Technical Manual

Volume 1

PBC 1000 Revision 4

pb Packard Bell Computer

A SUBSIDIARY OF PACKARD BELL ELECTRONICS
1905 ARMACOST AVENUE • LOS ANGELES 25, CALIFORNIA

July 15, 1961

PREFACE

The PB 250 Computer Technical Manual is presented in two volumes, and is designed for the use of technical personnel engaged in computer checkout, modification, and field service. It is assumed that such technical personnel are familiar with basic computer technology.

Volume I contains a general description of the computer, together with a detailed description of PB 250 Computer logic.

Volume II contains detailed descriptions of installation, operation, power supply, Flexowriter, test points, and maintenance procedures. The appendices contain logic layout, applicable schematics, Flexowriter specifications, and material lists.

The logic equations and logic summary in this manual have been revised to include the "P" change.

CONTENTS

CONTENTS (continued)

FIGURES

TABLES

I. DESCRIPTION

A. GENERAL

The Packard Bell PB 250 is a high-speed, completely solid-state general purpose digital computer in which both the data and the commands required for computations are stored in an internal memory. The storage medium is a group of magnetostrictive delay lines along which acoustical pulses are propagated. At one end of each of these lines a "writing" device converts electrical pulses into acoustical energy. At the other end of each delay line a "reading" device converts acoustical energy back into electrical pulses.

The PB 250 provides more than 50 commands to permit coding of a broad range of scientific and engineering problems. Double precision commands are provided for operating upon large numbers. Commands for scaling and normalizing numbers permit floating point operation. In addition, square root, and variable length multiplication and division operations are available. Other features include input-output buffering, and a large variety of external optional units such as punched card equipment, paper tape photo-readers, magnetic tape handlers, analog-to-digital and digital-to-analog converters, and magnetic core storage.

The memory of the basic PB 250 contains 10 magnetostrictive delay lines numbered octally from 00 through 11. These delay lines store both data and commands. Each long line, 01 through 11, contains 256 decimal or 400 octal locations. These locations, also called sectors, are numbered

000 through 377. Line 00 is a 16-word fast access line. Since line 00 is 1/16 the length of a long line, any unit of information contained in it is available 16 times during each complete circulation of the 256-word long lines. Fifty-three additional delay lines may be added, each of which may have a total of from 1 to 256 words. These additional lines are numbered from 10 through 36, and 40 through 77. If all of the additional lines are used, and if all lines hold 256 words, the memory capacity of the PB 250 is extended to 15,888 words.

Since the PB 250 memory stores either data or commands, the generic term "word" is used to cover both types of information. Every word stored in the internal memory is in binary form (negative numbers are stored in complementary form). Words stored in the PB 250 memory are represented by a series of pulses which correspond to the digits of a binary number.

Three arithmetic registers, A, B, and C, are provided for arithmetic operations and information manipulation. Each register has the same format as a memory location. Information may be interchanged between the three registers. The contents of a register may be tested for nonpositive values, or compared against the contents of any memory location. A record may be kept in one register of operations performed in the other registers.

B. LEADING PARTICULARS

Two models of the PB 250 Computer are available : the rack-mounted PB 250 R, and the table-mounted PB 250 T. Both models are identical except for height requirements. Modular construction is used throughout the computer for ease of accessibility and maintenance. All electrical connections are made to a row of connectors at the back of the computer. Additional leading particulars are shown in Table 1-1.

Table 1-1.

LEADING PARTICULARS

Type	Serial, binary, internal program
Command structure	Single address with index register
Number of commands	59
Operation times:	
Add/subtract	12 μsec
Multiply	276 μsec (max.)
Divide	252 μsec (max.)
Square root	252 μsec (max.)
Average access time	1536 μsec
Average access time to fast memory	96 μsec
Maximum operational rate	40,000 instructions per second
Memory capacity	2320 words (Up to 15,888 words internal storage available)
Dimensions:	
PB 250 R (rack-mounted)	78 in. high, 19 in. wide, 24 in. deep
PB 250 T (table-mounted)	63 in. high, 19 in. wide, 24 in. deep
Power requirements	115 v, 60 cps, single phase, 110 watts
Weight	110 pounds (approx.)

C. STANDARD EQUIPMENT

1-1. CONTROL UNIT

A Model FX-1 Flexowriter is supplied as standard equipment, and is used as the control unit for the PB 250 Computer. The Flexowriter is also used to prepare, duplicate, and read paper tapes. The Flexowriter can be used on-line (under control of the computer), or off-line (under control of the operator). The general appearance and operation of the Flexowriter are similar to a standard electric typewriter.

1-2. BASIC MODULES

The majority of the logic used in the PB 250 is constructed from four types of plug-in transistorized modules, three of which are diode modules, while the fourth is a high-frequency flip-flop module. Logic for the PB 250 uses low-leakage, high-frequency diodes to permit two-level gating. All logic is of the "true-negative" (PNP) type, in which a -12 v level represents a "true" condition, and a 0 v level represents a "false" condition.

In addition to the logic circuits, plug-in transistorized modular construction is used for the magnetostrictive registers, and the various amplifiers, emitter followers, clock drivers, etc., required for computer operation.

Description and schematics of the basic modules are contained in PB 250 Technical Manual, Volume 2.

II. PB 250 LOGIC

A. GENERAL

This section contains a description of the functional logic of the PB 250. The logic is defined by Appendix A, Description of Notation; Appendix B, Glossary of Terms; Appendix C, Complete Logical Equations; and Appendix F, Logic Summary.

The logic equations are presented term-by-term, together with block diagrams which illustrate the application of gating terms.

B. COMMAND LIST

The list of commands for the PB 250 is given in Table 2-1. These commands are numbered octally from 00 to 77.

The six binary digits (bits) of the command number are stored in pulse positions 9 through 14 of the command. These bits are stored in the operation code register flip-flops and may be defined octally (see Figure 2-1).

Example:

State	O6	$\overline{O5}$	$\overline{O4}$	O3	O2	$\overline{O1}$	
reads	1	0	0	1	1	0	in binary
or		4			6		in octal

Combinations of signals from the operation code register flip-flops are used to

Table 2-1. (Sheet 1 of 3)

COMMAND CLASSIFICATIONS

Class 1: Executed Between Command Location and Address Sector Number.		
NORMALIZE AND DECREMENT	NAD	(20)*
NORMALIZE	NOR	(20)*
LEFT SHIFT AND DECREMENT	LSD	(21)*
AB LEFT	SLT	(21)*
RIGHT SHIFT AND INCREMENT	RSI	(22)*
AB RIGHT	SRT	(22)*
SCALE RIGHT AND INCREMENT	SAI	(23)
NO OPERATION	NOP	(24)
INTERCHANGE A AND M	IAM	(25)
MOVE LINE X TO LINE 7	MLX	(26)
SQUARE ROOT	SQR	(30)
DIVIDE	DIV	(31)*
DIVIDE REMAINDER	DVR	(31)*
MULTIPLY	MUP	(32)
SHIFT B RIGHT	SBR	(33)*
LOGICAL RIGHT SHIFT	LRS	(33)*
WRITE OUTPUT CHARACTER	WOC	(6X)
PULSE TO SPECIFIED UNIT	PTU	(70)
MOVE COMMAND LINE BLOCK	MCL	(71)
BLOCK SERIAL OUTPUT	BSO	(72)
BLOCK SERIAL INPUT	BSI	(73)

*Asterisk indicates that the op code has at least two meanings, depending on the address used with the command.

Table 2-1. (Sheet 2 of 3)

Class 2: Executed in Address Sector Number		
INTERCHANGE A AND C	IAC	(01)
INTERCHANGE B AND C	IBC	(02)
LOAD A	LDA	(05)
LOAD B	LDB	(06)
LOAD C	LDC	(04)
STORE A	STA	(11)
STORE B	STB	(12)
STORE C	STC	(10)
ADD	ADD	(14)
SUBTRACT	SUB	(15)
EXTEND BIT PATTERN	EBP	(40)
GRAY TO BINARY	GTB	(41)
AND M & C	AMC	(42)
CLEAR A	CLA	(45)
CLEAR B	CLB	(43)
CLEAR C	CLC	(44)
AND OR COMBINED	AOC	(46)
EXTRACT FIELD	EXF	(47)
DISCONNECT INPUT UNIT	DIU	(50)
READ TYPEWRITER KEYBOARD	RTK	(51)
READ PAPER TAPE	RPT	(52)
READ FAST UNIT	RFU	(53)
LOAD A FROM INPUT BUFFER	LAI	(55)
COMPARE A AND M	CAM	(56)
CLEAR INPUT BUFFER	CIB	(57)
HALT	HLT	(00)*
MERGE A INTO C	MAC	(00)*

*Asterisk indicates that the op code has at least two meanings, depending on the address used with the command.

Table 2-1. (Sheet 3 of 3)

Class 3: Executed In Address Sector Number And Following Sector.		
ROTATE	ROT	(03)
LOAD DOUBLE PRECISION	LDP	(07)
STORE DOUBLE PRECISION	STD	(13)
DOUBLE PRECISION ADD	DPA	(16)
DOUBLE PRECISION SUBTRACT	DPS	(17)

Class 4: Executed Between Command Location And Address Sector Number		
TRANSFER UNCONDITIONALLY	TRU	(37)
TRANSFER IF A NEGATIVE	TAN	(35)
TRANSFER IF B NEGATIVE	TBN	(36)
TRANSFER IF C NEGATIVE	TCN	(34)
TRANSFER ON OVERFLOW	TOF	(75)
TRANSFER ON EXTERNAL SIGNAL	TES	(77)

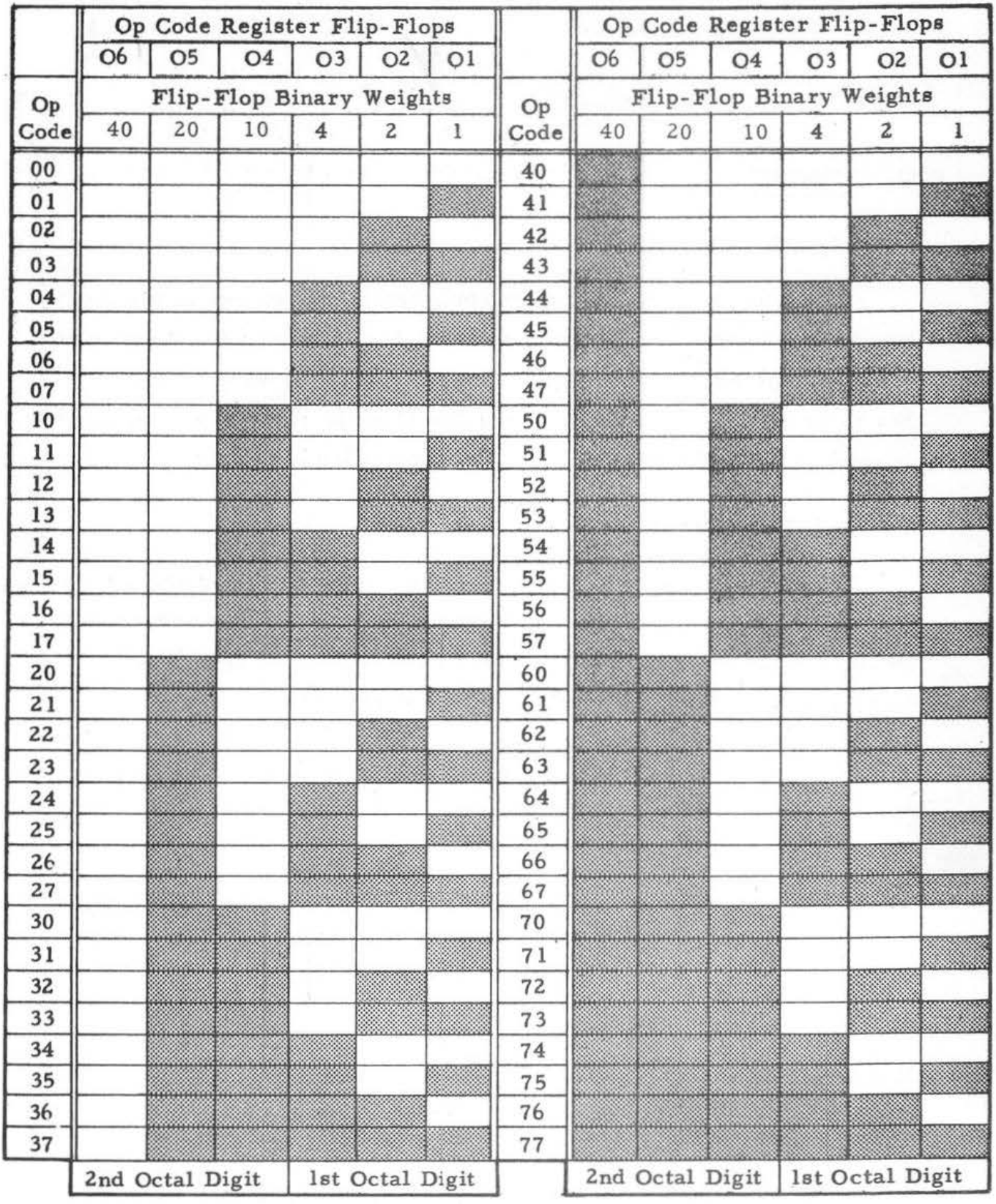


Figure 2-1. Command Codes In Operation Code Register

condition many of the logic gates.

C. TIMING

2-1. PULSE COUNTER

Basic pulse timing is provided by a binary five stage counter that serves as the pulse counter (see Figure 2-2). This counter consists of five flip-flops whose states define the pulse counts as shown in Figure 2-3. The pulse counter is similar to a regular 32-pulse counter, except that F5 is reset at the end of P23 and F4 is set only at the end of P7.

2-2. SECTOR COUNTER

For sector number information, a one-word (24-pulse) register is provided as a sector counter (see Figure 2-4). Serial sector numbers are obtained in the pulse intervals (P8-P15) and (P16-P23) by setting carry flip-flop, Sc, at P7 and P15 (see Figure 2-5).

$$sSc = \overline{F5}\ F3\ F2\ F1$$
$$rSc = P23 + \overline{Sr}\ (\overline{F3} + \overline{F2} + \overline{F1})$$

$$sSw = Sc\ \overline{Sr}\ \overline{Qg} + \overline{Sc}\ Sr\ \overline{Qg} + - - -$$
$$rSw = Sc\ Sr + \overline{Sc}\ \overline{Sr}\ (\ ) + Qg\ (\ )$$

$$sSr = (Sw \text{ delayed by 22 pulse times})$$
$$rSr = (\overline{Sw} \text{ delayed by 22 pulse times})$$

The sector counter clear term, Qg, is described in paragraph 2-3. The use of the interval (P24 - P7) as an input buffer is described in paragraph 2-9. This interval is protected by resetting the sector counter carry flip-flop, Sc, at P23.

The sector counter numbers in (P8 - P15) and (P16-P23) advance from

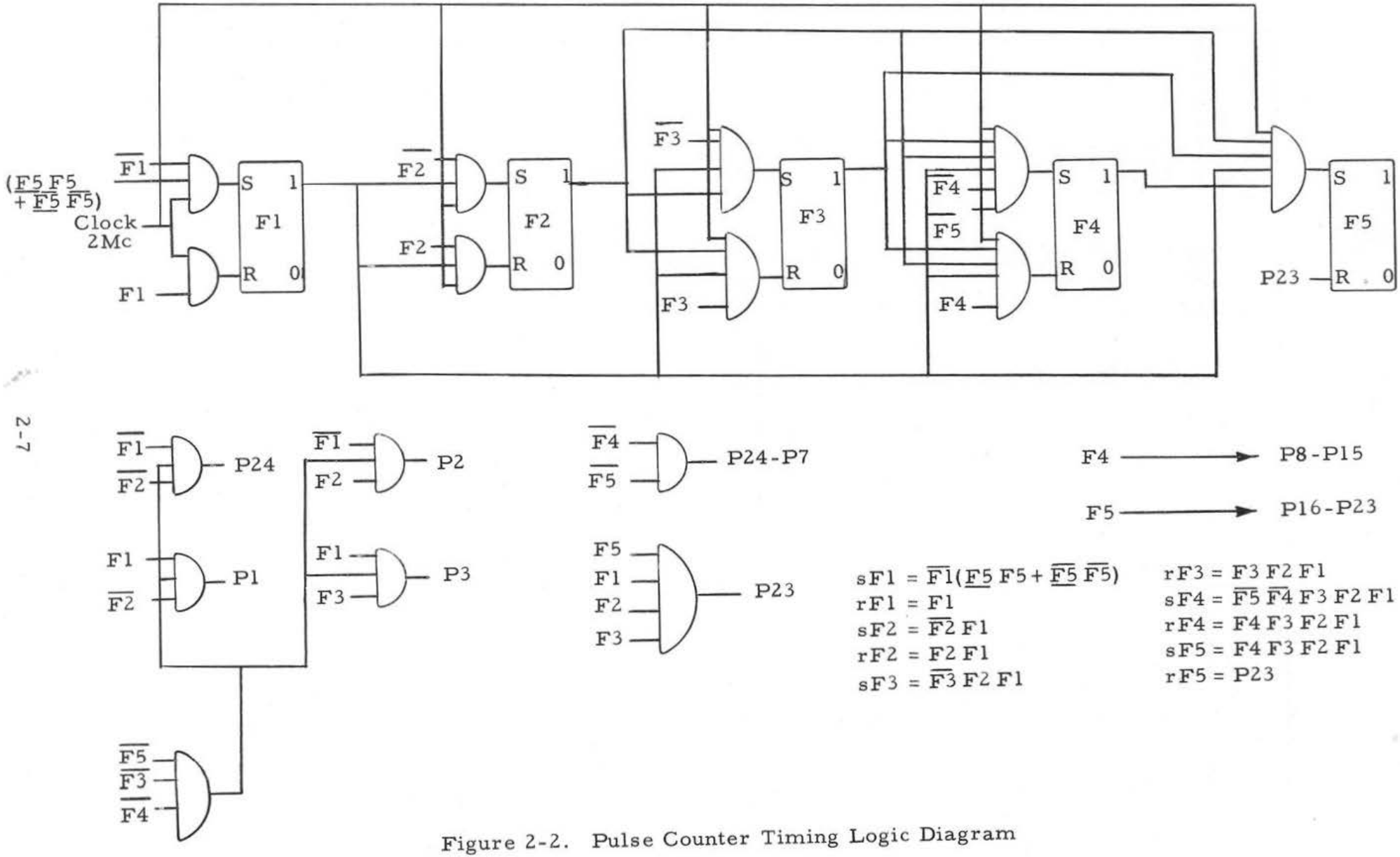


Figure 2-2. Pulse Counter Timing Logic Diagram

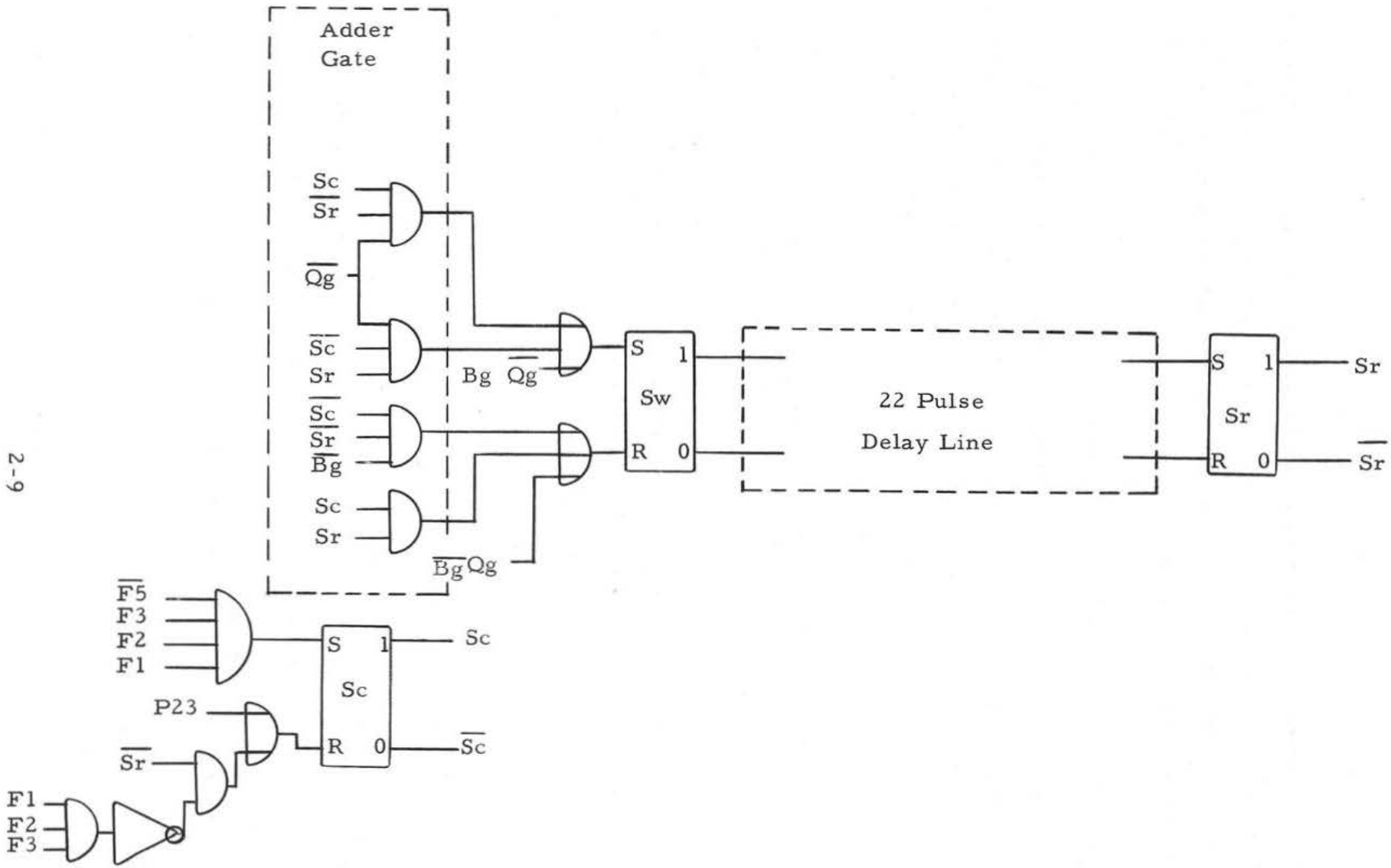
Maintenance Timing	F5	F4	F3	F2	F1	Programming Timing
P1					■	*
P2				■		T1
P3				■	■	T2
P4			■			T3
P5			■		■	T4
P6			■	■		T5
P7			■	■	■	T6
P8		■				T7
P9		■			■	T8
P10		■		■		T9
P11		■		■	■	T10
P12		■	■			T11
P13		■	■		■	T12
P14		■	■	■		T13
P15		■	■	■	■	T14
P16	■					T15
P17	■				■	T16
P18	■			■		T17
P19	■			■	■	T18
P20	■		■			T19
P21	■		■		■	T20
P22	■		■	■		T21
P23	■		■	■	■	T22
P24						*

□ Indicates that the flip-flop is in its reset or "false" condition.

■ Indicates that the flip-flop is in its set or "true" condition.

*P1 and P24 are not accessible to the programmer and are not numbered.

Figure 2-3. Pulse Counts



$\overline{F5}\ F3\ F2\ F1 = (P7 + P15)$

Figure 2-4. Sector Counter Timing Logic Diagram

000 to 255 for each machine cycle. These numbers are used in comparison with the one-word instruction register to define the numbers of the sectors which follow. An example of the numbers presented by the sector counter in memory is shown in Figure 2-6.

2-3. MULTIPLE COMPUTER OPERATION

To synchronize the pulse counters of two computers, the F5 signal from the master computer enters the slave computer and is designated $\underline{F5}$. With the clock generators operating from a single crystal oscillator, the first stage of the slave computer's pulse counter is controlled by $\underline{F5}$.

$$sF1 = \overline{F1}\,(\underline{F5}\ F5 + \overline{\underline{F5}}\ \overline{F5}).$$

This prevents counting in the slave computer when the most significant stages of the two pulse counters are different, which synchronizes the counters.

To synchronize the sector counter of the slave computer, the master computer executes a command 70 having a line number 37_8 during sector 255 (377_8). This command is detected in the Qg gate of the slave computer and serves to zero the sector counter.

$$Qg = - - - + \underline{Cpg}\ \underline{M3g}\ \underline{N7g} + - - -$$

D. OPERATION PHASES

The operation phases are controlled with the Ec and Rc flip-flops, based on the successive commands read into the instruction register and into the operation code register flip-flops. The basic timing of the operation phases is provided by comparing sector numbers in the instruction register with the sector counter using the comparison flip-flop, Is.

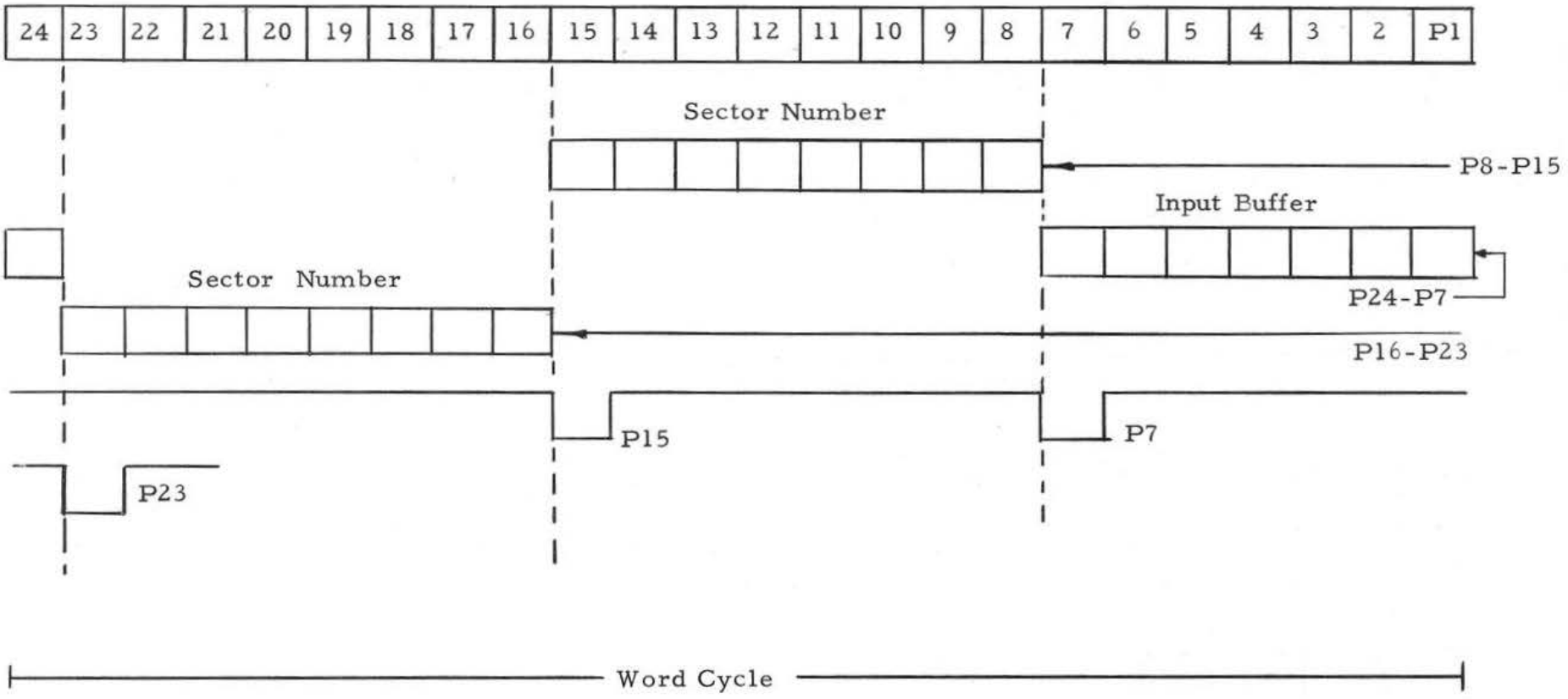


Figure 2-5. Pulse Time Format Timing Diagram

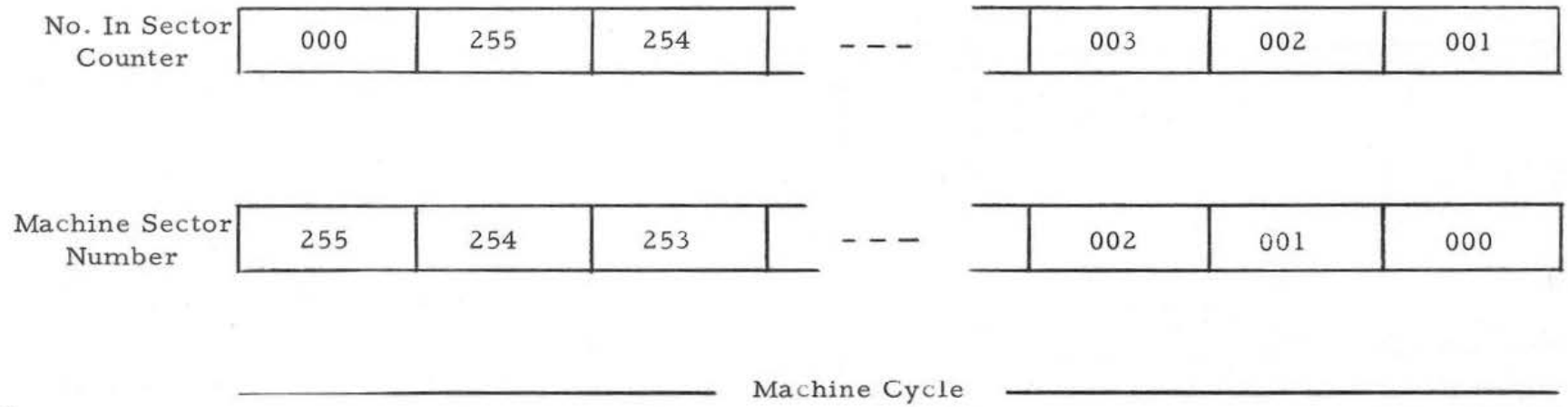


Figure 2-6. Sector Format Timing Diagram

The four operation phases are:

Phase I	$\overline{Ec}\ \overline{Rc}$	Wait to read next command
Phase 2	$\overline{Ec}$ Rc	Read command
Phase 3	Ec Rc	Wait to execute command
Phase 4	Ec $\overline{Rc}$	Execute command

Depending on the timing, phases 1 or 3 may be skipped if there is no requirement to wait for a command or an execution. When the computer is stopped, it idles in phase 1. The operation code register flip-flops exert some control over the operation phases. For commands 20 through 37, and 60 through 77 (characterized by the O5 flip-flop being "true"), the computer automatically skips from phase 2 to phase 4, and phase 4 is terminated by the address sector number. For all other commands, phase 4 terminates automatically after one or two sectors. In any command, when the operation code flip-flop, Oc, is set, the computer will skip phase 1 at the end of phase 4. In the case of branch commands, 34, 35, 36, 37, 75 and 77, the Oc flip-flop is used to detect the branch control condition. If Oc is set, the computer goes to phase 4. If Oc is reset, the computer skips from phase 2 to phase 4 for one pulse time (P1) and then returns to phase 2 so that the next command in sequence will be read. Indication of the phase changes is presented in Figure 2-7.

2-4. PHASE 1: WAIT TO READ COMMAND

During each sector of phase 1, the Is flip-flop compares the next command sector number in the instruction register with the sector counter to determine when to read the next command.

$$sIs = P24$$

$$rIs = (\overline{Sr}\ sIw + Sr\ rIw)\left[\overline{Ec}\ \overline{Rc}\ (P8\text{-}P15) + - - -\right]$$

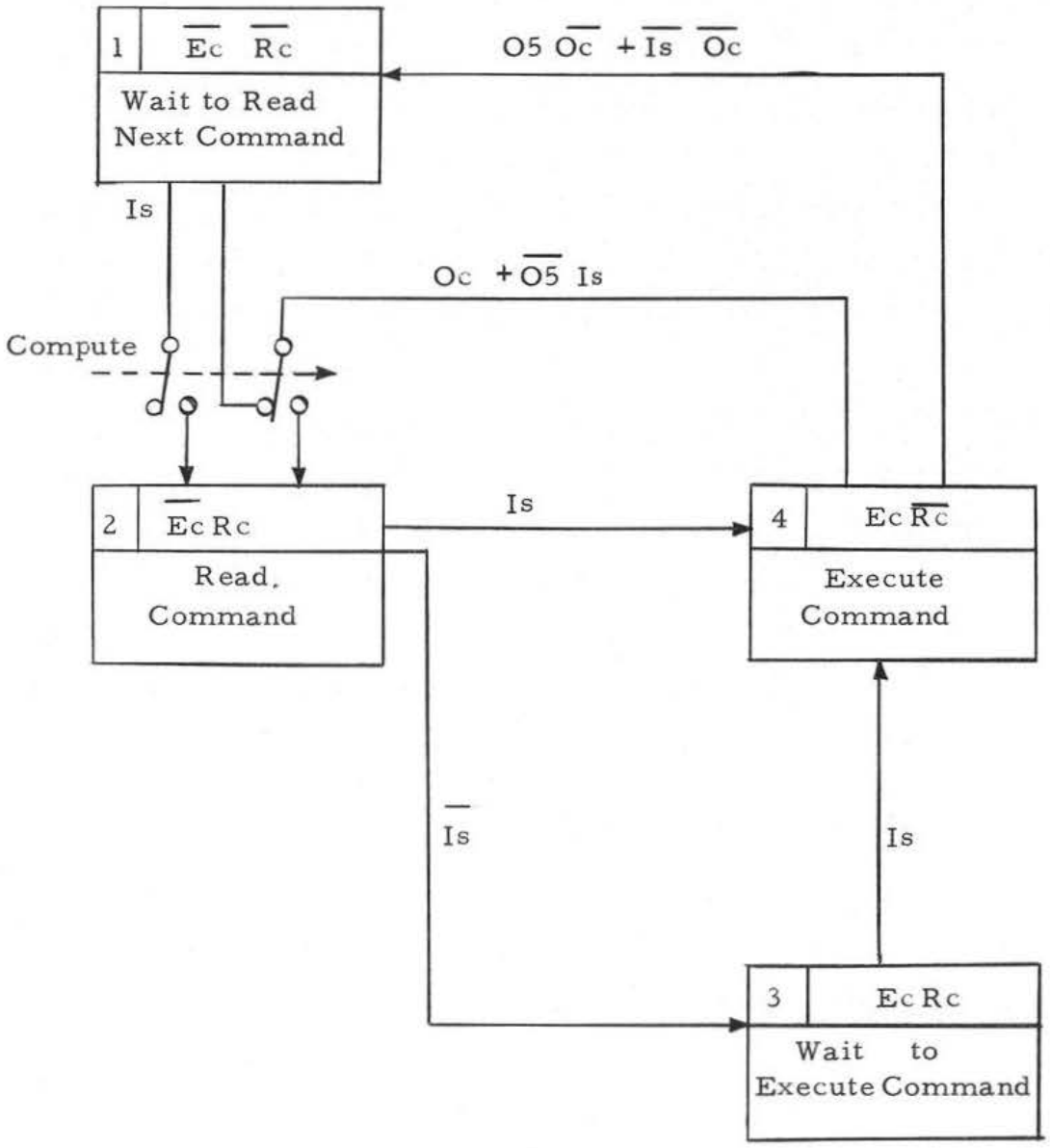


Figure 2-7. Simplified Phase Chart

2-5. PHASE 2: READ COMMAND

When the command sector number agrees with the sector counter, the Is flip-flop will still be set at the end of the sector, and phase 2 will be initiated (Rc set).

$$sRc = P24\ \overline{Ec}\ \overline{Rc}\ Is\ \left(\overline{En} + - - -\right)\ \overline{Fi} + - - -$$

Phase 2 is conditioned by depressing the ENABLE switch, which qualifies all sRc terms as a computation interrupt $\left(\overline{En}\right)$ Raising the FILL switch also interrupts computation (see Figure 2-8).

During the single sector of phase 2 the bits of the next command to be executed are read into the proper storage locations. The bit in bit position 2 of each command is temporarily stored in operation code register flip-flop, O2.

$$sO2 = \overline{Ec}\ Rc\ P2\ Vg + - - -$$
$$rO2 = \overline{Ec}\ Rc\ P2\ \overline{Vg} + - - -$$

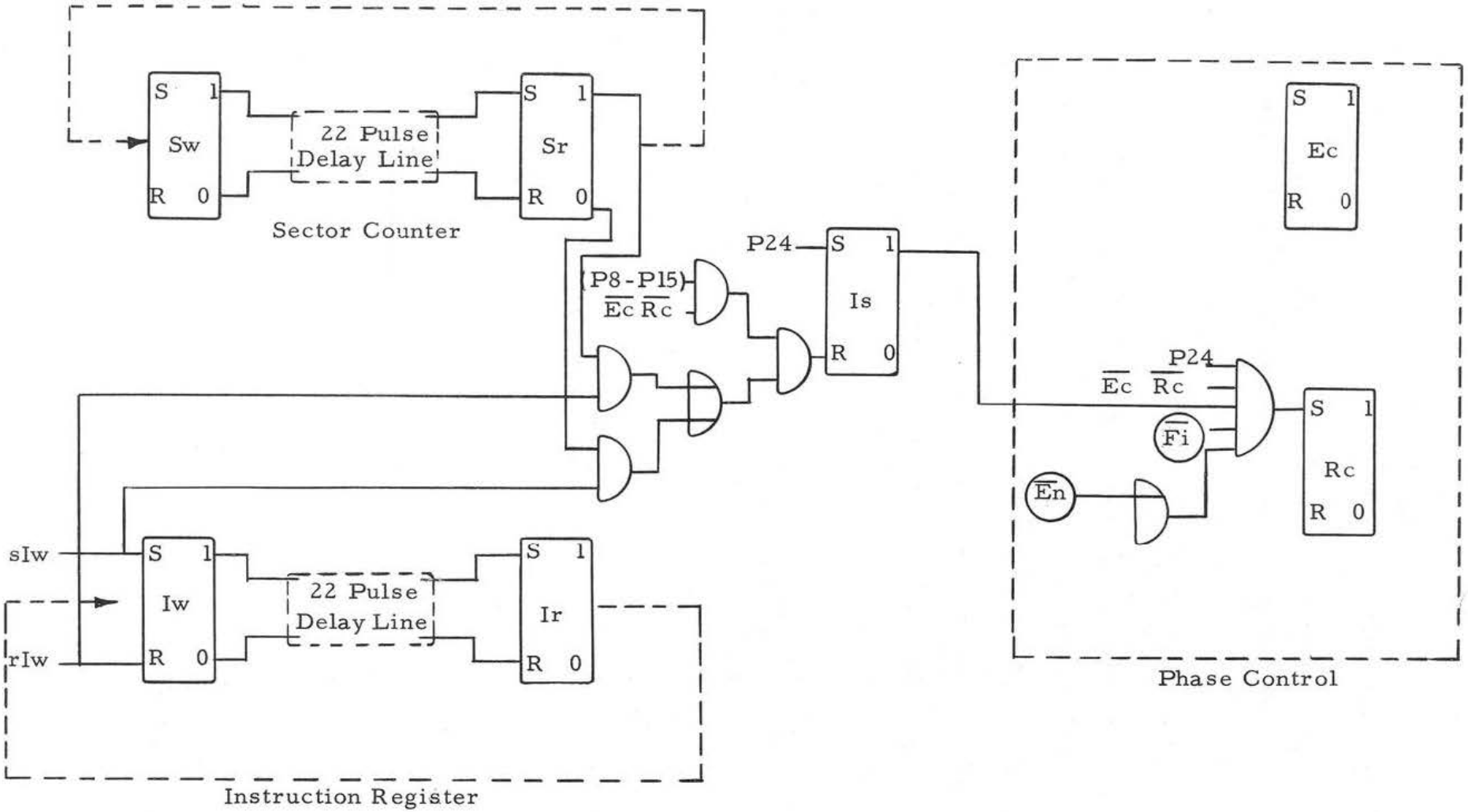
This bit is then used to select the operand line number from either the index portion of the instruction register if the O2 flip-flop is set, or from the line number portion of the command itself if the O2 flip-flop is reset, during bit positions 4 through 8 (see Figure 2-9).

$$sL5 = Lg\ \overline{O2}\ Vg + Lg\ O2\ Iw + - - -$$
$$rL5 = Lg\ \overline{O2}\ \overline{Vg} + Lg\ O2\ \overline{Iw}$$

$$sL4 = Lg\ L5 + - - -$$
$$rL4 = Lg\ \overline{L5} + - - -$$

$$sL3 = Lg\ L4 + - - -$$
$$rL3 = Lg\ \overline{L4} + - - -$$

$$sL2 = Lg\ L3 + - - -$$
$$rL2 = Lg\ \overline{L3} + - - -$$



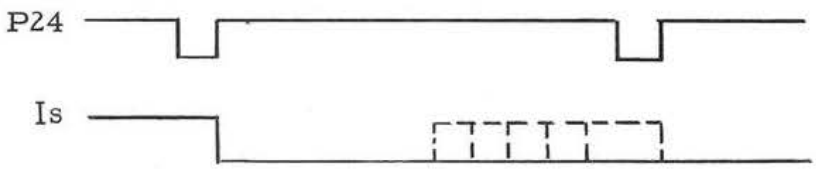


Figure 2-8. Phase Control (Ph 1 to Ph 2) Logic Diagram

$$sL1 = Lg\ L2 + - - -$$
$$rL1 = Lg\ \overline{L2} + - - -$$
$$Lg = \overline{Ec}\ Rc\ \overline{F5}\ (\overline{F4} + \overline{F3}\ \overline{F2}\ \overline{F1})\ \overline{P24}$$

The bits in bit positions 9 through 15 are shifted into the operation code register flip-flops.

$$sOc = Og\ Vg + - - -$$
$$rOg = Og\ \overline{Vg} + - - -$$
$$sO6 = Og\ Oc + - - -$$
$$rO6 = Og\ \overline{Oc} + - - -$$
$$sO5 = Og\ O6 + - - -$$
$$rO5 = Og\ \overline{O6} + - - -$$
$$sO4 = Og\ O5 + - - -$$
$$rO4 = Og\ \overline{O5} + - - -$$
$$sO3 = Og\ O4 + - - -$$
$$rO3 = Og\ \overline{O4}$$
$$sO2 = Og\ O3 + - - -$$
$$rO2 = Og\ \overline{O3} + - - -$$
$$sO1 = Og\ O2 + - - -$$
$$rO1 = Og\ \overline{O2} + - - -$$
$$Og = \overline{Ec}\ Rc\ F4$$

During the shift into the operation code register flip-flops, the next command sector number is advanced by writing the new number in the sector counter into the instruction register (see Figure 2-10).

$$sIw = \overline{Ec}\ Rc\ (P8\text{-}P15)\ Sr + - - -$$

The address sector number of the command is written into the instruction register following the next command location sector number.

$$sIw = + \overline{Ec}\ Rc\ (P16 - P23)\ Vg + - - -$$

This address sector number of the command is also compared with the sector counter to determine if phase 4 should follow phase 2.

$$rIs = (\overline{Sr}\ sIw + Sr\ rIw)\left[- - - + \overline{O5}\ Rc\ (P16\text{-}P23) + - - -\right]$$

Phase 2 is always changed to phase 3 or 4 after one sector.

$$sEc = P24\ \overline{Ec}\ Rc$$

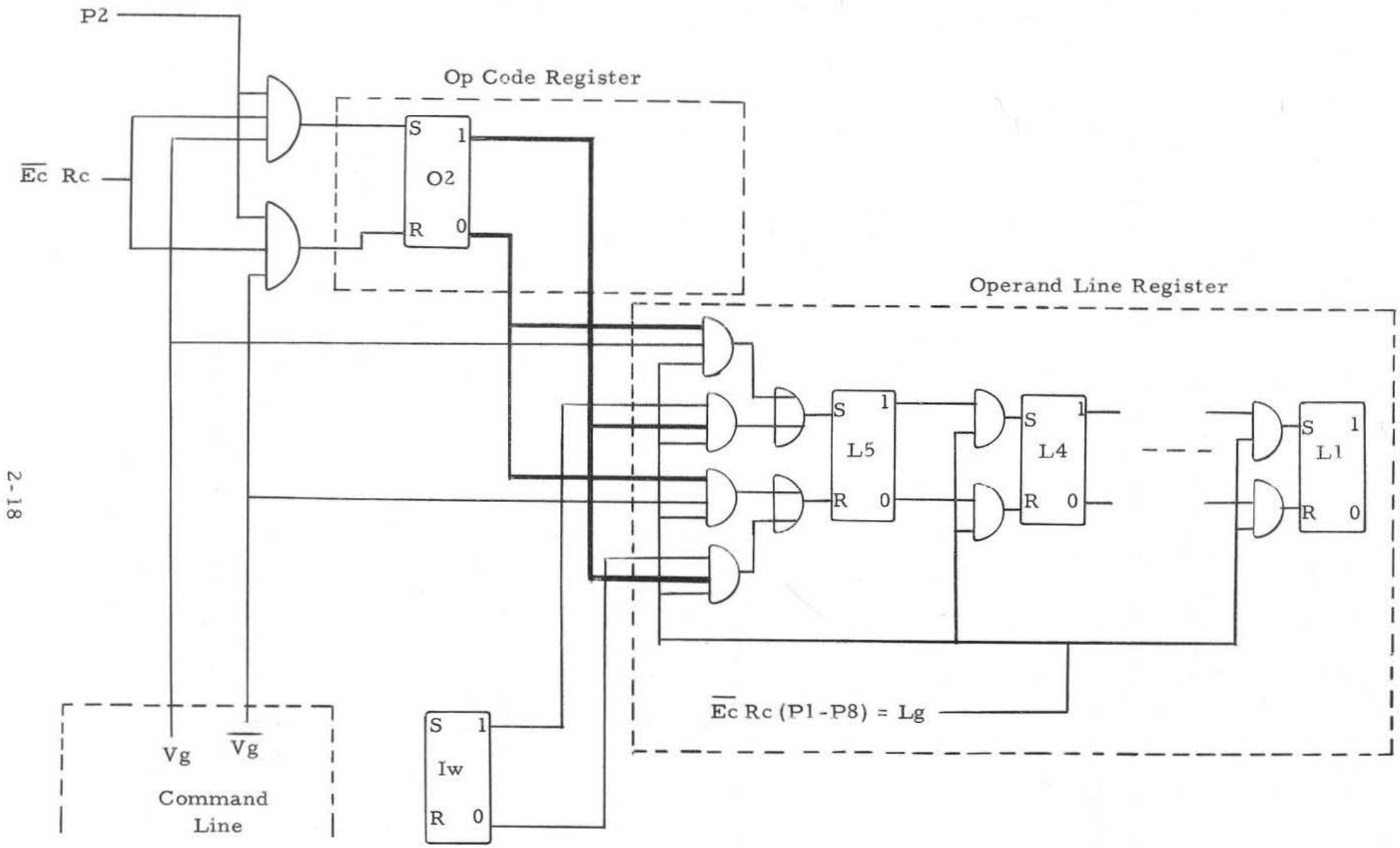


Figure 2-9. Operand Line Selection Logic Diagram

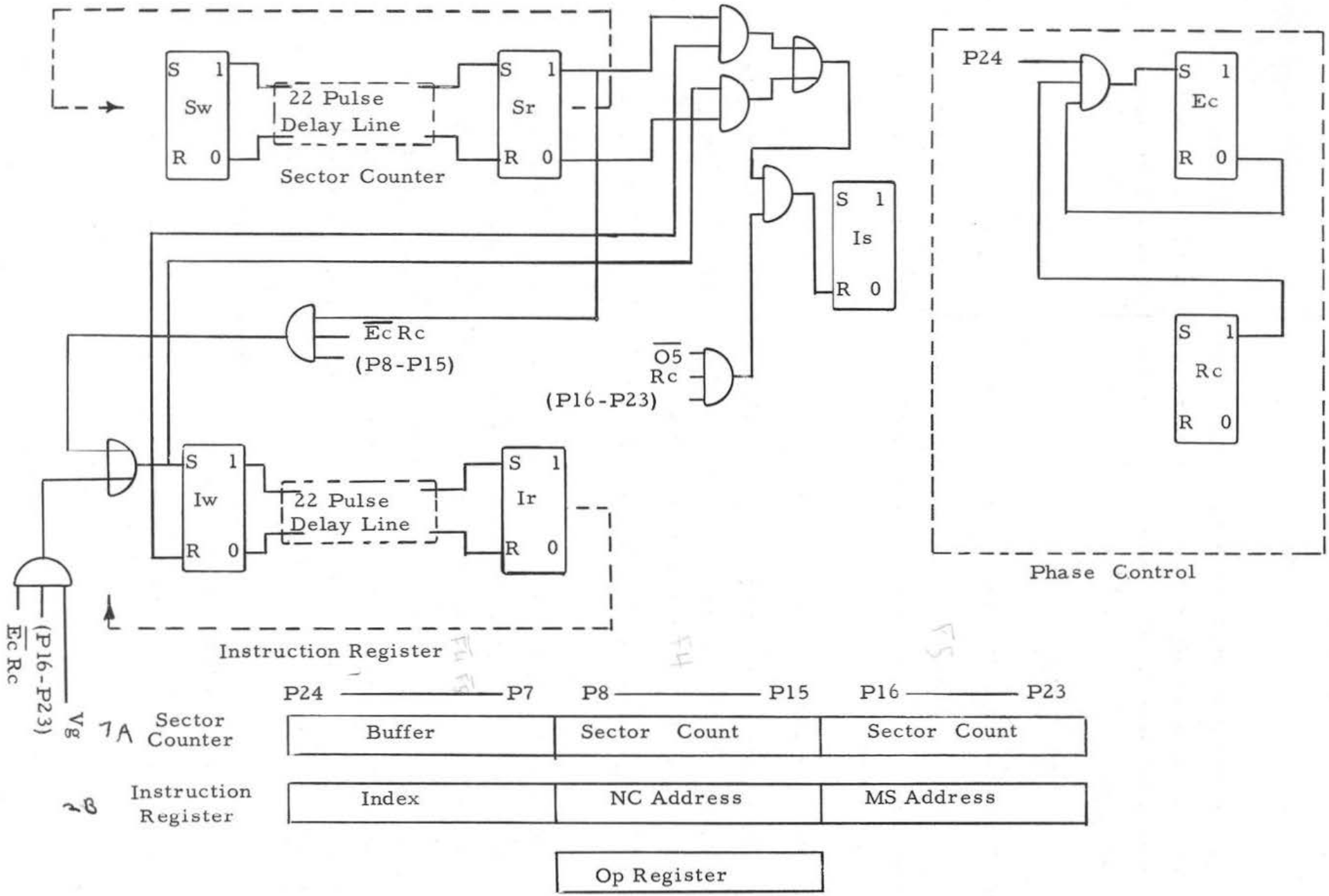


Figure 2-10. Command and Address Advance Logic Diagram

At the end of the read command sector, the flip-flop Is will be set and phase 4 will follow phase 2 directly if the command includes O5 (O5 in its "true" condition), or if the address sector number agreed with the sector counter in phase 2.

$$rRc = P24\ Rc\ Is + - - -$$

2-6. PHASE 3: WAIT TO EXECUTE COMMAND

If phase 3 is allowed to occur, the address sector number continues to be compared with the sector counter until these two numbers agree, causing phase 4 to be set.

2-7. PHASE 4: EXECUTE COMMAND

During phase 4 the command is executed. The end of the execute phase for commands 00, 01, 02, 04, 05, 06, 10, 11, 12, 14, 15, and 40 through 57, occurs after one sector (see Sheet 1, Figure 2-11).

$$Eg = P24\ Ec\ \overline{Rc}\ (\overline{O5}\ \overline{O2} + \overline{O5}\ O6 + \overline{O5}\ \overline{O3}\ \overline{O1} + \overline{O5}\ \overline{O4}\ \overline{O1} + - - -$$

$$rEc = Eg + - - -$$

Commands 03, 07, 13, 16, and 17 are effectively changed to commands 01, 05, 11, 14, and 15, respectively, after one sector of execution and, after two sectors, execution is terminated by the P24 Ec $\overline{Rc}$ $\overline{O5}$ $\overline{O2}$ term of Eg.

$$rO2 = - - - + P24\ Ec\ \overline{Rc}\ \overline{O6}\ \overline{O5}\ O1 + P24\ Ec\ \overline{Rc}\ \overline{O6}\ \overline{O5}\ O4\ O3$$

For commands 20 through 37, and 60 through 77, phase 4 is terminated by matching the address sector number and the sector number and the sector counter (see Sheet 1, Figure 2-11).

$$rIs = (\overline{Sr}\ sIw + Sr\ rIw)\left[- - - + Ec\ O5\ (P16\text{-}P23) + - - -\right]$$

$$Eg = P24\ Ec\ \overline{Rc} - - - + O5\ Is$$

If the Oc flip-flop is set during phase 4, the next command number in the

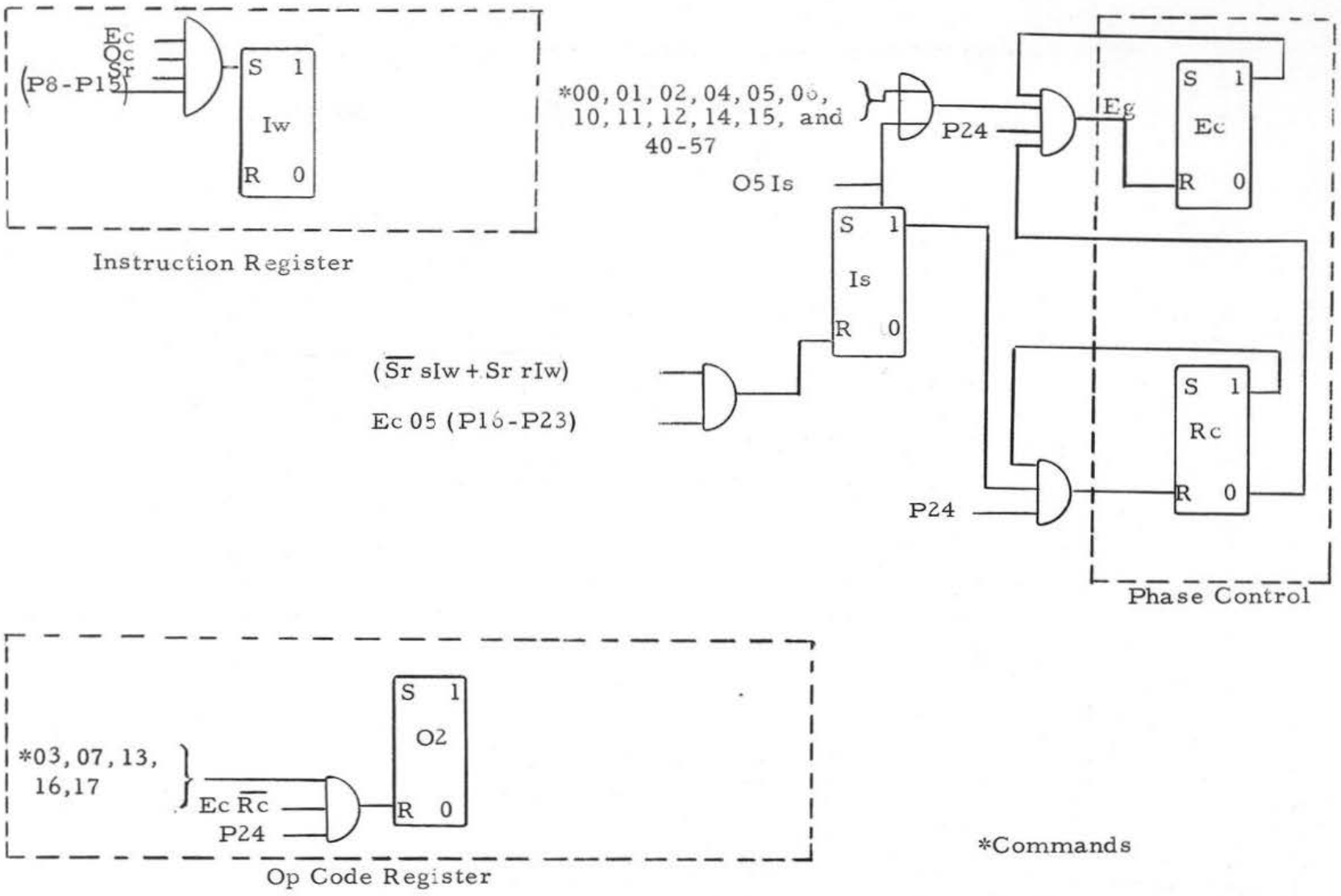


Figure 2-11. Phase 4 Termination Logic Diagram (Sheet 1 of 2)

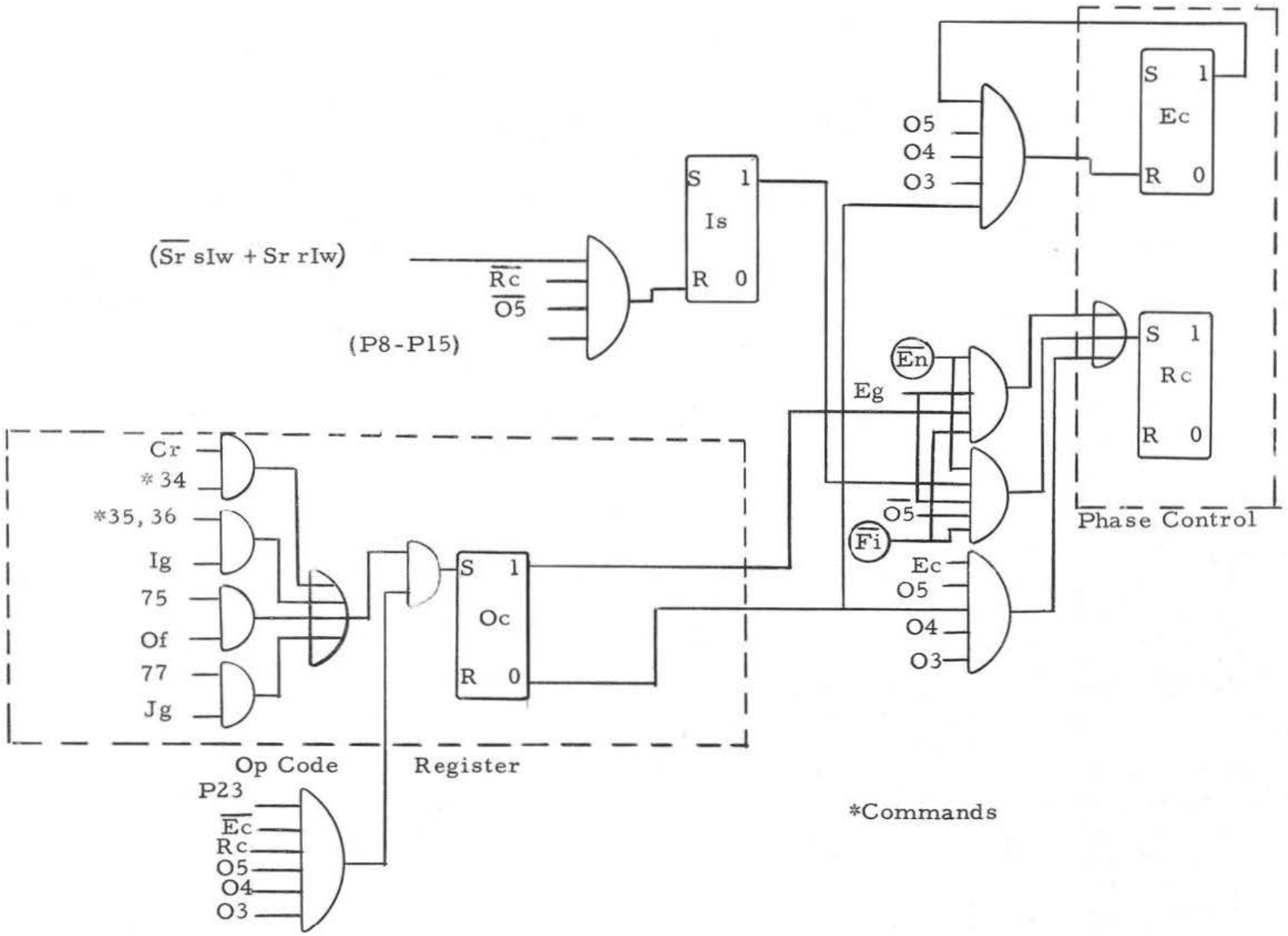


Figure 2-11. Phase 4 Termination Logic Diagram (Sheet 2 of 2)

instruction register is advanced by writing the number in the sector counter into the instruction register.

$$sIw = - - - + Ec\ Oc\ (P8\text{-}P15)\ Sr + - - -$$

This next command number is used only if computation is interrupted and then restarted, since Oc in its "true" condition normally causes phase 2 at the end of phase 4 (see paragraph D).

$$sRc = \overline{En}\ \ \overline{Fi}\ \ Eg\ Oc + - - -$$

During phase 4 of commands 00 through 17, and 40 through 57, the next command sector number is compared with the sector counter to determine whether phase 2 should be entered when phase 4 is terminated (see Sheet 2, Figure 2-11).

$$rIs = (\overline{Sr}\ sIw + Sr\ rIw)\ [- - - + \overline{Rc}\ \overline{O5}\ (P8\text{-}P15)]$$
$$sRc = - - -\ \overline{En}\ \ \overline{Fi}\ \ Eg\ \overline{O5}\ Is + - - -$$
$$rRc = P24\ Rc\ Is + - - -$$

Commands 34, 35, 36, 37, 75, and 77 operate somewhat differently. At the end of reading one of these commands, Oc will be set if the branch condition being tested is set (see Sheet 2, Figure 2-11).

$$sOc = - - - + P23\ \overline{Ec}\ Rc\ O5\ O4\ O3\ (\overline{O6}\ Ig + O6\ \overline{O2}\ O1\ Of + O6\ O2\ O1\ Jg + \overline{O2}\ \overline{O1}\ Cr)$$

These terms include Ig for the signs of the A and B registers; Cr for the sign of the C register; Of for overflow; and Jg for a selected branch control input line (see paragraph K for the construction of Jg). Phase 4 will immediately follow phase 2 because of the P24 Rc Is term of rRc but, if Oc is not set, phase 4 will immediately be changed to phase 2.

$$rEc = - - - + Ec\ O5\ O4\ O3\ \overline{Oc} + - - -$$
$$sRc = - - - + Ec\ O5\ O4\ O3\ \overline{Oc}$$

If Oc was set, phase 4 will extend until the address sector number agrees with the sector counter, and phase 2 will follow phase 4. When commands 34, 35, 36, 37, and 75 enter phase 4 due to Oc being set, the command line selector flip-flops are changed by the contents of the operand line selector flip-flops.

$$sK4 = Kg\,L4 \qquad sK2 = Kg\,L2$$
$$rK4 = Kg\,\overline{L4} + - - - \qquad rK2 = Kg\,\overline{L2} + - - -$$
$$sK3 = Kg\,L3 \qquad sK1 = Kg\,L1 + - - -$$
$$rK3 = Kg\,\overline{L3} + - - - \qquad rK1 = Kg\,\overline{L1}$$
$$Kg = Ec\,O5\,O4\,O3\,F4\,(\overline{O6} + \overline{O2})$$

2-8. MANUAL CONTROLS

A minimum of manual control is included. Depressing the ENABLE switch holds the computer in phase 1 by blocking the sRc terms. If the γI key is depressed while the ENABLE switch is closed, the command line selector register is set to line 01 and the next command sector is reset to zero (see Figure 2-12).

$$rKs = - - - + \text{(En)}\ \text{(T6)}\ \text{(T5)}\ \text{(T4)}\ \text{(T1)}\ \text{(P23)}$$
$$rK3 = - - - + \text{(En)}\ \text{(T6)}\ \text{(T5)}\ \text{(T4)}\ \text{(T1)}\ \text{(P23)}$$
$$rK2 = - - - + \text{(En)}\ \text{(T6)}\ \text{(T5)}\ \text{(T4)}\ \text{(T1)}\ \text{(P23)}$$
$$sK1 = - - - + \text{(En)}\ \text{(T6)}\ \text{(T5)}\ \text{(T4)}\ \text{(T1)}\ \text{(P23)}$$
$$sIw = \overline{+ - - -} + Ir\,\overline{[- - - + \text{(En)}\ \text{(T6)}\ \text{(T5)}\ \text{(T4)}\ \text{(T1)}\ \text{(P8-P15)}}$$

If the "C" key is depressed while the ENABLE switch is closed, the computer will read and execute one command. The Oc flip-flop provides the enabling conditions for this one cycle operation.

$$sOc = - - - + \text{(En)}\ P24\,Is\,\overline{Ec}\,\overline{Rc}\ \text{(T6)}\ \text{(T5)}\ \text{(T2)}\ \text{(T1)} + - - -$$
$$rOc = - - - + Eg$$
$$sRc = \overline{+ - - -} + P24\,\overline{Ec}\,\overline{Rc}\,Is\ \left(\text{(En)} + Oc\ \overline{\text{(T6)}\ \text{(T5)}\ \text{(T2)}\ \text{(T1)}}\right)\text{(F1)}$$

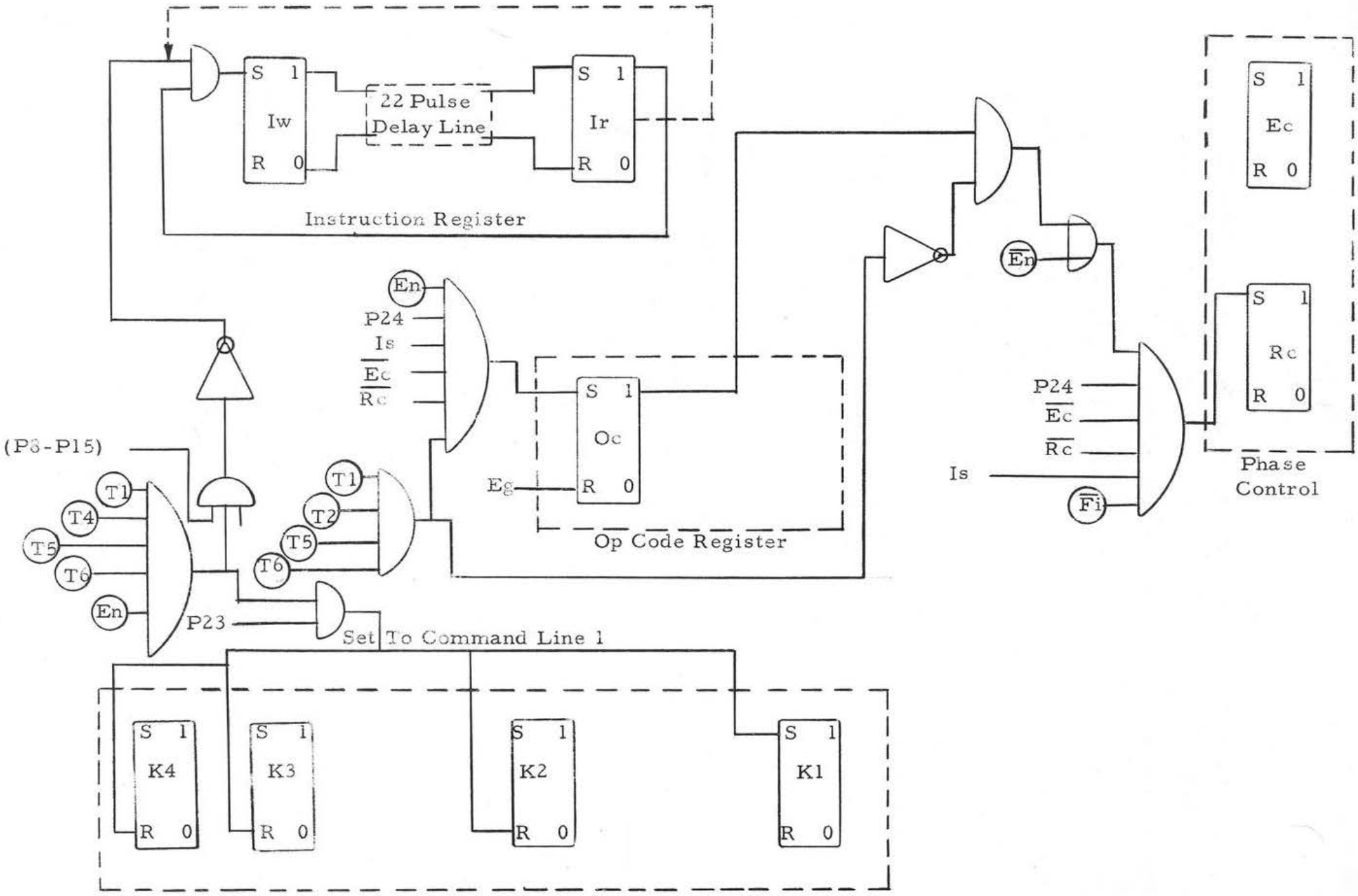


Figure 2-12. "I" and "C" Key Control Logic Diagram

The storage of a line number in the index register is achieved by sending it to line $(37)_8$. The useful portion of this number (bit positions 3 through 7) is entered into the instruction register.

$$sIw = - - - + \overline{P24}\,(P24\text{-}P7)\ M3g\ N7g\ Wg\ Ig + - - -$$

When transferred to the operand line selector register, the number in the instruction register (bit positions 4 through 8) is taken from the instruction register write flip-flop, Iw, rather than from the instruction register read flip-flop, Ir.

Phase 2 or phase 4 can be immediately changed to phase 1 if a parity error is indicated by the Pc flip-flop (see Figure 2-13).

$$rEc = - - - + Ec\,\overline{Rc}\,Pc\,P1$$
$$rRc = - - - + \overline{Ec}\,Rc\,Pc\,P1$$

A parity error will halt computation until the Pc flip-flop is cleared by depressing the ENABLE switch and the BREAK POINT switch.

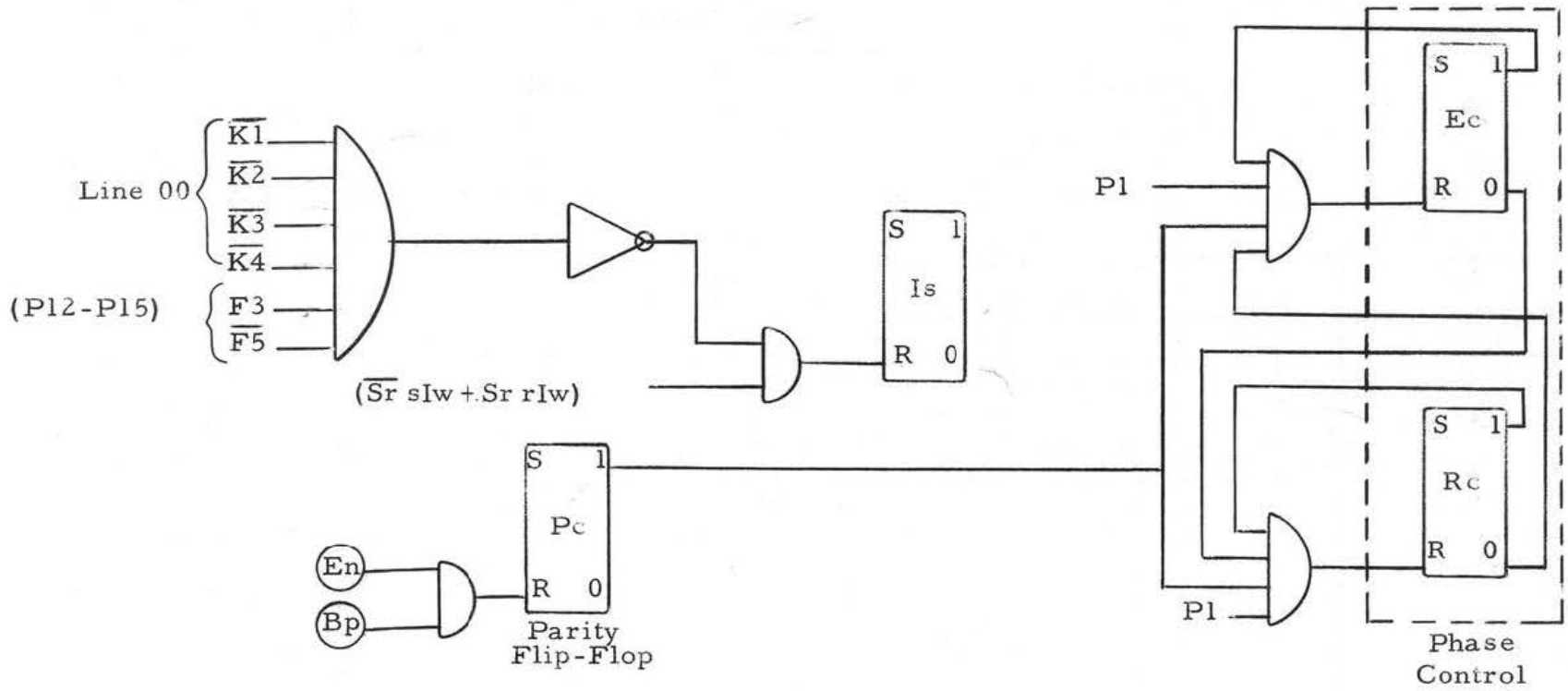
$$rPc = - - - + (En)\ (Bp) + - - -$$

To provide modulo 16 operation of the command sequence, when line 00 is the source of commands, pulse positions 12 through 15 are blocked from the comparison in the Is flip-flop (see Figure 2-13).

$$rIs = (\overline{Sr}\,sIw + Sr\,rIw)\ [- - -]\ \overline{[\overline{K4}\,\overline{K3}\,\overline{K2}\,\overline{K1}\,\overline{F5}\,F3]}$$

E. DATA TRANSFER

The basic data transfers are "fetches" from the memory lines to the registers, and "stores" from the registers to the memory lines. The data stored is formed by the Ig gate (see Figure 2-14). This gate presents the contents of the C register for command 10, the contents of the A register for commands 01, 11, 41, 25, and 35, and the contents of the B register



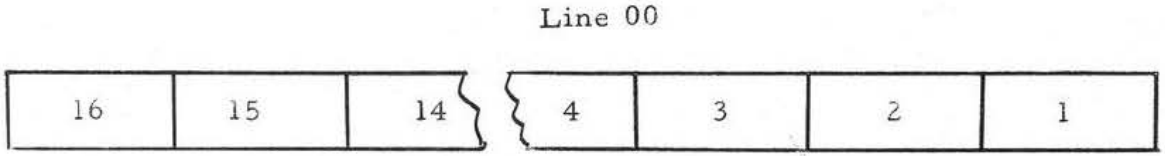


Figure 2-13. Parity Block, Modulo 16 Operation Logic Diagram

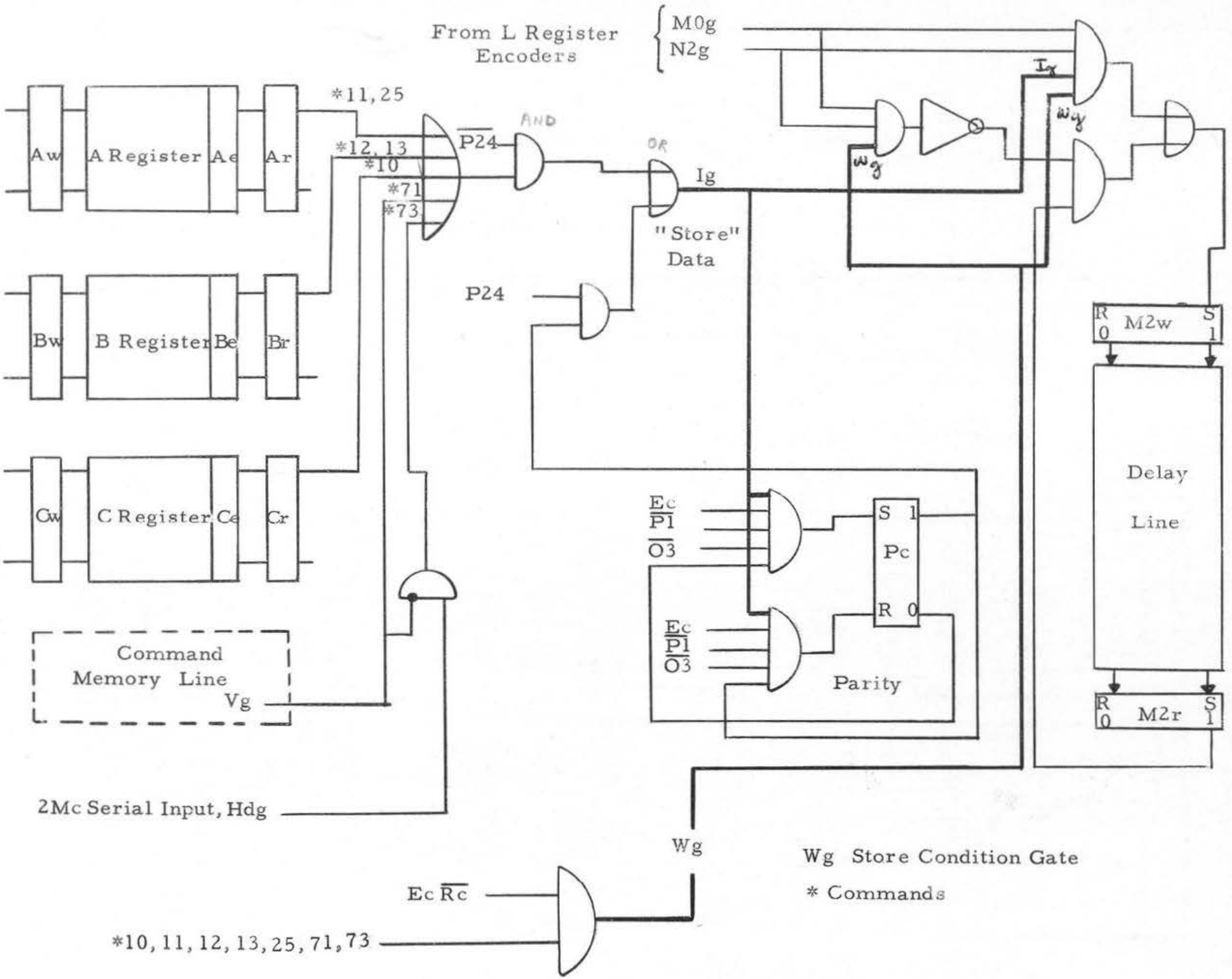


Figure 2-14. Data Transfer-"Stores" Logic Diagram

for commands 02, 03, 12, 13, and 36.

$$Ig = \overline{P24}\,\overline{O5}\,\overline{O2}\,\overline{O1}\,Cr + \overline{P24}\,\overline{O6}\,\overline{O2}\,O1\,Ar + \overline{P24}\,\overline{O6}\,O2\,Br + \overline{P24}\,\overline{O4}\,\overline{O2}\,Ar + - - -$$

For command 71, the stored data comes from the command line.

$$Ig = - - - + \overline{P24}\,O6\,O5\,\overline{O2}\,Vg + - - -$$

For command 73, the stored data comes from an input signal, Hdg, gated by Vg.

$$Ig = - - - + \overline{P24}\,O6\,O5\,O2\,Vg\,Hdg + - - -$$

The last bit of each stored word is a parity bit to produce an even number of "1's" in each word (excluding P1).

$$Ig = - - - + P24\,Pc$$
$$sPc = Ec\,\overline{Rc}\,\overline{P1}\,\overline{\overline{O5}\,O3}\,Ig\,\overline{Pc} + - - -$$
$$rPc = Ec\,\overline{Rc}\,\overline{P1}\,\overline{\overline{O5}\,O3}\,Ig\,Pc + - - -$$

Writing into a memory line is conditioned by phase 4 and commands 10, 11, 12, 13, 71, or 25.

$$Wg = \left(\overline{O6}\,\overline{O5}\,O4\,\overline{O3} + O6\,O5\,O4\,\overline{O3}\,O1 + \overline{O6}\,O5\,\overline{O4}\,O3\,\overline{O2}\,O1\right) Ec\,\overline{Rc}$$

Writing into a particular memory line is conditioned by the operand line selector. For example, to write in memory line 02

$$sM2w = M0g\,N2g\,Wg\,Ig + \boxed{\overline{M0g\,N2g\,Wg}}\;\boxed{M2r}$$
$$rM2w = M0g\,N2g\,Wg\,\overline{Ig} + \boxed{\overline{M0g\,N2g\,Wg}}\;\boxed{\overline{M2r}}$$

where

$$\begin{bmatrix} M0g = \overline{L0}\,\overline{L5}\,\overline{L4} \\ M1g = \overline{L0}\,\overline{L5}\,L4 \\ M2g = \overline{L0}\,L5\,\overline{L4} \\ \text{etc.} \end{bmatrix}$$

and

$$\begin{bmatrix} N0g = \overline{L3}\,\overline{L2}\,\overline{L1} \\ N1g = \overline{L3}\,\overline{L2}\,L1 \\ N2g = \overline{L3}\,L2\,\overline{L1} \\ \text{etc.} \end{bmatrix}$$

To make "fetches" from the memory, the line is selected by the address line selector (see Figure 2-15).

$$Fg = M0g\,N0g\,M0r + M0g\,N1g\,M1r + - - + M1g\,N7g\,M15r + M3g\,N7g\,Ir + Nxg$$

Parity is checked for commands 04, 05, 06, and 07, during phase 4 as well as during phase 2.

$$sPc = - - - + \overline{P1}\,Ec\,\overline{Rc}\,\overline{O6}\,\overline{O5}\,O3\,Fg\,\overline{Pc} + \overline{P1}\,\overline{Ec}\,Rc\,Vg\,\overline{Pc} + - - -$$

$$rPc = - - - + \overline{P1}\,Ec\,\overline{Rc}\,\overline{O6}\,\overline{O5}\,O3\,Fg\,Pc + \overline{P1}\,\overline{Ec}\,Rc\,Vg\,Pc + - - -$$

If the parity is odd, the Pc flip-flop will block computation. The data selected by the "fetch" command is sent to the proper register. For commands 05 and 25, data is sent to A, for O6 and O7 to B, and for O4 to C.

$$sAw = Ec\,\overline{Rc}\,\overline{O6}\,\overline{O4}\,O3\,\overline{O2}\,O1\,Fg + - - -$$

$$sBw = Ec\,\overline{Rc}\,\overline{O6}\,\overline{O5}\,\overline{O4}\,O3\,O2\,Fg + - - -$$

$$sCw = Ec\,\overline{Rc}\,\overline{O6}\,\overline{O5}\,\overline{O4}\,O3\,\overline{O2}\,\overline{O1}\,Fg + - - -$$

Command 01 provides an interchange between the A and C registers, and command 02 provides an interchange between the B and C registers.

$$sCw = - - - + Ec\,\overline{Rc}\,\overline{O6}\,\overline{O5}\,\overline{O4}\,\overline{O3}\,Ig + - - -$$

$$sAw = - - - + Ec\,\overline{Rc}\,\overline{O6}\,\overline{O5}\,\overline{O4}\,\overline{O3}\,\overline{O2}\,O1\,Cr + - - -$$

$$sBw = - - - + Ec\,\overline{Rc}\,\overline{O6}\,\overline{O5}\,\overline{O4}\,\overline{O3}\,O2\,Cr + - - -$$

Command 03 provides a rotation (interchange) of information between the A, B and C registers.

During the first sector of execution of the 03 command the B and C registers interchange. At the end of the first sector of execution, the 03 command is changed to a 01 command by resetting the O2 flip-flop. During the second sector of execution the A and C registers interchange. The results of this command is that the C register is copied into the B register, the

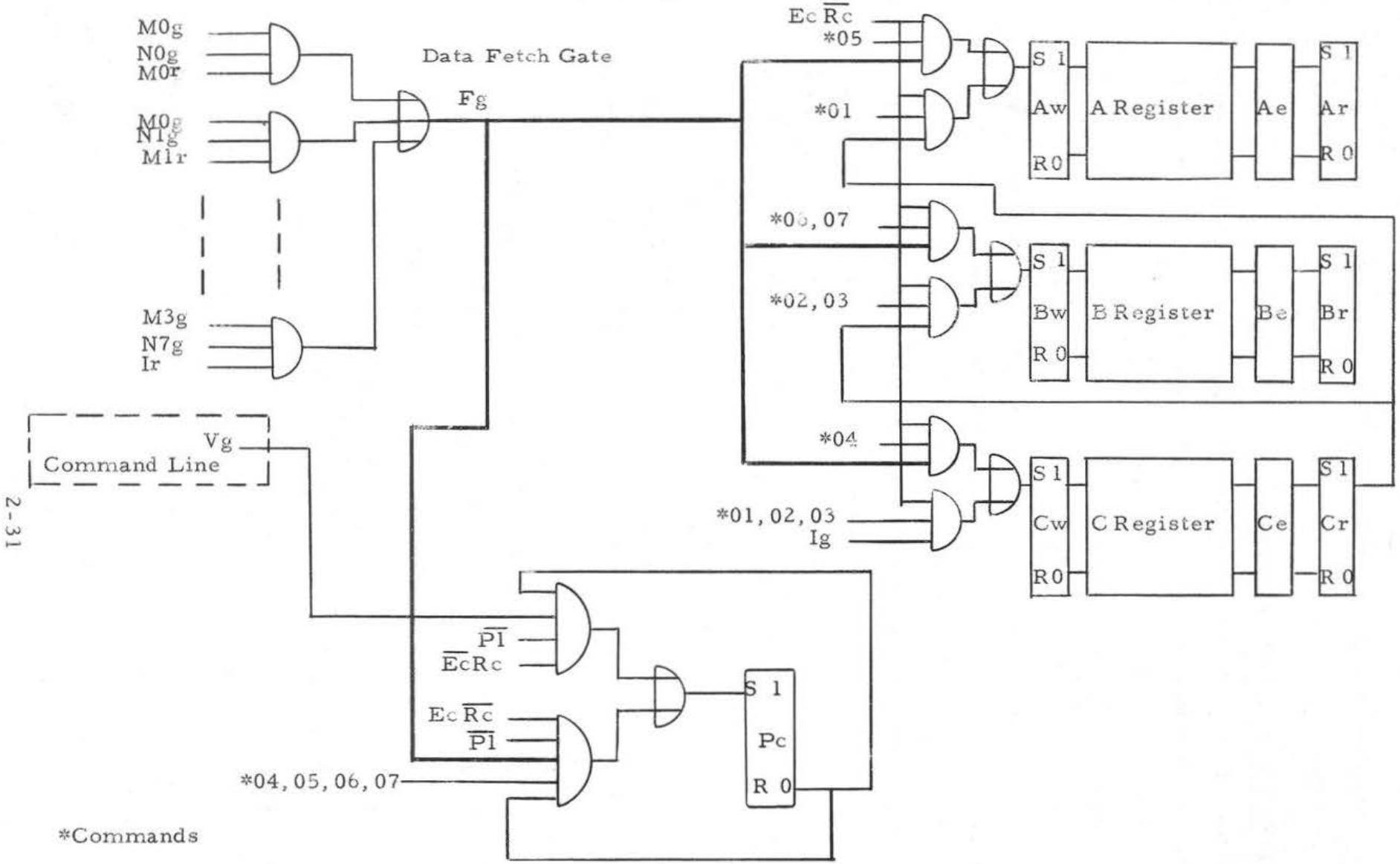


Figure 2-15. "Fetch" Data Transfer Logic Diagram

B register is copied into the A register, and the A register is copied into the C register.

F. ADDITION AND SUBTRACTION

Commands 14, 15, 16, and 17 provide addition and subtraction into the A register and AB register pair (see Figure 2-16). The sum entered into these registers is formed in the adder.

$$Zg = (\overline{\overline{Zg}})$$

$$\overline{Zg} = Xg\ Yg\ \overline{Ca} + Xg\ \overline{Yg}\ Ca + \overline{Xg}\ Yg\ Ca + \overline{Xg}\ \overline{Yg}\ \overline{Ca}$$

The input term from the memory is entered in "true" form for addition (commands 14 and 16) and inverted for subtraction (commands 15 and 17).

$$Xg = - - - + \overline{O5}\ \overline{O1}\ Fg + \overline{O5}\ O1\ \overline{Fg} + - - -$$

The input term from the registers is taken from the B register for commands 16 and 17, and from the A register for commands 14 and 15. For the double precision commands 16 and 17, O2 is reset at the end of the first sector of execution which changes the commands from 16 to 14 and from 17 to 15.

$$Yg = - - - + \overline{O6}\ O4\ \overline{O2}\ Ar + \overline{O5}\ O2\ Br + - - -$$

The adder carry flip-flop, Ca, is cleared before any execution and set at the start of execution of commands 15 and 17 to add the "true" complement of the input term from the memory.

$$sCa = \overline{P1}\ \overline{P24}\ \overline{Ca}\ Ec\ \overline{Rc}\ Xg\ Yg + P24\ Rc\ Is\ \overline{O6}\ \overline{O5}\ O4\ O3\ O1 + - - -$$

$$rCa = \overline{P1}\ \overline{P24}\ \overline{Xg}\ \overline{Yg} + Ca\ Rc + - - -$$

The resulting sum is entered into the B and A registers.

$$sBw = - - - + Ec\ \overline{Rc}\ \overline{O6}\ \overline{O5}\ O4\ O3\ O2\ Zg + - - -$$

$$sAw = - - - + Ec\ \overline{Rc}\ \overline{O6}\ \overline{O5}\ O4\ O3\ \overline{O2}\ Zg + - - -$$

Overflow at the sign position is used to set the Of flip-flop. An overflow occurs when the two adder inputs have the same sign and the result has

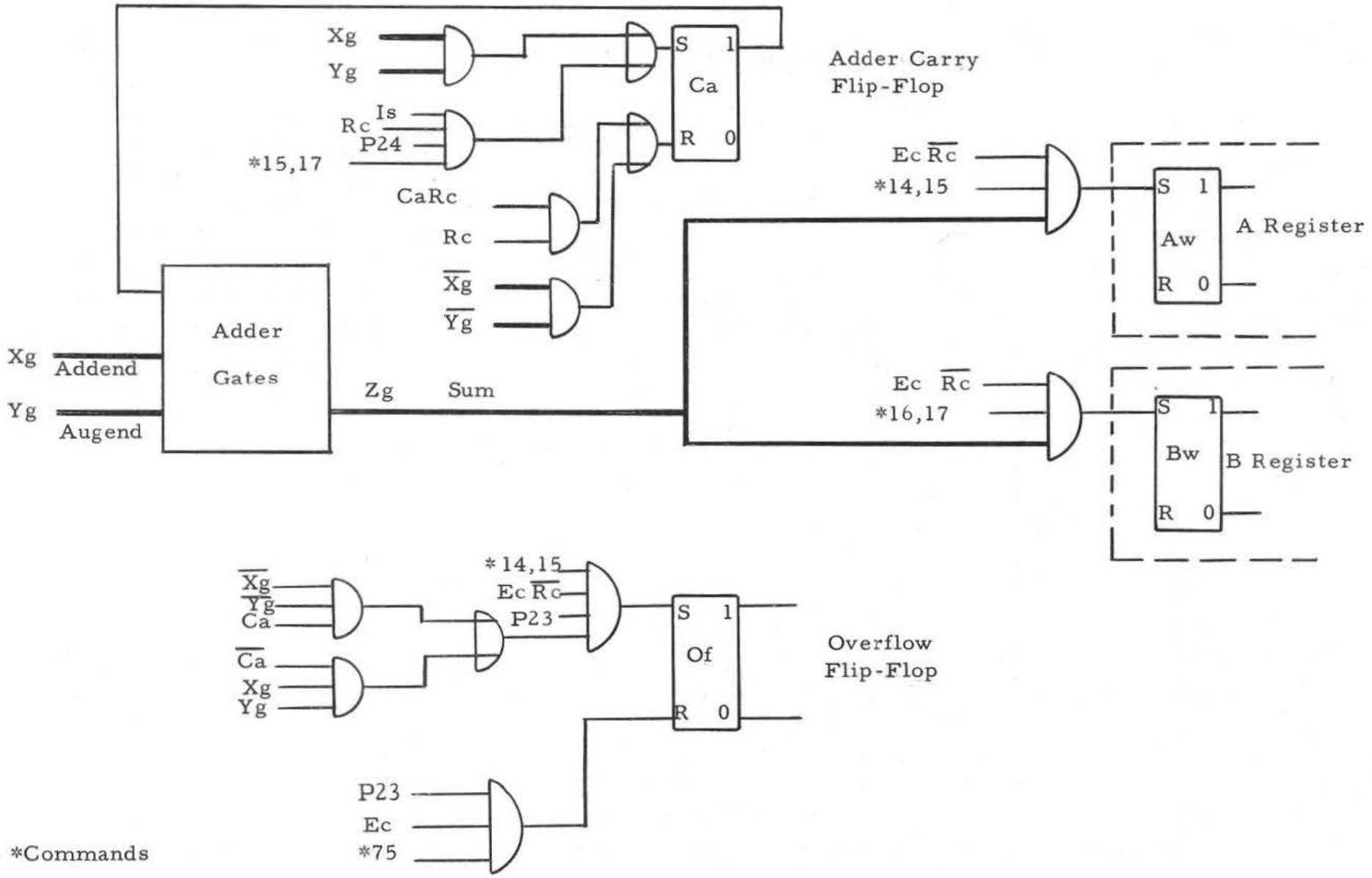


Figure 2-16. Addition and Subtraction Logic Diagram

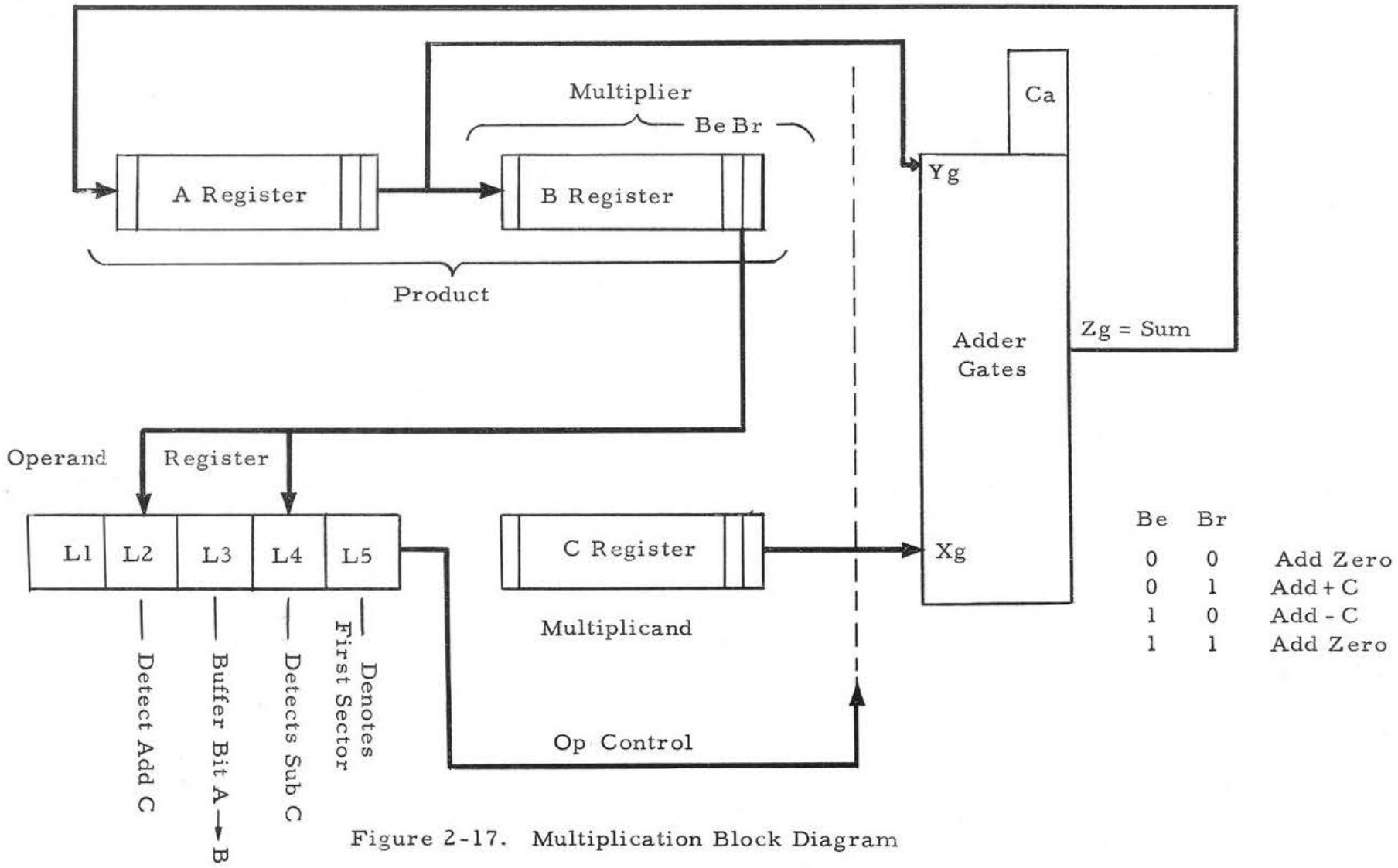


Figure 2-17. Multiplication Block Diagram

a different sign.

$$sOf = P23\ Ec\ \overline{Rc}\ \overline{O6}\ \overline{O5}\ O4\ O3\ \overline{O2}\ (\overline{Xg}\ \overline{Yg}\ Ca + Xg\ Yg\ \overline{Ca}) + - - -$$

Of is reset by the execution of command 75 that tests for overflow.

$$rOf = P23\ Ec\ O6\ O5\ O4\ O3\ \overline{O2}\ O1 + - - -$$

For commands 14, 15, 16, and 17, the parity of the information taken from the memory is checked through the Pc flip-flop.

G. MULTIPLICATION

During the execution of command 32, the contents of the C register are multiplied by the contents of the B register (see Figure 2-17). The contents of the AB register pair are shifted to the right by one bit position for each sector of the multiplication operation, which placed the multiplier bits at the right end of the B register. Based on these multiplier bits, the A register has the contents of the C register added to it, subtracted from it, or a zero added to it. By this process the most significant bits of the product appear in the A register, and the least significant bits of the product appear in the B register.

The first sector of multiplication operation is marked by L5. At the end of the first sector, L5 is set (see Sheet 1, Figure 2-18).

$$sL5 = - - - + \overline{O6}\ O5\ \overline{O3}\ P24\ Ec$$

The multiplier bit code to add (C) is detected by L2.

$$sL2 = - - - + \overline{O6}\ O5\ O4\ \overline{O3}\ O2\ \overline{O1}\ \overline{Be}\ Br\ P1 + - - -$$
$$rL2 = - - - + \overline{O6}\ O5\ O4\ \overline{O3}\ O2\ \overline{O1}\ P24$$

The multiplier bit code to subtract (C) is detected by L4.

$$sL4 = - - - + \overline{O6}\ O5\ O4\ \overline{O3}\ O2\ \overline{O1}\ P1\ Be\ \overline{Br}$$
$$rL4 = - - - + \overline{O6}\ O5\ O4\ \overline{O3}\ O2\ \overline{O1}\ P24$$

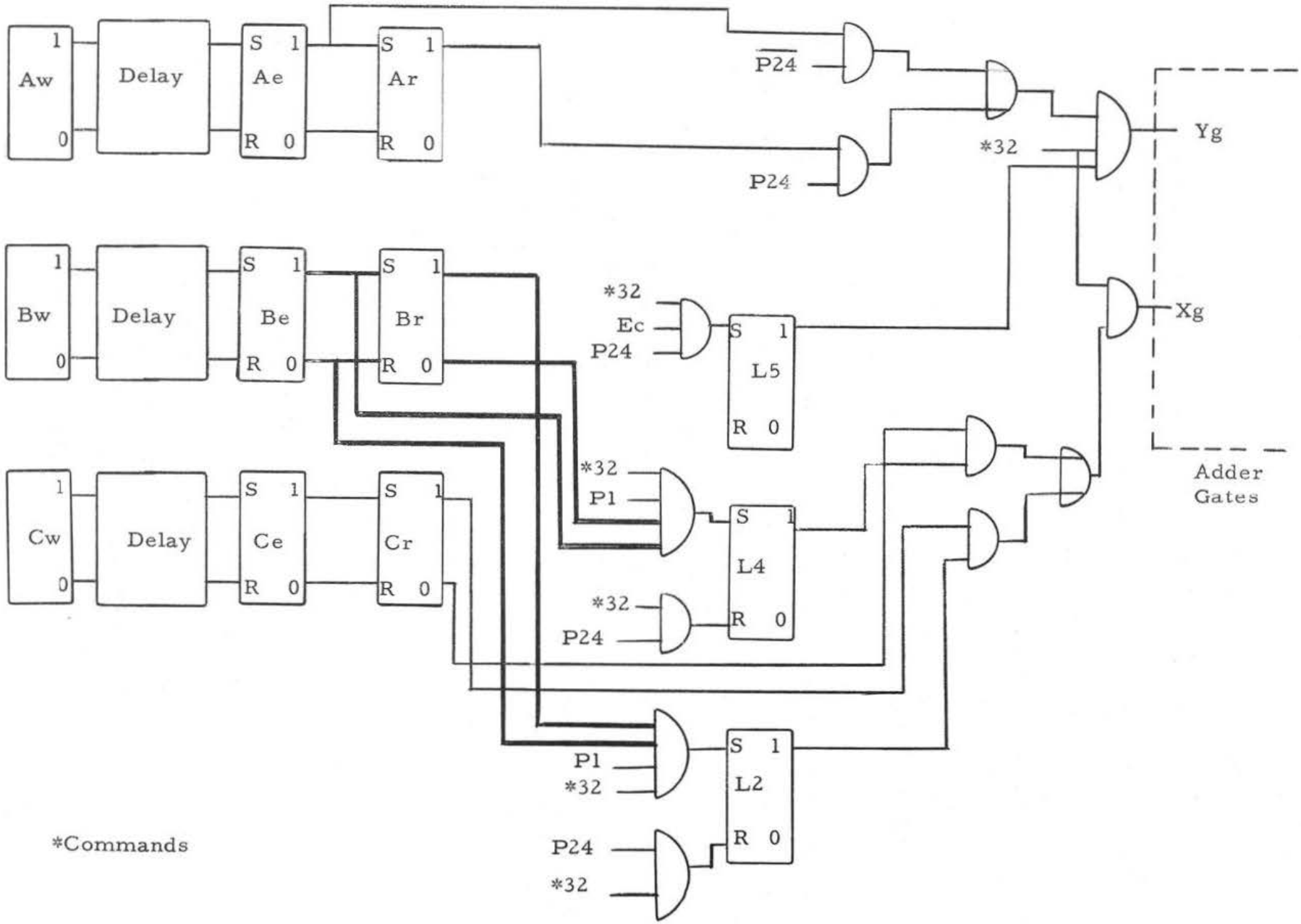


Figure 2-18. Multiplication Logic Diagram (Sheet 1 of 2)

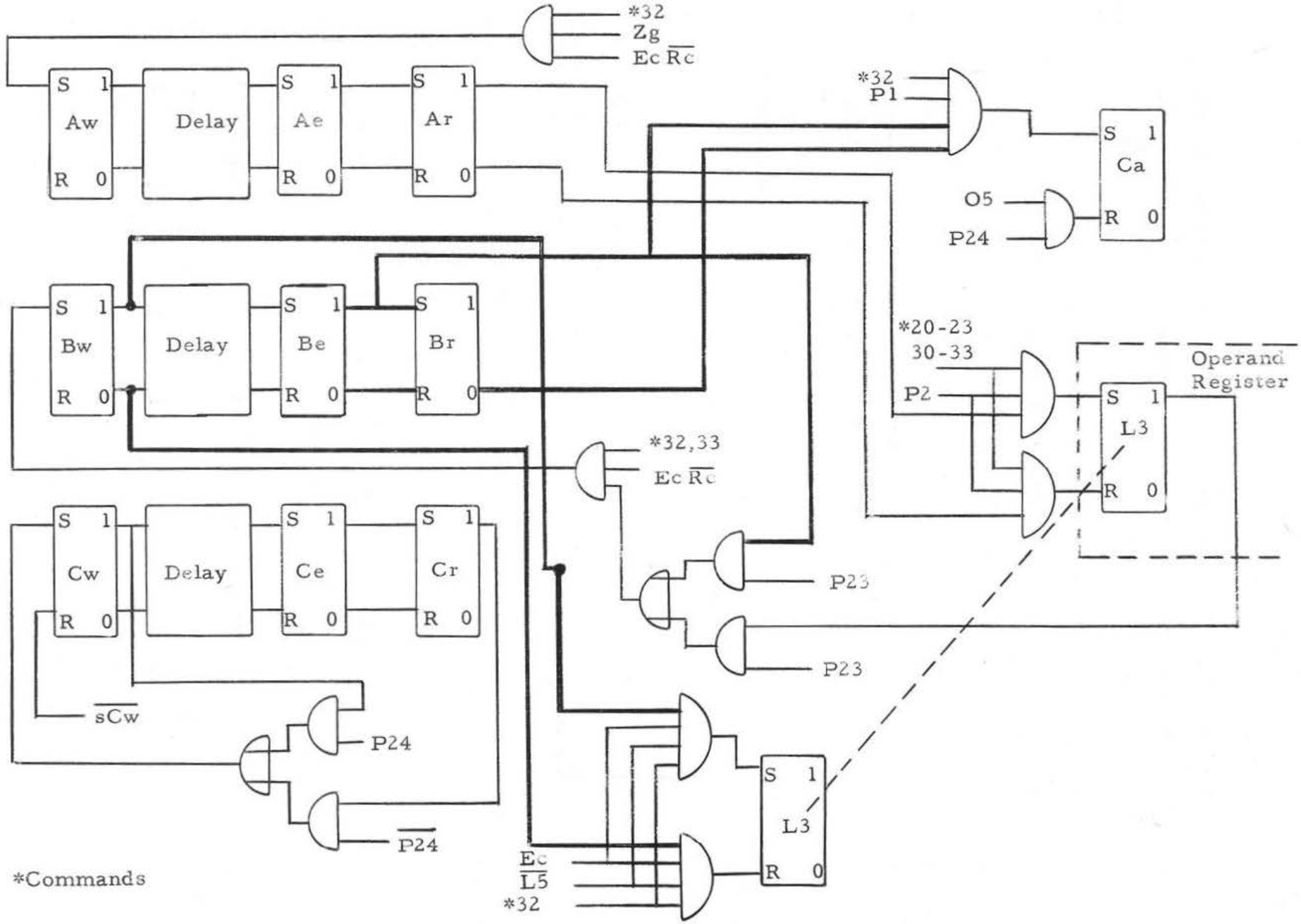


Figure 2-18. Multiplication Logic Diagram (Sheet 2 of 2)

The adder inputs, with the A register shifted to the right through Ae, and the entry of the C register controlled by L2 and L4, are as follows (see Sheet 1, Figure 2-18):

$$Xg = - - - + \overline{O6}\ O5\ O4\ \overline{O3}\ O2\ \overline{O1}\ (L2\ Cr + L4\ \overline{Cr}) + - - -$$

$$Yg = - - - + \overline{O6}\ O5\ O4\ \overline{O3}\ O2\ L5\ (\overline{P24}\ Ae + P24\ Ar) + - - -$$

$$sCa = - - - + \overline{O6}\ O5\ O4\ \overline{O3}\ O2\ \overline{O1}\ P1\ Be\ \overline{Br}$$

$$rCa = - - - + O5\ P24$$

Qualifying Yg by L5 clears the A register during the first sector of operation. When subtracting, the adder carry flip-flop, Ca, is set to obtain the "true" complements of the C register.

The output of the adder, Zg, is recorded in the A register (see Sheet 2, Figure 2-18).

$$sAw = - - - + Ec\ \overline{Rc}\ \overline{O6}\ O5\ O4\ \overline{O3}\ O2\ Zg + - - -$$

To start the multiplier bit code properly, the P1 bit position of the B register is kept clear during all recirculation. During multiplication, the B register is shifted to the right, by the B register early flip-flop, Be; and the bits from the A register enter the left end of the B register through the operand line selector register flip-flop, L3.

$$sBw = - - + Ec\ \overline{Rc}\ \overline{O6}\ O5\ \overline{O3}\ O2\ (\overline{P23}\ Be + P23\ L3) + \overline{P1}\ Br\ (\quad) + - - -$$

The bits shifted from the right end of the A register are detected and stored in L3 for entry into the left end of the B register.

$$sL3 = - - - + \overline{O6}\ O5\ \overline{O3}\ (- - - + P2\ Ar) + - - -$$

$$rL3 = - - - + \overline{O6}\ O5\ \overline{O3}\ (- - - + P2\ \overline{Ar}) + - - -$$

During the first sector of multiplication, the sign of the B register is retained in P23 of the B register so that the last sector of multiplication will

operate correctly. The $\overline{L5}$ signal is used to repeat the sign.

$$sL3 = - - - - + \overline{O6}\ O5\ O4\ \overline{O3}\ O2\ \overline{O1}\ Ec\ \overline{L5}\ Be$$

$$rL3 = - - - - + \overline{O6}\ O5\ O4\ \overline{O3}\ O2\ \overline{O1}\ Ec\ \overline{L5}\ \overline{Be}$$

For proper operation of the sign of the product in the A register, the sign bit of the C register is extended into P24.

$$sCw = - - - - + P24\ Cw + \overline{P24}\ Cr\left[\overline{\quad}\right] + - - -$$

The following timing chart indicates that process of multiplication for a shortened set of registers. The multiplier has four bits, a sign, and the sign repeated during the first sector of multiplication. The bit pair at the right end of the B register controls the arithmetic operation as follows:

00 add zero
01 add + (C)
10 add - (C)
11 add zero

(C) = +9/16

	(A)	(B) = +13/16	
	0 0.1 0 0 1 X		
	X - - - - - X	X 0.1 1 0 1 0	
1)	1 1.0 1 1 1 X	X 0.0 1 1 0 1	-
2)	0 0.0 1 0 0 X	X 1.0 0 1 1 0	+
3)	1 1.1 0 0 1 X	X 0.1 0 0 1 1	-
4)	1 1.1 1 0 0 X	X 1.0 1 0 0 1	0
5)	0 0.0 1 1 1 X	X 0.1 0 1 0 0	+
6)	0 0.0 0 1 1	X 1.0 1 0 1 0	0

(AB) = 117/256 (1/2)

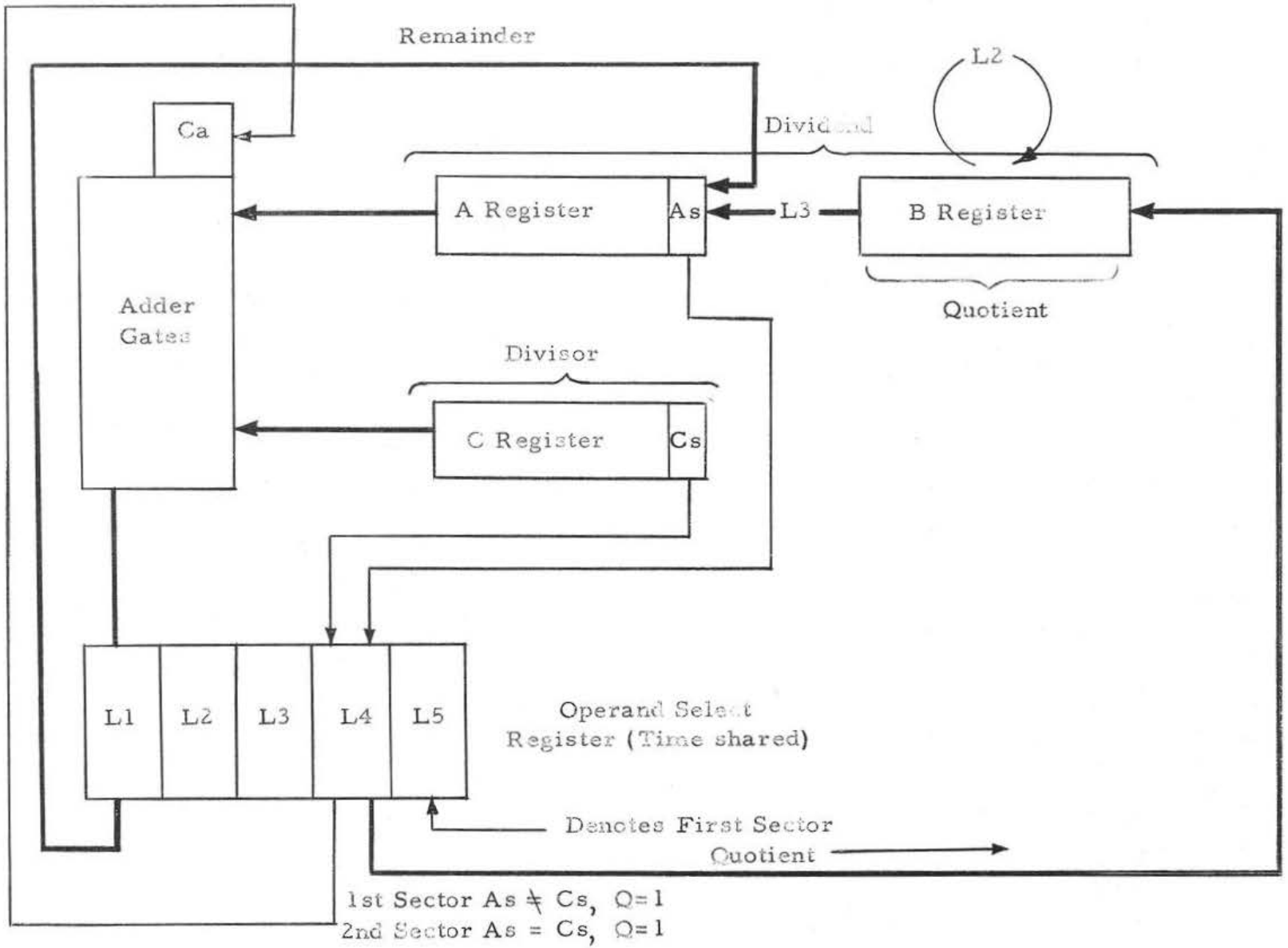
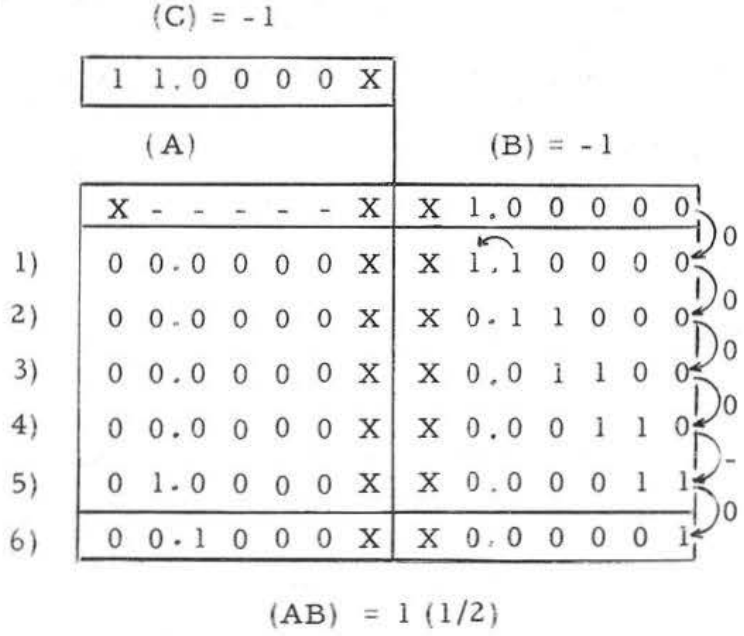


Figure 2-19. Division Block Diagram

For the sixth (last) sector, the sign and the repeated sign are used to provide a shift only, and produce 1/2 (C) (B) in AB. This last step is to ensure the correct product when B and C start as -1, as shown in the following timing chart.



(AB) = 1 (1/2)

H. DIVISION

During the execution of command 31, the contents of the AB register pair are divided by the contents of the C register (see Figure 2-19). During each sector of the division operation, the divisor is added to or subtracted from the remainder, depending on whether the sign of the divisor matches the sign of the remainder. For each sector of operation, the remainder is doubled by shifting the sum of the A and C registers (A ±C) to the left one bit position in the A register, and shifting the B register to the left one bit position. The bits of the quotient are entered into the right end of the B register as a function of the sign of each remainder.

The first sector of division operation is marked by $\overline{L5}$. At the end of

the first sector, L5 is set by the same term used for multiplication. The signs of the A and C registers are compared and, if the signs are the same, L4 is set to indicate that subtraction of the C register is required (see Sheet 1, Figure 2-20). This comparison is made while the division command is being read as well as during execution of the division command.

The adder inputs for division are (see Sheet 2, Figure 2-20):

$$Xg = \overline{O6}\ O5\ O4\ \overline{O3}\ \overline{O2}\ (\overline{L4}\ Dg + L4\ \overline{Dg}) + - - -$$

$$Yg = - - - + \overline{O6}\ O4\ \overline{O2}\ Ar + - - -$$

$$Dg = O1\ Cr + - - -$$

$$sCa = - - - + \overline{O6}\ O5\ O4\ \overline{O3}\ \overline{O2}\ P1\ L4$$

$$rCa = - - - + O5\ P24$$

When subtracting, the adder carry flip-flop, Ca, is set to obtain the "true" complement of the C register.

The output of the adder, Zg, is delayed through L1 and recorded into the A register (see Sheet 1, Figure 2-20).

$$sL1 = - - - + \overline{O6}\ O5\ O4\ \overline{O3}\ \overline{O2}\ Ec(Zg\ O4 + Ar\ \overline{O4}) + - - -$$

$$rL1 = - - - + \overline{O6}\ O5\ O4\ \overline{O3}\ \overline{O2}\ Ec(\overline{Zg}\ O4 + \overline{Ar}\ \overline{O4}) + - - -$$

$$sAw = - - - + Ec\ \overline{Rc}\ \overline{O6}\ O5\ \overline{O3}\ \overline{O2}\ (\overline{P23}\ \overline{P2}\ L1 + P2\ L3 + P23\ O4\ L1) + - - -$$

The storage of bits shifted from the left end of the B register for entry into the right end of the A register is handled by L3.

$$sL3 = - - - + \overline{O6}\ O5\ \overline{O3}\ (P24\ Bw + - - -) + - - -$$

$$rL3 = - - - + \overline{O6}\ O5\ \overline{O3}\ (P24\ \overline{Bw} + - - -) + - - -$$

The shifting of the contents of the B register is handled through L2.

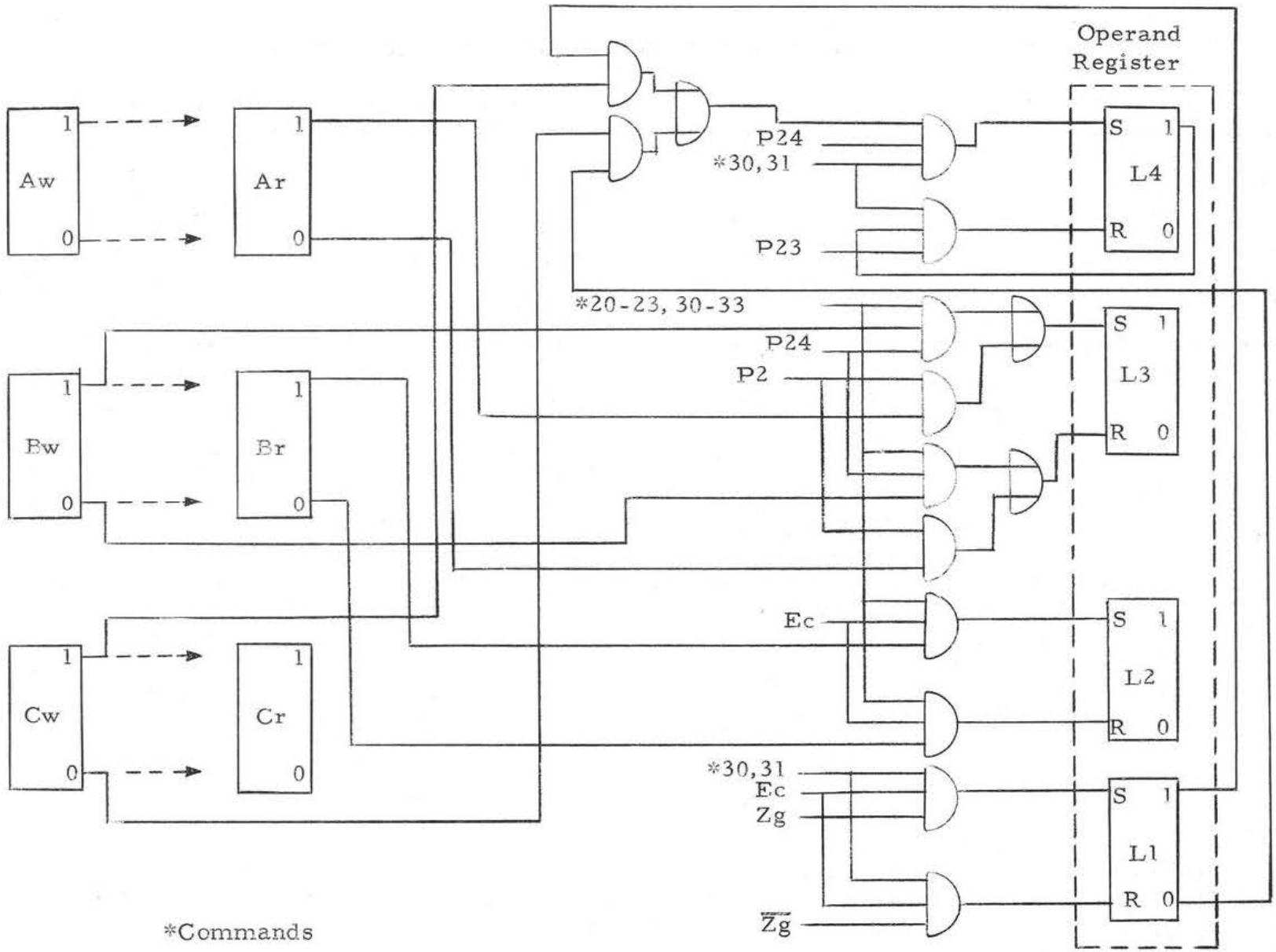


Figure 2-20. Division Logic Diagram (Sheet 1 of 2)

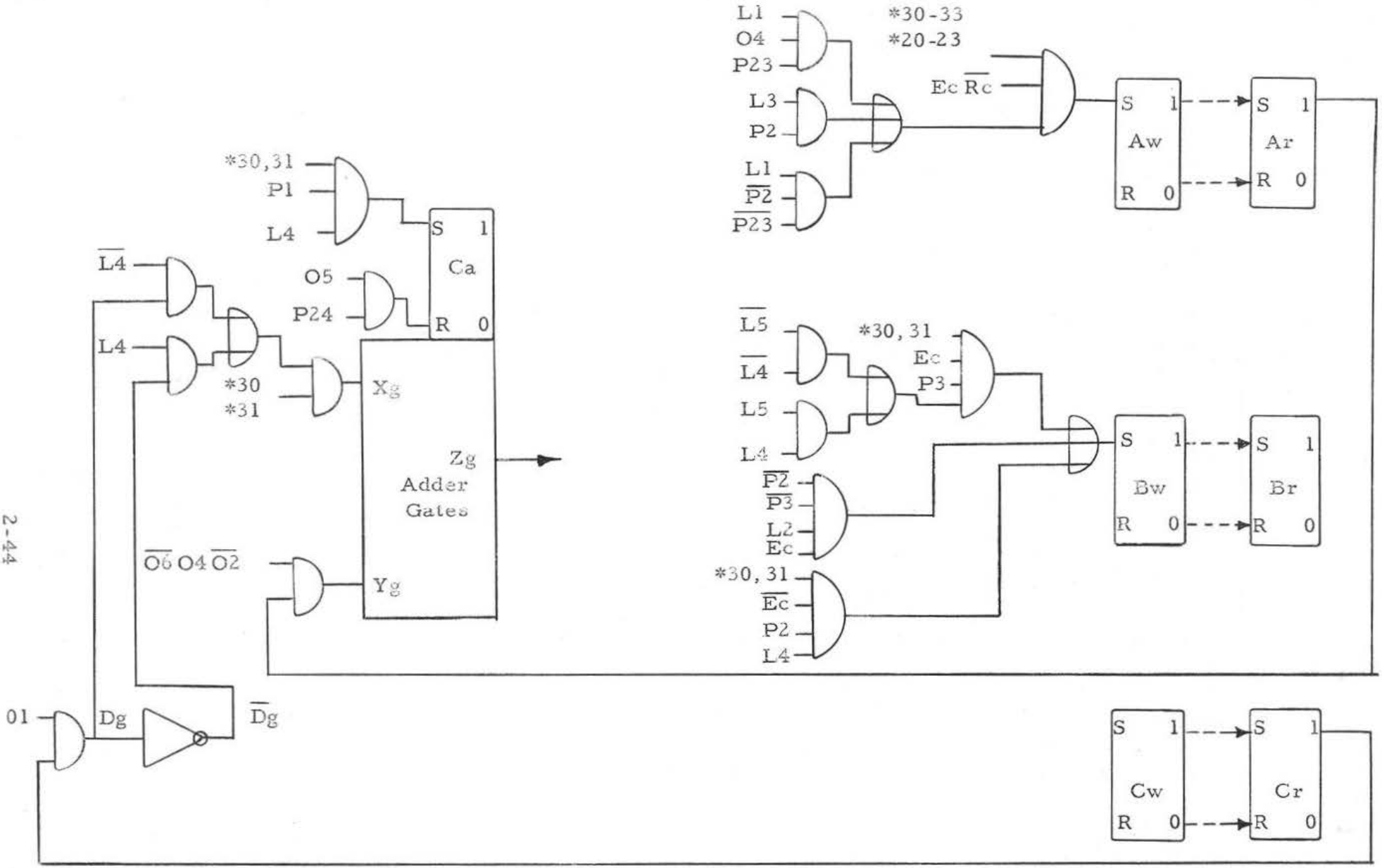


Figure 2-20. Division Logic Diagram (Sheet 2 of 2)

$$sL2 = - - - + \overline{O6}\, O5\, \overline{O3}\, \overline{O2}\, Ec\, Br + - - -$$

$$rL2 = - - - + \overline{O6}\, O5\, \overline{O3}\, \overline{O2}\, Ec\, \overline{Br} + - - -$$

$$sBw = - - - + \overline{O6}\, O5\, O4\, \overline{O3}\, \overline{O2} \left[Ec\, \overline{P2}\, \overline{P3}\, L2 \; + - - - \right]$$

During the division operation, the bits of the quotient are entered into the B register at pulse position P3.

$$sBw = - - - + \overline{O6}\, O5\, O4\, \overline{O3}\, \overline{O2} \left[Ec\, P3\, (\overline{L5}\, \overline{L4} + L5\, L4) + - - - \right]$$

Each quotient bit is a "1" when the remainder and divisor have like signs, except for the first sector of operation, when the sign of the quotient is formed. During the first sector, indicated by $\overline{L5}$, unlike signs produce a "1" for a negative quotient.

Immediately after the last sector of division, the last bit of the quotient is entered into bit position P2 of the B register.

$$sBw = - - - + \overline{O6}\, O5\, O4\, \overline{O3}\, \overline{O2} \left[\overline{Ec}\, P2\, L4 + - - - \right]$$

The L4 flip-flop compares the sign of the remainder in the A register with the sign of the divisor in the C register at P24.

$$sL4 = - - - \; \overline{O6}\, O5\, O4\, \overline{O3}\, \overline{O2}\, P24\, Ec\, (L1\, Cr + \overline{L1}\, \overline{Cr}) + - - -$$

$$rL4 = - - - \; \overline{O6}\, O5\, O4\, \overline{O3}\, \overline{O2}\, P23\, L4$$

The new sign of the register A is taken from the L1 flip-flop.

The first comparison is made as the division command is read. For normal division, the first signs are taken at P23. For division of a remainder, the first signs are taken at P24 under the control of the code bit in the L1 flip-flop.

$$sL4 = - - - + \overline{O6}\, O5\, O4\, \overline{O3}\, \overline{O2} \; \left[- - - P23\, Rc\, \overline{L1}\, \overline{Zg} + P24\, Rc\, L1\, \overline{Zg} \right]$$

When the division command is read at P23 and P24, with Ca = 0 and L4 = 0, $\overline{Zg} = Ar\, Cr + \overline{Ar}\, \overline{Cr}$.

The following timing charts show some short register division operations:

(C) = 1/2

	0	0.	1	0	0	0	X						
	(AB) = -1							I					
⊕	X	1.	0	0	0	0	X	X	0.	0	0	0	0
+	1	1.	0	0	0	0	X	X	0.	0	0	1	0	0
+	1	1.	0	0	0	0	X	X	0.	0	1	0	0	0
+	1	1.	0	0	0	0	X	X	0.	1	0	0	0	0
	1	1.	0	0	0	0	X	X	1.	0	0	0	0	0

(A) = -1 (B) = -1

(C) = +11/16

	0	0.	1	0	1	1	X						
	(AB) = 143/256							II					
⊖	X	0.	1	0	0	0	X	X	1.	1	1	1	0	0
+	1	1.	1	0	1	1	X	X	1.	1	1	0	0	0
-	0	0.	1	1	0	1	X	X	1.	1	0	0	1	0
-	0	0.	0	1	0	1	X	X	1.	0	0	1	1	0
	1	1.	0	1	0	1	X	X	0.	0	1	1	0	0

(A) = -11/16 (B) = 6/16

(C) = 3/4

	0	0	.	1	1	0	0	X						
	(AB) = 3/4								III					
⊖	X	0	.	1	1	0	0	X	X	0	.	0	0	0	0	0
-	0	0	.	0	0	0	0	X	X	0	.	0	0	0	1	0
+	1	0	.	1	0	0	0	X	X	0	.	0	0	1	0	0
+	1	0	.	1	0	0	0	X	X	0	.	0	1	0	0	0
	1	0	.	1	0	0	0	X	X	0	.	1	0	0	0	0

(A) = -6/4 (B) = +1/2

(C) =

	1	1	.	1	0	0	0	X						
	(AB) = 3/4								IV					
⊕	X	0	.	1	1	0	0	X	X	0	.	0	0	0	0	0
+	0	0	.	1	0	0	0	X	X	0	.	0	0	1	0	0
+	0	0	.	0	0	0	0	X	X	0	.	0	1	0	0	0
-	1	1	.	0	0	0	0	X	X	0	.	1	0	0	1	0
	1	1	.	0	0	0	0	X	X	1	.	0	0	1	1	0

(A) = -1 (B) = -13/16

With four-bit plus sign registers, $1/2\ \frac{(AB)}{(C)}$ is formed in the B register in four sectors of operation. The operation may be programmed for five sectors to give $\frac{(AB)}{(C)}$ if the quotient is less than one in magnitude. When dividing the non-restored remainder by the C register, the operation should be performed for five sectors to eliminate the quotient's sign. A negative divisor results in a quotient that is a "1's" complement of the desired quotient (see example 4 below).

The relationship between the uncorrected remainder in A and the true remainder, R, can be defined as follows:

$R = 1/2\ (A)$ if the sign of (A) is the same as the sign of (C)

$R = 1/2\ (A) + (C)$ if the sign of (A) differs from the sign of (C)

The fractional remainders for the four examples are as follows:

I. $\frac{R}{C} = \frac{1/2\ (-1) + (1/2)}{(1/2)} = 0$

II. $\frac{R}{(C)} = \frac{1/2\ (-11/16) + (11/16)}{(11/16)} = 1/2$

III. $\frac{R}{(C)} = \frac{1/2\ (-6/4) + (3/4)}{(3/4)} = 0$

IV. $\frac{R}{(C)} = \frac{1/2\ (-1)}{(-1/2)} = 1$

This fractional remainder, $\frac{R}{(C)}$, represents the correction which must be made to the least significant bit position in the B register. For example, in equation IV above; (3/4) (-1/2) = 2 (-13/16 + 1/16) = 2 (-12/16) = 1 1/2

I. SQUARE ROOT

During the execution of command 30, the square root of the contents of the AB register pair is extracted (see Figure 2-21). In the first sector of operation, a trial subtraction of $(0.1)^2$ is made into the A register. In the second sector, the $(0.1)^2$ is added back into the A register and, either $(0.11)^2$ is trial subtracted, if the $(0.1)^2$ will go (remainder in A register positive), or $(0.01)^2$ is trial subtracted, if the $(0.1)^2$ will not go (remainder in A register negative). For example, in the fifth sector, $(0.pqr\ 1)^2$ is on whether the "1" in $(0.pqr\ 1)^2$ goes or not. The value of p is a "1" if the first remainder is

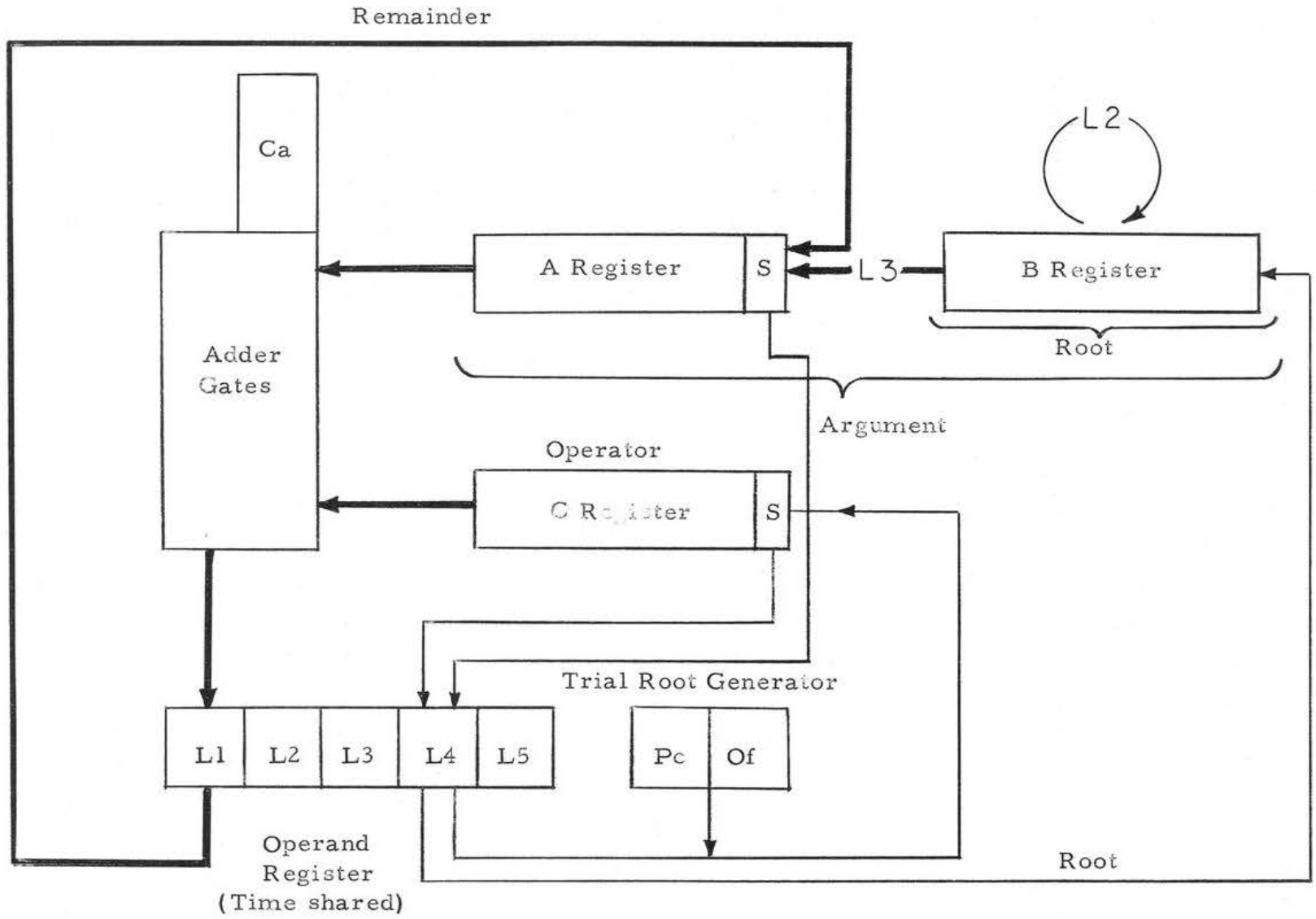


Figure 2-21. Square Root Block Diagram

positive; q is a "1" if the second remainder is positive; etc. Table 2-2 shows the values added into the A register for successive sectors.

Table 2-2.

SQUARE ROOT OPERATION

Sector No.	If Remainder in A Positive	If Remainder in A Negative
1	-0.0 1 -	- - -
2	-0.0 1 0 1 p = 1	p = 0 + 0.0 0 1 1
3	-0.0 0 p 1 0 1 q = 1	q = 0 + 0.0 0 p 0 1 1
4	-0.0 0 0 p q 1 0 1 r = 1	r = 0 + 0.0 0 0 p q 0 1 1
5	-0.0 0 0 0 p q r 1 0 1 s = 1	s = 0 + 0.0 0 0 0 p q r 0 1 1

These values follow from the arithmetic, since

$$(A): (0.pq1)^2 - (0.pq11)^2 = -(0.000pq101),$$

Rule

$$(B): (0.pq1)^2 - (0.pq01)^2 = +(0.000pq011), \text{ etc.}$$

The data handling procedure is similar to that of division; the remainder in the AB register pair is shifted to the left by one position for each sector of operation and the sign of each remainder is used to enter each bit of the answer into the right end of the B register. This allows the bits of the root, p, q, r, etc., to be inserted and retained in bit positions 22, 21, 20, etc., of the C register as they are determined during the basic division process.

The information of the new C register number is synchronized by the parity flip-flop, Pc, and the overflow flip-flop, Of, (see Figures

2-22 and 2-23).

$$sPc = - - - + P24\ Ec\ \overline{O6}\ O5\ O4\ \overline{O3}\ \overline{O2}\ \overline{O1}\ \overline{Of}\ Dg + - - -$$

$$rPc = - - - + Ec\ \overline{O6}\ O5\ O4\ \overline{O3}\ \overline{O2}\ \overline{O1}\ Of\ Pc$$

$$sOf = - - - + \overline{Of}\ Ec\ \overline{O6}\ O5\ O4\ \overline{O3}\ \overline{O2}\ \overline{O1}\ Pc + - - -$$

$$rOf = - - - + \overline{P23}\ \overline{O6}\ O5\ O4\ \overline{O3}\ \overline{O2}\ \overline{O1} + - - -$$

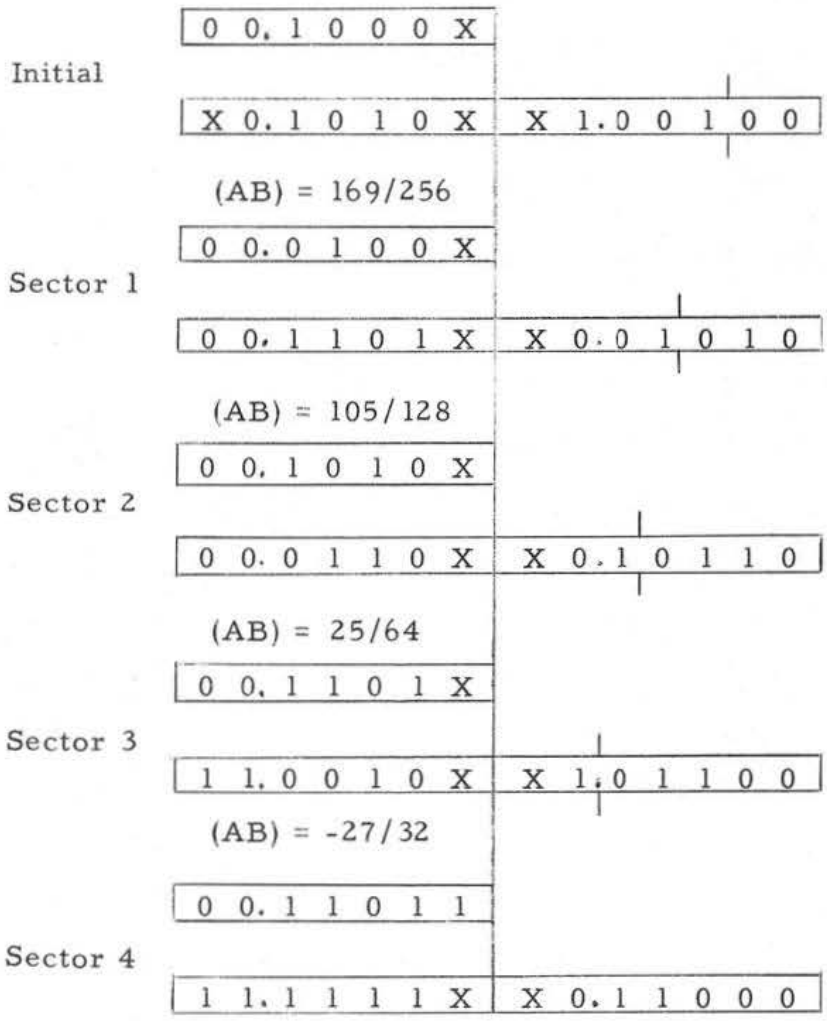
The new value for the C register is formed as

$$Dg = - - - + \overline{O1}\ L5\ (Of\ \overline{Pc}\ Cr + Of\ Pc\ L4 + \overline{Of}\ Pc\ \overline{L4} + \overline{Of}\ \overline{Pc}\ Ce)$$
$$+ \overline{O1}\ \overline{L5}\ F5\ F3\ \overline{F2}\ F1$$

$$sCw = - - - + Ec\ \overline{Rc}\ \overline{O6}\ O5\ O4\ \overline{O3}\ \overline{O2}\ \overline{O1}\ Dg + - - -$$

The L5 flip-flop in its "false" condition ($\overline{L5}$) blocks the normal terms and enters a "1" bit in P21, permitting the starting value of $(0.01)^2$ to be entered in the C register during the first sector of operation. This bit is shifted by the $\overline{Of}\ \overline{Pc}\ Ce$ term in each succeeding sector. After the bit has been shifted, the sign of each previous remainder and its inverse are entered in the C register by the L4 flip-flop. The balance of the C register is retained after the inverted sign. The new value for the C register is added to, or subtracted from, the A register as described on page 2-48. An example is shown in the following chart, using a set of shortened registers.

(C) = 1/2



(A) = -1/16 (B) = 12/16

A minor error is produced in the last sector since the "1" shifted into P1 of the C register is not used to compute the final remainder. The A register should have been zero and the answer in the B register should have been 13/16.

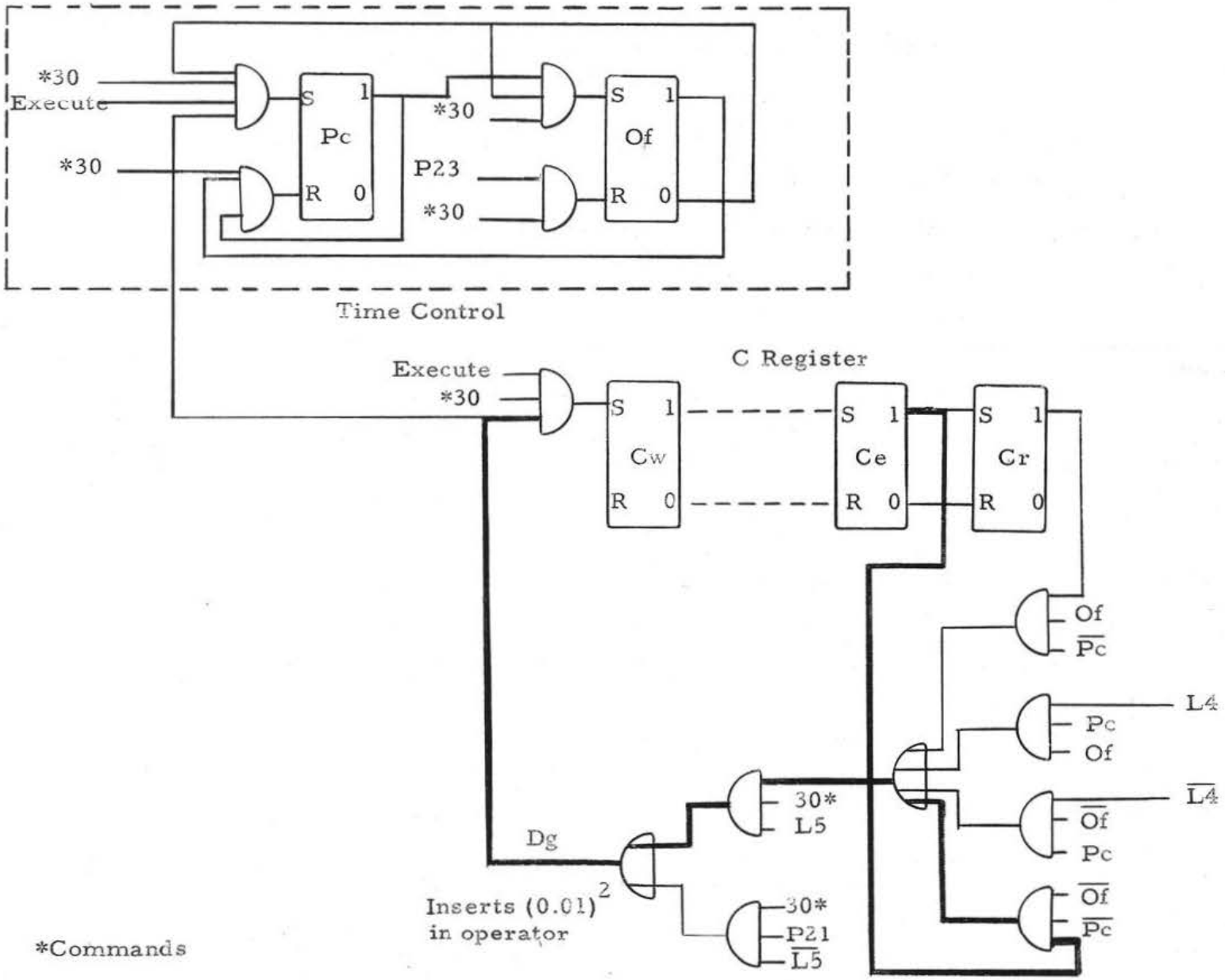


Figure 2-22. Square Root, Logic Diagram

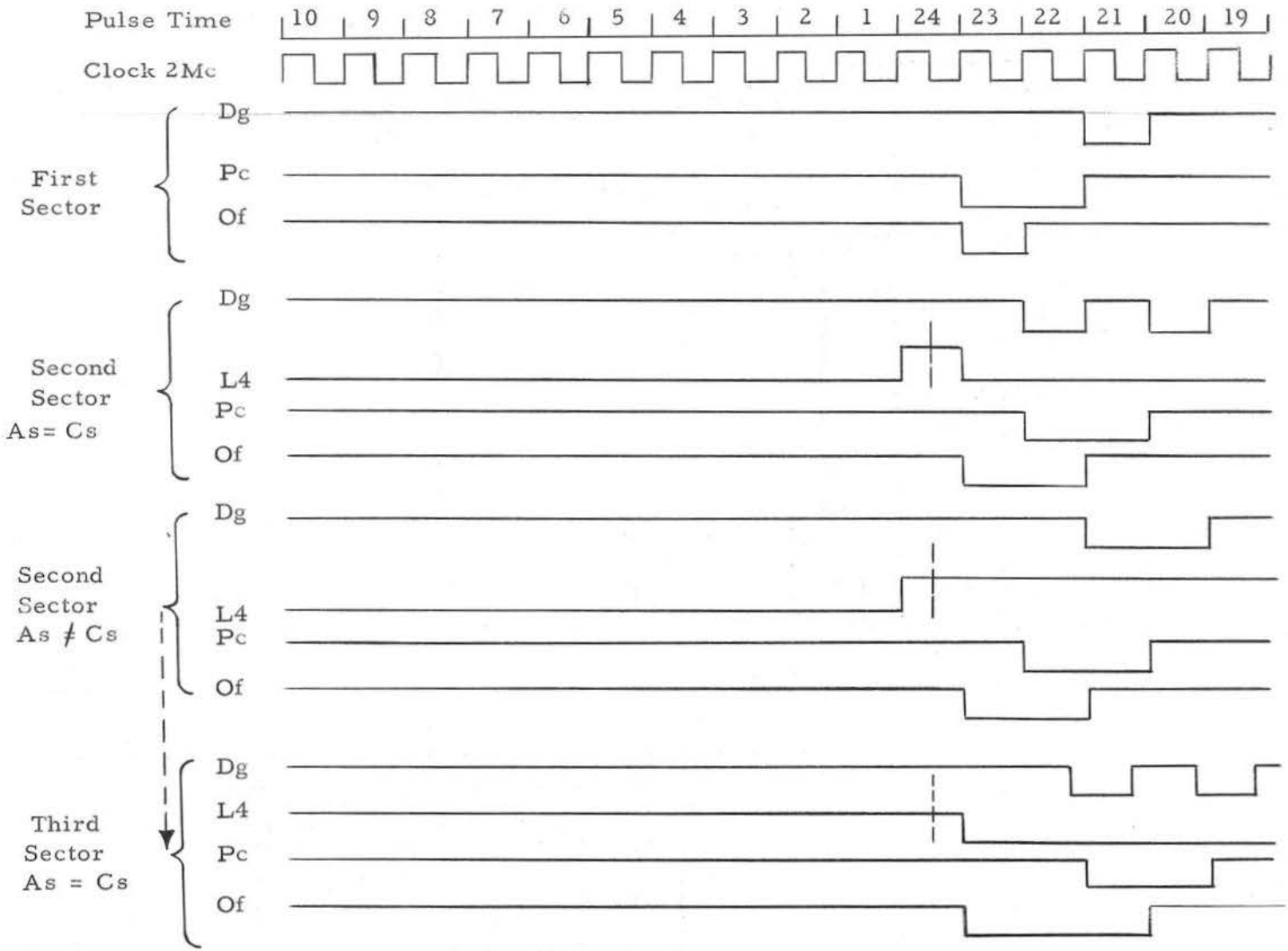


Figure 2-23. Square Root, Timing Diagram

J. SHIFT COMMANDS

Commands 20 and 21 initiate a shift left of the AB register pair (see Figure 2-24).

The A register is shifted left through L1,

$$sL1 = - - - + \overline{O6}\ O5\ \overline{O3}\ \overline{O2}\ Ec\ (Zg\ O4 + Ar\ \overline{O4}) + - - -$$

$$rL1 = - - - + \overline{O6}\ O5\ \overline{O3}\ \overline{O2}\ Ec\ (\overline{Zg}\ O4 + \overline{Ar}\ \overline{O4}) + - - -$$

$$sAw = - - - + Ec\ \overline{Rc}\ \overline{O6}\ O5\ \overline{O3}\ \overline{O2}\ (\overline{P23}\ \overline{P2}\ L1 + - - -) + - - -$$
$$+ \overline{O6}\ O5\ \overline{O4}\ \overline{O3}\ P23\ Ar + - - -$$

the B register is shifted left through L2,

$$sL2 = - - - + \overline{O6}\ O5\ \overline{O3}\ \overline{O2}\ Ec\ Br + - - -$$

$$rL2 = - - - + \overline{O6}\ O5\ \overline{O3}\ \overline{O2}\ Ec\ \overline{Br} + - - -$$

$$sBw = - - - + Ec\ \overline{Rc}\ \overline{O6}\ O5\ \overline{O4}\ \overline{O3}\ \overline{O2}\ \overline{P2}\ L2 + - - -$$

while the shift left from B to A is handled through L3.

$$sL3 = - - - + \overline{O6}\ O5\ \overline{O3}\ (P24\ Bw + - - -) + - - -$$

$$rL3 = - - - + \overline{O6}\ O5\ \overline{O3}\ (P24\ \overline{Bw} + - - -) + - - -$$

$$sAw = - - - + Ec\ \overline{Rc}\ \overline{O6}\ O5\ \overline{O3}\ \overline{O2}\ (- - - + P2\ L3 + - - -) + - - -$$

Commands 22 and 23 initiate a shift right of the AB register pair (see Figure 2-25). The A register is shifted right through Ae,

$$sAw = - - - + Ec\ \overline{Rc}\ \overline{O6}\ O5\ \overline{O4}\ \overline{O3}\ O2\ \overline{P23}\ Ae + \overline{O6}\ O5\ \overline{O4}\ \overline{O3}\ P23\ Ar$$
$$+ - - -$$

the B register is shifted right through Be,

$$sBw = - - - + Ec\ \overline{Rc}\ \overline{O6}\ O5\ \overline{O3}\ O2\ (\overline{P23}\ Be + - - -) + - - -$$

while the shift right from A to B is handled through L3,

$$sL3 = - - - + \overline{O6}\ O5\ \overline{O3}\ (- - - + P2\ Ar) + - - -$$

$$rL3 = - - - + \overline{O6}\ O5\ \overline{O3}\ (- - - + P2\ \overline{Ar}) + - - -$$

$$sBw = - - - + Ec\ \overline{Rc}\ \overline{O6}\ O5\ \overline{O3}\ O2\ (- - - + P23\ L3) + - - -$$

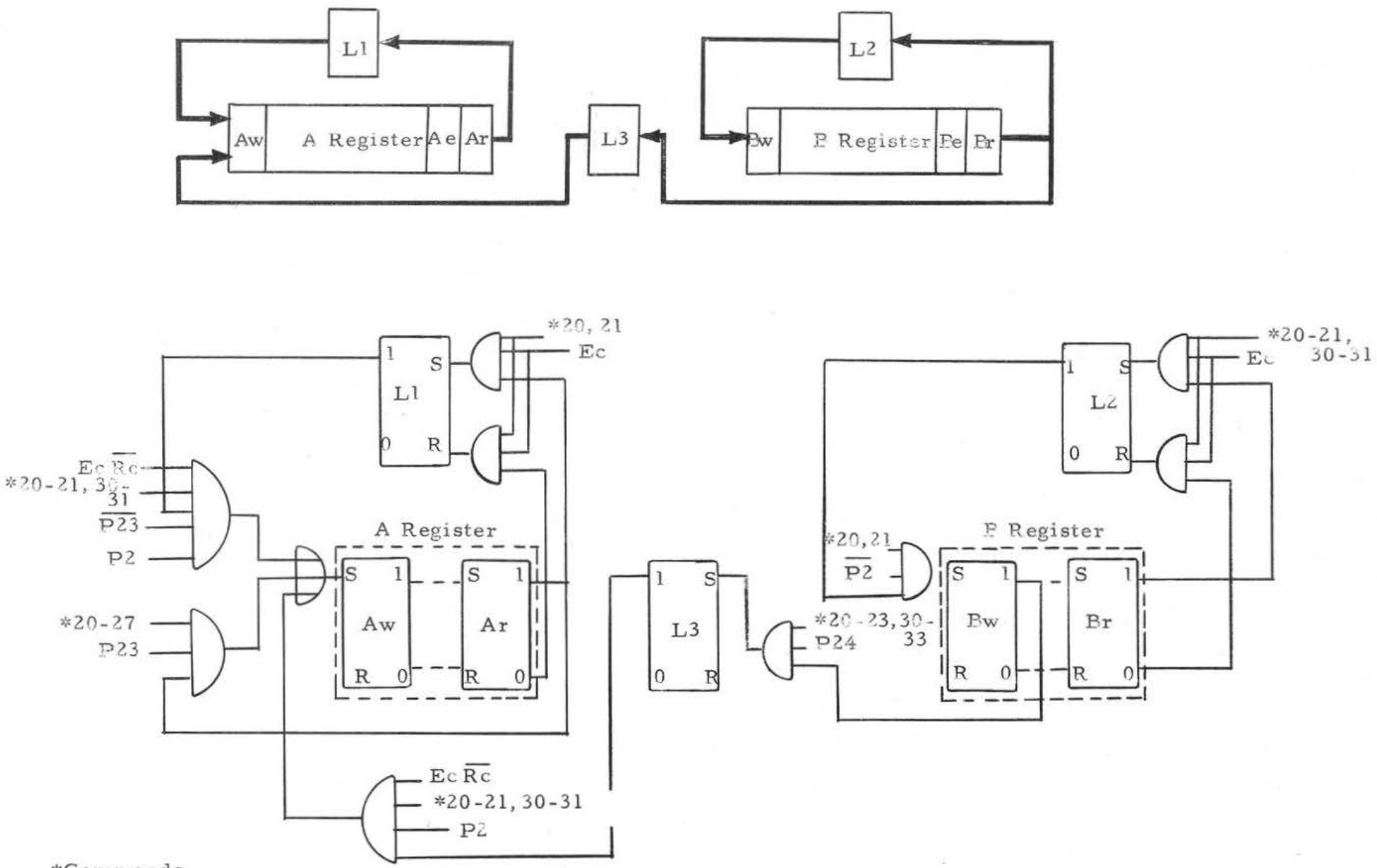
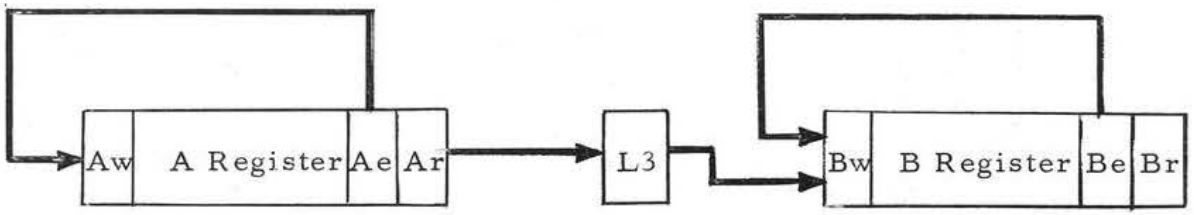
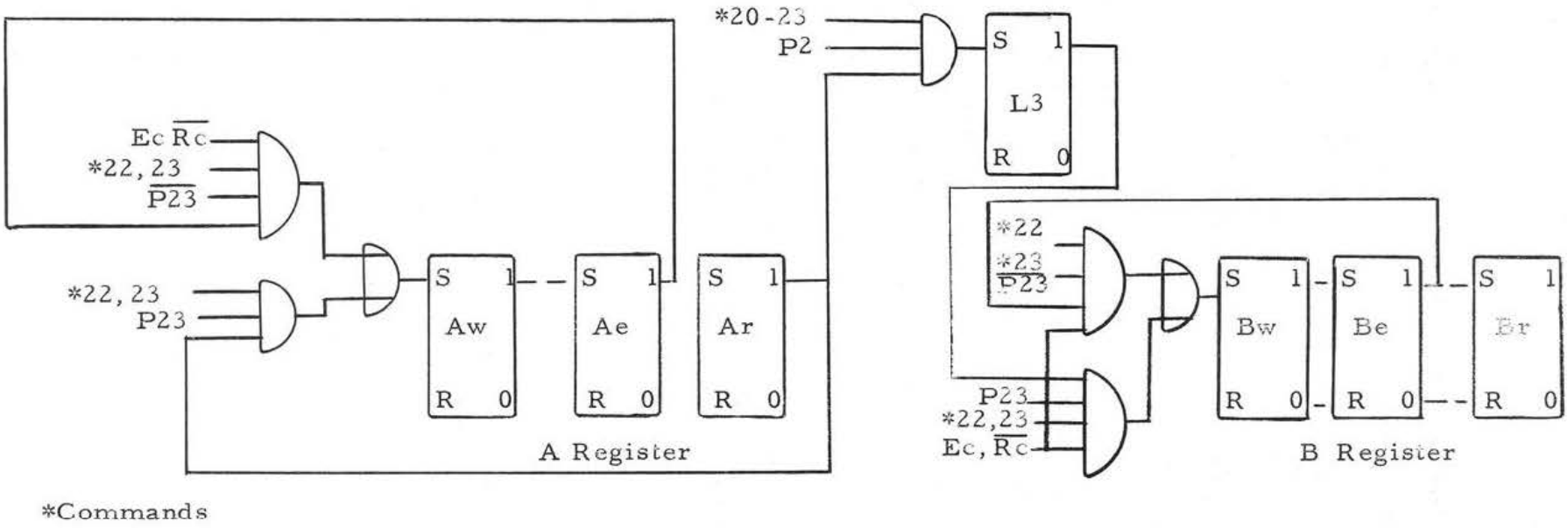


Figure 2-24. AB Shift Left Logic Diagram





*Commands

Figure 2-25. AB Shift Right Logic Diagram

During left shifts the C register is incremented by "-1's," while during right shifts, the C register is incremented by "+1's" (see Figure 2-26).

$$sCw = - - - + Ec\,\overline{Rc}\,O5\,\overline{O4}\,Zg + - - -$$
$$Xg = - - - + O5\,\overline{O4}\,\overline{O3}\,\overline{L4}\,(\overline{O2} + P2)$$
$$Yg = - - - + O5\,\overline{O4}\,Cr$$
$$rCa = - - - + O5\,P24$$

All shift commands are terminated by the sector number of the command, except that shifting stops when the sign and most significant digit of A become different (command 20 is changed to 24), or when C becomes positive (command 23 is changed to 27).

$$sO3 = - - - + P23\,\overline{O6}\,O5\,\overline{O4}\,\overline{O3}\,\overline{O2}\,\overline{O1}\,(Aw\,\overline{Ar} + \overline{Aw}\,Ar)$$
$$+ P24\,\overline{O6}\,O5\,\overline{O4}\,\overline{O3}\,O2\,O1\,\overline{Cw} + - - -$$

K. MISCELLANEOUS COMMANDS

The halt command, 00, is used to set the parity flip-flop, Pc, during phase 3 (wait to execute).

$$sPc = - - - + Ec\,Rc\,\overline{O6}\,\overline{O5}\,\overline{O4}\,\overline{O3}\,\overline{O2}\,\overline{O1}$$

This locks the computer in phase 1 (wait to read next command), until the parity flip-flop is cleared with the BREAK POINT switch.

Command 26 is used to transfer data from the memory line to line 7.

$$sM7w = - - - + \overline{[M0g\,N7g\,Wg]}\;[M7r\,(\overline{O6\,O5\,\overline{O4}\,O3\,O2\,\overline{O1}\,Ec})$$
$$+ (\overline{O6}\,O5\,\overline{O4}\,O3\,O2\,\overline{O1}\,Ec)\;Fg]$$
$$rM7w = - - - + \overline{[M0g\,N7g\,Wg]}\;[\overline{M7r}\,(\overline{O6\,O5\,\overline{O4}\,O3\,O2\,\overline{O1}\,Ec})$$
$$+ (\overline{O6}\,O5\,\overline{O4}\,O3\,O2\,\overline{O1}\,Ec)\,\overline{Fg}]$$

Command 40 extends the bit pattern in the A register where there are

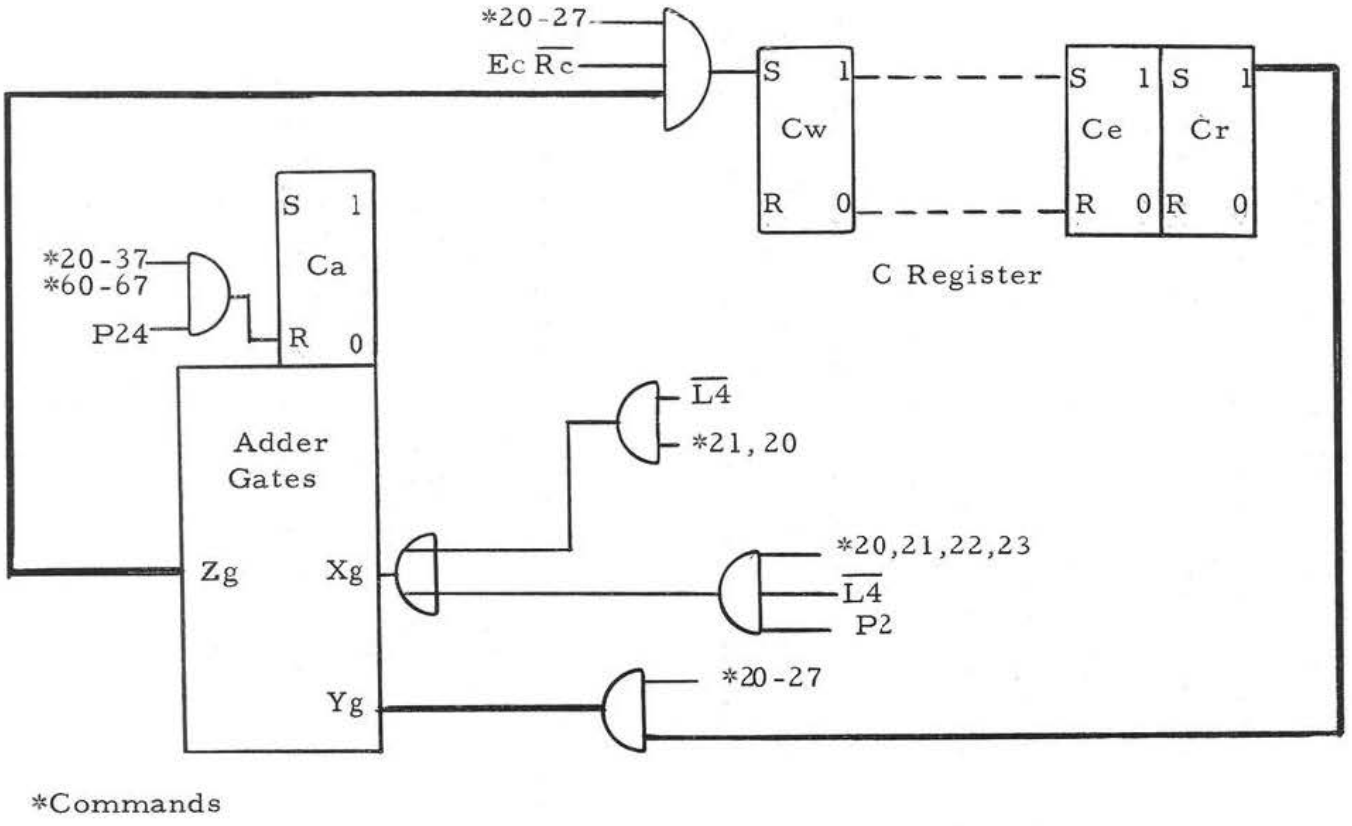


Figure 2-26. Incrementing C Register Logic Diagram

corresponding "1's" in memory.

$$sAw = - - - + Ec\ \overline{Rc}\ O6\ \overline{O5}\ \overline{O4}\ \overline{O3}\ \overline{O2}\ \overline{O1}\ Fg\ Aw + - - -$$
$$+ Ar\left[\overline{Ec\ \overline{Rc}\ (O6\ \overline{O5}\ \overline{O4}\ \overline{O3}\ \overline{O2}\ \overline{O1}\ Fg + - - -)}\right]$$

Command 41 records the signal from the parity flip-flop, Pc, into the A register while Pc is generating the parity signal for the data in the A register.

$$Ig = - - - + \overline{P24}\ \overline{O6}\ \overline{O2}\ O1\ Ar + - - -$$
$$sPc = Ec\ \overline{Rc}\ \overline{P1}\ \overline{O5}\ \overline{O3}\ Ig\ \overline{Pc} + - - -$$
$$rPc = Ec\ \overline{Rc}\ \overline{P1}\ \overline{\overline{O5}\ \overline{O3}}\ Ig\ Pc + - - -$$
$$sAw = - - - + Ec\ \overline{Rc}\ O6\ \overline{O5}\ \overline{O4}\ \overline{O3}\ \overline{O2}\ O1\ Pc + - - -$$

Commands 42, 46, and 47 perform logical operations on the contents of the B register. For both command 42 and 46, the contents of the C register are gated into the B register where there are corresponding "1" bits in memory.

$$sBw = - - - + Ec\ \overline{Rc}\ O6\ \overline{O5}\ \overline{O4}\ O2\ \overline{O1}\ Fg\ Cr + - - -$$

For both commands 46 and 47, each B register bit is cleared where there is a corresponding "1" bit in memory.

$$sBw = - - - + \overline{P1}\ Br\ \overline{\left[Ec\ \overline{Rc}\ (O6\ \overline{O5}\ \overline{O4}\ O3\ O2\ Fg + - - -)\right]}$$

The remaining clearing terms for B are interpreted as follows:

$$sBw = - - - + \overline{P1}\ Br\ \overline{\left[Ec\ \overline{Rc}\ (\overline{O6}\ \overline{O5}\ O3\ O2 + \overline{O6}\ O5\ \overline{O3} + - - - + \overline{O5}\ \overline{O4}\ \overline{O3}\ O2)\right]}$$

Clear (B) for these commands			
	06	20	02
	07	21	03
	16	22	42
	17	23	43
		30	
		31	
		32	
		33	

The clearing terms for the A register are as follows:

$$sAw = - - - + Ar\left[\overline{Ec\,\overline{Rc}\,(\overline{O5}\,\overline{O4}\,\overline{O2}\,O1 + \overline{O5}\,O4\,O3\,\overline{O2} + \overline{O6}\,O5\,\overline{O3} - - - +)}\right]$$

Clear (A) for these commands.

01	14	20
05	15	21
41	54	22
45	55	23
		30
		31
		32
		33

The clearing terms for the C register are as follows:

$$sCw = - - - + \overline{P24}\,Cr\left[\overline{Ec\,\overline{Rc}\,(\overline{O5}\,\overline{O4}\,O3\,\overline{O2}\,\overline{O1} + \overline{O6}\,\overline{O5}\,\overline{O4}\,\overline{O3}}\right.$$

Clear (C) for these commands.

04	00
44	01
	02
	03

$$\left.\overline{+\,O5\,\overline{O4} + \overline{O6}\,O5\,O4\,\overline{O3}\,\overline{O2}\,\overline{O1})}\right]$$

Clear (C) for these commands.

20,60	30
21,61	
22,62	
23,63	
24,64	
25,65	
26,66	
27,67	

The justification for clearing the contents of these registers for the indicated commands can be found by referring to the command list in Table 2-1. In the case of the C register, the indicated clearing for commands 24, 25, 26, and 27 is prevented by recirculation through the adder.

$$sCw = - - - + Ec\,\overline{Rc}\,O5\,\overline{O4}\,Zg + - - -$$

$$Xg = - - - + O6\,O5\,\overline{O4} + O5\,\overline{O4}\,\overline{O3}\,\overline{L4}\,(\overline{O2} + P2)$$

$$Yg = - - - + O5\,\overline{O4}\,Cr$$

Command 56 is used to set the overflow flip-flop, Of, and then reset it if there is any difference between the A register and memory.

$$sOf = - - - + P24\ Rc\ O6\ \overline{O5}\ O4\ O3\ O2\ \overline{O1}$$

$$rOf = - - - - + \overline{P1}\ \overline{P24}\ Ec\ \overline{Rc}\ O6\ \overline{O5}\ O4\ O3\ O2\ \overline{O1}\ (Fg\ \overline{Ar} + \overline{Fg}\ Ar)$$

Command 70 generates a signal to pulse a unit external to the computer. The control pulse execution signal is as follows:

$$Cpg = Ec\ O6\ O5\ O4\ \overline{O3}\ \overline{O2}\ \overline{O1}$$

The transmitted output code is taken from L1, L2, L3, L4, and L5.

Command 72 generates a serial output from the memory line. The format mask is taken from the same command line from which command 72 is read.

$$Gsg = Ec\ O6\ O5\ O4\ \overline{O3}\ O2\ \overline{O1}\ Vg\ \overline{P1}\ \overline{P24}$$

$$Gdg = Fg$$

Command 73 provides a serial input to the memory line. The format mask is taken from the same command line from which command 73 is read.

$$Hsg = Ec\ O6\ O5\ O4\ \overline{O3}\ O2\ O1\ Vg\ \overline{P1}\ \overline{P24}$$

$$Ig = - - - + \overline{P24}\ O6\ O5\ O2\ Vg\ Hdg + - - -$$

Command 77 is used for testing a number of external signals. The memory line number of the command is used to select 1 of 32 inputs.

$$\begin{aligned} Jg = {} & M0g\ N0g\ (J0) + - - - + M3g\ N0g\ (J24) \\ & + M0g\ N1g\ (J1) + - - - + M3g\ N1g\ (J25) \\ & + - \\ & + M0g\ N4g\ (J4) + - - - + M3g\ N4g\ (J28) \\ & + M0g\ N5g\ (J5) + - - - + M3g\ N5g\ (Bp) \\ & + M0g\ N6g\ (J6) + - - - + M3g\ N6g\ \overline{Rf}\ \overline{Tf} \\ & + M0g\ N7g\ (J7) + - - - + M3g\ N7g\ (Tb) \end{aligned}$$

The (J27) signal is connected to the magnetic tape clock signal, (Uc) and the (J28) signal is jumpered to the photo tape reader sprocket signal, (Sc).

L. INPUT AND OUTPUT

The input operation on the PB 250 is shown in Figure 2-27. Input and output is handled on an individual character basis. Each 8-bit input character is entered into the buffer section of the sector counter. Each 8-bit output character is transmitted by a character output command and is presented by the operand line selector register flip-flops, L5, L4, L3, L2, and L1, and the operation code register flip-flops, O3, O2, and O1.

For character input control, the Rf and Tf flip-flops are controlled by commands 50, 51, 52, and 53.

$$sRf = Ec\ O6\ \overline{O5}\ O4\ \overline{O3}\ O2 + - - -$$
$$rRf = Ec\ O6\ \overline{O5}\ O4\ \overline{O3}\ \overline{O2} + - - -$$

$$sTf = Ec\ O6\ \overline{O5}\ O4\ \overline{O3}\ O1$$
$$rTf = Ec\ O6\ \overline{O5}\ O4\ \overline{O3}\ \overline{O1} + - - -$$

The function of these flip-flops is to control the input devices. The condition $Rf\ \overline{Tf}$ is used to energize the reader release magnet, LR, and to energize the reader contacts return. The condition $\overline{Rf}\ Tf$ is used to energize the "type light" and the typewriter contacts return. A gate line for the fast readers is energized by the condition Rf Tf. The signals from the reader common, (Rc) and the signals from the typewriter common, (Tc), are used to reset Rf and Tf, respectively.

$$rRf = - - - + (Fi)\ (Mr)\ (P8 - P15) + (Rc)\ \overline{Ec}$$
$$rTf = - - - + (Tc)\ Ec + (Fi)\ (P8 - P15)$$

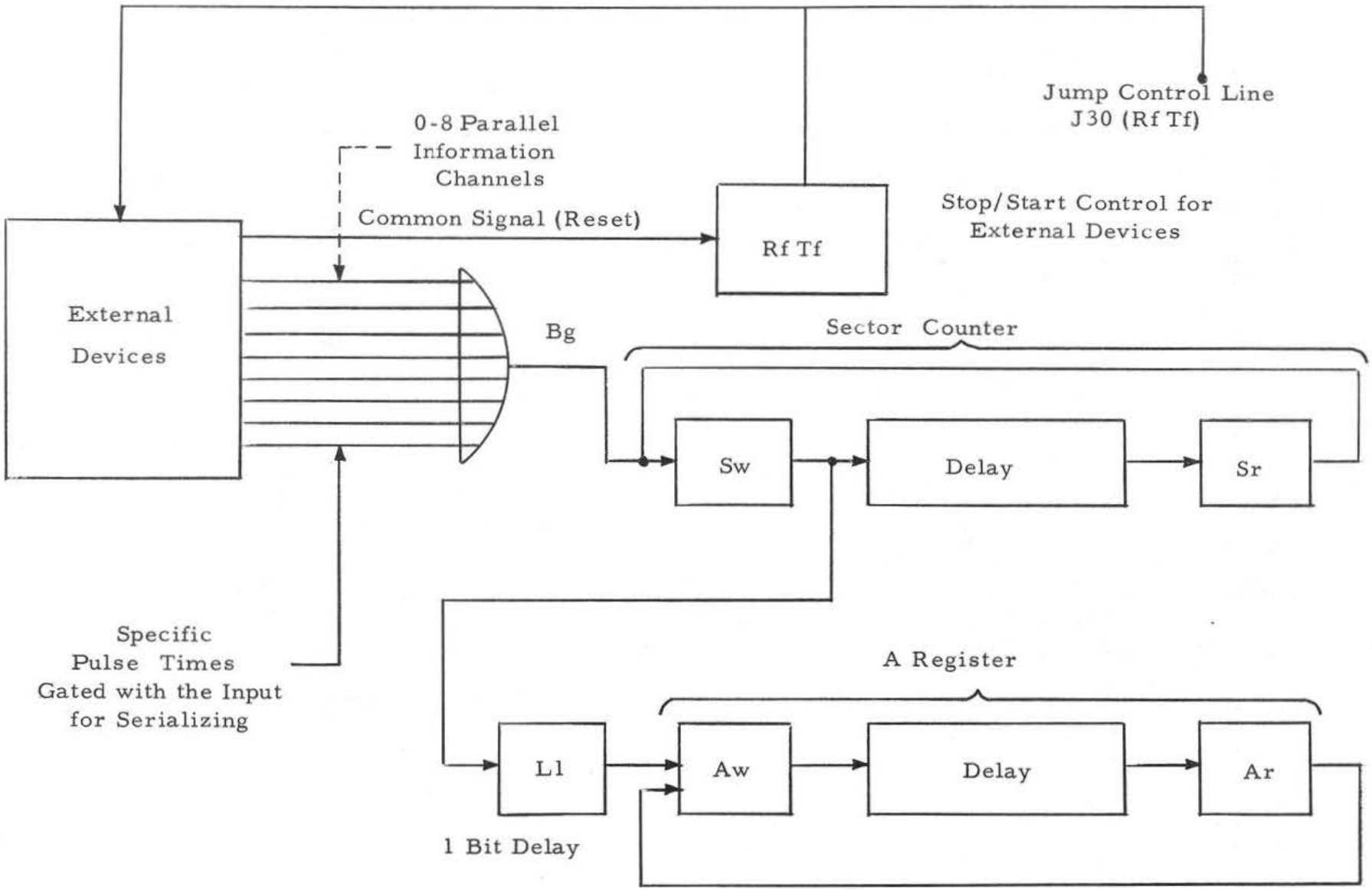


Figure 2-27. Buffer Input

The input code signals are

Ⓡ1, Ⓡ2, Ⓡ3, Ⓡ4, Ⓡ5, Ⓡ6, Ⓡ7, and Ⓡ8 from the mechanical tape reader.

Ⓣ1, Ⓣ2, Ⓣ3, Ⓣ4, Ⓣ5, and Ⓣ6 from the typewriter.

Ⓢ1, Ⓢ2, Ⓢ3, Ⓢ4, Ⓢ5, Ⓢ6, Ⓢ7, and Ⓢ8 from the photo tape reader.

Ⓤ1, Ⓤ2, Ⓤ3, Ⓤ4, Ⓤ5, Ⓤ6, Ⓤ7, and Ⓤ8 from the magnetic tape reader.

These parallel codes are serialized in pulse positions P24, P1, P2 - - - P6, and P7, and entered into the buffer section of the sector counter.

$$
\begin{aligned}
Bg = {} & P24\,(Ⓡ1 + Ⓢ1 + Ⓣ1 + Ⓤ1) \\
& + P1\,(Ⓡ2 + Ⓢ2 + Ⓣ2 + Ⓤ2) \\
& + P2\,(Ⓡ3 + Ⓢ3 + Ⓣ3 + Ⓤ3) \\
& + P3\,(Ⓡ4 + Ⓢ4 + Ⓣ4 + Ⓤ4) \\
& + (P24\text{-}P7)\,F3\,\overline{F2}\,\overline{F1}\,(Ⓡ5 + Ⓢ5 + Ⓣ5 + Ⓤ5) \\
& + (P24\text{-}P7)\,F3\,\overline{F2}\,F1\,(Ⓡ6 + Ⓢ6 + Ⓣ6 + Ⓤ6) \\
& + (P24\text{-}P7)\,F3\,F2\,\overline{F1}\,(Ⓡ7 + Ⓢ7 + Ⓤ7) \\
& + (P24\text{-}P7)\,F3\,F2\,F1\,(Ⓡ8 + Ⓢ8 + Ⓤ8)
\end{aligned}
$$

$$sSw = - - - + Bg\,\overline{Qg}$$

$$rSw = - - - + \overline{Sc}\,\overline{Sr}\,\overline{Bg} + Qg\,\overline{Bg}$$

Each eight-bit code from the mechanical tape reader and from the typewriter is entered into the buffer before the corresponding common signal sets the condition $\overline{Rf}\,\overline{Tf}$. This entry is made for the "1's" of the code to prevent dropouts due to contact chatter. When the state $\overline{Rf}\,\overline{Tf}$ is set, the reader and typewriter contact returns are deenergized to terminate the input signals to the buffer. The code is then cleared from the buffer and loaded into the A register with command 55.

$$Qg = \overline{Ec}\ O6\ \overline{O5}\ O4\ O3\ O1\ \overline{P24}\ (P24\text{-}P7) + - - -$$
$$+ Ec\ \overline{Rc}\ O6\ \overline{O5}\ O4\ O3\ O1\ P24$$

$$sSw = - - - + \overline{Sc}\ Sr\ \overline{Qg} + - - -$$
$$rSw = - - - + Qg\ \overline{Bg}$$

$$sL1 = - - - + O6\ \overline{O5}\ O4\ \overline{O2}\ Ec\ Sw$$
$$rL1 = - - - + O6\ \overline{O5}\ O4\ \overline{O2}\ Ec\ \overline{Sw}$$

$$sAw = - - - + Ec\ \overline{Rc}\ O6\ \overline{O5}\ O4\ O3\ \overline{O2}\ O1\ (L1\ Vg + Ar\ \overline{Vg}) + - - -$$

2-9. BUFFER CLEARING

The bits from the buffer are delayed through Sw and L1 flip-flops to place the first code bit in P2 of the A register. The number of code bits entered into the A register is controlled by a format word in the command line and by the L1 Vg term. The Ar $\overline{Vg}$ term preserves the balance of the A register. For the buffer clearing term, Qg, the P24 position is cleared as the command is executed, while positions P1 through P7 are cleared in the first sector after execution. This first sector after execution may be phase 1 or phase 2; therefore O2 which changes at P2 of phase 2, must be omitted from the clearing term. This requirement results in command 57 being a buffer clearing command also.

If commands 51 or 52 are repeated while the input contacts are still closed, the same input code will be entered into the buffer and the same common contact signal will again reset the condition $\overline{Rf}\ \overline{Tf}$. In the case of fast tape readers, the computer program must phase the use of commands 55 and 57, based on sensing an input clock signal with command 77.

The output operation of the PB 250 is shown in Figure 2-28. An 8-bit character output code is presented by O3, O2, O1, L5, L4, L3, L2, and L1 when commands 60 through 67 are executed. The destination is coded by

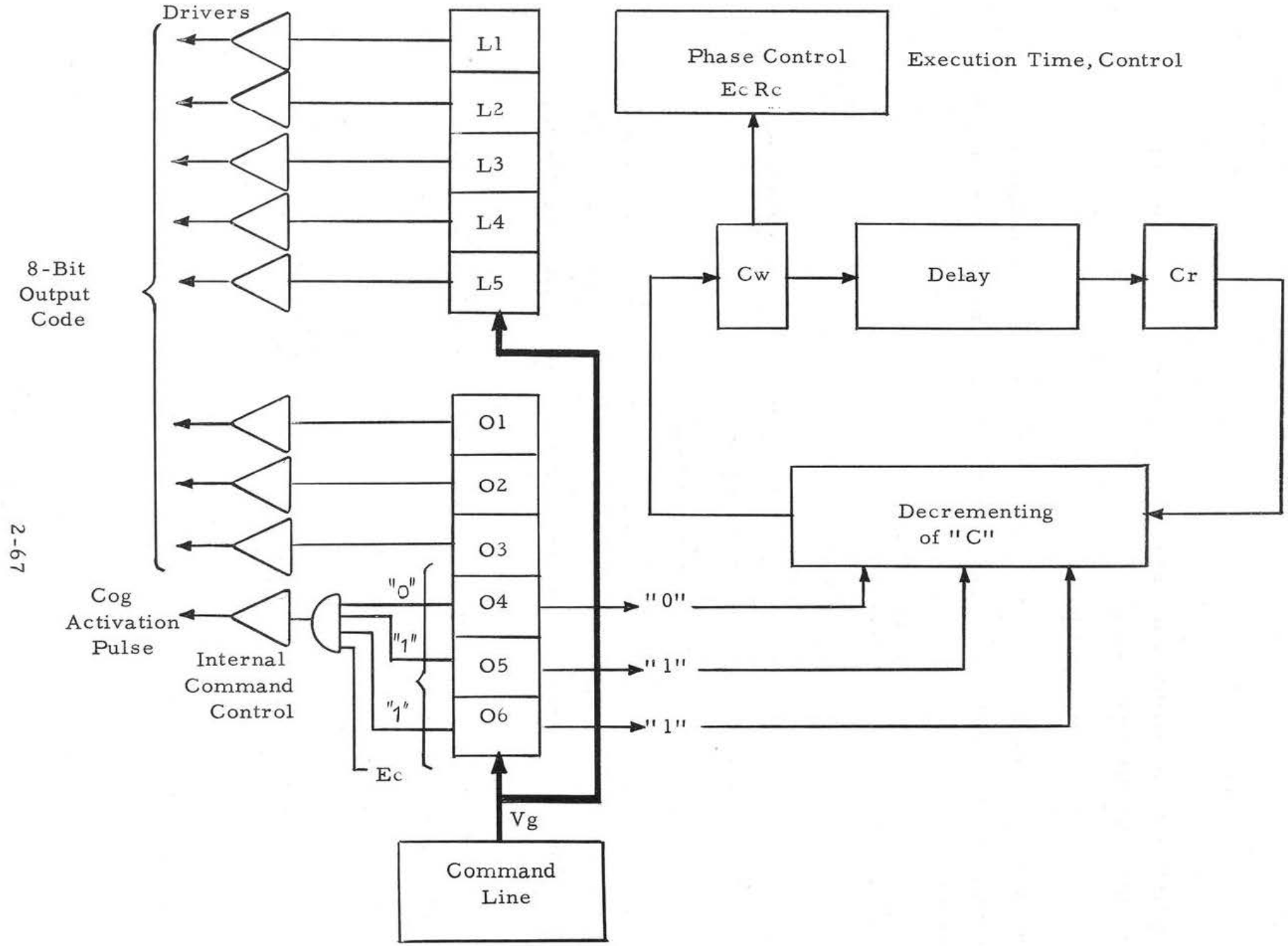


Figure 2-28. Character Output Operation

the command line selector register flip-flops, K3, K2, K1. The output character execute signal is Cog.

$$Cog = Ec\ O6\ O5\ \overline{O4}$$

The duration of execution of a character output command is controlled by decreasing a control number in the C register by one count for each sector of execution.

$$Xg = - - - + O6\ O5\ \overline{O4} + - - -$$

$$Yg = - - - + O5\ \overline{O4}\ Cr$$

$$sCw = - - - + Ec\ \overline{Rc}\ O5\ \overline{O4}\ Zg + - - -$$

While the register remains positive, the Is flip-flop is prevented from indicating a sector comparison to terminate phase 4.

$$rIs = - - - + P23\ Ec\ Rc\ O6\ O5\ \overline{O4}\ \overline{Cr}$$

If the C register starts with a large positive number, the execution phase may extend to 25 seconds. If the C register is zero or negative, the execution duration will be controlled entirely by the memory sector number of the command.

The "type output character" and "punch output character" execute signals are provided as follows:

$$Tg = Ec\ O6\ O5\ \overline{O4}\ K3\ \overline{K2}\ K1$$

$$Pg = Ec\ O6\ O5\ \overline{O4}\ K3\ K2\ \overline{K1}$$

The code selector signals are also provided for the typewriter and punch.

LT1 = L1 Tg	LP1 = L1 Pg
LT2 = L2 Tg	LP2 = L2 Pg
LT3 = L3 Tg	LP3 = L3 Pg
LT4 = L4 Tg	LP4 = L4 Pg
LT5 = L5Tg	LP5 = L5 Pg
LT6 = O1 Tg	LP6 = O1 Pg
	LP7 = O2 Pg
	LP8 = O3 Pg

M. BOOTSTRAP INPUT

Figure 2-29 illustrates the bootstrap input. Bootstrap is an input means for loading line 01 from a special tape without the use of a program in the computer. The special tape format is indicated as follows:

(B6)	(B5)	(B4)	(B3)	(B2)	(B1)	
1	0	1	1	1	0	Carriage Return
1	0	0	0	0	0	Zero for 0's
1	0	1	0	0	0	H for 1's
0	1	0	0	0	0	Space Code
0	0	1	0	1	1	Stop Code
0	0	0	0	0	0	Feed Code

C	C	"1"			S
O	O				T
M	M				O
M	M				P
O	O				
N	N				

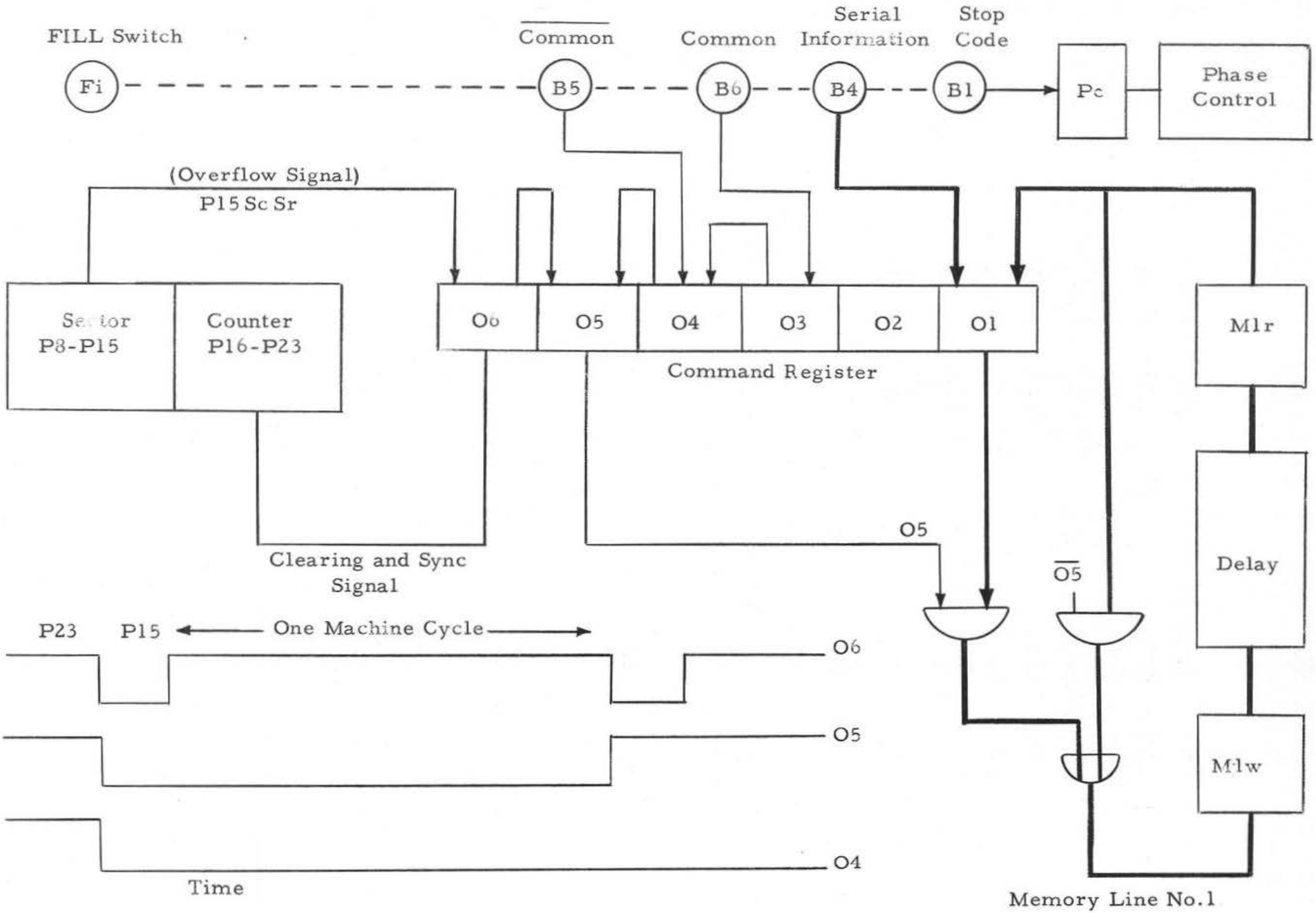


Figure 2-29. Bootstrap Input

Channel 4 provides the binary input data, while channel 1 provides the stop code. In the case of a photo tape reader or a magnetic tape reader, channel 6 provides an input common signal and channel 5 provides an input $\overline{\text{common}}$ signal. These signals are provided by (Rc) and ($\overline{\text{Rc}}$) in the case of a mechanical reader.

The input is started by closing the FILL switch and resetting the parity flip-flop if it is set. If a mechanical reader is used, (Mr) is "true" and Rf $\overline{\text{Tf}}$ is set to release the reader. Otherwise, Rf $\overline{\text{Tf}}$ is set.

$$\text{sRf} = \text{---+}\ (\text{Fi})\ \overline{\text{Pc}}\ (\text{P8-P15})\ (\text{Mr})$$
$$\text{rRf} = \text{---+}\ (\text{Fi})\ (\overline{\text{Mr}})\ (\text{P8-P15}) + (\text{Rc})\ \overline{\text{Ec}}$$
$$\text{rTf} = \text{---+}\ (\text{Fi})\ (\text{P8-P15})$$

To start a photo tape reader or a magnetic tape reader, start and stop signals are generated from the parity flip-flop.

$$\text{Bootstrap Start} = (\text{Fi})\ \overline{\text{Pc}}$$
$$\text{Bootstrap Stop} = (\text{Fi})\ \text{Pc}$$

The input process is stopped by setting the parity flip-flop as soon as the first "1" bit in channel 1 is detected.

$$\text{sPc} = \text{---+}\ (\text{Fi})\ (\text{B1})\ \text{+---}$$

While the FILL switch is depressed, computation is blocked by the ($\overline{\text{Fi}}$) term in sRc. When the FILL switch is released, computation will still be blocked by the $\overline{\text{Ec}}$ Rc Pc P1 term of rRc. If it is necessary to put a new tape in the reader, the "I" key of the typewriter must be depressed while the ENABLE switch is depressed, to allow computation to start at memory word zero. At this time the BREAK POINT switch can be depressed to reset the Pc flip-flop.

While the bootstrap input is in operation, each "common" signal is

detected by the O3 flip-flop. The "common" signal that follows is then detected by the O4 flip-flop.

sO3 = - - - + (Fi) (B6)

sO4 = - - - + (Fi) (B5) $\overline{O4}$ O3

After O4 is set, O5 is set for one machine cycle, synchronized by O6; then O3, O4, and O5 are reset simultaneously.

sO5 = - - - + (Fi) P2 3 O6 $\overline{O5}$ O4

rO5 = - - - + (Fi) P2 3 O6 O5

rO4 = - - - + (Fi) P2 3 O6 O5 + - - -

rO3 = - - - + (Fi) P2 3 O6 O5

sO6 = - - - + (Fi) F4 F3 F2 F1 Sr Sc

rO6 = (Fi) P2 3 + - - -

While the "common" signals are detected by O3, the "1" bits in channel 4 are entered into the O1 flip-flop.

sO1 = - - - + (Fi) (B4) $\overline{O5}$ + - - -

rO1 = - - - + (Fi) $\overline{O3}$ + - - -

Before each input character, O1 is cleared by the (Fi) $\overline{O3}$ term.

With the FILL switch depressed, O6 is set once per machine cycle when all "1's" are read in Sr during the interval (P8-P15). This marks each machine cycle and serves to synchronize the second number in the sector counter.

Qg = - - - + (Fi) O6 + - - -

While O5 is set, the contents of line 01 are shifted by one bit through the O1 flip-flop. This enters each bit from the O1 flip-flop into line 01.

$$sO1 = - - - - + (Fi)\ O5\ M1r$$

$$rO1 = - - - - + (Fi)\ O5\ \overline{M1r}$$

$$sM1w = - - - - + [M0g\ N1g\ Wg]\left[M1r\left(\overline{(Fi)\ O5}\right) + (Fi)\ O5\ O1\right]$$

$$rM1w = - - - - + [M0g\ N1g\ Wg]\left[\overline{M1r}\left(\overline{(Fi)\ O5}\right) + (Fi)\ O5\ \overline{O1}\right]$$

The following chart indicates signal sources for various readers.

	(B6)	(B5)	(B4)	(B1)	(Mr)
Mechanical Reader	(Rc)	(Rc)	(R4)	(R1)	1
Photo Reader	(S6)	(S5)	(S4)	(S1)	0
Mag. Tape Reader	(U6)	(U5)	(U4)	(U1)	0

The Bootstrap input connector selects the source of (B6), (B5), (B4), (B1), and (Mr).

The input bits are entered at P24 of word $(376)_8$ of line 01 and precess into words $(377)_8$, 00, 01, 02, 03, etc. The input rate is limited by the rate of precessing bits into line 01. A tape for use with the mechanical reader should be prepared as follows:

```
0 0 0 0 H 0 0 H 0 0 0 0 0 0 0 0 H H 0 0 0 0 0 0 CR  <-------- for 01-001
H 0 0 0 0 0 0 0 0 H 0 0 0 0 H H 0 0 0 0 0 0 0 0 CR  <-------- for 01-000
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 CR  <-------- for 01-255
0 ST                                                <-------- for 01-254
```

A tape for use with the photo reader should be prepared as in the preceding example, but with two-space codes before each code shown. A magnetic tape for use as a bootstrap should have 14-space codes between each code shown. These space codes are to allow two machine cycles between input characters.

N. MAGNETIC TAPE

With six bits per character and three or four characters per word, a 2 kc input rate can result in an input rate as high as 667 words per second, or two words per machine cycle. The program presented here allows for this rate, with blocks of up to 272 words or 816 characters. The end-of-block will be detected by the tape handler, using either code or gap detection.

The program assembles words in the A register with LSD commands, assembles 16-word groups in L0 with IAM commands, and precesses the 16-word groups into L4 through a 16-word buffer register, L15, with DUMP operations. The first group is precessed into memory locations 04-255 through 04-014, then into memory locations 04-015 through 04-030, etc.

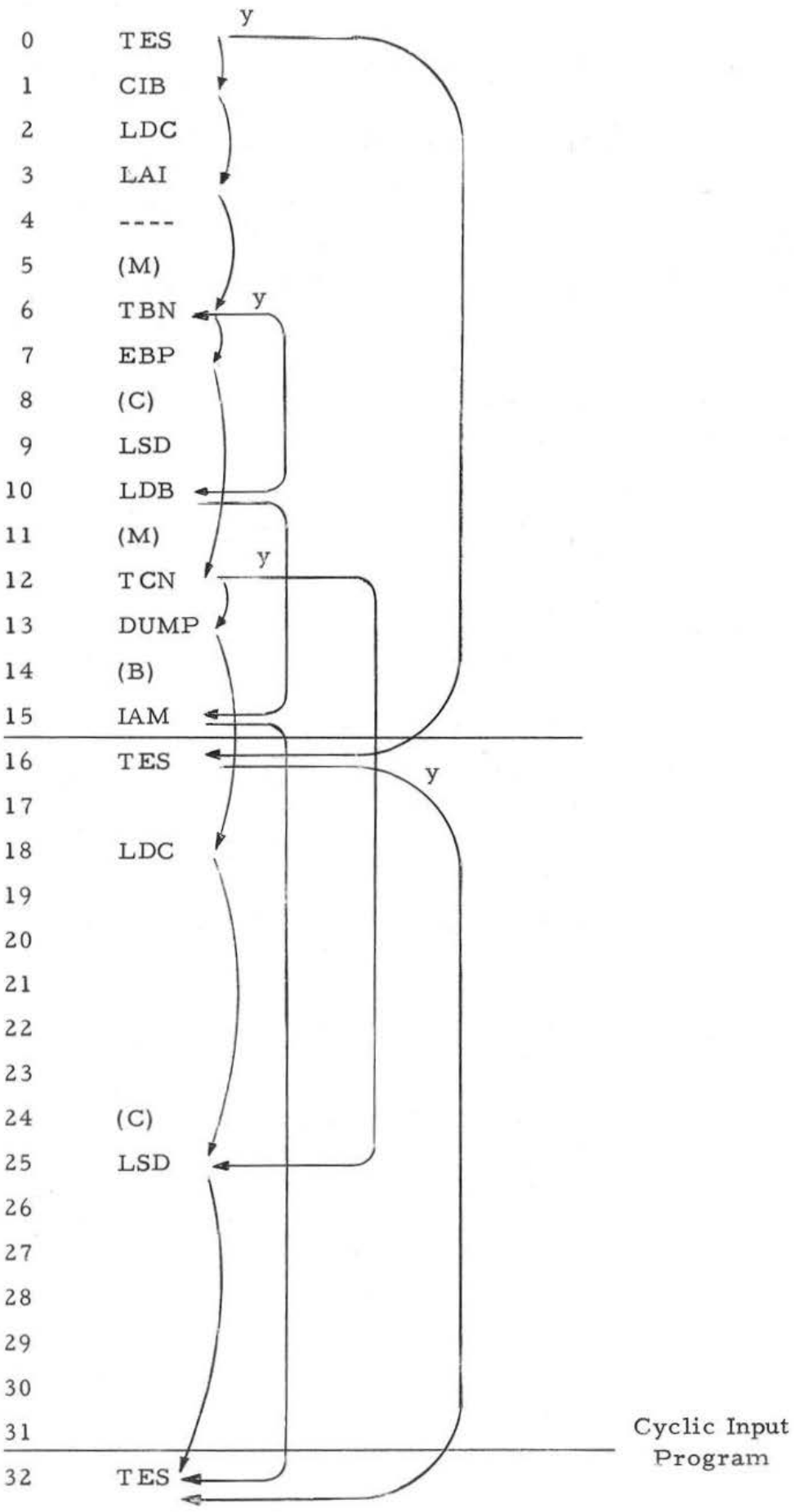
The input program is a cyclic set of 16 commands in one line. The tape reader is started by a PTU command and control is immediately switched to the input program. While waiting for input characters, the program tests the input clock every 192 μsec. When a character is detected, it is transferred to the A register and shifted left. If the character completes a word in the A register, the shift is replaced by an IAM transfer to L0. When a clock signal is detected, the input clock is tested after 384 μsec rather than 192 μsec. This limits the signal width between 232 μsec and 384 μsec, and the peak input rate is limited to 2600 characters per second.

The detailed commands are listed in Table 2-3.

Table 2-3.

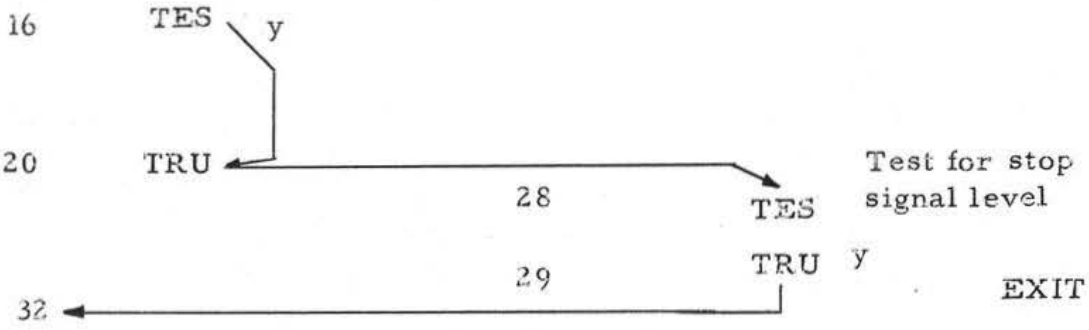
MAGNETIC TAPE INPUT COMMAND LIST

Word	Command	Function
0	TES	Tests input clock. "No" equals an input signal.
1	CIB	Clears input buffer immediately before each character is accepted.
3	LAI	Loads input character.
5	(M)	Input mask for LAI.
6	TBN	Tests for a "one-bit" shifted into sign of B register to indicate completed word in A register.
7	EBP	Stretches bit into sign position of A register.
11	(M)	Mask for EBP.
10	LDB	Restores B register after each word is formed in A register.
14	(B)	Marker bit for B register.
15	IAM	Precesses A register into L0 for 16 sectors.
12	TCN	Tests for completion of 16 new words in L0. C register starts out negative and counts down to positive with LSD commands.
13	DUMP	Starts transfer of 16 words in L0 to long line 04.
2	LDC	Restores count to C register after each 16 words in L0.
8	(C)	Number for LDC.
9	LSD	Shifts A register left. The LSD command may be placed in position 10 and LDB in position 9 for 5-bit input.



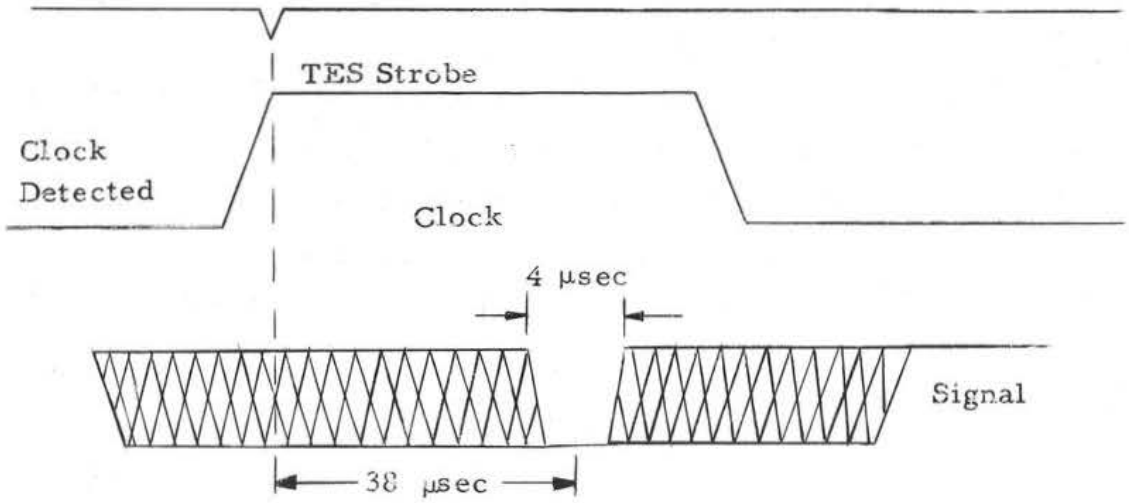
Cyclic Input Program

The program exit is not indicated. The tape handler detects its own end-of-block and signals the computer with a dc level. One 16-word program block can branch out to another command line to test for the stop signal. For example:

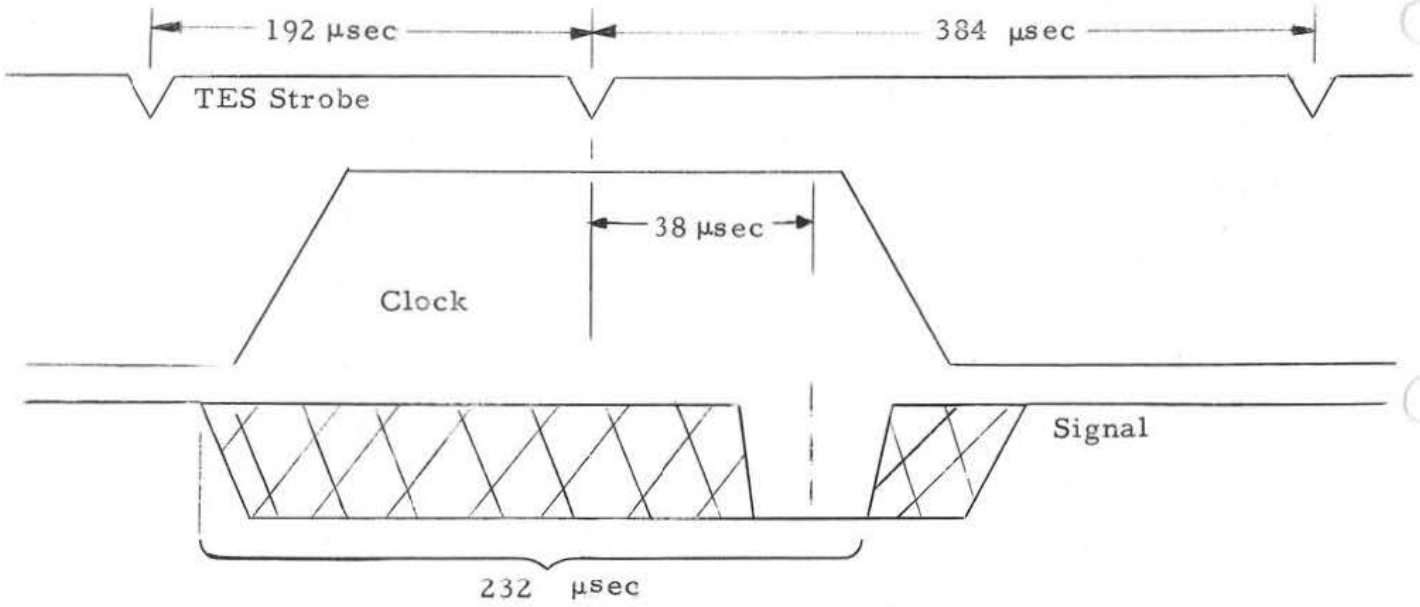


2-10. SIGNAL TOLERANCES

The input signals are accepted following the detection of a clock signal. This allows a tape skew of up to 36 μsec, i.e., a signal need not occur until 36 μsec following the clock.



The minimum duration of signals is also influenced by this 38 μsec, i.e., each signal must be present from 36 μsec following the clock to 232 μsec following the clock.



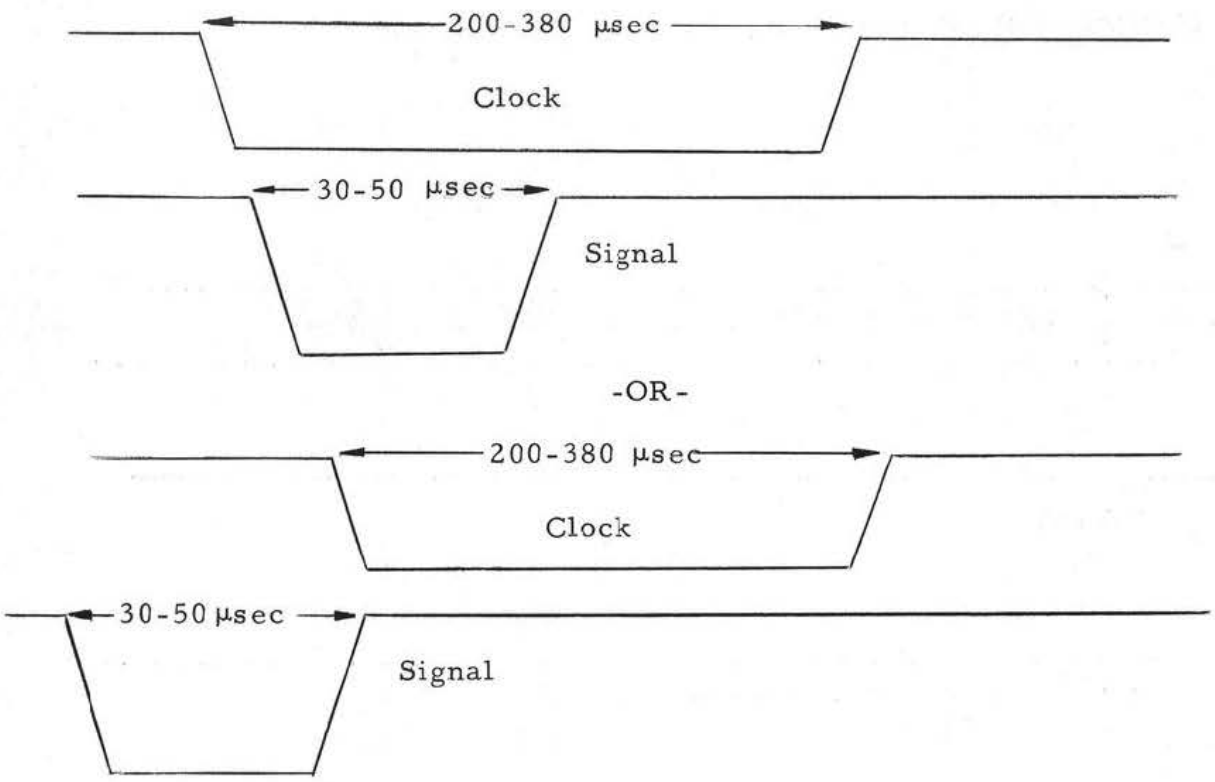
The DUMP command is a PTU command with an assigned line number. It sets in motion a sequence of data handling operations. The phases of operation are controlled by flip-flops.

Phase 2	Phase 1	
0	0	Idle, wait for DUMP command.
0	1	During IAM to L0, send shifted data from L0 ⟶ L15 and send data from L15 ⟶ L0.
1	1	Idle and wait for start of machine cycle.
1	0	Precess L4 through L15 for one machine cycle.

The DUMP command is useful for taking data from L4 for recording on magnetic tape as well as storing data in L4 when reading from magnetic tape.

2-11. ALTERNATE PROGRAMS

If the input signals are different, the input program may require some alteration. Two other possible input signals are shown below.



For the two types of signals shown above, the following program can be used. The LAI command clears the buffer.

```
0       TES
1       TRU
2       LDC
3       LAI
4       ----
5       (M)
6       TBN
   etc.
```

The timing of the preceding program is as follows: