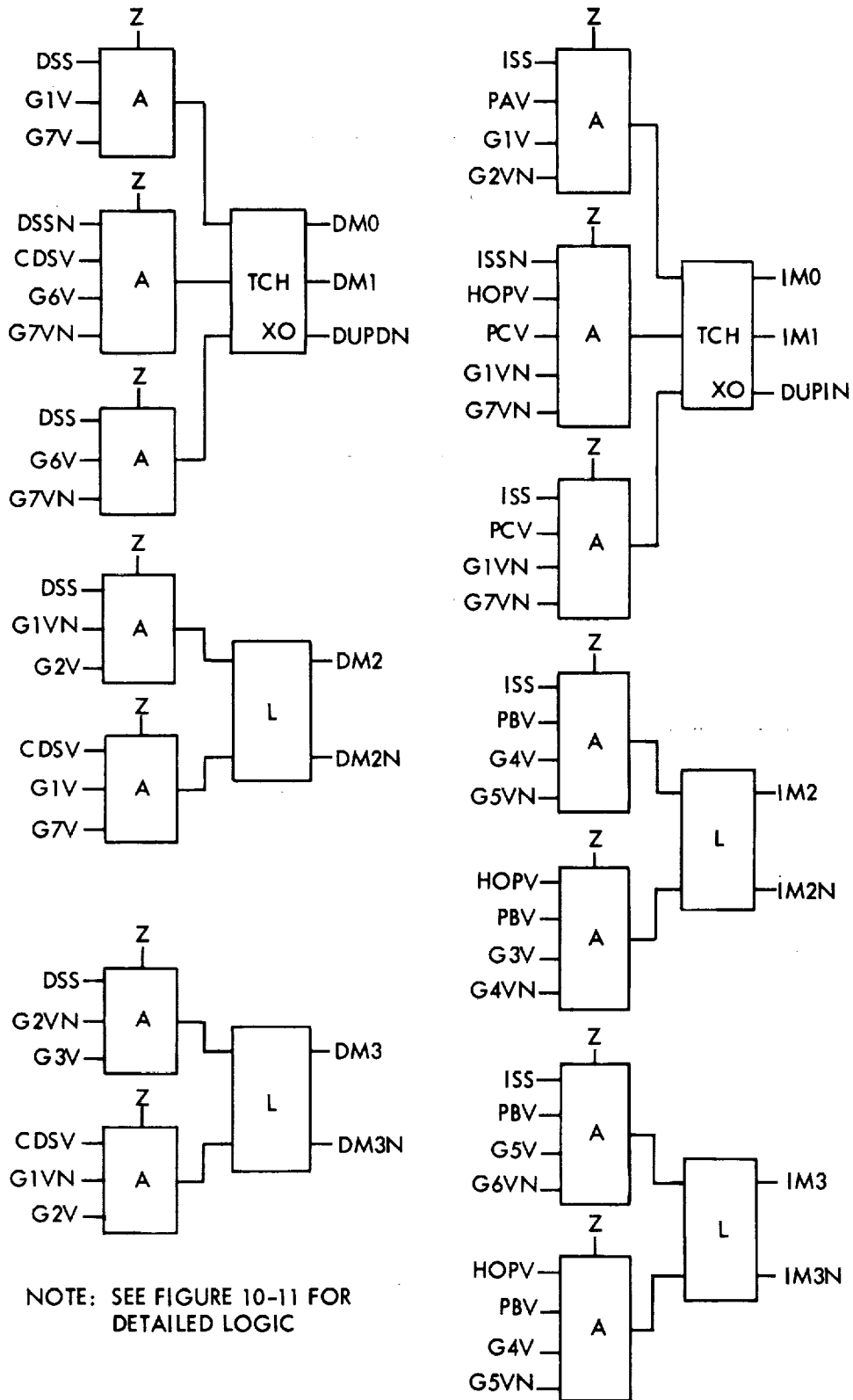


Figure 2-54. Memory Module Registers

2-178. The latch outputs are combined with the low order bit signal (DM or IM, 0 or 1) from the memory module registers to produce signals IMAVN and BN and DMAVN and BN. These signals are inversions of latch outputs DM or IM, 0 and 1 at the times selected by latches RED and REI.

2-179. Memory Module Select Gates. (See figure 2-55.) The two high order bits of the module selection code are decoded by part of the memory module select gates to produce MZO (module zero or one), MTT (modules 2 or 3), MFF (modules four or five) and MSS (modules six or seven). The outputs of these gates are combined with the timed low-order bits DMAVN, BN and IMAVN, BN to enable the gate-inverters M0 SYN through M7 SYNC. When these gate-inverters are enabled, they will pass and invert a synchronizing signal, SYNCN, to initiate memory module operation.

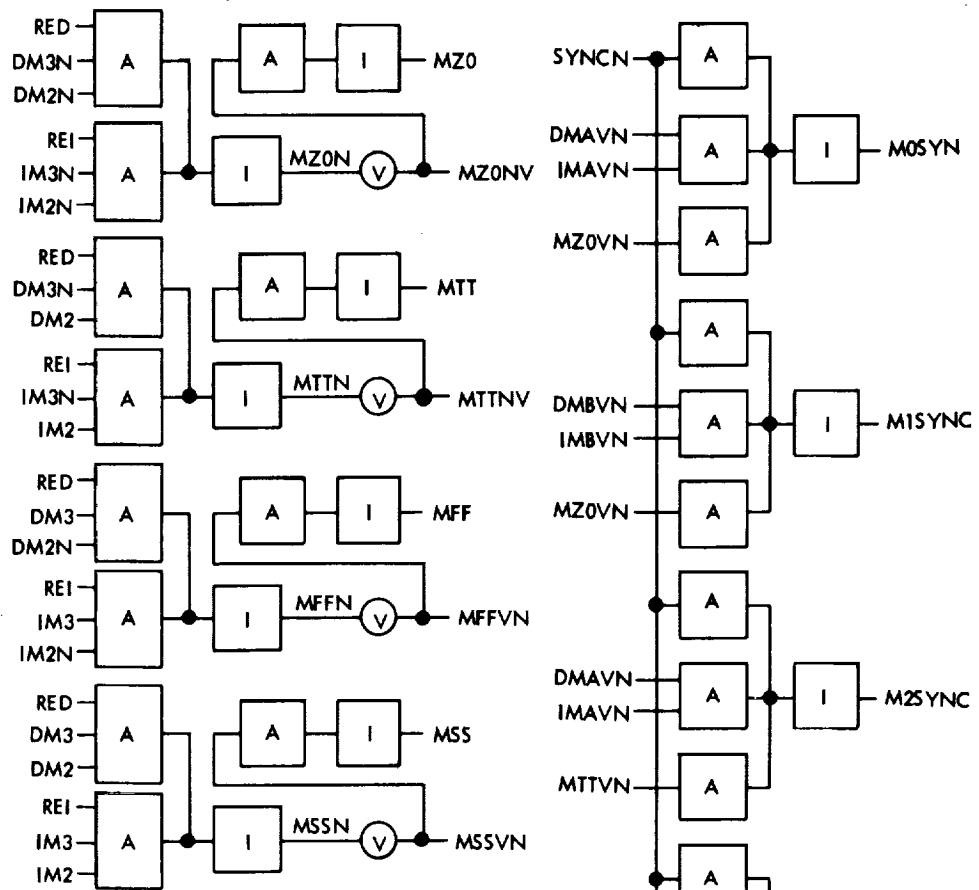
2-180. Assume that AZOVN is a "1" and that data is to be read in simplex mode. At phases B and C (essentially), RED will be a "1" and will generate either DMAVN or DMBVN = "0". If DMAVN is a "0", SYNCN will be free to drive M0SYN. If DMBVN is a "0", SYNCN will be free to drive M1SYN.

2-181. In duplex mode, both DMAVN and DMBVN will both be "zeros" in the example illustrated; in this case, both M0SYN and M1SYN will be driven by SYNCN.

2-182. MEMORY SELECT AND ERROR MONITOR. The Memory Select and Error Monitor circuit continuously checks the performance of the memory. During duplex operation, the memory buffer selection is controlled to correct transient errors and provide an uninterrupted flow of reliable information to the computer. Memory errors are detected by parity checking the buffer registers after every read cycle and by monitoring the outputs of the error detector circuits in the memory modules.

2-183. In duplex operation the A and B memories are cycled simultaneously but only one buffer register is selected to provide information to the computer. If an error is detected during a read cycle, the memory buffer containing good information is selected to control restoring both memories; thus the error, if transient, is corrected. If an error occurs in the memory selected to feed the computer, the selection is changed to the correct buffer before the information is used. Thus the computer receives only reliable information unless both memories fail at the same storage location simultaneously. Operation is not restored to the previously selected memory unless the good memory should develop an error. Monitor signals are provided at the computer interface which indicate the status of each memory and simultaneous failure of both memories.

2-184. The memory select and error monitor circuit includes separate parity check and error-detector sensing circuits for each memory; these circuits are called the "error monitors". The error detectors in the memory modules indicate three types of addressing failures; 1) multiple address selection; 2) loss of half-select current and 3) spurious selection during non-cycle times. These failures are combined into two types, "on current failures" and "no current failures" by the error-detector sensing circuits. Timing signals which enable the error monitors to sense the different types of errors at the appropriate times are common to both error monitors. The TIME latch gates outputs from the parity check circuits, and the Check On Current (COC) and Check No Current (CNC) latches gate their respective types of failures, into the monitor circuits. Outputs of the A and B monitors are combined in the TLC latch to indicate simultaneous failure of both memories. Memory buffer selection is performed by the MAO-MBO, BRAO and BRBO latches under control of the monitor outputs.



NOTE: SEE FIGURES 10-11, 10-13
AND 10-14 FOR DETAILED
LOGIC .

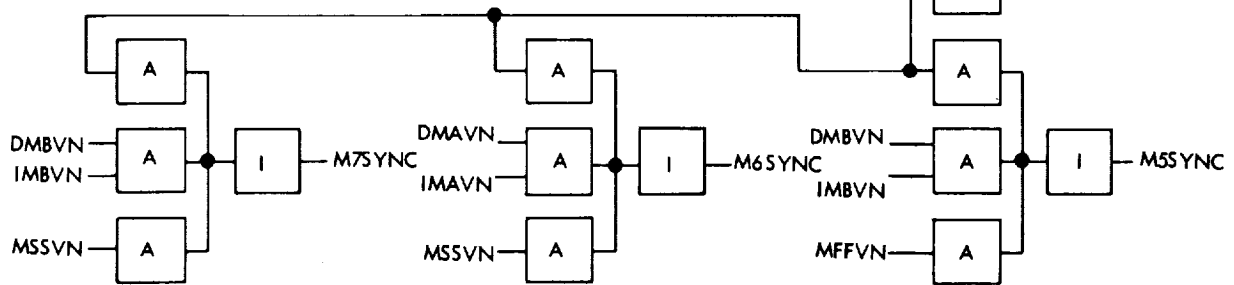


Figure 2-55. Memory Module Select Gates

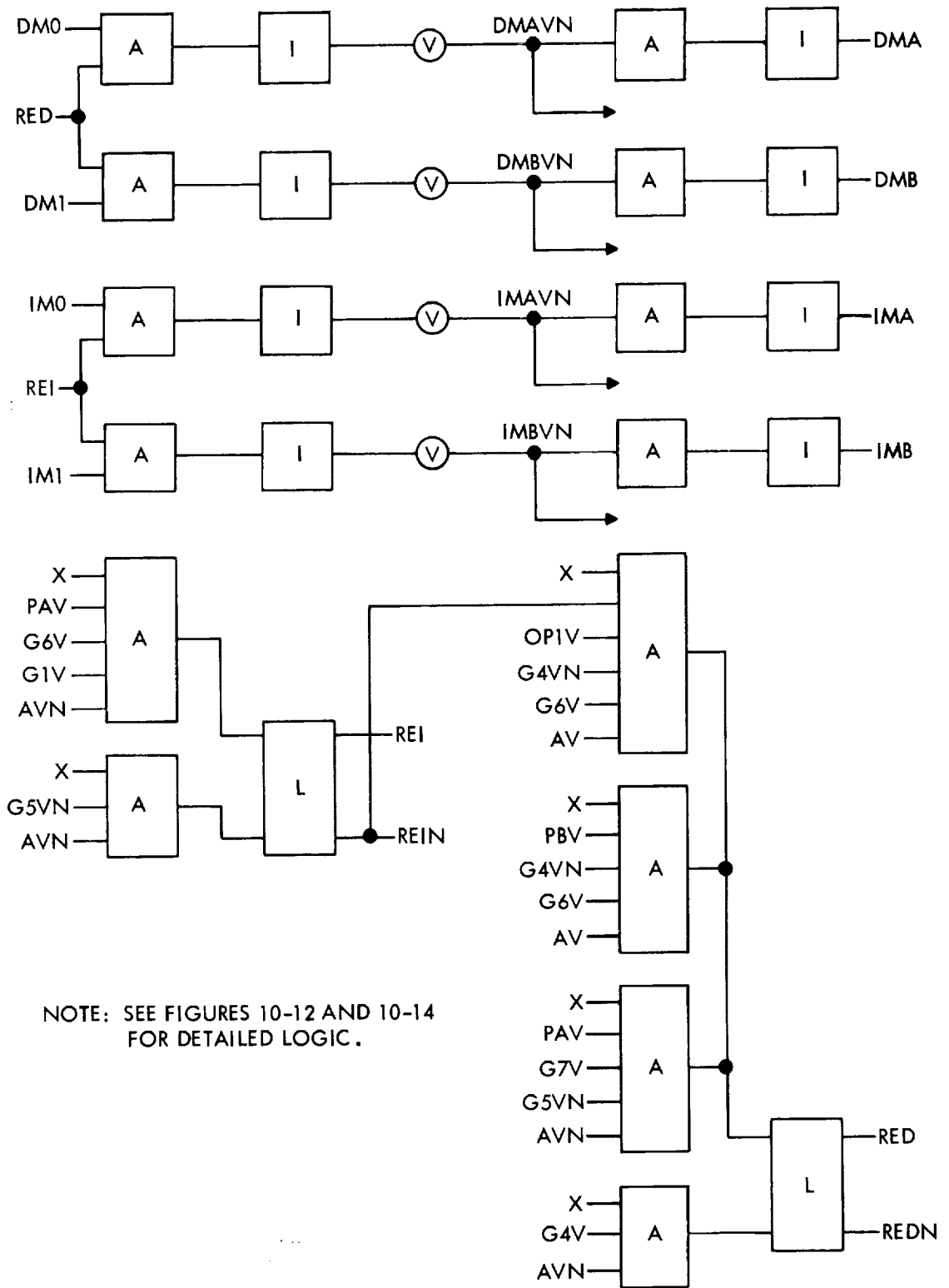


Figure 2-56. Memory Module Select Control Circuit

2-185. Parity Check Circuits. Parity checking is implemented in a pair of exclusive OR "trees", one for each memory buffer. The fourteen bits of each memory buffer are parity checked in parallel and an Error A or B Parity (EAP or EBP) signal generated if the fourteen bits contain an even number of "1's". Figure 2-57 shows the two configurations of AND circuits and inverters used to develop the exclusive OR (XOR) function. The circuit shown at A is used when complementary pairs of inputs are available for each bit. The circuit shown at B is used where only one signal is available for each bit. In either case, output C is a "1" when one and only one input (A or B) is a "1".

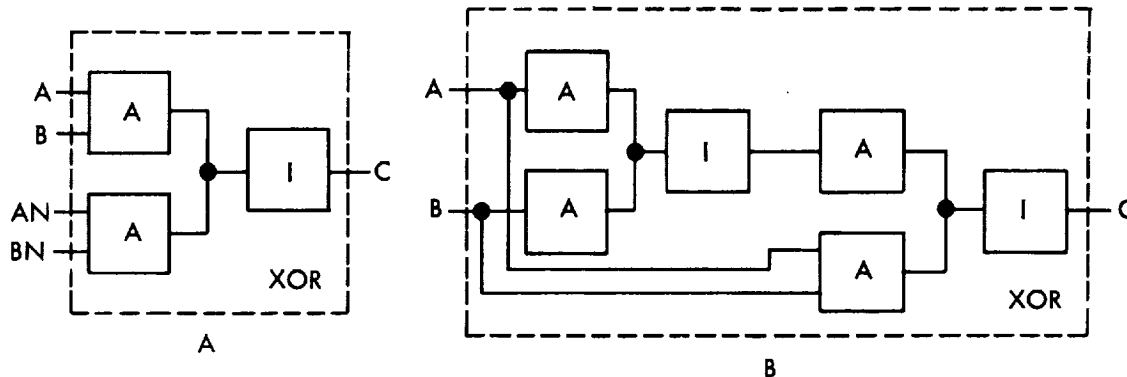
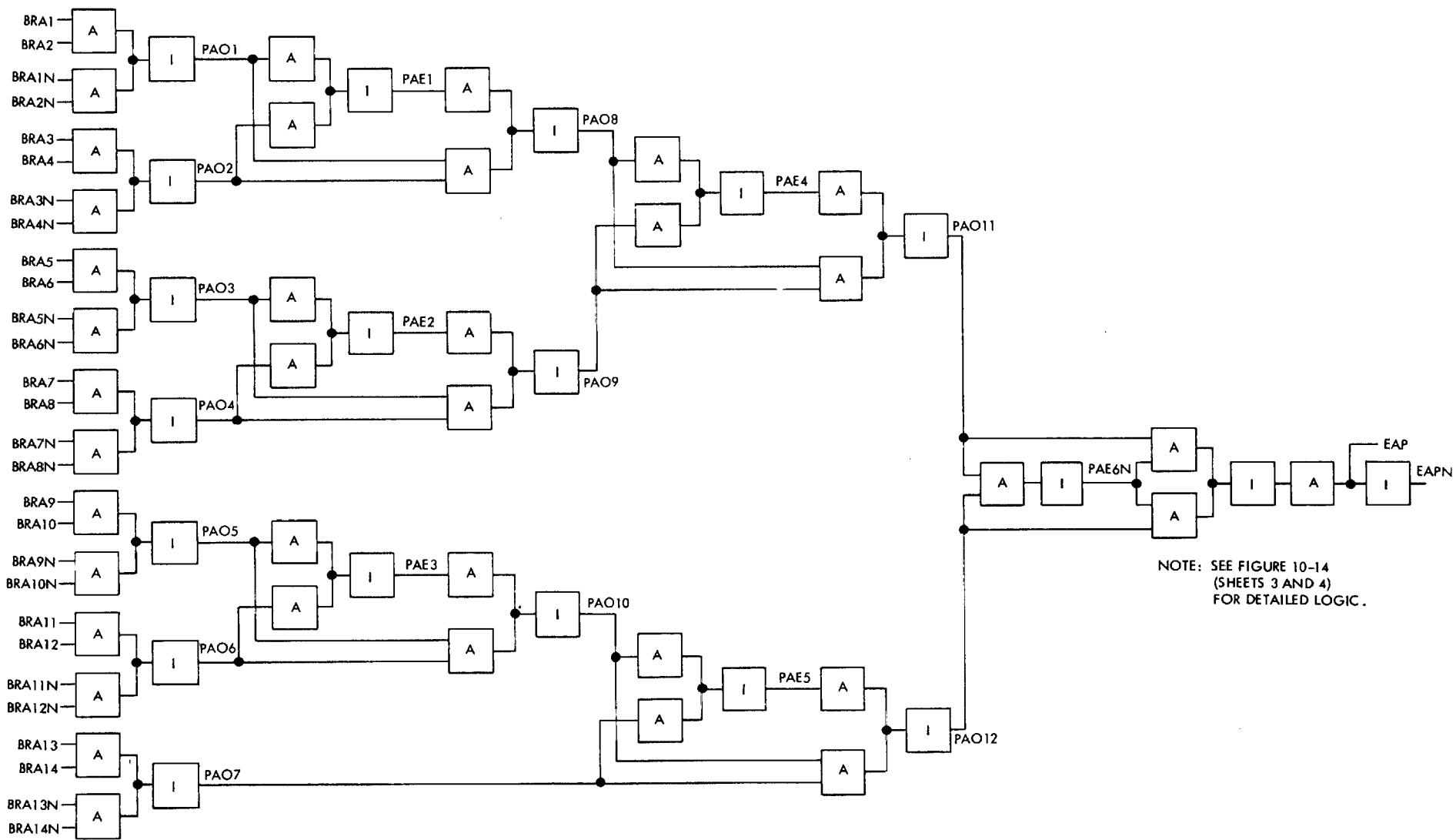


Figure 2-57. Exclusive OR (XOR) Circuits

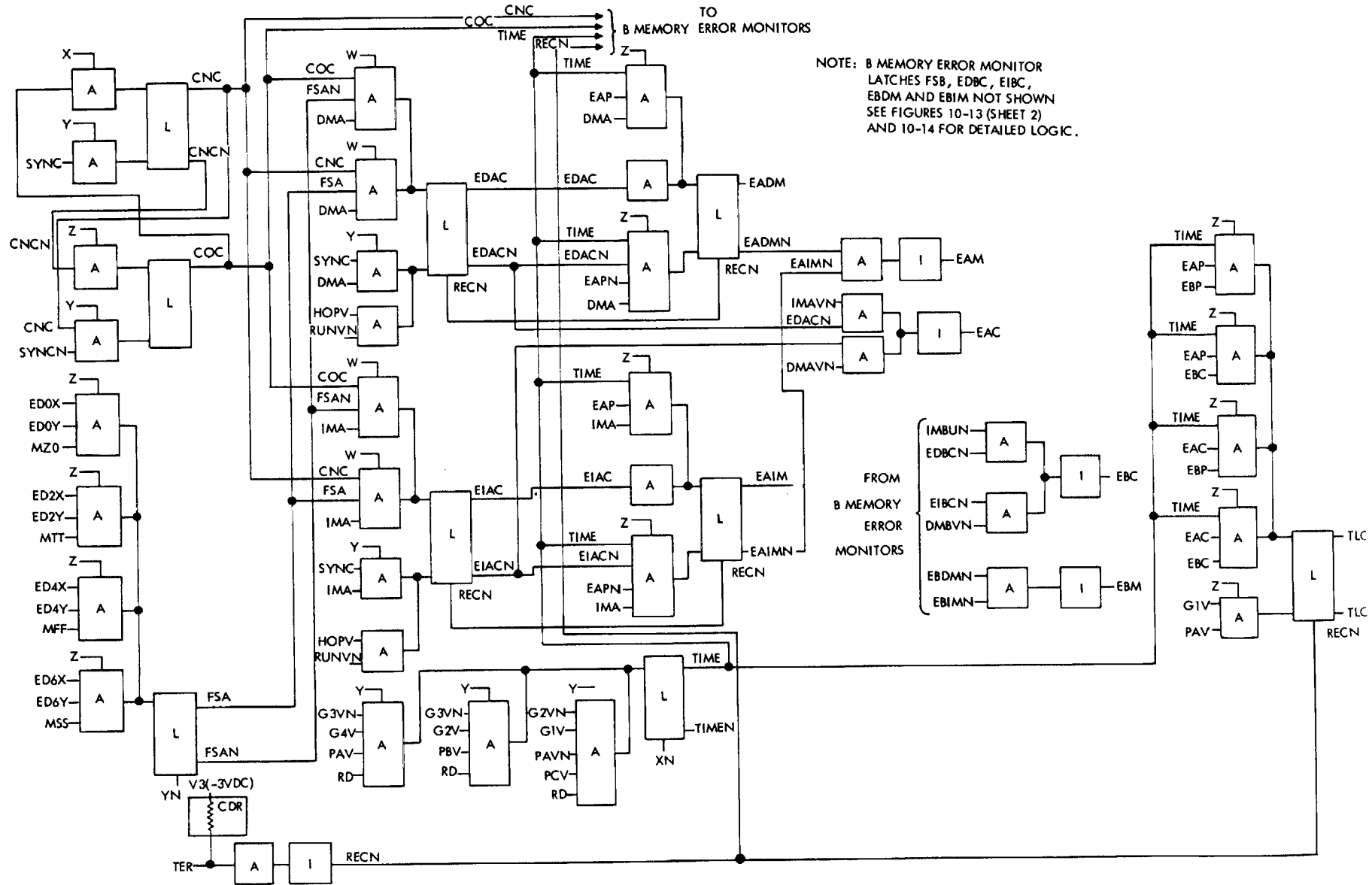
2-186. Figure 2-58 depicts the exclusive OR tree which checks the parity of the A memory buffer. Outputs from the memory buffer are combined into seven pairs, each pair feeds an exclusive OR circuit at the input to the tree. The output of the exclusive OR represents the parity of its inputs. For example, if BRA1 is a "0" and BRA2 is a "1" (the pair has odd parity) then PAO1 is a "1" (also odd parity). The outputs of the exclusive OR circuits are then paired to feed the next stage of the exclusive OR tree and continue until only two outputs remain; each output representing the parity of half the memory buffered. If the parity of the memory buffer is correct (odd), one of the bits must be a "1" and the other bit a "0". The "0" bit disables the input to the PAE6N inverter which drives PAE6N to a "1". The bit which is a "1" then enables one of the input ANDs for the EAP inverter, driving the error signal to a "0".

2-187. Memory Error Monitors. The Memory Error Monitors sense the outputs of the parity check circuits and the error detectors for the A and B memories to determine when a failure has occurred. The circuits which are separate for each memory are identified by an A or B in the names of the output signals. Figure 2-59 shows the error monitor circuit for the A memory including the circuits which are common for both memories. The error signals are developed in four stages. First, the outputs of the error detectors are sensed and stored in the FSA latch. Second, the set and reset outputs of the FSA latch are sensed at the appropriate times to detect on-current and no-current failures. A separate check is made for data and instruction modules by the EDAC and EIAC latches, respectively; this provides the timing discrimination necessary to distinguish normal on-current in one module from a no-current failure in the other, etc. If an error is detected at this stage, it is passed on to the third stage of the monitor which senses parity errors. The EADM and EAIM latches are set immediately by an error sensed in the preceding stage or at the proper time if a parity error occurs.



NOTE: SEE FIGURE 10-14
(SHEETS 3 AND 4)
FOR DETAILED LOGIC.

Figure 2-58. A Memory Parity Check Circuit



The "0" outputs of the EADM and EAIM latches are combined in the fourth stage of the monitor (EAM) to indicate an error in either the data or instruction portion of the A memory. A similar signal (EBM) is developed for the B memory; these signals permit monitoring the status of each memory individually at the computer interface.

2-188. Another AND-7 inverter circuit combines the outputs of the two second stage latches EDAC and EIAC, to form the Error A Current (EAC) signal. A corresponding signal is developed for the B memory. The TLC latch combines these signals with the outputs of the two parity check circuits to detect simultaneous failures in both memories. The TLC latch is set by double parity or addressing errors or by the combination of a parity error in one memory and an addressing error in the other. The TLC latch is reset at the beginning of each computer cycle so that multiple errors may be sensed by continuously monitoring the TLC output at the computer interface. A Test Equipment Reset (TER) signal is provided to reset the error monitors; this signal is returned to -3 VDC when the test equipment is disconnected to provide a continuous "0" input to the RECN inverter. Timing for the COC and CNC latches is keyed to the SYNC signal which starts the memory modules to cycle, thereby establishing the time reference for checking on current.

2-189. Memory Buffer Select Latches. (Figure 2-60.) From the memory buffer registers information flows in two directions, 1) through the inhibit drivers into the memory and 2) into the computer to be used. The memory select latches provide separate control over data flowing in each direction. The MAO-MBO latch selects the memory buffer to provide information to the computer and the BRAO and BRBO latches select the memory buffer to control the inhibit drivers. Separate control is maintained so that errors occurring in the non-selected memory can be corrected from the selected without changing the selection of the memory buffer providing information to the computer.

2-190. Operation of the MAO-MBO latch is straightforward. When the error monitors detect an error in one memory, the error monitor outputs drive the MAO-MBO latch to select the opposite memory. The latch then remains unchanged until an error is sensed from the newly selected memory.

2-191. The outputs of the A and B buffer registers are applied to the inhibit drivers in both memories so that errors in the A memory can be corrected by inputs from the B buffer register and vice versa. The BRAO and BRBO latches gate the inputs from the buffer registers into the inhibit drivers of the A and B memories, respectively. As soon as a module in the A memory is selected (DMA or IMA) and cycled without error (EADMN or EAIMN), the BRAO latch is set and remains set until an error is detected in the A memory. Immediately upon sensing the error the BRAO latch is reset and the contents of the B buffer register are gated into the inhibit drivers of the A memory to correct the error. Operation of the BRBO latch is identical to that for BRAO.

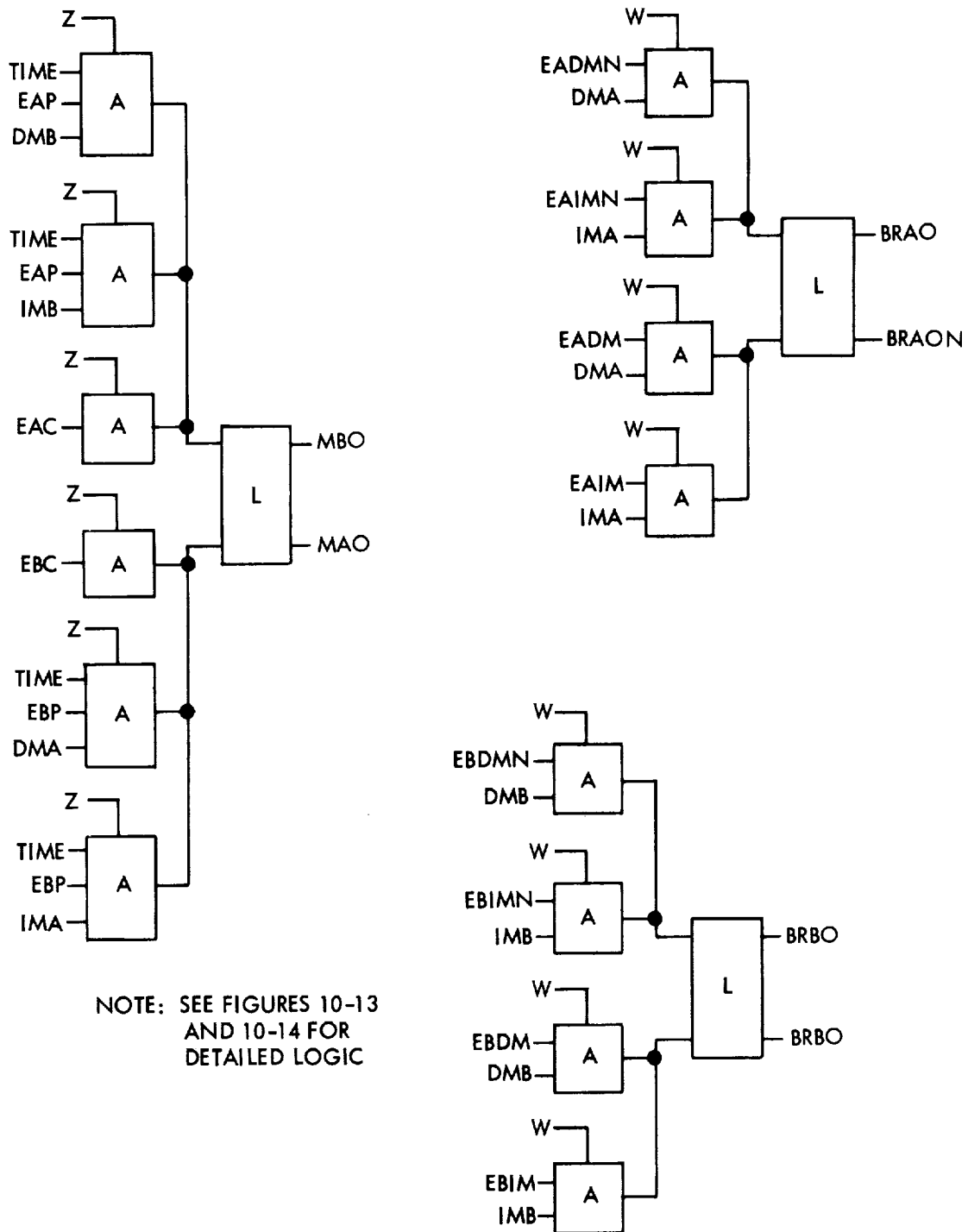


Figure 2-60. Memory Buffer Select Latches

2-192. DATA CONTROL ELEMENT.

2-193. The Data Control Element consists of the Transfer Register and the Parity Counter. The transfer register is the central storage location in the computer which provides a two-way path for transfer of data or instructions between itself and the memory or itself and other storage locations in the central computer. The parity counter assigns a parity bit to all syllables of data that are stored in the memory. When these syllables of data are subsequently read out of memory, their parity condition is checked against the parity bit.

2-194. **TRANSFER REGISTER.** The Transfer Register is a 13-bit shift register which receives from and sends inputs to the Memory Buffer Registers, the Arithmetic Element, the Multiply-Divide Element, and the external equipment.

2-195. The transfer register consists of 19 latches as shown in figure 2-61. Associated with the register are three control latches: SRTN, TBR, and CLTR.

2-196. Serial inputs are applied to the TR1 latch which are shifted bit-by-bit into the register. The serial inputs are DIN, MD2V, and AI1V. DIN is an input from the external equipment. When the external equipment is not supplying an input, DIN may be an open circuit; the input to gate A1 is held to a "0" by V3 (-3 volts). MD2 is an output of the Automatic HOP Save Circuit which is transferred through gate A2 into the transfer register for storage in the memory. The Accumulator-Register latch, AI1, provides two serial inputs: the 26-bit data word applied through gate A3, and the eight Instruction Counter bits, A1 through A8, applied through gates A4 and A5. The data word is shifted into the transfer register in order to be stored in the memory during an STO operation (provided the memory address is not 776 or 777, as evidenced by STMDN = "1"). The Instruction Counter bits are shifted into the register for eventual transfer to the Address Register.

2-197. Serial inputs are also applied to latches TR11 and TR12 during SHF operations from the AI2 latch of the accumulator-register. (Refer to Arithmetic Element for a description of the SHF operation.)

2-198. Data is transferred into the transfer register in parallel from Buffer Register A or Buffer Register B (BRA1 through BRA13 or BRB1 through BRB13) depending upon whether an odd or even memory is selected. The data is transferred into the corresponding transfer register latches TR1 through TR13 at X time, except for those latches (TR2, TR4, TR7, and TR10) which have data shifted into them at X time. Those latches are loaded at Y time to avoid a double read-in.

2-199. Shifting Data. In a serial shift register, such as the transfer register, the objective is for each latch to store a given bit for one bit time, then shift its contents to the next latch in sequence. During the next bit time, the latch stores the bit from the preceding latch, etc. The type of electronics used in this computer presents certain obstacles to such a register. Data cannot be transferred into a latch until the transfer of its contents to the next latch in the sequence is complete. Gating sequential latches in a shift register at the same time can result in a "rippling" action in which one bit runs the length of the register destroying its contents. To prevent rippling, successive

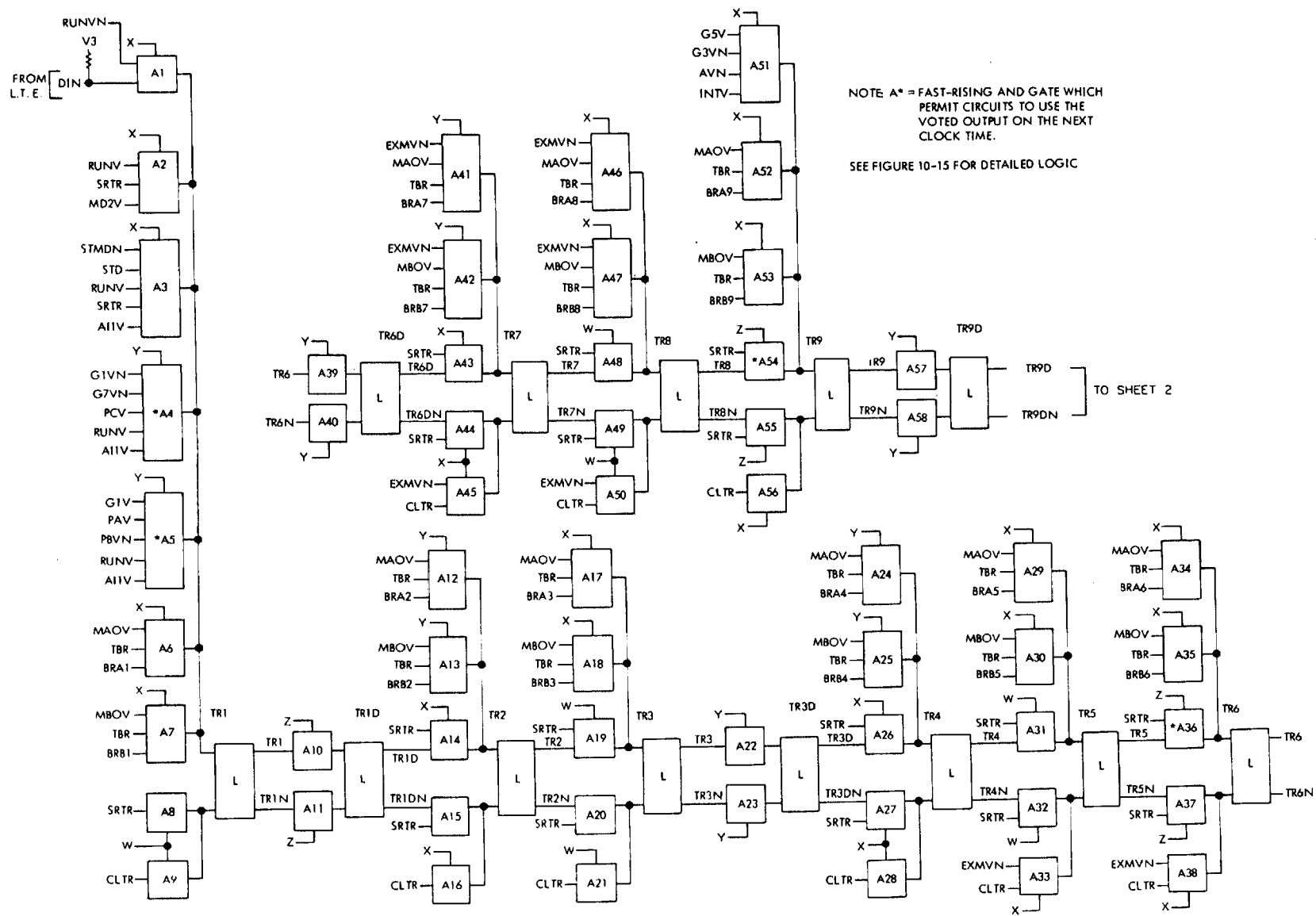
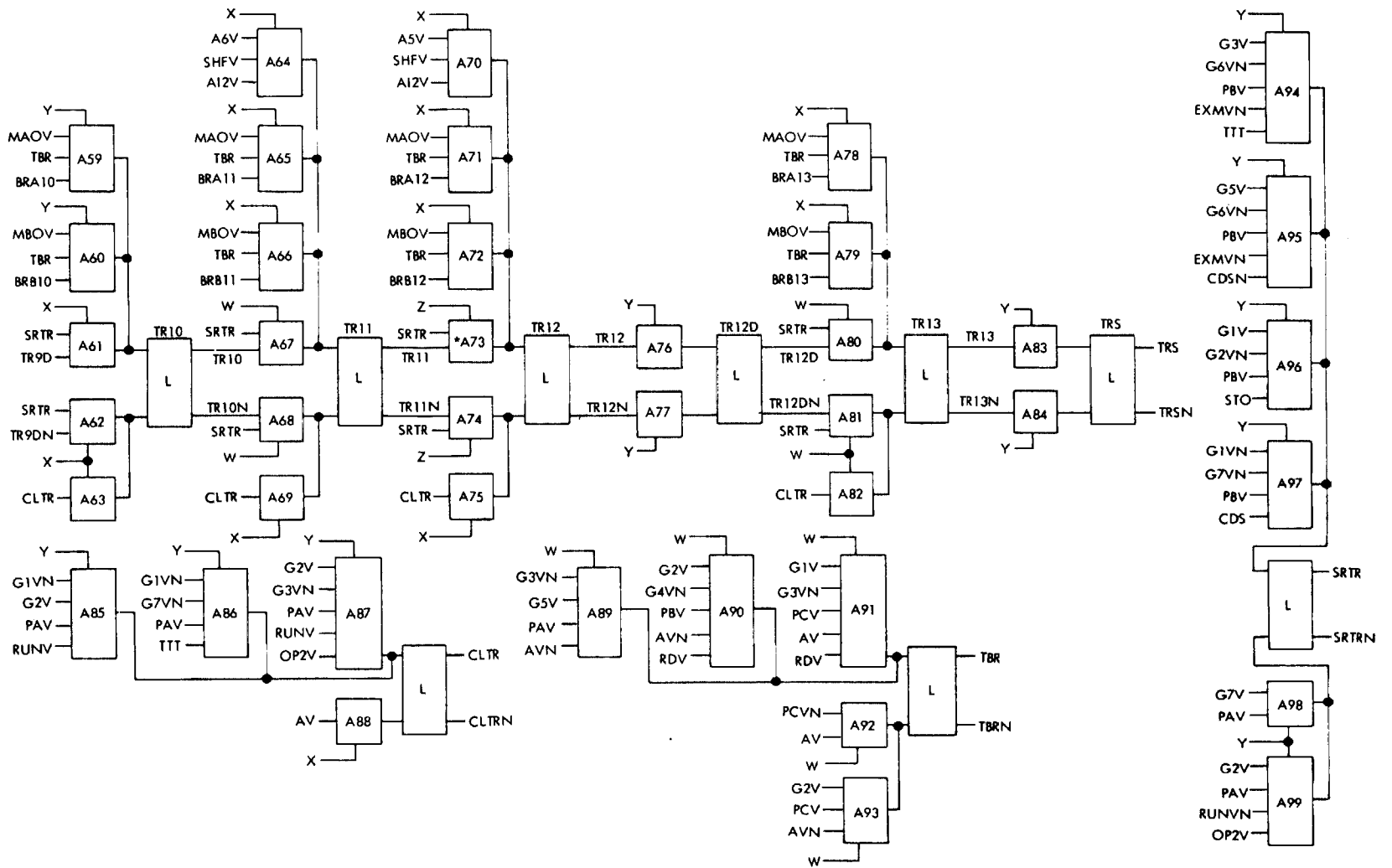


Figure 2-61. Transfer Register (Sheet 1 of 2)



NOTE: A* = FAST-RISING AND GATE
SEE FIGURE 10-6 FOR DETAILED LOGIC.

Figure 2-61. Transfer Register (Sheet 2)



latches in a shift register are gated with a reversed clock sequence. (See figure 2-62.) If latch 2 is gated at X time and latch 3 is gated at W time, and bit B1 enters latch 2 at 2-X time, then B1 will not appear in latch 3 until 3-W time.

NOTE

Only three clock times separate the gating times between latches.

Subsequent bits will set or reset the latches in a similar manner. A problem occurs at L4. Since L3 is gated at W time, L4 is gated at Z time (the clock time previous to W). However, Z time occurs during the same bit time as W time. Thus, the same bit is gated into L3 and L4 during the same bit time. Since only three clock times separate the gating of one latch from the other, it becomes apparent that the same data will eventually be available in two latches during the same bit time. The delay latches are inserted into the register to make up for the shortened gating time.

2-200. For timing convenience, the delay latches are gated at Y time. (See figure 2-61.) Latches TR3D through TR12D are gated in the order described in the previous paragraph. TR1D cannot be gated at Y time because TR1 is gated at Y time.

2-201. Transfer Register Control Latches. The control latches determine when data is shifted into, through, or out of the register (SRTR), when the register is cleared of data (CLTR), and when data is loaded into the register in parallel (TBR).

2-202. When the SRTR latch is set, data may be shifted into and through the register or data in the register may be shifted out through the TRS latch. Gate A94 permits the stored data to be shifted out of the TRS latch starting at B-3-Y time except when TNZ, TMI, TRA, or CDS operations are being performed (TTT = "0"). If a CDS operation is not being performed (CDSN = "1") and a transfer operation is being performed (TTT = "0"), gate A95 permits the stored data to be shifted out of the TRS latch starting at B-5-Y time.

NOTE

The operation of the register during an EXM operation is described later.

During a STO operation, gate A96 permits data to be shifted into the TRS latch starting at B-1-Y time. During a CDS operation (TTT and CDSN = "0"), gate A97 permits data to be shifted out of the TRS latch starting at B-14-Y time. The latch is set at A-7-Y time of the following computer cycle. During the time the computer is under control of external equipment, RUNVN = "1". When OP2 = "1", the latch is reset and the contents of the transfer register cannot be shifted out. This condition occurs during Memory Load and Verify operations.

2-203. When the TBR latch is set, gate A89 permits transfer of the instruction from the buffer register to the transfer register at A-11-W time; gate A90 permits transfer of syllable 0 of the data word and gate A91 permits transfer of syllable 1 of the data word at B-3-W and C-2-W respectively.

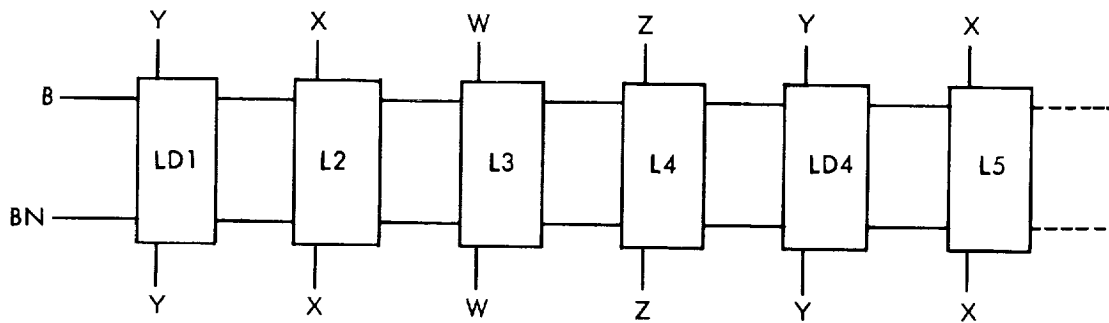
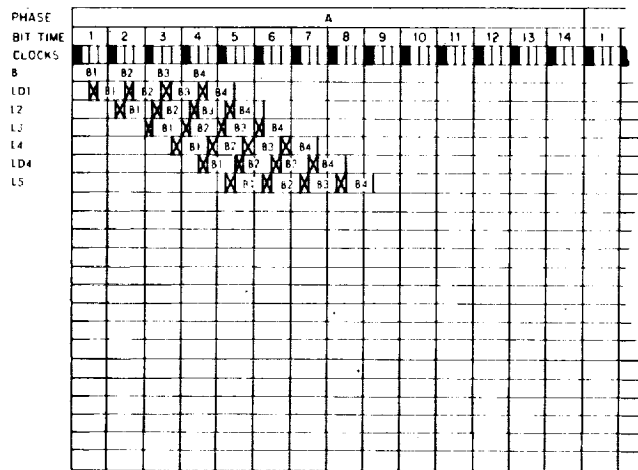


Figure 2-62. Transfer Register Simplified Block and Timing

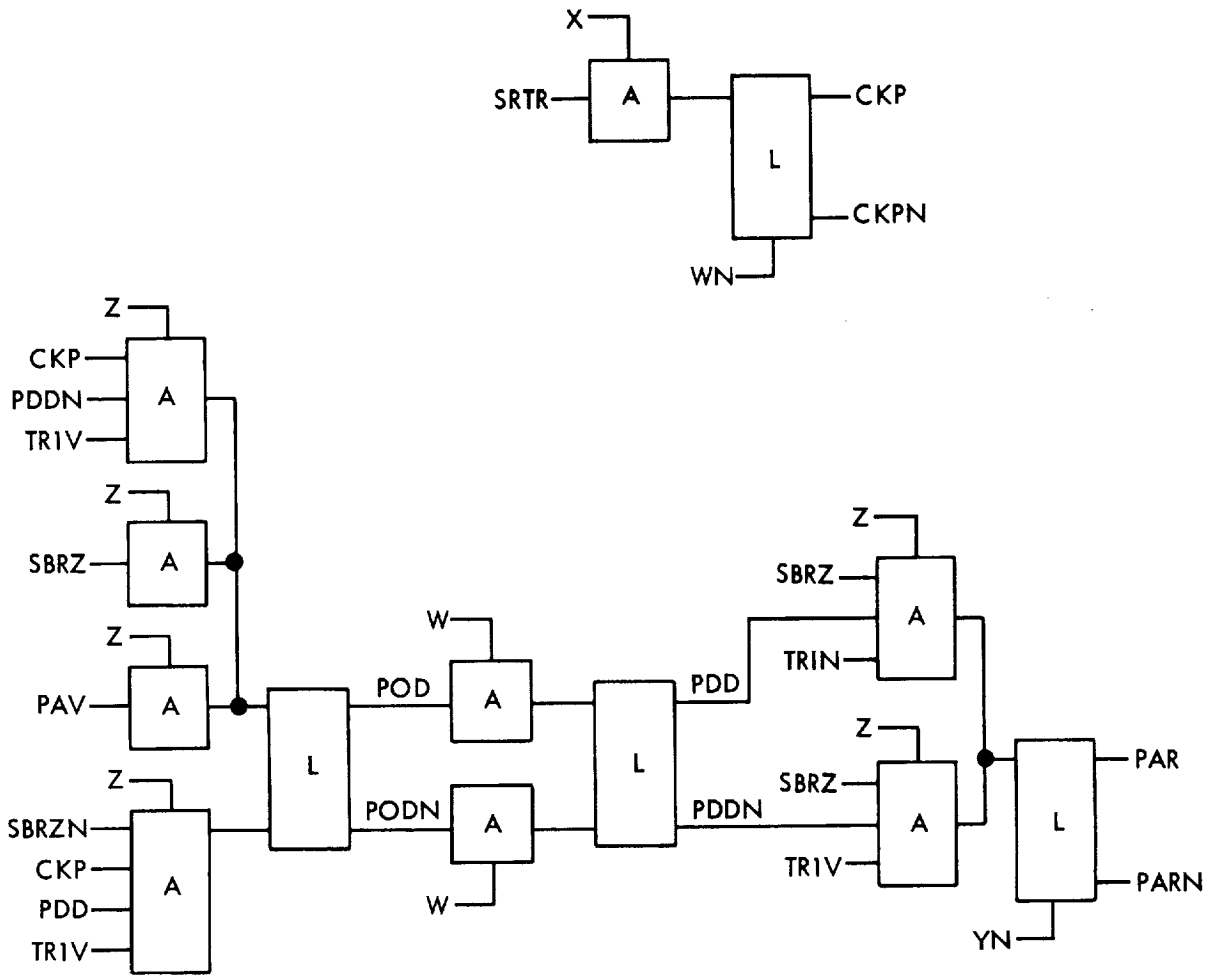
2-204. When the CLTR latch is set, gate A85 controls clearing the transfer register at A-8-Y time prior to read-in of the instruction from the buffer register; at A-14-Y time by gate A86 prior to data read-in provided a transfer or change data sector operation is not being performed (TTT = "1"); and when under control of external equipment (RUNVN = "1"), at A-2-Y time by gate A87 in order to prevent stored data from being read into the address register and an erroneous instruction being performed.

2-205. Operation During EXM. When an EXM operation is being performed, the SRTR latch remains reset and inhibits shifting data out of the transfer register. The register can be cleared, except for latches TR5, TR6, TR7, and TR8 the contents of which are retained because EXMVN = "0". The contents of latches TR5 and TR6 are ORed with the contents of buffer register latches 5 and 6 when the next instruction is transferred into the register. The contents of latches TR7 and TR8 cannot be altered when the next instruction is transferred in the register because gates A41, A42, A46, and A47 are closed when EXMVN = "0".

2-206. PARITY COUNTER. The Parity Counter assigns parity bits by making all syllables have "odd" parity. Odd parity means that the total number of "1's" in a syllable is odd. The parity counter "counts" the number of "1's" in a syllable and adds a "1", as necessary, so that the total number of "1's" is an odd number. Thus, if the total number of "1's" is even, the parity counter adds a bit; if odd, no bit is added. The parity counter operates during a Store operation when data is loaded into the memory.

2-207. The parity counter consists of three latches: POD, PDD, and PAR, and a control latch, CKP. (See figure 2-63.) At the beginning of a computer cycle, PAV initializes the POD latch to the zero (even bit) condition. The PDD latch stores this condition the following W time. During a STO operation, bits are read into the transfer register TR1 latch during phase times B and C. (See figure 2-64.) As syllable zero, bits 25 through 13, are shifted out of the TR1 latch into the TR2 latch, the bits are also read into the POD and PAR latches. The PAR latch is unaffected by bits 25 through 14 because the input gates are closed when SBRZ = "0". Since PDD = "1", the first "1" out of TR1 resets the latch indicating that the total number of bits is odd and an odd parity bit is not needed. The PDD latch is reset the following W time. Now PDDN = "1" and the next "1" sets the POD latch indicating that an odd parity bit is needed. Thus the first, third, fifth, etc. "1" resets the POD latch (and the PDD latch subsequently), while the second, fourth, sixth, etc. "1" sets the latch.

2-208. When the last bit in syllable zero is read into the TR1 latch, bit 13 at BG14 time, the entire syllable zero is stored in the transfer register. Syllable zero is now ready to be transferred to the buffer register. This transfer occurs at B-14-Z time when SBRZ = "1". The parity of the low order half of the data must be assigned prior to the transfer. At B-14-W time the PDD latch contains the parity "state" of the first 12 bits of data read into the POD latch. The TR1 latch contains bit 13 of the data word. These two pieces of information are applied to the set gates of the PAR latch. Two conditions may exist which necessitate a parity bit to be assigned to the syllable of data: if the number of "1's" in the first 12 bits is even (PDD = "1") and TR1 = "0", thus making the total even; or the number of "1's" in the first 12 bits is odd (PDD = "0") and TR1 = "1", also making the total even. When SBRZ = "1" both conditions will set the PAR latch and assign a "1" to the parity bit of the data placed in the memory. At the same time SBRZ will initialize the POD latch to the zero, or even bit, condition. Note that parity is checked anytime data is shifted into, or out of, the transfer register (SRTR = "1" sets CKP = "1"). However, only a Store operation can set SBRZ = "1".



NOTE: SEE FIGURE 10-7 FOR DETAILED LOGIC .

Figure 2-63. Parity Counter

2-209. PROGRAM CONTROL ELEMENT.

2-210. The Program Control Element steps the computer program through its instructions to execute the commanded operations. To accomplish this task, the Program Control Element performs the following functions:

- Starts or stops the program under external control.
- Conditions the memory address decoders to select programmed instructions or data words.
- Selects data and instruction memory sectors under program control.
- Selects instruction addresses within a memory sector either automatically or under program control by an instruction counter.

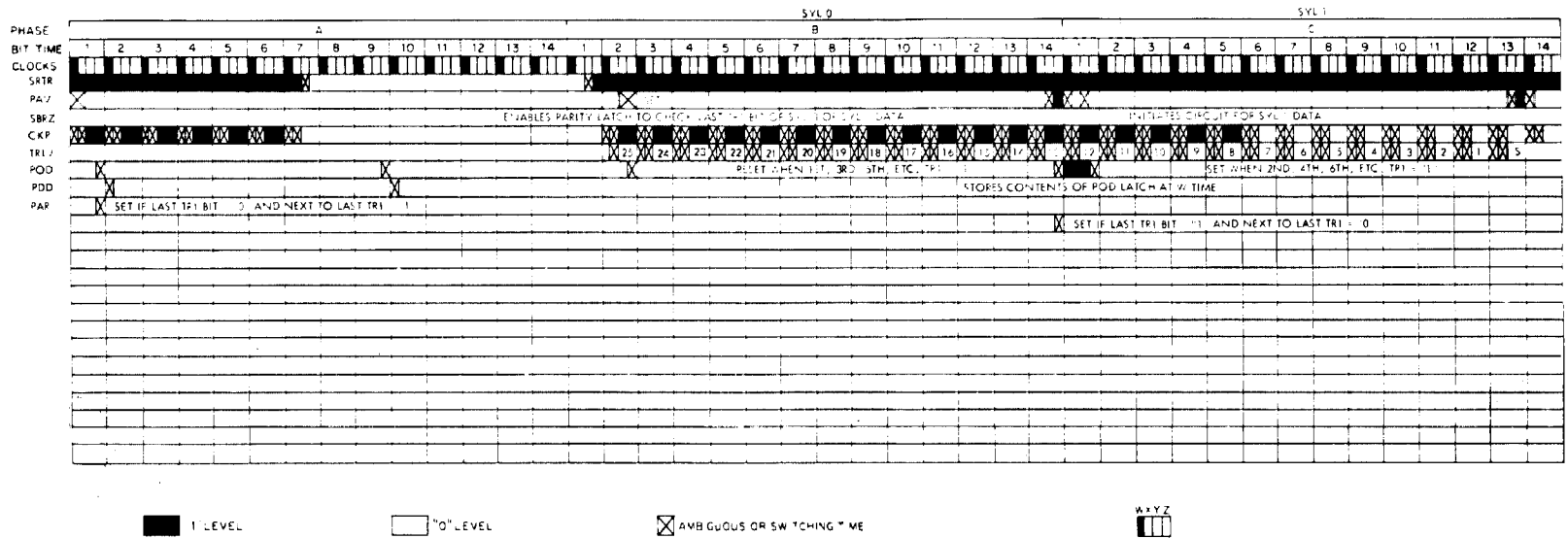


Figure 2-64. Parity Counter Timing

- Store, decode, and initiate operation codes.
- Interrupt the program, under external control, to perform a selected subroutine of the program.
- Generate a word (HOP constant) which reinitiates the computer program at the instruction where it was interrupted, after completion of an externally commanded subroutine.

2-211. The Program Control Element consists of an address register; data and instruction sector registers; operation code register and decoders; interrupt and start-stop control circuits, and automatic HOP save circuit, and instruction counter. The following paragraphs describe the operation of the circuits.

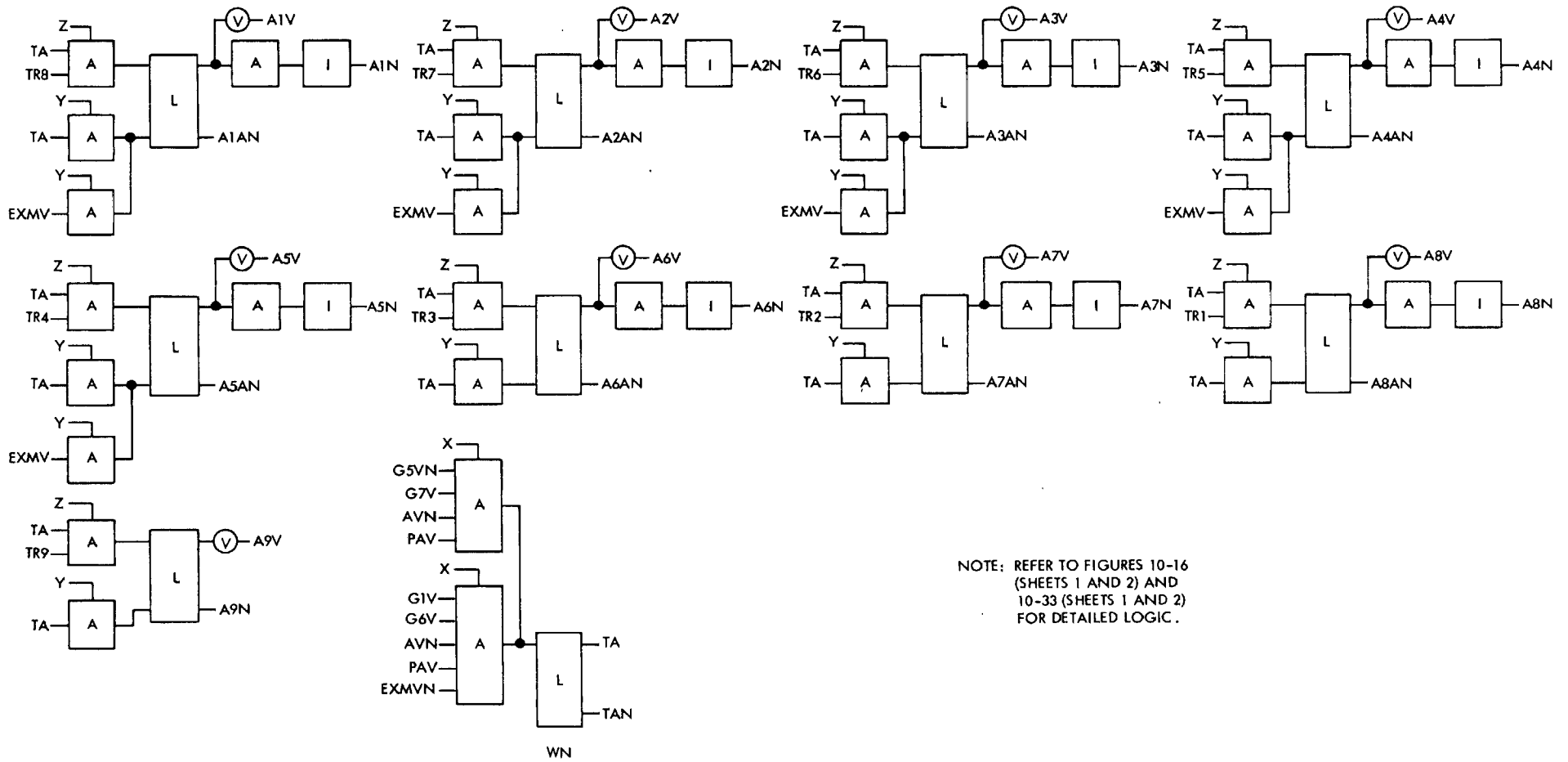
2-212. ADDRESS REGISTER. (See figure 2-65.) The Address Register stores instruction or data word addresses while the selected word is read from memory. During an instruction word time, the address register stores the instruction address (from the instruction counter) to select the instruction word address; the operand address portion of the instruction word, stored during data word time, is used to select the data word address.

2-213. The address register consists of nine latches, A1 through A9, with associated AND and INVERTER output circuits for added driving power. Latches A1 through A8 select one of the 256 memory locations in a sector; latch A9 determines whether the addressed data will come from a pre-selected sector, or from the residual memory. The addresses of selected memory locations are loaded into the address register from the transfer register. The address register outputs condition the address decoders in the Memory Control Element and provide control to the data adapter for PIO operations. The transfer address (TA) latch provides timing and control for the register.

2-214. The instruction address (from the instruction counter) is stored in the address register from A-8-W through A-13-X time. (See figure 2-66.) During this period, the addressed instruction is read from memory and placed in the transfer register. The operand address portion of the instruction is then transferred to the address register at A-13-Z time for storage while the data word is read from memory. The address register stores the data word address from A-14-W through the following A-7-X time. The address register is reset at A-7-Y and A-13-Y times, prior to storing the instruction and data addresses.

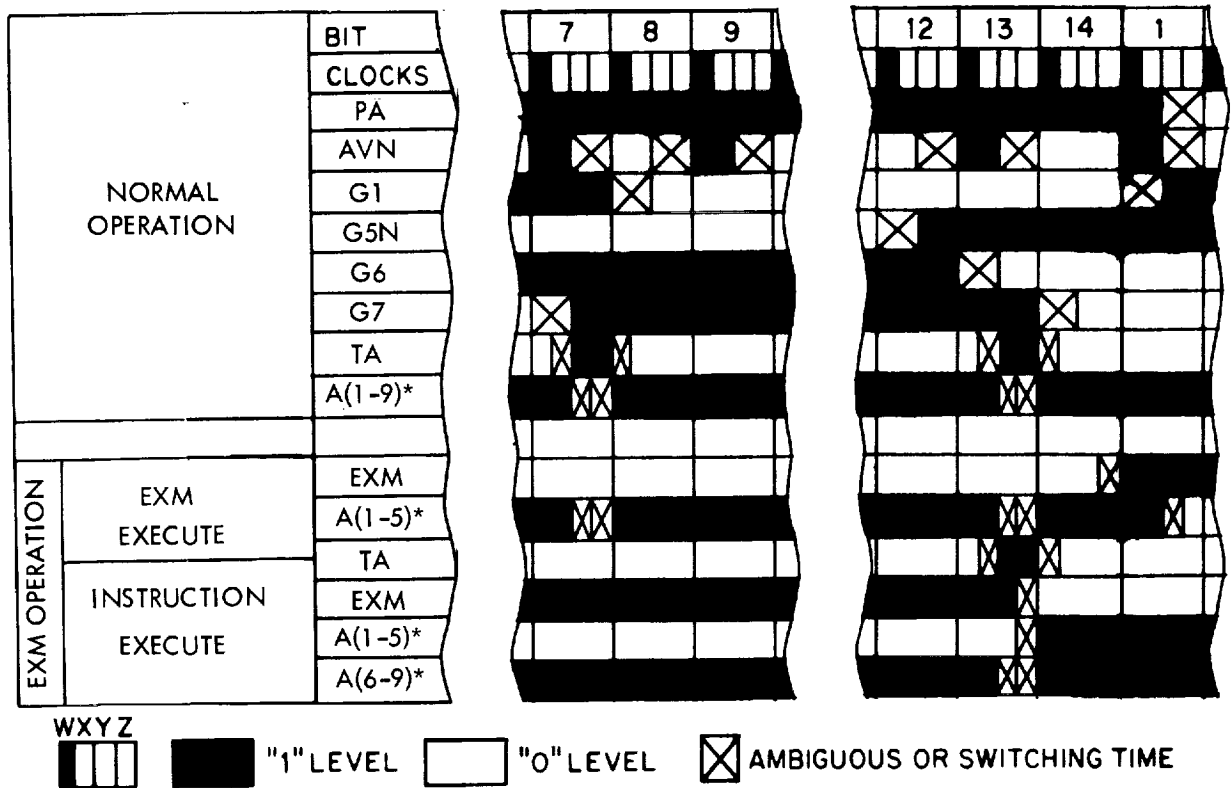
2-215. The TA latch is set during two different periods. The TA latch is set from A-7-X to A-8-W time, during this period the address register is reset at A-7-Y time and is loaded with the instruction address at the following Z clock time. The TA latch is set again from A-13-X to A-14-W time; at this time the address register is reset at A-13-Y time and is loaded with the data word address at the following Z clock time. Operation of the TA latch and address register differ slightly during an execute modified (EXM) instruction.

2-216. An EXM instruction executes and modifies the operand address of one of the instructions stored in residual memory locations 600, 640, 700, and 740. The operand address portion of the EXM instruction is transferred into the address register at A-13-Z time. The data stored in latches A1 through A5 remains in the transfer register to select the syllable of the instruction to be executed and modify its address, but this data must be cleared from latches A1 through A5 to select the desired residual memory locations.



NOTE: REFER TO FIGURES 10-16
(SHEETS 1 AND 2) AND
10-33 (SHEETS 1 AND 2)
FOR DETAILED LOGIC.

Figure 2-65. Address Register and Transfer
Address Latch

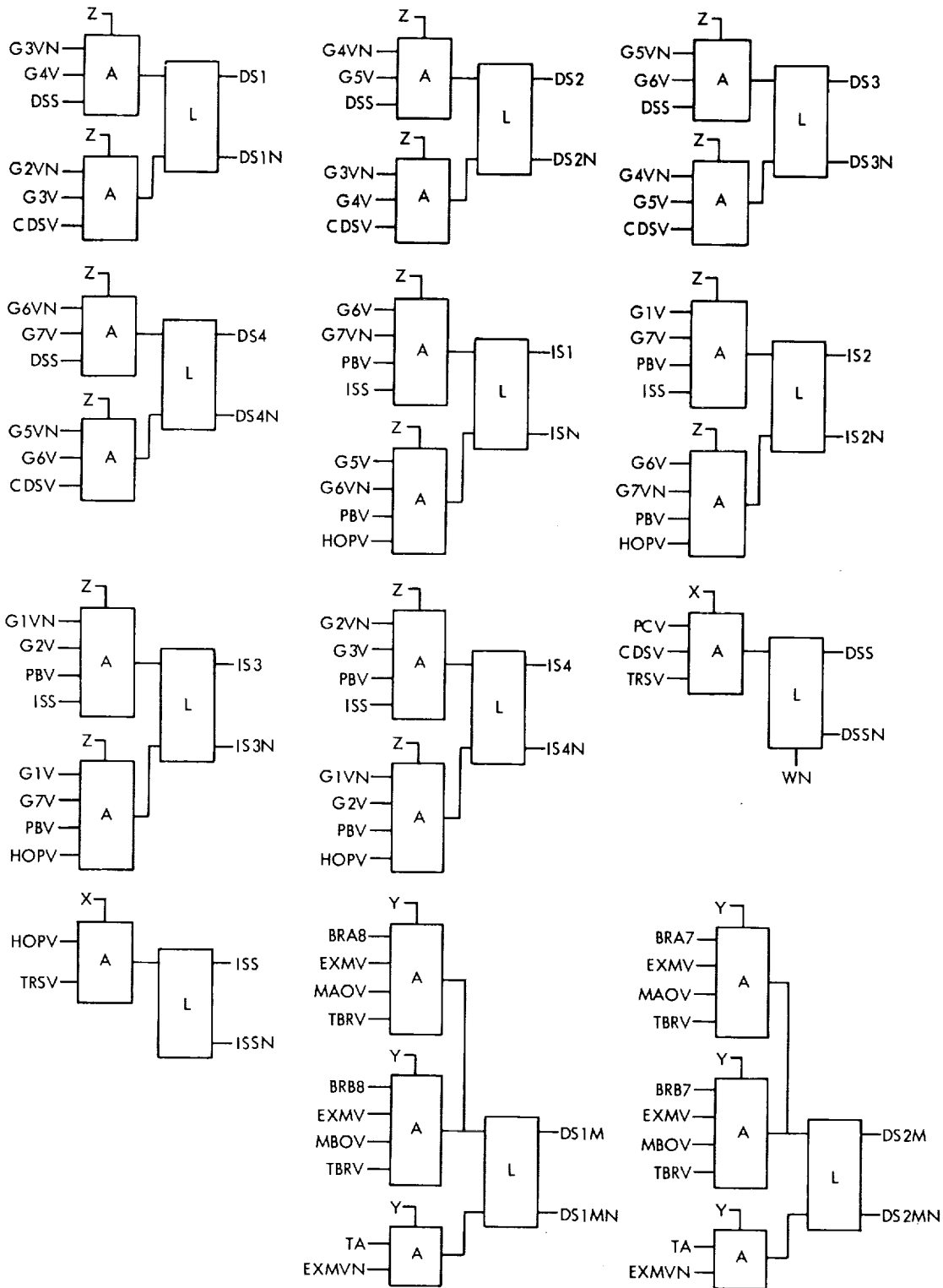


* SET WHEN TR (1-9) EQUALS "1" AT A-7-Z OR A-13-Z

Figure 2-66. Address Register Timing

2-217. Address register latches A1 through A5 are reset at B-1-Y time to select address 600, 640, 700 or 740. Since the address register already contains the address of the next instruction at A-7-Z time, TA equals "0" to prevent the instruction address from the instruction counter from being loaded into the address register; the EXMN level equals "0" preventing the TA latch from being set. The address portion of the instruction to be executed, already modified in the transfer register, is loaded into the address register at A-13-Z time. The address modifying bits A-1 through A-4 of the EXM instruction had remained unchanged in the transfer register during phase B and C times of the EXM instruction because memory operation was halted. After the address portion of the instruction to be executed is loaded into the address register the computer resumes normal operation.

2-218. DATA AND INSTRUCTION SECTOR REGISTERS. (See figure 2-67.) The Data and Instruction Sector Registers store sector codes to select one of 16 memory sectors during operation and instruction times, respectively. Sector codes are loaded into both registers during a HOP instruction (HOP sets CDSV to load the data sector register); the data sector register is also loaded during a change data sector (CDS) instruction. The registers store the sector codes until the next HOP (or CDS) instruction commands them to change. The sector register outputs condition the memory address decoders.



NOTE: REFER TO FIGURE 10-17 FOR DETAILED LOGIC.

Figure 2-67. Sector Registers and Control Latches

2-219. The data sector (DS) and instruction sector (IS) registers each contain four latches. The DS register is loaded by the data sector serialized (DSS) latch; likewise, the IS register is loaded by the instruction sector serialized (ISS) latch. The data sector selection is altered for one computer cycle following an execute modify (EXM) command. During the execution of an instruction following an EXM command, the two data sector modified latches (DS1M and DS2M) substitute for the DS1 and DS2 latches to condition the memory address decoders. The HOP constant appears as a 26-bit serial output of the transfer register serializer latch (TRS) during phase times B and C. The data sector code occupies bits 2 through 5 and the instruction sector code bits 20 through 23 of the HOP constant. The DS register can also be loaded from operand address bits A5 through A8 of a CDS instruction. The transfer register produces the data sector code at TRS during the same bit times for both HOP and CDS.

2-220. The DSS latch is gated to duplicate the TRS output during phase C. The data sector code emerges from the DSS latch at bit times C-10 through C-13. (See figure 2-68.) The set gates of latches DS1 through DS4 bear the timing signals which sense DSS at the proper times to gate each bit of the data sector code into the appropriate latch. Each latch is reset one bit time before sensing DSS, clearing the previous code before the new code is loaded.

2-221. The IS register is loaded in the same manner as the DS register with the following exceptions. The IS register is loaded only during a HOP command and therefore, its reset gates are enabled by HOP rather than CDS. The instruction sector code occupies a different position in the HOP constant than the data sector code and consequently requires different timing signals to accept it. The IS register is loaded at bit times B-6-Z through B-9-Z.

2-222. The data sector modified (DS1M & DS2M) latches are substituted for the DS1 and DS2 latches during the operation time of the instruction following an EXM command. From A-11-Y to the following A-6-Y time, the DSM latches store operand bits A1 and A2 of the instruction to be executed; the MAOV or MBOV, TBRV, and EXMV signals permit loading the DSM latches at A-11-Y time. The A1 and A2 bits are loaded into the DSM latches from the selected memory buffer.

2-223. During the execution of the operation following an EXM instruction, the data sector code substituted for the DS register outputs is as follows: The DSM latch outputs substitute for DS1 and DS2 latch outputs; the EXMD latch output substitutes for the DS3 latch output; and the A9 bit of the instruction to be executed substitutes for the DS4 latch output. See figure 2-6 for a brief account of EXM operation.

2-224. OPERATION CODE REGISTER. (See figure 2-69.) The four bit Operation (OP) Code Register stores the code of the commanded instruction while the instruction is executed. The OP code register outputs control the operation of the circuits which execute the instructions.

2-225. The OP code register, loaded by the transfer register, stores the operation code from A-12-Y to A-5-Y time, nearly a full computer cycle. See figure 2-70 for the OP code register load and reset conditions. Operation of the register differs slightly during a multiply and hold (MPH) operation and during computer single step operation.

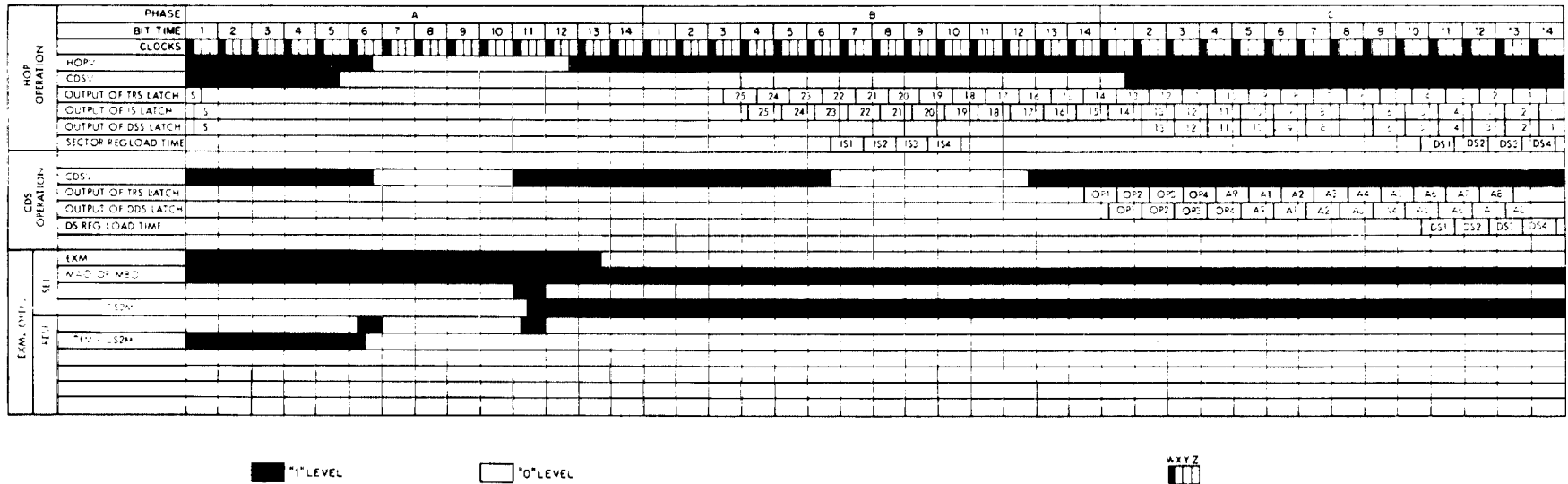
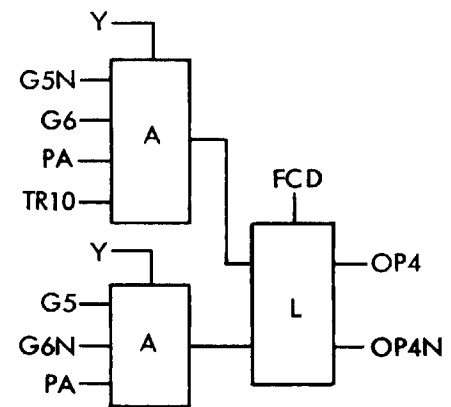
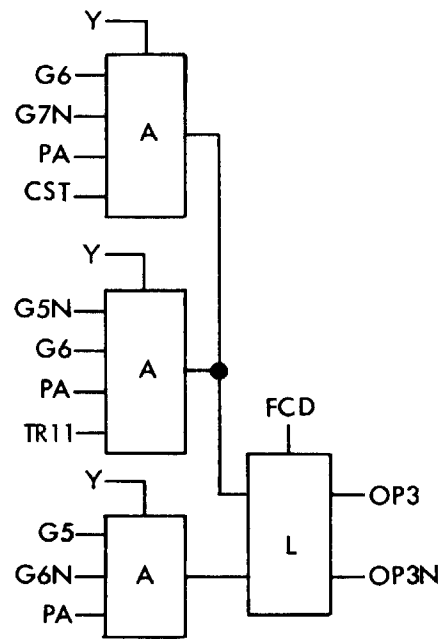
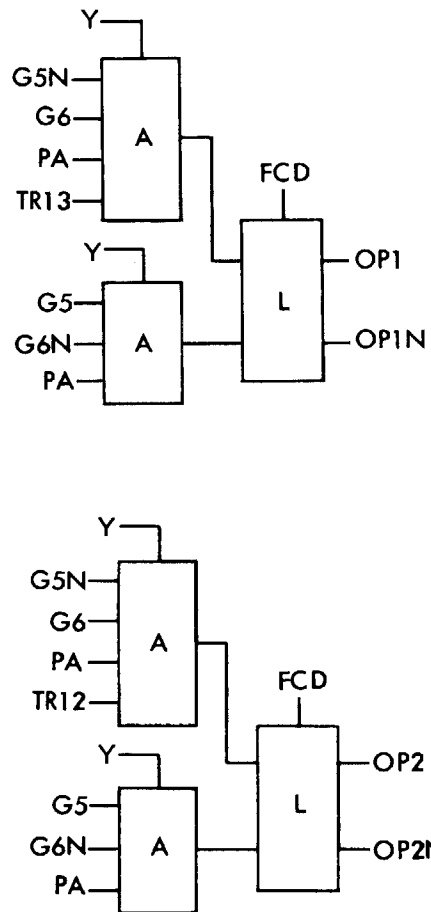


Figure 2-68. Data and Instruction Sector Register Timing



NOTE: REFER TO FIGURE 10-8 (SHEET 1) FOR DETAILED LOGIC.

Figure 2-69. Operation Code Register

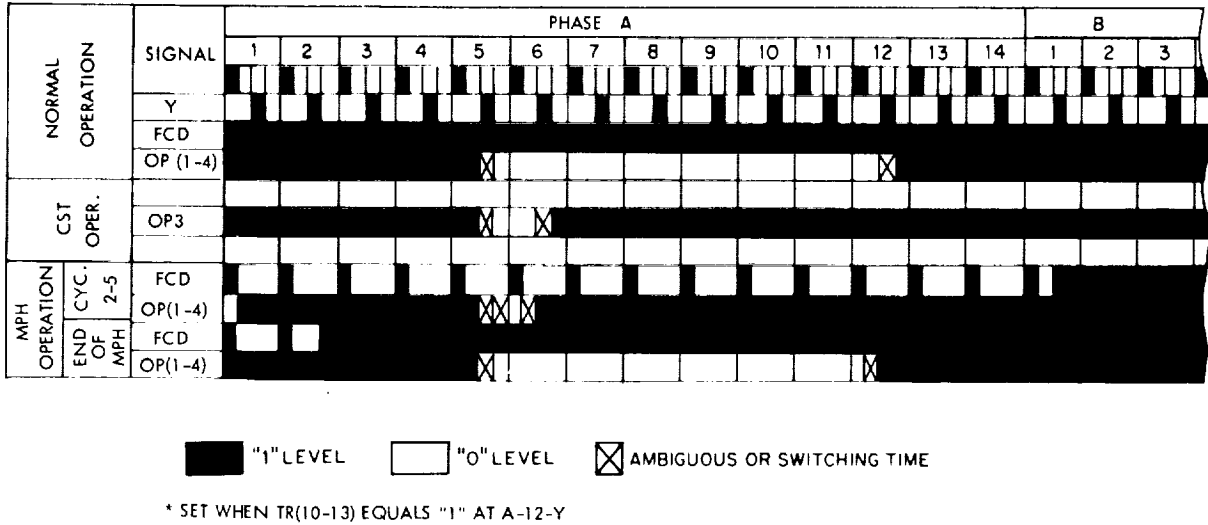


Figure 2-70. Operation Code Register Timing

2-226. At the completion of an MPH operation, the force clear and add (FCD) signal enables transfer of the contents of the PQ register into the accumulator. The FCD signal forces a clear and add (CLA) operation code into the OP code register and selects the PQ register as the data address. The MPH operation code is loaded into the OP code register, as per normal operation, during the first computer cycle (MPH requires five computer cycles). During the second computer cycle of an MPH operation, FCD equals "0" at A-1-Y time to set the OP code register to all "1's" (CLA). The product of the MPH operation, located in the PQ register, is not transferred into the accumulator until the fifth computer cycle. The CLA code is cleared from the OP code register at A-5-Y time of the next instruction, six computer cycles after initiation of the MPH operation.

2-227. Computer single step (CST) operation is used to step through selected instructions, one at a time, on command. The externally controlled CST signal, enabled during phase C time of the selected instruction, forces a transfer non-zero (TNZ) operation code into the OP code register. The TNZ operation code disables memory because it requires no operand. The OP code register is reset at A-5-Y time, as per normal operation; then the OP3 latch is set at A-6-Y time (PA, G6, G7N, CST equal "1") forcing the TNZ operation code (0100) into the OP code register. The TNZ code is loaded at A-6-Y time to disable the memory prior to instruction read time, thus preventing a new operation code from being loaded into the OP code register at A-12-Y time. The computer then idles in this condition until the externally controlled CST gate is momentarily disabled allowing the next operation code to be loaded into the OP code register at normal load time.

2-228. OPERATION DECODERS. (See figure 2-71.) The Operation Decoders supplement the OP code register in controlling the operation of circuits which execute the commanded operations. The operation decoders decode programmed operation codes and computer signal combinations to generate control signals. The programmed operation codes which condition the decoders are supplied via the transfer register, address register, or OP code register. The operation decoders are listed below in alphabetical order:

- Change Data Sector Latch
- Execute Modified and Execute Modified Delayed Latches
- HOP Latch
- Process Input Output Gate
- Shift And PIO Gate
- Shift Gate
- SS Gate
- SSF Gate
- Store And Store Not Gates
- TTL Latch
- TTT Gate

2-229. Change Data Sector (CDS) Latch. During CDS and HOP operations, the CDS latch permits the data sector register to store the data sector code. The CDS latch also initiates shifting the contents of the transfer register out to the TRS latch during a HOP operation.

2-230. During a CDS operation, the CDS latch is set from A-14-W through B-5-Y time and from B-13-W through A-5-Y time. The CDS latch is set by the CDS operation code stored in the transfer register. The shift right transfer register not (SRTRN) signal equals "0" during transfer register shift operation to prevent the CDS latch from being set by a false CDS code.

2-231. During a HOP operation, HOPV sets the CDS latch at C-1-Z time; the CDS latch remains set from C-2-W through A-5-Y time.

2-232. Execute Modified (EXM) and Execute Modified Delayed (EXMD) Latches. The EXM latch is set from B-14-W through A-13-Y during an EXM operation. The EXM latch is set by the EXM operation code via the OP code and address registers. The EXM latch enables the EXM operand address bit A5 to select the syllable of the instruction to be executed and clears address register latches A1 through A5 to select one of four residual memory locations. During phase A time of the instruction following an EXM command, the EXM latch enables loading of the data sector modified latches.

NOTE: REFER TO FIGURES 10-6, 10-7, 10-8 (SHEETS 1 AND 2),
 10-15, 10-18, 10-20 (SHEET 1), 10-21 (SHEET 2), AND
 10-35 (SHEET 3) FOR DETAILED LOGIC.

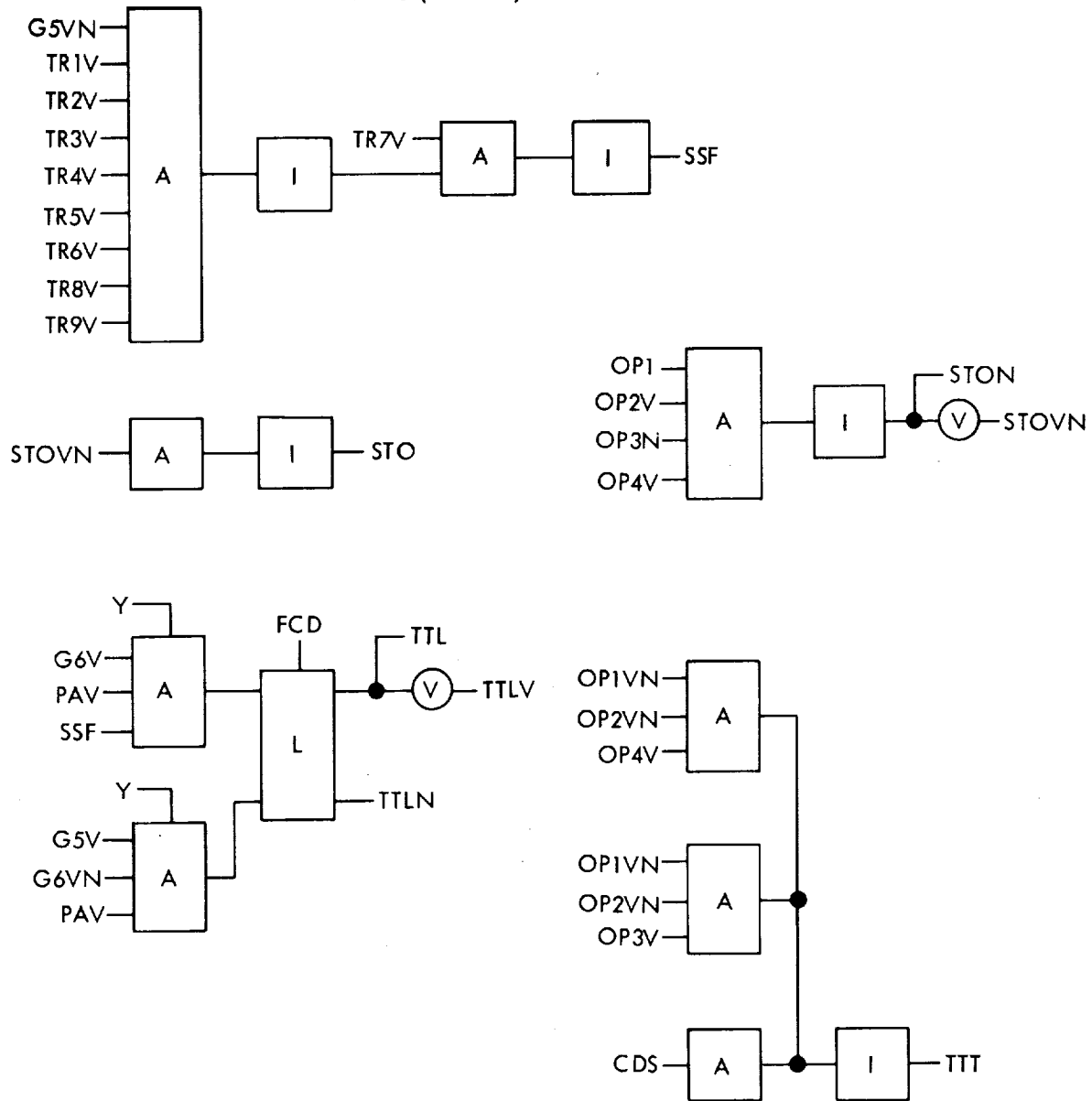


Figure 2-71. Operation Decoders (Sheet 1 of 2)

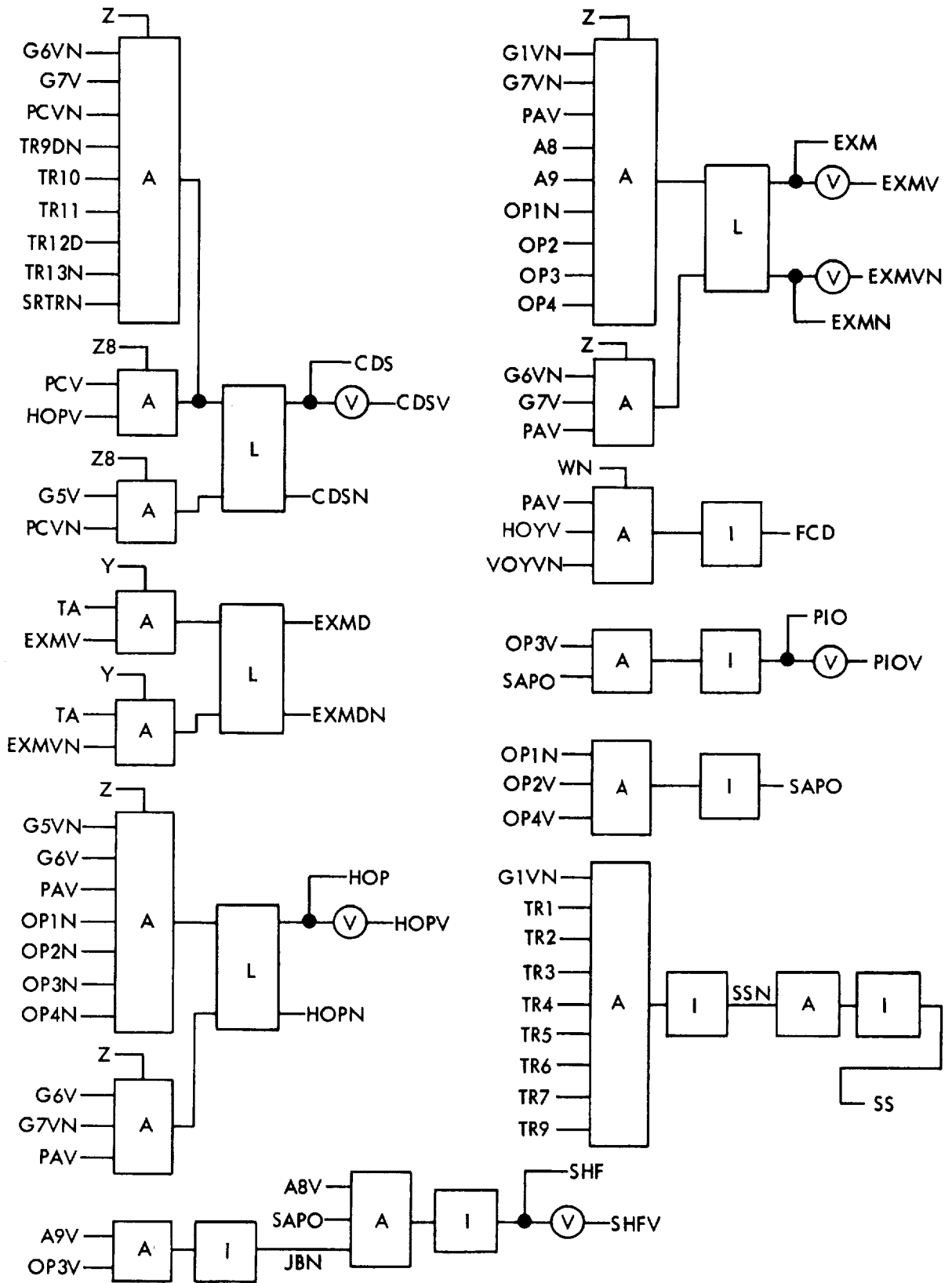


Figure 2-71. Operation Decoders (Sheet 2)

2-233. The EXMD latch is set by the EXM and transfer address (TA) signals at the end of an EXM operation. The EXMD latch is set from A-13-Z through A-7-X time. The EXMD latch conditions the memory address decoders to modify data sector selection during execution time of the instruction following an EXM command.

2-234. Force Clear and Add (FCD) Gate. FCD forces a CLA operation code into the OP code register during phase A of the second through fifth computer cycles of a multiply and hold (MPH) operation. (Signals HOYV and VOYVN equal "1" during an MPH operation.) FCD also initiates transfer of the contents of the P-Q register to the accumulator by setting the TTL latch.

2-235. HOP Latch. The HOP latch, set from A-13-W through A-6-Y time during an HOP operation, initiates loading of the instruction and data sector codes into their respective registers, disables memory, and conditions module current error latches in the Memory Control Element. The HOP latch is set by the HOP operation code stored in the OP code register.

2-236. Process Input Output (PIO) Gate. PIO equals "1" during the period that a PIO operation code is stored in the OP code register. (Input signals OP3V and SAPO equal "0".) The PIO signal conditions the accumulator to accept external data when commanded by the PIO instruction.

2-237. Shift And PIO (SAPO) Gate. SAPO equals "0" when CDS, EXM, PIO, and SHF operation codes are stored in the OP code register. The SAPO signal conditions the PIO and SHF decoders.

2-238. Shift (SHF) Gate. SHF equals "0" when a SHF operation code is stored in the OP code and address registers. The SHF signal is applied to the Arithmetic Element and transfer register to control the shift operation.

2-239. SS Gate. SS equals "1" from A-8-Y through B-1-W time when the transfer register stores an operand address of 776 or 777. Normally, during an interrupt operation a programmed STO 776 or 777 instruction will be given to store the automatic HOP constant in one of these locations. SS initiates transfer of the automatic HOP constant to the transfer register for storage in memory.

2-240. SSF Gate. When the transfer register contains operand address 775, SSF equals "1" from A-12-Y through B-1-W time. The operand address is cleared from the transfer register at B-1-X time during non-transfer operations. SSF initiates transfer of the P-Q register to the accumulator or vice versa.

2-241. Store (STO) And Store Not (STON) Gates. The STO and STON gates are conditioned when a store operation code is stored in the OP code register. The STO gate inverts the STOVN signal to generate a STO signal. The STO and STON signals initiate transfer of the accumulator into the transfer register; then initiates parallel transfer of the contents of the transfer register into the buffer register for storage in the memory address specified by the operand address of the store instruction.

2-242. TTL Latch. The TTL latch is set by the SSF signal to initiate transfer of the P-Q register to the accumulator or vice versa. The TTL latch is set from A-12-Z through A-5-X time.

2-243. TTT Gate. TTT inhibits clearing and shifting of the transfer register during transfer (TNZ, TMI, and TRA), CDS and HOP operations. The TTT signal equals "0" while transfer operation codes are stored in the OP code register or while the CDS latch is set.

2-244. INTERRUPT CONTROL. (See figure 2-72.) The Interrupt Control, on external command, stores the interrupt command (INTCV) until completion of the current instruction under execution, then halts the computer program and initiates a HOP to memory location 400.

2-245. The interrupt control consists of three latches; the INTA, INTB, and INT. The INTA latch delays initiation of the interrupt command if the computer is executing instructions requiring multiple computer cycles. These instructions are multiply (MPY), multiply and hold (MPH), divide (DIV), execute modified (EXM), and HOP. The INTB latch insures initiation of only one interrupt operation per interrupt command. The INT latch halts the computer program and initiates the HOP to 400.

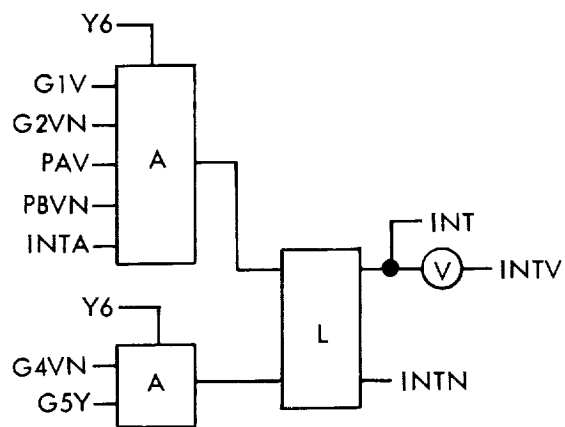
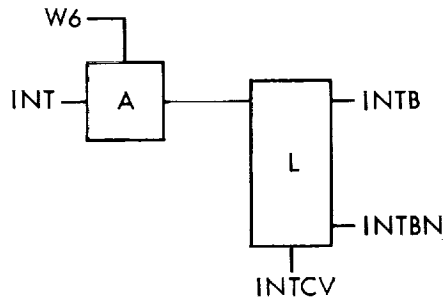
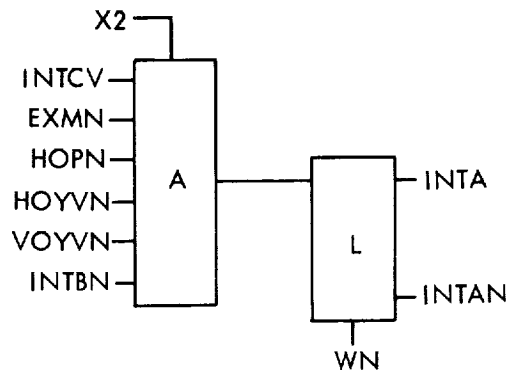
2-246. At the first X clock time following the interrupt command (INTCV = "1") the INTA latch is set providing HOYVN, VOYVN, HOPN, EXMN, and INTBN equal "1". The HOYVN and VOYVN signals equal "1" during non-execution periods of MPY, MPH, or DIV operations. The condition when INTBN equals "1" indicates that the current interrupt has not yet been initiated. Once set, the INTA latch resets at the following W clock time.

2-247. The INTA signal sets the INT latch at A-1-Y time; INT resets at the following A-11-Y time. Excluding the exceptions already noted, instructions require approximately one computer cycle to execute; consequently the INT latch can be set at A-1-Y time without interrupting the execution of an instruction. During its set period, the INT latch initiates a HOP to 400. The INT signal also sets the INTB latch to prevent initiating additional interrupts until completion of the interrupt operation. At the completion of an interrupt operation the INTCV signal is reset. When INTCV resets, the INTB latch also resets.

2-248. START-STOP CONTROL. (See figure 2-73.) The computer program start-stop operation is controlled by two externally provided signals; HALTV and CSTN. The HALTV signal conditions the Run Latch to control the computer program start-stop operation during normal operation. The CST signal conditions the CST gate to control computer program start-stop operation during computer single step operation.

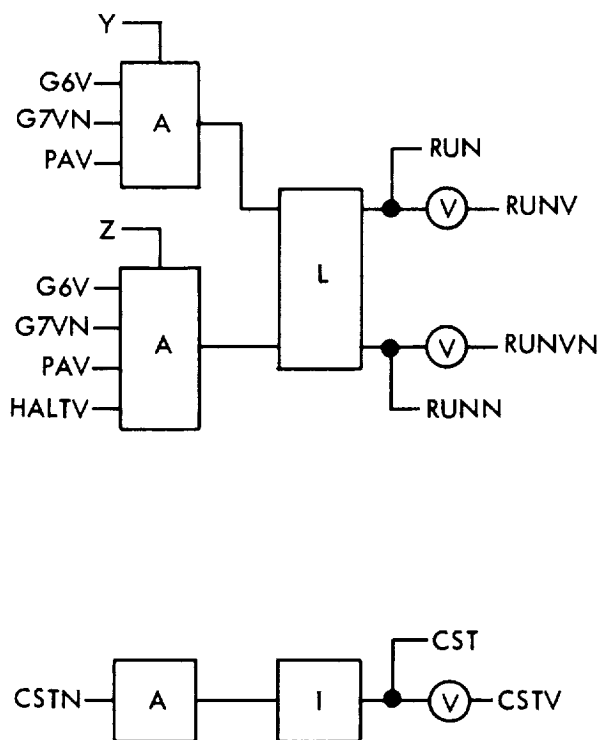
2-249. Run Latch. The Run Latch is set each A-6-Y time and remains set until the computer is externally commanded to halt; then HALTV equals "1" resetting the run latch at A-6-Z time. During the periods that the run latch is set the computer is permitted to operate as programmed. The computer program is initiated when the run latch becomes set by a simulated HOP to 000 instruction since the syllable latch, module, sector, OP code, and address registers contain "0's". A HOP constant will be stored in memory location 000 to start the computer program.

2-250. When the run latch is reset the computer program is halted in the following manner. The run latch disables memory during phase A, consequently instruction words cannot be read from memory. The transfer register and in turn the OP code and address registers will be loaded with "0's"; thus commanding a HOP to 000. When the HOP code is loaded into the OP code register the HOP latch becomes set. When both HOP and RUNVN equal "1" memory operation is also disabled during phases B and C. The reset run latch prevents the transfer of the instruction address into the transfer register; consequently the HOP to 000 command is continuously recycled.



NOTE: REFER TO FIGURE 10-8 (SHEET 1) AND 10-21 (SHEET 2) FOR DETAILED LOGIC

Figure 2-72. Interrupt Control



NOTE: REFER TO FIGURES 10-13 (SHEET 1) AND 10-30 (SHEET 2) FOR DETAILED LOGIC .

Figure 2-73. Start-Stop Control

2-251. In addition to halting the computer program the reset run latch initializes the multiply or divide phase counter to the quiescent state and prevents the accumulator and automatic HOP constant from being transferred into the transfer register. During the period that RUNVN and HOP equal "1" the module current error latches are reset for initialization. The RUNVN signal also enables the transfer register to be loaded from external sources via the DIN line. The DIN line is used during a memory load or verify operation.

2-252. During memory loading or verification, CLA and STO commands are loaded into the transfer register during instruction time via the DIN line. During execution of these instructions memory operation is enabled during phases B and C but remains disabled during phase A. During data word time of STO commands, data is loaded into the transfer register via the DIN line for storage in memory. The RUNVN signal clears the transfer register at the completion of these commanded operations.

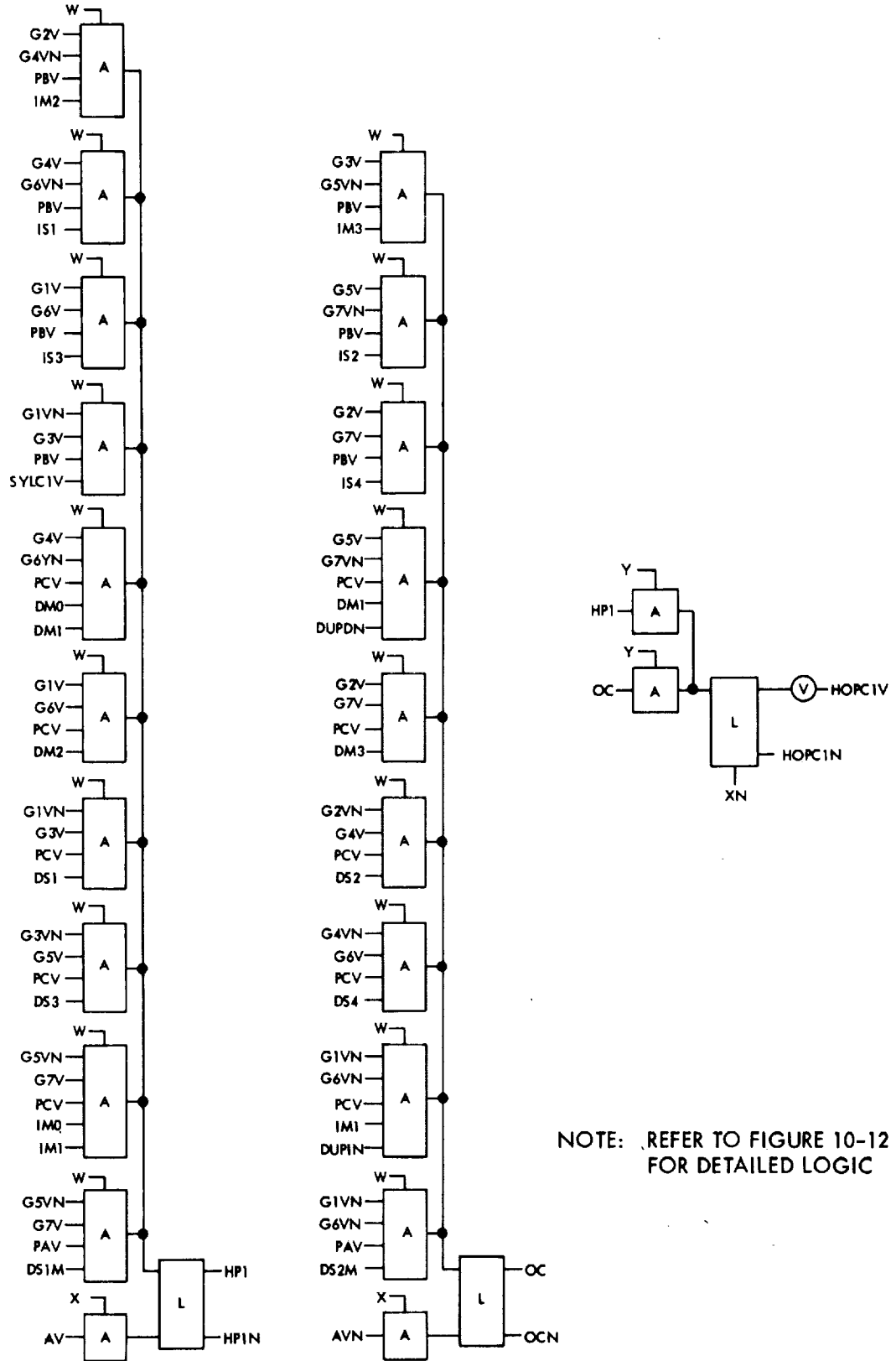
2-253. CST Gate. The CST gate inverts the externally provided CSTN signal to halt the computer program during single step operation. The CST signal forces a transfer non zero (TNZ) operation code into the OP code register, prevents the instruction counter from stepping, and prevents the operand address from transferring from the transfer register into the instruction counter if the accumulator contains a non-zero number. Since the instruction counter is unable to increase and transfer is inhibited, the TNZ operation is continuously repeated halting the computer program. When the CST gate is disabled the next programmed instruction is performed.

2-254. AUTOMATIC HOP SAVE CIRCUIT. The Automatic HOP Save Circuit assembles a HOP constant; stores the HOP constant for approximately one computer cycle (except during multiply or divide operations); and when commanded by the computer program enables the transfer register to accept the HOP constant for storage in memory. The automatic HOP constant will not be stored in memory unless a store instruction is given with an address of 776 or 777. The automatic HOP save circuit consists of a HOP constant serializer, delay circuits (located in the Multiply - Divide Element), and a control circuit (STMD latch).

2-255. The HOP constant serializer assembles a partial HOP constant every computer cycle. The instruction address (from the instruction counter) is merged into the HOP constant during a transfer operation. The HOP constant is shifted through the delay circuits (multiplicand-divisor register) for approximately one computer cycle during operations other than multiply or divide. The STMD latch, under program control, enables the HOP constant to be shifted into the transfer register for storage in memory.

2-256. HOP Constant Serializer. (See figure 2-74.) The HOP constant serializer assembles the portion of the HOP constant which contains the syllable code (SYLCIV), the instruction and data module codes (IM and DM), the instruction and data sector codes (IS and DS), and the simplex-duplex codes. The HOP constant serializer also assembles the data sector modified codes (DS1M and DS2M) to enable monitoring DS1M and DS2M operation during testing. The DS1M and DS2M codes are circulated through the delay circuits but are miss-timed for storage in memory. The HOP constant serializer output is loaded into channel one of DL31.

2-257. The HOP constant serializer consists of three latches, OC, HP1, and HOPC1. The OC and HP1 latches serialize data during even and odd bit times, respectively. The HOPC1 latch merges the OC and HP1 outputs.



NOTE: REFER TO FIGURE 10-12 FOR DETAILED LOGIC

Figure 2-74. HOP Constant Serializer

Changed 4 January 1965

2-258. The OC latch can be set during selected even bit times; the selected bit times are A-14, B-4, B-6, B-8, C-6, C-8, C-10, C-12 and C-14. (See figure 2-75.) The HP1 latch can be set during selected odd bit times; the selected bit times are A-13, B-3, B-5, B-7, B-9, C-5, C-7, C-9, C-11 and C-13. The OC and HP1 latches reset during odd-X and even-X times, respectively. The HOPC1 latch is alternately loaded with the odd and even data outputs of the OC and HP1 latches. The HOPC1 latch resets at X clock times prior to being loaded with the next bit. The HOPC1 latch serialized output is loaded into channel one of DL31.

2-259. Delay Circuits. The Delay Circuits circulate the HOP constant for approximately one computer cycle during non-multiply-divide operations. The delay circuits are loaded serially from both the HOP constant serializer and the instruction counter. The HOP serializer data is loaded unconditionally every computer cycle but the instruction portion of the HOP constant is only loaded during three operations; transfer, HOP, or interrupt. During an interrupt operation a HOP instruction is forced to enable the instruction address to be loaded into the delay circuits. The HOPC1V gate loads the HOP constant serializer into the delay circuit, whereas the instruction address is loaded into the delay circuit from the ACC0 latch.

2-260. The delay circuits consist of channel one of two delay lines (DL31 and DL44), four latches, and associated AND gates. (See figure 2-76.) The four latches are: NU, MD0, MD1, and MD2. The delay circuits are serially connected as shown in figure 2-76.

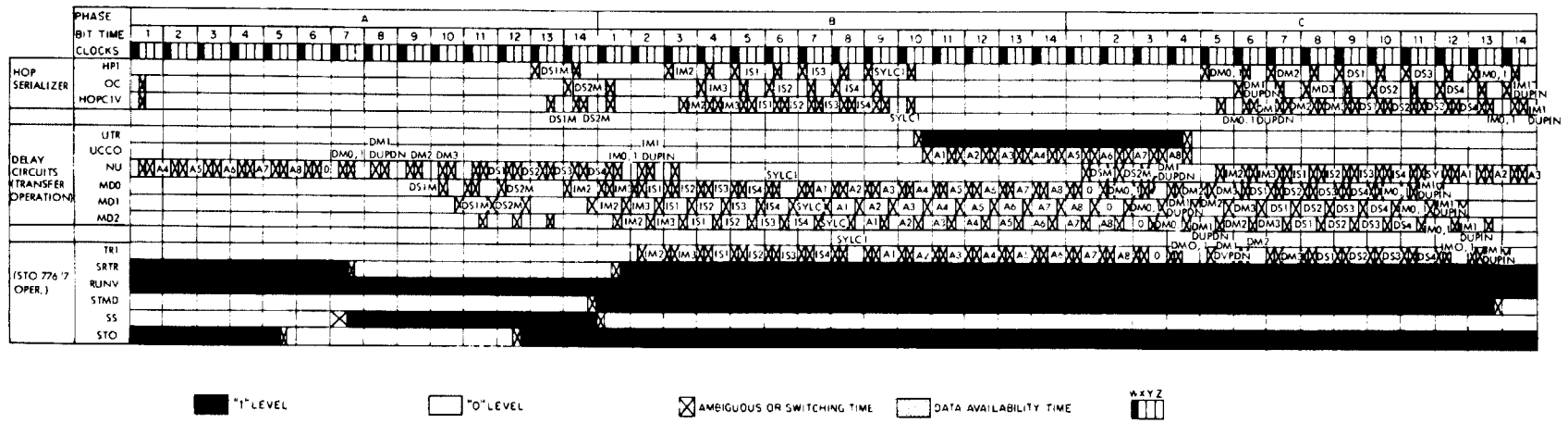
2-261. The DL31 delay line stores the HOP constant for 15 bit times plus 2 clock times. DL31 is loaded at W clock time (channel one) by the HOPC1 and ACC0 latches. UTR equals "0" from B-10-Z through C-4-X time during transfer operations to enable loading the instruction address via the ACC0 latch. The channel one output of DL31 is loaded into the NU latch at Y clock times. See figure 2-75 for delay circuit timing.

2-262. The DL44 delay line stores the HOP constant for 22 bit times plus a clock time. DL44 is loaded by the NU latch at W clock times (channel one); HOYN and VOYN equal "1" during non-multiply - divide operations. The channel one output of DL44 is loaded into the MD0 latch at X clock times.

2-263. The MD1 and MD2 latches provide a delay of two bit times from the time MD0 is loaded until the transfer register is loaded. The MD1 and MD2 latches sequentially load the HOP constant at Z and Y clock times, respectively, following the loading of MD0 at X clock time. When enabled, MD2 loads transfer register bit one (TR1) at X clock times of bit times B-2 through C-13. STMD, SRTR, and RUNV all equal "1" from B-1-Z through C-13-Y time during a store 776 or 777 instruction.

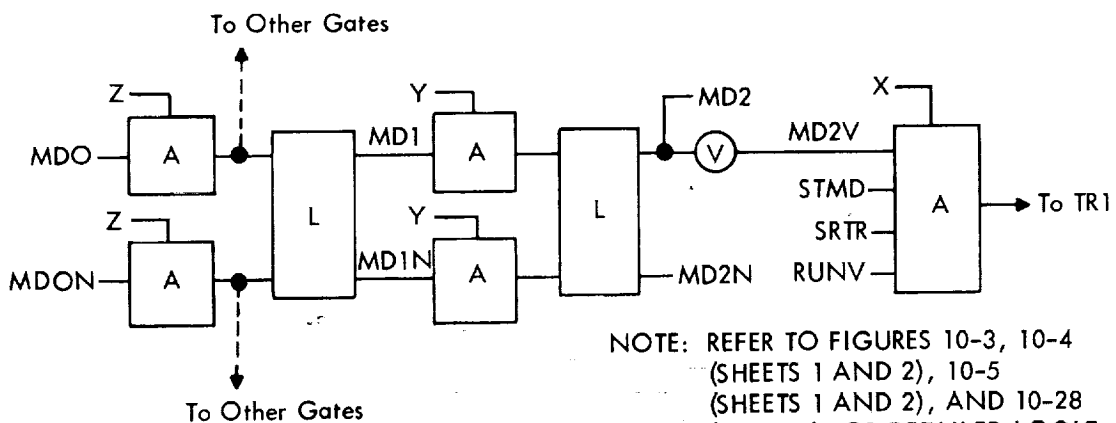
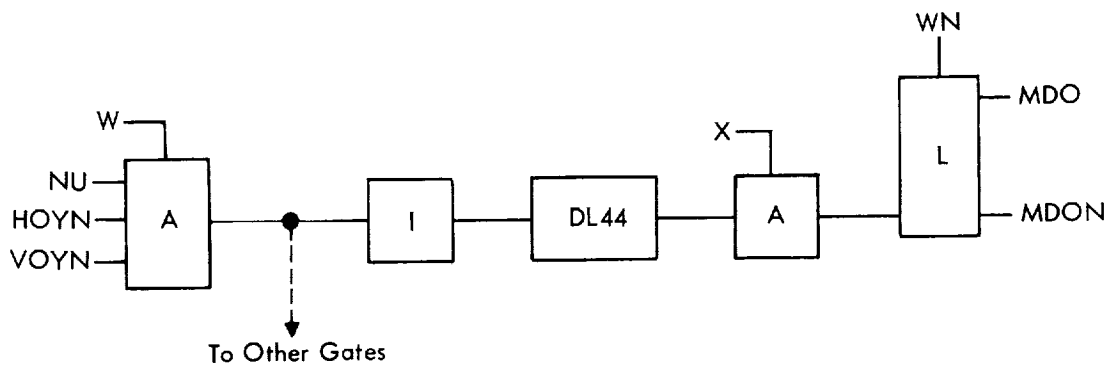
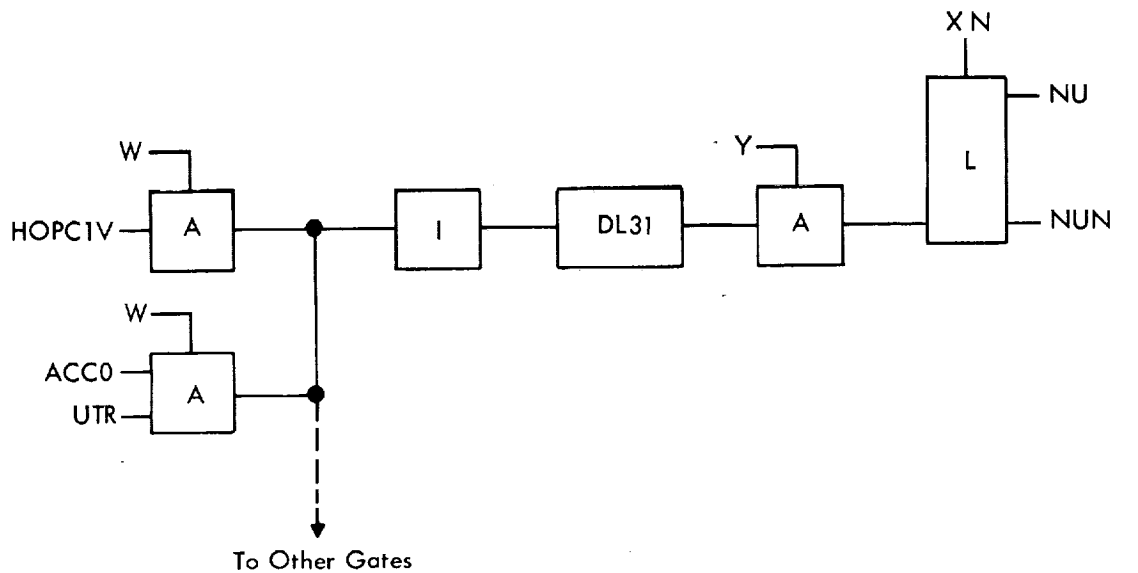
2-264. STMD Latch. (See figure 2-77.) During a store 776 or 777 instruction, which is normally commanded during an interrupt operation, the STMD latch is set to enable the transfer register to store the automatic HOP constant. The STMD latch is set, when enabled by STO and SS, from B-1-W through C-13-Y time. See figure 2-75 for STMD timing.

2-265. INSTRUCTION COUNTER. The instruction address is incremented in the Arithmetic Element and stored in the unused portion of the accumulator register. The instruction address consists of nine bits but only eight bits are stored in the accumulator register. When the instruction address is transferred to the address register from the accumulator register to select an instruction, a "0" is forced into address register bit



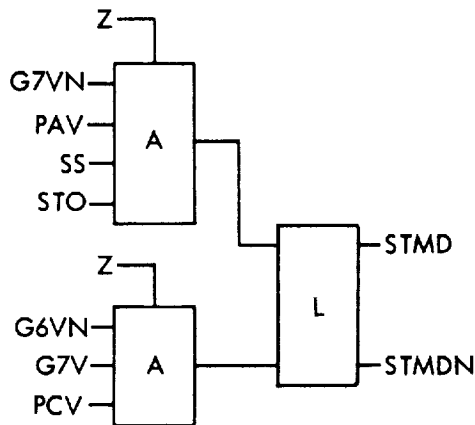
2-111

Figure 2-75. Automatic HOP Save Circuit Timing



NOTE: REFER TO FIGURES 10-3, 10-4 (SHEETS 1 AND 2), 10-5 (SHEETS 1 AND 2), AND 10-28 (SHEET 2) FOR DETAILED LOGIC.

Figure 2-76. Delay Circuits



NOTE: REFER TO FIGURE 10-15,
SHEET 1, FOR DETAILED
LOGIC.

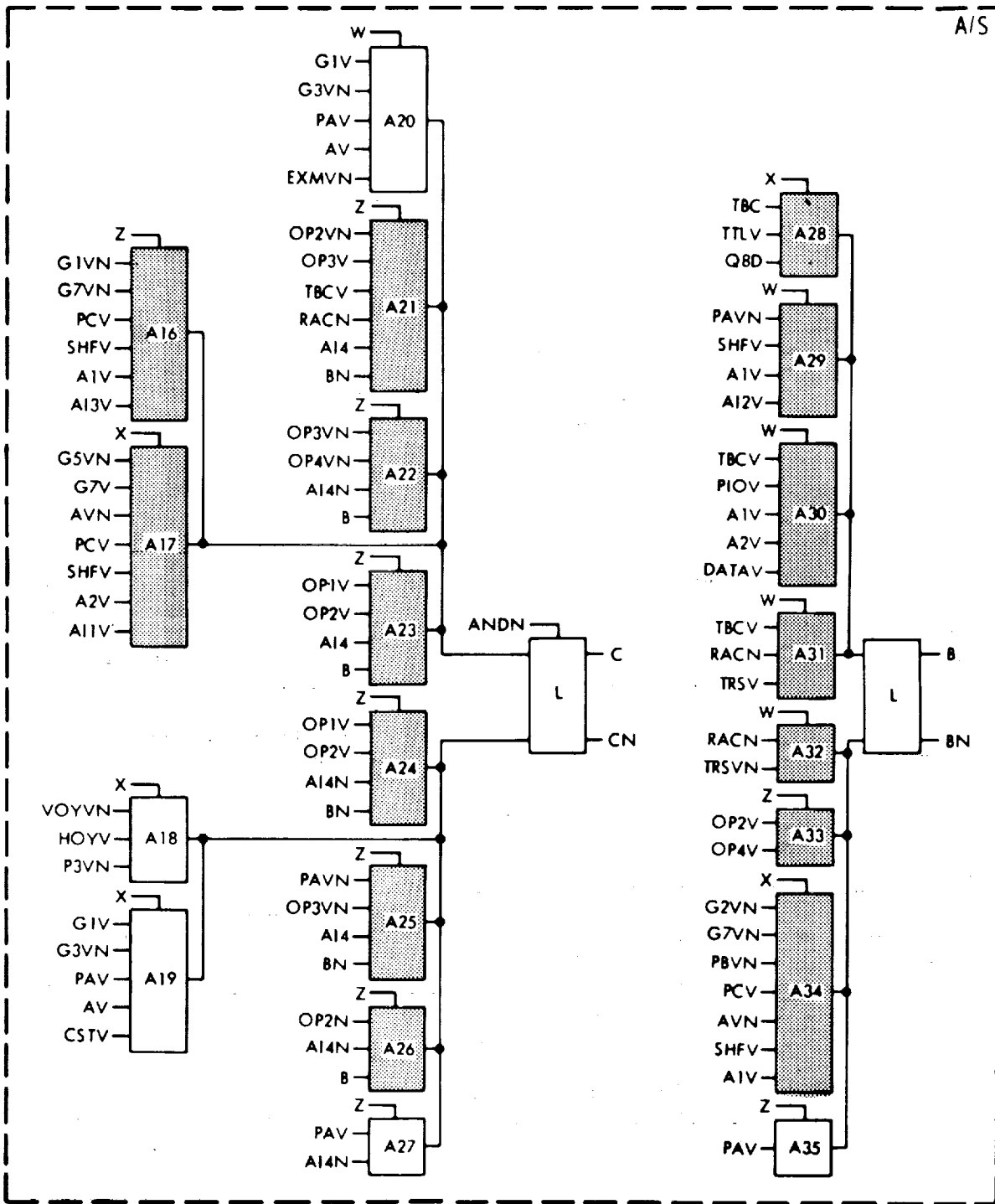
Figure 2-77. STMD Latch

A9 position. The instruction address is increased by one each instruction time during normal operation. The instruction address is not increased during an execute modified (EXM) instruction, during the first four computer cycles of a multiply and hold (MPH) operation, or after the commanded instruction is completed during computer single step (CST) operation. The instruction address is changed during a transfer operation; the instruction address currently stored in the accumulator-register is dropped and the transfer instruction operand address is loaded into the unused portion of the accumulator register. The Instruction Counter is actually part of the Arithmetic Element; for convenience, the Arithmetic Element circuits which increment the instruction address are duplicated in figure 2-78 (the circuits that are not used to increment and store the instruction address are shaded).

2-266. The instruction address is incremented by forcing a "1" into the C latch (gate A20), forcing the B latch to "0" during phase A (gate A35), and adding these two values, at the A/S circuit (gate A39 and A40), to the least significant digit of the instruction address when the bit arrives at AI4. The subsequent address bits are also added with the C and B latch values but the C latch is reset when the first "1" of the instruction address reaches AI4. (See Arithmetic Element for description of the add-subtract circuit.) Refer to figure 2-79 for C, B, and AI4 latch timing.

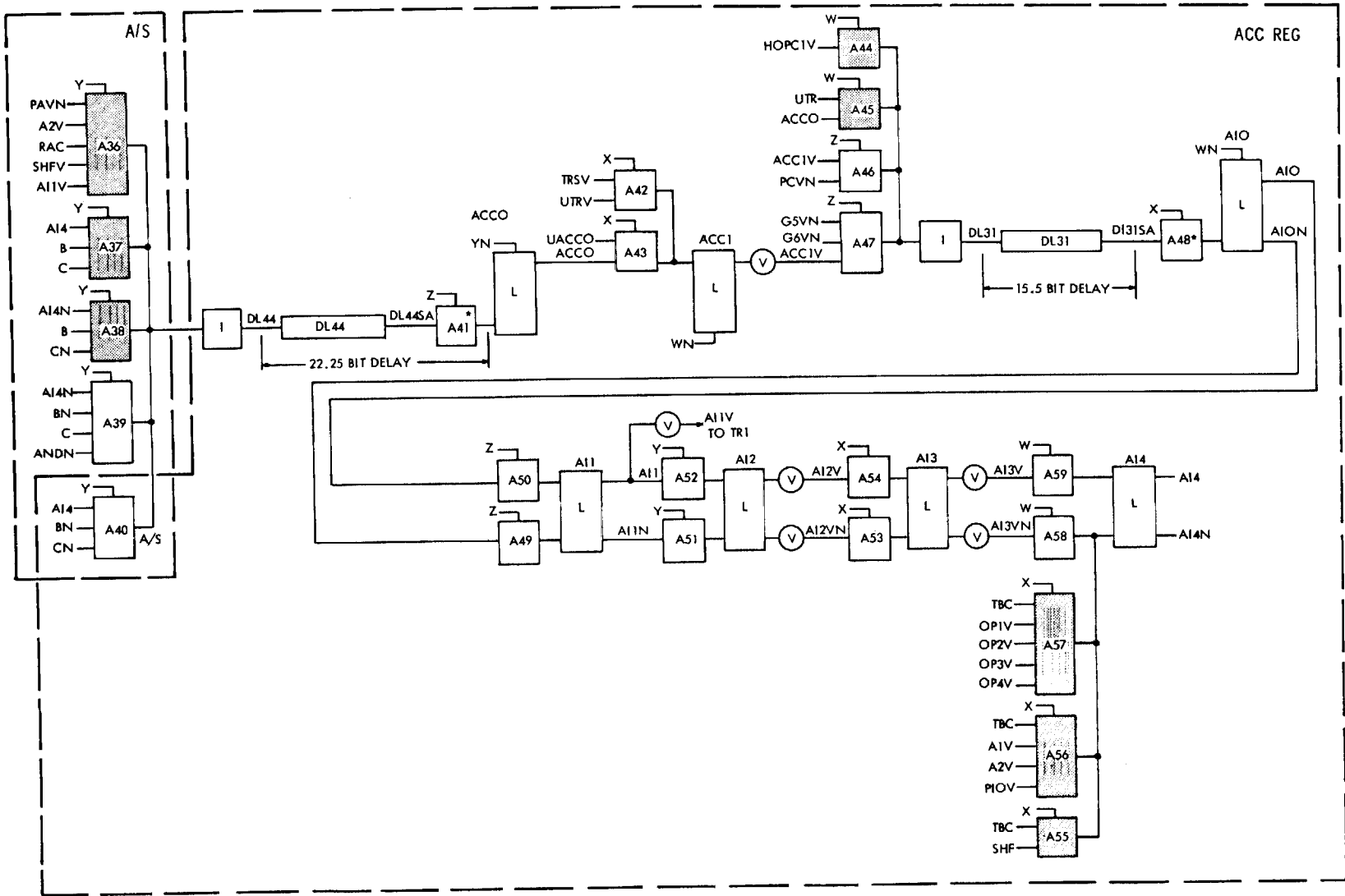
2-267. During normal operation, the C latch is reset when the first "0" of the instruction address is read from the accumulator register AI4 latch. (AI4N equals "1" to reset the C latch.) The C latch operation differs during EXM, MPH, and CST operation.

2-268. During an EXM instruction, EXMVN equals "0" to prevent the C latch from being set at A-2-W time, thus preventing the instruction address from being increased by one. The instruction address is also prevented from increasing during the first four computer cycles of an MPH operation; VOYVN, HOYV, and P3VN equal "1" enabling a X clock gate



NOTE: REFER TO FIGURE 10-5 SHEET 1, FOR DETAILED LOGIC

Figure 2-78. Instruction Counter (Sheet 1 of 2)



NOTE: REFER TO FIGURES 10-3, 10-5 (SHEET 2), AND 10-22 (SHEET 2) FOR DETAILED LOGIC

Figure 2-78. Instruction Counter (Sheet 2)

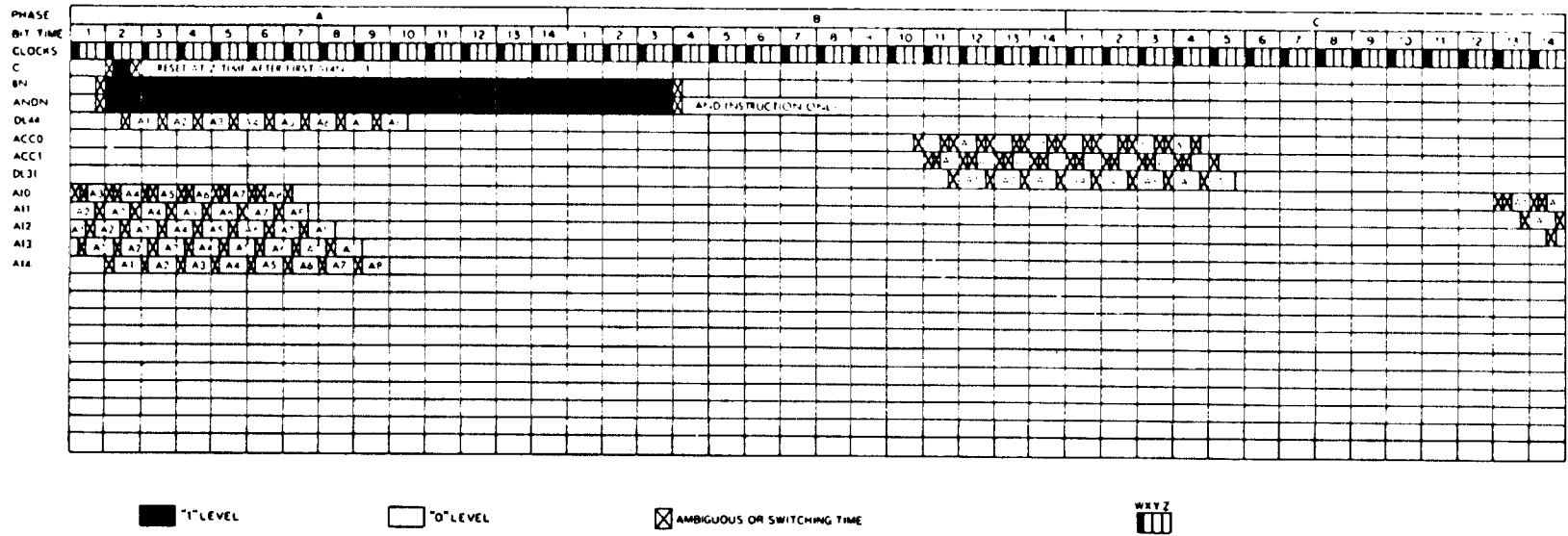


Figure 2-79. Instruction Counter Timing

to repeatedly reset the C latch. During the MPH fifth computer cycle P3VN equals "0" permitting the C latch to remain set after A-2-W time to increase the instruction address by one. The instruction address is also prevented from increasing during a CST operation while CST equals "1". The C latch is repeatedly reset at A-2-W time when CST equals "1" to keep the instruction address from increasing. To select the next programmed instruction during CST operation the externally controlled CST level becomes a "0" momentarily to allow the instruction address to increase by one.

2-269. The accumulator register stores the instruction address after the results of the arithmetic computations. The accumulator circulates the instruction address in the same manner that the arithmetic computations are circulated. The instruction address is loaded into the accumulator automatically by the add circuit and also, during transfer operations, by the transfer register serializer (TRSV). The accumulator also supplies the transfer register with the instruction address to select programmed instructions.

2-270. During a transfer operation, the transfer instruction operand address is serialized through the transfer register and TRSV latch to load ACC1 of the accumulator register. During this period, UTRV equals "1" and UACCO equals "0" to enable TRSV to load the new instruction address and inhibit the currently stored instruction address at ACCO. The UTRV and UACCO levels originate at the same latch so can never equal "1" simultaneously. UACCO equals "1" during non-transfer operations to enable the current instruction address to be circulated through the accumulator. The AII latch of the accumulator register sequentially loads the TR1 latch of the transfer register with the instruction address during bit times C-14 through A-7. The instruction address, while stored in the transfer register, selects the next programmed instruction.

2-271. ARITHMETIC ELEMENT.

2-272. The Arithmetic Element performs the following general type of operations:

- a. Arithmetic (addition, subtraction and shifting).
- b. Logical extraction.
- c. Stores new operand and a new instruction counter (previously described).
- d. Modifies the stored instruction counter.

2-273. The Arithmetic Element consists of the following (figure 2-80):

- a. Accumulator Control Circuits (Accum Control)
- b. Add/Subtract Circuit (A/S)
- c. Accumulator Register (Acc Reg)

The A/S Circuit and Acc Reg make up an accumulator which performs the arithmetic and logic operations and stores the results. The Acc Reg stores two pieces of data: the results of the arithmetic operation, called the stored operand, and the instruction counter.

2-274. The Accum Control Circuits provide timing and conditioning gates for the A/S and Acc Reg circuits. (The conditioning gates are instruction oriented.)

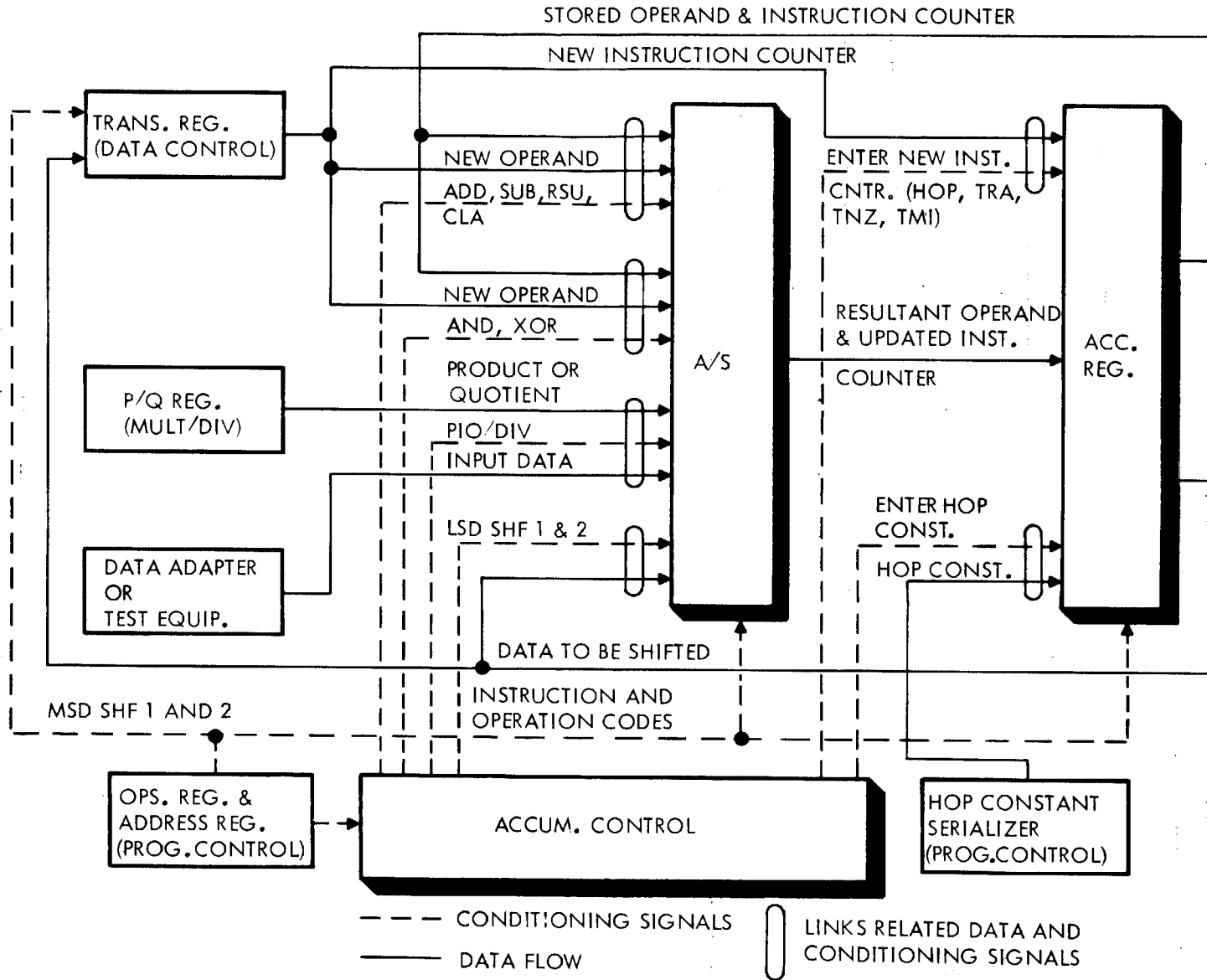


Figure 2-80. Arithmetic Element Block Diagram

2-275. The A/S Circuit performs the following arithmetic and logical extraction operations:

Addition	(ADD)	} Arithmetic
Subtraction	(SUB)	
Reverse-Subtraction	(RSU)	
Clear-and-Add	(CLA)	
And	(AND)	} Logical Extraction
Exclusive Or	(XOR)	

In each of the preceding operations only the stored operand and the new operand are affected. (The instruction counter is recirculated and updated automatically except during PIO operations.) During a CLA operation the stored operand is inhibited and the new operand enters the Acc Reg unaltered. Similarly, the product or quotient (from the Multiply-Divide Element) and input data (from the data adapter or test equipment) enters through the A/S circuit and replaces the stored operand in the Acc Reg. During a SHF operation, the stored operand leaves the Acc Reg and returns early to the Acc Reg directly through the A/S circuit for a LSD shift 1 or 2; the stored operand leaves the Acc Reg and returns from the Transfer Register late for a MSD shift 1 or 2. MSD-shifted data from the Transfer Register enters the A/S as a new operand.

2-276. The Acc Reg is a circulating shift register which utilizes the A/S circuit to complete the loop. When no operations are called forth, the stored operand and instruction counter are continually circulated; the stored operand is unaltered and the instruction counter updated during each pass through the A/S circuit. A new instruction counter (required when performing one of the transfer operations) or a new instruction counter and HOP constant (which becomes the stored operand) may be entered directly into the Acc Reg when performing a HOP.

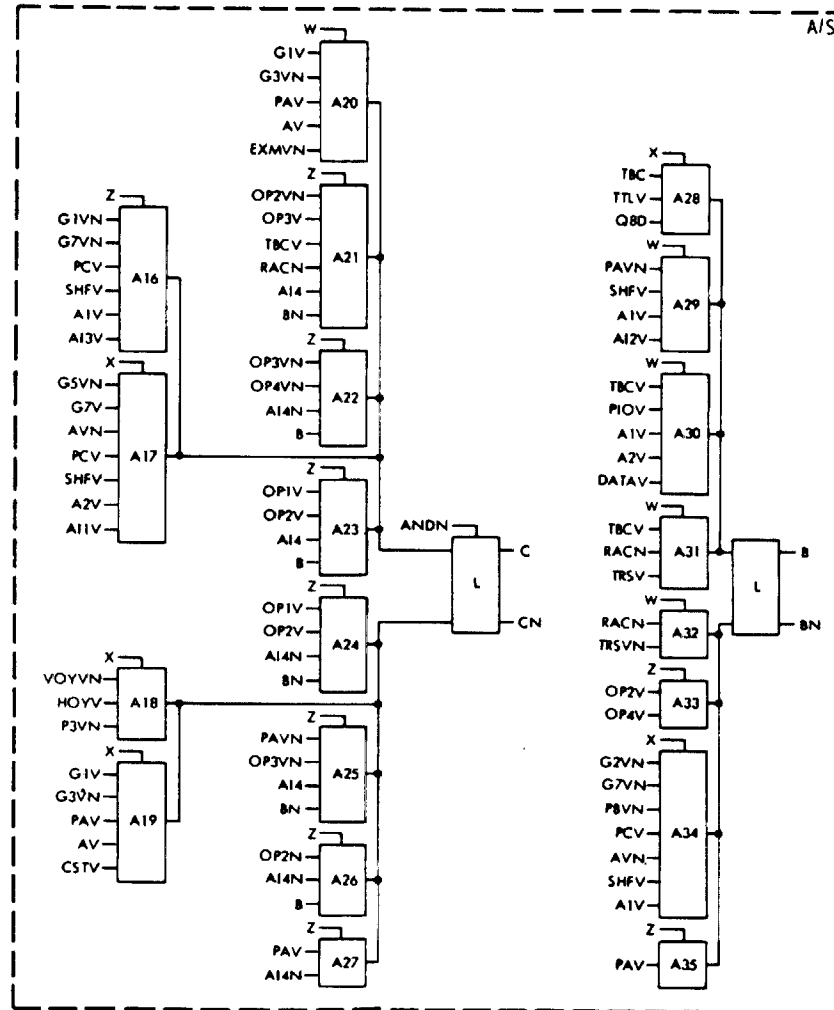
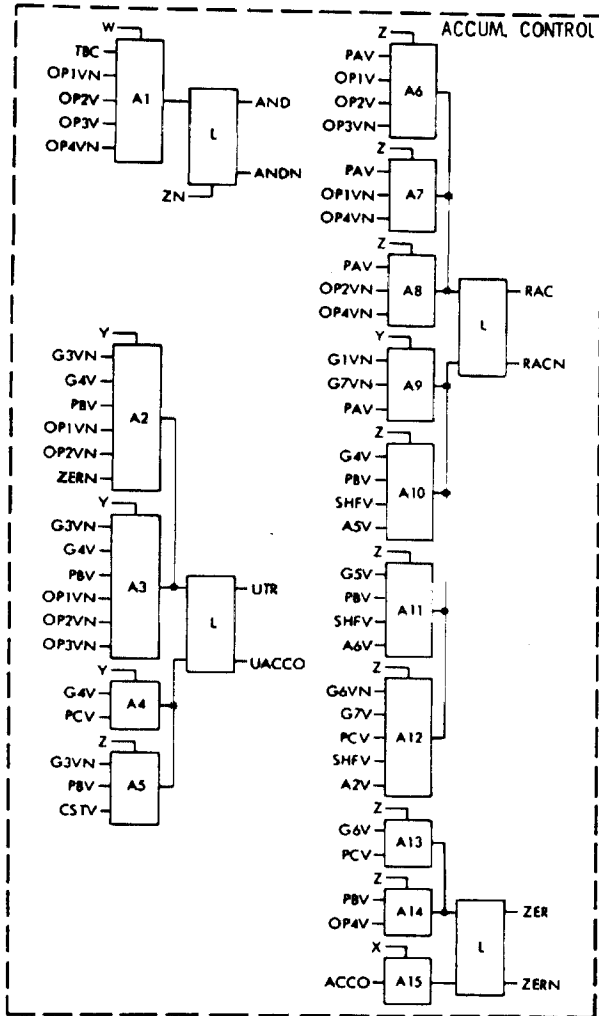
2-277. ARITH CONTROL. The Arith Control circuits consist of the following four latches (figure 2-81):

- a. AND
- b. RAC
- c. ZER
- d. UTR-UACCO

NOTE

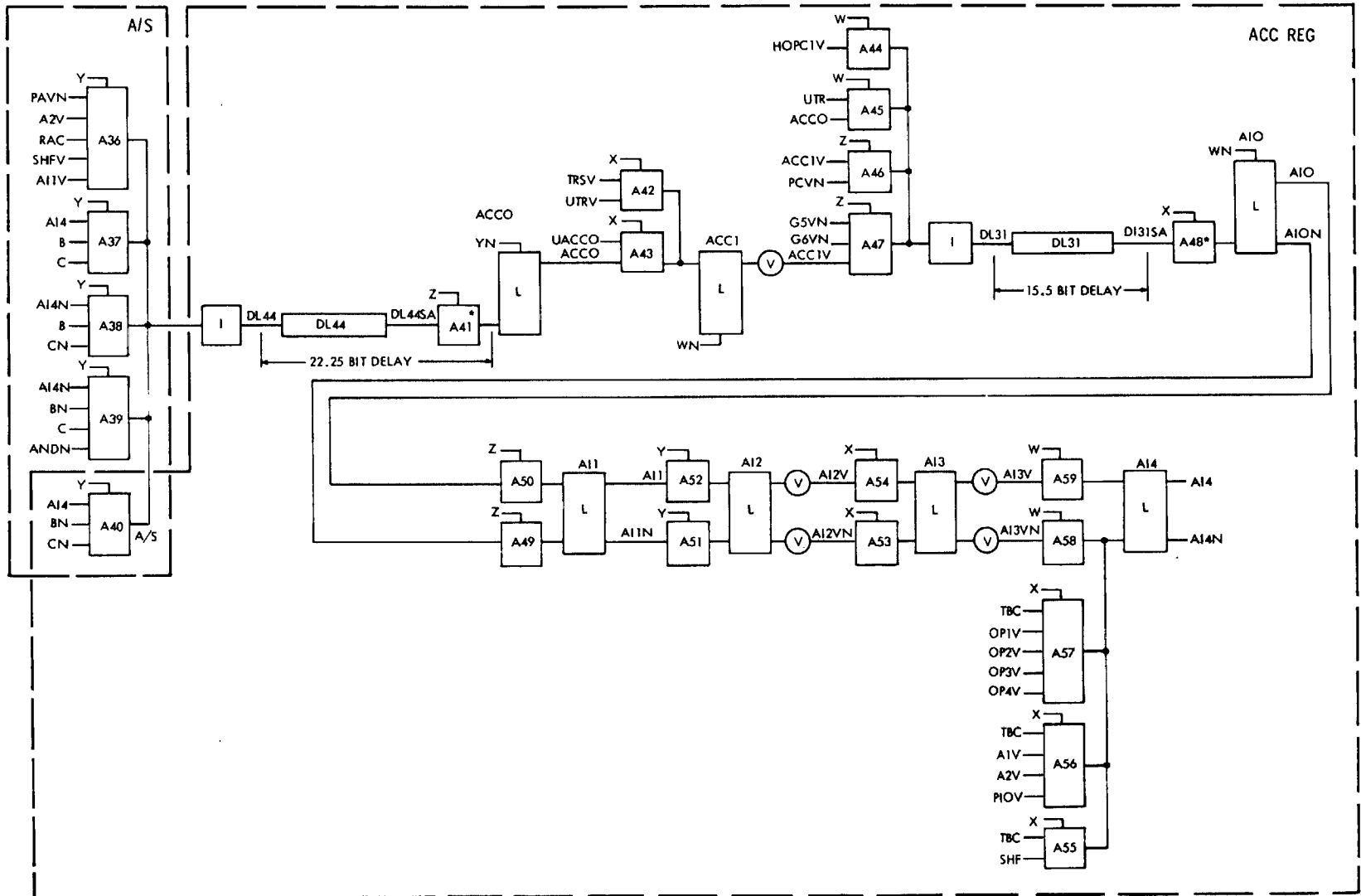
Refer to figure 2-82 as necessary for general timing information on the Arithmetic Element.

2-278. AND Latch. During an AND operation (defined by the input OP code) the AND latch is set at B-4-W, the beginning of an operation cycle (defined by TBC). When ZN goes to "0" (at Z time), the latch attempts to reset, but gate A1 overrides the zero drive. The latch is reset at A-2-W, the end of the AND operation cycle. ANDN is applied as a controlling gate to the C latch and summing gate A39.



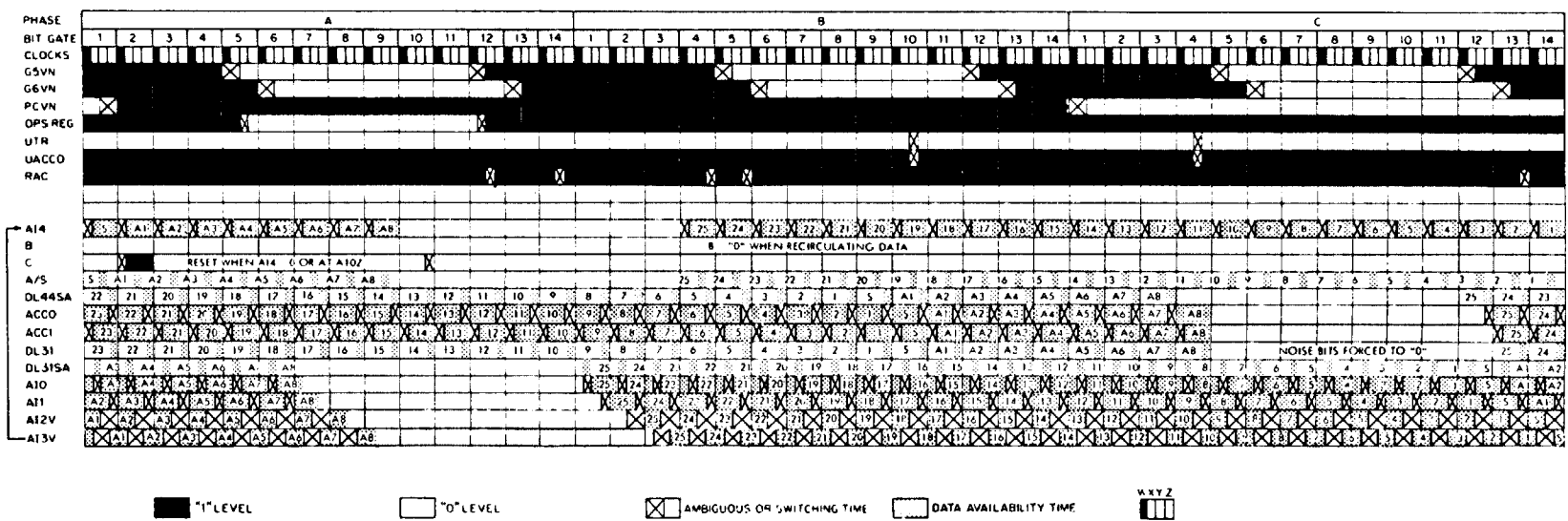
NOTE: REFER TO FIGURE 10-5 SHEET 1, FOR DETAILED LOGIC

Figure 2-81. Arithmetic Element (Sheet 1 of 2)



NOTE: REFER TO FIGURES 10-3, 10-5 (SHEET 2),
AND 10-22 (SHEET 2) FOR DETAILED
LOGIC

Figure 2-81. Arithmetic Element (Sheet 2)



Changed 4 January 1965

Figure 2-82. Arithmetic Element General Timing

2-279. RAC Latch. The RAC latch is set at A-12-Y time of the following operations by the indicated gates:

STO } DIV }	Gate A6
TMI } TRA } PIO } CDS } EXM } SHF }	Gate A7
MPY } MPH } TNZ } HOP }	Gate A8

NOTE

The following statement is true except for the MSD-shift operation.

When the RAC latch is set, the contents of the Acc Reg are recirculated because gates A31 and A32 of the B latch are closed. During a shift MSD-1 or -2 operation (A5V or A6V = "1"), gate A10 or A11 resets the RAC latch at B-4-Z or B-5-Z. RACN opens gates A21 and A32 to permit the transfer register to return the shifted stored operand to the Acc Reg as a new operand. During a shift LSD-2 operation (A2V = "1"), gate A12 resets the RAC latch at C-13-Z to close gate A36 and prevent the instruction counter from being LSD-shifted with the operand. Gate A9 is a housekeeping reset which assures a reset condition prior to the next operation cycle.

2-280. ZER Latch. (See figures 2-81 and 2-83.) The ZER latch stores the results of a check on the stored operand (ACCO). This check is necessary during TNZ and TMI operations. If the contents is not equal to zero or is negative (a "1" is present in ACCO), the latch is reset. Gate A13 sets the latch at C-6-Z time prior to each check of the data word. During a TMI instruction (OP4 = "1"), the latch is reset by gate A15 if ACCO contains a "1" and set by gate A14 from B-1-Z to B-14-Z.

NOTE

The reason that OP4 = "1" specifies TMI will be explained in the next paragraph.

2-281. UTR-UACCO Latch. (See figure 2-81.) The latch may be set to UTR when the following instructions are set in the Operation Code Register:

TNZ } TMI }	Gate A2
HOP } TRA }	Gate A3

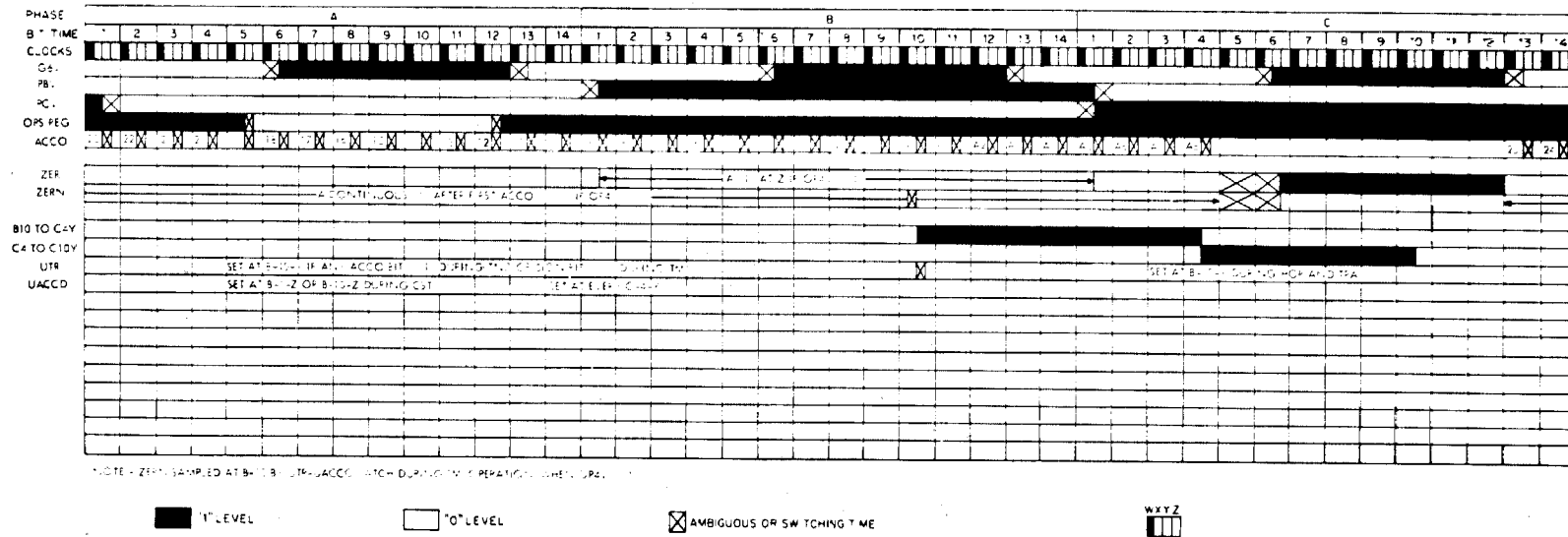


Figure 2-83. ZER Latch UTR-UACCO Latch Timing

NOTE

Gate A2 contains the code for all four instructions listed above. However since gate A3 contains the code for HOP and TRA alone, gate A2 only has to be considered in terms of TNZ (OP4VN = "1") and TMI (OP4V = "1"). All discussions concerning gate A2 will ignore the HOP and TRA possibilities.

Timing inputs G3VN, G4V, and PBV enable gates A2 and A3 at B-10-Y. If the conditions to set the latch to UTR = "1" are met, the latch cannot be reset until C-4-Y (gate A4). The period from B-10-Y to C-4-Y is the time when the instruction counter (A1 through A8) is available at the ACCO latch. (See figure 2-82.)

2-282. Figure 2-83 illustrates that, if OP4 = "1", the ZER latch is set each Z time after ACCO resets the latch; if OP4V = "0", the ZER latch is reset and stays reset the first time ACCO = "1". Since ZERN is an input to gate A2, OP4V has a bearing upon the state of UTR. If OP4V = "1", in combination with OP1VN and OP2VN, TMI is being performed; if OP4V = "0", TNZ is being performed. If OP4V = "1" and ZERN = "1" at B-10-Y, the ZER latch was reset by the sign bit in ACCO. Thus, UTR = "1" when ACCO is negative, satisfying the conditions for TMI. If OP4V = "0" and ZERN = "1" at B-10-Y, the ZER latch was reset by any bit in ACCO. Thus, UTR = "1" when ACCO is not zero, satisfying the conditions for TNZ.

2-283. A/S CIRCUIT. (See figure 2-81.) The A/S circuit consists of the B latch, C latch, and the A/S logic (gates A36 through A40). This circuit performs the arithmetic operations and logical extractions previously mentioned.

2-284. B Latch. (See figure 2-81.) The B latch provides the one-bit storage needed to enter serial data from the transfer register (TRSV and TRSVN) from the data adapter or test equipment (DATAV), or the contents of the P-Q register (Q8D) from the multiply-divide element. The B latch also provides the entry point for the stored operand (AI2) when the stored operand is being LSD-shifted one place (A1 = "1"). Gate A32 is used to serialize the transfer register data. Gate A33 is used to serialize the inputs to gates A28, A29, and A30. Gate A35 is a housekeeping reset to prepare the B latch for the next operation. In all cases the data output of the B latch occurs from B-4-W to A-1-W. Gate A34 resets the latch at A-1-X if the input bit (the A1 bit of the instruction counter is a "1"). This gate prevents the A1 bit from being shifted with the stored operand.

2-285. C Latch. (See figure 2-81.) Gate A27 resets the C latch prior to each operation cycle. Gates A21 through A23 provide a carry-borrow capability during arithmetic operations ADD, SUB, and RSU. (During CLA, no carry is required.) Gates 24 through 26 remove the carry-borrow condition when a carry or borrow is not needed. The C latch remains set (or reset) as long as a carry-borrow (or no carry-borrow) is needed. Note that the C latch is clocked at Z time; i. e. after the A/S gates (figure 2-81) have been clocked. Thus, the carry-borrow condition is always the result of the previous one-bit summation of AI4, B, and C.

2-286. During a XOR operation (described later) no condition exists to set the latch; thus the CN input to gates A38 and A40 acts as a logic conditioning signal. During an AND operation (also described later), ANDN = "0" and the C latch is set; thus the C input to gates A37 and A39 acts as a logic conditioning signal. (However, gate A39 is kept open because ANDN = "0".)

2-287. Gate A20 sets the C latch at A-2-W time in order to add one to the instruction counter A1 bit. This gate provides the instrumentation for updating the instruction address prior to each computer cycle.

2-288. A/S Logic. (See figure 2-81.) The A/S logic performs the arithmetic computations on inputs B, C, and AI4. These operations are performed by gates A37 through A40. Gate A36 is part of the loop for LSD-shifting data.

NOTE

The following paragraphs describe how the A/S Circuit operates.

2-289. **ADDITION, SUBTRACTION, REVERSE SUBTRACTION, AND CLEAR AND ADD LOGIC DESCRIPTION.** The A/S Circuit (figure 2-84) serially adds or subtracts the contents of the accumulator (AI4), the contents of the addressed memory location (B), and any carry (C) that develops from the previous bit summation. For each arithmetic operation the following equation is solved:

$$(ADD) A/S = AI4 + B + C \quad (1)$$

$$(SUB) A/S = AI4 - (B + C) \quad (2)$$

$$(RSU) A/S = B - (AI4 + C) \quad (3)$$

$$(CLA) A/S = B \quad (4)$$

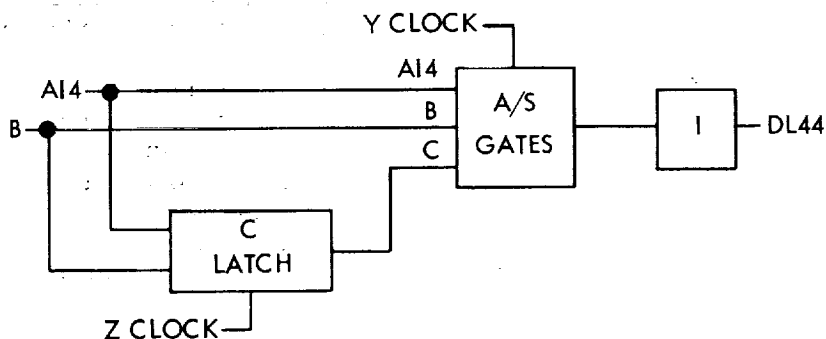


Figure 2-84. A/S Circuit, Simplified

2-290. Figure 2-85 shows the truth table for all combinations of values of AI4, B, and C required to solve equations (1) through (4). (Carry means the present value of carry that resulted from the last addition that will be used during this addition or subtraction; Next Carry means the value of the carry that will be used during the next addition or subtraction. The truth table shows that the values of A/S are the same for all addition and subtraction conditions of equations (1) through (4).

NOTE

Clear and Add (CLA) is a special addition when AI4 and C are both forced to zero. Gate 57 forces AI4 to zero. No condition exists to set the Carry latch once it is reset by gate 27.

			Sum or Remainder	ADD	SUB	RSU
AI4	B	Carry	A/S	Next Carry	Next Carry	Next Carry
0	0	0	0	0	0	0
1	0	0	1	0	0	1
0	1	0	1	0	1	0
1	1	0	0	1	0	0
0	0	1	1	0	1	1
1	0	1	0	1	0	1
0	1	1	0	1	1	0
1	1	1	1	1	1	1

← A

← D

← D

S

U

B

RSU

Note: A/S values are inverted to become DL44 in figure 2-83.

Figure 2-85. A/S Circuit Truth Table

2-291. Refer to figures 2-81 and 2-85. AND gates A37 through A40 define when A/S is a "1". Since the truth table for A/S is the same for ADD, SUB, and RSU, these gates are applicable for all three arithmetic operations. The gate which makes A/S = 1 when B = 1 and AI4 and C = 0 also applies to CLA. Thus, gates A37 through A40 apply for all four arithmetic operations.

2-292. The Carry latch is designed to change state according to the conditions established by the truth table (figure 2-85). The arrows indicate the values of B, AI4 and Carry which cause the Carry latch to change state for each arithmetic operation. The corresponding signals can be found on gates A21 through A26 (figure 2-81). For example: On

the second line of the truth table, if the Carry latch is reset (Carry = 0) and a "1" (AI4) is subtracted (during RSU) from a "0" (B) a borrow shall be generated (Next Carry = 1). OP code OP2VN and OP3V defines four operations. However, the only operation that is performed by the A/S Circuit is RSU. Since AI4 = 1 and B = 0 are possible conditions when the contents of the accumulator is recirculating, RACN is applied to prevent a carry from being generated during a recirculating condition. Gates A22 through A26 have OP code inputs which define their respective add-subtract operating conditions plus the respective values of B and AI4. The Carry latch will remain set until the Carry condition no longer exists. Gate A27 resets the Carry latch at the end of phase A to assure a zero carry prior to any arithmetic operations.

2-293. XOR LOGIC DESCRIPTION. During an XOR operation the contents of the Acc Reg (AI4) and the contents of the addressed memory location (B) are ORed exclusively bit-by-bit. The result is stored in the Acc Reg. The Add- Subtract Element is used to perform the operation. The following Boolean equation expresses the conditions required for an exclusive OR:

$$A/S = AI4 \cdot BN + AI4N \cdot B \quad (5)$$

2-294. The conditions which satisfy equation (5) are met by gates A38 and A40 (figure 2-81) provided the Carry latch is not set. Gates A21 and A22 have the AI4 and B conditions to set the Carry latch, but the input OP codes do not satisfy the XOR code. Thus, the Carry latch cannot be set during an XOR operation.

2-295. AND, LOGIC DESCRIPTION. During an AND operation the contents of the Accumulator (AI4) and the contents of the addressed memory location (B) are ANDed bit-by-bit. The result is stored in the Acc Reg. The A/S Circuit is used to perform the operation. The following Boolean equation expresses the conditions required for an AND:

$$A/S = AI4 \cdot B \quad (6)$$

2-296. Gate A37 satisfies the condition for AND as long as the Carry latch is set. During an AND operation ANDN = 0 (zero drive) and the Carry latch is set. Gate A39 is closed when ANDN = 0 to prevent "ones" from being introduced when AI4N and BN are "ones".

2-297. ACCUMULATOR REGISTER. The Acc Reg is a 3-phase, circulating shift register which stores the results of arithmetic and logical extraction operations and stores and updates the value of the instruction address. (See figure 2-81.) The Acc Reg, in conjunction with the A/S circuit and Accum Control, perform LSD shift operations. With the transfer register and A/S circuit, the Acc Reg also performs MSD shifts.

2-298. When no arithmetic operation is being performed, circulation of data is controlled by the following gates: A40, A43, A46 and A47, and A48 through A58. When G5VN and G6VN are "0", the bits between the instruction address and the data are forced to "0" by gate A47. Gates A37 through A40 are used for the arithmetic and logical extraction operations previously described. Gate A36 is used during a shift operation (described later). Gate A42 is used to enter the new instruction address during a transfer operation. The bits arrive at the same time that the previous address bits would arrive. Gates A44 and A45 are part of the HOP Save Circuit described previously. Gates A55, A56, and A57 inhibit the recirculation of data in the register during SHF, PIO, and CLA operations, respectively.

2-299. SHIFT LOGIC DESCRIPTION. Four shift operations can be commanded by the program:

LSD-1
LSD-2
MSD-1
MSD-2

NOTE

LSD stands for least significant digit.

A LSD-1 shift is a shift toward the LSD; i. e. , multiplying by 2^{-1} . A LSD-2 shift means to multiply by 2^{-2} . Similarly, a MSD-1 and a MSD-2 shift means to multiply by 2^1 and 2^2 , respectively. The contents of the Acc Reg is the data that is shifted. The direction and amount of shift is determined by the following control codes (in the operand address):

<u>Address Bit</u>	<u>Shift</u>
A1 = 1	LSD1
A2 = 1	LSD2
A5 = 1	MSD1
A6 = 1	MSD2

2-300. LSD-1 Shift. When an LSD-1 shift is required by the program, the voted output of the AI2 latch is applied to gate A29 of the B latch. (See figure 2-81.) The AI4 latch is reset by gate A55 at B-3-Y time, thus forcing bit 25 to "0" and inhibiting the normal recirculation of data. (See figure 2-86.) Starting at B-4-W time and continuing through C-14-W time (PAVN = 1), gate A29 sets the "1's" of the data word into the B latch. The A/S logic (gate A38) then re-enters the shifted data into the Acc Reg. At C-14-W time, the sign bit enters the B latch and leaves the A/S logic at C-14-Y time. Since bit 25 was dropped from the data word, the word is one bit short. This situation is corrected by adding the sign bit to the data word, an action which does not change the value of the data. At C-14-Y time the sign bit is stored in the AI3 latch. If the sign bit is a "1", AI3V sets the C latch (gate A16) at C-14-Z and an additional sign bit is added to the data word at A-1-Y time by the A/S logic (gate A39). At A-1-Y time, gate A55 is closed and the AI4 latch can start recirculating data again. The instruction count is therefore added to the shifted data immediately after the sign bit.

2-301. LSD-2 Shift. This shift is similar to the LSD-1 shift with three differences. (1) The AI1 latch output is applied directly to the A/S logic (gate A36); thus, data returns to the Acc Reg two bits early and bits 25 and 24 must be forced to zero. Bits 25 and 24 are entered into the Acc Reg through gate A36, but are forced to "0's" by gate A47. In effect, bit 23 of the shifted data becomes bit 25 after it was shifted. (2) The RAC latch is reset at C-13-Z time to open gate A36 and inhibit the instruction count from being shifted with the data. (3) Since the sign bit has been shifted two places and bits 25 and 24 are forced to "0's", two sign bits must be added to make the total number of bits 26 (25 bits plus sign). If the sign bit stored in the AI1 latch is a "1", the C latch is set at C-13-X time. The latch remains set until A-1-Y time when it is reset by gate A27. Thus, two additional sign bits are added to the data word during C-14 and A-1 times. Meanwhile the RAC latch is reset at C-13-Z time to close gate A36.

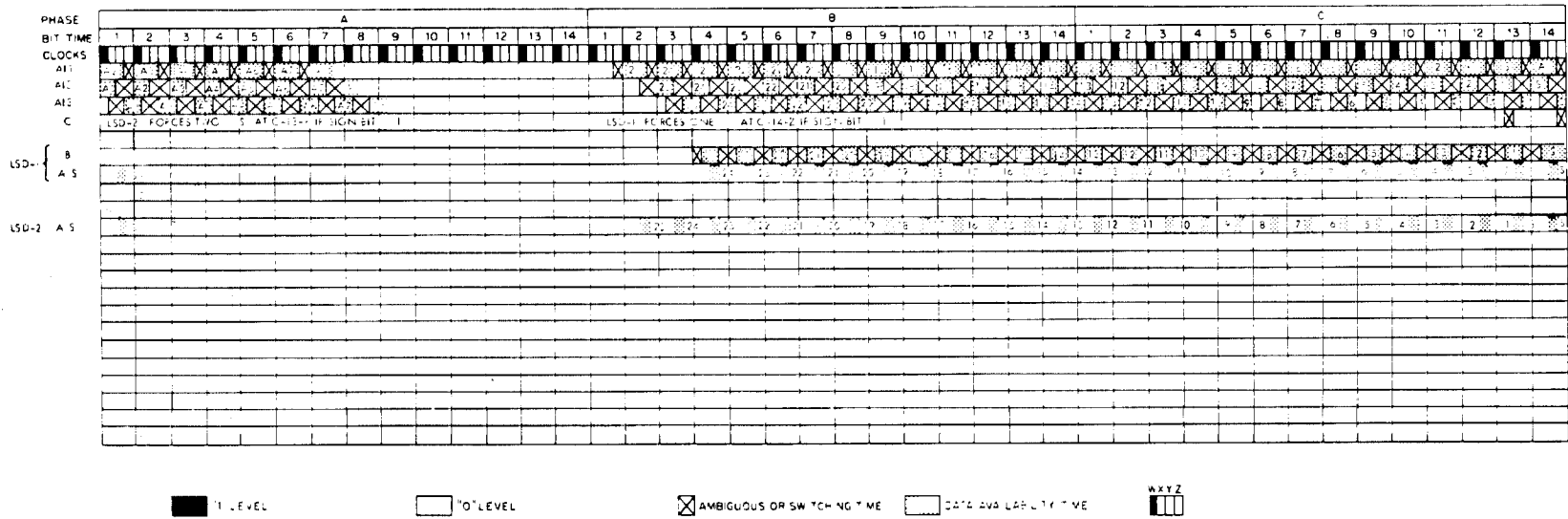


Figure 2-86. LSD-Shift Timing

2-302. MSD Shift. A data word is MSD shifted by circulating it through a longer than normal path. The contents of the AI2 latch are circulated through the Transfer Register and A/S circuit to make the path longer. (See figure 2-87.) This path is one bit longer for a MSD-1 shift and two bits longer for a MSD-2 shift. The most significant bit is lost when MSD-1 shifting and the two most significant bits are lost when MSD-2 shifting. (See figure 2-88.) The RAC latch is set by gate A7 to close gates A31 and A32 and inhibit the entry of data until the shifted data arrives. Thus, the low order bits are forced to "0's". At B-4-Z time (MSD-1), or B-5-Z time (MSD-2), gates A10 or A11 reset the RAC latch and enable the shifted data to enter the A/S circuit through gates A31 and A32. The B latch is reset at A-1-Y time to inhibit the partially shifted instruction count from the A/S circuit. The instruction count is recirculated by the AI4 latch in the normal manner. Inhibit-gate A55 is closed to permit recirculation of the instruction count.

2-303. MULTIPLY-DIVIDE ELEMENT.

2-304. The Multiply-Divide element contains the circuits which perform algebraic multiplication and division operations. Because many of these circuits are used in both multiplication and division, the two operations cannot be performed simultaneously. Consequently, this discussion treats the two operations separately.

2-305. MULTIPLICATION. The LVDC implements a "serial by four parallel" multiplication technique in which the serial multiplicand is multiplied by four multiplier bits in parallel, each iteration. The multiplier bits are evaluated five at a time with a one-bit overlap between iterations. The overlap in the multiplier bit evaluation sequence provides automatic sign correction and facilitates the use of two's complement arithmetic. This multiplication technique is best described by deriving the formula instrumented in the computer.

2-306. Derivation of Multiply Formula. Any fractional number, b , in the two's complement numbering system is defined by the series

$$b = -b_{\text{Sign}} \times 2^0 + b_1 \times 2^{-1} + b_2 \times 2^{-2} + \dots + b_n \times 2^{-n} + 0 \times 2^{-(n+1)} \quad (1)$$

One of the values necessary for the derivation is $b/2$. One-half b is shown by the product $b \times 2^{-1}$.

$$2^{-1} \times b = -b_{\text{Sign}} \times 2^{-1} + b_1 \times 2^{-2} + b_2 \times 2^{-3} + \dots + b_{n-1} \times 2^{-n} + b_n \times 2^{-(n+1)} \quad (2)$$

However, the number in equation (2) is no longer in the correct form for a two's complement fraction. The proper form can be restored by adding equation (2) to the proper expression for zero.

$$0 = -b_{\text{Sign}} \times 2^0 + 2b_{\text{Sign}} \times 2^{-1}$$

$$2^{-1} \times b = \frac{-b_{\text{Sign}} \times 2^{-1} + b_1 \times 2^{-2} + b_2 \times 2^{-3} + \dots + b_{n-1} \times 2^{-n} + b_n \times 2^{-(n+1)}}{-b_{\text{Sign}} \times 2^0 + b_{\text{Sign}} \times 2^{-1} + b_1 \times 2^{-2} + b_2 \times 2^{-3} + \dots + b_{n-1} \times 2^{-n} + b_n \times 2^{-(n+1)}} \quad (3)$$

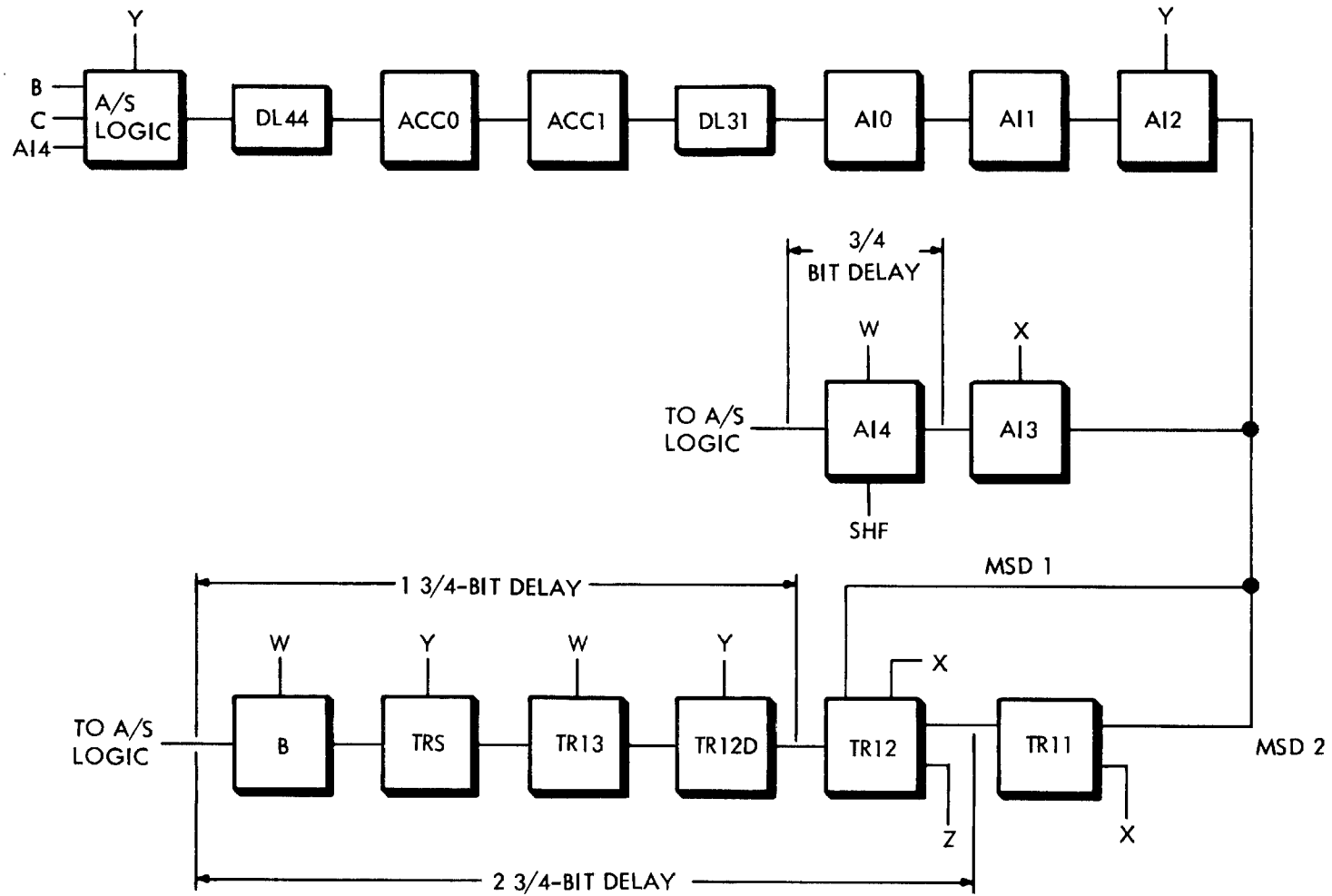
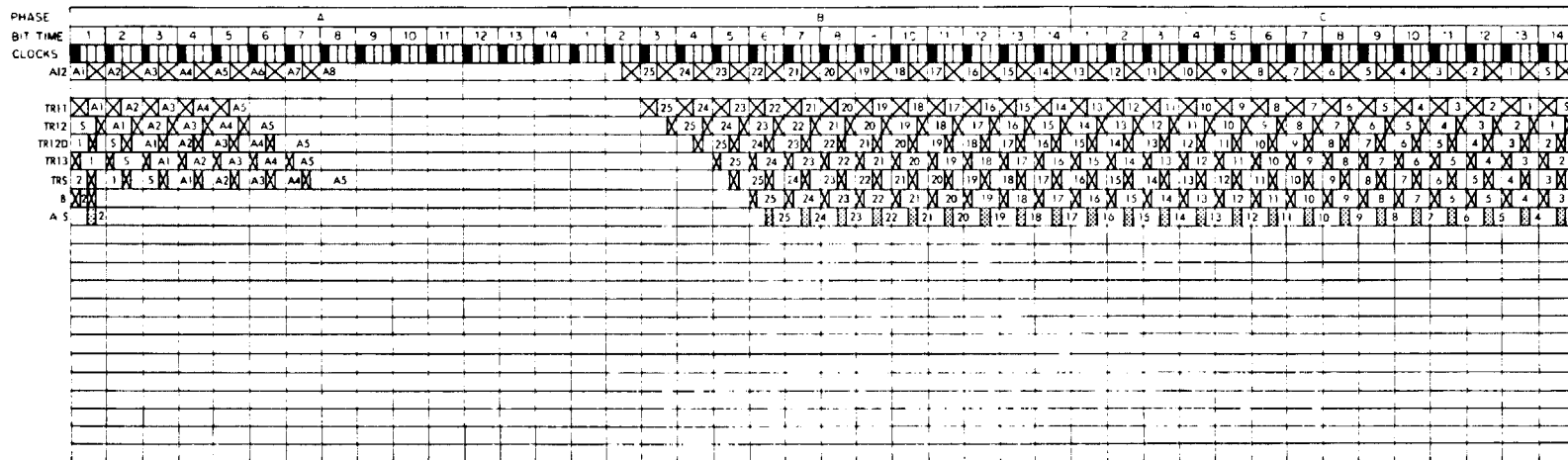


Figure 2-87. MSD-Shift Block Diagram



"1" LEVEL
 "0" LEVEL
 X AMBIGUOUS OR WAIT HOLD TIME
 DATA AVAILABILITY TIME
 A1-Z

NOTE: (1) SEE FIGURE 2-87.
 (2) SEE FIGURE 2-83 FOR TIMING OF ACC REG
 (3) TIMING IS SHOWN FOR MSD-2 SHIFT. MSD-1 SHIFT SAME, EXCEPT A12 APPLIED TO TR12 CAUSING ONE LESS BIT OF DELAY

Figure 2-88. MSD-Shift Timing

2-307. Another method of representing the number, b , is derived as follows.

$$b = 2 (b - 1/2 b) \quad (4)$$

Substituting equations (1) and (3) for b and $1/2 b$, respectively, gives the expression

$$\begin{aligned} b = 2 \left[- (b_{\text{Sign}} \times 2^0) + (b_1 \times 2^{-1}) + (b_2 \times 2^{-2}) + (b_3 \times 2^{-3}) + \dots + (b_n \times 2^{-n}) \right. \\ \left. + (0 \times 2)^{-(n+1)} + (b_{\text{Sign}} \times 2^0) - (b_{\text{Sign}} \times 2^{-1}) - (b_1 \times 2^{-2}) - (b_2 \times 2^{-3}) - \dots \right. \\ \left. - (b_{n-1} \times 2^{-n}) - (0 \times 2)^{-(n+1)} \right] \quad (5) \end{aligned}$$

Combining terms gives

$$\begin{aligned} b = (b_1 - b_{\text{Sign}}) 2^0 + (b_2 - b_1) 2^{-1} + (b_3 - b_2) 2^{-2} + \dots + (b_n - b_{n-1}) 2^{-n} \\ + (0 - b_n) 2^{-(n+1)} \quad (6) \end{aligned}$$

The expression in equation (6) is the key to instrumenting two's complement mathematics for multiplication. Looking back at equation (1), notice that the sign bit of the number b requires special consideration because it is negative and all the other bits are positive. Equation (6) gives a method of representing a two's complement fraction in which the sign bit is treated exactly the same as every other bit. A number represented in this form can, therefore, be instrumented as an operand for an arithmetic operation without regard for its sign. Notice also the zero in the last term of equation (6); its significance will be noted subsequently.

2-308. If the number, b , as represented in equation (6), is used as the multiplier in forming the product $P = b \times M$, then

$$\begin{aligned} P = (b_1 - b_{\text{Sign}}) 2^0 \times M + (b_2 - b_1) 2^{-1} \times M + (b_3 - b_2) 2^{-2} \times M + \dots \\ + (b_n - b_{n-1}) 2^{-n} \times M + (0 - b_n) 2^{-(n+1)} \times M \quad (7) \end{aligned}$$

Factoring $1/2$ from the second and each subsequent term and factoring $1/2$ from each group of terms that still contains a negative power of 2 gives the expression:

$$\begin{aligned} P = (b_1 - b_{\text{Sign}}) M + 1/2 \left[(b_2 - b_1) M + 1/2 \left\{ (b_3 - b_2) M + \dots \right. \right. \\ \left. \left. + 1/2 \left[(b_n - b_{n-1}) M + 1/2 (0 - b_n) M \right] \right\} \right] \quad (8) \end{aligned}$$

If the product is formed by an iterative process which evaluates one term of the multiplier per iteration, then

$$P_1 = (0 - b_n) M \quad (9)$$

$$P_2 = \frac{P_1}{2} + (b_n - b_{n-1}) M \quad (10)$$

$$P_3 = \frac{P_2}{2} + (b_{n-1} - b_{n-2}) M, \text{ etc} \quad (11)$$

and it can be said that

$$P_i = \frac{P_{i-1}}{2} + (b_{n+2-i} - b_{n+1-i}) M \quad (12)$$

where

P = product
 M = multiplicand
 i = iteration number (1, 2, 3 ... n)
 b = multiplier bit
 n = number of bits in multiplier

2-309. Equation (12) is the basic formula for multiplying two's complement numbers using two's complement arithmetic. Note that the formula contains no terms whose purpose is determining the sign of the product; this is one of the most important characteristics of this formula.

2-310. This method of multiplication examines two bits each iteration, one new bit and one bit from the previous iteration. Thus, each bit except the sign bit is examined twice in the course of a complete multiply operation. During the first iteration, there is no bit from a previous iteration. To account for this condition, an extra bit is added to the least significant digit (LSD) of the multiplier. This bit is the zero in the last term of equation (6), noted previously.

2-311. Because each bit must be examined twice, this method of multiplication steps through the multiplier only one bit per iteration. The multiply formula implemented in the computer is an expansion of equation (12). The expanded formula steps through four multiplier bits per iteration. This formula is derived by expanding equation (12) for product P_4 , as shown.

2-312. From the formula in equation (12)

$$P_1 = \frac{P_0}{2} + (b_{n+1} - b_n) M \quad (13)$$

$$P_2 = \frac{P_0}{4} + \left(\frac{b_{n+1} - b_n}{2} + b_n - b_{n-1} \right) M \quad (14)$$

$$P_3 = \frac{P_0}{8} + \left(\frac{b_{n+1} - b_n}{4} + \frac{b_n - b_{n-1}}{2} + b_{n-1} - b_{n-2} \right) M \quad (15)$$

$$P_4 = \frac{P_0}{16} + \left(\frac{b_{n+1} - b_n}{8} + \frac{b_n - b_{n-1}}{4} + \frac{b_{n-1} - b_{n-2}}{2} + b_{n-2} - b_{n-3} \right) M \quad (16)$$

Multiplying by 16 (2^4) gives

$$2^4 P_4 = P_0 + (2 b_{n+1} - 2 b_n + 4 b_n - 4 b_{n-1} + 8 b_{n-1} - 8 b_{n-2} + 16 b_{n-2} - 16 b_{n-3}) M \quad (17)$$

Combining terms selectively, gives

$$2^4 P_4 = P_0 + (2 b_{n+1} + 2 b_n - 4 b_{n-1}) M + (8 b_{n-1} + 8 b_{n-2} - 16 b_{n-3}) M \quad (18)$$

The computer is instrumented to compute the product $2^4 P_4$ in the first iteration and, for the computer, $2^4 P_4$ becomes $2^4 P_1$. The iteration number must, therefore, be redefined and the subscripts of the multiplier bits must be modified to step four bits per iteration. This modification results in the expression

$$2^4 P_i = P_{i-1} + (2 b_{n+5-4i} + 2 b_{n+4-4i} - 4 b_{n+3-4i}) M + (8 b_{n+3-4i} + 8 b_{n+2-4i} - 16 b_{n+1-4i}) M \quad (19)$$

where

i = iteration number (1, 2, 3, . . . $n/4$)

2-313. Equation (19) is the basic multiplication formula implemented in the computer. This expression can be simplified by using the following notation

$$2^4 P_i = P_{i-1} + \Delta 1_i + \Delta 2_i \quad (20)$$

where

$$\Delta 1_i = (2 b_{n+5-4i} + 2 b_{n+4-4i} - 4 b_{n+3-4i}) M$$

$$\Delta 2_i = (8 b_{n+3-4i} + 8 b_{n+2-4i} - 16 b_{n+1-4i}) M$$

2-314. Equation (20) is the general multiply formula that the computer has been instrumented to solve.

2-315. Multiplication Process. The computer multiplies by an iterative process which samples five bits of the multiplier per iteration. The subscripts of the multiplier bits in the expressions for $\Delta 1_i$ and $\Delta 2_i$ (20) specify which bits will be sampled during each iteration. For example, during the first iteration $i = 1$ and bits 25, 24 and 23 determine $\Delta 1_1$ and bits 23, 22 and 21 determine $\Delta 2_1$. Similarly, bits 21, 20 and 19 determine $\Delta 1_2$ and bits 19, 18 and 17 determine $\Delta 2_2$ in the second iteration when $i = 2$.

2-316. The examples above point out the overlap features of the multiply formula. The high-order bit of each five-bit group is the low-order bit of the next group. The high-order bit of the final group is the multiplier sign bit, which controls forming the final product. An overlap is also apparent between the expressions for $\Delta 1_i$ and $\Delta 2_i$; the high-order bit used to determine $\Delta 1_i$ is also the low-order bit used to determine $\Delta 2_i$. These overlaps permit the multiplication process to operate independent of sign.

2-317. It is also apparent that $\Delta 1_i$ is determined by the three low-order bits of each five-bit group sampled; similarly, $\Delta 2_i$ is determined by the three high-order bits of each group. By designating the bits in each group as MB1, MB2, MB3, MB4 and MB5 (with MB1 the lowest-order bit and MB5 the highest), the table in figure 2-89 can be developed. Here, $\Delta 1_i$ is determined by MB1, MB2 and MB3 while $\Delta 2_i$ is determined by MB3, MB4 and MB5.

2-318. Note that all the values of $\Delta 1_i$ and $\Delta 2_i$ are the product of the multiplicand and a power of two. These values all can be formed by shifting the multiplicand. Forming these values by shifting eliminates the sum and difference operations in the expressions for $\Delta 1_i$ and $\Delta 2_i$ (20) leaving only two sums to be instrumented in the computer. This also explains the selection used to form equation (18).

2-319. The following general steps outline the computer's multiplication process:

- 1) Determine the value of $\Delta 1_i$ from MB1, MB2 and MB3.
- 2) Determine the value of $\Delta 2_i$ from MB3, MB4 and MB5.
- 3) Compute the sum $P_{i-1} + \Delta 2_i$.

- 4) Compute the sum $(P_{i-1} + \Delta 2_i) + \Delta 1_i$ to form $2^4 P_i$.
- 5) Shift $2^4 P_i$ four places toward the LSD to form P_{i-1} for the next iteration.
- 6) Shift the multiplier four places toward the LSD to bring four new bits into sampling position for the next iteration.
- 7) Repeat steps 1 thru 6 until product P_6 is formed.

MB3	MB2	MB1	$\Delta 1_i$	
MB5	MB4	MB3		$\Delta 2_i$
0	0	0	0	0
0	0	1	+2M	+8M
0	1	0	+2M	+8M
0	1	1	+4M	+16M
1	0	0	-4M	-16M
1	0	1	-2M	-8M
1	1	0	-2M	-8M
1	1	1	0	0

Figure 2-89. Table of Delta Values

2-320. Some insight to the multiplication process can be gained by placing a single "1" in the multiplier at various positions, then evaluating its effect on the partial products. Figure 2-90 is an aid to such a study. The figures in the columns give the effective value each bit has on every quantity formed in the course of a multiply operation.

2-321. For example, consider a multiplier with a "1" in magnitude-bit 23 only. If the multiplier is assumed to be a fraction with its binary point after the sign, bit 23 should result in the value $2^{-23}M$ in the final product. Bit 23 is evaluated in the first iteration and therefore, any value it causes in the partial product will be shifted toward the LSD twenty four places in the course of the complete multiply. $2M$ is the value which, when shifted twenty four places toward the LSD, results in $2^{-23}M$. Figure 2-90 shows that bit 23 is evaluated correctly in forming $\Delta 1_1$ and that this evaluation results in $2^{-23}M$ in the final product.

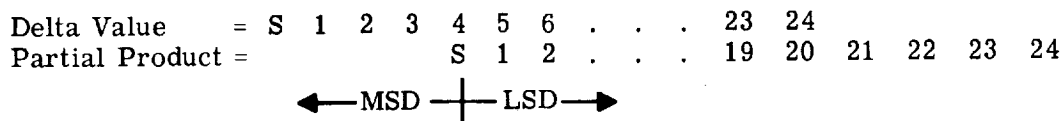
2-322. Consider also, a multiplier with a "1" in magnitude-bit 20 only. Again when the multiplier is a fraction with its binary point after the sign, bit 20 should cause the value $2^{-20}M$ in the final product. Bit 20 is evaluated during iteration one and, like bit 23, the value it causes will be shifted toward the LSD twenty four places in the course of the complete multiplication. The value $16M$, when thus shifted, gives the proper result ($2^{-20}M$) in the final product. But bit 20 causes $\Delta 2_1$ to take the value $-16M$, obviously an error.

Iteration Number	i = 6				i = 5				i = 4				i = 3				i = 2				i = 1				Extra Bit						
	MB5	MB4	MB3	MB2	MB1/5	MB4	MB3	MB2	MB1/5	MB4	MB3	MB2	MB1/5	MB4	MB3	MB2	MB1/5	MB4	MB3	MB2	MB1/5	MB4	MB3	MB2		MB1/5	MB4	MB3	MB2	MB1	
Multiplier Bits	SIGN	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23							
Δ_{1_1}																															
Δ_{2_1}																															
$2^4 P_1$																															
P_1																															
Δ_{1_2}																															
Δ_{2_2}																															
$2^4 P_2$																															
P_2																															
Δ_{1_3}																															
Δ_{2_3}																															
$2^4 P_3$																															
P_3																															
Δ_{1_4}																															
Δ_{2_4}																															
$2^4 P_4$																															
P_4																															
Δ_{1_5}																															
Δ_{2_5}																															
$2^4 P_5$																															
P_5																															
Δ_{1_6}																															
Δ_{2_6}																															
$2^4 P_6$																															
P_6																															

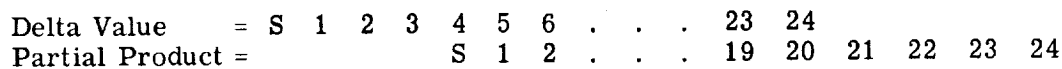
Figure 2-90. Table of Multiplier Bit Effective Values

2-323. However, bit 20 is evaluated again during iteration two. Values resulting from this evaluation will be shifted toward the LSD only twenty places in the final product. The correct value to shift in this iteration is +M. Bit 20 causes $\Delta 1_2$ to take the value +2M, but $\Delta 2_1$ has since been reduced from -16M to -M by the first LSD shift of four places. Thus, when these two values are summed, the total effect of bit 20 is +M in iteration two, the correct value. These operations are easily followed in figure 2-90.

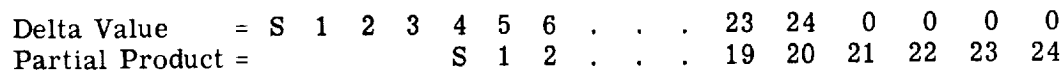
2-324. As noted previously, the values of $\Delta 1_i$ and $\Delta 2_i$ are formed by shifting the multiplicand toward the MSD from one to four places. The shift is relative to the partial product which offsets the delta values from the partial products, causing a problem in the summations required by the multiply formula (20). Counting from the partial product sign, the delta values have fewer bits toward the LSD than the partial products and the partial products have fewer bits toward the MSD than the delta values, thus



The delta values are easily corrected by forcing vacated bits to zeros as the number is shifted. Adding zeros to the LSD end of a number does not change its value.



The same solution, however, is not valid for extending the MSD end of the partial products. In order to extend a two's complement number to include more significant digits, the added bits must be filled with the sign of the number.



If the partial product is positive, the blank positions are filled with zeros. If the partial product is negative, it is in two's complement form and, when extended, must remain in the correct form for a two's complement number, equation (1). The reason for filling positions MSD of the sign with the sign is shown by evaluating a negative number containing all ones before and after extension.

Before Extension

Form $-b_{\text{Sign}} \times 2^0 + b_1 \times 2^{-1} + b_2 \times 2^{-2} + \dots$

Value $-1 + 1/2 + 1/4 + \dots$

After Extension

Form $-b_{\text{Sign}} \times 2^4 + b_{\text{Sign}} \times 2^3 + b_{\text{Sign}} \times 2^2 + b_{\text{Sign}} \times 2^1 + b_{\text{Sign}} \times 2^0$
 $+ b_1 \times 2^{-1} + b_2 \times 2^{-2} + \dots$

$$\text{Value} \quad - \quad 16 \quad + \quad 8 \quad + \quad 4 \quad + \quad 2 \quad + \quad 1 \quad + \quad 1/2 \quad + \quad 1/4 \quad + \dots$$

Both numbers are expressed as "sign plus magnitude" and the sign in both numbers has the same total value, -1. Thus, both numbers express the same value.

NOTE

For delta values which are not shifted the full four places, the unused places are filled with the value of sign in the same manner as the partial products.

2-325. **MULTIPLICATION LOGIC CIRCUITS.** The multiplication logic circuits are activated by three instructions; multiply (MPY), multiply and hold (MPH) and divide (DIV). The use of two multiply instructions provides the option of concurrent operations in the Add-Subtract Element; the instructions are described in detail in subsequent paragraphs.

2-326. Many of the circuits used in multiplication are also used for division. Where circuit operation is common or the differences minor, the circuit is described here; where the difference is major, the circuit is described here for multiplication and again for division. The shaded blocks in the illustrations are active only during division and are discussed later.

2-327. The multiply circuit, figure 2-91, consists of three parts; the Command Circuit, Timing Circuits and the Arithmetic Registers. These circuits multiply (and divide) by an iterative process of two phase-times per cycle. Fifteen phase-times are required to complete a multiply instruction. The command circuit stores the commanded instruction and provides enabling levels to the timing circuits and arithmetic registers. The timing circuits define the two phase-time cycle and terminate the process when complete. The arithmetic registers circulate the operands and implement the multiplication formula described previously.

NOTE

Figure 2-101 is a timing diagram which shows the detailed timing of the major multiply circuits. Though the figure is not referenced directly from the circuit descriptions, its frequent use is very helpful in correlating the various parts of the multiply logic.

2-328. **Command Circuit.** The command circuit, figure 2-92, consists of two latches, HOY and VOY which store the three instructions that activate the multiply circuits. Figure 2-92 gives the states of the HOY and VOY latches for each of the instructions they store.

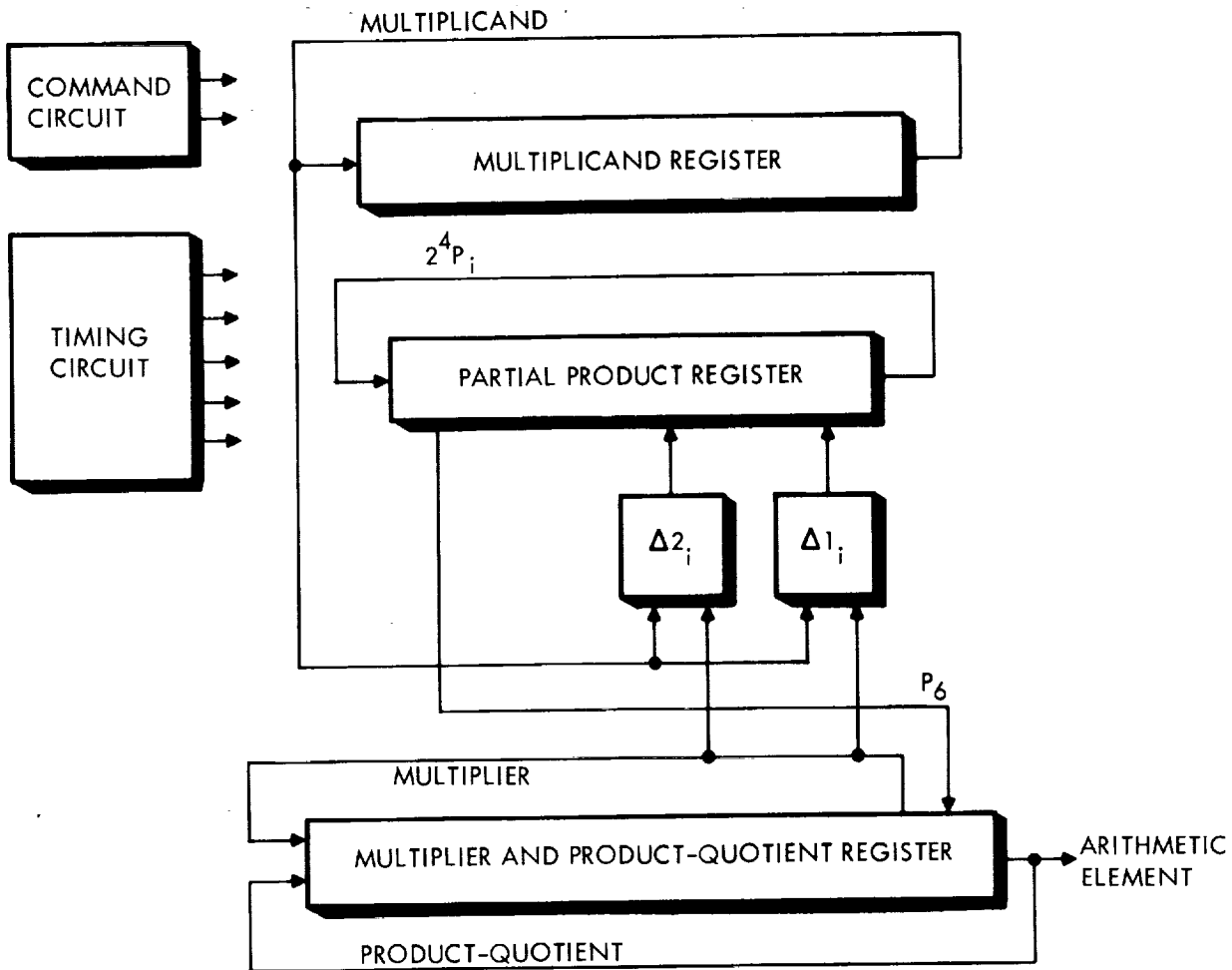
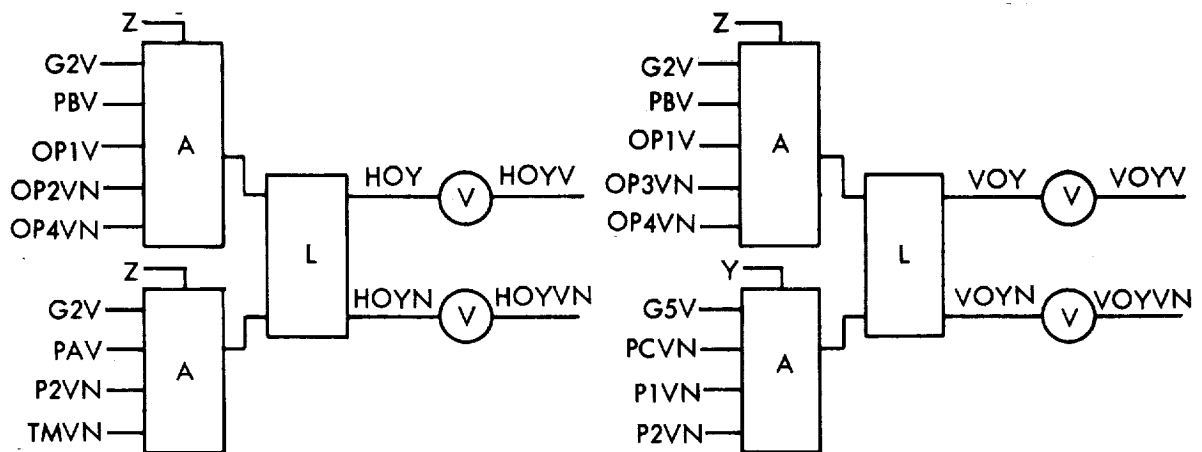


Figure 2-91. Multiply Circuit, Block Diagram

2-329. The set-input gates for the HOY and VOY latches sense the outputs of the Operation Code Register during bit-times 2 through 8 of every phase B time. Once set, these latches must not be disturbed until the commanded operation is complete. Thus, their resetting gates are controlled by the timing circuits which terminate the multiply and divide processes. The HOY latch is reset at bit-time 2-Z, and the VOY latch at bit-time 5-Y, of the first phase A following completion of the commanded operation.

2-330. Timing Circuits. The multiply and divide processes are divided into three phases. During the first phase, the operands are loaded into their respective registers and computations are started; during the second phase, the remaining computations are completed. The third phase is the period between operations during which the product (or quotient) is circulated for use elsewhere in the computer. The timing circuits consist of two latches which define the two-phase cycle upon which the multiplication and division processes are based and a phase counter. The phase counter controls the phase (or stage) of the processes and terminates them when complete.



INSTRUCTION	ABBR.	OP4	OP3	OP2	OP1	HOY	VOY
MULTIPLY	MPY	0	0	0	1	1	1
MULTIPLY AND HOLD	MPH	0	1	0	1	1	0
DIVIDE	DIV	0	0	1	1	0	1

NOTE: SEE FIGURE 10-4 (SHEET 1)
FOR DETAILED LOGIC

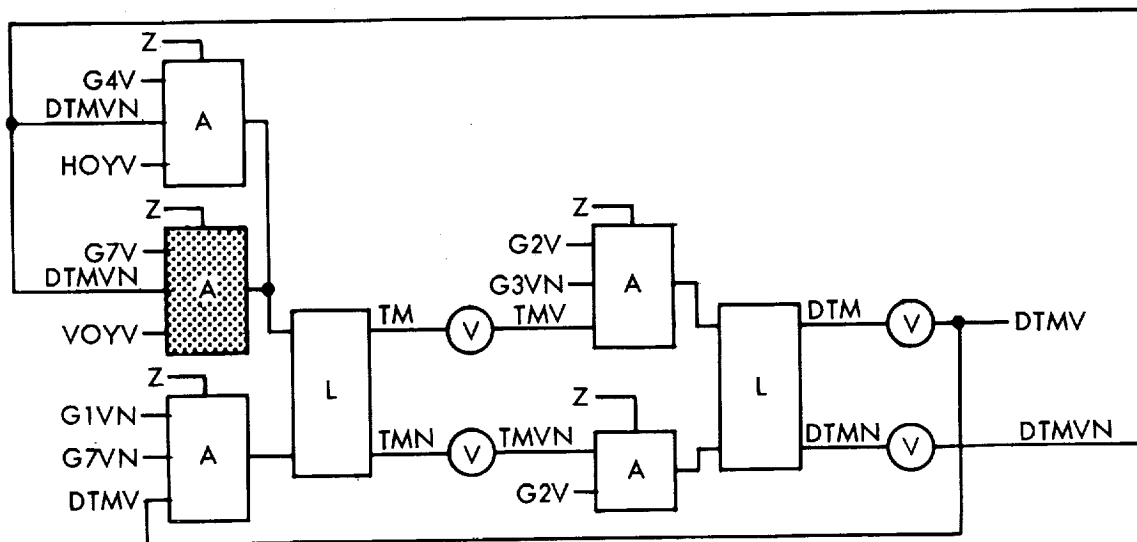
Figure 2-92. Command Circuit

2-331. The TM and DTM latches define the basic two-phase cycle used in multiplication and division. Figure 2-93 shows how these latches are connected. During a multiplication instruction, TM is a "1" from the 4-Z time near the start of the cycle until the 14-Z time at the end of the cycle; during division, TM starts at 7-Z time. DTM is a "1" during the last half of the cycle, from 2-Z time until 2-Z time, for both instructions.

2-332. The phase counter, figure 2-94, is composed of an exclusive zero tratch, channel three of the DL31 delay line and the "stop process" (STP) latch. The output of the tratch bearing the zero determines which phase of the process is underway. The tratch is set to the phase one and phase two conditions by timing signals under control of the command circuit. Phase one starts at bit-time 2-Y of the first phase B of the multiply instruction. Phase two starts two phase times later at A-1-Z time.

2-333. The tratch is also set to the phase three, or "process complete" condition by timing and the command circuit, but with the additional control of the STP latch. Phase three starts at bit-time 5-Y of phase C; the STP latch determines which phase C.

2-334. During phase one of a multiplication, gate A7 clocks a "1" into channel 3 of the DL31 delay line at B-11-Y time. The "1" emerges from the delay line fifteen and one-half bit-times later at the STP latch.



NOTE: SEE FIGURE 10-4 (SHEET 1)
FOR DETAILED LOGIC

Figure 2-93. Multiply-Divide Timing Latches

2-335. The ZN signal zero-drives the STP latch set every Z clock-time. The STP latch corrects the inversion at the input to the delay line by complementing its output. If a "1" emerges at DL31SA at W clock-time, indicating a "0" was clocked into the delay line, the STP latch is reset and then corresponds to the delay line input. (Refer to Delay Lines paragraph.)

2-336. During phase two of the multiply, the "1" inserted in delay line DL31 in phase one is recirculated through gate A9 every 16 bit-times. Since the period of circulation is two bit-times longer than a phase time, the "1" shifts two bit-times each circulation cycle. The "1" will appear at the STP latch during C-5 time only in the fifth phase C after the multiply was initiated. At that time the multiplication process is complete and is terminated by setting the phase counter to phase three, process complete.

2-337. Arithmetic Registers. The multiplication formula is instrumented by manipulating the contents of the circulating shift registers shown in figure 2-91.

2-338. The multiplicand and multiplier registers circulate the operands. The partial product register stores the partial products and its associated circuits sense the operands and from them form a new partial product according to the multiply formula each two phase-time cycle. When the final product has been formed, the multiplier register is extended to form the product-quotient (PQ) register. The PQ register stores the results of all multiply and divide operations for use elsewhere in the computer.

2-339. The multiplicand register consists of channel one of delay line DL44 and eight latches, MD0 through MD7, as shown in figure 2-95. This register circulates its contents once every two phase-times, and thus does not shift. During the first phase of a multiplication, the contents of the addressed memory location enter the MD1 latch from the TRS line. The entire 26-bit data word is transferred, but the two low-order bits are ignored throughout the multiplication process.

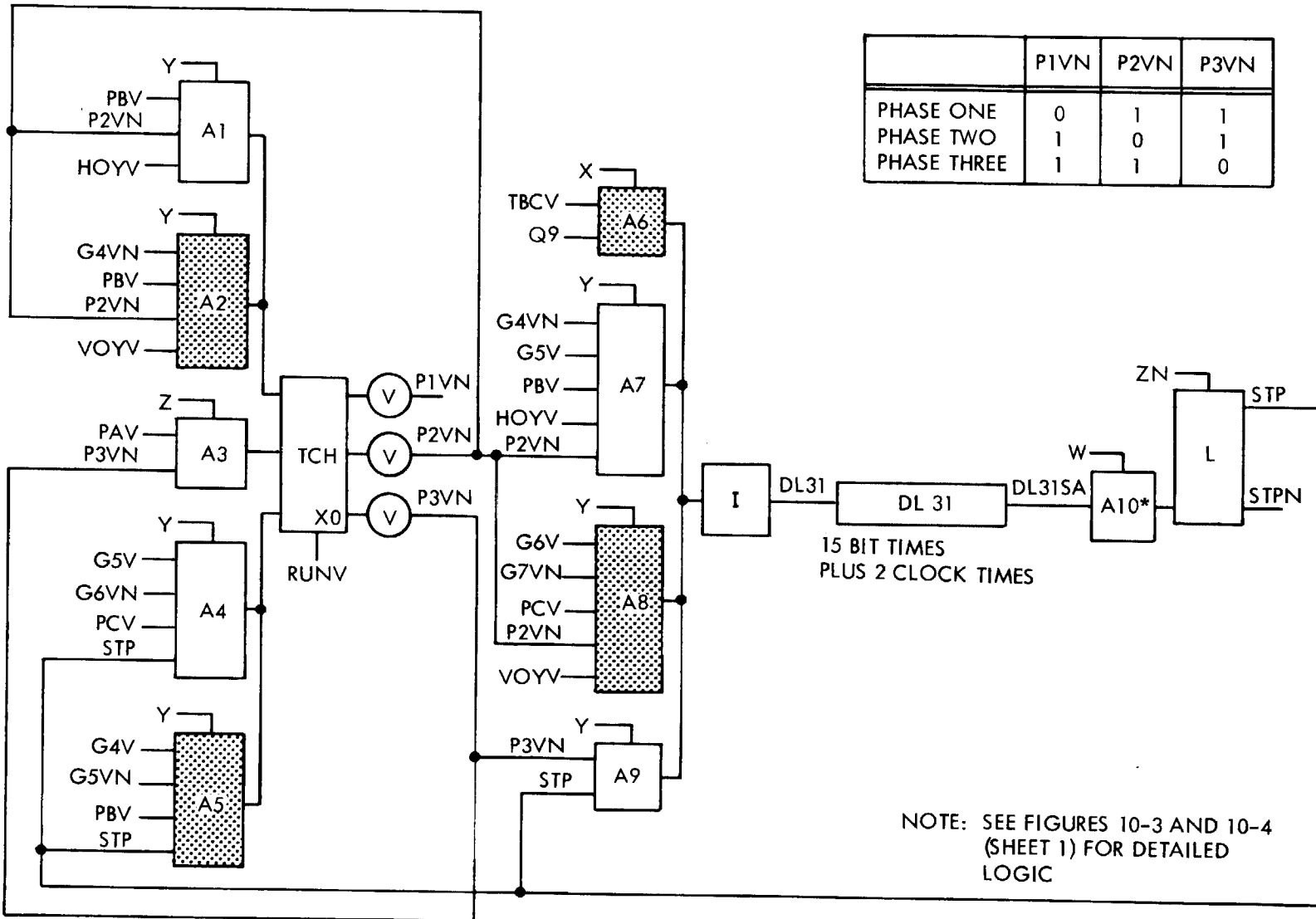


Figure 2-94. Phase Counter

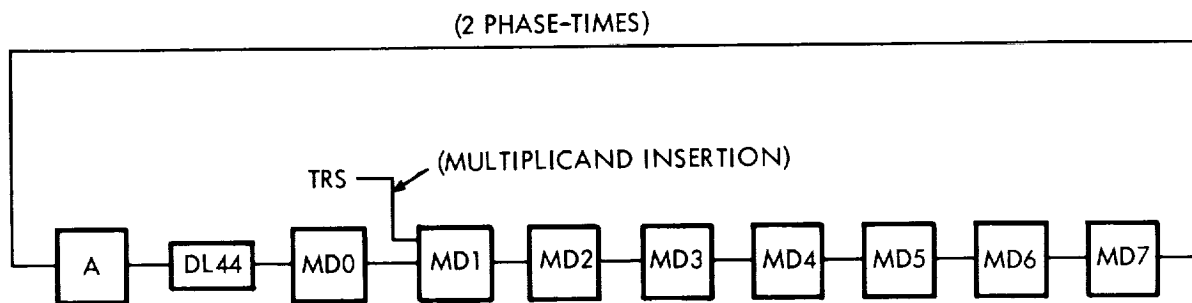


Figure 2-95. Multiplicand Register, Block Diagram

2-340. The multiplicand is entered through gates A4 and A1, figure 2-98, beginning at B-4-X time. The MD1 latch is reset by the W clock at bit-time B-4 and all following bit-times until the entire data word is entered. The latch is set a X clock-times, if TRS is a "1." At Y clock-times, the contents of the MD1 latch are transferred into the MD2 latch. The previous contents of the register enter the MD1 latch from the MD0 latch at Z clock-times. This data is destroyed in the MD1 latch when it is reset at the following W clock-time.

2-341. When the sign of the multiplicand has entered the MD1 latch, the entry gates are disabled and the register loop is closed. Gate A1 is disabled at bit-time 1-Y when TBCV goes to a "0" and gate A4 is disabled by P2VN at bit-time 1-Z. The low-order bit of the multiplicand returns to the MD1 latch at bit-time 3-Z. Thus, the previous contents of the register which entered MD1 at bit-times 1-Z and 2-Z are allowed to continue circulating. The values of these bits cannot be determined.

2-342. The multiplicand register is also used as temporary storage for the automatic HOP-constant save circuit. The HOP constant enters the multiplicand register from the NU latch via gate A9. Details of the HOP-constant save circuit are given in the description of the Program Control Element.

2-343. The multiplier register, figure 2-96, consists of channel two of delay line DL44 and latches MR0, MR1, MR2 and Q1. Note that this register contains fewer latches than the multiplicand register; therefore it shifts its contents four places toward the least significant digit (LSD) each two phase-time cycle. This shift brings four new multiplier bits into sampling position each iteration. The Q1 latch is not included in the loop of the register, but provides a delayed output of the MR2 latch to the circuits which sample the multiplier bits.

2-344. While the multiplicand is entering its register, the contents of the accumulator register are transferred into the multiplier register through the MR1 latch. This is a double line transfer from the AI2 latch through gates A6 and A9, figure 2-99, beginning at B-5-Y time. Only the 24 high-order bits are transferred, but during bit-times 3 and 4, "0"s are entered in the two low-order positions. At bit-time 3-X, the MR1 latch is reset by gate A10; it cannot be set again until bit-time 5 when TMV becomes a "1".

2-345. The "0" inserted at bit-time 3 has no significance, but the "0" from bit-time 4 is imperative for a valid multiplication. This "0" is the extra bit added to the multiplier shown in the last term of equation (6) of the multiply formula derivation; it is discussed in paragraph 2-310.

2-346. The Partial Product Register, figure 2-97, consists of channel four of delay line DL44 and latches PR, PR0, PR1 and PR2. Like the multiplier register, its capacity is 24 bits which are LSD-shifted four places each two phase-time cycle. The shift divides $2^4 P_i$ by sixteen to form P_{i-1} for the next iteration. The output of the PR0 latch is P_{i-1} .

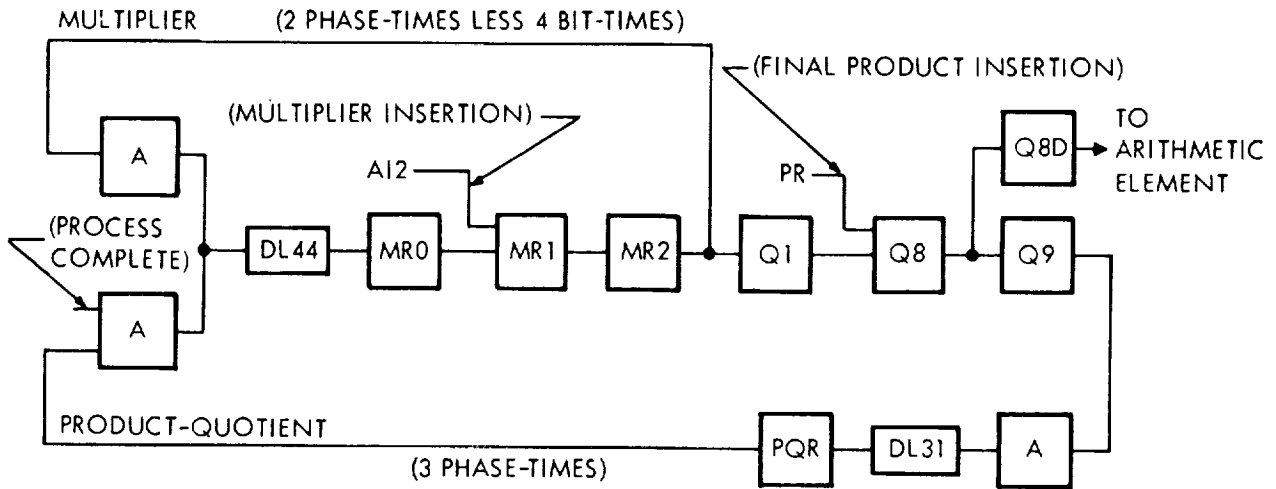


Figure 2-96. Multiplier and Product - Quotient Registers, Block Diagram

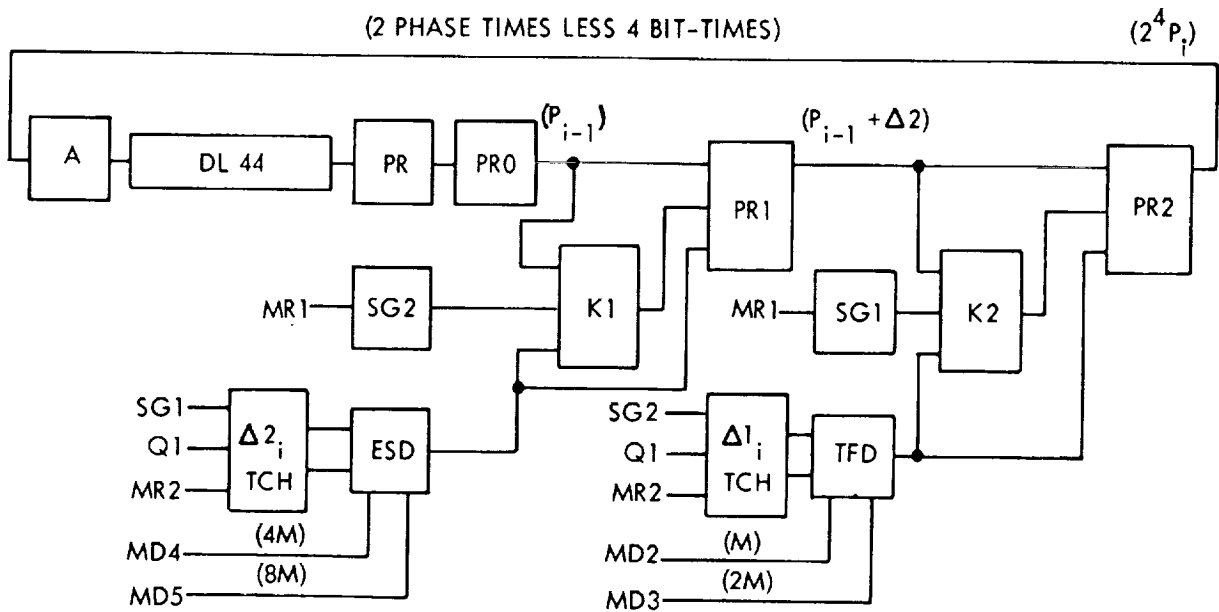


Figure 2-97. Partial Product Register, Block Diagram

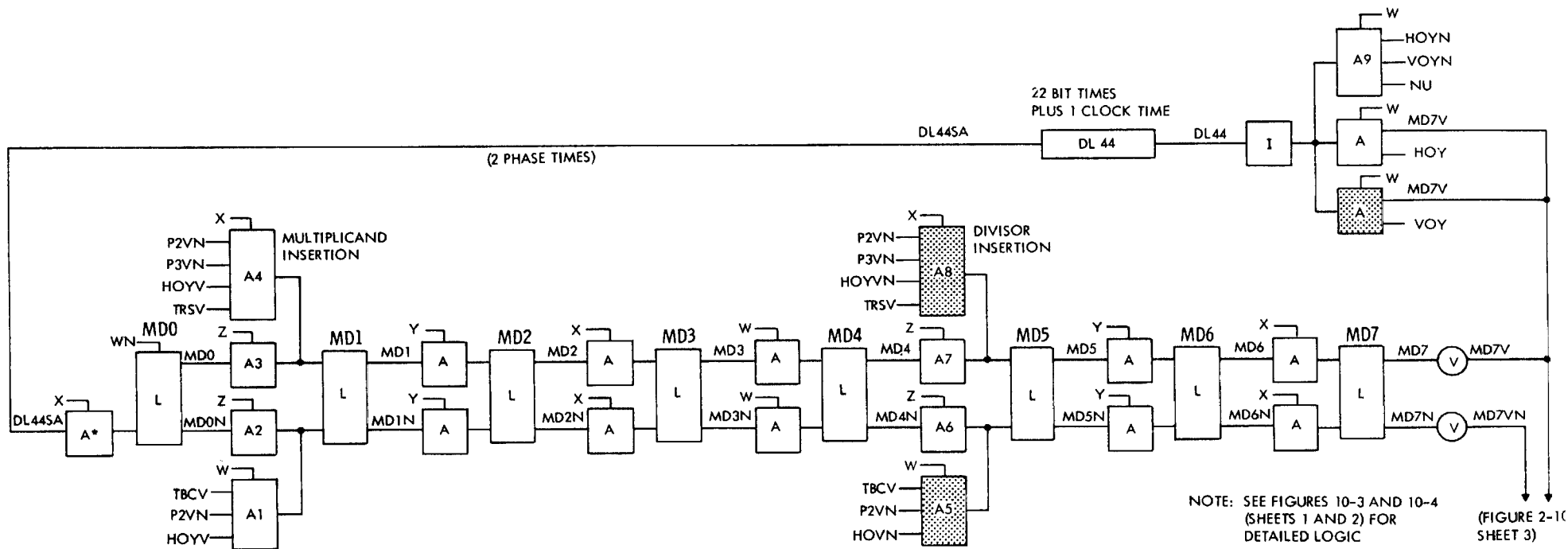


Figure 2-98. Multiplicand - Divisor Register



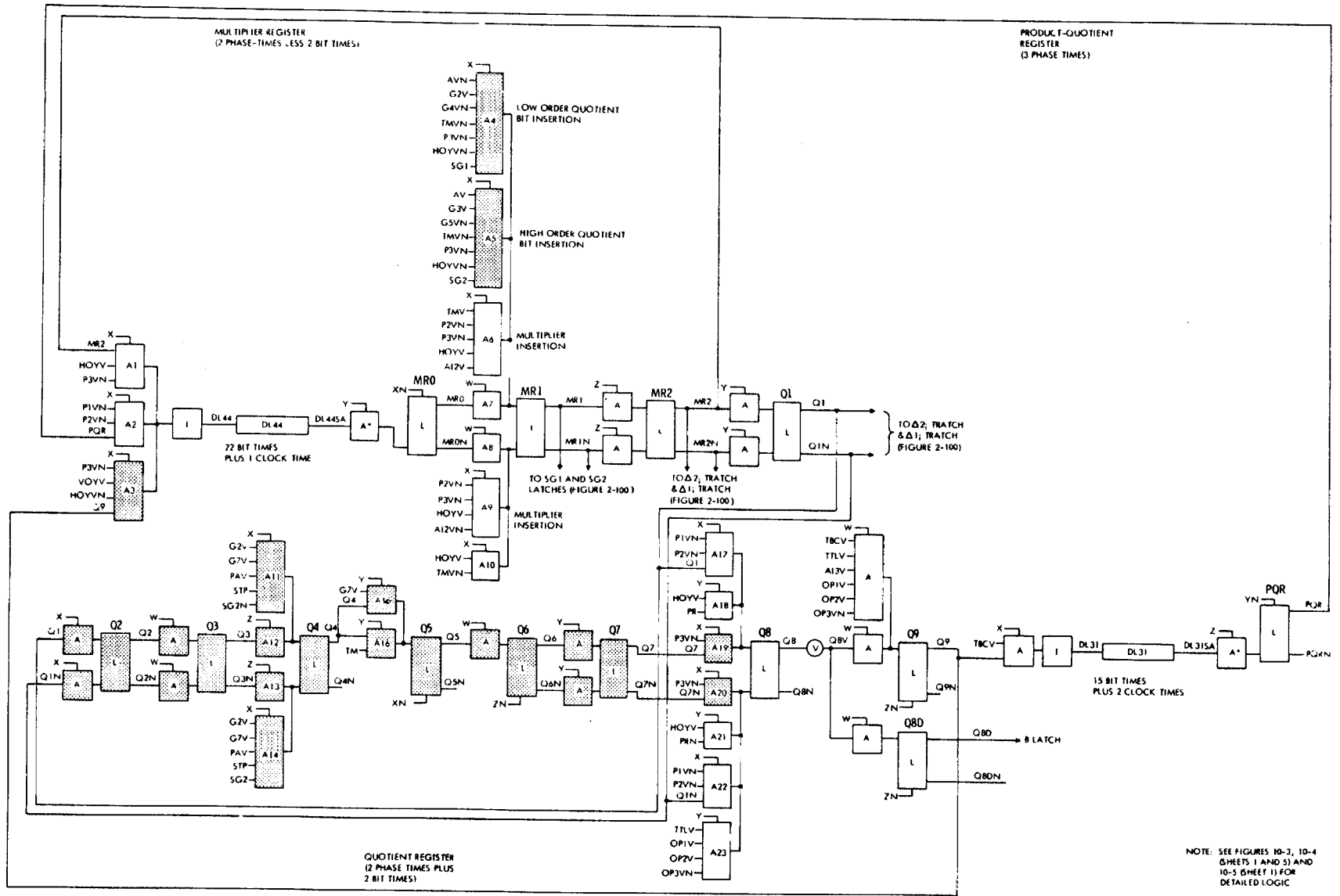


Figure 2-99. Multiplier Quotient and Product - Quotient Registers



2-347. During each iteration, the product of the 24-bit multiplicand and four new bits of the multiplier is formed at the output of the PR2 latch. The length of a binary product is the sum of the number of bits in the operands; thus, a 28-bit product must enter channel four of DL44 each cycle. Since the length of the product is equal to the length of the cycle, there are no gaps in the register. The sign bit of one product is followed immediately by the low-order bit of the next product.

2-348. Because the capacity of the partial product register is only twenty four bits, the four low-order bits of each product are truncated as they enter the PR0 latch. The double line transfer gates, A11 and A12, figure 2-100, are disabled by TMV for four bit times just after the sign bit is transferred into PR0. With the transfer gates disabled, bits 27 through 24 are ignored. The sign bit was the last bit transferred into the PR0 latch and since it is not reset, PR0 produces the value of sign for four extra bit times. These four bits account for the offset between the partial products and the delta values discussed previously.

2-349. Since the value of P_{i-1} must be zero for the first iteration of each multiplication, the partial product register is loaded with all zeros at the completion of each multiply command. When the phase counter switches to phase three, gates A63 and A64, figure 2-100, are disabled, forcing zeros into channel four of the DL44 delay line. The first zero emerges from the PR0 latch at the next A-14-Z time, well before the computation of $2^4 P_1$ could begin if the next instruction were another multiply command. Gates A63 and A64 remain disabled until another multiply or a divide command can restart the phase counter.

2-350. Circuits associated with the partial product register sense the MB bits in the multiplier register to determine the values of $\Delta 1_i$ and $\Delta 2_i$. The delta values are then gated through selector latches into adder-subtractor circuits which combine them with P_{i-1} as it circulates in the partial product register.

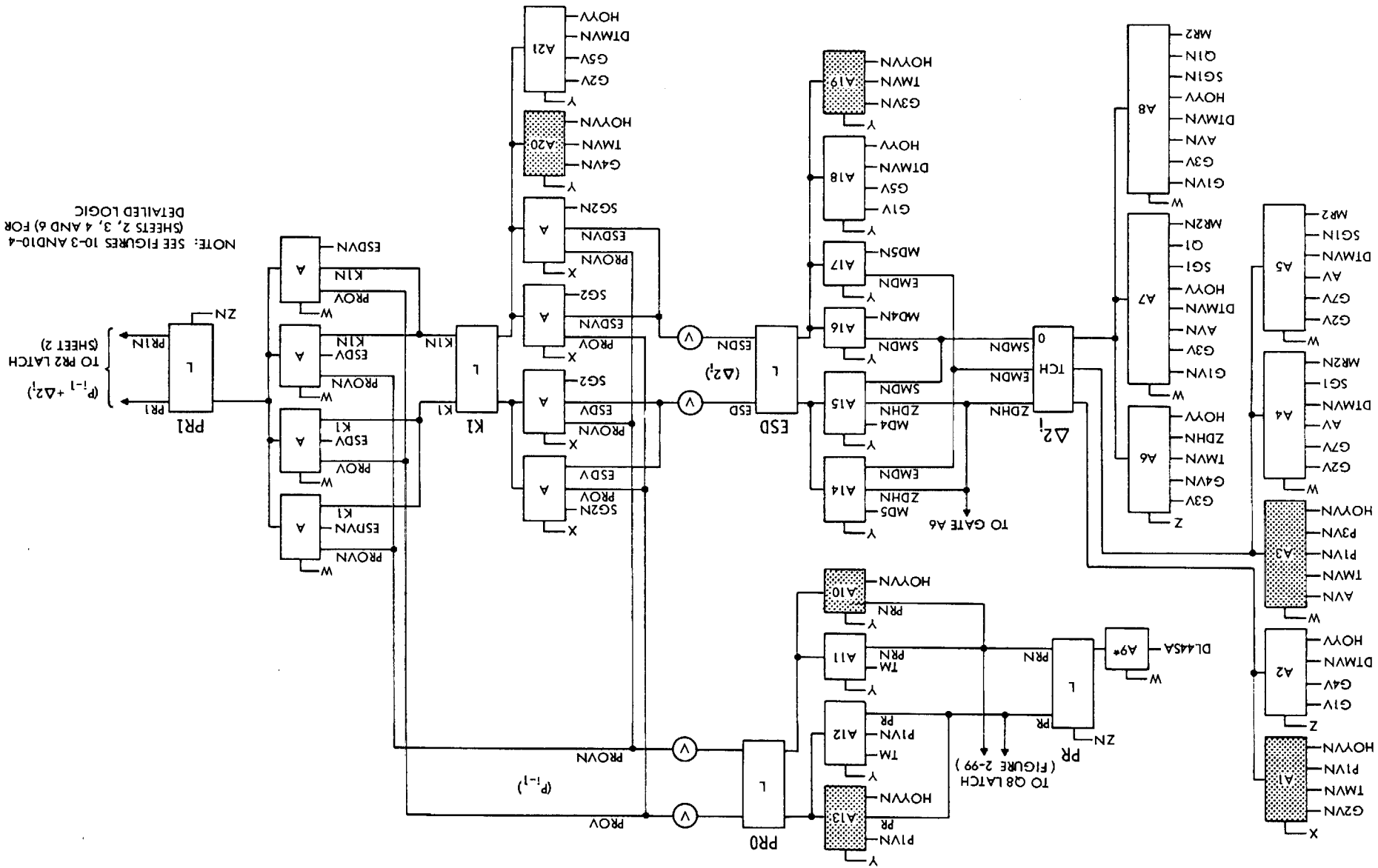
2-351. The ΔI_i tratch, figure 2-100, senses bits MB1 through MB3 and stores the absolute value of $\Delta 1_i$.

NOTE

The value of $\Delta 1_i$ is stored in inverse form as a "0" on a "not-value" line. For example: if the MB bits specify the value $2M$ for $\Delta 1_i$, the TMDN (Two times the Multiplicand Not) output of the $\Delta 1_i$ tratch is switched to a "0" and the other two lines to "1's."

At the beginning of every multiply cycle, gate A24 resets the $\Delta 1_i$ tratch to ZDLN = 0 in preparation for sensing the MB bits. At the following bit-time 6, gates A25 and A26 sense bits MB1 from the SG2 latch and MB2 from the MR2 latch (multiplier register). If MB1 and MB2 are different, the $\Delta 1_i$ tratch is set to TMDN = 0, indicating that the value of $\Delta 1_i$ is either plus or minus $2M$. See figure 2-89.

Figure 2-100. Partial Product - Remainder Register (Sheet 1 of 4)



NOTE: SEE FIGURES 10-3 AND 10-4 (SHEETS 2, 3, 4 AND 6) FOR DETAILED LOGIC

TO PR2 LATCH (SHEET 2)

TO PR1 LATCH

$(P^{-1} + \Delta_2)$

ZN

PR1

PR2

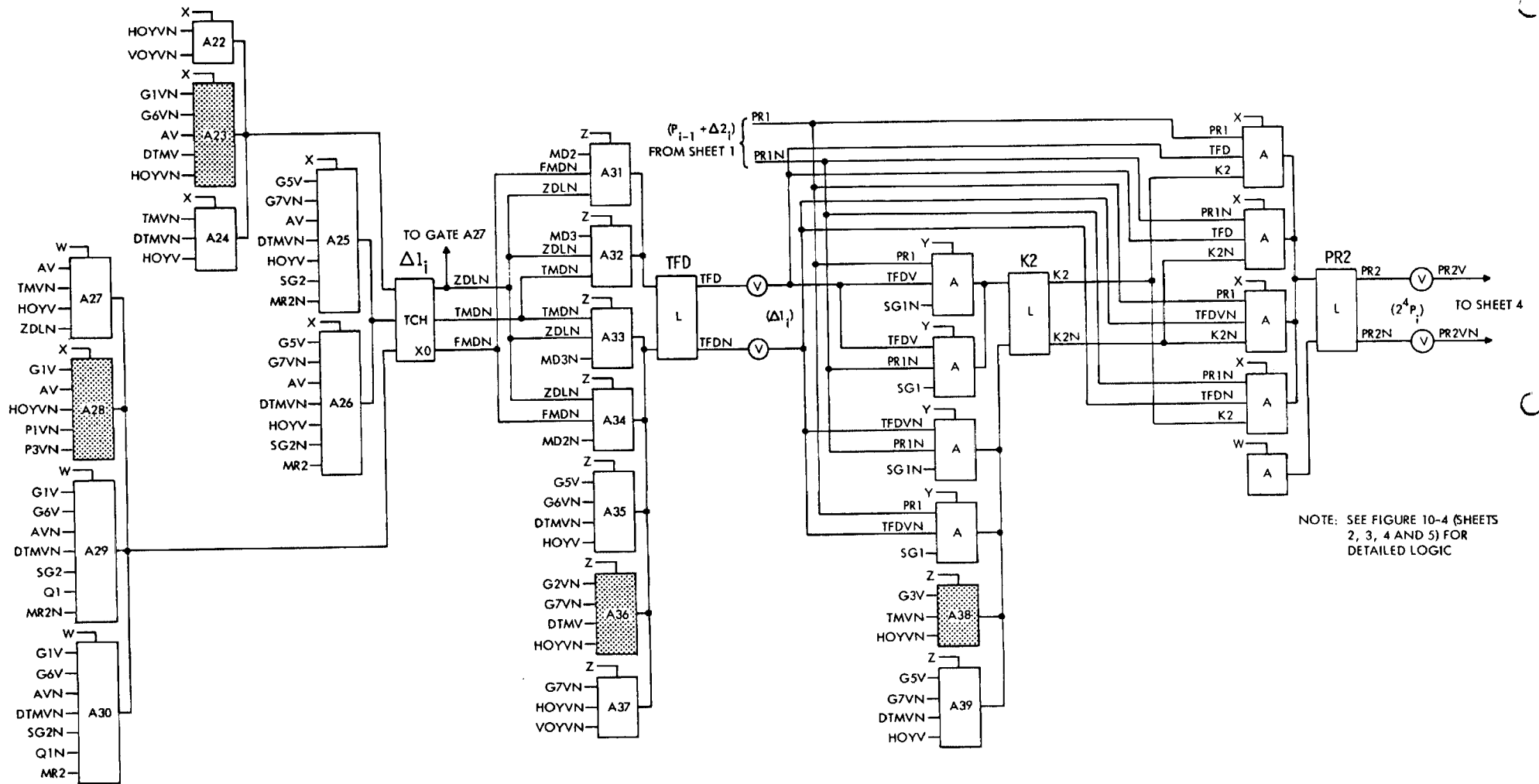


Figure 2-100. Partial Product - Remainder Register (Sheet 2)

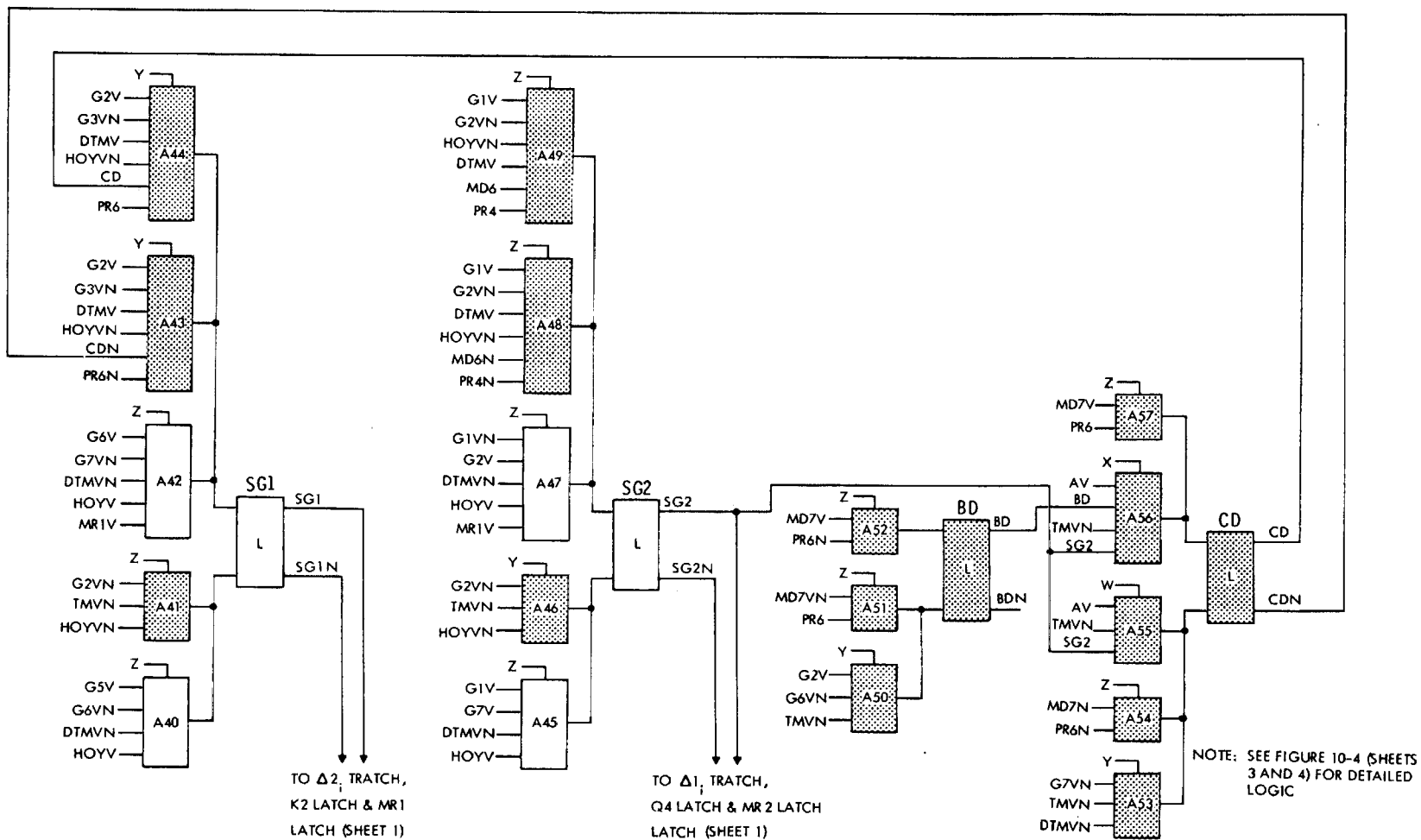


Figure 2-100. Partial Product - Remainder Register (Sheet 3)

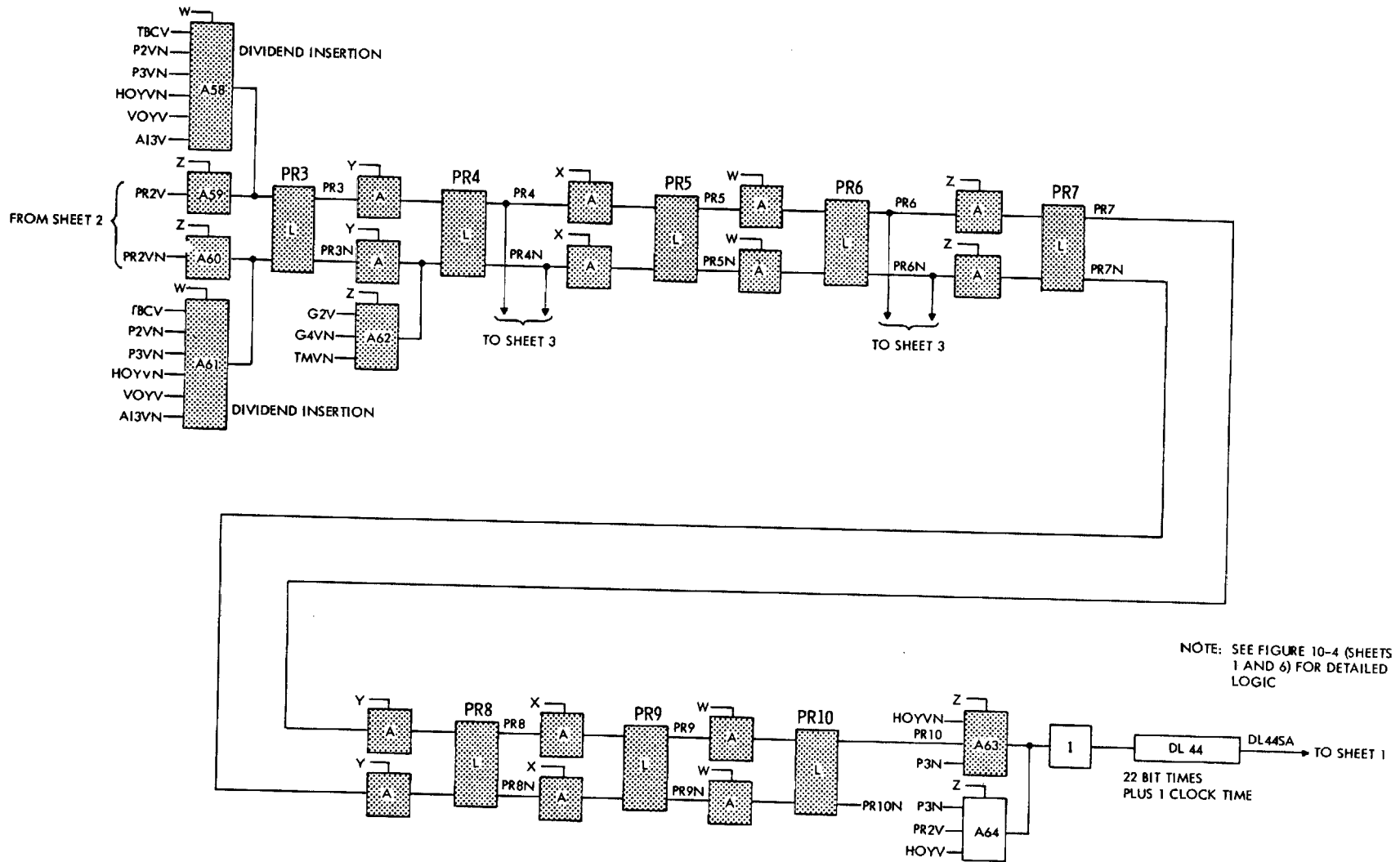


Figure 2-100. Partial Product - Remainder Register (Sheet 4)

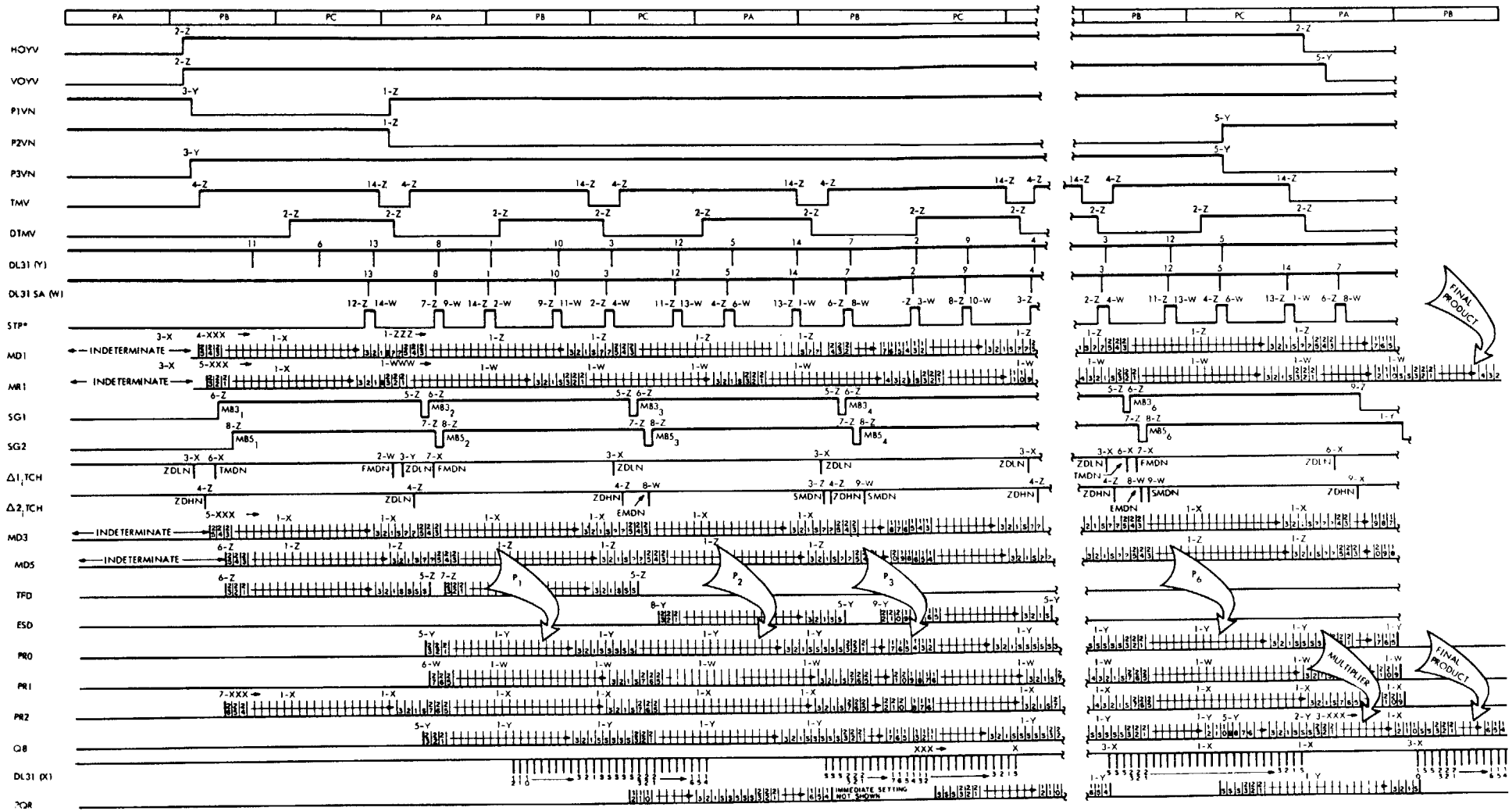


Figure 2-101. Multiply Timing Diagram



2-352. At bit-time 7, gates A29 and A30 sense all three MB-bits to determine if 4M is the $\Delta 1_i$ value specified. Bit MB1 is again sensed from the SG2 latch, but bit MB2 has since been shifted into the Q1 latch and the MR2 latch now contains bit MB3. If 4M is the value specified, the $\Delta 1_i$ tratch is set to FMDN = 0. Gate A27 is used in forcing the value of sign into vacant positions of the partial product; this operation is described in detail later. Gate A22 holds the $\Delta 1_i$ tratch at ZDLN = 0 forcing the value of $\Delta 1_i$ to zero when no multiply or divide instruction is being operated.

2-353. The $\Delta 2_i$ tratch senses bits MB3 through MB5 and stores the absolute value of $\Delta 2_i$; its operation is similar to that for the $\Delta 1_i$ tratch except for timing and the points where the MB bits are sampled. Gate A2 resets the tratch to ZDHN = 0 at bit-time 4-Z in preparation for sensing the MB bits. During bit-time 8, gates A4 and A5 sense bits MB3 and MB4 at latches SG1 and MR2 for the value 8M. At bit-time 9, gates A7 and A8 sense bits MB3 (SG1), MB4 (Q1) and MB5 (MR2) for the value 16M. Gate A6 is the counterpart of gate A27 for the $\Delta 1_i$ tratch.

2-354. The sign latches, SG1 and SG2, store the values of multiplier bits MB3 and MB5, respectively. Operation of the two latches is identical except for timing. Both latches are reset near the beginning of each multiply cycle and set to the new value one bit-time later. Since bits MB3 and MB5 determine the signs of the delta values (figure 2-89) the SG1 and SG2 latches provide sign control for the adder-subtractor circuits.

2-355. The timing of the SG2 latch has special significance. Multiplier bit MB5 of one iteration is also bit MB1 of the following iteration. The SG2 latch, which stores MB5, is not reset until bit-time 7-Z, two clock-times after its contents are sampled as MB1 at the $\Delta 1_i$ tratch. Thus, the SG2 latch stores bit MB5 during one iteration and (effectively) MB1 during the first part of the next.

2-356. The TFD and ESD latches, figure 2-100, are the selector latches through which the delta values are gated. The TFD latch produces the $\Delta 1_i$ value which can be either Two or Four times the multiplicand. The ESD latch produces the $\Delta 2_i$ value of either Eight or Sixteen times the multiplicand. The delta tratches determine which of the two values each latch will produce.

2-357. As noted previously, $\Delta 1_i$ and $\Delta 2_i$ are formed by shifting the multiplicand relative to the partial product. The shift is accomplished by allowing the multiplicand to circulate farther through its register (before it is sampled) than the partial product. The contents of the multiplicand register are sampled at four latches along its length and the outputs applied to the TFD and ESD latches. The outputs of the sampled latches (MD2, MD3, MD4 and MD5) are the multiplicand shifted 0, 1, 2 and 3 places, and correspond to the values M, 2M, 4M and 8M, respectively. The TFD and ESD latches shift the values one more place to form the correct delta values.

2-358. The outputs from latches MD2 and MD3 are applied to the TFD latch; outputs from the MD4 and MD5 latches are fed to the ESD latch inputs. The $\Delta 1_i$ and $\Delta 2_i$ tratches select which input is gated through each latch according to the delta values. For example: the MD5 (8M) inputs cause the ESD latch to produce 16M at its output. Thus, the MD5 inputs are enabled by $\Delta 2_i$ tratch outputs ZDHN and EMDN both being "1's" which indicates that the SMDN (16M not) output is a "0" (delta values are stored in inverse form).

2-359. Gate A18, figure 2-100, resets the ESD latch at bit-times 5 through 7 before the $\Delta 2_i$ value is stored in its latch. The ESD latch cannot then be set until bit-time 8, just before the first bit of the multiplier can appear at its inputs. Gate A35 resets the TFD latch at bit-time 5, which results in the same timing considerations for the $\Delta 1_i$ circuits. Gate A37 resets the TFD latch when the multiply process is complete; gate A19 resets the ESD latch after completion.

2-360. The outputs of the TFD and ESD latches are fed to the adder-subtractor circuits which combine the delta values with the previous partial product. The K1 and PR1 latches form one adder-subtractor which computes

$$P_{i-1} + \Delta 2_i$$

The K2 and PR2 latches form another adder-subtractor which computes

$$(P_{i-1} + \Delta 2_i) + \Delta 1_i$$

The output of the PR2 latch is $2^4 P_i$.

2-361. The PR1 and PR2 latches are basic serial binary adders; their four input gates (figure 2-100) correspond to the add-subtract logic in the Arithmetic Element. The K1 and K2 latches store the carry or borrow from one bit to the next; the signs of the delta values determine which conditions K1 and K2 will store for each computation cycle. The SG2 and SG1 latches store these signs and control K1 and K2 respectively.

2-362. The PR1 latch is reset at every Z clock-time by the ZN zero-drive signal. At the following W clock-time, PR1 is set if the sum (or difference) bit is a "1". At X clock-time (after the sum has been stored) the K1 latch is set to the carry or borrow condition for the next bit. The K1 latch is reset during bit-times 5 through 8 (gate A21) at the beginning of every cycle to assure that no carry or borrow is propagated from one iteration to the next. Bit-times 5 through 8 coincide with the reset time for the ESD latch. Thus, the K1 latch does not sense carry or borrow when one of the operands is not available.

2-363. Operation of the PR1 and PR2 latches is identical except for timing and both latches add continuously. When no addition is required, the K1 and K2 latches are reset and the delta values are forced to zero. The K1 and K2 latches also operate alike except for reset timing. The K2 latch reset timing coincides with that for the TFD latch. For a discussion of the gates which determine the sum and the carry or borrow, refer to the Arithmetic Element.

2-364. When the multiply process is complete, the Product-Quotient (PQ) Register, figure 2-96, is formed by extending the multiplier register to include channel two of delay line DL31 and latches Q8, Q8D, Q9 and PQR. This register stores the final product and circulates it every three phase times for use elsewhere in the computer. Once loaded, its contents can be changed only a store (STO) instruction addressing the PQ register, by a divide instruction (DIV) or by another multiply instruction. The PQ register is assigned memory address 775 and can be addressed with all the arithmetic instructions except MPY, MPH and DIV.

2-365. All partial products enter the Q8 latch through gates A18 and A21, figure 2-99, from the PR latch in the partial product register. However, due to the timing of TBCV at gate A15 only partial products P3 and P6 ever reach the PQR latch intact. No data can circulate beyond the PQR latch until phase three of the multiply process.

NOTE

Partial product P_3 can be used by addressing the PQ register with the second instruction following an MPY. The result is the product of the multiplicand and the twelve LSD bits of the multiplier. Sampling P_3 does not affect the operation of the multiply circuit and the final product will be computed as usual.

2-366. The phase counter switches to the phase three condition just as product P_6 , the final product, begins entering the PQR latch. Gates A2, A17 and A22 are enabled at this time and the loop of the PQ register is closed. Simultaneously, gate A1 opens the loop of the multiplier register. Gates A18 and A21 continue entering product P_6 into the PQ register until the command circuit is reset, signaling the end of the multiply operation.

2-367. The Q8D latch provides a delayed output from the PQ register for use in the Arithmetic Element. The input gates to the B latch (in the Arithmetic Element) synchronize the Q8D output with the other data sources in the computer. Corresponding bits of data from Q8D, TRS and IOREG are available to the B latch during the same bit time.

2-368. As noted earlier, data can be loaded into the PQ register by addressing memory location 775 with an STO instruction. This operation transfers the accumulator contents into the Q9 latch. The transfer is affected by gates A23 and A24, figure 2-99. The accumulator contents enter the Q9 latch through gate A24 on the AI3 line. Gate A23 blocks the previous contents of the register by holding the Q8 latch reset until the transfer is complete. Both gates bear the operation codes for STO and DIV and the TTLV signal. The TTLV signal is a "1" when memory location 775 is selected.

2-369. Computation Cycle. The math flow of a multiply computation cycle is shown in figure 2-102. The $\Delta 1_i$ and $\Delta 2_i$ tratches are initialized to show delta values of zero in preparation for sensing the MB bits. The SG1 latch is reset before sensing the sign of $\Delta 1_i$ and the TFD latch is reset to assure a starting condition of "0".

2-370. The $\Delta 1_i$ tratch then senses bits MB1 and MB2 to determine if the magnitude of $\Delta 1_i$ is 2M. If 2M is the specified value, the $\Delta 1_i$ tratch is set to gate the MD2 output through the TFD latch. The SG1 latch then senses MR1 for the sign of $\Delta 1_i$. If MR1 is a "1", the sign of $\Delta 1_i$ is negative and the SG1 latch is set. The SG1 latch then conditions the K2 latch to borrow for the duration of the cycle. If MR1 is a "0", the SG1 latch remains reset, indicating a positive delta value, and the K2 latch is conditioned to carry.

2-371. If the value of $\Delta 1_i$ had not been 2M, the sign of $\Delta 1_i$ would have been sensed immediately. The $\Delta 1_i$ tratch then senses bits MB1, MB2 and MB3 to determine if the value of $\Delta 1_i$ is 4M. At this point in the flow, the value of $\Delta 1_i$ must be either 4M or 0. If MB1, MB2 and MB3 are all alike, the value is zero and the $\Delta 1_i$ tratch remains reset. The TFD latch also remains reset, producing a delta value of zero, and the multiply circuit begins sensing the sign and magnitude of $\Delta 2_i$. If the MB3 bit is not the same as MB1 and MB2, the value of $\Delta 1_i$ is 4M and the $\Delta 1_i$ tratch is set to gate the MD3 output through the TFD latch.

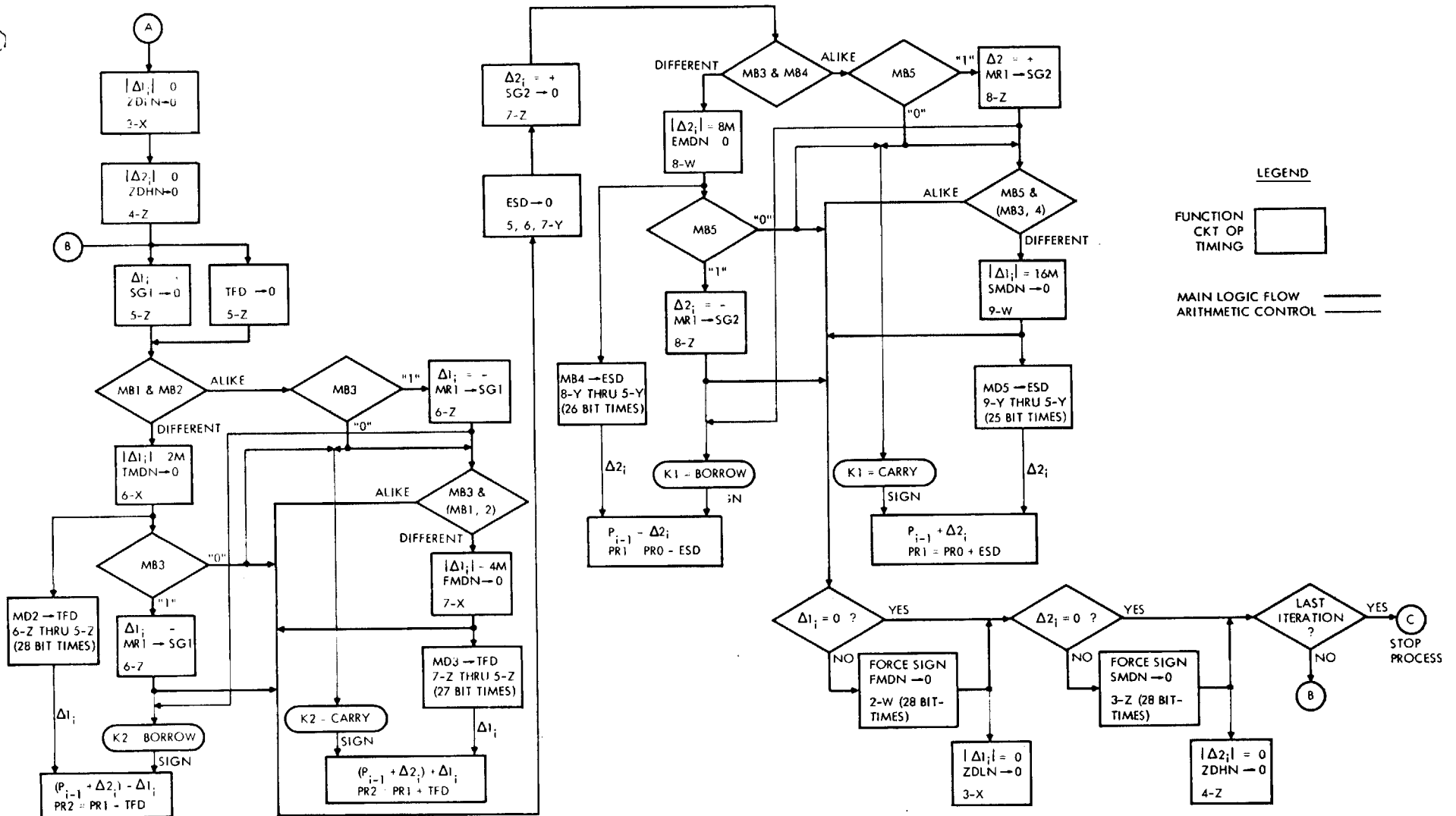


Figure 2-102. Multiply Computation Cycle from Flow Diagram

2-372. The ESD and SG2 latches are then reset to complete initialization of the Δ_2 circuits. Note that the SG2 latch is not reset until after the Δ_1 magnitude control has been set up. The MB1 bit used to determine the value of Δ_1 is actually the MB5 bit stored in SG2 during the previous iteration. This is the overlap between iterations which allows the multiply process to ignore the signs of the operands.

2-373. The Δ_2 tratch then senses whether multiplier bits MB3 and MB4 are alike. If the bits are alike, the magnitude of Δ_2 is either 16M or 0. Bit MB5 is then sensed at the MR1 latch and stored in SG2. Bit MB5 is the sign of Δ_2 and the SG2 latch conditions the K1 latch to carry or borrow depending on whether the sign is plus or minus.

2-374. Bits MB3 and MB4 are then sensed with bit MB5 to determine whether the magnitude of Δ_2 is 0 or 16M. If MB5 is different than MB3 and MB4, the magnitude of Δ_2 is 16M and the Δ_2 tratch is set to gate the MD5 output through the ESD latch. If bits MB3 and MB4 had been different when sensed alone, the magnitude of Δ_2 would be 8M. When 8M is the value specified, the Δ_2 tratch is set to gate MD4 through the ESD latch before the SG2 latch determines the sign.

2-375. It was noted earlier that Δ_2 is summed with the previous partial product before Δ_1 . An apparent paradox arises when this fact is considered with the fact that the value of Δ_1 is sensed first and begins emerging from TFD before Δ_2 is available from ESD. The solution to the paradox lies in the distances by which the delta values are offset from partial products.

2-376. The Δ_1 value is offset by either 1 or 2 places; the Δ_2 value is offset by either 3 or 4 places. Translated into machine operation, the offset places become delays of corresponding numbers of bit times. Thus, though the partial product passes through the Δ_2 summing circuit first, meaningful data is available for Δ_1 before it is available for Δ_2 . Bit-for-bit, Δ_2 is summed with the partial product first though Δ_1 data enters the summation earlier.

2-377. Once the magnitude and sign control has been set up, the adder-subtractor circuits (K1-PR1 and K2-PR2) compute the sums indicated by the multiply formula automatically until near the end of the cycle. At the second bit-time 2-W, the magnitude of Δ_1 is sensed for non-zero. If the value is not zero, the Δ_1 tratch is forced to 4M.

2-378. If the specified value of Δ_1 was 2M, the tratch is forced just after the sign of the multiplicand is transferred from MD2 to TFD. Forcing the tratch switches the TFD inputs from MD2 to MD3 and at the next bit time, the sign is transferred to TFD again. At bit 3-Y time, the Δ_1 tratch is reset, deconditioning all the data gates to the TFD latch. Since the TFD latch contains the sign and is not reset until the end of bit time 5, it stores the value of sign for an additional two bit times. Thus, the unused bit positions more significant than sign are filled with the value of sign. At bit-time 3-Z, the Δ_2 tratch is sensed for non-zero to force sign into any unused bit positions of Δ_2 .

2-379. DIVISION. The Multiply-Divide Element implements a modified form of non-restoring division in which two quotient bits are formed each two phase-time cycle. The resulting 24-bit quotient is available to the eighth and subsequent instructions following DIV by addressing the PQ register with an arithmetic instruction. Division is limited to problems in which the divisor is larger than the dividend so that the quotient will not



exceed the capacity of its storage register. The description of the division process begins with an introduction to both the restoring and the nonrestoring division techniques. A thorough understanding of basic nonrestoring division is an essential precept of the division process implemented in the computer.

2-380. Restoring Division. Figure 2-103 illustrates a binary division problem and its solution by the restoring divide technique. The first operation is to subtract the divisor (D) from the dividend (denoted as remainder R_1). This subtraction is unsuccessful since it changes the sign of the remainder (R_1 is positive, R_2 is negative). Since the divisor cannot be subtracted from the remainder, the remainder is not divisible by the divisor; thus, the first quotient bit (Q_1) is a "0". Because the correct remainder was destroyed by the subtraction, it must be restored (hence the name of the process) by adding back the divisor to form R_3 ($R_3 = R_1$). Remainder R_3 is then shifted one place toward the MSD and another subtraction is performed. This subtraction is successful as evidenced by the positive remainder; thus the second quotient bit (Q_2) is a "1".

2-381. Note that WHEN THE SIGN OF THE DIVISOR AND THE SIGN OF THE REMAINDER ARE ALIKE, THE QUOTIENT BIT IS A "1". This relation is true regardless of the signs of the dividend and divisor.

2-382. Remainder R_4 need not be restored since the subtraction which formed it was successful. Thus, it is merely shifted one place toward the MSD preparatory to the next subtraction. The divisor is then subtracted from $2R_4$, but the subtraction is unsuccessful since the sign of the result is unlike the sign of the divisor. Thus, the third quotient bit (Q_3) is a "0".

2-383. Before another subtraction can be performed, the previous remainder must be restored by adding the divisor to the present remainder, forming R_6 . Remainder R_6 is then shifted one place toward the MSD and the divisor is subtracted from the result. The remainder formed by this subtraction (R_7) is zero and since its sign and the sign of the divisor are alike, a "1" is placed in the fourth quotient bit.

2-384. The divide process can be continued to obtain as many quotient bits as are desired. In the sample problem, any additional quotient bits will all be "0's" because the remainder, having gone to zero, will remain unchanged. Subtraction from zero gives a negative remainder (and a "0" quotient bit) which must be restored, resulting in another zero remainder for the next subtraction, etc.

2-385. Nonrestoring Division. Nonrestoring division eliminates the need for an extra operation to restore the remainder when an unsuccessful subtraction occurs. This feature shortens the division process considerably, as shown by figure 2-104. Restoring and nonrestoring division are identical until an unsuccessful subtraction occurs, such as the first subtraction in the sample problem. Restoring division performs the following steps after an unsuccessful subtraction to obtain the next remainder sensed to determine a quotient bit:

- | | |
|-------------------------|------------------------|
| 1) Add the divisor | $R_2 + D = R_3$ |
| 2) Shift the result MSD | $2(R_2 + D) = 2R_3$ |
| 3) Subtract the divisor | $2(R_2 + D) - D = R_4$ |

Problem:

$$25 \times 2^{-6} \div 5 \times 2^{-3} = Q$$

$$\text{Divisor (D)} = 0 . 1 0 1 \quad (5 \times 2^{-3})$$

$$\text{Dividend (R}_1) = 0 . 0 1 1 0 0 1 \quad (25 \times 2^{-6})$$

Solution:

		Q ₁	Q ₂	Q ₃	Q ₄		
		0	. 1	0	1	Quotient (Q) = 5 × 2 ⁻³	
D	<u>0 . 1 0 1</u> /	0	. 0	1	1	0 0 1	R ₁
		0	. 1	0	1		Subtract D
R ₂		1	. 1	1	0	0 0 1	Negative remainder, Q ₁ = 0
		0	. 1	0	1		Add D to restore R ₁
R ₃		0	. 0	1	1	0 0 1	R ₃ = R ₁
2R ₃		0	. 1	1	0	0 1 0	Shift R ₃ MSD
		0	. 1	0	1		Subtract D
R ₄		0	. 0	0	1	0 1 0	Positive remainder, Q ₂ = 1
2R ₄		0	. 0	1	0	1 0 0	Shift R ₄ MSD
		0	. 1	0	1		Subtract D
R ₅		1	. 1	0	1	1 0 0	Negative remainder, Q ₃ = 0
		0	. 1	0	1		Add D to restore 2R ₄
R ₆		0	. 0	1	0	1 0 0	R ₆ = 2R ₄
2R ₆		0	. 1	0	1	0 0 0	Shift R ₆ MSD
		0	. 1	0	1		Subtract D
R ₇		0	. 0	0	0	0 0 0	Positive remainder, Q ₄ = 1

Figure 2-103. Sample Restoring Divide Problem

Problem:

$$25 \times 2^{-6} \div 5 \times 2^{-3} = Q$$

$$\text{Divisor (D)} = 0 . 1 0 1 \quad (5 \times 2^{-3})$$

$$\text{Dividend (R}_1) = 0 . 0 1 1 0 0 1 \quad (25 \times 2^{-6})$$

Solution:

		Q ₁	Q ₂	Q ₃	Q ₄	
		0	. 1	0	1	Quotient (Q) = 5 × 2 ⁻³
D	0 . 1 0 1	<hr/>				R ₁
		0	. 1	0	1	Subtract D
R ₂		1	. 1	1	0 0 0 1	Signs differ, Q ₁ = 0
2R ₂		1	. 1	0	0 0 1 0	Shift R ₂ MSD
		0	. 1	0	1	Add D
R ₃		0	. 0	0	1 0 1 0	Signs alike, Q ₂ = 1
2R ₃		0	. 0	1	0 1 0 0	Shift R ₃ MSD
		0	. 1	0	1	Subtract D
R ₄		1	. 1	0	1 1 0 0	Signs differ, Q ₃ = 0
2R ₄		1	. 0	1	1 0 0 0	Shift R ₄ MSD
		0	. 1	0	1	Add D
R ₅		0	. 0	0	0 0 0 0	Signs alike, Q ₄ = 1

Figure 2-104. Sample Nonrestoring Divide Problem

By multiplying and combining terms, this remainder can be represented by the simpler expression:

$$2R_2 + D$$

Restoring division forms the same remainder in only two steps:

- 1) Shift the present remainder $2R_2$
- 2) Add the divisor $2R_2 + D = R_3$

Remainder R₃ of the nonrestoring divide sample is equivalent to R₄ of the sample restoring divide.

2-386. Thus the procedure for nonrestoring division can be stated as follows:

- 1) Subtract the divisor (D) from the dividend (R_1).
- 2) Compare the divisor sign with the remainder sign and make the quotient bit a "1" for like signs, or a "0" if the signs are different.
- 3) Shift the remainder one place toward the MSD.
- 4) If the previous quotient bit was a "1", subtract the divisor from the shifted remainder or, if the quotient bit was a "0", add the divisor.
- 5) Repeat step 2 to determine the next quotient bit and steps 3 and 4 to form a new remainder until the desired number of quotient bits is obtained.

2-387. Nonrestoring division can be further condensed by shortcutting the determination of the quotient sign. Since the sign of the quotient is dependent only on the signs of the divisor and dividend, it can be determined by inspection and no shift or subtraction is prerequisite. It is a curious fact that the rules for determining the sign of the quotient are the inverse of the rules for determining the value of a magnitude bit. If the signs of the divisor and dividend are alike, a "0" should be placed in the quotient sign to indicate a positive result. If the divisor sign matches that of a remainder later in the process, a "1" is placed in the quotient. However, a divide circuit can use the latter relationship to form all quotient bits, including sign, and then complement the sign at a later time to correct it. Correcting the sign creates no special problems in division because the sign is the first bit formed and is not altered by subsequent computations as is the case in multiplication. The problem presented in previous samples is solved again in figure 2-105 using the sign shortcut.

2-388. Note that the first subtraction given in the rules for nonrestoring division is eliminated by the sign shortcut. Division then becomes essentially a two-step process:

- 1) Compare the divisor and present remainder signs to determine the quotient bit.
- 2) Form the next remainder by shifting the present remainder and either adding or subtracting the divisor.

This is the division technique upon which the computer's division process is based.

2-389. Computer Division. In computer division two quotient bits are formed each iteration. The first quotient bit is determined from the divisor and present remainder signs, as in the shortcutted nonrestoring process. The second quotient bit is formed by predicting the sign of the next remainder in advance and comparing it, by a unique method, with the divisor sign. The divide circuit then forms the next two remainders almost simultaneously. Although a quotient bit has already been based on the predicted value of the "next remainder" sign, that remainder is generated as an intermediate step in the development of the "second remainder". The second remainder then becomes the "present remainder" for the following iteration during which two more quotient bits are formed.

Problem:

$$25 \times 2^{-6} \div 5 \times 2^{-3} = Q$$

$$\text{Divisor (D)} = 0 . 1 0 1 \quad (5 \times 2^{-3})$$

$$\text{Dividend (R}_1) = 0 . 0 1 1 0 0 1 \quad (25 \times 2^{-6})$$

Solution:

		Q ₁	Q ₂	Q ₃	Q ₄		
Sign Correction:	→	0	1	0	1		
D		0	1	0	1	0 0 1	R ₁ , signs alike, Q ₁ = 1
2R ₁		0	1	1	0	0 1 0	Shift R ₁ MSD
		0	1	0	1		Subtract D
R ₂		0	0	0	1	0 1 0	Signs alike, Q ₂ = 1
2R ₂		0	0	1	0	1 0 0	Shift R ₂ MSD
		0	1	0	1		Subtract D
R ₃		1	1	0	1	1 0 0	Signs differ, Q ₃ = 0
2R ₃		1	0	1	1	0 0 0	Shift R ₃ MSD
		0	1	0	1		Add D
R ₄		0	0	0	0	0 0 0	Signs alike, Q ₄ = 1
							Complement sign

Figure 2-105. Nonrestoring Divide with Sign Shortcut

2-390. As previously noted, the "next remainder" is formed by shifting the present remainder one place toward the MSD and then either adding or subtracting the divisor. The sign of the next remainder is determined by the summation of the signs of the divisor and the shifted present remainder with the carry (or borrow if subtracting) into the sign position. The value of the sign bit can then be accurately predicted if the following information is available:

- 1) Sign of divisor
- 2) Sign of shifted present remainder
- 3) Value of carry into sign position, if adding
- 4) Value of borrow into sign position, if subtracting
- 5) Whether divisor will be added or subtracted.

The truth table for summation of the two signs with the carry or borrow is shown in figure 2-106, expanded to include the second quotient bit. The summation is the same for both the carry and borrow, thus one truth table shows the sign bit developed by either addition or subtraction of the divisor.

Sign of Shifted Remainder	Sign of Divisor	Carry or Borrow	Sign of Next Remainder	Second Quotient Bit
<u>0</u>	0	0	0	<u>1</u>
0	0	1	1	0
<u>0</u>	1	<u>0</u>	1	<u>1</u>
0	1	1	0	0
1	0	0	1	0
<u>1</u>	0	<u>1</u>	0	<u>1</u>
1	1	0	0	0
<u>1</u>	1	<u>1</u>	1	<u>1</u>

Figure 2-106. Truth Table for Second Quotient Bit

2-391. As shown in the truth table, the second quotient bit is a "1" whenever the sign of the divisor matches the predicted sign of the next remainder. This agrees with the rules for nonrestoring division presented earlier. Closer examination of the truth table reveals that in each case where the quotient bit is a "1", the sign of the shifted remainder and the carry or borrow condition also match. Thus the second quotient bit can be determined from these factors and predicting the sign of the next remainder need not be done. This is the "unique method" by which the divisor sign is compared with the "predicted next remainder sign". Although the prediction is not actually made, it is inherent in the information used to form the second quotient bit.

2-392. In terms of basic machine operation, the computer divides in the following manner (see figure 2-107). As the divisor and first remainder (dividend) enter their respective registers, they are monitored by two latches (Carry - Borrow). One latch keeps track of the carry function as though the divisor were being added to the shifted remainder. The other latch keeps track of the borrow as though the divisor were being subtracted from the shifted remainder. Since the remainder has not yet been shifted, the 2^{-1} bit is monitored by the carry and borrow latches to simulate the shift. As soon as the signs of the divisor and remainder are loaded, the first quotient bit is formed, based on a comparison of the two signs, and placed in temporary storage (Q_i). The first quotient bit, in accordance with the rules for nonrestoring division, determines whether the divisor will be added or subtracted to form the next remainder. This, in turn decides whether the carry or the borrow condition will be used to determine the second quotient bit. Thus the carry (or borrow) into the sign position is compared with the sign of the shifted remainder to determine the second quotient bit which is placed in Q_{i+1} . Because the remainder still has not been shifted, the shifted remainder sign is sensed from the 2^{-1} bit which will, after the shift, become the sign. As soon as the two quotient bits have been formed, they are loaded into the Quotient Register. During the first

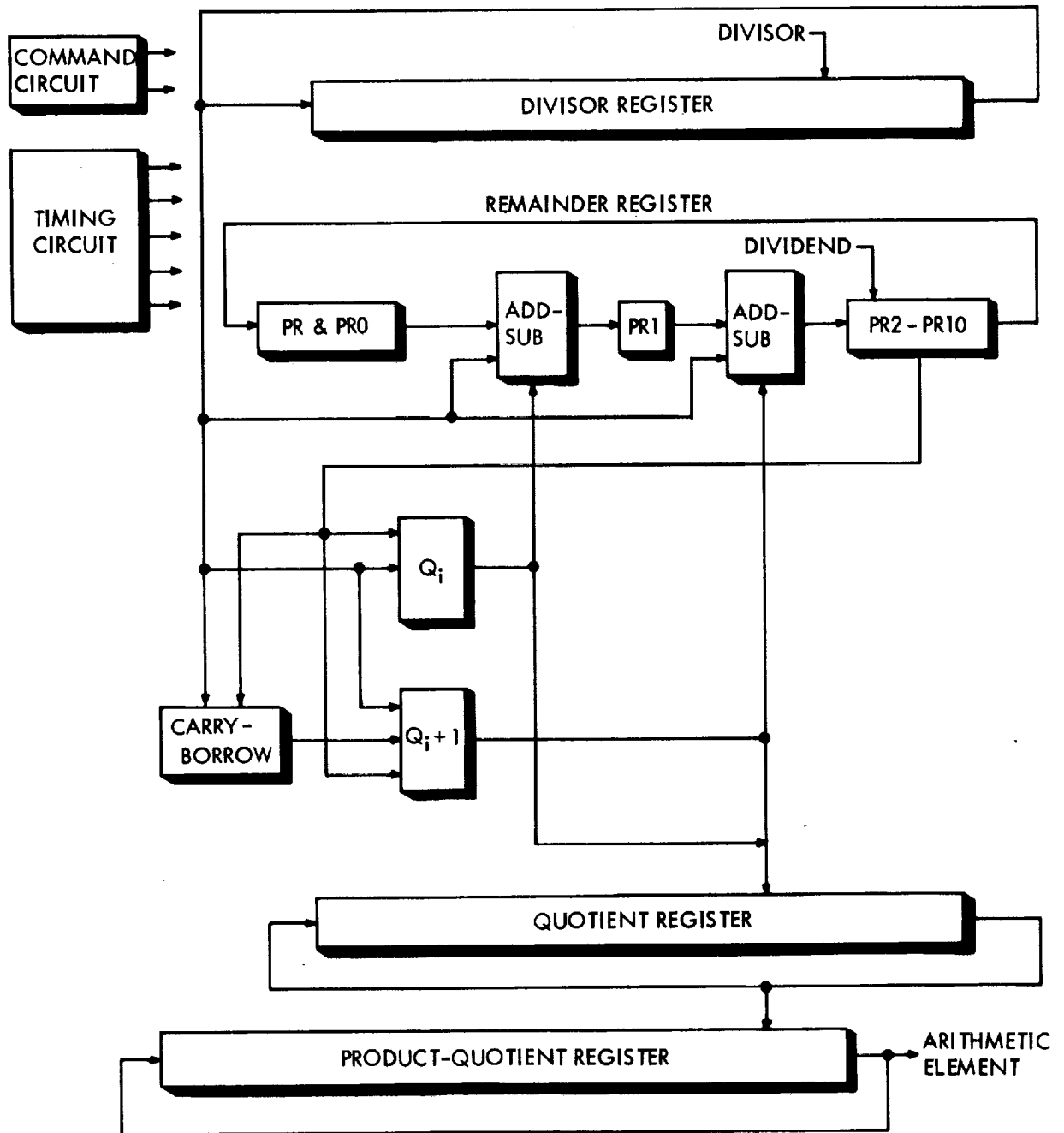


Figure 2-107. Divide Circuit, Block Diagram

Changed 4 January 1965

iteration, the first quotient bit is complemented in the quotient register to correct the sign. Recall from the description of nonrestoring division that the sign bit must be formed incorrectly in order to select the correct operation (add or subtract) to compute the next remainder. Thus the original quotient bits are retained in temporary storage to control formation of the remainders.

2-393. As the present remainder circulates in its register, it is shifted two places toward the MSD, one place for each of the two remainders which must be formed. The shifted remainder then enters two adder-subtractor circuits in serial. The outputs of these adder-subtractors are the two remainders formed each iteration.

2-394. The first adder-subtractor forms twice the "next remainder" by adding or subtracting (depending on the value of the first quotient bit) twice the divisor from four times (MSD shift of two) the "present remainder". Thus the output of the first adder-subtractor circuit is shown by the expression

$$4R_i \pm 2D \tag{1}$$

where i is the number of the "present remainder".

This expression can be simplified to

$$2(2R_i \pm D) \tag{2}$$

Since the "next remainder" (R_{i+1}) is formed by shifting the "present remainder" one place toward the MSD ($2R_i$) and adding or subtracting the divisor ($\pm D$), twice the "next remainder" is shown by equation (2), which represents the output of the first adder-subtractor. Thus, the output of the first adder-subtractor is the "next remainder" pre-shifted in preparation for forming the "second remainder". The output of the second adder-subtractor is the "second remainder" or the "present remainder" for the next iteration. As the last bits of the new remainder are formed, the first bits approach the carry and borrow latches. This begins the cycle for forming the next two quotient bits.

2-395. DIVISION LOGIC CIRCUITS. The division logic circuits perform the iterative divide process previously described when activated by the DIV instruction. The DIV instruction requires 27 phase times for completion during which the following operations are performed:

- 1) The Divide (DIV) instruction is read from memory and decoded.
- 2) Two quotient bits are developed during each iteration until the final 26-bit quotient is determined. This quotient contains 24 significant bits, the two low-order bits are zeros.
- 3) The final quotient is stored in the Product-Quotient (PQ) register.

2-396. The division process begins when the DIV instruction is decoded. The operand address of the DIV instruction specifies the memory location of the data word to be used as the divisor; the contents of the Accumulator Register are used as the dividend.

2-397. The divide circuit (figure 2-107) is composed of essentially the same three parts, Command Circuit, Timing Circuits and Arithmetic Registers, which make up the multiply circuit. Indeed, these parts perform the same general functions for division that they did for multiplication. The command circuit stores the DIV instruction until the division is complete. Outputs of the command circuit condition the timing circuits to

slightly alter the two-phase cycle and prolong the phase count to provide the extra time required for division. These outputs are also applied to the arithmetic registers, where they 1) condition circuits which were not used in multiplication; 2) decondition others which are not used for division; and 3) alter the purpose of others which are common to both, so that division is affected instead of multiplication. The arithmetic registers circulate the operands, and the circuits associated with the registers implement the division process.

2-398. Many of the division circuits are common to multiplication. For detailed descriptions of these circuits references are provided to the appropriate paragraphs of the Multiplication Logic Circuits description. These references will assist the reader in correlating the roles played by common circuits in the two operations. However, if the reader is thoroughly familiar with circuit operation for multiplication, these references may be overlooked to preserve continuity in the description of division.

2-399. Command Circuit. Operation of the command circuit is identical during multiplication and division except for the instruction codes involved. When a DIV instruction is decoded, the VOY latch is set; the HOY latch remains reset. See paragraph 2-328 and figure 2-92 for details on the command circuit.

2-400. Timing Circuits. (See figure 2-93.) As in multiplication, the TM and DTM latches define the two phase-time cycle upon which the division process is based. For division TMV is a "1" from the 7-Z time at the end of the cycle. The DTMV output is a "1" during the last half of the cycle, from 2-Z to 2-Z time. When neither a multiply or divide operation is in progress, both TM and DTM are reset.

2-401. The phase counter also parallels for division the operations it performed during multiplication (see paragraphs 2-332 through 2-336). As shown by the shaded AND gates in figure 2-94, separate inputs are used to start and stop the phase counter and to inject the first stop pulse into the delay line during division. Timing for the stop pulse is selected so that coincidence between STP and the remaining inputs to gate A5 first occurs 23 phase times less 2 bit times after the stop pulse is injected into the delay line. This allows just the proper time for a full accuracy division to be completed before switching the phase counter to the process complete state.

2-402. Arithmetic Registers. Division is accomplished by manipulating the contents of the shift registers shown in figure 2-107. These are the circuits which actually implement the computer division process described previously. Instrumenting the division involves some rather complex shifting and timed sampling operations. The reader will find it beneficial to note the logical value being sought when a particular position of a register is sampled at a given time. It may also help to quickly review the description of Computer Division from time to time.

2-403. The Divisor Register, figure 2-108, is the register which during multiply operations circulates the multiplicand (see paragraphs 2-339 through 2-342). The register is unchanged for use in division except for the divisor input gates and the input to the DL44 delay line. The divisor is entered through gate A8, figure 2-108. Since the circulation period of the divisor register is exactly two phase times, it does not shift its contents relative to the two phase-time cycle. Thus a particular bit of the divisor, sensed from the MD6 latch during bit-time 1 can be sensed again from the same latch at bit-time 1 two phase times later. For example, the sign of the divisor is available on the MD6 output at alternate 1-Z times. This is the actual output and timing used for sampling the divisor sign.

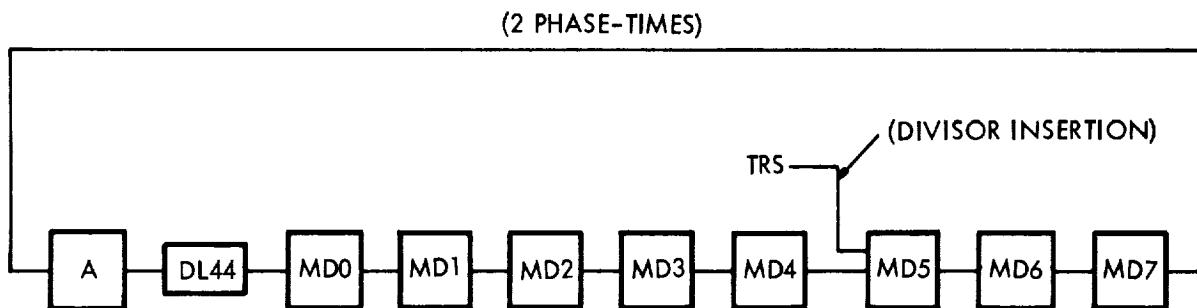


Figure 2-108. Divisor Register, Block Diagram

2-404. The Quotient Register, figure 2-109, circulates the accumulated quotient bits until the complete quotient is formed. When the last bits of the quotient have been formed and inserted into the quotient register, it is extended to form the product-quotient (PQ) register which circulates the final quotient for use elsewhere in the computer. The latches and delay line channel which compose the quotient register form a loop of two phase times plus two bit times. Consequently the quotient register shifts its contents two places toward the MSD each iteration. The shift vacates two storage locations for the quotient bits to be formed in the next iteration and moves the previously formed quotient bits toward their ultimate positions in the quotient. During the first iteration the quotient register also complements the first quotient bit to correct the sign of the quotient.

2-405. As shown in figure 2-99 (upper center), the two quotient bits enter the quotient register through the MR1 latch. Since the register circulates its contents toward the LSD, the second quotient bit formed is entered at 3-X time, ahead of the first quotient bit. The first quotient bit is entered at 4-X time; the entries are made near the end of the iteration which developed the bits. Latches SG1 and SG2 provide temporary storage for the second and first quotient bits, respectively.

2-406. During the first iteration, the quotient bits are allowed to circulate through the register into latches Q4 and Q5 before the sign is corrected. The sign bit enters the Q4 latch at Z clock time and at the following X clock time, gates A11 and A14 transfer the complement of the SG2 latch into Q4. Since the Q5 latch is not set until Y clock time, it sees the corrected sign, not the original. Gates A11 and A14 are conditioned by the stop pulse (STP) and A-7, 8-X timing. The stop pulse appears at STP during this time period only during iteration one, thus the complement is not made on subsequent cycles.

2-407. The Remainder Register includes the circuits diagrammed in figure 2-110. This register circulates and shifts the remainders generated by the division process. The circuits associated with the remainder register sense the remainders and the divisor to form the quotient bits and new remainders. The circulation time of the remainder register is the same as that of the quotient register, thus the remainder register, thus the remainder register too, shifts its contents two places toward the MSD each iteration. The current remainder is shifted one place for each of the two remainders which will be formed later in the iteration. Thus the input to the PR1 latch is four times the present remainder ($4R_i$).

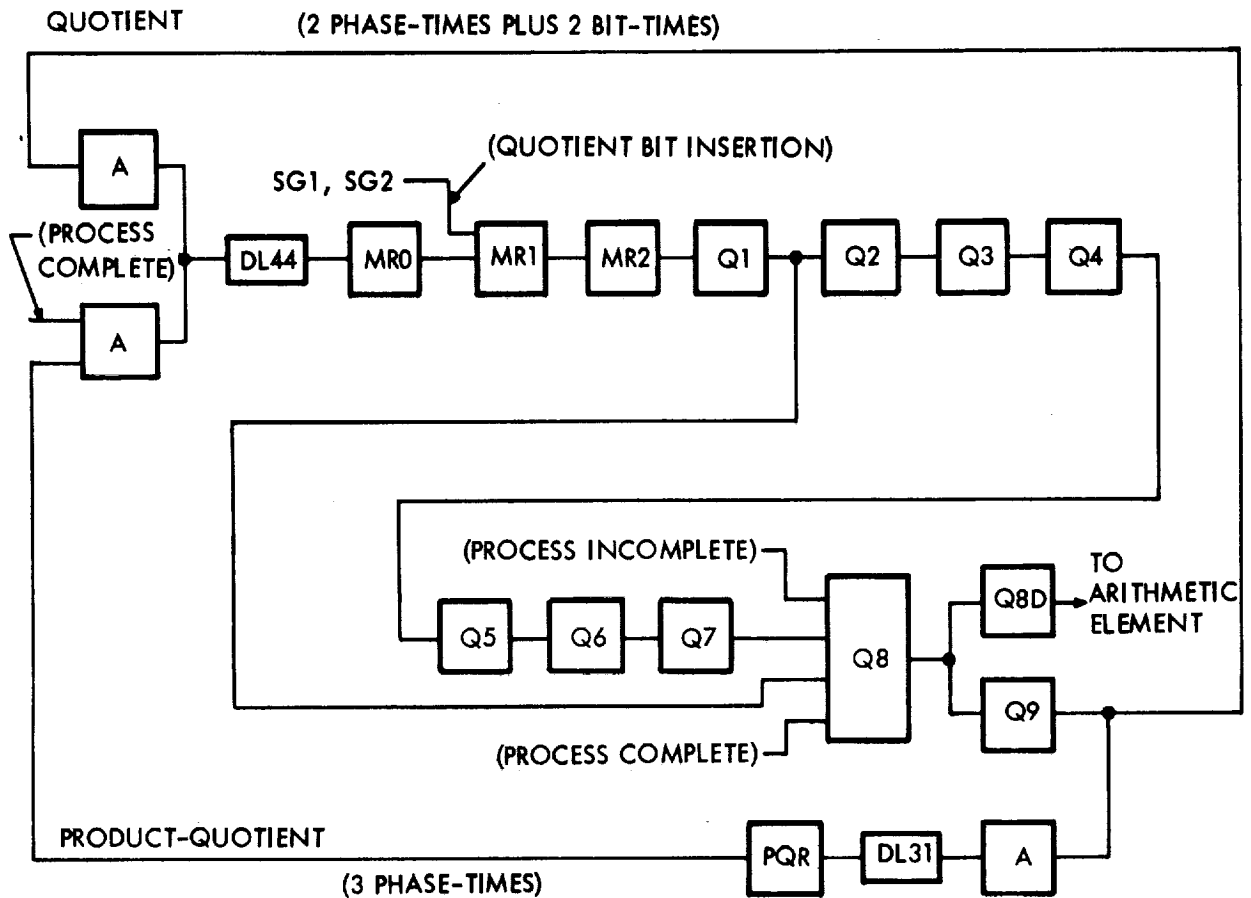


Figure 2-109. Quotient and Product-Quotient Registers

2-408. The dividend constitutes the first remainder and is entered from the accumulator register through the PR3 latch by double line transfer. Simultaneously, the divisor register is loaded from the transfer register. Loading the remainder register begins at B-4-W time of the first iteration.

2-409. The two adder-subtractors depicted in the divide circuit block diagram (figure 2-107) are those used to combine $\Delta 1_i$ and $\Delta 2_i$ with the previous partial products in multiplication (see paragraphs 2-360 through 2-363). For division the PR1 adder-subtractor computes the value $4R_i \pm 2D$. The significance of this expression was explained in paragraph 2-394. During division the $\Delta 2_i$ latch is controlled by timing signals. These timing signals cause the $\Delta 2_i$ latch to gate twice the divisor ($2D$) through the ESD latch to the summing gates at the input to PR1. The remaining inputs to the summing gates are the value $4R_i$ from PR0 and the carry or borrow of the summation developed by the K1 latch.

(2 PHASE TIMES PLUS 2 BIT-TIMES)

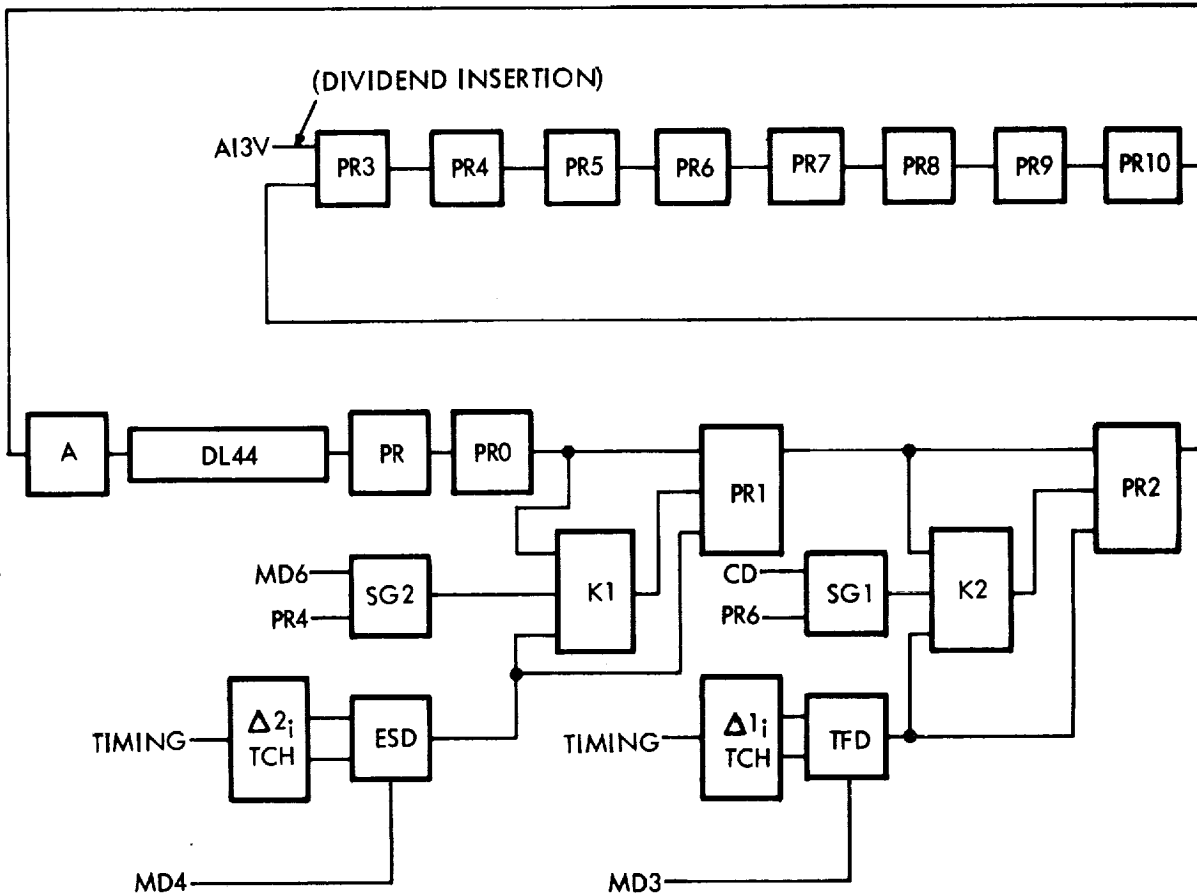


Figure 2-110. Remainder Register, Block Diagram

2-410. As in multiplication, sign latch SG2 determines whether K1 performs the carry or borrow function and consequently determines whether the summation results in addition or subtraction. The inputs to the SG2 latch compare the sign of the divisor (MD6) with the sign of the present remainder (PR4) at 1-Z time. If the signs match, the SG2 latch is set indicating that the first quotient bit (Q_i) is a "1". When the SG2 latch is set, K1 implements the borrow function and the subtraction required by the rules of non-restoring division for a "1" in the quotient bit is performed. Gates A48 and A49, figure 2-100 (sheet 3) implement the sign comparison.

2-411. Note from figures 2-108 and 2-110 that a delay line lies between the summing gates and both MD6 and PR4, the points at which the signs were sensed by the SG2 latch. This delay permits the SG2 latch to sample the signs and ascertain the quotient bit before the LSDs of the divisor and remainder reach the summing gates. When the LSDs do arrive, it has been predetermined from the quotient bit whether addition or subtraction is the operation required to form the next remainder.

2-412. Operation of the PR2 adder-subtractor is almost identical to that of the PR1 circuit. The $\Delta 1_i$ tratch provides the timing signal for the TFD latch which gates the divisor (D) to the PR2 summing gates. The SG1 latch forms and stores the second quotient bit Q_{i+1} (discussed later) in the bit time following formation of Q_i . The second quotient bit then conditions the K2 latch to borrow, if a "1", or carry if a "0". Thus the addition or subtraction necessary to form the second remainder is predetermined before the LSD bits arrive.

2-413. The second quotient bit (Q_{i+1}) is formed in the SG1 latch by comparing the sign of the shifted remainder (2^{-1} bit before the shift) with the carry or borrow into the sign position. If the shifted remainder sign, sensed at PR6, and the carry or borrow are both "1's" or both "0's" the Q_{i+1} bit is a "1", otherwise it is a "0". The comparison is accomplished by gates A43 and A44, figure 2-100 (sheet 3). Both the carry and the borrow into the sign position are represented by one signal, CD.

2-414. As the present remainder is shifted through the remainder register, two latches (CD and BD) keep running accounts of both the carry and borrow conditions which would occur if the divisor were being added to or subtracted from the shifted present remainder. However, no actual sum or difference is computed at this point. The CD and BD latches are shown at the lower right of figure 2-100 (sheet 3). As implied by the signal names, the CD latch stores the carry and the BD latch the borrow. The conditions which result in carries and borrows are given in the add-subtract truth table, figure 2-85; to correlate these conditions to the divide circuits, substitute PR6 and MD7 for AI4 and B respectively, in the truth table.

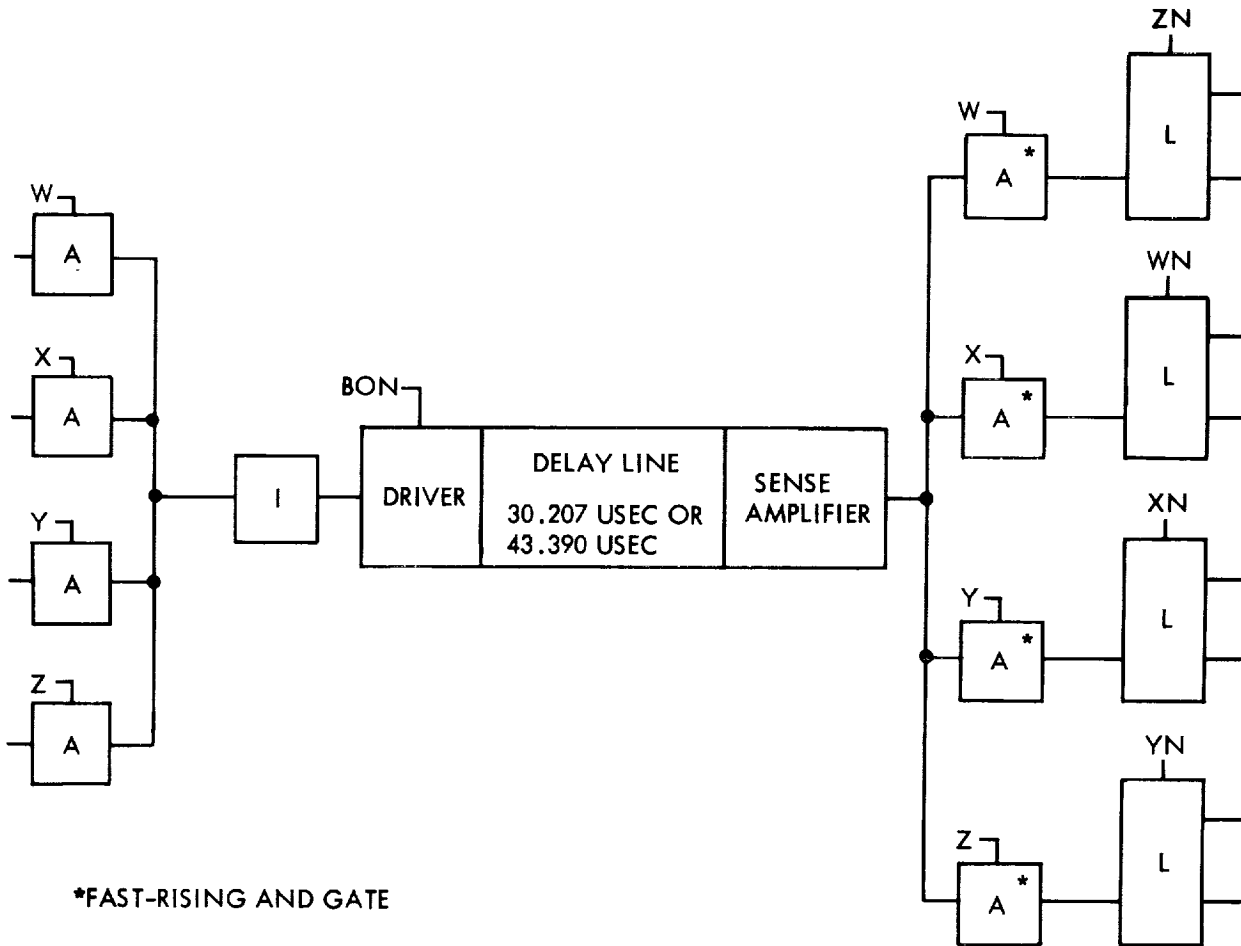
2-415. Notice that the CD and BD latches form all borrows and carries; the timing signals which select the carry or borrow condition into the sign position as the one to be compared are applied to gates A43 and A44 (upper left). The timing is such that the desired carry or borrow is formed simultaneously with the first quotient bit in SG2. This bit determines whether the carry or borrow will be used. However, only the outputs of the CD (carry) latch are routed to gates A43 and A44. If the borrow is to be used ($SG2 = 1$) the content of the BD latch is transferred into the CD latch. Thus, when the second quotient bit is formed during the following bit time, the output of the CD latch represents the borrow instead of the carry. Consequently, the appropriate function is represented at the sensing gates.

2-416. When the complete quotient has been formed, the quotient register is altered to form the PQ register described in paragraph 2-364. This is the same three phase-time register used to circulate the results of multiplications and it will now be used to circulate the result of the division.

2-417. During the last iteration of the division process, the two LSD quotient bits enter the quotient register through the MR1 latch. As the last bit enters MR1, the phase counter is set to the third phase condition. This substitutes the output of the Q1 latch for Q7 at the input to the Q8 latch. Thus latches Q2 through Q7 are bypassed and the final quotient enters the Q8 latch with the same timing as final product insertion at the end of a multiplication. The two low-order bits of the quotient are both "0's" which were present in latches MR2 and Q1 when the register was altered. Switching the phase counter to phase three also adds channel two of delay line DL31 and the PQR latch to the register and switches the input to DL44 from Q9 to PQR. This completes the circulation path of the PQ register. As in multiplication, the Q8D latch provides an output to the Arithmetic Element from which the quotient may be read.

2-418. DELAY LINES.

2-419. Two ultrasonic delay lines (DL31 and DL44) are utilized by the Arithmetic and Multiply-Divide Elements of the computer. The operation of both delay lines is identical except for the delay duration. A delay line with an associated driver, sense amplifier, and logic circuits which feed or retrieve delay line data is shown in figure 2-111.



*FAST-RISING AND GATE

NOTE - REFER TO FIGURES 10-3, 10-4 (SHEET 1), AND 10-5 (SHEET 2) FOR DETAILED LOGIC

Figure 2-111. Delay Line, Simplified Diagram

2-420. A delay line has four channels corresponding to the four clock times at which data is fed to the delay line. Channel 1 is fed at W clock time, and channels 2, 3, and 4 are fed at X, Y, and Z clock times, respectively. As data bits are loaded into a DL31 channel they emerge 16 bit times plus 2 clock times later; channel 1, 2, 3, and 4 data bits emerge at Y, Z, W, and X clock times, respectively (all 15 bit times later). Data bits entered into channels 1, 2, 3, and 4 of DL44 emerge at X, Y, Z, and W clock times, respectively (22 bit times later).

2-421. The output of the delay line is fed to four read latches. The read latches are set just prior to the emergence of each bit from the delay line and are either reset or not reset according to the nature of the bit from the delay line. The delay line output is logically inverted by the read latches to correspond to the inputs initially fed into the delay line input gates; the delay line inputs are logically inverted prior to being supplied to the delay line driver. A "1" fed into the delay line input gates is inverted to feed a "0" into the delay line. The "0" emerging from the delay line sense amplifier does not reset its associated read latch. Thus, the "0" initially read into the delay line is re-inverted. A "1" emerging from the delay line represents a "0" initially fed into the delay line input gates. Consequently, a "1" from the delay line resets its associated latch.

2-422. The delay line driver is conditioned not only by the inverter but also by buffer oscillator not (BON); BON equals "1" during the second half of each clock time. Consequently, when the inverter output is a "1", the delay line is not actually pulsed until the second half of the conditioning clock time. Therefore, the delay line output is only one-half of a clock time in width; for this reason fast rising AND gates are used with the read latches.

2-423. VOTING ELEMENT.

2-424. In the breadboard model II computer, the Voting Element performs two functions: 1) it serves as an interface circuit between the seven modules of the central computer and between the central computer, the memory and external equipment; and 2) it synchronizes the three Clock Generators. The ultimate function of the Voting Element, resolving errors between TMR circuits, is obviated by the simplex nature of the breadboard computer. The Electrical Description of the computer in Section I includes a brief description of voter operation in the TMR configuration.

2-425. The Voting Element is composed of a number of voter circuits interspersed through the computer. Each voter senses three inputs and produces an output identical to the majority of its inputs, hence the name "voter". The inputs to each voter are usually supplied by redundant circuits in channels 1, 2 and 3. Since these circuits are not provided in channels 2 and 3 of the breadboard computer, the corresponding inputs are held to a tie vote, i. e., one "0" and one "1". The input from channel 1 then breaks the tie and the voter effectively passes the channel 1 input straight through. Power gain and signal isolation are provided through the voter circuit.

2-426. As noted in the preceding paragraph, most of the redundant circuits are missing in the breadboard computer. However, where redundant circuits are provided, the voters must also be redundant to prevent malfunction of a single voter from causing the computer to fail. If a single voter is used, a failure in the voter results in erroneous signals being fed to all the circuits in the succeeding redundant set. This type of error cannot be corrected because it appears as a simultaneous failure of all the redundant circuits in one set. By using redundant voters, a separate voted signal is developed for each circuit in the succeeding set. (See figure 1-8.) A malfunction in one voter then appears as the failure of an individual circuit and is corrected by the next stage of voters. The number of redundant voters to be used is determined by the redundancy level of the circuits which use the voter outputs. If a stage of voters is to feed a triple redundant circuit, then three voters must be used; only two voters are required to feed a duplex circuit, etc.

2-427. Redundant voters are used in only two areas of the breadboard II model computer, the timing logic for the clock generators and at certain points between the computer and the memory. The voters in the clock generators are used for synchronization purposes rather than for error correction and are described under Timing Element. Duplexed voters are used to vote on signals which are common to both duplex memories. The logic pages on which the voters are mounted are identical to those for the TMR computer except that no circuits are provided for the voters which feed channels 2 and 3.

2-428. Typical examples of the two types of voters used in the computer are shown at A and B in figure 2-112 with their equivalent circuit, C. The operation of both type voters is essentially the same except for driving capability which is given in the Logic Symbols appendix. Each voter includes a separate AND input for the signals from the three redundant channels. The clock input of each AND circuit receives a separately controlled "module switching voltage". In the TMR model computers, the module switching voltages may be manipulated by the test equipment as an aid to trouble isolation. In the breadboard model II computer, one of the module switching voltages is held at 0 VDC to simulate a "0" input from one of the missing channels. A "1" input is simulated for the other missing channel by providing the normal module switching voltage (+12 VDC) and leaving the signal input to the AND open circuited.

2-429. Each "1" input to the voter causes a current I to flow through the current summing circuit to the enabled AND circuit. A current of $2I$ is sufficient to switch the output of the summing circuit from a "1" to a "0". When two "1's" are present at the voter input, the sum of the currents is sufficient to switch the summing circuit output; the inverter output then agrees with the majority input. The voter delays its input by approximately one-third of a clock pulse. Thus a voted signal, unless clocked by a fast-rising AND, may not be sensed until the second clock after it is switched.

2-430. COMPUTER OPERATIONS.

2-431. Descriptions of the computer operations are not provided for the breadboard model II computer. This section of the manual will be included in the revisions for the simplex prototype model computers.

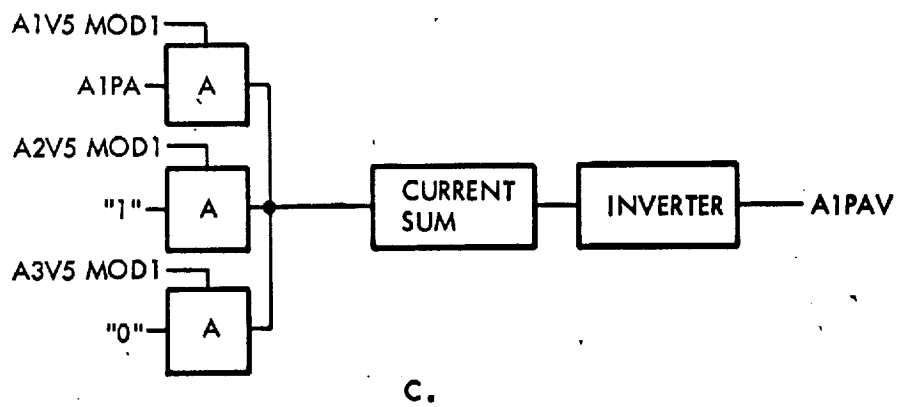
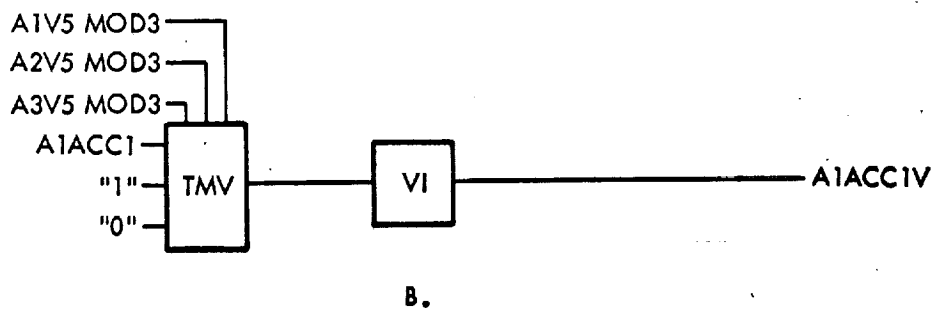
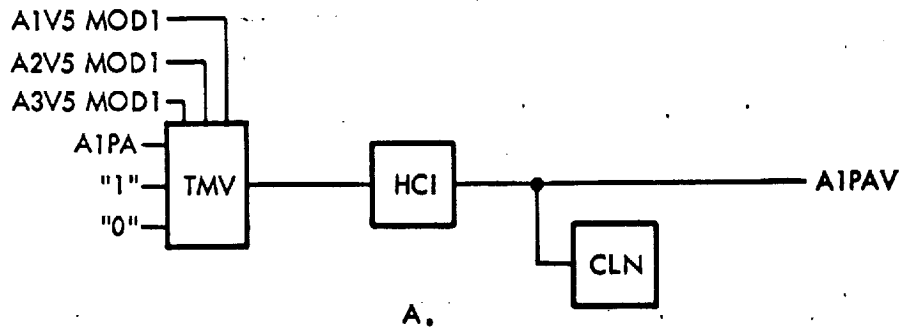
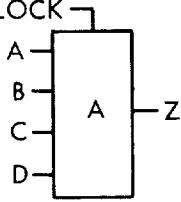
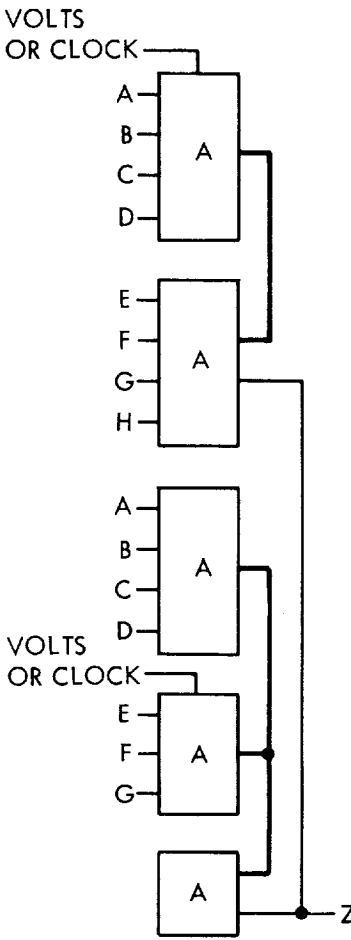
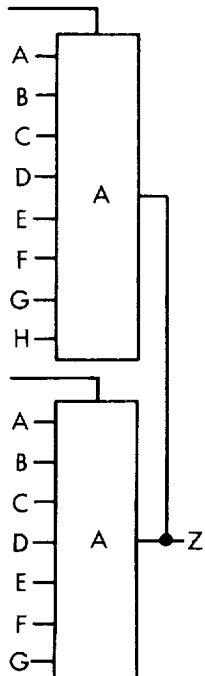


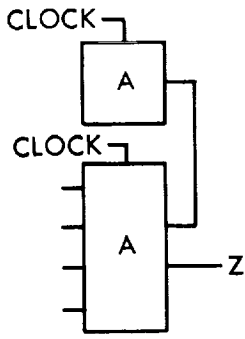


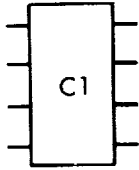
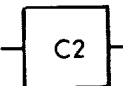
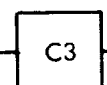
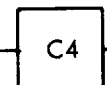
Figure 2-112. Voter Circuits


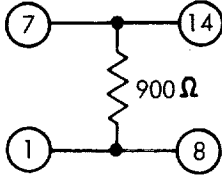
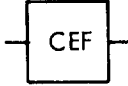

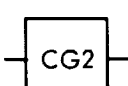

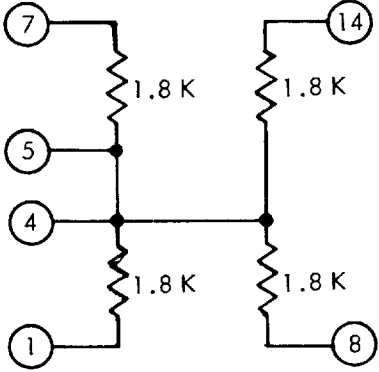
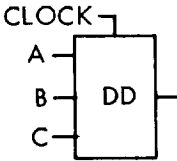
Changed 4 January 1965

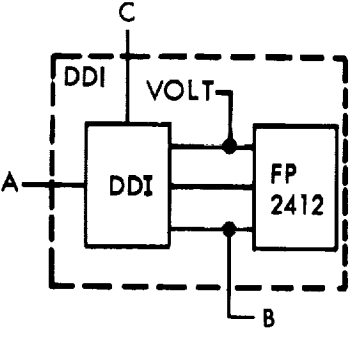

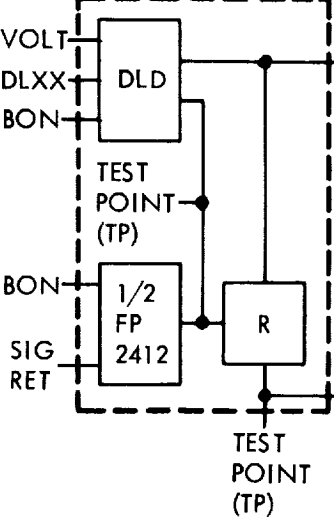

2-177/2-178

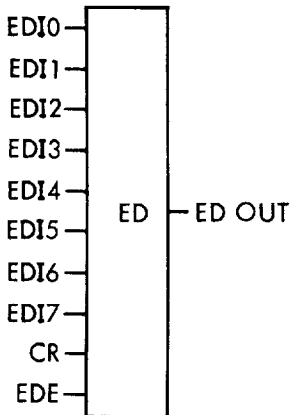
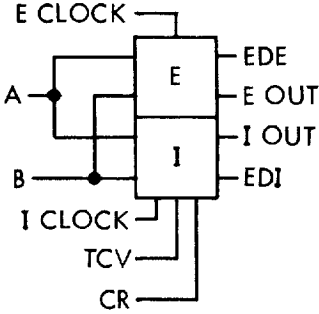

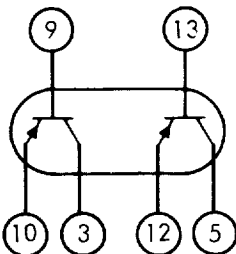


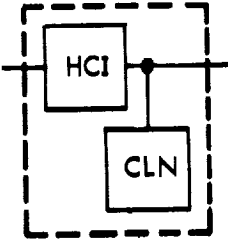
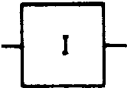
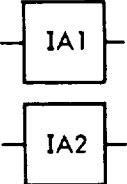
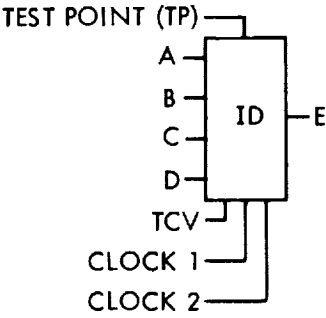

SYMBOL	NAME	DESCRIPTION
<p>VOLTS OR CLOCK</p> 	<p>AND Gate</p>	<p>$\text{Volts} + \text{Clock} \cdot A \cdot B \cdot C \cdot D = Z$</p>
<p>VOLTS OR CLOCK</p> 	<p>AND Gate, AND Extenders</p>	<p>VOLTS OR CLOCK</p>  <p>VOLTS OR CLOCK</p> <p>Heavy line (shown heavy for discussion purposes) indicates an AND extender. Gates connected by this type line make circuit the equivalent of one large AND gate.</p>

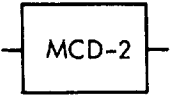
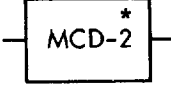
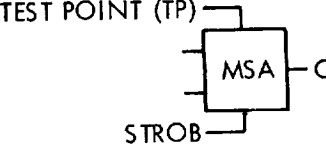

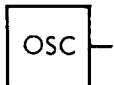

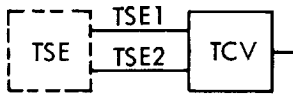
SYMBOL	NAME	DESCRIPTION
	<p>AND Gate, Fast-Rising</p>	<p>Functions like previous AND gate except this type has parallel load resistors to decrease rise time. Thus, its output can be used one clock time sooner.</p>
	<p>Buffer Power Amplifier</p>	<p>Special circuit used in buffer amplifier. Provides high power and special output levels for operating delay lines.</p>
	<p>Buffer Shaper</p>	<p>Special circuit used in buffer amplifier converts sine wave input to voltage square wave output.</p>
	<p>Capacitor (Four)</p>	<p>0.22 ufd, 25 VDC</p>
	<p>Capacitor</p>	<p>2.2 ufd, 20 VDC</p>
	<p>Capacitor</p>	<p>5.6 ufd, 10 VDC</p>
	<p>Capacitor</p>	<p>8.2 ufd, 6 VDC</p>

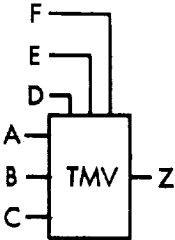
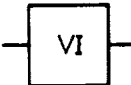
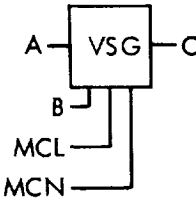
SYMBOL	NAME	DESCRIPTION
	<p>ULD Type CDR</p>	
	<p>Complementary Emitter-Follower</p>	<p>Two emitter-followers connected back-to-back so that "1" drives output up and "0" drives output down.</p>
	<p>Clock Gate 1</p>	<p>Two-input clock gate. Special circuit which AND's two inputs when "1" and produces a "1" output. Provides special driving levels for NSI.</p>
	<p>Clock Gate 2</p>	<p>Produces a "1" output with a "1" input. Provides special driving levels for NSI.</p>
	<p>ULD Type CLN</p>	
	<p>Disagreement Detector</p>	<p>If inputs A, B, and C are at same level ("1" or "0") when clocked, output is a "1". If one is different, output is a "0".</p>

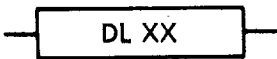
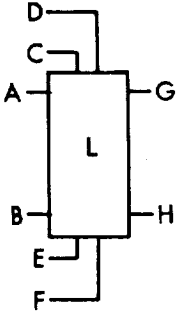
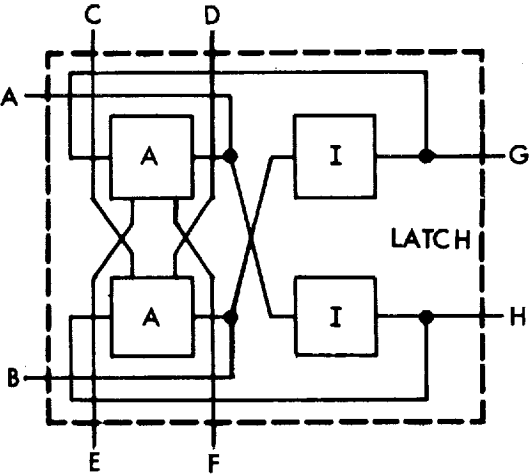
SYMBOL	NAME	DESCRIPTION
	<p>Disagreement Detector Inverter</p>	<p>Circuit consists of a DDI type ULD and flat pack. Output B is inverse of input A. Input C is a module switching control voltage.</p>
	<p>Delay Line</p>	<p>Input "1's" appear at output as voltage pulses after a specified delay time. Delay time in usec specified by symbol "DL XX" at input to DLD. ("XX" = 31 or 44.)</p>
	<p>Delay Line Driver</p>	<p>Circuit consists of: DLD type ULD, resistor R, and half of a flat pack. Delay line driver circuit gates drive pulses (BON) into delay line when DL XX = "1". ("XX" = 31 or 44.)</p>
	<p>Delay Line Sense Amplifier</p>	<p>Amplifies voltage pulses from delay line and converts to "1's".</p>

SYMBOL	NAME	DESCRIPTION
	<p>Error Detector (Memory)</p>	<p>When memory is addressed, an "up level" is supplied to the CR input causing ED OUT = "1". A loss of this "up level" forces ED OUT to "0". Inputs EDI 0 through EDI 7 forces ED OUT to "0" if more than one input is selected. Threshold voltage on EDE input rises when two or more EI voltage sources are selected, thus forcing ED OUT to "0".</p>
	<p>EI Driver (Memory)</p>	<p>Consider EI driver as two separate units. E is a voltage source and I a constant current source. If inputs A and B are "1's" when E clock is present, an output occurs at E OUT and EDE. If inputs A and B are "1's" when I clock is present, an output occurs at I OUT and EDI. TCV controls amount of current flow through I OUT. CR is a current return path which is connected to ground through an externally provided resistor. A and B are essentially ANDed together. When B is not connected, A is the controlling input.</p>
	<p>Flat Pack</p>	<p>A pair of transistors mounted in a ULD location.</p> 

SYMBOL	NAME	DESCRIPTION
	<p>High Current Inverter</p>	<p>Circuit consists of ULD types HCl and CLN. Output is the inverse of input and can drive twenty AND-gate inputs.</p>
	<p>Inverter</p>	<p>The inverter performs logical and electrical inversion. An open input is interpreted as a "0". Where inverter outputs are ORed together, an inverter producing a "0" output takes precedence.</p>
	<p>Inverter Amplifier Type 1 or Type 2</p>	<p>Special circuit used only in buffer amplifier. Performs electrical inversion and level shifting.</p>
	<p>Inhibit Driver Circuit</p>	<p>Produces current pulse at E coincident with clock 2 when one or both signals of both signal pairs (A, B) and (C, D) are "0's". Used to cancel Y selection current for planes in which "0's" are to be stored when writing "1's" into memory. If both signals in either pair (A, B) or (C, D) are "1's", output is inhibited allowing a "1" to be written into memory plane. Output current is varied with temperature of memory array by TCV input. Clock 1 prevents the circuit from drawing current when memory is not being operated.</p>
	<p>Memory Clock Driver 1</p>	<p>When both inputs are "1's", the MCD-1 produces a 3.5 usec output pulse coincident with the fall time of the first input disabled.</p>

SYMBOL	NAME	DESCRIPTION
	Memory Clock Driver 2	Generates a 2.5 usec pulse delayed 0.5 usec from leading edge of input pulse.
	Memory Clock Driver 2*	Generates a 1.5 usec pulse delayed 0.5 usec from leading edge of input pulse.
	Memory Sense Amplifier	The MSA receives a double ended input from a memory sense winding. The MSA shapes and amplifies memory output signal whenever a "1" is read from memory array. Width of signal is dependent upon width of STROB down level. When a "1" is sensed, MSA produces an output capable of driving one AND-gate input; a zero sense signal produces no output.
	Non-Saturating Inverter	Special circuit (not ULD NSI) which provides logic inversion. Held out of saturation to provide fast response time.
	Oscillator	Produces 2.048 MC (± 51 CPS) sine wave output. Has no supply voltage input; draws current from load.
	Thermistor	<p>D. C. Resistance:</p> <p>3.85K (± 385) ohms at $50^\circ (\pm 0.1^\circ)\text{C}$ 10K (± 200) ohms at $25^\circ (\pm 0.1^\circ)\text{C}$ 30K ($\pm 2.85\text{K}$) ohms at $0^\circ (\pm 0.1^\circ)\text{C}$</p>
	Temperature Control Voltage Regulator	The TSE is a temperature sensing element located in the memory array. The TCV output voltage varies such that each degree of temperature increase causes a fixed voltage decrease, and vice versa.

SYMBOL	NAME	DESCRIPTION
	<p>ULD Type TMV</p>	<p>Output Z is inverse of majority of inputs A, B, and C, when D, E, and F are all at +12 VDC. Switching D, E, or F to zero volts effectively forces a "0" at A, B, or C, respectively. This feature is used for module switching.</p>
	<p>ULD Type VI</p>	<p>Output is inverse of input and is capable of driving ten AND-gate inputs.</p>
	<p>Variable Delay Strobe Gate</p>	<p>Input pulse at A is delayed in the VSG and provides a down-level pulse at C. C is inhibited when B is a "1". MCL and MCN are used for marginal checking memory performance. Advanced strobe is performed by grounding MCN, and delayed strobe by applying +6 VDC to MCL; during normal operation these inputs are open circuited.</p>

SYMBOL	NAME	DESCRIPTION
 <p>XX = DELAY LINE TIME IN MICROSECONDS</p>	<p>Delay Line</p>	<p>The delay line circuit consists of a glass delay line, a delay line driver, and a delay line sense amplifier.</p> <p>A "1" at the input is delayed by the amount of time indicated on the symbol.</p>
	<p>Latch</p>	<p>A bistable storage device which can be operated by "1's" (on input A or B) or by "0's" (on inputs C, D, E or F). Inputs A and B are normally "0" and are called "1" drive inputs. Inputs C, D, E, and F are normally "1" and are called "0" drive inputs. A "1" at A will produce a "1" at G. The "1" at G will remain irrespective of conditions at A until one of the other inputs is driven. In similar fashion, a "1" at B produces a "1" at H. A "0" at C or D produces a "1" at G; a "0" at E or F produces a "1" at H.</p> 

GLOSSARY

<u>SYMBOL</u>	<u>DEFINITION</u>
A	Bit Gate Generator Latch A, used to decode even numbered bit gates during W and X clock times
AB	Bit Gate Generator Latch AB, used to control the A Latch
Accumulator	Circuit which consists of an adder and a register; the output of the adder is stored in the register
ADD	Programmed Add operation
AND	Latch which decodes logical AND operation
AX0N thru AX7N	Inverse low order X axis decoded memory addresses, octal 0 thru 7
A1 thru A8	Address Register Latches 1 thru 8
A9	Address Register Latch 9, used to select the Residual Memory Sector
A1AN thru A8AN	Zero output of Address Register Latches 1 thru 8
A9PADN	Inverse signal generated to select the Residual Memory
ACC0	Latch in the Accumulator-Register which stores serial output of 44.5 microsecond delay line
ACC1	Delay latch in Accumulator-Register
AI0	Latch in Accumulator-Register which stores serial output of 31 microsecond delay line
AI1 thru AI4	Delay latches in Accumulator Register
AX00N thru AX70N	Eight inverse high order X axis decoded memory addresses, octal 0 thru 7
AY0N thru AY7N	Inverse low order Y axis decoded memory addresses, octal 0 thru 7
AY00N thru AY70N	Eight inverse high order Y axis decoded memory addresses, octal 0 thru 7
B	Accumulator latch which controls one of the data entries to Accumulator

<u>SYMBOL</u>	<u>DEFINITION</u>
b	Multiplier
BD	Latch used during divide to generate the borrow of twice the remainder minus the divisor
BFR DVR	Buffer Driver
BFR SHP	Buffer Shaper
BO	Buffered Oscillator Signal
BRAO	Latch which gates Buffer Register A outputs into memory inhibit drivers
BRBO	Latch which gates Buffer Register B outputs into Memory Inhibit Drivers
BO1 thru BO3	Outputs of Buffer and Oscillator circuit for clock timing in channels 1 thru 3
BOT1 thru BOT3	Intermediate outputs of Buffer and Oscillator circuit for clock timing in channels 1 thru 3
BRA1 thru BRA14	Buffer Register A Latches 1 thru 14
BRA14P	Special output of memory parity bit for test equipment
BRB1 thru BRB14	Buffer Register B Latches 1 thru 14
BRB14P	Special output of memory parity bit for test equipment
C	Arithmetic Element Carry Latch
CBRN	Inverse signal generated to clear the Buffer Registers
CD	Latch used during divide to generate the carry of twice the remainder plus the divisor
CDS	Change Data Sector Register Latch - used during a HOP or Change Data Sector instruction
CEF	Complementary Emitter Follower
Channel	Group of pages which contain logic circuits. Computer consists of three TMR channels and two simplex channels
CKP	Check Parity Latch
CLA	Programmed Clear and Add operation
CLTR	Clear Transfer Register Latch

<u>SYMBOL</u>	<u>DEFINITION</u>
CNC	Check No Current Latch
COC	Check On Current Latch
Computer Cycle	A-1-Y time to C-14-Y time
CST	Signal which permits single step operation of the computer
CSTN	Inverse signal from external equipment used to control computer single step operation
DATAV	Normal data input from external equipment
Data word	28 bits including two parity bits
DIN	Special data input from LTE used to control memory load and verify operations
DIV	Programmed divide operation
DLD31B, DLD44B	Outputs of delay line driver circuits to delay lines
DL31	Input to the 31 microsecond Delay Line Driver
DL44	Input to the 44.5 microsecond Delay Line Driver
DL31SA	Output of the 31 microsecond Delay Line Sense Amplifier
DL44SA	Output of the 44 microsecond Delay Line Sense Amplifier
DMA	An inverter output indicating that the data is from an even numbered memory module
DMB	An inverter output indicating that the data is from an odd numbered memory module
DM0	Tratch output indicating that the low order position of the Data Module Register is a zero
DM1	Tratch output indicating that the low order position of the Data Module Register is a one
DM2	Second position of the Data Module Register
DM3	Third position of the Data Module Register
DSS	Data Sector Serialized Latch - gates data sector bits into the Data Sector Register for HOP or Change Data Sector instruction
DS1 thru DS4	Data Sector Latches 1 thru 4

<u>SYMBOL</u>	<u>DEFINITION</u>
DS1M	First Data Sector Register Modified Latch used during an EXM instruction
DS2M	Second Data Sector Register Modified Latch used during an EXM instruction
DTM	Timing latch in the Multiply-Divide Element
DUPDN	Tratch output indicating that the memory is in the simplex mode of operation for data
DUPIN	Tratch output indicating that the memory is in the simplex mode of operation for instructions
EAC	An inverter output indicating a failure in an even numbered module due to a half-select error in either a data or instruction cycle
EADM	Latch indicating a failure in one of the even numbered memory modules during an operation cycle
EAIM	Latch indicating a failure in one of the even numbered memory modules during an instruction cycle
EAM	Latch which indicates an error in even memories
EAP	Output from Buffer Register A parity check logic, error signal
EBC	An inverter output indicating a failure in an odd numbered module due to a half-select error in either an operation or instruction cycle
EBDM	Latch indicating a failure in one of the odd numbered memory modules during an operation cycle
EBIM	Latch indicating a failure in one of the odd numbered memory modules during an instruction cycle
EBM	Latch which indicates error in odd memories
EBP	Buffer Register B parity check logic, error signal
EDAC	Latch indicating a failure of one of the even numbered memory modules as detected by the half-select current monitors during an operation cycle
EDBC	Latch indicating a failure of one of the odd numbered memory modules as detected by the half-select current monitors during an operation cycle

<u>SYMBOL</u>	<u>DEFINITION</u>
EDmX, EDmY	Output of half-select current Error Detector circuit in memory
EIAC	Latch indicating a failure of one of the even numbered memory modules as detected by the half-select current monitors during an instruction cycle
EIBC	Latch indicating a failure of one of the odd numbered memory modules as detected by the half-select current monitors during an instruction cycle
EMDN	Zero during a Multiply operation if delta 2 equals eight, zero during Divide
ESD	Latch used as delta 2 during either Multiply operation, or as twice the divisor during Divide
EXM	Latch which decodes the Execute Modified instruction
EXMD	Execute Modified Instruction Delayed Latch
FCD	Signal which forces a Clear And Add operation into the Operation Code Register
FMDN	Zero during a Multiply iteration if delta 1 equals four, zero during Divide
FSA	Latch indicating that a failure has been sensed by the half-select current monitor in an even numbered module
FSB	Latch indicating that a failure has been sensed by the half-select current monitor in odd numbered module
G1 thru G7	Bit Gate Generator Latches 1 thru 7
HALTV	External signal used to control program initiation
HOP	Latch which decodes HOP instruction
HOY	Latch set by Multiply And Hold or Multiply
HOPC1	Latch which generates the HOP constant for storage during an Interrupt operation
HP1	Latch which generates the HOP constant during odd bit times
IMA	An inverter output indicating that the instruction is from an even numbered memory module
IMB	An inverter output indicating that the instruction is from an odd numbered memory module

<u>SYMBOL</u>	<u>DEFINITION</u>
INHBS	Signal which inhibits strobing of the Memory Sense Amplifiers
Instruction Cycle	A-1-Y time to B-1-Y time
Instruction Word	13 instruction bits + 1 parity bit
INT	Latch indicating an Interrupt
INTA	First Interrupt Interlock Control Latch
INTB	Second Interrupt Interlock Control Latch
INTCV	External signal which indicates that program should be interrupted
INV	Inverter
ISS	Instruction Sector Serialized Latch - gates instruction sector bits into the Instruction Sector Register for a HOP instruction
IM0	Tratch output indicating that the low order position of the Instruction Module Register is a zero
IM1	Tratch output indicating that the low order position of the Instruction Module Register is a one
IM2	Second position of the Instruction Module Register
IM3	Third position of the Instruction Module Register
IS1 thru IS4	Instruction Sector Latches 1 thru 4
JBN	Partial decode of Shift instruction
K1	Latch used as carry or borrow by PR1
K2	Latch used as carry or borrow by PR2
M	Multiplicand
MAO	Latch output which gates Buffer Register A output into the Transfer Register
MB	Multiplier Bit
MBO	Latch output which gates Buffer Register B output into the Transfer Register
MFF	An inverter output indicating computer operation in either Memory Module four or five

<u>SYMBOL</u>	<u>DEFINITION</u>
MmCRX	Memory Module m X Address Driver current return
MmCRY	Memory Module m Y Address Driver current return
MmEDEX, MmEDEY	Voltage monitor output from memory address drivers to half-select current Error Detector circuit in memory
MmTCV	Memory Module m Temperature Controlled Voltage
MmTSE	Memory Module m Temperature Sense Element
Module	Group of circuits on a page(s) performing the same or related functions. Seven modules in each of the three redundant channels
MOP	Signal indicates a read or store Memory operation
MPH	Programmed Multiply and Hold operation
MPY	Programmed Multiply operation
MSS	An inverter output indicating computer operation in either Memory Module Six or Seven
MTT	An inverter output indicating computer operation in either Memory Module Two or Three
MZO	An inverter output indicating computer operation in either Memory Module Zero or One
M0SYNC	Start signal to Memory Module 0 which initiates a memory cycle
M1SYNC thru M7SYNC	Start signal to Memory Module 1 thru Memory Module 7 which initiates a memory cycle
MD0 thru MD7	One output of latches in the Multiplicand-Divisor Register
MmEDI 0 thru MmEDI 7	Voltage monitor output from memory address drivers to half-select current Error Detector circuit in memory
MmEDIY 0 thru MmEDIY 7	Special outputs of Y memory address drivers for current monitoring by Error Detector circuits
MmHEY0 thru MmHEY7	Memory Module m High Y Address Driver voltage sources, octal 0 thru 7
MmHIY0 thru MmHIY7	Memory Module m High Y Address Driver current sources, octal 0 thru 7
MmINH1 thru MmINH14	Memory Module m Inhibit Drive 1 thru 14

<u>SYMBOL</u>	<u>DEFINITION</u>
MmLEX0 thru MmLEX7	Memory Module m Low X Address Driver voltage sources, octal 0 thru 7
MmLEY0 thru MmLEY7	Memory Module m Low Y Address Driver voltage sources, octal 0 thru 7
MmLIX0 thru MmLIX7	Memory Module m Low X Address Driver current sources, octal 0 thru 7
MmLIY0 thru MmLIY7	Memory Module m Low Y Address Driver current sources, octal 0 thru 7
MmRDP1 thru MmRDP3	Memory Module m Read Pulse 1 thru 3
MmS1E0 thru MmS1E7	Memory Module m Syllable 1 Address Driver voltage sources, octal 0 thru 7
MmS1I0 thru MmS1I7	Memory Module m Syllable 1 Address Driver current sources, octal 0 thru 7
MmSA1 thru MmSA14	Memory Module m Sense Amplifier, outputs 1 thru 14
MmSL1A thru MmSA14A	Memory Module m Sense Lines 1 thru 14, A end
MmSL1B thru MmSL14B	Memory Module m Sense Lines 1 thru 14, B end
MmS0E0 thru MmS0E7	Memory Module m Syllable 0 Address Driver voltage sources, octal 0 thru 7
MmS0I0 thru MmS0I7	Memory Module m Syllable 0 Address Driver current sources, octal 0 thru 7
MmSTROB	Memory Module m Sense Amplifier Strobe pulse
MmSTRP1 thru MmSTRP3	Memory Module m Store Pulses 1 thru 3
MR0 thru MR2	Latches in the Multiplier, Quotient, Product-Quotient Registers
N	Letter symbol suffix, when added to signal name, indicates a NOT condition
NSI	Non-Saturating Inverter
NU	Latch in Automatic HOP Save Circuit
OC	Latch which generates the HOP constant during even bit times
Operand	Data used in arithmetic operation

<u>SYMBOL</u>	<u>DEFINITION</u>
Operation Cycle	B-1-Z time to C-14-Y time
OP1 thru OP4	Operation Code Register Latches 1 thru 4
P	Letter symbol suffix, when added to signal name, indicates power amplification
P	Latch in clock generator timing logic
P	Product of a multiply operation
PA	Phase Generator Latch A
PAD	Phase A Delayed Latch
PAR	Latch which generates the final parity bit for transfer to the Memory Buffer Register
Parity bit (odd)	A bit, when added to computer word or syllable, makes total number of bits an odd number
PB	Phase Generator Latch B
PC	Phase Generator Latch C
PDD	Parity Generator Latch Delayed
PIO	Decoded Process Input Output operation
Plane	All cores in a memory module which store the same bit position in a syllable
POD	Parity Generator Latch
PQR	Latch in the Product-Quotient Register
PR	Latch in the Partial Product-Remainder Register
PWR RET	Power return
P1N	Zero during first iteration of Multiply, Multiply and Hold, and Divide
P2N	Zero during second and remaining iterations of Multiply, Multiply and Hold, and Divide
P3N	Set to zero after final iteration of Multiply, Multiply and Hold, and Divide
PAE1 thru PAE5	Buffer Register A parity check logic, even inverters 1 thru 5
PAE6N	Buffer Register A parity check logic, even inverter 6

<u>SYMBOL</u>	<u>DEFINITION</u>
PAO1 thru PAO12	Buffer Register A parity check logic, odd inverters 1 thru 12
PBE1 thru PBE5	Buffer Register B parity check logic, even inverters 1 thru 5
PBE6N	Buffer register B parity check logic, even inverter 6
PBO1 thru PBO12	Buffer register B parity check logic, odd inverters 1 thru 12
PR0 thru PR10	Latches In Product-Remainder Register
Q	Latch in clock generator timing logic
Q1 thru Q9	Latches in the Quotient, Product-Quotient Register
Q8D	Q8 Delayed Latch
R	Latch in clock generator timing logic
RAC	Latch which controls recirculation of contents of the Accumulator-Register
RD	Signal indicates a Read Memory operation
RDM	Signal from latch which controls reading from the memory
RECN	Manual reset for error detect logic
RED	Read Data Latch
REI	Read Instruction Latch
RUN	Latch which is externally controlled and starts the computer operating
S	Latch in clock generator timing logic
SAPO	Partially decoded SHF and PIO operations
SBRX	Signal generated to transfer the contents of the Transfer Register into the Buffer Registers at X time
SBRY	Signal generated to transfer the contents of the Transfer Register into the Buffer Registers at Y time
SBRZ	Signal generated to transfer the contents of the Transfer Register into the Buffer Registers at Z time
Sector	Preselected area of memory module for memory addressing

<u>SYMBOL</u>	<u>DEFINITION</u>
SHF	Programmed shift operation; output of the SHF Operation Decoder circuit
SINK	Latch which generates the initial start memory signal
SLD	Syllable Delayed Latch
SMDN	Zero during a Multiply operation if delta 2 equals sixteen
SRTR	Shift Right Transfer Register Latch
SS	Decoded octal address 776 or 777
SSF	Decoded operand address 775
SSFSN	Decoded operand address 775 or 777, used to develop SSF
STMD	Latch causing data from the Multiplicand Register to be stored
STO	Programmed Store operation, output of STO Operation Decoder circuit
STP	Latch in the Multiply-Divide Timing Circuit
SUB	Programmed Subtract operation
Syllable	Half of a computer word; half of a memory module
SYNC	Latch which develops the memory start signal when gated with the Memory Module Register outputs
S4	Latch which controls selection of the Memory Sector
SG1	Latch which stores the sign of delta 1 during either Multiply operation, or the lower order quotient bit during Divide
SG2	Latch which stores the sign of delta 2 during either Multiply operation, or the higher order quotient bit during Divide
SYL0N	Inverse outputs of latch which controls reading or writing of syllable zero or syllable one in the memory
SYLC1	Instruction Syllable Control Latch
TA	Latch which transfers the address in the Transfer Register to the Address Register
TBC	Timing latch which gives output from B-3-Y to A-1-Y

<u>SYMBOL</u>	<u>DEFINITION</u>
TBR	Latch which causes a transfer of data from the Buffer Register to the Transfer Register
TFD	Latch used as delta 1 during either Multiply operation, or as the Divisor during divide
TIME	Latch which indicates when the parity check logic should be sampled for an error condition
TLC	Latch indicating two simultaneous memory errors, an abort condition
TM	Timing Latch in the Multiply-Divide Element
TMDM	Zero during a Multiply iteration if delta 1 equals two
TMI	Programmed Transfer Minus operation
TNZ	Programmed Transfer Non- Zero operation
TRA	Programmed Transfer operation
TRS	Transfer Register Serial Output Latch
TTL	Latch used to gate data transfers between PQ Register and Accumulator-Register
TTT	Signal indicating no transfer operation
TR1 thru TR13	Transfer Register Latches 1 thru 13
TR1D	Transfer Register Latch 1 Delayed
TR3D	Transfer Register Latch 3 Delayed
TR6D	Transfer Register Latch 6 Delayed
TR9D	Transfer Register Latch 9 Delayed
TR12D	Transfer Register Latch 12 Delayed
UACC0	Signal from latch which specifies use of the Instruction Counter contents for the next instruction address
UTR	Signal from latch which specifies use of operand address to replace instruction address in Instruction Counter during any Transfer operation or HOP operation
V	Letter symbol suffix, when added to signal name, indicates a voted output
VIN	Voter Inverter

<u>SYMBOL</u>	<u>DEFINITION</u>
Voter	Circuit with three 2-state inputs; output is the same as majority of inputs
VOY	Latch which decodes Divide or Multiply instructions
V1	+6 VDC
V3	-3 VDC
V4MOD1 thru V4MOD7	+6 VDC, module switching
V20	+20 VDC
WDA	W Clock Pulse Driver to Data Adapter
WF	W Clock feedback signal to other Clock Drivers
WN	Inverse signal of W1 thru W8
W1 thru W8	W Clock Pulse Drivers 1 thru 8
XDA	X Clock Pulse Driver to Data Adapter
XF	X Clock feedback signal to other Clock Drivers
XN	Inverse signal of X1 thru X8
XOR	Programmed Exclusive OR operation
X1 thru X8	X Clock Pulse Drivers 1 thru 8
YDA	Y Clock Pulse Driver to Data Adapter
YF	Y Clock feedback signal to other Clock Drivers
YN	Inverse signal of Y1 thru Y8
Y1 thru Y8	Y Clock Pulse Drivers 1 thru 8
ZDA	Z Clock Pulse Driver to Data Adapter
ZF	Z Clock feedback signal to other Clock Drivers
ZDHN	Zero during a Multiply iteration if delta 2 equals zero
ZDLN	Zero during a Multiply iteration if delta 1 equals zero
ZER	Latch which checks for non-zero condition of the Accumulator contents
ZN	Inverse signal of Z1 thru Z8
Z1 thru Z8	Z Clock Pulse Drivers 1 thru 8

2

2

2

INDEX

- A
- Accumulator register 2-6, 2-128
 Address register 2-4, 2-93
 Add-subtract (A/S) circuit 2-125
 Add-subtract logic 2-8, 2-126
 And latch 2-17
 Arith control 2-119
 Arithmetic element 2-4, 2-117
- B
- B latch 2-125
 Bit time, identification 2-29
 Buffer-amplifier 2-23
 Buffer registers 2-59
- C
- Carry (C) latch 2-125
 Change data sector (CDS) latch . . . 2-101
 Clock drivers 2-26
 Clock generator 2-13
 Command circuit 2-141, 2-169
 Computer assemblies 1-2
 Computer, general handling 5-5
 Computer operations 2-176
 Computer organization 2-1
 Computer single step (CST)-
 gate 2-108
 operation 2-100
 Control circuit 2-29
 Core array 2-38, 2-40
- D
- Data control element 2-4, 2-85
 Data module register 2-4
 Data sector register 2-4, 2-95
 Delay circuits 2-110
 Delay lines 2-174
 Description-
 electrical 1-6
 structural 1-2
 Detectors, disagreement 1-6
 Division-
 arithmetic registers 2-169
 command circuit 2-169
 computer method 2-164
 logic circuits 2-168
- D (cont)
- nonrestoring method 2-161
 restoring method 2-161
 timing circuits 2-169
- E
- E-I drivers 2-45
 Equipment, purpose of 1-1
 Error detectors 2-54
 Execute modify (EXM) latch 2-101
 Execute modify delay (EXMD)
 latch 2-101
- F
- Forced clear and add (FCD)
 gate 2-104
 Ferrite core, toroidal 2-32
- H
- Hi X decoder 2-62
 Hi Y decoder 2-63
 HOP-
 constant serializer 2-16
 latch 2-104
 save circuit, automatic 2-108
- I
- Inhibit driver 2-51
 Instructions-
 ADD 2-17
 AND 2-17
 CDS 2-18
 CLA 2-20
 DIV 2-16
 EXM 2-19
 HOP 2-16
 MPH 2-17
 MPY 2-16
 PIO 2-17
 RSU 2-18
 SHF 2-18
 STO 2-17
 SUB 2-16
 TMI 2-18
 TNZ 2-17
 TRA 2-17
 XOR 2-17

INDEX (Cont)

I (cont)

Instruction counter 2-4, 2-110
 Instruction module register 2-4
 Instruction organization. 2-13
 Instruction sector register. 2-4, 2-95
 Interconnection-
 Computer - ATOM. 3-12
 Computer - data adapter. 3-10
 Computer - LVDC - ME 3-11
 Interface. 3-1
 Interface listing 3-2
 Interrupt control 2-5, 2-105

L

Logic-
 MIB diagrams 10-1
 section 1-2
 Logic description-
 addition (ADD) 2-126
 and (AND) 2-128
 clear and add (CLA) 2-126
 exclusive or (XOR) 2-128
 reverse subtraction (RSU) 2-126
 subtraction (SUB) 2-126
 shift (SHF) 2-129
 Lo X and Lo Y decoders 2-62

M

Manual, purpose of. 1-1
 Memory-
 address decoders 2-60
 address drivers 2-45
 buffer register 2-4, 2-59
 buffer select latches 2-83
 clock drivers. 2-2, 2-44
 control element 2-2, 2-140
 element. 2-2, 2-32
 error monitors. 2-80
 mode and module select 2-2, 2-70
 module registers 2-74
 module select control
 circuit 2-74
 module select gates 2-77
 section 1-2
 select and error monitor 2-2, 2-77
 sense amplifier 2-53
 sync timing 2-2, 2-69
 timing and sync select. 2-2

M (cont)

Multiplication-
 arithmetic registers 2-144
 command circuit 2-143
 computation cycle. 2-158
 logic circuits. 2-141
 multiplicand register 2-144
 multiplier register 2-146
 partial product register. 2-147
 phase counter 2-143
 process 2-137
 product-quotient register. 2-8, 2-144
 timing circuits. 2-142
 Multiply-divide element. 2-4, 2-131
 Multiply formula, derivation 2-131

O

Operation code. 2-11
 Operation code register. 2-4, 2-97
 Operation decoders. 2-8, 2-101
 Oscillator 2-15

P

Page assembly. 1-2
 Parity check circuits. 2-80
 Parity counter 2-4, 2-90
 Phase generator. 2-11, 2-29
 Printed circuit cables 1-2
 Program control element. 2-4, 2-91
 Process input-output (PIO)
 gate 2-104

R

Recirculate accumulator
 (RAC) latch 2-123
 Reference designators 1-2
 Repair 9-1
 Replacement-
 page assembly 9-1
 memory assembly. 9-5
 Run latch. 2-105

S

Seven, seven, seven, or seven,
 seven, six (SS) gate 2-104

INDEX (Cont)

S (cont)

Seven, seven, five (SSF)	
gate	2-104
Shift and process input-output	
(SAPO) gate	2-104
Shift (SHF)-	
gate	2-104
LSD-1	2-129
LSD-2	2-129
MSD-1	2-131
MSD-2	2-131
Shipment, preparation for	5-3
Signal tracing	10-2
Start-stop control	2-5, 2-105
Store multiplicand (STMD)	
latch	2-110
Store (STO) gate	2-104
Store not (STON) gate	2-104
Storage, preparation for	5-1
Syllable select	2-4, 2-67

T

Temperature control voltage	
(TCV) regulator	2-55
Test equipment, special	4-1
Test equipment, standard	4-1
Time B and C (TBC) latch	2-31
Timing element	2-2, 2-13
Timing gate generator	2-26
Timing logic	2-24
Timing organization	2-11
Tools, special	4-1
Transfer register	2-4, 2-85
Transfer register control	
latches	2-88
Triple modular redundancy	1-6
TTL latch	2-104
Time to transfer (TTT) gate	2-105

U

Use, preparation for	5-1
--------------------------------	-----

V

Voters	1-6, 2-175
Voting element	2-6, 2-175

W

Word organization	2-10
-----------------------------	------

1

2

3