IBM Field Engineering

Theory of Operation

System/360 Model 50

Capacitor Read Only Storage

SY22-2823-0

PREFACE

This manual describes the theory of microprogramming, logic, and the operation of Capacitor Read Only Storage in the 2050 Processing Unit.

The reader should have a basic knowledge of the data flow paths and functional units of the 2050 Processing Unit.

Fourth Edition (May, 1969)
This manual, Form Y22-2823-0 obseletes Form Z22-2823-2 and Z22-2823-1. Except for the removal of the IBM Confidential classification, this manual is identical to Form Z22-2823-2. Changes are periodically made to the specifications herein; any such changes will be reported in subsequent revisions or FE Supplements.

This manual has been prepared by the IBM Systems Development Division, Product Publications, Dept. B96, Poughkeepsie, N.Y. 12602. A form is provided at the back of the manual for readers' comments. If the form has been removed, comments may be sent to the above address.

Specifications contained herein are subject to change from time to time. Any such change will be reported in subsequent revisions or FE Supplements.

CONTENTS

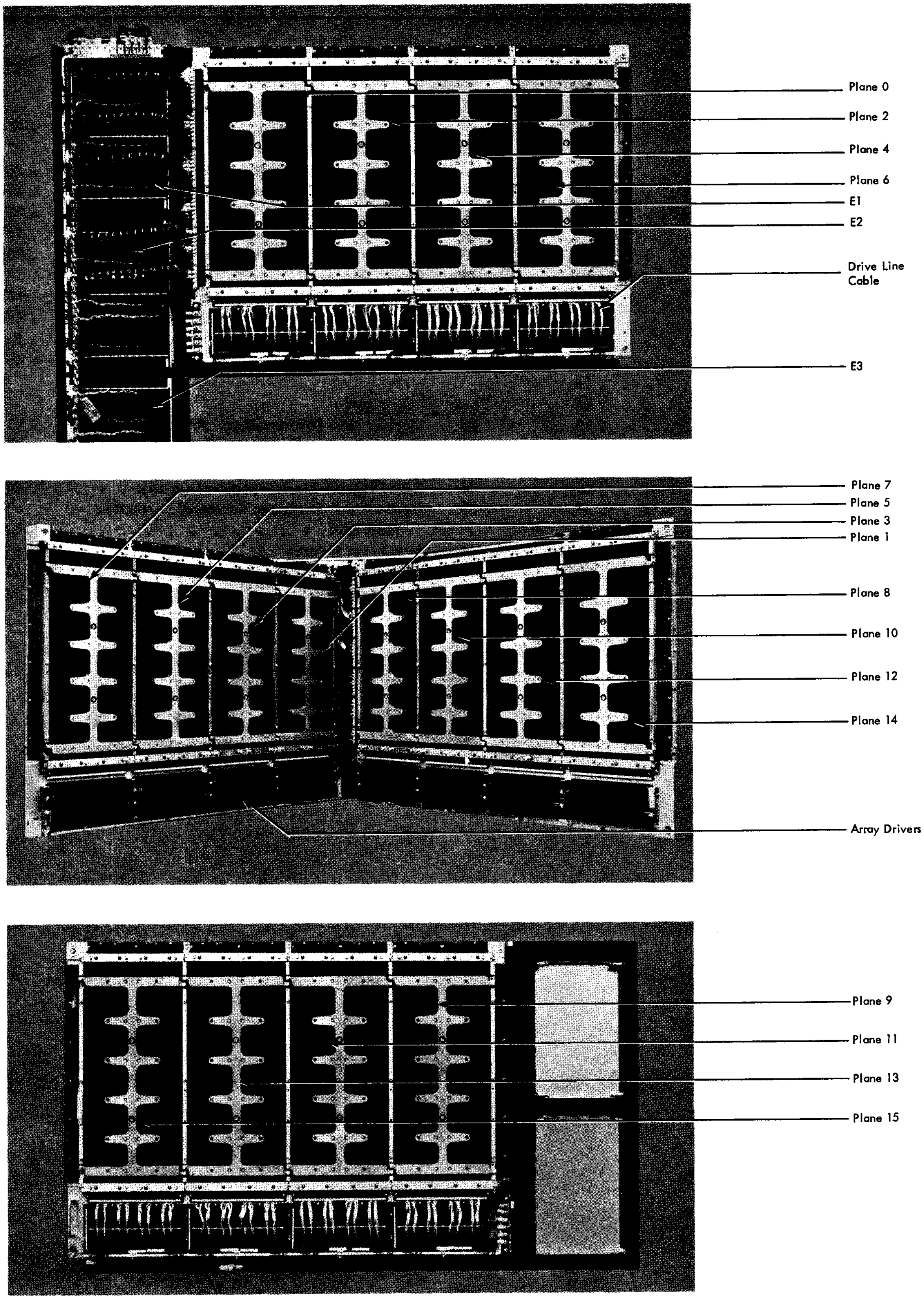
ILLUSTRATIONS

Figure Title Page

Introduction

Functional Units

Theory of Operation



CAPACITOR READ ONLY STORAGE

- IBM System/360 Model 50 is controlled by a microprogram.
- A microprogram word controls each machine cycle.
- System control lines and gates are controlled by fields in the microprogram word.
- Capacitor Read Only Storage (CROS) contains the microprogram.
- CROS contains 2816, 90 bit words.
- Bit patterns of CROS words are determined by the presence or absence of capacitors.
- CROS words can be changed by replacing printed circuit sheets.

Microprogram

Capacitor Read Only Storage (CROS) is the storage device that contains the IBM System/360 Model 50 microprogram. The microprogram is stored in 2816 words of 90 bits each. Each 90-bit word controls the gates and control lines of the system for one 500 ns machine cycle; gating for each functional unit is controlled by the bit combination within a field of the microprogram word. For example, 001 in bit positions 65-67 of the microprogram word gates the R register to the right adder input.

Figure 1 shows the relationship of CROS to the system clock and control lines; only the clock can operate independently of the microprogram words from CROS.

CROS

Capacitor Read Only Storage is a storage device for predetermined information that can be nondestructively read out. The bit pattern of a CROS word is determined by the presence or absence of capacitors within the CROS hardware; a CROS word cannot be changed by customer programs. However, the Customer Engineer can change the information in the CROS words by replacing printed circuit sheets. Capacitor Read Only Storage replaces most of the system control circuits and introduces a flexibility to machine design not previously available in control hardware. This flexibility allows making control circuit changes for a special feature by replacing printed circuit sheets in CROS.

CROS ELECTRICAL THEORY

- A capacitor represents a bit of information.
- One drive line drives the capacitors for one CROS word.
- Capacitors couple the drive lines to sense lines.
- There is one sense amplifier for each bit position of the CROS word.
- Sense amplifier outputs are stored in Read Only Storage Data Registers (ROSDR).
- Drive lines are selected by decoding a Read Only Address Register (ROAR).

Drive and Sense Lines

Capacitor Read Only Storage uses a capacitor between drive and sense lines to represent a bit of information. Figure 2 shows a basic capacitor storage matrix. Only one driver at a time may be active. If driver 1 is made active, the voltage shift on the drive line is capacity-coupled to the A-sense line. The sense amplifier amplifies the voltage shift and produces an A-bit output. If driver 3 is made active, the drive line voltage shift is capacity-coupled to the B and D sense lines, producing a B-bit and a D-bit output.

Some unwanted capacitive coupling exists in a capacitor storage matrix of this type. When driver 1 is made active, capacitor C1 couples the voltage shift to the A-sense line, capacitor C2 couples the voltage shift on the A-sense line to the driver 2 drive line, and capacitor C3 couples the voltage shift from the driver 2 drive line to the C-sense line. Because any unwanted signal must pass through at least three elements in cascade, the unwanted signal will be greatly decreased. The threshold of the sense amplifier is such that the desired signal is amplified while the unwanted, weakened signal is rejected.

Differential Sense Amplifiers

To further reduce unwanted signals and noise, each sense line is divided into a pair of sense lines which feed a differential amplifier that amplifies the difference between the two input lines. Each drive line is capacitor-coupled to one line of each pair of sense lines (Figure 3). A capacitor that causes a positive output from the differential sense amplifier represents

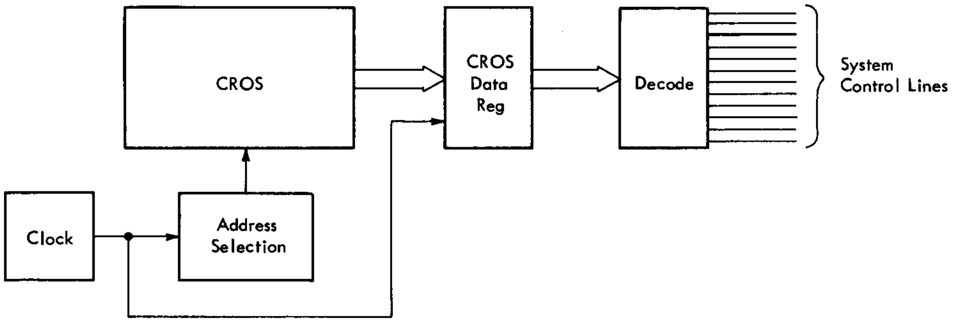


FIGURE 1. CROS DATA FLOW

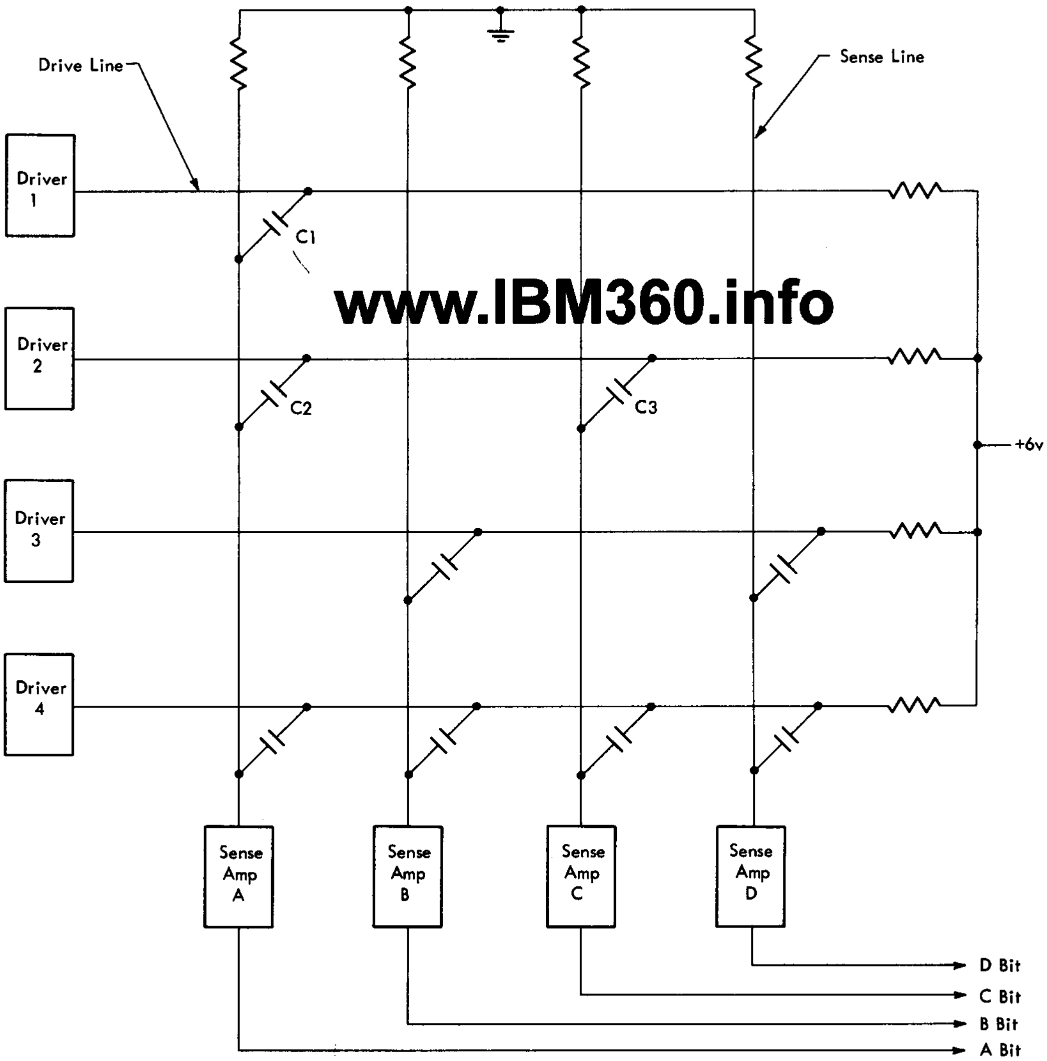


FIGURE 2. BASIC CAPACITOR MATRIX

a zero bit, while a capacitor that causes a negative output represents a one bit. Thus, each drive line has the same capacitive load regardless of the bit configuration.

Balance Lines

An extra line, called a balance line, is added for each drive line. The balance line is not connected to a driver; instead, it is connected to the drive line voltage source at one end and allowed to float at the other end. A capacitor is connected from the balance line to whichever sense line is not connected to the drive line. Thus, each sense line has the same capacitive load regardless of the bit configuration.

Data Register

The Read Only Storage Data Register (ROSDR) stores the sense amplifier outputs (Figure 4). ROSDR holds the CROS word available to the decode circuits for one machine cycle.

Address Register

The Read Only Address Register (ROAR) stores the address of the next CROS word. The address in ROAR is decoded to select one CROS driver. The address sent to ROAR is controlled by the CROS word for the machine cycle being executed.

THEORY OF MICROPROGRAMMING

- The processing unit is controlled by microprogram words.
- Microprogram words are grouped into routines.
- Microprogram routines control processing of customer program instructions.
- Each machine cycle is controlled by a microprogram word.

Microprogramming is the use of internal program words, built into the machine, that control the gates and control lines during each machine cycle. For example, a customer program instruction of Add RR Format is decoded to cause CROS to read out the first word of a microprogram routine. The microprogram routine fetches general purpose registers R1 and R2 from local storage, adds them together, and stores the answer in R1. Each machine cycle necessary for the operation is controlled by a microprogram word.

Microprogram Word

- Each microprogram word controls the address of the next microprogram word.
- Fields within the microprogram word control data flow gates and control lines.

The microprogram word is the set of bits read out of CROS and stored in the Read Only Storage Data Register (ROSDR). Figure 5 shows the breakdown of the microprogram word.

Control Fields: The microprogram word is divided into control fields that control the data flow gates in the computer and the addressing of the next CROS word.

Micro-Orders: A micro-order represents a specific bit combination within a control field. A micro-order is decoded into one or more control lines or gates. Micro-orders are written as a symbolic representation of the control lines or gates that result from decoding the control field. For example, the micro-order that gates the T latches to the A register has the symbolic representation T→A.

Controlling Data Flow With Microprogramming

Figure 6 shows a multiply operation on a fictitious CROS-controlled computer. Assume that the customer-program instruction to multiply register A times register B and store the product in register C has been decoded and has addressed CROS word 0000, the start of the multiply routine. Each CROS word controls the data flow for one machine cycle as follows:

0000 Zeros are sent through the adder and stored in register C. CROS word 1100 is addressed.

1100 Register A is sent through the adder and checked for all zeros. If register A is all zeros, the multiply operation is over and the branch on zero test causes CROS word 0011 to be addressed. If register A is not zero CROS word 0010 is addressed.

0010 Register B and register C are added together and stored in register C. CROS word 0100 is addressed.

0100 Register A is reduced by one by adding register A to complemented zeros with a no carry insert. If the result is zero, the multiply

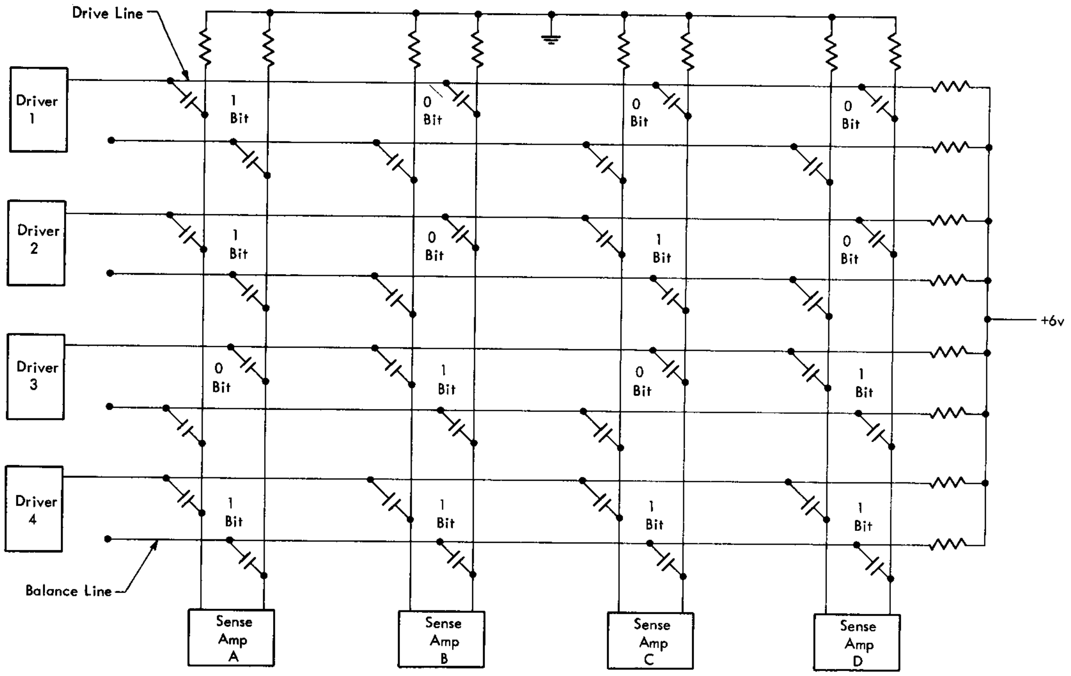


FIGURE 3. BASIC 4 X 4 CROS MATRIX

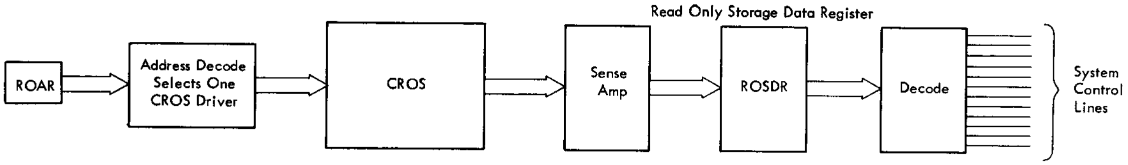

Read Only Storage Data Register
ROAR
Address Decode Selects One CROS Driver
CROS
Sense Amp
ROSDR
Decode
System Control Lines

FIGURE 4. BASIC CROS

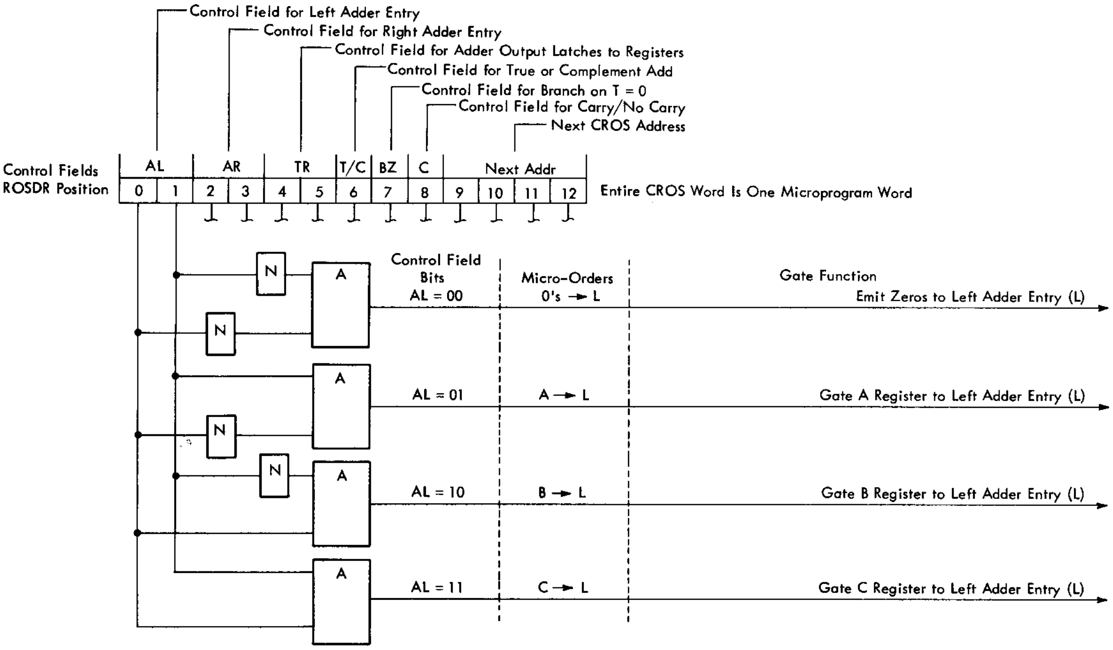
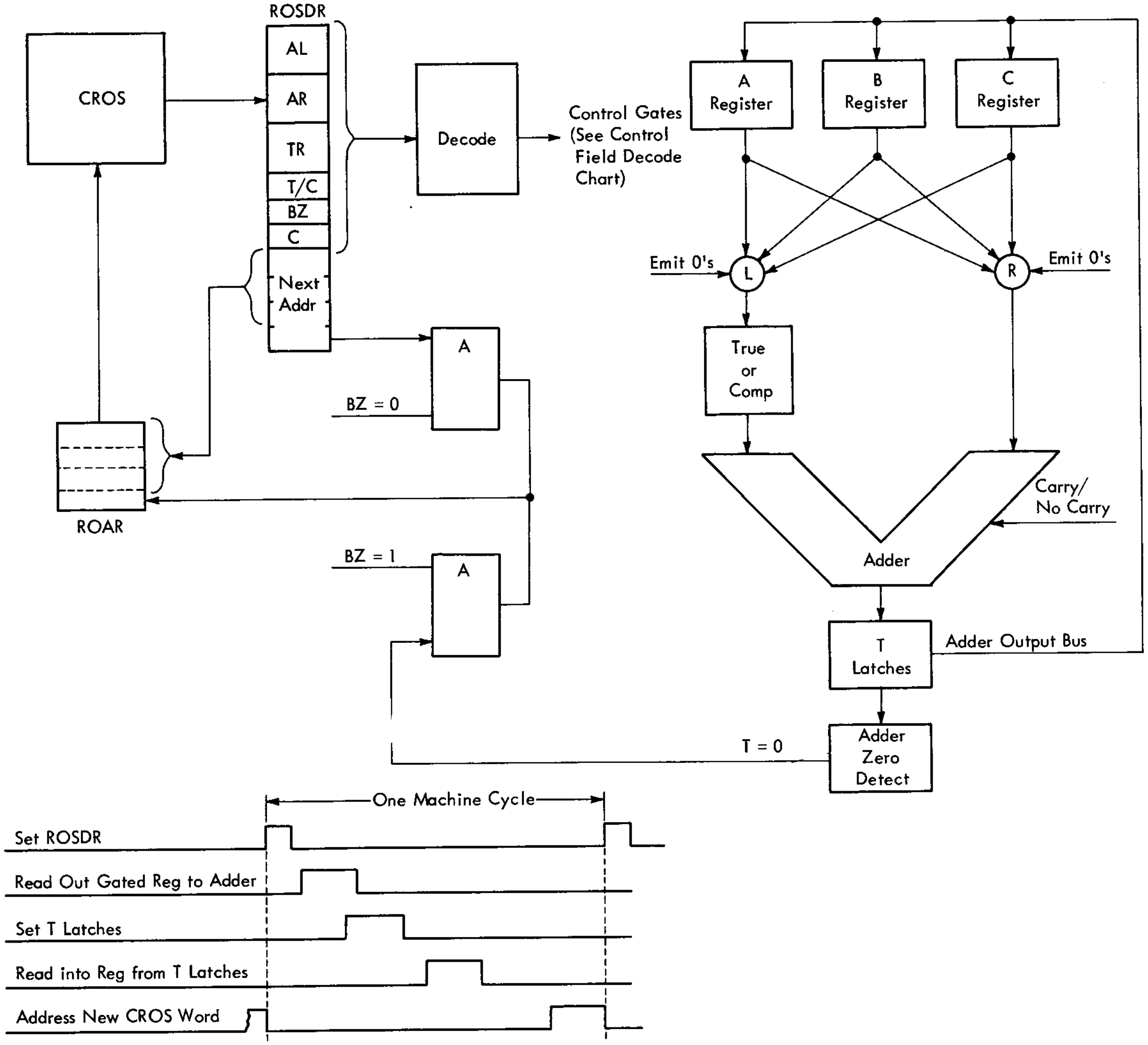
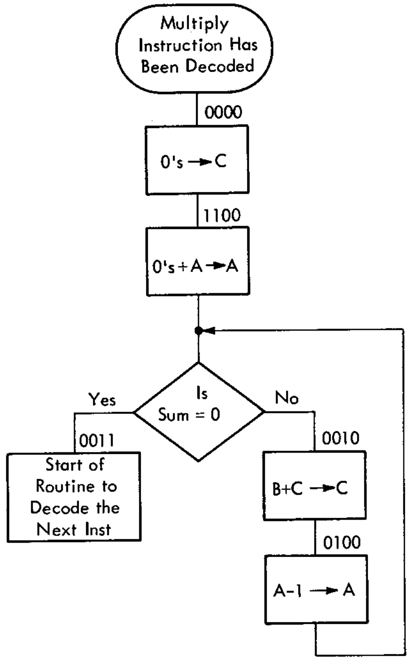

Control Field for Left Adder Entry
Control Field for Right Adder Entry
Control Field for Adder Output Latches to Registers
Control Field for True or Complement Add
Control Field for Branch on T = 0
Control Field for Carry/No Carry
Next CROS Address
Control Fields
ROSDR Position
AL
AR
TR
T/C
BZ
C
Next Addr
0
1
2
3
4
5
6
7
8
9
10
11
12
Entire CROS Word Is One Microprogram Word
N
N
N
N
A
A
A
A
Control Field Bits
Micro-Orders
Gate Function
AL = 00
0's → L
Emit Zeros to Left Adder Entry (L)
AL = 01
A → L
Gate A Register to Left Adder Entry (L)
AL = 10
B → L
Gate B Register to Left Adder Entry (L)
AL = 11
C → L
Gate C Register to Left Adder Entry (L)

FIGURE 5. MICROPROGRAM WORD FOR A FICTITIOUS COMPUTER



Control Field Decode Chart

Field	Value	Micro-Order	Function
AL	00	0's→L	Emit Zeros to L
AL	01	A →L	Gate A Register to L
AL	10	B →L	Gate B Register to L
AL	11	C →L	Gate C Register to L
AR	00	0's→R	Emit Zeros to R
AR	01	A →R	Gate A Register to R
AR	10	B →R	Gate B Register to R
AR	11	C →R	Gate C Register to R
TR	00	NOP	No Gating from T
TR	01	T →A	Gate T Latches to A Register
TR	10	T →B	Gate T Latches to B Register
TR	11	T →C	Gate T Latches to C Register
T/C	0	TA	True Add L Adder Entry
T/C	1	CA	Complement Add L Adder Entry
BZ	0	NB	No Branch - Use Last Position ROSDR
BZ	1	BTO	Branch Test - Insert 1 in ROAR if T = 0
C	0	NC	Insert No Carry
C	1	C	Insert Carry


Multiply Instruction Has Been Decoded
0000
0's →C
1100
0's+A→A
Is Sum = 0
Yes
No
0011
Start of Routine to Decode the Next Inst
0010
B+C →C
0100
A-1 →A

CROS Word Address	Microprogram Words							Resulting Micro-Orders
	AL	AR	TR	T/C	BZ	C	Next Addr	
0000	00	00	11	0	0	0	1100	0's→L, 0's→R, T→C, TA, NB, NC
1100	01	00	01	0	1	0	0010	A→L, 0's→R, T→A, TA, BTO, NC
0010	10	11	11	0	0	0	0100	B→L, C→R, T→C, TA, NB, NC
0100	00	01	01	1	1	0	0010	0's→L, A→R, T→A, CA, BTO, NC
0011	XX	XX	XX	X	X	X	XXXX	Next Instruction Decode Routine

FIGURE 6. MULTIPLY OPERATION ON A FICTITIOUS CROS-CONTROLLED COMPUTER

operation is over and CROS word 0011 is addressed. If the result is not zero CROS word 0010 is addressed.

0011 This word starts the routine to decode the next customer-program instruction.

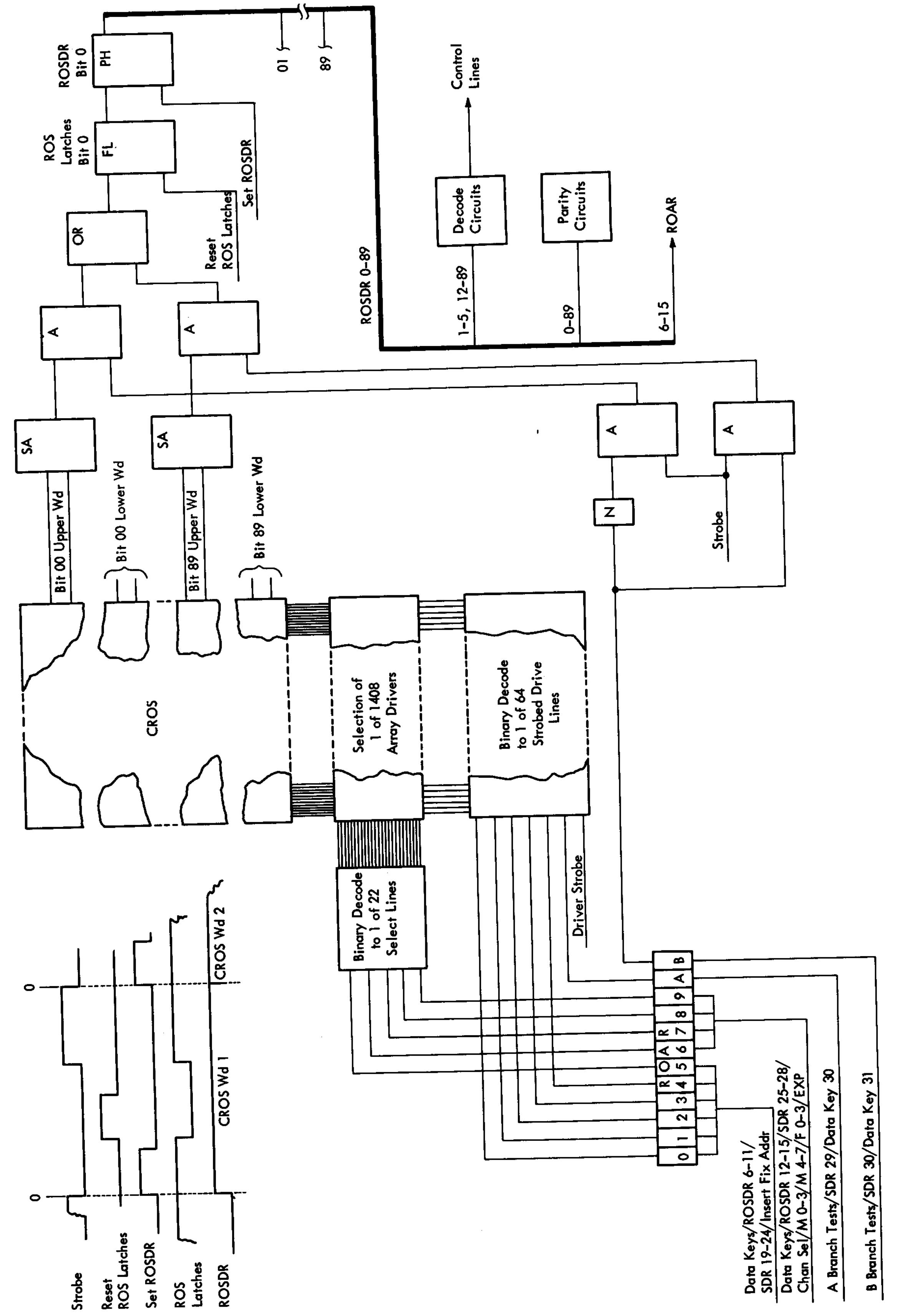


FIGURE 7. CROS FUNCTIONAL UNITS

- A 12-position address register (ROAR) controls which CROS word is read out.
- ROAR positions 5-9 control the select lines..
- ROAR positions 0-4 and A control the drive lines.
- Select and drive lines control the array drivers.
- Array drivers read two CROS words, at the same time, to the sense amplifiers.
- ROAR position B gates one of the two CROS words to the ROS latches.
- ROS latches hold the CROS word available to the ROSDR latches.
- ROSDR latches hold the CROS word available to the parity and decode circuits.
- Decode circuits decode the CROS word into control lines and gates.
- Parity circuits check the CROS word for missing or extra bits.

Figure 7 shows the CROS functional units and their relationship to each other.

CROS PLANES

- CROS is divided into 16 planes.
- Planes 0-7 are on gate C.
- Planes 8-15 are on gate D.
- Each plane contains 176 CROS words.
- Each CROS word contains 90 bits.

The 2,816 words of CROS are stored in 16 planes. Figure 8 shows the location of the planes on gates C and D. Each plane contains the capacitors, drive lines, balance lines, and sense lines for 176, 90-bit CROS words.

Each plane contains independent drive and balance lines, while the sense lines for all planes feed common sense amplifiers.

Drive and Balance Lines (Bit Plates)

- Drive and balance lines are photo-etched onto replaceable printed circuit sheets (bit plates).

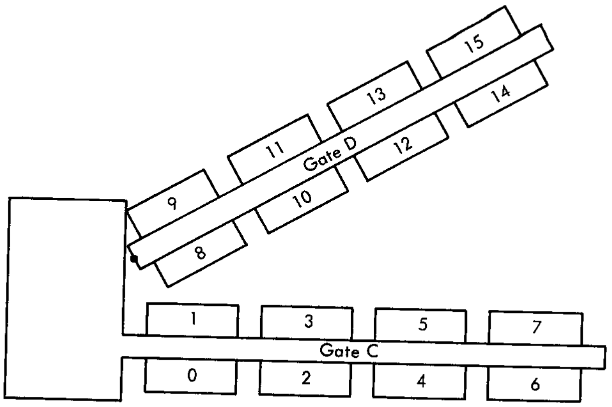


FIGURE 8. CROS PLANE LOCATIONS - TOP VIEW

- The drive and balance lines, a sheet of dielectric, and the sense lines form the bit capacitors.
- The shape of the drive and balance lines form the bit configuration for each bit plate.
- The CROS bit configuration can be changed by replacing bit plates.

The drive and balance lines are photo-etched from a sheet of copper that is bonded to epoxy glass (Figure 9). The resulting epoxy sheet with copper drive and balance lines is called a bit plate. A separate bit plate controls the bit configuration for each CROS plane.

Tabs at the top and bottom of the bit plate are used for electrical connections to the drive and balance lines. The top tabs connect the drive and balance lines to terminating resistors. The bottom tabs connect the drive lines to the driver circuits.

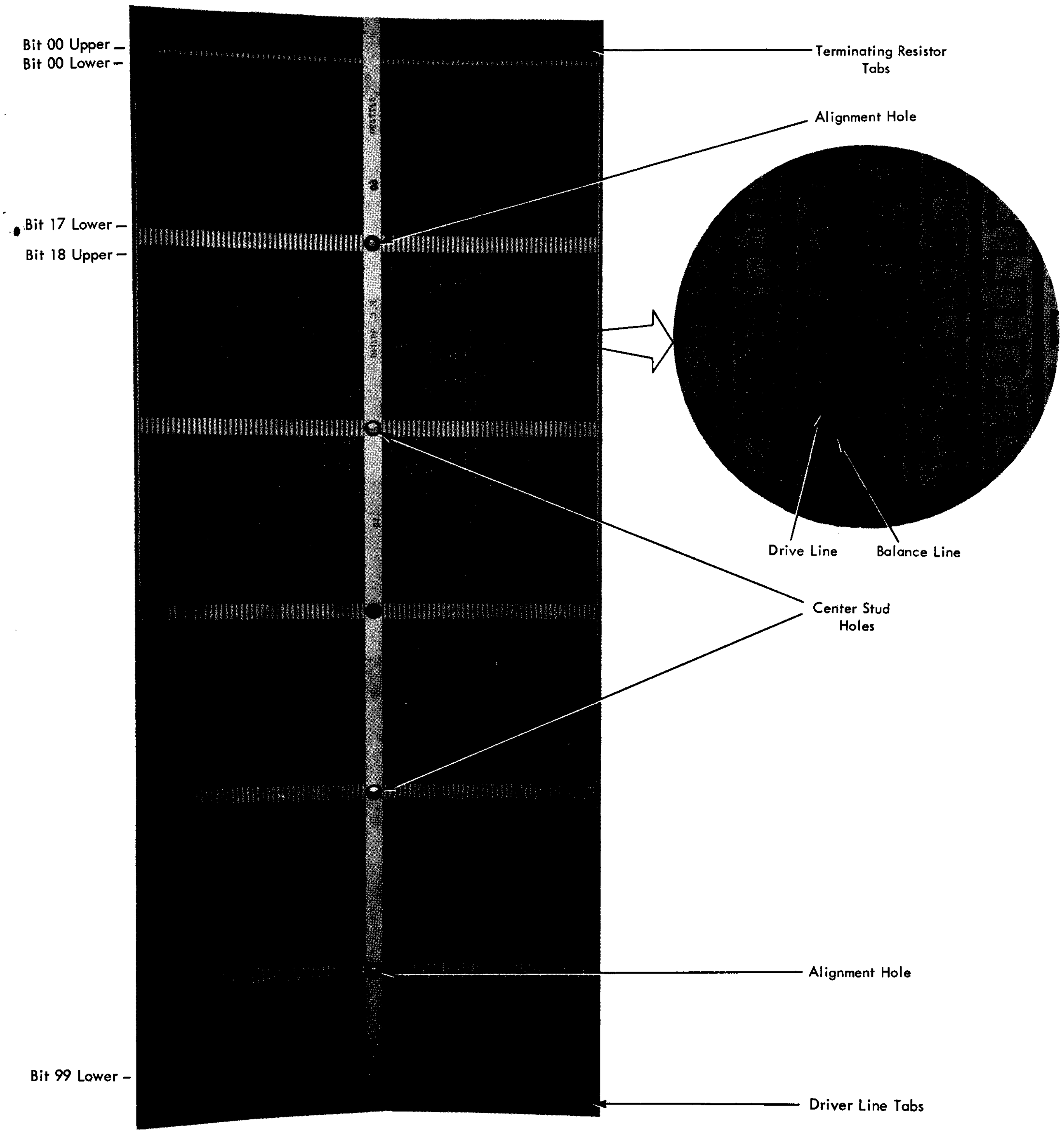
The four holes in the bit plate are used to align the bit plane over the sense plane. Two holes snap over locating studs in the plane, while the other two holes provide clearance for the center studs.

Bit Capacitors

The bit capacitors are formed by sandwiching a sheet of Mylar* between the bit plate and sense lines (Figure 10). Pressure plates hold these pieces firmly together. The Mylar is the dielectric, while the drive, balance, and sense lines become the plates of the capacitors .

Protrusions of the drive and balance lines increase the size of the capacitors to form the bit configuration. The effective capacitor coupling of a drive line to a sense amplifier is equal to C1 minus

* Trademark of E.I. duPont deNemours & Co. (Inc.)



Note: Each number on the left side refers to a pair of lines.

FIGURE 9. BIT PLATE

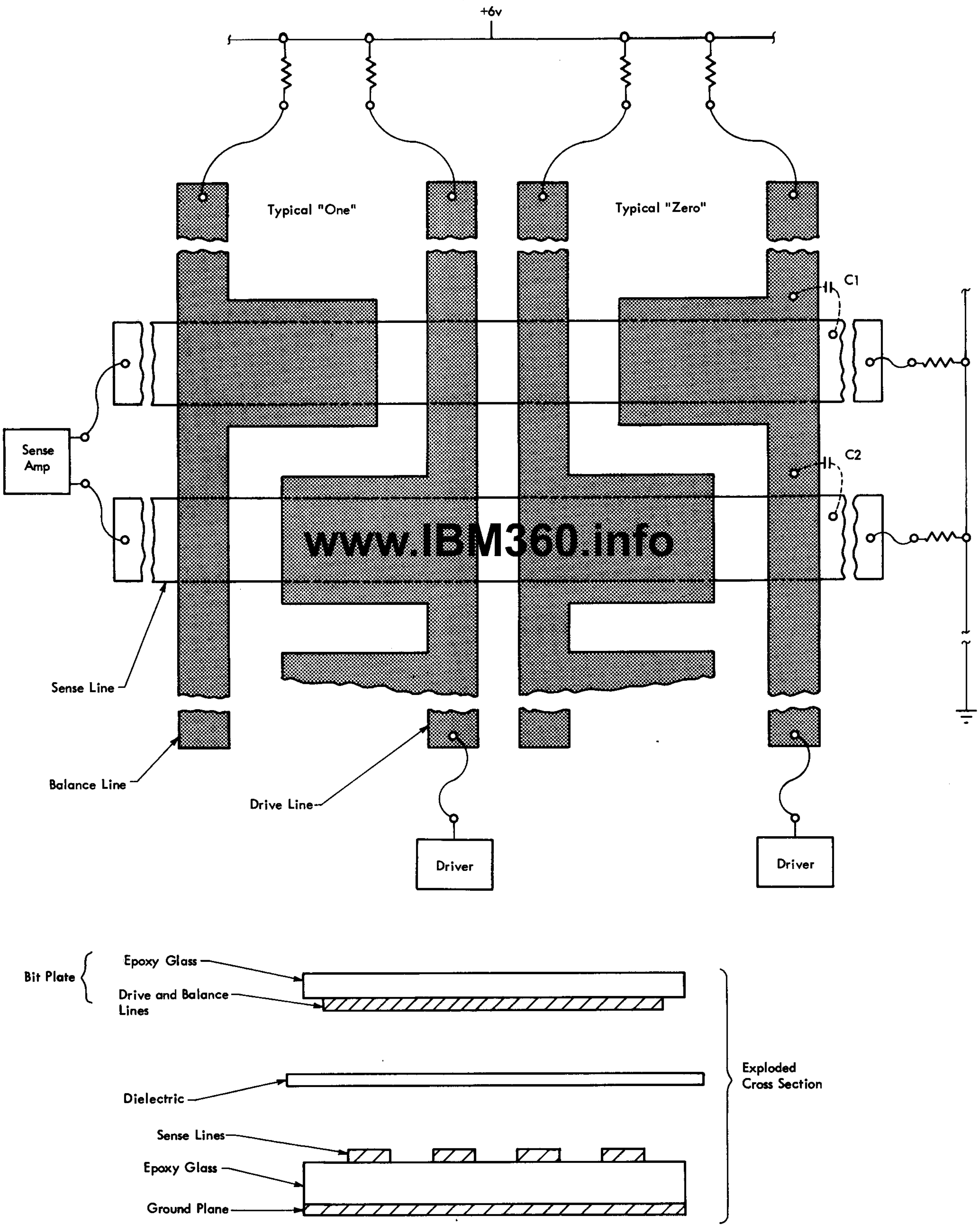


FIGURE 10. BIT CAPACITORS

C2. The size of this effective capacitor is approximately 0.5 picofarads.

The bit configuration within a CROS plane is controlled by the bit plate. Therefore, the microprogram instruction in a CROS word can be changed by replacing the bit plate that contains the word.

Sense Lines

- There are 400 sense lines in each CROS sense plane.
- A pair of sense lines carry the signal for one CROS-word-bit position.
- 360 sense lines read out two 90-bit CROS words simultaneously.
- The lower 40 sense lines are not used.

The sense lines are photo-etched into copper covered epoxy-glass plates (Figure 11). The sense-line plates are permanently mounted to the array gates. Electrical connections from the sense lines to the terminating resistors and sense amplifiers are made with low temperature solder.

There are 400 sense lines in each CROS plane. Two sense lines are required to read out one bit position of the CROS word. One drive line simultaneously reads out two 90-bit CROS words, which use 360 sense lines. The other 40 sense lines are not used in the standard Group 50 CROS, but they are available if needed for RPQ activity.

Figure 12 shows the bit numbering of the sense lines. The top pair of sense lines is bit 0 of the upper word. The next lower pair of sense lines is bit 0 of the lower word. Both the upper and lower words are read out simultaneously. Each sense line is terminated through a resistor to ground.

Figure 13 shows the distribution of a sense line through the planes to the differential sense amplifier. The sense lines through the planes on both sides of a gate are tied together for each bit. The pair of sense lines from each gate is then OR'ed together in the sense amplifier for each bit. Because only one plane has an active drive line for a given ROAR address, the sense amplifier receives only one input signal.

Physical Package

A CROS plane consists of a "sandwich" composed of the sense line board, a dielectric sheet, and a bit plate. These pieces are held firmly together by pressure plates (Figure 14).

A pressure plate, with a sponge-rubber pad, fits over each group of capacitors in the plane. The pressure plates are loosely connected to a pressure frame that is bolted to the gate. Pressure adjusting screws in the pressure frame press the pressure plate against the bit plate. Because the sense lines are on a rigidly mounted board, the pressure-plate pressure holds the bit-capacitor "sandwich" firmly together.

Electrical connections to the bit plate are made through pressure connections. Spring-operated pressure bars press the bit plane tabs against the driver line and resistor card tabs (Figure 14).

SENSE AMPLIFIERS

- Two words are read out and amplified each machine cycle.
- There is a separate sense amplifier for each bit of the two words.
- Each sense amplifier has three stages of amplification.
- The first amplifier stage is a differential amplifier.

The sense amplifiers amplify the voltage difference within a pair of bit sense lines. The first stage of the sense amplifiers (Figures 15 and 16) consists of differential amplifiers, one for gate C sense lines and one for gate D sense lines. Because only one array driver is active for a machine cycle, the sense lines of only one gate carry a signal during a machine cycle. The first stage differential amplifiers amplify the voltage difference within a pair of sense lines and feed the second stage amplifier. The second and third stage amplifier further amplifies the signal and sends the signal to the ROS latches.

The input to the sense amplifier is a pulse which is present on two different input lines. The first stage of the amplifier amplifies the positive or negative difference in voltage between the two input pulses and rejects common mode noise. There are two pairs of inputs and they cannot be used simultaneously.

The differential voltage is further amplified by a shunt feedback stage and a feedback pair. The amplified signal is then applied to the buffer input state of the latch which conditions the signal to be compatible with standard AOI-11 logic.

Standard AOI-11 logic is then used to store the "one" or "zero" until shortly before the next input pulse, at which time the latch is reset.

Capacitors C1 through C11 are used for ac ground or dc blocking only. Capacitors C12 and C13 are used to balance the impedance seen at each input terminal in order to reject the common mode noise. Capacitor C14 is used to extend the band width of the amplifier.

Inductor L1 is used for dc biasing and signal isolation from the six-volt power supply.

Resistors R1 through R16 are used for dc biasing. All transistors are biased for class A operation.

DATA REGISTERS

- ROS latches hold the selected CROS word available for the next machine cycle.

- ROSDR holds the CROS word that controls the present machine cycle.

- 88 ROSDR outgate latches control the decode, parity, and ROAR addressing circuits from 0 ns to 0 ns of a machine cycle.

- 20 ROSDR ingate latches control decode circuits from 130 ns of a machine cycle to 130 ns of the next cycle.

- ROSDR ingate latches allow one CROS word to operate on data during a machine cycle and to store the results in a register during the next machine cycle.

The data registers store the CROS word for use during the machine cycle. The output of the data

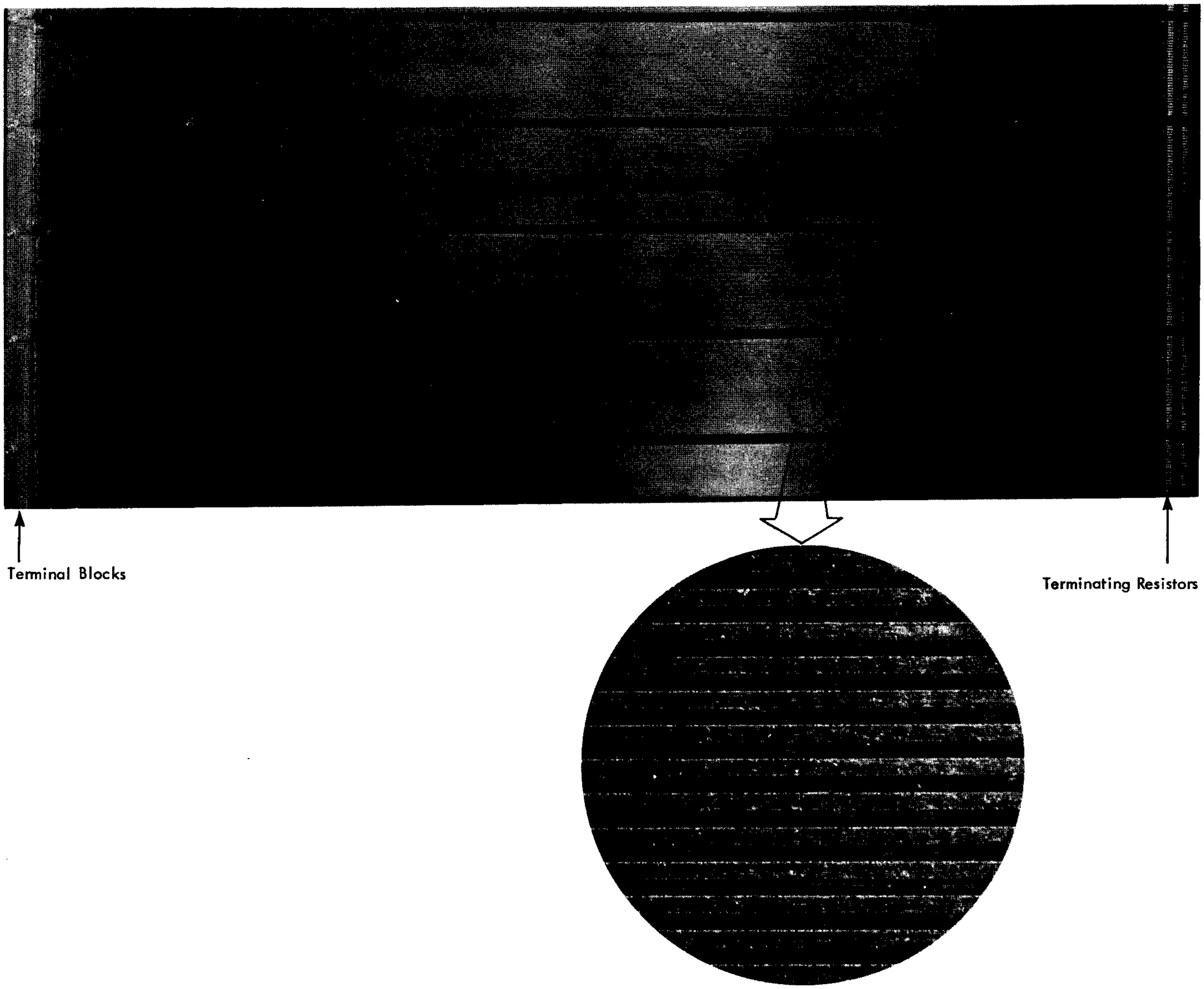


FIGURE 11. SENSE LINES

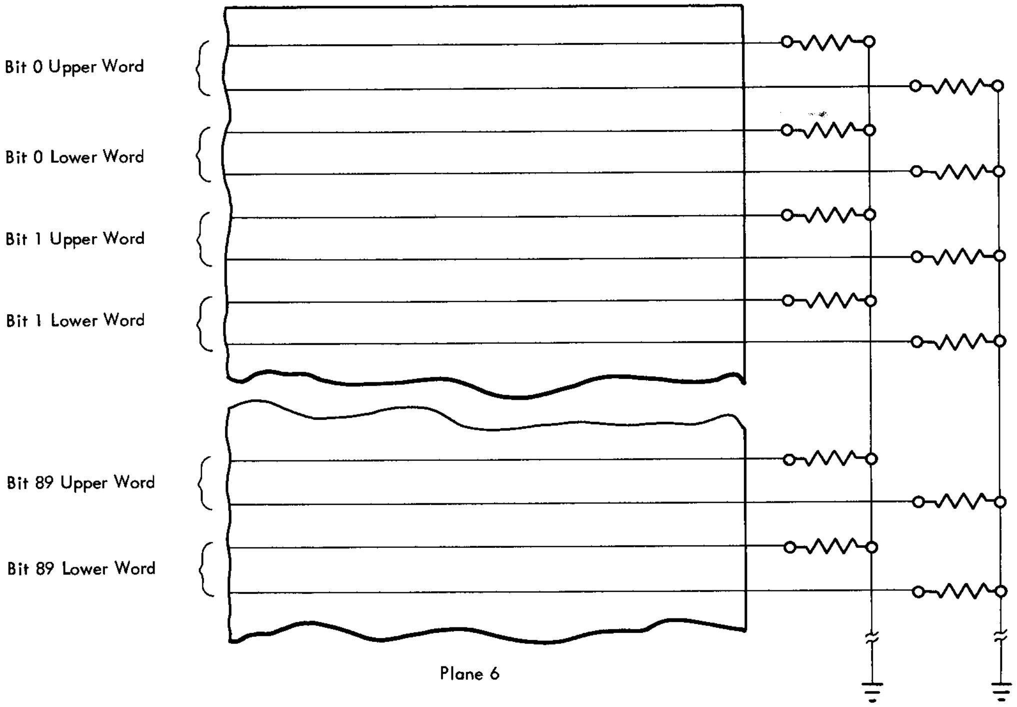


FIGURE 12. SENSE LINE BIT LAYOUT

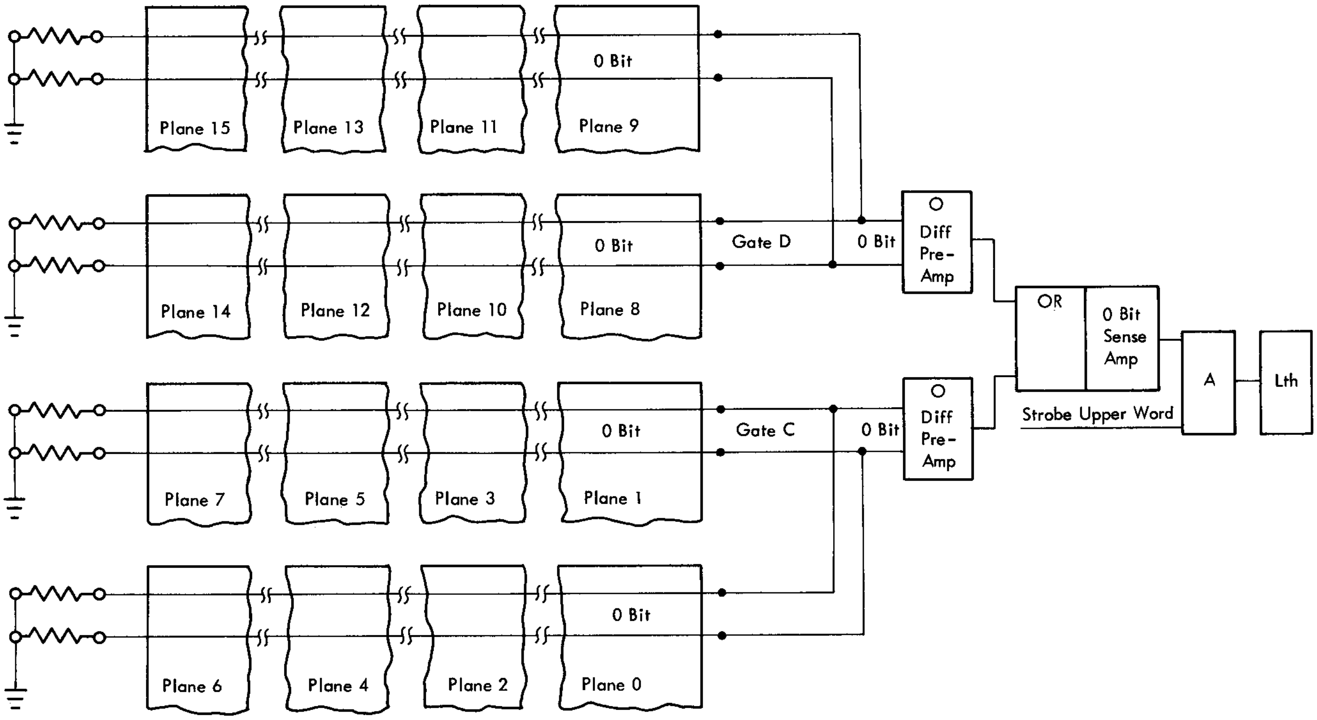


FIGURE 13. SENSE LINE DISTRIBUTION

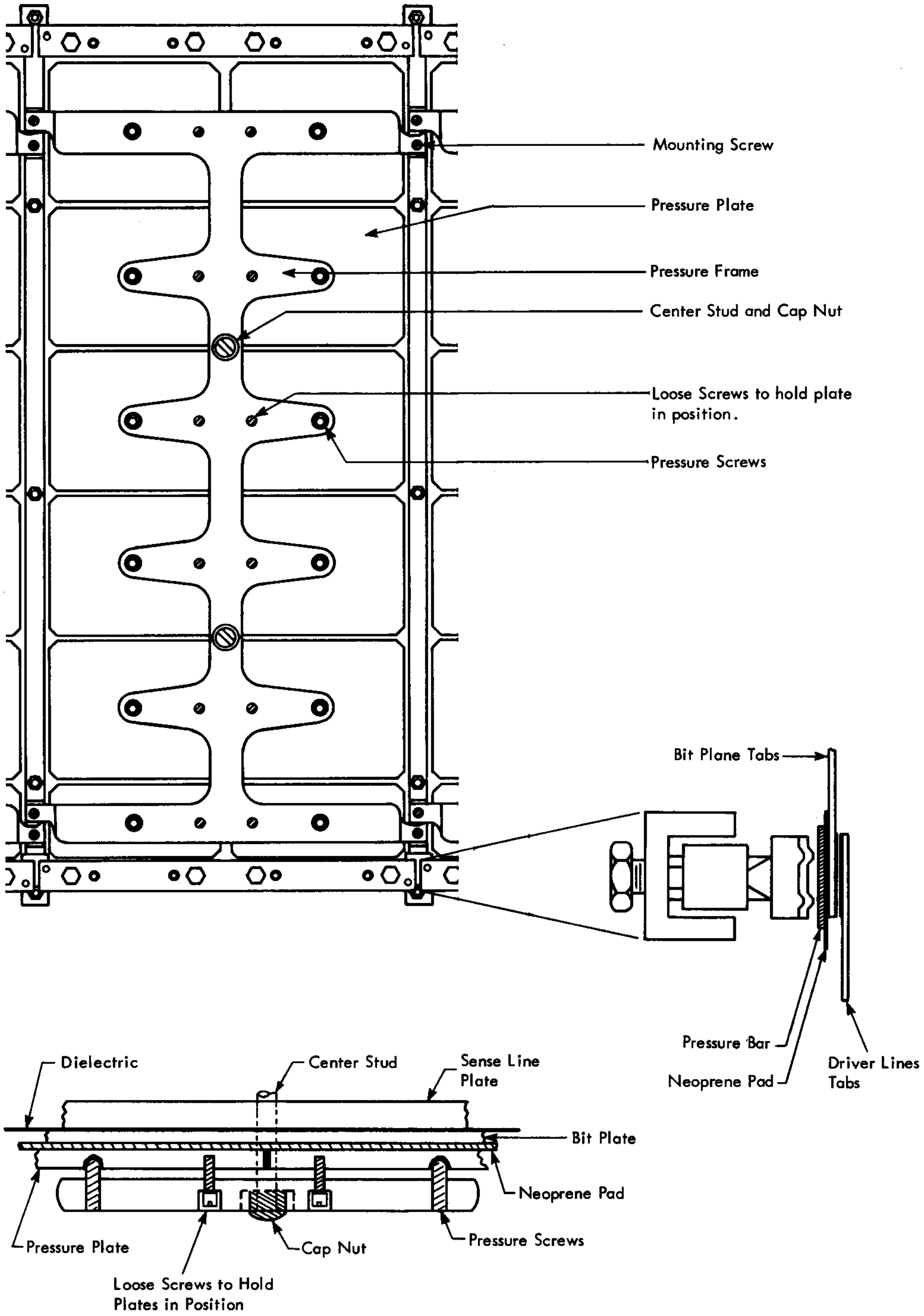


FIGURE 14. CROS PLANE PRESSURE MOUNTING ASSEMBLY

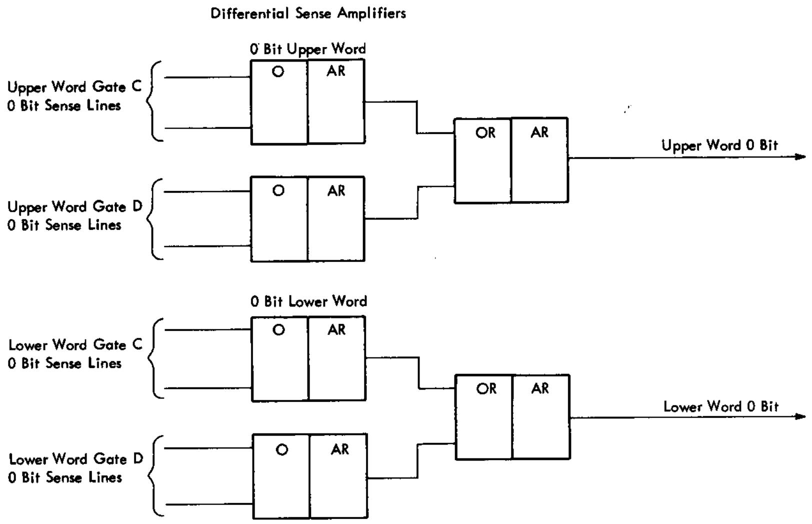


FIGURE 15. SENSE AMPLIFIERS

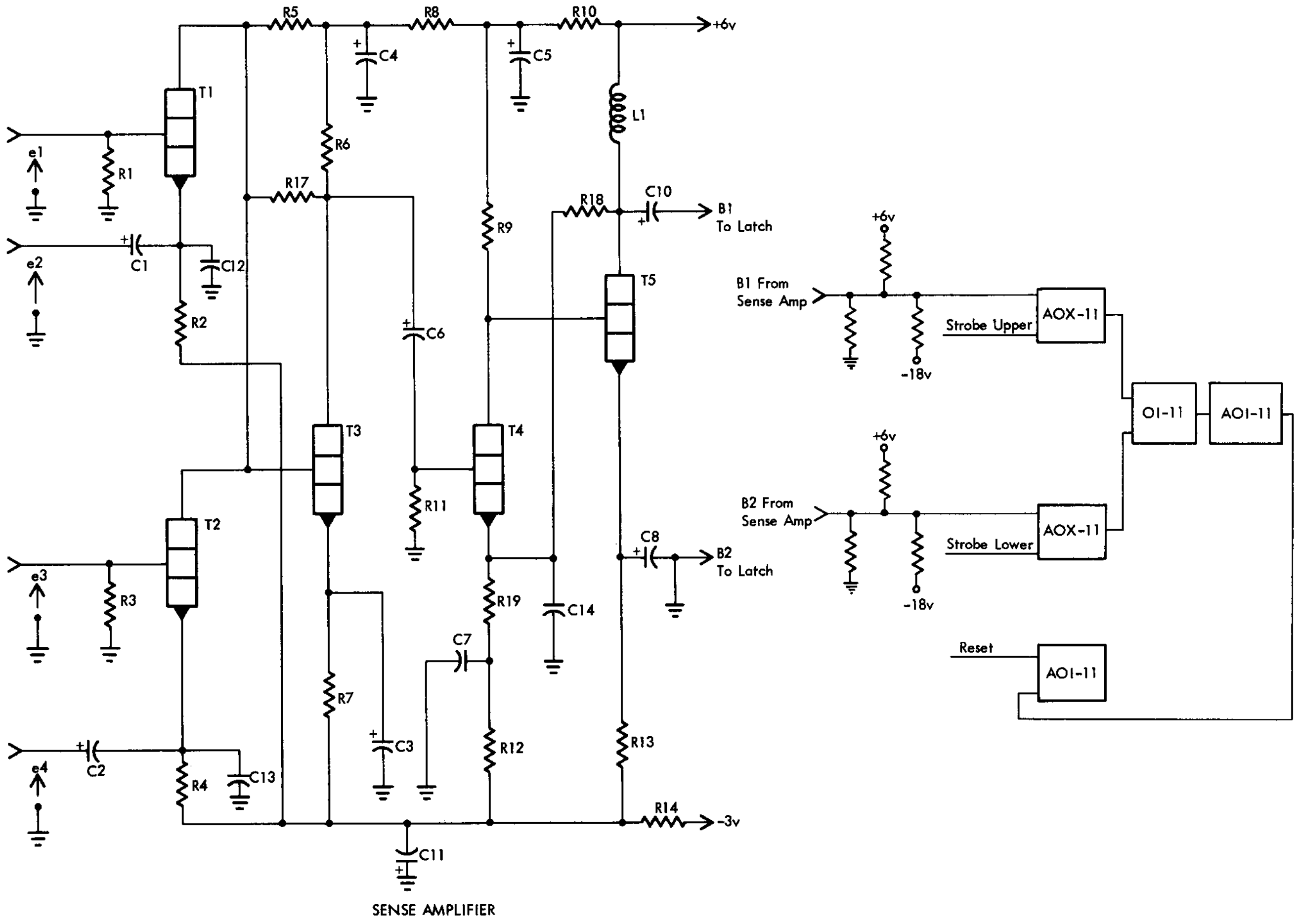


FIGURE 16. SENSE AMPLIFIER SCHEMATIC

registers controls the decode circuits, parity check circuits, and ROAR addressing circuits.

Two CROS words are available at the output of the sense amplifiers, about 3/4 of the way through a machine cycle. The ROAR - B bit position gates one of the two words to the ROS latches (Figure 7).

ROS Latches

There are 90 ROS latches, one for each bit position of the CROS word. The latches are set by strobing either the upper or lower word sense amplifier outputs (Figure 17). The latches are reset at 130 ns of the machine cycle.

The ROS latches hold the CROS word available to set the ROSDR register at 0 ns of the next machine cycle.

ROSDR

ROSDR is divided into two sets of registers, ROSDR outgate registers and ROSDR ingate registers. The ROSDR registers hold the CROS word available to the parity, decode, and ROAR addressing circuits. Figure 18 shows the ROSDR registers and their function for a CPU-mode machine cycle.

Note: Machine cycles are either CPU mode or I/O mode; both cycles are under CROS control. I/O mode cycles are machine cycles that use the processing unit circuits to transfer data to or from an I/O channel. A CPU mode cycle is any cycle in which the I/O channels are not using the processing unit circuits.

Outgate Registers

There are 88 outgate registers, ROSDR positions 00-89. CROS word positions 24 and 83 are spare positions for future expansion and have no ROSDR register positions.

The ROSDR outgate registers are set from the data in the ROS latches at 0 ns of the machine cycle; the registers remain set until 0 ns of the next cycle. The outgate registers are set in three groups.

Position 0 is used for parity checking and is set each machine cycle.

Positions 6-18 control the address of the next CROS word. On special conditions, such as detecting an invalid address, the address of a fixed routine is inserted into ROAR; an Insert Fixed Add line prevents setting these ROSDR outgate positions.

Positions 1-5 and 19-89 hold the information that is to be decoded to control the system control lines and gates. These positions are prevented from setting on an I/O Break In Cycle, the cycle used to switch from CPU mode to I/O mode. The OFF condition of these ROSDR outgate positions is decoded to cause the R register to be stored in a special position of local storage. Storing the R register allows the I/O mode routine to use the R register without destroying CPU-mode information.

Ingate Registers

There are 20 ingate register positions, ROSDR positions 19-23, 28-30, and 40-51. The ingate registers are set from the outgate registers at 130 ns of a

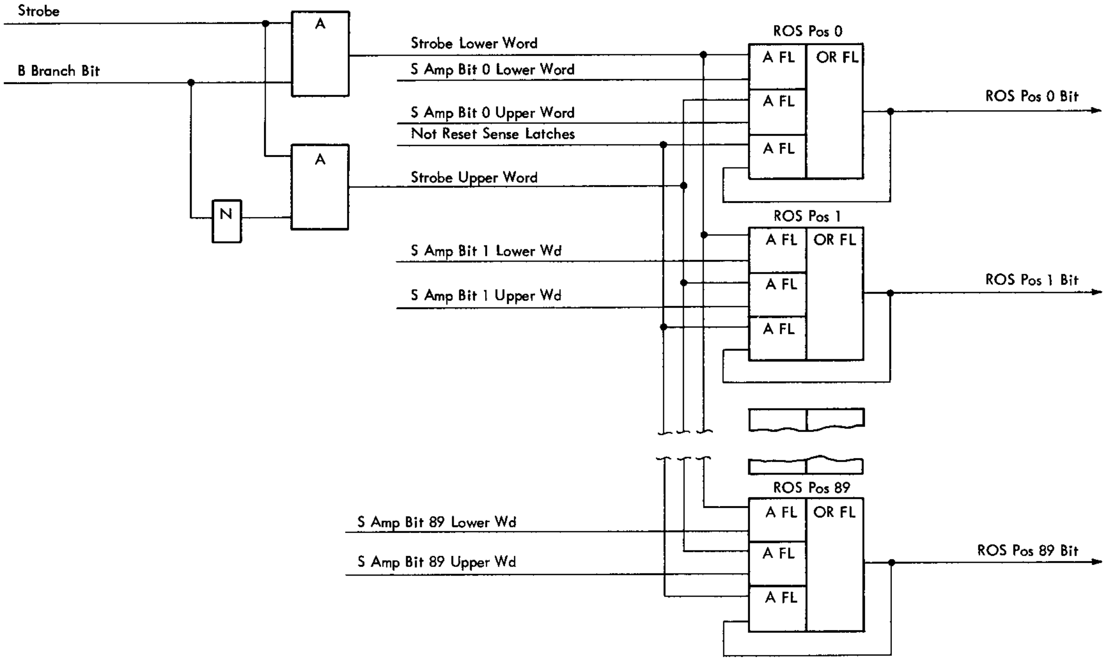


FIGURE 17. ROS LATCHES

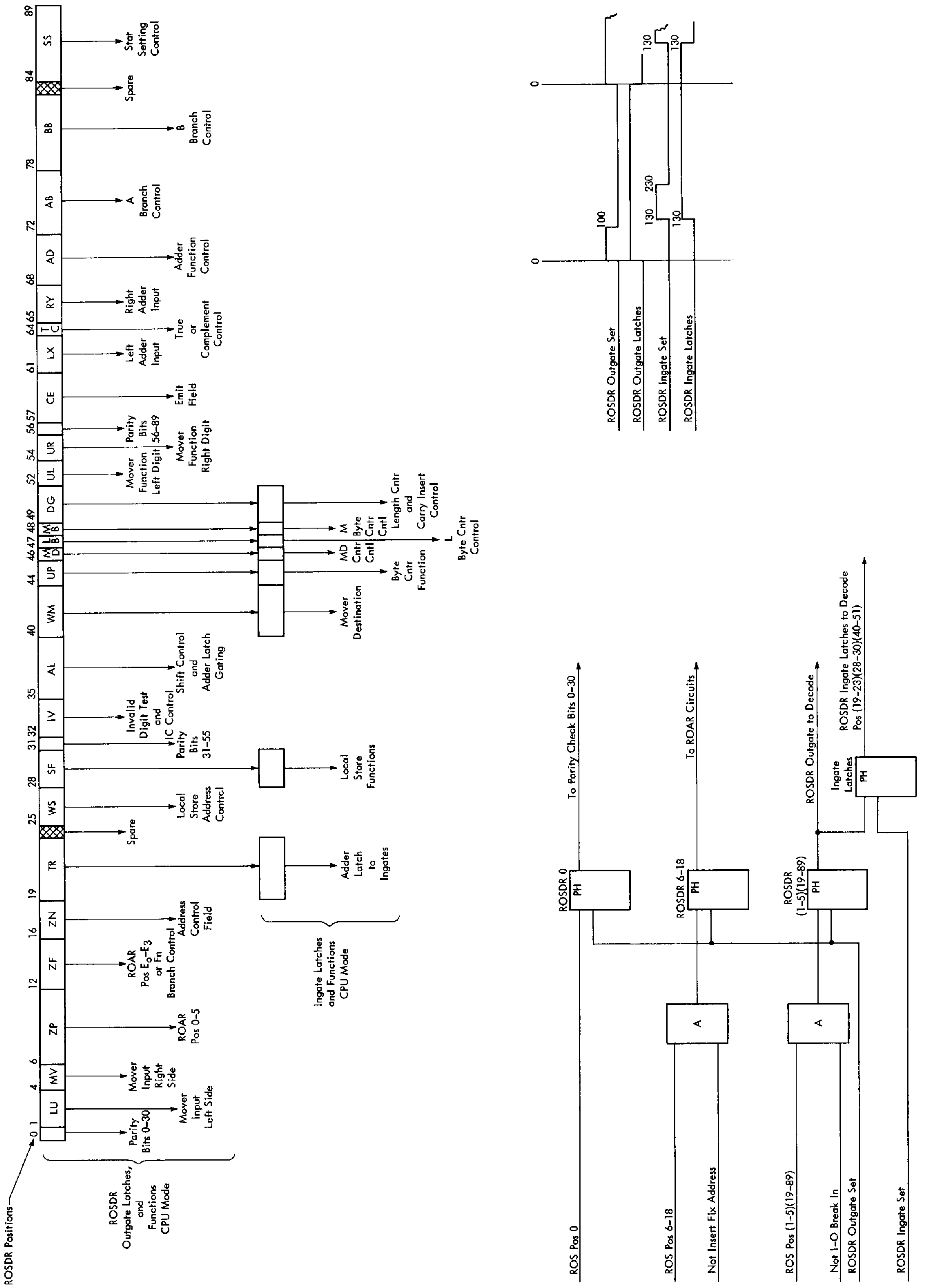


FIGURE 18. ROSDR

machine cycle. The ingate registers remain set until 130 ns of the next cycle.

The ingate registers allow a CROS word to control some gates during the register set time of the next cycle. For example, one CROS word may contain the micro-instruction: add the L and R registers and store the answer in the L register. This CROS word adds the registers on one machine cycle and stores the sum from the adder-out-bus at register set time of the next cycle.

CROS WORD DECODE CIRCUITS

- Decode circuits decode ROSDR fields into control lines.
- Some ROSDR fields are decoded differently for CPU mode than for I/O mode.
- Some ROSDR fields are different for CPU mode than for I/O mode.

The CROS word decode circuits combine specific bits from the ROSDR fields with CPU or I/O mode to activate control lines. In some cases the control lines are further combined with the output of latches, such as the M-Byte Counter, to activate more specific control lines.

The division of the ROSDR bit positions into fields is different for CPU mode than for I/O mode. Figure 19 shows the field division for CPU and I/O modes. The function of all the CPU mode fields are shown, but only the functions that differ from CPU mode functions are shown for I/O mode.

The table in Figure 20 shows the micro-orders that control the left mover input, the bit configuration in the ROSDR LU field that represents the micro-orders, and the resulting control lines and functions.

The left-mover-input decode circuit is a typical example of the CROS word decoding circuits. The left-mover-input decode circuit (Figure 21) uses ROSDR positions 1-3, CPU mode, I/O mode, BAM 0-1 (M-Byte Counter), and BAL 0-1 (L-Byte Counter) to control 13 left-mover-input control lines.

READ ONLY STORAGE ADDRESS REGISTER (ROAR)

- ROAR supplies the 12-bit address that selects the next CROS word.
- The 12 ROAR bit positions are 0-9, A, and B.
- ROAR positions 0-5 are called base address positions.
- ROAR positions 6-9 are called function bits.
- ROAR position A is the A-branch bit.
- ROAR position B is the B-branch bit.
- The I/O-backup register saves the CPU-ROAR address during an I/O break-in.
- ROAR addresses come from ROSDR, data flow paths, fixed address generation, or the I/O-backup register.

The Read Only Storage Address Register (ROAR) supplies the address that selects the next CROS word. The present CROS word controls the address sent through the address register. ROAR is composed of a 12-position address register and three 12-position buffer registers.

The address register is not a storage register; instead, the address register is an OR circuit that gates the output of other system-registers to the address-decode circuits (Figure 22).

The three ROAR buffer registers are: an I/O-backup register that saves the last CPU mode address sent to CROS; a current-ROS-address register that saves the address of the current ROSDR word; and a previous-ROS-address register that saves the address of the previous ROSDR word.

On an I/O break-in cycle, the address transferred to the I/O backup register is the address of the CROS word that is prevented from storing in ROSDR. On the last I/O mode cycle, the microprogram word contains the micro-order to transfer the I/O-backup register to ROAR. The address that the backup register sends to ROAR is the address of the CROS word that was prevented from storing in ROSDR on the break-in cycle.

The ROAR bit positions are divided, by the source of their input data, into four groups. These groups are the base-address bits (positions 0-5), the function-branch bits (positions 6-9), the A-branch bit, and the B-branch bit. Figure 23 shows the control circuits for all ROAR positions.

ROAR Positions 0-5

ROAR positions 0-5 are called base-address positions. Figure 24 shows a composite logic diagram that represents any base-address position. The inputs to all the base-address positions are similar, except for the fixed-address inserts. The name of the insert lines show which bit positions they affect.

There are eight inputs to the address register OR circuit: data keys, ROSDR, SDR, ROAR buffer, and four fixed address inserts.

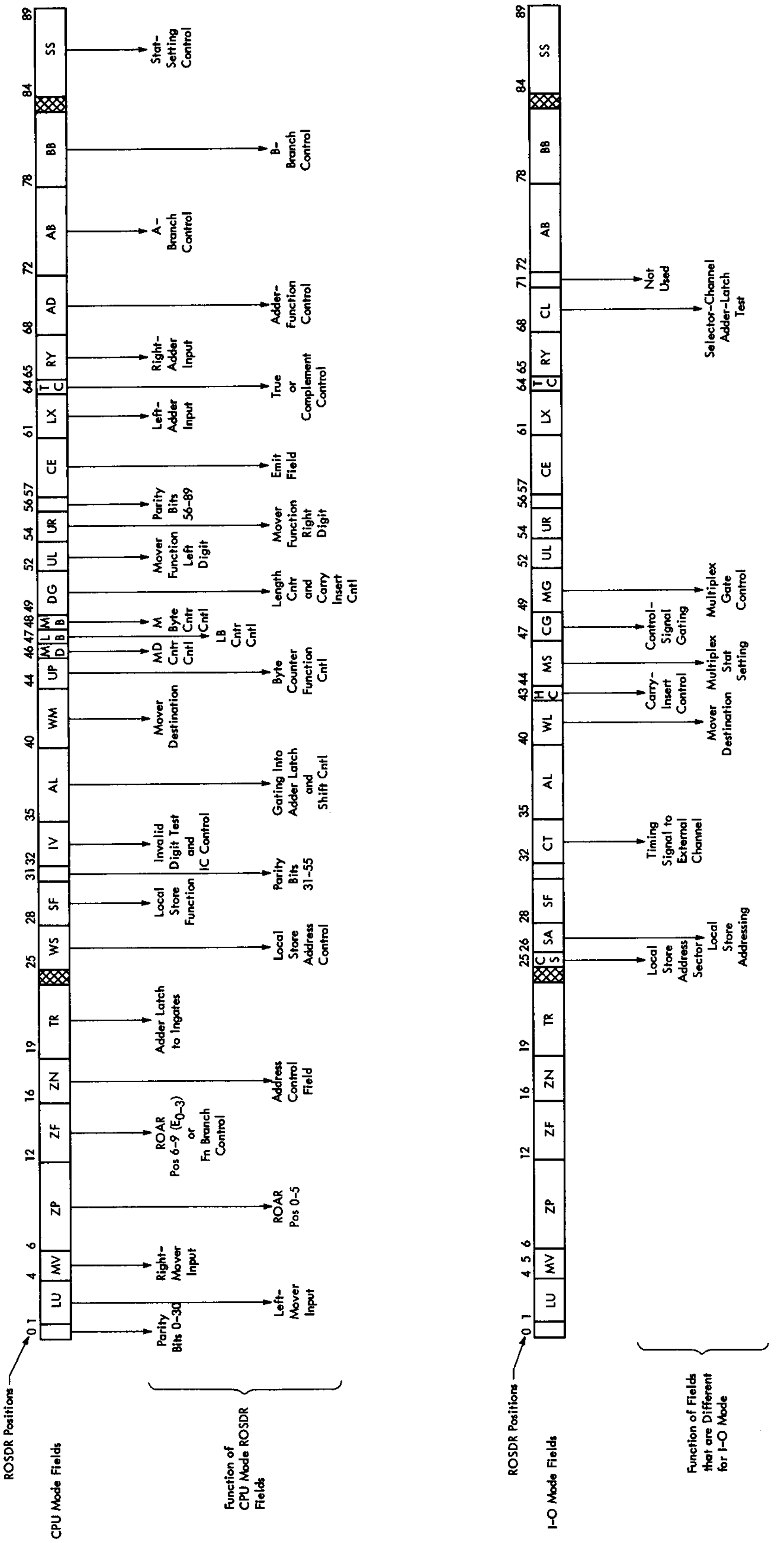


FIGURE 19. MICROPROGRAM WORD FIELDS FOR CPU AND I-O MODES

Micro Order	Bits	Function	Line Name
		CPU Mode or I-O Mode	
O's→U	000	Zeros to U	Zero to U
R3→U	010	R byte 3 to U	R24-31 to U
		CPU Mode Only	
MD,F→U	001	MD and F registers to U	MD and F to U
DD→U	011	Direct Data to U	Direct Data Ctrl to Mover
XTR→U	100	XTR to U, reset XTR	XTR to U Int Reg Reset
PSW→U	101	PSW (32-39) to U	PSW to U
L/MB→U	110	L to U per MB Ctr (Which line is active is determined by the MB Ctr)	L Byte 0 to U L Byte 1 to U L Byte 2 to U L Byte 3 to U
L/LB→U	111	L to U per LB Ctr (Which line is active is determined by the LB Ctr)	L Byte 0 to U L Byte 1 to U L Byte 2 to U L Byte 3 to U
		I-O Mode Only	
MPX→U	011	MPX Buffer-In-Bus to U	MPLX Buffer-In-Bus to U
L0→U	100	L byte 0 to U	L Byte 0 to U
L1→U	101	L byte 1 to U	L Byte 1 to U
L2→U	110	L byte 2 to U	L Byte 2 to U
L3→U	111	L byte 3 to U	L Byte 3 to U

FIGURE 20. MICRO-ORDER TABLE FOR LEFT MOVER INPUT - ROSDR 1 - 3

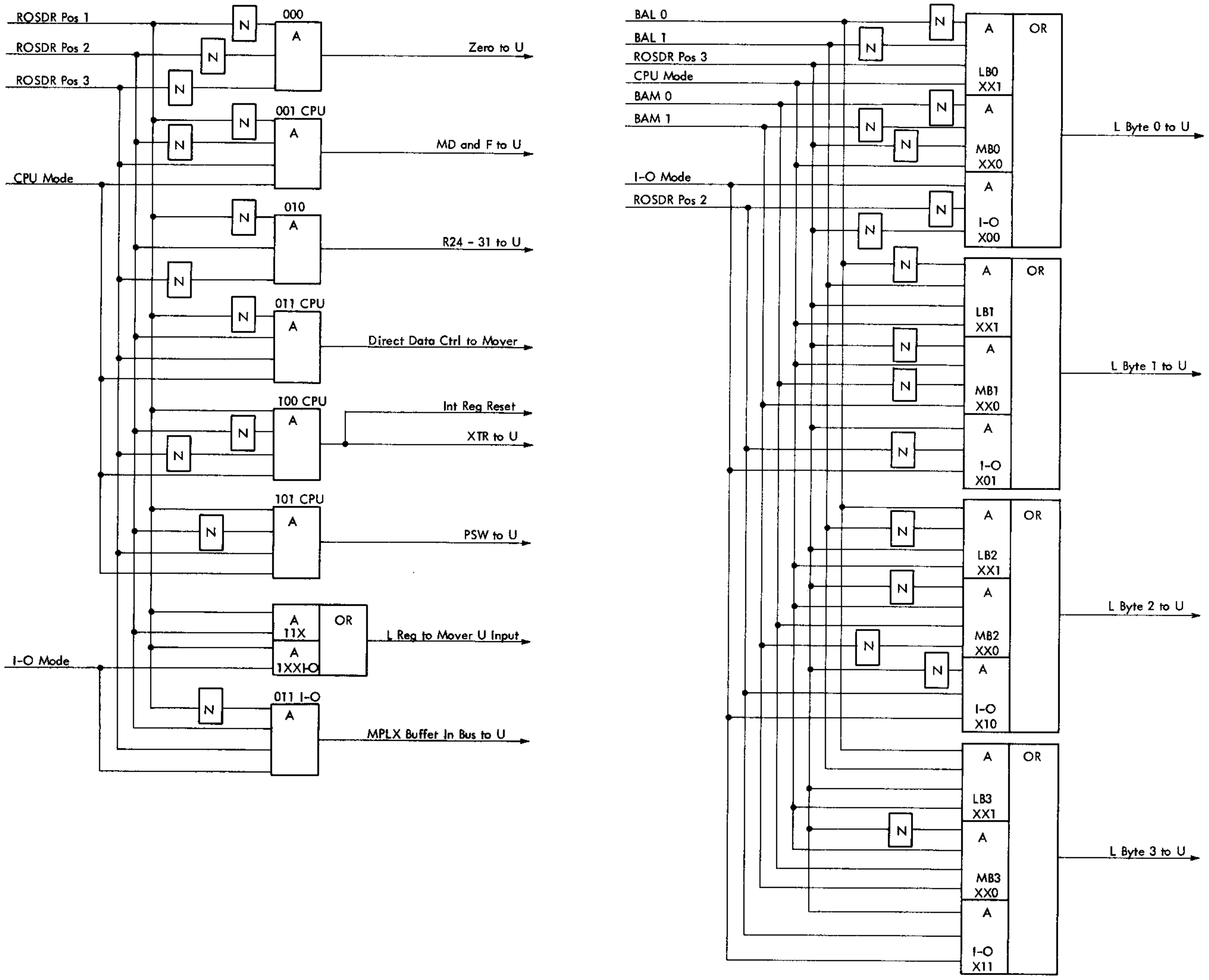


FIGURE 21. DECODE CIRCUITS FOR LEFT MOVER INPUT (ROSDR 1-3)

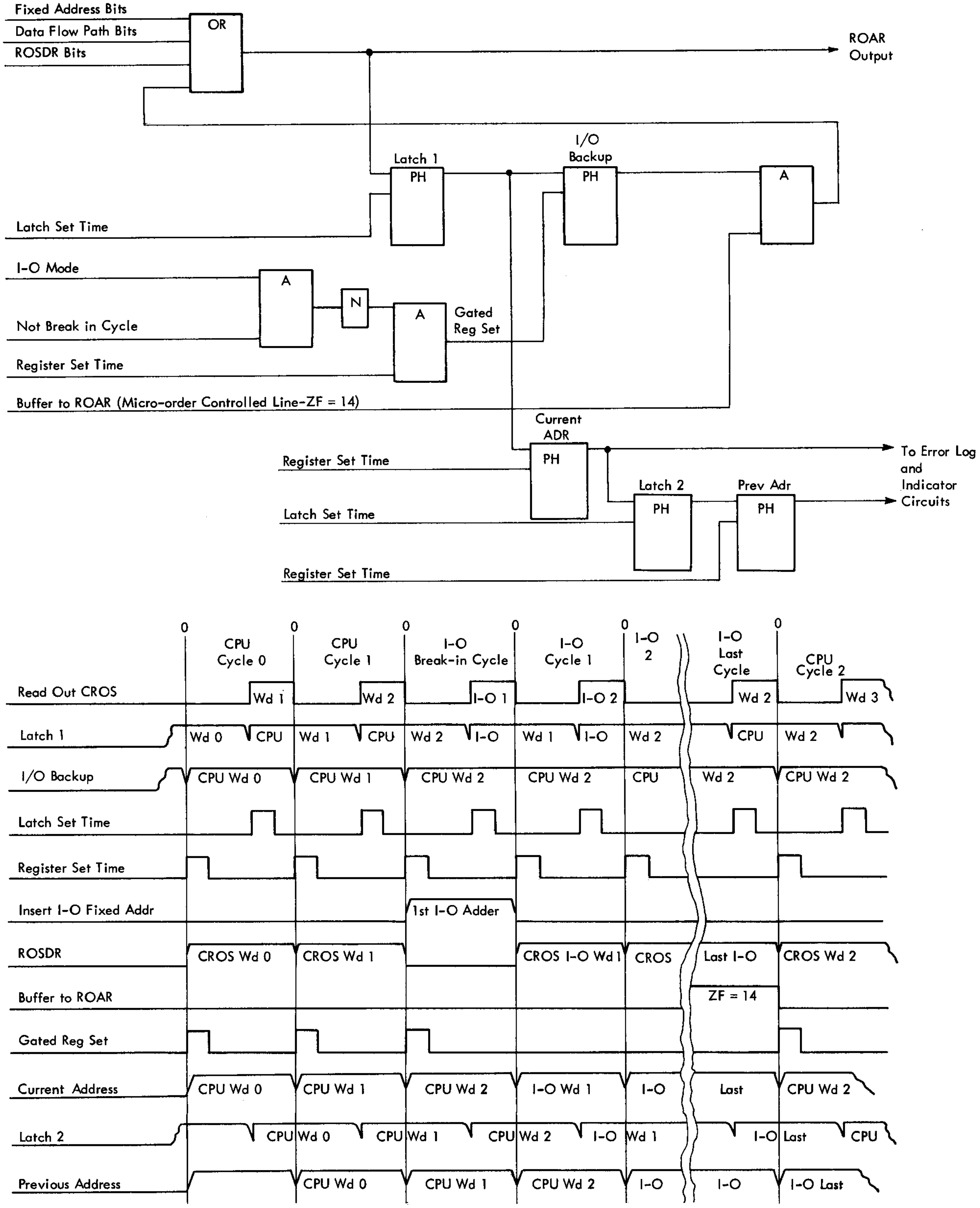


FIGURE 22. BASIC ROAR DATA FLOW

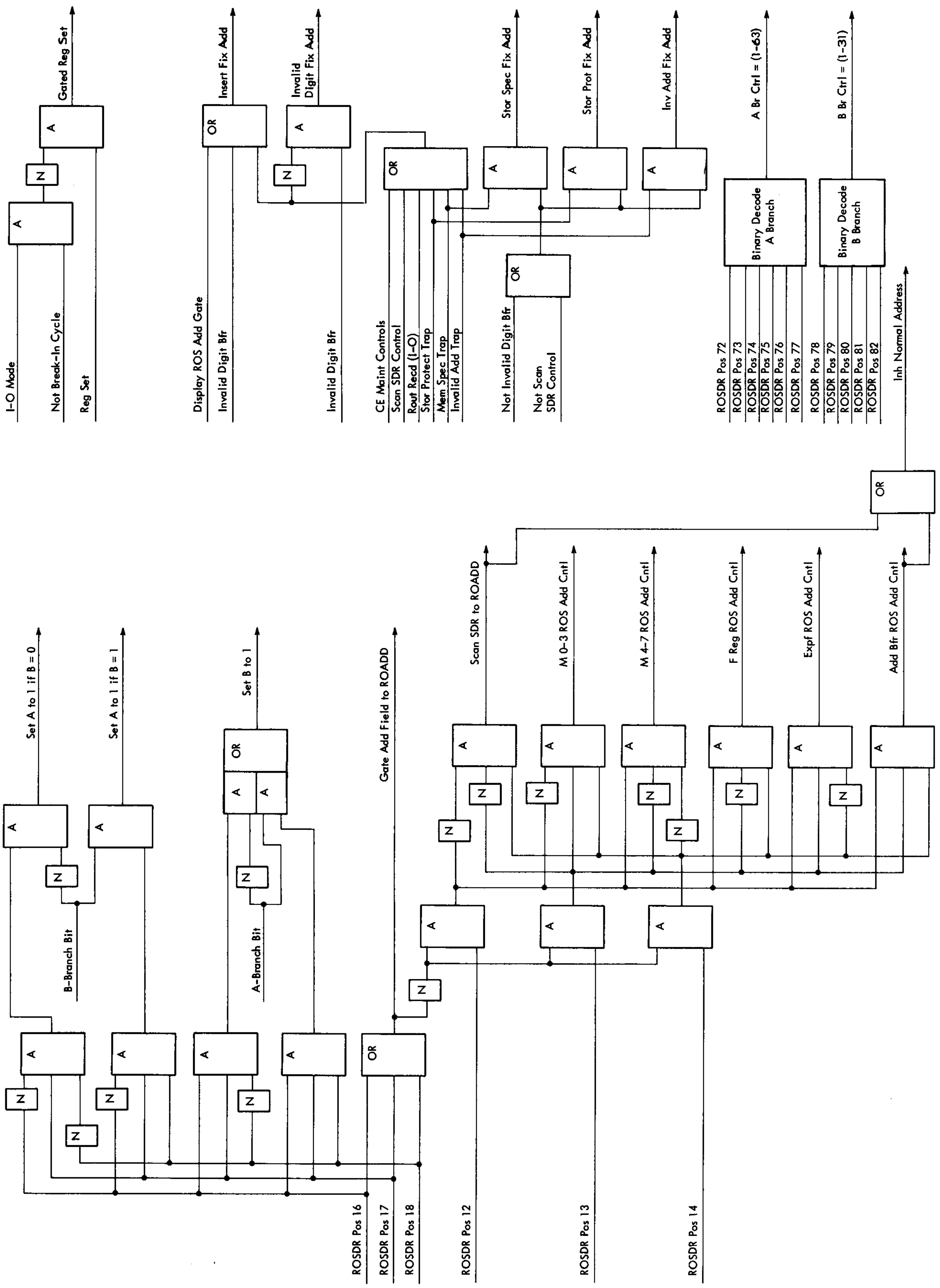
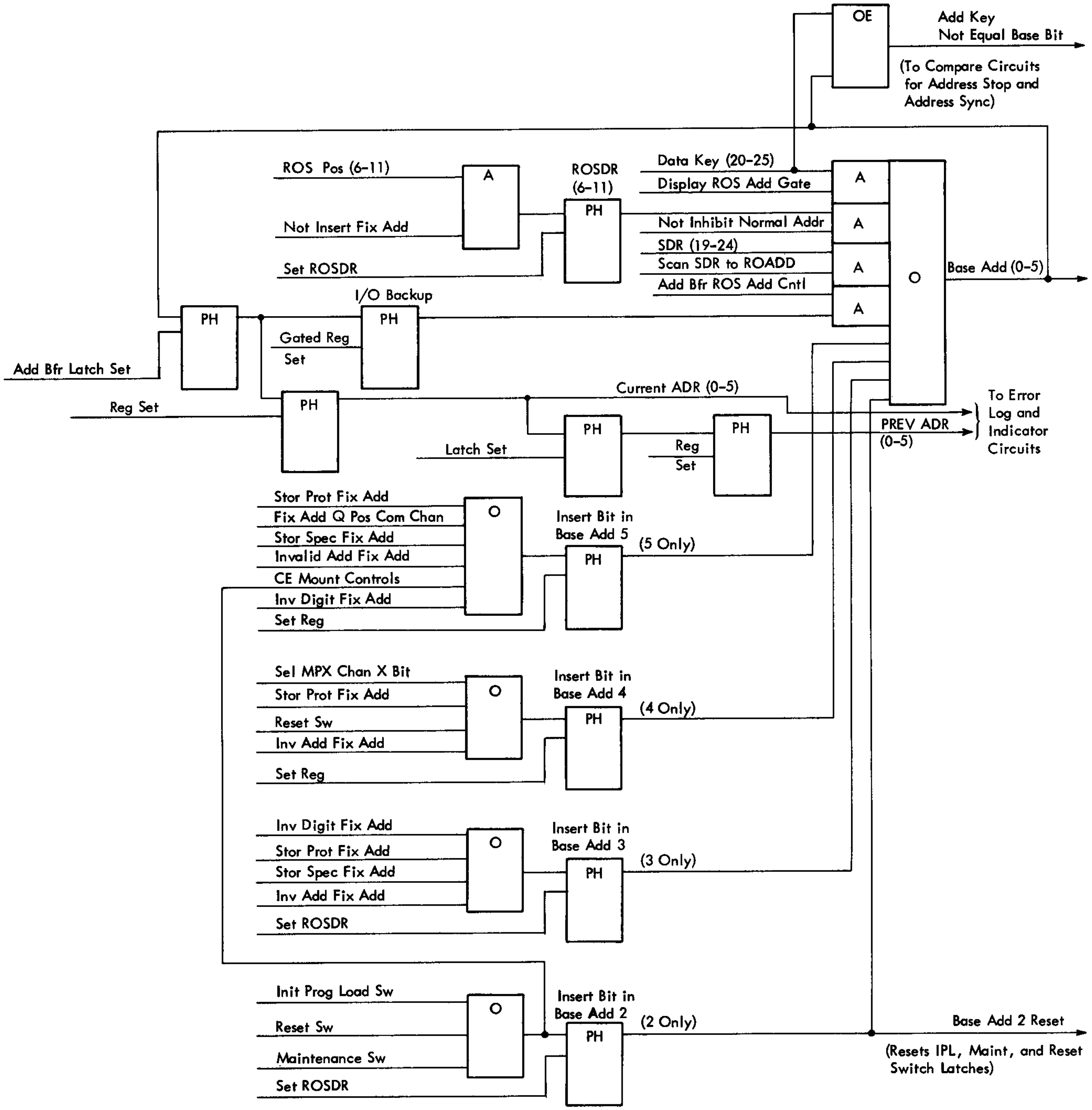


FIGURE 23. READ ONLY STORAGE ADDRESS REGISTER CONTROLS



Note: This diagram represents any one position of Read Only Storage Address Register positions 0-5. Some special insert lines are not in all positions. The special insert lines are marked for the positions that they control.
ROAR positions 0, 1, and 5 are on ALD page KK301.
ROAR positions 2 and 3 are on ALD page KK311.
ROAR position 4 is on ALD page KK321.

FIGURE 24. READ ONLY STORAGE ADDRESS REGISTER - POSITIONS 0-5

Data-Key Input: The data keys are console switches which allow the CE to insert addresses into ROAR.

ROSDR Input: ROSDR is the normal input to the base-address positions. The inhibit-normal-address line prevents sending ROSDR to the address register when micro-orders are gating SDR or ROAR buffer to the address register. The insert-fixed-address line prevents setting ROSDR when either the data-key input or fixed address input is used.

SDR Input: The SDR input gates main storage data-register positions 19-24 into the base-address positions 0-5. This address path is used for main-storage-controlled FLT's (Fault Locating Tests).

I/O Backup Register Input: The I/O-backup input is a micro-order controlled input that supplies the next CPU address following an I/O routine.

Fixed-Address Inputs: The fixed-address circuits supply a fixed address for error traps, I/O break-in, and console switches.

ROAR Positions 6-9

ROAR positions 6-9 are called function-branch bits. Figure 25 shows a logic diagram that represents any function-branch bit position. The function-branch bit positions can accept bits from the system's data flow path; this allows program data to select one of 16 ROAR addresses.

For example, the instruction-decode routine (I Fetch) uses a function branch on the first four op-code bits to determine the instruction format. The decode routine then uses the last four op-code bits to determine the exact instruction. The ROAR address at the end of the instruction-decode routine is the first word of the instruction routine.

There are nine inputs to the function-branch positions of the address register; data keys, exponent-function bits, M-register 0-3, M-register 4-7, F-register, ROSDR, SDR, channel-function bits, and ROAR buffer.

Data-Key Input: The data keys are console switches which allow the CE to insert addresses into ROAR.

Exponent-Function Input: The exponent-function bits are adder-output bits that represent the difference of two exponents. The bits are used to cause a function branch to the proper floating-point shift routine.

M-Register 0-3 Input: Data in M-register 0-3 can be entered as function bits. I Fetch uses this path to enter the first four bits of the op-code.

M-Register 4-7 Input: Data in M-register 4-7 can be entered as function bits. I Fetch uses this path to enter the last four bits of the op-code.

F-Register Input: Data in the F register can be entered as function bits. Hexadecimal multiply uses this function branch. The multiplier is placed in the F register and a function branch is taken to the microprogram routine that multiplies by that value.

ROSDR Input: ROSDR positions 12-15 can be entered into the function-bit positions. ROSDR is used when no function branch or fixed address is desired.

SDR Input: SDR positions 25-28 can be entered as function bits. The SDR entry is used in main-storage-controlled FLT's.

Channel-Function Bit Input: The channel circuits can insert bits in the function-bit positions. These bits are inserted to select the proper I/O mode routine for the channel operation.

I/O Backup Register Input: The I/O-backup inputs are micro-order controlled inputs that supply the next CPU address following an I/O routine.

ROAR Position A

The eleventh position of ROAR is called the A-branch position. Figure 26 shows a logic diagram for the A-branch position. The A-branch bit by itself allows a two-way microprogram branch, which is controlled by various system registers and latches. If the A-branch bit is used in conjunction with a function branch, the microprogram can branch to any one of 32 addresses.

There are six groups of inputs to the A-branch address-register position: tests controlled by ROSDR 72-77, tests controlled by ROSDR 16-18, SDR, data keys, ROAR buffer, and fixed-address inserts.

Tests Controlled by ROSDR 72-77: These tests are the inputs controlled by the A Br Ctrl 01-63 lines. The A Br Ctrl lines are the binary decoded output of ROSDR 72-77. In the ALD's this line is really two lines, an X line and a Y line. The X and Y lines are formed by octal decoding the six bits of the field.

Tests Controlled by ROSDR 16-18: These tests force the A-branch bit to a "1" state by testing the B-branch bit. This test can be used in conjunction with the tests controlled by ROSDR 72-77. If the conditions are satisfied for either test, the result is an A-branch bit.

Note: This A-branch test should not be used in conjunction with the two immediate B-branch tests, the adder carry test and the adder sum position 0 equals zero test. These two B-branch tests occur too late in the machine cycle for reliable A-branch bit operation if this test is used.

SDR Input: SDR position 29 can be entered into the A-branch bit position. The SDR entry is used in main-storage-controlled FLT's.

Data-Key Input: The data keys are console switches which allow the CE to insert addresses into ROAR.

I/O Backup Register Input: The I/O-backup inputs are micro-order controlled inputs that supply the next CPU address following an I/O routine. No A-branch test micro-orders should be used in conjunction with the micro-order to gate the ROAR buffer to ROAR.

Fixed-Address Input: The fixed-address circuits supply a fixed address for error traps, I/O break-in, and console switches.

ROAR Position B

The twelfth position of ROAR is the B-branch position. Figure 27 shows the logic of the B-branch position. The B-branch bit by itself allows the system data to control a two-way microprogram branch. If the B-branch bit is used in conjunction with the A-branch bit and the function-branch bits, the microprogram can branch to any one of 64 addresses.

The B-branch position is not used to select an array driver. Instead, it is used to gate one of the two CROS words read out by the ROAR address into the ROS latches. Because the B-branch bit is not used to read out CROS, it can be set later in the machine cycle than the rest of the ROAR address.

The A-branch tests and most of the B-branch tests are made on conditions at the start of the machine cycle. However, the B-branch has two tests that are made late in the machine cycle, branch-on adder carry and branch-on adder pos 0 sum equal zero. These two tests are made just before the CROS output is gated into the ROS latches. To speed up the upper-lower gating circuits, the B-branch position has two output circuits, one for the no-bit condition and one for the bit condition.

There are six groups of inputs to the B-branch address-register position; tests controlled by ROSDR 78-82, tests controlled by ROSDR 16-18, tests controlled by ROSDR 72-77, data keys, SDR, and ROAR buffer.

Tests Controlled by ROSDR 78-82: These tests are the inputs controlled by B Br Ctrl 01-31 lines. The B Br Ctrl lines are the binary decoded outputs of ROSDR 78-82. In the ALD's these lines are really AND'ed X and Y lines. The X and Y lines are formed by octal decoding the five bits of the field.

Note: If ROSDR 72-77 contains a test that affects the B branch bit, ROSDR 78-82 must be all zeros (no test).

Tests Controlled by ROSDR 16-18: These tests force the B branch bit to a "1" state, by testing the A-branch bit. These tests can be used in conjunction with other ROSDR controlled tests; if the conditions are satisfied for any test, the result is a B-branch bit.

Tests Controlled by ROSDR 72-77: There are four A-branch control tests that affect the B-branch bit. These four A-branch control tests allow one micro-order to control a four way branch. ROSDR 78-82 must be all zeros if these tests are used.

A Br Ctrl 35 sets the A branch bit with edit stat 1; it sets the B branch bit with edit stat 2.

A Br Ctrl 49 sets the A branch bit with mover position 6; it sets the B branch bit with mover position 7.

A Br Ctrl 56 sets the A branch bit with either an exception branch stat, or with no-bit in instruction counter position 30. A Br Ctrl 56 sets the B branch bit with either refetch stat and an instruction counter position 30 bit, or with an exception branch stat.

A Br Ctrl 58 sets the A branch bit with either a timer or an external channel signal; it sets the B branch bit with a channel interrupt signal.

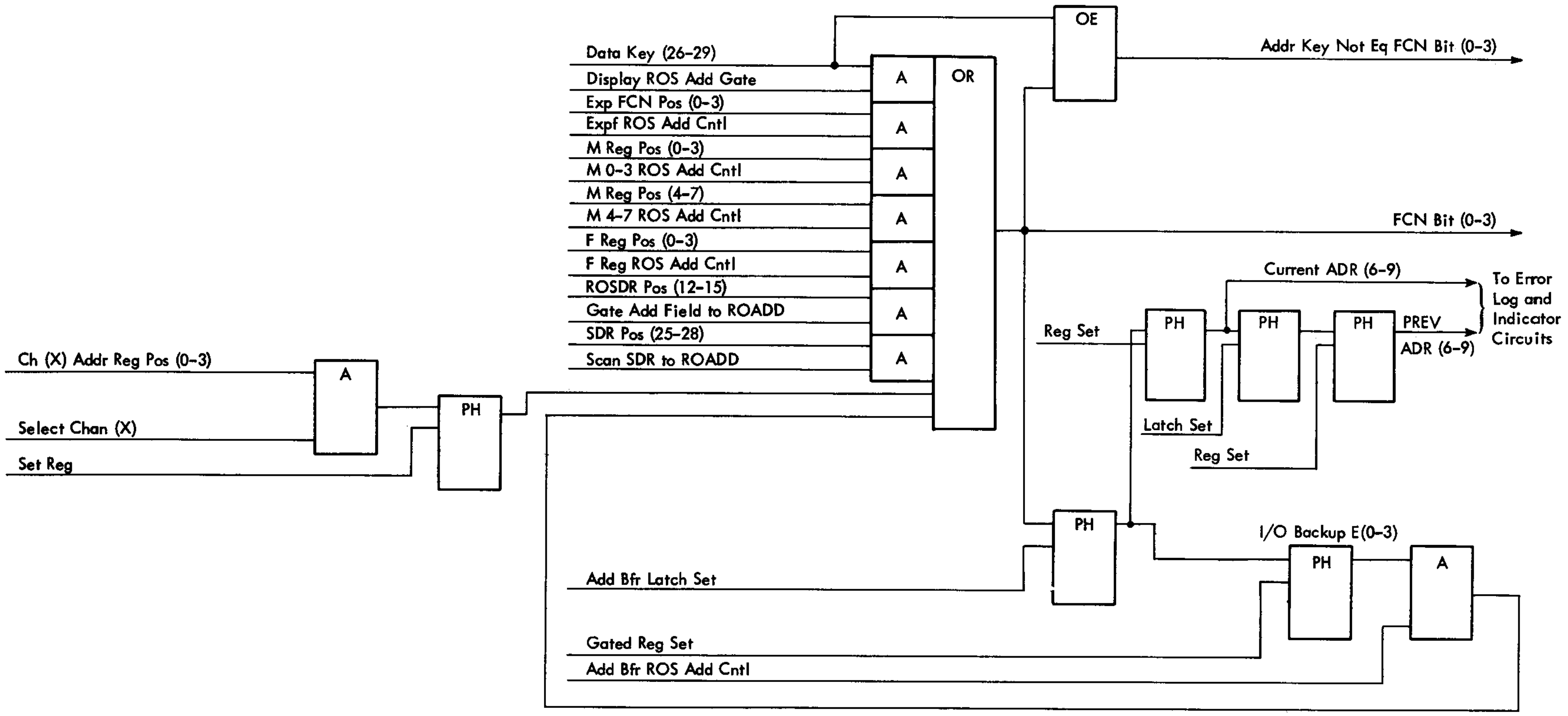
Data Key Input: The data keys are console switches which allow the CE to insert addresses into ROAR.

SDR Input: SDR position 30 can be entered into the B branch bit position. The SDR entry is used in main-storage-controlled FLT's.

I/O Backup Register Input: The I/O-backup inputs are micro-order controlled inputs that supply the next CPU address following an I/O routine. No B-branch test micro-order should be used in conjunction with the micro-order to gate the ROAR buffer to ROAR.

DRIVER SELECTION

- ROAR positions 5-9 are decoded into 1 of 22 select lines.
- ROAR positions 0-4 and A are decoded into 1 of 64 strobed drive lines.



Note: This diagram represents any one position of Read Only Storage Address Register positions 6-9 (FCN Bits 0-3)
FCN Bit 0 is on ALD page KK011
FCN Bit 1 is on ALD page KK021
FCN Bit 2 is on ALD page KK031
FCN Bit 3 is on ALD page KK041

FIGURE 25. READ ONLY STORAGE ADDRESS REGISTER - POSITIONS 6-9

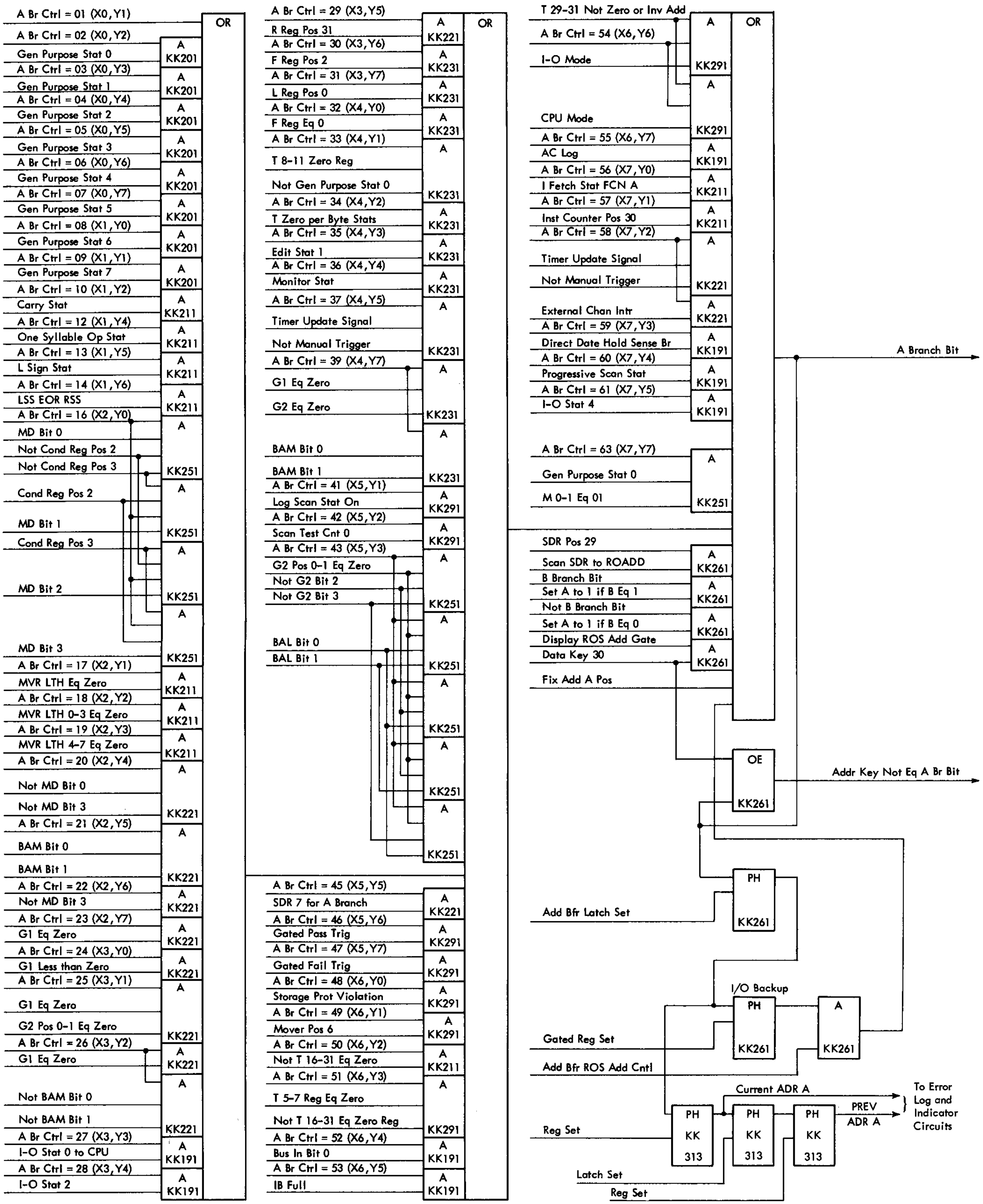


FIGURE 26. READ ONLY STORAGE ADDRESS REGISTER - POSITION A

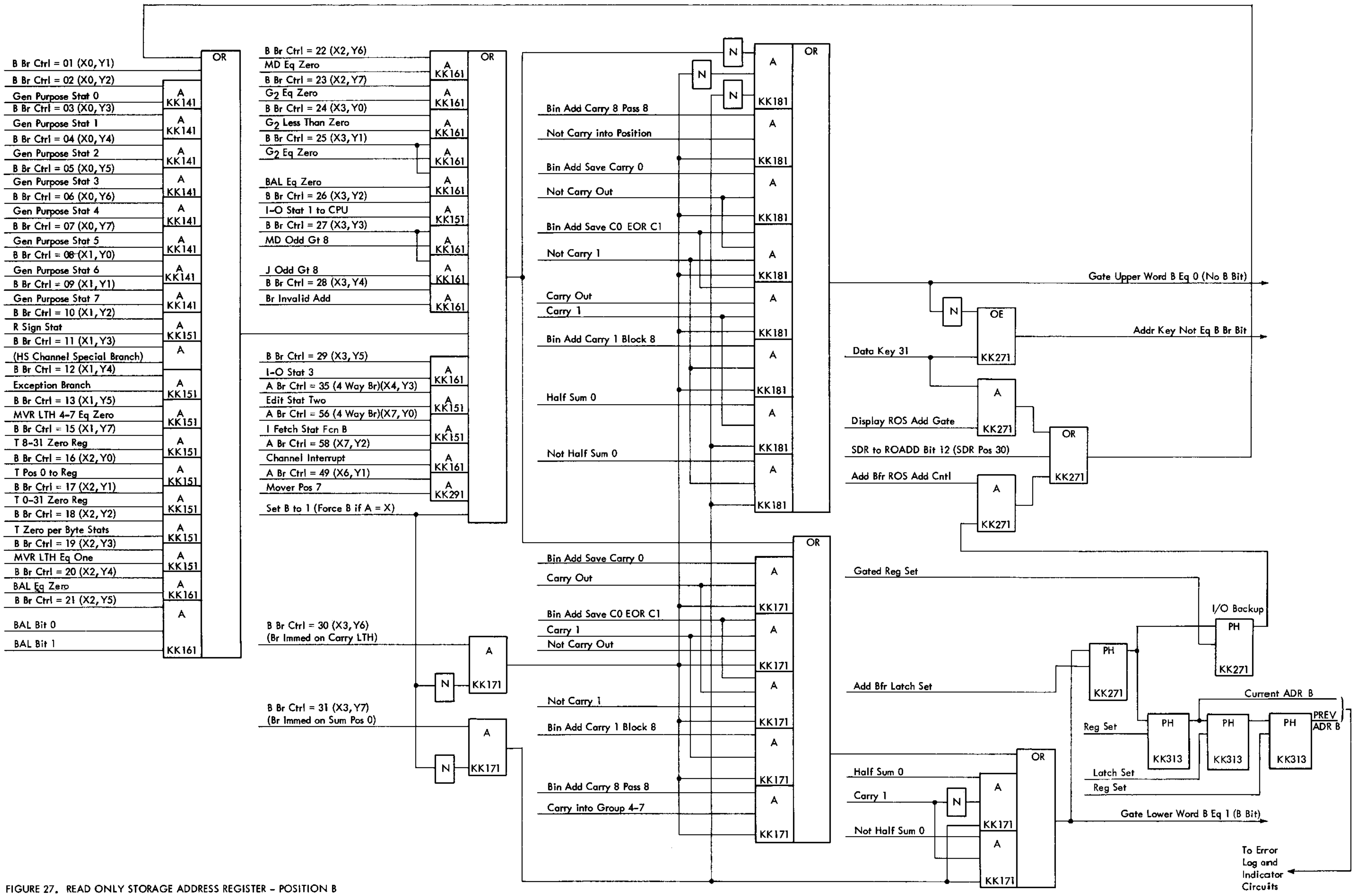


FIGURE 27. READ ONLY STORAGE ADDRESS REGISTER – POSITION B

- One select line and one strobed drive line selects 1 of 1,408 array drive lines.

The driver selection circuits decode the address in ROAR positions 1-A to select one of 1,408 CROS array drive lines. The ROAR address is decoded into a select line and a strobed drive line. The select and strobed drive lines are then AND'ed to select one array drive line.

Select Lines

Select lines are activated by decoding the base-address-5 position and the four function-branch positions of ROAR (Figure 28). Although these five ROAR bit positions can be decoded 32 different ways, only the first 22 combinations are considered valid addresses. The other ten combinations are not tested. If (by error) the bit combination for a 31 is entered into these ROAR positions, the decode circuit will treat it as a bit combination for a 15. Select line 15 will be activated.

Strobed Drive Lines

A ROAR address selects one of 64 strobed drive lines by decoding the base address 0-4 and the A branch positions (Figure 29). The decoded address is AND'ed with driver strobe to send a timed signal to the array drivers. Each strobed drive line controls the array drivers for one CROS quarter plane.

Base addresses 0-2 are AND'ed to select one of seven first drive lines. The first drive lines are AND'ed with base addresses 3-4 to select one of 32 second drive lines. The second drive lines are AND'ed with the A branch bit and driver strobe to select one of 64 strobed drive lines.

Array Drivers

There are 1,408 array drivers. Each array driver is a simple AND switch that AND's one of the 64 drive lines with one of the 22 select lines (Figure 30). The AND circuits are single transistors; the drive lines gate the emitters while the select lines control the bases. Voltage is supplied to the collectors through the array drive-line-terminating resistors.

Figure 31 shows the layout of the drive lines as they enter a CROS plane.

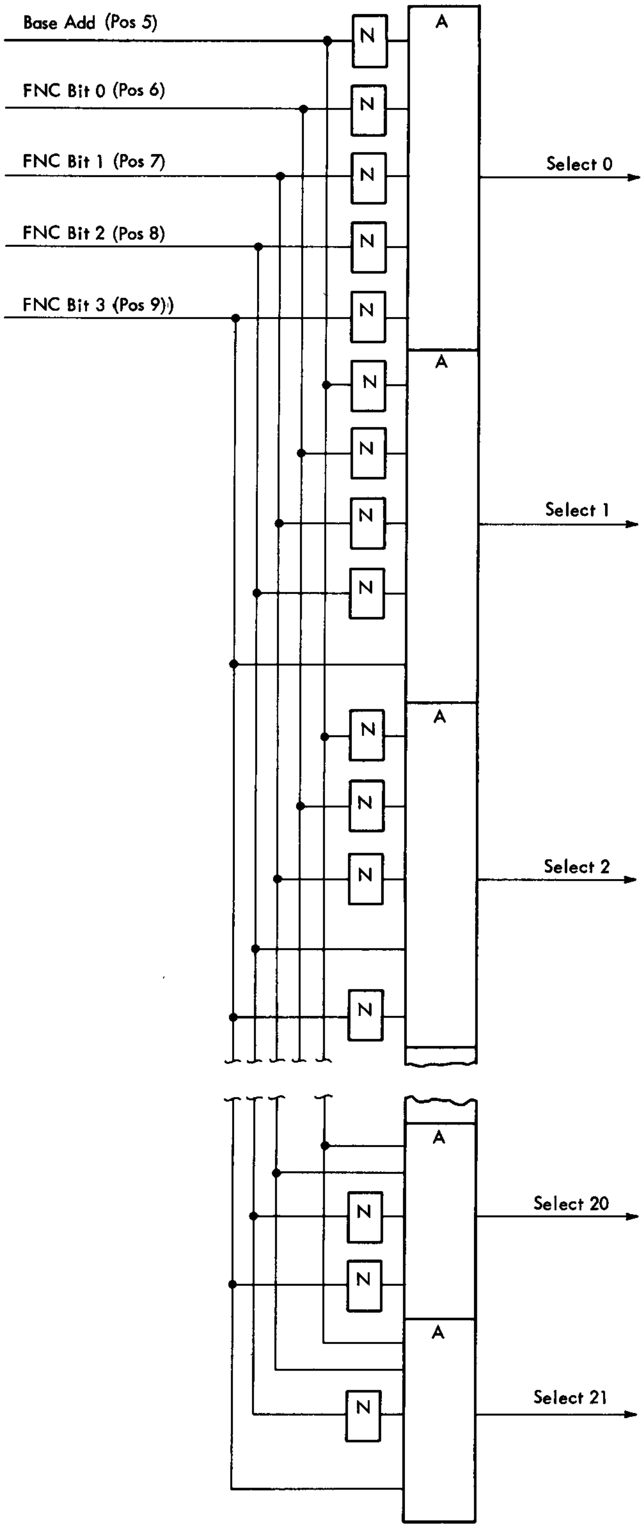
CROS WORD PARITY CIRCUITS

- The CROS word is parity checked in three groups, ROSDR 0-30, ROSDR 31-55, and ROSDR 56-89.
- The first bit of each group is a parity check bit.
- Each group must have odd parity to be valid.
- ROSDR 0-30 is not checked on an "Insert Fixed Address" cycle.
- ROSDR 0-89 is not checked on a "I/O Break In" cycle.

The CROS word in ROSDR is parity checked in three groups (Figure 32). Each group contains its own parity check bit and must total an odd number of bits to result in correct parity.

ROSDR 0-30 has a special circuit that forces correct parity on any cycle that inserts a fixed address into ROAR. This circuit is necessary because ROSDR 6-18 is prevented from storing CROS bits when the insert fixed address line is active.

ROSDR 0-89 is not parity checked on a "I/O Break In" cycle. Only CROS word position 0 is allowed to store in ROSDR on a break-in cycle; therefore, the ROSDR word would always result in a parity error if it is checked on a break-in cycle.



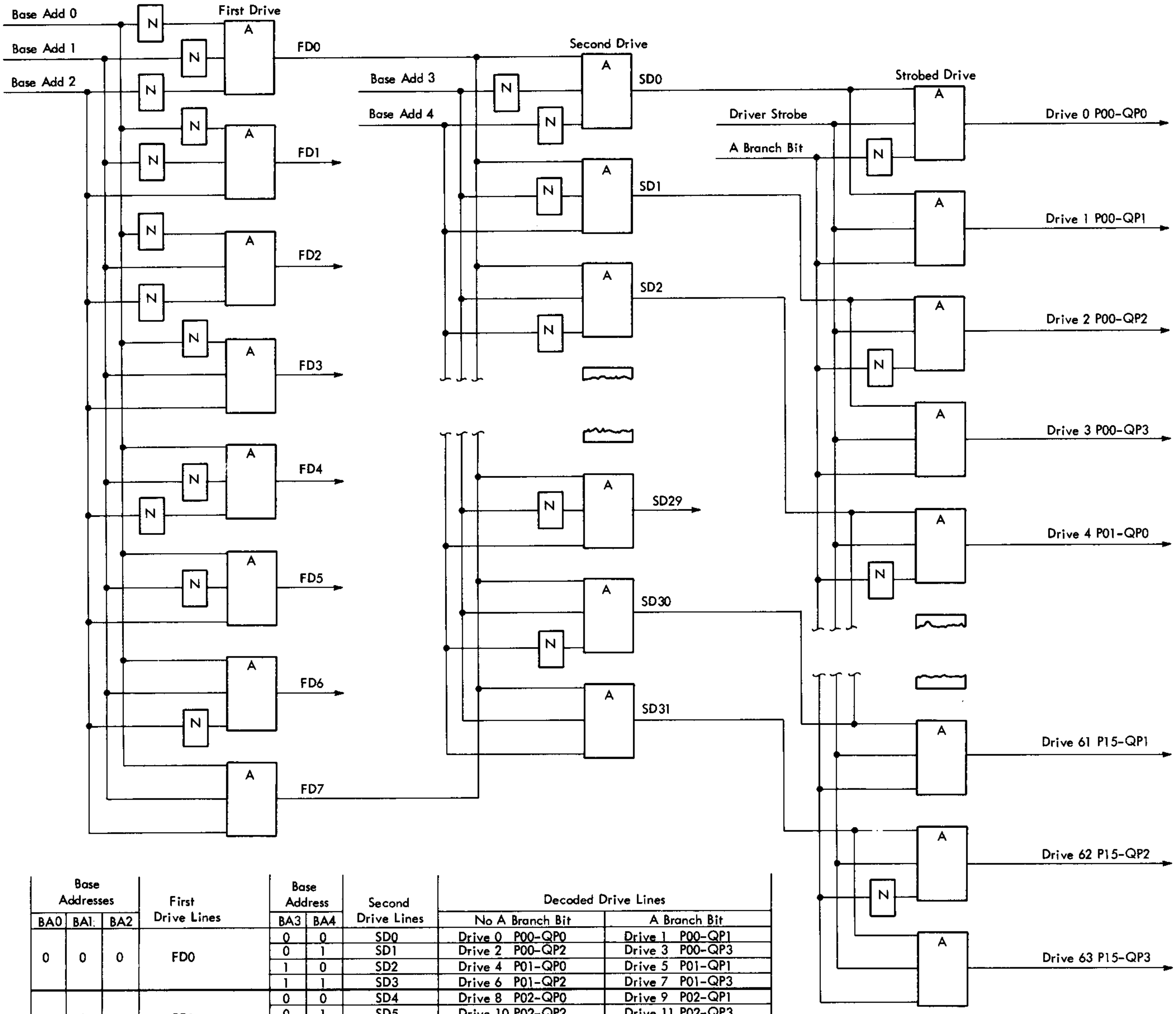
Select Lines	ROAR Positions				
	5	6	7	8	9
0	0	0	0	0	0
1	0	0	0	0	1
2	0	0	0	1	0
3	0	0	0	1	1
4	0	0	1	0	0
5	0	0	1	0	1
6	-	0	1	1	0
7	-	0	1	1	1
8	-	1	0	0	0
9	-	1	0	0	1
10	-	1	0	1	0
11	-	1	0	1	1
12	-	1	1	0	0
13	-	1	1	0	1
14	-	1	1	1	0
15	-	1	1	1	1
16	1	-	0	0	0
17	1	-	0	0	1
18	1	-	0	1	0
19	1	-	0	1	1
20	1	-	1	0	0
21	1	-	1	0	1

Select Lines 0-5 are decoded from ROAR Pos 5,6,7,8, and 9.

Select Lines 6-15 are decoded from ROAR Pos 6,7,8, and 9.

Select Lines 16-21 are decoded from ROAR Pos 5,7,8, and 9.

FIGURE 28. BINARY DECODE OF ROAR TO SELECT LINES



Base Addresses			First Drive Lines	Base Address		Second Drive Lines	Decoded Drive Lines	
BA0	BA1:	BA2		BA3	BA4		No A Branch Bit	A Branch Bit
0	0	0	FD0	0	0	SD0	Drive 0 P00-QP0	Drive 1 P00-QP1
				0	1	SD1	Drive 2 P00-QP2	Drive 3 P00-QP3
				1	0	SD2	Drive 4 P01-QP0	Drive 5 P01-QP1
				1	1	SD3	Drive 6 P01-QP2	Drive 7 P01-QP3
0	0	1	FD1	0	0	SD4	Drive 8 P02-QP0	Drive 9 P02-QP1
				0	1	SD5	Drive 10 P02-QP2	Drive 11 P02-QP3
				1	0	SD6	Drive 12 P03-QP0	Drive 13 P03-QP1
				1	1	SD7	Drive 14 P03-QP2	Drive 15 P03-QP3
0	1	0	FD2	0	0	SD8	Drive 16 P04-QP0	Drive 17 P04-QP1
				0	1	SD9	Drive 18 P04-QP2	Drive 19 P04-QP3
				1	0	SD10	Drive 20 P05-QP0	Drive 21 P05-QP1
				1	1	SD11	Drive 22 P05-QP2	Drive 23 P05-QP3
0	1	1	FD3	0	0	SD12	Drive 24 P06-QP0	Drive 25 P06-QP1
				0	1	SD13	Drive 26 P06-QP2	Drive 27 P06-QP3
				1	0	SD14	Drive 28 P07-QP0	Drive 29 P07-QP1
				1	1	SD15	Drive 30 P07-QP2	Drive 31 P07-QP3
1	0	0	FD4	0	0	SD16	Drive 32 P08-QP0	Drive 33 P08-QP1
				0	1	SD17	Drive 34 P08-QP2	Drive 35 P08-QP3
				1	0	SD18	Drive 36 P09-QP0	Drive 37 P09-QP1
				1	1	SD19	Drive 38 P09-QP2	Drive 39 P09-QP3
1	0	1	FD5	0	0	SD20	Drive 40 P10-QP0	Drive 41 P10-QP1
				0	1	SD21	Drive 42 P10-QP2	Drive 43 P10-QP3
				1	0	SD22	Drive 44 P11-QP0	Drive 45 P11-QP1
				1	1	SD23	Drive 46 P11-QP2	Drive 47 P11-QP3
1	1	0	FD6	0	0	SD24	Drive 48 P12-QP0	Drive 49 P12-QP1
				0	1	SD25	Drive 50 P12-QP2	Drive 51 P12-QP3
				1	0	SD26	Drive 52 P13-QP0	Drive 53 P13-QP1
				1	1	SD27	Drive 54 P13-QP2	Drive 55 P13-QP3
1	1	1	FD7	0	0	SD28	Drive 56 P14-QP0	Drive 57 P14-QP1
				0	1	SD29	Drive 58 P14-QP2	Drive 59 P14-QP3
				1	0	SD30	Drive 60 P15-QP0	Drive 61 P15-QP1
				1	1	SD31	Drive 62 P15-QP2	Drive 63 P15-QP3

FIGURE 29. BINARY DECODE OF ROAR TO 1 OF 64 DRIVE LINES

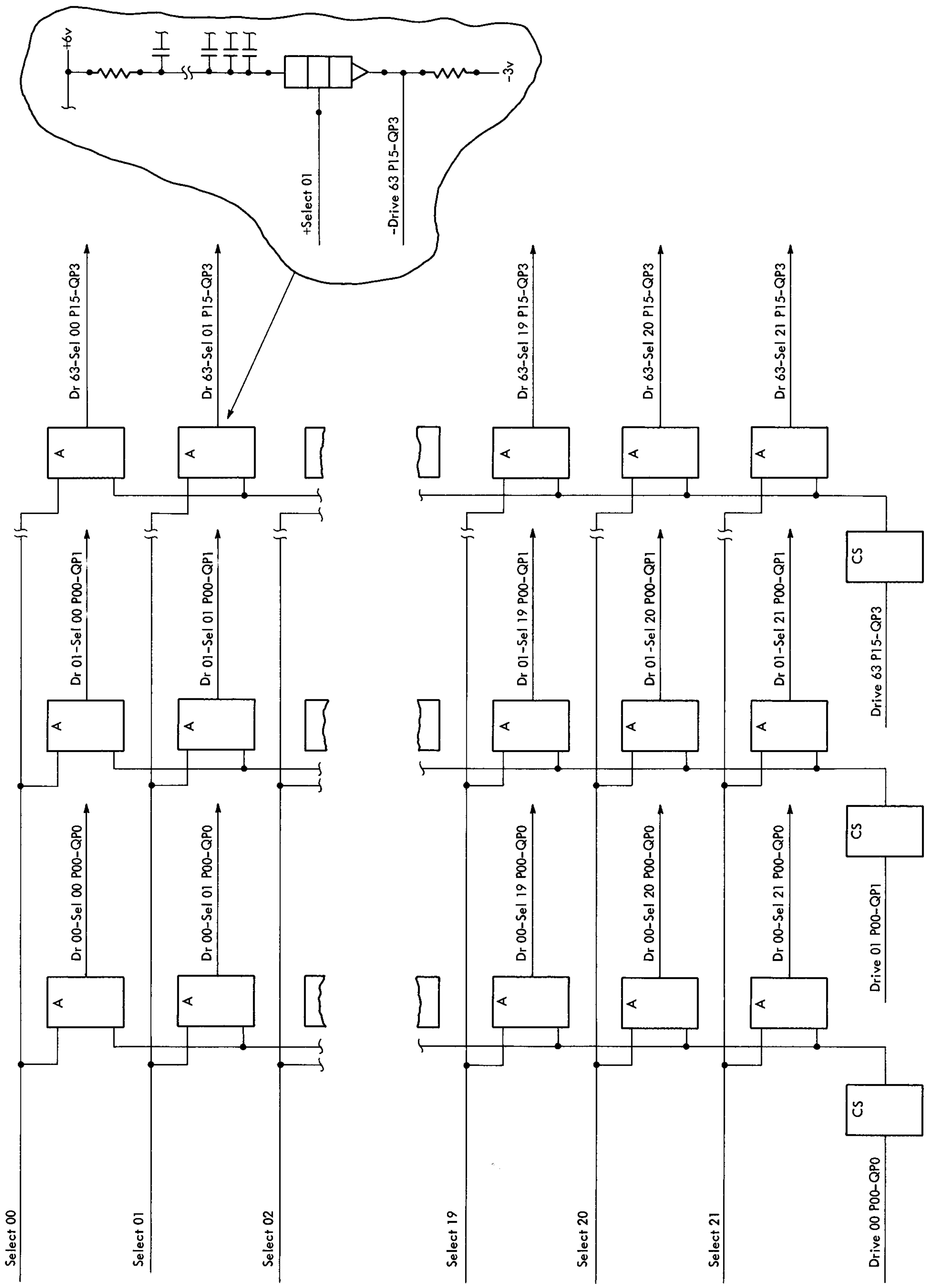


FIGURE 30. SELECTION OF ONE OF 1408 ARRAY DRIVERS

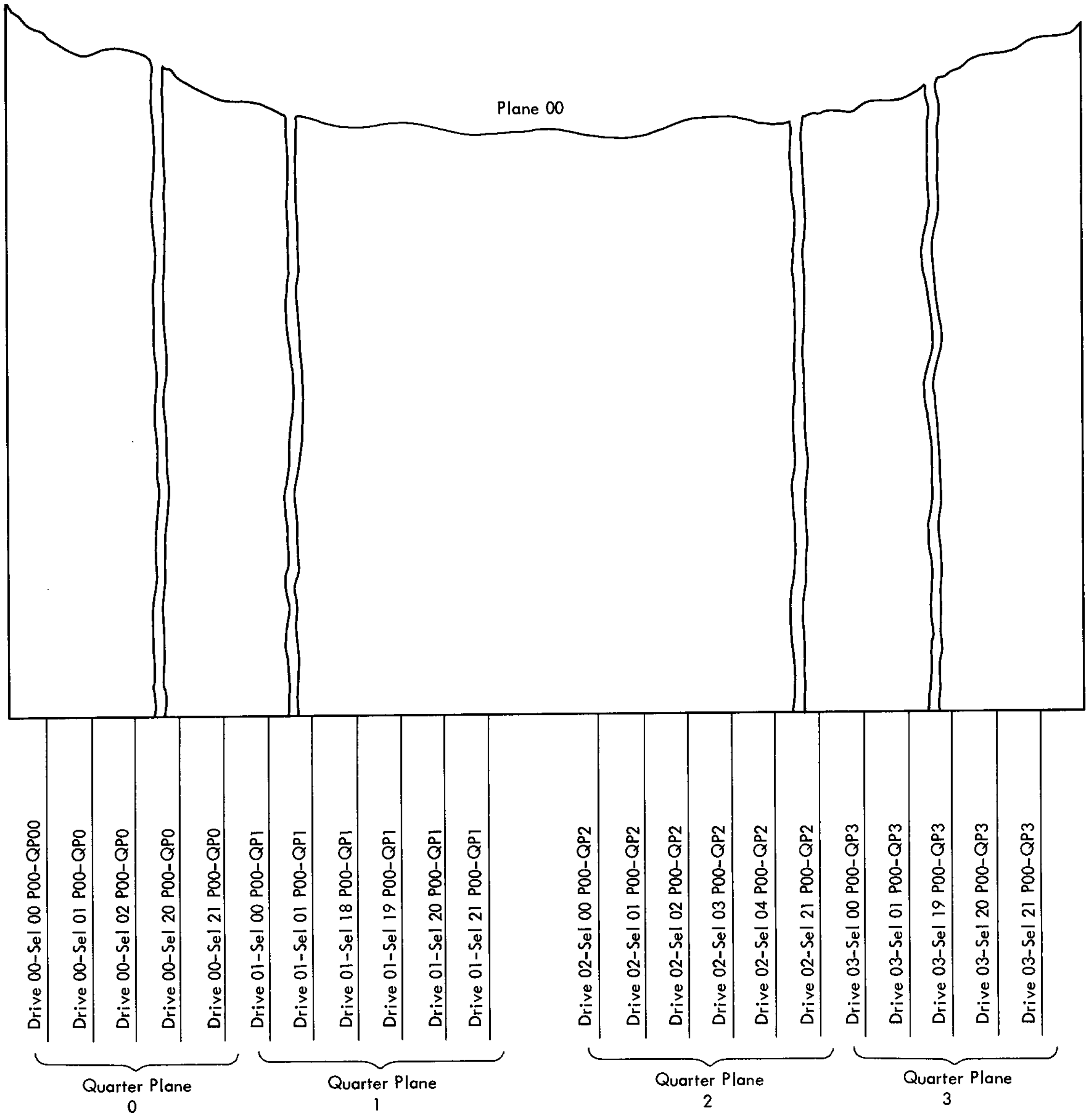


FIGURE 31. CROS PLANE DRIVE LINE ENTRY

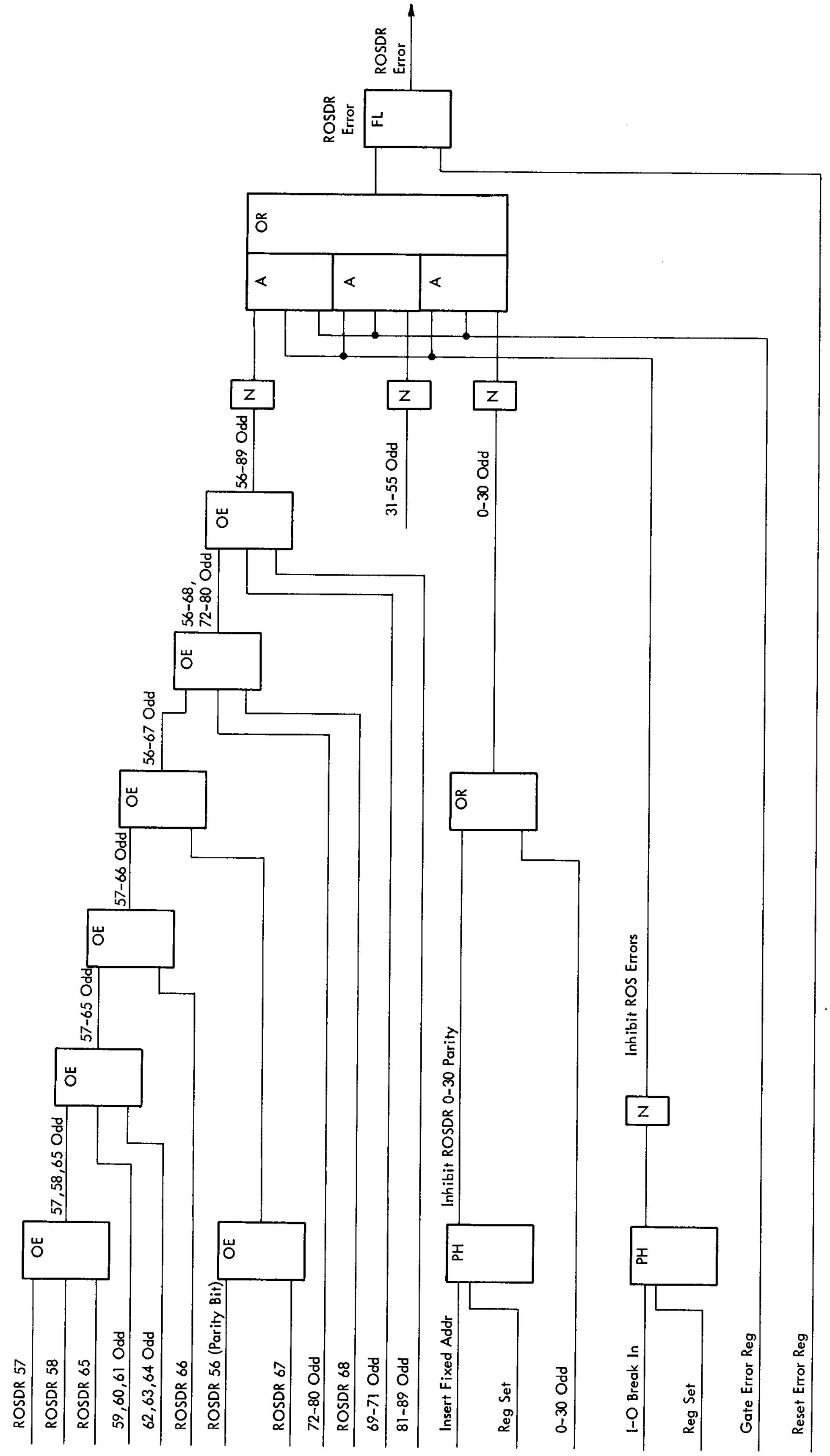


FIGURE 32. PARITY CHECKING ROSDR

- Fixed microprogram routines control the system operations.

A CROS word is read out and stored in ROSDR every machine cycle. This CROS word, the microprogram word, controls the processing unit's gates and control lines during the machine cycle that the word is stored in ROSDR. The address of the next microprogram word is gated into ROAR from ROSDR, or data flow paths under ROSDR control, or from fixed address insertion circuits.

A series of microprogram words cause the system to perform a specific routine. When the machine is manufactured, a microprogrammer specifies which bits each microprogram word must contain to make the routines work. The microprogrammer's routines are used to make the CROS bit plates and the CAS logic diagrams.

MICROPROGRAM INSTRUCTION WORD

- Fields within the microprogram word control the address of the next microprogram word.

- Fields within the microprogram word control the gates and control lines for one machine cycle.

A microprogram word is one CROS word. To control the system, the microprogram word must be read into ROSDR.

The microprogram word is divided into control fields. The control fields are used to address the next CROS word and to control the system's gates and control lines.

Control Fields

The control fields and their functions are shown in Figure 19). The bit combination within a control field is decoded into a micro-order. Only one micro-order is possible from a control field for any one microprogram word. However, one micro-order can control a number of control lines and gates.

Micro-Orders

A micro-order represents a specific bit combination within a control field. The microprogrammer writes the microprogram routines by specifying the micro-orders for each microprogram word. The CAS logic diagram is a listing of the micro-orders specified by the microprogrammer for each microprogram word. System/360 Model 50 Diagrams Manual, Z22-2833, shows a listing of all the micro-orders and their functions.

CAS Logic Diagrams (CLD's)

The CAS diagrams are control automation system diagrams that document the microprogram routines. Each block on the CAS diagrams represents one microprogram word. The micro-orders that were specified by the microprogrammer are shown in the CAS block. Control fields that have no micro-order specified by the microprogrammer are automatically set to a predetermined bit configuration when the bit plate is manufactured. These automatically inserted bits usually represent a no-operation condition for the control field. However, some control fields are set to a pertinent micro-order when no other micro-order has been specified. These control fields are shown in Figure 34.

Figure 35 shows the basic layout of a CAS logic diagram block with each line labeled as to its purpose. Also shown in Figure 34 is a list of special character definitions as used in the CAS logic diagram block.

Figure 36 shows, specifically, the control field mnemonics that may appear in any given print position of the CAS logic diagram block.

A typical CAS logic diagram (word XTTD0) is shown in Figure 33. Beginning at the top:

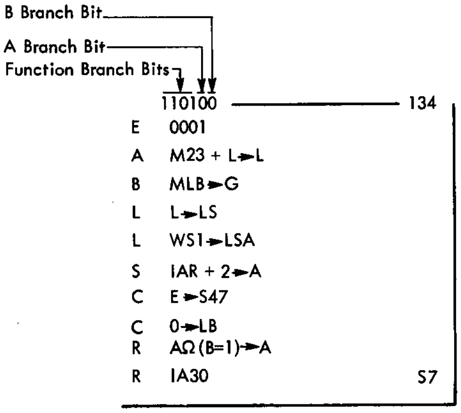


FIGURE 33. TYPICAL CAS DIAGRAM BLOCK (WORD XTTD0)

Item E - Is the bit configuration of the emit field.

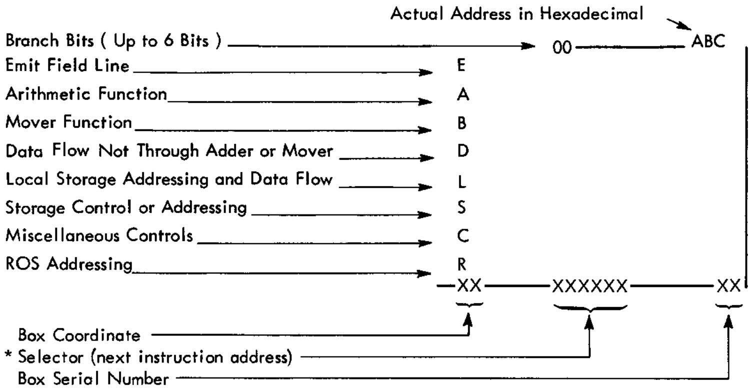
Item A - This micro-order true adds the right half of the M register to the L register and stores the result in the L register.

Item B - The micro-order gates the byte of the M register specified by the L byte counter through the mover to the G register.

Item L (1) - Is a micro-order which causes the sum that was just stored in the L register to be stored in local storage.

Field Bits	Micro-Order	Field Function	Micro-Order Function
TC = 1	+	True or Complement Add	True Add
AD = 1	BLANK	Adder Function	Binary Add, No Carry Saved
AL = 0	BLANK	Adder Latch Input	Shift Zero
UL = 1	UL→WL	Mover Action Left Digit	U 0-3 to W 0-3
UR = 1	UR→WR	Mover Action Right Digit	U 4-7 to W 4-7
UP = 2	-1	Byte Counter Function Control	Count the selected counter down
WS = 4	FN,J→LSA	Local Store Addressing (CPU)	Gate LSFN and J to LSAR
CS = 0	2,3	Local Store Sector (I-O)	Select the sector for the L and R backup area
SA = 0	0→LSA	Local Store Addressing (I-O)	Gate the I-O function + zero to LSAR Note: Used in conjunction with CS = 0, this micro-order addresses the R register backup word
MG = 3	BFR1→BIB	Multiplex Gate Control (I-O)	Gate Buffer 1 to MXB1B

FIGURE 34. NON-NOP AUTOMATIC MICRO-ORDERS


Actual Address in Hexadecimal
Branch Bits (Up to 6 Bits)
00
ABC
Emit Field Line
E
Arithmetic Function
A
Mover Function
B
Data Flow Not Through Adder or Mover
D
Local Storage Addressing and Data Flow
L
Storage Control or Addressing
S
Miscellaneous Controls
C
ROS Addressing
R
XX
XXXXXX
XX
Box Coordinate
* Selector (next instruction address)
Box Serial Number

Special Character Definitions

+	True Add or Positive	∀	Is "Exclusive OR'ed" with
-	Complement Add or Negative	Ω	Is OR'ed with
=	Is Equal	/	Or (non-logical)
≠	Is Unequal	<	Is less than
→	Is Set Into (Destructive Read-In is implied)	>	Is greater than
.	Is AND'ed with (logical)	¬	Not (Boolean-used as the not function on CLD's)
,	And (non-logical)	:	Is compared to
?	Indeterminate function (hardware controlled)		

* Any Selector Bit Equals:

0 Branch Bit	0
1 Branch Bit	1
Test For Branch Bit	*
No Test For Branch Bit (don't care)	X

FIGURE 35. CAS DIAGRAM BLOCK (BASIC LAYOUT)

Model 50 CPU Mode

	Left Edge Char		Print Positions												Right Edge Char
			1	2	3	4	5	6	7	8	9	10	11	12	
Print Line #2	E	X	CE	CE	CE	CE					C	P	U		
3	A	X	RY	RY	RY	TC	LX	LX	LX	→	TR	TR	TR	TR	
4	A	X	AD	AD	AD	AD		AL	AL	AL	AL	AL	AL	AL	
5	B	X	LU	LU	LU	LU	LU	LU		MV	MV	MV	MV	MV	
6	B	X	UL	UL	UL	UL	UL		UR	UR	UR	UR	UR		
7	B	X		WM	WM	WM	WM	WM	WM	WM	WM	WM	WM		
8	D	X	AL	AL	AL	AL	→	TR	TR	TR	TR				
9	D	X	DG	DG	DG	DG	DG	DG	DG	DG	DG	DG	DG	DG	
10	D	X	MB/UP	MB/UP	,/MB	LB/MB	LB/,	,/LB	MD/LB	MD/,	MD/MD	UP/MD	UP/MD		
11	L	X	WS	WS	WS	WS	WS	WS	WS	WS	WS	WS	WS		
12	L	X	SF	SF	SF	SF	SF	SF	SF	SF	SF				
13	S/C	X	IV	IV	IV	IV	IV	IV	IV	IV	IV				
14	C	X	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	
15	R	X	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZF			
16	R	X	AB	AB	AB	AB	AB	AB		BB	BB	BB	BB	BB	
17		X													

Model 50 I/O Mode #1

	Left Edge Char		Print Positions												Right Edge Char
			1	2	3	4	5	6	7	8	9	10	11	12	
Print Line #2	E	X	CE	CE	CE	CE						I	O		
3	A	X	RY	RY	RY	TC	LX	LX	LX	→	TR	TR	TR	TR	
4	A	X	HC	HC	HC	HC		AL	AL	AL	AL	AL	AL	AL	
5	B	X	LU	LU	LU	LU	LU		MX	MX	MX	MX	MX		
6	B	X	UL	UL	UL	UL	UL		UR	UR	UR	UR	UR		
7	B	X	WL	WL	WL	WL	WL	WL	WL	WL	WL	WL	WL		
8	D	X	MG	MG	MG	MG	MG	MG	MG	MG	MG				
9	L	X	CS	CS	CS	CS	CS	CS	SA	SA	SA	SA	SA	SA	
10	L	X	SF	SF	SF	SF	SF	SF	SF	SF	SF				
11	C/S	X	CT	CT	CT	CT	CT	CT	CT	CT	CT	CT			
12	C	X	CL	CL	CL	CL	CL	CL	CL	CL	CL				
13	C	X	CG	CG	CG	CG	CG		CC	CC	CC	CC	CC	CC	
14	C	X	MS	MS	MS	MS	MS	MS	MS	MS	MS	MS	MS	MS	
15	C	X	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	
16	R	X	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZF			
17	R	X	AB	AB	AB	AB	AB	AB		BB	BB	BB	BB	BB	

Model 50 I/O Mode #2

	Left Edge Char		Print Positions												Right Edge Char
			1	2	3	4	5	6	7	8	9	10	11	12	
Print Line #2	E	X	CE	CE	CE	CE						I	O		
3	A	X	RY	RY	RY	TC	LX	LX	LX						
4	B	X	LU	LU	LU	LU	LU		MX	MX	MX	MX	MX		
5	B	X	UL	UL	UL	UL	UL		UR	UR	UR	UR	UR		
6	B	X	WL	WL	WL	WL	WL	WL	WL	WL	WL	WL	WL		
7	D	X	TR/AL	TR/AL	TR/AL	TR/AL	TR/→	TR	TR	TR	TR				
8	D	X	MG	MG	MG	MG	MG	MG	MG	MG	MG				
9	L	X	CS	CS	CS	CS	CS	CS	SA	SA	SA	SA	SA	SA	
10	L	X	SF	SF	SF	SF	SF	SF	SF	SF	SF				
11	C/S	X	CT	CT	CT	CT	CT	CT	CT	CT	CT	CT			
12	C	X	CL	CL	CL	CL	CL	CL	CL	CL	CL				
13	C	X	CG	CG	CG	CG	CG	·	CC	CC	CC	CC	CC	CC	
14	C	X	MS	MS	MS	MS	MS	MS	MS	MS	MS	MS	MS	MS	
15	C	X	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	
16	R	X	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZN/ZF	ZF			
17	R	X	AB	AB	AB	AB	AB	AB		BB	BB	BB	BB	BB	

FIGURE 36. CAS DIAGRAM BLOCK -- SPECIFIC LAYOUTS

Item L (2) - This micro-order specifies the area of local storage to use (working-storage-one).

Item S - Is the micro-order which causes the IAR to be incremented by two (refetch is off) and sent to SAR.

Item C (1) - This micro-order causes general purpose stats 4-7 to be set from the emit field (in this case, stats 4-6 are set off, and stat 7 is set on).

Item C (2) - This micro-order sets the L byte counter to zero.

Note: Registers are set at the beginning of the next machine cycle. Therefore, setting the L byte counter to zero does not affect which byte of the M register is gated out.

Item R (1) - Is the micro-order which forces the A-branch bit to one if the B-branch bit is turned on by its test.

Items R (2) - This micro-order sets the A-branch bit to one if IAR (30) contains a one.

INSTRUCTION-FETCH ROUTINE

- I-Fetch reads and decodes the program instructions.
- Instructions start on halfword boundaries.
- The op-buffer saves the odd instruction-halfword of a storage read.
- The refetch latch records the possibility that the data in the op-buffer is invalid because of the execution of the last instruction.

The instruction fetch routine (I-fetch) reads the instruction from storage, decodes the instruction, and branches to the proper execute routine.

The instruction address register (IAR) contains the address of the first byte of the instruction. Instructions can start on a word boundary (IAR holds an even halfword address) or on a halfword boundary (IAR holds an odd halfword address). Instructions can be a halfword, word, or three halfwords long.

The last halfword of many instructions will be the first halfword of a storage word; the last half of the storage word is the first halfword of the next instruction. To save machine cycles, the halfword that belongs to the next instruction is stored in a reserved local storage position called the op buffer.

A refetch latch is turned on if the execution of the present instruction could possibly store data in the storage address positions of the next instruction. If the refetch latch is on, I-fetch will not use the halfword stored in the op-buffer. Instead, I-fetch reads the entire instruction from storage.

Figure 37 shows the instruction formats and the source of the instruction halfwords for all conditions.

The instruction routine is divided into two sections, first level I-fetch and second level I-fetch. First level I-fetch determines the instruction format. Second level I-fetch determines the specific operation to be performed.

First Level I-Fetch

- Stores the first two instruction halfwords in the M register (one halfword for RR format).
- Sets the instruction length counter (ILC).
- Stores the data from the local storage address specified by the B field in the L register.
- Decodes the instruction format from the first four bits of the op-code.

Figure 38 shows a logic flow chart of first level I-fetch. A more detailed flow chart of first level I-fetch is in System/360 Model 50 Diagrams Manual, Form Z22-2833.

The data flow in first level I-fetch depends on three conditions: the IAR address, the refetch latch, and the format of the instruction.

The IAR address determines the source of the first instruction halfword. If IAR contains an even address, the storage word is read out and stored in both the M register and the op buffer. If IAR contains an odd address, the refetch latch determines the source of the first instruction halfword.

The refetch latch determines whether or not the halfword stored in the op buffer is valid. If the refetch latch is on, the halfword may be invalid; the halfword is reread from storage and stored in the left half of the M register. If the refetch latch is off, the halfword from the op buffer is stored in the left half of the M register.

The instruction length in the first halfword is checked to determine if the instruction is in RR format. If the instruction is in RR format, the M register contains the entire instruction. If the instruction is not in RR format, the second instruction halfword is read from storage. The left half of the storage word (second instruction halfword) is stored in the right half of the M register, and the entire storage word is stored in the op buffer to save the odd halfword.

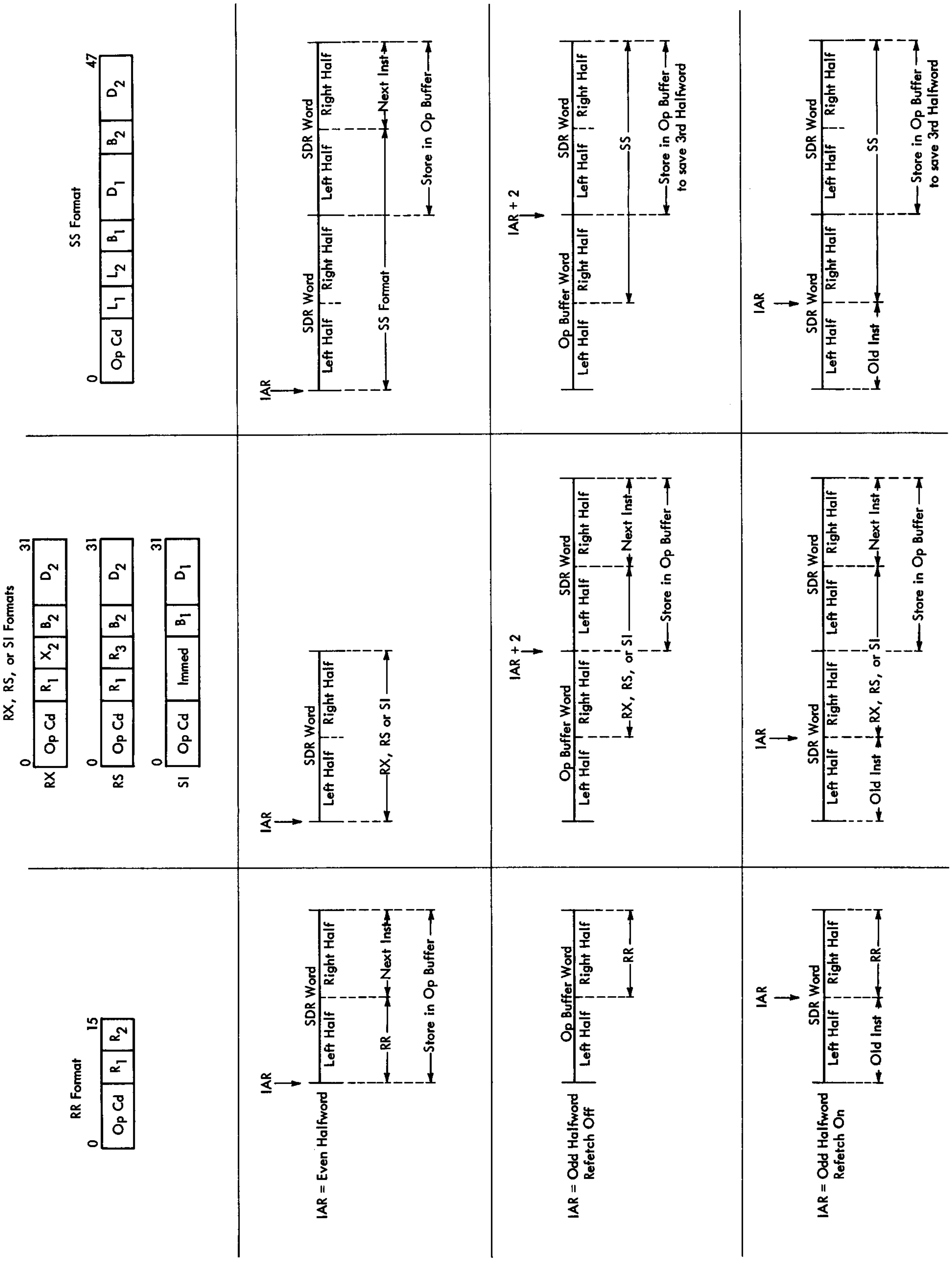

RR Format
0 Op Cd R1 R2 15
RX, RS, or SI Formats
RX 0 Op Cd R1 X2 B2 D2 31
RS 0 Op Cd R1 R3 B2 D2 31
SI 0 Op Cd Immed B1 D1 31
SS Format
0 Op Cd L1 L2 B1 D1 B2 D2 47
IAR = Even Halfword
IAR = Odd Halfword Refetch Off
IAR = Odd Halfword Refetch On
IAR
IAR + 2
SDR Word
Op Buffer Word
Left Half
Right Half
RR
Next Inst
Store in Op Buffer
RX, RS or SI
RX, RS, or SI
SS Format
SS
Old Inst
Store in Op Buffer to save 3rd Halfword

FIGURE 37. INSTRUCTION FORMATS AND SOURCE OF INSTRUCTION HALFWORDS

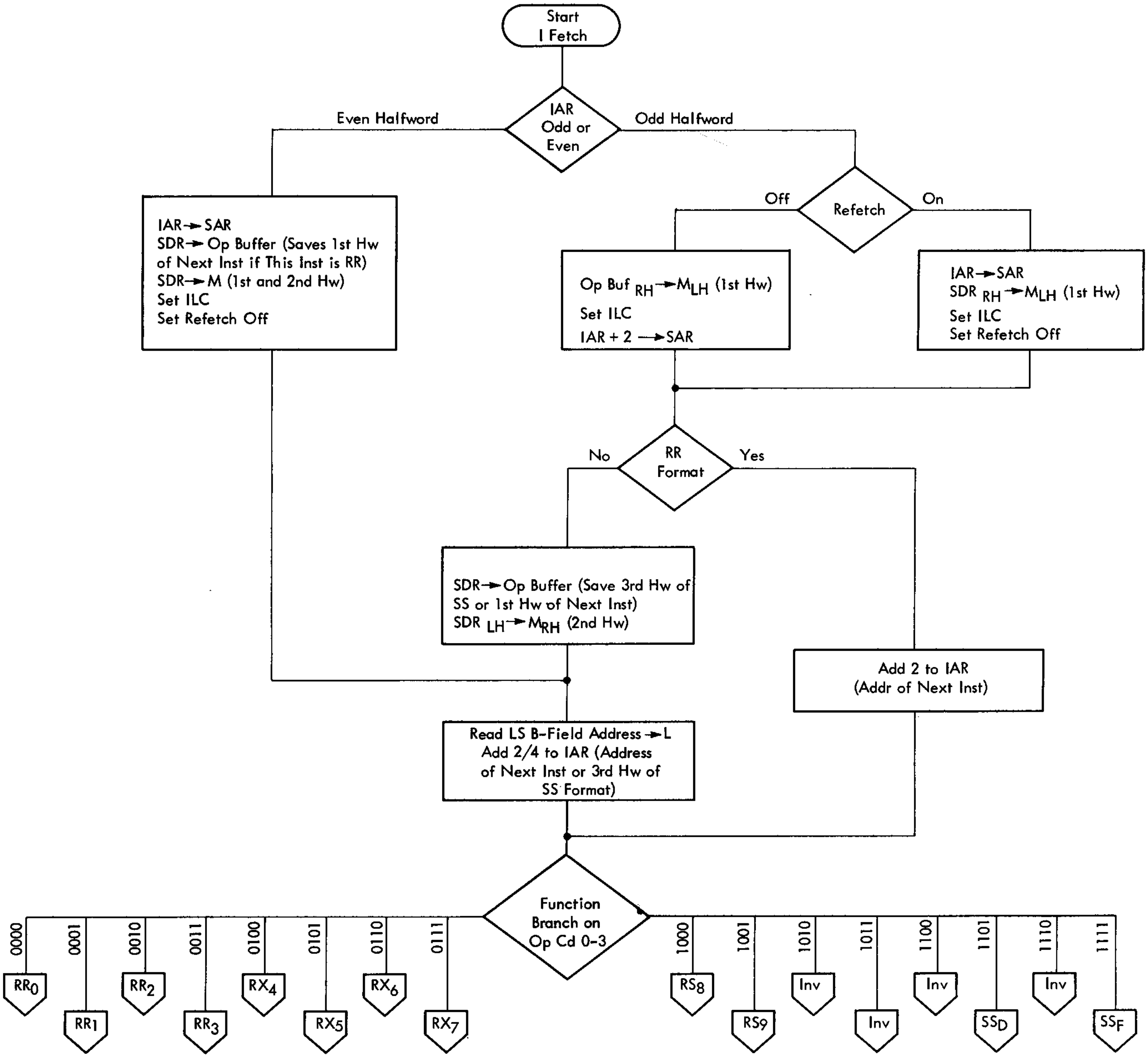


FIGURE 38. FIRST LEVEL I-FETCH

After the first two instruction halfwords are stored in the M register, the address specified by the B-field is read from local storage and stored in the L register. A function branch is taken, on the first four bits of the op code, to the proper format second level I-fetch.

Second Level I-Fetch

- Second level I-fetch differs for each format.
- Second level I-fetch fetches the third instruction halfword for SS formats.
- Second level I-fetch prepares the data addresses.
- Second level I-fetch saves the IAR address if the format can contain a branch and link instruction code.
- Second level I-fetch branches to the proper microprogram routine to execute the instruction.
- Unassigned op-code combinations cause a branch to a program trap routine.

Second level I-fetch completes the decoding of the instruction, stores the IAR address if the instruction format could result in a branch instruction, and branches to the proper execute microprogram.

Figures 39 and 40 show the second level I-fetch flow paths for all valid and invalid op-codes (invalid op-codes are the unassigned configurations of the eight-bit op-code field).

RR Formats: The RR0 format (0X codes) stores the contents of the general purpose register specified by the R2 field in the R-register. The contents of IAR are stored in the H-register to save the next instruction address if this instruction results in a branch operation.

RR formats 1-3 (1X-3X codes) store the contents of the general purpose register specified by the R2 field in the L-register.

RR formats 2-3 (2X-3X codes) set the carry stat as a signal to the floating-point routine whether to operate in floating-point long or floating-point short mode.

RX Formats: The RX4 format (4X codes) computes the effective address by adding the contents of the general purpose registers specified by the B and X fields to the displacement field, and stores the result in the L-register. The general purpose register specified by the R1 field is stored in the R-register. The contents of IAR are stored in the H-register to save the next instruction address if this instruction results in a branch operation.

The RX5 format (5X codes) computes the effective address by adding the contents of the general purpose registers specified by the B and X fields to the contents of the displacement field, and stores the result in the R-register. The contents of the general purpose register specified by the R1 field are stored in the L-register.

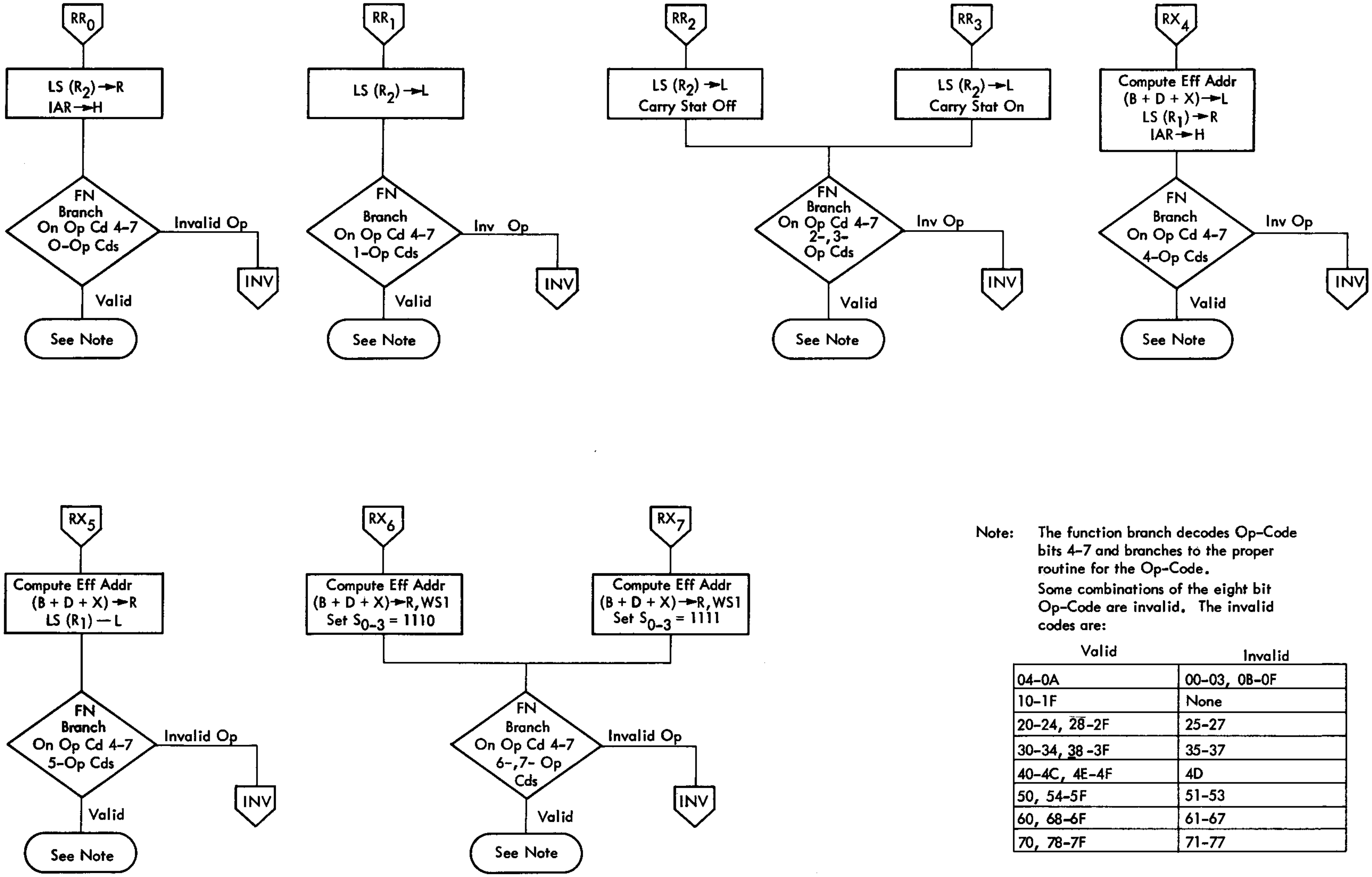
RX formats 6-7 (6X-7X codes) compute the effective address by adding the contents of the general purpose registers specified by the B and X fields to the contents of the displacement field, and stores the result in the R-register and WS1 of local storage. General purpose stat 3 is set as a signal to the floating-point routine whether to operate in floating-point long or short mode.

RS/SI Formats: The RS8 formats (8X codes) compute the effective address by adding the contents of the general purpose register specified by the B-field to the contents of the displacement field, and store the result in the R-register. The contents of the general purpose register specified by the R1 field are stored in the L-register.

The RS9 format (9X codes) computes the effective address by adding the contents of the general purpose register specified by the B-field to the contents of the displacement field, and stores the result in the R-register and in WS1 of local storage.

SS Formats: The SSD format (DX codes) computes the effective operand 1 address (B1 + D1) and stores the result in the R-register and in WS 1 of local storage. The L1 and L2 fields are stored in the G-register. The third halfword is read into the M-register (positions 16-31), and the effective operand 2 address is computed and stored in the L-register and in WS2 of local storage. The byte position of the operand 1 address is stored in the MB counter to use in locating the first data-byte-position in the operand 1 word.

The SSF format (FX codes) computes the effective operand 1 address (B1 + D1) and stores the result in the L-register and in WS9 of local storage. The L1 and L2 fields are stored in the G-register. The third halfword is read into the right half of the M-register, and the operand 2 effective address is computed and stored in WSA of local storage. The low-order address of the operand 1 field is computed and stored in WS1; the byte position of the low-order address is stored in the MB counter. The low-order address of the operand 2 field is computed and stored in WS2; the byte positon of the low order address is stored in the LB counter.

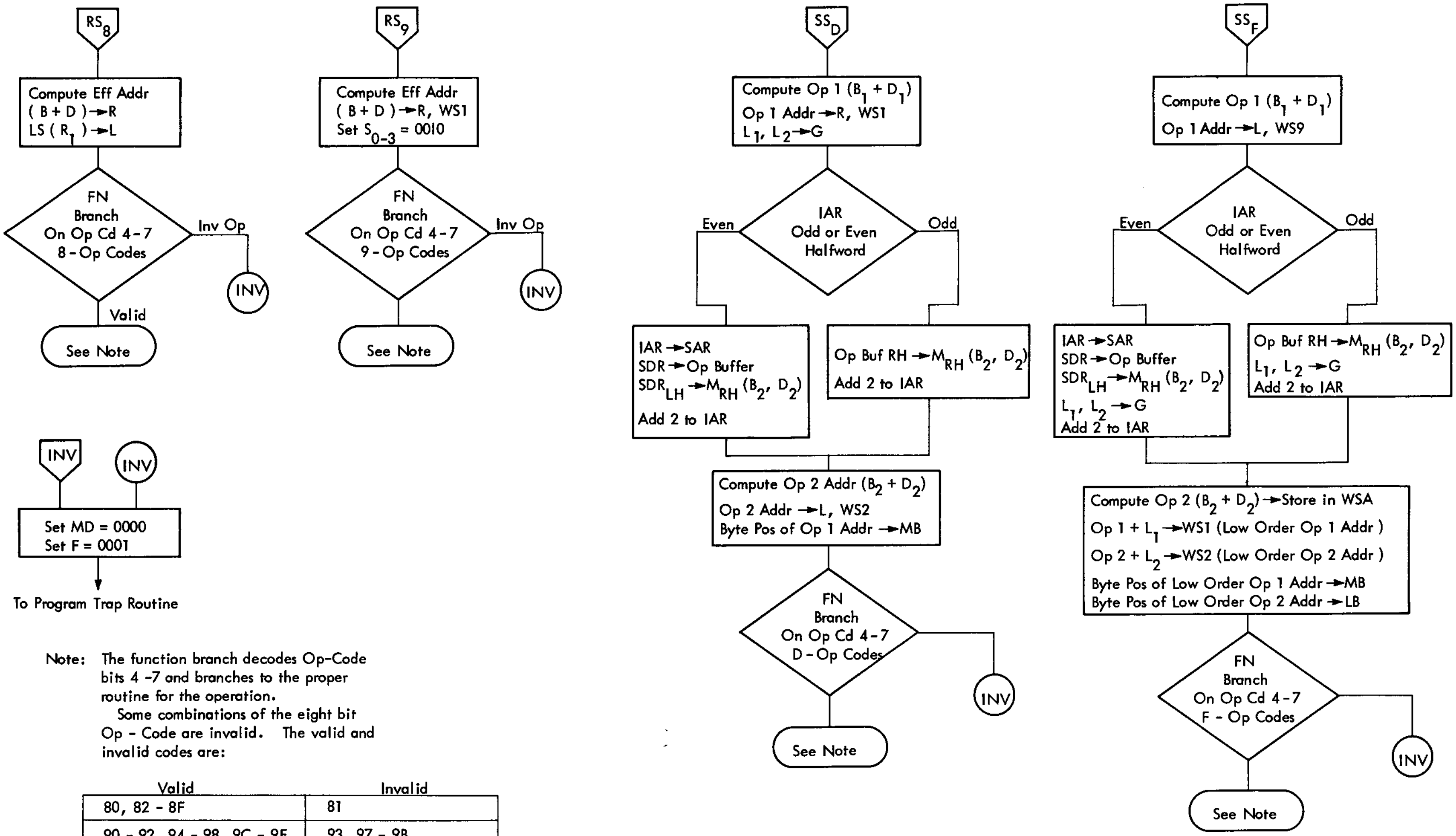


Note: The function branch decodes Op-Code bits 4-7 and branches to the proper routine for the Op-Code.

Some combinations of the eight bit Op-Code are invalid. The invalid codes are:

Valid	Invalid
04-0A	00-03, 0B-0F
10-1F	None
20-24, 28-2F	25-27
30-34, 38 -3F	35-37
40-4C, 4E-4F	4D
50, 54-5F	51-53
60, 68-6F	61-67
70, 78-7F	71-77

FIGURE 39. SECOND LEVEL I-FETCH RR AND RX FORMATS



Note: The function branch decodes Op-Code bits 4 -7 and branches to the proper routine for the operation.

Some combinations of the eight bit Op - Code are invalid. The valid and invalid codes are:

Valid	Invalid
80, 82 - 8F	81
90 - 92, 94 - 98, 9C - 9F	93, 97 - 9B
D1 - D7, DC - DF	D0, D8 - DB
F1 - F3, F8 - FD	F0, F4 - F7, FE - FF

FIGURE 40. SECOND LEVEL I - FETCH RS/SI AND SS FORMATS

INDEX