G-20 Central Processor Service Manual

VOLUME 1

TABLE OF CONTENTS

CENTRAL PROCESSOR SERVICE MANUAL

VOLUME I

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS

LIST OF TABLES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The intention of this manual is to provide the information necessary to an understanding of the electronic and logical design of the G–20 Central Processor. In these discussions, acquaintance with the following has been assumed:

1) the basic principles of electronics,
2) Boolean algebra,
3) the use of binary–coded octal notation.

Multitudinous published texts are devoted to the exposition of these ideas. The unique aspects of the design of a central processor as complex as the G–20 provide sufficient scope for a single manual; the inclusion of more general information would obscure the issue.

Similarly, familiarity with the G–20 computing system and the underlying system concepts has been assumed. The General Reference Manual is devoted to a thorough analysis of the system and its components; inclusion of this material in the current manual would involve unnecessary duplication. Actions taken by the Central Processor logic that involve communication with other units in the system can be understood only in terms of the system as a whole. In fact, a certain amount of confusion is bound to result from analyzing the detailed workings of the Central Processor without knowledge of the system requirements that dictated the implementation. This is particularly true of the communication and interrupt systems since these are the

most unique elements of the system design. It should be emphasized that a discussion of these systems has a proper place in this manual and that the material has been omitted only because it is so well presented in the Reference Manual. Discussions in this manual presuppose acquaintance with this information.

The manual has been published in two volumes with Volume 2 being devoted entirely to the logic. Volume 1 begins with a discussion of the overall design of the computer, proceeding to a detailed analysis of the electronic design and, finally, to particular information dealing with the maintenance of the Central Processor. Material contained in Chapter 2 (Information Format) constitutes an exception to the rule of not repeating information from the Reference Manual. This chapter includes a discussion of the various G–20 word formats, a list of the G–20 commands (operation code list), and a brief description of the operand assembly process. The latter is included here to provide orientation into the manner of program execution. (Sections 9.1 – 9.3 cover the same material from the point of view of the logical implementation.) Most of this information appears in the General Reference Manual. Its inclusion here can be justified on the grounds that it is referred to frequently and should, therefore, be readily available for reference. Further, its basic nature is such that all logical manipulations discussed in the manual result from some aspect of the format and, thus, a solid grasp of these ideas is essential. Chapter 3 (Introduction to Machine Organization) is devoted to a general description of the machine layout with particular emphasis being placed on the machine diagram. Chapter 4 describes the Central Processor control registers: their uses, the existing transfer paths, means available for shifting information, and the general relationships between these registers and the logic. The importance of the CD register (Command Decoding) is stressed since the decoding that takes place here controls all subsequent activities. Chapter 5 introduces the Arithmetic Unit:

the registers and their inter—relationships, adder circuitry and operation, exponent circuitry, etc. Chapter 6 covers the operation of the memory system: the circuits, handling of address decoding and memory parity, timing considerations, connections of the system to external memory, and the operation of the system using external memory units. The basic Central Processor electronic design is discussed in Chapter 7. This includes descriptions of logic circuits, communication circuits, and so on. Chapter 8 is devoted to a generalized discussion of maintenance of the Central Processor: the tools required and the procedures to be followed.

The analysis of the logic in Volume 2 begins with a description of Master Control, the section of the logic that determines the handling of each command word, in Chapter 9. Chapter 10 introduces several short, miscellaneous operations, Chapter 11, the logic involved in memory operations, 12, input/output, and 13, the use of the adder circuitry. This breakdown is discussed in detail in Sections 9.1 and 10.1.

Basic documentation for the G—20 Central Processor consists of the drawings, schematics, wire list, and logic flow charts. All of the flow charts appear in Volume 2. Relevant sections of the drawings are included in Volume 1. The wire list and schematics, which relate the logic to the circuits that implement it, are available to those who require use of them.

Certain conventions in terminology have been used throughout this manual: one is discussed here, the others are pointed out as they arise. In each instance an attempt has been made to choose the most generally applicable term. Even so, no set of terms could be expected to satisfy every case; exceptions are bound to occur.

As an example of the choice of terminology, consider the problem of register bit designations. G–20 floating–point arithmetic is carried out to an accuracy of 42 bits so that all registers involved in such operations must be 42 flip–flops long. However, some registers have extra flip–flops at one or both ends (see Figure 3.2–2) that provide a means for retaining significance when certain overflow conditions occur. (The value is brought into normal range by means of shifts in conjunction with exponent increments or decrements.) These extra bits are also useful in the performance of shifts between registers. The existence of extra positions means that the least significant flip–flop in some registers holds the 0 order bit (Bit X 2^{0}) while in others it holds the –1 (Bit X 2^{-1}), –3 (Bit X 2^{-3}), or the –4 (Bit X 2^{-4}) order bit. Thus, reference to a particular bit on the basis of the number of the flip–flop storing it would not indicate the order of the bit. For this reason the convention adopted is tied to the order of the information, not to the number of the flip–flop. Each bit is named for the power of 2 with which it is associated. For example, in register S which has a single extra flip–flop at the lower end, the 0 order bit is stored in the second flip–flop but is referred to as Bit 0. For the M register, which has three extra flip–flops at the lower end, the 0 order bit will still be called Bit 0, while the first flip–flop in M contains Bit –3, the second, Bit –2, etc., and the fifth, Bit 1, the sixth, Bit 2, etc. This avoids the confusion that would result from use of flip–flop designations.

CHAPTER 2

INFORMATION FORMAT

SECTION 2.1 – MACHINE LANGUAGE WORDS

In a computing device as versatile as the G–20 Central Processor, it is necessary to deal with various types of information. Yet, the memory modules are of standard design with 32 bits reserved for the storage of a Central Processor word. (A memory word consists of 33 bits, but the thirty–third bit is reserved for parity.) It therefore becomes necessary to establish a system that allows the programmer to know exactly what information each 32–bit Central Processor word contains, and at the same time allows the Central Processor to know how to process it. Figure 2.1–1 lists all possible variations of word format that the Central Processor can use. These words are all in machine language (binary number system of 1's and 0's) that the Central Processor can either decode or use directly in information manipulations. These machine language words present the eventual form of all commands and data used by the Central Processor no matter how sophisticated the original program language may be.

2.1–1 COMMAND WORD FORMAT All instructions of a program processed by the Central Processor will appear in the command word format shown in Figure 2.1–1. An examination of this figure reveals that the command word is broken down into several distinct sections. The 15 least significant bits (bits 0 to 14) are referred to as the Base

Address, A. This part of the command may be used as an operand or an address of an operand depending upon the mode of operation. (Operating modes are discussed in Section 2.3.) The range of numbers in this region is from 0 to $32,767_{10}$. Thus, it can be seen that if the A field is being used as an address of an operand, any location in memory can be selected.

The next 6–bit section of the command word (bits 15 to 20) is the index, I, field and designates index addresses. This I field makes it possible to reference an index location and some other memory location while using a single command word, thus saving time and memory space. Since 6 bits have been assigned to this section, the I field can address only the first 64 locations in memory. An I field equal to zero is a special case that means no index address is specified and consequently, only 63 index addresses are available. We will digress momentarily from the command word and discuss these index addresses since an early, basic understanding of them and their uses is quite important.

Programmers often have need for counters within their programs. These counters, or tallies, are incremented or decremented each time a predetermined event occurs during the processing of a program. The number remaining in the tally at the end of the processing of the program may be included as part of the result. Also, there are many cases where it is not only desirable to maintain a tally on a particular operation, but also to know when a predetermined number of repetitions has occurred. Thus, a tally and a test are needed. Any location in memory could be used as a tally, but it would take three commands to do a tally or a tally and test operation. A group of commands designed to provide modifications of the index locations in memory, however, can accomplish a tally or a tally and test operation by the use of only one of these special commands. The numerical value of this changing tally stored in one of the index locations can also be used to modify the

operand X of a command word. This use of the index locations is discussed in Section 2.3.

The 7 bits of the opcode section of the command word (bits 21 to 27) contain one of the opcodes (operation codes) listed in Table 2.2–1. The configuration of these 7 bits will later be decoded by the Central Processor to determine necessary steps and internal paths used in the processing of the opcode. It will be noted that Table 2.2–1 lists all of the opcodes as 3–digit octals (9 bits), whereas only 7 bits are available for the opcode in the command word. This is acceptable, however, since the most significant octal digit of an opcode is never greater than 1 octally (001 in binary). Thus, it is seen that indeed only 7 bits are needed to represent any opcode octal representation. The two most significant bits of the most significant octal of the opcode are used by the command word (bits 28 and 29) to indicate the mode of operation. There are four operating modes available (discussed in Section 2.3). The opcode octal representation as commonly listed will appear quite different from the corresponding octal in the command word. This relationship is shown in Table 2.1–1.

TABLE 2.1–1 Command Word Octal Corresponding to Most Significant Digit of Opcode List Presentation

	Most Significant Octal Digit in Opcode List	
Mode	0	1
0	0	1
1	2	3
2	4	5
3	6	7

The remaining 2 bits of the command word are flag bits. These are

enabled interrupt request portions available to the programmer. When either one of these 2 bits is high, bits of the Interrupt Request register J are set if the corresponding bits of the U register are set, and the program will be interrupted if the UJE flip–flop is set. The Interrupt Service Routine will process the interrupt caused by a flag bit being set, which, upon satisfactory completion, will return control to the next command of the program. These flag interrupts are very useful in specific cases.

For example, command flag 31 could initiate the tracing (i.e., the print out of each command and the contents of the Accumulator) of a program and command flag 30 could terminate it. It should be noted at this time that the flag bits of any word format other than logic format will always be cleared to zero prior to being placed into memory by one of the store opcodes listed in Table 2.2–1. If it is desired to place a flag in a word (either command or data) once the program or data is in memory, it is necessary to perform a logic operation on the desired word location, change the desired bits (bits 31 or 30), and then store the word in logic format.

2.1–2 DATA WORD FORMAT All computer words presented in Figure 2.1–1, with the exception of the command word are data words.

Single Precision Numbers. A single precision number word, as can be seen from Figure 2.1–1, is made up of several distinct sections. Bits 0 through 20 contain the mantissa of the operand. Bits 21 through 26 contain the exponent section. The exponent section contains a 2–digit octal number representing the power of eight by which the mantissa is to be multiplied. Bit 27 holds the sign of the exponent (0 = plus, 1 = minus) and bit 28 holds the sign of the number in the mantissa (0 = plus, 1 = minus). Bit 29 is the length tag which is 0 for single precision numbers (bit 29 = 1 indicates double precision). Bits 30 and 31 are

flag bits which provide programmable interrupts.

Double Precision Numbers. Double precision number words are made up of two 32–bit Central Processor words. The mantissa of the first of the two words contains the 21 least significant bits of the 42–bit operand and the mantissa of the second word contains the 21 most significant bits. Bits 21 through 31 are the same as in a single precision number word with the exception that bit 29, the length tag will be a 1, designating a double precision word. It should be noted that bits 21 through 31 of both words of the double precision number will be identical. Bits 31 through 21 of the second word, however, are not needed by the Central Processor decoding and are ignored. When bit B29 is high and logic opcodes are not being processed, all operations will be done in floating point even if pickapoint operations are indicated by the pickapoint mode flip–flop, UPE.

Floating Point Integer. Floating point integer words are also very similar to the single precision number word format, except that the exponent (bits 21 to 26) is always set to zero and the sign of the exponent is plus (bit 27 always set to zero). It should also be noted that the contents of the mantissa of the floating point integer are quite different in basic concept from the contents of the mantissa of a single precision number word. The mantissa of the single precision number word contains the rounded 21 most significant bits of a variable exponent number, while the mantissa of a floating point integer contains the truncated 21 least significant bits of a zero exponent (8^0) number.

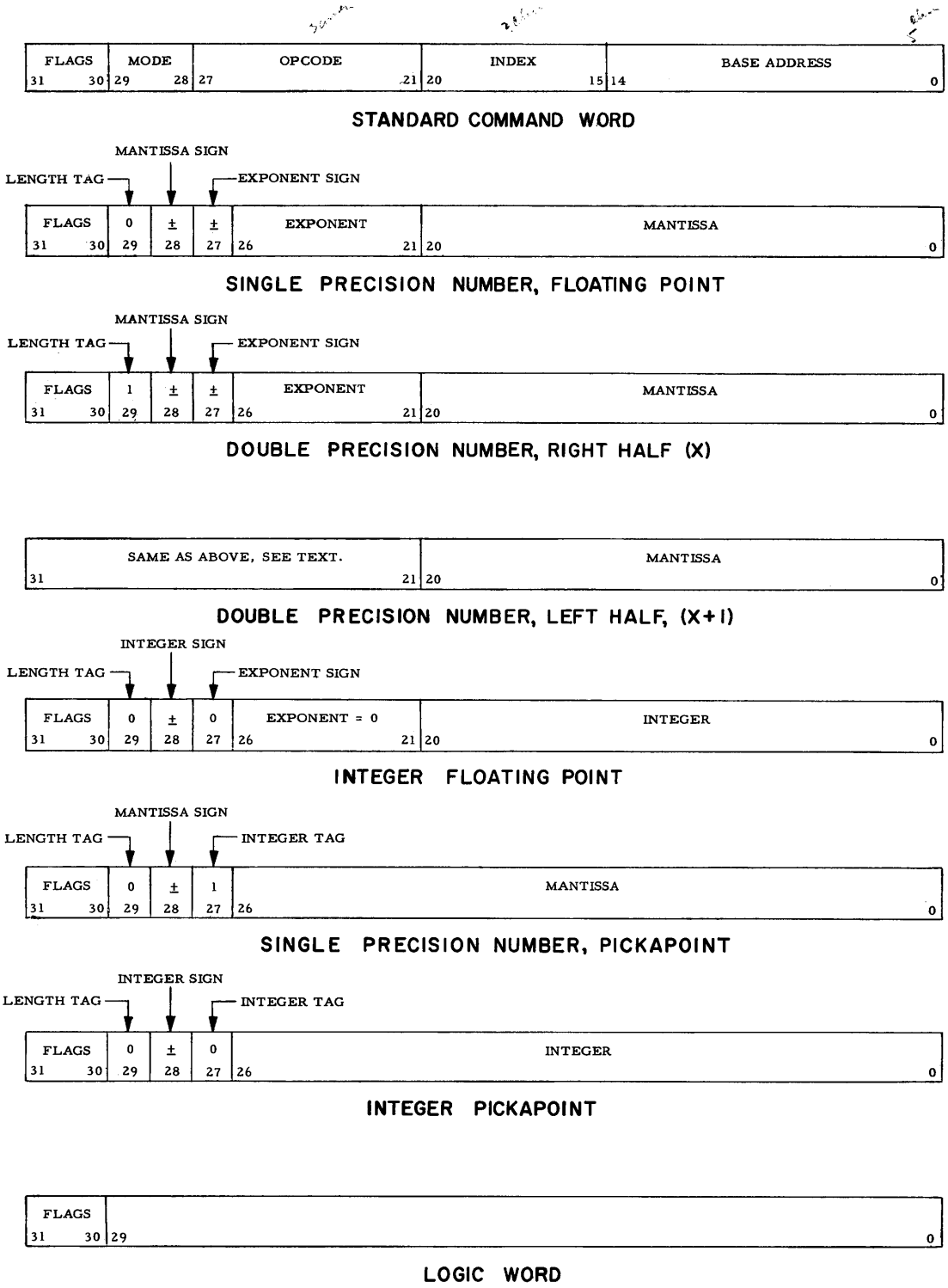
Pickapoint Single Precision Number. In the pickapoint single precision word, the mantissa (bits 0 through 26) contains the binary representation of the operand. The exponent and the sign of the exponent, normally positioned in bits 21 through 27, in the pickapoint mode of operation are held in the PE register (a 7–bit register). Since the

contents of the PE register can only be changed by the programmer, this allows computation referenced to some preassigned exponent value. This facilitates the processing of fixed point computations. Bit 27 is 1 indicating a pickapoint single precision number as opposed to a pickapoint integer where bit 27 equals 0. Bit 28 holds the sign of the mantissa (0 = plus, 1 = minus), and the length tag (bit 29) is 0 indicating a single rather than double precision word (pickapoint mode operations can only occur in single precision). Bits 30 and 31 are the flag bits.

Pickapoint Integer. The pickapoint integer is very similar to the pickapoint single precision number format. With pickapoint integer, however, bit 27 of the word is set to zero to indicate an integer word and, therefore, the 7 bits of the PE register are set to zero (zero exponent). Also, the basic concept of the mantissa of the pickapoint single precision number and of the pickapoint integer is quite different. The mantissa of the pickapoint single precision number contains the rounded 27 most significant bits of a word to some variable, predetermined exponent, whereas the mantissa of a pickapoint integer contains the truncated 27 least significant bits of a zero exponent number (8^0).

Logic Word. The logic word is made up of two parts, the flags (bits 30 and 31) and 30 bits of information (bits 0 to 29) that have no exponent associated with them. These words have no use in actual mathematical computations; however, used with the logic commands they provide an easy means of manipulating or changing command or data words within the machine. The previously discussed use of logic words to provide flags to command or data words once they are in memory is one illustration of their uses.

FIGURE 2.1–1 Central Processor Word Formats



SECTION 2.2 – OPCODE LIST

TABLE 2.2–1 Opcode List

OPCODE NOTATION CROSS REFERENCE

Octal	Alpha	Engr.
000	OCA	N0
001	FOP	T0
002	ADX	B2
005	CLA	A0
006	AXT	B6
011	IOZ	S0
012	LXP	B0
013 [opcode]	REP	M0
015	CAL	L0
016	XPT	B4
017	TRA	X0
020	OCS	N1
021	FOM	T1
022	SUX	B3
025	CLS	A1
026	SXT	B7
031	ICZ	S1
032	LXM	B1
033 [*]	BTR	M1
035	CCL	L1
036	XMT	B5
037	TRE	X1
040	OAD	N2
041	FSP	T2
045	ADD	A2
051	ISN	S2
052	ERO	R2
053	DIV	D0
055	ADL	L2
056	LDR	R0
057	RDV	D1
060	OSU	N3
061	FGO	T3
065	SUB	A3
071	IUO	S3
072	ERA	R3
073	STZ	P4
075	SUL	L3
076	EXR	R1
077	MPY	D2
100	OAN	N4
101	FSM	T4
105	ADN	A4
111	IEZ	S4
113	STS	P1
115	EXL	L4
117	TDC	X5
120	OSN	N5
121	FLO	T5
125	SUN	A5
131	IEC	S5
133	STI	P3
135	ECL	L5
137	SKP	X2
140	OAA	N6
141	FSN	T6
145	ADA	A6
151	IUZ	S6
153	STD	P0
155	UNL	L6
157	TLC	X4
160	OSA	N7
161	FUO	T7
165	SUA	A7
171	IUC	S7
173	STL	P2
175	UCL	L7
177	TRM	X3
* 000	BTD6	
* 020	BRD6	
* 040	BTC6	
* 100	BTD8	
* 120	BRD8	
* 140	BTC8	

ADDRESS PREPARATION

USERS CODE	OCTAL CODE	ENGR. CODE	NAME	OPERATION
OCA	000	N0	Clear and Add	X → (OA)
OCS	020	N1	Clear and Subtract	−X → (OA)
OAD	040	N2	Add	X + (Acc) → (OA)
OSU	060	N3	Subtract	−X + (Acc) → (OA)
OAN	100	N4	Add and Negate	−[X + (Acc)] → (OA)
OSN	120	N5	Subtract and Negate	−[−X + (Acc)] → (OA)
OAA	140	N6	Add and Take Absolute Value	X + (Acc) → (OA)
OSA	160	N7	Subtract and Take Absolute Value	−X + (Acc) → (OA)

ADD AND SUBTRACT

CLA	005	A0	Clear and Add	X → (Acc)
CLS	025	A1	Clear and Subtract	−X → (Acc)
ADD	045	A2	Add	X + (Acc) → (Acc)
SUB	065	A3	Subtract	−X + (Acc) → (Acc)
ADN	105	A4	Add and Negate	−[X + (Acc)] → (Acc)
SUN	125	A5	Subtract and Negate	−[−X + (Acc)] → (Acc)
ADA	145	A6	Add and Take Absolute Value	X + (Acc) → (Acc)
SUA	165	A7	Subtract and Take Absolute Value	−X + (Acc) → (Acc)

Note: The Accumulator is not disturbed in these opcodes.

ADD AND SUBTRACT TESTS

				Criterion
FOP	001	T0	If Operand Plus	X > 0
FOM	021	T1	If Operand Minus	−X > 0
FGO	061	T3	If (Acc) Greater Than Operand	−X + (Acc) > 0
FLO	121	T5	If (Acc) Less Than Operand	−[−X + (Acc)] > 0
FUO	161	T7	If (Acc) Unequal to Operand	−X + (Acc) > 0
FSP	041	T2	If Sum Plus	X + (Acc) > 0
FSM	101	T4	If Sum Minus	−[X + (Acc)] > 0
FSN	141	T6	If Sum Non–zero	X + (Acc) > 0

Note: If test is satisfied, go to next command of program (NC). If test is not satisfied, go to NC + 1.

LOGIC OPERATIONS

CAL	015	L0	Clear and Add Logic Word	31[X]0 → (Acc)
CCL	035	L1	Clear and Add Complement of Logic Word	31[$\overline{X}$]0 → (Acc)
ADL	055	L2	Add Logic Word	31[X + (Acc)]0 → (Acc)
SUL	075	L3	Subtract Logic Word	31[−X + (Acc)]0 → (Acc)
EXL	115	L4	Extract With Logic Word	31[X ∧ (Acc)]0 → (Acc)
ECL	135	L5	Extract With Complement of Logic Word	31[$\overline{X}$ ∧ (Acc)]0 → (Acc)
UNL	155	L6	Unite With Logic Word	31[X ∨ (Acc)]0 → (Acc)
UCL	175	L7	Unite With Complement of Logic Word	31[$\overline{X}$ ∨ (Acc)]0 → (Acc)

Note: $0 \rightarrow {}_{41}[(\text{Acc})]_{32}$ for all of these codes.

LOGIC TESTS

				Criterion
IOZ	011	S0	If Operand Zero	31[X]0 = 0
ICZ	031	S1	If Complement of Operand Zero	31[$\overline{X}$]0 = 0
ISN	051	S2	If Sum Non–zero	31 X + (Acc) 0 > 0
IUO	071	S3	If Unequal to Operand	31 −X + (Acc) 0 > 0
IEZ	111	S4	If Extraction Zero	31[X ∧ (Acc)]0 = 0
IEC	131	S5	If Extraction With Complement Zero	31[$\overline{X}$ ∧ (Acc)]0 = 0
IUZ	151	S6	If Union Zero	31[X ∨ (Acc)]0 = 0
IUC	171	S7	If Union With Complement Zero	31[$\overline{X}$ ∨ (Acc)]0 = 0

Note: If test is satisfied, go to next command of program (NC). If test is not satisfied, go to NC + 1.

MULTIPLY AND DIVIDE

MPY	077	D2	Multiply	(Acc) * X → (Acc)
DIV	053	D0	Divide	(Acc) / X → (Acc)
RDV	057	D1	Reverse Divide	X / (Acc) → (Acc)

USERS CODE	OCTAL CODE	ENGR. CODE	NAME	OPERATION
STORE OPERATIONS				
STL	173	P2	Store Logic Word	31(Acc)0 → 31 (X) 0
STD	153	P0	Store Double Precision	20(Acc)0 → 20 (X) 0 41(Acc)21 → 20(X + 1)0
STS	113	P1	Store Single Precision	
			[a] Floating Point Mode	41(Acc)21 → 20 (X) 0
			[b] Pickapoint Mode	41(Acc)15 → 26 (X) 0
STI	133	P3	Store Integer	
			[a] Floating Point Mode	20(Acc)0 → 20 (X) 0 0 → 21 (X)26
			[b] Pickapoint Mode	26(Acc)0 → 26 (X) 0
STZ	073	P4	Store Zero	0 → 31 (X) 0
INDEX OPERATIONS				
LXP	012	B0	Load Index Plus	X → (I)
LXM	032	B1	Load Index Minus	−X → (I)
ADX	002	B2	Add to Index	X + (I) → (I)
SUX	022	B3	Subtract from Index	−X + (I) → (I)
INDEX TESTS				
XPT	016	B4	Load Index Plus and Test	X → (I)
XMT	036	B5	Load Index Minus and Test	−X → (I)
AXT	006	B6	Add to Index and Test	X + (I) → (I)
SXT	026	B7	Subtract from Index and Test	−X + (I) → (I)
Note: If final value of (I) is not zero, go to next command, NC, of program. If final value of (I) is zero, go to NC + 1.				
REGISTER OPERATIONS				
LDR	056	R0	Load Register: * U, H, J	14 X 0 → (Reg. I)
			PE	6 X 0 → (Reg. I)
EXR	076	R1	Extract Register I into Itself	14[(Reg. I) ∧ X]0 → (Reg. I)
ERO	052	R2	Extract Register I into QA	14[(Reg. I) ∧ X]0 → (QA)
ERA	072	R3	Extract Register I into Acc	14[(Reg. I) ∧ X]0 → (Acc)
Note: Registers available for these operations are CA, U, H, J, and PE. * The CA register can be read only by an ERO or ERA opcode.				
TRANSFER OF CONTROL				
TRA	017	X0	Transfer [to Location X]	14[X]0 → (NC)
TRE	037	X1	Transfer and Enable Interrupts	14[X]0 → (NC)
SKP	137	X2	Skip [X Words]	14[X + (NC)]0 → (NC)
TRM	177	X3	Transfer [to Location X + 1] and Mark [in Location X]	(NC) → (X) 14[X + 1] 0 → (NC)
SINGLE CHARACTER OUTPUT				
TLC	157	X4	Transmit Line Command	7[X]0 → 7(LD)0 0 → (LD8)
TDC	117	X5	Transmit Data Character	7[X]0 → 7(LD)0 1 → (LD8)
BLOCK INPUT/OUTPUT				
BTR	033 [*]	M1	Block Transmit or Receive	X is address of first operand in the block
	BTC6		Block Transmit Commands	6-bit characters
	BTC8		Block Transmit Commands	8-bit characters
	BTD6		Block Transmit Data	6-bit characters
	BTD8		Block Transmit Data	8-bit characters
	BRD6		Block Receive Data	6-bit characters
	BRD8		Block Receive Data	8-bit characters
REPEAT OPERATIONS				
REP	013 [opcode]	M0	Repeat Next Command	X = address of first operand of block.

Note: All 32 opcodes of the Add/Subtract, Add/Subtract Test, Logic, and Logic Test commands can be performed using the Repeat mode. Each Repeat operation is designated by two words, the first of which contains the opcode 013. The opcode of the second word can be any of the above mentioned commands. Commands that are to be repeated are the same as their non-repeat counterpart except that an R (to represent Repeat mode) is affixed to the mnemonic of the command. Repeat commands have a block length associated with them (base address of second word of command). A Repeat Test command will continue until the condition tested for is met.

OPCODE NOTATION CROSS REFERENCE

Alpha	Octal	Engr.
ADA	145	A6
ADD	045	A2
ADL	055	L2
ADN	105	A4
ADX	002	B2
AXT	006	B6
BTR	033 [*]	M1
CAL	015	L0
CCL	035	L1
CLA	005	A0
CLS	025	A1
DIV	053	D0
ECL	135	L5
ERA	072	R3
ERO	052	R2
EXL	115	L4
EXR	076	R1
FGO	061	T3
FLO	121	T5
FOM	021	T1
FOP	001	T0
FSM	101	T4
FSN	141	T6
FSP	041	T2
FUO	161	T7
ICZ	031	S1
IEC	131	S5
IEZ	111	S4
IOZ	011	S0
ISN	051	S2
IUC	171	S7
IUO	071	S3
IUZ	151	S6
LDR	056	R0
LXM	032	B1
LXP	012	B0
MPY	077	D2
OAA	140	N6
OAD	040	N2
OAN	100	N4
OCA	000	N0
OCS	020	N1
OSA	160	N7
OSN	120	N5
OSU	060	N3
REP	013 [opcode]	M0
RDV	057	D1
SKP	137	X2
STD	153	P0
STI	133	P3
STL	173	P2
STS	113	P1
STZ	073	P4
SUA	165	A7
SUB	065	A3
SUL	075	L3
SUN	125	A5
SUX	022	B3
SXT	026	B7
TDC	117	X5
TLC	157	X4
TRA	017	X0
TRE	037	X1
TRM	177	X3
UCL	175	L7
UNL	155	L6
XMT	036	B5
XPT	016	B4
* BRD6	020	
* BRD8	120	
* BTC6	040	
* BTC8	140	
* BTD6	000	
* BTD8	100	

SECTION 2.3 – OPERAND ASSEMBLY

The assembling of the operand X prior to processing an opcode is common to all opcodes used by the Central Processor. Therefore, it is seen that a thorough understanding of the process and significance of assembling the operand X is necessary for effective use of the Central Processor's capabilities. It is assumed that at this point the reader has had some contact with what is meant by the operand X and thus only a general discussion will be presented. If this assumption is incorrect, or if the reader is at all hazy about the subject after reading the following discussion, then the reader should refer to Section 2.2 of the General Reference Manual or the first section of the Central Processor Machine Language Manual. Detailed discussions of operand assembly, with examples, are provided in the above-mentioned manuals. The value of a basic understanding of this material cannot be too greatly emphasized since a thorough understanding of much of the material presented in later chapters depends upon mastery of the concepts of operand assembly.

A command word in the Central Processor, as we have seen, carries an operating mode M, an index address I, and a base address, A. All of these, along with the contents of the OA register, can be used in assembling the operand X of a command word. The exceptions to this rule are the command words with opcodes which operate on one of the Bus registers or an index location, since in these cases the index address I is used to specify the register or memory location (1 through 63) to be operated on. It should also be noted that the use of the contents of the OA register in assembling the operand X is not common practice, since the contents of the OA register are always set to zero before assembling X if the previous opcode that was processed was not one of the address preparation opcodes. The function of the address preparation opcodes is discussed in the section below dealing with the general case

of operand assembly.

The operating modes associated with a command word specify the manner in which A, I, and OA are to be used in the assembling of X. Table 2.3–1 shows decoding that indicates the mode of operation.

TABLE 2.3–1 Operating Mode Decoding

Mode	B register bits 29	28	CD register bits 14	13
0	0	0	0	0
1	0	1	0	1
2	1	0	1	0
3	1	1	1	1

The general case of operand assembly is the most versatile and the most commonly used. In it, it is assumed that (OA), (I), and A are all available for use. There is a case that will be discussed later where the above assumption is not valid. The manner in which the operand is formed in the different modes is presented in Table 2.3–2.

By use of the general case of operand assembly, it is seen that a large number of modifications can be made to the operand by use of the contents of the OA register and indexing. In operand assembly, indexing can be very limitedly defined as using the contents of the index locations to change the operand of the command. A more complete discussion of the index locations and their uses is presented in Section 2.1. The address preparation opcodes provide a second means of modifying the operand, which, if desired, permits the contents of the Accumulator to be combined in various ways in the assembling of X.

TABLE 2.3–2 Operand Assembly, General Case

Mode	Action	Format
0	(OA) + A + (I) = X	Number
1	(OA) + (A) + (I) = X	Number
2	((OA) + A + (I)) = X	Number or Logic*
3	((OA) + (A) + (I)) = X	Number or Logic*

where OA = Operand Assembly register
X = operand designator
A = base address
I = index address
( ) = contents of (i.e., an address)

Logic* ⇒ logic format on final access for logic or logic test opcodes.

It was noted earlier that there are opcodes where the general case of operand assembly is not applicable. These opcodes belong to the Index or Bus register command groups. In these opcodes, the I field of the command word is used as an identification tag. If one of the index opcodes is being processed, the I field of the command word specifies which of the 63 index locations is to be operated on. If it is one of the Bus register opcodes, the I field indicates which of the five Bus registers (CA, J, H, U, or PE) is to be operated on. The method of assembling the operand X under these conditions is the same as shown in Table 2.3–2 with the exception that with these groups of opcodes, the I field, (I), does not enter into the assembling of X. The assembling of the operand X under these conditions is presented in Table 2.3–3.

In all instances, the arithmetic processes involved in assembling the

operand are done in number format. This means that numbers brought from memory to be added are brought out and processed as floating point full or double precision, or pickapoint numbers. If the operating mode is zero and only the A field exists, the number contained therein becomes the operand X with positive sign and zero exponent. If logic opcodes or logic test opcodes are being processed in modes 2 or 3, the final memory access is made in logic format and, thus, the final assembled operand X is in logic format for these opcodes. In all other modes of the various opcodes, if not one of the groups mentioned above, the final assembled operand X is in number format.

TABLE 2.3–3 Operand Assembly for Index Location and Bus Register Addressing Opcodes

Mode	Action	Format
0	(OA) + A = X	Number
1	(OA) + (A) = X	Number
2	((OA) + A) = X	Number
3	((OA) + (A)) = X	Number

CHAPTER 3

INTRODUCTION TO MACHINE ORGANIZATION

SECTION 3.1 – GENERAL

Similar to most all digital computer design, the Central Processor is comprised of the four basic logic areas of: (1) Control, (2) an Arithmetic Unit, (3) Memory, and (4) Input/Output. This in itself is neither new or unusual, however, the method in which these areas of logic are handled within this particular computer needs to be thoroughly discussed since the means of implementing these areas can vary greatly.

FIGURE 3.1–1 Block Diagram of Central Processor Organization

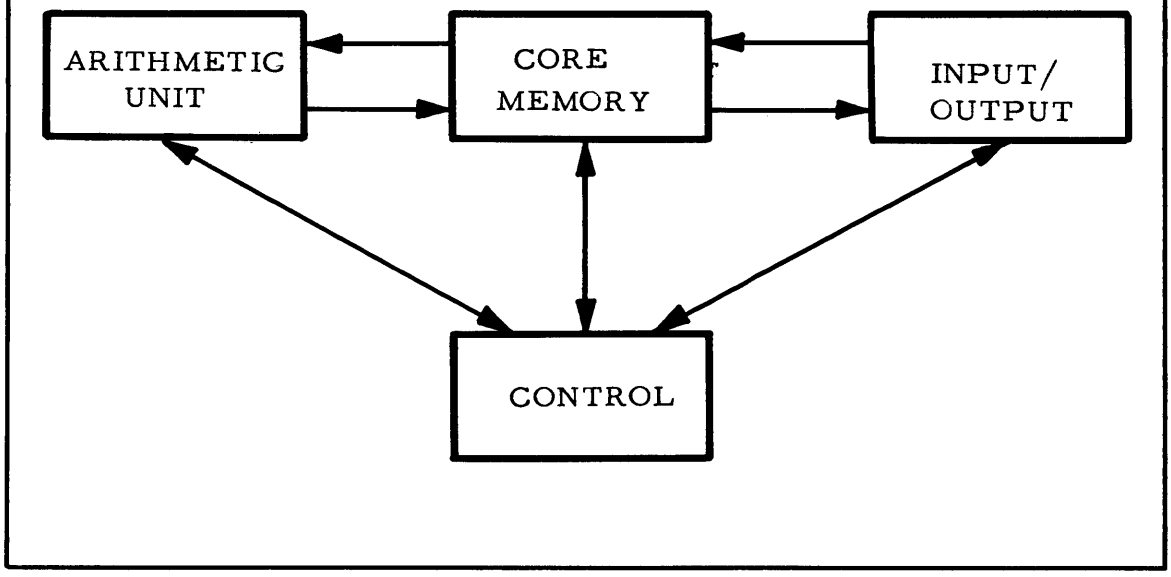
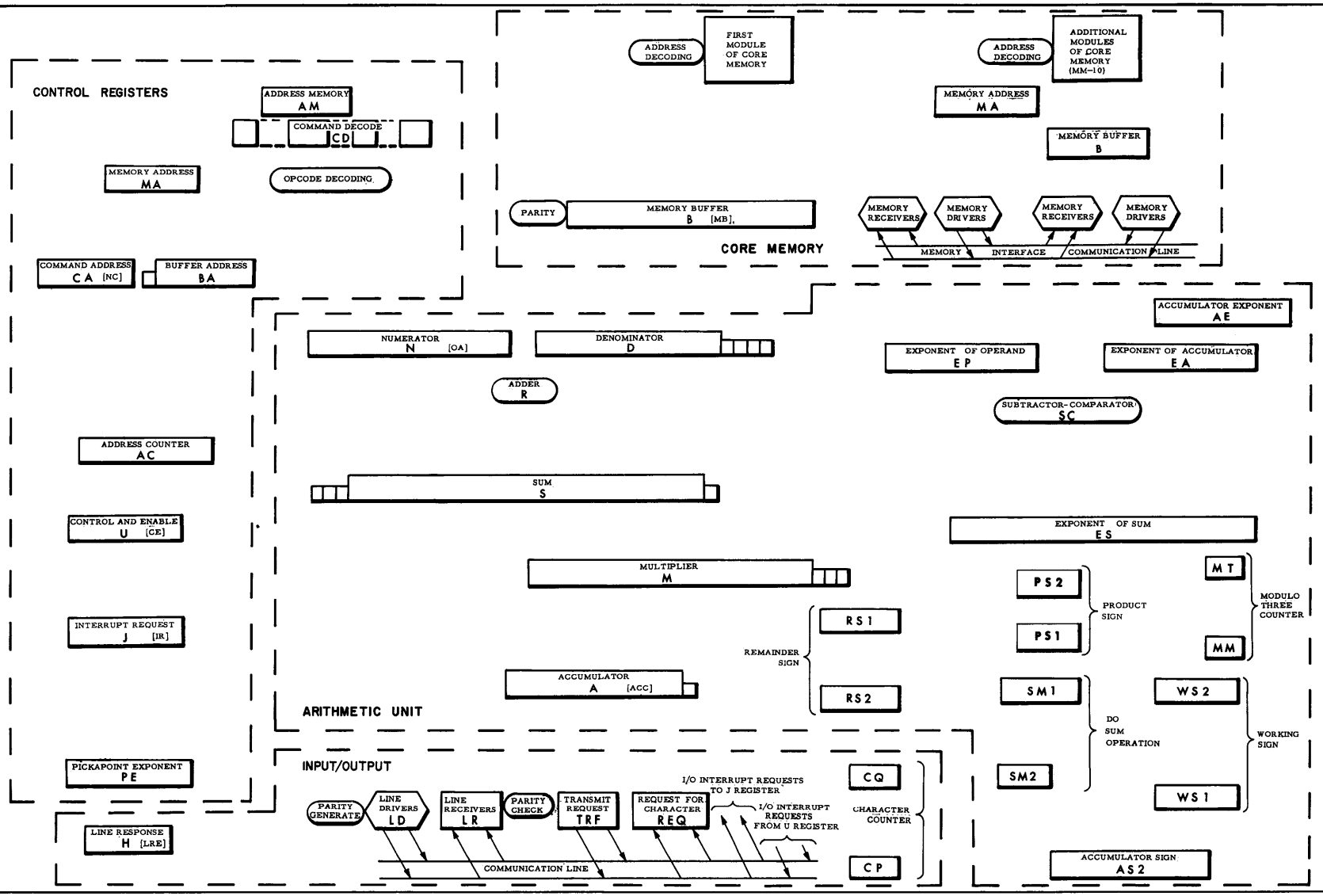


Figure 3.1–2 is a breakdown by Central Processor registers of Figure 3.1–1. This figure shows registers of the various logic areas, but this is not a complete logic breakdown. In particular, the area of control is comprised of control registers and sequencers. The sequencers are in reality a complex of logic circuitry that supervises the operation of the opcodes listed in Table 2.2–1. This logic circuitry is not shown in Figure 3.1–2 because of its complexity. Rather, for simplicity of presentation, all sequencer logic is discussed in Part IV, Sequencer Control. The discussion of control that is presented in Chapter 4 deals solely with control exercised by the registers of Figure 3.1–2 that are designated as the Control registers. Also, the reader should realize that the breakdown of input/output as presented by the Machine Diagram is not complete, since it deals only with provisions within the Central Processor for input/output. For a discussion of actual external input/output devices, the reader is referred to the General Reference Manual. If a more specific discussion of a particular input/output unit is desired, then that peripheral unit's technical manual must be consulted. It should also be pointed out that input/output operations of the Central Processor are not discussed in this part of the manual. Rather, due to the fact that a very thorough discussion of input/output is necessary in Chapter 12 of Part IV where Sequencer Control logic is discussed, the complete presentation of input/output is postponed until Chapter 12.

FIGURE 3.1–2 Central Processor Logic Breakdown of Figure 3.1–1



SECTION 3.2 – MACHINE DIAGRAM

Figure 3.1–2 shows the functional breakdown of the Machine Diagram by registers. This section, however, presents the complete picture of the inter–relations of these various registers. Figure 3.2–2 shows a simplified presentation of the various transfer paths and register modification signals of the Machine Diagram. The reader should acquaint himself with this figure since, although transfer paths are sometimes presented more simplified than they actually are, it is usually sufficient for most needs.

3.2–1 MACHINE DIAGRAM LEGEND The Machine Diagram contains an amazing amount of information concerning the organization of the machine. Unfortunately, to the uninitiated the meanings of the various symbols and their contents are not intuitively obvious. Therefore, it will pay the reader to pay close attention to the following material.

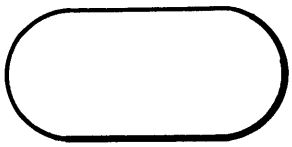
. This symbol on the Machine Diagram represents registers which are defined as locations within the machine where information can be temporarily stored. The length of the rectangle bears no direct relationship to the number of flip–flops it represents. The rectangle will normally contain the following information:

1) the engineering mnemonic name of the register,
2) the proper name of the register (if it has one),
3) the total number of flip–flops in the register,
4) the programmers mnemonic name of the register, in brackets, if one other than the engineering mnemonic exists.

On the Machine Diagram, the reader will find several registers that seemingly have projections on one or both ends. These projections are

flip-flops the same as the other bits of the register. The reason they are represented differently is to indicate to the reader that these bits serve a special purpose for the register. They provide extra storage for the register so that information will not be lost during shift transfers between registers.

. Non-register groups of signal circuits (such as line drivers, boosters, etc.) having no memory, which bear a one-to-one relationship to the bits of some register or a portion of some register, are designated by a hexagon. This symbol normally contains the same type of information as that of the register rectangle.

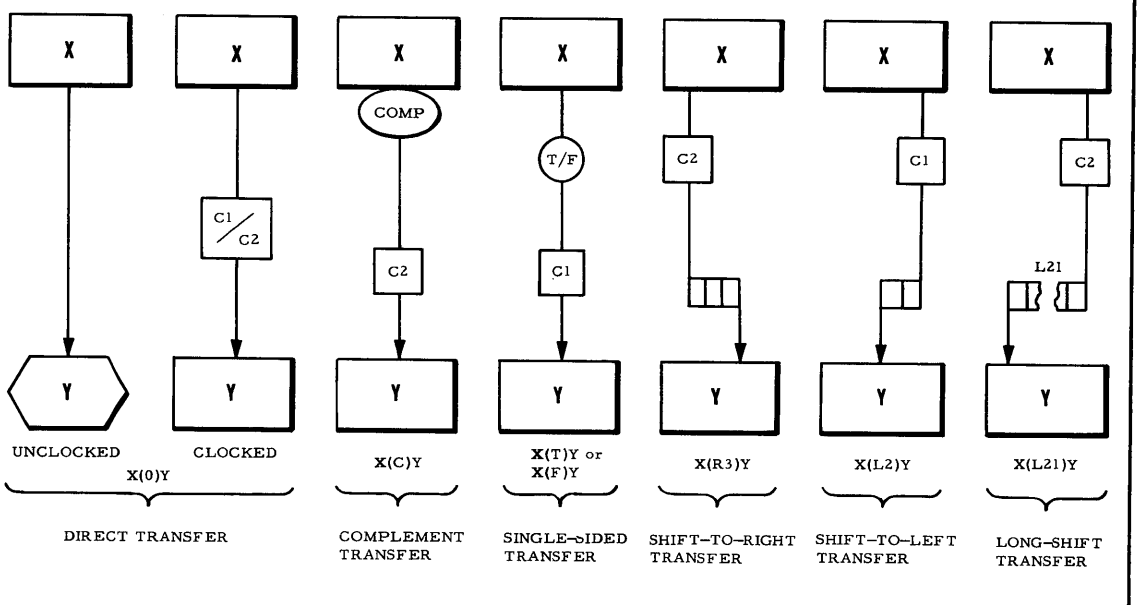


. Diode and inverter circuits that perform a logic function, such as parity checking, adding, etc., are designated by this symbol. These symbols contain sufficient information to indicate the function performed and any mnemonic symbols associated with them.

Transfer Paths. The Central Processor circuitry is such that two registers are needed to accomplish a shift (or alternatively, each of the registers is only half of a conventional shift register). To obtain the net effect of a shift, the contents of a register may be shifted to the left, to the right, or not at all during its transfer from one register to another. Transfers of information in Figure 3.1-2 are represented by single lines with arrows designating the direction of the transfer. Transfers of information are usually gated by a clock pulse, C1 or C2. If it is a clocked transfer, either C1 or C2 is placed in a square centered on the arrow. If transfers to the right or left are accomplished, this is indicated by additional squares offset in the direction of the shift. The number of squares will normally indicate the magnitude of the shift in the indicated direction. In some cases, however, the shifts are so

great that this method of indication of magnitude would be too messy. In such cases the offset squares will be broken and the magnitude of the shift printed above these boxes. Sometimes it is desirable to send the contents of one register in 1's complement form to another register. Such a transfer is indicated by an ellipse containing the letters COMP. There are also certain transfers that are single–sided. This means that only the true (XX high) or false ($\overline{XX}$ high) sides of the flip–flops of one register transfers information to another register or to memory. (To set or reset the flip–flops of a register to correspond to bits of another register, it is normally necessary to send both true and false signals from the register in question.) To indicate such single–sided transfers, a circle is centered on the arrow with the letters T or F enclosed. Figure 3.2–1 gives examples of the above–mentioned transfer paths.

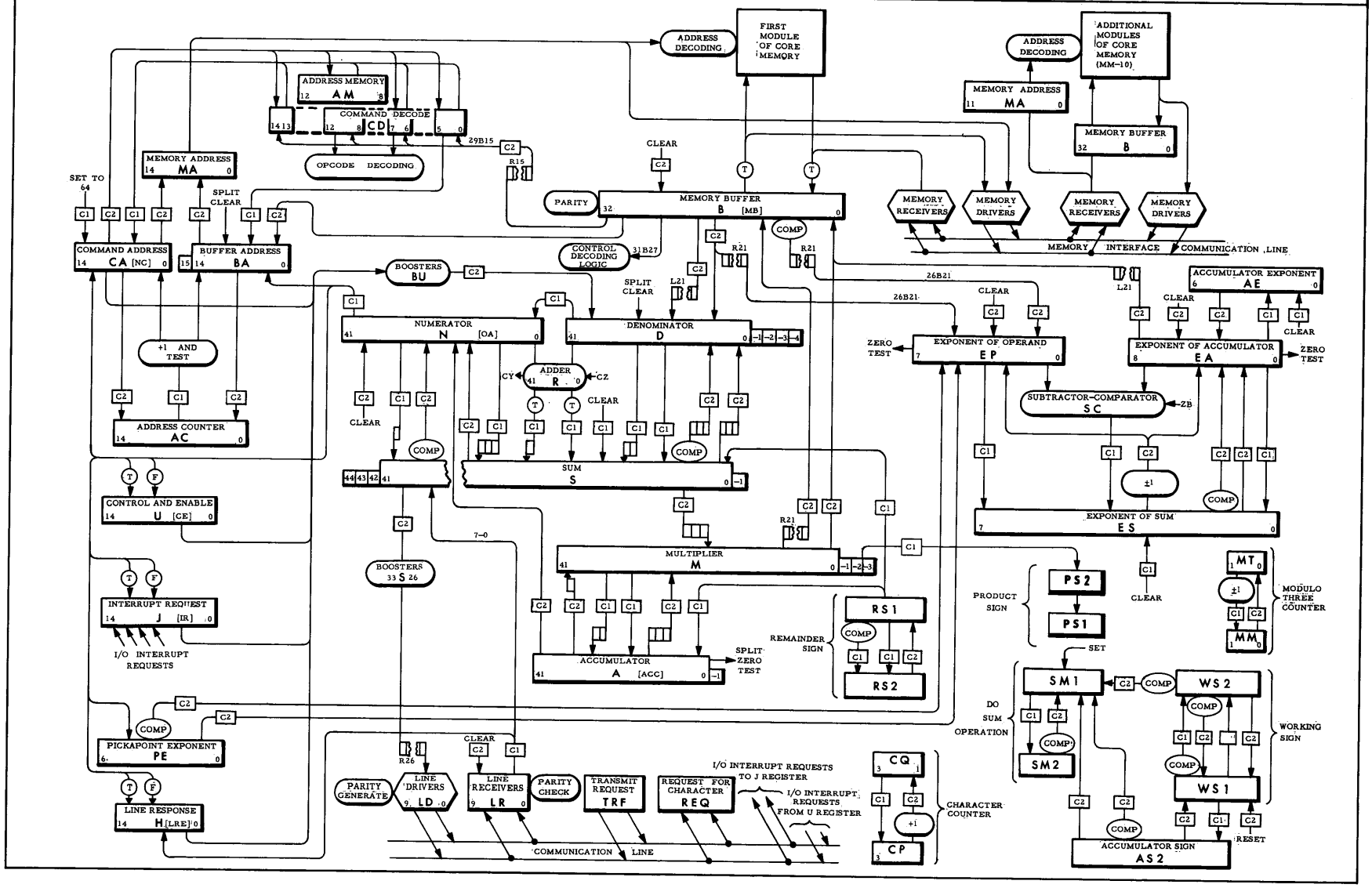
FIGURE 3.2–1 Transfer Path Symbols



Now that the reader is acquainted with the information necessary to

enable one to read and somewhat understand the Machine Diagram, a more detailed discussion of this area can be presented. The remainder of Part I of the manual is geared towards this objective.

FIGURE 3.2–2 Machine Diagram



CHAPTER 4

CONTROL REGISTERS

SECTION 4. 1 – ORIENTATION

This section is provided to give the reader a brief orientation to the control registers that are discussed in the remainder of this chapter. The material presented here is intended to provide a big picture rather than detailed information concerning these registers.

4. 1–1 MEMORY CONTROL REGISTERS The memory control registers consist of the MA, CA, BA and AC registers. Except for MA, their functions differ during computation operations and input/output or repeat operations.

The MA register always holds the address of the memory location that is to be read or filled. When the MA register is loaded, a signal is sent to the memory timing circuitry which initiates the memory cycle that takes the contents of the desired memory location and places it into the B register of the Arithmetic Unit.

During computation, the address of the next command is held in the CA register. When the command being processed by the computer is finished, the contents of CA are copied into the MA register and at the same time the address in the CA register is incremented by 1 by copying the (CA) into AC and then back into CA via the +1 logic circuitry. [( )

indicates "contents of".] The MA register starts the memory cycle which brings the new command from memory and places it in the B register of the Arithmetic Unit. When the new command arrives in B, the mode, opcode, and index bits of the command word are copied into the CD register. The opcode portion of the information in the CD register is then fed into the opcode decoding logic where the actual decoding that determines which operation is to be performed is done. The mode bits of the command word determine the destination of the base, A, of the command word. If the base, A, is to be used in unmodified form as the address of an operand, it is sent directly to the BA register, which is a buffer register for MA. The contents of BA are then copied into the MA register which starts the memory cycle which brings from memory the contents of the address designated by the base, A. If A is to be modified by the addition of the contents of the location designated by the index, I, and/or the contents of the OA register, then the A field is sent to the Arithmetic Unit rather than to BA. If the index bits of the command word are other than zero and are to be used to modify the base, A, they then become the address of a memory location and are sent to the BA register on their way to MA.

During block input/output or repeat operations, an address is sent from the Arithmetic Unit to the CA register. This acts as a transfer of control since the new contents of CA designate the starting location of the block input/output or repeat operation. This address is then repeatedly incremented by transfers to and from the AC register until the desired block has been processed. The block length of the block input/output or repeat operations is stored in 2's complement form in the BA register. Each time the address in the CA register is incremented by one the block length in BA is decremented by one by transfers between the BA and AC registers. When the contents of the BA register become zero, the block input/output or repeat operation is terminated. At the start of the block input/output or repeat commands,

the address of the next command word was sent from the CA register to the AM register for temporary storage. At the completion of this block-type operation, the next command address stored in AM is copied back into the CA register, and the regular program continues where it left off.

4.1-2 CONTROL INFORMATION STORAGE REGISTERS The control information storage registers are comprised of the U, J, and PE registers. The primary purpose of the U register is to enable inputs to the J register when so desired (by setting corresponding flip-flops of the U register). This should not be construed to indicate that U provides inputs to J. Rather, it means that if interrupt requests are present and the corresponding bit of U is set, then the interrupt requests are permitted to be registered in the J register. In addition, the U register controls whether or not any interrupt requests that are registered in J will be recognized by the machine. The U register also contains flip-flops that can transfer interrupts to external peripheral units, control the Central Processor tone signal, and determine whether the Central Processor will access and store information in floating point or pickapoint mode. The J register's sole purpose is to provide storage for interrupt requests. The PE register holds the exponent value selected for pickapoint operation.

SECTION 4.2 – COMMAND DECODING REGISTER – CD

The CD register is a 15–bit register whose primary function is to store the index opcode and mode fields of a command word while the command is being processed. These 15 bits are referenced by the Central Processor throughout the processing of commands to determine which of many paths to follow.

There is a secondary function of the CD register that concerns all bits of the CD register plus the flip–flops AM8 through 12. This function is to provide temporary storage for the address of the next command of a program in addition to storage for the opcode that is to be repeated when a block input/output or repeat opcode is being processed. Under these circumstances, all but bits 12 through 8 of the CD register are considered as part of the AM register which is discussed in Section 4.3–5. Block input/output and repeat operations are discussed in Section 4.2–4.

Figure 4.2–1 shows the CD register and transfer paths associated with it during normal command decoding. Detailed discussions of the artificially created divisions of the CD register shown in Figure 4.2–1 are presented in the following subsections.

4.2–1 INDEX SECTION OF CD REGISTER, NORMAL COMMAND DECODING* Figure 4.2–2 shows important register transfer and modification paths associated with the index section of the CD register during normal command decoding.

*Normal command decoding is taken to mean other than block input/output or repeat command decoding. Block input/output and repeat commands use the CD register in a somewhat different manner. For this reason, they are discussed separately in Section 4.2–4.

FIGURE 4. 2–1 Command Decoding Register Transfer Paths

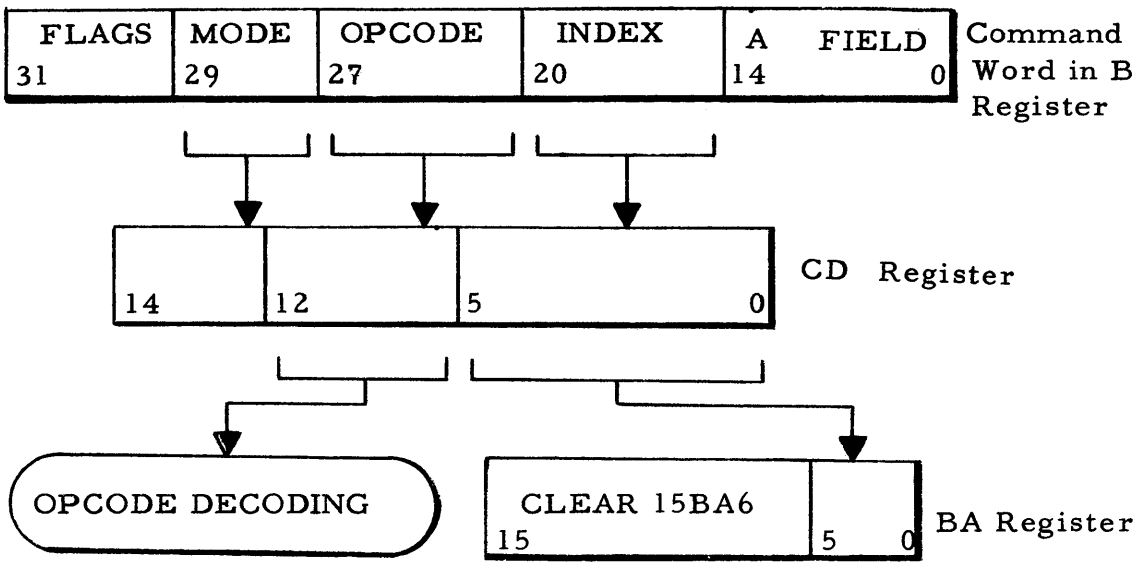
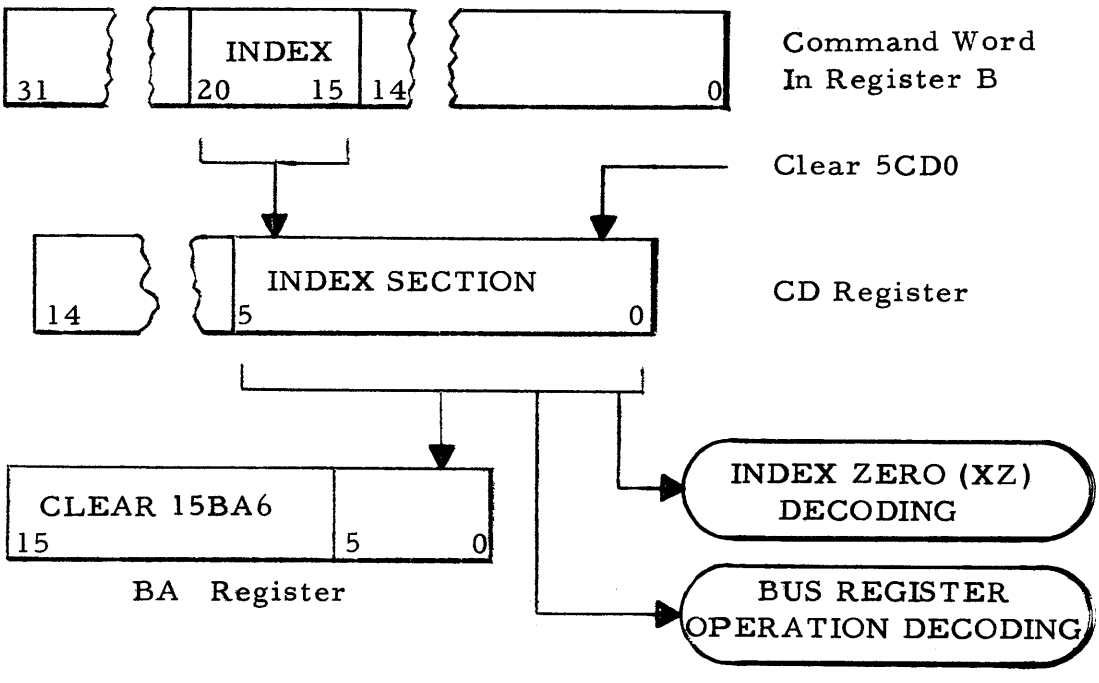


FIGURE 4. 2–2 Index Section Transfer and Modification Paths of CD Register, Normal Command Decoding

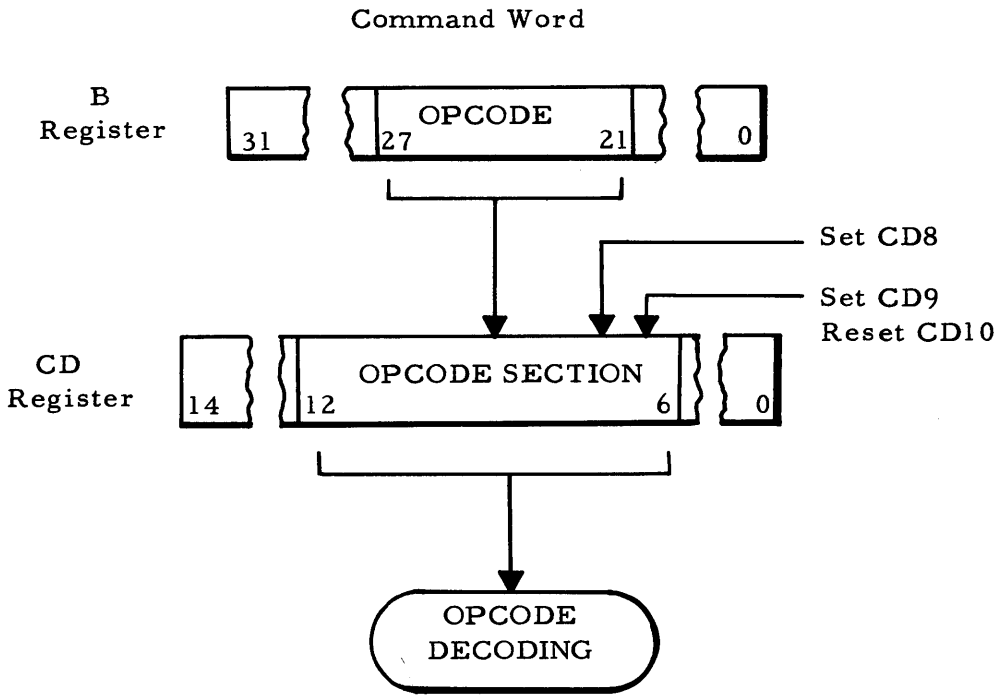


The most common path of Figure 4. 2–2 is the transfer path that takes the index from the command word in the B register via the CD register to the BA register. This is the path taken by the index field of all single word command words and by the first word of all two–word commands. If the command is other than one of the index or Bus register opcodes, the index field is the address of a location whose contents will be used to modify the operand X during operand assembly. If the index field is zero for these opcodes it is still sent to the BA register, but the XZ (Index Zero) signal goes high which prevents memory access of this address. If the command being processed is one of the index opcodes, the transfer path of the index field of the command word to the BA register is still the same. In this case, however, the number in the index field is used as an address which designates one of the index locations (memory locations 1 through 63) whose contents are to be modified. In these opcodes the index field is not used in the assembly of the operand X. If the command being processed is one of the Bus register opcodes, the contents of the index are still transferred from the CD register to the BA register as before, but the opcode decoding of the commands prevents a memory access. Instead, the information in the index section of the CD register is interpreted by the Bus register operation decoding logic which selects the register to be operated on.

4. 2–2 OPCODE SECTION OF CD REGISTER, NORMAL COMMAND DECODING Figure 4. 2–3 presents important register transfer and modification paths associated with the opcode section of the CD register during normal command decoding.

Bits 6 through 12 of the CD register, except for block input/output and repeat operations, always contain the opcode portion of the command word being processed. The information in this part of the CD register is then fed to the AND/OR opcode decoding logic. From this decoding logic, enabling signals are generated which cause the Central Processor

FIGURE 4. 2–3 Opcode Section Transfer and Modification Paths of CD Register, Normal Command Decoding



to take the correct paths and perform the necessary operations to process any one of the opcodes presented in Table 2. 2–1. The register coding for these 68 opcodes is presented in Table 4. 2–1. In later sections where it is necessary to follow the path of an opcode through the different sequencers that handle it, it will often be found that decoding from one or two of the bits of the CD register is all that is needed to specify the path to be followed. This occurs because, although all of the bits of the CD decoding may be needed to choose which of the many sequencers are to be used, once a particular sequencer is started, the coding can be simplified since only a few of the opcodes will be handled by any sequencer other than Master Control. Thus, it is only necessary

TABLE 4.2–1 CD Register Coding for Opcodes of Table 2.2–1

CD9	8	7	CD12 11 10 6	0 0 0 0	0 0 1 0	0 1 0 1	0 1 0 1	1 1 0 0	1 0 1 0	1 1 0 1	1 1 1 1
0	0	0	0	OCA 000	OCS 020	OAD 040	OSU 060	OAN 100	OSN 120	OAA 140	OSA 160
0	0	0	1	FOP 001	FOM 021	FSP 041	FGO 061	FSM 101	FLO 121	FSN 141	FUO 161
0	1	0	1	CLA 005	CLS 025	ADD 045	SUB 065	ADN 105	SUN 125	ADA 145	SUA 165
0	1	0	0	–	–	–	–	–	–	–	–
1	1	0	0	–	–	–	–	–	–	–	–
1	1	0	1	CAL 015	CCL 035	ADL 055	SUL 075	EXL 115	ECL 135	UNL 155	UCL 175
1	0	0	1	IOZ 011	ICZ 031	ISN 051	IUO 071	IEZ 111	IEC 131	IUZ 151	IUC 171
1	0	0	0	–	–	–	–	–	–	–	–
1	0	1	0	LXP 012	LXM 032	ERO 052	ERA 072	–	–	–	–
1	0	1	1	REP 013	BTR 033	DIV 053	STZ 073	STS 113	STI 133	STD 153	STL 173
1	1	1	1	TRA 017	TRE 037	RDV 057	MPY 077	TDC 117	SKP 137	TIC 157	TRM 177
1	1	1	0	XPT 016	XMT 036	LDR 056	EXR 076	–	–	–	–
0	1	1	0	AXT 006	SXT 026	–	–	–	–	–	–
0	1	1	1	–	–	–	–	–	–	–	–
0	0	1	1	–	–	–	–	–	–	–	–
0	0		0	ADX 002	SUX 022	–	–	–	–	–	–

to differentiate between opcodes handled by the sequencer whenever a particular sequencer is processing an opcode. For example, the KJ sequencer processes four opcodes: SKP, TRM, ERO and ERA. Although all 7 bits are needed to uniquely define each opcode in the CD register, only 3 bits (CD10, CD11, CD12) are necessary to distinguish the four opcodes within the KJ sequencer.

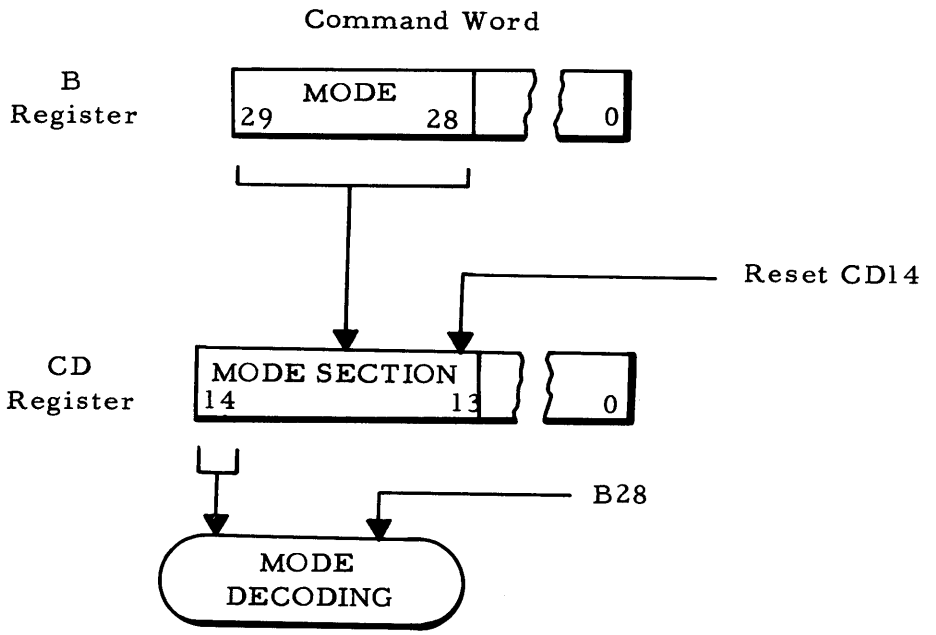
It will be noted that there are two inputs to the opcode section of the CD register that have not yet been discussed. One of these is the Set CD9 and Reset CD10 input. The sole purpose of this input is to permit modification of the coding of the index opcodes ADX and SUX to the code of the LXP opcode, and modification of the coding of the index test opcodes AXT and SXT to the XPT opcode code. This is done by the KC sequencer because after the access of the memory address contained in the I field, the ADX, SUX, AXT, and SXT commands are processed by the KC sequencer as if they were LXP or XPT opcodes.

The remaining input to the opcode section of the CD register is the Set CD8 signal. This signal, although it will be discussed both here and in Section 4.2–4, serves only one purpose and that is to insure that interrupt requests can be processed. The only way that interrupts can be processed is first that they be enabled and second that they pass through the Central Processor hardware–programmed portion of the Interrupt Service Routine. State KCB–5 of the KC sequencer (refer to Figure 9.5–1) is the initial step of the hardware–programmed portion of the Interrupt Service Routine and, as a result, its enabling terms must be satisfied or an interrupt request cannot be processed. The $\overline{OA}$ term is the critical gating term since it states that state KCB–5 of the KC sequencer cannot be entered if one of the address preparation, or the extract register I into OA (ERO) opcodes was the last opcode previously processed by the Central Processor. There is very sound reasoning behind this, since if an external interrupt request was

received and processed as soon as the opcode then being processed was finished, valuable information could be lost, thus invalidating the results of subsequent program operations. If one of the address preparation or the ERA opcodes had just previously been processed, it would mean that something vital to the correct solution of the program being processed had been placed in the OA register. State KCF–2 of the KC sequencer in the hardware–programmed portion of the Interrupt Service Routine, however, clears the OA register, thus if anything of value had been placed in OA by the preceding operation, it would be destroyed and lost to the program. Thus, the value of the $\overline{OA}$ gating term is evident. There are cases, however, when an interrupt request occurs internally during the processing of one of the address preparation and ERO opcodes due to a program error condition occurring. In these cases, it is desirable to process the interrupt request immediately since the results that would be placed in the OA register by these opcodes would be incorrect and not worth saving. State KCB–5 of the KC sequencer, however, will not let the interrupt be processed if an address preparation or the ERO opcode had just been processed. Thus, in this case, it becomes necessary to change the coding of the CD register so that these opcodes will no longer look like themselves to the opcode decoding circuitry, and therefore, allow the processing of the interrupt request. The modification of the CD register to allow internally generated interrupt requests to be processed for the above–mentioned opcodes is one of the uses of the Set CD8 term. There are other cases during bootstrapping of the machine or block input/output or repeat operations where the state of the CD register might indicate that one of the address preparation opcodes or the ERO opcode was being processed when such was not the case. In these cases, it is also necessary to set CD8 to allow processing of internal or external interrupt requests. This is discussed fully in Section 4. 2–4.

4.2–3 MODE SECTION OF CD REGISTER, NORMAL COMMAND DECODING Figure 4.2–4 shows important register transfer and modification paths associated with the mode section of the CD register during normal command decoding.

FIGURE 4.2–4 Mode Section Transfer and Modification Paths of CD Register, Normal Command Decoding



From Figure 4.2–4 it is seen that although both mode bits of the command word are sent to the CD register, only bit 14 of the CD register is sent to the mode decoding logic. In the assembly of the operand X, handled by the KC sequencer, bit B28, corresponding to CD13, is used directly for mode decoding.

If modes 0 or 2 are being processed, the A field of the command word is sent to the D register to be added to the contents of the OA register. If modes 1 or 3 are being processed, the A field of the command word is the address of memory location and it is sent directly to the BA

register for memory access. This is done before bits 29 and 28 of the B register are transferred to the CD register. Later in the operand assembly process, bit CD14 is decoded to determine the use of the operand X. (For clarification of this discussion, refer to Tables 2.3–1 and 2.3–2.)

The Reset CD14 term is generated by the KC sequencer on the final access of the operand assembly in modes 2 or 3. With CD14 reset, the CD register coding indicates modes 0 or 1 instead of modes 2 or 3. This prevents re–entry into the mode 2 or 3 paths of KC sequencer by subsequent operations performed during the processing of the command.

4.2–4 USE OF CD REGISTER IN BLOCK INPUT/OUTPUT, REPEAT, AND BOOTSTRAP OPERATIONS The block input/output and repeat operations are rather involved and they are discussed in detail in Chapters 12 and 13. For these two reasons, only that information necessary to an understanding of the function of the CD register in processing these two opcodes will be presented at this time.

The block input/output and repeat commands are made up of two Central Processor words. These are presented in Figure 4.2–5. From this figure, it is seen that although the two words of these commands are quite different in format, there is one point of similarity between them and that is that both words have opcode sections. The opcodes in both of the words of the command must be properly decoded to provide correct processing of the command. Another item that affects the CD register is that the address of the next command of the program must be temporarily stored somewhere other than its normal location, the CA register, during the processing of these opcodes. For reasons that will soon be apparent, the address of the next command of the program is temporarily stored in the AM register. The AM register, discussed in Section 4.3–5 is a composite register consisting of bits 0 through 7

and bits 13 and 14 of the CD register along with flip–flops AM8 through AM12.

FIGURE 4.2–5 Format of Block Input/Output and Repeat Commands

Repeat Command Format

First Word

FLAGS	MODE	OPCODE	I FIELD	A FIELD
31	29	27	20	14 0

Second Word

NOT USED	OPCODE*	NOT USED	BLOCK LENGTH
31	27	20	14 0

Block Input/Output Command Format

First Word

FLAGS	MODE	OPCODE	I FIELD	A FIELD
31	29	27	20	14 0

Second Word

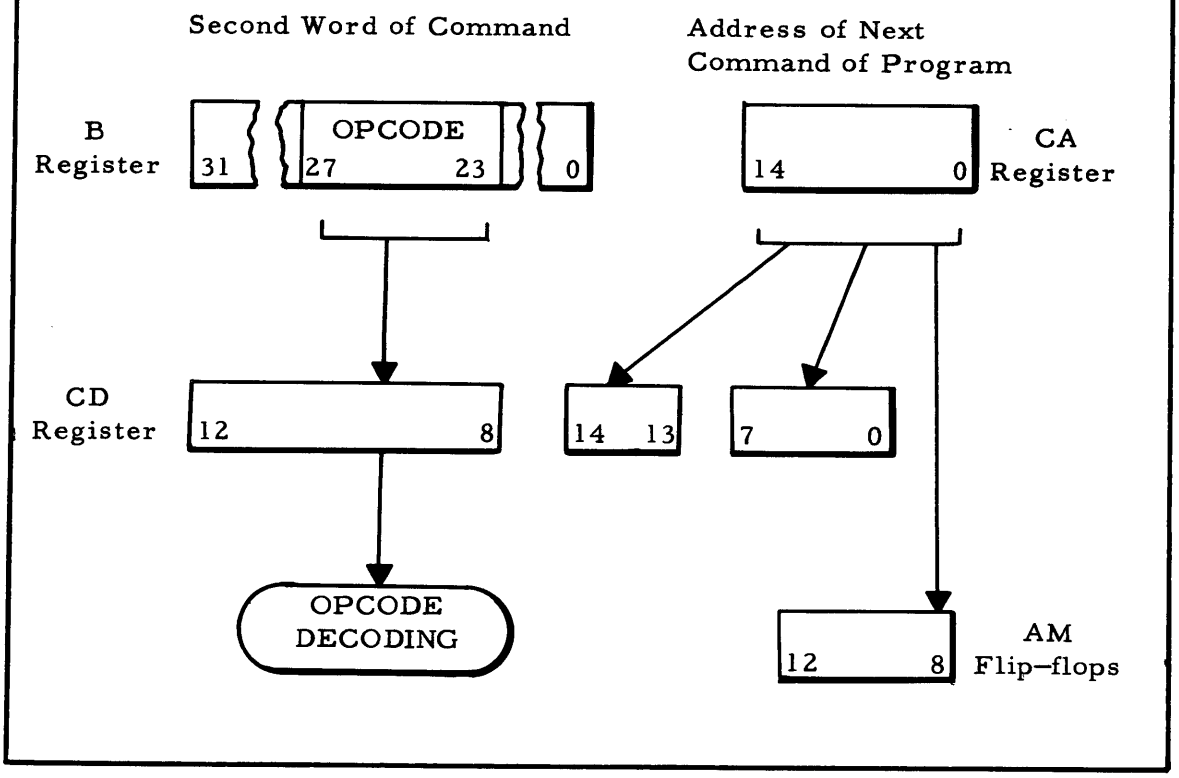
NOT USED	OPCODE**	STARTING INSTRUCTION	BLOCK LENGTH
31	27	22	14 0

* Of operation to be repeated.
** Of type of Block Input/Output operation.

When the first word of a block input/output or repeat command is brought out of memory, it is treated exactly as any other command word and bits 15 through 29 of the command are sent to the CD register. The opcode section of CD is then decoded which results in the KM sequencer being entered. Once the KM sequencer is entered, it will proceed through one of two paths, depending upon which of the two types of commands is being processed. It was shown in Section 4. 2–2 that an opcode can be recognized from fewer bits of the CD register once a sequencer is processing it. In the case of block input/output or repeat, once KM is entered, it is no longer necessary to define by CD decoding the opcode (i. e., block input/output or repeat) that is being processed since the path followed in the KM sequencer uniquely defines it. Therefore, when the second word of either of these commands is brought out of memory, the opcode of this word can be sent to the CD register for decoding without regard for the destruction of the previous contents. The opcode part of the second word of the command must be saved in the CD register until the end of the command processing, however, since this opcode determines which block input/output operation is to be performed, or which of the repeatable opcodes is to be repeated. Once again, because only certain sequencers will be used by these commands, it is possible to define the opcode being repeated or the block input/output operation being performed with fewer than the 7 bits normally present in the opcode section of the CD register. In the case of repeat, only 5 bits are needed, and for block input/output, only 3 bits are needed. Since the mode and index sections of the CD register are not used for the second word of the command, they, along with the least significant 2 bits that are not needed in the opcode section of the CD register, make a total of 10 bits of the CD register that are not necessary during the processing of these commands. These 10 bits, along with the 5 additional bits supplied by the AM8 through AM12 flip–flops, are used to store the address of the next command of the program while these commands are being processed.

The lopping off of the least 2 bits of the opcode section creates additional problems, however. Although it is true that while only a maximum of 5 bits of the CD register are needed for opcode decoding during processing of the block input/output or repeat commands, when these commands are finished and control has been returned to the KC sequencer to supervise the processing of the next opcode of the program, these 5 most significant bits of the opcode section of the CD register are not sufficient to insure proper operation. It is very possible that these commands were terminated because an error condition or a flagged word occurred during their processing. In this case, an interrupt request is generated and it is necessary to proceed next through the hardware–programmed portion of the Interrupt Service Routine. In Section 4. 2–2 the hardware–programmed portion of the Interrupt Service Routine was discussed, and it was found that this portion of the machine logic cannot be entered if the command previously processed was one of the operand assembly opcodes or the ERO opcode. None of the block input/output or repeat operations belong to this group of opcodes. However, since the least 2 bits of the opcode section of the CD register were last used as storage for the address of the next command location, the contents of these 2 bits are unreliable. Thus, the contents of the opcode section of the CD register under these conditions may appear to be one of the operand assembly opcodes or the ERO opcode. If this should happen, an interrupt request that was received during the command processing could not be processed immediately at the finish of the block input/output or repeat operation. For this reason the CD8 bit of the CD register is always set at the finish of a block input/output or repeat operation. With the CD8 bit set, it is impossible for the CD register coding to be decoded as one of the operand assembly opcodes or the ERO opcode barring a component failure in the decoding logic. Figure 4. 2–6 shows register transfer and modification paths associated with the CD register for the second word of block input/output or repeat commands.

FIGURE 4.2–6 Transfer and Modification Paths of CD Register for Second Word of Block Input/Output or Repeat Commands

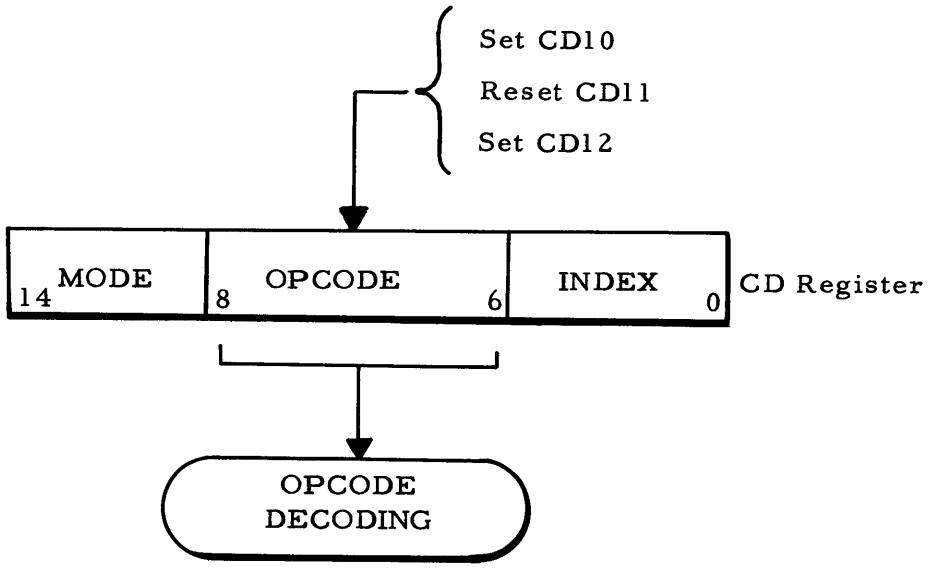


There is one remaining operation that affects the CD register still to be discussed and that is the Bootstrap operation. Transfer paths for this operation are presented in Figure 4.2–7.

The bootstrap operation is used to facilitate initial loading of the Central Processor's memory. It is possible to load the entire memory by bootstrap operation, but usually only a short routine, in turn, then supervises the loading of commands and data of the program that is to be processed. Since this bootstrap operation is an externally originated input operation, it is necessary to provide the opcode section of the CD register with information that will allow the correct paths of the sequencers involved to be traversed. The Set CD10, Reset CD11, Set CD12 signals are all that are necessary to fix the CD coding such that

the Central Processor will allow this modified input operation to proceed. A detailed discussion of the bootstrap operation is presented in Chapter 12.

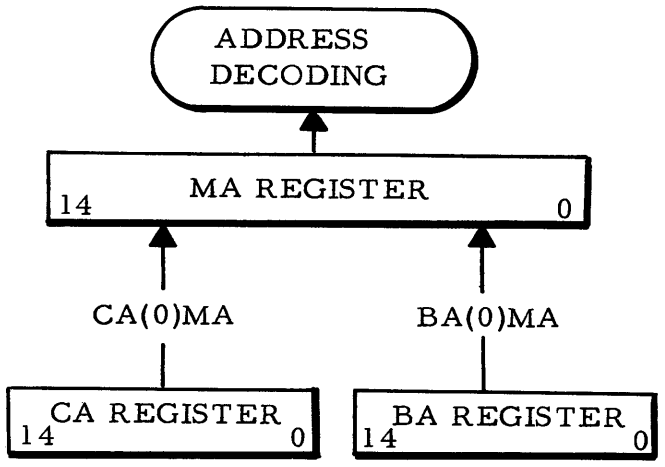
FIGURE 4. 2–7 Transfer Paths of CD Register for Bootstrap Operation



SECTION 4.3 – ADDRESS REGISTERS

4.3–1 THE MEMORY ADDRESS REGISTER, MA The MA register is a 15–bit register whose sole purpose is to hold the address of the memory location that is being accessed or intended to be accessed in the case of a memory overflow condition. There are eight different core memory modules available with a G–20 System, each of which has a storage capacity of 4,096 words. Therefore, the 3 highest order bits of the MA register are used to determine which memory module to select and the remaining 12 bits of the MA register are used to define which of the 4,096 words of that memory module is to be accessed. This determination of memory and location is provided by address decoding logic which is fed directly by the MA register. Figure 4.3–1 shows transfer paths associated with the MA register.

FIGURE 4.3–1 Transfer Paths of MA Register



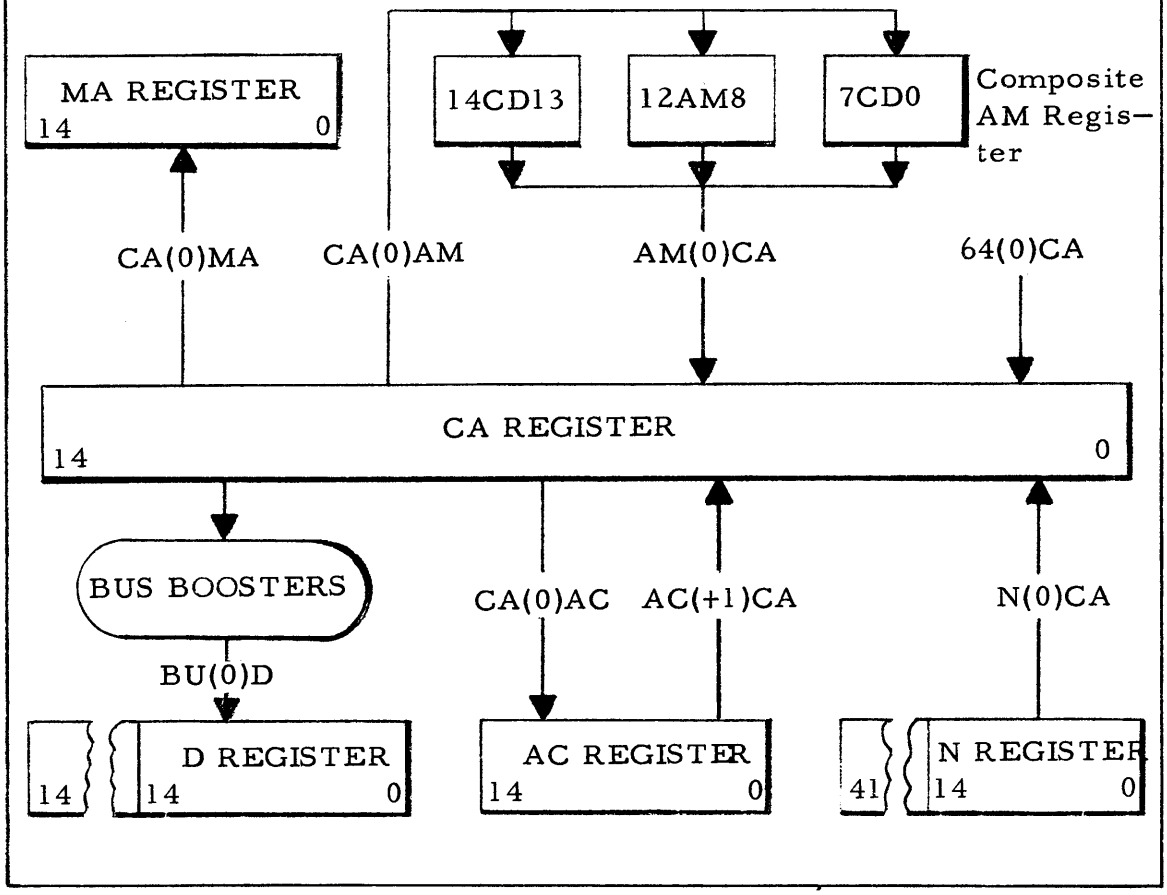
When the contents of the CA register are sent to the MA register, the location in memory that is accessed contains a command word if a block input/output or repeat command is not being processed. The address sent from CA to MA during block input/output is the storage location for data or commands in transit. During repeat operations the address sent from CA to MA is merely a data location.

When the contents of the BA register are sent to the MA register, the accessed location in memory contains a number that will be used in assembling the operand X of a command word if a store command or certain of the index and index test opcodes are not being processed. In the case of a store command, information transferred from BA to MA is the address of the memory location where the desired data is to be stored. During processing of certain of the index and index test opcodes, the contents of the index location are combined with the operand X and then stored in the same index location. In this case, the information transferred from BA to MA contains the address of the index location that is to be added to X.

4. 3–2 THE COMMAND ADDRESS (OR NEXT COMMAND) REGISTER, CA The Command Address register is one of several registers that has been given different titles by the Engineering and the Programming Groups. This is truly unfortunate since the reader will be referencing material from both groups. For this reason, both the Engineering and the Programming titles associated with a register are presented in the section headings where they are discussed. The CA register is a 15–bit register that normally holds the next command of the program being processed. There is one slight deviation from this rule which will be discussed later. Figure 4. 3–2 presents all transfer and modification paths associated with the CA register.

Whenever the Central Processor is turned on, it is first necessary to

FIGURE 4. 3–2 Transfer and Modification Paths of Register CA



bootstrap an initial program which will supervise the loading into memory of the program that is to be processed. This bootstrap program is located in successive memory location beginning with location 64_{10}. It was shown in Section 4. 2–4 how the CD register is set up to allow the input of the bootstrap program. From Figure 4. 3–2, the means of selecting the desired starting location becomes evident. This is one function of the 64(0)CA input signal to the CA register. Also, concerning bootstrap operations, once the bootstrap program has been read in, it is necessary to provide a means of returning to the starting command of the program (contents of location 65_{10}) so it can be processed. Using this same 64(0)CA signal and the incrementing

function of the AC register, this is achieved upon completion of reading in the bootstrap program. The final use of the 64(0)CA signal is to access location 64_{10} for use as the storage location of the address of the next command in the program when an interrupt request is being serviced. The Interrupt Service Routine (ISR) stores the location of the next command of the program in 64_{10} and then begins processing the programmed part of the ISR beginning with location 65_{10}.

When execution of a program begins, the addresses of the commands to be processed are transferred from the CA to the MA register and the contents of the CA register are incremented to the next address of the program by the transfer paths to and from the AC register. In Section 4. 2–4 it was shown that, in block input/output and repeat operations, the address of the next command of the program is stored in the bits of the composite AM register. Bits 0 through 14 of the N register, which at that time holds the starting location of the block input/output or repeat operation, are then sent to the CA register. This address is then successively sent to the MA register and incremented by the AC register until the block input/output or repeat operation is completed at which time the address of the next command of the program is retrieved from the AM register and processing of the program commands is continued. The N(0)CA path is also used to load the CA register during the processing of the transfer of control opcodes listed in Table 2. 2–1.

There are several cases for which it is desirable to read out the contents of the CA register into the Arithmetic Unit for further processing or for storage into another register or into memory. This is done by using the CA register as a Bus register. The Bus registers are discussed in Section 4. 4.

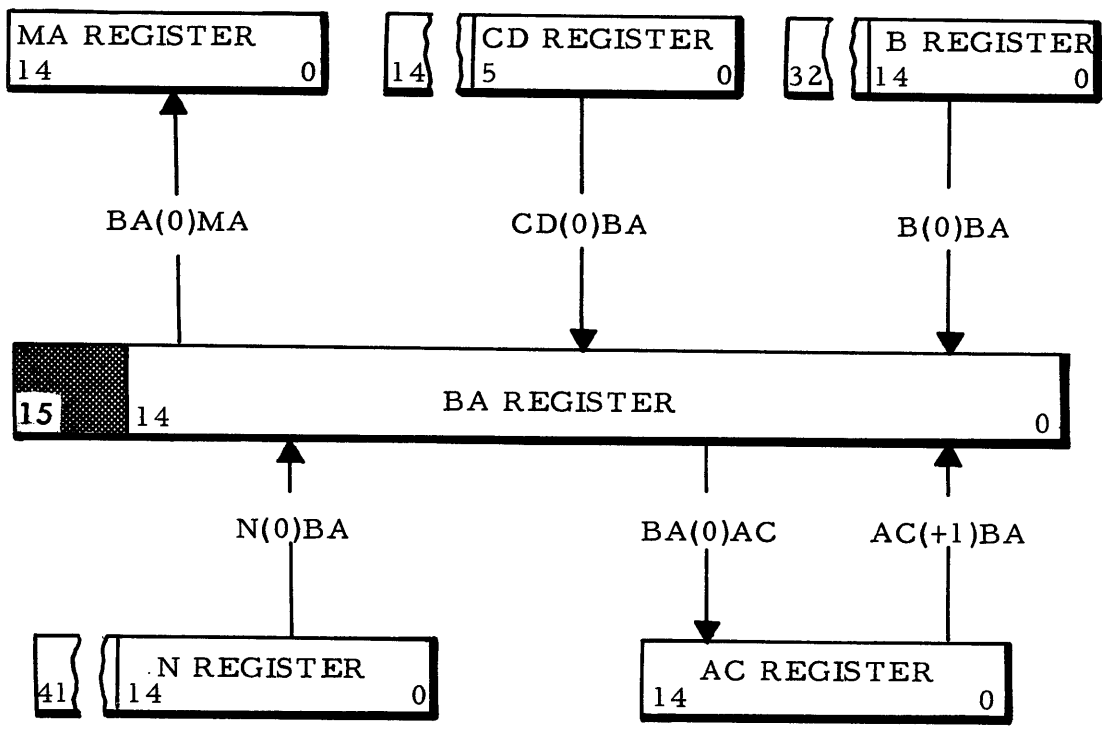
4. 3–3 THE BUFFER ADDRESS REGISTER, BA The BA register is

slightly different than the other address registers inasmuch as it consists of 16 bits rather than the regular 15 bits. The sixteenth bit (BA15) has a special purpose utilized only during block input/output or repeat operations. For this reason, discussions of all other operations that affect the BA register will ignore the presence of bit BA15.

One of the two uses of the BA register is to provide a means of loading the MA register and, thus, accessing memory without disturbing the CA register which holds the address of the next command of the program. The BA register is used in this manner during operand assembly operations and during the processing of store commands. In Table 2.3–2 it was shown that, during the assembling of the operand X, the A field of the command word can be either an operand or an address of an operand, depending upon the operating mode.

If the (A field) of a command word held in the B register is an operand, it is sent directly to the Arithmetic Unit to be added to the contents of the OA register. If, however, it is an address of an operand, it is necessary to first access the location indicated by the A field of the command word before an addition can occur. In this case, the 15 least significant bits of the command word in the B register are sent directly to the BA register for further processing. Also, during operand assembly operations, the memory location indicated by the contents of the I field of the command word must be accessed unless (I) = 0 or one of the index, index test, or bus register commands is being processed. (This is discussed in Section 2.3.) In this case, the contents of the I field of the command word, previously sent to the CD register, are transferred to the BA register for further processing. After either the transfer of the A field or the I field to the BA register, the BA register transfers this address to the MA register from which memory is accessed. These transfer paths are shown in Figure 4.3–3.

FIGURE 4. 3–3 Transfer and Modification Paths of BA Register



The second use of the BA register is as a counter for block input/output or repeat operations. In Figure 4. 2–5 it was shown that the block length of these operations is originally held in the second word of these commands. The block length is sent from the B to the S register and then transferred in 1's complemented form to the N register. The 1's complement of the block length is then transferred to the BA register. The BA register, using the incrementing facility provided by the AC register transfer paths, proceeds to increment the complemented block length each time a complete word is transmitted or received, or each time the command that is to be repeated has been processed. This incrementing process would result in an actual decrement of the block length each time it is done. When the number of repetitions

indicated by the block length had been accomplished, the 1's complement of the block length would be all 1's (7's in octal notation). This is very good, except that it means that a 15 input AND-gate would have to be used to decode when the end of the block is reached. However, since at end of block all bits of the BA register are 1's, if one more 1 was added to this register (i.e., one more increment operation) there would be a carry out and the 15 bits that had been 1's would be changed to 0's. This would then be the 2's complement of the block length and it would mean that only one output from the register could signal when end of block has occurred. This is the way it occurs in the Central Processor, except that the extra incrementing process is done prior to the first operation. This means that the BA register is carrying the 2's complement of the block length during processing of block input/output or repeat commands. When the end of block is reached, the ZBA signal is generated which indicates that bits 0 through 14 of the BA register are 1's and that on the next clock pulse (i.e., the next C1 pulse) the 2's complement of the end of block will be generated. Bit BA15 is used exclusively to provide storage for the carry out bit of the 2's complement of the end of block. This discussion is illustrated in Table 4.3-1. Octal notation is used to illustrate the complement block length to simplify the example. To be absolutely correct, only 1's and 0's should be used.

TABLE 4. 3–1 Different Representations of Block Length

Current Block Length (Octal Notation)	1's Complement of Block Length (Octal Notation)	2's Complement of Block Length (Octal Notation)
00002	77775	77776
00001	77776	77777
00000*	77777*	100000*

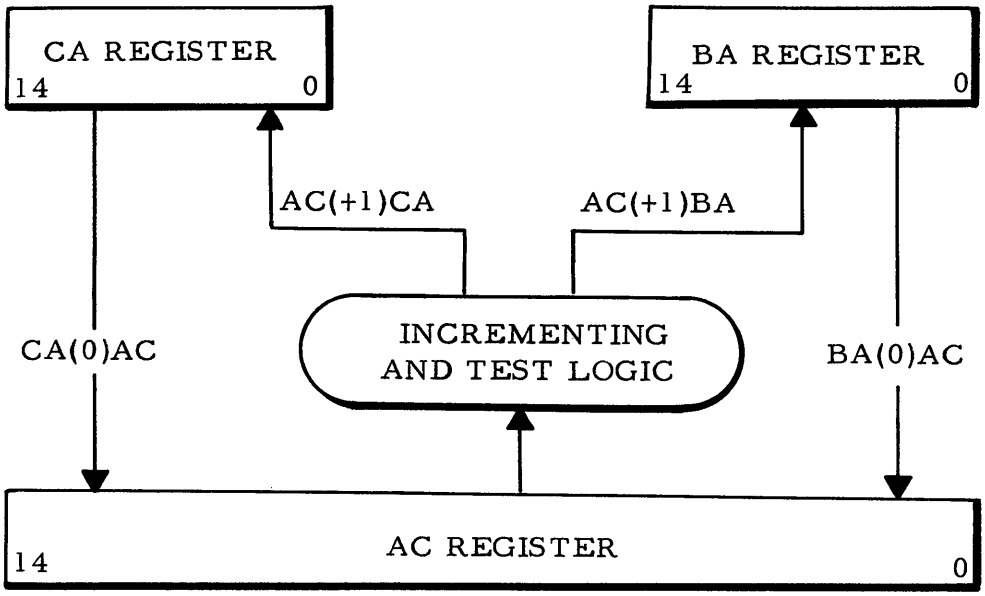
*End of Block

4. 3–4 THE ADDRESS COUNTER REGISTER, AC The Address Counter register is a 15–bit register whose main function is to provide storage locations in the incrementing logic associated with the CA and BA registers. The contents of these registers (CA or BA) are sent directly to the AC register and from the AC register to the incrementing logic where the contents of the CA or BA register, whichever the case may be, are increased by one and fed back into the CA or BA registers. The AC register and the transfer paths associated with it are presented in Figure 4. 3–4.

If the AC register and its transfer paths are used to increment the CA register, it means that the command sequence of a program or the address of a block input/output or repeat operation has been incremented by one. When the AC register and its transfer paths increment the BA register during block input/output or repeat operations, however, it indicates that the tally of the block length of the input/output or repeated operations has been decremented by one. During the processing of store commands, the AC and BA registers interact to increment the contents of the BA register by one in case a double precision store

command is being processed. With BA incremented by one in this situation, the second memory location of a double precision word is accessed on the next memory operation.

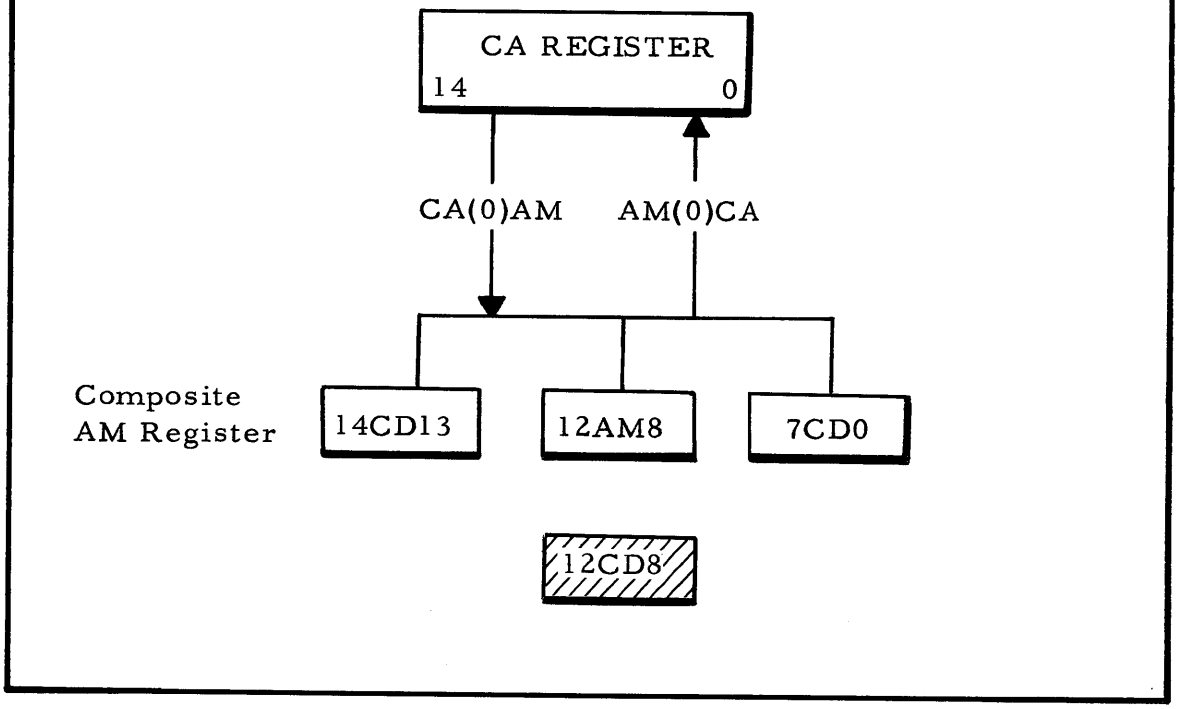
FIGURE 4.3–4 Transfer Paths of AC Register



In Figure 3.2–2 and in Figure 4.3–4 it is seen that there is a TEST function associated with the AC register. This TEST is used only during block input/output or repeat operations to give advance notice to the Central Processor that the end of block has occurred. This TEST function actually consists of the signal ZBA (Zero BA register) which is high only when bits 0 through 14 of the AC register are 1's. On the next +1 increment from the AC register (i.e., on the next C1 clock pulse) the 15 least significant bits of the BA register will be set to zero and the ZBA term will set BA15. BA15, it has been shown, is used to designate end of block for block input/output or repeat operations.

4. 3–5 THE ADDRESS MEMORY REGISTER, AM In Section 4. 2–4 it was shown that the AM register is, in reality, a composite register consisting of bits 14CD13 and 7CD0 along with the flip–flops 12AM8. It was also mentioned then that the sole purpose of this register is to provide temporary storage (i. e., memory) for the address of the next command of a program during the processing of block input/output or repeat operations. The AM register and transfer paths associated with it are presented in Figure 4. 3–5.

FIGURE 4. 3–5 Transfer Paths of Composite AM Register



SECTION 4.4 – BUS REGISTERS

There are a group of four control registers (CA, U, J, and H) containing quite dissimilar types of information which at one time or another share a common need to have their contents read out into the D register, either for further processing, or for storage in another register or in memory. These four registers along with the PE register form a loosely related group classified as the Bus registers. The PE register is included in this group due to the fact that the only opcode that can operate on the PE register (LDR) provides a function which is at times useful to three of the other four Bus registers. This opcode is grouped as one of the Bus register opcodes created to provide a means of implementing the desired Bus operations. The CA register is not affected by the two load register commands of the Bus register opcodes (LDR or EXR); however, the CA register can be loaded through use of the transfer of control opcodes. The Bus registers are listed in Table 4.4–1. The Bus register opcodes that operate on these registers when they are used as Bus registers are listed in Table 2.2–1.

It will be noted that in Table 4.4–1 the index, I field, of the command word is referenced. Also, in Table 2.2–1 the notation (Reg. I) means "the contents of the Bus register indicated by the I field of the command word". This is done because, in all operations that affect Bus registers, the contents of the I field are decoded to determine which of the Bus registers are to be operated on.

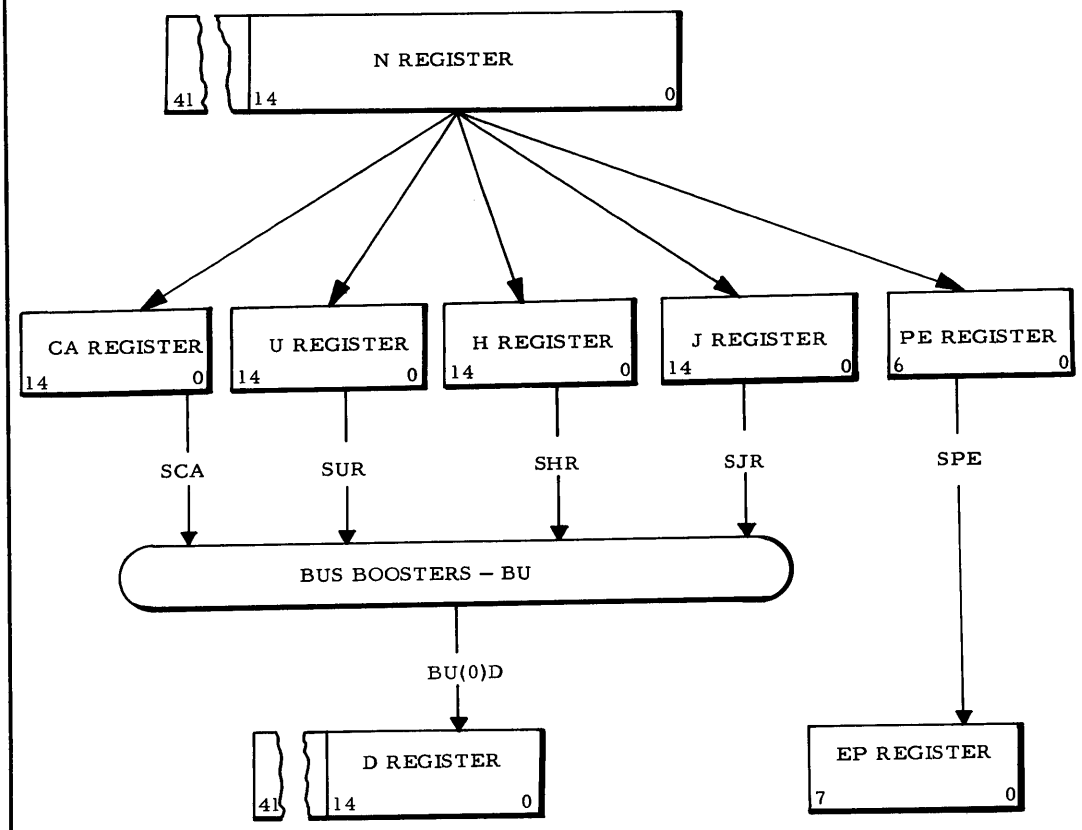
Figure 4.4–1 shows the basic transfer paths associated with these registers when they are used as Bus registers. These are broken down into more detail in the separate discussions of the various registers.

It will be noted from Figure 4.4–1 that, while four of the Bus registers

TABLE 4.4–1 The Bus Registers

Name	Engineer's Designation	Programmer's Designation	Designation in I Field of Command Word
Command Address or Next Command	CA	NC	00
Control and Enable	U	CE	01
Line Response	H	LRE	02
Interrupt Requests	J	IR	03
Pickapoint Exponent	PE	PE	04

FIGURE 4.4–1 Basic Transfer Paths of Bus Register Operations



feed into the Bus Boosters, there is only one path from the boosters to the D register. Thus, the Bus Boosters, which are in reality only a group of inverters, provide a selective switching function between the Bus registers and the D register. This allows one group of inverters to provide any necessary increased fan–out capabilities of the different register outputs without necessitating separate booster inverters on the outputs of all of the registers. The terms shown directly above the Bus Boosters are the enabling terms that determine which register is to use the boosters at any particular time.

4. 4–1 THE COMMAND ADDRESS REGISTER, CA Figure 4. 4–1 presents enough detail for the following discussion of the CA register as a Bus register. It will be noted that, in Figure 4. 4–1, an input is indicated from the N to the CA register during Bus operations. This transfer does not occur during the processing of the Bus register opcodes. Rather, this is the result of the transfer of control opcodes of which only the TRM opcode (Transfer and Mark opcode) is discussed in conjunction with Bus register operations since it not only loads the CA register with the contents of the 15 least significant bits of the N register, but also transfers the previous contents of the CA register via the Bus Boosters and various Arithmetic Unit registers to temporary storage in the memory location designated by the operand X of the TRM command. The initial processing of an interrupt request is handled similarly to the processing of the TRM opcode. In fact this is often referred to as a "machine–programmed transfer and mark". During the processing of interrupt requests, however, the contents of the CA register are sent via the Bus Boosters and Arithmetic Unit registers to memory location 64 where they are temporarily stored. The processing of TRM commands and enabled interrupts is discussed in more detail in Section 13. 4.

The other times that the (CA) are sent to the Bus Boosters and then

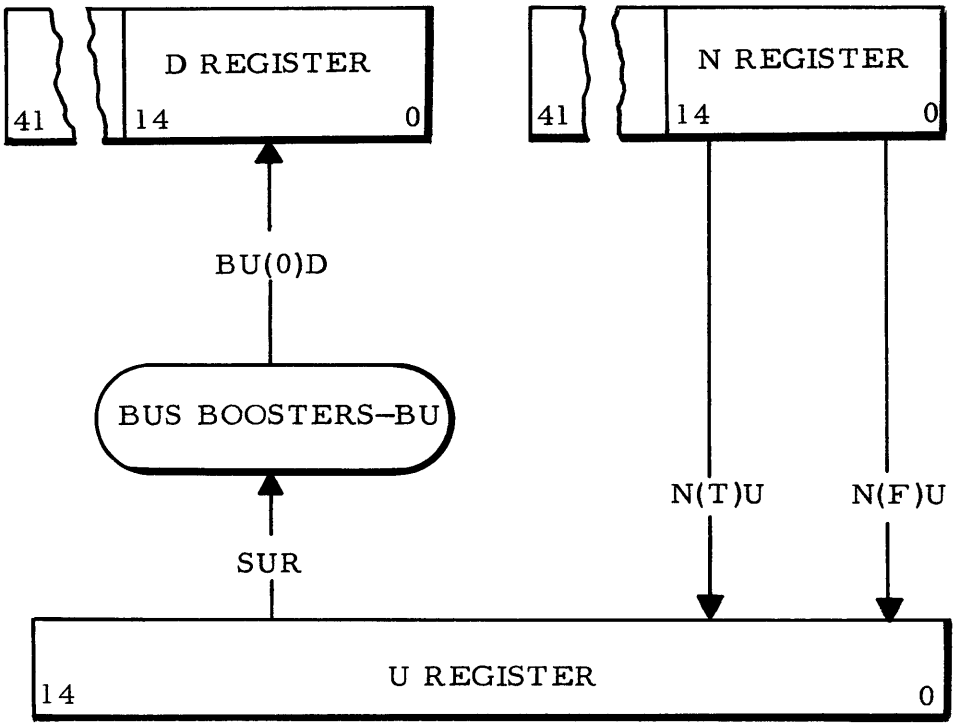
BU(0)D are during processing of the ERO and ERA opcodes of the Bus register opcode group and at the completion of block input/output and repeat operations. In the case of the ERO and ERA opcodes the (CA) are transferred to the Arithmetic Unit to be ANDed with the operand X. In the case of block input/output or repeat operations, (CA) contain the address of the next location in memory to be processed or the address of the next location plus 1, depending upon the operation being performed. The (CA) are sent via the Bus Boosters to the Arithmetic Unit on their way to the Accumulator for temporary storage. Reasons for storing this information in a location readily accessible to the programmer are discussed in detail in Chapters 12 and 13.

SECTION 4. 4–2 CONTROL AND ENABLE REGISTER, U The U register is a 15–bit register whose primary purpose is to provide a means of controlling which interrupt requests will be recognized and processed. It achieves this by serving as a source of enabling terms that first can determine if any interrupt requests at all are to be recognized and processed, and, second, that can selectively determine the particular interrupt requests that will be recognized and processed. The other purpose of this register is to enable the Central Processor to transmit interrupts to peripheral units. This material is discussed in detail in Section 2. 4 of the General Reference Manual and shown in Figure 4. 4–3 and Table 4. 4–2.

The purpose of this section is to discuss the uses of the U register as a Bus register. Figure 4. 4–2 presents the Bus register transfer paths associated with the U register.

From Figure 4. 3–2 it is seen that there are two transfer paths from the N register to the U register. These transfer paths implement the loading of the U register made possible by the two bus register opcodes

FIGURE 4. 4–2 Bus Register Transfer Paths of Register U



LDR and EXR. From Table 2. 2–1 it is seen that the operation of the EXR opcode is (Reg. I) $\wedge$ X $\rightarrow$ (Reg. I). The N(F)U transfer is sufficient to accomplish this entire operation. The example below illustrates how this is done. It is seen that the results of the two methods are identical.

number in U register	=	0 1 0 1 0 1 1 0 1 0 0 1 1 0 1
value of operand X in N register	=	1 1 1 0 1 0 1 1 0 0 1 0 0 1 1
ANDed result		0 1 0 0 0 0 1 0 0 0 0 0 0 0 1
number in U register	=	0 1 0 1 0 1 1 0 1 0 0 1 1 0 1
transfer N(F)U	=	0 0 0 0 0 0
result in U register		0 1 0 0 0 0 1 0 0 0 0 0 0 0 1

To accomplish the loading of the U register by the LDR opcode, both the N(F)U and the N(T)U transfer paths are used. This results in a transfer identically equal to a N(0)U transfer, which is desired for this opcode.

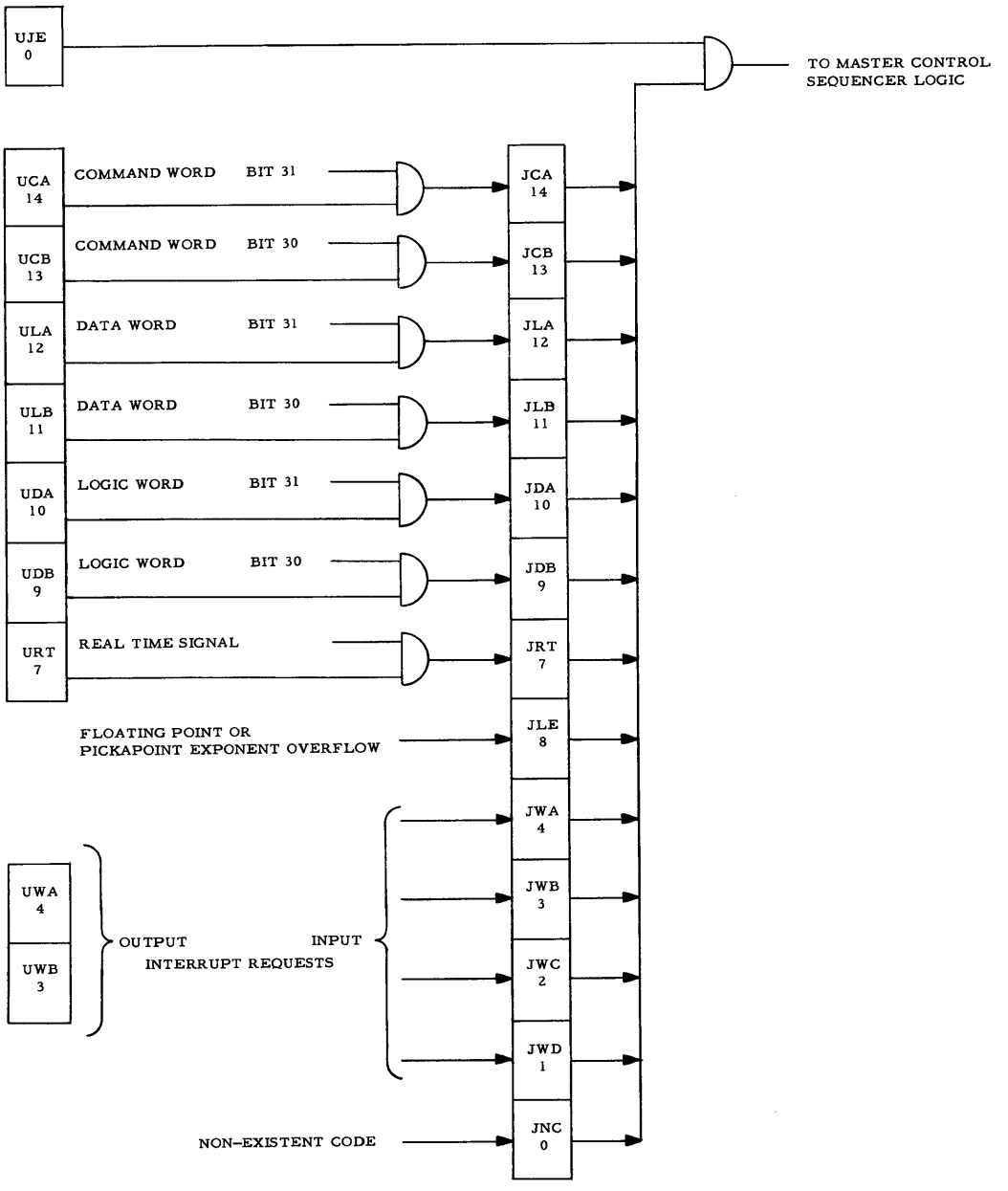
TABLE 4. 4–2 The Contents of Registers U and J

Bit Position		Register U		Register J
14	UCA	command flag 31	JCA	command flag 31
13	UCB	command flag 30	JCB	command flag 30
12	ULA	logic flag 31	JLA	logic flag 31
11	ULB	logic flag 30	JLB	logic flag 30
10	UDA	data flag 31	JDA	data flag 31
9	UDB	data flag 30	JDB	data flag 30
8	UPE	pickapoint mode	JLE	exponent overflow
7	URT	real–time reference enable	JRT	real–time reference interrupt
6		reserved		reserved
5	URB	audio tone		reserved
4	UWA	interrupt transmit	JWA	in–out interrupt
3	UWB	interrupt transmit	UWB	in–out interrupt
2		reserved	JWC	in–out interrupt
1		reserved	JWD	in–out interrupt
0	UJE	jump enable	JNC	non–existent code

The transfer path from the U register to the Bus Boosters is enabled by the SUR (Select the U Register) signal when the ERO and ERA opcodes of the bus register opcode group are being processed. This allows the programmer to examine the (U), or to temporarily store the

(U) during the processing of multiple programs.

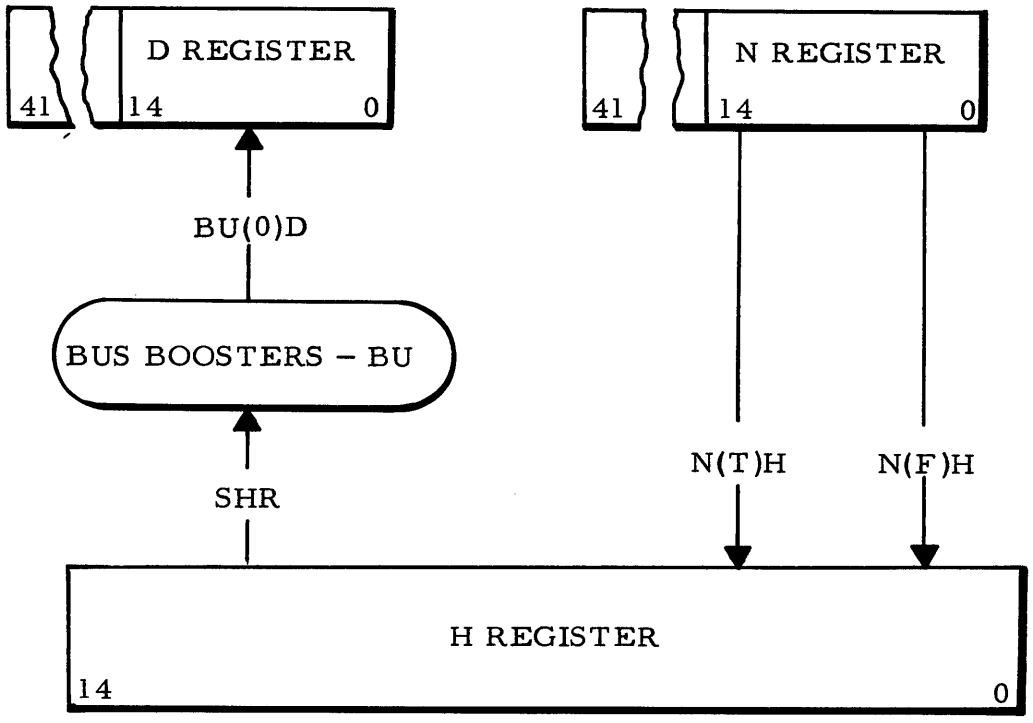
FIGURE 4. 4–3 Simplified Interrupt Request Logic



4. 4–3 LINE RESPONSE REGISTER, H The Line Response register is not a control register, but rather is used exclusively to temporarily

store information generated when error conditions occur and are recognized as such by the Central Processor during input/output operations. See Section 12. 1, especially Table 12. 1–3. In this section only Bus–register–type operations on the H register will be discussed. Figure 4. 4–4 presents the transfer paths associated with the H register when it is used as a Bus register.

FIGURE 4. 4–4 Bus Register Transfer Paths of Register H



The loading paths N(T)H and N(F)H are used to accomplish the two Bus register loading operations, opcodes LDR and EXR, in the same manner as that discussed in Section 4. 4–2. These opcodes are available to the programmer for use on the H register, but at the present are of little value. The transfer path from the H register to the Bus

Boosters, however, is very important.

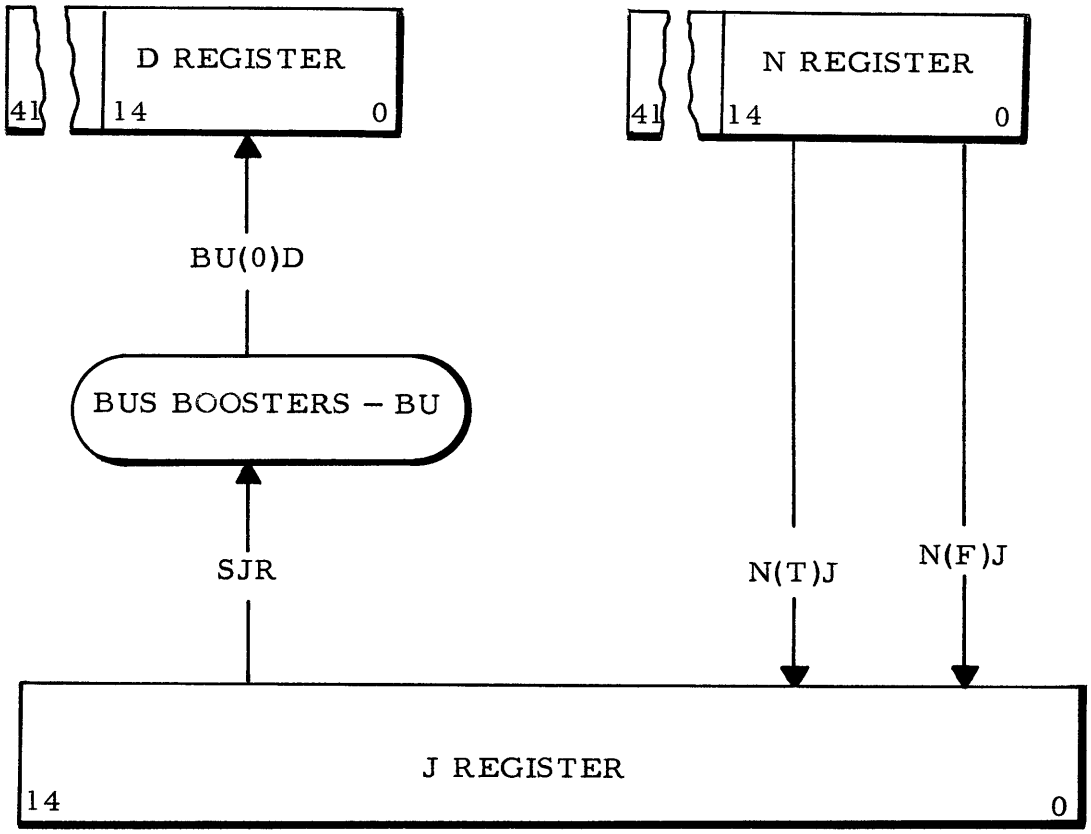
During the processing of input/output opcodes, if error conditions should occur, pertinent information concerning the cause and an indication of the position in the operation where the error occurred are temporarily stored in the H register. The Bus register opcodes ERO and ERA are used to place this information into the Arithmetic Unit of the Central Processor for further use by the programmer.

4. 4–4 THE INTERRUPT REQUEST REGISTER, J The J register is a 15–bit register used exclusively to store interrupt requests to the Central Processor. This function of the J register is discussed in detail in Section 2. 4 of the General Reference Manual and illustrated in Figure 4. 4–3 and Table 4. 3–2. Therefore, only the use of the J register as a Bus register will be discussed at this time. Figure 4. 4–5 gives the transfer paths associated with the J register when it is used as a Bus register.

The loading paths N(T)J and N(F)J are used during the processing of the Bus register loading opcodes, LDR and EXR, in the same manner as that discussed in Section 4. 4–2. These opcodes are available for use on the J register, but at the present are of little value. The transfer path from the J register to the D register (via the Bus Boosters) allows the condition of the various interrupt request flip–flops of the J register to be sent to the Arithmetic Unit of the Central Processor where they are more easily available for use by the programmer.

4. 4–5 THE PICKAPOINT REGISTER, PE The pickapoint mode format was discussed in Section 2. 1–2 and in this discussion it was noted that the exponent and its sign are stored in a separate register, the PE register, during pickapoint operations. When pickapoint operations are being processed, and if the UPE flip–flop is set (bit 8 of

FIGURE 4.4–5 Bus Register Transfer Paths of Register J



the U register set to 1), then any numbers, other than double precision, accessed from memory are considered to be multiplied by the power of eight indicated in the PE register. (The contents of PE are transferred to the EP register as the current operand exponent.) Also, the exponents of the results of all computations are adjusted to the value of the contents of the PE register prior to storage into memory. The PE register consists of 7 bits of which the 6 least significant bits indicate the value of the exponent. The most significant bit, PE6, indicates the sign associated with the pickapoint exponent (0 = plus, 1 = minus). A negative zero exponent in the PE register will cause

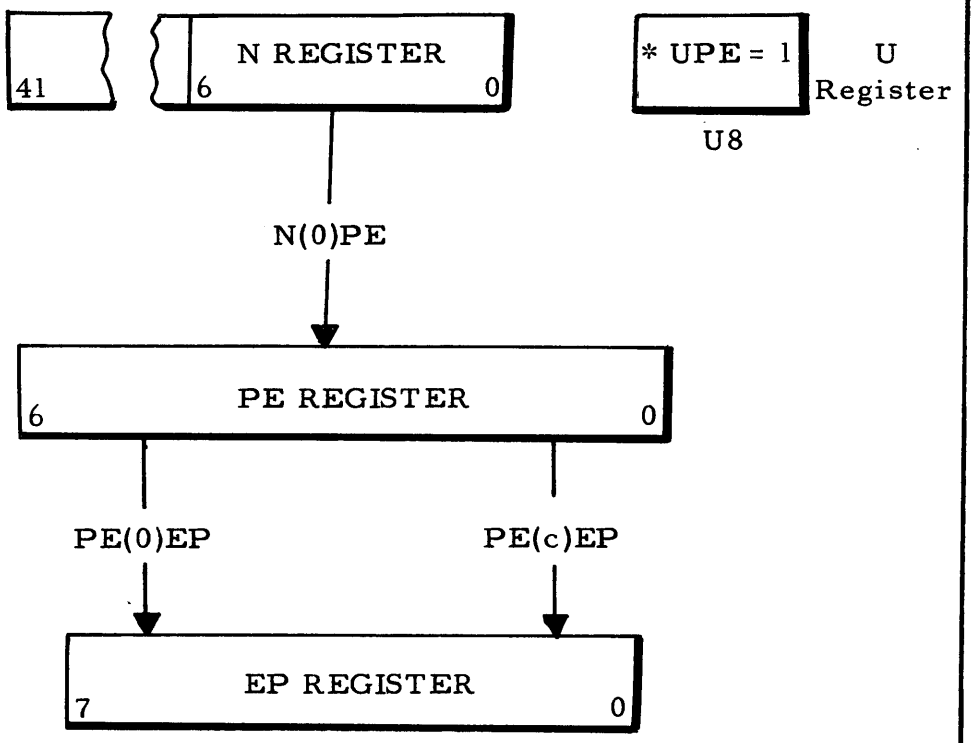
errors and is therefore forbidden. If bit B29 of an accessed word is high, indicating a double precision word, then the UPE flip–flop is ignored and computations proceed in floating point. Also, if the LP (Logic Product) flip–flop is high, computations will not be processed in pickapoint mode.

Before pickapoint operations can occur, the UPE flip–flop must be set to 1 and the PE register must be loaded to the desired value (not necessarily in that sequence). The path N(0)PE loads the register to the desired value. This is the only way that the contents of the PE register can be modified, and then only during a Bus register operation (the LDR opcode).

The desired exponent value is part of the operand X of the LDR command word and was put into the N register during the processing of the LDR opcode.

In Figure 4. 4–6 it is seen that there are two transfer paths from the PE register to the exponent register, EP. These transfers can only occur if the UPE flip–flop is set. The PE(0)EP transfer is caused by bit PE6 being low, which indicates a positive exponent in PE. The PE(C)EP transfer occurs when bit PE6 is high, indicating a negative exponent in PE. It is necessary to send the complement of a negative exponent to the EP register since the exponent registers are designed to handle negative exponents in this form. This is discussed in more detail in Section 5. 4.

FIGURE 4. 4–6 Transfer Paths of PE Register



* Condition necessary and sufficient to permit pickapoint operations.

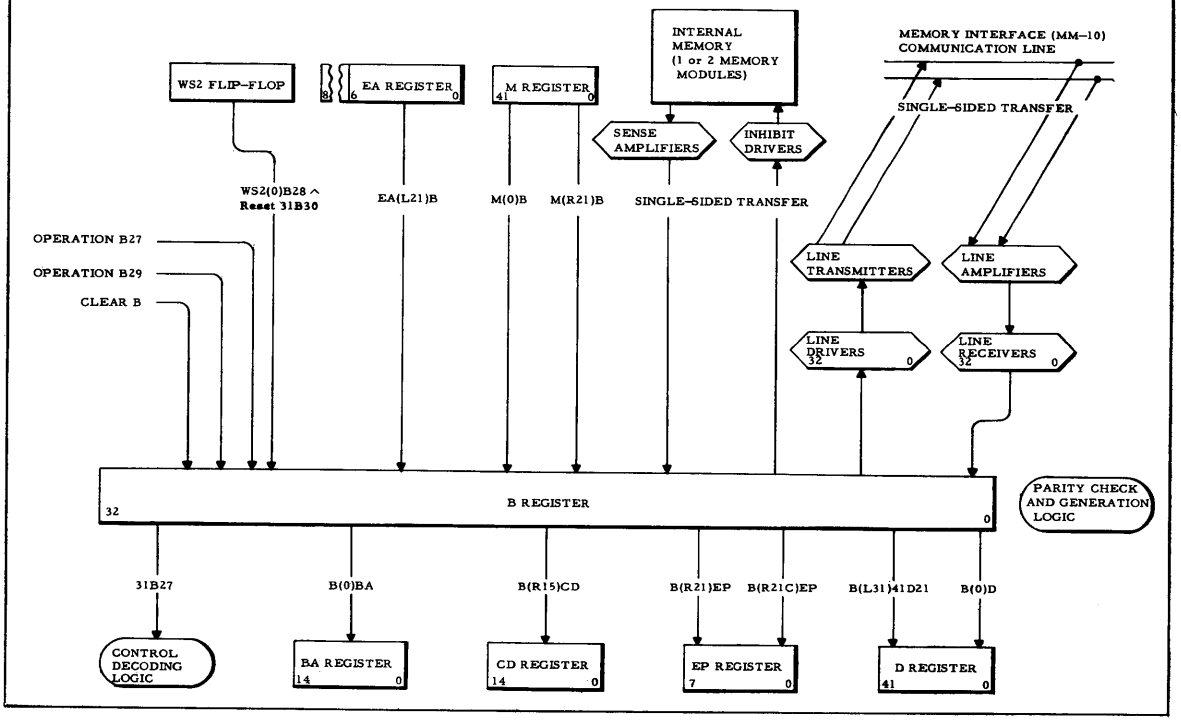
CHAPTER 5

ARITHMETIC UNIT

SECTION 5.1 – MEMORY BUFFER REGISTER – B (MB)

The B register is used exclusively as a buffer between the core memory modules and the Arithmetic and Control Units of the Central Processor. Transfer and modification paths of this register are presented in Figure 5.1–1.

FIGURE 5.1–1 Transfer and Modification Paths of B Register



From this figure, it is seen that there are separate transfer paths to internal and external memory modules. The transfer paths to the internal memory modules are quite simple since they are either single-sided transfers from the sense amplifiers of the memory to the corresponding bits of the B register or single-sided transfers from the bits of the B register to corresponding memory inhibit drivers. Single-sided transfers mean that only the true or false side of a register either transfers or receives a signal. The single-sided transfer from the memory sense amplifiers are to the true sides of the B register flip-flops. Thus, all 1's sensed in a memory location are stored in corresponding bits of the B register and, since the B register was cleared to zero prior to this transfer, the complete 1's and 0's configuration of the word in the selected memory location now exists in the B register. On the other hand, the false sides of the B register are sent to the inhibit drivers of the internal memory when it is desired to write the contents of the B register into some desired core memory location. The reason for this is that during a WRITE operation into core memory, the Write Drivers will try to force all bits of the memory location to the 1 state. (All bits of a memory location are forced to the 0 state during a READ operation and a READ operation must always precede a WRITE operation.) Inhibit Drivers fed by bits in the B register that are in the 0 state will "inhibit" the writing of a 1 in corresponding bits of the selected memory location. In this manner the configuration of 1's and 0's in the B register are impressed upon the desired memory location. A very complete and detailed discussion of the READ-WRITE process is available in Chapter 6.

As can be seen in Figure 5.1-1, the transfer paths to the auxiliary core memory modules are much more complicated than for the internal memory; however, in the final sense, the transfers to the B register or to an external memory module are identical to those discussed above. The additional circuitry shown in Figure 5.1-1 is necessary to

transmit over or receive from the Memory Interface Communication Line. This transfer is accomplished by sending or receiving all 33 bits of the word in parallel. A network similar to that shown in Figure 5.1–1 exists in the MM–10's to provide connection between the Memory Interface Communication Line and the B register in an MM–10.

The parity check and generation logic of Figure 5.1–1 is used whenever a word enters the B register from core memory, and prior to transferring a word from the B register to core memory. In the former case, the 33 bits of the B register are checked to see if odd parity (i.e., an odd number of 1's) exists in the word brought from memory. If this is not the case, then a memory parity error exists: the Parity Light on the Central Processor is lit, the Signal Tone is generated, and the Central Processor comes to an unconditional halt.

Figure 5.1–1 indicates two transfer paths from the M register to the B register. This, however, is not a completely accurate representation since the M(0)B path is actually comprised of three separate transfer paths. These paths are listed in Table 5.1–1.

The transfer path M(R21)B is used only to transfer the 21 most significant bits of a double precision word to the second word of storage of a double precision store operation.

The least significant 7 bits of the EA register are sent to bits 27B21 during store single precision operations and certain index operations, and to both words of a store double precision operation. The first 6 bits contain the value of the exponent associated with the number and the seventh bit (B27) contains the sign of the exponent (0 = plus, 1 = minus).

The condition of the Working Sign flip–flop (WS2) is transferred to bit

B28, and bits B30 and B31 are reset whenever a store operation other than a store logical or a store zero operation is performed. If a store pickapoint single precision operation is being performed, bit B27 is set. If a store pickapoint integer operation is being performed, bit B27 is reset. If a double precision store operation is performed, bit B29 is set. Otherwise, bit B29 is reset for all store operations except a store logical operation. On store logical, B29 takes on the configuration of the corresponding bit of the logic word in the M register. Information concerning the need for these transfers and modifications can be found in Sections 2.1–2 and 11.3.

TABLE 5.1–1 Complete Breakdown of the Composite M(0)B Transfer Path

Transfer Path	Use
M(0)20B0	Store for certain index operations, store single precision, store integer, store 21 least significant bits of double precision word.
M(0)26B0	Store for certain index operations – pickapoint mode, store pickapoint single precision, store pickapoint integer.
M(0)31B0	Store logic word, store zero, store input operation word, store (CA) by TRM opcode.

The Clear B signal is used to set all of the bits of the B register to zero prior to accessing a word from memory. This is necessary since the transfer from the memory to the B register is a single–sided transfer and only the set terms of the B register are affected by this transfer. (This is discussed earlier in this section.)

Figure 5.1–1 shows two transfer paths from the B register to the D register. Actually, the B(0)D path is composed of several individual

paths, similar to the M(0)B transfer. These individual paths are indicated in Table 5.1–2. The transfer path B(L21)41D21 is used exclusively to position the most significant bits of double precision word format.

TABLE 5.1–2 Breakdown of the Composite B(0)D Transfer Path

Transfer Path	Use
B(0)14D0	Operand assembly (A field of command word)
B(0)20D0	Single precision format, least significant bits of double precision format
B(0)26D0	Pickapoint format
B(0)31D0	Logic format, block transmit operations, temporary storage for second word of two–word block input/output or repeat command

There are two transfer paths from the B register to the EP register which serve one basic purpose and that is to transfer the exponent and its sign (bits 27 through 21 of the B register) of a floating point number brought out of memory to the exponent storage register, EP. The actual transfer paths are slightly more complicated than those given in Figure 5.1–1. The actual transfer paths and their uses are presented in Table 5.1–3. From this table it is seen that for a negative exponent, the exponent configuration (excluding exponent sign) of the B register is sent in complement form to the 6 least significant bits of the EP register and bits EP7 and EP6 are set. Bit EP7 represents the sign of the exponent (minus = 1) and bit EP6 is used to indicate exponent overflow (when a negative exponent exists and bit EP6 is reset, an exponent overflow condition exists). For a positive exponent, however, bits 27B21 are sent directly to the EP register and bits 7EP6 are reset.

Bit EP7 represents the exponent sign (plus = 0) and bit EP6 is used to indicate exponent overflow (when a positive exponent exists and bit EP6 is set, an exponent overflow condition exists).

TABLE 5.1–3 Breakdown of Transfer Paths from Register B to Register EP

Indicated Transfer	Actual Transfer	When Used
B(R21C)EP	B(R21C)5EP0 ^ Set 7EP6	If sign of exponent is negative
B(R21)EP	B(R21)5EP0 ^ Reset 7EP6	If sign of exponent is plus

The transfer paths of the CD and BA registers shown in Figure 5.1–1 were discussed in Section 4.3. The last path to be discussed, therefore, is the one showing bits 31B27 being fed into a block of control decoding logic. The decoding of these bits of the B register are shown in Table 5.1–4.

TABLE 5.1–4 Control Decoding of Bits 31 through 27 of the B Register

Bit	Decoded Meaning
B31	Interrupt Request Flag (when set)
B30	Interrupt Request Flag (when set)
B29	Length Tag (indicates double precision word when set on other than logic or command words)
B28	Mode bit in command word (indicated modes 1 or 3 when set, 0 or 2 when reset) mantissa sign in number word (set = minus, reset = plus)
B27	Integer bit in pickapoint word (reset = integer, set = use (PE) as exponent)

SECTION 5.2 – THE ARITHMETIC REGISTERS

5.2–1 THE N, D, S, M and A REGISTERS The arithmetic registers are defined as the N, D, S, M, and A registers. They are not as easy to discuss as were the control registers, since these registers make up the heart of the storage and shifting portions of the Arithmetic Unit. These registers are used in different capacities during the various arithmetic operations and often for diverse purposes during the processing of an individual operation. It can become extremely awkward and is confusing at best to try to discuss each transfer path and modification signal that affects the registers of this group. It was therefore decided to give the reader a good general discussion of these various registers at this time, leaving the detailed information concerning the transfers between registers to be picked up in Part IV where the sequencers that utilize these registers during the processing of commands are discussed.

The Adder, R, performs all of the summing or differencing of integer portions of numbers in the Central Processor. The N and D registers provide the inputs to the Adder circuitry. The N and D registers are also used in conjunction with the S register to perform shift operations. It is possible, by selecting appropriate paths between pairs of these registers, to combine a sum or difference operation with a shift operation. This is used as part of the implementation of multiply and divide as done by the Central Processor.

At the start of a sum operation, the augend is placed in the N register and the addend is placed in D. If the Subtractor–Comparator circuitry, which monitors the contents of the EA and EP registers, indicates that the exponet of the augend is greater than that of the addend, the N and S registers act together to accomplish octal shifts to the left while the

exponent associated with the augend is simultaneously decremented. If the augend has been shifted left until it is octally normalized in N and its exponent is still larger than the addends, then the addend is shifted right octally between the D and S registers in an attempt to equalize exponents by incrementing the exponent associated with the addend. If the Subtractor–Comparator indicates initially that the exponent of the addend is greater than that of the augend, then the contents of the N and D registers are interchanged before exponent equalization is attempted.

At the beginning of a subtract operation, the N and D registers hold the value of the minuend and the subtrahend, but not necessarily in that order. Since the heart of the Arithmetic Unit is an adder, performance of a subtract operation necessitates having one of the operands in 1's complement form. Which is complemented, the minuend or the subtrahend, is not really important, since it will not adversely affect the operation as far as the computer is concerned. Therefore, when the situation occurs during the processing of a subtract opcode where it is easier to complement one than the other, it is done.

The following rules control complementing during subtraction:

1) If the exponents are equal and no shifting is required, the contents of the N register are sent complemented (via the S register) to the D register and the contents of the D register are sent true to the N register.

2) If the exponents are not equal and shifting is required, then the contents of the register that is being shifted are complemented on the last shift that occurs when the exponents become equal or when the contents of the D register become zero.

The result of an add or subtract command is gated from the Adder to the S register. From S the results are sent to the N register where they will remain if they are to be the augend for the next sum or difference. If the number is to be the addend of the next operation, it is copied into the D register via S. If the number transferred from S to N is the final result of an opcode, it is copied into the Accumulator via S and M.

During multiplication, the multiplier is stored in the A register and the multiplicand is held in the N register. Before multiplication begins the multiplicand in N is shifted left until it is binarily normalized. It is not altered from this condition or transferrred from the N register during the multiplication. The multiplier, however, is constantly being modified by right shifts between the M and A registers. Multiplication is terminated when all of the 1 bits of the multiplier have been shifted out of the M and A registers. The partial product is formed by circulating between the D and S registers, constantly undergoing right shifts. The multiplicand stored in N is summed with the partial product stored in D by the Adder, R, and gated into the S register. The partial product in S is then shifted to the right and copied into the D register. Sometimes, however, only shifting between the D and S registers is necessary and the Adder is bypassed. The final product is octally justified and stored in the Accumulator.

Prior to any division operations, the divisor is shifted left in N until it is binarily normalized and it is then put in 1's complement form in D. The dividend is then placed in the N register. The divisor is then not altered or transferred during the subsequent division operation. The contents of the N register, originally the value of the dividend are shifted left between the N and S registers constantly being altered to a configuration corresponding to the partial remainder. The partial remainder may be combined with the divisor by the Adder before it is

shifted left or the partial remainder may be shifted to the left by transfers between S and N while completely bypassing the Adder. The quotient is formed in the M and A registers by shifting quotient bits inserted into the right–hand end of the A register to the left by transfers between the M and A registers. Division terminates when the quotient is octally normalized. The quotient is then octally justified and placed in the Accumulator.

It should be noted at this time that any information being stored into core memory because of bootstrap, input/output, or store operations must pass through the M register.

5.2–2 THE COMPOSITE REGISTERS: OA AND THE ACCUMULATOR

The reader will often come across mention of the Accumulator and OA registers in manuals describing the operation of the Central Processor. These registers are not physical entities; rather they are composite registers formed by using several registers. The primary use of the Accumulator is as a storage area where the results of most previous opcode operations are stored while the Central Processor is preparing to process the next command of a program. It is composed of the A register, which acts as storage for the mantissa of the results of an operation, the AE register which contains the value and the sign of the exponent associated with the contents of A, and the AS2 flip–flop which contains the sign associated with the contents of A.

The OA register is a composite register intended to act as temporary storage for the results obtained from processing operand assembly opcodes. The contents of the OA register are used in the formation of the operand of the next command. This allows the results of a previous non–operand assembly opcode to be retained in the Accumulator without fear of being clobbered while provisions are made to modify the operand of a succeeding command by the use of the operand

assembly opcodes. The OA register is composed of the N register which holds the mantissa of the results of the operand assembly operation, the EA register which contains the value and the sign of the exponent associated with the contents of N, and the WS flip–flop which contains the sign associated with the contents of N.

5.2–3 THE SIGN FLIP–FLOPS The SM, WS, AS, PS, and RS registers shown on the Machine Diagram, Figure 3.2–2, are in reality composed of one flip–flop each. These flip–flops bear the responsibility of maintaining sign conventions during arithmetic operations. As was stated previously, the sign associated with the Accumulator is held in AS. At the start of sum or difference operations, the sign AS is copied into WS (Working Sign). SM is fixed to record the type of operation that is being performed with SM set indicating a sum operation and SM reset indicating subtract. WS is then modified during the processing of the commands that involve sum or difference operations and at the end of the computation contains the sign of the result. At the finish of a sum or difference operation, the numerical value of the result is normally stored in the Accumulator or in the OA register with the sign of the result being stored in the corresponding composite–register sign flip–flop.

In multiply or divide operations, the sign of the number in the Accumulator is in AS and the sign of the number brought from memory is in WS. The signs of the two quantities being processed are compared and the sign of the final resultant is determined and stored in AS. The RS flip–flops provide only one function and that is to keep track of the sign of the remainder during division. The state of the RS flip–flops is used in forming the quotient of the divide operation. The PS flip–flops merely keep track of the sign of the partial product during multiply operations.

TABLE 5.2–1 Meanings Associated with Sign Flip–Flops

	SIGN FLIP–FLOPS				
Flip–flop State	SM	WS	AS	RS	B28*
High (1)	Add	–	–	–	–
Low (0)	Subtract	+	+	+	+

*Bit 28 of B register (when representing the sign of an operand).

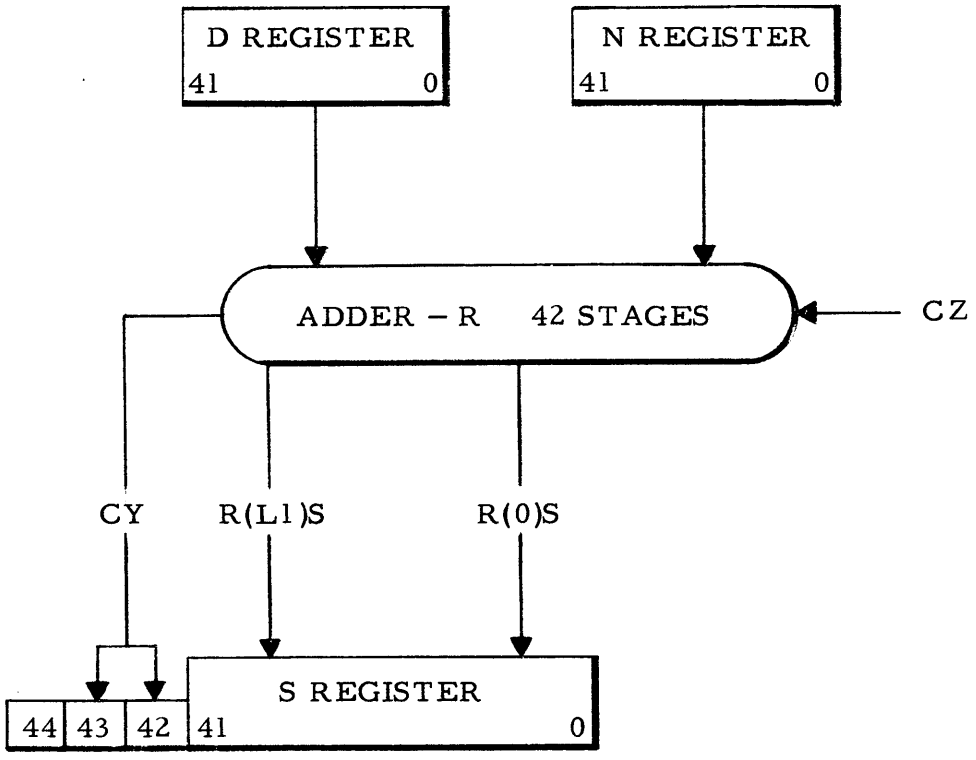
SECTION 5.3 – THE ADDER

Adder circuitry can be either serial or parallel in design. In a serial design there is one group of logic designated as the adder through which the two numbers to be added will progress, term–by–term. In a parallel design each bit of the two words to be added has its own adder logic associated with it and all bits are added simultaneously. Thus, at the expense of additional circuitry, increased speed of addition is obtained. Since, in a binary oriented machine, either addition or subtraction is normally used to accomplish the various arithmetic operations, it becomes very important to establish a very rapid means of implementing these processes. However, as speed increases, components, and therefore cost also increases, and the final design of the adder or subtractor results in some kind of a compromise between speed and cost. In the G–20 Central Processor, the Adder was chosen to be the heart of the machine with an end result that all arithmetic operations are accomplished by binary addition. By an ingenious method, referred to as leapfrog, a low cost, high–speed parallel Adder was developed which Bendix Computer Division has subsequently patented. This section will describe the operation of the G–20 Central Processor's Adder in detail.

5.3–1 GENERAL DESCRIPTION Figure 5.3–1 shows the transfer paths to and from the Adder, R. Since it is a parallel Adder, it consists of 42 stages that correspond to the bits of the N and D registers. Each stage is essentially identical to any other and each stage has one bit from both D and N registers fed into it. The configuration of the Adder stages are shown in Figure 5.3–2. The two operands to be combined are placed in the N and D registers and their sum is gated from R into the S register. It must be emphasized that the Adder will always contain the resultant of the contents of the N and D registers at any time during the operation of the Central Processor. It is only

during operations where this resultant is useful that the contents of R are gated into the S register. It might first appear advantageous to gate the resultant directly from the Adder back into either the N or D registers where it could be used as an operand in subsequent operations. This, however, is not feasible due to the high speed with which the Adder circuitry operates. The D and N registers must remain stable while the Adder, R, is being read.

FIGURE 5.3–1 Transfer Paths of the Adder, R



The resultant of the contents of the N and D registers, which is formed by the Adder, is sent to the S register when either the R(L1)S or the R(0)S transfer path is enabled. Both of these transfer paths are single–sided transfers from high bits of R and, thus, the S register

must be cleared prior to the insertion of the resultant. This resultant in R will be one of the following:

1) the sum of an add operation,
2) the difference of a subtract operation,
3) the partial or final product of a multiply operation,
4) the remainder in a divide operation,
5) the ANDed result of a logic extract operation,
6) the term that will be ORed with the contents of the S register during a logical unite operation (R to S is a single–sided transfer),
7) the previous resultant in R, +1, if a round–up operation is necessitated during the changing of exponents.

Whether this resultant is shifted left one or sent directly to the S register depends upon the operation being performed. The specific cases when the two transfer paths are used are too numerous and too detailed to be discussed at this time. Such discussions are presented in the appropriate sections of Chapter 13.

It was stated early in this section that the arithmetic operations of the Central Processor are performed by the use of binary addition. This statement was verified by the list presented in the above paragraph. It is obvious that all of these operations cannot be performed in exactly the same manner even though addition is considered to be basic to all. In addition and multiply operations the correct solution is obtained by an add process, or by a series of additions in the case of multiply. Subtract and divide operations can be handled by addition if the subtrahend and denominator are in complement form. This is discussed in detail in Chapter 13. Logic functions are handled differently by the Adder than the above cases and are discussed in detail in Sections 5.3–4 and 13.1.

The remaining terms associated with Adder of Figure 5.3–1 are the CY and CZ terms. The CY term when it is high indicates that a carry out has been generated from the last stage of the Adder. This means that bits N41 and D41 were both high or that either N41 or D41 was high and a carry in was present from the preceding stage of the Adder. The CY term, when it is generated, is transferred to either bit 42 or 43 of the S register, depending upon whether the R(0)S or R(L1)S transfer path is enabled.

The CZ term, when high, indicates that a carry in to the first stage of the Adder has been generated. This carry in is generated for the following cases:

1) prior to all difference operations,
2) prior to all divide operations,
3) whenever a round–up is indicated during the processing of add/subtract operations, multiply/divide operations, or a store single precision operation.

It should be noted that the CZ is never generated <u>prior</u> to the sum portion of an addition or multiply operation.

It was stated earlier that the subtrahend of a subtract operation is placed in complemented form and then added to the minuend. It will be shown in Chapter 13 that if the addition of these two numbers results in a carry out (CY) it indicates that the minuend was larger than the subtrahend and the correct answer will exist if the carry out bit is added to the least significant bit of the result in R. The Adder of the Central Processor anticipates this carry out during subtract operations and provides a carry in bit which is added along with the contents of the N and D registers, and if a carry out is generated, it is ignored. This eliminates the need for second addition during a subtract operation if a carry out (CY) is generated. It will also be shown in Chapter 13 that

if a carry out (CY) is not generated during a subtract operation, then the result in R will be correct only after it has been complemented. Since it is a subtract operation, however, a carry in (CZ) term has been automatically added anticipating the existence of a carry out (CY). If the resultant in R is complemented at this point, the complemented result will be one less than the actual correct answer. To compensate for this, the Adder circuitry of the Central Processor kills the carry in term (resets CZ) and re–adds whenever a carry out (CY) does not occur during a subtract operation. This can be done with a minimum loss of time since, as will be shown later in this section, the existence of a carry out (CY) term can be determined in the worst case in approximately 0.8 microsecond.

In divide operations, the CZ term is generated for reasons similar to those given for subtract operations. The operation is quite different, however, and the discussion of it will be delayed until Chapter 13.

The carry in term, CZ, to the first stage of the Adder is also generated whenever a round–up is required. This is caused by exponent adjustment during add/subtract, multiply/divide, or store single precision operations.

5.3–2 ADDER OPERATION The possible inputs to any stage of the Adder consist of:

1) the contents of the corresponding bit of the N and D registers,
2) a carry in signal from the preceding stage (CZ at the zero stage).

Since there are three possible inputs, the number of different combinations of these inputs is 2^3, or 8. These combinations and the resultant sum and/or carry out terms generated are listed in Table 5.3–1. It must be remembered that the generated carry out

terms of one stage are referred to as carry in terms at the next stage.

TABLE 5.3–1 Binary Sums Table

(N)	(D)	(Carry In)	Sum (S)	(Carry Out)	Remarks
0	0	0	0	0	$\overline{N} \wedge \overline{D} \wedge \overline{C}$
0	1	0	1	0	$\overline{N} \wedge D \wedge \overline{C}$
1	0	0	1	0	$N \wedge \overline{D} \wedge \overline{C}$
1	1	0	0	1	$N \wedge D \wedge \overline{C}$
0	0	1	1	0	$\overline{N} \wedge \overline{D} \wedge C$
0	1	1	0	1	$\overline{N} \wedge D \wedge C$
1	0	1	0	1	$N \wedge \overline{D} \wedge C$
1	1	1	1	1	$N \wedge D \wedge C$

From Table 5.3–1 it is seen that a sum will result when:

$$\text{Sum} = (\overline{N} \wedge \overline{D} \wedge C \text{ in}) \vee (\overline{N} \wedge D \wedge \overline{C \text{ in}}) \vee (N \wedge \overline{D} \wedge \overline{C \text{ in}}) \vee (N \wedge D \wedge C \text{ in}),$$

which reduces by Boolean manipulation to:

$$\text{Sum} = \overline{C \text{ in}} \wedge [(N \wedge \overline{D}) \vee (\overline{N} \wedge D)] \quad \vee \quad C \text{ in} \wedge [(\overline{N} \wedge \overline{D}) \vee (N \wedge D)] \ .$$

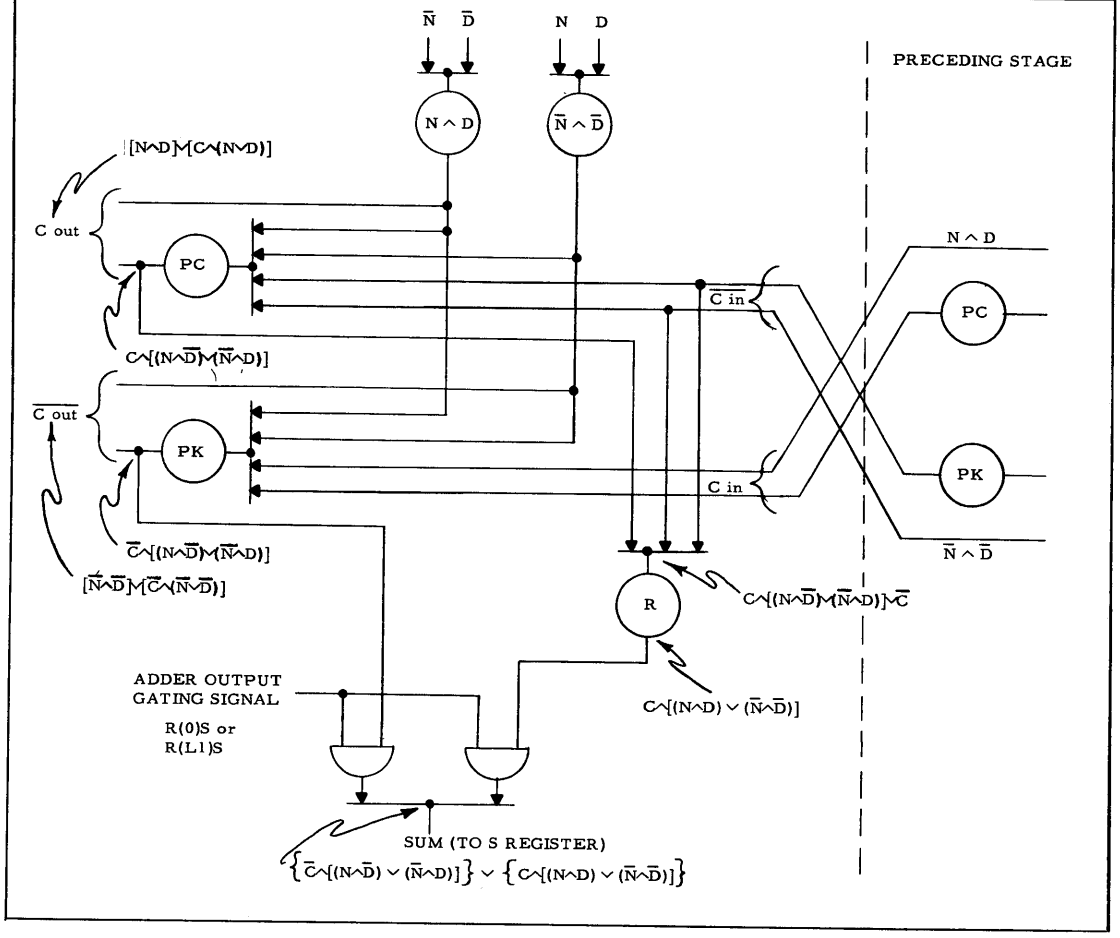
In the same way, the conditions for a carry out term can be reduced to:

$$\text{Carry out} = [N \wedge D] \vee [C \text{ in} \wedge (N \vee D)]$$

$$\overline{\text{Carry out}} = [\overline{N} \wedge \overline{D}] \vee [\overline{C \text{ in}} \wedge (\overline{N} \vee \overline{D})].$$

Figure 5.3–2 shows a representative, simplified stage of the Central Processor's Adder. More exact representations of stages are presented in Figures 5.3–3, 5.3–4, and 5.3–5. Figure 5.3–2 illustrates the basic method in which the aforementioned sum and carry out equations are implemented. From the Figure, it is seen that a carry out will occur when either the output of the $N \wedge D$ inverter or the PC

FIGURE 5.3–2 Simplified Adder Stage



inverter is high. PC is a "Propagate Carry" signal and it is high only when all of the terms of the OR–gate feeding it are low. This means that a $\overline{\text{carry out}}$ generated by one stage will automatically cause the PC signal of the next stage to be low. A similar analysis can be made for the PK, Propagate Kill, signal. In the equations associated with PC and PK, the $[(N \wedge \bar{D}) \vee (\bar{N} \wedge D)]$ term is the propagate term. When a condition exists such that the output of this equation is high for a stage and a carry or a kill term comes into the stage, then the same term leaves the stage. The remainder of Figure 5.3–2 should be self–evident with a little bit of study. The reader needs to have a knowledge

of the rules of Boolean algebra, however, to derive the equations listed there.

FIGURE 5.3–3 First Stage of Adder, R

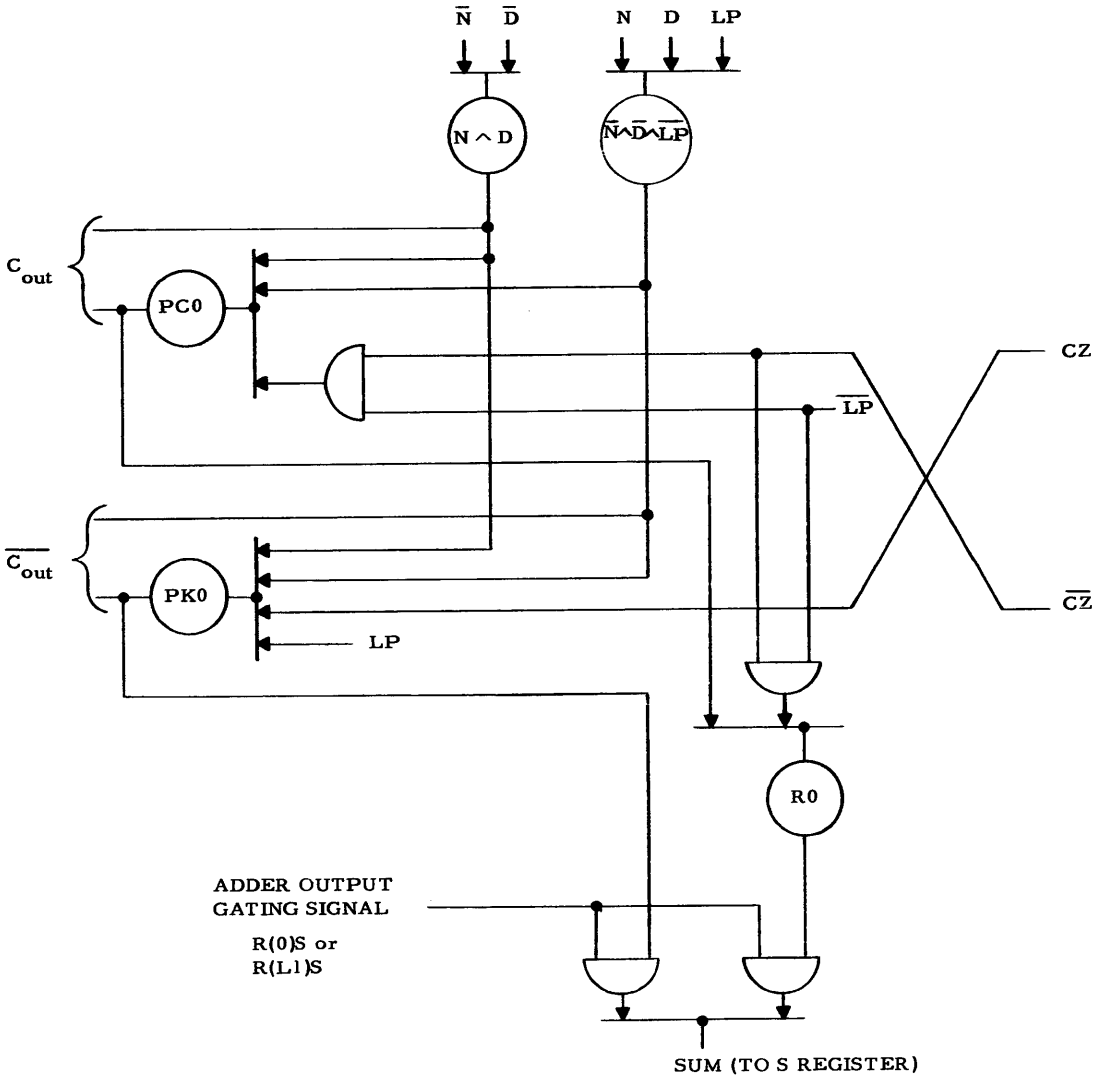


Figure 5.3–3 illustrates the actual configuration of the first stage of the Adder. It is quite similar to Figure 5.3–2, except that Logic Product (LP) terms and the zero order carry in terms (CZ) are added. The cases when a zero order carry in is desirable were discussed

earlier and logic operations of the Adder are discussed in Section 5.3–4. At this point, it is sufficient to mention the fact that, if a carry in exists or a logic product operation is being performed, one OR–gate term of both the PC0 and R0 inverters will always be low. The typical Adder stage of Figure 5.3–4 is identical to that of Figure 5.3–2 except that logic product, LP, terms which are discussed later in this section have been added. Figure 5.3–5 shows the final stage of the Adder. It, too, is the same as Figure 5.3–2, except that the final carry out, CY, is taken off $\overline{C}$ out portion of the last Adder stage and sent to the S register.

FIGURE 5.3–4 Typical Adder Stage (Stages 1 through 40)

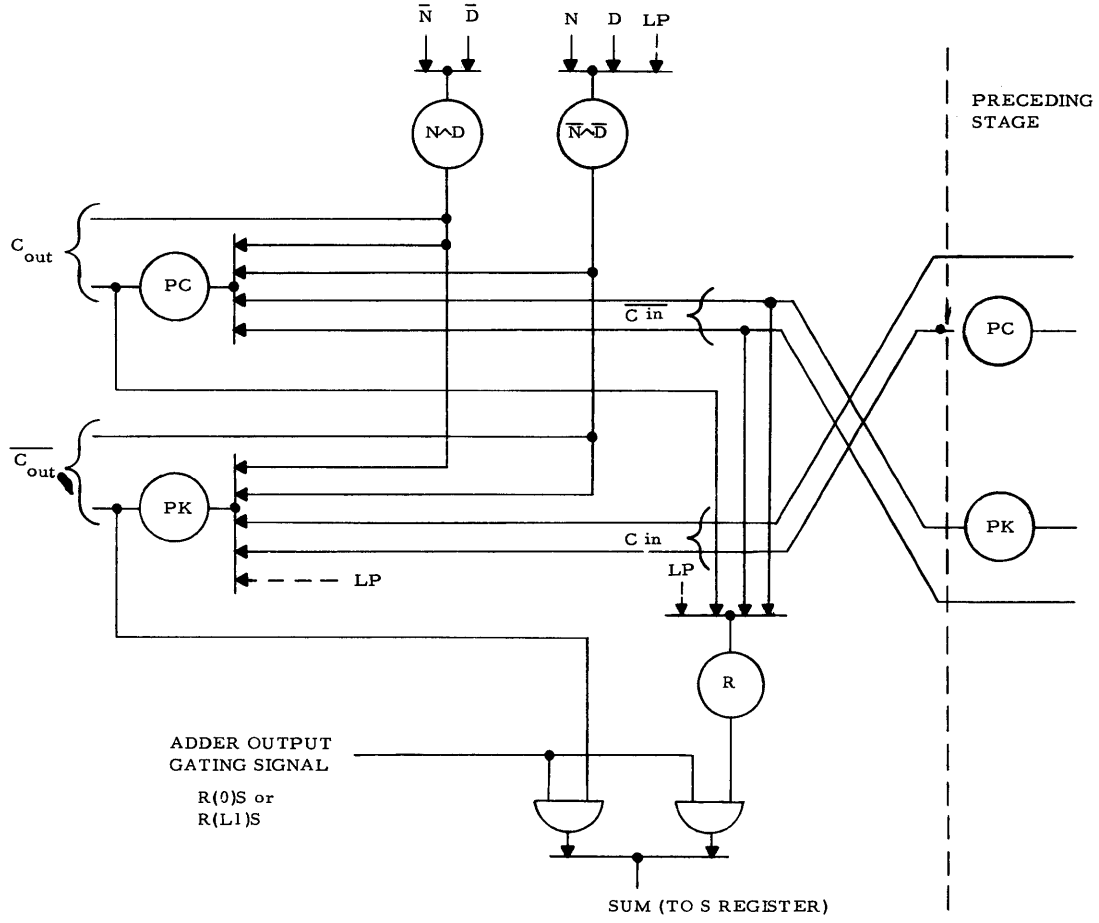
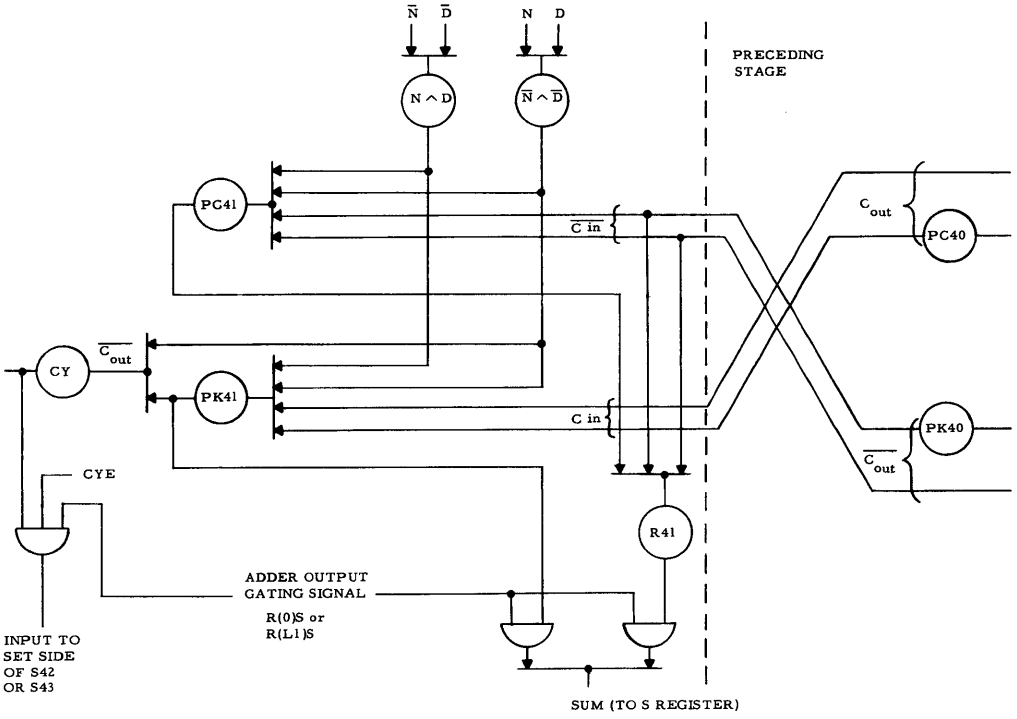


FIGURE 5.3–5 Last Stage of the Adder

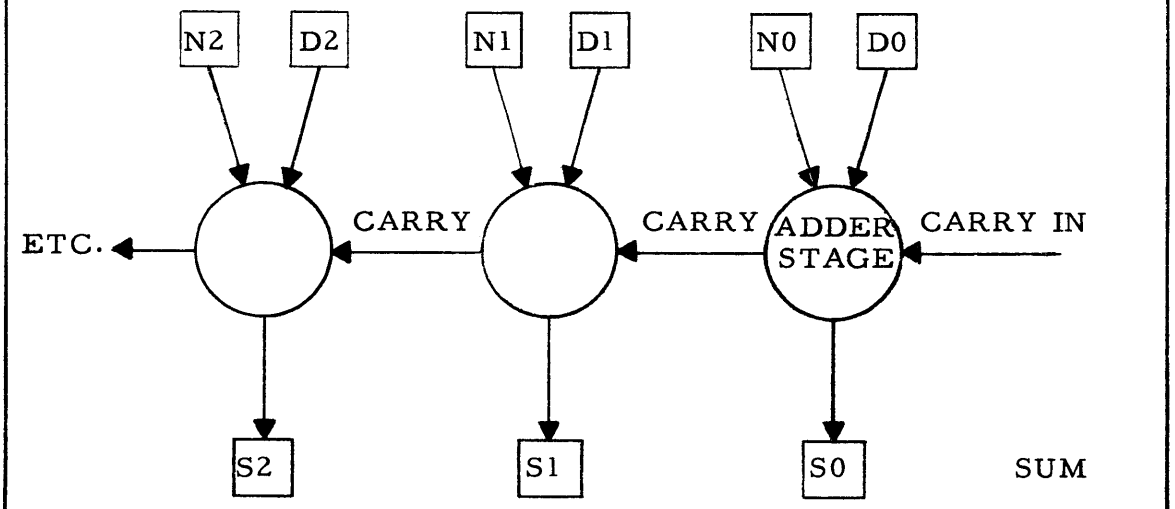


There is one group of inputs to various Adder stages that were not noted in Figure 5.3–2 through 5.3–5. They are the outputs of the leapfrog circuits that were designed to speed up addition in the Central Processor. The leapfrog concept is quite complex and therefore it is discussed separately.

5.3–3 LEAPFROG CONCEPTS In a parallel adder, it is not enough to just add 2 bits of information together to get a result. Instead, carries generated by the various stages must also be accounted for, and, in a parallel adder, the time required for propagation of carries through all the stages of the Adder is the determining factor in its speed of operation. Figure 5.3–6 shows a pictorial presentation of the G–20 Adder neglecting leapfrog circuits. Table 5.3–2 lists the time that would be required to propagate a carry through the 42 stages of

this type of Adder. From this table, it is seen that it would be necessary to allow approximately 2.3 microseconds or 2–1/2 clock times to sum two numbers using the adder shown in Figure 5.3–6. It is demonstrated later that, by using the leapfrog circuits with this adder, the worst case sum time (with a time safety margin) is 1.5 microseconds (1–1/2 clock times). This results in a 40 per cent reduction of adder summing time.

FIGURE 5.3–6 Basic Configuration of Adder, R



Leapfrog circuits are used to speed up the propagation of both carries and kills through the long chain of adder stages. Leapfrogs form loops that bridge a number of adder stages and "predict" whether a carry or $\overline{\text{carry}}$ will emerge from the group. This prediction is then ORed with the normal carry input to the stage following the high–order end of the leapfrog loop. Figure 5.3–7 shows a block diagram representation of a typical leapfrog of the Adder of the Central Processor.

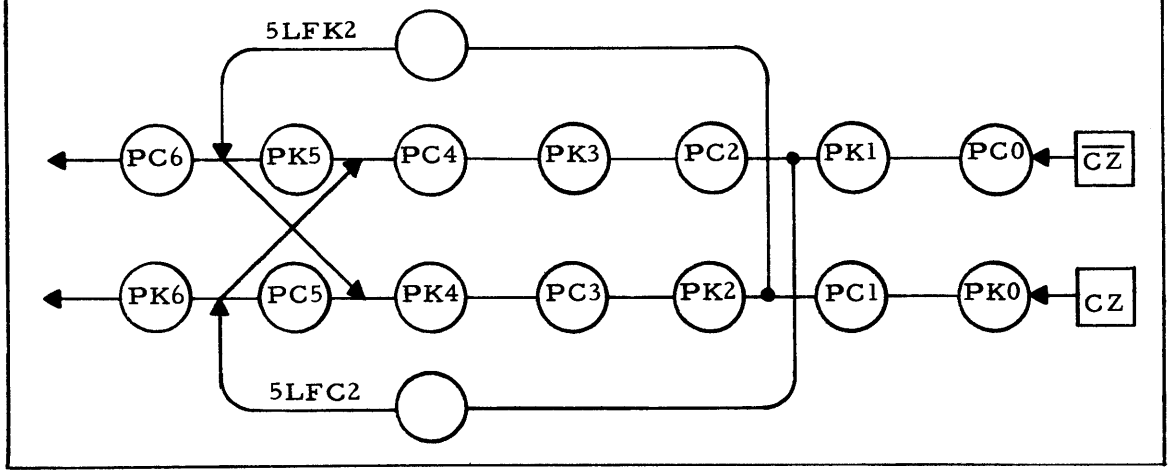
TABLE 5.3–2 Worst Case Carry Propagate Time for Adder of Figure 5.3–6

Process	Time	Elapsed Time
Load N and D registers	–	0.00 usec.
Time required for N and D flip–flops to stabilize	0.235 usec.	0.235 usec.
Signal enters first OR–gate to a carry inverter	0.048 msec.	0.283 usec.
Carry goes through 42 OR–gates and 42 inverters and leaves highest order Adder carry inverter	= 42 x 0.048 or 2.006 usec.	2.289 usec.

It will be recalled from Figure 5.3–2 that the PC signal of one stage of the Adder becomes one term of the OR–gate to the PK inverter of the next stage. If the PC signal is high, it causes the output of the PK inverter, which is one input to the PC inverter of the succeeding stage, to be low. If the remaining OR–gate terms of this PC inverter are low, its output will be high. Thus, a carry signal would be propagated under the right conditions from a PC inverter to a PK inverter, to a PC inverter, etc., through all stages of the Adder. The same would be true of a generated "kill" signal. The two strings of connected inverters in Figure 5.3–7 represent the two propagate paths; the top path is for the propagate carry condition, the lower path, the propagate kill condition. The leapfrog circuits are designed to take advantage of certain strings of propagate conditions, either carry or kill, whenever they occur during a sum operation. It should be noted that it is impossible to generate a PC and a PK signal at the same time from an Adder stage, but that it is not necessary to generate either. The same can be said of the LFC and LFK signals.

The configurations of an actual leapfrog carry (LFC) and a leapfrog

FIGURE 5.3–7 Typical Leapfrog of G–20 Adder



kill (LFK) circuit are generated in Figure 5.3–8.

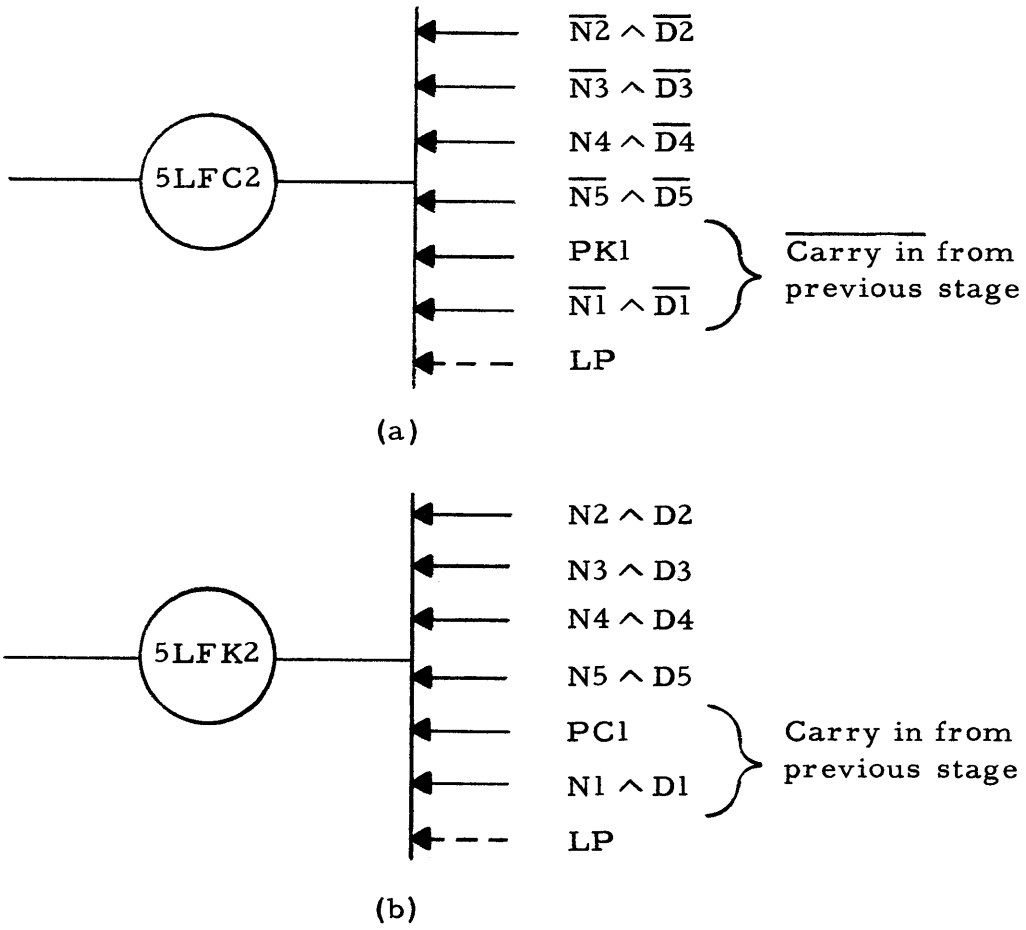
The output equations of the LFC and LFK inverters are of the form:

$$iLFCn = Cin \wedge [(N_n \vee D_n) \wedge (N_{n+1} \vee D_{n+1}) \wedge \ldots (N_{i-1} \vee D_{i-1}) \wedge (N_i \vee D_i)]$$

$$iLFKn = \overline{Cin} \wedge [(\overline{N}_n \vee \overline{D}_n) \wedge (\overline{N}_{n+1} \vee \overline{D}_{n+1}) \wedge \ldots (\overline{N}_{n-1} \vee \overline{D}_{n-1}) \wedge (\overline{N_i} \vee \overline{D}_i)]$$

where i is the most significant stage of the Adder that is encompassed by the leapfrog circuit and n is the least significant. It will be noticed that the logic product, LP, terms that appear in the figure are not included in the output equations. This term is added to prevent the leapfrog circuits from operating if a logical operation rather than an arithmetic process is being handled by the Adder. This is done since in LP operations there is not numerical coupling between Adder stages.

FIGURE 5.3–8 Representative Leapfrog Circuits
(a) Leapfrog Carry (b) Leapfrog Kill



With Figures 5.3–7 and 5.3–8 in mind, the following general statements can be made about a group of inverters (stages of the Adder) that are spanned by a leapfrog circuit.

1. A "generate kill" condition exists when the N and D bits feeding an Adder stage within the group are both 0's.

2. A "generate carry" condition exists when the N and D bits feeding an Adder stage within the group are both 1's.

3. If a carry enters the group of Adder stages and no generate kill

condition exists within the group, a carry must leave the group. (The LFC inverter output will be high.)

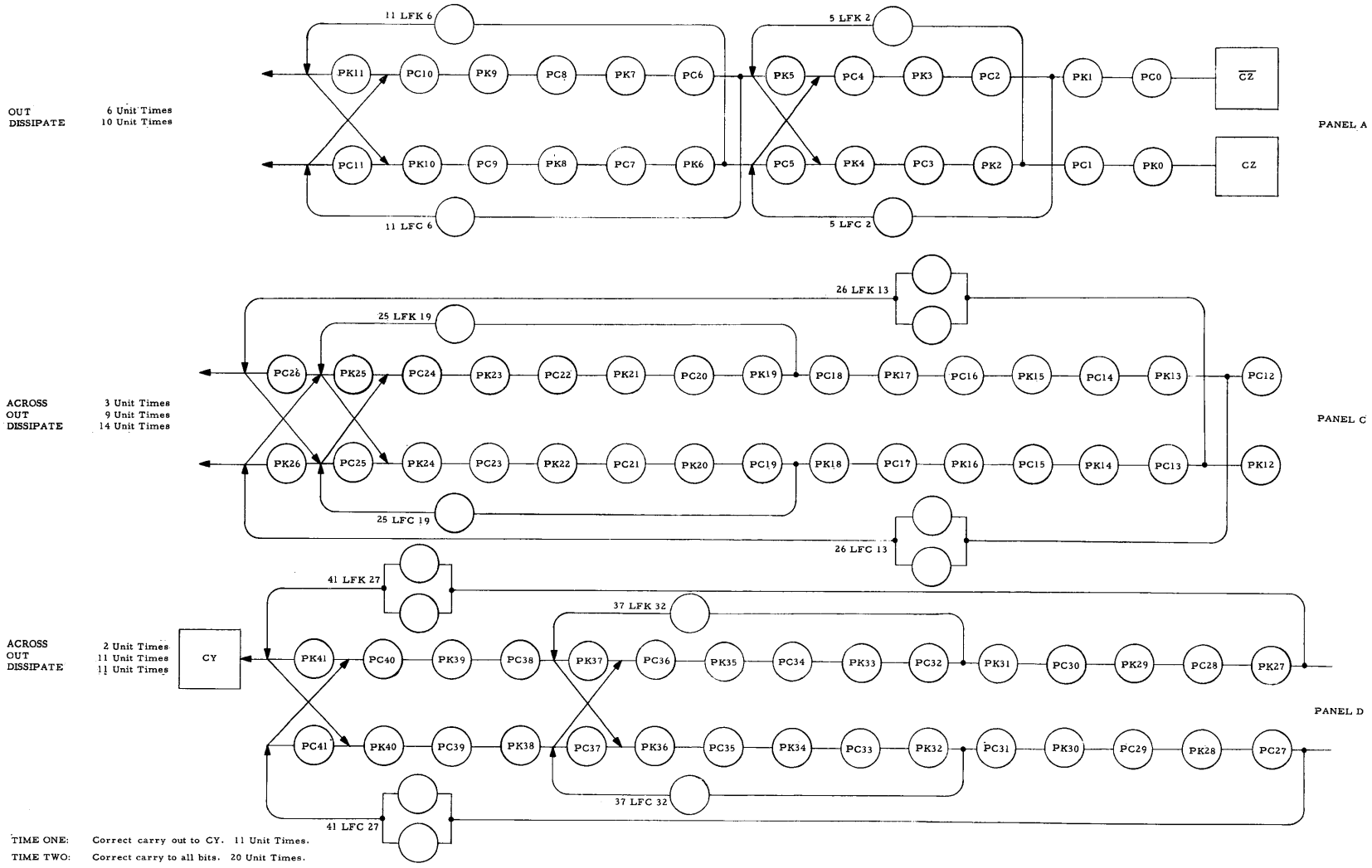
4. If a kill ($\overline{\text{carry}}$) enters the group and no generate carry condition exists within the group, a kill must leave the group. (The LFK inverter output will be high.)

5. LFC and LFK may be low simultaneously, in which case the leapfrog loop will be inoperative. This is a very important point, and can be explained as follows:
   (a) A $\overline{\text{carry}}$ may enter the group, disqualifying (3) above, but a generate carry may be generated within the group, disqualifying (4) above.
   (b) A carry may enter, disqualifying (4), but a generate kill exists in some stage, disqualifying (3).
   (c) Both a generate kill and a generate carry may exist within the group, disqualifying both (3) and (4).

6. Note that in all cases where both LFC and LFK are low, either a generate carry or a generate kill or both will be generated within the enclosed stages. The import of this fact is that, under these conditions, the leapfrogs are unnecessary. If a kill or a carry is generated within a group, nothing that happens prior to that position has any effect on the situation as far as carries are concerned. We are then concerned only with the propagate time from that point on, and the leapfrog signals are unnecessary for that particular group.

The reader may wonder at the "feedback" of the LFC and LFK signals to the inputs of the last Adder stage enclosed by the leapfrog loop of Figure 5.3–7. This assures a stable signal at this point if an LFC or LFK is high. Without this feature, the output of the final stage enclosed by the leapfrog would be random (until all carries were

FIGURE 5.3–9 Adder Leapfrog Paths



"settled down") and could confuse the LFC and LFK prediction.

Figure 5.3–9 shows all the leapfrog paths and gives timing information about the Adder. Some definitions of timing terms are necessary at this point in order to fully appreciate Figure 5.3–9.

1. Out Time – – maximum time required for a carry or $\overline{\text{carry}}$ generated on a panel to propagate out of the panel.
2. Dissipation Time – – time required for <u>all</u> the carry circuits of a panel to stabilize after correct carry information has entered the panel.
3. Across Time – – time required for carry information entering a panel to propagate across and leave the panel.
4. Time One – – time required for the end–carry (C42) to become stable and correct. Time One is the sum of panel A's Out Time and panel C and panel D's Across Times.
5. Time Two – – time required for <u>all</u> carry circuits to be stable. Time Two is the sum of panel A's Out Time, panel C's Across Time and panel D's Dissipation Time.
6. One unit of time of Figure 5.3–9 = 0.048 microseconds, the time required for propagation through one inverter stage (one OR–gate and one inverter).

Four of the leapfrogs in Figure 5.3–9 show two inverters in parallel. This is necessary only because of the large number of inputs required (long loops). Logically, all leapfrogs are equivalent.

The reader may be inclined to feel that a more uniform or symmetrical division of bits could have been made on the LFC and LFK paths. To understand the reason for the choise as shown, observe the following

facts:

1. Out Time from panel A (6) plus Across Time for panel C (3) is equal to the Out Time of panel C (9).
2. This figure (9) plus the Across Time of panel D (2) is equal to the Out Time of panel D (11).
3. Items 1 and 2 show that CY will be stable and correct at Time One (11).
4. Out Time of A (6) plus Dissipate Time of C (14) is equal to (20).
5. Out Time of A, plus Across Time of C (note item 1 above), plus Dissipate Time of D (11), is equal to (20).
6. In view of these facts, we can state that <u>all</u> carry circuits will be stable in (20) – – Time Two.

<u>Note:</u> Numbers in parentheses are unit times.

Earlier in this section it was stated that the leapfrog circuits allow a sum to be completed by the Adder in 1.5 microseconds. Table 5.3–3 presents the worst case times for the carry out term CY to be stable and for the carry circuits of the Adder to stabilize.

From Table 5.3–3 it is seen that the clock times allocated for these operations are adequate. The timing sequence of a sum operation of the Adder is presented below:

C2 – Load N and D registers
C1 – Clear S register
C2 – Wait
C1 – Gate R(0)S or R(L1)S

<u>5.3–4 LOGIC FUNCTION OF ADDER, R</u> The logic product input,

TABLE 5.3–3 Worst Case Times for Central Processor Adder
(a) CY Stabilized, (b) All Carry Circuits Stabilized

Time required for CZ, N, and D flip–flops to stabilize	0.235 usec.
Time One = 11 x 0.048	0.528 usec.
Two AND–gates on leapfrogs = 2 x 0.018	0.036 usec.
	0.799 usec.
1 full clock = 2 x 0.5	1.000 usec.
(a)	
CZ, N, and D stabilization	0.235 usec.
Time Two = 20 x 0.048	0.960 usec.
Time for two AND–gates	0.036 usec.
Time required to pass through R inverters	0.048 usec.
	1.279 usec.
1–1/2 clocks = 3 x 0.5	1.500 usec.
(b)	

LP, is supplied to only the first 33 stages of the Adder. The output of the remaining stages will automatically be zero whenever a LP operation is being performed, since the corresponding bits of the N and D registers are cleared prior to such an operation. The output of the thirty–third stage of the Adder will also be zero during LP operations, but it is necessary to supply an LP term to the OR–gate of the R inverter of this stage to insure it. This is the only stage that has an LP term supplied to the R inverter. This is done since an unconditional carry is generated through the first thirty–two stages of the Adder during LP operations and the presence of the LP term prevents the carry out generated by the thirty–second stage from being placed in the S register when the transfer is made.

It will be noticed that in Figure 5.3–4 the LP terms are represented by dashed lines. This indicates that these inputs are not common to all stages represented by this figure. Analysis of this figure shows that the first 32 stages of the Adder that have LP input terms to the $\overline{N} \wedge \overline{D}$ inverter will always have low (0) outputs from the $\overline{N} \wedge \overline{D}$ inverters during logic product operations. It was stated above that there is an unconditional carry generated by each of these stages. This means that $PC \vee (N \wedge D)$ signals of these stages will always be high during logic product operations. An investigation of Figure 5.3–4 shows that the output of the $PC \vee (N \wedge D)$ signal will be high if:

1) the output from the preceding PK inverter is low and output of the $N \wedge D$ signal inverter of the stage is low, or

2) the output of the $N \wedge D$ signal is high.

It cannot be guaranteed that the $N \wedge D$ signal of each stage of this group will be high, but if it can be guaranteed that the PK signal of the preceding stage is low during LP operations, then it is impossible, barring component failure, to have anything but a high output from the $PC \vee (N \wedge D)$ signals of these stages. Thus, if it can be insured that the $PC \vee (N \wedge D)$ signal of the first stage of the Adder is high and that PK is low, the propagation of an unconditional carry through of these stages is insured. Figure 5.3–3 shows the configuration of the first stage of the Adder. The logic product, LP, signals of this stage guarangee that PK0 is low and that the output of the $PC0 \wedge (N0 \wedge D0)$ signal is high. The LP input terms of the first Adder stage are sufficient to insure that the unconditional carry is generated throughout these stages; however, the required propagation time would be objectionable. To reduce this propagation time, additional LP terms are added to the PK inverters periodically through the first 30 stages.

Now that the unconditional carry is clear, it would be nice to know how

the results of the logic product are formed. Again referring to Figure 5.3–4, it is seen that if the corresponding bits of the N and D registers are high, the output of the N $\wedge$ D inverter of a stage is high and the output of the PC inverter is low. This low PC output is fed to the input of the R inverter which makes all the inputs to this inverter low. (Remember that the LP input term to the R inverter indicated in Figure 5.3–4, it is seen that if the corresponding bits of the N and D registers are high, the output of the N $\wedge$ D inverter of a stage is high and the output of the PC inverter is low. This low PC output is fed to the input of the R inverter which makes all the inputs to this inverter low. (Remember that the LP input term to the R inverter indicated in Figure 5.3–4 occurs only at the thirty–third Adder stage.) Therefore, the output of the R inverter is high and it is gated into the S register at the appropriate time. When the output of the PC inverter of a stage is high, indicating $(\overline{N} \vee \overline{D}) \vee (\overline{N} \wedge \overline{D})$, the output of the R inverter is low and a 0 will be gated into the S register.

Since there is an unconditional carry propagated through the first 32 stages of the Adder and a carry out into the thirty–third stage during LP operations, the leapfrog circuits would provide no useful information. Therefore, they are disabled during logic product operations.

The Adder is also used to form logical sum combinations of the (N) and the (D). This operation is performed by first transferring the (N) into the S register with a left shift of one [N(L1)S]. Then, N is cleared, and <u>without</u> clearing S, we gate R(L1)S. A moment's reflection will reveal that since the Adder output is single–sided, S now contains all the 1 bits from the previous contents of both N and D (shifted 1 bit to the left, a very minor inconvenience).

SECTION 5.4 – EXPONENT CIRCUITRY

Since the Central Processor was designed as a floating point computer with limited additional facilities for fixed point operation, exponent manipulation therefore becomes of great importance. The exponent circuitry of the Central Processor consists of the exponent registers (AE, EA, EP and ES), the Subtractor–Comparator circuitry, and the Modulo Three Counter (refer to the Machine Diagram, Figure 3.2–2).

5.4–1 THE EXPONENT REGISTERS The exponent registers, similar to the arithmetic registers, are used quite extensively during the processing of arithmetic commands. Detailed discussions of all transfer paths and modification signals would be quite complex and would probably be rather meaningless to the reader at this point. Therefore, the detailed discussions of transfers between these registers are left to the sections of Part IV where the various arithmetic operations are discussed. The aim of this section is to present a broad, generalized view of the registers and their basic uses.

The AE register is a 7–bit register used exclusively to store the exponent associated with a number stored in the Accumulator. The least significant 6 bits holds the octal value of the exponent while the seventh bit holds the sign of the exponent. The AE register holds the exponent in true form if it is a positive exponent and in 1's complement form if it is negative. During store operations, if it is a negative exponent, it is sent to the EA register and then recomplemented by transfers between the EA and ES registers before it is put into memory.

During address preparation, the Accumulator exponent remains in the AE register, but when execution of an opcode begins AE is copied into EA. When a word is brought from memory in floating point mode, the value of the exponent is transferred from the B register to the EP

register. If pickapoint mode operations are being performed, the exponent of the word is contained in the PE register and is transferred to the EP register. In either case, if the exponent of the word from memory is negative, then the value of the exponent (except for sign) is sent to the EP register in 1's complement form. Certain operands are always given zero exponent (logic words for example). This is accomplished by clearing EP to zero or by shifting the operand and modifying the exponent. The EP register consists of 8 bits, the least significant 6 bits of which are used to indicate exponent magnitude. The 2 most significant bits indicate sign and that overflow or underflow conditions exist when the contents of both bits are not identical.

In add or subtract operations, the EA register is used as a means of equalizing exponents and it is normally the register that contains the final value of the exponent of an operation. The EP register on the other hand is used primarily for exponent equalization and the ES register is used as a way station on the path that increments or decrements the EA or EP registers.

When add or subtract operations are being performed and the exponents of the two operands involved are unequal, the contents of the N register are shifted left in an attempt at equalization and the contents of the exponent register associated with the <u>number</u> in the register are decremented by one for each octal shift to the left. If the exponents are still unequal by the time the N register is octally normalized then the contents of the D register are shifted right and the exponent associated with it is incremented by one for each octal shift to the right. The EA register is normally associated with the N register and is normally incremented or decremented each time an octal shift occurs in N. The EP register performs a similar function for the D register. (There is a situation where the above two statements do not hold. This case is discussed in detail in Chapter 13.) If the contents of N were

initially zero, the exponent of the result will be in the EP register at completion. If the contents of N were other than zero, the exponent will be in the EA register at completion. In the zero case, the exponent is also in ES and in the non-zero case the correct exponent plus 1 is in ES.

The EA register consists of 9 bits, of which only the 6 least significant bits are used for exponent magnitude purposes. The other 3 bits are needed to indicate sign and overflow or underflow conditions (exponent greater than +63 or less than -63) similar to the EP register, but in addition the EA register must be able to determine if over-overflow or under-underflow conditions exist as the result of a multiply or divide operation (exponent greater than +127 or less than -127). This feature is presented in detail in Section 13.8.

During multiply and divide operations the EA and EP registers are used somewhat differently than in sum or difference operations. In multiply or divide there is no worry about the exponents of the operands being equal before operations can occur. Rather, all that is important is the resultant exponent of the two operands and this can be obtained by direct addition in the case of multiply or by subtraction for divide. Therefore, the EA and EP registers become inputs to a combining circuit similar to the N and D registers being inputs to the Adder during sum or difference operations. In this case, however, the combining circuit is a subtractor rather than an adder. (The subtractor is discussed in detail in Section 5.4-3.) The resultant of the two exponents is gated into the ES register. At the finish of multiply or divide operations, if an overflow or underflow condition exists, an attempt is made to save any and all information that might exist in the acceptable exponent range. To do this, the exponent resultant is incremented or decremented between the ES and EA registers while the mantissa resultant is being shifted octally to the right or to the left in the

arithmetic registers.

5.4–2 THE MODULO THREE COUNTER The exponents of numbers in the Central Processor are to the base 8 and therefore, shifts of 3 binary digits in the mantissas must occur before recognition that shifts have occurred can be recorded in the exponent registers. During multiply and divide operations, however, single binary shifts to the right or to the left that are made during binary normalization or on the product or quotient during processing are not necessarily in groups of three and, therefore, it is not always possible to make appropriate changes in the exponent registers. It goes without saying, however, that these binary shifts must be kept track of to keep the exponent correct. The Modulo Three Counter, composed of the MM and MT registers, is provided for this purpose.

The MM and MT registers, as shown in Figure 3.2–2, are composed of 2 bits each and are connected by ± 1 incrementing or decrementing circuitry. Both incrementing and decrementing provisions are necessary since, as was shown in Section 5.4–1, the mantissas may be shifted to right or to the left. The counting sequence of the Modulo Three Counter is shown in Table 5.4–1. If the Modulo Three Counter contains an octal fraction at the finish of a multiply or divide operation, it must be removed before the result can be stored in the Accumulator. Since it is a fraction the exponent registers cannot be modified to indicate this fact. It is therefore necessary to shift the resultant binarily to the right or left until the Modulo Three Counter is cleared. This is referred to as octally justifying the resultant.

Decoding to determine binary shifting for the divide/multiply opcodes is done on C1's so the decoding for octal shifts is also done on C1's. The exponent counter (ES and EA) accepts change count signals only on C2's, so a Change Count Delay flip–flop (CED) is provided to delay the Change

Octal Exponent signal (CHE). The exponent counter should count one if the Modulo Three Counter MT register contains 10 and an MT (+1) MM increment signal is received or if the MT register contains 00 and an MT (–1) MM decrement signal is received. In the first case CED acts like a carry out of the Modulo Three Counter, while in the second case CED acts like a borrow from the exponent counter. Interpretation of CED (whether it means count up or count down) is done by detecting the opcode and the sequencer state. CED is reset unless the divide/multiply opcode sequencer is running.

TABLE 5.4–1 Modulo Three Count Sequence

	Increment Count		Decrement Count	
Conditions	Counter Bits	Octal Count	Counter Bits	Octal Count
Initial State	00	0	00*	1
First Binary Shift	01	1/3	10	2/3
Second Binary Shift	10	2/3	01	1/3
Third Binary Shift	00**	1	00	0

*The exponent counter (ES and EA) is decremented by one.

**The exponent counter is incremented by one.

5.4–3 THE SUBTRACTOR–COMPARATOR Whenever the Central Processor is handling arithmetic operations not only must the correct value of the exponent be formed but, during processing of sum and difference operations, the Central Processor must also know the relative magnitude of the exponents involved (i.e., is the magnitude of one exponent greater, less, or equal to the magnitude of the other

number). The Subtractor–Comparator logic provides the necessary exponent information and forms the initial exponent of the resultant during multiply and divide operations.

As the name implies, the Subtractor–Comparator includes two loosely associated groups of logic that provide basically unrelated functions during the processing of arithmetic commands. The Subtractor, as we will see later, is quite similar to the Adder logic previously discussed. The Comparator circuitry, on the other hand, determines the relative size of the exponents of the two numbers involved in sum or difference operations. The relative magnitudes of the exponents in the EA and EP registers determine how the sum or difference operations are handled in the Central Processor. Since these two groups of logic perform basically different operations with the exponents they will be discussed separately.

The Subtractor. A subtractor was chosen instead of an adder since certain computed signals needed for the Comparator logic are generated by the Subtractor and thus, a more efficient utilization of logic circuitry is realized. The Subtractor consists of eight stages and it is used exclusively to determine the initial value of the combined exponents of the operands of a multiply or divide operation. The inputs from the EA and EP registers are fed, unclocked, into the Subtractor stages and the resultant is continually being formed as soon as information is placed or changed in the EA or EP registers. Only during the processing of multiply and divide opcodes, however, are the Subtractor stages gated into the ES register.

The Subtractor always forms the resultant (EA) – (EP). However, similar to the Adder, where differencing is done by putting one of the operands in 1's complement form and then adding, the Subtractor can perform summing by complementing one of the exponents and then

subtracting. The actual way it is done for multiplication is to complement EA, which gives (–EA) – EP, or – (EA + EP). When the results of this equation are in the ES register, the results are then complemented which gives the desired results of + (EA + EP).

As with the Adder, since there are three possible inputs to each Subtractor stage (corresponding bits of EA and EP register and a borrow term) there are eight different possible combinations. These are presented in Table 5.4–2.

TABLE 5.4–2 Binary Differences Table

(EA)	(EP)	(Borrow In)	Difference	(Borrow Out)	Remarks
0	0	1	1	1	$\overline{EA} \wedge \overline{EP} \wedge B$ in
0	1	1	0	0	$\overline{EA} \wedge EP \wedge B$ in
1	0	1	0	0	$EA \wedge \overline{EP} \wedge B$ in
1	1	1	1	1	$EA \wedge EP \wedge B$ in
0	0	0	0	0	$\overline{EA} \wedge \overline{EP} \wedge \overline{B \text{ in}}$
0	1	0	1	1	$\overline{EA} \wedge EP \wedge \overline{B \text{ in}}$
1	0	0	1	0	$EA \wedge \overline{EP} \wedge \overline{B \text{ in}}$
1	1	0	0	0	$EA \wedge EP \wedge \overline{B \text{ in}}$

By Boolean manipulation, the conditions found in the second section of Table 5.4–2, can be reduced to:

Difference = $\left\{B \text{ in} \wedge [(EA \wedge EP) \vee (\overline{EA} \wedge \overline{EP})]\right\} \vee \left\{\overline{B \text{ in}} \wedge [(EA \wedge \overline{EP}) \vee (\overline{EA} \wedge EP)]\right\}$

Borrow Out $[\overline{EA} \wedge EP] \vee [B \text{ in} \wedge (\overline{EA} \vee EP)]$

$\overline{\text{Borrow Out}}$ $[EA \wedge \overline{EP}] \vee [\overline{B \text{ in}} \wedge (EA \vee \overline{EP})]$

Figures 5.4–1 and 5.4–2 present the logic of the first stage of the Subtractor and a typical Subtractor stage with logic equations included.

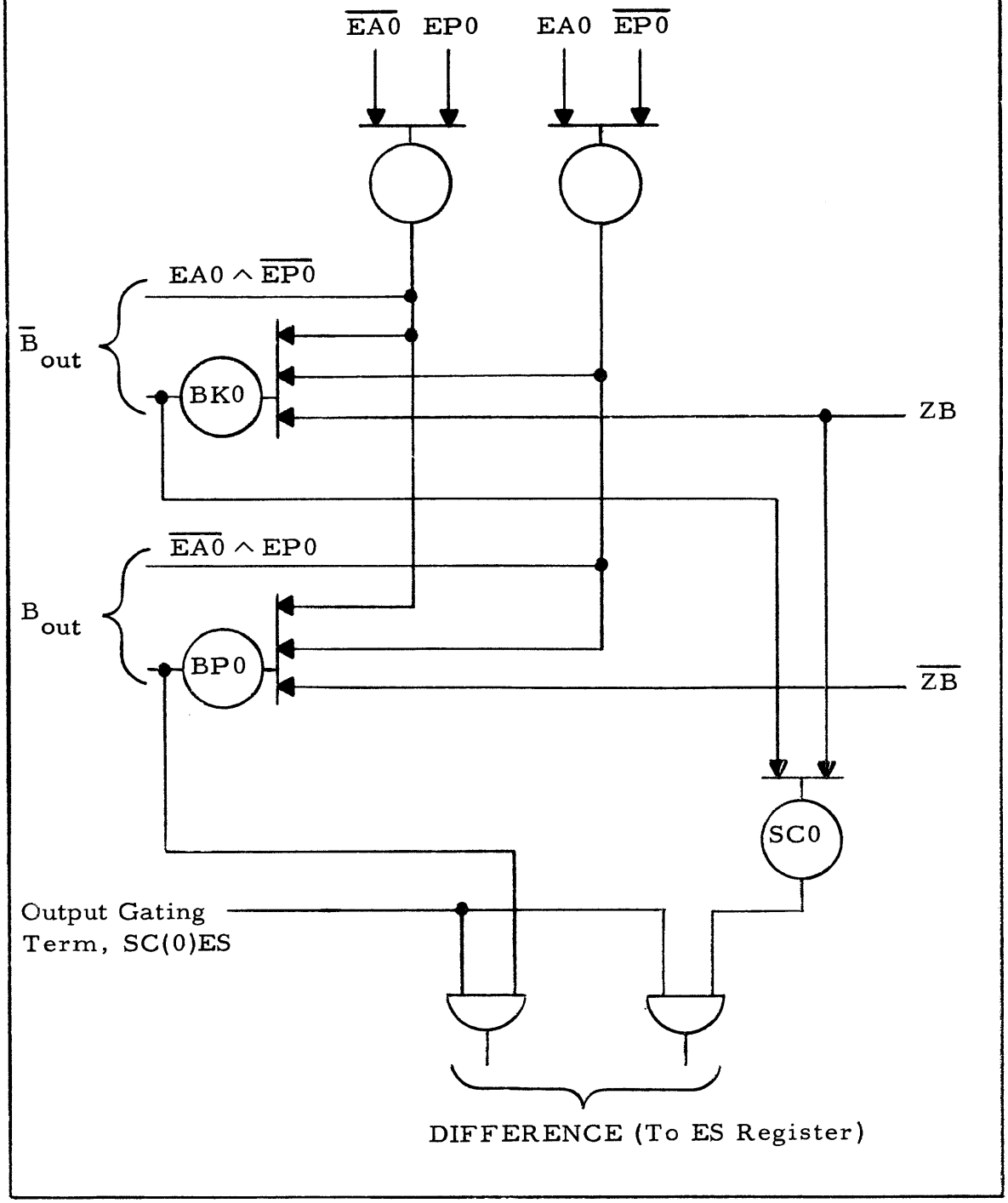
Implications of the borrow in and borrow out terms must be understood. Borrow out from a stage says that the subtrahend of that stage is greater than the minuend and that it must borrow from a subsequent stage (a borrow out to that stage) before the subtraction can occur. Thus, a borrow in depletes one stage by one, but increases the preceding stage by two, since each stage represents a power of two. When both a borrow in and a borrow out condition exist at a stage, the value of that stage is increased by one since a borrow out brings in a value of two, but a borrow in only depletes a stage by a value of one.

Figure 5.4–1 shows that the first stage of the Subtractor is different from succeeding stages only by the fact that the borrow in terms are replaced by the ZB (zero order borrow) signals. The ZB term is generated when an operation is being performed which would cause the result formed by the Subtractor to be one too great when the answer is in <u>true</u> form. It should be remembered that, other than bits actually used to designate sign, if the resultant of the Subtractor operation is negative it ends up in 1's complement form (it must be recomplemented to be in true form) while all positive resultants are in true form. A ZB signal is generated under the following conditions:

1) during multiply and reverse divide when the exponents of the two operands are equal and positive,
2) during multiply, divide, and reverse divide when the contents of EA are positive and the contents of EP are negative,
3) during multiply, divide, and reverse divide when the contents of EA and EP are both positive or both negative and the magnitude of the contents of EP is greater than the magnitude of the contents of EA. Remember that as the absolute value

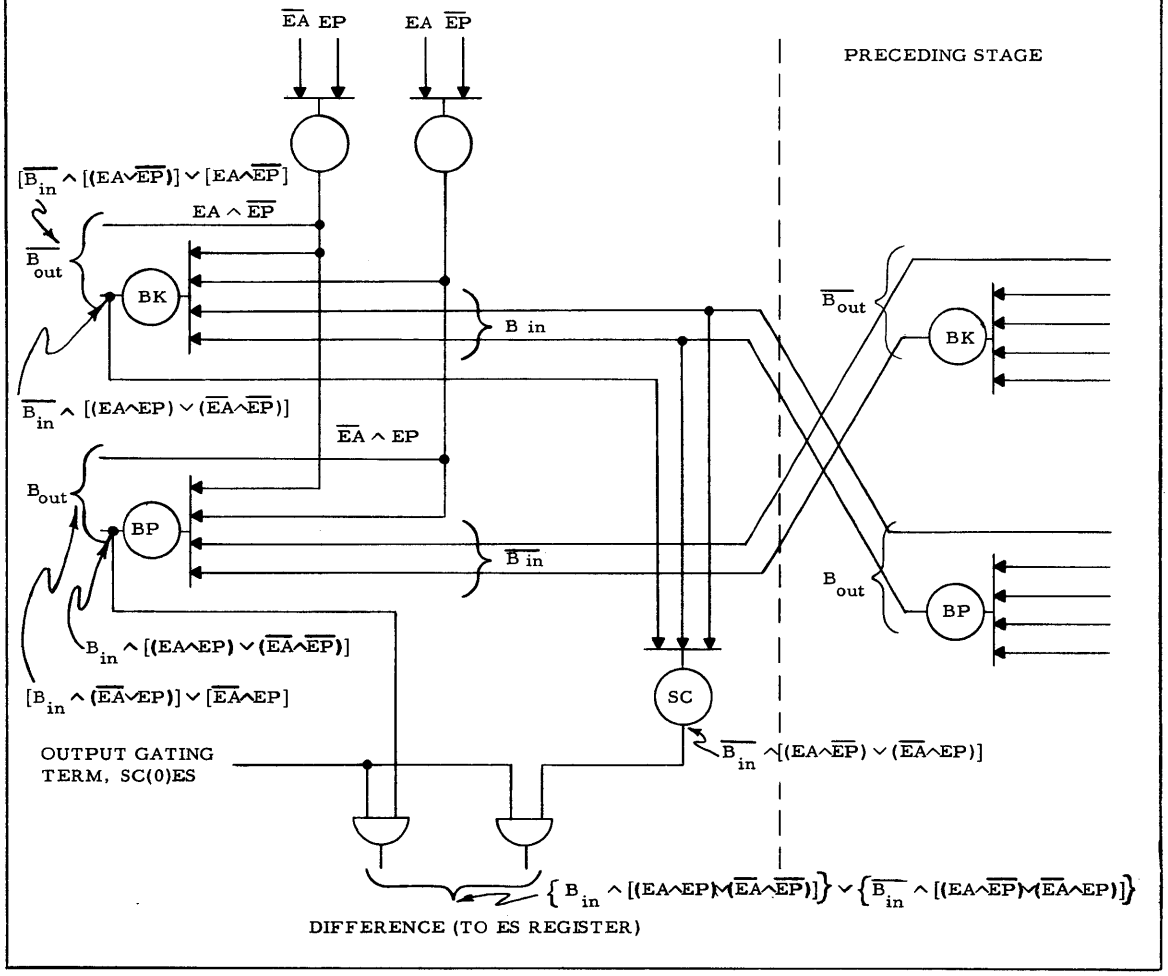
of a negative number gets larger, its real value gets smaller.

FIGURE 5.4–1 Initial Subtractor Stage



The reasons why ZB signals should be generated under the above conditions may not be intuitively obvious and if doubt exists in the readers mind concerning the validity of the above statements, the reader should test them on the basis of how the Central Processor handles the formation of the exponent multiply and divide operations and not by just simple binary addition or subtraction of exponents.

FIGURE 5.4–2 Typical Subtractor Stage



The Comparator. Since all sum and difference operations are performed in floating point, the exponents of the operands must be equalized prior to any combining of the mantissas of the operands. To

facilitate exponent equalization, it is first necessary to determine which exponent is greater in magnitude. The Comparator logic was designed to provide this information. The Comparator circuitry performs two functions in regard to exponent magnitude. First, if the exponents of the two operands are equal it indicates such to the Central Processor logic. Second, if the exponents are unequal it indicates which of the two is algebraically larger. The logic for these two Comparator functions is shown in Figures 5.4–3 and 5.4–4, respectively.

FIGURE 5.4–3 Comparator–Exponents Equal Logic

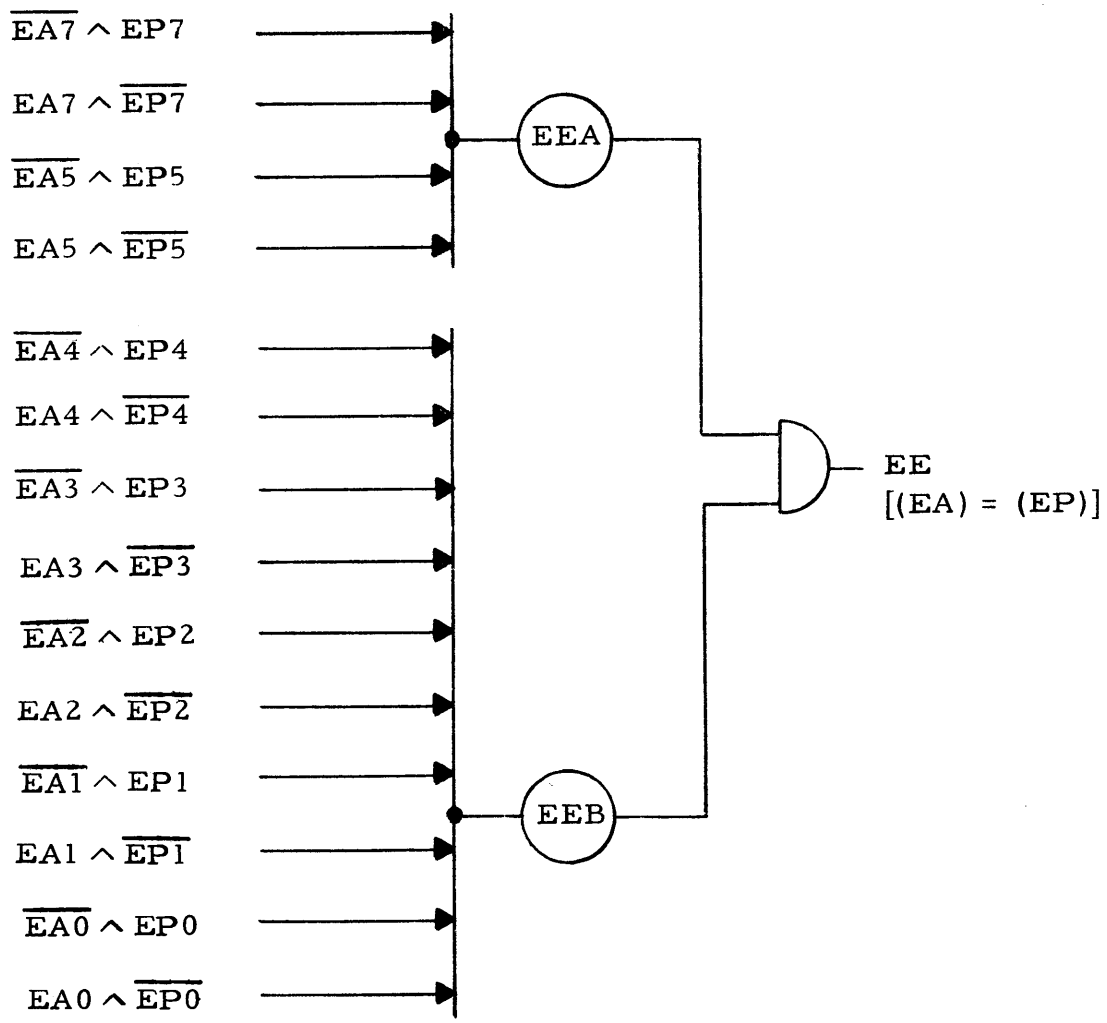
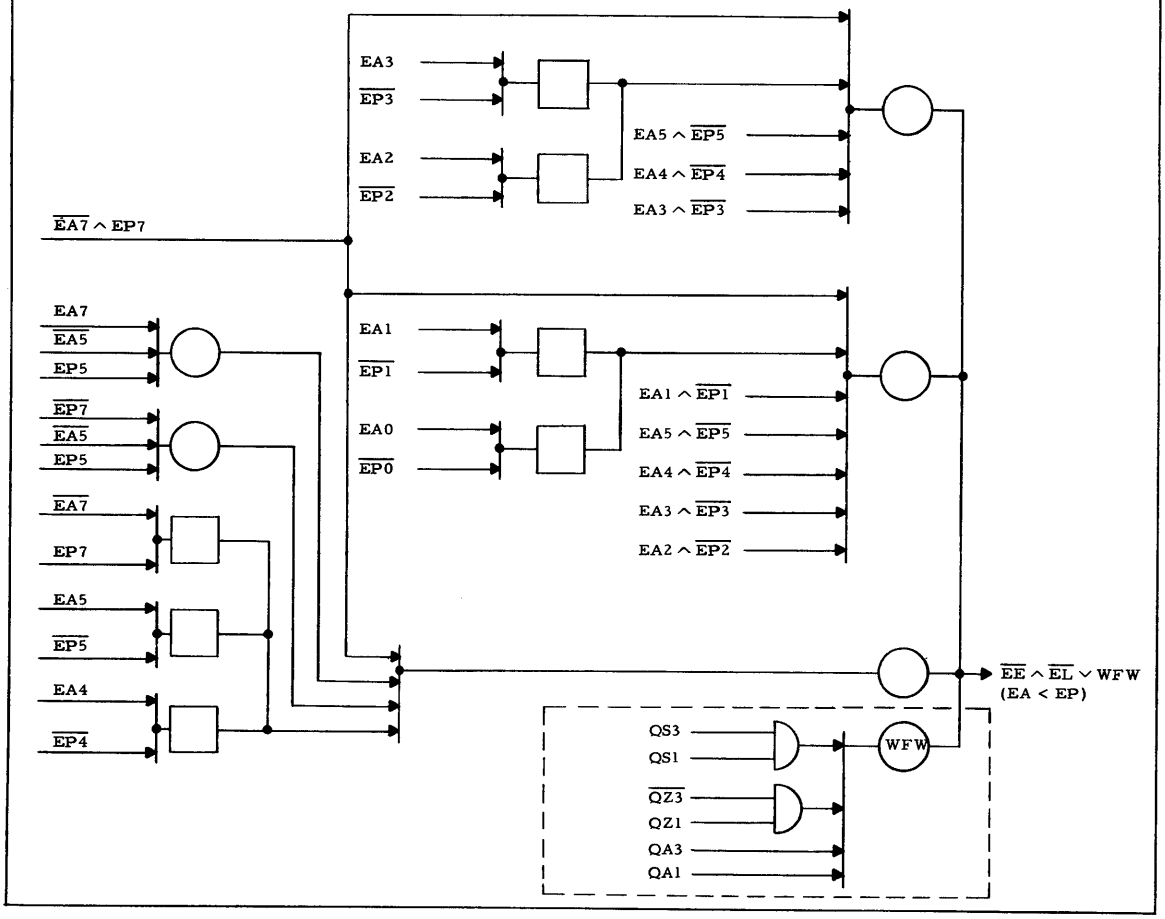


FIGURE 5.4–4 Comparator–Exponents Unequal Logic



Bits EA7 and EP7 contain the signs of the exponents; bits EA6 and EP6 are for the purpose of indicating when either one of the exponents have gone out of the normal range. If exponent under or overflow has not occurred, these bits contain the same sign information as held in bits EA7 and EP7. The remaining 6 least significant bits of the EA and EP registers hold the absolute value of the exponents of the operands (in 1's complement form if the exponent is negative).

During checking for unequal exponents the logic shown in Figure 5.4–4 affords each bit the "priority of decision" that its order rates. If the logic were to decode EA7 ∧ $\overline{EP7}$, it would indicate a plus exponent in

EP and a minus exponent in EA. If this is true, then nothing in the lower orders can be permitted to alter the decision that the exponent in EP is greater than the one in EA. If signs do not decide the magnitude comparison, then the numerical values of the exponents in the 6 least significant bits of the two registers must be compared.

In Figure 5.4—4 there is a section of comparator logic set off by dash—lines that contain the WFW logic symbol. The logic associated with this inverter has nothing to do with the determination of the relative sizes of the exponents. Rather, its purpose is to indicate at a later time (i.e., when the exponents have been shifted until they are equal during the processing of the sum or difference operations) which of the exponents was greater when the unequal condition existed. This information is necessary for the correct processing of the exponents during sum and difference operations.

CHAPTER 6

THE CORE MEMORY SYSTEM

SECTION 6.1 – GENERAL DESCRIPTION

6.1–1 INTRODUCTION TO CENTRAL PROCESSOR CORE MEMORY

The G–20 Central Processor's memory system is composed of planes of high speed, random–access, ferrite cores. Secondary, or bulk memory, is also available in the form of magnetic disc, magnetic tape, paper tape, and punched cards. In this manual, however, we are interested only in the core memory system since it can be considered as being internal to the Central Processor. Actually, of the eight core modules, of 4,096 words per module, that can be provided with a G–20 System, only two memory modules can be physically housed in the Central Processor cabinet due to space limitations. Additional core memory modules are housed in Auxiliary Memory Module (MM–10) cabinets. These MM–10 units hold either one or two additional memory modules per cabinet, depending upon the size of the core memory desired by the customer. It should be remembered, however, that the memory modules are identical, whether mounted internally in the Central Processor or externally in an MM–10. The discussion of core memory presented in this chapter will hold for both internal and external modules. Exceptions caused by the external modules not specifically mentioned in the general description are discussed in Section 6.6.

The MM–10's and the Central Processor are connected by the Memory Interface Communication Line and, from the standpoint of the Central Processor memory location decoding, there need be no distinction between internal and external memory. However, from the standpoint of total execution time and from the standpoint of programming for input/output simultaneous with computing [using the Data Communicator] there are two significant differences.

While the complete memory cycle from the start of next access is 6 microseconds for both internal and external memory, the time from the start of access to availability of the word is different. This time is 3 microseconds for internal memory and 4 microseconds for external memory. Any access to external memory ties up the external memory communication lines for 2 microseconds. Following any access to external memory by one control device, another control device can start an access to another cabinet in 2 microseconds. When a Central Processor has written in memory, it can start an access in another cabinet in 3 microseconds. For all other situations the effective memory cycle is the basic 6 microseconds.

The differences mentioned above result from the following facts: transfer of a word from an external cabinet to the Central Processor requires about 1 microsecond, which accounts for the difference in availability. There is no difference in cycle time because an external module has its own Memory Buffer register, and can start restoring the word before it has reached the Central Processor. For reading from either internal or external memory the time until another memory cycle can begin is always 6 microseconds.

A memory plane consists of a 64 x 64 array of 4,096 magnetic cores with each core representing 1 bit in a memory word. Thirty–three such memory planes constitute a memory stack and form the 33 bits

of the standard Central Processor word. A memory stack and its associated logic and driver packages provide the standard memory module shown in Figure 6.1–1.

6.1–1 G–20 Core Memory Module

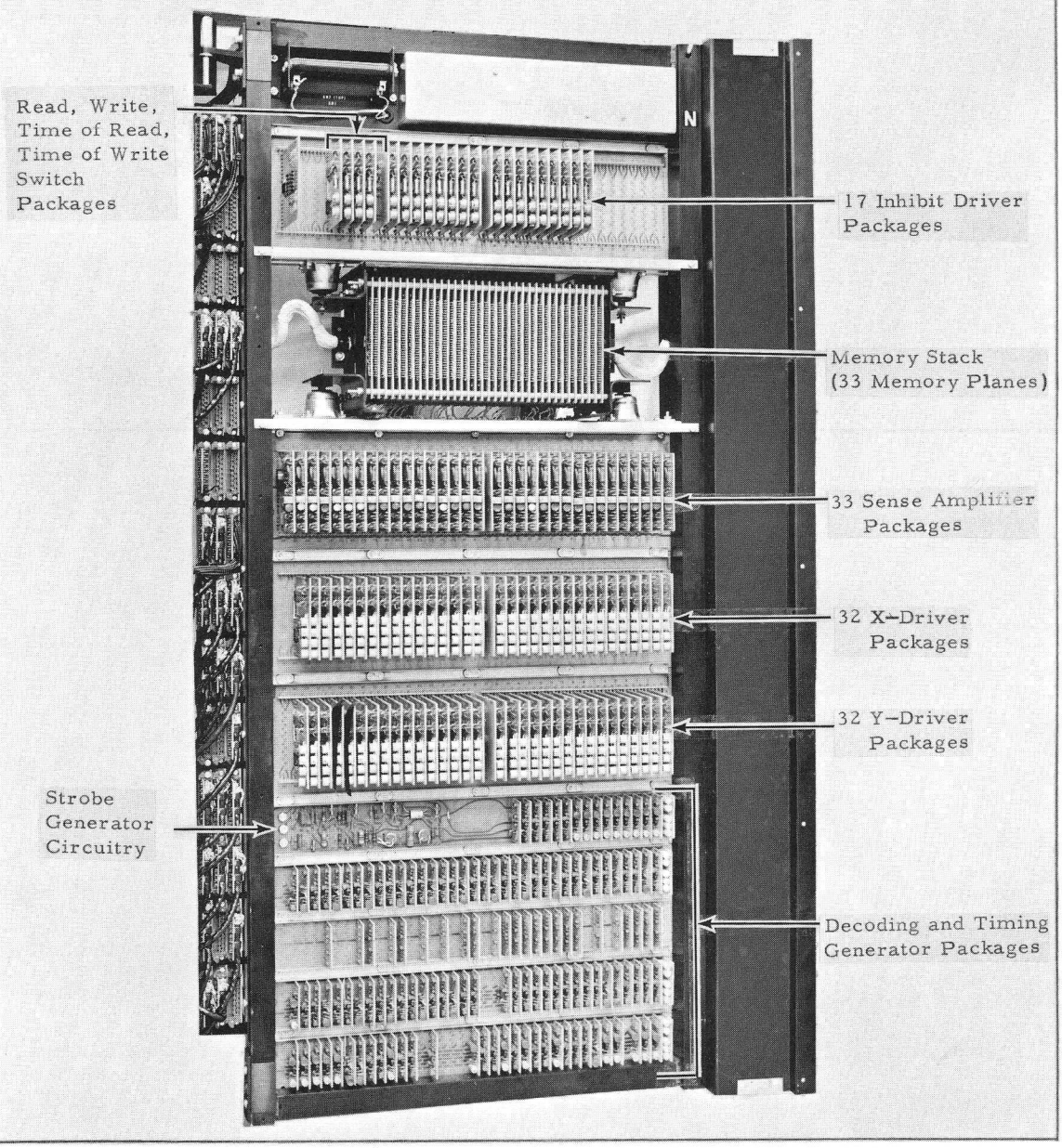


FIGURE 6.1–2 Representative View of a Central Processor Core Memory Stack

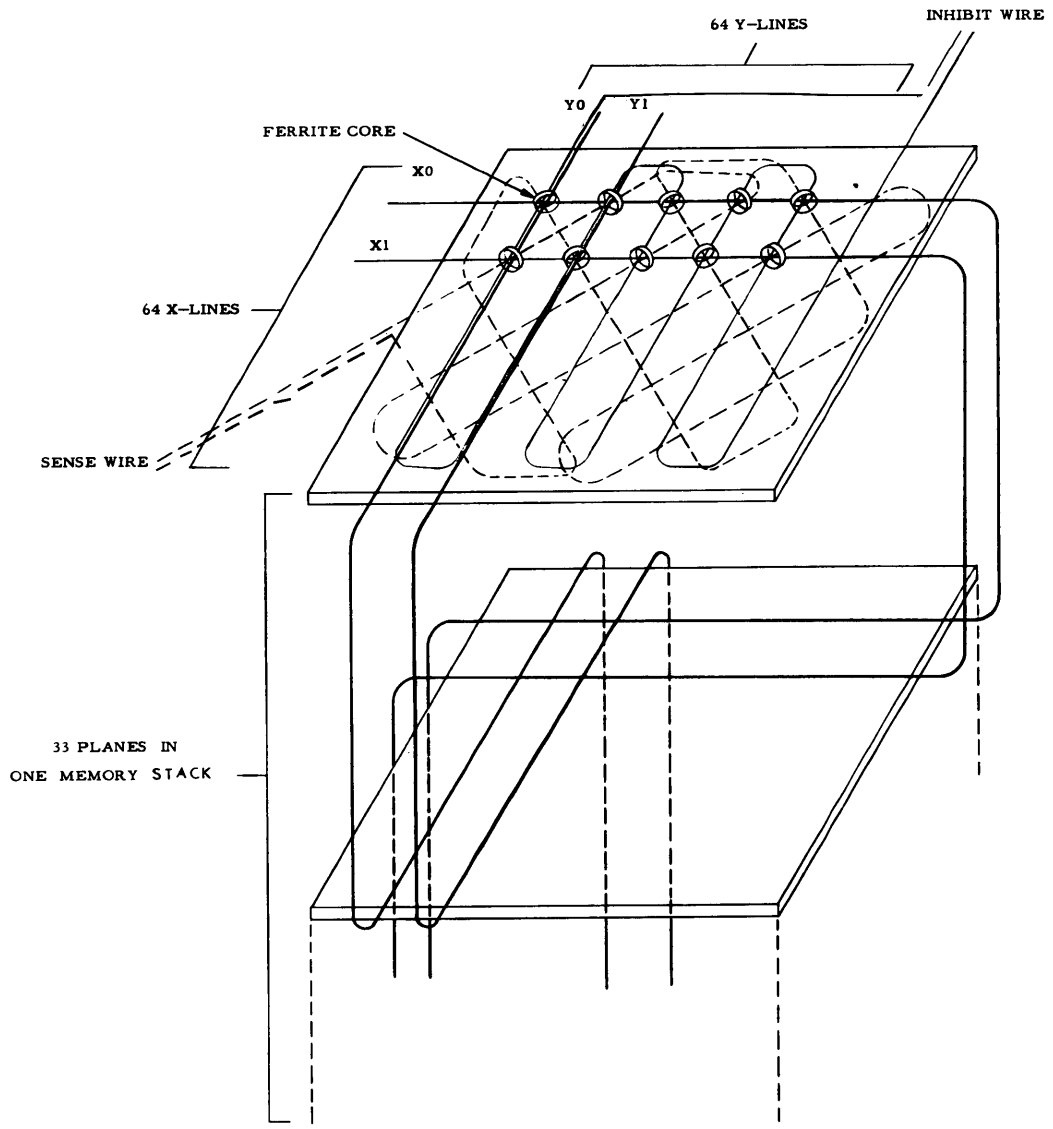
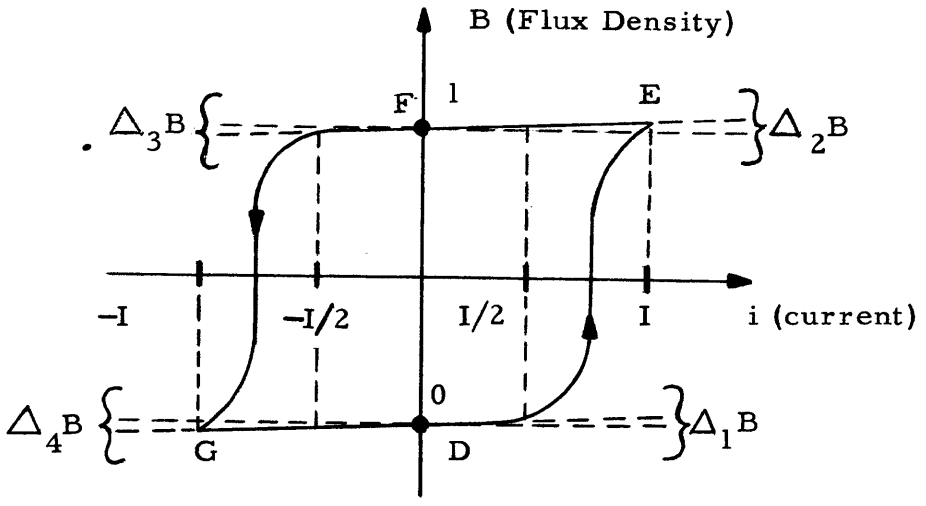


Figure 6.1–2 is a representation of a G–20 core memory stack. This figure shows that each core has an X–address wire, a Y–address wire, an inhibit wire, and a sense wire threading it. Figure 6.1–2 also indicates that each X– and Y–address wire passes through 64 cores per plane and through a total of 33 planes. Sense and inhibit wires, on

the other hand, pass through all 4,096 cores of a memory plane and there are separate sense and inhibit wires for each plane.

6. 1–2 COINCIDENT–CURRENT CORE MEMORY THEORY The G–20 has a coincident–current type of core memory. This means that there must be the prescribed current level in both the X and Y lines that thread a core before the core can be forced to change its state of magnetization. Figure 6. 1–3 shows the magnetization hysteresis loop of the ferrite material of the memory cores. The hysteresis loop merely represents the path that is taken when the core changes from one state of magnetic saturation to the other when the current through the wires threading the core is changed sufficiently. The two stable states of saturation, represented by D and F on the B axis of Figure 6. 1–3, represent the 0 and 1 logic states.

FIGURE 6. 1–3 Ferrite Core Hysteresis Loop



I = amount of current required to switch core.

Assume a core is initially at the magnetic stable state D (logic 0). Introducing a positive current will cause the state of magnetization of the core to follow the hysteresis path to the right. If a current magnitude of I/2 exists, a change of $\Delta_1 B$ occurs. When the current source is removed, the flux density returns to the value at D. If, on the other hand, a current magnitude of I exists, it is enough to force the magnetic state of the core to point **E** of the hysteresis curve. Now when the current is removed, the core moves to a state corresponding to point F of the hysteresis loop and the core has switched (i.e., changed its state of magnetization). If the core is initially at the D state, however, and a current of –I is applied, the degree of saturation will increase slightly to a maximum change of $\Delta_4 B$, but the core will not change its magnetic (i.e., logic) state. Basically the same statements can be made if the core is initially in the logic 1 state (point F of the hysteresis loop) and currents are introduced.

In order to select a particular core location in a coincident–current type memory, one–half of the current required to switch a core's state (I/2) is supplied by the X–wire (X3 for instance). Simultaneously, the same amount of current, in phase, is delivered to one Y–wire (Y4, for instance). The one core in a plane that receives the effect of both forces (I/2 + I/2) is the core at the coordinates X3, Y4. The 63 other cores on wire X3 and the other 63 on Y4 all receive only I/2. It was shown in Figure 6.1–3 that this amount of current has a negligible effect on the magnetic state on these cores.

If, as was discussed previously, the total value of current I is of the correct polarity (i.e., direction of flow), then the state of the core will be switched. If I is not of the correct polarity, then the magnetic state of the core is not noticeably affected.

Through the use of the X– and Y–wires, then, the desired <u>word</u> in

memory can be accessed since each X– and Y–wire threads through corresponding cores in all 33 planes of a G–20 memory module. To read out the contents of a memory word, all that is necessary is to establish the polarity of the current I from the X– and Y–drivers so that, during a read operation, all bits of the selected memory word are driven to a known magnetic state (i. e., logic level). In the case of the G–20 Core Memory System, the cores are forced to the 0 logic level. Using the sense wires associated with each plane, it is possible to detect which bits of the word switched state by the induced voltages that occur in the sense wires when the magnetic field of a core changes. Reference to Figure 6. 1–2 shows that the sense wires of a G–20 memory plane follow a rather unusual pattern. This is done to minimize noise caused in the sense wires due to the change in magnetization level, ΔB, of the other 126 cores of a memory plane that are fed by the I/2 currents of the X– and Y–drivers. If such precautions were not observed, the accumulative noise generated by the cores might become large enough to be sensed, indicating a core change of state when that might not necessarily be the case.

To write into this type of memory, all that is necessary is to again try to force all bits to a predetermined logic level (i. e., to logic 1 in the G–20 System). This time, however, it is desired to place a particular configuration of 1's and 0's into the 33 cores that constitute a memory word. Since a read operation always precedes a write operation when processing a memory location in the G–20, the cores of the desired word are all forced to the zero state before a write operation is initiated. Therefore, if a means of inhibiting the forcing of desired memory bits to the logic 1 state level is available, the desired word configuration can be read into memory. To accomplish this, each memory plane has its own inhibit wire which passes through all of the cores in that particular plane. In the G–20 System these inhibit wires are controlled by the outputs of the B register flip–flops. If a bit in

the B register contains a 1, no current is passed through the inhibit wire controlled by that bit.

The plane through which this inhibit wire passes will have its selected core switched to the logic 1 state by the currents I/2 flowing in the X– and Y–lines during the write operation. If, however, a B register bit contains a 0, the inhibit wire controlled by it will have a current of I/2 through it in a direction such that its generated magnetic field is opposite to the magnetic field supplied by the X– and Y–currents. The net magnetizing force is therefore reduced by one–half (I/2 + I/2 – I/2) which is enough to prevent switching of the core and, thus, the core will remain in the 0 state.

If new information is not to be stored in the accessed memory location, then when the write portion of a read–write cycle is started, it will write the word that was read from memory into the B register back into the same memory location. This is always necessary since our core memory is of the destructive read type.

SECTION 6.2 – ADDRESS DECODING

To address a word in memory, one core at the same coordinates in each of the 33 memory planes must be addressed. In each memory module there are 64 X–wires and 64 Y–wires which pass through 64 cores in each of the 33 planes. The desired memory location is addressed by these wires.

When a memory cycle is initiated, the address of the desired memory location is transferred from the CA or BA registers into the MA register. The outputs of bits 11 through 0 of the MA register are sent unclocked to address decoding matrices. A block representation of this decoding is shown in Figure 6.2–1 and detailed presentations of the actual X– and Y–axis decoding matrices of any of the memory modules are presented in Figures 6.2–2 and 6.2–3. Since each memory module is independent of other modules as far as decoding, timing, and driver circuitry are concerned, it is necessary to have a means of determining in which module the desired word is located. The 3 most significant bits of the MA register provide this function. This is shown in Table 6.2–1. When a memory start signal is generated, both memory cycle counters of the memories housed inside the Central Processor are started (assuming that the Central Processor contains two memory modules). Logic decoding then halts the memory cycle timing counter of the internal memory module (or modules in the case of a memory access in an MM–10 unit) that was not accessed.

If, however, the memory location is $20,000_8$ or greater, then the memory cycle counters of internal memory modules as well as the memory modules housed in the accessed MM–10 cabinet are started. The memory cycle counters of the internal memory modules are halted by logic decoding and the decoding of bit MA12 determines which of the

FIGURE 6.2–1 Core Memory Addressing

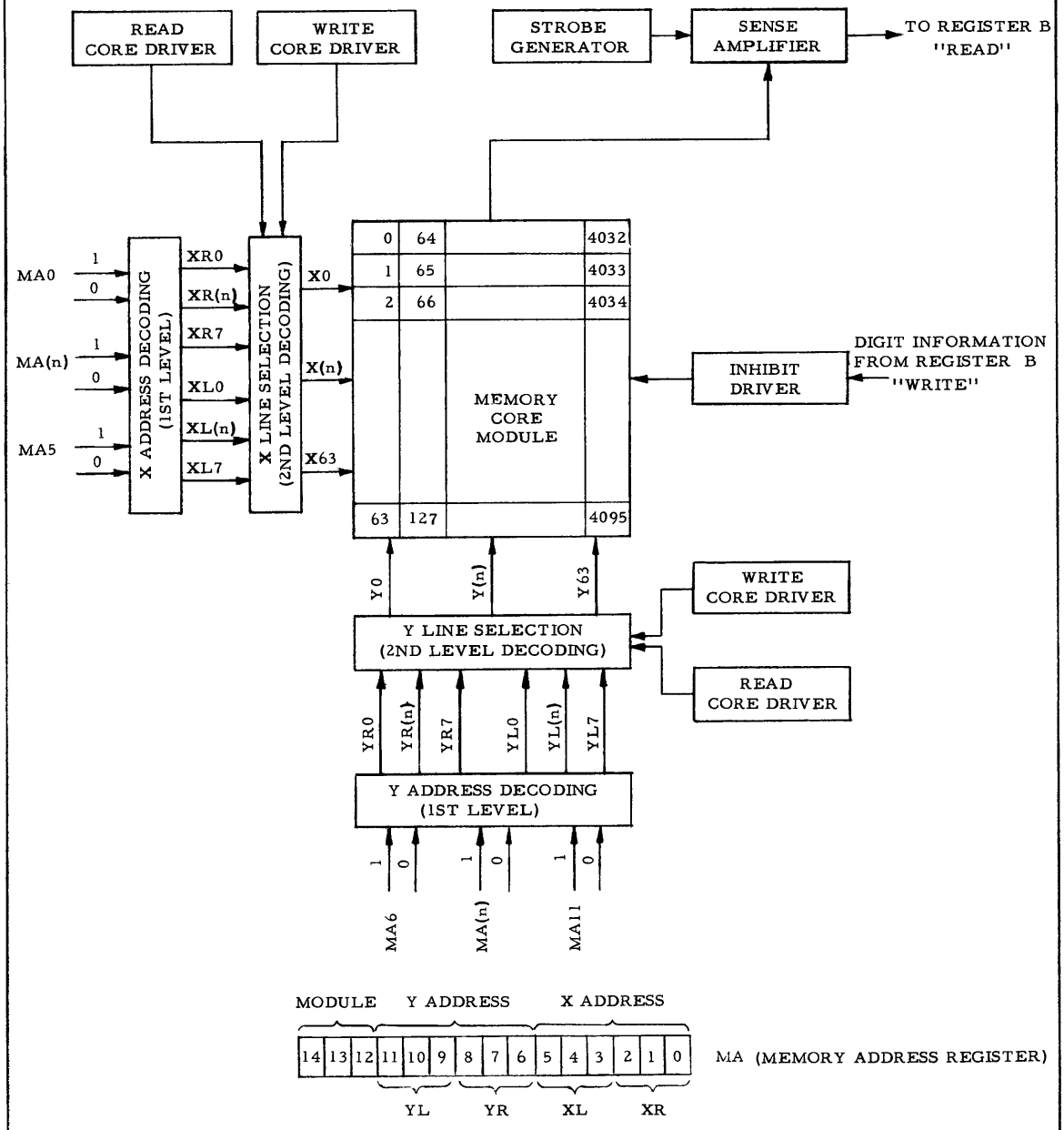


TABLE 6.2–1 Memory Module Decoding

Memory Module	Count in 14M12 (to base 8)	Maximum Address in Module 11MA9	8MA6	5MA3	2MA0	(to base 8)
1st	0	7	7	7	7	4,096
2nd	1	7	7	7	7	8,192
3rd	2	7	7	7	7	12,288
4th	3	7	7	7	7	16,384
5th	4	7	7	7	7	20,480
6th	5	7	7	7	7	24,576
7th	6	7	7	7	7	28,672
8th	7	7	7	7	7	32,768

external memory module memory cycle counters is allowed to continue and the other external memory cycle counter is stopped (assuming two memory modules per MM–10).

Figure 6.2–2 shows the X–axis decoding network. This figure corresponds to both the 1st and 2nd levels of decoding indicated in Figure 6.2–1. The three input OR–gates feeding the inverters and the emitter–followers are considered to be the first stage while the second stage is comprised of the two input AND–gates fed by the emitter–followers. By these two stages, any of the 4,096 or $1,000_8$, words of a memory stack can be addressed by the X–current drivers. The current drivers are actuated by the two input AND–gates of the second stage. The AND–gates of the second stage are really six input gates due to inverter logic that precedes them. Six inputs are all that are necessary to define any X row since $2^6 = 64$.

What has been said for the X–axis address decoding can be repeated for the Y–axis address decoding. The one difference between the two is that, while the X–wires address from 0 to 63_{10} in increments of 1, the Y–wires address from 0 to $4,096_{10}$ in increments of 64_{10}.

FIGURE 6.2–2 X–Axis Decoding Matrix

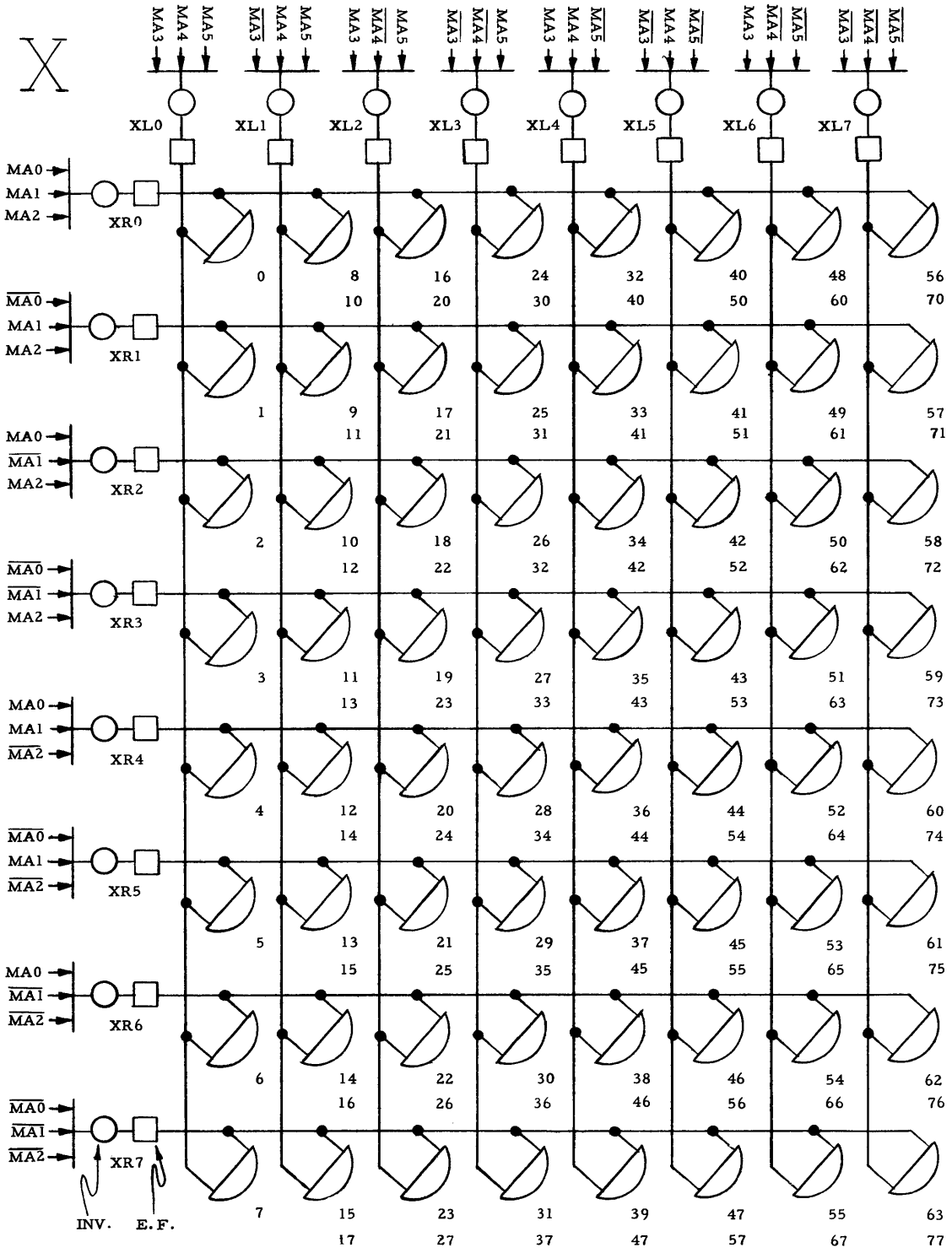
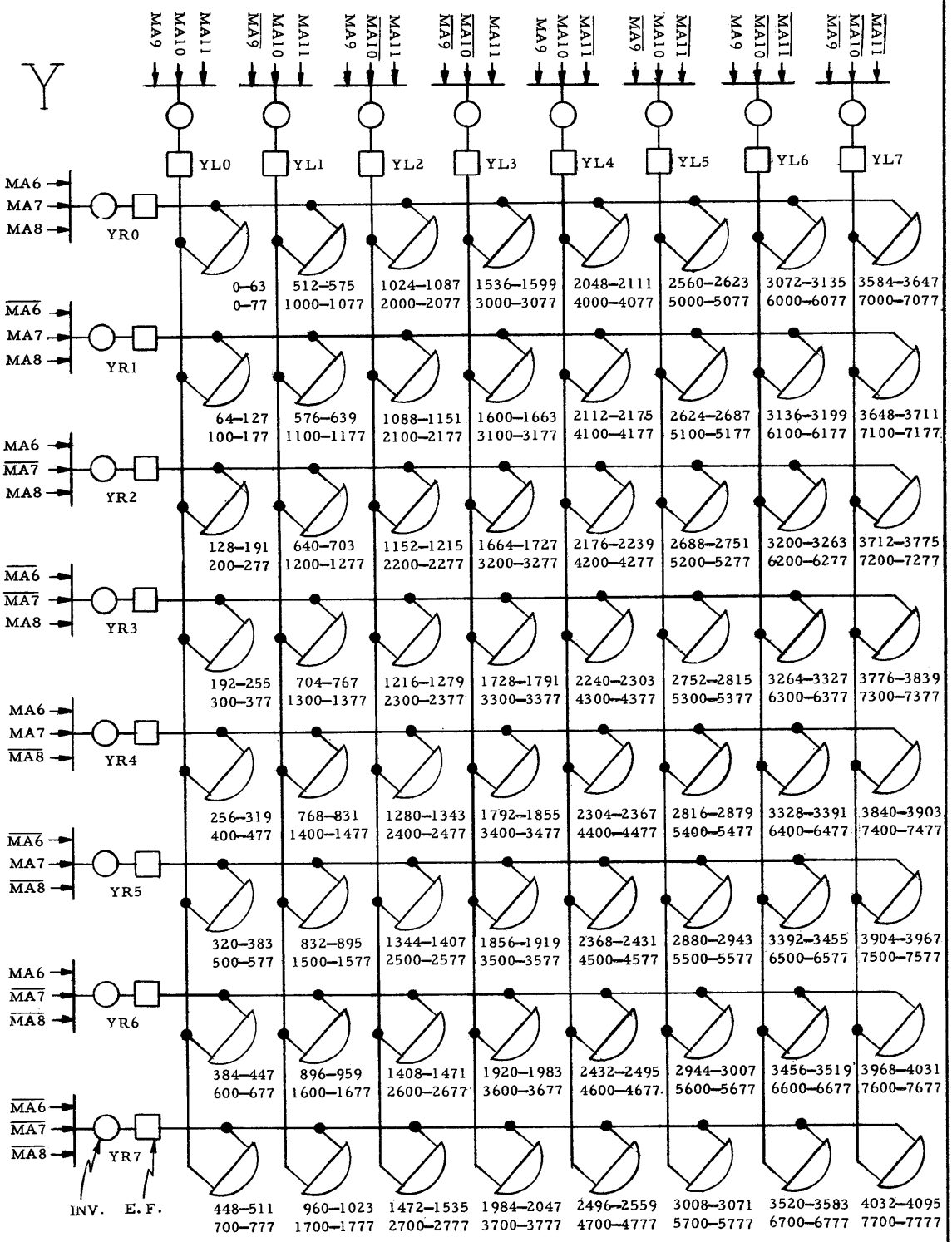


FIGURE 6.2–3 Y–Axis Decoding Matrix



SECTION 6.3 – MEMORY CIRCUITS

6.3–1 MEMORY DRIVERS The G–20 Core Memory System uses two types of driver packages, the inhibit driver and the X–Y driver. The inhibit driver package also is used for the Read and Write current switches and the Time–of–Read and Time–of–Write current switches. The inhibit driver is shown in Figure 6.3–1 and the X–Y driver is shown in Figure 6.3–2.

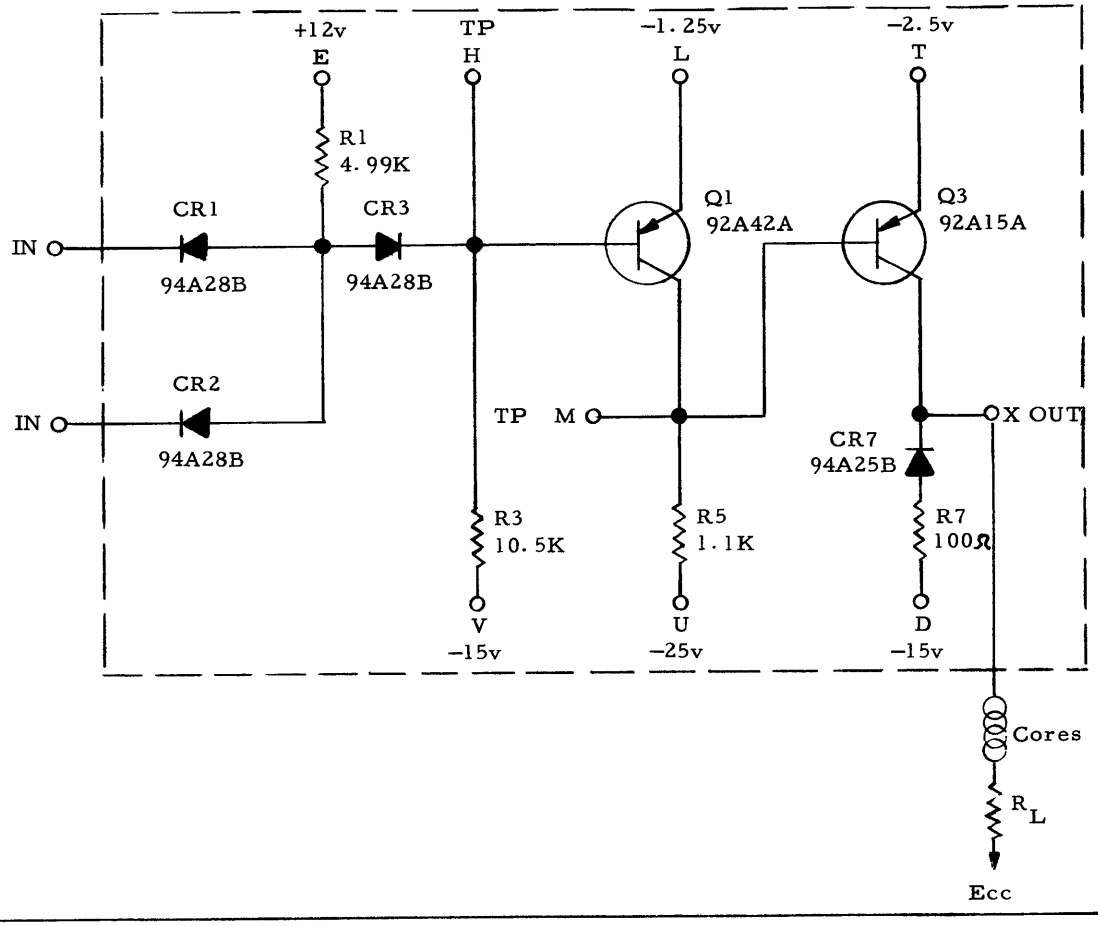
It will be recalled that when a new word is to be written into the memory, all the bits of the designated memory location are in the zero state, and the new word, which is held in the B register in digital form, is stored into the memory through the action of the inhibit and X–Y drivers. The new word contains information coded into a series of 1's and 0's. In each memory plane, current is applied to the X and Y lines through the X–Y drivers.

The inhibit driver supplies a current pulse of I/2 to the memory inhibit wires when a zero is to be stored in a memory bit. The current from the inhibit driver is in opposite direction to the X–Y driver currents during the write time of a memory cycle. This reduces the effective value of the core current to half, which is not enough to switch the core to the logic 1 state.

The Inhibit Driver. From Figure 6.3–1, it is seen that the inhibit driver consists of two cascade amplifier stages fed by a two input AND–gate. The two amplifier stages are necessary to provide the current necessary for the inhibit wire. The first term of the input AND–gate supplies the digit information and is fed by false side of the corresponding B register flip–flop. This input is high (0 volts) if a logic 0 exists in the corresponding bit of the B register flip–flop and low (–3.5 volts) if a logic 1 exists. The second term of the input

AND–gate supplies the timing information from the memory cycle timing generator. This input (WM) is high during the write portion of a memory cycle; at all other times, this input pulse is low.

FIGURE 6.3–1 Inhibit Driver Circuit



When both inputs to the AND–gate are high (0 volts) the output is also high, and the voltage at the base of Q1 is approximately zero volts; Q1 is back–biased and therefore cut off. When Q1 is off, the voltage at the emitter of Q3 (–2.5 volts) is more positive than the voltage at the base of Q3 and, therefore, Q3 conducts. When Q3 is conducting, its base is at approximately –3.5 volts (–2.5 supply voltage plus a –1.0

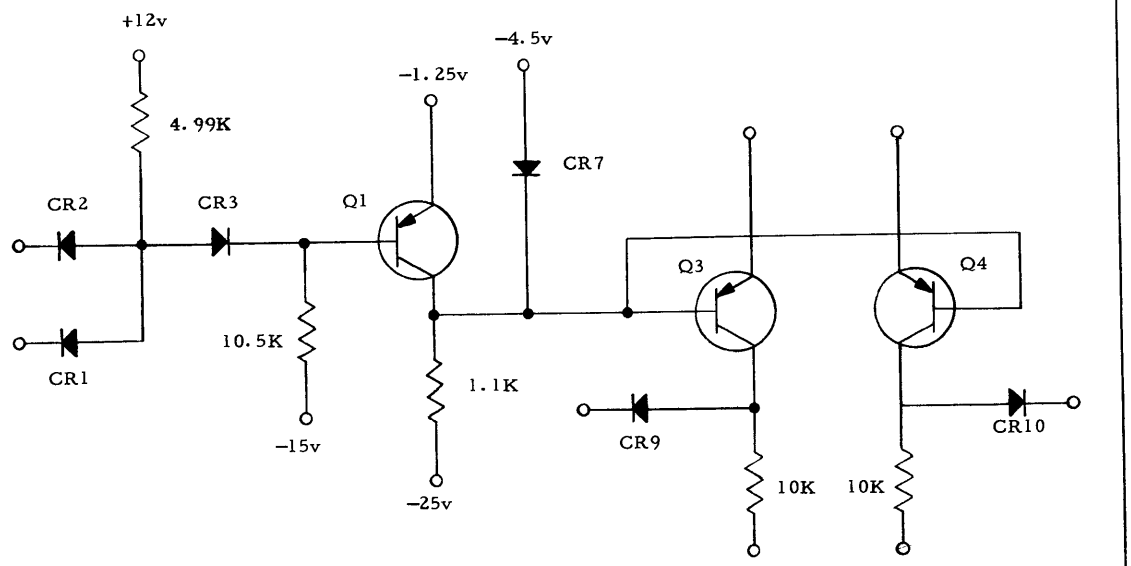
volt drop between the emitter and base of Q3). With Q3 in full conduction, the inhibit current of I/2 is supplied to the inhibit wire of the memory plane. This will cause a 0 to be stored in that particular bit of the memory word.

When the $\overline{\text{Bnn}}$ input to the AND–gate is low (–3.5 volts) indicating a 1 in that bit of the B register, and the WM input is high (0 volts), the output of the AND–gate is low (approximately –3.2 volts), and the voltage at the base of Q1 is approximately –1.55 volts (–1.25 supply voltage plus a –0.3 volt drop between the emitter and base of Q1). Therefore, Q1 is forward–biased and turned on. When Q1 is on, the voltage at the base of Q3 is approximately –1.45 volts (–1.25 supply voltage plus a –0.2 volt drop between the emitter and collector of Q1) and Q3 is turned off. With Q3 off, there is no inhibit current supplied to the inhibit winding of the memory plane. In this case, a 1 is stored in that particular bit of the memory word.

The collector load on Q3 is composed of the inhibit winding of the memory plane and Resistor R_L. The exact inhibit current, I/2, is regulated to 1 per cent by the collector supply voltage Ecc and the load Resistor R_L. Ecc is always the –15 volt regulated supply, but R_L will vary, depending upon the commercial supplier of the memory stack.

The X–Y driver circuit is shown in Figure 6.3–2. The X–Y drivers receive their input information from the first level of the address decoding matrix. The AND–gate input to Q1 is the same two input gate designated as the second level of address decoding in Figures 6.2–2 and 6.2–3. Their basic operation is to direct the read–write current pulses into the proper X or Y core line. To present a complete picture of the operation of the X–Y driver circuits, it is necessary to examine the electronics of the read–write cycles in detail. This is done in Section 6.4.

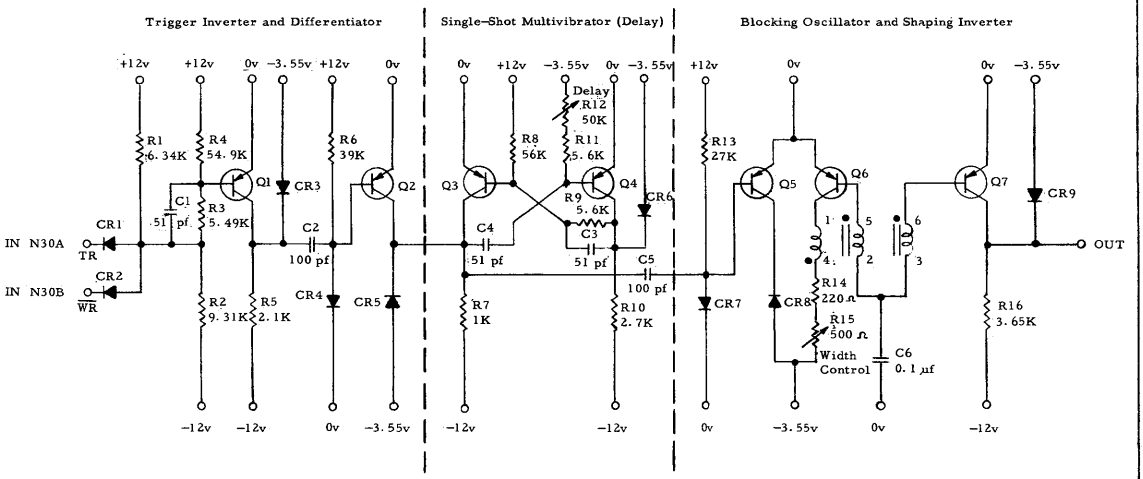
FIGURE 6.3–2 X–Y Driver Circuit



6.3–2 MEMORY STROBE GENERATOR The purpose of the strobe generator is to provide a pulse at Time–of–Read during a Read–Write or a Delay–Write memory cycle. The pulse output of the strobe generator is used to enable the output from the sense amplifiers. This allows logic 1 states detected by the sense amplifiers to be read into the B register. During Clear–Write cycles, however, the output of the strobe generator is inhibited. This action prevents logic 1 states detected by the sense amplifiers from being read into the B register and thus the bits of the B register remain in their previously cleared state.

The circuit diagram for the strobe generator is shown in Figure 6.3–3. The strobe generator consists of two basic sections: a triggered one–shot multivibrator and a triggered blocking oscillator. The multi–vibrator provides a delay before the blocking oscillator is triggered. The pulse which sets the strobe generator into operation is applied

FIGURE 6. 3–3 Strobe Generator Circuit



CIRC. SYM.	PART NO.
CR 1, 2, 3, 5, 6, 8, 9	94A28B
CR 4, 7	94A31A
Q 1, 3, 5, 6	92A41A
Q 2	92A43A
Q 4, 7	92A45A

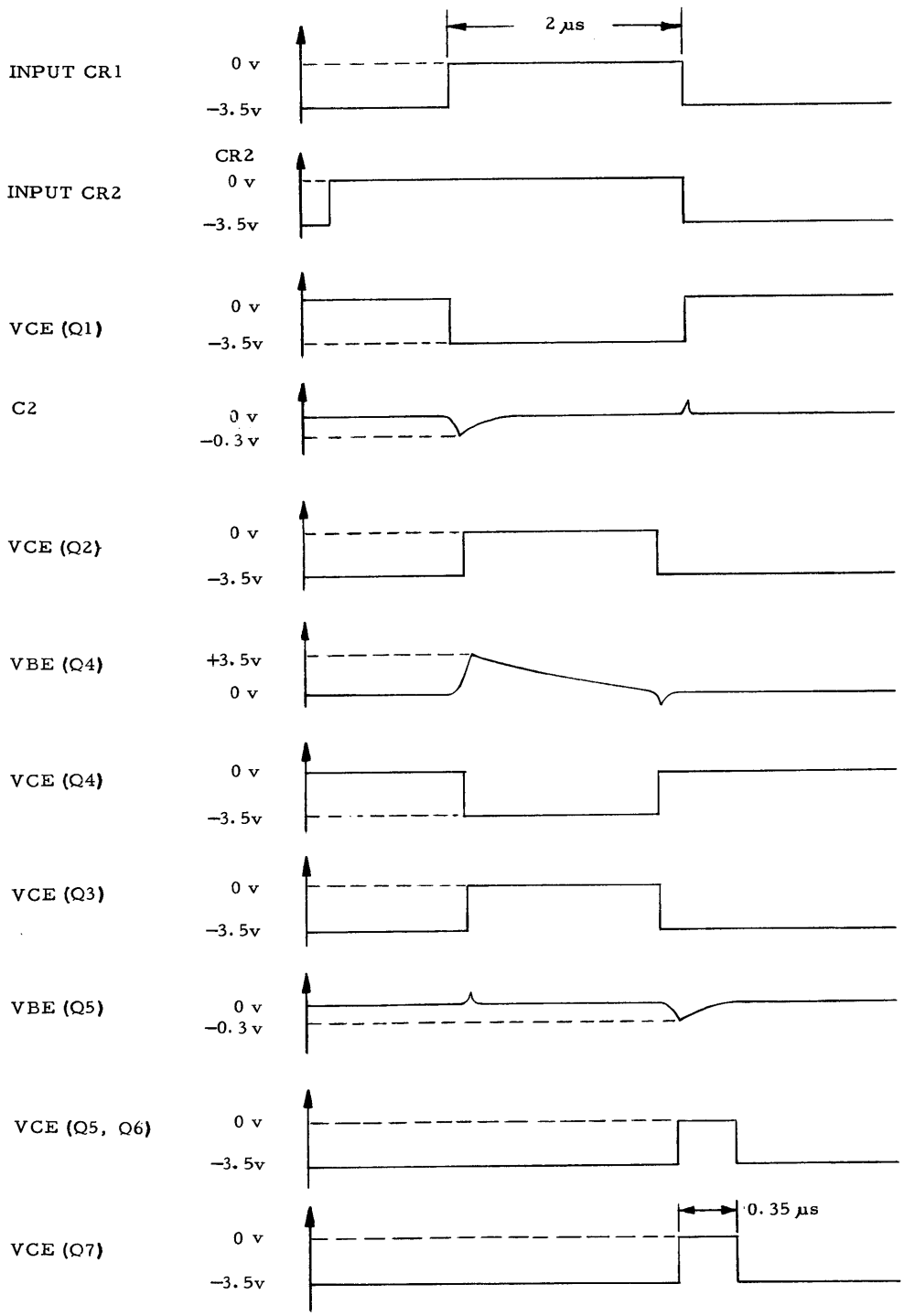
All +12v Supply Come From Terminal "N1H"
" 0v " " " " "N1K"
" −3.55v " " " " "N30C"
" −12v " " " " "N1M"

through CR1 and CR2, the combination of which is a two–term AND–gate. CR1 is supplied from the TRUE side of the Time–of–Read flip–flop (TR) while CR2 is supplied by the false side of the write flip–flop ($\overline{WR}$). Both signals are high during the read portion of a Read–Write or a Delay–Write memory cycle. During a Clear–Write cycle, however, the $\overline{WR}$ signal is low. (The WR flip–flop is controlled by the sequencer logic and $\overline{WR}$ is maintained in a low state during putaway or mark transfer operations.)

Under quiescent conditions, Q4 of Figure 6. 3–3 is normally conducting while Q2 and Q3, which are in parallel, are cut off.

When both terms of the AND–gate come high, the output of Q1 goes low. This negative–going pulse is differentiated by the RC network made up of C2 and R6 in parallel with the input impedance of Q2.

FIGURE 6. 3–4 Timing Diagram of Strobe Generator



The negative spike from the differentiating network turns Q2 on; the positive–going collector voltage is coupled through C4 to the base of Q4. As this positive–going voltage starts to turn Q4 off, its collector potential goes negative. This negative–going pulse is coupled to the base of Q3, turning Q3 on. The input trigger pulse to Q2 is then terminated and Q2 returns to its cut–off condition. The collectors of Q2 and Q3 remain at 0 volts since Q3 is conducting. Q3 will remain in the conducting state, and Q4 remains cut off until the charge on C4 leaks off through R11 and R12. When this charge leaks off and the base of Q4 goes slightly negative, Q4 goes into conduction and Q3 is turned off, which allows the collectors of Q2 and Q3 to go negative. This negative pulse is coupled through Q5 to the base of Q5, and triggers the blocking oscillator consisting of Q5, Q6 and pulse transformer T1. This produces a positive pulse at the output of Q7.

The pulse width is determined by the primary inductance of the transformer and the collector current of Q5 and Q6 which is allowed to flow. The value of this current is determined by the load resistance, R14 and R15. The delay of the output pulse from the input trigger pulse is determined by the time constant of C4, R11 and R12. The strobe pulse generator, as described, gives a positive pulse output which is variable in width and has, as well, a variable delay from the input pulse. This delay feature is necessary to ensure that the sense amplifiers receive the strobe pulse at the correct time. Figure 6. 3–4 shows the timing diagram of the strobe generator.

6. 3–3 SENSE AMPLIFIER The sense amplifier consists of a Class A difference amplifier followed by two inverters, the final inverter being gated by $\overline{\text{Strobe}}$. The sense amplifier accepts the low–level signals from the core line and amplifies these signals to bring them up to computer logic level.

The output of the core matrix consists of either positive or negative signals which will have a nominal amplitude of 50 millivolts. The sense amplifier, SA, consists of three stages of amplification. The input stage is a stabilized Class A difference amplifier, while the final two stages are operated in saturation or cutoff to give an output of –3.5 volts if a logic 0 is read and 0 volts if a logic 1 is read.

In a difference amplifier, the output voltage is proportional to the difference of two voltages supplied to its two separate inputs. Two transistors are necessary to perform the comparison operation. Actually, the difference amplifier consists of two Class A amplifiers. Figure 6.3–5 shows the difference amplifier portion of the sense amplifier.

Ideally, if the input signals were identical in a difference amplifier, the output signal would be zero. An identical signal on both inputs is known as common–mode signal. The difference of any two arbitrary input signals may be expressed in terms of a difference signal. In the sense amplifier, the input stage Class A difference amplifier has an a.c. voltage gain of approximately 35 for a difference signal input and a d.c. gain of approximately 0 for a common–mode signal input.

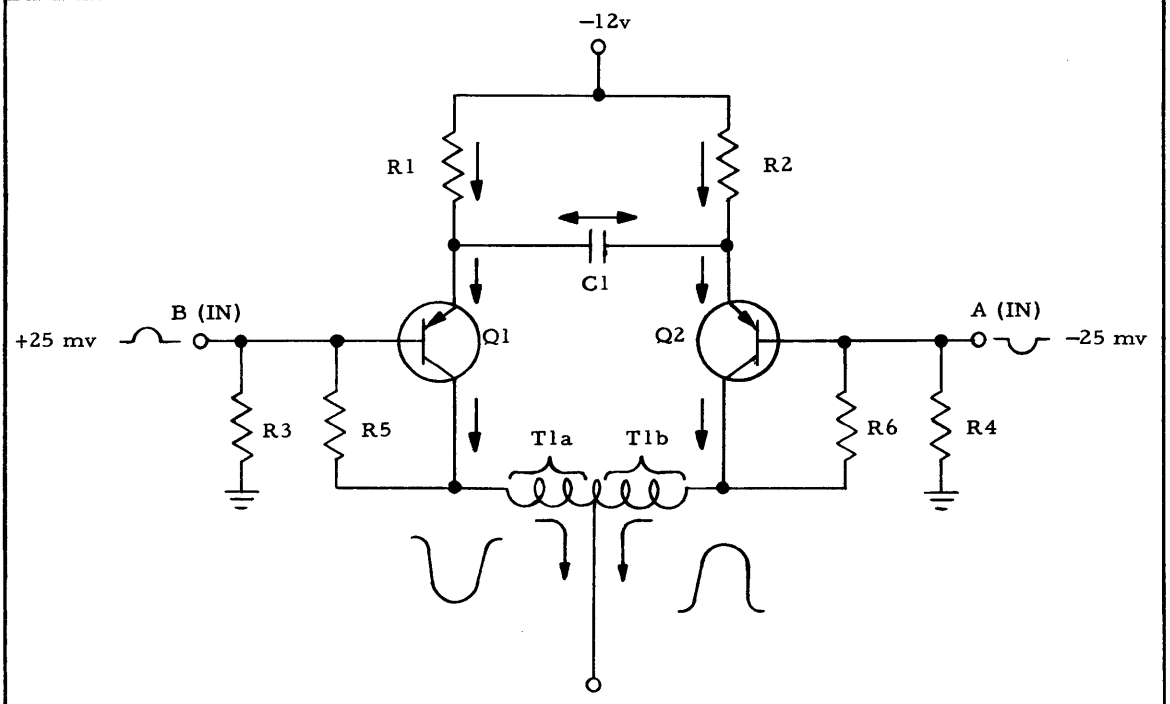
The d.c. operating point of the difference amplifier is stabilized by the large emitter Resistors, R1 and R2. A change in the collector current following through R1 causes the voltage developed across R1 to change. The voltage change developed in this way changes the base current so as to stabilize the collector current. The difference amplifier a.c. gain is stabilized by the use of voltage feedback. This feedback is introduced by R5, from the base to the collector of Q1, and by R6, from the base to the collector of Q2. The resistors in the base circuit, R3 and R4, serve as the terminating resistors for the sense winding and also as a return for the base current. The difference amplifier is ENABLED by

the Not Time–of–Read signal, $\overline{TR}$, at the center tap of the primary of the transformer. The output of the difference amplifier is at zero volts at all times, except during the time the memory is being read (i.e., when ENABLE, $\overline{TR}$, is at –3.5 volts). Referring to Figure 6.3–5, the common–mode input signal indicates that the signal on the base of Q1 (point B) is in phase with the signal on the base of Q2 (point A). (Such might be the case for noise picked up on the sense line.) To analyze the common–mode rejection of a difference amplifier, we shall consider a positive signal, of equal amplitude, being applied to each base of the amplifier. These positive–going signals also appear at the emitters of the difference amplifier. Since both emitter voltages rise together, there is no change in potential across the capacitor. Therefore, there is no current flow to or from the capacitor. The only change in current that can occur is through the emitter resistors. This decrease in current due to the rise in emitter voltage causes degeneration to occur within the amplifier and, hence, the gain of the amplifier is greatly reduced. There is also a corresponding decrease in collector current. Since each collector is tied to one half of a center tapped transformer, the effect of this change of current is cancelled within the windings of the transformer; thus, the gain of the amplifier is further reduced. Therefore, there are two reasons for the reduced gain of a difference amplifier when a common–mode signal is applied to its input terminals: (1) the large amount of degeneration due to the input signal appearing across the emitter resistors of the difference amplifier, (2) the cancellation effect of the current changes within the transformer.

Difference signal input defines the normal sense amplifier input, i.e., Q1 and Q2 base input signals are 180 degrees out–of–phase. The current and voltage curves in one amplifier are the mirror images of those in the other amplifier. Figure 6.3–5 shows the difference amplifier portion of the sense amplifier operating in the difference mode. Figure 6.3–7 shows the timing diagram of the sense amplifier with an

output of 1 (i. e., when a 1 is being read out of the memory).

FIGURE 6. 3–5 Difference Amplifier Circuit, Difference Operation



With reference to Figure 6. 3–5, the difference mode operation of the difference amplifier may be analyzed. Assume that the amplifier is enabled, i. e., –3. 5 volts applied to the center tap of the transformer, and that there is a positive signal applied to the base of Q1 from B and a negative signal applied to the base of Q2 from A. Under these conditions, there will be an amplified positive signal at the collector of Q2. These conditions will cause a decrease in current flow through Q1 and half of the transformer T1a, and an increase in current flow through Q2 and the other half of the transformer T1b. The currents in R1 and R2 will remain constant since any change in these currents would introduce degeneration within the amplifier and, hence, reduce the gain of the amplifier. Since the currents in R1 and R2 must remain constant, the

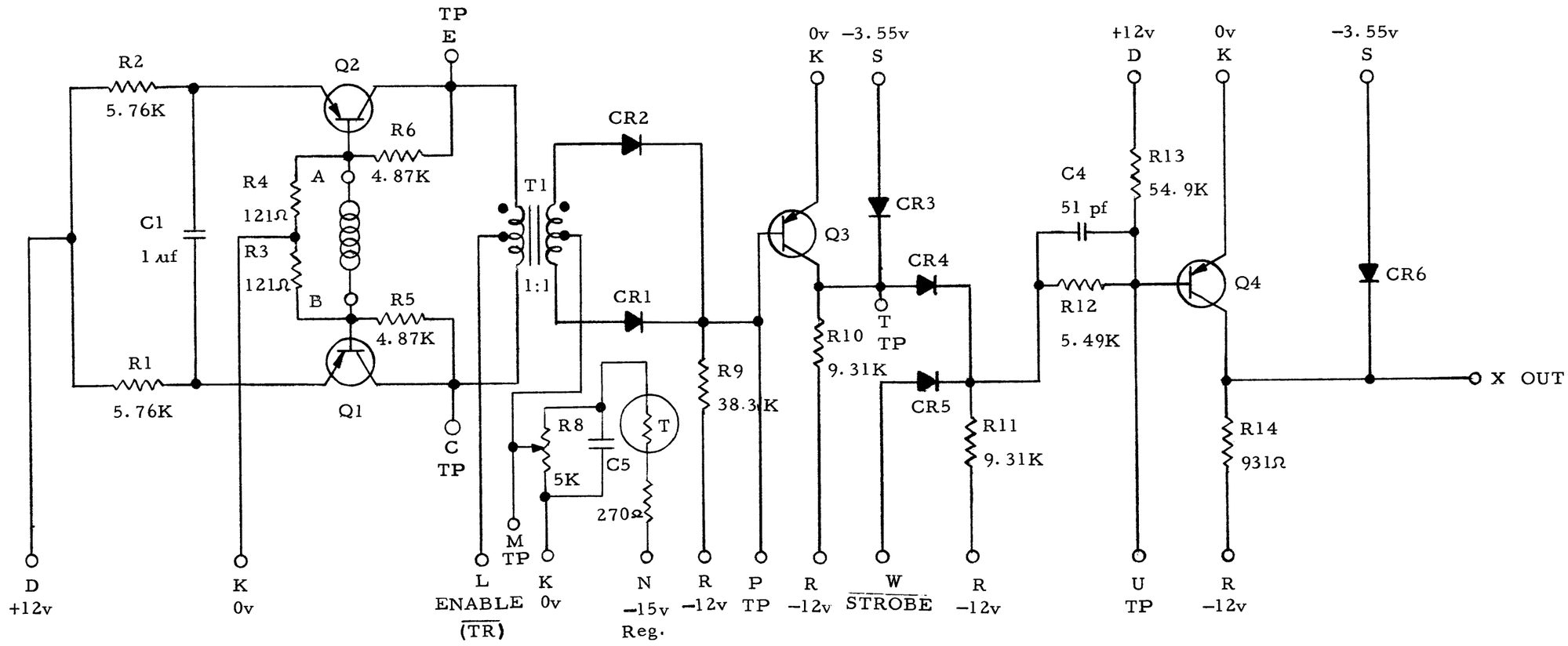
amount of current reduction in Q1 and T1a must be equal to the increase in current in Q2 and T1b. This exchange of current is switched through capacitor C1 which ties the emitters of Q1 and Q2 together. The direction and magnitude of this current that is switched through capacitor C1 is a function of the input signals that appear at the base of Q1 and Q2.

Transformer T1 couples the difference amplifier to the final stages which are operating in saturation. The secondary center tap is returned to a negative voltage whose value is determined by the variable resistance setting of R8. This voltage level determines the clip level of the sense amplifier. The secondary of transformer T1 is connected to Q3 through two diodes, CR1 and CR2. These diodes are normally back–biased while transistor Q3 is normally conducting. When Q3 is conducting the collector of Q3 is at approximately 0 volts which causes the output of the OR–gate, composed of diodes CR4 and CR5, to be high regardless of the level of $\overline{\text{STROBE}}$. When the output of the OR–gate is high, Q4 is cut off; when Q4 is off its output is low (–3. 5 volts).

We may now consider the case when a 1 is being read from the memory. In this event, a voltage pulse will appear on the sense wire. This pulse is applied directly to the base of Q1 and Q2. The polarity of the pulse will be opposite at each base giving a difference signal input.

Assume the pulse is positive at Q1 and negative at Q2. With this assumption, the collector of Q1 will go negative and the collector of Q2 will go positive. These pulses are coupled to the second stage by transformer T1. With the polarities as indicated, a negative pulse appears at the anode of CR2, while a positive pulse appears at the anode of CR1. If the positive pulse is of sufficient amplitude to overcome the bias (as determined by the setting of variable resistor, R8), CR2 will go into forward conduction which will turn off Q3. The collector of Q3 now goes low. If the $\overline{\text{STROBE}}$ pulse goes low at the same time, the output of the

FIGURE 6.3–6 Sense Amplifier Circuit



CIRC. SYM.	PART NO.
CR1 thru CR6	94A28B
Q1 and Q2	92A43A
Q3	92A41A
Q4	92A45A

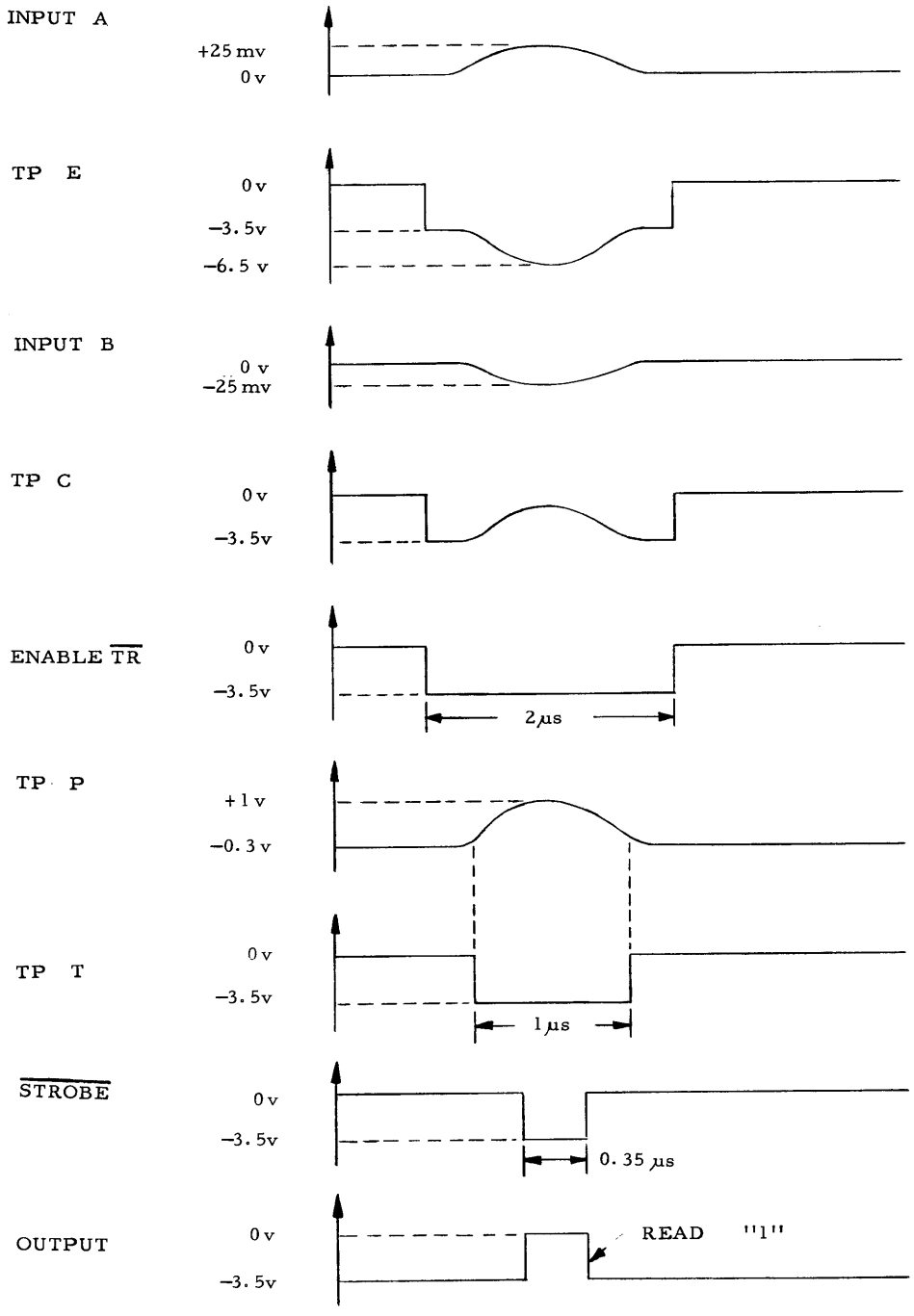
OR–gate will be low. This sequence of events will turn Q4 on; hence, the collector of Q4 will be high indicating a 1 has been read from the memory.

The $\overline{\text{STROBE}}$ pulse is controlled as described in the discussions of the strobe generator; therefore, if the memory cycle is a Clear–Write cycle, the $\overline{\text{STROBE}}$ pulse remains high and the output of Q4 is low, even though a 1 was stored in the memory.

The nominal logic 1 input signal to the sense amplifier is 50 millivolts. The clip level is adjusted so that a signal of 30 millivolts will overcome the bias on the coupling diode (CR1 or CR2) and give an output from the sense amplifier. With a difference amplifier gain of 35, the variable resistor, R8, is adjusted so that the voltage appearing at the secondary center tap is approximately –0.60 volts. This setting will allow a sense amplifier input signal of 30 millivolts to overcome the bias and give an output from the sense amplifier for each 1 bit read from memory.

Since the input stage is a difference amplifier, either polarity of input signal will cause the same output. The reason for this provision is that the sense wire may produce outputs of either polarity. In any event, the sense amplifier output must be high for a 1 and low for a 0. Figure 6.3–7 shows the timing diagram of the sense amplifier with an output of logic 1.

FIGURE 6. 3–7 Timing Diagram of a Sense Amplifier with a Logic 1 Output



SECTION 6.4 – MEMORY OPERATION

Memory modules housed in MM–10 cabinets can be accessed by either the Central Processor or the DC–11. In this chapter, however, unless specifically indicated otherwise, the information presented is referenced specifically to the core memory system working in conjunction with the Central Processor.

Each memory unit requires three types of information before a memory operation can be completed.

1. A signal to start the memory cycle timer.
2. Address information to designate the location of the desired memory word.
3. Digit information to be written into the desired location.

6.4–1 MEMORY CYCLE TIMING Figure 6.4–1 shows the memory cycle timing diagram for all G–20 core memory modules (excluding the core memory of the CB–11). The MS signal in this figure is responsible for starting the memory timing cycle. This MS signal can be initiated by only five of the 19 Central Processor sequencers. Table 6.4–1 lists these sequencers and the operations performed. Detailed discussions of these sequencers are available in Part III.

The address of the memory location to be accessed is placed in the Memory Address register at the start of the memory cycle and must remain in the same configuration for the complete memory cycle. At the same time, the flip–flops of the B register are reset in preparation to receive the information from the addressed memory location. This must be done, it will be remembered, since the transfer from the memory to the B register is single–sided with only logic 1 bits being transferred. The first operation of a memory cycle is to either read

FIGURE 6.4–1 Memory Cycle Timing Diagram

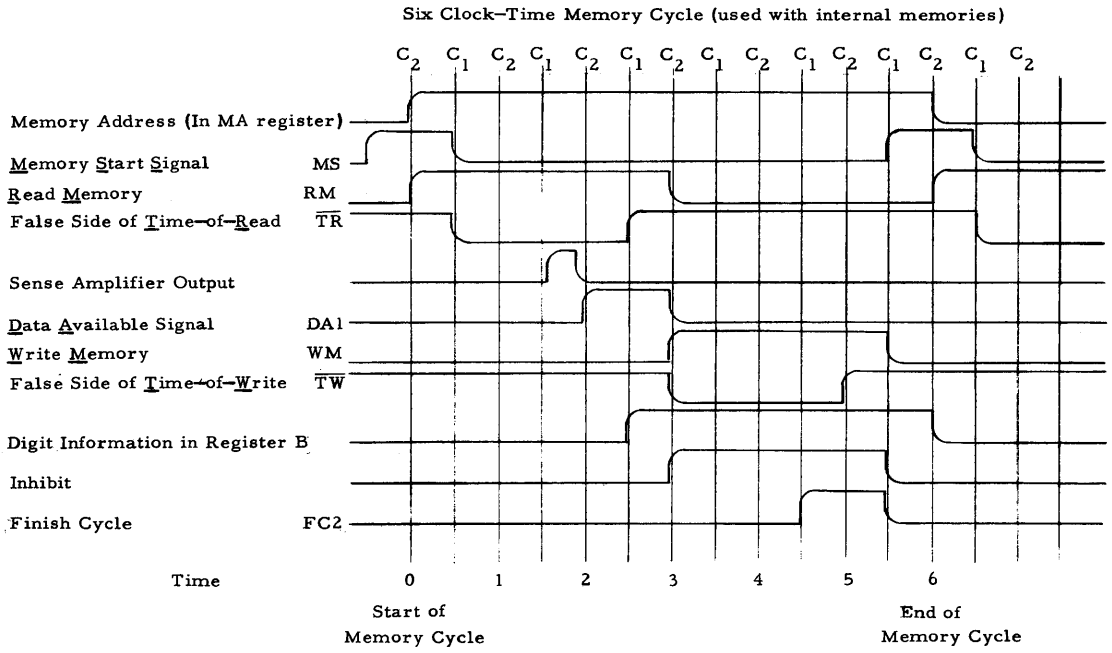


TABLE 6.4–1 Operations That Initiate Memory Access

Sequencer	Operation
KC	Command Access, Operand Assembly, Final Access for Mode 2 or 3
QC	Second Word of Multiple Access Command
KW	Block Input/Output Operations
KP	Store Operations
KJ	Mark Transfer Operations

the desired memory location into the B register or to clear the B register. (A clear operation in this case merely requires the prevention

of a read operation since the B register is already cleared.) A clear or read operation is then always followed by a write operation; however, this write operation may be delayed by several clock times. A write operation is necessary even when the information in a memory location is not being changed by a putaway or a block input command since the core memory is of the destructive read type.

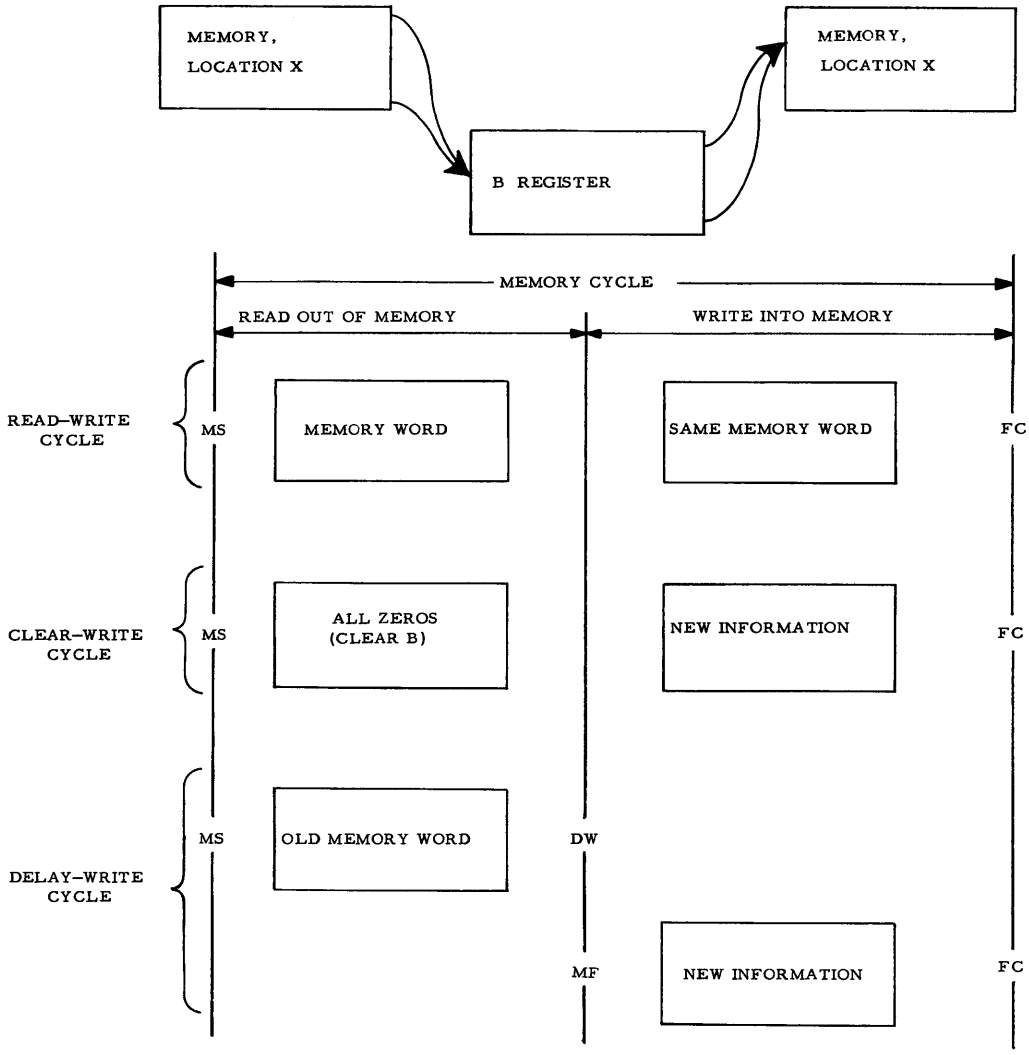
The DA and FC signals of the memory timing cycle are used by the computer control logic. The DA(Data Available) signal is brought high at approximately the same time that the memory information is transferred to the B register. The computer control logic then delays the DA signal by one–half clock time. This delayed signal allows for the bits of the B register to settle and indicates the earliest time that the information read from memory is available for further use. The FC (Finish Cycle) signal is high just before the completion of a memory cycle. Its function is to inform computer control that a new memory cycle may be started at the next clock time.

6. 4–2 MEMORY CYCLES Figure 6. 4–2 illustrates the three different memory cycles that are used with the Central Processor core memory system. It will be noted that all of the cycles begin with an MS signal and end with the FC signal.

The Read–Write cycle is quite simple with the contents of the addressed memory location placed in the B register during the read operation and then restored in the memory location during write operation. This type of memory cycle is used on all memory accesses except for putaway, mark transfer, or block receive operations.

The Clear–Write memory cycle is quite similar to the Read–Write cycle, with the exception that the $\overline{\text{Strobe}}$ signal is high which inhibits the output of the sense amplifiers and prevents the transfer of

FIGURE 6.4–2 Central Processor Memory Cycles



information from memory to the B register during the read portion of the Read–Write cycle. This type of memory cycle is used exclusively for putaway or mark transfer operations.

The Delay–Write cycle is used exclusively for block receive operations. During block receive operations new information from units external to the Central Processor is being recorded into core locations. The

contents of the core location that is to be changed is read into the B register the same as in a Read–Write cycle. The new information that is to replace this old information is not immediately available, however, since it must be brought into the Central Processor by the Communication Line at a rate of 6 or 8 bits at a time. This process is time consuming and it becomes necessary to delay the write portion of a Read–Write cycle until all bits of a new word are received in position to be written into memory. To accomplish this, a DW signal is generated during a block receive operation which temporarily halts the memory cycle timing counter at the completion of the read operation. This memory cycle will remain in this suspended state indefinitely until the sequencer that is handling the block receive operation causes an MF (Memory Finish) signal to be generated indicating that the new information has been received and is ready for storage into memory. The MF signal retires the memory cycle from its suspended condition and permits the new information, placed in the B register during the receive operation, to be written into the desired memory location.

6.4–3 ELECTRONICS OF A READ–WRITE OPERATION Figure 6.4–1 shows the memory cycle timing sequence of the read portion of a memory cycle. This section, however, is concerned primarily with the actual circuit operation during the performance of a Read and a Write operation and as such is concerned mainly with Figure 6.4–3. Figure 6.4–3 is the circuit configuration that provides the addressing and read or write functions during a memory cycle. Inhibit driver and sense amplifier connections are completely independent of this circuit and are not represented or discussed at this time.

The circuit configuration of Figure 6.4–3 is made up of an X–Y driver and current drivers of the type used for the inhibit drivers. The inhibit driver packages in this configuration have only one input and

thus only provide current carrying capabilities for their input signals. Their input signals in turn are timing signals from the memory cycle timing counter.

During computer operation only one X and Y address wire per memory module can be driven at the same time. All other address wires will be in a quiescent, undriven state.

In the quiescent state, address information inputs to the X–Y driver packages are low, the output of the AND–gate is low (–3.5 volts) and transistor Q1 is in conduction. With Q1 conducting, the collector voltage, and therefore the base voltage to both Q2 and Q3, is at approximately –1.45 volts (–1.25 volts supply voltage plus a –0.2 volt drop between the emitter and the collector of Q1). If a memory operation is <u>not</u> being processed, the RM and WM inputs to the Read and Write current drivers are low which means that transistors Q5 and Q9 are both cut off and their collectors are at –15 volts. Thus, the emitters of Q2 and Q3 are also at –15 volts and are therefore back–biased (–1.45 volts at base and –15 volts at emitter). Under these conditions, there can be no current through the address wires. It will also be noted here, for future reference, that when either the $\overline{TR}$ (Not Time–of–Read) or $\overline{TW}$ (Not Time–of–Write) timing signals are high, transistor Q11 or Q7 will be conducting with their collectors at –2.8 volts. (–2.5 volts supply voltage and –0.3 voltage drop from emitter to collector.) This means that the collector of Q2 or Q3 will be at –2.8 volts. In this particular case, however, the collectors of both Q2 and Q3 are at –2.8 volts since both $\overline{TR}$ and $\overline{TW}$ will be high.

If a memory operation is <u>not</u> being processed and address information is available at the X–Y driver (a very common occurrence), then Q1 will be back–biased and the collector will be clamped at approximately –4.5 volts. This puts the base of Q2 and Q3 at –4.5 volts and since the

emitter is at –15 volts as shown previously, then both Q2 and Q3 are back–biased and cut off. Therefore, there will be no current through the address wire.

When an X–Y driver is in the quiescent state (memory address inputs low) and a memory operation is being performed, either the RM or WM inputs of the Read or Write current drivers will be high. In this case, either Q5 or Q9 will be in conduction with the emitter at approximately –2. 8 volts (–2. 5 volts supply voltage plus –0. 3 volt drop between emitter and collector). The base of Q2 and Q3, however, will be at –1. 45 volts as shown previously. Q2 and Q3 are therefore back–biased and again there will be no current through the address lines.

Now, consider the case when address information is available to the input of the X–Y driver and a memory operation <u>is</u> started (refer to timing diagram of Figure 6. 4–1). The base of Q2 and Q3 will be at –4. 5 volts since Q1 is cut off, but there will be no current in the address line until the RM signal comes high. With RM high, Q5 is turned on and the collector goes to –2. 8 volts (supply voltage plus drop across emitter to collector junction). This causes the emitter of Q3 to be –2. 8 volts and Q3 is forward–biased since the base is at –4. 5 volts. However, there will be little current in the address wire until the $\overline{TR}$ timing signal goes low one–half clock time after RM comes high. (When TR and TW are high the collectors of Q11 and Q7, and thus the cathodes of CR1 and CR2, are at –2. 8 volts which is approximately the same potential as on the emitter of Q3.) When $\overline{TR}$ goes low, Q11 is turned off.

The voltage appearing at the collector of Q11 and at the cathode of CR1 starts to fall towards –100 volts, but is limited by the fact that the current can now flow through Q5, Q3, the core line, diode CR1, and R1. The exact value of voltage appearing at the cathode of CR1 is a

function of the current through the core line, but it will be greater than –15 volts. Current value is determined by R1, the IR drop across the core line, and the inductance of the core line and is in such a direction as to force all cores on the address line to the logic 0 state. The IR drop across the core line is very low (approximately 3.0 volts). Therefore, we must be primarily concerned with the inductive voltage present on the core line during the rise and fall of the current pulse. To prevent a high inductive voltage from damaging the transistor, a clamp voltage of –15 volts is provided. This clamp voltage also protects the transistors in case of a failure in the read or write current switch circuits. Without this protection, such a failure could permit –100 volts to be placed across the line address transistors.

Two clock times after the FALSE side of the Time–of–Read signal is made low, it is returned to zero volts, Q10 is turned off, and Q11 goes into conduction. This action diverts the current from the core line through Q11, and the cathode of CR1 is again at –2.8 volts. One–half clock time later, the Read signal goes low, Q4 is turned on, and Q5 is back–biased. The <u>read</u> portion of the memory cycle is now completed and we are ready to start the <u>write</u> portion.

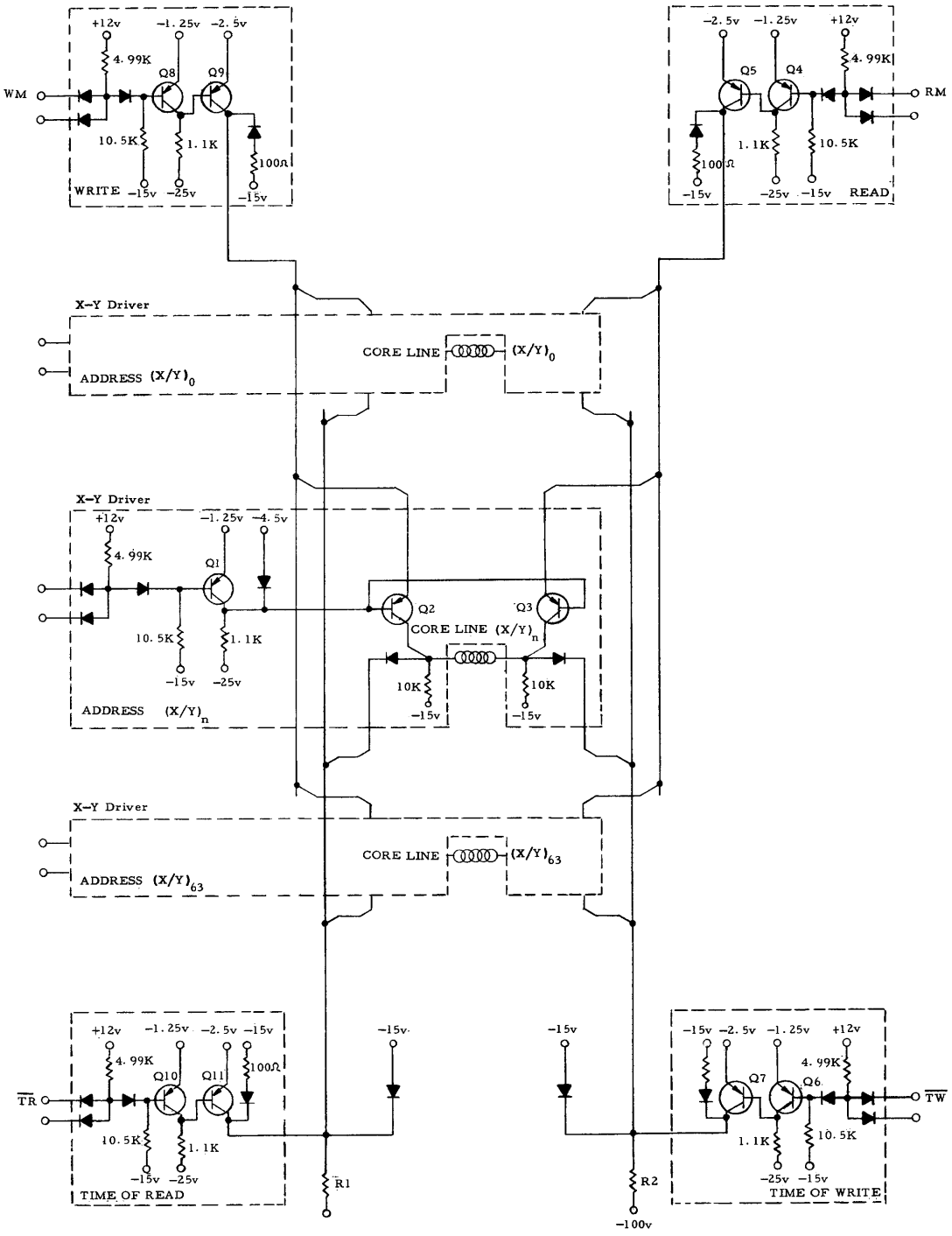
At this time the <u>write</u> signal, WM, comes high, Q8 is turned off, and Q9 goes into conduction. At the same time, the FALSE side of the Time–of–Write signal, TW, goes low and Q6 is turned on which causes Q7 to be back–biased. This sequence of events diverts a pulse of current through Q9, Q2, the core line, CR2, and R2. This current pulse through the core line is in the opposite direction to that of the read current pulse and if not inhibited will drive the core to the logic 1 state. Two clock times later, the FALSE side of the Time–of–Write pulse is brought high again; Q5 is turned off and Q7 is turned on. This diverts the current back through Q7 and brings the cathode of CR2 up to –2.8 volts, thus preventing current from flowing through the core

line. One–half clock time later, the Write signal goes low causing Q8 to be in conduction. Thus, Q9 is back–biased. This completes the write portion of the Read–Write cycle.

The address may be changed one–half clock time after the Write signal goes low. When this happens, another Read–Write cycle can proceed.

The current pulses during the read portion and the write portion of a memory cycle are in opposite directions through the core line, thus giving positive and negative current pulses.

FIGURE 6.4–3 Core Line Driver



SECTION 6.5 – MEMORY PARITY

It is felt that the possibility of errors occurring during transfer of information within the Central Processor is greatest when information is transferred from the memory cores to a flip–flop type register (i.e., a changing of storage media) or vice versa. Since reliability of information is a prime requisite of any computer, a means of immediate detection of a machine error in the representation of a number is vital. The purpose of the Central Processor memory parity generating and checking system is to provide correct parity, when a word is stored into memory and to check for correct parity when a word is brought from memory into the Arithmetic Unit.

In the event that an uneven number of bits are dropped or picked up during the process of storing a word from the B register into core memory or retrieving a word from the memory and placing it in the B register, the memory parity checking circuitry provides an indication to the Central Processor that an error condition exists.

Bit 33 of every G–20 core memory word is reserved to indicate parity. This bit is never interpreted as part of the information word. It is used exclusively for parity checking, being set to make the total number of 1 bits odd. (If the first 32 information bits of a word contain an even number of 1 bits, the thirty–third bit contains a 1; if the first 32 bits contain an odd number of 1 bits, bit 33 contains a zero.) When information is written into memory, the appropriate parity value is assigned to bit 33. Then, when the word is read out of the memory, it is checked for odd parity. If the parity of a word brought from memory is even, a memory parity error has occurred. This causes the Central Processor to unconditionally halt, the MEMORY PARITY error light to come on, and the tone generator to sound its signal (unless the SIGNAL TONE Switch is in the OFF position).

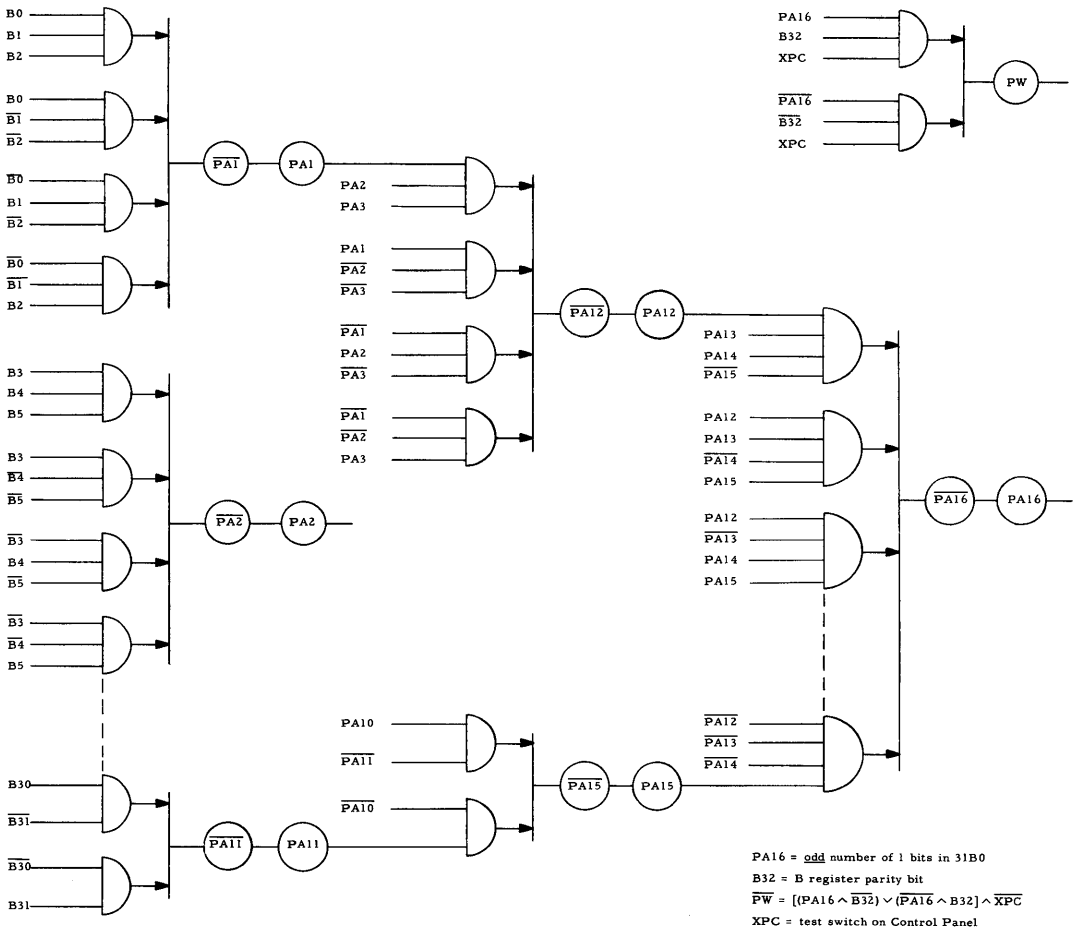
Odd parity is used since it allows a means for checking all zero words. (With odd parity, an all zero word requires a 1 parity bit.) Thus, if we try to read a word from memory and nothing is read, we have a parity error since there will be 33 zeros in the B register. (The B register is cleared when a memory cycle is started.) If the choice of an even parity had been made, 33 zeros would be a legitimate configuration.

Drawing 3E766 shows the logic schematics of the Central Processor memory parity generating and parity checking system. A simplified form of this schematic is shown in Figure 6.5–1. B0, B1, B2 – – B32 of Figure 6.5–1 represent the bits of the B register. (Information to, or from, memory must go through the B register.) PA1, PA2, PA3 – – PA11 are signals derived by decoding odd numbers of 1 bits in groups of three B register bit positions. PA12, PA13, PA14 and PA15 high indicate odd 1 bit combinations from PA1, PA2, etc. PA16 high results from an odd number of 1 bits in B0 through B31.

If a write operation is being performed, then correct parity must be generated when the word is written into memory. If the parity of a 32–bit word in the B register is odd, signified by PA16 being high, then the inhibit driver to bit 33 of the memory location is actuated which causes that core to remain in the 0 state when the write operation occurs. If PA16 is low, indicating even parity, then the inhibit driver to bit 33 is not actuated and a 1 is written into bit 33, thus providing the desired odd parity for the word in memory.

If a read operation is being performed, PA16 is compared to bit B32 (the parity position) to see if the total number of 1's of the word read from memory is odd. If the number is odd, the $\overline{PW}$ inverter comes high, indicating to the computer that a parity condition does not exist. If $\overline{PW}$ is low, a memory parity error is indicated and the Central

FIGURE 6.5–1 Central Processor Memory Parity Checking System



Processor will unconditionally halt its operations.

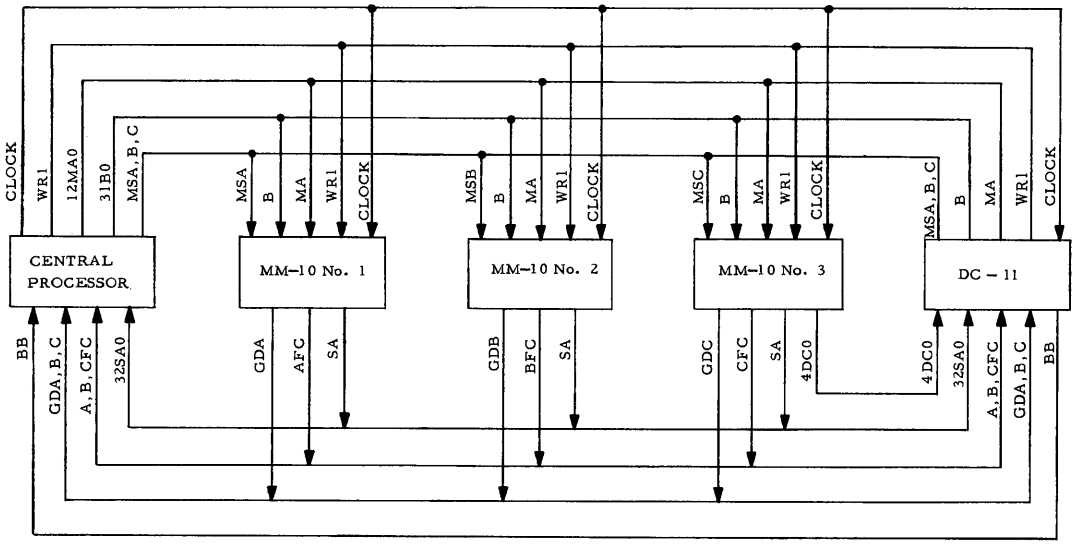
TABLE 6.5–1 Memory Parity Example

Input Bits		PA1 to PA11	PA12 to PA15	PA16
B0	1	PA1 = 0		
B1	0			
B2	1			
B3	0	PA2 = 0	PA12 = 1	
B4	0			
B5	0			
B6	1	PA3 = 1		
B7	1			
B8	1			
B9	0	PA4 = 0		
B10	1			
B11	1			
B12	0	PA5 = 1	PA13 = 1	
B13	0			1
B14	1			
B15	1	PA6 = 0		
B16	1			
B17	0			
B18	1	PA7 = 1		
B19	0			
B20	0			
B21	0	PA8 = 1	PA14 = 0	
B22	1			
B23	0			
B24	0	PA9 = 0		
B25	0			
B26	0			
B27	1	PA10 = 0		
B28	1		PA15 = 1	
B29	0			
B30	0	PA11 = 1		
B31	1			

SECTION 6.6 – EXTERNAL MEMORY MODULES (MM–10)

6.6–1 GENERAL The MM–10 is designed to provide expandable core memory facilities for a G–20 System. Each MM–10 module may contain one or two complete memory modules, similar to the one shown in Figure 6.1–1. A maximum of three MM–10 cabinets can be connected to a Central Processor in a G–20 System and, assuming the maximum number of memory modules are installed, the core memory can be expanded to 32,768 words.

FIGURE 6.6–1 MM–10 Configuration Per G–20 System



The MM–10 is connected to the Central Processor by the Memory Interface Communication Line which consists of three cables of 50 twisted pairs per cable. This communication line is limited to 65 feet maximum length and designed such that a Central Processor or a Data Communicator (DC–11) must be connected at one end of the Communication Line. Figure 6.6–1 shows the maximum system connection that

can be made using the Memory Interface Communitcation Line. If the system configuration should cause an MM–10 to be the end unit on its communication line, then the Memory Interface Communication Line must be terminated by a dummy connector wired with 100 ohm terminating resistors. The Central Processor is intended for use at one end of this communication line and, thus, the terminating resistance is built into the machine. Table 6. 6–1 gives a listing of the information transferred by the Memory Interface Communication Line. Figure 6. 6–1 and Table 6. 6–1 show that MM–10's can be used by both the Central Processor or the DC–11; however, a single MM–10 cannot be used simultaneously by both. The DC–11 has no core memory of its own and, thus, takes priority over the Central Processor when using the MM–10 for memory operations.

Signals transmitted to the MM–10 are either memory data or information necessary to initiate and supervise memory cycle operations. Information from the MM–10's to the Central Processor are memory data and signals that indicate completion of external memory operations. When a memory word is transferred between units, it is done in parallel with all 32 bits plus parity being sent simultaneously. Figure 3. 2–2 indicates the transfer paths for memory words. Words being sent from the Central Processor to the MM–10 are transmitted from B register to B register and then placed into memory. Words being read out of external memory locations, however, are sent directly to both the internal and external B registers. The MM–10's do not have their own parity generating and checking circuitry and thus parity must be handled by the Central Processor and DC–11. This internal parity check of external memory words not only provides a memory parity check, but also transmission parity check on the Memory Interface Communication Line.

It was stated previously that each memory module of the system has

its own memory cycle timing counter which, once started, operates identically independent of its physical location. There is one exception to this rule and that is that Delay–Write cycles are not performed by external memory modules. Delay–Write operations are still performed, but external memory control in the Central Processor processes these operations in such a manner that only a Read–Write cycle need be done by external memory timing.

Memory start signals to the MM–10 are in the form of MSA (Memory Start, Cabinet A), MSB, and MSC depending upon which of the three possible MM–10 units is to be accessed. The MA12 bit configuration is used to determine which of the two possible modules in a cabinet is to be accessed. Both memory timers of the modules in an MM–10 are started when it receives its MS signal, however, the MA12 signal determines which timer is to be allowed to continue. The generation of MS signals to the MM–10 units is dependent upon the ERD (External Ready) signal. When ERD is low, it indicates that certain MM–10 units are in use and that the Central Processor will have to wait if it tries to access such an MM–10 unit. Provisions have also been made to handle the case where both the Central Processor and the DC–11 attempt to access an available MM–10 unit simultaneously. To avoid confusion and establish DC–11 priority, the DC–11 sends out a Bus Busy signal (BB) a short time before it sends out its MS signal. This signal informs the Central Processor that the DC–11 is anticipating a memory operation and that the Central Processor must wait until the DC–11 is finished before it can access the same MM–10 unit.

The MM–10 does not have its own clock generator and thus, the clock pulses must be transmitted to the MM–10's. The communication line creates somewhat of a problem since the maximum transmission rate will not accommodate the clock pulse rate. The means of transmitting and reconstructing the clock pulses are discussed later in this section.

It should be noted that there are six signals carried by the Memory Interface Communication Line that only affect MM–10 modules and the DC–11. Five of these signals, DC1 through DC5, are used to indicate to the DC–11 when the Central Processor has loaded something into one of the 16 most significant words of the system's memory. These signals will be decoded by the DC–11 and may initiate a DC–11 channel operation. For further information, it will be necessary for the reader to consult the DC–11 Manual. The remaining signal, DA1, is used to indicate a memory early done to the DC–11.

TABLE 6.6–1 Memory Interface Communication Line Signals

Signal	Number of Twisted Pairs	C.P.		MM–10 1		MM–10 2		MM–10 3		DC–11	
		Xmit	rcv	Xmit	rcv	Xmit	rcv	Xmit	rcv	Xmit	rcv
MM–10 On (+24 volts DC)	1	X			X		X		X	X	
Memory On (+24 volts DC)	1	X			X		X		X	X	
Clock	4	X			X		X		X		X
12MA0	26	X			X		X		X	X	
31B0 and PA16	33	X			X		X		X	X	
32SA0	33		X	X		X		X			X
WR1	2	X			X		X		X	X	
MSA	1	X	X		X					X	X
MSB	1	X	X				X			X	X
MSC	1	X	X						X	X	X
GDA	1		X	X							X
GDB	1		X			X					X
GDC	1		X					X			X
AFC	1		X	X							X
BFC	1		X			X					X
CFC	1		X					X			X
BB	2		X							X	X
Manual ZM	1	X			X		X		X	X	
Set URB	1		X	X		X		X		X	
DA1	1			X		X		X			X
DC0	1							X			X
DC1	1							X			X
DC2	1							X			X
DC3	1							X			X
DC4	1							X			X

6.6–2 MM–10 LOGIC The MM–10 has limited logic associated with it since it is comprised mainly of standard memory modules and must be used in conjunction with a control unit. The logic that is associated with the MM–10, because of its passive nature, is distributed between

the MM–10 itself, the Central Processor, and the DC–11. In this manual, the primary concern is MM–10 operation in conjunction with the Central Processor and thus, MM–10 associated logic in the DC–11 will not be discussed at this time.

MM–10 Logic in the Central Processor. Table 6. 6–2 lists logic signals associated with the MM–10 that are generated within the Central Processor and Figure 6. 6–2 presents the source of these signals. In these figures, flip–flops are designated by blocks with their Set and Reset terms indicated. Most of these signals are straight forward and need little explanation.

Referring to the memory start signals for external memories (MSA, MSB, and MSC of Figure 6. 6–2) it will be seen that it is called by several different names in the logic. Actually, the MSA, MSB, and MSC signals are not transmitted to the MM–10's. Instead, the LM signal is received by the MM–10, amplified, and used to set the MM–10 MS flip–flop. The MSA, MSB, and MSC signals are only used by the Central Processor. These signals may be generated as the result of a pulse transmitted from a DC–11 when it is addressing one of the units or by the Central Processor itself when it is addressing a particular unit. This latter action by the Central Processor is rather unusual inasmuch as both line driver and line receiver packages are activated at the same time. The reason for this is that the MSA, B or C signal is used to reset the appropriate Finish Cycle flip–flop. It will be noticed that the Memory Ready flip–flops are merely copies of the Finish Cycle flip–flops and are therefore also reset. Thus, if either the Central Processor or a DC–11 accesses an external memory module the Central Processor is notified that that particular MM–10 unit will not be available for a memory access until the appropriate signal (AFC, BFC, or CFC) is received from the MM–10. It will also be noticed that the outputs of the Memory Ready flip–flops are gated

with the appropriate memory module decoding (SMC through SMH) as partial conditions to generating the ERD (External Ready) signal. This signal is quite important to the QB sequencer which is discussed in Part IV.

When a Gate Data signal (GDA, GDB, GDC) is received from an MM–10 cabinet during a read external memory operation, it is stored in the DAL (Data Available from Line) flip–flop. This signal is used to generate the SA(T)B enable path which allows information obtained by the MM–10 sense amplifiers to be sent directly to the Central Processor B register. The SA(T)B signal is then held high for two microseconds to allow for memory variation and communication line delays.

WARNING: Notice that the sense amplifier gates, SA(T)B, are opened on a C2. In an MM–10 memory cycle, read current starts flowing on the previous C1. Approximately one microsecond later (at about the next C1) there will be sense amplifier current. If the clock is running, there is no problem. Read current starts on a C1; the gates are opened on the next C2; and at approximately the following C1 the sense amplifiers set the B register. Now consider the same sequence under hand clock operation. The hand clock switch is first moved to the C1 position. About one microsecond later there is sense amplifier information. This is long before a person can even think about moving the hand clock switch to C2, let alone actually move it. As a result, the sense amplifier information will come flying out of core, go into the MM–10 B register correctly, but will not enter the Central Processor B register because the SA(T)B gate is still waiting for a C2.

MM–10 Logic in the MM–10 Unit. Table 6.6–3 lists logic signals associated with the MM–10 that are generated within the MM–10 units. Figure 6.6–3 illustrates the generating of these signals. Blocks 31B0

FIGURE 6.6–2 MM–10 Logic in the Central Processor

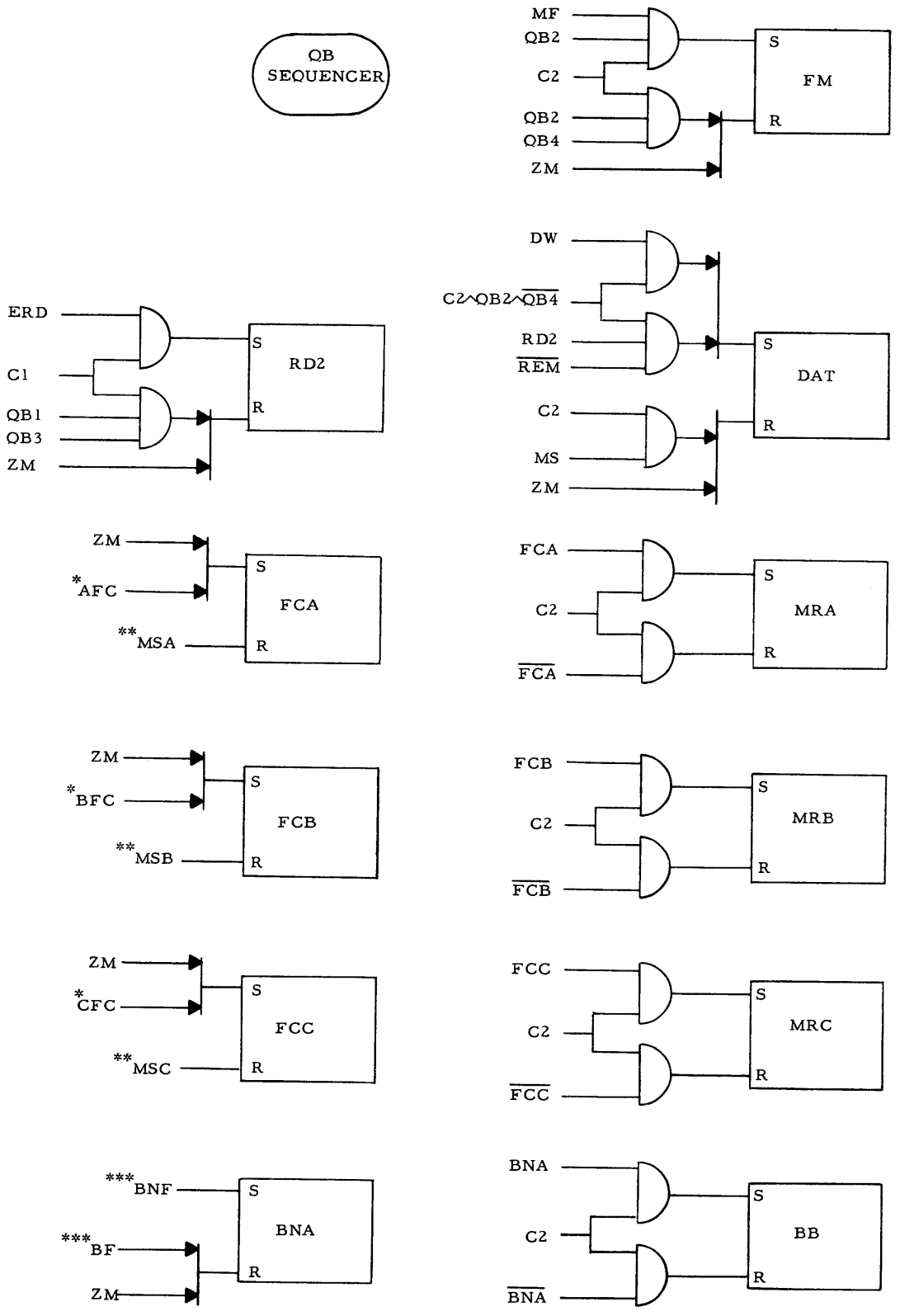
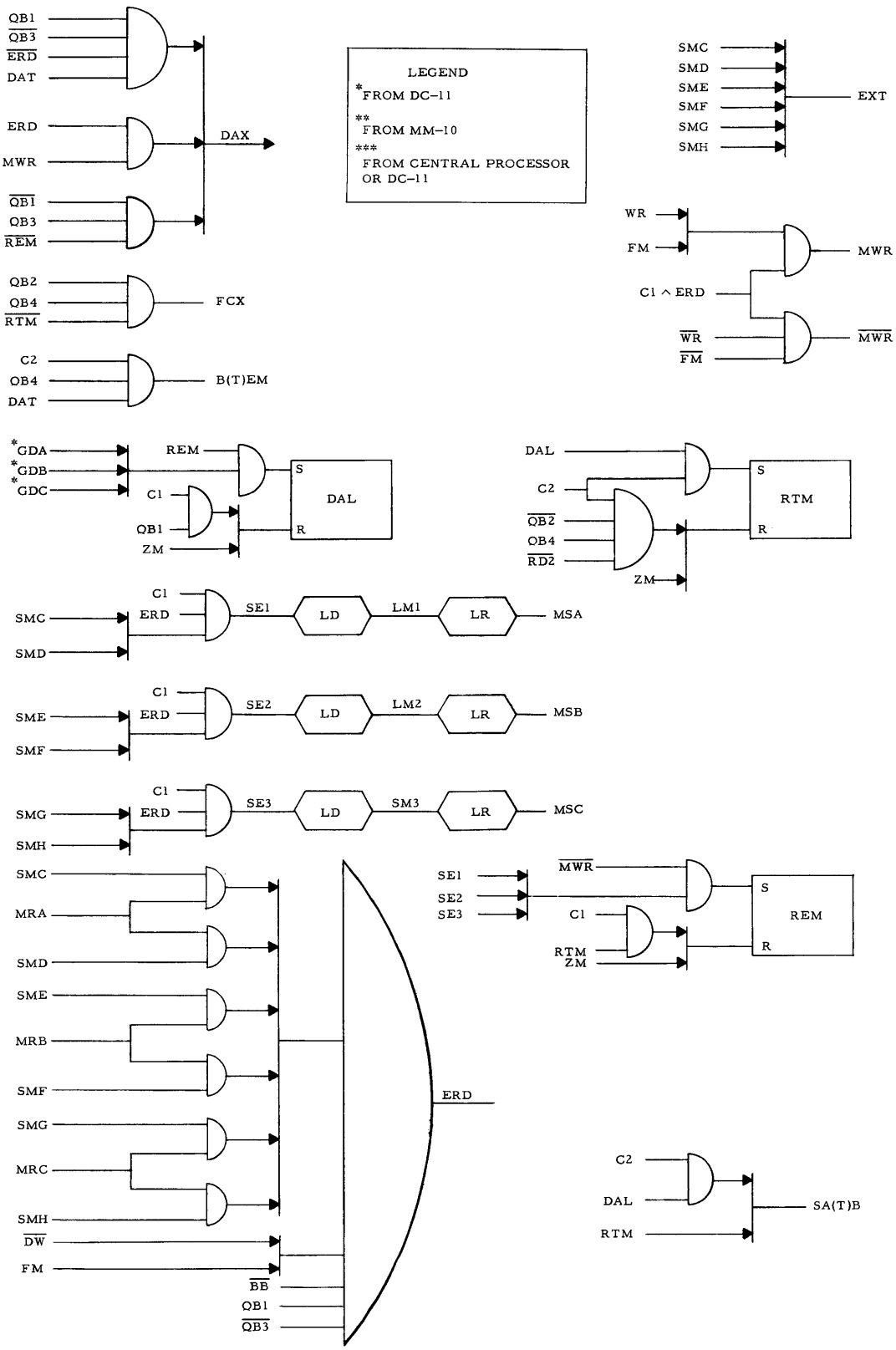
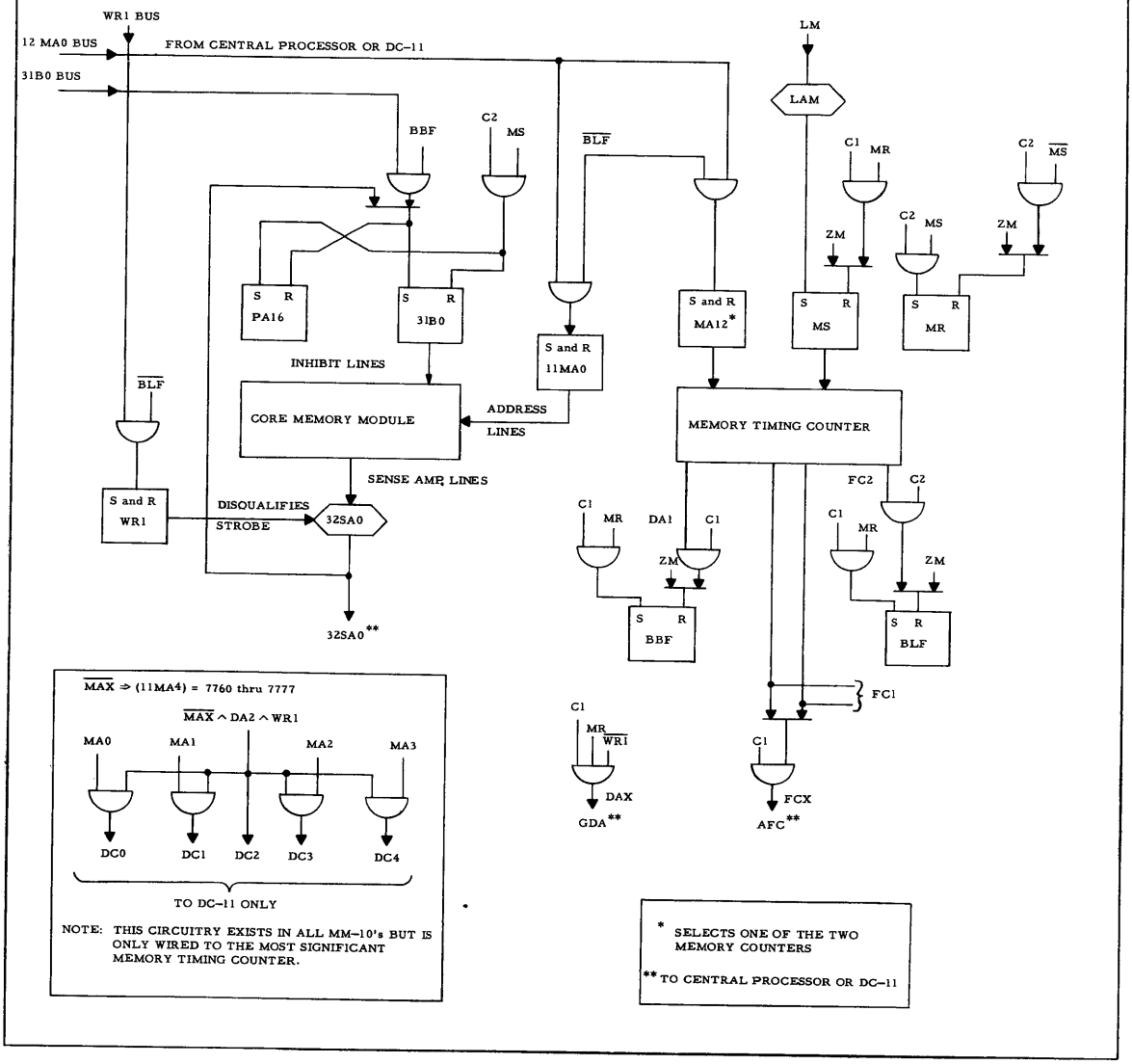


FIGURE 6.6–2 . MM–10 Logic in the Central Processor (Continued)



and 11MA0 are registers, block 32SA0 represents the sense amplifiers, and the remaining mnemonic designated blocks represent flip–flops.

FIGURE 6. 6–3 MM–10 Signals Generated in MM–10 Units



When an external memory module is started, the address bits and the flip–flop WR1 state are sent to the appropriate registers in the external module. One clock time later, the BBF and BLF flip–flops are set. The BLF flip–flop blocks the transfer of further information into the MM–10 MA register or the WR1 flip–flop. This condition exists until

TABLE 6.6–2 MM–10 Signals Generated in Central Processor

Signal	Drawing	Notes
12MA0	MA	memory address register
31B0	MB and MC	"B" register
BB	MN	bus busy
BNA	MN	bus not available
B(T)EM	MK	"B" register true–sided transfer to external memory
CK1	MK	clock frequency divider ff
CK2	MK	clock frequency divider ff
DAL	MN	data available from line
DAT	MK	data transmit
DAX	MK	data available "X" (⇒ set DAS, DAR)
ERD	MK	external ready
EXT	MK	external memory
FCA, B, C	MN	finish cycle, A, B, C
FCX	MK	finish cycle X (generated by QB, used by QM)
FM	MN	finish memory operation (set by MF)
MRA, B, C	MN	memory ready A, B, C
MSA, B, C	MH	memory start, module A, B or C
MS	MC	memory start
MWR	MJ	memory write
4QB1	MK	buffer sequencer ff's
RD2	MK	ready ff
REM	MN	read external memory
RTM	MN	read transmitting memory
SA(T)B	MK	sense amp. true–sided transfer to "B" register

TABLE 6.6–2 (Continued)

Signal	Drawing	Notes
SE1, 2, 3	MH	start external memory (SE1, 2, 3 ⇒ MSA, B, C)
WR1	MC	write ff (controls strobe generator)
PA16	MC	parity bit

TABLE 6.6–3 MM–10 Signals Generated in MM–10 Units

Signal	Drawing	Notes
AFC		finish cycle, MM–10 Unit A
BFC		finish cycle, MM–10 Unit B
CFC		finish cycle, MM–10 Unit C
12MA0	MA	memory address register
31B0	MB and MC	"B" register
BBF	MA	"B" bus flip–flop (enable)
BLF	MA	block flip–flop (MA and WR buses)
DA1	MA	data available
DAX	MD	data available "X" (⇒ GDA, B, C)
5DC0	ME	data communicator (DC–11) channel words
FC2, 1	MA	finish cycle
FCX	MD	finish cycle X (⇒ A, B, CFC)
GDA, B, C	MJ	gate data A, B, C (data available from external memory)
MR	MC	copy of MS flip–flop
MS	MC	memory start

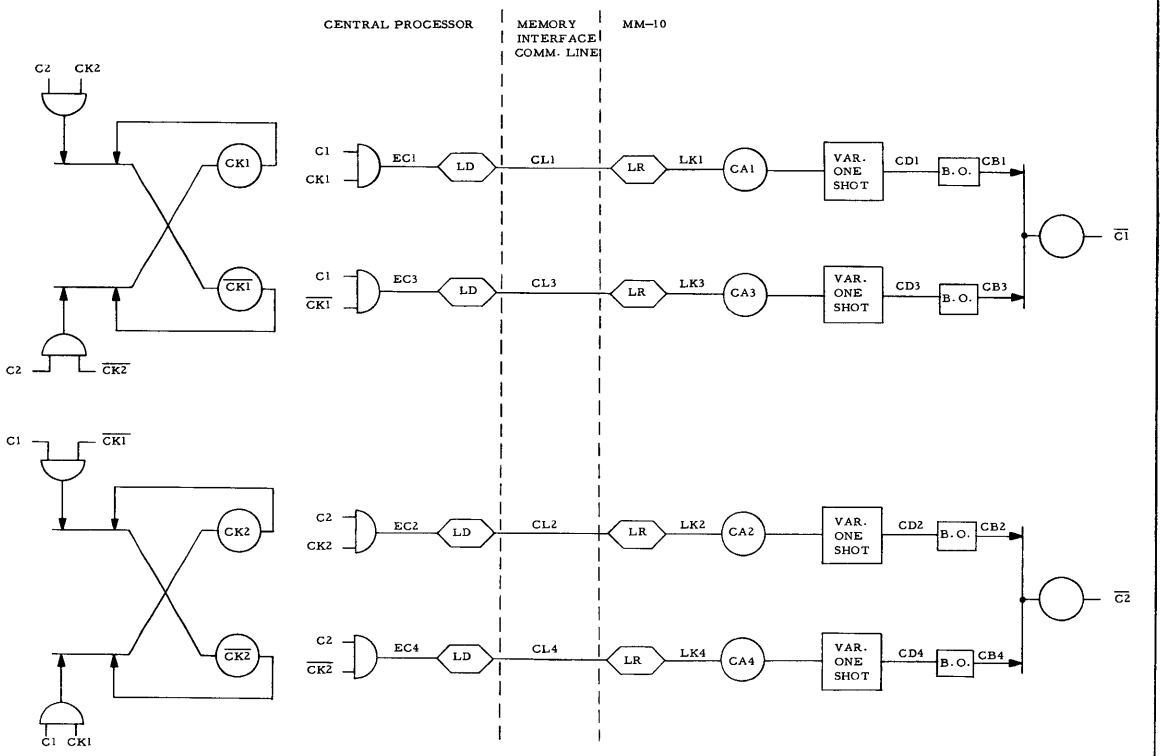
the external memory cycle is finished. The BBF signal, on the other hand, enables the transfer of information from the internal B register to the MM–10B register. The BBF signal is reset, thus closing the MM–10B register to the Central Processor, when the DA1 signal from the memory cycle timing counter comes high. This allows the B register to settle and prevents disturbance of the B register during the Time–of–Write.

Also shown in Figure 6.6–3 is the generation of the 5DC0 signals that are used by the DC–11.

6.6–3 MM–10 SPECIAL CIRCUITS; MM–10 CLOCK The MM–10 does not have its own clock generator. Therefore, it must develop a C1 and a C2 from the clock pulses that are transmitted over the Memory Interface Communication Line from the Central Processor. This communication line, however, cannot handle signals at a one megacycle clock rate. To compensate for this, every other C1 pulse is placed on one communication line twisted pair and the remaining C1 pulses are transmitted through a second twisted pair. The same is done for the C2 clock pulses. This procedure reduces the transmitted clock rate per twisted pair to 500 kilocycles, which the communication line can handle.

Figure 6.6–4 shows the logic needed to transmit and reconstruct the clock and Figure 6.6–5 presents the MM–10 clock timing diagram. As a pulse traverses the Communication Line there will be an average delay of 1.6 nanoseconds per foot and a worst case delay of 2.9 nanoseconds per foot. Since the different MM–10 units will necessarily be at different distances from the Central Processor, there will be different time delays of the clock pulses between cabinets. It is necessary, however, that all units act in concert with the Central

FIGURE 6. 6–4 MM–10 Clock Transmitting and Reforming Logic



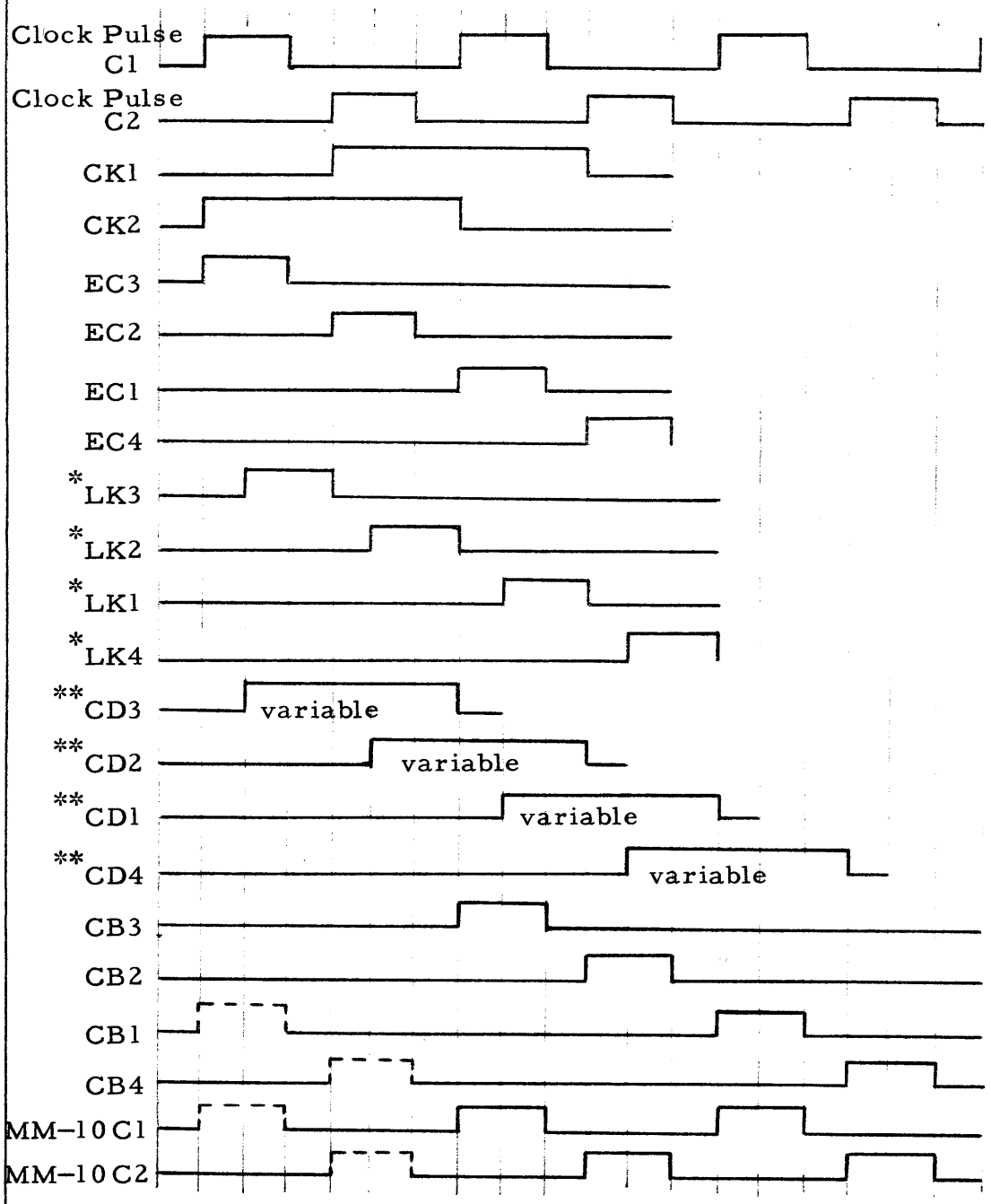
Processor. Since it is impossible to regain time that is lost, it therefore becomes necessary to create a uniform delay of the clock signals to all MM–10 units. Variable one–shot multivibrators are used in the circuit to create a standard total delay of one microsecond. This means that a C1 or a C2 in the MM–10 will occur at the same time as a C1 or a C2 in the Central Processor.

On the clock generator in the Central Processor there are controls to adjust the frequency, width, and symmetry of the clock. There are no such controls in the MM–10. If the clock frequency is varied in the Central Processor, the same variation will be felt in the MM–10 as long as the variation is held between ± 10 per cent of the normal one megacycle clock frequency. If the C1 or C2 pulse width is varied in

the Central Processor, it will not affect the clock pulse width in the MM–10. This is true since the output of the blocking oscillator of Figure 6.6–5 has a pulse width of 0.35 microseconds regardless of the pulse width of the input signal. The input to the blocking oscillator is fed by a trigger TRG1 package. The TRG1 package is sensitive only to the trailing edge of the one–shot multivibrator delay, and the one–shot is sensitive only to the leading edge of the Communication Line signal. When the clock symmetry is varied in the Central Processor, the time between the C1 and C2 pulses are varied since the one–shot multivibrators in the MM–10 are sensitive to the leading edges of received pulses, the symmetry variation will also appear in the MM–10. Not shown in Figure 6.6–5 is a hand clock switch in the MM–10. This switch effectively applies ground at the OR–gates on the input to the $\overline{C1}$ or $\overline{C2}$ inverters. Moving the switch to the C1 position will cause C1 to always be high in the MM–10. A similar action occurs if the switch is moved to the C2 position. This switch has no affect whatsoever on the clock in the Central Processor.

MM–10 Core Memory Selector and Start–Run–Slave Switches. In the general discussion of the MM–10, Section 6.6–1, it was seen that each MM–10 unit receives an individual memory start signal (MSA, MSB, or MSC) and in turn sends out corresponding Gate Data (GDA, GDB or GDC) and Finish Cycle (AFC, BFC, or CFC) signals. Since all MM–10 units are identical, a Core Selector switch is built into each unit. This switch enables any MM–10 to be designated as an A, B, or C unit. This switch is located below Logic Panel T in the power supply section of the MM–10. Also located in the power supply section of the MM–10 is a START–RUN–SLAVE Switch. This switch is similar in operation and purpose to the ON–SLAVE–OFF Switches on other peripheral cabinets. The START position will unconditionally turn on the MM–10. The RUN position will merely maintain the present status of the MM–10; that is either On or Off. In this position, the MM–10 is not affected

FIGURE 6. 6–5 MM–10 Clock Timing Diagram



*Difference between leading edges of EC3 and LK3 is Communication Line delay. (Same applies to LK2, 1, and 4.)

**The CD3 one shot is adjusted to produce a total delay of one clock cycle. (Same applied to CD2, 1, and 4.)

by Central Processor action. In the SLAVE position, the MM–10 will turn on if the Central Processor is turned on and similarly will turn off if the Central Processor is turned off. Notice that other than disconnecting the power cable or tripping the circuit breaker, the only way to turn off the MM–10 is to turn off the Central Processor.

Turn–on Cycle. Refer now to the turn–on schematic, drawing 3D1173. When the START–RUN–SLAVE Switch, 5S1, is placed in the START position, it energizes 12K5 which will turn on the d. c. voltages and provide a Zero Machine, ZM1, signal. The d. c. voltage energizes relays 12K1, 12K4, and 5K1. 12K1, being energized, terminates the ZM signal. Solenoid 12K4 energized provides a hold contact for 12K5, and 12K5 keeps the d. c. voltages on. The start position of the START–RUN–SLAVE Switch is a self–return to Run position. After the MM–10 has been turned on and the switch is released to the Run position, the d. c. voltages keep 12K4 energized. 12K4 provides a hold contact for 5K1, which provides a hold contact for 12K5, which keeps the d. c. voltages on. Consider now the example where the MM–10 is in the SLAVE position with the Central Processor being turned on. The continuous +24 volt d. c. On signal from the Central Processor energizes relay 5K1. Relay 5K1, in turn, energizes 12K5, which turns on the d. c. voltages and provides the ZM signal. The d. c. voltages energize relay 12K4 to hold 5K1 with the continuous +24 volt d. c. MM–10 On signal from the Central Processor. A short time later in the Central Processor Turn–on cycle, another +24 volt signal, the memory On signal, is sent to the MM–10 to energize and continuously hold relay 12K1. The contacts of 12K1 terminate the ZM signal, provide the memory stack voltages and also the $12 volt d. c. line amplifier and line driver voltage. In the case of a malfunction wherein the –3.55 over voltage is triggered, relay 12K2 is energized which in turn energizes 12K3, which drops out 12K5 and causes the MM–10 to lose all d. c. voltages. With the d. c. voltages gone, 12K4 and 5K1

also drop out. It is helpful to remember when working in these circuits, that 12K2 and 12K3 are normally never pulled in.

TABLE 6. 6–4 MM–10 Logic and Power Supply Drawing Numbers

in Central Processor	AC	3E677	Memory Timing Counter and Decoding
	MH	3E1126	Inhibit and Control Signals to MM–10
	MJ	3E1130	Sense Amp. Line Receivers
	MK	3E1133	QB Bldg. Block and Control Signals
	ML	3E1128	Memory Address and Control for MM–10
	MN	3E1168	QB Bldg. Block and Control Signals
in MM–10	MA	3E1121	MM–10 Memory Reg. and Control Signals
	MB	3E1122	MM–10 "B" Reg. Bits 0–24
	MC	3E1123	MM–10 "B" Reg. Bits 25–32 and Control Signals
	MD	3E1124	MM–10 Address Lines
	ME	3E1125	MM–10 Inhibit Lines
	MF	3E1132	Sense Amp. Line Drivers
	PS	3E991	Schematic – Power Supply and Control
	PT	3E990	Schematic – Regulators and Thermostat Control
	PW	3D1173	Turn–on Schematic

CHAPTER 7

ELECTRONICS

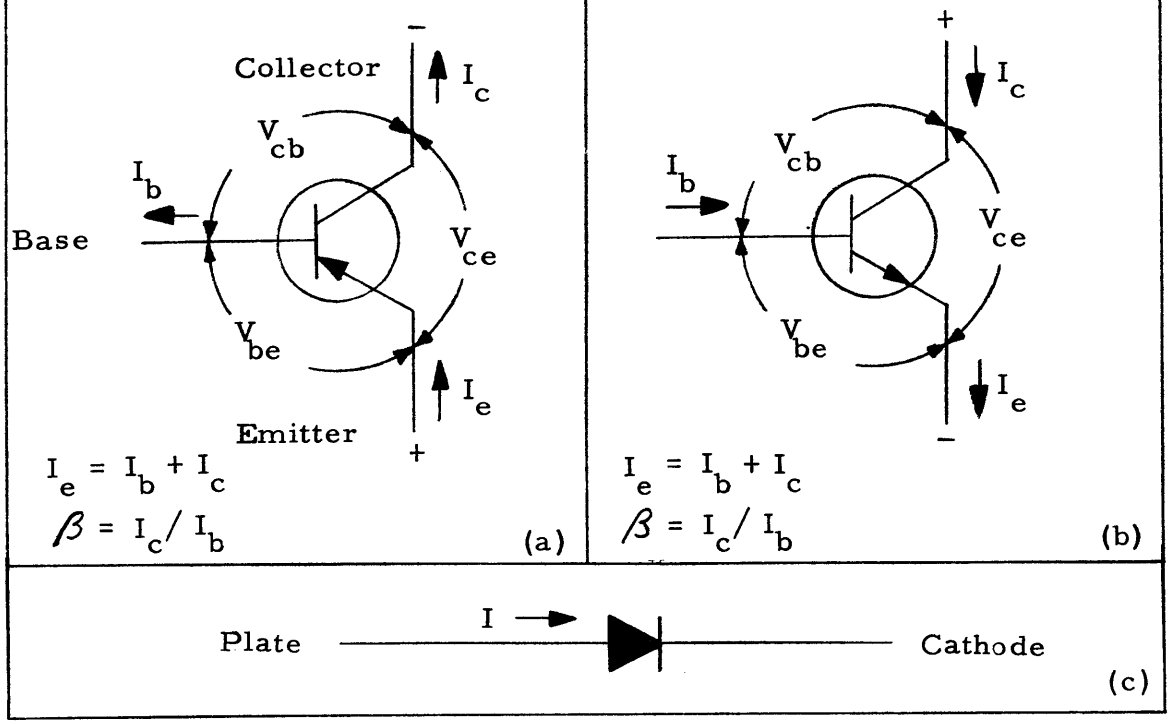
In all of the sections of this chapter on electronics, the convention of conventional current flow as opposed to electron current flow will apply. This means that in the following discussions current will be assumed to flow from + to −. Thus, when a diode is forward biased (i.e., the plate is more positive than the cathode) then the current is assumed to flow from the plate to the cathode. This convention allows one to remember easily the direction of current flow of an active element (diode or transistor) since the current will always flow in the direction of the arrow head. This is illustrated in Figure 7.1.

The transistors in the logic are used as d.c. switches. The transistor is either turned on (current through it) or turned off (no current through it). For example, a PNP transistor with the emitter more positive than the base is forward biased, and all but a small per cent of the emitter current flows out of the collector. If the base is more positive than the emitter, it is back biased: the transistor is cut off and no current flows. The collector of the PNP transistor is usually negative biased in respect to <u>both</u> the emitter and the base.

In the calculations that occur in this chapter, internal voltage drops of the diodes and transistors may be neglected (since they are but a few

tenths of a volt) to simplify calculations.

FIGURE 7.1 Active Elements (a) PNP Transistor (b) NPN Transistor (c) Diode



SECTION 7.1 – LOGIC CIRCUITS

7.1–1 INTRODUCTION Reliability, simplicity, miniaturization, and ease of servicing were of prime importance in the design of the G–20 electronics. Electrical components are assembled on printed circuit boards to form various types of computer circuits. The packaging of each circuit is such that combinations of many simple packages may be used to form a variety of more complex circuits. These basic computer circuits are then wired together to perform the logic functions.

The two signal voltage levels generally used in the Central Processor are approximately 0 volts and –3.5 volts. Input and output signals of most logic circuits are at one of these two voltage levels. Reference to a signal as high, or a logical 1 indicates a voltage level near zero; similarly, reference to a signal as low, or a logical 0 indicates –3.5 volts. The possible synonyms for the bi–stable conditions are listed in Table 7.1–1.

TABLE 7.1–1 Terminology for Bi–stable States

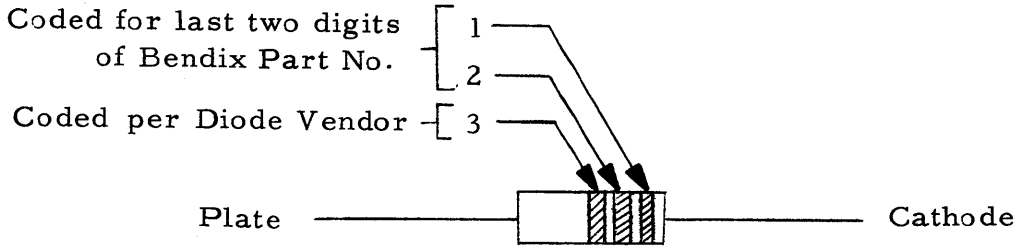
Binary or Boolean	1	0
Boolean, Binary, Electrical	High	Low
Electrical	0 volts	–3.5 volts

The amount of current each circuit can supply or receive and still safely maintain its proper voltage level determines which, and how many other circuits it is capable of driving.

Most of the components are of standard type and it is felt that further

discussion of them is unwarranted. The diodes, however, are unusual inasmuch as a Bendix Computer system of color coding is used on them. The scheme for this color coding is presented in Figure 7.1–1.

FIGURE 7.1–1 Bendix Computer's Color Code for Diodes

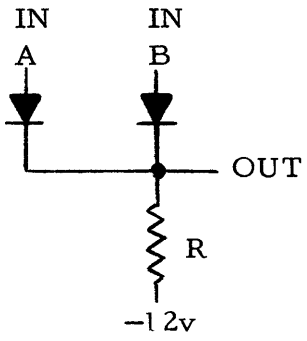


7.1–2 DIODE GATES In the Central Processor, diodes and resistors are combined to form gates that perform Boolean AND and Boolean OR functions. Electrical and logical representations of these gates are presented in Figure 7.1–2.

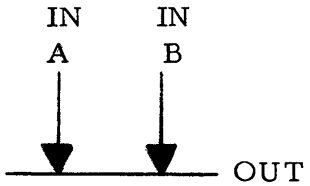
It will be noted from Figure 7.1–2 that each gate has but one output whereas, theoretically, the number of input terms is unlimited. However, circuit limitations due to diode forward and reverse resistances and currents allow only a maximum of six input terms for each AND–gate and ten input terms for each OR–gate.

Consider the circuits of Figure 7.1–2. If the input terms A and B of the OR–gate are both low (–3.5 volts) the output term is also low. If either A or B or both A and B are high (0 volts) then the output is high. Thus, this circuit truly performs the Boolean A $\vee$ B logic function. The resistor R is referred to as the pull–down resistor since it pulls the output voltage level down to the highest input voltage level.

FIGURE 7.1–2 Electrical, Logical and Functional Representation of (a) AND–gates (b) OR–gates



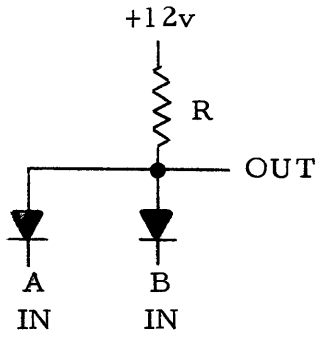
Electrical



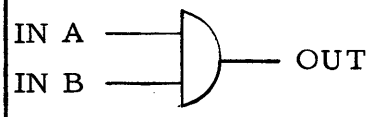
Logical (a)

Input A	Input B	Output
–3.5v	–3.5v	–3.5v
0v	–3.5v	0v
–3.5v	0v	0v
0v	0v	0v

Functional



Electrical

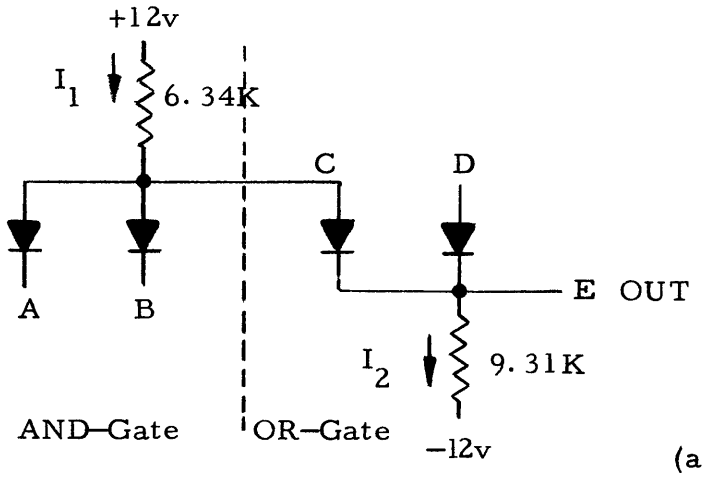


Logical (b)

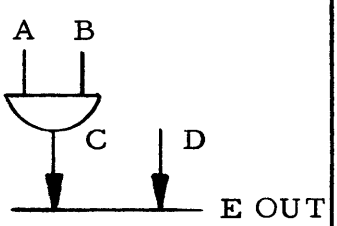
Input A	Input B	Output
–3.5v	–3.5v	–3.5v
0v	–3.5v	–3.5v
–3.5v	0v	–3.5v
0v	0v	0v

Functional

FIGURE 7.1–3 Typical AND – to – OR Circuit (a) Electrical Circuit (b) Symbolic Diagram



(a)



(b)

If the input signals A and B of the AND–gate of Figure 7.1–2 are both high, then the output is high. On the other hand, if the inputs of either A or B, or both, are low, then the output is low. Thus, this circuit truly performs the Boolean $A \wedge B$ logic function. In this case the resistor R is referred to as the pull–up resistor.

The circuit shown in Figure 7.1–3 is that of an AND–gate driving an OR–gate. Note that when the input D is high, the output of the OR–gate will be high regardless of the voltage levels of A and B. Consider the case where the AND–gate output, C, is low, and the OR–gate input, D, is low. (Diode voltage drops are ignored in this discussion since they are so slight as to be negligible.)

$$I_1 = [+12v - (-3.5v)]/6.34K = 2.45 \text{ ma}$$
$$I_2 = [-12v - (-3.5v)]/9.31K = 0.91 \text{ ma}$$

Since I_1 is greater than I_2, the AND–gate is capable of driving the OR–gate. I_1 is distributed among the AND–gate input terms that are low and the OR–gate. A portion of I_2 is supplied by the base current of the following inverter (not shown) which is connected to the output terminal E. On the other hand, if the AND–gate inputs are high and D is high, then:

$$I_1 = (+12 - 0)/6.34K = 1.89 \text{ ma}$$
$$I_2 = 0 - (-12)/9.31K = 1.29 \text{ ma}.$$

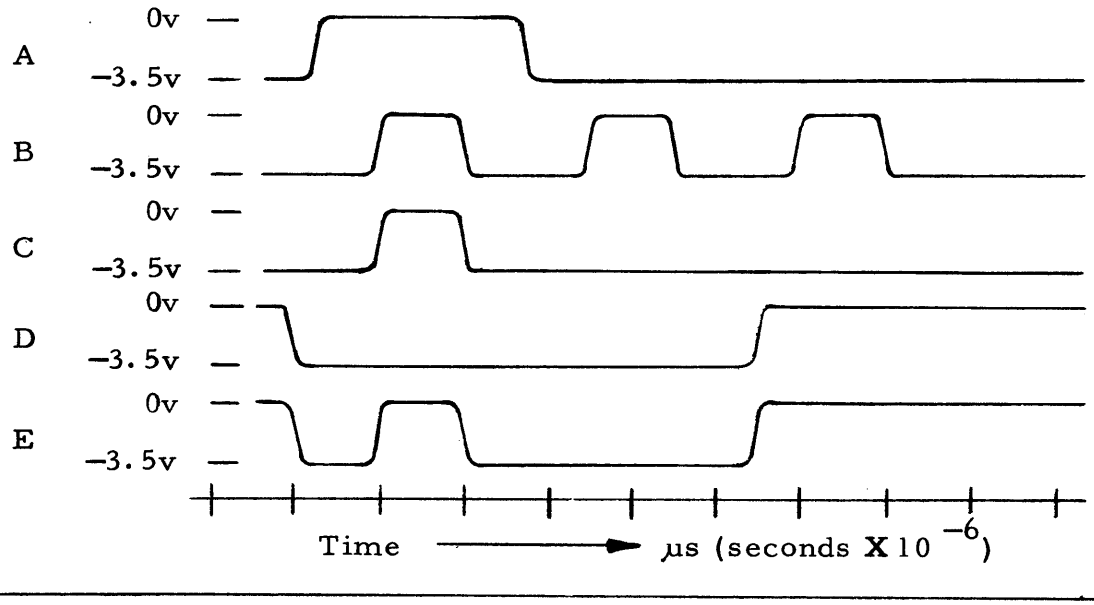
Here again I_1 is greater than I_2 so that the AND–gate is capable of driving the OR–gate. Notice, however, that the AND–gate cannot supply enough current to drive more than one OR–gate.

It should be noted that circuits may be designed so that AND–gates drive OR–gates or vice versa. The diode logic circuits of the Central Processor are all designed on the AND–to–OR principal. An investigation of the previous example reveals that, if that OR–gate were used to

drive the AND–gate, the currents are such that the OR–gate could not maintain the proper signal levels.

Thus far the operation of diode gates has been described in terms of d. c. levels only. Refer now to the circuit of Figure 7. 1–3 and assume that the input terms A, B, and D are those shown in Figure 7. 1–4. From what has been learned about gate operation, the output terms C and **E** can be drawn. These same shapes could be observed on an oscilloscope.

FIGURE 7. 1–4 An Example of Gate Wave Shapes



Notice that a finite period of time is required for the wave forms of Figure 7. 1–4 to switch or change from one voltage level to another. These periods of time are referred to variously as switching time, turn–on and turn–off time, or rise and fall time. Switching time is largely a function of circuit capacitance. Square waves and pulses are merely combinations of a wide range of sinusoidal frequencies. Rise and fall times of a pulse represent the very high frequency components.

Since any capacitance will lower the impedance of a circuit at high frequencies, shunt capacitance will attenuate the high frequency components of a pulse more than the lower frequencies.

Figure 7.1–5 illustrates the symbolic layout of all diode logic packages used in the Central Processor. Note that diode and resistors of some packages are not pre–wired as gates. This allows flexibility and a more efficient use of diodes. The number at the bottom of each square is the BCD part number for that particular diode package.

FIGURE 7.1–5 Diode Packages Used in Central Processor

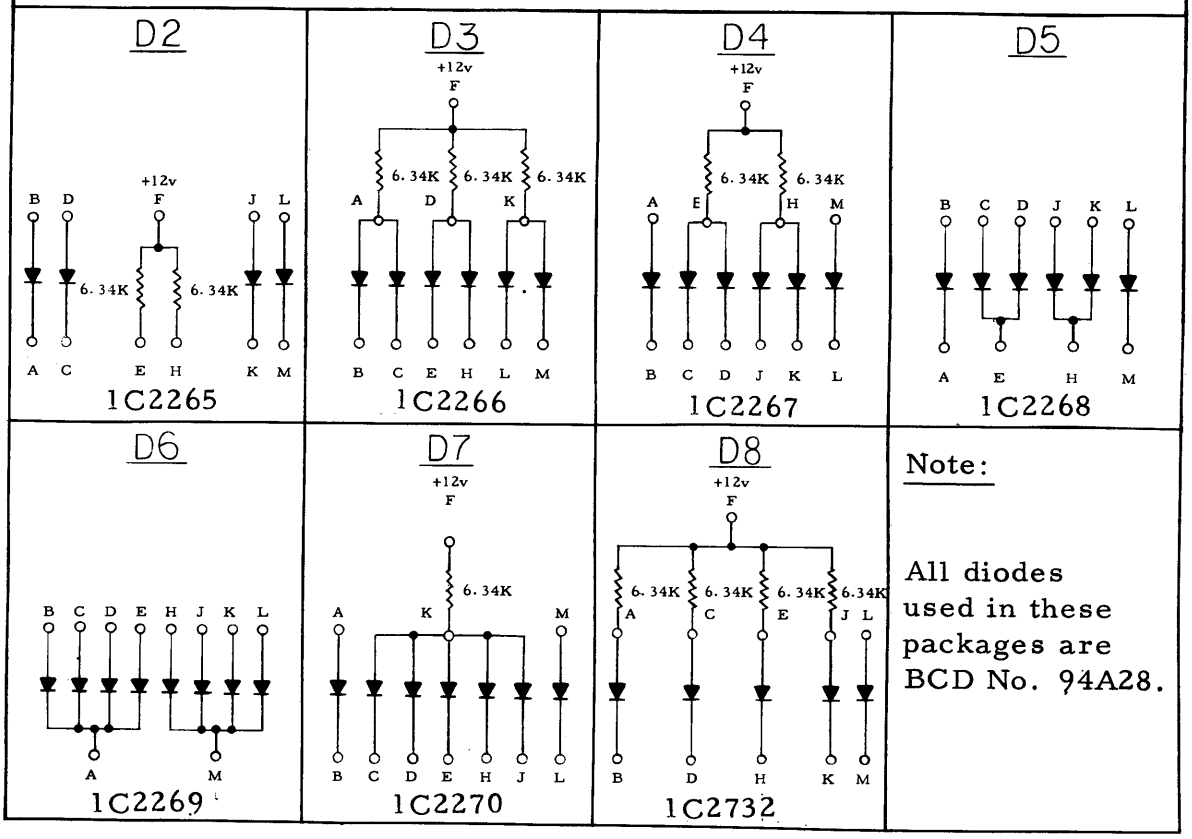
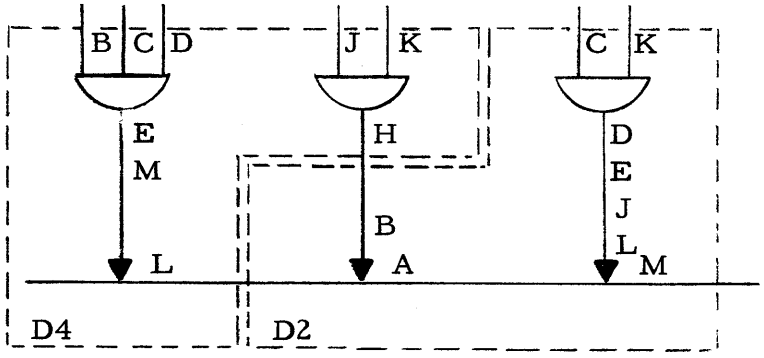


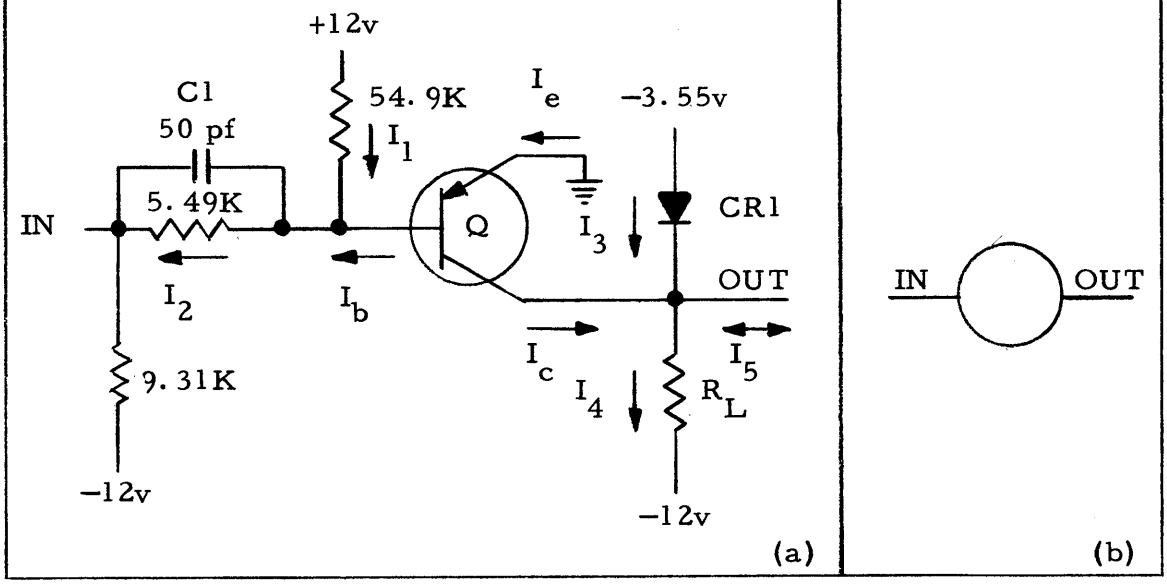
Figure 7.1–6 shows how diode logic performed by the diode packages would appear on a Central Processor logic schematic.

FIGURE 7.1–6 Typical Diode Gate Network



7.1–3 INVERTERS The G–20 inverter is a transistor circuit used to invert and amplify incoming signals. This circuit performs the Boolean NOT function. Figure 7.1–7 illustrates the circuit diagram of an inverter and the logic symbol for an inverter used in the Central Processor.

FIGURE 7.1–7 An Inverter (a) Electrical Circuit
(b) Logic Symbol



Assume that the voltage on the input terminal is zero volts and that the transistor is turned off. Then

$$I_b = 0$$

$$I_1 = I_2 = 12v / (54.9K + 5.49K) = 0.2 \text{ ma}$$

and

$$V_b = (0.2ma) \times (5.49K) = +1.1 \text{ volts}$$

Therefore, the base voltage is more positive than the emitter and the assumption that the transistor is turned off is validated.

Only a few microamperes of leakage current flow in the transistor leads. Ignoring CR1 and the −3.55 voltage source, if there were no external load, that is, if $I_5 = 0$, then

$$I_c = I_4 = 0 \text{ ma.}$$

would be the case, and the collector voltage, V_c, would be −12 volts.

Since the transistor collector breakdown voltage is limited to −6 volts maximum, CR1 has been added. This diode becomes forward biased and its current, I_3, clamps the output voltage, V_c, at approximately −3.7 volts. With no external load on this circuit, I_3 is equal to I_4. $I_3 = I_4 = [-3.7 - (-12)]/R_L$. If an external load, I_5, is applied to this inverter, then $I_3 = I_4 \pm I_5$. The direction of current flow of I_5 depends upon the type of load attached.

Let us now consider the inverter circuit if the input is −3.5 volts and assume that the transistor is saturated with a base voltage 0.3 volts more negative than the emitter. Then

$$V_b = -0.3v$$

$$I_1 = [+12 - (-0.3)]/54.9K = 0.224 \text{ ma}$$

$$I_2 = [-0.3 - (-3.5)]/5.49K = 0.584 \text{ ma}$$

$$I_b = I_2 - I_1 = 0.36 \text{ ma.}$$

We have shown that 0.36 ma of base current is flowing through the transistor base. For the transistor to saturate, the collector voltage V_c will be approximately equal to the base voltage V_b. Therefore, the output voltage V_c must be near zero volts.

$$I_3 = 0$$
$$I_c = I_4 + I_5 \text{ (OR)} - I_5 \text{ (AND)}$$
$$I_4 = 12/R_L$$

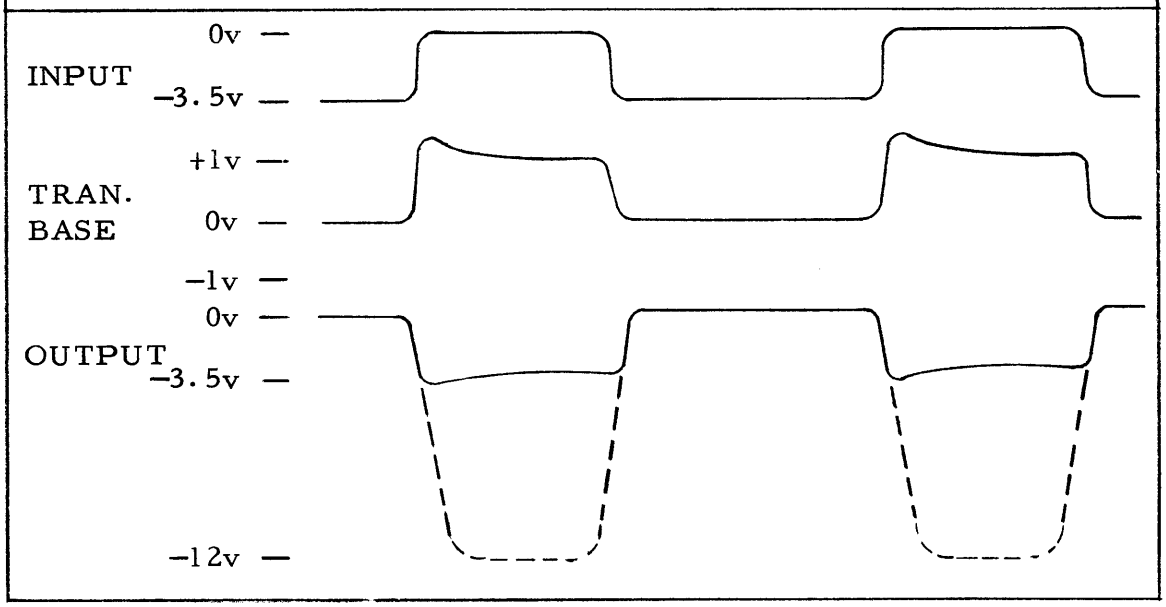
The diode current, I_3, is zero because the diode is reverse biased. The transistor current gain, β, must be at least equal to I_c/I_b where I_c is a function of the resistor R_L and the external gate load. The value of the transistor current gain and the resistor, R_L, determine how many AND–gates, OR–gates, or combinations of both may be driven by an inverter. These same variables determine what maximum length of wire may be tied to an inverter output. (Wire adds what is referred to as circuit capacitance to the inverter output and affects the rise and fall time of the signals at the collector.)

With a d.c. input signal of 0 volts or –3.5 volts we have found that the inverter output voltage became –3.5 volts or 0 volts, respectively. Relating these voltages once again to 1's and 0's we have found that with a 1 on the input we find a 0 on the output and vice versa. In this respect an inverter supplies the converse of its input signal. Figure 7.1–8 shows the voltage wave forms at the input, transistor base and output of a typical inverter.

Here again some time is required to switch from one voltage level to another. In fact, a period of time elapses from the time the input signal changes until any noticeable change occurs in the output. The rise and fall time of the output signal of the inverter is a function of the input signal, excess base current, transistor frequency response, and the circuit capacitance. The current gain of all transistors is

initially greater than the minimum β required. A safety factor has been allowed for circuit tolerances and aging effects. This means that excess base current will flow. The capacitor, C1, is used for low impedance coupling at high frequencies during turn–on and turn–off. C1 provides additional base current over and above the equilibrium value during the turn–on time and provides a current in the reverse direction to remove the excess stored charge during the turn–off time. Excess base current will reduce the turn–on time of a transistor and increase the turn–off time. For this reason C1 is often referred to as a speed–up capacitor.

FIGURE 7.1–8 Typical Inverter Voltage Levels



Notice the dotted portion of the output wave form of Figure 7.1–8. The inverter output would follow this curve if the clamp diode, CR1, were not supplied. Therefore, CR1 has effectively shortened the rise and fall time in addition to establishing the voltage level of the signal.

Figure 7.1–9 shows a schematic circuit of the basic Central Processor

FIGURE 7.1–9 Standard Central Processor Inverter Package Circuit

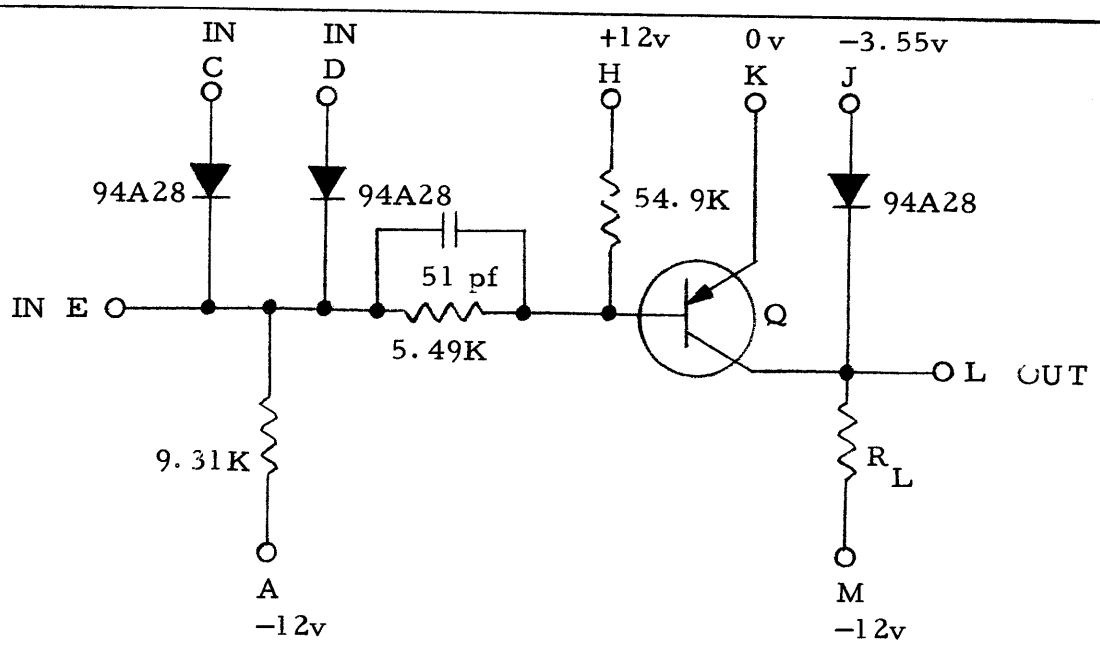
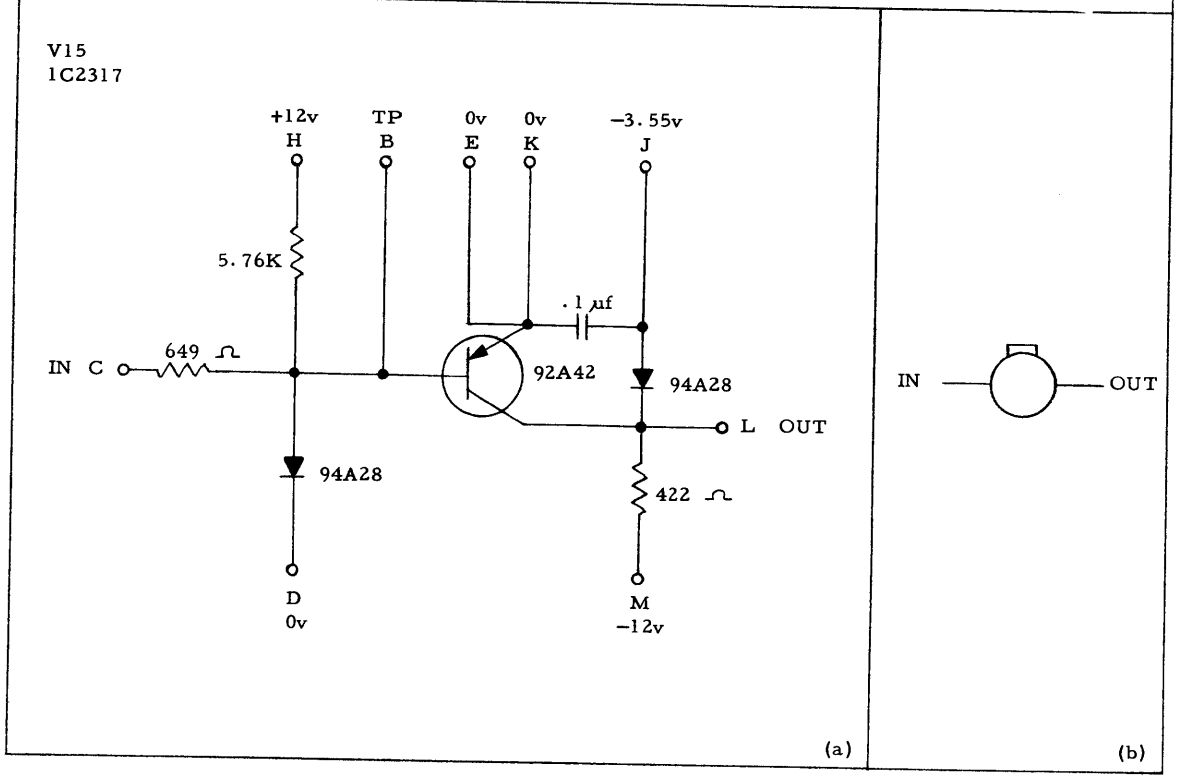


FIGURE 7.1–10 Clock Inverter (V15) (a) Electrical Circuit (b) Logic Symbol

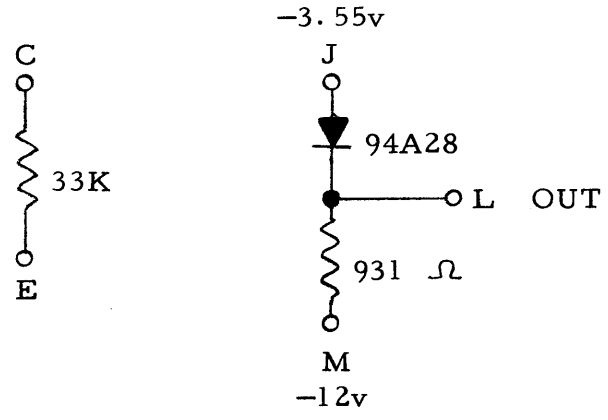


inverter package. Each standard inverter package has a built–in OR–gate attached to the inverter input. This arrangement was chosen because most inverters are driven by OR–gates in the G–20 System. Letters placed at various points on the schematic indicate pin letters on the package.

There are 14 standard inverter packages used in the Central Processor, and 12 of them (V01 through V14) are the same except that a different transistor and load resistor, R_L, are used. These inverters are of the same configuration as Figure 7.1–9. The V0X inverter is similar to Figure 7.1–9, except that a load resistor, R_L, is not provided in the package. The V15 is a special inverter known as the clock inverter and this is shown in Figure 7.1–10. This inverter requires more drive current and delivers more output power. This circuit is particularly convenient when it is necessary to drive several AND–gates with one signal.

Figure 7.1–11 shows the schematic of a DC2 package, an inverter diode clamp circuit, which is used as a load and a clamp for the V0X inverter. The output, L, is clamped to approximately –3.5 volts.

FIGURE 7.1–11 Inverter Diode Clamp Circuit (DC2)



The output of two or more inverters tied together forms what is referred to as a "Wired OR–gate". This connection is quite common in the Central Processor and is illustrated in Figure 7.1–12.

FIGURE 7.1–12 Inverters Connected to Form "Wired OR–gate"

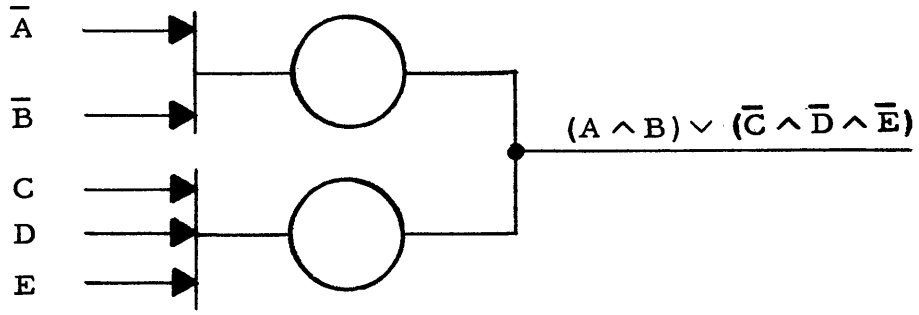
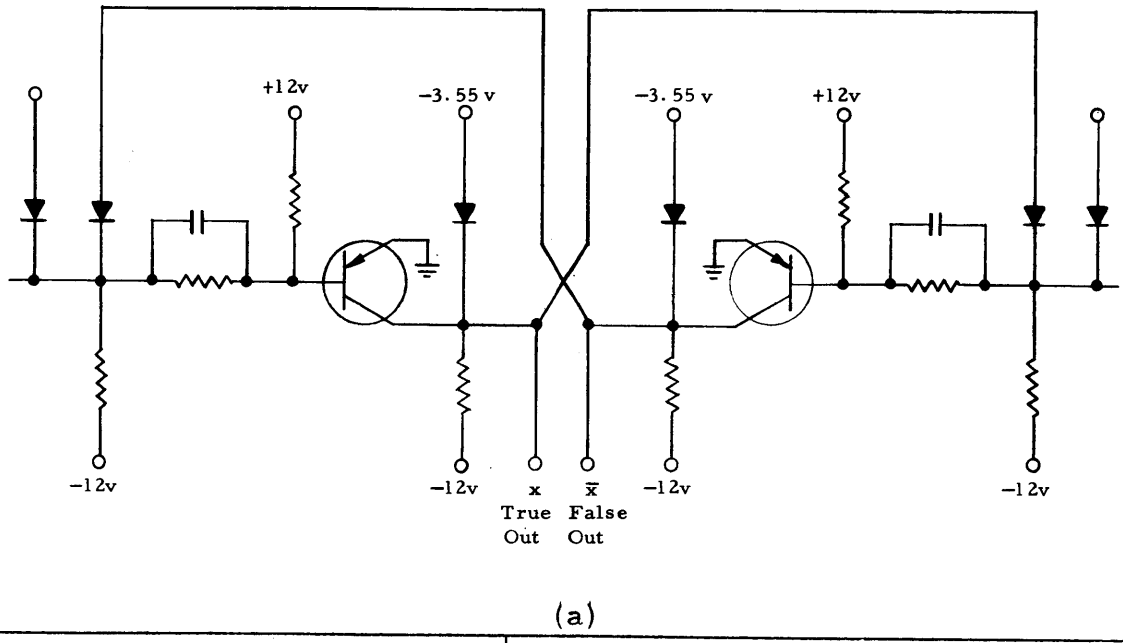


TABLE 7.1–2 Central Processor Inverter Packages

Pkg. No.	BCD No.	R_L	β Min.	Transistor Type	Diode Type
V01	1C2301	7.15K	58	92A41	94A28
V03	1C2303	7.10K	58	92A41	94A28
V04	1C2304	5.76K	73	92A43	94A28
V05	1C2305	2.10K	73	92A43	94A28
V06	1C2306	1.54K	73	92A43	94A28
V08	1C2308	1.87K	88	92A44	94A28
V09	1C2309	1.54K	88	92A44	94A28
V10	1C2310	1.24K	88	92A44	94A28
V11	1C2311	3.65K	112	92A45	94A28
V12	1C2312	1.87K	112	92A45	94A28
V13	1C2312	0.931K	112	92A45	94A28
V14	1C2314	1.24K	112	92A45	94A28
V15	1C2317	422Ω	60	92A42	94A28
V0X	1C2341	None	112	92A45	94A28

7.1–4 FLIP–FLOPS A G–20 flip–flop is constructed from two inverter packages as shown in Figure 7.1–13.

FIGURE 7.1–13 Flip–Flop (a) Electrical Circuit
(b) Logic Symbol (c) Truth Table



Input Set x^n	Input Reset x^n	True Out x^{n+1}	False Out $\bar{x}^{n+1}$
0	0	x^n	$\bar{x}^n$
1	0	1	0
0	1	0	1
1	1	Excluded	

$n = n^{th}$ state.

(c)

The state of this flip–flop is described as being SET (or logic 1) if the x inverter output is near zero volts, and RESET (or logic 0) if the x inverter output is near –3.5 volts. Figure 7.1–13 shows that the output of the x inverter feeds the input, via the OR–gate, of the $\bar{x}$ inverter. It follows that the output of the $\bar{x}$ inverter is the inverted form of the x inverter output. Figure 7.1–13 (c) represents the truth table for the

flip–flop. When both set and reset inputs are 0's, the outputs will remain in the previous state. The flip–flop is SET if one of the two inputs to the $\bar{x}$ inverter is 1. Therefore, the outputs of x and $\bar{x}$ will be 1 and 0 respectively. Similarly, the flip–flop is RESET if one of the two inputs to the x inverter is 1. Therefore, the outputs of x and $\bar{x}$ will be 0 and 1 respectively. At any given time, the flip–flop can only be either SET or RESET.

Figure 7.1–14 shows a flip–flop with two AND–gate inputs. The set and reset terms are:

$$\text{SET } x = z \wedge y$$
$$\text{RESET } x = z \wedge \bar{y}$$

FIGURE 7.1–14 Flip–Flop and Two AND–gates

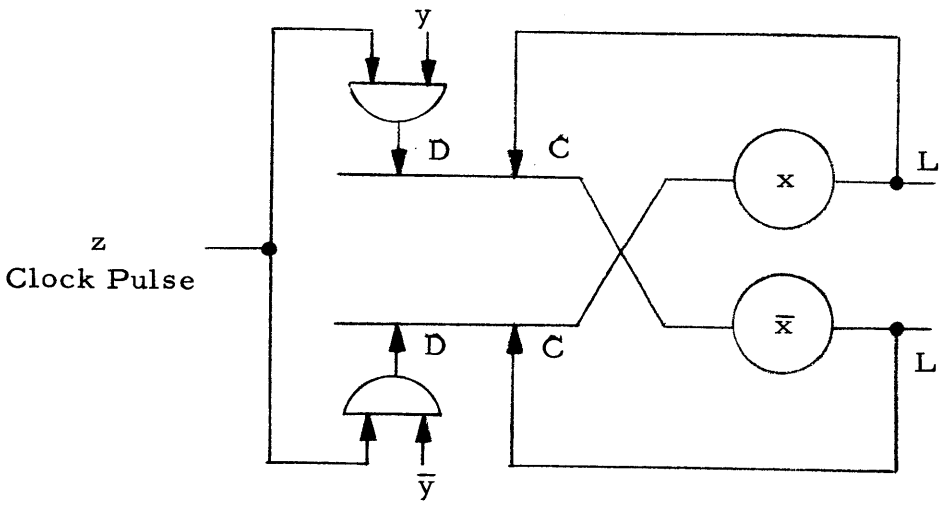
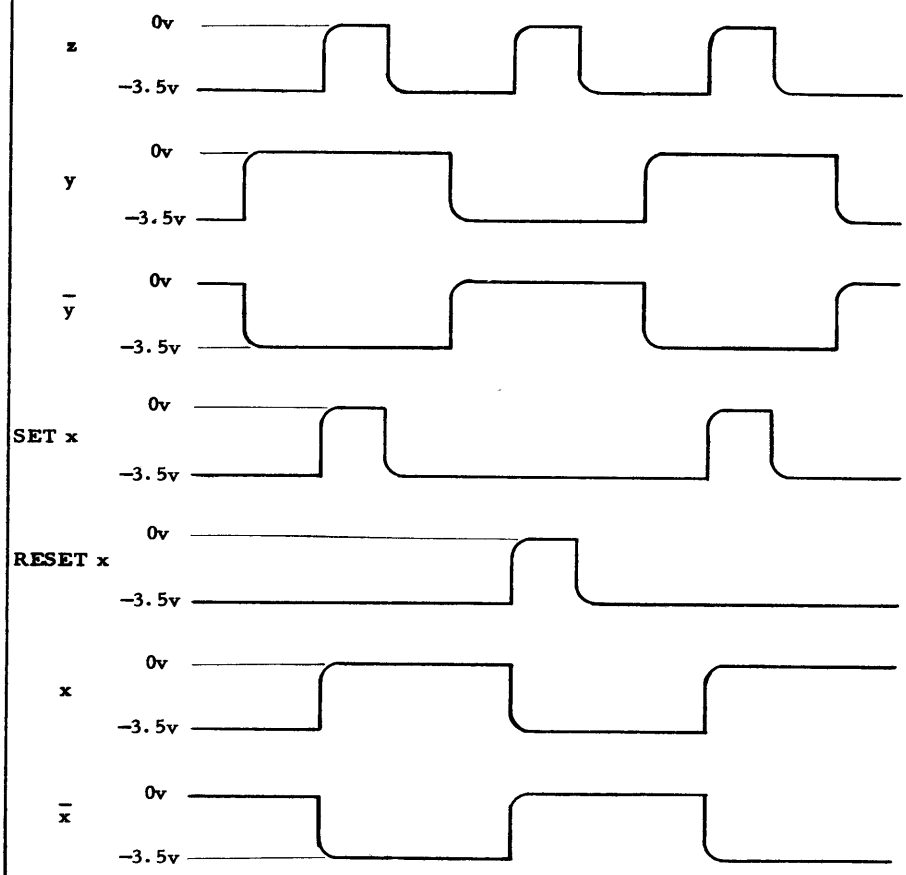


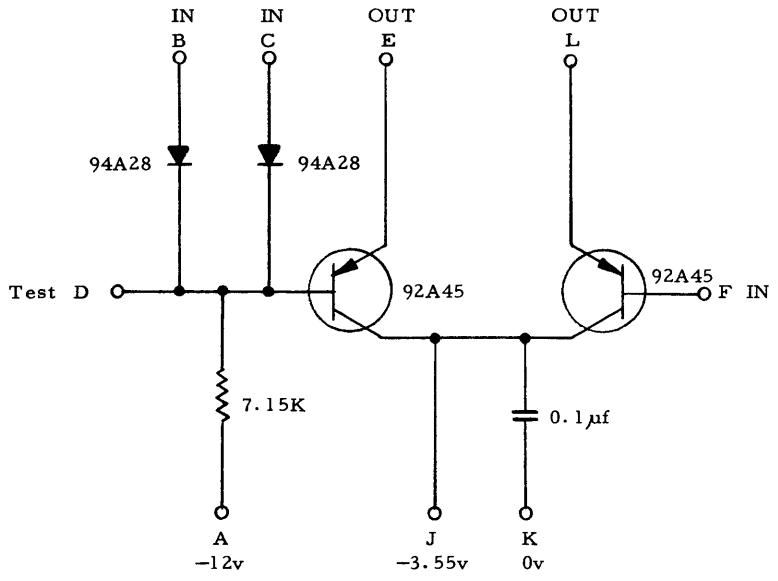
Figure 7.1–15 illustrates the voltage levels that would correspond to Figure 7.1–14.

FIGURE 7.1–15 Voltage Levels Corresponding to Figure 7.1–13



7.1–5 EMITTER FOLLOWERS An emitter follower is a circuit used when current gain is required without signal inversion. Figure 7.1–16 shows the circuit of the EF0, the Central Processor emitter follower package. Two emitter followers and one OR–gate are available on each package. The two transistors may be used as two separate emitter followers or as one emitter follower with a common input terminal, i.e., the two transistor base terminals can be connected together. Its input voltage levels are 0 or –3.55 volts nominal, while its output current is limited to 30 milliamperes maximum per transistor if separately driven.

FIGURE 7.1–16 Emitter Follower Package EF0 (1C2316)
(a) Electrical Circuit (b) Logic Symbol



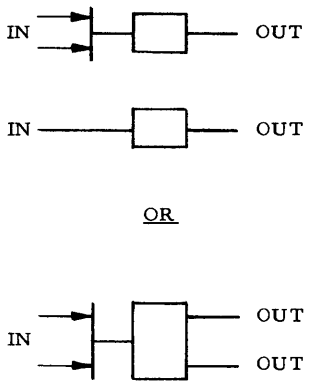
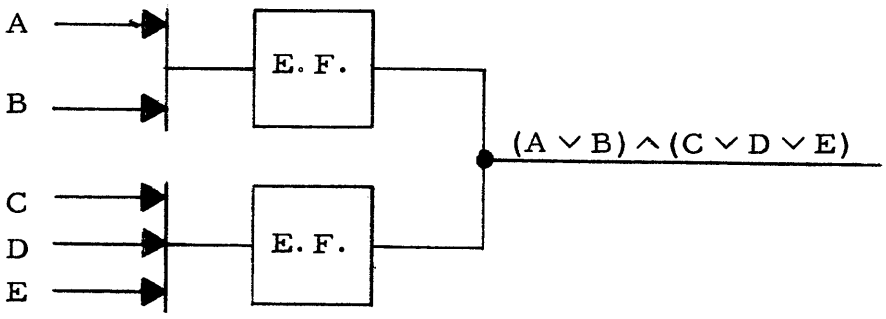
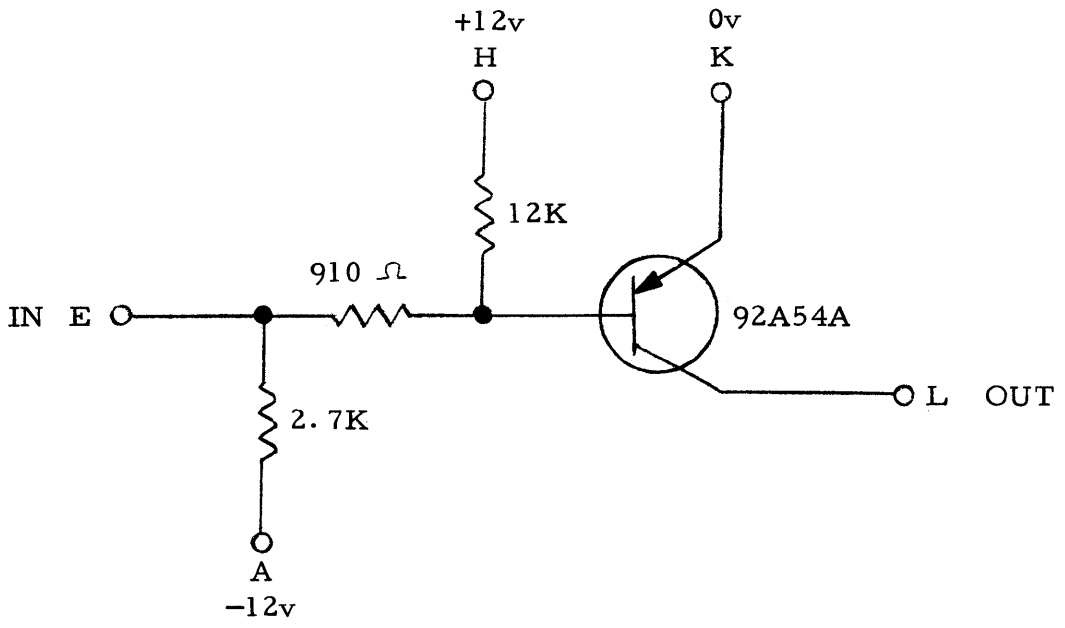


FIGURE 7.1–17 Emitter Followers Used as "Wired AND–gates"



An emitter follower normally performs no logic function. However, two emitter followers can be connected together to form what is referred to as a "Wired AND–gate". This circuit configuration finds limited use in the Central Processor.

FIGURE 7.1–18 Incandescent Bulb Driver Package CD1 (1C2823)



7.1–6 CURRENT DRIVER Figure 7.1–18 shows the circuit of CD1, a slow speed incandescent bulb driver. It is simply a transistor which is used to supply 0 volts to one terminal of the indicator light. When 0 volts are applied to the base of the transistor from the logic, the transistor will not conduct and the bulb will not light. When –3.5 volts are applied, the transistor permits flow of the current necessary to brilliantly light the bulb (at least 65 milliamperes). Its input voltage levels are 0 or –3.5 volts nominal. With a –3.5 volt input, the output is capable of supplying 100 milliamperes to the load.

SECTION 7.2 – COMMUNICATION SYSTEM CIRCUITS

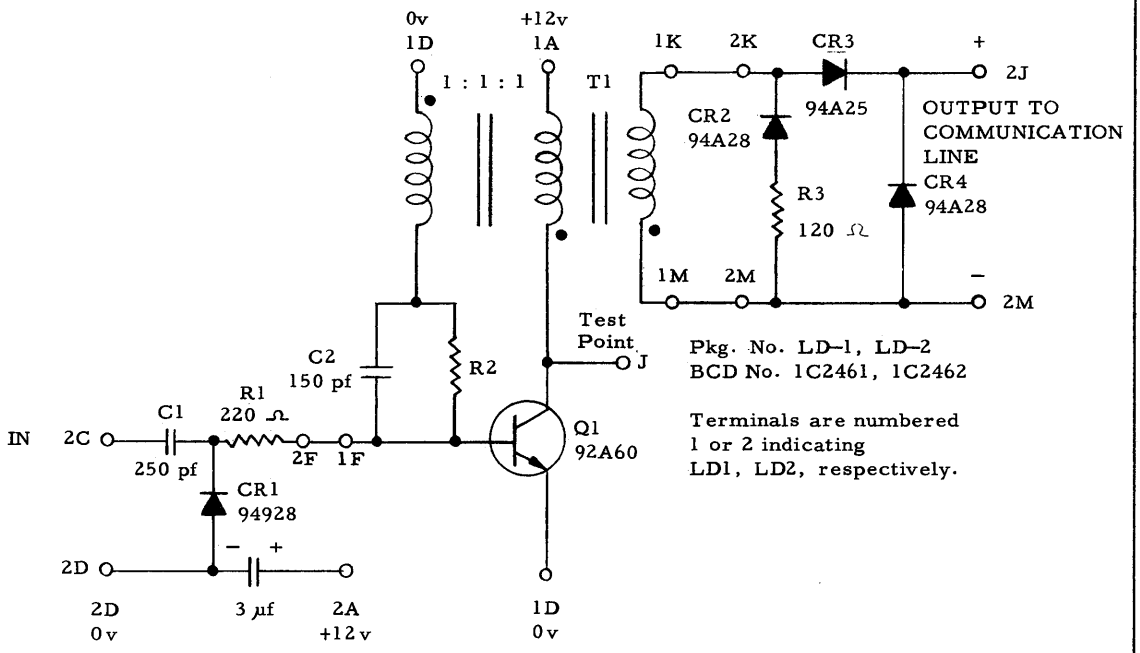
7.2–1 LINE DRIVERS The line driver circuit consists of two packages, LD1 and LD2 or LD3. Each line driver must generate a pulse of sufficient power to overcome the attenuation and noises of the Communication Line. Attenuation is caused by losses of the line and by reflections from discontinuities which occur where other line drivers and/or line receivers are connected to the line. The input to the line driver may be a positive going leading edge pulse, 2 to 8 volts in amplitude with a slope $\geq 7 \times 10^6$ volts/second. The line driver input is equivalent to a 3 OR–gate load as seen by the input signal source. Its output is a positive pulse ≥ 8 volts (pulse width nominally 0.30 microsecond). When a line driver is "driving" the line, it is working into an impedance of about 50 ohms (line Z0/2) since the Communication Line is terminated at each end with 100 ohms. Line driver circuits are, therefore, designed to operate into a 50 ohm load which is the resultant parallel resistance of the terminating resistors.

The line driver, shown in Figure 7.2–1, consists of an input coupling circuit, a blocking oscillator, and an output–to–line coupling circuit. In the quiescent state, Q1 is cut off since its emitter and base are at 0 volts.

A positive going trigger pulse at the input from C1 is fed through R1 to the base of Q1, bringing Q1 into conduction. As the collector current increases through the transformer, a voltage is developed across the base winding of the transformer T1 that swings the base even more positive. This regenerative action increases collector current until the charge on C2, the current through R2, and the feedback voltage are such that the collector current can no longer increase and Q1 is in full conduction. When current through T1 is no longer on the rise, voltage is not induced in the base winding. Since the base is no longer held

positive, current through Q1 begins to decline. The T1 base winding now swings the base negative and cuts off Q1.

FIGURE 7.2–1 Line Driver Circuit



The pulse width developed by the line drivers is determined primarily by the values of C2 and R2 and is approximately 0.30 microseconds.

After the useful portion of the pulse is passed, the energy remaining in the transformer must be dissipated. The recovery of T1 to its quiescent state is aided by the network CR2, R3. This network also damps T1 by a critical amount and prevents the base of Q1 from being brought positive by overshoot during the recovery period.

By way of the T1 output winding and the coupling circuit a pulse of energy has been coupled onto the Communication Line.

7.2–2 LINE RECEIVERS The line receiver circuit consists of two packages, LR1 and LR2 or LR3. Each line receiver is required to accept the transmitted pulse and generate a standard pulse for use with the Line Register. The line receiver also must be able to accept pulses of high power level as well as low and be able to discriminate reliably against noise.

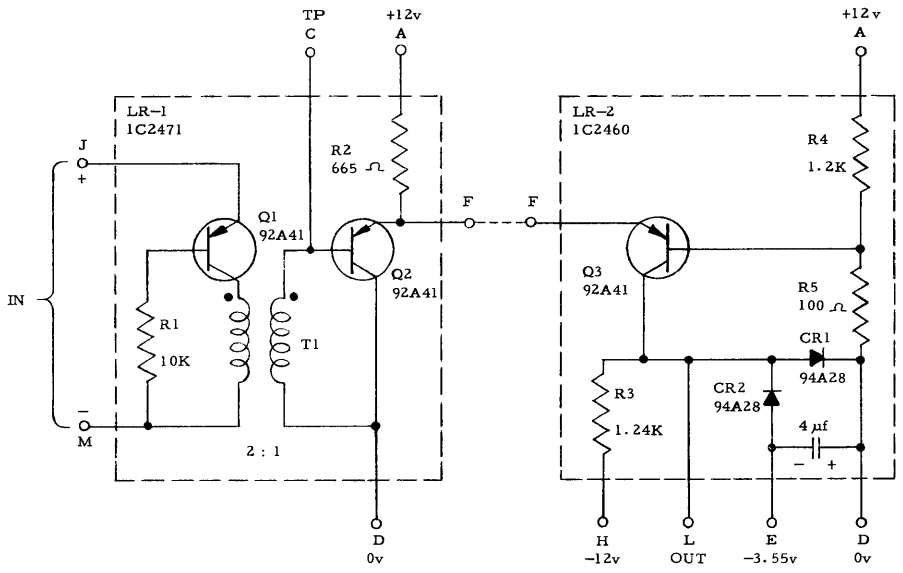
The input to the line receiver, LR1, LR2, is from the Communication Line with limits of $2.4 \leq E_{in} \leq 12$ volts. Its minimum pulse width is 100 nanoseconds and the maximum pulse width is 1 microsecond. The output of the line receiver is a pulse of −3.5 to 0 volts with a maximum rise time of 50 nanoseconds.

Figure 7.2–2 shows the line receiver circuit. During quiescent state, Q2 is rather heavily conducting, due to the grounded base and positive return on the emitter. With a little drop across Q2, its emitter, and the emitter of Q3 are very close to ground potential. The base of Q3 is held at about +0.9 volts by the voltage divider consisting of R4 and R5. Thus, Q3 is cut off, and its collector, the output of the receiver, is low (return to −12 volts through R3, but clamped at −3.5 volts by CR2).

Q1 acts as an isolater, which will prevent the recovery current of T1 from feeding back into the Communication Line.

When the signal arrives at the input of T1 (from either direction on the line or from the line driver, but always with the same polarity), the base of Q2 is driven positive. The emitter of Q2 "follows" the base positive, carrying the emitter of Q3 positive until clamped to +1.2 volts by the base voltage on Q3 (+0.9) volts. As the base voltage of Q2 continues to rise, cutting Q2 off, Q3 conduction now raises the output to near ground where it is clamped by CR1.

FIGURE 7.2–2 Line Receiver Circuit



SECTION 7.3 – CLOCK GENERATOR CIRCUITS

7.3–1 INTRODUCTION The Central Processor Clock Generator produces two phase clock pulses termed C1 and C2. These pulses alternate in time, with the C2 pulses falling between those of C1. (The timing of these pulses is shown in Figure 7.3–3.) The Clock Generator itself is divided into the following basic sections:

- Clock Generator
  - Timing Section
    - Frequency Source Generator
    - Dual Purpose Flip–flop
  - Clock Pulse Generator Section
    - Clock One Pulse Generator
    - Clock One Pulse Amplifier
    - Clock Two Pulse Generator
    - Clock Two Pulse Amplifier
  - Start–Stop Logic Section

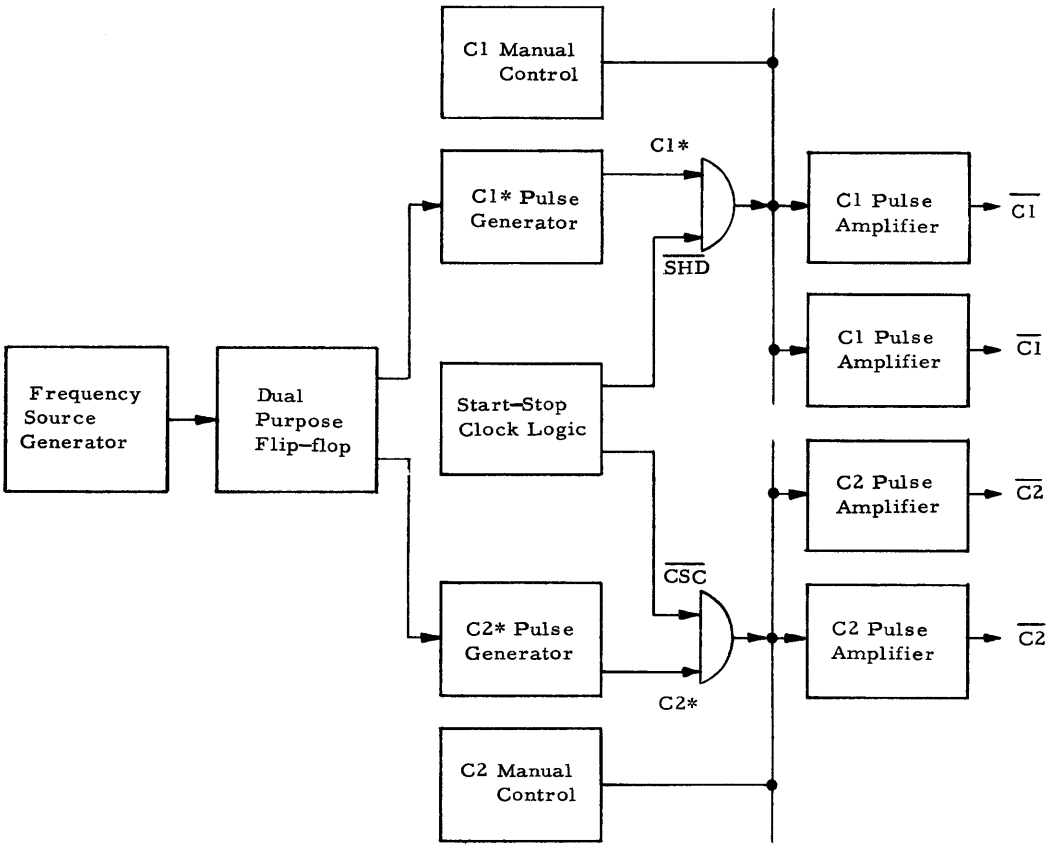
Figure 7.3–1 presents this information in block diagram form and Figure 7.3–2 shows the actual circuit configuration of the Clock Generator.

There are three modes of Clock Generator operation: fixed frequency, variable frequency, and manual control. These modes of operation are controlled by the CLOCK MODE and FREQUENCY MODE Switches located on the Central Processor control panel.

The Clock Generator normally operates in the fixed frequency mode. In this mode clock pulses are produced at a 1 megacycle rate, with a 0.35 microsecond width. The pulse width can be adjusted internally through the use of trimmer potentiometers. The Clock Generator is put in the fixed frequency mode by setting the FREQUENCY MODE Switch to FIXED, and the CLOCK MODE to RUN. In this operating mode, there are no external controls for the frequency, symmetry, or

pulse width. The frequency source generator operates at a fixed frequency, established by a quartz crystal, which is twice the basic clock frequency. The output pulses of this frequency source generator are fed to the dual purpose flip–flop. One side of this flip–flop feeds the C1 clock pulse generator, and the other side feeds the C2 clock pulse generator. Thus, the C2 pulses are precisely halfway between the C1 pulses.

FIGURE 7.3–1 Block Diagram of Central Processor Clock Generator

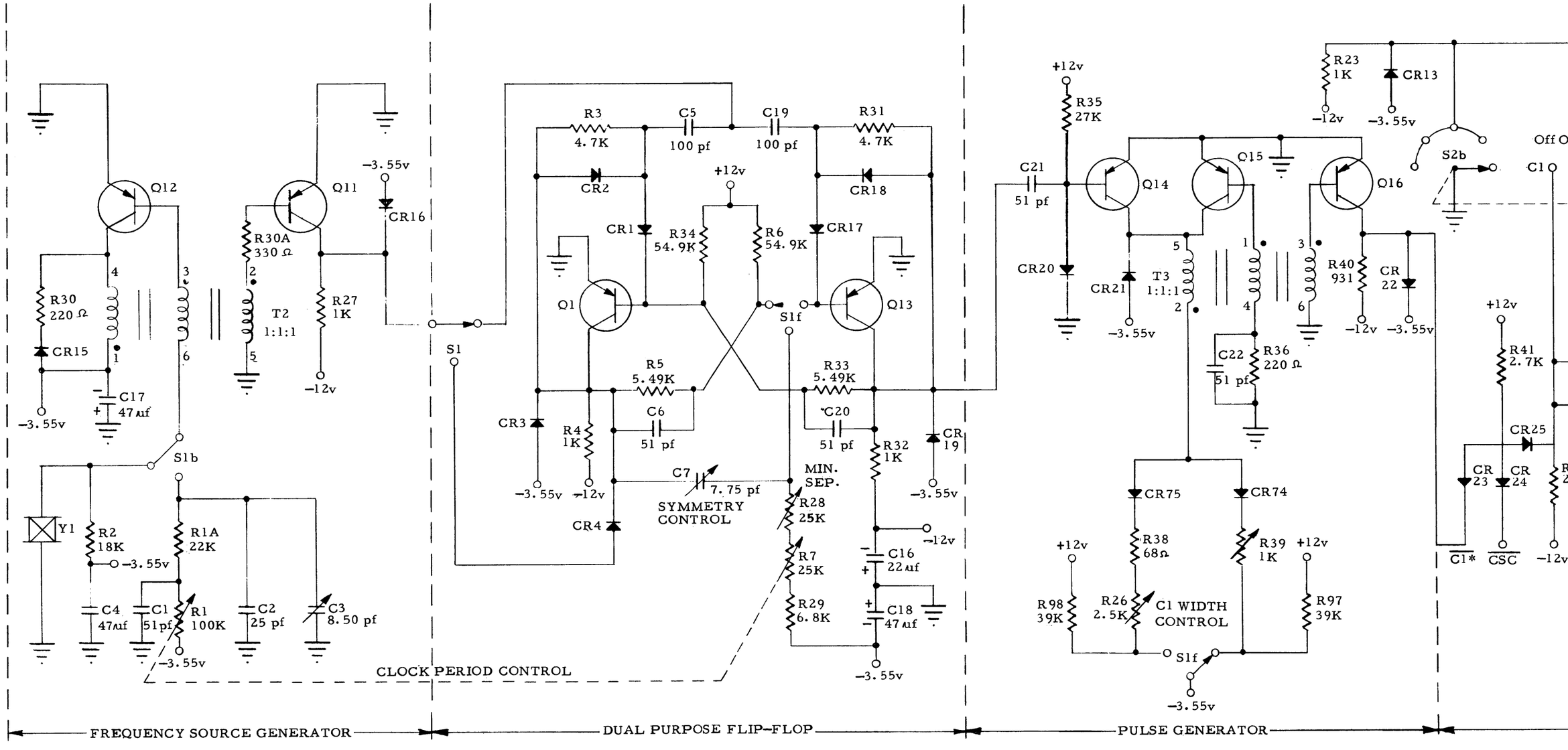


The variable frequency mode is used for marginal checking and maintenance. The Clock Generator is put in the variable frequency mode by setting the FREQUENCY MODE Switch to VARIABLE, and the CLOCK

MODE Switch to RUN. In this mode the frequency source generator is manually adjustable from approximately 400 kc to 1400 kc. With an MM–10 unit in the system, the frequency can only be adjusted ± 10 per cent from the 1 megacycle clock rate and still maintain proper clock pulses in the MM–10. The clock pulses in the MM–10 are based on a fixed delay and, if the frequency is varied appreciably, pulse synchronization is lost. In the variable frequency mode, the dual purpose flip–flop operates as a one–shot multivibrator, with the on time determining the delay between C1 and C2. The delay control is mechanically coupled to the CLOCK PERIOD control so that C2 is maintained in approximately the same relative position with respect to C1 throughout the range of its operation. At any one setting, the symmetry may be controlled by the manual SYMMETRY control. Also, the Central Processor clock pulse widths are each manually adjustable, by means of the C1 and C2 WIDTH controls, between 0. 15 and 0. 4 microsecond. The MM–10 clock pulse widths are fixed at 0. 35 microsecond.

In the manual control mode of operation, the output of the Central Processor Clock Generator is a d. c. level. It is used to trace out logical signals, to check flip–flop states, etc., at any point in a program. It is possible, in trouble–shooting the Central Processor, to check each state of the computer by stepping it, one clock pulse at a time, through any program or section of a program. The Central Processor Clock Generator is put in the manual control mode from either the variable frequency mode or the fixed frequency mode by turning the CLOCK MODE Switch to STOP. When neither C1 nor C2 is present, both outputs of the Clock Generator ($\overline{C1}$ or $\overline{C2}$) are maintained at 0 volts. The CLOCK MODE Switch is then turned to OFF position. When the CLOCK MODE Switch is then turned to either the C1 or C2 position, the indicated clock pulse at the output of the Clock Generator

FIGURE 7.3–2 Central Processor Clock Circuit (C2 Pulse Amplifier Not Shown)



(C2 Pulse Amplifier Not Shown)

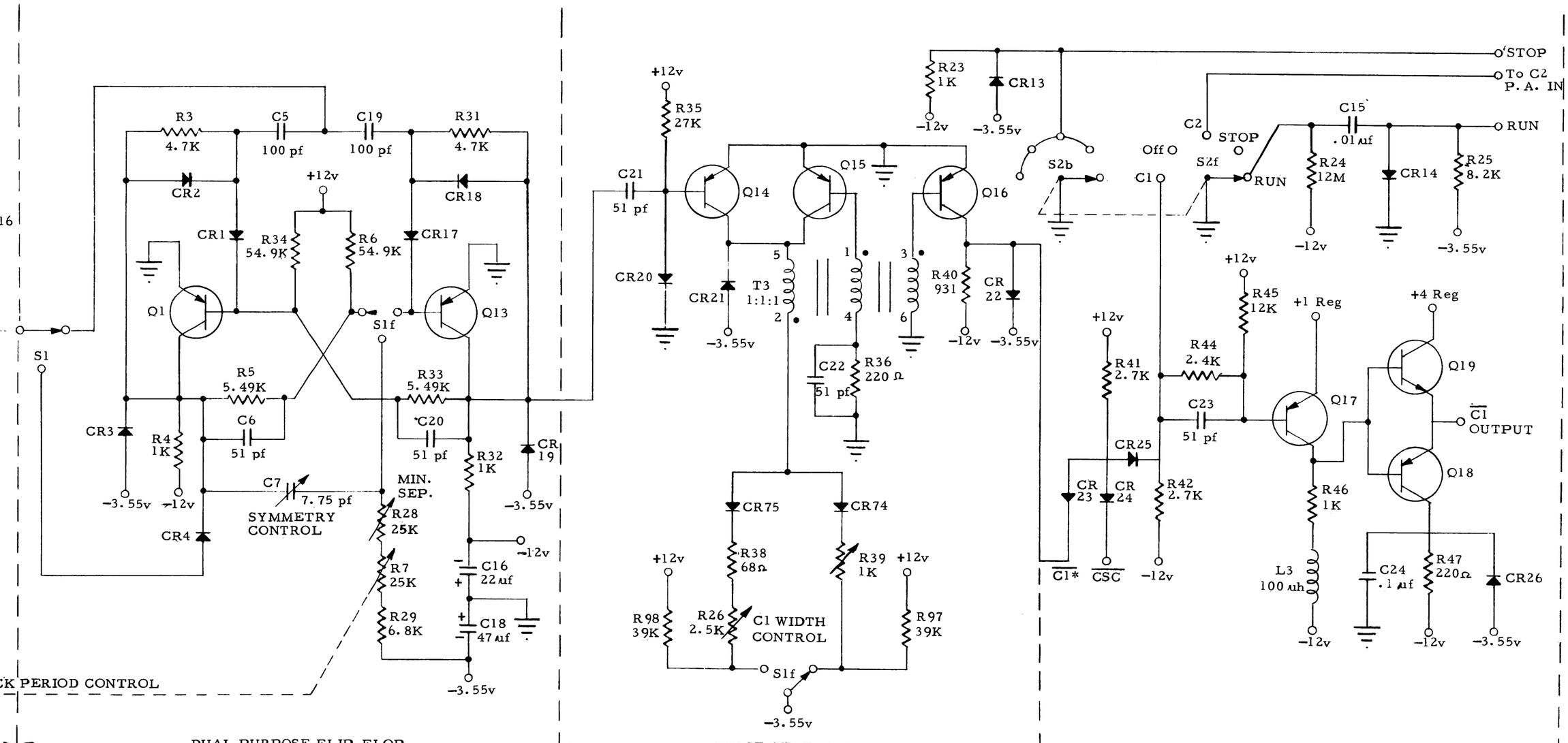
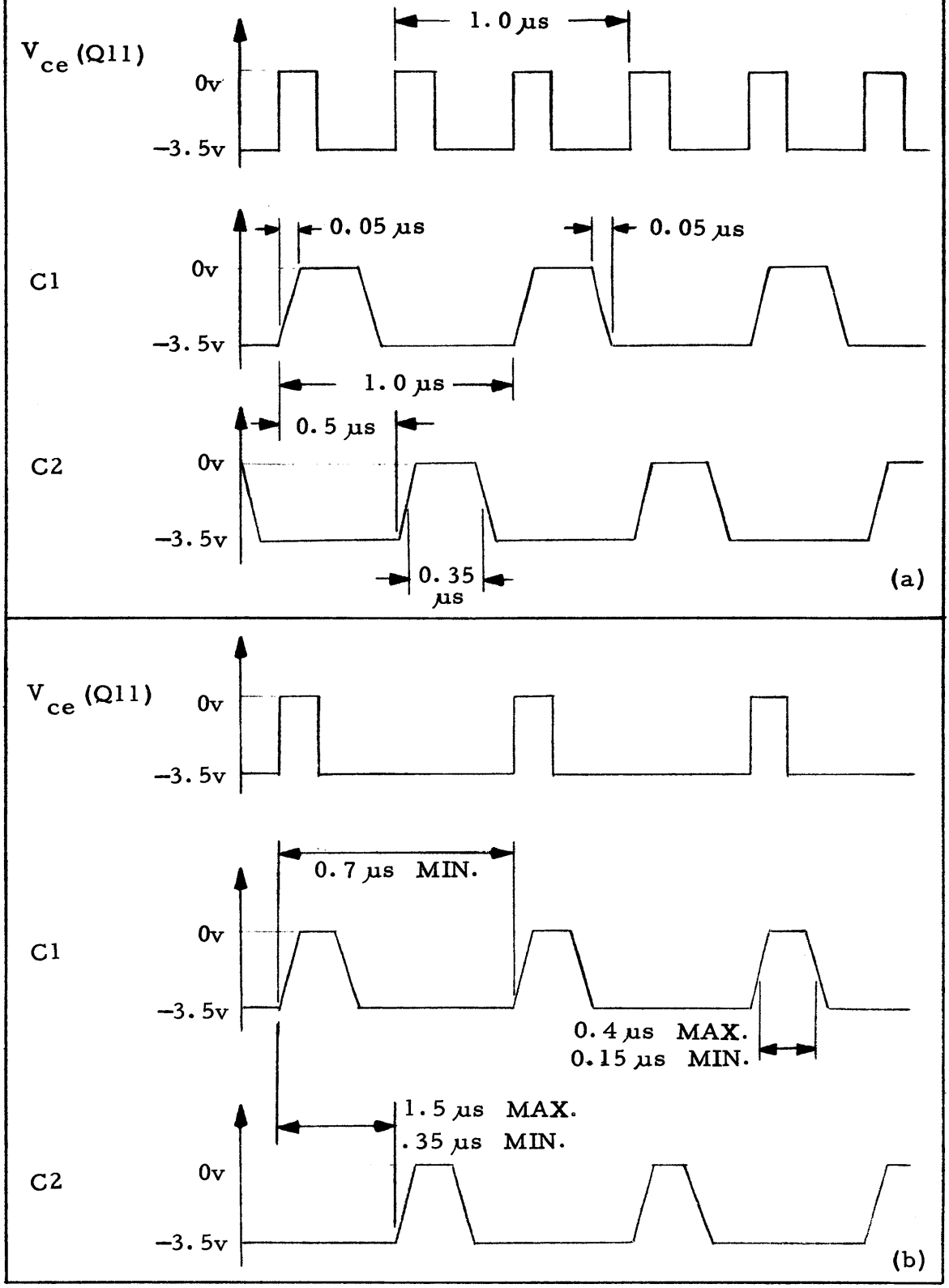


FIGURE 7.3–3 C1 and C2 Pulses: (a) Fixed Frequency (b) Variable Frequency



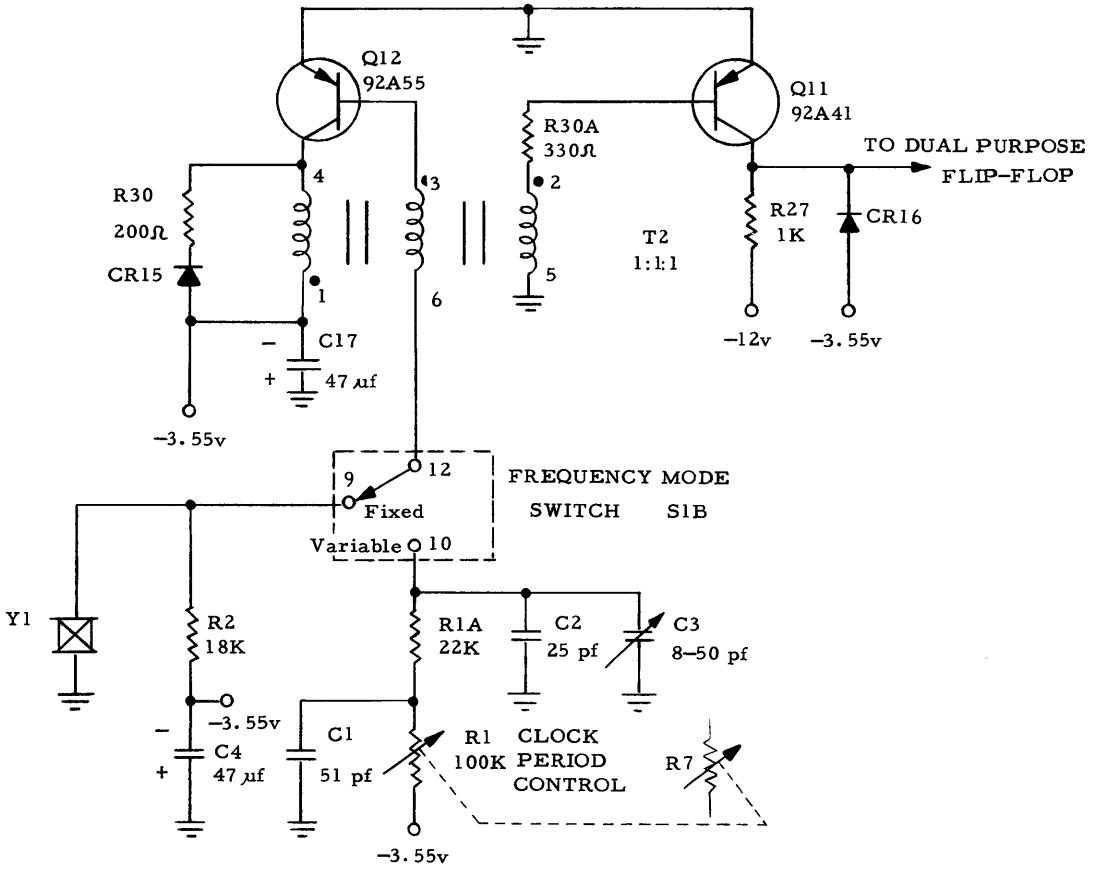
($\overline{C1}$ or $\overline{C2}$) is maintained at –3.55 volts and the other at 0 volts. The Clock Generator is taken out of the manual control mode by turning the CLOCK MODE Switch back to RUN. Manual operation of the Central Processor clock also provides manual clock control to the MM–10 units. (A pulse generated by manual control of the Central Processor Clock Generator causes a pulse to appear at the MM–10's, but this pulse is not maintained at 0 volts as in the Central Processor.) The MM–10's are equipped with a hand clock switch which can provide manual clock control to the MM–10 without affecting the Central Processor clock. This switch provides a constant C1 or C2 pulse within the MM–10 unit. This hand clock switch is primarily intended for off–line debugging of the MM–10 units.

7.3–2 FREQUENCY SOURCE GENERATOR The frequency source generator circuit is shown in Figure 7.3–4.

The frequency source generator is operated in either the fixed or variable frequency mode, depending upon the position of the FREQUENCY MODE Switch. In the variable frequency mode, the FREQUENCY MODE Switch is connected to point 10 of Figure 7.3–4.

At the time transistor Q12 turns on, collector current in the 1 – 4 winding of T2 induces current in the 3 – 6 winding with a polarity which aids the turn on of Q12. In this manner, Q12 is driven into saturation very rapidly. Capacitor C1 is a filter which is used to minimize the noise due to the lead connecting R1A to R1. Capacitors C2 and C3 charge up via the base current to a potential equaling the feedback potential, reducing the drive to Q12. As the collector current is reduced due to reduced base current, the induced currents drop to zero, at which time the polarity of the 3 – 6 winding reverses, initiating a rapid turn–off of Q12. When the capacitors have discharged, via R1 and R1A, the cycle can then be repeated. The frequency of the gener–

FIGURE 7.3–4 Frequency Source Generator Circuit

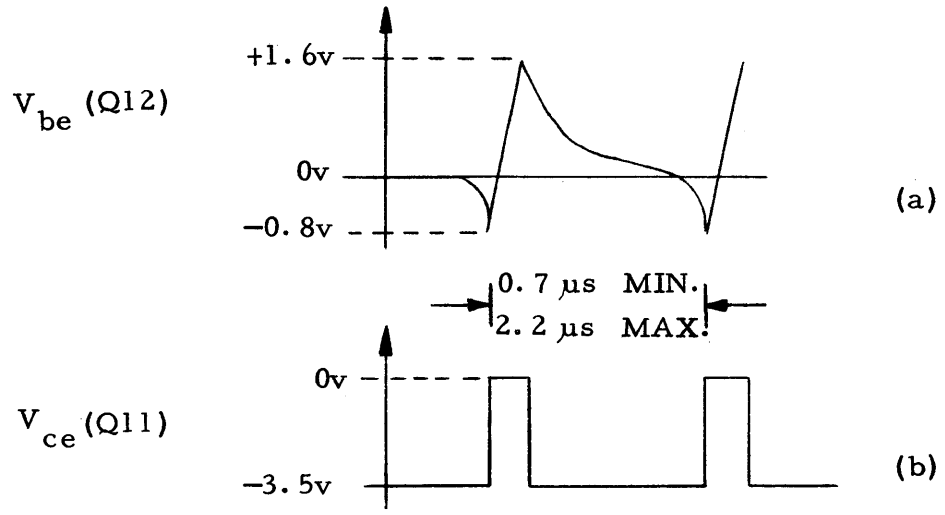


ator can be controlled by part of the CLOCK PERIOD control (R1 of Figure 7.3–4) which determines the time necessary to discharge the capacitors.

In the fixed frequency mode, the FREQUENCY MODE Switch is connected to point 9 of Figure 7.3–4. When crystal Y1 is excited into oscillation, the resultant voltage across R2 triggers the blocking oscillator at a fixed frequency of 2 megacycles.

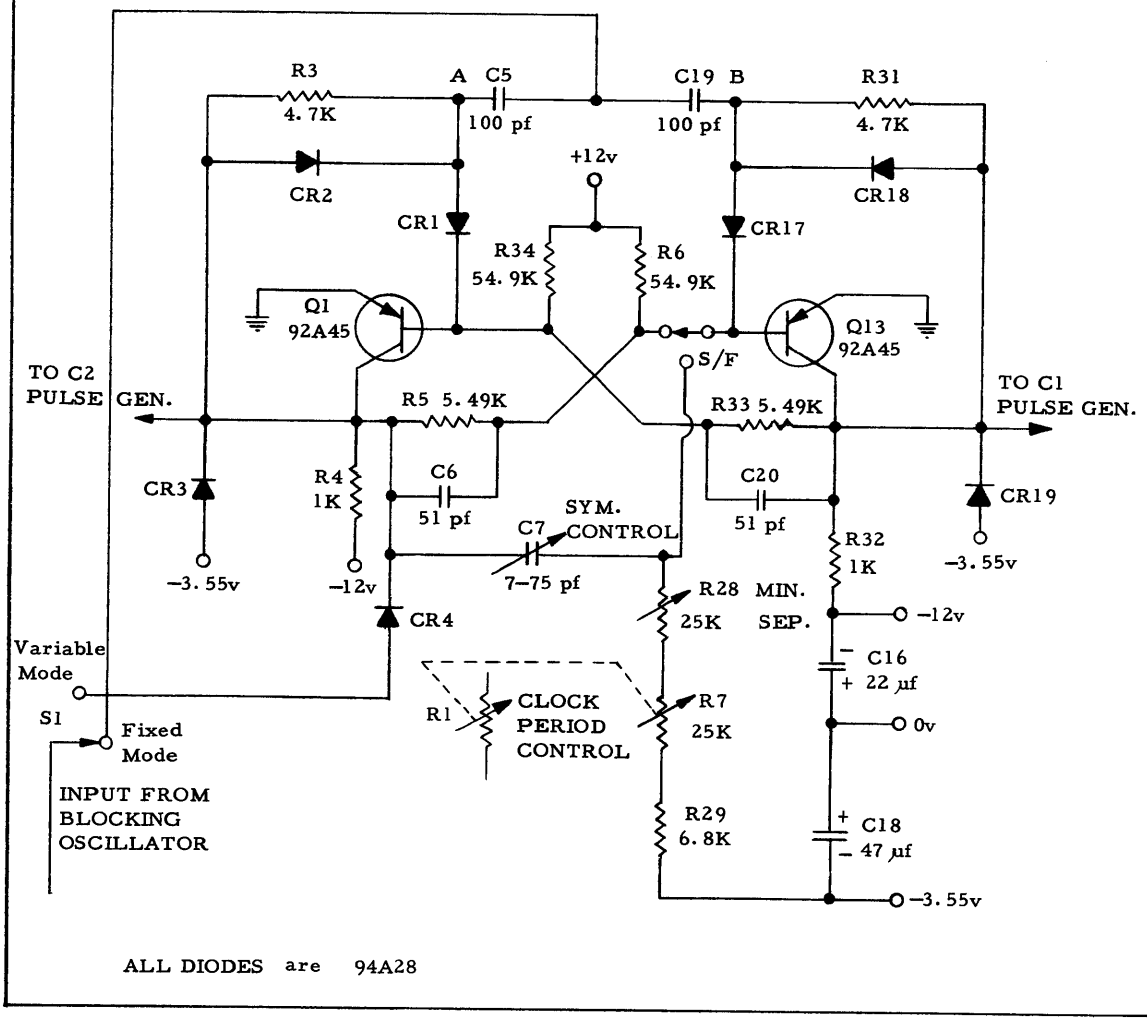
In both modes of operation, winding 2 – 5 of T2 picks off the blocking oscillator pulses, which are then inverted by transistor Q11. The wave shapes at the base of Q12 and the collector of Q11 for variable frequency mode operation are shown in Figure 7.3–5.

FIGURE 7.3–5 In the Variable Frequency Mode (a) Base Voltage of Q12 (b) Collector Voltage of Q11



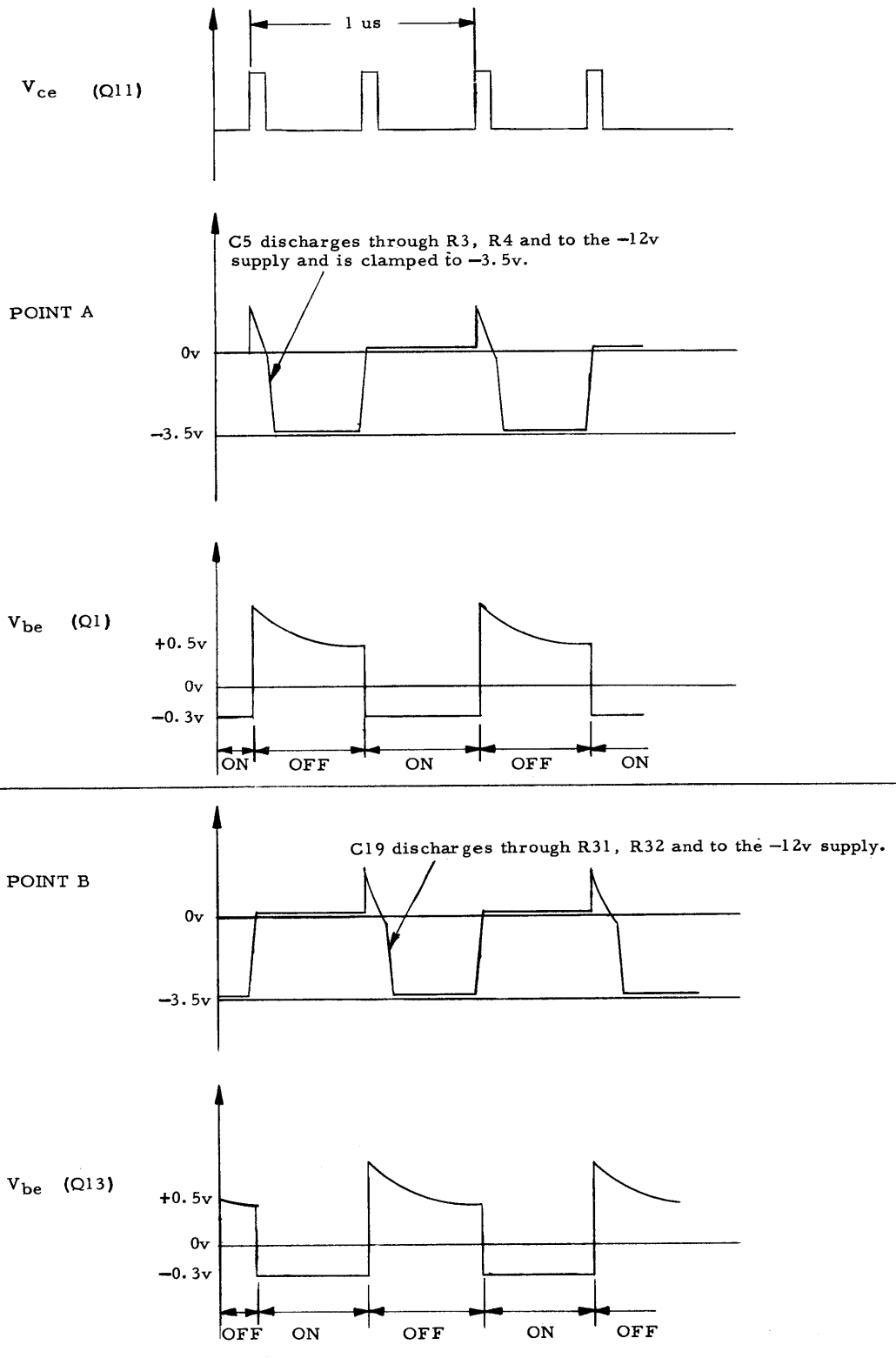
7.3–3 DUAL PURPOSE FLIP–FLOP The dual purpose flip–flop, composed of transistors Q1 and Q13, is shown in Figure 7.3–6.

FIGURE 7.3–6 Dual Purpose Flip–Flop



During the fixed frequency operating mode, the DPFF is alternately triggered from one stable state to the other by the train of trigger pulses from transistor Q11 of the frequency source generator. Through the contacts of S1, these pulses are fed to the junction of C5 and C19. If Q1 is in conduction, the near ground potential of the collector of Q1 is fed back to the base of Q13 via capacitor C6 and resistor, R5, thereby holding Q13 cut–off. The positive going pulse from the frequency source generator goes through C5 and CR1 and cuts–off Q1. As the collector of Q1 drops to –3.5 volts, clamped by

FIGURE 7.3–7 Commutating Voltages of Clock Generator



the diode CR3, the negative potential, applied via the collector–to–base coupling of Q1 and Q13, turns Q13 on. The next positive going pulse from the frequency source generator will go through C19 and CR17 and turn Q13 off. The negative potential on the collector of Q13, applied via the collector–to–base coupling, composed of C20 and R33, turns Q1 on. The commutating voltages are the voltages at points A and B of the Dual Purpose Flip–Flop circuit in Figure 7.3–6. In Figure 7.3–7, the voltage curves at points A and B are identical but with a time displacement of 1/2 microsecond. The fall of the commutating voltages is controlled by the time constants C19, R31 or C5, R3.

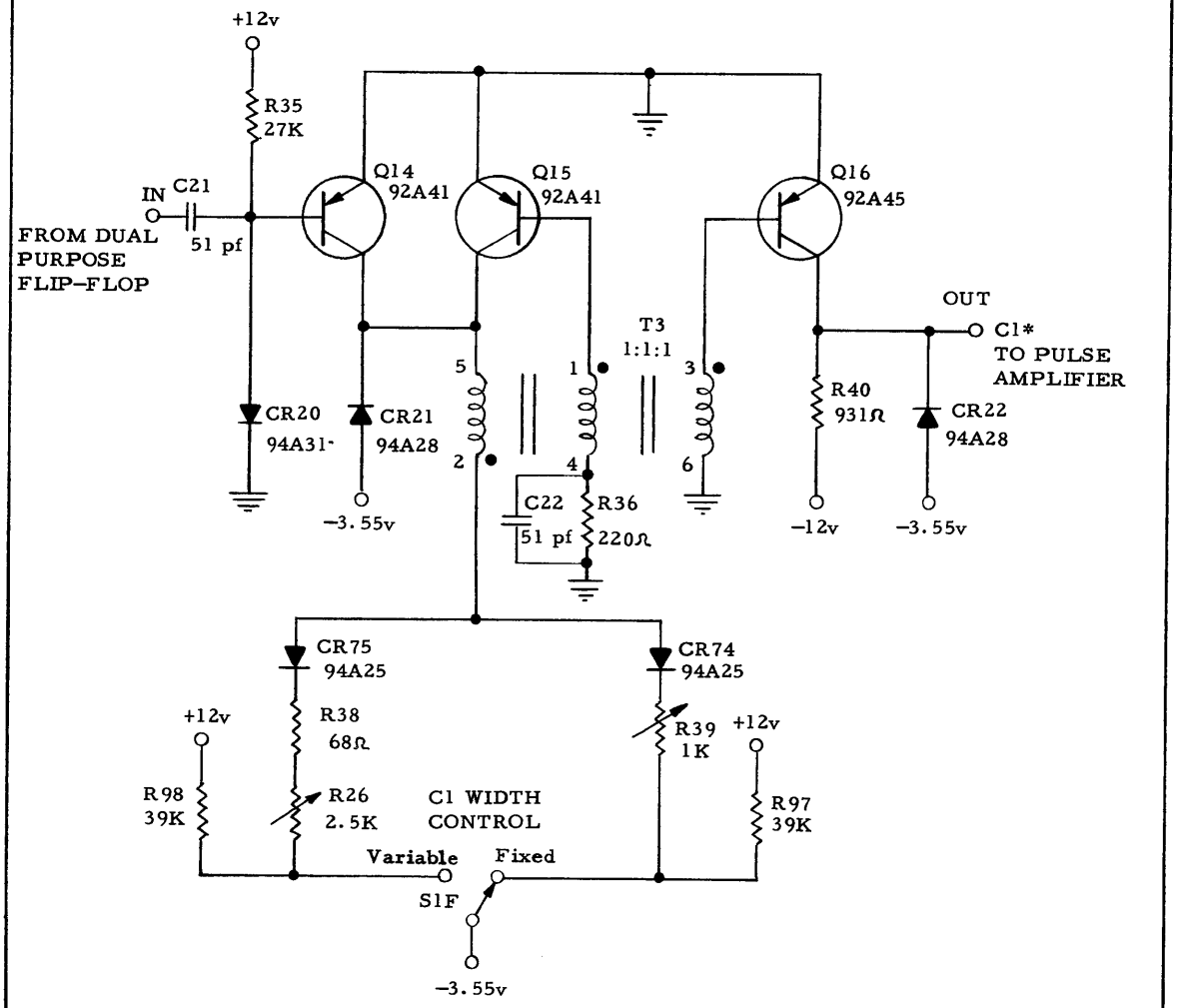
When it is in the variable frequency operating mode, the DPFF is connected as a one–shot multivibrator triggered by transistor Q11 of the frequency source generator. Transistor Q13 is held in conduction by R7, R28 and R29, Q1 is cut off and a positive going pulse from transistor Q11, via switch S1 and the symmetry capacitor C7, turns transistor Q13 off. The negative potential on the collector of Q13 in the cut–off state (clamped to –3.5 volts) is coupled back to Q1 through C20 and R33 to turn Q1 on. C7 discharges through R28, R7 and R29 until the base voltage of Q13 is low enough to cause Q13 to turn on. As Q13 turns on, the positive going potential from the collector is fed back to Q1, turning Q1 off and completing the cycle.

The clock two (C2) delay is controlled by the time constants of C7 and R7, R28 and R29. R7 is coupled mechanically to the frequency source timing resistor R1 so that symmetry between clock one and clock two is held reasonably constant when the frequency is changed. The symmetry itself is controlled by manually adjusting the SYMMETRY control C7. R28 is provided to set a minimum separation between a C1 and a C2 pulse under variation of the CLOCK PERIOD control.

7.3–4 CLOCK PULSE GENERATOR The clock one and clock two

pulse generators and pulse amplifiers are identical. Therefore, only the circuits for the clock one pulse generator will be discussed. Figure 7.3–8 shows the clock one pulse generator circuit.

FIGURE 7.3–8 Clock One (C1) Pulse Generator Circuit



The input to the clock pulse generator is a negative going pulse (0 to −3.5 volts nominal) which is coupled by C21 to the base of Q14. Prior to the application of the input pulse, transistor Q14 is off with the base of Q14 maintained at approximately +0.5 volt. This voltage is developed across diode CR20 by the current flowing through R35 from

the +12 volts supply. Diode CR20 is a controlled forward voltage unit.

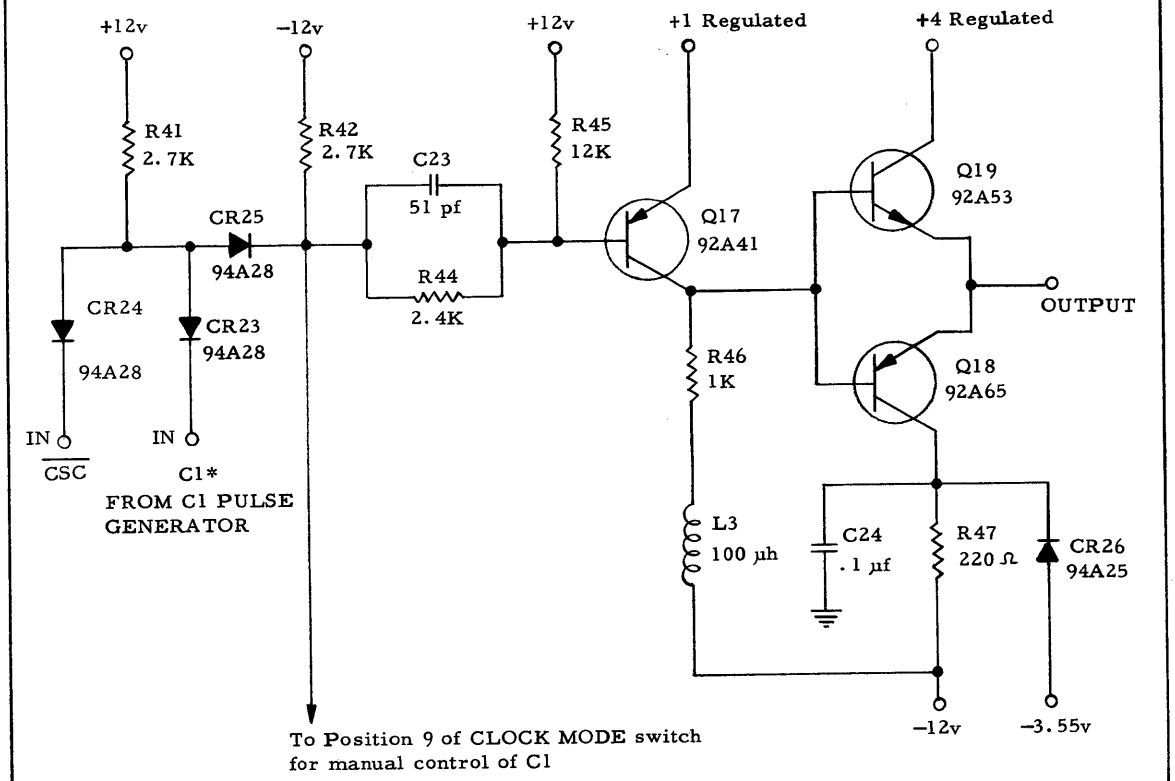
At this time the application of the negative going pulse, from the output of the dual purpose flip–flop, turns the trigger on (i.e., Q14 is on). This triggers the blocking oscillator which is comprised of Q15 and T3. Q14 raises the voltage on the collector of Q15 and causes current to flow in the primary of T3 (5–2 winding). A current is induced in the 1 – 4 winding of the transformer which flows out of the base of Q15. This, in turn, turns Q15 on. Thus, regenerative action is established and Q15 is maintained in the <u>on</u> condition after the trigger is removed. The current pulses formed by the blocking oscillator are also induced into the 3 – 6 winding of T3 and turn Q16 on. During the time that Q15 is on, the collector current is increasing. But when the maximum current allowed by R26 or R39 is reached, (depending upon whether switch 1 is in the VARIABLE or FIXED FREQUENCY MODE position), the change in current approaches zero. Consequently, the induced currents in the 1 – 4 and 3 – 6 windings approach zero, at which time the polarity reverses and Q15 and Q16 are driven to cut off.

After going through Q16, the pulse generator output is a rectangular pulse. The output pulse of Q16 is referred to as $C1^*$ and the output pulse of Q4 is $C2^*$. Figure 7.3–3 illustrates these pulses. Clock pulses $C1^*$ and $C2^*$ cannot be turned off as long as power is on in the G–20. However, if the CSC signal is high, the $C1^*$ and $C2^*$ clock pulses are prevented from being applied to the respective pulse amplifiers and, thus, prevented from becoming C1 and C2 (see Sections 7.3–5 and 7.3–6).

The operation of the clock pulse generator circuit is identical in the variable and fixed frequency mode except that, in the variable frequency mode, collector load resistance on Q14 and Q15 (and, therefore, pulse width) is variable. This feature is useful for marginal checking.

7.3–5 PULSE AMPLIFIER The Pulse Amplifier Circuit is shown in Figure 7.3–9.

FIGURE 7.3–9 Clock Pulse Amplifier Circuit



As is shown in Section 7.3–7, the $\overline{CSC}$ signal to the input AND–gate of Figure 7.3–9 is high under normal machine operation. When the C1* signal of the preceding discussion comes high, the output of CR25 is approximately 0 volts. The voltage divider network comprised of R45 and R44 provides a positive base voltage to Q17 that is great enough to cut off Q17. The collector of Q17 starts falling towards −12 volts which cuts off Q19 and turns on Q18. With Q18 on, the collector voltage of Q17 is clamped to −4.5 volts by the clamping diode CR26 and the voltage drop across the base to collector of Q18. The 0.5 volt drop from the

base to the emitter of Q18 places the output voltage of $\overline{C1}$ at –4.0 volts.

When $\overline{CSC}$ is high and $C1^{*}$ is low, then the output of the gate is low and the output of CR25 is approximately –3.5 volts. The voltage divider network of R45 and R44 now provides a negative voltage at the base of Q17 which turns it on. With Q17 conducting, its collector becomes approximately +0.8 volt (due to 0.2 volt drop from emitter to collector). This voltage turns off Q18 and turns on Q19. The voltage drop across the base–to–emitter junction of Q19 results in an output of approximately 0 volts for $\overline{C1}$.

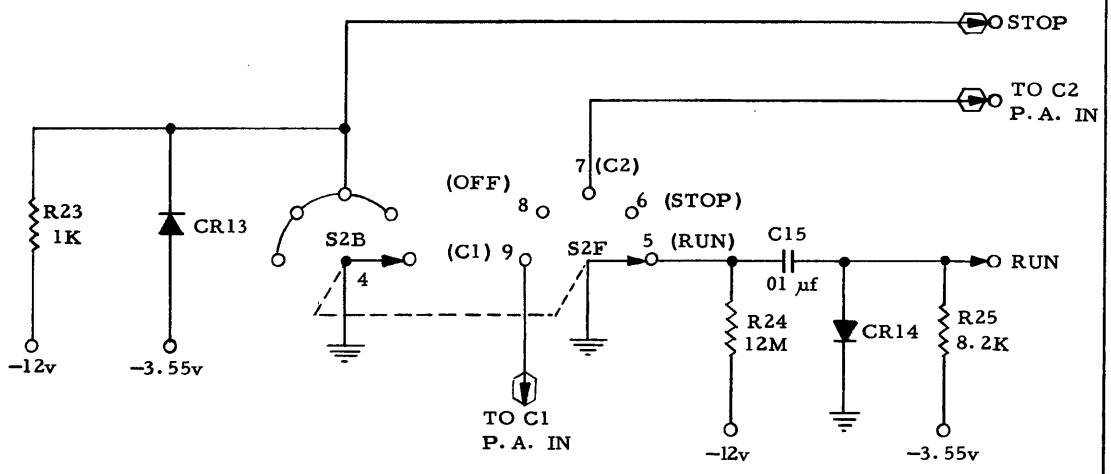
In the above manner the $\overline{C1}$ pulse chain is created. Before being fed to inverters at the various panels to create the C1 pulse chain, the outputs of the clock pulse amplifiers are clamped to 0 volts or –3.5 volts to eliminate some of the ringing on the clock pulses caused by line inductance and capacitance.

The reason for the push–pull output stage is that considerable current must be delivered to the load to make it switch rapidly. This rapid switching requires both <u>push</u> and <u>pull</u> current capabilities. Inductor L3 aids in securing the desired wave shape on the output. Its action is similar to the speed–up capacitor used on inverters.

The clock pulse amplifiers on the C2 line are identical in operation to the one just described.

<u>7.3–6 CLOCK MODE SWITCH</u> Figure 7.3–10 shows the clock mode switch circuit connections.

FIGURE 7.3–10 Clock Mode Switch Schematic



7.3–7 CLOCK START–STOP LOGIC The Central Processor clock can be stopped by four different means:

1) the CLOCK MODE Switch,
2) a ZM signal,
3) the Central Processor Turn–off cycle,
4) logic decoding.

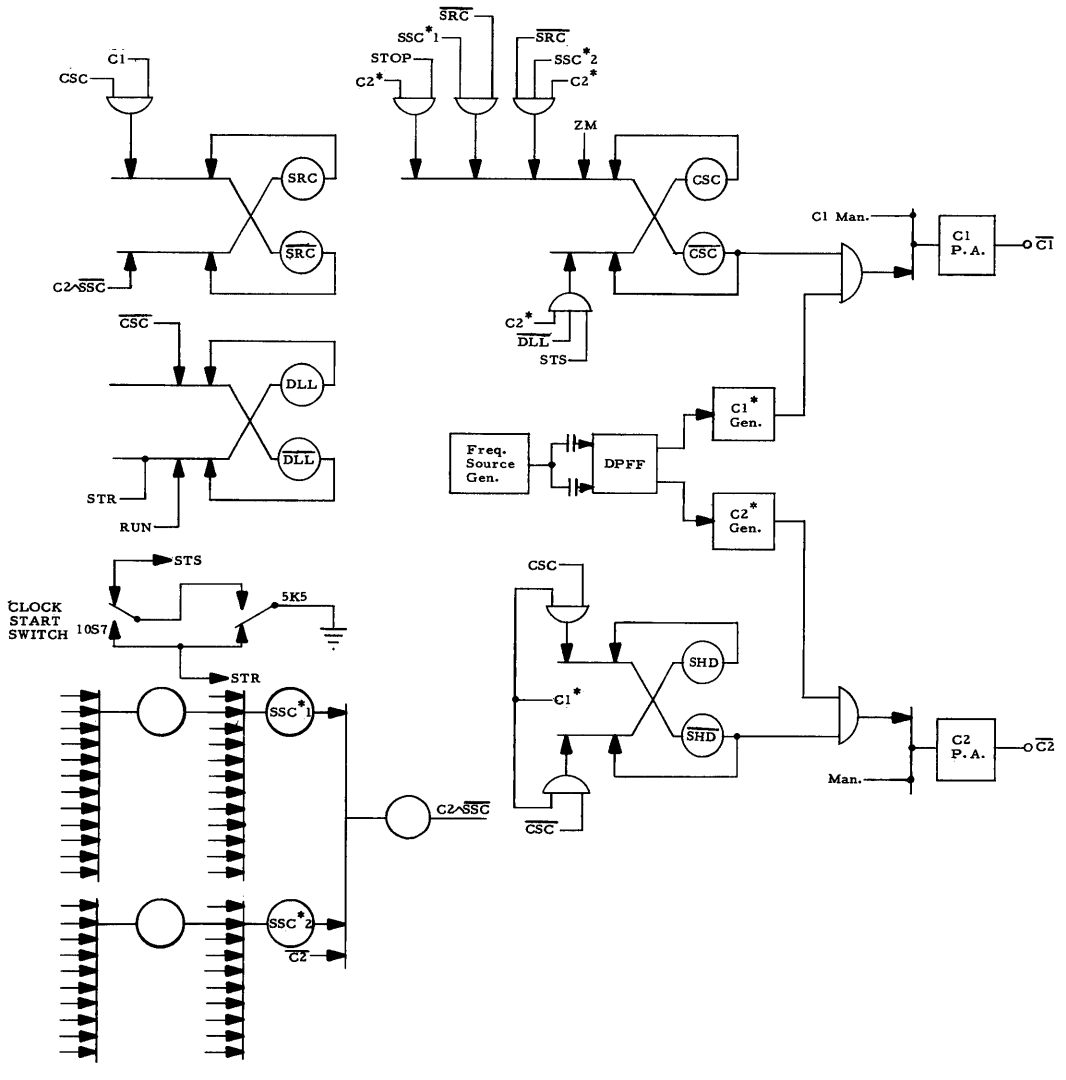
The Start–Stop logic is shown in Figure 7.3–11.

When the CLOCK MODE Switch is turned from the RUN position, the STOP signal comes high. The STOP signal sets the CSC flip–flop which in turn sets the SHD flip–flop. With these two flip–flops set, the C1 and C2 pulse amplifiers cannot be activated by the Clock Generator and, thus, the automatic clock is inhibited.

The Central Processor may be cleared by the ZM (Zero Machine) signal

With the ZM signal high, the CSC flip–flop is set and, thus, the Clock Generator inputs to the C1 and C2 pulse amplifiers are inhibited. The ZM signal can be either generated manually by the INITIAL LOAD Switch or automatically as part of the sequence of events provided by the Central Processor Turn–off cycle.

FIGURE 7.3–11 Clock Start–Stop Logic



A final, extremely flexible, means of inhibiting the automatic clock generation output is through the use of logic decoding. Figure 7.3–11

shows a grouping of inverters with numerous unused OR–gate inputs. These gates and inverters are made available to allow the service engineer to inhibit the automatic clock output at any desired predetermined time during the operation of the Central Processor. The outputs of this gating, the SSC *1 or SSC *2 signals, are used to set the CSC flip–flop which inhibits the automatic clock inputs to the C1 and C2 pulse amplifiers. The first stage of this stop–clock logic is, in effect, two ten–input AND–gates, which are in turn fed into two ten–input OR–gates. The inputs to these AND–gates and OR–gates are provided by clip–lead connections to desired logic signals. The logic functions of this group of inverters are such that the SSC *1 inverters should not be fed by flip–flop signals that are set on C1 clock pulses and are stable during C2 clock times, and SSC *2 should not be fed by flip–flop signals that are set on C2 clock pulses and are stable during C1 clock times. By connecting desired signal outputs to the appropriate diode inputs to SSC *1 and/or SSC *2, the Clock Generator can be stopped during the processing of a program. This allows great latitude on the part of the service engineer to hand clock and observe register contents of desired areas of a program. This is a very powerful debugging aid.

If the CSC flip–flop is to be set by either the SSC *1 or SSC *2 flip–flops, it will be noticed that these signals must be accompanied by the $\overline{\text{SRC}}$ signal. When CSC comes high, it in turn sets the SRC flip–flop which will not be reset until both the SSC *1 and SSC *2 signals are low. If the signals that made either or both the SSC *1 or SSC *2 signals high are not removed prior to restoring the clock, then the SRC signal will remain high and prevent the clock from being turned off repeatedly by the SSC signals.

The CSC and SHD signals are used to inhibit the automatic clock pulses. The circuitry is set up so that the CSC flip–flop will always be set on a C2 clock pulse. Once the CSC flip–flop is set the $\overline{\text{C1}}$ and $\overline{\text{C2}}$ clock pulses

are inhibited at the pulse amplifiers and the automatic clock output is stopped. Thus, the Central Processor's automatic clock will always end on a C2 clock pulse. Conversely, when the automatic clock is restarted, it always begins with a C1. Thus, it can be seen that the automatic clock can be stopped and restarted without disrupting the program being processed.

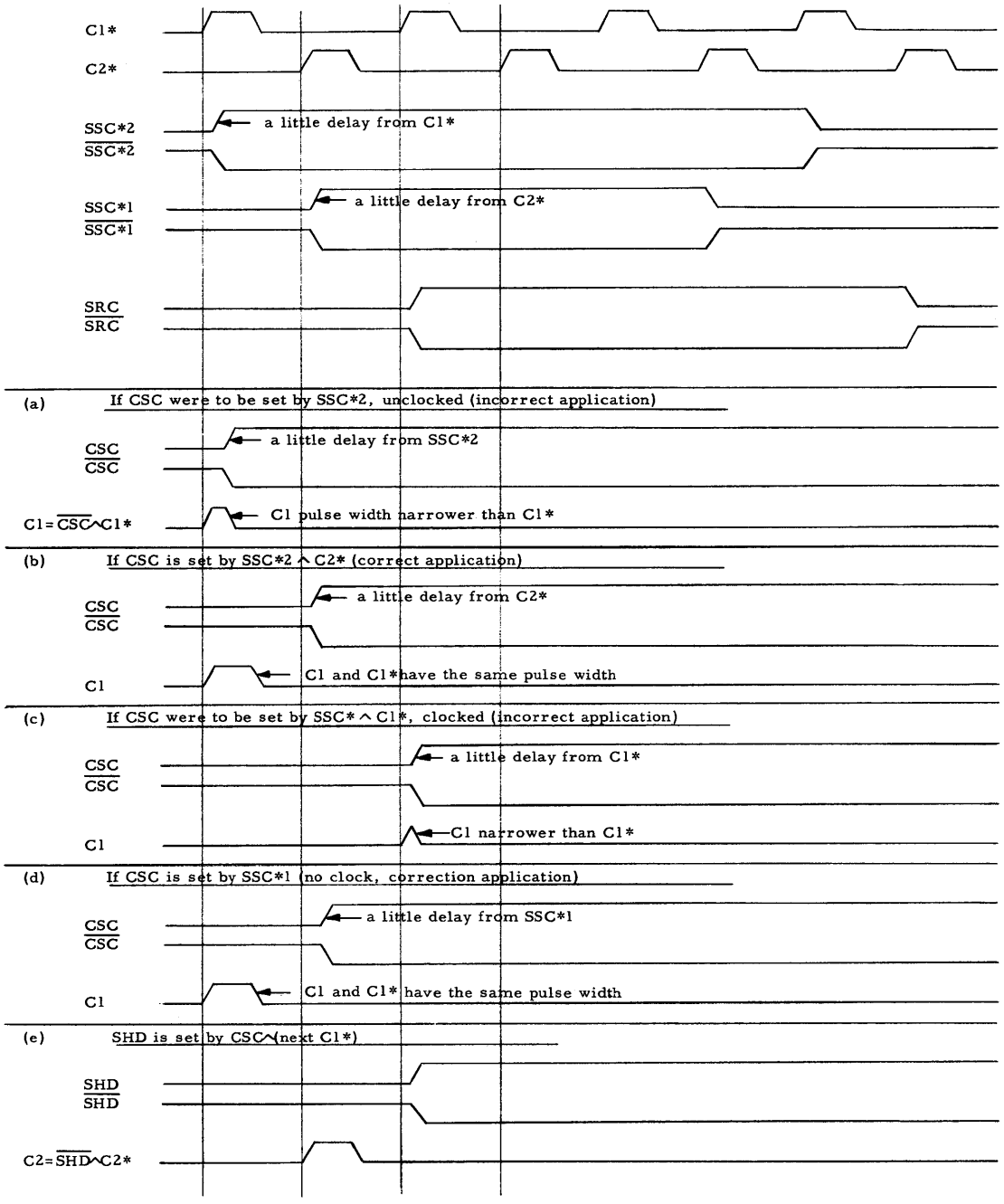
It will be noticed in Figure 7.3–11 that the SSC *1 signal is sent unclocked to the set side of the CSC flip–flop, whereas the SSC *2 signal needs a C2 pulse before it can be gated. The reason for this is that the SSC *1 and the SSC *2 inverters are intended to be used with signals from flip–flops that are <u>set</u> on C2 and C1 clock pulses respectively (see previous discussion of SSC signals). (Remember that a signal set on a C2 pulse is available for reading on the next C1 clock pulse.) If the SSC *1 and SSC *2 signals were not gated into the set side of the CSC flip–flop as shown, it would be possible to decrease the width of the <u>last</u> C1 clock pulse generated before the automatic clock output is stopped. This C1 pulse could be so narrow as to be unreliable or completely insufficient as a gating term. If this should happen, it would mean that one of the sequential chain of C1 and C2 clock pulses was lost and processing of the program beyond that point would be worthless. Figure 7.3–12 shows the <u>final</u> C1 pulse shape for the various possible ways of gating the SSC *1 and SSC *2 signals to the CSC flip–flop.

The Central Processor clock can be started by three methods:

1) the Central Processor Turn–on cycle,
2) the CLOCK MODE Switch,
3) the Central Processor START–CLOCK Switch.

When the Central Processor is initially energized, a Turn–on cycle of events occur. Two actions of the Turn–on cycle operation are to start

FIGURE 7. 3–12 SSC*1 and SSC*2 Gating Influence on Final C1 Pulse Shape



the Clock Generator and to generate a ZM signal. It was shown in the preceding discussion that a ZM signal sets the CSC flip–flop which

inhibits the automatic clock pulses. Thus, the Turn–on cycle must reset CSC to enable the clock. At the beginning of the Turn–on cycle, the 5K5 relay is positioned as shown in Figure 7.3–11. In this position, the STR signal, which resets the DLL flip–flop, is high. At the end of the Turn–on cycle and after the ZM signal is generated, the Turn–on cycle operates the 5K5 relay and removes STR from ground, thus making the STS signal high. At this time the $\overline{\text{DLL}}$ and STS signals are high and, on the next C2 * generated by the Clock Generator, the CSC flip–flop is reset. This enables the automatic clock inputs to C1 and C2 pulse amplifiers.

If the clock pulses were inhibited by the STOP signal from the CLOCK MODE Switch, the clock can be enabled by merely returning the switch to the RUN position. When the RUN signal is high, the $\overline{\text{DLL}}$ signal is high and, since the STS signal is normally high during Central Processor operation, conditions necessary to the resetting of the CSC flip–flop exist. $\overline{\text{CSC}}$ in turn enables the C1 and C2 pulse amplifiers.

If the clock pulses were inhibited by the logic decoding discussed earlier in this section or by a manual ZM signal, then the clock is enabled by depressing the START CLOCK Switch. This causes the STR signal, which resets the DLL flip–flop,to come high. The remaining sequence of events is the same as that discussed above.

7.3–8 SUMMARY OF THE CENTRAL PROCESSOR CLOCK GENERATOR PULSE REQUIREMENTS

1. Two phase
2. Pulse amplitude 0 volts $^{+.5V}_{-.2V}$ to −3.55 volts $^{+.2V}_{-.5V}$
3. Rise time (t_r) 50 nanoseconds (nano = 10^{-9})
4. Fall time (t_f) 50 nanoseconds
5. Pulse width (t_p):

fixed: .35 microsecond ± 50 nanoseconds

variable: .15 microsecond to .4 microsecond

6. Pulse period (t_c):

fixed: 1 microsecond

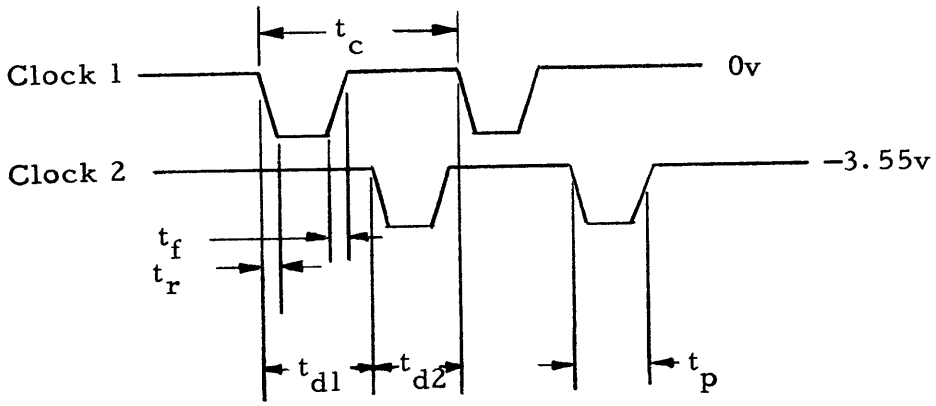
variable: .7 to 2.5 microsecond

7. Pulse delay (t_{d1}; t_{d2}):

fixed: $t_{d1} = t_{d2} = 1/2\ t_c \pm 5$ per cent

variable: depending upon variable pulse period, symmetry variable from CL. 2 occurring immediately following CL. 1 to the opposite extreme.

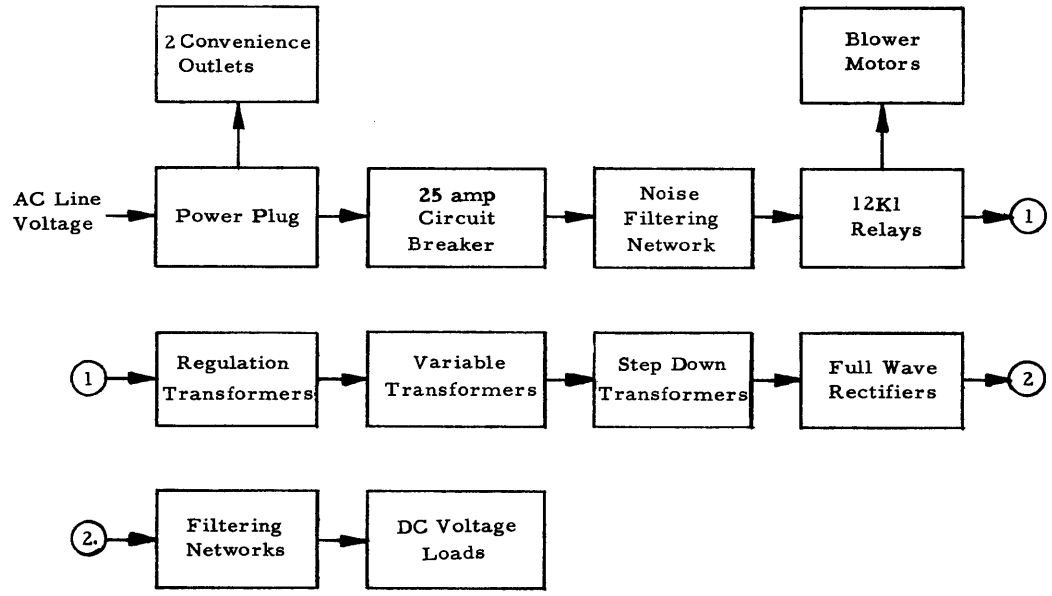
8. Pulse current: At 0 volts capable of 160 milliampere
At −3.55 volts capable of pulling the capacitive load down in 50 nanoseconds.



SECTION 7.4 – CENTRAL PROCESSOR POWER SUPPLY

7.4–1 GENERAL Figure 7.4–1 presents a block diagram of the Central Processor power supply from the a.c. line to the d.c. supplies. Drawing 3E780 presents the same information, but in schematic form. This schematic should be referenced by the reader during the following discussion.

FIGURE 7.4–1 Block Diagram of Central Processor Power Supply



The a.c. line voltage is applied directly from the power plug to two convenience outlets. The outlets are grounded to the Central Processor frame and should be used for connection of test equipment and soldering irons when work on the Central Processor requires their use. Also connected directly to the power plug is the line noise filtering network. This network is protected by a 25 ampere circuit breaker. This

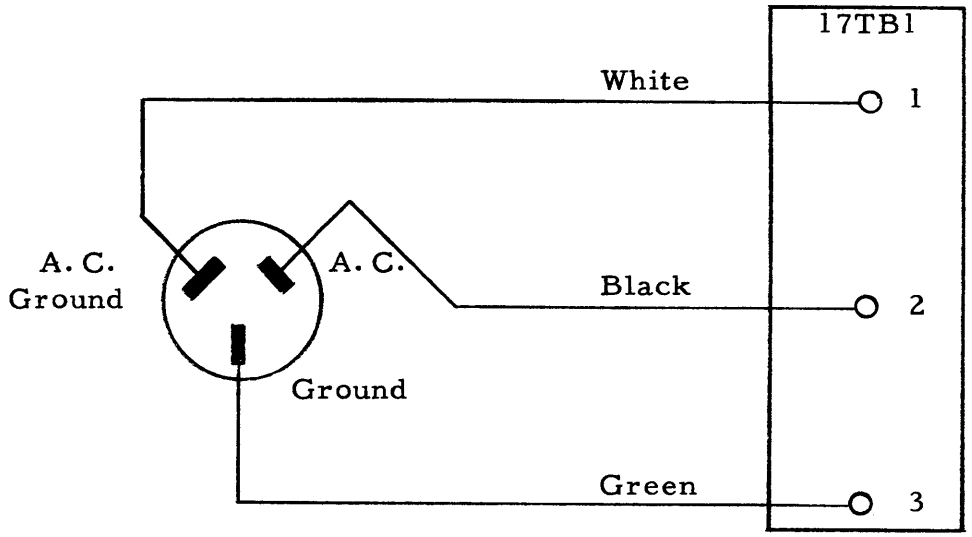
breaker protects the power supply from shorts in the power lines prior to the fusing network. After filtering the a.c. voltage is applied to the Central Processor's Turn–On cycle circuits and also to the open contacts of the 12K1 relay. When the turn–on cycle is initiated, relay 12K1 is energized. (See Section 7.5.) When the contacts of relay 12K1 close, a.c. voltage is distributed to the blower motors of the Central Processor cooling system and to the first stage of the d.c. power supplies.

The line voltage is initially regulated by the three 1KVA regulating transformers, 18VR1, 2, and 3. These transformers distribute 115 volts RMS, ± 1 per cent, to the various step–down transformers of the individual d.c. supplies. In the case of the marginally checked voltages (+12H, +12F, –12A, –12M, –3.55J, –15D, –25U, and –102.5), however, the 115 volt a.c. supply is first applied to individual variable transformers and then to the step–down transformers. The outputs of the power supplies supplied by the step–down transformers are wired to the voltmeter and the control setup that constitutes the marginal voltage checking system on the control panel. Each voltage can be varied to approximately +5 per cent and –25 per cent of its normal value.

All the d.c. voltages are rectified by full wave bridges. All of the rectified voltages, excluding +50 volts, are applied to single–sectioned choke input filtering systems. The +50 voltage is filtered by a resistance capacitor network and regulated by a 50 watt Zener diode. All filter networks are designed to limit the ripple to not greater than 1 per cent peak–to–peak.

The ground reference (0 volts) used through the entire machine is the same ground supplied by terminal 17TB1–3 of the power plug (see Figure 7.4–2). The a.c. ground supplied by terminal 17TB1–1 is isolated from the rest of the machine by the 1 KVA transformer regulators.

FIGURE 7.4–2 Central Processor Power–Plug Polarity



The supplies of the –3.55 special voltage, the –3.55 clamp voltage, and the +4 voltage are actually the regulators of these voltages. A complete explanation appears in the regulator write–ups. A description of every d.c. voltage in the Central Processor is presented in Table 7.4–1.

TABLE 7.4–1 D. C. Voltages of the Central Processor

Voltage	Description
–1.25 and –2.5	Reference voltages for memory; tapped off a voltage divider across the –4.5 volt supply

TABLE 7.4–1 (Continued)

Voltage	Description
–3.55J (regulated sink)	Clamp voltage used throughout entire Central Processor
–3.55 (Special regulated)	Clock Generator and memory (sense amplifier)
+4 (regulated)	Clock Generator
–4.5	Memory: Clamp voltage and source of –1.25 and –2.5 volt supplies
+6.8 (regulated)	Reference voltage and supply for +4 volt regulator
+12F	AND–gates and wherever else needed
+12H	Pin H of inverter packages and wherever else needed
–12A	Pin A of inverter packages and wherever else needed
–12M	Pin M of inverter packages and wherever else needed
–15 (regulated)	Memory: Actually supplied as only –12.5 volts but referenced to –2.5 volts on memory
–15D	Memory: Pin D of X–Y drivers
+24	Turn on relays, Communication Lines and source of +6.8 reference voltage
–25U	Memories: Pin U of the inhibit driver
+50	Plate supply for indicator lights
1 RMS	Heater supplies for indicator lights (supplied by transformer 11T11)
–102.5	Memory: Actually supplied as –100 volts but referenced to –2.5 volts in memory

7.4–2 VOLTAGE REGULATORS There are five regulated voltages in the Central Processor. These voltages consist of:

(1) –3.55 volt Special, used only in the memory and the Clock Generator circuits,

(2) –3.55J clamp voltage,

(3) +4 volts, used only on the Clock Generator,

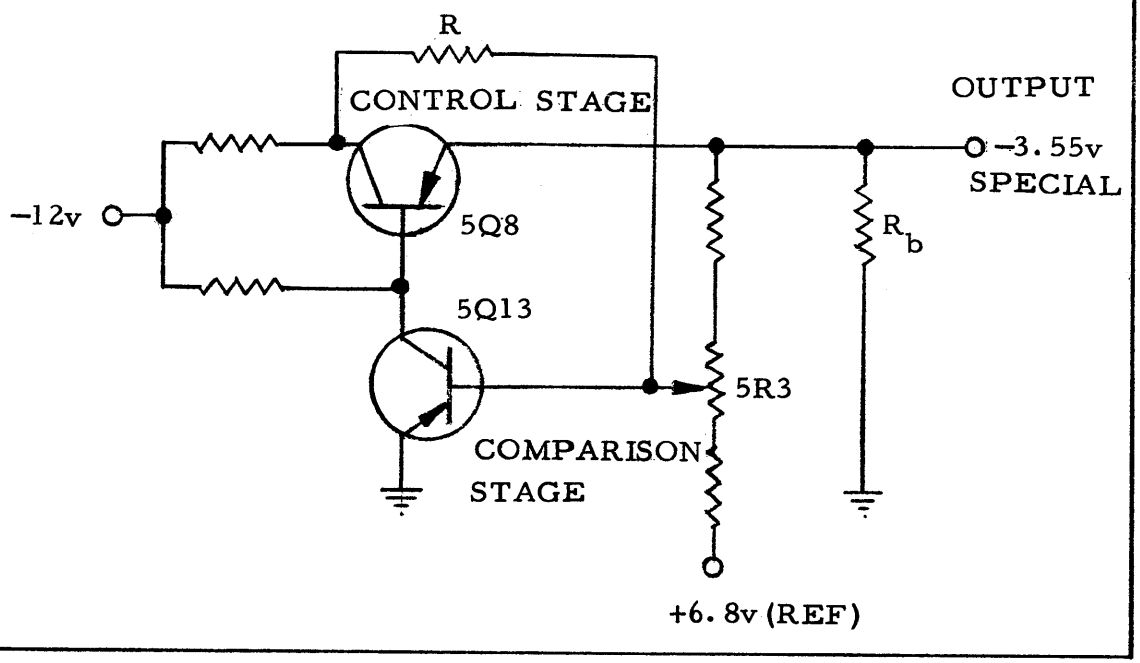
(4) –15 volts, used only in the memory,

(5) +6.8 volts, used only as a reference voltage.

All of these circuits, with the exception of the +6.8 volts supply, are basically the same design. Drawing 3D900 presents the schematic of the complete regulator circuits.

The –3.55 SPECIAL VOLTAGE REGULATOR exhibits the most basic design of the regulators remaining to be discussed (see Figure 7.4–3).

FIGURE 7.4–3 The –3.55 Special Voltage Regulator Circuit



The base of the comparison stage transistor samples a portion of the –3.55 Special supply, compares it to the 0 volts on the emitter and produces a signal which is proportional to the difference. If the –3.55 Special output drops, (becomes more negative) the comparison stage transistor output causes the base voltage on the control stage transistor to rise. The control stage transistor then allows less current to pass and the voltage on the emitter will rise back up toward –3.55 volts. A rise above –3.55 volts causes an opposite reaction. Therefore, a nearly constant voltage is always maintained across the output.

In order to maintain constant output during marginal checking of the –12 volt supply, the compensating resistor, R, is added to the circuit. A network similar to this appears in all regulator circuits which are marginally checked. The resistor, R_b, acts as a bleeder so that a minimum current is allowed to pass through the regulator.

The +6.8 reference voltage can initially vary ± 5 per cent. Potentiometer 5R3 of Figure 7.4–3 can be adjusted to compensate for this variance. Similar potentiometers are used in other regulator circuits for the same purpose.

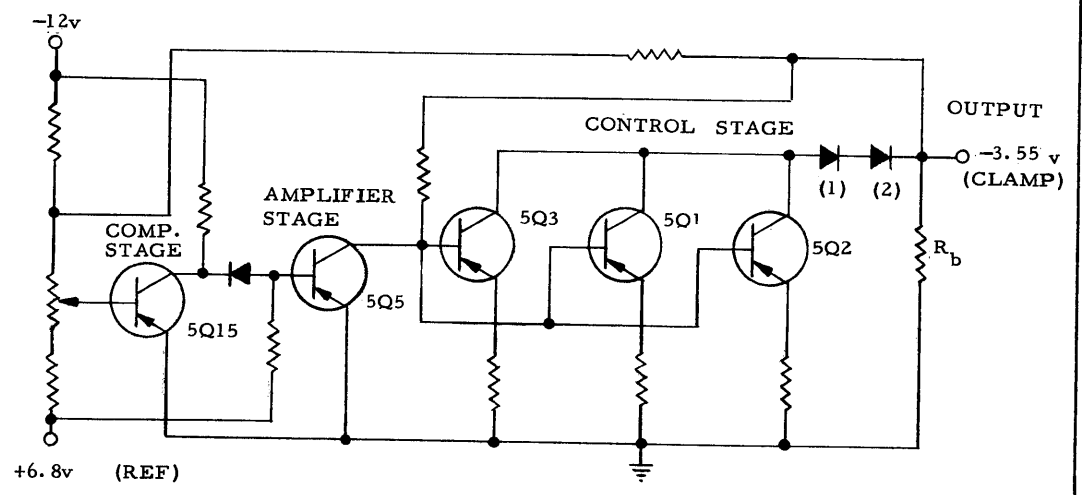
The –3.55J CLAMP VOLTAGE REGULATOR fulfills a twofold purpose. It regulates the –3.55J clamp voltage and also supplies enough current so that this clamp voltage can be applied wherever needed in the Central Processor.

This –3.55 volt regulated supply is developed across the network containing diodes 1 and 2 and the three control transistors. A portion of this voltage is sampled by the comparison stage and any variations will augment the same type of compensation as mentioned for the –3.55 Special voltage regulator. However, since a large amount of current (35 amps) must be available for all clamp diodes in the machine, a

large control stage, consisting of three power transistors in parallel, was necessary. Each of these transistors supplies 1/3 of the required output current. An amplifier stage was added to supply sufficient base current to the transistors of the control stage.

The bleeder resistor, R_b, serves two purposes. First, it relieves some of the high current burden from the regulator transistors since a portion of the load current flows through it and, secondly, it will maintain a negative clamp voltage (although somewhat higher than –3.55 volts) in the event that the voltage regulator completely fails.

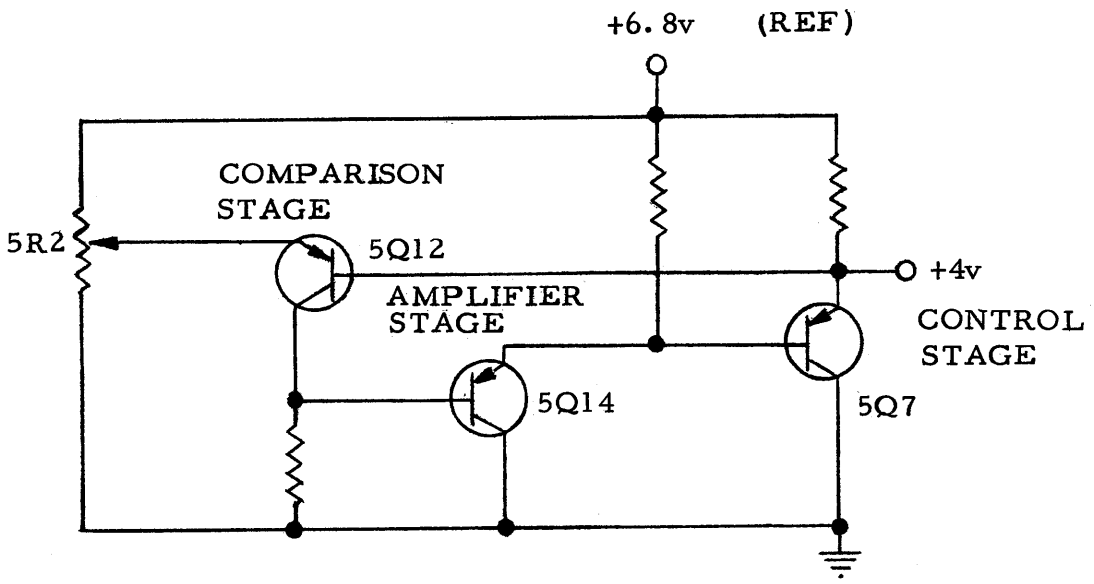
FIGURE 7.4–4 The –3.55J Clamp Voltage Regulator Circuit



The +4 VOLTS REGULATOR is slightly different from the others since the regulated output voltage is positive. Also, the comparison stage transistor samples the output voltage directly and not a portion of this voltage as in other regulators. For example: if the +4 voltage rises, V_{ce} of the comparison stage transistor increases and causes the voltage

at the base of the amplifier stage transistor to decrease. The emitter voltage of the amplifier stage follows the base voltage towards 0 volts and thus the base voltage of the control stage transistor also drops toward 0 volts, causing that transistor to conduct harder. When this happens, V_{ce} of the control stage transistor drops and thus the output voltage drops back towards +4 volts.

FIGURE 7.4–5 The +4 Volts Regulator Circuit

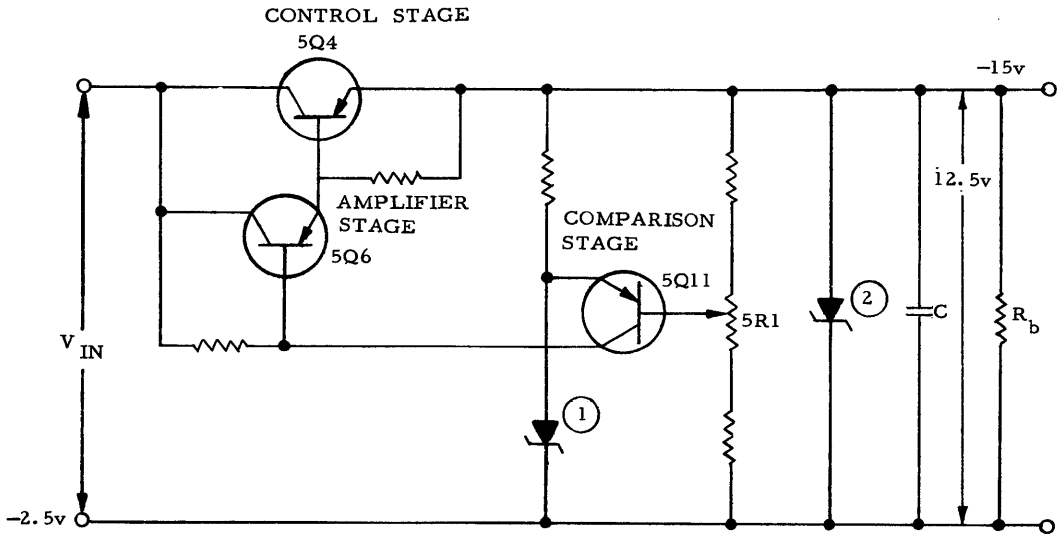


The –12.5 VOLTAGE REGULATOR regulates a voltage (–15 volts) already available from one of the supplies. Actually, only –12.5 volts is regulated since the supply is referenced to –2.5 volts. The same principles are applied in this regulator as have been applied in the others, but there are a few added modifications, as shown in Figure 7.4–6.

Zener diode 1 provides a constant reference voltage of –6.8 volts ± 5 per cent, which is supplied to the emitter of the comparison stage

transistor. Thus, any variation in the –12.5 volts is sampled and compared to a constant voltage, causing the amplifier and control elements to compensate accurately for the variation.

FIGURE 7.4–6 The –12.5 Volts Regulator Circuit



Zener diode 2 was added as a precautionary measure. It is possible for a certain type of memory failure to subject this regulator circuit to a drastic increase in voltage. This increase could be high enough to cause the regulator circuit to become inoperative. The addition of the Zener diode across the output makes it impossible for the voltage to rise over –18 volts since the Zener will begin to draw current at this level.

The bleeder resistor, R_b, provides a minimum current flow through the regulator. The capacitor, C, helps the regulator during fast changes in load by supplying current until the regulator can compensate for the load change.

The +6.8 REGULATED VOLTAGE, which is tapped off the +24 volts supply, is regulated by means of a Zener diode. The Zener tolerances are such that the voltage will not vary more than ± 5 per cent.

SECTION 7.5 – CENTRAL PROCESSOR TURN–ON/TURN–OFF CIRCUITS

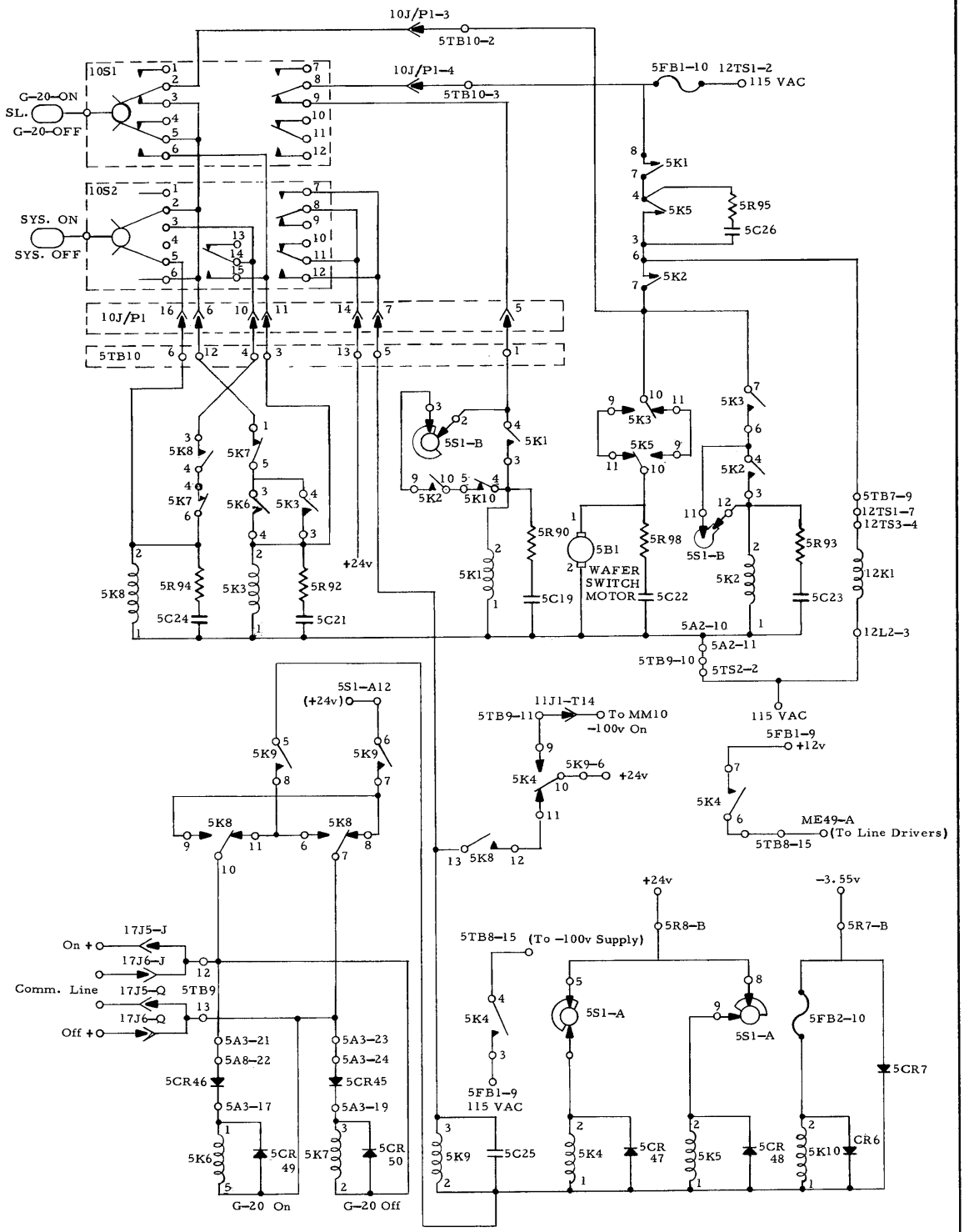
The Central Processor can be turned on and off by three sources: manually by the G–20 Switch or the SYSTEM Switch, or logically through the Communication Line. However, the G–20 Switch has unconditional control of a.c. voltage to the Central Processor. If this switch is in the OFF position, neither a signal from the Communication Line nor the SYSTEM Switch can turn on the Central Processor.

The Turn–on/Turn–off cycles are timed by a motor driven wafer switch which revolves at 4 RPM. This switch turns through two–thirds of one revolution (10 seconds) in completing the Turn–on cycle. During this interval, the d.c. voltages, initial load (ZM), memories, line drivers, and Clock Generator are turned on in the order given. While the computer is running, the wafer switch is stationary, but once an OFF indication is made, the switch rotates through the final one–third revolution (5 seconds) and completes the Turn–off cycle. Whenever the Central Processor is turned OFF, PSC is set. Following this, the voltages on the line drivers and the memories are turned off and, finally, the Clock Generator and all the d.c. voltages are cut off. Figure 7.5–1 shows the complete schematic of the Turn–on/Turn–off cycle circuits.

7.5–1 CENTRAL PROCESSOR TURN–ON CYCLE In the following discussion of the Turn–on cycle, the sequence of events is explained in chronological order. They can be followed visually by referring to the schematic on Drawing 3E903 or Figure 7.5–1 which is a simplified version of 3E903.

When the G–20 Switch is placed in the ON position, relay 5K3 is energized by 115 volts a.c. Once the contacts of 5K3 close, the wafer

FIGURE 7.5–1 Turn–on/Turn–off Cycle Schematic

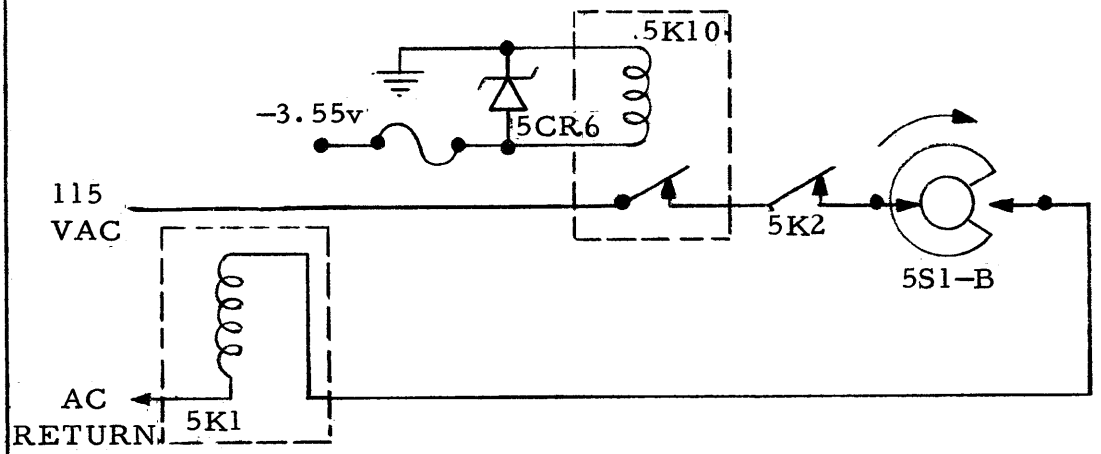


switch motor is energized and the wafer switch starts rotating, the initial load switch is readied (the ZM signal is high), and immediately following, relay 5K2 is energized. The contacts then allow 12K1 to be energized and as soon as the contacts of 12K1 close, all the d.c. supplies receive 115 volts a.c. (see Drawing 3E780). Once the d.c. voltages are developed, the rest of the Turn–on cycle can be initiated.

The logic circuits in the Central Processor must be protected against any appreciable decrease in the –3.55 voltage. The necessary protection is provided by the network shown in Figure 7.5–2. As soon as the –3.55 voltage is developed, the contacts 5K10 (–3.55 volt relay) will open and prevent the wafer switch from energizing the coil of 5K1 (115 volt drop–out relay). Once 5K1 is energized, it locks itself in and the only way to de–energize it is to depress the G–20 OFF Switch. If, however, the voltage decreases to a level of about –4.7 volts, the contacts of 5K10 will not open, since the Zener diode 5CR6 will begin drawing enough current to blow fuse 5FB2–10. As a result, coil 5K1 will be energized by the rotating switch and the 115 volts a.c. supply will drop out.

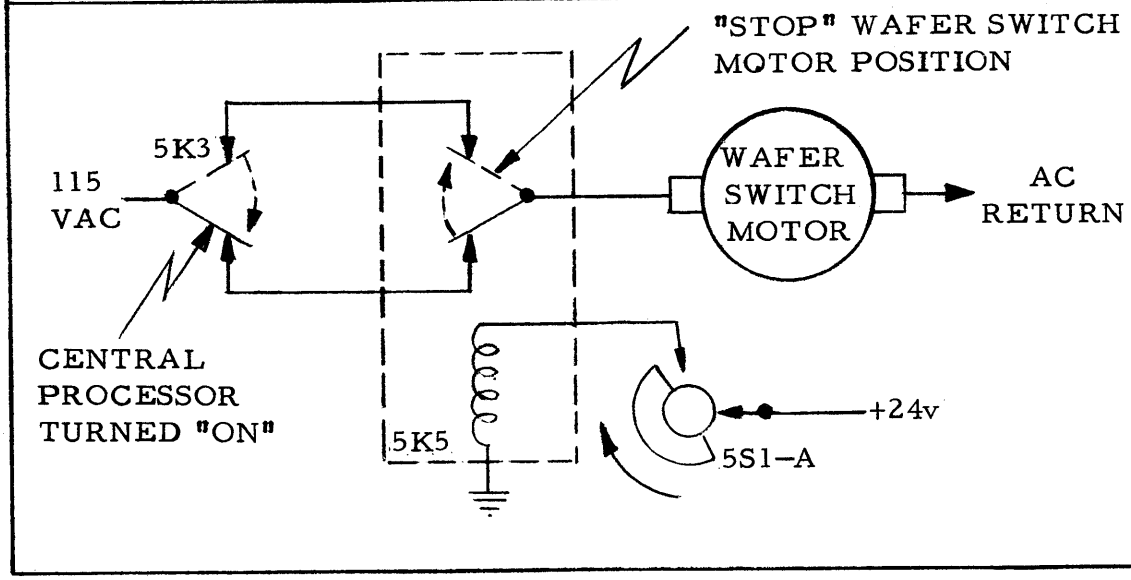
The Turn–on cycle will continue as soon as the wafer switch allows +24 volts to be available to relays 5K4 and 5K5 (see Figure 7.5–4). The coil of 5K4 is energized first. The closing of its contacts supplies 115 volts a.c. to the –100 volt memory supply, +12 volts to the line drivers, and +24 volts to the Auxiliary Memory Module, MM–10. 5K5 is energized shortly after 5K4. When its contacts close, the Clock Generator, which had been set up as soon as the d.c. voltages were developed, is turned on by making STS high, which in turn resets CSC. The wafer switch will then stop since the wafer switch motor is in series with a set of normally closed contacts of the 5K5 relay which are opened when 5K5 is energized. This is shown in Figure 7.5–3. Both the 5K5 and 5K4 relays will stay energized after the wafer switch motor has stopped.

FIGURE 7.5–2 Condition Just Before Contacts 5K10 Should Open



This completes the Turn–on cycle caused by depressing the G–20 ON Switch. The wafer switch will not rotate again until the contacts of 5K3 and 5K5 of Figure 7.5–3 are in the position indicated by the dotted lines. This will occur when an OFF signal is given to the Central Processor.

FIGURE 7.5–3 Condition Just Before Wafer Switch Motor Stops



A second method of turning on the Central Processor is by means of the SYSTEM Switch. Note that this switch can energize the Central Processor only if the G–20 Switch is in the SLAVE position. The Turn–on procedure is identical to that given using the G–20 Switch, except that relays 5K3, 5K8 and 5K9 are energized at the same time (see Figure 7.5–1). Once the contacts of 5K3 are closed, the wafer switch starts rotating and the d.c. voltages are developed as before. With the advent of the d.c. voltages, the rest of the Turn–on cycle is initiated as discussed earlier. When the contacts of 5K9 and 5K8 are closed, +24 volts are momentarily supplied to the ON and OFF sides, respectively, of the Communication Line. This turns the power on in all units on the Communication Line whose power switches are in the SLAVE position.

The third method of turning on the Central Processor is by means of the Communication Line. As in the case of the SYSTEM Switch, the G–20 Switch must also be in the SLAVE position if this method is to be used. The procedure followed for this method of Turn–on is this: when an external device applies a +24 volts signal to the ON line and 0 volts to the OFF line of the Communication Line, relay 5K6, which is connected directly to the ON line, is energized. When the contacts of 5K6 close, 115 volts a.c. is supplied to 5K3. Once the contacts of 5K3 are closed, the wafer switch motor is energized and the wafer switch starts rotating. This results in the development of the d.c. voltages and the initiation of the remainder of the Turn–on cycle.

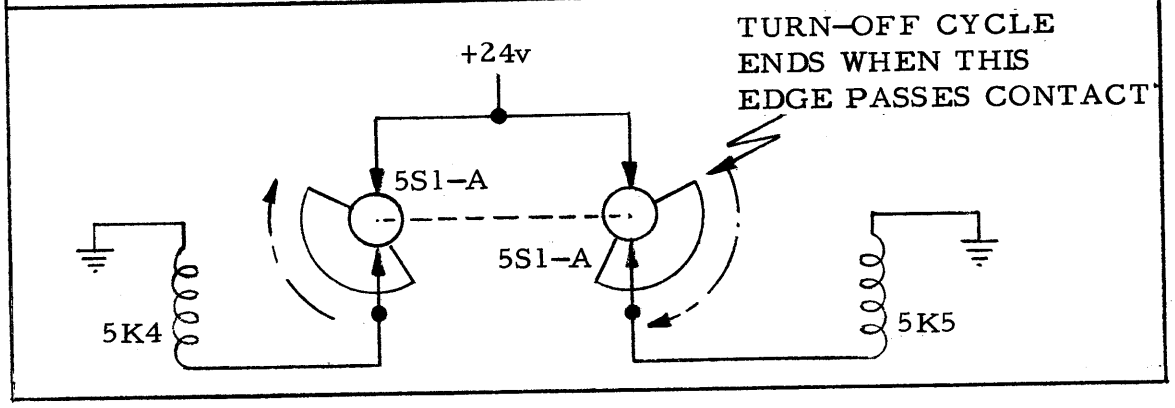
7.5–2 CENTRAL PROCESSOR TURN–OFF CYCLE When an OFF signal is generated by depressing the G–20 Switch (or the SYSTEM Switch if the G–20 Switch is in the SLAVE position), the contacts of relay 5K3 immediately drop out. This action supplies 115 volts a.c. to the wafer switch motor and the Turn–off cycle is initiated. If the G–20 Switch is in the SLAVE position and the Central Processor is to be

turned off via the Communication Line, then the +24 volt signal is applied to the OFF line and 0 volts is applied to the ON line of the Communication Line by some peripheral unit. This causes relay 5K7, which is connected directly to the Communication Line in the Central Processor, to be energized. When relay 5K7 is energized, its contacts open and the 115 volt a.c. supply is removed from relay 5K3. This then initiates the Turn–off cycle as discussed previously, and also sets flip–flop PSC which inhibits memory starts.

When the wafer switch motor is energized, the contact 5S1–A which previously energized relays 5K4 and 5K5 begins to rotate as shown in Figure 7.5–4. When the edge of wafer switch 5S1–A passes the contact point of relay 5K4, that relay is de–energized. This action cuts off power to the memories and the line drivers. The Central Processor is not completely turned off, however, since relay 5K5 is still energized. The contacts of relay 5K5, in series with the coil 12K1, maintain the d.c. voltages even though relay 5K2 drops out after relay 5K3 is de–energized (see Figure 7.5–1).

When the edge of 5S1–A passes the contact point of relay 5K5, then the d.c. voltages and the Clock Generator are turned off and the Turn–on cycle is complete.

FIGURE 7.5–4 Control By Relays 5K4 and 5K5



SECTION 7. 6 – SPECIAL CIRCUITS

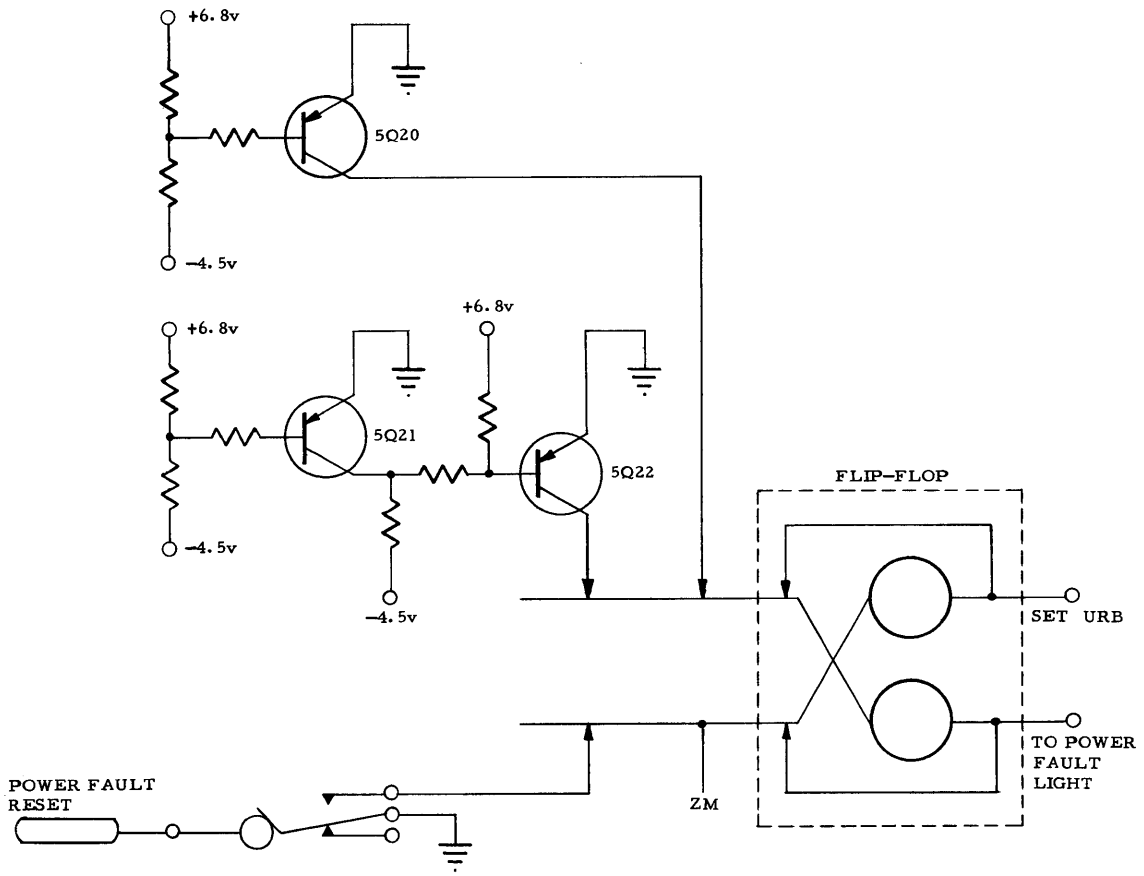
7. 6–1 POWER FAULT DETECTOR Sudden changes in the line voltage to the Central Processor which are greater than ± 8 per cent of the 115 volt a. c. supply are detected by the POWER FAULT detector. When a line fault is detected by this circuit, the Power Fault Indicator light on the control panel is lit and the audio alarm is sounded. These power faults will normally be too short to allow the power line relays to drop out, but long enough to cause damaging fluctuation in the d. c. power supplies. Detection of a. c. line fluctuations is accomplished by two transistors sampling a d. c. voltage created directly from the output of one of the voltage regulator transformers. One of the transistors detects increases in the 115 volt line supply and the other transistor detects decreases. A special d. c. voltage (–4. 5 volts) is employed in the sampling circuit since all the other d. c. voltages in the Central Processor are either regulated or can be marginally checked. This –4. 5 volt d. c. supply is adjustable by potentiometer 5R5, however, to accommodate testing and to compensate for drift caused by varying circuit parameters.

The simplified version of the detection circuitry is shown in Figure 7. 6–1. Transistors 5Q20 and 5Q21 are biased by voltage dividers between the –4. 5 volt and the regulated +6. 8 volt supplies with 5Q21 normally turned on and 5Q20 normally cut off. The outputs of these two transistors feed the Set side of the flip–flop shown in Figure 7. 6–1. (The output of 5Q21 is fed through an inverter stage first, though, so that the inputs to the flip–flop are low in the normal operating state.)

When 124 volts or greater are detected, transistor 5Q20 goes high and sets the flip–flop. When 106 volts or less are detected, transistor 5Q21 cuts off, but this signal is then inverted and thus sets the flip–flop.

With the flip–flop set, 0 volts is at the URB (Ring Bell) flip–flop which turns on the alarm tone, and –3.5 volts is sent to the POWER FAULT indicator light bulb driver which turns it on. These two power fault signals can be removed by the POWER FAULT RESET and the SIGNAL TONE Switches on the control panel or by the ZM signal generated by the ZM Switch or the Turn–on cycle.

FIGURE 7.6–1 Power Fault Detection Circuitry



7.6–2 HIGH AND LOW TEMPERATURE THERMOSTATS Since the magnetic core memory planes are designed to operate between 64°F and 90°F, high and low temperature protective thermostats are provided. In the Central Processor and in each MM–10 unit, one low

temperature thermostat is provided since a low temperature warning could occur because of ambient conditions within the unit. A high temperature thermostat is provided for each memory module in both the Central Processor and the individual MM–10 units, however, because of the possibility of overheating in an individual module.

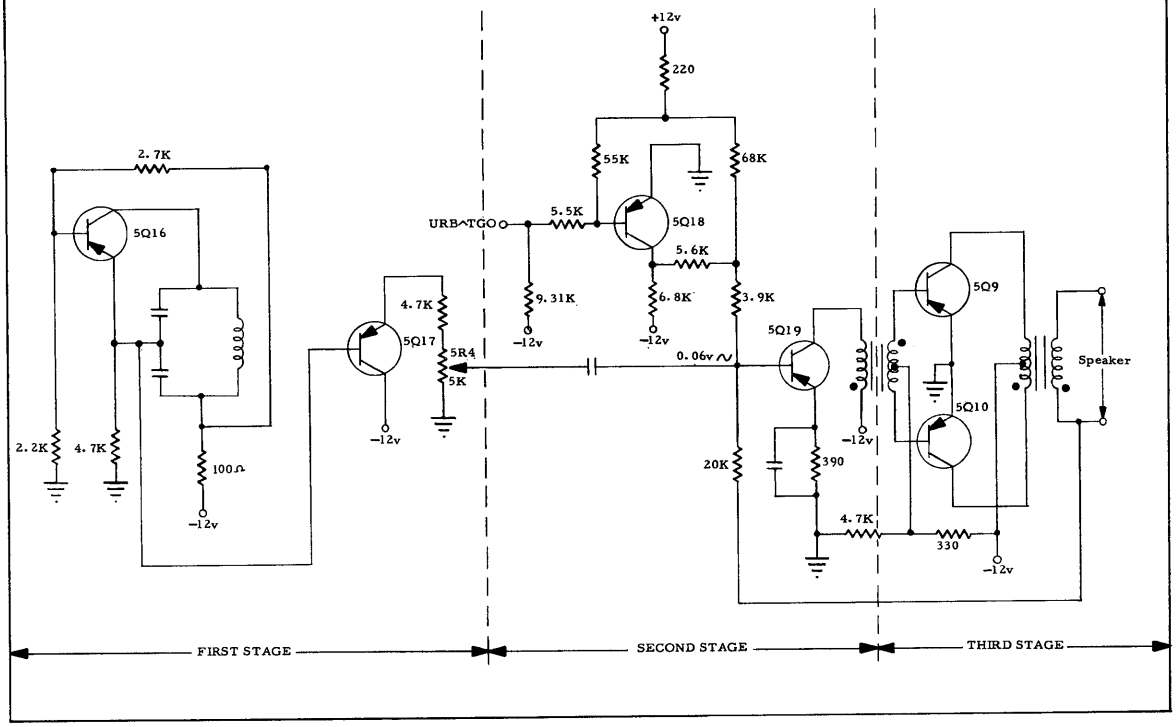
The alarm tone and the LOW TEMP indicator light (mounted on the control panel) are energized when the internal cabinet temperature is $65^{\circ}F \pm 1^{\circ}$. The HIGH TEMP indicator light and the alarm tone are likewise energized when the cabinet temperature rises to $90^{\circ}F \pm 1^{\circ}$. The temperature rise in the cabinets is expected to be between $6^{\circ}F$ and $10^{\circ}F$. Therefore, the Central Processor may initiate a high temperature warning even though the room temperature is still around $80^{\circ}F$ to $84^{\circ}F$. The Central Processor and MM–10 units can be expected to operate immediately after Turn–on in an ambient temperature of $66^{\circ}F$. However, if the ambient temperature is $63^{\circ}F$, then the Central Processor and MM–10 units should be allowed 20 minutes warmup time.

When high or low temperature conditions exist, the alarm tone can be turned off by the SIGNAL TONE OFF Switch, but the HIGH TEMP or LOW TEMP indicator lights will remain on until the condition is corrected. The SIGNAL TONE RESET must be actuated to reset the tone generator after the temperature is restored to an acceptable level.

7.6–3 THE TONE GENERATOR The tone generator delivers an audio alarm signal whenever it is turned on by a high output (0 volts) from the URB flip–flop. Aside from being set and reset by program, URB can be set by high or low temperature indications from the thermostats, memory parity errors, or power line faults, and reset manually from the control panel by the SIGNAL TONE RESET position or by the INITIAL LOAD Switch. The URB flip–flop is also reset during the Central Processor Turn–on cycle. Basically, the tone generator is a

three–stage device putting out a 1400 cps signal (for complete circuit see Drawing 3D883).

FIGURE 7.6–2 Tone Generator, Simplified



The first stage is a Colpitts type oscillator which has a sine wave output of about 3 volts peak–to–peak. The wave shape and amplitude of this output is completely independent of the rest of the circuit due to the high impedance emitter follower which it feeds, and also the decoupling network on its –12 volt supply. Therefore, adjustment of any parameters in the rest of the circuit, including volume control, will not affect its output. The audio signal volume control, potentiometer 5R4, is connected to the output in the emitter follower of this first stage.

The second stage consists of a driver amplifier and the on–off switch circuitry. The amplifier operates Class A, and to insure against

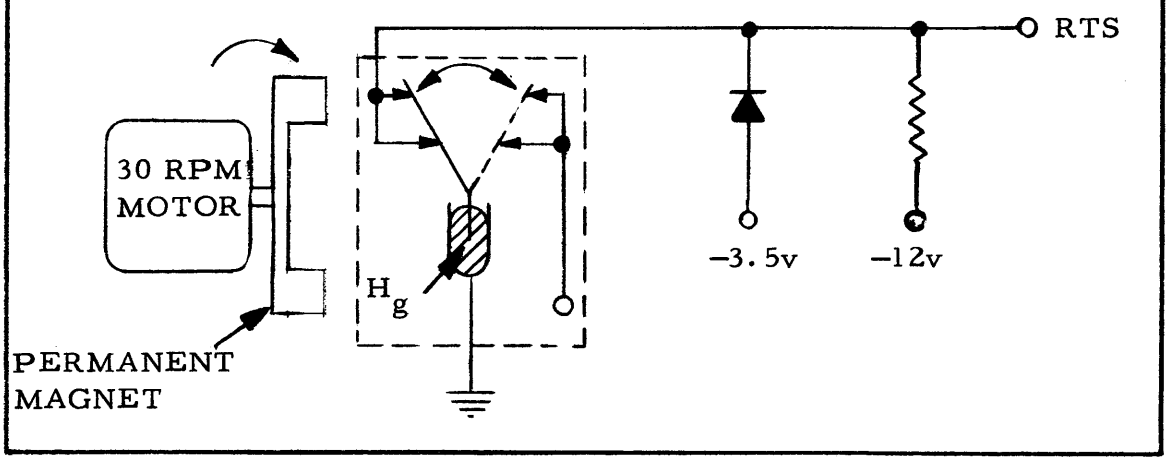
distortion, it has negative feedback to its base from the output of the last stage. The on–off switch is merely an inverter controlled by the URB signal. When the output of URB is high, the base voltage on the driver amplifier is –2.5 volts over–ridden by a signal of 0.06 volt peak–to–peak from the oscillator via the 5R4 potentiometer. When URB is low, this base voltage is about 0.5 volt (gnd ref.). The signal TGO is normally low, but when the SIGNAL TONE Switch is in the OFF position, the TGO signal goes high (0 volts) and the signal tone cannot be sounded.

This final stage is a push–pull amplifier which has a maximum undistorted output of 10 volts peak–to–peak. It delivers 3.5 watts to a 3.2 ohm speaker and is capable of driving several more speakers. The polarity of the output transformer and the input transformer of this stage should be carefully checked so that the feedback will be effective.

7.6–4 REAL TIME REFERENCE GENERATOR This circuit supplies the Central Processor with a reference to real time by generating signal RTS which sets the RTA flip–flop once every second. RTA is used along with other logic to provide the timing pulse, RTR, once every second. The real time pulse generator is a simple device consisting of a 30 RPM motor, a permanent magnet and a mercury–wetted switch as shown in Figure 7.6–3.

Once every second, the magnet lines up with the switch and pulls the switch armature to the RTS contact thus applying 0 volts to it. At all other times, RTS is clamped to –3.5 volts. Due to the action of the mercury on the contacts, switch bounce is eliminated, thus preventing multiple (and unwanted) timing pulses.

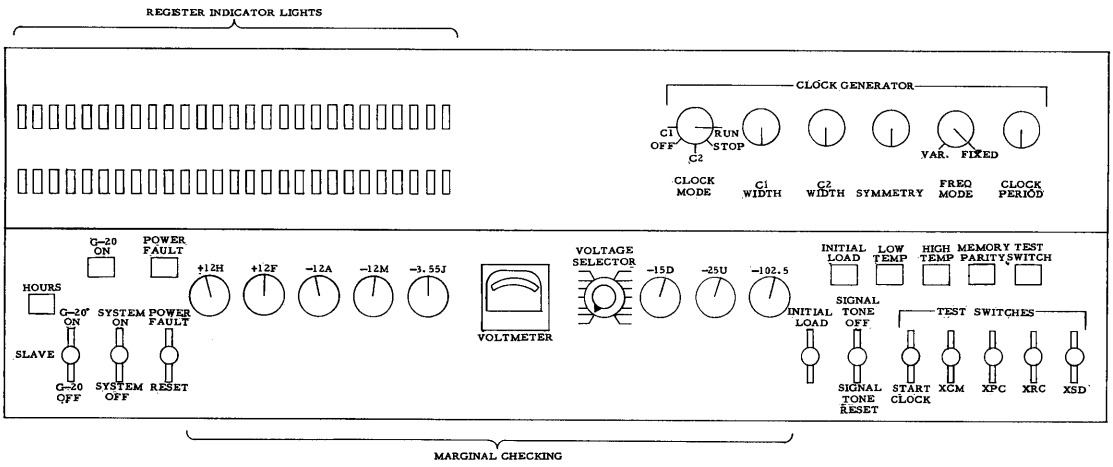
FIGURE 7.6–3 Real Time Pulse Generator



SECTION 7. 7 – THE CONTROL PANEL

Figure 7. 7–1 shows the layout of the Central Processor control panel.

FIGURE 7. 7–1 Central Processor Control Panel



7. 7–1 CONTROL PANEL SWITCHES It was shown in Section 7. 5 that the Central Processor can be turned on by three sources: manually by the G–20 Switch and the SYSTEM Switch, and logically through the Communication Line. The G–20 Switch, however, over–rides the other sources by having complete control of a. c. voltages to the Central Processor. The G–20 Switch locks in all three of its possible positions. In the ON position, the Turn–on cycle is started and the Central Processor is turned on completely independent of any other unit in the system. As long as the switch is in the ON position, it cannot be turned off by any other unit. In the SLAVE position, the G–20 Switch allows the Central Processor to be turned on by certain peripheral units via the Communication Line, or by the use of the SYSTEM Switch in the Central Processor control panel. When the G–20 Switch is in the OFF position, the Central Processor is unconditionally turned off.

The SYSTEM Switch is a momentary contact type switch whose function is to provide a means of turning on the complete system, including the Central Processor, from a central point. All units on the Communication Line that are in the SLAVE position will be affected by this switch. When the switch is placed in the ON position, a +24 volt pulse is applied to the ON line of the Communication Line and 0 volts is applied to the OFF line. In the OFF position, the SYSTEM Switch causes a pulse of reverse polarity on these two lines of the Communication Line. Section 7.5 indicates how the SYSTEM Switch controls the Central Processor.

The INITIAL LOAD Switch, a momentary contact switch, zeros the machine, and sets the Central Processor to an operating condition where it can correctly accept and store commands of the Bootstrap operation. This switch is effectively closed during the Central Processor Turn–on cycle and can be manually operated at any time.

The POWER FAULT RESET Switch is provided for the operator as a means of resetting the flip–flop that is set (see Section 7.6) whenever a power fault condition occurs. When this flip–flop is reset, the POWER FAULT indicator light is turned off, but to turn off the signal tone generator, it is necessary to depress the SIGNAL TONE Switch to the RESET position.

The SIGNAL TONE Switch is used to control the signal tone generator. It was previously shown that if the SIGNAL TONE Switch is not in the OFF position, the signal tone will occur whenever flip–flop URB is set. The positioning of SIGNAL TONE to RESET is one means of resetting the URB flip–flop and thus turning the signal tone off. When the SIGNAL TONE Switch is in the OFF position, the signal tone <u>cannot</u> be sounded and the TEST SWITCH indicator light will be lit.

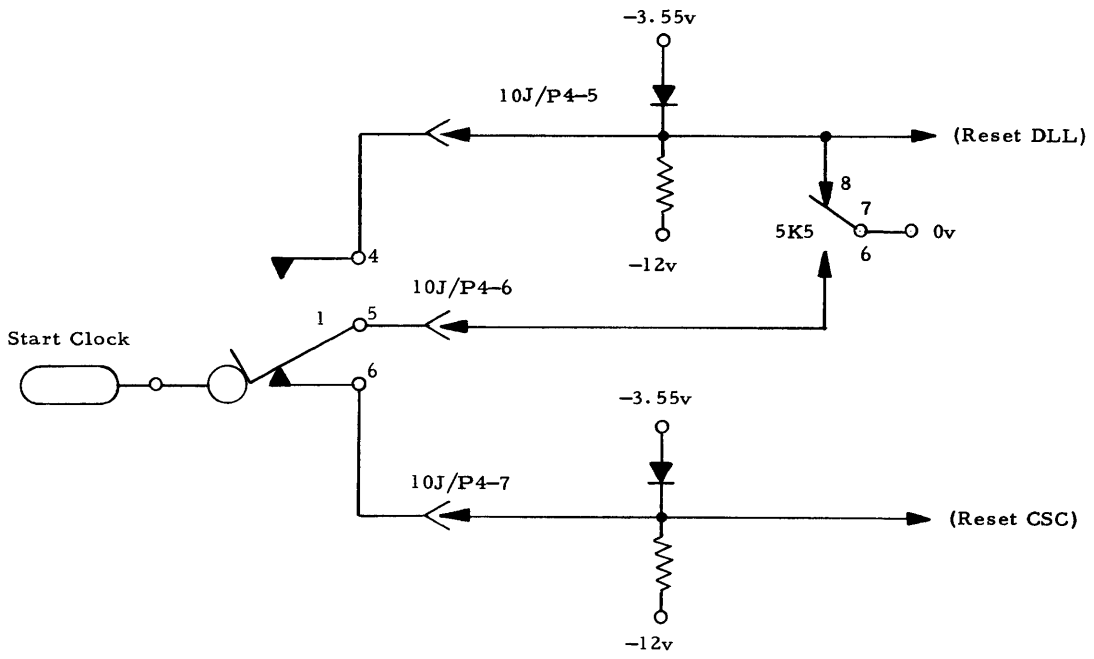
There are five switches on the control panel that are designated as TEST SWITCHES. All of these switches but one, the XSD Switch, are primarily used for debugging purposes. The XSD Switch, a permanent contact toggle, is used to provide the operator with a means of blocking out memory locations 100_8 to 1777_8 to Putaway commands. This area of memory is normally used to store the monitor for the program. When the XSD Switch is depressed, it protects the information in this area of the memory from being accidentally destroyed by a Putaway operation.

The START CLOCK Switch hookup is shown in Figure 7.7–2. This momentary contact switch allows the operator to restart the clock pulses when clock has previously been inhibited by zeroing the machine (ZM signal high) or by the stop clock circuitry discussed in Section 7.3. From Figure 7.7–2, it is seen that upon depression of the START CLOCK Switch the DLL flip–flop is reset and upon release of the switch, flip–flop CSC is reset. The use of these flip–flops in starting the clock is discussed in Section 7.3.

When the XPC Switch is closed, the memory parity checking circuit is inhibited. Therefore, memory accesses can be made without fear of failure due to the detection of parity errors since the parity check feature has been removed. This is especially useful for hardware debugging purposes.

When XCM, a permanent contact switch, is depressed, core memory accesses are inhibited. This is particularly useful when it is desirous to zero the machine during program processing, but still insure that memory information is not affected. To do this, it is first necessary to depress the XCM Switch and then the INITIAL LOAD Switch. The XCM Switch must also be depressed before the XRC Switch will be effective.

FIGURE 7. 7–2 Start Clock Switch Hookup



The XRC Switch serves no useful purpose at the present.

7. 7–2 CONTROL PANEL INDICATOR LIGHTS There are two distinct groups of indicator lights on the control panel. One group indicates the operating conditions of the Central Processor as a unit, while the second group of indicator lights allows the operator to check the internal operations of the Central Processor. The seven individual indicator lights of the control panel are included in the first group mentioned, while the second group is composed of the two banks of indicator lights referred to as the Register Indicator Lights.

The G–20 ON indicator light remains lit as long as the G–20 is turned on. The POWER FAULT light will go on when a power fault condition occurs, but it can be turned off at will by the POWER FAULT RESET

Switch. The INITIAL LOAD light is turned on when the ZM signal is generated by the INITIAL LOAD Switch or by the Turn–on cycle of the Central Processor. This light remains on until the Bootstrap operation, necessitated by the ZM signal, is loaded, and execution of it, starting at memory location 65_{10}, is begun. (At this time the Bootstrap, BS, signal goes low.) The HIGH TEMP or LOW TEMP indicator lights go on whenever the cabinet temperature exceeds $90^{\circ}F \pm 1^{\circ}$ or drops below $65^{\circ}F \pm 1^{\circ}$. These lights remain on until the improper temperature condition within the Central Processor or MM–10 unit is corrected. The MEMORY PARITY indicator light is turned on whenever a memory parity is detected. The Central Processor is unconditionally halted when a memory parity error occurs, and it is necessary to clear the machine and start the program over. The Central Processor will be cleared and the MEMORY PARITY indicator light turned off by depressing the INITIAL LOAD Switch. The TEST SWITCH light is turned on when the XCM or XPC Switches are on, or whenever the SIGNAL TONE Switch is in the OFF position.

The Register Indicator Lights consist of two rows of lights with 61 lights per row. These indicator lights are provided to assist the service engineer in trouble shooting the Central Processor and as such will normally provide useful information to the operator only during hand clocking through a program. The two rightmost lights of this bank of indicator lights have a fixed purpose, and that is to indicate C1 and C2 clock pulses. The rest of these indicator lights, however, may be used pretty well at the discretion of the operator to indicate the output of most of the flip–flops in the Central Processor, plus the condition of other selected points, such as certain register enables. The inputs to all of the indicator lights, with the exception of the C1 and C2 clock indicators, are connected to a group of eight plugs which can be plugged into jacks situated on the tops of the various logic and memory panels. The inputs to the panel jacks are fixed; however, the indicator

light plugs can be connected to these jacks in almost any combination to exhibit the desired information on the Register Indicator Light bank. On the Central Processor, the reader will note that the indicator lights have various numbers and letters associated with them. The indicator lights are first broken up into groups designated by the numbers 10, 20, 30, 40, 11, 21, 31, and 41. These divisions correspond to the eight plugs connected to the light bank and the same numbers are stamped on the corresponding plugs. The series of letters under the first row of lights serves both rows of lights and corresponds to the pin letter of the particular plug. The series of numbers associated with the light banks have no specific meaning, but rather are provided as an aid to the service engineer and will be of increasing value as familiarity with the Central Processor increases. Signals associated with various jacks mounted at the top of the panels are listed in Appendix II of Part V of the manual.

7.7–3 MARGINAL CHECKING CONTROLS Marginal checking adjustment controls are provided for all of the logic voltages and for several key memory voltages to allow the service engineer to detect components that are slowly drifting out of tolerance before they cause errors or machine downtime. Each voltage can be varied to a maximum or approximately +5 per cent and –25 per cent of its normal value.

The d.c. voltmeter and the VOLTAGE SELECTOR Switch on the control panel make it possible to check the voltage on any of the d.c. power supplies in the computer, except the +4 volt, +24 volt, and +50 volt supplies.

7.7–4 CLOCK GENERATOR CONTROLS The Clock Generator controls are discussed in detail in Section 7.3–1, CLOCK GENERATOR CIRCUITS.

7.7–5 ELAPSED TIME INDICATOR The Elapsed Time Indicator is provided to record Central Processor operating time. This indicator will register a maximum of 9,999.9 hours. When the maximum reading occurs, the Elapsed Time Indicator unit must be completely replaced since the drive motor has a useful life of about 12,000 hours.

CHAPTER 8

MAINTENANCE

SECTION 8.1 – INTRODUCTION

8.1–1 PURPOSE It is fully realized that field service personnel have had formal training on the operation and repair of the G–20 Computing System. However, it is felt that experience and techniques gained by personnel with extensive practice in the maintenance and repair of the Central Processor could be of great help to the service engineer. The material presented in this chapter is not intended as a set of hard and fast rules that must be rigidly adhered to. Rather, it is presented as a guide and a reference in the hope that it will make for better and easier repairs of the equipment. These may not be the best approaches to the problem; however, information included in this chapter represents the way things are being done at the present. It is strongly urged that all suggestions and experiences related to the material in this section be sent to the Customer Engineering Office at the main plant for referral to those responsible for this manual.

8.1–2 FIELD SERVICE TOOL LIST Table 8.1–1 gives a list of tools provided in the G–20 Field Maintenance Kit.

TABLE 8.1–1 G–20 Basic Field Service Tool Kit List

Qty.	Description	Manufacturer	Part Number
1	Adapter, 7 Pin Min.	Vector	T7M
1	Allen Driver	Hunter	5E–5/32
1 1	Allen Set, Straight Shank Allen Set, Case Hardened 1/64 to 1/4	Hunter	.050" to 1/4", K5M
1	Blanket	Sears, Roebuck	96A–8700
1	Blueprint Easel	BCD Maint. Shop	
1	Brush 2" Wide		
2 cans	Cleaner, Magnetic Tape	Tecsolv	928
1	Cleaner, Plastic Type	A. W. Faber	7540
1 gal.	Cleaner, Wype	Easterday Supply	
1	Contact Cleaning Tool	Neuses	No. 3–316
1	Diode Tester	BCD	80T2
1	Extension Cord	Beldon	17159–S
1	Feeler Gauge	Starrett	172A
1	Flashlight	Eveready	2251
1	Fuse Puller	General Cement	5525
1	Grab–All Tool	Proto	2342
8 oz.	Grease, Commercial No. 6	Shell Oil	Retnix "T"
1	Grease Gun	Lincoln Engr.	5948
1	Hammer Ball Pien	True Temper 8 oz.	No. 1308
1	Hand Cleaner	GO–JO	
1	Haz–Bin Cabinet	Pressteel Co.	Model J–128–SD
1	Hydraulic Coupler, Midget	Lincoln Engr.	5852
1	Inspection Mirror	Ullman	Model A–2

TABLE 8.1–1 (Continued)

Qty.	Description	Manufacturer	Part Number
1	Lamp Extractor	Micro Switch	15PA19–3/59
1/2 pt.	Magna See	Soundcraft	PRR
24	Mini–gator Clips	Mueller	30–C
12	Mini–gator Insulators (Black)	Mueller	32 BLACK
12	Mini–gator Insulators (red)	Mueller	32 RED
1–2 oz. tube	Molycote, Type G		
8 oz.	Oil, Low Temperature	Texaco Oil	MIL–L–7870A
1	Oscilloscope	Tektronix	Model 535
1	Oscilloscope Cart	Mobil–Tronics	Model 1CB–2
1	Oscilloscope Current Probe	Tektronix	Type 6016
1	Oscilloscope Current Probe Amp.	Tektronix	131
1	Oscilloscope Preamp, Type CA	Tektronix	Type–CA
1	Oscilloscope Preamp, Type D	Tektronix	Type – D
1	Paint Spray Can G–20 Grey		
1	Pliers, Chain Nose	No. 6" Williams	76
1	Pliers, Diagonal	No. 4" Klein	209–5C
1	Pliers, Diagonal	No. 6" Klein	317–5C
1	Pliers, Gas	No. 6" Williams	6
1	Pliers, Needle	No. 6" Williams	116
1	Scope Hood	Tektronix	No. 016–001
1	Screw Extractor 0.050	Greenfield	
1	Screw Driver, Large	Xcelite	146

TABLE 8.1–1 (Continued)

Qty.	Description	Manufacturer	Part Number
1	Screw Extractor 1/16 to 1/25	Greenfield	1815
1	Screw Driver, Offset, 5" Long	Proto	No. 36 5/16
1	Screw Driver, Phillips	No. 1 Williams	TP–1
1	Screw Driver, Phillips	No. 2 Xcelite	X–102
1	Screw Driver, Small	Xcelite	R–186
1	Screw Driver, Stubby	Proto	9651
1	Screw Driver	Stanley	1010
1	Screw Driver, 6"	Stanley	415
1	Screw Starter	Quick Wedge	1738
1	Scribe	Brown and Sharpe	778 Style 1
1	Scriber, Right–Angle Point	Brown and Sharpe	778 Style 3
1	Seizer, Curved	Xcelite	43H
1–1 roll	Solder, Type 63/37, .32 Dia.	Kester	Type 44
1	Soldering Aid	CBS Hytron	SH20–A
1	Soldering Iron, Oryx	Oryx	Model No. 60A
1	Soldering Iron, 60 Watt	Hexacon	P–30
1	Soldering Iron, 300 Watt	Hexacon	P–300
2	Soldering Iron Tip 60 Watt	BCD	44A50
1	Soldering Iron Tip 300 Watt	BCD	1C3029–1
1	Soldering Iron Tip 300 Watt	BCD	1C3079–2
1	Spray Bottle With Applicator		
1	Spring Puller and Pusher	IBM	9002136
1	Spring Scale 1–20 lb.	Chatillan	719–20

TABLE 8.1–1 (Continued)

Qty.	Description	Manufacturer	Part Number
1	Spring Scale 0–32 oz.	Chatillan	0325
1 roll	Tape, Elec., Blk., Plastic, 1/2"	Scotch	22
1	Transformer, Soldering Iron	Triad	10023
1	Tube Puller	General Cement	5092
1	Vacuum Cleaner	Hoover "Pixie"	2830
1	Valve–spout Oiler	West Coast Platen	ASB–175–6"
1	Wheel Puller	Proto	4060A
1 roll	Wire, Test Lead Yellow	Alpha Wire	453C
1	Wire Stripper	Miller	101–S
1	Wire Unwrapping Tool	Engr. Data	A25461–L
1	Wire Wrap Gun, Hand Operated, With Bit	Keller	14H
1	Wrench, Adjustable	No. 8 Williams	8B
1	Wrench, Adjustable, 4"	Crescent	AT14
1	Wrench Kit, Allen	General Cement	666
1	Wrench Kit, Bristol	General Cement	5070
1	Wrench Kit	Williams	1142PR
1	Multimeter	630 Triplett	630

SECTION 8.2 – INSTALLATION PROCEDURE

8.2–1 SHIPPING All G–20 Central Processors sent to installations within the U.S. will be sent by special moving van unless specifically ordered to be delivered by other means. These vans are supplied with a special suspension system that insures a softer ride and are very popular for moving large electronic assemblies. Another feature of the van service is that once the equipment is loaded, it remains in the same van with the same driver until the end of the line. This insures a minimum of handling and an original source of any problems that occur during the trip. Also, shipping by van simplifies the packing that must be done to insure safe delivery.

If a Central Processor is ordered delivered overseas or in the U.S. by some means other than "Electronic" van, special packing will be provided to fit the situation. Special instructions for unpacking will be supplied with each special shipment.

At the main plant the Central Processor is covered with shipping blankets, moved into the van by Rol–A–Lifts, and placed on cardboard supports. These cardboard supports prevent the computer from riding on its own, more rigid mounts which would transmit shocks more readily than the cardboard. These cardboard supports, because of their positioning, also provide more complete support for the Central Processor's frame.

When the Central Processor arrives at its destination, certain precautions in handling should be observed. First, check to see that a reinforced loading ramp capable of handling in excess of a one ton load has been supplied. If this should not be the case, then special support to the ramp must be provided. It is possible that such a situation might exist since the Central Processor is loaded into the van at the plant

using the plant's ramp which is designed for this purpose, but off–loading is done on a ramp supplied by the van line. After the loading ramp has been checked and approved, then off–loading of the system can begin. Due to positioning of the units in the van, it may not be possible to off–load the Central Processor first; however, if other units are to be abutted to it, the Central Processor must be moved to its designated location first. This is necessary since the design of the Central Processor's cabinet is such that it can be moved without causing possible damage to the frame only when it is supported at both ends of the frame by Rol–A–Lifts or similar moving apparatus. For this reason, it is never acceptable to move or position the Central Processor with support provided only in the middle of the frame. When moving the Central Processor, it is necessary to pay close attention to the cabinet doors since transport over uneven floors will cause vertical movement of the doors. This movement, if extreme, can cause warping of the frame and bending of the doors. When moving the Central Processor over particularly uneven surfaces, it is recommended that the doors be removed to prevent possible damage to them.

8.2–2 INSPECTION Because of the unusual value of the G–20 System, a thorough inspection routine after delivery is necessary to protect the company from loss due to damage in transit. An outline of a suggested inspection procedure is presented to facilitate a thorough inspection at a new installation:

1) Do not sign the delivery carrier's delivery receipt before a thorough examination for apparent damage has been made.

2) Always obtain a copy of any paper you sign.

3) Remove the impact graph from the van and return it to the Customer Engineering Office at the main plant.

4) Check the floor in the immediate area of the equipment to

determine if any screws or nuts had worked out during shipment.

5) Check with the driver to see if anything unusual occurred en route.

6) After the equipment is in its final position at the computer site, remove the packing material and make a very thorough inspection to ascertain any damage that might have occurred during shipment.

7) If irregularities are noticed, try to ascertain their source and cause.

8) Report and discuss all irregularities that are noted.

If there is evidence of damage to the equipment which could have happened in transit, the following procedure should be followed:

1) If damage is found, endorse the delivery receipt and note the damage. For example: "Received with following damage noted. Left fan housing dented. Typewriter case cracked", etc., as the case may be.

2) Request the carrier's driver to sign his name against your notation.

3) Preserve the packing material of the damaged equipment.

4) Proceed with a detailed examination of the damaged equipment, determining the total extent of damage as closely as possible.

5) Request a claim inspection from the Claims Department of the line haul carrier who handled the equipment. This request for inspection should be made by telephone and followed up by a confirming letter with a copy sent to the Customer Engineering Office at Bendix Computer. This inspection must be requested not later than 24 hours after arrival of the shipment.

6) If the shipment is in such condition that it must be returned to the factory, request the inspector to have it returned as "dead head" freight. He will make arrangements with his office.

7) Issue a complete report on the entire proceedings to the Traffic Department at Bendix Computer, transmitting to them any papers left by the inspector together with your copy of the delivery carrier's delivery receipt.

The foregoing procedure applies only in the case of visible damage. Where the damage is of a concealed nature, the only thing that can be done is to call the Claims Department of the line haul carrier involved, explain the situation and ask for a "concealed damage inspection". This inspection must be requested within 15 days from receipt of the shipment.

8.2–3 INSTALLATION REQUIREMENTS There are certain minimum installation requirements for a G–20 Computer System and these are discussed quite fully in the Bendix Computer Manual, G–20 Site Selection and Preparation. In addition to this manual, Bendix Computer makes available an experienced Site Preparation Team that visits the proposed computer site to help with system layout and advise on site preparation prior to installation. Thus, the Installation Group should have little trouble along these lines at the time of installation. The following checks, however, should be made by the Installation Group before, or during, the installation:

1) Check with the electrician who installed the power cable to insure that the wire is capable of handling the current load of the Central Processor.

2) Check all connecting cables to make sure that proper plugs have been attached.

3) Check out power plugs for proper voltage and polarization (see Figure 7.4–2).

4) Check the air conditioning and humidity regulation system for proper operation.

5) Check to see that the position of the Central Processor has been laid out on the floor with masking tape to facilitate positioning it during installation.

6) Check to see that a minimum aisle of three feet has been provided both in front and back of the Central Processor's location.

7) Depending upon component placement in the system layout, insure that adequate space has been allowed for opening the front panel doors to their desired position.

8.2–4 OPERATION CHECKOUT This section describes the test procedure for the Central Processor that must be followed when the power is turned on for the first time at a new installation. The Central Processor's power supplies and voltage regulators must be adjusted for proper output, and warning devices must be checked for proper operation.

Several preliminary operations are necessary prior to performing the various test procedures. First, using a Multimeter check that all fuses on fuse blocks 5FB1 through 5FB5 are good. Then, set the Signal Tone Switch on the control panel to the SIGNAL TONE OFF position. This will eliminate the annoying signal tone in case the power fault detector is initially out of adjustment.

Power Supply Adjustment Procedure:

1) With the 115–volt power supply plug connected to the 115–volt

line and the line circuit breaker in the ON position, turn on the Central Processor using the SYSTEM ON or the G–20 ON Switch.

2) Place the VOLTAGE SELECTOR Switch mounted on the control panel to the positions shown in Table 8.2–1, and adjust the control knob or designated potentiometer until the meter on the control panel indicates 100 per cent.

TABLE 8.2–1 Internal Voltage Adjustments

Position of Voltage Selector Switch	Adjustment Knob or Potentiometer
+12H	+12H
+12F	+12F
–12A	–12A
–12M	–12M
–15D	–15D
–25U	–25U
–3.55	5R3
–15	5R1

3) With voltage supply –12M set at 100 per cent, set voltage supply –3.55J to 110 per cent as indicated by the meter on the control panel. With the meter still reading the –3.55J supply, vary the –12M supply control knob to both right and left of its original position. If the –3.55J supply varies by more than ± 1 per cent of the 110 per cent value, switch the meter to the –12M supply and read the value at which this occurs. If –12M reads between 90 per cent and 110 per cent then the voltage regulator of the –3.55J supply is not operating correctly and

must be repaired.

4) After completing the test outlined above, set the –3.55J supply to 100 per cent as indicated by the control panel meter.

5) Turn the VOLTAGE SELECTOR Switch on the control panel to the –4.5T position. The meter should read between ± 5 per cent of 100 per cent. If it does not, the following procedure should be followed:

NOTE: If the Central Processor is to be used with only one memory panel, the primary connections to transformer 4T8 should be made on terminals 4T8–1 and 4T8–3. If the Central Processor is to be used with two memory panels, the primary connections to transformer 4T8 should be made on terminals 4T8–1 and 4T8–2. See Power Supply Schematic 3E780.

a) Disconnect the wires between 4R1 and 4TS1–1, and between 4R1 and 4T8–1.

b) Determine what resistance is needed between 4TS1–1 and 4T8–1 so that the voltage on jack pin FC51B is –4.5 volts with respect to pin FC51D.

c) Determine which two terminal connections on 4R1 give a value of resistance closest to the value determined in "b" above. Connect one terminal to 4TS1–1 and the other to 4T8–1. The control panel meter should now indicate 100 per cent ± 5 per cent.

6) Adjust potentiometer 5R2 until the voltage on jack pin 11J1–3 on the clock distributor is +4 volts (return on 11J1–6).

7) Check the voltage at the points shown in the table below with respect to the return point indicated:

TABLE 8. 2–2 –12. 5 Volt Supply Test Points

VOLTAGE	MEASUREMENT POINT	RETURN POINT
–12. 5V	F–TS1–1 N–TS1–1	F–TS1–5 N–TS1–5
–12. 5V	F–TS1–2 N–TS1–2	F–TS1–6 N–TS1–6
–12. 5V	F–TS1–3 N–TS1–3	F–TS1–7 N–TS1–7
–12. 5V	F–TS1–4 N–TS1–4	F–TS1–8 N–TS1–8

8) Set the VOLTAGE SELECTOR Switch on the control panel to the –102. 5 position and adjust the –102. 5 control knob until the per cent meter indicates 100 per cent.

9) Set the VOLTAGE SELECTOR knob on the control panel first to the –1. 25 KL position and then to the –2. 5T position. The per cent meter should read 100 per cent ± 10 per cent.

Warning Devices–Adjustment Procedures

1) Place the clock generator CLOCK MODE Switch in the RUN position.

2) Move the SIGNAL TONE Switch to its normal, central position. If the tone generator turns on, adjust potentiometer 5R5 and operate the POWER FAULT RESET and SIGNAL TONE RESET Switches until the tone generator is turned off.

3) Rotate 5R5 clockwise to the point where the tone generator just turns on and make a note of this point. The POWER

FAULT light should now be on.

4) Turn the tone generator off by turning 5R5 a little counter–clockwise and actuating the POWER FAULT RESET and SIGNAL TONE RESET Switches.

5) Continue rotating 5R5 counterclockwise until the tone generator is just turned on again and note this point also.

6) Find the midpoint of the two settings determined in steps 3 and 5 above, and set 5R5 at this point.

7) Actuate the POWER FAULT RESET Switch. The POWER FAULT light should turn off. Actuate the SIGNAL TONE RESET Switch and the signal tone should turn off.

8) Place a heat source near thermostat 9ST1 so that its temperature can increase above 90°F. After a short period of time the tone generator should turn on and the high temperature light should light.

   a) While the high temperature light is on, move the SIGNAL TONE Switch to the SIGNAL TONE RESET position for an instant. The signal tone should go off while the switch is in the reset position and should go back on when the switch is released.

   b) While the high temperature light is on, move the SIGNAL TONE Switch to the SIGNAL TONE OFF position for a few seconds. The signal tone should remain off while the switch is in the off position and should go back on when the switch is moved back to the normal position.

9) Remove the heat source from near 9ST1. The high temperature light should turn off after a short delay and the signal tone should remain on.

10) Move the SIGNAL TONE Switch to the SIGNAL TONE RESET

position to stop the signal tone.

11) If the INITIAL LOAD light is not on, actuate the INITIAL LOAD Switch. The INITIAL LOAD light should now go on.

Miscellaneous Adjustments

1) Check once more to see that the adjustments made in steps No.'s 2, 4 and 9 of the Power Supply Adjustment Procedure have not drifted out of adjustment due to heating up of components. Make readjustments where necessary.

2) Loosen all voltage control knobs located on the control panel, reposition the knobs so that all the white pointers are vertical, and retighten the knobs.

3) Tighten the lock nuts on potentiometers 5R1, 5R2, 5R3, 5R4 and 5R5, being careful not to move the potentiometers out of adjustment.

SECTION 8.3 – ACCESS PROBLEMS AND HIGH TEMPERATURE CONDITIONS

8.3–1 ACCESS PROBLEMS The Central Processor is so designed and laid out that there are no special tools needed to effect repairs. All repairs that the service engineer is expected to attempt can be accomplished with the tools provided in the maintenance kit listed in Table 8.1–1.

The Central Processor is also laid out such that there are very few access problems. The ones that do exist are in the lower quarter of the machine where the large a.c. voltage regulators, the connector box, and the blower boxes are located. The problems created here are mainly due to awkward size and weight, and the necessity of removing blower boxes to work on the lower two voltage regulators. The connection box connectors are also difficult to replace due to the limited access space available. Due to system design changes, it may become necessary to replace complete side–panels of the connection box. In such a situation, it may be advisable to remove the connectors from the side panel before removing old, or positioning new side panels into place.

In any case where removal of a part is necessary, it is advisable to pull the assembly drawing of the part in question and study it to determine just exactly where, and how many, fasteners must be undone to allow removal of the part. This can save much time and trouble, especially where fasteners may be hidden by an adjacent part or by the part itself. The method for removing a blower box will be discussed since it is one of the more difficult tasks.

When removing a blower box, it is not possible to merely slip the blower box horizontally out of the machine, rather, it is necessary to

lift the blower box vertically inside the machine and then pull it out of the cabinet. Also, prior to actual removal of the blower box, it is necessary to position the logic and memory panels so they are as far out of the way as possible, and remove part of the trim from the front of the Central Processor. The first step to take prior to removal of the blower box itself is to remove the blower box grill, and the upper grill trim, and the panel seal.

The upper grill trim is a U–section in which the 115 volt a.c. convenience outlets are mounted. The blower grill is held in place by a total of six screws. Two screws located at each end of the grill and two screws located in the middle of the grill secure the grill to the frame and it is very easily removed. The removal of the upper grill trim, however, is a bit more complicated since its securing screws are located under the front edge of the side panels. This necessitates the loosening of the side panels so that they can be pulled out far enough to allow access to the upper grill trim securing screws, of which there are two under each side panel. Once the four securing screws have been removed, it is a simple matter to remove the upper grill trim. There are six screws holding each outer–skin side panel to the frame. Only four of these screws should be removed, however, whenever the side panels must be moved for access to the upper grill trim securing screws. This will prevent alignment problems with the outer side panels. The four securing screws that are to be removed from the side panel are accessible through holes in the inside side panels of the Central Processor. At this time, it will also be necessary to remove the panel seal attached to the inside side panel. At this point we are ready to proceed with the actual removal of the blower box.

8.3–2 HIGH TEMPERATURE CONDITIONS The Central Processor memory is designed to operate below 90^{o}F and there is a much greater chance of errors due to memory malfunction if it is operated above this

temperature. Thus, it is seen that it is very important to keep the blowers functioning and the air filters clean. If the machine temperature rises above 90°F, the high temperature light and warning signal will be actuated. This will probably indicate that the air circulation within the Central Processor cabinet is inadequate.

If a high temperature indication occurs which could not be caused by too high a temperature in the computing room, then the first and easiest thing to check would be the air filters, especially if a system of regular periodic checks of the filters has not been instigated. There is one set of air filters in the Central Processor. This set of filters is mounted between the blower boxes and the front grill and these filters can be removed by swinging the logic panels out and then pulling up on the filter packages. The normal frequency of filter replacement will depend upon the machine environment and will have to be established individually at each installation.

The air filters are throw–away units and should <u>not</u> be cleaned and replaced. Replacement filters are available initially in the Maintenance Kit of each installation. When the last replacement filter has been installed, a package of filters should be immediately reordered from the Customer Engineering Office at Bendix Computer's main plant.

If the high temperature condition exists after the inspection of the filters, then the next step would be to check the blowers. To do this, it is necessary to remove the blower box grill and the air filters. If a motor is not operating, one or more of the following conditions may exist:

1) loss of a.c. power,
2) motor winding or lead open or shorted,
3) faulty capacitor (these are capacitor start–capacitor run motors),

4) bad bearings (these motors use sealed ball bearings).

In all cases where a blower motor is inoperative for reasons other than loss of power, the main plant should be notified immediately. A replacement blower motor will then be dispatched to the installation. Table 8.3–1 lists replacement part numbers for the motor capacitor and bearings for situations where the above procedure is not practical. If the motor winding is burned or open, there is little choice but to send to the main plant for a replacement.

TABLE 8.3–1 Blower Motor Replacement Parts

Bearing	New Departure Bearing No. CWC 8500
Capacitor	5.0 mfd. ± 10 per cent, 330 volts, oil filled

A few words of advice in regard to the replacement of blower motor bearings in the field. The operation of bearing replacement, to be handled properly, requires special equipment designed specifically for this purpose. True, bearings can often be replaced by make–shift means. However, such practices can result in a bent motor shaft or damage to the bearing. Therefore, it is mandatory that bearing replacements in the field be done in shops that are equipped for such operations.

SECTION 8. 4 – PACKAGES

8. 4–1 GENERAL INFORMATION There are many reasons for using printed circuit modules, especially in mass production operations. In these cases printed circuits afford the advantage of high reliability, ease of handling, and reduced cost. On the other hand they have drawbacks on servicing, especially since G–20 packages are soldered into place.

Printed circuit boards and their circuits are very rugged and it takes a lot of rough handling to damage them appreciably. Even if a section of printed circuitry is damaged by nicking or cutting, they can usually be repaired by soldering over the damaged area. The greatest damage that can be done is to raise the printed circuitry from the board by overheating it. Once the printed circuit has been raised, the board must be replaced if it cannot be repaired by methods presented later in this section. In case a circuit board or circuitry should be damaged beyond repair while positioned in the equipment, due to failure of components, circuitry, etc., the card must be removed and replaced. In the maintenance kit shipped with each system there will be at least one replacement package for each type package used in the system. This maintenance kit will also include loose components for replacing any individual component on any package of the system.

8. 4–2 TOOLS A list of tools for each service engineer is given in Table 8. 1–1. It is conceivable that situations may arise where many of the tools on the list may be needed to check or effect repairs on a package; however, the soldering irons are the tools that require the most instruction concerning their use in repairing or replacing modules. As you can see from the list, three different soldering irons are supplied; these will now be discussed.

The 300 watt iron must be used to remove a module from a master board whenever a complete package must be removed. This iron is supplied with two tips, one for removing 11 pin logic packages and one for removing 20 pin memory packages.

The 60 watt iron can be used to remove individual components from the packages whether the packages are mounted in the equipment or removed for repairs; however, it is primarily intended to be used on packages that have been removed. This iron has a special smaller tip and, although the distance between mounted packages is very limited, if the operator is careful this iron can reach the soldered component connections of most of the smaller packages. This iron is often desirable for such use in spite of its relatively large size, because its maximum tip temperature is much less than the smaller Oryx iron, and is less apt to cause raising on the etched printed circuits due to heat.

The Oryx soldering iron is supplied with each system for the express purpose of removing hard–to–get–at individual components from mounted packages. The iron is pencil sized and operates at 6 volts supplied by its own transformer. This is a very handy soldering iron with a long, thin, quick heating tip, especially suited to reach into the tight spaces between the packages. However, it should be made a practice to use this iron only when absolutely necessary since the tip temperature is much greater than the 60 watt iron and if rapid, very careful application of this iron is not observed, it is extremely easy to lift the etched circuit. Also, this iron cannot be left on for a very long period of time or the high temperatures will ruin the tip.

Modifications have been suggested with respect to the problem of burning up the tips of the Oryx iron. One modification is to mount a toggle switch in the secondary of the transformer. This switch would be controlled by the operator and used to open the 6 volt line to the Oryx iron

whenever the iron was not in use. Another suggested modification is to connect a lever operated microswitch in the 6 volt secondary of the transformer. This microswitch would be mounted on the transformer in such a manner that when the Oryx iron is placed into the securing clips mounted on top of the transformer, the microswitch lever arm would be displaced and break the 6 volt supply to the soldering iron. This would automatically protect the soldering iron tip from damage when the iron is not actually being used.

It has been found from experience that there is a potential difference between the 60 or 300 watt irons and the logic panels. This voltage difference may be due to a static charge accumulation on the panels, a voltage present on the soldering iron due to insulation leakage, or both. This voltage difference is great enough that diodes and transistors have been ruined when the iron was applied to the lead of components in a mounted package.

Usually, electric tools have a ground provided as one of three prongs of the electrical connector. The plugs of the 60 and 300 watt irons supplied to the field should have this type of ground connection. If irons so equipped are plugged into a convenience outlet of the unit under repair, the frame of the iron will be grounded to the frame of the machine. This will prevent the danger of component damage from an existing voltage difference between tool and machine.

A 3–prong–to–2–prong adapter is supplied with each maintenance kit. This adapter must be used each time an iron is not plugged into a convenience outlet of the machine being worked on. This adapter must always be used when working on the CC–10, since it is the only piece of equipment in the system that is not provided with its own convenience outlet. The adapter has a ground lead attached to one side of it. The end of the ground lead has a clip which must be connected to the

machine being repaired whenever the adapter is used.

If a soldering iron without a built in ground connection is used, connect a ground wire between the soldering iron and the equipment. A wire 4 to 5 feet long with a clip at both ends has been provided for this purpose.

8.4–3 TECHNIQUES Normally, the packages should be repaired while still mounted in the equipment. All of the components of the small modules and the first row of exposed components of the larger packages can usually be tested and replaced while still in the equipment. There are notable exceptions and one is the case when a package is mounted in a location next to a filter package. The capacitors on the filter packages are so large and closely spaced that it is almost impossible to get at the terminals of components on a board mounted next to it. Under these circumstances, it is necessary to remove the package to effect repairs. Another exception is if repairs must be made on the large packages in positions other than the exposed first row of components. These boards must be removed for repair since the space is too confined to use a soldering iron beyond the first row.

WHEN REPAIRING MOUNTED PACKAGES,

THE ROWS OF PACKAGES BENEATH THE PACKAGE BEING WORKED ON MUST BE PROTECTED FROM FALLING SOLDER, WIRE, ETC.

This can be accomplished by using a piece of cardboard, a rag, etc., positioned to completely cover the top of the first exposed row of packages and secured with masking tape.

When repairing any package that is removed from the panel, the 60 watt iron should be used; when repairing small sized mounted packages

or replacing components from the first row of exposed components of large packages, either the 60 watt or the Oryx soldering irons may be used depending upon the circumstances and the preference of the operator.

To remove a component, normally diodes and transistors, while still mounted in the equipment, certain techniques have evolved and these will now be discussed.

Since the most common cause of raised circuitry is from the heat transferred during soldering, the service personnel must be very careful to get in and out fast with the soldering iron to avoid excessive contact time between the iron and printed circuit. This procedure should be especially adhered to when using the Oryx iron, since it has such high tip temperatures. Also, when using the Oryx iron, it is advisable not to place the tip of the iron directly on the point of connection of component lead to printed circuit bus. Place the tip of this iron on the outer edges of the printed circuit at the point of contact. This method supplies enough heat transfer to melt the solder and loosen the connection. It also lessens the possibility of excessive heat transfer which may raise the printed circuitry.

To actually remove a component, it has been found that insertion of a scribe under the component and then twisting the scribe while heating the connection will often pry the component out of its connection. Difficulties can be encountered with this method, and it may be necessary to use long nose or needle nose pliers instead. When replacing components, first open the holes by heating one side with the soldering iron, while pressing on the opposite side with the scribe. Next, trim the leads of the components to the desired length, insert, and solder.

To replace a type T024 transistor case, solder the leads in place, and

then push the transistor into the securing clip from the end of the clip rather than from the top. Never "snap" into place.

To remove a complete package from a panel, first unwrap and remove all wires from the terminal pins. The 300 watt iron with one of the large tips is then used to loosen the connection while the operator is simultaneously pulling on the package. It is necessary that the soldering tip be flat against the master board in both the X and Y directions to insure that all contacts are freed as rapidly as possible. It is again necessary to get in and out fast with the soldering iron to prevent raising the printed circuits. To replace a package, first clear the holes, then insert the package and push it all the way in until it is flush with the master board. Finally resolder the connections. On the master boards, there are many printed circuit pads that have no electrical connections and whose only purpose is to secure the module to the master board. These securing pads are especially susceptible to lifting. During a module replacing procedure several of these pads may lift and be ruined for securing purposes. If only a few of the securing pads are lifted, they may be neglected. If enough pads are lifted that the module is not firmly secured to the master board, steps to remedy this condition must be taken. The best means to effect a repair on destroyed securing pad locations seems to be to make one complete turn about the terminal with a piece of connection wire, push the wire flush with the master board, and solder it to the terminal. This procedure will provide the necessary mechanical support for the module.

If a piece of printed electrical circuit is lifted on the master board, or on a module, it is a more serious problem and other methods must be used. It may be sufficient to solder a short length of connection wire over the lifted section and to adjacent good sections of the circuit. If a large length of printed circuit is lifted or appears badly damaged, it may be necessary to solder a length of connection wire over the circuit

path from terminal to terminal. If a short length of printed circuit is lifted on a master board, a short length of wire may also be soldered across the bad area. If a bus has been lifted or damaged for the distance of several package locations, Wire–Wrap the pins of the packages on the bad length of printed circuit together. If this is not possible, due to previous Wire–Wraps on the terminals in question, then connection wire laid and soldered over these lengths must be used. A master board should be replaced only after every other conceivable method of repair has failed.

The memory unit, although logically not a part of this section, is related to the problems of modules. It may be that the memory unit will never be replaced in the field, but in case it should become necessary a few words on the suggested procedures will be discussed. First of all, leads leaving the memory must be unwrapped and disconnected at their point of termination. When this has been done, the four shock absorber mounts of the memory unit should be unbolted and the unit removed. The replacement procedure is the reverse of the above.

8.4–4 PREVENTIVE MAINTENANCE Preventive maintenance of transistorized circuits is not economically sound and it is not recommended for packages or circuit cards of the G–20 System. Experience at Bendix Computer indicates that there is no direct relationship between failure rate and preventive maintenance procedures on transistorized circuits. Thus, the G–20 transistorized circuits will be let run until the transistors fail since sufficient faulty components will not be detected to warrant the effort needed for a thorough preventive maintenance program. It should be noted here that this policy concerning preventive maintenance is valid only for solid state electronic sections of the equipment. Regular preventive maintenance procedures will be provided for other sections of the equipment where necessary.

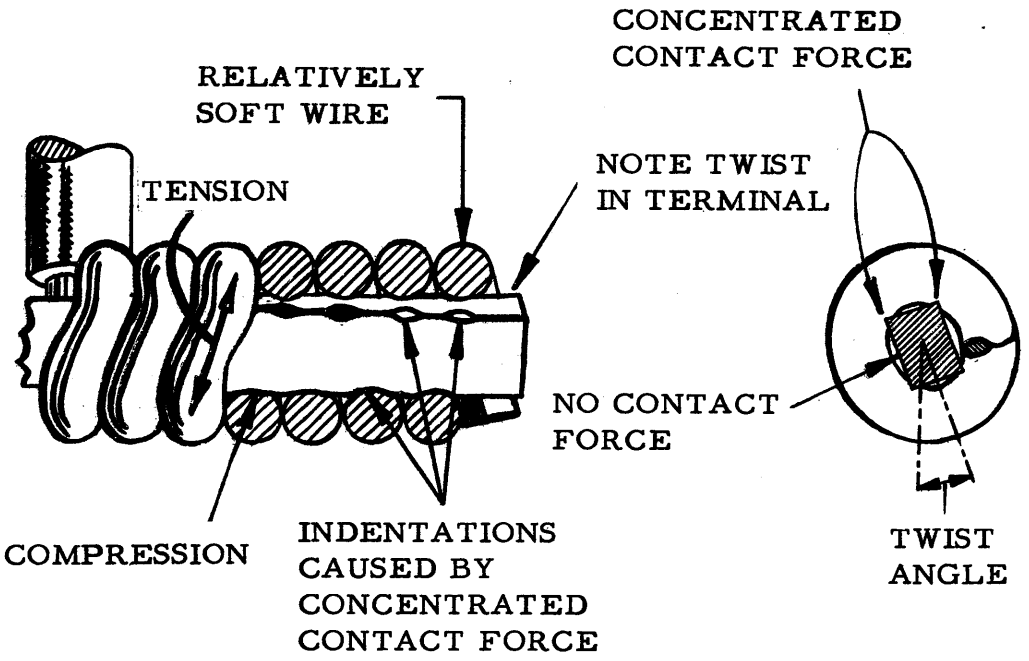
SECTION 8.5 – WIRE–WRAP

Since Wire–Wrap is used extensively in the G–20 System, it is felt that a basic understanding of the theory will be quite helpful to the engineer or service representative. However, the material concerning Wire–Wrap procedures and quality control specifications must be understood and followed by anyone who at any time performs Wire–Wrap operations.

8.5–1 GENERAL INFORMATION Since electronic equipment tends to become more compact and still more complex, methods for making millions of connections in close quarters have been investigated. One result of these investigations was the introduction of the Wire–Wrap solderless connection. Wire–Wrap was initially used extensively by the telephone companies, but as electrical or electronic firms become familiar with it and discover that it can be used to advantage in their assembly processes, more and more companies are turning to Wire–Wrap as the solution to their wiring problems. Thus, when Bendix Computer was faced with the problem of assembling the complex G–20 Computing System, it is not surprising that Bendix chose Wire–Wrap. Not only does it offer a method of fast, reliable connection; it also eliminates possible thermal damage to components or insulation during assembly.

Wire–Wrap is similar in many ways to a rubber band wrapped around a finger. The "barrel" of the Wire–Wrap gun is a rotating spindle that slips over the terminal post. As the spindle of the gun rotates, the connecting wire is wrapped around the terminal post about six times with a very high tension. The connection wire is somewhat elastic and grips the terminal by means of residual tension. The terminal post is also deformed so that a strong connection is ultimately obtained from the elastic stresses in the two members.

FIGURE 8.5–1 Wire–Wrap Showing Stress Points



Of prime consideration in a solderless connection, is good metal–to–metal contact. In the Wire–Wrap scheme, the terminal post has several sharp corners. The wrapping force is so great that at each corner of the terminal the high shearing force of the wire scrapes off surface film from both the wire and the terminal to produce a clean metal–to–metal contact, free from oxidation. Average pressure between wire and terminal at the contact points after wiring is above 29,000 psi; this maintains good metal–to–metal contact and prevents the wrap from being easily dislodged. Also, as this type of connection ages at room temperature or higher, the metal–to–metal bond becomes mechanically stronger due to solid state diffusion between the wire and terminal at contact points. In addition, the contact pressures are great enough to assure a gastight connection between wire and terminal. If the

connections are initially gas free, they will remain that way throughout their life so that corrosion cannot occur at the contact points. Another advantage of Wire–Wrap is that the contact area and pressure are not up to the discretion of each operator, but rather are controlled by the tool, the wire, and the terminal, all of which are generally uniform.

Also, in an ordinary hook–and–solder connection, there is a vibration stress concentration at the junction of wire, solder and terminal. A solderless wrapped connection, however, develops a cantilever spring action that develops little or no vibrational stress concentrations. This consideration, along with high contact pressures, increase the mechanical stability of the connection.

If further information is desired concerning the theory of solderless wrapped connections, the following references are recommended:

1) "Solderless Wrapped Connections", R. F. Mallina, et. al., Bell Systems Technical Journal, Vol. 32, May, 1953.

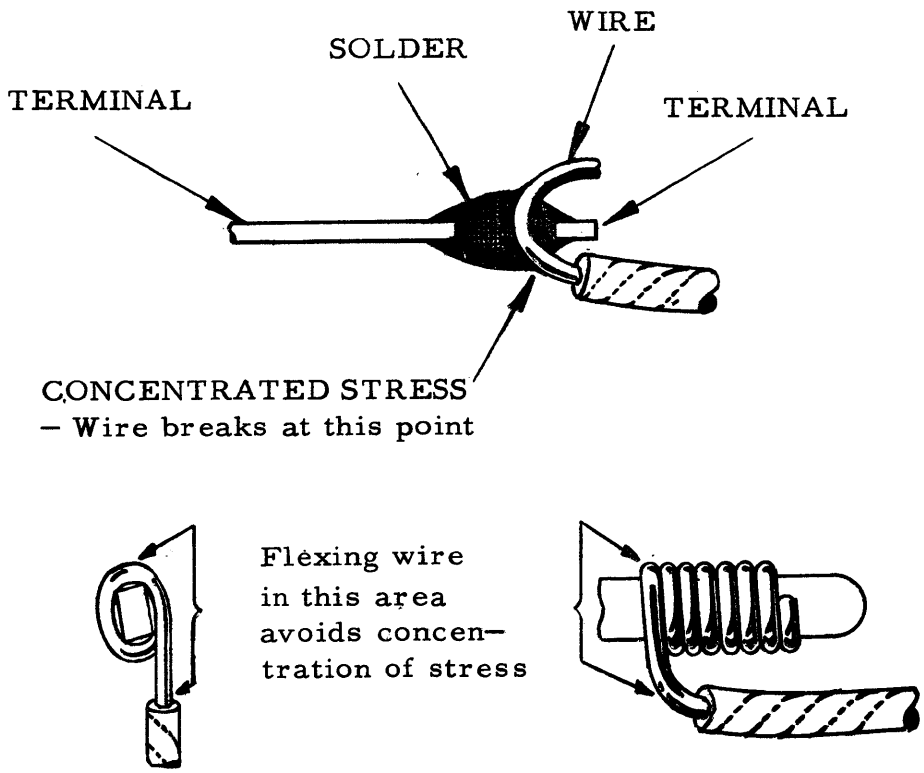
2) "Practical Quality Control Measures Insure Wire–Wrap Reliability", H. F. Wilson for Gardner–Denver Company.

8.5–2 TOOLS Under normal operating conditions, the only tools needed are a Wire–Wrap gun, a wire unwrapping tool, and a wire stripper. Actually, however, it may often be necessary to use soldering equipment and long nosed pliers, especially in the case of field service personnel.

There are several types of wire guns available, but we are interested only in two types: the air operated gun and the hand operated gun. The air operated gun is used at the main plant, while the hand actuated gun will be supplied to service representatives for use in the field. Both guns are normally trouble–free, reliable pieces of equipment, and little

trouble is anticipated in these guns. However, if the gun is dropped or otherwise misused, the wrapping bit and sleeve may become damaged and inoperative. In such cases, the field service engineer should notify the Customer Engineering Office at Bendix Computer's main plant for a new bit and sleeve assembly. If after replacement of the bit and sleeve assembly the wrapping gun is still inoperative, or if it is felt that something other than the bit and sleeve assembly is inoperative, then the Customer Engineering Office should be notified and the complete assembly should be returned to Bendix Computer's main plant for repair. The wire unwrapping tool and the wire stripper will not be discussed since they are such simple devices. Further information concerning Wire–Wrap guns may be obtained from Gardner–Denver Company, Quincy, Illinois.

FIGURE 8.5–2 Comparison of Wire–Wrap and Soldered Connection

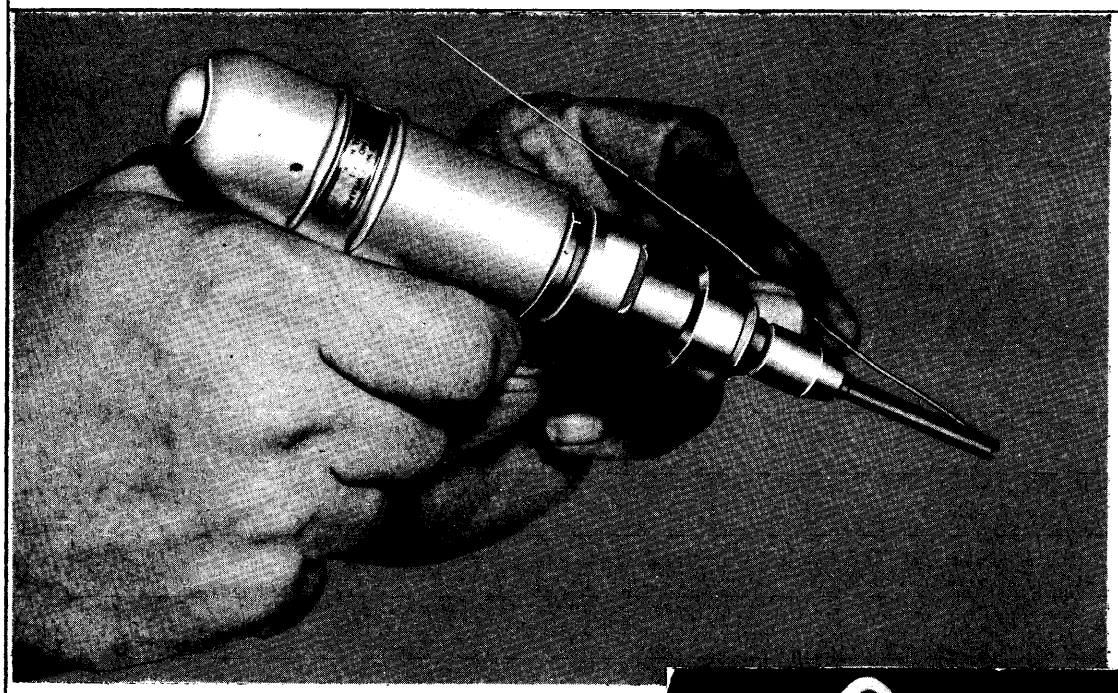


8.5–3 TECHNIQUES Diagrams and a description of the method for making Wire–Wrap connections are shown in Figure 8.5–3. A discussion of Wire–Wrap solderless wrapped connection procedures and restrictions follows.

In the G–20 Computing System, all solderless wrapped connections are made with solid 22 or 24 gauge, tinned copper wire. The connector wire should be stripped back for a length of 1 inch or 1–1/4 inch depending on whether a package or a connector pin is to be wrapped. This wire is then applied to the terminal, looking at the free end of the terminal, in a clockwise direction only. If the wire is nicked or damaged during the stripping operation, or if the wire insulation shows physical damage, then this wire must be replaced. The terminal pin itself must be free of solder for a minimum of 0.3 inch, measured from the free end of the terminal, if solderless wrapped connections are to be made. The terminal should also be free of foreign matter, burrs, or signs of deterioration that could cause an unreliable connection.

The start of the first solderless wrapped connection should be at least twice the wire diameter from the solder bud at the terminal base. The second wrap shall be at least twice the wire diameter from the end of the first Wire–Wrap. The wrap itself shall consist of a minimum of five consecutive non–overriding turns of bare wire in contact with the terminal. More than seven turns are permissible, but not desirable. Also, although it is not desirable, overriding the turns (the wire is not in contact with the terminal) is permissible only after the minimum of five turns has been completed. The finished wrap should have all the turns in contact with each other. However, complete turns may be separated and acceptable provided that three of the spaces between turns (not necessarily adjacent) as viewed from one side, do not exceed approximately 0.005 of an inch at the closest point. As a guide, it might be mentioned that the thickness of this paper is approximately 0.0045 of

FIGURE 8. 5–3 Making the Solderless Wrapped Connection

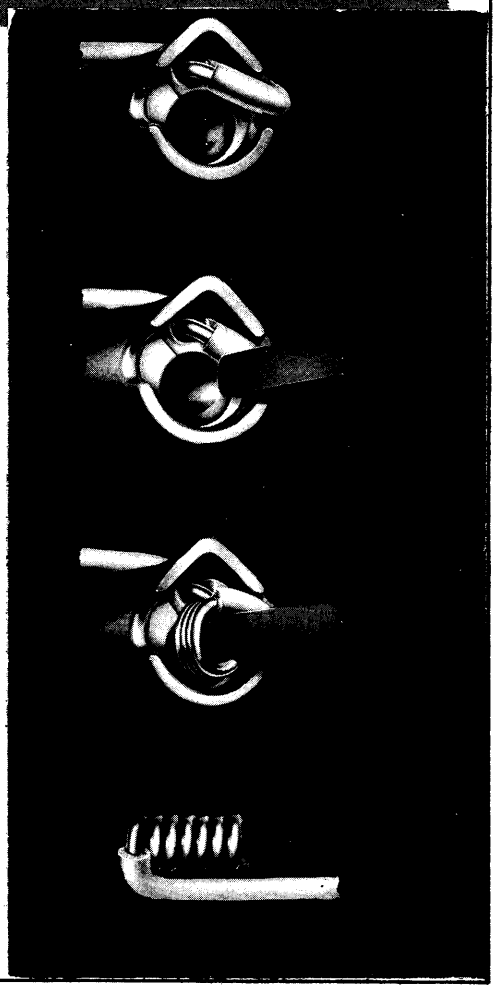


Making the solderless wrapped connection is simple . . .

The end of the wire is skinned 1 inch or 1 1/4 inches, depending on whether a module or a connector pin is to be wrapped.

The skinned end of the wire is inserted into the wrapping bit of the Wire–Wrap Tool, the bit then placed over the terminal and the trigger squeezed. In a fraction of a second, the rotating wrapping bit winds the wire around the terminal to produce a permanent, tightly–wrapped electrical connection.

The illustration at the right shows the simplicity of the wrapping cycle in making a solderless wrapped connection.



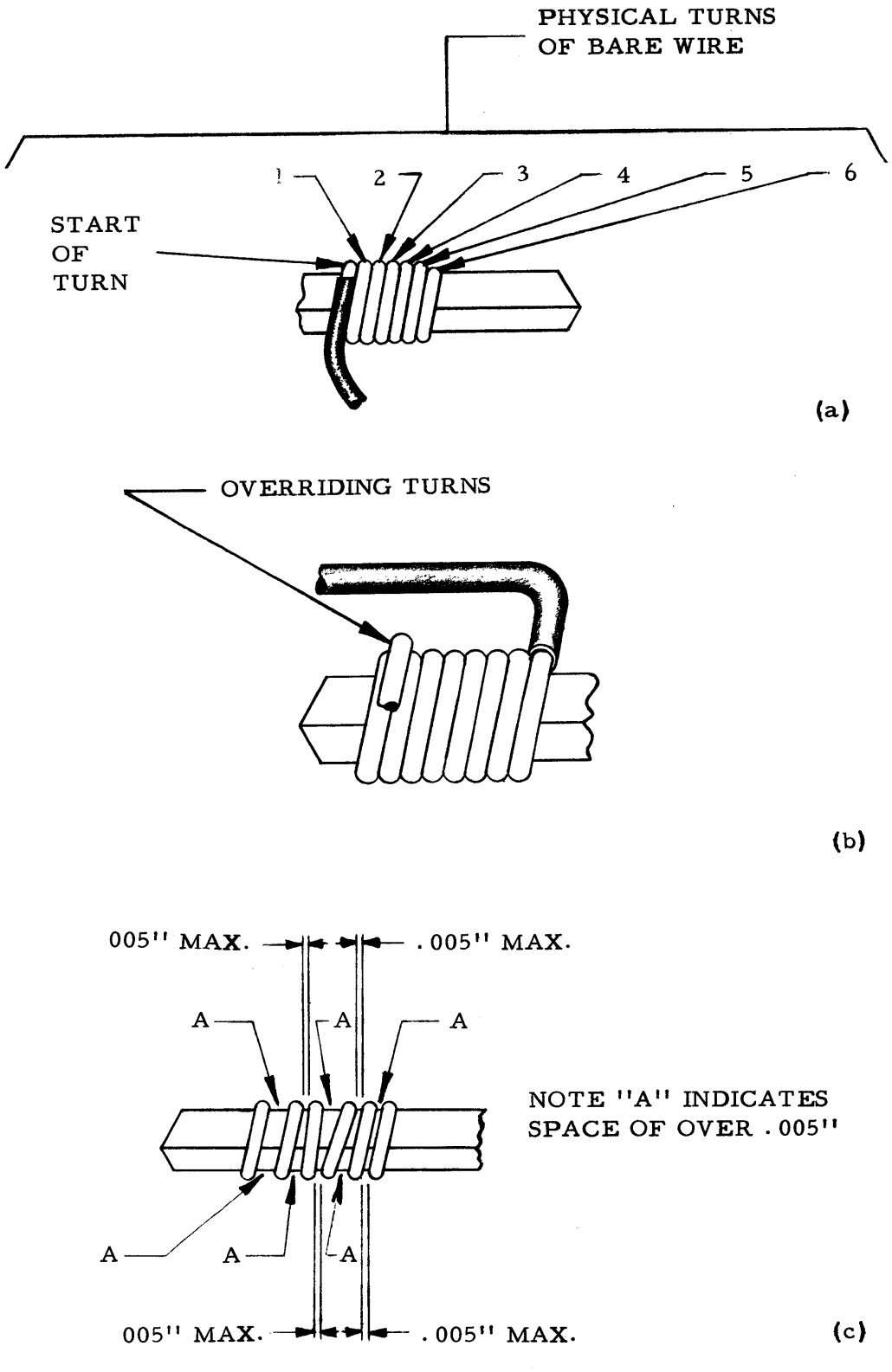
an inch. Also, on an acceptable wrap there shall be no more than 1/32 inch maximum separation between the insulation and the first bare wire contact to the terminal. The insulation, on the other hand, shall not contact or encircle more than one side of the terminal. In addition, the free end of the Wire–Wrap shall not exceed more than 1/16 of an inch beyond the last point of contact with the terminal. The above specifications define an acceptable Wire–Wrap, solderless wrapped connection and abridgement or total disregard of these specifications may produce a mechanically questionable connection subject to failure and a professionally unacceptable piece of work.

> REWORK CANNOT BE ACCOMPLISHED BY DOCTORING THE TERMINAL! THE USE OF PLIERS, SOLDERING AID, OR WIRE UNWRAPPING TOOL, TO TIGHTEN WRAPPED CONNECTIONS, IS UNACCEPTABLE!

All unacceptable wire wrapped connections must be redone, and if necessary to insure a good, mechanically strong connection, the terminals and wire wrapped connections may be soldered.

Strangely enough, although one beauty of this method of connecting is that soldering is eliminated, there are numerous exceptions to this rule that all personnel that might be involved in actual wire wrapping must know. First of all, whenever a terminal pin has been wrapped and unwrapped one time, it is mandatory to solder the next connection that is made to this terminal. Also, if a connector wire that has been unwrapped from a terminal is to be rewrapped (a new area of the connector should be used if possible), then this connection must be soldered no matter what the condition of the terminal to be wrapped. Also, where a terminal contains solder either on an existing connection or resulting from a previously soldered connection, then all connections

FIGURE 8.5–4 Wire–Wrap Connection (a) Preferred (b) Acceptable (c) Separation Between Turns



added to this terminal must be soldered. The connection should be soldered if there is visual damage or burring or if the terminal shows signs of insufficient plating. Another time, although infrequent, when the connection must be soldered is when stranded wire is laid adjacent to the terminal pin and a connecting wire is wrapped around both stranded wire and terminal pin. Finally, whenever there is any question about the soundness of any connection, especially in the field where proper test methods are not available, the connection should be soldered. Remember, however, that excessive soldering will defeat the purpose of Wire–Wrap connections, and also, when modifications or repairs are to be made, soldered connections are much more difficult to remove. Normal unwrapping is easily accomplished using the unwrap tool; however, when soldered wrapped connections are to be removed, excess solder must first be removed from the connection, and then long nosed pliers should be used to carefully unwrap the connection.

In the above sections we have discussed what a good Wire–Wrap looks like and what to do when the Wire–Wrap does not meet these specifications. Sometimes, however, the wrapped connection may look good, but not be tight. This leads to future problems. In the production area, the operators are highly skilled and periodic quality control practices are used to test the acceptability of their work. In the field, the field service personnel cannot be expected to be experts in the use of a Wire–Wrap gun, and proper test methods are not usually available.

The Wire–Wrap gun in the hands of a skilled operator will normally produce acceptable, uniform wrapped connections. The wrapping gun, however, is not infallible and the operator can influence the stripping force of a connection by exerting too much force on the tool during the wrapping operation, or by withdrawing the tool too rapidly during the operation. Strip test tools and gauges are used to determine the force required to pull a connection off a terminal. In the field, these are not

normally available and the Wire–Wrap operator has to estimate whether a wrapped connection is acceptable or not. There are ways even in these circumstances, however, to prevent a majority of the poor connections. First of all, the service engineer should have a test board to practice a few wraps before attempting to make a wrapped connection in the machine. Next, after the connection has been made, the service engineer can make a modified strip test to see if the connection is tight enough. A good wrapped connection requires nine pounds or more of force to strip the complete connection from the terminal. If the field engineer, pulling at the rear of the completed connection with his fingernails, can move it or notice any signs of looseness, then the wrap should be redone and/or soldered. The operator should not be concerned about applying too much force since a 5 to 6 pound force is normally the maximum that can be achieved with a steady pull.

8.5–4 WIRE–WRAP CHECK LIST

1) Direction of turns .... clockwise,
2) Size and type of wire .... No. 22 or No. 24 solid, tinned copper wire,
3) Length of insulation stripped .... 1 inch or 1–1/4 inch,
4) Terminals .... free of deterioration and foreign matter,
5) Start wrap .... twice wire diameter distance from solder bud or other wrap,
6) Number of turns .... five to seven,
7) Separation between turns .... see main write–up,
8) Overriding .... permissible after minimum of five turns completed,
9) Separation between insulation and terminal .... 1/32 inch maximum between insulation and the first base wire contact to terminal, ... insulation shall not contact or encircle more than one side of terminal,

10) Projecting free end of wrap .... no more than 1/16 inch beyond last point of contact with terminal,
11) Insulation .... if physically damaged, replace wire,
12) Nicked or damaged wire .... replace connection,
13) Doctor connection .... NEVER!
14) Solder connections ....
    a. if first rewrap of a terminal,
    b. if first rewrap of length of terminal wire,
    c. if terminal contains solder, or has ever been soldered,
    d. if damage to terminal exists,
    e. if stranded wire laid next to terminal and then wrapped,
    f. if in doubt about connection.

MEMORANDUM

THE Bendix CORPORATION

BENDIX COMPUTER DIVISION

TO: Whom It May Concern FROM: Raymond Fillingim

SUBJECT: Central Processor Manual Erratum DATE: January 24, 1963

We realize that the accompanying errata listing is rather lengthy. Also at some future date the pages incorporating errors will be reprinted. Therefore, the following method of handling these errata is suggested.

Instead of penciling in all of the corrections, you might prefer to simply mark the pages in error (being sure these marks are distinctive enough that they will not be missed in subsequent references to these pages). These marks will then remind you to refer to the accompanying list for corrections.

Ray Fillingim

Raymond Fillingim

jk
Enc.

CENTRAL PROCESSOR MANUAL, VOL. I (BER 10622) ERRATUM

TABLE OF CONTENTS, LIST OF ILLUSTRATIONS, LIST OF TABLES

PAGE	LOCATION		CORRECTION
i	Section 2.3, Page Location	Reads Should Read	2-11 2-10
vi	Figure No. Column, Between 4.2-6 and 4.3-1	Add	4.2-7*
vi	Bottom of Page	Add	*4.2-7 Transfer Paths of CD Register for Bootstrap Operation 4-17
viii	Figure No. 7.1-15, Description Column, 2nd Line	Reads Should Read	Figure 12.1-13 Figure 7.1-14
x	Table No. 2.3-1, Page Location	Reads Should Read	2-12 2-11
x	Table No. 2.3-2, Page Location	Reads Should Read	2-13 2-12
x	Table No. 2.3-3, Page Location	Reads Should Read	2-14 2-13

CHAPTERS 1 THROUGH 8

PAGE	LOCATION		CORRECTION
2-6	Line 12	Reads Should Read	. . . PE register are set to zero . . . . . PE register are ignored . . .
2-8	Operation Column of Address Preparation Commands, 7th Formula	Reads Should Read	X + (ACC) → (OA) |X + (ACC)| → (OA)
2-8	Operation Column of Address Preparation Commands, 8th Formula	Reads Should Read	-X + (ACC) → (OA) |-X + (ACC)| → (OA)
2-8	Operation Column of Add and Subtract Commands, 7th Formula	Reads Should Read	X + (ACC) → (ACC) |X + (ACC)| → (ACC)
2-8	Operation Column of Add and Subtract Commands, 8th Formula	Reads Should Read	-X + (ACC) → (ACC) |-X + (ACC)| → (ACC)
2-8	Operation Column of Add and Subtract Tests Commands, 5th Formula	Reads Should Read	-X + (ACC) > 0 |-X + (ACC)| > 0

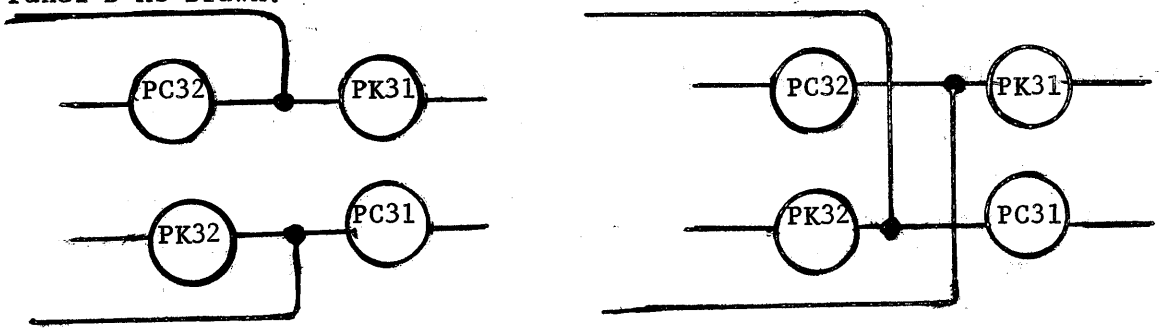
PAGE	LOCATION		CORRECTION
2-8	Operation Column of Add and Subtract Tests Commands, 8th Formula	Reads Should Read	X + (ACC) > 0 |X + (ACC)| > 0
2-8	Operation Column of Logic Tests Commands, 3rd Formula	Reads Should Read	31 X + (ACC) 0 > 0 31|X + (ACC)|0 > 0
2-8	Operation Column of Logic Tests Commands, 4th Formula	Reads Should Read	31 -X + (ACC) 0 > 0 31|-X + (ACC)|0 > 0
2-9	Note Under Register Operations, Last Line	Reads Should Read	*The CA register can be read only by an ERO or ERA opcode. *The CA register can only be used with an ERO or ERA opcode. The PE register can only be used with the LDR opcode.
2-9	Note Under Repeat Operations, Last Line	Reads Should Read	. . . for is met. . . . for is met or until the specified block length is reached.
2-10	Lines 23 and 24	Read Should Read	. . . location (1 through 63) to be operated on. . . . location (0 through 63) to be operated on.*
2-10	Bottom of Page	Add	*An I field of 00 in an Index Command specifies memory location 00000, and this location will be operated on by these commands. However, an I field of 00 as used with the general case of Operand Assembly indicates no index address is specified.
2-12	Line 6	Reads Should Read	. . . of the 63 index locations . . . . . of the 64 index locations . .
3-2	Line 8	Reads Should Read	. . . discussed in Part IV, . . . discussed in Volume II,
3-2	Line 21	Reads Should Read	. . . Chapter 12 of Part IV . . . . . . Chapter 12 of Volume II . .
4-2	Lines 1 and 2	Read Should Read	The MA register starts the memory cycle which brings the new command from memory and places it in the B At the start of a memory cycle the new command is brought from memory and placed in the B

PAGE	LOCATION		CORRECTION
4-2	Line 12	Reads	. . . register which starts the memory cycle which brings . . .
		Should Read	. . . register. The memory cycle now brings . . .
4-3	Line 16	Reads	. . . peripheral units, control . .
		Should Read	. . . peripheral units,* control. .
4-3	Bottom of Page	Add	*At the present time only the Central Processor is capable of receiving and acting upon interrupts. Therefore, with present equipment, the Central Processor will transmit interrupts to another Central Processor only.
4-6	Line 14	Reads	. . . locations 1 through 63) . .
		Should Read	. . . locations 0 through 63) . .
4-8	CD7 Column at Bottom ($\overline{9} \wedge \overline{8} \wedge \overline{6}$)	Add	1
4-8	CD Decoding $9 \wedge \overline{8} \wedge 7 \wedge \overline{6} \wedge 12 \wedge \overline{11} \wedge \overline{10}$	Reads	TIC 157
		Should Read	TLC 157
4-15	Line 1	Reads	. . . the least 2 bits . . .
		Should Read	. . . the least significant 2 bits . . .
4-31	Line 26	Reads	. . . Figure 4.3-2 it is seen . .
		Should Read	. . . Figure 4.4-2 it is seen . .
4-33	Table 4.4-2, Bit Position 3 of J Register	Reads	UWB
		Should Read	JWB
4-34	Figure 4.4-3, Immediate Right of ULA and ULB	Reads	DATA word
		Should Read	LOGIC word
4-34	Figure 4.4-3, Immediate Right of UDA and UDB	Reads	LOGIC word
		Should Read	DATA word
4-34	Figure 4.4-3, Between UWA, UWB and JWA, JWB JWC, JWD	Reads	OUTPUT Interrupt Requests INPUT
		Should Read	TRANSMIT Interrupt Requests RECEIVE
4-36	Line 12	Reads	. . . and Table 4.3-2.
		Should Read	. . . and Table 4.4-2.
4-38	Line 1	Reads	. . . is therefore forbidden.
		Should Read	. . . is therefore forbidden.*

PAGE	LOCATION		CORRECTION
4-38	Bottom of Page	Add	*A negative zero exponent in the PE register is not detected by the hardware. It therefore becomes the programmer's responsibility to avoid this condition.
4-39	Figure 4.4-6, Bottom	Reads	. . . necessary and sufficient to . . .
		Delete	. . . and sufficient . . .
5-1	Figure 5.1-1, Lower Right, Above D Register	Reads	B(L31)41D21
		Should Read	B(L21)41D21
5-3	Line 23	Reads	. . . double precision operation. . . .
		Should Read	. . . double precision operation.*
5-3	Bottom of Page	Add	*During index operations, the number to be stored is always shifted to zero exponent and truncated. Therefore, the EA(L21)B transfer, when used in an index operation, will always result in zeros being stored in 27B21.
5-4	Line 2	Reads	. . . operation is performed.
		Should Read	. . . operation is performed.*
5-4	Bottom of Page	Add	*Bits 30 and 31 of the B register are reset during a store zero operation, but this is accomplished by the M(0)31B0 transfer (the M register being cleared before the transfer).
5-5	Line 11	Reads	. . . for a negative exponent, . . .
		Should Read	. . . for a negative, non-zero exponent, . . .
5-5	Line 16	Reads	. . . overflow (when a . . .
		Should Read	. . . underflow (when a . . .
5-5	Line 17	Reads	. . . exponent overflow condition . . .
		Should Read	. . . exponent underflow condition . . .
5-5	Line 17	Reads	. . . For a positive exponent . . .
		Should Read	. . . For a positive or zero exponent . . .

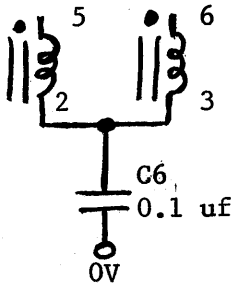
PAGE	LOCATION		CORRECTION
5-6	Table 5.1-3, Under When Used Column, 2nd Line	Reads Should Read	. . . is negative . . . . . . is negative and exponent is not zero . . .
5-6	Table 5.1-3, Under When Used Column, 4th Line	Reads Should Read	. . . is plus . . . . . . is plus or exponent is zero. . .
5-6	Table 5.1-4, Title Box 1st Line	Reads Should Read	. . . of Bits 31 through 27 . . . . . . of Bits 31 through 21 . . .
5-6	Table 5.1-4, Last Line	Reads Should Read	. . . (PE) as exponent) . . . . . . (PE) as exponent) exponent sign in single or double precision floating point numbers. (Set = minus, Reset = plus)
5-6	Table 5.1-4, Bottom	Add	26B21 All six bits reset in a single or double precision floating point number, indicates a zero exponent (EXZ).
5-7	Line 12	Reads Should Read	. . . in Part IV where. . . . . . in Chapter 12 of Volume II where . . .
5-9	Line 1	Reads Should Read	. . . of an add or subtract command is . . . . . . of a sum or difference operation is . . .
5-11	Line 12	Reads Should Read	. . . WS (Working Sign). . . . WS (Working Sign).*
5-11	Line 14	Reads Should Read	. . . indicating subtract. WS . . . . . indicating a difference. WS . . .
5-11	Bottom of Page	Add	*Provided the contents of the accumulator is being used as one of the operands.
5-15	Lines 25 and 26	Read Should Read	. . . if the subtrahend and denominator are . . . . . . if one of the operands of a subtract operation and the denominator of a divide operation are . . .
5-16	Line 19	Reads Should Read	. . . that the subtrahend of a. . . . . that one of the operands of a . . .

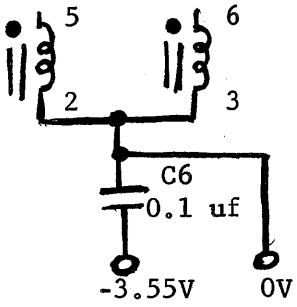
PAGE	LOCATION		CORRECTION
5-16	Line 20	Reads	. . . to the minuend. It . . .
		Should Read	. . . to the other. It . . .
5-16	Lines 22 and 23	Read	. . . that the minuend was larger than the subtrahend and the . . .
		Should Read	. . . that the smaller of the two operands was complemented and the . .
5-24	Table 5.3-2, Time Column, 2nd Line	Reads	0.048 msec.
		Should Read	0.048 μsec.
5.25	Line 5 (Leapfrog Kill Formula)	Reads	. . . ∧ . . . $(\overline{N}_{n-1} \vee \overline{D}_{n-1}) \wedge$ . .
		Should Read	. . . ∧ . . . $(\overline{N}_{i-1} \vee \overline{D}_{i-1}) \wedge$ . .
5-26	Figure 5.3-8, Input OR Gate to LFC Inverter 3rd Term Down	Reads	$N4 \wedge \overline{D4}$
		Should Read	$\overline{N4} \wedge \overline{D4}$
5-24	Table 5.3-2 Time Column, Last Line	Reads	or 2.006 μsec.
		Should Read	or 2.016 μsec.
5-24	Table 5.3-2, Elapsed Time Column, Last Line	Reads	2.289 μsec.
		Should Read	2.299 μsec.
5-25	Line 1	Reads	. . . are generated in Figure . . .
		Should Read	. . . are illustrated in Figure . .
5-28	Panel C, Output of Parallel Inverters Above PC16 and PK17 Inverters	Reads	26LFK13
		Should Read	26LFC13
5-28	Panel C, Output of Parallel Inverters Below PK16 and PC17 Inverters	Reads	26LFC13
		Should Read	26LFK13
5-28	Panel D As Drawn:		Should Be:



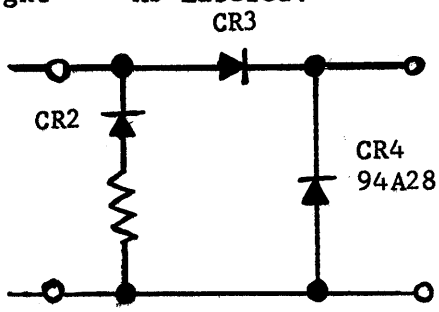
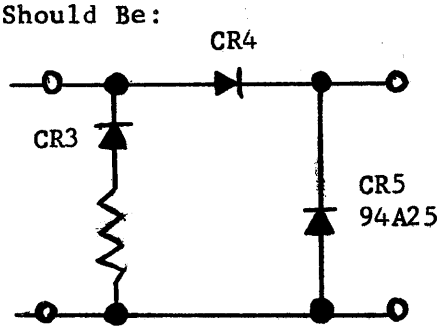
5-33	Lines 7 through 11	Delete	
5-34	Line 13	Reads	. . . sections of Part IV . . .
		Should Read	. . . sections of Volume II . . .

PAGE	LOCATION		CORRECTION
5-35	Line 5	Reads Should Read	. . . complement form. Certain . . . . . complement form.* Certain. .
5-35	Bottom of Page	Add	*When transferring the value of the exponent to the EP register from the B register complementing is done only if the exponent is negative and non-zero (B27 ∧ EXZ).
5-40	Table 5.4-2, (Borrow Out) Column, 2nd Number Down	Reads Should Read	0 1
6-11	Line 9	Reads Should Read	. . . or $1,000_8$, words . . . . . . or $10,000_8$, words . . .
6-13 and 6-14	Figure 6.2-2 and Figure 6.2-3, Bottom of Each	Add	NOTE: The memory addresses on this page are shown in octal and decimal.
6-19	Figure 6.3-3, Blocking Oscillator Circuit	Add	A line connecting the collector of Q5 with the collector of Q6
6-19	Figure 6.3-3, Blocking Oscillator Circuit	As Drawn:	Should Be:
6-21	Line 12	Reads Should Read	. . . through Q5 to . . . . . . through C5 to . . .
6-24	Figure 6.3-5, Top, Center, D.C. Voltage Input	Reads Should Read	-12V +12V
6-29	Line 18	Reads Should Read	. . . available in Part III . . . . . . available in Volume II. . .
6-33	Lines 20 and 21	Delete	. . . of the read portion of a memory cycle. . .
6-38	Center of Page, Diode Immediately Below Q2	Label	CR1
6-38	Center of Page, Diode Immediately Below Q3	Label	CR2

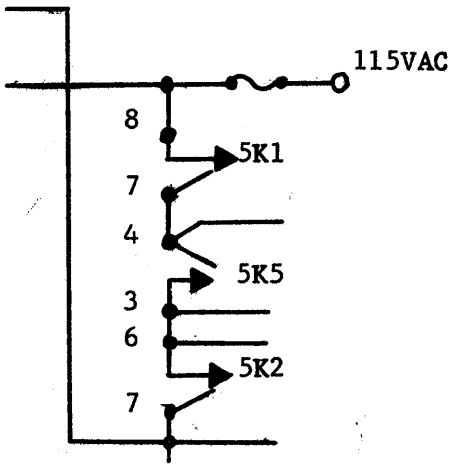


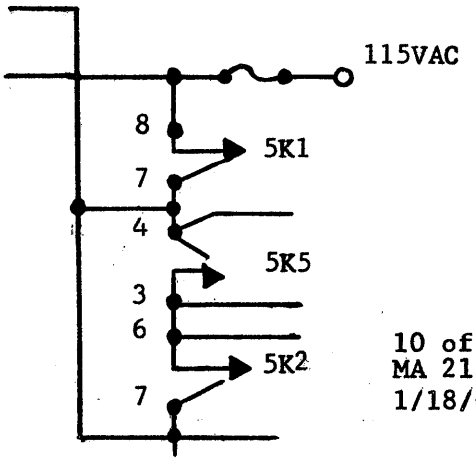


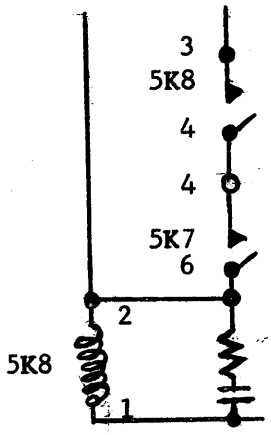
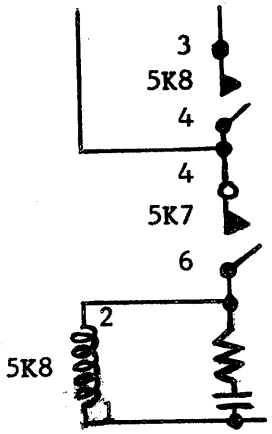
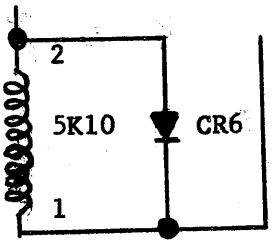
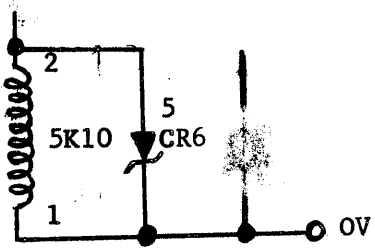
PAGE	LOCATION		CORRECTION
6-38	Bottom, Right of Center, D. C. Voltage Input at R2	Reads Should Read	-100V -102.5V
6-38	Bottom, Left of Center, D. C. Voltage Input at R1	Add	-102.5V
6-39	Line 16	Reads Should Read	Bit 33 of . . . Bit 32 of . . .
6-39	Line 21	Reads Should Read	. . . bit 33 contains . . . . . . bit 32 contains . . .
6-39	Line 23	Reads Should Read	. . . to bit 33. Then . . . . . . to bit 32. Then . . .
6-40	Line 20	Reads Should Read	. . . to bit 33 of the . . . . . . to bit 32 of the . . .
6-40	Line 23	Reads Should Read	. . . to bit 33 is not . . . . . . to bit 32 is not . . .
6-40	Line 23	Reads Should Read	. . . into bit 33, thus . . . . . . into bit 32, thus . . .
6-50	Top, Center, Legend Box	Reads Should Read	*From DC-11 **From MM-10 ***From Central Processor or DC-11 *From MM-10 ** From Central Processor or DC-11 ***From DC-11
6-50	Center of Page, 3rd Line Receiver Down, Input	Reads Should Read	SM3 LM3
6-56	Line 3	Reads Should Read	Figure 6.6-5 has . . . Figure 6.6-4 has . . .
6-56	Line 12	Reads Should Read	. . . Figure 6.6-5 is a . . . . . . Figure 6.6-4 is a . . .
6-58	Lines 11 and 12	Delete	. . . solenoid 12K4 energized provides a hold contact for 12K5, and 12K5 keeps the D. C. voltages on.
6-58	Line 27	Reads Should Read	. . . the $12 volt . . . . . . the +12 volt . . .

PAGE	LOCATION		CORRECTION
7-9	Figure 7.1-6, D4 Package, Immediately Below M	Add	A
7-15	Table 7.1-2, V03 Package, R_L Column	Reads Should Read	7.10K 2.10K
7-15	Table 7.1-2, V13 Package, BCD No. Column	Reads Should Read	1C2312 1C2313
7-18	Figure 7.1-15, Title Box	Reads Should Read	. . . to Figure 7.1-13 . . . to Figure 7.1-14
7-22	Figure 7.2-1, Upper Right	As Labeled:  	Should Be: 
7-22	Figure 7.2-1, Center, Test Point Pin Number	Reads Should Read	J 1J
7-22	Figure 7.2-1, Lower Left CR1 Part Number	Reads Should Read	94928 94A28
7-24	Figure 7.2-2, Lower Right, Value of Capacitor Between Pins D and E	Reads Should Read	4uf 47uf
7-26	Figure 7.3-1, Top AND Gate, Lower Input Term	Reads Should Read	$\overline{\text{SH}}$ CS(
7-26	Figure 7.3-1, Bottom AND Gate, Upper Input Term	Reads Should Read	$\overline{\text{CSC}}$ SHD
7-28	Lower Right, Logic Term at Anode of CR23	Reads Should Read	$\overline{\text{C1}}$ C1
7-28	Lower Right, Junction of Anodes of CR23, 24, and 25	Add	Tie Point (⊕)
7-30	Line 12	Add	It is impossible to hand clock the G-20 through an external memory read cycle and read into the internal (Central Processor) B register.

PAGE	LOCATION		CORRECTION
7-33	Figure 7.3-6, Center, Switch at Base of Q13	Reads Should Read	S/F S1F
7-33	Figure 7.3-6, Center, Switch at Base of Q13 Upper Contact (To R6)	Label	Fixed
7-33	Figure 7.3-6, Center, Switch at Base of Q13 Lower Contact (To R28)	Label	Variable
7-40	Figure 7.3-10, Right Center, Value of Capacitor C15	Reads Should Read	01uf .01µf
7-41	Figure 7.3-11, Lower Right, OR Gate Feeding C2 Pulse Amplifier, Lower Input Term	Reads Should Read	Man. C2 Man.
7-42	Line 8	Reads Should Read	two ten-input AND-gates which are in turn fed into two ten-input OR- two ten-input OR-gates, which are in turn fed into two ten-input AND-
7-43	Line 4	Reads Should Read	. . . Thus, it can be seen that the . . . . . . If no asynchronous events (e.g. input/output) are in process the . . .
7-43	Lines 17 and 18	Delete	. . . or completely insufficient .
7-43	Line 18	Delete	. . . if this should happen, it would mean that
7-43	Lines 19 and 20	Delete	
7-55	Figure 7.4-6, Upper Left, DC Voltage Input	Add	~~-15V~~ approx. -18V
7-58	Upper Right	As Drawn:	Should Be:





PAGE	LOCATION		CORRECTION
7-58	Left, Center	As Drawn:	Should Be:
			
7-58	Right, Center, AC Voltage Input at 12L2-3	Reads Should Read	115VAC 115VAC RETURN
7-58	Lower Right	As Drawn:	Should Be:
			
7-59	Lines 2 and 3	Read Should Read	. . . and immediately following relay . . . . . . and relay . . .
7-61	Line 3	Reads Should Read	. . . SLAVE position. The . . . . . . SLAVE position*. The . .
7-61	Line 5	Reads Should Read	. . . 5Ks, 5K8 and 5K9 are . . . . . . 5K3 and 5K8 are . . .
7-61	Bottom of Page	Add	*Note also that since the system switch being referred to is the system switch on the Central Processor, the Central Processor must be in the SLAVE condition or ON condition before it can send a SYSTEM ON signal out over the communication line.
7-66	Figure 7.6-2, Center, Logic Input Signal	Reads Should Read	URB∧TGO URB∧TGO
8-31	Line 8	Reads Should Read	. . . connector pin is to . . . . . . connector pin, respectively, is to . . .

PAGE	LOCATION		CORRECTION
8-32	Line 6	Reads	pin is to be . . .
		Should Read	pin, respectively, is to be . .
2-8	Immediately Below Address Preparation Commands	Add	NOTE: The accumulator is not disturbed in these opcodes.
2-8	Note, Immediately Below Address Preparation Commands	Delete	
4-10	Line 4	Reads	the ERA opcodes . . .
		Should Read	the ERO opcodes . . .
4-17	Figure 4.2-7, Center, Immediately Below and to Left of "opcode"	Reads	8
		Should Read	12
5-41	Line 23	Reads	. . . equal and positive
		Should Read	. . . equal at time of subtraction
6-34	Line 23	Reads	. . . the collector of Q2 or Q3 will . . .
		Should Read	. . . the cathode of CR1 or CR2 will . . .
6-34	Lines 24 and 25	Reads	. . . the collectors of both Q2 and Q3 are . . .
		Should Read	. . . the cathodes of both CR1 and CR2 are . . .
6-35	Line 22	Reads	(When TR and TW are . . .
		Should Read	(When $\overline{TR}$ and $\overline{TW}$ are . . .
6-41	Figure 6.5-1, Lower Right, Formula for PW	Reads	. . . B32] $\wedge$ $\overline{XPC}$
		Should Read	. . . B32] $\vee$ XPC
6-46	Line 3	Reads	. . . signals DC1 through DC5 are . . .
		Should Read	. . . signals DC0 through DC4 are . . .
6-48	Line 4	Reads	Part IV
		Should Read	Volume II
7-41	Figure 7.3-11, Upper Left Input Term of AND gate on set side of SRC F/F	Reads	C1
		Should Read	C1*
7-44	Figure 7.3-12, (C), First Line	Reads	. . . set by SSC* _ $\wedge$ C1* . .
		Should Read	. . . set by SSC* 1 $\wedge$ C1* . .

Note: Figure 7.5-1 (Page 7-58), as corrected by this erratum serves as a valid guide to the operation of the turn-on/turn-off cycle circuitry. It should not, however, be used as a troubleshooting schematic because several of the terminal strip connections are called out wrong. A corrected drawing, suitable for trouble-shooting, will be sent out in the near future.