Customer Engineering
Instruction Manual

CONTROL DATA® 1604-A COMPUTER

PART 1.
THEORY of OPERATION

CONTROL DATA
CORPORATION

Customer Engineering
Instruction Manual

CONTROL DATA® 1604-A COMPUTER

PART 1.
THEORY of OPERATION

CONTROL DATA
CORPORATION

RECORD of REVISIONS

REVISION	NOTES
A	Publication Change Order 14546, no Product Designation
(8-29-66)	change. Pages 2-3, 4-25, 6-1, 6-2, 6-4 and 6-5 revised.

Pub. No. 60118700
September, 1966

© 1966, Control Data Corporation
Printed in the United States of America

Address comments concerning this manual to:

Control Data Corporation
Technical Publications Department
4201 North Lexington Avenue
St. Paul, Minnesota 55112

or use Comment Sheet located in back of this manual.

CONTENTS

Chapter 1. Introduction

Description 1-1
Input/Output Section 1-4
Arithmetic Section 1-4
Storage Section 1-4
Control Section 1-4

Chapter 2. Control Section

Introduction 2-1
Instruction Format 2-2
Operation Code 2-2
Translation of Operation Code 2-3
Designator 2-4
Translation of the Designator 2-6
Base Execution Address 2-6
Program Control Register 2-7
The U_2 Accumulator 2-7
Index Registers 2-9
Address Buffer Register 2-9
Functions of the R Register 2-10
Counting and Complementing In R 2-10
Program Address Register 2-13
Counting in P 2-13
Parallel Transmission Inverter Ranks 2-14
Control Sequences 2-15
Relation of Control and Storage Sequences 2-18
Read Next Instruction Sequence 2-20
Acquisition of New Instruction Word 2-20
Start and Stop 2-20
Initial Start 2-20
Step 2-23
Stop Instruction 2-23
Breakpoint Stop 2-23
Fault Stop 2-23
Start from Stop 2-23
Preliminary Steps in Address Modification 2-24
Preliminary Control and Arithmetic Steps 2-24
Interrupt Termination 2-24
Indirect Addressing 2-26
Normal Jump Sequence 2-27
Zero Address Sequence 2-29
Read Operand Sequence 2-29
Write Operand Sequence 2-30
Search and Transfer Sequence 2-31
Iterative Sequence 2-32
External Function Sequence 2-33
Interrupt Sequence 2-34
Interrupt 2-34
Advance Clock 2-34
Buffer Control Section 2-35
Auxiliary Sequence 2-35
High Speed Storage Sequence 2-36
Control Registers, Common Control Register and Comparator 2-36
Inverter Ranks I^7 and I^8 2-36
Auxiliary Scanner 2-36
Static Control 2-38

Breakpoint Address 2-38
Console Display 2-38
Resynchronizing 2-40
Resynchronizing Circuit 2-40
Runt Pulses 2-41
Resynchronizing Counter and Pulses 2-41
Jump Instructions 2-42
Normal Jump 2-42
Return Jump 2-43
Program Example 2-44
Skip Instructions 2-44
Computer Operating Controls 2-45
Start-Step Switch 2-45
Clear Switch 2-46
Computer Master Clear 2-46
External Master Clear 2-48
Selective Jump Switches 2-48
Selective Stop Switches 2-49
Storage Test Switches 2-49
Master Clock 2-50

Chapter 3. Arithmetic Section

Arithmetic Registers 3-1
Accumulator 3-1
Q Register 3-4
X Register 3-4
Registers Used in Arithmetic Operations 3-4
Basic Operations 3-5
Binary Arithmetic 3-5
Addition 3-7
Sensing for First Condition 3-7
Sensing for Second Condition 3-7
Combining Two Conditions 3-8
Borrow Pyramid 3-9
Subtraction 3-9
Shifting 3-10
Shift Instructions 3-11
Shifting in Multiplication and Division 3-13
Shifting in Floating Point Instructions 3-13
Shifting in Scale Instructions 3-13
Iterative Sequence 3-13
Multiplication 3-13
Division 3-18
Initial Sign Correction 3-18
Division Phase 3-19
Final Sign Correction 3-20
Floating Point 3-21
Coefficient 3-21
Exponent 3-21
Floating Point Operation 3-23
Addition 3-24
Subtraction 3-25
Multiplication 3-26
Division 3-27
Logical Product 3-27
Register Sensing Circuits 3-29
Arithmetic Faults 3-31
Divide Faults 3-31
Shift Faults 3-31
Overflow Faults 3-33
Exponent Faults 3-33
Fault Control 3-33

Chapter 4. Storage Section

Introduction 4-1
Principles of Magnetic Core Storage 4-3
Magnetic Cores 4-5
Memory Planes 4-7

Memory Plane Assembly 4-8
Address Selection 4-10
S Register 4-12
Horizontal and Vertical Drivers 4-12
Horizontal and Vertical Diverters 4-16
Memory Place Circuits 4-17
Sensing Circuit 4-17
Inhibit Circuits 4-20
Storage Sequence Control 4-21
Initiate Storage Reference Circuit 4-23
Sequence Controls 4-23
Timing Pulse Generator 4-23
Drive Generators and Pulse Generators 4-27
Fault Detector 4-31
Electronic Theory of Storage Circuits 4-31
Drive Generator (Type 51) 4-32
Diverter (Type 52) 4-33
Selector (Type 53) 4-34
Current Source (Type 54) 4-35
Inhibit Generator (Type 55) 4-36
Sense Amplifier (Type 56) 4-37

Chapter 5. Input/Output Section

Data Transmission 5-1
Buffering 5-1
Communication Paths 5-1
Data and Control Information Lines 5-2
Buffer Control 5-5
Buffer Operations 5-6
Store Terminal Address 5-8
Sensing for Channel Inactive 5-8
Selecting Equipment and Mode 5-12
Sensing Condition of Equipment 5-12
Activating the Channel 5-13
High Speed Storage Sequence 5-13
Ready/Resume Logic (Output) 5-14
Auxiliary Scanner 5-14
Auxiliary Sequence 5-17
Ready/Resume Logic (Input) 5-18
Buffering to or From 00001 - 00006 5-19
Transfer 5-20
Preparation 5-21
Input Transfer - 62 5-21
Interrupt or Clock Request During 62 5-22
Output Transfer - 63 5-22
Interrupt on Clock Request During 63 5-24

Chapter 6. Interrupt and Real Time Clock

Interrupt 6-1
Types of Interrupts 6-1
Internal Interrupts 6-2
External Interrupts 6-3
Interrupt Scanner 6-4
Interrupt Sequence Entrance 6-7
Interrupt Entrance from Exit 6-7
Internal Interrupt During RNI 6-7
Interrupt During Output Transfer 6-7
Interrupt During Input Transfer 6-7
Interrupt During Storage Search 6-7
Interrupt Sequence 6-11
Interrupt Subroutine 6-12
Return to Main Program 6-13
Real Time Clock 6-14

Chapter 7. Console Input/Output Equipment

Modes for Handling Data 7-1

External Functions 7-2
Select Codes 7-2
Sense Codes 7-4
Input Distributor 7-5
Output Distributor 7-7
Paper Tape Reader 7-9
Manual Controls 7-9
Paper Tape 7-9
Reader Operation 7-10
Auto Load Control 7-13
Paper Tape Punch 7-14
Punch Controls 7-14
Punch Operation 7-14
Typewriter 7-15
Typewriter Codes 7-15
Keyboard Operation 7-17
Typewriter Output Operation 7-20

Chapter 8. Power and Cooling

Power Requirements 8-1
Main Power Distribution 8-1
Cooling System 8-3
Protective Interlock System 8-3

Appendixes

A Borrow Pyramid A-1
B Glossary of Terms B-1
C List of Instructions C-1

FIGURES

1-1 Typical 1604-A System 1-2
1-2 Simplified Diagram of the Computer 1-3
1-3 Composition of 48-bit Instruction Word 1-5
2-1 Block Diagram of Computer Control 2-1
2-2 Structure of Operation Code Translator 2-3
2-3 Translator, Second Level 2-4
2-4 U Register 2-7
2-5 Adding in U_2 Accumulator 2-8
2-6 R Register Counting Structure 2-11
2-7 Parallel Transmission Paths 2-14
2-8 Example of a Sequence 2-17
2-9 Over-all Sequence Control 2-18
2-10 Relation of Control and Storage Sequences 2-19
2-11 Form of RNI for Acquiring Instructions 2-21
2-12 RNI For Start and Stop 2-22
2-13 RNI For Interrupt Termination 2-25
2-14 Indirect Addressing Part of RNI 2-28
2-15 Normal Jump Sequence 2-28
2-16 Basic Chain of Control Delays in Zero Address Sequence 2-29
2-17 Basic Chain of Control Delays in RO Sequence 2-30
2-20 Iterative Sequence 2-32
2-21 Block Diagram of EXF Sequence 2-33
2-22 Block Diagram of Interrupt Sequence 2-34
2-23 Block Diagram of Auxiliary Sequence 2-35
2-24 Pictorial Representation of Scanner 2-37
2-25 Digit Display, Lowest Octal Digit of A 2-39
2-26 Console Display 2-39
2-27 Resynchronizing Circuit 2-40
2-28 Resynchronizing Counter 2-42
2-29 Connection of Start-Step Switch to RNI 2-47
2-30 Sampling Selective Jump Conditions 2-48

2-31 Master Clock Oscillator Waveforms 2-51

3-1 Over-All Block Diagram of Arithmetic Section 3-2

3-2 Typical State of A Register 3-3

3-3 Relation of Borrow Pyramid to A and X Registers 3-10

3-4 Shift Control 3-12

3-5 Iterative Sequence 3-14

3-6 Multiply Step 3-17

3-7 Divide Step 3-19

3-8 Relation of Coefficients in Addition 3-25

3-9 Register Sensing Networks 3-30

3-10 Arithmetic Faults 3-32

4-1 Logical Divisions of the Storage Section 4-2

4-2 Magnetic Core Matrix 4-4

4-3 Memory Plane Stack 4-5

4-4 Hysteresis Diagram 4-6

4-5 Voltage on Sense Winding as a Result of Read Drive 4-7

4-6 Memory Board 4-7

4-7 Distribution of Memory Plane Assemblies 4-8

4-8 Intersection of H and V Wires 4-9

4-9 Connection of Drive Lines and Diversion Lines 4-11

4-10 Drivers and Diverters Selected by S^1 or S^2 Registers 4-13

4-11 S^1 and S^2 Registers 4-14

4-12 Typical Horizontal or Vertical Drive Circuit 4-15

4-13 Horizontal or Vertical Diversion Circuit 4-16

4-14 Path of Sense Wire Through a Four-Core Matrix 4-18

4-15 Typical Stage of the Memory Plane Control 4-19

4-16 Path of Inhibit Wire Through a Four-Core Matrix 4-20

4-17 Basic Pulse Sequence for Storage Reference 4-21

4-18 Storage Reference Circuit 4-22

4-19 Sequence of Pulses Generated by Control Sequence 4-24

4-20 Timing Pulse Generator of Even Storage Sequence Control 4-25

4-21 Drive and Pulse Generator of Even Storage Sequence Control 4-30

4-22 Fault Detector of Even Storage Unit 4-31

4-23 Drive Generator (Card Type 51) 4-32

4-24 Diverter (Card Type 52) 4-33

4-25 Selector (Card Type 53) 4-34

4-26 Current Source (Card Type 54) 4-35

4-27 Inhibit Generator (Card Type 55) 4-36

4-28 Sense Amplifier (Card Type 56) 4-37

5-1 Block Diagram of Buffer Control 5-7

5-2 Structure of Control Words 5-8

5-3 External Function Select/ Sense Code 5-9

5-4 External Function Sequence 5-11

5-5 External Function Inverters 5-13

5-6 Ready/Resume Logic - Channel 2 5-15

5-7 Auxiliary Scanner 5-16

5-8 Ready/Resume Logic - Channel 1 5-18

5-9 Block Diagram of Search and Transfer Sequence 5-20

5-10 Temporary Termination of S & T Sequence During 63 5-24

6-1 Fault FFs and Internal Interrupt Request 6-2

6-2 Masked Interrupt Register and Interrupt Scanner 6-5
6-3 Interrupt Sequence 6-7
6-4 Real Time Clock 6-14
6-5 Setting Advance Clock FF 6-15
7-1 External Function Translator 7-3
7-2 Sense Circuit Console Equipment 7-5
7-3 Input Distributor, Simplified Diagram 7-6
7-4 Character Orientation in Registers and on Paper Tape 7-7
7-5 Output Distributor Simplified Diagram 7-8
7-6 Seven Level Punched Paper Tape 7-10
7-7 Reader Control Circuit 7-11
7-8 End-of-Tape Circuit 7-13
7-9 Keyboard Control Circuit 7-18
7-10 Carriage Return, Interrupt Circuit 7-19
7-11 Typewriter Control Circuit (Output) 7-21

TABLES

2-1 Designation of Base Execution Address 2-6
2-2 Control Sequences 2-15
3-1 Decimal and Binary Equivalents 3-5
3-2 Binary Subtraction 3-8
4-1 Timing Pulse Generator of Even Storage Sequence Control: Complete Counter Cycle 4-26
4-2 Timing Pulse Generator, Source of Timing Pulses 4-29
5-1 Line Assignment of a Given Cable Group - Data Lines 5-3
5-2 Line Assignment of a Given Cable Group - Control Lines 5-4
5-3 Sense Codes 5-9
6-1 Interrupt Locations 6-1
6-2 Internal Fault Codes 6-3
6-3 External Interrupt Codes 6-3
6-4 Typical Interrupt Subroutine 6-10
7-1 Select Codes for Console Equipment 7-4
7-2 Sense Codes for Console Equipment 7-4
7-3 Reader Operation 7-12
7-4 Typewriter Codes 7-16

CHAPTER 1
INTRODUCTION
DESCRIPTION

This manual describes the operation of the central computer and console. Chapter 1 is an introduction to the computer. Chapters 2 - 5 analyze in detail the major sections of the computer: control, arithmetic, storage, and input/output. Chapter 6 describes the interrupt and real time clock operations. Chapters 7 - 8 describe the console-mounted external equipment and the power system.

Supplementary material:

1604-A Customer Engineering Diagrams Manual CDC 60024300
1604-A Input/Output Specifications CDC 60024400
1604-A Reference Manual CDC 60024500
1604-A Customer Engineering Instruction Manual, Part 2 Maintenance CDC 60118800

The 1604-A computer consists of the main computer cabinet and the console. A 1604-A system (figure 1-1) may include several external equipments which are connected to the computer by means of the adaptors. The adaptors provide data buffers and control circuits for card reader and punch, line printer and tape units.

The main cabinet contains the computer and the control circuits for the external equipment at the console. The operator's panel on the console provides indicators and operating controls. The monitor typewriter, paper tape punch and paper tape reader are on the console.

There are four major sections in the computer (figure 1-2): (1) Input/output provides communication between the computer and external equipments; (2) Arithmetic performs the arithmetic and logical operations to execute instructions; (3) Storage provides internal storage for data and instructions; and (4) Control coordinates and sequences all the operations which carry out the execution of an instruction.

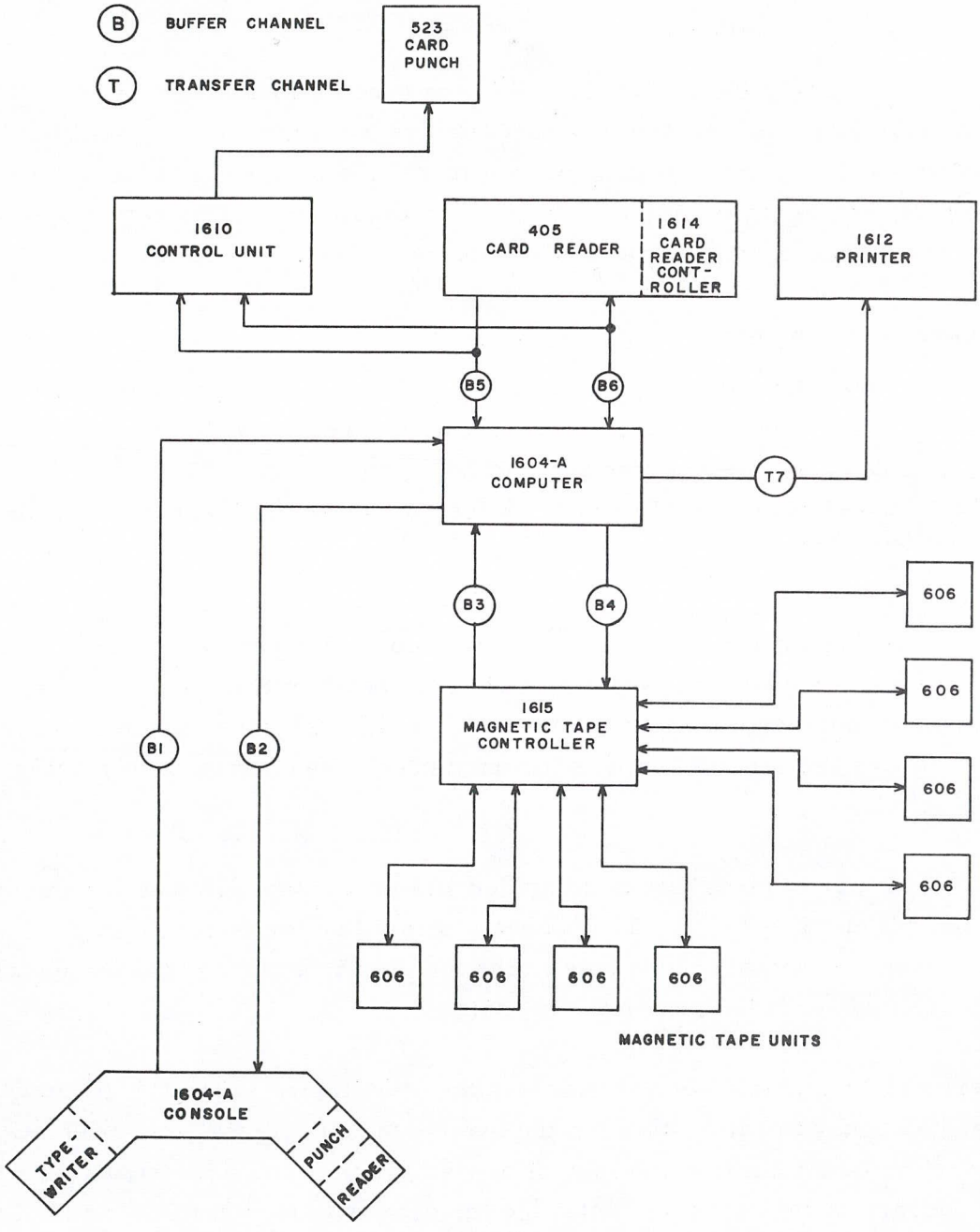


Figure 1-1. Typical 1604-A System

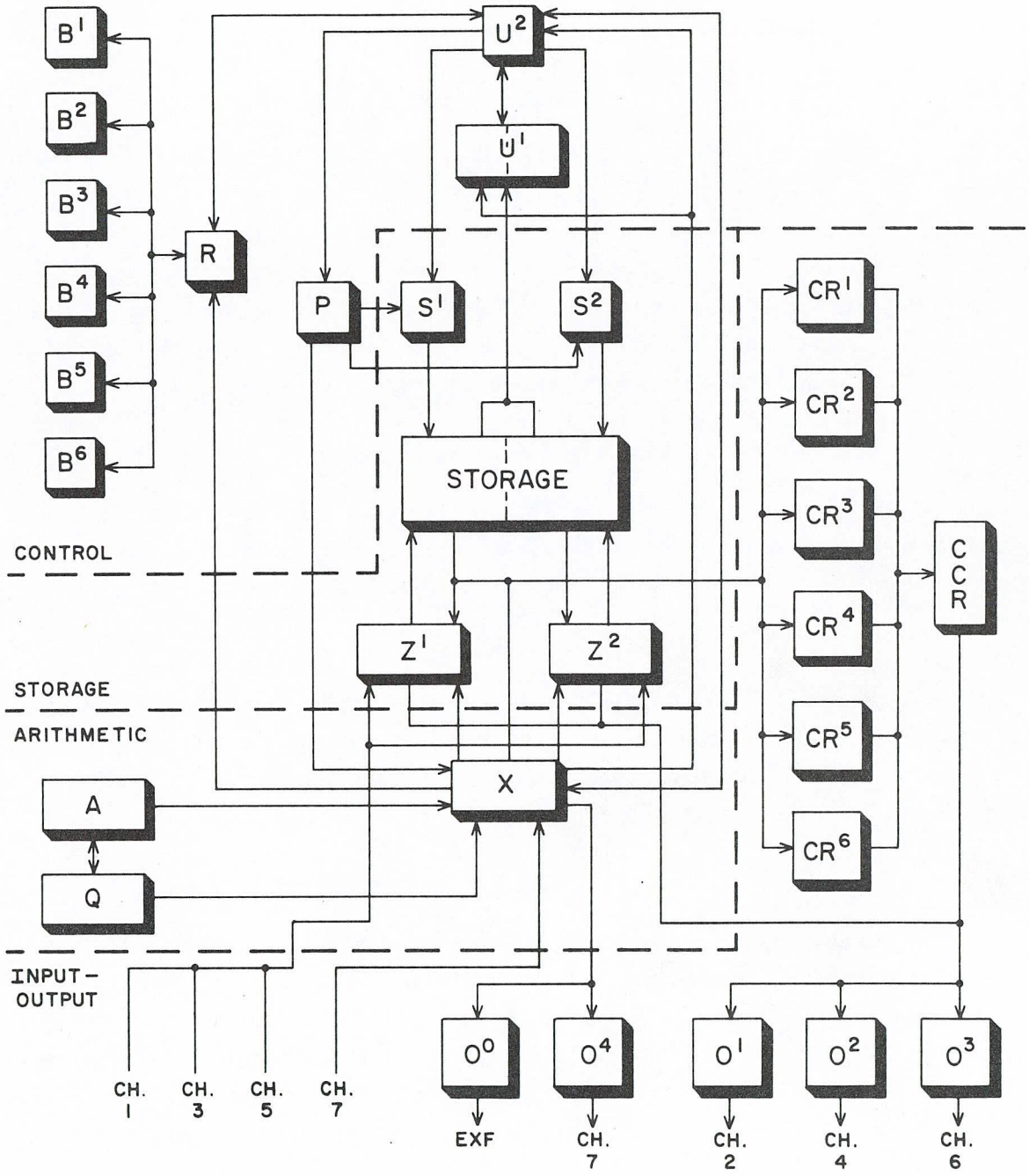


Figure 1-2. Simplified Diagram of the Computer

INPUT/OUTPUT SECTION

The Input/Output section of the computer provides the methods for data exchange and for proper control of information transmission between the computer and the various external equipments.

ARITHMETIC SECTION

The A register (Accumulator) is the principal arithmetic register. This register provides for the parallel addition of the X register to its content; it can be shifted either separately or in conjunction with the Q register.

The Q register is an auxiliary arithmetic register used in more complicated arithmetic operations. In combination with the X register it forms logical products and can be shifted either separately or in conjunction with A.

STORAGE SECTION

Each of the two magnetic core storage units contain 16, 384 locations or addresses for 48-bit words; the total storage capacity is 32, 768 words. All odd storage addresses reference one storage unit; all even addresses, the other.

Words to be read out of storage are transferred via inverter rank I^5 (even storage) or I^6 (odd storage) to the appropriate register. Words to be written into a storage unit are transmitted via I^5 or I^6 to the Z register, which is then sampled to determine what is to be entered into storage.

CONTROL SECTION

The control section acquires an instruction from storage, interprets it, and sends the required commands to other sections. A 24-bit instruction is composed of three parts or codes, designated by the letters f, b and m (figure 1-3). A program word is a pair of 24-bit instructions, which together occupy one storage location. The higher-order 24 bits of such a word are called the upper instruction and the remaining 24 bits, the lower instruction.

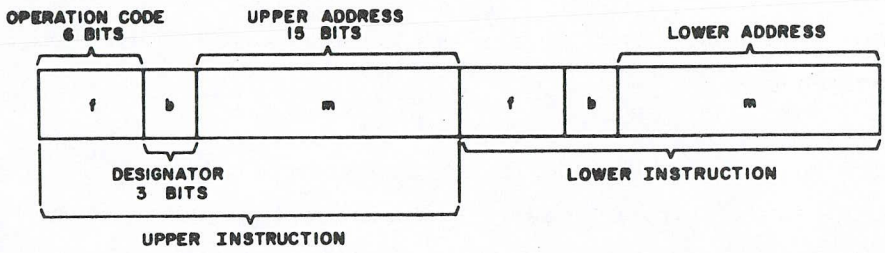


Figure 1-3. Composition of 48-Bit Instruction Word

The six index registers (B^1 through B^6) provide for modification of the execution addresses of instruction.

The P register, program address register, provides continuity by generating in sequence the storage addresses in which the individual steps of the program are contained.

The program control register, U^1, holds a program word while the two instructions contained in it are executed. The 48-bit instruction word is taken from the storage location specified by P and entered into U^1. The upper instruction is always executed first. Execution of the lower instruction follows, except when the upper instruction is a jump or conditional skip.

The auxiliary program control register, U^2, is an accumulator used primarily in the modification of the base execution address.

The address buffer register, R, is used in transmissions to and from the B registers and is also used as a counter during the execution of several instructions.

The following is an example of how this typically operates:
Address 00500 contains the following pair of instructions: upper instruction 14 (Add) and lower instruction 50 (Enter Index). The P register holds the address 00500 (even storage). The upper 14 bits of P are sent to S^1. The storage reference is initiated and the 48-bit word (instruction pair) is read from address 00500. The word is entered into U^1.

From this point on the operations of the computer are conditioned directly or indirectly by the 24-bit instruction in the upper half of U^1, Add. The purpose of this

instruction is to add the quantity in the storage location specified by the execution address to the contents of A.

The index code, b, and operation code, f, are now translated and (B^b)* is transmitted to R^1. At the same time the 15-bit base execution address is transmitted from U^1 to U^2. The contents of R^1 are added to U^2 to yield the execution address, M, which specifies the location of the operand.

Depending on whether the lowest bit of U^2 is a "0" or a "1", the remaining 14 bits are transmitted to S^1 or S^2 in preparation for reading the operand to be added to A. The storage reference is initiated, the operand is entered into X, and X is added to A.

The computer is ready to execute the next instruction contained in the lower 24 bits of U^1. The upper half of U^1 is cleared and the lower half transmitted to the upper half, so that U^1 upper holds instruction 50 (Enter Index). The purpose of this instruction is to place the 15-bit base execution address in index register b. Following translation of f and b, the base execution address is transmitted from U^1 to U^2. Now U^2 is transmitted to R, and R in turn is transmitted to the B register specified by b and the execution of the instruction is complete.

*(B^b) = the contents of the B register specified by the b portion of the instruction.

CHAPTER 2
CONTROL SECTION
INTRODUCTION

The control section of the computer is composed of two parts, main control and buffer control.

Main control directs the interpretation and execution of instructions and establishes the timing to perform the instructions in the proper sequence. When an exchange of data with external equipment is called for, main control initiates the exchange and then gives control of the exchange to buffer control. Main control is then free to continue executing other instructions while data exchange is taking place.

Buffer control accepts control of data exchange operations, supervises the exchange, and signals main control when the operation is completed. Buffer control is described lated in this chapter and in chapter 5.

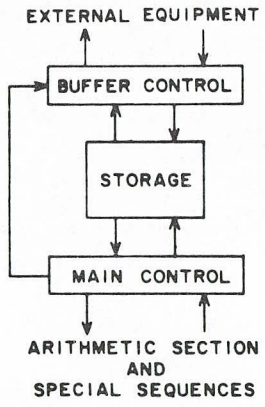
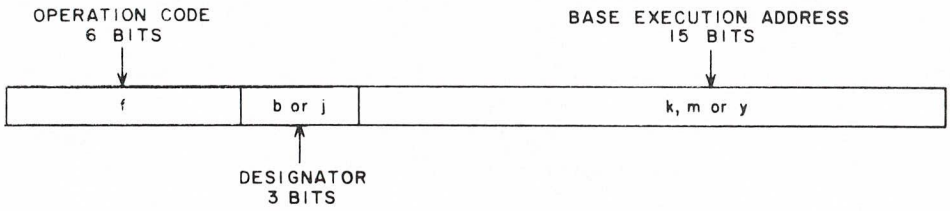


Figure 2-1. Block Diagram of Computer Control

INSTRUCTION FORMAT

The commands which execute instructions and exchange data between the computer and external equipment are issued by a sequence. The issuing of commands is controlled directly or indirectly by the instruction in the Program Control register, U^1. The 24-bit instruction format is shown below.



Each of the 62 instructions has a unique 6-bit operation code, f, which designates the instruction. The translation of f establishes the condition required within the control section for the execution of the instruction.

The next three bits are the designator, b or j. When the designator refers to an Index register, B, it is denoted by b. The contents of the designated Index register, (B^b), are usually added to the base execution address, the right 15 bits of the instruction. The base execution address usually refers to a memory location, m. (B^b) is added to m to give the actual address, M, of the operand. When b = 0, m is the operand address. When b = 7, indirect addressing is used. Indirect addressing is explained on page 2-26.

In Jump and Stop instructions the designator is called j and indicates the condition necessary for the jump or stop.

OPERATION CODE

This 6-bit code specifies an instruction and controls the operation of the computer during the execution of the instruction. There are 64_{10} Octal codes; 62 Specify instructions, and codes 00 and 77 represent fault conditions that halt the computer.

Translation of Operation Code

Prior to the execution of the instruction designated by the value of f, the operation code is translated by a network of single inverters which samples the upper six bits of U^1. The results of the translation go to the various sections of the machine to gate commands which carry out the required operations.

The translator (figure 2-2) uses several levels of logic in forming the outputs which gate commands. Outputs from the upper six FFs of U^1 are connected to single inverter slaves. The set side of each FF has a U4-4 and U4-6 inverter slave; the clear side of each FF has a U4-5 and a U4-7 inverter slaves.

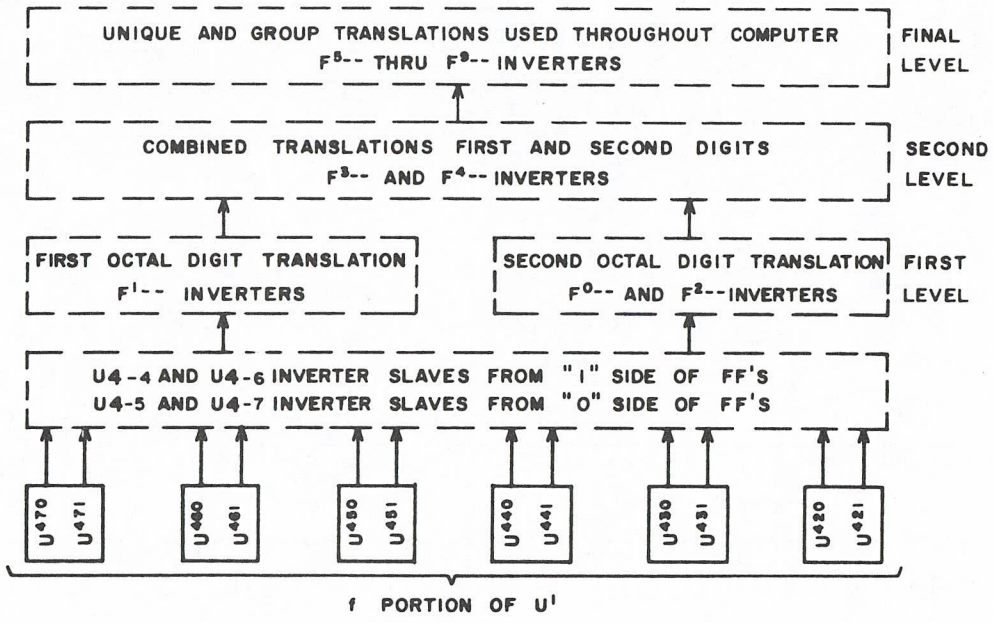


Figure 2-2. Structure of Operation Code Translator

The first level translation is divided into two parts; i.e., one concerned with the second octal digit of f (U42-, U43-, U44-), and the other with the first octal digit (U45-, U46-, U47-). All F0-- and F2-- inverters translate the second octal digit. All F1-- inverters translate the first octal digit.

Unique (single-value) translations of the second octal digit are provided by F000 through F007. When the output of one of these inverters is "1", the second octal digit has the value given in the third superscript digit of the inverter designation; e.g., a "1" from F007 indicates that the second digit is 7. The F2-- inverters provide partial, or incomplete, translations of the second octal digit. These translations are duplicated by several slave inverters.

The F1-- inverters, which uniquely translate the first octal digit, use the last digit of the inverter designation to indicate the value translated. Thus, F105 indicates that the first octal digit is 5.

In the second level of the translator the outputs of first level inverters that translate individually the first and the second octal digits are combined to specify either a unique value of the Operation code or a group of values. The F3-- and F4-- inverters combine the translations of the first and second digits in figure 2-3. Outputs of the F3-- and F4-- inverters go to F5-- or F6-- which supply the translations throughout the computer.

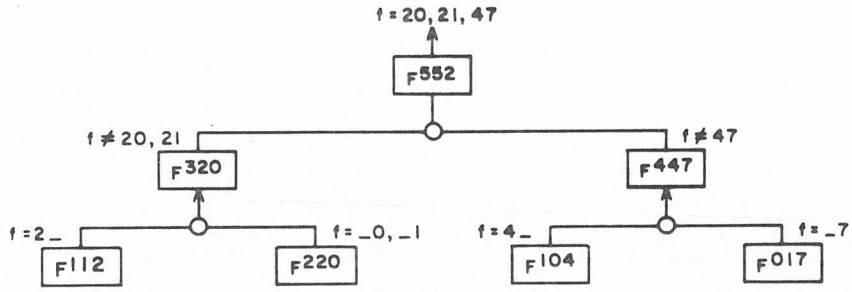


Figure 2-3. Translator, Second Level

DESIGNATOR

When the 3-bit designator in an instruction specifies the Index register, B, it is denoted by the letter b. When b = 0, no modification of the execution address occurs; with b = 7, indirect addressing is used. Instruction 50 with b = 0 performs no operation but "passes" to the next instruction. In instructions 62-67, b = 0 specifies that exactly one word is to be transferred or searched.

When used as a condition designator, as in instructions 22, 23, and 74-76, the 3-bit quantity is denoted by the letter j which is interpreted as follows:

22 A Jump:
23 Q Jump:

0 - Jump if register content is zero
1 - Jump if register content is not zero
2 - Jump if register content is positive
3 - Jump if register content is negative
4 - Return jump if register content is zero
5 - Return jump if register content is not zero
6 - Return jump if register content is positive
7 - Return jump if register content is negative

75 Selective Jump:

0 - Jump unconditionally
1 - Jump if jump key one is set
2 - Jump if jump key two is set
3 - Jump if jump key three is set
4 - Return jump unconditionally
5 - Return jump if jump key one is set
6 - Return jump if jump key two is set
7 - Return jump if jump key three is set

76 Selective Stop:

0 - Stop uncondtionally (normal jump)
1 - Stop if stop key one is set (normal jump)
2 - Stop if stop key two is set (normal jump)
3 - Stop if stop key three is set (normal jump)
4 - Stop unconditionally (return jump)
5 - Stop if stop key one is set (return jump)
6 - Stop if stop key two is set (return jump)
7 - Stop if stop key three is set (return jump)

Only the stop may be conditioned; the jump occurs unconditionally when the computer is restarted or when the stop condition is not met.

74 External Function:

0 - Select external equipment
1 - Activate communication channel one
2 - Activate communication channel two

74 External Function:	3 - Activate communication channel three
	4 - Activate communication channel four
	5 - Activate communication channel five
	6 - Activate communication channel six
	7 - Sense external condition

Translation of the Designator

Inverters F700 through F707 translate the eight possible values of the designator. A "0" output from one of these inverters indicates the value given by the third superscript digit in the symbol; a "0" output from F705 indicates that the value is 5. Since the outputs of these inverters feed inverter slaves, the designator translation is in normal form ("1") when it is used for gating.

BASE EXECUTION ADDRESS

The base execution address (lower 15 bits of an instruction) has three functions, depending upon the instruction used (see table 2-1).

TABLE 2-1. DESIGNATION OF BASE EXECUTION ADDRESS

Instructions	Use	Denoted		
12-33, 36-47, 52, 53, 55-73, 75, 76	Specifies storage location of operand	m	M	After modification by addition of (B^b)
14, 10, 11, 50 51, 54, 74	As operand	y	Y	
01-03, 05-07, 34, 35	As shift count	k	K	

The base execution address is transmitted from the upper instruction in U^1 to U^2 in the execution of all instructions. If it is to be modified, B^b is transmitted to R^1 (the Address Buffer register) and then R^1 is added to U^2. When the operand is procured from or sent to storage, U^2 is transmitted to S^1 or S^2. When U^2 is to be used as the operand or shift count, it is transmitted to X^1 or R^1.

PROGRAM CONTROL REGISTER

The Program Control register, U, holds the 48-bit instruction word during execution of the two 24-bit instructions (figure 2-4). All operations necessary to execute an instruction are governed by the contents of this register.

The U register consists of two ranks of FFs. Rank U^1, 48-bits in length, stores the instruction word during execution of the two instructions. Rank U^2, 15-bits in length, is a subtractive accumulator with a borrow pyramid. Transmission paths connect U^2 with the m portion of the upper half of U^1.

The instruction word is read from storage and entered into U^1. Execution of the upper instruction occurs first; the lower instruction is then transmitted to the upper half of U^1 and executed. Thus the current instruction is always in the upper half of U^1.

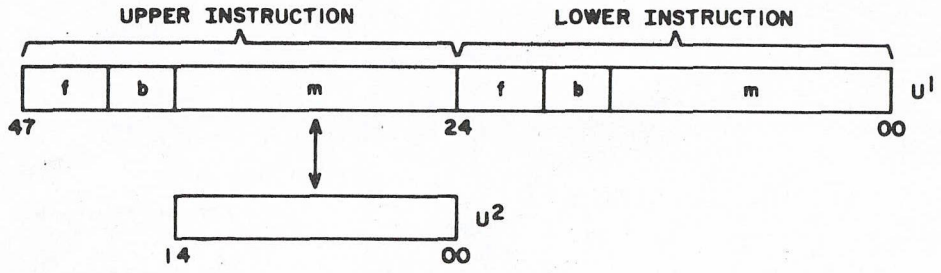


Figure 2-4. U Register

The primary function of U^2 is to modify the execution address, m, of the instruction in the upper half of U^1 by adding the contents of B^b to it.

After transmitting m from U^1 to U^2 and transmitting (B^b) to R, R is added to U^2 and the sum is formed in U^2. The modified execution address in U^2 for most instructions specifies the location of the operand in storage.

THE U^2 ACCUMULATOR

U^2 is a 15-bit subtractive accumulator that provides for the addition of R^1 to its content. This accumulator is simular in structure to the A register (the 48-bit accumulator).

Two ranks of FFs are necessary for the addition operation. Stages U^{24} through U^{38} of U^1 (upper address) form the rank that is sampled by the pyramid (figure 2-5). The U^2 FFs constitute the other rank, which receives the sum.

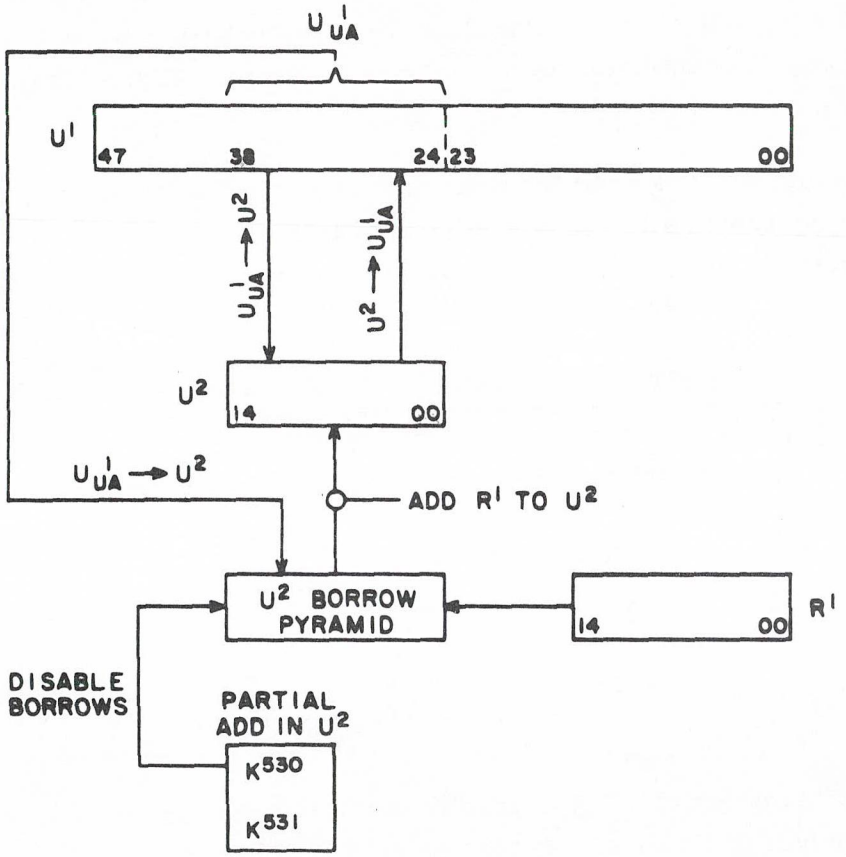


Figure 2-5. Adding in U^2 Accumulator

Prior to the addition of R^1 to U^2, the $U^1_{UA} \rightarrow U^2$ or $U^2 \rightarrow U^1_{UA}$ command insures that both ranks hold the same quantity. Following this, the borrow pyramid samples U^1_{UA} and R^1 to determine the stages of U^2 that must be toggled in order for $U^1_{UA} + R^1$ to be formed in U^2. This occurs when the command Add R^1 to U^2 is given.

The U^2 accumulator has provisions for disabling the borrow pyramid so that each bit of U^2 is toggled if the corresponding bit of R^1 is "1". The pyramid is disabled when the Partial Add in U^2 FF (K530/531) is set. Thus the Add R^1 to U^2 command accomplishes a selective toggling function when K530/531 is set. However, when this FF is cleared, a full addition results from the Add R^1 to U^2 command. The partial addition of R^1 to U^2 is often used in transmitting B to another register such as X. B goes to R and R^1 is partially added to U^2, with U^2 sent to X.

In addition, if either B^b or U^2 has a "1" in the highest bit, it is treated as a negative number expressed in one's complement form. Thus if the quantities m = 00005 and B^b = 77776 are added, the sum is 00004. The same result is obtained when the values of B^b and U^2 are interchanged.

INDEX REGISTERS

There are six Index registers, B^1 through B^6. In most instructions the B registers hold quantities to be added to the base execution address. For Search instructions (64-67), (B^b) indicates the number of items to be searched. The B registers have no provision for arithmetic operations. When such an operation is required on an index quantity, (B^b) is entered in R or U^2 and the operation is performed there; the result is returned to B^b.

Transmissions into a B register come from R^2 via the I^4 single-inverter rank (figure 2-7). Each B register provides outputs to either I^2 or I^3, which are in turn gated to R^1. For this transmission the "0" output of the register FFs is used due to the single inversion of I^2 or I^3.

Equation symbol assignments for the B registers are different from other registers. The first digit of the superscript identifies the register; second and third digits indicate the stage. The "0" side is not identified by the odd character of the third digit; instead, the "0" side is indicated if the second and third digits are 50 or greater. The stage with which such a symbol is associated is found by subtracting 50 from the last two digits. For example, B563 is the "0" side of stage 13 of B^5.

ADDRESS BUFFER REGISTER

The 15-bit Address Buffer register R provides for counting and complementing as well as for storage. As a counter it operates subtractively.

The R register consists of two ranks of FFs, R^1 and R^2. Inverter rank I^2 receives inputs from control which set it to predetermined values; I^2 is then transmitted to R^1 (figure 2-6).

FUNCTIONS OF THE R REGISTER

1) All transmissions to and from the B registers go through R.

2) The base execution address of the current instruction is modified by addition of the quantity in R (obtained from B^b) to U^2.

3) R acts as the shift counter in Shift instructions.

4) In the integer and fractional Multiply and Divide instructions, R records the number of partial multiplications and divisions which remain to be performed.

5) In floating point instructions, R performs arithmetic operations on one of the two exponents.

COUNTING AND COMPLEMENTING IN R

The R register is a two's complement, open-ended subtractive counter with a modulus of 2^{15}. In counting, the bits of R are toggled to form the quantity (R^1) -1 in R^2. The command Reduce R^1 to R^2 subtracts "1" from the first stage, R^{00} (figure 2-6). When R^{00} is "0", a borrow from R^{01} is required. Similarly, a borrow is required from R^{02} if both R^{00} and R^{01} are "0". Thus, in general, a borrow is required from stage n if all stages lower than n are "0". Borrows are accomplished by toggling a stage of R^2 with the corresponding stage of R^1 when the command occurs.

For sensing the borrows, R^1 and R^2 are organized as five 3-bit groups, each of which is a small counter. Just as "1" is borrowed from R^{01} only if R^{00} is "0", "1" is borrowed from the second group only if the stages of the first group are each "0". A borrow is made from a group only when all stages of the next lower group are "0".

In each group, H87- senses when the three stages of the group are "0". This condition is indicated by a "0" output from H87-. Outputs of H87- of lower-order groups are used by higher-order groups to determine whether a borrow is required. A borrow is required from the fifth group only when the outputs of H870, H872, H874, and H876 are "0".

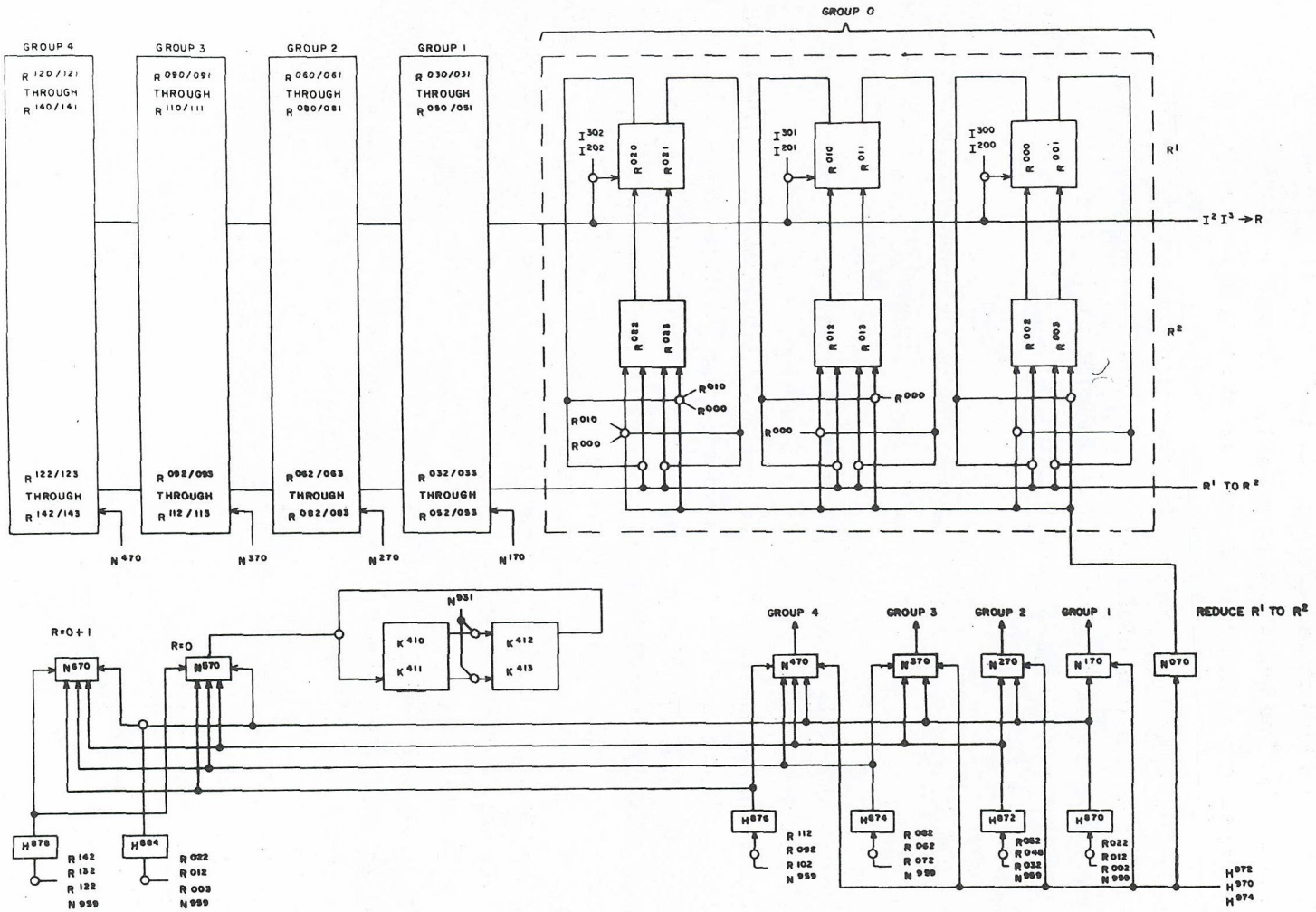


Figure 2-6. R Register Counting Structure

Within a group from which a borrow is required, the first stage is toggled by the Reduce R^1 to R^2 command. The second stage is toggled if the first stage is "0". The third stage is toggled only if both the first and second stages are "0".

Complementing is performed in R by transmitting the complement of the bits in R^2 to R^1. For many uses of R it is necessary to sense when the quantity in R is "0"; the H87- terms sense this condition. The outputs of all five H87- terms are combined by both N570 and N670. When the five inputs to each are "0", all 15 stages of R contain "0". This condition is indicated by a "1" from N570 or N670.

There is an important difference, however, between the indications of the R = 0 condition given by N570 and N670. To account for this difference it is necessary to consider the interval between the time when the Reduce R^1 to R^2 command is given and the time when the state of R resulting from this reduction is reflected at the output of N570. The interval is two clock periods, one for toggling R^2 and one for the R = 0 control delay. In certain cases, such as Shift instructions, sensing the R = 0 condition must not lag the Reduce command by such an interval. Furthermore, since in these cases it is known that R is reduced to "0", it is possible to anticipate the time at which this occurs by sensing when R = 1. The output of N670 is a "1" not only when R = 0 but also when R = 1.

By using FFs K410/411 and K412/413, it is possible to obtain a pulse two clock times after R is reduced to "0" and to obtain just one such pulse despite the fact that when R is reduced to "0" the output of N570 may be a "1" for some time. This is done by setting K410/411 to "1", which in turn sets K412/413 to "1". After R = 0 the latter FF causes the former to be cleared. This results in K412/413 being cleared one clock time later. In order to uniquely specify the first time the output of N570 is a "1", it is only necessary to combine a "1" output from K412/413 in an AND with N570.

When R is employed as an additive rather than a subtractive counter, the sequence of commands is:

Complement R^2 to R^1
$R^1 \rightarrow R^2$
Reduce R^1 to R^2
Complement R^2 to R^1
$R^1 \rightarrow R^2$

Following this sequence R^2 holds the initial quantity plus one.

PROGRAM ADDRESS REGISTER

The Program Address register, P register, holds the address of each instruction word. After the upper and lower instructions of a word are executed, the quantity in P is advanced by one to the address of the next instruction. Thus the P register is a counter.

The initial address of a program is entered into P manually at the control console.

When the computer executes a Return Jump instruction, (P) + 1 is stored to provide for return to the next instruction in the program. The address specified in the Jump instruction is then entered into P from the U register.

The contents of P^1 are transmitted to P^2 every odd clock phase so that P^2 always equals P^1.

COUNTING IN P

The P register is a two's complement additive counter with a modulus of 2^{15}. The bits of P are toggled to form the quantity $P^2 + 1$ in P^1. The command to advance P occurs each time the Read Next Instruction (RNI) sequence is entered from a full exit. This command always adds one to the first stage, P^{00}. If P^{00} is a one when the command occurs, a carry to P^{01} is required. Similarly, a carry to P^{02} is required if both P^{00} and P^{01} are one. A carry to any stage, n, is required if all stages lower than n are "1's". Carries are accomplished by toggling a stage of P^1 with the corresponding stage of P^2.

P^1 and P^2 are organized as five three-bit groups, each of which is a small counter. Group 1 is increased for each Advance P command. Group 2 is increased only if all the bits of group one are ones when the Advance P command is received. Group 3 is increased only if both groups 1 and 2 are all ones, etc. Except for the fact that P is an additive counter, its counting structure is similar to that of the R register (figure 2-6).

Special gates provide for setting P to addresses 00007-00017 when an interrupt condition occurs. Storage locations 0007-000017 are used to enter into an exit from interrupt routines.

The P register and the setting of the Breakpoint switch are sampled and compared during every RNI sequence. The logic for this comparison is discussed under breakpoint.

PARALLEL TRANSMISSION INVERTER RANKS

Many of the parallel transmission paths between registers involve a rank of inverters. Figure 2-7 shows the major transmission paths and the nine inverter ranks. The purpose of inverter ranks is to increase the input or output capacity of a register. For example, rank I^4 as a slave for R increases the number of outputs from R to the Index registers. Each inverter of I^4 requires only one output from R, and the inverter provides six outputs to the Index registers.

In addition to increasing the number of transmission paths available, some inverter ranks perform other operations. I^2 has special logic connected to its input that is used to set R to the proper shift count for Iterative instructions.

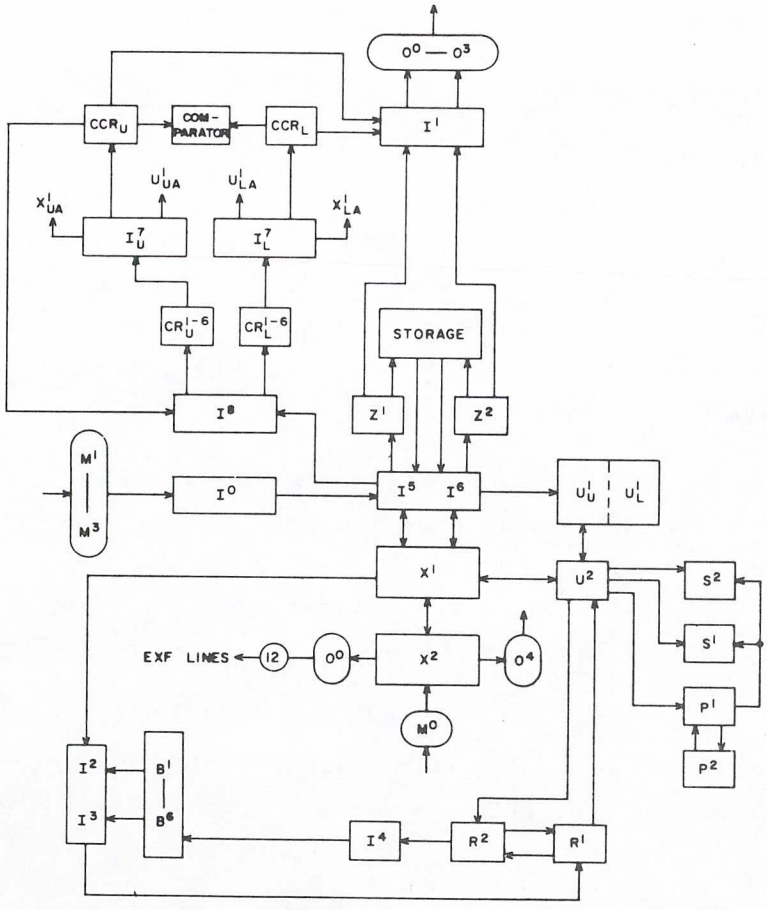


Figure 2-7. Parallel Transmission Paths

CONTROL SEQUENCES

The timing of commands within the computer is done by control sequences. The main sequences and the computer instructions which they control are listed in table 2-2. Command timing charts, contained in Part 2 of the Customer Engineering Instruction Manual, show the complete list of sequence commands for each instruction or operation.

TABLE 2-2. CONTROL SEQUENCES

SEQUENCE		SYMBOL	INSTRUCTION
Read Next Instruction	RNI	H09-	All instructions
Normal Jump	NJ	H10-	(22, 23, 75, 76) (j = 0-3)
Zero Address	ZA	H2--	01-11, 34, 35, 50, 51, 54, 55
Read Operand	RO	H3--	12-17, 36-46, 52, 53, 70-73, Advance Clock
Write Operand	WO	H4--	20, 21, 47, 56-61, (22, 23, 75, 76) (j = 4-7), Interrupt
Search and Transfer	S & T	H5--	62-67
Iterative	I	H6--	24-33
External Function	EXF	H7--	74
Auxiliary	Aux	H71-	Buffer
		H76-	
High Speed Storage	HSS	H17-	74.1-6, Buffer (addresses 00001-6)
		H18-	
Interrupt	INTER	H78-	Interrupt and Advance Clock

Instructions are executed by the Read Next Instruction (RNI) sequence and one other. For example, instruction 14 (Add), is executed by the RNI sequence followed by the Read Operand (RO) sequence. RNI enters the 24-bit instruction word in the upper half of U^1 and RO obtains the operand from storage and performs the addition.

Each sequence consists of a series of control delays. In the hypothetical sequence of figure 2-8 the basic series of control delays have H3-- and V3-- symbols.

The sequence is initiated when a single pulse from RNI is gated into H301 by F530. This pulse moves down the chain of control delays at the rate of one control delay per clock period (0.2 microsecond). H301 receives this pulse at time 0. The time scale shows relative time positions for later control delays.

There are two methods for generating commands. The first is illustrated by the command Add R^1 to U^2. This means of generating commands is fully clocked. A control delay with clocked output applies the Command signal at a definite time to the FFs of the register in use. After the sequence is initiated, the H846-V846 control delay is set at time one if the condition given by F541 is met. At time two, H953-N-53 is entered by the output of V846. At time three, the command reaches the FFs of U^2. By time four, the command is complete and the quantity available at the outputs of the U^2 register FFs.

It is pertinent at this point to explain why the Add R^1 to U^2 command is, in this case, generated by the use of H846-V846. The command could be generated and occur at time three by providing an input to H953 from V302, instead of taking an output from V301 and going through H846-V846 to H953. In fact, this is done in some of the several instances for which the Add R^1 to U^2 command is generated. However, the total number of instances in which this command is generated is greater than the maximum number of inputs possible for H953. Additional capacity of H953 is provided by H846. All control delays with H8-- symbols increase the input capacity of the H9-- control delays which bring the command to the register FFs. The H8-- control delays are called initiates, indicating they do not actually supply the command to the register.

The second method of generating commands is illustrated by the Clear A^1 command (figure 2-8). A Clear A^1 FF is set; the "1" output goes from single inverters J81- through W81- to the "0" side of A^1. When K580/581 is set to "1", W81- provides a "1" input to the "0" side of the A^1 FFs which clears A^1. The Clear A^1 FF is set at time one and cleared at time three. The transmissions from the output of the Clear A^1 FF through J81- and W81- to the A^1 FFs are not clocked. A^1 is cleared between time two (minimum) and time three (maximum).

Whether a full or half exit is taken in figure 2-8 is determined by J064 and J065, slaves of the Exit FF. If the current instruction is a lower instruction, a full exit is taken and

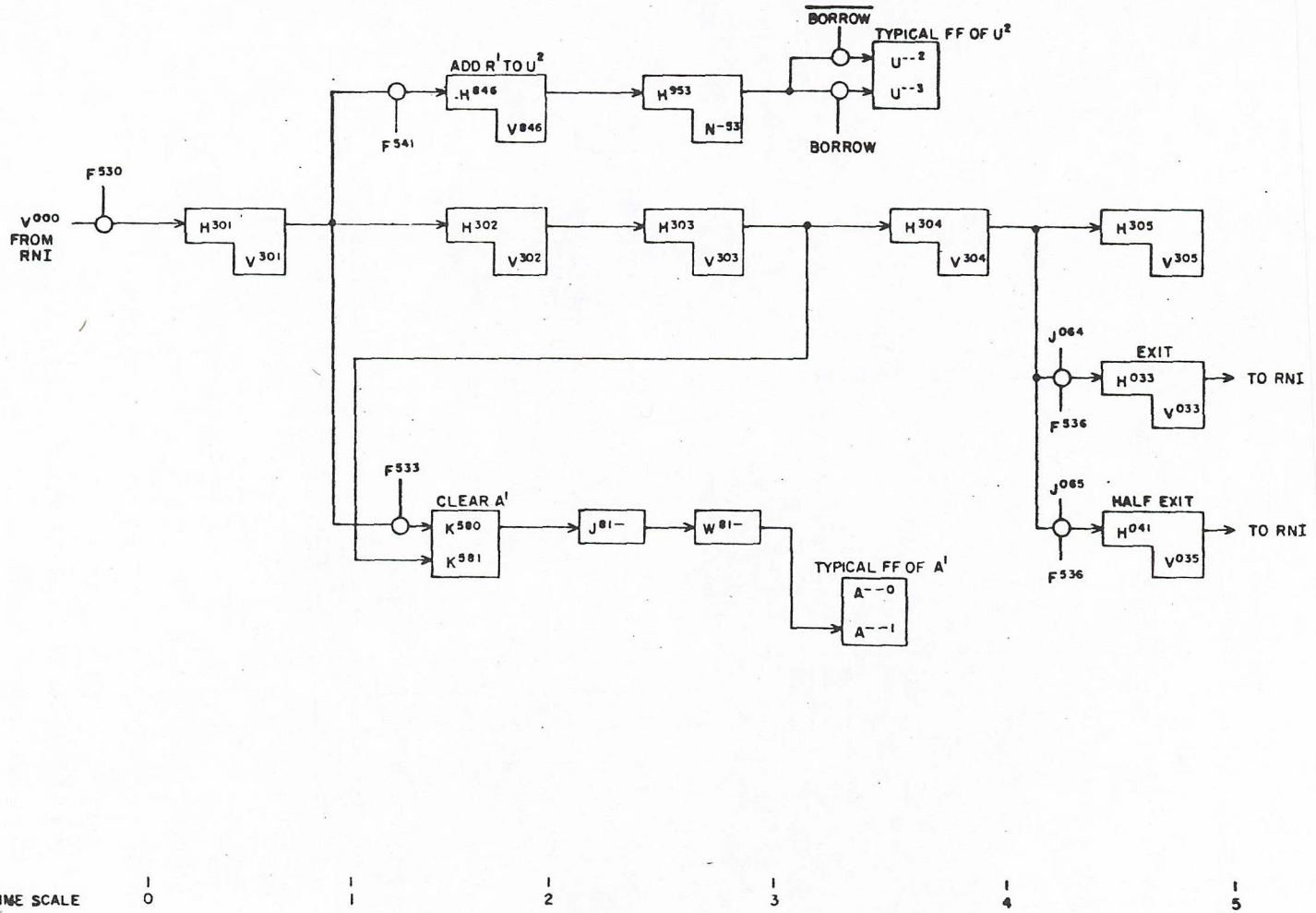


Figure 2-8. Example of a Sequence

RNI reads up another instruction word from storage. In some cases the exit is taken before the present instruction is completed. This allows RNI to prepare for the next instruction at the same time the present instruction is being finalized. The over-all relationship of exits, RNI, and the sequences is shown in figure 2-9.

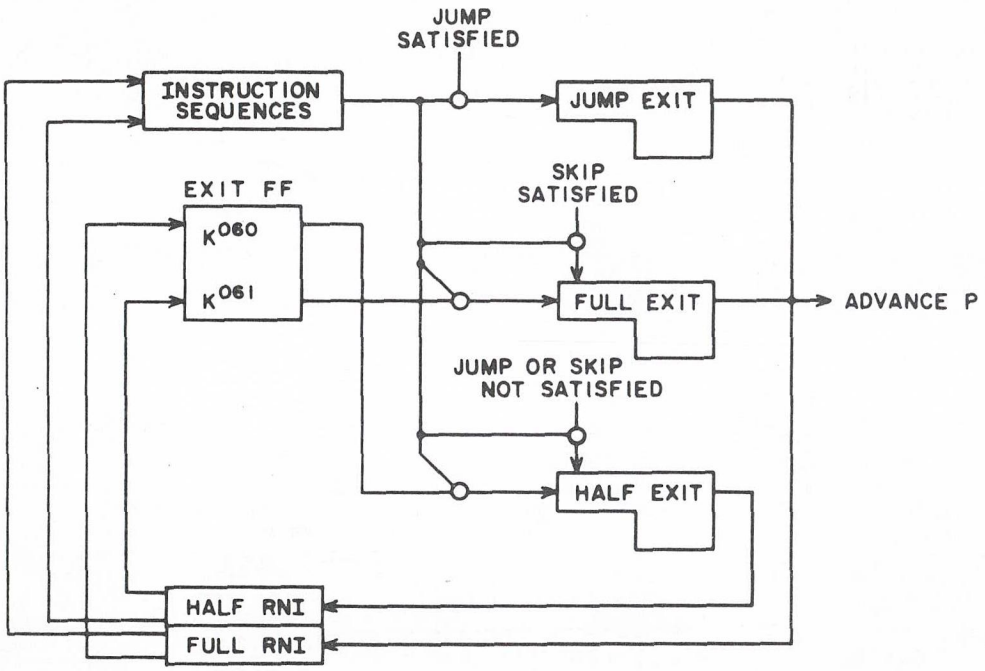
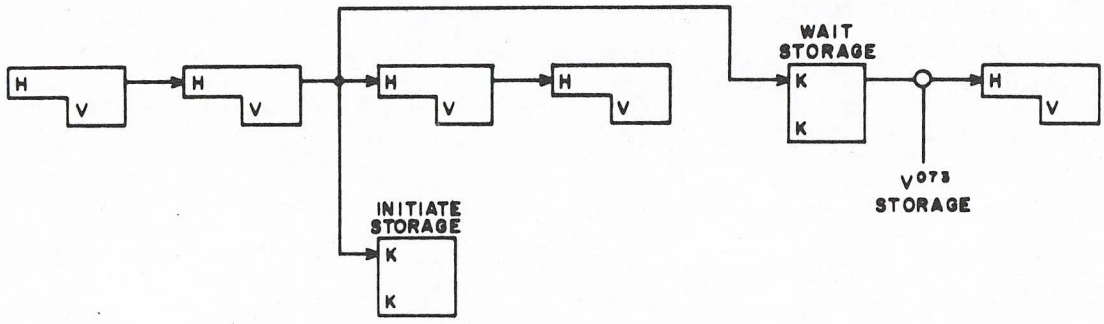


Figure 2-9. Over-all Sequence Control

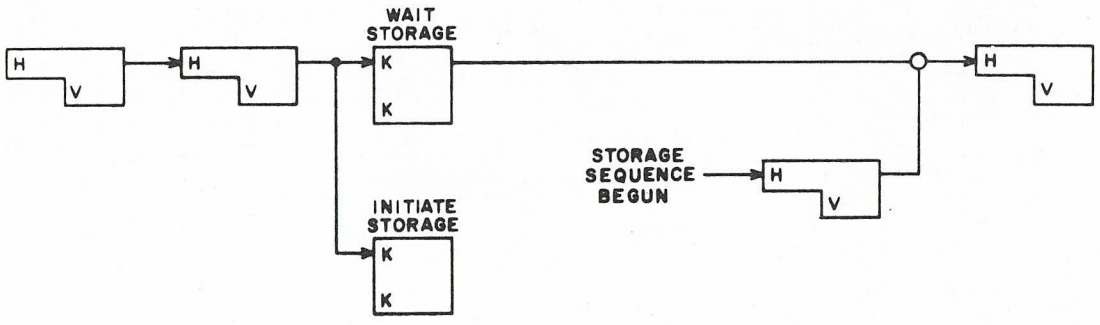
RELATION OF CONTROL AND STORAGE SEQUENCES

For Control sequences which initiate a storage reference it is necessary to maintain proper timing between the Control sequence and the Storage sequence. Successive references to different storage units (even and odd) can overlap, but successive references to the same unit cannot overlap.

Wait Storage FFs in the Control sequence allow for variability in the time when a storage reference may begin and maintain synchronization with the Storage sequence (figure 2-10). Usually this FF is set when the Initiate Storage FF is set. The Control sequence continues with those commands that need not be timed with the Storage sequence. The Storage Resume signal indicates the Storage sequence has read the word from the specified address and can be sampled from I^5 or I^6. The Resume signal and the set output of the Wait Storage FF are combined in an AND to restart the Control sequence.



A. SYNCHRONIZING CONTROL SEQUENCE WITH READ TIME OF STORAGE SEQUENCE.



B. SYNCHRONIZING CONTROL SEQUENCE WITH BEGINNING OF STORAGE SEQUENCE.

Figure 2-10. Relation of Control and Storage Sequences

READ NEXT INSTRUCTION SEQUENCE

The Read Next Instruction (RNI) sequence performs the following functions:

1) Acquisition of a new instruction word.
2) Start and stop
3) Preliminary steps in address modification
4) Preliminary control and arithmetic steps
5) Indirect addressing

Acquisition Of New Instruction Word

Two 24-bit instruction words may be stored in a 48-bit storage location. The RNI sequence (figure 2-11) can transfer the 48-bit quantity to the U^1 register, making the 24-bit quantity in U^1 upper (U^1_U) available as the next instruction or transfer the 24-bit quantity from U^1 lower (U^1_L) to U^1_U.

The sequence steps from initial start, full exit, or jump exit are:

Initiate storage (P Address)
Wait storage
Storage resume
Set Exit FF
Clear U^1 upper and lower
Transmit storage to U^1

The corresponding steps from half exit are:

Clear Exit FF
Clear U^1 upper only
Transmit U^1 lower to U^1 upper.

Start And Stop

The RNI sequence, along with manual control logic, provides for initial starting, stopping, starting from stop, and stepping the 1604-A. A simplified diagram of RNI for these operations is shown in figure 2-12.

Initial Start: To start the computer initially, a Master Clear signal is required to set the Initial Start FF (K054/055). When the Start switch is operated, K055 gates the Start pulse into H090 to initiate RNI.

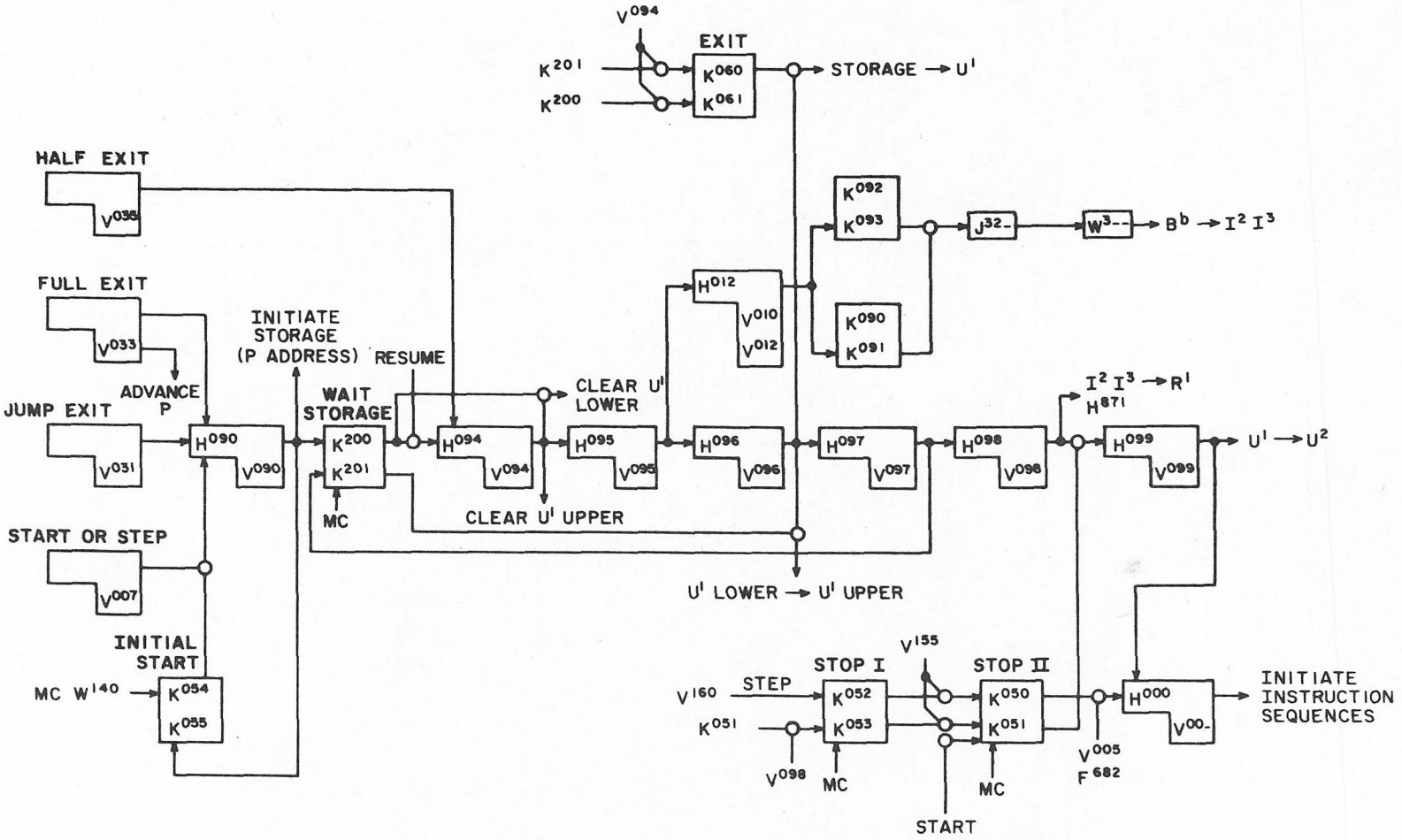


Figure 2-11. Form of RNI for Acquiring Instructions

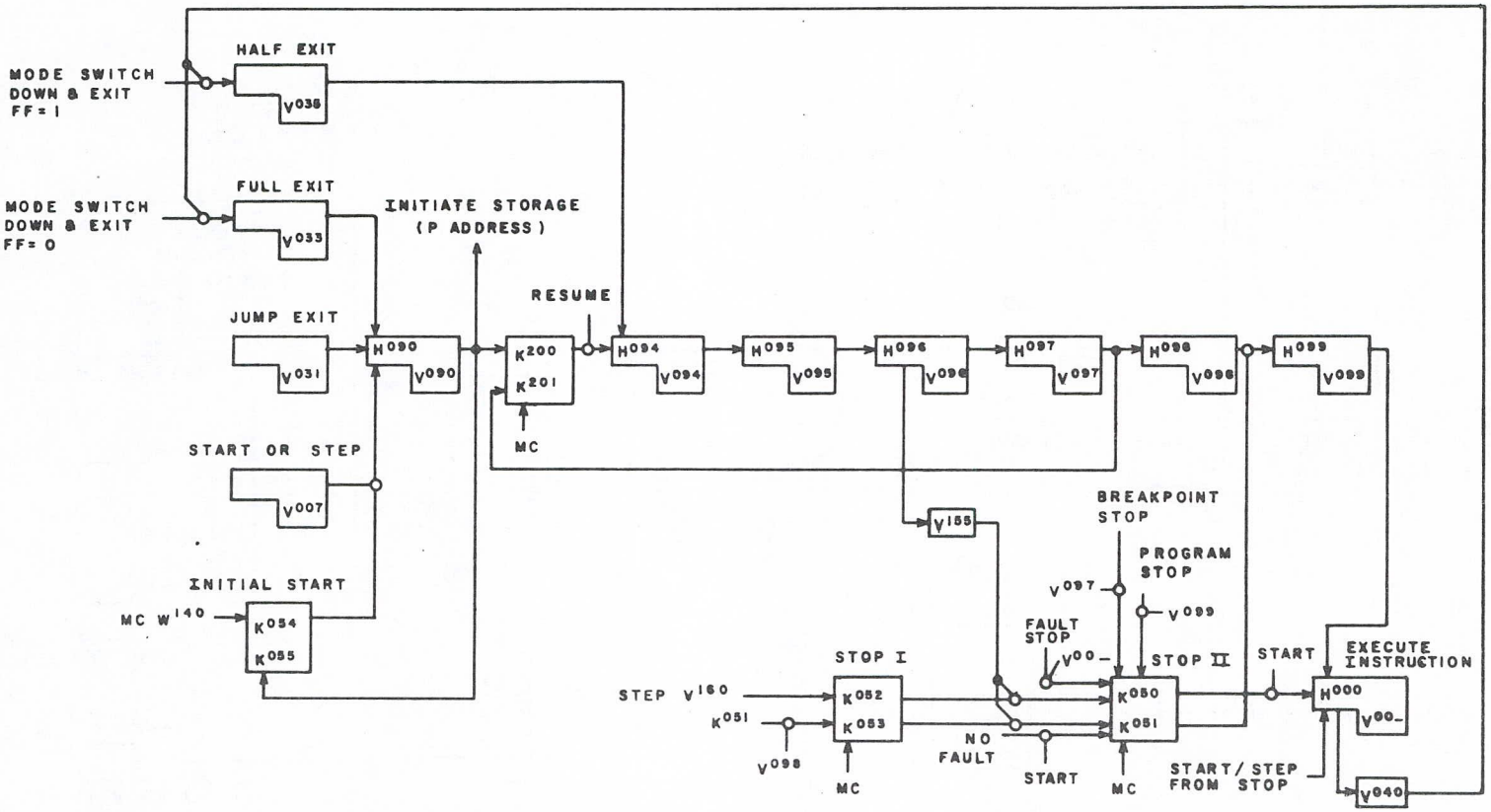


Figure 2-12. RNI For Start And Stop

Step Stepping is used to execute one instruction at a time and stop after each one. If the Start/Step key is pressed to Step when the Initial Start FF is set, RNI is initiated at H090. However, stepping sets the Stop I FF, V155 transfers Stop I to Stop II, and the last control delay of RNI is disabled. Stepping again (step from stop condition) initiates RNI at H000 and one instruction is executed. Thereafter, one instruction is executed each time the Start/Step switch is pressed to Step. If the Start/Step switch is pressed to Step while the computer is operating, Stop I is set; the next RNI transfers Stop I to Stop II, and the computer stops. The computer can then be stepped or started from the stop condition in the normal manner.

Stop Instruction When the condition specified in a Selective Stop instruction (76) is satisfied, the Stop II FF is set. In this case Stop II is not set in time to disable H099 but it is set in time to stop the normal jump sequence which is the only sequence a 76 instruction can initiate. When the computer is restarted, f still equals 76 and the normal jump sequence is executed.

Breakpoint Stop The breakpoint translator compares the address of each new instruction word (content of P register) with the setting of the Breakpoint switches (see page 2-38). When the address and the breakpoint setting become equal, Stop II is set, and H099 is disabled before execution of the upper instruction of the breakpoint address.

Fault Stop The 1604-A has no function codes 00 or 77 and these two codes are considered faults. If the f portion of the instruction read by RNI equals 00 or 77, Stop II is set and the last control delay of RNI (H099) is disabled.

Start From Stop When the computer is restarted from stop, Stop II is cleared, RNI is entered at H000, and the computer proceeds directly to an instruction sequence except in the case where the computer is stopped by a fault condition. Fault conditions must be cleared (e.g., set f manually to some number other than 00 to 77) before Stop II can be cleared and the computer started from stop. When the computer is started from a breakpoint stop, the upper instruction at the breakpoing address is executed and the computer stops again in RNI (because P still holds the breakpoint address). Starting a second time causes the computer to execute the lower breakpoint instruction and then proceed to the next instruction without stopping.

Sweep Mode

When the Mode switch on the 1604-A console is in the Down (Sweep) position, RNI does not initiate an instruction sequence but takes a full or half exit according to the setting of

the Exit FF. Thus in Sweep mode, the content of all storage locations can be read by RNI and displayed without executing any of the instructions.

Preliminary Steps in Address Modification

When the b designator is 1-6, the base execution address may be modified by the addition of (B^b). RNI takes the following preliminary steps in address modification (figure 2-11).

B^b to I^2 or I^3
I^2 or I^3 to R^1
U^1 to U^2

Preliminary Control and Arithmetic Steps

Clear X^1
Set Partial Add in A FF
Q^1 to Q^2
R^1 to R^2

Interrupt Termination

An Interrupt signal causes the computer to temporarily interrupt the main program to perform a special routine of instructions (figure 2-13).

When the interrupt routine is finished, the main program is returned to by:

1) Jump to address 00007 - 00017:
   a) sets P to 00007 - 00017
   b) initiates full RNI from jump exit
2) Full RNI from step 1b:
   a) reads content 00007 - 00017 into U^1
   b) sets Interrupt Complete FF (K072/073)
3) Execution of upper instruction at 00007 - 00017:
   a) jumps to next instruction of main program
   b) initiates full RNI from jump exit

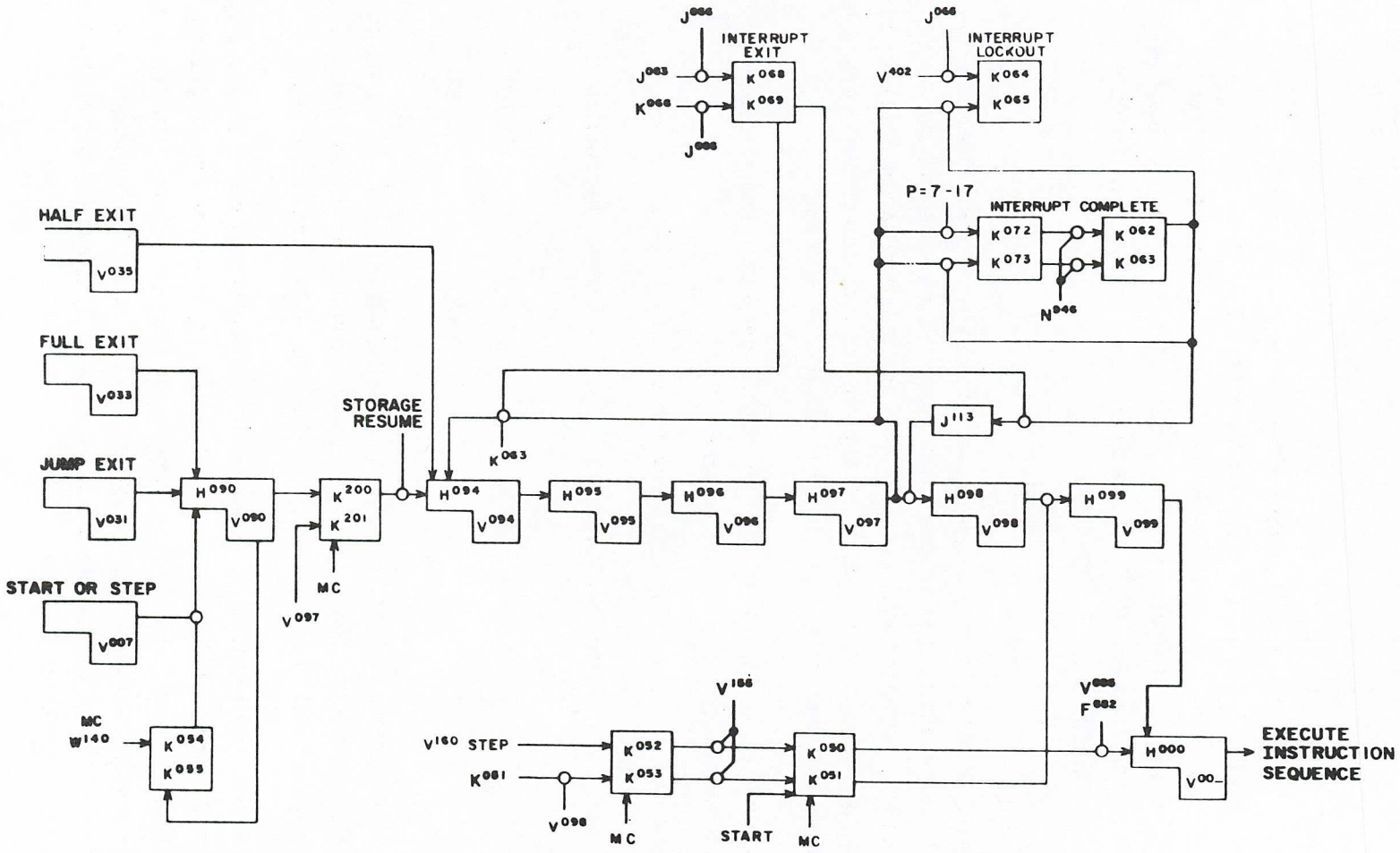


Figure 2-13. RNI for Interrupt Termination

4) Full RNI from step 3b:

a) clears Interrupt Lockout FF

b) clears K072/073 (Interrupt Complete FF)

c) continues from V097 to H098 if Interrupt Exit FF is set

d) goes to H094 from V097 if Interrupt Exit FF is "0" (half RNI)

e) after step 4d, K062/063 is cleared to enable AND from V097 to H098

Indirect Addressing

Indirect addressing is often chosen for programs involving a great deal of address modification because it simplifies programming and reduces the running time. In direct addressing the execution address indicates the location of the operand; in indirect addressing the execution address indicates the location of the operand address. An additional memory reference is required to obtain the operand.

All instructions except 22, 23, 74, 75 and 76 may be used with either direct or indirect addressing. Indirect addressing occurs when b = 7.

Examples: Below are two examples of indirect addressing.

Address	Upper Instruction			Lower Instruction		
	f	b	m	f	b	m
05012	36	3	71331	14	7	00367
00367				42	2	11135

Because b is 3 in the upper instruction, direct addressing is used. B^3 is added to 71331 to produce the address of the operand. In the lower instruction, because b is 7, indirect addressing is used; therefore m is the location of the new operand.

Now the lower 18 bits are read out of address 00367 (remaining upper bits are ignored). These 18 bits are substituted in the Program Control register for the original 18-bit quantity made up of b and m. As a consequence, the current instruction becomes 14 2 11135. The designator is examined again; since it is not 7, the address of the operand is 11135 + (B^2). If the new value of b had been 7, a second indirect addressing operation would have resulted.

Address	Upper Instruction			Lower Instruction		
	f	b	m	f	b	m
05013	01	7	04006	12	6	71331
04006					7	11466
11466					0	00012

Since b is 7 in this instruction, the lower 18 bits at address 04006 are substituted in the Program Control register which then holds 01 7 11466. Since b is again 7, the lower 18 bits in address 11466 are entered into the Program Control register. Because b is zero, 00012 is used as the execution address.

Indirect addressing is accomplished by RNI (figure 2-14). By the time RNI has progressed to V099, f and b have been translated; direct or indirect addressing is selected. Conditions for the choice are:

Indirect Addressing - f ≠ 22, 23, 74-76 and b = 7
Direct Addressing - f = 22, 23, 74-76 or b ≠ 7

The steps for indirect addressing are:

Initiate storage (on address m)
Clear b and m parts of U^1_U
Wait storage
Transmit storage to b and m part of U^1_U

At the end of this sequence a new 18-bit quantity occupies the b and m portions of U^1_U. A pulse from V153 (figure 2-14) then re-enters the RNI sequence at H098 and the new instruction is executed.

NORMAL JUMP SEQUENCE

The Normal Jump (NJ) sequence is used by instructions 22, 23, 75 and 76, when j = 0-3.

When the jump condition is met, this sequence transmits U^2 to P^1 (figure 2-15) entering the address of the first instruction of a new program in P. The jump exit is then taken to initiate the full RNI and read the pair of instructions at address P. If the jump condition is not met, the full exit or half exit is taken to initiate the full or half RNI.

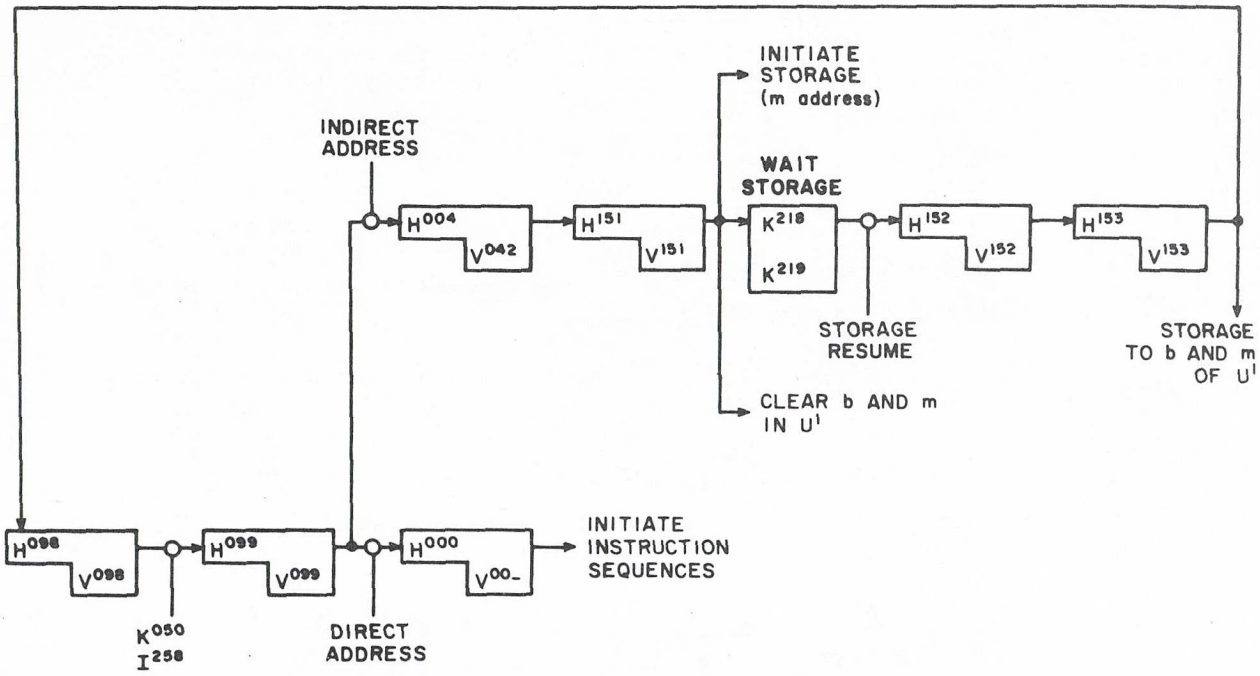


Figure 2-14. Indirect Addressing Part of RNI

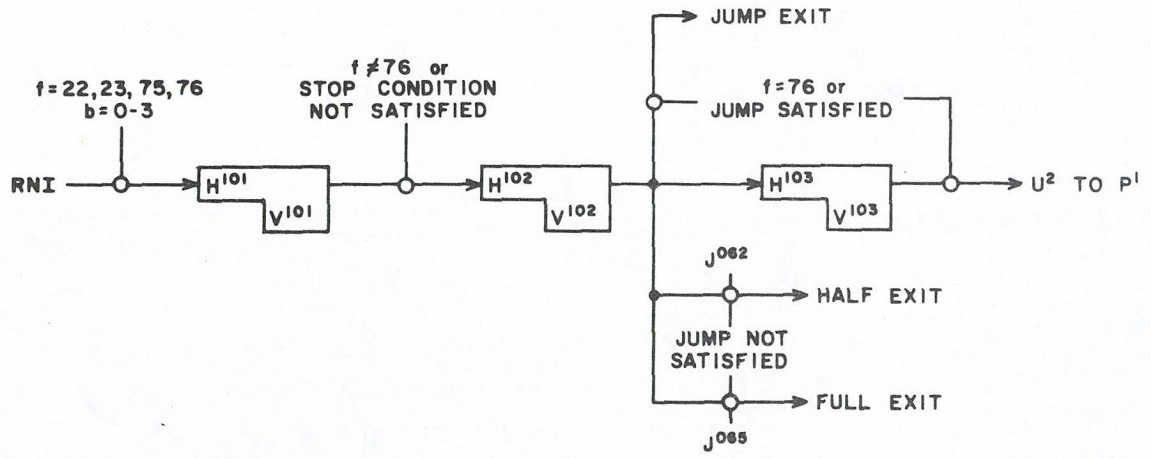


Figure 2-15. Normal Jump Sequence

Instruction 76, Selective Stop, provides for a jump whether or not the conditions for stopping are met. When the stop condition is met, the AND to H102 is disabled and operation stops. When operation is resumed by either start or step, the NJ sequence is initiated, the AND to H102 is enabled and the jump is completed.

ZERO ADDRESS SEQUENCE

The Zero Address (ZA) sequence performs the basic operation for instructions 01-11, 34, 35, 50, 51, 54, and 55. This sequence makes no storage reference; the execution address is the operand for these instructions.

The chain of control delays forming the basis of the ZA sequence consists of two parts (figure 2-16). All ZA instructions use the first part; only instructions 04, 10, 11, and 54 (with $R \neq 0$) use the last part. Most of the commands for ZA instructions are generated from this chain of control delays. However, the commands for shifting and reducing R in instructions 01-03, 05-07, 34, and 35 are generated independently of the main chain by the shift control. The chain establishes the conditions which permit these commands to occur until the terminating conditions are reached.

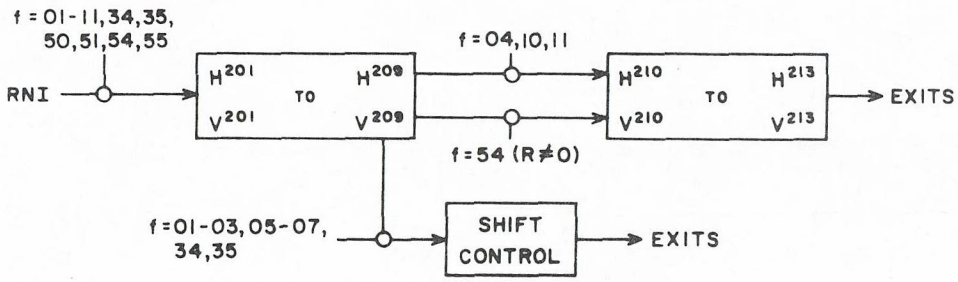


Figure 2-16. Basic Chain of Control Delays in Zero Address Sequence

READ OPERAND SEQUENCE

Instructions 12-17, 36-46, 52, 53, 70-73, and Advance Clock use the RO (Read Operand) sequence (figure 2-17). These instructions all read from storage; Advance Clock and instructions 37 and 70-73 also store an operand at the conclusion of the sequence.

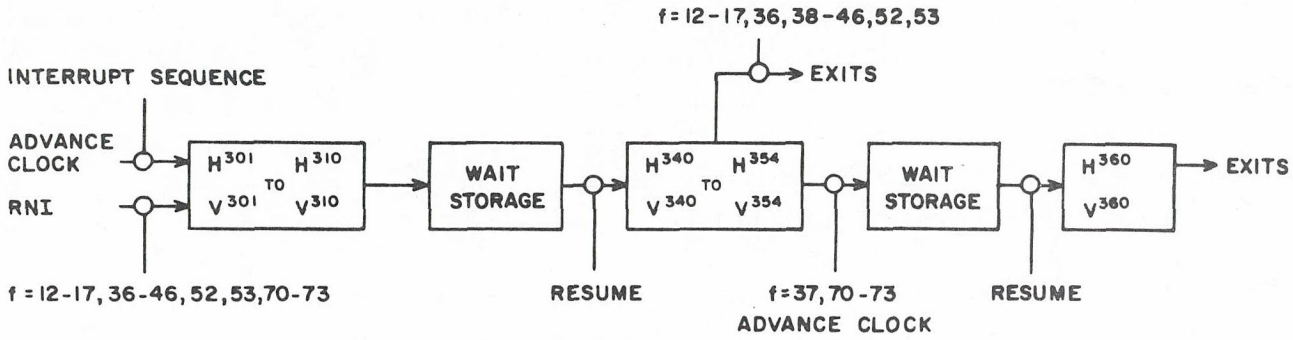


Figure 2-17. Basic Chain of Control Delays in RO Sequence

WRITE OPERAND SEQUENCE

Interrupt and instructions 20, 21, 47, 56-61 for all values of the designator and Return Jump instructions (22, 23, 75 and 76 with designator values of 4-7) use the WO (Write Operand) sequence. It is initiated at H401 and all WO instructions except 56 and 57 which initiate the sequence at H406 after initiating a storage sequence.

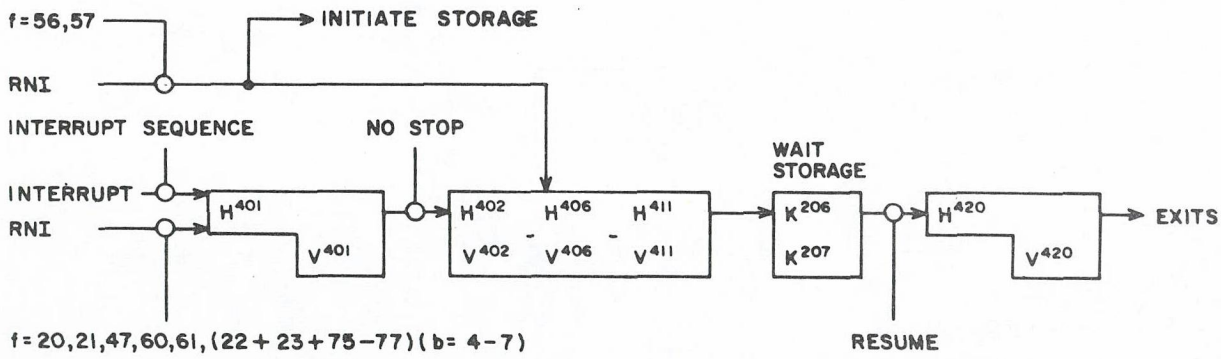


Figure 2-18. Write Operand Sequence

SEARCH AND TRANSFER SEQUENCE

The Search and Transfer sequence (S & T) performs search instructions 64-67 and transfer instructions 62 and 63. The search instructions inspect a specified list of operands for one which meets the condition "operand equal to A" or "operand greater than A." The transfer instructions exchange a block of data with Input-Output equipment.

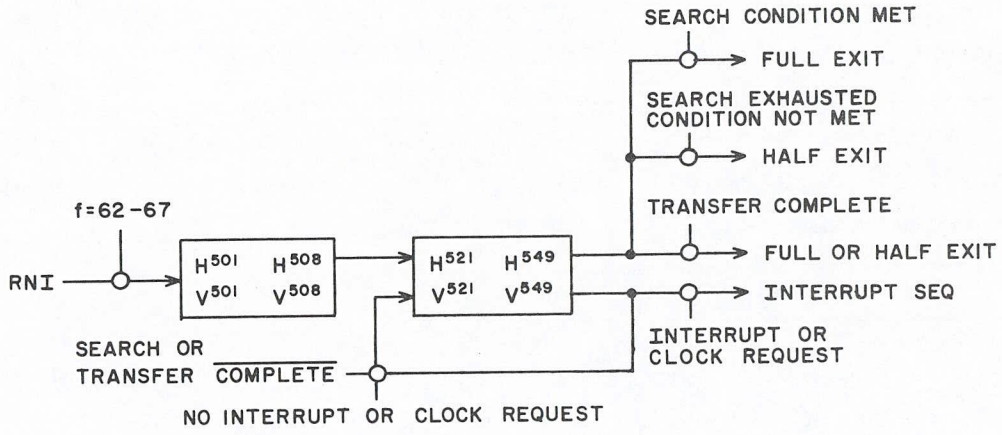


Figure 2-19. Basic Block Diagram of Search and Transfer

The S & T sequence (figure 2-19) consists of two main parts; preparation and loop. Preparation sets up the addresses and other initial conditions. The loop produces a series of search or transfer commands which is repeated for each word searched or transferred. The number of repetitions required is indicated by the quantity in B^b which is reduced by one during each repetition. The block of storage locations involved in Search and Transfer instructions is specified by B^b and m, the execution address. The first word searched or transferred is at the address $(B^b) + m-1$. The last word is at address m. When b = 0, only the word at address m is searched or transferred.

A Search instruction provides for a conditional skip and is used in the upper position of a program step. Ordinarily it involves repeating the loop for each word until:

1) A word is found that meets the search condition; search terminates and a full exit is made to the next instruction step.
2) The entire list of words has been searched without finding one that meets the search condition, in which case a half exit is made to the lower instruction of the current step.

A Transfer operation consists of one repetition of the loop for each word. When all have been transferred, a full or half exit is taken depending upon whether the instruction is in the lower or upper position.

ITERATIVE SEQUENCE

The commands which execute the Multiply, Divide, and Floating-Point instructions (24-27, 30-33) derive from the Iterative sequence (I). This sequence (figure 2-20) executes instructions by iterative or repetitive action. For example, the repetitive additions of the multiplicand form the product in multiplication.

A detailed discussion of the Iterative sequence appears in chapter 3 where the complex arithmetic operations using this sequence are described.

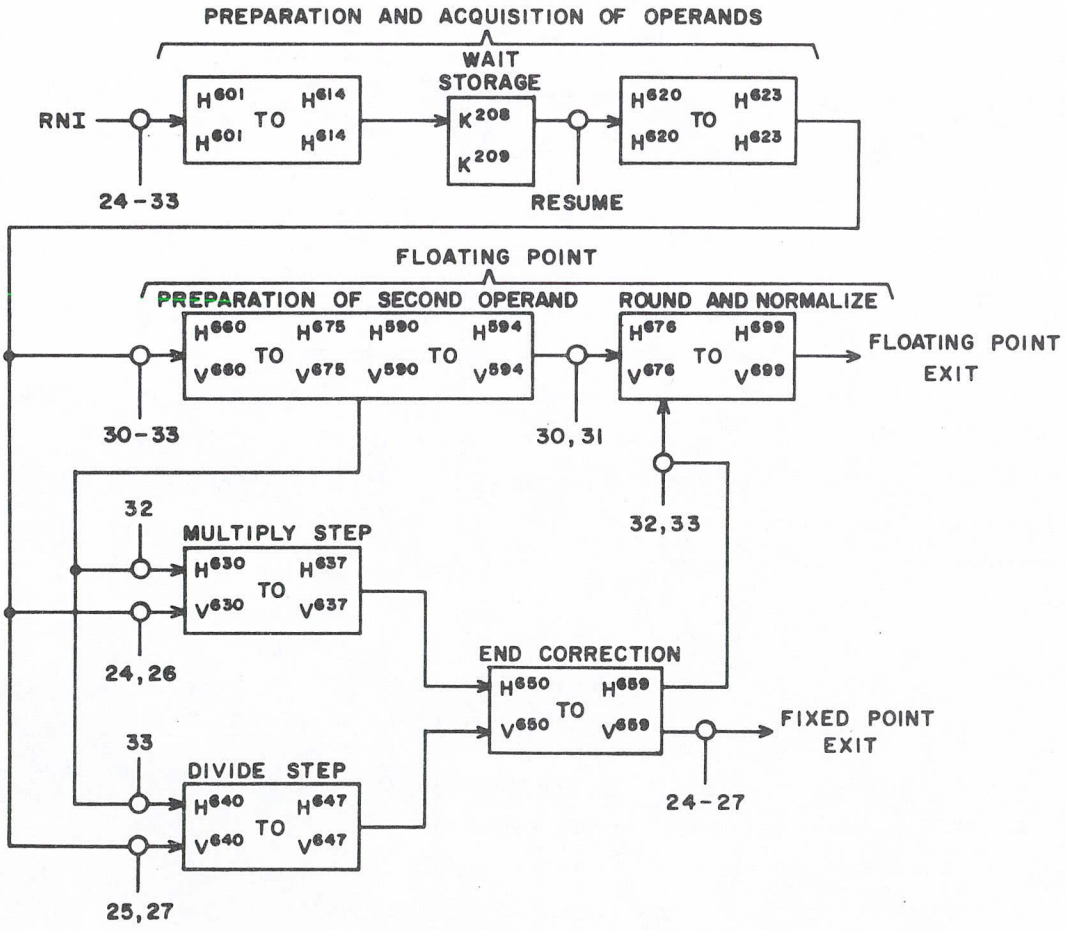


Figure 2-20. Iterative Sequence

EXTERNAL FUNCTION SEQUENCE

The External Function (EXF) sequence (figure 2-21) executes the 74 instruction to select operating conditions for external and console equipment, sense the condition of external and console equipment, sense internal faults, and activate buffer communcation channels 1-6. The value of the designator, j, is used to determine which of its duties the EXF sequence is to do. High speed storage sequence is used in conjunction with EXF sequence to execute the Activate command (j = 1-6). A 74.0 instruction selects the operating condition for external or console equipment. A 74.7 instruction is used to sense equipment conditions and faults within the computer. When the 74 instruction is used with j = 1-6, a buffer channel is activated.

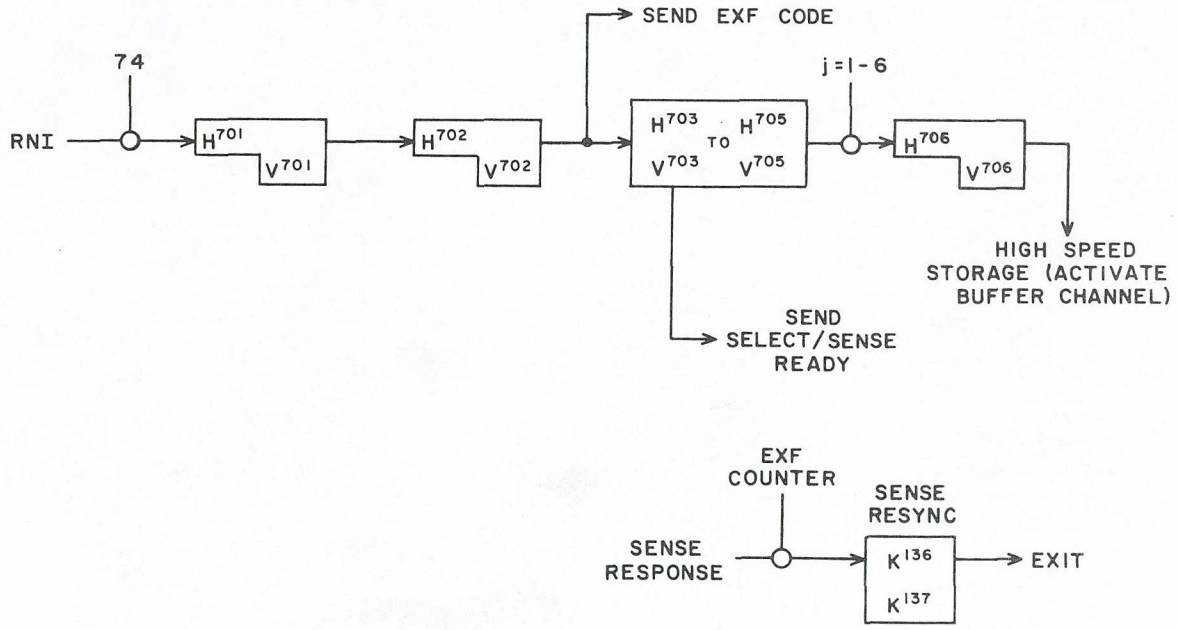


Figure 2-21. Block Diagram of EXF Sequence

INTERRUPT SEQUENCE

When an interrupt or clock request occurs, the Interrupt sequence is initiated. This sequence (figure 2-22) can be entered between any two program steps or during a Search or Transfer instruction.

Interrupt

To process an interrupt request, the WO sequence follows the Interrupt sequence. The content of P is stored to provide for return to the main program and a jump is made to an interrupt subroutine. The interrupt subroutine senses and corrects the condition causing interrupt and then returns to the main program. Interrupt is discussed in detail in chapter 6.

Advance Clock

The Interrupt sequence is also used to advance the real time clock. In this case it is followed by the RO sequence which adds one to the word in address 00000. When the real time clock is selected, Interrupt sequence advances the clock every 16 2/3 milliseconds.

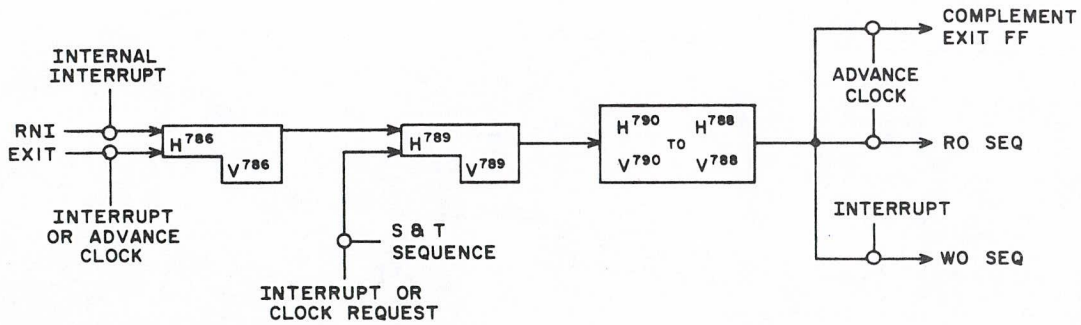


Figure 2-22. Block Diagram of Interrupt Sequence

BUFFER CONTROL SECTION

The Input/Output operations of the 1604-A are controlled by the Buffer Control section. Buffer Control consists basically of:

Auxiliary Sequence
High Speed Storage Sequence
Control Registers 1-6
Common Control Register
Comparator
Inverter Ranks I^7 and I^8

Buffer Control assumes control of buffer operations initiated by the main program and carries them to completion. When a buffer operation is completed, Buffer Control signals Main Control by clearing the appropriate Buffer Active FF. Up to six buffer operations may be handled concurrently by Buffer Control.

AUXILIARY SEQUENCE

The commands which control buffer operations are issued by the auxiliary sequence of control delays H^{440} through H^{478} (figure 2-23). After an auxiliary operation is started by the main program, auxiliary sequence operates independently in completing the operation. The Aux Seq. is initiated by an auxiliary request signal if storage is not busy.

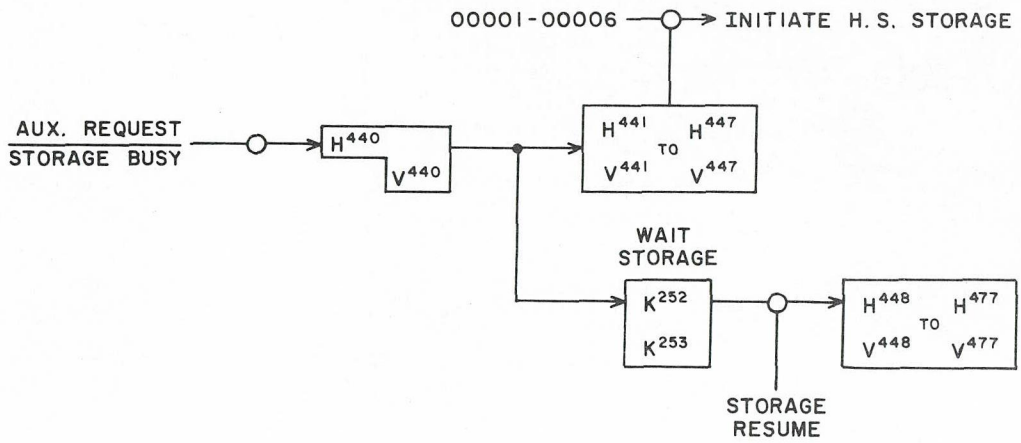


Figure 2-23. Block Diagram of Auxiliary Sequence

Since buffer operations are independent of program operations, there is no exit from auxiliary sequence.

High speed storage sequence is initiated by auxiliary sequence when data is to be buffered to or from cells 00001-00006.

HIGH SPEED STORAGE SEQUENCE

The high speed storage sequence (HSS) is used to reference (read or write) Control Registers 1-6. Whenever a storage reference is made to locations 00001-6, the high speed storage sequence runs simultaneously with storage sequence control. HSS sequence is also used when doing an EXF activate instruction and when buffering into or out of locations 00001-6. The operation of HSS is explained in more detail in Chapter 5.

CONTROL REGISTERS, COMMON CONTROL REGISTER, AND COMPARATOR

Control Registers 1-6 are used to store the address portions of words 00001-00006 (buffer control words). These words are used to monitor buffer operation. After each word is transferred in a buffer operation, the Common Control Register (CCR) increases the upper address of the control word and the comparator compares it with the lower address. When the upper and lower addresses of a control word are equal, the buffer operation is complete and the comparator deactivates the buffer channel.

INVERTER RANKS I^7 AND I^8

Inverter Ranks I^7 and I^8 are parallel transmission paths in buffer control. I^8 provides additional outputs from I^5, I^6 to CR 1-6. I^7 provides inputs from CR 1-6 to the CCR.

AUXILIARY SCANNER

To ensure that one buffer channel cannot monopolize buffer control time, an auxiliary scanner is used to initiate buffer operations. The auxiliary scanner (figure 2-24) samples each buffer channel in the order 1-3-2-6-4-5.

When the scanner detects an auxiliary request, it stops and initiates an auxiliary operation.

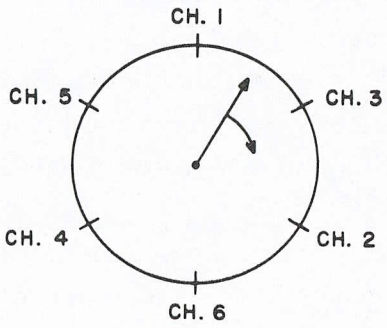


Figure 2-24. Pictorial Representation of Scanner

When the scanner detects an auxiliary request, it stops and initiates an auxiliary operation.

The scanner is released in time to scan 4 other channels before the present auxiliary operation releases storage to program operation. This gives buffer operations priority over program steps in initiating storage references.

The scanner can handle requests from all six buffer channels by allowing each to take a turn at buffering a word. Thus the scanner gives each channel equal priority in requests for storage time.

STATIC CONTROL

Static control, consisting of Breakpoint switches and the operational registers, enables the operator select an address at which the program is to stop and to observe the contents of the operational registers and other displays.

BREAKPOINT ADDRESS

A digit switch assembly on the console provides for setting an address at which the program is to stop; this address is called the breakpoint. The breakpoint address is continually compared with P, the program address; when the two quantities are equal, the computer stops just before performing the upper instruction at the breakpoint address (figure 2-12). Stopping occurs only during the full RNI.

There are five Rotary Digit switches in the switch assembly; each has eight positions numbered 0 through 7. Thus every address is represented by some combination of the five wheels. The breakpoint is disabled by setting it to an address not used by the program, generally 77777.

CONSOLE DISPLAY

The contents of operational registers are displayed in octal on the console only when the computer is stopped. The display circuits of the lowest octal digit (3 bits) of the A register are shown in figure 2-25. The digit display gate is turned on when the computer is stopped. Output amplifiers (L500-L502) sample the state of the A register. Each L5-- energizes a relay if the associated bit is a "1". Three relays are interconnected to form a binary to octal translation; the output of the translator illuminates the octal digit at the associated position. By means of colored indicators (figure 2-26), the console provides visual indication of several other conditions in the computer and console equipment. The background indicators can be illuminated when the computer is operating. Designation is according to the associated flip-flop. Exceptions: Reader Ready, turned on when Reader End-of-Tape FF is cleared; Punch-Out-of-Tape, turned on by a switch at the punch; Lower Instruction indicator, turned on when Exit FF is cleared.

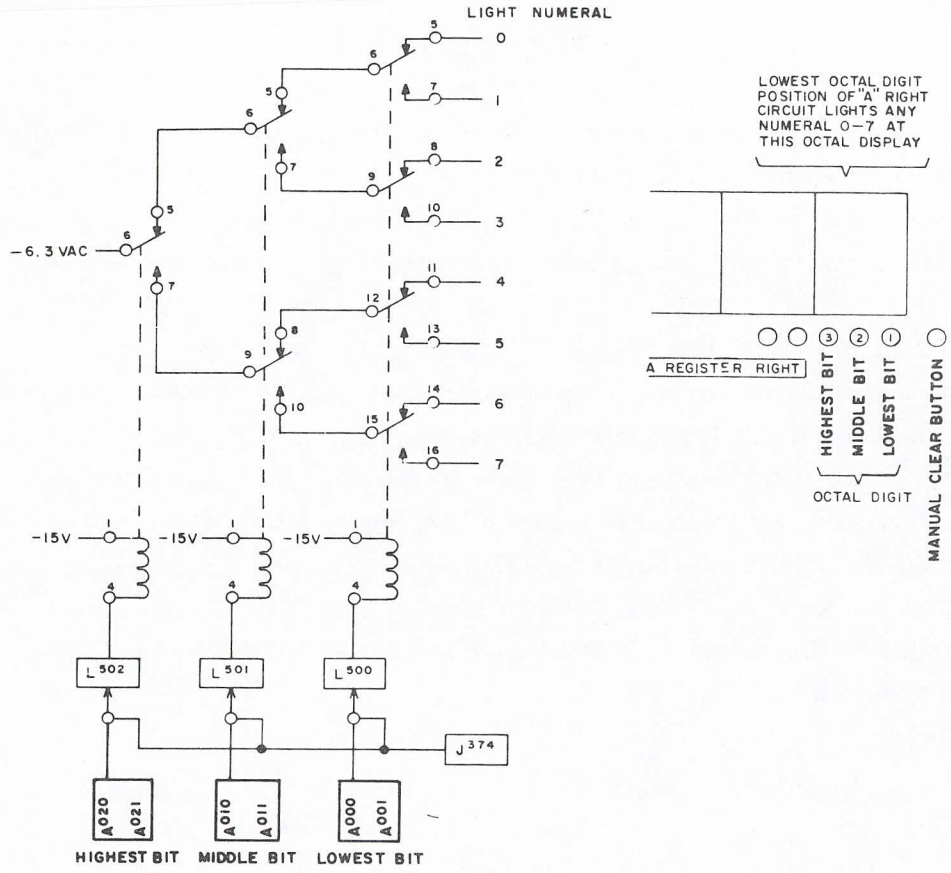


Figure 2-25. Digit Display, Lowest Octal Digit of A

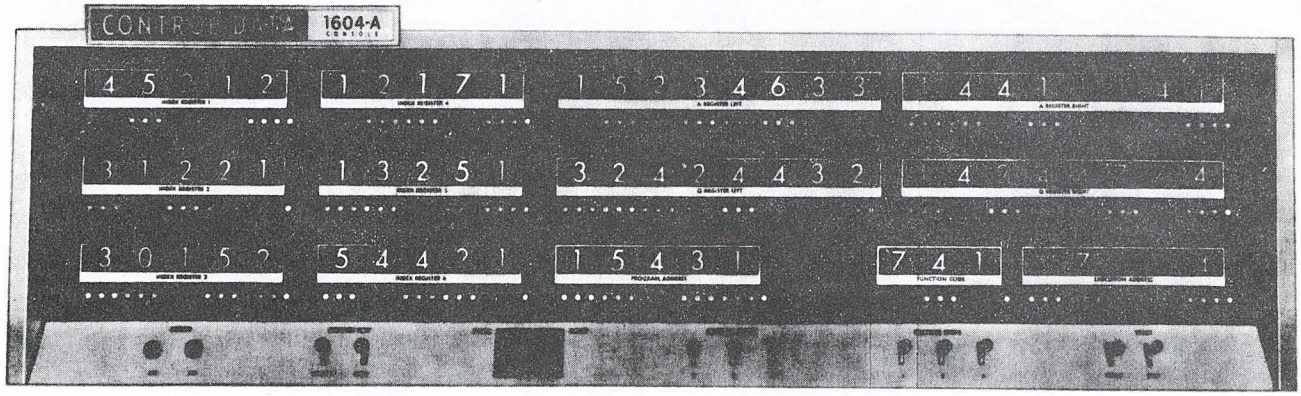


Figure 2-26. Console Display

RESYNCHRONIZING

Signals from external equipment and switches which are asynchronous to the timing of the computer must be resynchronized. This involves insuring that only one synchronized pulse results from an asynchronized signal, regardless of the duration of such a signal. Another purpose of resynchronizing is to resolve runt pulses.

RESYNCHRONIZING CIRCUIT

The first objective of resynchronizing is to convert an asynchronous signal to one pulse which is timed by the computer. In the resynchronizing circuit even pulses occur every 1.6 usec; odd pulses also occur every 1.6 usec with a lag of 0.2 usec between the even and odd pulse. In figure 2-27, the input to M164 is -20 volts; output is "0". K994/995 is cleared and K996/997 is set. On the next odd sync pulse after M164 receives a signal, K994/995 is set. During the following even sync pulse, H295 receives a pulse. The output of V295 clears K996/997 to prevent another pulse on the second even sync pulse.

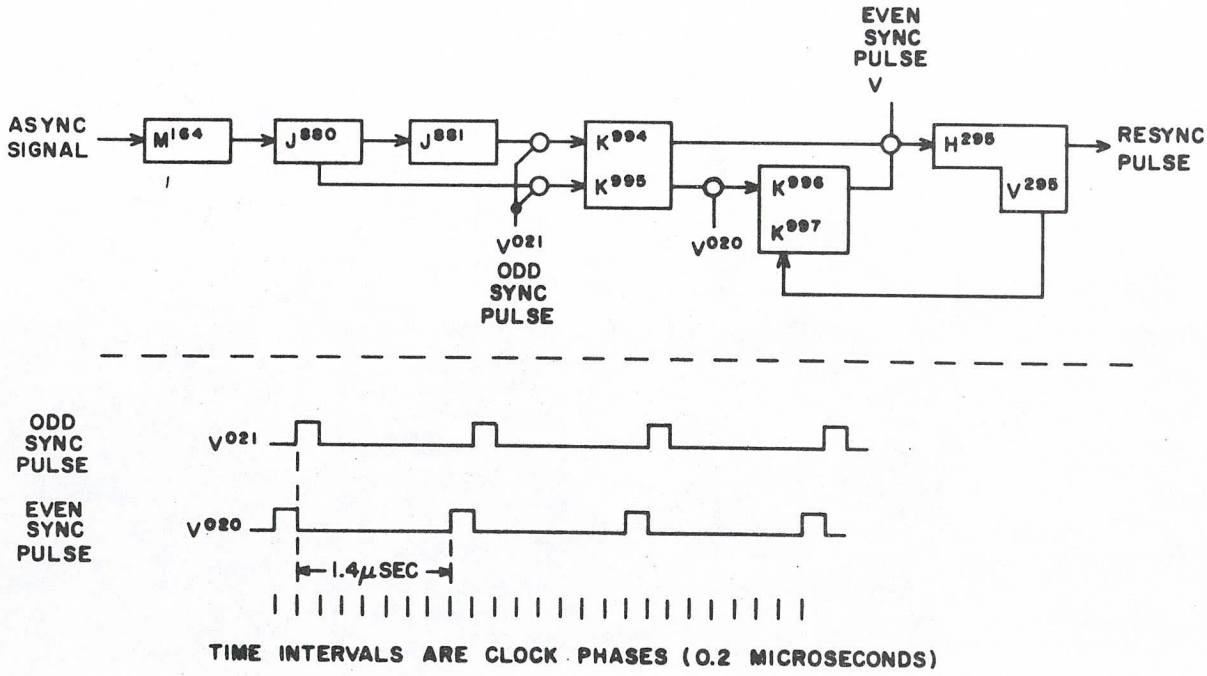


Figure 2-27. Resynchronizing Circuit

RUNT PULSES

A runt pulse is a pulse between -20v and 0v which does not define either a "1" or a "0" on a control line. A runt pulse input to M164 (figure 2-27) results in an output between -3.0v and -0.5v which does not definitely indicate either a logical "1" or "0". If the output of M164 is -1.5v, the output from J880 and J881 is also -1.5v.

When the odd sync pulse from V021 occurs, the set and clear inputs to K994/995 receive half-size inputs which attempt to both set and clear this FF. In this circumstance, the FF is not fully set or cleared within 0.2 usec, the normal switching time of a FF. When runt pulses are applied, a longer period is required for the FF to settle into a full "1" or "0" state. The maximum period required for such settling is 1.4 usec, which is the interval between the odd sync pulse (when runt pulses may be applied to K994/995) and the even sync pulse (when K994/995 is sampled at the input to H295). Only after the runt input is resolved into a full "1" or "0" can H295 sample the state of K994/995.

The sync pulses resolve runt pulses in the first FF of the resynchronizing circuit. The odd sync pulse sets the time for sampling the asynchronous signal, and the even sync pulse sets the time for sampling the first FF of the circuit.

RESYNCHRONIZING COUNTER AND PULSES

The sync pulses used in resynchronizing circuits throughout the computer are produced by a two-stage counter (figure 2-28). This counter is advanced every even clock phase (every 0.4 usec). Each time the counter reaches three (11), H020 receives a pulse. This pulse produces even sync pulses which are distributed throughout the computer by V020/022/024/026. A pulse sent by V020 to H021 causes the odd sync pulses that are distributed by V021/023/025/027. The counter returns to zero (00) on the next even clock phase after reaching the count of three.

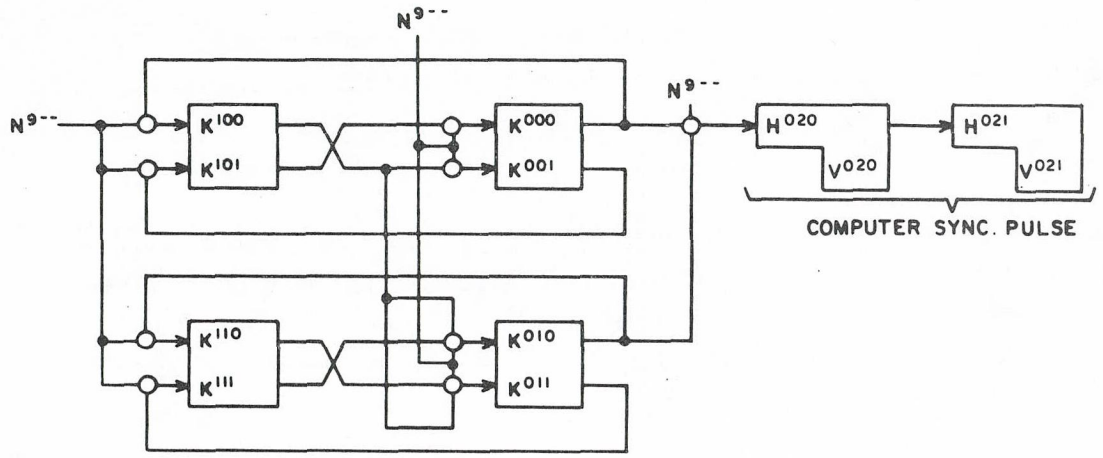


Figure 2-28. Resynchronizing Counter

JUMP INSTRUCTIONS

The instructions for program jumps are 22, 23, 55, 75, and 76; 76 also provides for stopping. Instructions 22, 23, 75, and 76 with j = 0-3 and instruction 55 with all values of j cause a normal jump. Instructions 22, 23, 75, and 76 with j = 4-7 cause a return jump.

When a normal Jump condition is satisfied, the current program is suspended and a new sequence of instructions at the storage location specified by the Jump instruction is executed. A Return Jump instruction prepares for return to the original instruction sequence upon completion of the new sequence. When the Jump condition is not satisfied, the program continues in a normal manner. A Jump instruction may appear in either the upper or the lower position of an instruction word. If a jump is taken from the upper instruction, the lower instruction is never executed.

NORMAL JUMP

Normal Jump instructions are executed by the Normal Jump sequence (NJ), with the exception of instruction 55, which is executed by the Zero Address sequence. For conditional jumps an examination is first made to determine if the condition is satisfied. If the jump is taken, the unmodified m portion of the Jump instruction is transmitted from U^2 to P^1. The jump exit is then taken to the RNI sequence.

Entering RNI by the jump exit differs from entering by the full exit only in that P is not advanced before the P^1 and S^1 transmission. An RNI following a jump reads an instruction from the storage location given by m of the Jump instruction.

If the Jump condition is not met, then a half exit is taken to RNI if the jump was an upper instruction; a full exit is taken if the jump was a lower instruction.

RETURN JUMP

A Return Jump instruction is always executed by the Write Operand (WO) sequence. In the program example shown here the lower instruction at relative address c has been executed and the instruction word in c + 1 has been read. The left instruction in c + 1 is a return jump to be performed if A $\neq$ 0. If this condition exists, the execution of the return jump involves four basic steps:

Step 1. Advance P to c + 2.

Step 2. Initiate at address d a storage reference which reads the contents of d into U^1 and writes the contents of P, c + 2, in the m portion of the upper instruction at d.

Step 3. Transmit d from U^2 to P.

Step 4. Half exit.

The first step yields the address of the next instruction of the main program. The instruction prepares for return to this address. Step 2 reads the pair of instructions at address d; the lower is the first to be executed in the subroutine. Step 2 also stores the quantity c + 2 as an address in the upper instruction at location d. Step 3, by entering the quantity d in P, prepares for sequential execution of instructions in the subroutine. The half exit to RNI in step 4 results in the subsequent execution of the lower instruction in address d. The upper instruction at d is executed in the actual return to the main program after completion of the subroutine.

After step 4, the Return Jump instruction is completed and the subroutine begins with the lower instruction at d. When the upper instruction at d + n has been executed, the basic function of the subroutine is accomplished; the main program is re-entered by the normal jump in the lower part of d + n. This instruction enters the quantity d in the P register and initiates RNI through the jump exit.

Program Example

A. Main Program

Relative Address	Instruction	
	Upper	Lower
c	f b m	f b m
c + 1	22 5 d	f b m
c + 2	f b m	f b m

B. Subroutine

d	75 0 c + 2	f b m
-		
-		
-		
d + n	f b m	75 0 d

Consequently, both instructions in address d are entered in U^1 and the upper one is executed. It is this instruction which received c + 2 as its execution address. Since this instruction is another unconditional normal jump, c + 2 is entered in P and RNI is initiated through the jump exit. With the execution of the upper instruction in c + 2, the main program is resumed.

SKIP INSTRUCTIONS

Instructions 36, 37, 54, 64-67, and 74.7 provide for skipping the next instruction of a program. These instructions are limited to the upper position of an instruction word as they provide for skipping the lower instruction by using the full exit when a certain condition is met. If this condition is not met a half exit is used to return to RNI.

The 74.7 instruction (External Function Sense) when used as a lower instruction causes an indefinite period of waiting. The half exit is taken and the 74.7 is repeated until a specified condition is met; then a full exit is taken. Further details on 74.7 are found in chapter 5.

The logical networks associated with the switches on the lower panel of the console (figure 2-26) are:

Start - Step
Clear
Selective Jump
Selective Stop
Storage Test

START-STEP SWITCH

The Start-Step switch selects the mode of computer operation. Positions are momentary. The Start (up) position selects the high-speed mode in which a program of instructions and auxiliary operations proceed until completed, or until program stop occurs. The Step (down) position selects a mode in which a single instruction is executed; operation then stops to await further manual selection. One instruction is executed each time the switch is pressed to Step position. The Start-Step switch is ineffective as long as any buffer channel is active.

Stepping while a buffer is active, however, sets Stop I FF and causes the computer to stop during the first RNI after all buffers go inactive.

The Step position has priority over the Start position. If the computer is operating at high-speed, pressing the switch to Step causes operation to stop. Start and Step selections are felt at the RNI sequence (figure 2-29). The Start, Step, and Neutral contacts of the switch are connected to a Resynchronizing circuit which converts to single pulses the d-c levels produced by closing these contacts. Either a Start or a Step pulse causes the digit display to be turned off; after 16.6 ms the pulse is applied to RNI. This delay allows transients, which result from turning off the indicators, to settle before computer operation begins. In analyzing the effect of a Start or Step pulse, it is necessary to distinguish between a pulse immediately following a Master Clear signal and one that does not follow Master Clear. A Master Clear, in addition to clearing registers and control FFs, sets the Initial Start FF. Figure 2-29 shows that a Start pulse following master clear,

1) Initiates a full RNI at H090
2) Clears Stop II
3) Begins execution of program at high speed

A Step pulse following a master clear,

1) Initiates a full RNI at H090
2) Sets Stop I
3) Sets Stop II from Stop I by V096
4) Halts RNI after V098

This pulse reads a pair of instructions. Another Step pulse is necessary to execute the first instruction. A Step pulse not preceded by a master clear,

1) Sets Stop I
2) Initiates execution of instruction at H000
3) Sets Stop II from Stop I during following RNI
4) Halts following RNI at V098 because of Stop II

A Start pulse not preceded by a master clear,

1) Clears Stop II
2) Initiates execution of instruction at H000

The program execution continues since Stop II is cleared and remains cleared.

CLEAR SWITCH

The Clear switch at the console provides for master clearing the computer (Down position) and the external equipment connected to the computer (Up position). A master clear completely erases all traces of the previous mode of operation. Either Master Clear signal may be used at any time by the operator, regardless of the mode of operation or whether the equipment is operating or stopped. The Master Clear takes effect immediately.

Computer Master Clear

Clear switch Down master clears the computer by:

1) Forcing all stages of the registers to the "0" state except the P register which is set to 00020
2) Forcing almost all Control FFs throughout the computer to the "0" state
3) Setting the Initial Start FF K054/055 to "1"
4) Operation stops

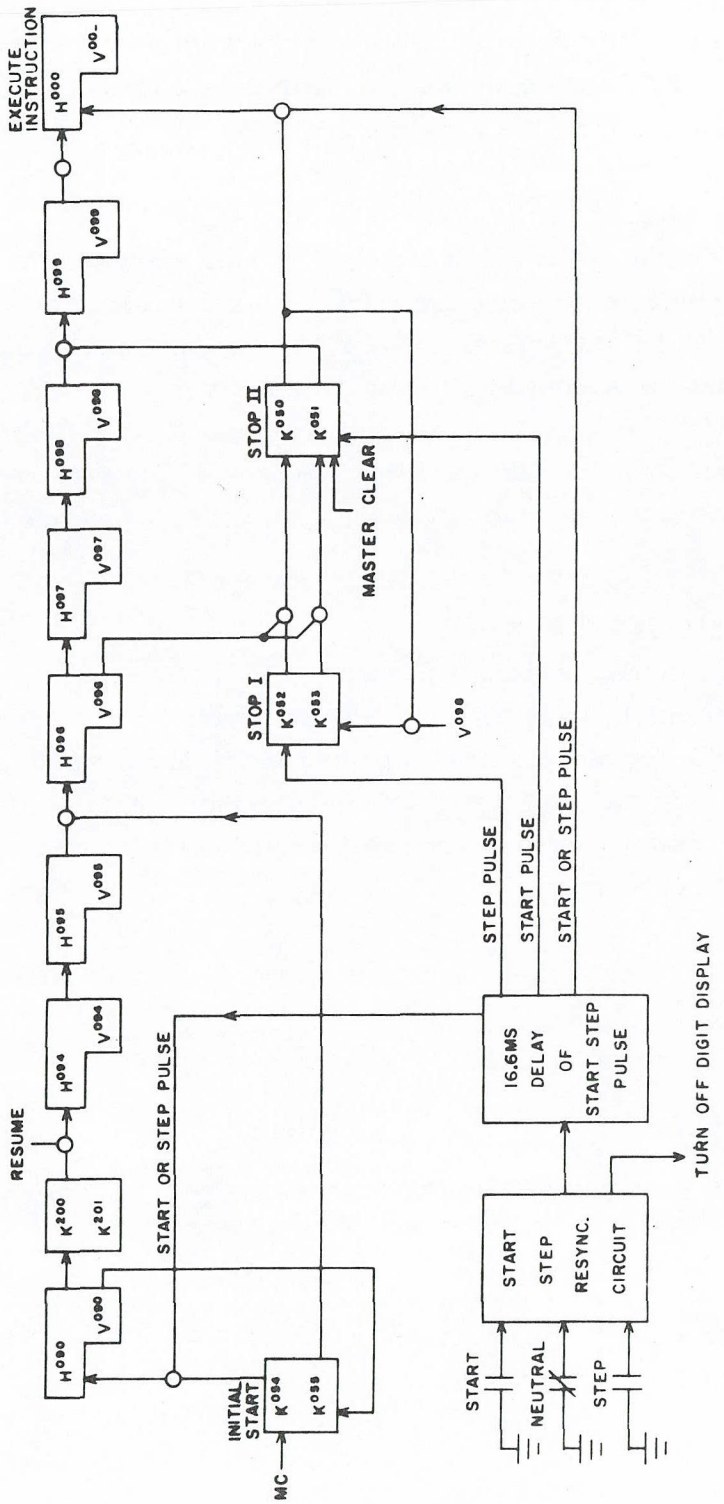


Figure 2-29. Connection of Start-Step Switch to RNI

A computer master clear does not alter any quantity placed in core storage before the clear. Not every FF in the computer is cleared by a Master Clear; the Secondary registers and many Control FFs are not affected.

External Master Clear

When the Clear switch is placed in the Up position, a Master Clear signal is sent to all equipments connected to the computer. An external Master Clear is not selective. Within each equipment the Master Clear signal forces the critical registers and control FFs to the "0" state and all operation stops in every equipment. (Certain FFs may be set to "1" by Master Clear.) Following the Master Clear, each equipment is ready for any selected mode of operation. However, an External Master Clear does not in any way alter the information recorded on the storage medium of the external equipment.

SELECTIVE JUMP SWITCHES

The three Selective Jump switches (figure 2-30) provide the manual conditions for instruction 75, normal jumps with j = 1-5 and return jumps with j = 5-7. Each switch has a Resynchronizing circuit, the outputs of which are sampled at the inputs of F910. The switches lock in Up position and are momentary in Down position. Although either position meets the manual condition for jumping, normally only the Up (locking) position is used.

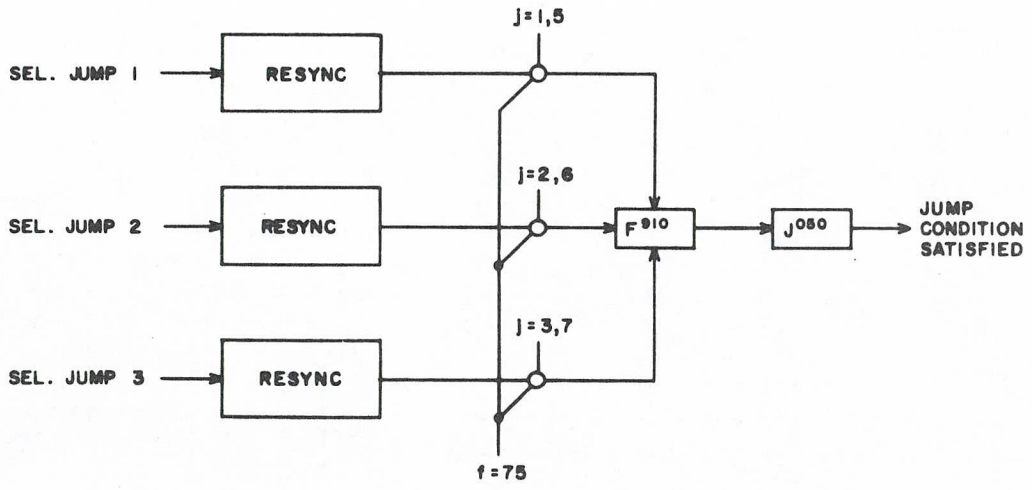


Figure 2-30. Sampling Selective Jump Conditions

SELECTIVE STOP SWITCHES

The three Selective Stop switches function in a similar manner for instruction 76. Stopping operation by execution of 76 with j = 1-3, 5-6, is conditioned by the switch positions. Each switch has a resynchronizing circuit, the outputs of which are combined with the translations of j. The switches lock in Up position and are mementary in Down position. Either position meets the manual condition for jumping, but ordinarily only the Up position is used.

STORAGE TEST SWITCHES

There are two storage test switches, Mode and Margin. The switches lock in both Up and Down positions. The switches are in the Neutral position during normal operation.

The Mode switch in Up position provides for repeatedly reading and executing the same pair of instructions by disabling the Advance P command normally initiated by a full RNI. Mode switch Up does not prevent a jump instruction from doing a $U^2 \rightarrow P$ transfer. In Down position the Mode switch provides for sweeping through (successively) all the addresses in storage. As shown in figure 2-12, during this operation, instructions are not executed; only the RNI sequence is performed. A full RNI, which initiates a storage reference, is performed and followed immediately by a half RNI, which does not initiate a storage reference.

Another full RNI follows immediately. Since this next full RNI advances P, the storage address referred to is one greater than that referred to in the preceding full RNI.

In the Sweep mode storage references are initiated at the high speed fixed rate. Sweeping through storage may also be accomplished by stepping so that the operator can view the contents of each address as it is read. The upper 24 bits of a word appear in the Program Control register at the console when the full RNI is performed; the succeeding half RNI transfers the lower 24 bits into the Program Control register for visual inspection.

The Margin switch varies the bias applied to storage sense amplifiers and is used for maintenance purposes only. It should be in Neutral position at all other times.

MASTER CLOCK

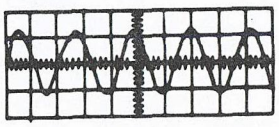
The master clock operates all the time that power is applied to the computer; it provides the timing pulses used throughout the computer. Directly or indirectly the timing of all signals is determined by this clock. The master clock system consists of over 100 interconnected oscillators, each contained on a type 01A card. The system sometimes employs single inverters as slaves to provide additional outputs from the oscillators.

Each oscillator operates at 2.5 megacycles and provides ten sine wave outputs. Five of the outputs are 180 degrees out of phase with the remaining five. One set is designated "even" the other "odd". The circuits which receive the sine wave outputs convert them into rectangular waves (figure 2-31). The asymmetrical form of the rectangular wave (part B) is due to the bias used in clipping the sine wave peaks. Only the negative, smaller, portion of the rectangular wave is effective in gating; thus the clock pulses used in the computer are 0.2 usec in duration.

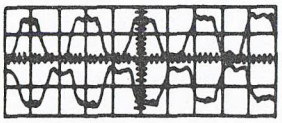
The circuits of the master clock appear in the File of Equations with the symbol C---. Each oscillator has two symbols with consecutive superscripts. The first has an even digit in the third position, the second an odd digit; e.g., C222 for the even clock pulse and C223 for the odd clock pulse.

Synchronization is achieved by connecting pin one (even clock) of each clock card on a chassis to pin one of all other clock cards on that chassis. The same is done with pin 7 (odd clock). Each chassis contains a 00 card through which the system of oscillators on that chassis is connected to the other seven chassis. The 00 card can be removed from its jack to isolate the clock system of any chassis for maintenance purposes.

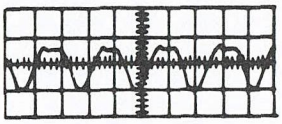
Oscillators are loaded as symmetrically as possible. In cases where a great many outputs of one phase are required, symmetrical loading is maintained by the use of single inverter slaves (N9--) which increase the available outputs.



A. Oscillator Test Point



C. Comparison of Phases at the Output of Slave Inverter



B. Oscillator Output Pin

Vertical: A and B, 5v/cm; C, 2v/cm
Sweep: 0.2 usec/cm

Figure 2-31. Master Clock Oscillator Waveforms

CHAPTER 3
ARITHMETIC SECTION

The arithmetic section of the 1604-A computer is composed of registers A, Q and X (figure 3-1). Procedures followed in the arithmetic section are derived from the fundamental logic processes. The arithmetic is one's complement binary.

ARITHMETIC REGISTERS

ACCUMULATOR

The principal Arithmetic register, A, forms the results of arithmetic operations by the process of accumulation. Some of the more important functions of A are:

1) Arithmetic operation - A initially holds one of the operands in addition, subtraction, multiplication and division. The result is usually held in A.

2) Shifting - A may be shifted separately or in conjunction with Q to the right or left. Right shifting is open-ended with the lowest bits discarded and sign extended.

3) Control for conditional instructions - A holds the word which conditions Jump and Search instructions.

The A register is composed of two major functional parts: (1) two ranks (A^1 and A^2) of 48 flip-flops each, which are used to hold the operand; and (2) the borrow pyramid, a network for sensing and generating the borrows required in subtracting the complement of X (X') from A.

Normally, rank A^1 follows rank A^2 unconditionally (figure 3-2). The signal $A^1 \rightarrow A^2$ is produced each even phase time as long as the separate storage facilities of A^2 are not required. For example, the commands Add X^2 to A^1 and $Q^2 \rightarrow A^1$ act on A^2 as well, duplicating in A^2 the data transferred to A^1. Similarly, the command Clear A is applied only to A^1, but its effect is felt by A^2. A^2 receives the next higher or lower bits in the right or left shift operations and supplies the quantity for the transfer $A \rightarrow Q$.

Only the Q and X registers communicate with A (figure 3-1). The transmission from Q to A is accomplished in the ordinary manner. Transmission of X^2 to A^1 uses the add path; A^1 is first cleared, then X^2 is added to A^1.

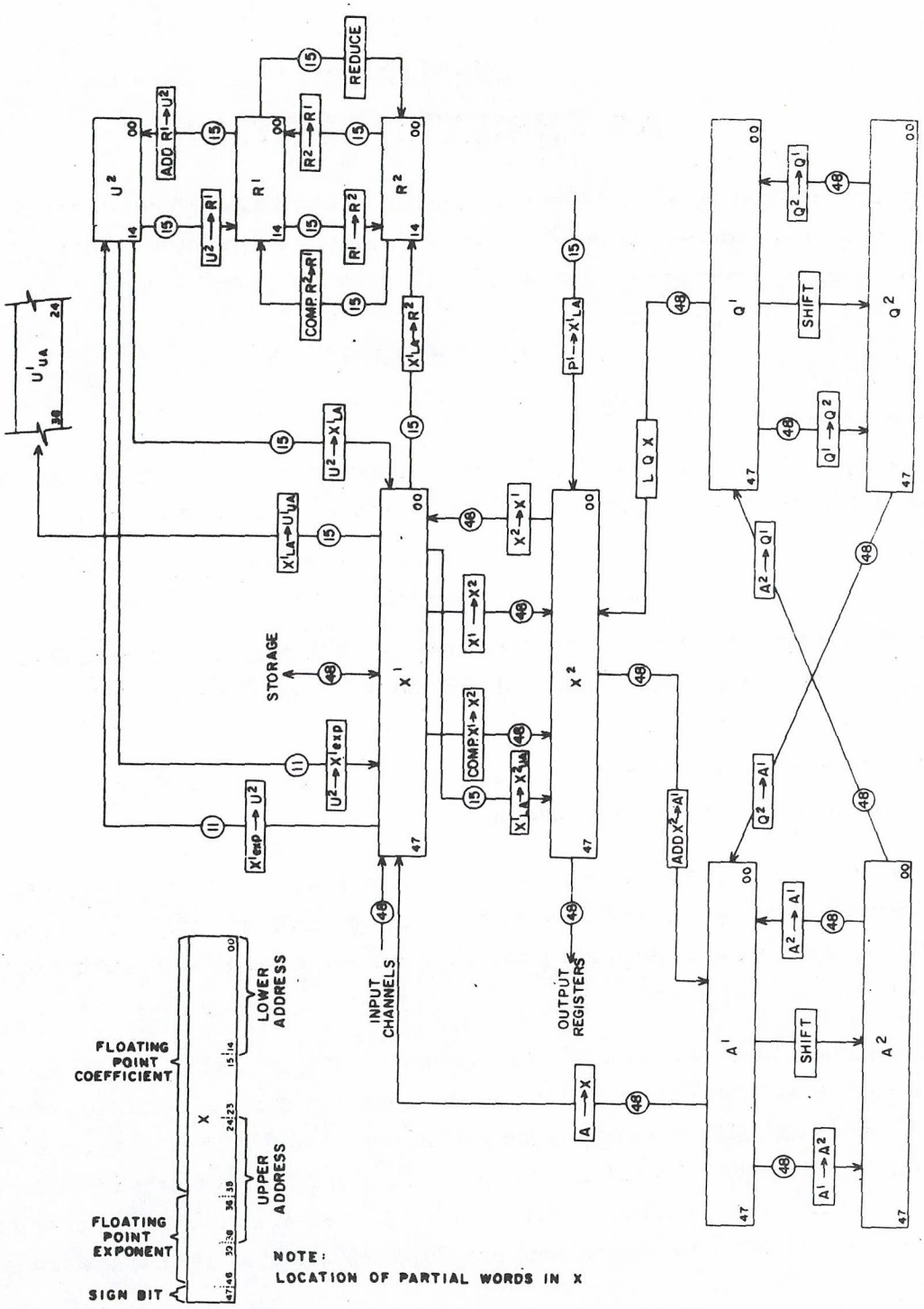


Figure 3-1. Over-all Block Diagram of Arithmetic Section

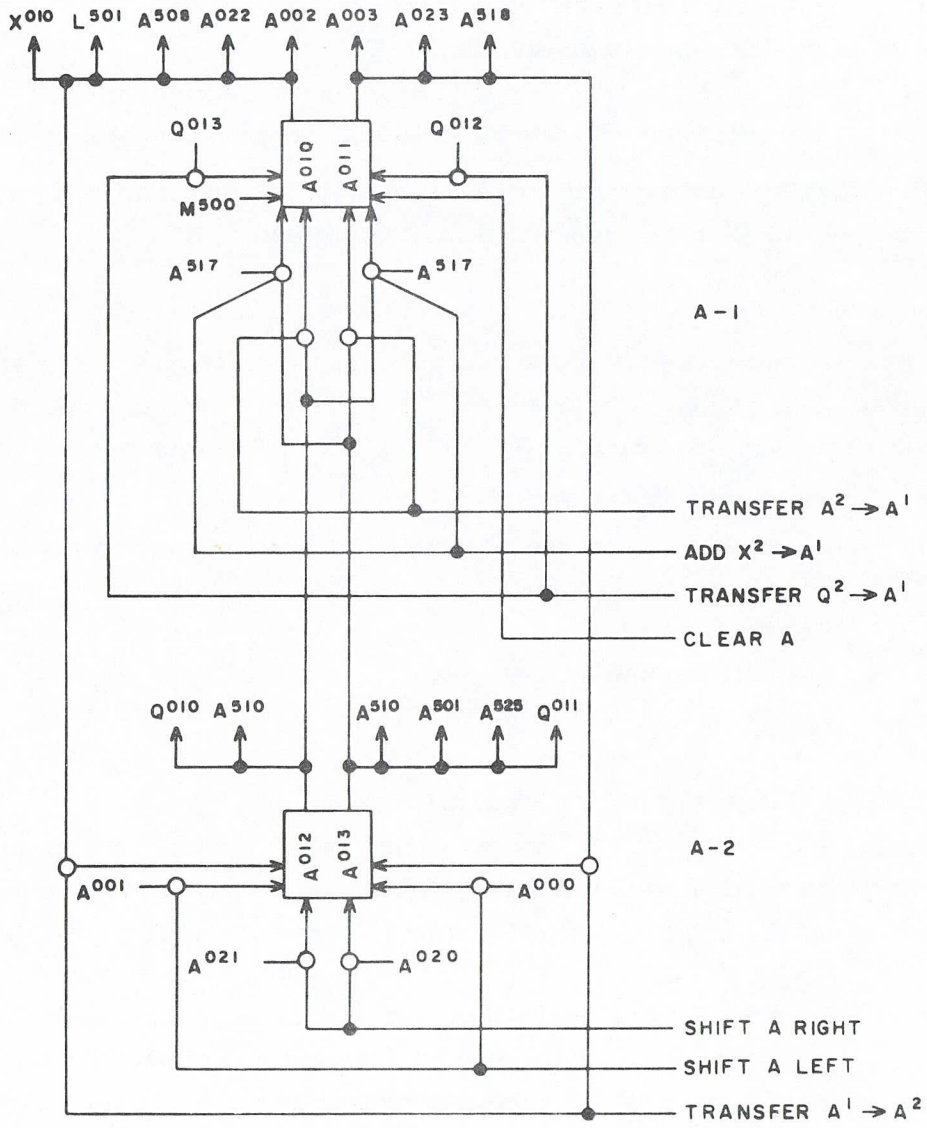


Figure 3-2. Typical Stage of A Register

Q REGISTER

The auxiliary Arithmetic register, Q, is composed of 48 stages of double-rank FF storage. The two ranks are independent. The signals causing transmission between them are generated conditionally. The principal functions of Q are:

1) Temporary storage for contents of A
2) Double-length register, AQ or QA
3) Right or left shift, separately or in conjunction with A
4) Multiplication, division and logical product operations (masking)

Q communicates directly with the A register only. In forming logical products the clear outputs of Q^1 are transmitted to clear inputs of X^1.

X REGISTER

The communication center of the computer is the Exchange X register which is composed of 48 stages of double-rank FF storage. Communication between the ranks is dependent upon the Transfer $X^1 \rightarrow X^2$ or $X^2 \rightarrow X^1$ commands. The principal operations involving the X register are:

1) Communication between various sections of the computer
2) Arithmetic operations
3) Complementing
4) Logical products
5) Assembly and disassembly of floating point words

REGISTERS USED IN ARITHMETIC OPERATIONS

Certain arithmetic operations use elements of the control section to effect completion. For Multiply, Divide and Shift instructions a control quantity is placed in R to govern the operation; Floating-Point instructions use the U and R registers.

Multiplication and division operations proceed as repetitions of a two-part step, adding and shifting for multiplication or subtracting and shifting for division. The R register controls the repetitions and is preset to indicate the number required. After each step, R is reduced by one. When R = 0, the operation concludes.

BASIC OPERATIONS

BINARY ARITHMETIC

Computers operate faster and more efficiently by using the binary number system. Only two digits, "0" and "1", are used in this system. Two voltage levels are sufficient to encode data presented in binary form. A -3.0v represents a logical "1"; a -0.5v represents a logical "0".

A binary number uses the digit 2 as its radix in the same manner that a decimal number uses 10 (see table 3-1). The decimal number 653 is:

$$6 \times 10^2 + 5 \times 10^1 + 3 \times 10^0 = 600 + 50 + 3$$

The binary number 1011 represents:

$$1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 8 + 0 + 2 + 1$$

which is equivalent to decimal 11.

TABLE 3-1. DECIMAL AND BINARY EQUIVALENTS

Decimal	Binary	Decimal	Binary
0	00000	9	01001
1	00001	10	01010
2	00010	11	01011
3	00011	12	01100
4	00100	13	01101
5	00101	14	01110
6	00110	15	01111
7	00111	16	10000
8	01000		

Binary numbers are added according to the following rules:

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 0 with a carry of 1

The addition of two binary numbers proceeds as follows (the decimal equivalents verify the result):

Augent	0111	7
Addend	+0100	+4
Partial Sum	0011	
Carry	1	
Sum	1011	11

Subtraction may be performed as an addition. Decimal examples:

8	minuend	or	8	minuend
-6	subtrahend		+4	10's complement of subtrahend
2	difference		2	difference - omit carry

Subtraction performed by adding the complement:

$$8 - 6 =$$
$$8 + (10 - 6) - 10 =$$
$$8 + 4 - 10$$

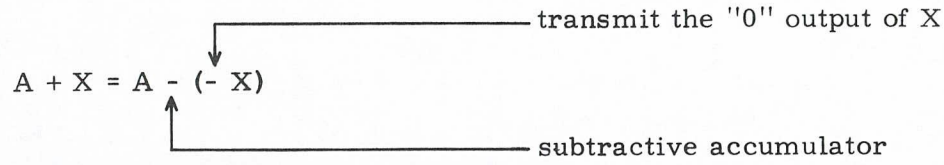
Omitting the carry has the effect of reducing the result by 10.

To complement a binary number (one's complement), subtract each bit of the number from 1.

1111	
-1001	9
0110	one's complement of 9

The one's complement of a binary number may also be formed by substituting "1's" for "0's" and "0's" for "1's" in the number.

In subtracting by the complement method the computer need perform only one basic arithmetic operation, addition. The Arithmetic circuits of the accumulator are subtractive. The following equation shows that it is possible to add by a subtractive process:



The equation states that the quantity in X is added to the quantity in A by subtracting the complement of X, usually written X', from A. Even though A is subtractive, the net result in additive.

Generating borrows is fundamental in subtraction. A borrow in a subtractive accumulator compares to a carry in an additive accumulator; a borrow backs a stage one count, a carry advances a stage one count. It is this borrow feature which makes A subtractive rather than additive.

ADDITION

The basic operation of addition is involved in most other operations. Both the accumulator and the X register are used in addition. In adding X to A the result is governed by two conditions: (1) what must be done to A_n to produce the difference without borrows of A^n and X'_n; and (2) whether the arithmetic difference of A_{n-1} and X'_{n-1} requires that a borrow be made from A_n.

Sensing For First Conditon

Whenever "0" is subtracted from a minuend of either "0" or "1" the difference has the same value as the minuend; conversely, whenever "1" is subtracted from a minuend of either "0" or "1" the difference is the reverse of the minuend. Where "1" is subtracted from "0" a borrow from the next higher order bit is required. The following rules (table 3-2) summarize binary subtraction: (1) whenever the subtrahend is "0", the difference is the same as the minuend, and (2) whenever the subtrahend is "1", the difference is the reverse of the minuend, with a borrow required from the next higher order stage when the minuend is "0". If the borrow is disregarded, the difference column is the logical or bit-by-bit difference.

In addition of X to A, the minuend is converted into the logical difference of the minuend and the subtrahend by complementing the bits of the minuend for which the corresponding bits of the subtrahend are "1" (table 3-2). The minuend is in A and the subtrahend is the complement of the quantity in X. The logical difference of A and X is formed by toggling each bit of A for which the corresponding bit of X is "0".

Sensing For Second Condition

The second condition in the addition of X to A indicates what borrows must be made to form, not merely the logical difference, but the arithmetic difference of A and X. It is

necessary to borrow from stage A_n when both A_{n-1} and X_{n-1} are initially "0"; a borrow from A_n means that the logical difference of A_{n-1} and X_{n-1} is "0". This follows, since the complement of X is subtracted from A. When X_{n-1} is a "0", its complement is "1", and it is the complement of X_{n-1} which is subtracted from A_{n-1}.

TABLE 3-2. BINARY SUBTRACTION

Minuend A		Subtrahend X		Difference
1	-	0	=	1
1	-	1	=	0
0	-	0	=	0
0	-	1	=	1*

*borrow required from next higher-order bit.

Combining The Two Conditions

For each stage of A the two conditions for toggling are combined to gate the command. Add X to A. If this AND is satisfied for A_n, the A_n is toggled and A holds the sum of A and X.

Since the two conditions are independent, it is possible for both of them to be satisfied at the same time for a given stage. If this is the case, the A_n should not be toggled at all since toggling a stage twice restores it to the initial state. If neither is satisfied, A_n is not to be toggled. The exclusive OR combination of the two conditions is required for gating the Add command. Thus, A_n is to be toggled if: (1) X_n is "0" and no borrow is required from A_n, (2) X_n is "1" and a borrow is required from A_n.

If a borrow is required from A_n and A_n is "0", then a borrow is required from A_{n+1}. If A_{n+1} is a "0", then a borrow is required from A_{n+2}, and so on. The propagation of a borrow continues until a stage is reached which is "1". A borrow generated in A_{47} is made from A_{00}. This is the end-around borrow which makes the accumulator a one's complement arithmetic device. Note especially that a borrow may be required from A_n which is not toggled by the Add command. This situation exists whenever X_n is "0".

Example:							
	Augend, in A initially	0	1	0	1	+	5
	Addend, in X initially	0	0	1	1	+	3
	Logical Difference of A and X^1	1	0	0	1	←	
	Stages to be Toggled	T	T	T			
	Sum of A and X (produced by toggling A_i in each bit with a T)	1	0	0	0	+	8

A 4-bit system is used for convenience; however, the 48-bit system of the accumulator is exactly the same. The logical difference of A initial (A_i) and X_i is formed by complementing each bit of A_i where the corresponding bit of X_i is "0"; the other bits of the logical difference are the same as those of A_i. The determination of borrows can be made with the aid of this logical difference. A borrow is generated in any stage A_n where A_n and X_n are both "0". The borrow is represented by an arrow which points to stage A_{n+1} from which the borrow is made. In the case where the borrow points to a stage of the logical difference of A and X^1 that is "0", another borrow is required.

Borrow Pyramid

The borrow pyramid senses the stages of A from which borrows are required and sends signals to toggle the necessary stages. Borrows are sensed by inverters A^{--0} through A^{--6}. The A^{--7} inverter senses where A must be toggled. The borrow is sensed by sampling A^2 and X^2. The result is combined with X^2 as the input to the toggle control. The latter then determines what stages of A^1 are to be toggled by the Add X^2 to A^1 command. Appendix A contains a detailed examination of the borrow pyramid.

Figure 3-3 shows the relation of the pyramid to the A and X registers. This portion of the accumulator determines whether the conditions exist which require A_n to be toggled. These conditions are: X_n is "0" and no borrow is required from A_n; and X_n is "1" and a borrow is required from A_n.

For the full additions, the Partial Add In A FF is cleared. By setting it, the borrow sensing is disabled. This permits the Add command to toggle X^2 into A^1. Since this is an Add without carries it is called a Partial Add.

SUBTRACTION

In subtraction the minuend is located in A; the subtrahend is in X. It is necessary to first complement X^1 with the command Complement $X^1 \rightarrow X^2$. X' is subtracted from A.

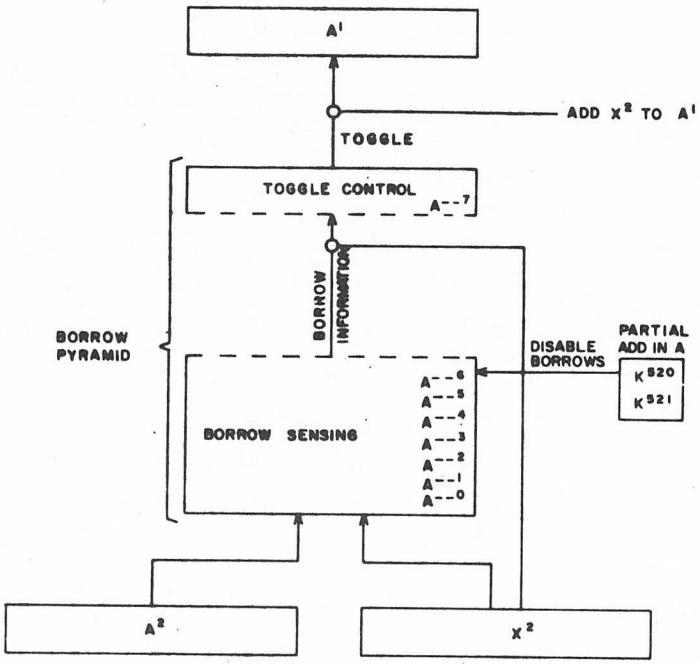


Figure 3-3. Relation of Borrow Pyramid to A and X Registers

Since X is complemented twice during the operation, A - X' is reduced to A - X:

$$A - X = A - (\bar{-}\ \bar{X})$$

transmit the "0" output of X

complement X

subtractive accumulator

Subtraction is accomplished in three steps: (1) complement X; (2) sense borrows by means of the logical difference of A and X; and (3) form the arithmetic difference by toggling.

SHIFTING

Fundamental to the arithmetic operations are the shifting properties of A and Q. These registers can be shifted either to the right or to the left, separately or as a combined 96-bit unit. Registers are shifted one stage at a time.

The transmission paths for shifting are always from the upper rank of the register to the next stage in the lower rank. Direction depends upon the type of shift, right or left.

A subsequent command places the shifted word into the upper rank. When A and Q shift as a unit, A_{00} is connected directly to Q_{47}.

The right shifts are all open-ended; as a word is shifted to the right the least significant bit is lost after each stage shift. The sign bit is maintained in the most significant stage and is extended to the right for as many stages as the word is shifted.

All left shifts are circular. The left Shift sequence connects the highest stage directly to the lowest stage. Thus the content of the sign bit stage appears in the least significant bit stage after each stage shift, and all other positions move one place to the left. The shift operation can be called for directly by instruction and occurs as a minor sequence within the Mulitply, Divide, Floating Point and Scale instructions.

Shift Instructions

The following instructions are used in programs to specify a shift operation:

01	Shift A Right	05	Shift A Left
02	Shift Q Right	06	Shift Q Left
03	Shift AQ Right	07	Shift AQ Left

The number of places to be shifted is designated by the operand address portion of the instruction. The shift count can have any value through 127; however, 96, the number of stages in the double-length register AQ, should be the largest significant count. (The lowest seven bit positions of the execution address are required to encode 96, since the capacity of these seven bit positions is 127. Numbers in the range 97-127 are acceptable counts.) Any "1's" in bit positions greater than 2^6 cause a fault indication which may be sensed by the External Function instruction.

The Zero Address sequence initiates shifting by setting one or more of the four FFs (K310 - K317) in the Shift Control circuits (figure 3-4). These FFs enable the Shift and Reduct R commands to occur every even phase. During the odd phase the R^2 to R^1 and A^2 to A^1 or Q^2 to Q^1 commands occur to prepare for the next Shift and Reduce commands.

Shift Exit FF (K320/321) is set to record that one of the Shift instructions is being executed. When R = 1 the shift FFs (K310 - K317) are cleared on the even phase. This terminates shifting at the correct point, since a Shift command is being executed at the same time. The Reduce R command accompanying this last Shift command reduces R from "1" to "0". This signal indicating R = 0 (even phase) allows the exit to be taken.

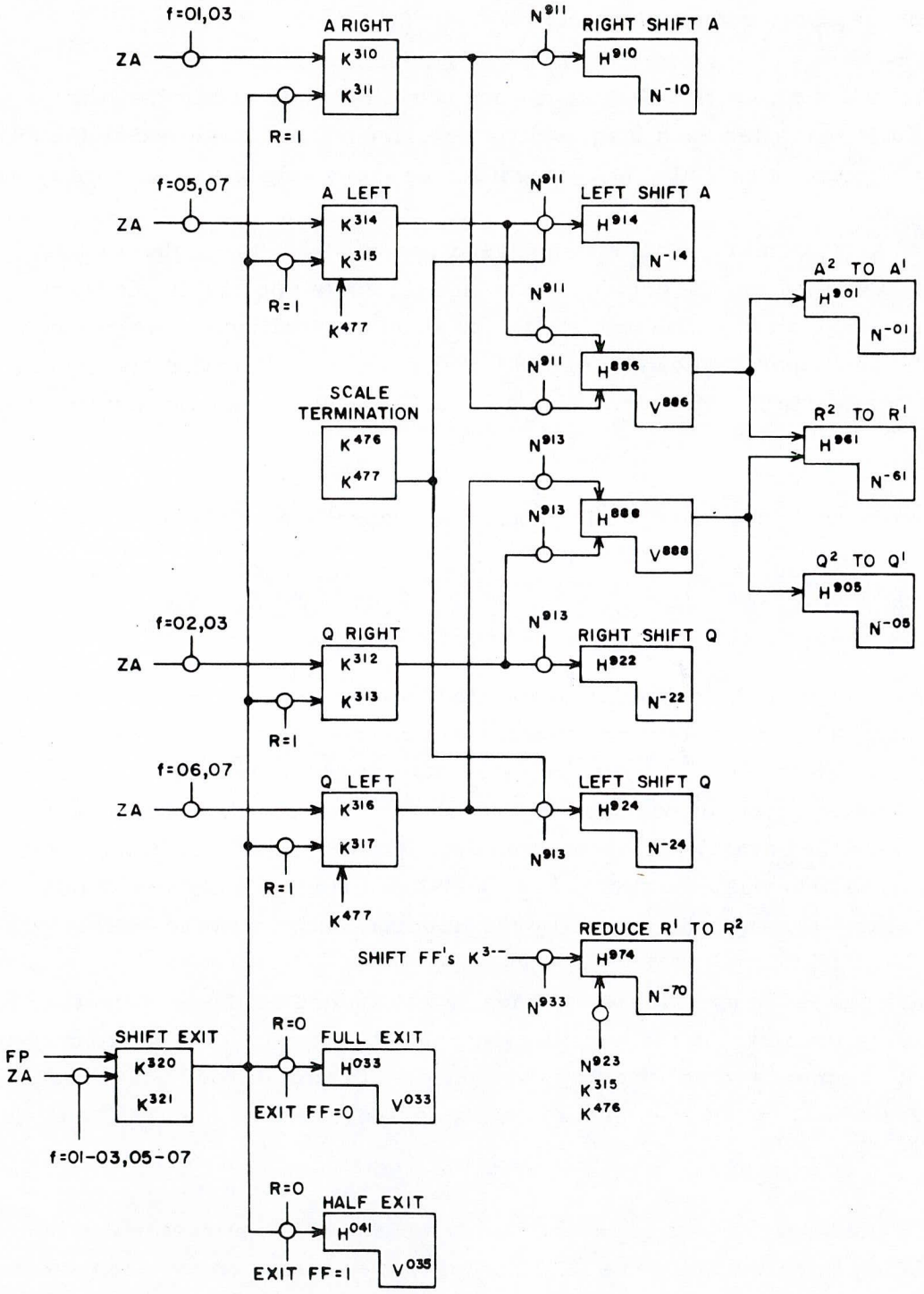


Figure 3-4. Shift Control

Shifting in Multiplication and Division

In multiplication, the AQ Right Shift positions the current sum of partial products in AQ for the formation of the next partial product. The shift also positions the next multiplier bit in Q_{00} to determine the action required for generating the next partial product. The number of shifts depends on the type of multiplication (integer, fractional or floating-point). The control for the number of shifts is preset in R by the instruction.

In division, left shifts position the partial dividend as a minuend for the subtraction of the divisor to determine the individual bits of the quotient. The type of division being performed determines the number of shifts.

Shifting in Floating Point Instructions

Floating point operations employ shifting for determining the product or quotient of two floating point operands.

Shifting in Scale Instructions

The Scale instructions 34 and 35 are essentially Left Shift instructions. The contents of A or AQ are shifted to the left until R = 0, as in Normal Shift instructions; or the shift concludes when a "1" appears in the stage immediately to the right of the sign bit. The A Left FF or both the A Left and Q Left FFs are set to begin shifting. Shifting terminates by clearing these FFs when R = 0 or when scaling is achieved.

ITERATIVE SEQUENCE

The longer arithmetic operations of multiplication and division and all floating point operations are performed by the Iterative sequence. For any of the instruction (24-33) using the Iterative sequence, one of the operands is initially in A as the result of a previous instruction. The sequence begins at H601 (figure 3-5). The first part of the chain acquires the second operand from storage. Various portions of the chain are then used as shown in figure 3-5, depending upon the instruction.

MULTIPLICATION

The Multiplication instructions, (24) Multiply Integer, (26) Multiply Fractional and (32) Floating Multiply, form the product of two operands. The multiplier is contained in A and the multiplicand in the address specified by the instruction. The result of the 24 and 26 instructions is a 96-bit quantity held in QA for 24 and in AQ for 26.

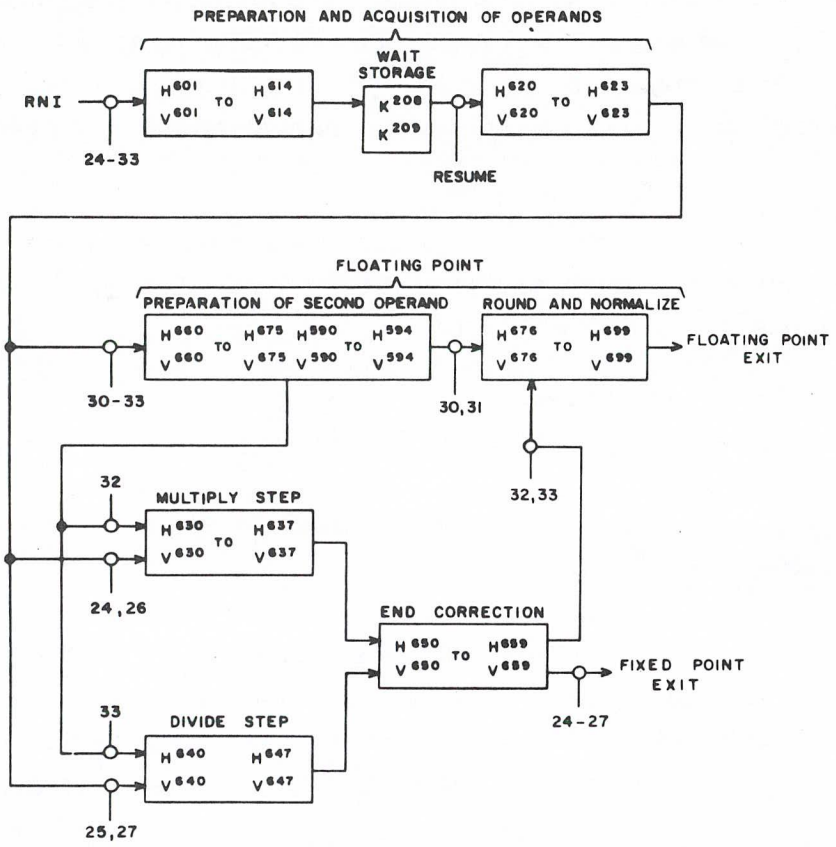


Figure 3-5. Iterative Sequence

The computer performs multiplication by repeated additions and shifts. The multiplicand in storage is transferred to X and added to the partial product being developed in AQ. The multiplier digits are transferred from A to Q and inspected, beginning with the least significant bit. The magnitude of each multiplier bit determines whether or not the multiplicand is to be added to the current sum of partial products during the particular step for which it is the control.

Multiplication is always performed on a bit-by-bit basis as carries do not result from multiplication; the product of any two bits is always a single bit. Binary multiplication proceeds according to the following rules:

0 x 0 = 0	1 x 0 = 0
0 x 1 = 0	1 x 1 = 1

Example 1: An example of binary multiplication follows.

	Decimal		Binary	
multiplicand	14		1110	
multiplier	12		1100	
partial products	28		0000	
	14	shift one place left	0000	shifts to place digits in proper columns
product	168		1110	
			1110	
			10101000	

The computer determines the running subtotal of the partial products. Rather than shift the partial product to the left to position it correctly, the computer right shifts the summation of the partial products one place before the next addition is made. When the multiplier bit is "1", the multiplicand is added to the running total and the results are shifted to the right one place. When the multiplier bit is a "0", the partial product subtotal is shifted to the right (in effect, the quantity has been multiplied by 10_2).

After each multiplier bit in Q is considered, it is discarded and Q is shifted right one place. This always positions the current multiplier bit in Q_{00}. The upper stages of Q become available to receive the lower order digits of the product as it develops and is right shifted in A.

Example 2: Register contains 4 bits and sign (01110).

	A	Q		
Initial	0 1 1 0 0	0 0 0 0 0		
Step 1	0 0 0 0 0	0 1 1 0 0	Interchange A and Q	
Step 2	0 0 0 0 0	0 0 1 1 0	Add Q_{00} times X to A.	Shift right one.
Step 3	0 0 0 0 0	0 0 0 1 1	Add Q_{00} times X to A.	Shift right one.
Step 4	0 0 1 1 1	0 0 0 0 1	Add Q_{00} times X to A.	Shift right one.

	A	Q	
Step 5	0 1 0 1 0	1 0 0 0 0	Add Q_{00} times X to A. Shift right one.
Step 5	1 0 0 0 0	0 1 0 1 0	Express lower bits of the product in A, upper bits in Q.

Multiplication by the Iterative sequence consists of: initial sign correction, multiplication phase, final sign correction.

During initial sign correction both the multiplier and multiplicand are made positive, if they were not. If the product is to be negative, the Sign Record FF is set. If the signs are alike the FF is cleared. Later during final sign correction, the Sign Record FF indicates whether the product obtained from the multiplication phase must be negative (complementing required).

Following the initial sign correction, the sequence generates a control quantity which determines the number of steps necessary to conclude the multiplication. This number, which varies with the type of multiplication, is set in I^2 and transmitted to R. As each step is completed R is reduced by one. When R = 0, the sequence terminates.

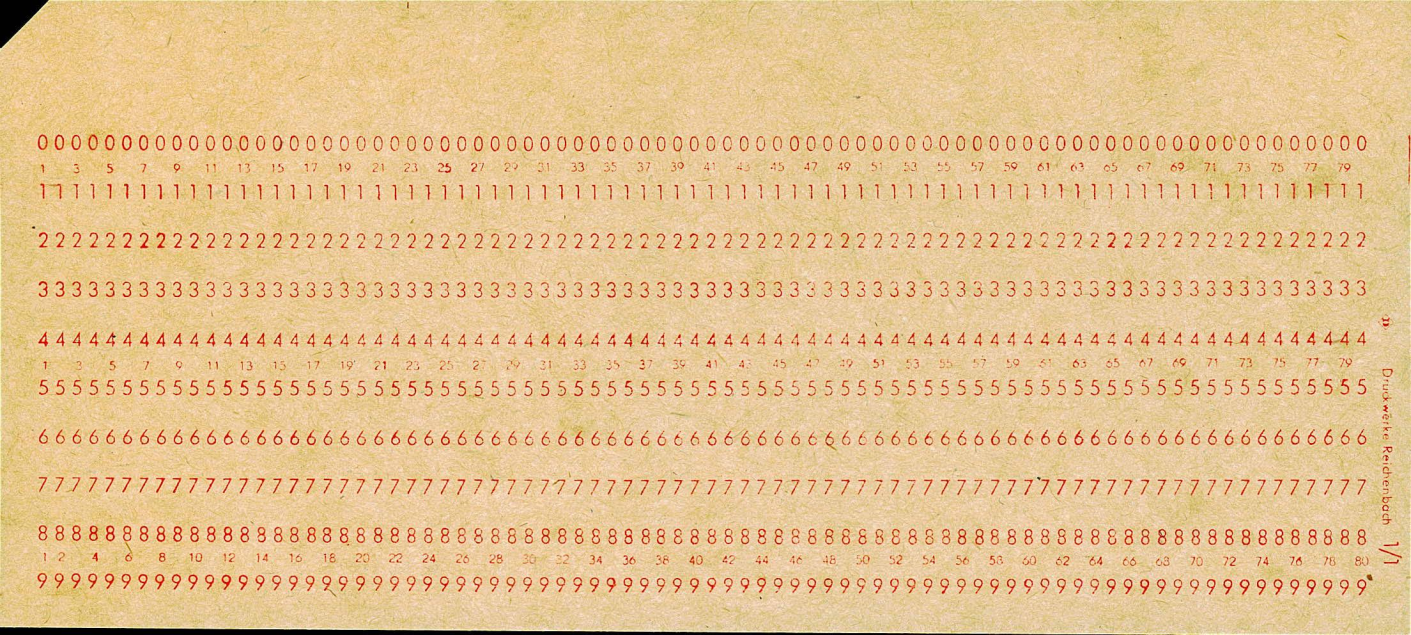
The step control quantities set in I^2 for the Multiplication instruction are:

48 for instruction 24, Multiply Integer
47 for instruction 26, Multiply Fractional
36 for instruction 32, Floating Multiply

The difference in the control quantities provides for the proper positioning of the product in Q and A (A register for Floating Multiply). Integer products are positioned with the binary point at the right side of A; the fractional products are positioned with the binary point located immediately to the right of the sign bit, A_{47}. For floating point products the point is just left of stage A_{35}.

Determination of a product with operands packed in floating point format proceeds in a manner similar to that described herein. However, as floating point multiplicands and multipliers contain only 36 bits, multiplication is accomplished in fewer steps. The product is rounded to 36 bits.

The multiplication phase is accomplished by repeating the multiply step (figure 3-6) the number of times specified by the control quantity. Each repetition forms the partial


0 0 0

product of a bit of the multiplier and the multiplicand and adds this to the running sum of partial products.

The first multiply step shifts AQ right. This disposes of the multiplier bit in Q; however, the effect of Q_{00} is maintained in slaves long enough to determine the path for the first loop. If Q_{00} is a "0", the short loop is selected; if Q_{00} is a "1", the long loop is selected. The short loop right shifts the partial product in AQ; this has the effect of multiplication by a "0" bit. The long loop first adds the multiplicand X to the sum of partial products in A and then initiates a shift AQ right, multiplying one bit. R is reduced by one at each shift. When R = 0 the product accumulation is terminated.

The product as assembled is a positive number, the sign is "0", and its bits are not complemented. If the product is to be expressed as a negative number; that is, if the operands were initially of opposite signs, AQ is complemented. In the case of 24, Integer Multiply, A and Q are interchanged to place the least significant part of the product in A.

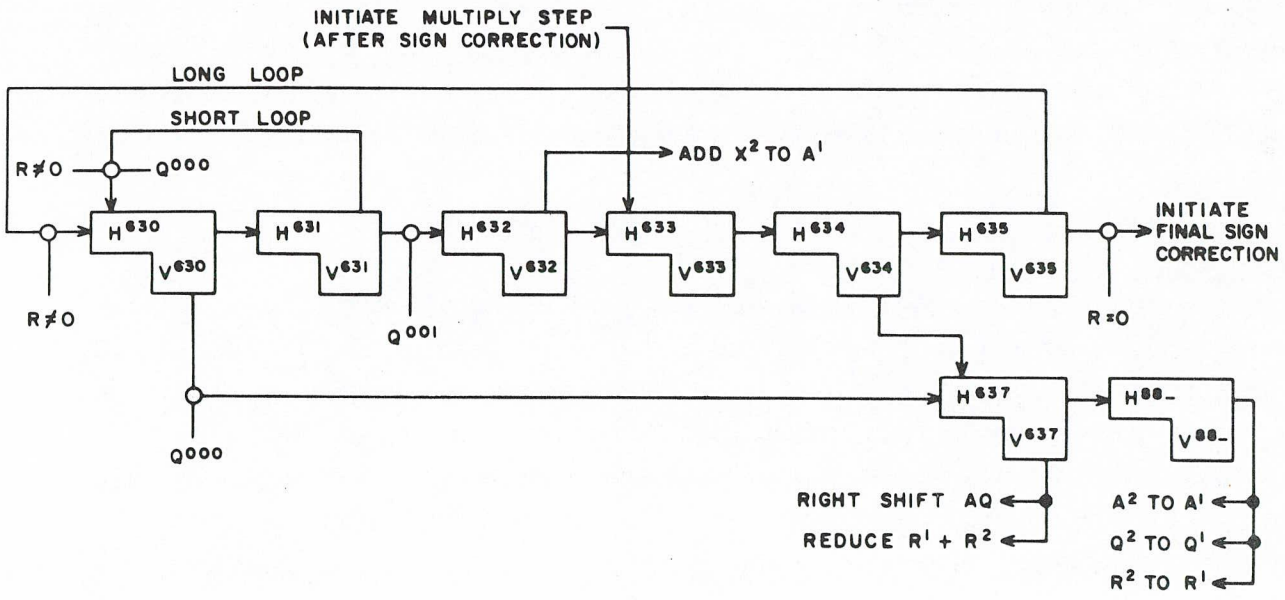


Figure 3-6. Multiply Step

DIVISION

For the Divide instructions, (25) Divide Integer, (27) Divide Fractional, and (33) Floating Divide, the dividend is in the A and Q registers; the divisor is in storage at an address specified by the instruction. Division is executed in three phases: initial sign correction, division phase, and final sign correction. An example is illustrated below:

	Decimal				Binary		
		14	Quotient				1110
Divisor	13	⟌185	Dividend	Divisor	1101	⟌10111001	
		13				1101	
		55	Partial Dividend			10100	
		52				1101	
		3	Remainder			1110	
						1101	
						11	

Instead of shifting the divisor right to position it for subtraction from the partial dividend, the computer shifts the partial dividend left, accomplishing the same purpose and permitting the arithmetic to be performed in the A register. The computer counts the number of shifts (number of quotient digits). After the correct number of counts the routine is terminated.

In programming a division the relationship of the size of the dividend and the divisor must be considered in order to express the quotient within the capacity of the quotient register.

Initial Sign Correction

Since division requires a positive divisor and dividend, initial sign correction complements either or both of these operands if they are negative. The quotient is positive if the signs of the operand are alike, negative if the signs are unlike. An indication of the sign of the quotient is stored in the Sign Record FF. During the final sign correction, this FF directs whether to complement the positive quotient that always results from the division phase. The sign of the dividend is recorded so that the remainder may be given the same sign.

For the division phase the dividend must be positioned in AQ (higher-order bits in A). During initial sign correction for 25, Divide Integer, A and Q are interchanged because the dividend is initially in QA for this instruction.

Division Phase

The divide step (figure 3-7) involves:

1) left shifting AQ one place
2) subtracting X (divisor) from A (partial dividend) if $A \geq X$
3) setting Q_{00} to "1" if subtraction was made; otherwise entering "0" in Q_{00}.

Two different control quantities govern the number of repetitions of the divide step, which performs each partial division and generates the individual quotient bits. The quantity is 48 for both Integer and Fractional Divide instructions (25 and 27); for the Floating Point Divide (33) it is 36. The control quantity is set in I^2 and then transmitted to the R register. Each repetition of the divide step reduces R by "1" until R = 0, the full quotient is transmitted to Q and the remainder is in A.

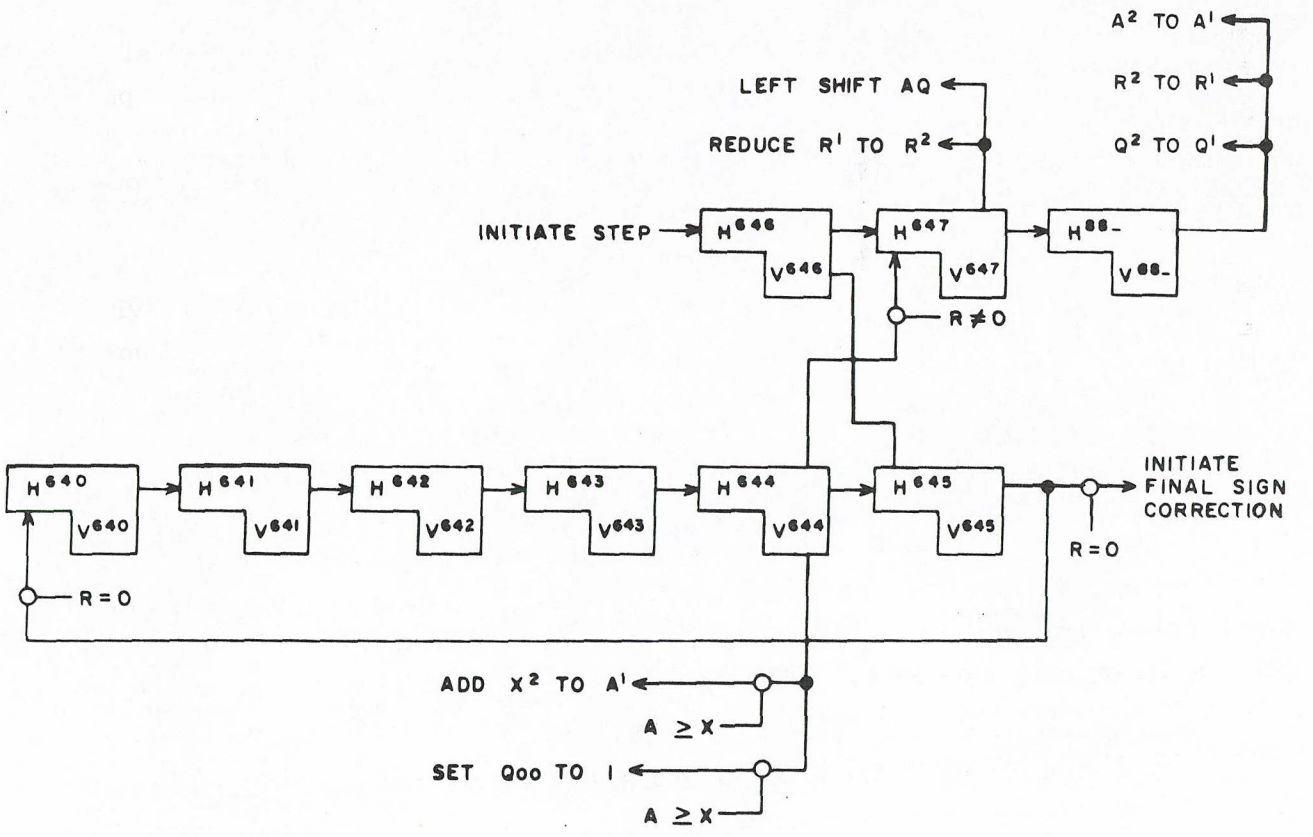


Figure 3-7. Divide Step

Subsequent commands position the quotient in A and the remainder in Q. An example follows.

Divisor		X 0 1 1 0 1		
		R = 5		
Dividend	AQ	0 0 1 0 1 1 1 0 0 1		
		0 1 0 1 1 1 0 0 1 0		Shift AQ left
		R = 4		
		1 0 1 1 1 0 0 1 0 0	A < X	Shift AQ left
		R = 3		
		0 1 0 1 0 0 0 1 0 1	A ≥ X	Subtract X from A Set Q^{00} to 1. Shift AQ left
		1 0 1 0 0 0 1 0 1 0		
		R = 2		
		0 0 1 1 1 0 1 0 1 1	A ≥ X	Subtract X from A Set Q^{00} to 1. Shift AQ left
		0 1 1 1 0 1 0 1 1 0		
		R = 1		
		0 0 0 0 1 1 0 1 1 1	A ≥ X	Subtract X from A Set Q^{00} to 1. Shift AQ left
Remainder Quotient		0 0 0 1 1 0 1 1 1 0		
		R = 0		Initiate Final Sign Correction

Final Sign Correction

The final sign correction phase:

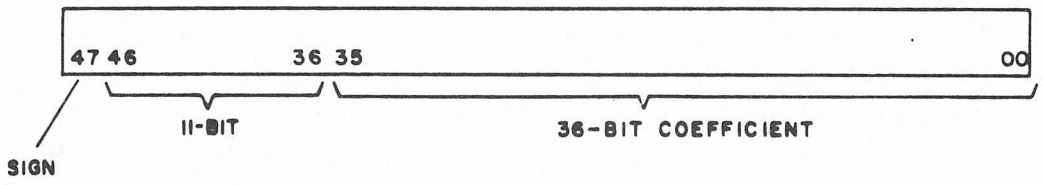
1) Senses for a divide fault.
2) Complements Q (the quotient) if the Sign Record FF is set (dividend and divisor initially have unlike signs).
3) Complements A (the remainder) if Dividend Sign FF is set (dividend initially negative).
4) Places quotient in A and remainder in Q.

A divide fault (Quotient register overflow) results when the quotient requires more than 47 bits. If a division involves a fault, after the first shift of the division phase, A is greater than X and a "1" is entered in Q_{00}. But if a divide fault is not involved, after the first shift A is less than X and a "0" is entered in Q_{00}. In either case, this bit appears in Q_{47} after the 47 remaining shifts of the division phase. At this point if Q is negative there is a fault; if Q is positive there is not a fault. Sensing a divide fault is sensing the sign of Q; the Divide Fault FF is set when Q is negative. The fault condition can be sensed later by the 74 External Function instruction.

After completing the final sign correction phase, instructions 25 and 27 exit. Instruction 33, Floating Divide, initiates the round and normalize part of the Iterative sequence.

FLOATING POINT

Any number can be described by the general expression kB^n, where k is a coefficient, B a base number, and the exponent n the power to which the base number is raised. The Floating Point mode of operation uses this means of expression by including in its operand format the coefficient, the sign of the number, and the exponent. The make-up of a floating point word is shown below:



Coefficient

The coefficient is made up of a 36-bit fraction located in the lower order positions of the floating point word and a single bit located in the conventional highest order bit position. The fraction has a value ranging from one-half to one (not including one). Negative number notation is one's complement.

Exponent

The exponent is an 11-bit number with a value from 0 to 2^{11} -1 (2047 in decimal, 3777 in octal). A bias of 2^{10} (1024 in decimal, 2000 in octal) is added to the true exponent when the floating point number is formed. A number with true exponent equal to zero appears as 2000 (octal). A number with exponent equal to 264 appears as 2264; a number

with exponent equal to minus 137 appears as 1641. As in algebra, positive exponents are used for integral numbers and negative for fractional numbers. When a number is negative, the exponent is included in the one's complement representation. Thus if the above examples of exponents were for negative numbers, they would appear in the floating point word as the bit-by-bit complement.

The bias used with the exponent makes floating point operation more versatile since floating point operands can be compared with one another in the normal Fixed Point mode. The transition from fixed point to floating point format is given below to illustrate the encoding of numbers by fixed point methods and the way this format lends itself to comparisons. For simplicity the fraction is limited to three bits and the exponent to four bits including sign. The exponent is in parentheses.

Fixed Point			Floating Point		
+4.0	=	0100×2^{0}	=	0.100×2^{3}	= 0(1011) . 100
+0.04	=	0.000100×2^{0}	=	0.100×2^{-3}	= 0(0100) . 100
-4.0	=	1011×2^{0}	=	1.011×2^{3}	= 1(0100) . 011
-0.04	=	1.111011×2^{0}	=	1.011×2^{-3}	= 0(1000) . 100
+0.4	=	0.100×2^{0}	=	0.100×2^{0}	= 0(1000) . 100
					or 0(0111) . 100

In a bit-by-bit comparison of two numbers the larger or more positive has a "1" in the first higher position for which the bits of the two numbers are unlike. The signs (first bit) are compared independently; a positive "0" quantity is always indicated as larger than a negative "1" quantity. Since 4.0 is greater than 0.04, the floating point form of 4.0 has a "1" in a more significant position than does 0.04. Thus fixed point comparison of the two floating point numbers yields the desired result. Similarly, -0.04 is greater than -4.0, since the first is less negative. Thus the floating point form of -0.04 contains a "1" in a higher position than the highest "1" of -4.0.

The two forms possible for +0.4 present a special case. The first is considered standard; that is, if all floating point operands that fall in the range of magnitude zero exponents are encoded in the positive form, then results of floating point operations which fall in this range are also represented by exponents of positive form. Unless the original operands have the negative representation of an exponent of magnitude zero, this form of exponent is not generated by the computer. These comments apply to the state of the 11-bit exponent before any complementing is performed if the entire number is negative.

Floating Point Operation

Any of the four Floating Point instructions is performed by the following steps. All operands should first be in floating point format.

1) Unpack
2) Execute coefficient and exponent arithmetic
3) Round
4) Normalize
5) Pack

The unpack (1) step involves one operand in the accumulator and one from storage. Each operand is separated into its exponent and coefficient. The coefficients are sent to the Arithmetic registers and the exponents to the Address Modification registers, U^2 and R. In unpacking, if the sign of the number is negative both the exponent and coefficient are complemented.

To illustrate actual floating point format several sample quantities encoded in binary are shown below.

Decimal	Sign \ Exponent	Coefficient
+1604.0	010 000 001 011	**110** 010 001 000 000 000 000 000 000 000 000 000
(octal 3104)		
-1604.0	101 111 110 100	001 101 110 111 111 111 111 111 111 111 111 111

(The negative character is indicated by "1" in the sign position of the fraction. The fraction is expressed in one's complement form.)

-.1604	110 000 000 010	010 110 111 100 000 000 011 010 000 000 111 010

In the arithmetic (2) step the exponent operations are handled in the U^2 and R registers. Coefficient arithmetic is performed as a fractional fixed point operation. In addition and subtraction the coefficients are aligned by shifting to make the exponents alike.

The round (3) step modifies the coefficient answer by adding "1" to A for positive answers or by subtracting "1" for negative answers. Rounding is necessary since the coefficient answer may contain more than 36 bits. The condition for rounding is inequality of the sign bits of A and Q. This means that the next lower significant bit to the right of the number in A is equal to or greater than one-half.

Coefficient arithmetic may yield rounded answers from zero to 2^{37}. The normalize (4) step brings this answer back to a fraction from one-half to one with the binary point to the left of the 36th bit. The final normalized number in A ranges from 2^{36} to $2^{37}-1$. Normalizing is performed by either a right shift or the required number of left shifts. The exponent is corrected for every shift. The residue in Q is not shifted.

The pack (5) step positions the final exponent and coefficient in A. If the sign of the answer is negative, both the coefficient and exponent are complemented. The exponent range is tested to determine overflow of the exponent and to set the Fault indicator.

Addition

The Floating Add instruction, 30, produces the sum of the floating point operands in A (augend) and address M (addend). In accordance with the floating point format, the coefficients and exponents of the operands are separated. Coefficient operations are performed in the A and X registers; the exponent operations are performed in the R and U^2 registers. The resultant exponent and coefficient are assembled into proper format in A. Basic steps in Floating Add are:

1) Enter augend in A by previous instruction; complement if negative.
2) Acquire addend from address M and place in X; complement if negative.
3) Compare exponents; save larger in U^2; difference in R.
4) Position coefficients so that one with smaller exponent is in A.
5) Shift AQ right by number of places indicated by content of R.
6) Add coefficients (add X to A).
7) Round off portion of coefficient in Q, if $A_{47} \neq Q_{47}$.
8) Normalize A so that $A_{35} \neq A_{36}$. If normalizing requires left shifting reduce R by one for each shift. If $A_{36} \neq A_{37}$ right shift one place and increase R by one.
9) Test for exponent fault (exponent greater than $2^{10}-1$; set FF in this case).
10) Assemble floating point sum in X: exponent comes from U^2 and coefficient from A.
11) Transmit X to A.

Steps 3, 4 and 5 make the two exponents equal by right shifting the coefficient with the smaller exponent. Step 5 shifts both A and Q to the right in aligning the exponents. Bits of the augend coefficient are moved into Q. As shown in figure 3-8a, the addition of X to A in step 7 may produce a sum of more than 36 bits. Step 8, therefore, rounds off the excess portion in Q by adding 1 to A if $Q_{47} = 1$.

Normalizing (step 8) provides for expressing the sum in proper floating point format. If the sum of the coefficients has more than 36 bits, it must be shifted right and the exponent increased. (At most the sum in A can have 37 bits, and thus only one right shift is ever required.) If the sum in A has less than 36 bits (its most significant bit is not in A_{35}) it must be shifted left and the exponent increased by one for each place shifted.

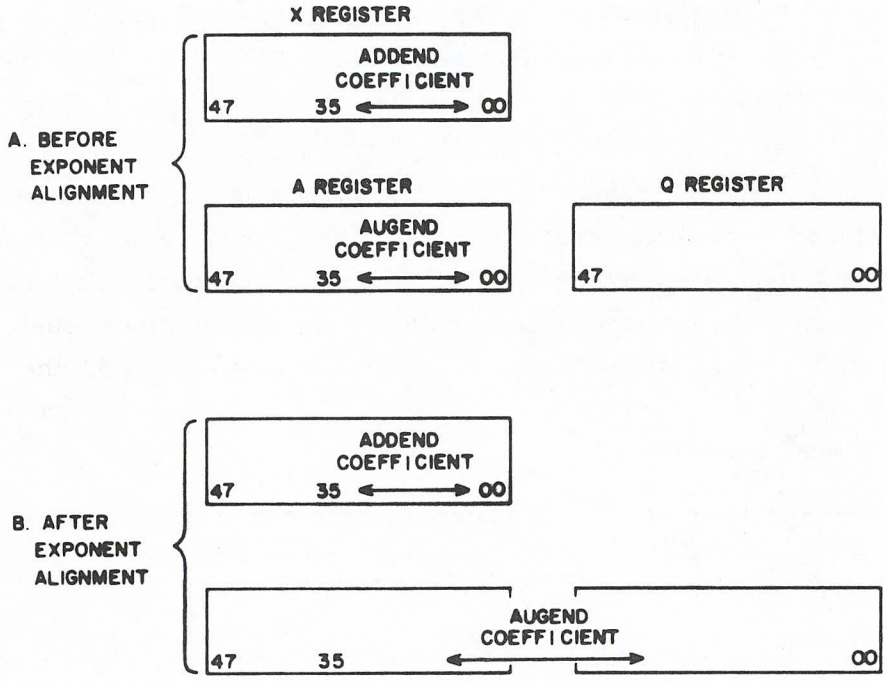


Figure 3-8. Relation of Coefficients in Addition

Subtraction

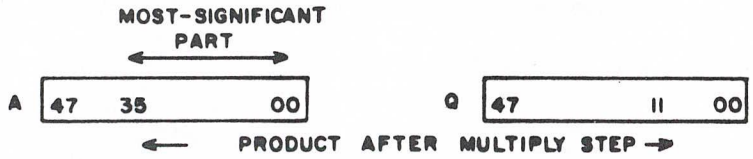
The Floating Subtract instruction, 31, produces the difference of two floating point operands in A (minuend) and address M (subtrahend). As in addition, the exponents and coefficients are separated and the operations performed separately. Basic steps in Floating Subtract are:

1) Enter minuend in A by previous instruction; complement if negative.
2) Acquire subtrahend from address M and place in X; complement if negative.
3) Separate and compare exponents; save larger in U^2; difference in R.
4) Position coefficients so that one with smaller exponent is in A.

5) Shift AQ right by number of places indicated by content of R.
6) Subtract coefficients (subtract X from A).
7) Round off portion of coefficient in Q, if $A_{47} \neq Q_{47}$.
8) Normalize A so that $A_{35} = A_{36}$. If normalizing requires left shifting reduce R by one for each shift. If $A_{37} \neq A_{36}$ right shift one place and increase by one.
9) Test for exponent fault (exponent greater than 2^{10} -1; set FF in this case).
10) Assemble floating point difference in X; exponent from U^2 and coefficient from A.
11) Transmit X to A.

Multiplication

The Floating Multiply instruction, 32, forms the floating point product of two operands in A (multiplier) and address M (multiplicand). Exponents and coefficients are separated and operations performed separately. Multiplication of two coefficients is identical to that for fixed point. However, the multiply step is repeated 36 times instead of 48. The location of the double length product in AQ is bits 11-47 in Q, bits 00-35 in A.



To express the product in floating point format the least-significant 36 bits of the product (located in Q) are rounded off by adding one to A if $Q_{47} \neq A_{47}$. Basic steps are:

1) Enter multiplier in A by previous instruction; complement if negative.
2) Acquire multiplicand from M and enter in X; complement if negative.
3) Set Sign Record FF if product is to be negative.
4) Extract exponents from A and X; send to U^2 and R.
5) Add R to U^2; leave sum in U^2.
6) Initiate multiply step to form product of coefficients; repeat step 36 times; final product in AQ.
7) Complement if Sign Record = 1.
8) Round product to 36 bits by adding "1" to A if $Q_{47} \neq A_{47}$.
9) Transmit sum of exponents from U^2 to R.
10) Normalize product so that $A_{35} \neq A_{36}$; shift AQ left and increase R by "1" per shift, or right shift AQ once and reduce R by "1".
11) Test for exponent fault (exponent greater than 2^{10} -1) and set Exponent Fault FF.

12) Assemble floating point product in X: exponent from R and coefficient from A.

13) Transmit X to A.

Division

The Floating Divide instruction, 33, produces the quotient of the two operands in A (dividend) and address M (divisor). If the dividend is "0", the instruction terminates (with a dividend equal to "0" the quotient would be "0").

The quotient of the two coefficients is formed in a manner identical to that used for fixed point division. However, the divide step for partial division is repeated only 36 times. Basic steps in Floating Divide are:

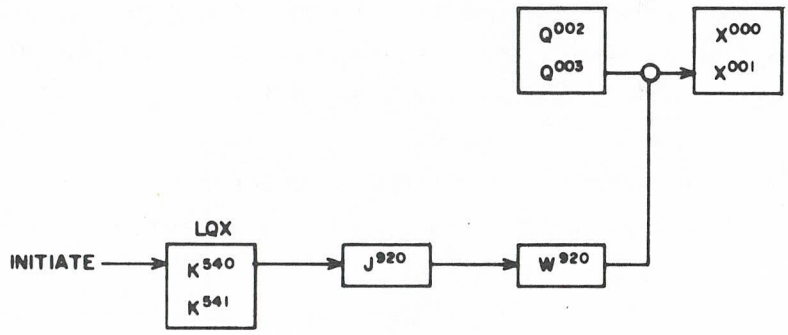
1) Enter dividend in A by previous instruction; complement if negative.
2) Acquire divisor from storage address M and enter in X; complement if negative.
3) Set Sign Record FF if quotient is to be negative.
4) Extract exponents from A and X and sent to U^2 and R.
5) Subtract R from U^2; leave difference in U^2.
6) Initiate divide step to form quotient of coefficients (repeat step 36 times); quotient in Q at conclusion.
7) Complement if Sign Record FF = 1.
8) Transmit quotient from Q to A.
9) Transmit exponent from U^2 to R.
10) Normalize quotient if $A_{35} = A_{36}$ (shift A left and reduce R by one per shift until $A_{35} \neq A_{36}$ or shift right one place and increase R).
11) Test for exponent fault (exponent greater than $2^{10} - 1$; set FF in this case).
12) Assemble floating point quotient in X: exponent from R and coefficient from A.
13) Transmit X to A.

LOGICAL PRODUCT

A logical product is the result of the bit-by-bit multiplication of two numbers. A bit in the logical product is "1" only when both bits are "1"; if either or both bits are "0" the bit of the logical product is "0".

1	0	1	1	multiplicand
1	1	0	0	multiplier
1	0	0	0	logical product

The logical product of two quantities is formed by the computer in X as the result of interaction between X and Q as shown below for one stage of Q and X.



Setting the LQX FF clears every stage of X for which the associated stage of Q holds "0". But for stages of Q holding a "1", the associated stages of X remain undisturbed. Thus the interaction between Q and X as governed by K540/541 obeys the laws of binary multiplication:

Q_i		X_i		X_f
0	x	0	=	0
0	x	1	=	0
1	x	0	=	0
1	x	1	=	1

Forming the logical product of Q and X is a discrete logical function of the arithmetic section of the computer. However, this function is used to accomplish more complex operations such as selecting specific portions of an operand for entry into another operation. As it passes through X the operand is subjected to a mask which has been loaded in Q. The mask, a pattern of "0's" and "1's", causes X to retain its original content only in those stages which have corresponding "1's" in Q. When the selected bits are all that remain in X, the instruction concludes. In some instances, the mask is in storage and the operand to be masked in Q. The result is the same and in either case appears in X. The computer instructions involving a logical product are: (43) Selective Substitute, (44) Load Logical, (45) Add Logical, (46) Subtract Logical, (47) Store Logical, (66) Masked Equality and (67) Masked Threshold.

REGISTER SENSING CIRCUITS

The performance of several computer operations or commands is conditioned by the state of Arithmetic registers.

1) In the divide step, A $\geq$ X conditions the subtraction of X from A and the entering of a "1" as a partial quotient digit.
2) Equality Search instruction (64) searches a list of operands for one equal to A; exit from the search is conditioned by X = A.
3) In the A Jump instruction with j = 0, the choice of jumping is conditioned by A = 0.

Figure 3-9 shows the pyramids which indicate Register condition. The upper pyramid tests for equality between A and X. The first level inverters of the accumulator borrow pyramid determine this condition. Since the inputs of this level require similarity between A and X to produce a "1" output from A994, "1" outputs indicate dissimilarity and produce a "0" output from A994.

The middle pyramid illustrates the test for A = 0. The "0" sides of the A register FFs are brought together in whiffletree fashion. Any "1's" suppress the A = 0 signal. The test for A = 0 is complicated when A = -0 (all ones). A similar whiffletree samples the "1" side of each stage in A, A986 receives OR inputs from A = +0 and A = -0.

A similar pyramid tests for Q = 0. The A = 0 and Q = 0 signals combine to form an AND input to J126 which indicates that the double length register AQ = 0.

Slave inverters connected to the outputs of the sign bit stage of the A, Q and X registers indicate the sign of the content of these registers.

In some operations (test for divide fault, scale, and floating point) the results of the comparison of two bits determines the course of action the computer is to take. (1) A "0" output from J110 indicates inequality between A_{47} and Q_{47} causing overflow during the formation of the quotient. (2) In the Scale instruction J124 indicates whether scaling has been achieved ("1" has been shifted into A46). (3) J131 and J123 position the co-efficient during the normalizing process of the Floating Point instruction. The logic of these comparisons is shown by the inverters in the bottom row in figure 3-9.

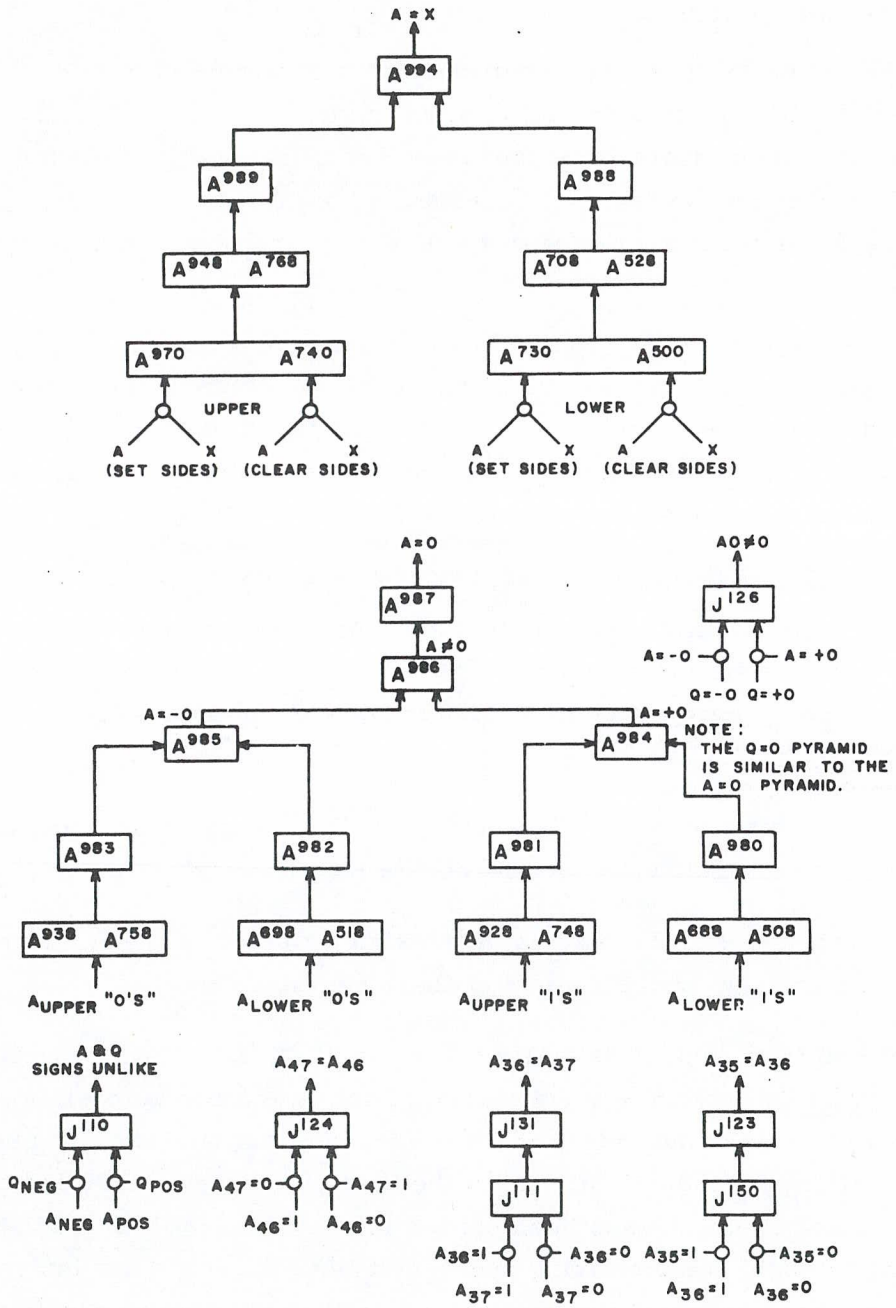


Figure 3-9. Register Sensing Networks

ARITHMETIC FAULTS

The registers used in the arithmetic processes are monitored for Fault conditions by four FFs, Divide, Shift, Overflow, and Exponent Faults. The error-checking circuits are shown in figure 3-10. The FF recognizes the Error condition immediately and lights an indicator on the console. If the program has selected Interrupt On Arithmetic Fault by means of the 74.0 instruction, an interrupt is produced by the fault and the Interrupt sequence is initiated.

Faults may also be detected by following an instruction which might produce one, such as a divide, with a 74.7 instruction having the code for sensing the Divide Fault FF.

DIVIDE FAULTS

The divide fault occurs in the Fixed Point Divide instructions (25 and 27) if the quotient exceeds the capacity of the Quotient register.

In division the operands are first set as positive quantities. (The sign of the quotient is established by the initial sign condition of the operands and stored by the Sign Record FF). Thus the quotient is initially positive and is corrected later if it is to be expressed negatively. In testing for a divide fault the sign of the quotient is inspected before the final correction. At this time the sign bit is "1" only if overflow has occurred. Overflow occurs if the divisor is "0", or if a significant bit falls in the sign bit position.

The Divide Fault FF K232/233 is set by the combination of the following signals: W773 which indicates that Q is negative; F325 which indicates that a 25 or 27 instruction is being executed; and V650 which indicates the quotient has been determined and the sign corrections have not been made.

SHIFT FAULTS

The shift instructions 01, 02, 03, 05, 06 and 07, provide for shifting the contents of the A, Q, or AQ registers to the right or left up to 127 (decimal) places. Any attempt to shift a register more than 127 places results in a shift fault.

The shift count is specified by the modified base execution address. Since 127 is the greatest number that can be coded in the last 7 binary positions, "1's" in positions greater than the 2^6 position in the modified base execution address indicate a fault.

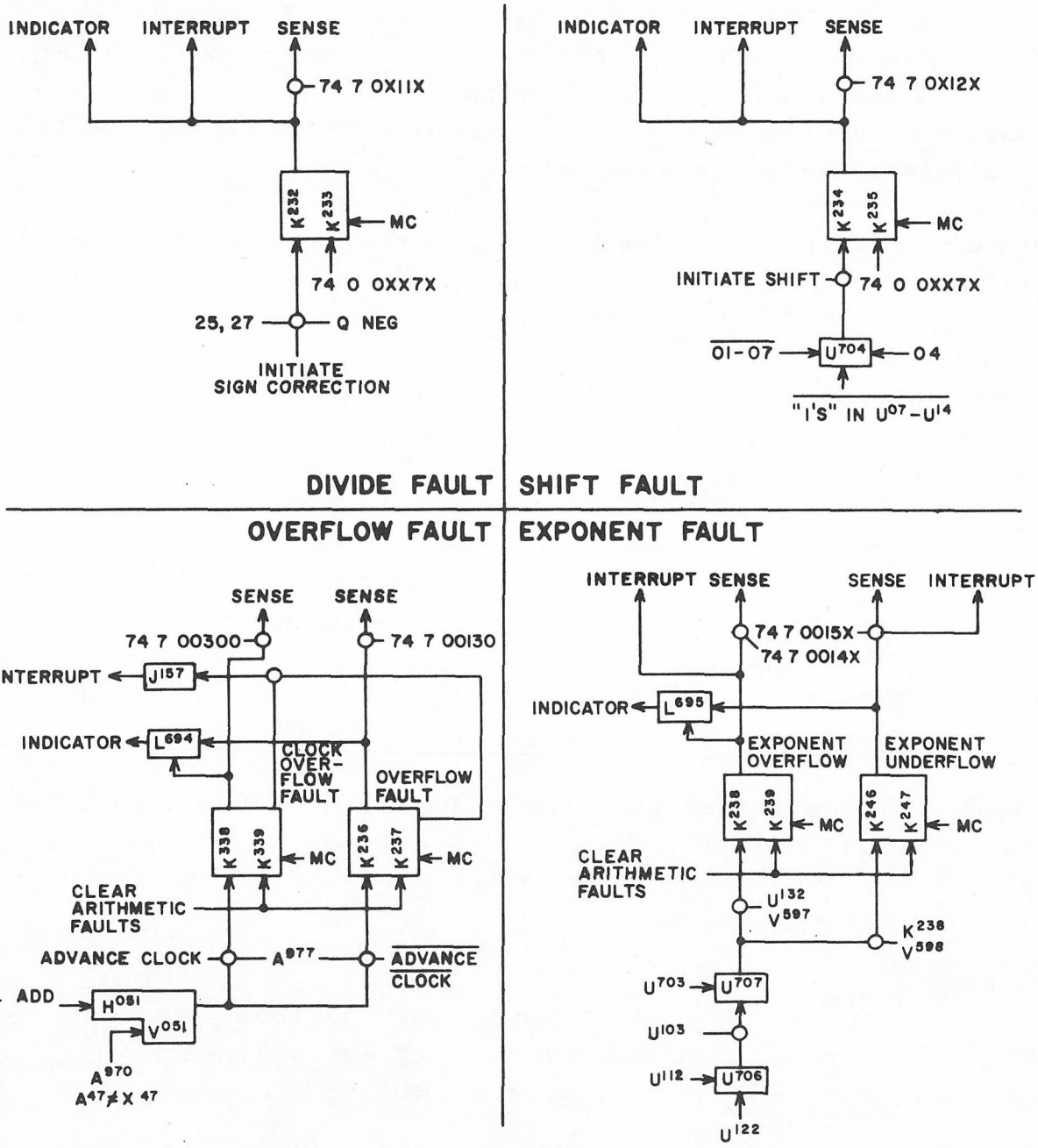


Figure 3-10. Arithmetic Faults

OVERFLOW FAULTS

Overflow faults occur when the sum or difference of two quantities exceeds the capacity of the A register. The possibility of overflow exists only after the addition or subtraction of two quantities of like sign. Since the sum or difference of two like-signed quantities retains the sign of its operands, overflow is indicated by a change in the sign of A^1 (the sum or difference is formed in A^1).

In figure 3-10, A977 indicates the sign of A^1 is changing and V051 issues a pulse one phase time after the command Add $X^2 \rightarrow A^1$ only if A^2 and X^2 have like signs.

EXPONENT FAULTS

An exponent fault occurs during Floating Point instructions when the exponent of the result, after rounding and normalizing, is 2^{+10} (exponent overflow) or 2^{-10} (exponent underflow).

FAULT CONTROL

The Arithmetic Fault FFs are cleared by a Master Clear signal from the console or by the Clear Arithmetic Faults instruction 74 0 00070. The cleared state represents the "no fault" condition. The presence of a Fault condition can be sensed by the program, using the 74.7 instruction with the codes indicated in figure 3-10. Furthermore, the program can select to be notified by an interrupt each time a fault occurs. A 74.0 instruction with the code 00100 produces an interrupt when any of the four faults occur.

In general, an internal interrupt as a result of an arithmetic fault is followed by an interrupt subroutine which checks for all possible interrupt conditions by means of Sense instructions. After determining which selected condition causes the interrupt, a jump is made to that portion of the routine which processes the interrupt. After being interrupted, the computer cannot again be interrupted without returning to the interrupted program (see Interrupts in chapter 6 of this manual). The last instruction is an interrupt subroutine is usually 74 0 00070 (Clear Arithmetic Fault). This instruction should be preceded, however, by a 74 7 00131 (Sense For Overflow Fault) since it is possible that there could be a clock overflow while the interrupt subroutine was being executed. If this Sense instruction is not included as a part of the subroutine and the Clear Arithmetic Faults instruction is given, the Clock Overflow FF is cleared and the programmer has no indication that a clock overflow occurred.

CHAPTER 4
STORAGE SECTION

INTRODUCTION

The storage section of the 1604-A computer provides high-speed, non-volatile, random-access storage for 32,768 48-bit words; it has two independent magnetic core storage units, each with a capacity of 16,384 48-bit words. These units operate together during the execution of a stored program. All odd storage addresses refer to one storage unit; all even addresses, the other.

A word is transferred to or from a storage location by a single instruction. The operation code of the instruction specifies the type of reference (read or write) and the register which serves as the source of destination. The execution address of the instruction identifies the storage location. A read reference is performed by transferring the word at a selected storage address to a specified destination via the X register, and restoring the word at the address. A write reference is performed by clearing the selected storage address, then transferring the word at a specified source to the address via the X register.

The cycle time, or time for a complete storage reference, is 6.4 microseconds. The access time, or time from request to delivery of data from storage, is 2.2 microseconds. The storage cycles of the two sections overlap resulting in an effective cycle time of less than 6.4 microseconds.

The basic logical divisions of the storage section are shown in figure 4-1. The odd and even storage units which are identical, consist of four principal parts:

1) Memory plane assembly contains the magnetic core storage elements of the system.
2) Address selection interprets the address from the control section and selects the specified storage location.
3) Bit plane circuits handle the transfer of information between the memory plane assembly and the X register.
4) Storage sequence control generates the signals that control the storage references.

The magnetic core assemblies occupying approximately one-half of each chassis are evenly distributed among the eight chassis of the computer cabinet. Data transmissions

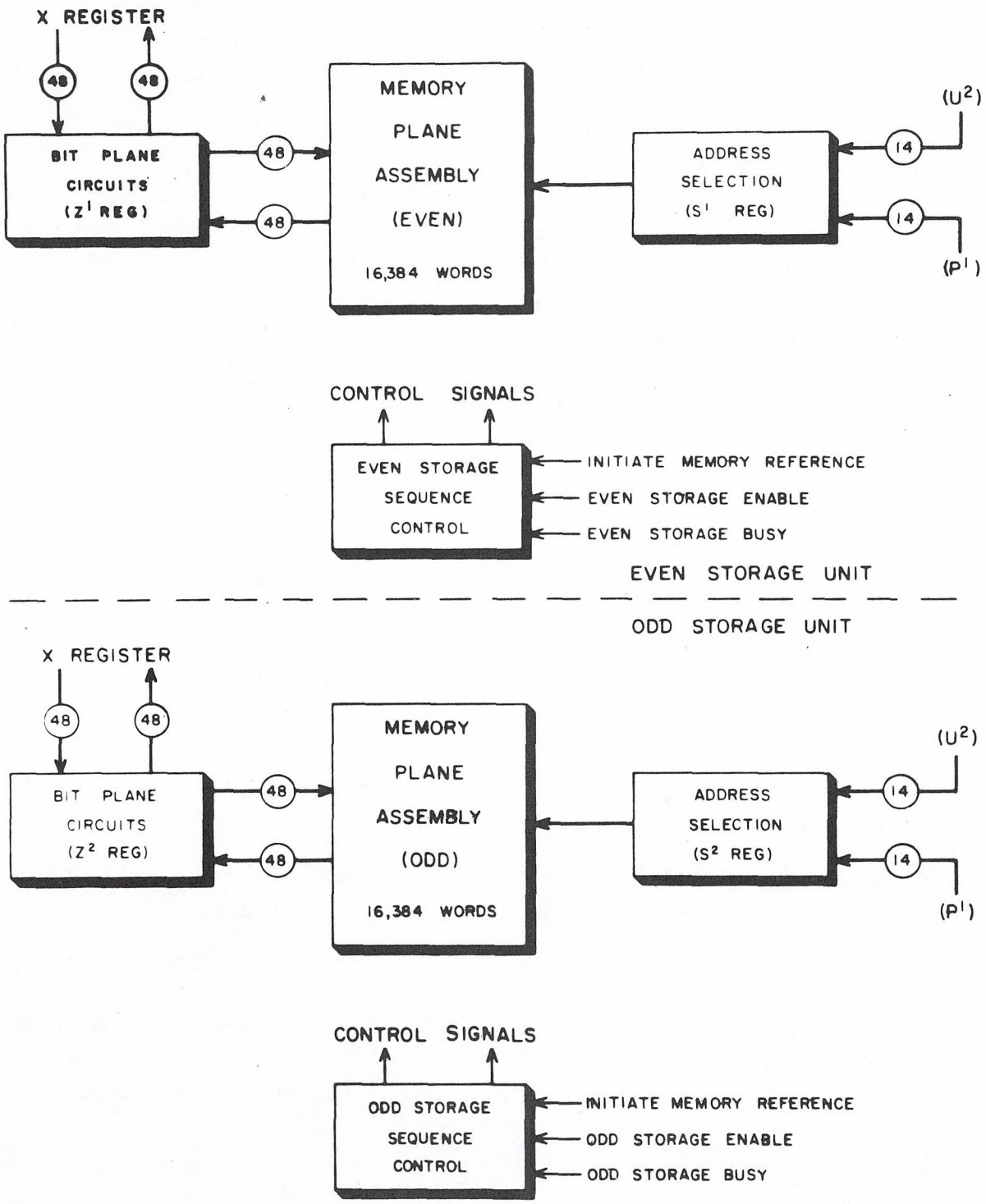


Figure 4-1. Logical Divisions of the Storage Section

into storage are channeled through Z^1 and Z^2, the Storage Restoration registers. The Storage Address registers, S^1 and S^2, hold the address of the storage location involved in a given cycle of operation. The Input/Output, arithmetic and control sections of the computer have independent access to the Storage registers through the Z and S registers.

PRINCIPLES OF MAGNETIC CORE STORAGE

The storage section uses the permanent magnetic properties of ferrite cores to store the bits of computer words. A magnetic core is a bistable device capable of storing a "1" or a "0", depending upon its state of remanent magnetization.

These cores are magnetized in one direction or the other by current-carrying wires which pass through them. The direction of magnetication is determined by the direction of the current flow. The characteristics of the cores are such that approximately 800 ma of current in one turn for a period of one microsecond is required to switch them.

The cores are assembled in square matrices (figure 4-2). Five wires pass through each core: a horizontal H wire, a vertical V wire, a horizontal I wire, a vertical I wire and a diagonal S wire. The coincident-current switching technique is employed. A core is addressed by simultaneously passing half-amplitude current pulses through a selected V wire and a selected H wire.

Only the core at the intersection of the selected H and V wires will be subjected to a magnetizing force sufficiently large to switch its magnetic state. All other cores in the same row or column as the selected core receive half-amplitude current pulses and are said to be half-selected. In figure 4-2, where the left-most V wire and the upper H wire carry current pulses, core A is selected, cores B and C are half-selected, and core D is unselected.

Binary information stored in a core is determined by the polarity of its residual magnetization. A "0" is stored by the magnetizing force of read current pulses on the selected H and V wires and remains stored when current in the I wires inhibits (cancels) the effect of write current pulses in the H and V wires. The absence of current in the I wires permits a "1" to be stored by the write pulse.

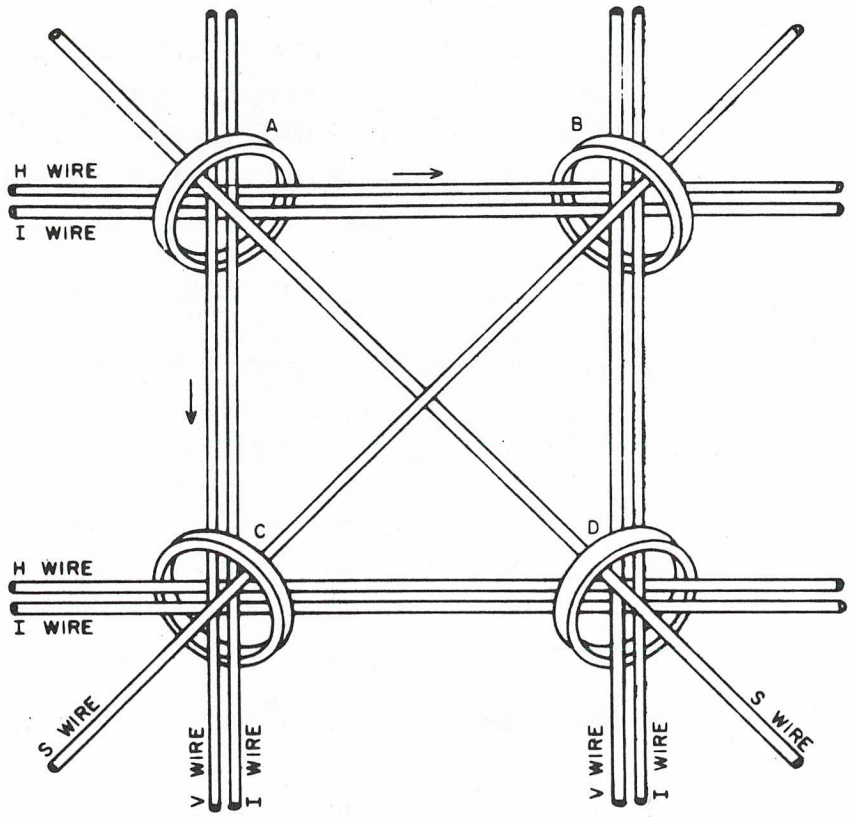


Figure 4-2. Magnetic Core Matrix

The information is extracted (read) from the core by applying read current pulses to the selected H and V wires. If the core stored a "1", the pulses drive it to "0" and a pulse is induced in the S wire. This voltage is interpreted as a "1" bit from the core. If the core stored a "0", it is unaffected by the pulses and no pulse is induced in the S wire. The absence of a voltage is interpreted as a "0" bit from the core.

The matrix arrangement of the magnetic core storage system is called a memory plane. This consists of 16,384 cores in a 128 x 128 array. In order that the coincident-current storage technique may be employed, each bit of a word must be stored on a separate memory plane. Thus, the 48-bit words used in the computer are stored by 48 memory planes. This array is called a memory plane assembly.

There are two memory plane assemblies in the storage section, one associated with the odd storage unit and the other with the even storage unit. Both assemblies are evenly divided among the chassis of the computer. A stack of six "even" and six "odd" memory planes is mounted on each chassis (figure 4-3).

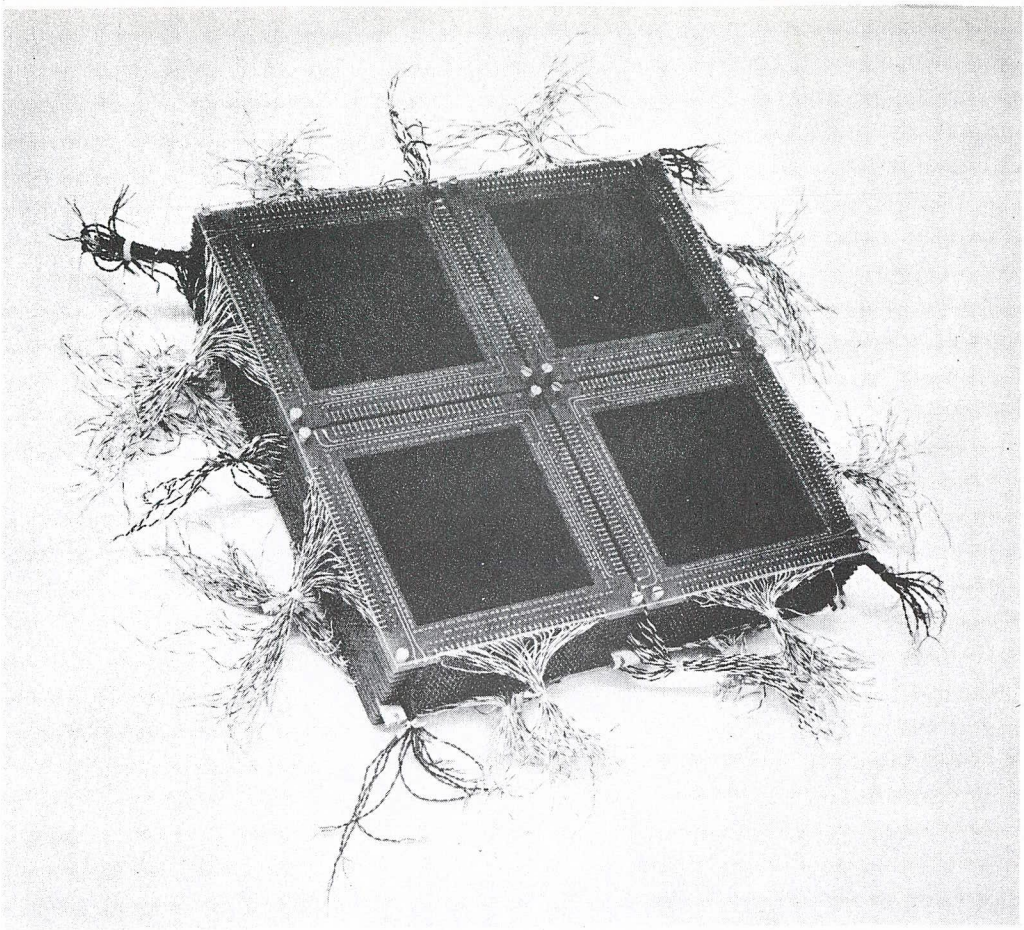


Figure 4-3. Memory Plane Stack

MAGNETIC CORES

The magnetic properties of a core are represented by its hysteresis diagram, which plots magnetic flux density (B) as a function of the field intensity (H) (figure 4-4). If current flow sufficient to cause a field intensity of $+H_m$ is applied to the drive lines, the flux density increases to saturation ($+B_s$). When the current is removed, the flux density drops to the residual positive value ($+B_r$), which has been designated the "0" state, and remains there. Another pulse of $+H_m$ would merely shift the core to $+B_s$ again and after the pulse is removed, it would drop back to $+B_r$. Application of current flow sufficient to cause a field intensity of $-H_m$ reverses the flux density to $-B_s$ and, when the current is removed, the flux density drops to the residual negative value ($-B_r$), the "1" state.

The basic memory cycle is composed of half-amplitude pulses, each capable of producing a field intensity of $H_m/2$. A half-amplitude pulse is insufficient to switch the core; instead the flux density returns to the residual value, or a slightly lower value, after the pulse is removed. The coincidence of two half-amplitude pulses, one on the H

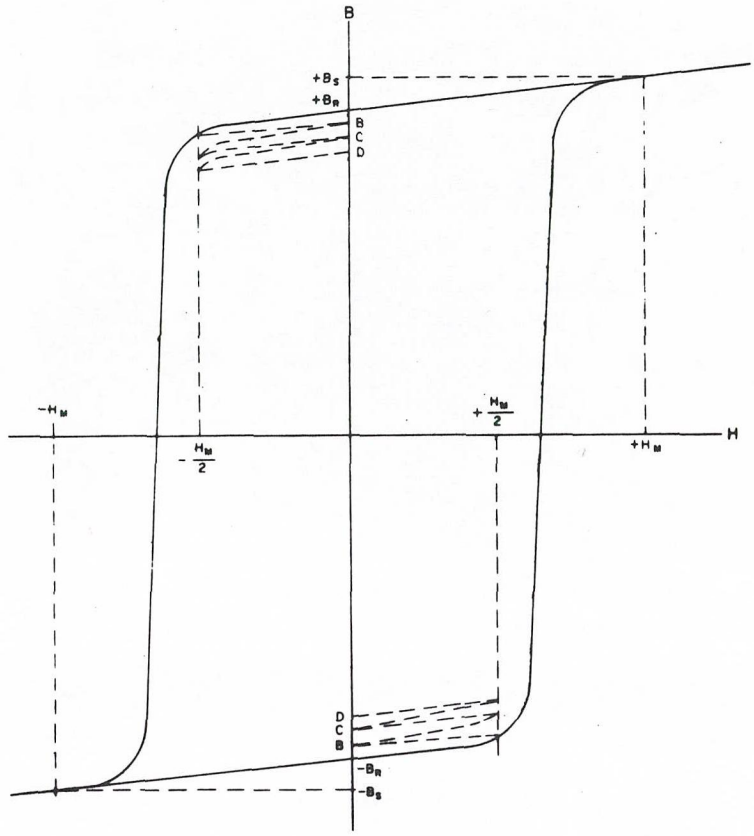


Figure 4-4. Hysteresis Diagram

drive line and the other on the V drive line of a core, produces a net field of H_m which is sufficient to switch the core. When a half-amplitude pulse drives the flux density toward the knee of the hysteresis loop, the flux travels up (or down) the knee somewhat and then returns to a slightly lower residual value, such as B. Since the core is now operating on a smaller loop, further half-pulses reduce this remanent flux again, but this effect soon reaches a limit, as at point D.

Any change in the magnetic state of a core causes a change in the total flux linking the core and any winding passing through it. Such a change produces a voltage output on the sense winding (figure 4-5). During the period that H is applied, the voltage is sampled to see if the core switches. If a large voltage is sensed, the core was in the "1" state and has switched. If only a small voltage is sensed, the core was in the "0" state and has merely shifted from $+B_r$ to $+B_s$ and back again.

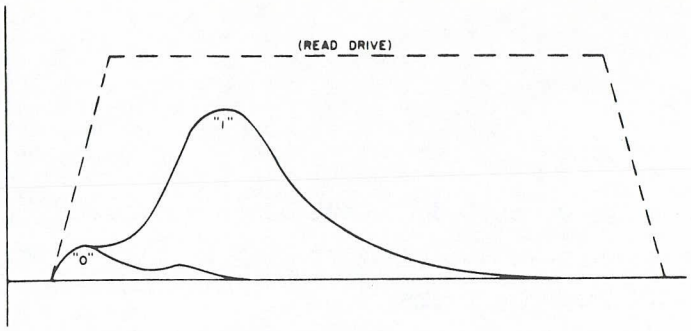


Figure 4-5. Voltage on Sense Winding as a Result of Read Drive

MEMORY PLANES

A memory plane consists of four quadrants or boards, numbered 0, 1, 2 and 3. Each board consists of a phenolic frame with printed circuit wiring on both sides. The board holds 4096 cores, held in a 64 x 64 array by 64 horizontal H wires and 64 vertical V wires. One I wire threads all the cores both horizontally and vertically and one S wire threads all the cores diagonally. The H and V wires thread across the board vertically and horizontally and terminate at tabs on either side of the edge of the board (figure 4-6).

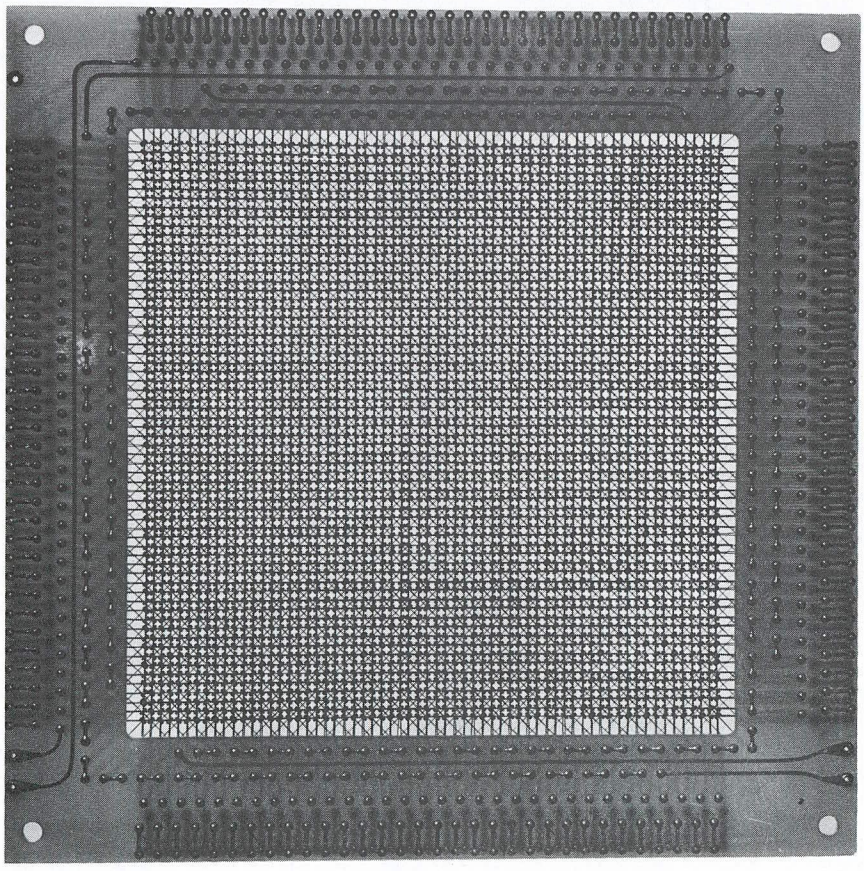


Figure 4-6. Memory Board

A wire connected to a front tab on one edge of the board terminates on a rear tab on the opposite edge. Each I wire terminates at two tabs in a corner of the board, from which connections are made to the inhibit drivers.

Each S wire terminates at a pair of tabs in a second corner of the board, from which connections are made to the sense amplifiers. The four boards of a memory plane are so oriented that the sense wire tabs are at the corners of the memory plane. Corresponding H wires and corresponding V wires of adjoining boards are connected together by short lengths of wire.

MEMORY PLANE ASSEMBLY

Twelve memory planes, stacked one behind another, make up a stack (figure 4-3). The memory planes are bolted together through the four corners of each and are separated by aluminum spacers. An aluminum plate at the back and a plexiglass plate at the front shield the stack. A memory plane assembly is made up of six memory planes from each of eight such stacks, or a total of 48 planes. Since each plane stores one bit of a word, and there are 16, 384 cores on a plane, the memory plane assembly provides storage for 16, 384 48-bit words.

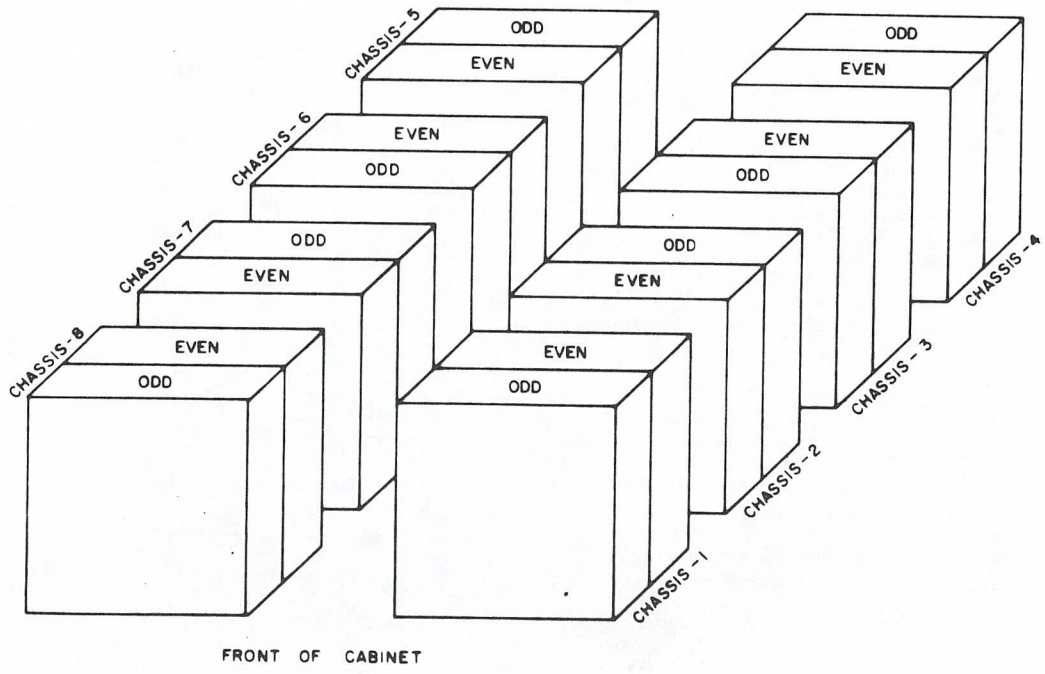


Figure 4-7. Distribution of Memory Plane Assemblies

The memory plane assemblies for odd and even storage are evenly distributed among the eight computer chassis (figure 4-7). The stack of chassis 1 stores bits 0 through 5 of all the words; the stack of chassis 2 stores bits 6 through 11, etc. The even memory planes are on the card side of the chassis; odd memory planes are on the wiring side.

Corresponding H wires and corresponding V wires of each group of six memory planes of a stack are connected in series by short lengths of wires soldered to the tabs at the edges of the boards. The H and V wires of each group of memory planes are connected to separate driver and diversion circuits. The corresponding drivers and diverters of the even memory are connected in parallel to the S^1 register, located on chassis 3; those of the odd memory unit are connected in parallel to the S^2 register, located on chassis 4. Thus, a complete memory plane assembly is the logical equivalent of 48 memory planes, stacked one behind another.

The cores in each horizontal plane are connected in series by an H wire (figure 4-8). Similarly, the cores in each vertical plane are connected in series by a V wire. Coincident currents on a selected pair of H and V wires affect only those cores at the intersection of the horizontal and vertical planes formed by those wires.

Separate connections are made to the sense and inhibit lines of each quadrant of a memory plane. Each sense line is brought to a pair of tabs at a corner of the memory

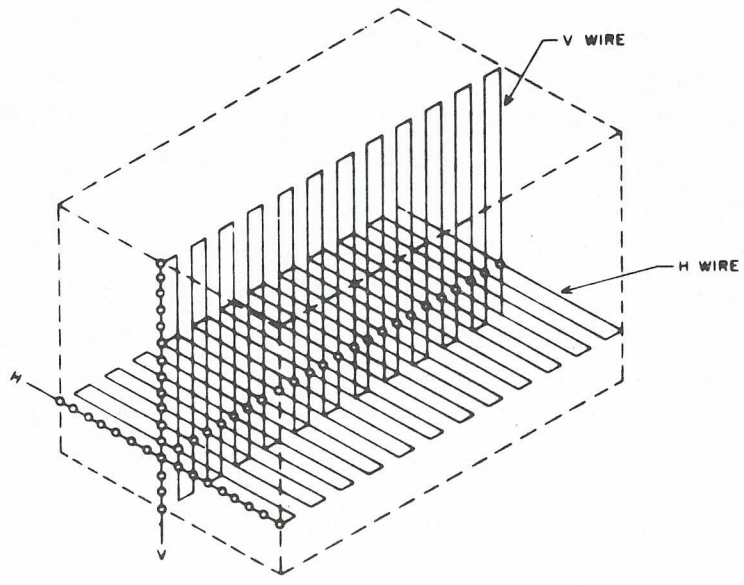


Figure 4-8. Intersection of H and V Wires

board. These tabs are connected to the sense amplifier circuits by twisted pairs. Each inhibit line is brought to a pair of tabs half way between two of the corners of the memory board. The tabs are connected to the inhibit circuits by twisted pairs.

ADDRESS SELECTION

A storage location is the row of cores at the intersection of a selected pair of H and V wires of a memory plane assembly. Each memory plane assembly has 16, 384 storage locations. The Address Selection circuit selects a discrete storage location for each different address in S^1 or S^2.

Exactly 15 bits are required to separately identify each of the 32, 768 storage locations. The identifications, called addresses, range from 00000 to 77777 inclusive. The lowest-order bit of the 15-bit address selects either the odd or even address selection system. The remaining 14 bits are placed in the S register of the selected system and translated to select a single H wire and a single V wire of each matrix plane of the memory plane assembly.

The Address Selection circuits for the odd and even storage units are identical. Each is composed of four logical systems:

1) The vertical drive system, consisting of a translator and eight drivers for each of the storage units on a chassis, selects a group of 16 V wires, among which is the desired V wire, and supplies the necessary drive current for this wire.
2) The horizontal drive system, consisting of a translator and eight drivers for each of the storage units on a chassis, selects a group of 16 H wires, among which is the desired H wire, and supplies the necessary drive current for this wire.
3) The vertical diversion system, consisting of a translator and 16 diverters for each of the storage units on a chassis, selects the desired V wire from the group of V wires selected by the vertical drive system.
4) The horizontal diversion system, consisting of a translator and 16 diverters for each of the storage units on a chassis, selects the desired H wire from the group of H wires selected by the horizontal drive system.

The address selection system logically divides each stack into eight vertical and eight horizontal sections (figure 4-9). Each horizontal section consists of 16 H wires; each vertical section consists of 16 V wires.

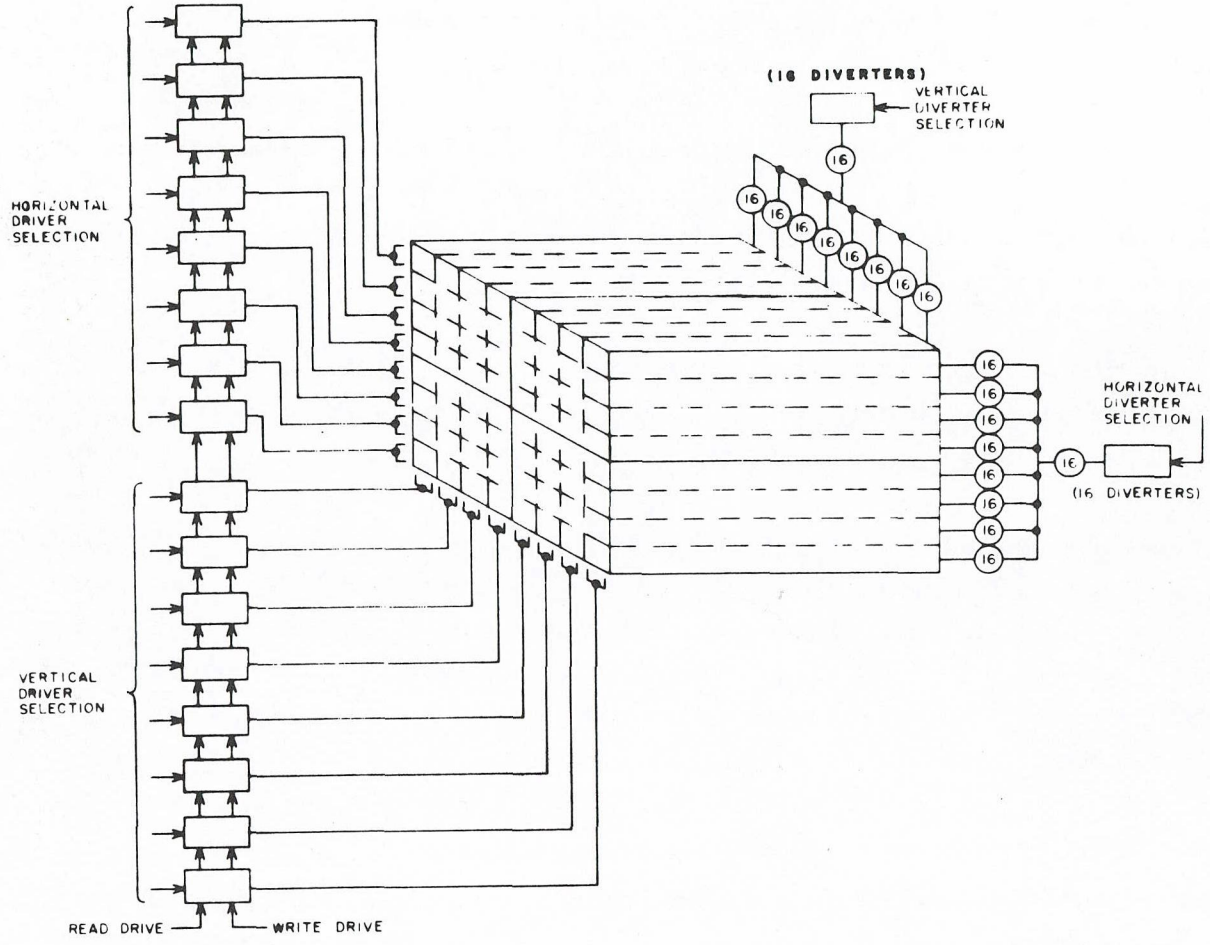


Figure 4-9. Connection of Drive Lines and Diversion Lines

The region generated by the intersection of an H section and a V section and common to both is called a quarter section. One end of each of the wires in a horizontal section is connected to a common horizontal driver; similarly, one end of each of the wires in a vertical section is connected to a common vertical driver. Thus, the selection of a horizontal and vertical driver by the respective translators selects a region of 256 storage locations, one of which is the desired one. The other end of the H wires is connected to the diverters; corresponding H wires of each horizontal section are connected to a single diverter. The selection of a horizontal and vertical diverter completes the electrical circuits to the selected horizontal and vertical drivers, allowing current to flow from the drivers through the respective H and V wires, through the diverters, and back to the current source.

The selection of drivers and diverters by consecutive storage addresses follows an orderly pattern. The bits of a storage address are combined, for selection purposes, in the pattern shown at the top of figure 4-10. The lowest order bit specifies the odd or even memory unit; the next two higher order bits specify the quadrant within that unit. Thus, the three lowest order bits divide each stack into eight portions, and are termed the "octant selection bits". The next six higher order bits of the address select the vertical drivers and diverters in the octant while the six highest order bits select the horizontal drivers and diverters in the octant.

The selection of a vertical driver is given by bits 1, 8, 7 where bit 1 is treated as the highest order one. Similarly the selection of a horizontal driver is given by bits 2, 14, 13 where bit 2 is treated as the highest order one.

The ordered selection of drivers and diverters within an octant is shown in the lower part of figure 4-10. The selection of vertical diverters proceeds from left to right, in repetitive cycles, at 1/8 the frequency of address change. The selection of vertical drivers proceeds from left to right, in repetitive cycles, at 1/128 frequency of address change. Horizontal diverters and drivers are selected in the same manner but at 1/64 the rate of the vertical.

S REGISTER

The S^1 and S^2 registers (figure 4-11) hold the storage address during a storage reference. Each register consists of 14 single-rank FF stages. An address is entered into one of the registers by one of three commands. The command $U^2 \rightarrow S^1$ (or S^2) enters the quantity stored in U^2 into S; the command $P^1 \rightarrow S^1$ (or S^2) enters the quantity in P^1 into S. The command $CCR_U \rightarrow S^1$ (or S^2) enters the quantity in CCR_U (the address to or from which a word is being buffered) into S. The $U^2 \rightarrow S$ and $P^1 \rightarrow S$ transfers are forced transmissions; they transfer both outputs of a stage of U^2 or P^1 to the corresponding stages of S.

HORIZONTAL AND VERTICAL DRIVERS

There are eight vertical and eight horizontal driver circuits for the odd memory plane assembly and the same number for the even on each chassis (figure 4-12). The vertical driver circuits are connected in parallel stages 01, 07 and 08 of the S register; the horizontal driver circuits are connected in parallel stages 02, 13 and 14.

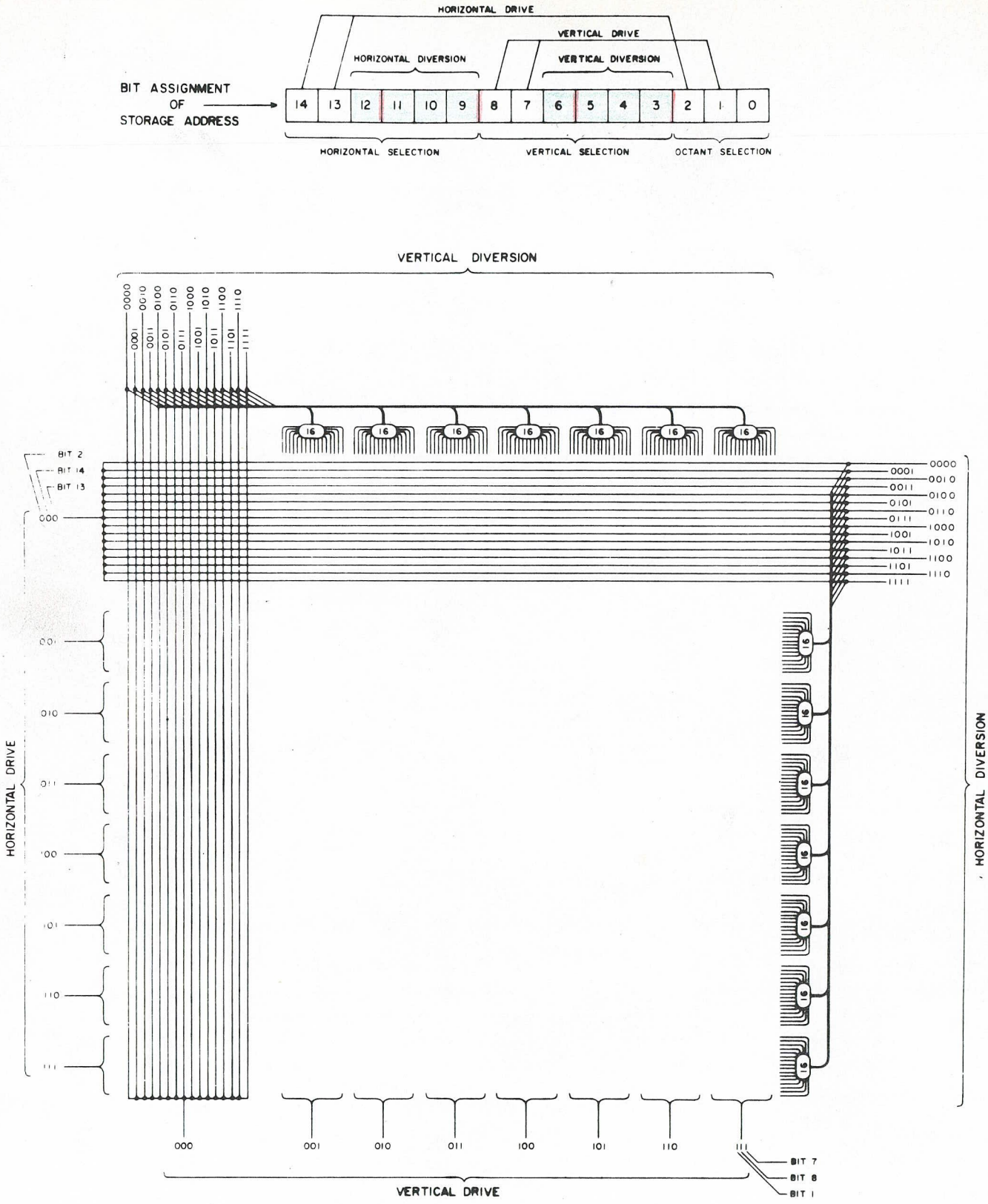
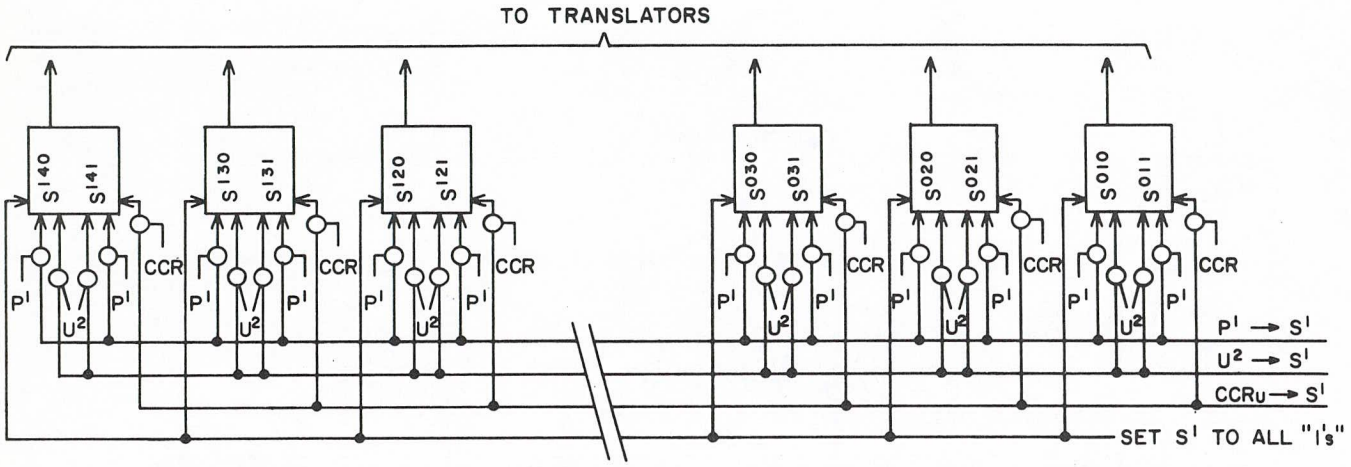
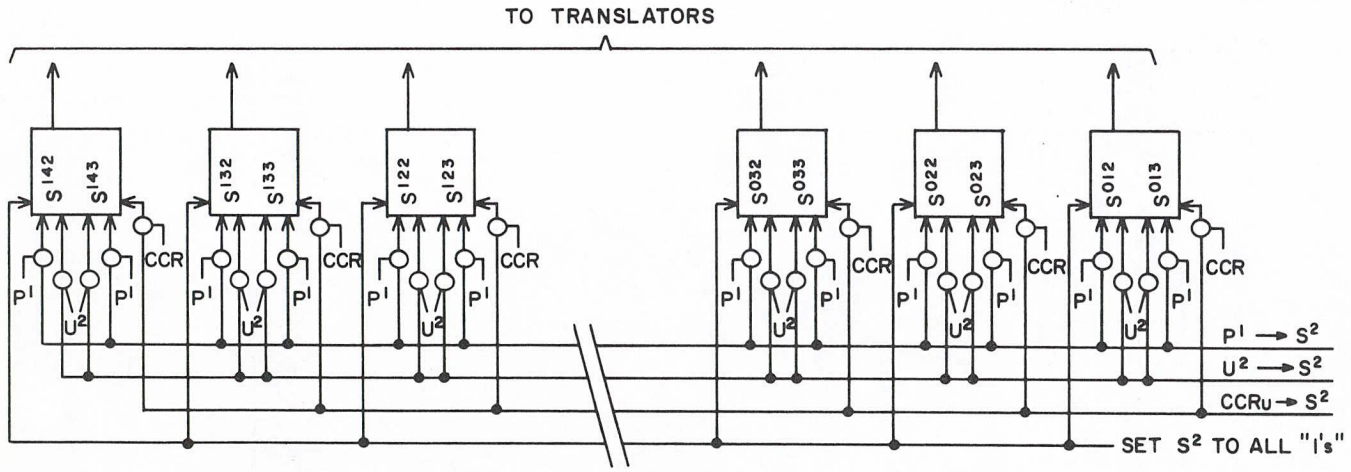


Figure 4-10. Drivers and Diverters Selected by S^1 or S^2 Registers



S^1 Register (Even Memory)



S^2 Register (Odd Memory)

Figure 4-11. S^1 and S^2 Registers

The translator associated with each driver circuit selects one of the eight drivers (G^{---} in figure 4-12) on the basis of the contents of the three stages of S. The translation is performed by single-inverter cards (E^{---}) and selector cards (T^{---}). The E^{---} cards provide positive "1" and "0" outputs on the basis of the information stored in the FFs. The outputs of the E^{---} cards, along with the signals Read Drive and Write Drive from the storage sequence control, are applied to the T^{---} cards to produce the following output signals: Read XX0, Write XX0, Read XX1, Write XX1, 00X, 01X, 10X and 11X. In each case, X represents a bit of S which does not affect the output of the T^{---} cards.

The outputs of the T^{---} cards are combined at AND inputs to the G^{---} cards in such a manner that a read output is applied to one AND of each card and a write output to the other, and each is ANDed with one of the other four outputs. Thus, for any combination of bits in the three stages of S, one G^{---} card is selected which provides both read and write currents to the memory plane stack.

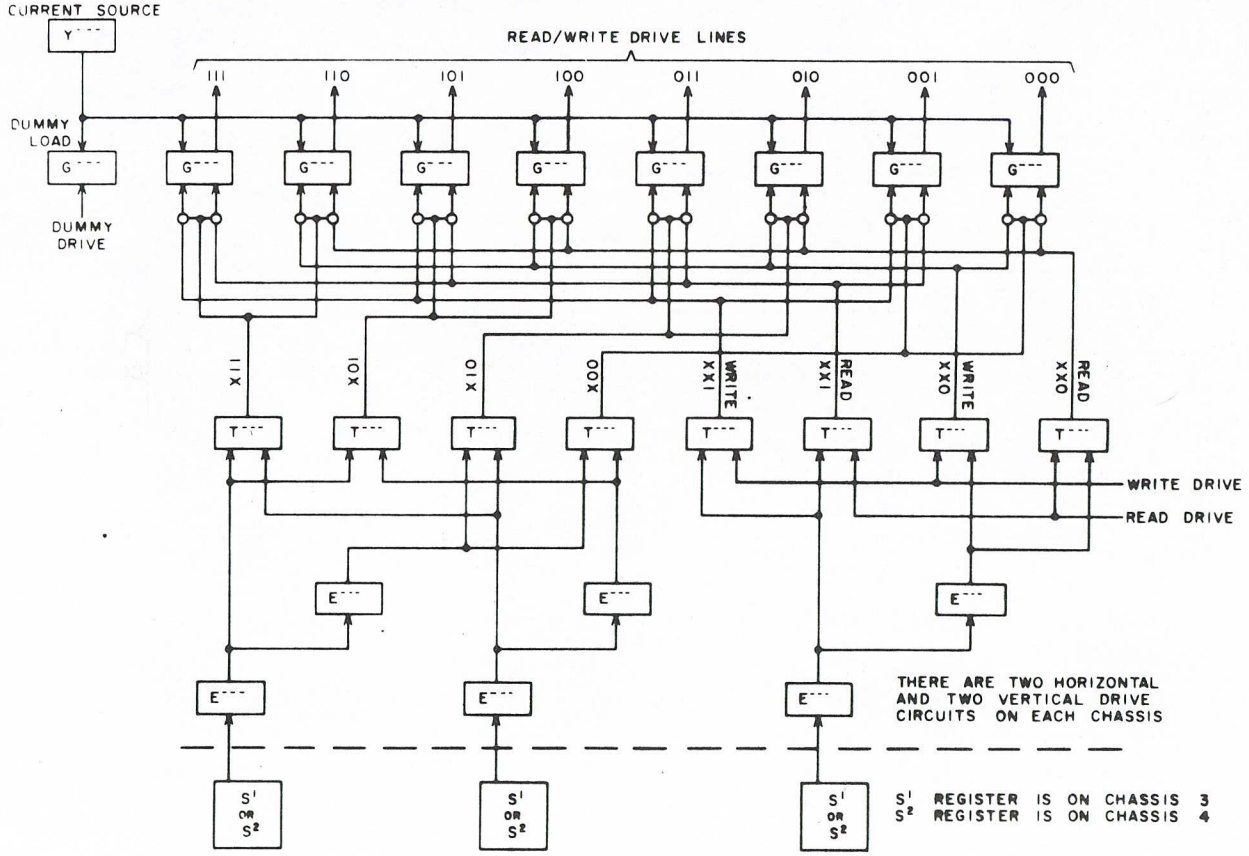


Figure 4-12. Typical Horizontal or Vertical Drive Circuit

The current source card (Y^{---}) provides d-c read-write current to the selected driver card. To maintain a constant load on the power supply, a G^{---} card is connected as a dummy load to the current source card. During periods of no storage references, the dummy load is continuously selected; during a storage reference the dummy load is disabled for a period of 6.2 microseconds by the Dummy Drive signal. (A-C fluctuations in the load do not adversely affect the power supply.) The Dummy Drive signal originates at the storage sequence control.

HORIZONTAL AND VERTICAL DIVERTERS

Each stack of odd and even memory units has 16 horizontal and 16 vertical diverter circuits (figure 4-13). Vertical diverter circuits are connected in parallel to stages 03, 04, 05 and 06 of the S register; horizontal diverter circuits to stages 09, 10, 11 and 12.

The translator selects one diverter card (D^{---}) on the basis of the contents of the four stages of S. The E^{---} cards provide positive "1" and "0" outputs depending on the information in the FFs. The outputs from E^{---} are applied to T^{---} (selectors) to produce negative output signals: XX00, XX01, XX10, XX11, 00XX, 01XX, 10XX and 11XX. The outputs of T^{---} are combined at the inputs of D^{---} so that only one diverter card is selected for any combination of bits in S. The card completes the current path for a single H or V wire within the group of 16 wires fed by the selected driver.

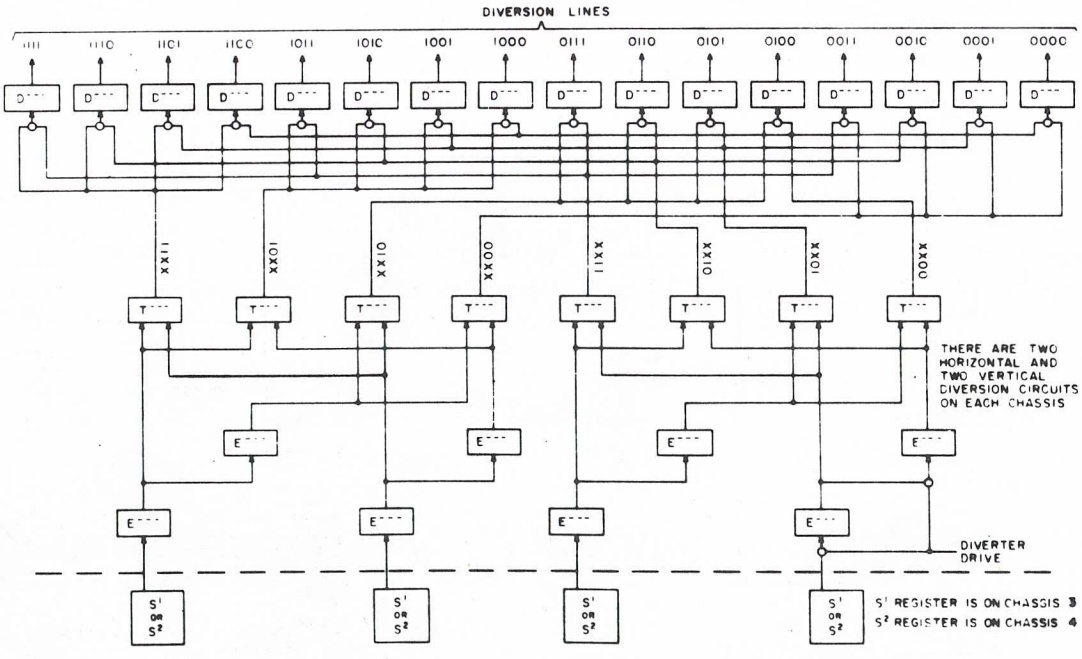


Figure 4-13. Horizontal or Vertical Diversion Circuit

MEMORY PLANE CIRCUITS

The memory plane circuits transfer data between storage and the rest of the computer. Data flows out of storage during the first half of the storage cycle. The selected cores are sampled by detecting the voltage induced in the sense windings when the read drive pulse clears the cores to the "0" state. Those cores which stored "1" bits produce pulses on the amplified sense windings. During read operations, the Read signal and the quadrant selection signal gate the pulses from the sense amplifiers to the I^5 (even storage unit) or I^6 (odd storage unit) rank of inverters. (During write operations, the Write signal gates the word in X^1 into I^5 or I^6). The Set Z pulse gates the pulses from I^5 or I^6 to the Z register during a critical portion of the read drive pulse. During read operations, pulse $I^5I^6 \rightarrow$ X or $I^5I^6 \rightarrow$ U gates the pulses from I^5 or I^6 to the X or U registers.

Data flows into storage during the second half of the storage cycle. The quantity to be stored is transferred from X to Z via I^5 or I^6 or from the sense windings to Z via I^5 or I^6. The inhibit current generators prevent the write drive pulse from setting the cores to "1" by producing an inhibit pulse corresponding with each "0" bit in Z.

SENSING CIRCUIT

For the duration of the read drive pulse, all 48 cores of the selected word receive full-amplitude pulses to switch their flux states to "0". A voltage is induced in the sense line of each core (figure 4-5). Voltages may be of either polarity because of the manner in which the sense line is strung through the cores.

Since the "1" output is the desired signal, the output from "0" is regarded as noise. This signal arises from the shape of the hysteresis loop but is also dependent on the number of half-write pulses the core has received.

The flux density of half-selected cores in the "1" condition is reduced slightly, while that of cores in the "0" condition is increased slightly, by the read drive pulse. In both cases, noise voltages of 1 to 2 millivolts are produced on the sense winding. The noise voltages are reduced: (1) by threading the sense line through the cores of a quadrant so that the noise signals from half-selected cores cancel each other, and (2) by sampling the sense amplifiers at a time when the "1" output voltages are near peak and the noise voltages have decayed.

The path of the sense wire through a four core matrix is shown in figure 4-14. If drivers 1 and 3 generate read drive current pulses, A receives a full field, B and C half fields, and D no field. Core A, which is selected, induces the "1" signal on the sense line while cores B and C induce noise voltages on the sense line. Most of the noise signals in a quadrant cancel each other if they are nearly equal.

In the sensing circuits for each memory plane, signals from the sense lines are applied to the sense amplifiers, (in figure 4-15 Y100, Y101, Y102 or Y103) and the output of the amplifier is gated to I^5 by the Read Quadrant signal. A "1" signal in the sense line results in a "0" signal from the sense amplifier and a "1" signal from I^5.

The Set Z signal from the storage sequence control gates the output of I^5 (or I^6) into the Z register. The register is initially cleared by the Clear Z signal. A "1" output from a stage of I^5 allows the Set Z signal to pass the AND and set the stage of Z to "1". In a read reference, either signal $I^5\ I^6 \rightarrow X$ or $I^5\ I^6 \rightarrow U$ is also generated to set the stages of X and U according to "1's" in I^5 or I^6. During buffer operations inputs come from I^0 to I^5 or I^6 and outputs are gated directly from Z to I^1 to the O register.

The "1" signal from the sense amplifier reaches its peak after the "0" signal has decayed to about 6 millivolts (figure 4-5). By generating the Set Z, $I^5\ I^6 \rightarrow X$ and $I^5\ I^6 \rightarrow X$ and $I^5\ I^6 \rightarrow U$ signals approximately 1.6 microseconds after the start of the read drive pulse, most of the "0" signal is avoided.

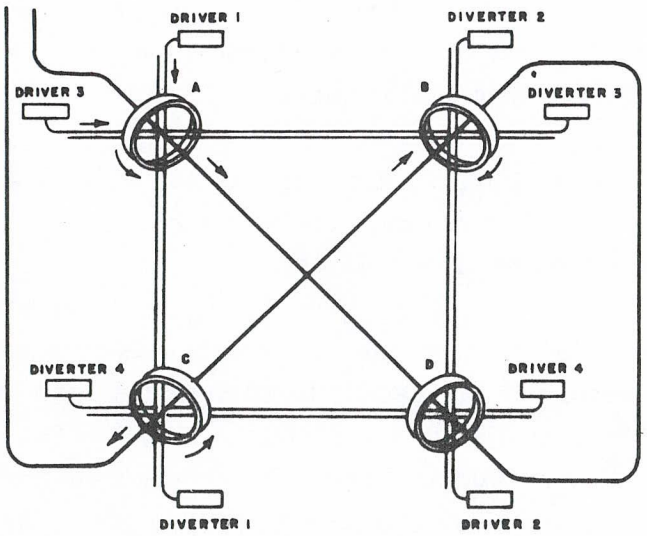


Figure 4-14. Path of Sense Wire Through a Four-Core Matrix

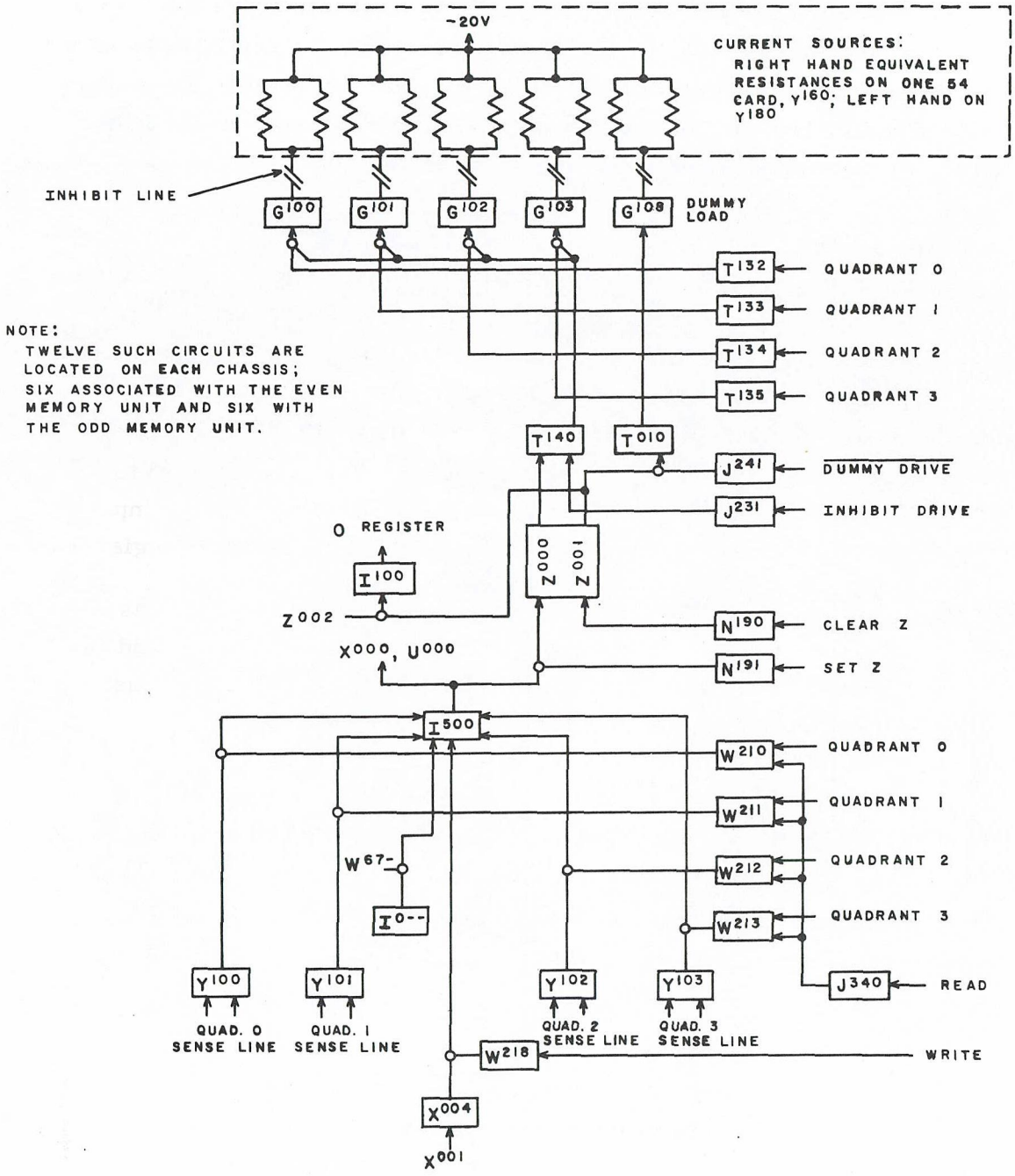


Figure 4-15. Typical Stage of the Memory Plane Control

INHIBIT CIRCUITS

Figure 4-16 shows the path of the inhibit wire through a four-core matrix. The inhibit current flows in opposite directions in adjacent lines. The direction of the write drive current reverses from line to line. For example, in figure 4-16 the write drive current flows down from driver 1, up from driver 2, right from driver 3 and left from driver 4. Following the path of the inhibit line through the matrix, it is apparent that the inhibit current is at all times opposite in direction to the write drive current.

The inhibit circuit for a typical memory plane is shown in figure 4-15. The write drive pulse, following the read drive pulse, attempts to switch all the cores of the selected memory location to the "1" state. The inhibit drive pulse passes inverter T^{140} if the stage of Z stores a "0" bit and probes the AND inputs to the inhibit current generators (G^{100}, G^{101}, G^{102} and G^{103}). One of these AND inputs is enabled by the quadrant selection signal; the output of the AND enables the inhibit current generator, which draws

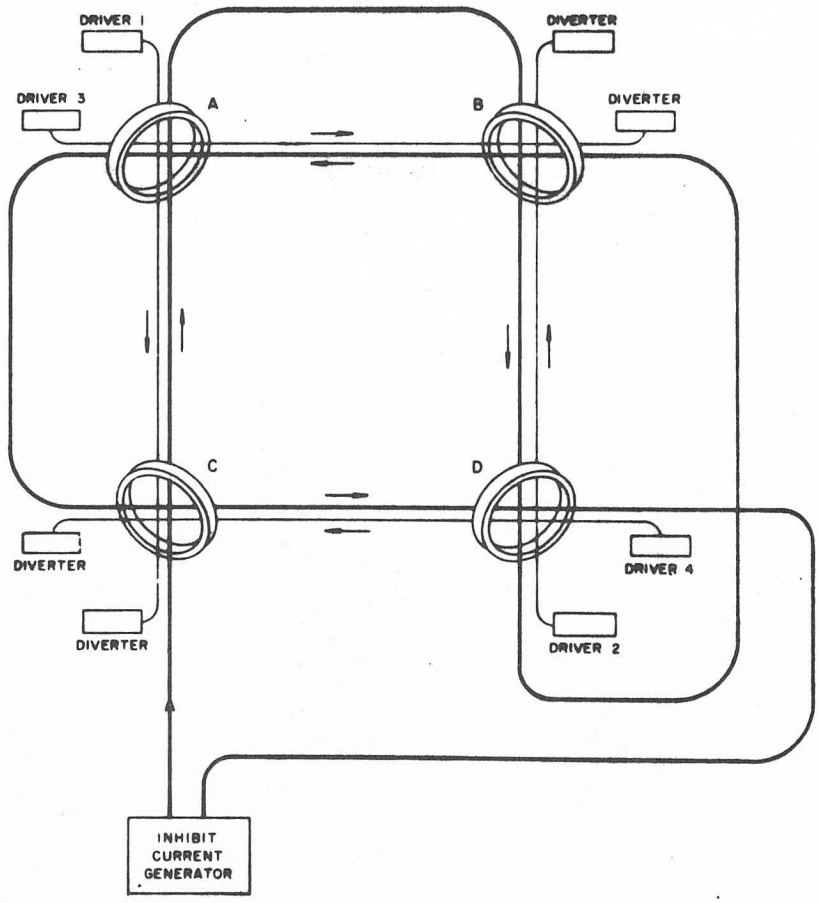


Figure 4-16. Path of Inhibit Wire Through a Four-Core Matrix

current through the inhibit wire from the current source (Y^{160} and Y^{180}). The inhibit current, occurring at the time of the write current but opposite in polarity, cancels the effect of the write current within that quadrant and the core is not switched to "1".

The dummy load associated with an inhibit circuit is continuously enabled during periods of no storage references and also during those references which do not result in inhibit current generation. For references which result in inhibit current generation, the "0" output of Z enables the AND input of T^{010}. As a result, the Dummy-Drive signal passes T^{010} and disables the dummy load for a period of 6.2 usec during the reference.

STORAGE SEQUENCE CONTROL

The storage sequence control, in response to initiating signals from the control section of the computer, executes reading and writing operations by generating the signals which control the address selection and bit plane circuits. The basic pulse sequence for reading and writing is shown in figure 4-17. The read drive (first pulse) drives the selected core to "0". The inhibit drive (second pulse) allows a "0" bit to be retained in the core by inhibiting the effect of the write drive pulse. The write drive (third pulse) drives the core to "1" if the inhibit drive is absent.

The storage sequence control consists of the initiate storage reference circuit and identical sequence controls for the odd and even storage units. The circuit initiates a reference by entering the address in the S register and selecting odd or even sequence control. The sequence control then generates a fixed sequence of control pulses which executes the reference.

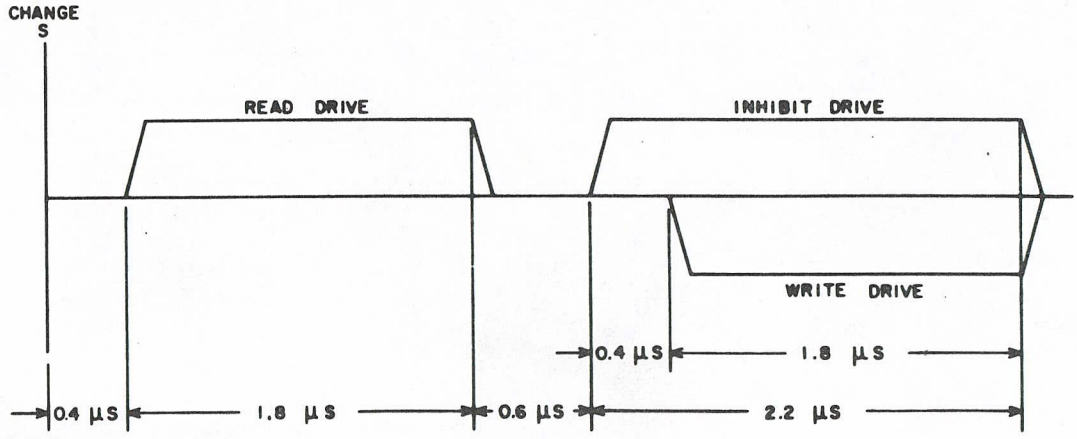


Figure 4-17. Basic Pulse Sequence for Storage Reference

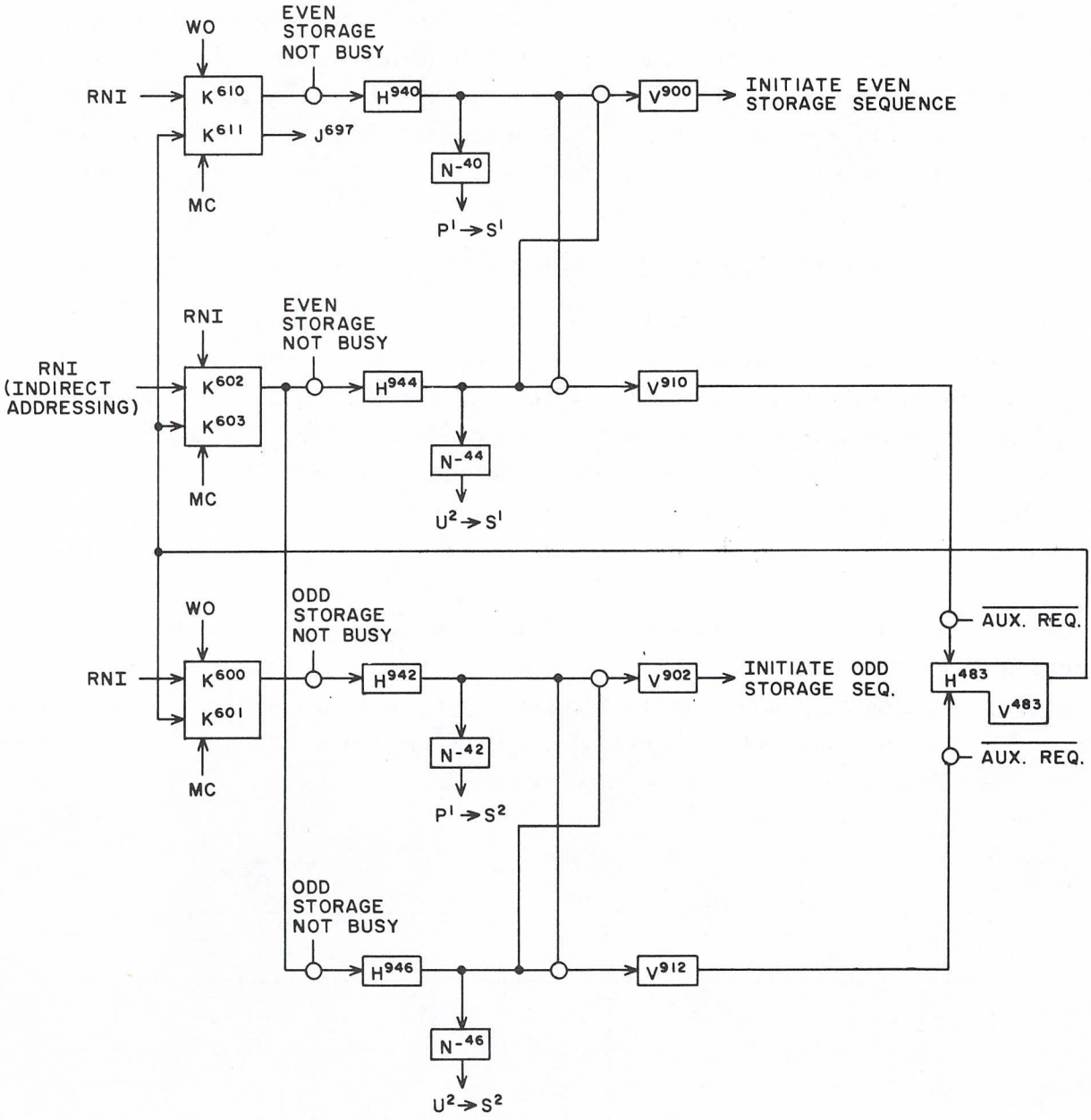


Figure 4-18. Storage Reference Circuit

INITIATE STORAGE REFERENCE CIRCUIT

A sequence chain of the control section generates the Initiate Storage Reference signal. This signal performs two functions: (1) It controls the transmission of the storage address to the S register; (2) It initiates the basic Storage sequence within the storage unit.

The Initiate Storage Reference circuit is shown in figure 4-18. The address may originate from one of three sources, depending upon the sequence which initiates the reference.

Source of Address	Sequence
Program Address Register (P^1)	Read Next Instruction
Program Control Register (U^2)	Read Operand
	Write Operand
	Search and Transfer
	Iterative
	External Function
Buffer Control	Common Control Register

SEQUENCE CONTROLS

The two identical sequence controls generate a fixed sequence of control signals in response to the Initiate Storage Sequence signal (figure 4-19).

TIMING PULSE GENERATOR

A sequence control consists of a timing pulse generator, drive generators, pulse generators and fault detector. A sequence of timing pulses begins during the 6.4 microseconds immediately following the Initiate Storage Sequence signal. In figure 4-20 a loop of eight control delays (H061 - H068) generates the basic 8-pulse cycle; the pulses are separated by 0.2 microsecond.

A two-rank, two-stage counter is advanced by a signal from each group of control delays. During the first half of an 8-pulse cycle, the output of V062 transfers the count in rank I to rank II. During the second half of the cycle, the output of V066 transfers the count in rank II to rank I, advancing it by one. The counter is advanced through its cycle of four counts by four successive 8-pulse cycles.

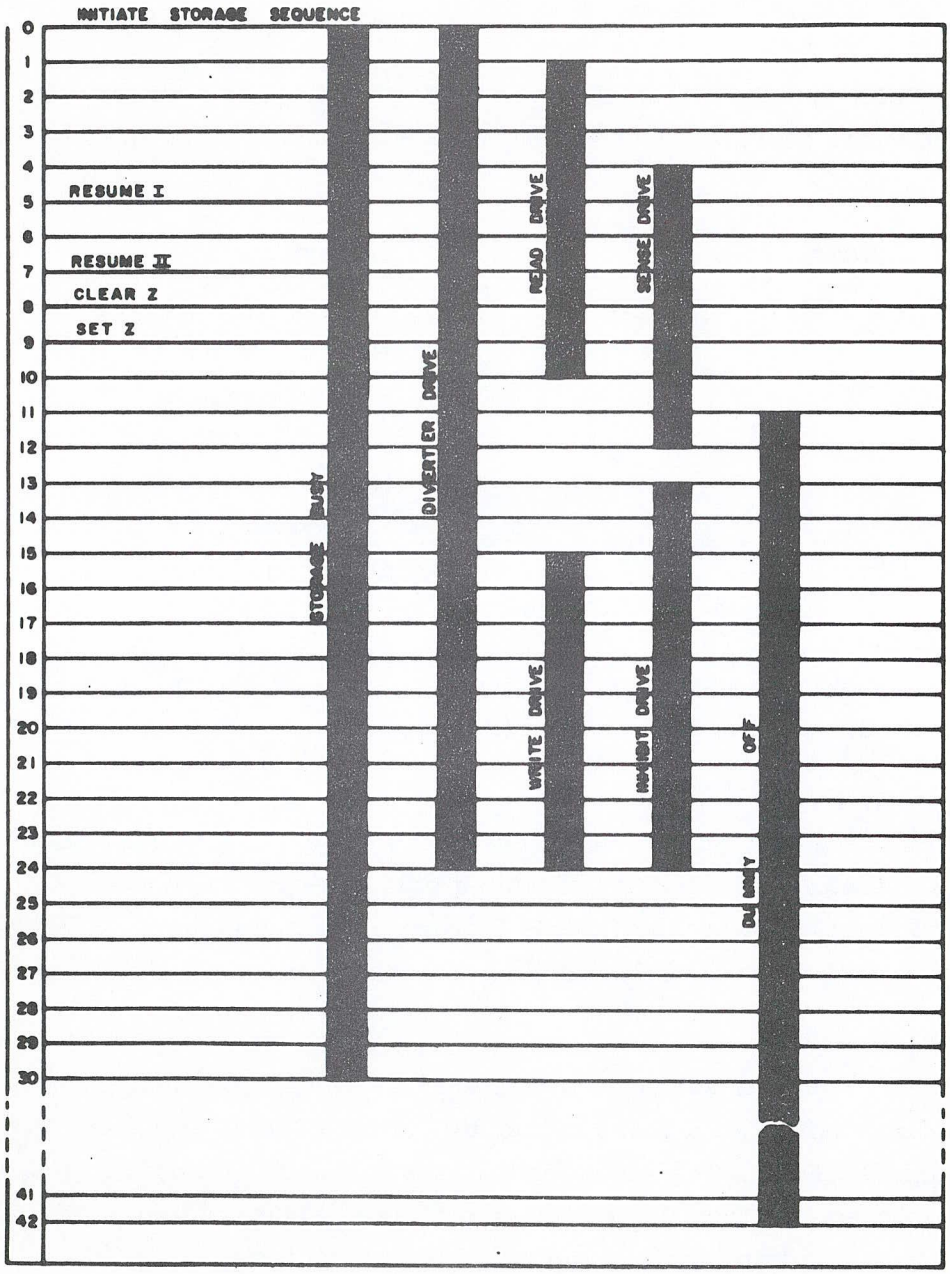


Figure 4-19. Sequence of Pulses Generated by Control Sequence

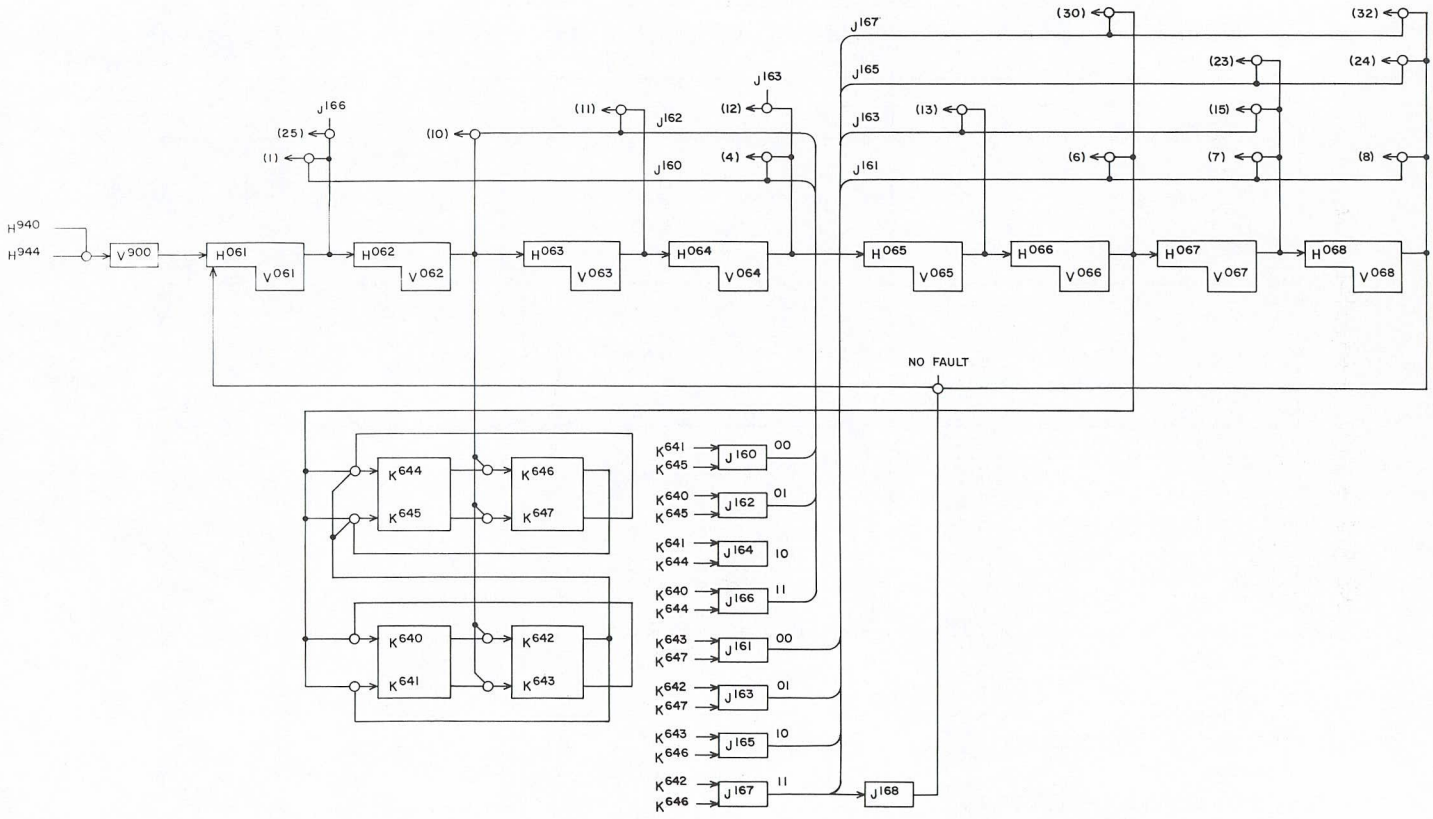


Figure 4-20. Timing Pulse Generator of Even Storage Sequence Control

TABLE 4-1. TIMING PULSE GENERATOR OF EVEN STORAGE SEQUENCE CONTROL: COMPLETE COUNTER CYCLE

Timing Pulse	Source	Rank I	Rank II
0	V^{900}	00	11
1	V^{061}	00	} Rank I → Rank II
2	V^{062}	00	
3	V^{063}	00	
4	V^{064}	00	
5	V^{065}	} Rank II + 1 → Rank I	00
6	V^{066}		00
7	V^{067}		00
8	V^{068}		00
9	V^{061}	01	} Rank I → Rank II
10	V^{062}	01	
11	V^{063}	01	
12	V^{064}	01	
13	V^{065}	} Rank II + 1 → Rank I	01
14	V^{066}		01
15	V^{067}		01
16	V^{068}		01
17	V^{061}	10	} Rank I → Rank II
18	V^{062}	10	
19	V^{063}	10	
20	V^{064}	10	
21	V^{065}	} Rank II + 1 → Rank I	10
22	V^{066}		10
23	V^{067}		10
24	V^{068}		10
25	V^{061}	11	} Rank I → Rank II
26	V^{062}	11	
27	V^{063}	11	
28	V^{064}	11	
29	V^{065}	} Rank II + 1 → Rank I	11
30	V^{066}		11
31	V^{067}		11
32	V^{068}		11

Table 4-1 lists the 32 steps of a complete counter cycle. During the first half of each 8-pulse cycle, rank I holds the count while rank II is being changed and during the second half of the cycle, rank II holds the count while rank I is being changed. A distributor, combining selected outputs of the control delays with the outputs of the counter, takes advantage of this feature. Four inverters, J160, 162, 164, 166, translate the outputs of rank I to provide negative outputs for counts 00, 01, 10, and 11; inverters J161, 163, 165, 167 provide the same translation for the outputs of rank II. The outputs of the first group of inverters are combined with the outputs of the first half of the timing chain; the outputs of the second group of inverters are combined with the outputs of the second half of the timing chain (table 4-2).

DRIVE GENERATORS AND PULSE GENERATORS

Each drive generator consists of a FF and two or more inverters (figure 4-21). The FF is set and cleared by timing pulses which coincide with the leading and trailing edge of the Drive signal. The inverters serve as slaves providing the Multiple Output signals required. The inverters are connected in parallel to either the "1" or "0" output. The pulse generators are control delays which produce Z Register commands and the storage resumes.

The following discussions refer to both the odd and even sequence controls. The logical designations used, however, are those of the even sequence control.

1) Diverter Drive. FF K624/625 is set by pulse 0 and cleared by pulse 24; during this interval, inverters J030/031 provide the positive Diverter Drive signal to the horizontal and vertical diverters of the Address Selection circuit.
2) Read Drive. FF K626/627 is set by pulse 1 and cleared by pulse 10; inverters J032/033 provide the negative Read Drive signal to the horizontal and vertical drivers of the Address Selection circuit.
3) Write Drive. FF K628/629 is set by pulse 15 and cleared by pulse 24; inverters J034/035 provide the negative Write Drive signal to the horizontal and vertical drivers of the Address Selection circuit.
4) Inhibit Drive. FF K630/631 is set by pulse 13 and cleared by pulse 24; inverters J036/037 provide the Inhibit Drive signal to the Inhibit circuits.
5) Sense Drive. FF K632/633 is set by pulse 4 and cleared by pulse 12. K632 provides a positive Sense Drive signal to inverters J330/331/332 and K633 provides a negative Sense Drive signal to inverters J333/334/335.

Inverter signals are controlled by Enable Partial Write (upper) and Enable Partial Write (lower). Enable Partial Write (lower) enables the set input gate of K612/613 which is set to "1" during the read half of the Storage sequence. Enable Partial Write (upper) enables the input gate to K614/615. The output of K613 is applied to the OR inputs of inverters J330/331 and to the AND inputs of inverters J333/334. The output of K615 is applied to the OR inputs of inverters J331/332 and to the AND inputs of inverters J334/335.

The absence of all of the Partial Write Enables from control results in negative signals from J330/331/332 during the period of the Sense Drive. These signals, Read Lower Address, Read Non-address and Read Upper Address, effect a read reference by transferring a complete word from the sense lines of the selected quadrant of a memory plane to I^5.

If both Partial Write signals are present, inverters J333/334/335 provide negative output signals during the period of the Sense Drive signals. These signals, Write Lower Address, Write Non-address and Write Upper Address, effect a write reference by transferring a complete word from the X register to I^5.

6) Dummy Drive. FF K620/621 is set by pulse 11 cleared approximately 6.2 usec later by pulse 42. A special chain of 10 FFs K160/161 through K178/179 delays pulse 32 for 10 periods to obtain pulse 42. Each FF provides approximately a 0.2 usec delay; the precise delay period provided by control delays is not necessary in this application since the timing of the Dummy Drive signal is not critical.

During the interval the Dummy Drive FF is set, inverters J038/039 provide negative Dummy Drive Off signals which disable the dummy loads of the horizontal and vertical drivers and selected dummy loads of the Inhibit circuits.

7) Clear Z. Control delays H990 and H992 receive timing pulses 7 and 25 providing Clear Z output pulses to the bit plane control. The output of H990 is sent to chassis 1, 2, 3 and 4; the output of H992 is sent to chassis 5, 6, 7 and 8.

8) Set Z. Control delays H991 and H993 receive timing pulse 8 and provide Set Z output pulses to the bit plane control. The output of H991 is sent to chassis 1, 2, 3 and 4; the output of H993 is sent to chassis 5, 6, 7 and 8.

TABLE 4-2. TIMING PULSE GENERATOR, SOURCE OF TIMING PULSES

Timing Pulse	Source	
	Inverter	Control Delay
0		V^{900}
1	J^{160}	V^{061}
4	J^{160}	V^{064}
6	J^{161}	V^{066}
7	J^{161}	V^{067}
8	J^{161}	V^{068}
10	J^{162}	V^{062}
11	J^{162}	V^{063}
12	J^{162}	V^{064}
13	J^{163}	V^{065}
15	J^{163}	V^{067}
23	J^{165}	V^{067}
24	J^{165}	V^{068}
25	J^{166}	V^{061}
29	J^{167}	V^{065}
32	J^{167}	V^{068}

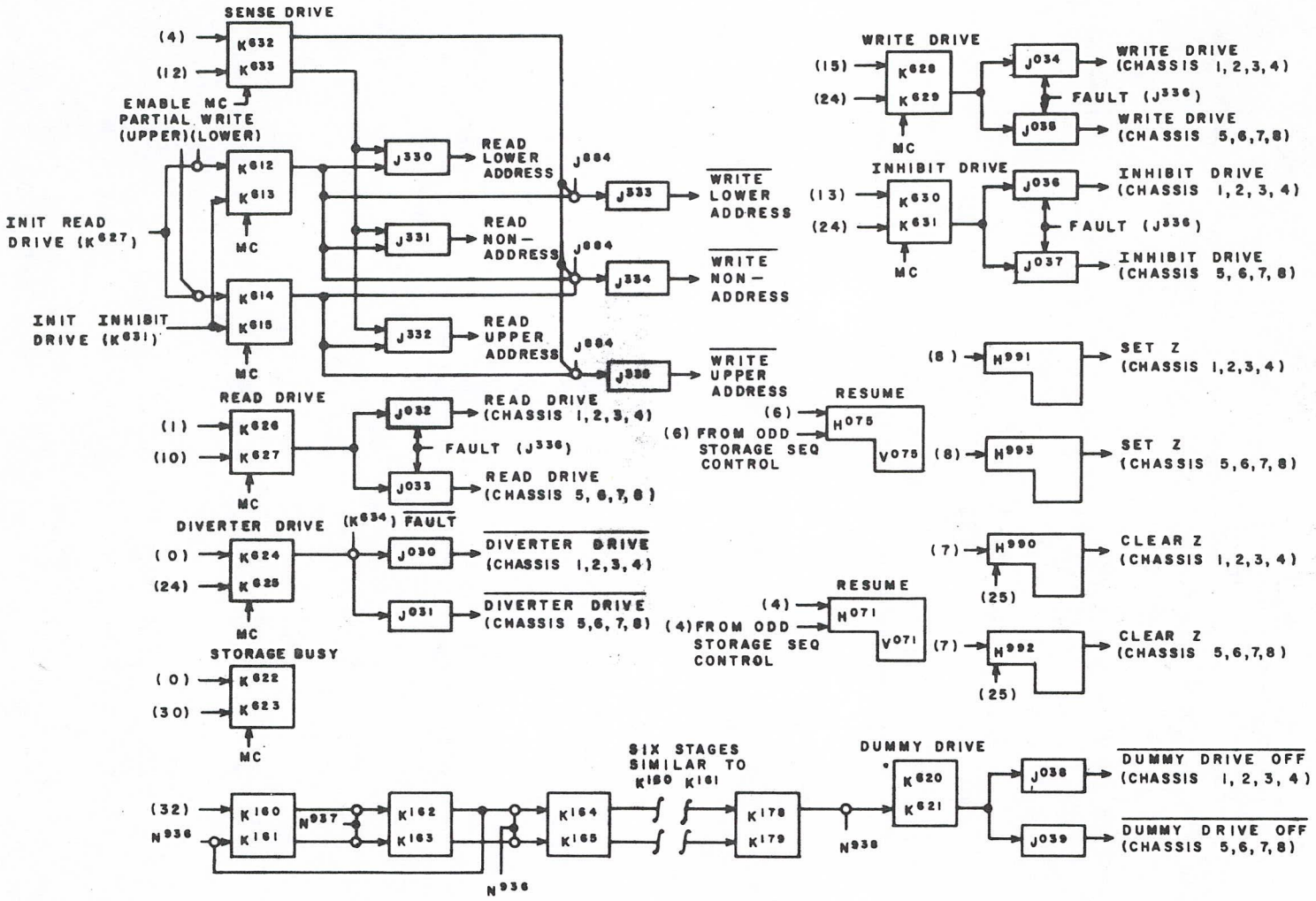


Figure 4-21. Drive and Pulse Generator of Even Storage Sequence Control

9) Resume. Two control delays, shared by the odd and even storage sequence controls, generate the Resume I and Resume II pulses. Control delay H071 receives timing pulse 4 from both the odd and even storage sequence controls and provides the Resume I pulse to the control section. Control delay H075 receives timing pulse 6 from both the odd and even storage sequence controls and provides the Resume II pulse to the control section.

FAULT DETECTOR

The fault detector recognizes the presence of more than one pulse in the timing loop of a sequence control. Such unwanted pulses cycle the drive generators on and off rapidly and overheat the generators. Since the first control delay of both even and odd timing loops is odd, only even-phase pulses can enter the loops. The fault detector of the even storage unit (figure 4-22) combines the outputs of control delay V063 and V065 at the AND input to the set side of FF K634/635. If two successive even pulses enter the loop, the FF will be set and the Storage Fault signal generated. If two pulses separated by the six phase periods enter the loop, the FF will be set during the second recirculation of the pulses (when they are separated by two phase periods).

When K634/635 is set, the AND input of J336 is not satisfied and generates a "1" which is recognized as the Storage Fault signal. This signal is applied to the slave inverters of the drive FFs to hold off the drive generators. A master clear from the console clears the drive FFs and K634/635. While the Clear lever switch is pressed, the AND input of J336 is inhibited by a "0" from J337. Thus, the drive generators are held off during the period of the master clear.

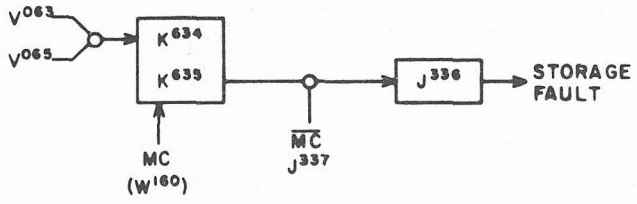


Figure 4-22. Fault Detector of Even Storage Unit

ELECTRONIC THEORY OF STORAGE CIRCUITS

The storage section performs non-logical functions such as amplification, pulse generation and switching. The card types are: drive generator type 51, diverter 52, selector 53, current source 54, inhibit generator 55, and sense amplifier 56.

DRIVE GENERATOR (Type 51)

The drive generator develops the read-write current applied to the H and V wires. Opposite ends of the primary winding of transformer T01 are fed by two channels consisting of transistor Q01, connected as an emitter-follower, and transistors Q02 and Q03, connected in parallel as amplifiers. The input signal is an AND combination of two selector outputs. A -1v input results in approximately 0v at the base of Q01. The emitter of Q01 is clamped to ground by CR03; neither Q02 nor Q03 conduct. Consequently, no current flows in the primary T01.

A -12v input signal causes Q01 to conduct; conduction is held below saturation by feedback diode CR01. The negative voltage across R06 applied to the bases of Q02 and Q03 causes these transistors to conduct. Current flows through the emitters of Q02 and Q03 to the collectors, through the primary of T01, to the current sources. The current pulse from the secondary of T01, amplified by the step-down action of T01, is applied to H and V wires of the memory plane assembly. It then flows through the selected diverter card and back to the secondary of T01. Current flow in the primary determines polarity of the output current from T01. Direction of current flow is determined by the selected channel. A -12v input to channel A generates a read pulse; a channel B input generates a write pulse.

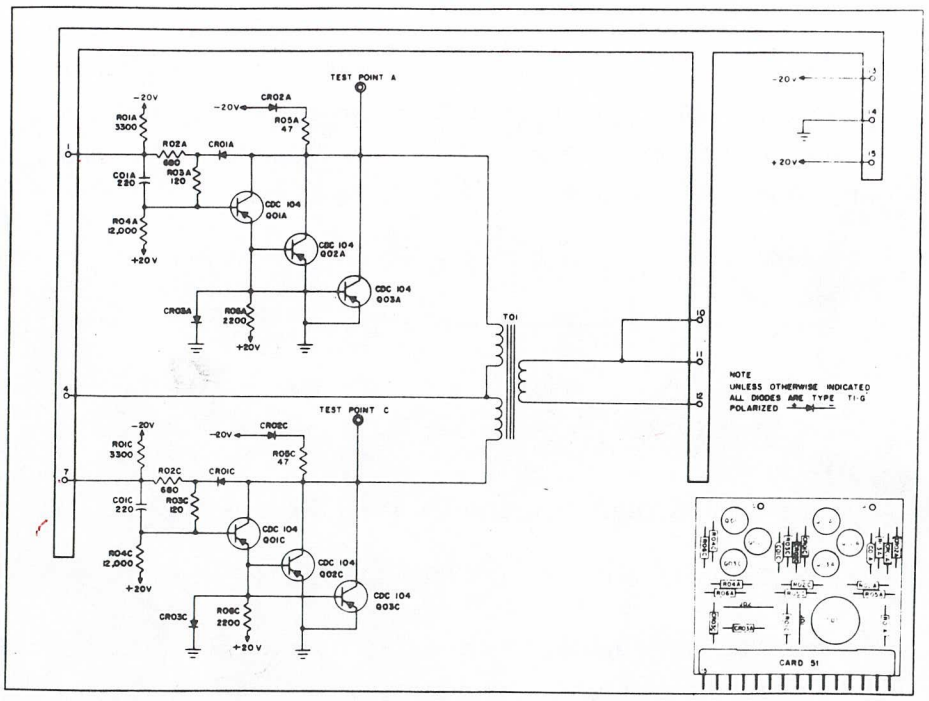


Figure 4-23. Drive Generator (Card Type 51)

DIVERTER (Type 52)

The diverter circuit serves as an electronic switch in series with an H or V wire of the memory plane assembly. Transistor Q01 is connected as an emitter follower, and transistors Q02 and Q03 as switches. A -3v input causes Q01 to conduct; the negative signal from Q01 enables Q02 and Q03. One or the other transistor passes the current pulse on the H or V wire to which the diverter is connected.

A positive pulse passes one of the pairs of diodes CR03/CR04, CR07/CR08, CR11/CR12, etc., depending upon the driver selection, and passes Q02. A negative pulse passes one of the pairs of diodes CR01/CR02, CR05/CR06, etc., and passes Q03. In either case, the current pulse is returned to the R/W driver. The bleeder networks of all diverters are connected in parallel via terminals 11 and 12 to equalize the current flow tnrough the bleeders and reduce heating.

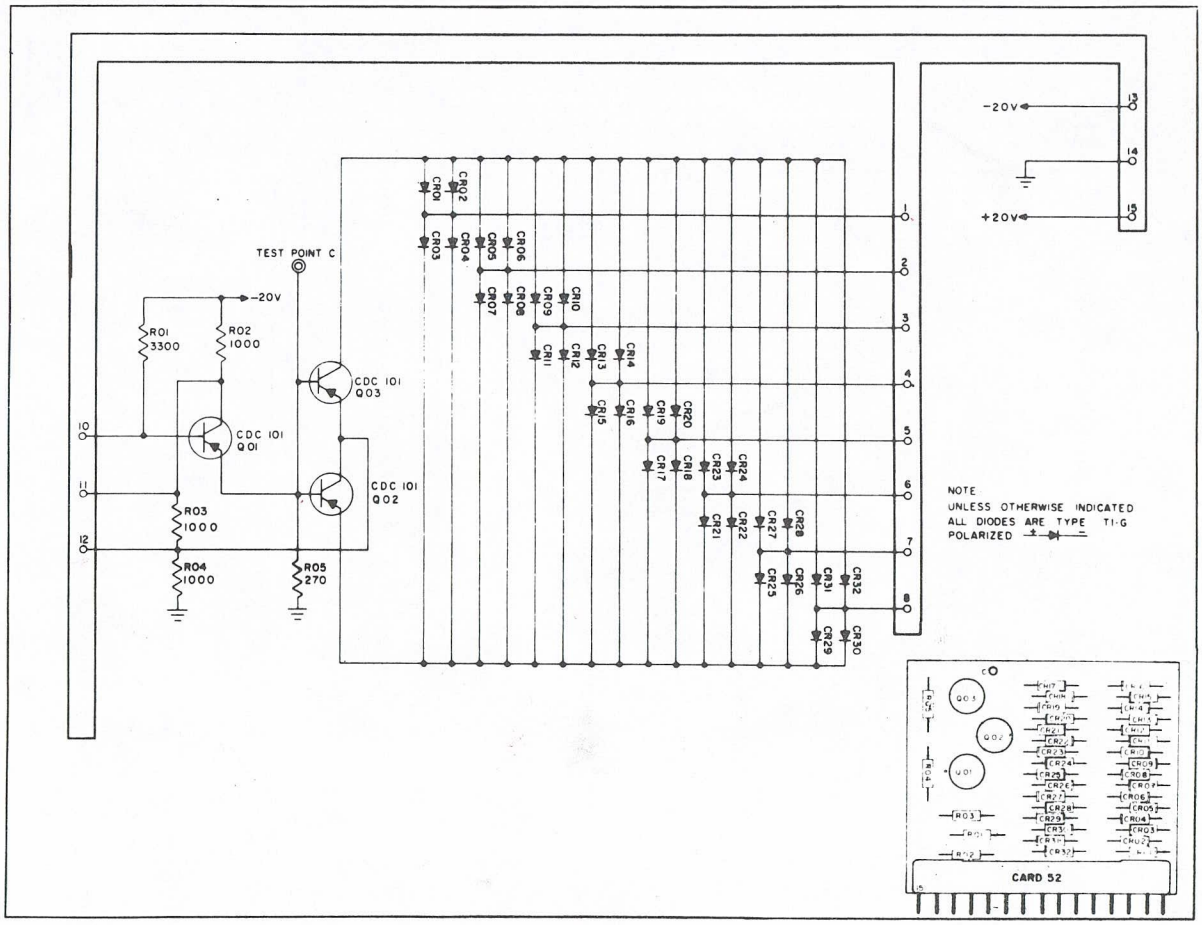


Figure 4-24. Diverter (Card Type 52)

SELECTOR (Type 53)

Each selector card consists of two identical selector circuits. A selector circuit is similar to the standard inverter except that the resistance results in output signal levels of -1v and -12v. Each selector circuit has two input diodes CR01/CR02 and four output diodes CR09/CR10/CR11/CR12.

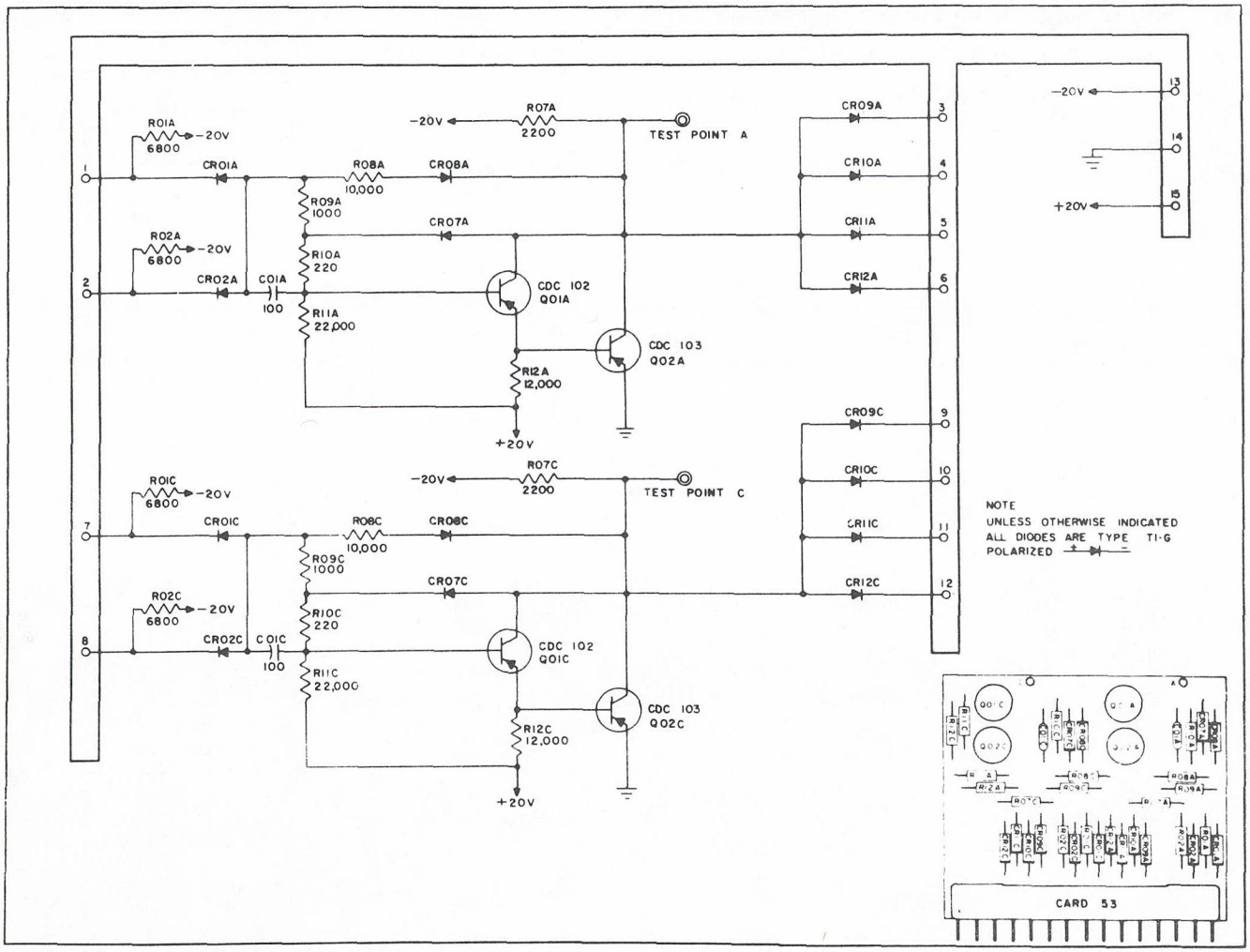


Figure 4-25. Selector (Card Type 53)

CURRENT SOURCE (Type 54)

The current source card consists of five banks of parallel resistors (four banks 150 ohms each and one bank 303 ohms). One end of each bank is connected to the -20v output of the power supply. The 150 ohm banks supply current to the H, V and inhibit current generators; the 303 ohm bank supplys current to the dummy loads.

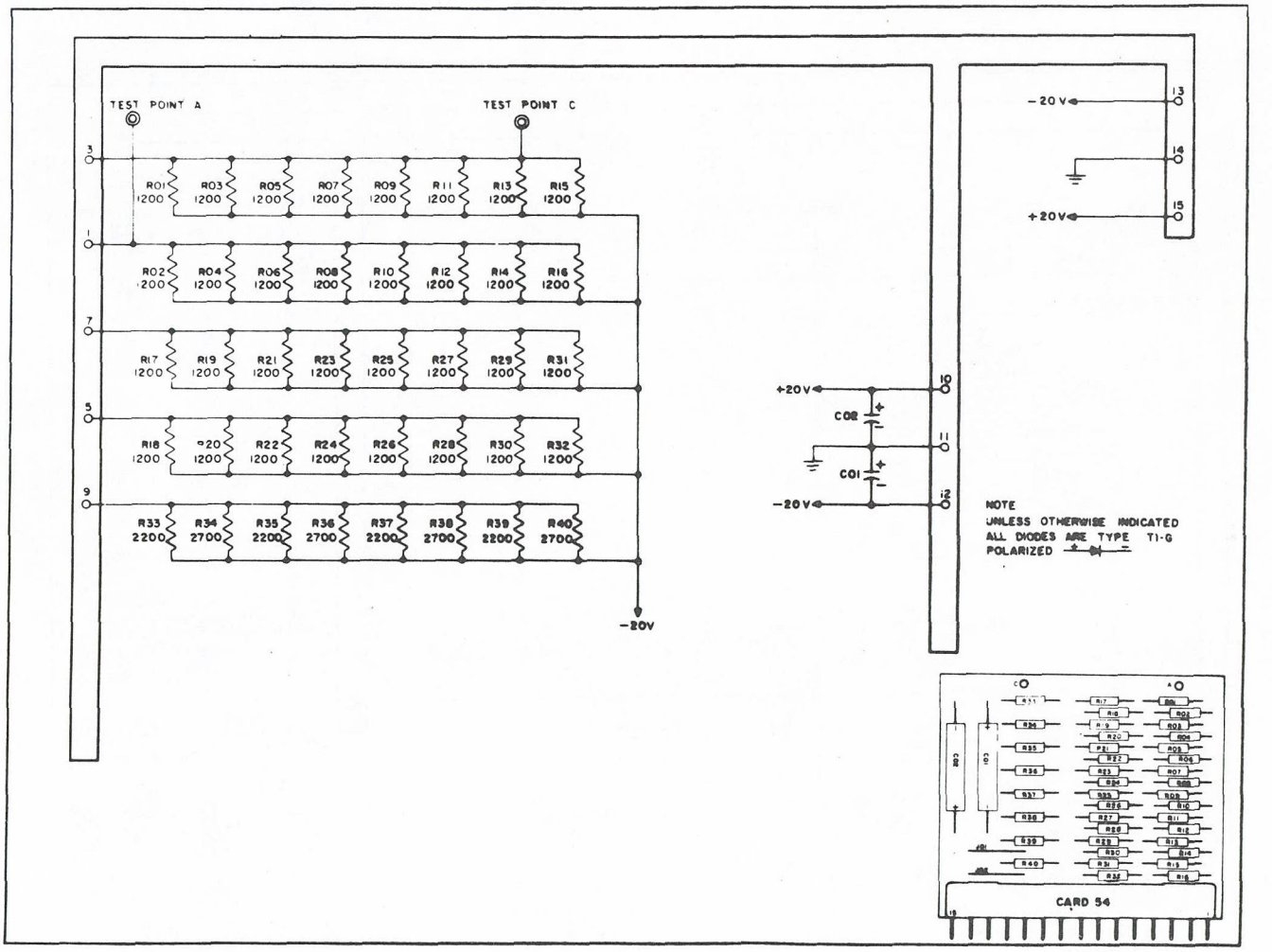


Figure 4-26. Current Source (Card Type 54)

INHIBIT GENERATOR (Type 55)

Each card has two generator circuits which are identical to the type 51 drive generator channels except for the absence of an output transformer. The output of each channel is independently connected to a terminal of the card.

A -12v input signal to either generator of a 55 card causes Q01 to conduct and thus enable Q02 and Q03. Current from the external source connects to the generator via terminal 6 or 12 and passes through Q02 and Q03 to ground.

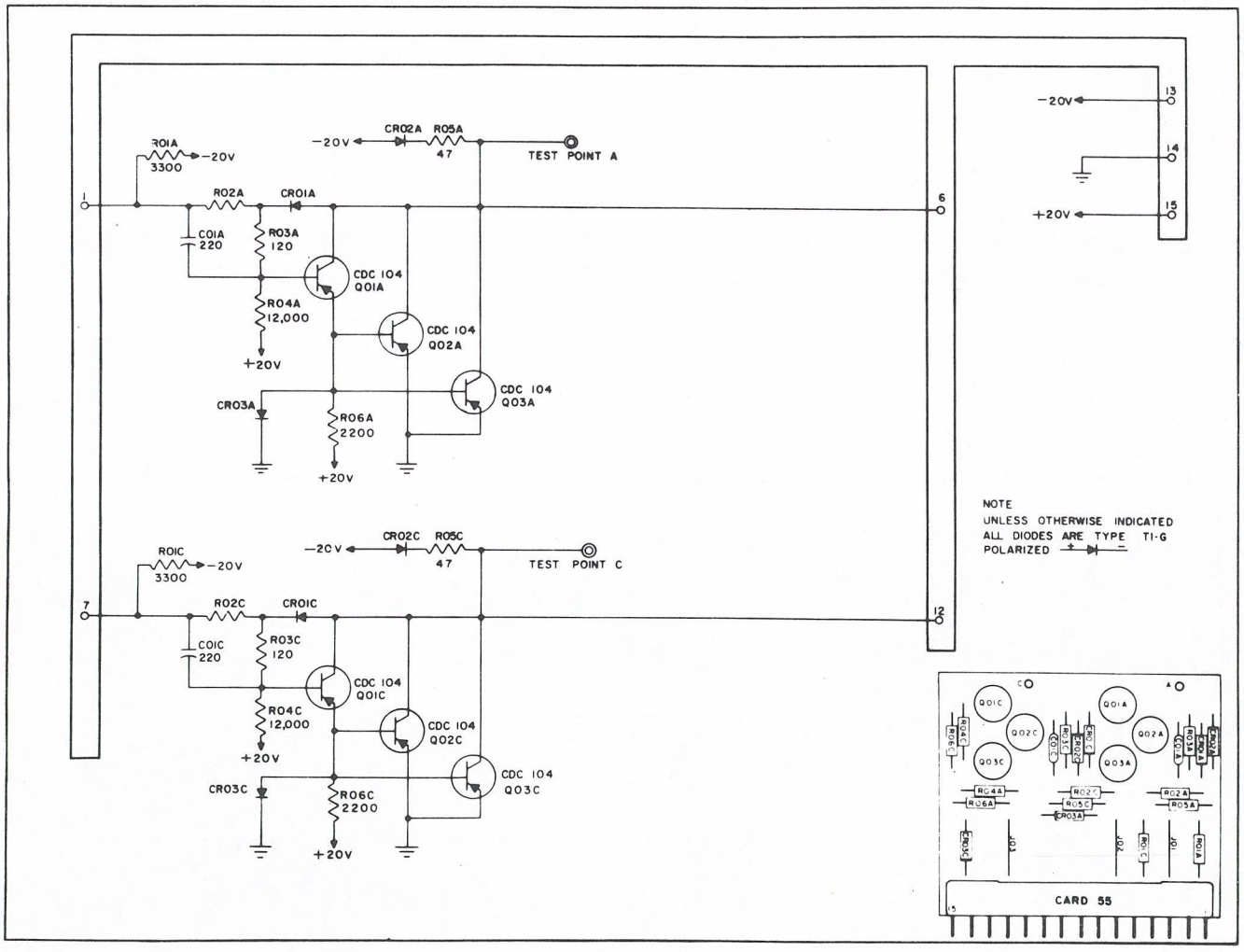


Figure 4-27. Inhibit Generator (Card Type 55)

SENSE AMPLIFIER (Type 56)

The 56 card amplifies the signals from a memory plane as the result of a read pulse. Transistors Q01 and Q04 are connected in a differential amplifier circuit. The signals from either end of the sense windings are applied to Q01 and Q03. The emitters are held at the difference voltage by a difference network composed of R04 and C01; noise voltages on the sense line are cancelled.

Capacitors C02 and C03, in the collector circuits of Q02 and Q04, provide d-c stabilization. Diodes CR01 and CR02 pass the negative-going components of the signals from Q02 and Q04, and serve as clippers. The bias across these diodes, and thus the clipping level, is adjustable by the Margin switch on the operator's console. When the switch is up, +20v is applied to the junction of R14 and R15 raising the reference voltage across the input diodes to the last stage. When the switch is down, -20v is applied to R14 and R15 raising the reference voltage across the input diodes of the last stage. The circuit is more sensitive to the signal from the sense line and spurious pulses tend to look like "1's". Transistors Q05 and Q06 are connected in an amplifier-inverter circuit. The output, a signal from CR05 as a result of a "1" signal from the sense wire, is -0.5v.

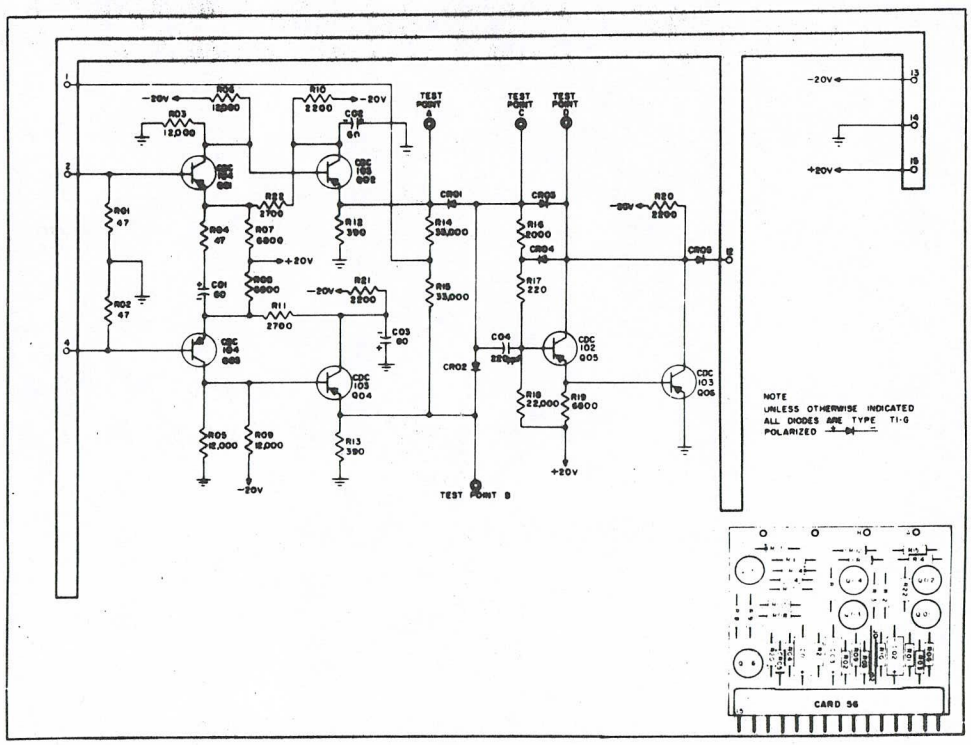


Figure 4-28. Sense Amplifier (Card Type 56)

CHAPTER 5
INPUT/OUTPUT SECTION

The Input/Output (I/O) section of the 1604-A provides the means of data transmission between the computer and the various pieces of external equipment. The 1604-A system is capable of handling all conventional input/output equipment. Three units of external equipment - a paper tape reader, a paper tape punch, and a modified electric typewriter - are an integral part of the computer console. A description of the operation of the three console I/O units is given in chapter 7 of this volume. Separate manuals describe the operation of other external equipment such as magnetic tape units, line printers, etc.

DATA TRANSMISSION

BUFFERING

Buffer communication is the transfer of individual words between computer storage and external equipment. In the 1604-A, buffer operations are independent of program operations; i.e., once a buffer is initiated by the main program, buffer operations and program operations proceed concurrently. Buffering requires one storage reference per word and must time-share storage with program operations. Buffering has priority in requests for storage time.

The 1604-A can handle buffer operations at the rate of one word every 7.4 usec but the actual speed of buffer operations is determined by the external equipment. The maximum data rate with one channel active is 116,000 words per second. With all six channels active the maximum rate is 22,500 words per second for each channel.

COMMUNICATION PATHS

External equipment communicates with the computer through four cable groups. All information enters or leaves the computer through one of these cable groups.

Three of the cable groups contain the three buffer channel pairs (one input and one output buffer channel in each group) and the fourth group is used for the bi-directional transfer channel (channel 7). Channel designations within a cable group are as follows:

Cable Group	Channel	
	Input	Output
1	1	2
2	3	4
3	5	6
4	7 Transfer	7 Transfer

Each cable group is composed of a set of six cables (each cable is made up of 24 twisted-pair common ground lines). A cable group contains one input channel and one output channel, one set of function lines and one set of control lines (see tables 5-1 and 5-2).

DATA AND CONTROL INFORMATION LINES

The Input/Output circuits are static, direct-coupled, and parallel. Data is presented on the communication lines as one of two d-c voltage levels; the binary "1" condition is represented by -0.5 vdc, the binary "0" condition is represented by -16 vdc. All binary digits of a word are presented simultaneously on the wires of the cable group.

A parallel set of connectors in each external equipment enables more than one external equipment to be connected to the computer via one cable group. Unique select codes provided by the computer determine which equipment attached to a cable group is to communicate with the computer.

TABLE 5-1. LINE ASSIGNMENT OF A GIVEN CABLE GROUP - DATA LINES

Input Equipment to Computer		
Input data (48 lines)		Comprise two complete cables and two lines of a third cable of a buffer cable group.
Input data ready (1 line)	Function:	Indicates to computer that input lines contain information which it may sample.
	Operation:	Turned off by Input Data Resume signal from computer. (Computer Resync circuitry orients itself about leading edge of Ready signal; auxiliary scanner is stopped and input word is passed to computer).
Computer to Input Equipment		
Input data resume (1 line)	Function:	Indicates to equipment that computer has accepted input word.
	Operation:	Turned off by Input Data Ready signal; turned on when computer has accepted and stored input word.
Computer to Output Equipment		
Output data (48 lines)		Comprise two complete cables and two lines of a third cable of a buffer cable group.
Output data ready (1 line)	Function:	Accompanies output data from computer.
	Operation:	Turned on when computer has word of information ready for equipment; off by Resume signal from equipment.
Output Equipment to Computer		
Output data resume (1 line)	Function:	Indicates that equipment has accepted word.
	Operation:	Turned on when equipment has accepted word. (Computer Resync circuitry orients itself about the trailing edge of Resume signal; when signal drops auxiliary scanner stops and another word is exchanged. Computer prepares a word for exchange while Output Data Resume signal is up).

TABLE 5-2. LINE ASSIGNMENT OF A GIVEN CABLE GROUP - CONTROL LINES

Computer to External Equipment		
External function (12 lines)	Function:	Carries external function (EXF) codes to select or sense a condition within the equipment.
	Operation:	Lines continuously monitored by all equipment. Function or Sense Ready enables equipment on a given channel to sample EXF code on lines.
External master clear (1 line)	Function:	Clears all equipment on all channels.
	Operation:	Occurs when Clear switch at console is in Up position.
External Equipment to Computer		
Sense response (1 line)	Function:	A logical "1" indicates presence of the exit condition specified by the sense code, if the lowest bit of the sense code is 0.
	Operation:	A logical "0" indicates presence of the exit condition if the lowest bit of the sense code is a 1.
Interrupt (2 lines)	Operation:	External equipment or internal computer control sends signal when an interrupt condition arises that was previously selected by a 74.0 instruction.
Computer to Input Equipment		
Input function ready (1 line)	Function:	Accompanies EXF Select code.
	Operation:	Turned on by instruction 74.0. Enables input equipment to translate EXF code.
Input sense ready (1 line)	Function:	Accompanies EXF Sense code.
	Operation:	Turned on by instruction 74.7. Enables input equipment to translate EXF code and send response back to computer.
Input buffer active (1 line)	Function:	Indicates computer is prepared to receive a block of data.
	Operation:	Turned on when input buffer channel is activated by instruction 74.1, 74.3 or 74.5. Remains on until final word of block is entered in storage and resynchronization of the input data ready occurs.

TABLE 5-2. (CONT'D)

Computer to Output Equipment		
Output function ready (1 line)	Function:	Accompanies EXF Select code.
	Operation:	Turned on by instruction 74.0. Enables output equipment to translate EXF code.
Output sense ready (1 line)	Function:	Accompanies EXF Sense code.
	Operation:	Turned on by instruction 74.7. Enables output equipment to translate EXF code and send response back to computer.
Output buffer active (1 line)	Function:	Indicates computer is prepared to transmit a block of data.
	Operation:	Turned on when output buffer channel is activated by instruction 74.2, 74.4 or 74.6. Remains on until final word of block is transmitted to output equipment and the output data resume signal is resynchronized by the computer.

BUFFER CONTROL

Buffer control is a specialized control section for controlling the operations of data transmission between computer storage and external and console equipment. The main parts of buffer control are:

Auxiliary sequence (AUX): times and initiates the commands of buffering.

High Speed Storage sequences (HSS): times and controls the special operations necessary for referencing storage locations 00001-00006.

Auxiliary scanner: the scanner is used to insure that each buffer channel has equal priority in buffering information.

Control registers (CR) 1-6: 30-bit FF registers used to store the address portions of storage locations 00001-00006 (buffer control words). The upper 15 bits (CR_U) hold the current address of a buffer operation. The lower 15 bits (CR_L) hold the terminal address.

Common Control register (CCR): a 30-bit FF register used to "update" the control word after each word is buffered.

Comparator: the comparator is used to compare the upper and lower address portion of the control words to sense when the buffer operation is complete. Input to the comparator is from the CCR.

Inverter ranks I^7 and I^8: used to provide more transmission paths in the buffer control section (figure 5-1).

U^2, P^1, and CCR_U translators: these translators sense when a storage reference is made to 00001-6. The translator outputs are used as gating conditions.

BUFFER OPERATIONS

Buffer operations of the 1604-A are initiated by the program but controlled thereafter by the buffer control section (figure 5-1). This leaves the main control section free to perform program operations at high speed while buffering takes place at the slower speed of the peripheral equipment.

In a buffer operation a block of 48-bit words is transmitted into or out of storage one word at a time. The block of words is defined by an initial address and a terminal address. The initial address is the storage location of the first word to be transmitted. The terminal address is one greater than address of the last word. A buffer operation involves the following steps:

Program Steps:

- Store terminal address in CR_L
- Sense for Channel Inactive
- Select equipment and mode
- Sense condition of equipment
- Activate buffer channel (stores initial address in CR_U)

Carried Out Automatically By Buffer Control:

- Transmit first word to or from storage
- Increase CR_U (initial address) by one
- Compare CR_U with CR_L
- Continue if $CR_U \neq CR_L$
- _____
- _____
- Transmit last word to or from storage
- Increase CR_U (address of last word) by one
- Compare CR_U with CR_L (CR_U now equals CR_L indicating that the buffer is complete)
- Signal from comparator ($CR_U = CR_L$) deactivates buffer channel

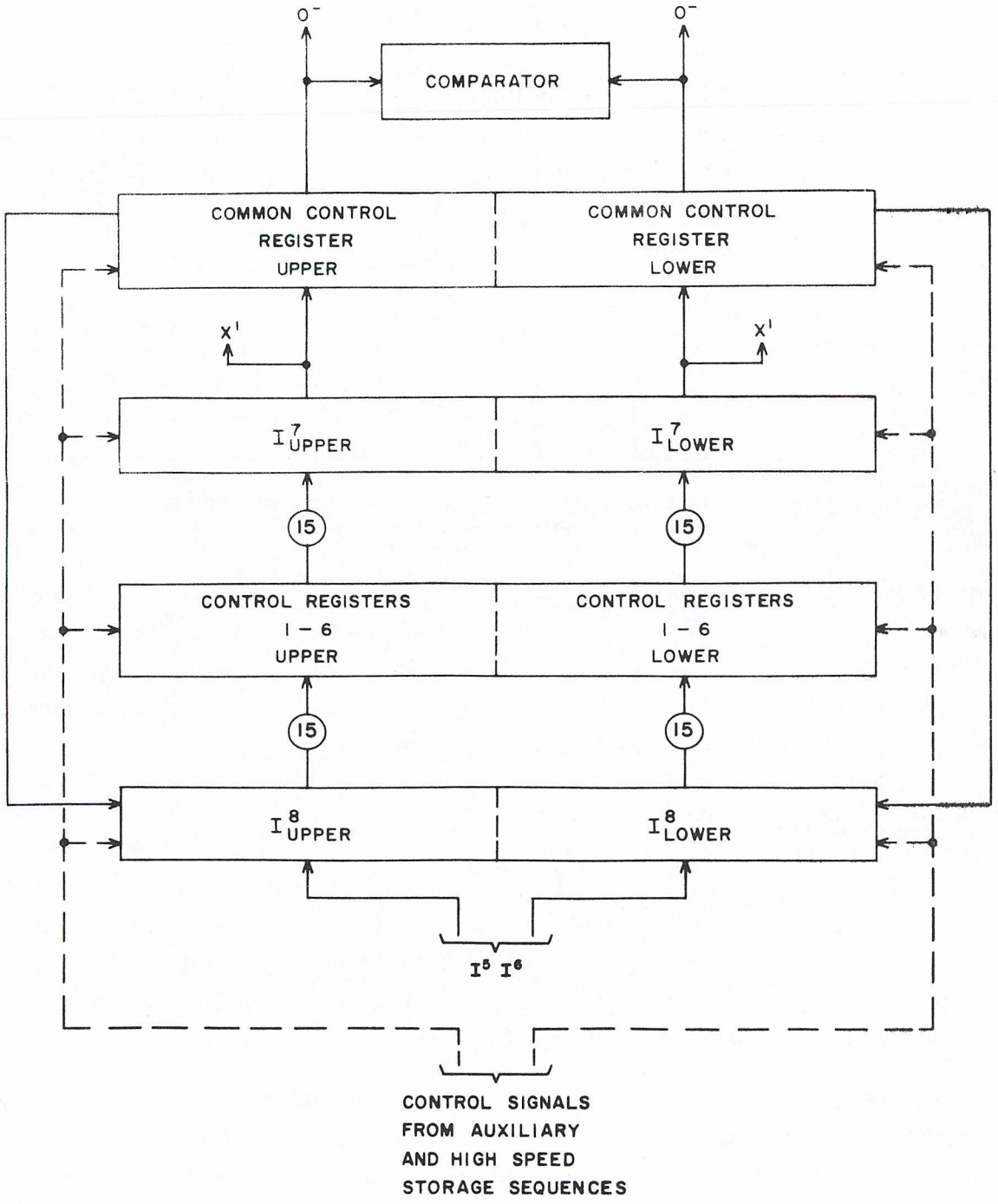


Figure 5-1. Block Diagram of Buffer Control

Store Terminal Address

Whenever storage locations 00001-00006 (buffer control words) are referenced (read from or written into storage) the corresponding control register (CR) is also automatically referenced. Thus the terminal address can be stored in the lower portion of the appropriate control register by any instruction that writes in the lower address (20, 21, 47, 57, 61). See figure 5-2.

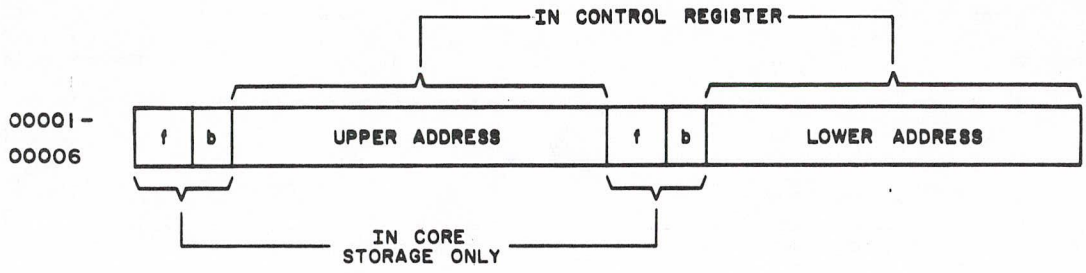


Figure 5-2. Structure of Control Words

Sensing for Channel Inactive

An instruction which selects a particular external equipment for operation on a buffer channel automatically de-selects all other equipments on that channel. Therefore, in order to avoid disconnecting a device before its operation is completed, it is necessary to sense for Channel Inactive (or Active) before executing a Select instruction. Sensing is done by the 74.7 instruction. Table 5-3 shows some 1604-A Sense codes and the conditions for which they sense.

The 74.7 instruction is executed by the EXF (external function) sequence. During a 74.7 instruction, the sequence (figure 5-4) sends the code contained in the execution address to all buffer channels and to computer control. Only the channel specified in the code receives a Sense Ready signal and is able to translate the code.

The Sense code (figure 5-3) specifies the condition to be examined; the presence or absence of the condition is indicated by the Sense Response signal.

TABLE 5-3. SENSE CODES*

CODE	DEFINITION
74 7 000C0 74 7 000C1	Exit On Channel C Active Exit On Channel C Inactive C = buffer channel number
74 7 001A0 74 7 001A1	Exit On Arithmetic Fault A Exit On No Arithmetic Fault A A = 1 = Divide fault A = 2 = Shift fault Etc.
74 7 00300 74 7 00301	Exit On Clock Overflow Exit On No Clock Overflow

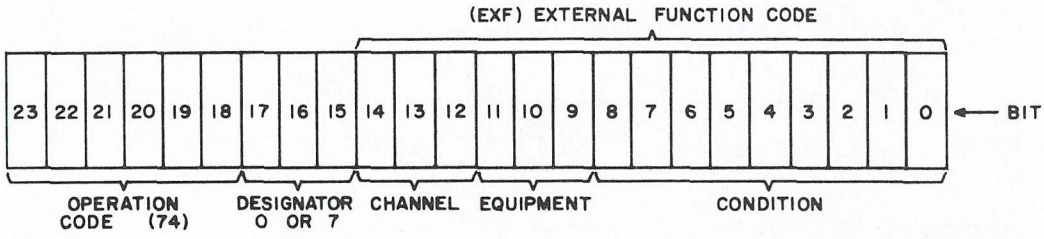


Figure 5-3. External Function Select/Sense Code

* A complete list of Sense codes can be found in the 1604-A peripheral equipment codes manual, publication number CDC 60025700.

For the 74.7 instruction, the following commands are generated:

1) $U^2 \rightarrow X^1$ with sign extension

2) $X^1 \rightarrow X^2$

   The 74.7 instruction is not contained in X^2_L. X^2_L is always connected to the external function inverters O^o as shown in figure 5-5; only the absence of a Sense/Function Ready signal prevents interpretation of the code by external equipment. The code is available to all channels but can be interpreted only by the channel specified by the upper three bits of the code (figure 5-3).

3) Set Sense FF K718/719

   Setting the sense FF starts the EXF counter. 1.6 microseconds later Sense Ready (J450) becomes a "1". J450 combines with a translation for the channel (page 62, Customer Engineering Diagrams) to send a Sense Ready signal to the channel specified in the code. The Sense Ready signal is held on the line for 12.8 usec to allow sufficient time for sampling by external equipment. Then the count in the EXF counter restarts the EXF sequence at H709.

4) Sense response from J914 is combined with the lowest bit of the EXF code.

5) The output of J916 is resynchronized.

6) A full or half exit is taken, depending on the sense response.

A pair of Sense codes is associated with each condition. The lowest bit of the code is not sent out but rather is combined with the response received from J914. For example, if the channel is active, the response received by J914 is a "1" whether the instruction sensed for Channel Active or Channel Inactive.

If the channel is inactive, J914 receives a "0". After this response is combined with the lowest bit, the output of J916 is a "1" if the code is 000X0 and the channel is active, or the code is 000X1 and the channel is inactive. The full exit is taken for these cases. When the output of J916 is a "0" the half exit is taken. The exit chosen indicates whether the specified condition is present.

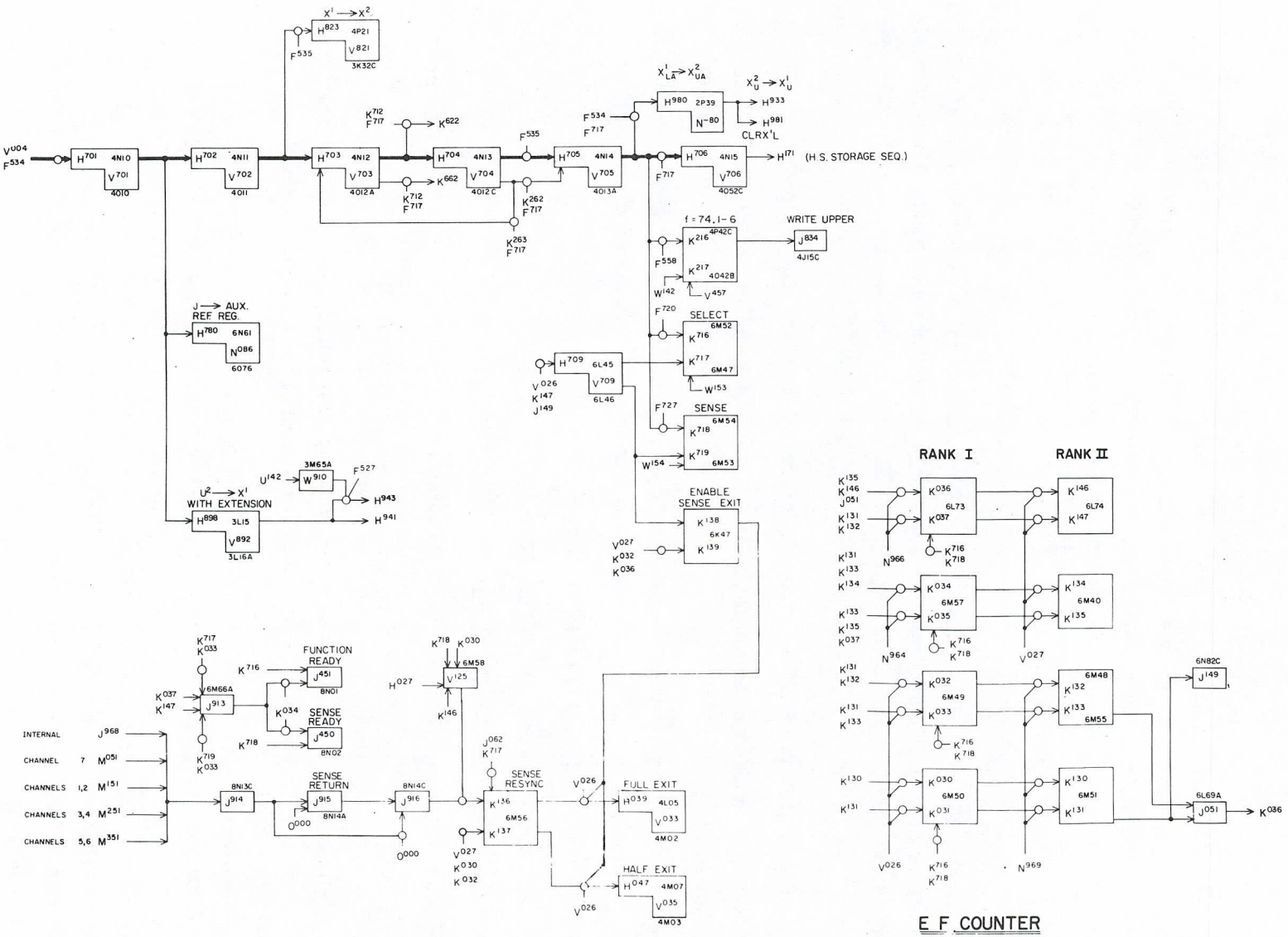


Figure 5-4. External Function Sequence

When the 74.7 instruction is used in the upper position, the full exit skips the lower instruction and the half exit leads to the execution of the lower instruction. When the 74.7 instruction is used in the lower position, the full exit leads to the next instruction when the specified condition exists. When the condition does not exist, the half exit causes the 74.7 to be repeated. Repetition of the instruction continues until the condition does exist.

Selecting Equipment and Mode

Before activating the buffer channel, the desired equipment must be selected and the desired mode of operation established. To select equipment and mode the 74.0 (Select) instruction is executed by the EXF sequence. During a 74.0 instruction the EXF Select code (figure 5-3) is transmitted to all channels just as the 74.7 code.

For a 74.0 instruction the EXF sequence performs the following operations:

1) $U^2 \rightarrow X^1$ with sign extension
2) $X^1 \rightarrow X^2$
   The EXF code is now on the lines (figure 5-5) and the Function Ready signal enables its translation by the specified equipment.
3) Set Select FF (starts EXF counter)
4) Set Sense Resync FF if Exit FF = 0
5) Issue Function Ready signal
6) Wait 12.8 usec (timed by EXF counter)
7) Full or half exit (chosen by state of Sense Resync FF).

By the time the Function Ready is dropped and the exit taken, the code has been translated and the proper equipment is selected for the desired mode of operation.

Sensing Condition of Equipment

In some cases the external equipment may not be ready for use even though the channel is inactive; i.e., a magnetic tape unit may be rewinding, the stacker on a card reader may be full, the typewriter may be in the wrong case, etc. EXF codes are used to determine whether equipment is ready for use and, if desired, wait until it is ready. The action of the EXF sequence for these codes is the same as described under sensing for Channel Inactive.

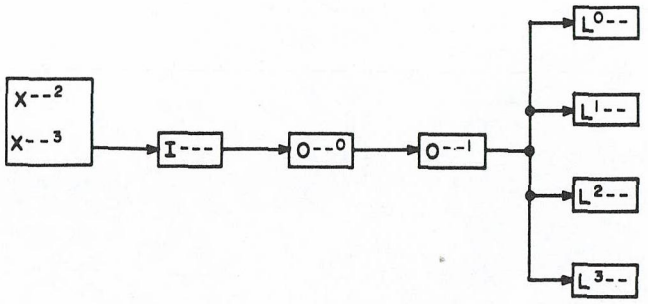


Figure 5-5. External Function Inverters

Activating the Channel

A 74.j instruction is executed by the EXF sequence to activate a buffer channel. The channel to be activated is specified by j (j = 1-6). The sequence of events is:

1) j → Auxiliary Reference register (ARR)
2) $U^2 \rightarrow X^1$
3) $X^1_{LA} \rightarrow X^2_{UA}$
4) Set even and odd Storage Busy FFs
   As shown in figure 5-4, if an auxiliary operation is in progress at this time, a loop provides for waiting until the auxiliary operation is completed.
5) Set K216/217 -f = 74.1-6 FF
6) Initiate high speed storage sequence

High Speed Storage Sequence

The High Speed Storage sequence is used to enter the initial address (address of the first word to be buffered) into a Control register. High Speed Storage sequence performs the following steps:

1) Set Storage Enable FF
2) $X^2_{UA} \rightarrow I^6$
3) Set Storage Busy FFs
4) ARR → ARD (translations from ARD select desired control register)

5)	$I^5\ I^6 \rightarrow I^8$	sends initial address to proper control register
6)	Set $CR_U \rightarrow$ "1's"	
7)	$I^8 \rightarrow CR_U$	
8)	Clear CCR	
9)	$CR \rightarrow I^7$	to compare initial and terminal addresses
10)	$I^7 \rightarrow CCR$	
11)	Exit	
12)	Set Buffer Active FF	
13)	Clear Storage Busy FFs	

At the end of the High Speed Storage sequence, the initial address is contained in the proper control register and the buffer channel is active. The actual buffering information is now under the control of the buffer control section. Except for sharing core storage, the buffer operation now proceeds independently.

Ready/Resume Logic (Output)

Once a buffer channel is activated, the computer and external equipment exchange control signals via the Ready/Resume circuitry. The ready/resume logic for channel 2 is shown in figure 5-6. When channel 2 is inactive the output of M149 is a "0"; K724/725 and K726/727 are clear. When buffer channel 2 is activated by the 74.2 instruction, Buffer Active FF K702/703 is set. When the next odd Resync pulse, V121, occurs, Action Request FF K786/787 is set, indicating to the auxiliary scanner that channel 2 is ready to buffer a word.

Auxiliary Scanner

The auxiliary scanner (figure 5-7) samples the six buffer channels in search of action requests from the buffer channels. The scanner is a specialized 3-bit, two-rank counter in which only one FF at a time changes state. Setting of FFs in rank II is dependent upon the Action Request FFs being clear. If an Active Request FF is set, the scanner stops when rank I holds the number of the channel for which the Action Request signal is present.

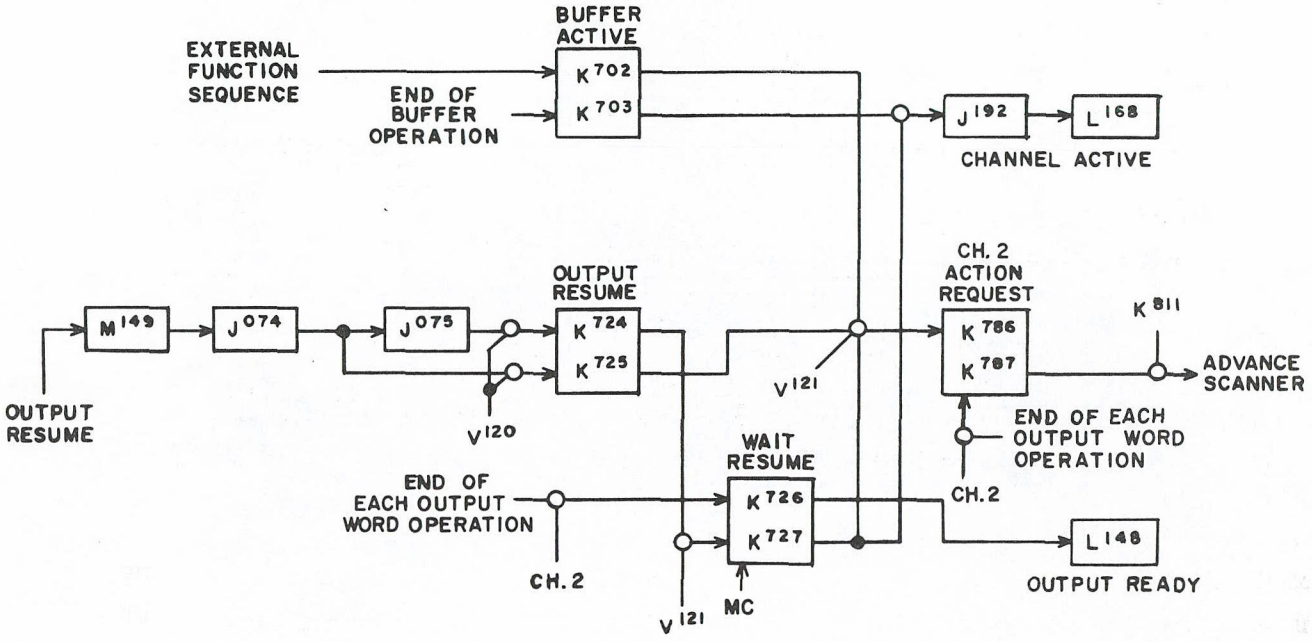


Figure 5-6. Ready/Resume Logic - Channel 2

The Advance Scanner pulses V791 and V793 alternately set and clear K822/823 every odd phase time while there are no action requests. The state of K822/823 is gated to K824/825 during even phase time and from there to K826/827 the next odd phase time, etc. K822/823 and K826/827 are in opposite states (one set and the other clear) as long as the scanner continues to advance. When an Action Request stops the scanner, K822/823 and K826/827 assume the same state (set or cleared) and the Auxiliary Request FF K712/713 is set. A "1" from K713 sends the contents of rank I of the scanner (number of channel with Action Request) to the Auxiliary Reference Designator (ARD), a three bit register. The output of the ARD is translated by inverters W821 - W826 and W831 - W836. The inverter outputs are used to establish gates for the Auxiliary and High Speed Storage sequence operations.

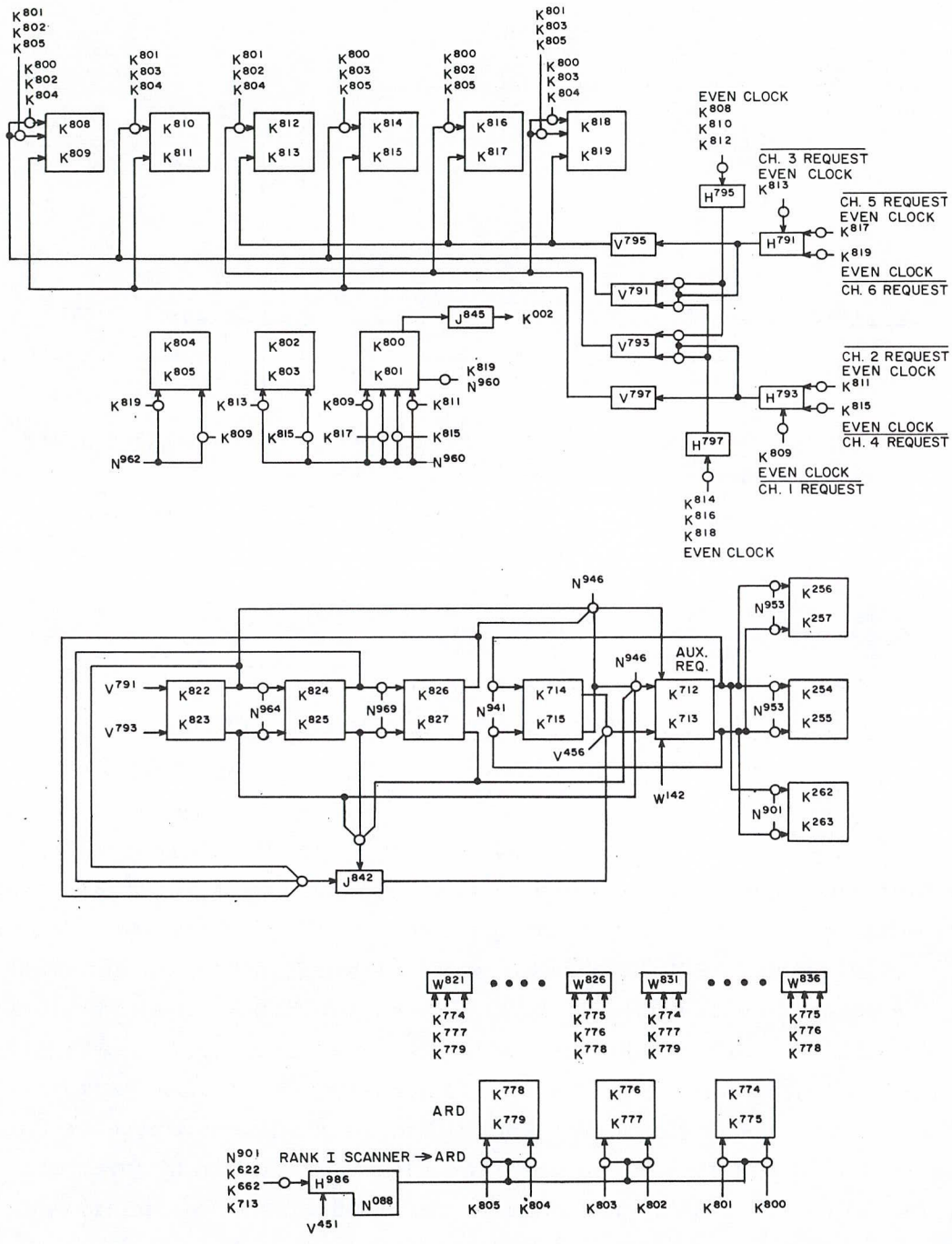


Figure 5-7. Auxiliary Scanner

Auxiliary Sequence

The Auxiliary sequence (AUX) is initiated by an Auxiliary Request (K713). AUX then times and issues the commands which buffer a word to or from storage.

The Auxiliary sequence issues the following commands:

1)	Set Storage Busy FFs	
2)	Clear CCR $CR \rightarrow I^7$ $I^7 \rightarrow CCR$	Sends control word to CCR
3)	$M \rightarrow I^0$ $I^0 \rightarrow I^5\ I^6$	Accepts input for input buffer
4)	Clear O^-	Prepare output register for new word
5)	Set $S^1\ S^2 \rightarrow$ all "1's" $CCR_U \rightarrow S$	Sends current address to S to make storage reference
6)	Initiate Storage	
7)	$CCR \rightarrow I^1$ Initiage H. S. Storage	If a buffer control word is being buffered, the address portion must come from or go to a CR
8)	$I^7 + 1 \rightarrow CCR_U$	Increase current address
9)	Set $CR_U \rightarrow$ "1's" $CCR_U \rightarrow I^8$ $I^8 \rightarrow CR_U$	Send updated current address back to CR.
10)	$I^1 \rightarrow O^-$	Put word in output register for output buffer
11)	$CR \rightarrow I^7$ Clear CCR $I^7 \rightarrow CCR$	Take control word to CCR for comparison of addresses. If $CCR_U = CCR_L$, the buffer is completed.
12)	Clear Wait FF (for input) or set Wait FF (for output) Clear Request FF Clear Buffer Active FF only if $CCR_U = CCR_L$	
13)	Release scanner	
14)	Release storage	

Ready/Resume Logic (Input)

In the case of an output buffer on channel 2, when AUX sets the Wait Resume FF K726/727 (figure 5-6) the Output Ready signal L148 is sent to the external equipment on channel 2. The Output Ready indicates to the external equipment that a data word is on the line. The external equipment then accepts the word at its own speed.

When the external equipment has accepted the word it sends an Output Resume to the computer. The Output Resume causes the computer to set K724/725, clear K726/727, and drop its Output Ready signal. When the Output Ready signal is dropped the external equipment drops its Output Resume. When the Output Resume goes down, K724/725 is cleared and another Action Request is enabled. Ready/Resume circuitry for input channels (figure 5-8) differs from figure 5-6 in that the signal from external equipment (Input Ready) must come before the Auxiliary sequence is initiated, whereas the Output Resume signal from output equipment comes after the Auxiliary sequence has put a word on the output lines.

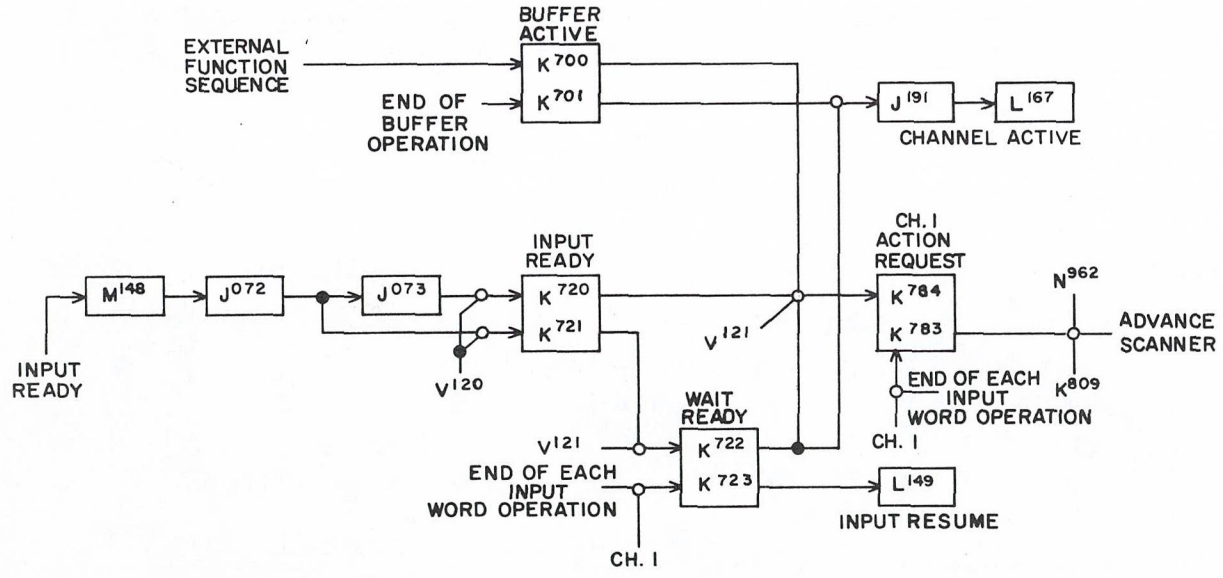


Figure 5-8. Ready Resume Logic - Channel 1

Buffering To or From 00001-00006

During buffer operations the content of Control registers is changed but the corresponding locations in core storage are not changed. Therefore, when a word is to be buffered to or from one of the buffer control words (00001-00006), the corresponding CR must also be referenced. To do this the Auxiliary sequence initiates the High Speed Storage sequence (HSS) when the CCR equals 00001-00006. HSS then issues the commands necessary to read from or write into a CR. Since the control register referenced by HSS is selected by a translation of the ARD, the address of the word being referenced is transferred from S^1 or S^2 to the ART during the time that the actual transfer of data takes place. The commands issued are:

1)	Rank I Scanner → ARD	Translations from ARD are used to select CR corresponding to the channel in use.
2)	Clear CCR	
3)	$CR \rightarrow I^7$ $I^7 \rightarrow CCR$ $CCR \rightarrow S^1$ or S^2	Sets S to address of current buffer operation (address to which or from which a word is being buffered.
4)	$CCR \rightarrow I^1$	If address of current buffer operation is 00001-00006.
5)	Initiate HSS	
6)	$I^7 + 1 \rightarrow CCR_U$ Set $CR_U \rightarrow$ 1's $I^8 \rightarrow CR_U$	Increase current buffer address and restore to CR.
7)	$S^1\ S^2 \rightarrow$ ARD	S must be transferred to ARD in order to get a translation to select the CR to which or from which a word is to be buffered.

AUX now does nothing while HSS buffers a word into or out of a CR.

8)	Rank I Scanner → ARD	AUX must now compare CR_U with CR_L to determine whether the buffer is complete (last word transferred).
9)	$CR \rightarrow I^7$ Clear CCR $I^7 \rightarrow CCR$	The CR corresponding to the channel being used is now in the CCR available for checking by the comparator.

During an output buffer of a control word the address portions are taken from the CR and the non-address portion from core storage. During an input buffer the entire word is written into core storage and the address portions are also stored in a CR.

TRANSFER

Transfer instructions 62 (Input Transfer) and 63 (Output Transfer) are executed by the Search and Transfer sequence (S & T). During the S & T sequence all other operations are halted (except previously initiated buffers). For this reason Transfer instructions (capable of transferring approximately one word every 5 usec) are normally used only to communicate with equipment capable of very high speed. Transfer instructions use channel 7 for both input and output.

The S & T sequence (figure 5-9) consists of two main parts, preparation and loop, plus provision for temporarily stopping to allow processing of interrupt or clock requests.

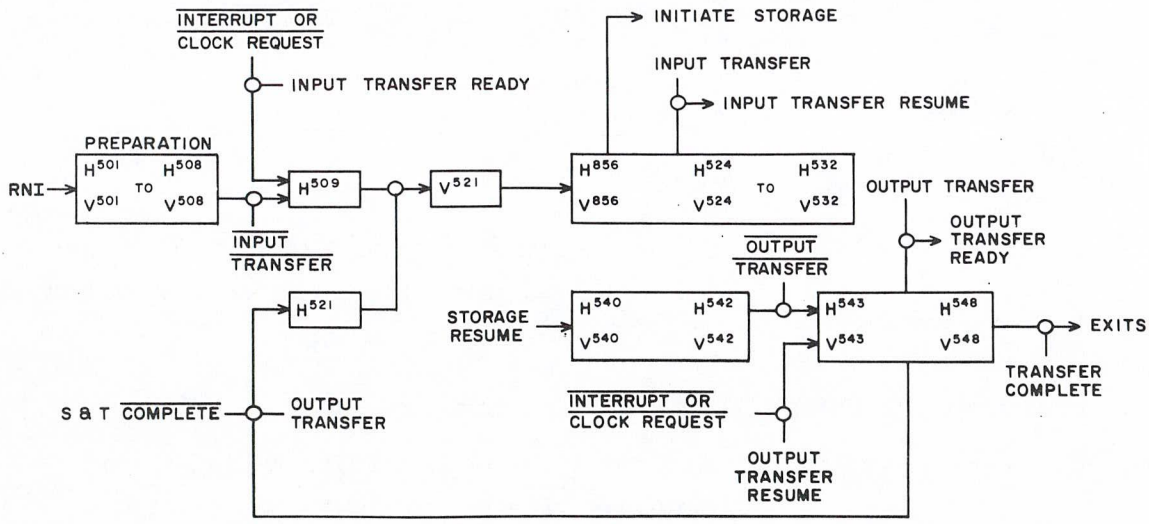


Figure 5-9. Block Diagram of Search and Transfer Sequence

PREPARATION

In a 62 or 63 instruction the number of words to be transferred is specified by (B^b). The first word is transferred to or from storage location m + (B^b) - 1; the last word to or from location m. The preparation part of S & T issues the following commands:

1) $(B^b) \rightarrow R$ — Done by RNI

2) Reduce $R^1 \rightarrow R^2$; Transfer $R^2 \rightarrow R^1$ — Form (B^b) -1

3) Set Input Transfer Active FF

4) Add $R^1 \rightarrow U^2$ — Form first address - m + (B^b) - 1

The loop portion of S &T issues a series of commands to transfer (or search) one word. The loop is repeated once for each word. As one word is transferred the loop also prepares another address.

INPUT TRANSFER - 62

After the first address of an Input Transfer instruction (62) is formed by the preparation part of S & T, the computer must wait for the Input Transfer Ready signal. The Input Transfer Ready initiates the loop at H509 (figure 5-9). The loop then takes the word from channel 7 data lines and stores it in core storage. The sequence of command is:

1) Clear X^1 — Prepares X^1 to receive first word.

2) Clear B — Prepares to send reduced word count back to B^b.

3) Clear Input Transfer Active FF K770/771.

4) Initiate Storage

5) Send Input Transfer Resume — The Resume causes the external equipment to drop its Input Transfer Ready. When the Ready goes down, the Input Transfer Resume goes down and the S & T sequence is prepared to receive another Input Transfer Ready.

6) $U^1 \rightarrow U^2$ — Restore original m to U^2

7)	$X^1 \rightarrow X^2$	Clears X^2
8)	$R^2 \rightarrow B$	Sends reduced word count to B^b
9)	Set R ≠ 0 FF K410/411 Clear S & T Complete FF	Prepares to check if S & T complete
10)	$M^0 \rightarrow X^2$	Take word from data lines - channel 7
11)	$X^2 \rightarrow X^1$	Sends word to X^1; from X^1 it goes to storage.
12)	Set S & T Complete FF	
13)	Reduce $R^1 \rightarrow R^2$	
14)	$R^2 \rightarrow R^1$	Prepare next address
15)	Add $R^1 \rightarrow U^2$	
16)	Storage Resume	
17)	Clear Wait Storage FF	
18)	Set Input Transfer Active FF	If R ≠ 0
19)	If transfer complete, exit	
20)	If transfer incomplete, wait for Input Transfer Ready signal	

Interrupt or Clock Request During 62

If an interrupt or clock request occurs during a 62 instruction, the next Input Transfer Ready signal is unable to initiate the loop at H509. At this time the next address of the transfer has been formed but (B^b) has not been changed. After the interrupt or clock request has been processed, the Transfer instruction is read again by RNI. The preparation part of S & T forms the address of the next transfer, sets the Input Transfer Active FF, and the transfer resumes at the point at which it was halted.

OUTPUT TRANSFER - 63

For an Output Transfer instruction (63) the preparation part of S & T is the same as for 62. The S & T sequence continues from preparation and the sequence of command is:

1)	Clear X^1	Prepare X^1 to receive word from storage.
2)	Clear B	Prepare B^b to receive reduced word count.
3)	Initiate Storage	
4)	$U^1 \rightarrow U^2$	Restore original m to U^2.
5)	$X^1 \rightarrow X^2$	Clears X^2.
6)	$R^2 \rightarrow B$	Place reduced word count in B^b.
7)	Set R ≠ 0 FF Clear S & T Complete FF	Prepare to check if transfer complete.
8)	Set S & T Complete FF	If R ≠ 0.
9)	Reduce $R^1 \rightarrow R^2$ $R^2 \rightarrow R^1$ Add $R^1 \rightarrow U^2$	Prepare next address
10)	Storage Resume	
11)	$X^1 \rightarrow X^2$	Transfer word from storage to X^2

At this point in the sequence the timing depends upon the output transfer ready/resume timing. For the first word of an output transfer the sequence continues without pausing. For subsequent words the Output Transfer Resume signal must have been received and turned off* before S & T may continue.

12)	$X^2 \rightarrow O^4$	Puts word on channel 7 output data lines.
13)	Restart loop at H521 if transfer not complete.	
14)	Send Output Transfer Ready signal.	
15)	Exit if transfer complete.	

*When an Output Transfer Resume is received the Output Transfer Ready is turned off. Turning off the Ready causes the external equipment (or other computer) to turn off its Resume. Loss of the Resume signal enables the S & T sequence to continue.

Interrupt or Clock Request During 63

If an interrupt or clock request occurs during an Output Transfer instruction, the Output Transfer Resume is disabled near the end of the loop. At this time X^2 holds a word that has not been transferred and R has been reduced in preparation for the next word. (B^b) must therefore be increased by two to provide for returning to the proper word when the transfer is continued. This is done by the logic shown in figure 5-10.

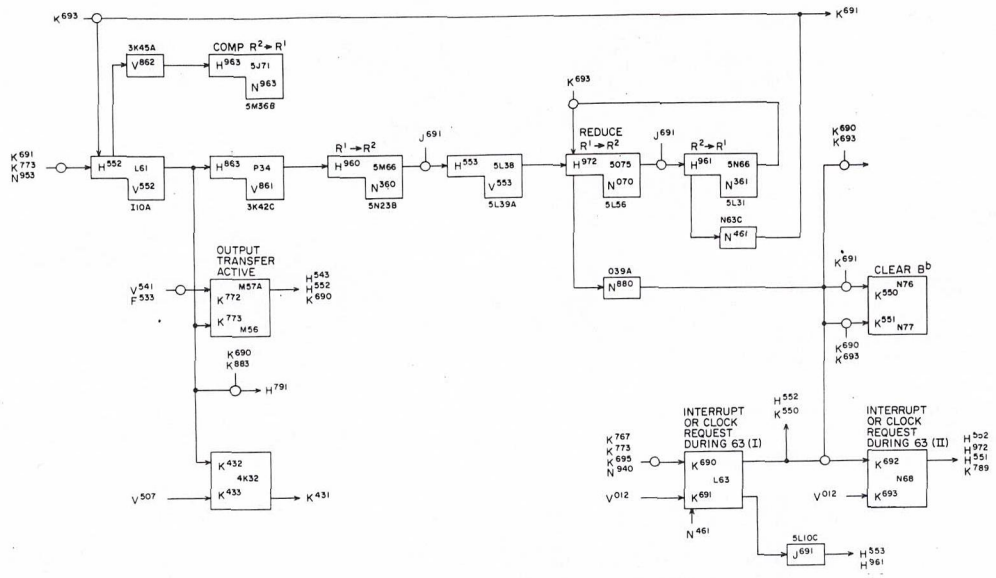


Figure 5-10. Temporary Termination of Search and Transfer Sequence During 63

R is increased by complementing, reducing twice, and recomplementing. This increased value of R is sent to B^b by the interrupt sequence.

After the interrupt or clock request has been processed, RNI obtains the Transfer instruction from storage again. The value of (B^b) provides for resuming the transfer at the point at which it was halted.

CHAPTER 6
INTERRUPT REAL-TIME CLOCK

The main program of the 1604-A may be temporarily halted in order to process requests by the real time clock or interrupt logic.

INTERRUPT

In each external equipment as well as in the computer itself certain conditions may arise which make it necessary to interrupt the main program. These conditions may be arithmetic faults, error in reading from or writing onto magnetic tape, etc. When these conditions occur the main program is halted and a routine of instructions (interrupt subroutine) is performed which determines the cause of the interruption and takes appropriate action to correct the condition. The signal which notifies the computer of the presence of these conditions is called an interrupt.

The program determines whether the occurrence of a given condition is to be indicated by an interrupt. An interrupt is produced by the presence of a condition only if a 74.0 instruction has previously selected an interrupt on that condition.

An interrupt indicates to the computer that the condition causing the interrupt is on a particular communication channel or is in the internal logic of the computer. The computer then executes an interrupt subroutine to locate the specific cause of the interrupt and correct it. After the interrupt subroutine is run, the main program continues from the point at which it was interrupted.

TYPES OF INTERRUPTS

Interrupts in the 1604-A are divided into two parts, internal and external. In addition, each interrupt source has its own interrupt address as shown in Table 6-1.

TABLE 6-1. INTERRUPT LOCATIONS

DESCRIPTION	LOCATION
Internal (Arithmetic)	00007
External - Ch. 7 Input	00010
External - Ch. 1	00011
External - Ch. 2	00012
External - Ch. 3	00013

TABLE 6-1. (Cont'd)

DESCRIPTION	LOCATION
External - Ch. 4	00014
External - Ch. 5	00015
External - Ch. 6	00016
External - Ch. 7 Output	00017

When an interrupt occurs the P register is automatically set to the address corresponding to the source of the interrupt as explained later in this chapter.

Internal Interrupt

There are six conditions which may cause an internal interrupt:

Divide Fault
Shift Fault
Overflow Fault
Clock Overflow Fault
Exponent Overflow Fault
Exponent Underflow Fault

When one of these conditions occurs the corresponding fault FF is set (figure 6-1). In order for these conditions to cause an interrupt the Arithmetic Interrupt FF, K242/243, must be set by a 74 0 00100 instruction (allow internal interrupts). Setting K242/243 enables any of the fault flip-flops to set the Internal Interrupt Request FF K796/797.

Regardless of whether internal interrupts are selected (allowed), the six internal fault conditions may be sensed for by 74.7 instructions.

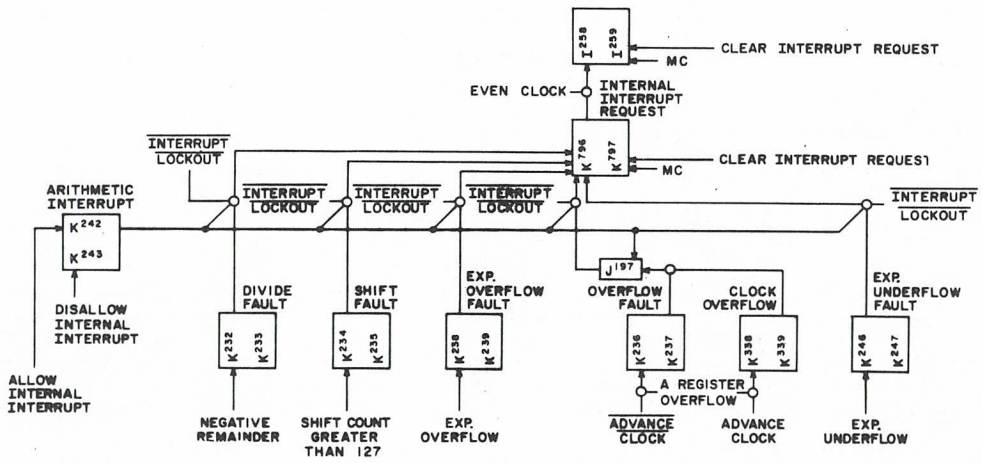


Figure 6-1. Fault FFs and Internal Interrupt Request

Table 6-2 contains the select and sense codes for internal (arithmetic) faults.

TABLE 6-2. INTERNAL FAULT CODES

CODE	DESCRIPTION
74 0 00100	Allow internal interrupts
74 0 00101	Disallow internal interrupts
74 7 001A0	Exit on internal (arithmetic) fault A
74 7 001A1	Exit on no internal (arithmetic) fault A
	A = 1 Divide Fault
	A = 2 Shift Fault
	A = 3 Overflow Fault
	A = 4 Exponent Overflow
	A = 5 Exponent Underflow
74 7 00300	Exit on Clock Overflow
74 7 00301	Exit on no Clock Overflow

External Interrupts

In external equipment, as within the 1604-A, only previously selected fault conditions may cause an interrupt. The EXF instructions used to select interrupts are listed in the 1604-A Peripheral Equipment Codes manual.

Each buffer channel has a separate interrupt line. The interrupt lines from external equipment come to the Masked Interrupt Register (MIR) in the 1604-A (figure 6-2). The MIR consists of eight flip-flops, one for each interrupt line. The inputs to the MIR can be masked (disallowed) by the Interrupt Masking Register (IMR). A FF in the IMR must be clear to enable an interrupt signal to enter the Masked Interrupt Register. For example, I344 (the clear output of I344/345) must be a "one" in order to allow an interrupt signal to set the channel one FF, K342/343, in the MIR. The FFs of the IMR are set or cleared by the following EXF instructions:

TABLE 6-3. EXTERNAL INTERRUPT CODES

CODE	DESCRIPTION
74 0 03X00	Allow interrupt-channel X
74 0 03X01	Disallow interrupt-channel X
X = 0	Channel 7 (Output)
X = 1	Channel 1
X = 2	Channel 2

TABLE 6-2. (Cont'd)

CODE	DESCRIPTION
X = 3	Channel 3
X = 4	Channel 4
X = 5	Channel 5
X = 6	Channel 6
X = 7	Channel 7 (Output)

In addition to the IMR which allows or disallows individual interrupts, an overall mask is available. When I364/365 is set, a clear input is held on all FFs of the Masked Interrupt Register via W180 and W181. I364/365 is controlled by the EXF codes.

74 0 04000	Allow selected External Interrupts
74 0 04001	Disallow External Interrupts

Interrupt Scanner

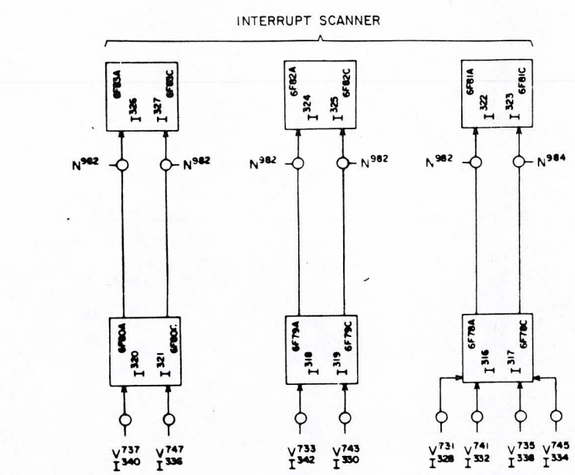
In order to give each external interrupt source equal priority and to prevent conflicts between simultaneous Interrupt signals, the external interrupts are handled by the interrupt scanner.

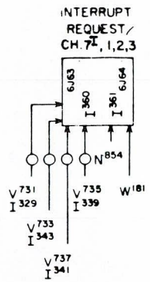
The interrupt scanner (figure 6-2) consists of a two rank, 3-bit counter driven by a rank of eight inverters. The inputs to rank I of the counter are enabled by the clear outputs of the MIR FFs. Rank I of the counter is transferred to rank II every even clock time. The inverters V7-- translate the count in rank II of the counter. When no interrupts are present the scanner counts in the order Channel 7^{I}-1-3-2-6 Channel 7^{0}-5-4.

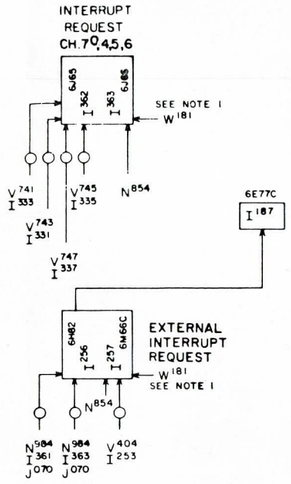
As an example of interrupt scanner operation, assume that:

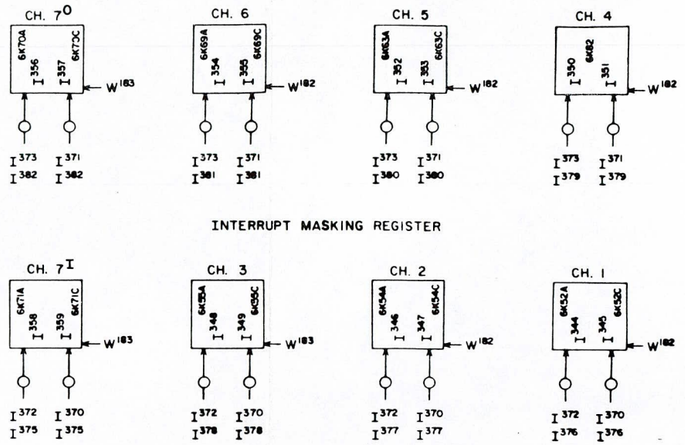
The scanner holds a count of 4
An interrupt signal is present on channel 3
Channel 3 interrupts have not been masked

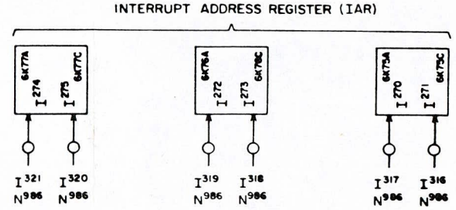
When the count in rank II reaches 000 a pulse from V731 enables the channel 3 interrupt to set I338/339 in the MIR. The scanner counts 0-1-3. When rank II of the counter holds a count of 3 a pulse from V735 probes the lowest bit of rank I of the counter. Since I338 is a "0" (I338/339 is set), the counter cannot change to 2. The pulse from V735 "ands" with I339 to set Interrupt Request I360/361. If no other interrupt request is being processed (interrupt lockout is not set), I361 sets the External Interrupt Request FF I256/257. At this time the counter is stopped at 3 and the Interrupt Address Register (IAR) holds the count of 3 indicating that the interrupt request came from channel 3.

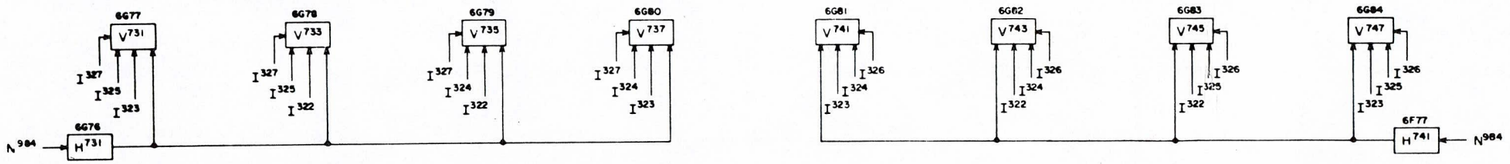












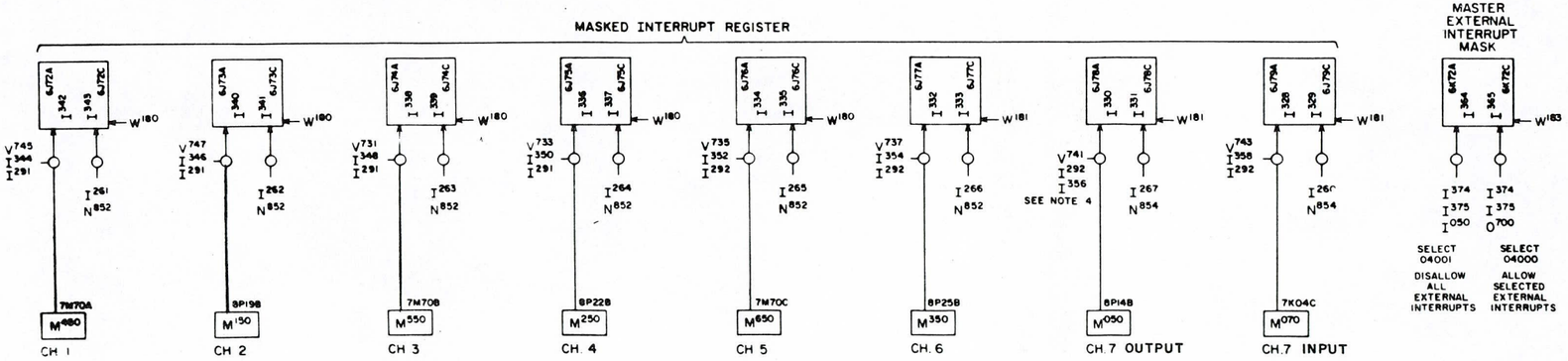


Figure 6-2. Masked Interrupt Register and Interrupt Scanner

INTERRUPT SEQUENCE ENTRANCE

When either an internal or external interrupt request occurs RNI is disabled and the Interrupt sequence is enabled. There are five different conditions under which the Interrupt sequence may be initiated:

1) An interrupt request is present when the computer exits from a sequence.
2) An internal interrupt occurs during the first part of RNI.
3) An interrupt occurs during an Output Transfer (63) instruction.
4) An interrupt occurs during an Input Transfer instruction.
5) An interrupt occurs during a Storage Search instruction.

Interrupt Entrance From Exit: As shown in figure 6-3, J061 becomes a "1" when either the Internal or External Interrupt Request FF is set. When any exit (Full, Half, or Jump) occurs, the Interrupt Sequence is initiated at H786.

Internal Interrupt During RNI: In some cases an internal fault occurring during an instruction may not be recognized before the next RNI is begun. Since it may be necessary to correct the fault before the next instruction can be executed properly, the RNI sequence can be stopped by an internal interrupt request just before another instruction sequence is initiated. The Interrupt Sequence is then entered at H789.

Interrupt During Output Transfer: If an interrupt request occurs during an Output Transfer instruction, special logic is required to ensure resumption of the transfer at the proper point after the Interrupt has been processed. This logic is associated with the S & T sequence. After the S & T sequence has taken the proper steps the Interrupt sequence is entered at H789.

Interrupt During Input Transfer: If an interrupt request occurs during an input transfer, the transfer stops between words and no special logic is required to ensure that the transfer is resumed at the proper place. A "one" from the Input Transfer Active FF is used to enter the interrupt sequence at H789.

Interrupt During Storage Search: When an interrupt request occurs during a storage search instruction the search is halted between words and the interrupt sequence is initiated at H789. The search is resumed by re-reading the instruction after the interrupt request has been processed.

INTERRUPT SEQUENCE

The Interrupt Sequence performs the preliminary steps in going to the interrupt subroutine when an interrupt request is recognized. The steps are:

1)	Clear Input Transfer Active FF	
2)	Clear X^1	Prepare to transmit P → X
3)	Set R^2 → B FF	If Interrupt occurred during Output Transfer
4)	Clear K^{830}_{831}	Disable Input Transfer Resume
5)	Set Internal (I^{252}_{253}) or External (I^{254}_{255}) Interrupt FF	Determined by source of interrupt
6)	$P^1 \rightarrow X^2_L$ $X^2_L \rightarrow X^1_L$	Prepare to store P at 00007-00017
7)	Clear Control FFs	Stop present operations and prepare to start a new sequence.
8)	Initiate Write Operand Sequence	

The Write Operand (WO) sequence is used to store the main program address at which the interruption occurred. This address is stored at the upper address portion of the interrupt location (00007-00017) corresponding to the source of the interrupt. The commands issued are:

1)	Set P^1	If the interrupt is external I254/255 is set, the content of the Interrupt Address Register is sent to P and the fourth bit of P is set. If the interrupt is internal, P is set to 7.
2)	Set Interrupt Lockout FF K064/065	Prevents recognition of further interrupts.
3)	Clear External Interrupt (MIR)	A translation from the IAR is used to select the FF in the MIR which is cleared. Only the interrupt being processed is cleared. This action releases the scanner to search for other external interrupts.
4)	Clear External Interrupt Request FF	If the interrupt being processed is internal, clear the external interrupt FF.
5)	Store condition of Exit FF in Interrupt Exit FF	To provide for returning to proper half of instruction in main program.
6)	Initiate Storage	To obtain interrupt word (00007-00017) from storage
7)	Set WO Wait Storage	

8)	Set Return Jump FF K^{460}_{461}	
9)	Enable Partial Write Upper $X^1_L \rightarrow X^2_U$ $X^2_U \rightarrow X^1_U$	$P \rightarrow X^1_U$ to store address of main program step in upper address of interrupt address
10)	Clear X^1_L	
11)	Clear U^1_L $I^5 I^6 \rightarrow U_1$	Instruction from interrupt word placed in U^1_L
12)	Half Exit	

The half exit advances the instruction in U^1_L (which came from the lower half of the interrupt word) to U^1_U for execution. This instruction is normally a jump to an interrupt subroutine.

INTERRUPT SUBROUTINE

An interrupt subroutine is a routine of instructions which executes a series of Sense (74.7) instructions to detect the source of an interrupt and then executes instructions to correct the fault. Since there are nine interrupt locations (table 6-1), there may be nine separate interrupt subroutines for one main program. Table 6-4 contains an example of an interrupt subroutine.

TABLE 6-4. TYPICAL INTERRUPT SUBROUTINE

00011	75 0	75 0 01000	
	Address of next instruction in main program is stored here		
01000	Exit on no Carriage Return or Tab	Jump to Carriage Return or Tab	Detect Specific Fault
01001	Exit on End of Tape (Message)	Jump to End of Tape (Message)	
01002	Exit on _____	Jump to _____	
Carriage Return or Tab	_____	_____	Process Carriage Return or Tab Interrupt
	_____	_____ Select typewriter with no interrupt on Carriage Return	
	_____ Jump to 00011		

TABLE 6-4. (Cont'd)

End of Tape (Message)			
			Process End of Tape (Message) Interrupt
		Select Paper Tape Reader with no interrupt on End of Tape (Message)	
	Jump to 00011		

Once the interrupt subroutine has corrected the fault which caused the interrupt, a jump is made to the interrupt address.

RETURN TO MAIN PROGRAM

When RNI sequence obtains from storage the upper instruction at an interrupt address (00007-00017) it sets the Interrupt Complete FF K072/073. The upper instruction at an interrupt address is a jump instruction; the upper address is the address of the main program step which was interrupted. The jump instruction sets P to the main program address and executes a jump exit. RNI then determines whether the upper or lower instruction should be executed by issuing the following commands (figure 2-11):

1)	Initiate Storage	To obtain main program word from storage
2)	Set Wait Storage FF	
3)	Set Exit FF	
4)	Clear U^1 $I^5\ I^6 \rightarrow U^1$	Main program word to U^1
5)	Clear Interrupt Lockout FF K064/065	
6)	Clear Interrupt Complete FF K072/073	

If the Interrupt Exit FF K068/069 is set, RNI continues in the normal manner and the upper instruction is executed. If K068/069 is clear RNI is inhibited by J113 at the input to H098 and the following action is initiated:

1)	V097 clears Wait Storage FF K200/201	
2)	RNI is restarted at H094	
3)	Clear Exit FF K060/061	
4)	$U^1{}_L \rightarrow U^1{}_U$	Lower instruction transferred to $U^1{}_U$ for execution

The Interrupt Complete FF slave K062/063 is now clear, J113 equals "1", and RNI continues in the normal manner.

The main program has now been resumed at the point at which it was interrupted and continues normally.

REAL TIME CLOCK

Storage address 00000 is used as a 48-bit register to provide an indication of elapsed real time. When the real time clock is selected, a binary one is added to the content of address 00000 every 1/60 of a second.

The clock (figure 6-4) is controlled by the EXF codes:

74 0 01000	Start Real Time Clock
74 0 02000	Stop Real Time Clock

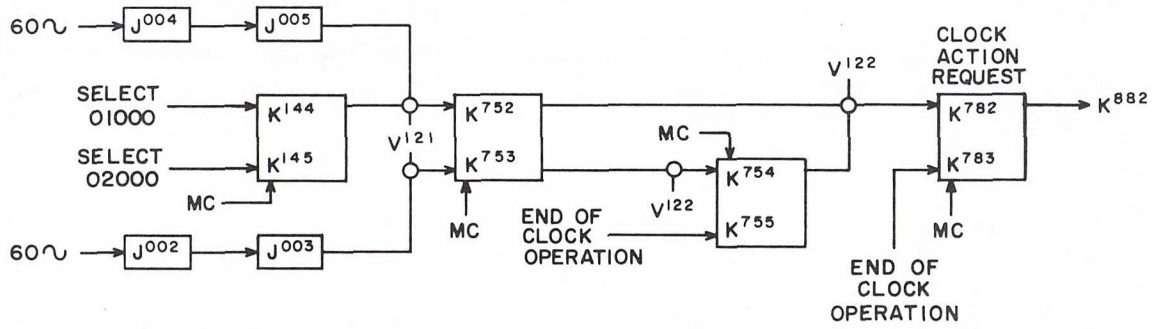


Figure 6-4. Real Time Clock

When the clock is running (K144/145 is set), the Clock Action Request FF is set every 16 2/3 usec.

A clock action request inhibits RNI and initiates the Interrupt sequence in the same manner as an Interrupt request. When K782/783 is set J061 (figure 6-3) becomes a "1" and K694/695 is set. A clock request temporarily stops the execution of the main program under the same conditions as an Interrupt request (except for internal interrupt during RNI. However, once the Interrupt sequence is started the procedure is different.

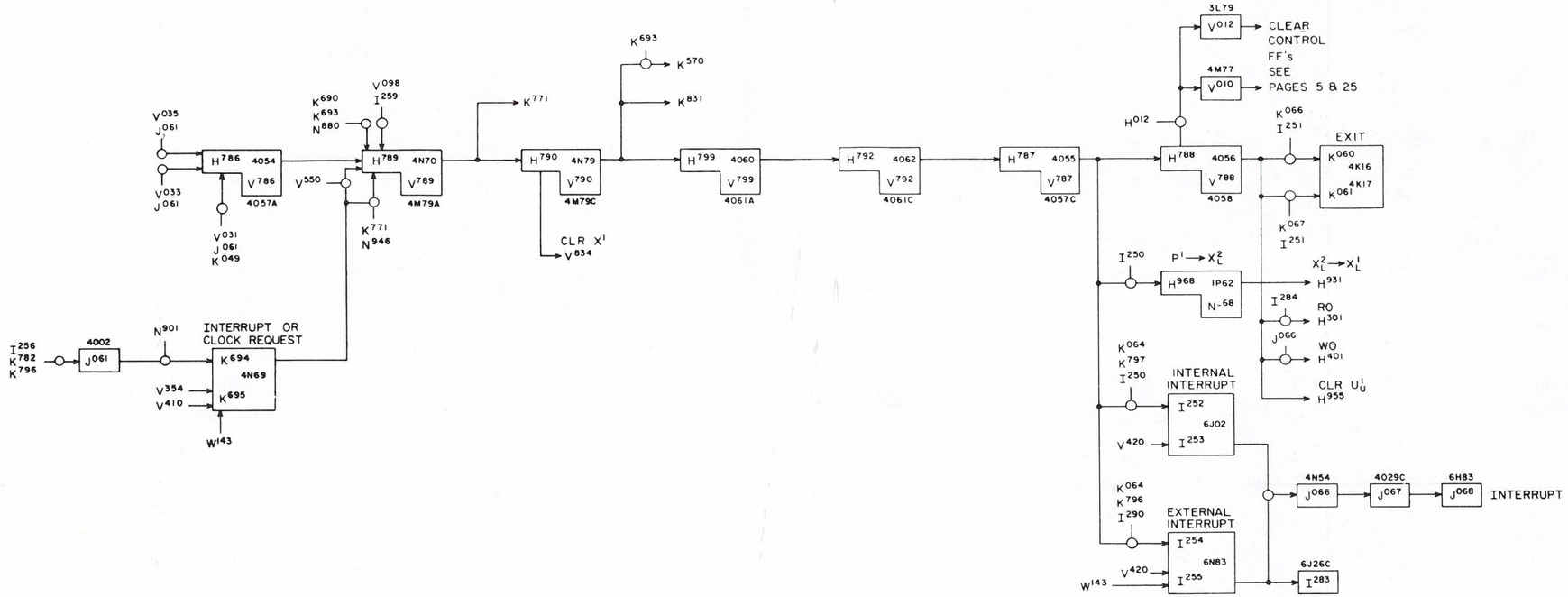


Figure 6-3. Interrupt Sequence

In addition to initiating the Interrupt sequence as discussed under Interrupt, the S & T sequence sets the Advance Clock FF, I250/251, as shown in figure 6-5. The outputs of I250/251 are used as gating conditions. Commands issued by the Interrupt sequence are:

1)	Clear X^1	Prepares for operations of the RO sequence
2)	Clear U^1	
3)	Complement Exit FF, K060/061	Prepare for return to main program
4)	Initiate Read Operand (RO) Sequence	

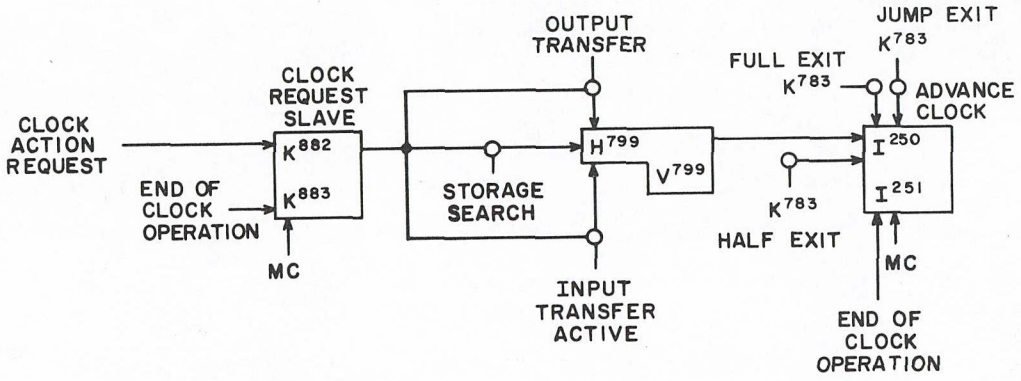


Figure 6-5. Setting Advance Clock FF

The Read Operand sequence is now used to add one to the word in address 00000. The sequence of commands is listed in Part 2 of the Customer Engineer Instruction Manual. The exit taken from RO sequence is either jump exit or half exit (P is not advanced) depending upon the setting of the Exit FF. Since P is not advanced when exiting from the RO sequence, RNI reads the instruction word which was halted by the clock request. The half or jump exit taken from RO sequence provides for executing the proper half of the instruction word. The program then continues normally.

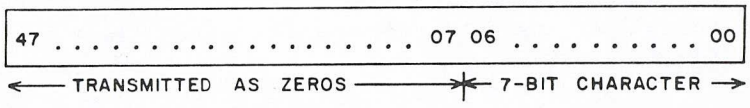
CHAPTER 7
CONSOLE INPUT/OUTPUT EQUIPMENT

Three units of input/output equipment, mounted on the console, are an integral part of the 1604-A computer. A Teletype high speed tape punch and a high speed tape reader provide for processing perforated paper tape. An IBM typewriter, modified by Soroban Engineering, Inc., provides for direct keyboard entry of data and for printed copy output. The console input/output units communicate with the computer via buffer channel 1 (input) and 2 (output). Other input/output units may share these channels but the console input/output units cannot use any of the other channels.

The control and data-handling circuits for the console-mounted units are contained in the main computer cabinet on chassis 7 and 8. For information concerning the operation of the units refer to the diagrams in Volume C in this manual and to the manual for each of the units in Book 3, Volume D, publication number CDC 304.

MODES FOR HANDLING DATA

Data may be transmitted between the console equipment and the computer in either the Character or the Assembly mode. In the Character mode either a 7- or an 8-bit character (depending upon the program) is buffered to or from the computer one bit at a time. If a 7-bit character is used, it occupies the lowest bit positions of a 48-bit word; "0's" are transmitted in the upper 41 bits.



In the Assembly mode the 48-bit word consists of eight 6-bit characters.

CHAR. 7	CHAR. 6	CHAR. 5	CHAR. 4	CHAR. 3	CHAR. 2	CHAR. 1	CHAR. 0
47...42	41...36	35...30	29...24	23...18	17...12	11...06	05...00

During an input buffer in the Assembly mode (with the reader, for example) eight successive characters are assembled into a 48-bit word, beginning with character 7. After the word is assembled it is sent to the computer. For an output buffer in the Assembly mode (with the punch, for example) a 48-bit word from the computer is disassembled into eight characters. These are punched successively beginning with character 7.

EXTERNAL FUNCTIONS

Program control of the console Input/Output equipment is accomplished by instruction 74 (External Function). The execution address of a 74.0 or 74.7 instruction is an EXF code. Console equipments are specified by codes:

112XX	reader
212XX	punch
111XX	typewriter input, referred to as keyboard
211XX	typewriter output, referred to as type

A complete EXF code consists of 15 bits; however, only the lower 12 bits are sent out to the equipment. The upper 3 bits, translated within the computer, select the channel to which the remainder of the code is sent.

SELECT CODES

The console equipments and their operating modes are selected by the 74.0 instruction with one of the Select codes listed in table 7-1. The Select codes are translated and stored by the external function translator (figure 7-1). The combination of Function Ready and a Select code sets one or more FFs on the translator. Four of the FFs store equipment selection:

Reader K910/911	Keyboard K902/903
Punch K912/913	Type K900/901

The Select code that sets one of these FFs also clears the FF for the other equipment on that channel. Selecting the typewriter (output) clears the Select Punch FF; selecting the reader clears the Select Keyboard FF. The Select Reader FF is also set by an External Master Clear signal. Thus, the reader is automatically selected by an external clear; if a manual load operation follows the Clear signal, the steps necessary to select the reader are eliminated.

The Select codes may also set or clear other FFs of the translator, depending upon the operating conditions specified for the selected equipment. These FFs and the codes which set them are:

Interrupt On CR	K988/989	1114X
Reader End-Of-Tape	K904/905	1121X
Interrupt On End-Of-Tape	K906/907	1122X
Assembly Mode	K914/915	2XX0X
Punch Motor	K916/917	2120X, 2121X, 2122X, 2123X

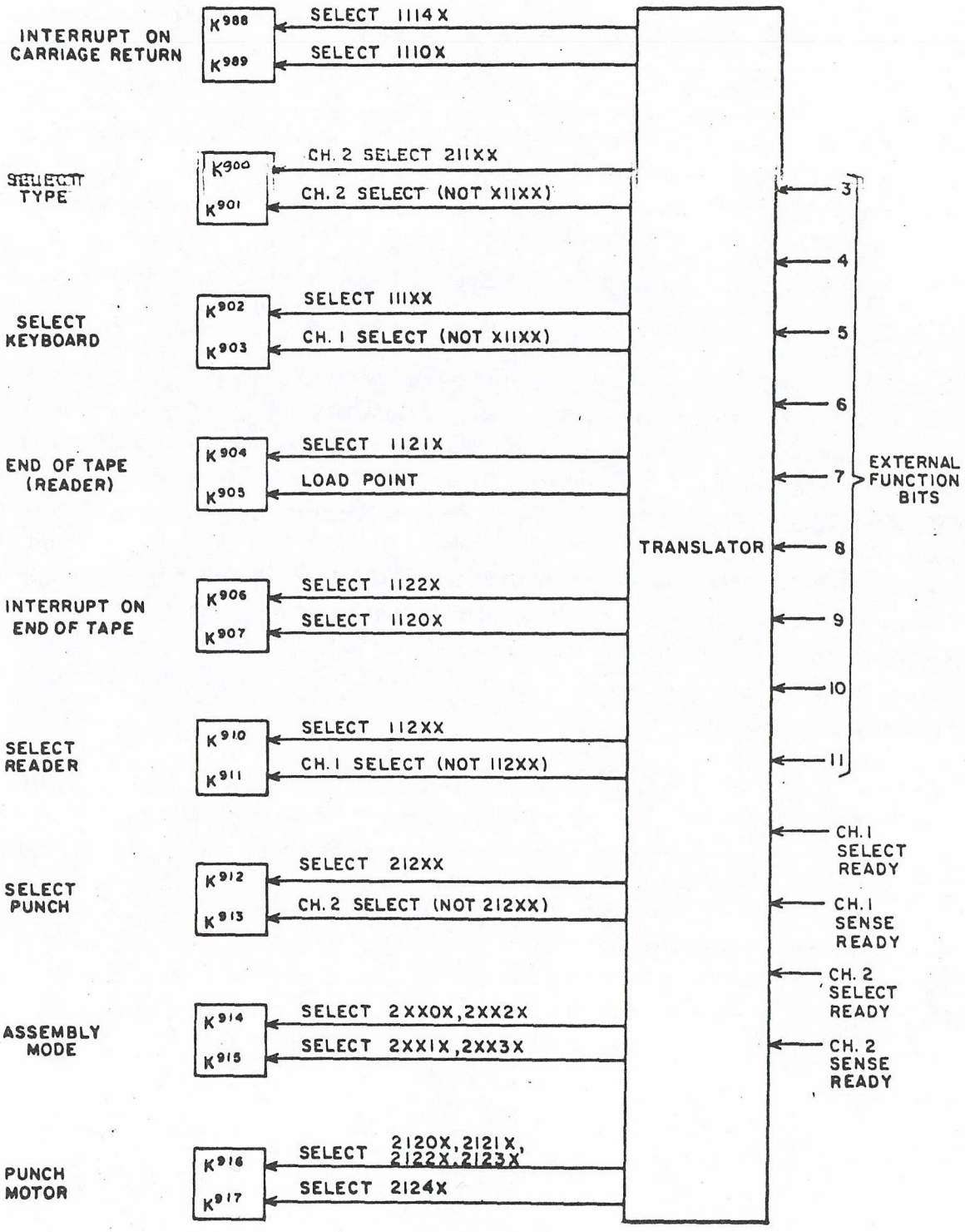


Figure 7-1. External Function Translator

TABLE 7-1. SELECT CODES FOR CONSOLE EQUIPMENT

CHANNEL 1		CHANNEL 2	
11100	Keyboard Entry And No Interrupt On CR	21100	Type Assembly Mode
11140	Keyboard Entry And Interrupt On CR	21110	Type Character Mode
11200	PT Reader And Interrupt On End-Of-Tape	21200	Punch Assembly Mode
11210	PT Reader And Set End-Of-Tape Indicator	21210	Punch Character Mode
11220	PT Reader And No Interrupt On End-Of-Tape	21240	Turn Punch Motor Off

SENSE CODES

Five different conditions within the console equipments are sensed by the 74.7 instruction with the Sense codes listed in table 7-2. Translation of the code is combined with a signal indicating the specified condition (figure 7-2). The output is the Sense Return signal sent to the computer. The Sense Return from J968 is a "1" when the specified condition exists.

The Sense circuit of the console equipment ignores the lowest bit of the Sense code; the Sense Return signal sent to the computer is therefore the same for either of the codes in the pair associated with a condition (table 7-2). The computer Sense circuit combines the ignored bit with the signal from J968, thereby distinguishing between the two codes in the pair.

TABLE 7-2. SENSE CODES FOR CONSOLE EQUIPMENT

CHANNEL 1		CHANNEL 2	
11100	Exit On Keyboard CR Or Tab	21200	Exit On Punch Out-Of-Tape
11101	Exit On No Keyboard CR Or Tab	21201	Exit On No Punch Out-Of-Tape
11140	Exit On Keyboard Lower Case		
11141	Exit On Keyboard Upper Case		
11200	Exit On Reader End-Of-Tape		
11201	Exit On No Reader End-Of-Tape		
11210	Exit On Reader Assembly Mode		
11211	Exit On Reader Character Mode		

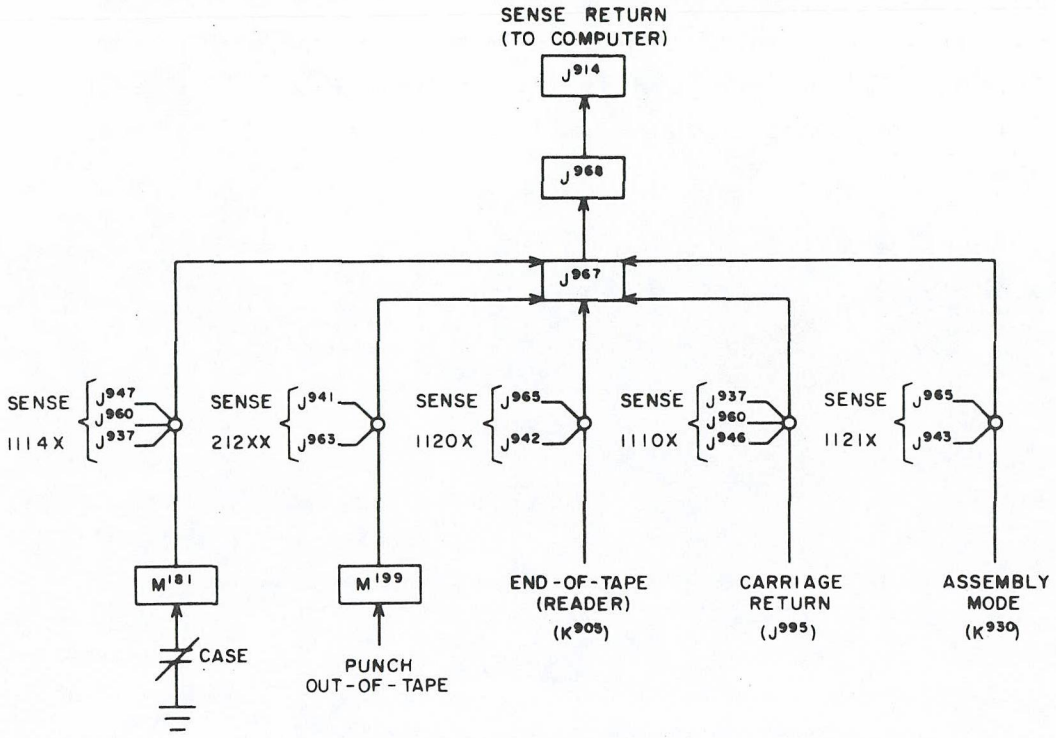


Figure 7-2. Sense Circuit Console Equipment

INPUT DISTRIBUTOR

The input distributor (figure 7-3), which consists of the Input Assembly register (AR) and the assembly counter, receives characters from the typewriter keyboard and the paper tape reader and forms 48-bit words from these characters for the computer. The 48-bit word is formed in AR under the control of the assembly counter, which determines the position of AR into which each character is entered.

For reader input operations in the Assembly mode, figure 7-4b shows how the counter gates the characters from paper tape into AR. After each character is gated into AR the counter is advanced.

During an input operation from the reader in the Character mode, the assembly counter is forced to the count that produces the Gate Character 0 signal. Each 7-bit or 8-bit character from the reader is entered into the lowest seven or eight stages of AR. The remaining stages hold "0". This 48-bit word is sent to the computer.

Input from the typewriter keyboard always takes place in the Character mode. Six-bit characters from the keyboard go directly to the lowest stages of AR independently of the assembly counter. After a keyboard character enters AR the entire 48-bit word (consisting of "0's" in the upper 42 stages and the 6-bit character in the lower stages) is transmitted to the computer.

AR is transmitted to the X register of the computer via the I^0 inverters. When neither the reader nor the keyboard is selected, AR is held clear.

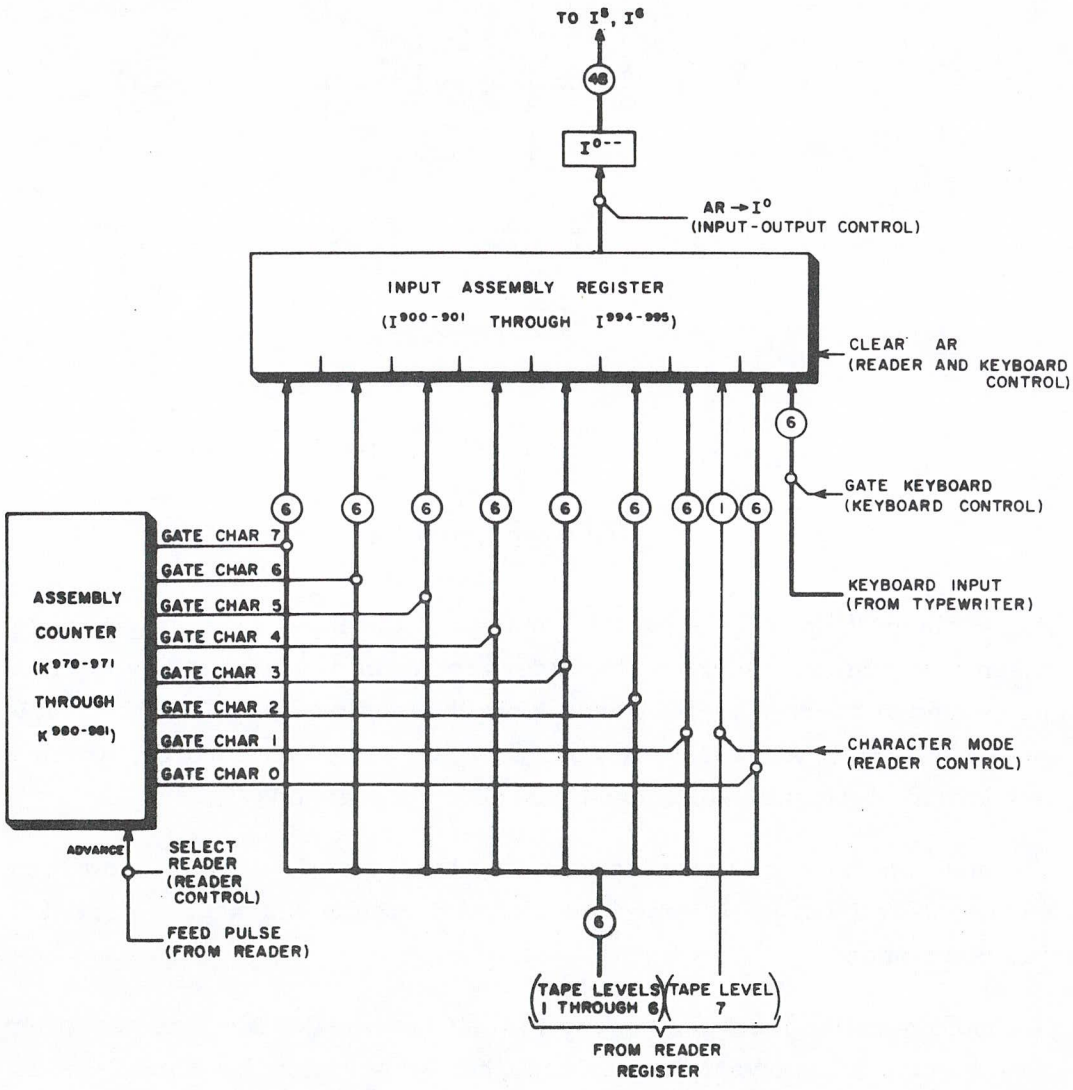


Figure 7-3. Input Distributor, Simplified Diagram

OUTPUT DISTRIBUTOR

The output distributor (figure 7-5), which consists of Output register O^1 and the disassembly counter, receives 48-bit words from the computer and from them extracts characters which are sent to the punch or typewriter.

During output operations in the Assembly mode, the distributor disassembles each 48-bit word held in O^1 into eight 6-bit characters. These characters are sent successively to the punch or typewriter. Figure 7-4a illustrates the order of disassembly, which is controlled by the counter. After each character is extracted, the counter is advanced to prepare for extracting the next one.

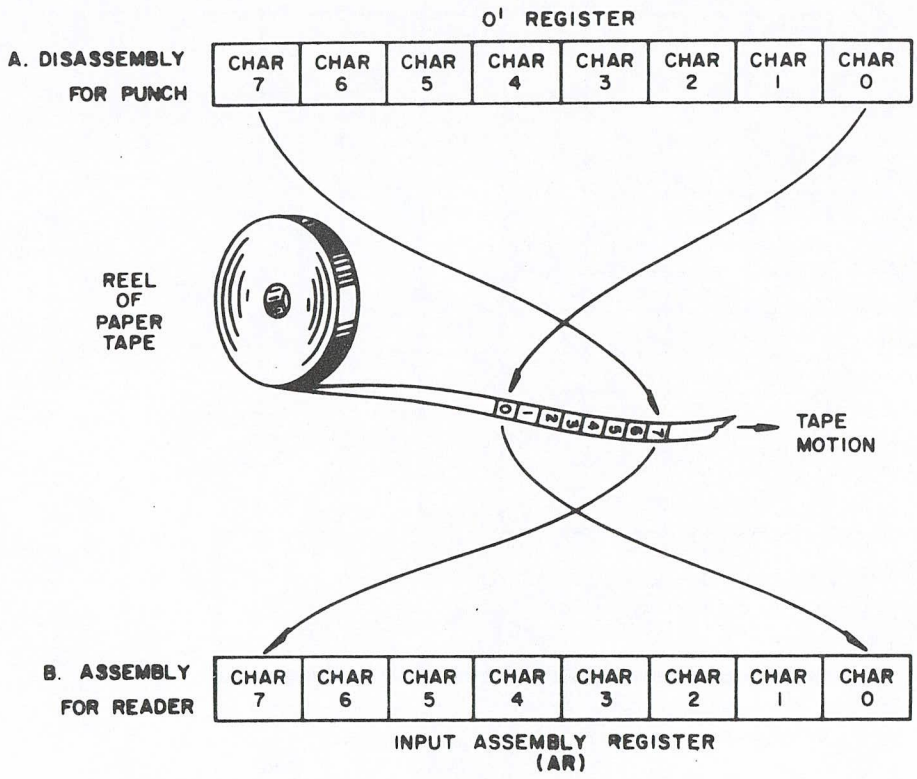


Figure 7-4. Character Orientation in Registers and on Paper Tape

During output operations in the Character mode the counter is forced to the count which produces the Gate Character 0 signal. Thus only the character in the lowest stages of O^1 is sent to the punch or typewriter. The upper bits of O^1 are ignored.

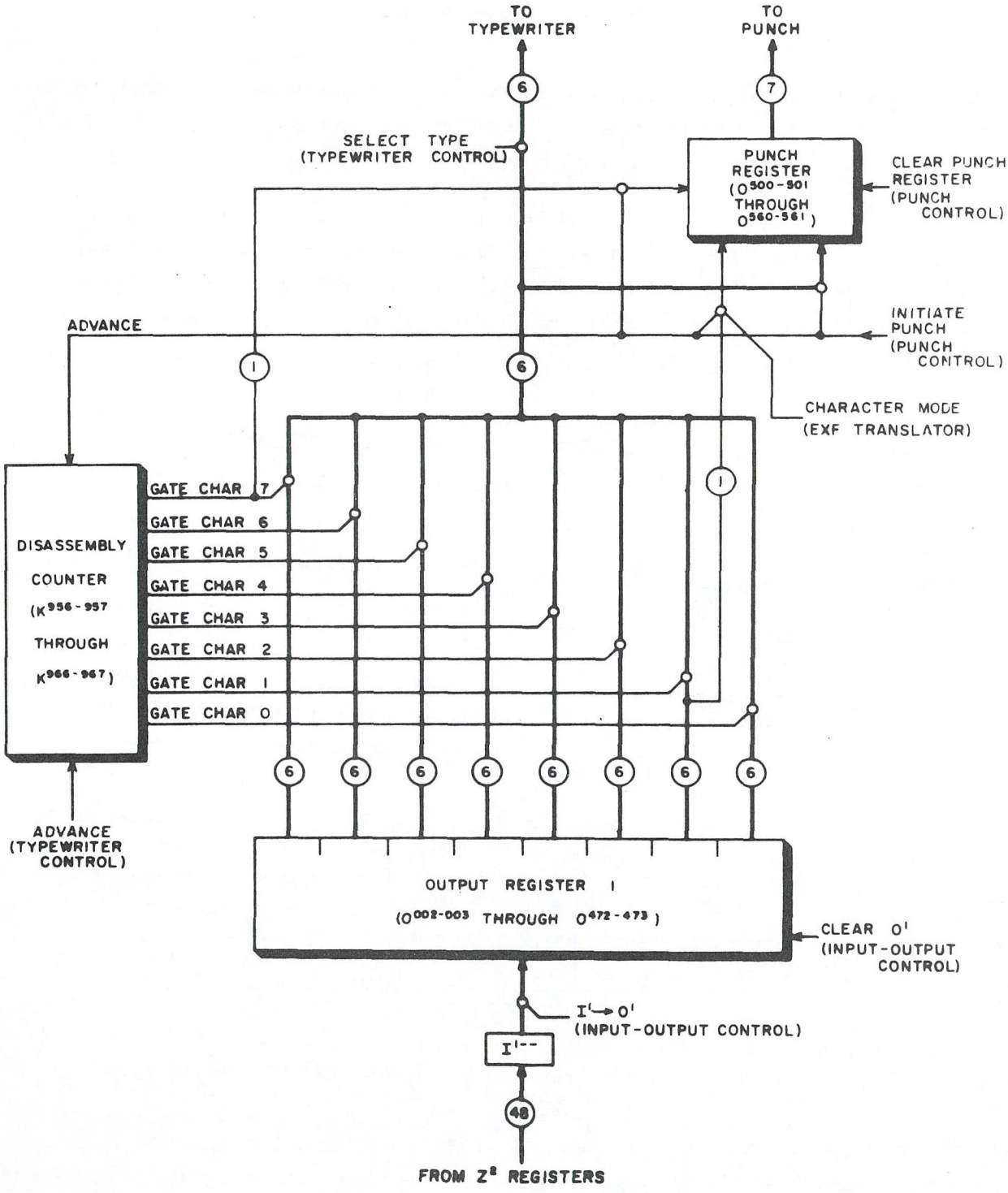


Figure 7-5. Output Distributor, Simplified Diagram

PAPER TAPE READER

The tape reader which enters information stored on punched paper tape into the computer is always connected to channel 1. It operates at a maximum rate of 350 characters (lines) per second; the time interval between successive characters from the reader is 3.3 ms. The complete tape reader is composed of a tape feed mechanism, an optical projection system, the reading and location photocells, the power supply, and the brake and clutch drive circuits.

MANUAL CONTROLS

Manual controls for the reader are two lever switches on the punch and reader control panel of the console. The Reader Motor switch provides for manually turning on and off the reader motor, and must be set to the Up position whenever the reader is used. This sets the Reader Motor FF K918/919; the "1" output energizes the reader motor.

The Reader Mode switch selects either the Character or Assembly mode for operation for a reader input operation: Assembly (up) positions the tape at the first frame of the first word (load point); Character (down) moves the tape ahead one frame.

PAPER TAPE

Information is stored on paper tape with seven or eight levels accross the tape width. Since seven level tape is most often used, the following descriptions of the paper tape reader and paper tape punch assumes that seven level tape is being used. However, it should be noted that both the reader and punch are equipped to handle eight level tape. It is necessary, however, to adjust the paper tape guide on the paper tape reader when changing between 7-level and 8-level paper tape.

The levels are 7, 6, 5, 4, 3, 2, 1 from top to bottom (figure 7-6). The sprocket or feed holes between levels 3 and 4 generate timing and control signals. In the Assembly mode, level 7 is a control rather than an information level. A hole in the control level indicates the first of the eight characters in a word.

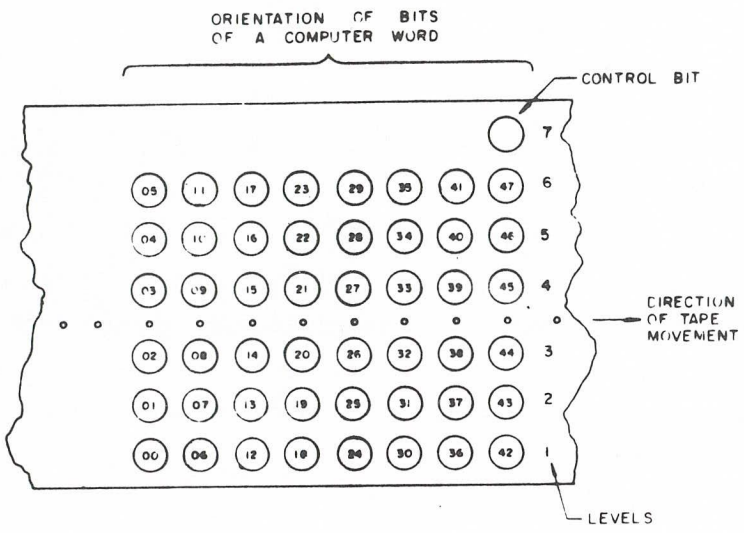


Figure 7-6. Seven Level Punched Paper Tape

READER OPERATION

In the Assembly mode, each character is transmitted to its proper position in AR by the feed pulse (generated by the feed hold in the frame) under the control of the assembly counter. After each transmission of a character, the counter is advanced one count. When a 48-bit word is assembled in AR, a Ready signal is sent to the computer. See figure 7-7 for Reader Control circuit and table 7-3 for sequence of actions.

In the Character mode (table 7-3) each consecutive character from the reader is entered into the lowest order seven or eight stages of AR. After a character is loaded into AR the Ready signal is sent to the computer. The computer samples AR after each character has been read.

The tape motion stops on either of two conditions: (1) if the input buffer active line drops, signaling the end of the buffer operation; (2) if, in the Assembly mode, a hole in level 7 does not accompany the first character of a word on the tape (this condition is interpreted as end-of-tape).

In the End-of-tape circuit (figure 7-8) K904/905 is cleared when a Mode selection is made; this FF is set to indicate the end of a tape on either of two conditions:

1) By External Function (EXF) code 1121X. With this code, the End-Of-Tape FF can be set at the end of a character mode operation. This is necessary when program steps are conditioned by the result of a 74.7 instruction that senses end-of-tape.
2) In the Assembly mode of operation, by the absence of a Level 7 signal (from I806) when the assembly counter is at count 000.

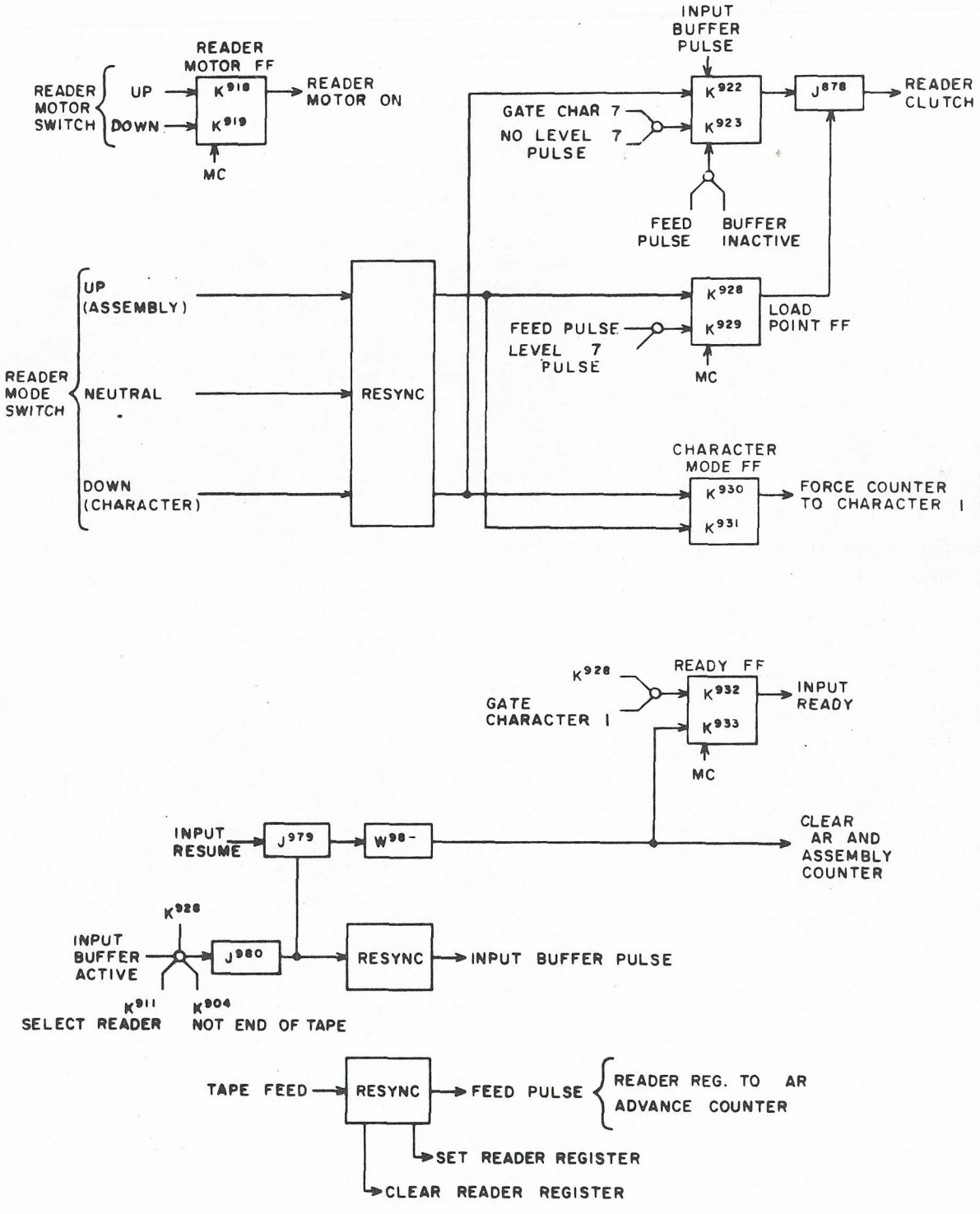


Figure 7-7. Reader Control Circuit

TABLE 7-3. READER OPERATION

ASSEMBLY MODE	1) Select Assembly mode (switch up)	Set Load Point FF (energize reader clutch). Clear Character Mode FF. Tape moves through leader until it reaches first hole in level 7 (indicates load point); this is first frame with information. Clear Load Point FF (de-energizes clutch).
	2) Buffer Active signal	Energize clutch. Remove clear on AR, assembly counter, and Ready FF. Gate first character into AR.
	3) Feed pulse	Transmit character from reader photocells to Reader register. Transmit Reader register to AR. Advance assembly counter. Wait for next feed pulse; repeat 3. On 8th feed pulse (counter at character0) set Reader Ready FF to send Ready to computer.
	4) Computer samples AR and sends Resume	Clear AR. Clear Reader Ready FF. Wait for next feed pulse; repeat 3.
	If no level 7 hole for character	Clear Clutch FF.
	If buffer terminates (made inactive)	Clear Clutch FF.

CHARACTER MODE	1) Select Character mode (switch down)	Set Character Mode FF. Force assembly counter to character 0 Set Clutch FF (energizes reader clutch). Move tape one frame. Clear Clutch FF since buffer inactive.
	2) Buffer Active signal occurs	Energize clutch. Remove clear on AR, assembly counter, and Ready FF. Gate first character into AR.
	3) Feed pulse occurs	Transmit character to Reader register. Transmit Reader register to AR. Set Ready FF to send Ready to computer.
	4) Computer samples AR, responds with Resume	Clear AR. Clear Ready FF. Wait for next feed pulse; repeat 3
	If buffer terminates	Clear Clutch FF.

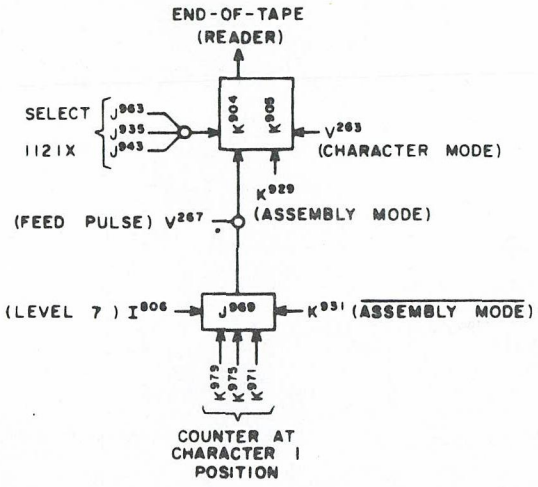


Figure 7-8. End-of-Tape Circuit

AUTO LOAD CONTROL

Pressing the Auto Load switch on the punch and reader control panel loads a bootstrap routine into memory locations 00020 and 00021.

The program appears as:

(00020)	74 1 00022	Activate Reader
	74 7 11200	Wait for end of tape
(00021)	74 1 00000	Deactivate buffer
	50 0 00000	Pass to next instruction
(00022)	XX X XXXXX	Will be the first
	XX X XXXXX	word read from tape

If Breakpoint switch is not set to 00020 or 00021, the routine is executed. The first word read from paper tape is read into location 00022 and executed as soon as the tape is read.

Pressing the Auto Load switch activates a rotary stepping switch on the Auto Load chassis located under the right wing of the 1604-A console. As the switch rotates it makes contact with terminals, permitting sequential performance of the steps necessary to load the bootstrap program. The complete Auto Load program as well as the stepping switch terminal board are shown on page 26 of the Customer Engineering Diagrams manual.

PAPER TAPE PUNCH

The Teletype BRPE punch which prepares paper tape output is always connected to buffer channel 2. Its nominal operating rate is 110 characters per second. When the Character mode is selected, 8-bit characters are punched; 6-bit characters are punched when the Assembly mode is selected.

PUNCH CONTROLS

Punch manual controls are two lever switches on the punch and reader control panel of the console. The punch motor may be manually turned on by placing the Punch Motor switch in the Up position, and turned off, in the Down position. EXF codes for punch motor control are:

2120X	Select Punch, Assembly mode	2124X	Turn Punch Motor Off
2121X	Select Punch, Character mode		

The Punch Mode switch must be in the Up position to operate the punch with the computer. The Down position provides for tape feed; that is, for punching out leader.

On the punch itself, the feedout lever also provides for punching out leader. A micro-switch is mounted near the roll of paper tape. When the supply is low, the switch closes and provides an out-of-tape indication which may be sensed.

PUNCH OPERATION

When the punch motor is on, the punch mechanism rotates. However, tape does not advance unless the punch feed magnets are energized. For each revolution of the punch mechanism, a timing signal is sent to the Punch Control circuit (pages 69 and 70, Customer Engineering Diagrams). This signal is resynchronized to produce a Timing signal that initiates character punching.

When the Punch Mode key is down, Feed I and Feed II FFs remain set; the output of the Feed II FF energizes the feed magnet, tape is advanced, and feed holes are punched out.

With the punch motor on and the Punch Mode switch Up, punching can begin when the computer: (1) selects the punch, (2) sends Output Buffer Active signal, and (3) loads O^1 and sends accompanying Ready signal. The steps in punch operation are:

1) Set Ready FF during absence of Punch Timing signal.
2) Punch Timing signal occurs to initiate remaining steps.
3) Clear punch register.

4) Transmit character determined by counter from O^1 to Punch register to energize punch magnets and feed magnet.
5) Clear Ready FF.
6) Set Resume FF if counter at character 0 (sends Resume signal to computer).
7) Advance disassembly counter.
8) Return to step 1 if counter is not at character 0 before step 7.
9) Computer responds to Resume (step 6) by turning off Ready.
10) Clear Resume FF.
11) Computer loads O^1 and sends Ready.
12) Return to step 1.

The disassembly counter controls the distribution of characters from O^1 to the Punch register. In the Assembly mode, the counter disassembles each word into characters in the order shown in figure 7-4; the highest order character is recorded first, the lowest order character last. A Resume signal is generated by K968/969 each time the lowest order character is entered into the Punch register; this signal notifies the computer that another word may be entered into O^1.

In the Character mode, the disassembly counter selects only the lowest order characters of each word in O^1 for recording, ignoring all the other bits. The Resume signal is generated after each character has been punched; a new word is received from the computer each punch cycle. Even when 7-level tape is used, the 8th level punch pin is driven when there is a "1" in the 8th bit (bit 7).

TYPEWRITER

The typewriter is an IBM Electric, modified by Soroban Engineering, Inc. by the addition of a mechanical coder and decoder. The typewriter may be used as either a keyboard input device or as an output device for producing printed copy. For output the typewriter can operate at a rate of approximately 10 characters per second.

TYPEWRITER CODES

All of the typewriter characters and functions are represented by unique combinations of 6 bits. The complete set of codes is given in table 7-4, where the characters and functions are listed with their Octal codes.

During a keyboard input operation striking a character key causes the coder to produce, as an output, the code for that key. Space is the only typewriter function for which a code is formed. For other functions (tab, backspace, upper case, etc.) no code is formed during input operations. The coded character is then sent to the computer.

For a typewriter output operation, a 6-bit character consisting of one of the codes in table 7-4 is sent to the decoder; the decoder actuates the key for printing a character or closing one of the six function contacts.

TABLE 7-4. TYPEWRITER CODES

a. Characters UC	LC	Code	a. Characters UC	LC	Code	a. Characters UC	LC	Code
A	a	30	P	p	15	#	3	64
B	b	23	Q	q	35	$	4	62
C	c	16	R	r	12	%	5	66
D	d	22	S	s	24	¢	6	72
E	e	20	T	t	01	&	7	60
F	f	26	U	u	34	½	8	33
G	g	13	V	v	17	(	9	37
H	h	05	W	w	31	_	-	52
I	i	14	X	x	27	?	/	44
J	j	32	Y	y	25	"	'	54
K	k	36	Z	z	21	°	+	46
L	l	11				.	.	42
M	m	07	)	0	56	:	;	50
N	n	06	*	1	74	,	,	40
O	o	03	@	2	70	÷	=	02

b. Functions	Code	b. Functions	Code
Tab	51	Upper Case	47
Space	04	Carriage Return	45
Back Space	61	Lower Case	57

KEYBOARD OPERATION

The Keyboard Control circuit (figure 7-9) provides for loading characters from the contacts of the typewriter coder into AR. Selecting the keyboard allows the circuit to enable sampling of the keyboard common contacts and remove the Clear signal to AR.

Each striking of a key actuates the coder which closes some combination of the six contacts TC1-TC6. These contacts provide Output signals that form the 6-bit code for the character associated with the key. At the same time that one of the contacts TC1-TC6 is actuated, the common contacts of TCC are actuated. This signals to the control circuit that a character from TC1-TC6 can be gated to AR. The steps involved for one character are:

1) Striking character key.
2) Key actuates coder closing contacts TC1-TC6 to generate code.
3) Key actuates common contacts of TCC.
4) Resynchronized signal from TCC contacts:
   Gates character from TC1-TC6 to AR.
   Sets Ready FF (sends Ready to computer)
   Clears K984/985
5) TCC contacts return to normal state later in typewriter cycle.
6) Computer responds to Ready with a Resume which:
   Clears AR
   Clears Ready FF
7) Return to step 1.

The levers for typewriter functions do not actuate the coder. The space contacts do, however, cause the code 04 to be entered into AR.

If the keyboard is selected by code 1114X, the Interrupt signal is generated each time the carriage return (CR) or tab keys are struck. This feature allows the operator to enter information via the keyboard.

After selecting the keyboard with interrupt on CR, (figure 7-10) the state of the circuit is: K988/989 set, K994/995, K986/987 and K996/997 clear.

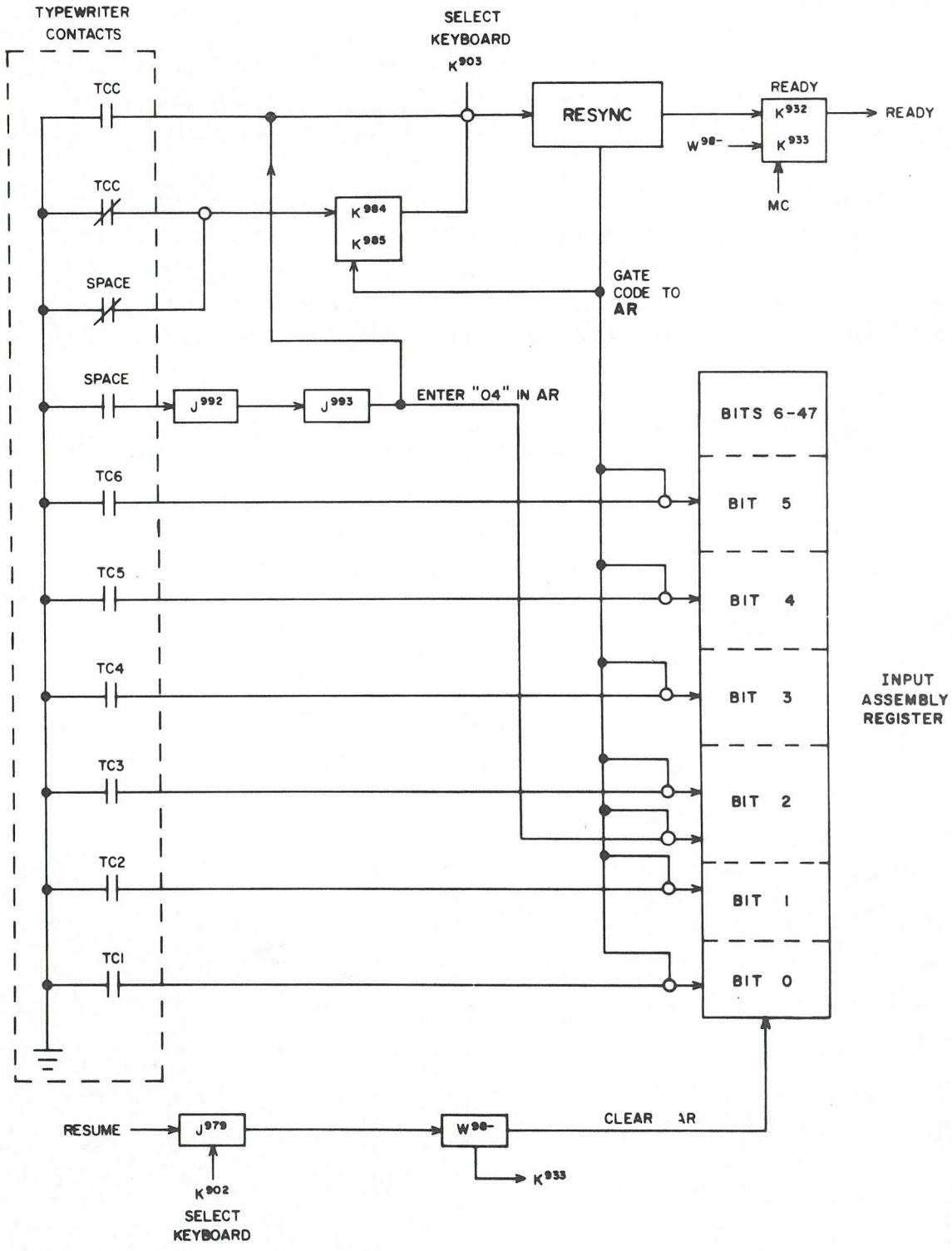


Figure 7-9. Keyboard Control Circuit

The signal from CR or tab is resynchronized and stored by K996/997 to produce the Interrupt signal. The sequence that results from a CR or tab is:

1) Striking CR or tab
   sets K994/995 which
   sets K996/997 which
   sets K986/987 to prevent contacts from being sampled again during typewriter cycle.

2) Computer interrupt program
   determines by Sense code 1110X that typewriter is source of interrupt because
   K996/997 is set
   selects 1110X which turns off interrupt and
   removes interrupt selection by clearing K988/989

3) Several ms later CR or tab contacts return to normal state which clears K994/995 and K986/987.

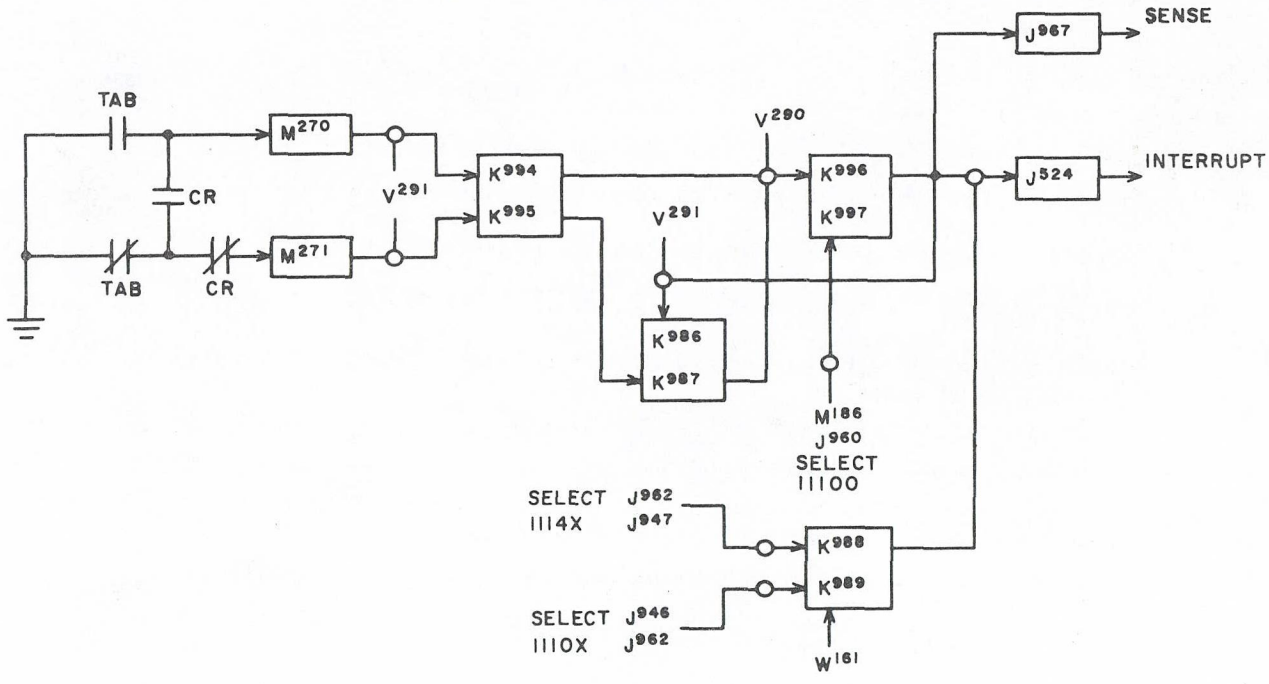


Figure 7-10. Carriage Return, Interrupt Circuit

TYPEWRITER OUTPUT OPERATION

The Output Control circuit (figure 7-11) provides for accepting a Character code from the O^1 register and sending it to the translator magnets (TM1 - TM6) of the typewriter decoder. The translator cam magnet (TCM) is energized by the circuit to type the character or perform the function.

Typewriter operation begins when the typewriter is selected and the computer signals by a Ready that O^1 is loaded. The Control circuit gates the code to TM1-TM6 and starts the mechanical cycle of operation in the typewriter. Later, this cycle closes ribbon feed or one of the function contacts. Closing one of these contacts signals to the Control circuit that TCM can be energized and that the code has energized some combination of TM1-TM6. After resynchronizing, this signal prepares the next code.

During Character mode operation the disassembly counter is held at the count for character 0; a Resume occurs after each operation. When an illegal code (one not defined in table 7-4) is gated to TM1-TM6 the decoder does not respond. As a result neither ribbon feed nor any function contact closes, and operation hangs up. Striking the CR, backspace or shift keys allows operation to resume.

A "0" code (all "0's") constitutes a do-nothing code. It is used, for example, to fill out a 48-bit word in the Assembly mode. Since a "0" code does not energize TM1-TM6, the code is sensed by J999 which initiates the necessary sequence of control signals. The steps in typewriter operation are:

1) Computer sends Ready after loading O^1 register.
2) A character in O^1, determined by disassembly counter, is transferred to TM1-TM6.
3) TCM energizes to begin typewriter cycle that prints the character or performs the function.
4) If code is for typewriter character, ribbon feed contact closes; if typewriter function, function contact closes.
5) If Character mode or Assembly mode, counter is at character 0. Step 4 sets Resume FF; this sends Resume to computer.
6) Advance disassembly counter (ineffective in Character mode).
7) Computer drops Ready because of typewriter Resume; this clears typewriter Resume FF.
8) Near the end of the typewriter cycle, the contacts open.
9) Return to step 1.

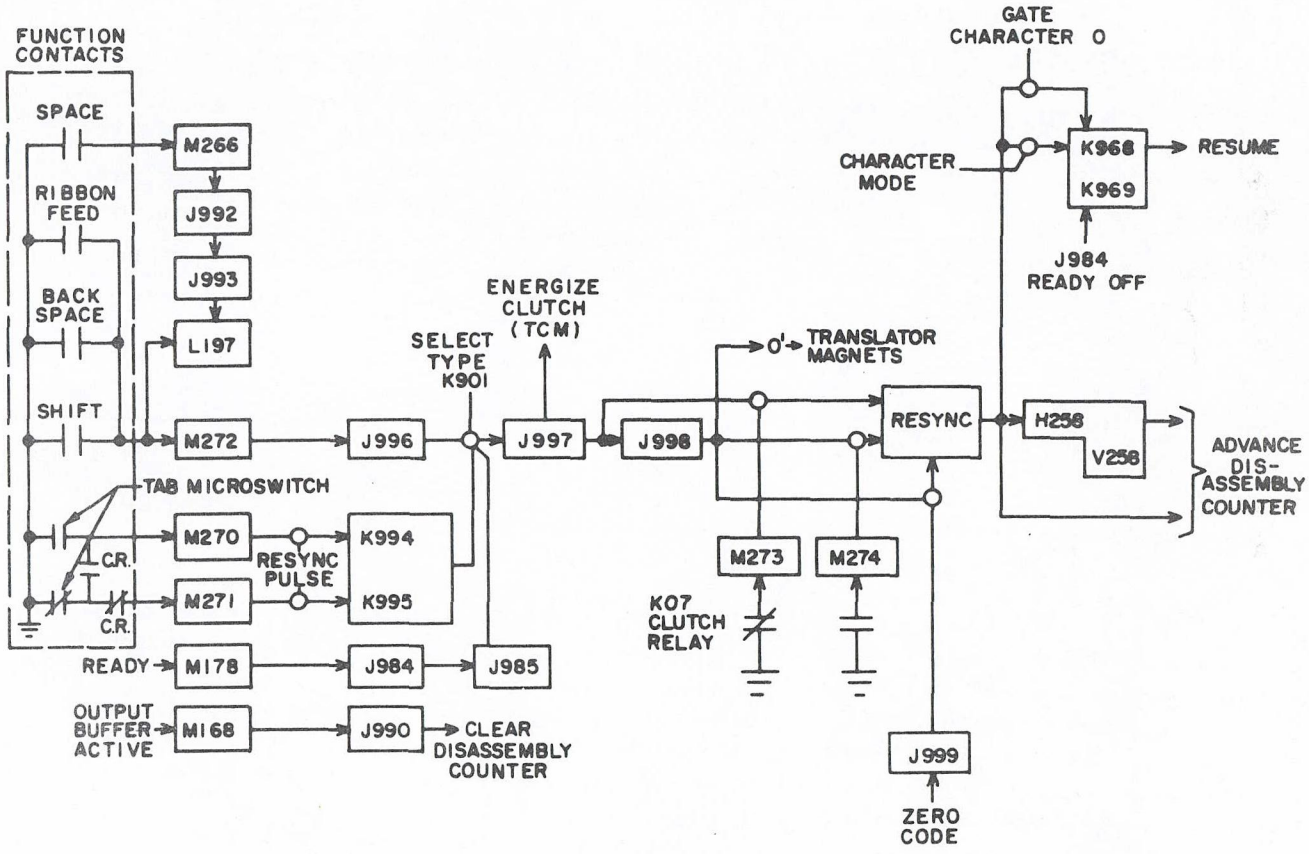


Figure 7-11. Typewriter Control Circuit (Output)

CHAPTER 8
POWER AND COOLING SYSTEM
POWER REQUIREMENTS

The 1604-A computer system requires two basic power sources: 208V, 60 cycle, 3 phase; and 208V, 400 cycle, 3 phase. The 208V, 60 cycle, 3 phase power is used for the motor-generator sets and peripheral equipment. Though all 3 phases are brought into both the 1604-A console and the logic cabinet, only the 120V phase-to-neutral is used in these two units. 400 cycle power is furnished by a motor-generator (MG) set; its output current is relatively free from power surges occurring on the input to the motor and it is easily filtered after rectification. The MG control cabinet contains dual controls for normal and standby MG sets. (See Electric Machinery Manual for a description of MG sets and Control Cabinet.)

MAIN POWER DISTRIBUTION

Figure 8-1 shows the main power distribution for the computer and the associated peripheral equipment cabinets. The 208V, 400 cycle, 3 phase power is brought directly from the MG control cabinet to the 400 cycle distribution panel.

The 208V, 60 cycle, 3 phase input is brought in directly from the line through a line contactor to the 60 cycle distribution panel. The line contactor is energized by the 1604-A Console Power On switch.

Power cables originating at the 400 cycle and 60 cycle distribution panels distribute power to the 1604-A computer and all peripheral equipments. Each of the circuit breakers on these panels protects the power inputs to peripheral equipment and allows manual disconnection if necessary. However, the usual procedure is to leave these circuit breakers on and to control the power to all peripheral equipment from the 1604-A console.

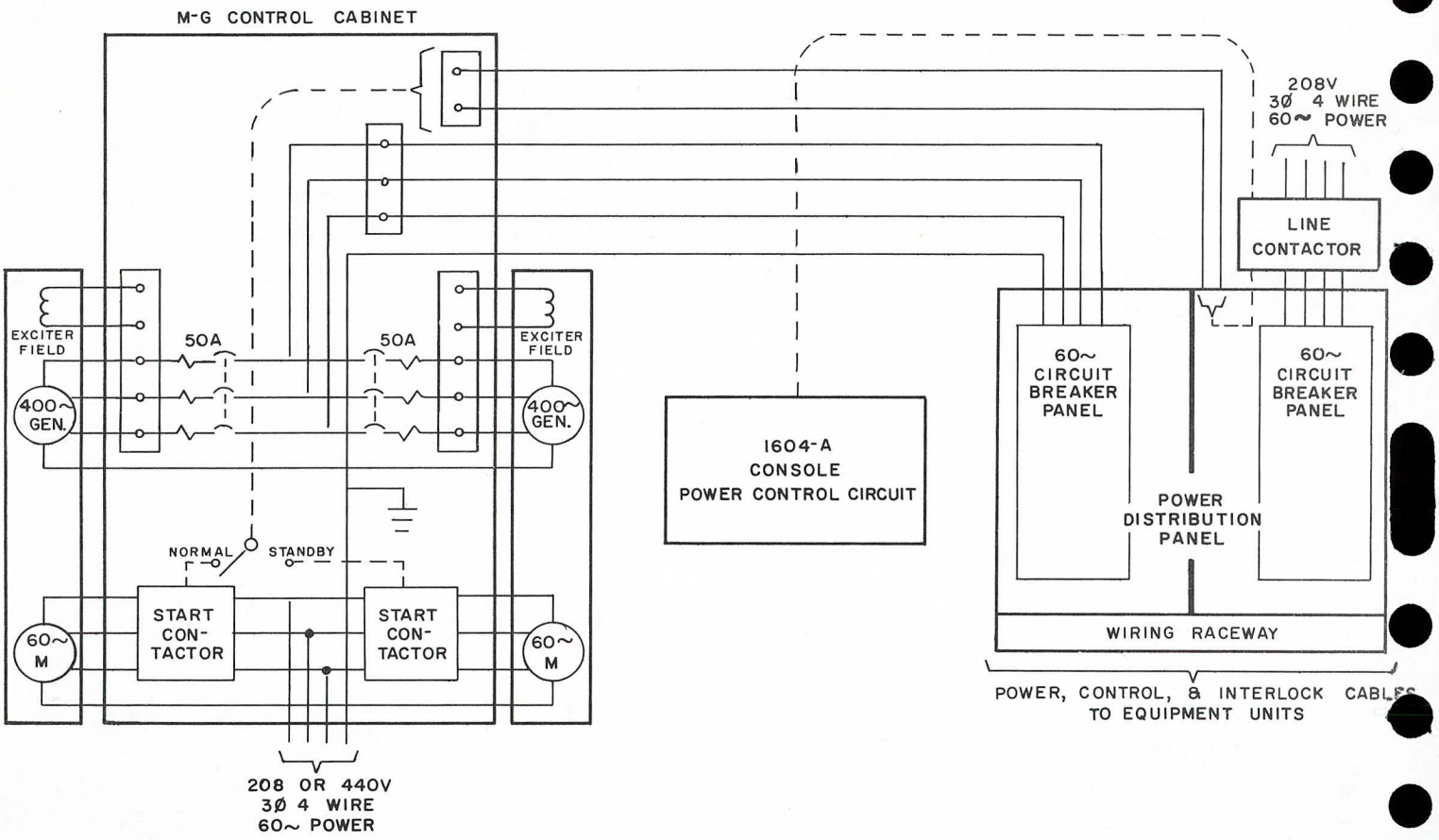


Figure 8-1. Main Power Distribution

COOLING SYSTEM

The 1604-A Main Cabinet is cooled by one of two methods:

1) Centrifugal blowers which cool the chassis by forcing air-conditioned room air through the cabinet.
2) Underfloor plenum blower system which forces air-conditioned air through the cabinet.

PROTECTIVE INTERLOCK SYSTEM

A protective interlock system, consisting of high temperature thermostats and door interlocks, protects the printed circuit cards against excessive temperatures. High temperature thermostats are mounted above the chassis in the computer cabinet. These thermostats are connected in series with a time delay relay.

If the temperature in a section of the cabinet reaches 85°F, the thermostat opens the Interlock circuit, turns on an indicator light in the cabinet, and causes a warning buzzer to sound. After a three minute delay all 400 cycle power is disconnected from the 1604-A and all equipment connected to the 400 cycle power distribution panel. The thermostats automatically reset when the temperature falls to the proper level and 400 cycle power is restored.

The door interlock system consists of four switches (one on each door). When a cabinet door is opened, the corresponding switch is closed and a time delay motor is started. If the door is not closed within 30 minutes, the timer turns on an indicator light in the cabinet and a warning buzzer sounds. After a three minute delay all 400 cycle power is disconnected. Closing the door automatically restores 400 cycle power. The door should not be reopened for at least 10 minutes after it has been closed.

The door interlock circuit also includes other equipment cabinets connected to the same 400 cycle power distribution panel. When a door on one of these cabinets is opened, the same sequence of events occurs as when one of the 1604-A cabinet doors is opened.

APPENDIX A
BORROW PYRAMID

The borrow pyramid of the accumulator is discussed in detail in this appendix. Since the pyramid associated with the U^2 register differs only in the number of bits, this discussion applies to it as well. Figure A-6 (end of appendix) shows the general structure of the pyramids and gives the state of the output of each inverter during the performance of an actual addition. The operands used for illustration in figure A-6 are:

Addend (A_i) =	000	000	000	000	011	110	110	000	111	110	000	010	110	101	001	111
Augend (X_i) =	000	000	000	000	001	010	000	111	111	100	111	101	110	110	001	011
Sum (A_f) =	000	000	000	000	101	000	111	000	111	011	000	000	101	011	011	010

The output of each inverter is indicated by the "0" or "1" in the block symbol. The derivation of the output for any block can be proven by examining the figures which accompany the discussion of the various levels.

Each level consists of single inverters which are associated with a class of symbols; for example, the bit borrow enables are identified by the symbol A--0. The name of each level is descriptive of the function it performs, or, more precisely, of the condition it senses. For any level except toggle control (A--7) the following rules can be used to determine whether it is a "1" or a "0" output from the inverters that indicates the presence of the condition sensed by that level:

1) If the last digit of the superscript is even, a "1" output represents the presence of the condition denoted by the name of the level.
2) If the last digit is odd, a "0" output represents the presence of the condition denoted by the name of the level.

In the case of A--7, a "1" output indicates that the stage is to be toggled.

The term enable refers to the condition in a stage, group or section when a borrow request cannot be satisfied and passing to the next higher stage, group, or section becomes necessary. The term generation refers to the condition in a stage, group, or section when a borrow originating within it requires action from a higher stage, group or section. The borrow enable and borrow generation conditions in a given group or section are

mutually exclusive. For example, if the enable condition exists in a given group, this group cannot generate an intergroup borrow. If the generation exists, then an intergroup borrow can be completed in the group.

INTERPRETATION OF EQUATION SYMBOLS IN PYRAMID

All symbols of inverters in the pyramid have superscripts that lie within the range 500 to 977. The last digit indicates the level or function of a class of inverters. The first two digits indicate the stage, group or section with which an inverter is associated. Thus, stage 00 is denoted by 50 while stage 47 is denoted by 97. For inverters associated with groups or sections, the first two digits are the same as those of the first stage borrow enable in the group or section. However, for the group and section borrow enables the first two digits correspond to those of the second stage in the group or section.

STAGE BORROW ENABLE

Each A--0 inverter forms the logical difference of A_n and X_n. This information about the relation of A_n and X_n is used to determine if: (1) a borrow is generated in that stage; (2) a borrow can be made from that stage without propagating further borrows; or (3) a borrow from that stage will not be completed but propagated to higher order stages.

That each A--0 forms the logical difference of A_n and X_n can be seen more easily by the following transformation performed on the equation for A500, which forms the logical difference of A_{00} and X_{00}. The horizontal bar represents the "not" function of the single inverter.

$$A^{500} = \overline{X^{003}\ A^{003} + A^{002}\ X^{002}} \qquad 1$$

$$= \overline{(X^{003}\ A^{003})\ (A^{002}\ X^{002})} \qquad \text{from 1 by De Morgan's theorem} \qquad 2$$

$$= (\overline{X^{003}} + \overline{A^{003}})\ (\overline{A^{002}} + \overline{X^{002}}) \qquad \text{from 2 by De Morgan's theorem} \qquad 3$$

$$= \overline{A^{003}}\ \overline{X^{002}} + \overline{A^{002}}\ \overline{X^{003}} \qquad \text{Multiply out 3} \qquad 4$$

(A_{00} = "0" and X_{00} = "1") (A_{00} = "1" and X_{00} = "0")

The output of A--0 is "1" when the logical difference of A_n and X_n is "1". Such an output from A--0 indicates a stage borrow enable; that is, a borrow from this stage in turn

requires a borrow from the next higher stage. A "0" output from an A--0 indicates that either a borrow is generated in this stage or a borrow can be completed in this stage.

GROUP BORROW ENABLE

Each of the A--1 inverters in the group borrow enable level senses whether an intergroup borrow into the group can be completed or if it necessitates an intergroup borrow from the next group. A "0" output from a group borrow enable indicates that the borrow cannot be completed. This condition is sensed by combining, in an AND function, the outputs of the three stage borrow enables in the group.

GROUP BORROW GENERATION

Each of the A--1 inverters in the group borrow generation level senses whether an intergroup borrow is generated within its group. When this is the case, the output of the inverter is "0". A "1" output indicates that no intergroup borrow is generated.

In figure A-1 the three inputs to A501 illustrate how a group borrow generation senses the presence of an intergroup borrow. The input consisting of X022 and A022 specifies that X_{02} and A_{02}, the last stages in the group, are both "0". Since it is the complement of X that is subtracted from A, and this case involves subtracting "1" from "0" in the last stage of a group, an intergroup borrow is necessary.

The input consisting of X012, A012, and A520 indicates that A_{01} generated a borrow from A_{02} and that in the latter stage a borrow cannot be completed. Thus a borrow is again required from A_{03}. Finally, the A002, X002, A510, and A520 input indicates that A_{00} generated a borrow, but a borrow cannot be completed in either A_{01} or A_{02}. An intergroup borrow is therefore required from A_{03}.

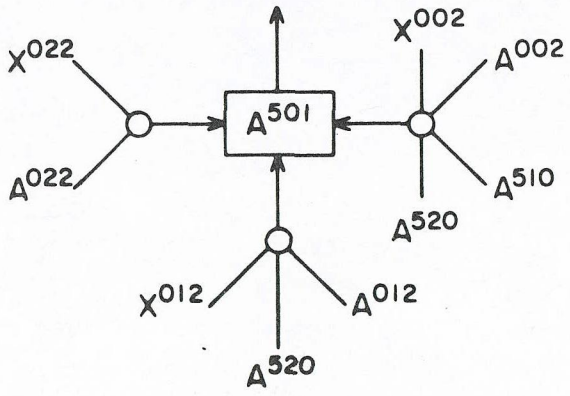


Figure A-1. Typical Group Borrow Generation Inverter

SECTION BORROW ENABLES

Each of the A--2 inverters in the section borrow enable level senses whether a borrow into its section can be completed or whether it necessitates a borrow from the next section to the left. A "1" output shows that a borrow cannot be completed in that section. A section borrow enable is sensed by combining the ouput of the four group borrow enables in the section at the OR inputs of the inverter. Since the signals are represented by "0's" in this case, the circuit performs a logical AND function.

SECTION BORROW GENERATION

Each of the A--2 inverters in the section borrow generation level senses whether a borrow is generated within its section. When this is the case the output of the inverter is "1".

The inputs of a section borrow generation actually sense for the four cases when no intersection borrow is generated. Thus for A502 in figure A-2, the input consisting of A501, A531, A561, and A591 provides for the case of no intergroup borrows being generated in the section.

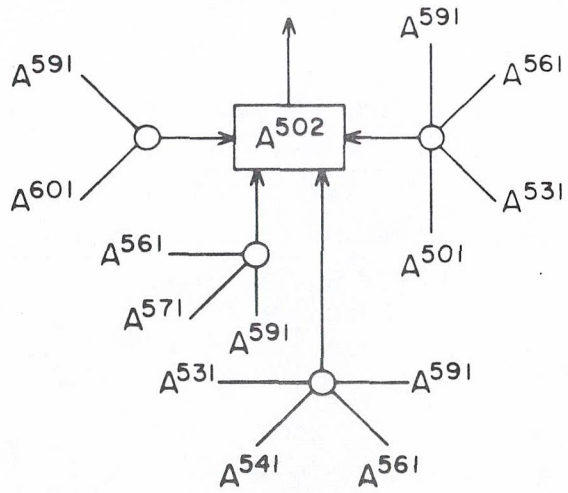


Figure A-2. Typical Section Borrow Generation Inverter

The A531, A541, A561, and A591 input senses the case when no intergroup borrows are generated in groups 3 and 4 (A561 and A591) and a borrow can be completed in group 2 (A531 and A541). The case of no borrow generated in group 4 and the possibility

of completing a borrow in group 3 is sensed by A591, A571, and A561. Finally, the A591 and A601 input handles the case when a borrow can be completed in group 4.

SECTION BORROW INPUT

Each of the four sections has an A--3 lead which senses when an intersection borrow is to be made from its section. This condition, which is shown by a "0" output from the A--3, is sensed by means of the output of the section borrow enables and generations.

A detailed examination of the four inputs to A503 (figure A-3) will make clear the function of the section borrow input inverters. Each of the inputs indicates an intersection borrow; these are also end-around borrows since they come from section 4. The inputs are mutually exclusive.

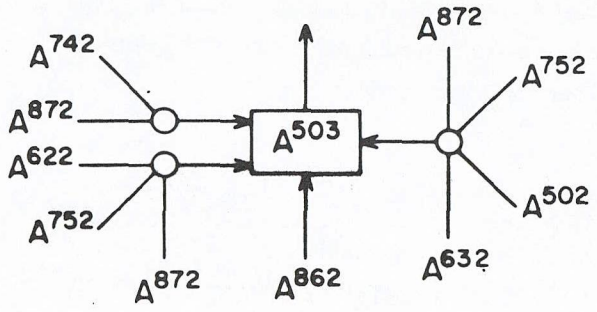


Figure A-3. Typical Section Borrow Input Inverter

The input from A862, the section borrow generation inverter in section 4, provides for the simple case when an intersection borrow is generated in section 4. The AND input consisting of A742 and A872 provides for an intersection borrow generated in section 3 (A742) which cannot be completed in section 4 (A872). The AND input consisting of A622, A752, and A872 provides for an intersection borrow generated in section 2 (A622) which cannot be completed in either section 3 (A752) or section 4 (A872). Finally, the AND input consisting of A502, A632, A752, and A872 provides for an intersection borrow generated in a higher order stage of section 1 which cannot be completed in sections 2, 3, or 4 and must, therefore, be made from a lower stage of section 1.

GROUP BORROW INPUT

Each group has an A--4, the purpose of which is to determine whether an intergroup borrow is required from the group. When one is required all the inputs to A--4 and "0" and, therefore, its output is "1".

Inputs to A564 are shown in figure A-4. Because of the inversion in A--4 its inputs may be said to sense the cases when no intergroup borrow is required. The input consisting of A503, A531, and A501 senses that there is no intersection borrow (A503) and there are no intergroup borrows generated in either of the two lower groups (A531 and A501). The A531, A511, and A501 input shows that an intergroup borrow is not generated in the second group (A531) and that an intergroup borrow can be completed in the first group (A511 and A501).

The latter condition is determined by the information that an intergroup borrow is neither generated nor enabled in the first group. Finally, the A541 and A531 input indicates a borrow can be accomplished in the second group and therefore, an intergroup borrow is not required of the third group.

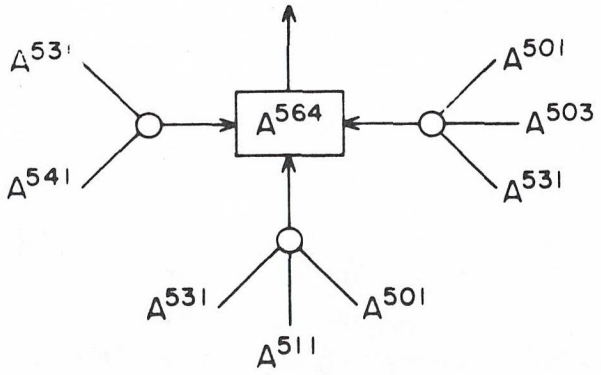


Figure A-4. A Group Borrow Input Inverter

STAGE BORROW INPUT

Each stage has an A--5, the purpose of which is to sense when a borrow is required from that stage. The inputs to the three A--5 inverters of the second group (figure A-5) are typical of the A--5 inverters of other groups which have similar inputs.

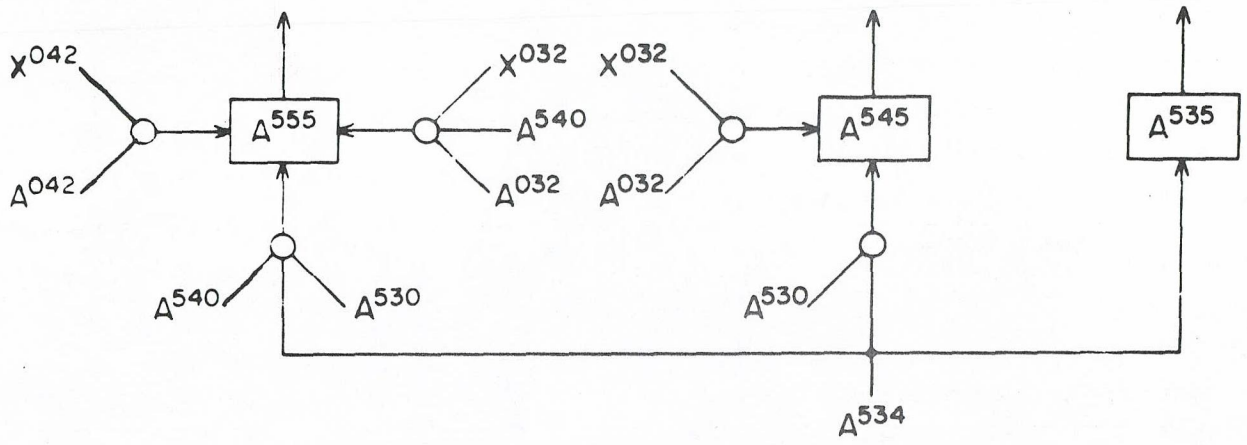


Figure A-5. Typical Stage Borrow Input Inverter

A borrow is required from A535, the lowest stage of the group, only when an intergroup borrow is made from the group. The input from A534 handles this.

There are two cases when a borrow must be made from A545, the middle stage. The first instance, when a borrow is generated in the lowest stage of the group, is sensed by the A032 and X032 input, which indicates that both A_{03} and X_{03} are "0". This denotes a "0 minus 1" operation since it is the complement of X that is subtracted. The second case exists when an intergroup borrow (A534) must be made from the group and a bit borrow enable is present in the first stage (A530).

A borrow is required from the third stage (A555) in three cases: (1) when a borrow is generated in the second stage of the group (A042 and X042); (2) when a borrow is generated in the first stage (A032 and X032) and a bit borrow exists in the second (A540); (3) (A534, A540, and A530), when this group receives an intergroup borrow (A534) and there are bit borrow enables in the first (A530) and second (A540) stages.

If a borrow is required from a stage, the output of the A--5 associated with that stage is "0". Every A--5 has an input from W00-. During the addition of X^2 to A^1 the input to A--5 from W00- is "0". But when A^2 is to be complemented on the basis of "1's" in X^1, then W00- is used to disable the borrow sensing portion of the pyramid by providing a "1" input to every A--5. Thus the output of every A--5 is "0" regardless of whether a borrow is required from its stage.

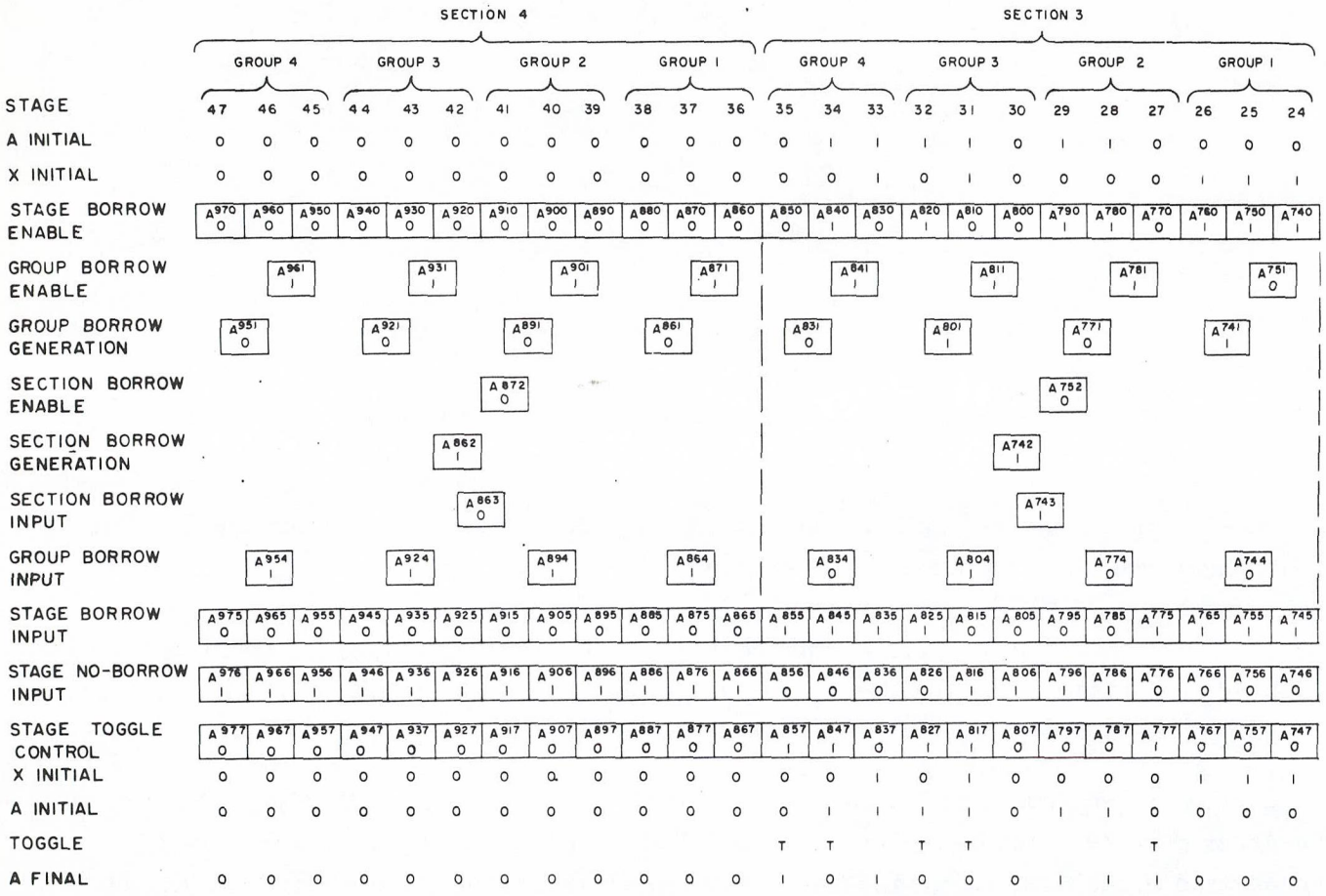


Figure A-6. General Structure of the Pyramid

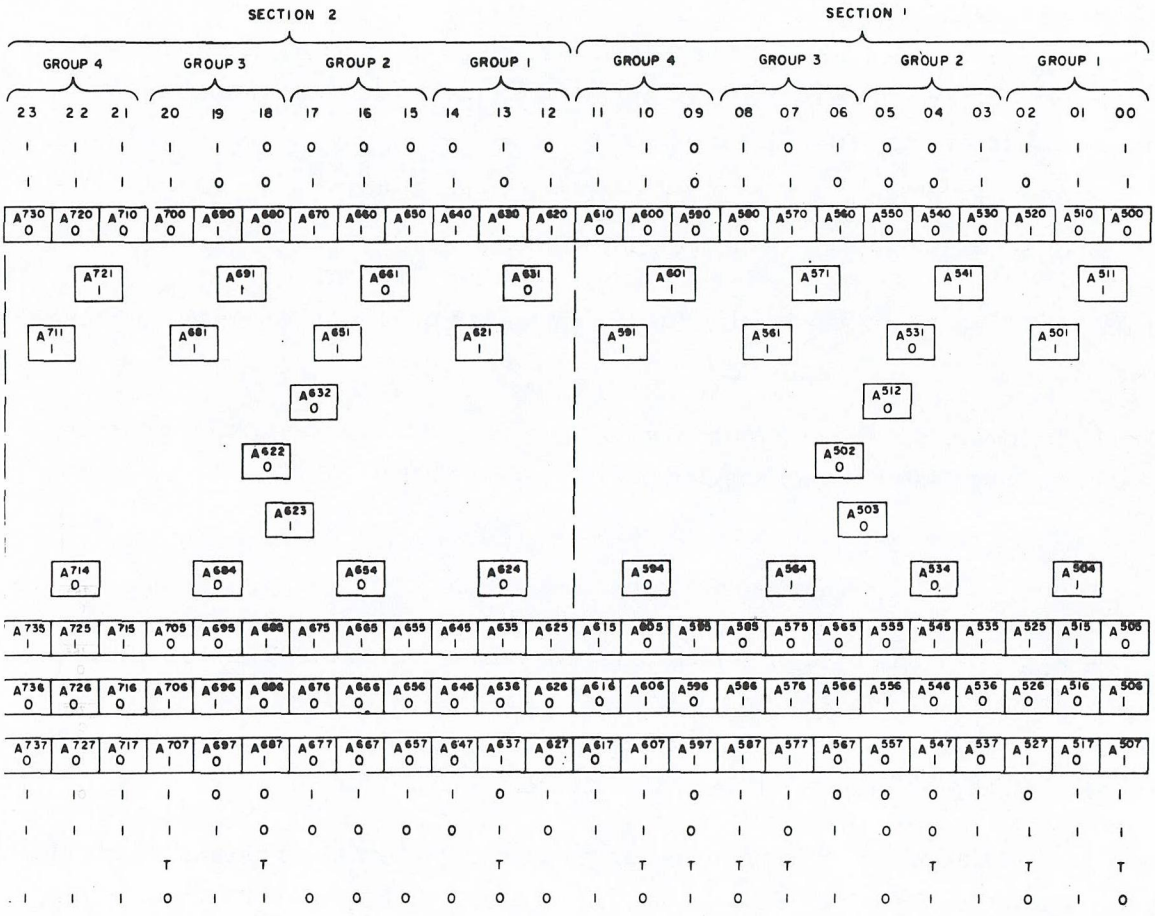


Figure A-6. General Structure of the Pyramid (cont'd.)

STAGE NO-BORROW INPUT

Each stage has an A--6 which inverts the borrow information obtained from the A--5 of the stage. The output of an A--6 is "1" when a borrow is required from the stage. The function of A--6 is shown in the analysis of the A--7 inverters.

STAGE TOGGLE CONTROL

The A--7 of each stage senses whether or not that stage is to be toggled. At this point it is well to have clearly in mind the two cases when the addition of X to A requires a given stage of A to be toggled. The cases are:

1) X_n = "0" and no borrow is required from A_n
2) X_n = "1" and a borrow is required from A_n

In addition, a "1" out of an A--7 indicates the presence of one or the other of these two cases.

Because of the inversion of A--7 their inputs may be considered as sensing the two cases when the stage is not to be toggled. These cases are:

1) X_n = "0" and a borrow is required from A_n
2) X_n = "1" and no borrow is required from A_n

Using stage A_{00} as an example, the first case is sensed by the X002 and A506 input to A507 while the second is sensed by the X003 and A505 input. In the first case, X002 indicates X_{00} = "0" and A506 indicates a borrow is required. In the second case, X_{00} indicates X003 = "1" and A505 indicates no borrow is required.

When this part of the pyramid is used in complementing A^1 on the basis of "1's" in X^2, the Partial Add in A FF (K520/521) is set. It in turn causes the W00- inverters to enter a "1" in each A--5. Therefore, the A--5 and X--3 AND input to each A--7 is not satisfied. The other input, A--6 and X002, is satisfied if X is "0". If X_n = "0" then the A--7 receives a "1" and its output is "0". Consequently, A_n is not toggled. However, when X_n = "1" neither input is satisfied and a "1" output is provided, causing A_n to be toggled.

APPENDIX B
GLOSSARY OF TERMS

This glossary gives the meaning of terms that are used in a relatively specialized sense in this manual.

ACCUMULATOR	A register with provisions for the addition of another quantity to its content. It is also the name of the A register.
ADDRESS	The number assigned as a designation for a memory location; also used to refer to the memory location itself.
AND FUNCTION	A logical function in Boolean algebra that is satisfied (has the value "1") only when all of its terms are "1's". For any other combination of values it is not satisfied and its value is "0".
BIT	Binary digit, either "1" or "0".
BORROW	In a subtractive counter or accumulator, a signal indicating that in stage n, a "1" was subtracted from a "0".
BUFFER	To store data temporarily. The operation in which either a word from storage is sent to an external equipment via an output channel (output buffer), or a word is sent from an external equipment via an input channel to storage (input buffer).
CARD	Any etched wiring board and attached components mounted on a chassis by means of a 15-pin connector.
CARRY	In an additive counter or accumulator, a signal indicating that in stage n, a "1" was added to a "1".
CHAIN	A group of control delays connected serially for timing commands and signals.
CHANNEL	A transmission path that connects the computer to a given external equipment.

CHARACTER	Two types of information handled by the computer: 1) A group of 6 bits which represent a digit, letter or symbol. In the Assembly mode, eight 6-bit characters make up a computer word. 2) A group of 7 bits which represent an item of information. In the Character mode, this item is one 7-bit character with "0's" in the remaining (upper) 41 bits.
CLEAR	A command that removes a quantity from a register by placing every stage of the register in the "0" state.
CLOCK PHASE	One of two outputs from the master clock, "even" or "odd". Inputs of a control delay are gated by one clock phase, outputs by the other.
COMMAND	A signal that performs a unit operation, such as transmitting the content of one register to another, shifting a register one place or setting a FF.
CONTENT	The quantity or word held in a register or storage location.
CONTROL DELAY	Receives and holds a signal pulse for a period determined by the basic clock frequency (0.2 microsecond).
CORE	A ferromagnetic toroid used as the bistable device for storing a bit in a memory plane.
COUNTER	A register with provisions for increasing or decreasing its content by 1.
ENABLE	To satisfy one of the conditions required for the occurrence of a command.
END-AROUND BORROW	A borrow generated in the highest-order stage of an accumulator or counter and sent directly to the lowest-order stage.
ENTER	To place a quantity, not from storage, in a register.
EVEN STORAGE	The storage unit which contains the 16,384 even addresses.

EXECUTION ADDRESS	The lower 15 bits of a 24-bit instruction, most often used to specify the storage address of an instruction operand, sometimes used as the operand.
EXIT	Initiation of a second Control sequence by the first, occurring when the first is near completion. The circuit involved in exiting.
EXTERNAL FUNCTION	1) External Function Select (74. 0) sends a code to an external equipment to direct its operation. 2) External Function Sense (74. 7) sends a code to an external equipment to sense its operating condition.
FAULT	Operational difficulty either stops operation or sets an indicator.
FLIP-FLOP (FF)	A bistable storage device. A "1" input to the set side puts the FF in the "1" state; a "1" input to the clear side puts the FF in the "0" state. The FF remains in a state indicative of its last "1" input. A stage of a register consists of a FF.
FUNCTION CODE	The upper 9 bits of a 24-bit instruction consisting of the Operation and index Codes.
GATE	To satisfy one of the conditions required for the occurrence of a signal. As a noun, may denote either the AND or the OR function.
INDEX CODE	A 3-bit quantity, bits 15, 16, and 17, of an instruction; usually specifies an index register whose contents are added to the execution address; sometimes specifies the conditions for executing the instruction.
INSTRUCTION	A 24-bit quantity consisting of a Function code, index designator, and execution address.
INTERRUPT REQUEST	A signal indicating to the computer that a fault or other Interrupt condition exists.
INVERTER	A circuit, which provides as an output a signal that is opposite to its input.

LOAD	To place a quantity from storage in a register.
LOGICAL PRODUCT	In Boolean algebra, the AND function of several terms. The product is "1" only when all the terms are "1"; otherwise it is "0". Sometimes referred to as the result of "bit-by-bit" multiplication.
LOGICAL SUM	In Boolean algebra, the OR function of several terms. The sum is "1" when any or all of the terms are "1"; it is "0" only when all are "0".
LOWER ADDRESS	The execution address portion of a lower instruction, bits 0 through 14 of a 48-bit register or storage location.
LOWER INSTRUCTION	See Program Word.
MASK	In the formation of the logical products of two quantities, one quantity may mask the other, i.e., determine what part of the other quantity is to be considered. If the mask is "0" that part of the other quantity is cleared; if the mask is "1", the other quantity is left unaltered.
MASTER CLEAR (MC)	A general command produced by placing the CLEAR switch up (external MC) or down (computer MC) which clears all the crucial registers and control FFs.
MODULUS	An integer which describes certain arithmetic characteristics of registers, especially counters and accumulators, within a digital computer. The modulus of a device is defined by r^n for an open-ended device and r^n-1 for a closed (end-around) device, where r is the base of the number system used and n is the number of digit positions (stages) in the device. Generally, devices with modulus r^n use two's complement arithmetic procedures while devices with modulus r^n-1 use one's complement procedures.
NORMAL JUMP	An instruction that jumps from one sequence of instructions to a second, and makes no preparation for returning to the first sequence.

ODD STORAGE	The storage unit which contains the 16, 384 odd addresses.
ONE'S COMPLEMENT	With reference to a binary number, that number which results from subtracting each bit of the given number from the bit "1". A negative number is expressed by the one's complement of the corresponding positive number.
OPERAND	Usually refers to the quantity specified by the execution address. This quantity is operated upon the execution of the instruction.
OPERATION CODE	The upper six bits of a 24-bit instruction which identify the instruction. After the code is translated, it conditions the computer for execution of the specified instruction.
OR FUNCTION	A logical function in Boolean algebra that is satisfied (has the value "1") when any of its terms are "1". It is not satisfied when all terms are "0". Often called the 'inclusive' OR function.
OVERFLOW	The capacity of a register is exceeded.
PARTIAL ADD	An addition without carries. Accomplished by toggling each bit of the augend where the corresponding bit of the addend is a "1".
PROGRAM	A precise sequence of instructions that accomplished a computer routine; a plan for the solution of a problem.
PROGRAM WORD	Two 24-bit instructions contained in one 48-bit storage address; the higher-order 24 bits are the upper instruction, lower-order 24 bits, bht lower instruction. A pair of instructions is read from storage and the upper instruction is executed first. The lower one is then executed, except when the upper one provides for skipping the lower one.
PYRAMID	For the A or U^2 registers, a network of inverters that senses borrow conditions and produces Borrow signals.
RANK	Registers composed of a pair of flip-flops per stage consist of two ranks, each containing one FF from the pair for each stage. Inverters arranged for parallel transmissions with one inverter per bit are also called ranks.

READ	To remove a quantity from a storage location.
READY	The input-output control signal sent by either the computer or an external equipment to alert the device that is to receive a transmission. The ready signal indicates that the word or character has been transmitted.
REPLACE	In the title of an instruction, the result of the execution of the instruction is stored in the location from which the initial operand was obtained.
RESUME	The input-output control signal sent by either the computer or an external equipment to indicate that it is prepared to receive another word (48 bits) or character (usually 6 bits). The resume signal is thus a request for data.
RETURN JUMP	An instruction that jumps from a sequence of instructions to initiate a second sequence and prepares for continuing the first sequence after the second is completed.
ROUTINE	The sequence of operations which the computer performs under the direction of a program.
SHIFT	To move the bits of a quantity right or left.
SIGN BIT	In registers where a quantity is treated as signed by use of one's complement notation, the bit in the highest-order stage of the register. If the bit is "1", the quantity is negative; if the bit is "0", the quantity is positive.
SIGN EXTENSION	The duplication of the sign bit in the higher-order stages of a register.
SKIP	To omit the execution of a lower instruction in a program; occurs only if the upper instruction provides for skipping on a specified condition, and the condition is met.
SLAVE	An inverter or FF which receives an unconditional input from another circuit and is used to provide more outputs than are available from the first.

STAGE	The FFs and inverters associated with a bit position of a register.
SUBINSTRUCTION	The index code specifies one of eight forms of the instruction indicated by the operation code. Such forms are called "subinstructions". Thus, 74.0 is a subinstruction of instruction 74.
TOGGLE	To complement each bit of a quantity as a result of an individual condition.
TRANSMISSION CLEARED	A transmission where only "1's" are transferred into a register which has not been previously cleared.
TRANSMISSION FORCED	A transfer of bits into a register which has not been cleared previously.
TRANSLATION	An indication of the content of a group of bit registers. A complete translation gives the exact content, while a partial translation indicates only that the content is within certain limits.
TWO'S COMPLEMENT	Number that results from subtracting each bit of a number from "0". The two's complement may be formed by complementing each bit of the given number and then adding one to the result, performing the required carries.
UPPER ADDRESS	The execution address portion of an upper instruction; bit positions 24 through 38 of a 48-bit register or storage address.
UPPER INSTRUCTION	See Program Word.
WORD	A unit of information which has been coded for use in the computer as a series of bits. Normal word length is 48 bits.
WRITE	To enter a quantity into a storage location.

APPENDIX C

LIST OF INSTRUCTIONS

00	ZRO	(not used)	Fault
01	ARS	A Right Shift	Shift (A) Right by K
02	QRS	Q Right Shift	Shift (Q) Right by K
03	LRS	AQ Right Shift	Shift (AQ) Right by K
04	ENQ	Enter Q	Y → Q, Extend Sign Y
05	ALS	A Left Shift	Shift (A) Left by K
06	QLS	Q Left Shift	Shift (Q) Left by K
07	LLS	AQ Left Shift	Shift (AQ) Left by K
10	ENA	Enter A	Y → A, Extend Sign Y
11	INA	Increase A	$[Y + (A)] \rightarrow A$, Extend Sign Y
12	LDA	Load A	(M) → A
13	LAC	Load A, Complement	(M)' → A
14	ADD	Add	$[(A) + (M)] \rightarrow A$
15	SUB	Subtract	$[(A) - (M)] \rightarrow A$
16	LDQ	Load Q	(M) → Q
17	LQC	Load Q, Complement	(M)' → Q
20	STA	Store A	(A) → M
21	STQ	Store Q	(Q) → M
22	AJP	A Jump *	Jump to m on condition j
23	QJP	Q Jump *	Jump to m on condition j
24	MUI	Multiply Integer	(M) (A) → QA
25	DVI	Divide Integer	(QA)/(M) → A; Remainder=Q_f
26	MUF	Multiply Fractional	(M) (A) → AQ
27	DVF	Divide Fractional	(AQ)/(M) → A; Remainder=Q_f
30	FAD	Floating Add	$[(A) + (M)] \rightarrow A$
31	FSB	Floating Subtract	$[(A) - (M)] \rightarrow A$
32	FMU	Floating Multiply	(M) (A) → A
33	FDV	Floating Divide	(A)/(M) → A
34	SCA	Scale A	A left until $\lvert(A)\rvert \geq .5$ or k=0; k-No. of Shifts → B^b
35	SCQ	Scale AQ	AQ left until $\lvert(AQ)\rvert \geq .5$ or k=0; k-No. of Shifts → B^b
36	SSK	Storage Skip #	(M_{47}) Neg: EXIT; (M_{47}) Pos: Half EXIT
37	SSH	Storage Shift #	(M_{47}) Neg: EXIT, left 1; (M_{47}) Pos: Half EXIT, left 1
40	SST	Selective Set	Set (A_n) for (M_n) = 1
41	SCL	Selective Clear	Clear (A_n) for (M_n) = 1
42	SCM	Selective Complement	Complement (A_n) for (M_n) = 1
43	SSU	Selective Substitute	(M_n) → (A_n) for (Q_n) = 1
44	LDL	Load Logical	L(Q) (M) → A
45	ADL	Add Logical	$[(A) + L(Q) (M)] \rightarrow A$

Legend

b - desig. for indexing
j - desig. for 22, 23, 74-76
k - exec. add. as shift cnt.
K - $k + (B^b)$
m - exec. add. as op. add.
M - $m + (B^b)$
y - exec. add. as operand
Y - $y + (B^b)$
- skip inst. (up. posit.)

46	SBL	Subtract Logical	$[(A) - L(Q)\ (M)] \rightarrow A$
47	STL	Store Logical	$L(Q)\ (A) \rightarrow M$
50	ENI	Enter Index	$y \rightarrow B^b$; b=0: pass
51	INI	Increase Index	$y + (B^b) \rightarrow B^b$
52	LIU	Load Index (upper)	$(m_{UA}) \rightarrow B^b$
53	LIL	Load Index (lower)	$(m_{LA}) \rightarrow B^b$
54	ISK	Index Skip #	$(B^b) \neq y$: $B^b + 1 \rightarrow B^b$, NI; $(B^b) = y$: $0 \rightarrow B^b$ Skip NI
55	IJP	Index Jump	$(B^b) \neq 0$: $B^b - 1 \rightarrow B^b$, Jump to m; $(B^b) = 0$: NI
56	SIU	Store Index (upper)	$(B^b) \rightarrow m_{UA}$
57	SIL	Store Index (lower)	$(B^b) \rightarrow m_{LA}$
60	SAU	Substitute Address (up.)	$(A_{00-14}) \rightarrow M_{UA}$
61	SAL	Substitute Address (lwr.)	$(A_{00-14}) \rightarrow M_{LA}$
62	INT	Input Transfer	(B^b) words to memory start at M-1
63	OUT	Output Transfer	(B^b) words to memory start at M-1
64	EQS	Equality Search #	Search (B^b) words if (M-1), (M-2), etc. = (A) Skip NI
65	THS	Threshold Search #	Search (B^b) words if (M-1), (M-2), etc. > (A) Skip NI
66	MEQ	Masked Equality #	Search (B^b) words if L(Q) (M-1), (M-2), etc. = (A) Skip NI
67	MTH	Masked Threshold #	Search (B^b) words if L(Q) (M-1), (M-2), etc. > (A) Skip NI
70	RAD	Replace Add	$[(M) + (A)] \rightarrow M$ & A
71	RSB	Replace Subtract	$[(M) - (A)] \rightarrow M$ & A
72	RAO	Replace Add One	$[(M) + 1] \rightarrow M$ & A
73	RSO	Replace Subtract One	$[(M) - 1] \rightarrow M$ & A
74	EXF	External Function	j=1 - 6; activate ch. j, j=0: sel. ext. equip. m, j=7#: skip on cond. m
75	SLJ	Selective Jump *	Jump to m on condition j
76	SLS	Selective Stop *	Stop on j, and Jump to m*
77	SEV	(not used)	Fault

DESIGNATOR FOR * INSTRUCTIONS

22	23	75	76
0 (A) = 0: Jump	(Q) = 0: Jump	Jump	Jump, Stop
1 (A) ≠ 0: Jump	(Q) ≠ 0: Jump	Key 1: Jump	Jump; Key 1: Stop
2 (A) Pos: Jump	(Q) Pos: Jump	Key 2: Jump	Jump; Key 2: Stop
3 (A) Neg: Jump	(Q) Neg: Jump	Key 3: Jump	Jump; Key 3: Stop
4 (A) = 0: Ret. Jump	(Q) = 0: Ret. Jump	Ret. Jump	Ret. Jump, Stop
5 (A) ≠ 0: Ret. Jump	(Q) ≠ 0: Ret. Jump	Key 1: Ret. Jump	Ret. Jump; Key 1: Stop
6 (A) Pos: Ret. Jump	(Q) Pos: Ret. Jump	Key 2: Ret. Jump	Ret. Jump; Key 2: Stop
7 (A) Neg: Ret. Jump	(Q) Neg: Ret. Jump	Key 3: Ret. Jump	Ret. Jump; Key 3: Stop

* 1, 2, AND 3 REFER TO SELECTIVE JUMP ON STOP KEY SWITCHES.

COMMENT SHEET

CONTROL DATA 1604A COMPUTER
Customer Engineering Instruction Manual, Part 1, Theory of Operation
Pub. No. 60118700

FROM NAME : _____

BUSINESS ADDRESS : _____

COMMENTS: (DESCRIBE ERRORS, SUGGESTED ADDITION OR DELETION AND INCLUDE PAGE NUMBER, ETC.)

CUT ALONG LINE

NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE STAPLE

FOLD FOLD

FIRST CLASS
PERMIT NO. 8241

MINNEAPOLIS, MINN.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION
8100 34TH AVENUE SOUTH
MINNEAPOLIS 20, MINNESOTA

ATTN: TECHNICAL PUBLICATIONS DEPT.
COMPUTER DIVISION
PLANT TWO

FOLD FOLD

CUT ALONG LINE

STAPLE STAPLE

►►CUT OUT FOR USE AS LOOSE-LEAF BINDER TITLE TAB

CONTROL DATA
CORPORATION
8100 34th AVE. SO., MINNEAPOLIS, MINN. 55440

PRINTED IN U.S.A.

Customer Engineering
Instruction Manual

CONTROL DATA 1604-A COMPUTER

PART 2,
MAINTENANCE

CONTROL DATA
CORPORATION

Customer Engineering
Instruction Manual

CONTROL DATA 1604-A COMPUTER

PART 2,
MAINTENANCE

CONTROL DATA
CORPORATION

Record of Revisions

REVISION	NOTES
A	Correction of misnumbered pages. Replace pages 4-55
(2-2-65)	through 4-70 with revised pages 4-55 through 4-70.

Pub. No. 60118800
February 1965

©1965, Control Data Corporation
Printed in the United States of America

Address comments concerning this manual to:

Control Data Corporation
Technical Publication Department
4201 North Lexington Avenue
St. Paul, Minnesota 55112

or use Comment Sheet located in the rear of this book.

CONTENTS

Chapter 1. Preventive Maintenance

Introduction 1-1
Preventive Maintenance 1-1
Diagnostic Routines 1-2
Marginal Tests 1-3

Chapter 2. Corrective Maintenance

Locating Failing Equipment 2-1
Locating Failing Components 2-2
Recommended Maintenance and Test Equipment 2-5
Computer Identification Numbering System 2-5
Chassis Numbering 2-6
1000 Cabinet (Main Computer) 2-6
2000 Cabinet (Console) 2-6
Connectors on a Standard Chassis 2-7
Component Numbering 2-8
Components on a Standard Chassis 2-8
Printed Circuit Cards on a Standard Chassis 2-8
Components in a Cabinet but not on a Chassis 2-8

Chapter 3. Diagnostic Maintenance

Introduction 3-1
Test Programs 3-1
1604-A Command Test 3-1
Description 3-1
1604-A Memory Test 3-2
Description 3-2
General Description and Philosophy of Test 3-2
Description of Individual Routines 3-2
1604-A Console Punch and Reader Test 3-4
Description 3-4
1604-A Console Typewriter Test 3-5
Description 3-5
Diagnosis from Console 3-5
Logical Circuit Maintenance 3-6
Test Mode 3-7
Storage Waveforms 3-7
Preventive Maintenance Schedule 3-7
Hourly 3-7
Daily 3-8
Weekly 3-8
Monthly 3-8
Semiannually and Annually 3-9
Cabling Information 3-9

Chapter 4. Command Timing Charts

Introduction 4-1
Glossary of Abbreviations 4-2
Timing Charts 4-3

Chapter 5. Maintenance Parts List

FIGURES

2-1 Cable Connector Identification 2-7
2-2 Card Side of a Typical Chassis 2-9

TABLES

3-1 Cable Identification 3-10
3-2 Connector Pin Number Assignments 3-11

CHAPTER 1
PREVENTIVE MAINTENANCE

INTRODUCTION

This manual presents general information for maintaining the basic 1604-A system. Its scope is not intended to provide absolute instructions for locating and repairing circuits in error, but rather to serve as a guide towards efficient maintenance of the system. Of primary importance to proper maintenance is a complete and thorough knowledge of the equipment.

Supplementary reference sources are:

1) Customer Engineering Diagrams Manual CDC 60024300
2) Customer Engineering Instruction Manual, Part 1
   Theory of Operation CDC 60118700
3) Reference Manual CDC 60024500
4) Printed Circuits Manual CDC 60042900
5) File of Equations
6) Chassis Maps
7) Illustrated Parts Catalog, Printed Circuits CDC 60040800

There are two basic types of maintenance: preventive and corrective. Preventive maintenance limits "down time" by anticipating and averting problems before they occur. It consists of such procedures as lubricating, cleaning, running test programs, and checking for worn or marginal parts. Corrective maintenance is the systematic location of an operating fault and its correction in order to return the system to operation in the shortest possible time.

PREVENTIVE MAINTENANCE

Many potential failures can be avoided by simple routine procedures, designed to keep already functional equipment operating at peak efficiency. One of the most important, yet most elementary aspects of preventive maintenance is cleanliness. Unless the computer area is kept clean, faults due to dirt, cigarette ashes, etc. will eventually occur. Cleanliness should be a constant consideration. The operators must also be aware of their responsibilities toward a clean orderly area, especially in regard to smoking.

Mechanical adjustments are the most frequently neglected periodic procedures. Most mechanical equipment manuals contain lists of routine maintenance procedures and the advised frequency of their application. Visual inspection of mechanical devices should take place as often as possible. Minor changes in the adjustment of a particular piece of equipment can often be detected. These frequent checks will also increase personnel's familiarity with the equipment; this will prove especially useful in cases of down time because the maintenance man will know what the settings should be.

The exact voltage output of every power supply in the system should be measured, otherwise trouble can occur if a supply has to be replaced or the voltage adjustments changed. If, after changing a supply, the voltage outputs are not set at the same value as they were originally, the times of all delays in that unit will be different. This will especially cause difficulty in tape units.

Coordination is an essential part of periodic maintenance. Since there are a great number of people working on a particular device at different times, strict control of scheduled maintenance, especially adjustments, should be kept. Where possible, responsibility for specific work on equipment should be delegated. Exactly what was done and the date on which it was done should be recorded in the maintenance log. Frequency of preventive maintenance should depend upon operating hours rather than chronologic time. Careful control of schedule records can eliminate the possibility of over-servicing a device. This does not mean that visual inspection should be limited.

DIAGNOSTIC ROUTINES

If malfunctions can be predicted before they occur, down time can be greatly reduced. Standard tests are available which increase the possibility of predicting probable failing areas. In order that these maintenance tests be utilized to their fullest capacity, they must be run periodically. Using them only for localizing a problem or distinguishing between hardware and software faults wastes much of their potential. Scheduled down time required for periodic maintenance routines might occasionally appear excessive, but it is far preferable to unscheduled down time.

MARGINAL TESTS

Marginal power tests are designed to locate points of weakness, areas which will probably become defective soon. These potential breakdown points are usually caused by deterioration of components. Voltage and heat problems are the most common causes of weakened components. The three basic marginal checks are lowering voltage levels, raising voltage levels, and varying clock frequencies. Varying clock frequencies can help detect deficient rise time and circuit response. Checks of the power supplies for nominal values and high ripple voltages should be included when voltage swing tests are made. Detailed marginal testing should be done at least weekly.

CHAPTER 2
CORRECTIVE MAINTENANCE

LOCATING FAILING EQUIPMENT

There are countless possible and probable error sources in a computer system. When a fault does occur in a system, the first step is to eliminate as many possibilities as practical in the least amount of time in order to return the system to operational status.

It is important to isolate the trouble to a general area as quickly as possible. A valuable initial contact is the system operator. He should be encouraged to save samples of the output which indicate a malfunction and its symptoms. When a malfunction is indicated, the operator should be consulted for information regarding the nature of the symptom, the equipment configuration, etc. The maintenance man should then attempt to reproduce the error by approximating the original conditions.

If after several attempts the error cannot be reproduced, it indicates an operator error, a marginal fault, or an intermittent failure. In any case, there is little to do but return the system to operation and see if the trouble will recur. If the error is reproduced, the maintenance man can begin to eliminate areas of suspicion.

The type of error will dictate the exact procedure to be followed. Often the status response from the device in use will indicate the type of error and give some idea of the source. The system output (printed copy, punched cards or punched paper tape) may also give an indication of the trouble area. If the problem is isolated to a cable, crosspoint or an Input/Output source, substitution of the malfunctioning component solves the problem.

If the problem is isolated to a composite hardware device (e.g., computer, Input/Output device, controller or adaptor), the maintenance man must then determine the exact nature of the failure. This involves operation of the computer console to produce a failure with the simplest possible program.

There are three categories of trouble that warrant attention at this point:

1) Electrical failures
2) Mechanical failures
3) Out-of-tolerence conditions. This type could be related to the above two but indicates the necessity of adjustment rather than replacement.

Once the problem can be reproduced in the simplest form, it is necessary to visually inspect moving parts and/or electrical pulses to locate the failing condition. It may be necessary to consult an operation manual for the particular system components to find and correct the error-causing condition.

Much time can be wasted in troubleshooting a system if the problem is not approached in a manner designed to quickly eliminate possibilities. As a maintenance man gains experience with the equipment, he will find his own shortcuts for arriving at the ultimate solution.

LOCATING FAILING COMPONENTS

Once the fault has been isolated to a specific device or a restricted group of devices, it must be further analyzed to determine which of the numerous components are failing. In many cases, the maintenance man will have used a looping program (either on tape or punched directly into the computer) to locate the general failing area. If he has not used a loop as yet, he will probably use one to reproduce the failure. The failure must be repeated continuously to allow detailed observation. If the failure cannot be reproduced, the fault is considered intermittent. (Intermittent failures are discussed later in this section.) Assuming that the failure can be repeated, the first systematic step toward pin-pointing the defective component can be taken.

A short looping program identifies and holds the failing function. If the logic diagrams are referenced, keeping in mind the failing function, the circuit which contains the fault and its associated terms can be identified. When the failing circuit has been located, the maintenance man should set up his oscilloscope and begin testing for the failing term. Normally, scope probing will begin near the end of the circuit at a point where a known condition should exist and progress back toward the origin of the pulse.

Even though the failure has been isolated to a few terms, it is not always evident which term is at fault (especially if the problem seems to be associated with an output). By referencing the file of equations, the maintenance man can gain additional information that is useful for further isolating the fault. Because the logic diagrams omit multiple outputs, the file of equations fills a gap in the troubleshooter's reference aids. Complex AND gates can be handled with relative ease through use of the file of equations.

If the fault seems to be associated with timing, the timing diagrams can be used to determine what should be happening. Timing diagrams can be applied almost directly to the oscilloscope if the scope sync is referenced properly. After utilizing the timing diagrams, maintenance personnel can use the timing chain more efficiently and more effectively.

If the fault can be isolated to a failing card by either scoping or exchanging cards, the faulty card should be replaced. When the suspected term is a flip-flop, both sides must be checked. The most common indication of faulty cards is a constant "1" or a constant "0" output.

Assuming that a card is not at fault, the maintenance man should check for wiring and cable errors. Wire tabs, cable tabs, and chassis maps are extremely useful for checking possible faults in these areas. In some instances, defective wires and cable connectors must be replaced, but more commonly, stabilizing them by either soldering or tightening will solve the problem. Recent Engineering Change Orders can occasionally affect the operation of a previously run program. This can be checked in the wire tabs and diagrams.

After the suspected fault has been located and corrective action taken, the failing circuit should be rechecked. It is not unusual to find and remedy a fault without solving the operating problem. This can be caused by either a multiple failure or the existence of a faulty card which had not been recently activated.

The maintenance log should give a thorough description of the type of failure, the means of locating it, and the method of correction. Problems which have been solved are valuable sources for future reference; those which have not been solved must be properly logged in order to prevent duplication of unsuccessful diagnostic effort by other personnel. A detailed description of the configuration of the equipment at the time of the failure should be included in the log.

The standard procedure of punching a looping program into the computer in order to reproduce the fault is not always successful. If the failing program has been thoroughly checked, and the fault still cannot be repeated, the probable conclusion is that an intermittent fault has occurred. Intermittent faults are the worst problem faced by maintenance personnel. They may recur frequently or seldom, and they may originate from a variety of causes. It is often difficult to isolate an intermittent fault

beyond a large portion of the entire system. Even after it has been traced to a small group of equipment where a number of tests can be tried, location of the fault is still not certain.

The first step in solving an intermittent failure is obtaining as much information as possible about it. The short period of error indication does not usually furnish enough facts to follow. The operator and programmer can often prove to be very valuable assets by describing how, when, and under what circumstances the fault occurred. Discussing the problem with other maintenance men and referencing the maintenance log will provide additional information if it has occurred in the past.

There are many possible intermittent failures, but space restricts this discussion to some of the most common. These might be classified as temperature, continuity, marginal pulse, and inter-device errors.

Intermittent temperature faults are usually caused by cards which have been weakened by some previous voltage or heat problem. The reason that they are so difficult to locate is that they occur only at certain temperatures under certain running conditions. When the program stops, the temperature varies, making duplication difficult. If an error appears or disappears when a chassis is opened, the fault is probably temperature linked. Utilization of a portable blower, such as a hair dryer, which can be used to vary temperatures in a specific area is a great help in locating this type. Usually the best solution to the problem is changing cards in the failing area until the fault is eliminated. Defective air conditioning equipment can also be responsible for temperature-caused intermittent failures.

Intermittent continuity errors are often caused by loose wires, loose cables, defective cable connectors, or defective card components. They can be detected by shock testing individual cards and agitating wires and cables while the device is running. Stabilizing loose connections is the normal corrective action.

Intermittent marginal pulse errors can usually be traced to slow or weak cards, timing faults, or incorrect power supply voltages. They can be detected by checking timing and comparing power supply voltages against equipment standards. Adjustment of the faulty equipment normally solves these problems.

Intermittent inter-device errors are often caused by loose or defective cable connectors, line drivers, or parity generators. They can be detected by agitating cable connectors, testing L and M cards, or checking the parity generators. Standard corrective procedures, replacing faulty cards and stabilizing cables should eliminate the problems.

It is important to accumulate and preserve as many facts as possible about intermittent failures. The first time a problem occurs, write down all information available; such as the program, register contents, and switch settings. Note what card decks are being used, which tape units are involved, and all other peripheral gear used. If the failure occurs again, with this additional information available, failing patterns may be recognized. The same tape unit may be in use at the time of the failure, or some other similarities might be recognized.

There are no definite solutions to the intermittent problem. At best, the above may be useful hints. The only certain preparations for solving intermittent problems are experience and a thorough knowledge of the equipment.

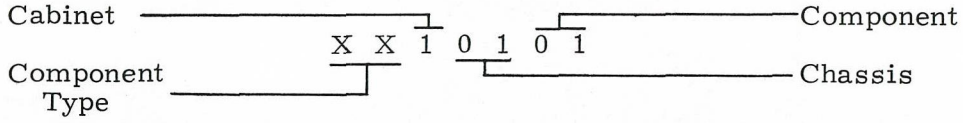
RECOMMENDED MAINTENANCE AND TEST EQUIPMENT

In addition to the ordinary hand tools commonly employed in electrical and mechanical maintenance, the following items or their equivalents are recommended for proper servicing of the system.

1) VOM Multimeter (T riplett 630A)
2) Digital VTVM (Hewlett-Packard HP405BR)
3) Oscilloscope (Tektronix type 543A)
4) Attenuation Probe (Tektronix P6017)
5) Printed Circuit Extender (CDC 6198)
6) Crimping Tool for terminal taper pins (AMP #48698)
7) Insertion Tool for terminal taper pins (AMP #380306-2)

COMPUTER IDENTIFICATION NUMBERING SYSTEM

A coordinate numbering system is used to locate and identify all items within the computer. Its purpose is ease of reference, and may be used to advantage by maintenance personnel to quickly determine the exact location of components. This method of identification numbering is depicted as follows:

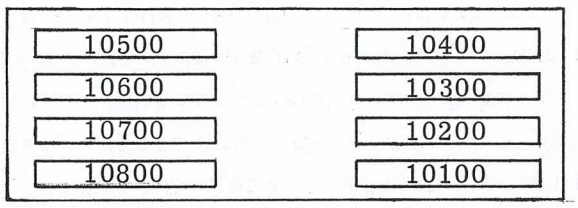


CHASSIS NUMBERING

Cabinet	Chassis	Component
0	0 0	0 0

10000 Cabinet (main computer)

The 8 chassis of the main computer, as viewed from the top, are numbered as illustrated at the right. Fuses for each chassis are considered as mounted on the cabinet rather than the chassis.



Front of Main Cabinet

20000 Cabinet (console)

Relay chassis	20100	Paper tape switch panel	20500
Relay chassis	20200	Electric typewriter	20600
Connector panel	20300	Paper tape reader	20700
Control panel (switches and indicators)	20400	Paper tape punch	20800

Transformers, E-strips and the loudspeaker are considered as mounted on a cabinet rather than a chassis.

CONNECTORS ON A STANDARD CHASSIS

The method for numbering each group of two cable connectors on the periphery of a standard chassis is shown in figure 2-1. Note: A third connector mounted on a side bracket is used at rows requiring additional cabling.

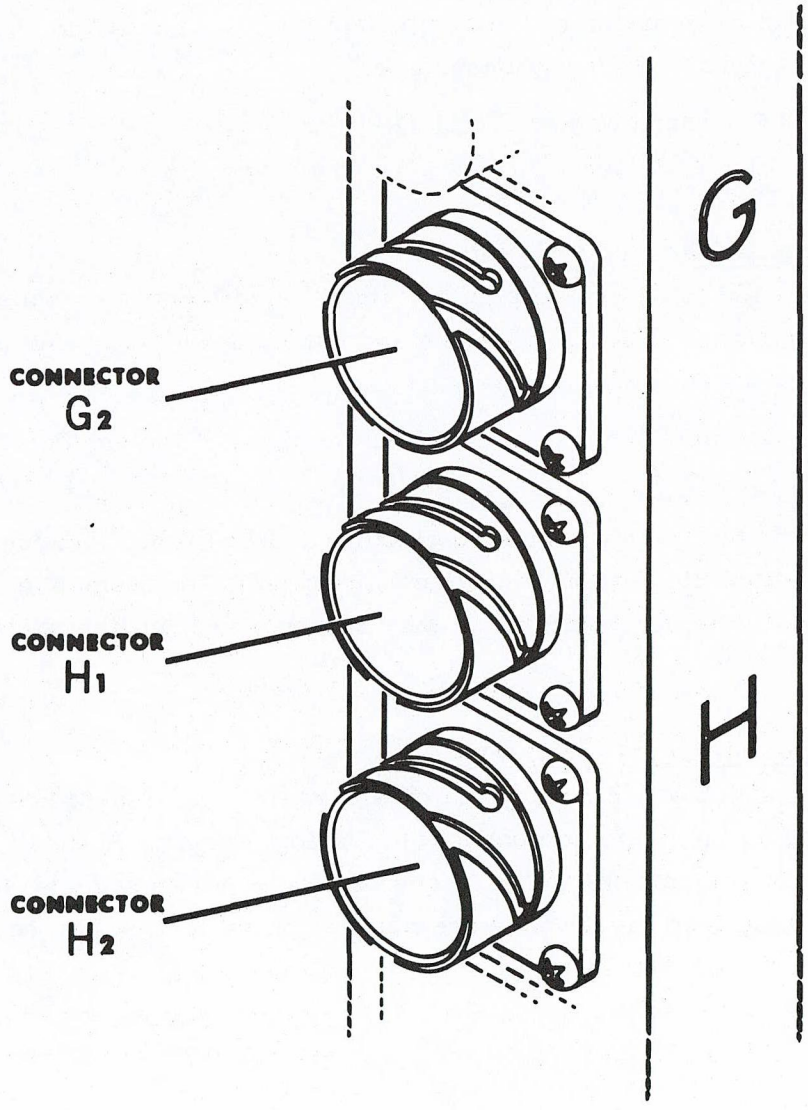


Figure 2-1. Cable Connector Identification

COMPONENT NUMBERING

Basic component numbering format:

Component Type	Cabinet	Chassis	Component
X X	0	0 0	0 0

Components on a Standard Chassis

Components on a standard chassis are numbered consecutively on the unit schematic diagram. The alphabetic designation of the component type is prefixed to the component identification number. Alphabetic designations are:

T - transformer	Q - transistor
CR - rectifier	R - resistor

Printed Circuit Cards on a Standard Chassis

The coordinate system used to designate printed circuit card locations on a standard chassis is illustrated in figure 2-2. The letters and numbers which appear on the chassis are combined in the following format:

Chassis	Ordinate (row)	Abscissa (column)	Test Point
0 0	X	0 0	X

Cabinet numbers are omitted from the printed circuit card locations because equations and card placement are individual to each cabinet. Test point locations are identified by letter (A - top, B - middle, C - bottom) as they are viewed from the wiring side of the card.

Components in a Cabinet but not on a Chassis

All components located within a cabinet but not on a chassis (e.g. fuses) are numbered consecutively according to the basic component numbering format. A special case exists in the 10000 (main computer) cabinet where fuses are numbered with respect to the chassis they protect even though they are not physically located on the chassis.

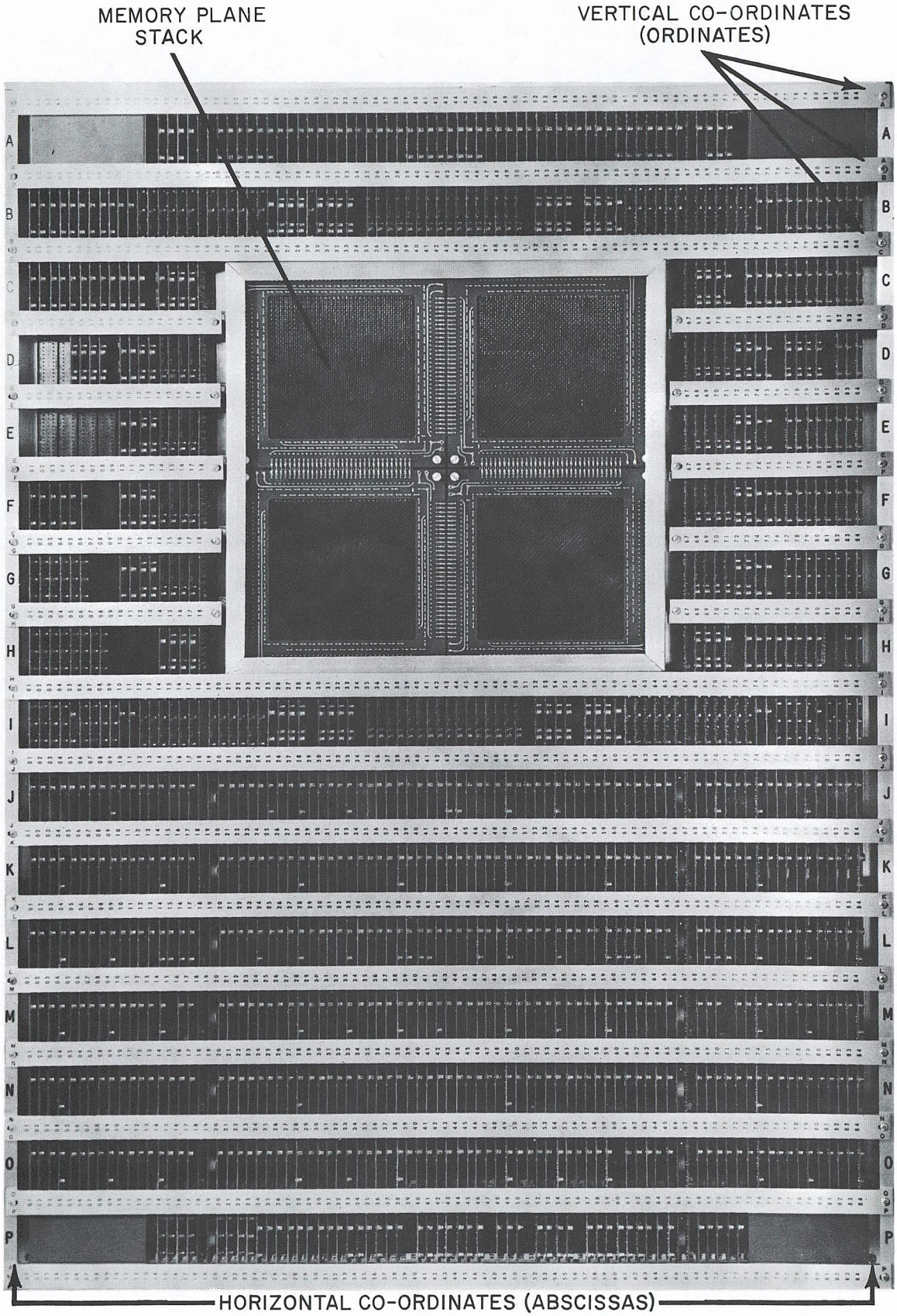


Figure 2-2. Card Side of a Typical Chassis

CHAPTER 3

DIAGNOSTIC MAINTENANCE

Diagnosis of failure symptoms and location of their causes is one of the chief maintenance activities. Actual correction of a failure usually consists of the simple replacement of a card. The procedures of diagnostic maintenance are dictated by the prominence of logical structure in the computer and the variety of possible causes of initial symptoms. Analysis of symptoms, based on a thorough understanding of computer logic, is required.

TEST PROGRAMS

The functioning of a given part of the computer is checked by a test program; execution of the program causes operations to be performed in the part under test. The results of the operations are checked to determine if they are proper; an improper result produces one of several indications of a malfunction.

The test programs are available in a separate packet. Some of the programs are briefly described in the following paragraphs.

1604-A COMMAND TEST

Description

The 1604-A Command test checks the operation of internal computer logic. Because, at the start, the program must use untested instructions to perform the checks, the test is constructed as an inverted pyramid. The apex, or starting point, involves only the most basic instructions. Also, the test uses a system of double checking before it passes from one instruction to the next.

Although it is impossible to test every bit configuration in each instruction or even to devise a list of the most difficult constants, the command test approaches this goal by using a very short set of constants in conjunction with unlimited random numbers. Unlimited in this sense means that each complete cycle will generate a new set of numbers. During one pass through the test the program generates and tests 5000_8 random numbers and repeats the test.

Since the command test is based on random numbers, it is impossible to have a stored set of answers. Instead, each part of the test simulates the particular instruction which is being checked.

If an error is ignored in the early part of the test, the error will probably occur again, only this time a false diagnosis will result.

1604-A MEMORY TEST

Description

General Description And Philosophy Of Test:

The 1604-A Memory test is a comprehensive examination of 1604-A Storage circuits. It checks the ability of these circuits to operate with varied bit configurations and under backgrounds of induced crosstalk and inhibit-driver noise. The comprehensive memory test program consists of 11 routines which are described below.

Description Of Individual Routines:

Test A (Hold All Zeros Test): Locations 4046 - 4047. This routine checks whether the computer can hold and restore zeros. Zeros are written into the memory to be tested and then are read out and checked. This read and check is repeated four times. Total time: five seconds.

Test B (Hold All Ones Test): Locations 4072 - 4115. This routine checks whether the computer can hold and restore ones. This test is the same as Test A, except that ones are written into the memory being tested. Total time: five seconds.

Test C (S-Register Test): Locations 4116 - 4233. This routine checks the address register. First each memory location being tested is loaded with its address and this is then checked. Next each location is loaded with the complement of its address and checked. Total time: five seconds.

Test D (Inhibit Sensitivity Test): Locations 4234 - 4361. This routine checks the ability of each core of each memory location to retain the "1" magnetic state after being subjected to 100, 000 (octal) repeated half magnitude inhibit pulses. Total time: five seconds.

Test E (Partial Write Test): Locations 4362 - 4473. This routine checks the partial write upper and the partial write lower. The memory being tested is first cleared and then 77777 is written into the upper address portion of each word. This is then read and checked. Next the memory being tested is loaded with negative zero and then 00000 is written into the upper address portion of each word. This is then read and checked. The test is then repeated for the lower address portions. Total time: five seconds.

Test F (Hold Worst Pattern): Locations 4474 - 5057. This routine checks the ability of the memory being tested to hold the worst pattern under the worst possible noise conditions. The worst pattern is loaded into the memory being tested. Indirect addressing then is used to butt a memory cycle involving a location of zeros with a cycle involving a read to determine whether the inhibit drive from the first cycle will carry over as noise in the second cycle. Then the entire test is repeated using the complement of the worst pattern instead of the worst pattern. Total time: 35 seconds.

Test G (Circulate Worst Pattern): Locations 5060 - 5243. This routine checks the ability of the sense amplifiers to distinguish a "0" from a "1" under varying noise conditions. The worst pattern is written into the memory being tested and is then circulated with the memory being read and checked after each circulation. After 15 circulations, the memory is restored back to the original worst pattern position. Total time: 45 seconds.

Test H (Crosstalk with Worst Pattern): Locations 5244 - 5273. This routine checks for both crosstalk and also the ability to hold the worst pattern in the presence of the worst possible noise conditions. Crosstalk is the result of inductive and capacitive pickup between wiring associated with the individual memory planes. The worst pattern is written into the memory being tested except for the complement of the worst pattern in one place. The memory is then read and checked. The complemented plane is then shifted by one, and the memory is read and checked. This is repeated until the complement plane has been shifted through the 48 memory planes. Then the entire test is repeated with the complement of the worst pattern in 47 planes and the worst pattern in a single plane. Total time: 100 seconds.

Test I (Indirect Addressing in the Presence of the Worst Pattern): Locations 5274 - 5655. The worst pattern is written into the upper 30 planes in the octant being tested. Then indirect addressing is used to reference each location in the octant, the memory cycles butting each other. The upper 30 planes are then checked. This is repeated ten times for each octant, and then the entire test is repeated replacing the worst pattern with its complement. Total time: 50 seconds.

Test J (Second Hold Worst Pattern Test): Locations 5656 - 5717. This test is a regular hold worst pattern test. The worst pattern is first written into the memory being tested. Each word is complemented and recomplemented and then the memory is checked. This is repeated ten times and then the entire test is repeated with the complement of the worst pattern. Total time: 30 seconds.

Test K (Diverter Test): Locations 5722 - 6027. This test checks the proper function of diverters during periods of rapid selection and deselection of these diverters.

Each of these routines makes use of return jump subroutines in locations 6066 - 6300.

The error subroutine in locations 6300 - 7227 determines the failing test, location, bits and provides for the buffering out of this information. This subroutine also corrects the failure in memory and sets up the console error display.

1604-A CONSOLE PUNCH AND READER TEST

Description

The console punch test checks the ability of the paper tape punch to punch 300 octal 1604-A words which have ripple pattern data. The number of words may be changed by setting Stop key 2 before starting the test. When the program stops on key 2, set Index register 6 to the desired number of words. Half of the tape is punched in Assembly and half in Character mode, both halves being identical in form.

This "test" tape then serves as input to the reader for the console reader test. The tape is read in whatever mode the operator places the Reader Mode switch. The program compares the input received by the reader with the ripple word pattern that it originally gave the punch. Upon completion of a read, the operator may reload the tape in the reader and re-read the tape in the same or in a different mode.

Setting Jump 1 causes the tape to be punched and/or read with a delay.

1604-A CONSOLE TYPEWRITER TEST

Description

This test activates typewriter output using upper and lower case and using Assembly and Character mode. The program then performs a space and backspace check and an interrupt and no interrupt on carriage return check. After each part of the Typewriter test the program ascertains if (1) the command was executed, and (2) if the sense response was correct.

The operator should examine the typeout to ascertain if the typewriter correctly performs the tasks assigned. The typewriter will type five lines:

Typeout	
Line 1	All characters, Character mode, lower case.
Line 2	All characters, Character mode, upper case.
Line 3	All letters, Assembly mode, upper case and lower case.
Line 4	All characters except letters, Assembly mode, upper and lower case.
Line 5	Space and backspace test.

DIAGNOSIS FROM CONSOLE

The console with its display of register contents, background lights and operating controls provides for the first level of diagnosis. A test program reveals the presence of a malfunction and the general area of computer logic causing it. For a description of the operating controls and background indicators, see Customer Engineering Instruction Manual, Part 1, Theory of Operation.

Suppose, for example, the original symptom of the malfunction was an improper result for instruction 14, Add. Since there are several possible causes of such a malfunction, the first step is to eliminate some of these possibilities. The basic procedure at the console is to execute in the Step mode several of the other instructions (11, Increase A; 45, Add Logical, etc.) which involve the adding operation.

After stepping through each of these instructions, the actual result displayed in A is compared with the correct result. If instruction 11 also fails, the malfunction must be in an area common to 11 and 14. Thus certain of the potential causes have been eliminated as possibilities. If, on the other hand, instruction 11 does not fail, the malfunction must be in an area not common to 11 and 14. This also eliminates certain

other potential causes as possibilities. This procedure is continued, using more instructions, until the number of possibilities is greatly reduced. At this point, the methods of the next section can be employed for complete identification of the cause of the malfunction.

LOGICAL CIRCUIT MAINTENANCE

After console diagnosis has indicated the circuits which may be causing the malfunction, the operation of these circuits is examined by means of an oscilloscope.

In some cases, observation of circuits in a static condition is sufficient; however examination of dynamic circuit conditions is often required. This is done by repeated execution of an instruction that uses the circuit. The upper position of the Storage Mode switch provides a convenient way of making such repetitions.

Information relevant to localizing the cause to a group of circuits and then to an individual circuit is contained in:

1) the command timing charts (Chapter 4 of this manual)
2) 1604-A Customer Engineering Diagrams Manual

The jack location and test point information required in taking waveforms for each circuit is provided by logic diagrams.

The operation of a circuit card is examined by means of waveforms taken at its test points. The test points are on card output. Since the cards are basically inverters, waveforms are the inversion of the card inputs. The common ground connection for the oscilloscope is made at the outer chassis edge. A synchronizing signal for the oscilloscope can be obtained from the test point of another circuit. Typically the synchronizing source is chosen to produce a signal just in advance of the time when a circuit is to be examined.

Occasionally it is necessary to look at signals on the individual pins of a card. This is done by removing the adjacent bars which hold the cards in position, removing the card, inserting the card extender, and plugging the card into the extender. On the extender, the pins of the card under test are easily accessible.

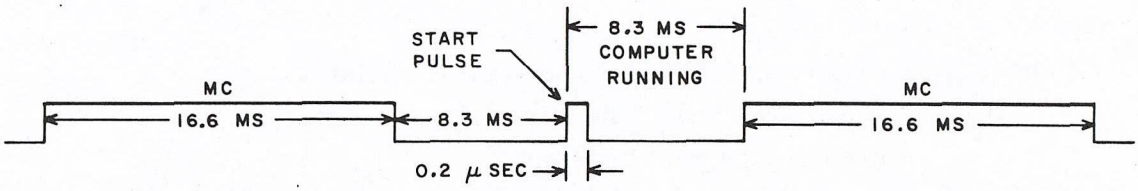
TEST MODE

There are some situations for which the simple repetition of an instruction does not yield satisfactory dynamic waveforms. Examples of such situations are:

1) deep end - a sequence fails to exit
2) clean start is required; that is, observations are to be made after master clear.

The Test mode is established by simultaneously pressing the Clear switch and raising the Start/Step switch. Raising the Clear switch (external master clear) terminates the Test mode.

In the Test mode, the 60-cycle line frequency is employed as a low-speed oscillator to produce alternate master clears and start pulses (see below). During one cycle (16.6 ms) the internal master clear is held on. The following cycle produces a start pulse 8.3 milliseconds after the master clear drops. The computer is allowed to run until the next master clear (8.3 milliseconds later).



STORAGE WAVEFORMS

The Memory Test describes techniques for determining whether there is a malfunction in the operation of storage. These techniques also reveal the address and bit of the malfunction. Further isolation to a specific card is accomplished by means of waveform analysis. Observed waveforms from pertinent cards are compared with normal waveforms from cards of the same type.

PREVENTIVE MAINTENANCE SCHEDULE

HOURLY

PT Reader: After 300 hours on line operation apply oil (weight 20), a drop or two, to the bronze input gear and to the differential gears. Wipe off excess oil. Do not allow any oil to get on the surfaces of the clutch drum, tape drive drum, or tape deck.

DAILY

Janitorial services: clean computer room, especially console top, tape baskets, and floors.

Clear

PT Reader: remove tape setting clip to clear photocell block

PT Punch: chad and paper lint

Lubricate

PT Punch: tape reel bearings if required

Operating Checks

Run Test programs

PT Punch: registration of punches

WEEKLY

Clean

Air filters in cabinets

Lubricate

PT Punch:
- Toggle arm shaft, saturate felt washers
- Punch bail shaft, saturate felt washers
- grease tape reel bearings

Typewriter:
- heavy gear grease on points where metal is moved on metal
- light oil on speings and pivot points

Operating Checks

Perform marginal Power tests

MONTHLY

Clean

Typewriter: keys, platen and actuator solenoids

PT Reader: clean all surfaces above console top

Lubricate

PT Punch: each end of motor, feed wheel ratchet and punch block

Operating Checks

Typewriter:	worn ribbon
PT Punch:	punch for wearing
PT Reader:	check festoon lamp
All Cabinets:	check blowers

SEMIANNUALLY AND ANNUALLY

MG Control Cabinet and Relays: clean and check semiannually.

Typewriter:	clean and lubricate semiannually
MG Bearing:	replace annually

CABLING INFORMATION

The identification of Input/Output cables and the information carried on their lines is treated in the following tables. Table 3-1 lists the labels on the individual cables of the four groups. Each label indicates the function of the cable in the group by a prefix letter, i.e., ABC input, DEF output. The expression following the slash gives the computer connector for the cable. Table 3-2 lists the information on each line of the six cables in a group.

Other cables in the computer system such as those connecting chassis within the main cabinet or those connecting the main cabinet and console are labeled as required.

TABLE 3-1. CABLE IDENTIFICATION

Cable Group 1	A/7J2 CH 1	Cable Group 3	A/7M2 CH 5
	B/7K1 CH 1		B/7N1 CH 5
	C/7K2 CH 1		C/7N2 CH 5
	D/8J2 CH 2		D/8M2 CH 6
	E/8K1 CH 2		E/8N1 CH 6
	F/8K2 CH 2		F/8N2 CH 6
Cable Group 2	A/7L1 CH 3	Cable Group 4	A/7O1 CH 7
	B/7L2 CH 3		B/7O2 CH 7
	C/7M1 CH 3		C/7P1 CH 7
	D/8L1 CH 4		D/8O1 CH 7
	E/8L2 CH 4		E/8O2 CH 7
	F/8M1 CH 4		F/8P1 CH 7

Group 1	
	channel 1 - buffer input
	channel 2 - buffer output
Group 2	
	channel 3 - buffer input
	channel 4 - buffer output
Group 3	
	channel 5 - buffer input
	channel 6 - buffer output
Group 4	
	channel 7 - transfer input and output

Note: See page 80, Customer Engineering Diagrams, for detailed cable connections and pin assignments.

TABLE 3-2. CONNECTOR PIN NUMBER ASSIGNMENTS

Pin No.	Input Buffer or Transfer Channel			Output Buffer or Transfer Channel		
	Cable A	Cable B	Cable C	Cable D	Cable E	Cable F
A	bit 47	bit 24	bit 01	bit 00	bit 23	bit 46
B	46	23	00	01	24	47
C	45	22	Input Ready[3]	02	25	Output Ready[3]
D	44	21	Input Resume[3]	03	26	Output Resume[3]
E	43	20	Input Buffer Active[2]	04	27	Interrupt
F	42	19	External Master Clear	05	28	Input Function Ready[2]
H	41	18	Interrupt	06	29	Input Sense Ready[2]
J	40	17	Not Used	07	30	Output Function Ready
K	39	16	↓	08	31	Output Sense Ready
L	38	16	↓	09	32	Sense Response
M	37	14	↓	10	33	Output Buffer Active[1]
N	36	13	↓	11	34	Function Bit 00
P	35	12	↓	12	35	↓ 01
R	34	11	↓	13	36	↓ 02
S	33	10	↓	14	37	↓ 03
T	32	09	↓	15	38	↓ 04
U	31	08	↓	16	39	↓ 05
V	30	07	↓	17	40	↓ 06
W	29	06	↓	18	41	↓ 07
X	28	05	↓	19	42	↓ 08
Y	27	04	↓	20	43	↓ 09
Z	26	03	↓	21	44	↓ 10
a	25	02	↓	22	45	↓ 11
b	gnd	gnd	gnd	gnd	gnd	↓ gnd

1) Output Buffer Active signal is designated Output Transfer Active on Transfer channel.

2) Buffer cable only, unused in transfer.

3) Input/Output Ready/Resume signals designated Input/Output Transfer Ready/ Resume on Transfer channel.

CHAPTER 4

COMMAND TIMING CHARTS

INTRODUCTION

The computer successively executes instructions from internally-stored programs by a sequence of commands. A command accomplishes one act, for example, transmitting data from one register to another or clearing a register. The operation code of the instruction to be executed selects one of the control sequences. This sequence is then initiated to generate the appropriate commands as determined by the operation code.

All commands involved in the execution of an instruction are listed in the order of occurrence in the command timing charts.* The instruction sequence used to generate the commands is specified under the heading "Sequence".

Entries in the time column indicate the phase time (0.2 usec in duration) at which the associated Command signal occurs. These phase times are related to the phase times at which the sequence is initiated. Initiate time is always considered as time 00. For Command signals rising from control flip-flops (FFs) rather than control delays, the entry in the time column indicates the last time the signal is clocked. Usually this is the time when the Control flip-flop is set. The resulting command does not actually take effect until approximately two phase times later.

The three entries given under Execution Times take account of the time for three cases of instruction use. Variations in execution time are caused by such factors as:

1) Upper or lower position in instruction word
2) Consecutive references to the same storage unit
3) Storage reference at end of preceding instruction

All three time entries are determined by averaging the times for a long list of the same instruction. Minimum time is an average of a list arranged so that the factors above have minimum values; maximum time is an average of a list in which these factors have maximum values; and average time applies to a list arranged for typical values of the factors.

Comments in the Remarks column describe the function of the command in the execution of the instruction.

*Commands generated but not pertinent to the execution of the instruction have been omitted from the charts.

GLOSSARY OF ABBREVIATIONS

A	Arithmetic register
ARD	Auxiliary reference designator
ARR	Auxiliary reference register
Adv Clk	Advance Clock
AQ	The double-length register comprosed of A and Q
B^b	The designated Index register
Buf	Buffer
Comp	Complement
CR	Common register
CCR	Common control register
Exp	Exponent
FF	Flip-flop
IMR	Interrupt masked register
Init	Initiate
Inst	Instruction
Int	Interrupt
I^2I^3	The inverter rank preceding R
I^5I^6	The inverter rank between the storage circuits and the arithmetic and control circuits
I^7I^8	The inverter rank control and common control register
LQX	The logical (bit-by-bit) product of Q and X
m	The base execution address
M	The modified execution address
MIR	Masked interrupt register
Neg	Negative
P	Program address register
Part	Partial
Pos	Positive
Q	Auxiliary arithmetic register
R	Address buffer register
Red	Reduce
SR	Sign record
U^1	Program control register
U^2	Auxiliary program control register
X	Exchange register
Z	Storage restoration register
→	(arrow) transmit the contents
( )	(parentheses) contents of a register
subscript f	Final contents of a register
subscript i	Initial contents of a register
subscript L	Lower half of a register
subscript LA	The address portion (lowest 15 bits) of the lower instruction
subscript U	Upper half of a register
subscript UA	The address portion (lowest 15 bits) of the upper instruction

TIMING CHARTS

CODE	INSTRUCTION	FUNCTION
RNI	Read Next Instruction	Prepare computer for receipt of instruction word from storage and execution of next instruction.

SEQUENCE: Read Next Instruction

EXECUTION TIME:

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V033	Adv $P^2 \rightarrow P^1$	Full Exit	Add 1 to (P_i)
	V090	Set K600/601	Full Exit	Set Initiate Storage FF. Reference address $P_i + 1$
	V090	Set K200/201	Full Exit	Set Wait Storage FF
08	V094	Set K060/061	Full Exit	Establish mode for concluding the instruction
08	V094	Clear K060/061	Half Exit	
09	V094	Clear U	Full Exit	
09	V094	Clear U^1_u	Half Exit	Set up current instruction in U^1_u
10	V012	$B^b \rightarrow I^2\ I^3$		
11	V155	Set K050/051		Set Stop II FF. Step or stop or break-point
11	V154	$I^5\ I^6\ I^7 \rightarrow U^1$	Full Exit	
11	V154	$U^1_l \rightarrow U^1_u$	Half Exit	
11	V154	Clear R^1		Prepare R^1 for receipt of (B^b)
11	V155	Clear K064/065	Interrupt Complete	P = 00007-17 terminates the interrupt instruction routine
12		Wait Step		RNI stops to await subsequent start or step pulse
14	V098	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Transfer $B^b \rightarrow R^1$
14	V099	$U^1 \rightarrow U^2$		
15	V834	Clear X^1		Prepare X register to receive new data

CODE	INSTRUCTION	FUNCTION
01 ARS	A Right Shift	Shift A Right K Places

SEQUENCE: Zero Address (H^{2--} V^{2--})

4.0 usec min. (Lower Inst.) 2.8 usec + .4 usec/shift avg., 54.5 usec max. 5.6 usec min. (Upper Inst.)

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	I^2 $I^3 \rightarrow R^1$	$b \neq 0$	Prepare to form K - RNI
	V099	$U^1 \rightarrow U^2$		
04	V203	Add $R^1 \rightarrow U^2$		Add (B^b) to k to form K
06	V205	$U^2 \rightarrow R^2$		Transfer shift count to R
	V206	Set K410/411		Set the R = 0 FF
07	V206	$R^2 \rightarrow R^1$		Set R^1 equal to R^2 (Equalize the shift count in R)
	N991	Set K414/415		Set the R ≠ 0 FF slaves
	N931	Set K412/413		
	N570	Clear K410/411	$R^2 = 0$	Clear the R ≠ 0 FF if no shifting is to be performed
09	V209	Initiate Shift		H220, V220 starts shift control
		Set K234/235	$K > 127_{10}$	Set Shift Fault FF if shift count is greater than 127_{10}
	N931	Clear K412/413	$R^2 = 0$ at time 08	Clear R ≠ 0 slaves of $R^2 = 0$
	N991	Clear K414/415		
10	V220	Set K310/311	$R^2 \neq 0$ at time 08	Set A Right Shift FF
11	V221	Set K320/321		Set Shift Exit Control FF to enable exit as soon as shift operation is completed

TIME	TERM	COMMAND	CONDITION	REMARKS
12	N-70	Reduce $R^1 \rightarrow R^2$		Reduce Shift Count and Shift
	N-10 N-12	Shift A Right 1 place		Repeat until Shift Count = 0
	K321	Exit	R = 0	Exit when shifting is complete

CODE	INSTRUCTION	FUNCTION
02 QRS	Q Right Shift	Shift Q Right K Places

SEQUENCE: Zero Address (H^{2--} V^{2--})

EXECUTION TIME: 4.0 μsec min. (Lower Inst.)
5.6 μsec min. (Upper Inst.) 2.8 μsec + .4 μsec/shift avg., 54.4 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	I^2 $I^3 \rightarrow R^1$	b ≠ 0	Prepare to form K - RNI
	V099	$U^1 \rightarrow U^2$		
04	V203	Add $R^1 \rightarrow U^2$	b ≠ 0	Add (B^b) to k to form K
06	V205	$U^2 \rightarrow R^2$		Transfer shift count to R
	V206	Set K410/411		Set the R ≠ 0 FF
07	V206	$R^2 \rightarrow R^1$		Set R^1 equal to R^2
	N991	Set K414/415		Set the R ≠ 0 FF slaves
	N931	Set K412/413		
08	N570	Clear K410/411	$R^2 = 0$	Clear the R ≠ 0 FF if no shifting is to be performed
09	V209	Initiate Shift		H220, V220 starts shift control
		Set K234/235	K > 127_{10}	Set Shift fault FF if Shift count is greater than 127_{10}
09	N931	Clear K412/413	$R^2 = 0$ at time 08	Clear R ≠ 0 slaves if $R^2 = 0$
	N991	Clear K414/415		
10	V220	Set K312/313	$R^2 \neq 0$ at time 08	Set Q Right Shift FF
11	V221	Set K320/321		Set Shift Exit Control FF to enable exit as soon as the shift operation is completed
	N570	Exit	R = 0	

TIME	TERM	COMMAND	CONDITION	REMARKS
12	N-70	Reduce $R^1 \rightarrow R^2$	R = 0	
	N-20 N-22	Shift Q right 1 place		Reduce shift count and shift repeat until shift count = 0
	N321	Exit	R = 0	Exit when shifting is completed

CODE	INSTRUCTION	FUNCTION
03 LRS	AQ Right Shift	Right Shift (AQ) K places

SEQUENCE: Zero Address (H^{2--} V^{2--})

EXECUTION TIME: 4.0 μsec min. (Lower Inst.), 5.6 μsec min. (Upper Inst.) 2.8 μsec + .4 μsec/shift avg., 54.4 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	I^2 $I^3 \rightarrow R^1$	b ≠ 0	Prepare to form K - RNI
	V099	$U^1 \rightarrow U^2$		
04	V203	Add $R^1 \rightarrow U^2$	b ≠ 0	Add (B^b) to k to form K
06	V205	$U^2 \rightarrow R^2$		Transfer shift count to R
	V206	Set K410/411		Set the R ≠ 0 FF
07	V206	$R^2 \rightarrow R^1$		Set R^1 equal to R^2
	N931	Set K412/413		Set the R ≠ 0 FF slaves
	N991	Set K414/415		
08	N570	Clear K410/411	$R^2 = 0$	Clear the R ≠ 0 FF if no shifting is to be performed
09	V209	Initiate Shift		H220, V220 starts shift control
	V209	Set K234/235	$K > 127_{10}$	Set shift fault FF if shift count is greater than 127_{10}
	N931	Clear K412/413	$R^2 = 0$ at time 08	Clear R ≠ 0 slaves if $R^2 = 0$
	N991	Clear K414/415		
10	V220	Set K310/311	$R^2 \neq 0$ at time 08	Set A Right Shift FF
		Set K312/313		Set Q Right Shift FF
11	V221	Set K320/321		Set shift exit control FF to Enable exit as soon as the shift operation is completed

TIME	TERM	COMMAND	CONDITION	REMARKS
12	N-70	Reduce $R^1 \rightarrow R^2$		
	N-10	Shift A Right 1 Place		Reduce shift count and shift Repeat until shift count = 0
	N-12			
	N-20	Shift Q Right 1 Place		
	N-22			
	K321	Shift exit control	R=0	Exit when shifting is completed

CODE	INSTRUCTION	FUNCTION
04 ENQ	Enter Q	Transfer Y → Q, Extend sign Y

SEQUENCE: Zero Address (H^{2--}, V^{2--})

EXECUTION TIME: 2.8 μsec min., 3.0 μsec avg., 3.2 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	$I^2\ I^3 \rightarrow R^1$	b ≠ 0	Prepare to form Y - RNI
	V099	$U^1 \rightarrow U^2$		
03	V201	$A^2 \rightarrow Q^1$		Store (A_i) in Q^1 temporarily
	V203	Set K580/581		Set Clear A^1 FF
04	V203	Add $R^1 \rightarrow U^2$	b ≠ 0	Add (B^b) to y to form Y
06	V206	Clear K470/471		Clear F = 04 FF
		Clear K472/473		Clear F = 04, 10, 11 FF
		Clear K580/581		Clear Clear A FF
07	V205	$U^2 \rightarrow X^1$ with sign extension		Transfer Y to X^1
	V207	Set K472/473		Set F = 04, 10, 11 FF } These FF's condition later commands
		Set K470/471		Set F = 04 FF }
08	V206	$X^1 \rightarrow X^2$		Transfer Y to X^2
09	V207	Full Exit	R = 0	
13	V211	Add $X^2 \rightarrow A^1$		Transfer Y into A^1
14	V212	$Q^1 \rightarrow Q^2$		Transfer (A_i) to Q^2
15	V213	$A^2 \rightarrow Q^1$		Enter Y in Q^1
		$Q^2 \rightarrow A^1$		Restore (A_i) to A

CODE	INSTRUCTION	FUNCTION
05 ALS	(A) Left Shift	Left Shift (A) K places, left circular

SEQUENCE: Zero Address (H^{2--}, V^{2--})

EXECUTION TIME: 4.0 μsec min. (Lower Inst.)
5.6 μsec min. (Upper Inst.) 2.8 μsec + .4 μsec/shift avg., 54.4 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Prepare to form K - RNI
	V099	$U^1 \rightarrow U^2$		
04	V203	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to k to form K
06	V205	$U^2 \rightarrow R^2$		Transfer the shift count (K) to R
	V206	Set K410/411		Set R ≠ 0 FF. Used to control when the shift stops and determines if any shift is to be performed.
07	V206	$R^2 \rightarrow R^1$		Set R^1 equal to R^2
	N931	Set K412/413		Set the R ≠ 0 FF slave
	N991	Set K414/415		Prepare to clear K410/411 to stop shifting
08	N570	Clear K410/411	$R^2 = 0$	Clear the R ≠ 0 FF if no shifting is to be performed
09	V209	Set K234/235	$K > 127_{10}$	Set Shift Fault FF if shift count is greater than 127_{10}
09	V209	Initiate Shift		H^{220}, V^{220} starts shift control
09	N931	Clear K412/413	$R^2 = 0$ at time 08	Clear R ≠ 0 slaves if $R^2 = 0$
	N991	Clear K414/415		
10	V220	Set K314/315	$R^2 \neq 0$ at time 08	Set A Left Shift FF
11	V221	Set K320/321		Set shift exit control FF to enable exit as soon as the shift operation is completed

TIME	TERM	COMMAND	CONDITION	REMARKS
12	N-70	Reduce $R^1 \rightarrow R^2$		
	N-14	Shift A left 1 place		Reduce shift count and shift Repeat until shift count (K) = 0
	N-16			
	K321	Exit	R = 0	Exit when shifting is completed

CODE	INSTRUCTION	FUNCTION
06 QLS	(Q) Left Shift	Left Shift (Q) K places, left circular

SEQUENCE: Zero Address (H^{2--}, V^{2--})

EXECUTION TIME: 4.0 μsec min. (Lower Inst.)
5.6 μsec min. (Upper Inst.) 2.8 μsec + .4 μsec/shift avg., 54.4 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Prepare to form K - RNI
	V099	$U^1 \rightarrow U^2$		
04	V203	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to k to form K
06	V205	$U^2 \rightarrow R^2$		Transfer shift count to R
	V206	Set K410/411		Set the $R \neq 0$ FF. Determines if any shift is to be performed when the shift stops.
07	V206	$R^2 \rightarrow R^1$		Set R^1 equal to R^2
	N931	Set K412/413		Set the $R \neq 0$ FF slaves. Prepare to clear K410/411 to stop shifting.
	N991	Set K414/415		
08	N570	Clear K410/411	$R^2 = 0$	Clear the $R \neq 0$ FF if no shifting is to be performed
09	V209	Initiate Shift		H220, V220 starts shift control
		Set K234/235	$K > 127_{10}$	Set Shift Fault FF if shift count is greater than 127_{10}
09	N931	Clear K412/413	$R^2 = 0$ at time 08	Clear $R \neq 0$ slaves if $R^2 = 0$
	N991	Clear K414/415		
10	V220	Set K316/317	$R^2 \neq 0$ at time 08	Set Q Left Shift FF
11	V221	Set K320/321		Set Shift Exit Control FF to Enable exit as soon as the shift operation is completed

TIME	TERM	COMMAND	CONDITION	REMARKS
12	N-70	Reduce $R^1 \rightarrow R^2$		
	N-24 N-26	Shift Q Left 1 place		Reduce shift count and shift Repeat until shift count = 0
	K321	Exit	R = 0	Exit when shifting is completed

CODE	INSTRUCTION	FUNCTION
07 LLS	(AQ) Left Shift	Left Shift (AQ) K places, left circular

SEQUENCE: Zero Address (H^{2--}, V^{2--})

EXECUTION TIME: 4.0 μsec min. (Lower Inst.), 5.6 μsec min. (Upper Inst.) 2.8 μsec + .4 μsec/shift avg., 54.4 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Prepare to form K - RNI
	V099	$U^1 \rightarrow U^2$		
04	V203	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to k to form K
06	V205	$U^2 \rightarrow R^2$		Transfer shift count to R
	V206	Set K410/411		Set the $R \neq 0$ FF. Determines if any shift is to be performed when the shift stops.
07	V206	$R^2 \rightarrow R^1$		Set R^1 equal to R^2
	N931	Set K412/413		Set the $R \neq 0$ FF slaves. Prepare to clear K410/411 to stop shifting.
	N991	Set K414/415		
08	N570	Clear K410/411	$R^2 = 0$	Clear the $R \neq 0$ FF if no shifting is to be performed
09	V209	Initiate Shift		H220, V220 starts shift control
		Set K234/235	$K > 127_{10}$	Set Shift Fault FF if the shift count is greater than 127_{10}
09	N931	Clear K412/413	$R^2 = 0$ at time 08	Clear $R \neq 0$ slaves if $R^2 = 0$
	N991	Clear K414/415		
10	V220	Set K314/315	$R^2 \neq 0$ at time 08	Set A and Q Left Shift FF
		Set K316/317		
11	V221	Set K320/321		Set Shift Exit Control FF to Enable exit as soon as the shift operation is completed

TIME	TERM	COMMAND	CONDITION	REMARKS
12	N-70	Reduce $R^1 \rightarrow R^2$		
	N-14 N-16	Shift A Left 1 place		Reduce R and shift Repeat until shift count = 0
	N-24 N-26	Shift Q Left 1 place		
	K321	Exit	R = 0	Exit when shifting is completed

CODE	INSTRUCTION	FUNCTION
10 ENA	Enter A	Transfer Y to A^1, extend the sign of Y

SEQUENCE: Zero Address (H^{2--}, V^{2--})

EXECUTION TIME: 2.8 μsec min., 3.0 μsec avg., 3.2 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$ $I^2\ I^3 \rightarrow R^1$	b ≠ 0	Prepare to form Y - RNI
03	V203	Set K580/581		Set Clear A^1 FF
04	V203	Add $R^1 \rightarrow U$	b ≠ 0	Add (B^b) to y to form Y
06	V206	Clear K580/581		Clear Clear A^1 FF
07	V205	$U^2 \rightarrow X^1$ Ext.		Transfer (U^2) to X^1 and extend the sign
	V207	Set K472/473		Set F = 04, 10, 11. This FF conditions later commands by remembering what the function code was before the exit.
08	V206	$X^1 \rightarrow X^2$		Transfer Y to X^2
11	V209	Exit		
13	V211	Add $X^2 \rightarrow A^1$		Transfer Y to A^1

CODE	INSTRUCTION	FUNCTION
11 INA	Increase A	Add Y to (A), store the result in A

SEQUENCE: Zero Address (H^{2--}, V^{2--})

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		
			b ≠ 0	Prepare to form Y - RNI
	V871	$I^2\ I^3 \rightarrow R$		
04	V203	Add $R^1 \rightarrow U^2$	b ≠ 0	Add y to (B^b) to form Y
07	V205	$U^2 \rightarrow X^1$ with Ext.		Enter Y in X^1 and extend sign
	V207	Set K472/473		Set F = 04, 10, 11 FF. This FF conditions later commands by remembering what the Function code was before the exit.
08	V206	$X^1 \rightarrow X^2$		Transfer Y to X^2
11	V209	Half Exit		
		Full Exit		
13	V211	Add $X^2 \rightarrow A^1$	K473 is set	Add Y to A

CODE	INSTRUCTION	FUNCTION
12 LDA	Load A	Transfer (M) to A

SEQUENCE: Read Operand (H^{3--}, V^{3--})

4.8 usec min., 7.2 usec avg., 9.6 usec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$	b ≠ 0	Prepare to form M
	V871	$I^2\ I^3 \rightarrow R^1$		
01	V000	Clear X^1		Prepare to receive word from storage (RNI)
	V006	Set K520/521		Set Partial Add in A FF (RNI)
04	V302	Add $R^1 \rightarrow U^2$	b ≠ 0	Add (B^b) to m to form M
04	V304	Set K602/603		Initiate storage
05	V304	Clear R^1		
06	V306	Set K580/581		Clear A^1 FF
12	V310	Set K202/203		Set Wait Storage I FF
13	V073 V075	Storage Resume		Resume Sequence
14	V340	Clear K202/203		Clear Wait Storage I FF
15	V844	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (M) $\rightarrow X^1$
16	V821	$X^1 \rightarrow X^2$		Place (X^1) in X^2 for transfer to A
19	V370	Half Exit		
		Full Exit		
21	V345	Part. Add X^2 to A^1		Transfer (M) to A

CODE	INSTRUCTION	FUNCTION
13 LAC	Load A Complement (Negative A)	Transfer $(M)'$ to A

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099 V871	$U^1 \rightarrow U^2$ $I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Prepare to form M Prepare to receive word from storage (RNI)
01	V000	Clear X^1		
04	V302	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
	V856	Set K602/603		Set Initiate Storage FF
06	V306	Set K580/581		Set Clear A^1 FF
08	V308	Clear K580/581		Clear Clear A^1 FF
10	V310	Set K202/203		Set Wait Storage I FF
13	V073 V075	Storage Resume		
14	V340	Clear K202/203		Clear Wait Storage FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (M) $\rightarrow X^1$
16	V340	Comp. $X^1 \rightarrow X^2$		Complement (M)
19	V370	Half Exit		
		Full Exit		
21	V345	Part. Add X^2 to A^1		Transfer $(M)'$ to A

CODE	INSTRUCTION	FUNCTION
14 Add	Add to A	Add (A) and (M), store the sum in A

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Prepare to form M (RNI)
	V099	$U^1 \rightarrow U^2$		
01	V000	Clear X^1		Prepare to receive word from storage (RNI)
04	V302	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
	V303	Set K602/603		Set Initiate Storage FF
10	V310	Set K202/203		Set Wait Storage I FF
13	V073	Storage Resume		
	V075			
14	V340	Clear K202/203		Clear Wait Storage FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (M) to X^1
	V341	Clear K520/521		Clear Partial Add in A FF
16	V340	$X^1 \rightarrow X^2$		Transfer (M) to X^2
19	V344	Half Exit		
		Full Exit		
21	V345	Add $X^2 \rightarrow A^1$		Form [(A) + (M)] in A

CODE	INSTRUCTION	FUNCTION
15 SUB	Subtract from A	Subtract (M) from (A) and store the difference in A

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Prepare to form M (RNI)
	V099	$U^1 \rightarrow U^2$		
01	V000	Clear X^1		Prepare to receive word from storage (RNI)
04	V303	Set K602/603		Set Initiate Storage FF
	V302	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
10	V310	Set K202/203		Set Wait Storage I FF
13	V073 V075	Storage Resume		
14	V340	Clear K202/203		Clear Wait Storage I FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (M) $\rightarrow X^1$
	V341	Clear K520/521		Clear Partial Add A FF
16	V340	Comp. $X^1 \rightarrow X^2$		Form the complement of (M)
19	V344	Full Exit Half Exit		
21	V345	Add $X^2 \rightarrow A^1$		Add $(M)'$ to A to form $[(A)-(M)]$ in A

CODE	INSTRUCTION	FUNCTION
16 LDQ	Load Q	Transfer (M) to Q

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Prepare to form M (RNI)
	V099	$U^1 \rightarrow U^2$		
01	V000	Clear X^1		Prepare to receive word from storage (RNI)
04	V302	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
	V303	Set K602/603		Set Initiate Storage FF
06	V306	Set K580/581		Set Clear A^1 FF
07	V305	$A^2 \rightarrow Q^1$		Set (A) temporarily in Q
08	V308	Clear K580/581		Clear Clear A^1 FF
10	V310	Set K202/203		Set Wait Storage I FF
13	V073 V075	Storage Resume		
14	V340	Clear K202/203		Clear Wait Storage I FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (M) $\rightarrow X^1$
	V341	Set K480/481		Set f = 16, 17 FF. Remembers the instruction after the exit (conditions later commands).
16	V340	$X^1 \rightarrow X^2$		
19	V344	Full Exit	Exit FF Clear	
		Half Exit	Exit FF Set	
21	V345	Add $X^2 \rightarrow A^1$		Transfer (M) to A^1

TIME	TERM	COMMAND	CONDITION	REMARKS
22	V346	$Q^1 \rightarrow Q^2$	K480/481 is set	Place (A_i) in Q^2 for transfer back to A
23	V347	$A^2 \rightarrow Q^1$	K480/481 is set	Transfer (M) to Q^1
		$Q^2 \rightarrow A^1$	K480/481 is set	Restore (A_i) to A
27	V353	Clear K480/481		Clear f = 16, 17 FF

CODE	INSTRUCTION	FUNCTION
17 LQC	Load Q Complement (Negative Q)	Transfer the complement (M) to Q

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Prepare to form M (RNI)
	V099	$U^1 \rightarrow U^2$		
01	V000	Clear X^1		Prepare to receive word from storage (RNI)
04	V302	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
	V303	Set K602/603		Set Initiate Storage FF
06	V306	Set K580/581		Set Clear A^1. Prepare A^1 for receipt of (M)′
07	V307	Clear K070/071		
	V305	$A^2 \rightarrow Q^1$		Store (A) temporarily in Q
08	V308	Clear K580/581		Clear Clear A^1 FF
10	V310	Set K202/203		Set Wait Storage FF
13	V073 V075	Storage Resume		
14	V340	Clear K202/203		Clear Wait Storage I FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (M) to X
	V341	Set K480/481		Set F = 16, 17 FF. Remembers the instruction after the exit (conditions later commands).
16	V340	Comp. $X^1 \rightarrow X^2$		Form (M)′ in X^2 for transfer to A
19	V344	Exit		

TIME	TERM	COMMAND	CONDITION	REMARKS
21	V345	Part. Add $X^2 \rightarrow A^1$		Transfer $(M)'$ to A
22	V346	$Q^1 \rightarrow Q^2$	K480/481 is set	Place (A_i) in Q^2 for transfer back to A
23	V347	$A^2 \rightarrow Q^1$	K480/481 is set	Transfer $(M)'$ to Q
		$Q^2 \rightarrow A^1$	K480/481 is set	Restore (A_i) to A
27	V353	Clear K480/481		Clear f = 16, 17 FF

CODE	INSTRUCTION	FUNCTION
20 STA	Store A	Transfer (A) to M

SEQUENCE: Write Operand (H^{4--}, V^{4--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Prepare to form M
	V099	$U^1 \rightarrow U^2$		
01	V000	Clear X^1		Clear X^1 to receive word to be stored
04	V402	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
	V403	Set K602/603		Initiate Storage
07	V405	$A^1 \rightarrow X^1$		Transfer (A) to X
	V407	Set K206/207		Set Wait Storage FF
08	K207	Enable Full Write		Enable storage to write A_i in address M
13	V073	Storage Resume		
15	V420	Exit		

RNI

CODE	INSTRUCTION	FUNCTION
21 STQ	Store Q	Transfer (Q) to M

SEQUENCE: Write Operand (H^{4--}, V^{4--})

EXECUTION TIME: 4.4 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	$I^2\ I^3 \rightarrow R^1$		Prepare to form M
	V099	$U^1 \rightarrow U^2$		
01	V000	Clear X^1		Prepare to receive word to be stored (RNI)
02	V004	$Q^1 \rightarrow Q^2$		Place Q^1 in Q^2 for transfer to A (RNI)
03	V401	$Q^2 \rightarrow A^1$		Store Q^2 in A^1 temporarily
		$A^2 \rightarrow Q^1$		Store A_i in Q^1 temporarily
04	V402	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
	V403	Set K602/603		Set Initiate Storage FF
06	V404	$Q^1 \rightarrow Q^2$		Place A_i in Q^2 for return to A
	V406	Set K530/531		Set Partial Add In A FF
07	V405	$A^1 \rightarrow X^1$		Transfer Q^2 to X
	V407	Set K206/207		Set Write Operand Wait Storage FF
08	F577	Write		Enable full write
11	V409	$Q^2 \rightarrow A^1$		Restore A_i to A
		$A^2 \rightarrow Q^1$		Restore Q_i to Q
13	V073	Storage Resume		
15	V420	Exit		

24 22 !

TIME	TERM	COMMAND	CONDITION	REMARKS
	V408	Set K460/461		Set Return Jump FF. Conditions later commands.
		Set K560/561		Set $U^2 \rightarrow P^1$ FF. P^1 now contains first address of routine jumped to.
09	V409	$X^2 \rightarrow X^1$	Jump	Position P_i in X^1_{LA} for transfer to X^1_{UA}
11		Clear U^1		Prepare U^1 to receive new instruction
12	V971	$X^1_L \rightarrow X^2_u$		
13	N380	$X^2_u \rightarrow X^1_u$		Transfer P + 1 to X^1_u to write it into storage
		Clear X^1_L		
	V073	Storage Resume		
14	V420	Clear K206/207		Clear Write Operand Wait Storage FF
15		$I^5\ I^6\ I^7 \rightarrow U^1$		Transfer instruction from storage to U^1
		Half Exit		Execute lower instruction at new address

CODE	INSTRUCTION	FUNCTION
22 AJP	A Jump	Normal Jump: j = 0, 1, 2, or 3 and specified condition of (A) Return Jump: j = 4, 5, 6, or 7 and specified condition of (A)

SEQUENCE: Normal Jump (H^{1--}, V^{1--}); Write Operand (H^{4--}, V^{4--})

EXECUTION TIME: 4.0 usec min., 7.2 usec avg., 11.6 usec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V009	U^1uA U^2		Place m in U^2 for transfor to P
01	V000	Clear X^1		Prepare to receive address from P
Normal	Jump	Sequence	(j = 0-3)	
03	J050	Jump Exit	(j=0) (A=0) (j=1) (A≠0) (j=2) (A Pos) (j=3) (A Neg)	Jump to new address
	J053	Exit	No Jump	Jump condition not met; control passes to the next instruction.
	V103	Set 560/561	Jump	Transfer $U^2 \rightarrow P^1$
Write	Operand	Sequence	(j = 4-7)	
04	V043	Set K602/603	(j=4) (A=0) (j=5) (A≠0) (j=6) (A Pos) (j=7) (A Neg)	Initiate storage at new address if jump condition is met
06	V405	Adv. P^2 P^1	Jump	Add 1 to present address to provide for return to current program.
07		Exit	No Jump	Jump condition not met; control passes to new instruction.
	V407	Set K206/207		Set Write Operand Wait Storage FF - enables partial write upper and storage resume V971
08	V407	$P^1 \rightarrow X^2_L$	Jump	Prepare to store P + 1 (P has been increased by 1) in upper address portion of first word of new instruction routine.
	V407	Enable Partial Write Upper	Jump	Prepare to write next address of main routine into storage.

CODE	INSTRUCTION	FUNCTION
23 QJP	Q Jump	Normal Jump: j = 0, 1, 2 or 3 and specified condition of (Q) Return Jump: j = 4, 5, 6 or 7 and specified condition of (Q)

SEQUENCE: Normal Jump (H^{1--}, V^{1--}); Write Jump (H^{4--}, V^{4--})

4.0 usec min., 7.2 usec avg., 11.6 usec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	U^1uA → U^2		Place m in U^2 for transfer to P
01	V000	Clear X^1		Prepare to receive address from P RNI
Normal	Jump	Sequence	(j = 0-3)	
03	J050	Jump Exit	(j=0) (Q=0) (j=1) (Q≠0) (j=2) (Q Pos) (j=3) (Q Neg)	Jump to new address
	J053	Exit	No Jump	Jump condition is not met; control passes to the next instruction.
	V103	Set K560/561	Jump	Transfer U^2 → P^1
Write	Operand	Sequence	(j = 4-7)	
04	V403	Set K/602/603	(j=4) (Q=0) (j=5) (Q≠0) (j=6) (Q Pos) (j=7) (Q Neg)	Initiate Storage at new address if Jump condition is met
06	V405	Adv. P^2 → P^1	Jump	Add 1 to present address to provide for return to current program
07	V405	Exit	No Jump	Jump condition not met; control passes to the next instruction.
	V407	Set K206/207		Set Wait Operand Wait Storage FF - enables partial write upper and storage resume V971
08	V407	P^1 → X^2_L		Prepare to store P + 1 (P has been increased by one) in upper address portion of first word of new instruction routine
	V407	Enable Partial Write Upper	Jump	Prepare to write next address of main routine into storage

TIME	TERM	COMMAND	CONDITION	REMARKS
	V408	Set K460/461		Set Return Jump FF
		Set K560/561		Transfer $U^2 \rightarrow P^1$ - P now contains first address of routine jumped to
09	V409	$X^2 \rightarrow X^1$	Jump	Position P_i in X^1_{LA} for transfer to X^1u
11		Clear U^1		Prepare U^1 to receive new instruction
12	V971	$X^1_L \rightarrow X^2_u$		Transfer P + 1 to X^1_u to write into storage
13	V971	$X^2_u \rightarrow X^1_u$		
		Clear X^1_L		
	V073	Storage Resume		
14	V420	Clear K206/207		Clear Write Operand Wait Storage FF
15	V420	$I^5\ I^6\ I^7 \rightarrow U^1$		Transfer instruction from storage to U^1
		Half Exit		

CODE	INSTRUCTION	FUNCTION
24 MUI	Multiply Integer	Multiply (M) by (A); store the 96-bit product in QA

SEQUENCE: Iterative (H^{6--}, V^{6--})

25.2 usec. min., 25.2 usec + .8 usec/"1" in Q avg., 66.4 usec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Prepare to form M — RNI
	V000	Clear X^1		Prepare to set X^2 to "1's" — RNI
04	V602	Comp. $X^1 \rightarrow X^2$		Set X^2 to all "1's"
	V602	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
	V603	Set K602/603		Initiate storage
	V604	Set K208/209		Set wait storage
05	V605	Set K500/501	A Neg	Record the sign of A
07	V605	Clear R^1		Clear R - Prepare R to receive the shift count
	V077	Storage Resume		
08	V608	Set I^2		Generate shift count (00060)
12	V611	$I^2\ I^3 \rightarrow R^1$		Transfer shift count to R^1
13	V611	Partial Add $X^2 \rightarrow A^1$	Sign record FF set	If A_i is negative, complement A
	V073	Late Resume		
14	V614	Set K582/583		Set clear A FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (storage) to X
	V613	$A^2 \rightarrow Q^1$		Transfer the multiplier to Q
	V621	Clear K520/521		Clear the Partial Add in A FF
	V621	Initiate multiply step		

TIME	TERM	COMMAND	CONDITION	REMARKS
16	V620	$X^1 \rightarrow X^2$		Transfer multiplicand to X^2 for formation of partial products
	V621	$R^1 \rightarrow R^2$		Set R^2 equal to R^1
	V622	Clear K582/583		Clear the Clear A FF
17	V623	Complement the sign record FF	X Neg.	Establish the sign of the product
18	V623	Comp. $X^1 \rightarrow X^2$	X Neg.	If multiplicand is negative, make it positive. All operands are positive during multiply step.
Execute Multiply Step				
00	V634	Inhibit $A^1 \rightarrow A^2$		Prevent $A^1 \rightarrow A^2$ during shifting
		Clear K208/209		Clear Iterative Wait Storage FF
		Set K400/401		Set Shift Right FF
		Set K402/403		Set $R \neq 0$ FF
01	V635	Enter H630	K403 = 1	If $R \neq 0$ FF is set, initiate long or short multiply loop
02	V637	Reduce $R^1 \rightarrow R^2$		Reduce shift count and shift AQ right
		Right Shift A		
		Right Shift Q		
03	V637	$R^2 \rightarrow R^1$		
		$A^2 \rightarrow A^1$		
		$Q^2 \rightarrow Q^1$		
	V631	Enter H632	Q005 = 1	If lowest bit of Q is a "1" complete long loop which adds $X^2 \rightarrow A^1$
04	V630	Reduce $R^1 \rightarrow R^2$		If lowest bit of Q is a "0", shift AQ right and sample again - short loop.
		Right Shift A	Q000 = 1	
		Right Shift Q		
05	V637	$R^2 \rightarrow R^1$		
		$A^2 \rightarrow A^1$		
		$Q^2 \rightarrow Q^1$		

TIME	TERM	COMMAND	CONDITION	REMARKS
07	V633	Add $X^2 \rightarrow A^1$		If lowest bit of Q was "1" at 03 time, add X^2 to A^1 to form a partial product
	V637	Reduce $R^1 \rightarrow R^2$		
		Right Shift A		Reduce shift count and shift AQ right
		Right Shift Q		
				Repeat multiply loops until R is reduced to zero. When R = 0 initiate End Sign Correction.
End	Sign Correction			
00	V650	Clear K408/409		Clear the Inhibit $A^1 \rightarrow A^2$ FF
01	V651	Set K520/521		Set Partial Add in A FF
03	V651	Clear X^1		Set X^2 to all "1"s
04	V652	Comp. $X^1 \rightarrow X^2$		
	V653	Exit		
05	V653	Add $X^2 \rightarrow A^1$	K501 = 1	If SR FF = 1, complement the high order bits of the product.
06	V654	$Q^1 \rightarrow Q^2$		
07	V655	$Q^2 \rightarrow A^1$		Position product in QA
		$A^2 \rightarrow Q^1$		
09	V657	Add $X^2 \rightarrow A^1$	K501 = 1	If SR FF = 1, complement the low order bits of product.

CODE	INSTRUCTION	FUNCTION
25 DVI	Divide Integer	Divide (QA) by (M). Store the quotient in A, and the remainder in Q.

SEQUENCE: Iterative (H^{6--}, V^{6--})

EXECUTION TIME: 63.6 μsec min., 65.2 μsec avg., 66.4 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M (RNI)
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	
01	V000	Clear X^1		Prepare to set X^2 to "1's" (RNI)
02	V602	Clear K474/475		Clear the Dividend sign FF - in preparation to record the sign of the dividend.
04	V602	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
	V602	Comp $X^1 \rightarrow X^2$		Set X^2 to all "1's"
	V603	Set K602/603		Initiate storage
	V604	Set K208/209		Set Wait Storage FF
05	V603	Partial Add $X^2 \rightarrow A^1$	Q Neg.	Complement A if AQ is negative
	V605	Set K474/475	Q Neg.	Record the sign of the dividend
	V605	Set K500/501	Q Neg.	Set the sign record FF - record sign of quotient.
07	V606	Clear R^1		Prepare R to receive shift count
	V077	Storage Resume		
08	V608	Set K090/091		Generate the shift count in I^2
10	V609	$Q^1 \rightarrow Q^2$		Set Q^2 equal to Q^1
11	V609	$Q_2^2 \rightarrow A_1^1$ $A^2 \rightarrow Q^1$		Place highest order bits of QA in A Place lowest order bits of QA in Q.
12	V611	$I^2\ I^3 \rightarrow R^1$		Transfer shift count (608) to R
13	V611	Clear X^1		Prepare X to receive the divisor

TIME	TERM	COMMAND	CONDITION	REMARKS
08	V645	Enter H650	R = 0	If R = 0 initiate end sign correction
		Enter H640	R ≠ 0	If R ≠ 0 initiate divide timing chain
09	V647	Reduce $R^1 \rightarrow R^2$		
		Left Shift Q		Shift and reduce shift count
		Left Shift A		
10		$R^2 \rightarrow R^1$		
		$Q^2 \rightarrow Q^1$		
		$A^2 \rightarrow A^1$		
				Repeat divide step until R = 0, then initiate end-sign correction. Quotient is in Q when divide step is completed.
Divide	Step			
00	V646	Set K408/409		Set Inhibit $A^1 \rightarrow A^2$ FF
		Clear K208/209		Clear the Iterative Wait Storage FF
		Set K402/403		Set R ≠ 0 FF
01	V645	Set K406/407		Set Stop Shift FF
		Enter H640	R ≠ 0	If R ≠ 0 initiate divide timing chain
02	V647	Reduce $R^1 \rightarrow R^2$		
		Left Shift Q		Shift and reduce shift count
		Left Shift A		
03	V647	$R^2 \rightarrow R^1$		
		$Q^2 \rightarrow Q^1$		
		$A^2 \rightarrow A^1$		
07	V644	Add $X^2 \rightarrow A^1$	A>X	If A>X perform first subtraction and record first bit of quotient in Q
		Set $Q^{000} \rightarrow$ "1"		

TIME	TERM	COMMAND	CONDITION	REMARKS
		Enter H647	$R \neq 0$	If $R \neq 0$, initiate shift and reduce operations
End	Sign	Correction		
00	V650	Clear K408/409		Clear Inhibit $A^1 \rightarrow A^2$ FF
		Set K232/233	Q Neg.	If Q_f is negative set Divide Fault FF
01	V651	Set K520/521		Set Partial Add in A FF
03	V651	Clear X^1		Set X^2 to all "1's"
04	V652	Comp. $X^1 \rightarrow X^2$		
05	V653	Add $X^2 \rightarrow A^1$	A Neg.	If the dividend is negative, complement the remainder.
		Exit		
06	V654	$Q^1 \rightarrow Q^2$		Prepare to place Q (quotient) in A
07	V655	$Q^2 \rightarrow A^1$		Place quotient in A and remainder in Q
		$A^2 \rightarrow Q^1$		
09	V657	Add $X^2 \rightarrow A^1$	AQ SR = "1"	If quotient should be negative, complement (A).
	V611	Partial add $X^2 \rightarrow A^1$	Sign record FF set	Complement the highest order bits of the dividend if the dividend was initially negative
	V073	Storage resume		
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer the divisor from storage into X
	V621	Clear K520/521		Clear the Partial Add in A FF
16	V621	$R^1 \rightarrow R^2$		Equalize the R registers for shift counting
17	V623	Complement the sign record FF	X Neg.	Determine the sign of the quotient
18	V622	Comp. $X^1 \rightarrow X^2$	X Pos.	Make the divisor, M, negative.

CODE	INSTRUCTION	FUNCTION
26 MUF	Multiply Fractional	Multiply the fractional quantity in M by the fractional quantity in A, store the 96-bit product in AQ.

SEQUENCE: Iterative (H^{600}, V^{600})

EXECUTION TIME: 25.2 μsec min., 25.2 μsec + .8 μsec/"1" in Q avg., 66.4 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$	$b \neq 0$	Prepare to form M (RNI)
	V871	$I^2\ I^3 \rightarrow R^1$		
01	V000	Clear X^1		Prepare to set X^2 to all "1's" (RNI)
04	V602	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
		Comp. $X^1 \rightarrow X^2$		Prepare to complement the contents of A if it is negative
	V603	Set K602/603		Set Initiate Storage FF
	V604	Set K208/209		Set Iterative Wait Storage FF
05	V605	Set K500/501	A Neg.	Record the sign of A (Multiplier)
07	V605	Partial Add $X^2 \rightarrow A^1$	A Neg.	Complement the multiplier if negative
	V606	Clear R^1		Prepare R to hold the number of multiplication steps
	V677	Storage Resume		
08	V608	Set K090/091		Set I^2 Set FF multiplication count to 57_8
10	V609	$Q^1 \rightarrow Q^2$		Transfer the multiplier to Q
11		$A^2 \rightarrow Q^1$		
		$Q^2 \rightarrow A^1$		
12	V611	$I^2\ I^3 \rightarrow R^1$		Place shift count in R
13	V611	Clear X^1		Prepare X^1 to receive multiplicand
14	V612	$Q^1 \rightarrow Q^2$		Set Q^2 equal to Q^1
	V614	Set K582/583		Set Clear A FF. Prepare A to receive partial product.

TIME	TERM	COMMAND	CONDITION	REMARKS
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Enter multiplicand from storage into X^1
	V621	Clear K520/521		Clear Partial Add In A FF. Execute multiply step.
16	V620	$X^1 \rightarrow X^2$		Transfer multiplicand to X^2 for formati of partial products
	V621	$R^1 \rightarrow R^2$		Set $R^2 = R^1$
	V622	Clear K582/583		Clear the Clear A FF
17	V623	Complement AQ Sign Record	X Neg.	Establish the sign of the product
	V623	Comp. $X^1 \rightarrow X^2$	X Neg.	If multiplicand is negative, make it positive. All operands are positive durin multiply step.
Execute	Multiply	Step		
16	V634	Set K400/401		Set the Shift Right FF. This FF control the exit-time from multiply step.
		Set K402/403		Set $R \neq 0$ FF
00	V634	Inhibit $A^1 \rightarrow A^2$		Prevent $A^1 \rightarrow A^2$ during shifting
		Clear K208/209		Clear Iterative Wait Storage FF
		Set K400/401		Set Shift Right FF
		Set K402/403		Set $R \neq 0$ FF
01	V635	Enter H630	K403 = 1	If $R \neq 0$ FF is set, initiate long or short multiply loop
02	V637	Reduce $R^1 \rightarrow R^2$		Reduce shift count and shift AQ right
		Right Shift A		
		Right Shift Q		
03	V637	$R^2 \rightarrow R^1$		
		$A^2 \rightarrow A^1$		
		$Q^2 \rightarrow Q^1$		

TIME	TERM	COMMAND	CONDITION	REMARKS
	V631	Enter H632	Q005 = 1	If lowest bit of Q is a "1" complete long loop which adds $X^2 \rightarrow A^1$
04	V630	Reduce $R^1 \rightarrow R^2$		
		Right Shift A	Q000 = 1	If lowest bit of Q is a "0", shift AQ right and sample again - short loop
		Right Shift Q		
05	V637	$R^2 \rightarrow R^1$		
		$A^2 \rightarrow A^1$		
		$Q^2 \rightarrow Q^1$		
07	V633	Add $X^2 \rightarrow A^1$		If lowest bit of Q was "1" at 03 time, add X^2 to A^1 to form a partial product.
	V637	Reduce $R^1 \rightarrow R^2$		
		Right Shift A		Reduce shift count and shift AQ right
		Right Shift Q		
				Repeat multiply loops until R is reduced to zero. When R = 0 initiate End Sign Correction.
End	Sign	Correction		
00	V650	Clear K408/409		Clear the Inhibit $A^1 \rightarrow A^2$ FF
01	V651	Set K520/521		Set Partial Add In A FF
02	V651	$Q^1 \rightarrow Q^2$		
03	V652	$Q^2 \rightarrow A^1$		Position lower order bits of product in A for complementing if sign negative
		$A^2 \rightarrow Q^1$		
	V651	Clear X^1		
04	V652	Comp. $X^1 \rightarrow X^2$		Set X^2 to all "1's" to complement A if required
	V653	Exit		
05	V653	Partial Add $X^2 \rightarrow A^1$	AQ SR = 1	If product should be negative, complement low order bits.

TIME	TERM	COMMAND	CONDITION	REMARKS
06	V654	$Q^1 \rightarrow Q^2$		Transfer low order bits of product to Q and high order bits to A
	V655	$Q^2 \rightarrow A^1$		
		$A^2 \rightarrow Q^1$		
09	V657	Partial Add $X^2 \rightarrow A^1$	AQ SR = 1	If product should be negative, complement high order bits.

CODE	INSTRUCTION	FUNCTION
27 DVF	Divide Fractional	Divide a fractional quantity in AQ by a fractional quantity at M; store the quotient in A and the remainder in Q.

SEQUENCE: Iterative (H^{6--}, V^{6--})

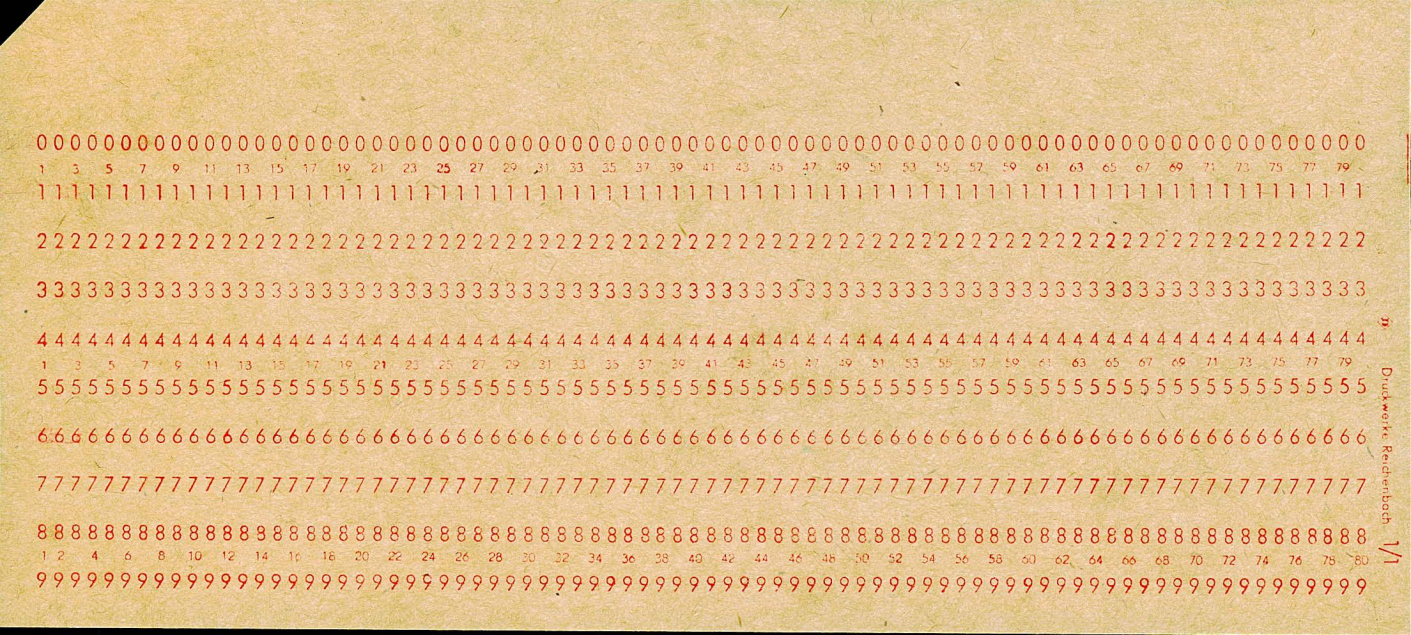
63.6 usec min., 65.2 usec avg., 66.4 usec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M (RNI)
	V871	$I^2\ I^3 \rightarrow R$	$b \neq 0$	Prepare to set X^2 to all "1's" (RNI)
01	V000	Clear X^1		
02	V602	Clear K474/475		Clear the Dividend Sign FF
04	V602	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
		Comp. $X^1 \rightarrow X^2$		Set X^2 to all "1's"
		Set K602/603		Initiate storage
		Set K208/209		Set Wait Storage FF
05	V605	Set K474/475	A Neg.	Record the sign of the dividend
		Set K500/501	A Neg.	Set AQ Sign Record FF
06	V606	Clear K490/491		Clear the F = 27 FF
07	V605	Part. Add $X^2 \rightarrow A^1$	A Neg.	Complement A if AQ is negative
	V606	Clear R^1		Prepare R for receipt of divide step count
	V077	Storage Resume		
08	V608	Set K090/091		Set $I^2 = 60_8$ Divide Step count
10	V609	$Q^1 \rightarrow Q^2$		Switch A and Q in order to complement Q
11	V609	$A^2 \rightarrow Q^1$		
		$Q^2 \rightarrow A^1$		
	V611	Clear K582/583		Clear the Clear A FF

TIME	TERM	COMMAND	CONDITION	REMARKS
12		$I^2\ I^3 \rightarrow R^1$		Transfer shift count to R
13	V611	Clear X^1		Prepare X to receive the divisor
		Part. Add $X^2 \rightarrow A^1$	Sign record = 1	Complement the low order bits of dividend
	V073	Storage Resume		
14	V620	Set K490/491		Set F = 27 FF
	V612	$Q^1 \rightarrow Q^2$		
		$Q^2 \rightarrow A^1$		Switch A and Q to original positions
		$A^2 \rightarrow Q^1$		
15	V844	$I^5\ I^6\ I^7 \rightarrow X^1$	V073	Transfer the divisor from memory to X
	V621	Clear K520/521		Clear Partial Add in A FF
16	V621	$R^1 \rightarrow R^2$		Set R^2 equal to R^1
	V620	$X^1 \rightarrow X^2$		Set X^2 equal to X^1
	V622	Clear K582/583		Clear the Clear A FF
		Set K408/409		Set Inhibit $A^1 \rightarrow A^2$
		Enter H637		Use part of multiply step to shift right once
17	V623	Set K500/501	X Neg.	Establish the sign of quotient
		Enter H646		Initiate Divide Step to divide
18	N934	Set K502/503	X Neg.	Complement sign record slave
Multiply Step				
18	V637	Right Shift Q		
		Right Shift A		
19		$A^2 \rightarrow A^1$		Right Shift AQ without changing shift count
		$Q^2 \rightarrow Q^1$		
		$R^2 \rightarrow R^1$		

TIME	TERM	COMMAND	CONDITION	REMARKS
Divide	Step			
00	V646	Set K402/403		Set R ≠ 0
		Set K408/409		Set Inhibit $A^1 \rightarrow A^2$ FF
		Clear K208/209		Clear the Iterative Wait Storage FF
01	V645	Set K406/407		Set the Stop Shift FF
		Enter H640		If R ≠ 0 initiate divide timing chain
02	V647	Reduce $R^1 \rightarrow R^2$		
		Left Shift A		Shift and reduce shift count
		Left Shift Q		
03	V647	$R^2 \rightarrow R^1$		
		$A^2 \rightarrow A^1$		
		$Q^2 \rightarrow Q^1$		
07	V644	Add $X^2 \rightarrow A^1$	$A \geq X$	If $A \geq X$ perform the first subtraction and record first bit of quotient in Q
		Set Q^{000} to "1"		
08		Enter H650	R = 0	If R = 0 initiate end sign correction
		Enter H640	R ≠ 0	If R ≠ 0 restart divide timing chain
09	V647	Reduce $R^1 \rightarrow R^2$		
		Left Shift A		Shift and reduce shift count
		Left Shift Q		
10		$R^2 \rightarrow R^1$		
		$A^2 \rightarrow A^1$		
		$Q^2 \rightarrow Q^1$		
				Repeat divide loop until R = 0; then initiate End Sign Correction. Quotient is held in Q when divide step is completed.

TIME	TERM	COMMAND	CONDITION	REMARKS
End	Sign	Correction		
00	V650	Clear K408/409		Clear the Inhibit $A^1 \rightarrow A^2$ FF
		Set K232/233	Q Neg.	Set the Divide Fault FF
01	V651	Set K520/521		Set Partial Add in A FF
03	V651	Clear X^1		Set $X^2 \rightarrow$ "1's"
04	V652	Comp. X^1 to X^2		
05	V653	Exit		
	V653	Add $X^2 \rightarrow A^1$	A Neg.	If the dividend is negative, complement the remainder.
06	V654	$Q^1 \rightarrow Q^2$		
07	V655	$Q^2 \rightarrow A^1$		Place quotient in A and remainder in Q
		$A^2 \rightarrow Q^1$		
09	V657	Part. Add $X^2 \rightarrow A^1$	AQ SR = "1"	If the quotient should be negative, complement (A)


Druckwerke Recherbach 1/1

CODE	INSTRUCTION	FUNCTION
30 FAD	Floating Add	Add two quantities packed in floating point format, one in A, one at M. Store the result in A, the residue in Q.

SEQUENCE: Iterative (H^{6--} V^{6--})

EXECUTION TIME: 11.2 usec min., 18.8 usec avg., 26.8 usec max.

TIME	TERM	COMMAND	CONDITION	REMARKS	
00	V099	$U^1 \rightarrow U^2$		Prepare to form M	
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$		
01	V000	Clear X^1		Prepare to set X^2 to "1's"	RNI
02	V602	Clear K474/475		Clear Dividend Sign FF	
04	V604	Comp. $X^1 \rightarrow X^2$		Set X^2 to all "1's"	
	V603	Set K602/603		Initiate storage	
		Set K208/209		Set wait storage	
05	V603	Part. Add $X^2 \rightarrow A^1$	A Neg.	If A is negative, complement it	
	V605	Set K500/501	A Neg.	Record the sign of the augend	
07	V605	$A^1 \rightarrow X^1$		Transfer the augend to X - prepare to separate the exponent and characteristic	
	V077	Storage Resume			
08	V607	$X^1_U \rightarrow X^2$		A 24 bit transfer. Bit 47 of X^2 is used to translate the "X^1 exponent to U^2 with sign extension" sign bit.	
09	V609	Set K582/583		Clear A at T = 10. Prepare A for the augend minus the exponent.	
10	V609	X^1 exp $\rightarrow U^2$ with sign extension		Remove the exponent from the characteristic. Also remove the bias value.	
	V609	Clear X^1 exp.		Retain the characteristic. The exponent is in U^1 and U^2.	
	V611	Clear K582/583		Clear Clear A FF	

TIME	TERM	COMMAND	CONDITION	REMARKS
12	V610	Comp. $X^1 \rightarrow X^2$	Sign Record = 1	Store the characteristic in X^2 in its correct signed format
		$X^1 \rightarrow X^2$	Sign Record = 0	
	V612	Clear K500/501		Clear the sign record FF. Prepare to record the sign of the addend.
13	V073	Late Resume		
	V611	Clear X^1		Prepare X^1 to receive Addend from storage
		Partial Add $X^2 \rightarrow X^1$		Transfer the augend characteristic to A^1 . The augend will stay in A if it has to be shifted to equalize exponents.
14	V612	$U^2 \rightarrow R^2$		Transfer the augend exponent into R. Prepare to compare exponents.
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer the addend to X^1
	V613	Comp. $R^2 \rightarrow R^1$		Make augend exponent negative. This will be used to compare exponents.
16	V620	$X^1 \rightarrow X^2$		Transfer the addend to X^2
17	V623	Set K500/501	X Neg.	Record the sign of the addend
Floating Point Sequence				(Time 00 equals time 18 of main sequence)
18	V623	Comp. $X^1 \rightarrow X^2$	X Neg.	Make the addend positive if it is negative.
	N934	Set K502/503	X Neg. At T=17	Set sign record slave
01	V660	$X^2 \rightarrow X^1$		If X had been complemented at T = 18 time, restore positive format into X^1.
02	V661	X^1 exp $\rightarrow U^2$ with extension		Remove exponent and bias from the addend
	V662	Clear K208/209		Clear wait storage
03	V661	$U^2 \rightarrow U^1$		Transfer addend exponent to U^1 to compare exponents
05	V663	Clear X^1 exp.		Separate exponent from the characteristic exponent is in U^2

TIME	TERM	COMMAND	CONDITION	REMARKS
06	V664	Comp. $X^1 \rightarrow X^2$	Sign Record = 1	Correct sign of characteristic
		$X^1 \rightarrow X^2$	Sign Record = 0	
07	V665	$X^2 \rightarrow X^1$		Transfer corrected characteristic to X^1
	V667	Clear K500/501		Clear sign record FF
08	V666	Add $R^1 \rightarrow U^2$		Form in U^2 the difference of the exponents
	V668	Set K450/450		Set disable clear "f" FF. Prepare to clear U^1_u but retain the function code. Used if the augend exponent is larger than the addend exponent.
09	V667	Clear X^1		Prepare X^1 to hold the augend if the addend has to be shifted
	V669	Set K510/511	U^2 Neg.	Set the U^2 sign record FF
	V669	Set K530/531		Set partial add in U^2 FF. Prepare to transfer the augend exponent to U^2 if the augend exponent was the larger.
11	V669	$A^1 \rightarrow X^1$		Transfer augend to X^1
	V670	Clear U^1_u	U^2 sign record = 1	Prepare for an add $R^1 \rightarrow U^2$ to store augend exponent if it is larger
	V671	Set K580/581	U^2 sign record = 1	Set clear A. Clear A to receive addend if it has to be shifted.
12	V670	$U^1 \rightarrow U^2$		Clear U^2. Used on a partial add $R^1 \rightarrow U^2$ if the augend has the larger exponent.
	V672	Clear K450/451		Clear Disable Clear "f" FF
13	V671	$X^2 \rightarrow X^1$	U^2 sign record = 0	Equalize X^1 and X^2 if addend exponent was larger
	V672	Clear K580/581		Clear Clear A FF
14	V673	Partial add $R^1 \rightarrow U^2$	U^2 sign record = 1	Transfer augend exponent to U^2 for storage if it was the larger
15	V673	Part. Add $X^2 \rightarrow A^1$	U^2 sign record = 1	Transfer the addend to A to be shifted if the augend sign was the larger

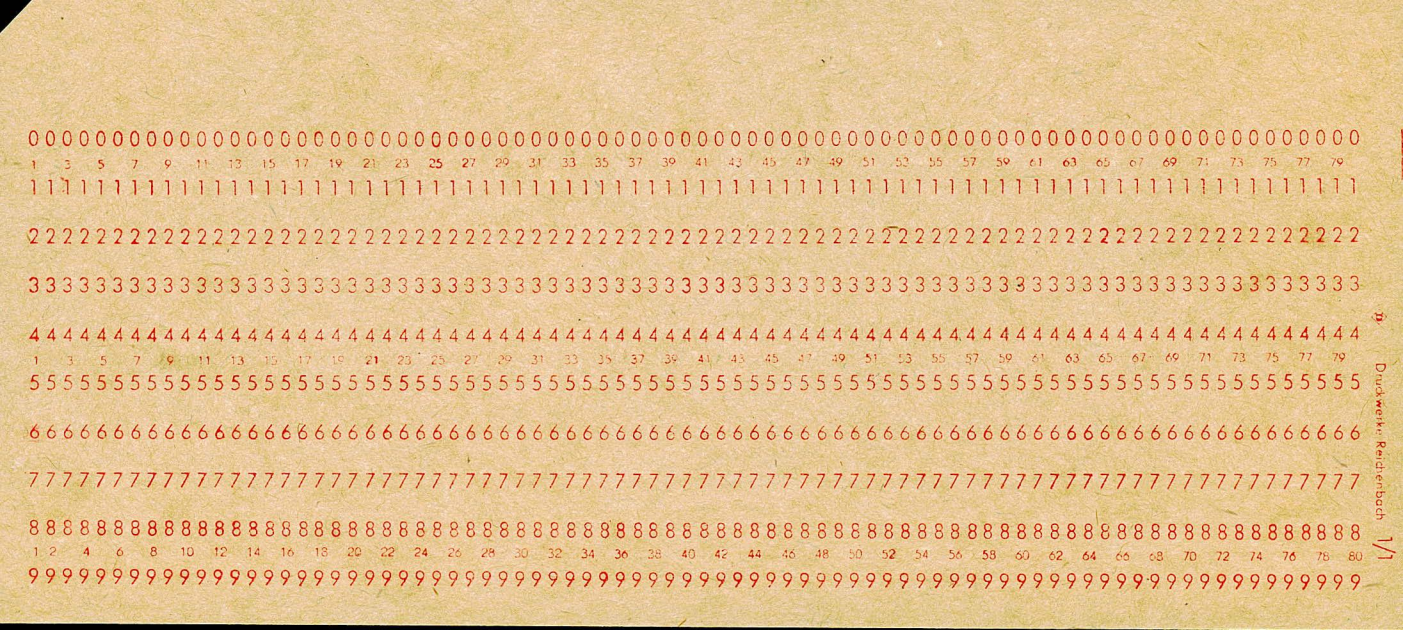
TIME	TERM	COMMAND	CONDITION	REMARKS
		Comp. $R^2 \rightarrow R^1$	U^2 sign record = 1	Form the positive shift count
		$R^2 \rightarrow R^1$	U^2 sign record = 0	Transfer shift count to R^1
16	V674	$R^1 \rightarrow R^2$		Equalize R register for shift counting
		$X^1 \rightarrow X^2$		Transfer characteristic of the value with the largest exponent to X^2
	V590	Set K410/411		Set R = 0 FF. Used to determine if a shifting operation will be executed.
17	N991	Set K414/415		Set R = 0 FF slaves
	N931	Set K412/413		
	V951	Clear K520/521		Clear the Partial Add in A FF. Prepare to form sum.
18	N570	R = 0	R = 0	Disable shifting if none is to be executed (R = 0)
		Clear K410/411	R = 0	
	N991 N931	Clear R = 0 FF slaves	R = 0	
19	V591	$A^2 \rightarrow Q^1$		Make upper bits of Q positive to check after shifting
20	V594	Set K582/583	R 36_{10}	Clear A if a shift of 36 or more places is necessary
		Set K322/323		Set floating shift exit control. Enable entrance to completion of this operation after the shifting is finished.
		Set K310/311	R 36_{10}	Enable shift right if the shift count is 36_{10} or less
		Set K312/313		
	V594	Set K310/311	R 36_{10}	Enable shift right. Start first shift and reduce; shift and reduce until R = 0 (See shift control for shift operation).
		Set K312/313	R 36_{10}	

TIME	TERM	COMMAND	CONDITION	REMARKS
21	V221	Set K320/321		Set shift exit control. Enable the cleaning of the Shift Right FFs.
		Into H676	$R > {}^{36}10$	Continue with sequence if no shifting is to be executed. Enter round, normalize and final assembly.
22	V676	Clear K322/323		Clear the floating shift exit control
		Clear K408/409		Clear the Inhibit $A^1 \rightarrow A^2$ FF
		Set K506/507	A and Q signs unlike	Set the Execute Round. If a significant bit in Q is greater than one half, prepare to round the answer.
23	V866	Clear R^1		Prepare R to complement the exponent
	V676	$U^2 \rightarrow U^1$		Transfer largest exponent into U^1 for an add $R^1 \rightarrow U^2$ to complement it
	V619	Clear K582/583		Clear Clear A FF
24	V676	$R^1 \rightarrow R^2$		Prepare to complement the exponent
25	V677	Clear X^1		Prepare X^1 to receive rounding count
		Add $X^2 \rightarrow A^1$		Form the sum of the characteristics
	V678	Comp. $R^2 \rightarrow R^1$		Prepare to complement the exponent
26	V680	Set K070/071		Set the Set $X^2 \rightarrow 1$ FF
	V679	Set K508/509	A Neg.	Remembers A was negative
		Comp. $X^1 \rightarrow X^2$	A Neg.	Set X^2 to a negative one
		$X^1 \rightarrow X^2$	A Pos.	Set X^2 to a positive one
		Partial Add	U^2 sign record = 1	Complement the exponent; make it positive
		$R^1 \rightarrow U^2$		
	V670	Clear K530/531		Clear the partial add in U^2 FF. Prepare for a full add of $R^1 \rightarrow U^2$ to form sum of the exponent and any normalizing count.

TIME	TERM	COMMAND	CONDITION	REMARKS
27	V680	$U^2 \rightarrow U^1$		Prepare to form sum of exponent and normalizing count
	V681	Clear K070/071		Clear the Set $X^2 \rightarrow 1$ FF
28	V681	$R^1 \rightarrow R^2$		Prepare to form normalize count
	V682	Clear K508/509		
		Clear K070/071		Clear Set $X^2 \rightarrow 1$ enable FF
31	V683	Add $X^2 \rightarrow A^1$	Execute Round = 1	Execute rounding operation
Execute Round				
33	V686	Clear X^1		Prepare to transfer the sum to X^1 for final assembly
34	V688	Set K408/409	A37 ≠ A36	Inhibit $A^1 \rightarrow A^2$ for normalize right shift
35	V688	Shift A Q right	A37 ≠ A36	
		Reduce $R^1 \rightarrow R^2$		
	V689	Clear K318/319	A = 0	Clear normalize shift left FF if A = 0
		Set K324/325	A37 = A36	Set $\overline{\text{Right Shift Exit}}$ Control. Prepare for exit if normalizing is done by a left shift
		Into H690	A37 ≠ A36	Continue on in sequence if a right shift is performed
	V691	Clear K324/325	Left shift or No Normalize	Clear $\overline{\text{Right Shift Exit}}$ FF
		NOTE:		On a normalize shift left, the sequence could be contined at any even time from T = 34 on
36	V690	Clear K408/409	A37 ≠ A36 at T = 35	Clear inhibit $A^1 \rightarrow A^2$ FF
39	V698	Comp. $R^2 \rightarrow R^1$	A37 ≠ A36 at T = 35	Prepare to increase the exponent if the normalize was a right shift

TIME	TERM	COMMAND	CONDITION	REMARKS
Execute Final Assembly				
41	V691	$A^1 \rightarrow X^1$		Transfer normalized sum to X^1 for final assembly
	V693	Clear K510/511		Clear U^2 Sign Record FF
		Set K404/405	A ≠ 0	Set Inhibit Add. Provide enable to transfer final answer into A. If A is zero, no bias value is added to the sum.
		Set K512/513	A Neg.	Bit X^1 sign record. Record sign of final answer.
		Set K520/521		Set Partial Add in A FF. Prepare for final transfer of sum to A.
42	V692	$X^1 \rightarrow X^2$	A Pos.	Form the positive sum in X in preparation to adding the exponent and bias
		Comp. $X^1 \rightarrow X^2$	A Neg.	
43	V693	$X^2 \rightarrow X^1$		Form positive sum in X; only necessary if previous step was comp. $X^1 \rightarrow X^2$
	V695	Set K582/583		Clear A. Prepare A to receive floating packed sum.
		Clear K326/327		Clear Connect A47 to A00 FF
44	V694	Add $R^1 \rightarrow U^2$	Inhibit Add FF = 1	Form the final exponent. Normalizing count plus the original exponent.
45	V696	U^2 exp $\rightarrow X^1$		Insert exponent and bias into X^1. Form final positive answer in X^1.
	V619	Clear K582/583		Clear Clear A FF
	V699	Clear K512/513		Clear X^1 Sign Record FF
46	V696	$X^1 \rightarrow X^2$	X^1 sign record FF = 0	Form the correct signed sum in X^2 for transfer to A
		Comp. $X^1 \rightarrow X^2$	X^1 sign record FF = 1	
	V596	Clear K404/405		Clear $\overline{\text{Inhibit}}$ Add FF

TIME	TERM	COMMAND	CONDITION	REMARKS
47	V699	Exit		
		Partial add	Not $\overline{\text{Inhibit}}$	Transfer sum to A
		$X^2 \rightarrow A^1$	Add FF = 1	
	V597	Set K584/585	Underflow fault	Clear A if an underflow fault occurs
		Set K238/239	Overflow fault	Record overflow fault
48	V598	Set K246/247	Underflow fault	Record underflow fault
49	V599	Clear K584/585		Clear the Clear A FF


0000000000000000000000000000000000000000000000000000000000000000000000000000000000
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79
1111111111111111111111111111111111111111111111111111111111111111111111111111111111
2222222222222222222222222222222222222222222222222222222222222222222222222222222222
3333333333333333333333333333333333333333333333333333333333333333333333333333333333
4444444444444444444444444444444444444444444444444444444444444444444444444444444444
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79
5555555555555555555555555555555555555555555555555555555555555555555555555555555555
6666666666666666666666666666666666666666666666666666666666666666666666666666666666
7777777777777777777777777777777777777777777777777777777777777777777777777777777777
8888888888888888888888888888888888888888888888888888888888888888888888888888888888
1 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80
9999999999999999999999999999999999999999999999999999999999999999999999999999999999
Druckwerke Reichenbach
1/1

CODE	INSTRUCTION	FUNCTION
31 FSB	Floating Subtract	Subtract two quantities packed in floating point format, one in A, one in M. Store the results in A, the residue in Q.

SEQUENCE: Iterative (H^{6--}, V^{6--})

11.2 usec min., 18.8 usec avg., 26.8 usec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Form M
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Form M RNI
01	V000	Clear X^1		
02	V602	Clear K474/475		Clear Dividend Sign FF
04	V602	Comp. $X^1 \rightarrow X^2$		Prepare to complement (A) if negative
		Add $R^1 \rightarrow U^2$	$b \neq 0$	Add b to m to form M
		Set K602/603		Initiate storage
		Set K208/209		Set wait storage
05	V603	Partial Add $X^2 \rightarrow A^1$	A Neg.	Make the minuend positive
	V605	Set K500/501	A Neg.	Record the sign of the minuend
06	N934	Set K502/503	A Neg.	Set sign record slave
07	V605	$A^1 \rightarrow X^1$		Transfer the minuend to X. Prepare to separate the exponent and characteristic
	V077	Storage Resume		
08	V607	$X^1_U \rightarrow X^2_U$		A 24 bit transfer. Bit 47 of X^2 is used to translate the X^1 exponent to U^2 with extension sign bit.
09	V609	Set K582/583		Clear A at T = 10. Prepare A for the minuend minus the exp.
10	V609	X^1 Exp $\rightarrow U^2$ with sign extension		Remove the exponent from the characteristic; also remove the bias value.
11	V609	Clear X^1 Exp.		Retain the characteristic. The exponent is in U^1 and U^2

TIME	TERM	COMMAND	CONDITION	REMARKS
	V611	Clear K582/583		Clear Clear A FF
12	V610	Comp. $X^1 \rightarrow X^2$	S. R. = 1	Store the characteristic in X^2 in its correct signed format
		$X^1 \rightarrow X^2$	S. R. = 0	
	V612	Clear K500/501		Clear the S. R. FF. Prepare to record the sign of the minuend.
13	V073	Late Resume		
	V611	Clear X^1		Prepare X^1 to receive subtrahend from storage
		Part. Add $X^2 \rightarrow A^1$		Transfer minuend back to A
14	V612	$U^2 \rightarrow R^2$		Transfer the minuend exp. into R. Prepare to compare exps. to see which is larger.
	V612	$Q^1 \rightarrow Q^2$		Clear Q^2
15	V613	Comp. $R^2 \rightarrow R^1$		Make minuend exp. negative
	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer subtrahend to X^1
16	V620	$X^1 \rightarrow X^2$		Equalize X registers
		Clear K582/583		Clear Clear A FF.
17	V623	Enter H660		Floating Point Sequence
		Set K500/501	X Neg.	Record the sign of the subtrahend
18	V623	Comp. $X^1 \rightarrow X^2$	X Neg.	Complement the subtrahend
	N934	Set K502/503	X Neg. at T = 17	
Floating Point Sequence (Time 00 equals T18 of Main Sequence)				
01	V660	$X^2 \rightarrow X^1$		If X had been complemented at T = 18, restore positive format into X^1.
02	V661	X^1 Exp $\rightarrow U^2$ with extension		Remove exp. and bias from the subtrahend
		Clear K208/209		Clear wait storage

TIME	TERM	COMMAND	CONDITION	REMARKS
03		$U^2 \rightarrow U^1$		Transfer subtrahend to U^1 to compare exponents
05	V663	Clear X^1 Exp.		
06	V664	Comp. $X^1 \rightarrow X^2$	S. R. = 1	Complement subtrahend if positive Store augend in X^2
		$X^1 \rightarrow X^2$	S. R. = 0	
07	V665	$X^2 \rightarrow X^1$		Set $X^1 = X^2$
		Clear K500/501		Clear Sign Record FF
08	V666	Comp. $X^1 \rightarrow X^2$		Set up subtraction
	V668	Set K450/451		Set Disable Clear f FF. Prepare to clear U^1_u but retain the function code. Used if the minuend is larger than the subtrahend exp. U^1.
	V666	Add $R^1 \rightarrow U^2$		Form in U^2 the difference of the exponents
09	V667	Clear X^1		Prepare X^1 to hold the minuend if the subtrahend has to be shifted
	V669	Set K510/511	U^2 Neg.	Set the U^2 Sign Record FF
	V669	Set K530/531		Set Partial Add in U FF. Prepare to transfer the minuend exp. to U^2 if the minuend exp. was the larger
10	V670	Enter H671		
11	V669	$A^1 \rightarrow X^1$		Store minuend in X^1. Subtrahend may have to be transferred to A.
	V670	Clear U^1_u	U^2 S. R. = 1	Prepare for an add $R^1 \rightarrow U^2$ to store minuend exp. if it is larger
	V671	Set K580/581	U^2 S. R. = 1	Set Clear A FF. Clear A to receive subtrahend if it is to be shifted.
12	V670	$U^1 \rightarrow U^2$		Clear U^2. Used on a partial add $R^1 \rightarrow U^2$ if the minuend has the larger exp.
	V672	Clear K450/451		Clear Disable Clear F FF

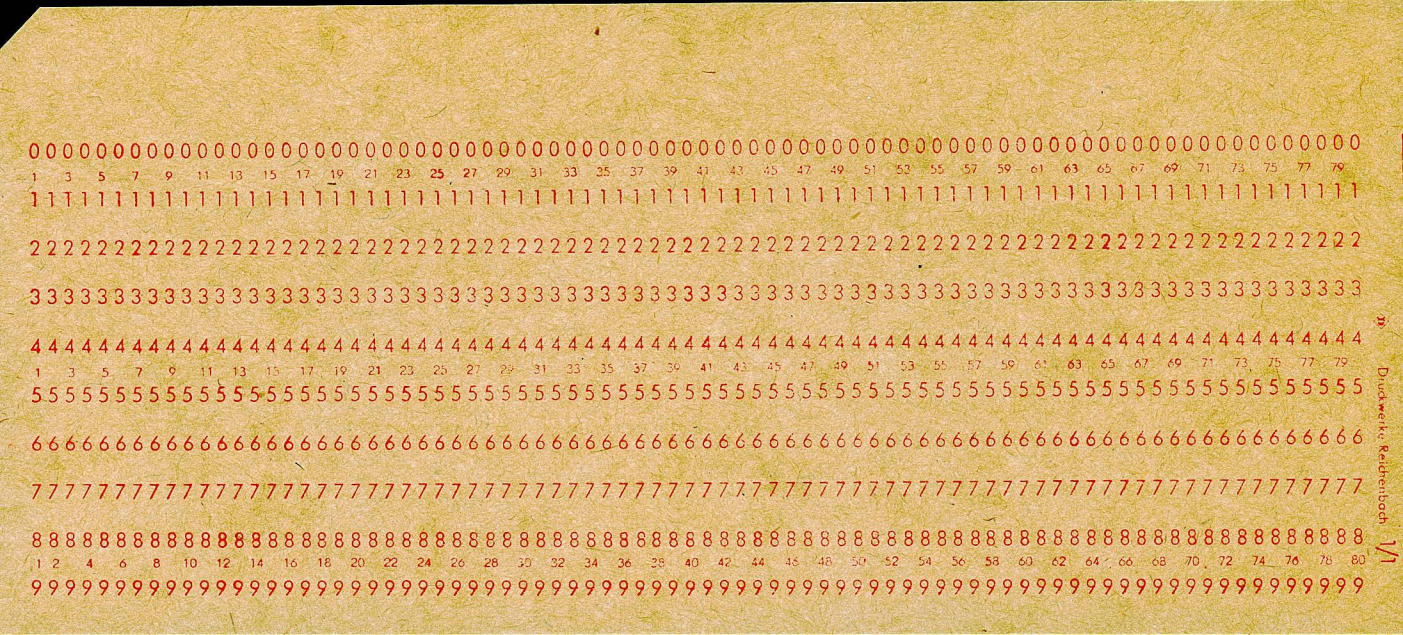
TIME	TERM	COMMAND	CONDITION	REMARKS
13	V672	Clear K580/581		Clear Clear A FF
	V671	$X^2 \rightarrow X^1$	U^2 S. R. = 0	Equalize X^1 and X^2 if subtrahend exp. was larger
14	V672	Part. Add $R^1 \rightarrow U^2$	U^2 S. R. = 1	Transfer minuend exp. to U^2 for storage if it was the larger
15	V673	Part. Add $X^2 \rightarrow A^1$	U^2 S. R. = 1	Transfer the subtrahend to A to be shifted
		Comp. $R^2 \rightarrow R^1$	U^2 S. R. = 1	Form the positive shift count
		$R^2 \rightarrow R^1$	U^2 S. R. = 0	Transfer shift count to R^1
16	V674	$R^1 \rightarrow R^2$		Equalize R register for shift counting
		$X^1 \rightarrow X^2$		Transfer characteristic of the value with the largest exp. into X^2
	V590	Set K410/411	R = 0	Set R = 0 FF. Used to determine if a shifting operation will be executed.
17	N991 N931	Set K414/415 Set K412/413		Set R = 0 FF slaves
	V951	Clear K520/521		Clear Partial Add in A FF. Prepare to form sum.
18	N570	Clear K410/411	R = 0	Disable if no shifting is to be executed (R = 0)
Shift Control				
19	V591	$A^2 \rightarrow Q^1$		Make upper bits of Q positive to check after shifting
20	V594	Set K582/583	$R \geq 36_{10}$	Clear A if a shift of 36 or more places is necessary
		Set K310/311 Set K312/313	$R \leq 36_{10}$	Enable Shift Right if the shift count is 36_{10} or less
		Set K322/323		Set floating shift exit control. Enable entrance to completion of this operation after the shifting is completed.

TIME	TERM	COMMAND	CONDITION	REMARKS
21	V221	Set K320/321		Set Shift Exit Control FF. Enable the clearing of the Shift Right FFs.
		Into H676	$R > {}^{36}_{10}$	Continue with sequence if no shifting is to be executed
				Shift the coefficient in AQ right, reducing R until R = 0. This establishes two quantities with equal exponents.
Round, Normalize and, Final Assembly				
(Timing sequence continued from Floating Point and Shift Control)				
22	V676	Clear K322/323		Clear Shift Exit Control FF
		Clear K408/409		Clear Inhibit $A^1 \rightarrow A^2$ FF
		Set K506/507	A and Q signs unlike	Set the Execute Round FF. If a significant bit is in Q^{47} - Q is greater than one half then prepare to round.
23	V676	$U^2 \rightarrow U^1$		Transfer larger exp. into U^1 for an add $R^1 \rightarrow U^2$ to complement it
	V619	Clear K582/583		Clear Clear A FF
	V866	Clear R^1		Prepare R to complement exp.
24	V676	$R^1 \rightarrow R^2$		
25	V677	Clear X^1		Prepare X^1 to receive rounding count
		Add $X^2 \rightarrow A^1$		Form the sum of the characteristics
	V678	Comp. $R^2 \rightarrow R^1$		Prepare to complement the exp.
	V679	Set K508/509	A Neg.	Remembers A was negative for set X^2 to 1
26	V679	Part. Add $R^1 \rightarrow U^2$	U^2 S. R. = 1	Complement the exp; make it positive
		Comp. $X^1 \rightarrow X^2$	A Neg.	Set X^2 to a negative one
		$X^1 \rightarrow X^2$	A Pos.	Set X^2 to a positive one
	V680	Set K070/071		Set X^2 to 1 FF. Disable the $X^1 \rightarrow X^2$ or Comp. $X^1 \rightarrow X^2$ enables into X^{002} and enable X^2 to set to a ±1. Rev. A

TIME	TERM	COMMAND	CONDITION	REMARKS
		Clear K530/531		Clear Part. Add in U^2 FF
	V678	Comp. $X^1 \rightarrow X^2$	A Neg.	Set X^2 to a negative one } Round
		$X^1 \rightarrow X^2$	A Pos.	Set X^2 to a positive one } Round
27	V680	$U^2 \rightarrow U^1$		Prepare to form sum of exp. and normalizing count
28	V681	$R^1 \rightarrow R^2$		Prepare to form normalize count
	V682	Clear K070/071		Clear Set X^2 to 1 FF
		Clear K508/509		
31	V683	Add $X^2 \rightarrow A^1$	Execute Round = 1	Execute rounding operation
Execute Normalize				
33		Clear X^1		Prepare to transfer the sum to X^1
34	V688	Set K408/409	$A^{37} \neq A^{36}$	Inhibit $A^1 \rightarrow A^2$ for normalize right shift
35	V688	Shift AQ Right	$A^{37} \neq A^{36}$	
	V689	Clear K318/319	A = 0	Clear normalize shift left FF if A = 0
		Set K324/325	$A^{37} = A^{36}$	Set $\overline{\text{Right Shift Exit}}$ Control - Prepare for exit if normalizing is done by a left shift
		Into H690	$A^{37} \neq A^{36}$	Continue in sequence if a right shift is performed
	V691	Clear K324/325	Left shift or No Normalize	Clear $\overline{\text{Right Shift Exit}}$ FF
		NOTE:		On a normalize shift left, the sequence could be continued at any even time from T = 34 on
36	V690	Clear K408/409	$A^{37} \neq A^{36}$ at T = 35	Clear Inhibit $A^1 \rightarrow A^2$ FF
39	V698	Comp. $R^2 \rightarrow R^1$	$A^{37} \neq A^{36}$ at T = 35	Prepare to increase the exponent if the normalize was a right shift

TIME	TERM	COMMAND	CONDITION	REMARKS
Execute Final Assembly				
41	V691	$A^1 \rightarrow X^1$		Transfer normalized sum to X^1 for final assembly
	V693	Clear K510/511		Clear U^2 Sign Record FF
		Set K404/405	A ≠ 0	Set $\overline{\text{Inhibit}}$ Add. Provide enable to transfer final answer into A. If A is zero, no bias value is added to the sum.
		Set K512/513	A Neg.	Bit X^1 sign record. Record sign of final answer.
		Set K520/521		Set Partial Add in A FF. Prepare for final transfer of sum to A.
42	V692	$X^1 \rightarrow X^2$	A Pos.	Form the positive sum in X in preparation to adding the exponent and bias
		Comp. $X^1 \rightarrow X^2$	A Neg.	
43	V693	$X^2 \rightarrow X^1$		Form positive sum in X. Only necessary if previous step was comp. $X^1 \rightarrow X^2$.
	V695	Set K582/583		Clear A. Prepare A to receive floating packed sum.
		Clear K326/327		Clear Connect A^{47} to A^{00} F
44	V694	Add $R^1 \rightarrow U^2$	$\overline{\text{Inhibit}}$ Add FF = 1	Form the final exponent. Normalizing count plus the original exponent.
45	V696	U^2 exp $\rightarrow X^1$		Insert exponent and bias into X^1_1 - form final positive answer in X^1
	V619	Clear K582/583		Clear Clear A FF
	V699	Clear K512/513		Clear X^1 Sign Record FF
46	V696	$X^1 \rightarrow X^2$	X^1 sign record FF = 0	Form the correct signed sum in X^2 for transfer to A
		Comp. $X^1 \rightarrow X^2$	X^1 sign record FF = 1	
	V596	Clear K404/405		Clear $\overline{\text{Inhibit}}$ Add FF

TIME	TERM	COMMAND	CONDITION	REMARKS
47	V699	Exit		
		Partial add $X^2 \rightarrow A^1$	$\overline{\text{Inhibit}}$ Add FF = 1	Transfer sum to A
	V597	Set K584/585	Underflow fault	Clear A if an underflow fault
		Set K238/239	Overflow fault	Record overflow fault
48	V598	Set K246/247	Underflow fault	Record underflow fault
49	V599	Clear K584/585		Clear the Clear A FF


0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79
1111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
2222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
3333333333333333333333333333333333333333333333333333333333333333333333333333333333333333333333333333333333333333333333333333333333
4444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79
5555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555
6666666666666666666666666666666666666666666666666666666666666666666666666666666666666666666666666666666666666666666666666666666666
7777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777
8888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888
1 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80
9999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999
Druckwerke Reichenbach
1/1

CODE	INSTRUCTION	FUNCTION
32 FMU	Floating Multiply	Multiply a number packed in floating point in A with a number, also in floating point, in M. Store the product in A, the residue in Q.

SEQUENCE: Iterative (H^{6--}, V^{6--})

EXECUTION TIME: 3.2 μsec min., 36.0 μsec avg., 57.2 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		
	V871	$I^2\ I^3 \rightarrow R^1$	b ≠ 0	Transfer m to U^2 (RNI)
01	V000	Clear X^1		Prepare X to receive the multiplier
04	V602	Comp $X^1 \rightarrow X^2$		Prepare to complement the multiplier
	V602	Add $R^1 \rightarrow U^2$	b ≠ 0	Add b to m to form M
	V603	Set K602/603	A ≠ 0	Initiate storage, if A equals zero. No multiplication takes place and the answer is zero.
	V603	Set K208/209	A ≠ 0	Set wait storage
05	V603	Partial add $X^2 \rightarrow A^1$	A Neg.	Complement the multiplier
	V605	Set K500/501	A Neg.	Record the sign of the multiplier
06	N934	Set K502/503	A Neg. at 05 time	Set sign record FF slave
07	V605	$A^1 \rightarrow X^1$		Transfer the multiplier to X^1 to separate the exponent and characteristic
	V077	Storage Resume		
08	V607	$X^1{}_U \rightarrow X^2{}_U$		Used on the extend sign operation for a X^1 exp → U^2 with sign extension
	V608	Set K090/091		Set the set I^2 FF. This generates the shift count for multiply step 36_{10}.
09	V607	Exit	A = 0	If the multiplier is zero, exit.
	V609	Set K582/583		Set Clear A FF. Clear A. Prepare to receive the multiplier with exponent extracted.

TIME	TERM	COMMAND	CONDITION	REMARKS
10	V609	X^1 exp → U^2		Extract exponent from multiplier
	V611	Clear K582/583		Clear Clear A FF
	V609	Clear X^1 exp		Clear the exponent portion of X
12	V610	X^1 → X^2		Transfer multiplier to X for transfer to Q via A. (Multiplier is always held in Q during multiply step).
13	V073	Late Resume		
	V611	Partial add X^2 → A^1		Transfer multiplier to A
	V611	A^2 → Q^1		
	V611	Clear X^1		Prepare X to receive multiplicand from storage
14	V612	U^2 → R^2		Store the multiplier exponent in R^2. Prepare to form the sum of the exponent
	V614	Set K582/583		Set Clear A. Prepare A to receive partial products.
15	V073	I^5 I^6 I^7 → X^1		Transfer multiplicand to X^1
	V613	R^2 → R^1		Prepare to form sum of exponents
		A^2 → Q^1		Transfer multiplier to Q
16	V620	X^1 → X^2		
	V622	Clear K582/583		Clear Clear A FF
17	V623	Comp. Sign Record FF	X Neg.	Record the sign of the product
		Into H^{660}		Enter floating point sequence
18	N934	Comp. Sign Record FF Slave	X Neg. at T = 17	

TIME	TERM	COMMAND	CONDITION	REMARKS
Floating Point Sequence (Time 00 equals T = 18 of main sequence)				
02	V662	Clear K208/209		Clear Wait Storage
03	V661	$A^2 \rightarrow Q^1$	X = A = 0	Clear Q if multiplicand or multiplier are equal to zero
04	V662	$Q^1 \rightarrow Q^2$	X = 0	Clear Q^2
	V662	Comp. $X^1 \rightarrow X^2$	X Neg.	Form positive format of the multiplicand
05	V663	Exit	X = 0	
	V663	$X^2 \rightarrow X^1$		Positive format of multiplicand to X^1 to extract the exponent
06	V665	X^1 exp $\rightarrow U^2$		Multiplicand exponent to U^2 minus the bias
07	V665	$U^2 \rightarrow U^1$		Prepare to add exponents
	V665	Clear X^1 exp		Clear the exponent portion of the multiplicand
08	V666	$X^1 \rightarrow X^2$		Transfer multiplicand to X^2 to form partial products during multiply step
12	V670	Add $R^1 \rightarrow U^2$		Form the sum of the exponents in U^2
	V672	Clear K450/451		Clear Disable Clear F FF
13	V672	Clear K580/581		Clear Clear A FF
	V671	Clear R^1		Prepare R to receive the shift count for the multiply operation
	V673	Set K530/531		Set partial add in U^2 FF
14	V673	I^2 $I^3 \rightarrow R^1$		Transfer shift count (36_{10}) to R^1
15	V675	Into H^{634}		To multiply step
16	V674	$R^1 \rightarrow R^2$		Equalize shift count in R
Multiply Step (Time continued from floating point sequence)				
16	V634	Set K402/403		Set R = 0 FF. Enable for termination of long loop.

TIME	TERM	COMMAND	CONDITION	REMARKS
		Set K400/401		Set Shift Right FF. Enable for termination of short loop.
		Clear K208/209		Clear Wait Storage
		Set K408/409		Set Inhibit $A^1 \rightarrow A^2$. Prepare for shifting.
		Into H635		Start first check
		Into H637		To shift and reduce
17	V951	Clear K520/521		Clear Partial Add in A FF. Prepare for the sum of the partial products.
	V635	Set K406/407		Set Stop Shift FF. Enable termination of shifting.
		Into H630		
18	V637	Right Shift AQ		Prepare to check second multiplier bit
		RS $A^1 \rightarrow A^2$		
		RS $Q^1 \rightarrow Q^2$		
		Reduce $R^1 \rightarrow R^2$		
	V630	Into H637	$Q^{000} = 1$	Prepare to shift if first multiplier bit was a 0
19	V888	$A^2 \rightarrow A^1$		
	V886	$Q^2 \rightarrow Q^1$		
		$R^2 \rightarrow R^1$		
	V631	Into H632	$Q^{005} = 1$	Take a long loop if first multiplier bit was "1"
23	V633	Add $X^2 \rightarrow A^1$		Form first partial product. The timing of the Multiply Step depends on the use of the multiplier.
			S.R. FF = 1	Complement the product
		Exit to End Correction	R = 0	

TIME	TERM	COMMAND	CONDITION	REMARKS
Execute Round, Normalize, Final Assembly				
23	V676	$U^2 \rightarrow U^1$		Prepare to form the sum of the normalizing count and the exponent
	V677	Set K506/507	A & Q signs unlike	Set the Execute Round FF if a significant bit is in Q^{47}
24	V676	$R^1 \rightarrow R^2$		Prepare to form normalizing count
25	V677	Clear X^1		Prepare to set $X^2 \rightarrow 1$
	V678	Comp. $R^2 \rightarrow R^1$		Prepare to form normalizing count
	V679	Set K508/509	A Neg.	Used to enable Set X^2 to 1 F terms
26	V678	Comp. $X^1 \rightarrow X^2$	A Neg.	Set X^2 to a negative 1
	V678	$X^1 \rightarrow X^2$	A Pos.	Set X^2 to a positive 1
	V680	Set K070/071		Set X^2 to 1
		Clear K530/531		Prepare to form the sum of the normalize count and exponent
27	V951	Clear K520/521		Clear the partial add in A FF. Prepare to form the rounded product.
	V681	$R^1 \rightarrow R^2$		Prepare for normalizing count
	V682	Clear K070/071		
		Clear K508/509		
31	V683	Add X^2 to A^1	Execute Round FF Set	Form Rounded product
Execute Normalize				
33	V686	Clear X^1		Prepare X for final assembly
	V687	Set K326/327	$A^{37} = A^{36}$	Set Connect $A^{47} \rightarrow A^{00}$. Prepare for left shift normalize.
	V687	Set K318/319	$A^{37} = A^{36} = A^{35}$	Set normalize Left Shift FF
34	N932	Set K314/315	$A^{37} = A^{36} = A^{35}$	

TIME	TERM	COMMAND	CONDITION	REMARKS
	V688	Set K408/409	$A^{37} \neq A^{36}$	Inhibit $A^1 \rightarrow A^2$
35	V688	Shift Right	$A^{37} \neq A^{36}$	Normalize right shift
	V689	Clear K318/319	A = 0	No left shift normalize if A = 0
	V689	Set K324/325	$A^{37} = A^{36}$	Set no right shift exit control
36	V690	Clear K408/409	$A^{37} \neq A^{36}$	
39	V689	Comp. $R^2 \rightarrow R^1$	$A^{37} \neq A^{36}$	
	V691	Clear K324/325		Clear the No Right Shift Exit Control FF
41	V693	$A^1 \rightarrow X^1$		Transfer the product to X^1 for final assembly
Execute Final Assembly				
	V693	Clear K510/511		Clear U^1 sign record
	V693	Set K404/405	A ≠ 0	Set the Not Inhibit Add FF
	V693	Set K512/513	A Neg.	Set X^1 sign record. Record the sign of final answer.
	V693	Set K520/521		Set the partial add in A FF. Prepare for final transfer of product to A.
42	V692	Comp. $X^1 \rightarrow X^2$	A Neg.	Form the positive produce in X^2 in preparation to add the exponent and bias
		$X^1 \rightarrow X^2$	A Pos.	
43	V693	$X^2 \rightarrow X^1$		Form the positive product in X^1 in preparation to adding the exponent and bias. Only necessary if the previous step was a Comp. $X^1 \rightarrow X^2$.
	V695	Clear K326/327		Clear the Connect A^{47} to A^{00} FF
		Set K582/583		Set Clear A. Prepare A to receive the final product
44	V694	Add $R^1 \rightarrow U^2$	Not inhibit Add FF = 1	Form the final exponent
45	V696	U^2 exponent to X^1		Form final exponent and bias in X^1. Final answer is in positive format.

TIME	TERM	COMMAND	CONDITION	REMARKS
	V699	Clear K512/513		Clear X^1 Sign Record FF
	V619	Clear K582/583		Clear Clear A FF
46	V696	$X^1 \rightarrow X^2$	X^1 sign record FF = 0 X^1 sign record FF = 1	Form the correct signed product in X^2 for transfer to A
	V619	Clear K404/405		Clear Not Inhibit Add FF
47	V699	EXIT		
	V699	Partial Add $X^2 \rightarrow A^1$	Not inhibit Add FF = 1	Transfer product to A
	V597	Set K584/585	Underflow fault	Clear A if underflow occurs
		Set K238/239	Overflow fault	Record overflow fault
48	V598	Set K246/247	Underflow fault	Record underflow fault
49	V599	Clear K584/585		Clear Clear A FF

CODE	INSTRUCTION	FUNCTION
33 FDV	Floating Divide	Divide a number packed, in floating point in A, by a number, also in floating point, from memory. Store the quotient in A, the residue in Q.

SEQUENCE: Iterative (H^{6--}, V^{6--})

EXECUTION TIME: 3.2 μsec min., 56.0 μsec avg., 57.2 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M (RNI)
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Prepare to form M (RNI)
01	V000	Clear X^1		
02	V602	Clear K474/475		Clear Dividend Sign FF
04	V602	Comp. $X^1 \rightarrow X^2$		Prepare to complement the dividend
	V602	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add b to m to form M
	V603	Set K602/603		Initiate storage at address M
	V604	Set K208/209		Set Wait Storage FF
05	V603	Partial Add $X^2 \rightarrow A^1$	A Neg.	Complement the dividend
	V605	Set K474/475	A Neg.	Record the sign of the dividend
	V605	Set K500/501	A Neg.	Record the sign of A
07	V605	$A^1 \rightarrow X^1$		Transfer the dividend to X^1
	V077	Storage Resume		
08	V607	$X^1_U \rightarrow X^2_U$		Used on the Extend Sign operation for a X^1 exp $\rightarrow U^2$ with sign extension
	V608	Set K090/091		Set I^2 to 36_{10}. This generates the shift count for the divide operation.
09	V607	Exit	A = 0	Exit since the dividend is zero.
	V609	Set K582/583		Clear A. Prepare to receive the divisor.
10	V609	X^1 exp $\rightarrow U^2$ with sign extension		Extract the exponent from the dividend

TIME	TERM	COMMAND	CONDITION	REMARKS
11	V609	$U^2 \rightarrow U^1$		Store the dividend exponent in the upper address portion of U^1
	V609	Clear X^1 Exp.		Clear bits 36 thru 47 in X^1 leaving the coefficient in X
	V611	Clear K582/583		Clear Clear A FF. Disables the clear on the A register.
12	V610	$X^1 \rightarrow X^2$		Transfer the coefficient to X^2 for transfer to Q
		Partial Add		
13	V611	Add $X^2 \rightarrow A^1$		Transfer the dividend coefficient to A^1
	V611	Clear X^1		Prepare X^1 to receive the divisor at M
	V073	Late Resume		
	V611	$A^2 \rightarrow Q^1$		Clear Q^1. Prepare Q to receive remainder
14	V612	$U^2 \rightarrow R^2$		Store the dividend exponent in R^2 to form the difference of the exponents
	V612	$Q^1 \rightarrow Q^2$		Clear Q^2
15	V844	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (M) to X^1
	V613	Comp. $R^2 \rightarrow R^1$		Prepare R to form the difference of the exponents
16	V620	$X^1 \rightarrow X^2$		Set $X^2 = X^1$ or transfer (M) $\rightarrow X^2$
	V648	Set K408/409		Inhibit the $A^1 \rightarrow A^2$ transfer
17	V623	To Floating Point Sequence		
	V623	Comp. Sign Record FF	X Neg.	Record the sign of the quotient
18	N934	Comp. Sign Record FF Slave	X Neg. at T = 17 SR = 1	

TIME	TERM	COMMAND	CONDITION	REMARKS
Floating Point Sequence (Time 00 equals Time 18 of main sequence)				
01	V660	Shift Right AQ		
02	V662	Clear K208/209		Iterative Wait Storage FF
04	V662	Comp. $X^1 \rightarrow X^2$	X Neg.	Make the divisor positive
05	V663	$X^2 \rightarrow X^1$		Set $X^1 = X^2$
06	V665	X^1 exp. $\rightarrow U^2$ (EXT)		Transfer the exponent to U^2 and extend the sign of the exponent
07	V665	Clear X^1 exp.		Leave the divisor's coefficient in X^1
	V665	$U^2 \rightarrow U^1$		Store the divisor exponent in the upper address portion of U^1
08	V666	Comp. $X^1 \rightarrow X^2$		Complement the coefficient of the divisor
12	V670	Add $R^1 \rightarrow U^2$		Determine the exponent of the quotient
	V672	Clear K450/451		Clear Disable Clear F
13	V671	Clear R^1		
	V673	Set K530/531		Set Partial add in U^2 FF
14	V672	$U^2 \rightarrow R^2$		Load the divide step control quantity in R^2
	V673	I^2 $I^3 \rightarrow R^1$		Transfer the shift count to R^1 ($^{36}10$)
	V674	Set K410/411		Set R ≠ 0 FF
15	V675			Enter the divide sequence and execute the divide
	V674	Set K410/411		Set R ≠ 0 slaves
16	V674	$R^1 \rightarrow R^2$		Prepare to reduce the shift count
17	V951	Clear K520/521		Enable a full add in A. Prepare for a difference of partial products.
18	V592	Set K510/511		Set U^2 Sign Record FF
	V646	Left Shift $A^1 \rightarrow A^2$		Left Shift A and Q
	V646	Left Shift $Q^1 \rightarrow Q^2$		

TIME	TERM	COMMAND	CONDITION	REMARKS
19	V647	$R^2 \rightarrow R^1$		Equalize shift count
	V886	$A^2 \rightarrow A^1$		Perform the division. Set Q^{00} to a "1" if $X \leq A$, to a "0" if $X > A$.
	V888	$Q^2 \rightarrow Q^1$		
	V644	Exit to T = 18	$R \neq 0$, $A < X$	Shift AQ left once after comparing X to A. Reduce R by one count for each shift. Exit to End Sign Correction when R = 0.
	V645	Exit to End Sign Correction	R = 0	
23	V644	Add $X^2 \rightarrow A^1$	$A \geq X$	
	V644	Set Q^{00} to 1	$A \geq X$	
	V644	Exit to T = 18	$R \neq 0$	
	V645	Exit to End Sign Correction	R = 0	
Execute End Correction				
24	V650	Set K232/233	Q Neg.	Set Divide Fault FF if Q^{47} = 1
		Clear K408/409		Clear Inhibit $A^1 \rightarrow A^2$ FF
25	V651	Set K520/521		
27	V651	Clear X^1		
28	V652	Comp. $X^1 \rightarrow X^2$		
29	V653	Add $X^2 \rightarrow A^1$	Div. Sign FF = 1	Complement remainder if dividend is negative
30	V654	$Q^1 \rightarrow Q^2$		Place quotient in A remainder in Q
31	V655	$Q^2 \rightarrow A^1$		
		$A^2 \rightarrow Q^1$		
33	V657	Part. Add $X^2 \rightarrow A^1$	Sign Record FF = 1	Complement quotient
34	V676	Exit to Round, Normalize, Final Assembly		

TIME	TERM	COMMAND	CONDITION	REMARKS
Execute Round (Times Contined as given in Volume 3, 1604-A)				
22	V676	Clear R^1		
		$U^2 \rightarrow U^1$		Prepare to form the sum of the normali: ing count and the exponent
23	V677	Set K506/507	A & Q signs unlike	Set the Execute Round FF if a significan bit is in Q^{47}
24	V676	$R^1 \rightarrow R^2$		Prepare to form normalizing count
25	V677	Clear X^1		Set $X^2 \rightarrow 1$
	V678	Comp. $R^2 \rightarrow R^1$		Prepare to form normalize count
26		Comp. $X^1 \rightarrow X^2$	A Neg.	Set X^2 to a negative 1
		$X^1 \rightarrow X^2$	A Pos.	Set X^2 to a positive 1
	V680	Set K070/071		Set X^2 to a 1
		Clear K530/531		Prepare to form the sum of the normalize count and the exponent
27	V951	Clear K520/521		Prepare to form the rounded quotient
	V680	$U^2 \rightarrow U^1$		
28	V681	$R^1 \rightarrow R^2$		Prepare for normalizing count
	V682	Clear K508/509		
		Clear Set X^2 to 1		
31	V683	Add $X^2 \rightarrow A^1$	Execute Round FF = 1	Form round quotient
Execute Normalize				
33	V686	Clear X^1		Prepare X for final assembly
	V687	Set K326/327	A^{37} = 36	Set Connect $A^{47} \rightarrow A^{00}$. Prepare for left shift normalize.
		Set K318/319	A37 = A36 = A35	Set Normalize Left Shift FF

TIME	TERM	COMMAND	CONDITION	REMARKS
	N932	Set K314/315	A37 = A36 = A35	
	V688	Set K408/409	A37 ≠ A36	Inhibit $A^1 \rightarrow A^2$
		Shift Right	A37 ≠ A36	Normalize right shift
	V689	Clear K318/319	A = 0	No Left Shift Normalize if A = 0
		Set K324/325	A37 = A36	Set No Right Shift Exit Control
36	V690	Clear K408/409	A37 ≠ A36	
39	V689	Comp. $R^2 \rightarrow R^1$	A37 ≠ A36	
	V691	Clear K324/325		Clear No Right Shift Exit Control FF
41	V693	$A^1 \rightarrow X^1$		Transfer the quotient to X^1 for final assembly
Execute Final Assembly				
41	V693	Clear K510/511		Clear U^1 Sign Record
	V693	Set K404/405	A ≠ 0	Set the Not Inhibit Add FF
	V693	Set K512/513	A Neg.	Set X^1 sign record. Record the sign of final answer.
	V693	Set K520/521		Set the partial add in A FF. Prepare for final transfer of quotient to A.
42	V692	Comp. $X^1 \rightarrow X^2$ $X^1 \rightarrow X^2$	A Neg. A Pos.	Form the positive quotient in X^2 in preparation to adding the exponent and bias
43	V693	$X^2 \rightarrow X^1$		Form the positive quotient in X^1 to add the exponent and bias. Only necessary if the previous step was a Comp. $X^1 \rightarrow X^2$.
	V695	Clear K326/327		Clear the Connect A^{47} to A^{00} FF
		Set K582/583		Set Clear A. Prepare A to receive the final quotient.
44	V694	Add $R^1 \rightarrow U^2$	Not inhibits Add FF = 1	Form the final exponent
45	V696	U^2_1 exponent to X^1		Form final exponent and bias in X^1. Final answer is in positive format.
	V699	Clear K512/513		Clear X^1 sign record FF
	V619	Clear K582/583		Clear Clear A FF

TIME	TERM	COMMAND	CONDITION	REMARKS
46	V696	$X^1 \rightarrow X^2$	X^1 sign record FF = 0	
		Comp. $X^1 \rightarrow X^2$	X^1 sign record FF = 1	Form the correct signed quotient in X^2 for transfer to A
	V619	Clear K404/405		Clear Not Inhibit Add FF
47	V699	EXIT		
	V699	Partial Add $X^2 \rightarrow A1$	Not inhibit Add FF = 1	Transfer quotient to A
	V597	Set K584/585	Underflow fault	Clear A if underflow occurs
		Set K238/239	Overflow fault	Record overflow fault
48	V598	Set K246/247	Underflow fault	Record underflow fault
49	V599	Clear K584/585		Clear Clear A FF

CODE	INSTRUCTION	FUNCTION
34 SCA	Scale A	Shift (A) left until the bit position to the right of the sign bit contain a "1". Store Mf in B^b.

SEQUENCE: Zero Address (H^{2--}, V^{2--})

2.8 usec min., 2.8 usec + .4 usec/shift avg.; 22 usec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Transfer normalize count (shift count k) to U^2 for transfer to R RNI
01	V201	Clear K520/521		Clear Partial Add in A FF
05	V205	Set K550/551		Prepare to clear B^b
06	V205	$U^2 \rightarrow R^2$		Transfer k to R
	V206	Set K410/411		Set R ≠ 0 FF. Initiate shift operation.
07	V207	Clear K550/550		Clear Clear B^b FF
	N991	Set K414/415		Set R = 0 FF slaves to terminage shifting
	N931	Set K412/413		
	V206	$R^2 \rightarrow R^1$		Equalize R for shift counting
10	V209	Initiate shift	$A^{47} = A^{46} \neq 0$	
	V220	Set K314/315	$R^2 \neq 0$ at T = 08	Set A Left FF. (See Shift Control, page 51, Volume 3.)
11	V209	$R^2 \rightarrow$ B Exit		Exit if the quantity in A is normalized or equal to zero
	V221	Set K320/321		Set shift exit control
12	N933	Reduce $R^1 \rightarrow R^2$		Equalize shift count
13	N570	Exit	R = 0	
	V223	Clear K320/321		Clear shift exit control

CODE	INSTRUCTION	FUNCTION
35 SCA	Scale AQ	Shift (AQ) left until the bit position to the right of the sign bit contains a "1". Store M in B^b.

SEQUENCE: Zero Address (H^{2--}, V^{2--})

2.8 usec min., 2.8 usec + 4 usec/shift avg., 41.2 usec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
	V099	$U^1 \rightarrow U^2$		Transfer normalize count (shift count k) to U^2 for transfer to R RNI
01	V201	Clear K520/521		Clear Partial Add in A FF
05	V205	Set K550/551		Prepare to clear B^b
06	V205	$U^2 \rightarrow R^2$		Transfer normalize count to R to be used as a shift count
	V206	Set K410/411		Set the R ≠ 0 FF
07	V207	Clear K550/551		Clear Clear B FF
	N991	Set K414/415		Set R = 0 FF slaves Used to terminate the shifting
	N931	Set K412/413		
	V206	$R^2 \rightarrow R^1$		Equalize R for shift counting
10	V209	Initiate Shift	$A^{47} = A^{46}$ AQ ≠ 0	
	V220	Set K314/315	$R^2 \neq 0$ at 08 time	Set A Left FF — Prepare to Scale a Left Shift (See Shift Control Logic, page 51, Vol. 3)
		Set K316/317		Set Q Left FF
11	V209	$R^2 \rightarrow B$ Exit		Exit if the quantity in AQ is normalized or equal to zero
	V221	Set K320/321		Set exit shift control
12	N933	Reduct $R^1 \rightarrow R^2$		Equalize shift count
13	K570	Exit	R = 0	
	V223	Clear K320/321		Clear shift exit control

CODE	INSTRUCTION	FUNCTION
36 SSK	Storage Skip*	Skip next instruction if (M) is negative

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 6.8 μsec min., (Upper Instruction), 8.8 μsec avg., 16 μsec avg. max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Transfer m to U^2
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	RNI
01	V000	Clear X^1		Clear X^1 to receive (M)
04	V302	Add $R^1 \rightarrow U^2$		Add b to m to form M
	V303	Set K602/603		Initiate storage
05	V304	Clear R^1		
10	V310	Set K202/203		Set Wait Storage 1 FF
13	V073 V075	Storage Resume		
14	V340	Clear K202/203		Clear Wait Storage 1 FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (M) to X^1
16	V342	Clear K550/551		Clear B^b
19	V370	Half Exit	X Pos.	Perform the next instruction
	V370	Full Exit	X Neg.	Skip the next instruction

* Instruction is usually restricted to the upper position

CODE	INSTRUCTION	FUNCTION
37 SSH	Storage Shift*	(M) Neg., full exit (skip next instruction) (M) Pos., half exit; in either case, shift (M) left one

SEQUENCE: Read Operand (H^{3--}, V^{3--})

10.8 usec min., 12.8 usec avg., 19.2 usec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	RNI
01	V000	Clear X^1		
04	V302	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
	V303	Set K602/603		Set Initiate Storage FF
05	V304	Clear R^1		
06	V304	$Q^1 \rightarrow Q^2$		
	V306	Set K580/581		Store A_i in Q^1; Clear A^1 for use as operation register
07	V305	$A^2 \rightarrow Q^1$		
10	V310	Set K202/203		Set Wait Storage I FF
13	V073 V075	Late Resume		
14	V340	Clear K580/581		Clear Clear A^1 FF
		Clear K202/203		Clear Wait Storage I FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Place (M) in X^1 for transfer to A
16	V340	$X^1 \rightarrow X^2$		Transfer (M) to X^2
18		Set K780/781	X Neg.	Set Exit Slave FF, Sense proper exit
		Set K486/487		Set f = 37 FF (conditions later command)
19	V345	Set K204/205		Set R.O Wait Storage FF
20	V346	Set K408/409		Set Inhibit $A^1 \rightarrow A^2$ FF, prepare for shift

TIME	TERM	COMMAND	CONDITION	REMARKS
21	V345	Part. Add $X^2 \rightarrow A^1$		Transfer (M) to A for shifting
		Clear X^1		
	V347	Clear K402/403	$R \neq 0$	Clear $R \neq 0$ FF to control 1 left shift
22	V346	Left Shift		Left shift A one place
		$A^1 \rightarrow A^2$		
23	V348	$A^2 \rightarrow A^1$		Transfer shifted quantity to A^1
25	V349	$A^1 \rightarrow X^1$	K204/205 is set	Transfer shifted (M) to X^1
27	V352	$Q^2 \rightarrow A^1$		Place Q_i in A^1
28	V352	$Q^1 \rightarrow Q^2$		Place A_i in Q^2
	V354	Clear K486/487		Clear F = 37 FF
	V353	Set K602/603		Set Initiate Storage FF. Prepare to restore shifted (M) to memory.
29	V353	$A_2^2 \rightarrow Q_1^1$ $Q^2 \rightarrow A^1$	}	Restore (A_i) and (Q_i)
45	V075	Late Resume		
47	J830	$X^1 \rightarrow Z^1\ Z^2$		Enable a full write
47	V360	Half Exit Full Exit	(M) Pos. (M) Neg. }	Exit determined at time 18 by K780/781

*Instructions usually restricted to upper position.

CODE	INSTRUCTION	FUNCTION
40 SST	Selective Set	Set bits of (A) to "1's" corresponding to "1's" of (M)

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 4.8 usec min., 7.2 usec avg., 9.6 usec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M RNI
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	
01	V000	Clear X^1		
04	V302	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
	V303	Set K602/603		Set Initiate Storage FF
05	V304	Clear R^1		
08	V308	Clear K580/581		Clear Clear A^1 FF
10	V310	Set K202/203		Set Wait Storage 1 FF
13	V073	Late Resume		
	V075			
14	V340	Clear K202/203		Clear Wait Storage 1
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Superimpose (M) and (A); "1's" in either or both words will cause corresponding bits in the combined word to be 1.
	V340	$A^1 \rightarrow X^1$		
	V341	Set K580/581		Set Clear A FF
16	V340	$X^1 \rightarrow X^2$		Transfer X^1 to X^2 for transfer to A
19	V370	Exit		
21	V345	Part. Add $X^2 \rightarrow A^1$		Transfer (X) to A

CODE	INSTRUCTION	FUNCTION
41 SCL	Selective Clear	Clear bits of (A) corresponding to "1" of M

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M (RNI)
	V871	$I^2\ I^3 \rightarrow X^1$	$b \neq 0$	
01	V000	Clear X^1		
04	V302	Add R^1 to U^2	$b \neq 0$	Add (B^b) to m to form M
	V303	Set K602/603		Initiate Storage
05	V304	Clear R^1		
06	V304	Comp. $X^1 \rightarrow X^2$		Complement (A) - this sets up proper $A^1 \rightarrow X^1$ transfer
07	V305	Part. Add $X^2 \rightarrow A^1$		
08	V308	Clear K580/581		Clear Clear A^1 FF
10	V310	Set K202/203		Set Wait Storage I FF
13	V073	Late Resume		
	V075			
14	V340	Clear K202/203		Clear Wait Storage I FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Superimpose (M) and (A); "1's" in either or both words will cause corresponding bits in the combined word to be "1".
	V340	$A^1 \rightarrow X^1$		
	V341	Set K580/581		Set Clear A FF to receive result in A
16	V340	Comp. $X^1 \rightarrow X^2$		Set result to proper order in X^2; bits corresponding to "1"s in (M) are now "0"
19	V370	Exit		
21	V345	Part. Add $X^2 \rightarrow A^1$		Transfer (X^2) to A

CODE	INSTRUCTION	FUNCTION
42 SCM	Selective Complement	Complement bits of (A) corresponding to "1" of (M)

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec min.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M (RNI)
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	
01	V000	Clear X^1		
04	V302	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
	V303	Set K602/603		Initiate Storage
05	V304	Clear R^1		
08	V308	Clear K580/581		Clear Clear A^1 FF
10	V310	Set K202/203		Set Wait Storage I FF
13	V073 V075	Late Resume		
14	V340	Clear K202/203		Clear Wait Storage I FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (M) $\rightarrow X^1$
16	V340	$X^1 \rightarrow X^2$		Equalize the X register
19	V370	Exit		
21	V345	Part. Add $X^2 \rightarrow A^1$		Complement (A) for corresponding "1" in M

CODE	INSTRUCTION	FUNCTION
43 SSU	Selective Substitute	Replace bits in (M) by bits in A corresponding to "1's" in (Q)

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 5.2 μsec min., 7.4 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Prepare to form M (RNI)
	V099	$U^1 \rightarrow U^2$		
01	V000	Clear X^1		
04	V302	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
05	V304	Clear R^1		
06	V304	Comp. $X^1 \rightarrow X^2$		Prepare to complement (A) by a Partial Add $X^2 \rightarrow A^1$ and perform LQX
		$Q^1 \rightarrow Q^2$		Equalize Q for a LQX
07	V305	$X^2 \rightarrow X^1$		Complement A and X
		Part. Add $X^2 \rightarrow A^1$		
	V307	Set K540/541		Set LQX FF. Logical Transfer Q to X^1
09	V309	Clear K540/541		Clear LQX FF
10	V310	Set K202/203		Set Wait Storage FF
11	V311	Set K580/581		Set Clear A^1 FF
	V309	$A^1 \rightarrow X^1$		Superimposes A on X^1; selectively clears comp. of A_i by forcing bits to "1" if Q = 1
12	V310	Comp. $X^1 \rightarrow X^2$		X^1 equals the selectively cleared A
13	V073	Late Resume		
	V075			
	V311	Clear X^1		Prepare X^1 to receive (M)

TIME	TERM	COMMAND	CONDITION	REMARKS
14	V340	Clear K502/503		Clear Wait I Storage FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (M) $\rightarrow X^1$
	V341	Set K540/541		Set LQX FF. Mask (M) for "0's" in Q
	V341	Set K482/483		Set f = 43-46 FF. This conditions later commands.
17	V342	Part. Add $X^2 \rightarrow A^1$		Transfer selected bits of memory into selectively cleared bits of A
	V343	Clear K540/541		Clears LQX FF
21	V346	Exit		
25	V349	Part. Add $X^2 \rightarrow A^1$		Substitute masked M for cleared bits of A_i
28	V354	Clear K482/483		Clear f = 43-46 FF

CODE	INSTRUCTION	FUNCTION
44 LDL	Load Logical	Load the logical product of (Q) and (M) in A

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 5.2 μsec min., 7.4 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M (RNI)
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	
01	V000	Clear X^1		
04	V302	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
	V303	Set K602/603		Initiate storage
05	V304	Clear R^1		
06	V304	$Q^1 \rightarrow Q^2$		Equalize Q^1 and Q^2 for logical multiply
	V306	Set K580/581		Clear A^1 to receive logical product
10	V310	Set K202/203		Set Wait Storage 1 FF
13	V073 V075	Late Resume		
14	V340	Clear K202/203		Clear Wait Storage 1 FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		
	V341	Set K540/541 Set K482/483		Form logical product of (Q) and (M). See X register and X register control (Vol. 3) for actual operation.
17	V343	Clear K540/541		Clear LQX FF
20	V344	$X^1 \rightarrow X^2$		Equalize X register
21	V346	Exit		
25	V349	Part. Add $X^2 \rightarrow A^1$		Load logical product in A
28	V354	Clear K482/483		Clear f = 43-46 FF

CODE	INSTRUCTION	FUNCTION
45 ADL	Add Logical	Add the logical product of (Q) and (M) to (A_i); store the sum in A.

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 5.4 μsec min., 7.4 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M (RNI)
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	
01	V000	Clear X^1		
04	V302	Add $R^1 \rightarrow U^2$		Add (B^b) to m to form M
	V303	Set K602/603		Initiate Storage
05	V304	Clear R^1		
06	V304	$Q^1 \rightarrow Q^2$		Equalize Q^1 and Q^2 for logical multiply
10	V310	Set K202/203		Set Wait Storage 1 FF
13	V073	Late Resume		
	V075			
		Clear K202/203		Clear Wait Storage 1 FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Form logical product of (Q) and (M). See X register and X register control (Vol. 3) for actual operation.
	V341	Set K540/541		
		Clear K520/521		
		Set K482/483		
17	V343	Clear K540/541		Clear LQX FF
20	V344	$X^1 \rightarrow X^2$		Equalize X registers
21	V346	Exit		
25	V349	Add $X^2 \rightarrow A^1$		Add LQX to (A) - This is final sum.
28	V354	Clear K482/483		Clear f = 43-46 FF

CODE	INSTRUCTION	FUNCTION
46 SBL	Subtract Logical	Subtract the logical product of (Q) and (M) from (A_i); store the difference in A.

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 5.4 μsec min., 7.4 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M (RNI)
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	
01	V000	Clear X^1		
04	V302	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
	V303	Set K602/603		Initiate Storage
05	V304	Clear R^1		
06	V304	$Q^1 \rightarrow Q^2$		Position Q for logical product
10	V310	Set K202/203		Set Wait Storage 1 FF
13	V073 V075	Late Resume		
		Clear K202/203		Clear Wait Storage 1 FF
15	V341	Clear K520/521		Clear Partial Add in A FF
	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Form logical product of (Q) and (M). See X register and X register control (Vol. 3) for actual operation.
	V341	Set K540/541		
		Set K482/483		
17	V343	Clear K540/541		Clear LQX FF
20	V344	Comp. $X^1 \rightarrow X^2$		Prepare for a subtract logical
21	V346	Exit		
25	V349	Add $X^2 \rightarrow A^1$		Subtract LQX from A
28	V354	Clear K482/483		Clear f = 43-46 FF

CODE	INSTRUCTION	FUNCTION
47 STL	Store Logical	Store the logical product of (Q) and (A) at M

SEQUENCE: Write Operand (H^{4--}, V^{4--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M (RNI)
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	
01	V000	Clear X^1		
04	V402	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add (B^b) to m to form M
	V403	Set K602/603		Initiate Storage
06	V404	$Q^1 \rightarrow Q^2$		Form the logical product of (Q) and (A). See X register and X register control (Vol. 3) for actual operation.
07	V405	$A^1 \rightarrow X^1$		
	V407	Set K540/541		
	V407	Set K206/207		Set Write Operand Wait Storage FF
08	J830	Enable Full Write		Use memory cycle to prepare to store L (Q) (A) at M
09	V409	Clear K540/541		Clear LQX FF
13	V073	Late Resume		
15		$X^1 \rightarrow Z^1\ Z^2$		Store L (Q) (A) at M
	V507	Exit		

CODE	INSTRUCTION	FUNCTION
50 ENI	Enter Index; b ≠ 0	Enter the base execution address in B^b. Becomes a Pass Instruction if b = 0.

SEQUENCE: Zero Address (H^{2--}, V^{2--})

EXECUTION TIME: 3.2 μsec min., 5.6 μsec avg., 8.2 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Transfer y to U^2 } RNI
00	V871	$I^2\ I^3 \rightarrow R^1$	b ≠ 0	Transfer B^b to R } RNI
05	V205	Set K550/551		Set Clear B FF. Clear B for receipt of y
06	V205	$U^2 \rightarrow R^2$		Transfer y → R^2
07	V207	Clear K550/551		Clear Clear B FF
08	V207	Set K570/571		Set R^2 → B FF. Transfer y to B^b.
09	V207	Exit		

CODE	INSTRUCTION	FUNCTION
51 INI	Increase index; b ≠ 0	Add the base execution address y to B^b, store the sum Y in B^b. Becomes a Pass Instruction if b = 0.

SEQUENCE: Zero Address (H^{2--}, V^{2--})

EXECUTION TIME: 3.2 μsec min., 5.6 μsec avg., 8.2 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Transfer y to U^2
	V871	$I^2\ I^3 \rightarrow R^1$	b ≠ 0	Transfer B^b to R^1 for modification } RNI
04	V203	Add $R^1 \rightarrow U^2$	b ≠ 0	Add (B^b) to y to make Y
05	V205	Set K550/551		Set Clear B FF. Clear B to receive Y.
06	V205	$U^2 \rightarrow R^2$		Transfer Y to R^2
07	V207	Clear K550/551		Clear Clear B FF
08	V207	Set K570/571		Set $R^2 \rightarrow$ B FF. Transfer Y to B^b.
09	V207	Exit		

CODE	INSTRUCTION	FUNCTION
50 ENI	Enter Index; b ≠ 0	Enter the base execution address in B^b. Becomes a Pass Instruction if b = 0.

SEQUENCE: Zero Address (H^{2--}, V^{2--})

EXECUTION TIME: 3.2 μsec min., 5.6 μsec avg., 8.2 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Transfer y to U^2 } RNI
00	V871	$I^2\ I^3 \rightarrow R^1$	b ≠ 0	Transfer B^b to R } RNI
05	V205	Set K550/551		Set Clear B FF. Clear B for receipt of y
06	V205	$U^2 \rightarrow R^2$		Transfer y → R^2
07	V207	Clear K550/551		Clear Clear B FF
08	V207	Set K570/571		Set R^2 → B FF. Transfer y to B^b.
09	V207	Exit		

CODE	INSTRUCTION	FUNCTION
51 INI	Increase index; b ≠ 0	Add the base execution address y to B^b, store the sum Y in B^b. Becomes a Pass Instruction if b = 0.

SEQUENCE: Zero Address (H^{2--}, V^{2--})

EXECUTION TIME: 3.2 μsec min., 5.6 μsec avg., 8.2 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Transfer y to U^2 } RNI
	V871	$I^2\ I^3 \rightarrow R^1$	b ≠ 0	Transfer B^b to R^1 for modification } RNI
04	V203	Add $R^1 \rightarrow U^2$	b ≠ 0	Add (B^b) to y to make Y
05	V205	Set K550/551		Set Clear B FF. Clear B to receive Y.
06	V205	$U^2 \rightarrow R^2$		Transfer Y to R^2
07	V207	Clear K550/551		Clear Clear B FF
08	V207	Set K570/571		Set $R^2 \rightarrow$ B FF. Transfer Y to B^b.
09	V207	Exit		

CODE	INSTRUCTION	FUNCTION
50 ENI	Enter Index; b ≠ 0	Enter the base execution address in B^b. Becomes a Pass Instruction if b = 0.

SEQUENCE: Zero Address (H^{2--}, V^{2--})

EXECUTION TIME: 3.2 μsec min., 5.6 μsec avg., 8.2 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Transfer y to U^2 } RNI
00	V871	$I^2\ I^3 \rightarrow R^1$	b ≠ 0	Transfer B^b to R } RNI
05	V205	Set K550/551		Set Clear B FF. Clear B for receipt of y
06	V205	$U^2 \rightarrow R^2$		Transfer y → R^2
07	V207	Clear K550/551		Clear Clear B FF
08	V207	Set K570/571		Set R^2 → B FF. Transfer y to B^b.
09	V207	Exit		

CODE	INSTRUCTION	FUNCTION
51 INI	Increase index; b ≠ 0	Add the base execution address y to B^b, store the sum Y in B^b. Becomes a Pass Instruction if b = 0.

SEQUENCE: Zero Address (H^{2--}, V^{2--})

EXECUTION TIME: 3.2 μsec min., 5.6 μsec avg., 8.2 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Transfer y to U^2
	V871	$I^2\ I^3 \rightarrow R^1$	b ≠ 0	Transfer B^b to R^1 for modification } RNI
04	V203	Add $R^1 \rightarrow U^2$	b ≠ 0	Add (B^b) to y to make Y
05	V205	Set K550/551		Set Clear B FF. Clear B to receive Y.
06	V205	$U^2 \rightarrow R^2$		Transfer Y to R^2
07	V207	Clear K550/551		Clear Clear B FF
08	V207	Set K570/571		Set $R^2 \rightarrow$ B FF. Transfer Y to B^b.
09	V207	Exit		

CODE	INSTRUCTION	FUNCTION
52 LIU	Load Index Upper	Replace (B^b) with the upper address of the word specified by the base execution address.

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Transfer m → U^2 } RNI
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Transfer $B^b \rightarrow R^1$ } RNI
01	V000	Clear X^1		Prepare X register for use as an exchange register
04	V304	Set K602/603		Set Initiate Storage FF
	V304	Clear R^1		Clear R^1 to receive (M_{UA})
12	V310	Set K202/203		Set Wait Storage FF
13	V073	Storage Resume		
14	V340	Set K550/551		Set Clear B FF. Clear B to receive (M_{UA}).
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (m) to X^1
16	V342	Clear K550/551		Clear Clear B FF
17	V343	Set K484/485		Set f = 52 + 53 FF. Conditions later commands.
18	V344	Set K092/K093		Set $X^1_u \rightarrow I^2$ FF. Transfer X^1_u to I^2
20	V344	$I^2\ I^3 \rightarrow R^1$		Transfer X^1_u to R^1
22	V346	$R^1 \rightarrow R^2$		Equalize R^1 and R^2
	V347	$R^2 \rightarrow$ B Exit		
	V347	Set K570/571		Set $R^2 \rightarrow$ B FF. Transfer (M_{UA}) to B.
23	V347	Exit		
27	V353	Clear K484/485		Clear f = 52 + 53 FF

CODE	INSTRUCTION	FUNCTION
53 LIL	Load Index Lower	Replace (B^b) with the lower address of the word specified by the base execution address

SEQUENCE: Read Operand (H^{3--}, V^{3--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Transfer m $\rightarrow U^2$ } RNI
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Transfer $B^b \rightarrow R^1$ } RNI
01	V000	Clear X^1		
04	V304	Set K602/603		Set Initiate Storage FF
	V304	Clear R^1		Clear R^1 to receive (M_{LA})
12	V310	Set K202/203		Set Wait Storage FF
13	V073	Storage Resume		
14	V340	Set K550/551		Set Clear B FF. Clear B to receive (M_{LA}).
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (m) to X^1
16	V341	$X^1{}_L \rightarrow X^2{}_U$		Transfer (M_{LA}) to $X^2{}_U$
17	V342	Clear K550/551		Clear Clear B FF
	V342	$X^2{}_U \rightarrow X^1{}_U$		Transfer (M_{LA}) to $X^1{}_U$
	V343	Set K484/485		Set f = 52 + 53 FF conditions later commands
18	V344	Set K092/K093		Set $X^1{}_U \rightarrow I^2$ FF. Transfer $X^1{}_U$ to I^2
20	V344	$I^2\ I^3 \rightarrow R^1$		Transfer $X^1{}_U \rightarrow R^1$
22	V346	$R^1 \rightarrow R^2$		Equalize R^1 and R^2
	V347	$R^2 \rightarrow$ B Exit		
	V347	Set K570/571		Set $R^2 \rightarrow$ B FF. Transfer (M_{LA}) to B

TIME	TERM	COMMAND	CONDITION	REMARKS
23	V347	Exit		
27	V353	Clear K484/485		Clear f = 52 + 53 FF

CODE	INSTRUCTION	FUNCTION
54 ISK	Index Step*	$(B^b) = (y)$, Clear B^b and full exit; $(B^b) \neq (y)$, $[(B^b) + 1] \rightarrow B^b$ and half exit.

SEQUENCE: Zero Address (H^{2--}, V^{2--})

EXECUTION TIME:

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to compare (B^b) with base execution address y — RNI
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	
	V201	Set K530/531		Set Partial Add in U FF. Prepare to compare U^2 with R^1.
02	V002	$R^1 \rightarrow R^2$		Equalize R^1 and R^2
04	V203	Part. Add $R^1 \rightarrow U^2$	$b \neq 0$	Subtract (B^b) from y, load the difference in R^2.
05	V203	Comp. $R^2 \rightarrow R^1$		Transfer the complement of R^2 to R^1. Prepare to increase R.
	V205	Set K550/551		Set Clear B FF
06	V206	$U^2 \rightarrow R^2$		Transfer U^2 to R^2. A R = 0 check will be performed from R^2 to see if (B^b) is equal to (U^2).
	V206	Set K410/411		Set R ≠ 0 FF
07	N991	Set K414/415		Set R = 0 slaves
	N931	Set K412/413		Set R = 0 slaves
	V207	Clear K550/551		Clear Clear B FF
08	N570	R = 0		
	N570	Clear K410/411	R = 0 at T = 07	This prevents an $R^2 \rightarrow B$ later in the sequence if $R = U^2$
09	N570	Full Exit		
	N931	Clear K412/413	R = 0 at T = 07	
	N991	Clear K414/415		Clear R = 0 FF slaves

TIME	TERM	COMMAND	CONDITION	REMARKS
10	V208	$R^1 \rightarrow R^2$	$R \neq 0$ at T = 07	Prepare to increase R
	V210	Clear 410/411	$R \neq 0$ at T = 07	Clear R = 0 FF if it was not cleared at T = 08
12	V211	Reduce $R^1 \rightarrow R^2$	$R \neq 0$ at T = 07	Reduce (R) by one (this increases R_i by one)
13	V211	Comp. $R^2 \rightarrow R^1$	$R \neq 0$ at T = 07	Express (R) in non-complement form
14	V213	$R^1 \rightarrow R^2$	$R \neq 0$ at T = 07	Load (R) in B^b
	V214	Set K570/571	$R \neq 0$ at T = 07	
15	V214	Half Exit	$R \neq 0$ at T = 07	

*Ordinarily this instruction is limited to the upper instruction position.

CODE	INSTRUCTION	FUNCTION
55 LJP	Index Jump	$(B^b) \neq 0$: $[(B^b)-1] \rightarrow B^b$, Execute NI at m $(B^b) = 0$: Execute next instruction

SEQUENCE: Zero Address (H^{2--}, V^{2--})

EXECUTION TIME:

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Transfer m to U^2 } RNI
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Transfer B^b to R^1 } RNI
02	V002	$R^1 \rightarrow R^2$		Equalize R^1 and R^2. R^2 contains B^b.
	V202	Set K410/411	$R \neq 0$	Set $R \neq 0$ FF. Conditions later commands.
03	N991	Set K414/415		$R \neq 0$ FF slaves are set the next odd time after K410/411 is set at T = 02
	N931	Set K412/413		
04	N570	Clear K410/411		Clear $R \neq 0$ FF
05	V205	Set K550/551		Set Clear B FF. Clear B to receive modified (R).
	V205	$U^2 \rightarrow P^1$	$R \neq 0$	Transfer m to P
07	V205	Exit (half or full)	$R = 0$	Execute next instruction
	V206	Jump Exit	$R \neq 0$	Execute next instruction at m
08	V207	Reduce $R^1 \rightarrow R^2$	$R \neq 0$	Reduce R^1 by 1 and transfer to R^2
	V208	Set K570/571		Set $R^2 \rightarrow$ B FF. Transfer the modified (R^1 to B^b).

CODE	INSTRUCTION	FUNCTION
56 SIU	Store Index (Upper)	Store (B^b) in upper address of the location specified by the base execution address (m)

SEQUENCE: Write Operand (H^{4--}, V^{4--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS	
00	V099	$U^1 \rightarrow U^2$		Transfer m to U^2	
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Transfer B^b to R^1 to transfer to (m) later on	RNI
	V000	Set K206/207		Set Wait Storage FF	
		Set K602/603		Set Initiate Storage FF. Initiate storage at U^2.	
01		Clear X^1		Prepare X register for use as an exchange register	
	V006	Set K450/451		Set Disable Clear f FF. This retains the Function code.	
03		Clear $U^1{}_U$			
	V077	Storage Resume			
04	V077	$U^1 \rightarrow U^2$		Clear U^2	
	V406	Set K530/531		Enable Partial Add in U^2 to transfer (R^1) to U^2	
06		Partial Add $R^1 \rightarrow U^2$		Transfer (R^1) to U^2	
	V407	Enable Partial Write, Upper			
07		$U^2 \rightarrow X_{LA}$		Transfer B^b to X_{LA} with sign extended	
08	V410	Clear K450/451		Clear Disable Clear FF	
		Clear K694/695		Clear Interrupt or Clock Request FF	
09	V971	$X^1{}_L \rightarrow X^2{}_U$ Clear $X^1{}_L$		Position the quantity for proper storage into memory	

TIME	TERM	COMMAND	CONDITION	REMARKS
	V073	Storage Resume		
10	V420	Clear K206/207		Clear wait storage
11	V420	Exit		
		$X^1_U \rightarrow Z^1\ Z^2$		Store (B^b) at m

CODE	INSTRUCTION	FUNCTION
57 SIL	Store Index (Lower)	Store (B^b) in the lower address of the location specified by the base execution address

SEQUENCE: Write Operand (H^{4--}, V^{4--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Transfer m to U^2
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Transfer B^b to R^1 to transfer to (m) later on
	V000	Set K206/207		Set Wait Storage FF
		Set K602/603		Set Initiate Storage FF. Initiate storage at U^2.
01		Clear X^1		Prepare X register for use as an exchange register
	V006	Set K450/451		Set Disable Clear f FF. This retains the Function code.
03		Clear U^1_U		
	V077	Storage Resume		
04	V077	$U^1 \rightarrow U^2$		Clear U^2
	V406	Set K530/531		Enable Partial Add in U^2 to transfer R^1 to U^2
06		Partial Add $R^1 \rightarrow U^2$		Transfer (R^1) to U^2
	V407	Enable Partial Write Lower		
07	V407	$U^2 \rightarrow X^1_{LA}$		Transfer B^b to X_{LA} with sign extended
08	V410	Clear K450/451		Clear Disable Clear FF
		Clear K694/695		Clear Interrupt or Clock Request FF

RNI

TIME	TERM	COMMAND	CONDITION	REMARKS
09	V971	$X^1_L \rightarrow X^2_U$		Position the quantity for proper storage into memory
		Clear X^1_L		
	V073	Storage Resume		
10	V420	Clear K206/207		Clear Wait Storage
11	V420	Exit		
		$X^1_U \rightarrow Z^1 + Z^2$		Store (B^b) at m

CODE	INSTRUCTION	FUNCTION
60 SAU	Substitute Address (Upper)	Replace the upper address M (M_{38} - M_{24}) with the lowest bits of A (A_{14} - A_{00})

SEQUENCE: Write Operand (H^{4--}, V^{4--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Transfer B^b to R^1
01	V000	Clear X^1		Clear X^1 to use it as an exchange register
04	V402	Add R^1 to U^2	$b \neq 0$	Add B^b to m to form M
	V403	Set K602/603		Initiate Storage at U^2
07	V405	$A^1 \rightarrow X^1$		Transfer (A^1) to (X^1) to be transferred to M later on
	V407	Set K206/207		Set write operand wait storage
08		Enable Partial Write Upper		
12	V971	$X^1{}_{LA} \rightarrow X^2{}_U$		
13	V971	$X^2{}_U \rightarrow X^1{}_U$		Transfer (A_{14} - A_{00}) to $X^1{}_U$
		Clear $X^1{}_L$		
	V073	Storage Resume		
14	V420	Clear K206/207		Clear write operand wait storage
15		Exit		
		Set Z		Transfer $X^1 \rightarrow I^5\ I^6\ I^7 \rightarrow Z^1 + Z^2$ to memory

RNI

CODE	INSTRUCTION	FUNCTION
61 SAL	Substitute Address (Lower)	Replace the lower address M (M_{14} - M_{00}) with the lowest bits of A (A_{14} - A_{00})

SEQUENCE: Write Operand (H^{4--}, V^{4--})

EXECUTION TIME: 4.8 μsec min., 7.2 μsec avg., 9.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Transfer B^b to R^1
01	V000	Clear X^1		Clear X^1 to use it as an exchange register
04	V402	Add $R^1 \rightarrow U^2$	$b \neq 0$	Add B^b to m to form M
	V403	Set K602/603		Initiate Storage at U^2
07	V405	$A^1 \rightarrow X^1$		Transfer (A^1) to (X^1) to be transferred to M later on
	V407	Set K206/207		Set Write Operand Wait Storage
08		Enable Partial Write Lower		Write the lower 15 bits of X^1 into memory
13	V073	Storage Resume		
14	V420	Clear K206/207		Clear Write Operand Wait Storage
15		Exit		
		Set Z		Transfer $X^1 \rightarrow I^5\ I^6\ I^7 \rightarrow Z^1 + Z^2$ to memory

RNI

CODE	INSTRUCTION	FUNCTION
62 INT	Input Transfer	Transfer (B^b) words to storage beginning at m + (B^b - 1)

SEQUENCE: Search and Transfer Sequence (H^{5--}, V^{5--})

EXECUTION TIME:

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	$I^2\ I^3 \rightarrow R^1$	b ≠ 0	Prepare to form the Transfer Address
	V099	$U^1 \rightarrow U^2$		
01	V000	Clear X^1		
02	V560	Set K410/411		Set R ≠ 0 FF. Check if transfer will be made.
		Clear K416/417		Clear ST $\overline{\text{Complete}}$ FF
04	V503	Reduce $R^1 \rightarrow R^2$	b ≠ 0	Prepare to form first transfer address
	N570	Clear K410/411	$R^2 = 0$	Disable ST $\overline{\text{Complete}}$ FF if no transfer is to be made
05	V503	$R^2 \rightarrow R^1$	b ≠ 0	Transfer count to R^1 to form transfer address in U^2
	V505	Set K416/417	b = 0	Records that one transfer is to be made
	V561	Set K416/417	$R^2 \neq 0$ at T = 04	Records that a transfer is being executed
07	V507	Set K770/771	ST $\overline{\text{Complete}}$ FF set	Enable an input transfer on receipt of an input transfer ready
08	N946	Into H789	Interrupt And Clock Request	Into interrupt sequence
				The following timing assumes an Input Transfer Ready signal was present at T = 06. If the Ready was not present at T = 06 the sequence would just halt and wait for an Input Transfer Ready signal. Only an interrupt or clock request could break in on the 62.

TIME	TERM	COMMAND	CONDITION	REMARKS
08	N948	Set K880/881		Synchronize Input Transfer Ready signal to the computer timing
09	V508	Exit	ST $\overline{\text{Complete}}$ FF = 0	Exit if no transfer is to be made
	N949	Set K832/833		Enable continuing of sequence
10	N946	Into H509		Continue Sequence
	V508	Add $R^1 \rightarrow U^2$	$b \neq 0$	Form transfer address
11	V521	Clear K770/771		Clear the Input Transfer Active FF. Disable the input to H509.
		Set K210/211		Set Wait Storage
12	V856	Set K602/603		Initiate Storage
15	V523	Clear K762/763		Send out Input Transfer to external equipment
16	V560	Set K410/411		Prepare to check if another word is to be transferred
		Clear K416/417		
18	V524	$U^1 \rightarrow U^2$		Prepare to form next transfer address
		$X^1 \rightarrow X^2$		Prepare X^2 to receive input word
	V526	Clear K570/571		Clear $R^2 \rightarrow B^b$ FF
		Set K330/331		Set the $M^0 \rightarrow X^2$ FF enable input word to X^2
19	V561	Set K416/417		Set ST $\overline{\text{Complete}}$ FF. Check if another transfer is to be executed.
20	V527	Reduce $R^1 \rightarrow R^2$		Prepare to form next address
21		$R^2 \rightarrow R^1$		
		$X^2 \rightarrow X^1$		Transfer input word to X^1 to be written into memory
	V075	Late Resume		

TIME	TERM	COMMAND	CONDITION	REMARKS
23	V541	Set K770/771	ST $\overline{\text{Complete}}$ FF = 1	Records that another transfer is to be executed
24	V542	Clear K210/211		Clear Wait Storage
	V948			Return to time T = 08 if another transfer is to be made
26	V523	Add $R^1 \rightarrow U^2$		Form next transfer address
31	V548	Exit	Transfer Complete	Transfer terminated; go to next instruction.

CODE	INSTRUCTION	FUNCTION
63 OUT	Output Transfer	Transfer (B^b) from storage beginning at m + (B^b - 1)

SEQUENCE: Search and Transfer (H^{5--}, V^{5--})

EXECUTION TIME:

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	I^2 $I^3 \rightarrow R^1$	b ≠ 0	Prepare to form the Search Address
	V099	$U^1 \rightarrow U^2$		
01	V000	Clear X^1		
02	V560	Set K410/411		Set R ≠ 0 FF. Prepare to check if transfer will be made.
		Clear K416/417		Clear ST $\overline{\text{Complete}}$ FF
04	V503	Reduce $R^1 \rightarrow R^2$	b ≠ 0	Subtract first count from the contents of the Index Register
	N570	Clear K410/411	R^2 ≠ 0 slave = 1	Clear R ≠ 0 FF, if no transfer is to be made
05	V503	$R^2 \rightarrow R^1$	b ≠ 0	Prepare to add $R^1 \rightarrow U^2$
	V561	Set K416/417	R ≠ 0 FF = 1	Set the ST $\overline{\text{Complete}}$ FF if a transfer is to be made
		Set K416/417	b = 0	
08	V508	Into H509	ST $\overline{\text{Complete}}$ FF = 1	Continue transfer sequence
09		Full Exit	ST $\overline{\text{Complete}}$ FF = 0 and Exit FF = 0	
		Half Exit	ST $\overline{\text{Complete}}$ FF = 0 and Exit FF = 1	
	V521	Set K210/211		Set Wait Storage FF
10	V508	Add $R^1 \rightarrow U^2$	b ≠ 0	Form the first transfer address
	V856	Set K602/603		Initiate storage

TIME	TERM	COMMAND	CONDITION	REMARKS
13	V523	Clear K550/551		Clear Clear B FF
14	V560	Set K410/411		Set R ≠ 0 FF
	V560	Clear K416/417		Clear ST $\overline{\text{Complete}}$ FF; prepare to check if another word will be transferred.
16	N570	Clear K410/411	R^2 = 0 and R ≠ 0 slave set	Clear the R ≠ 0 FF. Indicates that the transfer is complete.
	V524	$U^1 \rightarrow U^2$		Prepare to form second transfer address
17	V561	Set K416/417	R ≠ 0 FF = 1	Records that another word has to be transferred
18	V527	Reduce $R^1 \rightarrow R^2$		Prepare to form next transfer address
	V073 V075	Storage Resume		
21	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Storage $\rightarrow X^1$; prepare to transfer the word to Output register
	V541	Set K772/773		Set the Output Transfer Active FF. Records that the transfer is in operation and enables continuation of the sequence.
22	V542	Clear K210/211		Clear Wait Storage
	N946	Into H543		If there is no Output Transfer Ready and no Output Transfer Resume, initiate this transfer. The sequence could stop at this point and wait for the Resume signals from the previous word transferred.
	V540			Prepare for transfer to Output register
23	V543	Clear K772/773		Clear Output Transfer Active FF. Disable the input to H543.
	V543	Set K080/081		Set $X^2 \rightarrow O^4$ FF. Transfer the word to the Output Register.
24	V532	Add $R^1 \rightarrow U^2$		Form the next word to be transferred out

TIME	TERM	COMMAND	CONDITION	REMARKS
	V544	Into H521	Transfer Complete	Return to T = 09
25	V545	Clear K080/081		Clear the $X^2 \rightarrow O^4$ FF
		Set K766/767		Set Output Transfer Ready FF. Indicates to external device that the word is available.
29	V548	Exit	Transfer Complete	

CODE	INSTRUCTION	FUNCTION
64 EQS	Equality Search	Search (B^b) words beginning with m + B^b - 1 for (M) = (A). Exit if (M) = (A), Half Exit if (M) ≠ (A)

SEQUENCE: Search and Transfer (H^{5--}, V^{5--})

EXECUTION TIME: 3.6 μsec min., 4.0 + 3.6r* avg., 6.8 + 3.6r* max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	I^2 I^3 → R^1	b ≠ 0	Prepare to form M
	V099	U^1 → U^2		
01	V000	Clear X^1		Prepare X to receive operand from M
02	V560	Set K410/411		Set the R ≠ 0 FF. This FF determines if the search will be made. If R is equal to 0, no search is made and N570 clears this FF.
		Clear K416/417		Clear ST $\overline{\text{Complete}}$ FF. Determines if the last word has been searched.
04	V503	Reduce R^1 → R^2	b ≠ 0	Subtract first count from the contents of the index register
	N570	Clear K410/411	R ≠ 0	Clear R ≠ 0 FF if no transfer is to be made
05	V503	R^2 → R^1	b ≠ 0	Prepare to form first word address
	V561	Set K416/417	R ≠ 0 + b = 0	Set ST $\overline{\text{Complete}}$ FF if a transfer is to be made
09	V508	Full exit	Exit FF = 0 and R = 0	Exit if no search is to be performed
		Half exit	Exit FF = 1 and R = 0	
	V521	Set K210/211		Set wait storage
		Set K550/551		Set Clear B FF. Clear B^b_2. Prepare the index register to receive (R^2).
10	V508	Add R^1 → U^2	b ≠ 0	Form the first address to be searched
	V856	Set K602/603		Initiate storage at (U^2)

*r = number of words to be searched

TIME	TERM	COMMAND	CONDITION	REMARKS
11	V521	Clear X^1		Prepare X to receive memory
14	V560	Set K410/411		Set R ≠ 0 FF. Prepare to check for second search address.
		Set K570/571		Set the $R^2 \rightarrow B^b$ FF. Transfer $R^2 \rightarrow B^b$.
		Clear K416/417		Clear ST $\overline{\text{Complete}}$ FF
16	V526	Clear K570/571		Clear $R^2 \rightarrow B^b$ FF
	V524	$U^1 \rightarrow U^2$		Prepare to form second address in U^2
	N570	Clear K410/411	$R^2 \neq 0$	
17	V561	Set K416/417	R ≠ 0	Records that the Search is not completed
18	V527	Reduce $R^1 \rightarrow R^2$		Prepare formation of second address
19		$R^2 \rightarrow R^1$		Prepare to Add $R^2 \rightarrow R^1$ for second address
	V073 V075	Late Resume		
21	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer the search word to X^1
22	V540	Comp. $X^1 \rightarrow X^2$		Transfer search word into X^2. The adder enables will be used for the comparison of A and memory (M).
	V542	Clear K210/211		Clear Wait Storage
24	V532	Add $R^1 \rightarrow U^2$		Form next address to be searched
29	V548	Full Exit**	(X = A) + Search terminated and Exit FF = 0	Search condition satisfied. Skip next instruction.
	V548	Half Exit	Search terminated and Exit FF = 1 and Search condition not satisfied	Search words exhausted

**Ordinarily this instruction is used in upper position

TIME	TERM	COMMAND	CONDITION	REMARKS
	V549	Into H550	Search conditions not satisfied and Search not terminated	Continue loop if earch is not complete
30	V550	Into H521	No Interrupt or clock request	Continue Search. Return to time 09.
		Into H789	Interrupt or clock request	To interrupt sequence. Temporarily halt ST Sequence.

CODE	INSTRUCTION	FUNCTION
65 THS	Threshold Search	Search B^b words beginning with m + B^b - 1 for (M) > (A). Exit if (M) > (A); Half Exit if (M) ≤ (A).

SEQUENCE: Search and Transfer (H^{5--}, V^{5--})

EXECUTION TIME: 3.6 μsec min., 4.0 + 3.6r* avg., 6.8 + 3.6r* max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	I^2 I^3 → R^1	b ≠ 0	Prepare to form M
	V099	U^1 → U^2		
01	V000	Clear X^1		Prepare X to receive operand from M
02	V560	Set K410/411		Set the R ≠ 0 FF. This FF determines if the search will be made. If R is equal to 0, no search is made and N570 clears this FF.
		Clear K416/417		Clear ST $\overline{\text{Complete}}$ FF. Determines if the last word has been searched.
04	V503	Reduce R^1 → R^2	b ≠ 0	Subtract first count from the contents of the index register
	N570	Clear K410/411	R ≠ 0	Clear R ≠ 0 FF if no transfer is to be made
05	V503	R^2 → R^1	b ≠ 0	Prepare to form first word address
	V561	Set K416/417	R ≠ 0 + b = 0	Set ST $\overline{\text{Complete}}$ FF if a transfer is to be made
09	V508	Full exit	Exit FF = 0 and R = 0	Exit if no search is to be performed
		Half exit	Exit FF = 1 and R = 0	
	V521	Set K210/211		Set wait storage
		Set K550/551		Set Clear B FF. Clear B^b. Prepare the index register to receive (R^2).
10	V508	Add R^1 → U^2	b ≠ 0	Form the first address to be searched
	V856	Set K602/603		Initiate storage at (U^2)

*r = number of words to be searched

TIME	TERM	COMMAND	CONDITION	REMARKS
11	V521	Clear X^1		Prepare X to receive memory
14	V560	Set K410/411		Set $R \neq 0$ FF. Prepare to check for second search address.
		Set K570/571		Set the $R^2 \rightarrow B^b$ FF. Transfer $R^2 \rightarrow B^b$
		Clear K416/417		Clear ST $\overline{\text{Complete}}$ FF
16	V526	Clear K570/571		Clear $R^2 \rightarrow B^b$ FF
	V524	$U^1 \rightarrow U^2$		Prepare to form second address in U^2
	N570	Clear K410/411	$R^2 \neq 0$	
17	V561	Set K416/417	$R \neq 0$	Records that the Search is not completed
18	V527	Reduce $R^1 \rightarrow R^2$		Prepare formation of second address
19	V527	$R^2 \rightarrow R^1$		Prepare to Add $R^2 \rightarrow R^1$ for second address
	V073, V075	Late Resume		
21	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer the search word to X^1
22	V540	Comp. $X^1 \rightarrow X^2$		Transfer search word into X^2. The adder enables will be used for the comparison of A and memory (M).
	V542	Clear K210/211		Clear Wait storage
24	V532	Add $R^1 \rightarrow U^2$		Form next address to be searched
29	V548	Full Exit**	(X A) + Search terminated and Exit FF = 0	Search condition satisfied. Skip next instruction.
		Half Exit	Search terminated and Exit FF = 1 and Search condition not satisfied	Search words exhausted

**Ordinarily this instruction is used in upper position

TIME	TERM	COMMAND	CONDITION	REMARKS
	V549	Into H550	Search conditions not satisfied and Search not terminated	Continue loop if Search is not complete
30	V550	Into H521	No Interrupt or clock request	Continue Search. Return to time 09.
		Into H789	Interrupt or clock request	To Interrupt Sequence. Temporarily halt ST Sequence.

CODE	INSTRUCTION	FUNCTION
66 MEQ	Masked Equality Search	Search B^b words beginning with m + B^b - 1 for L (Q) (M) = A. Exit if L (Q) (M) = A. Half Exit if L (Q) (M) ≠ A.

SEQUENCE: Search and Transfer (H^{5--}, V^{5--})

EXECUTION TIME: 3.6 μsec min., 4.0 + 3.6r* avg., 6.8 + 3.6r* max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	I^2 I^3 → R^1	b ≠ 0	Prepare to form M
	V099	U^1 → U^2		
01	V000	Clear X^1		Prepare X to receive operand from M
02	V560	Set K410/411		Set the R ≠ 0 FF. This FF determines if the search will be made. If R is equal to 0 no search is made and N570 clears this FF.
		Clear K416/417		Clear ST $\overline{\text{Complete}}$ FF. Determines if the last word has been searched.
04	V503	Reduce R^1 → R^2	b ≠ 0	Subtract first count from the contents of the index register
	N570	Clear K410/411	R ≠ 0	Clear R ≠ 0 FF if no transfer is to be made
05	V503	R^2 → R^1	b ≠ 0	Prepare to form first word address
	V561	Set K416/417	R ≠ 0 + b = 0	Set ST $\overline{\text{Complete}}$ FF if a transfer is to be made
09	V508	Full exit	Exit FF = 0 and R = 0	Exit if no search is to be performed
		Half exit	Exit FF = 1 and R = 0	
	V521	Set K210/211		Set Wait Storage
		Set K550/551		Set Clear B FF. Clear B^b. Prepare the index register to receive (R^2).
10	V508	Add R^1 → U^2	b ≠ 0	Form the first address to be searched
	V856	Set K602/603		Initiate storage at (U^2)

*r = number of words to be searched

TIME	TERM	COMMAND	CONDITION	REMARKS
11	V521	Clear X^1		Prepare X to receive memory
14	V560	Set K410/411		Set R ≠ 0 FF. Prepare to check for second search address.
		Set K570/571		Set the $R^2 \rightarrow B^b$ FF. Transfer $R^2 \rightarrow B^b$.
		Clear K416/417		Clear ST $\overline{\text{Complete}}$ FF
16	V526	Clear K570/571		Clear $R^2 \rightarrow B^b$ FF
	V524	$U^1 \rightarrow U^2$		Prepare to form second address in U^2
	N570	Clear K410/411	$R^2 \neq 0$	
17	V561	Set K416/417	R ≠ 0	Records that the search is not completed
18	V527	Reduce $R^1 \rightarrow R^2$		Prepare formation of second address
19	V527	$R^2 \rightarrow R^1$		Prepare to Add $R^2 \rightarrow R^1$ for second address
	V073 V075	Late Resume		
21	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer the search word to X^1
	V541	Set K540/541		Set LQX FF. Form logical product of Q and M in X.
22	V542	Clear K210/211		Clear Wait Storage
24	V532	Add $R^1 \rightarrow U^2$		Form next address to be searched
	V542	Comp. $X^1 \rightarrow X^2$		Transfer logical product to X^2
	V544	Clear K540/541		Clear LQX FF
29	V548	Full Exit**	(X = A) + Search terminated and Exit FF = 0	Search condition satisfied. Skip next instruction.

**Ordinarily this instruction is used in upper position

TIME	TERM	COMMAND	CONDITION	REMARKS
	V548	Half Exit	Search terminated and Exit FF = 1 and Search condition not satisfied	Search words exhausted
	V549	Into H550	Search conditions not satisfied and Search not terminated	Continue loop if search is not complete
30	V550	Into H521	No Interrupt or clock request	Continue search. Return to time 09.
		Into H789	Interrupt or clock request	To Interrupt Sequence. Temporarily halt ST Sequence.

CODE	INSTRUCTION	FUNCTION
67 MTH	Masked Threshold Search	Search B^b words beginning with m + B^b - 1 for L (Q) (M) > A. Exit if L (Q) (M) > A. Half Exit if L (Q) (M) ≤ A.

SEQUENCE: Search and Transfer (H^{5--}, V^{5--})

EXECUTION TIME: 3.6 μsec min., + 4.0 + 3.6r* avg., 6.8 + 3.6r* max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V871	I^2 I^3 → R^1	b ≠ 0	Prepare to form M
	V099	U^1 → U^2		
01	V000	Clear X^1		Prepare X to receive operand from M
02	V560	Set K410/411		Set the R ≠ 0 FF. This FF determines if the search will be made. If R is equal to 0, no search is made and N570 clears this FF.
		Clear K416/417		Clear ST $\overline{\text{Complete}}$ FF. Determines if the last word has been searched.
04	V503	Reduce R^1 → R^2	b ≠ 0	Subtract first count from the contents of the index register
	N570	Clear K410/411	R ≠ 0	Clear R ≠ 0 FF if no transfer is to be made
05	V503	R^2 → R^1	b ≠ 0	Prepare to form first word address
	V561	Set K416/417	R ≠ 0 + b = 0	Set ST $\overline{\text{Complete}}$ FF if a transfer is to be made
09	V508	Full exit	Exit FF = 0 and R = 0	Exit if no search is to be performed
		Half exit	Exit FF = 1 and R = 0	
	V521	Set K210/211		Set wait storage
		Set K550/551		Set Clear B FF. Clear B^b_2. Prepare the index register to receive (R^2).
10	V508	Add R^1 → U^2	b ≠ 0	Form the first address to be searched
	V856	Set K602/603		Initiate storage at (U^2)

*r = number of words to be searched

TIME	TERM	COMMAND	CONDITION	REMARKS
11	V521	Clear X^1		Prepare X to receive memory
14	V560	Set K410/411		Set R ≠ 0 FF. Prepare to check for second search address.
		Set K570/571		Set the $R^2 \rightarrow B^b$ FF. Transfer $R^2 \rightarrow B^b$
		Clear K416/417		Clear ST $\overline{\text{Complete}}$ FF
16	V526	Clear K570/571		Clear $R^2 \rightarrow B^b$ FF
	V524	$U^1 \rightarrow U^2$		Prepare to form second address in U^2
	N570	Clear K410/411	$R^2 \neq 0$	
17	V561	Set K416/417	R ≠ 0	Records that the search is not completed
18	V527	Reduce $R^1 \rightarrow R^2$		Prepare formation of second address
19	V527	$R^2 \rightarrow R^1$		Prepare to Add $R^2 \rightarrow R^1$ for second address
	V073, V075	Late Resume		
21	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer the search word to X^1
	V541	Set K540/541		Set LQX FF. Form logical product of Q and M in X.
22	V542	Clear K210/211		Clear wait storage
24	V532	Add $R^1 \rightarrow U^2$		Form next address to be searched
	V542	Comp. $X^1 \rightarrow X^2$		Transfer logical product to X^2
	V544	Clear K540/541		Clear LQX FF
29	V548	Full Exit**	(X A) + Search terminated and Exit FF = 0	Search condition satisfied. Skip next instruction.
	V548	Half Exit	Search terminated and Exit FF = 1 and Search condition not satisfied	Search words exhausted

**Ordinarily this instruction is used in upper position

TIME	TERM	COMMAND	CONDITION	REMARKS
	V549	Into H550	Search conditions not satisfied and search not terminated	Continue loop if search is not complete
30	V550	Into H521	No Interrupt or clock request	Continue search. Return to time 09.
		Into H789	Interrupt or clock request	To interrupt sequence. Temporarily halt ST sequence.

CODE	INSTRUCTION	FUNCTION
70 RAD	Replace Add	Form the sum $[(M) + (A)]$ and store at M and in A.

SEQUENCE: Read Operand (H3--, V3--)

EXECUTION TIME: 10.2 usec minimum, 13.2 usec average, 16 usec maximum

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		} Prepare to form M
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	} RNI
01	V000	Clear X^1		
03	V303	Set K602/603		Set Initiate Storage FF
04	V302	Add $R^1 \rightarrow U^2$		Add b to m to form M
05	V304	Clear R^1		
06	V306	Set K580/581		Set Clear A^1 FF. Clear A^1
08	V308	Clear K580/581		Clear Clear A^1 FF
10	V310	Set K202/203		Set Wait Storage FF
13	V073 V075	Storage Resume		
14	V340	Clear K202/203		Clear Wait Storage 1 FF
15	V341	Clear K520/521		Clear Partial Add A FF. Enable a Full Add in A.
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer $(M) \rightarrow X^1$
16	V340	$X^1 \rightarrow X^2$		Transfer (M) to X^2
21	V345	Clear X^1		Clear X^1 to receive $[(M) + (A)]$
		Add $X^2 \rightarrow A^1$		Form the sum $[(M) + (A^1)]$ in A^1
19	V345	Set K204/205		Set R. O. Wait Storage FF
25	V349	$A^1 \rightarrow X^1$		Transfer $[(M) + (A^1)]$ to X^1

TIME	TERM	COMMAND	CONDITION	REMARKS
28	V353	Set K602/603		Set Initiate Storage FF
45	V075	Storage Resume		
47	V360	Exit		
48	N^{-91}	Set Z		Transfer (X) → I^5 + I^6 → Z^1 + Z^2. Write (X) into memory. Store [(M) + (A)] at M. (A^1) which now also contains [(M) + (A)] remains unchanged.

CODE	INSTRUCTION	FUNCTION
71 RSB	Replace Subtract	Form the difference $[(M) - (A)]$ and store at M and in A.

SEQUENCE: Read Operand (H3-- V3--).

EXECUTION TIME: 10.2 usec minimum, 13.2 usec average, 16.0 usec maximum.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M } RNI
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	
01	V000	Clear X^1		
03	V303	Set K602/603		Set Initiate Storage FF
04	V302	Add $R^1 \rightarrow U^2$		Add b to m to form M
05	V304	Clear R^1		
06	V304	Comp. $X^1 \rightarrow X^2$		
07	V305	Part. Add X^2 to A^1		Complement A^1
10	V310	Set K202/203		Set Wait Storage 1 FF
13	V073	Storage Resume		
	V075			
14	V340	Clear K202/203		
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (M) to X^1
16	V340	$X^1 \rightarrow X^2$		Transfer (M) to X^2
15	V341	Clear K520/521		Clear Partial Add A FF. Enable a Full Add in A
19	V345	Set K204/205		Set R.O. Wait Storage FF
21		Add $X^2 \rightarrow A^1$		Subtract $[(M) - (A)]$ and store in A^1
		Clear X^1		Clear X^1 to receive (M) - (A)
25	V349	$A^1 \rightarrow X^1$		Transfer $[(M) - (A)]$ to X^1. (A^1) remains unchanged.

TIME	TERM	COMMAND	CONDITION	REMARKS
28	V353	Set K602/603		Set Initiate Storage FF
45	V075	Storage Resume		
47	V360	Exit		
48	N^{-91}	Set Z		Transfer (X) → I^5 I^6 → Z^1 + Z^2. Write (X) into memory. Store [(M) - (A)] at M. (A^1) which now also contains [(M) - (A)] remain unchanged.

CODE	INSTRUCTION	FUNCTION
72 RAO	Replace Add One	Add 1 to (M) and store (M + 1) at M and in A.

SEQUENCE: Read Operand (H3-- V3--).

EXECUTION TIME: 10.2 usec minimum, 13.2 usec average, 16.0 usec maximum.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to form M
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	Prepare to form M; RNI
01	V000	Clear X^1		
03	V303	Set K070/071		Set X^2 to 1 FF
04	V303	Set K602/603		Set Initiate Storage FF
	V302	Add $R^1 \rightarrow U^2$		Add b to m to form M
05	V304	Clear R^1		
06		$X^1 \rightarrow X^2$		Place a count of 1 in X^2
06	V306	Set K580/581		Set Clear A^1 FF. Clear A^1
07	V307	Clear K070/071		Clear Set X^2 to 1 FF
08	V308	Clear K580/581		Clear Clear A^1 FF
10	V310	Set K202/203		Set Wait Storage 1 FF
11	V309	Add X^2 to A^1		Place a count of 1 in A^1
13	V311	Clear X^1		Clear X^1 to use as an exchange register
13	V073	Storage Resume		
	V075			
14	V340	Clear K202/203		Clear Wait Storage FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (M) $\rightarrow X^1$
16	V340	$X^1 \rightarrow X^2$		Transfer (M) to X^2

TIME	TERM	COMMAND	CONDITION	REMARKS
15	V341	Clear K520/521		Clear Partial Add A FF to enable Full Add
19	V345	Set K204/205		Set Wait Storage FF. (Prepare to write (M + 1) back into memory)
21		Add $X^2 \rightarrow A^1$		Form (M + 1) in A^1
		Clear X^1		
25		$A^1 \rightarrow X^1$		Transfer (M + 1) to X^1
28	V353	Set K602/603		Set Initiate Storage FF
45	V075	Storage Resume		
47	V360	Exit		
48	N^{-91}	Set Z		Transfer (X) $\rightarrow I^5 + I^6 \rightarrow Z^1 + Z^2$. Write (X) into memory. Store (M + 1) at M.

CODE	INSTRUCTION	FUNCTION
73 RAO	Replace Subtract One	Subtract 1 from (M) and store the difference at M and in A.

SEQUENCE: Read Operand (H3-- V3--).

EXECUTION TIME: 10.2 usec minimum, 13.2 usec average, 16.0 usec maximum.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		} Prepare to form M
	V871	$I^2\ I^3 \rightarrow R^1$	$b \neq 0$	
01	V000	Clear X^1		
03	V303	Set K070/071		Set X^2 to 1 FF
04		Set K602/603		Set Initiate Storage FF
	V302	Add $R^1 \rightarrow U^2$		Add b to m to form M
05	V304	Clear R^1		
06		Comp. $X^1 \rightarrow X^2$		
	V306	Set K580/581		Set Clear A^1 FF
07	V307	Clear K070/071		Clear Set X^2 to 1 FF
08	V308	Clear K580/581		Clear Clear A^1 FF
10	V310	Set K202/203		Set Wait Storage 1 FF
11	V309	Add X^2 to A^1		Place a count of negative 1 in A^1
13	V311	Clear X^1		Clear X^1 to use as an exchange register
13	V073	Storage Resume		
	V075			
14	V340	Clear K202/203		Clear Wait Storage FF
15	V073	$I^5\ I^6\ I^7 \rightarrow X^1$		Transfer (M) $\rightarrow X^1$
15	V341	Clear K520/521		Clear Partial Add A FF to enable Full Add

TIME	TERM	COMMAND	CONDITION	REMARKS
16	V340	$X^1 \rightarrow X^2$		Transfer (M) to X^2
19	V345	Set K204/205		Set R.O. Wait Storage FF. Prepare to write (M - 1) back into memory.
21		Add $X^2 \rightarrow A^1$		Form (M - 1) in A^1
		Clear X^1		
25		$A^1 \rightarrow X^1$		Transfer (M - 1) to X^1
28	V353	Set K602/603		Set Initiate Storage FF
45	V075	Storage Resume		
47	V360	Exit		
48	N^{-91}	Set Z		

CODE	INSTRUCTION	FUNCTION
74.0 CWXYZ EXF	External Function (Select)	Select the external equipment on Channel C or the mode of operation of external equipment, or the various internal conditions depending on code 0WXYZ.

SEQUENCE: External Function (H^{7--}, V^{7--})

EXECUTION TIME:

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to transfer the Select code to X^1
01	V000	Clear X^1		Prepare X to receive the Select code
03	V701	$U^2 \rightarrow X^1$ with sign extension		Prepare to transfer Select code to X^2 where it will be translated. The sign is not extended. This only occurs on an (F527 = 1).
04	V702	$X^1 \rightarrow X^2$	j = 0 + 7	Select code to X^2 to be translated
05	V705	Set K716/717	j = 0	Set the Select FF. This releases the external function counter to count. The EXF counter delays the exit for 13.8 μsec to allow the equipment to be selected. It holds the code on the lines for 12 usec.
				NOTE:
				Depending on when V026 occurs the exit may vary by 1.6 usec. Insert timing for the conditions of the external function counter you are using.
	V026	Into H709		Clear Select FF
	V709	Clear K716/717		Clear Select FF. Stop EXF counter.
	V709	Set K138/139		Set enable sense exit. Enable Exit condition.
	V026	Exit		Sense Resync FF will be set or cleared depending on the condition of the Exit FF.
	V027	Clear K138/139		

CODE	INSTRUCTION	FUNCTION
74.7 CWXYZ EXF	External Function (Sense)	Sense the condition of an external equipment on Channel C or the internal condition of the computer depending on the code 0WXYZ.

SEQUENCE: External Function (H^{7--}, V^{7--})

EXECUTION TIME:

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Prepare to transfer Sense code to X^1
01	V000	Clear X^1		Prepare X to receive Sense code
03	V701	$U^2 \rightarrow X^1$ with sign extension		Prepare to transfer code to X^2 for translation. Sign is not extended (F527 = 0).
04	V702	$X^1 \rightarrow X^2$	j = 0 + 7	Sense code to X^2 for translation
05	V705	Set K718/719	j = 7	Set Sense FF. This enables the EXF counter to count delaying the exit until the code has been on the lines for 12 usec.
				NOTE:
				The Sense code enables must be followed through (on page 30, Vol. 3) until the Sense Resync FF K136/137 is set.
				Depending on when V026 occurs, the exit can occur 1.6 μsec later. Insert the timing that you use for the EXF counter.
	V026	Into H709		
	V707	Clear K718/719		Clears the Sense FF. Disables EXF counter.
		Set K138/139		Set Enable Sense Exit FF. Prepare to exit.
	V125	Set K136/137	Sense Condition Present	Enable full exit if condition is present
	V026	Exit		Terminate operation
	V027	Clear K138/139		
				NOTE: 74.7 is a Skip instruction when used as a upper instruction and a Wait instruction when used as a lower instruction.

CODE	INSTRUCTION	FUNCTION
74 j XXXXX j = 1-6	External Function (Activate)	Activate buffer channel j. The execution address, XXXXX, designates the starting address of the buffer region.

SEQUENCE: External Function (H^{7--}, V^{7--})

EXECUTION TIME:

TIME	TERM	COMMAND	CONDITION	REMARKS
	V099	$U^1 \rightarrow U^2$		Transfer starting address into U^2 prior to a $U^2 \rightarrow X^1$ to store the starting address in the control word.
01	V000	Clear X^1		Prepare X to receive the starting address
02	V701	j to Aux. Ref. Reg.		Determines control register to be referenced. (See page 42, Logic diagrams Quadrant 2).
03	V701	$U^2 \rightarrow X^1$		Transfer starting address to X^1 to be stored
	V703	Set K622/623 Set K662/663	$\overline{\text{(Aux. Request)}}$ (j = 1-6)	Set the Storage Busy FFs if there is no auxiliary request. This prevents any memory references prior to storing the starting address into memory. If there is an auxiliary request, the timing chain will loop back on itself from V704 back to H703 as long as memory is busy. This is to prevent the memory from being referenced while an auxiliary request is referencing memory.
04	V704	Into H703	Auxiliary Request	Loop back and wait for the auxiliary request to terminate
	V704	Into H705	$\overline{\text{Auxiliary Request}}$	Continue in sequence to store initial address if there is no auxiliary request
05	V705	Set K216/217		Set the F = 74.1 6 FF. This remembers that an Activate instruction and not an auxiliary request is referencing the control word. It also enables a partial write upper.
06	V705	$X^1_{LA} \rightarrow X^2_{UA}$		Transfer the starting address into X^1_U so it can be written into memory

TIME	TERM	COMMAND	CONDITION	REMARKS
	V706	To H171		To high speed storage sequence (page 42, Vol. 3)
07	V705	$X^2_U \rightarrow X^1_U$ Clear X^1_L		

CODE	INSTRUCTION	FUNCTION
75 SLJ	Selective Jump	Cause a jump or a return jump to m if the specific jump key is set. The position of keys is specified by j = 0-7.

SEQUENCE: j = 0-3 Normal Jump (H^{1--}, V^{1--}) j = 4-7 Write Operand (H^{4--}, V^{4--})

EXECUTION TIME: 3.0 μsec min., 7.2 μsec avg., 11.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Transfer m to U^2
01	V000	Clear X^1		Clear X^1 for use in write operand sequence
Normal Jump				
03	V102	Jump Exit	Jump condition satisfied	
		Half Exit	Jump condition not satisfied and Exit FF = 1	
		Full Exit	Jump condition satisfied and Exit FF = 0	
		Set K560/561	Jump condition satisfied	Set $U^2 \rightarrow P^1$ FF. Insert next instruction address into P.
Write Operand				
04	V856	Set K602/603	Jump condition satisfied	Initiate storage
06	V405	Adv. $P^2 \rightarrow P^1$	Mode Switch Down	Determine next address of current routine
07	V407	Set K206/207		Set Write Operand Wait Storage FF
08	V408	$U^2 \rightarrow P^1$		Transfer m to p to select next instruction word
		Set K460/461		Set Return Jump FF. Conditions later commands.
	V407	Enable Partial Write Upper		

TIME	TERM	COMMAND	CONDITION	REMARKS
09	V407	$X^2 \rightarrow X^1$		
	V407	$X^2 \rightarrow X^1$		
11	V409	Clear U^1_u		Place P_i in X^1_u
12	V971	$X^1_{LA} \rightarrow X^2_u$		
13	N380	$X^2_u \rightarrow X^1_u$		
		Clear X^1_L		
	V073	Storage Resume		
14	V420	Clear K206/207		
15		$I^5\ I^6\ I^7 \rightarrow U^1$		Transfer next instruction to U^1
		Half Exit	Return Jump FF = 1	
		Set Z		Transfer $X^1 \rightarrow I^5\ I^6\ I^7$ to $Z^1 + Z^2$ to memory. Write return address (P_i) into storage.

CODE	INSTRUCTION	FUNCTION
76 SLS	Selective Stop	j = 0-3 jump unconditionally to upper instruction of m. If Stop Key condition is satisfied, stop and jump on restart. J indicates Stop Key condition. j = 4-7 Return jump to m if condition is met. Execute next instruction if condition is not met.

SEQUENCE: j = 0-3 Normal Jump (H^{1--}, V^{1--}) j = 4-7 Write Operand (H^{4--}, V^{4--})

EXECUTION TIME: 3.0 μsec min., 7.2 μsec avg., 11.6 μsec max.

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V099	$U^1 \rightarrow U^2$		Transfer m to U^2
		Set K050/051	Stop condition met b = 0-7	Set the Stop II FF and prepare to stop
01	V000	Clear X^1		Clear X^1 for later use
	V101	Stop	K050/051 = 1 at T = 00	Stop if the Stop condition is met
Restart in H00 Logic		on start or step pulse and clear K050/051 from V042. See Manual Control		
01	V101	Into H102	K050/051 = 0	
03	V102	Jump Exit		
	V103	Set K560/561	Jump condition met	Set $U^2 \rightarrow P^1$ FF. Place next instruction in P
Write Operand Sequence				
04	V856	Set K602/603	Jump conditions met	Initiate storage
06	V405	Adv. $P^2 \rightarrow P^1$	Mode Switch Down	Determine next address of current routine
07	V407	Set K206/207		Set Write Operand Wait Storage FF
08	V408	$U^2 \rightarrow P^1$		Transfer m to P to select next instruction word
	V407	$P^1 \rightarrow X^2_L$ Enable Partial Write Upper		
	V408	Set K460/461		Set Return Jump FF. Conditions later commands.

RNI

TIME	TERM	COMMAND	CONDITION	REMARKS
09	V407	$X^2 \rightarrow X^1$		Set $X^1 = X^2$
11	V409	Clear U^1		Prepare U^1 for new instruction
12	V971	$X^1_{LA} \rightarrow X^2_u$		
13	N380	$X^2_u \rightarrow X^1_u$		
14	V420	Clear K206/207		
15		$I^5\ I^6\ I^7 \rightarrow U^1$		Transfer next instruction to U^1
		Set Z		Transfer $X^1 \rightarrow I^5\ I^6\ I^7$ to $Z^1 + Z^2$ to memory. Write return address (P_i) into storage.
		Half Exit	Return Jump FF = 1	

CODE	INSTRUCTION	FUNCTION
		Auxiliary Sequence (Auxiliary Sequence is a sub-sequence of Buffer Control)

SEQUENCE: Auxiliary Sequence (H^{4--}, V^{4--})

EXECUTION TIME:

TIME	TERM	COMMAND	CONDITION	REMARKS
				NOTE: For purpose of this timing it is assumed that Auxiliary Request FF is set at T = -02
-01	N091	Into H440	Memory Not Busy And Aux. Request	Start auxiliary sequence
00	N091	Rank I of Aux. Scanner To ARD	(Memory $\overline{\text{Busy}}$) (Aux. Request)	This determines which control register will be referenced
	V440	Set K622/623 Set K662/663		Set Storage Busy FFs. At this point the computer has not determined which control register will be referenced. Both Storage Busy FFs are set to prevent the program from taking memory away from the auxiliary sequence. The FF that will not be used will be cleared at T = 04.
	V440	Set K612/613 Set K614/615		Set the CR → I^7 FFs. Prepare to extract the current address.
	V440	Set K252/253		Set the auxiliary sequence Wait Storage FF. Provide the enables to indicate that an auxiliary sequence is in progress.
01	V440	Clear CCR		Clear the common control register. Prepare CCR to receive the current address and terminal address.
	V481	M → I^0	ARD = 1, 8 or 5	Only on Input Buffer (page 61, Vol. 3)
	V441	Clear 0-	ARD = 2, 4 or 6	Clear the output register designated by ARD (page 63, Vol. 3)

TIME	TERM	COMMAND	CONDITION	REMARKS
02	V441	$I^7 \rightarrow CCR$		Transfer the Control Word to CCR. Prepare to extract current address update, and check for termination.
03	V442	Set S^1 and S^2 to all 1's		Prepare S^1 or S^2 for a zeros transfer of information from CCR_U. This will be the memory location specified by the current address portion of the CR.
04	V443	$CCR_U \rightarrow S^2$ Set K664/665 Clear K622/623 Into H081	Current Address is Odd	Transfer current address to S^2 if odd memory diverter drive clear the Even Memory Busy FF. This is done so both Memory Busy FFs will be cleared at the end of the buffer cycle. H081 is the start of the odd memory timing chain.
04	V443	$CCR_U \rightarrow S^1$ Set K624/625 Into H061 Clear K662/663	Current Address is Even	Transfer current address to even memory address register. Set Even Memory Diverter Drive FF. H061 is the start of the even memory timing chain. Clear the odd Storage Busy FF. This is done so both Storage Busy FFs will be cleared at the end of the buffer cycle. NOTE: If the current address specifies locations 00001-00006 the high speed storage sequence will be operating at the same time as the magnetic core memory sequence. The address portions (30 bits) will come from or go to the Control registers. Non address portions (18 bits) will come from or go to the magnetic core storage. If the current address specifies 00001-00006 the contents of ARD must be changed to specify the current address, as opposed to the control word for this buffer operation. ARD must also be changed after the input or output to again reference the control word. This is to check to see if the buffer is terminated. A complete timing for a reference to address 00001 is listed on the following page, indicated by a asterisk in the Command Column.
	V444	Into H479	CCR_u = 00001-00006	Go to high speed storage sequence if CCR_u equals 00001-00006.

TIME	TERM	COMMAND	CONDITION	REMARKS
04	V490	*Clear K154/155		These two FFs will be used to transfer the lowest two bits of the S registers to ARD for a buffer reference to 00001.
		*Clear K152/153		
05	V475	Set K848/849		Record that a buffer reference is being made to a control word area.
05 (01)	V171	*Set K016/017		Set Storage Enable FF
06	V445	$I^7 + 1 \rightarrow CCR_U$		Advance the current address
	V482	*Set K154/155		Record Bits 1 and 2 of S. Recall the S register not being used is set to all "1's". These two FFs will be used to transfer into the upper two bits of ARD to determine what control register will be referenced by the buffer. The lowest bit of ARD will come from the Storage Busy FFs. This is because S^1 and S^2 are 14-bit registers.
		*Set K152/153		
	V446	Set K004/005		Enable $CCR_U \rightarrow I^8$. Prepare to transfer the (CCR_U), which is the updated address into the control register.
06 (02)	V445	$I^7 + 1 \rightarrow CCR_U$		Advance the current address
	V482	*Set K154/155		Record Bits 1 and 2 of S. Recall the S register not being used is set to all "1's". These two FFs will be used to transfer into the upper two bits of ARD to determine what control register will be referenced by the buffer. The lowest bit of ARD will come from the Storage Busy FFs.
		*Set K152/153		
	V446	Set K004/005		Enable $CCR_U \rightarrow I^8$. Prepare to transfer the (CCR_U), which is the updated address into the control register.
07	V447	Clear K012/013		Clear the $CR \rightarrow I^7$ FFs
		Clear K014/015		
	V447	Into H484	CCRU = 00001-00006 at 05 time	Prepare to change ARD Transfer updated current address into the control register.
08	V447	$I^8 \rightarrow CR_U$		

TIME	TERM	COMMAND	CONDITION	REMARKS
09	V485	*S_1S_2 → ARD	CCRU = 00001-00006 at 05 time	Indicate what control register is designated by the S register. (page 45, Vol. 3'
11	V073	Into H448		Continue with auxiliary sequence via late resume
11 (07)	V176	*Clear CCR		Prepare CCR to receive word to be transferred in or out of the control register
	V177	*Set K018/019	Write Operation	Enable I^5 I^6 → I^8
12	V448	Clear K004/005		Clear the CCR_U → I^8 FF
13		Set Z (Mag.Core Storage)		Write input word into mag. core memory
13 (09)	V178	*Set CR_U = 1's	Write Upper	Prepare to write input word into the CCF
	N656 N658	*Set CR_L = 1's	Write Lower	Used on an input only
	V180	*Clear K012/013		Clear the CR → I^7 FFs
		*Clear K014/015		
16	V451	Rank I of the Scanner → ARD		Restore ARD so it designates the correct control word for this buffer operation. This is done in case the buffer operation was to or from a control word area (00001).
16 (12)	V186	*Clear K018/019		Clear the I^5 I^6 → I^8 FF
18	V454	Set K744/745		Enable I^0 → 0. Transfer the output word into the output register designated by ARD.
20	V472	Clear K744/745		(page 63, Vol. 3)
21	V455	Set Ready or Clear Resume FF		Clear the I^0 → 0 FF Send Output Ready or Input Resume signals depending on (ARD)
22	V466	Clear Channel Request FFs		Drop action request depending on (ARD)

TIME	TERM	COMMAND	CONDITION	REMARKS
22	V474	Set K014/015		Set the CR → I^7 FF. Prepare to transfer the (CR) into CCR for comparison to terminate the buffer.
	V480	Set K012/013		
25	V476	Clear CCR		Clear CCR. Prepare to transfer (CR) into CCR for comparison.
26	V477	I^7 → CCR		Transfer (CR) into CCR for comparison
31	V063	Into H478		Time 27 of memory cycle
	V083			
32	V456	Clear K712/713	Scanner is running	Clear the Auxiliary Request FF
		Clear K252/253		Clear the Auxiliary Sequence Wait Storage FF
		Clear K012/013 014 015		Clear the CR → I^7 FF
33	V457	Clear K216/217		
	V459	Set Buffer Active FF_0	$CCR_U \neq CCR_L$	Clear the F = 74.1-6 FF
		Clear Buffer Active FFs		Leave buffer active
			$CCR_U = CCR_L$	Terminate buffer

* (Starred steps not necessary for the operation of the auxiliary sequence)

CODE	INSTRUCTION	FUNCTION High Speed Storage (HSS)

SEQUENCE: (H^{1--}, V^{1--}) and (H^{4--}, V^{4--})

EXECUTION TIME:

TIME	TERM	COMMAND	CONDITION	REMARKS
07* (01)	V171	Set K016/017		Set the Storage Enable FF This indicates the memory reference will be to address 00001-6. This disables the read lower and read upper address term J330, J332, J350 and J352. It also provides an enable into J353 and J355, the write address portions for add storage.
08 (02)	V172	Set K606/607	Write Upper	Enable a write upper into the Control registers
	V172	Set K604/605	Write Lower	Enable a write lower into the Control registers
09 (03)	V487	Set K622/623	F = 74.1 - 74.6	Set the Storage Busy FF
		Set K662/663		
10 (04)	V174	Set K012/013	Read Lower	Enable the control register designated by the A.R.R. into I^7 for transfer. Used on a read only.
		Set K014/015	Read Upper	
11 (05)	V174	ARR → ARD	Aux. Seq. Wait Stop Cleared	Transfer the (ARR) to the auxiliary reference designator. This translates which control register is referenced. (page 45, Vol. 3).

*Times in parenthesis are absolute timings for HSS. Times outside the parenthesis continued from 74 1-6 m instruction on page 131.

TIME	TERM	COMMAND	CONDITION	REMARKS
13 (07)	V176	Clear CCR		Clear the common control register. Prepare CCR to receive the contents to be stored. CCR will be used to compare the upper and lower address which checks for termination of buffer and for transfer into the control registers. Used on a buffer operation.
	V177	Set K018/019	Write Operation	Set the I^5 I^6 → I^8 FF. Prepare to write into the Control register (page 40, Vol. 3).
14 (08)	V177	I^7 → CCR	Buffer	Transfer the control register designated to CCR. This is used when buffering from a control word area.
15	V178	Set CR_L → 1's	Write Lower Address	Prepare for a zeros transfer of information into the control register
	N656			
(09)	N658	Set CR_U → 1's	Write Upper Address	
	V178	Clear CCR	Activate	Clear the common control register in preparation to receive the (CR). Used to check the terminal address and starting address to see if a buffer is to be activated.
16	V179	I^8 → CR_L	Write Lower Address	Store Information in the control register
	N657			
(10)	N659	I^8 → CR_U		
	V180	Clear K012/013		Used on a memory reference to address 00001-6
17 (11)	V181	Set K012/013 Set K014/015	f = 74 1-6	Set the CR → I^7 FFs. This is used to transfer CR → I^7 → CCR.
18 (12)	V186	Clear 018/019		Clear the I^5 I^6 I^8 FF
	V186	Into H187	f = 74 1-6	

TIME	TERM	COMMAND	CONDITION	REMARKS
20 (14)	V187	$I^7 \rightarrow$ CCR		Transfer the (CR) to CCR for comparison. This checks to see if the Channel Active FF will be set.
21 (15)	V188	Exit		
22 (16)	V190	Clear K012/013 Clear K014/015		
	V462	Clear K662/663 Clear 622/623		Clear Memory Busy FF's This allows a buffer operation to take place.
	V190	Into H457		(Page 17.1 of CE Diagrams)
23 (17)	V457	Clear K216/217		Clear f - 74.1 = 6 FF
	V457	Set Buffer Active FF (One of 6 depending on ARD)	$CCR_U \neq$ CCR_L	Activate buffer channel
	V457	Clear Buffer Active FF (one of 6 depending on ARD).		Do not activate buffer if starting and terminal addresses are equal. (This is the way to program terminate the buffer).
	V459			Clear Inhibit RK. 1 scanner to ARD FF

CODE	INSTRUCTION	FUNCTION
		External/Internal Interrupt

SEQUENCE: Interrupt and Write Operand

TIME	TERM	COMMAND	CONDITION	REMARKS
-01	V035	Into H786	Interrupt	Enter interrupt sequence via exit
	V031			
	V033			
00	V098	Into H789	Interrupt during RNI	Arithmetic interrupt
	V550	Into H789	Interrupt during a Search	Interrupt during search and transfer operation.
02	V834	Clear X^1		Prepare X to receive (P) to be stored in memory.
03	V787	Set I252/253	(Internal Interrupt) ($\overline{\text{Advance}}$ Clock)	Record that this is an internal interrupt
	V787	Set I254/255	($\overline{\text{Internal Interrupt}}$) ($\overline{\text{Advance}}$ Clock)	Record that this is an external interrupt
04	V787	$P^1 \rightarrow X^2{}_L$	$\overline{\text{Advance}}$ Clock	Prepare to record where interrupt occurred.
	V010 V012	Clear Control FFs		
	V788	Into H401 Write Operand Sequence		Complete interrupt in W. O. sequence
05	V787	$X^2{}_L \rightarrow X^1{}_L$		Prepare to write (P) into memory
	V788	Clear $U^1{}_U$		Prepare function code for interrupt and for advance clock
				NOTE: Times in parenthesis are Write Operand Sequence Timing

TIME	TERM	COMMAND	CONDITION	REMARKS
05 (01)	V413	Into H402	Internal Interrupt	This is done in case an arithmetic interrupt occurs prior to a Stop instruction. (Page 5, Vol. 3).
06 (02)	V401	Set P^1		Set P^1 equal to 00007-17 depending on the type of interrupt. (Page 24, Vol. 3).
	V402	Set K064/065	Interrupt	Set the Interrupt Lockout FF. This prevents any interrupts from occurring until a jump is made back to the upper portion of an interrupt location.
08 (04)	V403	Clear External Interrupt Request	External Interrupt Present	Clear all external requests - (Page 32, Vol. 3).
	V404	Clear I256/257	Internal Interrupt Request Present	Prevent an interrupt request until the lockout is cleared. All requests that were available will reset and wait for the lockout to clear.
	V404	Set K068/069	Exit FF = 1	Set or clear Interrupt Exit FF. Record the position of the last instruction executed before the interrupt.
		Clear K068/069	Exit FF = 0	
	V404	Set K600/601		Initiate storage at (P)
		Set K610/611		Read interrupt trap location
	V404	Into H407	Interrupt Sequence	Continue sequence
09 (05)	V415	Clear K796/797		Clear Internal Interrupt Request FF. Request is recorded and satisfied.
	V407	Set K206/207		Set the write operand wait storage. Enable partial write upper.
10 (06)	V408	Set K460/461		Set the Return Jump FF to return to main program
12 (08)	V410	Clear K694/695		Clear the Interrupt or Clock Request FF
13 (09)	V410	Clear U^1_L		Clear U^1 lower, prepare U^1 to receive the lower portion of the trap location from memory. A half exit will always be performed on an interrupt.

TIME	TERM	COMMAND	CONDITION	REMARKS
16 (12)	V971	$X^1_L \rightarrow X^2_U$		Transfer the address where the interrupt occurred into X^1_u so it can be written into memory
17 (13)	V971	$X^2_u \rightarrow X^1_U$ Clear X^1_L		
17 (13)	V073	Late Resume		Contine with sequence
18 (14)	V420	Clear I252/253		Clear the Internal Interrupt Request FF
	V420	Clear I254/255		Clear the External Interrupt Request FF
19 (15)	V420	$I^5\ I^6\ I^7 \rightarrow U^1$		Transfer (trap location) to U^1 for execution of lower instruction
		Set Z		Store upper address with address of instruction on which the interrupt occurred
	V420	Half Exit	Return Jump FF = 1	Exit to lower half of trap location

CODE	INSTRUCTION	FUNCTION
		Advance Clock

SEQUENCE: Advance Clock and Read Operand

EXECUTION TIME:

TIME	TERM	COMMAND	CONDITION	REMARKS
00	V122	Set K782/783		Set the Clock Action Request FF
01	N951	Set K882/883		Set the Clock Request FF Slave FF
	V031	Set I250/251		Set the Advance Clock FF from an condition and enter sequence
	V033 V035	Into H786		
	N901	Set K694/695		Record advance clock request
03	V550	Set K250/251		Advance clock during a search and transfer
	N940	Set I250/251		
	V552	Set I250/251		
	V834	Clear X^1		Prepare to generate advance count in X^1
06	V010 V012	Clear Control FFs		
	V788	Set K060/061	Exit FF = 0	Complement the Exit FF. This prepares for the correct exit from the read operand sequence.
		Clear K060/061	Exit FF = 1	
	V788	Into H301		Into read operand sequence
07	V788	Clear U^1_U		Prepare Function Code and address for Advance Clock operation
				NOTE: Timing continued from clock request sequence. Parenthesis indicate R. O. sequence timing. Memory is assumed to be not busy.

TIME	TERM	COMMAND	CONDITION	REMARKS
09 (03)	V302	U^1 U^2		Transfer 0000 to U^2. This is the address that will be read from memory. Prepare for a $U^2 \rightarrow S$.
	V303	Set K070/071		Set the Set $X^2 \rightarrow 1$ FF. This enables X^2 to be set to a plus one on an advance clock operation.
10 (04)	V856	Set K602/603		Initiate Storage at (U^2)
12 (06)	V304	$Q^1 \rightarrow Q^2$		Prepare to temporarily store (A) in Q
	V304	$X^1 \rightarrow X^2$		Make X^2 equal to a plus one
	V306	Set K580/581		Set Clear A FF. Clear A. Prepare it to receive the plus one quantity.
13 (07)	V305	$A^2 \rightarrow Q^1$		Temporarily store A in Q
	V307	Clear K070/071		Clear Set $X^2 \rightarrow 1$ FF
14 (08)	V308	Clear K580/581		Clear Clear A FF
16 (10)	V310	Set K202/203		Set Wait Storage
17 (11)	V309	Partial add $X^2 \rightarrow A^1$		Transfer plus one quantity to A
19 (13)	V311	Clear X^1		Prepare X^1 to receive (Memory = 00000)
	V073	Late Resume		
	V075			
20 (14)	V340	Clear K202/203		
21 (15)	V073	I^5 I^6 $I^7 \rightarrow X^1$		Read (00000) into X^1 so the sum of (00000) and A can be formed
	V341	Clear K520/521		Clear Partial Add in A FF. Prepare to advance (00000) by 1.

TIME	TERM	COMMAND	CONDITION	REMARKS
22 (16)	V340	$X^1 \rightarrow X^2$		
	V342	Set I240/241		Set the F = 00 + 77 FF. Indicates that this is an Advance Clock Operation.
25 (19)	V345	Set K204/205		Set the R. O. Wait Storage 2 FF
27 (21)	V345	Add $X^2 \rightarrow A^1$		Form sum of (00000) + 1 in A
	V345	Clear X^1		Prepare X to receive (00000) + 1 for storage into memory
28 (22)	V348	Set K780/781	Exit FF = 0	Condition exit from R. O. sequence
30 (24)	V350	Clear K754/755		Clear the Clock Wait FF
31 (25)	V349	$A^1 \rightarrow X^1$		Prepare to store in 00000
	V351	Clear K782/783		Clear the Clock Request FF
		Set K486/487		Set the F = 37 FF. Its enables will be used to terminate this operation.
32 (26)	V352	Clear K886/883		Clear the Clock Request FF Slave
33 (27)	V352	$Q^2 \rightarrow A^1$		Prepare to restore A
34 (28)	V352	$Q^1 \rightarrow Q^2$		Prepare to restore A
	V353	Set K602/603		Initiate storage
		Clear K694/695		Clear the Interrupt or Clock Request FF
35 (29)		$Q^2 \rightarrow A^1$		Restore A and AQ
		$A^2 \rightarrow Q^1$		

TIME	TERM	COMMAND	CONDITION	REMARKS
51 (45)	V075	Storage Resume		
52 (46)	V760	Clear I250/251		Clear the Advance Clock FF
		Clear I240/241		Clear the F = 00 + 77 FF
53 (47)	V360	Half Exit	Exit Slave FF = 0	
	V360	Jump Exit	Exit Slave FF = 1	

CHAPTER 5

MAINTENANCE PARTS LIST

INTRODUCTION

The Parts List provides the identification and ordering data necessary for the replacement of electrical and mechanical parts for the CONTROL DATA 1604-A Computer.

Electrical Contents: All items are included except jumper wires and wire. Quantities are per one 1604-A Computer. Quantities on printed circuit card assemblies are per one 1604-A Computer (8 Chassis).

Hardware Contents: All items are included except standard hardware such as screws, nuts, bolts, washers and raw material. Quantities are per one 1604-A Computer.

All Control Data Corporation assemblies are listed and are broken down into individual parts (with the exception of printed circuit card assemblies) but are listed in alphabetical rather than disassembly order.

For the breakdown of printed circuit card assemblies refer to Control Data Pub. No. 60040800.

For the breakdown of Ault Power Supply(s), refer to Control Data Pub. No. 60107300.

For the breakdown of Teletype Punch, refer to Teletype Model BRPE II Punch.

For the breakdown of Cedar Paper 350 Tape Reader, refer to Cedar Engineering Publication 108A for Models A and B, and Cedar Engineering Publication 2209 for Models C, D, E and F.

For the breakdown of the Soroban Computeriter, refer to Soroban Model EC and ETC - Coder Unit and Soroban Model Decoder and Power Unit. Control Data Pub. No. 60008200 has the combination of the coder and decoder units.

ORDERING OF PARTS

When ordering Control Data Corporation parts, include the following information: Item number or Control Data drawing number (both if listed), description, quantity needed, equipment used on.

When ordering vendor parts use the procedure indicated by that vendor.

PARTS LIST

1604-A Console
(Final Assy Dwg. No. 120210)

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
8215	Adhesive Plastilock (25472) #604	AR
100754	Adjusting Plate, Switch Tray	4
100849	Angle, Connector, Typewriter	1
100781	Angle, Lock, Relay Chassis Assembly	2
100786	Angle, Long, Right Section	2
100787	Angle, Short, Right Section	2
101063	Angle, Stiffener, Right Section	4
100777	Angle, Switch Mount, Switch Panel Assembly, Right Section	1
100761	Angle, Trim, Hood Assembly	2
8551-17	Bearing, Sleeve-Flanged, Relay Chassis 1/4" I.D.	4
8149	Box, Conduit, junction	2
101062	Box, Punch Power Distribution Box Assembly, Right Section	1
100730	Brace, Light module, long	2
100811	Brace, Light Module, Short	2
101026	Bracket, Connector, J209014, J20902	1
101061	Bracket, Control Unit, Right Section	1
100740	Bracket, Front, Left, Frame Adjustment	1
100816	Bracket, Front, Right, Frame Adjustment	1
100741	Bracket, Hinge, Relay Chassis Assembly	4
100815	Bracket, Hold Down, Digital Switch	1
100952	Bracket, Magnetic Catch, Center Section	1
100929	Bracket, Mounting, Terminal Board, Center Section	1
100995	Bracket, Mounting, Time Meter	1
100956	Bracket, Mounting, Transformer, Center Section	1
100817	Bracket, Rear, Left Frame Adjustment	1
100719	Bracket, Rear Right Frame Adjustment	1
100759	Bracket, Switch, Rear, Lower Control Panel	1
100760	Bracket, Switch, Rear, Upper and Center Control Panel	2
100736-1 through 100736-7	Bracket, Switch Mounting, Lettered, 16 Hole B1, B2, B3, B4, B5, B6, B7	1 ea.
100735-1, -2	Bracket, Switch Mounting, Lettered, 24 Hole	1 ea

FORM CA 108

PARTS LIST

DATE: _____

CDC – DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
107732-1, -2	Bracket, Switch Mounting, Lettered, 25 Hole	1 ea
100737	Bracket, Switch Mounting, Lettered, 26 Hole	1
100700	Bracket, Switch Tray, Used on S05	1
100783	Bracket Zee, Center Section	2
8222	Bumper; Rubber	2
100733-1	Button, Module Switch, Light Blue	76
100733-2	Button, Module Switch, Medium Blue	75
100733-3	Button, Module Switch, White	11
100733-4	Button, Module Switch, Dark Blue	75
6602	Button, Switch Push, Used on 6601, Red	1
6604	Button, Switch Push, Used on 6601, Blue	1
100960	Buzzer Assembly	1
245335	Buzzer, 115 vac, 60 cycle	1
100882-1	Cable Assembly, Extension, Power (J20003)	1
100882-2	Cable Assembly, Extension, Power (J20005)	1
100879	Cable Assembly, Logic, Typewriter	1
100880	Cable Assembly, Power, Typewriter	1
100881	Cable Assembly, Punch	1
100904	Cable Assembly, Reader Logic	1
101042-1, -2	Cable Assembly, Rotary Switch	1 ea
100286-7	Cable Assembly, Special Purpose, 24 Pin Connector	1
5205	Capacitor Fixed, Electrolytic, 200 uf, 50 wvde	2
100767	Card, Deck, Divider, Digital Switch	5
8241	Catch, Magnetic	1
120209	Center Section Assembly	1
100893	Clamp, Assembly, Capacitor	2
8637-7	Clamp, Cable, Electrical 1/2" I. D.	6
8637-13	Clamp, Cable Electrical 7/8" I. D.	7
6103	Clamp, Connector, used on 6102	1
6105	Clamp, Connector, used on 6104	1
245145	Clamp, Connector, Rack and Panel, Used on 245120-1	1
8164	Clamp, Level Joint, Rim Clenching	6

FORM CA 108

PARTS LIST

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
245181-1	Connector, Flexible Conduit 1/2" Dia	2
6128	Connector, Plug, 50 Socket	1
100940	Connector, Panel Assembly 20300	1
6102	Connector, Plug, 4 Socket	1
6104	Connector, Plug,17 Socket	1
245144	Connector, Plug, Midget, 3 Pin Twistlock	2
245139-1	Connector, Plug, 24 Pin (Potting)	6
6118	Connector, Plug, 2 Pole Universal Receptacle	1
245147-1	Connector, Rack and Panel Socket Contacts	1
6119	Connector, Receptacle, Power, Typewriter	1
6123	Connector, Receptacle, 2 wire female J20006	1
6127	Connector, Receptacle, 50 pin	1
8151	Connector, Receptacle, Electrical, Right & Left Section Assemblies	2
245120-1	Connector, Receptacle, 4 Hole Panel Mount, 24 Socket	40
245300-1	Connector, Receptacle, 5 Pin, J20315	1
245300-2	Connector, Receptacle, 5 Pin, J20316	1
245300-3	Connector, Receptacle, 5 Pin, J20317	1
245124	Connector, Receptacle, Female, Twistlock	2
120210	Console Assembly, Final	1
8187	Contact, Electrical, Digital Switch Assembly	5
100717	Control Panel Assembly	1
100716	Control Panel Lettered	1
100947	Cover, Connector, Plate, Access, Center Section	1
100701	Cover, Digital Switch	1
100891	Cover, Terminal, Center Section	1
100789	Dial, Digital Switch	5
245050-10	Digital Display, 8 Character	4
100734	Digital Switch Assembly, Switch Tray S05	1
118035	Diode, Silicon Rectifier	12
100784-1, -2	Door, Hood	1 ea
100803-1, -2	Door, Rear, Center Section	1 ea
101055	Door, Punch	1

FORM CA 108

PARTS LIST

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
100953	Drawer Assembly, Center Section	2
100926-1	Drawer Assembly, Left Section	3
100926-2	Drawer Assembly, Right Section	2
100810	Drawer, Front, Center Section	2
100954	Drawer Weldment, Front, Center Section	2
100927-1	Drawer Weldment, Left	3
100927-2	Drawer Weldment, Right	2
120201	Emblem, Console	1
8127	Fastener, Stud, Turnlock	2
100796	Frame Assembly, Digital Switch, Center Section	1
100846	Frame, Control Assembly, Center Section	1
100729	Frame, Control Panel	1
100883	Frame, Drill Assembly, Digital Switch	1
100984	Frame, Left Section	1
100728-1	Frame, Light Module, Control Frame	1
101050	Frame, Right Section	1
100934-2	Frame, Sub-assembly, Center Section	1
245130-1	Fuse, Cartridge, 15 amp, F04	1
245130-8	Fuse, Cartridge, 8 amp F01 thru F03	4
245129-20	Fuse, Fast Action, 1 amp	1
245362-2	Fuse, Fast Action, 5 amp	1
245086	Fuseholder, indicating	6
100936	Fuse, Panel Assembly, Center Section	1
8135	Grommet, Rubber, Right and Left Section	3
8136	Grommet, Rubber, Right Section	1
100744	Hinge Angle Assy, Relay Chassis	4
8121	Hinge, Butt	6
101057	Hinge, Punch	2
100743	Hinge, Relay Chassis	4
120208	Hood Assembly	1
6121	Hood, Potting, Used on Connector 6128	1
100999	Housing, Emblem	1
101038	Housing, Punch	1
245050-2	Indicator Assembly, Digital Display, 5 unit	8

FORM CA 108

PARTS LIST

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
245050-8	Indicator Assembly, Digital Display, 3 unit	1
245050-10	Indicator Assembly, Digital Display, 8 unit	4
100726-1	Knob, Lever Switch, Center Section, Red	6
100726-3	Knob, Lever Switch, Center Section, Blue	6
100726-4	Knob, Lever Switch, Switch Tray, Center Section, Black	2
6805	Lamp, Incandescent, 6.3v, Used in Digital Display	800
8174	Latch, Magnetic	3
100983	Left Section Assembly	1
100951	Leg, Center Section	1
100965	Leg, Right & Left Section	4
100725-1	Lens, Digital Display Unit, 5-7/16 Lg	1
100725-2	Lens, Digital Display Unit, 8-9/16 Lg	1
100725-3	Lens, Digital Dispaly Unit, 12-1/4 Lg	1
100875	Light Module Modification, 3 Unit	1
100876-1 through 100876-7	Light Module Modification, 5 Unit, LH and RH Tab	1 ea
100877-1, -2	Light Module Modification, 8 Unit, RH Tab, Left A Block, Left Q Block	1 ea
100724	Light Module Modification, 5 Unit, LH Tab, Control Panel	1 ea
100878-1, -2	Light Module Modification, 8 Unit, LH Tab, Left A Block, Left Q Block	1
8177	Lug, Terminal, Solder, No. 6	30
8376	Nut, Speed, Sheet, Spring	8
101024	Panel, Access, Rotary Stepping Switch Assembly	1
100942	Panel, Connector, Center Section	1
101028	Panel, Center, Switch Panel Assembly, Right Section	1
100941	Panel, Connector, Lettered, 20300	1
100985	Panel, End, Inner, Left Section	1
101030	Panel, End, Inner, Right Section	1
100966-2	Panel, End, Outer, Left Section	1
100966-1	Panel, End, Outer, Right Section	1
100969	Panel, Front, Right Section	1
100946	Panel, Fuse Assembly	1
100928	Panel, Inner, Drawer Assembly	5

FORM CA 108

PARTS LIST

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
100852	Panel, Left Switch Panel Assembly, Right Section	1
100847	Panel, Rear, Center Section, Leg	1
100986	Panel, Rear, Left Section	1
101056	Panel, Rear, Right Section	1
100938	Panel, Relay Chassis, Lettered, 20100	1
100949	Panel, Relay, Chassis, Lettered, 20200	1
100851	Panel, Right, Switch Panel Assembly, Right Section	1
100853	Panel, Upper, Switch Panel Assembly, Right Section	1
100862	Pin, Hinge, Relay Chassis, and Punch	6
8227	Plastic, Sheet, Used as Lenscreen	AR
100973	Plate, Catch, Magnet	1
115109	Plate, Cover, Typewriter	1
100887	Plate, E-Block, Lettering, Center Section Numbered E20010 thru E20014	1
101035	Plate, E-Block, Mounting, Center Section	1
100792	Plate, Hinge, Center Section	8
100955	Plate, Lug, Identifier, Center Section, Numbered E20010 thru E20014	1
100961	Plate, Mounting, BV22ER	1
101068	Plate, Mounting, Switch, Center Section	1
100930	Plate, Mounting, Terminal, Center Section, Phenolic	1
100854-2	Plate, Serial, 1604-A	1
100702	Plate, Side, Digital Switch	2
101027	Plate, Support, Switch, Switch Panel Assembly, Right Section	1
101049	Plate, Tape Reader, Right Section	1
8150	Plate, Wall, Electrical, Right & Left Section	2
101037	Platform, Punch	2
118016	Power Supply, -28 vdc, regulated (Punch Power)	1
100812	Pull, Drawer, Drawer Assembly	10
1103	Punch, Paper Tape, 60 cps	1
8371	Receptacle, Friction Catch Stud #8	1
8128	Receptacle, Turnlock Fastener	2
6220	Relay, 24 vdc, SPDT, 10 amp, used on Punch Power Distribution box assembly	1
245132	Relay, 24 vdc, 2 PDT	186

FORM CA 108

PARTS LIST

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
245121	Relay, 24 vdc, 4 DPT	84
6208	Relay, time delay, 4 DPT	1
100937	Relay, Chassis Assembly, 20100	1
100948	Relay Chassis Assembly, 20200	1
100938	Relay Chassis, Lettered, Unit 01	1
100949	Relay Chassis, Lettered, Unit 02	1
245126-1	Resistor, Variable, Wire-Wound, L-PAD Attentuator, 6 ohms, 2 Watt, ± 15%	1
245370-1	Resistor, Variable; 500 ohm, 2 Watt	1
245000-31	Resistor, Fixed, Composition, 47 ohm, 1/4w	1
101046-2	Right Section Assembly	1
100790	Shaft, Digital, Switch	1
6117	Shall, Connector, Right Angle	1
101021	Shroud, Rotary Switch, Right Section	1
100943	Slide, Drawer, Modification, Center Section	2
8267	Slide Set, Drawer, Front, Center Section 11-7/8" travel	2
8268	Slide Set, Drawer, 20" Travel	5
6181	Socket, Octal, used on Punch Power Distr. Box Assy	1
245196	Socket, Relay, 10 Pin, used on Relay 245132	186
245122	Socket, Relay, 16-Pin, used on Relay 245121	84
6961	Speaker, 6-8 ohm	1
100925	Spline, Alignment, Top, 1/4" Plywood	3
101039	Spool, Tape	1
100764	Spring, Digital Switch	5
100745	Standoff, Transformer, Relay Chassis Assembly	4
8245	Steel, Sheet Corrosion, Resisting, Perforated, Left Section	AR
8140	Steel, Sheet, Perforated, Side-Staggered	AR
100944	Stop Assembly, Drawer, Center Section	2
6139	Strip, Lug Type, Terminal Board	2
100873	Strip, Marked, Light Module, 00-23	2
100872	Strip, Marked, Light Module, 00-14	8
100871	Strip, Marked, Light Module, 15-23	1
100874	Strip, Marked, Light Module, 24-47	2
101075	Strip, Tap, Tape Reader	2

FORM CA 108

PARTS LIST

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
245282-5	Strip, Terminal Board, 5 Contact	5
245282-8	Strip, Terminal Board, 8 Contact	4
8372	Stud, Friction Catch, 3/8" Dia Ball, Right Section Assy	1
8373	Stud Assembly, Turnlock Fastener	4
101022	Support, Switch, Left	1
101023	Support, Switch, Right	1
100987	Support, Typewriter, Left Section	1
100727	Switch, Key Lever, Modified	14
101034	Switch Panel Assembly, Right Section Reader Control	1
245411-6	Switch, Toggle 1-C, 1-C	14
6002	Switch, Toggle, DPST	1
6601	Switch, Push, S01, S02, Switch Tray	2
245410-1	Switch, Pushbutton, Momentary, Red	237
101029	Switch, Rotary - Relay Assembly	1
6217	Switch, Stepping, Rotary	1
100845	Switch Tray Assembly	1
118009	Tape Reader, (03998) Model 350	1
6964	Time Meter, 115 VAC, 60 cycle	1
101047	Top Assembly, Desk, Right Section	1
100945	Top, Desk, Center Section	1
100988	Top, Desk, Left Section	1
101048	Top, Desk, Right Section	1
100026	Transformer, Choke Assembly 4.5 amp	2
245313	Transformer, Filament, 6.3 v.c.t., 10 amp, 115v PRI	2
100714	Tray, Plate, Switch Tray	1
100762	Trim Assembly, Hood	1
100778	Trim, Switch Panel Assembly, Right Section	1
100797	Typewriter, Modification	1
118032	Typewriter, Monitor	1
100998	Well, Tape	1
100768	Wiper Contact, Digital Switch	5
241019	Wire List, Center Section	1

FORM CA 108

PARTS LIST

1604-A Main Cabinet (Final Assy
Dwg. No. 118625

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
8234	Adaptor, Junction Box 4 In. Octagon	3
100547	Angle, Baffle, Top Panel	2
100226	Angle, Door Stop, Outer	8
100421	Angle, Filter, Floor	4
100298	Angle, Inner Chassis Stop	8
100210	Angle, Mounting, Chassis Seal	4
100441	Angle, Mounting, Outer Door Latch	4
100442	Angle, Spring Retaining, Outer Door Latch	2
100649	Angle, Support, Bottom Panel	4
100572-1	Arm, Assembly, Chassis, Left	2
100574-1	Arm Assembly, Chassis, Right	2
118592	Baffle, Center, Right	1
100548	Bar, Chassis Arm	4
100022	Bar, Mounting, Connector	80
100255-1	Bar, Mounting, Connector, Short, 01-18	40
100255-2	Bar, Mounting, Connector, Short, 67-84	40
100263-1, -2	Bar, Mounting, Memory	8 ea
100240	Bar, Support, Memory	16
100391	Base, Stand-off, Card Spacer, Memory	80
8551-54	Bearing, Flange, Oilite	16
100424	Block, Chassis, Top	8
118601	Block, Connector, LH Angle	11
118603	Block, Connector, RH Angle	21
117563	Block, Hinge, Door	8
100438	Block, Outer, Door Latch, Long	2
100439	Block, Outer Door Latch, Narrow	2
100440	Block, Outer Door Latch, Wide	2
100470	Block, Support, Inner Chassis	2
100471	Block, Support, Outer Chassis, Right	2
100478	Block, Support, Outer Chassis, Left	2
8149	Box, Conduit, Junction	1
100589	Box, Conduit, Modification	2
8288	Box, Connector, Electrical	2
100343	Box, Timer	1

FORM CA 108

PARTS LIST

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
100466	Bracket, Bottom Panel	4
100300	Bracket, Chassis Inner	8
100251-1, -2	Bracket, Chassis Outer	8 ea
100359-1, -2	Bracket Fuse Panel	1 ea
100275	Bracket, Spring, Inner Door Latch	4
101303	Bracket, Trim	4
118576	Bumper, Door	24
8120	Bumper, Rubber	24
100605-1	Cable Assembly, Chassis, Power	8
241018-1	Cable Installation, Inter-chassis	1
241008-1	Card Placement, Chassis 10100	1
241009-1	Card Placement, Chassis 10200	1
241010	Card Placement, Chassis 10300	1
241011	Card Placement, Chassis 10400	1
241012	Card Placement, Chassis 10500	1
241013	Card Placement, Chassis 10600	1
241014	Card Placement, Chassis 10700	1
241015	Card Placement, Chassis 10800	1
100364-1	Card Spacer Assembly 01-18, C-D	8
100375-1	Card Spacer Assembly, 01-17, D-E	8
100364-2	Card Spacer Assembly, 01-18, E-F	8
100375-2	Card Spacer Assembly, 01-18, F-G	8
100364-3	Card Spacer Assembly, 01-18, G-H	8
100367-1	Card Spacer Assembly, 01-84, A	8
100368-3	Card Spacer Assembly, 01-84, A-B	8
100394-1	Card Spacer Assembly, 01-84, B-C	8
100393-1	Card Spacer Assembly, 01-84, H-I	8
100366-1	Card Spacer Assembly, 01-84, I-J	8
100366-2	Card Spacer Assembly, 01-84, J-K	8
100366-3	Card Spacer Assembly, 01-84, K-L	8
100366-4	Card Spacer Assembly, 01-84, L-M	8
100366-5	Card Spacer Assembly, 01-84, M-N	8
100366-6	Card Spacer Assembly, 01-84, N-O	8

FORM CA 108

PARTS LIST

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
100369-1	Card Spacer Assembly, 01-84, O-P	8
100367-2	Card Spacer Assembly, 01-84, P	8
100365-1	Card Spacer Assembly, 67-84, C-D	8
100363-1	Card Spacer Assembly, 67-84, D-E	8
100365-2	Card Spacer Assembly, 67-84, E-F	8
100363-2	Card Spacer Assembly, 67-84, F-G	8
100365-3	Card Spacer Assembly, 67-84, G-H	8
100293	Catch, Chassis, Inner, Left	2
100306	Catch, Chassis, Inner, Right	2
100222	Catch, Door, Outer	4
100289	Catch, Inner Door Latch, Left	2
100273	Catch, Inner Door Latch, Right	2
100446	Catch, Outer Door Latch, Left	2
100444	Catch, Outer Door Latch, Right	2
8415	Catch, Luggage	24
100270	Channel, Mounting, Left, Thermostat	1
100269	Channel, Mounting, Right, Thermostat	1
100376-1, -2	Channel, Stand-off Left	4 ea
100377-1, -2	Channel, Stand-off Right	4 ea
118637-1	Chassis Assembly, Wired, 10100	1
117620	Chassis Assembly, Wired, 10200	1
118607	Chassis Assembly, Wired, 10300	1
118609	Chassis Assembly, Wired, 10400	1
118621	Chassis Assembly, Wired, 10500	1
118611	Chassis Assembly, Wired, 10600	1
118613	Chassis Assembly, Wired, 10700	1
118615	Chassis Assembly, Wired, 10800	1
8637-7	Clamp, Loop, 1/2 ID	9
8363	Clip, Spring Tension	12
8252	Conduit, Metal, Flexible, 1/2" Dia	AR
245182-2	Connector, 90° angle, 1/2" Dia	2
245181-1	Connector, Cable, 1/2" Dia	4
8151	Connector, Receptacle, Duplex	2

FORM CA 108

PARTS LIST

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
245300-1	Connector, Receptacle, 11 Pin	1
245300-2	Connector, Receptacle, 11 Pin	1
245155-2	Connector, Receptacle, 14 Pin	8
245154-1	Connector, Receptacle, 14 Socket	16
245120-1	Connector, Receptacle, 24 Socket	257
100018	Connector, Receptacle, 30 Socket	4032
118582	Cover, Air Seal	2
118586-1	Cover Assembly, Air Seal	2
100339	Cover, Chassis	4
100268	Cover, Connector Hole	7
8235	Cover, Junction Box, 4" Dia	2
118035	Diode, Silicon Rectifier, Type 4300	144
118036	Diode, Silicon Rectifier, Type 4301	48
118565-1	Door Assembly, Bay 1 and 3	2
118564	Door, Bay 1 and 3	2
118635-1	Door Assembly, Bay 2 and 4	2
118636	Door, Bay 2 and 4	2
100698	Edging, Door	4
100689	Emblem (19 inch)	2
245299-2	Enclosure, Terminal, Switch	4
8154-5	Filter, Air, Washable	4
100512	Floor, False	2
100281	Frame, Chassis, Inner	4
118552-1	Frame, Chassis, Lettered, Chassis #1	1
100481-2	Frame, Chassis, Lettered, Chassis #2	1
100480-1	Frame, Chassis, Lettered, Chassis #3	1
118555-1	Frame, Chassis, Lettered, Chassis #4	1
118552-2	Frame, Chassis, Lettered, Chassis #5	1
100481-4	Frame, Chassis, Lettered, Chassis #6	1
100480-3	Frame, Chassis, Lettered, Chassis #7	1
118555-2	Frame, Chassis, Lettered, Chassis #8	1
118551	Frame, Chassis, Outer	4
100215	Frame, Drilling, Main Cabinet	1
118623	Frame, Drilling, Sub-Assembly	1

FORM CA 108

PARTS LIST

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
100225	Frame, Floor	2
245129-1	Fuse, Fast Action, 2 amp	24
245086	Fuseholder, indicating	24
8135	Grommet, Rubber	15
8166	Grommet, Rubber 1/2" I.D.	5
100515	Guide, Air, Left	2
100514	Guide, Air, Right	2
100278	Guide, Inner Door Latch, Left	2
100271	Guide, Inner Door Latch, Right	2
118584	Handle, Seal, Air	2
8454-4	Hinge, Butt, Continuous	AR
118570	Hinge, Door	8
118562	Hinge, Door, Small	8
101044	Identifier, Timer Box	1
8279	Jack, Tip	1
118572	Knob, Pin, Door Hinge	8
100297-1	Latch, Assembly, Inner Door, Left	2
100296-1	Latch, Assembly, Inner Door, Right	2
100451-1	Latch, Assembly, Outer Door, Left	2
100450-1	Latch, Assembly, Outer Door, Right	2
100390	Leg, Stand-Off, Card Spacer, Memory	80
6807	Light Indicator	5
118625-2	Main Cabinet Assembly, Wired	1
118624	Main Cabinet, Sub-assembly	1
118629	Memory Core Assembly; 128 x 128 x 12	8
100691	Number (1) Casting	2
100692	Number (6) Casting	2
100693	Number (0) Casting	2
100694	Number (4) Casting	2
120197	Number (-) Casting	2
120198	Number (A) Casting	2
118591-2	Panel, Air-Seal, Left	1
118591-1	Panel, Air Seal, Right	1
118597-1	Panel, Baffle, End	1

FORM CA 108

PARTS LIST

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
118597-2	Panel, Baffle, End	1
118598	Panel, Baffle, End	1
100683	Panel, Bottom, Frame	2
100287	Panel, End, Inner, Frame	2
100264	Panel, Fuse, Lettered, F11-F23	1
100265	Panel, Fuse, Lettered, F31-F43	1
100266	Panel, Fise, Lettered, F51-F63	1
100267	Panel, Fise, Lettered, F71-F83	1
118571	Pin, Door Hinge	8
100571	Pin, Lock, Chassis Arm	4
100549	Pin, Pivot, Chassis Arm	4
245007-1	Pin, Taper	6144
8283-1	Plastic Strip, Adhesive Coated, Foam	AR
100585	Plate, Air Seal	2
118567	Plate, Chassis Arm	4
100216	Plate, Cover, Support Channel	2
100254	Plate, Cover, Lettered, Left	1
100253	Plate, Cover, Letered, Right	1
100317	Plate, Designation, Narrow, 01-21	48
100318	Plate, Designation, Narrow, 01-12	16
100319	Plate, Designation, Narrow, 13-72	16
100320	Plate, Designation, Narrow, 73-84	16
100321	Plate, Designation, Narrow, 19-66	16
100322	Plate, Designation, Narrow, 01-18	56
100409	Plate, Designation, Wide, 22-42	48
100410	Plate, Designation, Wide, 43-63	48
100411	Plate, Designation, Wide, 64-84	48
100413	Plate, Designation, Wide, 43-72	16
100414	Plate, Designation, Wide, 67-84	56
100329	Plate, Designation, Wide, 13-42	16
100258	Plate, Hinge, Short	8
100276	Plate, Inner Door Latch	4
100083	Plate, Mounting, Hinge	8
100443-1, -2	Plate, Outer Door Latch	2 ea

FORM CA 108

PARTS LIST

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
118602	Plate, Retainer, Gate, Block, Connector	32
118587	Plate, Seal, Center, Left	2
118596	Plate, Seal, Center, Right	2
100500	Plate, Serial	1
100279	Plate, Side, Inner Door Latch, Left	2
100272	Plate, Side, Inner Door Latch, Right	2
100459	Plate, Spacer, Chassis Arm	4
100448	Push, Outer Door Latch	4
8255-1	Register, Metal - Adjustable Louvers	4
118581	Retainer, Air Seal	2
118583	Retainer, Cable Air Seal	1
100492	Retainer, Chassis Seal	8
100244	Retainer, Seal, Outer	4
100301	Seal, Chassis, Center	4
100491-1,-2	Seal, Chassis, Neoprene	4 ea
100230	Seal, Chassis, Outer	4
100228	Sheet, Floor	4
100449	Sheet, Push Mounting, Outer Door Latch	2
100234-1	Sheild, Memory	8
100341	Shroud, Memory	8
100280	Skin, End, Cabinet	2
100200	Skin, Top Cabinet	1
100274	Slide, Inner Door Latch	4
100447	Slide, Outer Door Latch, Left	2
100445	Slide, Outer Door Latch, Right	2
100404	Spacer, Chassis Arm	8
100569	Spring, Chassis Arm	4
100396	Spring, Inner Latch, Long	4
100397	Spring, Inner Latch, Short	4
100485	Spring, Outer Door Latch, Long	4
100379	Spring, Outer Door Latch, Short	4
118573	Spring, Retainer, Pin, Door	8
100550	Stop Assembly, Pin, Chassis Arm	4
100546	Stop Bar, Chassis Arm, Left	2

FORM CA 108

PARTS LIST

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
100570	Stop, Inner Chassis	4
100250	Strip, Center Seal	4
100415	Strip, Tap	4
100402	Strip, Tap, Chassis Arm	4
100681	Support Assembly, Hinge Chassis, Inner	8
100679	Support Assembly, Hinge, Chassis, Outer, Lower	4
100680	Support Assembly, Hinge, Chassis, Outer, Upper	4
100522	Support, Filter Cover	4
245298	Switch, Roller Lever, SPDT	4
300923-2	Thermostat, 90°F, Flange Mounting	4
100342-1	Timer Assembly	1
6960	Timer	1
100043	Transformer, Choke Assembly	32
100302	Trim Assembly	2
8206	Tube, Rectangle, Aluminum	AR
241000-1	Wire Tabulation, Logic Chassis 10100	1
241001-1	Wire Tabulation, Logic Chassis 10200	1
241002	Wire Tabulation, Logic Chassis 10300	1
241003	Wire Tabulation, Logic Chassis 10400	1
241004	Wire Tabulation, Logic Chassis 10500	1
241005	Wire Tabulation, Logic Chassis 10600	1
241006	Wire Tabulation, Logic Chassis 10700	1
241007	Wire Tabulation, Logic Chassis 10800	1
241020-1	Wire Tabulation, Main Cabinet Subassembly	1
241017-1	Wire Tabulation, Power	1
241016-1	Wire Tabulation, Memory	1

FORM CA 108

PARTS LIST

Printed Circuit Card Assemblies
(Quantity is per 8 Chassis)

DATE: _____

CDC - DRAWING NUMBER	DESCRIPTION	QUANTITY EACH MACHINE
102058	Printed Circuit Card Assembly, Type 00	10
100050	Printed Circuit Card Assembly, Type 01A	175
102018	Printed Circuit Card Assembly, Type 11	195
102019	Printed Circuit Card Assembly, Type 12	648
102020	Printed Circuit Card Assembly, Type 13	232
102025	Printed Circuit Card Assembly, Type 14	505
102026	Printed Circuit Card Assembly, Type 15	259
102027	Printed Circuit Card Assembly, Type 16	244
102028	Printed Circuit Card Assembly, Type 21	884
102034	Printed Circuit Card Assembly, Type 22	292
102035	Printed Circuit Card Assembly, Type 23	84
102036	Printed Circuit Card Assembly, Type 24	36
102037	Printed Circuit Card Assembly, Type 31	700
102038	Printed Circuit Card Assembly, Type 32	267
102039	Printed Circuit Card Assembly, Type 33	58
102040	Printed Circuit Card Assembly, Type 41	338
102046	Printed Circuit Card Assembly, Type 42	65
102047	Printed Circuit Card Assembly, Type 43	50
102048	Printed Circuit Card Assembly, Type 44	28
102049	Printed Circuit Card Assembly, Type 45	24
103392	Printed Circuit Card Assembly, Type 50	34
120008	Printed Circuit Card Assembly, Type 51	266
100062	Printed Circuit Card Assembly, Type 52	527
100063	Printed Circuit Card Assembly, Type 53	444
101122	Printed Circuit Card Assembly, Type 54	266
102017	Printed Circuit Card Assembly, Type 55	198
102007	Printed Circuit Card Assembly, Type 56	399
103386	Printed Circuit Card Assembly, Type 59	68
102060	Printed Circuit Card Assembly, Type 61	195
100059	Printed Circuit Card Assembly, Type 62	200
102067	Printed Circuit Card Assembly, Type 65	8
102075	Printed Circuit Card Assembly, Type 66	2
102102	Printed Circuit Card Assembly, Type 67	44
103352	Printed Circuit Card Assembly, Type 73A	3
103375	Printed Circuit Card Assembly, Type 86	2

FORM CA 108

COMMENT SHEET

CONTROL DATA 1604-A COMPUTER
Customer Engineering Instruction Manual, Part 2, Maintenance
Pub. No. 60118800

FROM NAME : _____

BUSINESS ADDRESS : _____

COMMENTS: (DESCRIBE ERRORS, SUGGESTED ADDITION OR DELETION AND INCLUDE PAGE NUMBER, ETC.)

CUT ALONG LINE

NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE

FOLD

FOLD

FIRST CLASS
PERMIT NO. 8241

MINNEAPOLIS, MINN.

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION
8100 34TH AVENUE SOUTH
MINNEAPOLIS 20, MINNESOTA

ATTN: TECHNICAL PUBLICATIONS DEPT.
COMPUTER DIVISION
PLANT TWO

FOLD

FOLD

CUT ALONG LINE

STAPLE

STAPLE

►►CUT OUT FOR USE AS LOOSE-LEAF BINDER TITLE TAB

CONTROL DATA
CORPORATION
8100 34th AVE. SO., MINNEAPOLIS, MINN. 55440

PRINTED IN U.S.A.