File No. 1620/1710-01
Form A26-5706-3

IBM Systems Reference Library

IBM 1620 Central Processing Unit, Model 1

This manual contains the basic programming and operating information required to use the 1620 Central Processing Unit, Model 1, as the computer element of the IBM 1620 Data Processing System and the IBM 1710 Control System. The text includes:

All program instructions for the 1620 Central Processing Unit (CPU) Model 1, including those for the following IBM units: 1311 Disk Storage Drive, 1443 Printer, 1621 Paper Tape Unit, 1622 Card Read-Punch, and 1627 Plotter. (Instructions peculiar to the 1710 System are explained in the 1710 manual referred to below.)

1620 Operator's Console Model 1, including the console input/output typewriter.

1620 8-track paper tape coding and IBM 80-column card coding.

Console operating procedures and program load routines.

The reader is referred to the following publications for details concerning the input/output and auxiliary units of the 1620 and 1710 Systems:

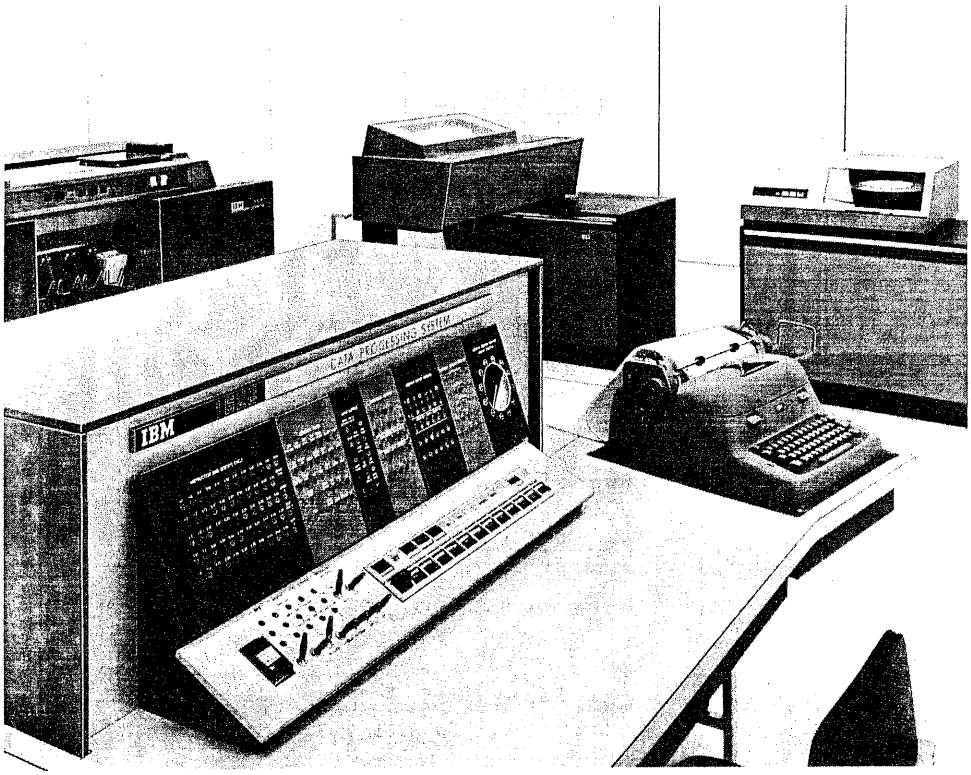
IBM 1311 Disk Storage Drive	A26-5991
IBM 1443 Printer	A26-5730
IBM 1621 Paper Tape Unit	A26-5836
IBM 1622 Card Read-Punch	A26-5835
IBM 1623 Core Storage	A26-5856
IBM 1627 Plotter	A26-5710
IBM 1710 Control System	A26-5709

This is a reprint of an earlier publication; it incorporates the following Technical Newsletters:

Form No.	Pages	Dated
N26-0116	24, 26, 46	4/1/65
N26-0130	80	7/19/65

Copies of this and other IBM publications can be obtained through IBM Branch Offices.
Comments concerning the contents of this publication may be addressed to:
IBM, Product Publications Department, San Jose, Calif. 95114

Contents


IBM

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. Actually, there are two 1620 Data Processing Systems, the 1620 Model 1 and the 1620 Model 2. This manual, as its title infers, is concerned with the 1620-1. Information on the 1620-2 is available in the IBM publication, A26-5781. Briefly, however, the 1620-2 consists of the 1620-2 Central Processing Unit (CPU) and the 1625 Core Storage Unit. The 1620-2, which uses the same input/output units as the 1620-1, possesses up to four times the computing and processing speeds of the 1620-1. The operating features of both systems are almost identical and with the exception of the 1620-2 special features, Index Registers and Binary Capabilities, complete programming compatibility exists between the two. Thus, a user whose initial requirements are within the range of the 1620-1 can, with a minimum of effort, transfer his expanding work load to the 1620-2.

The units of the 1620 System—and henceforth, we are concerned only with Model 1—are as follows:

The 1620 CPU is the controlling unit of the system; it contains arithmetic and logic circuitry, twenty-thousand positions of core storage, an operator's console, and a console input/output typewriter.

The 1621 Paper Tape Unit reads and punches 8-track paper tape.

The 1622 Card Read-Punch reads and punches 80-column cards.

The 1443 Printer provides up to 120 or 144 characters per line of printed output.

The 1627 Plotter provides a graphic output of digital data.

The 1623 Core Storage Unit provides 20,000 or 40,000 additional core storage positions for the CPU. All core storage data is available at random—no sequential searching for the desired data is required — and within microseconds (a microsecond, μsec, is one millionth of a second).

The 1311 Disk Storage Drive provides unlimited random access storage for the CPU. Information stored on removable disk packs is available within milliseconds (a millisecond (ms) is one thousandth of a second). Each disk pack has a storage capacity of 2,000,000 characters. The disk packs are easily interchanged by the operator.

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 6-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 uses a 2-address instruction format. 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 simplifies programming and reduces 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 look-up 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 special 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.

The operator's console includes control keys, switches, indicators, and an input/output typewriter. The control keys and switches are used for manual and/or automatic operation of the system. The indicators provide visual indication of the status of various registers, program indicators, and input/output units. The typewriter enables operator entry of data and instructions into core storage; it also provides a permanent log of operator intervention during program operation. As an output unit, the typewriter provides operator-oriented messages of job progress and detail.

Information is entered into the system via the following input units: 1621 Paper Tape Unit, 1622 Card Read-Punch, 1311 Disk Storage Drive, and the console typewriter. Maximum input speeds are:

1621—150 characters per second
1622 Model 1—250 cards per minute
1622 Model 2—500 cards per minute
1311—100 characters (one sector of data) per 2 ms. Seek time (the time required to move the read/write heads to the desired disk track) is 75 ms minimum, 250 ms average, and 392 ms maximum. This time is zero if the heads are already located at the proper disk track.
Console typewriter—operator's speed

The system provides output information via the 1622, the tape punch in the 1621, the console typewriter, the 1443 Printer, and the 1627 Plotter. (The 1621 and 1627 units cannot be installed on the same system except by special order.) Maximum output speeds are:

1443 Model 1—430 lines per minute
1443 Model 2—600 lines per minute
1621—15 characters per second
1622 Model 1—125 cards per minute
1622 Model 2—250 cards per minute
1627 Model 1—18,000 steps per minute (11 inch horizontal plot)
1627 Model 2—12,000 steps per minute (29½ inch horizontal plot)
Console typewriter—10 characters per second

The 1443 and 1622 units contain buffer storage media, which enables CPU operation concurrent with printing, card reading, and card punching. When printing and/or punching, the CPU transfers the output data from core storage to the 1443 print buffer and the 1622 punch buffer in 8.06 ms and 3.4 ms, respectively, and then continues program operation while the line is being printed and the card is being punched. Simultaneously, the data from a card in the 1622 read feed can be read into the read buffer. Following loading of the read buffer, the CPU transfers the card data to core storage and then continues processing while the next card is being read into the read buffer. Thus, reading, punching, and printing can be done simultaneously.

Detailed information concerning 1620 input/output units and special features is available in the IBM publications referred to on the cover page of this manual.

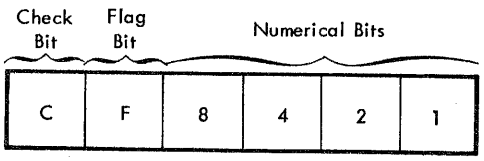
The 1620 CPU (Figure 1) contains the arithmetic and logic section of the system.

Data Representation

Data can be classified as digits, fields, or records, depending upon the operation in which the data is addressed.

Digits

BCD Bit Array. Each core storage position is addressable and can store one digit of information in 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 numeric bit positions. Only bit combinations whose sum is nine or less are used. A negative numeric expression has a sign flag in the units position of its field. Considering only the numeric bit positions, the decimal 6 is represented as 0110, the decimal 7 as 0111, etc., as shown in Figure 2.

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, when required, to each digit position as 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). Depending on its location and the operation performed, the flag bit is used in five ways:

1. *Sign Control*
   A numeric data field is negative if the units (low-order) digit contains a flag bit, and positive 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. *Field Mark*
   A flag bit as a field mark defines the leftmost (high-order) digit of a numeric 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 (Appendix B) 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 ($\bar{0}$).

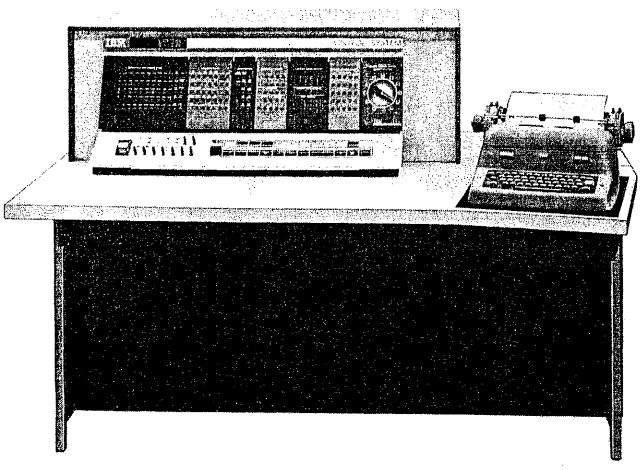


Figure 1. IBM 1620 Model 1 Central Processing Unit

	C	F	8	4	2	1
0	X					
1						X
2					X	
3	X				X	X
4				X		
5	X			X		X
6	X			X	X	
7				X	X	X
8			X			
9	X		X			X

Figure 2. BCD Digits 0-9

5. *Indirect Addressing (Special Feature)*
   A flag bit over the units position of an instruction address (P or Q) indicates an indirect address.

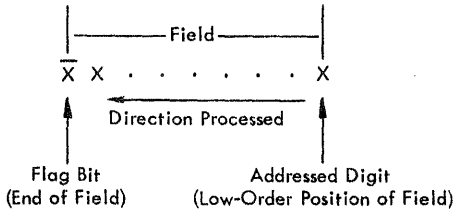
Record Mark (≠). The record mark is a nondecimal machine digit coded C-8-2. It is used primarily in input/output operations and in record transmission within the 1620; it cannot be used as a significant digit in an arithmetic or compare operation.

Group Mark (≠). The group mark, coded C-8-4-2-1, is used in disk storage operations to verify the correct length of records written on or read from disk storage.

Numerical Blank. The numerical blank (coded C-8-4) is used for format control of blank columns in card punching, and cannot be used in arithmetic or compare operations. The I/O instructions, Read Numerically and Write Numerically, further detail the use of the numerical blank.

Field

A field is composed of related digits that are treated as a unit of information (temperature, flow rate, etc.). The digits of a field are consecutively addressed. 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 which may or may not contain a flag (negative or positive) and the high-order digit containing the flag bit or field mark.

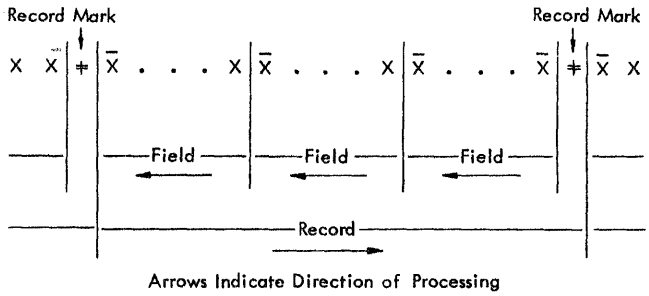


Record

A record consists of a field or fields of related data normally grouped for input/output operations and internal data 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.

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.



Arrows Indicate Direction of Processing

Magnetic Core Storage

A core storage module, which is 20,000 addressable positions of magnetic core storage, is located in the 1620. Two additional modules are available in the IBM 1623, Models 1 or 2, to increase the total core storage capacity of the 1620 System to 40,000 or 60,000 positions. Data and instructions in core storage are 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 before power is turned off.

Core Array

Each core storage module (20,000 positions) is made up of 12 core planes as shown in Figure 3. Each core plane contains all cores for a specific bit 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 in either the on condition, or the off

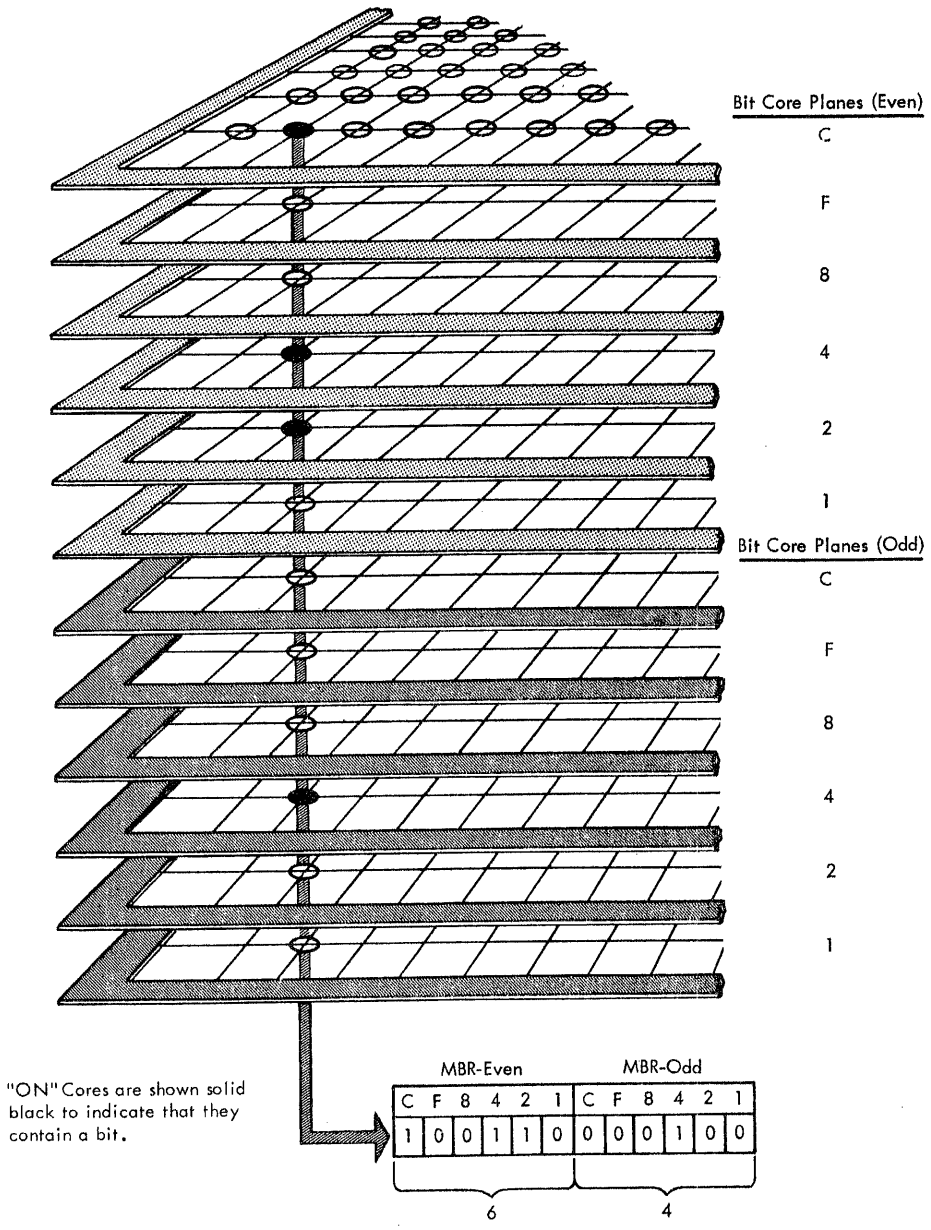


Figure 3. 1620 Core Storage Readout

condition. If on, the core contains a bit; if not, it is said to be off.

Two-Character Transfer

During a core storage readout cycle, which takes 20 μsec, the addressed core in each plane is read out, as illustrated by the vertical line in Figure 3. However, only cores that are on cause data to enter the 2-digit Memory Buffer Register (MBR) 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 Memory Address Register (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 on in

the even-digit planes. The 4 core is on in the odd-digit plane. Thus MBR-E and MBR-O contain 64.

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. 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. Address 00501 (odd) also 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.

Sequential Core Storage Addressing

Core storage positions are addressed sequentially from 00000 to the highest-numbered address of the core storage positions installed—19999, 39999, or 59999. Character transfer to, from, and within core storage embodies the "wrap around" principle, i.e., the highest-numbered address is followed by the lowest-numbered address when incrementing and the lowest-numbered address is followed by the highest-numbered address when decrementing. Thus, assuming a 20,000-position core storage capacity, the incrementing sequence is 19998, 19999, 00000, 00001, etc.; and the decrementing sequence is 00001, 00000, 19999, 19998, etc.

Character Representation

The 1620 can be programmed to read and write numeric and alphameric data. The input/output instruction (numeric or alphameric) determines whether data is read and/or written numerically or alphamerically.

Numerical Representation

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

Alphameric Representation

Two decimal digits are required in core storage to represent an alphameric character, i.e., an alphabetic character, a special character, or a numeric character. A 2-digit alphameric representation of numeric characters is provided to permit reading of mixed alphabetic, special and numeric characters without changing from an alphameric to a numeric instruction.

Figure 3 shows the bit configuration of a "U" if the 1620 is in the alphameric mode. The two alphameric digits of a character must occupy adjacent core storage positions, and the zone digit must occupy the even address. This storage requirement is satisfied by programming; alphameric read/write instructions must contain an odd-numbered P address. Figure 4 shows the zone and numeric digits that have been assigned to represent all the alphameric characters used in the 1620.

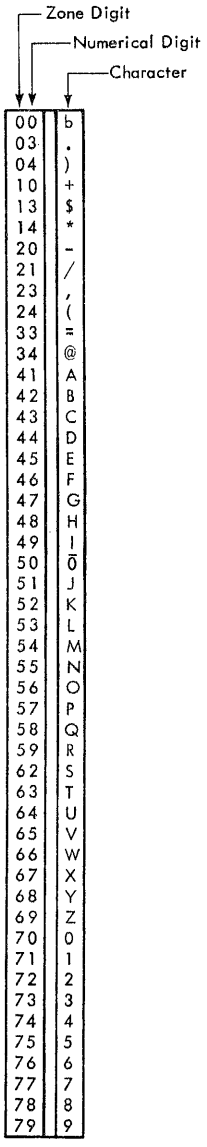


Figure 4. Alphameric Codes

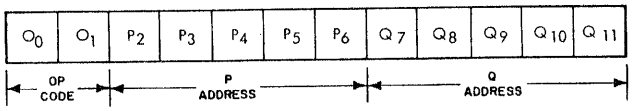
Stored Program Concept

The 1620 CPU 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 by an instruction placed in core storage to perform a specific operation. The programmer can select the most suitable operations, from various computer operations, to solve a problem or process data. 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 somewhere other than the next sequential position.

Instruction Characteristics

The 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. An instruction as it appears in core storage may be divided into O, P, and Q subscripted numbers, as follows:



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.

P Address

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

Q Address

The Q address specifies: (1) the location from which data is transmitted, (2) the indicator being interrogated, (3) the input/output device being used, or (4) the location of the numeric field in the Transfer Numerical Strip and Transfer Numerical Fill special feature instructions. Also, instruction modifier digits are placed in the Q address for those instructions with the same Op code number.

Instruction Execution Time

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 interpreted; control circuitry is established. The I-cycle always takes eight 20-microsecond machine cycles (160 microseconds).

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, size of the data fields, and signs of the fields (in arithmetic operations). The last E-cycle of an instruction is followed by the first I-cycle of the next instruction.

The formula for computing the total 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 (μsec) or milliseconds (ms), as noted.

Additional symbols used only in Load Dividend, Load Dividend Immediate, Divide, and Divide Immediate instructions are defined under *Execution Time*, following the explanation of the individual instruction.

Immediate Instructions

Certain arithmetic, internal data transmissions, compare, and branch instructions are labeled "immediate." Immediate instructions use the digits in the Q_7, Q_8, Q_9, Q_{10} and Q_{11} positions of the instruction as data instead of as a core storage address.

Thus, the Q data is located immediately within the instruction. For example, when the Transmit Field Immediate instruction, 16 00543 $\bar{1}$8765, is executed, the Q part of the instruction, $\bar{1}$8765, is transmitted to the P address (Figure 5). Data transfer begins at Q_{11} of the instruction, and continues until a flag bit is found, Q_7 in this case. If the flag bit was at Q_{10}, $\bar{6}5$ would be transferred to 00543. The difference between the Transmit Field and Transmit Field Immediate instructions can best be shown by comparing Figures 5 and 6. The Transmit Field instruction, Figure 6, tranfers the data at the Q address to the P addresses.

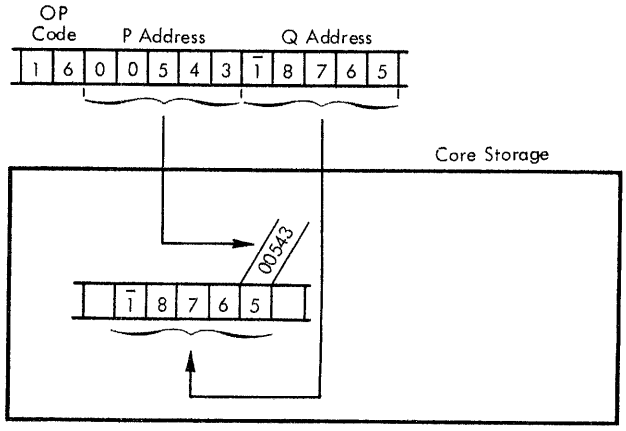


Figure 5. Transmit Field Immediate - Data Flow

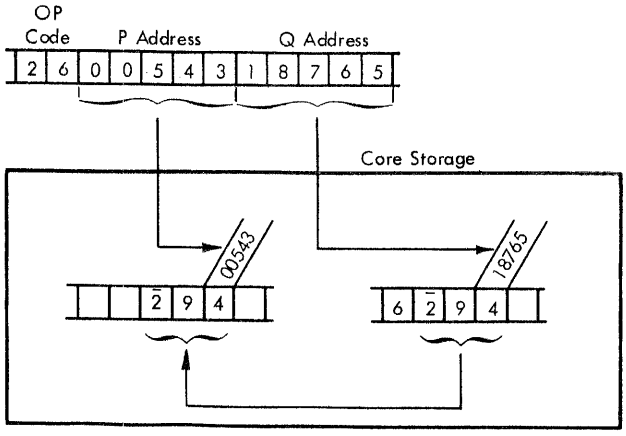


Figure 6. Transmit Field - Data Flow

The Indirect Addressing special feature saves program steps and computer time by providing a *direct* method of address modification. Its primary use is in programs where multiple 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 instead of the address of data. This "second address" is the core storage address of the data to be used unless the second address is yet another indirect address. In effect, the address at the indirect address location is a subsitute for the address of the instruction.

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

The P or Q address of an instruction is indirect when a flag bit is over the units position. Figure 7 shows that (1) the instruction (21 00500 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 00500 00650) is not altered in core storage; only the instruction register of the 1620 is 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. The address is then treated as a direct address.

Any P or Q address of an instruction that specifies the location of data can be an indirect address. See Table 1 in the next section of this manual. When the P address of an immediate instruction is an indirect address, the Q data cannot be more than six digits in

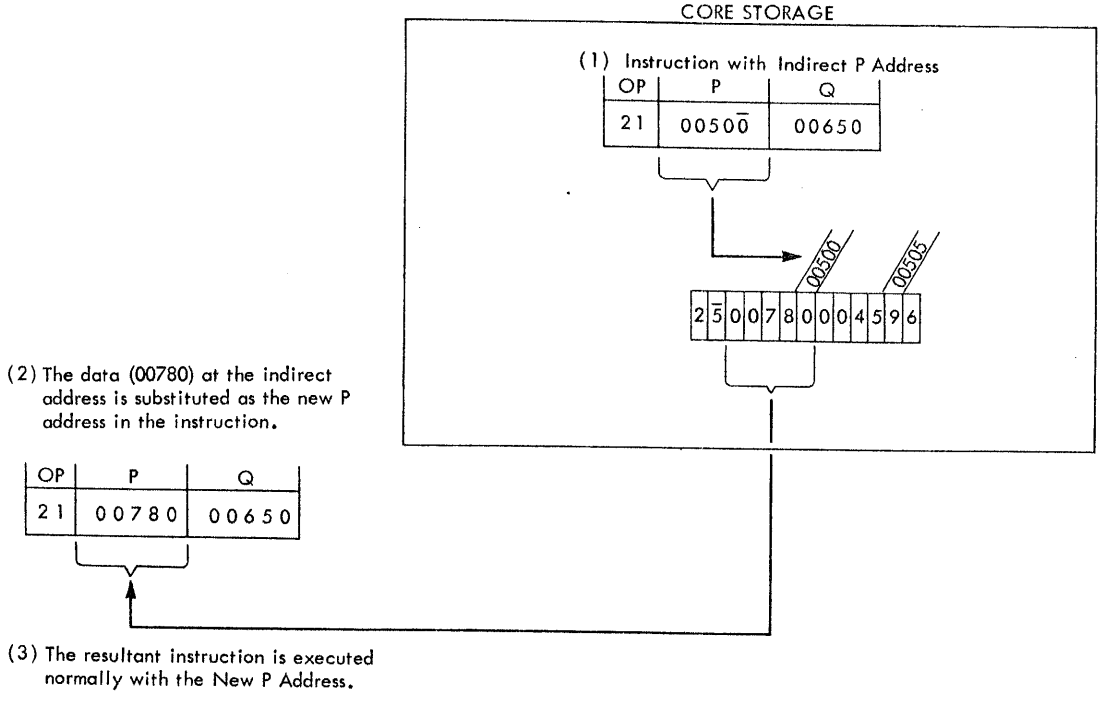


Figure 7. Indirect Addressing Data Flow

length because the flag bit over the units position of the P address also defines the end of the immediate data.

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

Examples

The Add instruction, 21 00500 00650, is shown in Figure 8 with both direct and indirect Q addresses. Line 1 shows direct addressing; the Q data is obtained from 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 9. The Q data 0̄00650, is added to the data at the address specified by the indirect P address. The result, 1̄155078, replaces the original P data, 1̄154428, at 09400.

A data flow diagram for a Branch instruction is shown in Figure 10. The first five digits at that indirect address are the address to which the computer branches for its next instruction.

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 8. Examples of Indirect Addressing

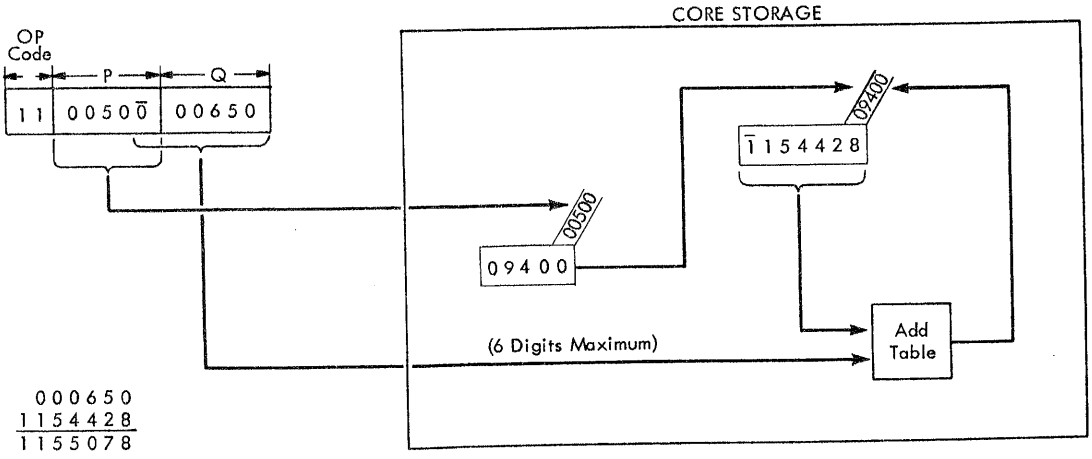


Figure 9. Indirect Addressing, Add Immediate Instruction

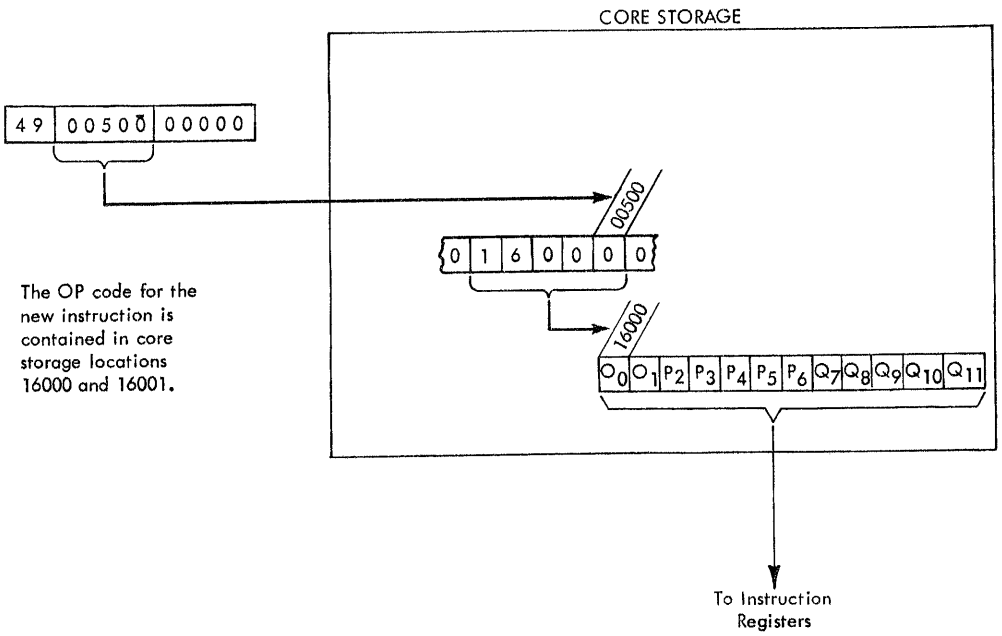


Figure 10. Indirect Addressing, Branch Instruction

Instruction Types

All 1620 instructions fall into six general categories according to function:

1. Arithmetic
2. Compare
3. Branch
4. Internal Data Transmission
5. Input/Output
6. Program Control

The Divide and Floating-Point instructions are 1620 Model 1 special features and must be ordered as such for inclusion in the 1620 Model 1 repertoire of instructions—Move Flag, Transfer Numerical Fill, and Transfer Numerical Strip.

All 1620 instructions with their associated Op codes, mnemonics, modifier digits, and allowable indirect addresses are shown in Table 1. Modifier digits are those required in Q_8, Q_9, and Q_{11} to differentiate between instructions having the same Op codes.

Arithmetic Instructions

Data flow, field length definition, indicator control, and sign analysis are common to all 1620 Arithmetic instructions, and are therefore explained before the actual instructions.

Table 1. 1620 Instructions

Instructions	Mnemonic	Code	P&Q	P
Arithmetic				
Add	A	21	X	
Add (I)	AM	11		X
Subtract	S	22	X	
Subtract (I)	SM	12		X
Multiply	M	23	X	
Multiply (I)	MM	13		X
Load Dividend*	LD	28	X	
Load Dividend (I)*	LDM	18		X
Divide*	D	29	X	
Divide (I)*	DM	19		X
Floating Add*	FADD	01	X	
Floating Subtract*	FSUB	02	X	
Floating Multiply*	FMUL	03	X	
Floating Divide*	FDIV	09	X	
Compare				
Compare	C	24	X	
Compare (I)	CM	14		X
Branch				
Branch	B	49		X
Branch No Flag	BNF	44	X	
Branch No Record Mark	BNR	45	X	
Branch No Group Mark*	BNG	55	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 (I)	BTM	17		X
Branch Back	BB	42		
Branch and Transmit Floating*	BTFL	07	X	

Instructions	Mnemonic	Code	P&Q	P
Internal Data Transmission				
Transmit Digit	TD	25	X	
Transmit Digit (I)	TDM	15		X
Transmit Field	TF	26	X	
Transmit Field (I)	TFM	16		X
Transmit Record	TR	31	X	
Transfer Numerical Strip*	TNS	72	X	
Transfer Numerical Fill*	TNF	73	X	
Floating Shift Right*	FSR	08	X	
Floating Shift Left*	FSL	05	X	
Transmit Floating*	TFL	06	X	
Input/Output				
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
Seek*	K	34		X
Program Control				
Control	K	34		
Set Flag	SF	32		X
Clear Flag	CF	33		X
Move Flag*	MF	71	X	
Halt	H	48		
No Operation	NOP	41		

* Special Feature
(I) Immediate

Data Flow and Field Length Definition

Data is read serially from right to left (low-order to high-order) until terminated by a flag bit defining the high-order position of the field. For example, where the data in a field is $\overline{2}85$, the dash (flag bit) over the high-order digit indicates a field mark. The *Program Testing* section of this manual provides more specific data flow information.

The minimum length of both the P and Q fields is two digits: a units digit which contains the sign, and at least one higher-order digit which is needed for field definition.

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.

High/Positive. The High/Positive (H/P) 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. The Equal/Zero (E/Z) 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.

Arithmetic Overflow (O'flow). The Arithmetic Check 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 (ten or more subtractions indicate the divisor is mispositioned).

The Arithmetic Check indicator is turned off by the execution of a Branch Indicator or Branch No Indicator instruction, or by manual pressing of the Reset key, and does *not* automatically turn on at the beginning of each Arithmetic instruction.

Table Look-up

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 Appendix B.) A digit with a flag bit in the table indicates that a carry is associated with that digit.

In addition, 20 positions, 00080 through 00099, are used to receive the product or partial product in Multiply operations.

Sign Analysis

Addition and Subtraction. The data in the Q field is either true-added or complement-added to the data in the P field. A true-add operation causes the Q data to be added just as it is. A complement-add operation causes the Q data to be altered before addition, as follows: the units digit is tens-complemented and the remaining higher-order digits are nines-complemented (95 becomes 05, 139 becomes 861, 2476 becomes 7524, etc.). The sign analysis chart in Figure 11 shows that (1) the Q data is complement-added during addition and subtraction when the signs of the P and Q fields are different and alike respectively, (2) if the Q field is complemented and if the value of the original Q data is higher than the value of the P data, the sum or difference is recomplemented, and (3) if a recomplement occurs, the original sign of the P field is changed.

For example:

Add $+\overline{1}5$ (Q data) to $-\overline{3}\overline{5}$ (P data).

According to the sign control analysis chart

1. The Q data is complement-added ($\overline{1}5$ becomes $\overline{8}5$), and $\overline{8}5 + \overline{3}\overline{5} = \overline{2}\overline{0}$. The hundreds carry is lost. A carry is always lost when it causes the sum or difference to exceed the size of the P field. Arithmetic Check is not turned on since the carry in this case indicates that recomplementing the P field is not required.
2. The sum is not recomplemented (15 is less than 35).
3. The sign of the P field is not changed since no recomplement occurred.

		ADD				SUBTRACT			
	Sign of P Field	+	+	−	−	+	+	−	−
	Sign of Q Field	+	−	+	−	+	−	+	−
1	True or Complement Add Q Field	True	Comp	Comp	True	Comp	True	True	Comp
2	Recomplement only if value of Q Field is greater than value of P Field		↑	↑		↑			↑
3	Change P Field sign only if recomplement occurs (changed sign shown).		−	+		−			+

Figure 11. Sign Control Chart

Multiplication and Division. The sign of each product and quotient is determined algebraically from the signs of its factors, as follows:

$$+a \times +b = +c$$
$$-a \times +b = -c$$
$$-a \times -b = +c$$
$$+a \div +b = +c$$
$$-a \div +b = -c$$
$$-a \div -b = +c, \text{ etc.}$$

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 12, the sum (14) is "looked up" in the Add table and replaces the 12 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 field with the larger value 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 H/P indicator is on if the sum is positive and not zero; the E/Z 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 numeric value of the Q field is greater than the absolute numeric value of the P field.

Basic Execution Time: $T = 160 + 80D_p$ μsec
Recomplement Time: $T = 80D_p$ μsec

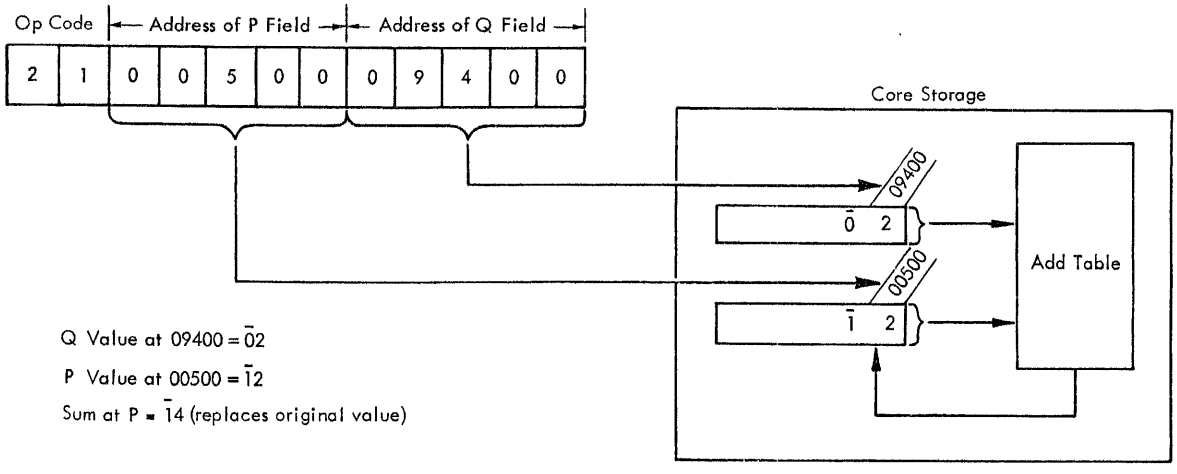


Figure 12. Add Instructions - Data Flow

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

Execution Time. Same as Add.

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.

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 H/P indicator is on if the difference is positive and not zero; the E/Z 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 instruction 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 numeric value of the field at the Q address is greater than the absolute numeric value of the field at the P address.

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.

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 is not changed by the operation.

In Figure 13, 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.

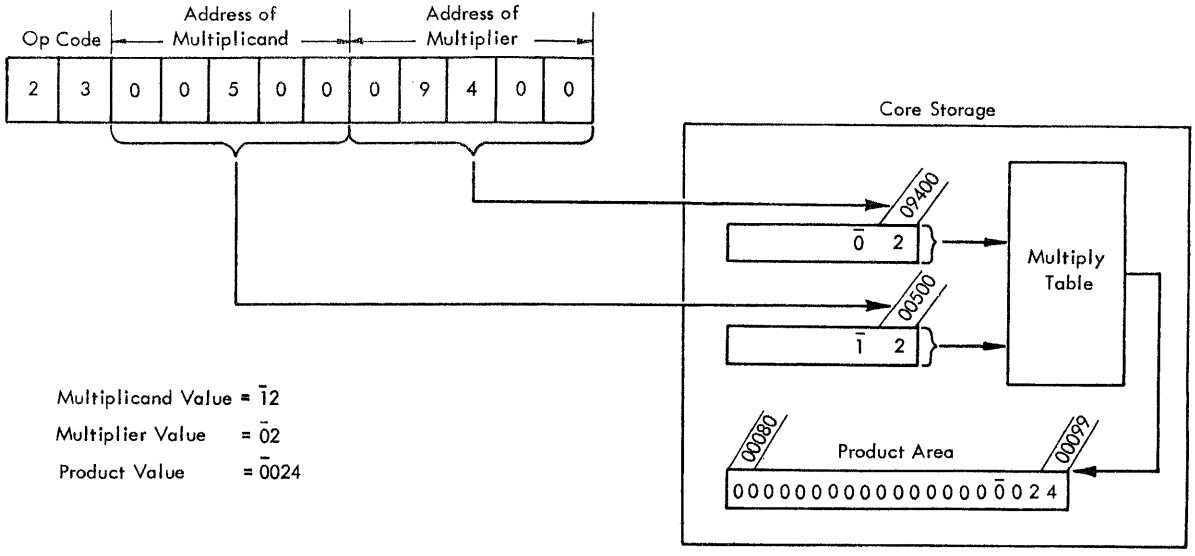


Figure 13. Multiply Instruction - Data Flow

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 the sign of the product 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 Arithmetic Check indicator is not turned on when the product exceeds 20 digits in length. The H/P indicator is on if the product is positive and not zero; the E/Z 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.

$$T = 560 + 40D_q + 168\ D_qD_p\ \mu sec$$

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. $T = 560 + 40D_q' + 168D_q'D_p$ μsec

The Automatic Division special feature simplifies programming and increases the processing speed of division problems by two to four times that of programmed routines. Only one command need be given. Four commands are provided, however, to facilitate positioning of the dividend and divisor in core storage. There are no practical limitations 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 are required for the quotient and remainder, 00074-00079 would have to be reset to zeros before the divide command was given.

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 formula for computing the total execution time follows the description of each instruction.

Load Dividend (LD-28)

Description. 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.

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 algabraic 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 14.

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.

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 14. Load Dividend Instructions

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.

Execution Time. T = 400 +40D_n μsec, where D_n equals the number of digits in the dividend.

Load Dividend Immediate (LDM-18)

Description. The description for Load Dividend applies except that the data in the Q part of the instruction is transmitted to the P address.

Execution Time. Same as a Load Dividend (LD-28).

Divide (D-29)

Description. 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 positions 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 15 shows the manner in which the 1620 solves this problem.

Problem 2: -212 ($\bar{2}1\bar{2}$) $\div \bar{2}4 = -8.83$ ($\bar{0}08\bar{8}\bar{3}$) and a remainder of $\bar{0}\bar{8}$. Figure 16 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 position of the Multiplier/Quotient register. An overdraw initiates a correction cycle (the divisor is added once), and the next subtraction occurs one place to the right.

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 sign 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 H/P indicator is on if the quotient is positive and not zero; the E/Z 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).

Execution Time. T = 160 + 520$D_v Q_t$ + 740 Q_t μsec. D_v and Q_t equal the number of digits in the divisor and quotient, respectively. The formula assumes an average quotient digit of 4.5. If a Load Dividend or Load Dividend Immediate instruction is used, the divide operation execution time may be considered as the total time for both the Load Dividend and Divide instructions.

$$T = 560 + 40D_n + 520D_vQ_t + 740Q_t \ \mu\text{sec}$$

Divide Immediate (DM-19)

Description. The description of Divide (D-29) applies except that the data in the Q part of the instruction is used as the divisor.

Execution Time. Same as Divide (D-29).

Decimal Point Location

The computer is unaware of decimal points, except for Automatic Floating-Point operations (special feature). Decimal point location for any given divide calculation is easily determined by simply subtracting the number of decimal digits in the divisor from the number of decimal digits in the dividend. The result is the number of decimal digits in the quotient. For example, if the divisor and dividend values in Problem 2 (Figure 16) are 2.4 and 21.200, the quotient value is 008.83 (3 — 1 = 2). Note that the original dividend 21.4, became 21.400 as a result of its placement by the Load Dividend instruction. Thus, the number of dividend decimal digits must include the zeros to the right of the loaded dividend.

Instruction	Data at Core Storage Addresses		Description	00092	00093	00094	00095	00096	00097	00098	00099
28 00099 00500	00500 $\bar{4}906$	00600 $\bar{2}3$	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	$\bar{4}$		
			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 15. 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	$\bar{0}$	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	$\bar{0}$	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	8	2	0	0	$\bar{0}$
	8 successful subtractions (8 x 24 = 192)									− 1 9 2			
										0 0 8			
	(Overdraw and correction not shown)												
	Store quotient digit (8)			0	0	0	$\bar{0}$	0	8	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	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 16. Divide, Problem 2

Incorrect Divisor Positioning

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

Overflow. As illustrated in Figure 17, an incorrectly positioned divisor can cause more than nine successful subtractions and an incorrect quotient. The divide operation is terminated, the Arithmetic Check indicator and light are turned on, but processing does not stop unless the Overflow Check switch is set to STOP. Note the absence of a field-length flag in position 00095 when division is terminated. The flag is not placed automatically because the first quotient digit, which normally causes the flag bit to be generated and stored, is not achieved.

If, after a division overflow, the field remaining in the product area is to be used for further operations, the program must provide for a flag to be set in the desired position.

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 18 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.

Incorrect Termination. If the P address is less than 10000, i.e., between 00100 and 09999, the divide operation will terminate when a subtraction occurs at 0XX99. This, in effect, restricts the size of the dividend to 10,020 digits, if only 20,000 positions of core storage are installed.

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	2	0	$\bar{0}$

Figure 17. Divide Overflow

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 18. Division Error, Incorrect Programming

Summary of Automatic Division Rules

1. Decimal point location: the number of dividend decimal digits minus the number of divisor decimal digits equals the number of quotient decimal digits (decimal digits are those digits to the right of the assumed decimal point).
2. Dividend position (P address of LD instruction): determined by the number of quotient digits desired. An anaylsis of the problem and its relationship to these rules is necessary.
3. First subtraction (P address of divide instruction): 00100 minus the length of the quotient (the minimum length of the quotient is generally the length of the dividend).
4. Quotient length: 100 minus the P address (minimum of 2).
5. Quotient address: 00099 minus the length of the divisor.
6. Remainder length: same as divisor length.
7. Remainder address: 00099 (assuming dividend at 00099).
8. Sign of quotient: determined by algebraic signs of dividend and divisor.
9. Sign of remainder: same as sign of dividend.

This special feature provides the 1620 with the ability to do floating-point arithmetic, using floating-point instructions instead of program sub-routines.

The use of automatic floating-point operations can result in a 50 to 100 per cent increase in the computing power of the 1620 CPU, depending on the amount of floating-point computations required. Also, up to 15 per cent of the basic 1620 core-storage capacity can be saved through the elimination of subroutines and call sequence instructions associated with Floating Add, Floating Subtract, Floating Multiply, and Floating Divide.

The Automatic Division special feature is a prerequisite to the installation of Automatic Floating Point Operations.

Floating-Point Arithmetic

Scientific and engineering computations frequently involve lengthy and complex calculations in which it is necessary to manipulate numbers that may vary widely in magnitude. To obtain a meaningful answer, problems of this type usually require that as many significant digits as possible be retained during calculation and that the decimal point always be properly located. When applying such problems to a computer, several factors must be taken into consideration, the most important of which is decimal point location.

Generally speaking, a computer does not recognize the decimal point present in any quantity used during the calculation. Thus, a product of 414154 will result regardless of whether the factors are 9.37 x 44.2, 93.7 x .442, or 937 x 4.42, etc. It is the programmer's responsibility to be cognizant of the decimal point location during and after the calculation and to arrange the program accordingly. In a floating-add operation, for example, the decimal point of all numbers must be lined up to obtain the correct sum. To facilitate this arrangement, the programmer must shift the quantities as they are added. In the manipulation of numbers that vary greatly in magnitude, the resulting quantity could conceivably exceed allowable working limits.

The processing of numbers expressed in ordinary form, e.g., 427.93456, 0.0009762, 5382, —623.147, 3.1415927, etc., can be accomplished on a computer only by extensive anaylsis to determine the size and range of intermediate and final results. This anaylsis and subsequent number scaling frequently takes longer than does the actual calculation. Furthermore, number scaling requires complete and accurate information as to the boundaries of all numbers that come into the computation (input, intermediate, output). Since it is not always possible to predict the size of all numbers in a given calculation, anaylsis and number scaling are sometimes impractical.

To alleviate this programming problem, a system must be employed in which information regarding the magnitude of all numbers accompanies the quantities in the calculation. Thus, if all numbers are represented in some standard, predetermined format which instructs the computer in an orderly and simple fashion as to the location of the decimal point, and if this representation is acceptable to the routine doing the calculation, then quantities which range from minute fractions having many decimal places to large whole numbers having many integer places can be handled. The arithmetic system most commonly used, in which all numbers are expressed in a format having the above feature, is called "floating-point arithmetic."

The notation used in floating-point arithmetic is basically an adaptation of the scientific notation widely used today. In scientific work, very large or very small numbers are expressed as a number, between one and ten, times a power of ten. Thus 427.93456 is written as 4.2793456×10^2 and 0.0009762 as 9.762×10^{-4}. In the 1620 floating-point arithmetic system, the range of numbers is modified to extend between .1 and .99999999; that is, the decimal point of all numbers is placed to the left of the high-order (leftmost) nonzero digit. Hence, all quantities may be thought of as a decimal fraction times a power of ten (e.g., 427.93456 as $.42793456 \times 10^3$ and 0.0009762 as $.97620000 \times 10^{-3}$) where the fraction is called the *mantissa*, and the power of ten, used to indicate the number of places the decimal point was shifted, the *exponent*. In addition to the advantages inherent in scientific notation, the use of floating-point numbers during processing eliminates the necessity of analyzing the operations to determine the positioning of the decimal point in intermediate and final results, since the decimal point is always immediately to the left of the high-order nonzero digit in the mantissa.

1620 Automatic Floating-Point Operations

In 1620 Automatic Floating-Point Operations, a floating-point number is a field consisting of a variable length mantissa and a 2-digit exponent. The exponent is in the two low-order positions of the field, and the mantissa is in the remaining high-order positions, as shown:

$$\overline{M}. \ . \ . \ . \ M\overline{E}E$$

The mantissa must have a minimum of two digits and can have a maximum of 100 digits. However, when two fields are operands (quantities being added, subtracted, multiplied, divided), they must have mantissas of the same length. The extremity of the field is marked by a flag over the high-order digit.

The exponent is established on the premise that the mantissa is less than 1.0 and equal to or greater than 0.1. The exponent is always two digits and has a range of −99 to +99. The length of the exponent field is defined by a flag over the high-order (tens) digit.

The mantissa and the exponent each have an algebraic sign represented by the presence (negative) or absence (positive) of a flag over the units position. A floating-point number with a negative mantissa and a negative exponent is represented as follows:

$$\overline{M}. \ . \ . \ \overline{M}\overline{E}\overline{E}$$

Sign control of the results of all computations is maintained according to the standard rules of arithmetic operations.

Eight floating-point instructions are provided: four are for arithmetic computations — Floading Add, Floating Subtract, Floating Multiply, and Floating Divide; three are used to control field size and location — Floating Shift Right, Floating Shift Left, and Transmit Floating. The eighth instruction provides for Branch and Transmit in floating-point operations. All instructions are in the 1620 format of a 2-digit Op code, 5-digit P address, and 5-digit Q address. Floating-point instructions depend on the presence of flags over the high-order digits of the mantissa and exponent. Therefore they should be used only with data in the floating-point format.

As an aid to the programmer or operator in checking program logic and computation results, the operation of the computer in aligning decimal points, normalizing results, etc., is described with each instruction. These operations are automatic and need not be programmed. Of particular note is Floating Divide, which requires only one instruction; the dividend is positioned, division is accomplished, and the quotient is transmitted to the P field without further command.

In descriptions of instructions and operations, the following symbols are used for clarity and brevity:

Mp = mantissa of the field at the P address (P)
Mq = mantissa of the field at the Q address (Q)
Ep = exponent of the field at the P address
Eq = exponent of the field at the Q address
L = number of digits in the mantissa
d = Ep − Eq

In all floating-point numbers, the decimal point is assumed to be at the left of the high-order digit, which must not be zero. Such a number is referred to as "normalized." When a number has one or more high-order zeros, it is considered to be "unnormalized." An unnormalized number resulting from a floating-point computation is normalized automatically, but unnormalized terms are not recognized as such when entered as data. They will be processed but correct results cannot be assured. Therefore, it is necessary that all data be entered in normalized form. For example, the number $\overline{0}68234940\overline{5}$ should be entered as $\overline{6}8234940\overline{0}4$, assuming the fixed-point number is 6823.494, and an 8-digit mantissa is required.

With the exception of Floating Shift Right and Floating Shift Left, the P address and Q address of floating point fields are the addresses of the low-order positions of the exponents.

Floating Add (FADD-01)

Description. Mq is added to Mp and the result replaces Mp. Mp and Eq are not altered in core storage. Dependent on L and the value of d, the appropriate field is shifted to align decimal points before addition is performed. If d = 0, no shift is made (Figure 19).

If d is greater than zero and less than L, in effect, Mq is shifted d positions to the right before being added to Mp. The number of low-order digits of Mq equal to d are truncated as the shift is made (Figure 20). If d is less than zero, and the absolute value

Core Storage Locations 01590 ← → 01599				Instruction			Core Storage Locations 01590 ← → 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
$\overline{1}$ 2 3	$\overline{0}$ 4	$\overline{7}$ 8 9	$\overline{0}$ 4	0 1	0 1 5 9 4	0 1 5 9 9	$\overline{9}$ 1 2	$\overline{0}$ 4	$\overline{7}$ 8 9	$\overline{0}$ 4

Figure 19. Addition Without Mp or Mq Shift

Core Storage Locations 01590 ⟷ 01599				Instruction			Core Storage Locations 01590 ⟷ 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
1 2 3	0 2	7 8 9	0 1	0 1	0 1 5 9 4	0 1 5 9 9	2 0 1	0 2	7 8 9	0 1

Figure 20. Mq Shifted Right to Align Decimal Points

Core Storage Locations 01590 ⟷ 01599				Instruction			Core Storage Locations 01590 ⟷ 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
9 8 7	0 4	4 5 6	0 4	0 1	0 1 5 9 4	0 1 5 9 9	1 4 4	0 5	4 5 6	0 4

Figure 24. Mantissa Overflow, Number Normalized

of d is less than L, Mp is shifted | d | positions to the right before Mq is added to Mp. The number of low-order digits of Mp equal to | d | are truncated as the shift is made. Eq replaces Ep (Figure 21). If d is plus and equal to or larger than L, Mp is above the range of Mq and no addition is performed (Figure 22). If d is less than zero and | d | is equal to or greater than L, Mq is above the range of Mp, and no addition is performed. Mq replaces Mp, and Eq replaces Ep (Figure 23).

Core Storage Locations 01590 ⟷ 01599				Instruction			Core Storage Locations 01590 ⟷ 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
1 2 3	0 1	7 8 9	0 2	0 1	0 1 5 9 4	0 1 5 9 9	8 0 1	0 2	7 8 9	0 2

Figure 21. Mp Shifted Right to Align Decimal Points

Core Storage Locations 01590 ⟷ 01599				Instruction			Core Storage Locations 01590 ⟷ 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
1 2 3	0 5	7 8 9	0 2	0 1	0 1 5 9 4	0 1 5 9 9	1 2 3	0 5	7 8 9	0 2

Figure 22. Mp Above Range of Mq

Core Storage Locations 01590 ⟷ 01599				Instruction			Core Storage Locations 01590 ⟷ 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
1 2 3	0 1	7 8 9	0 3	0 1	0 1 5 9 4	0 1 5 9 9	7 8 9	0 3	7 8 9	0 3

Figure 23. Mq Above Range of Mp

After addition has been completed, the number of Mp digits is checked to determine if it exceeds L. If so, this is an overflow condition; the low-order digit of Mp is truncated, and the mantissa is shifted one position to the right. A one is entered in the high-order position of the mantissa, and a one is added to Ep (Figure 24). When an overflow does not exist, Mp is scanned for zeros beginning with the high-order position. High-order zeros are counted (z), and Mp is shifted z positions to the left; vacated positions are set to zeros. Flag bits in Mp are not altered or moved. Eq − z replaces Ep (Figure 25).

Core Storage Locations 01590 ⟷ 01599				Instruction			Core Storage Locations 01590 ⟷ 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
1 2 3	0 1	1 1 9	0 1	0 1	0 1 5 9 4	0 1 5 9 9	4 0 0	0 1	1 1 9	0 1

Figure 25. High-Order Zeros, Number Normalized

Execution Time. T (average) = 400 + 100L μsec. If the result is recomplemented, add 80L μsec.

Floating Subtract (FSUB-02)

Description. The floating subtract operation is the same as the floating add operation except that sign control procedures for Mq are reversed.

Execution Time. Same as Floating Add (FADD-01).

Floating Multiply (FMUL-03)

Description. Mp is multiplied by Mq, and the result replaces Mp. Ep is added to Eq, and the sum replaces Ep. Mp and Ep are normalized, as required, after multiplication. Mq and Eq are not altered in core storage. The product is formed in the product area, beginning at 00099 and extending through lower-numbered core storage positions to 00100 − 2L. The product area, 00080-00099, is cleared automatically prior to multiplication. If L is greater than 10, the program must provide for clearing positions 00100 − 2L through 00079. After multiplication, the digit at position 00100 − 2L is tested for zero. If the digit is other than a zero, the field at 00099 − L replaces Mp (Figure 26). If the digit tested is a zero, the field at 00100 − L replaces Mp and Ep + Eq − 1 replaces Ep (Figure 27).

Core Storage Locations 01590 ⟷ 01599				Instruction			Core Storage Locations 01590 ⟷ 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
$\bar{7}$ 8 9	$\bar{0}$ 3	$\bar{4}$ 5 6	$\bar{0}$ $\bar{1}$	0 3	0 1 5 9 4	0 1 5 9 9	$\bar{3}$ 5 9	$\bar{0}$ $\bar{2}$	$\bar{4}$ 5 6	$\bar{0}$ $\bar{1}$

Figure 26. Product Equal to 2L

Execution Time. T (average) = 1120 + 80L + $168L^2$ μsec.

Core Storage Locations 01590 ⟷ 01599				Instruction			Core Storage Locations 01590 ⟷ 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
$\bar{1}$ 2 3	$\bar{0}$ 2	$\bar{4}$ 5 6	$\bar{0}$ 4	0 3	0 1 5 9 4	0 1 5 9 9	$\bar{5}$ 6 0	$\bar{0}$ 5	$\bar{4}$ 5 6	$\bar{0}$ 4

Figure 27. Product Less than 2L

Floating Divide (FDIV-09)

Description. Mp is divided by Mq, and the quotient replaces Mp. Eq is deducted from Ep, and the result replaces Ep. Mp and Ep are normalized, as required, after division; Mq and Eq are not altered in core storage. The quotient and remainder are developed in the product area, beginning at 00099 and extending through lower-numbered core storage positions to 00100 — 2L. The product area, 00080-00099, is cleared automatically prior to division. If L is greater than 10, the program must provide for clearing positions 00100 — 2L through 00079. Prior to division, the absolute values of Mp and Mq are compared. If |Mp| is equal to or greater than |Mq|, Mp is transmitted to 00100 — L, and division is performed, starting at 00100 — L, according to the procedure for automatic division. The quotient at 00099 — L replaces Mp, and Ep — Eq + 1 replaces Ep (Figure 28). If |Mp| is less than |Mq|, Mp is transmitted to 00099 — L; division starts in 00100 — L, and proceeds according to the procedure for automatic division. The quotient at 00099 — L replaces Mp, and Ep — Eq replaces Ep (Figure 29).

Core Storage Locations 01590 ⟷ 01599				Instruction			Core Storage Locations 01590 ⟷ 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
$\bar{7}$ 8 9	$\bar{0}$ 4	$\bar{1}$ 2 3	$\bar{0}$ 1	0 9	0 1 5 9 4	0 1 5 9 9	$\bar{6}$ 4 1	$\bar{0}$ 4	$\bar{1}$ 2 3	$\bar{0}$ 1

Figure 28. Divisor Equal to or Less than Dividend

Core Storage Locations 01590 ⟷ 01599				Instruction			Core Storage Locations 01590 ⟷ 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
$\bar{1}$ 2 3	$\bar{0}$ 1	$\bar{7}$ 8 9	$\bar{0}$ 4	0 9	0 1 5 9 4	0 1 5 9 9	$\bar{1}$ 5 5	$\bar{0}$ $\bar{3}$	$\bar{7}$ 8 9	$\bar{0}$ 4

Figure 29. Divisor Greater than Dividend

Division by zero causes the Arithmetic Check indicator (code 14) to be turned on. Mp is not altered, but Ep is replaced by Ep — Eq.

Execution Time. T = 880 + 940L + $520L^2$ μsec. The formula is based on an average quotient digit of 4.5.

Floating Shift Right (FSR-08)

Description. The field at the Q address (the portion of the mantissa to be retained) is shifted right to the location specified by the P address. The exponent is not moved or altered. The effect of this instruction is to shrink the mantissa by shifting it to the right and truncating the low-order digits. The P address is normally the units position of the mantissa; the digit at the Q address becomes the new low-order digit of the mantissa. Vacated high-order positions are set to zeros. An existing flag bit at the P address is retained for algebraic sign; the field flag bit is transmitted with the high-order digit of the Q field and terminates the operation (Figure 30).

The P address should always be greater than the Q address; a P address less than or equal to the Q address will cause errors.

Execution Time. T = 200 +40L.

Core Storage Locations 01590 ⟷ 01599				Instruction			Core Storage Locations 01590 ⟷ 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
0 $\bar{1}$ 2	$\bar{0}$ 2	$\bar{7}$ 8 9	$\bar{0}$ 5	0 8	0 1 5 9 7	0 1 5 9 6	0 $\bar{1}$ 2	$\bar{0}$ 2	0 $\bar{7}$ 8	$\bar{0}$ 5

Figure 30. Floating Shift Right

Floating Shift Left (FSL-05)

Description. The field at the Q address, which is the low-order position of the mantissa, is shifted left so that the high-order digit is moved to the location specified by the P address. The exponent is not moved or altered. The effect of this instruction is to expand the mantissa by shifting it to the left and filling the vacated positions with zeros. It is important to note that the Q address is the low-order position of the field moved, and the P address is the high-order

position of the resulting field. An existing flag bit at the Q address is retained for algebraic sign; the field flag bit is transmitted with the high-order digit of the Q field (Figure 31).

Core Storage Locations 01590 ← → 01599				Instruction			Core Storage Locations 01590 ← → 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
$\bar{1}$ 2 3	$\bar{0}$ 2	0 7 $\bar{8}$	$\bar{0}$ 5	0 5	0 1 5 9 5	0 1 5 9 7	$\bar{1}$ 2 3	$\bar{0}$ 2	$\bar{7}$ 8 $\bar{0}$	$\bar{0}$ 5

Figure 31. Floating Shift Left

If the mantissa is expanded to a length greater than 2L, any extraneous flag bits in core storage positions between the old high-order position and the new low-order position of the mantissa must be cleared before the FSL instruction is given. Therefore, if Q — P is equal to or greater than 2L, locations P + L through Q — L must be free of flags.

Contrary to other instructions in the floating-point series, FSL is executed in the transmit record manner of transmitting individual digits in the high-order to low-order sequence. After the units digit has been transmitted, the positions of the expanded mantissa are set to zero, in ascending core storage location sequence. After each position is set to zero, the succeeding position is checked for a flag bit. If the fraction is positive, the flag bit is assumed to be the high-order position of the exponent and the operation stops without altering the flag bit position. If the fraction is negative, the flag bit is assumed to be the units position of the fraction, and a negative zero is inserted in the units position before the operation stops. Thus, a flag bit detected prior to the previous high-order position of the mantissa stops the operation and results in an incorrect mantissa.

For example, if P = 01590, Q = 01601, and L = 4, core storage locations 01590 through 01603, with an extraneous flag bit in 01596, appear as follows:

$$XXXXXXX\bar{X}X\bar{M}MMM\bar{E}E$$

After transfer of the mantissa, but before the zero-fill operation, the core storage locations appear as follows (not that the flag bit in 01598 has been cleared):

$$\bar{M}MMMXXX\bar{X}XMMMM\bar{E}E$$

Upon completion of the operation, the mantissa is incorrect, as follows:

$$\bar{M}MMM00\bar{X}XMMMM\bar{E}E$$

If 01596 had not contained a flag bit, the mantissa would have been expanded correctly, as follows:

$$\bar{M}MMM00000000\bar{E}E$$

Execution Time. T = 200 + 40L + 40L′ μsec. (L′ = length mantissa is increased by shift.)

Transmit Floating (TFL-06)

Description. The field at the Q address is transmitted to the location designated by the P address. Mq and Eq are not altered in core storage. The Q address is normally the low-order position of the exponent, and the operation is the same as the regular Transmit Field instruction (TF-26), except that flag bits in the three low-order positions are ignored as indications to terminate the transmission. Beginning with the fourth low-order digit, a flag bit terminates the operation. All flag bits in the field are transmitted (Figure 32).

Execution Time. T = 240 + 40L.

Core Storage Locations 01590 ← → 01599				Instruction			Core Storage Locations 01590 ← → 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
$\bar{1}$ 2 3	$\bar{0}$ 2	$\bar{7}$ 8 9	$\bar{0}$ 5	0 6	0 1 5 9 4	0 1 5 9 9	$\bar{7}$ 8 9	$\bar{0}$ 5	$\bar{7}$ 8 9	$\bar{0}$ 5

Figure 32. Transmit Floating

Branch and Transmit Floating (BTSF-07)

Description. The address of the next instruction is saved in IR-2, and the field at the Q address is transmitted to the P address minus one. The instruction at the P address is the next one executed. Mq and Eq are not altered in core storage. The Q address is normally the low-order position of the exponent. The operation is the same as the regular Branch and Transmit instruction (BT-27), except that in the transmit function the three low-order position flags are ignored as indications to terminate the transmission. Beginning with the fourth low-order position, a flag bit terminates the operation. All flag bits are transmitted.

Execution Time: T = 280 + 40L μsec.

Mantissa and Exponent Analysis

Zero Mantissa

When a floating-point computation results in a zero mantissa, a special floating-point zero is created in the

form $\bar{0}0$ . . . . $0\bar{9}\bar{9}$, which is the smallest positive quantity that can be represented (Figure 33). A zero mantissa causes the E/Z indicator (12) to be turned on. Zeros entered as data should be in floating-point zero form. Zero quantities in other forms, e.g., $\bar{0}0$ . . . . $0\bar{0}0$ will be processed, but results cannot be assured.

Core Storage Locations 01590 ← → 01599				Instruction			Core Storage Locations 01590 ← → 01599			
Before							After			
Mp	Ep	Mq	Eq	OP	P	Q	Mp	Ep	Mq	Eq
$\bar{7}$ 8 9	$\bar{0}$ 5	$\bar{7}$ 8 9	$\bar{0}$ 5	0 2	0 1 5 9 4	0 1 5 9 9	$\bar{0}$ 0 0	$\bar{9}$ $\bar{9}$	$\bar{7}$ 8 9	$\bar{0}$ 5

Figure 33. Zero Mantissa

Indicators

The four indicators associated with automatic floating-point operations are represented by lights on the 1620 console. The light for each indicator is turned on when the corresponding indicator is turned on. The H/P and E/Z lights are located in the Control Gates section of the console, and the Arithmetic Check and Exponent Check lights and Overflow switch are in the Indicator Displays and Switches section (Figure 34).

High/Positive (11). The H/P indicator and light are turned on when the mantissa resulting from a floating-point computation is greater than zero.

Equal/Zero (12). The E/Z indicator and light are turned on to indicate a zero mantissa resulting from a floating-point computation.

Arithmetic Check (14). During floating-point operations, the Arithmetic Check indicator is turned on when division is attempted by zero. Division by an unnormalized number may result in an incorrect quotient through incorrect positioning of the divisor.

Exponent Check (15). The exponent Check indicator is turned on by exponent overflow or underflow.

Exponent Overflow

When an exponent greater than +99 is generated, the mantissa is set to nines. The sign is determined by the computed result that caused the overflow. The exponent is set to +99. This is the largest floating-point number ($\bar{9}9$ . . . . $9\bar{9}9$) that can be represented. If the generated mantissa is positive, the H/P indicator (11) is also turned on.

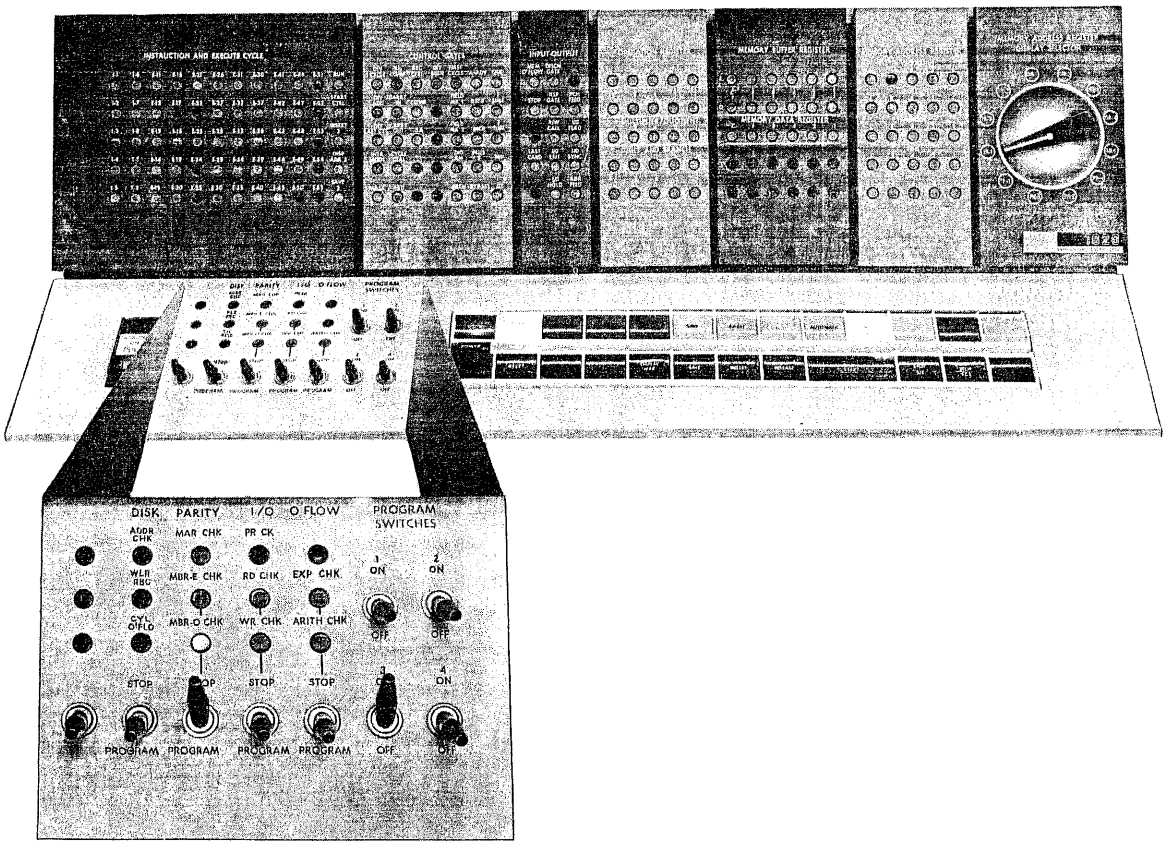


Figure 34. Indicators and Switches on 1620 Console

Exponent Underflow

When an exponent less than –99 is generated, the mantissa is set to plus zeros, and the exponent is set to –99. This is the smallest floating-point number ($\bar{0}0$ . . . . $0\bar{9}\bar{9}$) that can be represented. The E/Z indicator is also turned on.

An exponent underflow is *not* indicated when one or both operands are zero.

When the Exponent Check indicator (15) is turned on, program operation is controlled by the console Overflow Check switch, which is also connected to the Arithmetic Check indicator (14). The Exponent Check indicator (15) is turned off by programmed interrogation or by pressing the 1620 Reset key.

MARS Display Selector (1620 Console)

Operation Register 4. OR-4 is used to store and control the address of Eq.

Operand Register 5. OR-5 is used to store and control the address of Ep.

Counter Register 1. CR-1 is used to store the algebraic difference between Ep and Eq for determination of decimal alignment. It is also used to count high-order zeros when normalizing–the contents of CR-1 are subtracted from Ep.

Compare Instructions

Although Compare instructions are arithmetic in nature, they perform a distinctly logical function.

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 data from the P data and by discarding the digits of the difference. Neither field is altered. The result of the comparison is shown by the on-off condition of the indicators.

Condition (Algebraic)	Indicators		
	High/Positive	Equal/Zero	H/P or E/Z
P Greater than Q	ON	OFF	ON
P Less than Q	OFF	OFF	OFF
P Equal to Q	OFF	ON	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 H/P indicator is turned on if the P address data is algebraically higher than the Q address data, and off, if not higher. The E/Z indicator is turned on if the P address data is algebraically equal to the Q address data, and off, if not equal.

In Figure 35 the Q address data (12) is subtracted (complement-added) 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 Arithmetic Check 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 terminates.

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 E/Z indicator is turned on.

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

1. Numerical Sequence:
   0 1 2 3 4 5 6 7 8 9
2. Alphameric Sequence:
   b (blank) .) + $ * - / , (= @ A B C D E F G H I $\bar{0}$ (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 (≠), group mark (≢), and numeric 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).

$$T = 160 + 80D_P \ \mu sec$$

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.

$$T = 200 + 80D_z \ \mu sec$$

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.

Branch Instructions

All branch instructions except Branch Back (BB-42) 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 performed, depending on the condition tested.

Branch (B-49)

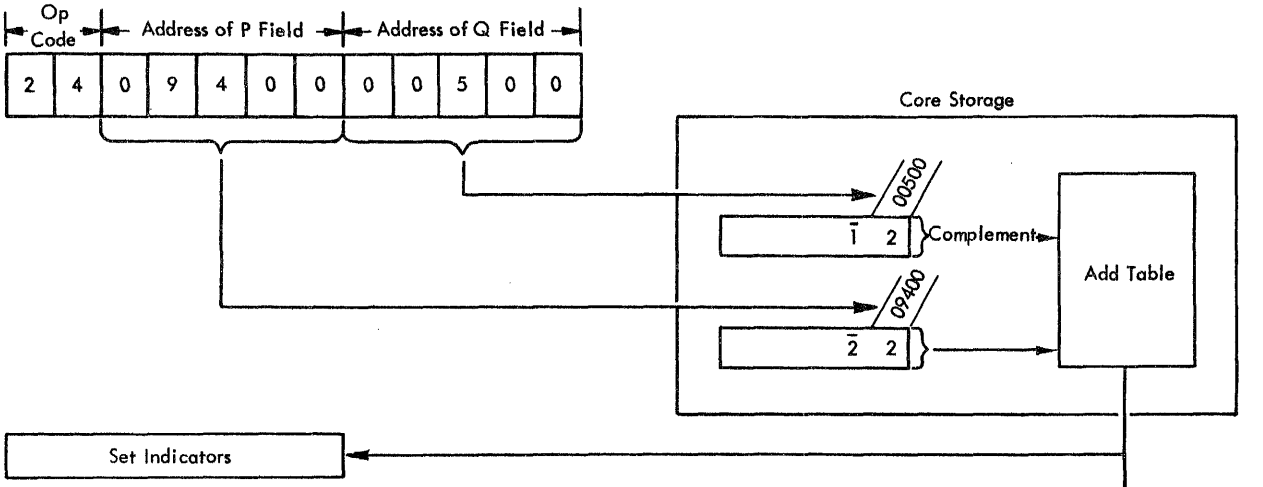
Description. This instruction branches unconditionally to the instruction at the P address, which is the next instruction to be executed. The Q part of the Branch instruction is not used.

Execution Time. T = 200 μsec

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). The data in the field at the Q address is transmitted to the P address minus one and to successively lower-numbered core storage positions. The field at Q remains unchanged. The instruction at the P address is the next one executed.

Figure 36 shows how the instructions Branch and Transmit and Branch Back are utilized to make use of a common subroutine. Note that the data for the subroutine is stored beside the subroutine by the



1. High/Positive
   Set On - P Higher than Q
   Set Off - P Not Higher than Q
2. Equal/Zero
   Set On - P Equal to Q
   Set Off - P Not Equal to Q

The Q data (12) is subtracted from the P data (22). The result (+10) causes the High/Positive indicator to be turned on and the Equal/Zero indicator to be turned off.

Figure 35. Compare Instructions - Data Flow

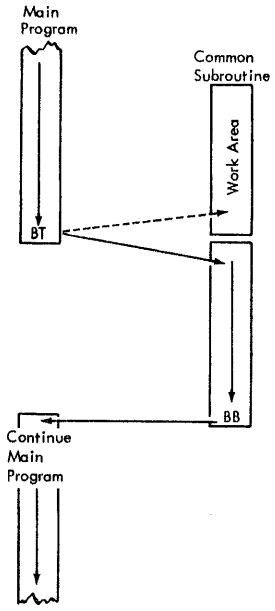


Figure 36. Subroutine Linkage

Branch and Transmit instruction. The last instruction of the subroutine, Branch Back, returns the program to the main routine (the address saved in IR-2 when the Branch and Transmit was executed).

In Figure 37, 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. $T = 200 + 40D_q$ μsec

Branch and Transmit Immediate (BTM-17)

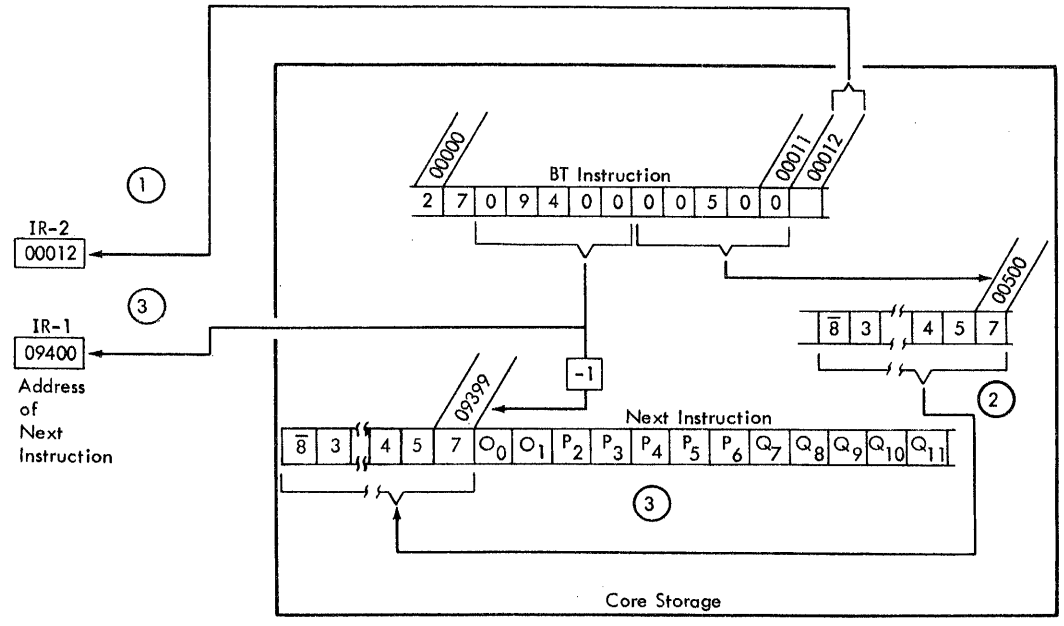
Description. Same as Branch and Transmit (BT-27) except that the digits in the Q part of the instruction are used as Q data.

Execution Time. $T = 200 + 40D_q'$ μsec

Branch Back (BB-42)

Description. This instruction causes the computer to branch unconditionally to either (1) 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 (2) the address saved in PR-1 by previous depression of the Save key on the console when the CPU was in manual mode.

The Save key function is examined first since it has priority over a Branch and Transmit instruction. If



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 37. Branch and Transmit - Data Flow

the Save key function is active, the console 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 transfered to IR-1, and IR-2 is cleared. If a second Branch Back instruction occurs without an intervening Branch and Transmit instruction or a Save key depression, a MARS check results. The P and Q addresses of the Branch Back instruction are not used.

Execution Time. T = 200 μsec

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. If the branch occurs (digit is not zero) T = 240 μsec. If the branch does not occur, T = 200 μsec.

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. If the branch occurs (flag bit not present), T = 240 μsec. If the branch does not occur, T = 200 μsec.

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 Branch No Record Mark 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 (right-most 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 two EL punches at the end. Actually, the second EL punch becomes the first and only character of the last record, i.e., after reading the last *data* record (terminated by the first of the two successive EL punches), the next read instruction causes the second EL punch to read in.

A Branch No Record Mark instruction, which tests the first (leftmost) character of each record, follows each read instruction and normally permits data to be processed. When the first character read in a record is an EL punch, processing of data is stopped by this instruction, which causes the program to branch to the end-of-tape routine.

Execution Time. If the branch occurs (record mark not present), T = 240 μsec. If the branch does not occur, T = 200 μsec.

NOTE: If the 1311 Disk Storage Drive is attached to the system, an additional special character, a group mark, is used in disk storage operations. The Branch No Record Mark instruction treats a group mark in the same manner as a record mark because both contain 8 and 2 bits.

Branch Indicator (BI-46)

Description. The indicator specified by Q_8 and Q_9 of the instruction is interrogated; if the indicator is ON, a branch to the P address occurs. Indicators are always in one of two conditions, ON or OFF. The Q_7, Q_{10}, and Q_{11} positions of the instruction are not used.

Information concerning 1620 indicators is given in Table 2 and further explained, as follows:

Program Switches (01-04). The status of these four indicators is determined by the on/off conditions of their respective Program switches on the 1620 console.

Read and Write Check (06 and 07). The Rd/Wr Check indicators are turned on when erroneous data is transferred to or from an input/output device.

Last Card (09). This indicator is turned on whenever the data from the last card is correctly transferred from 1622 input buffer storage to core storage.

Arithmetic (11, 12, 13, and 14). The arithmetic indicators, 11, 12, and 14, are explained under ARITHMETIC INSTRUCTIONS. The single indicator, High/Positive or Equal/Zero (13), provides the means of interrogating both the H/P (11) and E/Z (12) indicators with one Branch Indicator or Branch No Indicator instruction — no indicators are turned off by this instruction.

Exponent Check (15) Special Feature. The Exponent Check indicator is turned on by an exponent underflow or overflow. This indicator is described in more detail under MANTISSA AND EXPONENT ANALYSIS.

MBR-E and MBR-O (16 and 17). All data entering and leaving core storage does so via the MBR registers.

Table 2. 1620 Indicators

Machine	Code	Name	Light	Turned on by	Turned off by
1620	01-04	1620 Program Switches 1-4	No	Operator (Program switch On)	Operator (Program switch Off)
	06	Read Check	Yes	I/O Input Error	BI, BNI, or Reset key
	07	Write Check	Yes	I/O Output Error	BI, BNI, or Reset key
	09	Last Card (1622 Card Read)	Yes	Last Card Data Transfer to Core Storage	BI, BNI, or Reset key
	11	High-Positive (H/P)	Yes	Arithmetic Result positive and greater than zero	Reset key or next arithmetic instruction
	12	Equal-Zero (E/Z)	Yes	Arithmetic Result of zero	Reset key, or next arithmetic instruction
	13	H/P or E/Z	No	Indicator 11 or 12	Indicators 11 and 12 Off
	14	Arithmetic Check	Yes	Arithmetic Check	BI, BNI, or Reset key
	15	Exponent Check	Yes	Exponent Overflow/Underflow	BI, BNI, or Reset key
	16	MBR-E Check	Yes	Parity Error in MBR-E	BI, BNI, or Reset key
	17	MBR-O Check	Yes	Parity Error in MBR-O	BI, BNI, or Reset key
	19	Any Check	No	Indicator 06, 07, 16, 17, or 39 On	Indicators 06, 07, 16, 17, and 39 Off
1311	36	Address Check	Yes	Unequal address, or no address found in disk storage or multiple heads or multiple drives are selected	BI, BNI, or Reset key, or disk operation
	37	Wrong-Length Record/ Read-Back Check	Yes	Incorrect record length, or corresponding data in disk storage and core storage does not compare	BI, BNI, or Reset key, or disk operation
	38	Cylinder Overflow	Yes	Disk operation completes last sector and sector count is not 000	BI, BNI, or Reset key, or disk operation
	39	Any Disk Error	No	Indicator 36, 37, or 38 On	Indicators 36, 37, and 38 Off
1443	25	Printer Check	Yes	Parity error or sync. check in 1443	If a parity error: BI, BNI, 1620 or 1443 Reset keys. If a sync. check error: 1443 Reset key only.
	33	Channel 9	No	Punched hole in Channel 9 of carriage control tape	BI, BNI, 1620 Reset key, or a punched hole in Channel 1 of carriage control tape.
	34	Channel 12	No	Punched hole in Channel 12 of carriage control tape	BI, BNI, 1620 Reset key, or a punched hole in Channel 1 of carriage control tape.
	35	Printer Busy	No	1443 printing (buffer is unavailable for loading)	1443 Completion of printing (buffer available for loading)
	42	Seek Complete	No	End of Seek Operation	BI, BNI, or Power-on Reset

Their associated indicators (16 and 17) are turned on if a parity error occurs.

Any Check (19). This indicator provides the means of interrogating five error conditions (Indictors 6, 7, 16, 17, and 39) with a single Branch Indicator or Branch No Indicator instruction—no indicators are turned off by the instruction. Indicator 19 is turned off only when all of the five error conditions are off.

Disk Storage Indicators (36-39). These indicators are used in conjunction with the IBM 1311 Disk Storage Drive. They are described briefly under 1620 CONSOLE and described in more detail in the publication *IBM 1311 Disk Storage Drive* (Form A26-5650).

Execution Time. If branch occurs (indicator on), T = 200 μsec. If the branch does not occur, T = 160 μsec.

Branch No Indicator (BNI-47)

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

Execution Time. If the branch occurs (indicator off), T = 200 μsec. If the branch does not occur, T = 160 μsec.

Branch No Group Mark (BNG-55)

Description. This instruction is available when the 1311 Disk Storage Drive is attached to the System. The purpose of this instruction is to enable the program to test for the presence or absence of a group mark in core storage.

If the core storage location specified by the Q address does *not* contain a group mark, the program branches to the instruction at the P address. (The

P address must be an even-numbered address.) If the location tested contains a group mark, the next instruction in sequence is executed.

It should be noted that the Branch No Record Mark (BNR-45) instruction treats a group mark in the same manner as a record mark because both contain 8 and 2 bits.

Execution Time. T = 240 μsec if a branch occurs (no group mark); T = 200 μsec if a branch does not occur.

Internal Data Transmission Instructions

The following instructions 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. A flag bit at the Q address is also transmitted.

Execution Time. T = 200 μsec

Transmit Digit Immediate (TDM-15)

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

In Figure 38, if the Transmit Digit Immediate instruction (15 09400 00500) is at 09200, the digit (0) at 09211 is transmitted to the core storage location specified by the P address (09400). 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. T = 200 μsec

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 39, 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 Q field.

T = 160 + 40D_q μsec

Transmit Field Immediate (TFM-16)

Description. The description is the same as that for Transmit Field 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 + 40D_q' μsec

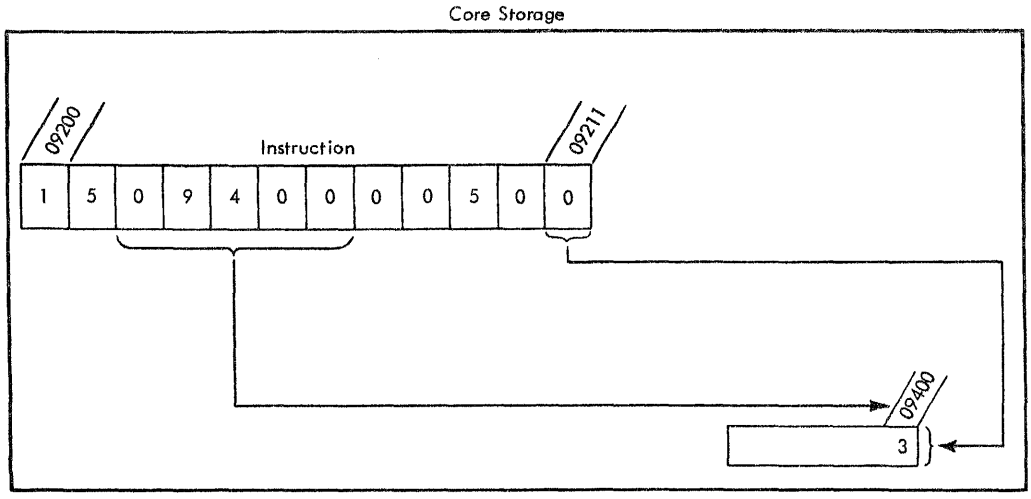


Figure 38. Transmit Digit Immediate - Data Flow

Transmit Record (TR-31)

Description. The record with its high-order (leftmost) position at the Q address is transmitted to the P address and successively higher-numbered core storage locations. The record at the Q address remains unchanged by the transfer.

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

Execution Time. The execution time varies according to the number of characters (high-order zeros included) in the record at the Q address.

$T = 160 + 40D_q$ μsec

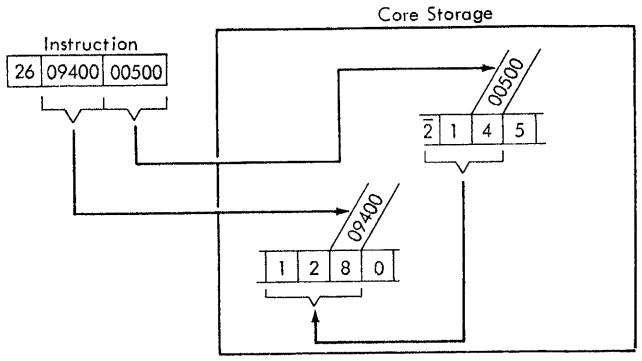


Figure 39. Transmit Field - Data Flow

The instructions — Move Flag, Transfer Numerical Strip, and Transfer Numerical Fill — are 1620 Special Features and must be ordered as such.

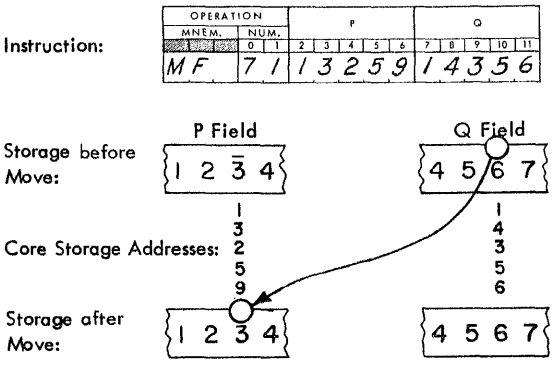
Move Flag (MF-71)

Description. 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 specified by the Q address is without a flag, the flag at the P address is cleared. If it 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.

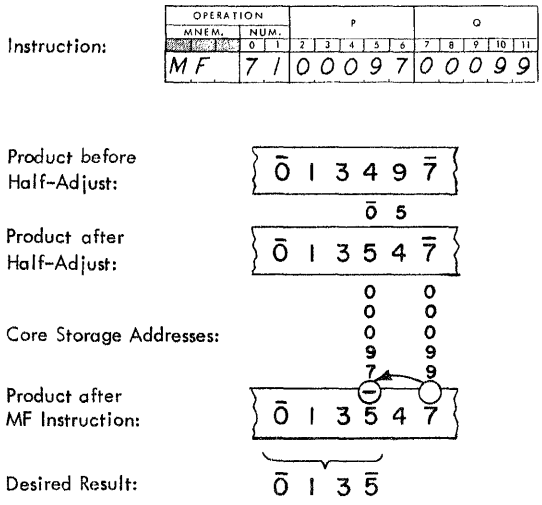
Figure 40 illustrates the movement of a positive sign which, in effect, removes a flag; Figure 41, the movement of a negative sign; Figure 42, the lengthening of a field. All three illustrate the simplified programming required.

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 a 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 43. 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



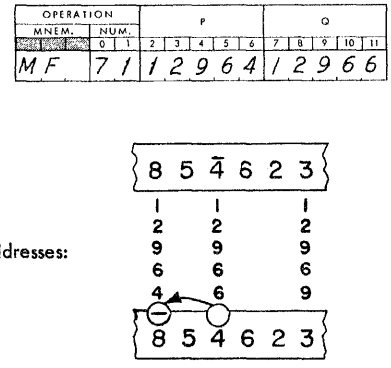
Flag over the "3" is cleared because there is no flag over the "6".

Figure 40. Move Flag, Positive Sign



Flag is moved within same field. Note that no flag remains over the "7" after execution of the MF instruction.

Figure 41. Move Flag, Negative Sign



Flag is moved within same field. Note that no flag remains over the "4" after execution of MF instruction.

Figure 42. Move Flag, Lengthen Field

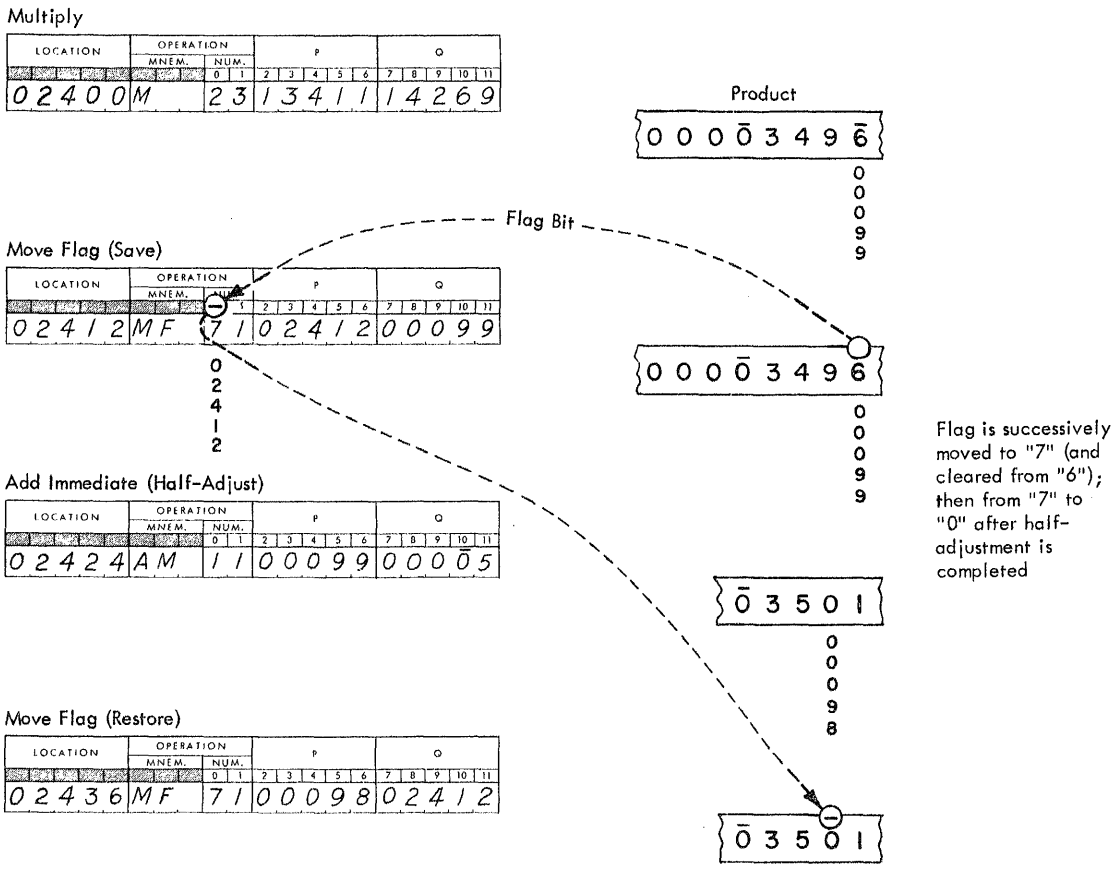


Figure 43. Move Flag to Half-Adjust One Position

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. T = 240 μsec

Transfer Numerical Strip (TNS-72)

Description. This instruction converts numeric data in the two-digit alphameric mode into single-digit numeric data, with sign. The units numeric position of the alphameric field is specified by the P address of the instruction which must always be an odd-numbered core storage location. The units position of the numeric field is specified by the Q address. Transmission of the numeric field is specified by the Q address. Transmission of the numeric digits from the odd-numbered positions 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 numeric field. The flag bit must be placed in the numeric field prior to the TNS instruction to define the high-order position. It remains unchanged by the instruction. For example, the numeric digits $\bar{4}$, 3, 2, and 1 in Figure 44 are stripped from their alphameric codes 54, 73, 72 and 71, respectively. The flag bit previously stored at 17461 terminates the operation.

The zone digits in the even-numbered core storage locations of the alphameric (P) field are ignored except for a 5, 2, or 1 in the units zone position. A

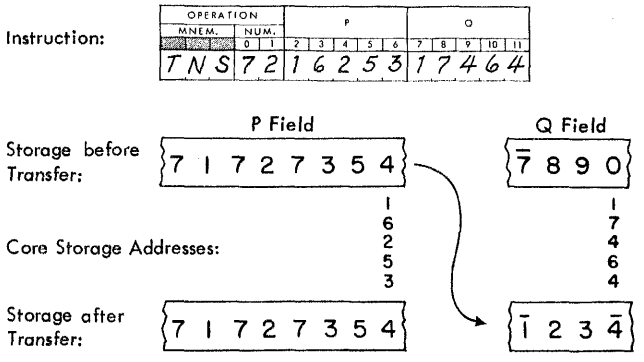


Figure 44. Transfer Numeric Strip

5, 2, or 1 in the units zone position is converted by TNS to a flag bit over the units digit of the numeric (Q) field except when the zone 2 is followed by a nonzero digit in the corresponding numerical position. Any number other than a 5, 2, or 1 results in no flag over the units digit. A 5 or 1 in a units zone position of an alphamerically coded number field indicates a negative number read from an input card or paper tape. A 2 in a units zone position occurs when an X alone, representing a negative zero, is read from an input card. This 2, 0 combination is converted to a flagged 0 by the instruction. For example, *71727320 becomes* $\bar{1}23\bar{0}$. A 2 in the units zone position with a nonzero digit in the corresponding numerical position will be ignored *(*$\bar{7}$*1727325 becomes* $\bar{1}$*235).*

The digit in each odd-numbered core storage position of the alphameric field is transmitted without change to the corresponding position of the numeric field, concluding with the digit transmitted to the high-order position of the numeric field containing the flag that defines the field. Except for the field flag, all previous contents of the numeric 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 numeric field.

Because such flag bits, when transmitted, may affect the length or sign of the numeric 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 a 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 numeric field.

Execution Time. $T = 160 + 40D_p$ μsec

Transfer Numerical Fill (TNF-73)

Description. The TNF instruction moves and expands single-digit numeric data with sign, into two-digit alphameric data. The units numeric position of the alphameric field 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 numeric (Q) field. The digits in the numeric 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 numeric field remains unchanged.

In Figure 45, the numeric digits $\bar{7}$, 8, 9 and $\bar{1}$ fill in the alphameric field locations 16251, 16253, 16255, and 16257, respectively. 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 numeric field is converted to a 5 in the even-numbered units zone position of the alphameric (P) field. Absence of a flag in the units position of the numeric field results in a 7 being placed in the even-numbered units zone position. All other even-numbered zone positions of the alphameric field are automatically filled with 7s.

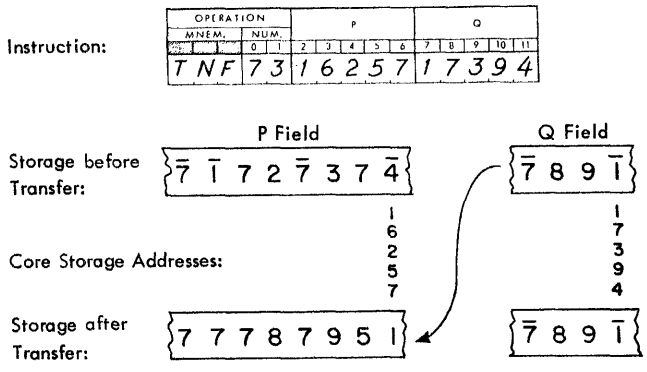
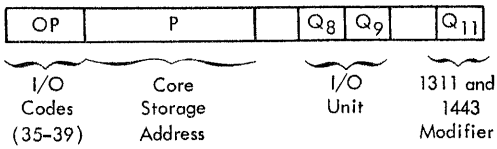


Figure 45. Transfer Numeric Fill

This section is concerned with instructions for 1620 input/output units. Complete operating details for these units are described in the IBM publications on the cover page of this manual.

1620 Input/Output instructions enable the transfer of data between core storage and disk storage, and core storage and input/output units. The Read instructions transfer data from the input unit *to* core storage, and the Write and Dump instructions transfer data *from* core storage to the output unit. The Q_8 and Q_9 digits specify the I/O unit, as follows:

01 – Typewriter
02 – Tape Punch*
02 – Plotter*
03 – Paper Tape Reader*
04 – Card Punch
05 – Card Reader
07 – 1311 Disk Storage Drive
09 – 1443 Printer



The Q_{11} position of the instruction is used to define 1311 operations and 1443 carriage spacing. Although the 1311 is controlled with standard input/output codes (36 and 38) and Q_{11} modifiers, the number and variety of instructions make their placement at the end of this section desirable.

Read Check and Write Check Indicators

The Read Check and Write Check indicators (codes 06 and 07) are turned on if a parity error occurs in the 1620 during input and output operations. Once the Read or Write Check indicator is turned on, it is not turned off by the reading or writing of subsequent correct characters; program interrogation or manual reset is required to turn it off.

System Restart

The CPU will hang-up if the program addresses an I/O unit that is in a "not ready" condition. In this event, the CPU should be stopped, the not ready condition of the I/O unit corrected – Meter switch on, Power-on switch on, correct placement of disk pack, etc. – and the CPU restarted. (The master 1311's Meter switch must be on to operate any slave 1311.)

Read Numerically (RN-36)

Description. Numeric information from an input unit is transmitted serially to the P address and to successively higher-numbered 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 the 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. 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 pressed before pressing the Release key on the console.
3. Reading the 80th character from the *Card* input buffer storage into core storage.

Each numeric 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. All data that is replaced, including flag bits, is destroyed.

The – (dash) and J through R characters from the paper tape reader are entered into core storage as numeric digits with flag bits. Numeric blanks from the card reader are entered into core storage as C, 8, and 4 bits and are punched out as blanks on a Write Numerically operation. Actual blanks (from unpunched card columns) are entered into core storage as plus zeros (C bits) and are punched out as zeros on a Write Numerically operation. No other alphabetic or special characters (except the record mark) are transmitted correctly to core storage when this instruction is used.

* The Plotter (1627) and Paper Tape Reader and Punch (1621) cannot be installed on the same system except by special order.

When the typewriter is selected by a Read Numerically instruction, the 1620 stops in automatic mode to await the manual entry of information from the typewriter keyboard.

Although a Read Numerically instruction may be used to transfer alphameric data from the 1622 without a parity error occurring, several characters (equal sign, period, dollar sign, and comma) besides the record mark will have an 8, 2 representation in core storage. These characters will be treated as record marks during execution of Write Numerically and Transmit Record instructions. Thus, it behooves the programmer to be aware of card data and format before using the Read Numerically instruction to read alphameric data. Use of the Read Alphamerically instruction eliminates this problem.

The core storage characters that result from the transfer of alphameric card data with a Read Numerically instruction are shown in Figure 46.

Execution Time. Execution times for the paper tape reader and typewriter depend upon the speed of the input unit selected and the number of characters read. Execution time is 3.4 ms for transferring 80 columns of data from the 1622 input buffer storage to core storage.

Read Alphamerically (RA-37)

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

The units digit (P_6) of the P part of the instruction must be an odd number; otherwise, the input informa-

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

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

NOTE: Dollar sign, equal sign, period, and comma are interpreted as record marks on Write Numerically and Transmit Record Instructions.

Figure 46. Core Storage Data Resulting from Reading Alphameric Card Data with RN Instruction

tion is not placed in core storage correctly and parity errors may occur when the input information is read in. This is because of the 2-character transfer operation of core storage. The odd-numbered location must contain the right-hand (numeric) 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 numeric 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. Pressing 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 pressed before the Release key on the console is pressed.
3. Reading the 80th character from card *input* buffer storage into the 159th and 160th positions of core storage. A record mark is not generated in storage.

Information from an input unit may be a random mixture of numeric,alphabetic, and special characters. Each character from an input unit is stored in core storage as two digits. Flag bits are not transmitted on characters read by an input unit; 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 unit is stored in core storage as a numeric zero digit (C bit) followed by a single record mark character (coded C-8-2).

Numeric data stored in the two-digit alphameric mode must be converted by programming to single-digit numeric data before being used in arithmetic commands. The Transfer Numerical Strip instruction (special feature) may be used for this conversion.

When the typewriter is selected, the 1620 stops in automatic mode to await the manual entry of information from the typewriter keyboard.

Execution Time. Same as Read Numerically.

Write Numerically (WN-38)

Description. Numeric information in the P address and in successively higher-numbered core storage locations is transmitted serially to an output unit. 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. Pressing the Release key on the *console.* If the Release key is not pressed, and a record mark is not encountered before the data at the highest-numbered core storage address is written, the machine "loops back" to 00000 and transmission continues.
3. Writing the 80th position in *card output* buffer storage.
4. Loading the 197th character into the *Printer* buffer, or sensing of a record mark or group mark in core storage. The print buffer contains 197 positions even though the 1443 prints only 120 or 144 positions. A record mark or group mark should be placed in positions 121 or 145 to prevent possible Printer Checks caused by the transmission of unedited positions. The terminating record mark or group mark is not transferred to the print buffer and all remaining buffer positions are reduced to blanks, including the record mark or group mark position. A record mark or group mark at the P address causes the buffer to be loaded with blanks and a "blank line" to be printed.
5. A record mark or group mark terminates operation of the *Plotter* also.

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

Execution Time. Execution times for the paper tape reader and typewriter depend on the speed of the output unit selected and the number of characters written.

Execution time equals 3.4 ms for transferring 80 columns of data from Core storage to 1622 output buffer storage; the print buffer is loaded in 8.06 ms (120 or 144 characters).

Although the Models 1 and 2 Plotters require 3.3 ms and 5 ms for each character, respectively; the CPU is released after 300 μsec if one-character records are used.

Write Alphamerically (WA-39)

Description. Alphameric information from the P address and from successively higher-numbered core storage locations is transmitted serially to an output unit. 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.* If the character at the P address is a record mark, the CPU-typewriter operation will be suspended.
2. Pressing the Release key on the console. If this is done before an alphameric record mark has been encountered in core storage, a record mark character is not written, and if the unit is the paper tape punch, an end-of-line character is not punched. If the Release key is not pressed 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 159th and 160th core storage positions into the 80th position of *card output* buffer storage.
4. Writing the 197th position in the *Printer* buffer, or sensing an alphameric record mark or group mark in core storage. The print buffer contains 197 positions even though the 1443 prints only 120 or 144 positions. A record mark or group mark should be placed in position 121 or 145 to prevent possible Printer Checks caused by the transmission of unedited positions. Either of these characters at the P minus 1 and P addresses causes the buffer to be loaded with blanks and a "blank line" to be printed.

   Flag bits in the data area do not affect BCD buffer translation.
5. An alphameric record mark or group mark also terminates a *Plotter* operation. Positive and negative numeric characters, *except minus zero,* in the two-digit or alphameric format may be used in the Plotter record. Other characters can cause either conflicting commands or no command to be sent to the Plotter. Only the absolute value of the digit is translated for a plotter signal; that is, the sign does not affect the direction of Plotter action.

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

The P address must designate an odd-numbered core storage position.

Execution Time. Same as Write Numerically.

Dump Numerically (DN-35)

Description. Numeric information is transmitted serially to an output unit, beginning with the P address and continuing through successively higher-numbered addresses. Transmission terminates after the character has been written from the highest-numbered position of the addressed core storage module. This address is 19999, 39999, or 59999, depending on the 20,000-position module specified by the P address.

If the output unit selected is the *tape punch,* an end-of-line character is punched in the tape immediately following the last character.

If the output unit selected is the *card punch,* punching continues until all 80 columns of the last card have been punched out. 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. Transmission also may be terminated at any time by pressing the Release key on the 1620 console.

The printer buffer is loaded with 197 characters, including record marks and group marks – these characters do not terminate the operation as they do for the Write Numerically and Write Alphamerically instructions. The output or result of each character is shown in Table 3 for each output instruction. Only one print line (first 120 or 144 characters of the buffer load) occurs per Dump instruction. The "Wrap around" principle applies.

A "dump memory" can be effected by a loop of three instructions: Dump, followed by Add Immediate (120 or 144) to the Dump P address, followed by Branch Back to the Dump. The Stop key can be used to halt the routine.

Except for the $\bar{0}$ punchout on cards, each numeric character, as well as any single record mark character, is written on the output unit along with its flag bit (if any); the Plotter, of course, accepts each character as a plotting command. The character in core storage remains unchanged. This causes only the flag (X punch) of a $\bar{0}$ to be punched in an output card, so that any subsequent printout of that character from the card will be as a hyphen. (See Appendix C.)

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

Execution Time. Same as Write Numerically.

Table 3. Character Coding

ALPHAMERIC DATA

Character	Core Storage		Printer Output (Print Alphamerically)
	Alpha.	Num.	
(Blank)	C	C	blank
. (period)	C	C 21	.
)	C	4	)
+	1	C	+
$	1	C 21	$
*	1	4	*
- (hyphen or minus)	2	C	-
/	2	1	/
, (comma)	2	C 21	,
(	2	4	(
=	C 21	C 21	=
@	C 21	4	@
A	4	1	A
⋮	⋮	⋮	⋮
I	4	C8 1	I
0 (-)	C 4 1	C	-
J	C 4 1	1	J
⋮	⋮	⋮	⋮
R	C 4 1	C8 1	R
S	C 42	2	S
⋮	⋮	⋮	⋮
Z	C 42	C8 1	Z
0	421	C	0
⋮	⋮	⋮	⋮
9	421	C8 1	9
‡	C	C8 2	terminates
≢	C	C8421	terminates

NUMERICAL DATA

Character	Core Storage	Printer Output	
		Print Numerical	Printer Dump
0, 1, 2, ..., 9	C, ..., C18	0, ..., 9	0, ..., 9
numerical blank	C84	blank	@
record mark (≠)	C82	terminates	≠
group mark (≢)	C8421	terminates	G
$\bar{0}, \bar{1}, ..., \bar{9}$ (flagged)	F, CF1, ..., F18	-, J, ..., R	-, J, ..., R
numerical blank (flagged)	F84	blank	*
record mark (flagged)	F82	terminates	W
group mark (flagged)	F8421	terminates	X

1311 Disk Storage Drive Instructions

The 1311 publication referred to on the cover page of this manual contains complete 1311 details. The format for disk storage instructions (Figure 47) is the same as for other 1620 operations and consists of a 2-digit Op code, a 5-digit P address, and a 5-digit Q address. (The P-address must be even.)

The Q_8 Q_9 digits of the instruction must contain 07 and the Q_{11} digit must contain a modifier digit to define a particular function within the regular Read Numerically or Write Numerically operation. The general areas of function definition are as follows:

1. Read or write a specified number of data sectors.
2. Read or write one full disk track, both data and sector addresses.
3. Read data from disk storage and compare it with the data in core storage from which it was written.
4. In any of the above functions, the Q_{11} code also specifies whether or not the correct length of the record read or written is checked.

Disk Control Field

The disk control field which must be located at an even address is a 14-digit field that specifies the disk address where the data is to be read from or written to, the disk storage drive to be used, and the number of sectors to be read from or written to (Figure 48). The number of positions in core storage reserved for the data field must be large enough to contain all the data read from the disk. A group mark at the end of this field defines the total length of the area reserved for data.

The *sector address* contains the disk address of the data to be read or written. If more than one sector is read or written, the address of the first sector is specified.

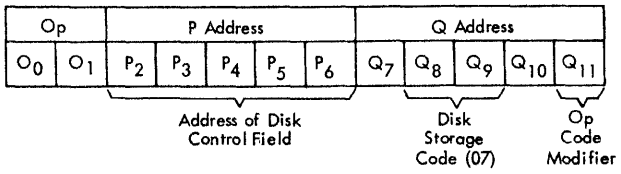


Figure 47. 1620 Instruction Format

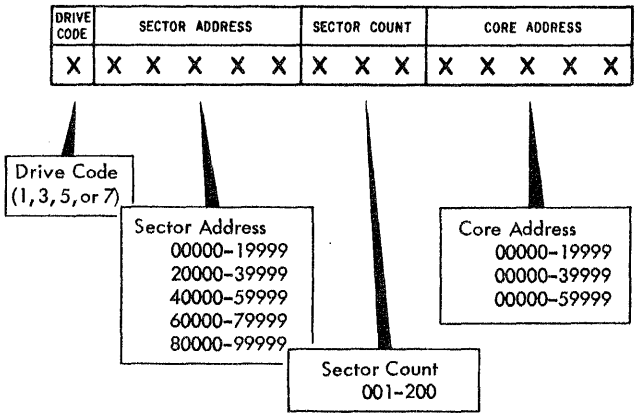


Figure 48. Disk Control Field

The disk address of the data indicates which storage drive is to be used by the system for an instruction (if more than one is installed on the system).

The *drive code* enables reading and writing on any block of disk addresses regardless of the disk drive on which the pack is located, and when the occasion requires, to have more than one disk pack containing the same disk addresses on-line at one time.

The selection of a disk storage drive by either a drive code or a sector address is summarized in the chart below.

Drive Code	or	Address Range	Selects Drive
1		00000-19999	0
3		20000-39999	2
5		40000-59999	4
7		60000-79999	6

Single Disk Drive System. If only one disk storage drive is attached to the system, the drive code has no significance; the drive is addressed regardless of the digit in the drive code or the range into which the sector address falls. Also, in a single drive system, the range of addresses is not limited to 00000-19999; any range of disk storage addresses between 00000-99999 can be used.

Multiple Disk Drive Systems. If an address and/or drive code used in a read or write operation addresses a nonexistent disk drive, the computer is interlocked The Disk light (under the Instruction and Execute lights) and the Punch/Disk Interlock light turn on. A seek operation to an invalid address or to a nonexistent drive does *not* result in an interlocked condition; in this case, an error is detected during the following read operation by the address compare feature.

The *sector count* specifies the number of sectors read or written. At the beginning of the operation, the sector count is transferred to a 1620 register (PR-3), which is decremented by one *before* each sector is processed. The operation is terminated when the sector count reaches 000. At any time during a disk operation, the number in PR-3 is one less than the number of sectors yet to be processed, and the sector count in the disk control field has the total number of sectors in the operation.

The *core address* specifies the location of the leftmost position of the data transferred to or from disk storage. The core storage address of a disk storage record must be an even number.

Disk Storage Data Protection

Two safeguards, controlled by programming, are incorporated into the 1311 Disk Storage Drive to prevent incorrect reading and writing of disk storage data.

Read-Only Flag

A flag in the high-order (leftmost) position of the sector address, recorded on the disk, permits the sector to be read only. An attempt to write in the sector (with the exception of a write disk track operation) results in an address check and termination of the operation. The Address Check indicator (36) is turned on.

Wrong-Length Record Check

This function provides a check, in addition to the sector count, that ensures the transfer of the correct length record and prevents the loss of data through reading or writing beyond the intended point. In this respect, the wrong-length record check also verifies that the correct sector count was entered in the disk control field.

To enable a record to be checked for the correct length, a group mark must be placed in core storage in the position following the end of the record. The data transfer is made with a Read, Write, or Check Disk instruction, and is modified to verify the length of the record by checking the position of the group mark and the sector count in PR-3. If the group mark and a sector count of 000 do not coincide, the operation terminates and the Wrong-Length Record/Read-Back Check indicator (37) is turned on.

Program Instructions

The description of program instructions has been arranged in the following manner:

Seek instruction

Instructions using groups marks to establish correct record length:

Sector Mode
Read Disk/WLRC
Write Disk/WLRC
Check Disk/WLRC

Disk operations with records of less than or in multiples of 100 characters:

Track Mode
Read Disk Track/WLRC
Write Disk Track/WLRC
Check Disk Track/WLRC

Instructions not using group marks:

Sector Mode
Read Disk
Write Disk
Check Disk

Track Mode
Read Disk Track
Write Disk Track
Check Disk Track

In the formulas for execution time of disk storage instructions, the following symbols are used:

S = Number of sectors read or written
T = Time in milliseconds

A 2 ms read/write head delay occurs from the time an I/O instruction is executed until data can be transferred. In addition, it is assumed that one-half of a rotation (20 ms) occurs before reading or writing begins because this is the average distance the read/write head is from the sector addressed in any given number of operations.

A summary of disk storage program instructions is contained in Table 4.

Seek

SK or 34 — Q_{11} of 1

The execution of the Seek instruction causes the access mechanism, in the drive unit addressed, to be positioned at the cylinder designated by the sector address in the disk control field. When the operation is initiated, the access mechanism retracts to the "home" position, which places the read/write heads near the periphery of the disks. It then moves to the cylinder specified and remains in that location until another Seek instruction is given. The access mechanism need not be repositioned for subsequent read/write instructions with the *same cylinder.*

Execution Time. Average seek time is 250 ms; maximum time is 392 ms. *The 1620 CPU is interlocked only until the sector address is transferred to the disk control unit* (360 μsec). Internal 1620 processing that does not involve disk storage can then continue and overlap the seek operation.

When the 1311 is attached to the 1710 Control System, a Seek Complete interrupt is available as part of the Input/Output Interrupts special feature. The Seek Complete interrupt signals the CPU that the Seek instruction has been completed and that data can be transferred to or from disk storage. This allows the CPU to optimize available processing time during the seek operation. The Seek Complete indicator (42) enables a program test of this interrupt.

Read Disk/WLRC

RDGN or 36 — Q_{11} of 0

Execution of this instruction causes a read/write head to be selected, as specified by the sector address in the disk control field. The selected head then scans the sector for addresses recorded on the disk track until a matching address is found. When a sector address is found in disk storage that matches the sector address in the disk control field, reading begins and continues for the number of sectors specified. As each character is read, it is stored in the core storage location specified and in sequentially higher-numbered positions. It should be noted that if the data transferred exceeds storage capacity, reading will continue in location 00000 and in sequentially higher positions.

If a matching address is not located within one complete revolution of the disks (index point sensed twice), the operation is terminated and the Address Check indicator (36) is turned on.

The sector address in the 1620 register OR-1 is incremented by one each time a sector is read, and the sector address for each succeeding sector that is read is compared to ensure correct sequential progression. If an address fails to compare, the operation is terminated and the Address Check indicator (36) is turned on. The address (plus one) that failed to match is in OR-1 and the number of sectors yet to be read, excluding the one whose address failed to match, is in PR-3.

If the number of sectors read goes beyond one disk surface, a shift to the read/write head for the next disk surface is made automatically and reading continues without loss of time. If the end of the cylinder is reached however, and the sector count

Table 4. Summary of Disk Storage Program Instructions

Mode	Op Code	Q_{11}	Instruction	Mnemonic	Operation
	34	1	Seek	SK	Return access mechanism to "home" position and then move in to cylinder specified.
Sector Mode	36	0	Read Disk/WLRC	RDGN	Transfer data from specified number of disk sectors to core storage. Check length of record.
	38	0	Write Disk/WLRC	WDGN	Transfer data from core storage to specified number of disk sectors. Check length of record. Note: The Write Address key must be off.
	36	1	Check Disk/WLRC	CDGN	Compare data in specified number of disk sectors with data in core storage. Check length of record.
Track Mode	36	4	Read Disk Track/WLRC	RTGN	Transfer addresses and data from the 20 sectors of one track to core storage. Check length of record.
	38	4	Write Disk Track/WLRC	WTGN	Transfer address and data from core storage to the 20 sectors of one disk track. Check length of record. Note: The Write Address key must be on.
	36	5	Check Disk Track/WLRC	CTGN	Compare addresses and data from the 20 sectors of a disk track with addresses and data in core storage. Check length of record.
Sector Mode	36	2	Read Disk	RDN	Transfer data from specified number of disk sectors to core storage.
	38	2	Write Disk	WDN	Transfer data from core storage to specified number of disk sectors. Note: The Write Address key must be off.
	36	3	Check Disk	CDN	Compare data from specified number of disk sectors with data in core storage.
Track Mode	36	6	Read Disk Track	RTN	Transfer addresses and data from the 20 sectors of one track to core storage.
	38	6	Write Disk Track	WTN	Transfer addresses and data from core storage to the 20 sectors of one disk track. Note: The Write Address key must be on.
	36	7	Check Disk Track	CTN	Compare addresses and data from the 20 sectors of a disk track with addresses and data in core storage.

has not been decremented to 000, the operation is terminated and the Cylinder Overflow indicator (38) is turned on. Therefore, the greatest number of sectors that can be read with one instruction is 200, one full cylinder.

As each character is read from disk storage, it is checked for parity. Failure to meet the parity check causes the Read-Check indicator (06) to be turned on. The MBR-E indicator (16) or the MBR-O indicator (17 is turned on to indicate a parity error in a character going into core storage. Any parity check will terminate the operation.

This instruction checks, in addition to sector count, that the correct number of characters is transferred from disk storage to core storage. A group mark stored in core storage in the location following the read-in area provides the correct termination of the operation.

A group mark can cause an incorrect termination, which is indicated by the Wrong Length Record Check indicator (37) being turned on. This occurs when there is a group mark in core storage immediately following the last character of any sector except the last sector of the record.

Group marks in core storage positions other than those immediately following a sector are simply replaced by data from disk storage; they do not affect the operation.

Execution Time. Average T = 22 + 2S ms

Write Disk/WLRC
WDGN or 38 — Q_{11} of 1

This instruction causes a read/write head to be selected as specified by the sector address in the disk control field. The selected head then scans the sector addresses recorded on the disk track until a matching address is found. If a matching address is not found within one complete revolution of the disks, or if a read-only flag is sensed in the matching address, the operation is terminated and the Address Check indicator (36) is turned on. When the sector address in the disk control field matches a sector address in disk storage, writing begins from the core storage location specified and continues sequentially through higher-numbered positions for the indicated number of sectors.

The sector address in the 1620 register OR-1 is incremented by one each time a sector is written, and the address preceding each sector in disk storage is compared against it to ensure the correct sequential succession. If an address fails to compare or if a read-only flag is sensed in a matching address, the operation terminates and the Address Check indicator (36) is turned on.

If the number of sectors written goes beyond one disk surface, a shift to the read/write head for the next disk surface is made automatically and writing continues, without loss of time. However, if the end of the cylinder is reached and the sector count has not been decremented to 000, the operation terminates and the Cylinder Overflow indicator (38) is turned on. Therefore, the greatest number of sectors that can be written with one instruction is 200, one full cylinder.

MBR-E (16) and MBR-O (17) are turned on to indicate a parity error in data from core storage. In addition, each character transferred to disk storage is checked for parity. Failure to meet the parity check causes the Write Check indicator (07) to turn on. Any parity check will terminate the operation.

During Write Disk instructions the Write Address switch must be off.

This instruction checks, in addition to the sector count, that the correct number of characters is transferred from core storage to disk storage. A group mark stored in core storage in the location following the last position of the record provides the correct termination of the operation. It is important to use the wrong-length record check whenever practical to avoid the loss of disk storage data by writing beyond the intended number of sectors.

Group marks in *disk* storage do not affect the operation, and are replaced by data from core storage.

Execution Time. Average T = 22 + 2S ms

Check Disk/WLRC
CDGN or 36 — Q_{11} of 1

This instruction provides the means for checking data written in disk storage against the same data in core storage. This verification is in addition to the record length check and parity check. Data written in disk storage should be verified by a Check Disk instruction after every Write Disk instruction, while the original data is still in core storage.

A Check Disk instruction can also be used to ascertain if core storage data, such as tables, constants, etc., has been changed.

Execution of this instruction causes the head to be selected as specified by the sector address in the disk control field. The selected head then scans the sector addresses recorded on the disk track until a matching address is found. If a matching address is not found within one complete revolution of the disks, the operation terminates and the Address Check indicator (36) is turned on.

When the sector address in the disk control field matches a disk sector address, reading begins at the sector specified and continues for the indicated number of sectors. As each character is read from disk storage it is compared, bit-by-bit and character-by-character, with the data in core storage beginning with the address specified. Failure to compare terminates the operation *at the end of the sector being read* and turns on the Wrong-Length Record Check indicator (37).

During Check Disk instructions, the sector address in the 1620 register OR-1 is incremented by one each time a sector is checked, and the disk address preceding each sector is compared to ensure the correct sequential succession. If an address fails to compare, the operation is terminated and the Address Check indicator (36) is turned on.

If the number of sectors read goes beyond one disk surface, a shift to the read/write head for the next disk surface is made automatically and reading continues without loss of time. If the end of the cylinder is reached however, and the sector count has not been decremented to 000, the operation is terminated and the Cylinder Overflow indicator (38) is turned on. Therefore, the greatest number of sectors that can be checked with one instruction is 200, one full cylinder.

Each character read out of disk storage is checked for parity. Failure to meet the parity check causes the operation to terminate at the end of the sector being read and the Read Check indicator (06) to turn on. In addition, the MBR-E indicator (16) or the MBR-O indicator (17) turns on to indicate a parity error in a character from 1620 core storage. Any parity check terminates the operation.

This instruction checks, in addition to sector count, that the correct number of characters is transferred from disk storage for comparison with core storage. A group mark stored in core storage in the location following the last position of the record provides the *correct* termination of the record. The operation will be terminated however, at the end of the sector being read, by the *first group mark encountered* in either core storage or disk storage, and the Wrong-Length Record Check indicator will be turned on.

Execution Time. Average T = 22 + 2S ms

Disk Operations with Records of Less than or in Multiples of 100 Characters

Read, Write, and Check Disk instructions can be performed with records of less than 100 characters or with records not in even increments of 100 characters, that is, 85, 145, etc. For example, an 85-character record could be used in a program in the following manner:

1. The first time the record is written in disk storage, the core storage location it is written *from* must have a group mark following the last data character. For this example, this would mean reserving 86 positions in core storage and placing a group mark in the 86th position.
2. The 85 positions of data *and the group mark* could then be written in disk storage and the 14 core storage positions following the group mark would also be written into disk storage to fill out the remaining positions in the 100-position disk storage record. The Wrong-Length Record Check indicator would be turned on.
3. Subsequently, whenever this record is read into core storage from disk storage, only the 85 data characters and the group mark are placed in core storage; the remaining 14 positions of the 100-position disk storage record *are not read into core storage.* The Wrong-Length Record Check indicator would be turned on.

During a Check Disk instruction, a group mark in either core storage or disk storage stops the operation; if the record does not consist of an even increment of 100 characters, the Wrong-Length Record Check indicator turns on.

Read Disk Track/WLRC
RTGN or 36 — Q_{11} of 4

This instruction causes a full track of addresses and data (20 sectors) to be read into core storage and to be checked for correct record length. The group mark in core storage must be placed in the location following the 2100th-character position to allow sufficient core storage positions for the five address positions followed by the 100 data positions for each of the 20 sectors. While it is not a requirement, the sector count in the disk control field should be set at 20 for consistent programming.

Reading from the disk track always begins at the index point and continues for 20 sectors. This instruction can be used to read a track on which an address check has occurred during a read operation in the sector mode by merely changing the sector count to 20 and the Q_{11} digit to 4 in the disk control field. *No comparison is made between the disk sector address and the sector address in the disk control field during this operation*

A group mark, in the data being transmitted from disk storage, halts the operation at the end of the sector being read and turns on the Wrong-Length Record Check indicator (37). Group marks in core storage, in other than the position following the 2100 positions of track data and addresses, do not affect the operation and are replaced by data from disk storage.

Execution Time. Average T = 62 ms

Write Disk Track/WLRC
WTGN or 38 — Q_{11} of 4

The Write Address switch on the 1311 must be on in order to perform a Write Disk Track operation. This instruction causes a full track of addresses and data (20 sectors) to be written into disk storage and to be checked for correct record length. The group mark in core storage must be placed in the location following the 2100th-character position to allow for the five address positions followed by the 100 data positions for each of the 20 sectors. While it is not a requirement, the sector count in the disk control field should be set at 20 for consistent programming.

This instruction can be used for the initial recording of sector addresses, for changing read-only status, or for correcting an address. *A read-only flag does not prevent writing when the Write Disk Track instruction is used.*

Writing with this instruction always begins at the index point located on the disk surface between the end of the last sector and the beginning of the first one. Any sector address that is recorded on the track selected may be specified in the disk control field. When the address is matched, writing begins *the next time the index point is sensed.* When a match cannot be obtained with any of the track addresses, an IBM Customer Engineer should be consulted.

A group mark anywhere in the record being transmitted from core storage turns on the Wrong-Length Record Check indicator (37) and terminates writing at the end of the sector containing the group mark. Group marks in the record in disk storage do not affect the operation and are replaced by data from core storage.

Execution Time. Average T = 62 ms

Check Disk Track/WLRC
CTGN or 36 — Q_{11} of 5

This instruction causes a full track of addresses and data (20 sectors) to be read from disk storage, compared with addresses and data in core storage, and checked for correct record length. The group mark in core storage must be in the location following the 2100th-character position to allow enough core storage positions for the five address positions followed by the 100 data positions for each of the 20 sectors. While it is not a requirement, the sector count in the disk control field should be set at 20 for consistent programming.

Reading with the Disk Track instruction always begins at the index point located on the disk surface between the end of the last sector and the beginning of the first one. Any sector address recorded on the track addressed may be specified in the disk control field. When the address is matched, reading begins the next time the index point is sensed.

A group mark stored in core storage in the location following the last position of the record provides the *correct* termination of the record. However, the operation will be terminated (at the end of the sector being read) by the *first group mark encountered* in either core storage or disk storage, and the Wrong-Length Record Check indicator will be turned on.

Execution Time. Average T = 62 ms

Read Disk
RDN or 36 — Q_{11} of 2

Data is read from disk storage and stored in 1620 core storage in 100-character multiples for the number of sectors specified. The operation is the same as for Read Disk/WLRC, except that group marks do not affect transmission or checking. Group marks in the record in disk storage are transmitted as data; group marks in the record in core storage are replaced by data.

Execution Time. Average T = 22 + 2S ms

Write Disk
WDN or 38 — Q_{11} of 2

Data is transferred from core storage and written in disk storage in 100-character multiples for the number of sectors specified. The operation is the same as for Write Disk/WLRC, except that group marks do not affect transmission or checking. Group marks in the record in core storage are transmitted as data; group marks in the record in disk storage are replaced by data.

Execution Time. Average T = 22 + 2S ms

Check Disk
CDN or 36 — Q_{11} of 3

Data is read from disk storage and compared with data in core storage in 100-character multiples for the number of sectors specified. The operation is the same as for Check Disk/WLRC, except that group marks do not affect checking. Group marks in disk storage and core storage are compared as data.

This instruction can be used to check records of less than 100-character multiples and all records read or written without wrong-length record check. With this instruction, the group marks in the succeeding data, written to fill out the 100-sector positions, are compared bit-by-bit and character-by-character.

Execution Time. Average T = 22 + 2S ms

Read Disk Track
RTN or 36 — Q_{11} of 6

This instruction causes a full track of addresses and data (20 sectors) to be read from disk storage and stored in 1620 core storage. The operation is the same as for Read Disk Track/WLRC, except that group marks do not affect transmission or checking. Group marks in the record in disk storage are transmitted as data; group marks in the record in core storage are replaced by data.

Execution Time. Average T = 62 ms

Write Disk Track
WTN or 38 — Q_{11} of 6

This instruction causes a full track of addresses and data (20 sectors) to be written in disk storage from core storage. The operation is the same as for Write Disk Track/WLRC, except that group marks do not affect transmission or checking. Group marks in the record in core storage are transmitted as data, group marks in disk storage are replaced by data in the record.

Execution Time. Average T = 62 ms

Check Disk Track
CTN or 36 — Q_{11} of 7

This instruction causes a full track of addresses and data (20 sectors) to be read from disk storage and compared with addresses and data in core storage. The operation is the same as for Check Disk Track/WLRC, except that group marks do not affect checking. Group marks in disk storage and core storage are compared as data.

Execution Time. Average T = 62 ms

Branch No Group Mark
BNG or 55

See Branch instructions.

Summary of the Effects of Group Marks in Disk Storage Operations

A group mark is used to establish record length when the Wrong-Length Record Check feature is used. The group mark must be placed in *core storage* following the last position of the record to be written in to disk storage, or following the last position of the area in core storage reserved for reading data from disk storage. Figure 49 is a block diagram illustrating the effects of group marks in disk storage operations. The effect of group marks is the same for both track mode and sector mode operations.

The following two points are programming situations that may arise, and the programmer should be aware of how they affect a disk storage operation.

1. If the group mark terminates a record that does not consist of an even increment of 100 positions (i.e., 85, 145, 470, etc.) the group mark is *written in disk storage* and the Wrong-Length Record Check indicator turns on. When the same record is read back into core storage, the group mark in *disk storage* stops the operation and turns on the Wrong-Length Record Check indicator.
2. If 300-position records (for example) are to be used in a program, and the sector count in the disk control field has been set to 3, but the group mark has been incorrectly placed in position 201 of the record instead of in position 301; then, during a *write* operation, the group mark in core storage stops the operation and the Wrong-Length Record Check indicator is turned on, but the group mark is *not written in disk storage.* In this example, although the group mark had *not* been placed in the correct position, it did establish a record in an increment

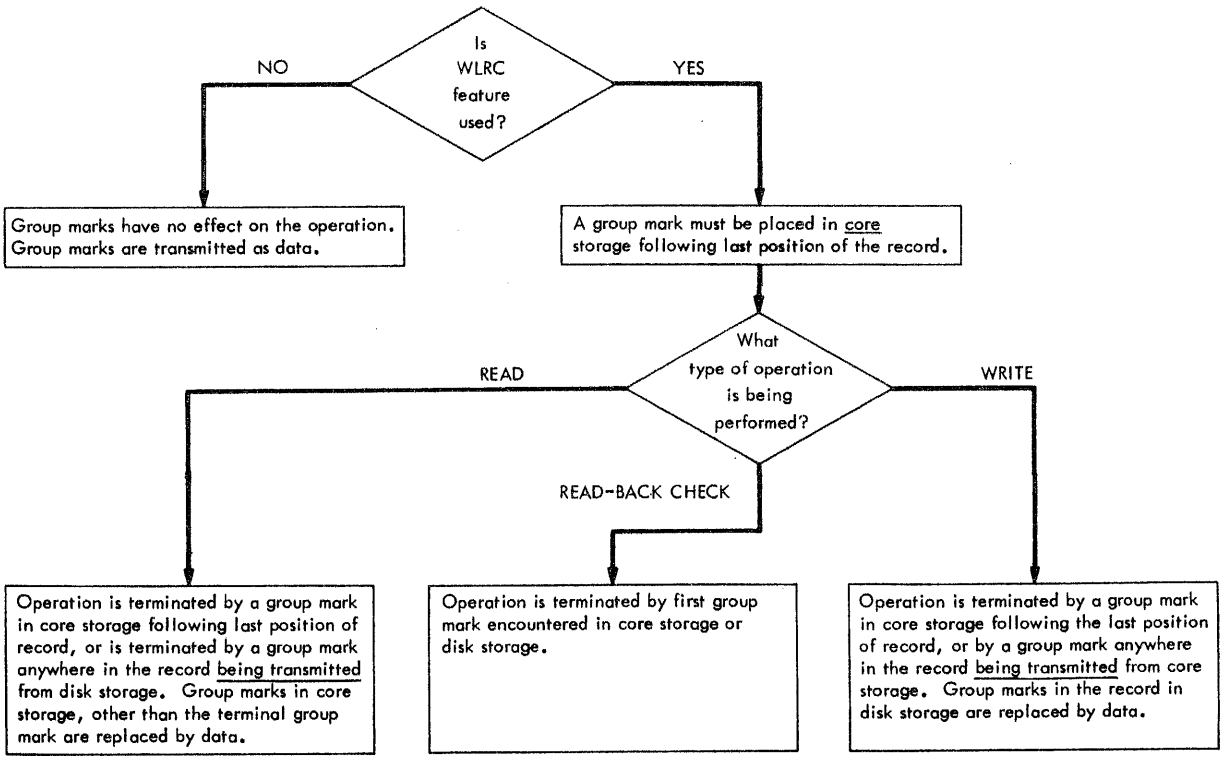


Figure 49. Effects of Group Marks on Disk Storage Operations

of 100 (i.e., 200-position record) and, therefore, the group mark was *not* written in disk storage.

Program Control Instructions

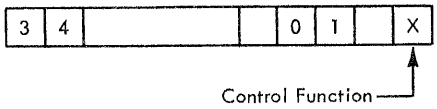
Control (K-34)

Description. The Control instruction is used to space, tabulate, and return the carriage of the 1620 Console Typewriter. The Q_8 and Q_9 digits must contain the typewriter code, 01, and the Q_{11} digit must specify the control code, as follows:

1 - Space
2 - Return Carriage
8 - Tabulate

The P address and the Q_7 and Q_{10} positions of the instructions are not used.

Execution Time. Execution time depends on the control function.



Seek (SK-34)

See 1311 Disk Storage Drive 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 maintain correct parity. The original digit at the P address remains unchanged. The Q part of the instruction is not used.

Execution Time. T = 200 μsec.

Clear Flag (CF-33)

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

Execution Time. T = 200 μsec.

Halt (H-48)

Description. The Halt instruction stops the 1620 in manual mode. With the computer in manual mode, if the Start key is pressed, 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. T = 160 μsec.

No Operation (NOP-41)

Description. This instruction performs no operation and advances the computer to the next instruction in sequence. The P and Q parts of the instruction are not used.

Execution Time. T = 160 μsec.

Console Typewriter

The 1620 console typewriter (Figures 50 and 51) is the principal communicating link between the operator and the system.

Typewriter Input

The 1620 console typewriter is used to enter data and instructions directly into core storage. Off-line use is not possible because the keyboard (Figure 51) is locked except when entering data. Pressing the console Insert key (see Figure 50, I) unlocks the keyboard and permits data to be entered into core storage, starting at location 00000. Each depression of a typewriter key enters a 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. The Start key may then be used to start program operation at 00000.

When less than 100 characters are entered, entry of the last desired character should be followed by pressing the console Release and Start keys, or by pressing 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.

Programmed selection of the typewriter (Read Alphamerically or Read Numerically instructions) unlocks the keyboard, leaving the computer in automatic mode for manual entry of data on the typewriter. Data entry starts at the address-numbered location (P address) of the instruction and enters core storage at successively higher-numbered locations until the Release key is pressed.

It should be noted that the decimal point or period character in either upper shift or lower shift may be entered only with a Read Alphamerically instruction. If entry is made with a Read Numerically instruction, incorrect data is put into core storage.

If a record mark is required in core storage following the last character entered, the Record Mark key on the typewriter must be pressed before pressing either the R-S key on the typewriter or the Release key on the console (Figure 50, R). Pressing either key again locks the keyboard and gives the computer an end-of-input/output indication.

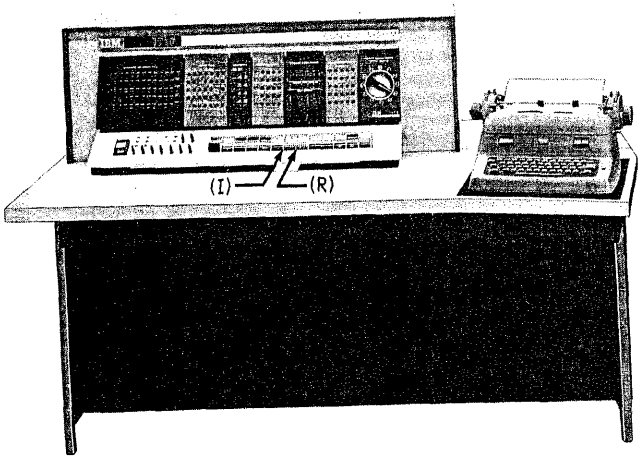


Figure 50. IBM 1620 CPU with I/O Typewriter

Typewriter Output

The typewriter prints data from core storage when it is 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 pressed. The Release key may be used, for example, 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 illuminates the console Read Check light and turns on the Read Check indicator (06).

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 (07) and the console Write Check light. Also, 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 ocurs, the input or output operation continues until completion, and the machine either stops or continues under program control, depending on the position of the console I/O Check switch.

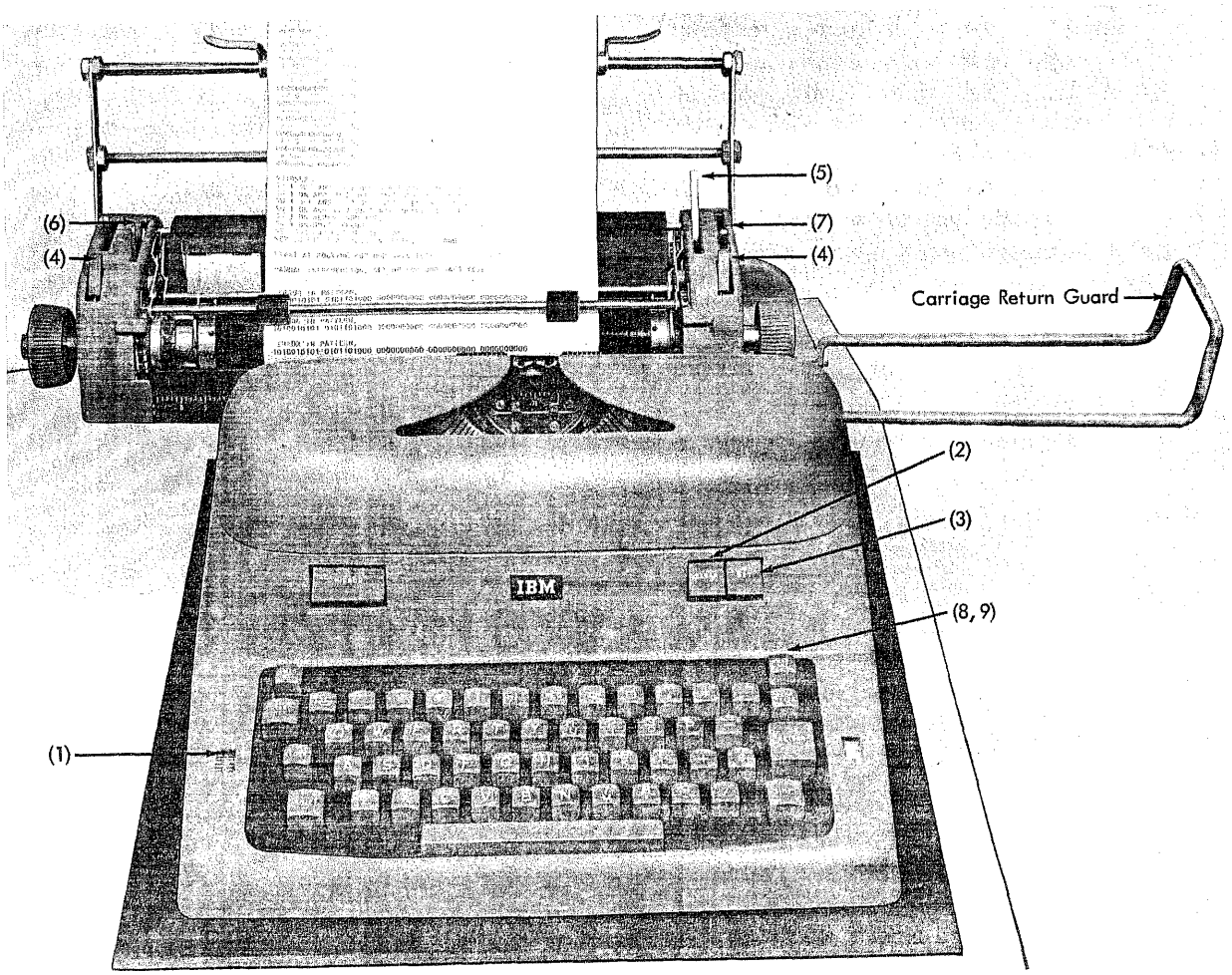


Figure 51. I/O Typewriter, Manual Controls and Keys

Manual Adjustments

The numbers preceding each of the following headings refer to numerals on Figure 51 designating particular keys, etc., (1), for example, designates the Impression indicator.

1. Impression Indicator. The lever under this window can be positioned in settings from 0 to 10, and determines 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 (7) 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 press the Tab Clear lever. To clear all stops at once, position the carriage at the right margin, hold down the Tab 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 press the Tab Set lever. Set tab stops only when the indicator pointer and the white marking on the front paper scale below it are aligned.
4. Carriage Release Lever. Press the lever on either side to free the carriage and move the carriage manually to the right or to the 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 platen is too far forward. The slash symbol (/) is a good character to use in checking multiple copy settings.

8. Left-Hand Margin Set. The left margin stop is set as follows:
   a. Return the carriage to the present left margin stop.
   b. Press and hold down the Margin Set key.
   c. Move the carriage manually as near as possible to the position desired. With the Margin Set key pressed, use the Back Space key and Space Bar to obtain the exact position desired.
   d. Release the Margin Set key.
9. Right-Hand Margin Set. The right margin stop is set as follows:
   a. Move the carriage to the left until stopped by the right margin stop.
   b. Press and hold down the Margin Set key.
   c. Move the carriage right or left to the desired position.
   d. Release the Margin Set key.

The 1620 console (Figure 52) 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

Control Switches, Keys, and Signal Lights

Control keys (Figure 52) 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 the step of the keying procedure last completed.

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 temperatures and voltage reach proper operating values. There is a delay from 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.

Automatic and Manual Lights. When the Manual light is on, it indicates that the computer is in manual mode. In manual mode, the computer has terminated all operation and is prepared to accept operator intervention. The Manual light is off when the computer is in automatic mode.

When the Automatic light is on, it indicates that the computer is in automatic, e. g., while executing

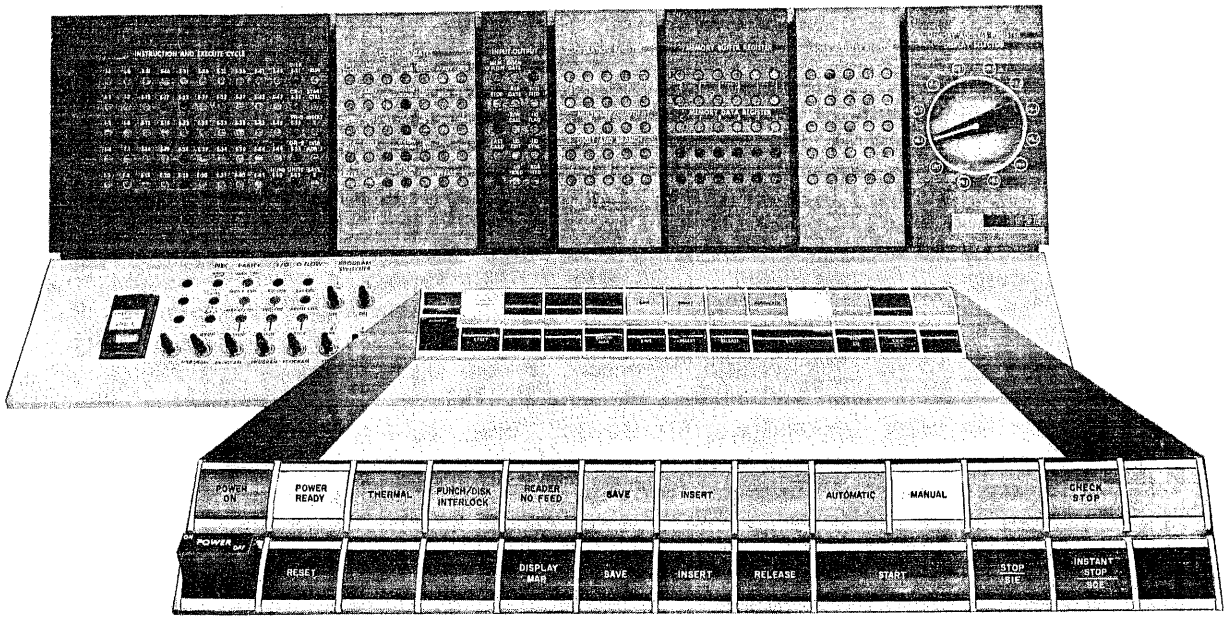


Figure 52. 1620 Control Keys and Signal Lights

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 pressing the Release key (on an I/O operation only), Instant Stop key, or Stop key. Pressing the Start key, the Insert key, or Display MAR key initiates automatic mode and turns off the Manual light. The Save 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 mask all interrupts (1710 Control System) and to restore all machine status indicators, machine check indicators, and signal lights to their initial or reset condition. The Reset key can be used only when the computer is in the manual mode. When the computer is in the automatic mode, the Reset and Release keys can be pressed simultaneously to effect a reset.

Insert Key and Insert Light. Depression of the Insert key places the 1620 in automatic mode. Pressing the Insert key also turns on the Insert light and activates the typewriter keyboard so that direct entry of up to 100 characters may be made in numeric mode, starting at 00000 and continuing into higher-numbered storage positions. The Insert key is operative only when the computer is in the manual mode.

Save Key and Save Light. Pressing 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.

Release Key. When this key is pressed, 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 is performing an I/O operation.

Stop/SIE (Single Instruction Execute) Key. Pressing the Stop/SIE key stops the computer in manual mode at the end of the instruction being executed when the key is pressed.

The Stop/*SIE* key also serves as a single instruction 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. Pressing the Instant Stop/SCE key causes the machine to stop at the end of the 20 μsec machine cycle in progress when the key is pressed. Successive depressions of the key cause single machine cycles. Both the 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 operates only when the Manual light is on and the Automatic light is off. Pressing 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 pressed.

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.

Punch/Disk Interlock Light. This 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 card read and card punch operations because the buffer is interlocked while the read and punch cycles are in progress.
5. The 1311 Disk Storage Drive is not ready.
6. The 1443 Printer is not ready.

Any of these conditions stops the computer in automatic mode with both the Automatic and Punch/Disk Interlock lights turned on. When a parity error occurs, the I/O Write Check light is also turned on. Pressing the Release Key disconnects the punch and the disk storage drive and puts the computer in manual mode. Pressing the Reset key while in manual mode turns off the Punch/Disk Interlock and I/O Write Check lights. Manual correction and restart procedures can begin after depression of the Release and Reset keys.

In disk storage operations, the Release and Reset keys must be depressed simultaneously.

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 pressing 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 pulled to OFF, all power in the machine is turned off and the blowers that cool the electronic circuits are stopped as well. Turning the blowers off in this manner may result in damage to the machine.

Program Switches and Indicators

Small incandescent lights are used to represent the ON and OFF conditions of internal check indicators. Eight console switches (four Program and four Machine Check) 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 operation may be altered by the condition of a machine check indicator and an associated check switch (Figure 53). 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.

Pressing the Reset key turns all check indicators and lights off. Disk storage, Parity, I/O, and Overflow Check indicators are provided.

Disk Storage Indicators

Three disk storage indicators and lights are used when the 1311 Disk Storage Drive is attached to the system.

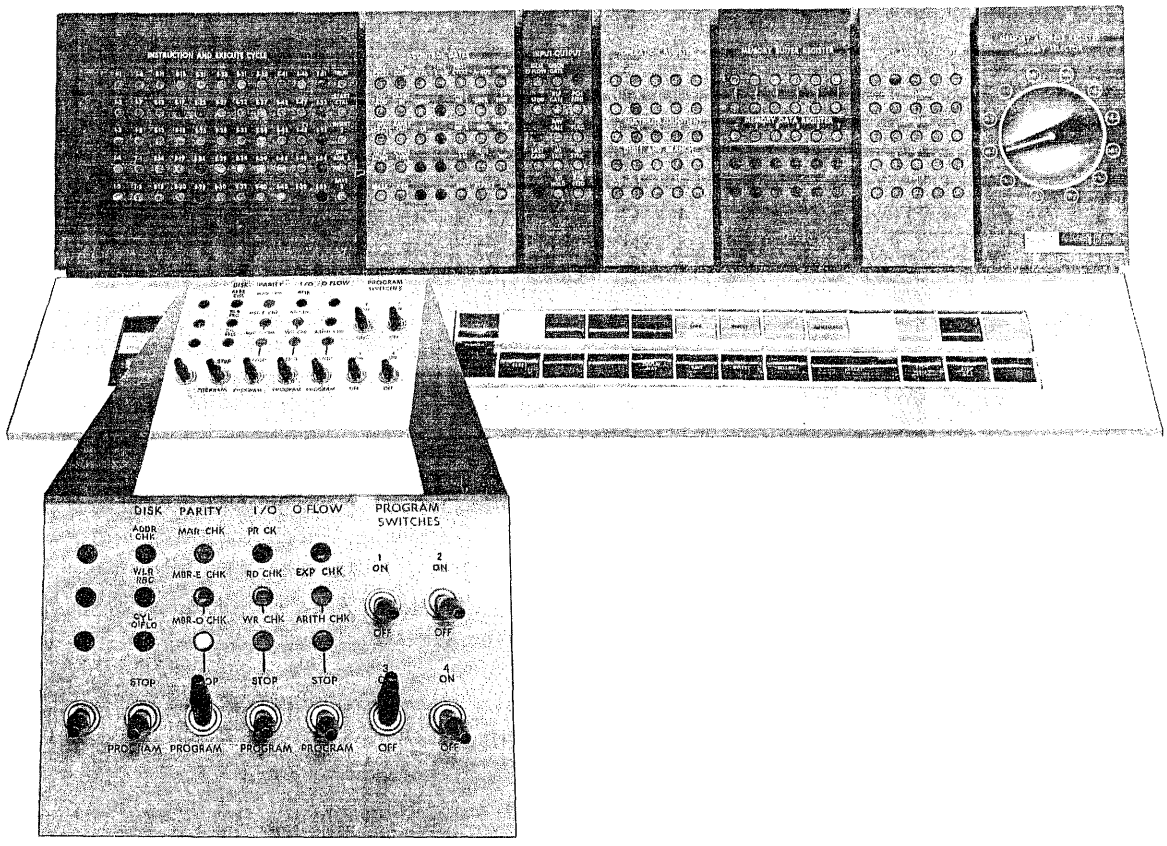


Figure 53. Indicator Lights and Switches

ADDR CHK (Address Check) Indicator. If the disk address specified in the disk storage instruction (except Read Disk Track) is not found on the designated track within one complete revolution of the disk pack, the operation is terminated and the Address Check indicator and light are turned on. The indicator and light are also turned on if more than one sector is specified in the operation, and the address of the second sector or the address of any additional sector in physical sequence fails to compare to its corresponding incremented address in the Disk Control Field in core storage.

This indicator is also turned on when more than one disk drive or more than one read/write head is selected during a disk operation. If this condition exists, a Read Disk Track or a Seek operation causes the Address Check indicator to be turned on.

WLR/RBC (Wrong-Length Record/Read-Back Check) Indicator. This dual-purpose indicator is turned on by two conditions:

1. If a 000 sector count and a group mark fail to correspond at the end of a read, write, or check disk operation, or if data records read from disk storage or written to disk storage are not in even increments of 100 characters.
2. If, during a check disk operation, the data in disk storage does not compare character-by-character and bit-by-bit with the corresponding data in core storage.

CYL O'FLOW (Cylinder Overflow) Indicator. This indicator and light are turned on if a disk storage operation completes the reading or writing of the last sector on a cylinder without the sector count being decremented to 000. This condition terminates the operation.

A more complete description of the operation of these indicators is contained in the 1311 publication referred to on the cover page of this manual.

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 PROGRAM so that errors can be interrogated by programming.

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 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 is turned on when a digit in MARS has a parity error or an invalid address. These errors halt the machine immediately.

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 before conversion 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 from BCD to output code. The effect on machine operation as a result of detection of this parity error varies, depending on the output unit selected and the position of the I/O Check switch, as shown in Table 5.

PR CHK (Printer Check) Light. This light and indicator (code 25) are turned on by the 1443 Printer when either of two conditions occurs: (1) A character with an even number of bits is detected entering or leaving the print buffer, or (2) a 1443 Sync Check occurs due to improper type bar synchronization. Either condition and the I/O Check switch on STOP halts the 1620 *after* the line is printed.

Overflow Indicators

EXP CHK (Exponent Check) Indicator. This indicator is used during Automatic Floating-Point operations (special feature). An exponent overflow or underflow that occurs during an arithmetic operation using floating-point instructions turns on the Exponent Check indicator and light. When the Overflow Check switch is set to STOP and the Exponent Check indicator is turned on, the computer halts at the end of the instruction being executed. If the Start key is pressed, the Exponent Check indicator remains on and the computer continues to execute instructions.

When the Overflow Check switch is set to PROGRAM and the Exponent Check indicator is turned on, the computer continues to operate in the automatic mode. The indicator can be interrogated and turned off by the program. The indicator and light can be turned off by pressing the Reset key.

ARITH CHK (Arithmetic Check) Indicator. An overflow that occurs as a result of an add, subtract, divide, or compare operation turns on the Arithmetic Check

Table 5. Write Check Errors

Output Device	I/O Check Switch Position	
	Stop	Program
Typewriter	1620 halts after printing all characters from transmitted record.	Output operation is completed and 1620 can branch to an error subroutine.
Tape Punch	1620 halts as soon as an erroneous character is punched. Tape does not advance.	1620 halts as soon as an erroneous character is punched. Tape does not advance.
Card Punch	1620 halts after core-storage to buffer storage transfer. Card is not punched.	Output operation is completed and 1620 can branch to an error subroutine.
1311 Disk Storage	1620 halts at the end of a sector, or at the end of the last sector on a track if a complete track is being read.	Same as above.

indicator and light. When the Overflow Check switch is set to STOP and the Arithmetic Check indicator is turned on, the computer halts at the end of the instruction being executed. If the Start key is pressed, the Arithmetic 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 Arithmetic 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

The modifier switches in this group, labeled PROGRAM SWITCHES on the console, are numbered 1 through 4. A branch occurs when a switch specified by a Branch Indicator instruction is set to ON.

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

When a Branch No Indicator 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 small incandescent lights that represent the bits present in each digit of a register (Figure 54). 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 are displayed in the MBR. When the core storage location addressed for display is an even-numbered adress, 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 alphameric mode, the complete two-digit representation of an alphameric character may be viewed at one time.

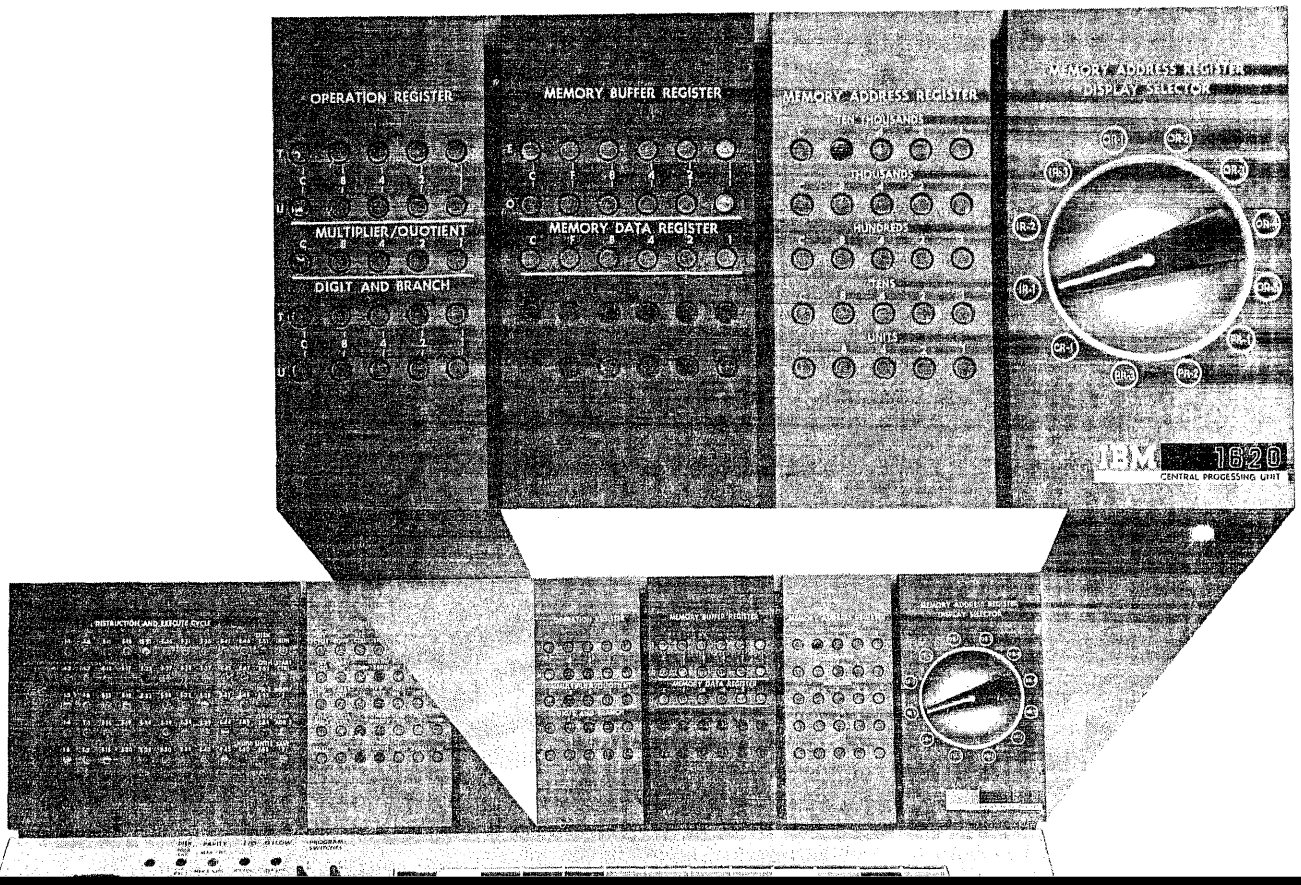
Memory Data Register (MDR). One line of six indicator lights displays the bit configuration of each digit in core storage as it is read out. These digits can be seen on single-cycle operation, using the 1620 console SCE key. The digit displayed in the MDR display is duplicated in the MBR-E or MBR-O display, depending on whether the digit read out is located at an even-numbered or an odd-numbered core storage location.

Operation (Op) Register. Two lines, each with five lights, 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.

Digit and Branch. Two lines, each with five lights, display the contents of the Digit and Branch Register. This register serves a dual purpose in the 1620:

1. It decodes the Q_8 and Q_9 digits of Branch Indicator, Branch No Indicator, and Input/Output instructions.
2. It temporarily stores digits affecting MARS (Memory Address Register Storage) during all I cycles, and stores partial product digits during multiplication.

Multiplier/Quotient. This five-light register display shows each multiplier digit as it is used during a mul-


OPERATION REGISTER
MULTIPLIER/QUOTIENT
DIGIT AND BRANCH
MEMORY BUFFER REGISTER
MEMORY DATA REGISTER
MEMORY ADDRESS REGISTER
TEN THOUSANDS
THOUSANDS
HUNDREDS
TENS
UNITS
MEMORY ADDRESS REGISTER
DISPLAY SELECTOR
IBM 1620
CENTRAL PROCESSING UNIT

Bypass. The Bypass light shows that the MAR address neither decreased nor increased. When this light is on, either the Increment or Decrement light is also on.

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

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

Plus 2. The Plus 2 light shows that the address routed from MAR to MARS increases by 2. When this light is on, the Increment light must also be on.

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 the 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 was placed 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.

I/A (Indirect Addressing). The Indirect Addressing light shows that an indirect addressing operation (special feature) is in process. 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.

COMP (Complement). When the Complement light is on, it shows that the Q address data is complemented during arithmetic operations.

Mask. (1710 Control System only). The Mask light is turned on by the execution of a Mask instruction, and is turned off by the execution of an Unmask instruction. This light is also turned on when the 1620 Reset key is pressed, or by a Power-on operation.

Interrupt Mode. (1710 Control System only.) The Interrupt Mode light is turned on only when an interrupt is recognized. It is on while the interrupt is being serviced, and is turned off by the execution of a Branch Out of Noninterruptible Mode or a Branch Out of Noninterruptible Mode and Load instruction. In effect, the Interrupt Mode light is on when IR-3 is used in place of IR-1.

Instruction and Execute Cycle Lights

The Instruction and Execute Cycle lights (Figure 56) are visual aids 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 repeated depressions of this key.

Input/Output Lights

The Input/Output lights are primarily used by IBM Customer Engineers for diagnostic testing.

LC (Last Card). The Last Card light has significance for the operator and programmer. 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.

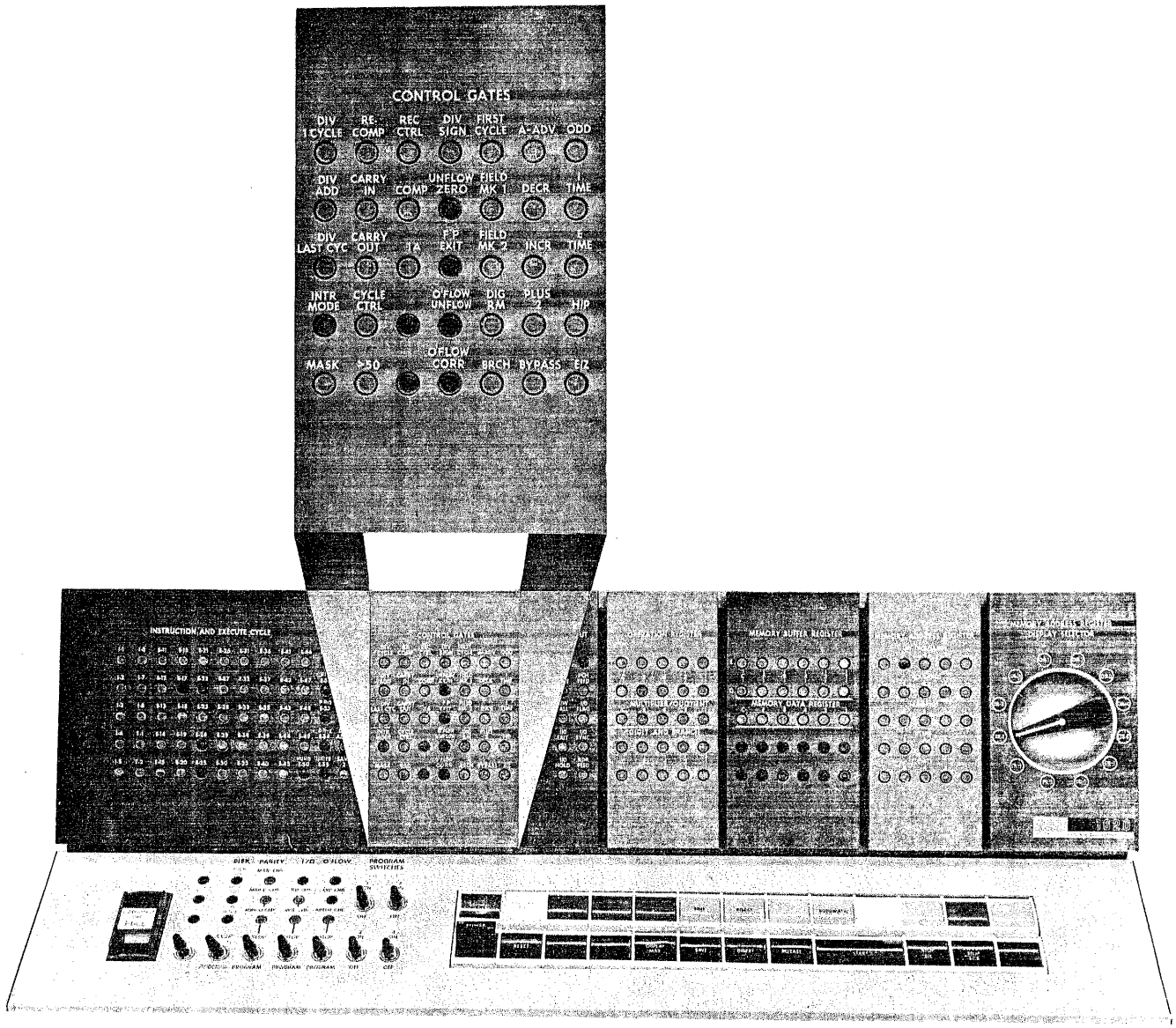


Figure 55. Control Gates Lights

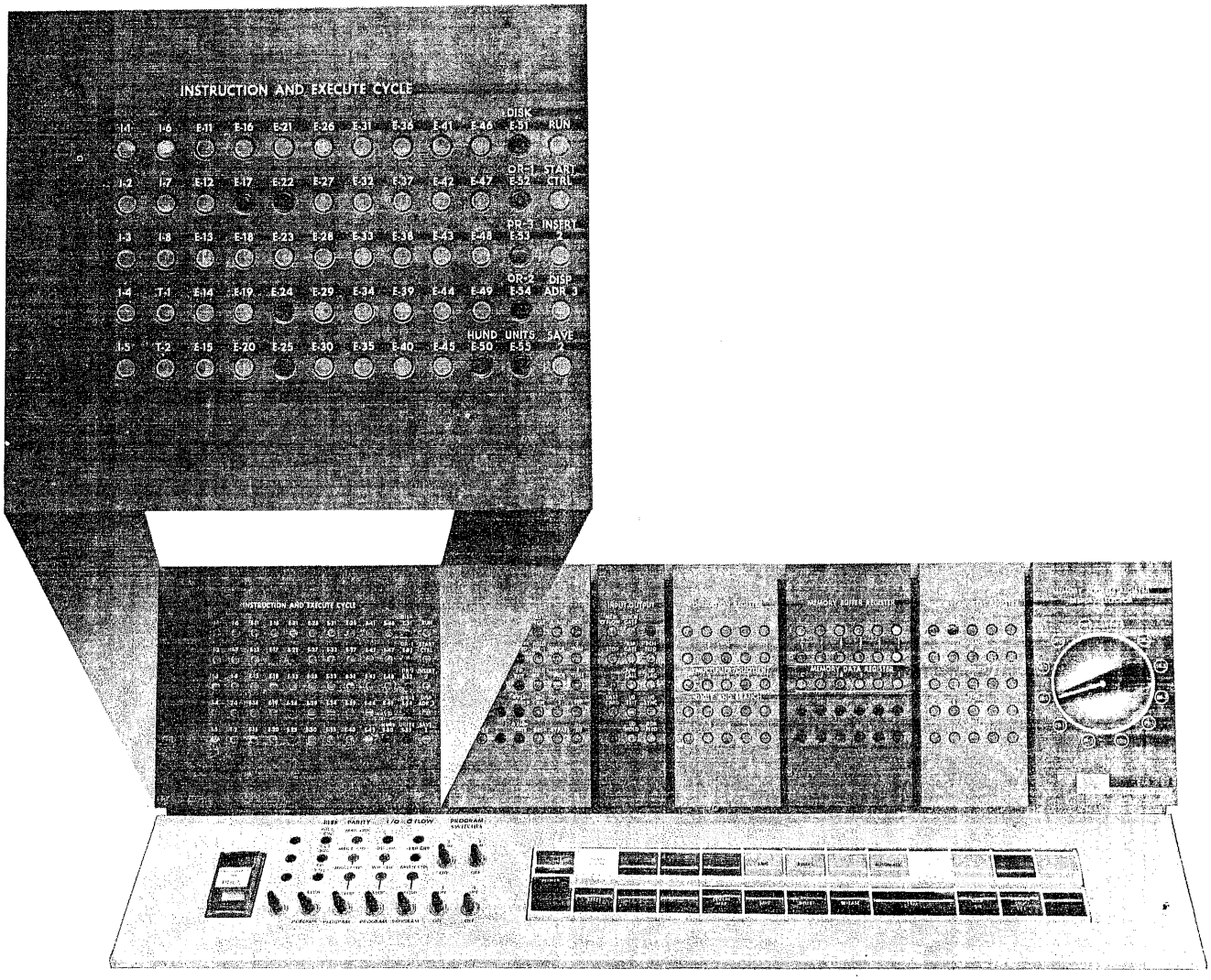


Figure 56. Instruction and Execution Cycle Lights

1620 Console Operating Procedures

The following procedures are included to aid the console operator. They may be modified to meet individual requirements.

Program Entry from Typewriter

	Operator Action	Explanation
1.	Press 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.	Press Release key.	Releases typewriter.
4.	Press 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-numbered core storage positions.
6.	Press Release key.	Terminates read instruction.
7.	Press 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.	Press 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.	Press Release key.	Release typewriter.
4.	Press 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.	Press Stop key.	Halts processing and initiates manual mode.
2.	Press Save key.	The address of the next instruction in sequence is saved in Product Address Register 1 (PR-1).
3.	Press 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.	Press Release key.	Releases typewriter.
6.	Press 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-numbered core storage positions.
8.	Press Release key.	Terminates Read instruction.
9.	Press 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.

Typewriter Output

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

Check Program Step Sequence and Operation

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

Reset Storage to Zeros

	Operator Action	Explanation
1.	Press Stop key.	Halts processing and initiates manual mode.
2.	Press 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.	Press Release key.	The instruction entered in step 3 is located at 00000-00011. 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 of a second is required to clear 20,000 positions of core storage. A Transmit Field Immediate instruction or Transmit Record instruction may be used in the same manner.
6.	Press Instant Stop key.	The operation is stopped with the machine in manual mode.

Display P and Q Addresses

	Operator Action	Explanation
1.	Press Stop key.	Halts processing and initiates manual mode.
2.	Press SIE key until the instruction that contains the desired address is next.	One instruction is executed with each depression of the SIE key.
3.	Press the SCE key eight times.	This steps through the eight I cycles.
4.	Press Reset key.	Initiates manual mode.
5.	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.
6.	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.

Note: If it is desired to continue program operation after Display, the following actions should be taken:

1. Prior to Reset (step 4) the on/off conditions of machine status and check indicators should be noted.
2. After Display (step 6) a Branch back to some instruction in the program should be inserted. This instruction should be one that will go back to where the indicators reset (step 4) will not affect program operation.

If Reset and the subsequent necessity for branching and restart are undesirable, steps 4-6 can be omitted and the P and Q addresses can be viewed, two digits at a time, during step 3 as they move through MBR to OR-2 and OR-1. (See Program Testing section.)

Thorough documentation and preparation should precede the testing of all programs. Careful console operation is next in importance. The following console operating procedures are helpful in analyzing programs in the computer.

Instruction (I) Cycle

Pressing the Instant Stop/SCE key stops the computer at the end of the machine cycle in operation at the time the key is pressed. Successive depressions of the key step the computer through 20-μsec 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.

Pressing 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 data to print.

Figure 57 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-E and MBR-O and transmitted to the two-digit operation register. Simultaneously, the contents of

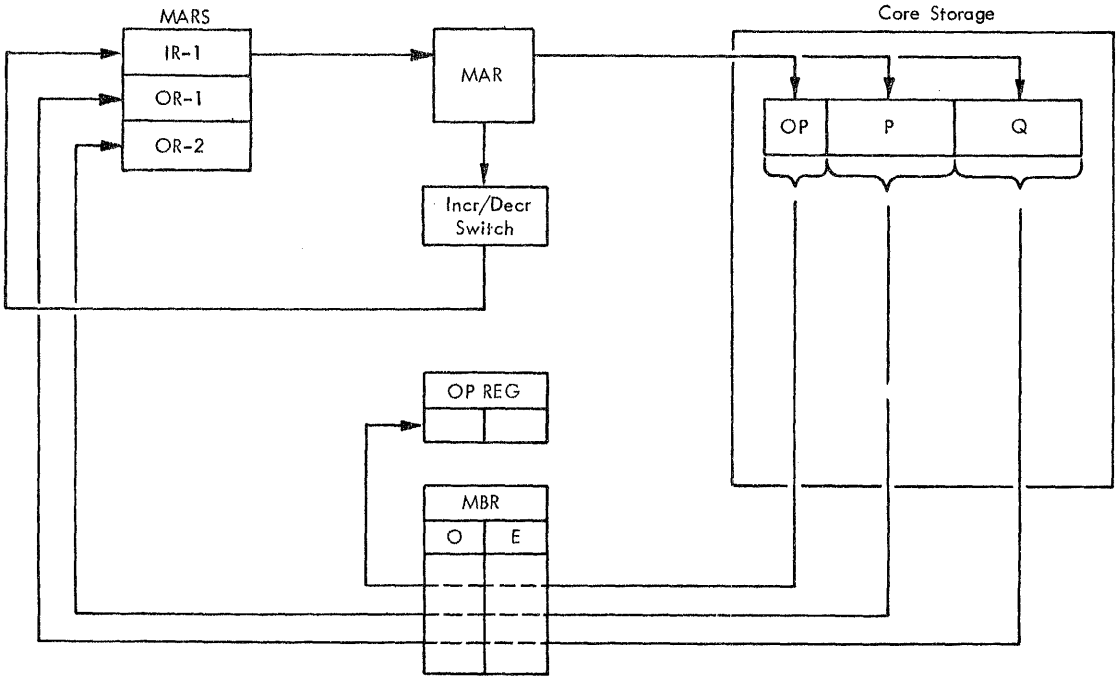


Figure 57. Diagram of I Cycle

MAR are 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 placed in the fifth and fourth high-order positions, respectively, 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 instruction in the program. It remains at this address during the following E cycle.

For immediate instructions, the following differences occur:

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. Register displays can be used for analysis at any point in an operation. The MARS registers can be displayed by using the following procedures.

1. Press the Reset key — this prevents further progression of the instruction being executed.
2. Position the MARS Display switch to select the desired register.
3. Press the Display MAR key.

During machine operation, MAR is addressed by the 12 MARS registers. 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 58.

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.

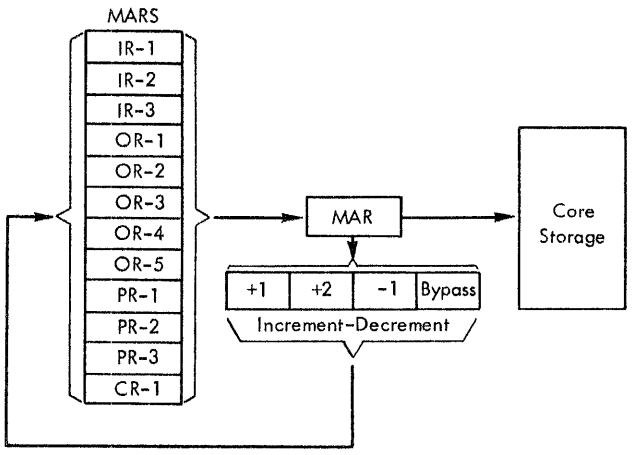


Figure 58. Diagram of E Cycle

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 Figure 59, which shows how the amounts 185 and 52 are added.

The circled numbers on Figure 59 correspond to the numbered steps that follow.

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 digit (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 ($\overline{5}$) goes into the units position of MAR.
7. The sum ($\overline{3}$), 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 ($\overline{1}$) goes into the tens position of MAR.

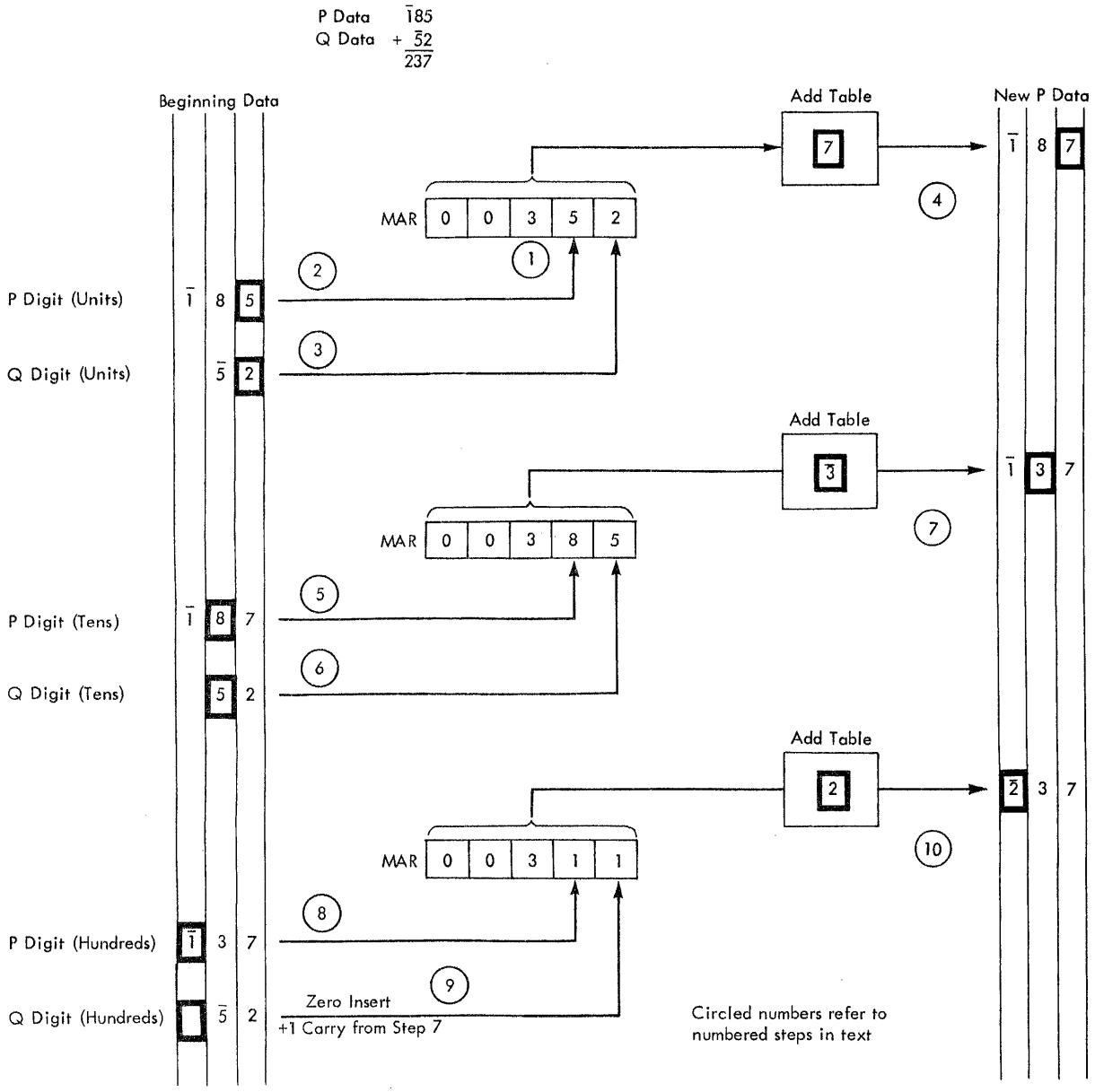


Figure 59. Add Operation

9. There is no Q hundreds digit, but a zero is inserted internally. The zero, together with the carry from Step 7, causes a 1 to go into the units position of MAR.
10. The sum (2) which is located at 00311 of the Add table, replaces the P hundreds digit.

Figure 60 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 and Branch 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.

The number 003 is automatically inserted in MAR for 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.

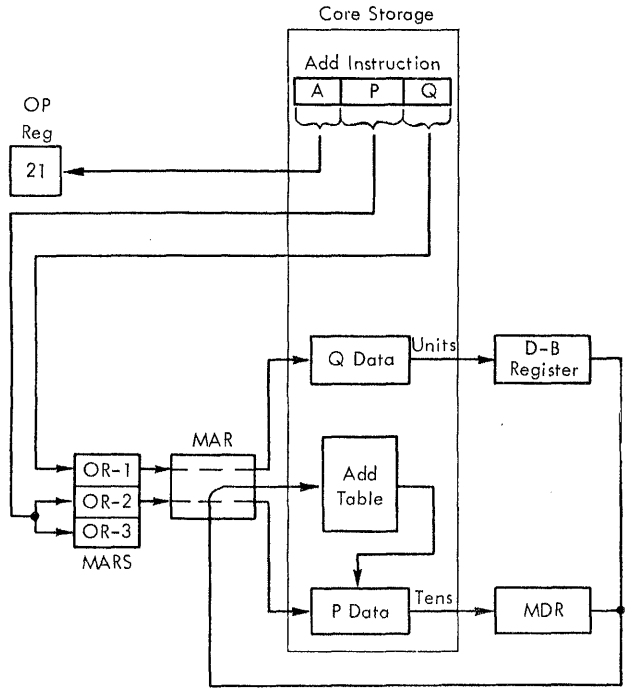


Figure 60. E Cycle of Add Operation

Subtraction

The following steps with reference to Figure 61 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.
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 00358 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 60) except that the Q digits are complemented as they are routed from the Digit and Branch register to MAR.

The console Complement light is turned on 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 B will aid in following the procedure shown in Figure 62, which shows how each multiplier digit is used with the multiplicand. Twelve 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 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 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.

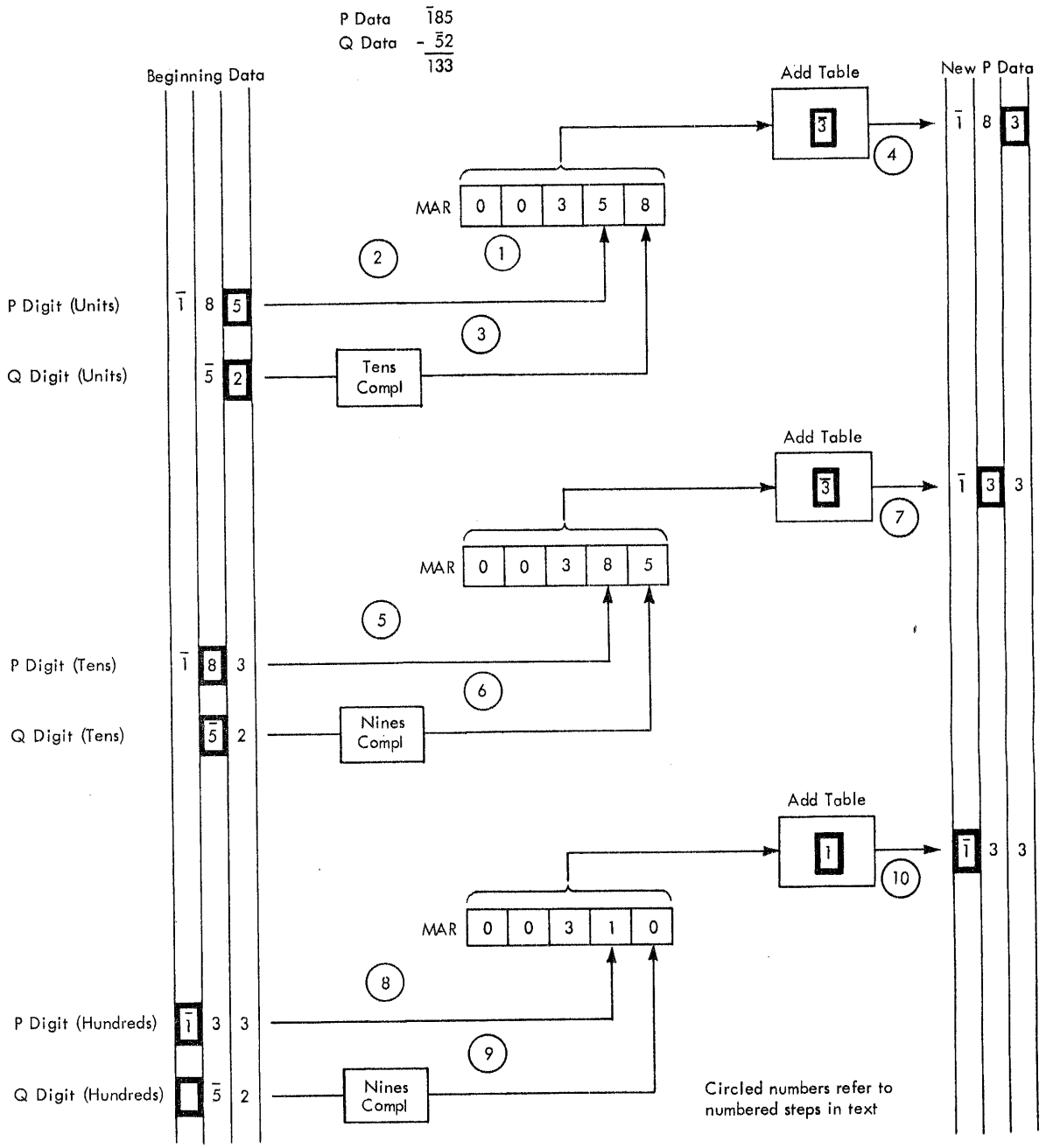


Figure 61. Subtract Operation

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 ($\overline{1}$) 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 ($\overline{1}$) goes into the tens position of MAR.
15. The Q tens digit ($\overline{1}$) is doubled, etc., as described in Step 3.

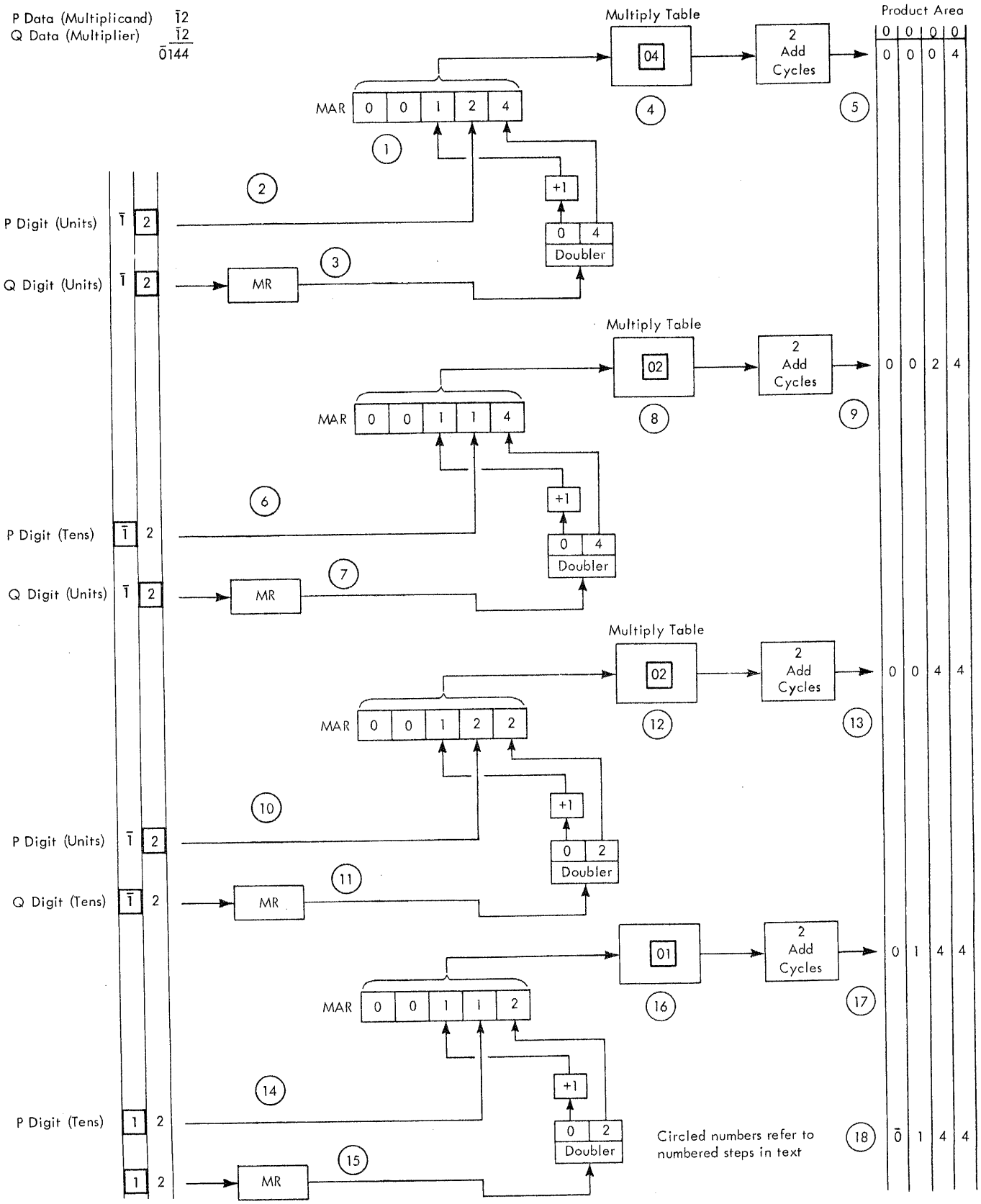


Figure 62. Multiply Operation

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 63, is as follows:

1. The multiplier digit is routed to the Multiply register and doubled. A one 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.
3. A 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.

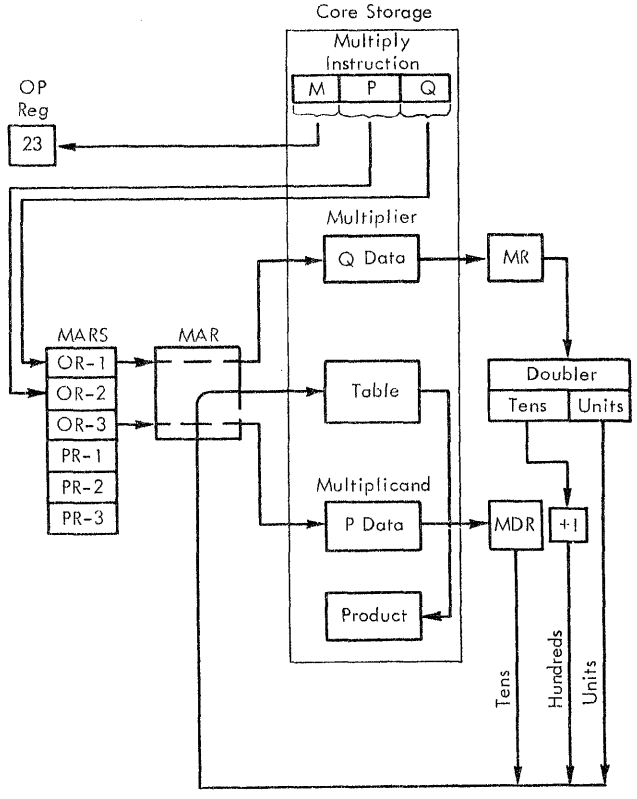


Figure 63. E Cycle of Multiply Operation

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.

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.

Internal Data Transmission

Figure 64 shows how the digit at the Q address of the Transmit Digit instruction is routed in through MDR, MBR, and to the P address of the instruction.

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/Output Data Flow

Input Data Flow

Figure 65 shows how data from the input device is routed through MDR and MBR to the address specified by MAR. Data enters succeedingly higher-numbered core storage positions until the operation is terminated.

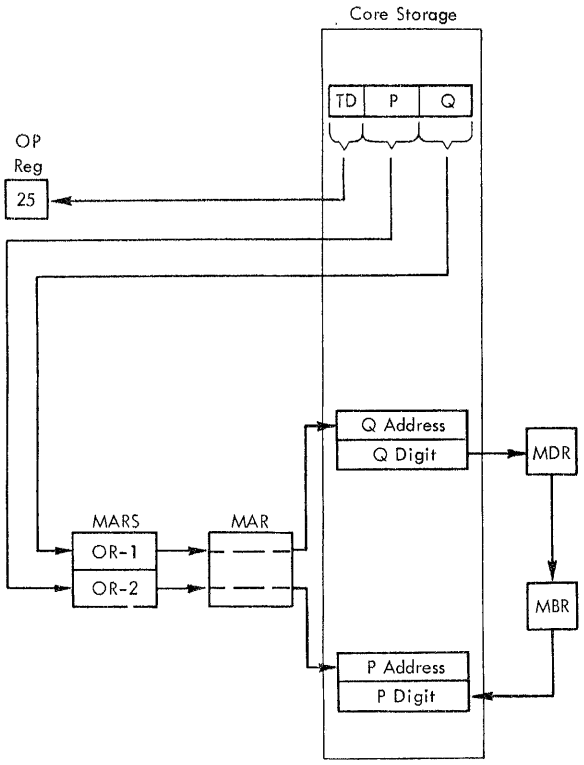


Figure 64. E Cycle of Transmit Digit Operation

Output Data Flow

Figure 66 shows how MAR causes the address of the first character to be routed through MBR and MDR to the output device. Succeedingly higher-numbered positions in core storage follow until a record mark or the 80th buffer storage position of the output card is reached.

1620 Data Flow

The data flow diagram shown in Figure 67 is used by Customer Engineers for diagnostic analysis of the 1620. As shown in the upper section of the diagram, core storage is addressed by MAR. Two digits, the addressed digit and its adjacent position, read out of core storage into the MBR-E and MBR-O registers. The addressed digit is transferred to MDR for processing. Both digits read back into core storage.

MAR receives core storage addresses from the twelve MARS registers. The function of each MARS register is given in Appendix C.

Internal Checking

The 1620 is a self-checking unit. As shown in Figure 67 (lower left corner), both input and output data are parity checked in the translators. Also, internal parity checking of data occurs in the MBR-E, MBR-O, and MAR registers.

Program Control

The Op register and decoding block are shown in the lower right corner of Figure 67. Program execution control lines, which result from decoding the instruction Op code, are initiated throughout the system as required.

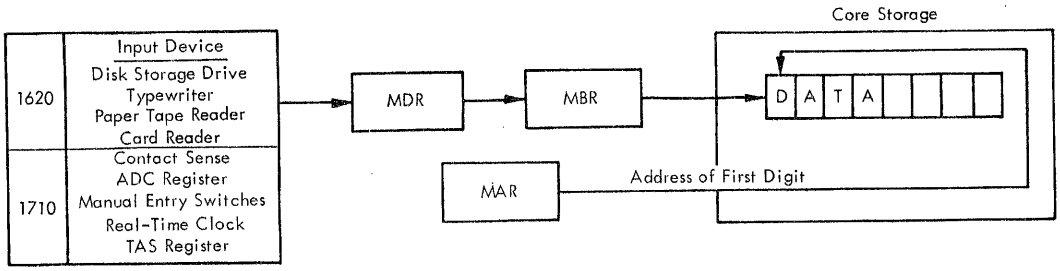


Figure 65. Data Flow of Input Operation

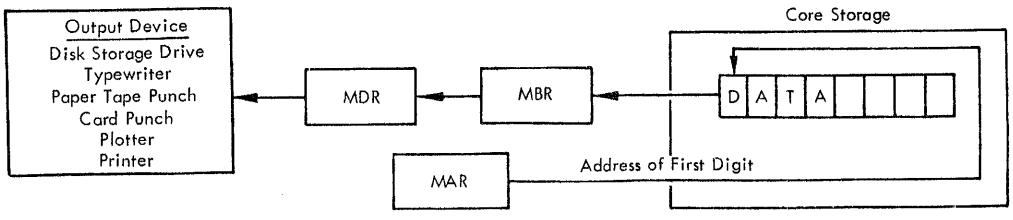


Figure 66. Data Flow of Output Operation

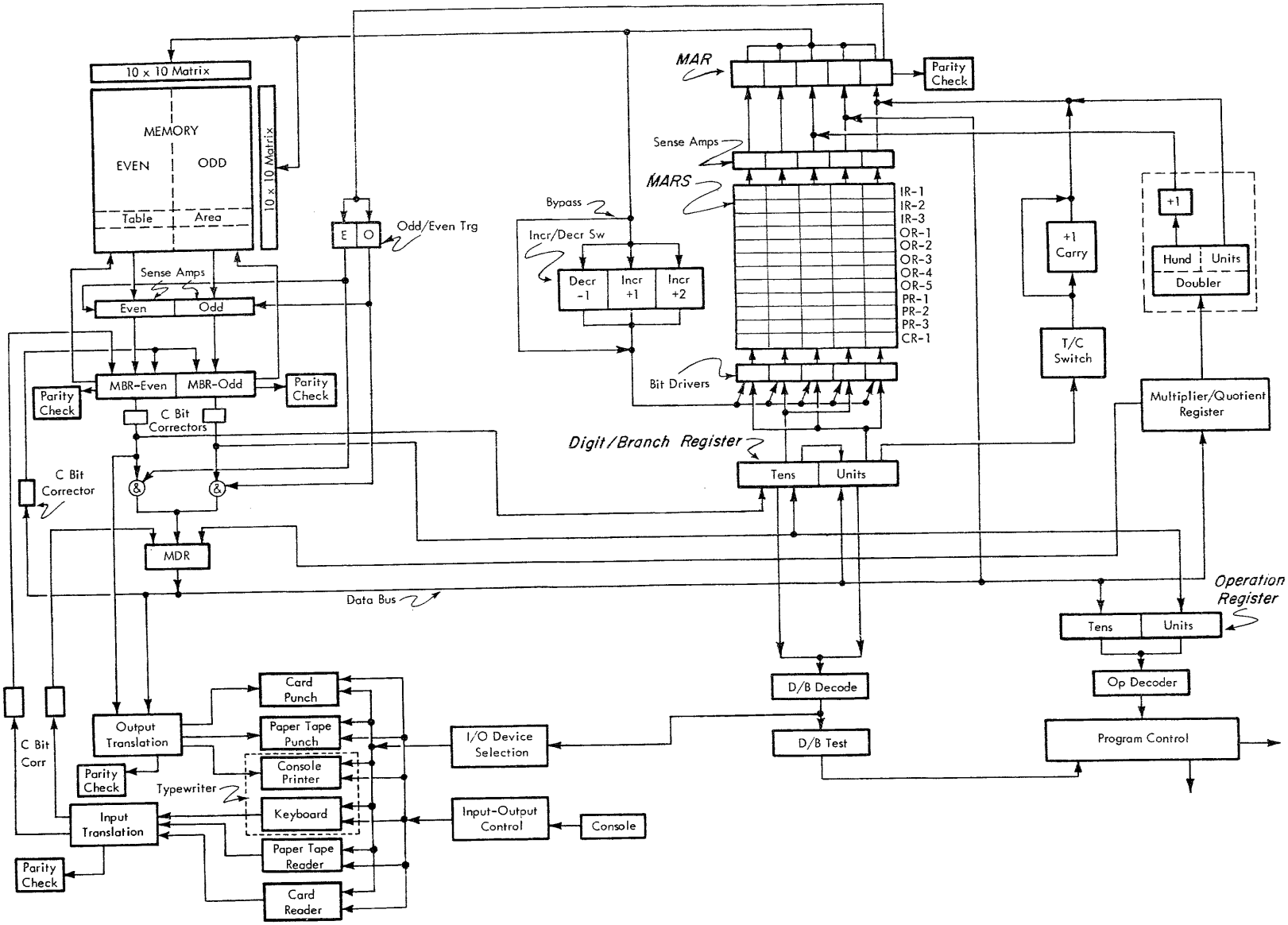


Figure 67. 1620 CPU Data Flow

Appendix A

Instruction Summary

Instruction	Operation Code	Operation Code Modifier	SPS Mnemonic	Instruction Time 1620 Model 1	Notes
Add	21		A	160 + 80D_p Basic Time 80 D_p recomp time	If signs initally unlike and numerical value of Q data is greater than P data.
Add (I)	11		AI	Same as Add	
Branch	49		B	200	
Branch and Transmit	27		BT	200 + 40D_q	
Branch and Transmit (I)	17		BTM	Same as above	
*Branch and Transmit Floating	07		BTFL	240 + 40L	
Branch Back	42		BB	200	
Branch Indicator	46		BI	160 No Branch 200 Branch	
Branch No Flag	44		BNF	200 No Branch 240 Branch	
Branch No Indicator	47		BNI	160 No Branch 240 Branch	
*Branch No Group Mark	55		BNG	200 No Branch 240 Branch	
Branch No Record Mark	45		BNR	200 No Branch 240 Branch	
Branch On Digit	43		BD	200 No Branch 240 Branch	
*Check Disk	36	Q_{11} = 1 Q_{11} = 3 Q_{11} = 5 Q_{11} = 7	CDGN CDN CTGN CTN	320 + 20,000 + 2,000S, Average Time (S = Number of Sectors)	Q_8-Q_9 must be 07 Check one sector - with WLRC Check one sector - no WLRC Check Full Track - with WLRC Check Full Track - no WLRC
Clear Flag	33		CF	200	
Compare	24		C	200 + 80 D_Z - unlike signs 160 + 80 Dp - like signs	D_Z = Number of positions compared until a digit other than zero is detected in either field.
Compare (I)	14		CM	Same as above.	
Control	34		K	Depends on control function and speed of I/O unit.	
*Divide	29		D	160 + 520 $D_V Q_T$+740Q_T	Avg quotient digit = 4.5
*Divide (I)	19		DM	Same as above	Same as above
Dump Numeric	35		DN	Depends on speed of I/O unit and number of characters involved.	
*Floating Add	01		FADD	400 + 100L Basic Time 800L Recomp. Time	If signs initially unlike and numerical value of Q data is greater than P data.
*Floating Divide	09		FDIV	880 + 940L + 520L^2	Avg quotient digit = 4.5
*Floating Multiply	03		FMUL	1120 + 80L + 168L^2	
*Floating Shift Left	05		FSL	200 + 40L + 40L^1	L^1 = Number of digits mantissa is increased by shift left.
*Floating Shift Right	08		FSR	200 + 40L	

D_P = Number of digits, including high-order zeros, in the field at P
D_Q = Number of digits, including high-order zeros, of Q.
Q_T = Number of digits, including high-order zeros, in the quotient.
D_V = Number of digits, including high-order zeros, in the divisor.
^{1} = Number of digits in mantissa.

(I) = Immediate
ms = Milliseconds
WLRC = Wrong Length Record Check
* = Special Feature

Instruction Summary

Instruction	Operation Code	Operation Code Modifier	SPS Mnemonic	Instruction Time 1620 Model 1	Notes
*Floating Subtract	02		FSUB	400 + 100L Basic Time 80L Recomp. Time	If signs initially alike and numerical value of Q data is greater than P data.
Halt	48		H	160	
*Load Dividend	28		LD	400 + 40 D_N	D_N = Number of digits, including High-order zeros in the dividend.
*Load Dividend (I)	18		LDM	Same as above.	
*Move Flag	71		MF	240	
Multiply	23		M	560 + 40 D_Q + 168 $D_P D_Q$	
Multiply (I)	13		MM	Same as above	
No Operation	41		NOP	160	
Read Alphameric	37		RA	Card I/O 3.4 ms All other: See notes	Time for other units depends upon speed of unit and number of characters involved.
*Read Disk	36	Q_{11} = 0 Q_{11} = 2 Q_{11} = 4 Q_{11} = 6	RDGN RDN RTGN RTN	320 + 20,000 + 2,000S Average Time S = Number of Sectors	Q_8 - Q_9 must be 07 Read sectors - with WLRC Read sectors - no WLRC Read Full Track - with WLRC Read Full Track - no WLRC
Read Numerical	36		RN	Card I/O: 3.4 ms All other: See Notes	Time for other units depends upon speed of unit and number of characters involved.
Set Flag	32		SF	200	
Subtract	22		S	160 + 80 Dp Basic Time 80 D_P Recomp. Time	If signs initially alike and numerical value of Q data greater than P data.
Subtract (I)	12		SM	Same as above	Same as above
*Transfer Numerical Fill	73		TNF	160 + 40 Dp	
*Transfer Numerical Strip	72		TNS	160 + 40 D_P	
Transmit Digit	25		TD	200	
Transmit Digit (I)	15		TDM	200	
Transmit Field	26		TF	160 + 40 D_Q	
Transmit Field (I)	16		TFM	Same as above	
*Transmit Floating	06		TFL	240 + 40L	
Transmit Record	31		TR	160 + 40 D_Q	
Write Alphameric	39		WA	Card I/O: 3.4 ms All other: See notes	Time for other units depends upon speed of unit and number of characters involved.
Write Disk	38	Q_{11} = 0 Q_{11} = 2 Q_{11} = 4 Q_{11} = 6	WDGN WDN WTGN WTN	320 + 20,000 + 2,000S Average Time (S = Number of Sectors)	Q_8 - Q_9 must be 07 Write sectors - with WLRC Write sectors - no WLRC Write Full Track - with WLRC Write Full Track - no WLRC
Write Numerical	38		WN	Card I/O: 3.4 ms All other: See notes	Time for other units depends upon speed of unit and number of characters involved.

D_P = Number of digits, including high-order zeros, in the field at P
D_Q = Number of digits, including high-order zeros, of Q.
Q_T = Number of digits, including high-order zeros, in the quotient.
D_V = Number of digits, including high-order zeros, in the divisor.
L = Number of digits in mantissa.

(I) = Immediate
ms = Milliseconds
WLRC = Wrong Length Record Check
* = Special Feature

Appendix B

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

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	$\bar{0}$
0032	2	3	4	5	6	7	8	9	$\bar{0}$	$\bar{1}$
0033	3	4	5	6	7	8	9	$\bar{0}$	$\bar{1}$	$\bar{2}$
0034	4	5	6	7	8	9	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$
0035	5	6	7	8	9	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$	$\bar{4}$
0036	6	7	8	9	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$	$\bar{4}$	$\bar{5}$
0037	7	8	9	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$	$\bar{4}$	$\bar{5}$	$\bar{6}$
0038	8	9	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$	$\bar{4}$	$\bar{5}$	$\bar{6}$	$\bar{7}$
0039	9	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$	$\bar{4}$	$\bar{5}$	$\bar{6}$	$\bar{7}$	$\bar{8}$

Appendix C

Bit Configuration

Digit	Bit Configuration					
	C	F	8	4	2	1
0	X					
1						X
2					X	
3	X				X	X
4				X		
5	X			X		X
6	X			X	X	
7				X	X	X
8			X			
9	X		X			X

Indicator Codes

Code	Name
01-04	1620 Program Switches 1-4
06	Read Check
07	Write Check
09	Last Card
11	High/Positive (H/P)
12	Equal/Zero (E/Z)
13	H/P or E/Z
14	Arithmetic Overflow Check
*15	Exponent Check
16	MBR-E Check
17	MBR-O Check
19	Any Check
*36	Address Check
*37	Wrong-Length Record/ Read-Back Check
*38	Cylinder Overflow
*39	Any Disk Indicator

*Special Feature

Table Areas in Core Storage

Address	Area
00000-00099	Console Area
00080-00099	Product Area
00100-00299	Multiply Table
00300-00399	Add Table

1620 Storage Register Functions

IR-1	Contains address of next instruction if machine is stopped with Stop key or Halt instruction. Saves return address when interrupt is serviced.
IR-2	Saves return address when BT and BTM instructions are executed.
IR-3	Contains interrupt address — used in place of IR-1 during interrupt program operation
IR-4	Saves return address when any Branch and Transmit instruction is executed in the Interrupt Program (1710 Control System only).
OR-1	Contains Q address after I cycle of an instruction. In disk storage operations, used to store and control disk sector address.
OR-2	Contains P address after I cycle of an instruction. In disk storage operations, contains core storage address where data from disk storage is written to or read from.
OR-3	Retains address of low-order multiplier digit during multiplication.
OR-4	Used to store and control the exponent address E_q during automatic floating-point operations.
OR-5	Used to store and control the exponent address E_p during automatic floating-point operations.
PR-1	Saves return address when a Save key operation occurs. Decremented for each new multiplier 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. In disk storage operations, used to store and control number of sectors in operation.
MAR	Addresses core storage.
MBR	Receives digits entering or leaving core storage.
MDR	Receives addressed digit entering or leaving core storage.
OP	Contains Op code of instruction just executed if machine is stopped with Stop key or Halt instruction.
CR-1	Used to store the algebraic difference between E_p and E_q for determination of decimal alignment during automatic floating-point operations. CR-1 is also used during floating-point operations to count high-order zeros when normalizing— the contents of CR-1 are subtracted from E_p.
Multiplier/Quotient	Contains multiplier and quotient digits during multiply and automatic divide operations.
Digit and Branch	Decodes Q_8 and Q_9 digits of BI, BNI, and I/O instructions. Stores partial product digits during multiply instructions. Stores digits affecting MARS during all I cycles.

Appendix C (cont'd.)

Character Coding

ALPHABETIC MODE

Character	Input					Numeric Code	Core Storage		Output					
	Typewriter	Paper Tape	Disk Storage (1) Zone	Disk Storage (1) Numeric	Card		Zone	Numeric	Typewriter	Paper Tape	Disk Storage (1) Zone	Disk Storage (1) Numeric	Card	Printer
(blank)	(space)	C	C	C	blank	00	C	C	space	C	C	C	blank	blank
.(period)	.	X0821	C	C21	12-3-8	03	C	C21	.	X0821	C	C21	12-3-8	.
)	)	CX084	C	4	12-4-8	04	C	4	)	CX084	C	4	12-4-8	)
+	+	CX0	1	C	12	10	1	C	+	CX0	1	C	12	+
$	$	CX821	1	C21	11-3-8	13	1	C21	$	CX821	1	C21	11-3-8	$
*	*	X84	1	4	11-4-8	14	1	4	*	X84	1	4	11-4-8	*
-(hyphen)	-	X	2	C	11	20	2	C	-	X	2	C	11	-
/	/	C01	2	1	0-1	21	2	1	/	C01	2	1	0-1	/
,	,	C0821	2	C21	0-3-8	23	2	C21	,	C0821	2	C21	0-3-8	,
(	(	084	2	4	0-4-8	24	2	4	(	084	2	4	0-4-8	(
(special)						26	2	42		C0842	2	C42		
=	=	821	C21	C21	3-8	33	C21	C21	=	821	C21	C21	3-8	=
@	@	C84	C21	4	4-8	34	C21	4	@	C84	C21	4	4-8	@
A	A	X01	4	1	12-1	41	4	1	A	X01	4	1	12-1	A
B	B	X02	4	2	12-2	42	4	2	B	X02	4	2	12-2	B
C	C	CX021	4	C21	12-3	43	4	C21	C	CX021	4	C21	12-3	C
D	D	X04	4	4	12-4	44	4	4	D	X04	4	4	12-4	D
E	E	CX041	4	C41	12-5	45	4	C41	E	CX041	4	C41	12-5	E
F	F	CX042	4	C42	12-6	46	4	C42	F	CX042	4	C42	12-6	F
G	G	X0421	4	421	12-7	47	4	421	G	X0421	4	421	12-7	G
H	H	X08	4	8	12-8	48	4	8	H	X08	4	8	12-8	H
I	I	CX081	4	C81	12-9	49	4	C81	I	CX081	4	C81	12-9	I
0	(none)	(none)	C41	C	11-0	50	C41	C	-	X	C41	C	11-0	-
J/-1	J	CX1	C41	1	11-1	51	C41	1	J	CX1	C41	1	11-1	J
K/-2	K	CX2	C41	2	11-2	52	C41	2	K	CX2	C41	2	11-2	K
L/-3	L	X21	C41	C21	11-3	53	C41	C21	L	X21	C41	C21	11-3	L
M/-4	M	CX4	C41	4	11-4	54	C41	4	M	CX4	C41	4	11-4	M
N/-5	N	X41	C41	C41	11-5	55	C41	C41	N	X41	C41	C41	11-5	N
O/-6	O	X42	C41	C42	11-6	56	C41	C42	O	X42	C41	C42	11-6	O
P/-7	P	CX421	C41	421	11-7	57	C41	421	P	CX421	C41	421	11-7	P
Q/-8	Q	CX8	C41	8	11-8	58	C41	8	Q	CX8	C41	8	11-8	Q
R/-9	R	X81	C41	C81	11-9	59	C41	C81	R	X81	C41	C81	11-9	R
S	S	C02	C42	2	0-2	62	C42	2	S	C02	C42	2	0-2	S
T	T	021	C42	C21	0-3	63	C42	C21	T	021	C42	C21	0-3	T
U	U	C04	C42	4	0-4	64	C42	4	U	C04	C42	4	0-4	U
V	V	041	C42	C41	0-5	65	C42	C41	V	041	C42	C41	0-5	V
W	W	042	C42	C42	0-6	66	C42	C42	W	042	C42	C42	0-6	W
X	X	C0421	C42	421	0-7	67	C42	421	X	C0421	C42	421	0-7	X
Y	Y	C08	C42	8	0-8	68	C42	8	Y	C08	C42	8	0-8	Y
Z	Z	081	C42	C81	0-9	69	C42	C81	Z	081	C42	C81	0-9	Z
0	0	0	421	C	0 or 12-0	70	421	C	0	0	421	C	0	0
1	1	1	421	1	1	71	421	1	1	1	421	1	1	1
2	2	2	421	2	2	72	421	2	2	2	421	2	2	2
3	3	C21	421	C21	3	73	421	C21	3	C21	421	C21	3	3
4	4	4	421	4	4	74	421	4	4	4	421	4	4	4
5	5	C41	421	C41	5	75	421	C41	5	C41	421	C41	5	5
6	6	C42	421	C42	6	76	421	C42	6	C42	421	C42	6	6
7	7	421	421	421	7	77	421	421	7	421	421	421	7	7
8	8	8	421	8	8	78	421	8	8	8	421	8	8	8
9	9	C81	421	C81	9	79	421	C81	9	C81	421	C81	9	9
‡	‡	082	082		0-2-8		C	C82	‡	082	082		0-2-8	(none)
‡̄	(none)	X82	CX082		11-2-8		C41	C82	‡̄	X82	CX082		11-2-8	(none)
‡	‡	08421	08421		0-7-8		C	C8421	‡	08421	08421		0-7-8	(none)
‡̄	(none)	X8421	CX08421		12-7-8		C41	C8421	‡̄	X8421	CX08421		12-7-8	(none)

(2)

(1) Disk storage operations can be in numeric mode only.
Two-digit character representations is shown for convience.

(2) Dump Numerically instruction only. For Write Alphamerically and Write Numerically instructions an EOL character is punched in paper tape and no output is provided on the typewriter.

Character Coding

NUMERIC MODE

Character	Input				Numeric Code	Core Storage		Output				
	Typewriter	Paper Tape	Disk Storage	Card		Zone	Numeric	Typewriter	Paper Tape	Disk Storage	Card	Printer
Blank	Space			Blank			C	0	0	C82	0	0
0(+)	0	0	C82	0 or 12-0			C	0	0	C82	0	0
1	1	1	1	1			1	1	1	1	1	1
2	2	2	2	2			2	2	2	2	2	2
3	3	C21	C21	3			C21	3	C21	C21	3	3
4	4	4	4	4			4	4	4	4	4	4
5	5	C41	C41	5			C41	5	C41	C41	5	5
6	6	C42	C42	6			C42	6	C42	C42	6	6
7	7	421	421	7			421	7	421	421	7	7
8	8	8	8	8			8	8	8	8	8	8
9	9	C81	C81	9			C81	9	C81	C81	9	9
0(-)	$\bar{0}$	X or CX0	X82	11-0			F	$\bar{0}$	X	X82	11-0 (3)	-
-1	$\bar{1}$	CX1	CX1	11-1			CF1	$\bar{1}$	CX1	CX1	11-1	J
-2	$\bar{2}$	CX2	CX2	11-2			CF2	$\bar{2}$	CX2	CX2	11-2	K
-3	$\bar{3}$	X21	X21	11-3			F21	$\bar{3}$	X21	X21	11-3	L
-4	$\bar{4}$	CX4	CX4	11-4			CF4	$\bar{4}$	CX4	CX4	11-4	M
-5	$\bar{5}$	X41	X41	11-5			F41	$\bar{5}$	X41	X41	11-5	N
-6	$\bar{6}$	X42	X42	11-6			F42	$\bar{6}$	X42	X42	11-6	O
-7	$\bar{7}$	CX421	CX421	11-7			CF421	$\bar{7}$	CX421	CX421	11-7	P
-8	$\bar{8}$	CX8	CX8	11-8			CF8	$\bar{8}$	CX8	CX8	11-8	Q
-9	$\bar{9}$	X81	X81	11-9			F81	$\bar{9}$	X81	X81	11-9	R
‡	‡	082	082	0-2-8			C82	‡	082	082	0-2-8	(none)
$\bar{‡}$	$\bar{‡}$	X82	CX082	11-2-8			F82	$\bar{‡}$	X82	CX082	11-2-8	(none)
⧺	⧺	08421	08421	0-7-8			C8421	⧺ (2)	08421	08421	0-7-8	(none)
$\bar{⧺}$	$\bar{⧺}$	X8421	CX08421	12-7-8			F8421	$\bar{⧺}$	X8421	CX08421	12-7-8	(none)
numeric blank	@	C84	C	4-8			C84	@	C84	C	blank	blank

(2) Dump Numerically instruction only. For Write Alphamerically and Write Numerically instructions an EOL character is punched in paper tape and no output is provided on the typewriter.

(3) Card output is (11) on Dump Numeric instruction.

Appendix D

Significance of P and Q Addresses

OP Code	Instruction	P Address	Q Address
*01	Floating Add	Location of units position of Exponent of Augend and Result	Location of units position of Exponent of Addend.
*02	Floating Subtract	Location of units position of Exponent of Minuend and Result.	Location of units position of Exponent of Subtrahend.
*03	Floating Multiply	Location of units position of Exponent of Multiplicand and Product.	Location of units position of Exponent of Multiplier.
*05	Floating Shift Left	Location of high-order position of resulting field.	Location of units positionof Field shifted.
*06	Transmit Floating	Location of units position of Exponent of resulting field.	Location of units position of Exponent of field transmitted.
*07	Branch and Transmit Floating	P—1: location of units position of field to which Q field is transmitted. P: location of next instruction executed.	Location of units position of Exponent of field transmitted.
*08	Floating Shift Right	Location of units position of field shifted.	Location of units position of resulting field.
*09	Floating Divide	Location of units position of Exponent of Dividend and Quotient.	Location of units position of Exponent of Divisor.
11	Add (I)	Location of units position of Augend and Result.	Q_{11} is units position of Addend.
21	Add	Same as Code 11.	Location of units position of Addend.
12	Subtract (I)	Location of units position of Minuend and Result.	Q_{11} is units position of Subtrahend.
22	Subtract	Same as Code 12.	Location of units position of Subtrahend.
13	Multiply (I)	Location of units position of Multiplicand.	Q_{11} is units position of Multiplier.
23	Multiply	Same as Code 13.	Location of units position of Multiplier.
14	Compare (I)	Location of units position of field compared with Q field.	Q_{11} is units position of field compared with P field.
24	Compare	Same as Code 14.	Location of units position of field compared with P field.
15	Transmit Digit (I)	Location to which digit is transmitted.	Q_{11} is digit transmitted.
25	Transmit Digit	Same as Code 15.	Location of digit transmitted.
16	Transmit Field (I)	Location to which units position of field is transmitted.	Q_{11} is units position of field transmitted.
26	Transmit Field	Sames as Code 16.	Location of units position of field transmitted.
17	Branch and Transmit (I)	Same as Code 07.	Q_{11} is units position of field transmitted.
27	Branch and Transmit	Same as Code 07.	Same as Code 26.
*18	Load Dividend (I)	Location in Product Area to which units position of Dividend is transmitted.	Q_{11} is units position of Dividend.
* 28	Load Dividend	Same as Code 18.	Location of units position of Dividend.
*19	Divide (I)	Location in Product Area of units position of Divisor for first subtraction.	Q_{11} is units position of Divisor.

* Special Feature
(I) Immediate

Significance of P and Q Addresses

OP Code	Instruction	P Address	Q Address
*29	Divide	Same as Code 19.	Location of units position of Divisor.
31	Transmit Record	Location to which high-order position of record is transmitted.	Location of high-order position of record transmitted.
32	Set Flag	Location at which flag is set.	Not used.
33	Clear Flag	Location at which flag is cleared.	Not used.
*34	Seek	Address of Disk Control Field	Q_8 and Q_9 specify disk storage (07) Q_{11} specifies disk storage function performed.
34	Control	Not used.	Q_8 and Q_9 specify I/O device. Q_{11} specifies control function performed.
35	Dump Numerically	Location of first character written.	Q_8 and Q_9 specify output device.
36	Read Numerically	Location where first character is stored or address of Disk Control Field.	Q_8 and Q_9 specify disk storage or input device. Q_{11} specifies function performed.
37	Read Alphamerically	P—1: location where zone digit of first character is stored. P: location where numerical digit of first character is stored.	Q_8 and Q_9 specify input device.
38	Write Numerically	Location of first character written or address of Disk Control Field.	Q_8 and Q_9 specify output device or disk storage. Q_{11} specifies function performed.
39	Write Alphamerically	P—1: location of zone digit of first character written. P: location of numerical digit of first character written.	Same as Code 35.
41	No OP	Not used.	Not used.
42	Branch Back	Not used.	Not used.
43	Branch on Digit	Branch: location of next instruction executed. No Branch: Not used.	Location tested for digit other than zero.
44	Branch No Flag	Same as Code 43	Location tested for flag bit.
45	Branch No Record Mark	Same as Code 43.	Location tested for Record Mark character.
46	Branch On Indicator	Same as Code 43.	Q_8 and Q_9 specify program switch or indicator tested.
47	Branch No Indicator	Same as Code 43.	Same as Code 46
48	Halt	Not used.	Not used.
49	Branch	Location of next instruction executed.	Not used.
*55	Branch No Group Mark	Not used.	Location tested for record mark.
*71	Move Flag	Location to which flag is moved.	Location of flag to be moved.
*72	Transfer Numerical Strip	Location of units position of alphameric field.	Location of units position of numerical field.
*73	Transfer Numerical Fill	Same as Code 72.	Same as Code 72.

* Special Feature

Appendix E

The IBM Card

The IBM card measures 7-3/8 inches by 3-1/4 inches and is .007 inches in thickness. The card stock is of controlled quality, manufactured according to rigorous specifications in order to provide strength and long life. This is necessary to ensure the accuracy of results, the proper operation of IBM data processing machines, and the continued usability of information long after it is recorded.

The card is divided into 80 vertical areas called "columns" or "card columns." They are numbered 1 to 80 from the left side of the card to the right. Each column is then divided horizontally into 12 punching positions. Thus the IBM card has 960 punching positions in all. The punching positions are designated, from top to bottom of the card, 12, 11 or X, 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9. The punching positions for digits 0 to 9 correspond to the numbers printed on the card. The top of the card is known as the "12 edge" and the bottom as the "9 edge." These designations are made because cards are fed through machines either "9 edge first" or "12 edge first." "Face up" means the printed side is facing up; "face down," the reverse.

Each column of the card is able to accommodate a digit, a letter, or a special character. Thus the card may contain up to 80 individual pieces of information. Digits are recorded by holes punched in the digit punching area of the card from 0 to 9. For example, the card in Figure 68 shows a 1 punched in column 63, a 9 in column 72, and a 4 in column 77.

The top three punching positions of the card (12, 11 or X, and 0) are known as the zone punching area of the card. (It should be noted that the 0 punch may be either a zone punch or a digit punch.) In order to accommodate any of the 26 letters in one column, a combination of a zone punch and a digit punch is used. The various combinations of punches which represent the alphabet are based upon a logical structure or code.

The first nine letters of the alphabet, A to I, are coded by the combination of a 12 punch and digit punches 1 to 9. Letters J through R are coded by an 11 or X punch and digit punches 1 through 9. S through Z, the last eight letters, are the combination of the 0 zone punch and digit punches 2 through 9. Figure 69 illustrates alphabetic coding. The conversion of letters to and from this coding structure is done automatically by the various machines used to record or process data and it is rarely necessary to refer to data in its coded form. The eleven special characters, which are considered alphameric data, are recorded by one, two, or three punches.

Figures 68 and 69 illustrate the two most common types of corner cuts — upper left and upper right. The corner cut is used to identify visually a card type or to ensure that all of the cards in a group are facing the same direction and are right side up. Card types may also be distinguished by the use of colored cards or by a colored stripe on cards of a similar nature.

Field Definition

The fields in a card normally consist of 1 to 80 columns of data, depending on the length of a particular type of information. However, a field in 1620 core storage must consist of at least 2 digits or positions.

NUMERICAL DATA FIELD DEFINITION

The high-order column of a field is punched with an 11 punch as well as the digit punch. Thus, the field defining column of a numeric field contains an alphameric character (J through R) which becomes a digit with a flag when read into core storage by a Read Numerically instruction. (see CHARACTER CODING, Appendix C.)

ALPHAMERIC RECORD DEFINITION

The record-defining record mark character must be stored in core storage before or after a Read Alphamerically instruction is used to read alphameric data into core storage.

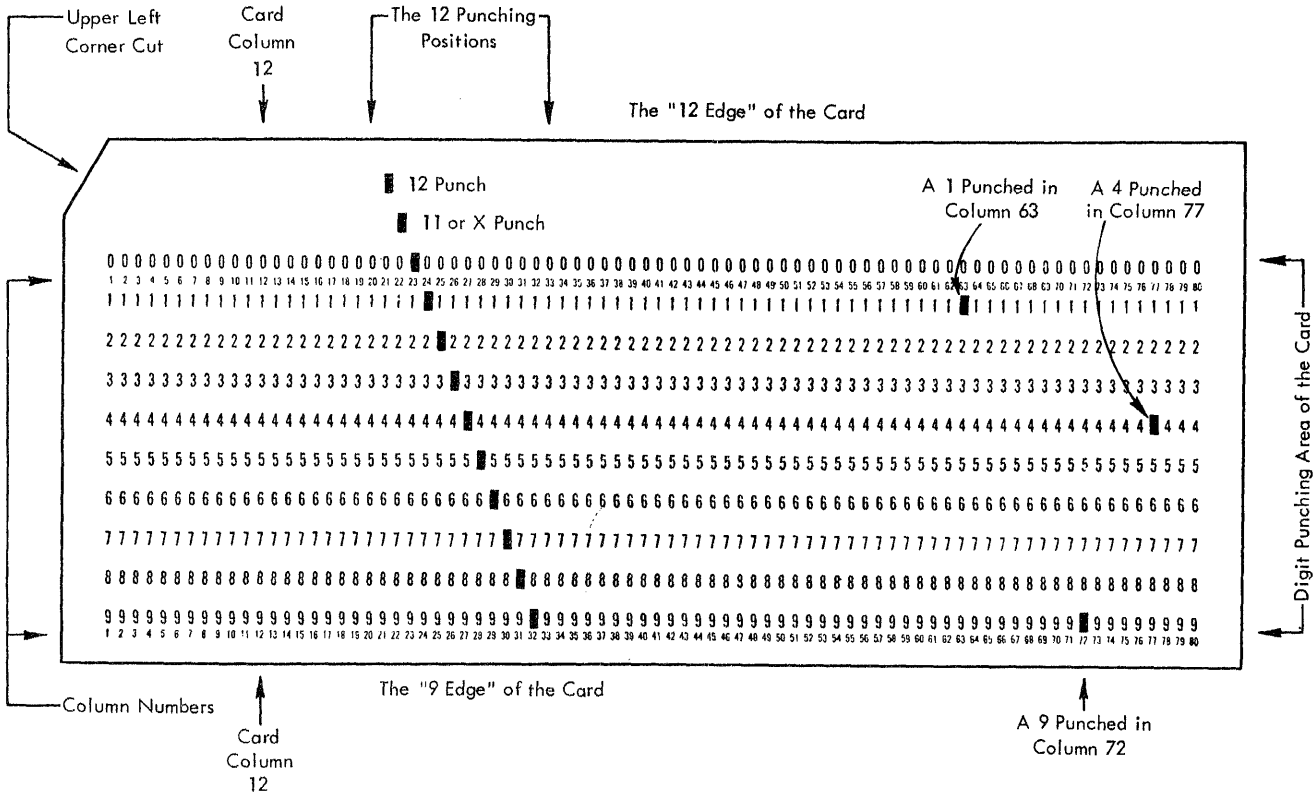


Figure 68. IBM Card

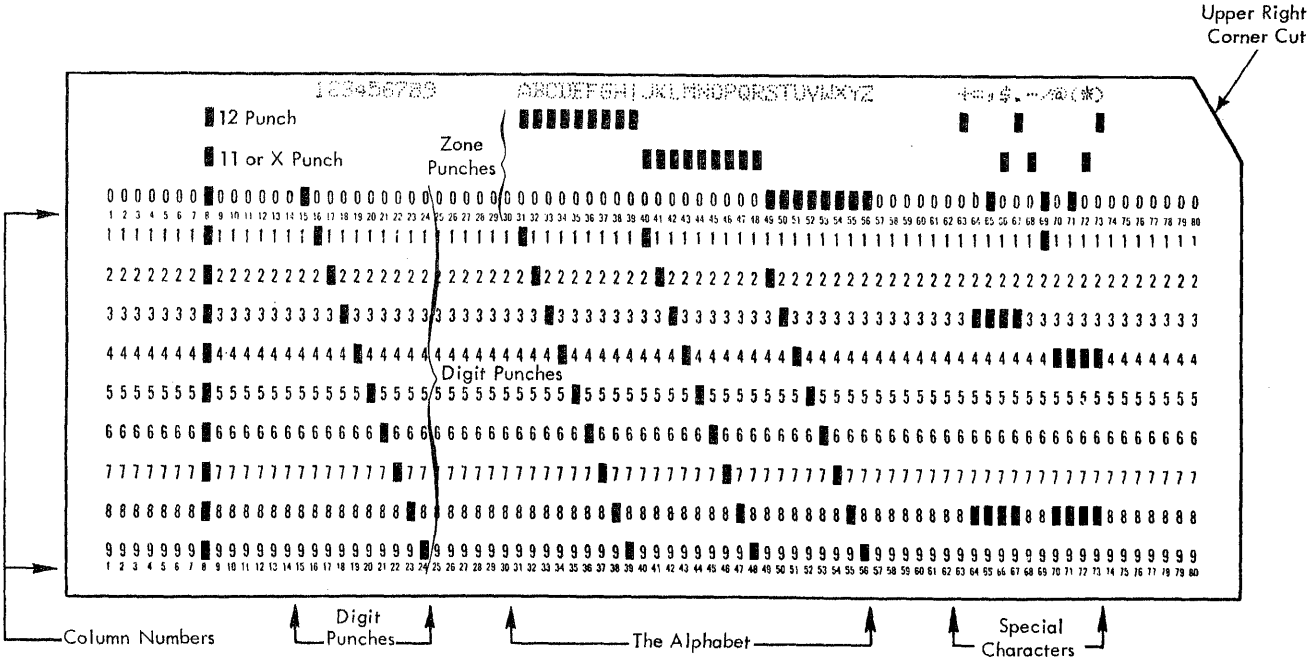


Figure 69. Card Character Coding

Appendix F - 1621 Paper Tape

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. An 8-track paper tape code is used. Seven positions, or tracks, across the width of the tape are used for coding numeric, alphabetic, and special characters. One track is used for EL (end-of-line) characters. Figure 70, representing a section of paper tape, illustrates the eight tracks and all coded characters.

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

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

The check track is used to establish correct parity. As a check that every 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 Specifications

The 1621 Paper Tape Unit is designed to operate with IBM paper tape, P/N 304469 (Figure 71). Other paper tape of equivalent paper stock may be used, but it must conform to Electronic Industries Association specifications, RS-227.

The specifications for dimensions of punched tape can be determined after conditioning the tape to the following requirements for 24 hours:

75°F ± 3.5°
50% RH ± 2%

Tape Splicing

Paper tape handling and processing will occasionally require tape splicing when paper tape needs to be altered in length, edited, or repaired. If possible, a splice should be made in nondata portions of the tape. The ability of the paper tape reader to successfully and reliably read spliced tape depends upon the quality of the splice. The following is a procedure for manually splicing two lengths of paper tape together:

1. Punch tape feed codes into the two ends of the tape to be spliced together.
2. Cut the tapes at approximately a 45° angle.
3. Holding the ends of the tape with the tape feed holes, overlap the tape end in the left hand over the tape end in the right hand approximately 1/16 inch.
4. Glue in this position with holes aligned, using a quick-setting glue such as IBM tape mucilage, P/N 221030.

Other methods of tape splicing require the use of special tape splicing equipment.

The use of tape splicing equipment should be considered if it becomes necessary to repeatedly edit tape or alter the length of tape. Special splicing equipment and materials will provide efficient, accurate, and more permanent tape splices. The selection of appropriate splicing equipment, from the many types now being offered by various manufacturers depends

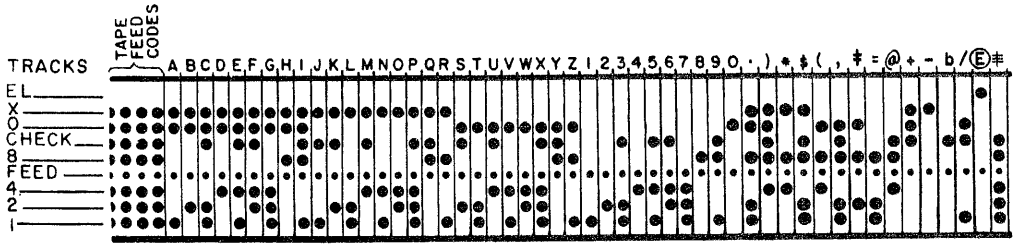


Figure 70. Paper Tape Codes

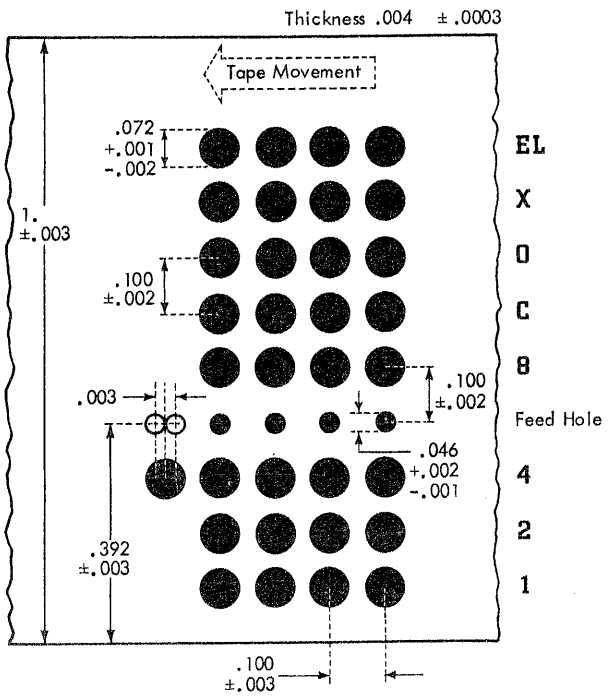


Figure 71. Paper Tape Specifications

upon the quality of the splice desired, life expectancy of the splice, time allotted to make the splice, and a price justification. The best splicing results will be obtained by first analyzing paper tape splicing needs and then purchasing the tape splicer and splicing materials best suited to these needs.

Whatever the splicing method used, the finished splice should satisfy the following conditions:

(a) Total thickness of the spliced tape area must be less than 0.010 inch.

(b) The splice must be approximately as strong as IBM paper tape.

(c) The splice width must match the width of the tape itself.

(d) The splice must be flexible.

(e) The splice must not create a gum or hindrance in the tape feed area.

Types of Tape Splices

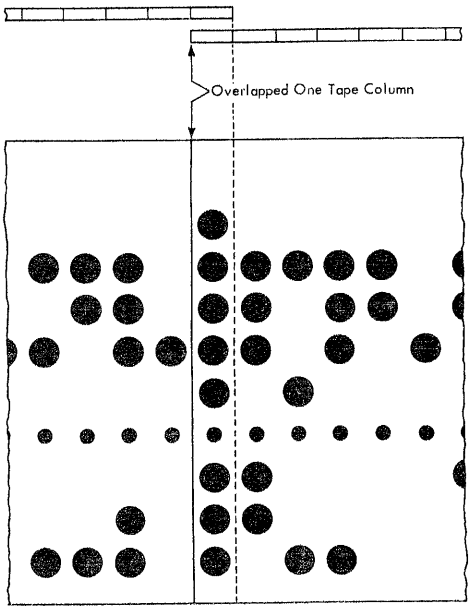
There are two types of splices: overlap splice and butt-joint splice.

Overlap Splice

The overlap splice consists of two matching paper tape ends overlapped by at least one tape column and held together with an adhesive.

With some splicing equipment, the pieces are welded together through a process of heat, pressure, and a liquid bonding agent. With this type of equipment, alignment accuracy of the tape is not required of the splice equipment operator, and the tape splicing rate is approximately one per minute.

With other overlap-type splice equipment, the tapes are glued together with a quick drying adhesive. In this process, some alignment accuracy of the tapes is required of the splice equipment operator. Tape splicing rate is about three to five minutes per splice.



ADVANTAGES OF OVERLAP SPLICING

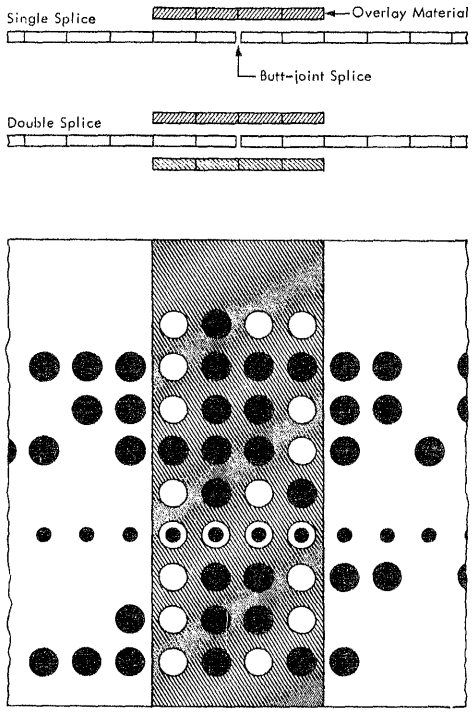
1. A large variety of overlap splicing equipment is commercially available.
2. Many splicers are available at a low investment cost.
3. Quality of splice is not usually dependent upon the operator's skill.

DISADVANTAGES OF OVERLAP SPLICING

1. An overlap-type of splice is not suitable within the tape data area because overlapped columns will result in lost data, parity errors, and/or invalid codes.
2. Low production on some splicing equipment, due to the lengthy time required for the glue to dry.
3. Short life of splice.

Butt-Joint Splice

The butt-joint splice consists of two symmetrically matched paper tape ends, butted together and held in position by a bonding agent and an overlay material. The overlay material can be plastic or paper, and can be placed on one or both sides of the tape.



With some splicing equipment, the overlay material is heat sensitive and is bonded to the tape through the use of a heated iron. Alignment accuracy of the tapes is not required of the tape splicing operator. Tape splicing rate is approximately one per minute.

With other butt-joint splicing equipment, the paper is bonded to the overlay material by an adhesive on the overlay material. Alignment accuracy of the tapes depends upon the skill of the splice equipment operator. Tape splicing rate is approximately one per minute.

ADVANTAGES OF BUTT-JOINT SPLICE

1. The splice can be made in a data-portion of tape without losing or altering data.
2. This type of splice permits tape repairing due to tears or damage.
3. A thinner splice – the total thickness of the tape and splice (with a plastic overlay) is usually thinner than two thicknesses of paper tape.

DISADVANTAGES OF BUTT-JOINT SPLICE

1. The quality and accuracy of each splice may depend upon the skill of the splice equipment operator.
2. There is a limited variety of accurate butt-joint splicing equipment.

Appendix G - Program Load Routines

Examples of a paper tape load routine and a card load routine are explained in detail in this section to illustrate the concept of program loading.

Paper Tape Load Routine

Large records, like small records, consist of data (program instructions are also considered data) and an EL punch in paper tape. The EL punch, which terminates the record, enters core storage as a record mark (≠). The format below represents a large record in core storage.

DATA. . . . . ≠

The following format represents the same record separated into four smaller records by EL punches in the paper tape.

DATA. .≠ DATA. .≠ DATA. .≠ DATA. .≠

Each EL punch causes a record mark to replace a character in core storage. Where individual records are interspersed in core storage, rather than stored in continuous line, it is sometimes necesary to restore the characters that the record marks replace. This is done by transferring each character to another location before the record mark replaces it, and then, after the record mark has been stored, to transfer the character back, wiping out the record mark. The following load routine saves these characters. Read the explanation completely for clarification of early steps.

The format of the load routine and of data records in paper tape is:

LOAD ≠ $\bar{d}$ddddd$_1$36aaaaa$_1$ ≠ DATA$_1$. . ≠ $\bar{d}$ddddd$_2$36aaaaa$_2$ ≠ DATA$_2$. . ≠. The load routine is represented by LOAD. Each $\bar{d}$ddddd36aaaaa is an addressing record for the next data record (DATA. . ≠). ddddd is the address of the character that will be replaced by the following DATA record mark, and aaaaa is the address that the following DATA record will be read into. The LOAD and addressing record marks are of no concern.

The load routine (LOAD) is read in by use of the Insert (36 00000 00100), Release, and Start keys. The load routine is as follows:

00000	41 00000 00000	No Op
00012	36 00031 00300	Read paper tape into 00031
00024	25 00071 ddddd	Transfer digit from location ddddd to 00071
00036	36 aaaaa 00300	Read paper tape into aaaaa
00048	26 00066 00035	Transfer field from 00035 to 00066
00060	15 00000 00000	Transfer Digit Immediate
00072	49 00012	Branch

After the load routine reads from paper tape into core storage locations 00000-00079 as a result of the above Insert operation; Release and Start key depressions are again required to initiate computer operation at 00012. The second Load instruction is executed, and the first addressing record, ddddd$_1$36aaaaa$_1$ ≠ , replaces part of the third and fourth Load instructions. The placement of the ≠ in Q_7 of the fourth instruction is of no concern.

ddddd is the predetermined location of the character in core storage that the next DATA record mark temporarily replaces. The character is saved (third instruction) and transferred back (sixth instruction) to replace the ≠.

aaaaa is the core storage location for each DATA record.

The third instruction, now 25 00071 $\bar{d}$ddddd, is executed. The character at ddddd is transferred to 00071 (Q_{11} of the sixth instruction).

The fourth instruction, now 36 aaaaa 00300, is executed. The next DATA record enters core storage, beginning at location aaaaa.

The fifth instruction, 26 00066 00035, is executed, and $\bar{d}$ddddd becomes the P address of the sixth instruction.

The sixth instruction, now 15 $\bar{d}$ddddd 0000X (X is the saved character) transfers the saved character back to its original location. Thus, the ≠ at the end of each data record is replaced by the character that was there to begin with.

The last instruction of the load routine, 49 00012, branches the computer to the second instruction, and reads in the addressing record for the next data record.

The last addressing and data records are used to branch the computer to the starting address of the loaded program. They are $\bar{0}$00793600074 ≠ and sssss ≠ (sssss is the starting address).

Card Load Routine

The following five-instruction load routine is also an example of Indirect Addressing (special feature). Without Indirect Addressing, the program would require 12 instructions. The program is straightforward and simple, and in addition simplifies the preparation of program cards. Other advantages are:

1. Program cards may be loaded in any sequence.
2. The address being loaded is punched in card columns 1 to 5 for easy card identification.
3. From 1 to 60 digits (as many as 5 instructions) may be loaded by each program load card.
4. After loading data from the last program cards, the program will branch to any address specified by the programmer in the last card and start the main program.
5. Data cards may be placed in the card reader with the program deck, thereby reducing card handling.
6. Cards containing corrections to the original program (patch cards) can be added to the program deck whenever necessary.
7. To simplify the explanation, 19901-19980 is used as a read-in area. It cannot be loaded by the program. The last paragraph describes a program that avoids this restriction.

Operating Instructions

To load the program, reset the 1620, place the card deck in the 1622 Card Read-Punch, and press the Load key.

Card Sequence

The cards are arranged in the following sequence:

1. The program loader card (one card, punched with the program below).
2. Program cards (see their format below).
3. The last program card, containing the branch address.
4. Data cards, if any.

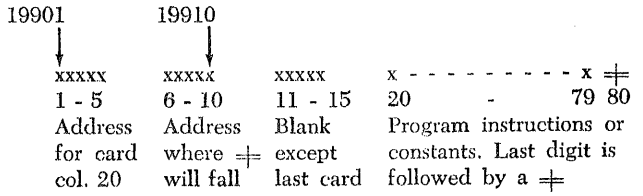
The Program Loader Card

This card is punched in card columns 1 to 56 with the following information:

36 19901 00500 25 00080 19910 31 19905 19920 25 19910 00080 49 19915 ≠. It is loaded to locations 00000 to 00079 by the Load key. Note that each of the last four instructions contains an indirect address, referring to a field in the input area. Refer to the PROGRAM CARD FORMAT for an explanation of each instruction. The program begins at 00000, and continues:

Location	Mnemonic Op Code	Instruction	Explanation
00000	RNCD	36 19901 00500	Read card to input area
00012	TD	25 00080 19910	Save digit where record mark will fall
00024	TR	31 19905 19920	Store the instructions from the card
00036	TD	25 19910 00080	Put the digit back again
00048	B	49 19915 ≠	Branch to 00000 or the program

Program Card Format



CARD COLUMNS	EXPLANATION
1-5	The address to which the data from card column 20 and on, will be transferred. It is referred to indirectly in the third instruction. No flag is needed with this or the other addresses.
6-10	The address to which the record mark will be transferred by the third instruction. This address is referred to indirectly to save and later to restore the digit that will be erased by the record mark. If the digit at that address will be loaded by a card that follows, and if proper program card sequence can always be maintained, the address may be left blank.
11-15	Blank in all but the last program load card to cause a branch to 00000. In the last program load card, these columns contain the address at which the main program is to start.
16-19	Not used. Card sequence number or program number may be punched here.
20-80	The data to be loaded, followed by a record mark. The data may be any numerical data, flagged or not. It may not contain a record mark. The first character is in card column 20. If one instruction is to be loaded, the record mark should be in column 32.

Load Routine Analysis

The first instruction, 36 19901 00500, reads the data from each card into the load area (19901-19980 in this case). A record mark is placed in 19980.

The second instruction, 25 00080 1991$\bar{0}$, transfers the digit that the record mark in 19980 will replace during the third instruction to 00080. Note the indirect Q address (1991$\bar{0}$) – the field at 19910 was loaded by columns 6-10 of the card. For example, if the data in 19901-19980 is to be transferred to 10000-10079 by the third instruction, the digit at 10079 is transferred to 00080 by the second instruction. The fourth instruction returns the saved digit to 10079.

The third instruction, 31 1990$\bar{5}$ 19920, transmits the five instructions and the record mark from 19920-19980 to the address specified by the data at 19905 (1990$\bar{5}$ is an indirect address). This address was punched in columns 1-5 of the card.

The fourth instruction, 25 1991$\bar{0}$ 00080, transfers the saved digit back to its original position, and wipes out the record mark.

The fifth instruction, 49 1991$\bar{5}$, branches the computer back to 00000, and the next card is read. The P address is indirect (1991$\bar{5}$) – the field at 19915 was loaded by card columns 11-15. These columns are blank for all but the last card. Blank card columns are read into core storage as zeros.

Program Card Examples

1-5	6-10	11-15	16-19	20-79	80
00600	00660	Blank	1	Five instructions	≠
00660	00720	Blank	2	Five instructions	≠
00100	00160	Blank	3	Data for locations 100-159	≠
00340	00401	Blank	4	Data for locations 340-399	≠

Cards 1 and 2 are standard 5-per-card load cards. Card 3 loads 60 digits of the Multiply table. Card 4 loads the last 60 digits of the Addition table and also loads a record mark to 00400.

Loading Record Marks

1-5	6-10	11-15	16-19	20-79	80
00800	00860	Blank	5	Five instructions	≠
00811	00812	Blank	6	≠ followed by blanks	Blank

Cards 5 and 6 show a method of loading a record mark between instructions. If a record mark is to be loaded to location 00811, the record mark in column 31 of card 5 must be omitted because it would terminate the third load instruction too soon. The record mark is loaded by Card 6. The instructions from Card 5 are loaded into 00800-00859.

Patch Cards

1-5	6-10	11-15	16-19	20-31	32	33-80
00600	00612	Blank	7	A single instruction	≠	Blank
00700	00712	Blank	8	Instruction with ≠ in Q_{11}	Blank	Blank

Cards 7 and 8 load single instructions. If there were patches to the program in cards 1 and 2, they would have to follow cards 1 and 2 in the program deck, though not necessarily directly behind cards 1 and 2.

Last Program Load Card

1-5	6-10	11-15	16-19	20-79	80
19000	19060	00500	9	Five instructions	≠

In this example, the loader stores the instructions in card 9 to locations 19000-19059 and then branches to 00500 to start the program. A card like this is the last card of the program deck. It should follow all patch cards.

Program Initialization

To load and execute an initialization program (defining fields, establishing constants, etc.) before loading the rest of the program, punch the starting address of the initialization program in columns 11-15 of its last program load card. Do not alter the loader (locations 00000-00060) while initializing. After initialization, branch to 00000 and loading will resume. This allows the initialization program deck and the main program deck to be loaded as a single deck by a common program loader.

Alternative Read-in Area

To use locations 00081-00160 as a program read-in area, the program in the loader should be changed to:

36 00081 00500 25 00060 0009$\bar{0}$ 31 0008$\bar{5}$
00100 25 0009$\bar{0}$ 00060 49 0009$\bar{5}$

In the last program load card, columns 20-79 must contain the multiply table digits for locations 00100-00159, followed by a record mark, and columns 1-15 must contain 00100 00160 xxxxx, where xxxxx is the address at which the program is to start.

Index

IBM 1620/1710 Printed in U.S.A. A26-5706-3

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]