1604 COMPUTER

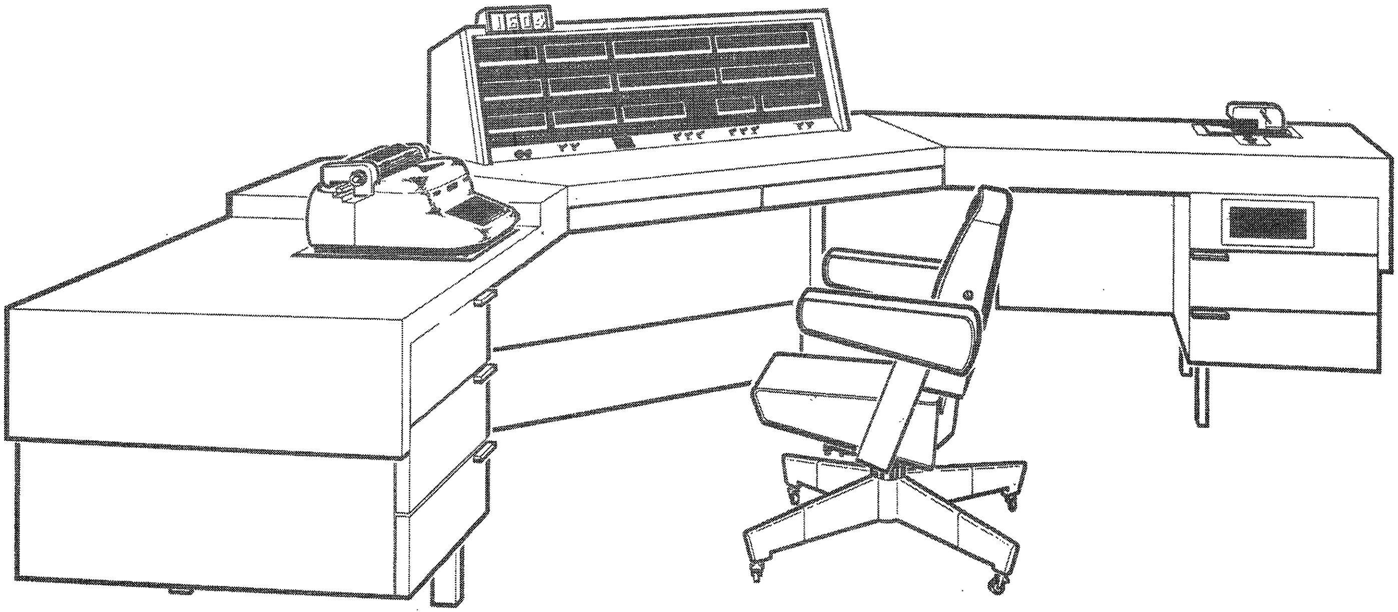
Volume 1:
DESCRIPTION AND OPERATION

INSTRUCTION BOOK

CONTROL DATA CORPORATION
MINNEAPOLIS, MINNESOTA

1604 COMPUTER

Volume 1:
DESCRIPTION AND OPERATION

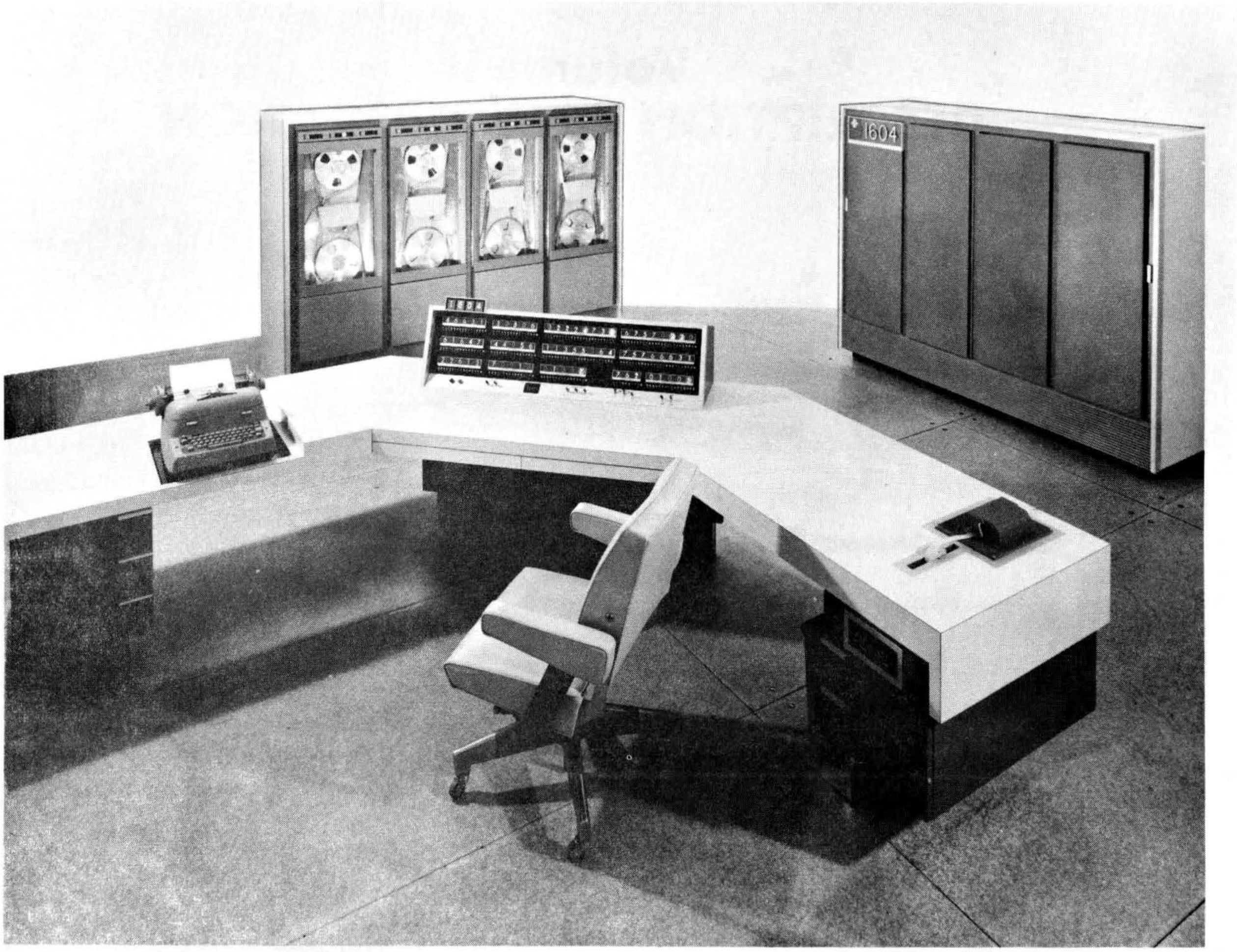


INSTRUCTION BOOK

PUBLICATION 031a

CONTROL DATA CORPORATION
MINNEAPOLIS, MINNESOTA

Revised 12/60



Typical 1604 Computer Installation

FOREWORD

The instruction book treats only the basic units in a 1604 system (main computer and console). Other external equipments that may be in a system have separate instruction books. The coverage of this book is indicated in the general table of contents.

Information included herein is subject to correction and change.

For additional copies or other information write to:

Control Data Corporation
Minneapolis 15, Minnesota

GENERAL TABLE OF CONTENTS

TABLE OF CONTENTS

CHAPTER ONE

GENERAL DESCRIPTION

The 1604 computer is a general purpose digital system. Having a large storage capacity, exceedingly fast computation and transfer speeds, and special provisions for input-output communication the 1604 is designed both to handle large-volume data processing and to solve large scale scientific problems. The compact equipment, constructed from solid-state components throughout, is suitable for use in a semi-permanent office environment. Characteristics and features are summarized below.

1604 CHARACTERISTICS AND FEATURES

Stored-program, general-purpose
digital computer

Parallel mode of operation

48-bit word length

Single address logic, 2 instructions
per 48-bit word

Operation code	6 bits
Designator	3 bits
Base Execution Address	15 bits

Six index registers

Indirect addressing

Magnetic core storage
32,768 48-bit words

Two alternately-phased, independent
16,384 word banks

4.8μsec effective cycle time
(representative program)

6.4μsec total cycle time

Input-output
Transmission of 48-bit words
Three separate buffer input channels
Three separate buffer output channels
High-speed transfer channel (input
and output). 4. 8 μsec per word.

Program interrupt

Console, includes:
Photo-electric paper tape reader
Paper tape punch
Electric typewriter
Display of register contents
translated to Arabic numerals
(octal)

Flexible list of instructions
Fixed-point arithmetic
(integer and fractional)

Floating-point arithmetic

Logical and masking operations

Indexing

Storage searching

Multiple-precision capability

Binary arithmetic
Modulus 2^{48}-1 (one's complement)
Parallel addition in 1.2 μsec
without access

Real-time clock

Completely solid-state
Diode logic
Transistor amplifiers

PHYSICAL DESCRIPTION

A typical 1604 installation consists of three units, as shown in the frontispiece; the console, main computer cabinet, and 1607 magnetic tape cabinet. An installation may also include other IBM equipments along with the 1605 adaptor which provides for connecting them to the computer.

The operator's panel and controls are on the console, as are the monitoring typewriter and paper tape reader. The paper tape punch is located just beneath the reader.

The main cabinet contains eight chassis hinged like pages for easy access (figure 1-1). Each chassis holds part of the core storage assembly and the printed circuit cards which plug into the chassis. The approximate dimensions of the cabinet are: height 5 feet 8 inches, width 7 feet 5 inches, depth, 2 feet 3 inches and weight 2200 pounds.

The magnetic tape cabinet which is comparable in size and weight to the main cabinet, contains four tape handlers and a data synchronizer.

The system operates from 208 vac, three-phase, 60 cycle primary power. A motor-generator set included in the system provides the required 400-cycle power. Total power consumption is approximately 5 kw.

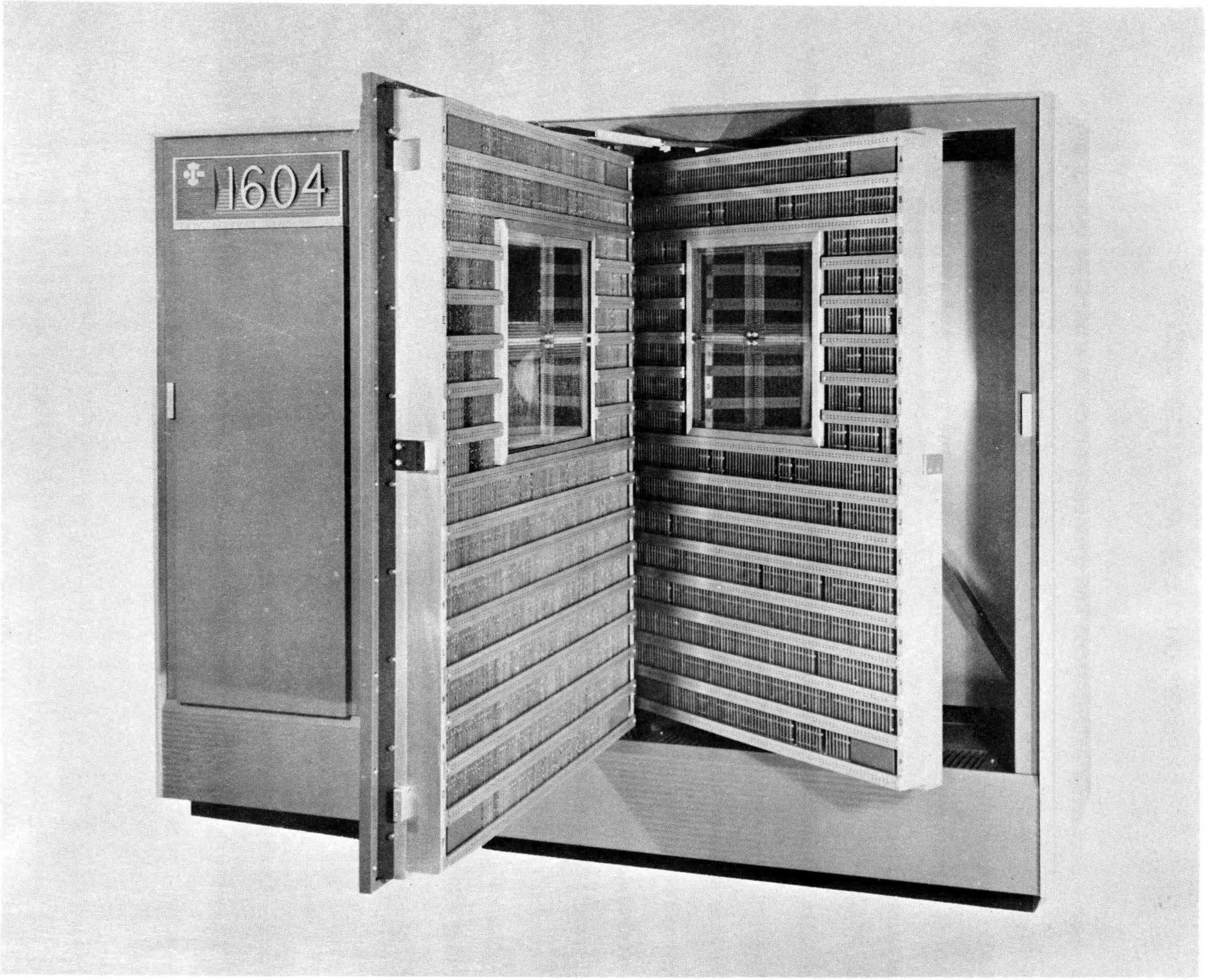


Figure 1-1. Interior of Main Cabinet.

GENERAL LOGICAL DESCRIPTION

The computer performs calculations and processes data in a parallel binary mode. This is done by the step-by-step execution of individual instructions of a program that is stored internally along with the data. The 62 different instructions and sub-instructions, from which a program is formed, can perform a wide variety of arithmetic, logical, searching and input-output operations.

Functionally the computer can be divided into four major sections: (1) Input-output section which provides the means of communication between the computer and the various external equipment, (2) Arithmetic section which performs both the arithmetic and logical operations required for the execution of instructions, (3) Storage section which provides internal storage for both data and instructions and (4) Control section which successively obtains the instructions from storage, then interprets each instruction to send the required commands to other sections. The control section coordinates and sequences all the operations which carry out the execution of an instruction. A simplified block diagram of the computer is shown in figure 1-2.

As in figure 1-2 the registers are identified by letter names. Table 1-1 gives the descriptive names of the registers and classifies them into two groups: operational and secondary. Operational registers are explicitly manipulated by the computer instructions. They store quantities from one instruction to another and their content is displayed on the operator's console.

The secondary registers are only implicitly manipulated by instructions. They do not necessarily store quantities from one instruction to another and are not a part of the console display.

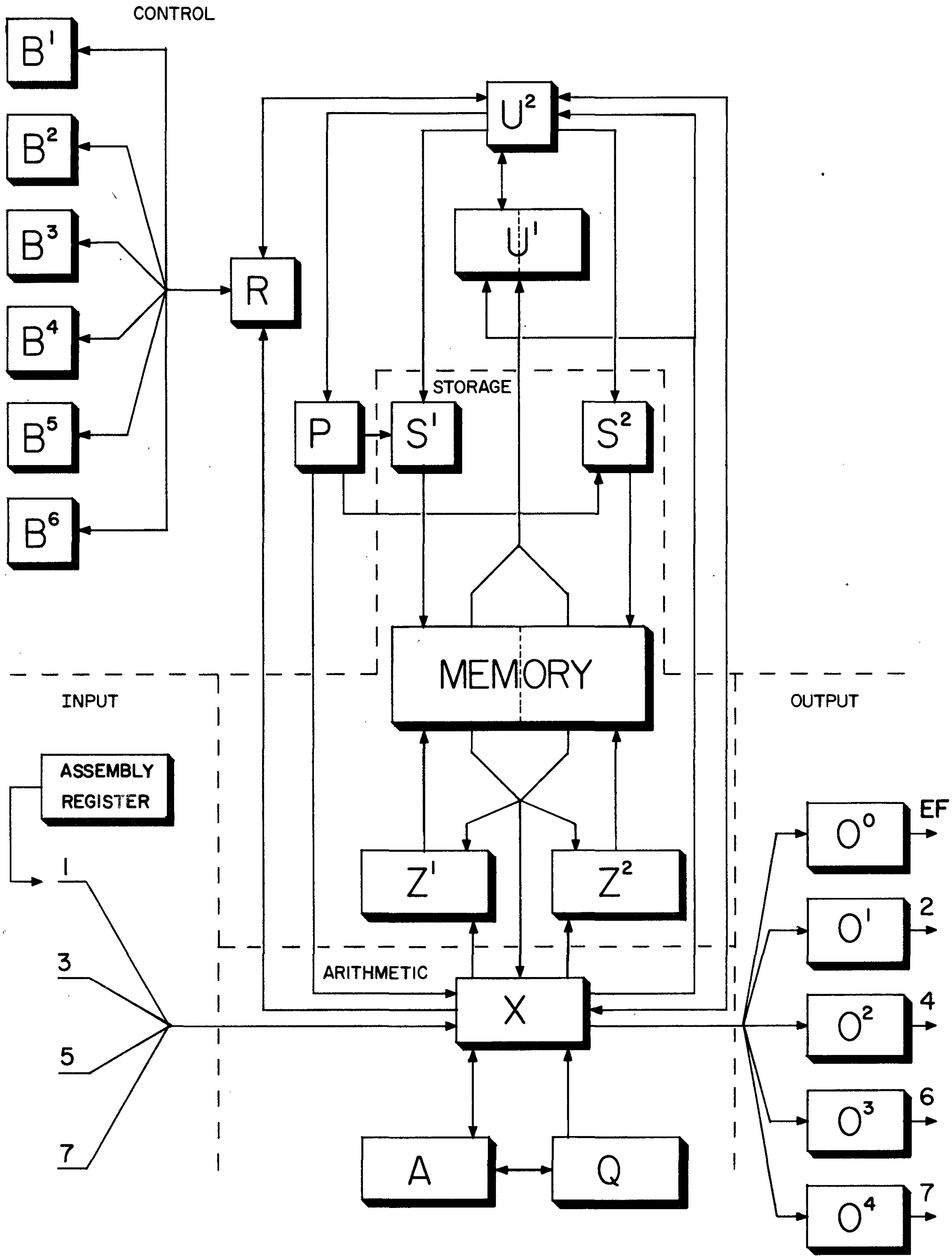


Figure 1-2. Simplified Block Diagram of the Model 1604 Computer.

TABLE 1-1. REGISTERS OF THE COMPUTER

Operational Registers	Function	Secondary Registers	Function
A register	Accumulator	U^2 register	Auxiliary Program Control
Q register	Auxiliary Arithmetic	S^1 register	Storage Address (even)
B registers	Index registers	S^2 register	Storage Address (odd)
(B^1 thru B^6)	(six provided)	Z^1 register	Storage Restoration (even)
P register	Program Address	Z^2 register	Storage Restoration (odd)
U^1 register	Program Control	R register	Address Buffer
		X register	Exchange
		O registers	Data Output
		(O^1 thru O^4)	

INPUT-OUTPUT SECTION

Input-output communication takes place with 48-bit words transmitted in a parallel manner. There are four independent channels which bring information into the computer storage via the X register.

Channels 1, 3, and 5 are used for buffer communication; channel 7 is used in transfer communication - a very high-speed method of exchanging data. (These two modes of communication are described later.) Information from the input equipment located at the console always is received from the assembly register via channel 1. Output information goes from storage to one of the output register, O^1, O^2, O^3 or O^4. These registers feed the four independent output channels 2, 4, 6 and 7. Control information to the various external equipments is set via O^0.

ARITHMETIC SECTION

The A register, or accumulator, is the principle arithmetic register. Nearly all arithmetic and logical operations make use of A. This register has provisions for the parallel addition of (X) to the contents of A. It can be shifted either separately or in conjunction with the Q register.

The Q register is an auxiliary arithmetic register which assists the accumulator in the performance of the more complicated arithmetic operations. It is used in combination with the X register in the formation of logical products. Q may be shifted either separately or in conjunction with A.

The X or exchange register is used in arithmetic operation as well as in most data transmission between various sections of the computer.

STORAGE SECTION

The 32,768 48-bit word magnetic core storage section is controlled by a two-phase timing system, each phase controlling one-half (16,384 48-bit words) of the total storage. All odd storage addresses reference one storage unit; all even addresses reference the other storage unit. The read access time of each section is 2.2 microseconds after which, without delay, the next operation is initiated. Each unit has a total cycle time of 6.4 microseconds. The storage cycles of the two sections overlap one another in the execution of a program, with the result that the effective cycle time is 3.2 microseconds when addresses of alternate memory banks are referenced. The average effective cycle time for random addresses is about 4.8 microseconds for a representative program.

The address register for the even storage unit is S^1; the address register for the odd storage unit is S^2. In addition, each unit has a storage restoration register (Z^1 and Z^2)

which holds the word to be written into a given storage location.

Words to be read out of either storage unit are entered in the X register, and from there transmitted to the appropriate register. Words to be entered or written into a storage unit are transmitted from X to the appropriate Z register, and thence to storage.

CONTROL SECTION

The control section directs the operations required to execute instruction and to exchange data with external equipment. The coordination of operations in the various sections of the computer is maintained by control. The major portion of this section consists of command sequences, static networks for sensing and storing special conditions, and several registers.

The P register functions as the program address counter. It provides continuity between the steps of a program by generating in a sequential manner the address of each new step. At the completion of a step, P is advanced to specify the address of the next step.

A program step is a pair of two 24-bit instructions, which together occupy one storage location as a 48-bit word.

The program control register is U^1 which holds a program step while the two instructions contained in it are executed. The 48-bit two-instruction word is taken from the storage location specified by P.

The six index registers (B^1 through B^6) provide for modification of the execution addresses of instructions in program loops. The R and U^2 registers assist in the addition of an index quantity to the execution address of an instruction in U^1.

INSTRUCTION FORMAT AND ADDRESS SYSTEM

A step in a computer program consists of two 24-bit instructions held in one 48-bit storage location. The step is treated by halves: "upper" instruction and "lower" instruction. The two instructions can be considered as logically separate entities of a program sequence. As a practical matter in program coding however, the pair is similar to a two-address instruction. The two single-address instructions are inseparable in the sense that the lower may not be executed without the upper. (When it is desired to place only one instruction in a step, the other half of the instruction should contain a "do-nothing" or pass instruction described later.)

The general format of a 24-bit instruction is given below.

6 bits	3 bits	15 bits
Operation Code	Designator	Base Execution Address

The operation code specifies the general character of the instruction. There are 62 such codes, identified 01 through 76 (octal). (Codes 0 and 77 are interpreted as faults and stop computation.)

For all but five of the instructions the designator value specifies an index register (B^1 through B^6). In some cases the content of B^b is added to the base execution address for indexing program loops. In the other cases B^b is the subject of the basic operation of the instruction. With $b = 0$ no register or address modification is specified. With $b = 7$ indirect addressing (discussed later) is used.

For jump instruction and the external function instructions the designator specifies a condition (j). The interpretation of j for each of these instructions is unique and thus given in the description of the instructions.

The base execution address holds the quantity that is basic to the execution of the instruction. Generally it denotes the storage location of the instruction operand. In some cases it is the operand, shift count, etc. The operation code determines the particular role and definition of the base execution address. After the addition of B^b to the base execution address, the sum is denoted M, Y, or K as the case may be. For example, M is equal to m plus B^b.

The four specific instruction types obtained by a variance in the interpretation of the designator and base execution address are given below.

Indexed Instruction with Storage Reference for Operand

f = 6 bits Operation Code	b = 3 bits Index	m = 15 bits Operand Address

Indexed instruction with self-contained operand

f = 6 bits Operation Code	b = 3 bits Index	y = 15 bits Operand

Shift Instructions

f = 6 bits Operation	b = 3 bits Index	k = 15 bits Shift Count

Jump and External Function Instructions

f = 6 bits Operation Code	j = 3 bits Condition	y = 15 bits Jump address, Buffer address, or EF code

LIST OF INSTRUCTIONS

The 62 computer instructions are given below with a brief description. The title line contains the octal code and format, verbal name, mnemonic code in parentheses, and symbolic description of the instruction. Abbreviations and symbols used in the list are defined as follows:

A_n	the bit in position n of the A register
(A)	Contents of A register
→	transmit to
EF	External Function
Exit	Proceed to next program step
Half Exit	Proceed to lower instruction of program step
LA	lower address - the lowest 15 bits of a 48-bit word or register. Execution address of lower instruction
( )'	complement of contents of register
UA	upper address - 15 bits forming address of an upper instruction

01 b k   A RIGHT SHIFT   (ARS)   Shift (A) Right by K

This instruction shifts the contents of the A register to the right the number of bit positions specified by the shift count, K. The sign bit is extended and the lowest-order bits are discarded as the shift is performed. Shift counts greater than 127 (decimal) are treated as shift faults; they produce an interrupt (if selected) and set an indicator which may be sensed by an external function instruction.

02 b k   Q RIGHT SHIFT   (QRS)   Shift (Q) Right by K

The contents of the Q register are shifted to the right the number of bit positions specified by the shift count, K. The sign bit is extended and the lowest-order bits are discarded as the shift is performed. Shift counts greater than 127 (decimal) are treated as shift faults; they produce an interrupt (if selected) and set an indicator which may be sensed by an external function instruction.

03 b k   AQ RIGHT SHIFT   (LRS)   Shift (AQ) Right by K

The contents of the A and Q registers are shifted to the right as one 96-bit register. The A register is considered as the leftmost 48-bits and the Q register as the rightmost 48-bits. The number of bit positions is specified by the shift count, K. The sign bit of the A register is extended as the shift is performed. The lowest-order bits of the A register replace the highest order bits of the Q register and the lowest-order bits of the Q register are discarded as the shift is performed. Shift counts greater than 127 (decimal) are treated as shift faults; they produce an interrupt (if selected) and set an indicator which may be sensed by an external function instruction.

04 b y   ENTER Q   (ENQ)   Y → Q, Extend Sign Y

The execution address portion, Y, of the instruction is entered into the Q register. The operand, Y, is entered into the Q register as a 14-bit quantity plus sign. The highest-order bit of Y is copied into the remaining higher-order bits of the Q register. No operand storage reference is made in this instruction.

05 b k   A LEFT SHIFT   (ALS)   Shift (A) Left by K

This instruction shifts the contents of the A register circularly to the left the number of bit positions specified by the shift count, K. The lower-order bits are replaced with the higher-order bits as the shift is performed. Shift counts greater than 127 (decimal) are treated as shift faults; they produce an interrupt (if selected) and set an indicator which may be sensed by an external function instruction.

06 b k Q LEFT SHIFT (QLS) Shift (Q) Left by K

The contents of the Q register are shifted circularly to the left the number of bit positions specified by the shift count, K. The lower-order bits are replaced with the higher-order bits as the shift is performed. Shift counts greater than 127 (decimal) are treated as shift faults; they produce an interrupt (if selected) and set an indicator which may be sensed by an external function instruction.

07 b k AQ LEFT SHIFT (LLS) Shift (AQ) Left by K

The contents of the A and Q registers are shifted circularly to the left as one 96-bit register. The number of bit positions is specified by the shift count, K. The rightmost bits of the A register are replaced with the leftmost bits of the Q register as the shift is performed. The rightmost bits of the Q register are replaced with the leftmost bits of the A register during the shift. Shift counts greater than 127 (decimal) are treated as shift faults; they produce an interrupt (if selected) and set an indicator which may be sensed by an external function instruction.

10 b y ENTER A (ENA) Y → A, Extend Sign Y

This instruction enters the execution address portion, Y, of the instruction into the A register. The A register is cleared and the operand Y is entered into the cleared A register as a 14-bit quantity plus sign. The highest-order bit of Y is copied into the remaining higher-order bits of the A register. No operand storage reference is made in this instruction.

11 b y INCREASE A (INA) $\left[Y + (A)\right] \rightarrow A$, Extend Sign Y

The operand, Y, is added to the previous contents of the A register. The operand Y is treated as a 14-bit quantity plus sign in this operation. The addition is performed as if

Y were a 48-bit quantity with the higher-order bits copies of the sign bit. No operand storage reference is made in this instruction. An overflow condition produces an interrupt (if selected) and sets an indicator which may be sensed by an external function instruction.

12 b m LOAD A (LDA) $(M) \rightarrow A$

Instruction 12 replaces the contents of the A register with an operand, contained in the location specified by the execution address. The A register is cleared, a storage reference is then made to obtain the 48-bit quantity designated; and the 48-bit operand is copied into the cleared A register. Negative zero may be loaded into the A register.

13 b m LOAD A, COMPLEMENT (LAC) $(M)' \rightarrow A$

This instruction replaces the contents of the A register with the complement of an operand contained in the location specified by the execution address. The A register is cleared, a storage reference is made to obtain the 48-bit quantity designated; and the 48-bit operand is complemented and entered into the cleared A register. Negative zero may be thus loaded into the A register.

14 b m ADD (ADD) $\left[(A) + (M)\right] \rightarrow A$

A 48-bit operand is added to the previous contents of the A register. A storage reference is made to obtain the 48-bit quantity contained in the location specified by the execution address, and the operand is then added to the previous contents of the A register. Occurrence of an overflow condition produces an interrupt (if selected) and sets an indicator which may be sensed by an external function instruction. A negative zero may be produced by this instruction if and only if both operands are initially negative zeros.

15 b m SUBTRACT (SUB) $\left[(A) - (M)\right] \rightarrow A$

Instruction 15 subtracts a 48-bit operand from the previous contents of the A register. A storage reference is made to obtain the 48-bit quantity contained in the location specified by the execution address, and the operand is then subtracted from the previous contents of the A register. An overflow condition produces an interrupt (if selected) and sets an indicator which may be sensed by an external function instruction. A negative zero may be produced by this instruction if the initial content of A is negative zero and the quantity in storage is a positive zero.

16 b m LOAD Q (LDQ) $(M) \rightarrow Q$

The contents of the Q register are replaced with an operand contained in the location specified by the execution address. The Q register is cleared, a storage reference is made to obtain the 48-bit quantity designated, and the 48-bit operand is then entered into the cleared Q register. Negative zero may be loaded in Q.

17 b m LOAD Q, COMPLEMENT (LQC) $(M)' \rightarrow Q$

This instruction replaces the contents of the Q register with the complement of an operand contained in the location specified by the execution address. The Q register is cleared, a storage reference is then made to obtain the 48-bit quantity specified, and the 48-bit operand is complemented and entered into the cleared Q register. A negative zero may be thus loaded in Q.

20 b m STORE A (STA) $(A) \rightarrow M$

The contents of the A register are stored at the storage location specified by the execution address. The contents of the A register are not modified by this instruction.

21 b m STORE Q (STQ) (Q) → M

Instruction 21 stores the contents of the Q register at the storage location specified by the execution address. The contents of the Q register are not modified by this instruction.

22 j m A JUMP (AJP) Jump to m

This instruction has eight subinstructions which cause a change in the program sequence because of a specified condition of the A register. The index registers are not used for address modification in this instruction. The jump designator, j, in the instruction specifies which subinstruction is to be performed. In the jump condition both negative and positive zero are treated as zero.

The subinstructions and the conditions required to cause a jump in the program sequence are as follows:

22 0 m - Jump if the A register content is zero
22 1 m - Jump if the A register content is not zero
22 2 m - Jump if the A register content is positive
22 3 m - Jump if the A register content is negative
22 4 m - Return jump if the A register content is zero
22 5 m - Return jump if the A register content is not zero
22 6 m - Return jump if the A register content is positive
22 7 m - Return jump if the A register content is negative

23 j m Q JUMP (QJP) Jump to M

This instruction has eight subinstructions which cause a change in program sequence because of a specified condition of the Q register. The index registers are not used for address modification in this instruction. The jump designator, j, in the instruction specifies which subinstruction is to be performed. In the jump conditions both negative and positive zero are treated as zero.

The subinstructions and the conditions required to cause a jump in the program sequence are as follows:

23 0 m - Jump if the Q register content is zero

23 1 m - Jump if the Q register content is not zero

23 2 m - Jump if the Q register content is positive

23 3 m - Jump if the Q register content is negative

23 4 m - Return jump if the Q register content is zero

23 5 m - Return jump if the Q register content is not zero

23 6 m - Return jump if the Q register content is positive

23 7 m - Return jump if the Q register content is negative

24 b m MULTIPLY INTEGER (MUI) (A) (M) → QA

Instruction 24 forms a 96-bit product from two 48-bit operands. The multiplier must be loaded into the A register prior to the execution of this instruction. The execution address specifies the location of the multiplicand in storage. The resulting product is contained in the QA register as a 96-bit quantity. If the operands are considered as integers, the product is correctly positioned as an integer in the QA register (the higher-order bits are in Q and the lower-order bits in A).

25 b m DIVIDE INTEGER (DVI) $\left[(QA) / (M)\right] \rightarrow A$; Remainder = Q_f

Instruction 15 divides a 96-bit integer dividend by a 48-bit integer divisor. The 96-bit dividend must be formed in the QA register prior to the execution of this instruction; the 48-bit divisor is read from the storage specified by the execution address. The quotient is formed in the A register, and the remainder is left in the Q register at the end of the operation. The dividend and remainder bear the same algebraic sign. A divide overflow produces an interrupt (if selected) and sets an indicator which may be sensed by an external function instruction.

NOTE: In the case of Integer Multiply and Divide, it should be noted that the position of the most-significant bits in the product and dividend differ from the usual positioning of bits in the AQ register. Since the most-significant digits are found in Q, this combined use of A and Q is referred to as QA.

26 b m MULTIPLY FRACTIONAL (MUF) $\left[(A)\ (M)\right] \rightarrow AQ$

This instruction forms a 96-bit product from two 48-bit operands. All quantities involved in this operation are treated as fractions with the binary point immediately to the right of the sign digit. The multiplier must be loaded into the A register prior to the execution of this instruction; the multiplicand is read from the storage location specified by the execution address. The product is formed in the AQ register and the multiplier is discarded in the multiplication process.

27 b m DIVIDE FRACTIONAL (DVF) $\left[(AQ)\ /\ (M)\right] \rightarrow A$; Remainder = Q_f

Instruction 27 divides a 96-bit quantity by a 48-bit divisor. All quantities involved in this operation are treated as fractions with the binary point immediately to the right of the sign digit. The 96-bit dividend must be loaded into the AQ register prior to the execution of this instruction; the 48-bit divisor is read from the storage location specified by the execution address. At the end of the operation the quotient is left in the A register. The remainder and the dividend bear the same algebraic sign. A divide overflow produces an interrupt (if selected) and sets an indicator which may be sensed by an external function instruction.

30 b m FLOATING ADD (FAD) $\left[(A) + (M)\right] \rightarrow A$

Instruction 30 forms the sum of two 48-bit quantities which are packed in floating-point format. An operand is read from the storage location specified by the execution address and is added to the previous contents of the A register. The result is normalized and rounded and left in the A register at the end of the operation, and the Q register contains the residue from the rounding operation at the end of the sequence. Floating-point range faults (exponent overflow or underflow) produce an interrupt (if selected) and set an indicator which may be sensed by an external function instruction.

31 b m FLOATING SUBTRACT (FSB) $\left[(A) - (M)\right] \rightarrow A$

An operand in floating-point format is subtracted from the previous contents of the A register, also in floating-point format. The operand is read from the storage location specified by the execution address. The result is normalized and rounded in the A register, and the residue from the rounding operation is left in the Q register at the end of the sequence. A floating-point range fault produces an interrupt (if selected) and sets an indicator which may be sensed by an external function instruction.

32 b m FLOATING MULTIPLY (FMU) $\left[(M)\ (A)\right] \rightarrow A$

This instruction forms the product of an operand in floating-point format with the previous contents of the A register, also in floating-point format. The operand is read from the storage location specified by the execution address. The result is rounded and normalized in the A register, and the residue from the rounding operation is left in the Q register at the end of the sequence. A floating-point range fault produces an interrupt (if selected) and sets an indicator which may be sensed by an external function instruction.

33 b m FLOATING DIVIDE (FDV) $\left[(A)\ /\ (M)\right] \rightarrow A$

This instruction forms the quotient of two 48-bit quantities in floating-point format. The dividend must be loaded into the A register prior to the execution of this instruction; the divisor is read from the storage location specified by the execution address. The quotient is rounded and normalized in the A register at the end of the operation, and the residue from the rounding operation is left in the Q register at the end of the operation. A floating-point range fault produces an interrupt (if selected) and sets an indicator which may be sensed by an external function instruction.

34 b k SCALE A (SCA) A left until (A) = 1/2 or K = 0 Reduce K by one per shift; $K_f \rightarrow B^b$

Instruction 34 shifts the quantity in the A register circularly to the left until the most-significant digit is immediately to the right of the sign digit. The shift count K is reduced by the number of bit positions shifted; the shift is terminated if K becomes zero before the normalizing operation is completed. In any event the reduced shift count is then entered into the designated index register.

35 b k SCALE AQ (SCQ) AQ left until (AQ) =1/2 or K = 0 Reduce K by one per shift; $K_f \rightarrow B^b$

The quantity in the AQ register is shifted circularly to the left until the most-significant digit is immediately to the right of the sign digit. The shift count K is reduced by the number of bit positions shifted; the shift is terminated if K becomes zero before the normalizing operation is completed. In any event the reduced shift count is then entered into the designated index register.

36 b m STORAGE SKIP (SSK) (M_{47}) Neg: Exit
(M_{47}) Pos: Half Exit

This instruction senses the sign digit of the quantity in the storage location designated by the execution address. If the quantity is negative, an exit is performed; if the quantity is positive, a half exit is performed. None of the quantities in the operational registers are modified by this instruction.

37 b m STORAGE SHIFT (SSH) (M_{47}) Neg: Exit, Shift Left One
(M_{47}) Pos: Half Exit Shift Left One

Instruction 37 senses the sign digit of the quantity in the storage location designated by the execution address. If the quantity is negative, an exit is performed if the quantity is positive, a half exit is performed. In either case the quantity in storage is then shifted circularly to the left one bit position. None of the quantities in the operational registers are modified by this instruction.

40 b m SELECTIVE SET (SST) Set (A_n) = 1 for (M_n) = 1

Individual bits of the A register are set to one where there are corresponding ones in the quantity in the storage location designated by the execution address. This is a bit-by-bit function and does not involve normal addition. Bits in the accumulator corresponding to zeros in the operand are not modified.

41 b m SELECTIVE CLEAR (SCL) Clear (A_n) to zero for (M_n) = 1

This instruction clears individual bits of the A register to zero where there are corresponding ones in the quantity in the storage location designated by the execution address. This is a bit-by-bit function and does not involve normal addition. Bits in A corresponding to zeros in the operand are not modified.

42 b m SELECTIVE COMPLEMENT (SCM) Complement (A_n) for (M_n) = 1

Individual bits of the A register are complemented where there are ones in the quantity in the storage location designated by the execution address. This is a bit-by-bit function and does not involve normal addition. Bits in A corresponding to zeros in the operand are not modified.

43 b m SELECTIVE SUBSTITUTE (SSU) $(M_n) \rightarrow A_n$ for $(Q_n) = 1$

Instruction 43 substitutes portions of an operand in the A register using the Q register contents as a mask. This may be considered in two steps. Individual bits of the A register are cleared to zero where there are ones in corresponding bits of the Q register. Then those same individual bits of the A register are replaced with corresponding bit values from the storage location specified by the execution address.

44 b m LOAD LOGICAL (LDL) $L\ (Q)\ (M) \rightarrow A$

The A register is loaded with the bit-by-bit logical product of the Q register contents and the quantity in the storage location designated by the execution address.

45 b m ADD LOGICAL (ADL) $\left[(A) + \ L(Q)\ (M)\right] \rightarrow A$

Instruction 45 adds to the A register contents the logical product of the Q register contents and the quantity in the storage location designated by the execution address. This is a normal addition of the selected portion of the operand with all other bits interpreted as zero. Occurrence of an overflow condition produces an interrupt (if so selected) and sets an indicator which may be sensed by an external function instruction.

46 b m SUBTRACT LOGICAL (SBL) $\left[(A) - L(Q)\ (M)\right] \rightarrow A$

This instruction subtracts from the A register contents the logical product of the Q register contents and the quantity in the storage location designated by the execution address. This is a normal subtraction operation for the selected portion of the operand with all other bits interpreted as zero. Occurrence of an overflow condition produces an interrupt (if selected) and sets an indicator which may be sensed by an external function instruction.

47 b m STORE LOGICAL (STL) L(Q) (A) → M

This instruction stores the logical product of the A register and the contents of the Q register at the storage location specified by the execution address. Neither the A nor the Q register contents are modified by this instruction.

50 b y ENTER INDEX (ENI) $y \rightarrow B^b$

Instruction 50 replaces the contents of the designated index register with the operand y contained in the instruction itself. No storage reference is made in this instruction. If zero is used as the index designator, this instruction becomes the pass instruction.

51 b y INCREASE INDEX (INI) $\left[y + (B^b)\right] \rightarrow B^b$

The operand y is added to the contents of the designated index register. The addition is performed modulus 2^{15} minus one. No storage reference is made in this instruction.

52 b m_u LOAD INDEX (upper) (LIU) $(m_{UA}) \rightarrow B^b$

This instruction replaces the contents of the designated index register with the address from the upper instruction at the designated storage location.

53 b m_l LOAD INDEX (lower) (LIL) $(m_{LA}) \rightarrow B^b$

Instruction 53 replaces the contents of the designated index register with the address from the lower instruction at the designated storage location.

54 b y INDEX SKIP (ISK) $(B^b) = y$: Exit Clear B^b
$(B^b) \neq y$: Adv (B^b), Half Exit

This instruction compares the quantity in the designated index register with the operand, y. If the two quantities are equal, then the designated index register is cleared to zero and an exit is performed. If the quantity in the index register is not equal to y, then the quantity in the index register is increased one count and a half exit is performed.

55 b m   INDEX JUMP   (IJP) $(B^b) \neq 0$: Reduce (B^b), Jump to m
$(B^b) = 0$: Execute Next Instruction

Instruction 55 examines the quantity in the designated index register. If this quantity is not zero, then the quantity is reduced one count and a jump is executed to the base execution address. If this quantity is zero, then the present program sequence is continued.

56 b m_U STORE INDEX (Upper)   (SIU) $(B^b) \rightarrow m_{UA}$

The quantity in the designated index register is stored in the address portion of the upper instruction contained in the storage location specified by the base execution address. The remaining bits at the specified storage location are not modified in this operation. This instruction effectively inserts an address in the first instruction at the specified storage location.

57 b m_L STORE INDEX (lower)   (SIL) $(B^b) \rightarrow m_{LA}$

This instruction stores the quantity in the designated index register in the address portion of the lower instruction contained in the storage location specified by the base execution address. The remaining bits at the specified storage location are not modified in this operation. This instruction effectively inserts an address in the second instruction at the specified storage location.

60 b m_U SUBSTITUTE ADDRESS (upper)   (SAU) $(A_0 - 14) \rightarrow M_{UA}$

The address portion of the upper instruction word in the storage location designated by the execution address is replaced with the lowest-order 15 bits of the A register contents. The remaining bits of the designated word in storage are not modified by this operation. This instruction effectively inserts an address in the first instruction at the designated storage location. The contents of A are, not modified by this instruction.

$61\ b\ m_L$ SUBSTITUTE ADDRESS (lower) (SAL) $(A_0 - 14) \rightarrow M_{LA}$

This instruction replaces the address portion of the lower instruction word in the storage location designated by the execution address with the lowest-order 15 bits of the A register contents. The remaining bits of the designated word in storage are not modified by this operation. This instruction effectively inserts an address in the second instruction at the designated storage location. The contents of A are not modified by this instruction.

62 b m INPUT TRANSFER (INT) Transfer (B^b) words to memory beginning at address $m + B^b - 1$

Instruction 62 transfers a block of data from an external equipment into the central computer storage. The number of words to be transferred is specified by the contents of the designated index register, B^b. These words are located in a consecutive list which begins at the location specified by the base execution address, m. The transfer begins by storing the first word in the last position in the list, namely at address $m + B^b - 1$. The content of the designated index register is reduced by one for each word that is transferred, and the transfer continues until the contents of the designated index register are reduced to zero. If $b = 0$, one word is transferred to address m.

63 b m OUTPUT TRANSFER (OUT) Transfer (B^b)words from memory beginning with address $m+B^b - 1$

This instruction transfers a block of data from computer storage to an external equipment. The number of words to be transferred is specified by the contents of the designated index register, B^b. The words to be transferred are located in a consecutive list which begins at the location specified by the execution address, m. The transfer begins by obtaining the first word from the last position in the list, namely at address $m + B^b - 1$. The content of the designated index register is reduced by one for each word that is transferred, and the transfer continues until the contents of the designated index register are reduced to zero. If $b=0$, one word is transferred from address m.

64 b m	EQUALITY SEARCH	(EQS)	Search (B^b) words, beginning with word at address m + B^b - 1 (M) = A: Exit

A list of operands is searched to find one that is equal to the content of the A register. The number of items in the list is specified by the content of the designated index register. These items are located in a consecutive list beginning at the location specified by the base execution address. The search begins with the last operand in the list, namely the one at address m + B^b - 1. The content of the designated index register is reduced by one for each operand examined. The search continues until an operand is reached that is equal to the content of the A register or until the contents of the designated index register are reduced to zero. If the search is terminated by finding an operand equal to the value in A, an exit is performed. The address of the operand which satisfied the criterion is given by the sum of the base execution address and the final contents of the index register. If no operand in the list is equal to the value in A, then a half exit is performed. In the equality comparison made here, plus zero (that is, all zeros) and minus zero (that is, all ones) are treated as equal. If b = 0 only the word at m is searched.

65 b m	THRESHOLD SEARCH	(THS)	Search (B^b) words, beginning with word at address m + B^b - 1 (M)>(A): Exit

Instruction 65 searches a list of operands to find one that is greater than the contents of the A register. The number of items in the list is specified by the contents of the designated index register. These items are located in a consecutive list beginning at the location specified by the base execution address. The search begins with the last operand in the list, namely, the one at address m + B^b - 1. The content of the designated index register is reduced by one for each operand examined. The search continues until an operand is reached that is greater than the content of the A register or until the contents of the designated index register are reduced to zero. If the search is terminated by finding an operand greater than the value in A, an exit is performed. The address of the operand which satisfied the criterion is given by the sum of the base execution address and the final contents of the index register. If no operand in the list is greater than the value in A, then a half exit is performed. In the comparison made here plus zero is considered as greater than minus zero. If b=0, only the word at m is searched.

66 b m MASKED EQUALITY (MEQ) Search (B^b) words beginning with word at address $m + B^b - 1$.
L(Q) (M) = (A): Exit

This instruction searches a list of operands to find one such that the logical product of the operand and the contents of the Q register (that is, the masked operand) is equal to the contents of the A register. The number of items in the list is specified by the content of the designated index register. These items are located in a consecutive list beginning at the location specified by the base execution address. The search begins with the last operand in the list namely, the one at address $m + B^b - 1$. The content of the designated index register is reduced by one for each operand examined. The search continues until an operand is reached that, when masked, is equal to the value in the A register, or until the contents of the designated index register are reduced to zero. If the search is terminated by finding a masked operand that is equal to the value in A, an exit is performed. The address of the operand which satisfied the criterion is given by the sum of the base execution address and the final contents of the index register. If no operand in the list satisfies the criterion then a half exit is performed. If b=0, only the word at m is searched.

67 b m MASKED THRESHOLD (MTH) Search (B^b) words, beginning with word at address $m + B^b - 1$.
L(Q) (M) >A: Exit

The instruction searches a list of operands to find one such that the logical product of the operand and the contents of the Q register (that is, the masked operand) is greater than the contents of the A register. The number of items in the list is specified by the contents of the designated index register. These items are located in a consecutive list beginning at the location specified by the base execution address. The search begins with the last operand in the list namely, the one at address $m + B^b - 1$. The content of the designated index register is reduced by one for each operand examined. The search continues until an operand is reached that, when masked, is greater than the value in the A register or until the contents of the designated index register are reduced to zero. If the search is terminated by finding a masked operand that is greater than the value in A, an exit is performed. The address of the operand which satisfied the criterion is given by the sum of the base execution address and the final contents of the index register. If no operand in the list satisfied the criterion then a half exit is performed. If b=0, only the word at m is searched.

70 b m   REPLACE ADD   (RAD) $\left[(M) + (A)\right] \rightarrow M \& A$

The quantity specified by the execution address is replaced with its original value plus the value in the A register. The resultant sum is left in the A register at the end of the operation. An overflow condition produces an interrupt (if selected) and sets an indicator which may be sensed by an external function instruction.

71 b m   REPLACE SUBTRACT   (RSB) $\left[(M) - (A)\right] \rightarrow M \& A$

Instruction 71 replaces the quantity specified by the execution address with its original value minus the value in the A register. The resultant difference is left in the A register at the end of the operation. An overflow condition produces an interrupt (if selected) and sets an indicator which may be sensed by an external function instruction.

72 b m   REPLACE ADD ONE   (RAO) $\left[(M) + 1\right] \rightarrow M \& A$

This instruction replaces the quantity specified by the execution address with its original value plus one. The resultant quantity is left in the A register at the end of the operation and the original contents of the A register are destroyed by this operation. An overflow condition produces an interrupt (if selected) and sets an indicator which may be sensed by an external function instruction.

73 b m REPLACE SUBTRACT ONE (RSO) [(M) - 1] → M & A

The quantity specified by the execution address is replaced with its original value minus one. The resultant quantity is left in the A register at the end of the operation, the original contents of the A register are destroyed by this operation. An overflow condition produces an interrupt (if selected) and sets an indicator which may be sensed by an external function instruction.

74 j y EXTERNAL FUNCTION (EXF) j = 1-6: activate channel j
j = 0: select condition y
j = 7: on condition y
exit or half exit

This instruction has eight subinstructions which are used to control the transfer of information between the computer and external equipments. The index registers are not used for address modification in this instruction. The designator is used to specify one of eight operations to be performed.

The subinstructions and the operation performed for each are as follows:

74 0 y - Select external equipment or internal condition y

74 1 y - Activate communication channel one

74 2 y - Activate communication channel two

74 3 y - Activate communication channel three

74 4 y - Activate communication channel four

74 5 y - Activate communication channel five

74 6 y - Activate communication channel six

74 7 y - Sense external or internal condition, y

Subinstructions 74 1 y through 74 6 y are used to begin buffering a block of data between the computer and a previously selected external equipment. The base execution address is used to designate the starting address in the computer central storage. This address is automatically recorded in the upper address position of the appropriate special storage location (00001 - 00006). The terminal address (plus one) of the block of data must have been previously recorded by the program in the lower address position of the appropriate special storage location prior to the execution of this external function instruction.

Subinstructions 74 0 y and 74 7 y provide for selecting and sensing conditions within external equipment or the computer. The execution address y is a code that specifies the equipment or part of the computer and the condition. The 24 bits of the instruction are interpreted as follows:

6 bits	3 bits	3 bits	3 bits	9 bits
Operation Code 74	0 - Select 7 - Sense	0 - Internal 1 thru 6 Buffer 7 - Transfer	Equipment 0 thru 7	Conditions or Mode 000 thru 777

A 74 0 y yields the same result when used in the upper or lower position of an instruction word. However, a 74 7 y causes a skip or wait depending upon its position.

When used in the upper position, a 74 7 y is a skip instruction. That is, the lower instruction is skipped if the condition given by the EF code is present, but the lower instruction is executed if the condition given by the EF code is not present. In the first case, the 74 7 y exits to the next pair of instructions. In the second case the 74 7 y half exits to the lower instruction.

When the 74 7 y is used in the lower position, it is not a skip instruction. Instead the sense is executed repeatedly until the condition given by the EF code occurs. At this time an exit is performed to the next pair of instructions. Until the condition given by the EF code is present, the instruction simply half exits to repeat itself. A 74 7 y in the lower position is therefore a means of awaiting the occurrence of a specified condition.

75 j m SELECTIVE JUMP (SLJ) Jump to m

This instruction has eight subinstructions which cause a jump in program sequence on specified conditions of operator lever keys. The index registers are not used for address modification in this instruction. The jump designator in the instruction specifies which lever key is sampled in determining the jump decision.

The subinstruction and the operation performed by each when a positive result is obtained from sampling the appropriate lever key are as follows:

75 0 m - Jump unconditionally

75 1 m - Jump if lever key one is set

75 2 m - Jump if lever key two is set

75 3 m - Jump if lever key three is set

75 4 m - Return jump unconditionally

75 5 m - Return jump if lever key one is set

75 6 m - Return jump if lever key two is set

75 7 m - Return jump if lever key three is set

76 j m SELECTIVE STOP (SLS) Stop, jump to m

Instruction 76 consists of eight subinstructions which cause the program to stop on specified conditions of operator lever keys. The index registers are not used for address modification in this instruction. The jump designator in the instruction specifies which lever key is sampled in determining the stop decision. A jump to the base execution address occurs regardless of the stop decision.

The subinstructions and the operation performed by each when a positive result is obtained from sampling the appropriate lever key are as follows:

76 0 m - Stop unconditionally (normal jump)
76 1 m - Stop if lever key one is set (normal jump)
76 2 m - Stop if lever key two is set (normal jump)
76 3 m - Stop if lever key three is set (normal jump)
76 4 m - Stop unconditionally (return jump)
76 5 m - Stop if lever key one is set (return jump)
76 6 m - Stop if lever key two is set (return jump)
76 7 m - Stop if lever key three is set (return jump)

SPECIAL FEATURES

FLOATING POINT

Words packed in floating-point format take advantage of the ability to express a quantity as the product of a fraction and a base number with an integer exponent. The 1604 format includes only the fractional coefficient and the exponent with their respective signs. The assignment of the factors within the 48-bit word is shown below.

coefficient sign	sign + exponent	coefficient (magnitude)
1 bit	1 bit + 10 bits	36 bits

The 10-bit exponent enables the encoding of quantities within the exponent range $-(2^{10}-1)$ to $+(2^{10}-1)$, or - 1023 to + 1023. For purposes of precision, the coefficient is always written as a fraction (f_n) of the order $1/2 \leq f_n < 1$. In order to make floating-point operation more versatile, operands are encoded in such a manner that they may be compared with each other in the fixed-point mode.

INDIRECT ADDRESSING

The indirect addressing feature provides additional flexibility for programs involving a great deal of address modification. In indirect addressing, the execution address portion of an instruction is used as the address of a storage location that holds the operand address, whereas for direct addressing the operand location is obtained immediately from the execution address (modified by the contents of an index register when desired). For obtaining the operand address, indirect addressing requires an additional storage reference.

All instructions except 22, 23, 74, 75 and 76 may be used with either direct or indirect addressing. Indirect addressing occurs when b=7; otherwise direct addressing is used. Note that this applies to instructions which do not normally use the execution address as an operand address but rather as the operand itself, for example the shift instructions.

INTERRUPT

A program may, by making the appropriate selection, be interrupted when a certain condition arises in an external equipment or in internal computer control. The interrupt feature thus offers a method for the program to have information about such conditions without requiring it to monitor them. The interruption, which takes place at the completion of the instruction currently in process, involves a return jump to storage address 00007. The return jump initiates the interrupt routine which interrogates external equipments or examines internal conditions to determine the cause of the interrupt. The routine takes appropriate action and returns to the main program.

Examples of conditions in external equipments which may be selected to cause interrupts are "ready to read", "end of tape", and various errors. Internal control conditions that may produce interrupts are arithmetic faults, and the termination of a buffer.

REAL-TIME CLOCK

When address 00000 in the central storage system is selected it provides a continually operating time record. The 48-bit quantity is advanced one count every 1/60 of a second. Accuracy is maintained by the 60-cycle power source. The clock is started or stopped by the execution of an external function instruction.

By selecting interrupt on arithmetic overflow, and presetting the contents of address 00000, the real-time clock may be used to provide periodic interrupt of the main computer program. The periodicity is variable in increments of 1/60 of a second. Overflow sensed by seeing that location 00000 is negative.

INPUT-OUTPUT COMMUNICATIONS

The computer communicates with external equipment either through buffer channels or the transfer channel. The buffer channels provide for the normal exchange of data while the transfer channel provides for very high-speed communication.

Data exchange on each of the six separate buffer channels (3 input and 3 output) proceeds under a control that is independent of the main program. The program initiates the buffer and specifies the block of storage locations involved. After initiation, the buffer exchange runs to termination independently. The program may select to be notified of termination by an interrupt.

A scanner continuously and successively examines each buffer channel to determine if the equipment is ready to send data to or receive it from computer storage. When equipment is ready, scanning halts while the exchange takes place, following which scanning continues.

Instructions 62 Input Transfer or 63 Output Transfer provide for transferring one wordor a block of information via the transfer channel.

INPUT-OUTPUT EQUIPMENT

Input-output equipment for the 1604 consists of the units at the console, which are standard with every computer, and several optional units. The optional equipment includes the 1607 Magnetic Tape System and the 1605 Adaptor.

CONSOLE EQUIPMENT

Reader

A transistorized Ferranti 7-level photo-electric paper tape reader is used for paper tape input. This reader operates at a maximum of 350 lines per second on a line-by-line basis.

Typewriter

A modified electric typewriter monitors information during computer operation. The computer can present typed copy for the operator by buffering data to the typewriter; or the operator can use the typewriter keyboard to make computer entries during computation by buffering data into the computer.

Punch

A Teletype BRPE paper tape punch is used to prepare 7-level paper tape. This punch operates at 60 characters per second.

1607 MAGNETIC TAPE SYSTEM

A tape system consists of four Ampex handlers housed in a single cabinet. Included are data handling and control circuitry, 48-bit assembly and disassembly registers; parity checking and error detection.

Each tape system can be connected to any of the three pairs of buffer channels, and each is independently addressable. The system has the facility for simultaneously reading from one tape handler and writing on another while the remaining two are rewinding.

Tapes of the 1607 are completely compatible electrically and mechanically with those of IBM Model 727 magnetic tape handlers.

1605 ADAPTOR

The adaptor permits communication between the 1604 computer and any of the following IBM external equipments:

714 Card Reader (via 759 Control Unit)

727 Magnetic Tape Units (via 754 Synchronizer)

717 Line Printer (via 757 Control Unit)

722 Card Punch (via 758 Control Unit)

The adaptor selects the equipment, as well as the operation to be performed, on the basis of an instruction from the main computer program. A parity check is made on all information transmitted through the adaptor.

Each adaptor can be connected to any of the three pairs of buffer channels, and is independently addressable. A number of adaptors together with a number of 1607 tape systems can be operated with a single computer.

EXTERNAL FUNCTION CODES

The select codes (74.0) and sense codes (74.7) for the computer and several external equipments are listed below.

INTERNAL

SELECT		SENSE	
00010	Interrupt on channel 1 inactive	00010	Exit on channel 1 active
00011	Remove selection above	00011	Exit on channel 1 inactive
00020	Interrupt on channel 2 inactive	00020	Exit on channel 2 active
00021	Remove selection above	00021	Exit on channel 2 inactive
00030	Interrupt on channel 3 inactive	00030	Exit on channel 3 active
00031	Remove selection above	00031	Exit on channel 3 inactive
00040	Interrupt on channel 4 inactive	00040	Exit on channel 4 active
00041	Remove selection above	00041	Exit on channel 4 inactive
00050	Interrupt in channel 5 inactive	00050	Exit on channel 5 active
00051	Remove selection above	00051	Exit on channel 5 inactive
00060	Interrupt on channel 6 inactive	00060	Exit on channel 6 active
00061	Remove selection above	00061	Exit on channel 6 inactive
00070	Clear arithmetic faults	00110	Exit on divide fault
00100	Interrupt on arithmetic faults	00111	Exit on no divide fault
00101	Remove selection above	00120	Exit on shift fault
01000	Start real-time clock	00121	Exit on no shift fault
02000	Stop real-time clock	00130	Exit on overflow fault
		00131	Exit on no overflow fault
		00140	Exit on exponent fault
		00141	Exit on no exponent fault

CONSOLE EQUIPMENT

11100	Keyboard entry & no interrupt on carriage return	11100	Exit on keybaord carriage return
11140	Keyboard entry & interrupt on carriage return	11101	Exit on no keyboard carriage return
11200	Reader & no interrupt on end-of-tape	11140	Exit on keyboard lower case
11210	Reader & end-of-tape indicator	11141	Exit on keyboard upper case
11220	Reader & interrupt on end-of-tape	11200	Exit on reader, end-of-tape
21100	Print assembly mode	11201	Exit on reader, no end-of-tape
21110	Print character mode	11210	Exit on reader, assembly mode
21200	Punch assembly mode	11211	Exit on reader, character mode
21210	Punch character mode	21200	Exit on punch out-of-tape
21240	Turn punch motor off	21201	Exit on punch no out-of-tape

1605 ADAPTOR

SELECT

54000	Begin cycle or start read binary
54001	Start read coded or begin cycle
54002	Select read binary for read while write
54003	Select read coded for read while write
54005	Rewind
54006	Backspace
54007	Interrupt on end of operation
54100	Turn on indicator
54101	Turn off indicator
54200	Clear interrupt selection
544n0	Start read binary, unit n
544n1	Start read coded, unit n
544n2	Select read binary, unit n
544n3	Select read coded, unit n
54500	Begin cycle card reader, binary
54501	Begin cycle card reader, coded
64000	Begin cycle or write binary
64001	Write coded
64004	Write end-of-file
64005	Rewind
64006	Backspace
64007	Interrupt on end of operation
64100	Turn on indicator
64101	Turn off indicator
64200	Clear interrupt selection
644n0	Write binary, unit n
644n1	Write coded, unit n
64600	Begin cycle card punch, binary
64601	Begin cycle card punch, coded
64700	Begin cycle line printer, binary
64701	Begin cycle line printer, coded

SENSE

54000	Exit on end of operation
54001	Exit on no end of operation
54002	Exit on parity error
54003	Exit on no parity error
54004	Exit on indicator
54005	Exit on no indicator
54006	Exit on buffer length error
54007	Exit on no buffer length error
64000	Exit on end of operation
64001	Exit on no end of operation
64002	Exit on parity error
64003	Exit on no parity error
64004	Exit on indicator
64005	Exit on no indicator

1607 CODES

SELECT

320n1	Select read tape n in binary
320n2	Select read tape n in coded
32001	Read selected read tape, binary
32002	Read selected read tape, coded
32004	Interrupt when selected read tape ready
32005	Rewind selected read tape
32006	Backspace selected read tape
32007	Rewind selected read tape with interlock
420n1	Select write tape n in binary
420n2	Select write tape n in coded
42001	Write selected write tape, binary
42002	Write selected write tape, coded
42003	Write end-of-file mark on selected write tape
42004	Interrupt when selected write tape ready
42005	Rewind selected write tape
42006	Backspace selected write tape
42007	Rewind selected write tape with interlock

SENSE

32000	Exit on ready to read
32001	Exit on not ready to read
32002	Exit on read parity error
32003	Exit on no read parity error
32004	Exit on read length error
32005	Exit on no read length error
32006	Exit on end-of-file mark
32007	Exit on no end-of-file mark
42000	Exit on ready to write
42001	Exit on not ready to write
42002	Exit on write reply parity error
42003	Exit on no write reply parity error
42004	Exit on write reply length error
42005	Exit on no write reply length error
42006	Exit on end of tape marker
42007	Exit on no end of tape marker

SATELLITE 1607 CODES

32501	Read control to 160
32502	Read control to 1604
42501	Write control to 160
42502	Write control to 1604
42503	Direct 160 to 1604
32503	Direct 1604 to 160
42500	Release direct selections
32505	Release 160 Interrupt

32500	Read control available
32501	Read control not available
42500	Write control available
42501	Write control not available
32504	160 interrupt
32505	No 160 interrupt

Notes: Codes for alternate 1607, 33xxx or 43xxx.

CHAPTER TWO

OPERATION

The main computer and external equipment are placed in operation by procedures which include loading and unloading data and programs, making necessary manual selections, and starting a program. Steps are listed in the recommended order of performance, beginning with the computer and external equipment in a shut down condition.

The frontispiece shows the operator's console. The center panel of the console (figure 2-1) contains the controls and indicators for the main computer. Controls for the external equipments are shown in figure 2-2 through 2-4.

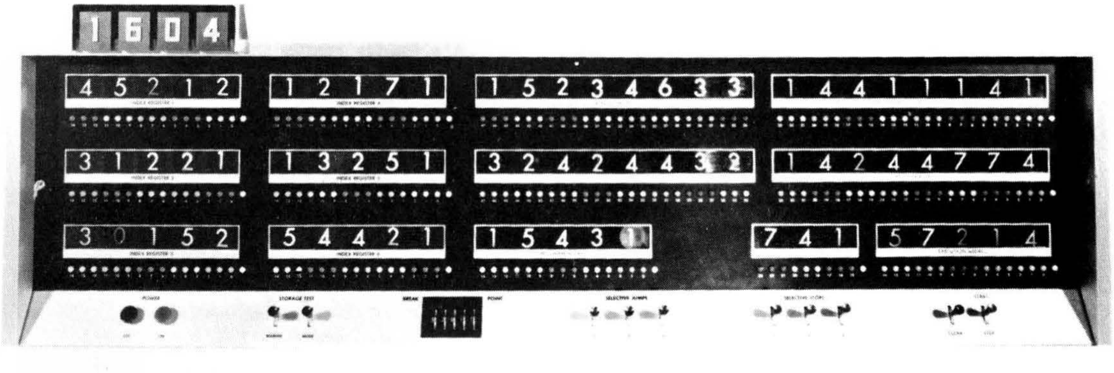


Figure 2-1. Center Panel of Console.

STARTING OPERATION WITH PRE-STORED LOAD PROGRAM

When a general loading program which provides for loading other programs is held in storage, the starting procedure is as follows:

1) Turn on power (depress POWER ON button, figure 2-1).

2) If punch is to be used, raise PUNCH switch to SELECT position (figure 2-2) and check for sufficient paper in reel. (If supply of tape is low see paragraph on additional procedures.)

3) If typewriter is to be used, place paper in it.

4) Make required manual selections (figure 2-1):

   Selective Jumps

   Selective Stops

   Breakpoint

5) Prepare paper tape baskets and empty chad box.

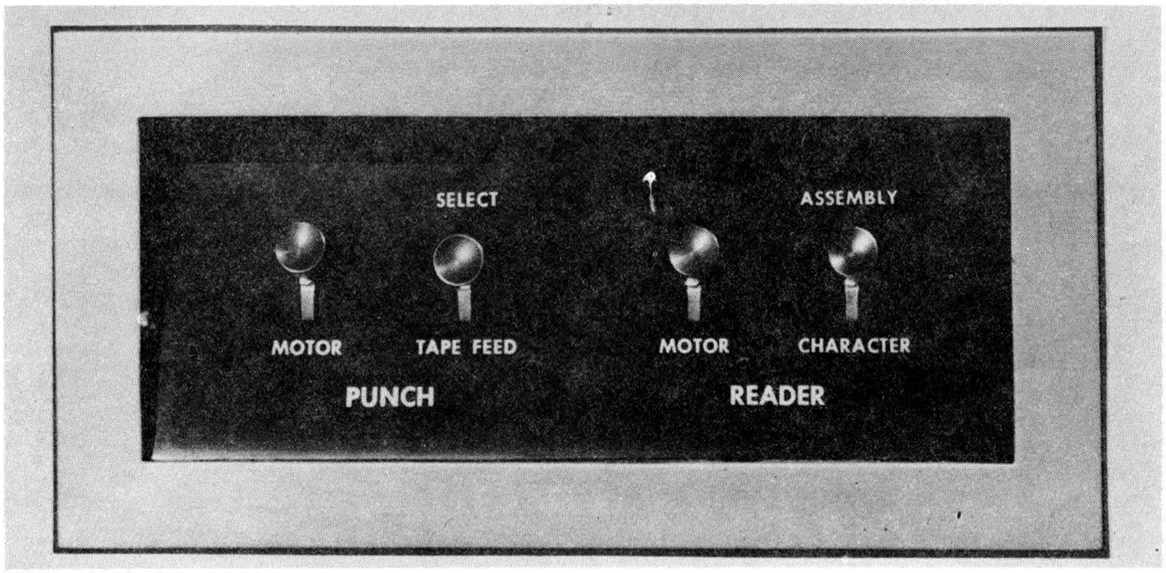


Figure 2-2. Reader and Punch Controls.

6) If paper tape is to be read, load into reader. (figure 2-3).

a. Turn tape release lever counter clockwise to raise tape guide plate.

b. Insert tape as shown in figure 2-3.

c. Turn on reader motor (READER MOTOR switch up).

d. If tape is bi-octal with 7th level control holes, select assembly mode (MODE switch raised to ASSEMBLY).

e. If tape is in flex or other code, select character mode (MODE switch depressed to CHARACTER).

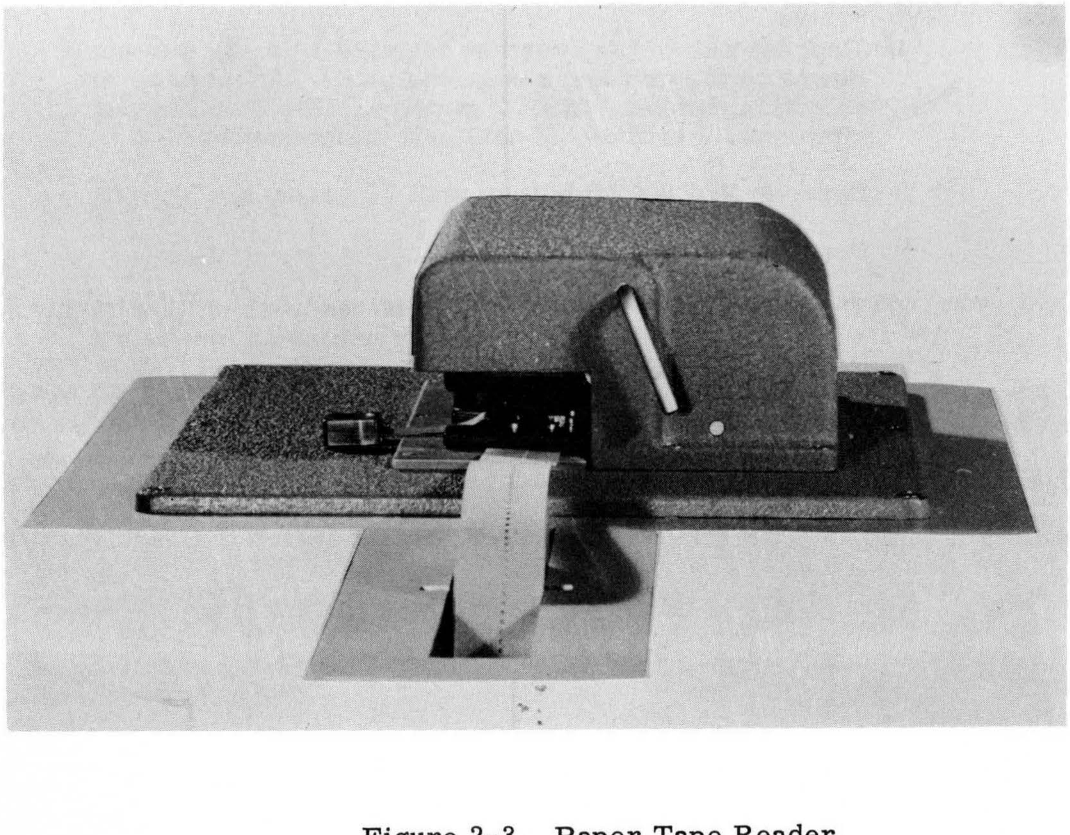


Figure 2-3. Paper Tape Reader.

7) If magnetic tape is to be used: (figure 2-4)

a) Open door to handler.

b) Check that file reel to be loaded has been file protected as necessary.

c) Mount the reel on the file reel hub and tighten the hub knob. To insure proper reel alignment push the reel firmly against the reel hub stop before tightening the knob. If the file protection ring has been removed from the reel check that the Write Lockout lamp turns on when the reel is loaded. If the lamp does not turn on call maintenance.

d) Press upper REEL BRAKE pushbutton (to release mechanical brake) and check that pulling tape from reel causes it to rotate clockwise. Pull sufficient tape from reel to reach end of permanent machine leader held by leader clamp.

e) Connect file tab to permanent machine leader.

f) Take up slack by turning file reel while depressing upper REEL BRAKE pushbutton.

g) Lift leader clamp and close door.

h) Depress one of the switches labelled 1, 2, 3, 4 to apply power to the unit and assign the unit a logical program selection number. Wait 2 minutes. The Stop Manual lamp should turn on, if not, call maintenance.

i) Depress FORWARD button; wait 10 seconds.

j) Depress STOP MANUAL.

k) Depress REWIND button. Unit is ready when Rewind lamp turns off. If Stop Manual lamp remains on unit is not ready; call maintenance.

8) Master Clear both internal and external (depress CLEAR then raise it).

9) Set Program Address register to address of first instruction of program.

10) Begin computer operation (raise START-STEP switch to START position).

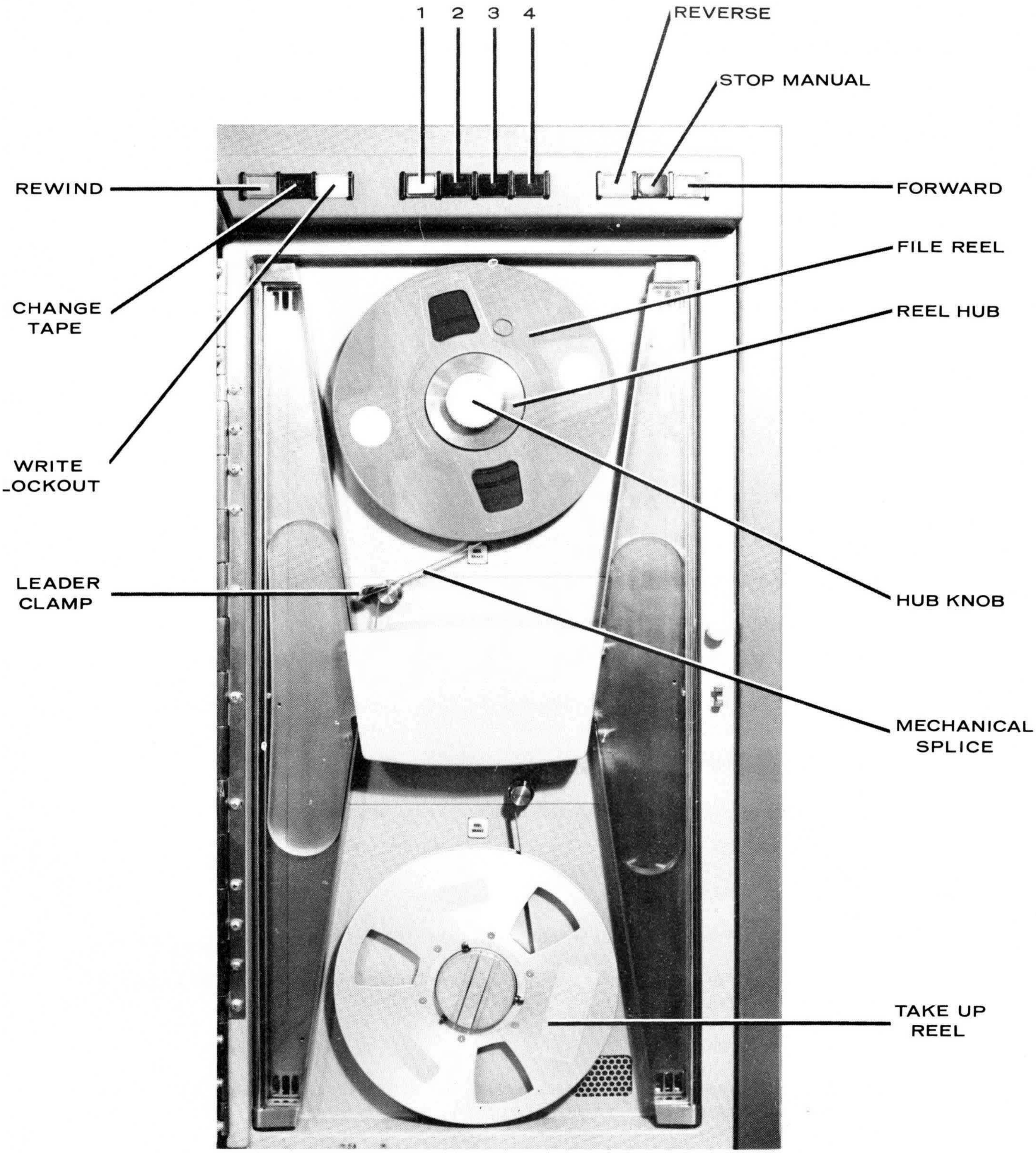


Figure 2-4. Tape Unit.

STARTING OPERATION WITHOUT PRE-STORED LOAD PROGRAM

When a load program must be entered in storage it is usually contained on bi-octal paper tape. The following procedure provides for entering the load program:

1) Turn on power.

2) Master clear both internal and external.

3) Depress Start-Step switch once.

4) Clear function code and set to 200.

5) Clear execution address and set to 00001.

6) Set terminal address of buffer in lowest five octal digits of A register right.

7) Depress Start-Step switch once.

8) Load tape into reader.

9) Turn on reader motor (wait 10 seconds).

10) Raise reader mode switch to ASSEMBLY position.

11) Clear function code and set to 741.

12) Clear execution address and set to initial address of buffer.

13) Depress Start-Step switch once. Wait until tape loads (console lights come on).

14) Depress CLEAR switch.

15) Perform steps 2 through 10 of the preceding procedure.

SHUTTING DOWN EQUIPMENT

After operation has stopped, shut down the equipment in the following manner:

1) If the reader was used, remove paper tape from reader and baskets; rewind tapes.

2) Turn off reader motor.

3) If punch was used, generate a foot of leader by depressing punch mode switch (Tape Feed); remove tape and wind it up.

4) To unload magnetic tape, proceed as follows:

a) Depress STOP MANUAL button to select manual mode.

b) Depress REVERSE button to move tape backwards to change tape position.

c) Open front door of tape unit.

d) To secure tape, lower leader clamp.

e) Depress the upper REEL BRAKE button (to release the mechanical brake) and pull tape from file reel to provide slack.

f) Unfasten mechanical splice which connects the file tab to the permanent machine leader.

g) Loosen file reel hub knob and remove the file reel.

h) Check if reel needs to be file protected and also if it is labelled adequately prior to storage.

5) Depress POWER OFF button, which disconnects power from all equipments.

DESCRIPTION OF INDICATORS AND CONTROL SWITCHES

MAIN COMPUTER

Controls and indicators for operating the main computer are located at the center of the console (figure 2-5).

The push buttons associated with each operational register are numbered in the powers of two, from right to left, starting with zero. Depressing a push button results in that particular stage of the register being forced to the SET state. Each group of three buttons may be thought of as an octal digit, and as an aid in distinguishing octal groups from one another, different shades of blue are used in adjacent octal groups. Within an octal group the three buttons are of the same color.

Associated with each operational register is a CLEAR push button, colored white. This button will clear the individual FFs within that register. Use both SET and CLEAR push buttons only when the computer is stopped. Depressing these buttons during operation may result in errors.

The indicators consist of lamp modules, each of which displays a single octal digit. The lamps, in response to signals from the computer, display the contents of the operational registers in octal form. Each octal indicator is associated with the three push buttons beneath it. The contents of registers are displayed only when the computer is stopped. Thus the display is blank when the computer is running.

Conditions which result in the stopping of the computer are listed below. When these conditions exist the contents of registers may be altered by setting or clearing.

1) Illegal function codes 00 and 77
2) Selective Stops (instruction 76)
3) Breakpoint Stop
4) Depressing START-STEP switch
5) Depressing CLEAR switch (internal master clear)

At some of the modules there are colored background lights which provide an indication of certain conditions within the computer. In general red lights signify faults and blue lights signify special operating conditions. The background lights may be illuminated when the computer is running as well as when it is stopped. The module positions with background lights are shown in figure 2-5. Table 2-1 gives the functional significance of each light. A light is identified by the register in which it is located and its position in the register. For example, AL-4 is fourth from the left in A register left.

The manual controls for the main computer are located beneath the indicator panel (figure 2-5). Their functions are described in Table 2-2.

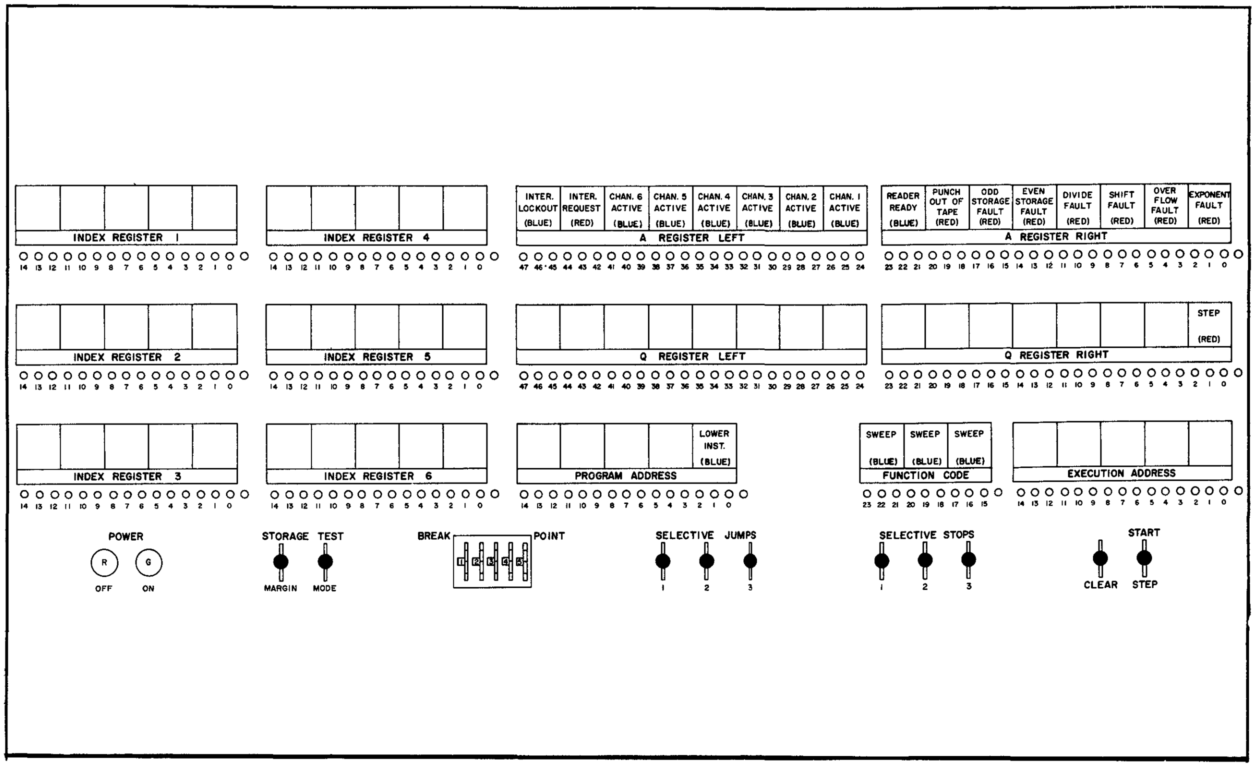


Figure 2-5. Console Display.

TABLE 2-1. CONDITIONS INDICATED BY CONSOLE BACKGROUND LIGHTS

Light	Condition
AL-1 (blue)	Interrupt Lockout - Computer is in interrupt routine.
AL-2 (red)	Interrupt Request - Interrupt request signal is being received from interrupt circuit.
AL-3 (blue)	Channel 6 Active - Channel 6 is in use for output buffer.
AL-4 (blue)	Channel 5 Active - Channel 5 is in use for input buffer.
AL-5 (blue)	Channel 4 Active - Channel 4 is in use for output buffer.
AL-6 (blue)	Channel 3 Active - Channel 3 is in use for input buffer.
AL-7 (blue)	Channel 2 Active - Channel 2 is in use for output buffer.
AL-8 (blue)	Channel 1 Active - Channel 1 is in use for input buffer.
AR-1 (blue)	Reader Ready - (1) Paper tape is positioned at load point, ready for an input buffer; or (2) an input buffer involving the paper tape reader is in progress.
AR-2 (red)	Punch Out of Tape - Punch tape reel is nearly empty.
AR-3 (red)	Odd Storage Fault - Fault in sequence chain of odd storage unit; storage unit is inoperative until master clear is performed.
AR-4 (red)	Even Storage Fault - Fault in sequence chain of even storage unit; storage unit is inoperative until master clear is performed.
AR-5 (red)	Divide Fault - Improper divide instruction executed.
AR-6 (red)	Shift Fault - Shift instruction executed with shift count greater than 127 (decimal).
AR-7 (red)	Overflow Fault - Required sum or difference exceeds capacity of A register.
AR-8 (red)	Exponent Fault - In a floating-point instruction, exponent of result is 2^{10} or greater.
QR-8 (blue)	Deep End - Computer fails to complete operation in step mode.
PA-5 (blue)	Lower Instruction - Lower instruction is being indicated.
FUNCTION CODE (blue) (3 lights)	Sweep - Computer is in storage sweep mode (MODE switch is down).

TABLE 2-2. MAIN COMPUTER CONTROLS

Control		Function
POWER push button	ON - green	Applies d-c power to computer by energizing contactor in primary power lines of motor-generator.
	OFF - red	Removes d-c and a-c power from computer by de-energizing contactor in primary power lines of motor-generator.
STORAGE TEST Lever switch locks in up, down and neutral positions.	MARGIN	Varies the bias applied to storage sense amplifiers. Used for maintenance purposes ONLY; should be in neutral position at all other times.
	MODE	Up position: an instruction is executed repeatedly in in either the STEP or START mode. Down position: contents of consecutive storage locations may be manually examined by depressing STEP. Consecutive half-words are displayed in function code and execution address registers but are not executed.
BREAKPOINT Five 8-position switches can be set to octal address 00000 through 77777.		Provides for selection of any storage address as a breakpoint address. Computer stops when program address and breakpoint address are equal, just prior to performing the upper instruction at the breakpoint address.
SELECTIVE JUMPS 1, 2, 3 Three lever switches lock in upper positions, momentary in down positions.		Provide manual conditions for: normal jumps, instruction 75 (b = 1, 2 or 3) return jumps, instruction 75 (b = 5, 6 or 7) (Performs same function in all positions.)
SELECTIVE STOPS, 1, 2, 3 Three lever switches lock in upper position, momentary in down positions.		Provide manual conditions for stopping the computer on instructions 76 (b = 1, 2, 3, 5, 6 or 7) (Performs same function in all positions.)
CLEAR Lever switch, momentary in up and down positions.		Down position master clears the computer, causing all operational registers and most control FFs to be cleared. Up position master clears external equipment, causing most of the registers and control FFs of the external equipment to be cleared and the paper tape reader to be selected.

TABLE 2-2 (CONT'D.)

Control	Function
START-STEP Lever switch, momentary in up and down positions.	START (up) selects high-speed mode in which a program of instructions and auxiliary operations proceeds until completed or stopped. STEP (down) selects step mode. Each time switch is depressed a single instruction is executed and computer stops (all buffer requests are completed before operation stops). STEP selection overrides any previous selection of START.
VOLUME CONTROL Black knob under console desk	Controls volume of signal from console loudspeaker.

READER AND PUNCH CONTROLS

The controls for operating the paper tape reader and punch (located on right wing of console) appear in figure 2-2. The controls are described in table 2-3.

TABLE 2-3. READER AND PUNCH CONTROLS

Switch	Function
PUNCH MOTOR	Turns on punch motor. (Motor may also be turned on under program control.)
SELECT-TAPE FEED	In SELECT position enables use of the punch. In TAPE-FEED position causes leader to be punched.
READER MOTOR	Turns reader motor on or off. (Motor cannot be turned on by any other means.)
CHARACTER-ASSEMBLY	In CHARACTER position the reader operates in character mode (each character is sent to computer separately). In ASSEMBLY position the reader operates in assembly mode (eight consecutive characters are assembled into a word that is sent to computer).

1607 CONTROLS AND INDICATORS

Each tape unit is provided with push button indicator switches for manual operation. These controls are mounted on a panel located above the front door of each tape unit (figure 2-4). The functions of the controls are shown in table 2-4.

TABLE 2-4. FUNCTION OF 1607 CONTROLS AND INDICATORS

Control		Function
Rewind	S*	Controls manual rewind to load point.
	I**	Indicates rewind in progress.
Change Tape	S	Drops any manual selection and places tape unit in automatic or program control mode.
	I	When lighted indicates tape rewound under program control and interlocked at load point. The interlock prevents operation of the tape unit and prevails until the Stop Manual switch is operated.
Write Lockout	S	Drops power from unit and removes program designation.
	I	When lighted indicates that tape unit is loaded with a reel which does not contain a file protection ring. The tape cannot be written as long as the light is on, but may be read.
1, 2, 3 or 4	S	Designates program selection of unit and applies power to unit. Each new unit designation cancels an existing designation.
	I	Indicates unit selection and power-on condition.
Reverse	S	Initiates reverse tape motion during manual operation.
	I	Indicates reverse tape motion.
Stop Manual	S	Drops unit from program control or drops FORWARD or REVERSE selection and places unit in manual mode.
	I	Indicates manual mode.
Forward	S	Indicates forward tape motion during manual mode.
	I	Indicates forward tape motion.

* switch

** indicator

ADDITIONAL PROCEDURES

REPLACING TAPE ROLL AT PUNCH

The paper tape punch (figure 2-6) is mounted on a hinged rack at the rear of the right wing of the console. Punch tape feeds out of a slot in the compartment door and the chad box is located just inside the door. The procedure for replacing the roll of tape at the punch is:

1) Remove the tape reel from cradle at side of punch.
2) Unscrew tape hold-down assembly, remove old roll, and place new roll on reel. Replace hold-down assembly and mount reel in cradle.
3) Thread tape as shown in figure 2-6. Bring tape around lower roller and into guides leading to punch block.
4) Turn on punch motor and advance tape through the punch block by pressing the tape feed-out lever (located at top of punch block).
5) Bring leader out through slot in door. Swing punch back into its compartment.

TYPEWRITER

The typewriter (mounted on the left wing of the console) has all of the characters and functions of a standard electric machine. If the typewriter is to be used, the switch beneath the front righthand corner must be set to ON.

As a keyboard entry device the typewriter is used only in the character mode. After the program selects keyboard and initiates an input buffer, each striking of a key causes a six-bit coded character to be entered into the rightmost six positions of a computer word. The remaining bits of the word are all "0". If the keyboard is selected along with an interrupt feature, each striking of the carriage return key sends an interrupt signal to the computer. This notifies the program of the entry of data from the keyboard.

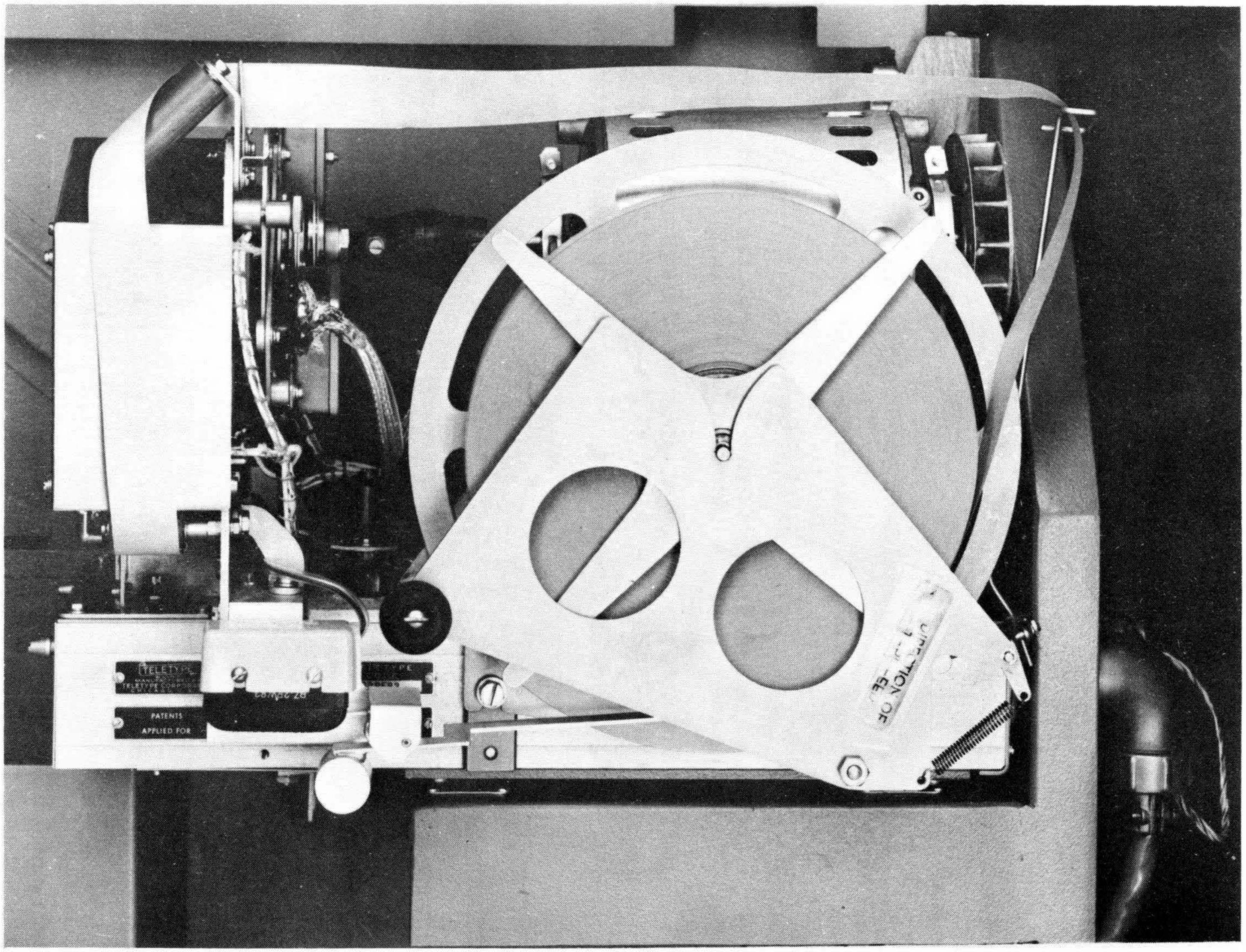


Figure 2-6. Paper Tape Punch.

When the typewriter is used as an output device certain conditions cause it to hang up until the space bar is struck: receipt of an illegal typewriter code, a code to shift up when the carriage is already up, or a code to shift down when the carriage is already down.

FILE PROTECTION RING

The back of the 1607 file reel has a slot near the hub which accepts a plastic ring called a file protection ring (figure 2-7). Writing on a tape is possible only when the reel contains a file protection ring. When the ring is in place the Write Lockout indicator goes out immediately after the reel is loaded onto the tape unit. The ring should be removed from the reel after writing is completed to avoid accidental rewriting. Tape may be read either with the ring in place or without it.

EMERGENCY PROCEDURES

A fault indication, or a warning signal from the buzzer, may call for special procedures on the part of the operator. These procedures are listed below:

TABLE 2-5. EMERGENCY PROCEDURES

Punch out of tape	Load new roll of tape in punch at end of current operation.
Odd Storage Fault	Master clear. Restart program.
Even Storage Fault	Master clear. Restart program.
Deep End	Restart operation. If unable to proceed, master clear and restart program. If condition persists, notify maintenance.
Sweep	Place MODE switch in neutral position.
Buzzer Signal	Notify maintenance engineer immediately.

Faults for which the program provides corrective action are: Divide, Shift, Overflow and Exponent Faults.

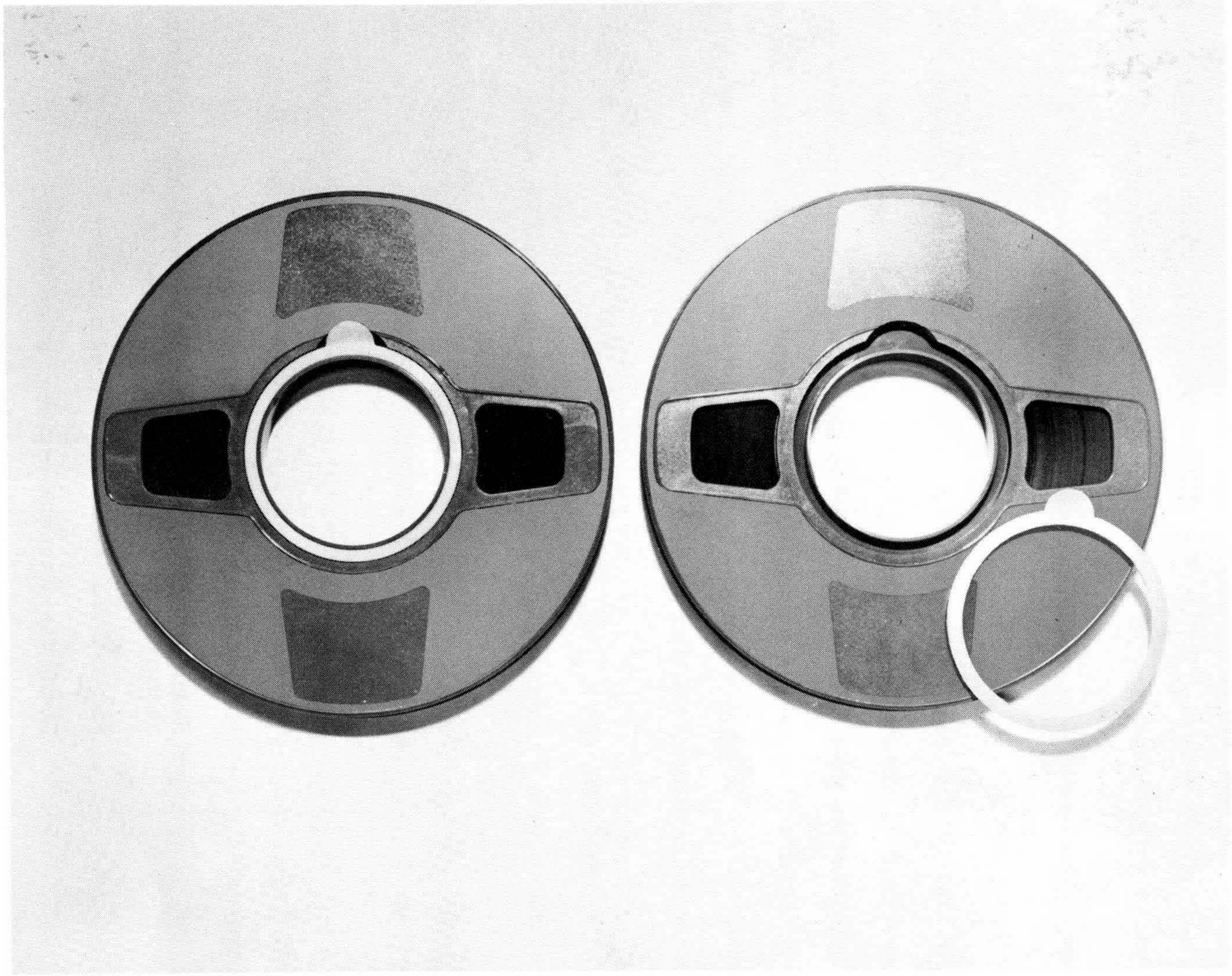


Figure 2-7. File Protection Ring.

LIST OF FIGURES

LIST OF TABLES

CONTROL DATA CORPORATION
501 Park Avenue, Minneapolis 15, Minnesota