IBM

Reference Manual

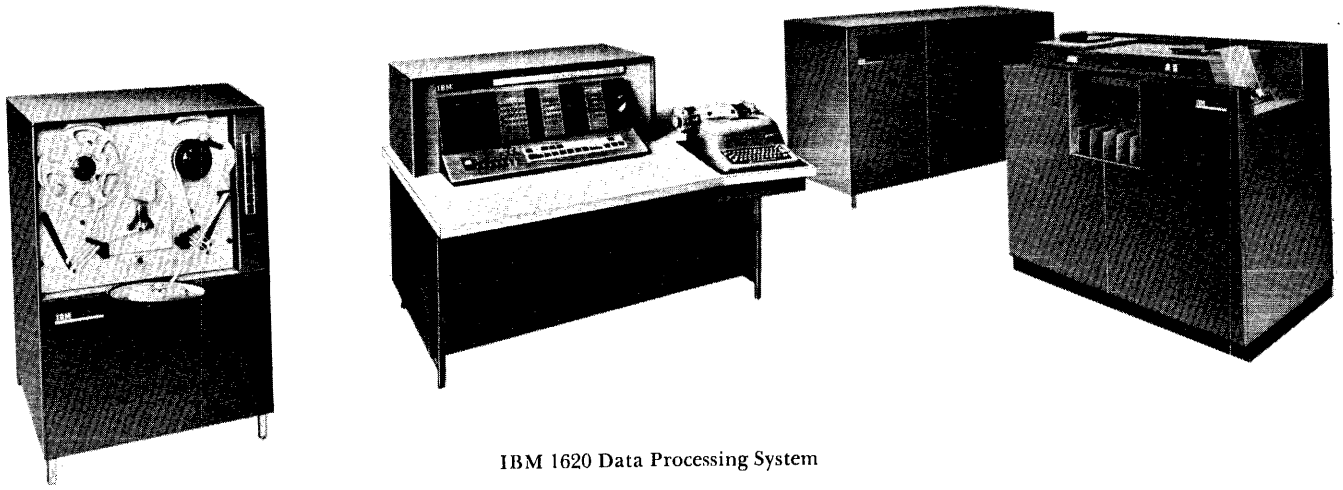
IBM 1620 Data Processing System

MAJOR REVISION (July 1961)

This edition, A26-4500-2, renders obsolete A26-4500-1. Because significant changes have been made throughout the manual, this new edition should be reviewed in its entirety. Information has been added on the following new subjects:

Address comments regarding this publication to:
IBM, Product Publications Department, San Jose, California

Contents



IBM 1620 Data Processing System

The IBM 1620 Data Processing System is an electronic computer system designed for scientific and technological applications. The use of solid-state circuit components and the availability of from 20,000 to 60,000 positions of core storage provide the 1620 system with the capacity, reliability, and speed to solve problems that in the past have required the use of larger and more expensive data processing systems.

Four units (see Frontispiece) are available with the IBM 1620 Data Processing System.

The IBM 1620 Central Processing Unit contains the computer, 20,000 positions of core storage, a console panel, and an input/output (I/O) typewriter. Paper tape I/O operations are permitted by the IBM 1621 Paper Tape Reader unit, which also includes the paper tape controls and the IBM 1624 Tape Punch.

The IBM 1622 Card Read-Punch is available for card I/O operations. The IBM 1623 Storage Unit expands the 20,000 core storage positions in the Central Processing Unit of the 1620 to 40,000 or 60,000 positions. The 1623 Model 1 contains 20,000 additional positions and the 1623 Model 2 contains 40,000 additional positions. Except where otherwise specifically noted, this manual is concerned with the standard 20,000-position 1620.

Data and instructions entered into the system are placed in core storage as decimal digits. Each position of core storage can be addressed individually and can store one digit of information by the use of a six-bit binary-coded decimal (BCD) code. The addressing system provides for the selection of any digit, or group of digits, in core storage. As a standard feature, the 1620 computer processes alphabetic and special characters.

The arithmetic and logic section of the computer is directed by the stored program. The computer performs 39 different operations, using a 2-address instruction format. The format is explained under INSTRUCTION CHARACTERISTICS. Each 12-digit instruction includes a 2-digit operation code and two 5-digit addresses. Use of the 2-address format and automatic sequential execution of the programmed instructions simplify programming and reduce the number of instructions required to solve a problem. The sequence of operations may be altered at any point in the program by unconditional or conditional branch instructions. Conditional branch instructions provide logical decisions through tests performed on a system of indicators and switches set by the computer or by the operator.

Addition, subtraction, and multiplication operations are accomplished by a table lookup method, in which Add and Multiply tables located in specified areas of core storage are referred to automatically when arithmetic operations are being performed. Division is accomplished by a division subroutine or by the automatic divide feature.

The IBM 1620 is a variable field length computer. The shortest admissible field is two digits (a field is a unit of information composed of related consecutively addressed digits); the longest field can be any number of digits within the capacity of available core storage positions. Not only can data fields be stored in core storage in varying sizes, but these same variable fields can also serve as factors in all arithmetic operations without being edited for size or position. Accuracy of results is ensured by automatic validity checking which operates when the data enters, exits, or is processed inside the system.

As shown in the Frontispiece, the console of the 1620 contains control keys, switches, an indicator panel, and a typewriter. The control keys and switches are used for manual or automatic operation of the system. The console panel provides visual indication of the status of various registers, indicators, and I/O conditions. The typewriter provides direct entry of data and instructions into core storage; it also provides a permanent log of the operator's intervention during the execution of a program.

Information is entered into the system by input devices: namely, the IBM 1621 Paper Tape Reader, the IBM 1622 Card Read-Punch, and the typewriter. The 1622 reads 80-column cards at a maximum rate of 250 cards per minute. The 1621 reads an 8-track paper tape at the rate of 150 characters per second. The operator's typing speed determines the rate of entry of information through the typewriter.

Output devices, the IBM 1622 Card Read-Punch, the 1624 Tape Punch, and a typewriter, record processed data. The typewriter prints at a maximum rate of 10 characters per second; the card punch and tape punch operate at the rate of 125 cards per minute and 15 characters per second, respectively.

Preparation of programs for the 1620 is simplified by IBM advanced programming systems and the IBM library of utility routines. These programs are similar to those used with the IBM 650, 1401, 704, 705, 709, 7070, and 7090 Data Processing Systems. The Symbolic Programming System (SPS) simplifies programming by reducing

the clerical work involved. SPS assembles a program written in mnemonic and symbolic notation by converting the symbols to machine language and assigning locations in core storage for both data and instructions. FORTRAN (FORmula TRANslation) is the term applied to another IBM programming system that translates a problem, expressed as a series of algebraic statements, into a complete machine language program, generating the step-by-step instructions necessary to solve the problem. A program written for the 1620 in FORTRAN can, with minor changes, also be compiled and executed on the IBM 7070, 704, 709, and 7090 Data Processing Systems. The library of utility routines provides a series of thoroughly tested programs that perform most of the more standardized computations and routine tasks occurring in many computer problems.

Stored Program Concept

The 1620 is a stored program computer; that is, it stores and executes its instructions internally. The computer can perform distinct operations such as adding, subtracting, multiplying, comparing, branching, and so on. It is directed to perform a specific operation by an instruction placed in core storage. To solve a problem or to process data, the programmer selects from various computer operations those necessary to do the desired work. A group of instructions representing the operations to be performed is called a program.

Once the program is placed in core storage, the computer can be directed to execute automatically the instructions composing the program. The program normally is executed in a sequential manner, that is, the computer starts with the first instruction and progresses serially through the program, interpreting and executing each instruction. However, this sequence of operations can be altered by the use of instructions that may direct the computer to an instruction located at other than the next sequential position.

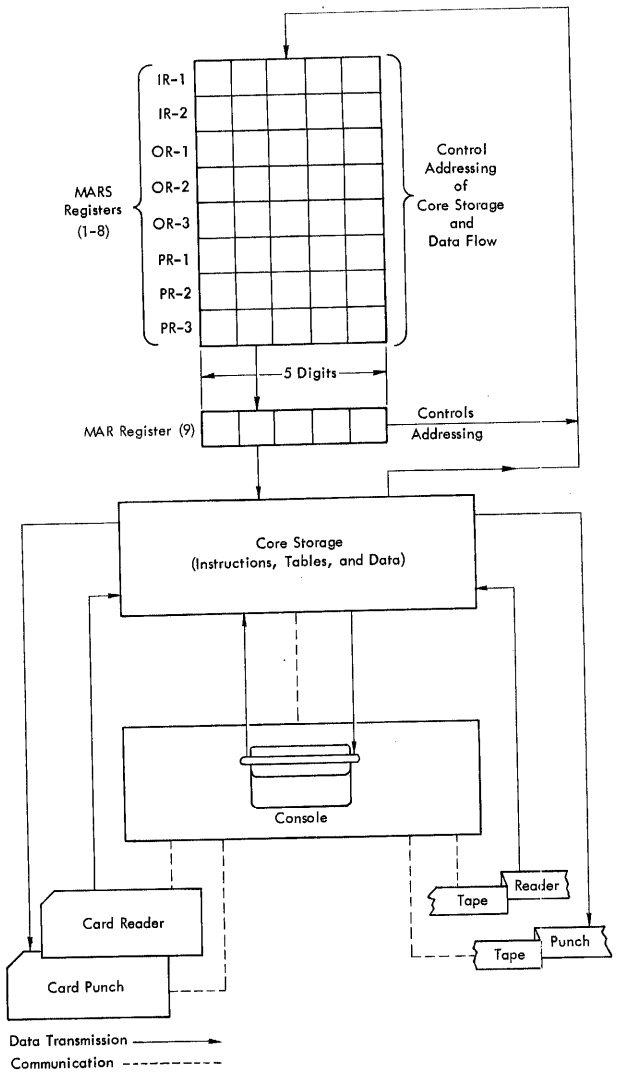


Figure 1. Diagram of 1620 System

System Configuration

As shown in Figure 1, all input enters core storage and all output is from core storage. Eight MARS (Memory Address Register Storage) registers (see Appendix E), comprising nonaddressable core storage, control the addressing of core storage locations and the flow of data during processing. These eight 5-digit registers are:

1. IR-1: Instruction Address Register 1
2. IR-2: Instruction Address Register 2
3. OR-1: Operand Address Register 1
4. OR-2: Operand Address Register 2
5. OR-3: Operand Address Register 3
6. PR-1: Product Address Register 1
7. PR-2: Product Address Register 2
8. PR-3: Product Address Register 3

In the basic 1620 System, these eight registers can contain 5-digit addresses from 00000 to 19999. The MARS registers address core storage through an additional 5-digit Memory Address Register (MAR), which also comprises nonaddressable cores. All addressing of core storage (for both instructions and data) is done through MAR. The operation code of the instruction determines the functions to be performed by the eight registers and the particular registers to be used.

The console is a connecting link between the computer and the I/O devices; it provides the operator with two-way communication to and from the 1620 System.

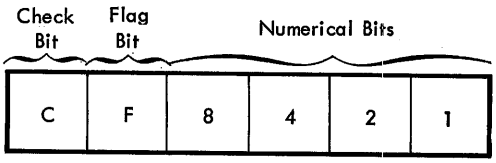
Data Representation

Data can be classified as digits, fields, or records, depending upon the operation in which the data is addressed. A field is composed of related digits (distance, time, etc.) treated as a unit of information. A record consists of related fields. Additional details can be found under FIELD and RECORD.

Digits

BCD BIT ARRAY

Each core storage position is addressable and can store one digit of information in binary-coded decimal (BCD) form (C, F, 8, 4, 2, and 1). The bit positions of each digit consist of four numerical bits, one flag (F) bit, and one check (C) bit.



The value of a decimal digit is the sum represented by the bits present in the 8, 4, 2, and 1 numerical bit positions. Only bit combinations whose sum is nine or less are used. A minus numerical expression has a sign flag in the units position of its field. Considering only the numerical bit positions, the decimal 6 is represented as 0110, the decimal 7 as 0111, etc., as shown in Figure 2, which gives configurations for both plus (without sign) and minus (with sign) numerals.

In Figure 2, it can be seen that the C and F bit configurations for each minus numeral are opposite to those for each plus numeral, while the 8, 4, 2, and 1 configurations are the same for both. This results from the fact that an odd-bit character is required for character validity (see also CHECK BIT).

CHECK BIT (C)

Each digit position within the computer must contain an odd number of coded bits, including a flag bit, if there is one, for correct parity. To create this odd-bit number, a C-bit is automatically added to each digit position when data enters core storage. Thereafter, during processing, a digit position with an even number of bits causes the machine to signal a parity error. A C-bit alone represents a plus zero.

FLAG BIT (F)

The flag bit is used in five ways, depending upon its location and the operation performed.

	Digit	C	F	8	4	2	1
No Field Definition or Sign	0	1	0	0	0	0	0
	1	0	0	0	0	0	1
	2	0	0	0	0	1	0
	3	1	0	0	0	1	1
	4	0	0	0	1	0	0
	5	1	0	0	1	0	1
	6	1	0	0	1	1	0
	7	0	0	0	1	1	1
	8	0	0	1	0	0	0
	9	1	0	1	0	0	1
Field Defined or Signed Numeral	0	0	1	0	0	0	0
	1	1	1	0	0	0	1
	2	1	1	0	0	1	0
	3	0	1	0	0	1	1
	4	1	1	0	1	0	0
	5	0	1	0	1	0	1
	6	0	1	0	1	1	0
	7	1	1	0	1	1	1
	8	1	1	1	0	0	0
	9	0	1	1	0	0	1

Figure 2. Bit Configuration for Decimal Digits 0 through 9

1. Sign Control
   A numerical data field is minus if the units digit contains a flag bit, and plus if the units digit does not contain a flag bit. Minus five is shown as $\bar{5}$; plus five is shown as 5. The BCD representations for minus and plus five are F-4-1 and C-4-1, respectively.
2. A flag bit as a field mark defines the leftmost (high-order) digit of a numerical data field. A field is shown as $\bar{X}XXX$, where the dash over the high-order digit is the field mark.
3. Carries
   Flag bits present in certain digits of the add table are interpreted in arithmetic operations as carries. For example, an eight with a carry is shown as $\bar{8}$. Flag bits are contained in table storage and transferred automatically, as required.
4. Minus Zero
   The F bit alone represents a minus zero.
5. Indirect Addressing (Special Feature)
   A flag bit over the units position of an instruction address (P or Q) indicates an indirect address, as explained in INDIRECT ADDRESSING.

RECORD MARK (‡)

The record mark is a nondecimal machine digit coded C-8-2. It is used primarily in input/output operations and in record transmission within the 1620 System; it cannot be used as a significant digit in an arithmetic or compare operation.

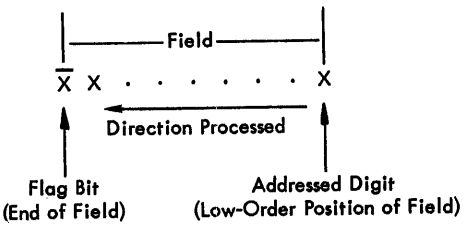
NUMERICAL BLANK

The numerical blank (coded C-8-4) is used for format control of blank columns when card punching, and cannot be used in arithmetic or compare operations. Read Numerically and Write Numerically, under INPUT/OUTPUT INSTRUCTIONS, give further details concerning the numerical blank.

Field

A field consists of a number of consecutively addressed digits related to arithmetic operations and internal field transmission. A field is addressed by its rightmost (low-order) digit which occupies the *highest-numbered* core storage position of the field. Fields are processed from *right to left* into successively *lower-numbered* core storage positions until a digit with a flag bit is sensed. The shortest admissible field consists of two digits: the addressed digit and the digit containing the flag bit or field mark.

Minus numerical fields are signed by a flag bit in the addressed low-order digit. The absence of a flag bit in the addressed digit is unconditionally interpreted as a plus field.

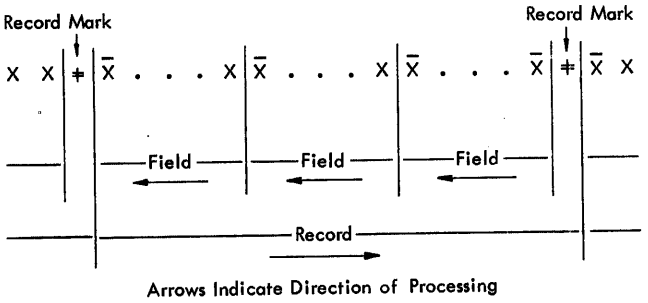


Record

A record consists of a field or fields of related data normally grouped for input/output operations and internal record transmission. A record is addressed at the *leftmost* (high-order) digit, which occupies the *lowest-numbered* core storage position of the record. Records are processed serially *from left to right* into successively *higher-numbered* core storage positions.

Output and internal record transmission are terminated when a record mark is sensed, except for card output which is terminated only after 80 columns are transferred to 1622 buffer storage. Buffer storage data transfer is described under IBM 1622 CARD READ-PUNCH UNIT.

A record is entered into core storage, starting at the addressed digit and continuing from left to right into successively higher-numbered core storage positions until terminated by an end-of-record signal from the input unit. The end-of-record signal from paper tape causes a record mark to be placed in core storage as the rightmost digit of the record. When input is from the typewriter, the Record Mark key must be depressed to place a record mark in core storage. When input is from punched cards, a record mark is automatically placed in core storage only when 0, 8, 2 are punched in a card column.



Magnetic Core Storage

The standard 1620 contains 20,000 addressable positions of magnetic core storage. Data stored in these cores is not affected by the manual turning on or off of power if care is taken to ensure that the 1620 System is in the manual mode when power is manually turned off. (A programmed "halt" automatically places the 1620 in the manual mode. See Appendix F for information on manual mode of operation.)

Additional storage is available in the IBM 1623, Models 1 and 2, to bring the total core storage capacity of the 1620 to 40,000 or 60,000 positions.

Core Array

Core storage in the 1620 is made up of 12 core planes as shown in Figure 3. Each core plane contains all cores for a specific value. The core planes are labeled C, F, 8, 4, 2, and 1. The even-address planes are the top six planes, and the odd-address planes are the bottom six planes. An even address has an even number as its units digit, while an odd address has an odd-numbered units digit.

The magnetic condition of the cores at any address determines which digit is stored at that address. A magnetic core is either in the magnetic "set" or "on" condition, or in the "reset" or "off" condition. If "set," the core contains a bit; if not, it is said to be "reset."

Readout from the 1620 core storage does not alter the "set" or "reset" condition of any core read out. Unless blocked by an immediate read-in of new data, a "data regeneration" occurs with each readout, leaving the cores in their "before readout" state. Thus read-in, but not readout, normally changes the data in the cores.

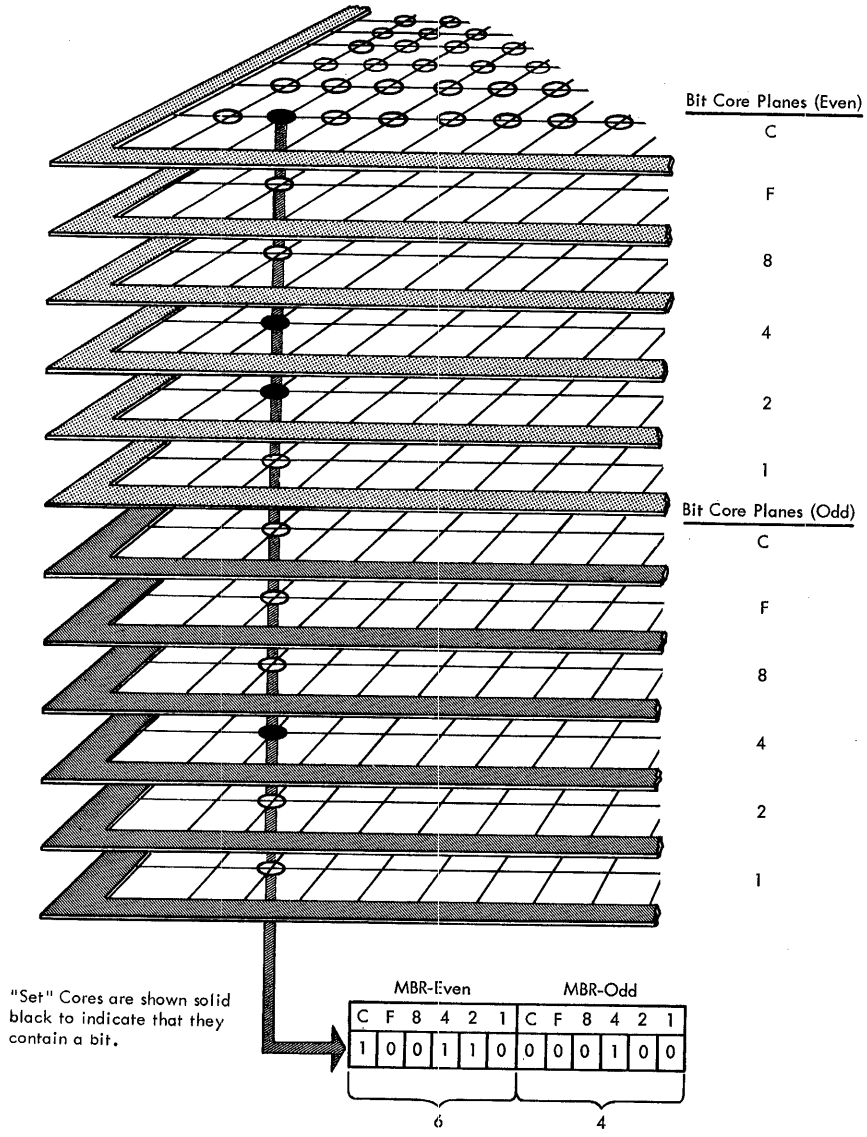


Figure 3. 1620 Core Array

Two-Character Transfer

During a core storage readout cycle, which takes 20 microseconds, all cores shown by the vertical line (one core in each bit plane, Figure 3) are read out at the same time. However, only "set" cores are read out to the 2-digit MBR (Memory Buffer Register) as bits.

The function of the MBR is to receive digits entering or leaving core storage. Digits leaving storage are "regenerated" through the MBR. In effect, the MBR is subdivided into two 1-digit registers: MBR-even and MBR-odd (abbreviated as MBR-E and MBR-O). From core storage, the even-address digits flow through MBR-E, while the odd-address digits flow through MBR-O. Digits entering core storage are handled similarly, under control of the units position of the associated MAR address. For example, an even digit in the units position of the MAR address causes data selection from MBR-E. Figure 3 shows that the C, 4, and 2 cores are "set" in the even-digit planes. The 4 core is "set" in the odd-digit planes. Thus, MBR-even and MBR-odd contain 64, the two-digit code for "U."

Because all 12 core planes are affected, any single core storage address affects two adjacent storage positions; one with an odd-numbered address and one with an even-numbered address. Even-numbered addresses affect the next higher position. For example, if the digit at address 00500 (even) is addressed and programmed to be read from core storage, the digit at address 00501 is also read. Odd-numbered addresses affect the next lower position. For example, address 00501 (odd) also

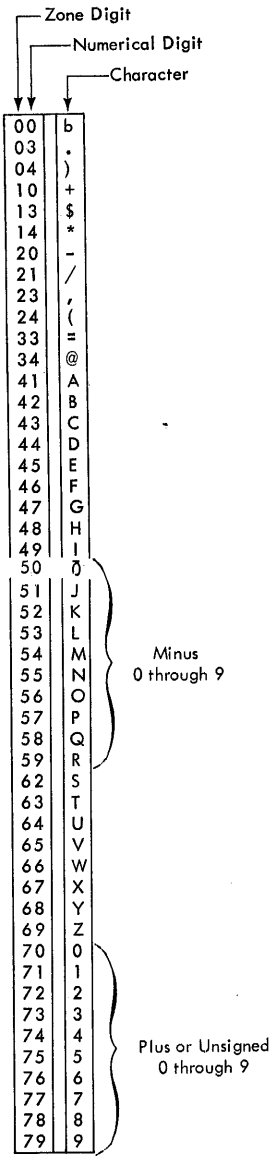


Figure 4. Alphameric Codes

affects address 00500. The selection of the digit to be used is determined by the operation to be performed. The digit actually addressed is moved to the 1-digit Memory Data Register (MDR).

Data transfer is illustrated under DATA FLOW in the Program Testing section of this manual.

Sequential Addressing

Core storage positions are addressed sequentially from 00000 to the highest-numbered address of the core storage positions installed: 19999, 39999, or 59999. The 1620 core storage has "wrap-around" storage: address 00000 follows the highest-numbered address when incrementing; the highest-numbered address follows 00000 when decrementing.

Modes of Operation

The 1620 System can either be in the numerical or the alphameric mode when reading or writing data. The input/output instruction determines the mode used. For given data, an output instruction should place the 1620 in the same mode as did the input instruction. All 1620 modes of operation are summarized in Appendix F.

Numerical Mode

One decimal digit is required in core storage to represent a numerical character. No alphabetic or special characters, except the record mark and numerical blank, can be represented in the numerical mode.

Alphameric Mode

Two decimal digits are required in core storage to represent an alphameric character, i.e., an alphabetic character, a special character, or a numerical character. A two-digit alphameric representation of numerical characters is provided to permit reading of mixed alphabetic, special, and numerical characters without changing from an alphameric to a numerical instruction. Figure 4 shows the zone and numerical digits that have been assigned to represent all the alphameric characters used in the 1620. Complete representations of all characters for all I/O devices are given in Appendix D. These two alphameric digits must be at adjacent core storage positions, and the zone digit must occupy the even address. The 1620 accomplishes this automatically during I/O operations by addressing the numerical digit to the odd address. Figure 3 shows the bit configuration for a "U" if the 1620 is in the alphameric mode.

All 1620 instructions fall into one of five general categories, according to function:

1. Arithmetic
2. Internal data transmission
3. Logic (compare and branch)
4. Input/output
5. Program control

During normal operation, program instructions are performed sequentially. Beginning at 00600, for example, the instructions located at 00612, 00624, and so on, are executed in order. However, this order can be changed if an instruction causes a branch to any but the next sequential instruction. This aspect of machine operation is covered under COMPARE INSTRUCTIONS and BRANCH INSTRUCTIONS.

In addition to the programming information given in this section, many programming suggestions are contained in the *IBM 1620 Data Processing System Bulletin, Program Writing and Testing* (Form J26-5547).

Instruction Characteristics

Instruction Format

The IBM 1620 uses a 12-digit machine language instruction divided into three parts: a 2-digit operation (OP) code, a 5-digit "P" address, and a 5-digit "Q" address. Figure 5 represents the format of an instruction as it appears in core storage. The O, P, and Q subscripted numbers are used throughout the manual.

In contrast to a data field, which is addressed at its rightmost (low-order) digit and read from right to left, instructions are addressed at O_0, the leftmost (high-order) digit, and read from left to right.

OP CODE

Upon initiation of an instruction, the OP code is placed in a 2-digit OP register and is analyzed to determine the operation to be performed. The address of an instruction must always be even, i.e., the O_0 digit of an operation code must be stored in an even-numbered address so that the OP register can receive both digits. Figure 6 lists the 1620 instructions, with their OP codes and associated mnemonics.

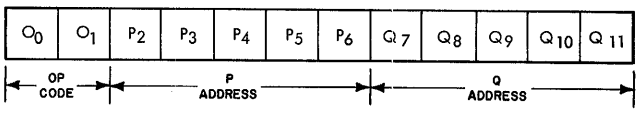


Figure 5. Instruction Format

P ADDRESS

The P address specifies: (1) location to which data is transmitted, (2) location to which the program branches, (3) location from which data is transmitted (output instructions), or (4) location of the alphameric field in the Transfer Numerical Strip and Transfer Numerical Fill instructions.

Arithmetic Instructions:	MNEMONIC	CODE
Add	A	21
Add Immediate	AM	11
Subtract	S	22
Subtract Immediate	SM	12
Multiply	M	23
Multiply Immediate	MM	13
Load Dividend *	LD	28
Load Dividend Immediate *	LDM	18
Divide *	D	29
Divide Immediate *	DM	19
Internal Data Transmission Instructions:		
Transmit Digit	TD	25
Transmit Digit Immediate	TDM	15
Transmit Field	TF	26
Transmit Field Immediate	TFM	16
Transmit Record	TR	31
Transfer Numerical Strip *	TNS	72
Transfer Numerical Fill *	TNF	73
Logic Instructions (Compare and Branch):		
Compare	C	24
Compare Immediate	CM	14
Branch	B	49
Branch No Flag	BNF	44
Branch No Record Mark	BNR	45
Branch On Digit	BD	43
Branch Indicator	BI	46
Branch No Indicator	BNI	47
Branch and Transmit	BT	27
Branch and Transmit Immediate	BTM	17
Branch Back	BB	42
Input-Output Instructions:		
Read Numerically	RN	36
Write Numerically	WN	38
Dump Numerically	DN	35
Read Alphamerically	RA	37
Write Alphamerically	WA	39
Control	K	34
Program Control Instructions:		
Set Flag	SF	32
Clear Flag	CF	33
Move Flag *	MF	71
Halt	H	48
No Operation	NOP	41

* Special Feature

Figure 6. 1620 Instructions

Q ADDRESS

The Q address specifies: (1) location from which data is transmitted, (2) which indicator is interrogated, (3) which input/output device is used, or (4) location of the numerical field in the Transfer Numerical Strip and Transfer Numerical Fill instructions. The OP code determines how the Q address is used in an instruction.

Immediate Instructions

Certain arithmetic, internal data transmission, and logic instructions are labeled "immediate." In immediate instructions, the Q part represents the data used rather than the address where the data is stored. The address of the units position (Q_{11}) is used for this data, and is always 11 higher than the address of the immediate instruction. The Q part of an instruction usually refers to the five digit positions Q_7, Q_8, Q_9, Q_{10}, and Q_{11}. However, when the Q field is used as data in an immediate instruction, the data is not restricted to five digits; the operation continues until a flag bit (field mark) is sensed. If more than five digits of data are required, the digits in excess of five must be valid data and must also be all or part of a valid P address (and of a valid, useful OP code if more than ten data digits are required). An example in which seven digits of data are used is given in Figure 7 for a TFM instruction. The Q data is transmitted to the field beginning at the P address. Transmission continues until terminated by the flag in the P part of the instruction.

CAUTION: Because a flag in the P_6 position indicates an indirect address, the use of six digits of data is not advisable with a 1620 containing Indirect Addressing (covered in a later section).

Operation Cycles

Each instruction or operation performed by the computer is divided into two parts: I (Instruction) cycle and E (Execution) cycle.

INSTRUCTION CYCLE

During the I-cycle, an instruction is read from core storage and analyzed to establish the type of operation that is to be performed. This analysis always takes eight 20-microsecond machine cycles, I_1 through I_8, for a total of 160 microseconds, and is displayed on the 1620 console during a Single Cycle Execute operation (explained elsewhere in this manual).

EXECUTION CYCLE

The operation specified by the instruction is carried out during the E-cycle. The number of machine cycles necessary to execute an instruction depends on the operation, the size of the data fields, and the signs of the fields (in arithmetic operations).

The last machine cycle of E-time is followed by the first machine cycle (of I-time) of the next instruction.

Execution Time

The formula for computing execution time follows the description of each instruction. Execution times for all instructions are summarized in Appendix A. The symbols used in the formulas are defined as follows:

D_P=Number of digits, including high-order zeros, in the field at the P address.

D_Q=Number of digits, including high-order zeros, in the field at the Q address.

$D_{Q'}$=Number of digits, including high-order zeros, in the data field of an immediate instruction.

D_Z=Number of positions compared prior to the detection of a digit other than zero.

T=Time, in microseconds (one microsecond = one millionth of a second).

Additional symbols used only in LD, LDM, D, and DM instructions are defined under EXECUTION TIME, following the explanation of the Divide instruction.

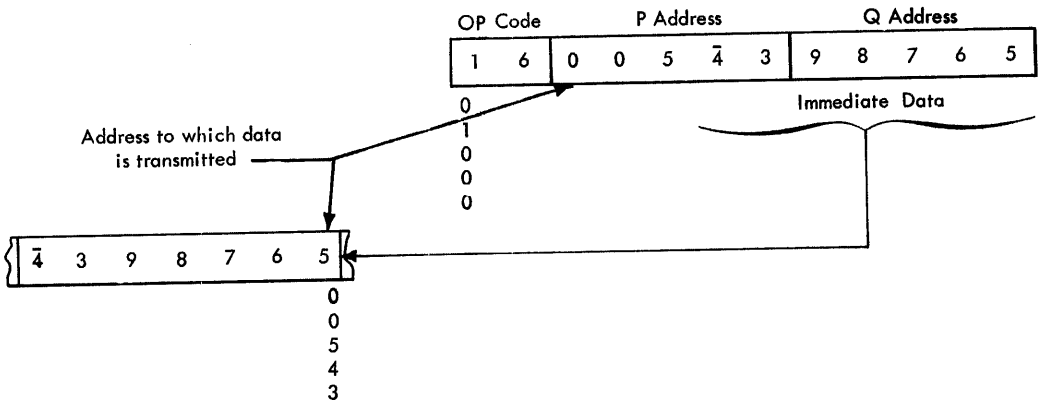


Figure 7. Transmit Field Immediate

Arithmetic Instructions

Characteristics that are similar in all 1620 arithmetic instructions are described in this section.

Description

Data flow, field length definition, sign indication, and indicator control are similar in all 1620 arithmetic instructions.

DATA FLOW AND FIELD LENGTH DEFINITION

Data is read serially from right to left (low-order to high) until terminated by a flag bit defining the high-order position of the field. Where the data in a field is shown as $\bar{2}85$, for example, the dash (flag bit) over the high-order digit indicates a field mark.

The minimum length of both the P and Q fields is two digits. The units digit contains the sign; at least one higher-order digit is needed for field definition (flag bit).

SIGN INDICATION

The algebraic sign of the factors and result is indicated by the presence (negative) or absence (positive) of a flag bit in the units position.

ARITHMETIC INDICATORS

Three arithmetic indicators and their associated console lights are controlled by arithmetic instructions and turned off by the Reset key on the 1620 Console. Functions of this key are explained under RESET KEY.

High/Positive (H/P). The high/positive indicator is turned on at the beginning of each arithmetic instruction and remains on if the result is positive and not zero. It is turned off if the result is negative or zero.

Equal/Zero (E/Z). The equal/zero indicator is turned on at the beginning of each arithmetic instruction and remains on if the result is zero. It is turned off if the result is not zero.

Overflow (O'flow). The overflow indicator is turned on during the execution of add, subtract, and compare instructions, if either of the following conditions exists:

1. The number of digits in the Q data exceeds the number of digits in the P data. Only the number of digits in the Q data that equal the number of digits in the P data are used in developing the result.
2. The result causes a carry beyond the high-order position of the initial field at P. The carry is lost.

This indicator is also turned on during a divide operation if more than nine successful subtractions occur, indicating mispositioning of the divisor.

The overflow indicator is turned off by the execution of a Branch Indicator or Branch No Indicator instruction, or by manual depression of the reset key. It is not turned on automatically at the beginning of each arithmetic instruction.

TABLE LOOKUP

A unique method of doing arithmetic calculations is used in the 1620. Two tables (Multiply and Add) stored in the "table area" of core storage are automatically referred to by the computer during arithmetic operations. The positions of core storage containing the table data are addressable but must not be altered; altering can cause incorrect arithmetic operations to result.

Three hundred positions of core storage have been assigned to the table area. Two hundred positions, 00100 through 00299, are assigned to the storage of the Multiply table. One hundred positions, 00300 through 00399, are assigned to the storage of the Add table used in all arithmetic operations (see Appendixes B and C). A digit with a flag bit in the table indicates that a carry is associated with that digit.

In addition, twenty positions, 00080 through 00099, are used to receive the product or partial product in multiply operations.

Add (A-21)

Description. The data in the field at the Q address is added to the data in the field at the P address and the sum replaces the P field data. The Q field data remains unchanged.

In Figure 8, the sum ($\bar{1}4$) is "looked up" in the add table and replaces the $\bar{1}2$ at 00500 (P address). The field mark remains at the high-order position. When the sum is zero, the sign of the P field is retained. For sums other than zero, the sign of the larger value field is retained. High-order zeros are supplied if the number of significant digits in the Q field is less than the number of significant digits in the initial field at P.

The high/positive indicator is on if the sum is positive and not zero; the equal/zero indicator is on if the sum is zero. Neither indicator is on if the sum is negative.

Execution Time. Execution time varies according to the number of digits (high-order zeros included) in the field at P and according to whether recomplementing is necessary. Recomplement time must be added to the basic time when the signs of the fields at the Q and P addresses are different initially and the absolute numerical value of the Q field is greater than the absolute numerical value of the P field.

Basic Execution Time: $T=160+80\ D_P$

Recomplement time: $T=80\ D_P$

Add Immediate (AM-11)

Description. The description is the same as that for Add (A-21) except that the data in the Q part of the instruction is used as the Q data. For example, if the OP

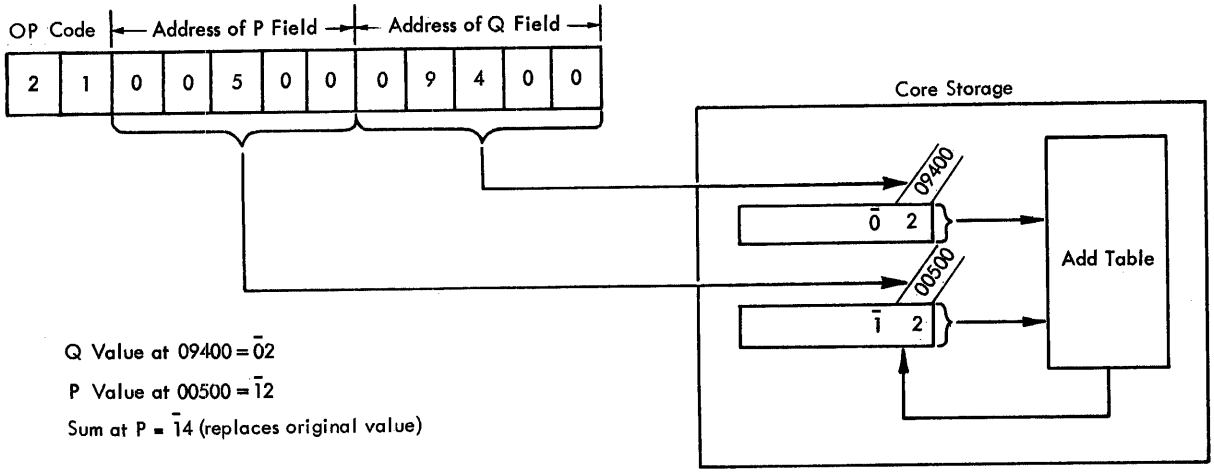


Figure 8. Add Operation

code were 11 in Figure 8, the Q data would be $\overline{0}9400$ and the result would be 12, $(00+\overline{0}2)$. The three high-order positions of the Q data ($\overline{0}94$) are not used, because the P data flag bit (over the one in $\overline{1}2$) stops the add operation. The overflow indicator is turned on because the Q data field ($\overline{0}9400$) exceeds the two digits of data ($\overline{1}2$) contained in the field defined by the P address (00500).

Execution Time. Same as Add (A-21).

Subtract (S-22)

Description. The data in the field at the Q address is subtracted from the data in the field at the P address and the difference replaces the data in the field at the P address. The data in the field at the Q address remains unchanged.

The data in the field at the Q address is complemented if it has the same sign as the data in the field at the P address. The sign control chart in Figure 9 shows the conditions under which the result (stored at the P address) is recomplemented.

A zero result retains the sign of the field at the P address. The sign of a result, other than zero, is determined by algebraic analysis of the P and Q fields. High-order zeros are supplied if the number of significant digits in the Q field is less than the number of significant digits in the initial field at the P address.

The high/positive indicator is on if the difference is positive and not zero; the equal/zero indicator is on if the difference is zero. Neither indicator is on if the difference is negative.

Execution Time. Execution time is computed by using the Add (A-21) formula. Recomplement time is added to basic time when the signs of the fields at the Q and P addresses are the same initially and the absolute numercial value of the field at the Q address is greater than the absolute numerical value of the field at the P address (Figure 9).

	ADD				SUBTRACT			
Sign of P Field	+	+	−	−	+	+	−	−
Sign of Q Field	+	−	+	−	+	−	+	−
Stored P Field Sign	+	+	−	−	+	+	−	−
True or Complement Add Q Field	True	Comp	Comp	True	Comp	True	True	Comp
Recomplement Answer if Q Field Value is Greater than P Field Value		X	X		X			X
Resulting Sign of P Field (Change on Recomplement)	+	−	+	−	−	+	−	+

Figure 9. Sign Control Chart

Subtract Immediate (SM-12)

Description. The description is the same as for Subtract (S-22) except that the digits in the Q part of the instruction are used as the Q data.

Execution Time. Same as Subtract (S-22).

Multiply (M-23)

Description. The data in the field at the P address is multiplied by the data in the field at the Q address, and the result (product) is placed in core storage, beginning at position 00099 and extending through successively lower-numbered positions. The data in the fields at the Q and P addresses are not changed by the operation.

In Figure 10, the multiplicand ($\bar{1}2$) at 00500 is multiplied by the multiplier ($\bar{0}2$) at 09400. The product ($\bar{0}024$) is developed and stored at 00096-00099.

The 20 digits of the area in core storage specified as the "product area" (positions 00080 through 00099) are automatically cleared to zeros before multiplication begins. Formation of the product then proceeds serially from right to left until terminated by the flag bit marking the high-order position of the field at the Q address. A flag bit is stored in the high-order position of the product, and its sign is indicated by the presence (negative) or absence (positive) of a flag bit in position 00099. A zero product may have a negative or positive sign, depending upon the signs of the fields at the Q and P addresses.

The number of digits in the product is equal to the sum of the digits (high-order zeros included) in the fields at the Q and P addresses. The size of the product is limited only by the core storage positions available. A product longer than the 20 positions of the product area may be formed, but positions in excess of 20 digits must be cleared to zeros by program instructions preceding the Multiply instruction.

It is possible to develop a product so large that it extends from its units position (location 00099), leftward to location 00000, continues at the highest-order core storage location (19999, 39999, or 59999), and finally terminates with its high-order digit at some location lower than the highest-order location. The overflow indicator is not turned on when the product exceeds 20 digits in length. The high/positive indicator is on if the product is positive and not zero; the equal/zero indicator is on if the product is zero. Neither indicator is on if the product is negative.

Execution Time. The execution time varies according to the number of digits in the fields at the Q and P addresses.

$T=560+40\ D_Q+168\ D_Q\ D_P$

Multiply Immediate (MM-13)

Description. The description for Multiply (M-23) applies except that the data in the Q part of the instruction is used in place of the data in the field at the Q address.

Execution Time. $T=560+D_{Q'}+168\ D_{Q'}\ D_P$

Automatic Division — Special Feature

Automatic division simplifies programming and increases the processing speed of division problems by two to four times that of programmed routines. Either

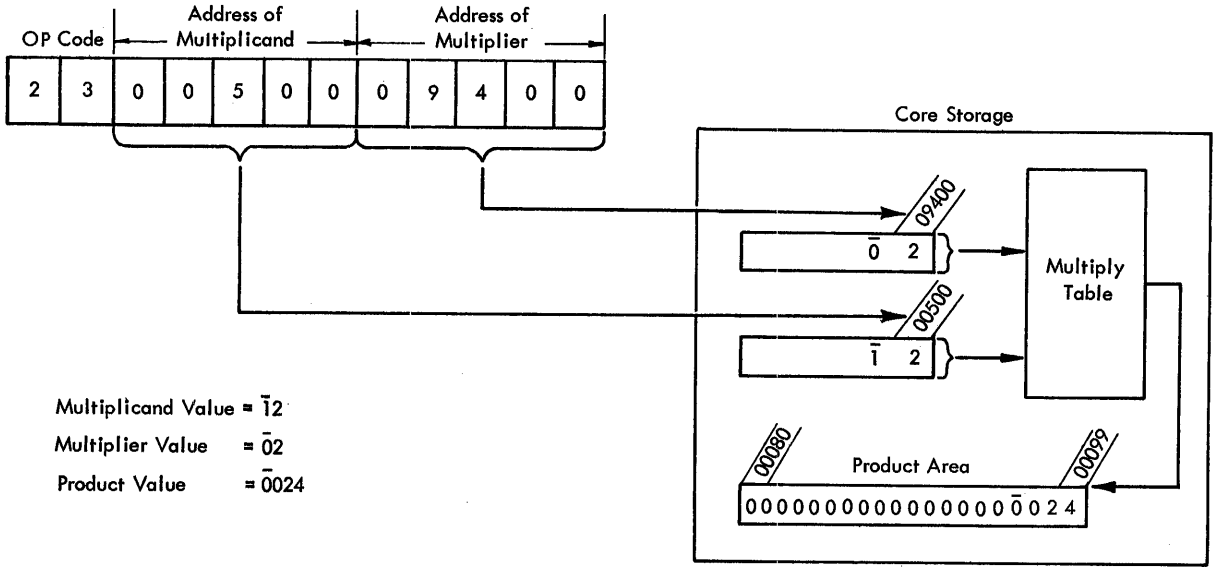


Figure 10. Multiply Operation

the Divide or Divide Immediate command causes division to occur. Two additional commands are also provided to position the dividend in core storage. No practical limitations are placed upon the size of the dividend, divisor, or quotient.

A quotient and remainder of 20 digits are developed in the product area (00080-00099). When the quotient plus the remainder exceeds 20 digits, core storage positions lower than 00080 (00079, 00078, etc.) must be reset to zeros by programming. One additional position should also be cleared to allow for a possible overdraw. For example, if 25 positions were required for the quotient and remainder, 00074-00079 would be reset to zeros before the divide command was given.

OPERATION

The four instructions provided with the divide feature are:

Load Dividend (LD-28)
Load Dividend Immediate (LDM-18)
Divide (D-29)
Divide Immediate (DM-19)

The "immediate" commands function in the same way as the basic commands except that the Q portion of the instruction contains the data to be used rather than the core storage address of the data field.

The dividend must be stored in the product area before a Divide command is given. The Load Dividend instruction may be used to satisfy this requirement.

Load Dividend (LD-28)

The product area (00080-00099) is automatically reset to zeros. The dividend (Q address) is transmitted to the product area (P address), beginning at the low-order dividend digit and terminating at the flag bit marking the high-order position of the dividend field. The P address is 00099 minus the number of zero positions desired to the right of the dividend.

The algebraic sign of the dividend is automatically placed in location 00099, regardless of where the low-order dividend digit is placed by the P address. A flag bit automatically marks the high-order digit of the dividend.

Example: Two Load Dividend instructions and one Load Dividend Immediate instruction are shown in Figure 11.

1. The Load Dividend instruction, 28 00096 00650, causes the low-order position of the dividend to be placed at 00096. The sign (minus) is stored at 00099.
2. The Load Dividend instruction, 28 00099 00650, causes the low-order position of the dividend to be placed at 00099. The sign (plus) is stored at 00099.
3. The Load Dividend Immediate instruction, 18 00098 $\bar{0}$0650, causes the low-order position of the dividend (the Q part of the instruction) to be placed in the field beginning at 00098. The sign (plus) is stored at 00099.

Divide (D-29)

The divisor (Q address) is successively subtracted from the dividend. The P address of the Divide instruction positions the divisor for the first subtraction from the high-order position(s) of the dividend, as in manual division. The P address is determined by subtracting the number of digits in the quotient from 100.

Examples: Problem 1: $\bar{4}906 \div \bar{2}3 = \bar{0}213$ and a remainder of $\bar{0}7$. Figure 12 shows the manner in which the 1620 solves this problem.

Problem 2: $-212\ (\bar{2}1\bar{2}) \div \bar{2}4 = -8.83\ (\bar{0}0883\bar{3})$ and a remainder of $\bar{0}8$. Figure 13 shows how the 1620 solves this problem.

As illustrated in these examples, each subtraction without overdraw causes the quotient digit to be increased by 1. Quotient digits are developed in the units positions of the Sense and Branch register. An overdraw initiates a correction cycle (the divisor is added once) and the next subtraction occurs one place to the right.

Instruction	Data at Core Storage Address 00650	Description	00080	00081	00082	....	00092	00093	00094	00095	00096	00097	00098	00099
(1) 28 00096 00650	$\bar{2}136\bar{5}$	Load Dividend	0	0	0	....	$\bar{2}$	1	3	6	5	0	0	$\bar{0}$
(2) 28 00099 00650	$\bar{0}1234$	Load Dividend	0	0	0	....	0	0	0	$\bar{0}$	1	2	3	4
(3) 18 00098 $\bar{0}$0650	56789	Load Dividend Immediate	0	0	0	....	0	0	$\bar{0}$	0	6	5	0	0

Figure 11. Load Dividend Instructions

Instruction	Data at Core Storage Addresses 00500	00600	Description	00092	00093	00094	00095	00096	00097	00098	00099
	$\bar{0}$0500	$\bar{0}$0600									
28 00099 00500	$\bar{4}$906	$\bar{2}$3	Load dividend	0	0	0	0	$\bar{4}$	9	0	6
29 00096 00600			Subtract divisor			−	2	3			
			Overdraw			9	8	1			
			Add divisor back to correct overdraw.			+	2	3			
						0	0	4			
			Store first (high-order) digit of quotient (0) and flag bit	0	0	$\bar{0}$	0	4	9	0	6
			Subtract divisor one place to the right				−	2	3		
			No overdraw					2	6		
			Subtract divisor				−	2	3		
			No overdraw				0	0	3		
			Subtract divisor				−	2	3		
			Overdraw				9	8	0		
			Add divisor back to correct overdraw				+	2	3		
							0	0	3		
			Store second digit of quotient (2)	0	0	$\bar{0}$	2	0	3	0	6
			Subtract divisor one place to the right					−	2	3	
			No overdraw					0	0	7	
			Subtract divisor					−	2	3	
			Overdraw					9	8	4	
			Add back divisor to correct overdraw					+	2	3	
								0	0	7	
			Store third digit of quotient (1)	0	0	$\bar{0}$	2	1	0	7	6
			Subtract divisor one place to the right						−	2	3
			No overdraw						0	5	3
			Subtract divisor						-	2	3
			No overdraw						0	3	0
			Subtract divisor						−	2	3
			No overdraw						0	0	7
			Subtract divisor						−	2	3
			Overdraw						9	8	4
			Add back divisor to correct overdraw						+	2	3
									0	0	7
			Store fourth digit of quotient (3) and flag bit, if negative. Operation stops with quotient ($\bar{0}$213) and remainder ($\bar{0}$7) in product area.	0	0	$\bar{0}$	2	1	3	$\bar{0}$	7

Figure 12. Divide, Problem 1

Instruction	Description	00650	00500	00090	00091	00092	00093	00094	00095	00096	00097	00098	00099
	Data	$\bar{2}4$	$\bar{2}1\bar{2}$										
LD 28 00097 00500	Reset 00080 - 00099 to zeros. Transmit dividend to 00097. Dividend sign to 00099.			0	0	0	0	0	$\bar{2}$	1	2	0	$\bar{0}$
D 29 00095 00650	Subtract divisor from dividend starting at 00095.							−	2	4			
	Overdraw							9	7	8			
	Correction							+	2	4			
								0	0	2			
	Store first quotient digit (0) and flag bit			0	0	0	$\boxed{\bar{0}}$	0	2	1	2	0	$\bar{0}$
	Subtract one place to the right							−	2	4			
	Overdraw							9	9	7			
	Correction							+	2	4			
								0	2	1			
	Store 2nd quotient digit (0)			0	0	0	$\bar{0}$	$\boxed{0}$	2	1	2	0	$\bar{0}$
	Subtract one place to the right								−	2	4		
	Successful subtraction								1	8	8		
	7 more successful subtractions (7 x 24 = 168)								−1	6	8		
									0	2	0		
									−	2	4		
	Overdraw								9	9	6		
	Correction								+	2	4		
									0	2	0		
	Store quotient digit (8)			0	0	0	$\bar{0}$	0	$\boxed{8}$	2	0	0	$\bar{0}$
	8 successful subtractions (8 x 24 = 192)								−	1	9	2	
	(Overdraw and correction not shown)									0	0	8	
	Store quotient digit (8)			0	0	0	$\bar{0}$	0	8	$\boxed{8}$	0	8	$\bar{0}$
	3 successful subtractions (3 x 24 = 72)										−	7	2
												0	8
											−	2	4
	Overdraw										9	8	4
	Correction										+	2	4
												0	0	8
	Store quotient digit (3)			0	0	0	$\bar{0}$	0	8	8	$\boxed{3}$	0	$\bar{8}$
	Store flag over high-order position of remainder. Sign of quotient over units position (00099 - length of divisor).			0	0	0	$\bar{0}$	0	8	8	$\bar{3}$	$\bar{0}$	$\bar{8}$

Figure 13. Divide, Problem 2

The first (high-order) quotient digit is stored at the address equal to the P address of the divide instruction minus the length of the divisor. A flag bit is generated and stored with the first quotient digit. Subsequent quotient digits are stored to the right of the last-stored quotient digit. Division is terminated, after the last quotient digit is developed by subtractions, with the units position of the divisor at 00099.

The quotient and remainder replace the dividend in the product area. The address of the quotient is 00099 minus the length of the divisor. The algebraic sign of the quotient (determined by the signs of the dividend and divisor) is automatically placed in the low-order position of the quotient. The address of the remainder is 00099. A flag bit is automatically placed in the high-order position. The remainder has the sign of the dividend and the same number of digits as the divisor.

The high/positive indicator is on if the quotient is positive and not zero; the equal/zero indicator is on if the quotient is zero. Neither indicator is on if the quotient is negative.

The quotient must be at least two digits in length. One position is required for the sign and one for the field mark (flag bit).

Incorrect Divisor Positioning. The following error conditions are caused by an incorrect P address in the Divide instruction:

1. Overflow. As illustrated in Figure 14, an incorrectly positioned divisor can cause more than nine successful subtractions and an incorrect quotient.

Instruction	Description	00650	00090	00091	00092	00093	00094	00095	00096	00097	00098	00099
		$\bar{2}$ 1	0	0	0	0	0	$\bar{2}$	1	2	0	$\bar{0}$
D 29 00097 00650	Successful subtraction No. 1							−	2	1		
								1	9	1		
	" " No. 2							−	2	1		
								1	7	0		
	" " No. 3							−	2	1		
								1	4	9		
	" " No. 4							−	2	1		
								1	2	8		
	" " No. 5							−	2	1		
								1	0	7		
	" " No. 6							−	2	1		
								0	8	6		
	" " No. 7							−	2	1		
								0	6	5		
	" " No. 8							−	2	1		
								0	4	4		
	" " No. 9							−	2	1		
								0	2	3		
	" " No. 10							−	2	1		
		0	0	0	0	0	0	0	0	2	0	$\bar{0}$

Note that a field-length flag is *not* generated and stored in the product area. If the field is to be used for further operations, the program must provide for a flag to be set.

Figure 14. Divide Overflow

The divide operation is terminated, the overflow indicator and Overflow Arithmetic Check light are turned on, but processing does not stop unless the Overflow Check switch is set to STOP. (Functions of the overflow check indicator and switch are described in the Check Indicator section under CONSOLE.)

2. Loss of one or more high-order digits of the dividend. The high-order digit of the dividend is assumed by the 1620 to be one position to the left of the high-order digit of the divisor. Figure 15 shows how the high-order digits of the dividend are lost if the divisor is positioned too far to the right. Processing continues with no indication of an incorrect quotient.
3. Incorrect termination. The Divide instruction may not be terminated, by subtraction, at 00099 unless the P address (if greater than 00099) is 10000 or greater.

Execution Time. The total execution time (in microseconds) is the sum of the Load Dividend and Divide instruction times.

Load Dividend (LD-28) $=400+40\ D_N$
Divide (D-29) $=160+520\ D_V Q_T+740\ Q_T$
Total Time $=560+40\ D_N+520\ D_V Q_T+740\ Q_T$
D_N = Number of digits in dividend.
D_V = Number of digits in divisor
Q_T = Number of digits in quotient

This formula assumes the average value of each quotient digit to be 4.5.

SUMMARY OF AUTOMATIC DIVISION RULES

1. Load Dividend (LD-28 or LDM-18)
   a. P address = 00099 minus the number of zeros desired to the right of the units position of the dividend.
   b. Q address = core storage address of the dividend.
2. Divide (D-29 or DM-19)
   a. P address = 00100 minus the length of the quotient. The quotient length must be at least two digits.
   b. Q address = core storage address of the divisor.
3. Quotient address = 00099 minus the length of the divisor.

Instruction	Description	00650	00095	00096	00097	00098	00099
29 00098 00650	Divide (Incorrect P Address)	$\bar{1}9$	$\bar{2}$	0	2	3	$\bar{0}$
				−	1	9	
				0	0	4	
				−	1	9	
				9	8	5	
				+	1	9	
				0	0	4	
			$\bar{2}$	$\bar{1}$	0	4	0
					−	1	9
					0	2	1
					−	1	9
					0	0	2
					−	1	9
					9	8	3
					+	1	9
					0	0	2
			$\bar{2}$	$\bar{1}$	2	$\bar{0}$	2

Figure 15. Divide, Incorrect Position of Divisor

4. Remainder address = 00099.
5. Sign of quotient: determined by the algebraic signs of the dividend and divisor.
6. Sign of remainder: same as that of the dividend.

Internal Data Transmission Instructions

The instructions that follow provide for the transmission of data from one core storage address to another.

Transmit Digit (TD-25)

Description. The single digit at the Q address is transmitted to the P address. The digit at the Q address is not changed. If a flag bit is at the Q address, it is also transmitted.

Execution Time. Execution time is a constant 200 microseconds.

Transmit Digit Immediate (TDM-15)

Description. The single digit in the units position (Q_{11}) of the Q part of the instruction is transferred to the P address. The original digit in the units position of the Q part remains unchanged.

In Figure 16, if the Transmit Digit Immediate instruction (15 09400 00500) is at 09200, the digit (0) at 09211 is transmitted to the P address (09400) of the instruction. The 3 at 09400 is replaced by the 0, which also remains in 09211.

If a flag bit is located at Q_{11}, it is also transmitted.

Execution Time. Execution time is a constant 200 microseconds.

Transmit Field (TF-26)

Description. The data in the field at the Q address is transmitted to the field at the P address. The data in the field at the Q address remains unchanged by the transfer.

Transmission proceeds serially from right to left until terminated by the flag bit that marks the high-order position of the field at the Q address. The transmitted field replaces all data in the field at the P address, including flag bits.

In Figure 17, the data ($\bar{2}14$) in the field defined by the Q address (00500) replaces the data (128) in the field defined by the P address (09400). The data at the Q address is not changed.

Execution Time. The execution time varies according to the number of digits (high-order zeros included) in the field at Q. $T = 160 + 40\ D_Q$

Transmit Field Immediate (TFM-16)

Description. The description is the same as that for Transmit Field (TF-26) except that the data in the Q part of the instruction is used in place of the data at the Q address.

Execution Time. $T = 160 + 40\ D_Q$.

Transmit Record (TR-31)

Description. The record, which has its high-order position at the Q address, is transmitted to the P address and successively higher core storage locations. The record at the Q address remains unchanged by the transfer.

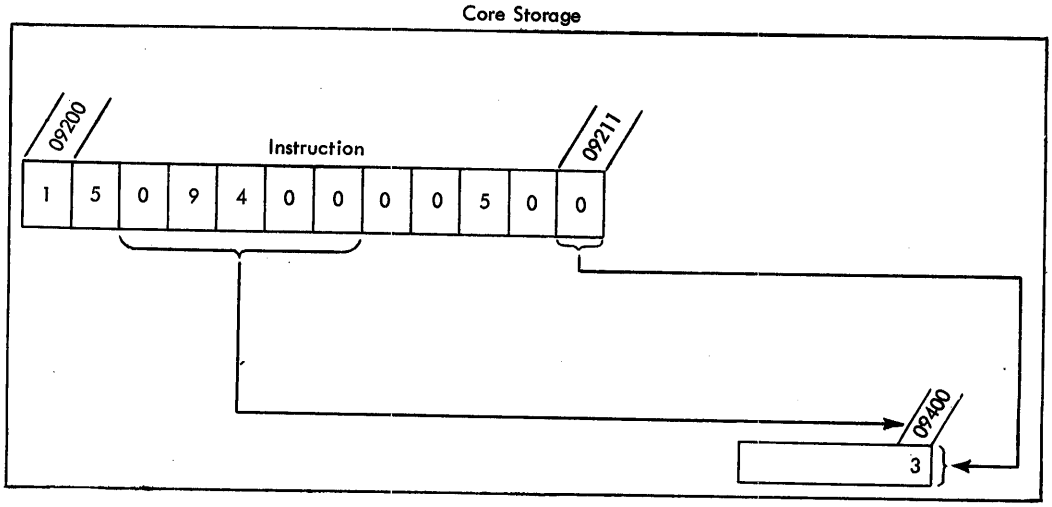


Figure 16. Transmit Digit Immediate

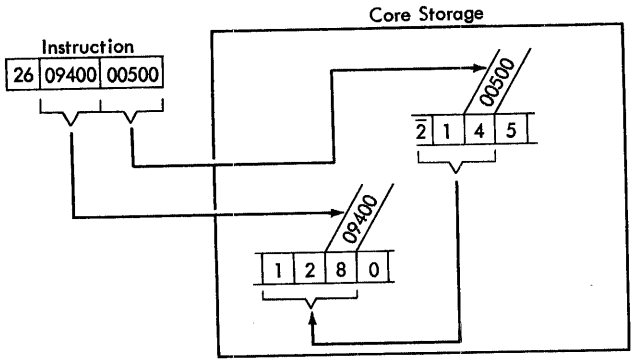


Figure 17. Transmit Field Operation

Transmission proceeds serially from left to right until terminated by a record mark. The transmitted record replaces all data in the record at the P address, including flag bits and the record mark.

Execution Time. The execution time varies according to the number of characters (high-order zeros included) in the record at the Q address. $T=160+40\ D_Q$

Transfer Numerical Strip (TNS-72) — Special Feature

Description. This instruction converts numerical data in the two-digit alphameric mode into single-digit numerical data, with sign. The units numerical position of the alphameric field (see Figure 4) is specified by the P address of the instruction and must always be an odd-numbered core storage location. The units position of the numerical field is specified by the Q address. Transmission proceeds from the position addressed, through successively lower-numbered core storage locations, until a flag bit is sensed in other than the units position of the numerical field. The flag bit must be placed in the numerical field prior to the TNS instruction to define the high-order position. It remains unchanged by the instruction.

The zone digits in the even-numbered core storage locations of the alphameric (P) field are ignored except for a five, two, or one in the units zone position. A five in a units zone position of an alphamerically coded number field indicates a negative number read from an input card or paper tape. A two in a units zone position occurs when an X alone, representing a negative zero, is read from input card or paper tape. A one occurs when a negative zero (X, 0) is read from paper tape. A five, two, or one in the units zone position is converted by TNS to a flag bit over the units digit of the numerical (Q) field. Any number other than a five, two, or one results in no flag over the units digit.

The digit in each odd-numbered core storage position of the alphameric field is transmitted without change to the associated position of the numerical field, concluding with the digit transmitted to the high-order position of the numerical field containing the flag that defines the field. Except for the field flag, all previous contents of the numerical field are erased by the new contents. The erasure includes any sign flag contained in the units position to designate a previous negative value. The alphameric field remains unchanged.

Flag bits in the even-numbered zone positions of the alphameric field are ignored. However, flag bits present in the odd-numbered core storage locations of the alphameric field are transmitted to the corresponding positions of the numerical field.

Because such flag bits, when transmitted, may affect the length or sign of the numerical field, all flag bit positions of the alphameric field should be cleared by instructions at the beginning of the program. Such extraneous flag bits are the result of previous use of the core storage locations and the fact that the Read Alphamerically instruction ignores the flag bits in the read-in field. If flags are developed in the alphameric field during the program, care should be taken before the TNS instruction that the flags do not disturb the numerical field. Figure 18 illustrates the operation of Transfer Numerical Strip.

Execution Time. $T=160+40\ D_P$

Transfer Numerical Fill (TNF-73)—Special Feature

Description. This instruction moves and expands single-digit numerical data with a sign into two-digit alphameric data. The units numerical position of the alphameric field (see Figure 4) is specified by the P address of the instruction and must always be an odd-numbered core storage location. The units position of the numerical field is specified by the Q address. Transmission proceeds from the location addressed, through successively lower-numbered core storage locations, until a flag bit is sensed in other than the units position of the numerical (Q) field. The field flag bit that terminates the transfer remains in the Q field and is neither transmitted nor converted.

A sign flag in the addressed units position of the numerical field is converted to a five in the even-numbered units zone position of the alphameric (P) field. Absence of a flag in the units position of the numerical field results in a seven being placed in the even-numbered units zone position. All other even-numbered zone positions of the alphameric field are automatically filled with sevens.

During a Write Alphamerically instruction, a negative zero, represented by a zone digit five and a numerical digit zero, is converted to X coding in paper

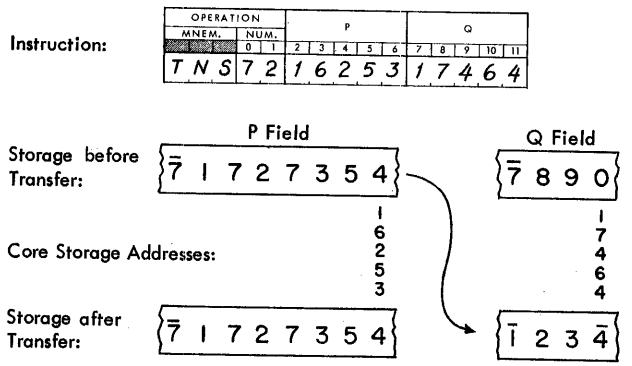


Figure 18. Transfer Numerical Strip

tape, to X, 0 coding in an output card, and to a minus sign (−) on the typewriter. All other negative quantities in the units position having a zone digit five are typed out as the letters J through R, corresponding to the digits 1 through 9, respectively (see Figure 4).

The digits in the numerical field, including the digit in the high-order (flagged) position, are transmitted without change to the corresponding odd-numbered positions of the alphameric field. All of the previous contents of the alphameric field, including flag bits, are erased by the new contents. The numerical field remains unchanged. Figure 19 illustrates the operation of Transfer Numerical Fill.

Execution Time. $T=160+40\ D_P$

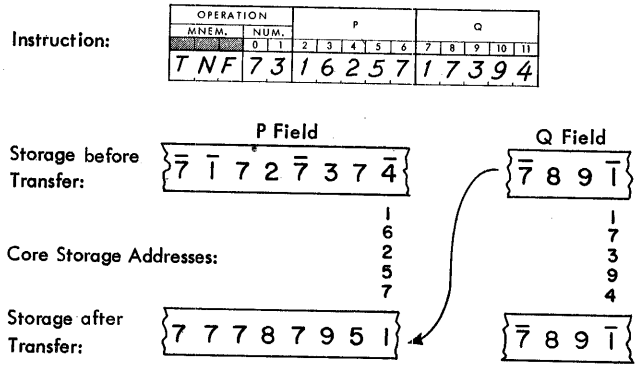


Figure 19. Transfer Numerical Fill

Compare Instructions

Compare (C-24)

Description. The data in the field at the Q address is compared with the data in the field at the P address to determine if the latter is greater than or equal to the former. This is accomplished by subtracting the Q address data (data in the field defined by the Q address) from the P address data (data in the field defined by the P address) and by discarding the digits of the difference. Neither field is altered. The result of the comparison is shown by the on/off conditions of the indicators.

Condition (Algebraic)	Indicator	
	High/Positive	Equal/Zero
P Greater than Q	ON	OFF
P Less than Q	OFF	OFF
P Equal to Q	OFF	ON
P = Data in Field at P Address Q = Data in Field at Q Address		

Comparison proceeds serially from right to left (low-order to high-order) until terminated by the flag bit marking the high-order position of the field at the P address. High-order zeros are supplied when the size of the Q field is less than that of the P field. The high/positive indicator is turned on if the P address data is algebraically higher than the Q address data, and off, if not higher. The equal/zero indicator is turned on if the P address data is algebraically equal to the Q address data, and off, if not equal.

In Figure 20, the Q address data (12) is subtracted from the P address data (22). The difference (+10) is discarded and the indicators are set as follows:

High/Positive: ON

Equal/Zero: OFF

Comparison is completed only if the number of digits in the field at the P address (high-order zeros included) is greater than or equal to the number of digits in the field at the Q address (high-order zeros included). If this condition is not met, the overflow indicator is turned on and the extra digits in the field at the Q address are not used. However, the result of the comparison (ignoring the extra digits) is correct to the point where the comparison was terminated.

If the signs of the two fields are different initially, comparison continues until a digit other than zero is detected in either the P or Q field. The on/off conditions of the indicators show the positive field to be the greater. When two fields containing all zeros are compared, the signs are disregarded and the equal/zero indicator is turned on.

Following are the ascending collating sequences upon which the results of comparisons are based:

1. Numerical Sequence:
   0123456789

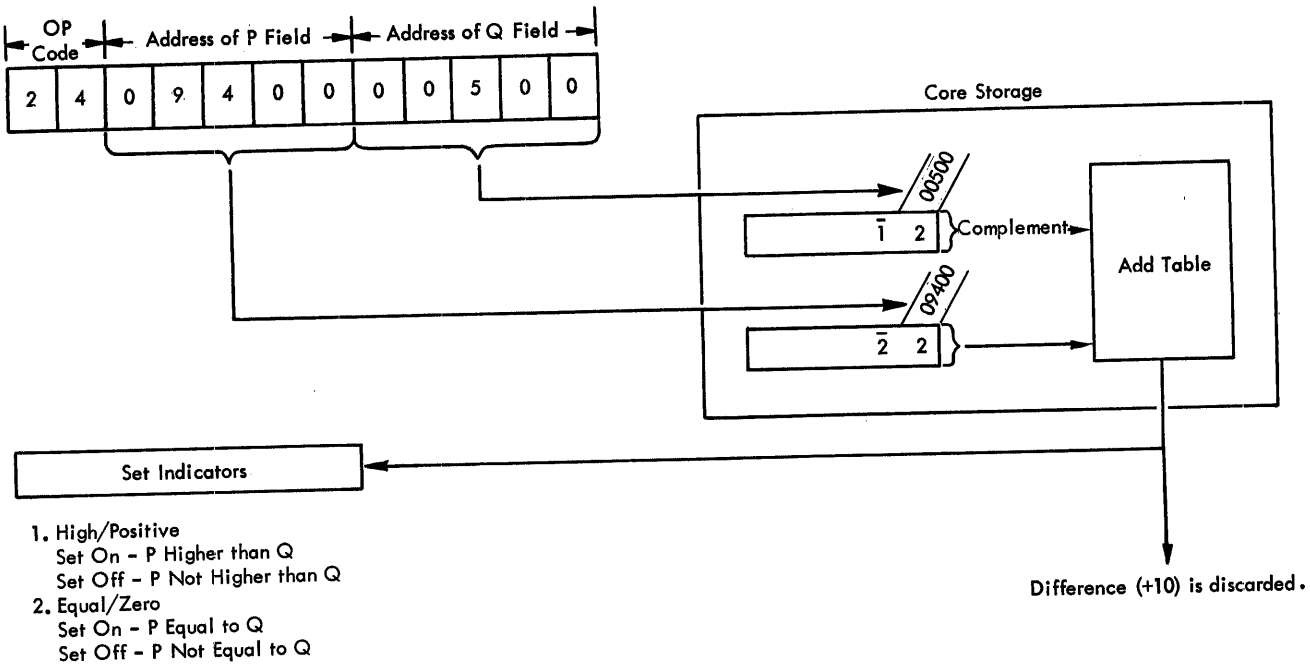


Figure 20. Compare Operation

2. Alphameric Sequence:
   b (blank) .) + $ * − / , (= @ A B C D E F G H I 0̄ (minus zero) J K L M N O P Q R S T U V W X Y Z 0 1 2 3 4 5 6 7 8 9. (Minus 1 through minus 9 occupy the same locations as J through R in the alphameric collating sequence.)

The record mark (‡) and numerical blank cannot be used as data in an arithmetic or compare operation.

Execution Time. When fields with like signs are compared, the execution time varies according to the number of digits in the field at the P address (high-order zeros included).

$$T=160+80\ D_P$$

When fields have unlike signs, the execution time depends on the number of positions compared until a digit other than zero is detected in either field.

$$T=200+80\ D_Z$$

Compare Immediate (CM-14)

Description. Comparison proceeds the same as with the Compare (C-24) instruction, except that the data contained in the Q part of the instruction, rather than the data at the Q address, is compared with the data at the P address.

Execution Time. Same as Compare (C-24).

Branch Instructions

All branch instructions except Branch Back must contain an even-numbered P address because a branch is to the high-order digit (O_0) of an instruction, which must be in an even-numbered location if the operation code is to be interpreted correctly.

Branch instructions may be unconditional or conditional. Unconditional branches are executed as the OP code directs. Conditional branch instructions are performed or not, depending on the condition tested.

Branch (B-49)

Description. This instruction branches unconditionally to the instruction at the P address, which is the next instruction to be executed. The Q part of the Branch instruction is not used.

Execution Time. Execution time is a constant 200 microseconds.

Branch and Transmit (BT-27)

Description. The address of the next instruction in sequence is saved automatically by being stored in an address register (IR-2 listed under SYSTEM CONFIGURA-

TION). The data in the field at the Q address is transmitted to the P address minus one and to successively lower core storage positions. The field at Q remains unchanged. The instruction at the P address is the next one executed. A Branch Back instruction can be used in the new subroutine to return processing to the saved address (this instruction is explained under BRANCH BACK).

In Figure 21, the instruction 27 09400 00500 is executed as follows:

1. The address of the next sequential instruction is saved. This address will be 00012 if the branch and transmit instruction is at 00000.
2. The Q data is transmitted to the P address minus one (09399).
3. A branch to 09400 (the P address of the Branch and Transmit instruction) occurs.

Execution Time. Execution time varies according to the number of digits in the field at the Q address. $T=200+40\ D_Q$

Branch and Transmit Immediate (BTM-17)

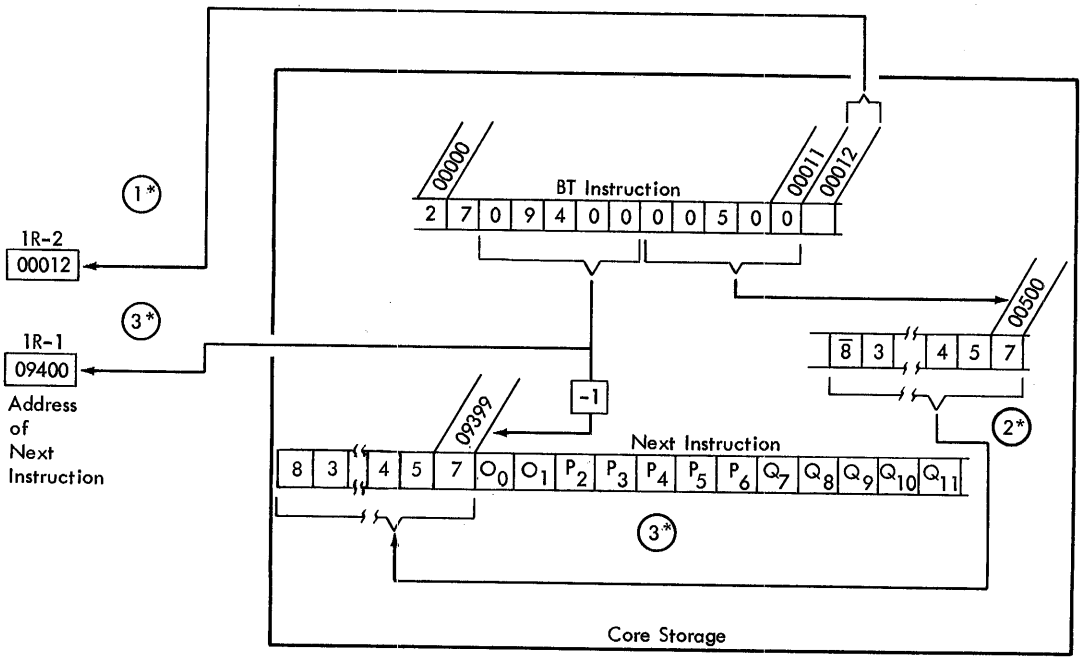
Description. Same as Branch and Transmit (BT-27) except that the digits in the Q part of the instruction are used as Q data.

Execution Time. $T=200+40\ D_Q$.

Branch Back (BB-42)

Description. This instruction causes the computer to branch unconditionally to the instruction at the address saved in IR-2 by the execution of the last Branch and Transmit or Branch and Transmit Immediate instruction, or saved in PR-2 by previous depression of the Save key on the console with the computer in manual mode. (In the manual mode, the computer has terminated all operations and is prepared to accept operator intervention. Manual mode is described under AUTOMATIC AND MANUAL LIGHTS and in Appendix F. Registers IR-2 and PR-2 are listed under SYSTEM CONFIGURATION.)

When the save key is depressed, the address of the next sequential instruction is transferred from IR-1 to PR-1, where it is saved until the save key function is



*Circled numbers refer to these operations:

1. Save address of next sequential instruction in register IR-2.
2. Transmit data in the field at the Q address to the P address minus one.
3. Branch to the P address.

Figure 21. Branch and Transmit Operation

interrogated during the next Branch Back instruction. The Save light on the console is also turned on. The save key function is examined first since it has priority over a Branch and Transmit instruction. If the save key function is active, the save light is turned off and the branch is to the instruction whose address was saved in PR-1.

If the save key function is inactive, the branch is to the address saved in IR-2 when the last Branch and Transmit or Branch and Transmit Immediate instruction was executed. The contents of IR-2 are transferred to IR-1, and IR-2 is cleared. Thus, if a second BB instruction occurs without an intervening BT instruction or a save key depression, a MARS check results. The MARS check function is described under PARITY CHECK INDICATORS. The P and Q addresses of this instruction are not used.

Execution Time. Execution time is a constant 200 microseconds.

Branch on Digit (BD-43)

Description. Branch to the instruction at the P address if the digit at the Q address is not a zero. If the digit at the Q address *is* a zero (either a plus zero or a minus zero), no branch occurs and the next instruction in sequence is executed.

Execution Time. Execution time is 240 microseconds if the branch occurs (digit is not zero) and 200 microseconds if the branch does not occur.

Branch No Flag (BNF-44)

Description. Branch to the instruction at the P address if a flag bit is not present at the Q address.

If a flag bit is present at the Q address, the next instruction in sequence is executed.

Execution Time. Execution time is 240 microseconds if the branch occurs (flag bit not present) and 200 microseconds if the branch does not occur.

Branch No Record Mark (BNR-45)

Description. Branch to the instruction at the P address if a record mark character is not present at the Q address. If a record mark character is present, the next instruction in sequence is executed.

Last-Record Check. The BNR instruction can be used with paper tape to perform a last-record check, similar to the last-card check used with cards.

Data read from paper tape is transferred to core storage in records. Each record is distinguished in storage by a record mark at the end (rightmost position) of the record, resulting from the end-of-line (EL) punch at the end of the tape record. Reading and processing of data continues through the last record, which can be distinguished by adding a second EL punch directly following the EL punch which ended the last record.

A BNR instruction, which tests the first (leftmost) character of each record, follows each read instruction and normally permits data to be processed as directed by the "branched-to" instruction or routine. When the first character read in a record is an EL punch, the read instruction causes this (second) EL punch to be stored in the leftmost position of the input area as a record mark. Processing of data is then stopped by the BNR instruction, which causes the program to proceed sequentially to the instruction or subroutine desired, following the last record on the tape.

Execution Time. Execution time is 240 microseconds if the branch occurs (record mark not present) and 200 microseconds if the branch does not occur.

Branch Indicator (BI-46)

Description. Branch to the instruction at the P address if the indicator or program switch, specified by Q_8 and Q_9 of the instruction, is on.

Indicator Code. The two-digit indicator codes used in Q_8 and Q_9 of this instruction are as follows:

01—Program switch 1
02—Program switch 2
03—Program switch 3
04—Program switch 4
06—Read check indicator
07—Write check indicator
09—Last card indicator
11—High/positive indicator
12—Equal/zero indicator
13—High/positive or equal/zero indicator
14—Overflow indicator
16—Memory buffer register—even check indicator
17—Memory buffer register—odd check indicator
19—Any data check

The positions of the Q part of the instruction, other than Q_8 and Q_9, may have any digital value.

The four Program switches are located and labeled on the console. They are positioned on or off manually.

The read check (code 06), write check (code 07), MBR-even check (code 16), and MBR-odd check (code 17) indicators reflect the results of the check for data parity errors during input/output operation and core storage read-in and readout cycles. They may be interrogated individually by use of the Branch Indicator (BI) instruction, or all at once by the BI instruction and the Any Data Check indicator (code 19). Individual interrogation or manual reset turns off the indicator;

interrogation with Code 19 does not. If the I/O Check switch (described under INPUT/OUTPUT CHECK INDICATORS) is set to PROGRAM, the program should interrogate the associated indicator (06 or 07) immediately after the execution of each read or write instruction. Otherwise, the indicator remains on and thus cannot provide a clear error check indication of the next read or write operation. Card reading or punching cannot occur if 06 or 07, respectively, is on.

The Any Data Check indicator interrogates all four data indicators with a single Branch Indicator instruction but does not turn off any indicator that may be on. A branch occurs if one or more of the four is on. There is no light for Any Data Check on the console.

The high/positive (code 11) and equal/zero (code 12) indicators are turned on or off, depending on the results of arithmetic or compare operations. There is a console light for each indicator. Detecting an overflow during an arithmetic operation turns on the overflow (code 14) indicator, which is turned off by testing the indicator, or by a reset (described under RESET KEY).

The High/Positive or Equal/Zero indicator (code 13) is turned on if either the high/positive indicator or the equal/zero indicator is turned on. The High/Positive or Equal/Zero indicator has no console light.

Except for the any data check, high/positive, and equal/zero indicators, all indicators are turned off by execution of the BI instruction. The status of any program switch, because it is turned on or off manually, remains unchanged. The Any Data Check indicator is turned off only when all four of the data check indicators are off.

Execution Time. Execution time is 200 microseconds if the branch does occur (indicator on) and 160 microseconds if the branch does not occur.

Branch No Indicator (BNI-47)

Description. Same as Branch Indicator (BI-46) except that the branch occurs when an indicator is off.

Execution Time. Execution time is 200 microseconds if the branch occurs (indicator off) and 160 microseconds if the branch does not occur.

Input/Output Instructions

Only one input/output device may be selected at any one time; it does not remain selected after the input or output control function has been executed. An invalid address or control function causes the machine to stop in automatic mode. Automatic mode is indicated by the Automatic light on the console. The computer is in automatic mode, for example, when executing a stored program. Automatic mode is described under AUTOMATIC AND MANUAL LIGHTS, and in Appendix F.

The address of an input/output device is specified in Q_8 and Q_9 of each I/O instruction, as follows:

01—Typewriter
02—Tape Punch
03—Paper Tape Reader
04—Card Punch
05—Card Reader

The 06 Read Check indicator is turned on if a parity error is detected in the 1620 during an input operation.

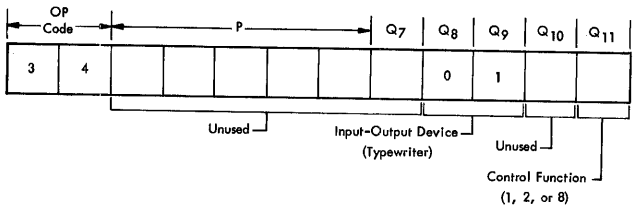
The 07 Write Check indicator is turned on if a parity error is detected in the 1620 during an output operation.

Once a read or write check indicator is turned on, it is not turned off by the reading or writing of subsequent correct characters; testing or resetting of the indicator is required to turn it off. Card I/O operations cannot occur when either indicator is on.

Parity errors that are sensed during typewriter output are indicated by a horizontal bar overprinted across the center of the character.

Control (K-34)

Description. Execute the control function, specified in Q_{11} of the instruction, on the input/output device (typewriter) specified in Q_8 and Q_9 of the instruction.



The control function codes (Q_{11}) apply to the typewriter only and are:

1. Space
2. Return Carriage
8. Tabulate

The P part of the instruction and positions Q_7 and Q_{10} are not used in the execution of the instruction.

Execution Time. Execution time depends upon the control function.

Dump Numerically (DN-35)

Description. Numerical information is transmitted serially to an output device, beginning with the P address and continuing through successively higher addresses. Transmission terminates after the character from the highest-numbered core storage position of the module addressed has been written. This address is

19999, 39999, or 59999, depending on the 20,000-position module specified by the P address. If the output device selected is the tape punch, an end-of-line character is punched in the tape immediately following the last character. If the output device selected is the card punch, punching continues until all 80 columns of the last card have been punched out (see DUMP NUMERICALLY under READ AND PUNCH INSTRUCTIONS for a description of this punchout). Transmission also may be terminated at any time by depressing the Release key on the console.

Except for the $\overline{0}$ punchout on cards, each numerical character, as well as any single record mark character, is written on the output device along with its flag bit (if any). The character in core storage remains unchanged. DN causes only the flag (x) of a $\overline{0}$ to be punched in an output card, so that any subsequent punchout of that character from the card will be as a hyphen.

An alphameric character (represented in core storage as two numerical digits) cannot be written as a single character on the output device by this instruction.

Only Q_8 and Q_9 of the Q part of the instruction are used in the execution of the instruction.

Execution Time. Execution time depends upon the speed of the output device selected and the number of characters written.

Read Numerically (RN-36)

Description. Numerical information from an input device is transmitted serially to the P address and successively higher core storage locations. Transmission continues until terminated by one of the following conditions:

1. Sensing of the end-of-line character when paper tape is being read. At this time, a record mark character is generated automatically by the machine and placed in core storage following the last character read from tape.
2. Depression of the release key on the console when the typewriter is used to enter information. The release key terminates typewriter I/O operations and puts the computer in manual mode (see RELEASE KEY). A record mark character is not generated automatically by the machine. If it is desired to place a record mark in core storage following the last character entered, the Record Mark key on the typewriter must be depressed before depressing the release key on the console.
3. Reading into core storage the 80th character from the card input buffer storage. Buffer storage is considered under IBM 1622 CARD READ-PUNCH UNIT.

Each numerical character from an input device, along with its flag bit (if any), is stored in a single core storage location. A parity check bit (C bit), if needed, is furnished by the machine and stored in the same location.

The – (dash) and J through R characters from the paper tape reader are entered into core storage as numerical digits with flag bits. Numerical blanks from the card reader are entered into core storage as C, 8, and 4 bits; actual blanks (from unpunched card columns) are entered into core storage as plus zeros (C bits). On a Write Numerically (WN) operation, these entries are punched out from core storage as blanks or zeros, respectively. No other alphabetic or special characters (except the record mark) are transmitted correctly to core storage. Only Q_8 and Q_9 of the Q part of the instruction are used.

Refer to the note under READ AND PUNCH INSTRUCTIONS for the handling of certain special characters in a Read Numerically operation.

When the typewriter has been selected, the 1620 stops in automatic mode to await the manual entry of information from the typewriter keyboard. For manual entry procedure, see PROGRAM ALTERATION AND DATA ENTRY.

Execution Time. Execution times for the paper tape reader and typewriter depend upon the speed of the input device selected and the number of characters read. Execution time is 3.4 milliseconds for transferring data between the 1620 and the Card Read-Punch.

Read Alphamerically (RA-37)

Description. Alphameric information from an input device is transmitted serially to the P address and successively higher core storage locations.

The units digit (P_6) of the P part of the instruction must be an odd number; otherwise, the input information is not placed in core storage correctly and parity errors may occur when the input information is read in. This is due to the 2-character transfer operation of core storage. The odd-numbered location must contain the right-hand (numerical) digit of the 2-digit alphameric code. Transmission continues until terminated by one of the following conditions:

1. Sensing of the end-of-line character when paper tape is being read. At this time, an alphameric record mark character (a numerical zero digit followed by a single record mark character) is generated automatically by the machine and placed in core storage following the last character read from tape.
2. Depression of the release key on the console when the typewriter is used to enter information. An alphameric record mark character is not generated

automatically by the machine. If it is desired to place an alphameric record mark in core storage following the last character entered, the record mark key on the typewriter must be depressed before the release key on the console is depressed.

3. Reading into core storage the 80th character from the card input buffer storage. A record mark is not generated in storage.

Information from an input device may be a random mixture of numerical, alphabetic, and special characters. Each character from an input device is stored in core storage as two digits. Flag bits are not transmitted on characters read by an input device; however, flag bits already in the core storage area where the information is read in remain unchanged. A single record mark character read by an input device is stored in core storage as a numerical zero digit (C bit) followed by a single record mark character (coded C-8-2).

The positions of the Q part of the instruction other than Q_8 and Q_9 are not used.

When the typewriter is selected, the 1620 stops in automatic mode to await the manual entry of information from the typewriter keyboard. (For manual entry procedure, see PROGRAM ALTERATION AND DATA ENTRY.)

Execution Time. Same as Read Numerically (RN-36).

Write Numerically (WN-38)

Description. Numerical information in the P address and successively higher core storage locations is transmitted serially to an output device. Transmission continues until terminated by one of the following conditions:

1. Sensing of a record mark character in core storage. The record mark character is not written on the typewriter, but causes an end-of-line character to be punched in paper tape.
2. Depressing the release key on the console. If the release key is not depressed, and no record mark is encountered before the data at the highest-numbered core storage address is written, the machine "loops back" to 00000 and transmission continues.
3. Writing of the 80th position in card output buffer storage.

Each numerical character in core storage, and its flag bit, (if any) is written on an output device, and the character in core storage remains unchanged. No alphameric or special character represented in core storage as two numerical characters can be written on an output device as a single character by this instruction.

The P address must not be the location of a record mark.

Only Q_8 and Q_9 of the Q part of the instruction are used.

Execution Time. Same as Read Numerically (RN-36).

Write Alphamerically (WA-39)

Description. Alphameric information from the P address and successively higher core storage locations is transmitted serially to an output device. The units digit (P_6) of the P part of the instruction must be an odd number; otherwise, the information in core storage is not converted correctly to the single-character output representation. Transmission continues until terminated by one of the following conditions:

1. Sensing of the alphameric record mark. A record mark character is not written on the typewriter but causes an end-of-line character to be punched in the tape.
2. Depression of the release key on the console. If this is done before an alphameric record mark has been encountered in core storage, no record mark character is written, and if the device is the paper tape punch, no end-of-line character is punched. If the release key is not depressed and no alphameric record mark is encountered before the data from the highest-numbered core storage address is written, the machine "loops back" to 00000, and transmission continues.
3. Writing of the 80th position in card output buffer storage.

Each alphameric character in core storage is written on the output device as a single character, and the character in core storage remains unchanged. No flag bit is written on the output device.

Only Q_8 and Q_9 of the Q part of the instruction are used.

The P address must designate an odd core storage position and must not be the location of a record mark.

Execution Time. Same as Read Numerically (RN-36).

Program Control Instructions

Set Flag (SF-32)

Description. A flag bit is placed at the P address, and a C-bit is either added or removed to adjust for correct parity. The original digit at the P address remains unchanged. The Q part of the instruction is not used.

Execution Time. Execution time is a constant 200 microseconds.

Clear Flag (CF-33)

Description. If a flag bit is present at the P address, it is removed and a C-bit is added to adjust for correct parity. The digit at the P address remains unchanged. The Q part of the instruction is not used.

Execution Time. Execution time is a constant 200 microseconds.

Move Flag (MF-71) — Special Feature

This instruction moves a sign or a field definition flag from the core storage location specified by the Q address to the location specified by the P address. For example, the MF instruction moves the sign from the units position of a product to the new units position of the half-adjusted result, or moves a field definition flag to lengthen or shorten a field.

If the location at the Q address is without a flag, the flag at the P address is cleared. If the location at the Q address has a flag, that flag position is cleared and a flag is placed at the P address. Thus, after the instruction is executed, the location specified by the P address reflects the original absence or presence of a flag at the Q address, and the flag position at the Q address is clear. Figure 22 illustrates the movement of a positive sign; Figure 23, the movement of a negative sign; and Figure 24, the lengthening of a field. All three illustrate the simplified programming required.

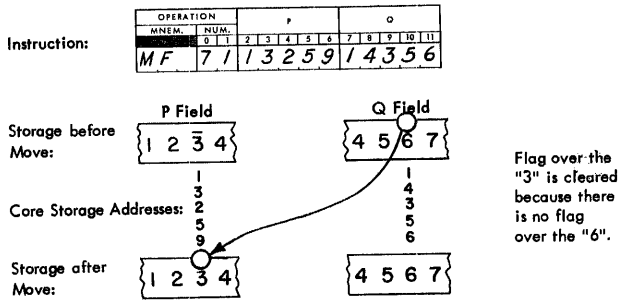


Figure 22. Move Flag, Positive Sign

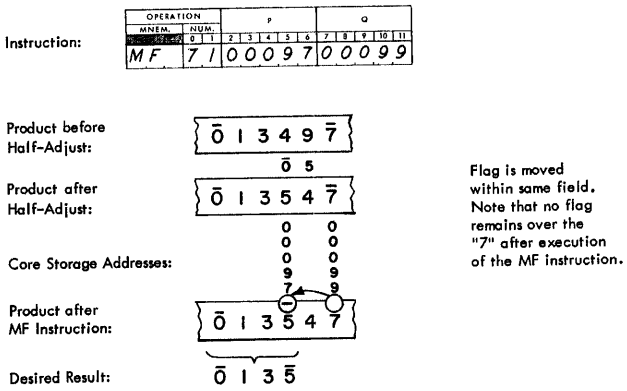


Figure 23. Move Flag, Negative Product

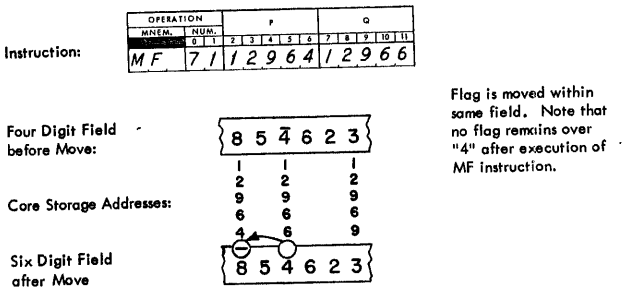


Figure 24. Move Flag, Lengthen Field

Programming is also simplified in the case where only one position of the product is dropped after half-adjustment and the product is either negative or positive. Without the Move Flag instruction, it is necessary to test for the presence of the negative flag and remove it before proper half-adjustment can take place. The simplified programming for such a situation, using the Move Flag instruction, is shown in Figure 25. The first Move Flag instruction saves the sign by placing it over its own O_0, or high-order, position. The second Move Flag instruction returns the sign to the new units position of the half-adjusted result. The flag bit can be stored in any position in which it cannot be mistaken for a field definition flag, a sign flag, or an indirect-address flag.

Execution Time. Execution time is a constant 240 microseconds.

Halt (H-48)

Description. The 1620 stops in manual mode. If the Start key on the console is depressed with the computer in manual mode, the 1620 changes to automatic mode and executes the next instruction in sequence. The P and Q parts of the instruction are not used.

Execution Time. Execution time is a constant 160 microseconds.

No Operation (NOP-41)

Description. Perform no operation and advance to the next instruction in sequence. The P and Q parts of the instruction are not used.

Execution Time. Execution time is a constant 160 microseconds.

Indirect Addressing — Special Feature

Indirect addressing saves program steps and computer time by providing a direct method of address modification. Its primary use is in programs where multiple

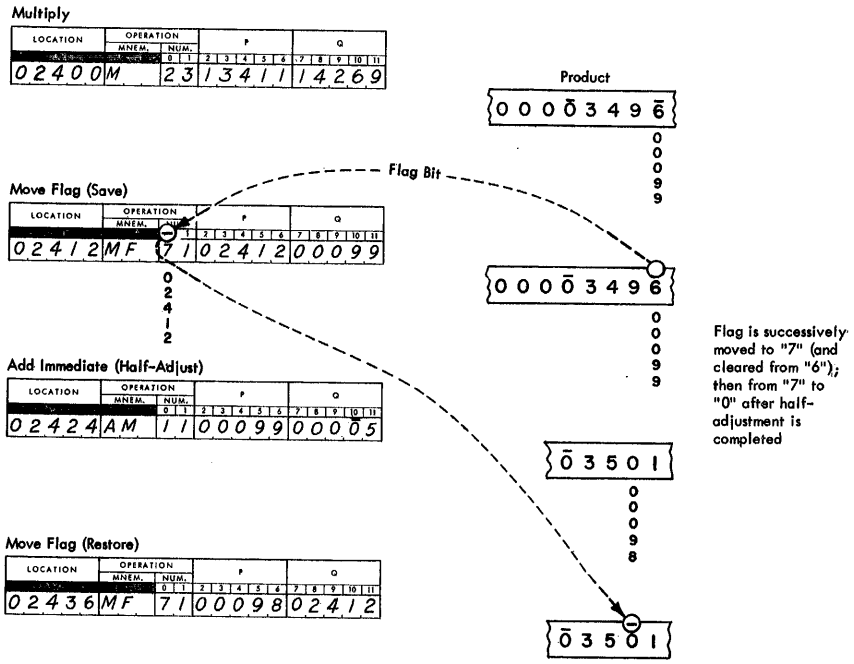


Figure 25. Move Flag to Half-Adjust One Position

instructions have the same address, and this address is to be modified by the program. Indirect addressing may also be used for linking subroutines.

Description

Normally, the P or Q address of an instruction is the location of the data used during execution of the instruction. An indirect address, however, is the address of a second address, and the second address is the location of the data. In effect, the address at the indirect address location is a substitute for the address of the instruction.

The P or Q address of an instruction is indirect when a flag bit is over the units position. Figure 26 shows that (1) the instruction (21 00500̄ 00650) has an indirect P address of 00500, (2) the data at 00500 is 00780, which is used as the P address during execution of the instruction, and (3) the instruction (21 00500̄ 00650)

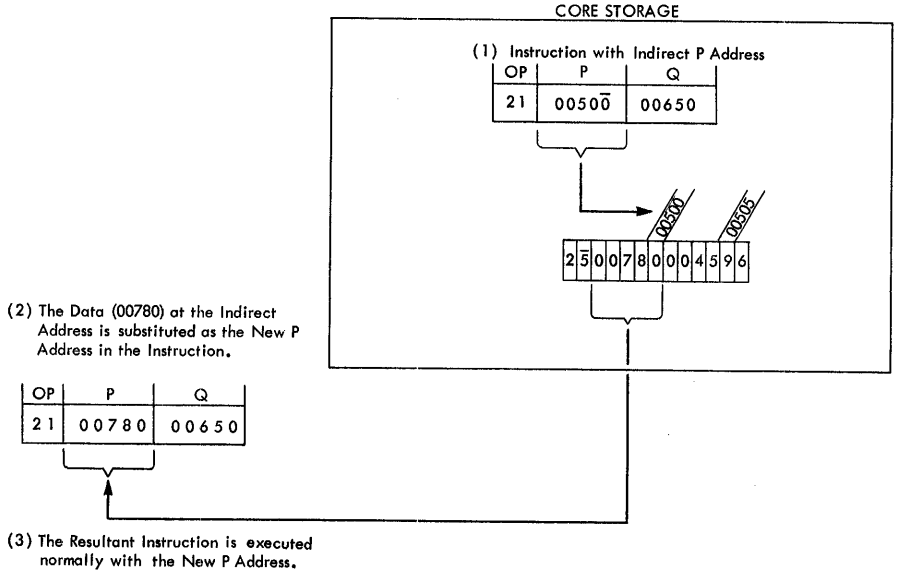


Figure 26. Data Flow, Indirect Addressing

is not altered in core storage; only the internal registers of the 1620 are changed.

The data at the location specified by the indirect address is also an indirect address if a flag bit exists in the units position. This chaining effect continues until a flag bit does not exist in the units position of the address, which is then treated as a direct address.

The data field specified by the indirect address is always five digits in length, and the upper digit does not require a flag bit to define the field. Moreover, the address is always five digits, even though flag bits exist within the field.

Any P or Q address of an instruction that specifies the location of data can be an indirect address. Figure 27 shows those instructions that can be used in indirect addressing. When the P address of an immediate instruction is an indirect address, the Q data cannot be more than six digits in length because of the flag bit over the units position of the P address.

Each address interpreted as an indirect address requires four additional 20-microsecond memory cycles. For example, an instruction with two indirect addresses requires an additional 160 microseconds.

Examples

The add instruction, 21 00500 00650, is shown in Figure 28 with both direct and indirect Q addresses. Line 1 shows direct addressing; the Q data is obtained from

Arithmetic Instructions:	MNEMONIC	CODE	P & Q	P
Add	A	21	X	
Add Immediate	AM	11		X
Subtract	S	22	X	
Subtract Immediate	SM	12		X
Multiply	M	23	X	
Multiply Immediate	MM	13		X
Load Dividend	LD	28	X	
Load Dividend Immediate	LDM	18		X
Divide	D	29	X	
Divide Immediate	DM	19		X
Internal Data Transmission Instructions:				
Transmit Digit	TD	25	X	
Transmit Digit Immediate	TDM	15		X
Transmit Field	TF	26	X	
Transmit Field Immediate	TFM	16		X
Transmit Record	TR	31	X	
Transfer Numerical Strip	TNS	72	X	
Transfer Numerical Fill	TNF	73	X	
Logic (Compare and Branch) Instructions:				
Compare	C	24	X	
Compare Immediate	CM	14		X
Branch	B	49		X
Branch No Flag	BNF	44	X	
Branch No Record Mark	BNR	45	X	
Branch On Digit	BD	43	X	
Branch Indicator	BI	46		X
Branch No Indicator	BNI	47		X
Branch and Transmit	BT	27	X	
Branch and Transmit Immediate	BTM	17		X
Branch Back	BB	42		
Input/Output Instructions:				
Read Numerically	RN	36		X
Write Numerically	WN	38		X
Dump Numerically	DN	35		X
Read Alphamerically	RA	37		X
Write Alphamerically	WA	39		X
Control	K	34		
Program Control Instructions:				
Set Flag	SF	32		X
Clear Flag	CF	33		X
Move Flag	MF	71	X	
Halt	H	48		
No Operation	NOP	41		

Figure 27. Table of Allowable Indirect Address Operation Codes

Instructions	Data at Storage Locations			Resultant Modified Instruction	Actual Q Address Used	Actual Q Data Used
	00650	15225	12500			
①21 00500 00650	15225				00650	15225
②21 00500 00650̄	15225	12500		21 00500 15225	15225	12500
③21 00500 00650̄	15225̄	12500	12345	(a)21 00500 15225̄		
				(b)21 00500 12500	12500	12345

Figure 28. Examples of Indirect Addressing

the Q address. Line 2 shows the Q address as indirect; the Q data is obtained from the address specified by the indirect address. Line 3 shows that the address specified by the indirect address is also indirect; the Q data is obtained from the address specified by the second indirect address.

The data flow diagram for an Add Immediate instruction, 11 00500̄ 00650, is shown in Figure 29. The Q data, ̄000650, is added to the data at the address specified by the indirect P address. The result, ̄1155078, replaces the original P data, ̄1154428, at 09400.

The data flow diagram for a Branch instruction is shown in Figure 30. The first five digits at the indirect address are the address to which the computer branches for its next instruction. This instruction, located beginning at the given address (16000, in Figure 30) and through successively higher-numbered locations, enters the instruction registers to provide the "branched-to" instruction.

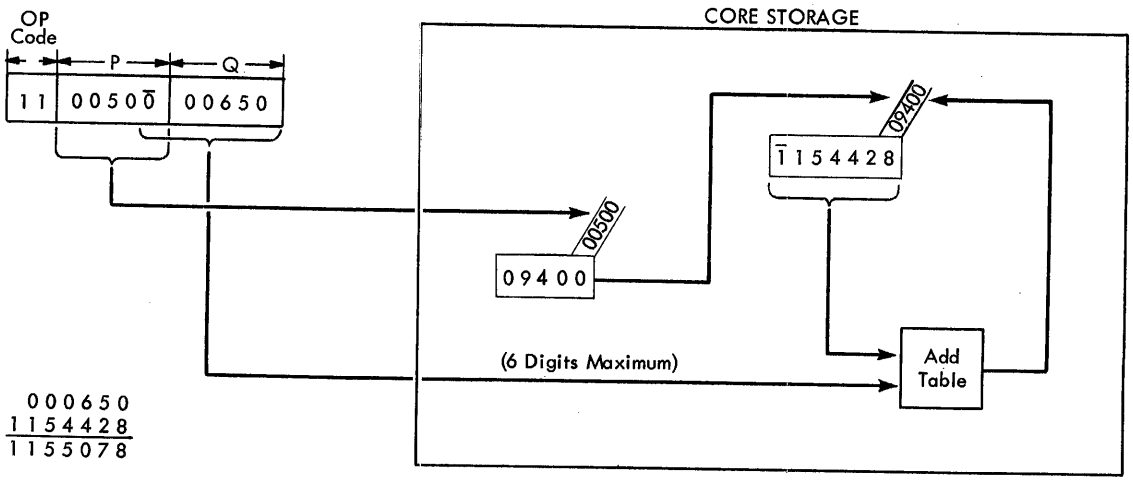


Figure 29. Indirect Addressing, Add Immediate

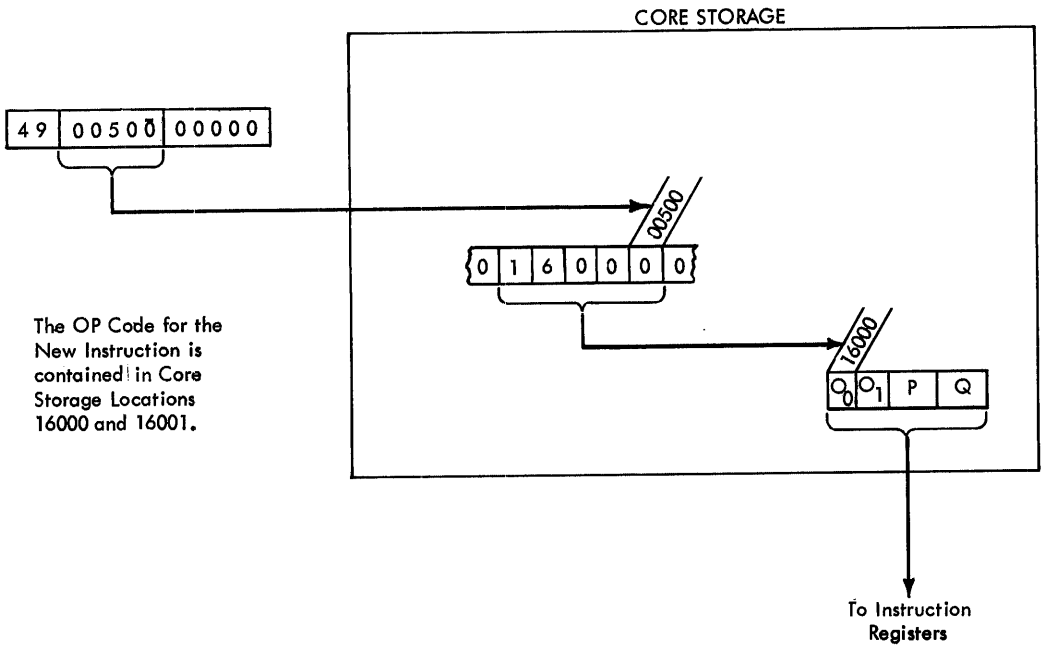


Figure 30. Indirect Addressing, Branch

Typewriter

The typewriter is part of the 1620 console (see Figure 31) and is used for both input and output.

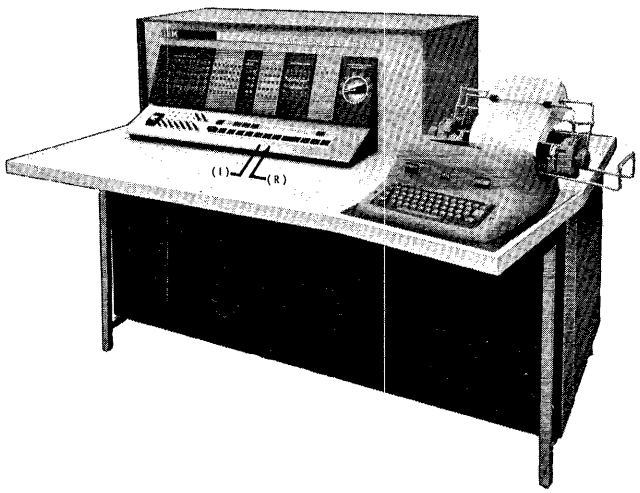


Figure 31. IBM 1620 Console and I/O Typewriter

Input

The typewriter is used to enter both data and instructions directly into core storage. Off-line use is not possible because the keyboard (Figure 32) is locked except when entering data. Depression of the console Insert key (I, Figure 31) unlocks the keyboard and permits data to be entered into core storage, starting at location 00000. Each depression of a typewriter key enters the character into core storage one location higher than the previous character. As many as 100 characters can be entered from the typewriter. After the 100th character is entered, an automatic release is initiated and the machine returns to manual mode.

When less than 100 characters are entered, entry of the last desired character should be followed by depression of the console Release and Start keys, or by depression of the R-S key on the typewriter keyboard. The R-S key combines the release and start functions of the console keys. The R-S symbol is typed as a permanent record that the R-S key has been used.

It should be noted that the decimal period, in either upper shift or lower shift, must be entered with a Read Alphamerically (RA-37) instruction. If entry is made with a Read Numerically (RN-36) instruction, incorrect data will be put into core storage.

Programmed selection of the typewriter (RA or RN instructions) also unlocks the keyboard, leaving the

Figure 32. IBM 1620 I/O Typewriter Keyboard

computer in automatic mode for manual entry of data on the typewriter. Data entry starts at the addressed location (P address) of the instruction and enters core storage at successively higher-order positions until the release key is depressed.

If a record mark is required in core storage following the last character entered, the record mark key on the typewriter must be depressed before depressing the release key on the console.

Depression of the console release key (R, Figure 31) relocks the keyboard and gives the computer an end-of-I/O indication. No record mark is entered into core storage by depression of the release key.

Output

The typewriter prints data from core storage when programmed to do so. When the right-hand margin is reached, the carriage returns automatically, and typing continues until a record mark is sensed or until the release key is depressed. For example, the release key may be used to terminate a Dump Numerically Operation from the typewriter.

Parity Checking

Input data from the typewriter is parity checked before entering core storage. Transmission of a character with incorrect parity turns on the console read check indicator.

Output data from core storage is parity checked as it is transmitted to the typewriter. Transmission of a character with incorrect parity turns on the write check indicator, and a horizontal bar is overprinted across the center of the character. An invalid character with correct parity causes a special symbol character to print (✱).

If a parity error occurs, the input or output operation is completed, and the machine either stops or continues under program control, depending on the position of the console I/O check switch.

Manual Adjustments to Typewriter

The numbers preceding each of the following headings refer to numerals on Figures 32 and 33 designating particular keys, etc.; (1) for example, designates the impression indicator.

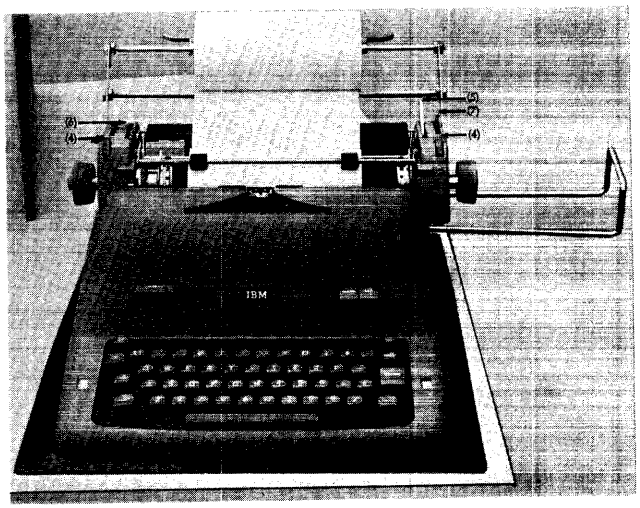


Figure 33. IBM 1620 I/O Typewriter

(1) *Impression Indicator.* The lever under this window can be positioned in settings from 0 to 10, to determine the force with which the type bars strike the paper. The higher the indicator setting, the harder the type bars strike. To test for the correct setting, move the indicator up until the comma and period print distinctly but not heavily. Use a higher setting for multiple copies, but be sure that the multiple copy lever is also correctly set before finally adjusting the impression.

(2) *Tab Clear Lever.* To clear tab stops, tabulate to the point to be cleared and depress the clear lever. To clear all stops at once, position the carriage at the right margin, hold down the clear lever, and return the carriage to the left margin stop.

(3) *Tab Set Lever.* To set tabular stops, move the carriage to the desired position and depress the set lever. Set tab stops only when the indicator pointer (see Figure 33) is in line with a white marking on the front paper scale below it.

(4) *Carriage Release Lever.* Depress the lever on either side to free the carriage and manually move the carriage to the right or left.

(5) *Paper Release Lever.* To free the paper for positioning or quick removal, move this lever forward.

(6) *Line Space Lever.* Moved to position 1, 2, or 3, the line space lever provides for single, double, or triple line spacing, respectively.

(7) *Multiple Copy Control.* This lever moves the platen backward to compensate for the greater thickness of additional copies. As a general rule the lever should be set at "A" for one to three copies and moved back one position for each additional three to five copies. Heavy print at the top of characters shows that the platen is too far back; heavy print at the bottom of characters shows that the platen is too far forward. The slash (/) is a good character to use in checking multiple copy settings.

(8) *Left-Hand Margin Set.* The left margin stop is set as follows:

1. Return the carriage to the present left margin stop.
2. Depress the margin set key.
3. Manually move the carriage as near as possible to the position desired. The back space key and space bar are convenient to use to obtain the exact position desired, with the margin set key depressed.
4. Release the margin set key.

(9) *Right-Hand Margin Set.* The right margin stop is set as follows:

1. Move the carriage to the left until stopped by the right margin stop.
2. Depress the margin set key.
3. Move the carriage right or left to the desired position.
4. Release the margin set key.

Paper Tape and Paper Tape Code

Data is punched and read as holes in a 1-inch-wide chad paper tape (in chad paper tape the holes are completely punched out) at a density of ten characters to the inch. Eight-track paper tape code is used. Seven positions, or tracks, across the width of the tape, are used for the coding of numerical, alphabetic, and special characters. One track is used for EL (End-of-Line) characters. Figure 34 represents a section of paper tape, illustrating the eight tracks and all coded characters (see Appendix D for character codes for all 1620 I/O devices).

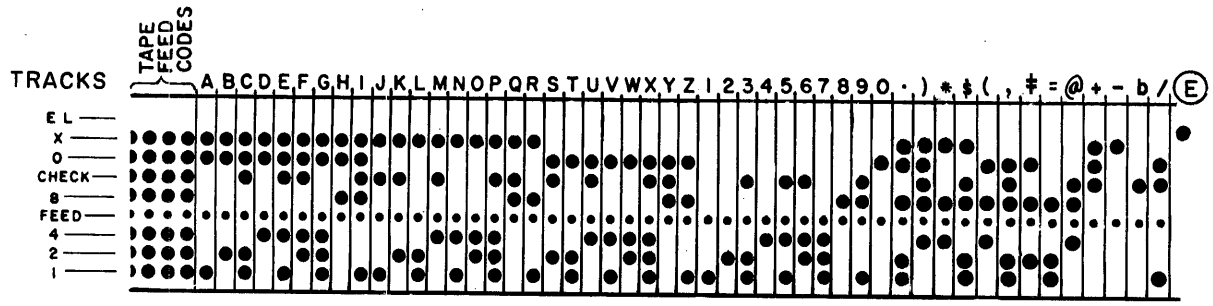


Figure 34. Paper Tape Tracks and Codes

The lower four tracks of the tape (excluding the feed holes) are used to record numerical characters in the BCD mode. For example, a hole in track 1 represents a numerical 1; a hole in track 2 represents a numerical 2; a combination of 1 and 2 punches represents a numerical 3; and so on.

The X and 0 tracks are used in combination with the numerical tracks to record alphabetic and special characters in a manner similar to zone punches in IBM cards. A single X punch reads into core storage as a flag bit (negative zero) in numerical mode.

The check track is used to establish correct parity. As a check that each character is recorded correctly, each column of the tape is punched with an odd number of holes. The EL track is not considered in the parity check.

Tape Splicing

A splice should only be made in non-data portions of paper tape because correct reading cannot be assured at the point of splice. Some methods of splicing chad tape in the data portions are possible, but the reading accuracy is not reliable. The reading mechanism feeds and guides the tape by means of the feed holes, therefore they should not be restricted in any way by splices. Splice specifications are:

1. The total thickness of the tape must not exceed 0.010 inch (nominal paper tape thickness is 0.004 inch).
2. The tape overlap at the splice should be no more than one tape code long (0.100 inch).
3. The splice must be as strong as the tape.
4. The splice must be no wider than the tape.
5. The splice must be free of staples and gummy substances.

The following procedure may be used to splice two lengths of paper tape together.

1. Punch tape feed code into the two ends of the tape to be spliced together.
2. Cut the tapes on approximately a 45° angle.
3. Holding the ends of the tape with the tape feed holes toward you, overlap the tape end in the left hand over the tape end in the right hand by approximately 1/16 inch.
4. Glue in this position with holes aligned, using a quick-setting glue such as IBM tape mucilage, Part No. 221030.

1624 Tape Punch

The tape punch (Figure 35), housed below the tape reader in the IBM 1621, punches data from core storage into paper tape at the rate of 15 characters per second. The characters are sent serially from core storage, starting with the location addressed by an output instruction. Each character is translated to 8-track code before being punched.

Figure 35. IBM 1624 Tape Punch

When a record mark is sensed during the execution of a Write Numerically (WN-38) or Write Alphamerically (WA-39) instruction, an EL hole is punched and the operation stops. A Dump Numerically (DN-35) command causes punching to continue, regardless of record marks, until punching occurs in the highest-numbered core storage address in the 20,000-position module containing the P address from the DN instruction. At this point an EL hole is punched and the operation stops.

If an invalid character is transmitted from core storage and punched, or a valid character is incorrectly punched, the tape feed does not advance. The computer stops in both the automatic and manual mode; the Automatic and Manual lights and the Punch No Feed and Write Check lights on the 1620 console are turned on. Functions of these lights are described under CONSOLE. Program processing can be resumed with the following procedure:

1. Position the 1624 tape feed switch ON.
   a. The feed code (all punches) is punched over the incorrect character.
   b. The punch no feed and write check lights are turned off.
   c. The machine is returned to manual mode only.
2. Depress the start key on the 1620 console.
   a. The original character from storage is again punched. If an incorrect character still persists, the record may be corrected, if desired, before processing continues.
   b. The computer continues processing.

If the 1624 runs out of paper tape, the machine stops in automatic mode and the punch no feed light turns on. The "character correction procedure" outlined is used to resume operation.

When this procedure is used to correct the incorrect punching of a valid character and the character is repunched incorrectly, the SIE key (described under STOP/SIE KEY) can be used as follows to determine the cause of the incorrect punching:

Use the SIE key to execute one instruction at a time. When the write check light is turned on, observe the MBR display (described under REGISTER DISPLAY INDICATORS AND SWITCHES) to determine if the character is valid. If it is, notify an IBM Customer Engineer.

A transient condition may cause the write check light (but not the punch no feed light) to come on even though a valid character has been correctly punched. Should this occur, briefly turn on the tape feed switch to turn off the write check light, then depress the console start key and proceed with the program.

The punch registration (proper hole spacing) can be verified by the use of a standard paper tape gauge. Off-line punching equipment can be checked in the same manner.

Loading the Tape Punch

Place the roll of unpunched tape on the turntable and thread as shown in Figure 35. The tape retainer (F) must be rotated to the left by pushing back on its extended left edge. This also moves the tape lever (D) forward to facilitate threading. An unwound section of tape is then threaded as follows:

1. Through tape guide (A).
2. Inside tape guide (B).
3. In front of tape tension guide (C).
4. In back of tape lever (D).
5. Between the punching mechanism and the punch guide block (E), which can be seen in front of the tape.
6. Between the guides on the tape retainer (F). With the end of the tape held to the left, the tape retainer (F) is returned to normal position, which causes the pins on the feed roll to pierce through the blank tape. The tape lever simultaneously returns to normal position with the top guide above the tape.

The tape feed key is used to repetitively punch automatic feed punches and to provide a leader section of paper tape. The approximately 60" of leader needed for threading paper tape on the 1621 can be obtained from the 1624 in 40 seconds. The leader is threaded into the 1624 takeup reel so that the top edge of the tape is at the outside of the reel.

1621 Paper Tape Reader

The paper tape reader reads coded alphameric characters from 8-track paper tape at the rate of 150 characters per second. The characters are photoelectronically sensed, converted to binary-coded-decimal (BCD), and placed in core storage. If a parity error is sensed, the read check indicator (console panel) is turned on. The computer remains in automatic mode and continues to read until the end-of-record indication (a hole in the EL channel) is reached. Whether the computer stops depends upon the setting of the I/O check switch. The end-of-record signal causes a record mark to be placed in core storage as the rightmost digit of the input record. NOTE: The read head area, including the lens, should be cleaned of paper dust with a lint-free cloth or tissue at least once each operating shift. Grease or oil on paper tape, as from hand lotions, renders it transparent, and may result in tape read errors.

Loading the Paper Tape Reader

Paper tape can be handled in three forms. The procedure for loading each one varies slightly. The names of machine components used in the following descriptions of loading procedures are given on Figure 36.

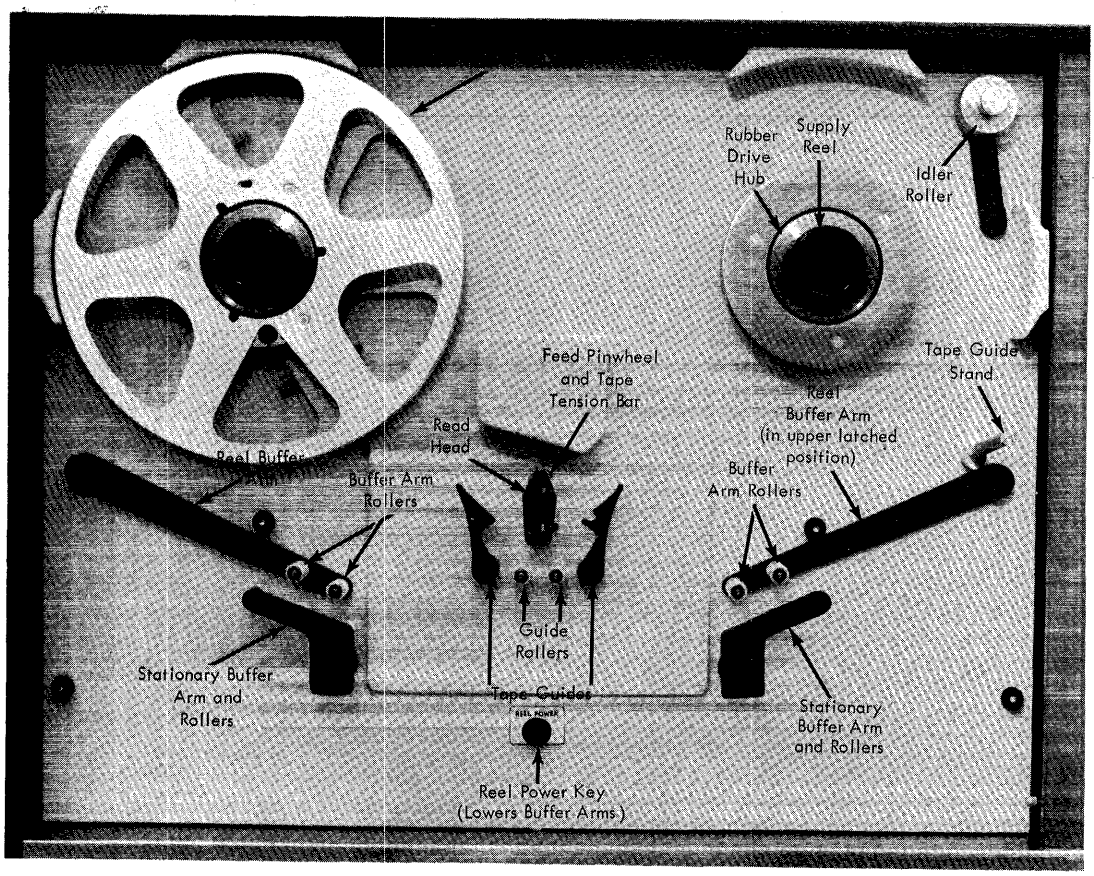


Figure 36. IBM 1621 Tape Loading Area

STRIP FORM

Small strips of tape may be loaded directly onto the read head, as shown in Figure 37, with the following procedure:

1. Position the reel strip switch to STRIP.
2. Open the tape guides, form an inverted U (∩) with the leading 12 inches of paper tape, and install the tape around the read head with sufficient tension to keep the runout and tape tension contacts closed. Start on the takeup reel side of the read head. Run a finger up over the tape on top of the read head, smoothing the tape down with a firm, moderate pressure so that the tape tension bar is slightly depressed and the right side of the feed pinwheel engages the tape feed holes. Use care not to tear the feed holes. The tape feed holes must mesh with both sides of the pinwheel.
3. Close the tape guides.

CENTER ROLL FEED

The center roll feed eliminates the necessity for rewinding paper tape rolls to expose the starting end of the tape on the outside of the tape roll. Figure 38 shows that tape is supplied from the inside of the center roll feed, to the supply reel, around the read head, and onto the takeup reel.

The procedure for loading paper tape from the center roll feed is as follows:

1. Position the reel strip switch to REEL.
2. Place the reel buffer arms in the upper latched positions.
3. Open the tape guides and form an inverted U (∩) with the center section of the first eight feet of paper tape. Wrap the paper tape around the read head with sufficient tension to keep the runout and tape tension contacts closed. Start on the takeup reel side of the read head. Run a finger up over the tape on top of the read head, smoothing the tape down with a firm, moderate pressure so that the tape tension bar is slightly depressed and the right side of the feed pinwheel engages the tape feed holes. Use care not to tear the feed holes. The tape feed holes must mesh with both sides of the pinwheel.
4. Close the tape guides.

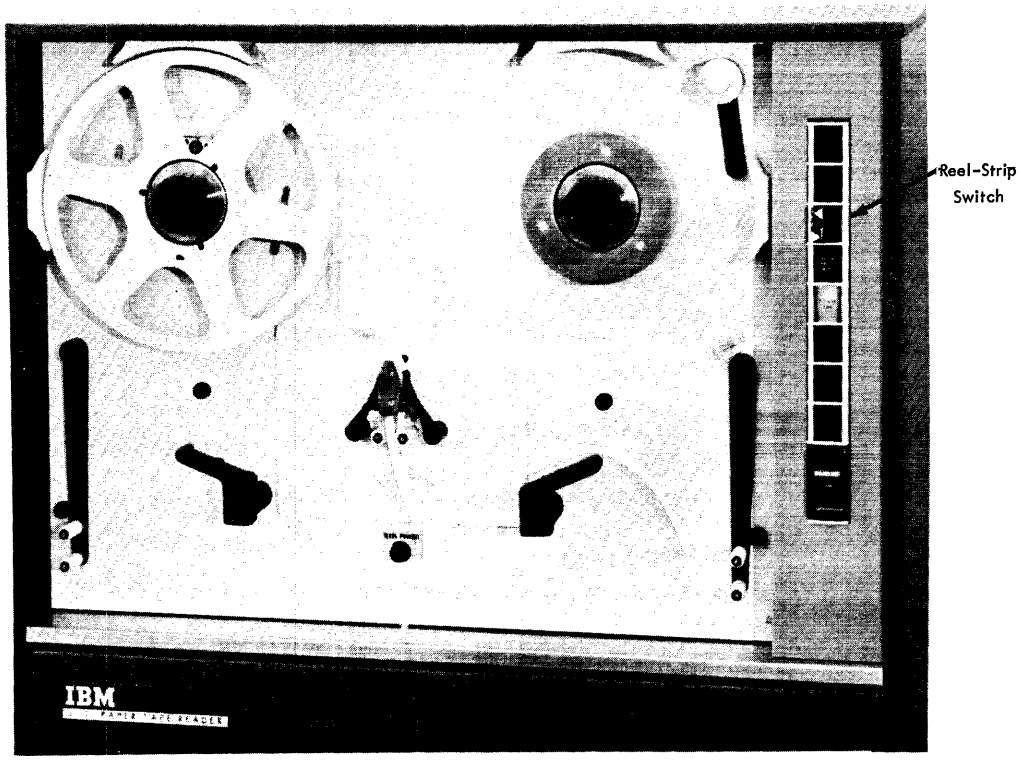


Figure 37. Strip Tape Loaded

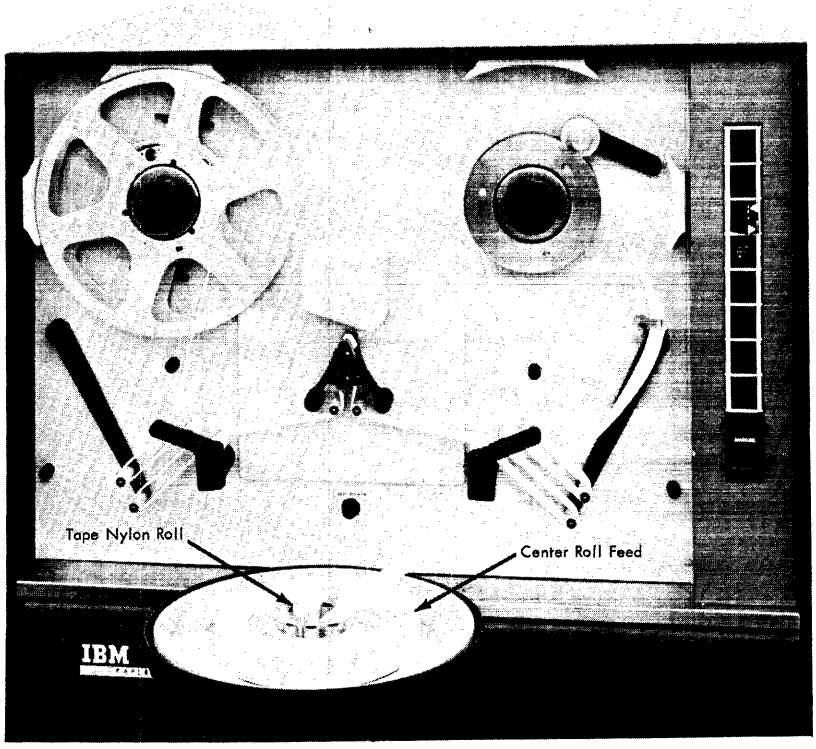


Figure 38. Center Roll Feed Loaded

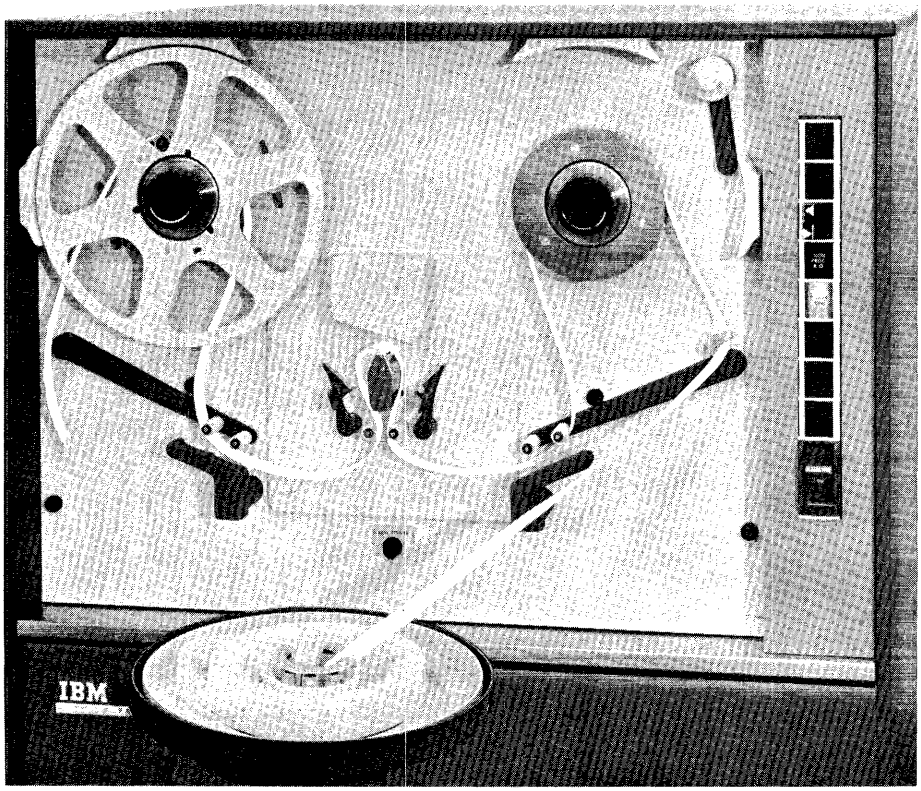


Figure 39. Threading Tape from Center Roll Feed

5. Thread the leading section of paper tape under the guide roller, between the stationary buffer rollers and buffer arm rollers, and onto the takeup reel, as shown in Figure 39.
6. Thread the paper tape from the right side of the read head, under the guide roller, between the stationary buffer rollers and buffer arm rollers, over the supply reel (the rubber drive hub must be installed), around the tape guide stand, and around the tape reel nylon roll.
7. Lower the idler roller onto the supply reel.
8. Lower the buffer arms gently.
9. Depress the reel power key. The buffer arms should swing down to a neutral position, applying tension to the paper tape.

NOTE: The roll of paper tape must be positioned centrally, or evenly, around the center rollers to prevent excessive vibration during reading.

REEL

A reel of paper tape may be read on the 1621 by removing the rubber drive hub from the supply reel and mounting the reel of tape in its place. The tape is threaded from the right-hand side of the reel, directly to the stationary buffer rollers, and to the takeup reel as described in the Center Roll Feed section. Figure 40 shows a reel of tape threaded on the 1621.

Operating Switches and Lights

The following switches and lights are used in the operation of the 1621:

Power Switch. With this switch ON, all necessary power for operation of the 1621 is supplied by the 1620.

Reel Strip Switch. In reel mode, tape is fed from the supply reel and to the left, onto the takeup reel. In strip mode, short pieces of tape may be read without reel operation.

Reel Power Key. Depression of this key operates the supply and takeup reels to position the paper tape for reading and placing the machine in ready status.

Nonprocess Runout Key. Depression of this key causes paper tape to feed. Ready status is terminated and all data transfer is blocked until all paper tape has passed. Paper tape must be reloaded and the reel power key depressed before the machine can be returned to ready status.

Power On Light. Light on indicates that power is supplied from the 1620.

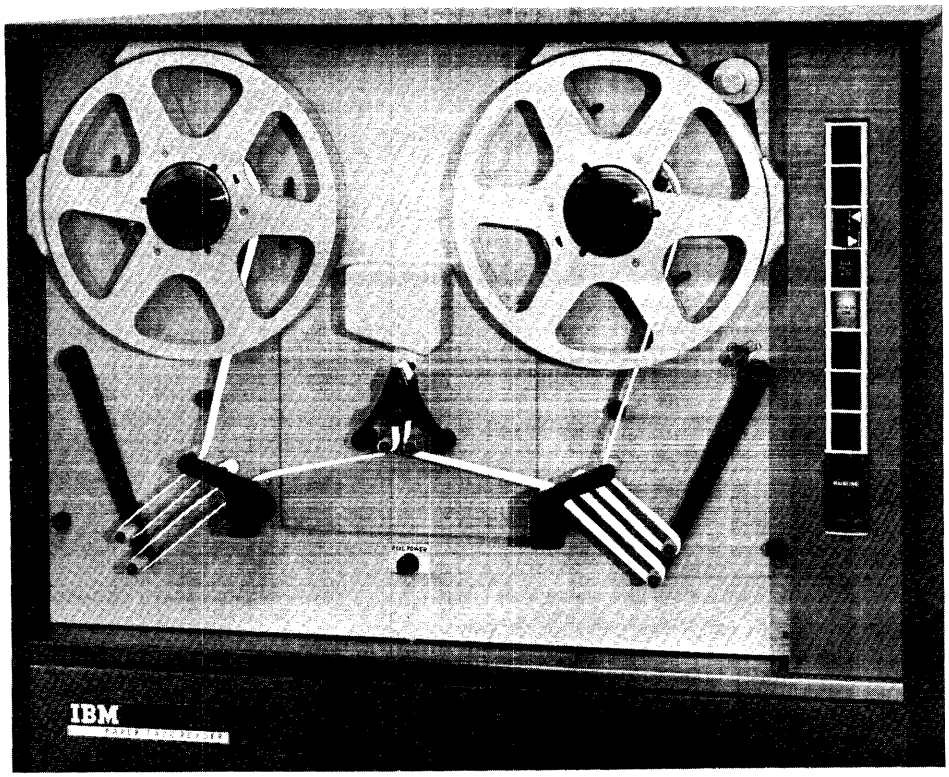


Figure 40. Paper Tape Reel Loaded

IBM 1622 Card Read-Punch Unit

The IBM 1622 Card Read-Punch (Figure 41) provides punched card input and output for the IBM 1620 Data Processing System. The reader and punch feeds are separate and functionally independent, with individual switches, lights, checking circuits, buffer storage, and instruction codes. Under program control, up to 250 cards per minute can be read and 125 punched. Reading, punching, and processing can occur simultaneously because of individual buffer storage. Buffer storage data is transferred in 3.4 milliseconds; the remainder of the reader and punch feed cycle time is available for processing (see Figure 42).

Additional advantages of card input/output on the 1620 Data Processing System are as follows:

Compatibility with other punched card equipment.

Input and output data can be altered without reproducing an entire file of data, as required with paper tape.

FORTRAN and Symbolic Programming System (SPS) assembly decks can be read directly into the 1620 without conversion to paper tape.

As shown in Figure 43, cards are fed from the read hopper on the right and the punch hopper on the left. Each hopper has a capacity of 1,200 cards. Both feeds have misfeeding and jam detection, and a select and nonselect stacker. The 1,000-card-capacity stackers are of the radial type: the cards are stacked on end to permit their removal while the 1622 is running.

Figure 41. IBM 1622 Card Read-Punch Unit

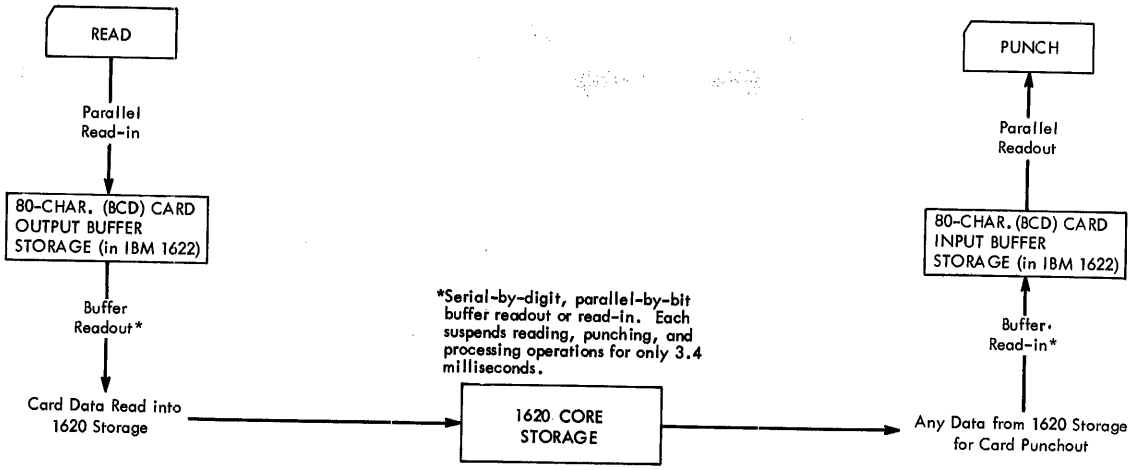


Figure 42. Data Flow Path through Card Buffer Storage

CARD READER AND PUNCH

If either the read or punch feed is not used for approximately one minute, the drive motor for that feed is turned off to reduce noise and wear. However, the 1622 is still in ready status and will respond to a read or write command.

Card Read

Cards are fed 9-edge first, face down, past two reading stations: check and read. Input buffer storage is initially loaded with 80 columns of card data during the Start key or Load key run-in operation. Thereafter, each card feed cycle is under program control. Data flow during card reader operation, shown in Figure 44, is as follows:

1. A read command causes a data transfer from input buffer storage to core storage. The transferred data is parity-checked in the 1620; if parity is correct, a card feed cycle follows immediately to reload buffer storage. Buffer storage data flow is shown in Figure 42.

   If a parity error occurs during data transfer to the 1620, the 1620 console read check light and 06 indicator (see BRANCH INDICATOR description) are turned on. The section on INDICATORS explains how the 06 indicator may be used to branch to an error-handling subroutine.
2. Following a correct transfer of data, a card is fed and new data is read into buffer storage.
3. The new data is compared at the read station against stored data previously read at the check station.

   An unequal comparison between check and read stations, or a 1622 parity error stops the

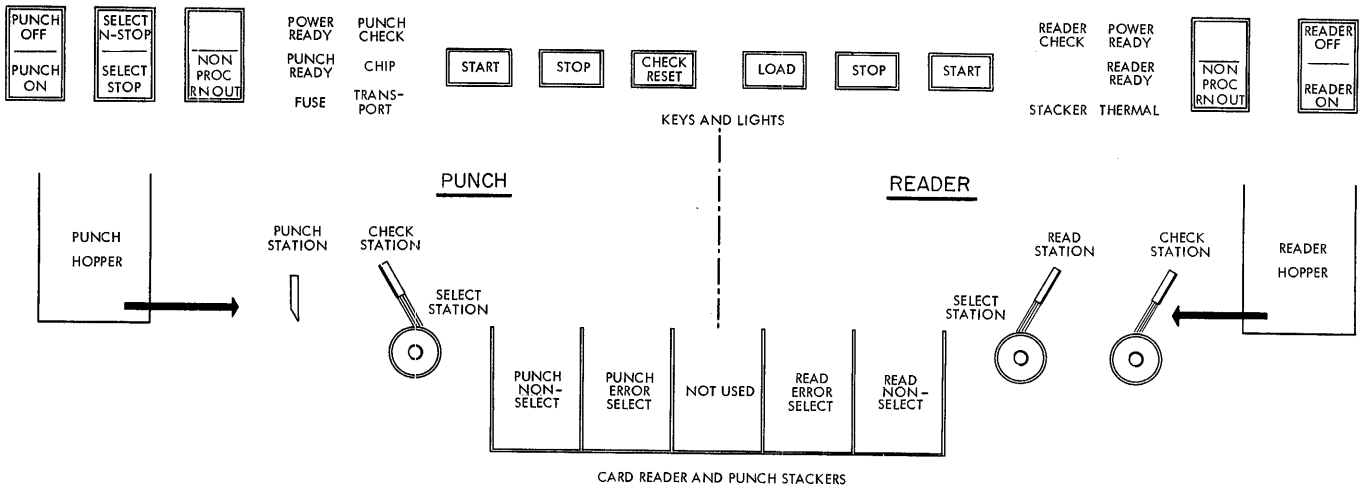


Figure 43. Schematic Diagram of 1622 Keys, Lights, and Card Feeds

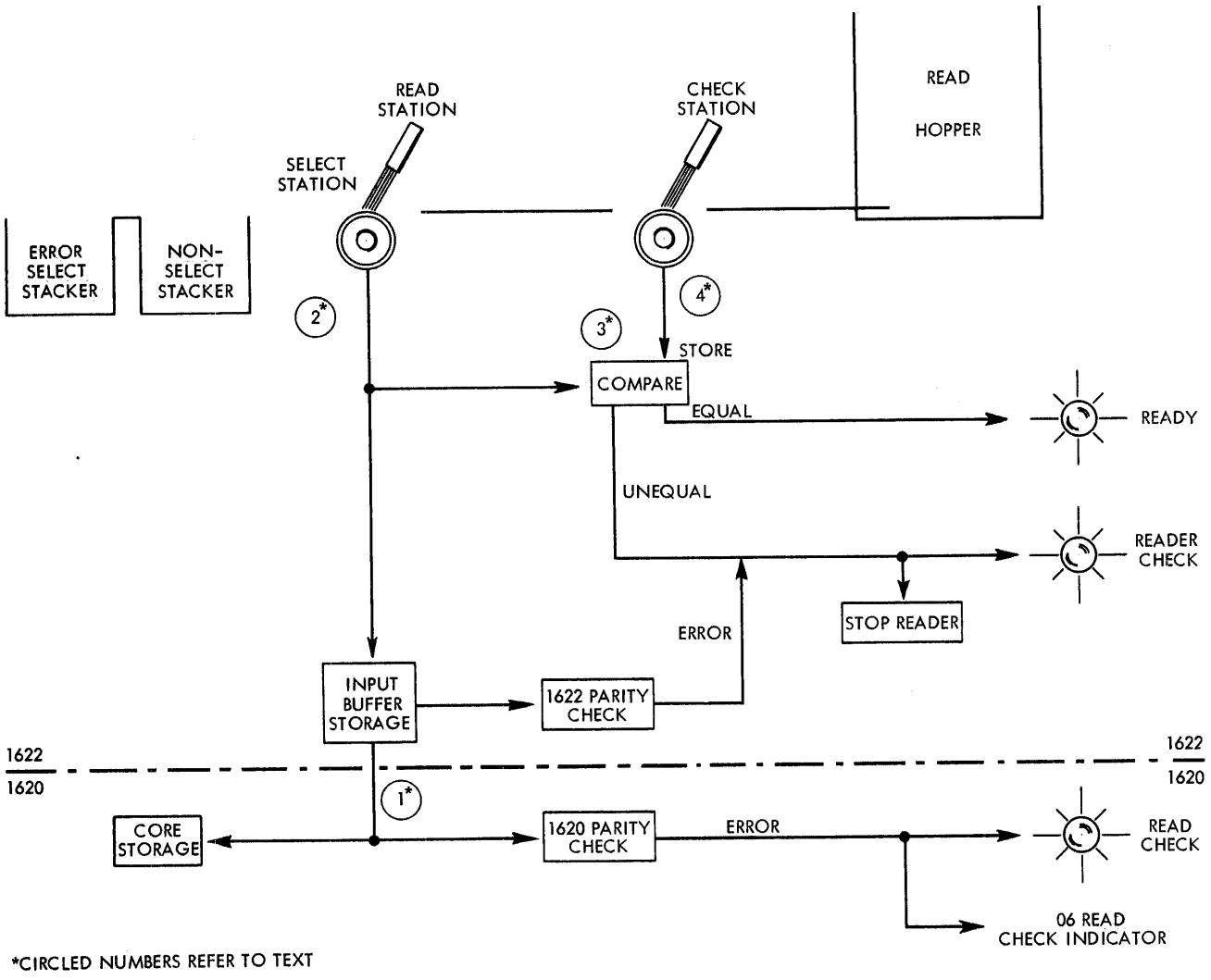


Figure 44. Read Operation Data Flow

reader, turns on the 1622 Reader Check light (described under OPERATOR KEYS AND LIGHTS, for CARD READER), and terminates ready status. No data transfer to the 1620 is permitted. The reader check light thus cannot be on simultaneously with the read check light.

4. At the same time that new data is read at the read station (step 2), data on the card following is read at the check station and stored for comparison on the next card feed cycle.

Card Punch

Cards are fed 12-edge first, face down, past the punch and check stations. Data flow during the card punch operation, shown in Figure 45, is as follows:

1. A write command causes a data transfer from core storage to output buffer storage. The transferred data is parity-checked in the 1620; if parity is correct, a card feed cycle follows immediately to punch the data into the card from buffer storage (see Figure 42). The data is also parity-checked in the 1622 as it is punched into the card.

   If a parity error occurs during data transfer, the 1620 console write check light and 07 indicator are turned on. This light and indicator are functionally described under BRANCH INDICATOR. The Indicators section explains how the 07 indicator may be used to branch to an error-handling subroutine.

2. Following a correct transfer, the data is stored for comparing (step 3), punched into the card, and parity-checked.

   If a 1622 parity error occurs, a cycle delay is initiated and the punch is stopped one card feed

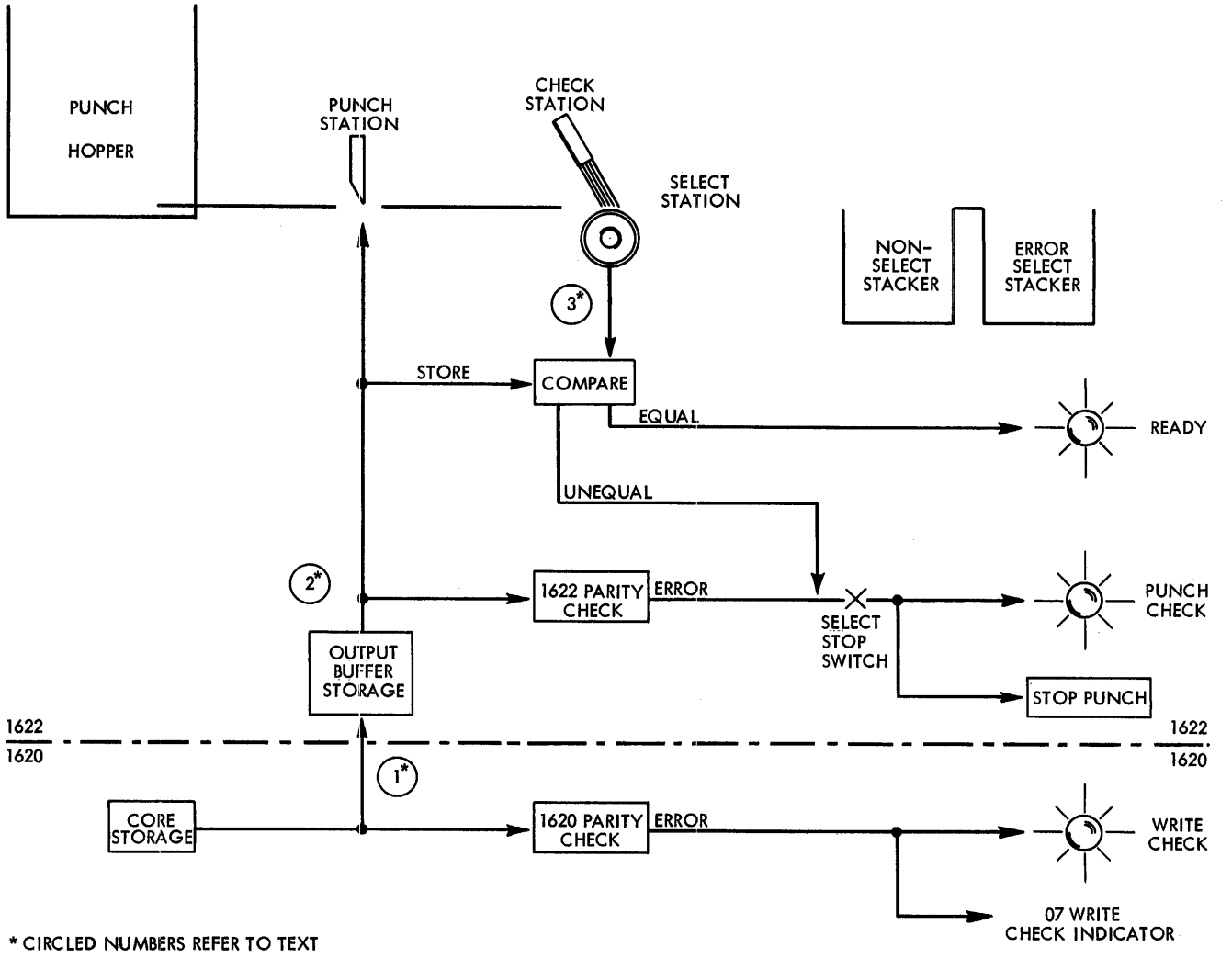
cycle after punching the incorrect data (Select Stop switch set to STOP; functions of this switch are described under OPERATOR KEYS AND LIGHTS, for PUNCH). The 1622 Punch Check light is turned on and ready status is terminated.

3. The card punched in step 2 is read at the check station one cycle later and compared with the data stored in step 2. An unequal comparison has the same effect as a 1622 parity error: the punch is stopped after the card cycle on which the unequal comparison occurred (no cycle delay), the punch check light is turned on, and ready status is terminated.

Read and Punch Instructions

For numerical and alphameric instructions, data is transferred between 1620 core storage and 1622 buffer storage in blocks of 80 digits, irrespective of record marks. A full 80 columns of card data are always transferred. When the highest-numbered core storage address of a 20,000-digit module falls within the transfer, core storage locations in the next highest module are used or a "loop-back" to location 00000 occurs. Buffer storage-to-core storage and core storage-to-buffer storage transfers require 3.4 milliseconds, whether numerical or alphameric transfers are involved. Figure 42 shows the data transfer that is explained in the following paragraphs.

Read Numerically (RN-36). This instruction causes the 80 columns of data to be transferred from input buffer storage to 80 successively higher-numbered positions in core storage, starting at the P address. Figure 46 shows the read instruction format. The card reader code (code for selection of card reader as the input device) is 05 and must be in Q_8 and Q_9.



* CIRCLED NUMBERS REFER TO TEXT

Figure 45. Punch Operation Data Flow

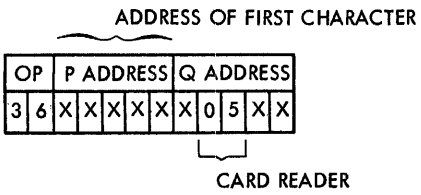


Figure 46. Instruction Format for Read Numerically Command

NOTE: When alphameric data, including all alphabetic, numerical, and special characters (see Figure 47), is transferred numerically from 1622 input buffer storage to 1620 core storage, the read check occurs only as a result of a parity error during transfer. Notice from the chart in Figure 47 that several characters (equals sign, period, dollar sign, and comma) besides the record mark have an 8, 2 representation in core storage. Because these characters are treated as record marks on Transmit Record operations, the programmer must be aware of the card data and format if he is to use the RN instruction successfully with these characters. If data format information is not available, the Read Alphamerically instruction (RA-37) must be used.

Read Alphamerically (RA-37). This instruction causes the 80 columns of data to be transferred from input buffer storage to 160 successively higher-numbered positions in core storage, starting at the P address.

Character in Buffer	Bits Entered into Core Storage by Read Numerically Instruction C	F	8	4	2	1
A	0	0	0	0	0	1
B	0	0	0	0	1	0
C	1	0	0	0	1	1
D	0	0	0	1	0	0
E	1	0	0	1	0	1
F	1	0	0	1	1	0
G	0	0	0	1	1	1
H	0	0	1	0	0	0
I	1	0	1	0	0	1
J	1	1	0	0	0	1
K	1	1	0	0	1	0
L	0	1	0	0	1	1
M	1	1	0	1	0	0
N	0	1	0	1	0	1
O	0	1	0	1	1	0
P	1	1	0	1	1	1
Q	1	1	1	0	0	0
R	0	1	1	0	0	1
S	0	0	0	0	1	0
T	1	0	0	0	1	1
U	0	0	0	1	0	0
V	1	0	0	1	0	1
W	1	0	0	1	1	0
X	0	0	0	1	1	1
Y	0	0	1	0	0	0
Z	1	0	1	0	0	1

Character in Buffer	Bits Entered into Core Storage by Read Numerically Instruction C	F	8	4	2	1
0	1	0	0	0	0	0
1	0	0	0	0	0	1
2	0	0	0	0	1	0
3	1	0	0	0	1	1
4	0	0	0	1	0	0
5	1	0	0	1	0	1
6	1	0	0	1	1	0
7	0	0	0	1	1	1
8	0	0	1	0	0	0
9	1	0	1	0	0	1
/	0	0	0	0	0	1
. (period)	0	0	1	0	1	1
, (comma)	0	0	1	0	1	1
@	1	0	1	1	0	0
(	1	0	1	1	0	0
)	1	0	1	1	0	0
=	0	0	1	0	1	1
*	0	1	1	1	0	0
-	0	1	0	0	0	0
+	1	0	0	0	0	0
Card I/O Only 11, 0	0	1	0	0	0	0
Card I/O Only 12, 0	1	0	0	0	0	0
≠	1	0	1	0	1	0
$	1	1	1	0	1	1
Blank	1	0	0	0	0	0

NOTE: $, =, ., and , (Dollar sign, equal sign, period, and comma) behave as record marks on a Transmit Record Instruction.

Figure 47. Core Storage Coding of Alphameric Characters after Read Numerically Instruction

The units digit of the P address must be an odd number. Numerical data stored in the two-digit alphabetic mode must be converted by programming to single-digit numerical data before being used in arithmetic commands. The Transfer Numerical Strip instruction is used for this conversion.

Write Numerically (WN-38). This instruction causes the data in 80 successively higher-numbered positions of core storage, starting at the P address, to be transferred to output buffer storage.

Write Alphamerically (WA-39). This instruction causes the data in 160 positions to be transferred in the same manner. The units digit of the P address must be an odd number. Figure 48 shows the write command format. Note that the punch code (code for selection of the card punch as the output device) is 04 and must be in Q_8 and Q_9.

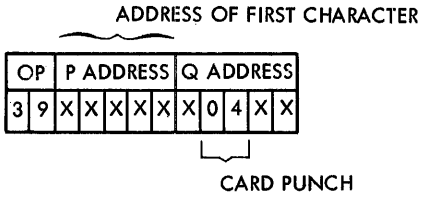


Figure 48. Instruction Format for Write Alphamerically Command

Dump Numerically (DN-35). This instruction causes successively higher-numbered positions of core storage, starting at the P address, to be punched in cards when Q_8 and Q_9 carry the digits 0 and 4, respectively. For example, the instruction 35 00000 00400 causes the first 20,000 digits in core storage to be punched into 250 cards. If a starting address chosen is not an exact multiple of 80 columns to the end of a 20,000-digit storage module, data from the last card will overflow to the "low" end of the next module, or "wrap around" to address 00000 and successively higher-numbered addresses as required to completely punch out all 80 columns of the last card.

Indicators

Internal 1620 indicators can be interrogated by programming to determine whether a branch to a subroutine should occur.

1620 Read and Write Check Indicators. An 06 Read Check or 07 Write Check indicator is turned on whenever a parity check occurs during a buffer storage-to-core storage or core storage-to-buffer storage transfer of data. Either indicator may then be used to stop the 1620 after completion of the instruction (console I/O switch set to STOP) or to branch to an error-handling subroutine. The subroutine should be written to again transfer the data several times when errors are repeated. The subroutine should also return the program to the main routine after a successful transfer, or halt the machine after several unsuccessful transfers.

NOTE: These indicators (06 and 07) must be turned off (by a BI or BNI instruction) after every read or write operation. No card reading or punching can occur, using the 1622, if the respective indicator (06 or 07) is on.

Last Card Indicator. The 09 Last Card Indicator is turned on whenever the data from the last card is correctly transferred from input buffer storage to core storage. The indicator may then be used to branch to an end-of-job routine. The indicator is turned off by program interrogation or by depression of the 1620 reset key.

Card Coding

Record marks and blank columns are processed in the following manner:

Record Marks. Card columns punched 0-2-8 are read into core storage as record marks, with either a Read Numerically or a Read Alphamerically instruction. Record marks are handled as data on both input and output, and do not end the transfer of data.

Blank Card Columns. Because blank columns are read numerically into core storage as zeros, they cannot be punched in the card as blanks with a Write Numerically instruction. Therefore, cards specially punched 8-4 in all columns must be read into core storage to be used when blank columns are required in output cards. The 8-4 punches are stored as C, 8, and 4 bits, and are decoded as numerical blanks when transferred to output buffer storage.

By programming, C, 8, and 4 bits (format blanks) are read into, or transmitted to, the output area of core storage. The output data is then transmitted into this 80-column record of blanks, leaving only the blanks required. The write instruction follows.

Operator Keys and Lights

The card reader and card punch have separate keys and lights (see Figures 41 and 43).

CARD READER

Reader On/Off Switch. The reader on/off switch is used to supply power to the reader and to turn on the Power Ready light. The 1620 Power On/Off switch must be on to make the 1622 reader on/off switch active.

Load Key. The load key causes data from the first card to be checked, read into buffer storage, and automatically transferred in numerical mode to core storage positions 00000-00079. Upon completion of this data transfer, another card feed cycle occurs which loads buffer storage with data from the second card. The 1620 then simulates release and program start at 00000. The instructions from the first card, now in 00000-00079, can be used to continue loading the program or to begin processing. The 1620 must be reset and in manual mode to make the load key operate correctly.

Start Key. The start key is used (1) to run in cards, which are then placed under program control (data from the first card is checked and loaded in input buffer storage); (2) to set up a runout condition, which permits programmed reading of the cards remaining in the feed when the hopper has become empty; and (3) to restore ready status after the reader has been stopped by either the stop key, an empty hopper, an error, a misfeed, or a transport jam.

Stop Key. The stop key is used to stop the read feed at the end of the card cycle in progress and/or to remove the reader from ready status. Data that is entered into buffer storage during the read cycle in progress is transferred to core storage. The computer continues processing until the next read card command causes a reader-no-feed stop.

Nonprocess Runout Key. The nonprocess runout key is used to run cards out of the read feed after a reader check error, or after the stop key has been used to stop the reader. The cards are run out into the read select stacker without a buffer storage-to-core storage transfer. The reader check light and check circuits are turned off. Cards must be removed from the hopper to make the nonprocess runout key active.

Reader Ready Light. The reader ready light is turned on to indicate that the first card has been loaded into buffer storage with the start key, without a reader check error. It remains on until the occurrence of: a depression of the stop key, a reader check error, a transport jam, a misfeed, or an empty hopper.

Reader Check Light. The reader check light is turned on by an unequal comparison between the read and check stations and by incorrect parity detected in buffer storage during card read. When there is an unequal comparison, the reader is stopped, ready status is terminated, and the buffer storage data just read cannot be transferred to core storage on the next read command.

1620 Console Read Check Light. The 1620 read check (06) indicator and console read check light are turned on by a 1620 parity error during a buffer storage-to-core storage transfer.

1620 Console Reader No Feed Light. The console reader no feed light is turned on each time the reader is selected by a read command. The light remains on, if for any reason the reader is not in ready status and the read command therefore cannot be executed. It appears to be on almost continuously when the time between read calls is less than 240 ms, indicating that processing time is available.

CARD PUNCH

Punch On/Off Switch. The punch on/off switch is used to supply power to the punch and to turn on the power ready light. The 1620 power on/off switch must be on to make the 1622 punch on/off switch active.

Start Key. The start key is used to feed cards to the punch station initially or after an error and nonprocess runout, and to re-establish ready status after an empty hopper, a misfeed, a transport jam, or a stop key depression.

Stop Key. The stop key is used to stop the punch feed at the end of the card cycle in progress and/or to remove the punch from ready status.

Check Reset. The check reset key is used to reset error circuits and turn off the punch check light. A start key or nonprocess runout key depression follows, as described under ERROR RESTART PROCEDURES.

Select N-Stop — Select Stop Switch. This switch is used to control the stopping of the punch when error cards are selected into the punch error select stacker. With the switch set to STOP, the punch feed stops with the error card in the select stacker.

Nonprocess Runout Key. Following a punch check error, depression of the nonprocess runout key resets the error circuits and causes the punched card that is between the punch station and the punch check station, if it is in error, to follow the error card into the select stacker. If this card is in error, the punch check light is turned on again. The next two (blank) cards go into the nonselect pocket. These cards should be removed before further processing.

This key is also used to run out and check the last punched card of a job. Cards must be removed from the hopper to make the nonprocess runout key operative.

Punch Ready Light. The punch ready light is used to indicate that the 1622 has a card in punch position and will respond to a write command from the 1620. The ready light is turned off by a punch check error, an empty hopper, a full chip box, a stop key depression, a transport jam, or a misfeed.

Punch Check Light. The punch check light is turned on when there is an unequal comparison between the data punched and the data read (one card feed cycle later, at the check station), or when a 1622 parity error occurs during punching (select stop switch set to STOP). The machine stops, and ready status is terminated.

Chip Light. The chip light is turned on to indicate that the chip box should be emptied.

1620 Console Write Check Light. The 1620 write check (07) indicator and console light are turned on by a parity error during a core storage-to-buffer storage transfer. The 07 indicator may be used, by programming, to transfer data several times, and to halt if a correct transfer cannot be obtained.

1620 Console Punch No Feed Light. The console punch no feed light is turned on each time the punch is selected by a write command. The light remains on until the punch unit is ready and executes the command. Normally, no light is seen if commands are farther apart than 480 milliseconds. The write command cannot be executed until the punch is in ready status.

CARD READER/PUNCH LIGHTS

The stacker, transport, fuse, and thermal lights are common to both the read and punch feeds and are used as follows:

Stacker Light. The stacker light is turned on when a stacker is full. Both feeds are stopped temporarily and removed from ready status. The ready light remains on. Operation resumes automatically after the stacker is emptied.

Transport Light. The transport light is turned on when a card jam has occurred in either the read or punch feed or above any stacker. When this occurs, both feeds are stopped and removed from ready status. Both start keys must be depressed to resume operation after the condition is corrected.

Fuse Light. The fuse light turns on to indicate a blown fuse.

Thermal Light. The thermal light is turned on if the internal temperature of the 1622 becomes excessive. After several minutes delay, the 1620 console reset key may be depressed to turn off the thermal light. If depression of the reset key turns off the thermal light, the 1620 power switch must be turned off and then on again. Operation may be resumed after the power ready light is turned on.

Error Restart Procedure

READER CHECK ERROR

Cause: Unequal comparison between the read and check stations, or a buffer storage parity error. The reader stops with the error card in the select stacker (last card).

Indicators: 1622 reader check light ON.
1622 ready light OFF.

Restart Procedure:

1. Remove cards from the read hopper.
2. Depress the nonprocess runout key.
3. Remove the last three cards from the select stacker.
4. Place these three cards in front of the cards removed from the hopper and replace the deck in the hopper.
5. Depress the start key. The card that caused the error is read into buffer storage again and if an equal comparison is obtained, the interlocked read instruction is executed and processing continues.

1620 READ CHECK ERROR

Cause: Parity error in the 1620 during data transfer from 1622 buffer storage to 1620 core storage. Reader stops with an "error" card (card associated with the error) in the nonselect stacker (last card). Under program control a reread can be initiated as often as desired by the programmer.

Indicators: 1620 read check light ON.
1622 reader ready light ON.
06 read check indicator ON.

Restart Procedure:

1. Remove cards from the read hopper.
2. Depress the nonprocess runout key.
3. Remove the last card from the nonselect stacker and the last two cards from the select stacker.
4. Place these three cards in front of the cards removed from the hopper. The "error" card from the nonselect stacker is to be read in first.
5. Insert a branch to the address of the instruction that transfers the error card data from input buffer storage to core storage.
6. Depress the start key.

PUNCH CHECK ERROR

Cause: Unequal comparison between the data punched and the data read (one card feed cycle later, at check station), or a 1622 parity error while punching data from buffer storage. If the select stop switch is set to STOP, the punch stops with the error card in the select stacker.

Indicators: 1622 punch check light ON.
1622 punch ready light OFF.

Restart Procedure:

To restart without (1) immediate manual correction of the error card or (2) reprocessing of the error card:

1. Depress the check reset key.
2. Depress the start key. Processing continues from the point at which the program stopped.

For manual correction of the error card:

1. Remove the last (error) card from the punch (error) select stacker and correct the error card. Place the corrected card behind those in the punch nonselect stacker.
2. Depress the check reset key.
3. Depress the start key. The interlocked write com-

mand for the second card following the error card can now be executed.

For reprocessing of the error card, *when one card is punched out for each card read:*

1. Remove cards from both hoppers.
2. Depress both nonprocess runout keys.
3. Remove the last two cards from the punch error select stacker and the last two (blank) cards from the punch nonselect stacker. Also, remove the last two cards from the read nonselect and the last two cards from the read select stacker.
4. Mark or destroy the two punched cards removed from the punch select stacker. Place four cards from the read stackers (nonselect in front of select) ahead of those removed from the read hopper. Place blank cards in the punch hopper.
5. Insert a branch to the address of the instruction that begins the reprocessing of the error card.
6. Depress both start keys.

1620 WRITE CHECK ERROR

Cause: 1620 parity error. The error has not been punched into a card.

Indicators: 1620 write check light ON.
07 write check indicator ON.

Restart Procedure:
A typeout of the core storage positions that were transferred indicates whether the data is correct in core storage. If the data is incorrect in core storage, reread the card or cards from which this data originated.

Loading procedures and a utility load routine are included in the IBM *Data Processing Bulletin, Program Writing and Testing* (Form J26-5547).

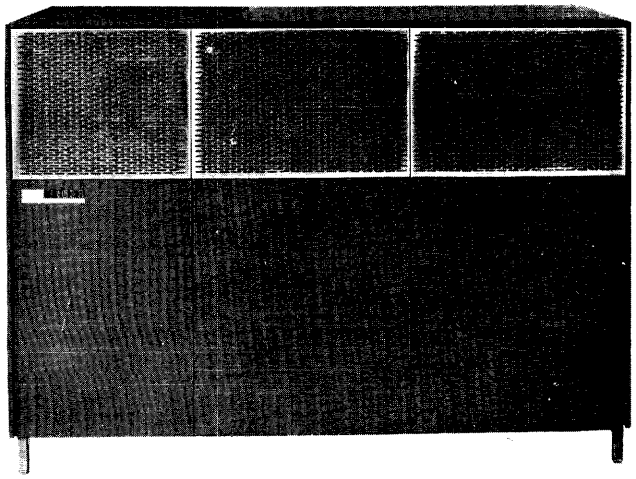


Figure 49. IBM 1623 Core Storage Unit

IBM 1623 Core Storage Unit

The 1623 Core Storage Unit (Figure 49) provides the additional program and data storage needed for applications that require more than 20,000 storage positions.

Description

Two 20,000-position modules of core storage are available. These expand the storage capacity of the 1620 from 20,000 positions to 40,000 or 60,000 positions, depending on the number of modules installed.

The programming and operating characteristics of the 1620 are not changed by additional core storage. Addressing is from 00000 to 39999 with the addition of one module and from 00000 to 59999 with both modules. The resulting storage is cyclical in that 00000 follows the largest allowable address when incrementing. Conversely, the largest allowable address is the next lower address below 00000. No storage reservations for table arithmetic are required in additional core storage.

1623 Core Storage Cabinet

The 1623 cabinet is approximately the same in size as the 1620 console without the table top and can contain one or both 20,000-position modules. There are no additional console controls. A power cord and signal cables 20 feet in length are provided for connecting the additional core storage cabinet to the 1620.

Checking

An invalid address is detected in MAR, and the MAR check indicator is turned on when the digits listed below are in the high-order position (P_2 or Q_7) of either the P or Q address:

Core Storage	*Error Digits in P_2 or Q_7*
One Module (00000-39999)	4, 5, 6, 7, 8, 9
Both Modules (00000-59999)	6, 7, 8, 9

In other words, a 40,000-position storage unit cannot have a valid address greater than 39999; thus, any digit greater than 3 (4, for example, for address 40000) is invalid. Similarly, an address of 60000 or greater is invalid for a 60,000-position unit, for which the largest allowable address is 59999.

Dump Numerically (DN-35) Instruction

Digits from core storage are transmitted to the output device selected by Q_8 and Q_9 of the instruction, starting at the P address and continuing through the highest-numbered position of the module addressed. For example, the instruction 35 19500 00100 causes the typeout of positions 19500 through 19999 inclusive, regardless of the number of modules installed. See DUMP NUMERICALLY, under READ AND PUNCH INSTRUCTIONS, for a description of a possible extension of the punchout, when the card punch is selected as the output device.

The console (Figure 50) is an integral part of the central processing unit and provides for manual or automatic control of the system. The console lights, keys, switches, and typewriter are used to:

Instruct the machine manually.
Display machine and program status indicators.
Display the contents of core storage and registers.
Place data and instructions in core storage.
Alter the contents of core storage.
Alter machine functions.

Keys, Indicator Displays, and Switches

Small incandescent lights are used to represent the on and off conditions of internal check indicators.

Seven console switches (four Program and three Machine Check switches) are provided to externally control the execution of machine functions for which two alternative logic paths are provided. One or the other of the paths is selected, depending upon the setting of the appropriate switch.

Machine Check Indicators and Switches

Machine operation may be altered by the condition of a machine check indicator and an associated check switch (Figure 51). An indicator that is turned on causes the computer to halt if the associated check switch is set to STOP, or to continue in automatic mode if the associated check switch is set to PROGRAM. Regardless of the check switch setting, the associated check light provides a visual sign of the indicator status.

Depression of the reset key turns all check indicators and lights off. Parity, I/O, and Overflow check indicators are provided.

PARITY CHECK INDICATORS

Internal data flow errors are recorded by the parity check indicators: MBR-E and MBR-O. Normally, the parity check switch is set to STOP.

MBR-E (Memory Buffer Register-Even) Check Light. This light and indicator are turned on when the digit in the even address portion of the MBR has a parity error. An error stops the machine immediately, if the parity check switch is set to STOP.

MBR-O (Memory Buffer Register-Odd) Check Light. This light and indicator are turned on when the digit in the odd address portion of the MBR has a parity error. An error halts the machine immediately if the parity check switch is set to STOP.

MARS (Memory Address Register Storage) Check Light. This light turns on when a digit in MARS has a parity error. This is an unconditional machine stop, and is not affected by the position of the parity check switch.

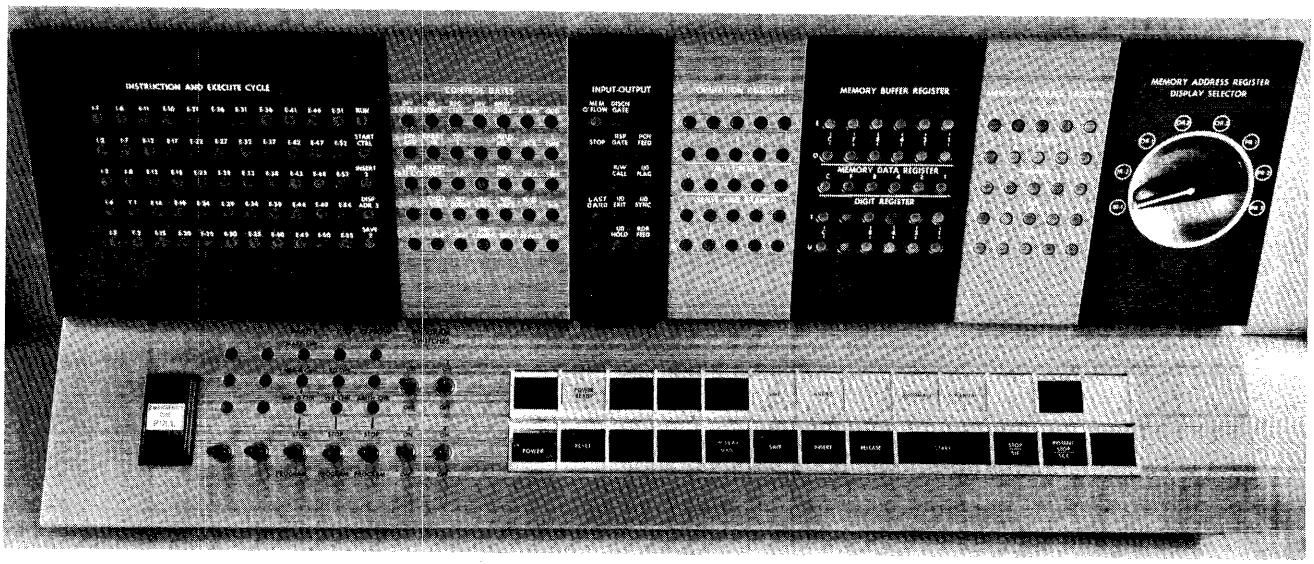


Figure 50. IBM 1620 Console

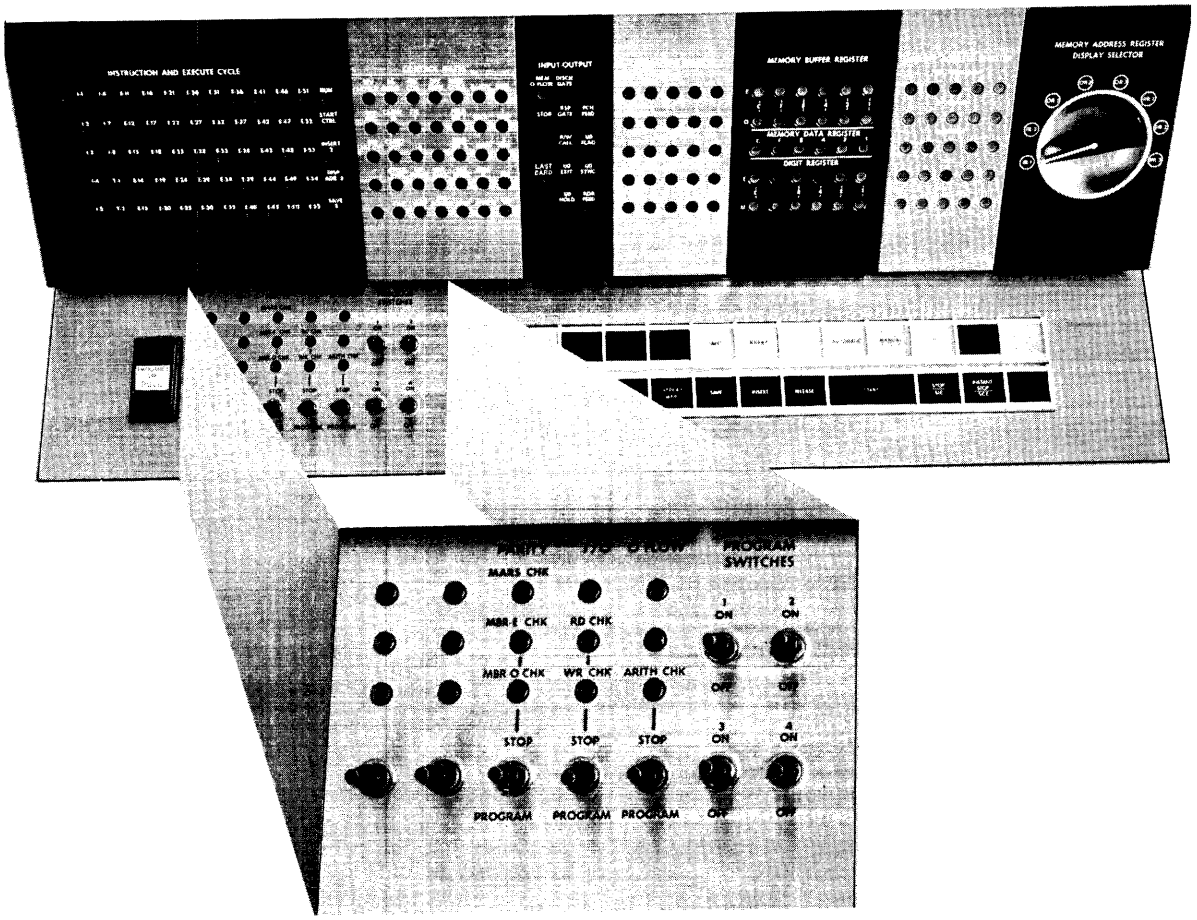


Figure 51. Indicator Displays and Switches

INPUT/OUTPUT (I/O) CHECK INDICATORS

RD CHK (Read Check) Light. This light and indicator are turned on when an input character with a parity error is detected, prior to conversion of input data to BCD code. An error halts the machine after the input operation is complete, if the I/O check switch is set to STOP.

WR CHK (Write Check) Light. This light and indicator are turned on when an output character with an even number of bits is detected during conversion of output data from BCD to output code. The effect on machine operation as a result of detection of this parity error varies, depending on the output device selected, as follows:

Typewriter: Error detection halts the 1620 after the output operation is complete, if the I/O check switch is set to STOP.

Card Punch: The card is not punched; error detection halts the 1620 at the end of an 80-character transfer to output buffer storage, if the I/O check switch is set to STOP.

Tape Punch: Error detection halts the 1620 as soon as the character is punched and prevents the tape feed from advancing, regardless of the check switch setting.

OVERFLOW ARITH CHK (ARITHMETIC CHECK) INDICATOR

An overflow that occurs as a result of an add, subtract, divide, or compare operation turns on the overflow check indicator and light. When the overflow check switch is set to STOP, and the overflow check indicator is turned on, the computer halts at the end of the instruction being executed. If the start key is depressed, the overflow check indicator remains on, and the computer continues to execute instructions in the automatic mode, until another overflow occurs.

When the overflow check switch is set to PROGRAM, and the overflow check indicator is turned on, the machine continues to operate in the automatic mode. The indicator can be interrogated and turned off by the program.

CONSOLE PROGRAM SWITCHES

There are four modifier switches in this group. They are labeled PROGRAM SWITCHES on the console and are numbered 1 through 4. A branch occurs when a switch specified by a Branch Indicator (BI-46) instruction is set to ON.

When the switch specified is set to OFF, no branch occurs from the BI instruction, and the next instruction in sequence is executed.

When a Branch No Indicator (BNI-47) instruction is used to interrogate one of these switches, the branch occurs when the switch is set to OFF.

REGISTER DISPLAY INDICATORS AND SWITCHES

The console panel displays the contents of registers by means of small incandescent lights, used to represent the bits present in each digit of a register (Figure 52). Each light, representing a particular bit position, is on only when its corresponding bit is present in the digit displayed.

Memory Buffer Register (MBR). The two stored digits affected by a core storage address (previously explained under TWO-CHARACTER TRANSFER) are displayed in the MBR. When the core storage location addressed for display is an even-numbered address, the digit at this location is placed in the MBR display in the E (even line); the O (odd) line contains the digit in the next *higher*-numbered location. If the core storage location addressed for display is an odd-numbered address, the digit at this location is placed in the MBR display on the O line; the E line contains the digit in the next *lower*-numbered location. When the machine is in alphabetic mode, the complete two-digit representation of an alphameric character may be viewed at one time.

Memory Data Register (MDR). One line of six indicator lights displays the bit configuration of each digit in core storage as it is read out. These digits can be seen on single cycle operation, using the SCE key (described under CONTROL SWITCHES, KEYS, AND SIGNAL

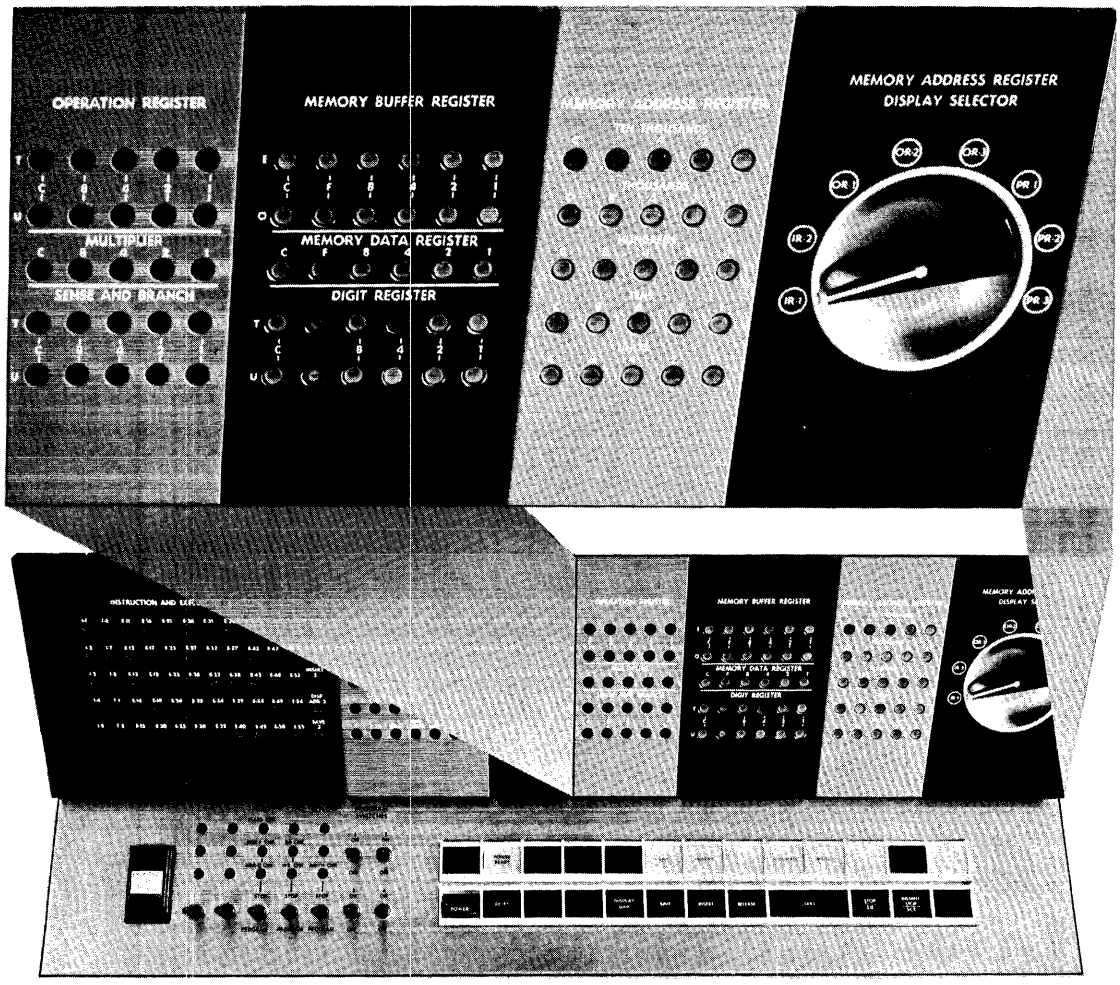


Figure 52. Register Display Indicators

LIGHTS). The digit displayed in the MDR display is duplicated in the MBR-even or MBR-odd display, depending on whether the digit read out is located at an even or an odd numbered core storage position.

Operation (OP) Register. Two lines of five lights each display the bit configuration of the two digits representing the operation code of the instruction last executed. Flag bits of these two digits are not displayed.

Sense and Branch (S-B). Two lines of five lights each display Q_8 and Q_9 of the Branch Indicator, Branch No Indicator, and Input/Output instructions, from the Sense and Branch Register. Input/output device codes (digits 01-05) are displayed for input/output and control instructions.

NOTE: On some machines the *Sense and Branch Register* and the Digit Register are combined into a *Digit and Branch Register.* The functions of the combined register are the same as those of the individual registers.

Digit Register. These two lines of six lights each are used primarily for diagnostic testing by Customer Engineers. They display the digits affecting MARS (Memory Address Register Storage) during all I cycles. As the multiplication progresses, they display, from the Digit Register, the two product digits "looked up" in the multiply table. The Digit Register stores the partial product during multiplication.

Multiplier. This five-light multiplier register display shows each multiplier digit as it is used during a multiply operation.

Memory Address Register (MAR). Five lines of five indicator lights each display the bit configuration of the five-digit address in any one of the eight MARS registers. The specific register displayed is selected by the MAR display selector switch and the display MAR key. There is no flag bit notation.

Memory Address Register Storage (MARS) *Display Selector.* This 8-position rotary switch permits selection of any of the eight MARS registers (listed under SYSTEM CONFIGURATION) for display in MAR by depressing the Display MAR key. The position of the switch can be changed without altering the display. The rotary switch should not be turned, however, while the display MAR key is depressed.

The MARS registers provide a visual indication of internal data flow for the console operator. The sections titled PROGRAM INSTRUCTIONS, CONSOLE OPERATING PROCEDURES, PROGRAM TESTING, and Appendix E explain the use of individual registers, and the incrementing and decrementing of the MAR addresses.

CONTROL GATE INDICATORS

The control gate indicators (Figure 53) are used primarily for diagnostic testing by Customer Engineers. Those listed below, however, may be used during manual Single Cycle Execute (SCE) key operation of instructions.

H/P (High/Positive). The high/positive light shows the condition of the internal high/positive indicator as a result of the last arithmetic or compare operation.

E/Z (Equal/Zero). The equal/zero light shows the condition of the internal equal/zero indicator as a result of the last arithmetic or compare operation.

Bypass. The bypass light shows that the MAR address was neither decreased nor increased. Either the increment or decrement light is also on.

DECR (Decrement). The decrement light shows that the address routed from MAR to MARS was decreased by 1, unless the bypass light is also on. Both lights on indicate that no decrease occurred.

INCR (Increment). The increment light shows that the address routed from MAR to MARS was increased by 1, unless the bypass light is also on. Both lights on indicate that no increase occurred.

Plus 2. The plus 2 light shows that the address routed from MAR to MARS was increased by 2. The increment light must also be on.

REC MARK (Record Mark). The record mark light shows that a record mark was sensed in core storage. The record mark is displayed in MDR.

Branch. The branch light comes on during the I cycle of Branch Indicator and Branch No Indicator instructions if the branch is to occur during E cycle.

RECOMP (Recomplement). The recomplement light shows that an add or subtract result will be recomplemented upon completion of the computation.

Carry Out. The carry out light shows that the result from the add table has a carry (flag bit) or that a carry went into a position containing a nine, which causes the carry out light to come on one to three storage cycles later.

Carry In. The carry in light is turned on by a carry out and shows that 1 will be added on the next machine cycle.

IA (Indirect Addressing). The IA light shows that an indirect addressing operation is in progress. It is turned on when a flag bit is present in the units position of the indirect address.

Field Mk 1 (Field Mark 1). The field mark 1 light comes on when the flag bit in the high-order position of the Q field is detected in the MDR.

Field Mk 2 (Field Mark 2). The field mark 2 light comes on when the flag bit in the high-order position of the P field is detected in the MDR.

T/C 1 (True/Complement 1). The true/complement 1 light, when on, shows that the Q address data is complemented during arithmetic operations.

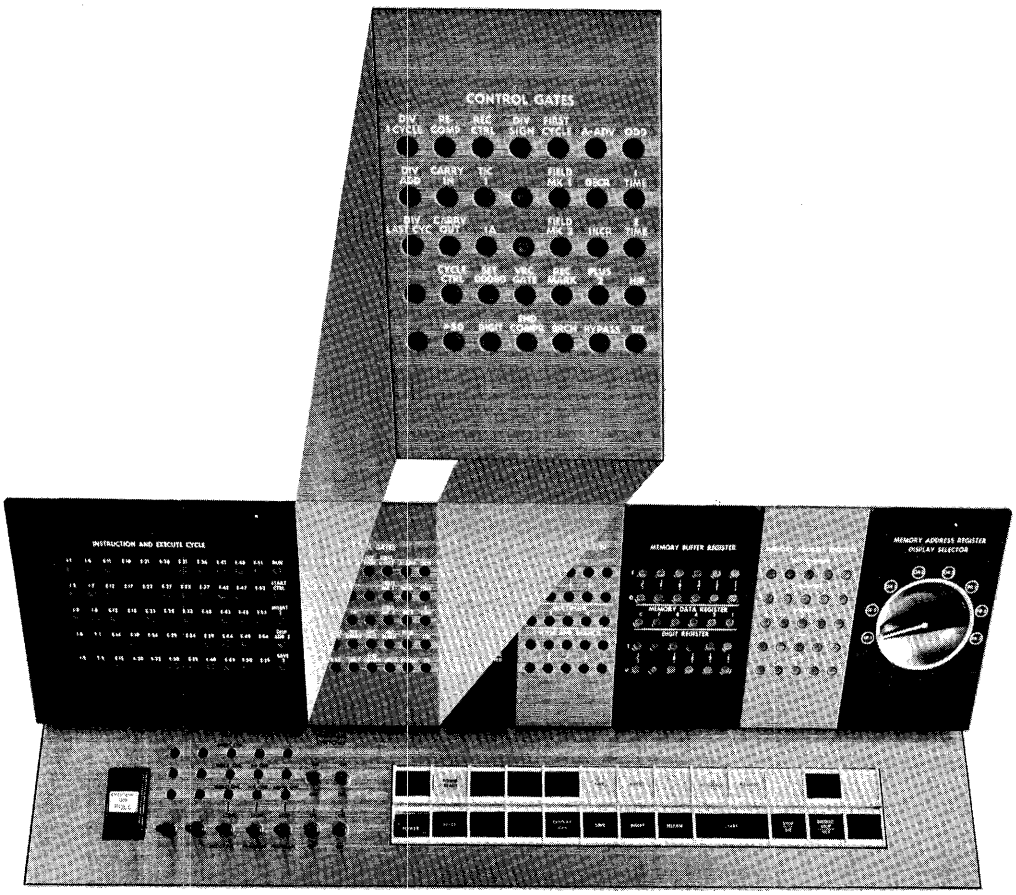


Figure 53. Control Gate Indicators

Instruction and Execute Cycle

The instruction and execute cycle lights (Figure 54) are a visual aid for the console operator in stepping an instruction through I and E cycles with the Instant Stop/SCE key. The I and E lights progress through each cycle with repetitive depressions of this key.

Input/Output

The input/output lights are used primarily for diagnostic testing by IBM Customer Engineers. The Last Card light (Figure 55) has significance for the operator and programmer.

LC (Last Card). This light turns on when data from the last card has been transferred from 1622 input buffer storage to 1620 core storage without a parity error.

Control Switches, Keys, and Signal Lights

Control keys (Figure 55) are used for performing certain manual operations and for convenient instruction entry. Signal lights associated with the control keys provide a visual indication of a specific operating condition of the computer and indicate which step of the keying procedure was last completed. Uses of many of the control keys are given under CONSOLE OPERATING PROCEDURES.

POWER ON/OFF SWITCH—POWER ON LIGHT

The power on/off switch has an ON and OFF position. Set to the ON position, it applies electrical power to the computer and turns on the power on light.

POWER READY LIGHT

The power ready light comes on when internal machine temperature and voltages reach proper operating values. There is a delay from the time the power on/off switch is positioned ON until operating temperature and voltages are obtained. This delay varies with room temperature and the elapsed time since power was turned off.

START KEY

The start key is used to start program processing and to put the computer in automatic mode. It is operative only when the computer is in manual mode.

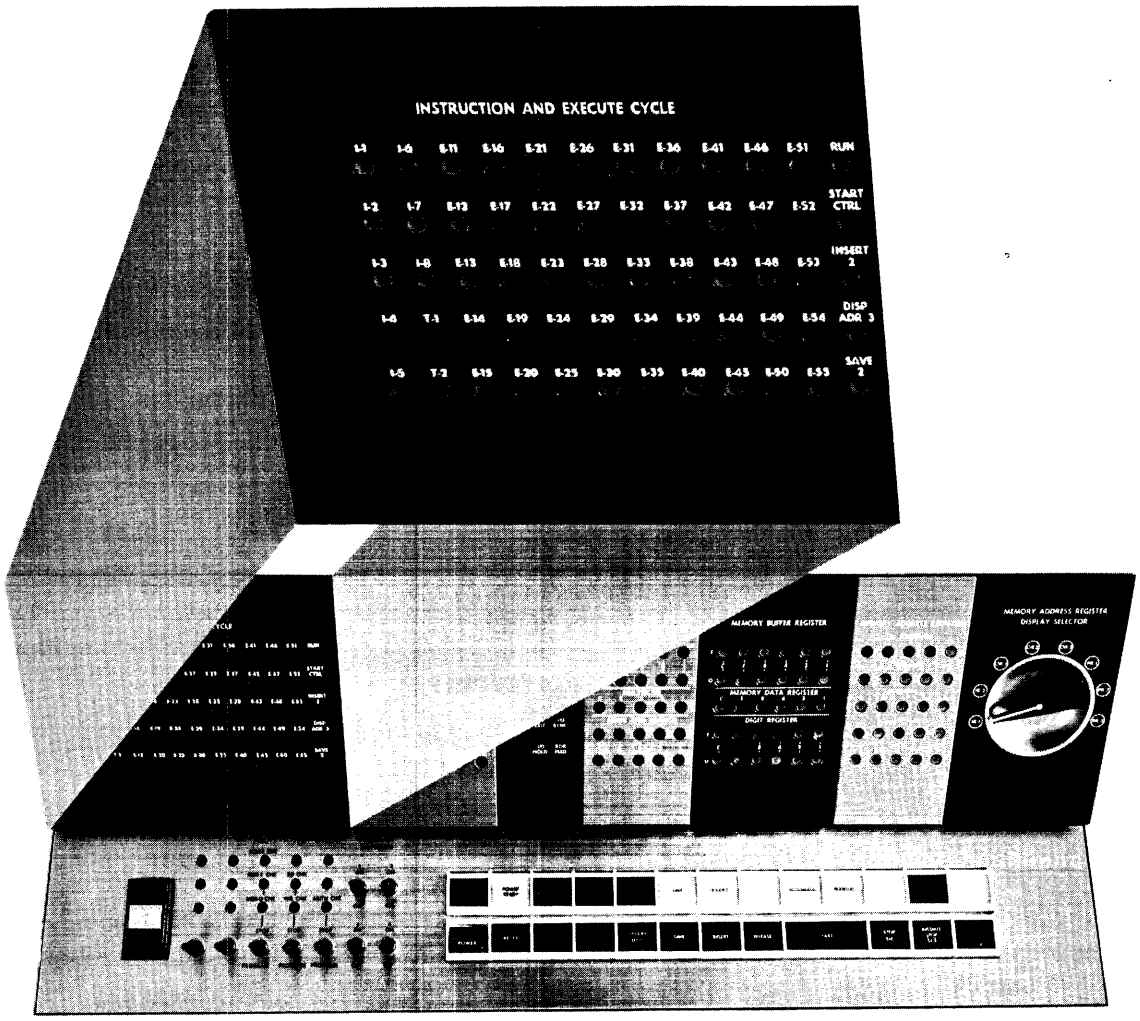


Figure 54. Instruction and Execute Cycle Lights

AUTOMATIC AND MANUAL LIGHTS

The manual light, when on, indicates that the computer is in manual mode; it is off when the computer is in automatic mode. Appendix F summarizes computer modes of operation. In manual mode, the computer has terminated all operation and is prepared to accept operator intervention.

The automatic light, when on, indicates that the computer is in automatic mode (e.g., while executing a stored program or while entering data into core storage from the typewriter keyboard).

Manual mode is initiated and the manual light is turned on by the execution of a Halt Instruction or by depression of the release key (on an I/O operation only), Instant Stop key, or Stop key. Depression of the Start key, Insert key, or Display MAR key initiates automatic mode and turns the manual light off. The Save light and/or the No Feed light can be on when the manual light is on.

Both the manual and automatic lights are on when an instruction is single-cycled with the SCE key.

RESET KEY

The reset key is used to restore all machine status indicators, machine check indicators, and signal lights to their initial or reset condition. The reset key functions only when the computer is in the manual mode (manual light on). Parity errors can occur if the reset key is used while the computer is in the automatic mode. When the computer is in the automatic mode, the instant stop key should be depressed to put the computer in the manual mode and permit use of the reset key.

INSERT KEY AND INSERT LIGHT

Depression of the insert key places the 1620 in automatic mode (all modes are summarized in Appendix F). Depression of the insert key also turns on the insert light and activates the typewriter keyboard so that direct entry of instructions may be made in numerical

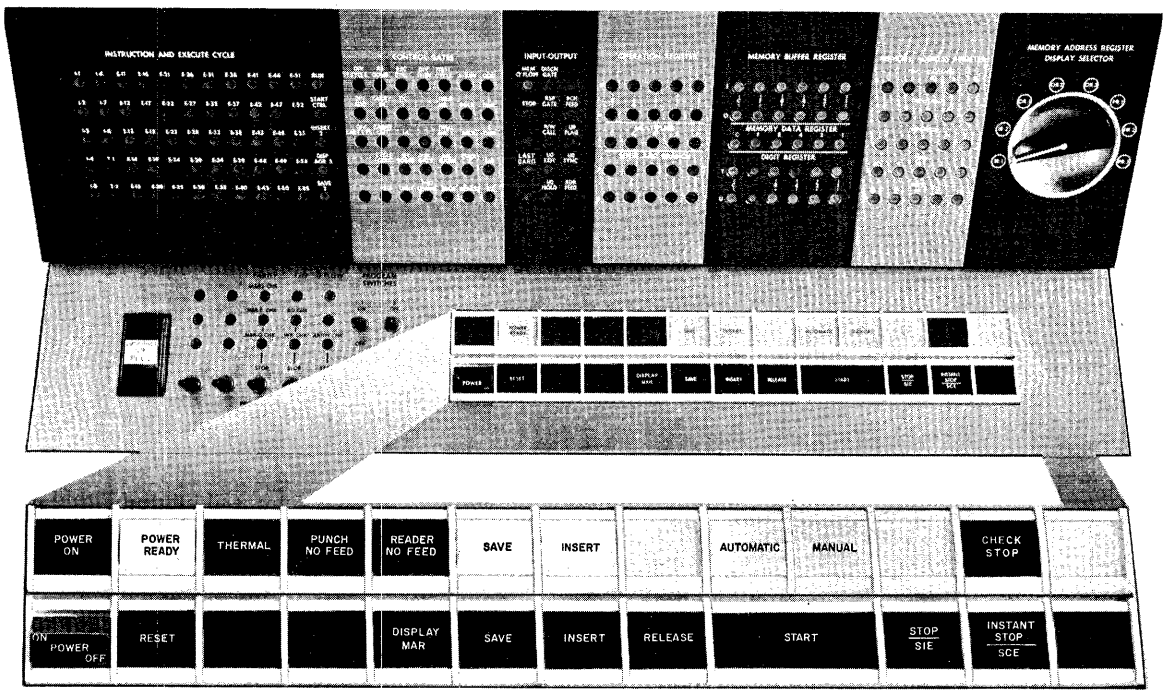


Figure 55. Control Keys and Signal Lights

mode, starting at 00000 and continuing into higher-numbered storage positions. As many as 100 digits may be keyed in. After the 100th digit is entered, an automatic release is initiated and the 1620 returns to manual mode.

When less than 100 characters are entered, entry of the last desired character should be followed by depression of the console Release and Start keys, or by depression of the R-S key on the typewriter keyboard. The R-S key combines the release and start functions of the console keys. The R-S symbol is typed as a permanent record that the R-S key has been used.

The insert key is operative only when the computer is in the manual mode.

SAVE KEY AND SAVE LIGHT

Depressing the save key turns on the save light and saves the address of the next sequential instruction to be executed. This address is saved in Product Address Register 1 (PR-1). If a multiply operation is performed before the saved address is used, the saved address is lost because the contents of PR-1 are decremented for each new multiply digit during a multiply operation.

The use of the save key is explained in PROGRAM ALTERATION AND DATA ENTRY under CONSOLE OPERATING PROCEDURES. See also BRANCH BACK under BRANCH INSTRUCTIONS.

RELEASE KEY

The release key is used to terminate any input/output operation, including console keyboard entry of data into core storage. When this key is depressed, manual mode is initiated, the manual light is turned on, and the insert light is turned off.

The release key is operative only when the computer is in automatic mode and performing an I/O operation.

STOP/SIE (SINGLE INSTRUCTION EXECUTE) KEY

Depression of the stop/SIE key stops the computer in manual mode at the end of the instruction being executed when the key is depressed.

The stop/SIE key also serves as a single instruction execute key. Successive depressions of the key cause one instruction to be executed for each depression. The manual light remains on.

INSTANT STOP/SCE (SINGLE CYCLE EXECUTE) KEY

Depression of the instant stop/SCE key causes the machine to stop at the end of the 20-microsecond machine cycle in progress when the key is depressed. Successive depressions of the key cause single machine cycles. Both manual and automatic lights remain on.

CHECK STOP LIGHT

The check stop light is turned on when the machine stops because of a parity check. One or more of the parity or I/O check indicators, that caused the stop is

also on. The check stop light is turned off when the check indicators are reset or the parity or I/O switch is set to PROGRAM.

DISPLAY MAR KEY

The display MAR key is operative only when the manual light is on and the automatic light is off. Depression of the display MAR key causes display of the MARS register to which the MARS display selector switch is set.

The rotary switch should not be turned while the display MAR key is depressed.

READER NO FEED LIGHT

The reader no feed light is turned on when the computer attempts a paper tape read or card read operation and the reader is not in the ready status. This not-ready status is often temporary in a card read operation because the buffer is interlocked while the read cycle is in process.

PUNCH NO FEED LIGHT

The punch no feed light is turned on if one of the following conditions exists:

1. The computer executes a write instruction using the tape punch and there is no paper tape on the feed reel.
2. A parity check occurs while punching paper tape.
3. The paper tape supply is exhausted.
4. The card punch is not ready. This not-ready status is often temporary on a card punch operation because the buffer is interlocked while the punch cycle is in process.

Any of these conditions stops the computer in automatic mode with both the automatic and punch no feed lights turned on. When a parity error occurs, the I/O write check light is also turned on. Depression of the release key disconnects the punch and puts the computer in manual mode. Depression of the reset key, while in manual mode, turns off the punch no feed and I/O write check light. Manual correction and restart procedures can begin after depression of the release and reset keys.

THERMAL LIGHT

The thermal light is turned on if the internal temperatures of the 1620, 1622, or 1623 become too high. Power is turned off, and the power ready light goes off. The thermal light may be turned off by depression of the reset key, after the internal machine temperatures return to normal. The power switch must be turned off and on again before power can be applied to the machine.

EMERGENCY OFF SWITCH

This switch is for emergency use only. If positioned OFF, all power is turned off in the machine and the blowers that cool the electronic circuits are stopped. Damage to the machine may therefore result.

Console Operating Procedures

The procedures that follow are included to aid the console operator. They may be modified as necessary to meet individual requirements.

Program Entry From Typewriter

Operator Action	Explanation
1. Depress insert key.	Typewriter is conditioned to enter data into core storage, beginning at location 00000.
2. Type: 36 xxxxx 00100 49 xxxxx (No Q address)	Enter instructions to read numerically from typewriter, beginning at the first position of program storage (xxxxx), and branch to first program instruction.
3. Depress release key.	Releases typewriter.
4. Depress start key.	The Read Numerically instruction, entered in step 2, is executed.
5. Type program steps and data.	As each character is typed, it is stored at location xxxxx and succeedingly higher core storage positions.
6. Depress release key.	Terminates read instruction.
7. Depress start key.	The next sequential instruction, which is the branch to the first program instruction at xxxxx, is executed.

Program Entry From Paper Tape Reader

Operator Action	Explanation
1. Depress insert key.	Typewriter is conditioned to enter data into core storage, beginning at location 00000.
2. Type: 36 xxxxx 00300 49 xxxxx (No Q address)	Enter instruction to read numerically from paper tape reader, beginning at the first position of program storage (xxxxx), and branch to first program instruction.
3. Depress release key.	Releases typewriter.
4. Depress start key.	The Read Numerically instruction, entered in step 2, is executed. The EL character punched in the tape causes a termination of the read instruction and execution of the next sequential instruction (branch to first program instruction, which was entered in step 2).

Program Alteration and Data Entry

Operator Action	Explanation
1. Depress stop key.	Halts processing and initiates manual mode.
2. Depress save key.	The address of the next instruction in sequence is saved in Product Address Register 1 (PR-1).

3. Depress insert key.	Typewriter is conditioned to enter data into core storage, beginning at location 00000.
4. Type: 36 xxxxx 00100 42 (No P or Q address)	Enter instructions to read numerically from typewriter beginning at the first position of data entry (xxxxx), and branch to address saved in PR-1 (step 2).
5. Depress release key.	Releases typewriter.
6. Depress start key.	The Read Numerically instruction, entered in step 4, is executed.
7. Type instructions and data.	As each character is typed, it is stored at location xxxxx and succeedingly higher core storage positions.
8. Depress release key.	Terminates read instruction.
9. Depress start key.	The next sequential instruction, which is Branch Back (step 4) to the address saved in PR-1 (step 2), is executed, and processing is resumed.

Print Core Storage Data on Typewriter

Operator Action	Explanation
1. Depress insert key.	Typewriter is conditioned to enter data into core storage, beginning at location 00000.
2. Type one of the following:	Enter instruction to:
39 xxxxx 00100	Write Alphamerically, beginning at xxxxx and continuing until a record mark is sensed, or,
38 xxxxx 00100	Write Numerically, beginning at xxxxx and continuing until a record mark is sensed, or,
35 xxxxx 00100	Write (Dump) Numerically, beginning at xxxxx and continuing until location 19999 is printed or the release key is depressed.
3. Depress release key.	Releases typewriter.
4. Depress start key.	Instruction entered in step 2 is executed.

Check Program Step Sequence and Operation

Operator Action	Explanation
1. Depress stop key.	Halts processing and initiates manual mode.
2. Depress SIE key.	Each depression causes the execution of one instruction.
3. Depress SCE key.	The OP code and the address of the next instruction to be executed are displayed in the OP register and MAR.
4. Depress SIE key.	The instruction displayed in step 3 is executed. Steps 3 and 4 can be alternated to display succeeding instructions.

Display P and Q Addresses

Operator Action	Explanation
1. Depress stop key.	Halts processing and initiates manual mode.
2. Depress SIE key until the instruction that contains the desired address is next.	One instruction is executed with each depression of the SIE key.
3. Depress the SCE key eight times.	This steps through the eight I cycles.
4. *Note the on/off conditions of the machine status and check indicators and of the signal lights to be reset by Reset (step 5), so that proper restart can be initiated after the display has been completed.* At this time a branch to the original instruction also must be inserted if it is desired to execute this instruction and proceed with the normal program. (If reset and the subsequent necessity for branching and proper restart are undesirable, step 5 can be omitted and the P and Q addresses can be viewed, two digits at a time, during SCE.)	
5. Depress reset key.	Initiates manual mode.
6. Turn MARS switch to OR-1 (Operand Address Register 1) and depress the display MAR key.	The Q address, which is in OR-1, is displayed in MAR.
7. Turn MARS switch to OR-2 (Operand Address Register 2) and depress the display MAR key.	The P address, which is in OR-2, is displayed in MAR.

Reset Core Storage to Zeros

Operator Action	Explanation
1. Depress stop key.	Halts processing and initiates manual mode.
2. Depress insert key.	Typewriter is conditioned to enter data into core storage, beginning at location 00000.
3. Type: 26 00008 00009	Enter instruction to transmit field from location 00009 to 00008.
4. Depress release key.	Releases typewriter.
5. Depress start key.	The instruction entered in step 3 is executed. The zero at 00009 is transmitted to 00008. Since the next location read is always the zero previously transmitted, there is no flag bit to halt the operation. Approximately 0.8 seconds is required to clear the entire 20,000 positions of core storage.
6. Depress instant stop key.	The operation is stopped with the machine in manual mode.

Program Testing

Thorough documentation and preparation should precede the testing of all programs. Careful console operation is next in importance. The following console oper-

ating procedures are helpful in analyzing programs in the computer:

Instruction (I) Cycle

Depression of the Instant Stop/SCE kep stops the computer at the end of the machine cycle in operation at the time the key is depressed. Successive depressions of the key step the computer through 20-microsecond machine cycles. The I and E cycle console lights show the progression of an instruction that is executed with each depression of the SCE key.

Eight depressions of the SCE key step the computer completely through the eight machine cycles of the I cycle of any instruction. During the I cycle, the following MARS registers are read into:

Operation (OP) Register — instruction code.

Instruction Address Register 1 (IR-1) — address of next instruction.

Operand Address Register 1 (OR-1) — Q address of the instruction in the OP register.

Operand Address Register 2 (OR-2) — P address of the instruction in the OP register.

Depression of the reset key puts the computer in manual mode.

Alternate depressions of the display MAR key and positioning of the MARS display switch (they must not occur simultaneously) show the operator the addresses of P and Q data. Once the location of the data is known, a write instruction inserted at 00000 can be used to cause printing of the data.

Figure 56 shows that both IR-1 and MAR are involved in the reading of an instruction from core storage. At the beginning of the I cycle, IR-1 contains the address of the first operation code digit, O_0, of the instruction.

During the first machine cycle, the address contained in IR-1 is placed in MAR. The two-digit operation code of the instruction is moved from core storage to MBR-even and MBR-odd and transmitted to the two-digit operation register. Simultaneously, the content of MAR is routed through the increment switch, incremented by 2 and placed back in IR-1.

During the next machine cycle, the address in IR-1 is placed in MAR, and the first two digits of the P address, P_2 and P_3, are read from core storage to the MBR and are placed, respectively, in the fifth and fourth high-order positions of OR-2. In the same manner, P_4 and P_5 are placed in the hundreds and tens positions of OR-2.

One cycle is then used to place P_6 into the units position of OR-2 and the address is routed through the increment switch, incremented and placed back in IR-1. The first digit of the Q address, Q_7, is placed in the high-order position of OR-1.

During the next two machine cycles, the remaining digits of the Q address are placed in OR-1.

By the end of the I cycle, IR-1 has stepped twelve addresses higher to the high-order digit (O_0) of the next

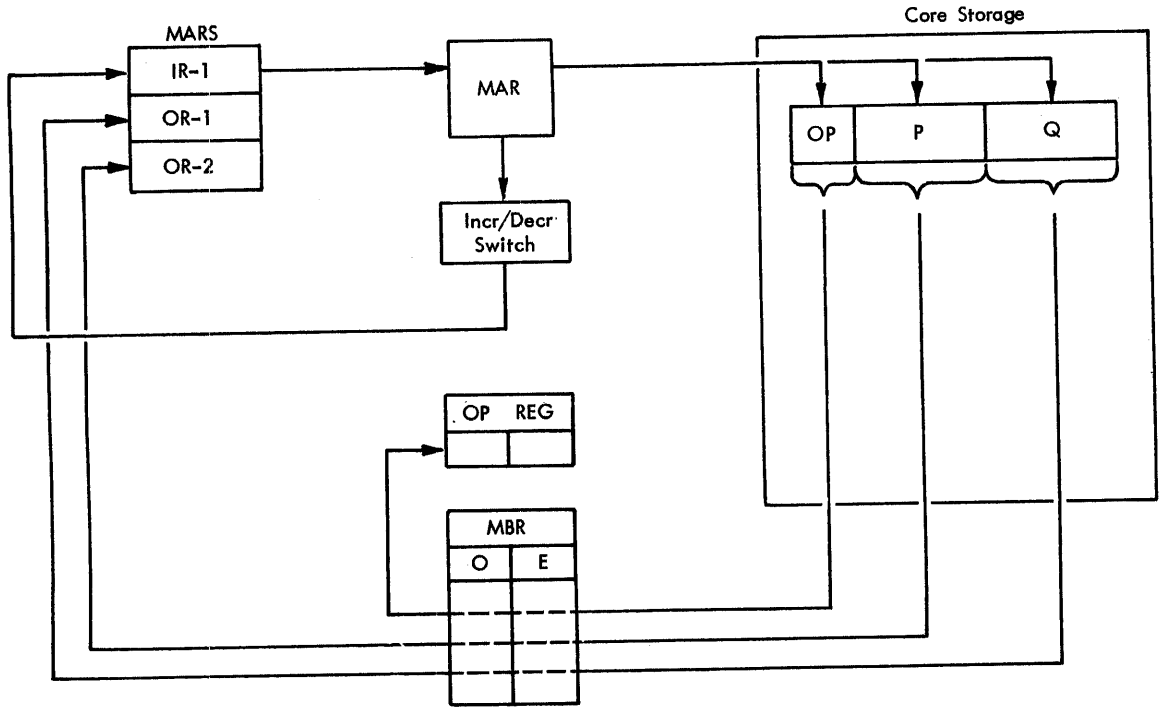


Figure 56. Schematic Diagram of I Cycle

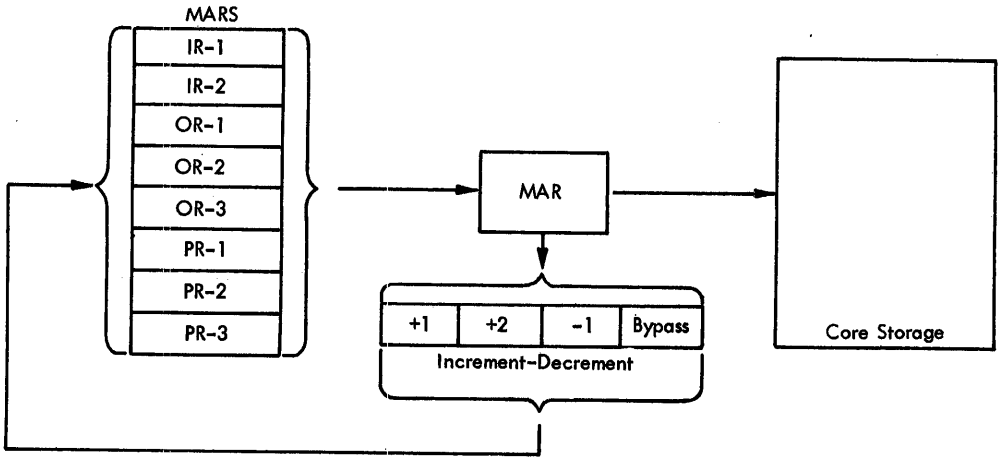


Figure 57. Schematic Diagram of E Cycle

instruction in the program. It remains at this address during the following E cycle.

The following differences occur for immediate instructions:

1. The Q address digits are not routed into OR-1.
2. During the last I machine cycle, the address of Q_{11} is sent from MAR to OR-1 (MAR was incremented during the first seven I cycles and contains the address of Q_{11}).

Execution (E) Cycle

The number of SCE key depressions necessary to step the computer through an E cycle is determined by the instruction and the size of the fields.

Manual operation (successive SCE key depressions) is used to visually check data flow in the computer. At any point in an operation, the register displays can be used for analysis. The MARS registers can be displayed by using the following procedure:

1. Depress the reset key – this prevents further progression of the instruction being executed.
2. Position the MARS display switch to select the desired register.
3. Depress the display MAR key.

MAR is addressed by the eight MARS registers (see SYSTEM CONFIGURATION) during machine operation. The MARS register that addresses MAR varies with individual instructions and machine cycles. The address received by MAR is returned to one or more MARS registers by way of the increment-decrement switch as shown in Figure 57.

The increment-decrement switch may increase by 1 or 2, decrease by 1, or bypass (not change) the address returning from MAR to the MARS register.

Arithmetic Operations

Arithmetic principles, data flow diagrams, and explanations are given below for three instructions to facilitate the analysis of instructions executed in the computer.

ADDITION

Addition is accomplished in the 1620 by combining data digits to form an add table address. The answer is then "looked up" in the add table. Reference to the table in Appendix B will aid in following the diagram (Figure 58), which shows, schematically, how the amounts 185 and 52 are added. It is accomplished as follows:

1. The machine automatically inserts 003 into the three high-order positions of MAR (the add table is located in core storage positions 00300-00399).
2. The P units digit (5) goes into the tens position of MAR.
3. The Q units digits (2) goes into the units position of MAR.
4. The sum (7), which is located at 00352 of the add table, replaces the P units digit.
5. The P tens digit (8) goes into the tens position of MAR.
6. The Q tens digit ($\bar{5}$) goes into the units position of MAR.
7. The sum ($\bar{3}$), which is located at 00385 of the add table, replaces the P tens digit. A carry is also present in the add table and is used to modify the next MAR address.
8. The P hundreds digit ($\bar{1}$) goes into the tens position of MAR.
9. There is no Q hundreds digit, but a zero is inserted internally. The zero, plus the carry from step 7, causes a 1 to go into the units position of MAR.

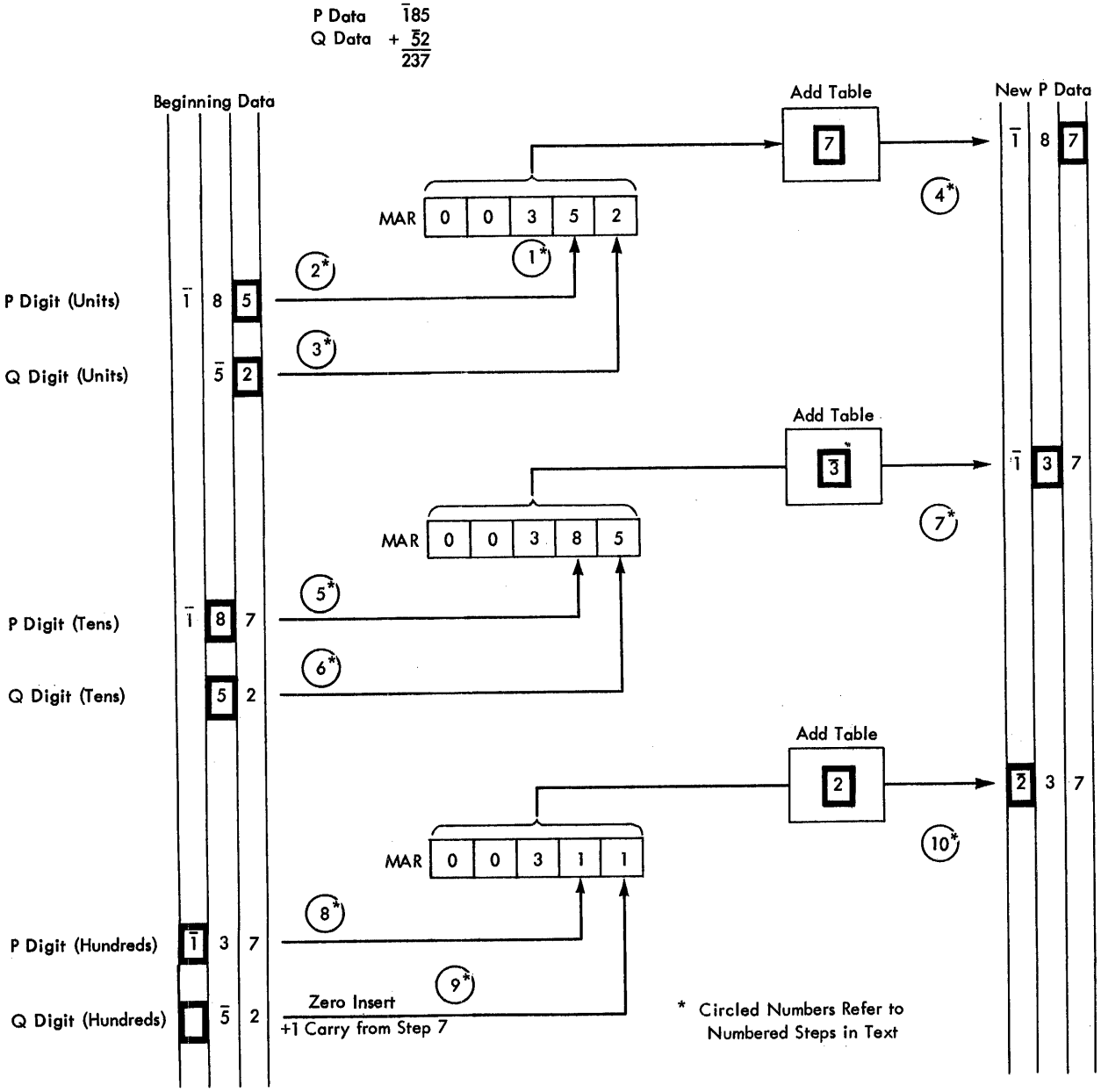


Figure 58. Add Operation Diagram

10. The sum (2), which is located at 00311 of the add table, replaces the P hundreds digit.

Figure 59 shows how OR-1 and OR-2 share MAR in providing locations of data in core storage. They are decremented since this is a digit-by-digit operation. OR-3 retains the P address for sum recomplement, if necessary.

The Q data digits are routed through the digit register to the units position of MAR for development of the add table addresses.

The P data digits are routed through MDR to the tens position of MAR for development of the add table address.

003 is automatically inserted in MAR as the three high-order positions of the add table address. MAR is used to address the add table. The sum "looked up" in the add table replaces the P data.

SUBTRACTION

The following steps with reference to Figure 60 indicate how 52 is subtracted from 185:

1. The machine automatically inserts 003 into the high-order positions of MAR. The add table (00300-00399) is used for subtraction also.
2. The P units digit (5) goes into the tens position of MAR.

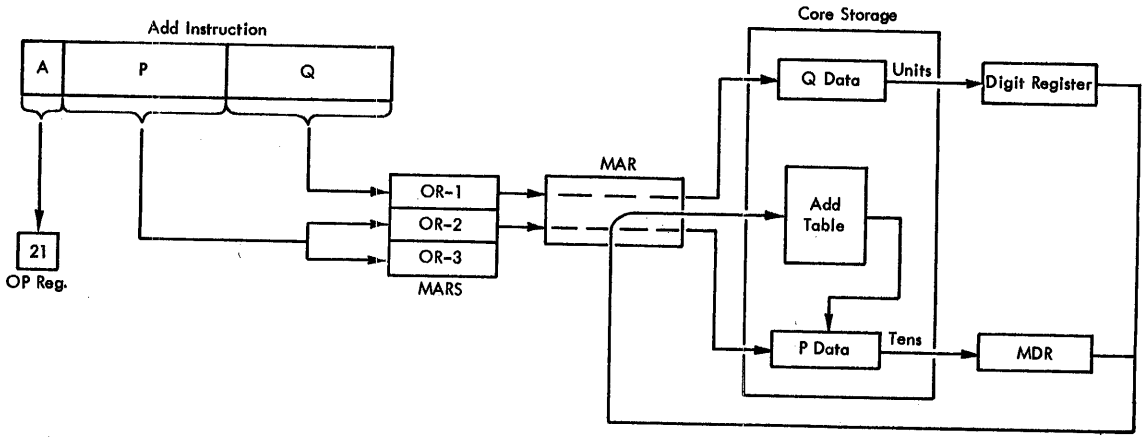


Figure 59. E Cycle of Add Operation

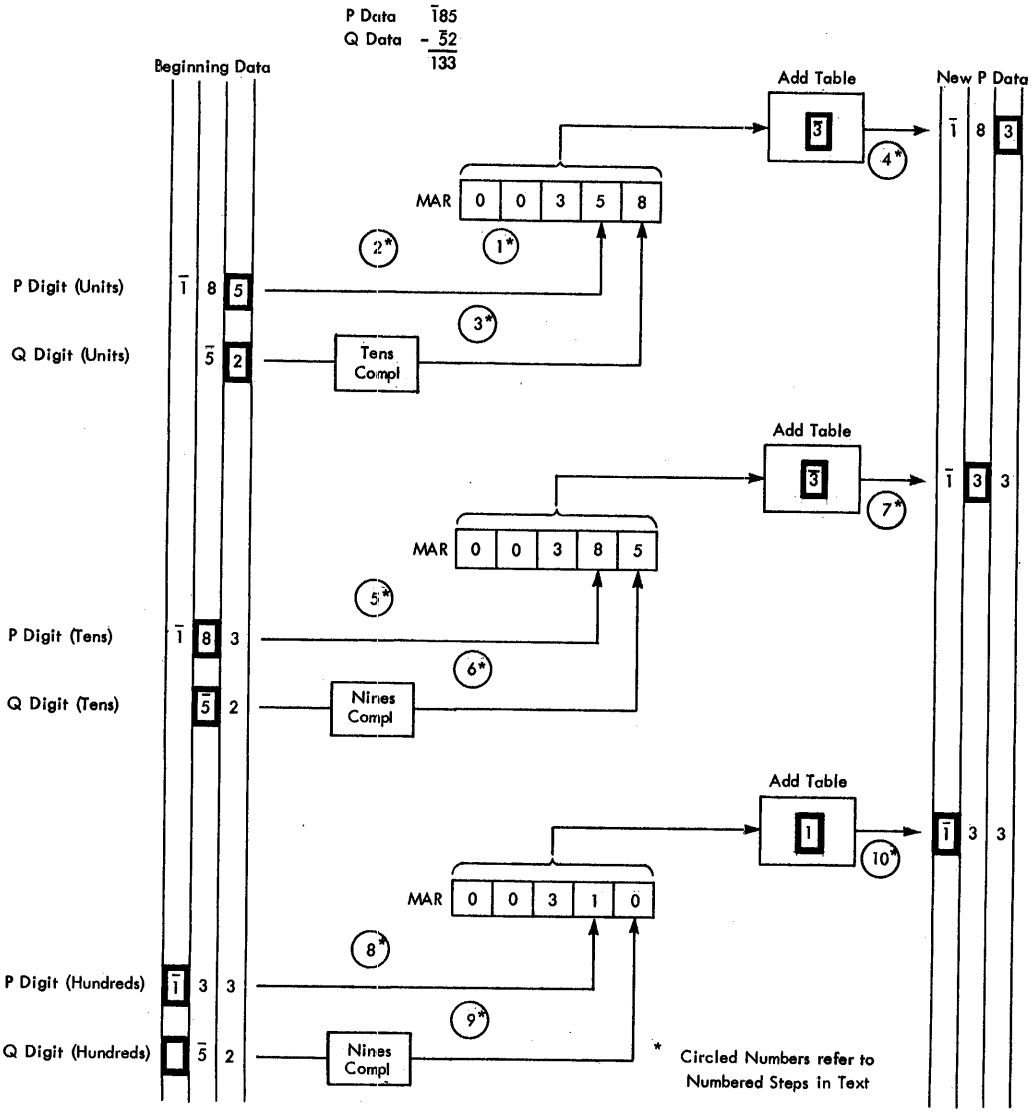


Figure 60. Subtract Operation Diagram

3. The Q units digit (2) is tens complemented and the complement (8) goes into the units position of MAR.
4. The difference ($\bar{3}$), which is located at 00358 of the add table, replaces the P units digit. A carry is also present at 00358.
5. The P tens digit (8) goes into the tens position of MAR.
6. The Q tens digit ($\bar{5}$) is nines complemented and the complement (4), plus the stored carry (step 4), causes a five (5) to go into the units position of MAR.
7. The difference ($\bar{3}$), which is located at 00385 of the add table, replaces the P tens digit. A carry is also present at 00385.
8. The P hundreds digit ($\bar{1}$) goes into the tens position of MAR.
9. There is no Q hundreds digit but a zero is inserted internally. The zero is nines complemented and the nine, plus the carry stored in step 7, causes a zero (0) to go into the units position of MAR.
10. The difference (1), which is located at 00310 of the add table, replaces the P hundreds digit.

Data flow for subtract is the same as that for add (Figure 59) except that the Q digits are complemented as they are routed from the digit register to MAR.

The console T/C light goes off when a digit is complemented.

MULTIPLICATION

Multiplication is accomplished in the 1620 by combining the digits to be multiplied into a multiply table address. The answer is then "looked up" in the multiply table. Reference to the table in Appendix C will aid in following the procedure shown in Figure 61.

The following steps refer to Figure 61 to show how each multiplier digit is used with the multiplicand. Twelve (12) is multiplied by 12 as follows:

1. The machine automatically inserts 00 into the two high-order positions of MAR. (The multiply table is located in core storage positions 00100-00299).
2. The P units digit (2) goes into the tens position of MAR.
3. The Q units digit (2) is routed through the Multiply Register (MR) and the doubler. The doubler is an internal device that doubles the units digit of the multiplier. The doubler increases the 2 to 04 and routes its units digit (4) to the units position of MAR. The tens digit of the doubler (0) is incremented by one and routed to the hundreds position of MAR.
4. The product is "looked up" in the multiply table. The developed MAR address is 00124 and the digit 4 is located at that address of the multiply table. Internal machine operation causes the two digits within the heavy black vertical lines of the multiply table (Appendix C) to be reversed and routed out. In this example a 4 is located at 00124. A zero is next to the 4 and both are found within the same heavy vertical lines. Thus 04 is routed out of the multiply table.
5. The 04 is added to 00 in the product area (the product area was reset to zeros as a result of the multiply instruction). Two add cycles are necessary to accomplish this addition.
6. The P tens digit ($\bar{1}$) goes into the tens position of MAR.
7. The Q units digit (2) is doubled etc., as described in step 3.
8. The developed MAR address (00114) causes 02 to be routed out of the multiply table as described in step 4.
9. The 02 is added one place to the left in the product area. The developed product is 024.
10. The P units digit (2) goes into the tens position of MAR.
11. The Q tens digit ($\bar{1}$) is doubled, etc., as described in step 3.
12. The developed MAR address (00122) causes 02 to be routed out of the multiply table as described in step 4.
13. The 02 is added to 02 in the product area. The developed product is 044.
14. The P tens digit ($\bar{1}$) goes into the tens position of MAR.
15. The Q tens digit ($\bar{1}$) is doubled etc., as described in step 3.
16. The developed MAR address (00112) causes 01 to be routed out of the multiply table as described in step 4.
17. The 01 is added one place to the left in the product area. The developed product is 0144.
18. A flag bit is inserted in the high-order digit of the developed product ($\bar{0}$144).

Data flow for the multiplier and multiplicand digits, shown in Figure 62, is as follows:

1. The multiplier digit is routed, to the multiply register (MR) and doubled. One (1) is added to the tens position of the doubler and the sum becomes the hundreds digit of MAR, which addresses the multiply table. The units digit of the doubler becomes the units digit of MAR, which addresses the multiply table.
2. The multiplicand digit is routed to the MDR and becomes the tens digit of MAR, which addresses the multiply table.

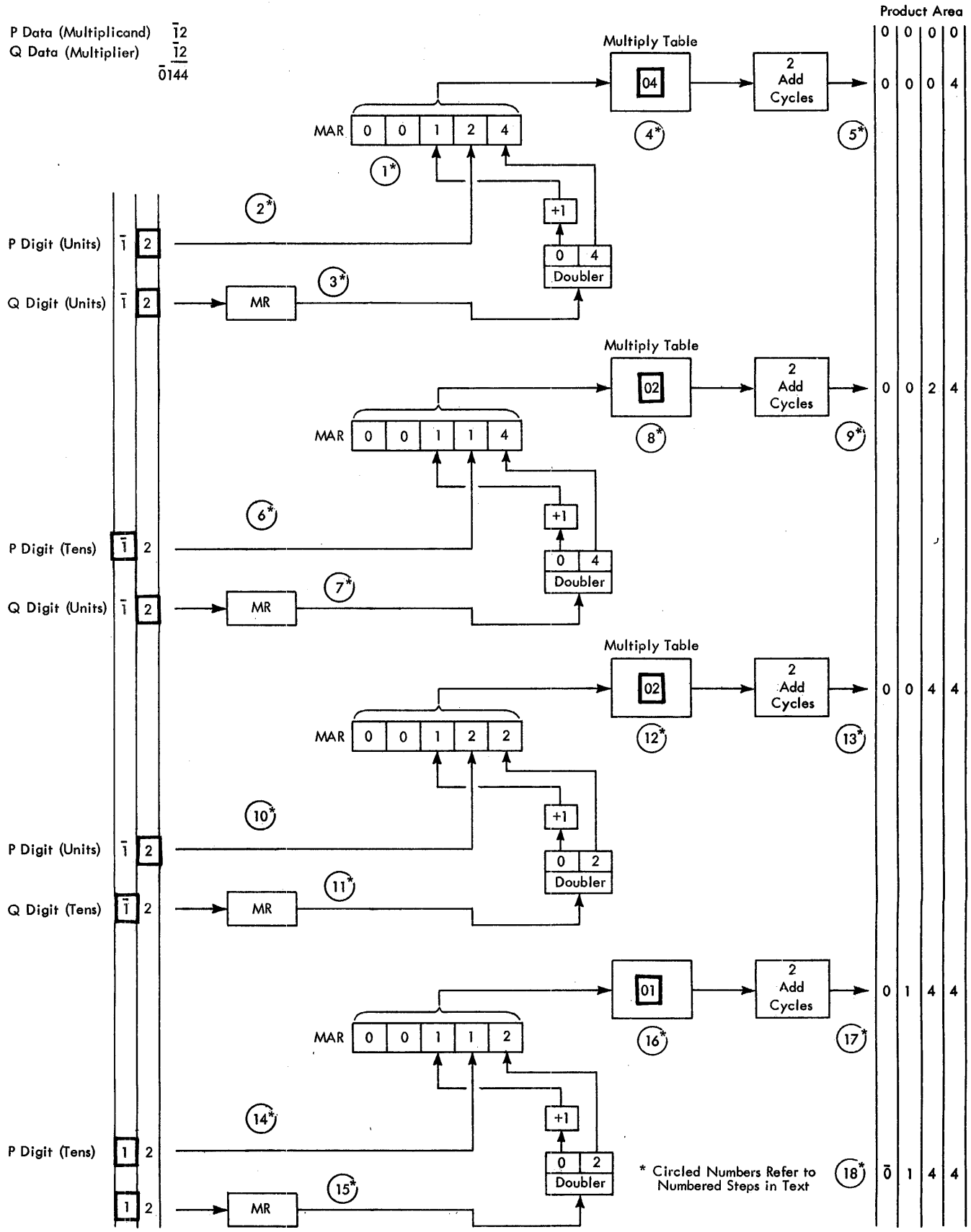


Figure 61. Multiply Operation Diagram

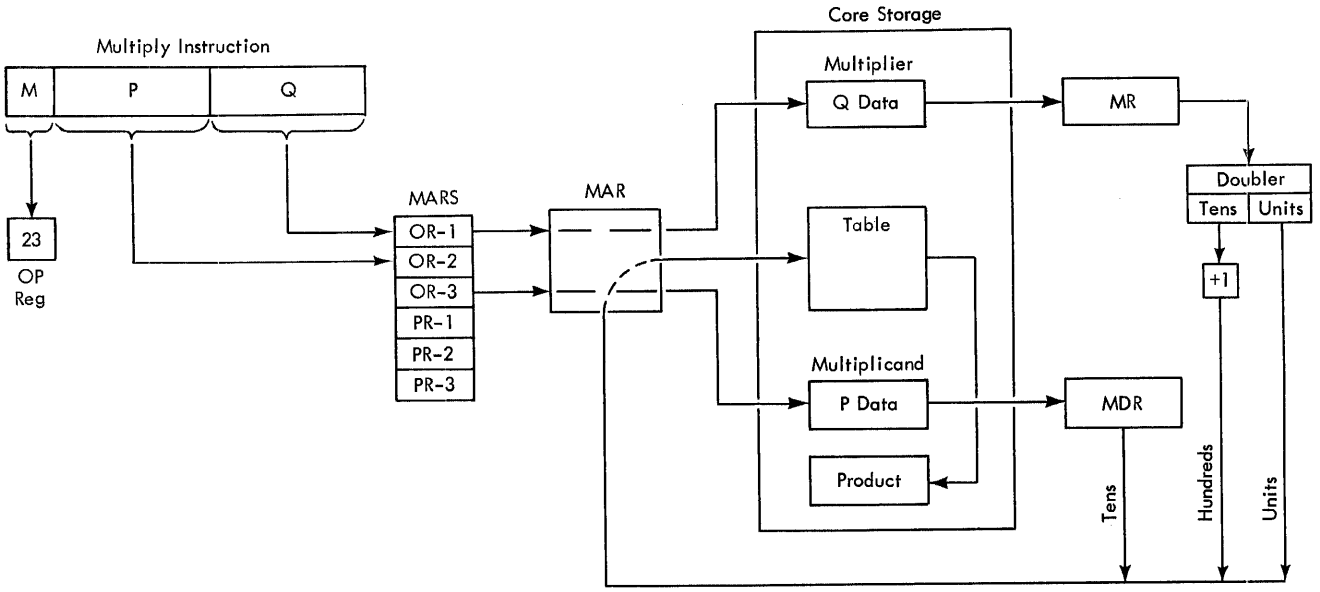


Figure 62. E Cycle of Multiply Operation

3. 00 is inserted in MAR as the two high-order digits of the address of the multiply table.
4. The product "looked up" in the multiply table is routed to 00099 and successively lower-numbered core storage positions.
5. Each multiplier digit is used for all the multiplicand digits. When a flag bit (field mark) is detected in the multiplicand, a new digit of the multiplier is obtained per OR-1; OR-2 is reset to the contents of OR-3 to begin the cycle again (see the following description of how the MARS registers are used).

The MARS registers are used during multiply operations, as follows:

1. PR-1 is set to 00099 (units position of product area).
2. OR-3 is set to OR-2 (units position of multiplicand). OR-3 retains that address so that each multiplier digit can start with the units digit of the multiplicand.
3. OR-2 is decremented 1 for each multiplicand digit. Each multiplier digit is used with all digits of the multiplicand (one digit at a time).
4. PR-1 is decremented for each new multiplier digit.
5. PR-2 is set to PR-1. PR-2 addresses the correct partial products position for the two add cycles that add a basic multiply-cycle result to the partial product.
6. PR-2 is decremented 1 for each multiplicand digit used.
7. Near the end of each basic multiply cycle, PR-3 is set to PR-2. PR-3 is decremented 1 and addresses the tens result digit for the second add cycle. PR-3 is decremented 1 again for a carry that may result from the second add cycle.

Data Flow

The console operator can observe data flow during execution of instructions by use of the SCE key. The three data flow diagrams included are representative of most operations.

TRANSMIT DIGIT DATA FLOW

Figure 63 shows how the digit at the Q address of the instruction is routed through MDR, MBR, and to the P address of the instructions.

OR-1 and OR-2 receive the Q and P addresses during the I cycle.

MAR is set by OR-1 and OR-2 during E time, and in turn addresses core storage.

INPUT DATA FLOW

Figure 64 shows how the input device (typewriter, card reader, or tape reader) causes data to be routed through MDR and MBR to the address specified by MAR. Succeedingly higher-numbered core storage positions are read into until the operation is terminated.

OUTPUT DATA FLOW

Figure 65 shows how MAR causes the address of the first character to be routed through MBR, MDR, and the output device (typewriter, paper tape, or card punch).

Succeedingly higher-numbered positions in core storage follow until a record mark or the 80th position of output card buffer storage is reached.

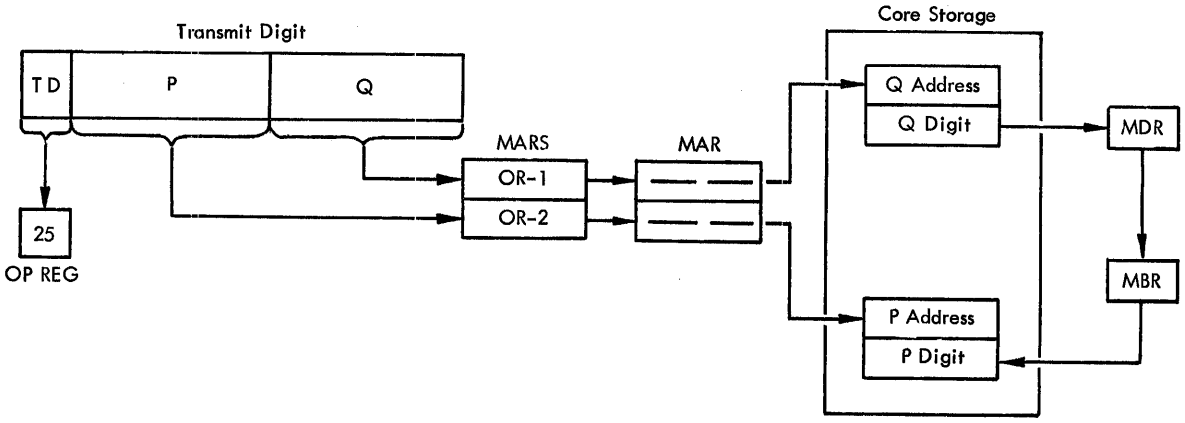


Figure 63. E Cycle of Transmit Digit Operation

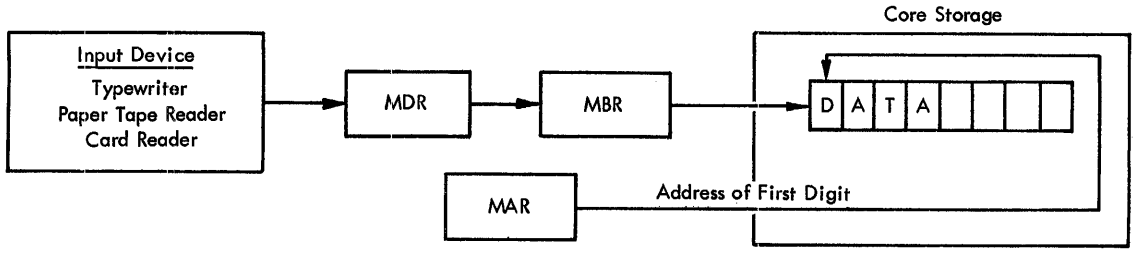


Figure 64. Data Flow of Input Operation

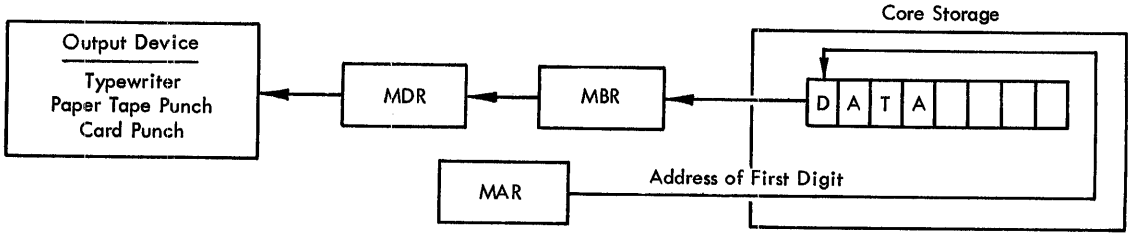


Figure 65. Data Flow of Output Operation

DETAILED DATA FLOW

A more complete picture of the internal operation of the 1620 is shown by the detailed data flow diagram in Figure 66. This diagram is primarily used by IBM Customer Engineers in diagnostic checking of the 1620 and is not necessarily a prerequisite to operating or programming the system.

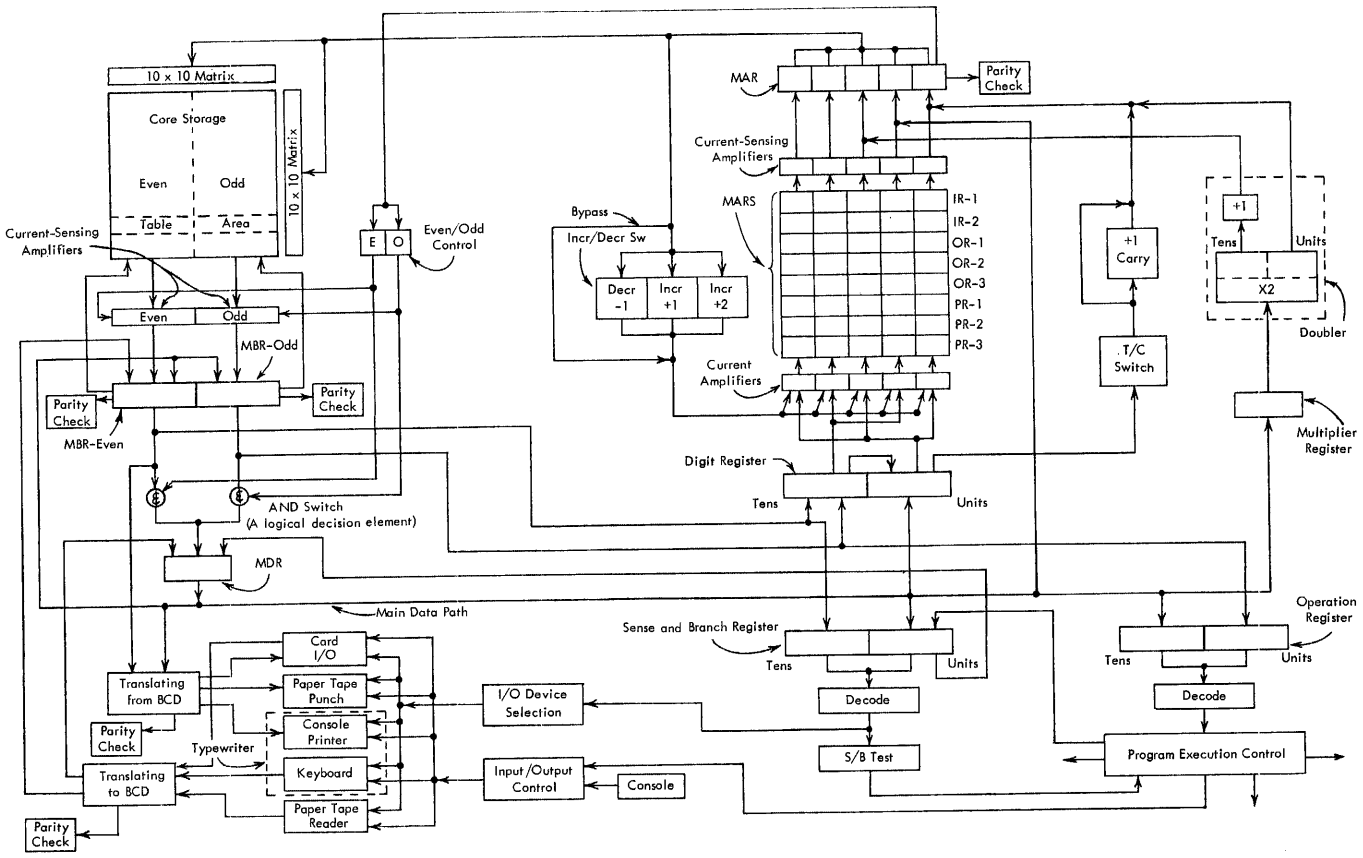


Figure 66. Detailed Data Flow Diagram

Program Languages

Any one of three programming languages may be used to program the 1620. They are FORTRAN, SPS, and machine (absolute) language.

Fortran

FORTRAN was developed by IBM primarily for scientific and engineering problems and consists of two major parts: the FORTRAN language and the FORTRAN translator.

FORTRAN language is comprised of a number of statements that the programmer uses in coding the problem to be solved. These statements are grouped into four classifications: arithmetic, control, input/output, and specification.

The FORTRAN translator is a program that accepts and translates FORTRAN statements. A program written in FORTRAN language is known as the "source program" and the machine language program ultimately produced is known as the "object program."

The programmer need concern himself only with the FORTRAN language, since the translation and compilation of the object program are automatic.

Use of the FORTRAN system consists of the following steps:

1. Read in the prepunched FORTRAN translator program via card or paper tape reader. This is sometimes called the processor or assembly program.
2. Enter the FORTRAN language program prepared by the programmer, via punched cards, typewriter keyboard, or paper tape reader. The object program will be punched out on the card or paper tape punch. The object program is an actual machine language program of numerical codes.
3. Read in the object program on the paper tape or card reader.

As an example, an equation

$$H=\frac{CA(K-T)}{W^2}$$

is stated in FORTRAN language as follows:

H=C*A* (K–T)/–W* *2.

The FORTRAN statement of this equation required the use of the following basic symbols:

* denotes multiplication

/ denotes division

* * denotes exponentiation; e.g., A * *3. means A^3

– denotes subtraction (+ denotes addition)

The IBM *1620* FORTRAN *Bulletin* (Form J26-4200) describes FORTRAN language in detail.

Symbolic Programming System

The Symbolic Programming System (SPS) is comprised of the following:

1. The "source program" is written by the programmer in a symbolic language that uses mnemonics in place of operation codes, and symbols for data or instructions.
2. The "processor" is the 1620 machine language program that performs the actual functions of translation and assembly.
3. The "object program" is the finished machine language program that the source and processor programs assembled.

The processor takes the source program in symbolic language, translates the mnemonic codes into machine language codes, assigns core storage addresses to instructions and symbolic data, and assembles a finished machine language program known as the object program. The *1620* SPS *Bulletin* (Form J26-4201) describes the SPS function in detail.

Appendix A

Execution Times and Operations Performed for All 1620 Operation Codes

Mnemonic	Code	Instruction	Operation	Time
A	21	Add	$F_P + F_Q$ replaces F_P	160 + 80 D_P, basic time 80 D_P, recomp. time†
AM	11	Add (I)	$F_P + Q$ replaces F_P	160 + 80 D_P, basic time 80 D_P, recomp. time†
B	49	Branch	Do I_P	200
BB	42	Branch Back	Do I_S	200
BD	43	Branch on Digit	If d_Q not zero, do I_P	200 No Branch 240 Branch
BI	46	Branch Indicator	If i_Q on, do I_P	160 No Branch 200 Branch
BNF	44	Branch No Flag	If no f_Q, do I_P	200 No Branch 240 Branch
BNI	47	Branch No Indicator	If i_Q off, do I_P	160 No Branch 200 Branch
BNR	45	Branch No Record Mark	If no r_Q, do I_P	200 No Branch 240 Branch
BT	27	Branch & Transmit	Save A_S, F_Q to $L_P - 1$, do I_P	200 + 40D_Q
BTM	17	Branch & Transmit (I)	Save A_S, Q to $L_P - 1$, do I_P	200 + 40$D_{Q'}$
C	24	Compare	F_P compared with F_Q	200 + 80D_Z Unlike signs 160 + 80D_P Like signs
CF	33	Clear Flag	Remove f from L_P	200
CM	14	Compare (I)	F_P compared with Q	200 + 80D_Z Unlike signs 160 + 80D_P Like signs
D	29	Divide	Product Area (00080-00099) ÷ F_Q	160 + 520$D_V Q_T$ + 740Q_T Average quotient digit 4.5
DM	19	Divide (I)	Product Area (00080-00099) ÷ Q	160 + 520$D_V Q_T$ + 740Q_T Average quotient digit 4.5
DN	35	Dump Numerically	I/O_Q writes from L_P to 19, 999, 39, 999 or 59, 999	§
H	48	Halt	Stop	160
K	34	Control	Do Q_{11} on I/O_Q	¶
LD	28	Load Dividend	F_Q to L_P	400 + 40D_N
LDM	18	Load Dividend (I)	Q to L_P	400 + 40D_N
M	23	Multiply	F_Q x F_P (result at 00099)	560 + 40D_Q + 168$D_P D_Q$
MF	71	Move Flag	f_Q to L_P	240
MM	13	Multiply (I)	Q x F_P (result at 00099)	560 + 40$D_{Q'}$ + 168$D_P D_{Q'}$
NOP	41	No Operation	Go to A_R	160
RA	37	Read Alphamerically	I/O_Q reads at $L_P - 1$	§ Except Card I/O (3.4 ms)
RN	36	Read Numerically	I/O_Q reads at L_P	§ Except Card I/O (3.4 ms)
S	22	Subtract	$F_P - F_Q$ replaces F_P	160 + 80D_P, basic time 80 D_P, recomp. time‡
SF	32	Set Flag	Place f at L_P	200
SM	12	Subtract (I)	$F_P - Q$ replaces F_P	160 + 80D_P, basic time 80 D_P, recomp. time‡
TD	25	Transmit Digit	d_Q to L_P	200
TDM	15	Transmit Digit (I)	Q_{11} to L_P	200
TF	26	Transmit Field	F_Q to L_P	160 + 40D_Q
TFM	16	Transmit Field (I)	Q to L_P	160 + 40$D_{Q'}$
TNF	73	Transfer Numerical Fill	F_Q to F_P	160 + 40 D_P
TNS	72	Transfer Numerical Strip	F_Q to F_Q	160 + 40 D_P
TR	31	Transmit Record	R_Q to L_P	160 + 40D_Q
WA	39	Write Alphamerically	I/O_Q writes from $L_P - 1$	§ Except Card I/O (3.4 ms)
WN	38	Write Numerically	I/O_Q writes from L_P	§ Except Card I/O (3.4 ms)

I Immediate.
† If signs initially unlike and numerical value of Q data greater than P data.
‡ If signs initially alike and numerical value of Q data greater than P data.
§ Depends on speed of I/O device and number of characters involved.
¶ Depends on control function and speed of I/O device.

Symbols and Definitions for "Operation" Column			
P	P part of instruction	R_Q	Record defined by Q
Q	Q part of instruction	I/O_Q	I/O defined by $Q_t Q_u$
F_P	Field defined by P	d_Q	Digit at L_Q
F_Q	Field defined by Q	f_Q	Flag bit at L_Q
I_P	Instruction defined by P	f	Flag bit
I_S	Saved instruction	r_Q	Record mark at L_Q
L_P	Location defined by P	i_Q	Indicator defined by $Q_t Q_u$
L_Q	Location defined by Q	A_S	Address of next seq. instr.

Symbols and Definitions for "Time" Column	
D_P	Number of digits, including high-order zeros, in the field at P.
D_Q	Number of digits, including high-order zeros, in the field at Q.
$D_{Q'}$	Number of digits, including high-order zeros, in the Q part of the instruction.
D_N	Number of digits, including high-order zeros, in the dividend.
Q_T	Number of digits, including high-order zeros, in the quotient.
D_V	Number of digits, including high-order zeros, in the divisor.
D_Z	Number of positions compared until a digit other than zero is detected in either field.
All times are in microseconds (1 microsecond = 1/1,000,000 second).	

Appendix B

Add Table

High-Order Positions of Address	Units Position of Address 0	1	2	3	4	5	6	7	8	9
0030	0	1	2	3	4	5	6	7	8	9
0031	1	2	3	4	5	6	7	8	9	$\overline{0}$
0032	2	3	4	5	6	7	8	9	$\overline{0}$	$\overline{1}$
0033	3	4	5	6	7	8	9	$\overline{0}$	$\overline{1}$	$\overline{2}$
0034	4	5	6	7	8	9	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$
0035	5	6	7	8	9	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$	$\overline{4}$
0036	6	7	8	9	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$	$\overline{4}$	$\overline{5}$
0037	7	8	9	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$	$\overline{4}$	$\overline{5}$	$\overline{6}$
0038	8	9	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$	$\overline{4}$	$\overline{5}$	$\overline{6}$	$\overline{7}$
0039	9	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$	$\overline{4}$	$\overline{5}$	$\overline{6}$	$\overline{7}$	$\overline{8}$

Appendix C

Multiply Table

High-Order Positions of Address	Units Position of Address 0	1	2	3	4	5	6	7	8	9
0010	0	0	0	0	0	0	0	0	0	0
0011	0	0	1	0	2	0	3	0	4	0
0012	0	0	2	0	4	0	6	0	8	0
0013	0	0	3	0	6	0	9	0	2	1
0014	0	0	4	0	8	0	2	1	6	1
0015	0	0	5	0	0	1	5	1	0	2
0016	0	0	6	0	2	1	8	1	4	2
0017	0	0	7	0	4	1	1	2	8	2
0018	0	0	8	0	6	1	4	2	2	3
0019	0	0	9	0	8	1	7	2	6	3
0020	0	0	0	0	0	0	0	0	0	0
0021	5	0	6	0	7	0	8	0	9	0
0022	0	1	2	1	4	1	6	1	8	1
0023	5	1	8	1	1	2	4	2	7	2
0024	0	2	4	2	8	2	2	3	6	3
0025	5	2	0	3	5	3	0	4	5	4
0026	0	3	6	3	2	4	8	4	4	5
0027	5	3	2	4	9	4	6	5	3	6
0028	0	4	8	4	6	5	4	6	2	7
0029	5	4	4	5	3	6	2	7	1	8

Appendix D

1620 Character Coding

Alphameric Character	Input			Core Storage		Output		
	Typewriter	Tape	Card	Alpha	Num	Typewriter	Tape	Card
(Blank)	(Space)	C	(Blank)	C	C	(Space)	C	(Blank)
. (Period)	.	X0821	12, 3, 8	C	3	.	X0821	12, 3, 8
)	)	X0C84	12, 4, 8	C	4	)	X0C84	12, 4, 8
+	+	X0C	12	1	C	+	X0C	12
$	$	XC821	11, 3, 8	1	3	$	XC821	11, 3, 8
*	*	X84	11, 8, 4	1	4	*	X84	11, 4, 8
-(Hyphen)	-	X	11	2	C	-	X	11
/	/	0C1	0, 1	2	1	/	0C1	0, 1
, (Comma)	,	0C821	0, 3, 8	2	3	,	0C821	0, 3, 8
(	(	084	0, 4, 8	2	4	(	084	0, 4, 8
=	=	821	3, 8	3	3	=	821	3, 8
@	@	C84	4, 8	3	4	@	C84	4, 8
A-I	A-I	X0, 1-9	12, 1-9	4	1-9	A-I	X0, 1-9	12, 1-9
0 (-)	(None)	(None)	11, 0	5	C	-(Hyphen)	X	11, 0
J-R	J-R	X, 1-9	11, 1-9	5	1-9	J-R	X, 1-9	11, 1-9
1-9 (-)	J-R	X, 1-9	11, 1-9	5	1-9	J-R	X, 1-9	11, 1-9
S-Z	S-Z	0, 2-9	0, 2-9	6	2-9	S-Z	0, 2-9	0, 2-9
0 (+)	0	0	0 or 12, 0	7	C	0	0	0
1-9 (+)	1-9	1-9	1-9	7	1-9	1-9	1-9	1-9
‡	‡	082	0, 2, 8	C	C28	(Stop)	EOL	0, 2, 8
Numerical Character								
(Blank)	(Space)	C	(Blank)		C	0	0	0
0 (+)	0	0	0		C	0	0	0
0 (-)	0	X, X0C	11, 0		F	0	X	11, 0
1-9 (+)	1-9	1-9	1-9		1-9	1-9	1-9	1-9
1-9 (-)	1-9	X, 1-9	11, 1-9		1-9	1-9	X, 1-9	11, 1-9
‡	‡	082	0, 2, 8		C82	(Stop, WN) ‡ (DN)	EOL(WN) 082 (DN)	0, 2, 8
Num Blank †	@	C84	4, 8		C84	@	C84	(Blank)

† For Card Format Use Only

Appendix E

1620 Storage Register Functions

Register	Function
IR-1	Contains address of next instruction if machine is stopped with stop key or halt instruction.
IR-2	Saves return address when BT and BTM instructions are executed.
OR-1	Contains Q address after I-cycle of an instruction.
OR-2	Contains P address after I-cycle of an instruction.
OR-3	Retains address of low-order multiplier digit during multiplication.
PR-1	Saves return address when key operation occurs. Decremented for each new multiply digit during multiply.
PR-2	Decremented for each new multiplicand digit during multiply.
PR-3	Used to add partial product to each multiply cycle result.
MAR	Addresses core storage.
MBR	Receives digits entering or leaving core storage.
MDR	Receives addressed digit entering or leaving core storage.
Digit	Stores partial product during multiplication.
OP	Contains OP code of instruction just executed if machine is stopped with stop key or halt instruction.
Multiplier	Contains multiplier digits during multiply operation.
Sense & Branch	Contains I/O device code during input/output operations. Units positions used to develop each quotient digit during divide operation.
Digit & Branch	On some machines, combines functions of Digit, Sense & Branch Registers.

Appendix F

1620 Operating Modes

MODE	CONSOLE INDICATOR	MANUAL HEADINGS UNDER WHICH DESCRIBED
Automatic	Automatic Light	Input/Output Instructions; Automatic and Manual Lights
Manual	Manual Light	Branch Back; Automatic and Manual Lights
Alphameric	OP Code Display: 37, 39	Mode
Numerical	OP Code Display: 36, 38	Mode

Index