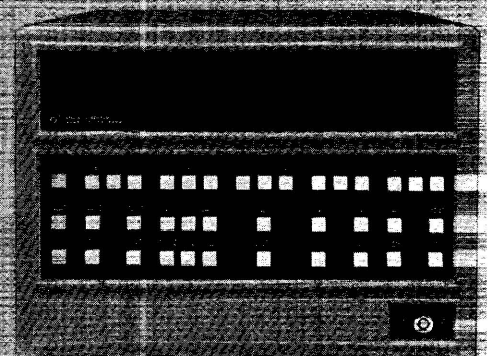
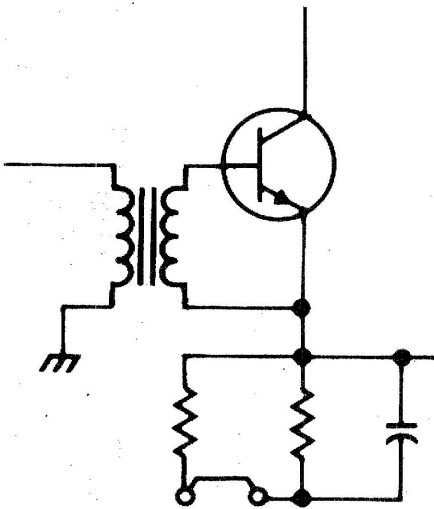
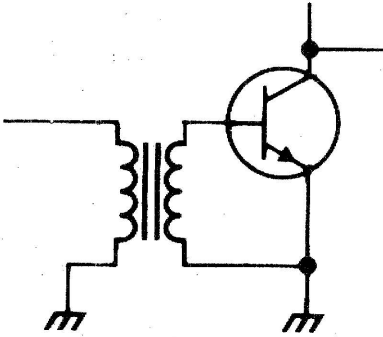




2100 Computer Hardware Training Manual

HEWLETT - PACKARD
COMPUTER MAINTENANCE COURSE

2100 Computer
Hardware Training Manual

STOCK NO. (02100-90159)

-NOTICE-

The information contained in this manual is for training purposes only. Consult the Hewlett-Packard documentation supplied with the computer for current information concerning the specific computer system furnished.

The information contained in this publication may not be reproduced in any form without the expressed consent of the Hewlett-Packard Company.

COPYRIGHT HEWLETT-PACKARD COMPANY 1972

11000 Wolfe Road, Cupertino, California 95014 Area Code 408 257-7000 TWX 910-338-0221

LESSON INDEX

introduction to hardware

1

2100A architecture and block diagrams

2

microprogramming and central processor unit

3

memory

4

power supply

5

input/output and direct memory access

6

front panel

7



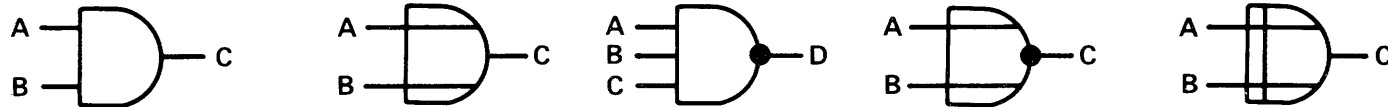
LESSON 1

INTRODUCTION TO HARDWARE

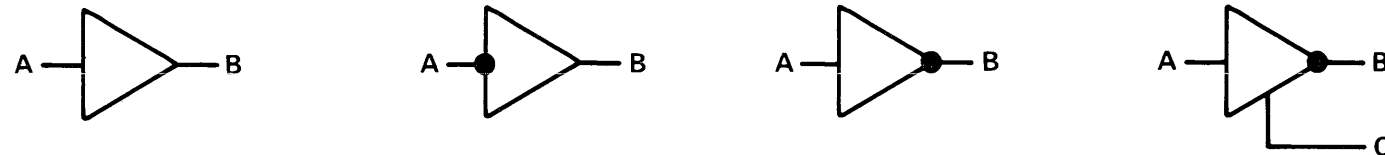
Four Major Classes of HP Logic Symbols	1-1	Delay Flip Flop	1-11
The "AND" Gate	1-2	Cross Coupled "NAND" and "NOR"	1-12
The "NAND" Gate	1-3	Gates	
The "IOR" Gate (Inclusive "OR")	1-4	Integrated Circuit Diagrams	1-13
The "NOR" Gate	1-5	Boolean Algebra Is	1-14
The "XOR" Gate (Exclusive "OR")	1-6	Eight Basic Identities	1-15
Comparison of Common TTL and	1-7	Ten Identities Unique to Boolean Algebra	1-16
CTL Gates		The Logic Equation	1-17
The R-S Flip Flop With Clock	1-8	Problems	1-18
The J-K Flip Flop	1-9	Logic Equations	1-19
Latching Flip Flop	1-10	Logic Equations (Cont'd.)	1-20

FOUR MAJOR CLASSES OF HP LOGIC SYMBOLS

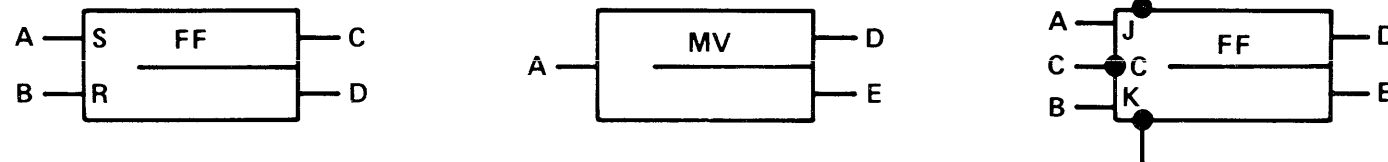
1. GATES



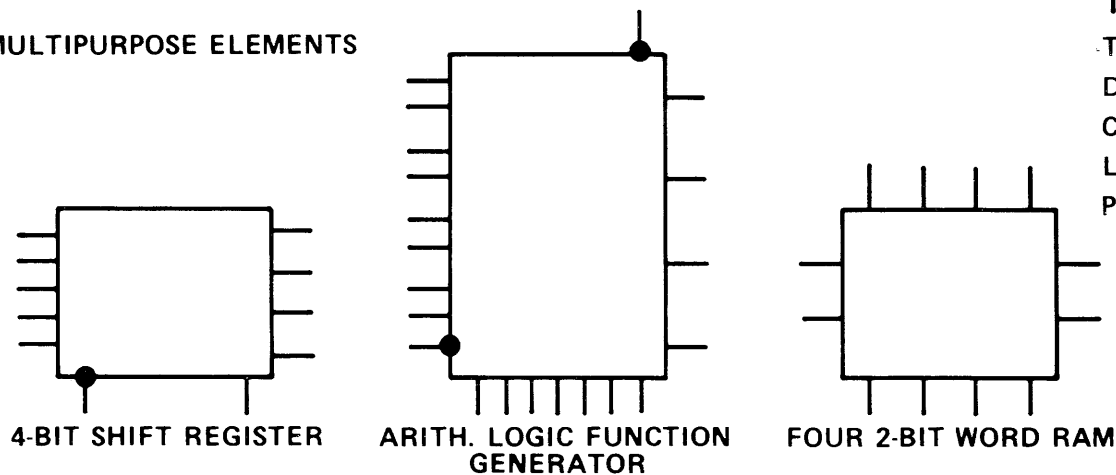
2. AMPLIFIERS



3. SWITCHING ELEMENTS



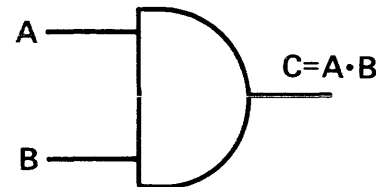
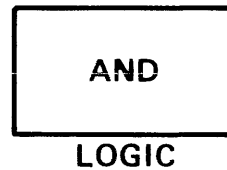
4. MULTIPURPOSE ELEMENTS



THE SYMBOL INCLUDES A DESCRIPTIVE NAME INDICATING THE OVERALL LOGIC FUNCTION PERFORMED.

THE "AND" GATE

"AND" PROVIDES LOGICAL MULTIPLICATION



THE OUTPUT "C" IS TRUE (POSITIVE) ONLY
WHEN "A" AND "B" ARE BOTH TRUE (POSITIVE)

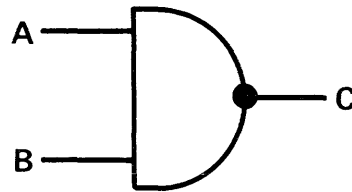
THE ARITHMETIC MULTIPLICATION SIGN ($\cdot$) ALWAYS INDICATES "AND" IN LOGIC EQUATIONS AND CIRCUITS. THE TERM $C=A \cdot B$ IS A LOGIC EQUATION MEANING BOTH "A" AND "B" MUST BE TRUE FOR "C" TO BE TRUE.

TRUTH TABLE

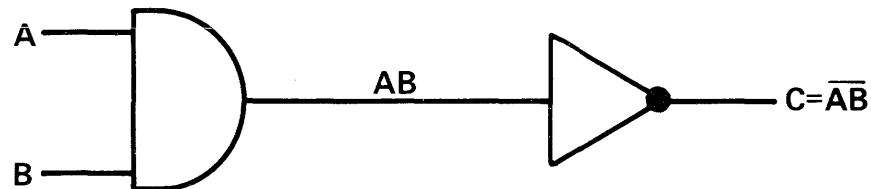
A	B	OUTPUT $C=A \cdot B$
0	0	0
1	0	0
0	1	0
1	1	1

THE "NAND" GATE

PROVIDES THE LOGICAL OPERATION "NOT-AND"



THE OUTPUT "C" IS FALSE (NOT) ONLY WHEN BOTH "A" AND "B" ARE TRUE.
THUS THE "NAND" GATE CAN BE CONSIDERED AN "AND" GATE
FOLLOWED BY AN INVERTER.

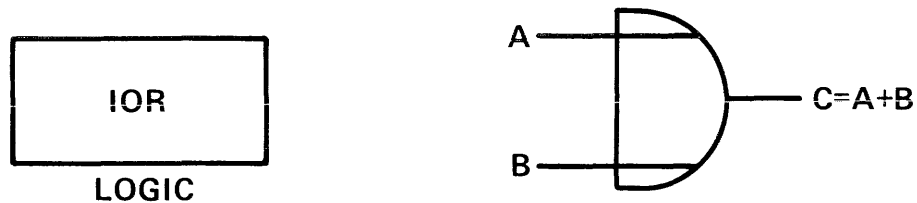


TRUTH TABLE

A	B	C
0	0	1
1	0	1
0	1	1
1	1	0

THE "IOR" GATE (INCLUSIVE "OR")

"IOR" PROVIDES LOGICAL ADDITION



THE OUTPUT "C" IS TRUE (POSITIVE) WHEN
"A" OR "B" OR "BOTH" ARE TRUE (POSITIVE).

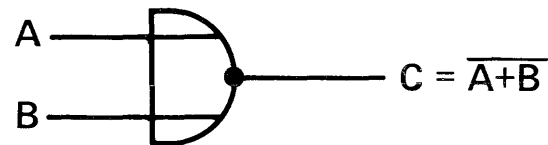
THE ARITHMETIC ADDITION SIGN (+) ALWAYS INDICATES "OR" IN
LOGIC EQUATIONS AND CIRCUITS. THE TERM $C=A+B$ IS A LOGIC
EQUATION MEANING EITHER "A" OR "B", OR "BOTH", MUST BE
TRUE FOR "C" TO BE TRUE.

TRUTH TABLE

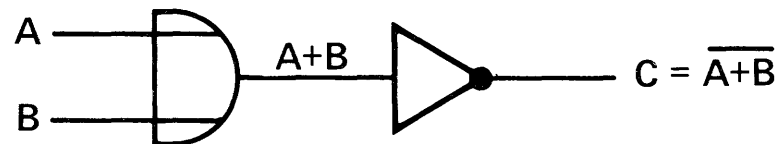
A	B	OUTPUT $C=A+B$
0	0	0
1	0	1
0	1	1
1	1	1

THE "NOR" GATE

THE "NOR" GATE PROVIDES THE LOGICAL OPERATION "NOT-OR"



THE OUTPUT "C" IS FALSE (NOT) ONLY WHEN EITHER "A" OR "B" ARE TRUE. THUS THE "NOR" GATE CAN BE CONSIDERED AS AN OR GATE FOLLOWED BY AN INVERTER.

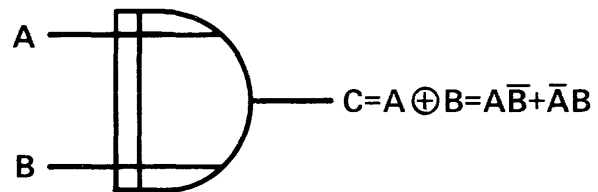


TRUTH TABLE

A	B	C
0	0	1
1	0	0
0	1	0
1	1	0

THE "XOR" GATE (EXCLUSIVE "OR")

PROVIDES THE LOGICAL OPERATION "EXCLUSIVE OR"



THE OUTPUT "C" IS TRUE (POSITIVE) ONLY WHEN "A" OR "B" BUT NOT BOTH ARE TRUE.

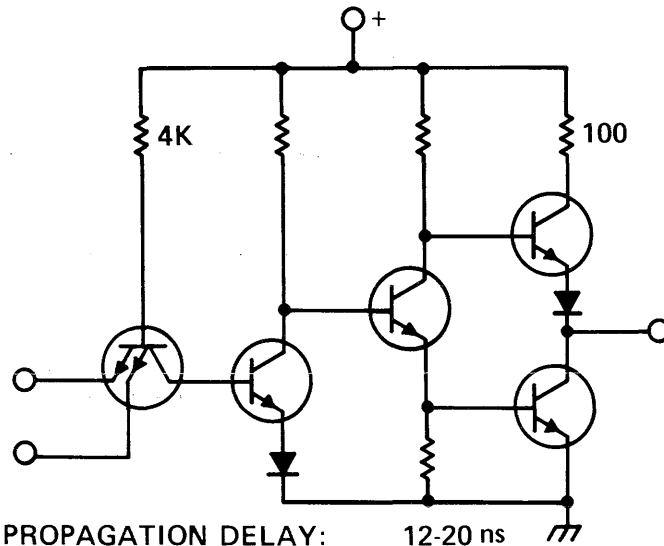
TRUTH TABLE

A	B	$C = \bar{A}B + A\bar{B}$
0	0	0
1	0	1
0	1	1
1	1	0

COMPARISON OF COMMON TTL AND CTL GATES

TTL:

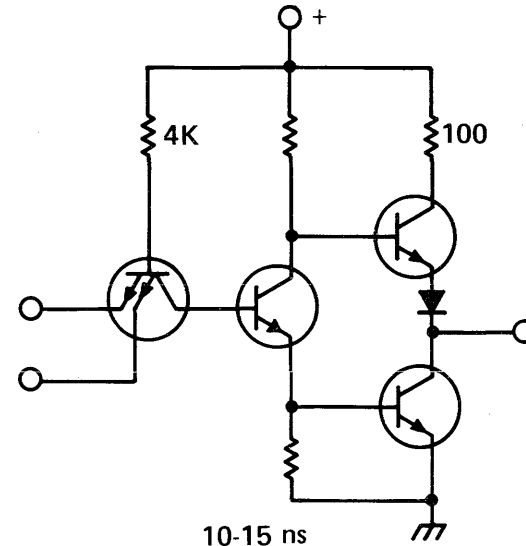
AND-GATE



TYP. PROPAGATION DELAY:
POWER REQUIREMENT:

12-20 ns
10mW

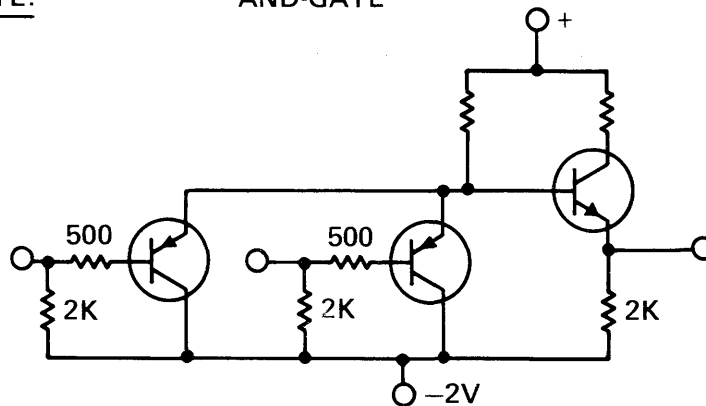
NAND-GATE



10-15 ns
10mW

CTL:

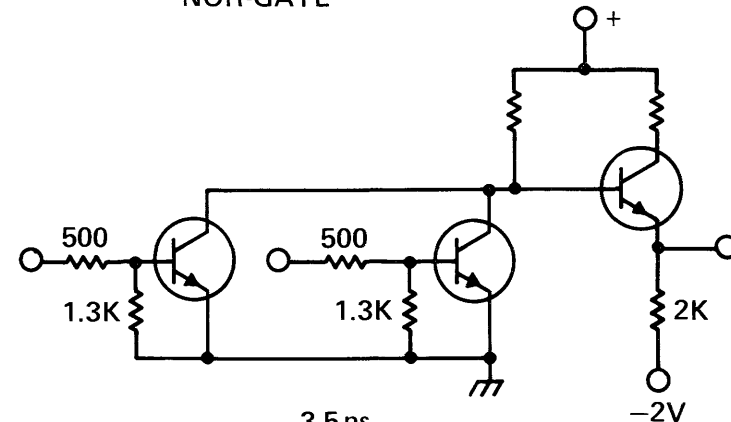
AND-GATE



TYP. PROPAGATION DELAY:
POWER REQUIREMENT:

3-5 ns
250mW

NOR-GATE

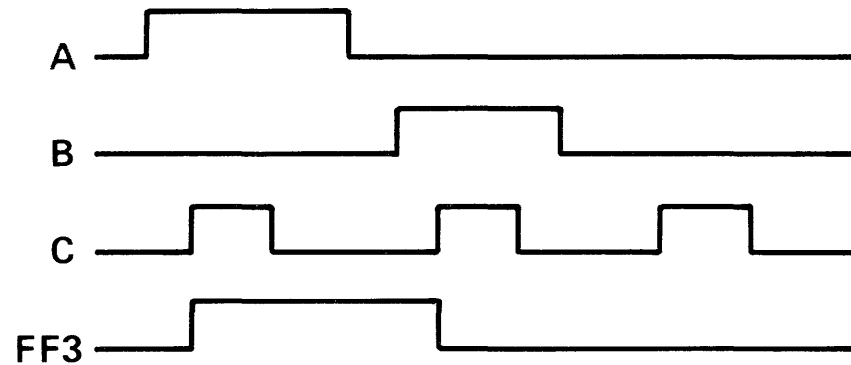


3-5 ns
250mW

THE R-S FLIP FLOP WITH CLOCK



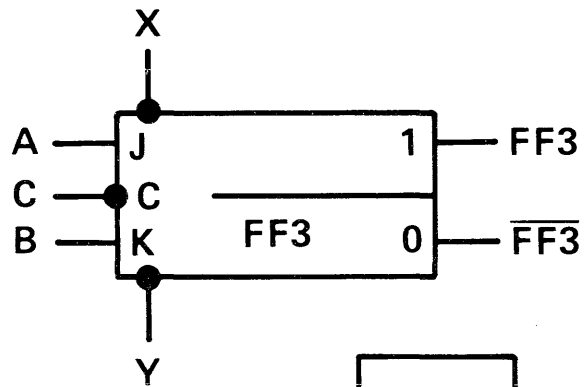
"A" AND "C" ARE REQUIRED TO SET (FF3 TRUE)
 "B" AND "C" ARE REQUIRED TO RESET ($\overline{\text{FF3}}$ TRUE)



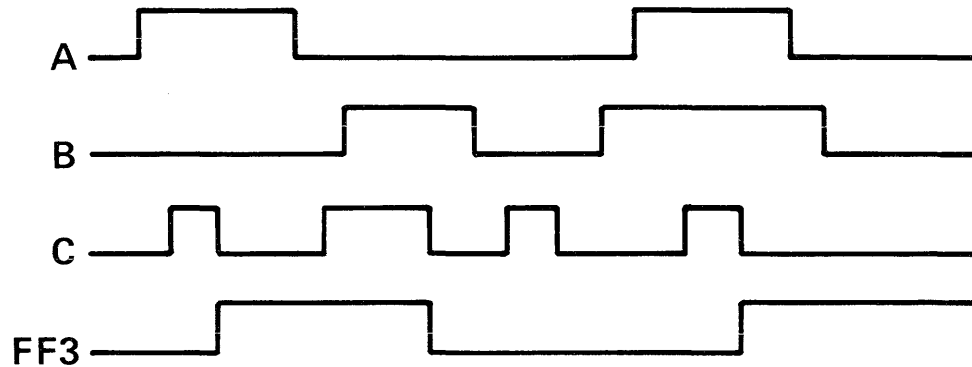
TRUTH TABLE

INPUT		OUTPUT	
A	B	FF3	$\overline{\text{FF3}}$
1	0	1	0
0	1	0	1
0	0	NO CHANGE	
1	1	UNDETERMINED	

THE J-K FLIP FLOP



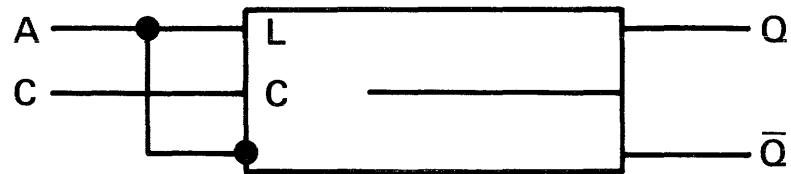
SIMULTANEOUS INPUTS "A" AND "B" REVERSE THE EXISTING STATE $FF3 \rightarrow \overline{FF3}$. DIRECT SET (X) OR DIRECT RESET (Y) WITHOUT CLOCK



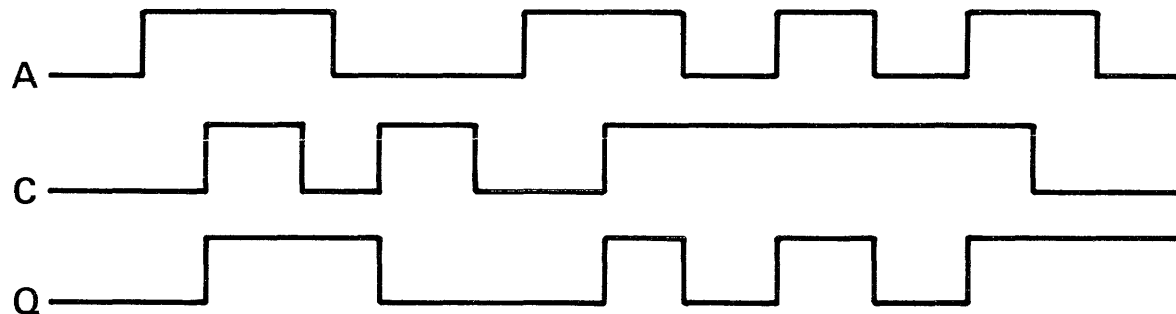
TRUTH TABLE

INPUTS		INITIAL STATE		FINAL STATE	
A	B	FF3	$\overline{FF3}$	FF3	$\overline{FF3}$
1	0	ANY		1	0
0	1			0	1
1	1	0	1	1	0
1	1	1	0	0	1
0	0	ANY		NO CHANGE	

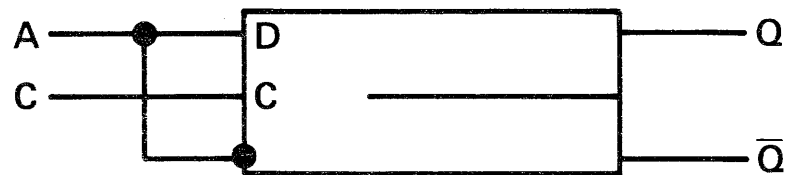
LATCHING FLIP FLOP



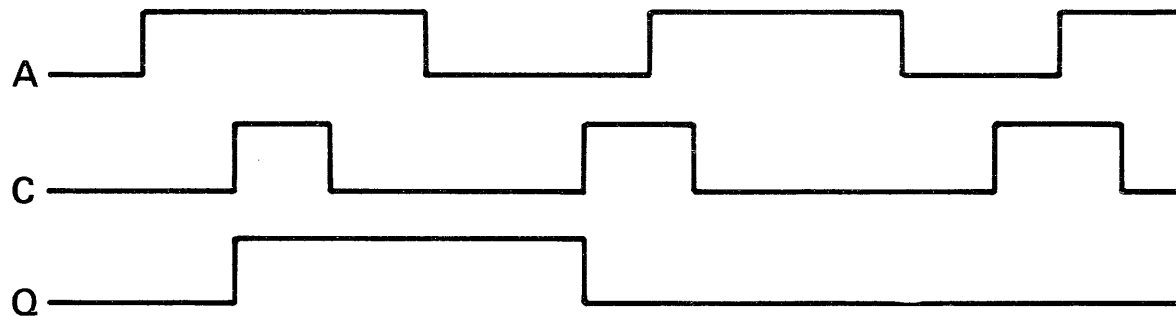
THIS FLIP FLOP WILL FOLLOW INPUT "A" AS LONG AS THE CLOCK INPUT (C) IS TRUE. THE FLIP-FLOP WILL LATCH INTO THE STATE EXISTING AT "A" WHEN THE CLOCK PULSE AT C TRANSITS FROM TRUE TO FALSE.



DELAY FLIP FLOP

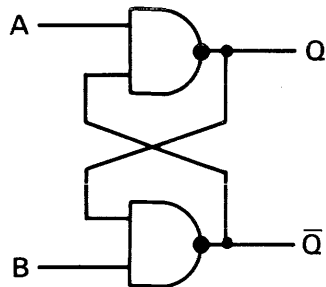


THE CONDITION PRESENT AT THE INPUT "A" WILL BE LOADED INTO THE FLIP FLOP AT THE POSITIVE GOING CLOCK PULSE. CONTRARY TO THE L-FLIP FLOP THE DELAY FLIP FLOP WILL NOT CHANGE ITS STATE WHEN THE CLOCK IS IN A STEADY STATE CONDITION.

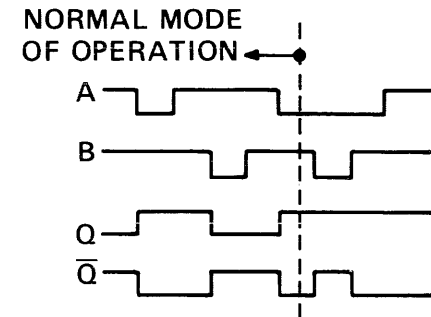


CROSS COUPLED "NAND" AND "NOR" GATES

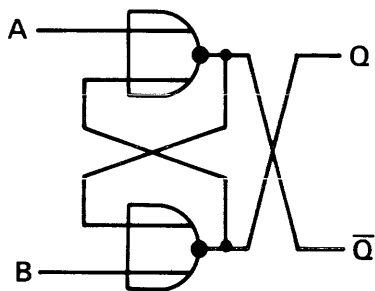
CROSS COUPLED "NAND" GATES:



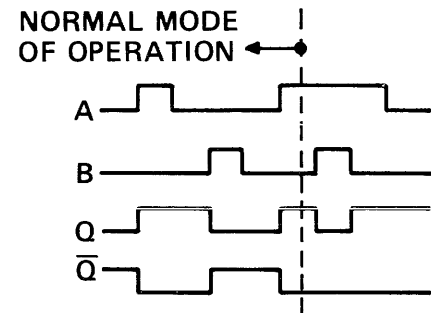
NORMALLY THE TWO INPUTS A & B ARE HIGH. WHEN THE "A" INPUT GOES LOW, THE FLIP FLOP SETS. WHEN "B" GOES LOW, THE FLIP FLOP CLEARS.

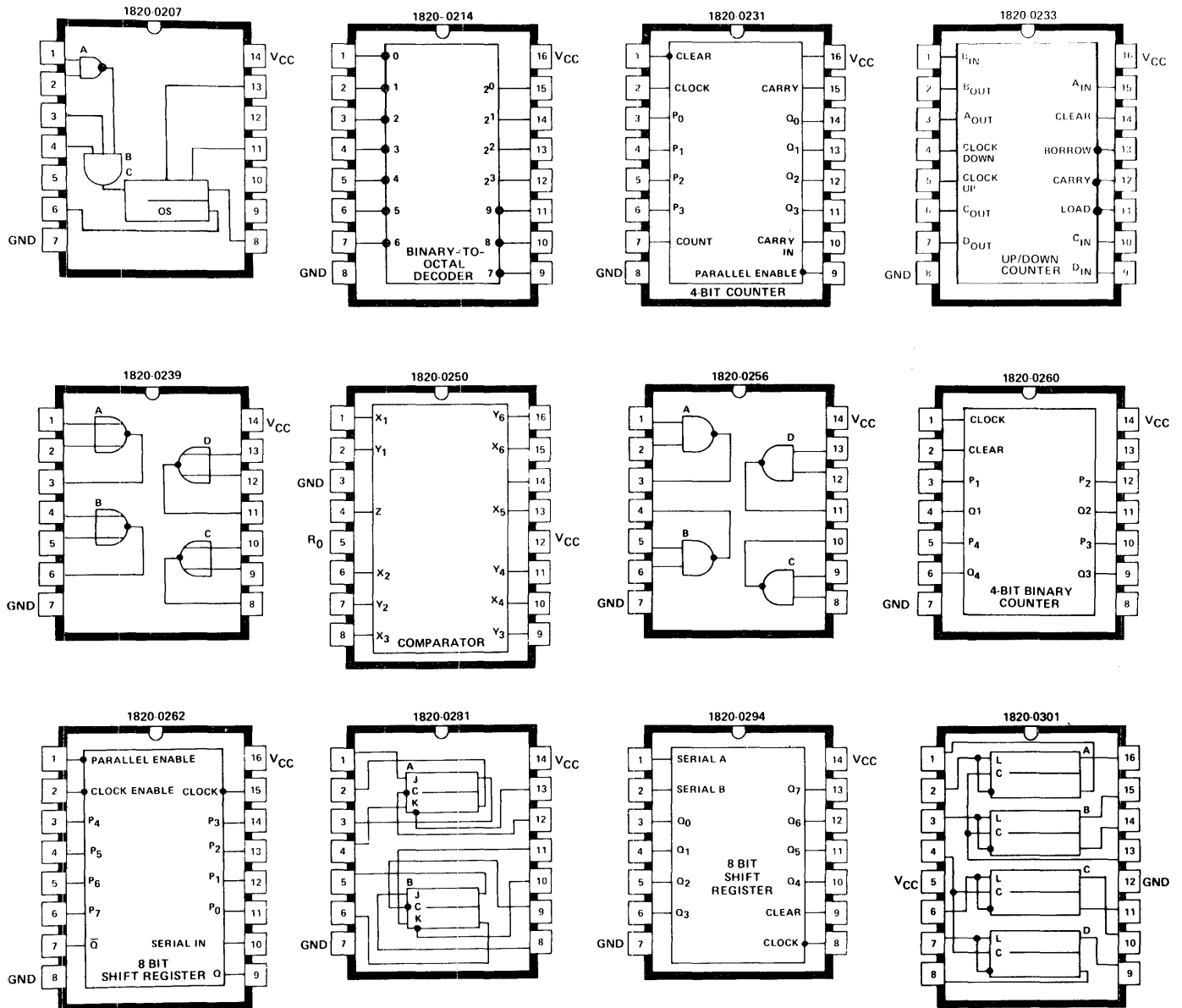


CROSS COUPLED "NOR" GATES:



NORMALLY THE TWO INPUTS "A" & "B" ARE LOW. WHEN THE "A" INPUT GOES HIGH THE FLIP FLOP WITH THE CROSSED OUTPUT (TO ALIGN THE SET AND CLEAR OUTPUT WITH THE INPUTS) SETS. WHEN "B" GOES HIGH THE FLIP FLOP CLEARS.





INTEGRATED CIRCUIT DIAGRAMS

BOOLEAN ALGEBRA IS

A SET OF LOGICAL PROPOSITIONS SIMILAR TO ORDINARY ALGEBRA WHICH IS A SET OF MATHEMATICAL PROPOSITIONS.

MATHEMATICAL SYMBOLOGY

1	=	ABSOLUTE NUMBER
0	=	ZERO
$\overline{A}$	=	LENGTH
$\rightarrow$	=	APPROACHES LIMIT
•	=	MULTIPLY
+	=	ADD
()	=	INNER PRODUCT
[]	=	AGGREGATE PRODUCT
A+B	=	ADD A AND B
AB	=	MULTIPLY A BY B

BOOLEAN SYMBOLOGY

1	=	YES (TRUE)
0	=	NO (FALSE)
$\overline{A}$	=	"NOT A"
$\rightarrow$	=	REPLACES
•	=	"AND"
+	=	"OR"
()	=	INNER "AND"
[]	=	AGGREGATE "AND"
A+B	=	A "OR" B
AB	=	A "AND" B

EIGHT BASIC IDENTITIES

THESE HOLD FOR BOTH ORDINARY AND BOOLEAN ALGEBRA.

1. $0 + A = A$
2. $0 \bullet A = 0$
3. $1 \bullet A = A$
4. $A + B = B + A$
5. $AB = BA$
6. $A + (B + C) = (A + B) + C$
7. $A (BC) = AB(C)$
8. $A (B + C) = AB + AC$

ALL THESE EXPRESSIONS MAY BE CONSTRUCTED USING
BASIC "AND-OR" GATES.

TEN IDENTITIES UNIQUE TO BOOLEAN ALGEBRA

WHICH MAKES SIMPLIFICATION OF COMPLEX CIRCUITRY
ECONOMICALLY FEASIBLE.

1. $A + A = A$

2. $A \bullet A = A$

3. $A + 1 = 1$

4. $A \bullet \bar{A} = \emptyset$

5. $A + \bar{A} = 1$

6. $\overline{A + B} = \bar{A} \bullet \bar{B}$

7. $\overline{AB} = \bar{A} + \bar{B}$

8. $A + AB = A$

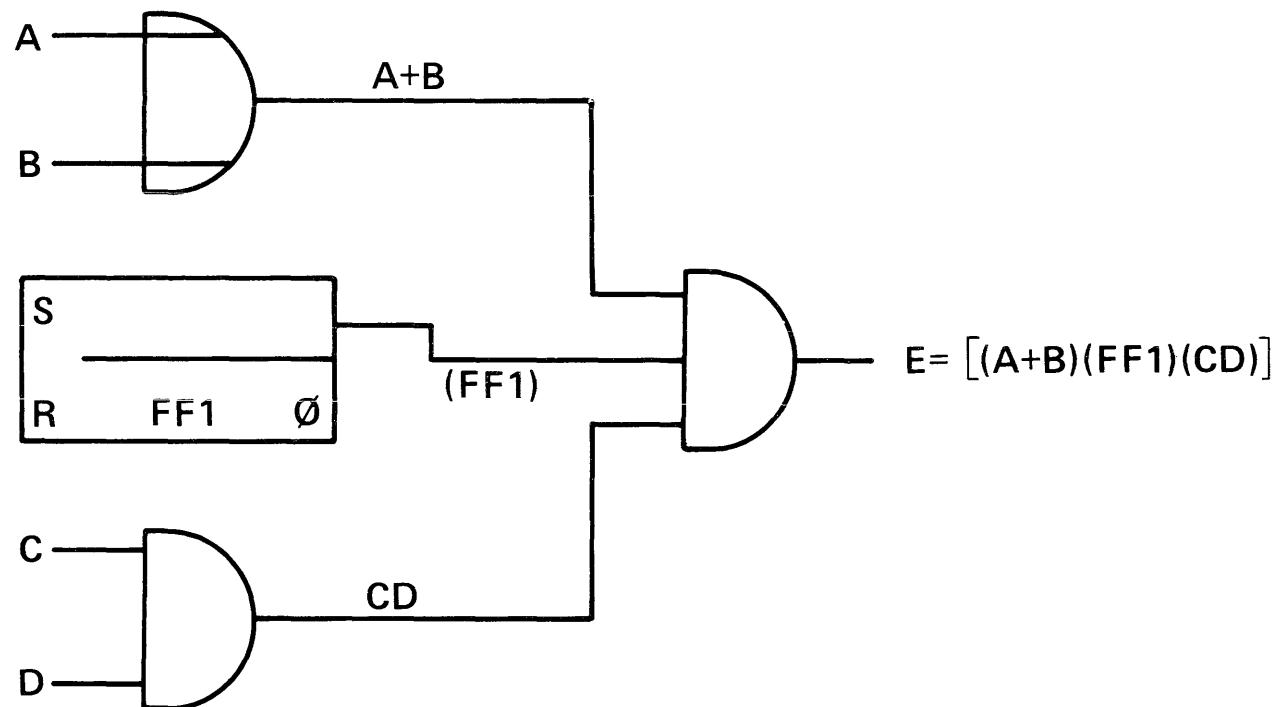
9. $A(A + B) = A$

10. $A + BC = (A + B)(A + C)$

ALL THESE EXPRESSIONS MAY BE CONSTRUCTED USING
BASIC "NOT", "AND", "OR" "NOR", AND "NAND" GATES.

THE LOGIC EQUATION

COMPLETELY DEFINES A LOGIC CIRCUIT

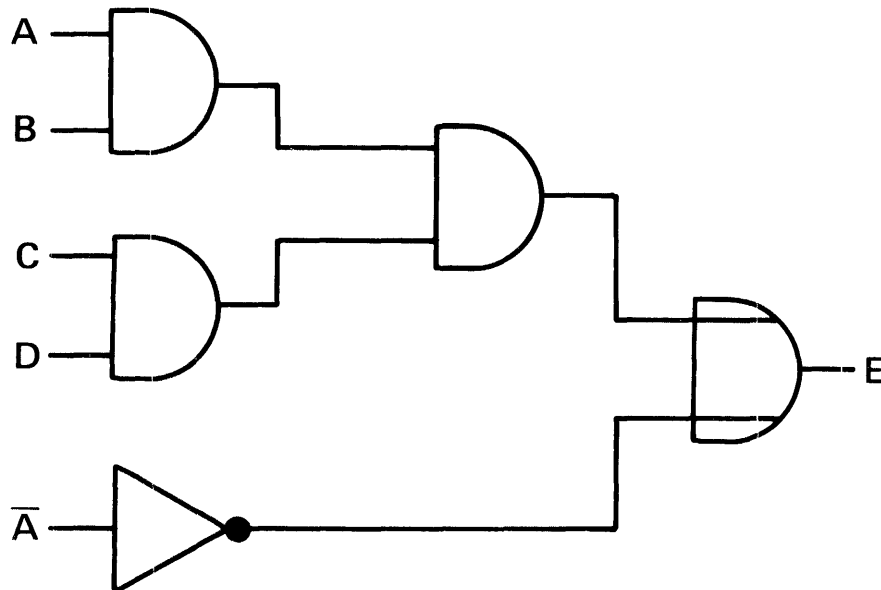


IF "E" IS NOT TRUE, THE CONTROL FUNCTIONS A,B,FF1,C
AND D MAY BE CHECKED TO ISOLATE A FAULTY COMPONENT.

PROBLEMS

1. REDUCE TO SIMPLEST FORM USING BOOLEAN IDENTITIES:
 $A (BC + AC)$

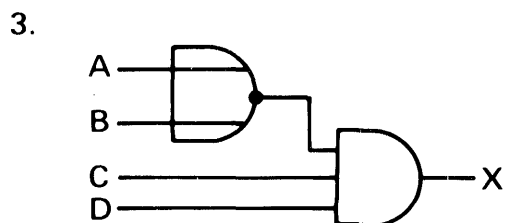
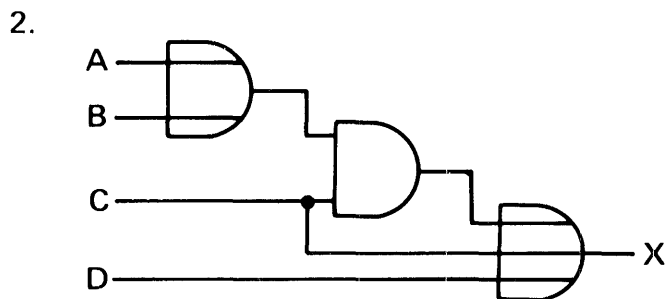
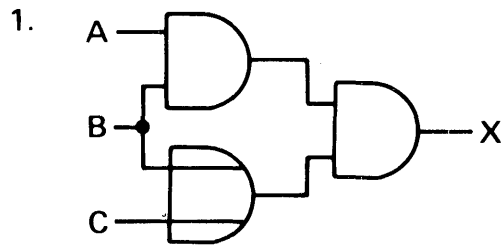
2. DETERMINE E FROM THE FOLLOWING CIRCUIT AND
REDUCE TO THE SIMPLEST FORM:



3. CONSTRUCT A SIMPLE CIRCUIT FROM THE FOLLOWING
LOGICAL EQUATION:
 $E = AB (C + D) + FF1$

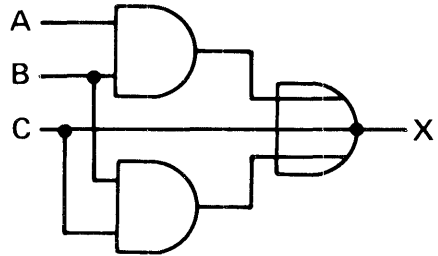
LOGIC EQUATIONS

WRITE THE LOGIC EQUATIONS FOR X, REDUCE IT TO THE SIMPLEST FORM AND DRAW AN EQUIVALENT CIRCUIT.

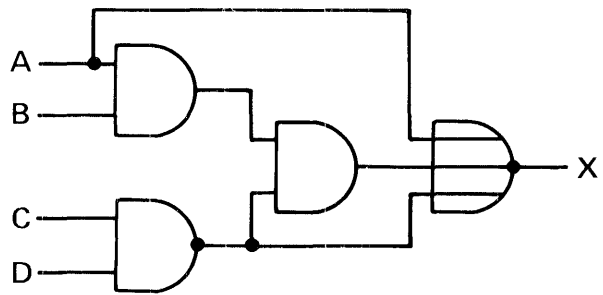


LOGIC EQUATIONS (CONT.)

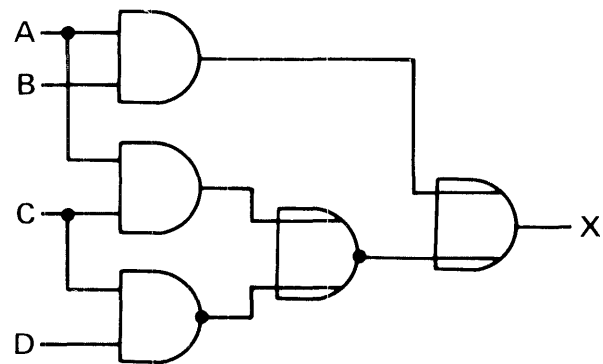
4.



5.



6.



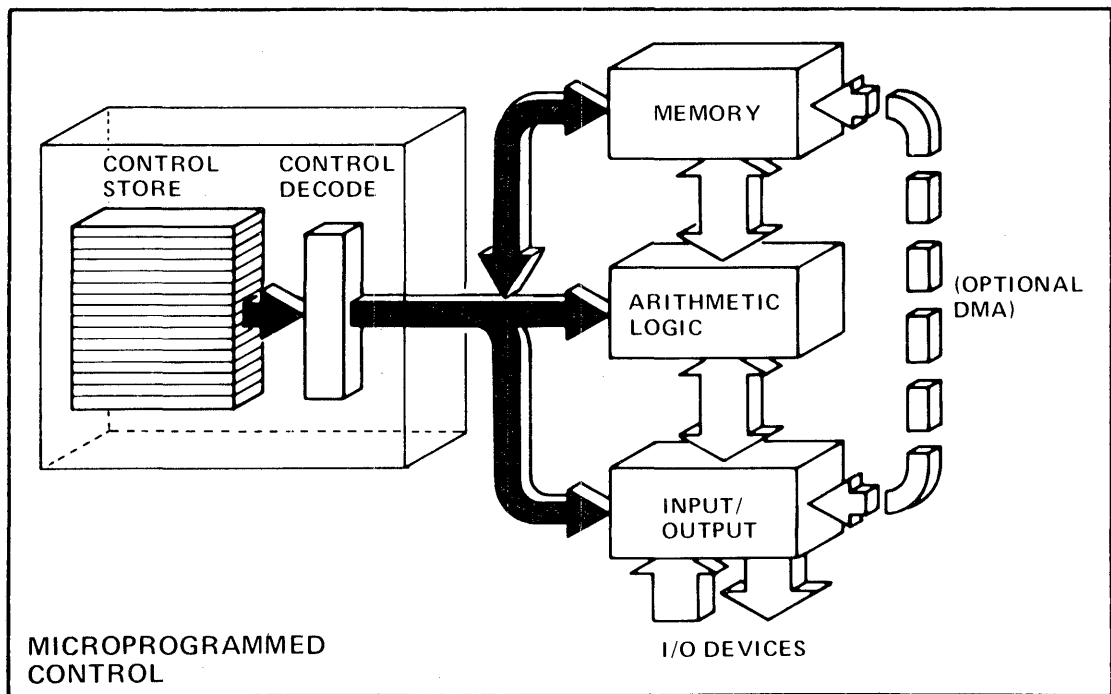
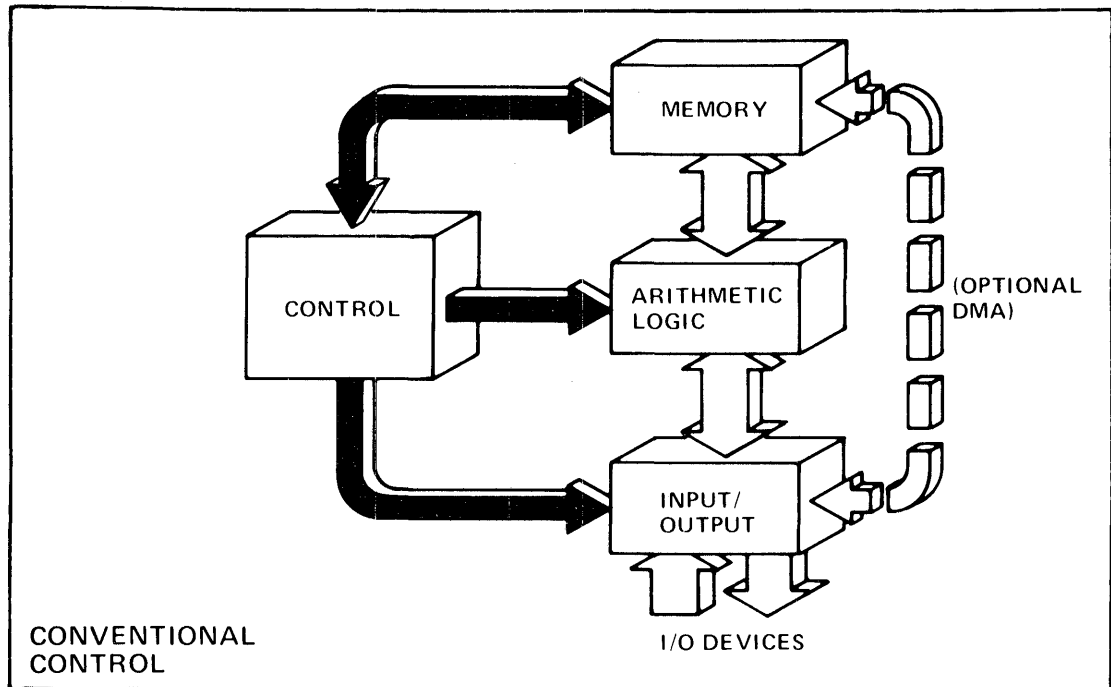
2100A architecture and block diagrams

2

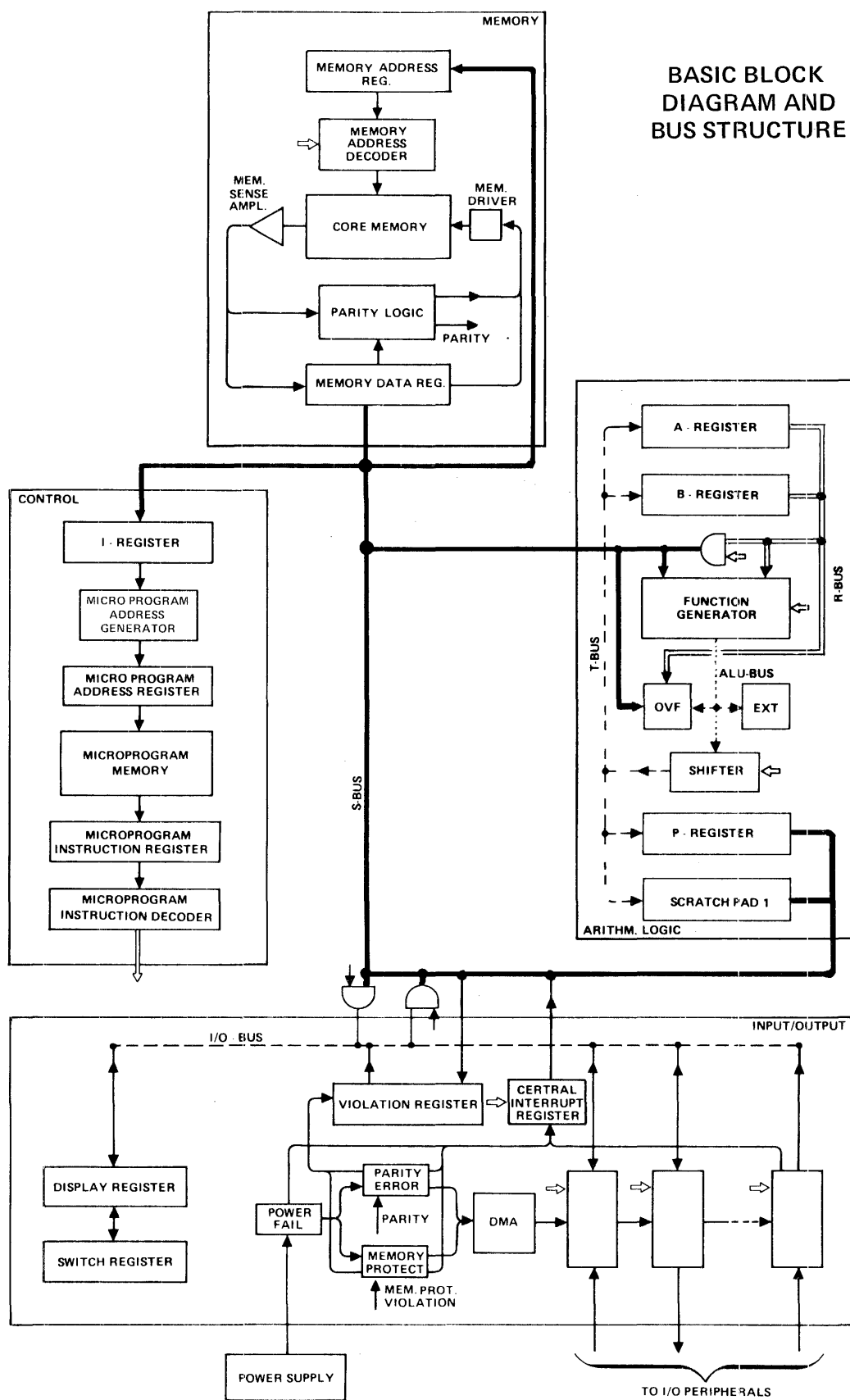
LESSON 2

2100A ARCHITECTURE AND BLOCK DIAGRAMS

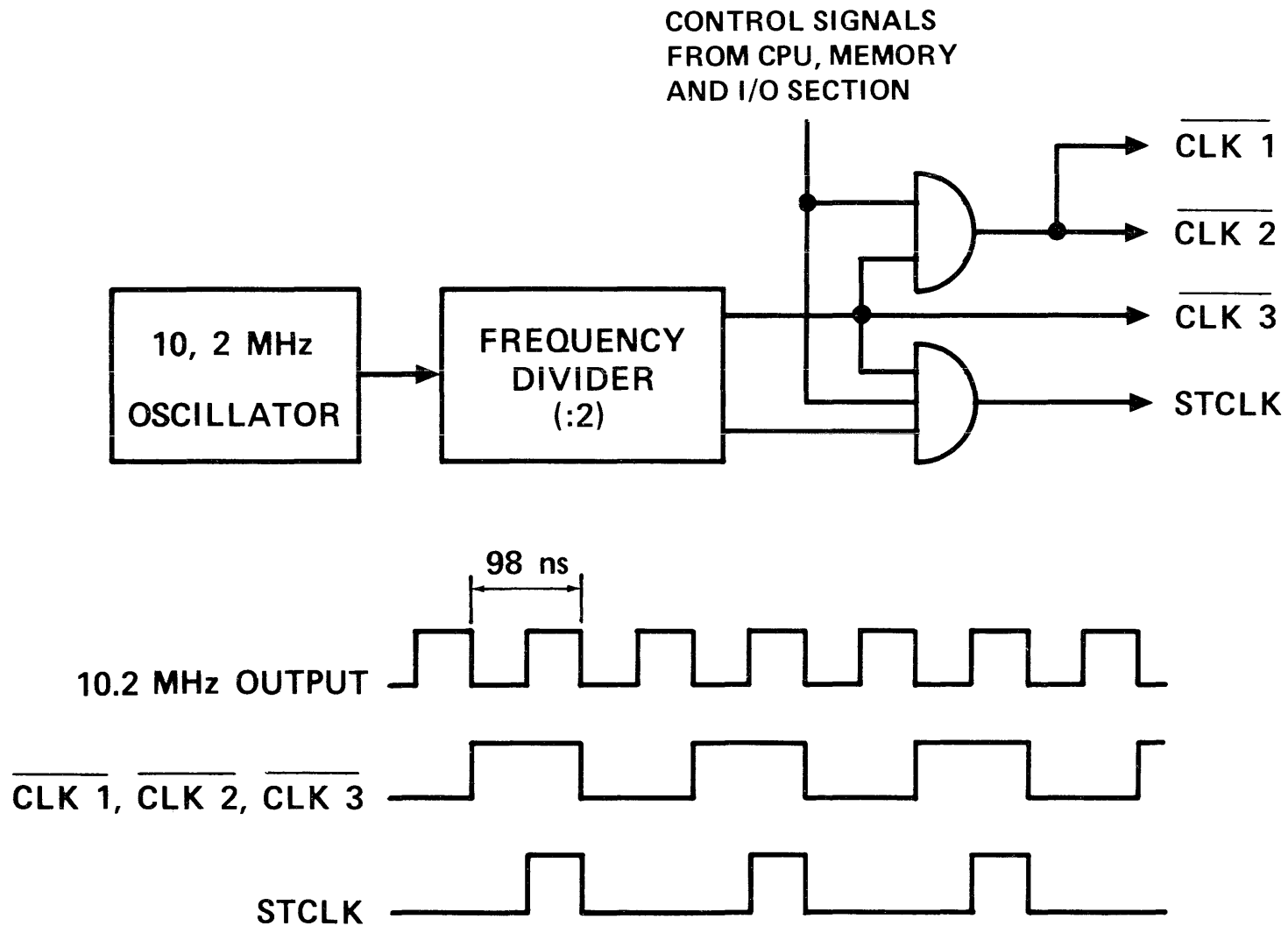
Comparison Between General Architecture of Standard and ROM Controlled Computers	2-1	Microprogram Definitions	2-14
		Microinstruction Coding	2-15
		Conditional Microinstructions	2-16
Basic Block Diagram and BUS Structure	2-2	Block Diagram of Arithmetic Logic Unit	2-17
Clock Generator	2-3	F-Register and Q-Register	2-18
Phase Control	2-4	Function Generator	2-19
Block Diagram of Control Unit	2-5	The Scratch Pads	2-20
Microprogramming	2-6	Block Diagram of Memory	2-21
ROM Address Mapper	2-7	Addressing and Read/Write Cycle in Core Memory	2-22
ROM Address Register	2-8	Parity Logic	2-23
The ROM Pack	2-9	Block Diagram of Input/Output Section	2-24
The 24 Bit ROM	2-10	Interrupt Priority System	2-25
The ROM Modules	2-11	A Typical I/O Interface Card	2-26
ROM Instruction Register	2-12	2100 Block Diagram	2-27
ROM Instruction Decoder	2-13		



COMPARISON BETWEEN GENERAL ARCHITECTURE OF STANDARD AND ROM CONTROLLED COMPUTERS

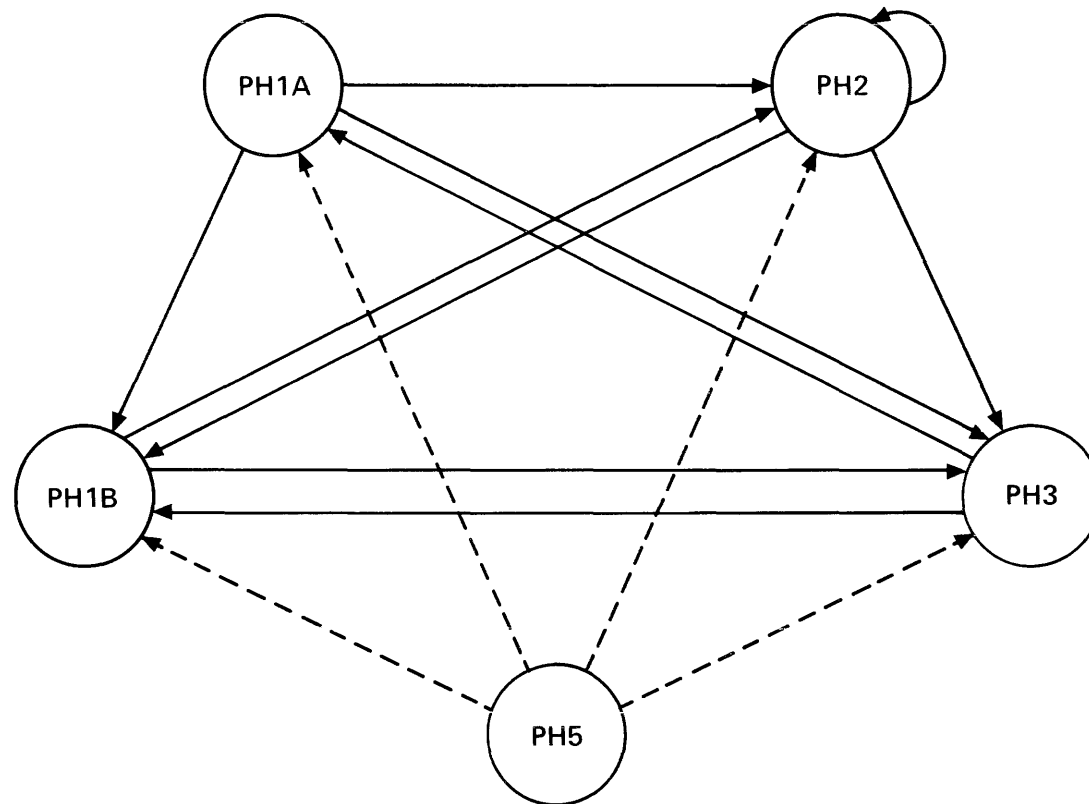


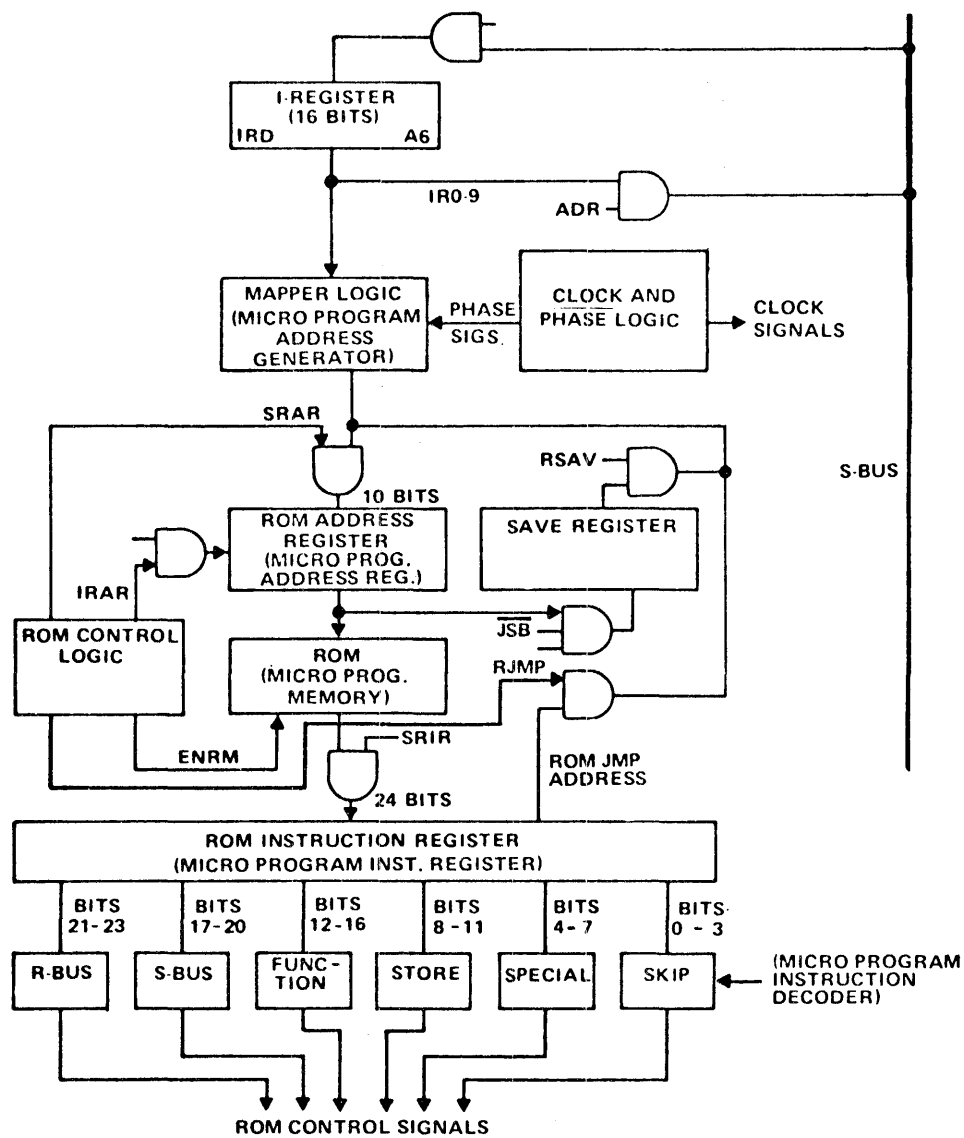
CLOCK GENERATOR



PHASE CONTROL

- AT ANY GIVEN TIME THE COMPUTER WILL BE IN ONE PHASE ONLY.
- ONE SPECIFIC MICROORDER WILL SPECIFY THAT THE NEXT MICROINSTRUCTION WILL BE THE LAST ONE OF THE PRESENT PHASE. IF THIS CONDITION IS REACHED THE PHASE LOGIC WILL ISSUE ONE OF FOUR "SET PHASE SIGNALS" WHICH IS REQUIRED TO CHANGE ANY PHASES.





BLOCK DIAGRAM OF CONTROL UNIT

MICROPROGRAMMING

A BINARY CODE IS ASSIGNED TO ALL OF THE CONTROL SIGNALS IN THE COMPUTER AND THE REQUIRED CODES ARE STORED IN ROM TO BE ACTIVATED BY THE MACHINE INSTRUCTIONS.

HORIZONTAL MICROPROGRAMMING USES A WIDE ROM WORD, AND EACH ROM BIT IS ASSIGNED A GATE.

THE 2100A USES VERTICAL MICROPROGRAMMING WHERE THE ROM WORD IS SHORT AND DECODERS ARE USED TO BREAK DOWN THE BINARY CODES TO THE GATE LEVEL.

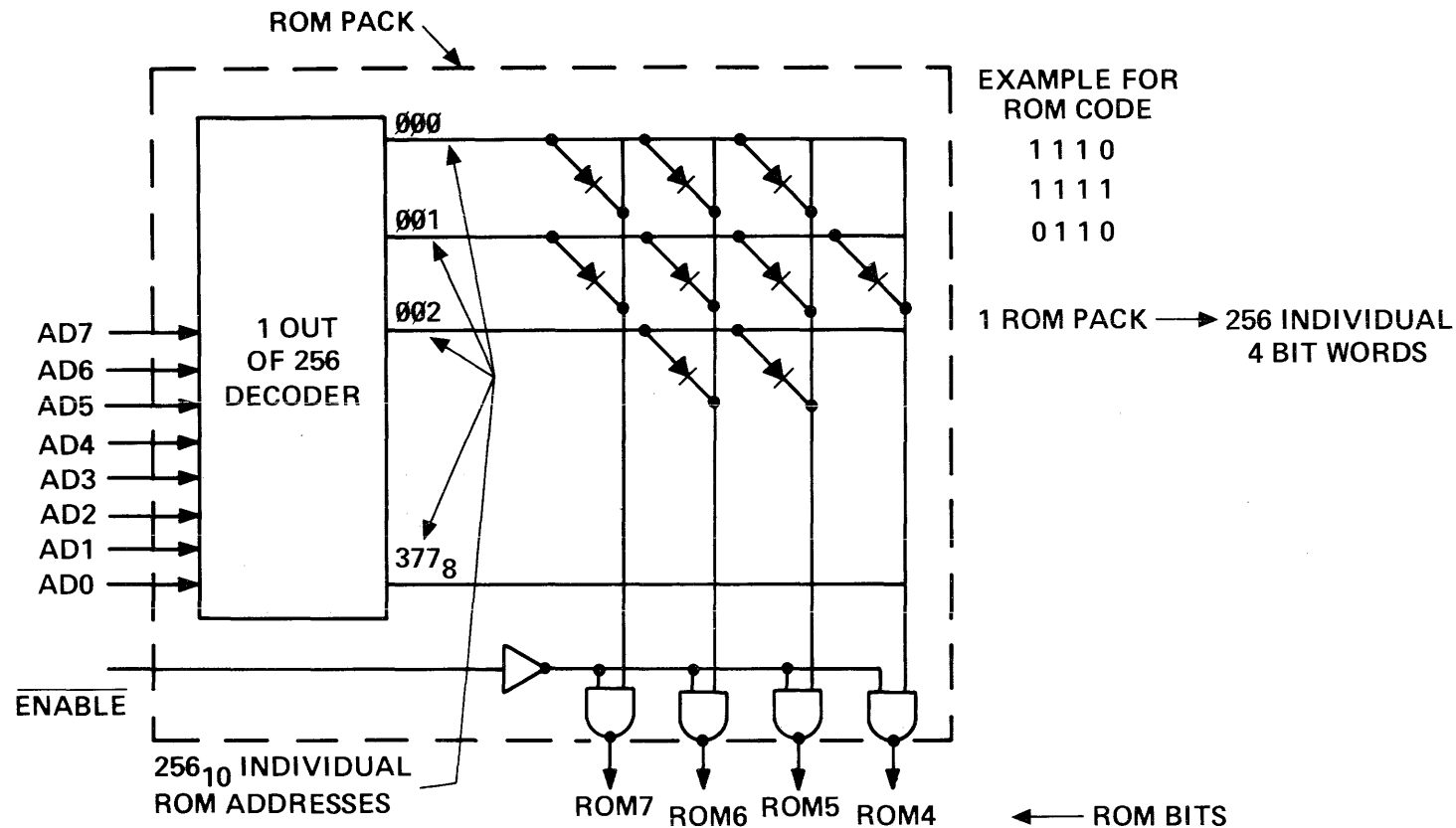
ROM ADDRESS MAPPER

THE ROM ADDRESS MAPPER DECODES THE CURRENT MACHINE INSTRUCTION AND PHASE LOGIC OUTPUT TO PROVIDE THE STARTING ADDRESS IN THE ROM MICROPROGRAM FOR THE NEXT PHASE. THIS ADDRESS IS LOADED INTO THE ROM ADDRESS REGISTER (RAR).

ROM ADDRESS REGISTER

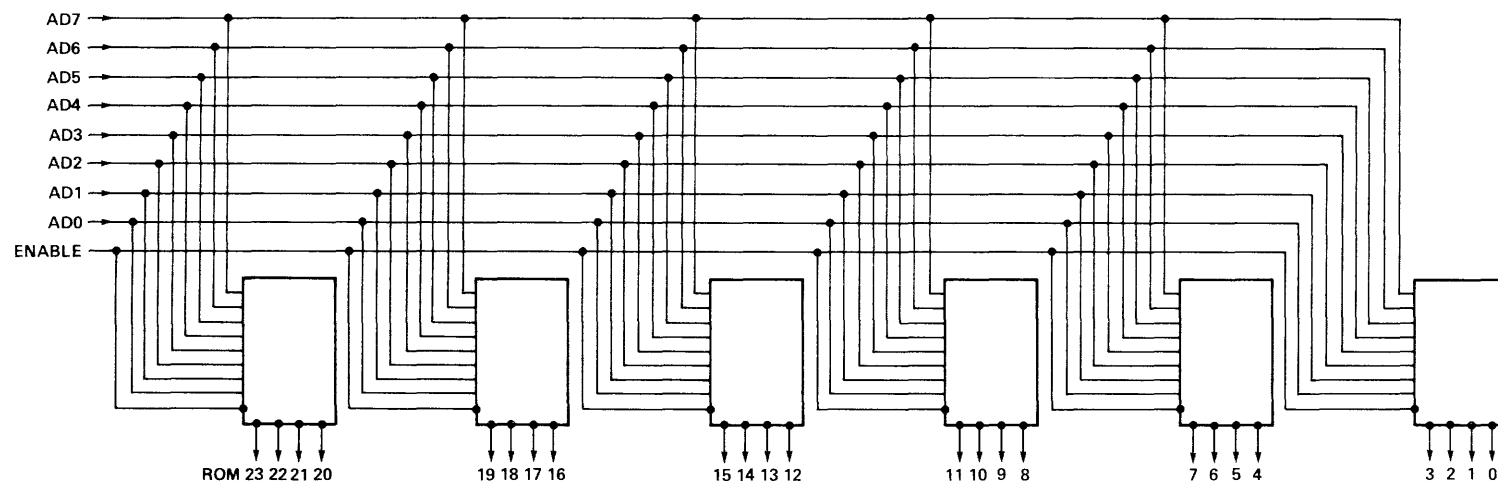
THE ROM ADDRESS REGISTER IS A 10 BIT REGISTER WITH PARALLEL INPUT. THE REGISTER CAN BE INCREMENTED. IT ALWAYS HOLDS THE ADDRESS OF THE NEXT ROM MICROINSTRUCTION TO BE EXECUTED.

THE ROM PACK



THE 8 BINARY ROM ADDRESS INPUTS ARE DECODED INTO 256₁₀ INDIVIDUAL ROM ADDRESSES (256₁₀ ROM WORDS)
 EACH ROM ADDRESS CREATES A PREDEFINED 4-BIT ROM WORD OUTPUT. THE ROM ACCESS TIME IS < 75ns.

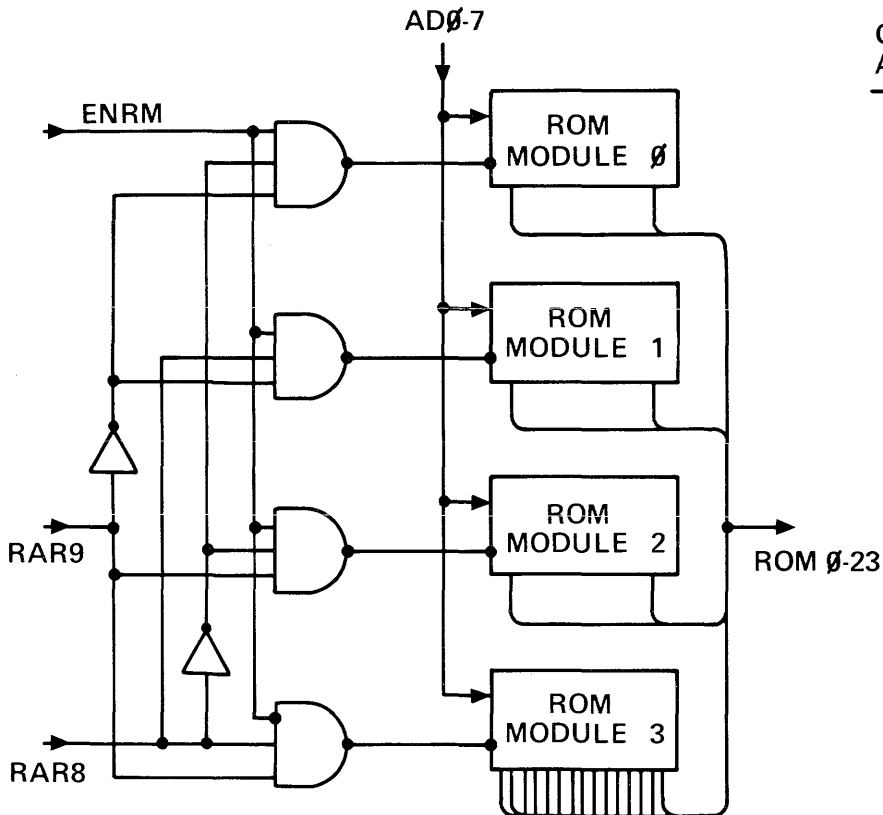
THE 24 BIT ROM



6 INDIVIDUALLY BURNED ROM PACKS WITH THE 8-BINARY ADDRESS INPUTS
CONNECTED IN PARALLEL FORM THE 24-BIT ROM WORD.

256 DIFFERENT 24-BIT ROM WORDS IS CONSIDERED TO BE ONE MODULE.

THE ROM MODULES



ABSOLUTE OCTAL ROM ADDRESSES	PURPOSE
0-377	ACCOMODATES THE STANDARD HP2100A MICROINSTRUCTION SET IN 6 ROM PACKS
400-777	ACCOMODATES USUALLY THE FLOATING POINT OPTION HP12901A IN 6 ROM PACKS
1000-1377	ACCOMODATES CUSTOM BUILT ROM, PROM OR WCS
1400-1777	ACCOMODATES CUSTOM BUILT ROM, PROM OR WCS

- ANY COMBINATION OF MODULES IS PERMISSIBLE. HOWEVER, MODULE 0 HAS TO BE PRESENT.

ROM INSTRUCTION REGISTER

- THE ROM INSTRUCTION REGISTER (RIR) PROVIDES A 24 BIT BUFFER STORAGE FOR THE ROM MICROINSTRUCTION.
- IT TRANSFERS THE MICROINSTRUCTION PARALLEL TO THE ROM INSTRUCTION DECODER.
- THE ROM INSTRUCTION REGISTER ALLOWS THE CURRENT MICRO-INSTRUCTION TO BE EXECUTED WHILE THE NEXT ONE IS ACCESSED IN ROM; THEREBY REDUCING EXECUTION TIME. THE RIR HOLDS THE CONTENTS OF THE PREVIOUSLY HELD ADDRESS IN THE RAR.

ROM INSTRUCTION DECODER

- THE ROM INSTRUCTION DECODER DECIPHERS THE CONTENTS OF THE ROM INSTR. REG.
- THE ROM INSTRUCTION DECODER IS DIVIDED INTO 6 SECTIONS.
- EACH SECTION DECODES ONE OF THE 6 FIELDS:

	R-BUS FIELD			S-BUS FIELD				FUNCTION FIELD					STORE FIELD				SPECIAL				SKIP			
ROM INSTRUCTION REGISTER BIT	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

R-BUS FIELD: SPECIFIES THE SOURCE OF THE INFORMATION TO BE READ ONTO THE R-BUS.

S-BUS FIELD: SPECIFIES THE SOURCE OF THE INFORMATION TO BE READ ONTO THE S-BUS.

FUNCTION FIELD: SPECIFIES THE ARITHMETIC- OR LOGIC FUNCTIONS, BIT SHIFTING IN REGISTERS OR SPECIAL CONTROL FUNCTIONS WHEN COMBINING R- AND S-BUS CONTENTS ONTO THE T-BUS. CONDITIONAL AND UNCONDITIONAL ROM JMPs OR -JSB CAN ALSO BE DECODED.

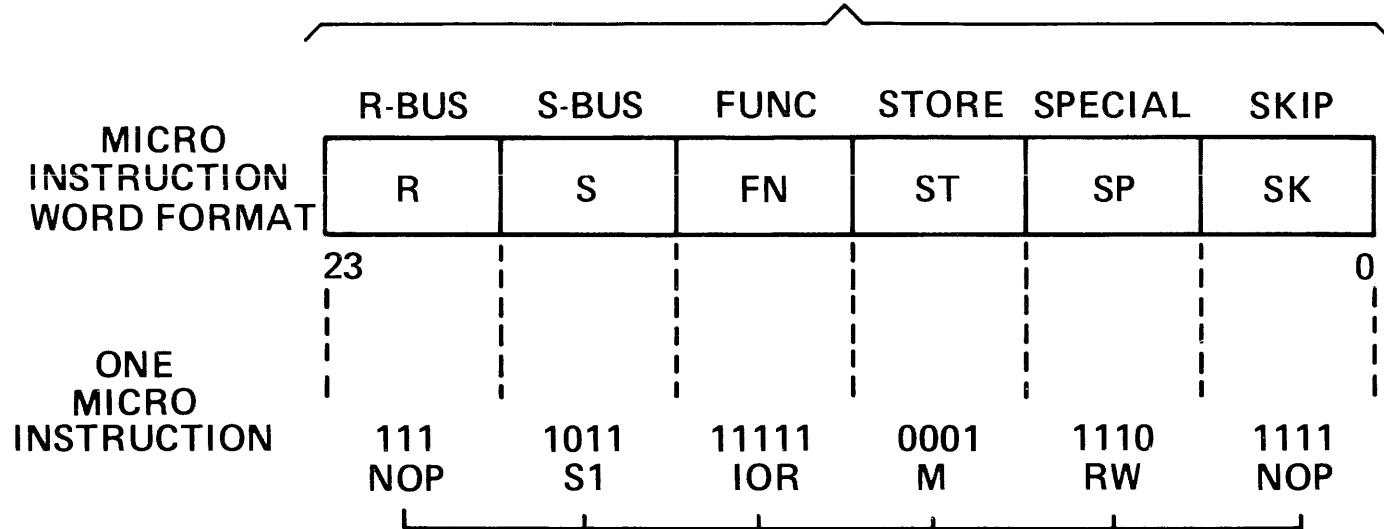
STORE FIELD: SPECIFIES THE DESTINATION OF THE INFORMATION ON THE T- OR S-BUS.

SPECIAL FIELD: ENABLES ALTER SKIP, SHIFT-ROTATE AND INPUT/OUTPUT GROUP INSTRUCTIONS, INITIATES MEMORY READ/WRITE OR CLEAR/WRITE OPERATIONS AND SPECIFIES SOME SPECIAL FUNCTIONS.

SKIP FIELD: DECODES CONDITIONAL AND UNCONDITIONAL ROM SKIPs.

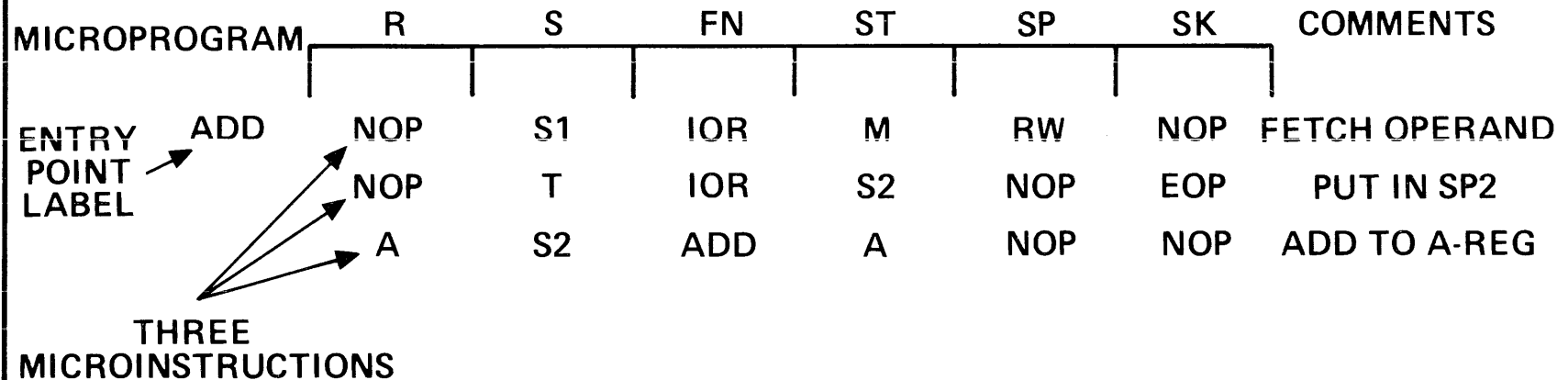
MICROPROGRAM DEFINITIONS

SIX FIELDS



SIX MICRO-ORDERS

A MICROINSTRUCTION (24 BITS LONG) CONSISTS OF 6 MICRO-ORDERS
 A MICROPROGRAM CONSISTS OF SEVERAL MICROINSTRUCTIONS.
 EACH MICRO-PROGRAM HAS AN ENTRY POINT LABEL.



MICROINSTRUCTION CODING

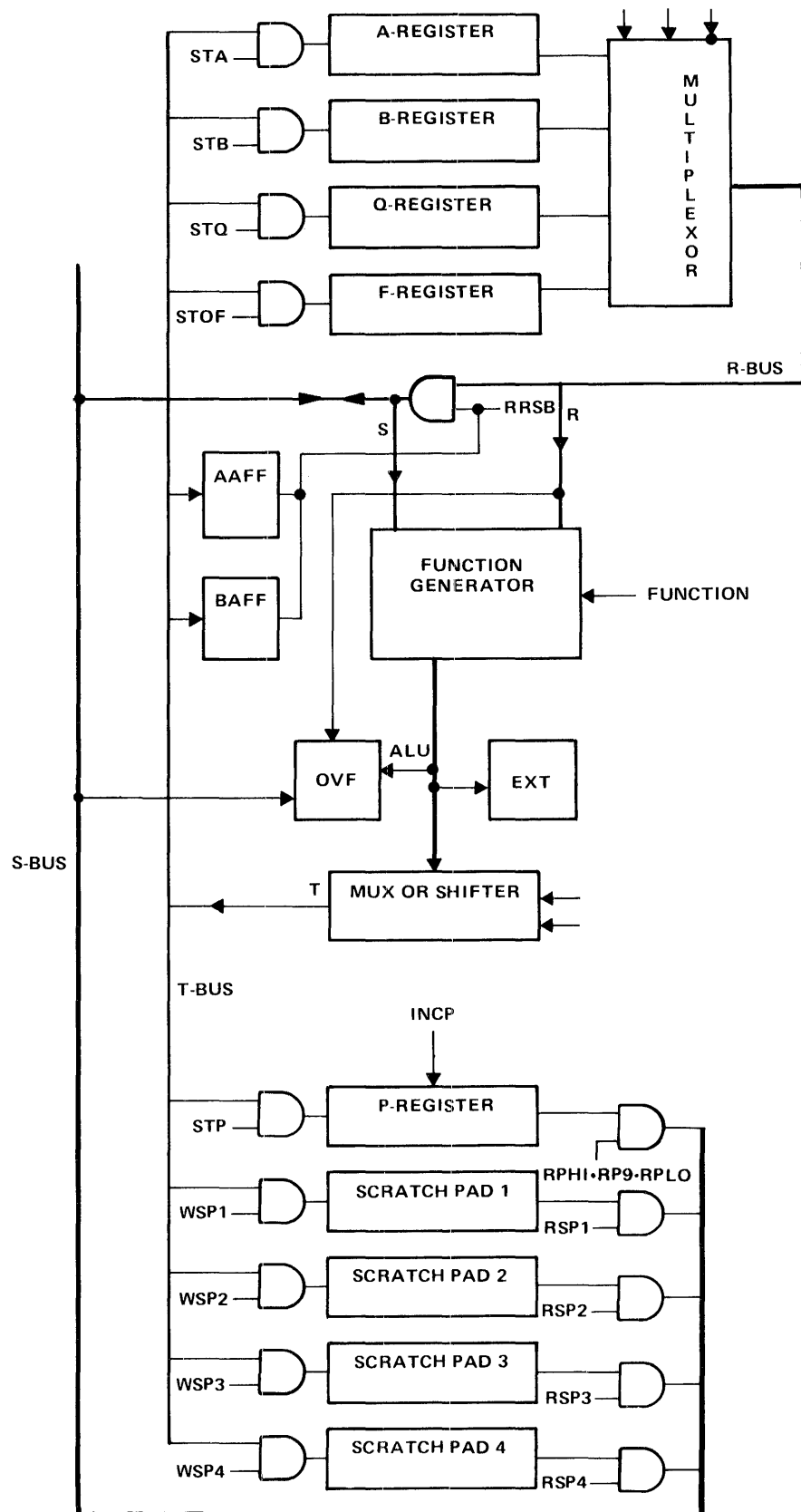
CODE					R-Bus Field	S-Bus Field	Function Field	Store Field	Special Field	Skip Field
5-Bit Field										
4-Bit Field										
3-Bit Field					3 Bits	4 Bits	5 Bits	4 Bits	4 Bits	4 Bits
1	1	1	1	1	NOP	NOP	IOR	NOP	NOP	NOP
1	1	1	1	0	CO	P	SOV	A	RW	UNC
1	1	1	0	1	AAB	CL	CLO	B	IOG1	EOP
1	1	1	0	0	CAB	CR	SFLG	AAB	CW	NAAB
1	1	0	1	1	F	S1	CFLG	CAB	ASG2	AAB
1	1	0	1	0	Q	S2	LWF	Q	ASG1	NMPV
1	1	0	0	1	B	S3	***	F	ECYN	CTR
1	1	0	0	0	A	S4	ARS	P	ECYZ	CTRI
1	0	1	1	1		COND	CRS	S1	LEP	TBZ
1	0	1	1	0		ADR	LGS	S2	AAB	FLG
1	0	1	0	1		CNTR	RSB	S3	SRG2	OVF
1	0	1	0	0		RRS	CJMP	S4	SRG1	COUT
1	0	0	1	1		M	JMP	IR	CNTR	NEG
1	0	0	1	0		T	JMP	T	R1	ODD
1	0	0	0	1		IOI	JSB	M	L1	RPT
1	0	0	0	0		CIR	JSB	IOO	RSS	ICTR
0	1	1	1	1			**			
0	1	1	1	0			XOR			
0	1	1	0	1			NOR			
0	1	1	0	0			AND			
0	1	0	1	1			ADD			
0	1	0	1	0			ADDO			
0	1	0	0	1			INC			
0	1	0	0	0			INCO			
0	0	1	1	1			**			
0	0	1	1	0			DEC			
0	0	1	0	1			SUB			
0	0	1	0	0			DIV			
0	0	0	1	1			MPY			
0	0	0	1	0			PIA			
** Undefined codes										
*** CJMP in later machines (A4 part No. 02100-60022)										

CONDITIONAL MICROINSTRUCTIONS

THE EXECUTION OF SOME MICROINSTRUCTIONS IS DEPENDENT ON CERTAIN STATUS SIGNALS FROM MEMORY OR I/O. THE ASYNCHRONOUSLY RUNNING CPU HAS TO WAIT FOR MEMORY- OR I/O- SIGNALS TO BE SUPPLIED AT A PREDEFINED TIME.

IF THE CONDITIONS ARE NOT SATISFIED, THE CPU FREEZES (SUSPENDS EXECUTION OF THE MICROPROGRAM BY ELIMINATING CERTAIN TIMING SIGNALS) UNTIL THE CONDITION IS SATISFIED.

<u>MICROORDER</u>	<u>FIELD</u>	<u>CONDITION</u>
RW / CW	SPECIAL	T6
T / COND	S-BUS	DTRY
M	STORE	<u>MBSY</u>
I/O1	SPECIAL	T2



BLOCK DIAGRAM OF ARITHMETIC LOGIC UNIT

F-REGISTER AND Q-REGISTER

- THE F— AND Q— REGISTER ARE TWO ADDITIONAL 16-BIT REGISTERS.
- BOTH REGISTERS ARE FED FROM THE T-BUS AND OUTPUT VIA THE MULTIPLEXER TO THE R-BUS.
- THE Q-REGISTER (QUOTIENT-REGISTER) IS NOT ACCESSIBLE THROUGH SOFTWARE BY THE PROGRAMMER.
- THE F-REGISTER (FENCE-REGISTER) CAN ONLY BE LOADED WITH AN OTA 05 OR OTB 05 SOFTWARE INSTRUCTION.
- BOTH REGISTERS ARE USED BY FIRMWARE (MICROPROGRAMMING) FOR THE EXECUTION OF THE EAG-INSTRUCTION:

ASL

LSL

RRL

DIV

FUNCTION GENERATOR

- THE FUNCTION GENERATOR COMBINES THE CONTENTS ON THE R-BUS WITH THE ONE ON THE S-BUS WITH A SPECIFIED ARITHMETIC OR LOGIC FUNCTION. THE RESULT APPEARS ON THE ALU-BUS.

LOGIC FUNCTIONS	ARITHMETIC FUNCTIONS
ALU = R+S (IOR)	ALU = R+S (ADD)
ALU = $\overline{R}$ (NOT)	ALU = R+S+1 (INCREMENT)
ALU = R $\oplus$ S (EOR)	ALU = R (TRANSFER)
ALU = $\overline{R+S}$ (NOR)	ALU = R-S-1 (DECREMENT)
ALU = R·S (AND)	ALU = R-S (SUBTRACT)

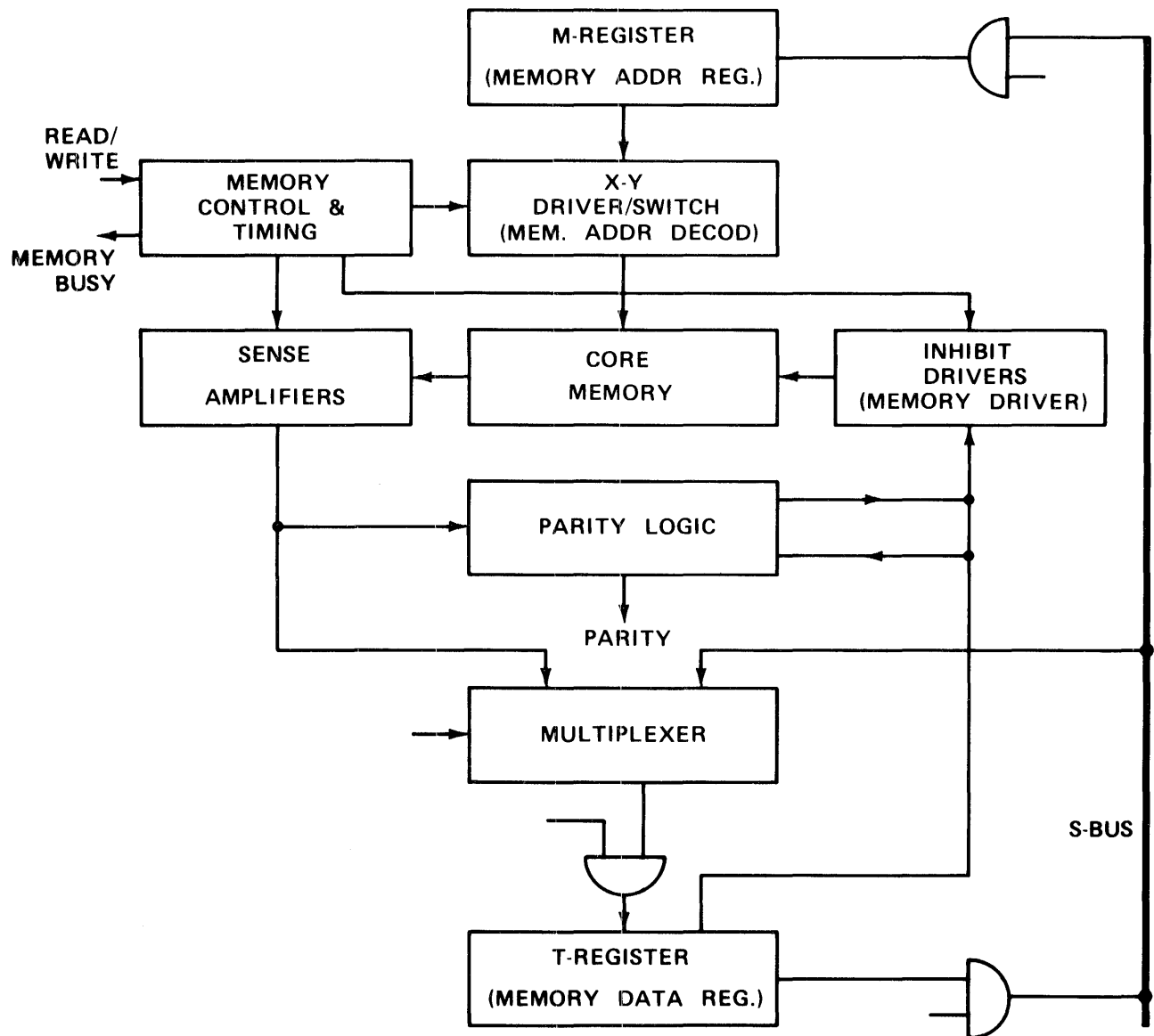
- THE ARITHMETIC OR LOGIC FUNCTION IS DETERMINED BY THE SOFTWARE INSTRUCTION AND DECODED BY THE MICRO-ORDER IN THE FUNCTION FIELD.

THE SCRATCH PADS

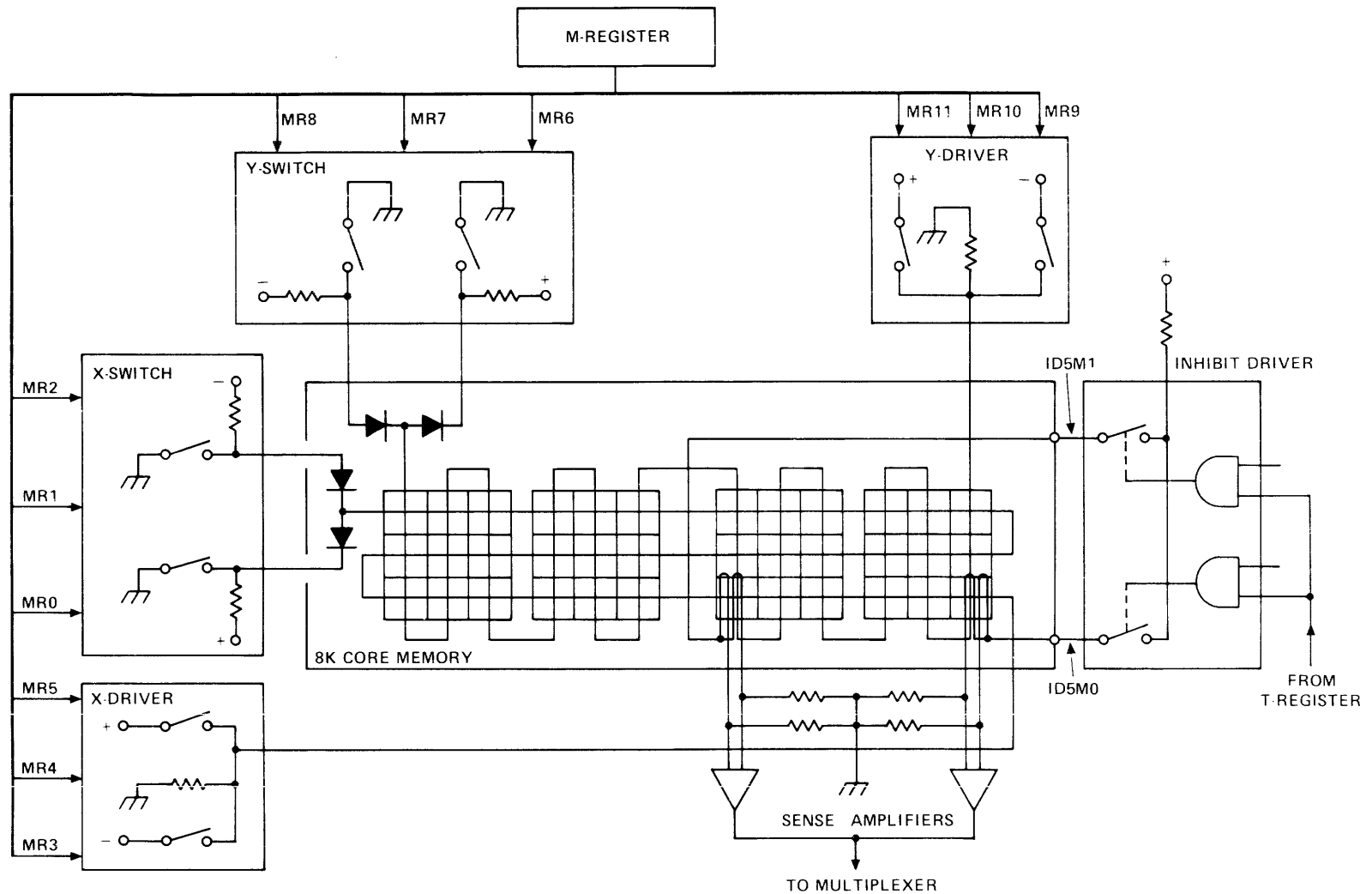
- FOUR 16 BIT REGISTERS FOR TEMPORARY STORAGE OF ADDRESSES AND DATA ARE DESIGNATED SCRATCH PAD 1-4.
- ALL FOUR SCRATCH PAD REGISTERS ARE LOADED FROM THE T-BUS AND OUTPUT TO THE S-BUS.
- THE REGISTERS ARE EXCLUSIVELY ACCESSIBLE BY MICROPROGRAMMING.
- THE GENERAL PURPOSE OF THE SCRATCH PADS IS AS FOLLOW:

SCP1	SCP2	SCP3	SCP4
HOLDS OPERAND ADDRESS OF MRG-INSTRUCTIONS & IS USED DURING DIV-INSTRUC.	HOLDS OPERAND DATA OF MRG-INSTRUCTIONS & IS USED DURING DIV-INSTRUC.	HOLDS F-REG. CONTENTS FOR ASL-, LSL- & RRL-INSTRUCTIONS & IS USED DURING DIV-INSTRUC.	HOLDS MULTIPLICAND FOR MPY-INSTRUCTION & F-REG CONTENT DURING DIV-INSTRUCTION

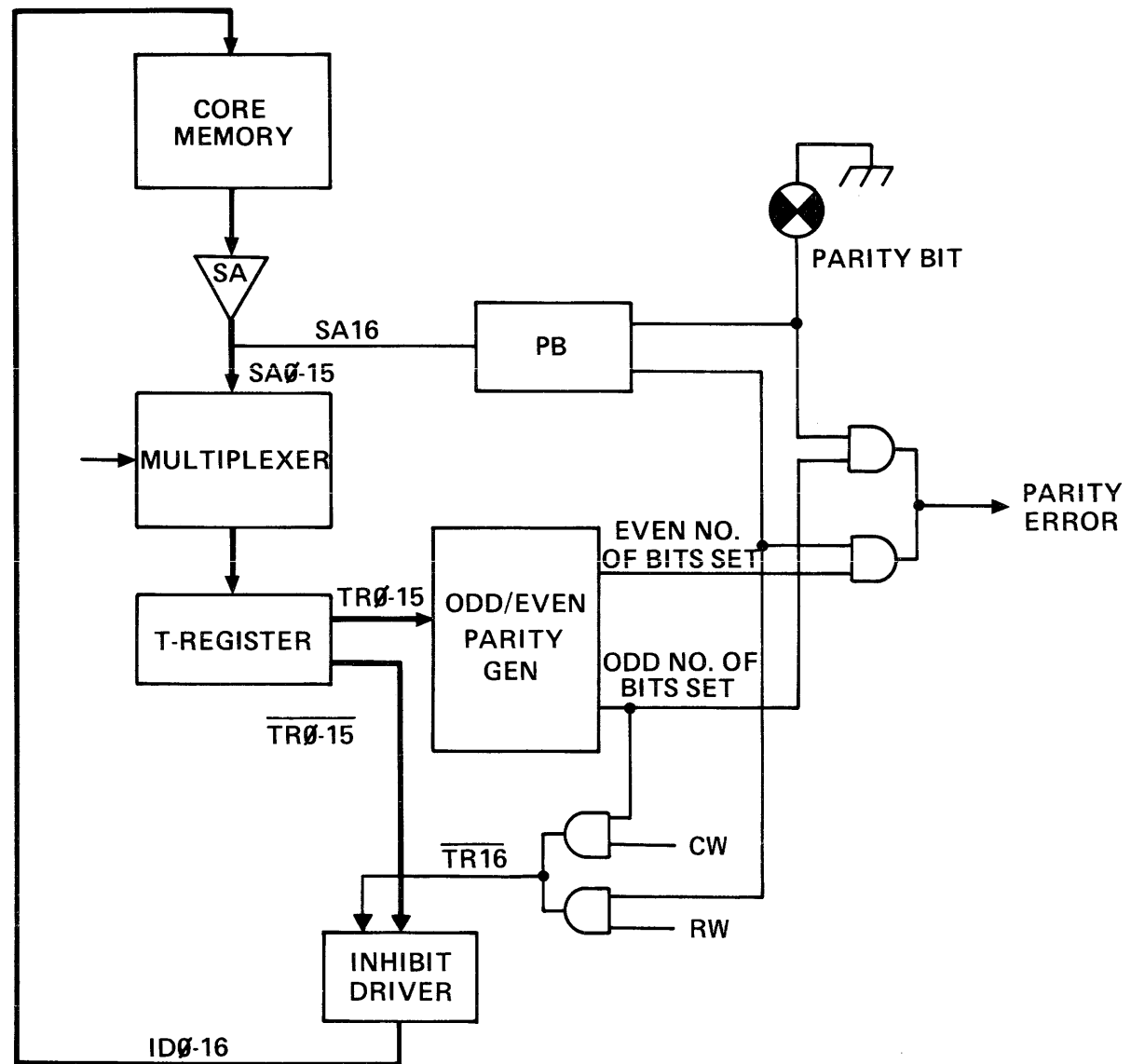
BLOCK DIAGRAM OF MEMORY

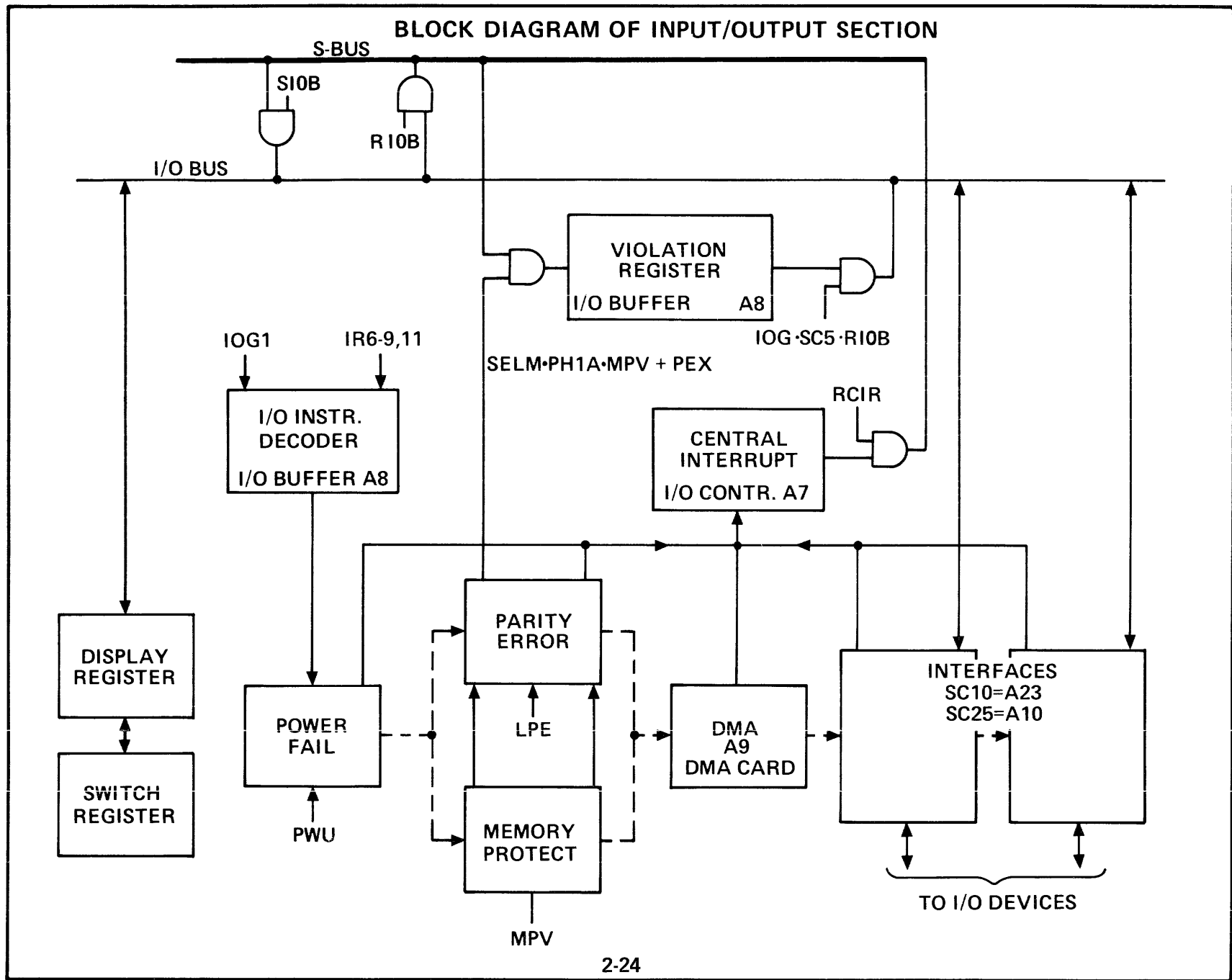


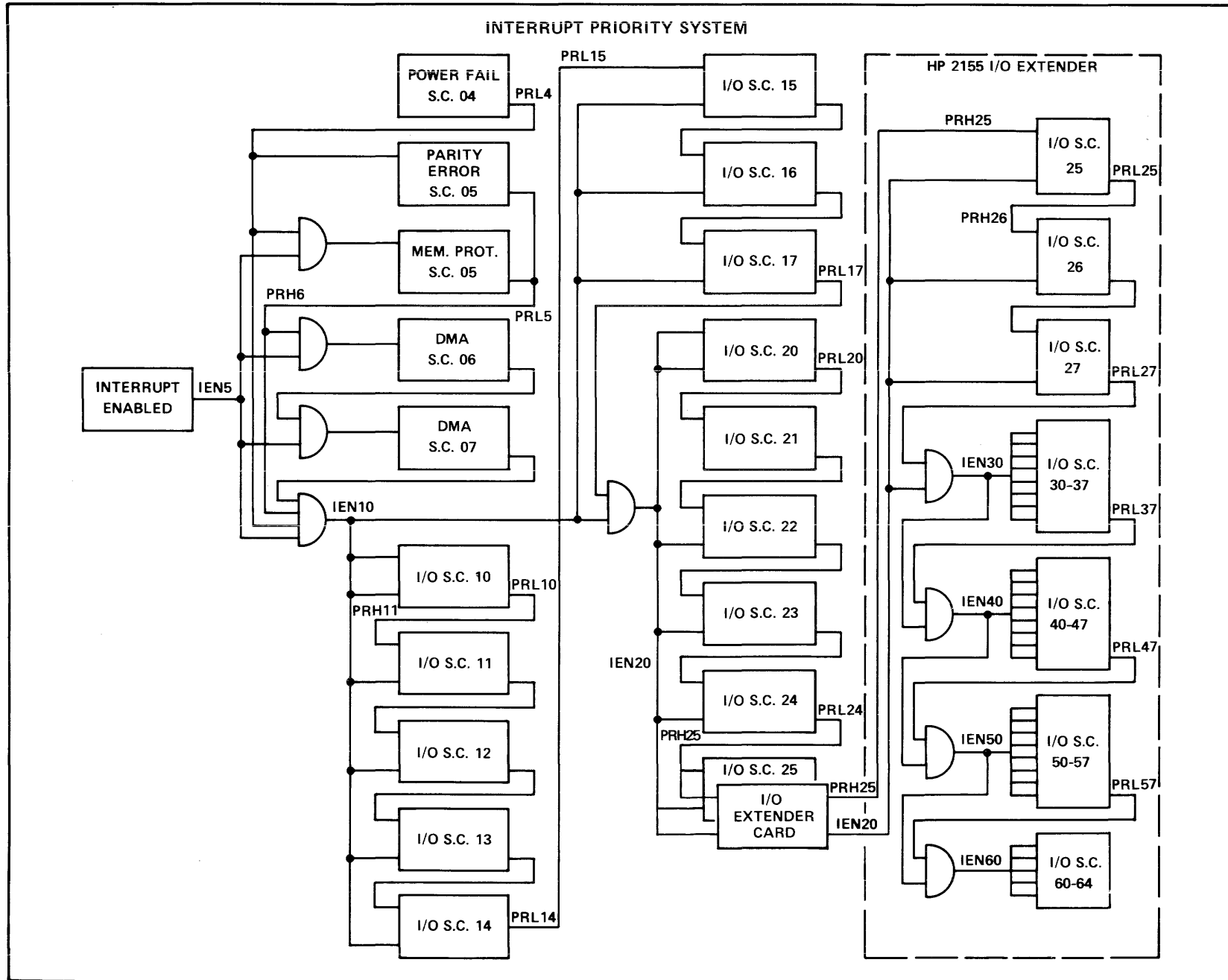
ADDRESSING AND READ/WRITE CYCLE IN CORE MEMORY



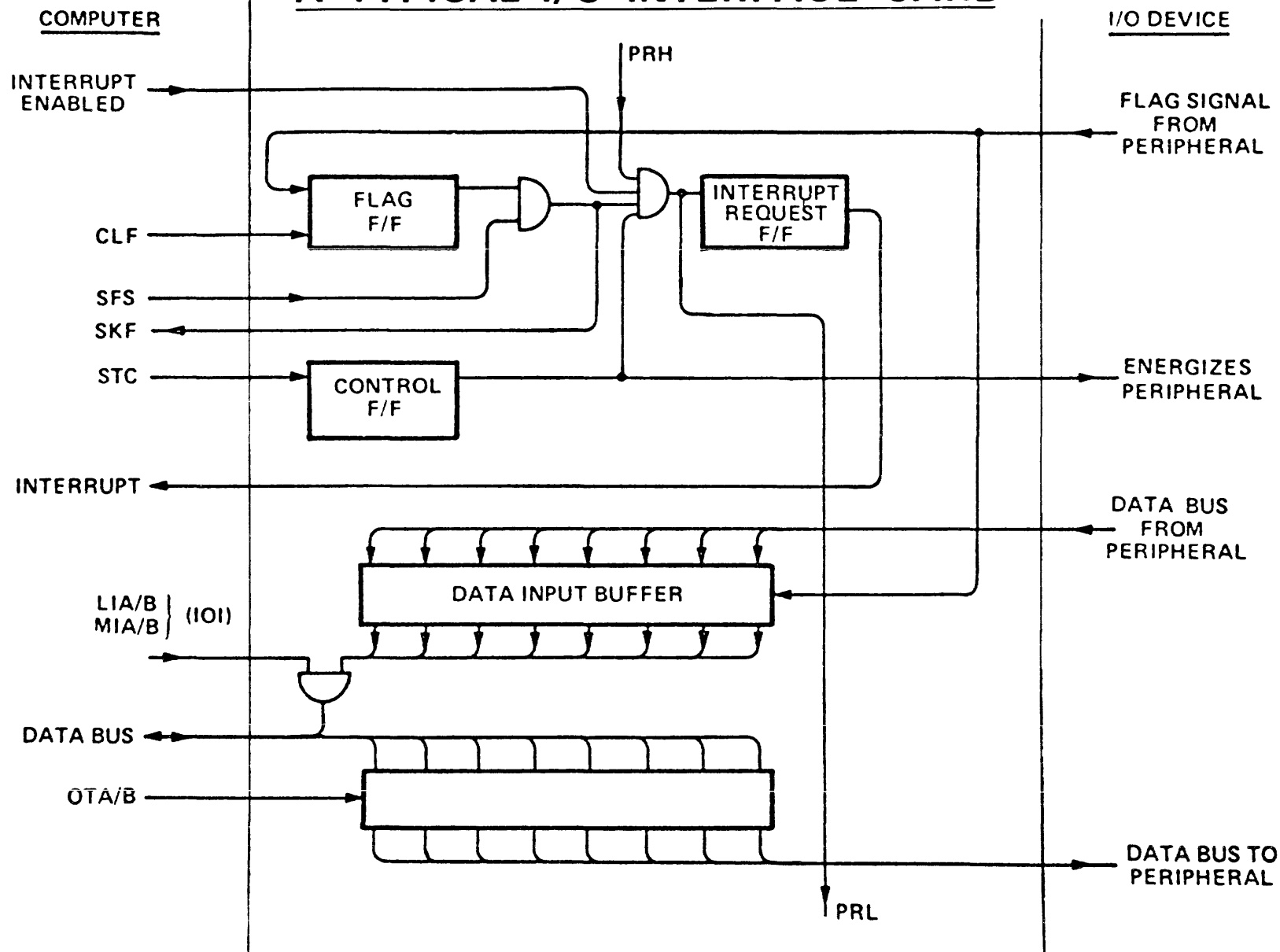
PARITY LOGIC



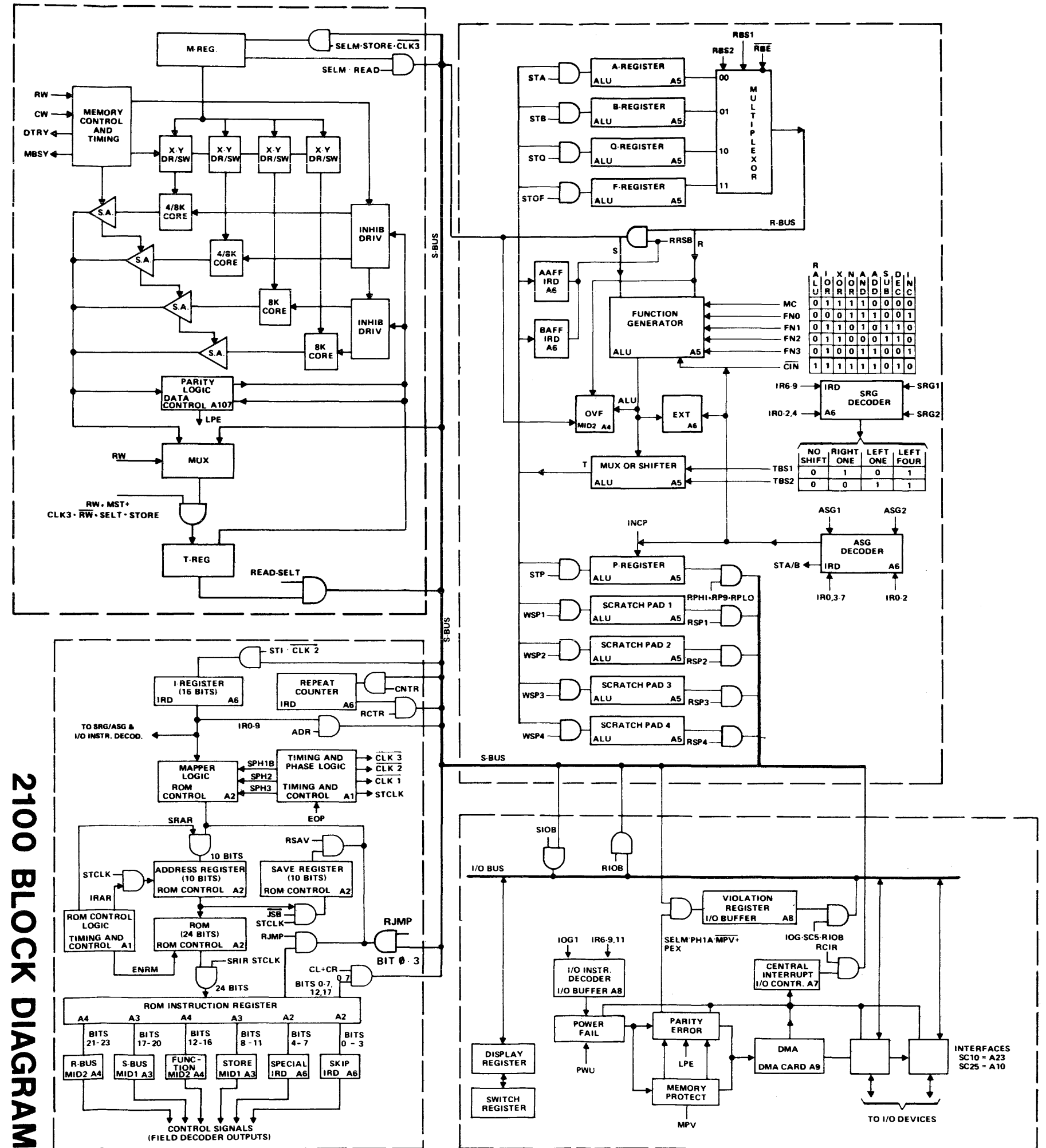




A TYPICAL I/O INTERFACE CARD



2100 BLOCK DIAGRAM



microprogramming and central processor unit

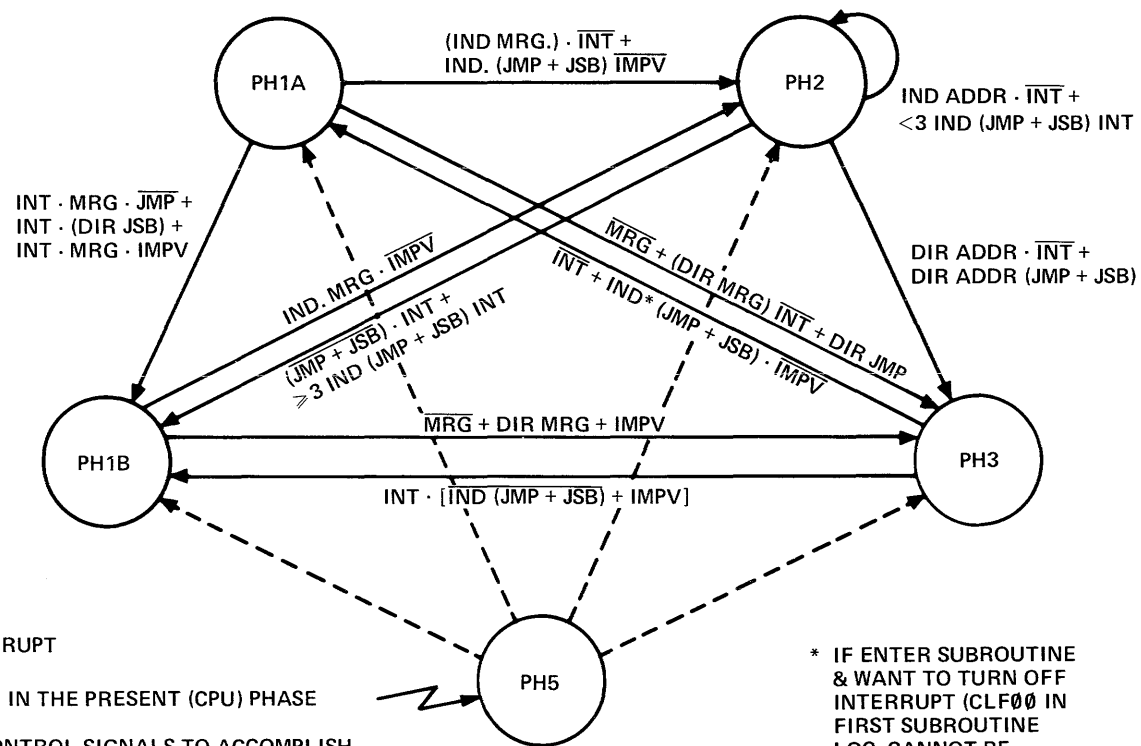
3

LESSON 3

MICROPROGRAMMING AND CENTRAL PROCESSOR UNIT

Phase Control Logic	3-1	Timing Diagram of PH3, CP—Instruction (Operand Address is in A/B-Register)	3-38
Boolean Equations for Phase Control	3-2	ISZ Instruction	3-39
Mapping the ROM Address	3-3	Flow Chart of ISZ—Instruction	3-40
Boolean Equations for Mapper Output	3-4	Flow Chart of ISZ—Instruction (Cont'd.)	3-41
Fetch Phase	3-5	Flow Chart of ISZ—Instruction (Cont'd.)	3-42
PH1A Flow Chart	3-6	Timing Diagram of PH3, ISZ—Instruction	3-43
CPU Freeze	3-7	Flow Chart of SRG—Instruction	3-44
Timing Diagram of PH1A (Instruction Does Not Reside in A/B-Register)	3-8	Hardware Signals to Implement SRG— Instructions	3-45
Simplified Diagram of A/B Addressable Logic	3-9	ASG—Instructions	3-46
A/B ADDR. FF, A/B CLRFF and A/B SEL FF. Logic in Halt/Single Cycle Mode	3-10	Function of ASG1 and ASG2	3-47
ROM Microinstruction Function Field Decoding and Equations Created By Arithmetic Logic Function Generator	3-11	Decoding ASG Instructions	3-48
Timing Diagram of PH1A (Instruction Resides in A/B-Register)	3-12	Execution Priority of Concatenated Alter-Skip Group Instructions	3-49
Flow Chart of IOR—Instruction	3-13	Timing Diagram of PH3; CLA, SLA, INA— Instruction	3-50
Timing Diagram of PH3 IOR Instruction (Operand is in A/B-Register)	3-14	STC, CLC, STF, CLF, SFS, SFC and HLT— Instructions	3-51
Timing Diagram of PH3 IOR Instruction (Operand is not in A/B-Register)	3-15	Timing Diagram for I/O—Instructions	3-52
Timing Diagram of PH3 IOR Instruction (Instruction Resides in A/B-Reg. and Operand is not in A/B-Register)	3-16	MI/LI—Instructions	3-53
Flow Chart of AND/XOR—Instruction	3-17	OT—Instruction	3-54
Flow Chart of LD—Instruction	3-18	Interrupt Phase	3-55
Indirect Phase	3-19	The Binary Multiplication	3-56
Flow Chart of PH2	3-20	How Can An Integer Multiplication Be Implemented By Hard/Firmware?	3-57
Flow Chart of ADD—Instruction	3-21	How Can An Integer Multiplication Be Implemented By Hard/Firmware? (Cont'd.)	3-58
STA/B—Instruction	3-22	Why Does the 2. Algorithm Execute A Binary Multiplication Correctly?	3-59
Flow Chart of ST—Instruction	3-23	MPY—Instruction	3-60
ROM Skips	3-24	MPY—Instruction (Cont'd.)	3-61
Flow Chart of JMP—Instruction	3-25	MPY—Microprogram	3-62
JSB—Instruction	3-26	Special Microorders Required For the Execution of EAG—Memory Ref. Instructions	3-63
Flow Chart of JSB—Instruction	3-27	EAG—Memory Ref. Instructions (Cont'd.)	3-64
Flow Chart of JSB—Instruction (Cont'd.)	3-28	ROM JSB	3-65
Timing Diagram of PH3 JSB—Instruction	3-29	Timing Diagram of PH3, DIR. MPY—Instr. (Operand Address not in A/B-Register)	3-66
Compare Instruction	3-30	Execution of Multiplication	3-67
ROM JMP, JSB or CJMP Executions	3-31	The Binary Division	3-68
ROM JMP and JSB Control Signals	3-32	DIV—Instruction	3-69
Hardware Control Signals Created By ROM JMP, JSB and CJMP	3-33	DIV—Instruction (Cont'd.)	3-70
ROM Control Logic	3-34	DIV—Instruction (Cont'd.)	3-71
Flow Chart of CP—Instruction	3-35	DIV—Instruction (Cont'd.)	3-72
Flow Chart of CP—Instruction (Cont'd.)	3-36	DST—Instruction	3-73
Timing Diagram of PH3, CP—Instruction (Operand Address is not in A/B-Register)	3-37	LSL—Instruction	3-74

PHASE CONTROL LOGIC



A WAITING DMA INTERRUPT
IMPOSES A PH5 WHICH:

- 1) INHIBITS A CHANGE IN THE PRESENT (CPU) PHASE
- 2) FREEZES THE CPU
- 3) CREATES ITS OWN CONTROL SIGNALS TO ACCOMPLISH
A DMA DATA TRANSFER

* IF ENTER SUBROUTINE
& WANT TO TURN OFF
INTERRUPT (CLF00 IN
FIRST SUBROUTINE
LOC. CANNOT BE
INTERRUPTED

BOOLEAN EQUATIONS FOR PHASE CONTROL

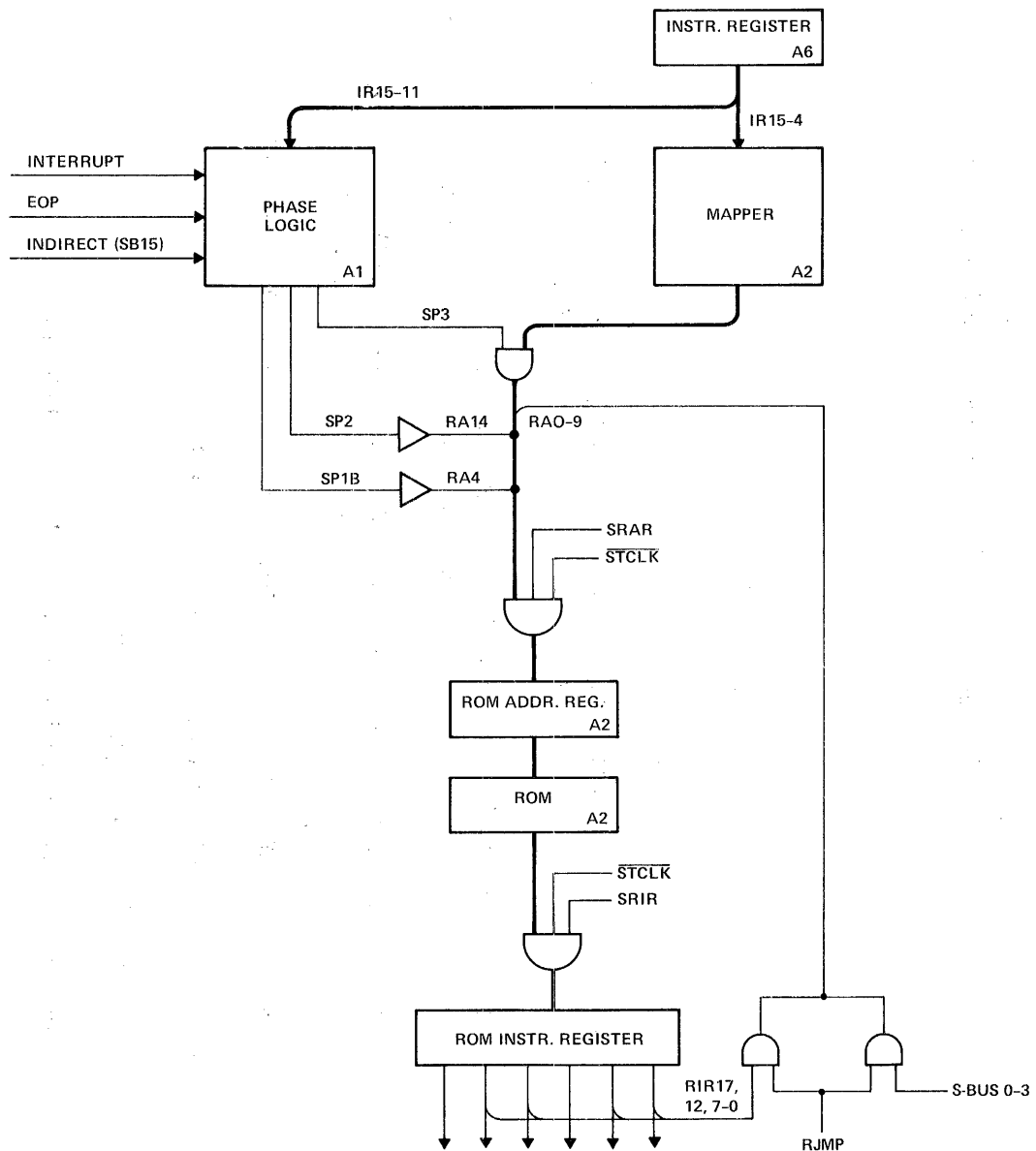
$$\text{SPH1A} = \text{PH3} \cdot \text{EOP} \cdot \overline{\text{LOOP}} \cdot [\text{INT} + \text{IR15} \cdot \overline{\text{IMPV}} (\text{JMP} + \text{JSB})]$$

$$\begin{aligned} \text{SPH1B} = & \text{PH2} \cdot \text{EOP} \cdot \overline{\text{LOOP}} \cdot \text{INT} (\text{SB15} \cdot \text{CT3} + \overline{\text{JMP} + \text{JSB}}) + \\ & \text{PH3} \cdot \text{EOP} \cdot \overline{\text{LOOP}} \cdot \text{INT} (\overline{\text{JMP} + \text{JSB}} + \overline{\text{IR15}} + \text{IMPV}) + \\ & \text{PH1A} \cdot \text{EOP} \cdot \overline{\text{LOOP}} \cdot \text{INT} \cdot \text{MRG} [\overline{\text{JMP} + \text{JSB}} + \overline{\text{JMP}} (\overline{\text{IR15}} + \text{IMPV})] \end{aligned}$$

$$\begin{aligned} \text{SPH3} = & \text{PH1A} \cdot \text{EOP} \cdot \overline{\text{LOOP}} [\overline{\text{INT}} (\overline{\text{IR15}} + \text{IMPV}) + \text{JMP} (\overline{\text{IR15}} + \text{IMPV}) + \overline{\text{MRG}}] + \\ & \text{PH2} \cdot \text{EOP} \cdot \overline{\text{LOOP}} [\overline{\text{INT}} \cdot \overline{\text{SB15}} + (\text{JMP} + \text{JSB}) \cdot \overline{\text{SB15}}] + \\ & \text{PH1B} \cdot \text{EOP} \cdot \overline{\text{LOOP}} [\overline{\text{IR15}} + \text{IMPV} + \overline{\text{MRG}}] \end{aligned}$$

$$\begin{aligned} \text{SPH2} = & \text{MRG} \cdot \text{EOP} \cdot \overline{\text{LOOP}} \{ \text{PH1A} \cdot \text{IR15} \cdot \overline{\text{IMPV}} (\overline{\text{INT}} + \text{JMP} + \text{JSB}) + \\ & \text{PH2} \cdot \text{SB15} [\overline{\text{INT}} + (\text{JMP} + \text{JSB}) \cdot \overline{\text{CT3}}] + \\ & \text{PH1B} \cdot \text{IR15} \cdot \overline{\text{IMPV}} \} \end{aligned}$$

MAPPING THE ROM ADDRESS



BOOLEAN EQUATIONS FOR MAPPER OUTPUT

$$RA0 = (ASG + ASR + ASL) \cdot SPH3$$

$$RA1 = (LSR + LSL) \cdot SPH3$$

$$RA2 = SP1B + SPH2 + [(STC + CLC + LI^*) + (JMP + AD^* + CP^* + LD^* + ST^*) + (RRR + RRL) + (CL^* + CC^*)] \cdot SPH3$$

$$RA3 = SPH2 + [(OT^* + STC + CLC) + (MPY + DLD) + (AND + IOR + JSB + ISZ + CP^* + ST^*) + (CM^* + CC^*)] \cdot SPH3$$

$$RA4 = [(DIV + DST) + (XOR + IOR + JMP + ISZ + LD^* + ST^*) + (MI^* + LI^* + OT^* + STC + CLC) + ASG] \cdot SPH3$$

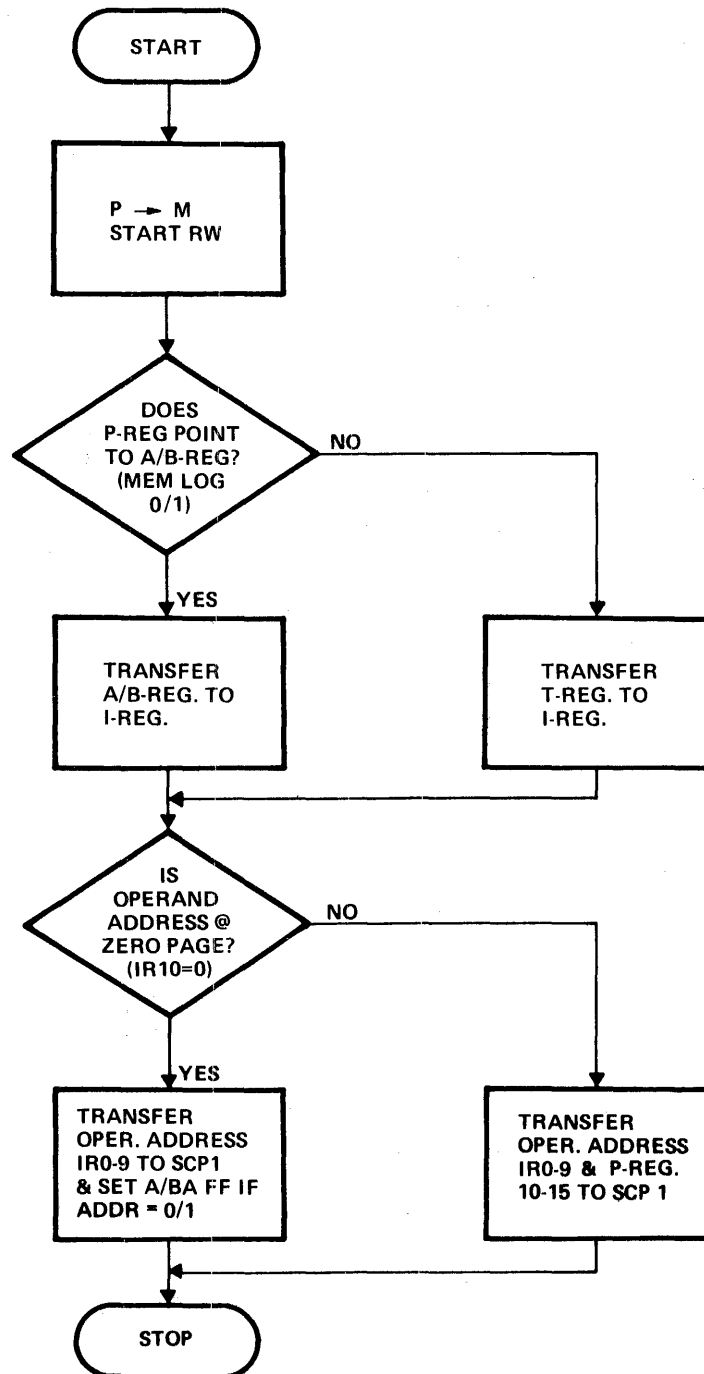
$$RA5 = [(ASR + LSR + RRR) + (JSB + ISZ) + (LD^* + ADD) + IOG] \cdot SPH3$$

$$RA6 = [(DST + DLD) + (MRG + SRG)] \cdot SPH3$$

$$RA7 = EAG_{(FIX\ POINT)} \cdot SPH3$$

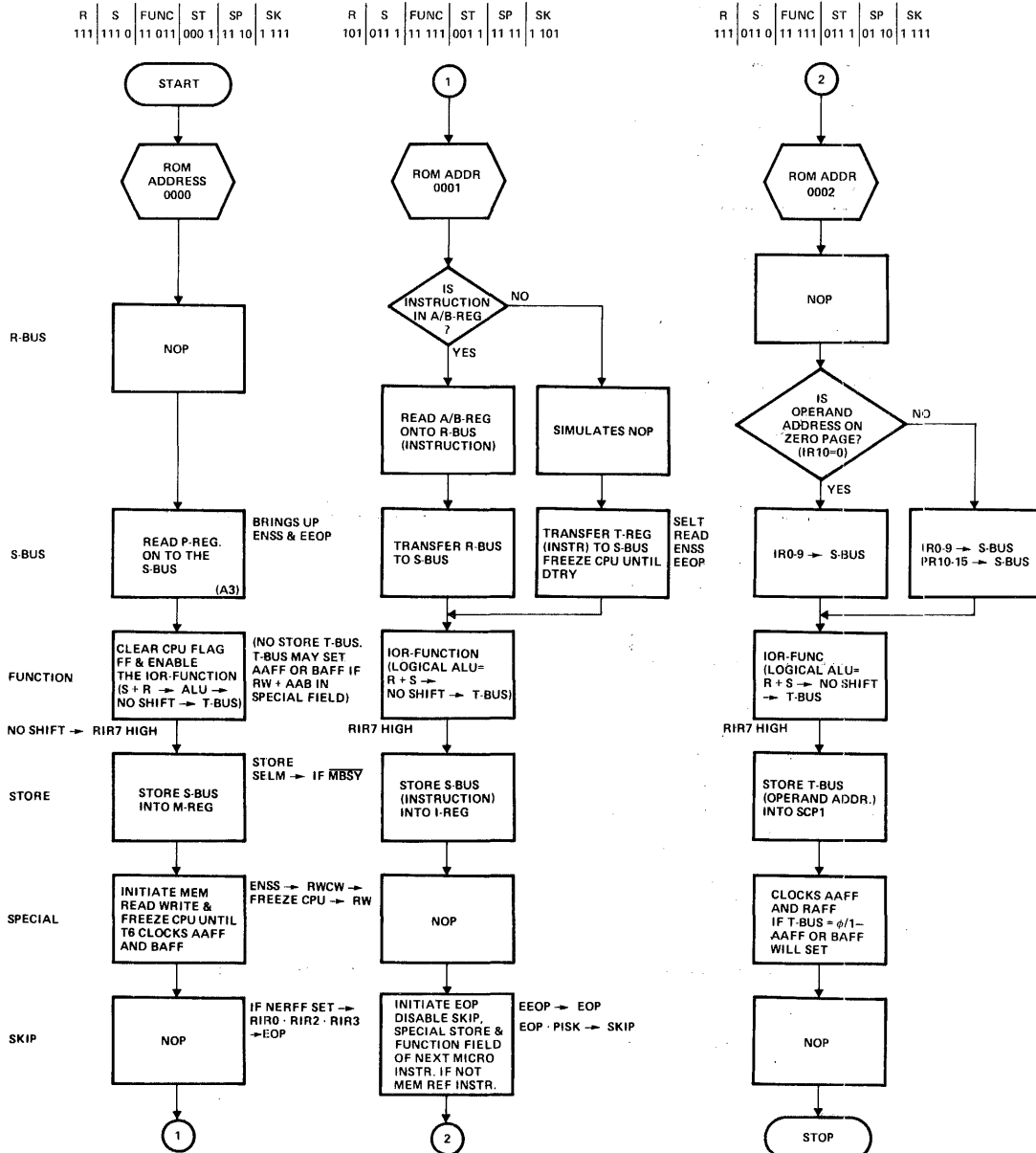
$$RA8 = (FAD + FSB + FMY + FDV + FIX + FLT) \cdot SPH3$$

FETCH PHASE



PH1A FLOW CHART

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS 23-21	S-BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0
0000	77330757	PH1A	—	P	CFLG	M	RW	—
0001	53771775		AAB	COND	IOR	IR	—	EOP
0002	73373557		—	ADR	IOR	S1	AAB	—



CPU FREEZE

WHEN A CPU FREEZE OCCURS ALL OF THE CLOCK SIGNALS WITH THE EXCEPTION OF $\overline{\text{CLK3}}$ ARE INHIBITED. THIS PREVENTS THE RAR AND RIR FROM BEING CHANGED AND INHIBITS EXECUTION OF THE ROM MICROINSTRUCTION.

FREEZE SOURCES:

PH5

MEM PROTECT

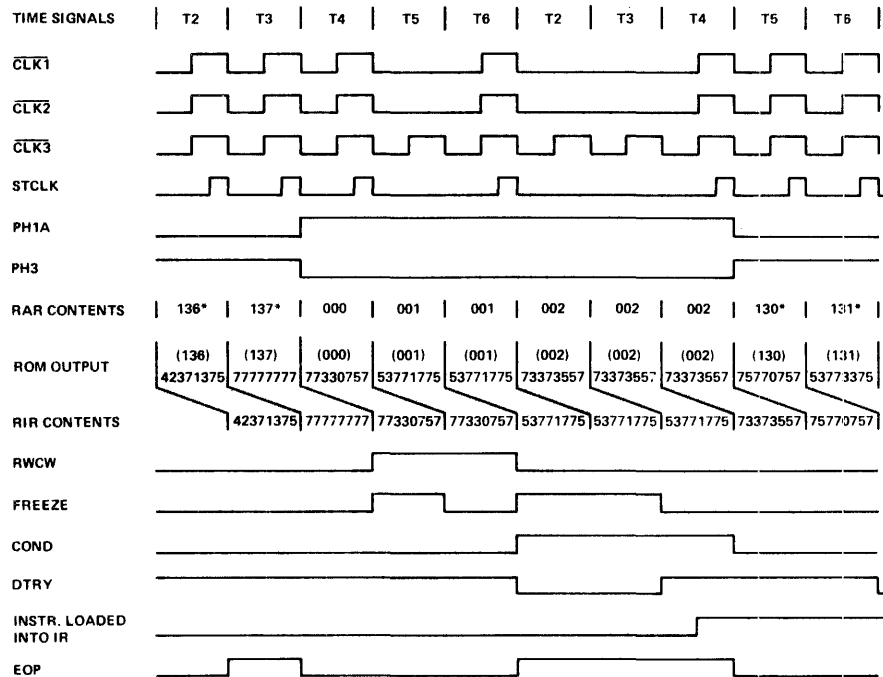
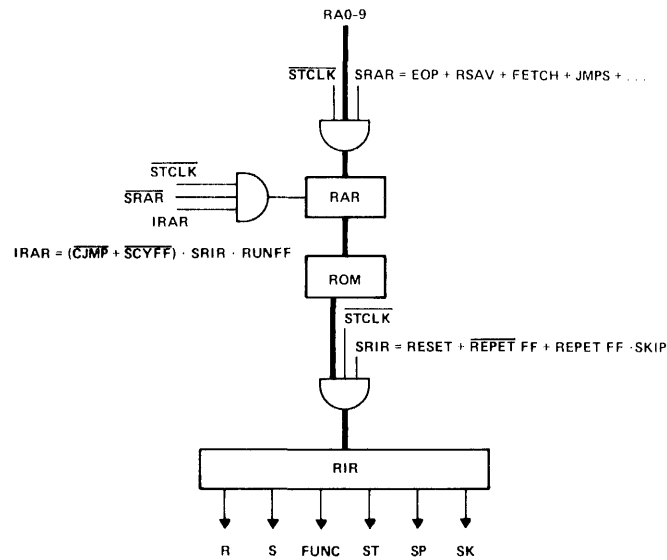
T OR COND IN S BUS AND $\overline{\text{ABFF}}$ → FREEZE UNTIL DTRY

RW OR CW AND $\overline{\text{T6}}$

MBSY AND M IN STORE FIELD

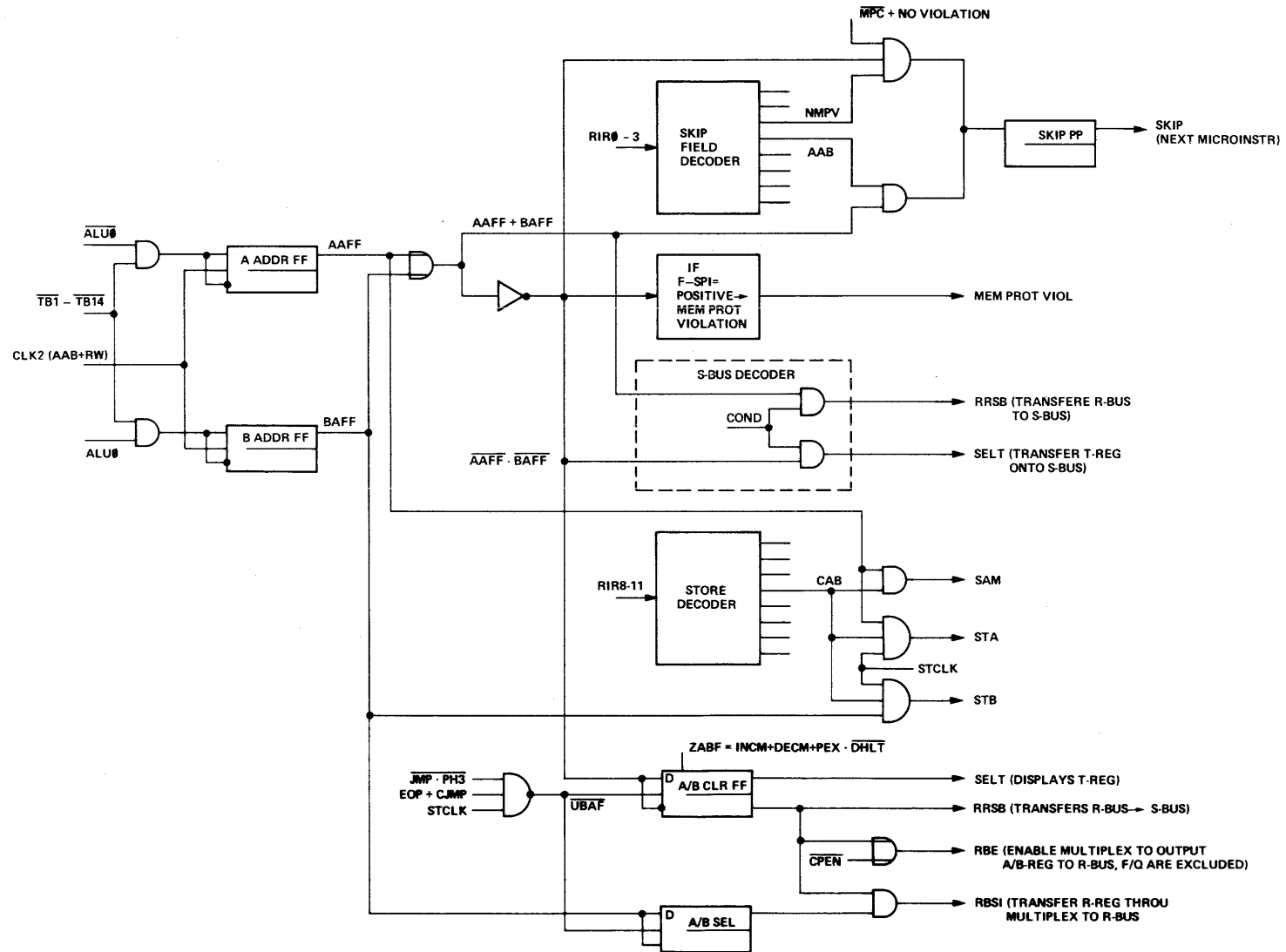
DIV AND $\overline{\text{DT}}$

TIMING DIAGRAM OF PH1A
(INSTRUCTION DOES NOT RESIDE IN A/B-REGISTER)

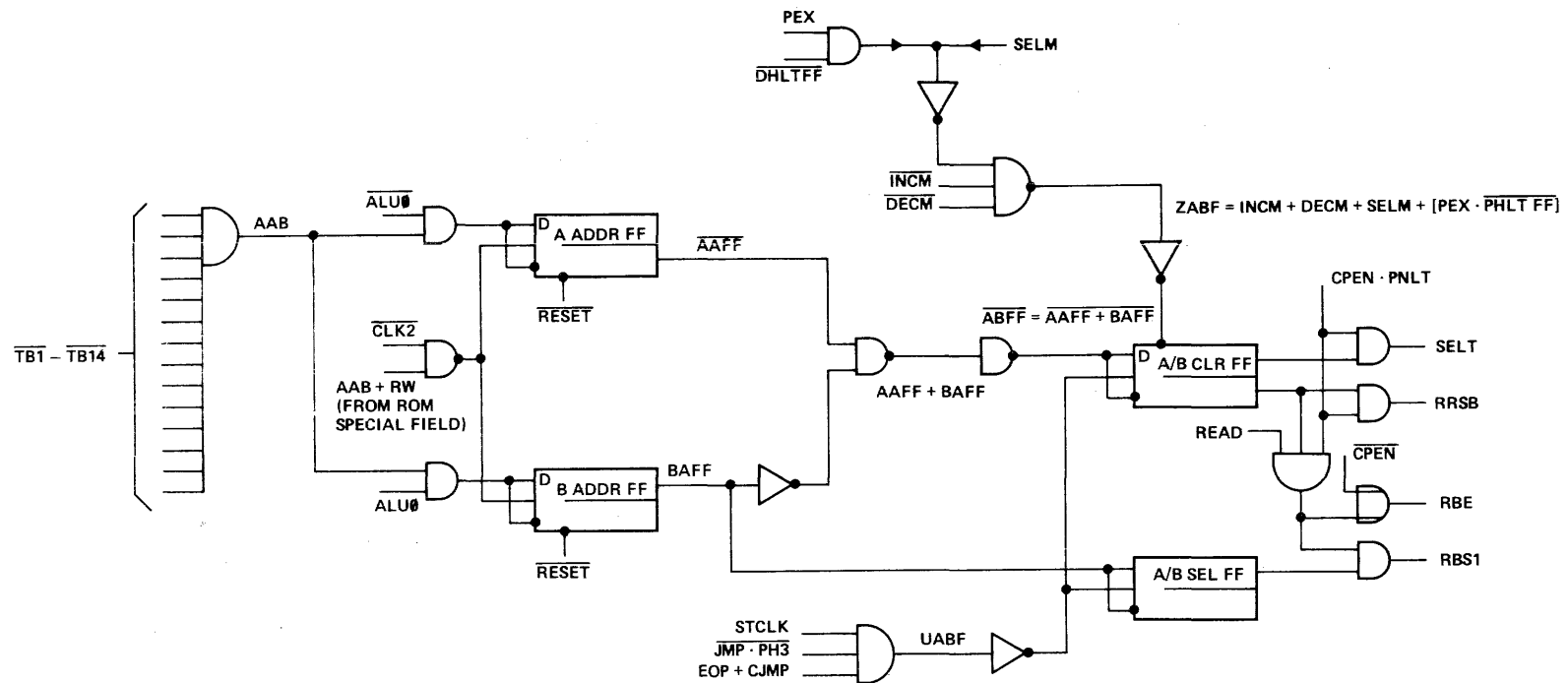


*THESE RAR CONTENTS ARE ARBITRARILY CHOSEN

SIMPLIFIED DIAGRAM OF A/B ADDRESSABLE LOGIC



A/B ADDR. FF, A/B CLRFF & A/B SEL FF. LOGIC IN HALT/SINGLE CYCLE MODE



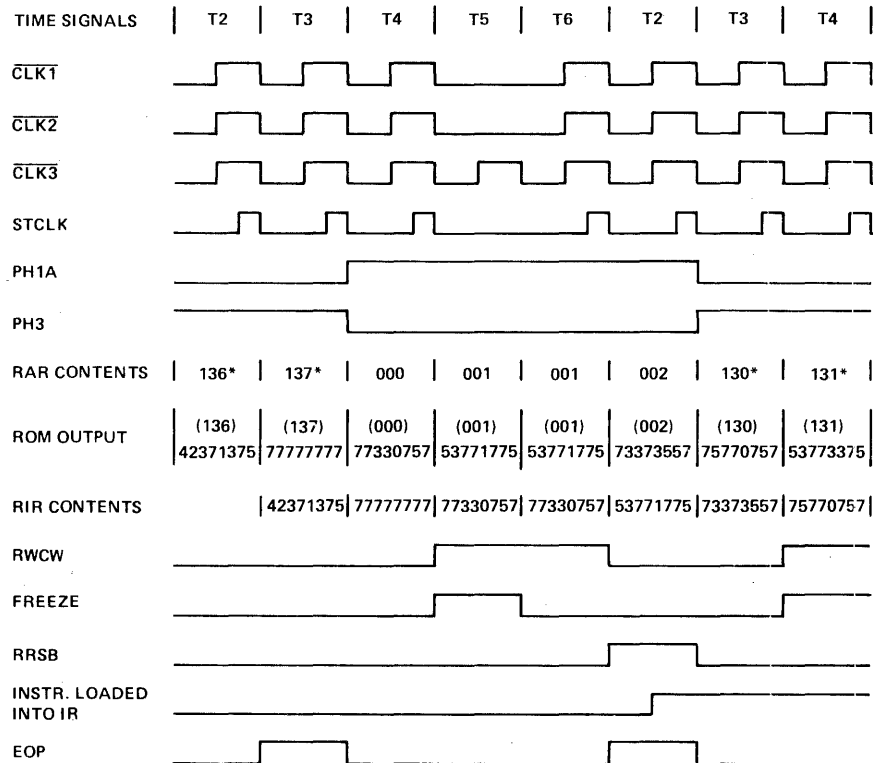
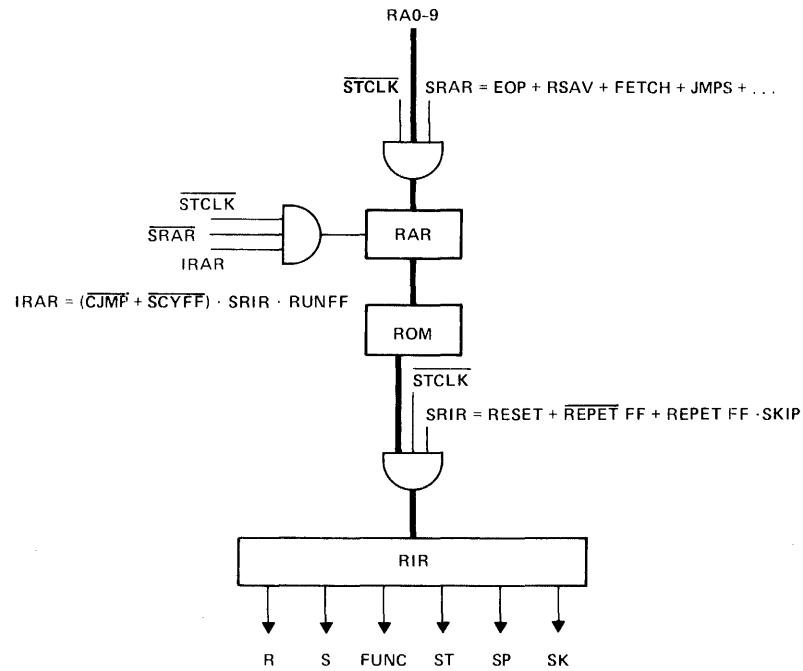
- 1.) A/B CLR FF SETS IF NEITHER A- NOR B-ADDR. FF'S HAVE BEEN SET. $\therefore$ SELT $\cdot$ READ $\rightarrow$ TRANSFERS T-REG TO S-BUS IF COMPUTER IN HLT MODE.
- 2.) A/B CLR FF RESET IF A- OR B-ADDR. FF IS SET. $\therefore$ RBE ENABLES MULTIPL., $\overline{RBS1} \rightarrow$ A-REG TO R-BUS, RBS1 $\rightarrow$ B-REG TO R-BUS. RRSB $\rightarrow$ TRANSFERS A/B-REG CONTENTS FROM R- TO S-BUS.
- 3.) S-BUS CONTENTS $\rightarrow$ I/O-BUS WITH SIOB (FROM FRONT PANEL) $\rightarrow$ TO DISPLAY REGISTER.
- 4.) C JMP IS INCLUDED IN THE EQUATION OF UABF TO ENABLE THE PROPER DISPLAY OF EAG-MEMORY REFERENCE INSTRUCTIONS (MPY, DIV, DST, DLD) IF IN SINGLE CYCLE MODE.
- 5.) UABF IS DISABLED AT THE END OF JMP $\cdot$ PH3. THIS WILL INHIBIT THE A/B CLR FF TO BE SET ($\overline{ABFF}$ IS HIGH AT JMP $\cdot$ PH3 DUE TO MICRO INSTRUCTION @ ROM ADDRESS 124) $\therefore$ SELT $\rightarrow$ THE JMP INSTRUCTION IS DISPLAYED DURING PH1A AND PH3.

ROM MICROINSTRUCTION FUNCTION FIELD DECODING AND EQUATIONS
CREATED BY ARITHMATIC LOGIC FUNCTION GENERATOR

ROM INSTRUCTION REGISTER BITS					FUNCTION FIELD (5 BITS)	MC	FN0	FN1	FN2	FN3	CIN	OPERATION	EQUATION OF ARITHMATIC LOGIC FUNCTION GENERATOR
16	15	14	13	12									
1	1	1	1	1	IOR	1	0	1	1	1	0	LOGIC OPERATIONS	ALU = R+S
1	1	1	1	0	SOV	1	0	1	1	1	0		
1	1	1	0	1	CLO	1	0	1	1	1	0		
1	1	1	0	0	SFLG	1	0	1	1	1	0		
1	1	0	1	1	CFLG	1	0	1	1	1	0		
1	1	0	1	0	LWF	1	0	1	1	1	0		
1	1	0	0	1	—	1	0	1	1	1	0		
1	1	0	0	0	ARS	1	0	1	1	1	0		
1	0	1	1	1	CRS	1	0	1	1	1	0		
1	0	1	1	0	LGS	1	0	1	1	1	0		
1	0	1	0	1	RSB	1	0	1	1	1	0		
1	0	1	0	0	CJMP	1	0	1	1	1	0		
1	0	0	1	1	JMP	1	0	1	1	1	0		
1	0	0	1	0	JMP	1	0	1	1	1	0		
1	0	0	0	1	JSB	1	0	1	1	1	0		
1	0	0	0	0	JSB	1	0	1	1	1	0		
0	1	1	1	1	—	1	0	0	0	0	0	ARITHMATIC OPERATIONS	ALU = $\bar{R}$
0	1	1	1	0	XOR	1	0	1	1	0	0		ALU = $R \oplus S$
0	1	1	0	1	NOR	1	1	0	0	0	0		ALU = $\overline{R+S}$
0	1	1	0	0	AND	1	1	1	0	1	0		ALU = R·S
0	1	0	1	1	ADD	0	1	0	0	1	0		ALU = R+S
0	1	0	1	0	ADDO	0	1	0	0	1	0		ALU = R+S
0	1	0	0	1	INC	0	1	0	0	1	1		ALU = R+S+1
0	1	0	0	0	INCO	0	1	0	0	1	1		ALU = R+S+1
0	0	1	1	1	—	0	0	0	0	0	0		ALU = R
0	0	1	1	0	DEC	0	0	1	1	0	0		ALU = R-S-1
0	0	1	0	1	SUB	0	0	1	1	0	1		ALU = R-S
0	0	1	0	0	DIV	0	0	1	1	0	1		ALU = R-S
0	0	0	1	1	MPY	0	1*	0	0	1*	0		ALU = R+S
0	0	0	1	0	P1A	0	0	0	0	0	0		ALU = R
0	0	0	0	1	RFE	0	0	0	0	0	0		ALU = R
0	0	0	0	0	RFI	0	0	0	0	0	0		ALU = R

*IF A-REG BIT 0 SET

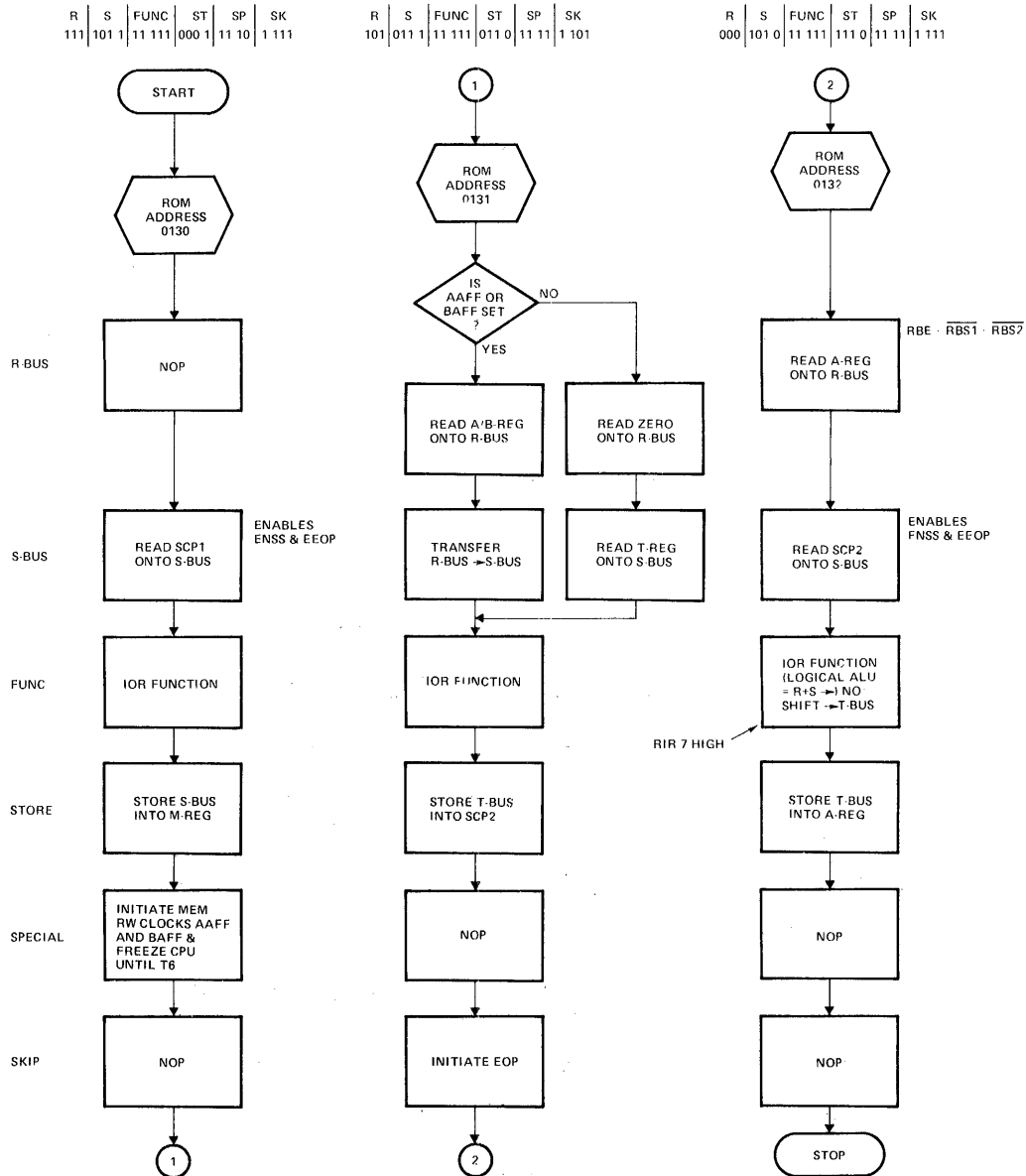
TIMING DIAGRAM OF PH1A
(INSTRUCTION RESIDES IN A/B-REGISTER)



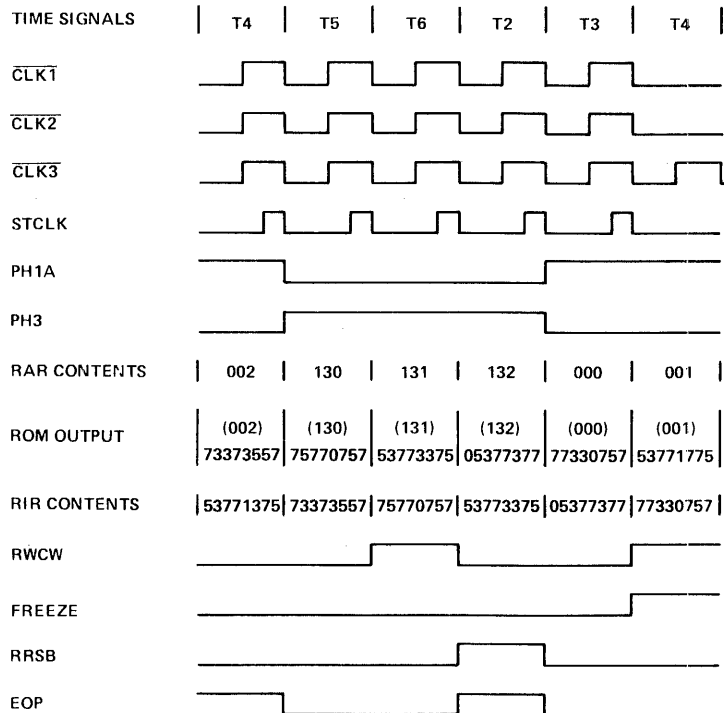
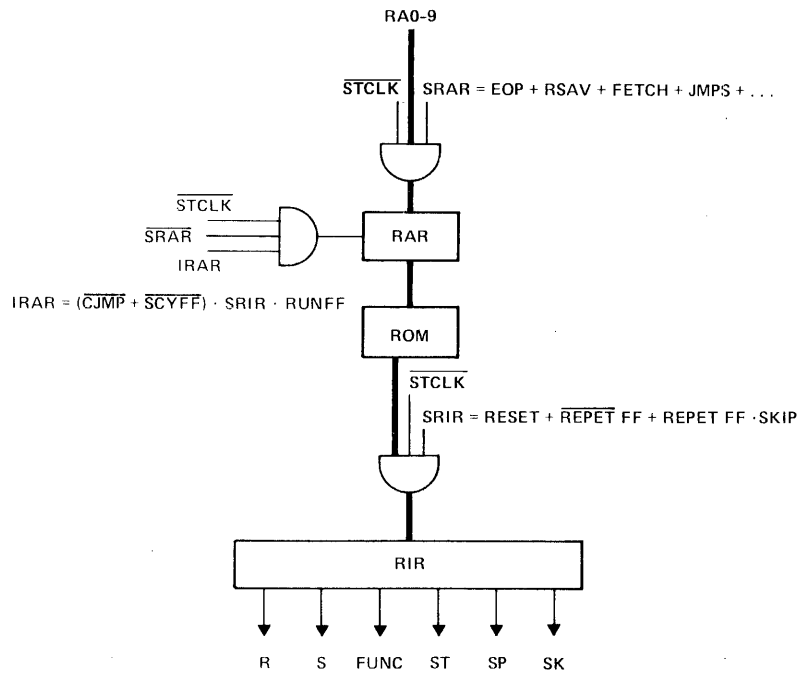
*THESE RAR CONTENTS ARE ARBITRARILY CHOSEN

FLOW CHART OF IOR -- INSTRUCTION

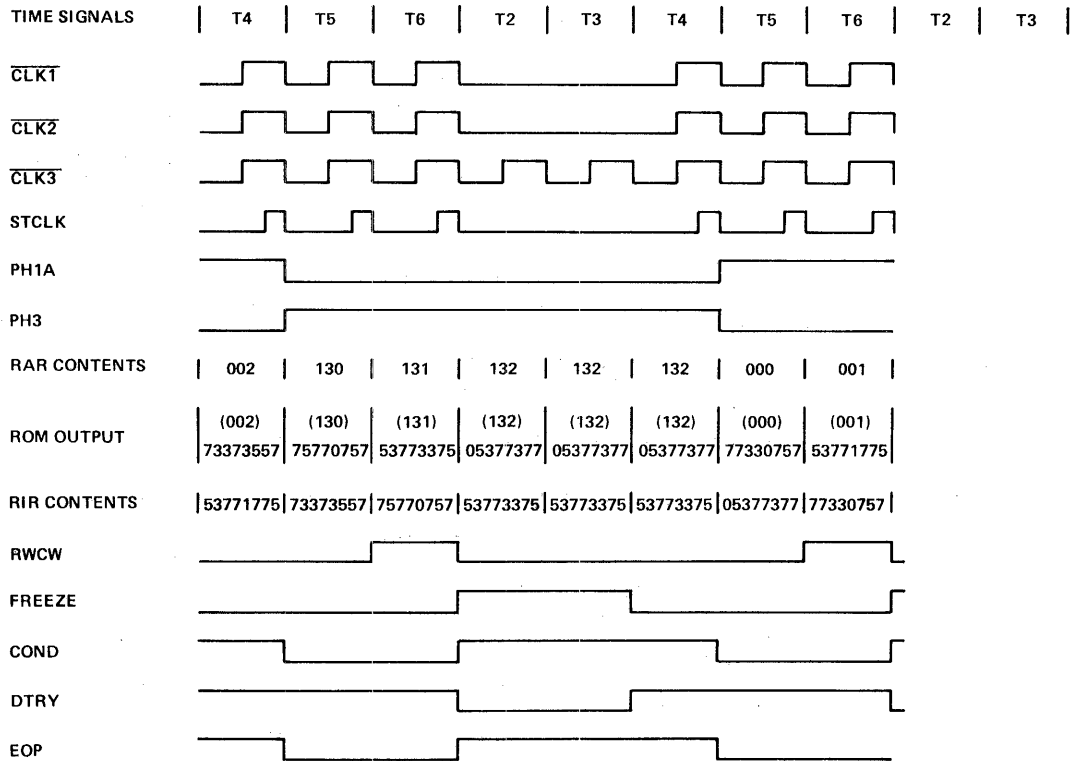
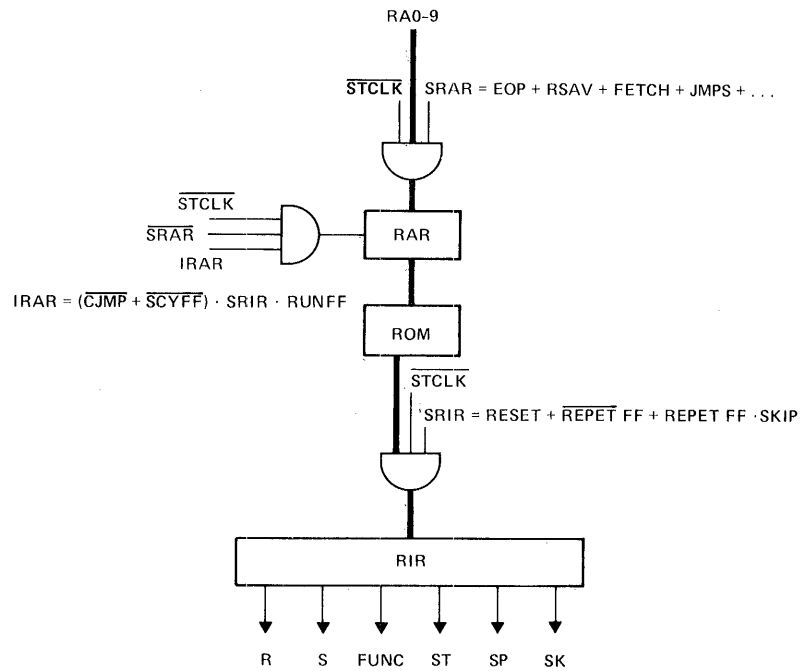
ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS 23-21	S-BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0
0130	75770757	IOR	--	S1	IOR	M	RW	--
0131	53773375		AAB	COND	IOR	S2	--	EOP
0132	05377377		A	S2	IOR	A	--	--



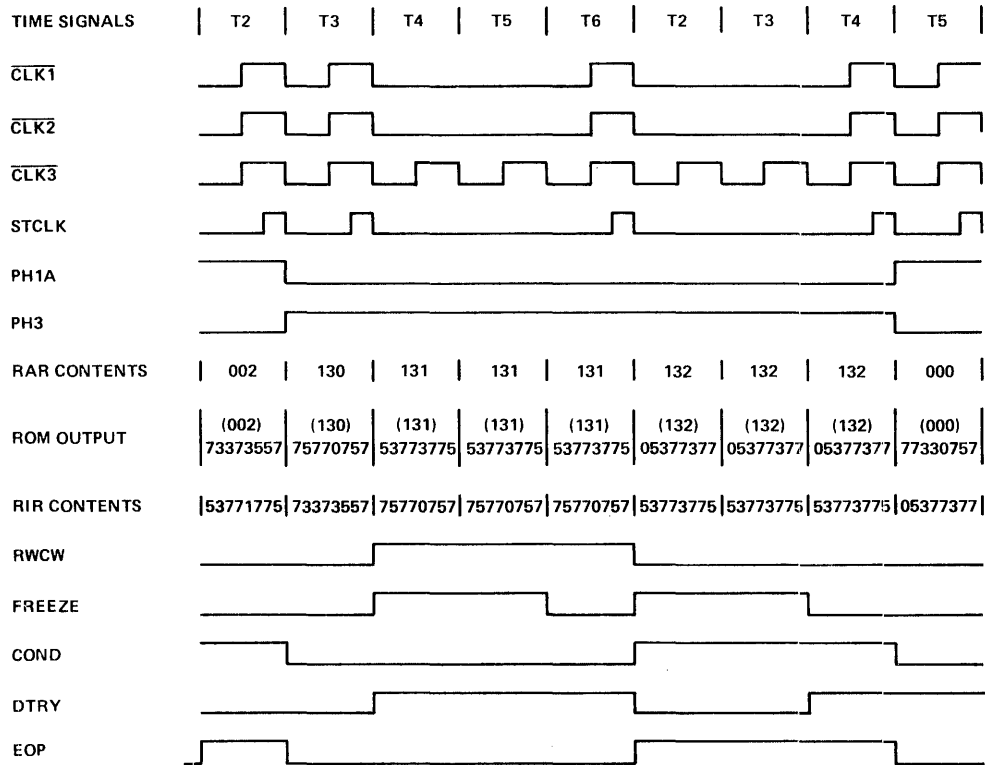
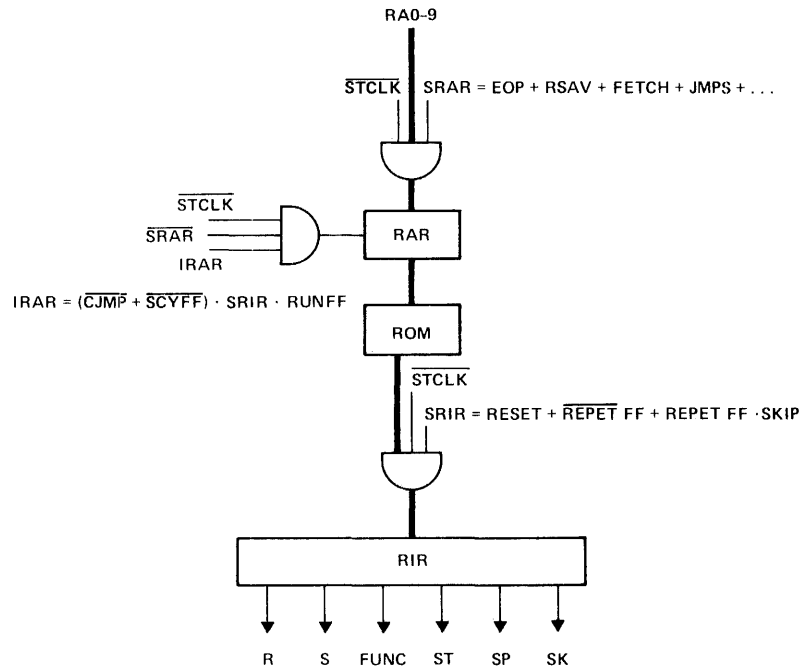
**TIMING DIAGRAM OF PH3 IOR INSTRUCTION
(OPERAND IS IN A/B-REGISTER)**



TIMING DIAGRAM OF PH3 IOR INSTRUCTION (OPERAND IS NOT IN A/B-REGISTER)



TIMING DIAGRAM OF PH3 IOR INSTRUCTION
 (INSTRUCTION RESIDES IN A/B-REG. & OPERAND IS NOT IN A/B-REGISTER)



FLOW CHART OF AND/XOR – INSTRUCTION

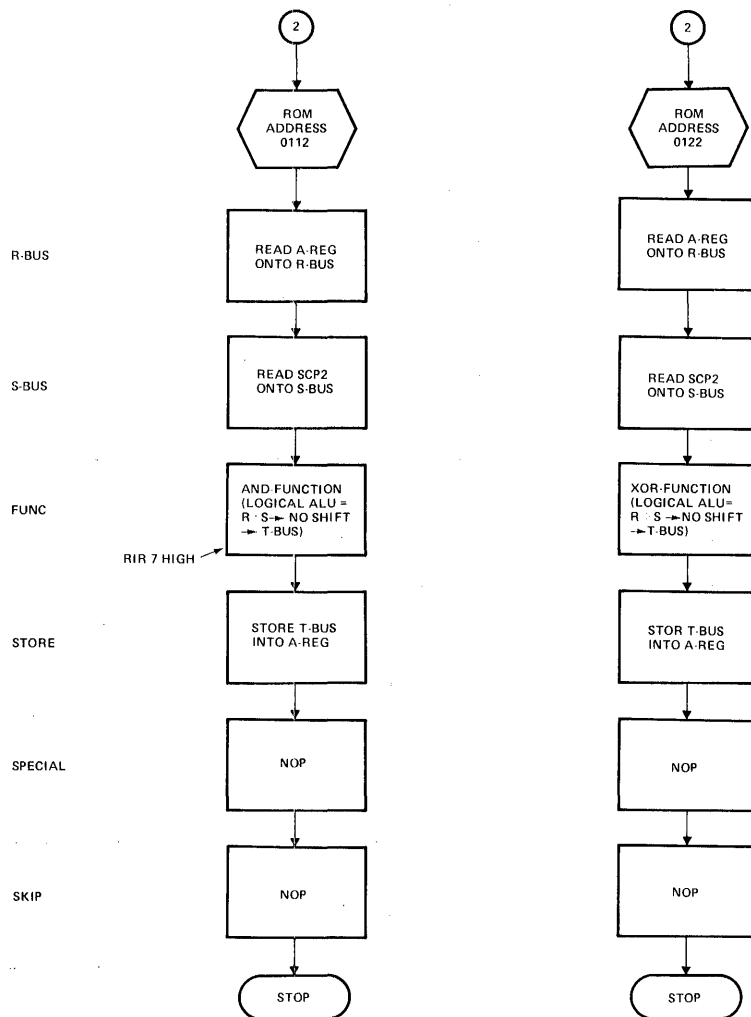
ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS 23-21	S-BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0
0110	75770757	AND	—	S1	IOR	M	RW	—
0111	53773375		AAB	COND	IOR	S2	—	EOP
0112	05147377		A	S2	AND	A	—	—
0120	75770757	XOR	—	S1	IOR	M	RW	—
0121	53773375		AAB	COND	IOR	S2	—	EOP
0122	05167377		A	S2	XOR	A	—	—

AND – INSTRUCTION

R	S	FUNC	ST	SP	SK
000	101 1	01 100	111 0	11 11	1 111

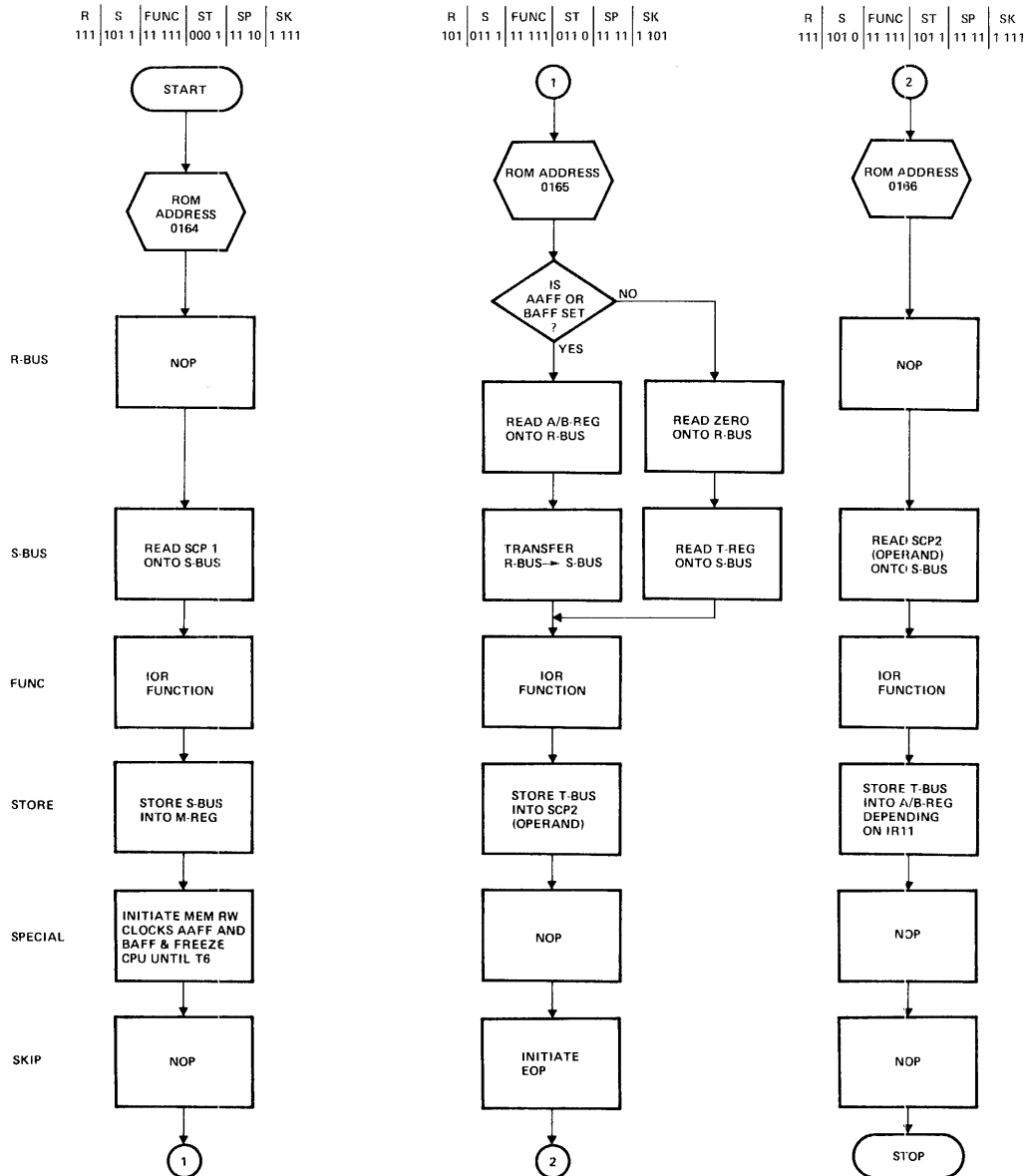
XOR – INSTRUCTION

R	S	FUNC	ST	SP	SK
000	101 0	01 110	111 0	11 11	1 111

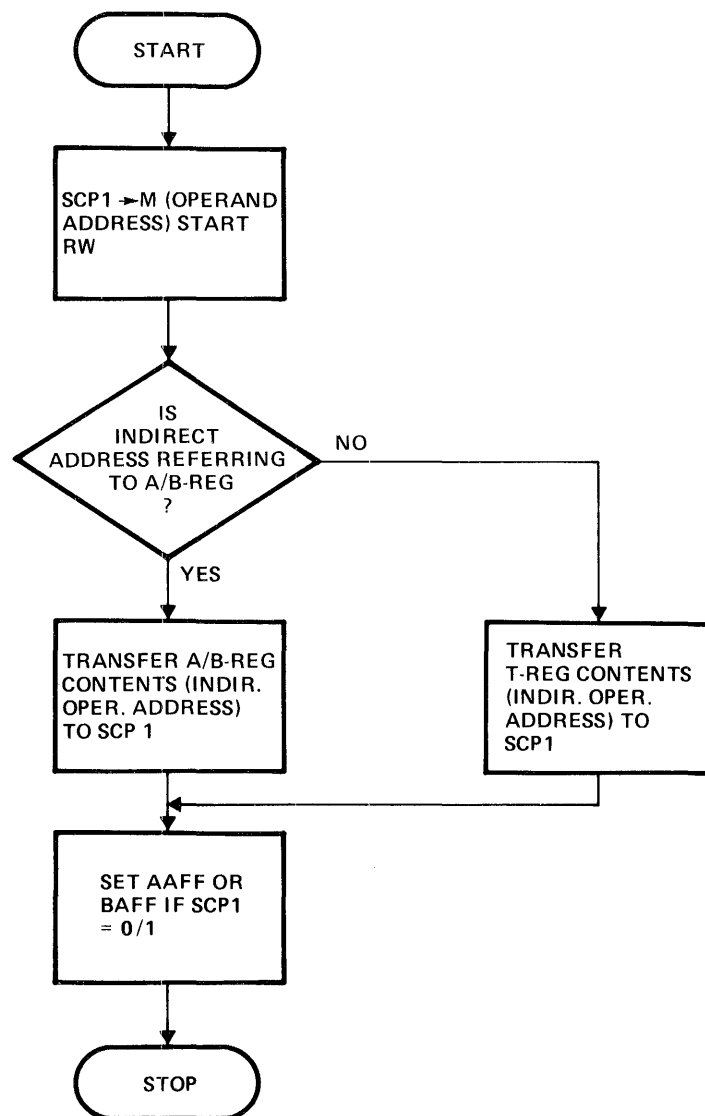


FLOW CHART OF LD* - INSTRUCTION

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS 23-21	S-BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0
0164	75770757	LD*	-	S1	IOR	M	RW	-
0165	53773375		AAB	COND	IOR	S2	-	EOP
0166	75375777		-	S2	IOR	CAB	-	-

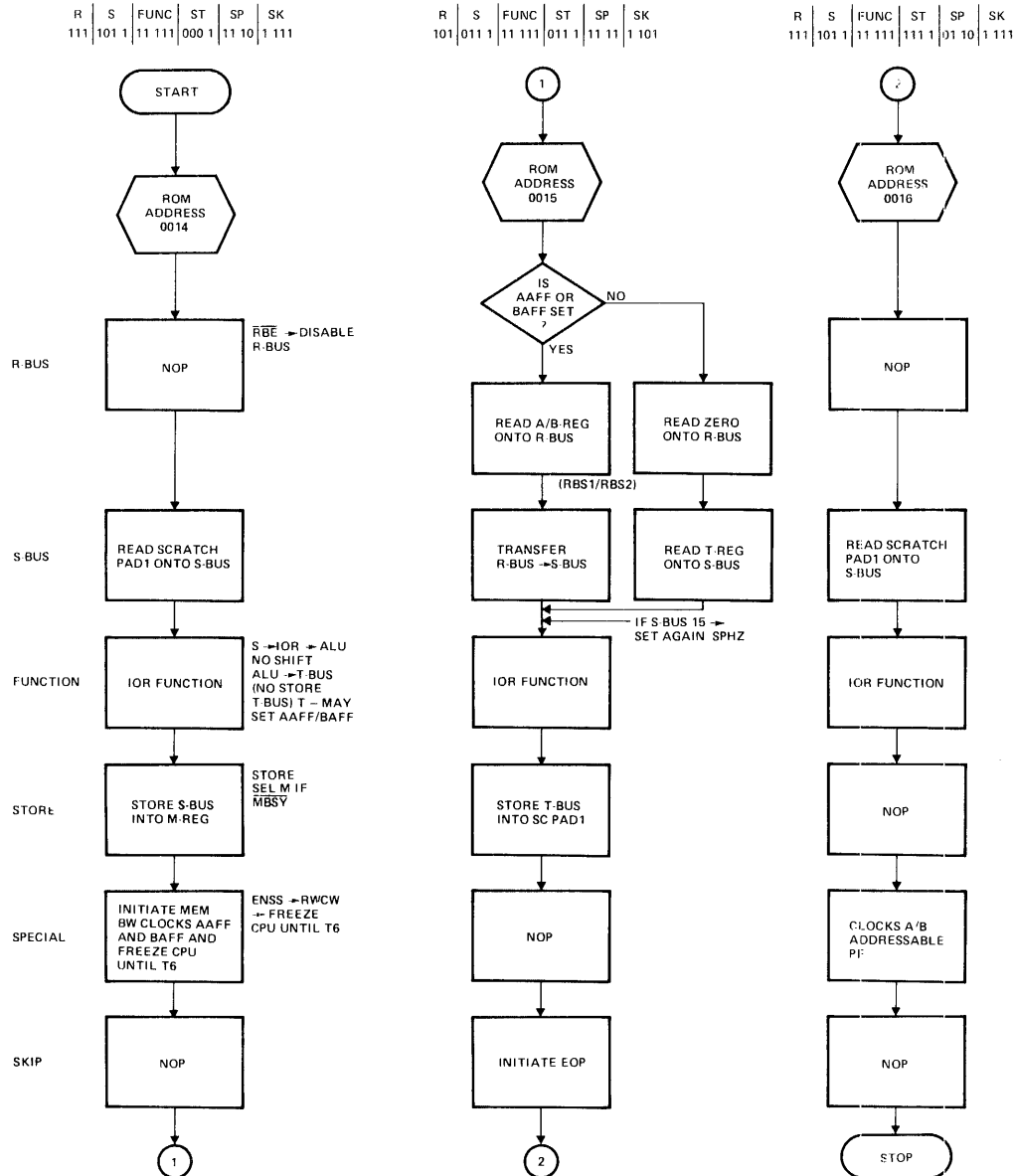


INDIRECT PHASE



FLOW CHART OF PH2

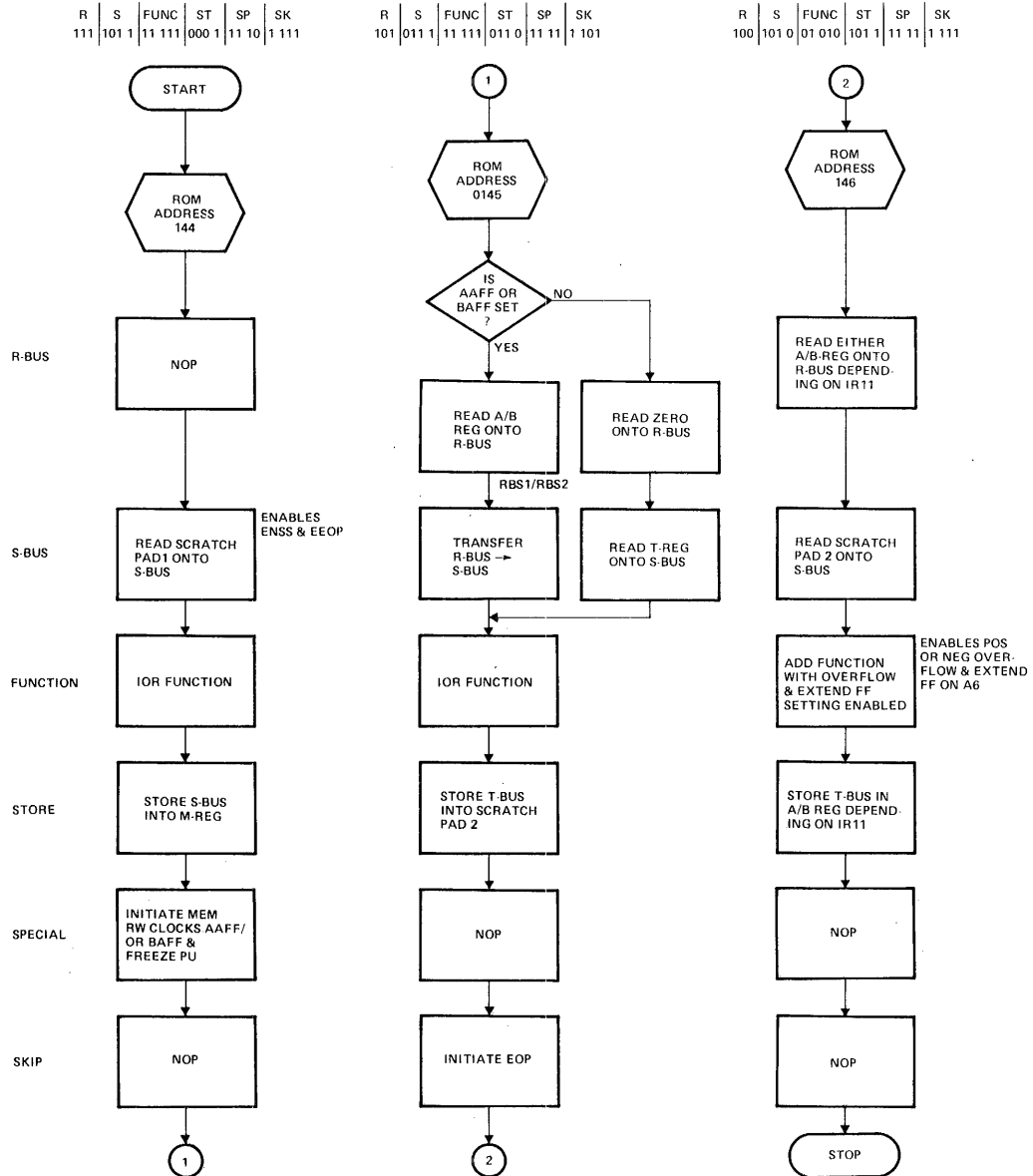
ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS 23-21	S-BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0
0014	75770757	PH2	—	S1	IOR	M	RW	—
0015	53773775		AAB	COND	IOR	S1	—	EOP
0016	75777557		—	S1	IOR	—	AAB	—



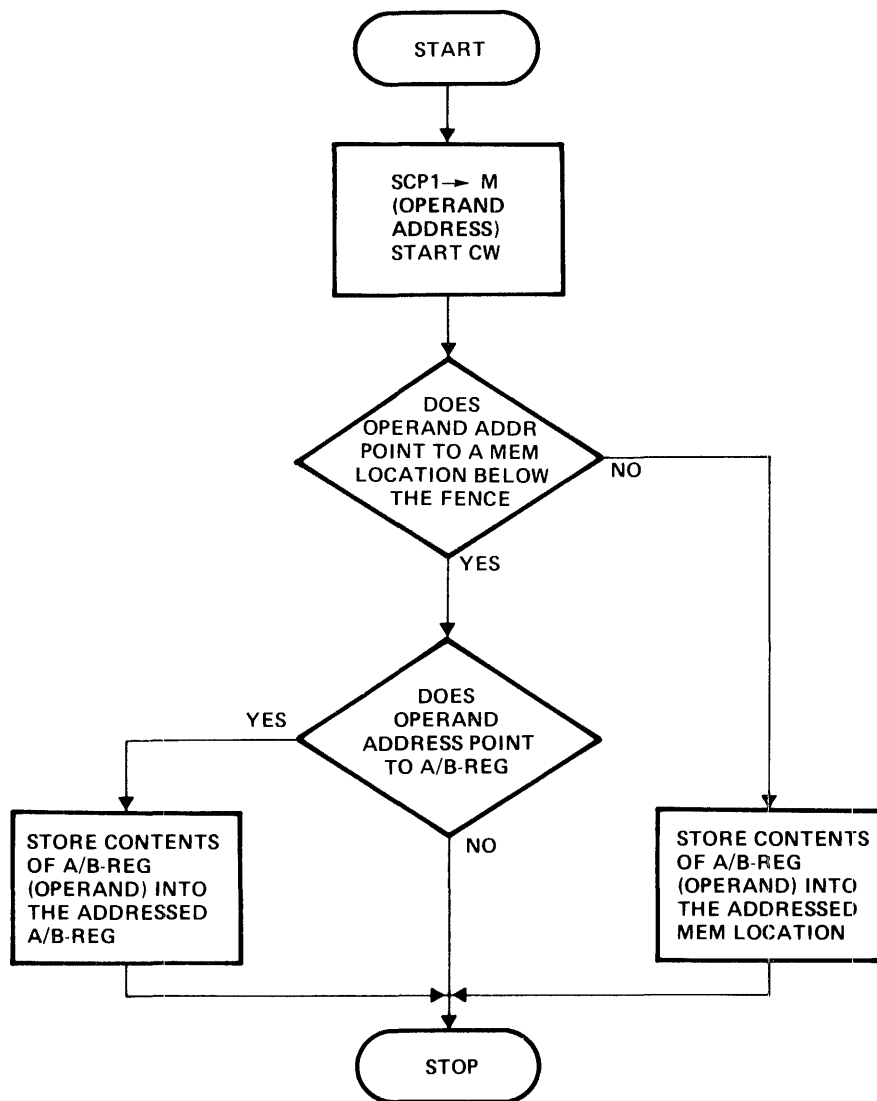
(CAN NOT LOOP ON PHASE 2 BECAUSE SCP1(STORE) IS OVERWRITTEN BY AAB/COND SCP1(S-BUS) IS ALTERED)

FLOW CHART OF ADD – INSTRUCTION

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS 23-21	S-BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0
0144	75770757	ADD	—	S1	IOR	M	RW	—
0145	53773375		AAB	COND	IOR	S2	—	EOP
0146	45125777		CAB	S2	ADDO	CAB	—	—

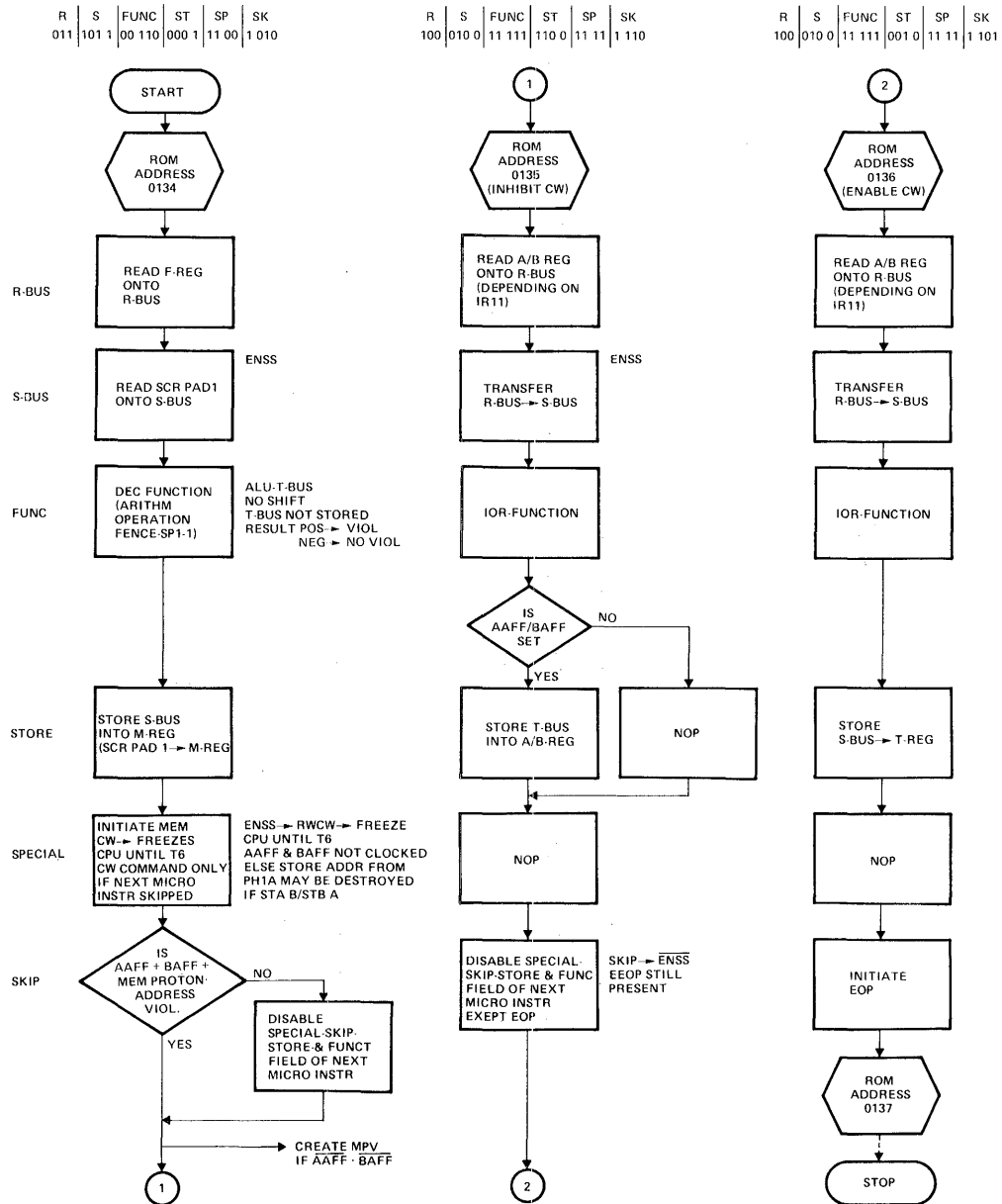


STA/B – INSTRUCTION



FLOW CHART OF ST* -- INSTRUCTION

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS 23-21	S-BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0
0134	35460712	ST*	F	S1	DEC	M	CW	NMPV
0135	42376376		CAB	RRS	IOR	AAB	..	UNC
0136	42371375		CAB	RRS	IOR	T	..	(EOP)
0137	77777777		..	..	IOR	..	..	..



ROM SKIPS

IF A ROM SKIP CONDITION IS SATISFIED, IT RESULTS IN THE NULLING OF THE NEXT INSTRUCTION. TO NULL THE INSTRUCTION, THE DECODING OF THE STORE, SPECIAL, AND SKIP FIELDS IS INHIBITED. THE DECODING OF SOME FUNCTIONS IS ALSO INHIBITED. (JMP, JSB, MPY, SFLG, ETC.)

THE SEQUENCE OF RAR AND RIR CONTENTS IN A MICROPROGRAM IS THE SAME WHETHER AN INSTRUCTION IS SKIPPED OR NOT!

EOP AND LEP CAN NEVER BE SKIPPED!

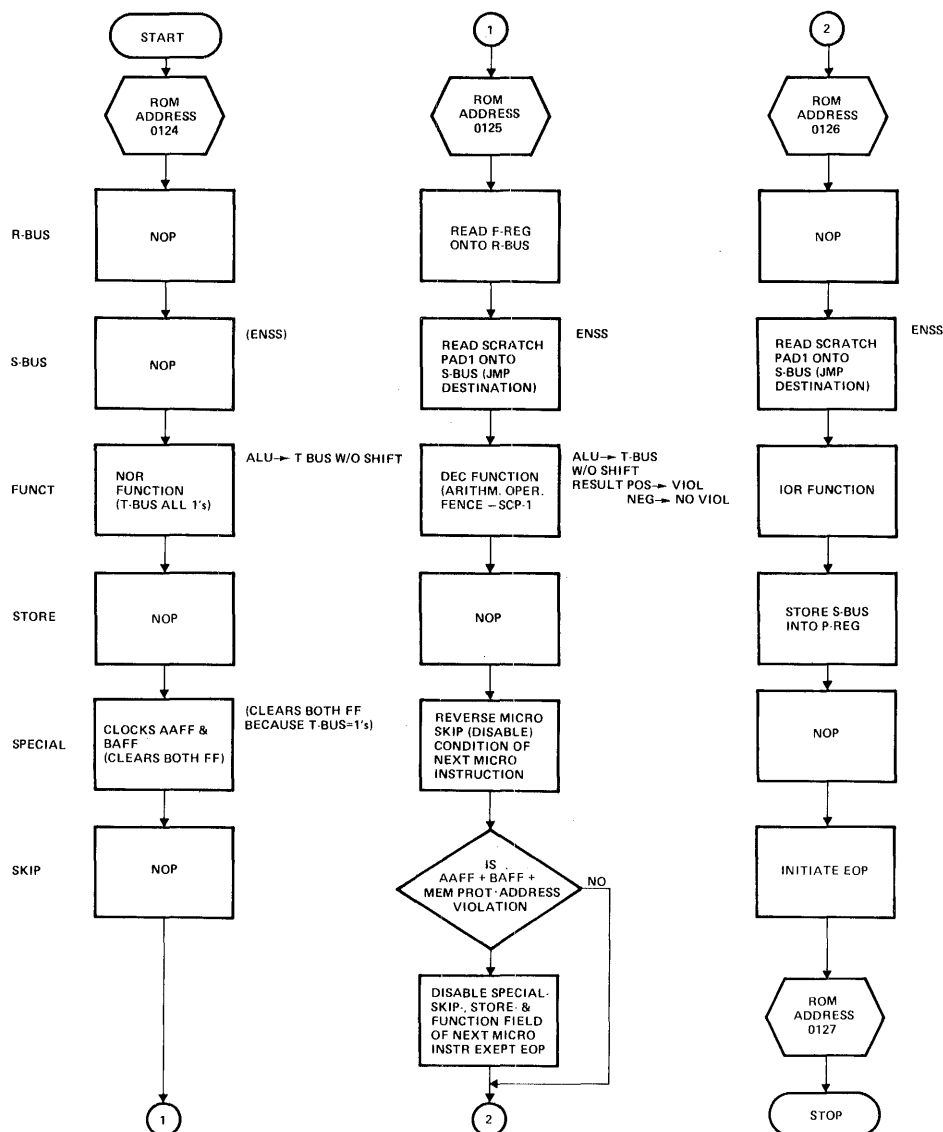
FLOW CHART OF JMP – INSTRUCTION

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS 23-21	S-BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0
0124	77557557	JMP	—	—	NOR	—	AAB	—
0125	35467412		F	S1	DEC	—	RSS	NMPV
0126	75774375		—	S1	IOR	P	—	EOP
0127	77777777		—	—	IOR	—	—	—

R S FUNC ST SP SK
111 111 1 01 101 111 1 01 101 111

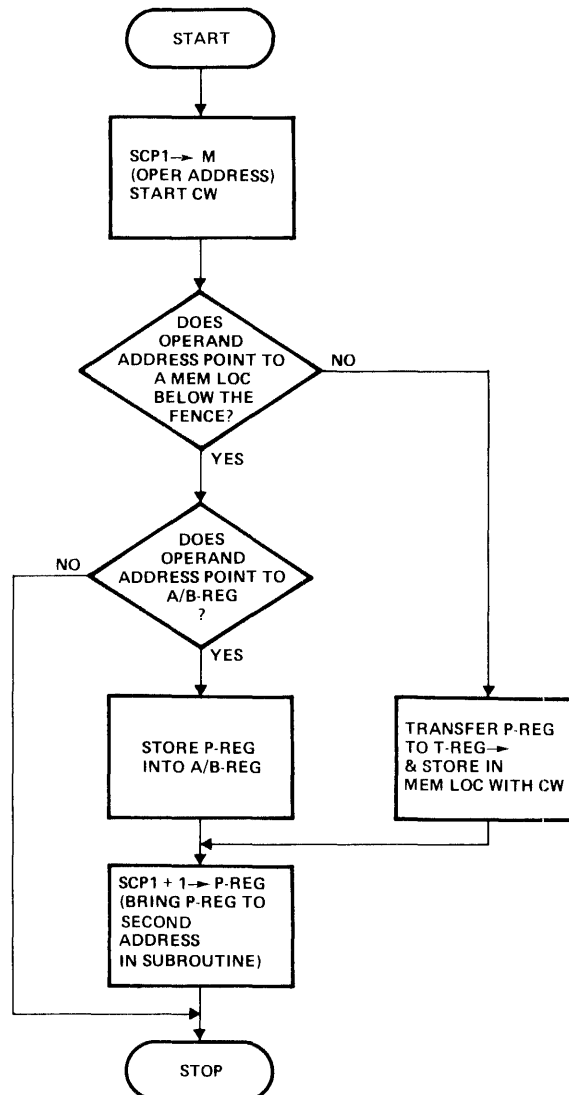
R S FUNC ST SP SK
011 101 1 00 110 111 1 00 00 1 010

R S FUNC ST SP SK
111 101 1 11 111 100 0 11 11 1 101



JSB – INSTRUCTION

MAIN PROGR.		SUBROUTINE	
LOC.	OP. CODE	LOC.	OP. CODE
125	LDA	200	NO
126	JSB	201	SZA
127	ADA	202	INA
130	HLT	203	JMP 200, I



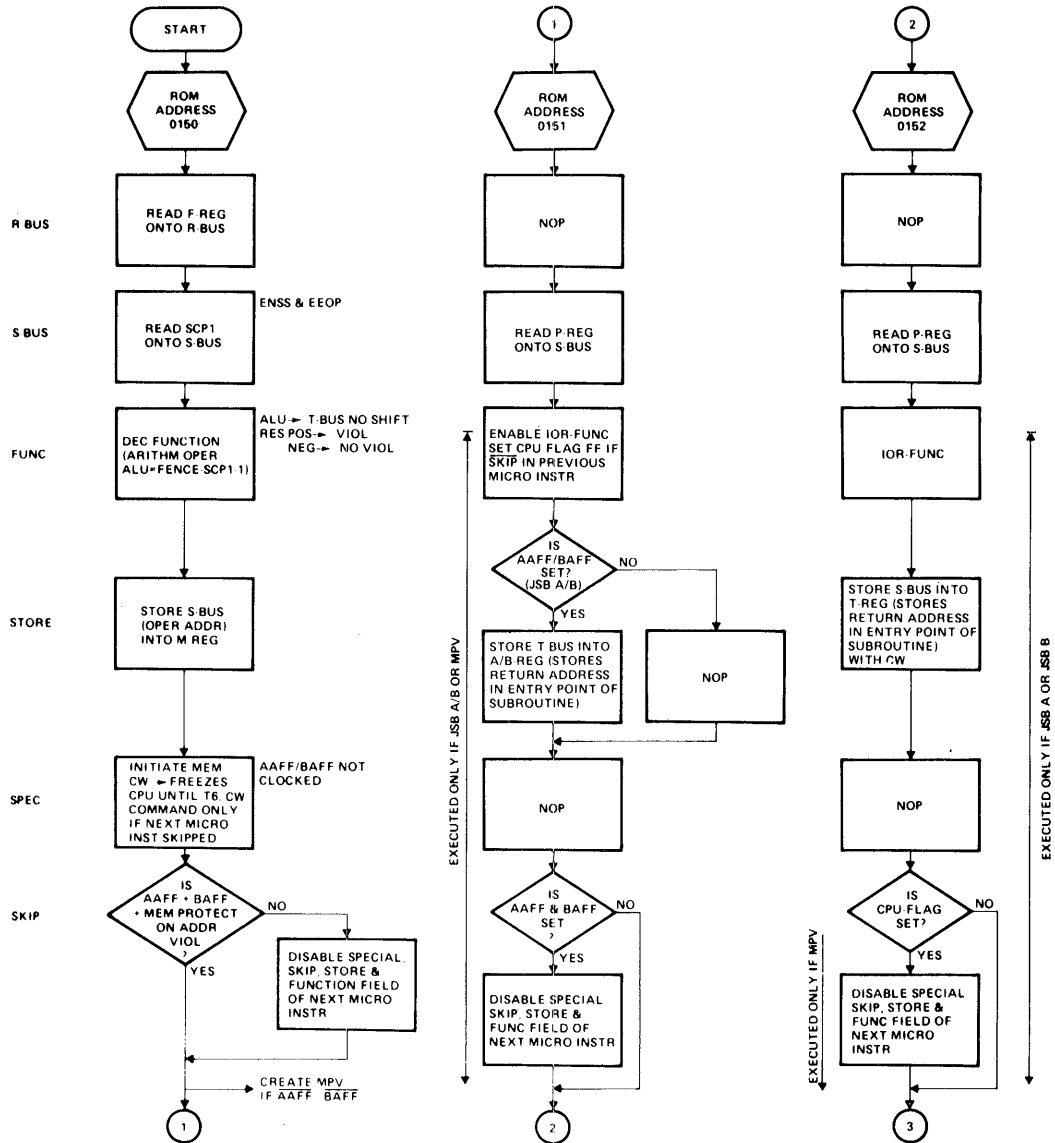
FLOW CHART OF JSB INSTRUCTION

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R BUS 23 21	S BUS 20 17	FUNCTION 16 12	STORE 11 8	SPECIAL 7 4	SKIP 3 0
0150	35460712	JSB	F	S1	DEC	M	CW	NMPV
0151	77346373		-	P	SFLG	AAB		AAB
0152	77371366		-	P	IOR	T		FLG
0153	75514375		-	S1	INC	P		EOP
0154	77777777		-	-	IOR	-	-	-

R S FUNC ST SP SK
011 101 1 00 110 000 1 11 00 1 010

R S FUNC ST SP SK
111 111 0 11 100 110 0 11 11 1 011

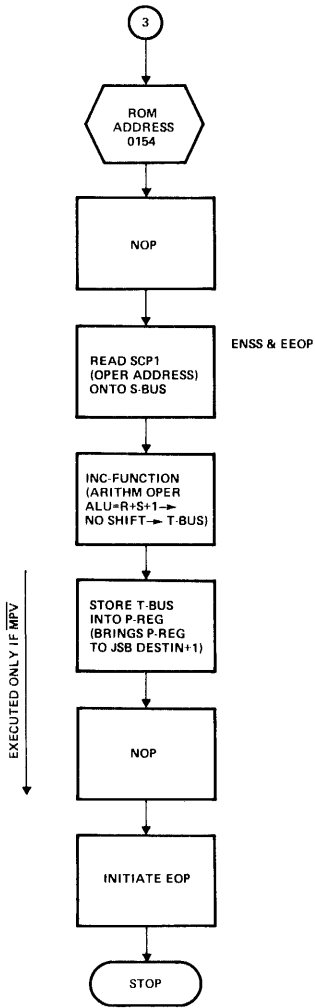
R S FUNC ST SP SK
111 111 0 11 111 001 0 11 11 0 110



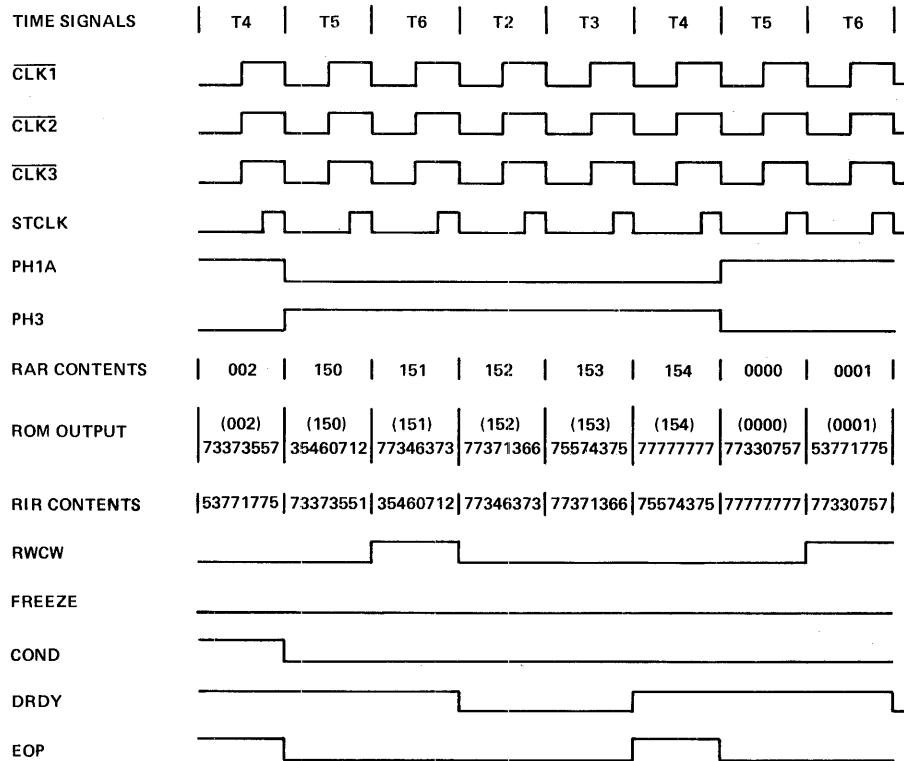
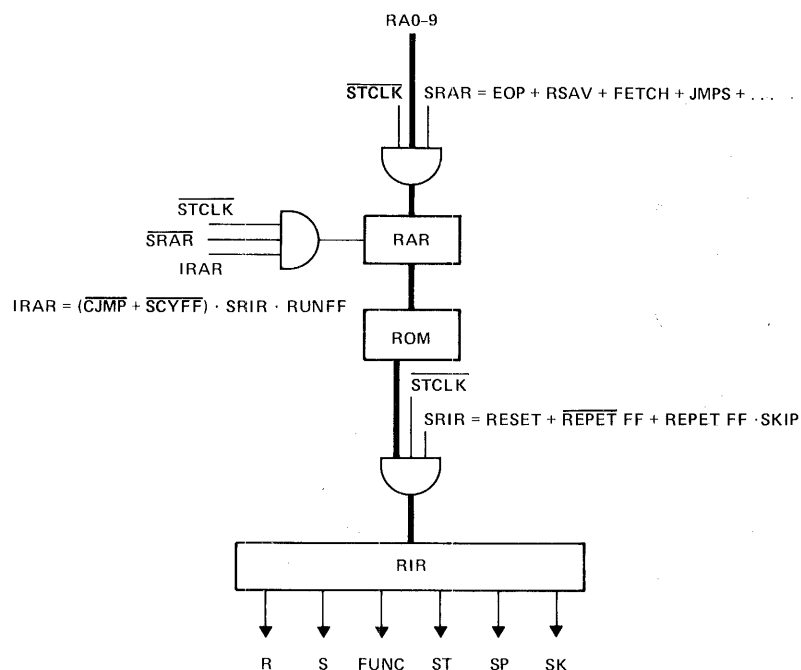
FLOW CHART OF JSB – INSTRUCTION (CONTINUE)

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS 23-21	S-BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0
0150	35460712	JSB	F	S1	DEC	M	CW	NMPV
0151	77346373		-	P	SFLG	AAB	-	AAB
0152	77371366		-	P	IOR	T	-	FLG
0153	75514375		-	S1	INC	P	-	EOP
0154	77777777		-	-	IOR	-	-	-

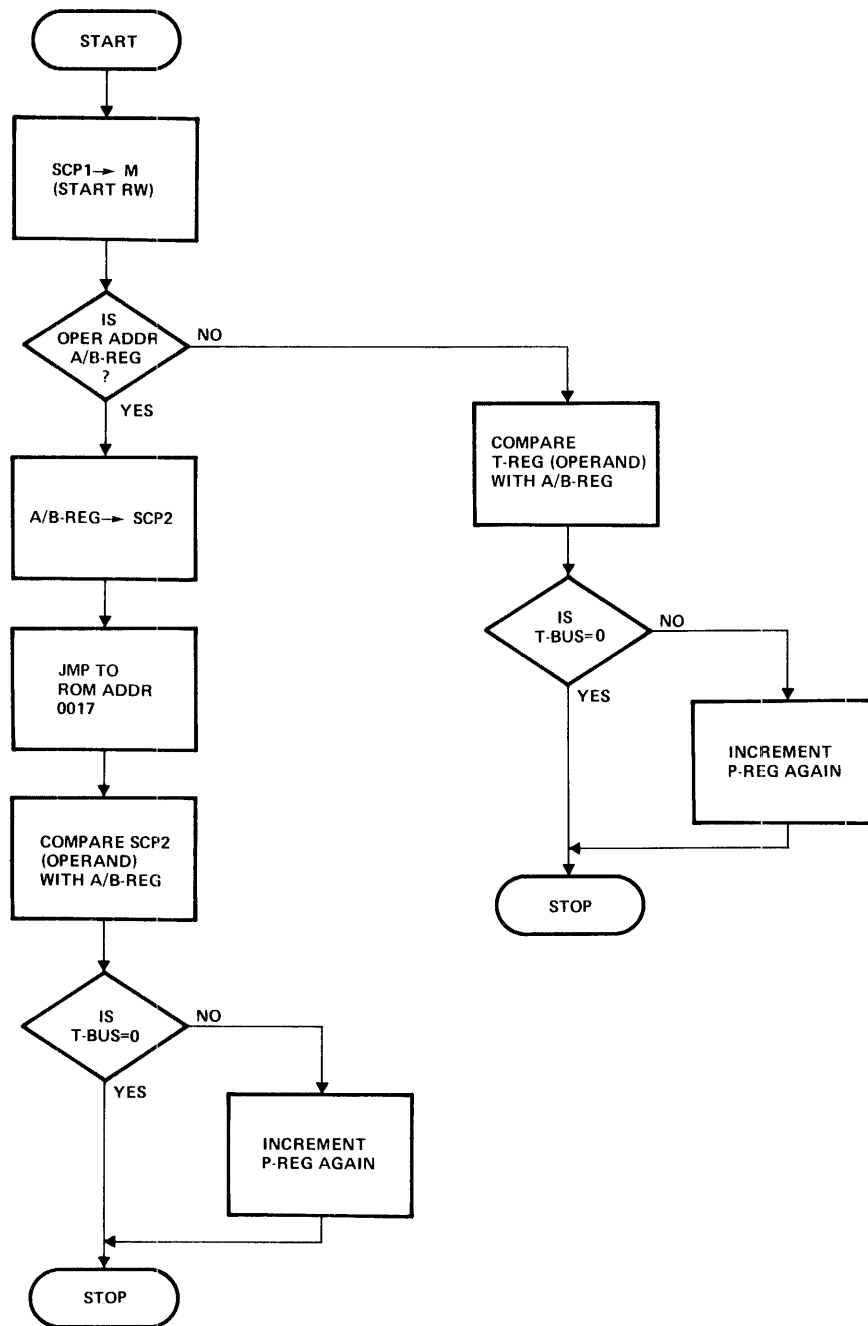
R	S	FUNC	ST	SP	SK
111	101 1	01 001	100 0	11 11	1 101



TIMING DIAGRAM OF PH3 JSB-INSTRUCTION



COMPARE INSTRUCTION



ROM JMP, JSB OR CJMP EXECUTIONS

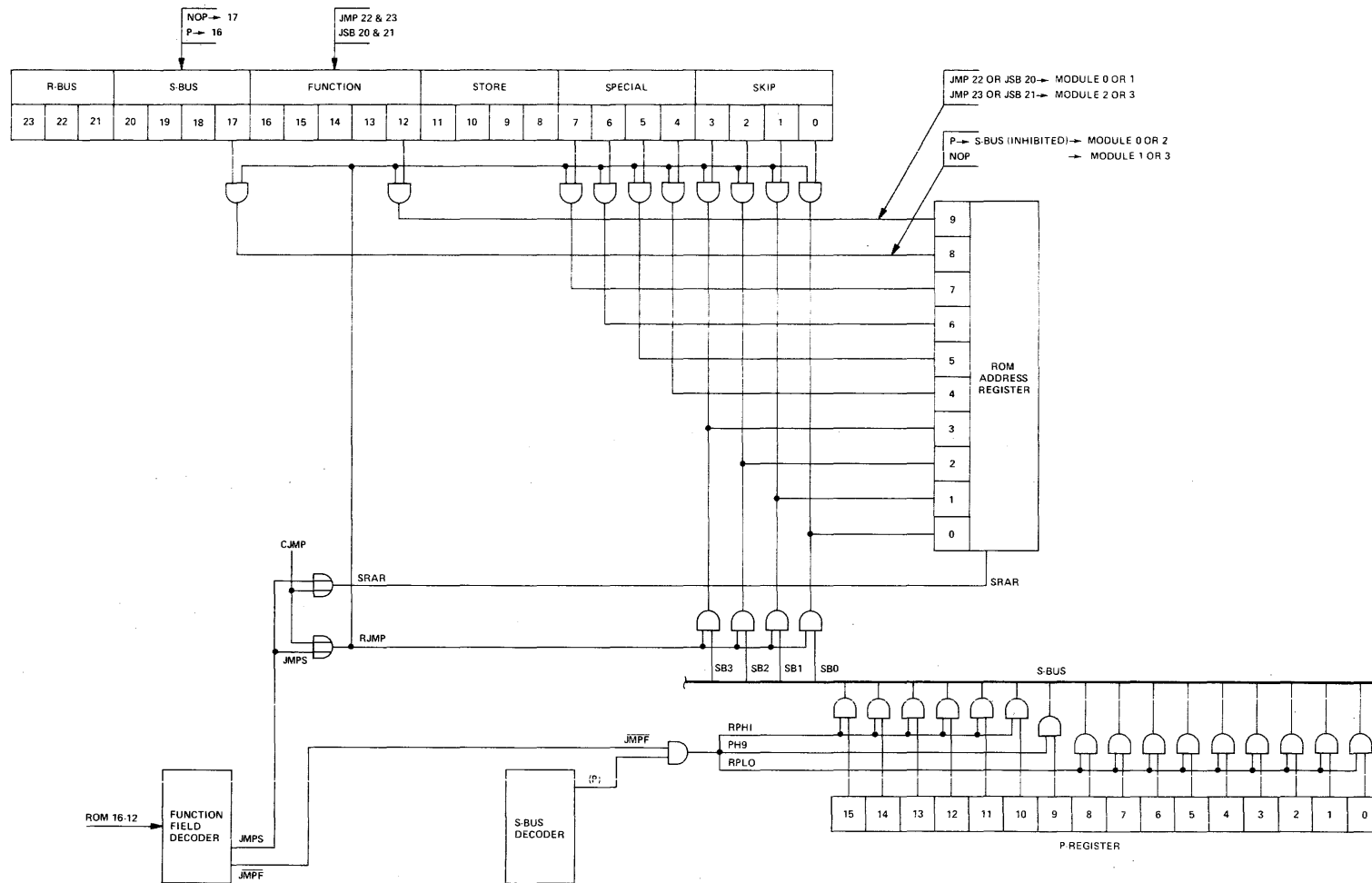
WHEN MICROINSTRUCTIONS CONTAINING JMP, JSB OR CJMP ARE EXECUTED ENRM GOES LOW AND THE OUTPUT FROM ROM IS ALL 1'S (NOP). AT THE FOLLOWING STCLK (END OF JMP, JSB OR CJMP-MICROORDER) TWO MAJOR OPERATIONS ARE PERFORMED:

1. THE NOP IS STORED INTO THE RIR AND THEREFORE NOP'S THE MICRO INSTRUCTION FOLLOWING THE JMP, JSB OR CJMP.
2. THE JMP, JSB OR CJMP DESTINATION ADDRESS (RIR 17, 12, 7-0) IS TRANSFERED TO THE RAR.

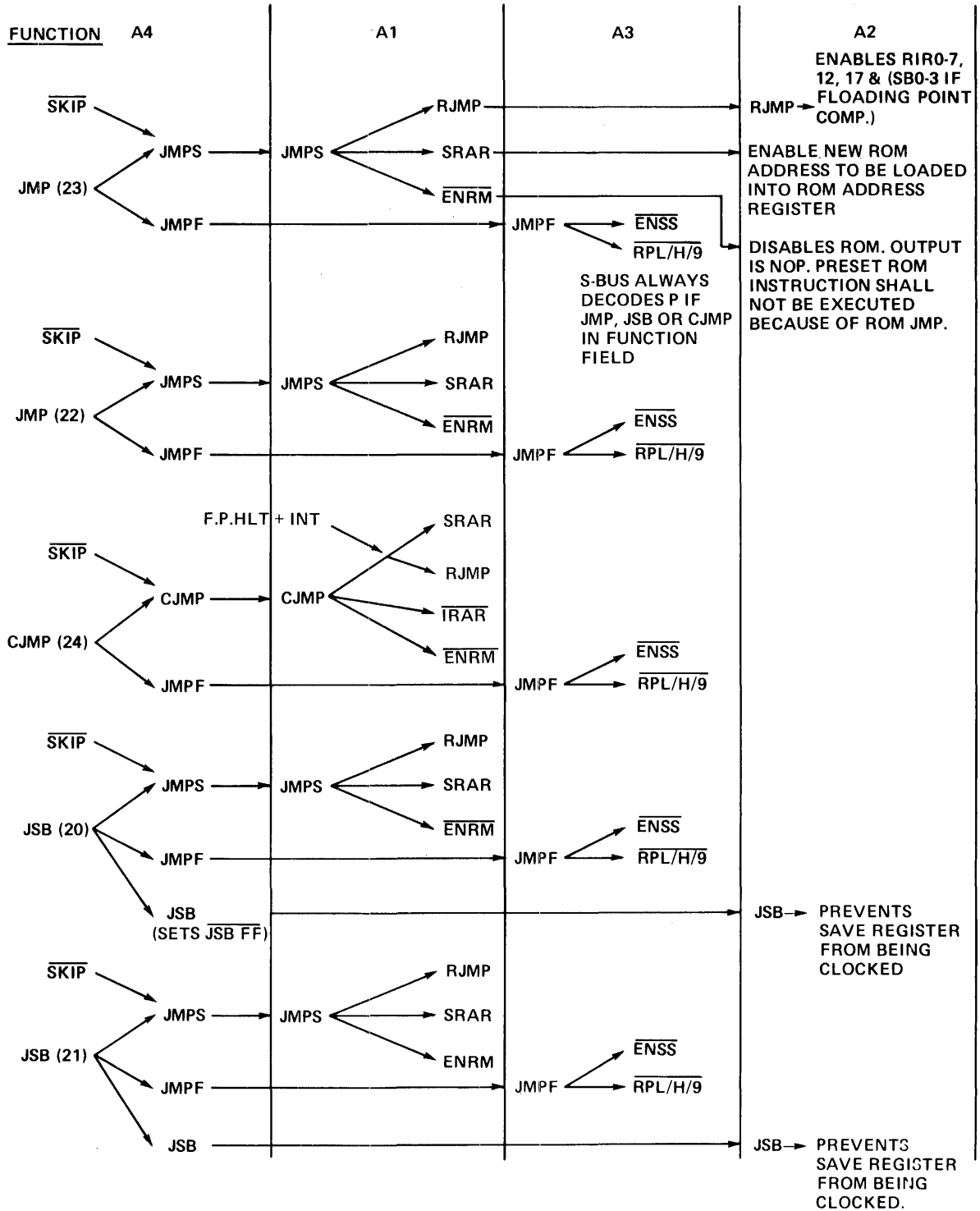
RIR 12	—————→	RAR 9
RIR 17	—————→	RAR 8
RIR 7-0	—————→	RAR 7-0

200 ns LATER (WITH THE NEXT STCLK) THE ROM CONTENTS OF THE ADDRESSED ROM LOCATION (IN PARAGRAPH 2) IS CLOCKED INTO THE RIR.

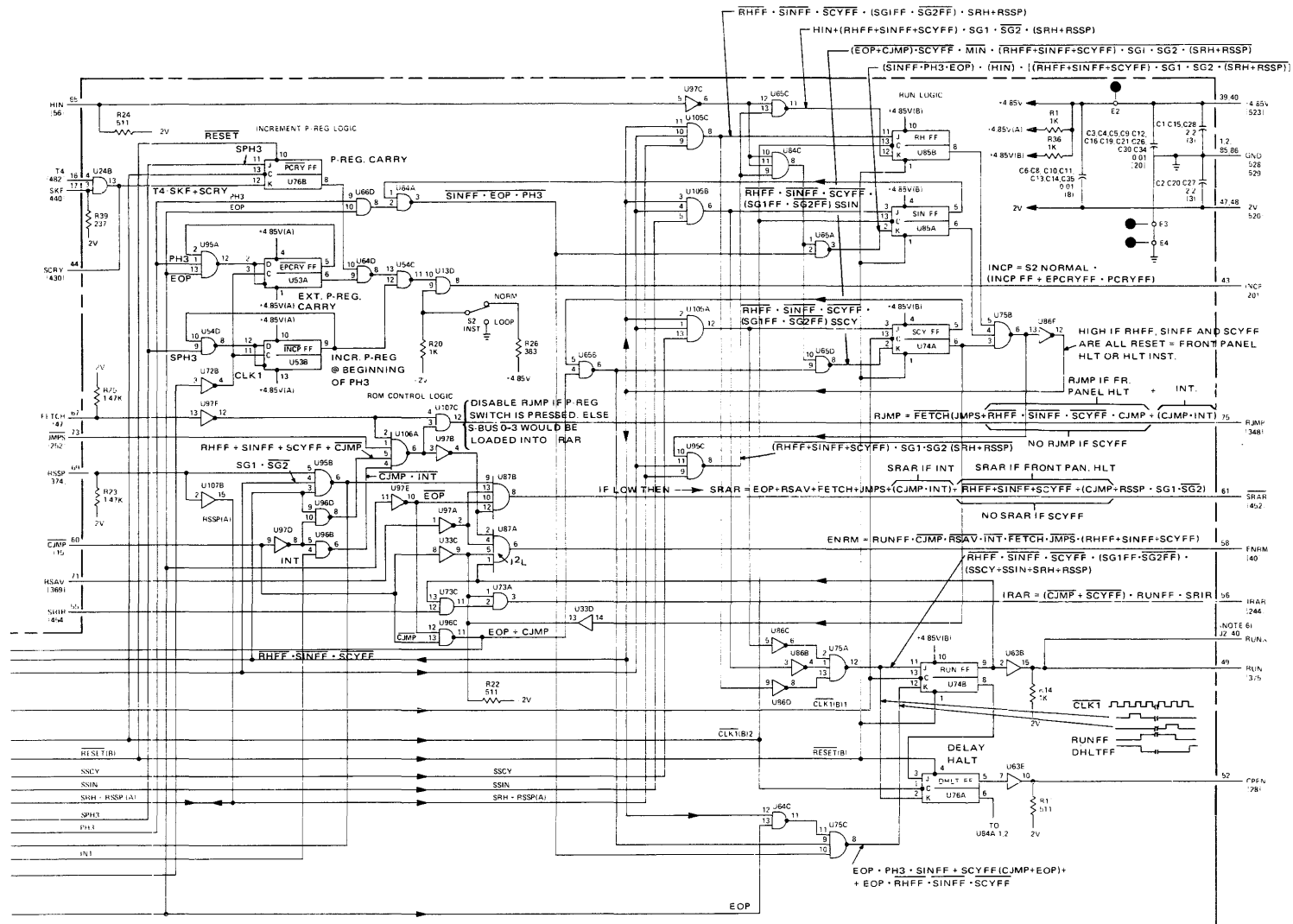
ROM JMP AND JSB CONTROL SIGNALS



HARDWARE CONTROL SIGNALS CREATED BY ROM JMP, JSB AND CJMP

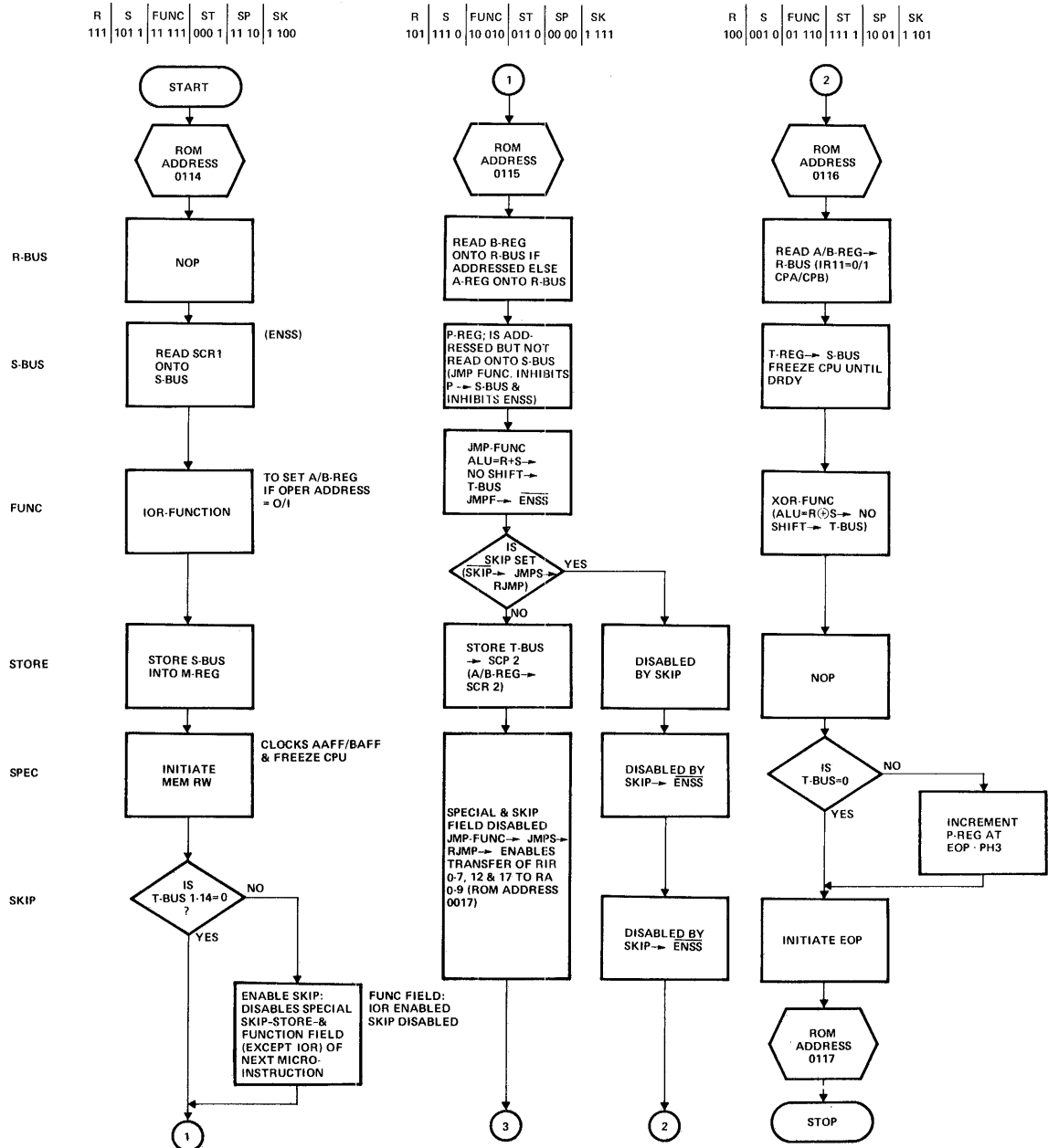


ROM CONTROL LOGIC



FLOW CHART OF CP* - INSTRUCTION

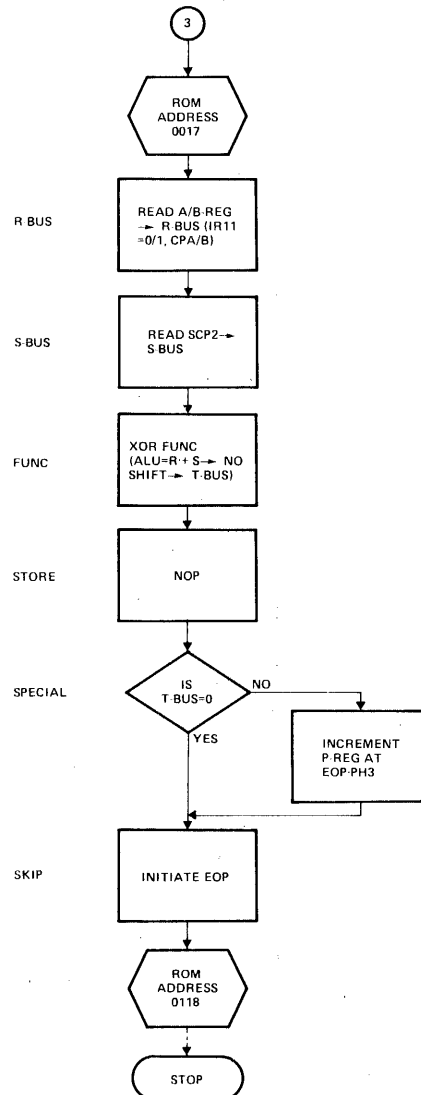
ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS 23-21	S-BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0
0114	75770754	CP*	—	S1	IOR	M	RW	NAAB
0115	57223017		AAB	P	JMP	S2	XX	—
0116	41167635		CAB	T	XOR	—	ECYN	EOP
0117	77777777		—	—	IOR	—	—	—



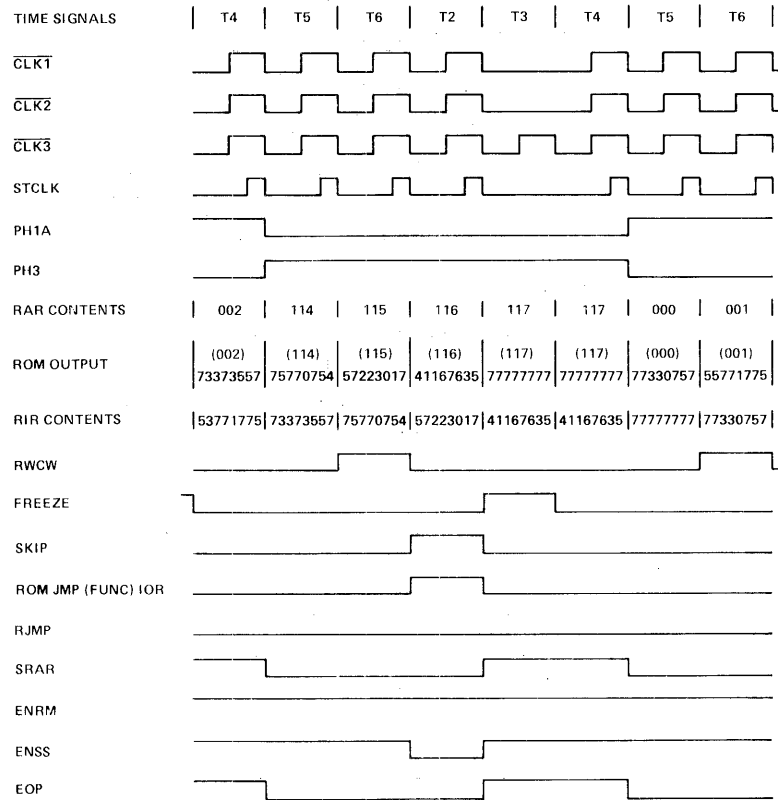
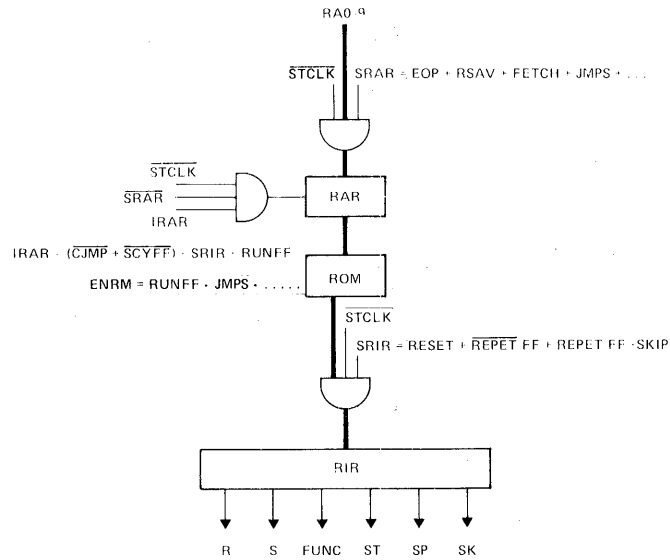
FLOW CHART OF CP* - INSTRUCTION (CONTINUE)

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R BUS 23-21	S BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0
0017	45167635	XX	CAB	S2	XOR	--	ECYN	EOP
0020	77777777		--	--	IOR	--	--	--

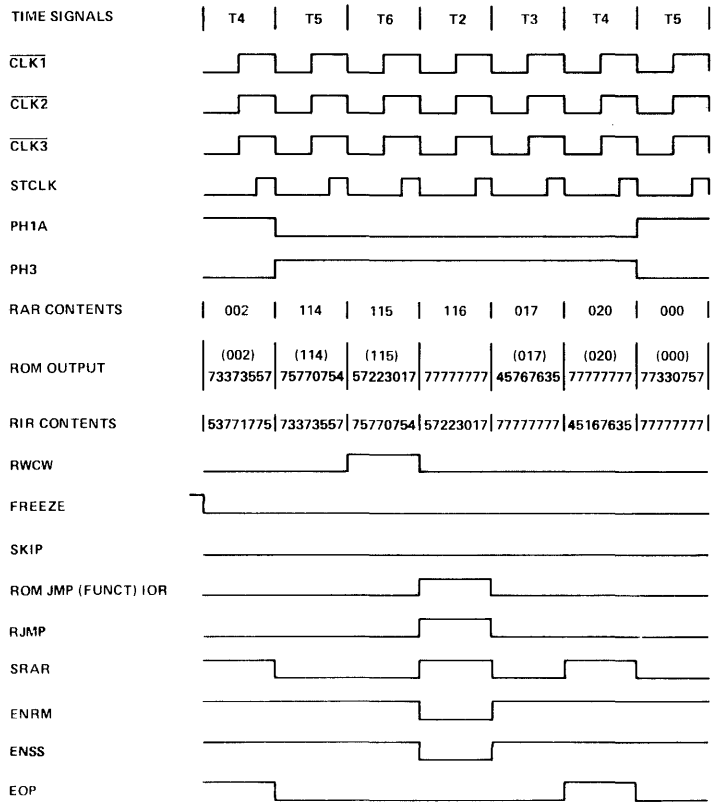
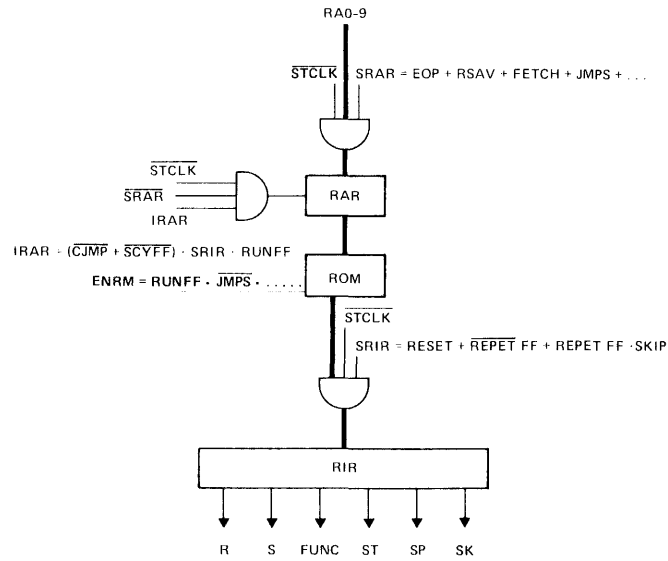
R S FUNC ST SP SK
100 101 0 01 110 111 1 10 01 1 101



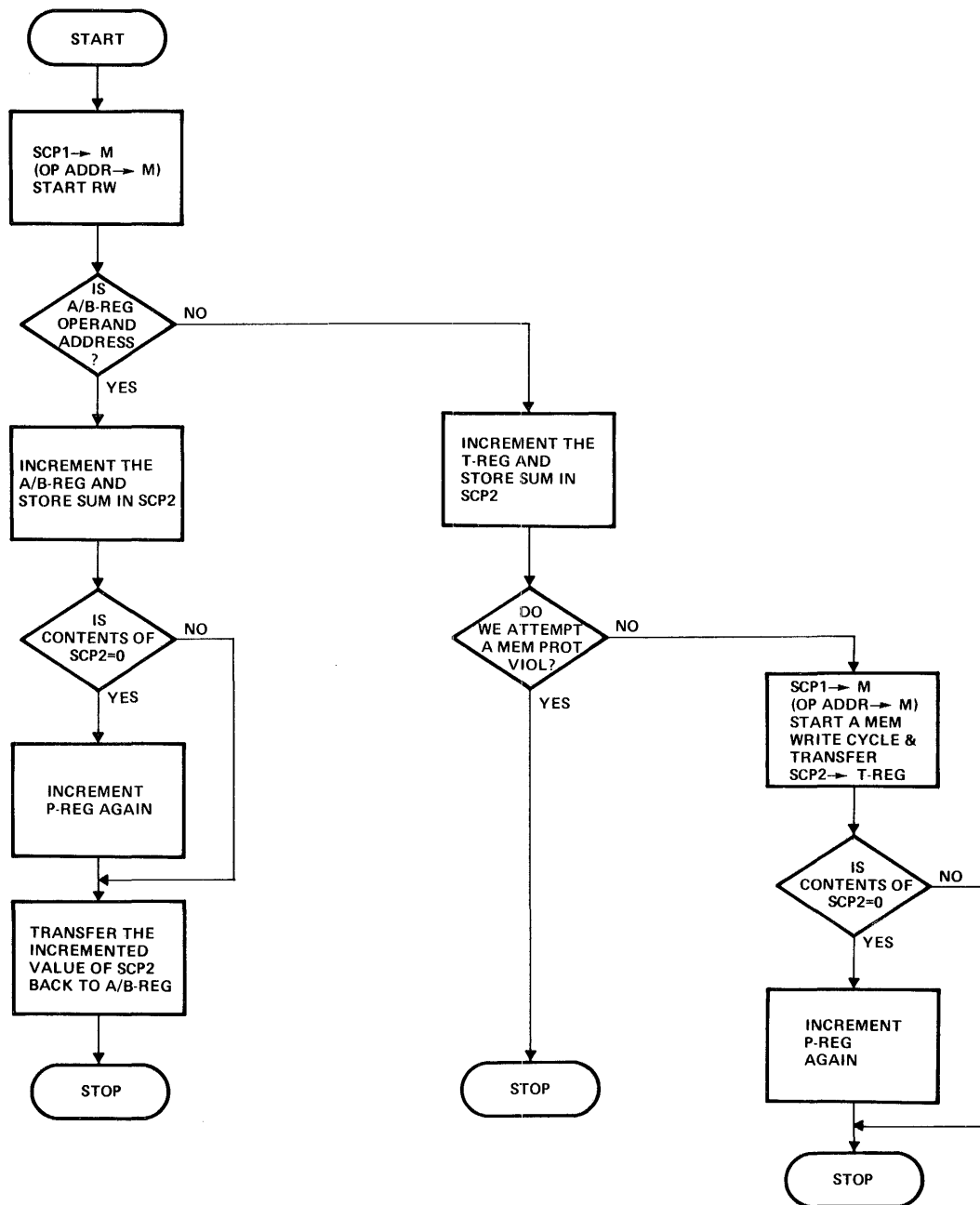
TIMING DIAGRAM OF PH3, CP*-INSTRUCTION
(OPERAND ADDRESS IS NOT IN A/B-REGISTER)



**TIMING DIAGRAM OF PH3, CP*-INSTRUCTION
(OPERAND ADDRESS IS IN A/B-REGISTER)**



ISZ INSTRUCTION



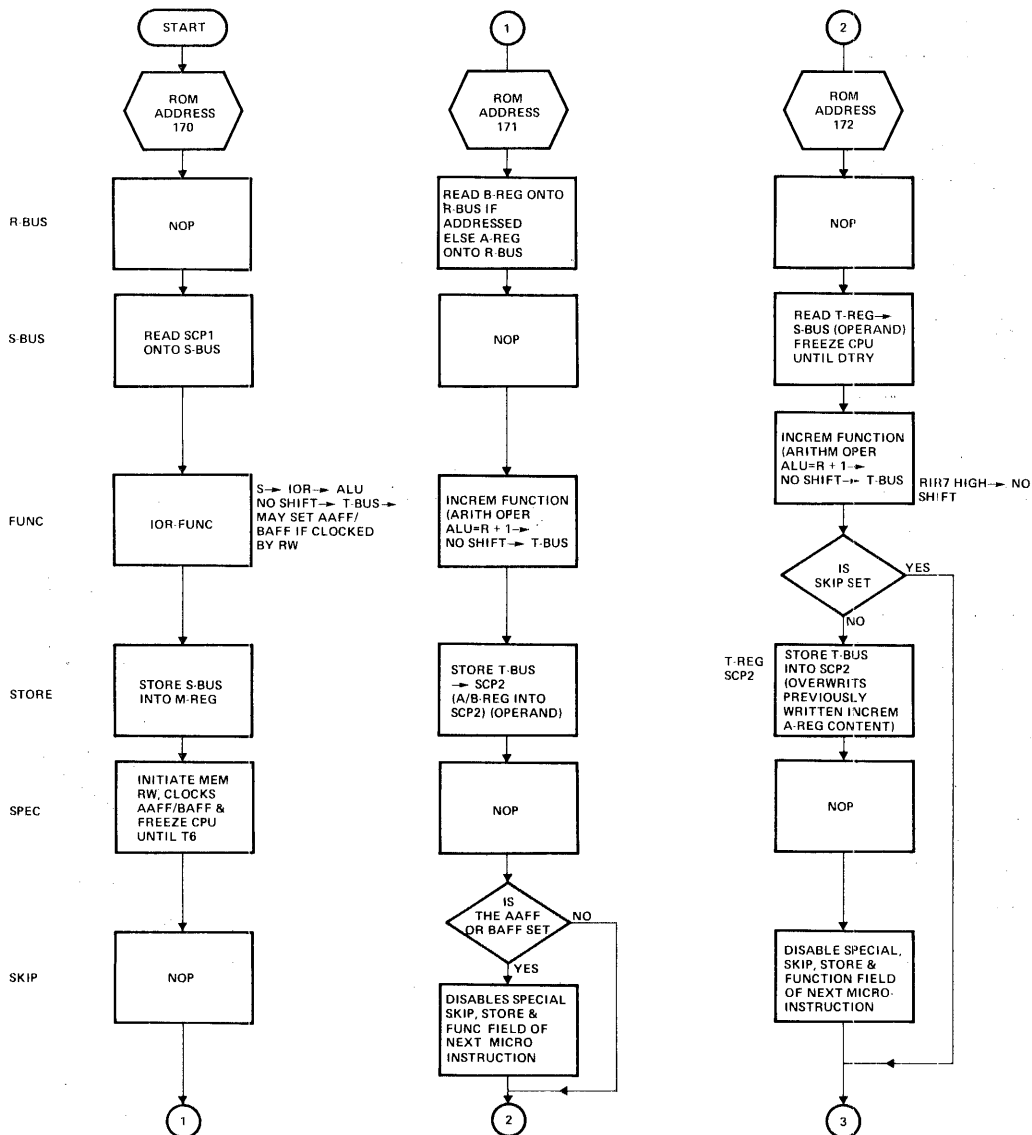
FLOW CHART OF ISZ — INSTRUCTION

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS 23-21	S-BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0
0170	75770757	ISZ	—	S1	IOR	M	RW	—
0171	57513373		AAB	—	INC	S2	—	AAB
0172	71113376		—	T	INC	S2	—	UNC
0173	75377616		—	S2	IOR	—	ECYZ	UNC
0174	35460712		F	S1	DEC	M	CW	NMPV
0175	75376376		—	S2	IOR	AAB	—	UNC
0176	75371215		—	S2	IOR	T	ECYZ	EOP
0177	77777777		—	—	IOR	—	—	—
0200	77777775		—	—	IOR	—	—	EOP

R S FUNC ST SP SK
111 101 1 11 111 000 1 11 10 1 110

R S FUNC ST SP SK
101 111 1 01 001 011 0 11 11 1 011

R S FUNC ST SP SK
111 001 0 01 001 011 0 11 11 1 110



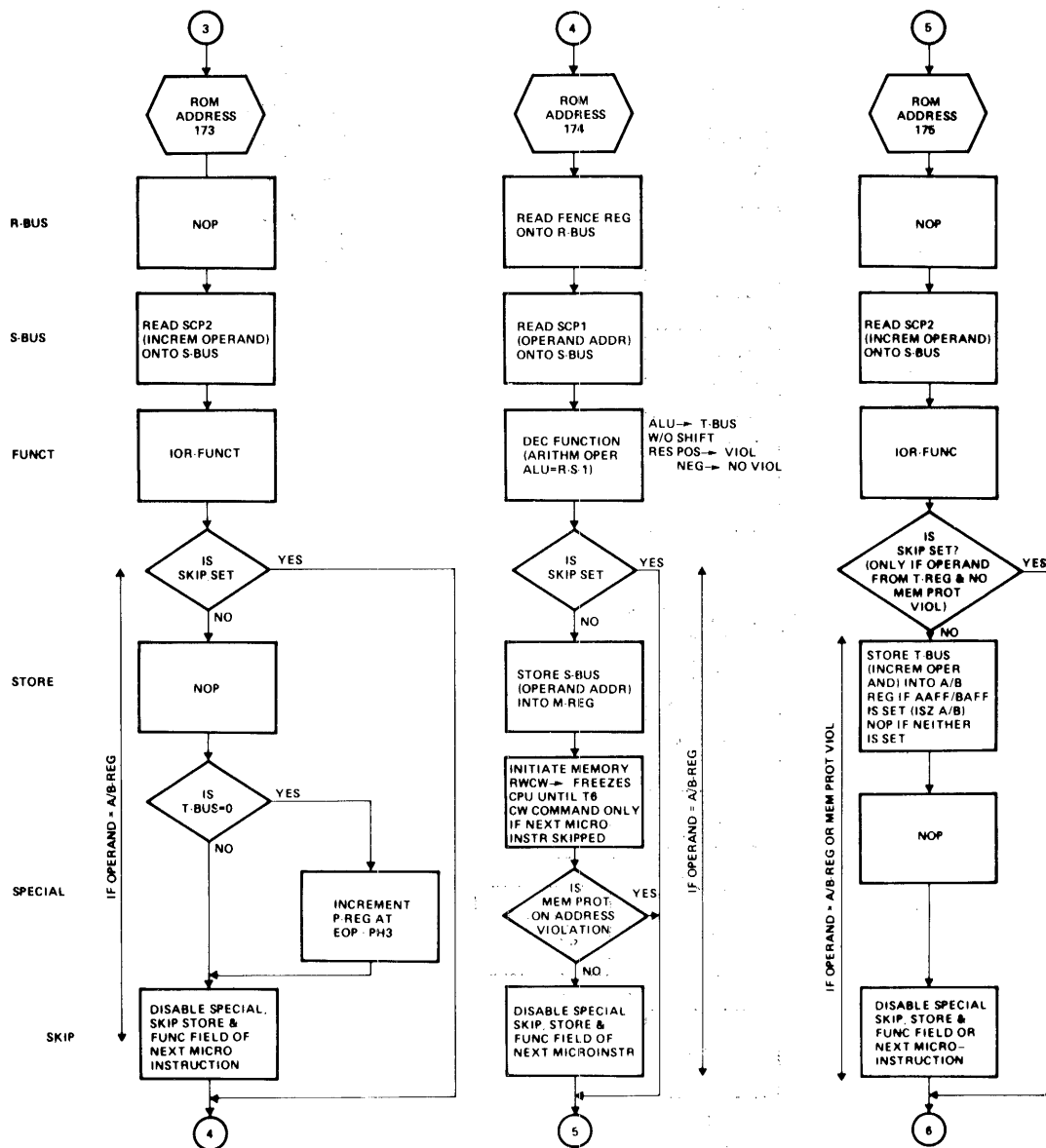
FLOW CHART OF ISZ – INSTRUCTION (CONTINUE 1)

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS 23 21	S-BUS 20 17	FUNCTION 16 12	STORE 11 8	SPECIAL 7 4	SKIP 3 0
0170	75770757	ISZ	—	S1	IOR	M	RW	—
0171	57513373		AAB	—	INC	S2	—	AAB
0172	71113376		—	T	INC	S2	—	UNC
0173	75377616	—	—	S2	IOR	—	ECYZ	UNC
0174	35460712		F	S1	DEC	M	CW	NMPV
0175	75376376		—	S2	IOR	AAB	—	UNC
0176	75371215	—	—	S2	IOR	T	ECYZ	EOP
0177	77777777		—	—	IOR	—	—	—
0200	77777775		—	—	IOR	—	—	EOP

R S FUNC ST SP SK
111 101 0 11 111 111 1 10 00 1 110

R S FUNC ST SP SK
011 101 1 00 110 000 1 11 00 1 010

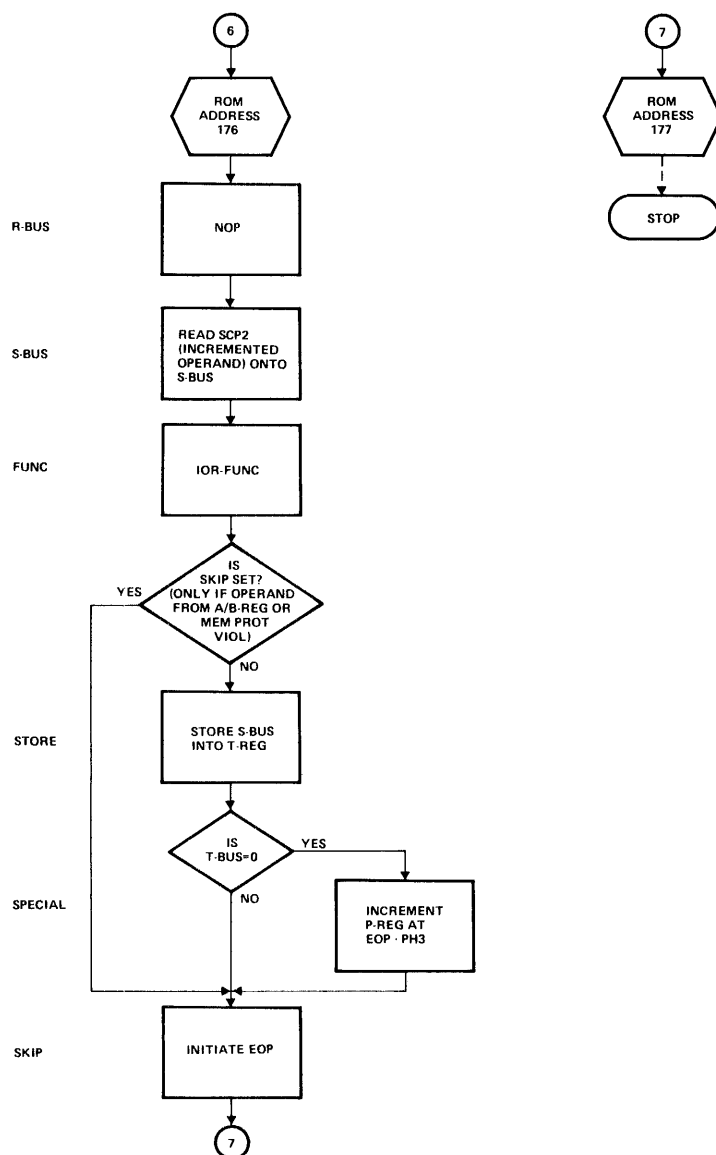
R S FUNC ST SP SK
111 101 0 11 111 110 0 11 11 1 110



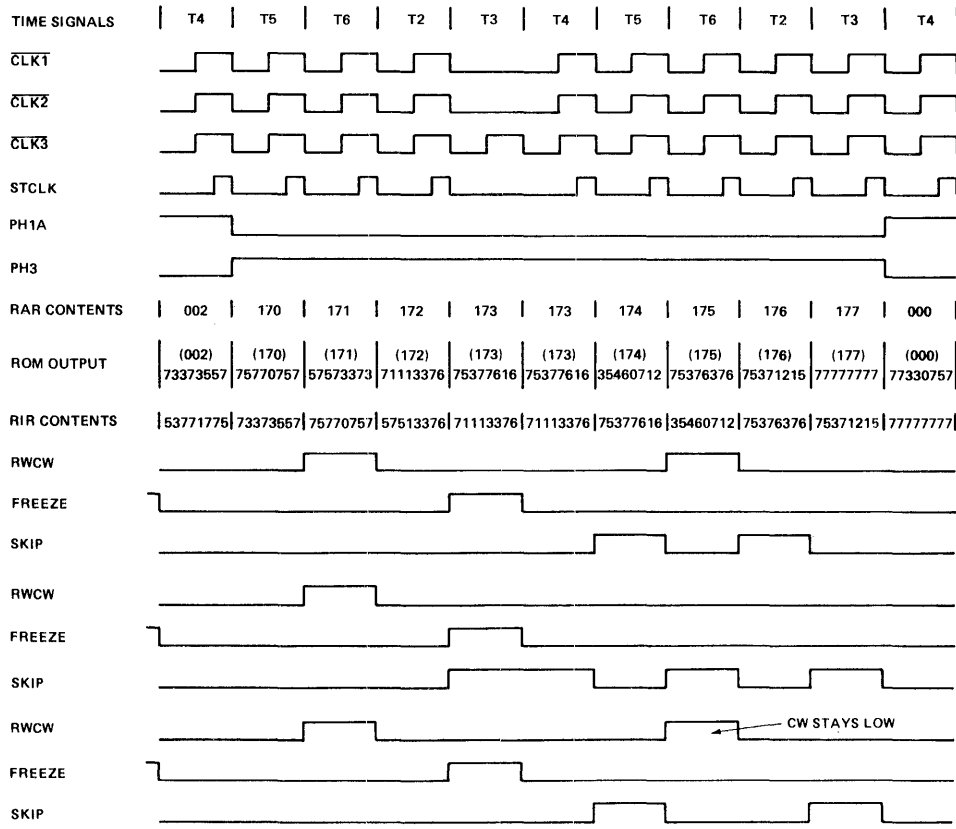
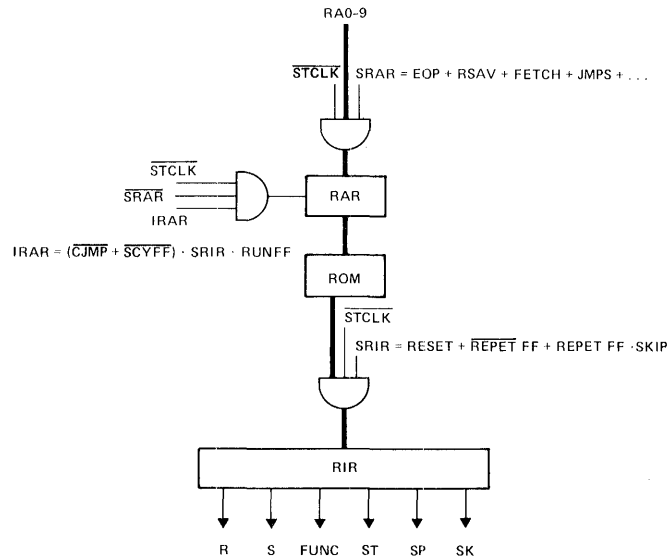
FLOW CHART OF ISZ – INSTRUCTION (CONTINUE 2)

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS 23-21	S-BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0
0170	75770757	ISZ	—	S1	IOR	M	RW	—
0171	57513373		AAB	—	INC	S2	—	AAB
0172	71113376		—	T	INC	S2	—	UNC
0173	75377616		—	S2	IOR	—	ECYZ	UNC
0174	35460712		F	S1	DEC	M	CW	NMPV
0175	75376376		—	S2	IOR	AAB	—	UNC
0176	75371215		—	S2	IOR	T	ECYZ	EOP
0177	77777777		—	—	IOR	—	—	—
0200	77777775		—	—	IOR	—	—	EOP

R | S | FUNC | ST | SP | SK
 111 | 101 0 | 11 111 | 001 0 | 1000 | 1 101



TIMING DIAGRAM OF PH3, ISZ INSTRUCTION



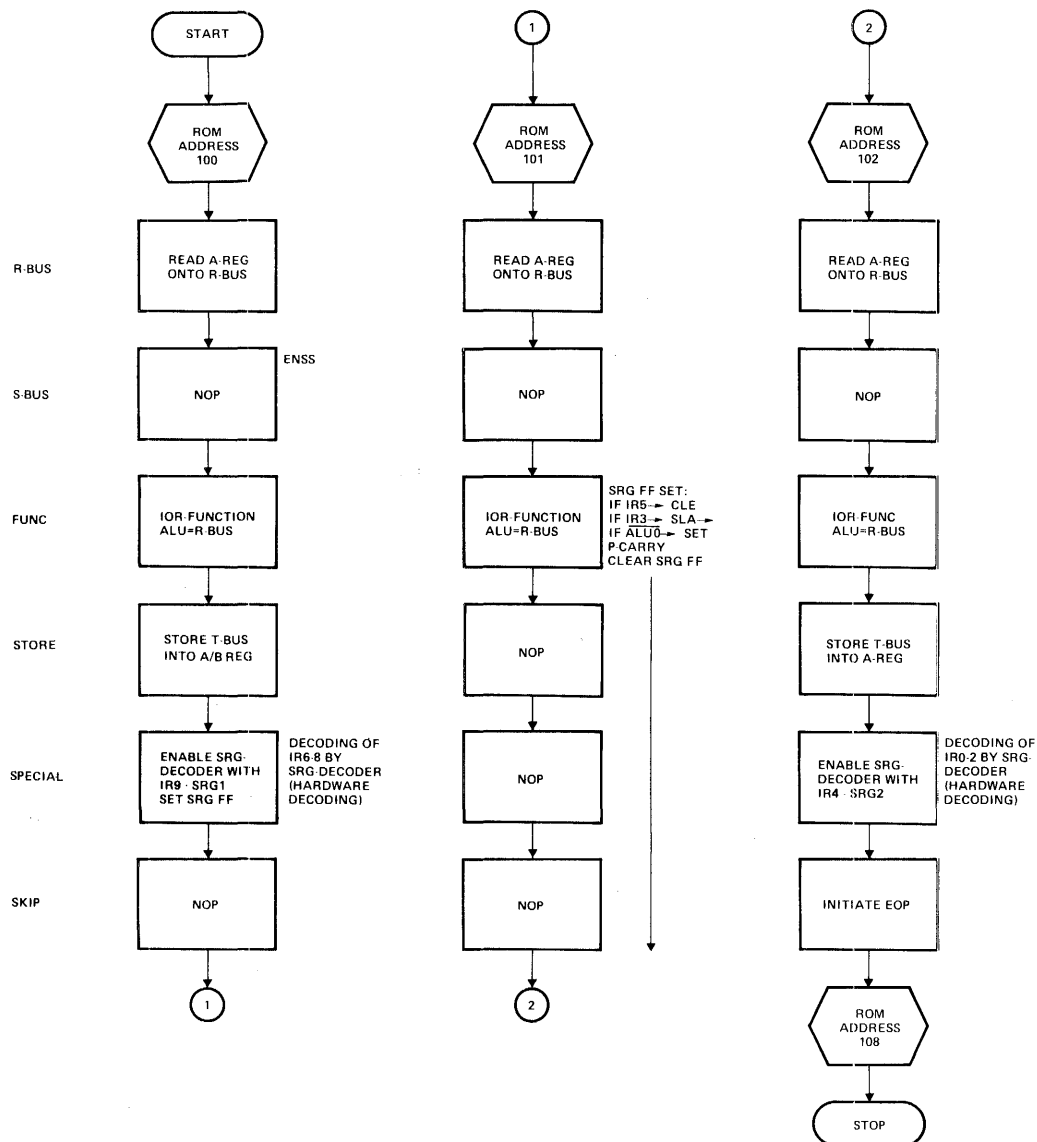
FLOW CHART OF SRG – INSTRUCTIONS

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS 23-21	S-BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0
0100	07777117	SRGA	A	—	IOR	A	SRG1	—
0101	07777777		A	—	IOR	—	—	—
0102	07777135		A	—	IOR	A	SRG2	EOP
0103	77777777		—	—	IOR	—	—	—

R | S | FUNC | ST | SP | SK
000 | 111 1 | 11 111 | 111 0 | 01 00 | 1 111

R | S | FUNC | ST | SP | SK
000 | 111 1 | 11 111 | 111 1 | 11 11 | 1 111

R | S | FUNC | ST | SP | SK
000 | 111 1 | 11 111 | 111 0 | 01 01 | 1 101



HARDWARE SIGNALS TO IMPLEMENT SRG-INSTRUCTIONS

I-REGISTER BIT 9 8 7 6 4 2 1 0	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
INSTRUCTION	*LS	*RS	R*L	R*R	*LR	ER*	EL*	*LF
SL1	X		X		X		X	
SR1		X		X		X		X
SL4								X
TBS1		X		X		X		X
TBS2	X		X		X		X	X
ALX14→ (ALU15→ TB15) (ALU14→ TB15)	X		X				X	
ALX16→ (ALU15→ TB15) (ALU0→ TB15) (EXT→ TB15)		X		X		X		
LSI → (ALU15→ TB0) (EXT→ TB0)			X				X	
SRG1/2→ (ALU15→ EXT) (ALU0→ EXT)						X	X	

ASG-INSTRUCTIONS

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS	S-BUS	FUNCTION	STORE	SPECIAL	SKIP
0021	47777657	ASGD	CAB	—	IOR	—	ASG1	—
0022	47525675		CAB	—	ADDO	CAB	ASG2	EOP
0023	77777777		—	—	IOR	—	—	—
0025	77775657	ASGA	—	—	IOR	CAB	ASG1	—
0026	47525675		CAB	—	ADDO	CAB	ASG2	EOP
0027	77777777		—	—	IOR	—	—	—
0031	47555657	ASGB	CAB	—	NOR	CAB	ASG1	—
0032	47525675		CAB	—	ADDO	CAB	ASG2	EOP
0033	77777777		—	—	IOR	—	—	—
0035	77555657	ASGC	—	—	NOR	CAB	ASG1	—
0036	47525675		CAB	—	ADDO	CAB	ASG2	EOP
0037	77777777		—	—	IOR	—	—	—

A/B-REG (DEPENDING ON IR11)
→ R-BUS→ ALU-BUS FOR POSSIBLE
EXECUTION OF SL*/SS* - INSTRUCTIONS

CLA/B (DEPENDING ON IR11)→ ALU-BUS
FOR POSSIBLE EXECUTION OF SL*/SS*
- INSTRUCTIONS

CMA/B (DEPENDING ON IR11)→ ALU-BUS
FOR POSSIBLE EXECUTION OF SL*/SS*
- INSTRUCTIONS

CCA/B (DEPENDING ON IR11)→ ALU-BUS
FOR POSSIBLE EXECUTION OF SL*/SS*
- INSTRUCTIONS

INA/B & SZA/B IF PROGRAMMED

FUNCTION OF ASG1 & ASG2

THE MICROORDERS ASG1 & ASG2 ENABLE THE EXECUTION OF THE FOLLOWING ASG-INSTRUCTIONS:

			<u>IC-NR ON A6</u>
ASG1:	SEZ	SEZ · RSS	U17A & B
	CLE		U73C & U97B
	CME		U97C, U91C, U73B & U63B
	CCE		U97A & 63C
	RSS		} U54 DECODER
	SL *	SL * · RSS	
	SS*	SS* · RSS	
	SS* · SL *	SS* · SL * · RSS	
ASG2:	IN*		U107A
	SZ*	SZ* · RSS	U61B, C & D, U91A, U32B & D

DECODING ASG INSTRUCTIONS

- THE MAPPER, WHEN DECODING AN ASG-INSTRUCTION WILL OUTPUT ROM ADDRESS 21, 25, 31 OR 35 DEPENDING ON INSTRUCTION REG. BIT 8 & 9 $\therefore$ CL*, CM* OR CC* ARE EXECUTED UNDER ROM CONTROL

$\overline{\text{IR9}} \cdot \overline{\text{IR8}}$	$\rightarrow$	$\overline{\text{CL}}^* \cdot \overline{\text{CM}}^* \cdot \overline{\text{CC}}^*$	$\rightarrow$	ROM ADDRESS 21
$\overline{\text{IR9}} \cdot \text{IR8}$	$\rightarrow$	CL*	$\rightarrow$	" " 25
$\text{IR9} \cdot \overline{\text{IR8}}$	$\rightarrow$	CM*	$\rightarrow$	" " 31
$\text{IR9} \cdot \text{IR8}$	$\rightarrow$	CC*	$\rightarrow$	" " 35

- ASG1 ENABLES DECODING OF CLE, CME, CCE, SEZ, SS*, SL* & RSS BY HARDWARE
- ASG2 ENABLES DECODING OF IN* & SZ* BY HARDWARE

EXECUTION PRIORITY OF CONCATENATED ALTER-SKIP GROUP INSTRUCTIONS

1.) EXTEND REGISTER

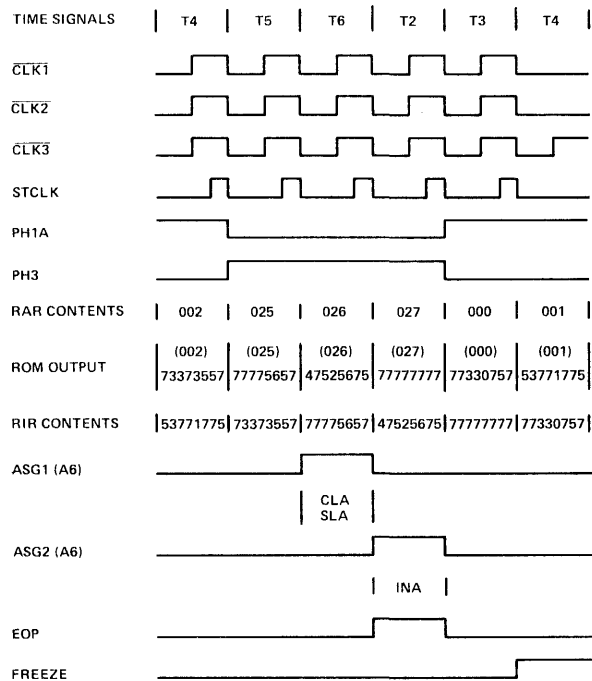
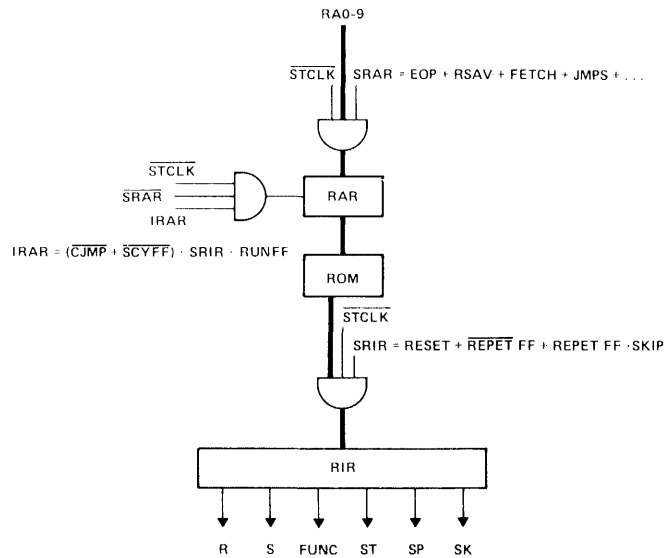
SEZ OR SEZ · RSS IS EXECUTED BEFORE CLE, CME OR CCE

2.) A— OR B—REGISTER

THE FOLLOWING SEQUENCE APPLIES TO THE A— OR B—REGISTER
RELATED ASG-INSTRUCTIONS:

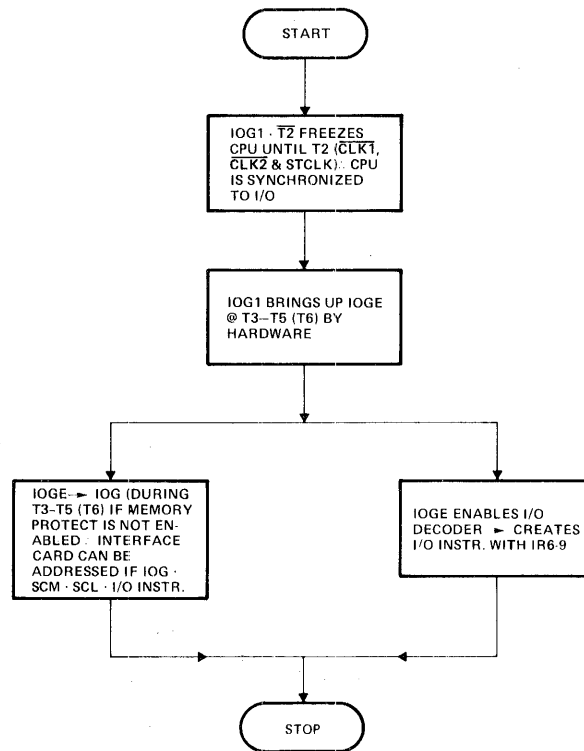
1. CL* OR CM* OR CC*
2. SS* OR SL* (RSS)
3. IN*
4. SZ* (RSS)

TIMING DIAGRAM OF PH3; CLA, SLA, INA – INSTRUCTION

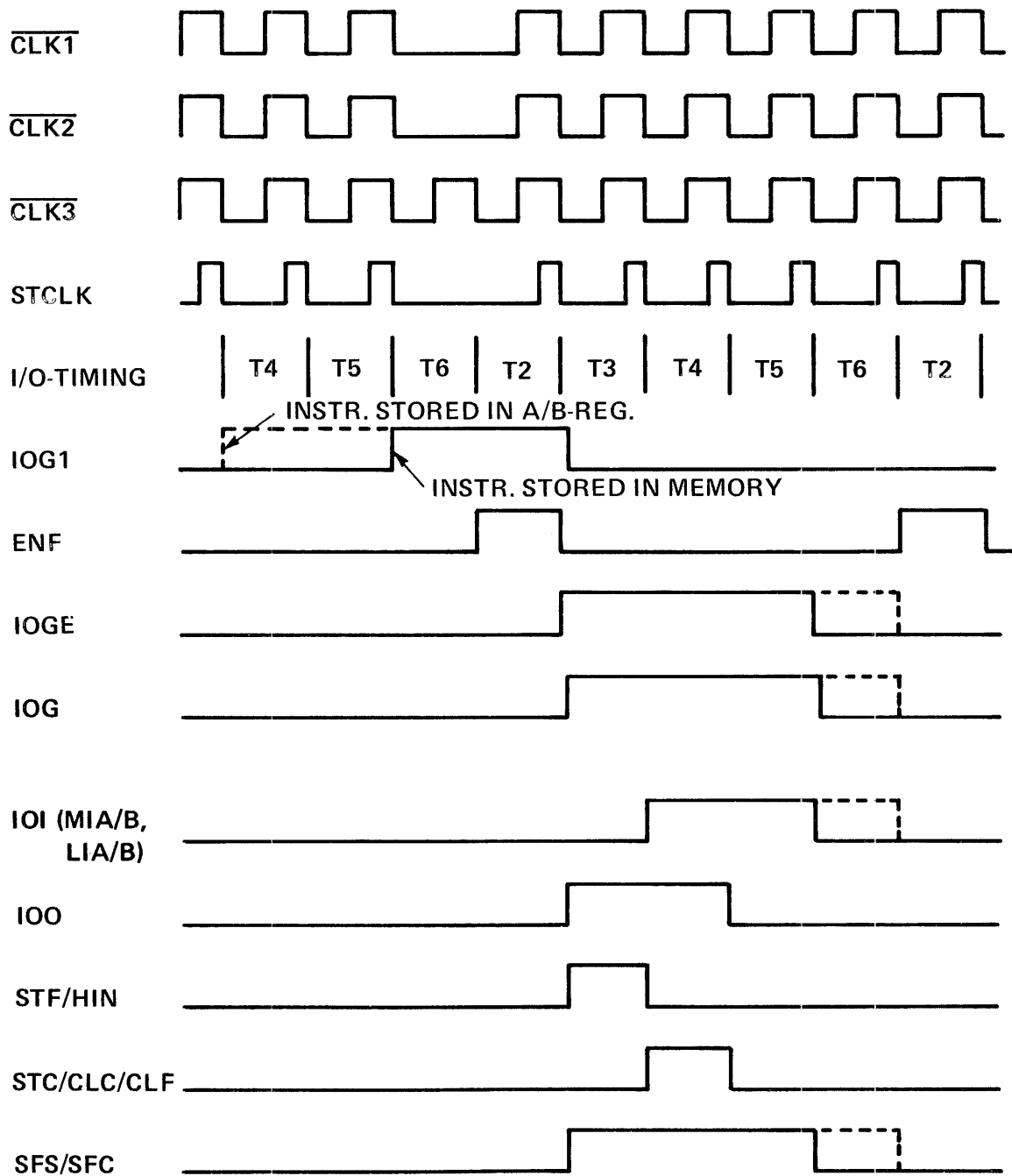


STC, CLC, STF, CLF, SFS, SFC & HLT – INSTRUCTIONS

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS	S-BUS	FUNCTION	STORE	SPECIAL	SKIP
0040	77777737	FLAG	—	—	IOR	—	IOG1	—
0041	77777777		—	—	IOR	—	—	—
0042	77777775		—	—	IOR	—	—	EOP
0043	77777777		—	—	IOR	—	—	—
0074	77777737	CTRL	—	—	IOR	—	IOG1	—
0075	77777777		—	—	IOR	—	—	—
0076	77777775		—	—	IOR	—	—	EOP
0077	77777777		—	—	IOR	—	—	—

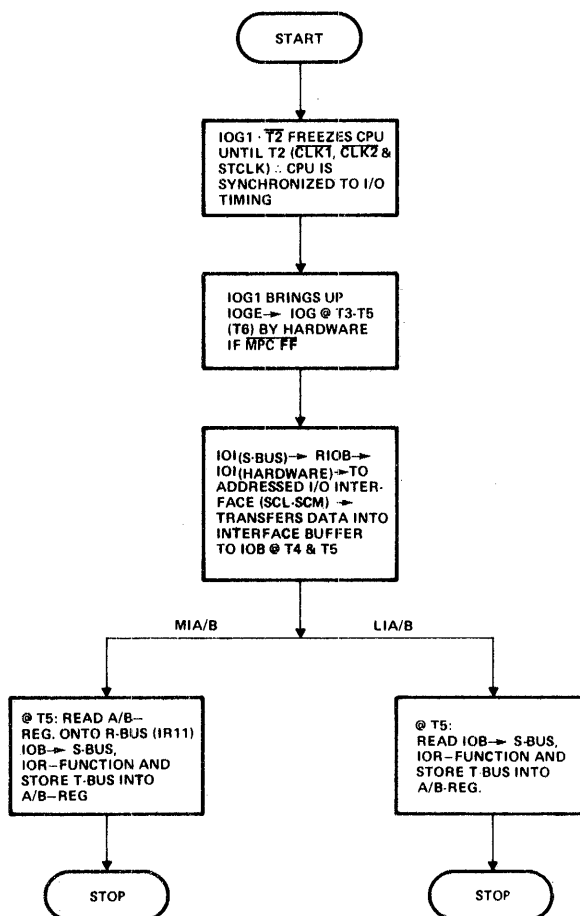


TIMING DIAGRAM FOR I/O – INSTRUCTIONS



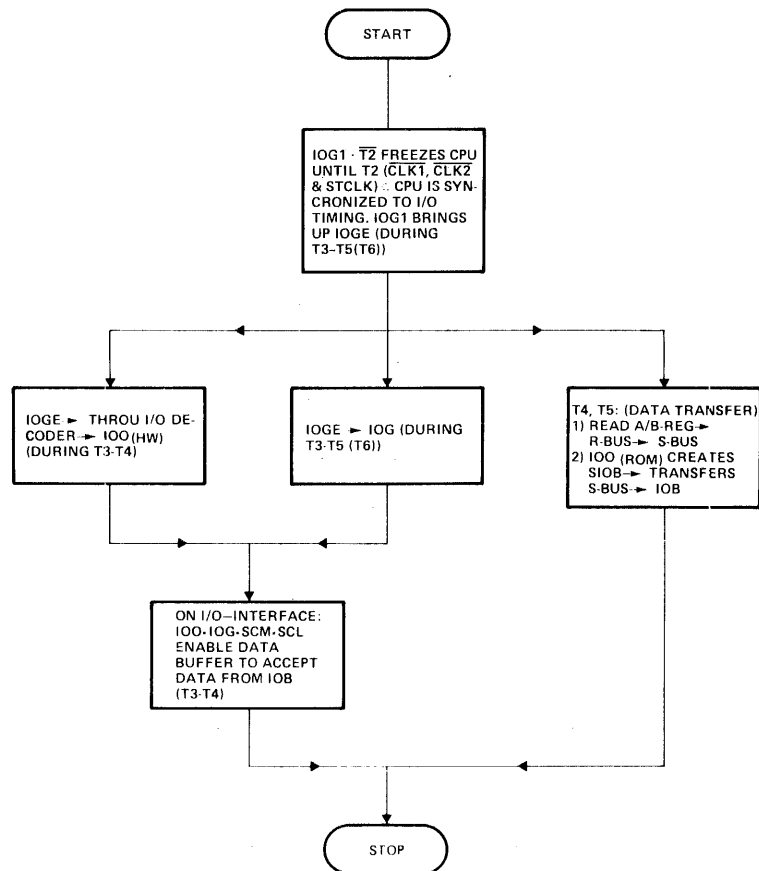
MI*/LI* – INSTRUCTIONS

OM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS	S-BUS	FUNCTION	STORE	SPECIAL	SKIP
0060	77777737	MI*	—	—	IOR	—	IOG1	—
0061	77777777		—	—	IOR	—	—	—
0062	70777775		—	IOI	IOR	—	—	EOP
0063	40775777		CAB	IOI	IOR	CAB	—	—
0064	77777737	LI*	—	—	IOR	—	IOG1	—
0065	77777777		—	—	IOR	—	—	—
0066	70777775		—	IOI	IOR	—	—	EOP
0067	70775777		—	IOI	IOR	CAB	—	—



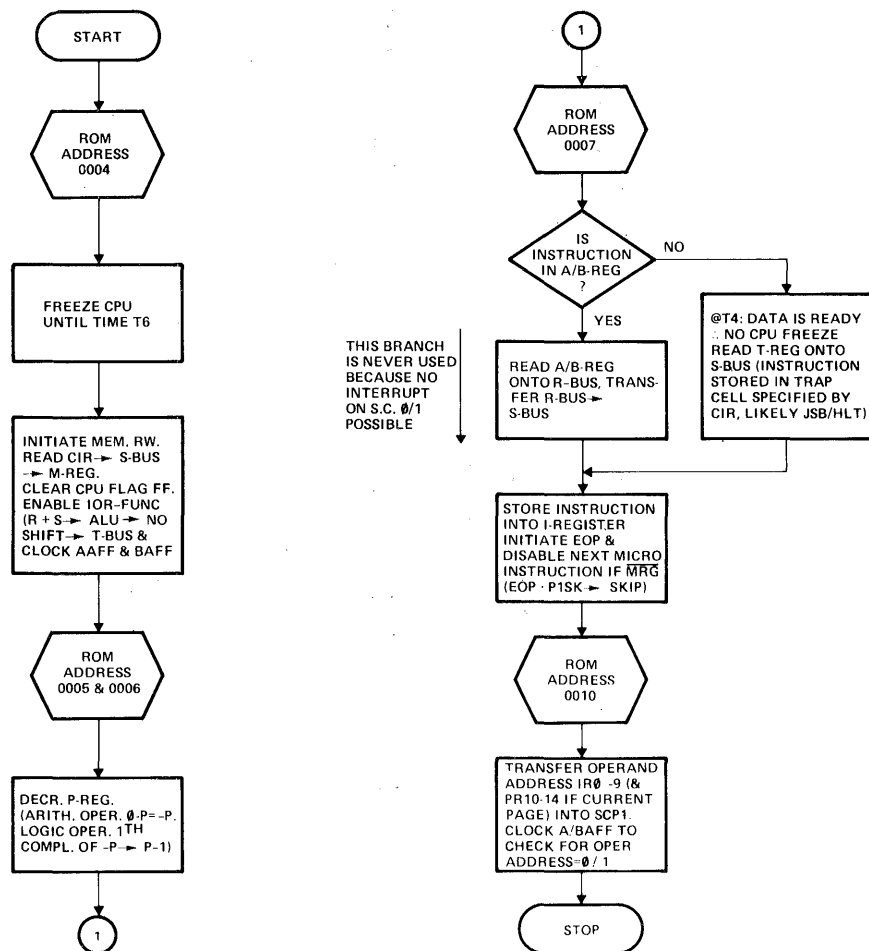
OT* – INSTRUCTION

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS	S-BUS	FUNCTION	STORE	SPECIAL	SKIP
0070	77777737	OT*	—	—	IOR	—	IOG1	—
0071	42377777		CAB	RRS	IOR	—	—	—
0072	42370375		CAB	RRS	IOR	IOO	—	EOP
0073	42370377		CAB	RRS	IOR	IOO	—	—

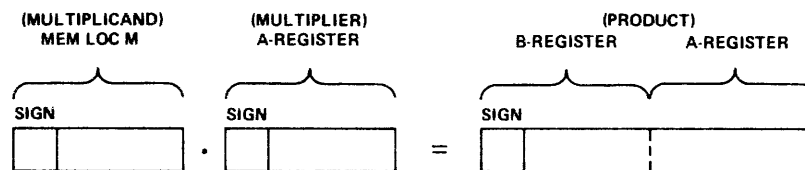


INTERRUPT PHASE

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS	S-BUS	FUNCTION	STORE	SPECIAL	SKIP
0004	70330757	PH1B	—	CIR	CFLG	M	RW	—
0005	77054377		—	P	SUB	P	—	—
0006	77154377		—	P	NOR	P	—	—
0007	53771775		AAB	COND	IOR	IR	—	EOP
0010	73373557		—	ADR	IOR	S1	AAB	—



THE BINARY MULTIPLICATION



MULTIPLICAND POS
MULTIPLIER POS

```

      610 · 310
    0110 · 0011
      0110
    0110
    0000
    0000
    -----
  00010010 → +1810
  
```

MULTIPLICAND POS
MULTIPLIER NEG

```

      610 · -3
    0110 · 1101
      0110
    0000
    0110
    0110
    -----
  01001110 → +7810
  
```

MULTIPLICAND NEG
MULTIPLIER POS

```

     -610 · 310
    1010 · 0011
      1010
    1010
    0000
    0000
    -----
  00011110 → +3010
  
```

MULTIPLICAND NEG
MULTIPLIER NEG

```

     -610 · -310
    1010 · 1101
      1010
    0000
    1010
    1010
    -----
  10000010 → -12610
  
```

HOW CAN AN INTEGER MULTIPLICATION BE IMPLEMENTED BY HARD/FIRMWARE?

THERE ARE TWO WELL KNOWN ALGORITHMS:

1. ALGORITHM

1. CHECK MULTIPLIER. IF NEGATIVE: STORE THE SIGN BIT AND
CONVERT MULTIPLIER TO
POSITIVE EQUIVALENT.
2. CHECK MULTIPLICAND. IF NEGATIVE: STORE THE SIGN BIT AND
CONVERT MULTIPLICAND
TO POSITIVE EQUIVALENT.
3. EXECUTE BINARY MULTIPLICATION
4. EXCLUSIVE OR THE TWO STORED SIGN BITS. IF RESULT IS "1" THE
PRODUCT SHOULD BE
NEGATIVE. $\therefore$ CONVERT THE
PRODUCT TO THE NEGATIVE
EQUIVALENT.

2. ALGORITHM

1. EXECUTE BINARY MULTIPLICATION (REGARDLESS OF SIGN OF MULTIPLICAND AND MULTIPLIER).
2. IF THE MULTIPLIER IS NEGATIVE SUBTRACT THE MULTIPLICAND FROM THE 16 MOST SIGNIFICAND BITS OF THE PRODUCT.
3. IF THE MULTIPLICAND IS NEGATIVE SUBTRACT THE MULTIPLIER FROM THE 16 MOST SIGNIFICAND BITS OF THE PRODUCT.

THE HP 2100 A MICROPROGRAM UTILIZES THE 2. ALGORITHM TO PERFORM A MULTIPLICATION.

WHY DOES THE 2. ALGORITHM EXECUTE A BINARY MULTIPLICATION CORRECTLY?

EXAMPLE:
$$\begin{array}{r} 0110 \cdot 1101 \\ 0110 \\ 0110 \\ 0110 \\ \hline 01001110 \end{array}$$

EXPECTED COMPUTATION: $6_{10} \cdot (-3_{10}) = -18_{10}$
 $X \cdot M_e = P_e$
 PERFORMED COMPUTATION: $6_{10} \cdot 13_{10} = 78_{10}$
 $X \cdot M_l = P_l$

THE MULTIPLICAND IS TOO BIG BY A VALUE OF $16_{10} = 2^n$

(n = NR. OF BITS OF MULTIPLICAND)

$$M_l = M_e + 2^n$$

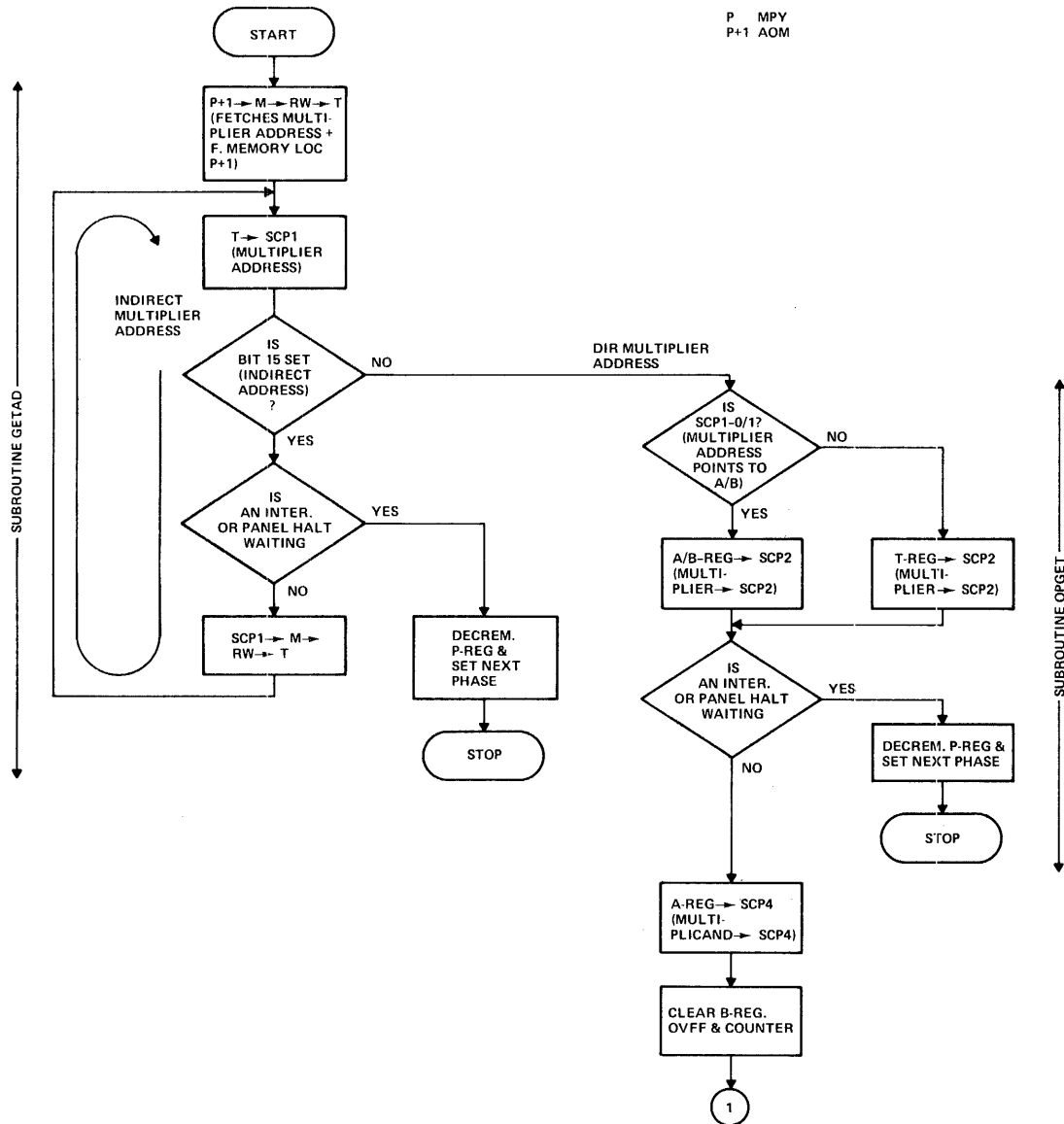
$$\therefore P_l = X \cdot M_l = X \cdot (M_e + 2^n) = P_e + X \cdot 2^n$$

$$P_e = P_l - X \cdot 2^n$$

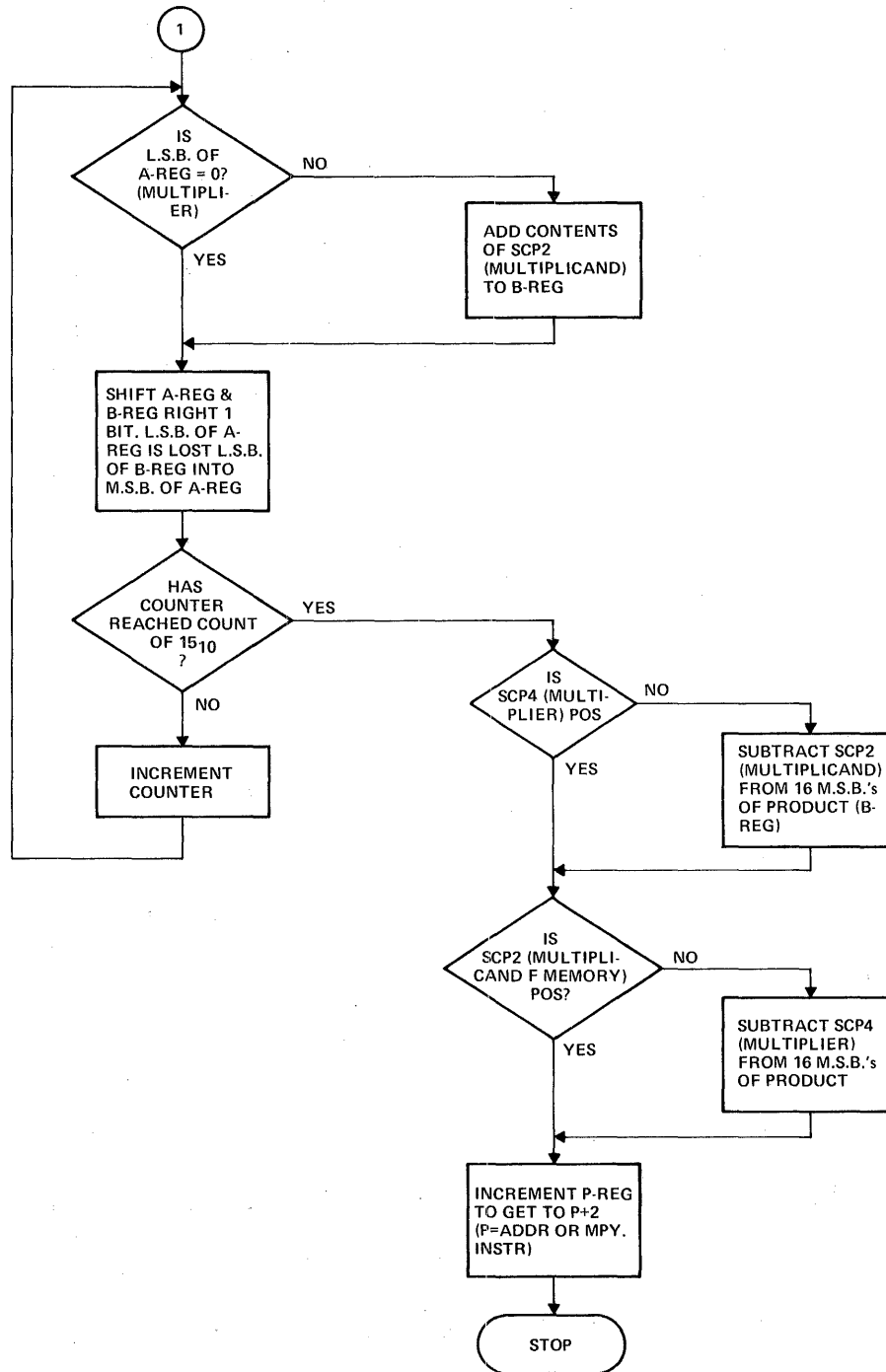
01001110	←	P _l
- 01100000	←	-(X · 2 ⁿ)
<u>11101110</u>	→	P _e = -18 ₁₀

X = MULTIPLICAND
 M_e = EXPECTED MULTIPLIER
 M_l = INTERPERED MULTIPLIER
 P_e = EXPECTED PRODUCT
 P_l = INTERPERED PRODUCT

MPY - INSTRUCTION



MPY – INSTRUCTION (CONTINUE)



MPY — MICROPROGRAM

ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS						COMMENTS
			R-BUS 23-21	S-BUS 20-17	FUNCTION 16-12	STORE 11-8	SPECIAL 7-4	SKIP 3-0	
0210	77777577	MULT	—	—	IOR	—	LEP	—	Legal entry point for MPY. Execute next line.
0211	77207762		—	P	JSB	—	GETAD	—	Execute GETAD subroutine. Puts multiplier address in S1.
0212	77207632		—	P	JSB	—	OPGET	—	Execute OPGET subroutine. Puts multiplier in S2.
0213	07222155		A	P	JMP	S4	MPY	—	Save multiplicand in S4. Jump to 0155.
0214	77777777		—	—	IOR	—	—	—	Unused.
0215	77777777		—	—	IOR	—	—	—	Unused.
0216	77777775		—	—	IOR	—	—	EOP	Unused.
0217	77777777		—	—	IOR	—	—	—	Unused.
0362	77370757	GETAD	—	P	IOR	M	RW	—	Send P-register address to memory, start read cycle.
0363	53773403	ONEMO	AAB	COND	IOR	S1	RSS	NEG	Normally fetches contents of a software DEF instruction.
0364	77227766		—	P	JMP	—	IND	—	Put fetched word into S1. Skip next line if not indirect (i.e., if bit 15 = 0).
0365	75657557		—	S1	RSB	—	AAB	—	Jump to 0366.
0366	77247411	IND	—	—	CJMP	—	PDEC	—	Set AAF/BAF if S1 contents = 0 or 1. Return to calling subroutine.
0367	75770757		—	S1	IOR	M	RW	—	Jump to 0011 if interrupt or panel halt. Else continue.
0370	77227763		—	—	JMP	—	ONEMO	—	Send operand address to memory and start read cycle.
0232	75770757	OPGET	—	S1	IOR	M	RW	—	Jump to 0363.
0233	53773377		AAB	COND	IOR	S2	—	—	Send operand address to memory and start read cycle.
0234	77247411		—	P	CJMP	—	PDEC	—	Put operand data into S2.
0235	77657777		—	—	RSB	—	—	—	Jump to 0011 if interrupt or panel halt. Else continue.
0155	77756461	MPY	—	—	CLO	B	CNTR	RPT	Return to calling routine.
0156	15036450		B	S2	MPY	B	R1	CTRI	Continued from 0213:
0157	14377403		—	S4	IOR	—	RSS	NEG	Clear counter, B-register, and Overflow. Set repeat mode.
0160	15056777		B	S2	SUB	B	—	—	Execute MPY on B and S2 16 times. Result in B, A-registers.
0161	75377403		—	S2	IOR	—	RSS	NEG	Skip next line if multiplicand (was A-register) is positive.
0162	14056775		—	S4	SUB	B	—	EOP	Subtract multiplier from high order word of result.
0163	77114377		—	P	INC	P	—	—	Skip next line if multiplier (from memory) is positive.
0011	77054375	PDEC	—	P	SUB	P	—	EOP	Subtract multiplicand from high word. Set next phase.
0012	77154377		—	P	NOR	P	—	—	Increment P-register past the DEF software instruction.
									Decrement the P-register and set next phase: P → -P
									One's comp of -P = P-1

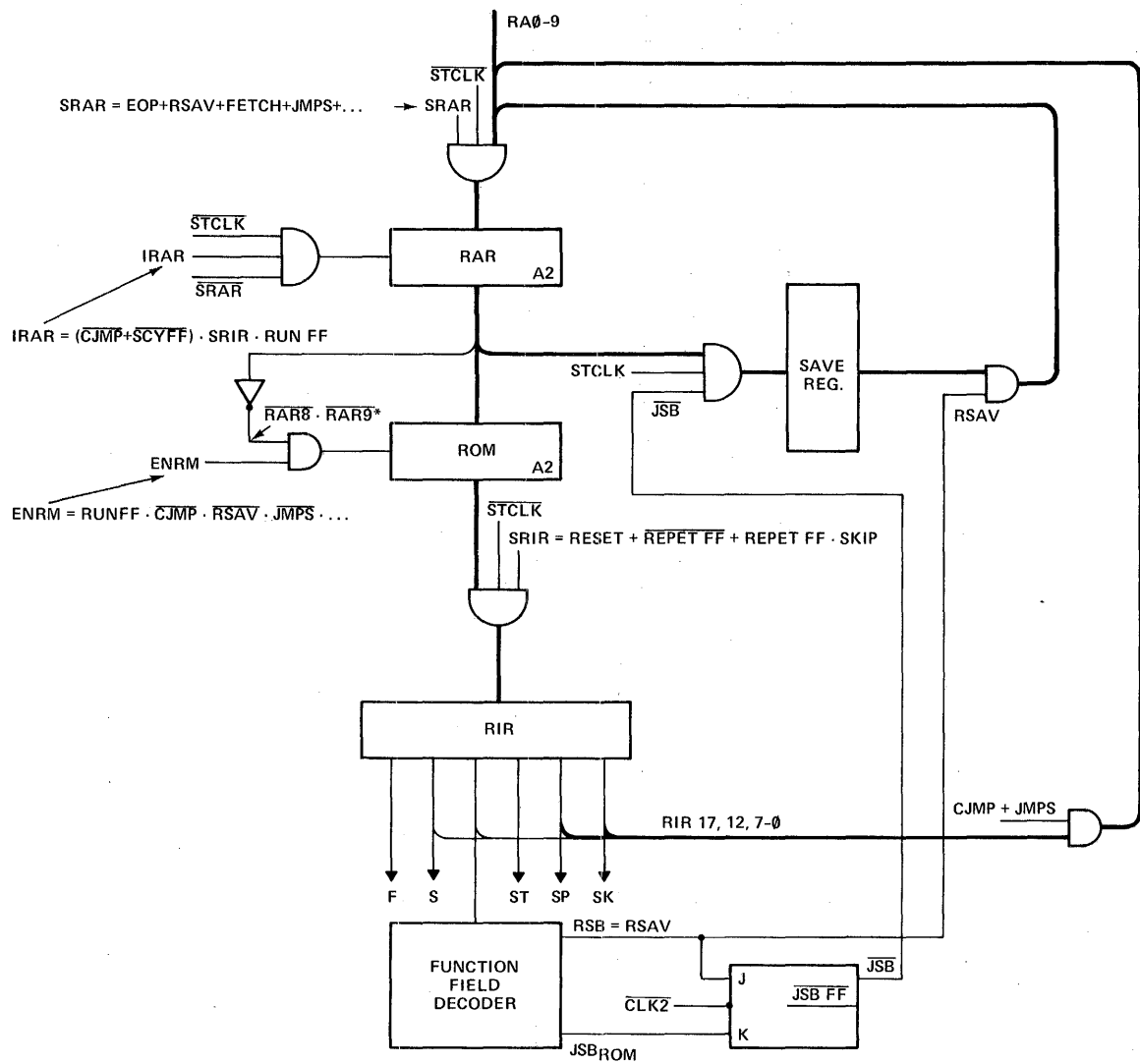
SPECIAL MICROORDERS REQUIRED FOR THE EXECUTION OF EAG-MEMORY REF. INSTRUCTIONS

ROM ADDR	MICROORDER	BIT CONF	COMMENT
0210	LEP	0111	MACRO-CODE ($\overline{IR15} \cdot \overline{IR14} \cdot \overline{IR13} \cdot \overline{IR12} \cdot \overline{IR10} \cdot \overline{IR11} \cdot \overline{IR9}$) @ SPH 3 CLEARS LEP FF $\therefore$ ALL MICROINSTR. WILL BE SKIPPED. LEP SETS LEP FF $\therefore$ INHIBITS SKIP. (ON A2)
0211	JSB GETAD	11110010	JSB TO RA 362 IN ROM MODULE 0. (SEE FIG. 3-33 & 3-34) SETS $\overline{JSB}$ FF (A4) $\rightarrow$ $\therefore$ SAVE REGISTER (A2) WILL NO MORE BE CLOCKED $\rightarrow$ $\therefore$ PRESENT ADDRESS IN SAVE REGISTER IS LOCKED IN UNTIL $\overline{JSB}$ FF CLEARS.
0363	RSS NEG	00000011	SKIP FIELD DECODER (A6) U24A & U92B $\rightarrow$ SKIP = RSS $\cdot$ NEG $\cdot \overline{ALU15}$ $\rightarrow$ SKIP NEXT MICROINSTR. IF NOT INDIRECT ($\overline{ALU15}$).
0366	CJMP PDEC	00001001	COND JMP TO RA11. (SEE FIG. 3-33 & 3-34) IF RUN MODE AND FRONT PANEL HLT SWITCH PRESSED (A24 S29 $\rightarrow$ RELEASES THE RESET SWBN FF $\rightarrow$ THROUGH DELAY CIRCUIT Q1 & Q2 SETS SWBN FF $\rightarrow$ SETS HLT SW FF THROUGH U41B & U43B $\rightarrow$ HIN TO A1 RESETS RH FF, SIN FF & SCY FF THROUGH U75B, U86F TO U96D WHERE ANDED WITH CJMP $\rightarrow$ RJMP, SRAR, $\overline{ENRM}$, $\overline{IRAR}$ $\therefore$ EXECUTES RJMP. TO ROM ADDR. 11 IF RUN MODE AND INTERRUPT (A1, PIN 22) TO U96B WHERE ANDED WITH CJMP $\rightarrow$ RJMP, SRAR, $\overline{ENRM}$, $\overline{IRAR}$ $\therefore$ EXECUTES RJMP TO ROM ADDR. 11 IF RUN MODE AND ANY OTHER MICROORDER $\rightarrow$ $\overline{RJMP}$, $\overline{SRAR}$, $\overline{ENRM}$ AND $\overline{IRAR}$ $\therefore$ EXECUTES NOP. IF SINGLE CYCLE MODE: $\overline{SCY}$ FF (A24) CLEARS $\rightarrow$ SSCY TO A1 THROUGH U105A SETS SCY FF UNTIL CJMP THROUGH U65B RESETS SCY FF @ NEXT CLK1 $\rightarrow$ $\overline{IRAR}$, $\overline{ENRM}$, RJMP AND $\overline{SRAR}$ (ROM JMP IS NOT EXECUTED). U105A ALSO SETS RUN FF DURING SCY FF IS SET. DHLT FF RESETS DURING RUN FF SET + 200 NS. (HALTS MACHINE)

SPECIAL MICROORDERS REQUIRED FOR THE EXECUTION OF EAG-MEMORY REF. INSTRUCTIONS (CONTINUE)

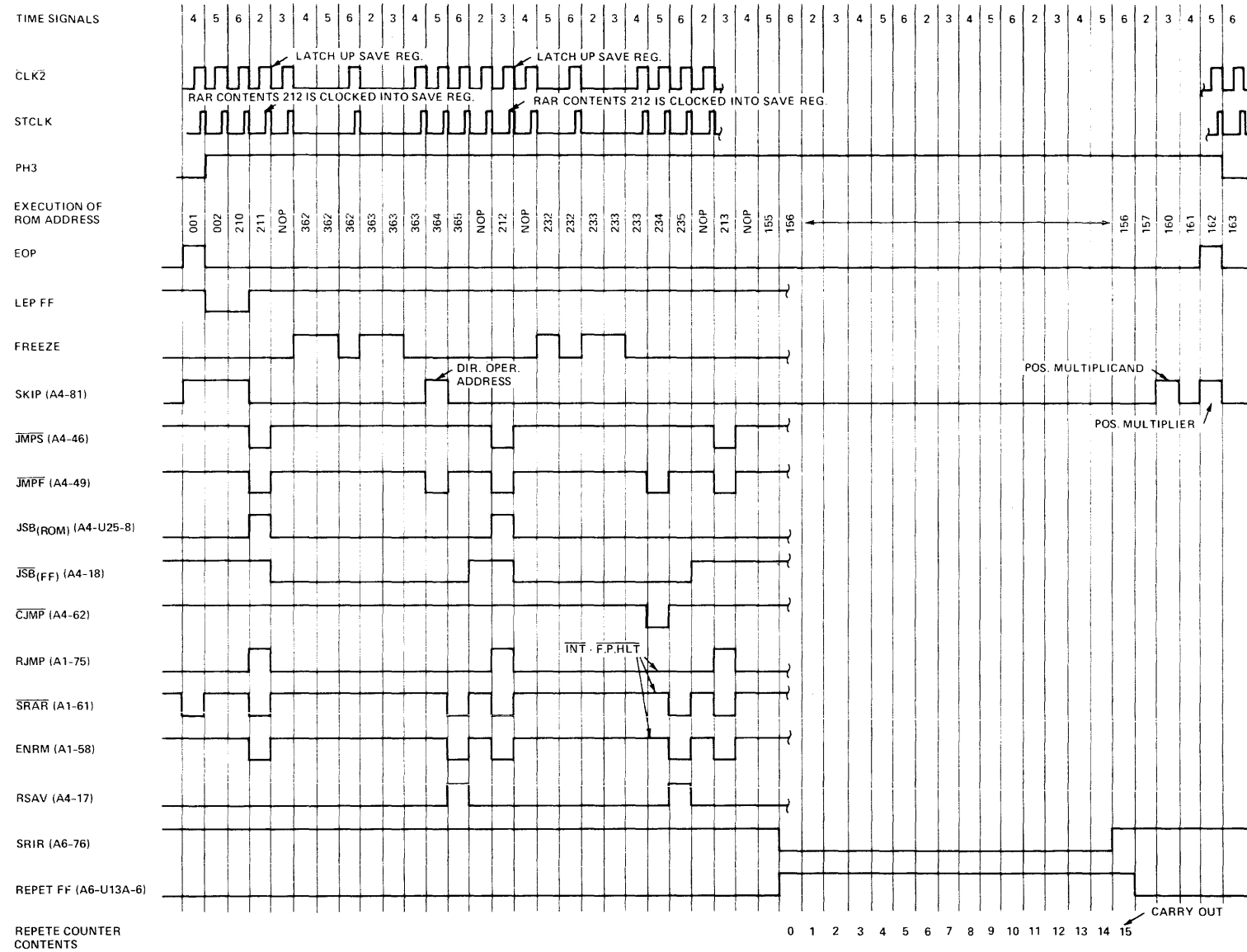
ROM ADDR	MICROORDER	BIT CONF	COMMENT
365	RSB	10101	RSB BRINGS UP R _{SAV} → ENABLES READ OUT FROM SAVE REG TO RAO-9 AND STORES RETURN ADDRESS INTO RAR. RSB ALSO SETS $\overline{\text{JSB FF}}$ → ENABLING THE REGISTER TO FOLLOW THE RAR OUTPUT AGAIN.
155	CNTR, RPT	00110001	ENABLES SB ₀₋₃ TO BE READ INTO THE REPEAT COUNTER. PRESENTLY $\overline{\text{O'S ONS-BUS}}$ ∴ CLEARS REPEAT COUNTER. RPT CLEARS $\overline{\text{RPT FF}}$ ∴ SRIR GOES LOW AND INHIBITS CLOCKING OF $\overline{\text{SKIP FF}}$.

ROM JSB

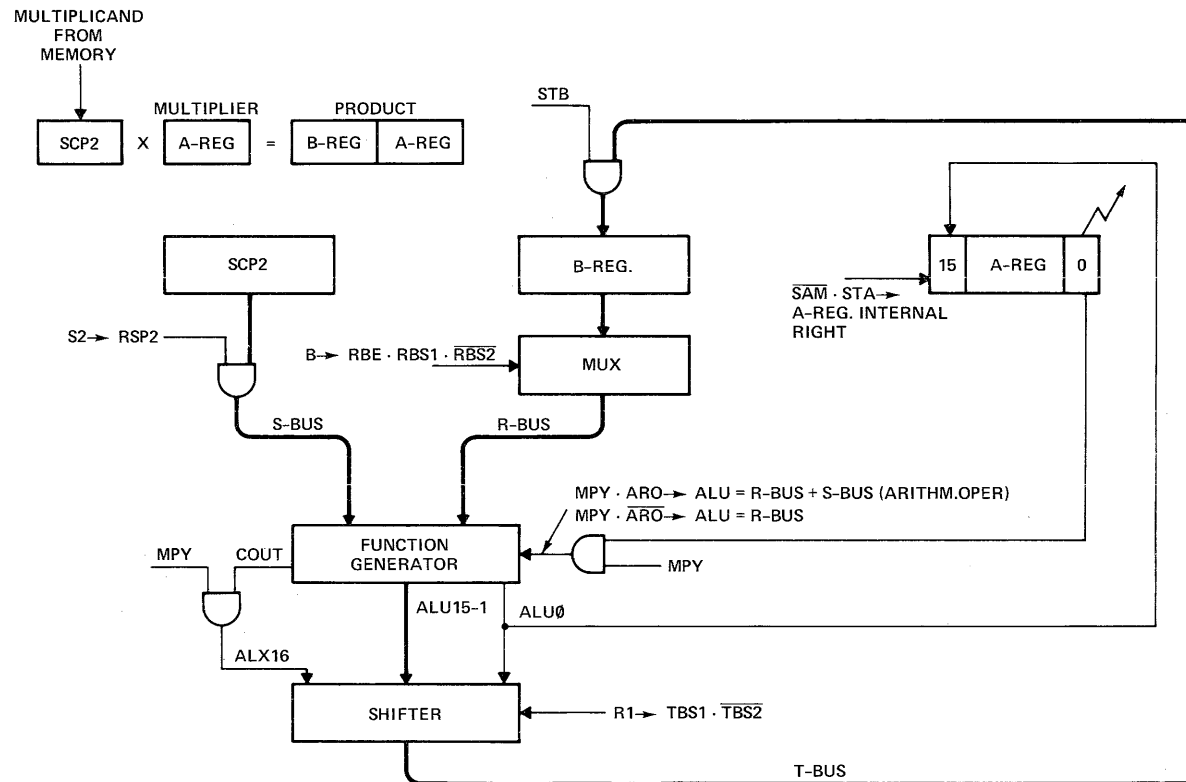


*CONSIDER ROM MODUL 0 ONLY

TIMING DIAGRAM OF PH3, DIR. MPY – INSTR. (OP. ADDR NOT IN A/B-REG.)



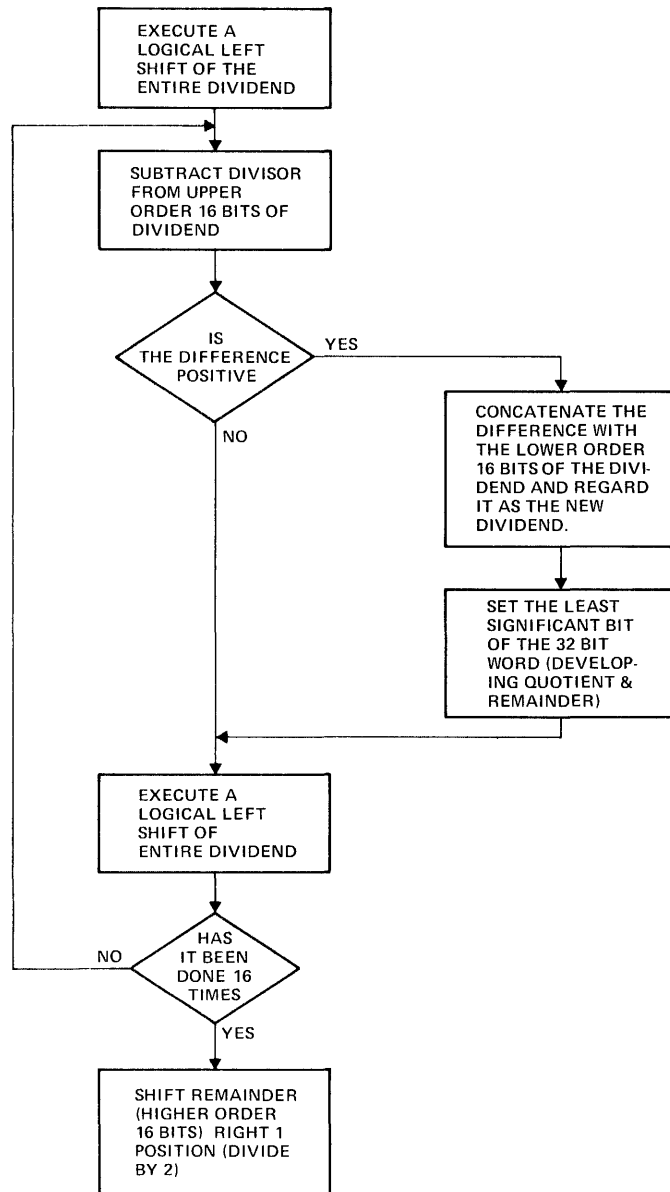
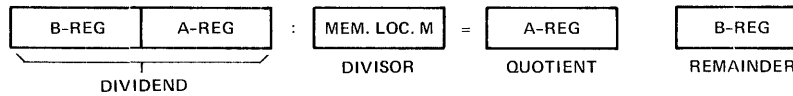
EXECUTION OF MULTIPLICATION



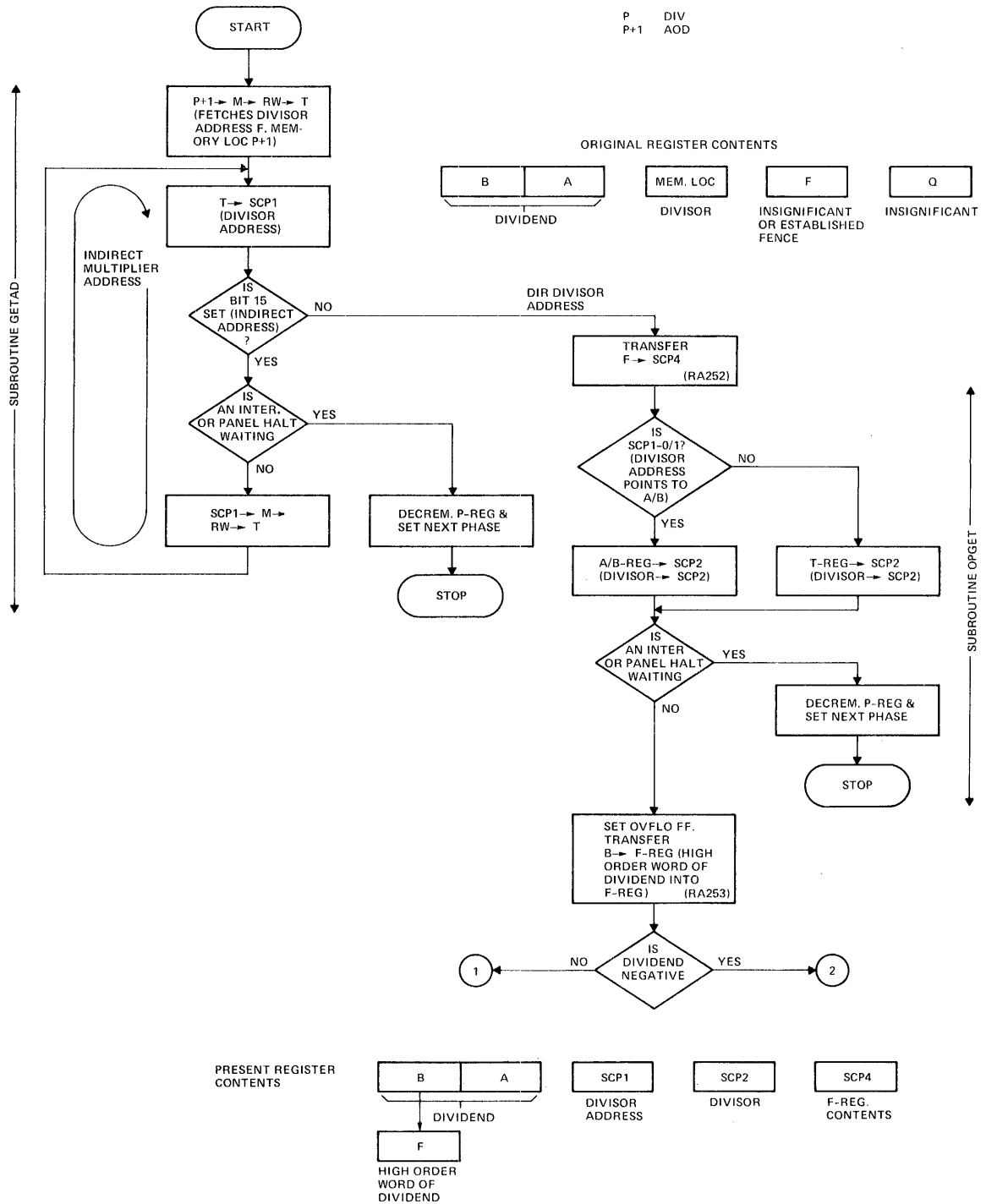
ROM ADDRESS	ROM WORD (OCTAL)	ENTRY POINT LABEL	FIELD CONTENTS					
			R-BUS	S-BUS	FUNCTION	STORE	SPECIAL	SKIP
0156	15036450		B	S2	MPY	B	R1	CTRI

THE BINARY DIVISION

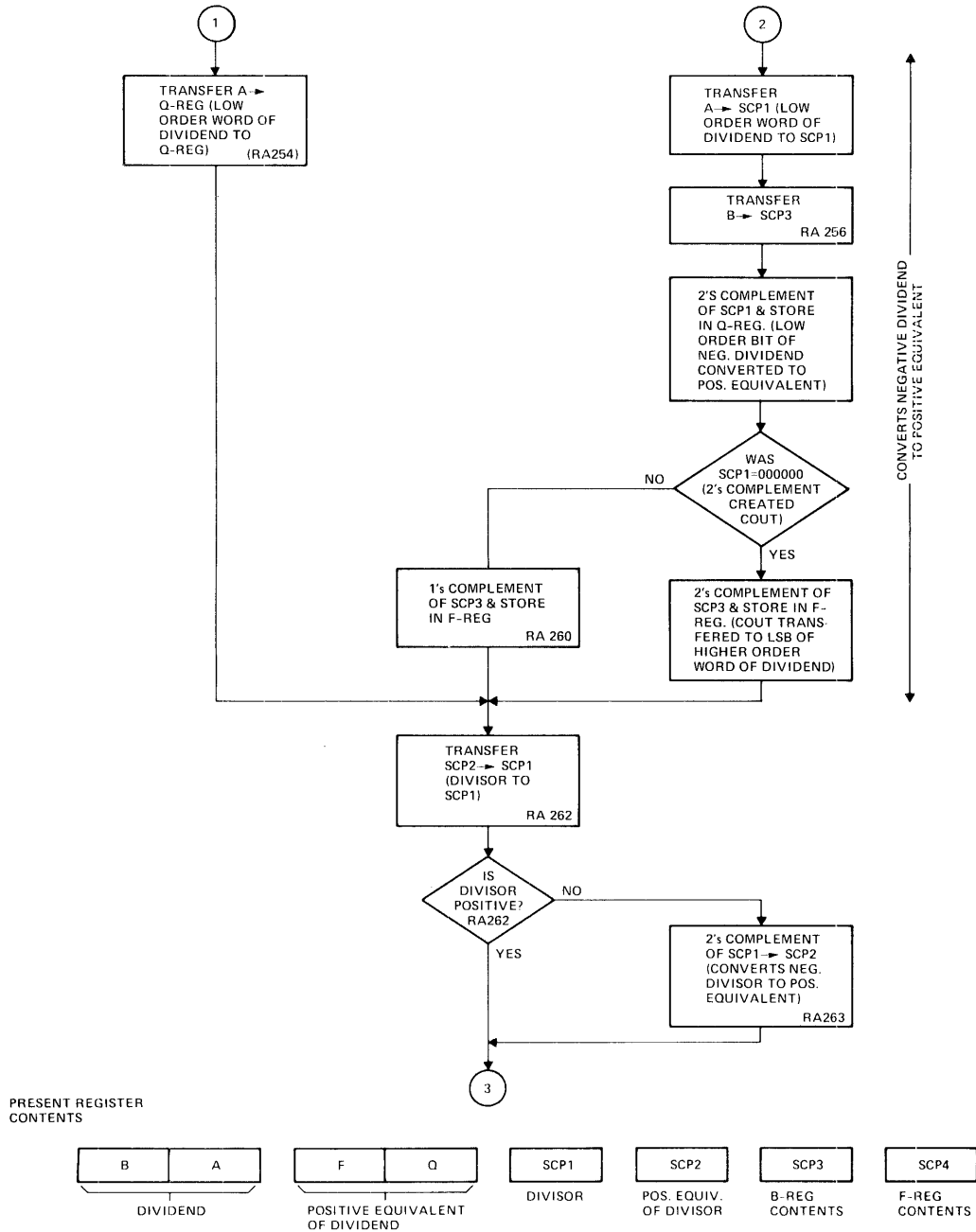
- TO EXECUTE A BINARY DIVISION CORRECTLY DIVIDEND AS WELL AS DIVISOR HAVE TO BE POSITIVE FOR THE SAME REASON AS ILLUSTRATED IN THE MULTIPLY INSTRUCTION FIGURE



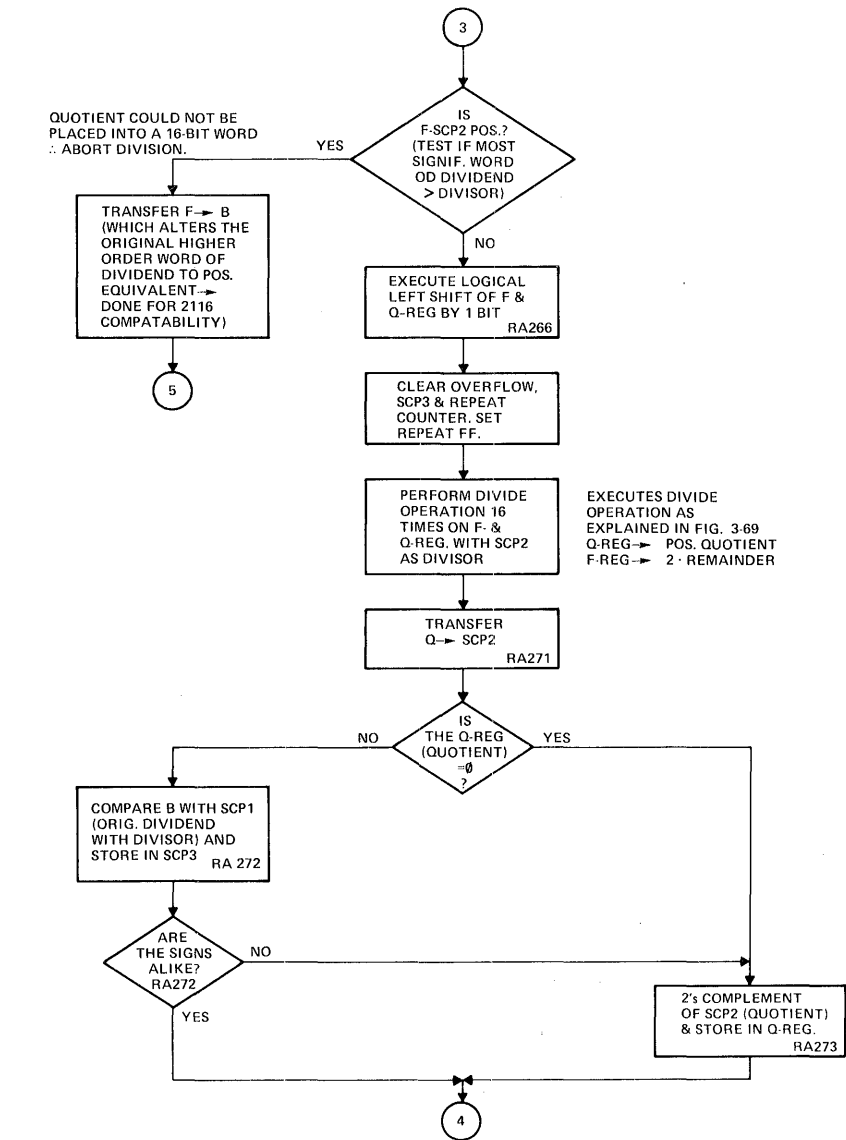
DIV - INSTRUCTION



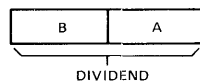
DIV - INSTRUCTION (CONTINUE)



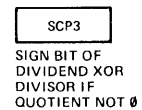
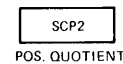
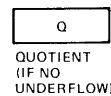
DIV - INSTRUCTION (CONTINUE)



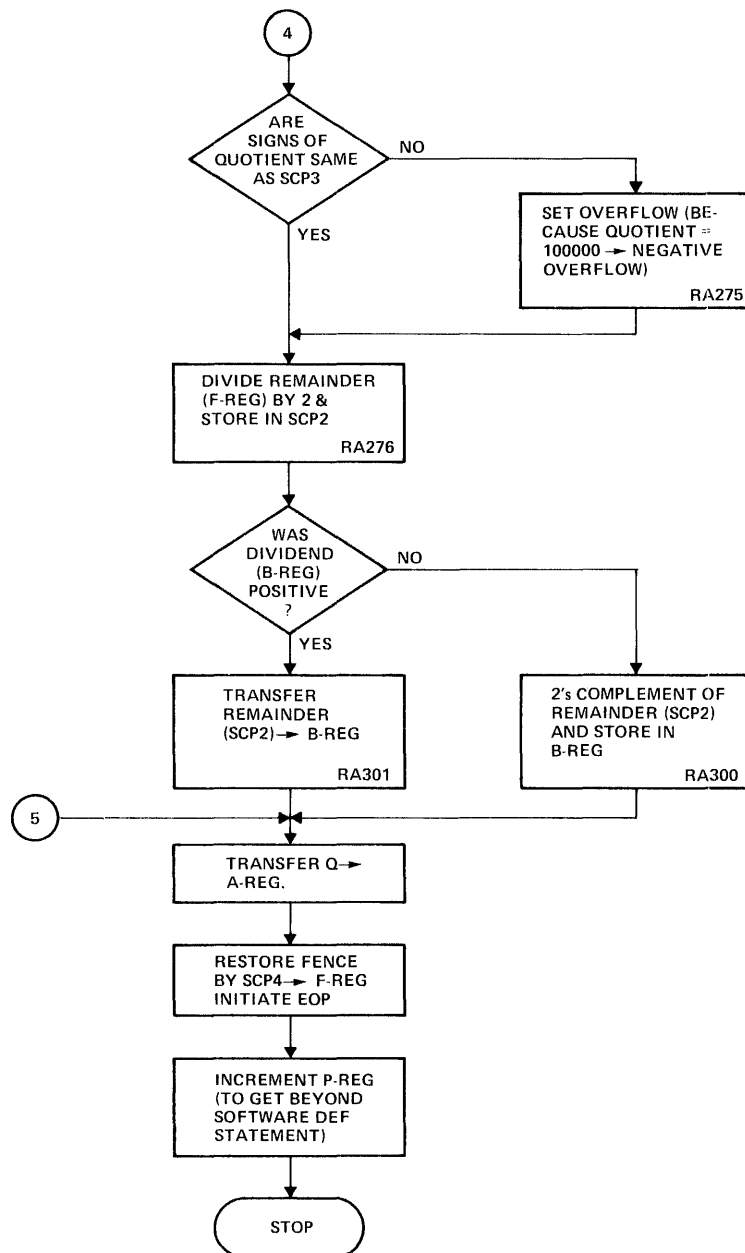
PRESENT REGISTER
CONTENTS IF
DIVISION NOT
ABORTED



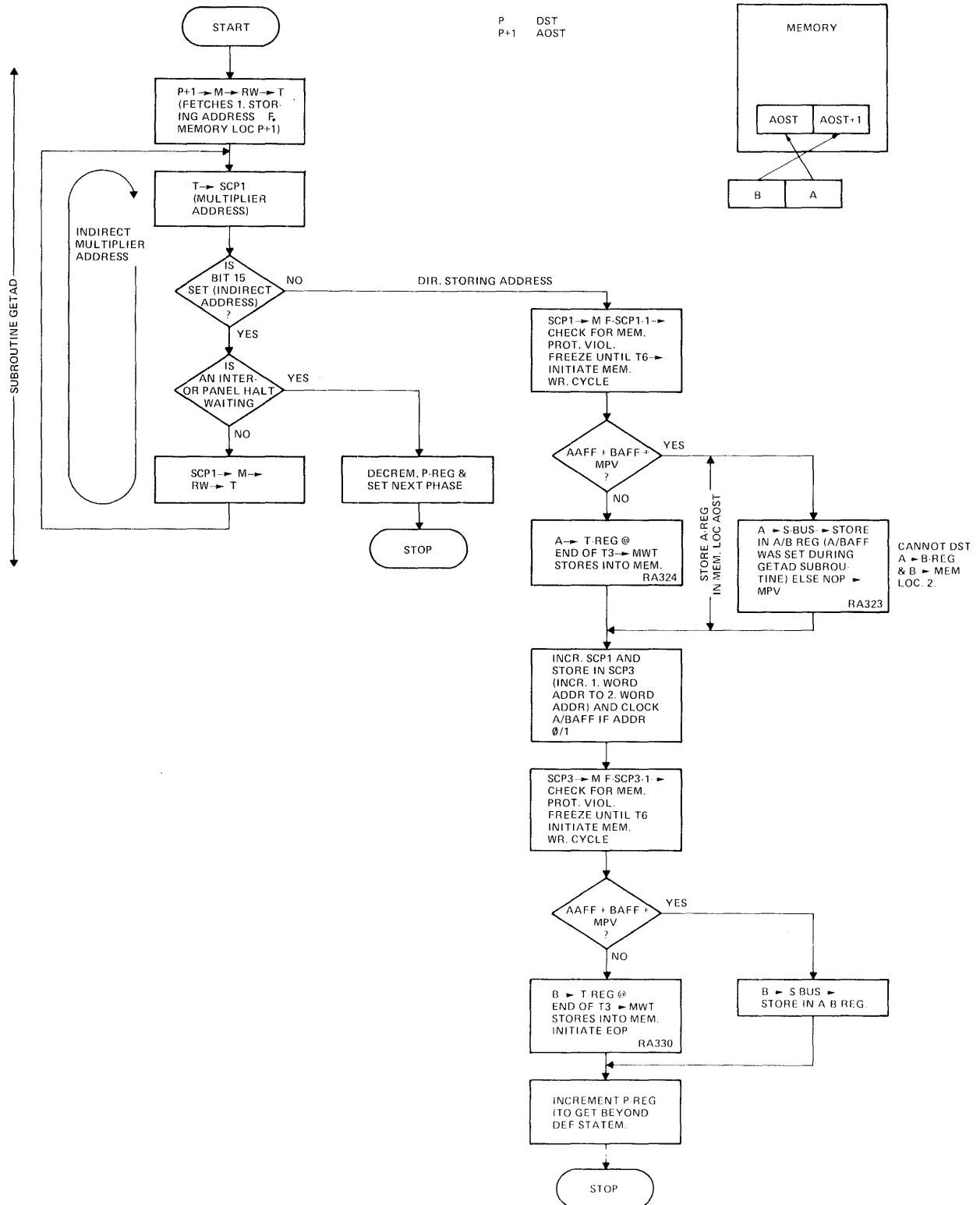
2 * REMAINDER



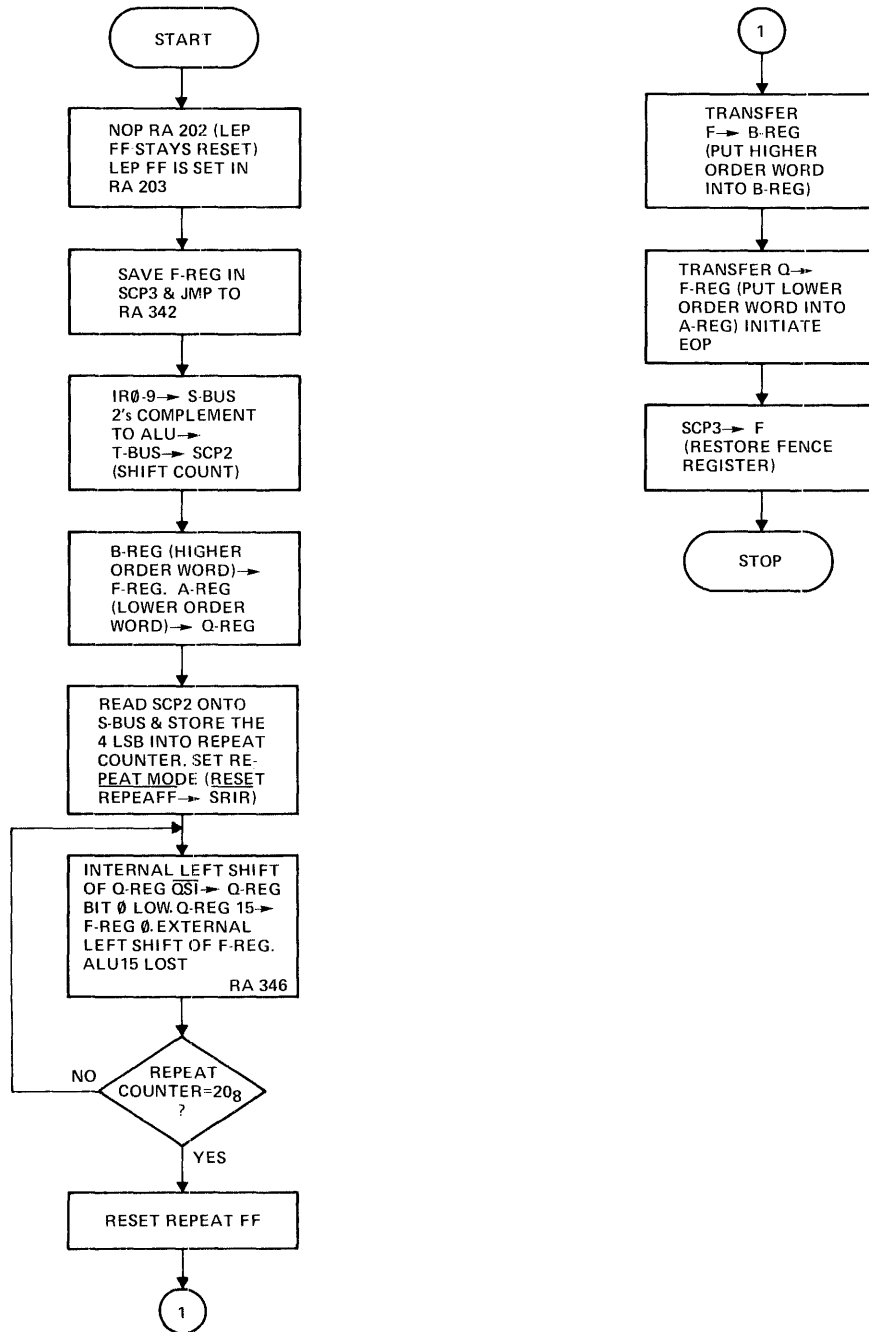
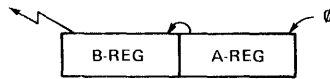
DIV – INSTRUCTION (CONTINUE)



DST - INSTRUCTION



LSL - INSTRUCTION



memory

4

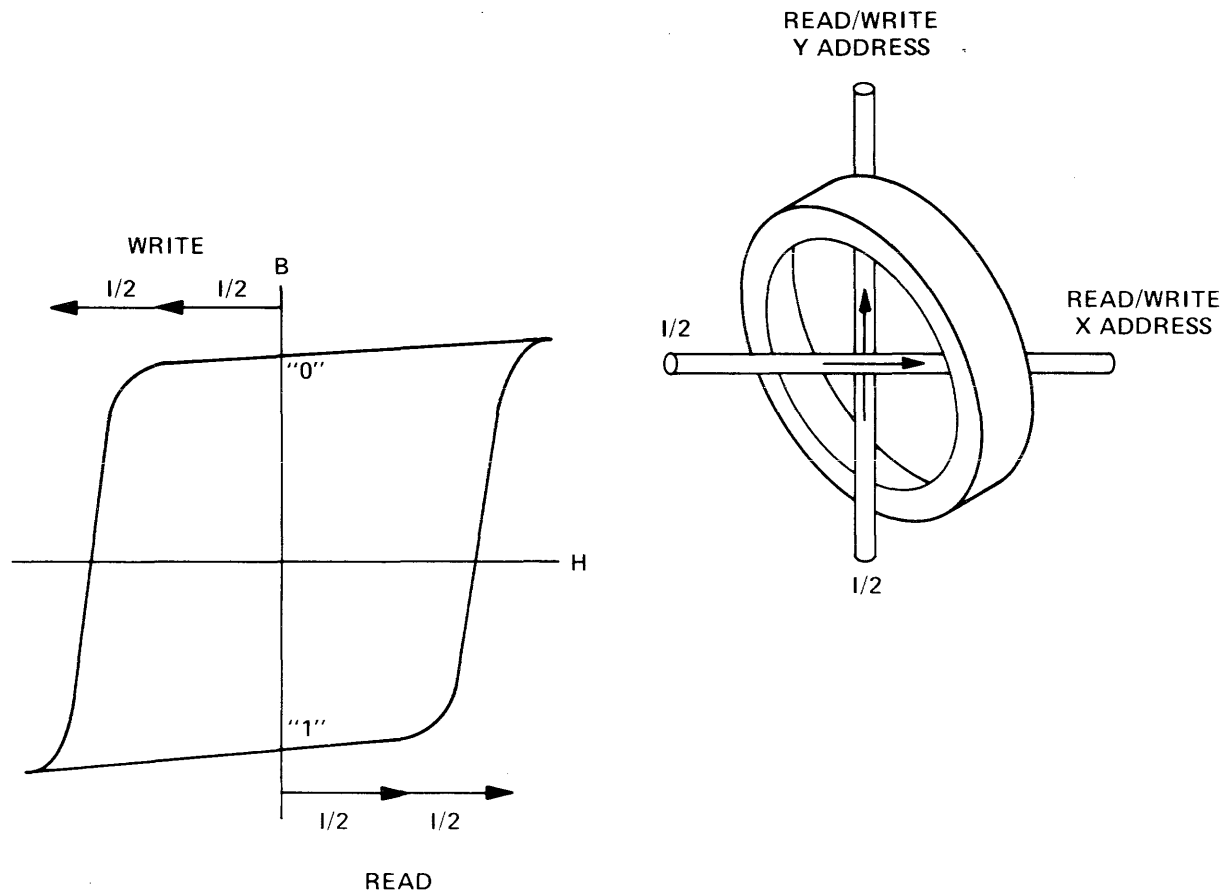


LESSON 4

MEMORY

The Magnetic Core	4-1	Wire Lay Out In 3-Wire Memory	4-9
Read/Write Currents VS Core Status	4-2	Typical Data Sense and Inhibit	4-10
Determining Core Status	4-3	Simplified Memory Addressing	4-11
The 3-Wire System	4-4	Memory Block Diagram	4-12
The 4K Bit Matrix	4-5	Memory Section Card Cage Loading	4-13
Memory Address Identification	4-6	Configurations	
4K Core Stack	4-7	8K Memory HP 2100A	4-14
Isolation Diodes On A 16 X 16 Matrix	4-8	Block Diagram of HP 2100A Memory	4-15
		HP 2100 Data Control Card Signals (A107)	4-16

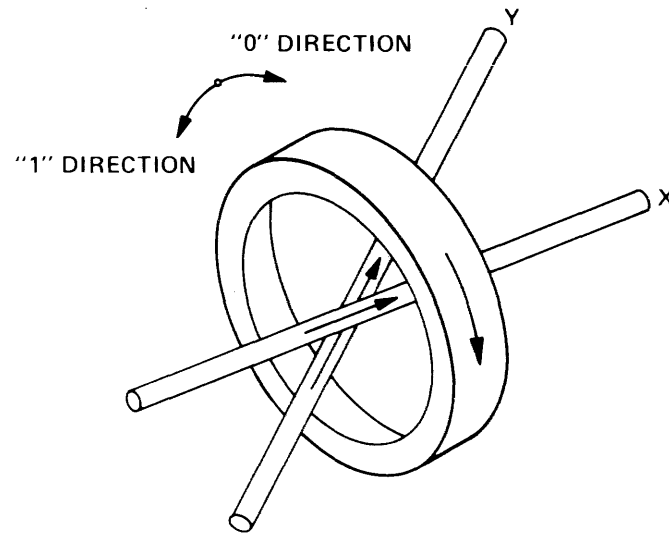
THE MAGNETIC CORE



THE CORE IS ALWAYS SATURATED WHEN IN A STATIC STATE (NRZI)
A CHANGE OF STATE INDICATES READING OR WRITING A "1"

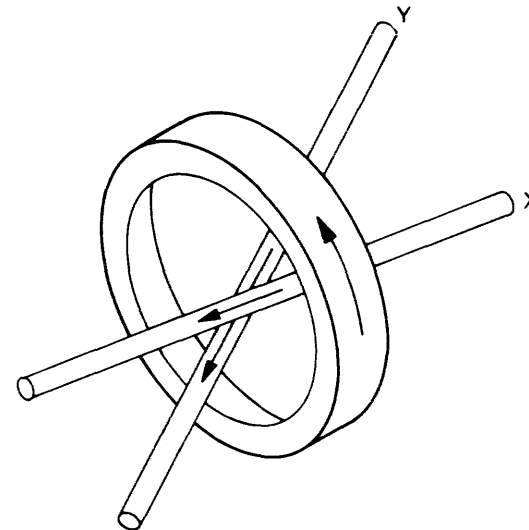
READ/WRITE CURRENTS VS CORE STATUS

READ CYCLE



CORE ORIGINALLY
MAGNETIZED IN "1" DIRECTION,
READ CURRENT SWITCHES CORE
(RESETS CORE TO ZERO DIRECTION)

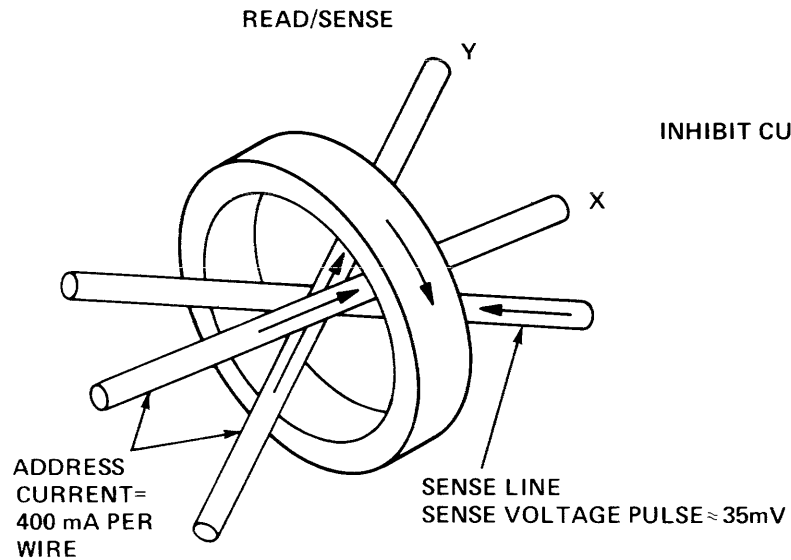
WRITE CYCLE



MAGNETIZED IN "0" DIRECTION,
WRITE CURRENT SWITCHES CORE
TO "1" DIRECTION

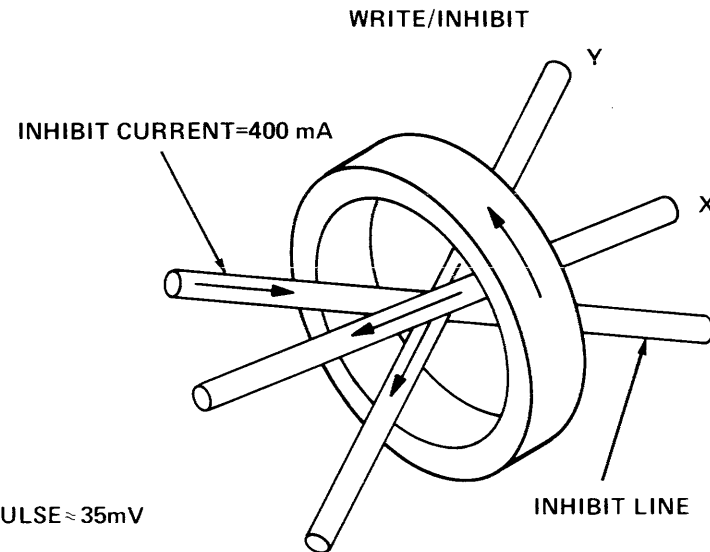
CURRENT DURING WRITE CYCLE
IS IN OPPOSITE DIRECTION THAN
DURING READ

DETERMINING CORE STATUS



SENSE "1"

READ CURRENT SWITCHES CORE
IF MAGNETIZED IN CLOCK DIRECTION →
NO CHANGE → NO SENSE PULSE

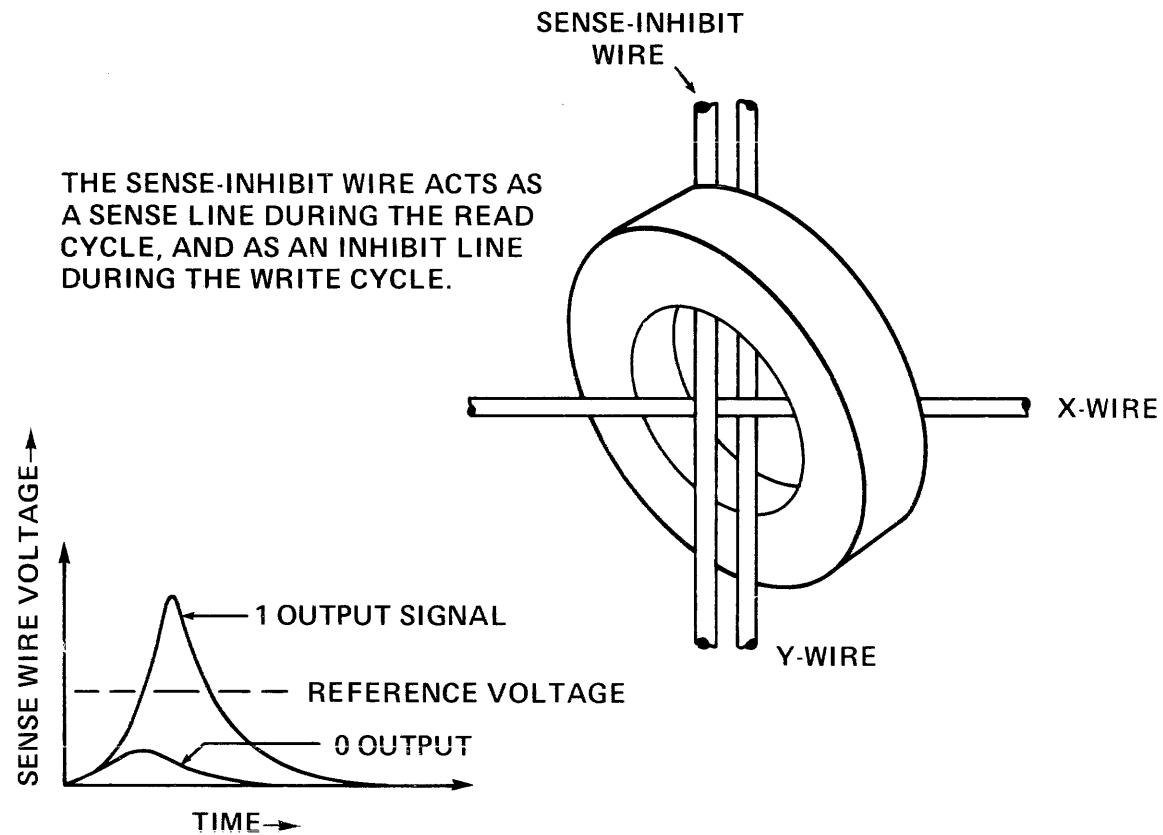


WRITE "0"

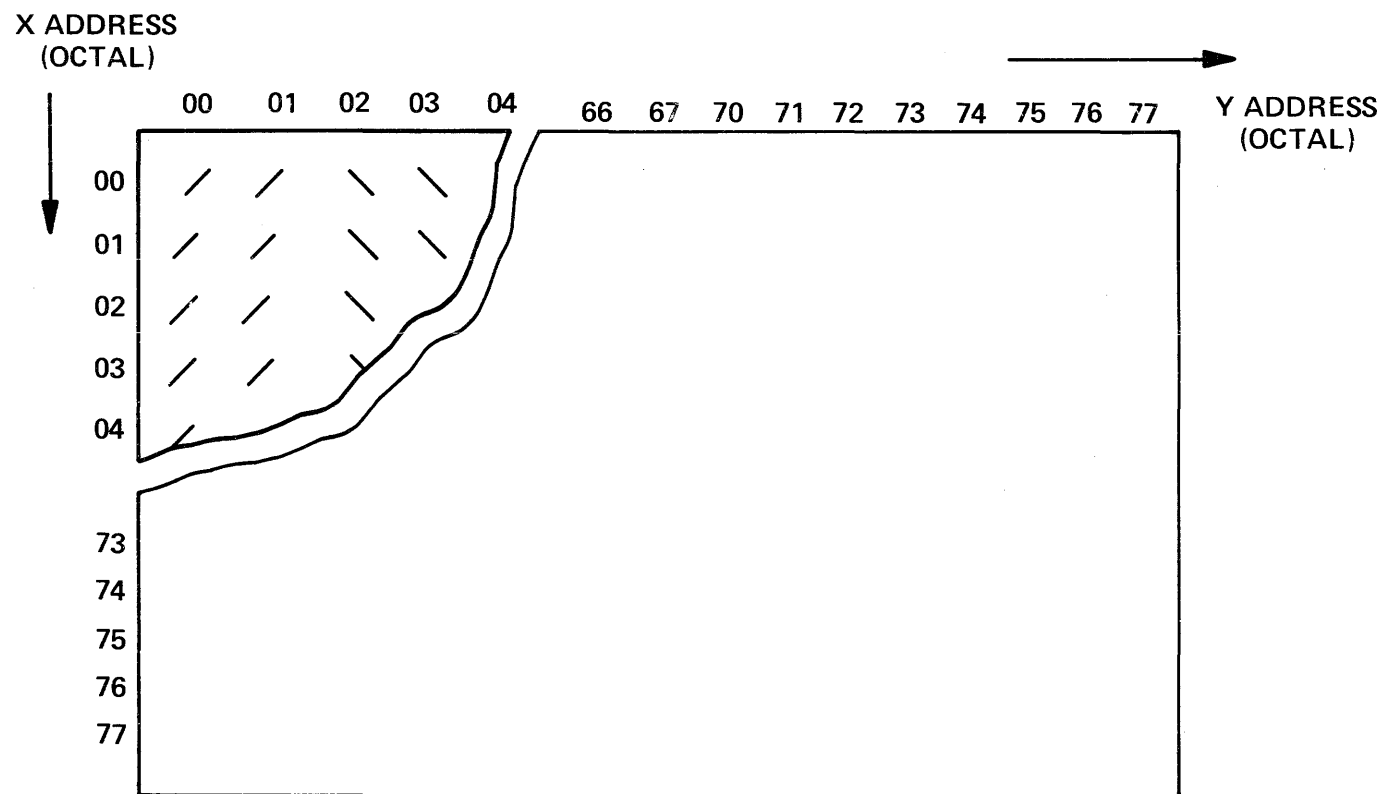
INHIBIT CURRENT PREVENTS
CORE FROM SWITCHING

THE 3-WIRE SYSTEM

THE SENSE-INHIBIT WIRE ACTS AS A SENSE LINE DURING THE READ CYCLE, AND AS AN INHIBIT LINE DURING THE WRITE CYCLE.



THE 4K BIT MATRIX

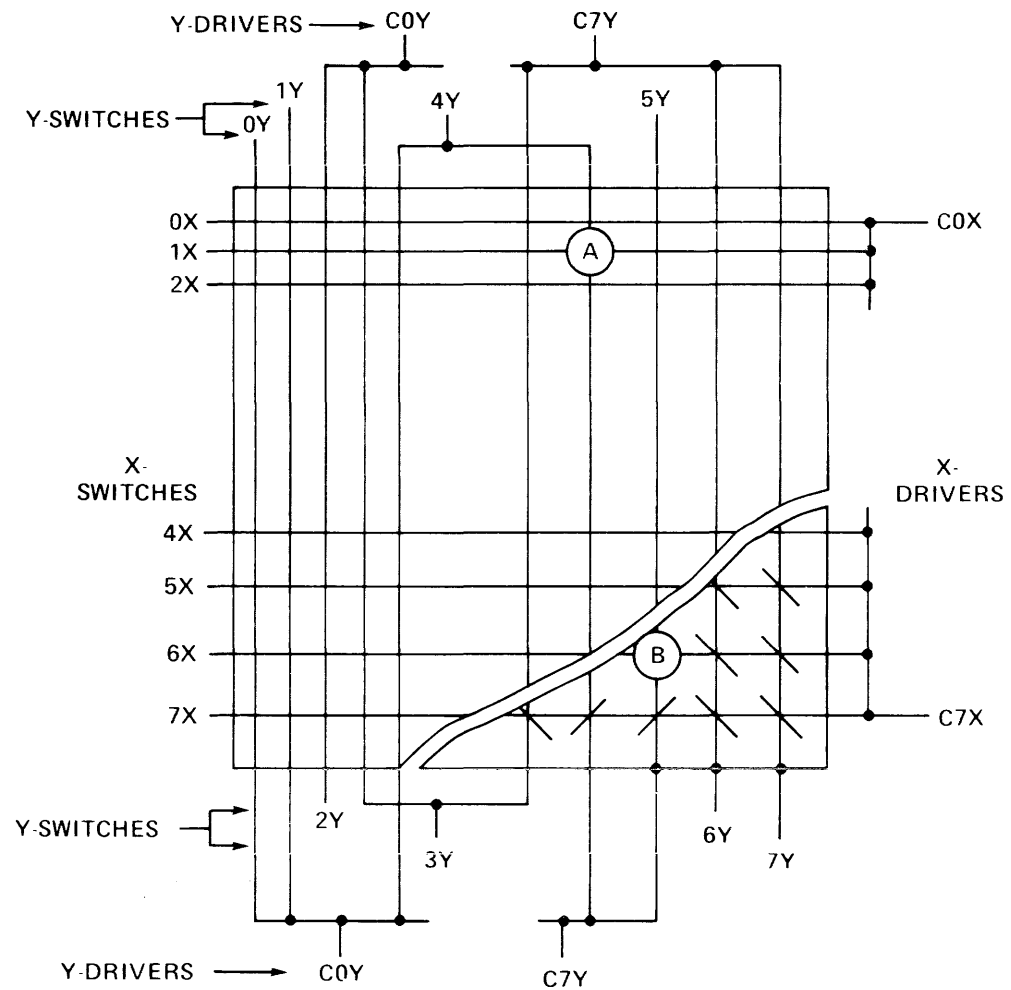


4K BIT MATRIX CONTAINS $64 \times 64 = 4096_{10}$ (10,000₈) CORES

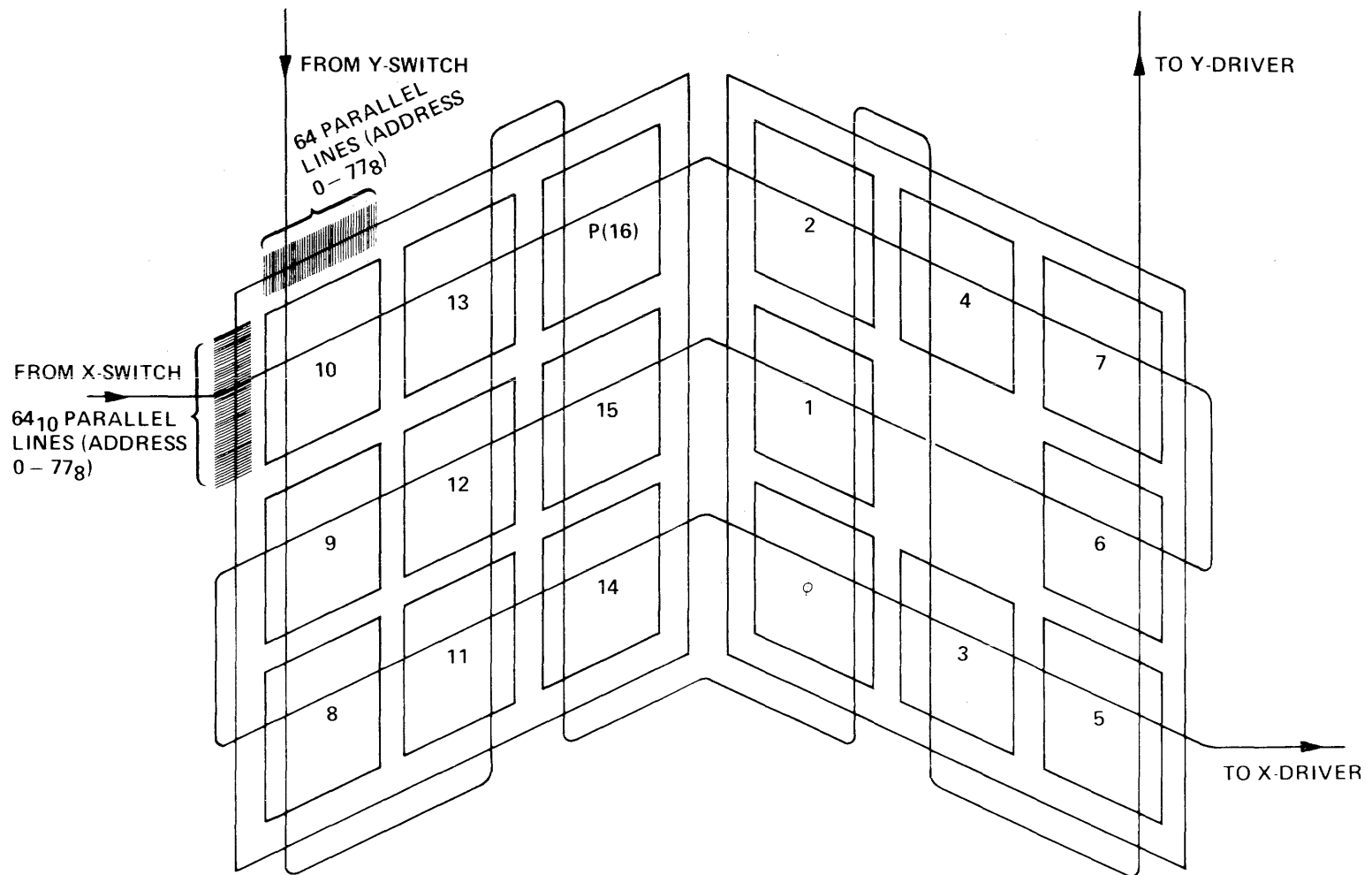
MEMORY ADDRESS IDENTIFICATION

M-REGISTER				
	Y		X	
	DR.	SW.	DR.	SW.
A	7	4	0	1
B	7	5	7	6

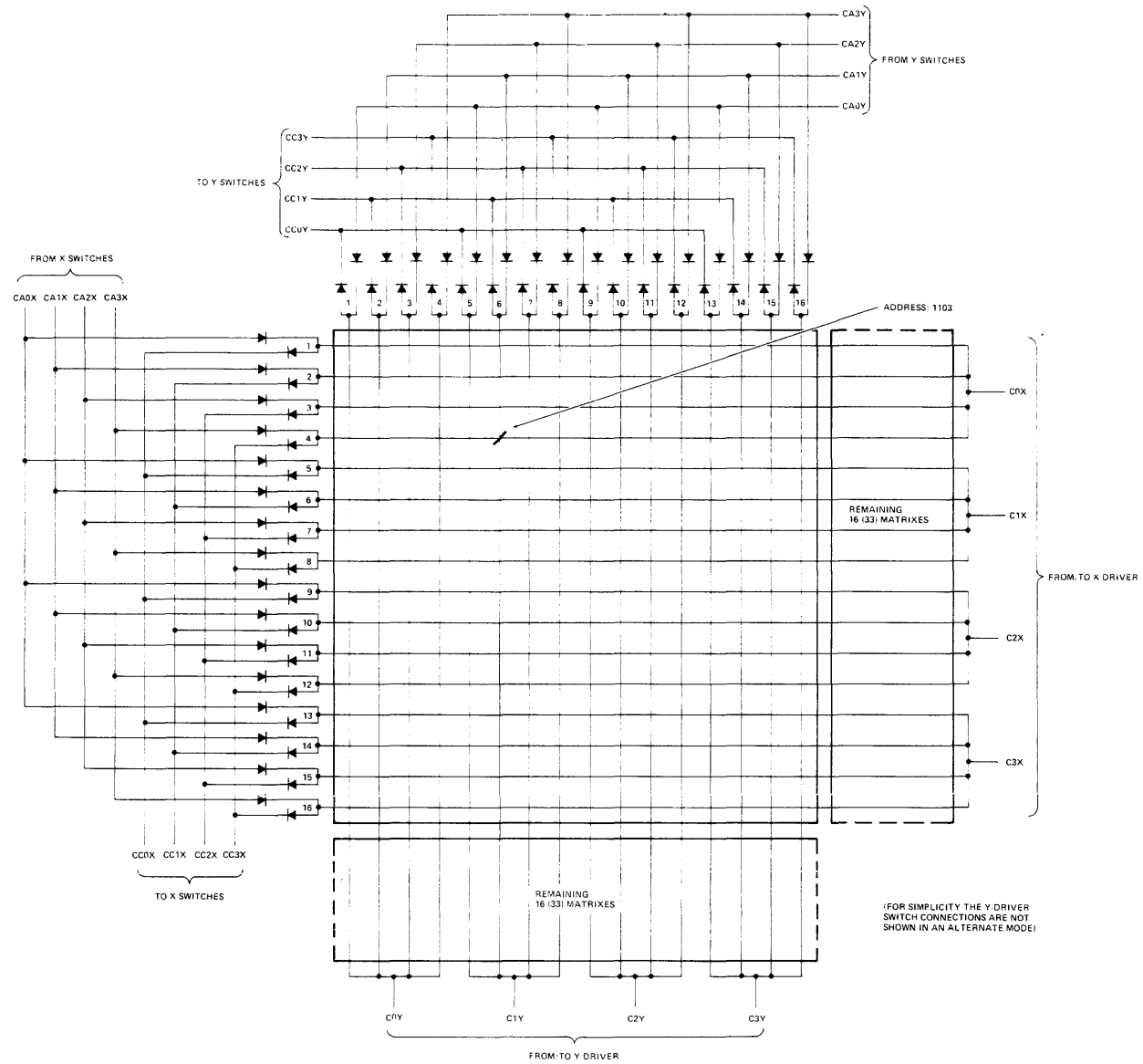
C0 - C7 ARE DRIVERS
0 - 7 ARE SWITCHES



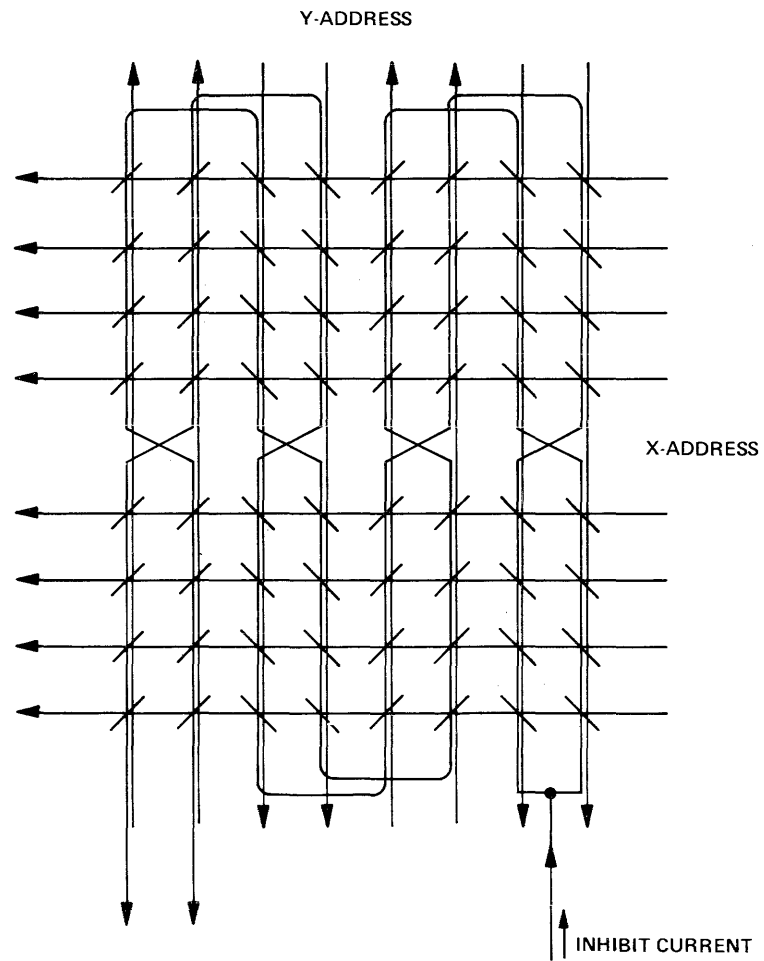
4K CORE STACK



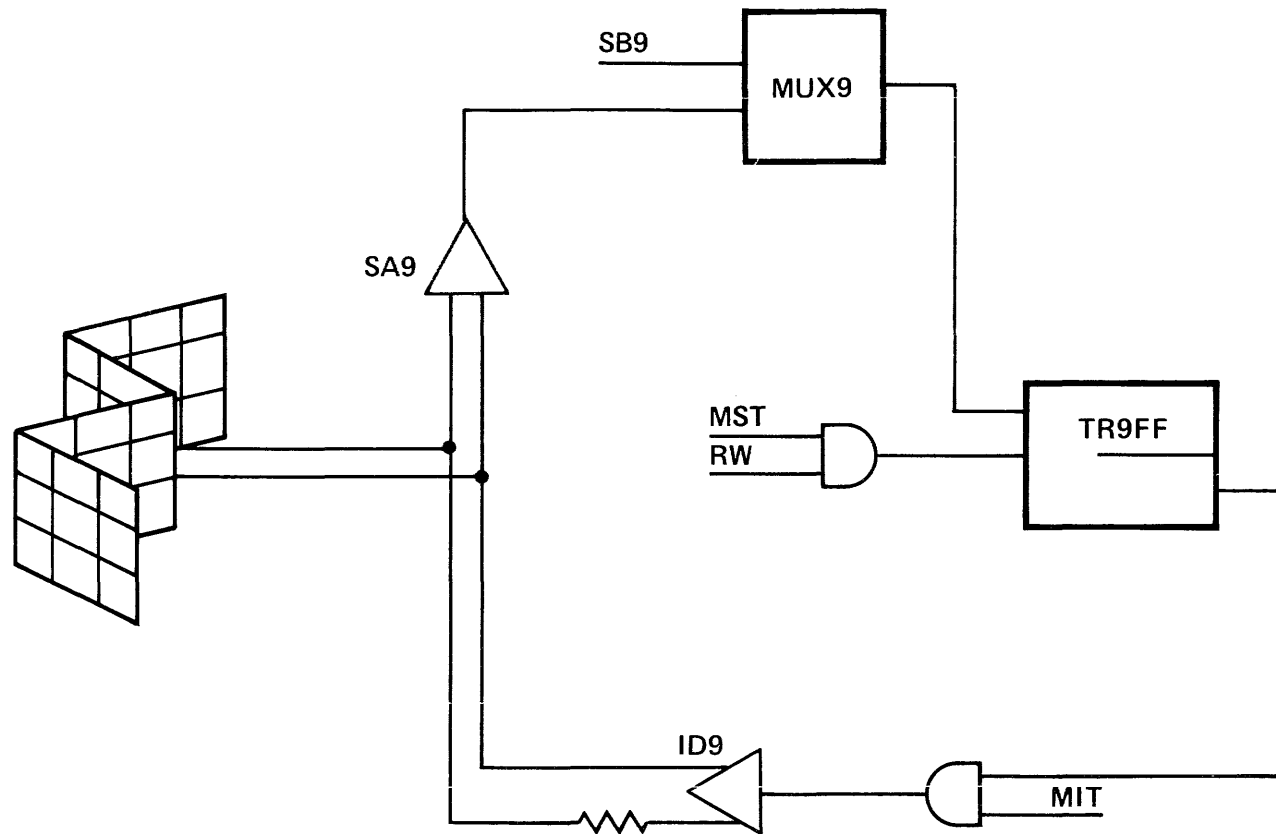
ISOLATION DIODES ON A 16 X 16 MATRIX



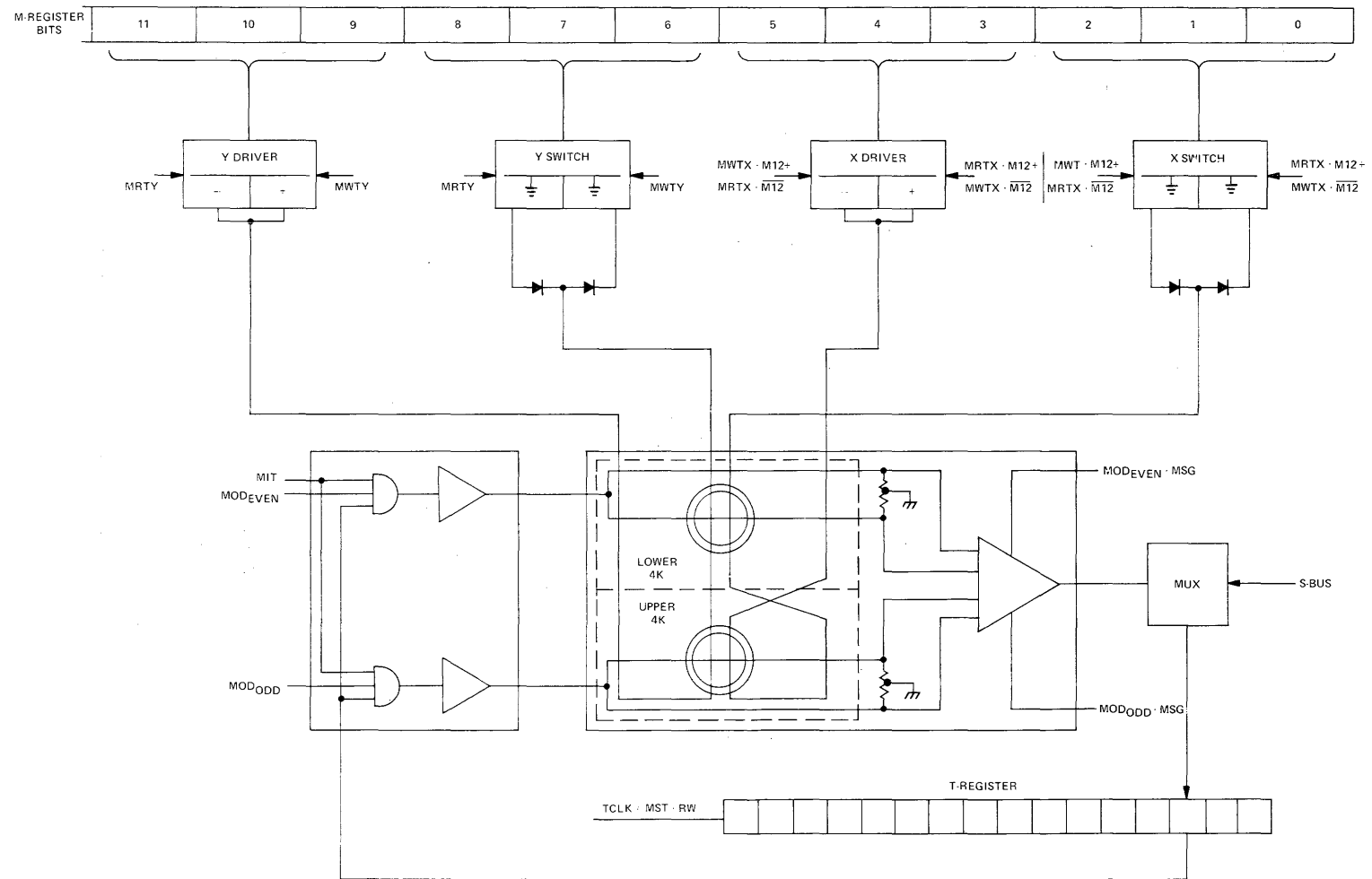
WIRE LAY OUT IN 3-WIRE MEMORY
(8 X 8 MATRIX)



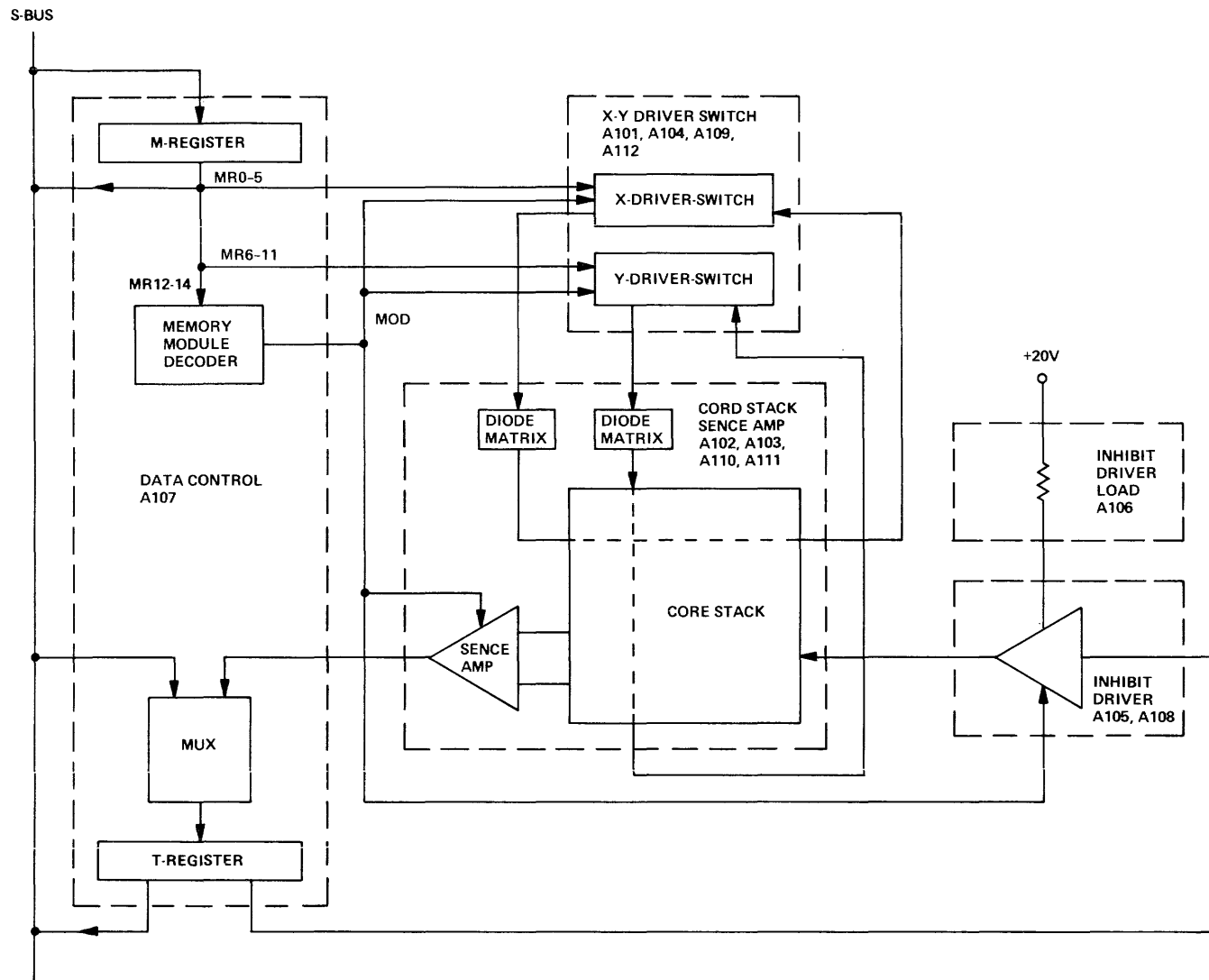
TYPICAL DATA SENSE AND INHIBIT



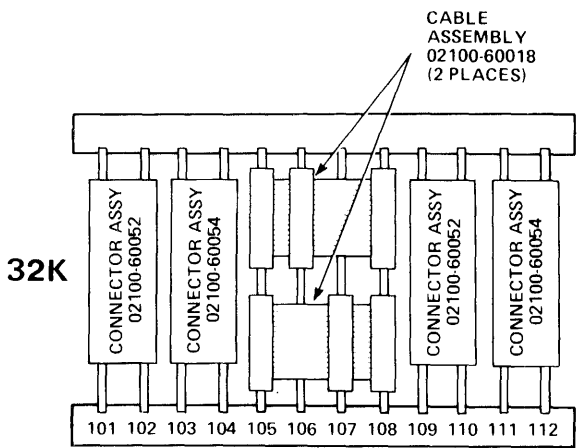
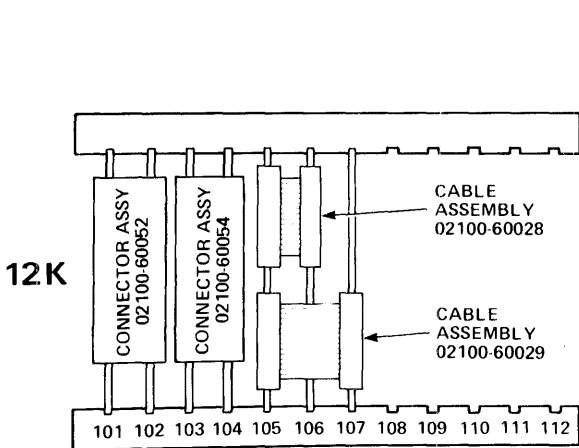
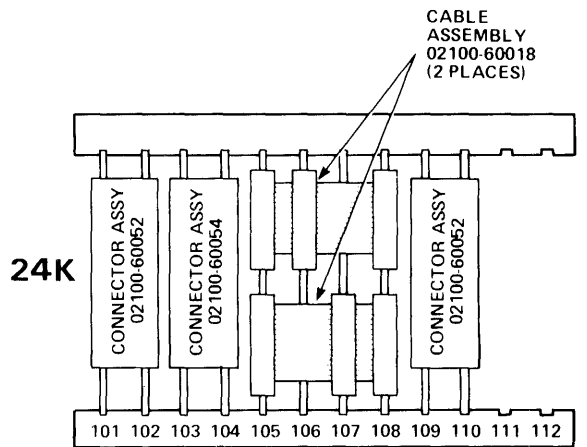
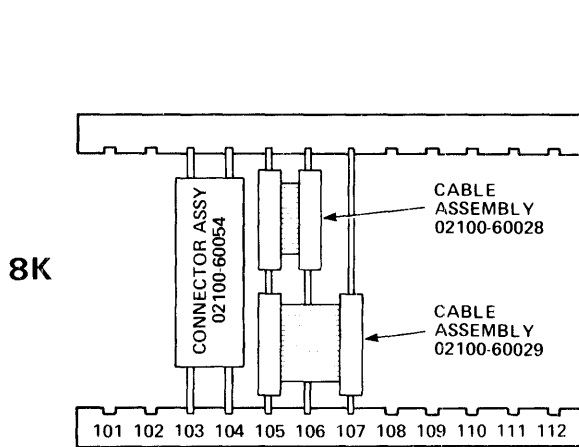
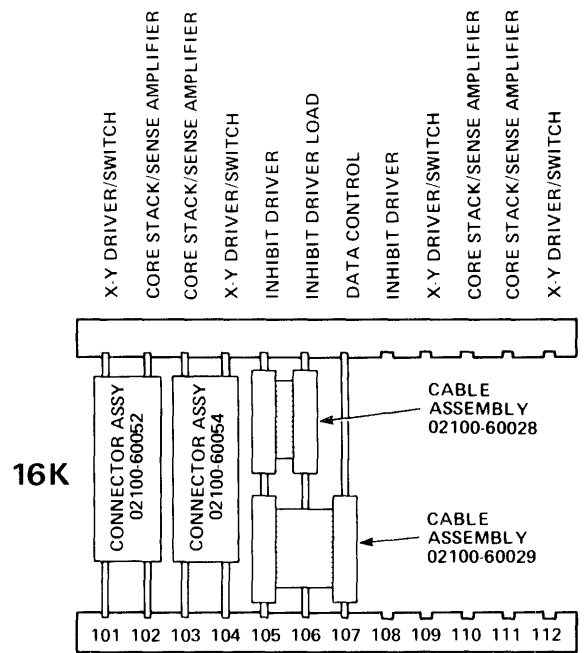
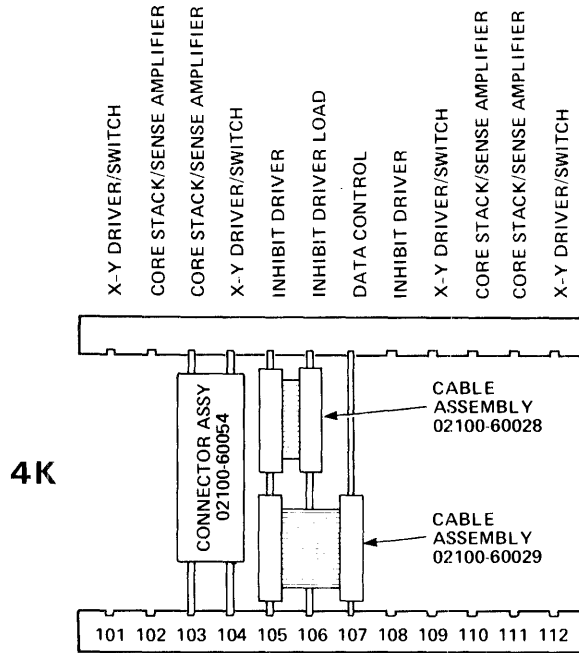
SIMPLIFIED MEMORY ADDRESSING

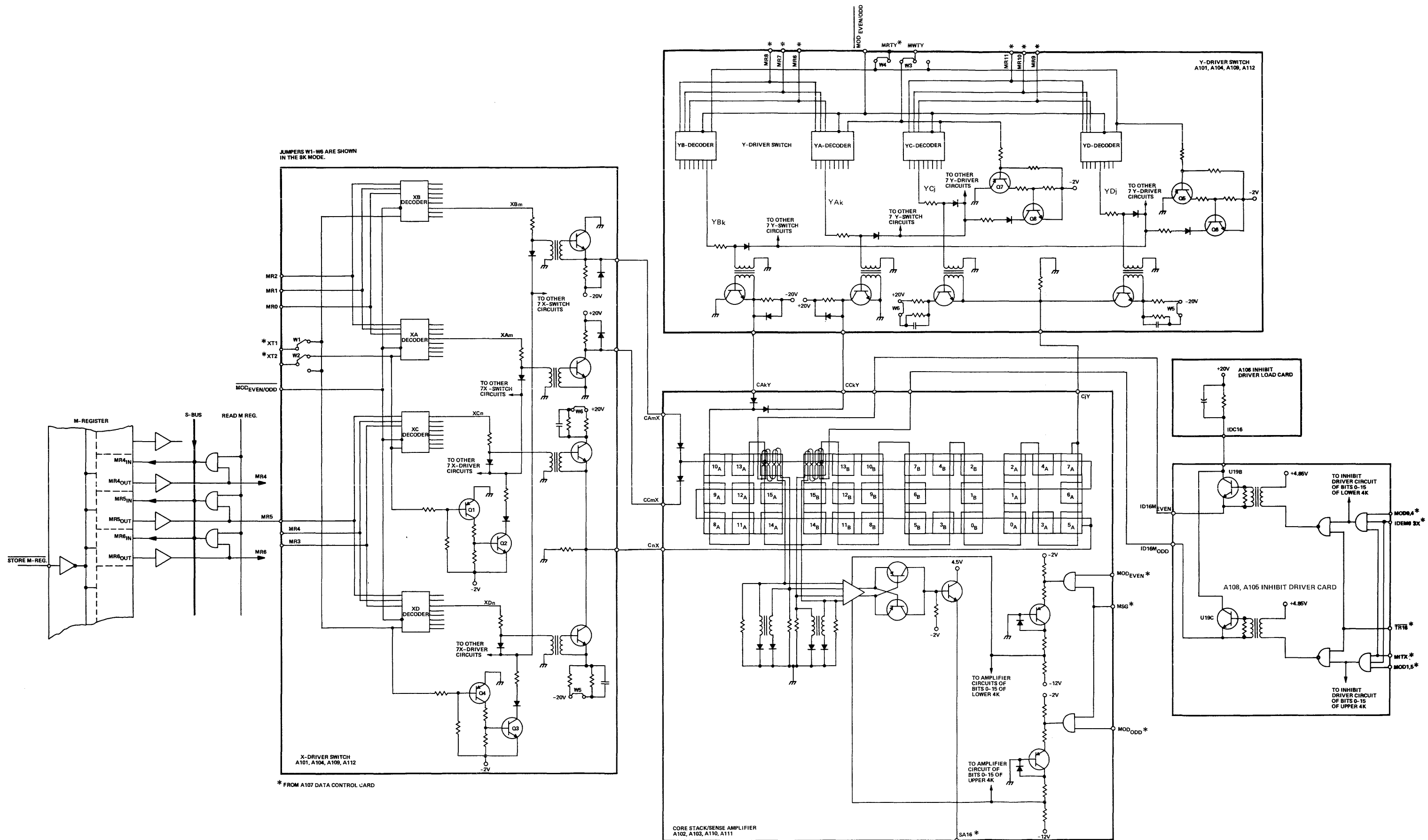


MEMORY BLOCK DIAGRAM

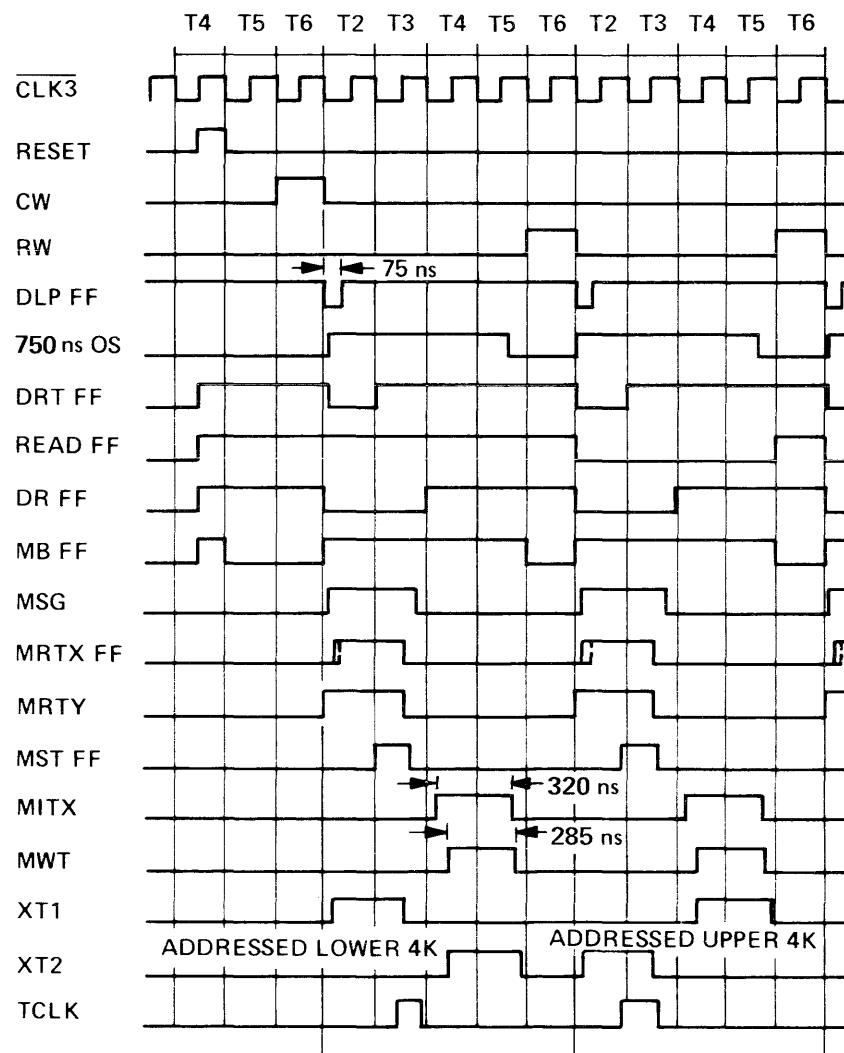


MEMORY SECTION CARD CAGE LOADING CONFIGURATIONS





HP 2100 DATA CONTROL CARD SIGNALS (A107)



power supply

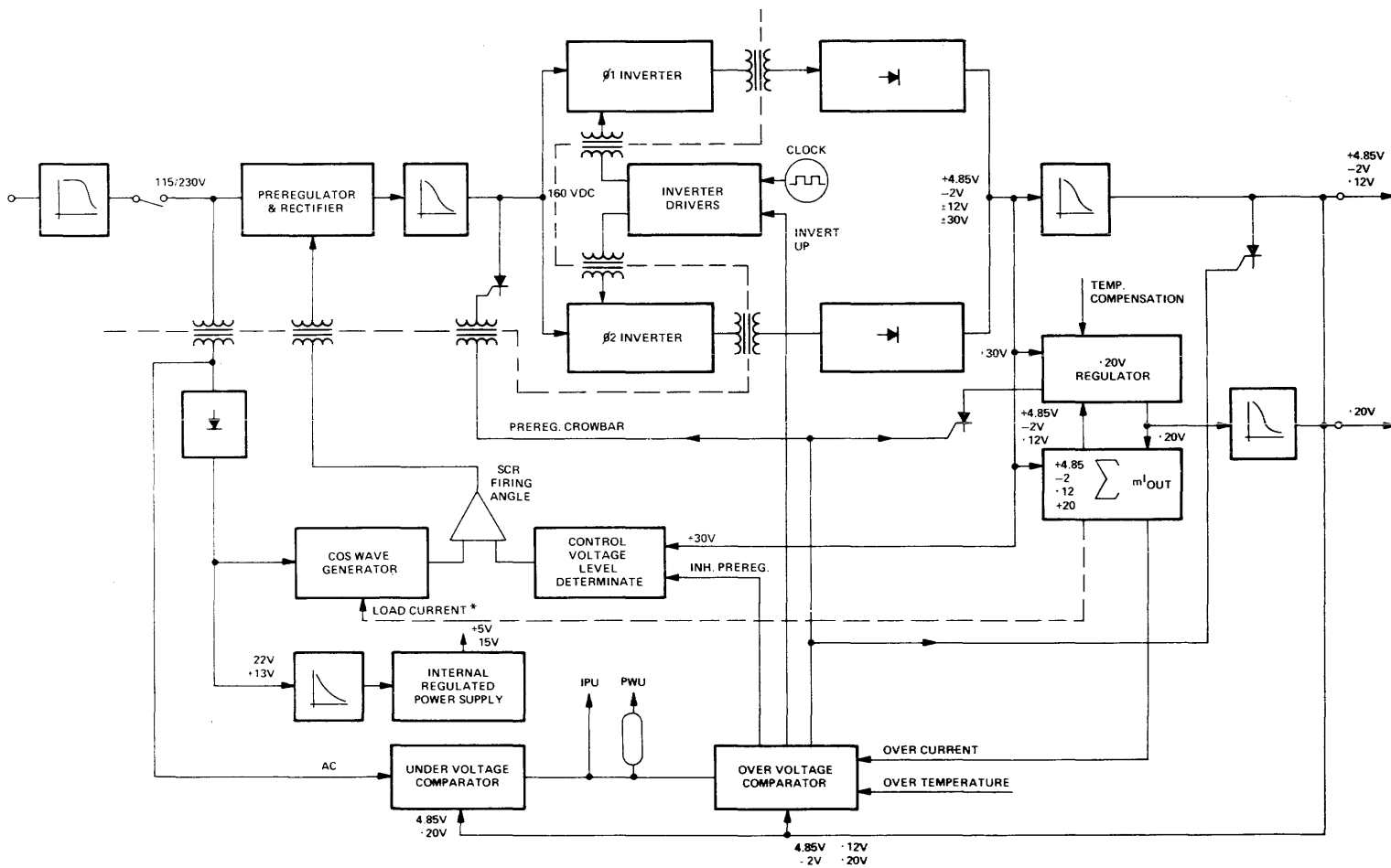
5



LESSON 5
POWER SUPPLY

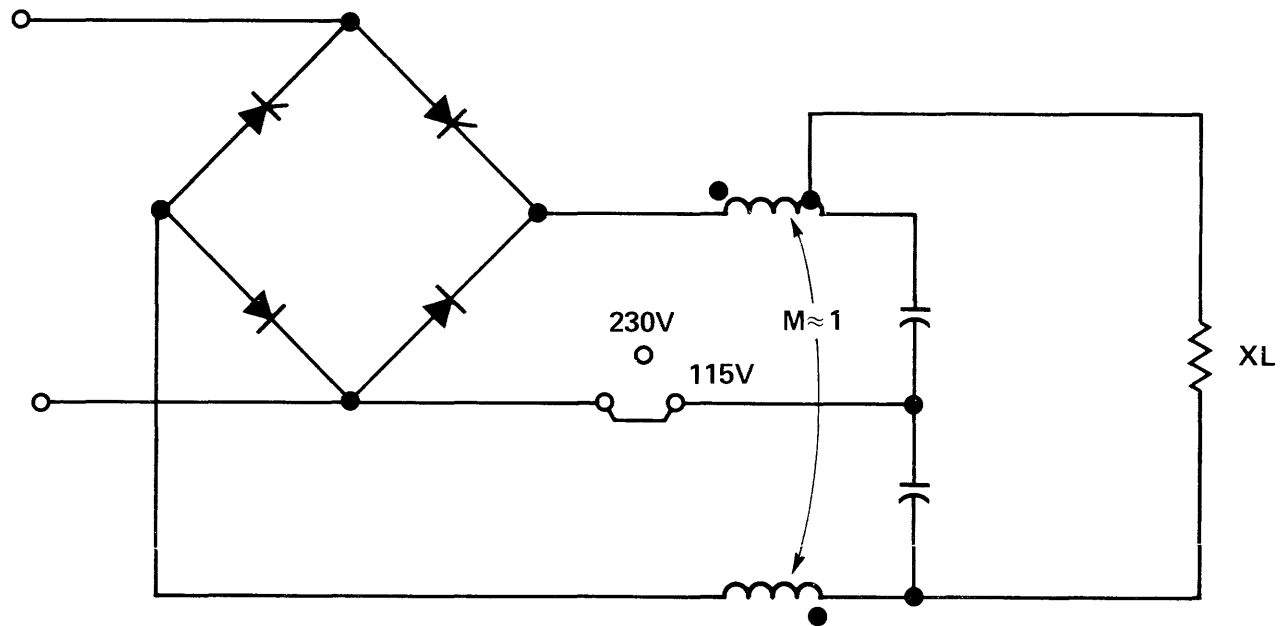
2100 Power Supply, Block Diagram	5-1	Block Diagram +20V Regulator	5-8
2100 Power Supply, Simplified Preregulator	5-2	+20V Regulator	5-9
2100 Power Supply, Preregulator	5-3	2100 Power Supply, Block Diagram	5-10
Voltages on Preregulator and 160V	5-4	Protection and Control Board	
Output Board		2100 Power Supply, Preregulator Control	5-11
2100 Power Supply, Inverter Driver	5-5	Block Diagram	
2100 Power Supply, Block Diagram of	5-6	Trigger Clamp (Regenerative Pulse	5-12
Internal Voltage Supply Regulation		Generator)	
2100 Power Supply, Internal Voltage	5-7		
Supply Regulator			

2100 POWER SUPPLY, BLOCK DIAGRAM

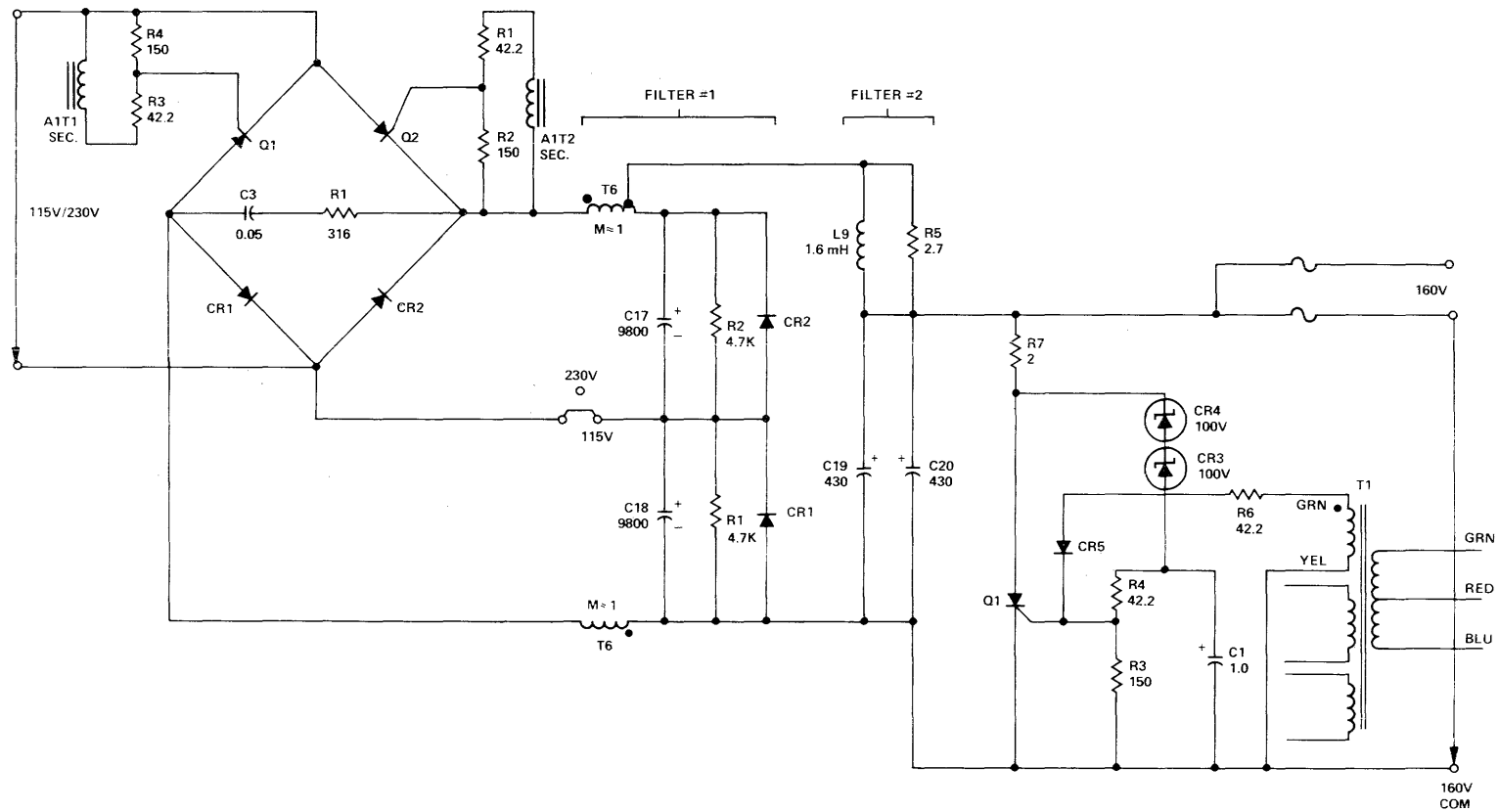


* LOAD CURRENT FEEDBACK TO
COS WAVE GENERATOR IN POWER
SUPPLIES PRIOR TO REV. 1240.

2100 POWER SUPPLY, SIMPLIFIED PREREGULATOR

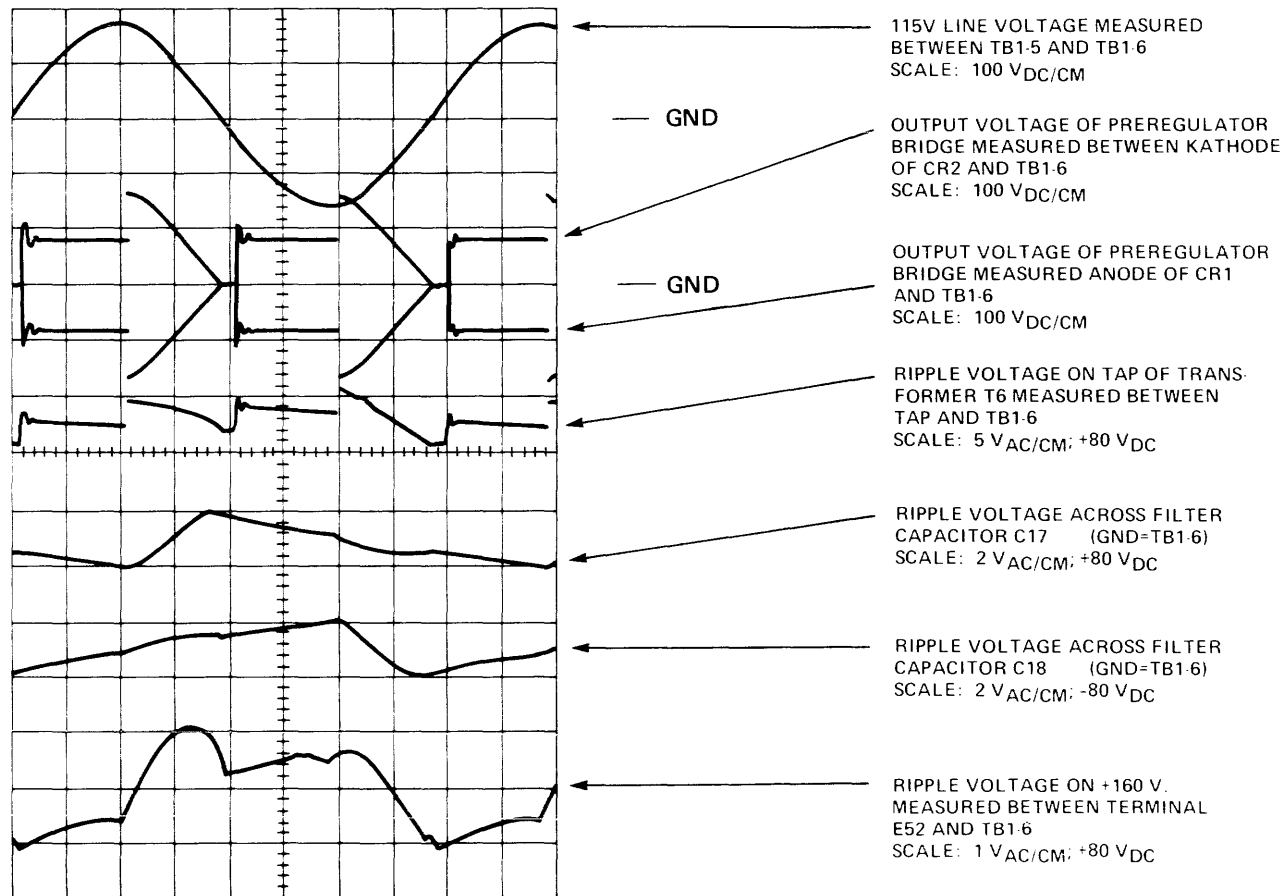


2100 POWER SUPPLY, PREREGULATOR

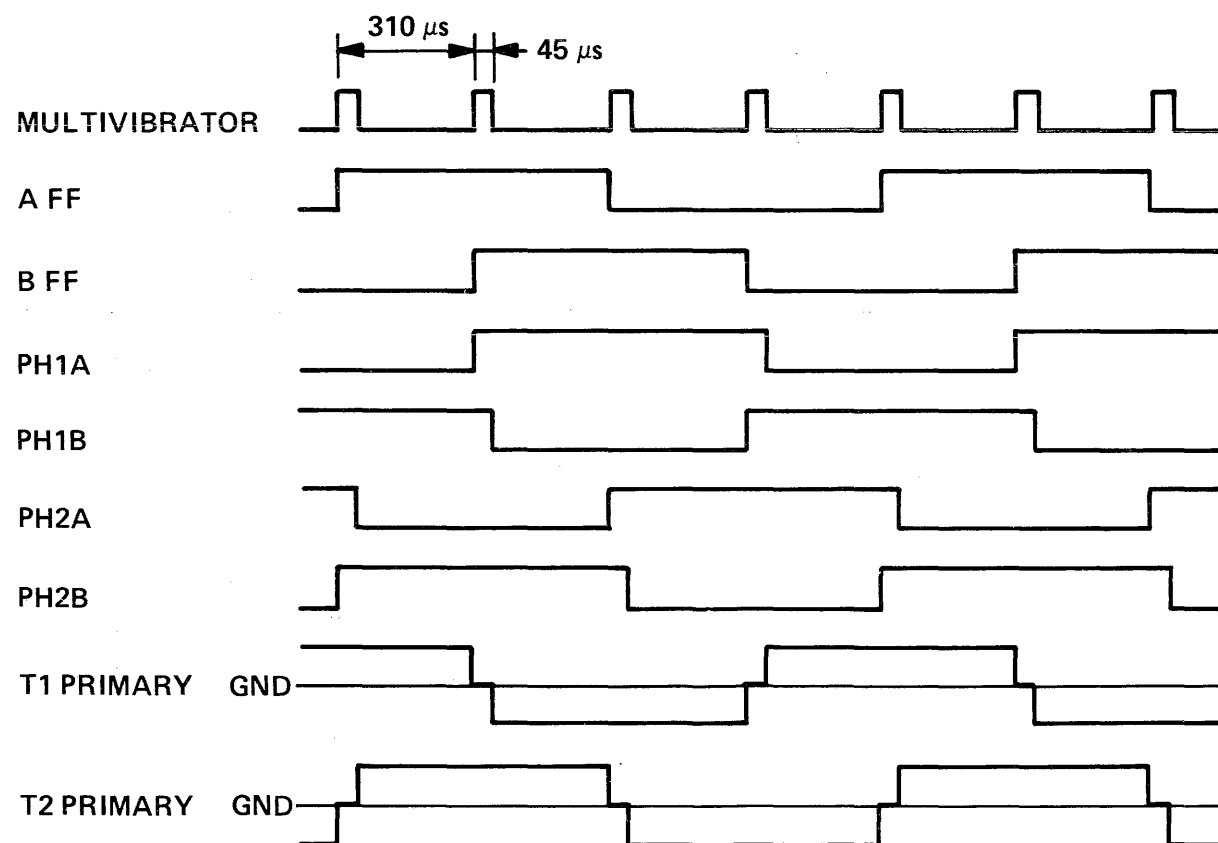


VOLTAGES ON PREREGULATOR & 160V OUTPUT BOARD

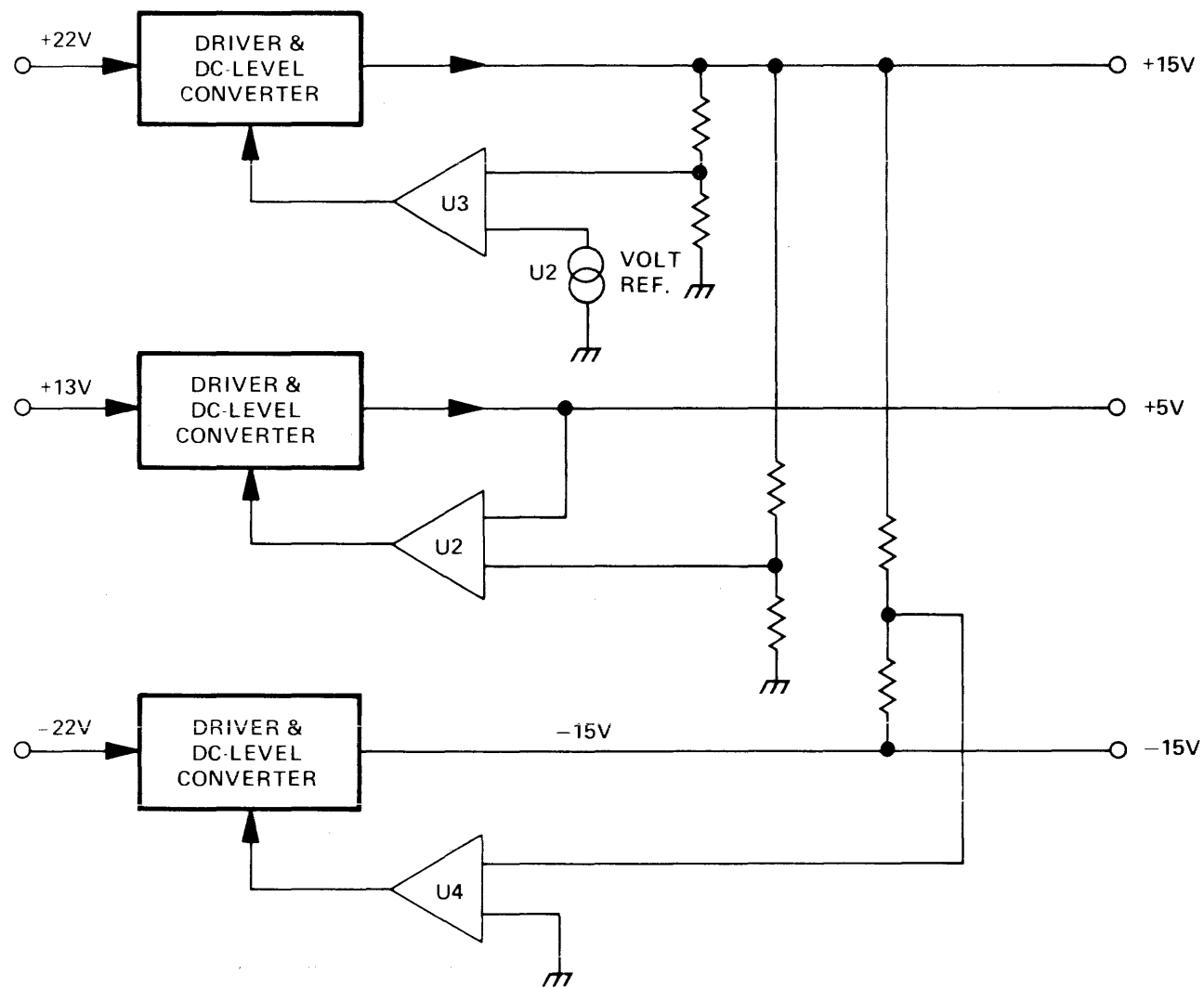
TIME SCALE: 2 ns/cm



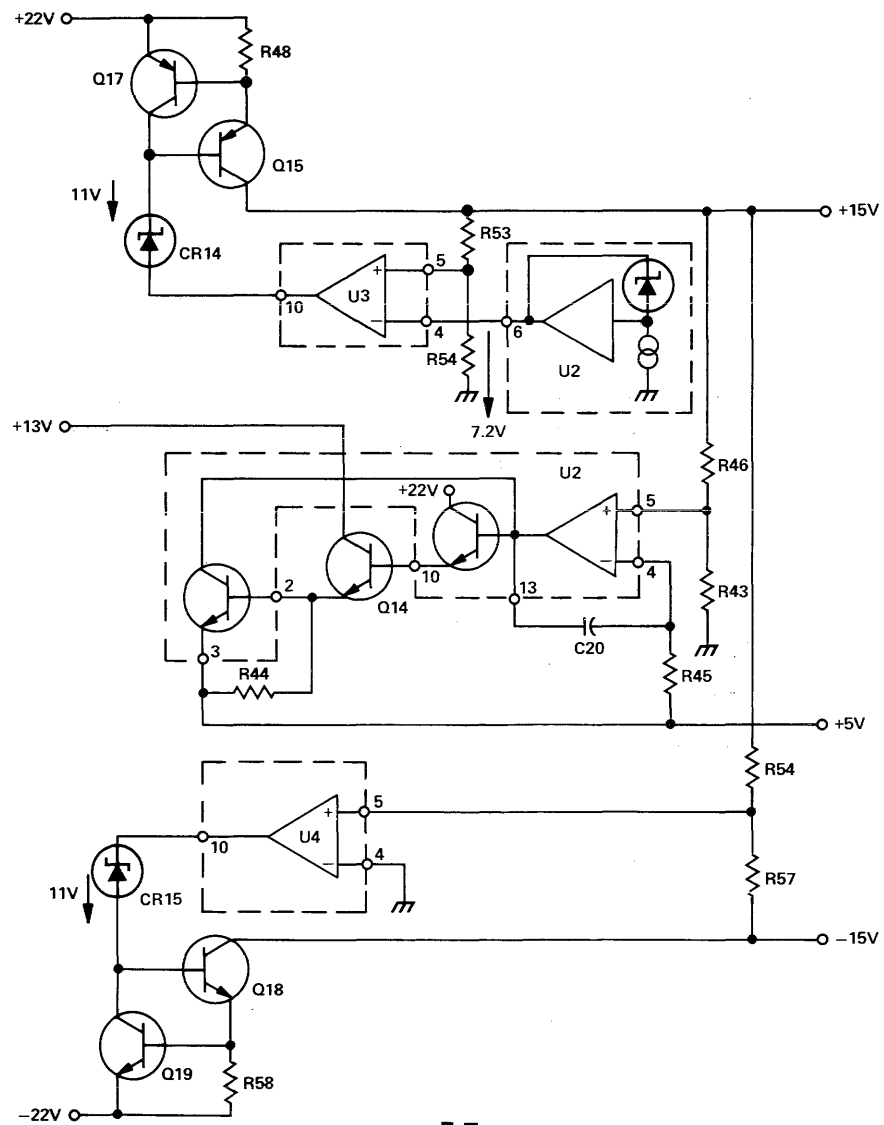
2100 POWER SUPPLY, INVERTER DRIVER



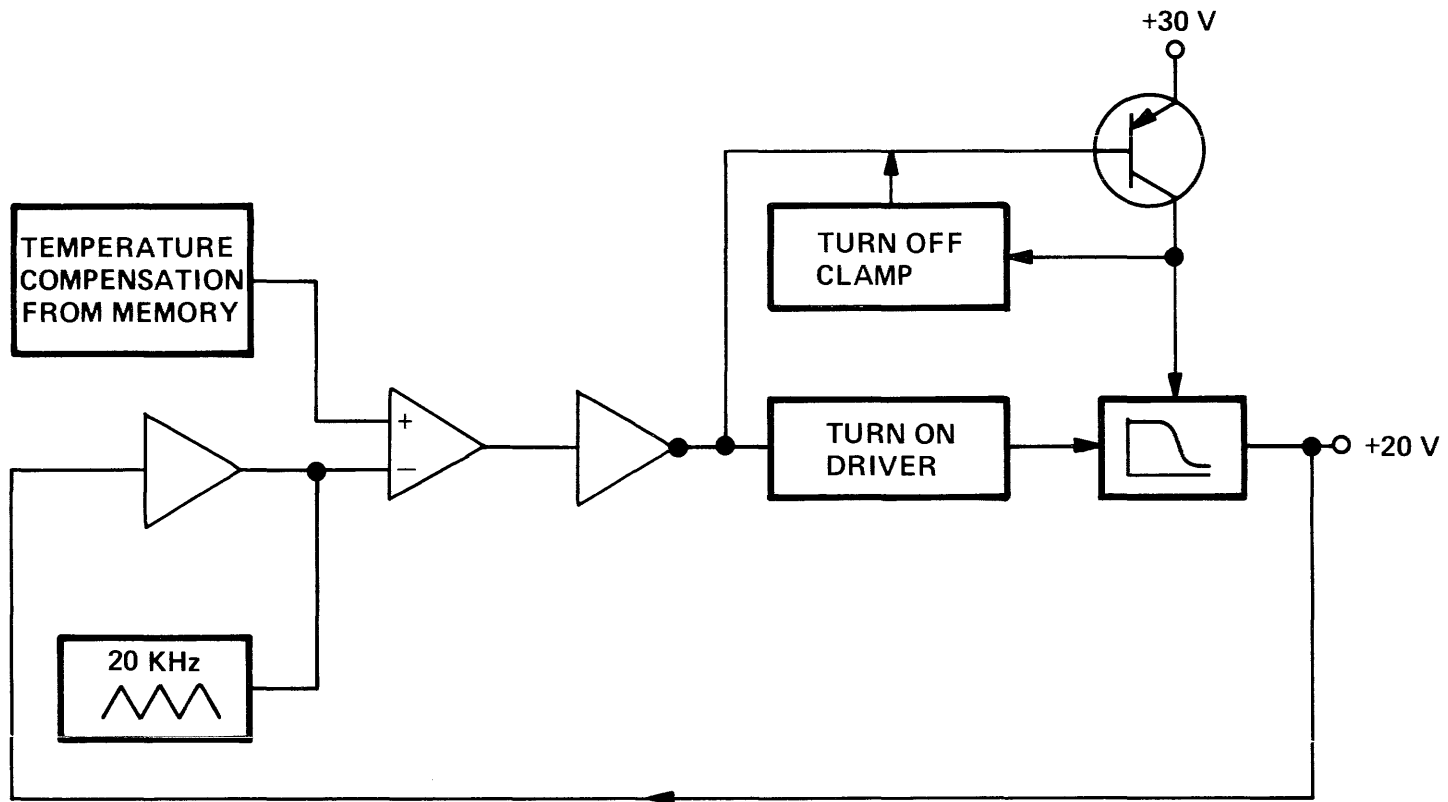
2100 POWER SUPPLY, BLOCK DIAGRAM OF INTERNAL VOLTAGE SUPPLY REGULATION



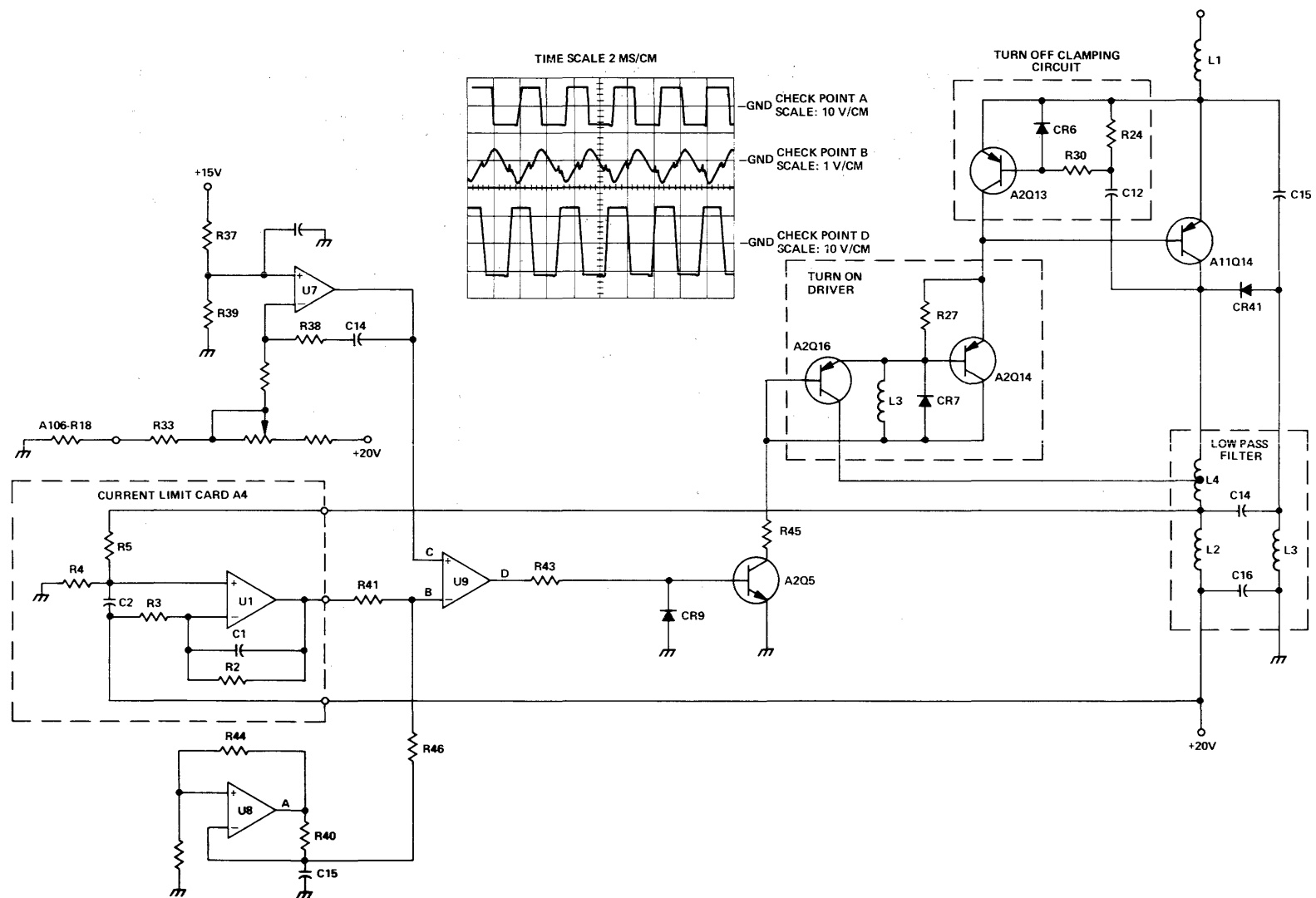
2100 POWER SUPPLY INTERNAL VOLTAGE SUPPLY REGULATOR



BLOCK DIAGRAM +20V REGULATOR

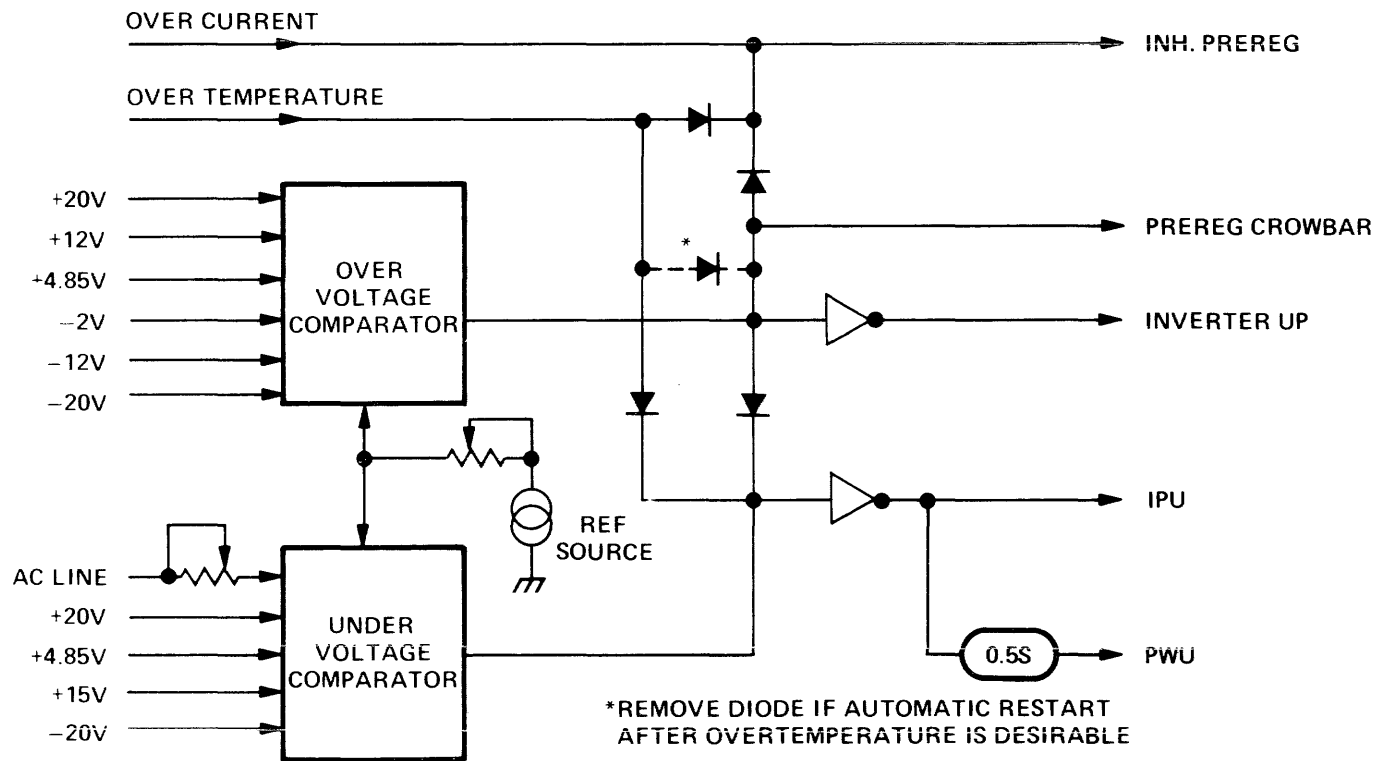


+20V REGULATOR

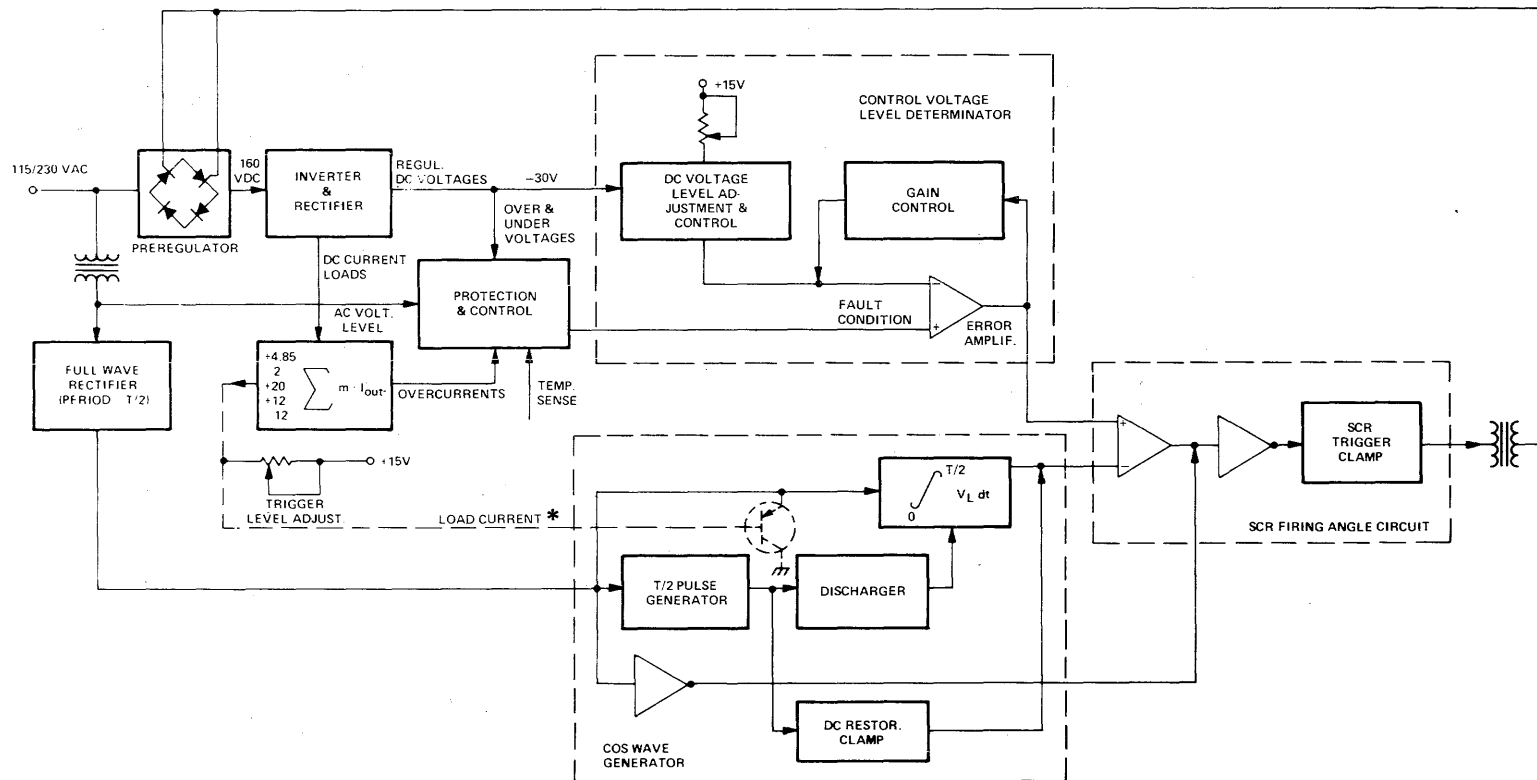


2100 POWER SUPPLY, BLOCK DIAGRAM

PROTECTION & CONTROL BOARD



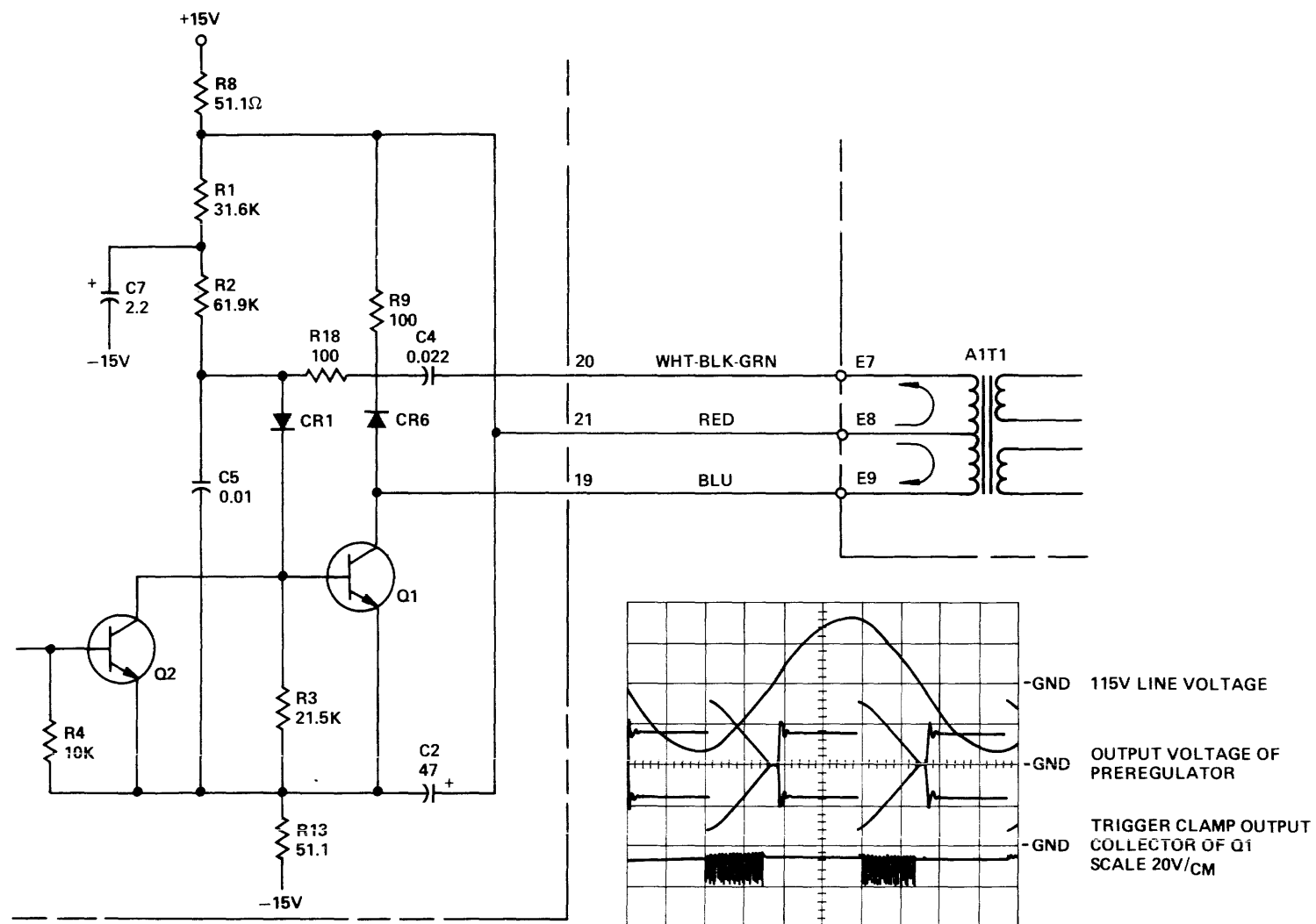
2100 POWER SUPPLY, PREREGULATOR CONTROL BLOCK DIAGRAM



* LOAD CURRENT FEEDBACK TO
COS WAVE GENERATOR IN POWER
SUPPLIES PRIOR TO REV. 1240.

TRIGGER CLAMP

(REGENERATIVE PULSE GENERATOR)



input/output and direct memory access

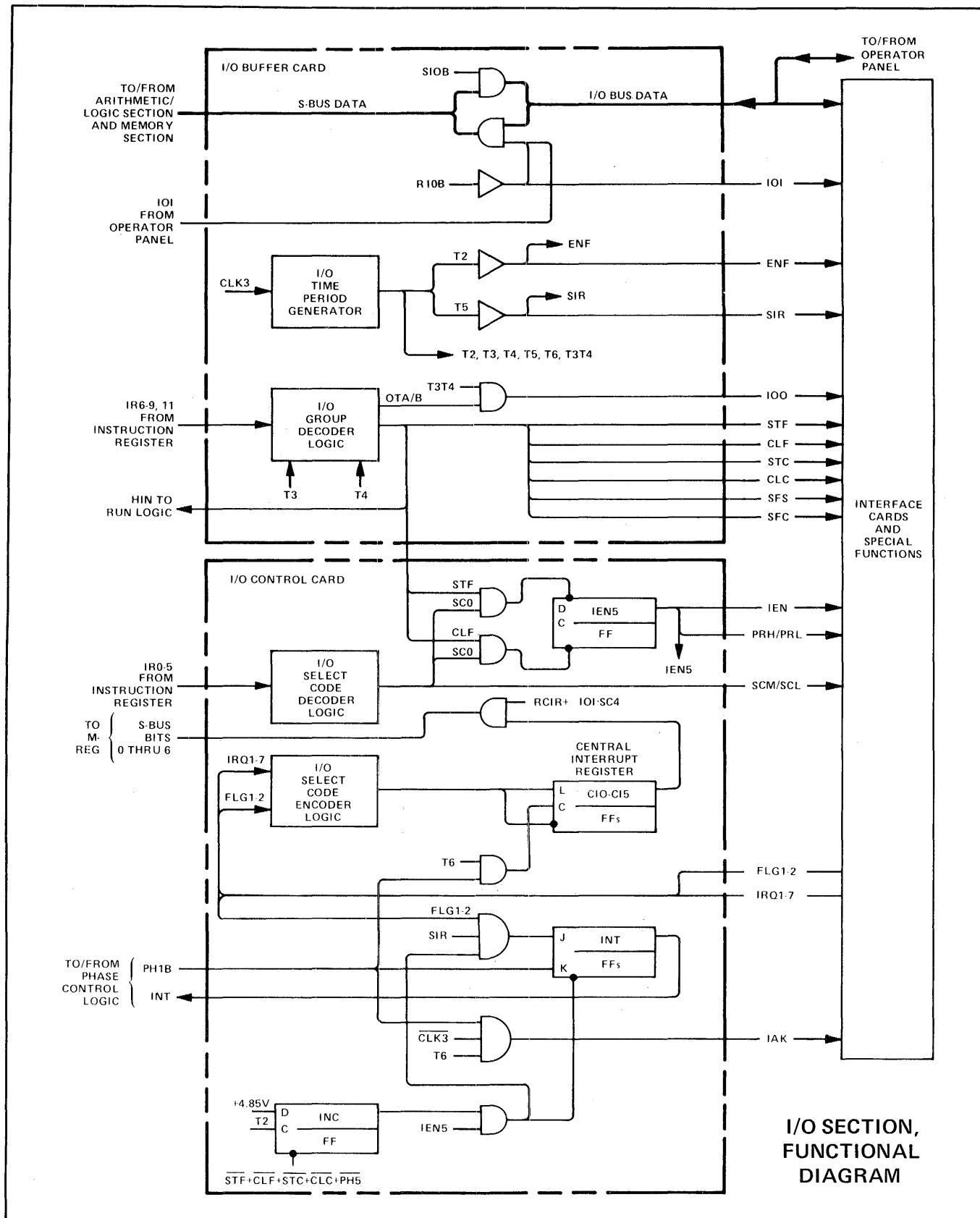
6

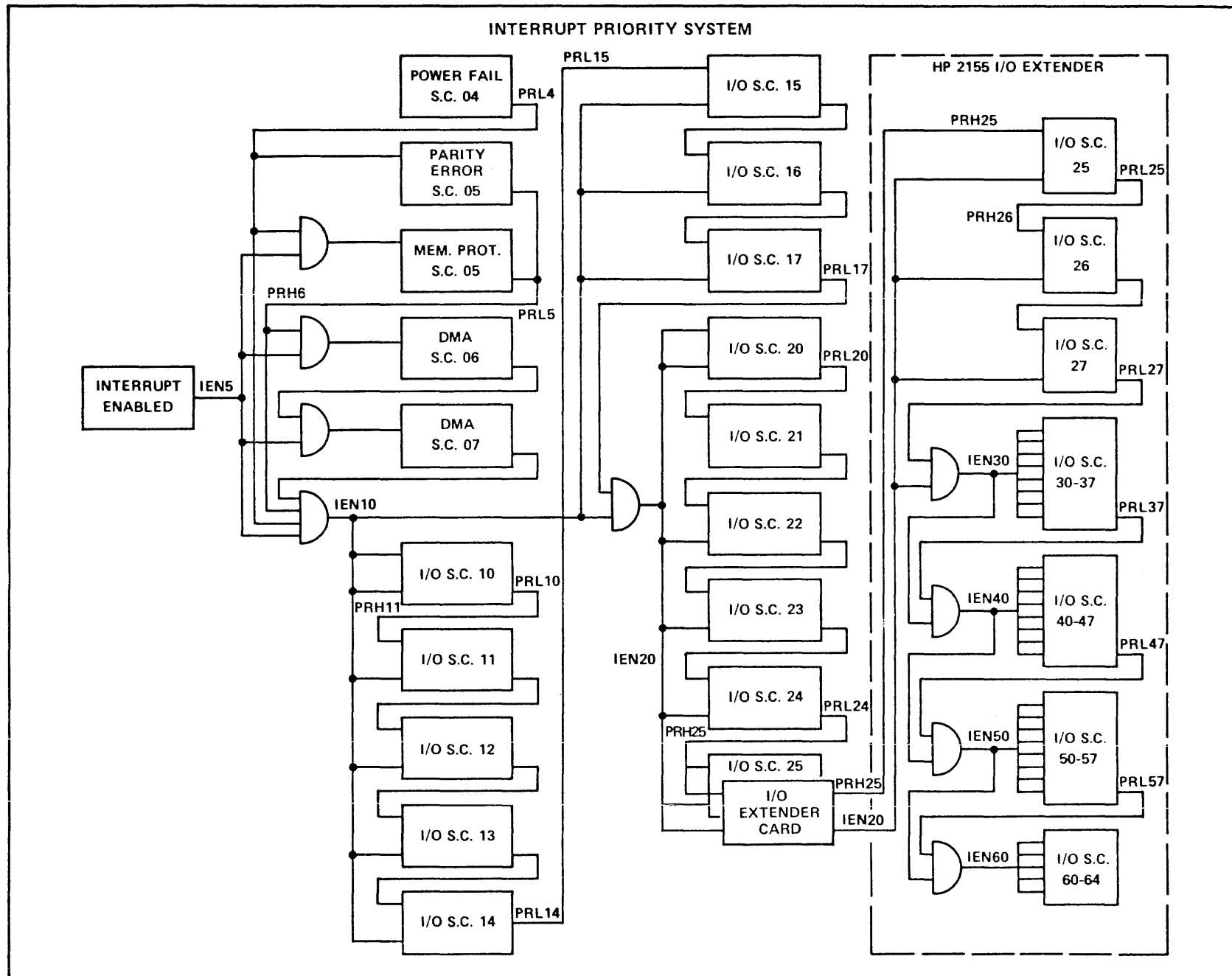


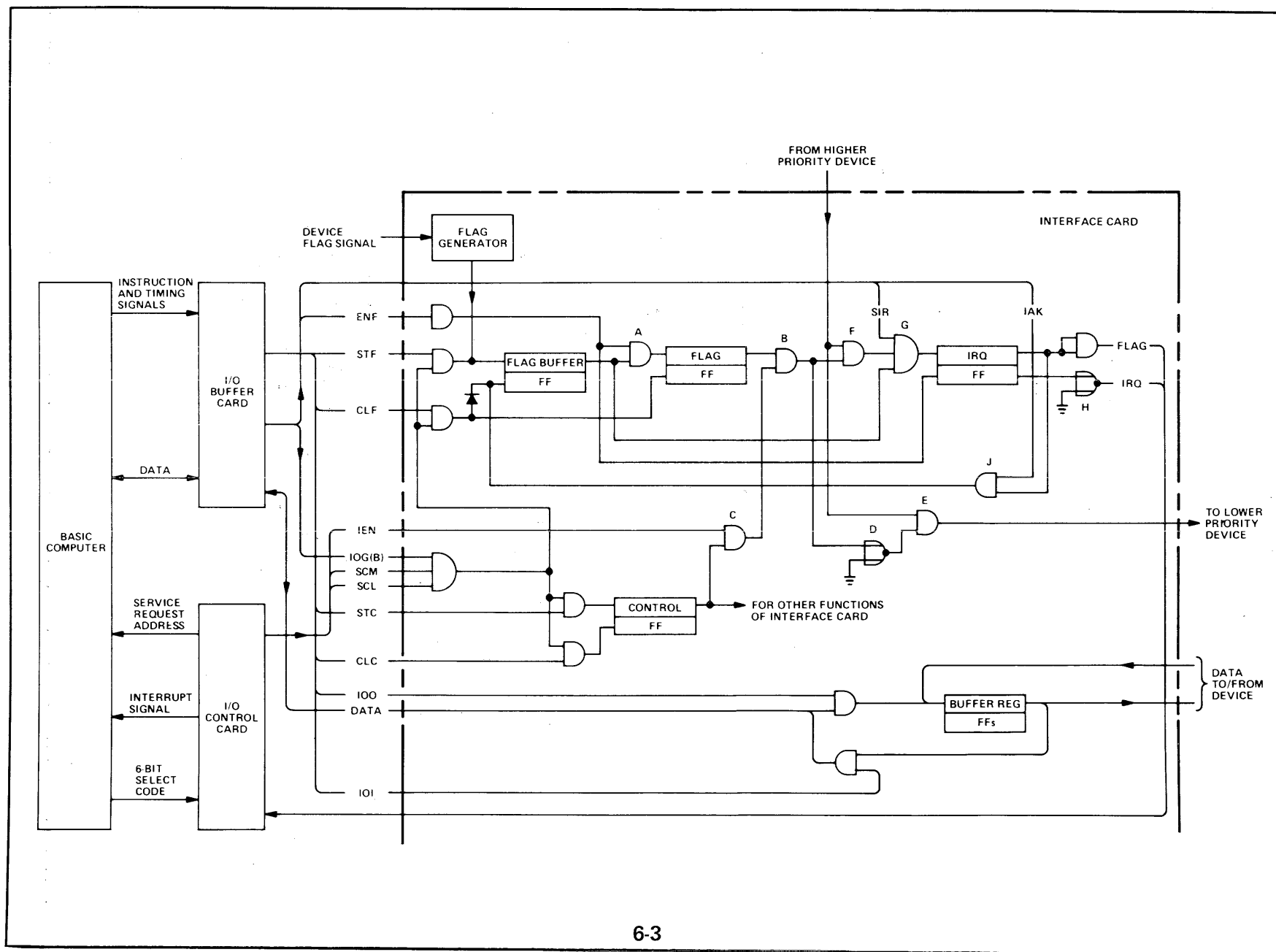
LESSON 6

INPUT/OUTPUT AND DIRECT MEMORY ACCESS

I/O Section, Functional Diagram	6-1	Simplified Memory Protect	6-8
Interrupt Priority System	6-2	Direct Memory Access	6-9
Typical I/O Interface Card	6-3	Which Parameters Have To Be Specified	6-10
Block Diagram of I/O Interrupt Priority	6-4	In The Initialization Routine?	
Interrupt Sequence From IRQ To PH1B	6-5	DMA Initialization Parameters	6-11
2100 I/O Interrupt System Timing	6-6	Program To Initialize DMA	6-12
Timing Diagram For Power Fail W. Autom. Restart	6-7	Simplified DMA Logic	6-13

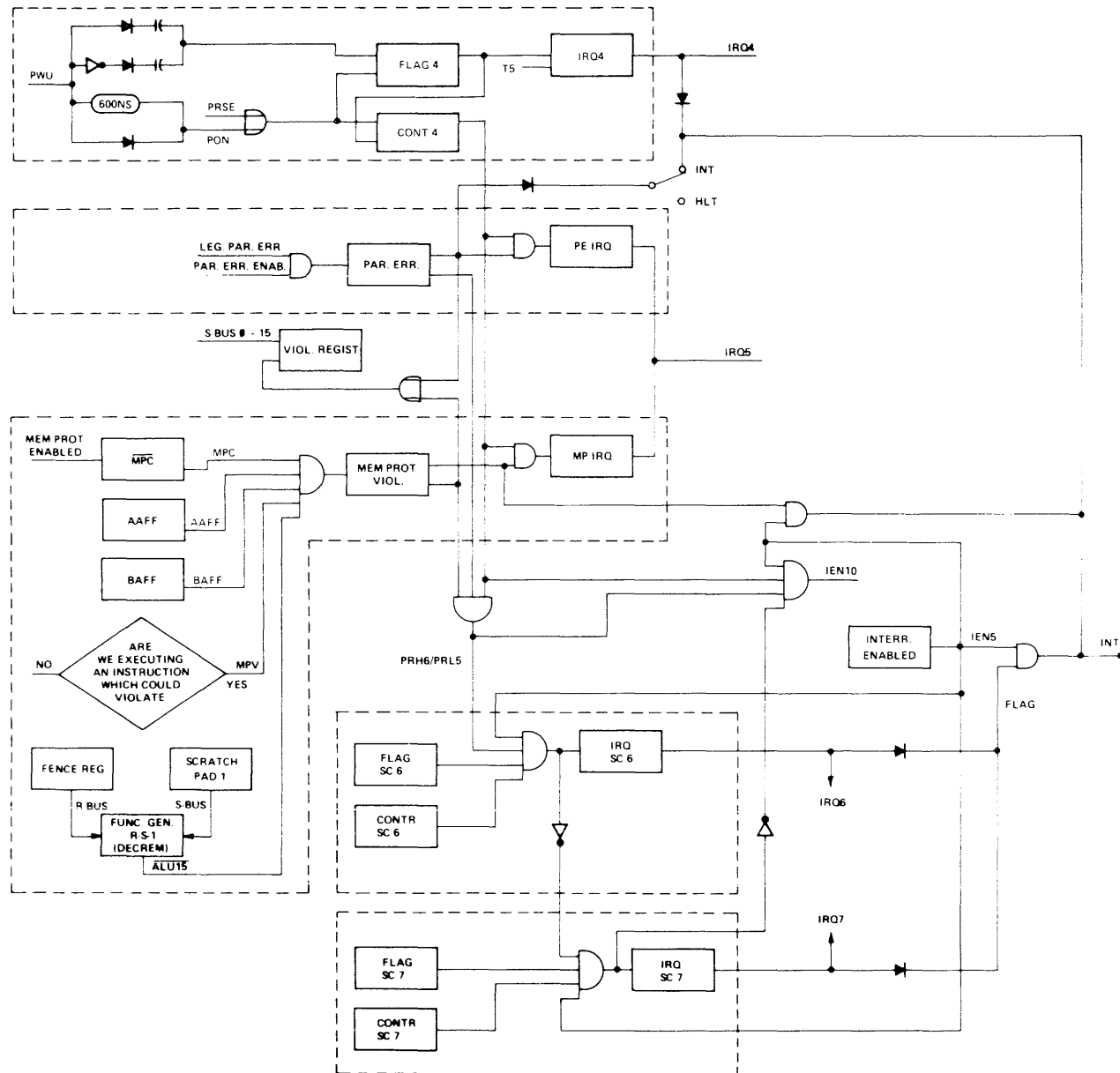




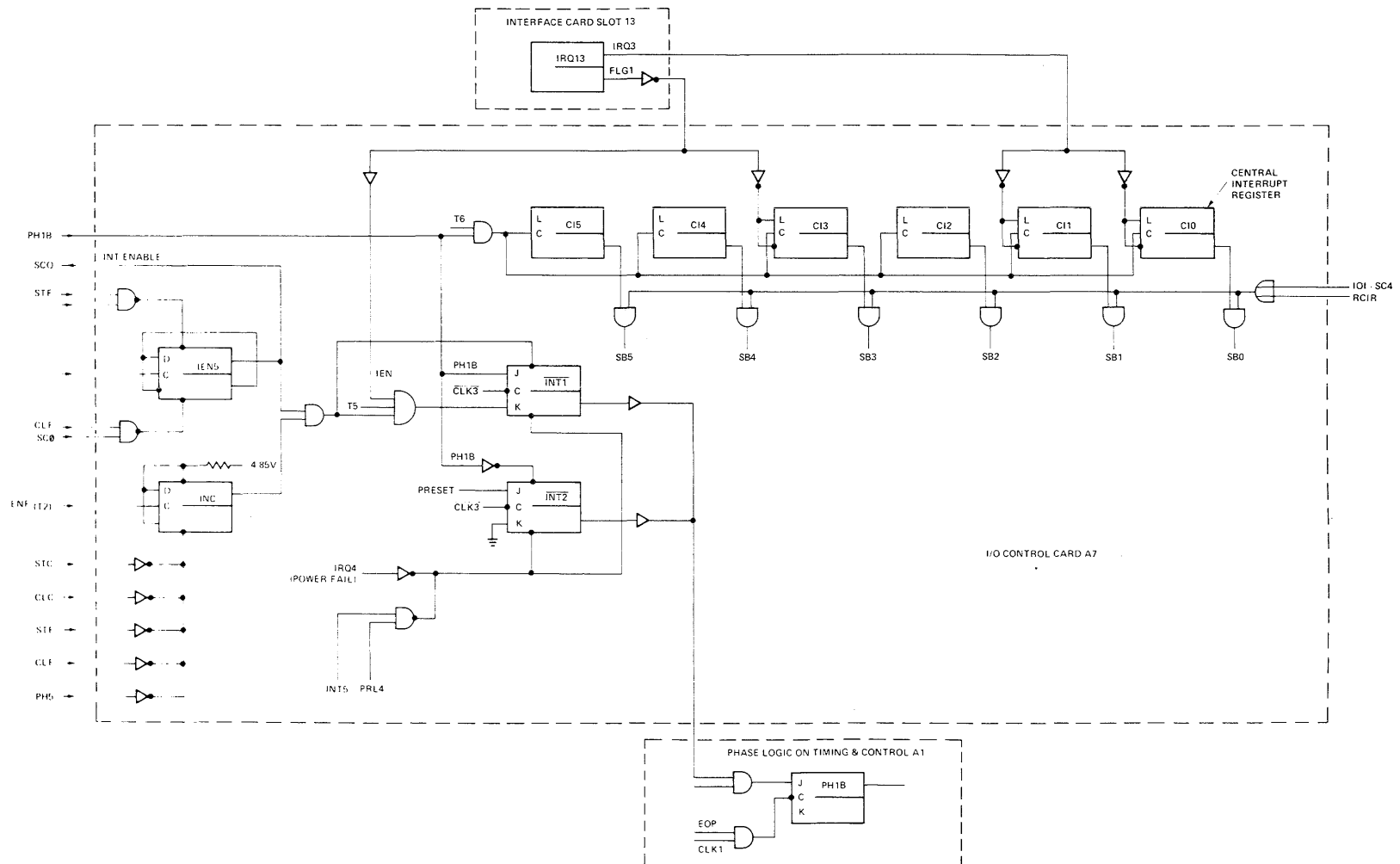


TYPICAL I/O INTERFACE CARD

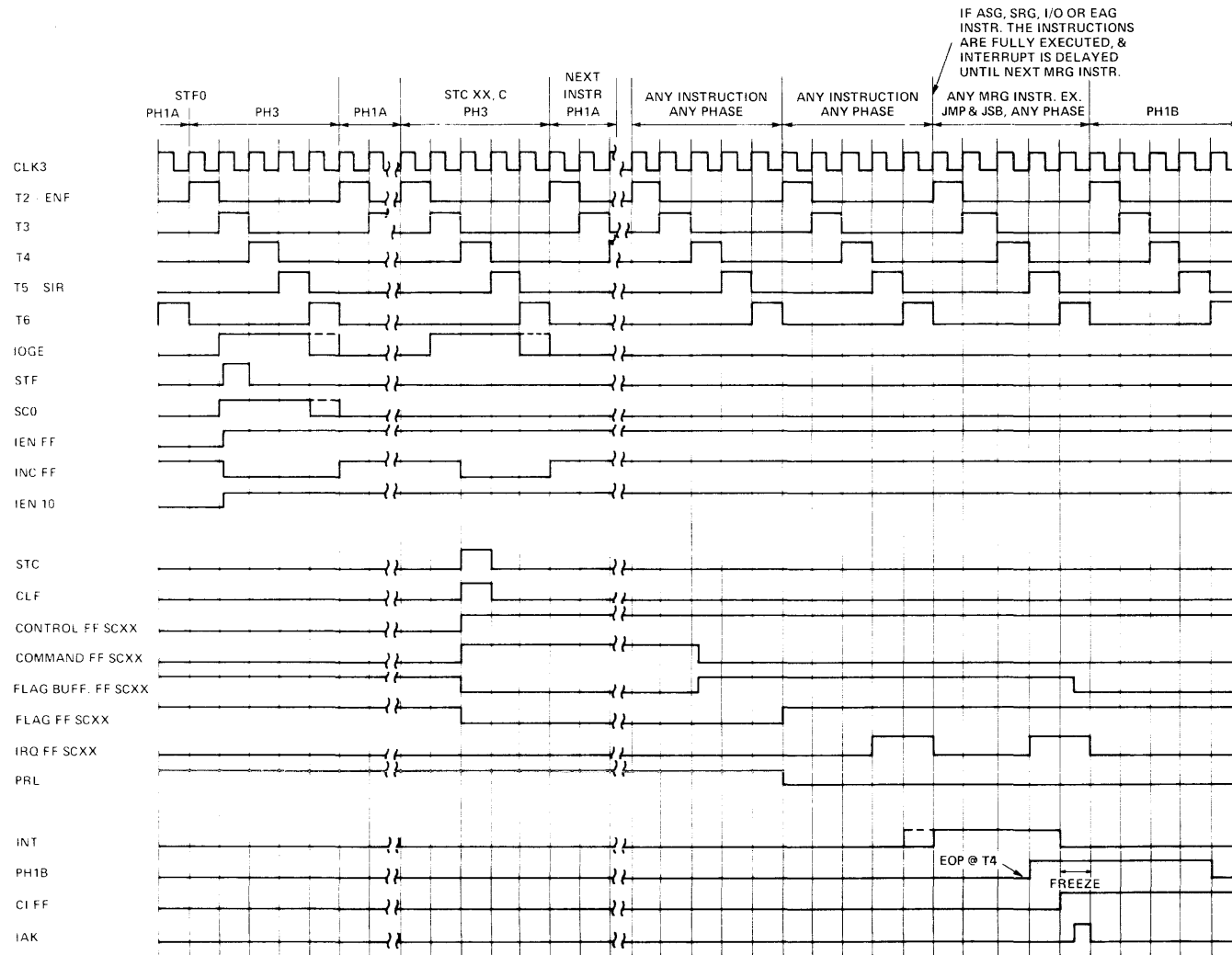
BLOCK DIAGRAM OF I/O INTERRUPT PRIORITY



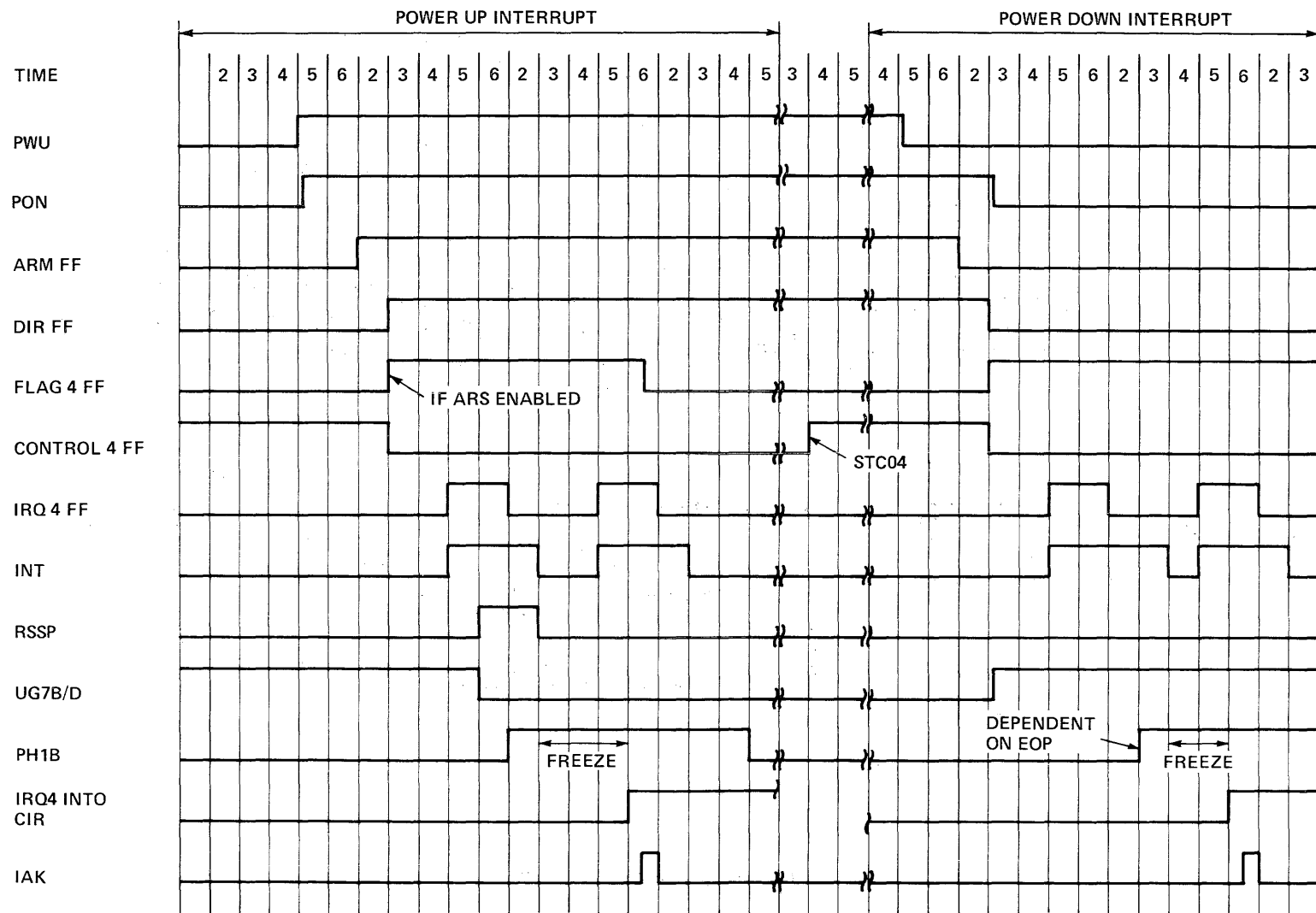
INTERRUPT SEQUENCE FROM IRQ TO PH1B



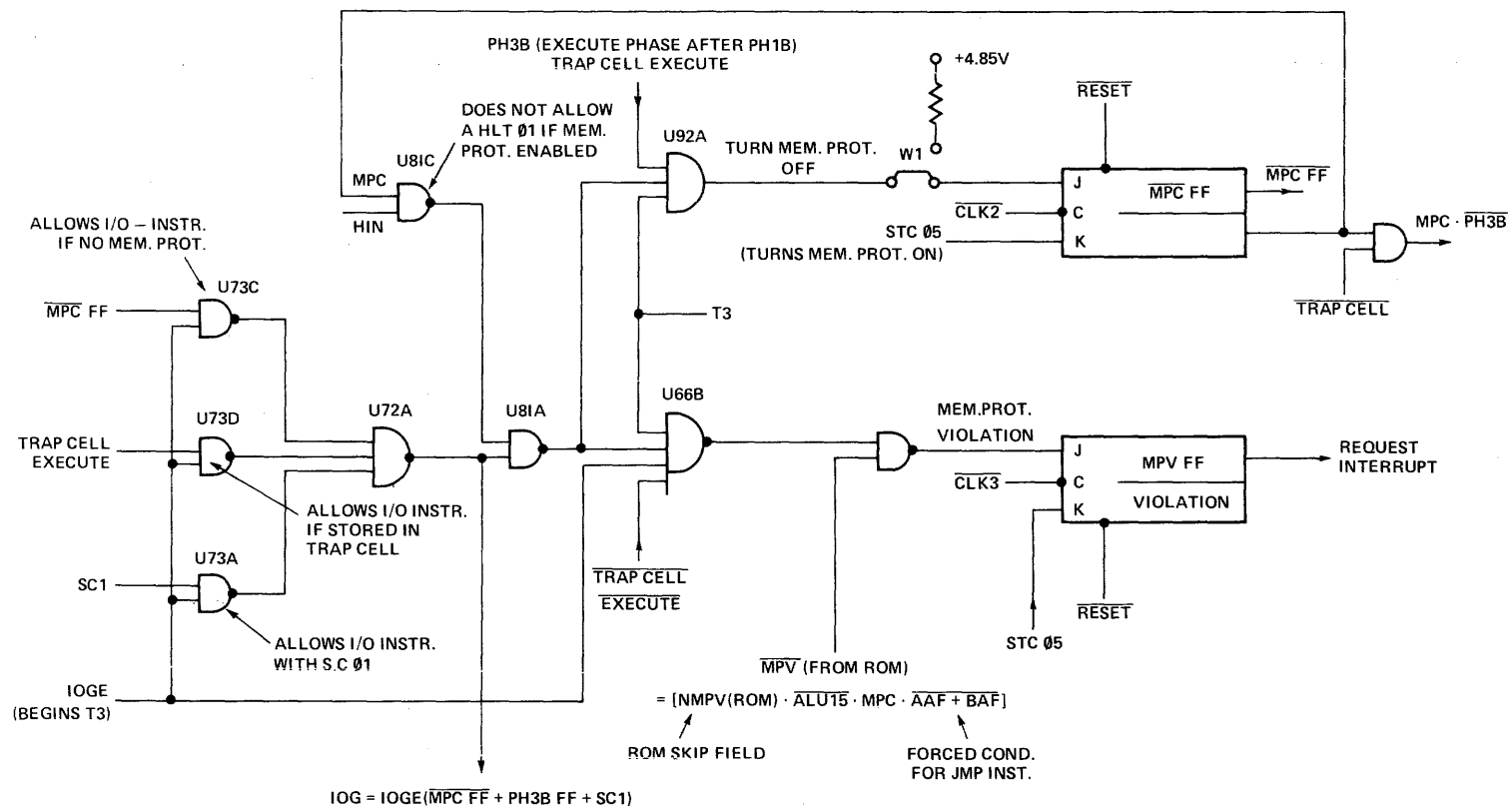
2100 I/O INTERRUPT SYSTEM TIMING



TIMING DIAGRAM FOR POWER FAIL W. AUTOM. RESTART



SIMPLIFIED MEMORY PROTECT



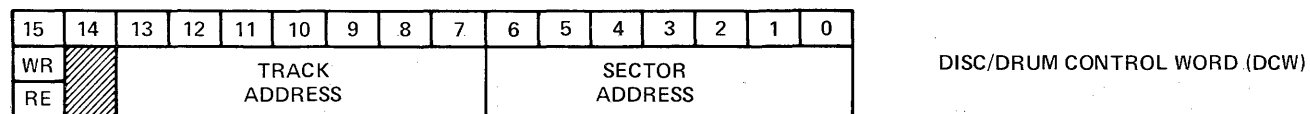
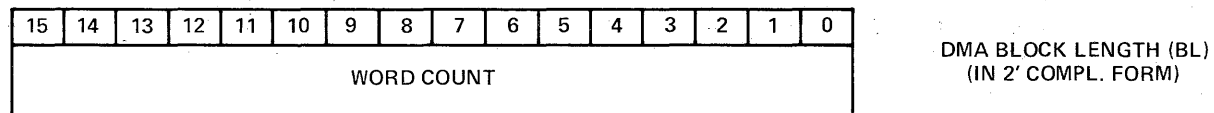
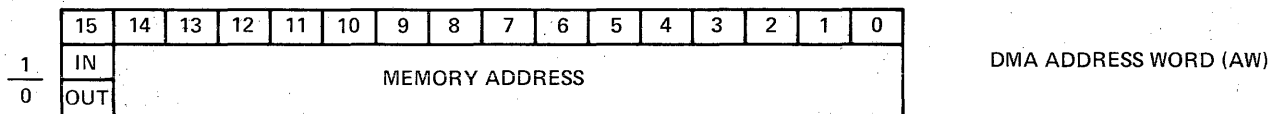
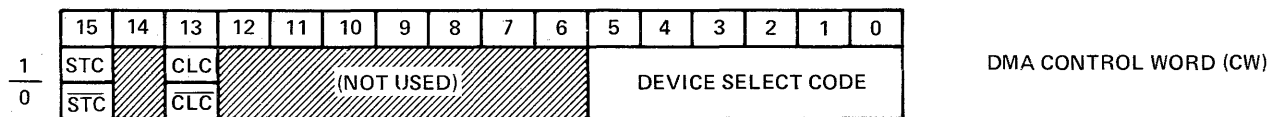
DIRECT MEMORY ACCESS

- **DMA PROVIDES A DIRECT TRANSFER OF AN ENTIRE DATA BLOCK FROM MEMORY TO A PREASSIGNED PERIPHERAL (OUTPUT) OR PERIPHERAL TO MEMORY (INPUT).**
- **DMA ACCOMPLISHES A WORD TRANSFER BY STEALING A MEMORY CYCLE (IMPOSES A PH5) FROM THE CPU IF THE PERIPHERAL VIA ITS INTERFACE IS READY TO PARTICIPATE IN SUCH A TRANSFER.**
- **DMA INTERRUPTS ONLY AT THE END OF AN ENTIRE DATA BLOCK TRANSFER IF INTERRUPT IS ENABLED.**
- **THE TWO DMA-CHANNELS CARRY SC.06 AND SC.07.**
- **BEFORE A DMA TRANSFER TAKES PLACE IT HAS TO BE INITIALIZED BY A SPECIAL ROUTINE.**

WHICH PARAMETERS HAVE TO BE SPECIFIED IN THE INITIALIZATION ROUTINE?

- | | | |
|--|---|---------------------------------|
| 1. SELECT CODE OF PERIPHERAL TO BE SERVICED BY DMA. | } | DMA CONTROL WORD
(CW) |
| 2. SHALL WE HAVE A CLC ON THE DEVICE I/O CHANNEL AFTER THE LAST WORD OF THE BLOCK HAS BEEN TRANSFERRED? | | |
| 3. SHALL WE HAVE A STC ON THE DEVICE I/O CHANNEL AFTER EACH WORD IN THE DATA BLOCK HAS BEEN TRANSFERRED (REINITIATE PERIPHERAL)? | | |
| 4. MEMORY STARTING ADDRESS OF FIRST WORD TO BE TRANSFERRED. | } | DMA ADDRESS WORD
(AW) |
| 5. DIRECTION OF DATA TRANSFER. | | |
| 6. BLOCK LENGTH. | } | DMA BLOCK LENGTH WORD
(BL) |
| 7. FOR RANDOM ACCESS MEMORIES (DISC/DRUM): DIRECTION, TRACK- AND SECTOR ADDRESS TO DISC CONTROL INTERFACE BOARD. | } | DISC/DRUM CONTROL WORD
(DCW) |

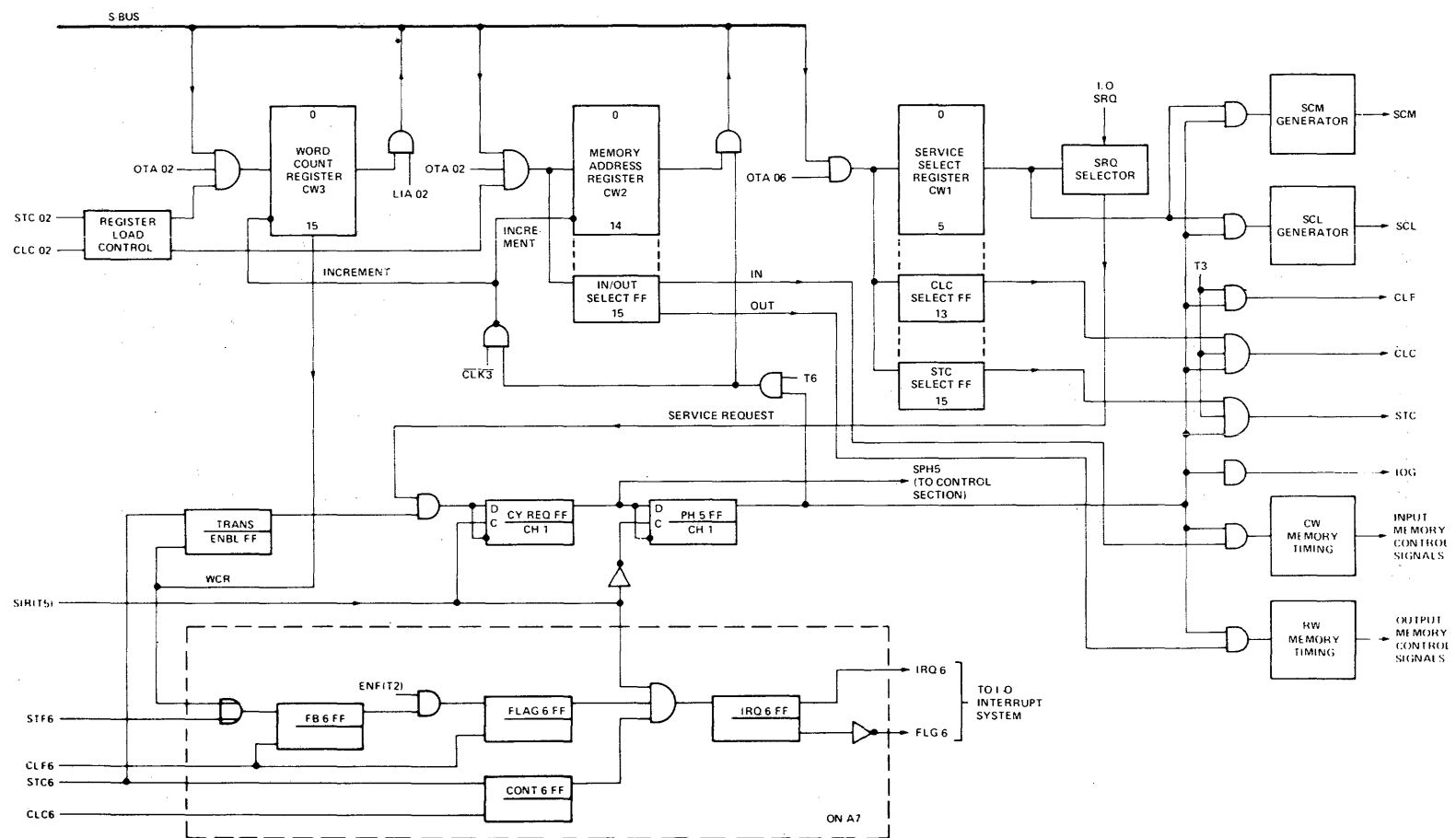
DMA INITIALIZATION PARAMETERS



PROGRAM TO INITIALIZE DMA

LABEL	OPCODE	OPERAND	COMMENTS
INIT	LDA	CW	FETCHES CONTROL WORD (CW) FROM MEMORY AND LOADS IT IN A-REGISTER.
	OTA	6	OUTPUTS CW TO DMA CHANNEL 1.
	CLC	2	PREPARES MEMORY ADDRESS REGISTER TO RECEIVE ADDRESS WORD (AW).
	LDA	AW	FETCHES AW FROM MEMORY AND LOADS IT IN A-REGISTER.
	OTA	2	OUTPUTS AW TO DMA CHANNEL 1.
	STC	2	PREPARES WORD COUNT REGISTER TO RECEIVE BLOCK LENGTH WORD (BL).
	LDA	BL	FETCHES BL FROM MEMORY AND LOADS IT IN A-REGISTER.
	OTA	2	OUTPUTS BL TO DMA CHANNEL 1.
STRT	STC	10B,C	START INPUT DEVICE.
	STC	6B,C	ACTIVATE DMA CHANNEL 1.
	SFS	6	WAIT WHILE DATA TRANSFER TAKES PLACE OR, IF
	JMP	*-1	INTERRUPT PROCESSING IS USED, CONTINUE PROGRAM.
.	.	.	
.	.	.	
.	.	.	
	HLT		HALT
CW	OCT	120010	ASSIGNMENT FOR DMA CHANNEL 1; SPECIFIES I/O CHANNEL SELECT CODE ADDRESS (10 _g), STC AFTER EACH WORD IS TRANSFERRED, AND CLC AFTER FINAL WORD IS TRANSFERRED.
AW	OCT	100200	MEMORY ADDRESS REGISTER CONTROL. DMA CHANNEL 1 SPECIFIES MEMORY INPUT OPERATION AND STARTING MEMORY ADDRESS (200 _g).
BL	DEC	-50	WORD COUNT REGISTER CONTROL. DMA CHANNEL 1 (WCR1); SPECIFIES THE 2'S COMPLEMENT OF THE NUMBER OF CHARACTER WORDS IN THE BLOCK OF DATA TO BE TRANSFERRED (50 ₁₀).

SIMPLIFIED DMA LOGIC



front panel

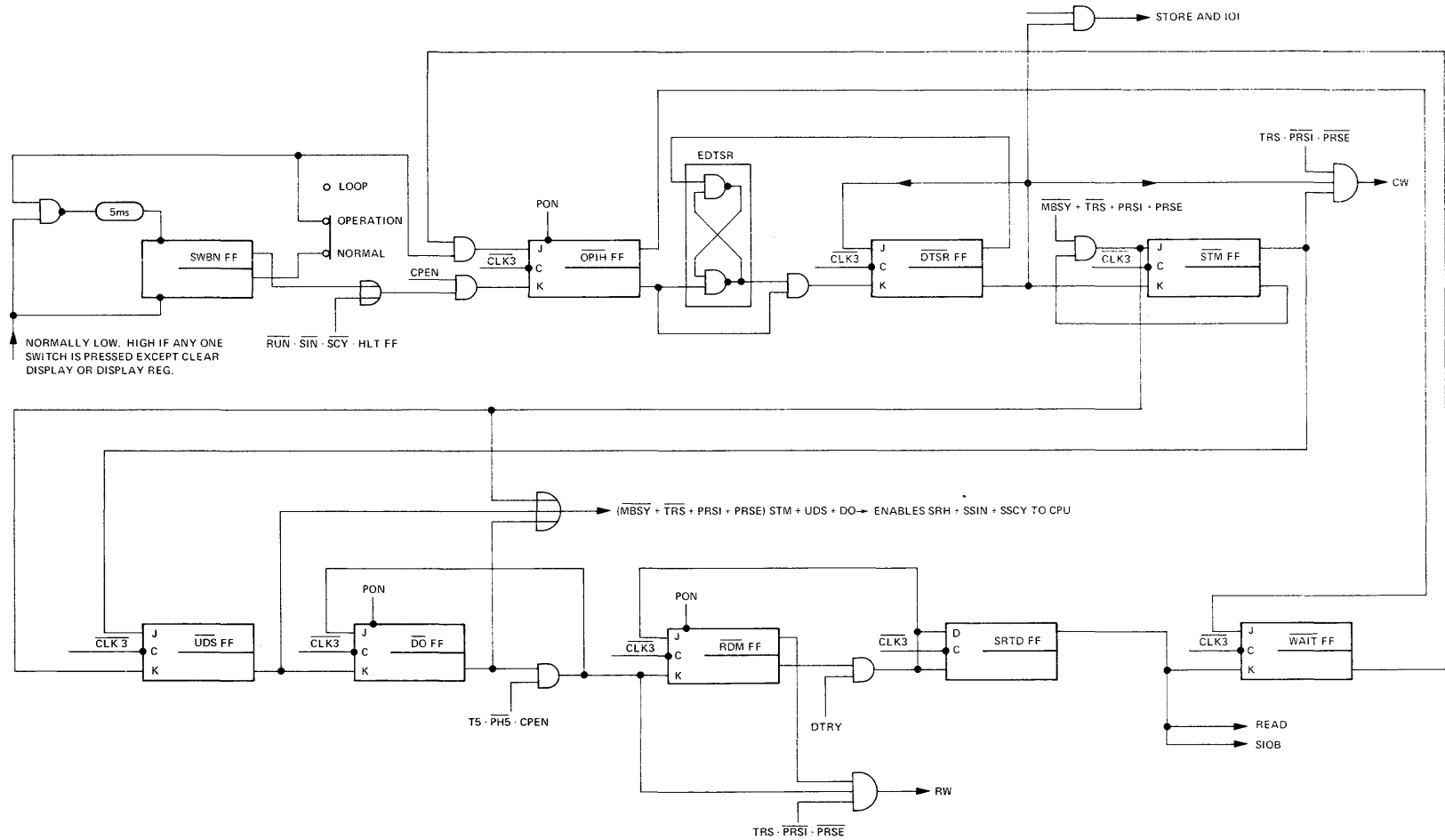
7

LESSON 7

FRONT PANEL

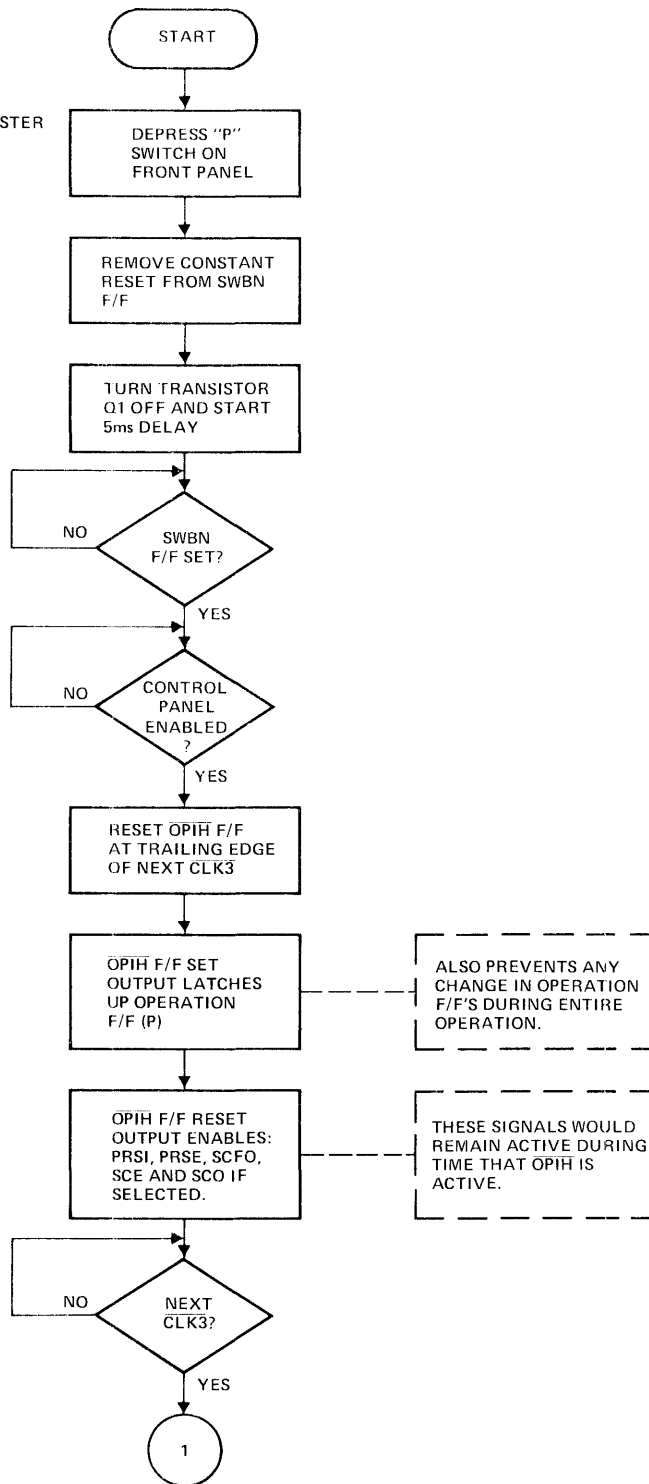
Operator Panel Control Logic	7-1	Block Diagram For Display B-Reg.	7-7
Front Panel "P" Operation	7-2	Block Diagram: Increment/Decrement 'M'	7-8
Front Panel "P" Operation (Cont'd.)	7-3	Register Switches (Operator's Panel)	
Front Panel "P" Operation (Cont'd.)	7-4	Simplified Diagram of 'Interrupt System'	7-9
Front Panel "P" Operation (Cont'd.)	7-5	Switch on Operator's Panel	
Block Diagram For Display P-Reg.	7-6	Front Panel Timing Diagram	7-10

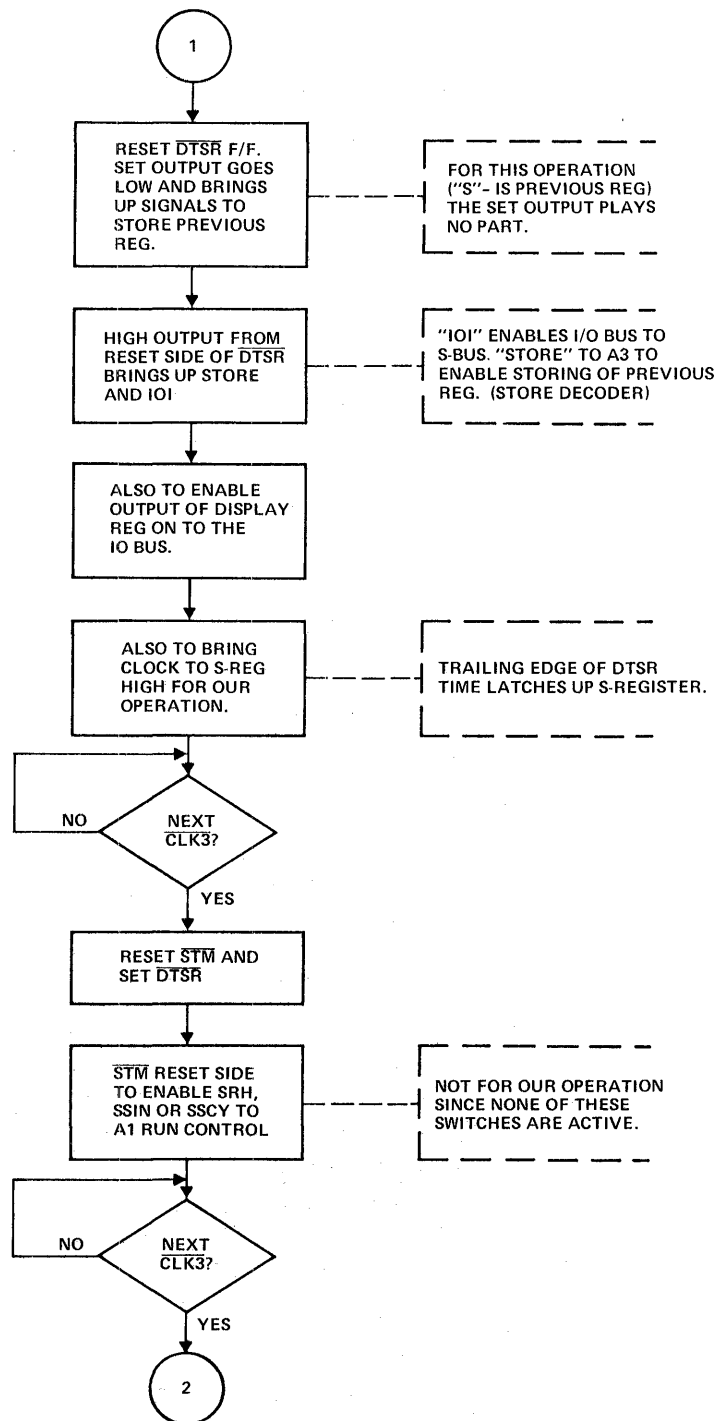
OPERATOR PANEL CONTROL LOGIC

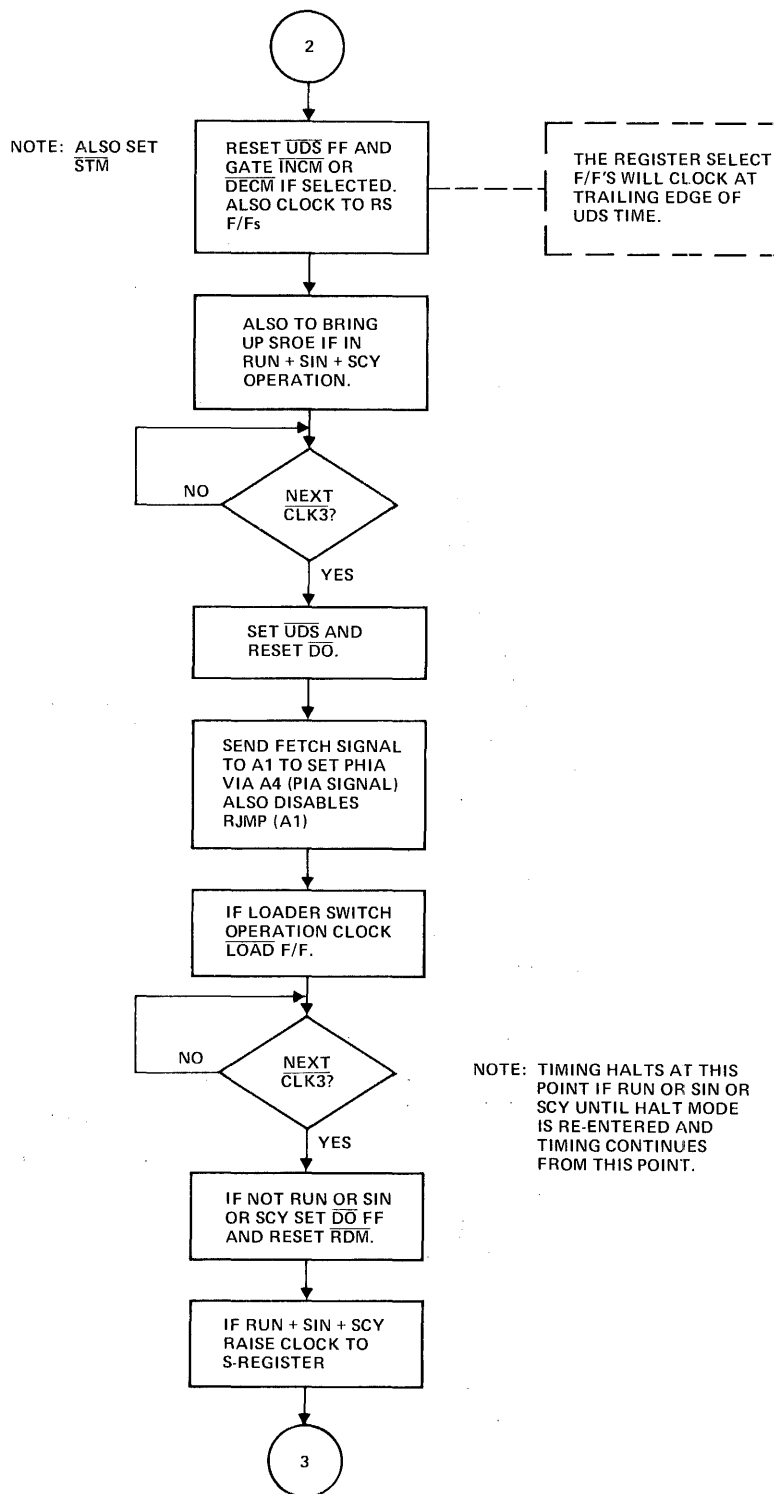


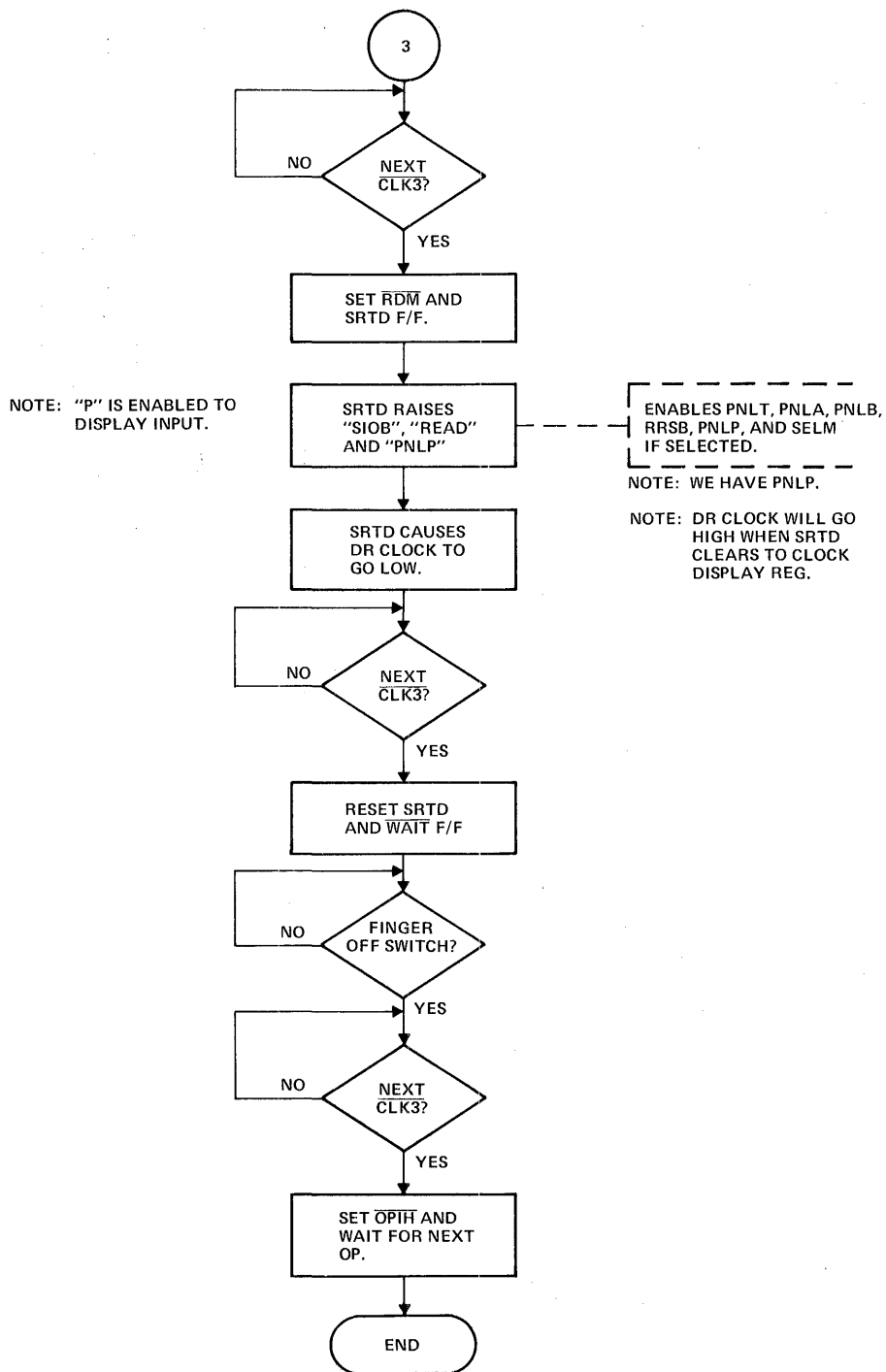
FRONT PANEL "P" OPERATION

NOTE 1. ASSUME S REGISTER IS CURRENTLY SELECTED

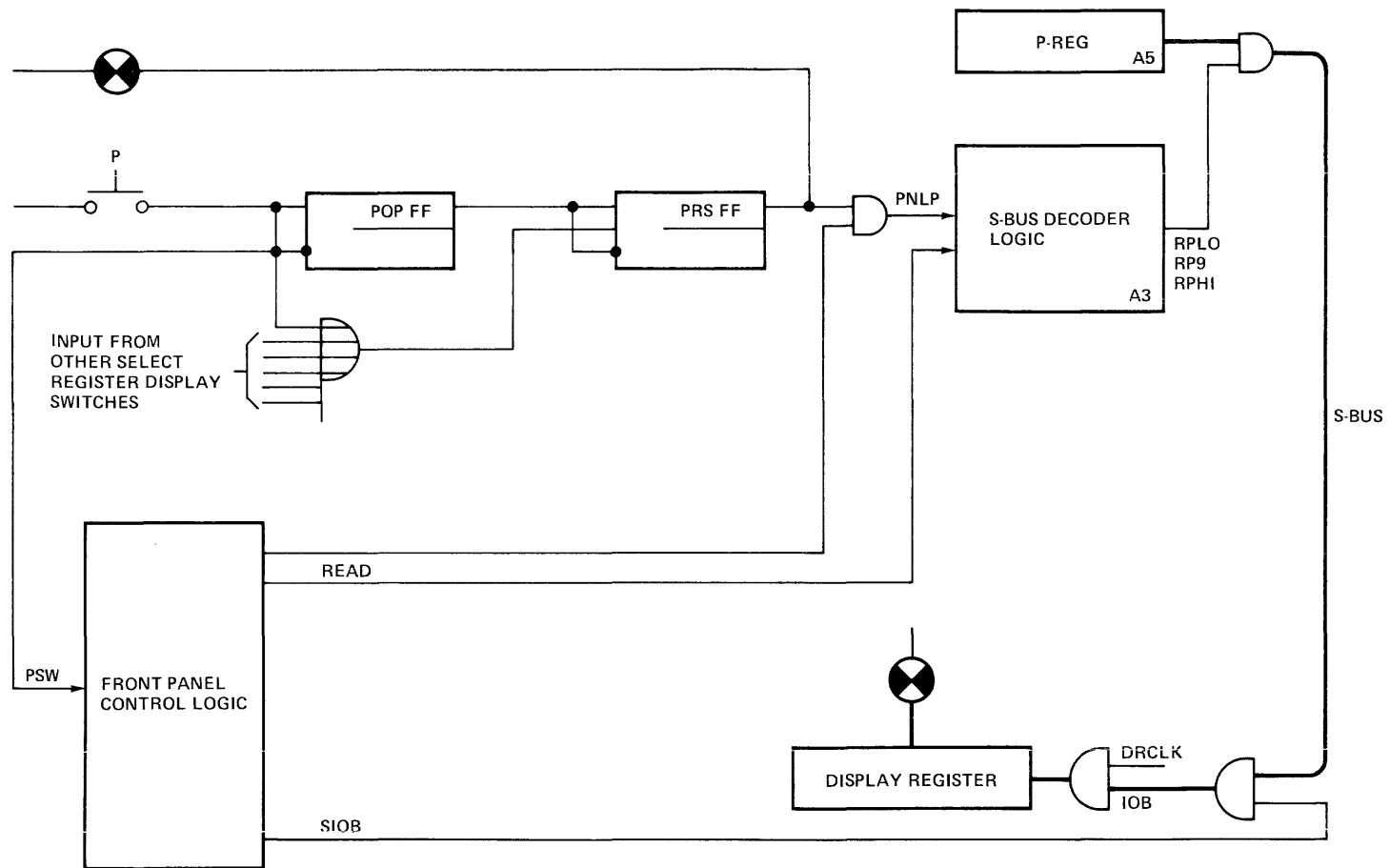




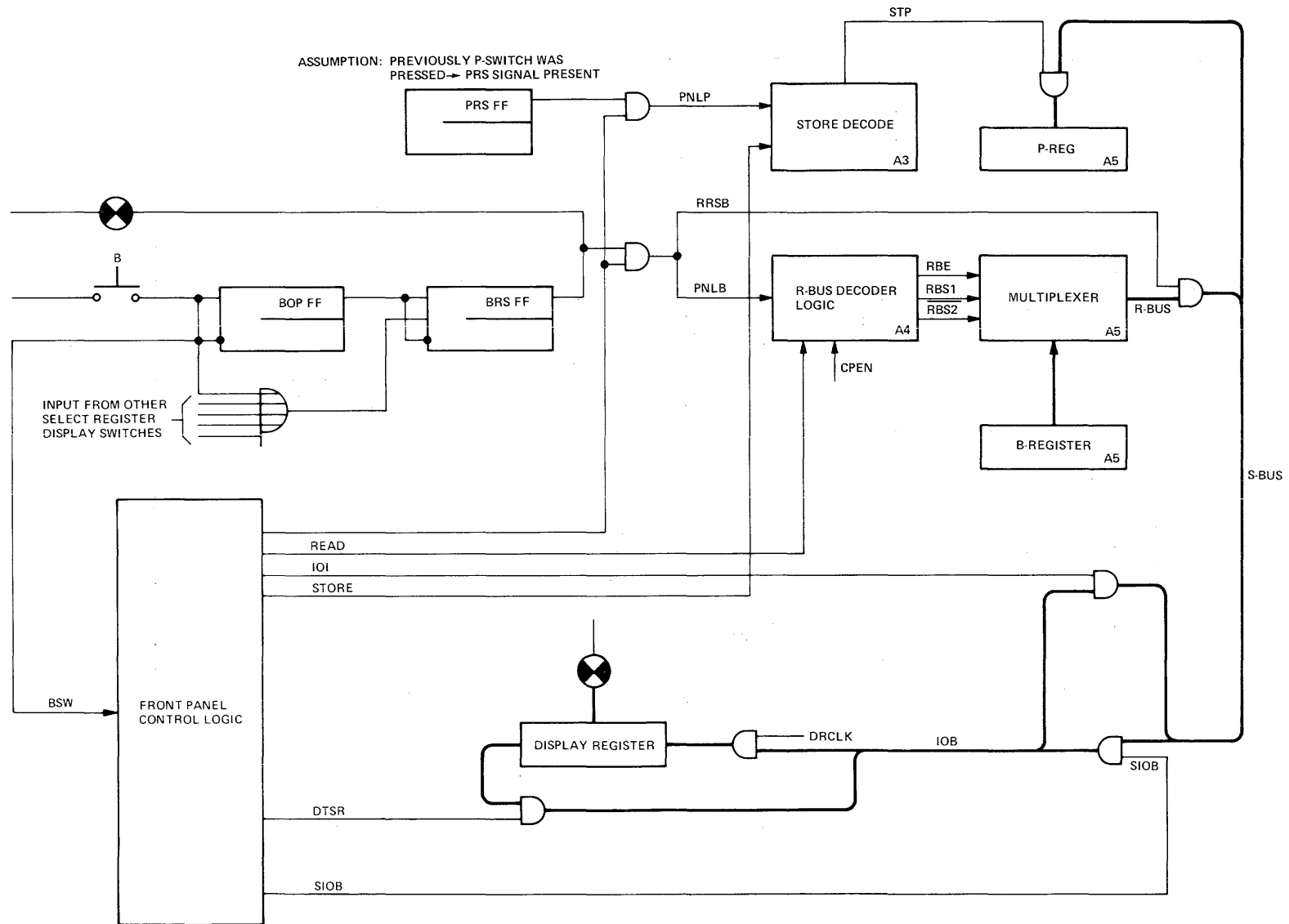




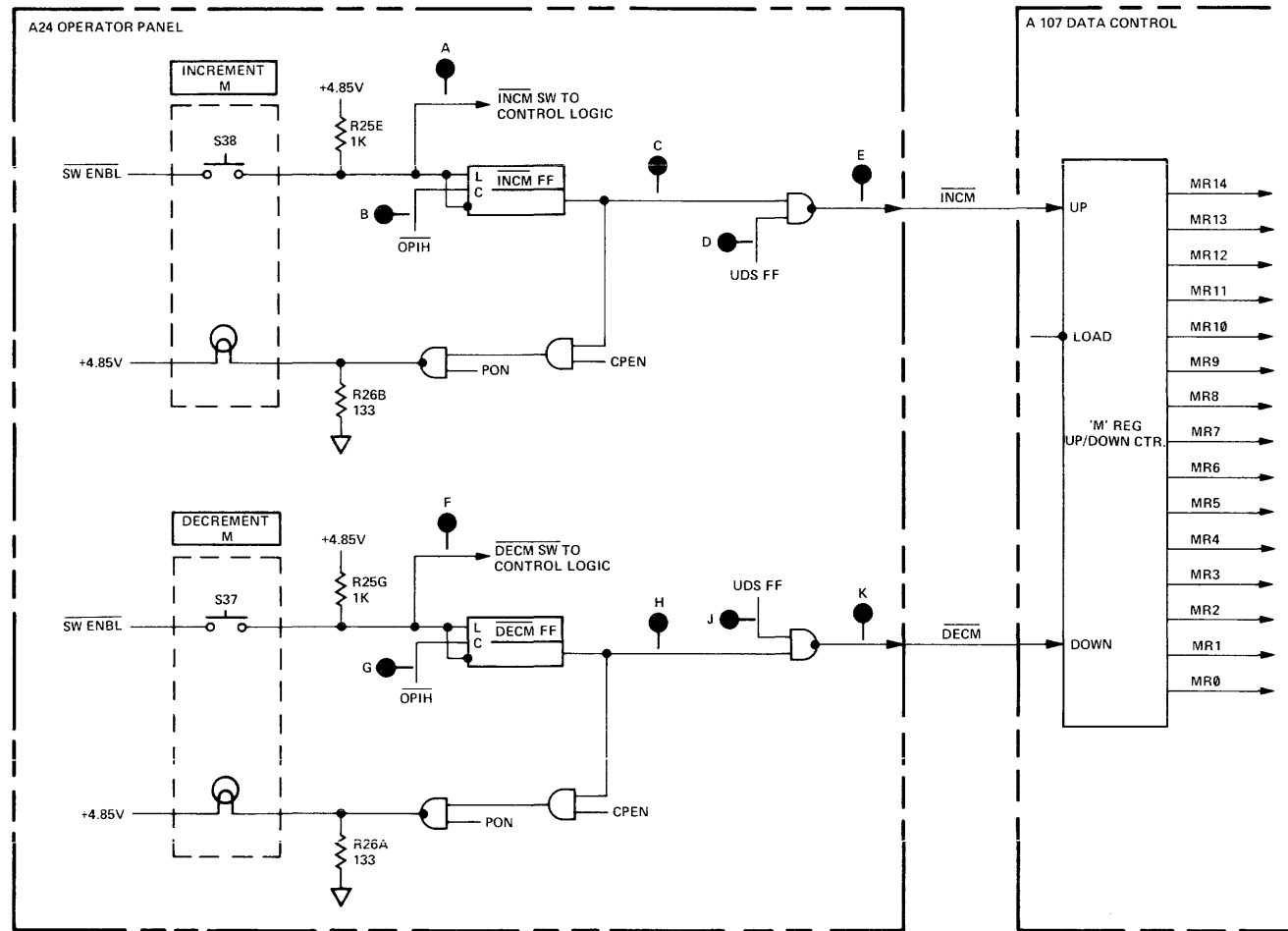
BLOCK DIAGRAM FOR DISPLAY P-REG.



BLOCK DIAGRAM FOR DISPLAY B-REG.
(1. P-REG IN DISPLAY → P-REG. 2. B-REG TRANSFERRED TO DISPLAY REG.)



BLOCK DIAGRAM: INCREMENT/DECREMENT 'M' REGISTER SWITCHES
(OPERATOR'S PANEL)



[illegible]

NOTES:

1. TEST POINTS SHOWN ARE FOR REFERENCE TO TIMING DIAGRAM ONLY.

FRONT PANEL TIMING DIAGRAM

