File No. 1620/1710-01
Form No. A26-5781-2

IBM Systems Reference Library

IBM 1620 Central Processing Unit, Model 2

This manual contains basic programming and operating information for the 1620 Model 2 Central Processing Unit as it is used in the 1620 Data Processing System and in the 1710 Control System. Comprehensive information is included on core storage, computer instructions, data flow, and console operation.

A brief introduction to the operation of 1620 Input/Output Units is provided including a description of I/O instructions. Information on 1620-2 Special Features is also included. Timing considerations for each unit of the system and throughput times for the total system are described. Additional sections of the manual provide program load routines, an instruction summary chart, and a character coding chart.

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

Form No.	Pages	Dated
N26-0117	47, 48, 55, 56, 57, 58	4/1/65
N26-0129	37, 38, 109, 110	7/19/65
N26-0140	13, 14, 21, 22, 41, 42, 43, 44, 49, 50, 69, 70, 75, 76, 95, 96	11/8/65

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 1620 Data Processing System

1620 Data Processing System, Model 2

The IBM 1620 Model 2 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.

Seven units are available with the IBM 1620 Model 2 Data Processing System.

Central Processing Unit

The IBM 1620 Model 2 Central Processing Unit contains the computer circuitry, an operator's panel, and an input/output (I/O) typewriter. The operator's panel of the 1620 contains control keys, switches, an indicator panel, and a typewriter. The control keys and switches are used for manual or automatic operation of the system. The indicators provide visual indication of the status of various registers, indicators, and I/O units. The typewriter allows operator entry of data and instructions into core storage; it also provides a permanent log of the operator's intervention during the execution of a program. The typewriter prints at a maximum speed of 15.5 characters a second. The operator's typing speed determines the rate of entry of information through the typewriter.

MODEL 1 SYSTEM

The 1620 Model 2 Data Processing System is completely compatible with the Model 1 system. In almost all cases a program that runs on the Model 1 will also run on the Model 2. Programming changes will be required only where unusual program conditions occur, such as where the program has used the timing of instructions (for example in a program loop) to synchronize an I/O function, or where add tables, which are not present in storage in the 1620-2, have been modified to change the function of the add instruction.

This publication describes the operation of the Model 2 only. The features of the Model 1 Central Processing Unit (CPU) are described in the publication *IBM 1620 Central Processing Unit, Model 1* (Form A26-5706).

Paper Tape Reader and Punch

Paper tape input/output operations are permitted by the IBM 1621 Paper Tape Unit, which also includes the paper tape controls and the Tape Punch. The 1621 reads 8-track paper tape at the rate of 150 characters a second (cps). The Tape Punch operates at a rate of 15 cps.

Card Read-Punch

The IBM 1622 Card Read-Punch is available in two models for card input/output operations. The 1622 Model 1 reads 80-column cards at the rate of 250 cards per minute (cpm) and punches cards at the rate of 125 cpm. The 1622 Model 2 reads at the rate of 500 cpm and punches at the rate of 250 cpm. However, throughput for the system is not dependent entirely upon reading and punching speeds because both models of the 1622 contain, as a basic feature, read and punch buffers that permit overlapping of reading and punching operations with other input/output operations and with processing operations.

Printer

The IBM 1443 Printer, Models 1 and 2, are available for large-volume output printing operations. The printing speeds for each model depends upon the particular character set used with the system. The Model 1 prints at the rates of 150, 190, or 430 lines a minute. The Model 2 prints at the rates of 240, 300, or 600 lines per minute. Both models of the 1443 contain, as a basic feature, a print buffer that permits overlapping of printing operations with other input/output operations and with processing operations. Both models can print either 120 or 144 characters on one line.

Disk Storage

The 1311 Disk Storage Drive provides virtually unlimited random and sequential access storage. A disk pack containing 2,000,000 digits of information can be removed from the 1311 and another pack put in its place in one to two minutes.

The ease of mobility of a disk pack (the weight of the pack is less than ten pounds) and the simplicity of its removal from the drive means that 2,000,000 digits of data can be placed in the system within seconds. Up to four disk drives, each equipped with one disk pack, can be placed "on-line" to make possible the availability of a total of 8,000,000 numerical characters at one time (equivalent to 100,000 80-column IBM cards).

Disk storage timing depends upon the *access time* required to position the read/write heads over the appropriate disk storage record, and upon the *rotational time* required for the record to be positioned under a read/write head.

The average access time which depends upon where the data is located in disk storage, may range from 132 ms to 250 ms. However, the 1620 Data Processing System has, as a basic feature, the ability to overlap seek, input/output and processing operations. The average rotational time is 22 ms.

Core Storage

Core storage for the 1620 Model 2 is contained in the 1625 Core Storage Unit. Three models of the 1625 are available. The 1625-1, which is a basic unit of the system, contains 20,000 positions of core storage; the 1625-2 contains 40,000 positions; the 1625-3 contains 60,000 positions of core storage. Except where otherwise specifically noted, this manual is concerned with the basic 20,000 position system.

Plotter

The IBM 1627 Plotter enables the 1620 Data Processing System to illustrate digital data in any desired physical form. Two models of the 1627 are available. The Model 1 has a plotting area of 11 inches by 120 feet and operates at a rate of 18,000 steps/minute. The Model 2 has a plotting area of 29½ inches by 120 feet and operates at a rate of 12,000 steps/minute. In both models the pen is moved in increments of 1/100 of an inch.

Special Features

Three significant special features are available for the 1620 Model 2; Automatic Floating-Point Operations, Index Registers, and Binary Capabilities.

AUTOMATIC FLOATING-POINT OPERATIONS

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

The use of automatic floating-point operations can result in up to 100 percent increase in the computing power of the system depending on the amount of floating-point computations required. Also, up to 15 percent of the basic 1620 core-storage capacity can be saved through the elimination of subroutines and call sequence instructions associated with programmed floating-point operations.

INDEX REGISTERS

Index registers offer savings in program steps, core storage, and computer processing time. In addition, the programming of repetitive calculations or operations is greatly simplified.

An index register allows modification of the P and Q addresses of 1620 program instructions without actually changing the addresses in the instructions themselves; the address modifications take place in address registers. The contents of the index register are algebraically added to the specified addresses during execution of the instruction.

Fourteen index registers are available with this feature.

BINARY CAPABILITIES

The Binary Capabilities feature enables the 1620 to process binary data for such applications as process control, missile and satellite tracking, and weather observation. Binary data is collected in octal form; it can then be converted to decimal form for computation, and then processed for binary, octal, or decimal output.

Computer Characteristics

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. The 1620 Computer processes numeric, alphabetic, and special characters.

The arithmetic and logic section of the computer is directed by the stored program. The computer uses a two-address instruction format. Each 12-digit instruction includes a 2-digit operation code and two 5-digit addresses. Use of the two-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 by branch instructions at any point in the program. Branch instructions provide logical decisions through tests performed on a system of indicators and switches that can be set by the computer or by the operator.

Addition, subtraction, and division operations are accomplished by computer circuitry. Multiplication operations are accomplished by a table look-up method, in which a table located in a specified area of core storage is referred to automatically when multiply operations are being performed.

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.

Reference Publications

Machine reference information for the 1620 Model 2 Data Processing System is contained in the following publications.

IBM 1620 Central Processing Unit, Model 2 (this manual)
IBM 1621 Paper Tape Unit (A26-5836)
IBM 1622 Card Read-Punch (A26-5835)
IBM 1443 Printer for 1620/1710 Systems (A26-5730)
IBM 1627 Plotter (A26-5710)
IBM 1311 Disk Storage Drive, Model 3 (A26-5650)

The 1620 Central Processing Unit Model 2 contains the arithmetic and logic sections of the system.

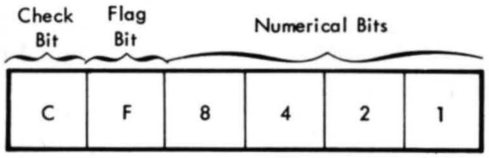
Figure 1. IBM 1620 Central Processing Unit, Model 2

Data Presentation

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

Digits

BCD Bit Array. Each core storage position is addressable and can store one digit of information in Binary-Coded-Decimal (BCD) form (C, F, 8, 4, 2, and 1). The bit positions of each digit consist of four numeric 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 by bits in the C, 4, and 2 positions, the decimal 7 by bits in the 4, 2, and 1 positions, 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 (rightmost) digit contains a flag bit, and positive if the units digit does not contain a flag bit. For example, on the typewriter, 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 digit of a numerical data field. A field is shown as XXXX where the dash over the leftmost digit is the field mark.
3. Minus Zero
   The F bit alone represents a minus zero ($\bar{0}$).
4. A flag bit over the units position of an instruction address (P or Q) indicates an indirect address.
5. Index Register Addressing (special feature)

Record Mark (≠). The record mark is a nondecimal machine digit coded C-8-2. It is used primarily in I/O

	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 through 9

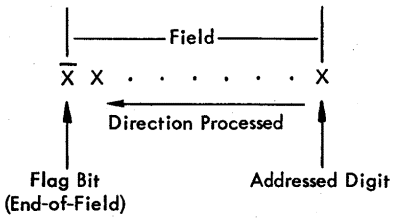
operations and in record transmission within the 1620; it cannot be used as a significant digit in an arithmetic or compare operation.

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.

Numeric Blank. The numeric 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 descriptions of the I/O instructions, Read Numerically and Write Numerically, contain more information regarding the use of the numeric blank.

Field

A field is composed of related digits that are treated as a unit of information. The digits of a field are consecutively addressed. A field is addressed by its rightmost 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 field permitted consists of two digits: the addressed digit which may or may not contain a flag (negative or positive) and the leftmost digit containing the flag bit that denotes end-of-field.

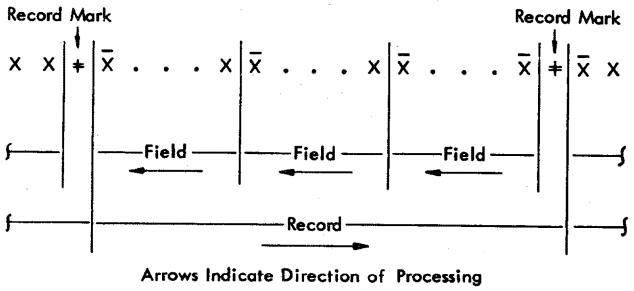


Record

A record consists of a field or fields of related data normally grouped for I/O operations and internal data transmission. A record is addressed at the leftmost 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.

Paper tape and typewriter output and internal record transmission are terminated when either a group mark or a record mark is sensed. Card output is terminated only after 80 columns are transferred to 1622 buffer storage. Disk storage records are terminated by a group mark. Printer records and Plotter records are terminated by either a record mark or a group mark.

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 pressed to place a record mark in core storage. When input is from punched cards, a record mark is placed in core storage only when 0, 8, 2 are punched in a card column. The effects of group marks in disk storage operations are described in more detail later in this manual.



Magnetic Core Storage

The 1625 is a basic unit of the system. The 1625 Model 1 contains 20,000 addressable positions of core storage. Two additional units are available. The 1625-2 provides a core storage capacity for the system of 40,000 positions and the 1625-3 provides a core storage capacity for the system of 60,000 positions, the maximum available.

Data and instructions in core storage are not affected by manually turning power on or off if care is taken to ensure that the 1620 System is in the manual mode before power is turned off.

Core Array

A core storage module of 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.

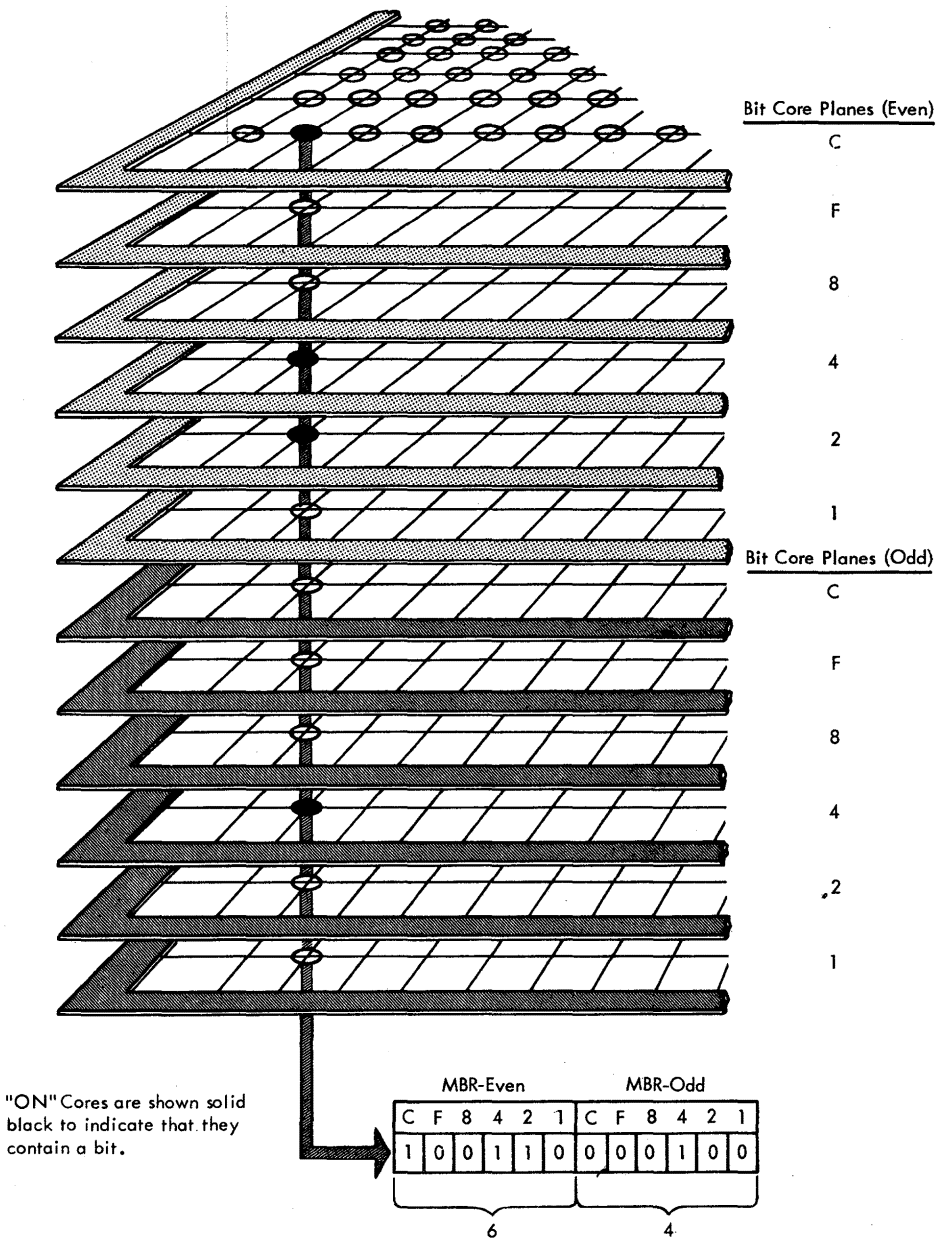


Figure 3. 1620 Core Storage Readout

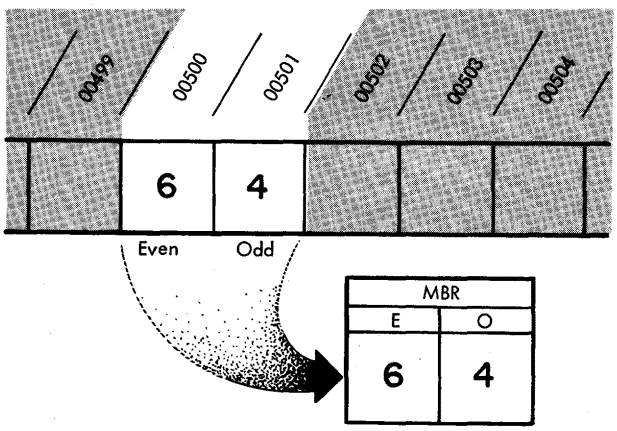
The magnetic condition of the cores at any address determines which digit is stored at that address. A magnetic core is in either the "on" condition or the "off" condition. If a bit is present, the core position is on; if a bit is not present, the core position is off.

Two-Character Transfer

During a core storage readout cycle, which takes 10 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 MBR (Memory Buffer Register) as bits.

The function of the MBR is to receive digits leaving core storage. 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 odd-address digits flow through MBR-O. Digits entering core storage are handled similarly, through the Memory Inhibit Register (MIR), under control of the units position of the associated Memory Address Register (MAR) address. Figure 3 shows 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 even-numbered address and one with an odd-numbered address. If, for example, address 00500 (even) is addressed and programmed to be read out of core storage, the digits at address 00500 and 00501 are read into MBR-E and MBR-O. If address 00501 (odd) is addressed, the operation is the same; 00500 and 00501 are still read into MBR-E and MBR-O. As shown in the following illustration, addresses are always processed into MBR in groups of two—"even" and "odd" respectively – regardless of which one is actually addressed by the program.



The selection of the specific digit to be used is determined by the *operation* to be performed.

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, that is, 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 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.

Numeric Representation

One decimal digit is required in core storage to represent a numeric character. No alphabetic or special characters except the record mark, group 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, that is, an alphabetic character, a special character, or a numeric character. A 2-digit alphameric representation of numeric characters is included 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 for 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 because 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 the alphameric characters used in the 1620. A more detailed character coding chart is contained in Appendix C.

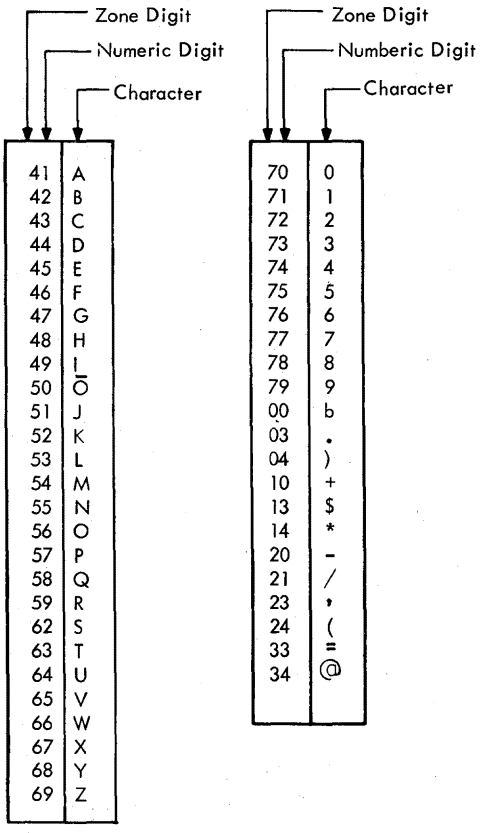


Figure 4. Alphameric Codes

Additional input/output units are available for the 1620 Data Processing System. This section of the manual provides a brief description of these units and includes operating features that are significant to the understanding of the material in this manual.

More detailed information regarding these units is contained in the following publications:

IBM 1620 Paper Tape Unit (A26-5836)
IBM 1620 Card Read-Punch (A26-5835)
IBM 1443 Printer for 1620/1710 Systems (A26-5730)
IBM 1311 Disk Storage Drive (A26-5650)
IBM 1627 Plotter (A26-5710)

Paper Tape Unit

The paper-tape reader (Figure 5), reads coded alphameric characters from 8-track paper tape at the rate of 150 characters per second. The characters are photoelectrically sensed, converted from 1621 paper tape code to 1620 BCD, and placed in core storage. If a parity error is sensed, the Read Check indicator is turned on. The computer remains in automatic mode and continues to read until the end-of-record indication (a hole in the EL channel) is reached. Whether the computer stops or continues processing depends upon the setting of the I/O Check switch. The end-of-record signal causes a record mark to be placed in core storage as the rightmost digit of the input record.

Figure 5. IBM 1621 Paper Tape Unit

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 end-of-line (EL) characters. Figure 6, 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 1; a hole in track 2 represents a numeric 2; a combination of 1 and 2 punches represents a numeric 3; and so on.

The X and O 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.

Paper Tape Load Routine

A paper tape load routine is described in Appendix E, Program Load Routines.

Tape Punch

The tape punch is housed below the tape reader in the IBM 1621 (Figure 7) and punches data from core storage into paper tape at the rate of 15 characters per

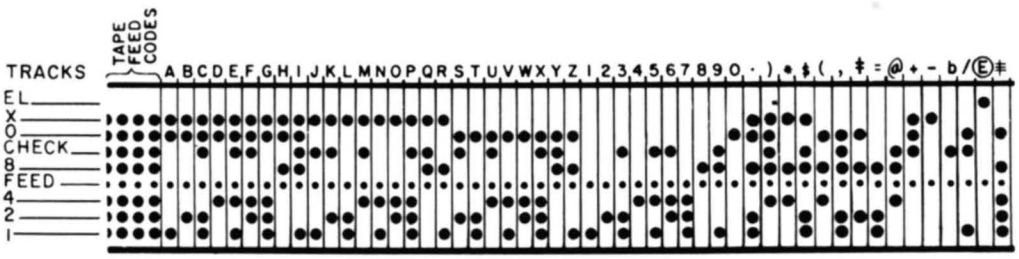


Figure 6. Paper Tape Codes

second. The characters are sent serially from core storage, starting with the location addressed by an output instruction. Each character is translated into an 8-track code before being punched.

When a record mark is sensed during the execution of a Write Numerically or Write Alphamerically instruction, an EL hole is punched and the operation stops. A Dump Numerically command causes punching to continue, regardless of record marks, until the highest-numbered core storage address of the 20,000 position module addressed by the instruction is read and punched. At this point an EL hole is punched and the operation stops. If a character with incorrect parity is transmitted from core storage and punched, or a valid character is incorrectly punched, the tape does not advance. Operation is the same regardless of the position of the I/O Check switch on the 1620 console.

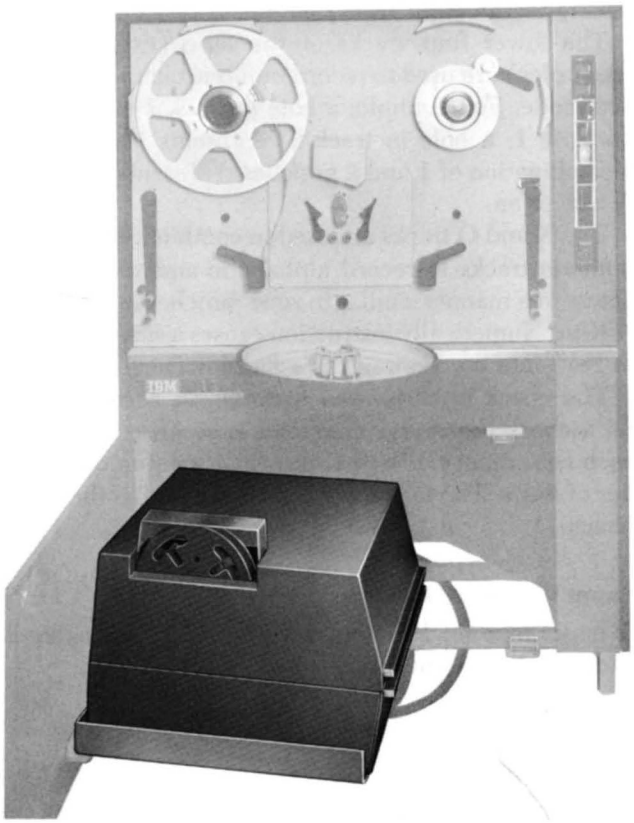


Figure 7. Tape Punch Location

Tape Punch Error Procedure. The tape feed does not advance. The computer stops; the Automatic and Manual lights and the Write Interlock and Write Check lights on the 1620 console are turned on. Program processing can be resumed with the following "restart" procedure:

1. Position the Punch Feed switch ON.
   a. The feed code is punched over the incorrect character. If the instruction is Write Binary paper tape, an EL character is also punched
   b. The Write Interlock and Write Check lights are turned off.
   c. The machine is returned to manual mode only
2. Press the Start key on the 1620 console.
   a. The original character from storage is again punched. If an incorrect character still persists the record may be corrected, if desired, before processing continues.
   b. The computer continues processing.

If the Tape Punch is used in a 1710 Control System, and the I/O Check switch is positioned to PROGRAM, the computer can branch to an error-handling routine. This routine can record the fact that an error has occurred and the computer resumes operation with the incorrectly punched character in paper tape.

If the Tape Punch runs out of paper tape, the machine stops in automatic mode and the Write Interlock light is turned on. Machine operation may be resumed by loading a new roll of tape and using the "restart" procedure just described.

When the restart procedure is used to correct the incorrect punching of valid character, and the character is again punched incorrectly, the SIE key can be used to determine the cause of the incorrect punching as follows:

1. Use the SIE key to execute one instruction at a time.
2. When the Write Check light is turned on, observe the MBR display to determine if the character is valid.

A transient condition may cause the Write Check light (but not the Write Interlock light) to come on even though a valid character has been correctly punched. Should this occur, turn on the Punch Feed switch briefly to turn off the Write Check light, then press the console Start key and proceed with the program.

Card Read-Punch

The IBM 1622 Card Read-Punch (Figure 8) provides punched card input and output for the system. The reader and punch feeds are separate and functionally independent, with individual switches, lights, checking circuits, buffer storage, and instruction codes. Under program control, up to 250 cards per minute (cpm) can be read and 125 cpm punched in the Model 1; the Model 2 reads at 500 cpm and punches at 250 cpm. Reading, punching, and processing can occur simultaneously because of individual buffer storage. Buffer storage data is transferred in 1.7 ms (Figure 9). A Last Card indicator (09) is provided for end-of-job interrogation by the programmer.

Figure 8. IBM 1622 Card Read-Punch

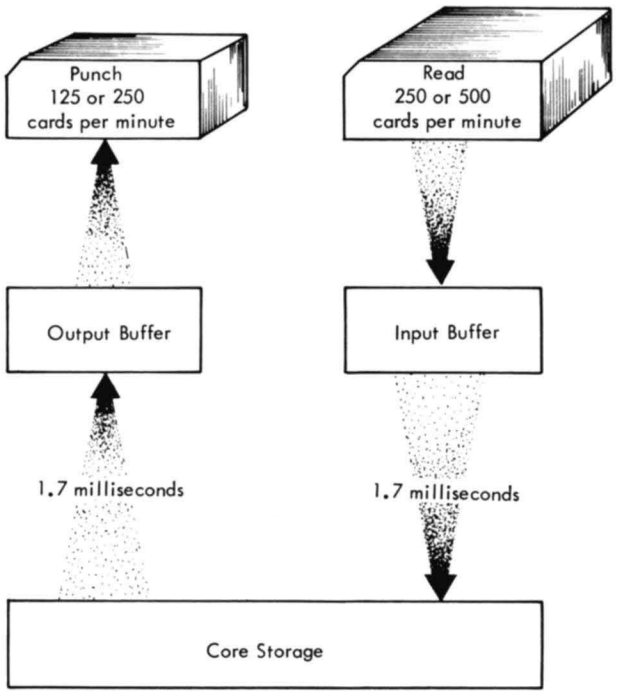


Figure 9. 1622 Buffer Storage

Data Transfer

For numeric and alphameric input/output instructions, data is transferred between 1620 core storage and 1622 buffer storage in blocks of 80 or 160 digits without consideration of record marks. A full 80 columns of card data are always transferred. Buffer storage to core storage and core storage to buffer storage transfers require 1.7 ms whether numeric or alphameric transfers are involved. When the highest-numbered core storage address of a 20,000-digit module falls within the transfer, core storage locations in the next highest 20,000 module are used or a "loop-back" to location 00000 occurs.

Card Coding

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

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

A negative record mark ($\bar{+}$), coded X-8-2, is placed in core storage as F-8-2, and punched X-8-2 in an output card.

Blank Card Columns. Because blank columns are read numerically into core storage as zeros, they can-

not be punched in the card as blanks with a Write Numerically instruction. Therefore, cards especially punched 8-4 in all columns must be read into core storage to be used when blank columns are required in output cards. The 8-4 punches are stored as C, 8, and 4 bits, and are decoded as numeric blanks when transferred to output buffer storage.

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

Double Punch Detection

Double punches are detected only if there is a duplication of BCD bits. For example, a nine (8, 1) punch and an eight (8) punch in the same column are detected as a reader check because of 8-bit duplication. However, a six (4, 2) punch and a one (1) punch in the same column are read without error as a seven (4, 2, 1) because there is no bit duplication.

BCD Coded Data

It is possible to read data recorded on IBM cards in Binary Coded Decimal form directly into the 1620 Data Processing System from the 1622 without first transforming the data to decimal codes. The BCD codes shown in Table 1, are translated without read checks or parity checks. However, if the 12 punch is used for positive indication, then the 8, 2, 12 combination results in a flagged record mark ($\overline{+}$). All other BCD card punches result in core storage values independent of a 12 punch, i.e., 4, 1, 12 gives 5.

Table 1. Valid Codes with Read Numerically

Positive Numbers		Negative Numbers	
Card Punches	Core Storage Value	Card Punches	Core Storage Value
1	1	1, 11 (or X punch)	$\overline{1}$
2	2	2, 11	$\overline{2}$
2, 1	3	1, 2, 11	$\overline{3}$
4	4	4, 11	$\overline{4}$
4, 1	5	4, 1, 11	$\overline{5}$
4, 2	6	4, 2, 11	$\overline{6}$
4, 2, 1	7	4, 2, 1, 11	$\overline{7}$
8	8	8, 11	$\overline{8}$
8, 1	9	8, 1, 11	$\overline{9}$
8, 2	+ (RM)	8, 2, 11	$\overline{+}$ (RM)

When a card is read by the 1622, the data is stored in the buffer in BCD notation. For example, if a 1 and 8 are punched in the same column of the card, they are stored in the buffer as a 1 and 8, which is the BCD coding for a 9. (Note that a 9 punch also produces a 1 and 8 code in the buffer.) Zone punches in the card are also converted to the appropriate BCD coding. Thus, an X, 1, 4 punched in a column is stored as an "N" in BCD. BCD buffer codes are shown in Table 2.

Table 2. Character Coding Chart

Character	Card	Read Buffer
Blank	Blank	C
.	12,3,8	XO821
)	12,4,8	CXO84
-	11	X
$	11,3,8	CX821
*	11,4,8	X84
,	0,3,8	CO821
(	0,4,8	O84
+	12	XOC
=	3,8	821
@	4,8	C84
A-I	12,(1-9)	XO,(1-9)
J-R	11,(1-9)	X,(1-9)
/	0,1	CO1
S-Z	0,(2-9)	O,(2-9)
0-9(+)	(0-9)	(0-9)
$\overline{1}$-$\overline{9}$(-)	11,(1-9)	X(1-9)
$\overline{0}$(-0)	11,0	CX841
0(+0)	12,0	O
‡	0,2,8	O28
‡	11,2,8	X28
Num. Blank	4,8	C,8,4
‡	0,7,8	O8421
‡	12,7,8	CXO8421

The data transferred to core storage is under control of the Read instruction, A Read Numerically instruction reads the X, 1, 4 into core storage as a flagged five (5). A Read Alphamerically instruction reads the X 1, 4 into core storage as a fifty-five (55).

Card Load Routine

A card load routine is described in Appendix E, Program Load Routines.

Printer

The IBM 1443 Printer (Figure 10) is an on-line output unit for the 1620 Data Processing System. Standard features include a tape-controlled carriage for transporting continuous paper forms, an interchangeable type bar, and buffer storage.

The easy interchangeability of the type bar enables the operator to select type style and character set for specific applications.

The printer buffer enables the computer to transfer core storage data to the buffer and then continue processing during the relatively slow printing operation. Printing speed is from 150 to 600 lines a minute, depending on the 1443 Model and the character set of the bar in use. A standard print line consists of 120 character positions. An additional 24 character positions are available as a special feature for each line.

Printing Operation

The 1443 type bar can print the standard 1620 character set, i.e., each of the 120 or 144 print positions can print any one of 48 different characters: 26 alphabetic, 10 numeric and 12 special characters:

.) + $ * — / , (= @ ≠

Horizontal spacing is 10 characters to the inch. Vertical spacing of six or eight lines to the inch is manually selected by the operator for continuous line printing. Selective spacing and skipping between printed lines are controlled by the 1620 program.

Several repetitions of the full character set are located on the type bar. There are 120 or 144 hammer magnets and print hammers, each of which has an individual print buffer position. Once the print buffer is loaded, individual characters are printed as quickly as they pass in front of the hammer that has the corresponding character stored in that buffer position.

Printing Speeds

The 1443 Model 1 prints from 150 to 430 lines a minute; the Model 2, from 240 to 600 lines a minute. The determining factor for both models is the size of the type bar character set:

Character Set	Lines Per Minute	
	1443-1	1443-2
13	430	600
39	190	300
•52	150	240

• 4 Characters not used

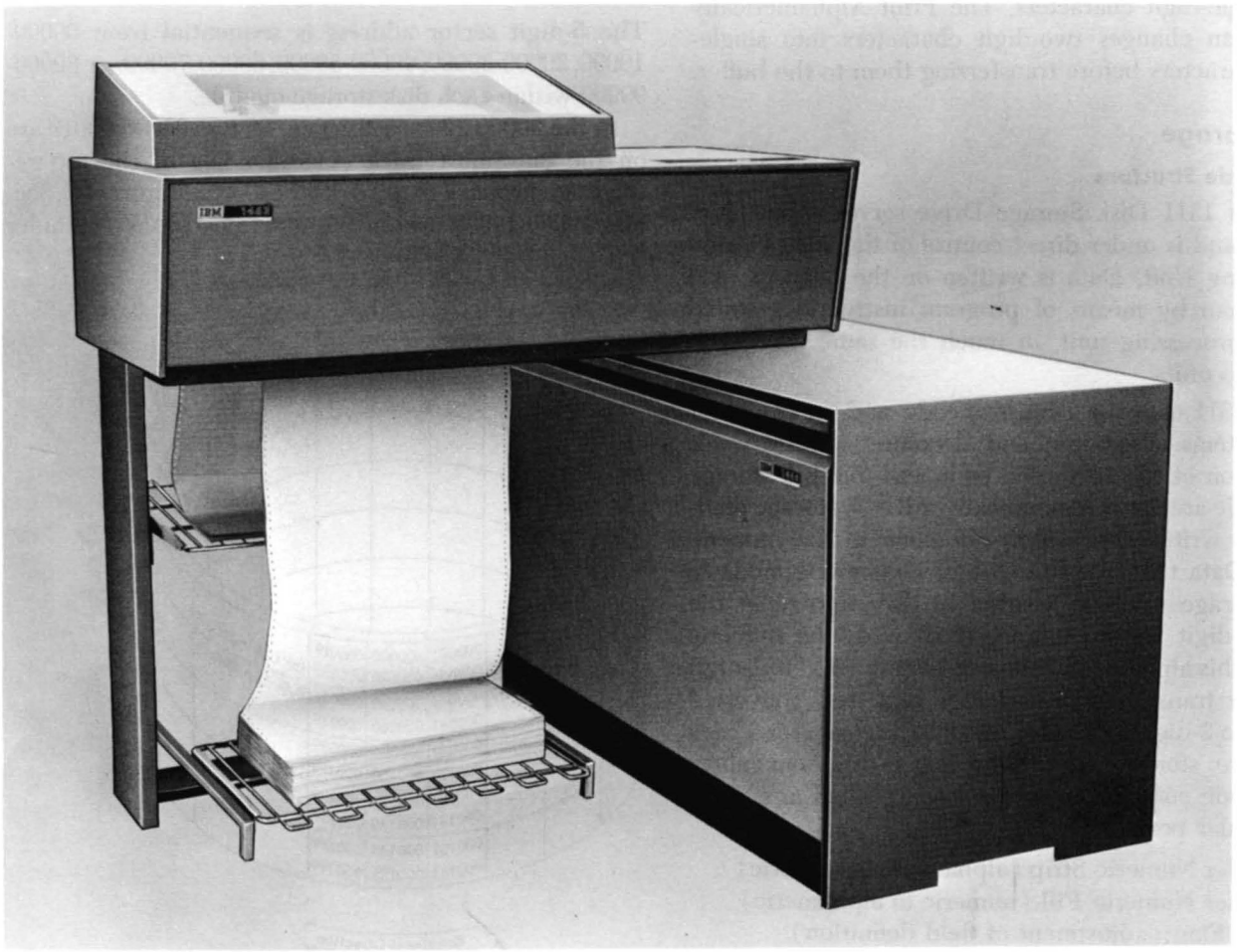


Figure 10. IBM 1443 Printer

Print Control

Data to be printed must first be edited and arranged in core storage in exactly the format to be printed. The appropriate 1620 output instruction can then be executed to transfer the data to the print buffer. The print buffer contains 197 positions even though the 1443 prints only 120 or 144 positions. The Printer Dump (PRD-35) instruction transfers a full buffer load of data, 197 characters, including group mark (≢) and record mark (≠) characters. The Print Numerically and Print Alphamerically instructions, however, transfer a full buffer load only if neither of the forenamed characters is detected. The detection of either character terminates data transfer and reduces the remaining buffer positions to blanks. Neither the terminating record mark nor group mark is transferred to the buffer.

When performing a Print Alphameric operation, a group mark or record mark should be placed in position 121 or 145 to prevent possible Printer Checks caused by the transfer of the unedited positions (121 through 197 or 145 through 197) to the print buffer.

The Print Numerically instruction encodes and transfers single-digit characters. The Print Alphamerically instruction changes two-digit characters into single-digit characters before transferring them to the buffer.

Disk Storage

Data Code Structure

The IBM 1311 Disk Storage Drive serves as auxiliary storage and is under direct control of the 1620 Central Processing Unit. Data is written on the disks or read from them by means of program instructions stored in the processing unit, in much the same way as in other I/O units.

The 1311 uses the 7-bit, BCD code as do many other IBM systems. Encoding and decoding necessary for conversion of the 1620 6-bit code and the disk storage 7-bit code are done automatically. All disk storage reading and writing operations are done in the *numeric mode*. Data that is in the 2-digit alphameric mode in core storage must be written in disk storage in the same 2-digit representation. If desired, the numeric data in this alphameric code can be converted to 1-digit form for transfer to disk storage and then converted back into 2-digit form after the data has been read back into core storage. The following instructions allow these code conversions to be accomplished as part of the regular program.

Transfer Numeric Strip (alphameric to numeric)
Transfer Numeric Fill (numeric to alphameric)
Move Flag (adjustment of field definition)

Other IBM systems record alphameric data in disk storage in a 1-character mode. If alphameric data from these systems is to be used at a later time on a 1620 System it must be translated to the 2-digit alphameric representation before being entered into the 1620 System.

One additional special character is necessary in disk storage operations; The group mark (≢). Group mark coding in various elements of the 1620 Data Processing System with identification of the same symbol in the 1400 Systems is as follows:

	≢ (Unflagged)	$\bar{≢}$ (Flagged)
1620 Core Storage	C 8 4 2 1	F 8 4 2 1
1311 Disk Storage	0 8 4 2 1	C X 0 8 4 2 1
Paper Tape I/O	0 8 4 2 1	X 8 4 2 1
Typewriter I/O	≢	$\bar{≢}$
Card I/O	0, 7, 8	12, 7, 8
1440/1401/1410	Tape Segment Mark	Group Mark

Use of the Group Mark is explained under DISK STORAGE DATA PROTECTION.

Sector Addresses

The 5-digit sector address is sequential from 00000-19999, 20000-39999, 40000-59999, 60000-79999, or 80000-99999 within each disk storage module.

In the 00000-19999 sequence, sectors 00000-00019 are on the outermost track (cylinder 00) of the top recording surface (read/write head 0), Figure 11. Sectors 19980-19999 are on the innermost track (cylinder

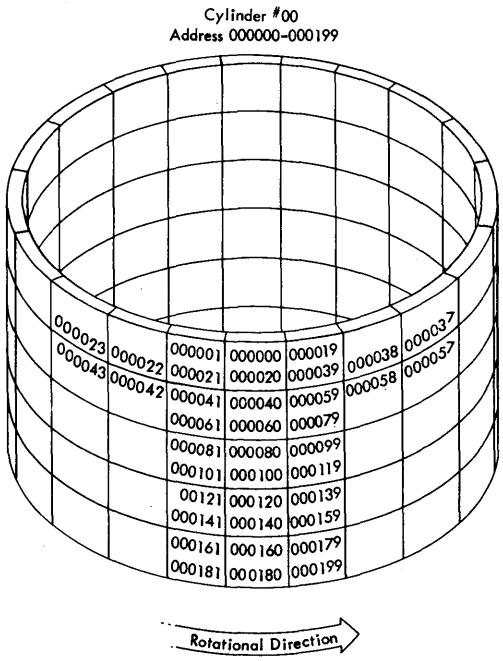


Figure 11. Addressing for Cylinder 00

99) of the bottom recording surface (read/write head 9). Using the sector address, the 1620 automatically selects the correct drive, cylinder, and read/write head.

NOTE: Six-digit sector addresses are provided in the 1311 Disk Storage Drive, although only five positions are required in the 1620/1710 Systems. During all seeking, reading, or writing operations, the internal circuitry of the 1620 Central Processing Unit automatically supplies a high-order 0 (zero) to all addresses; *no additional programming effort is necessary.* Throughout this publication, sector addresses are referred to as consisting of only five digits.

Instruction Format

The format for disk storage instructions (Figure 12) 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.

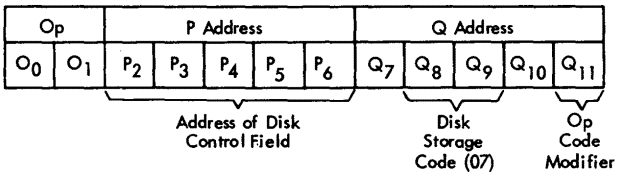


Figure 12. 1311 Instruction Format

The functions of the P and Q parts of the instruction differ from those of other input/output instructions. The P address, instead of being the core storage address of the *data* transferred to or from the I/O unit, is the core storage address of the disk control field, which is described in the next section. *The P address must always be an even number.*

The Q part of the instruction must contain 07 in Q_8-Q_9 to identify it as a disk storage operation, and must contain a code in Q_{11} 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 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 13). 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. The total length of the area reserved for data can be defined by a group mark at the end of this field.

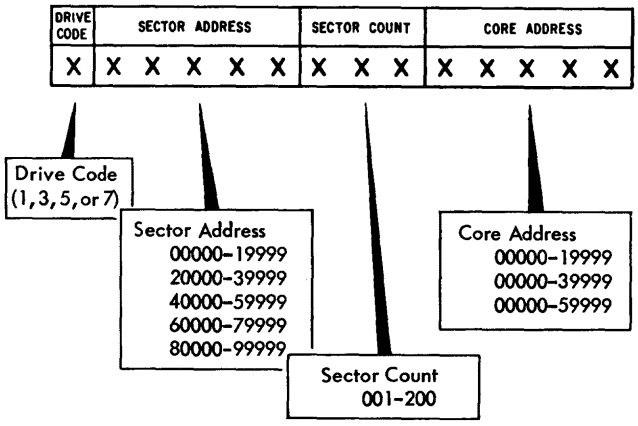


Figure 13. Disk Control Field

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.

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). For example:

ADDRESS BLOCK	SELECTS DISK DRIVE
00000-19999	0
20000-39999	1
40000-59999	2
60000-79999	3

If the disk addresses on a disk pack do not correspond to the number of the disk storage drive on which they are located, then the drive code is used.

DRIVE CODE

This coding provides the flexibility to operate upon any block of disk addresses regardless of which disk drive the pack is located on, and when the occasion requires, to have more than one disk pack containing the same disk addresses on-line at one time.

Selection of a particular disk storage drive can be made by using the following codes:

DRIVE CODE	SELECTS DISK DRIVE NUMBER
1	0
3	1
5	2
7	3

For example, if address block 20000-39999 is located on Disk Drive 3 (instead of Disk Drive 1), the drive code in the disk control field would be specified as 7.

When this flexibility is not required, the drive is selected directly from the sector address and a 0 (zero) is placed in the drive code portion of the Disk Control Field.

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	1
5		40000-59999	2
7		60000-79999	3

The effect of using an invalid address or drive code, or using an address or drive code that specifies a non-existent drive, depends upon whether only one drive or several drives are attached to the system. (An invalid address or drive code is any code not specified in the preceding chart. An example of addressing a non-existent drive would be addressing a fourth drive on a system when the system contains only three.)

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 non-existent disk drive, the computer is interlocked. The SCTR CYC light (under the I/O lights) and the Write Interlock light turn on. A seek operation to an invalid address or to a non-existent 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.

SECTOR ADDRESS

The sector address identifies the first sector of a disk operation. At the beginning of the operation, the sector address is transferred to OR-1, which is incremented by one as each sector is processed. Thus, at any time during a disk operation OR-1 contains an address one number higher than the current sector address; however, the disk control field in *core storage* still contains the address of the first sector processed. Instructions in the track mode are exceptions to the above; none of the sector addresses are compared.

SECTOR COUNT

This is the number of sectors read or written. At the beginning of the operation, the sector count is transferred to PR-2, 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-2 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.

CORE ADDRESS

This is the 1620 core 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. At the beginning of read, write, and check disk operations, the core storage address is put in OR-2 and incremented by two for each succeeding position addressed. The original address remains in the disk control field.

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) also turns 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-2. 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) turns on.

Programming Operations

The 1311 Disk Storage instructions consist of four basic operations.

Seek Disk
Read Disk
Write Disk
Check Disk

The read, write, and check disk operations can be varied to operate only upon the *data* from the disks or to utilize *both* data and the sector address in disk storage.

BASIC OPERATIONS

Seek Disk. The seek disk operation directs the read/write heads to the proper cylinder on the disk pack. It is followed by a read or write instruction.

The data on the disk records is not acted on by this instruction. The seek instruction merely positions the access arms over the proper cylinder. The P address of this instruction contains the core storage address of the disk control field.

Read Disk. The read disk operation transfers data from disk storage to core storage in the processing unit.

The area in core storage where the data is to be placed must be defined prior to the instruction. The P address of the instruction contains the core storage address of the disk control field.

Write Disk. The write disk operation transfers data from core storage to disk storage. The area in core storage from where the data is to be transferred must be defined prior to the instruction. The P address of the instruction contains the core storage address of the disk control field.

Check Disk. A Check Disk instruction should follow each Write Disk instruction. The check disk operation causes the data written in *disk* storage to be read and compared with the original source data in *core* storage. When the disk data does not compare, bit-by-bit and character-by-character, with the data in core storage from which it was written, a disk error indicator is set.

TRACK AND SECTOR MODES

Read, Write, and Check Disk instructions have two "modes" of operation: track mode and sector mode.

Track Mode. The track mode of operation transfers *both* data and disk sector addresses to and from disk storage, one complete track at a time. This mode of operation allows sector addresses recorded in disk storage to be changed. The operation requires that the disk control field contain an address of one of the sectors within the track. Sector count is set to 020 (twenty sectors are transferred).

Sector Mode. The sector mode is the normal mode of operation. Read, write, and check disk operations in the sector mode transfer data, but do not transfer disk sector addresses. The number of sectors to be handled within one operation is designated in the sector count portion of the disk control field. A sector is transferred only when the sector addresses in disk storage compare absolutely with those in the disk control field. The sector address in OR-1 is automatically increased by 1 for each sector transferred and thereby supports the comparison of successive sector addresses. Similarly, the sector count is reduced by 1 for each sector transferred, and indicates by a 000 setting that the required operation has been completed.

Plotter

The IBM 1627 Plotter (Figure 14) converts tabulated digital information into graphic form. Bar charts, flow charts, organization charts, engineering drawings, and maps are among the many graphic forms of data which can be plotted.

The 1626 Plotter Control Unit is the interconnecting unit between the 1627 and the 1620 Central Processing Unit. The 1626 contains the translating and control circuits for the 1627 and serves as a stand for the 1627.

Operation

Data from 1620 core storage is transferred serially (by digit) to the 1626 where it is translated into 1627 actuating signals. These signals are then converted into drawing movements by the 1627 Plotter.

The actual recording is produced by incremental movement of the pen on the paper surface (y-axis) and/or the paper under the pen (x-axis). The pen is mounted in a carriage that travels horizontally across the paper as viewed from the front of the plotter. The vertical plotting motion is achieved by rotation of the pin feed drum, which also acts as a platen (Figure 15).

Figure 14. IBM 1627 Plotter

The drum and the pen-carriage are bi-directional; i.e., the paper moves forward or backward, and the pen moves left or right. Control is also provided to lower or raise the pen to or from the paper surface. The pen remains in the "up" or "down" position until directed to change to the opposite status.

The drum and pen-carriage movements and the pen status are controlled by digits transferred to the 1626 Plotter Control Unit by a 1620 Output instruction. Each digit is decoded into a directional signal and relayed to the 1627 Plotter.

The plotter operation is controlled by any of the following instructions:

Dump Numerically
Write Numerically
Write Alphamerically.

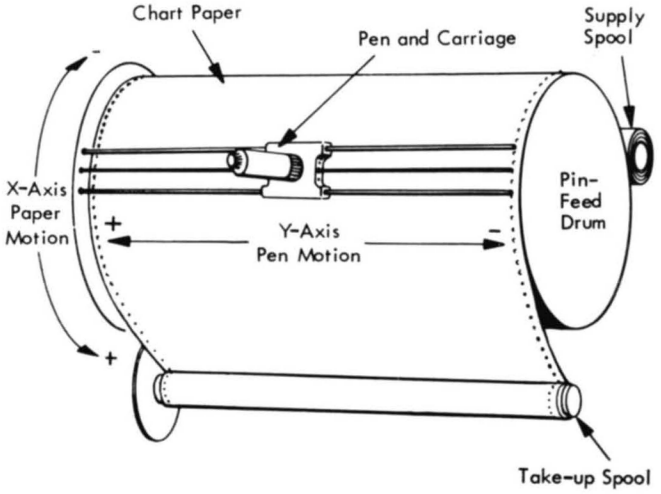


Figure 15. Paper and Pen Motions

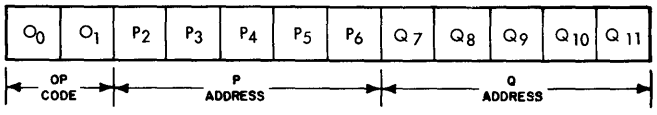
Stored Program Concept

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 the various computer operations available, 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. Normally, the program is executed in a sequential manner, that is, the computer starts with the first instruction and progresses, in sequence, 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 digit and read from right to left, instructions are addressed at O_0, the leftmost digit, and read from left to right.

Op Code

Upon initiation of an instruction, the operation 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; that is, the O_0 digit of an operation code must be stored in an even-numbered address so that the Op register can receive the correct 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), (4) the location of the alphameric field in the Transfer Numerical Strip and Transfer Numerical Fill instructions, or (5) the address of a control word used in disk storage operations.

Q Address

The Q address specifies: (1) the location from which data is transmitted, (2) the indicator being interrogated, (3) the I/O unit being used, or (4) the location of the numerical field in the Transfer Numerical Strip and Transfer Numerical Fill instructions. Also, instruction modifier digits are placed in the Q address for these instructions with the same Op code number.

Instruction Execution Time

The time required to execute an instruction is divided into two parts: the Instruction (I) cycle and the Execution (E) cycle.

I Cycle. The instruction is read out of core storage into instruction registers for decoding. Since most instructions have the same length (12 digits) I time is normally the same for all instructions; 60 μsec. There are six 10-μsec cycles in each I cycle. A microsecond – abbreviated as "μsec"– is one millionth of a second.

E Cycle. The instruction is executed as specified by its Op code and P and Q parts. The E cycle, which is the actual time required to execute the instruction after it has been read into the instruction registers, requires a number of 10-μsec cycles that varies with individual

instructions, size of data fields, speed of I/O units, etc.

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 of 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 μsec unless otherwise 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 positions of the instruction as *data* instead of using them as the address of data.

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 16). 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}, 65 would be transferred to 00543. The difference between the Transmit Field and Transmit Field Immediate instructions can best be shown by comparing Figures 16 and 17. The Transmit Field instruction, Figure 17, transfers the data at the Q Address to the P addresses.

Indirect Addressing

Indirect addressing saves program steps and computer time by providing a simplified method of address modification. Its primary use is in programs where many instructions have the same address and this address must be modified by the program. Indirect addressing may also be used for linking subroutines.

Description

Normally, the P or Q address in an instruction is the location of the data used during execution of the instruction. An indirect P or Q 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, except if the second address is another indirect address. In effect, the address at the indirect address location is a substitute 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 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 an indirect address when a flag bit is over the units position. Figure 18 shows that (1) the instruction (21 00500 00650) has an indirect P address of 00500, (2) the data at

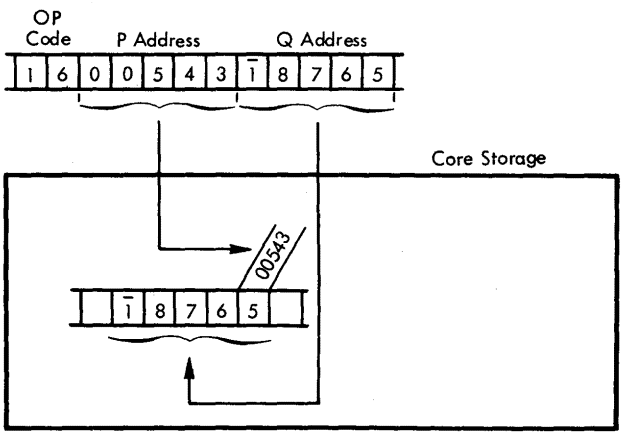


Figure 16. Transmit Field Immediate – Data Flow

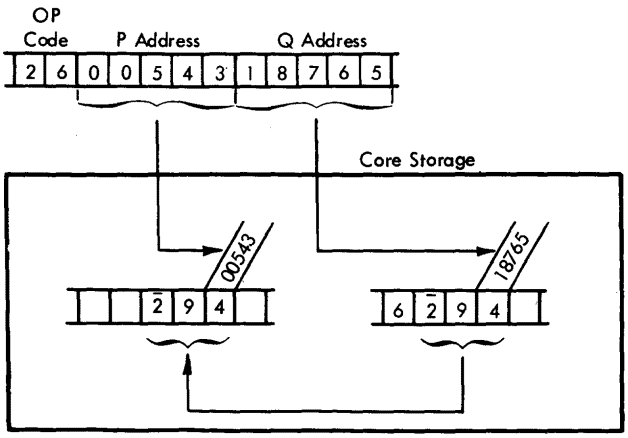


Figure 17. Transmit Field – Data Flow

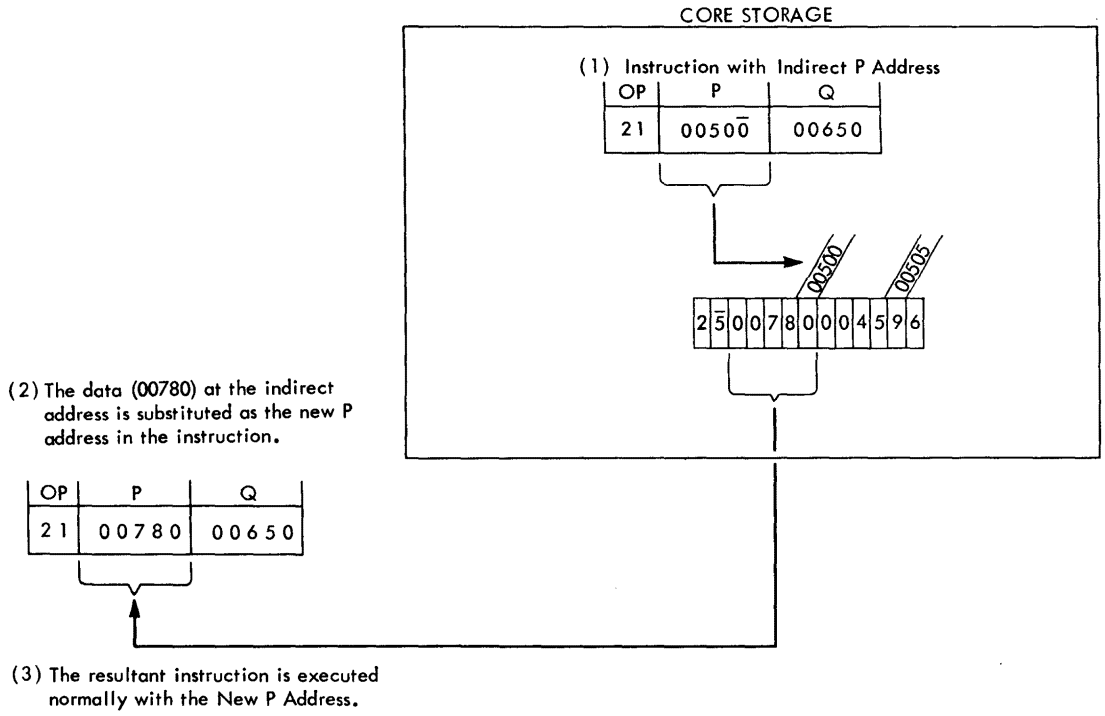


Figure 18. Indirect Addressing

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 an instruction register in 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; this 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. When the P address of an immediate instruction is an indirect address, the Q data cannot be more than six digits in length because the flag bit over the units position of the P address also defines the end of the immediate data.

The Indirect Addressing feature can be turned off when programs not using the feature are being processed. Indirect addressing becomes operative automatically when power is turned on. Indirect addressing can be turned off or turned on by the Branch and Select instruction.

Execution Time

Each address interpreted as an indirect address requires three additional 10-μsec memory cycles. For example, an instruction with two indirect addresses requires an additional 60 μsec.

EXAMPLES

The Add instruction, 21 00500 00650, is shown in Figure 19 with 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 00500 00650, is shown in Figure 20. The Q data 00650, is added to the data at the address specified by the indirect P address. The result 1155078, replaces the original P data, 1154428, at 09400.

A data flow diagram for a Branch instruction is shown in Figure 21. 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 19. Examples of Indirect Addressing

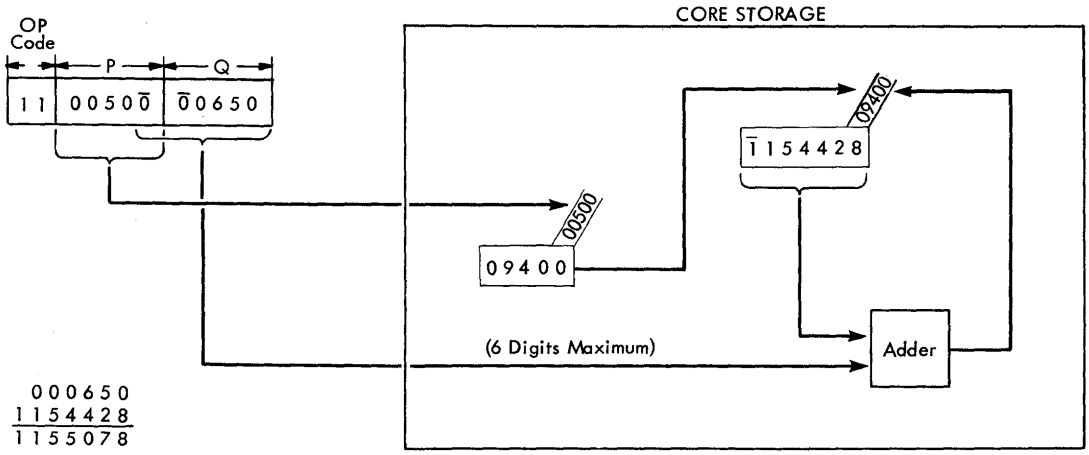


Figure 20. Indirect Addressing, Add Immediate Instruction

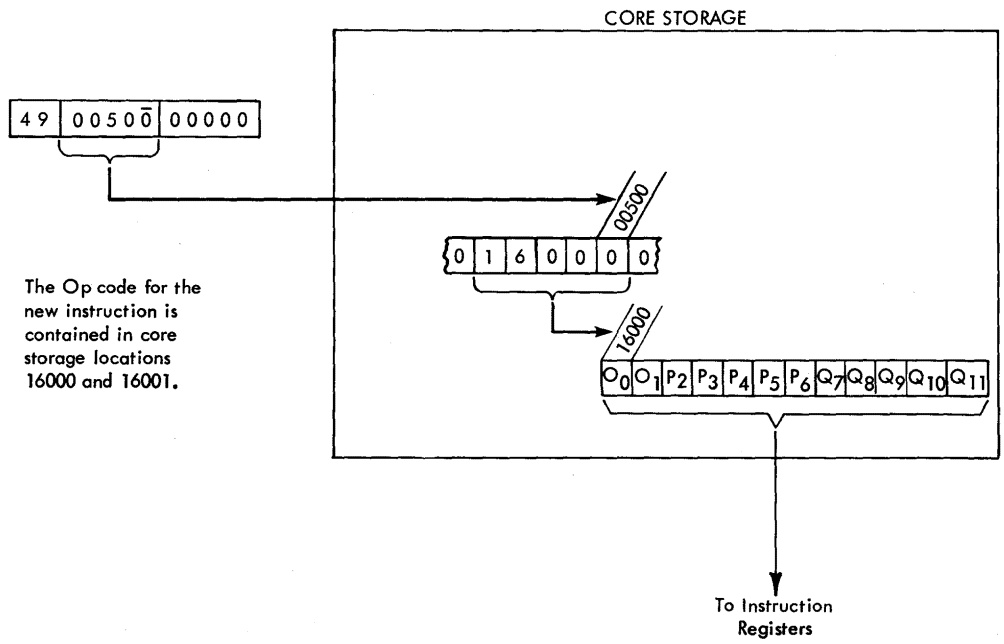


Figure 21. Indirect Addressing, Branch Instruction

Instruction Types

The description of the instructions are grouped into eight categories:

1. Arithmetic
2. Logic
3. Internal Data Transmission
4. Input/Output
5. Program Control
6. Floating Point (Special Feature)
7. Index Registers (Special Feature)
8. Binary Capabilities (Special Feature)

The special features instructions (categories 6, 7, 8) are described separately to facilitate the understanding of their operation even though the function of these instructions could be classified into categories 1 through 5. All instructions, with their associated operation codes and SPS mnemonics, are shown in Table 3 in a functional sequence. The Instruction Summary in Appendix A is listed in alphabetic sequence by instruction name.

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 instruction.

Data Flow and Field Length Definition

Data is read serially from right to left until terminated by a flag bit defining the leftmost position of the field. For example, where the data in a field is $\bar{2}85$, the dash (flag bit) over the leftmost digit indicates a field mark.

The *minimum* length of both the P and Q fields is two digits; a units digit which contains the sign, and at least one other 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 (H/P). The High/Positive indicator is turned on at the beginning of each arithmetic instruction and remains on if the result is positive and not zero. It is turned off if the result is negative or zero. The "on" or "off" status of this indicator is maintained until the next arithmetic or compare instruction is executed. It is turned off by pressing the Reset key on the console.

Equal/Zero (E/Z). The Equal/Zero indicator is turned on at the beginning of each arithmetic instruction and remains on if the result is zero. It is turned off if the result is not zero. The "on" or "off" status of this indicator is maintained until the next arithmetic or compare instruction is executed. It is turned off by pressing the Reset key on the console.

Arithmetic Check (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, and 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 depression 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 multiplication is used in the 1620. A multiply table is automatically referred to by the computer during multiply operations. The positions of core storage containing the table data are addressable but must not be altered; altering can cause incorrect operations to result.

Two hundred positions of core storage, 00100 through 00299, are assigned to the storage of the Multiply table.

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,

Table 3. 1620 Instructions

Instructions	Mnemonic	Code
Arithmetic		
Add	A	21
Add (I)	AM	11
Subtract	S	22
Subtract (I)	SM	12
Multiply	M	23
Multiply (I)	MM	13
Divide	D	29
Divide (I)	DM	19
Load Dividend	LD	28
Load Dividend (I)	LDM	18
Floating Add*	FADD	01
Floating Subtract*	FSUB	02
Floating Multiply*	FMUL	03
Floating Divide*	FDIV	09
Logic		
Compare	C	24
Compare (I)	CM	14
Branch	B	49
Branch on Digit	BD	43
Branch No Flag	BNF	44
Branch No Record Mark	BNR	45
Branch No Group Mark*	BNG	55
Branch Indicator	BI	46
Branch No Indicator	BNI	47
Branch and Transmit	BT	27
Branch and Transmit (I)	BTM	17
Branch and Transmit Floating	BTFL	07
Branch Back	BB	42
Branch and Select	BS	60
Branch and Modify Index Register*	BX	61
Branch and Modify Index Register (I)*	BXM	62
Branch Conditionally and Modify Index Register*	BCX	63
Branch Conditionally and Modify Index Register (I)*	BCXM	64
Branch and Load Index Register*	BLX	65
Branch and Load Index Register (I)*	BLXM	66
Branch and Store Index Register*	BSX	67
Branch on Bit*	BBT	91
Branch on Mask*	BMK	91

Instructions	Mnemonic	Code
Internal Data Transmission		
Transmit Digit	TD	25
Transmit Digit (I)	TDM	15
Transmit Field	TF	26
Transmit Field (I)	TFM	16
Transmit Floating	TFL	06
Transmit Record	TR	31
Transmit Record No RM	TRNM	30
Transfer Numerical Strip	TNS	72
Transfer Numerical Fill	TNF	73
Floating Shift Right*	FSR	08
Floating Shift Left*	FSL	05
OR to Field*	ORF	92
AND to Field*	ANDF	93
Complement Octal Field*	CPLF	94
Exclusive OR to Field*	EORF	95
Octal to Decimal Conversion*	OTD	96
Decimal to Octal Conversion*	DTO	97
Move Address*	MA	70
Input/Output		
Read Numerically	RN	36
Write Numerically	WN	38
Dump Numerically	DN	35
Read Alphamerically	RA	37
Write Alphamerically	WA	39
Read Binary Paper Tape*	RBPT	37
Write Binary Paper Tape*	WBPT	37
Program Control		
Seek*	SK	34
Control	K	34
Set Flag	SF	32
Clear Flag	CF	33
Move Flag	MF	71
Halt	H	48
No Operation	NOP	41

etc.). The sign analysis chart in Figure 22 shows that: (1) the Q data is complement-added during addition when the signs of the P and Q fields are different and during subtraction when the signs of the P and Q fields are alike; (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 complemented; and (3) if a recomplement occurs, the original sign of the P field is changed.

For example:

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

According to the sign control analysis chart

1. The Q data is complement-added ($\bar{1}5$ becomes 85), and $\bar{8}5 + \bar{3}5 = \bar{2}0$. 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. The Arithmetic Check indicator 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.

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 23, the sum ($\bar{1}4$) replaces the $\bar{1}2$ at 00500 (P address). The field mark remains at the high-order position. When the sum is zero, the sign of the P field is retained. For sums other than zero, the sign of the 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 High/Positive indicator is on if the sum is positive and not zero; the Equal/Zero indicator is on if the sum is zero. Neither indicator is on if the sum is negative.

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

Basic Execution Time: $T = 10\ (6.5 + .5D_Q + D_P)$
Recomplement Time: $T = 10\ D_P$

Add Immediate (AM-11)

Description. The description is the same as that for Add (A-21) except that the data in the Q part of the instruction is used as the Q data. For example, if the Op code were 11 in Figure 23, the Q data would be

		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 22. Sign Control Chart

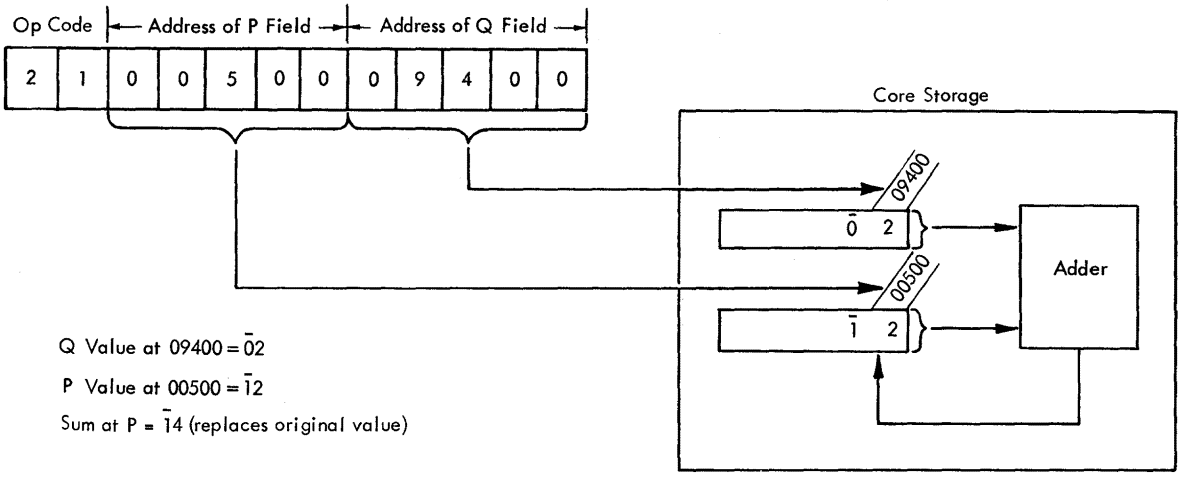


Figure 23. Add Instruction — Data Flow

0̄9400 and the result would be 12 (00 +1̄2). The three high-order positions of the Q data (0̄94) are not used because the P-data flag bit (above the one in 1̄2) stops the add operation. The Arithmetic Check indicator is turned on because the Q data field (09400) exceeds the two digits of data (1̄2) contained in the field defined by the P address (00500).

Timing. 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 High/Positive indicator is on if the difference is positive and not zero; the Equal/Zero indicator is on if the difference is zero. Neither indicator is on if the difference is negative.

Timing. Execution time is computed by using the Add instruction formula. Recomplement time is added to basic time when the signs of the field at the Q and P address 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.

Basic Execution Time: $T=10\ (6.5 + .5D_Q + D_P)$
Recomplement Time: $T=10\ D_P$

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.

Timing. 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 24, the multiplicand (1̄2) at 00500 is multiplied by the multiplier (0̄2) at 09400. The product (0̄024) is developed and stored at 00096-00099.

The 20 digits of the area in core storage specified as the "product area" (positions 00080 through 00099) are automatically cleared to zeros before multiplication begins. Formation of the product then proceeds serially from right to left until terminated by the flag bit marking the high-order position of the field at the Q address. A flag bit is stored in the high-order position of the product, and 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 core storage location. The Arithmetic Check indicator is not turned on when the product exceeds 20 digits in length. The High/Positive indicator is on if the product is positive and not zero; the Equal/Zero indicator is on if the product is zero. Neither indicator is on if the product is negative.

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

$$T=10\ (16 + D_Q + 4\,L_Z + 4\,D_P\ (D_Q - L_Z))$$

L_Z is the number of zeros in multiplier field

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.

Timing. $T=10\ (16 + D_Q + 4\,L_Z + 4\,D_P(D_Q - L_Z))$

L_Z is the number of zeros in multiplier field.

Divide (D-29)

Description. The divisor (Q address) is successively subtracted from the dividend. The dividend must be stored in the product area before a Divide instruction is given. A Load Dividend instruction (explained later) may be used to satisfy this requirement.

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 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 25 shows the manner in which the 1620 solves this problem.

Problem 2: $-\bar{2}1\bar{2}\ (212) \div \bar{2}4 = -8.83\ (\bar{0}088\bar{3})$ and a remainder of $\bar{0}8$. Figure 26 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.

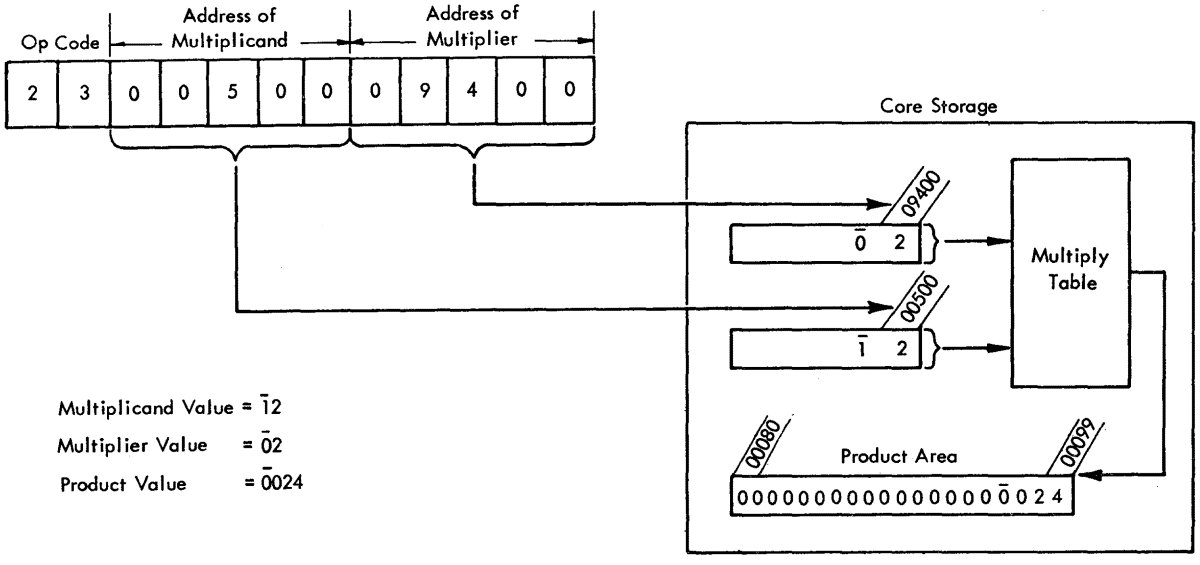


Figure 24. Multiply Instruction — Data Flow

Instruction	Data at Core Storage Addresses 00500	00600	Description	00092	00093	00094	00095	00096	00097	00098	00099
	$\overline{0}0500$	$\overline{0}0600$									
28 00099 00500	$\overline{4}906$	$\overline{2}3$	Load dividend	0	0	0	0	$\overline{4}$	9	0	6
29 00096 00600			Subtract divisor			−	2	3			
			Overdraw			9	8	1			
			Add divisor back to correct overdraw.			+	2	3			
						0	0	4			
			Store first leftmost digit of quotient (0) and flag bit	0	0	$\boxed{\overline{0}}$	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	$\overline{0}$	$\boxed{2}$	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	$\overline{0}$	2	$\boxed{1}$	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 ($\overline{0}213$) and remainder ($\overline{0}7$) in product area.	0	0	$\overline{0}$	2	1	$\boxed{3}$	$\overline{0}$	7

Figure 25. Divide, Problem 1

Instruction	Description	00650	00500	00090	00091	00092	00093	00094	00095	00096	00097	00098	00099
	Data	$\bar{2}4$	$\bar{2}1\bar{2}$										
LD 28 00097 00500	Reset 00080 - 00099 to zeros. Transmit dividend to 00097. Dividend sign to 00099.			0	0	0	0	0	$\bar{2}$	1	2	0	$\bar{0}$
D 29 00095 00650	Subtract divisor from dividend starting at 00095.							−	2	4			
	Overdraw							9	7	8			
	Correction							+	2	4			
								0	0	2			
	Store first quotient digit (0) and flag bit			0	0	0	$\boxed{\bar{0}}$	0	2	1	2	0	$\bar{0}$
	Subtract one place to the right							−	2	4			
	Overdraw							9	9	7			
	Correction							+	2	4			
								0	2	1			
	Store 2nd quotient digit (0)			0	0	0	$\bar{0}$	$\boxed{0}$	2	1	2	0	$\bar{0}$
	Subtract one place to the right								−	2	4		
	Successful subtraction								1	8	8		
	7 more successful subtractions (7 x 24 = 168)								−1	6	8		
									0	2	0		
									−	2	4		
	Overdraw								9	9	6		
	Correction								+	2	4		
									0	2	0		
	Store quotient digit (8)			0	0	0	$\bar{0}$	0	$\boxed{8}$	2	0	0	$\bar{0}$
	8 successful subtractions (8 x 24 = 192)								−	1	9	2	
	(Overdraw and correction not shown)									0	0	8	
	Store quotient digit (8)			0	0	0	$\bar{0}$	0	8	$\boxed{8}$	0	8	$\bar{0}$
	3 successful subtractions (3 x 24 = 72)										−	7	2
												0	8
											−	2	4
	Overdraw										9	8	4
	Correction										+	2	4
											0	0	8
	Store quotient digit (3)			0	0	0	$\bar{0}$	0	8	8	$\boxed{3}$	0	$\bar{8}$
	Store flag over leftmost 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 26. Divide, Problem 2

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

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

The High/Positive indicator is on if the quotient is positive and not zero; the Equal/Zero indicator is on if the quotient is zero. Neither indicator is on if the quotient is negative.

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

Timing. $T=10\ (6 + 13.5\ Q_T + 9.75\ D_V\ Q_T)$. 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 Divident Immediate instruction is used, the divide operation execution time may be considered as the total time for both the Load Dividend and Divide instructions.

$$T=10\ (23.5 + 1.5\ D_N + 13.5\ Q_T + 9.75\ D_V Q_T)$$

D_N is the number of digits in the dividend

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 26 are 2.4 and 21.200, respectively, the quotient 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.

INCORRECT DIVISOR POSITIONING

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

Overflow. As illustrated in Figure 27, 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 28 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, that is, 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.

SUMMARY OF DIVISION RULES

1. Divide (D-29- or DM-19)
   a. P address = 00100 minus the length of the quotient. The quotient length is 100 minus the P address.
   b. Q address = core storage address of the divisor.
2. Quotient address = 00099 minus the length of the divisor.
3. Quotient length = 100 minus P address.
4. Remainder address = 00099.
5. Sign of quotient: determined by the algebraic signs of the dividend and divisor.
6. Sign of remainder: same as that of the dividend.
7. Decimal point location: the number of dividend decimal digits minus the number of divisor decimal digits equals the number of quotient decimal digits.

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 27. Divide Overflow

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.

Timing. Same as Divide (D-29).

Load Dividend (LD-28)

Description. In division operations, 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 rightmost dividend digit and terminating at the flag bit marking the leftmost position of the dividend field. The P address is 00099 minus the number of zero positions desired to the right of the dividend.

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

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 28. Division Error, Incorrect Programming

Example. Two Load Dividend instructions and one Load Dividend Immediate instruction are shown in Figure 29.

1. The Load Dividend instruction,

   28 00096 00650,

   causes the low-order position of the dividend to be placed at 00096. The sign (minus) is stored at 00099.

2. The Load Dividend instruction,

   28 00099 00650,

   causes the low-order position of the dividend to be placed at 00099. The sign (plus) is stored at 00099.

3. The Load Dividend Immediate instruction,

   18 00098 $\bar{0}$0650,

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}$1365	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 29. Load Dividend Instruction

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.

Timing. $T=10\ (17.5 + 1.5\ D_N)$ 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.

Timing. Same as Load Dividend (LD-28).

SUMMARY OF LOAD DIVIDEND RULES

1. Load Dividend (LD-28 or LDM-18)
   a. P address = 00099 minus the number of zeros desired to the right of the units position of the dividend.
   b. Q address = core storage address of the dividend.

Logic Instructions

Logic instructions consist of two types of instructions:

Compare Instructions
Branch Instructions

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

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.

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 discarding the digits of the difference. Neither field is altered. The result of the comparison is shown by the on/off condition of the following 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 until terminated by the flag bit marking the leftmost position of the field at the P address. High-order zeros are supplied when the size of the Q field is less than that of the P field. The High/Positive indicator is turned on if the P address data is algebraically higher than the Q address data, and off, if not higher. The Equal/Zero indicator is turned on if the P address data is algebraically equal to the Q address data, and off, if not equal.

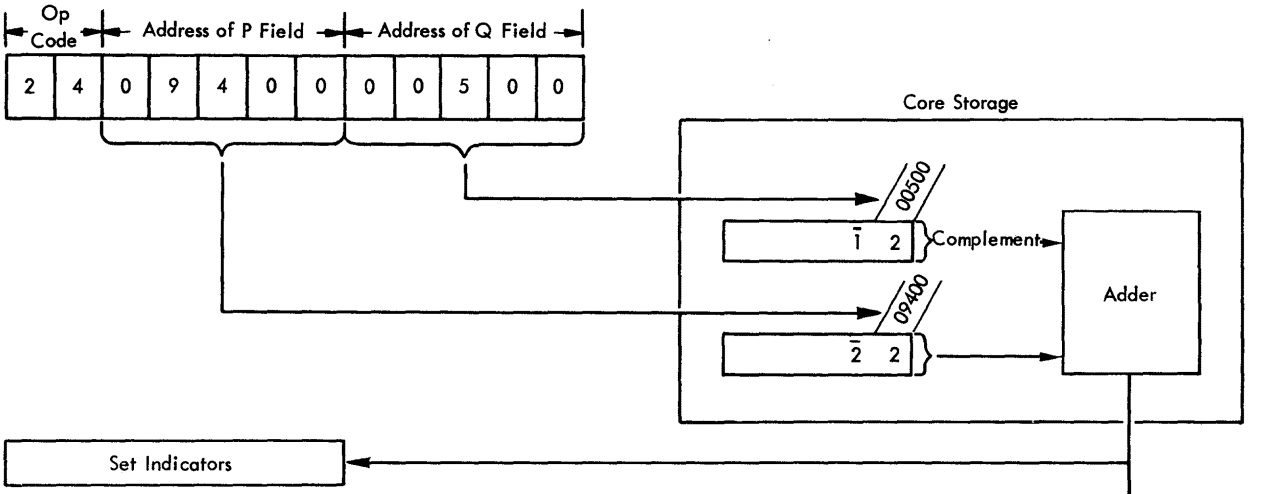
In Figure 30, the Q address data (12) is subtracted (complement-added) from the P address data (22). The difference (+10) is discarded and the indicators are set as follows:

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 Equal/Zero indicator is turned on.

The minimum length of the two fields being compared is two digits, with the exception that one-digit fields can be compared if the signs of the two fields are not alike.



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 30. Compare Instruction — Data Flow

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

1. Numeric mode

   0 1 2 3 4 5 6 7 8 9

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

The record mark (≠), group mark (≢), and numeric blank cannot be used as data in an arithmetic or compare operation.

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

$$T = 10\ (6.5 + .5\,D_Q + D_P)$$

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

$$T = 10\ (8 + 1.5\,D_Z)$$

Compare Immediate (CM-14)

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

Timing. Same as Compare.

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.

Timing. T = 40.

Branch on Digit (BD-43)

Description. Branch to the instruction at the P address if the digit at the Q address is not 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.

Timing. T = 70.

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.

Timing. T = 70.

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 (rightmost position) of the record, resulting from the end-of-line (EL) punch at the end of the tape record. Reading and processing 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.

Timing. T = 70.

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 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.

Timing. T = 70.

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.

Timing. T = 60. Information concerning 1620 indicators is given in Table 4 and further explained, as follows:

1620 INDICATORS

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 unit.

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 High/Positive (11) and Equal/Zero (12) indicators with one Branch Indicator or Branch No Indicator instruction—no indicators are turned off by this instruction. The H/P or E/Z indicator is turned off only when both the H/P (11) and E/Z (12) indicators are off.

Table 4. 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, Reset key, or Check Reset key
	07	Write Check	Yes	I/O Output Error	BI, BNI, Reset key, or Check 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, MIR-E	BI, BNI,Check Reset or reset key
	17	MBR-O Check	Yes	Parity Error in MBR-O, MIR-O	BI, BNI,Check Reset or reset key
	19	Any Check	No	Indicator 06, 07, 16, 17, 25, or 39 on	Indicators 06, 07, 16, 17, 25, and 39 off, or check Reset key
	30	IX Band 0	No	Power on or Branch and Select instruction	Power off or Branch and Select Instruction
	31	IX Band 1	Yes	Branch and Select instruction	Power off or Branch and Select instruction
	32	IX Band 2	Yes	Branch and Select instruction	Power off or Branch and Select instruction
1311	36	Address Check	Yes	Unequal address, or no address found in disk storage, or multiple heads, or multiple drives are selected.	BI, BNI, Check Reset, or Reset keys 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, Check Reset, or Reset key, or disk operation
	38	Cylinder Overflow	Yes	Disk operation completes last sector and sector count is not 000	BI, BNI, Check Reset or Reset keys, or disk operation
	39	Any Disk Error	No	36, 37, or 38 on	Reset of 36, 37, and 38
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	Chanel 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)

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 in the description of Floating-Point Operations.

MBR-E and MBR-O (16 and 17). All data leaving core storage does so via the MBR registers. Their associated indicators (16 and 17) are turned on if a parity error occurs. These indicators are also turned on by a parity error in the Memory Inhibit Register (MIR).

Any Check (19). This indicator provides the means of interrogating six error conditions (Indicators 6, 7, 16, 17, 25, 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 six error conditions are off.

Index Registers Indicators (30, 31, and 32). These indicators are used with the special feature Index Registers, and are described in more detail in the functional description of that feature. These indicators are turned on by the Branch and Select instruction; Indicator 30 is also turned on when power is turned on. These indicators are not turned off by testing them.

1311 INDICATORS

Address Check Indicator (36). With the exception of read track mode instructions, reading and writing operations in disk storage are contingent upon an equal comparison between a sector address on the disk and the sector address in OR-1. If an address matching the sector address in the disk control field is not found on the designated track within one complete disk revolution, the operation terminates and the Address Check indicator and light turn on.

Wrong-Length Record/Read-Back Check Indicator (37). This dual-purpose indicator turns on (1) upon detection of a Wrong-Length Record Check error, or (2) in a Disk Check operation when the data from disk storage does not compare, bit-by-bit and character-by-character, with the data in core storage.

Cylinder Overflow Indicator (38). This indicator and light turn on if a disk operation completes the last sector of a cylinder without the sector count being decremented to 000. This condition terminates the operation.

Any Disk Indicator (39). This indicator facilitates the testing of all indicators associated with disk storage. Indicator 39 is turned on by Indicators 36, 37, or 38; it is turned off when the same indicators are reset by interrogation.

1443 INDICATORS

Printer Check (Code 25). This indicator is turned on by a parity error or a Sync Check in the 1443. Because the 1443 prints buffered data, either of these errors can occur at any time with respect to 1620 operation. The 1620 Any Check indicator (19) is turned on when the Printer Check indicator is turned on.

Channel 9 (Code 33). This indicator is turned on when a punched hole is detected in Channel 9 of the carriage control tape.

Channel 12 (Code 34). This indicator is turned on when a punched hole is detected in Channel 12 of the carriage control tape.

Printer Busy (Code 35). This indicator is on when the 1443 is printing and the buffer is unavailable for loading. The indicator is off when printing is complete and the buffer can accept data from core storage.

Branch No Indicator (BNI-47)

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

Timing. T = 60.

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. Transmission proceeds serially from right to left until terminated by the flag bit that marks the leftmost position of the field at the Q address. The field at Q remains unchanged. The instruction at the P address is the next one executed.

Figure 31 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 Branch and Transmit instruction. The last instruction of the subroutine, Branch Back (explained later), returns the program to the main routine (the address saved in IR-2) when the Branch and Transmit was executed.

In Figure 32, 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 00000.

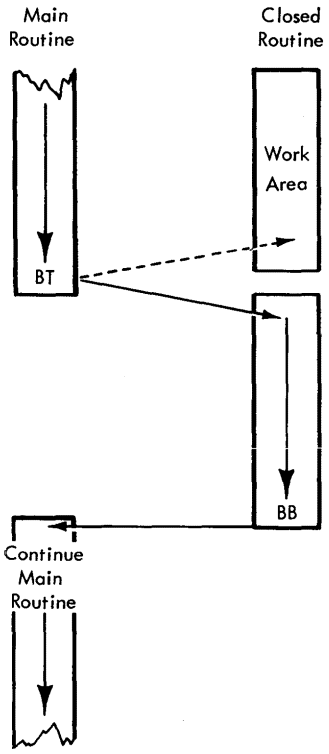


Figure 31. Subroutine Linkage

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.

Timing. $T = 10\ (7.5 + 1.5\ D_Q)$

Branch and Transmit Immediate (BTM-17)

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

Timing. $T = 10\ (7.5 + 1.5\ D_Q')$

Branch and Transmit Floating (BTFL-07)

Description. This instruction can be used to transmit fields during floating-point arithmetic operations. It can be used in either floating-point subroutines or used in conjunction with the special feature Automatic Floating-Point Operations. 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. The mantissa and the exponent in the Q field are not altered in core storage. The Q address is normally the rightmost position of the exponent. The operation is the same as the regular Branch and Transmit instruction (BT-27), except that in the transmit function flags in the three low-order positions of the Q address are ignored as indications to terminate the transmittal. Beginning with the fourth low-order position, a flag bit terminates the operation. All flag bits are transmitted.

Timing. $T = 10\ (9.5 + 1.5\ L)$
L = number of digits in mantissa.

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 instruction, or (2) the address saved in PR-1 by previous depression of the Save key on the console when the computer was in manual mode.

The Save key function is examined first since it has priority over a Branch and Transmit instruction. If the Save key function is active, the 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 instruction was executed. The contents of IR-2 are transferred 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 MAR check results. The P and Q addresses of the Branch Back instruction are not used.

Timing. $T = 20$.

Branch and Select (BS-60)

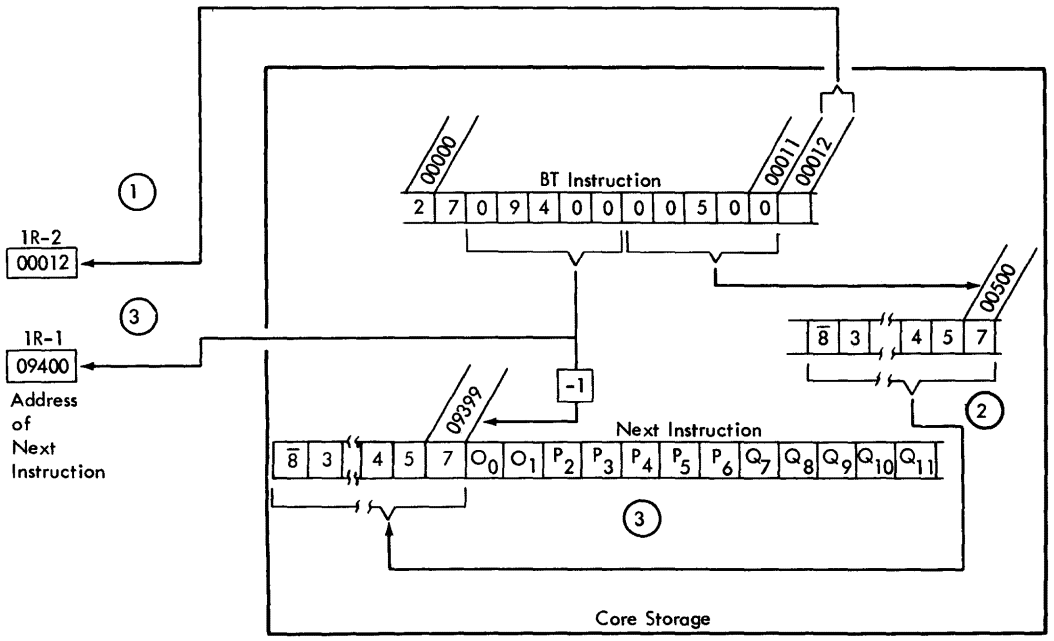
Description. This instruction enables programs to be run with or without indirect addressing. The Q_{11} digit of the instruction is examined for an 8 or 9: an 8 turns off indirect addressing, a 9 turns it on. Whichever selection is executed remains in effect until changed by another Branch and Select instruction or a power-off operation. A power-on operation causes indirect addressing to be operative. This instruction is also used with the special feature Index Registers. Its additional usage is described in the functional description of this feature.

The branch to the P address is unconditional. The Q_7-Q_{10} positions of the instruction are not used.

Timing. $T = 60$.

Branch and Transmit Address (BTA-20)

Description. The address of the next instruction in sequence is stored in 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.



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 32. Branch and Transmit — Data Flow

The field at the Q address remains unchanged. The instruction at the P address is the next one executed. The operation is the same as the regular Branch and Transmit instruction, except that in the transmit function any flags in the four low-order positions are ignored as indications to terminate the transmission. Beginning with the fifth low-order position, detection of a flag bit terminates the operation.

Figure 32.1 illustrates the data flow for a Branch and Transmit Address instruction. In this example, a six-digit field is illustrated; however, a flag over the fifth position (address 00496) would limit the size of the field to five digits.

Timing. 10 (7.5 + 1.5 D_Q) μsec.

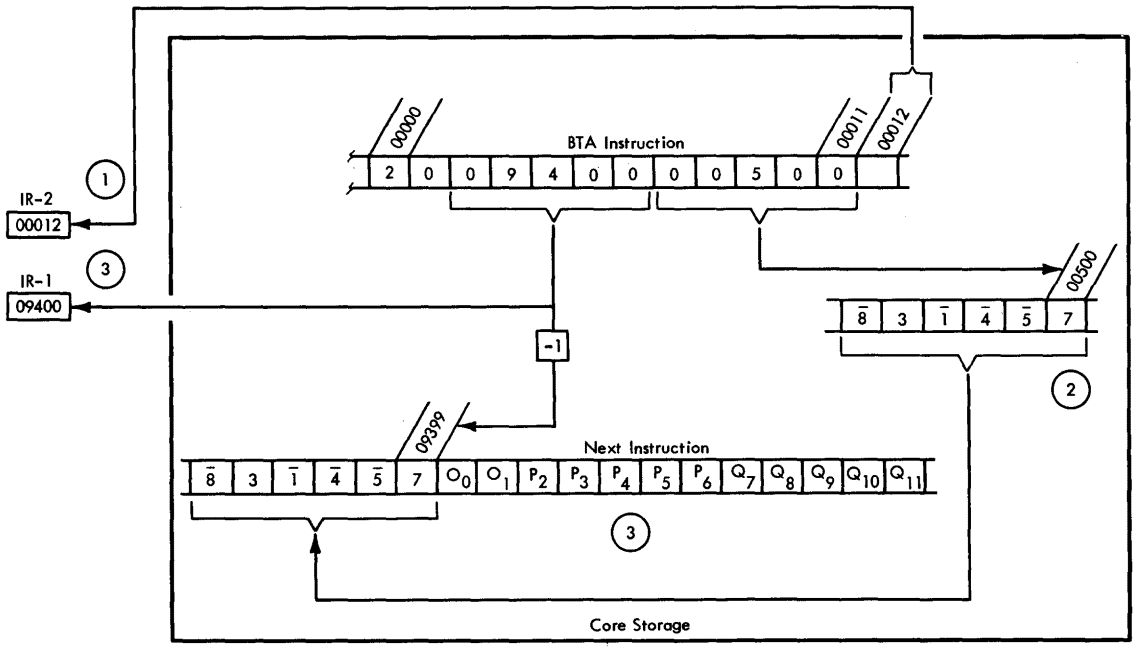
Branch and Transmit Address Immediate (BTAM-10)

Description. Same as the Branch and Transmit Address instruction, except that transmitted data starts with Q_{11}. A digit to the left of Q_8 must be flagged to end transmission of digits.

Timing. 10 (7.5 + 1.5 D_Q) μsec.

Internal Data Transmission Instructions

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



Circled numbers refer to these operations:

(1) Save address of next sequential instruction in 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 32.1. Branch and Transmit Address Operation

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.

Timing. T = 80.

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 33, 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.

Timing. T = 80.

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 leftmost 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 34, the data (214) 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.

Timing. The execution time varies according to the number of digits (high-order zeros included) in the Q field.

$T = 10(6.5 + 1.5\ D_Q)$ average

Transmit Field Immediate (TFM-16)

Description. The description is the same as that for Trasmit Field except that the data in the Q part of the instruction is used in place of the data at the Q address.

Timing. $T = 10\ (6.5 + 1.5\ D_Q')$ average.

Transmit Floating (TFL-06)

Description. This instruction can be used to transmit fields during floating-point arithmetic operations. It can be used in either floating-point subroutines or used in conjunction with the special feature Automatic

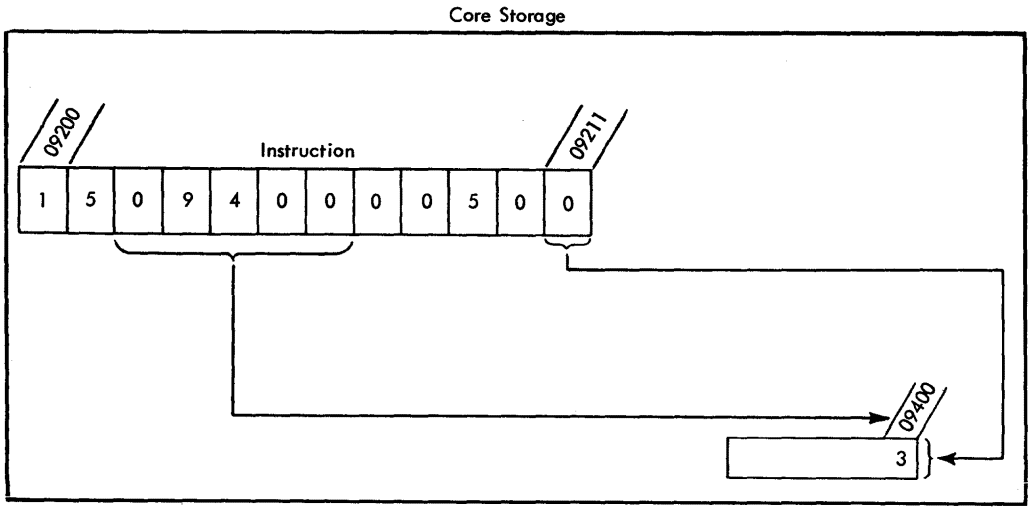


Figure 33. Transmit Digit Immediate – Data Flow

Floating-Point Operations. The field at the Q address is transmitted to the location designated by the P address. The mantissa and the exponent in the Q field are not altered in core storage. The Q address is normally the rightmost 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 rightmost positions of the Q address are ignored as indications to terminate the transmittal. Beginning with the fourth low-order digit, a flag bit terminates the operation. All flag bits in the field are transmitted.

Timing. $T = 10\ (9.5 + 1.5\ D_Q)$ average.

Transmit Record (TR-31)

Description. The record 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.

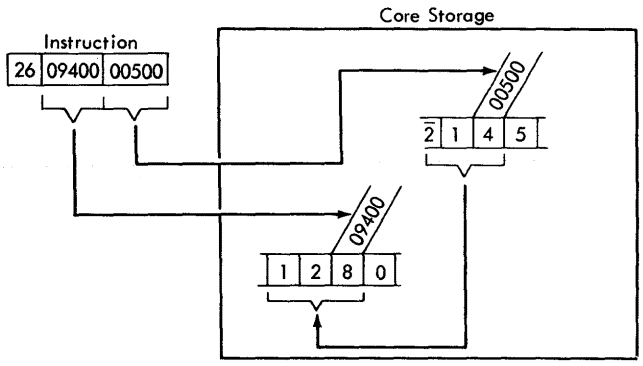


Figure 34. Transmit Field – Data Flow

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.

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

$T = 10\ (6.5 + 1.5\ D_Q)$ average

Transmit Record No Record Mark (TRNM-30)

Description. This instruction is the same as the Transmit Record (TR-31) instruction except that the record mark character in the Q field, which terminates the operation, is not transmitted to the P field.

Timing. $T = 10\ (6.5 + 1.5\ D_Q)$ average

Transfer Numeric Strip (TNS-72)

Description. This instruction converts numeric data that is in the two-digit alphameric mode into single-digit numerical data, with sign. The rightmost 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 rightmost position of the field that is to contain numeric data is specified by the Q address. Transmission of the numerical digits from the odd-numbered positions of the alphameric field 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 Q field prior to this instruction to define the leftmost position. It remains unchanged by the instruction. For example, the numeric digits $\overline{4}$, 3, 2, and 1 in Figure 35 are stripped

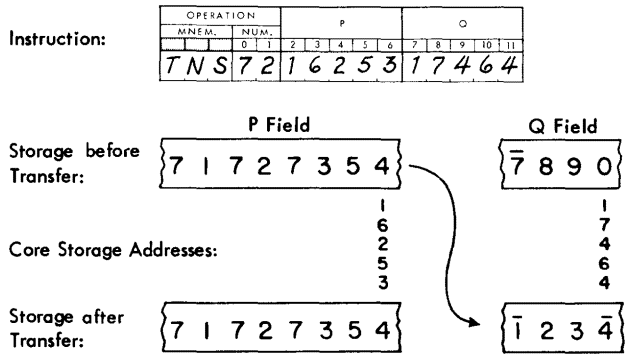


Figure 35. Transfer Numeric Strip

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 5 or 1 in the units zone position indicates a negative quantity from punched cards or paper tape and is converted by this instruction to a flag bit over the units digit of the numeric field (Q field). A 2 in the units zone position of the alphameric field occurs when a negative zero, indicated by an X alone, is read in from punched cards or paper tape. This 2, 0 combination is converted to a flagged 0 by this instruction. A 2 in the units zone position with a nonzero digit in the units numerical position does *not* result in a flag being placed over the units digit of the numeric field. The following three examples illustrate these points.

P Field *(Before Transfer)*	Q Field *(After Transfer)*
$\overline{7}1727354$	$\underline{1}23\underline{4}$
$\overline{7}1727320$	$\underline{1}230$
$\overline{7}1727325$	$\underline{1}235$

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 leftmost 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 rightmost 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 effect 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 this instruction is executed, that the flags do not disturb the numeric field.

Timing. $T = 10\ (6 + D_P)$ average

Transfer Numeric Fill (TNF-73)

Description. This instruction moves and expands single-digit numeric data with sign, into two-digit alphameric data. The rightmost 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 rightmost position of the numeric 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 rightmost position of the numeric (Q) field. The digits in the numeric field, including the digit in the leftmost (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 36, the numeric digits $\overline{1}$, 9, 8, and $\overline{7}$ fill in the alphameric field locations 16257, 16255, 16253, and 16251, 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 rightmost position of the numeric field is converted to a 5 in the zone position of the rightmost position of the alphameric (P) field. Absence of a flag in the rightmost position of the numeric field results in a 7 being placed in the zone position of the rightmost position of the P field. All other zone positions of the alphameric field are automatically filled with 7's.

Timing. $T = 10\ (6 + D_P)$ average

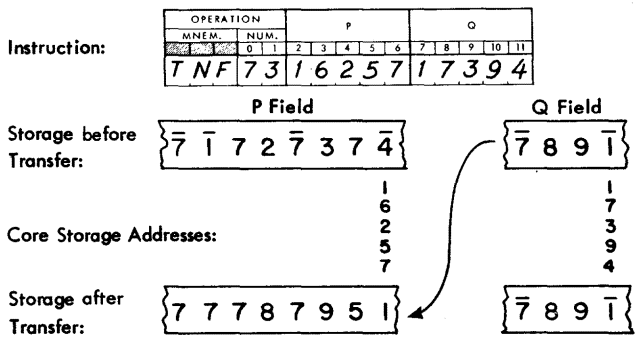


Figure 36. Transfer Numeric Fill

Input/Output Instructions

1620 Input/Output (I/O) instructions enable the transfer of data between core storage and disk storage, and core storage and I/O 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 P address of the instruction determines where data is transferred to or from. The Q_8 and Q_9 digits specify the I/O unit, as follows:

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

The Q_7 and Q_{10} positions of the instruction are not used. The Q_{11} position is used in 1311 Disk Storage and 1443 Printer operations. Because the Disk Storage instructions consist of many variations of the basic I/O instruction, they are described separately.

The operation of and the physical characteristics of the I/O units are described in the following publications:

IBM 1621 Paper Tape Unit (Form A26-5836)
IBM 1622 Card Read-Punch (Form A26-5835)
IBM 1443 Printer for 1620/1710 Systems (Form A26-5730)
IBM 1627 Plotter (Form A26-5710)
IBM 1311 Disk Storage Drive (Form A26-5650)

READ CHECK AND WRITE CHECK INDICATORS

The Read Check and Write Check indicators (Codes 06 and 07), respectively, are turned on if a parity 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.

Read Numerically (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. *Paper Tape Reader:*
   Sensing of the end-of-line character when paper tape is being read. At this time a record mark character is generated automatically by the machine and placed in core storage following the last character read from tape.
2. *Console Typewriter:*
   Depression of the Release key on the console and the R-S key on the typewriter 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. *Card Reader:*
   Reading the 80th character from the card input buffer storage into core storage. Each numeric character from an input unit 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.

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 subsequent 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 Appendix D.

Timing. Execution times for the paper tape reader and typewriter depend upon the speed of the input unit selected and the number of characters read. The

Paper Tape Reader reads 150 characters a second (cps), and the maximum typewriter speed is 15.5 cps. Execution time is 1.7 ms for transferring 80 columns of data from the 1622 input buffer storage to core storage.

Read Alphamerically (37)

Description. Alphameric information from an input unit is trasmitted 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 information 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. *Paper Tape Reader:*
   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 that last character read from tape.
2. *Console Typewriter:*
   Depression of the Release key on the console and the R-S key on the typewriter 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. *Card Reader:*
   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 numeric zero digit (C bit) followed by a single record mark character (coded C-8-2).

Numeric data stored in the two-digit alphabetic mode must be converted by programming to single-digit numeric data before being used in arithmetic commands. The Transfer Numeric Strip instruction 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.

Timing. Same as Read Numerically.

Write Numerically (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. *Tape Punch:*
   Sensing of a record mark or group mark character in core storage. The record mark or group mark character causes an end-of-line character to be punched in paper tape.
2. *Console Typewriter:*
   Sensing a record mark or group mark character in core storage, or pressing the Release key on the console. If the Release key is not pressed, and a record mark or group 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. *Card Punch:*
   Writing the 80th position in card output buffer storage.
4. *Printer:*
   Writing the 120th (or 144th) position in Printer buffer storage, or sensing a record mark or group mark in core storage. 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. *Plotter:*
   Sensing a record mark or group mark in core storage.

Each numeric character in core storage, and its flag bit (if any), is written to an output unit, except in 1627 Plotter operations, in which numeric characters sent to the Plotter are translated into plotting commands. The character in core storage remains unchanged. No alphameric or special character represented in core storage as two numeric characters can be written as a single character by this instruction.

Timing. Execution times for the Paper Tape Reader and Typewriter depend on the speed of the output unit selected and the number of characters written. The Tape Punch punches 15 cps and the Console Typewriter operates at 15.5 cps. For the Card Punch, the execution time equals 1.7 ms for transferring 80 col-

umns of data from core storage to 1622 output buffer storage. The time required to load the Printer Buffer is 2.1 ms. The Model 1 Plotter requires 3.3 ms for each character transferred; the Model 2 Plotter requires 5 ms for each character transferred. However, if one-character records are used, the CPU is released after 200 μsec.

Write Alphamerically (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. *Tape Punch:*
   Sensing of the alphameric record mark or group mark, which causes an end-of-line character to be punched in the tape. If the Release key is pressed first, the end-of-line character is not punched. If the Release key is not pressed and no alphameric record mark or group mark is encountered before the data from the highest-numbered core storage address is written, the machine "loops back" to 00000, and transmission continues.
2. *Console Typewriter:*
   Sensing of an alphameric record mark or group mark, or depression of the Release key on the console. If this is done before an alphameric record mark or group mark has been encountered in core storage, a record mark character is not written. If the Release key is not pressed and no alphameric record mark or group 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. *Card Punch:*
   Writing of the 80th position in card output buffer storage.
4. *Printer:*
   Writing the 120th (or 144th) position in Printer buffer storage, or sensing a record mark or group mark in core storage. An alphameric record mark or group mark at the P-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. *Plotter:*
   Sensing a record mark or a group mark in core storage.

   Each alphameric character in core storage is written on the output unit as a single character except for 1627 Plotter operations, during which alphameric characters sent to the Plotter are translated into plotting commands. 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.

Timing. Same as Write Numerically.

Dump Numerically (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 the 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 overflows to the "low" end of the next module, or "wraps around" to address 00000 and successively higher-numbered addresses as required to completely punch out all 80 columns of the last card.

Transmission to any output unit may also be terminated at any time by pressing the Release key on the 1620 console.

In Printer operations, data is transferred to the print buffer from core storage, beginning at the P address and continuing through higher-numbered addresses until the buffer is filled. If the highest-numbered position of core storage is reached before the buffer is filled, data transfer continues from core storage addresses 00000, 00001, etc. Record mark and group mark characters are transferred in the same manner as other characters. Only one print line (one buffer load) occurs per Dump instruction. The printer output of numeric characters is shown in Table 5.

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 0 punchout on cards, each numeric character, as well as any single record mark character,

Table 5. Printer Output

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

is written on the output unit along with its flag bit (if any), except in 1627 Plotter operations. (In Plotter operations numeric characters sent to the Plotter are translated into plotting commands.) 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 a hyphen (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.

Timing. Same as Write Numerically.

1311 Disk Storage Drive Instructions

NOTE: The following descriptions of disk storage instructions are arranged in the following manner.

1. Seek Instruction
2. Instructions using group marks to establish correct record length:
   a. Sector Mode
      Read Disk/WLRC
      Write Disk/WLRC
      Check Disk/WLRC
      Disk operations with records of less than or in multiples of 100 characters
   b. Track Mode
      Read Disk Track/WLRC
      Write Disk Track/WLRC
      Check Disk Track/WLRC
3. Instructions not using group marks:
   a. Sector Mode
      Read Disk
      Write Disk
      Check Disk
   b. Track Mode
      Read Disk Track
      Write Disk Track
      Check Disk Track

A summary of disk storage instructions is provided in Table 6. 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 (ms)

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 OR-1. 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 within *the same cylinder.*

Timing. The 1620 Central Processing Unit is interlocked only until the sector address is transferred to the disk control unit (160 μsec). Internal 1620 processing that does not involve disk storage can then continue and overlap the seek operation. Calculating average seek times is described in the section of the manual entitled TIMING.

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) provides 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 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 OR-1, 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 OR-1 is incremented by one each time a sector is read. The sector addresses for each succeeding sector read are compared with the addresses in OR-1 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-2.

Table 6. Summary of Disk Storage 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.

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 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 indicator (06) to be turned on. The MBR-even indicator (16) or the MBR-odd 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.

Timing. T = 22 + 2S average

Write Disk/WLRC (WDGN or 38 — Q_{11} of 0)

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 OR-1 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 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) turns 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) turns on. Therefore, the greatest number of sectors that can be written with one instruction is 200, one full cylinder.

Indicators 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 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.

Timing. T = 22 + 2S average

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 OR-1. 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) turns on.

When the sector address in OR-1 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 OR-1 is incremented by one each time a sector is written, 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 indicator (06) to turn on. In addition, the MBR-even indicator (16) or the MBR-odd indicator (17) turn 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.

Timing. T = 22 + 2S, average

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 into 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 (as in the example above), 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 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 OR-1 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.

Timing. T = 62 average

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.* The Compare-Disable switch on the master disk storage drive can be turned on to allow writing a full track without address comparison. *Caution* in the use of the Compare-Disable switch is necessary. Before writing tracks in this manner, determine that the proper disk pack is mounted and that the track should be written.

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.

Timing. T = 62 average

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 5 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 turn on.

Timing. T = 62 average

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.

Timing. T = 22 + 2S average

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.

Timing. T = 22 + 2S average

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, and the succeeding data written to fill out the 100 sector positions, are compared bit-by-bit and character-by-character.

Timing. T = 22 + 2S average

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.

Timing. T = 62 average

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.

Timing. T = 62 average

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.

Timing. T = 62 average

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 into disk storage, or following the last position of the area in core storage reserved for reading data from disk storage. Figure 37 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 of 100 (i.e., 200-position record) and, therefore, the group mark was *not* written in disk storage.

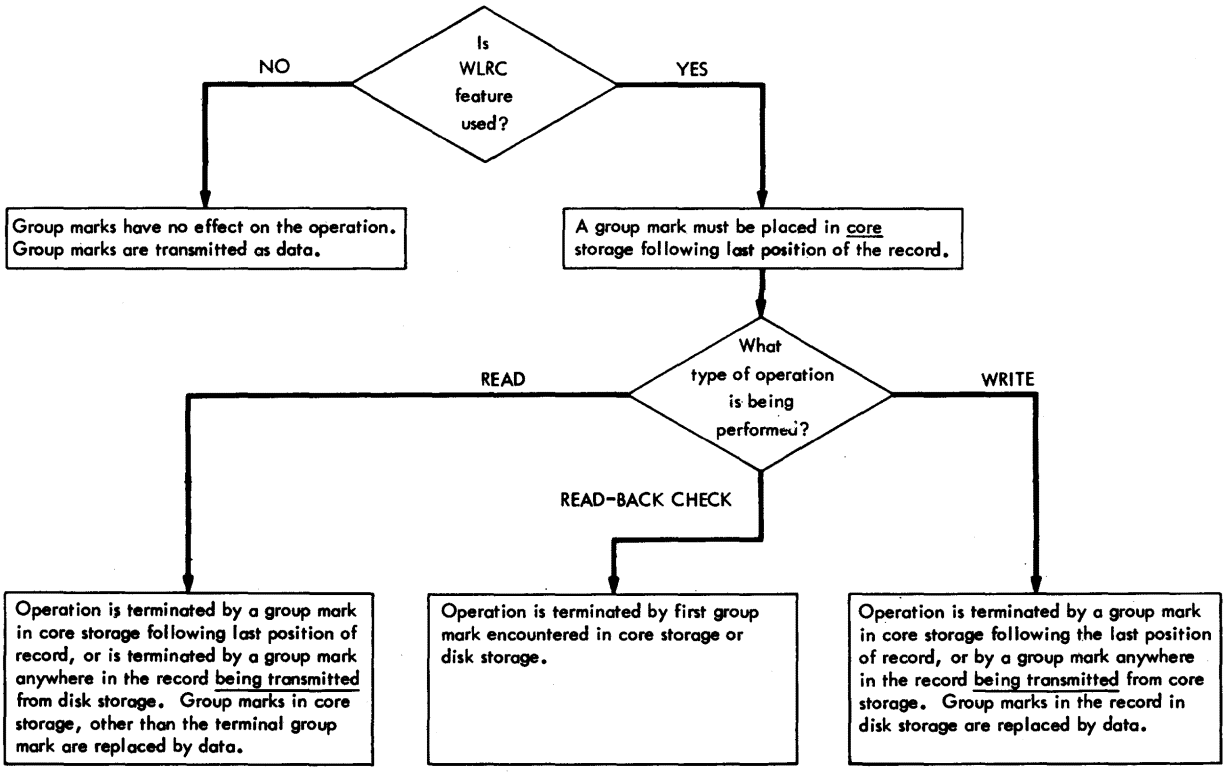


Figure 37. Effects of Group Marks on Disk Storage Operations

Program Control Instructions

Control (K-34)

Description. This instruction is used for forms control on the console typewriter and on the 1443 Printer. The Q_8 and Q_9 digits contain 01 for console typewriter operations and 09 for 1443 Printer operations. The digit Q_{11} contains the control code for typewriter operations, and Q_{10} and Q_{11} contain the control code for 1443 Printer operations, as illustrated in Figure 38.

Q_7 and the P address are not used.

CONSOLE TYPEWRITER	
Operation	Q Code
Space	1
Return element	2
Back space	3
Index (line space - no element return)	4
Tabulate	8

1443 PRINTER			
Operation		Q_{10} and Q_{11} Code	
		Space before print	Space after print
One space		51	21
Two spaces		52	62
Three spaces		53	63
Operation		Skip before print	Skip after print
Skip to Channel	1	71	41
	2	72	42
	3	73	43
	4	74	44
	5	75	45
	6	76	46
	7	77	47
	8	78	48
	9	79	49
	10	70	40
	11	33	03
	12	34	04

Figure 38. Forms Control Codes

Timing. The execution time depends upon the particular control function performed.

In operations involving the 1443 Printer, the CPU is interlocked for only 60 μsec; after this time other operations that do not involve the 1443 can be overlapped with the control operation. The time required to perform the skipping and spacing function is:

Spacing and skipping before printing	45 ms for the first line, plus 10 ms for each additional line
Spacing and skipping after printing	First two lines included in print time; 10 ms for each additional line

Approximate timings for operations involving the console typewriter are shown in the following table:

Operation	Time	CPU Interlocked	Available Overlap Time
Return Element	800 ms	124 ms	676 ms
Tabulate	250 ms	56 ms	194 ms
Space	56 ms	56 ms	None
Backspace	56 ms	56 ms	None
Index	124 ms	124 ms	None

The time to return the printing element is shown here as the maximum time, from position 85 to position 1; the time to tabulate is shown here as the time to tabulate across 20 of the printing positions. The timings given here are provided to indicate the overlap operations possible. These approximate times should be sufficient for most timing requirements. More precise time

can be computed for these two operations by multiplying the number of print positions that the printing element is returned or tabulated, by 5.88 ms (the time required to move the print element one position). The overlap time available is computed by subtracting the CPU interlock time from the total time.

NOTE: If the console typewriter is not used by the program for approximately five minutes, the drive motor is turned off to reduce noise and wear. However, the typewriter is still in ready status, although approximately 0.5 sec is required to respond to a write or control instruction.

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.

Timing. T = 70.

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.

Timing. T = 70.

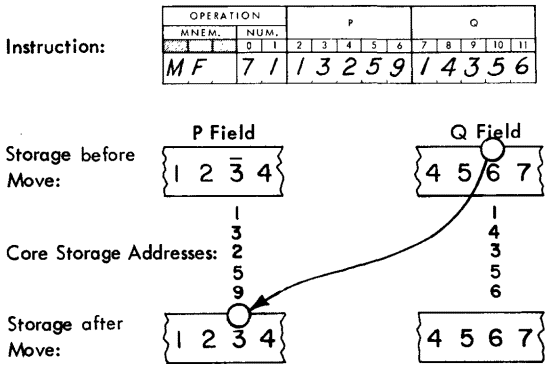
Move Flag (MF-71)

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, this 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 the location specified by the Q address has a flag, that flag position is cleared and a flag is placed at the P address. Thus, after the instruction is executed, the location specified by the P address reflects the original absence or presence of a flag at the Q address; a flag at the Q address is always cleared.

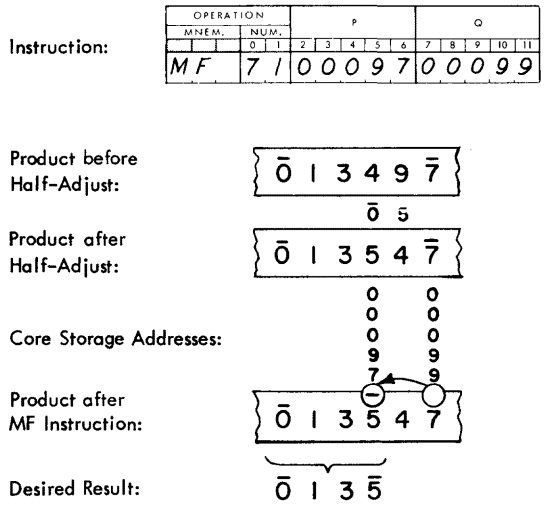
Figure 39 illustrates the movement of a positive sign which, in this example, *removes* a flag; Figure 40, the movement of the negative sign; Figure 41, the lengthening of a field. All three illustrate the programming required.

This instruction can also be used in the case where only one position of the product is dropped after half-adjustment and the product is either negative or positive. The programming for such a situation, using the



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

Figure 39. 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 40. Move Flag, Negative Sign

Move Flag instruction, is shown in Figure 42. The first Move Flag instruction saves the sign by placing it over its own O_0, or leftmost position. The second Move Flag instruction returns the sign to the new rightmost position of the half-adjusted result. The flag bit can be stored in any position in which it cannot be mistaken for a field-definition flag, a sign flag, or an indirect-address flag.

Timing. T = 80.

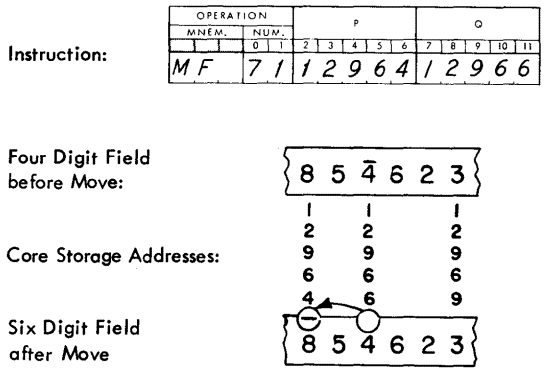


Figure 41. Move Flag, Lengthen Field

Halt (H-48)

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

Timing. T = 60.

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.

Timing. T = 60.

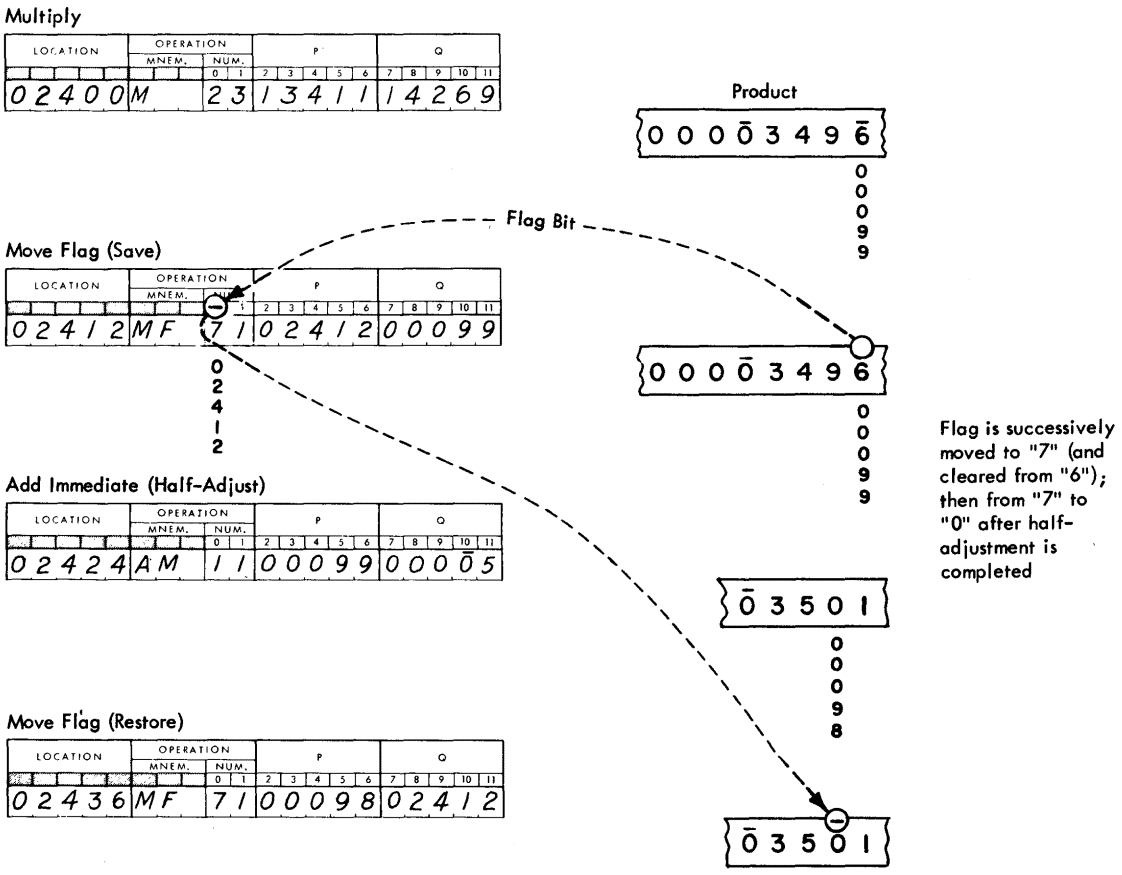


Figure 42. Move Flag to Half-Adjust One Position

Floating-Point Instructions

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 analysis to determine the size and range of intermediate and final results. This analysis 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, analysis 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 leftmost 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 rightmost positions of the field, and the mantissa is in the remaining high-order positions, as shown:

$$\bar{M} \ldots . M\bar{E}E$$

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 leftmost 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 leftmost (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 rightmost position. A floating-point number with a negative mantissa and a negative exponent is represented as follows:

$$\bar{M} \ldots . \bar{M}\bar{E}\bar{E}$$

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

Six floating-point instructions are provided: four are for arithmetic computations, Floating Add, Floating Subtract, Floating Multiply, and Floating Divide; two are used to control field size and location, Floating Shift Right and Floating Shift Left. Two additional instructions are used in floating-point operations but they are basic instructions in the system; Transmit Floating and Branch and Transmit Floating. The six floating-point instructions depend upon 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.

All instructions are in 1620 format of a 2-digit Op code, 5-digit P address, and 5-digit Q address.

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
Mq = mantissa of the field at the Q address
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 leftmost 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 $\bar{0}682349405$ should be entered as $\bar{6}823494004$, 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 rightmost positions of the exponents.

Floating Add (FADD-01)

Description. Mq is added to Mp and the result replaces Mp. Mq and Eq are not altered in core storage. Depending 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 43).

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	$\bar{0}$ 4	$\bar{7}$ 8 9	$\bar{0}$ 4	0 1	0 1 5 9 4	0 1 5 9 9	$\bar{9}$ 1 2	$\bar{0}$ 4	$\bar{7}$ 8 9	$\bar{0}$ 4

Figure 43. Addition Without Mp of Mq Shift

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 rightmost digits of Mq equal to d are truncated as the shift is made (Figure 44). If d is less than zero and the absolute value of d

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	$\bar{0}$ 1	0 1	0 1 5 9 4	0 1 5 9 9	$\bar{2}$ 0 1	$\bar{0}$ 2	$\bar{7}$ 8 9	$\bar{0}$ 1

Figure 44. Mq Shifted Right to Align Decimal Points

is less than L, Mp is shifted |d| positions to the right before Mq is added to Mp. The number of rightmost digits of Mp equal to |d| are truncated as the shift is made. Eq replaces Ep (Figure 45). If d is plus and equal to or larger than L, Mp is above the range of Mq and no addition is performed (Figure 46). 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 47).

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	$\bar{0}$ 2	0 1	0 1 5 9 4	0 1 5 9 9	$\bar{8}$ 0 1	$\bar{0}$ 2	$\bar{7}$ 8 9	$\bar{0}$ 2

Figure 45. 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 46. 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 47. 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 rightmost digit of Mp is truncated, and the mantissa is shifted one position to the right. A one is entered in the leftmost position of the mantissa, and a one is added to Ep (Figure 48).

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 48. Mantissa Overflow, Number Normalized

When an overflow does not exist, Mp is scanned for zeros beginning with the leftmost 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 the Ep (Figure 49).

Timing. T = 10 (15 + 2.2L) average
10L Recomp. Time.

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 49. High-Order Zeros, Number Normalized

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.

Timing. Same as Floating Add.

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,

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

Figure 50. Product Equal To 2L

the field at 00099 — L replaces Mp (Figure 50). If the digit tested is a zero, the field at 00100 — L replaces Mp and Ep + Eq —1 replaces Ep (Figure 51).

Timing. $T = 10\,(28 + 3L + 4L_z + 4L\,(L - L_z))$
L_z = number of zeros in mantissa.

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	4 5 6	0 4	0 3	0 1 5 9 4	0 1 5 9 9	5 6 0	0 5	4 5 6	0 4

Figure 51. 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. The quotient at 00099 —L replaces Mp, and Ep — Eq + 1 replaces Ep (Figure 52).

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 52. Divisor Equal To or Less Than Dividend

If |Mp| is less than |Mq|, Mp is transmitted to 00099 — L; division starts in 00100 — L. The quotient at 00099 —L replaces Mp, and Ep — Eq replaces Ep (Figure 53).

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 53. Divisor Greater Than Dividend

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

Timing. T = 10 (34.5 + 27L + 9.75L²) average (assuming 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 rightmost digits. The P address is normally the units position of the mantissa; the digit at the Q address becomes the new rightmost digit of the mantissa. Vacated leftmost 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 leftmost digit of the Q field and terminates the operation (Figure 54).

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

Timing. T = 10 (7 + 2L).

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 54. Floating Shift Right

Floating Shift Left (FSL-05)

Description. The field at the Q address, which is the rightmost position of the mantissa, is shifted left so that the leftmost 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 rightmost position of the field moved and the P address is the leftmost 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 leftmost digit of the Q field (Figure 55).

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 $\bar{7}$ $\bar{8}$	$\bar{0}$ 5	0 5	0 1 5 9 5	0 1 5 9 7	$\bar{1}$ 2 3	$\bar{0}$ 2	$\bar{7}$ 8 0	$\bar{0}$ 5

Figure 55. 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 leftmost position and the new rightmost 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, this instruction is executed in the transmit record manner of transmitting individual digits in a left to right sequence. After the rightmost 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 leftmost 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 rightmost position of the fraction, and a negative zero is inserted in this position before the operation stops. Thus, a flag bit detected prior to the previous leftmost 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:

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

After transfer of the mantissa, but before the zero-fill operation, the core storage locations appear as follows (note that the flag bit in 01598 has been cleared):

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

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:

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

Timing. T = 10 (7 + 2L + 2L′)
L′ = length mantissa is increased by shift.

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 $\bar{0}0 \ldots 0\bar{9}\bar{9}$, which is the smallest positive quantity that can be represented (Figure 56). A zero mantissa causes the Equal/Zero indicator (12) to be turned on. Zeros entered as data should be in floating-point zero form. Zero quantities in other forms, e.g., $\bar{0}0 \ldots .0\bar{0}0$ 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 56. 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 High/Positive and Equal/Zero 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.

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

Equal/Zero (12). The Equal/Zero 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 ($\bar{9}9 \ldots 9\bar{9}9$) that can be represented. If the generated mantissa is positive, the H/P indicator (11) is also turned on.

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 ($\bar{0}0 \ldots 0\bar{9}\bar{9}$) 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 depression of the 1620 Reset key.

MARS DISPLAY SELECTOR (1620 CONSOLE)

Operand 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.

Index Registers Instructions

It becomes necessary in some 1620 programs to perform the same operations repetitively, with a change only in the P or Q address. The changing of an address while retaining the rest of the instruction is called *address modification.* Address modification can result in savings in the number of program steps and in the number of storage positions required. In some cases, the program itself determines if and how addresses are to be changed in order to perform the correct program steps for conditions that arise during the processing of data.

Modifying storage addresses, each time a repetitive operation is performed, requires several program steps and additional storage locations which must be set aside for this purpose.

When the Index Registers feature is installed, 14 index registers are provided in the IBM 1620 Processing Unit to modify addresses automatically. This means that fewer instructions are needed and storage space is conserved, thus providing for faster program execution and over-all simplification of programming effort.

METHOD OF ADDRESS MODIFICATION

The primary use of index registers is to modify addresses automatically by adding the contents of an index register to an address. Both the P address and the Q address can be modified by the contents of any index register. Only core storage addresses can be modified.

The modification of the P and Q addresses occurs in their respective address registers. For instance, if the P address is indexed, the indexing occurs in the OR-2 register. This means the original instruction in storage is in no way changed or modified.

Instructions are provided with this feature to load and modify these registers.

PROGRAMMING ADVANTAGES

The Index Registers feature offers many programming advantages, the greatest of which is the ability to modify instructions in a program loop. For example, without index registers, to add four fields to another field, it is easier (and as efficient) to use four separate add instructions rather than to program a loop to use one add instruction four times. With index registers, however, the operation can be programmed with one add instruction and one instruction to modify the index register.

When more than one instruction address in a program loop requires modification, an even greater savings in core storage and programming time can result with the use of an index register. For example, assume that core storage contains two sets of ten quantities; A1 through A10 and B1 through B10. The problem is to program a loop to add A1 to B1, A2 to B2, ...A10 to B10. Without index registers, and disregarding initialization, this requires a minimum of four instructions.

With the indexing feature, the 1620 Model 2 can be programmed to solve the above problem in two steps, one add instruction, and a combination compare-modify-branch instruction. (See the Branch Conditionally and Modify Index Register instruction.)

Two bands of 7 index registers are available. The core storage addresses for the 14 index registers are shown in Figure 57. An instruction is provided to select the seven index registers in Band 1, or the seven in Band 2, or to select the "no" Index Register mode.

Only one register of a band is used at a time. The individual register is selected by a combination of flag bits in the tens, hundreds, and thousands positions of

	Index Register	Core Storage Location
Band 1	1	00305 - 00309
	2	310 - 314
	3	315 - 319
	4	320 - 324
	5	325 - 329
	6	330 - 334
	7	335 - 339
Band 2	1	345 - 349
	2	350 - 354
	3	355 - 359
	4	360 - 364
	5	365 - 369
	6	370 - 374
	7	375 - 379

Figure 57. Index Registers Addresses

the P and/or Q addresses of the instruction to be modified. As shown in Table 7, an instruction address with a flag bit in the tens position specifies Index Register 1 (of the band previously selected). Likewise, an instruction address with flag bits in all three positions specifies Index Register 7 of the selected band.

An instruction address that contains an index register flag is said to be "indexed."

Table 7. Index Registers Identification

Address	IX
000$\bar{0}$0	1
00$\bar{0}$00	2
00$\bar{0}\bar{0}$0	3
0$\bar{0}$000	4
0$\bar{0}$0$\bar{0}$0	5
0$\bar{0}\bar{0}$00	6
0$\bar{0}\bar{0}\bar{0}$0	7

EFFECTIVE ADDRESS COMPUTATION

Computation of the effective address uses five digits from the index register regardless of any flags in the index register. Flags in the index register are not added to the effective address.

All indexed addresses are considered positive even if a flag is over the units position; the value in the index register, however, can be negative; and when the algebraic sum of the two is negative, the result is expressed in 10's complement form (Figure 58). An invalid effective address, such as that shown in Figure 58, will result in a MAR check.

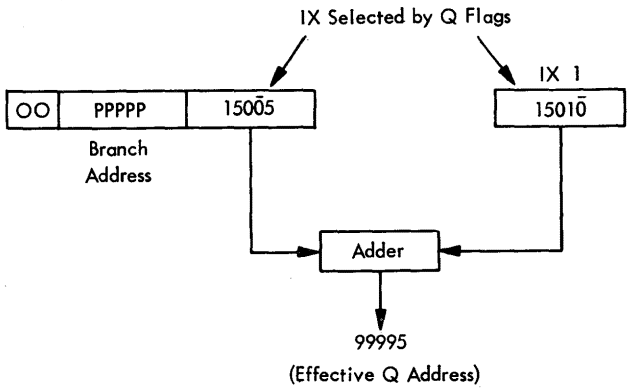


Figure 58. Indexing, Negative Result After Modification

NOTE: The computation of an effective address does not alter the condition of the High/Positive, Equal/Zero, or Overflow indicators.

INDEXING EXECUTION TIME

An additional 50 μsec are required during I-time for each address that is indexed.

INDEXING AND INDIRECT ADDRESSING

Any P or Q address that is a core storage location can be indexed. Any address that can be indexed can also be an indirect address. However, the Q address of the instructions used for indexing (operation codes 60-67) cannot have an indirect address. When an address is both indirect and indexed, the indexing takes precedence; the resulting effective address then becomes an indirect address. The address specified by the indirect address can also be indexed and/or indirect. This chain can continue until the final effective address is not an indirect address.

INDEX REGISTER MODIFICATION

The new contents of a modified index register is the algebraic sum of the former contents and the modifying amount (Figure 59). When a sign change occurs, the correct result is in the index register. Negative amounts are expressed in true form, for example, minus 196 as 0019$\bar{6}$.

When Band 0 has been selected (no index register mode) and an index register operation code (61-67) is given, the instruction is considered invalid and the computer stops. However, if an index register operation code (61-67) is given and no flags are present in the Q_8-Q_{10} positions of the instruction – in effect, no index register is specified – then the operation is performed, the computer does not stop, but of course, no index register is acted upon. It should be noted that in an error of this type, the data from the Q address is actually placed into core storage as follows:

Band 1 selected	positions 00300- 00304
Band 2 selected	positions 00340- 00344

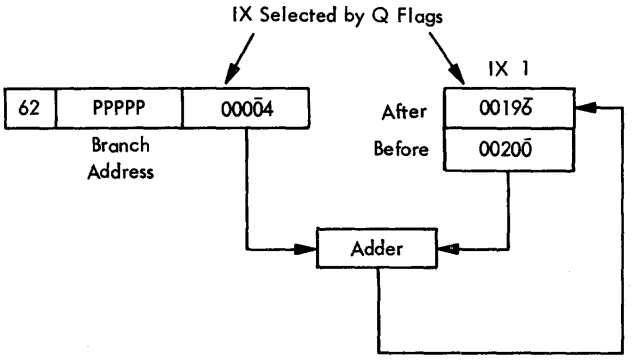


Figure 59. Index Register Modification

These two core storage locations can actually be considered as Index Registers "0." They can be modified and tested or used as a counter; however they *cannot be used for* address modification.

PROGRAM INDICATORS

Three additional indicators are provided with the Index Registers feature. These indicators provide the ability to determine, by programming, which band of index registers is selected. The condition of the indicators can be tested by the Branch Indicator or Branch No Indicator instructions. The indicators are not affected by the testing.

The indicators as shown in Table 8 are considered to be ON when the condition shown exists.

Table 8. IX Band Program Indicator

Indicator	Condition
30	Neither Band Selected
31	Band 1 Selected
32	Band 2 Selected

INDEX REGISTER INSTRUCTIONS

Eight additional instructions are provided with the Index Registers feature: seven for loading, storing, and modifying, and one Move Address instruction. Provision is made through the Branch and Select instruction to select Band 1, Band 2, or "no band."

The Q addresses of the new instructions either specify the location of data or in immediate instructions, contain data. Flag bits are used in the Q_8, Q_9, and Q_{10} positions to specify an index register. (Refer to Table 7.) The P address is a branch address.

Five of the instructions are unconditional branch instructions; after index register, loading, storing, or modifying, the computer branches to the P address. Two instructions are of the conditional branch type; after modification of an index register, a branch is initiated only if the contents of the modified index register are not zero or have not changed sign.

Branch and Select (BS-60)

Description. The Branch and Select instruction provides for IX band selection or *no band* selection. The selected band remains selected until another Branch and Select instruction is executed or until power is removed from the system. The Reset key has no effect on band selection. The "no band" mode is selected automatically when power is applied to the system. The Q_7 - Q_{10} positions of the instruction are not used. The selection desired is indicated by the Q_{11} digit as shown below:

0 — IX Band 0 (no band)

1 — IX Band 1 (Index Registers 1-7)

2 — IX Band 2 (Index Registers 1-7)

The P address specifies the next instruction to be executed. This instruction is also used to turn the indirect addressing feature on and off; this function is described under Logic Instructions.

Timing. T = 60.

Branch and Modify Index Register (BX-61)

Description. This instruction is used for index register modification. The field designated by the Q address is added to the selected register. The index register is selected by flags in the Q_8 - Q_{10} positions of the instruction (Figure 60).

The field to be added to the register is *not* limited to five digits; *thus, one or more registers can be modified with one instruction.* For example, to modify index registers 4, 5, and 6, the length of the Q field must be 15 positions and the Q_8 - Q_{10} positions of the Q address would be flagged for index register 6; however, prior to this instruction the flags in index registers 4, 5, and 6 would have to be removed. (The leftmost flag in register 4 would not have to be removed.) The length of the index register field is determined by the leftmost flag bit in that field; the leftmost flag bit in the Q field defines its length. As in a normal add operation, the number of positions added depends upon the length of Q field and of the index register field(s).

The High/Positive or Equal/Zero indicator is set according to the results of the modification. The Overflow Check indicator is turned on *only* if the Q field is longer than the index register field(s). An overflow that occurs as a result of a "carry" out of the leftmost position of the index register field(s) does *NOT* turn on the Overflow indicator and the overflow digit is lost. The P address specifies the next instruction to be executed.

Timing. $T = 10\ (6.5 + .5\ D_Q + L_X)$

L_X = length of index register field(s).

Branch and Modify Index Register Immediate (BXM-62)

Description. This instruction is similar to the Branch and Modify Index Register instruction except that *the*

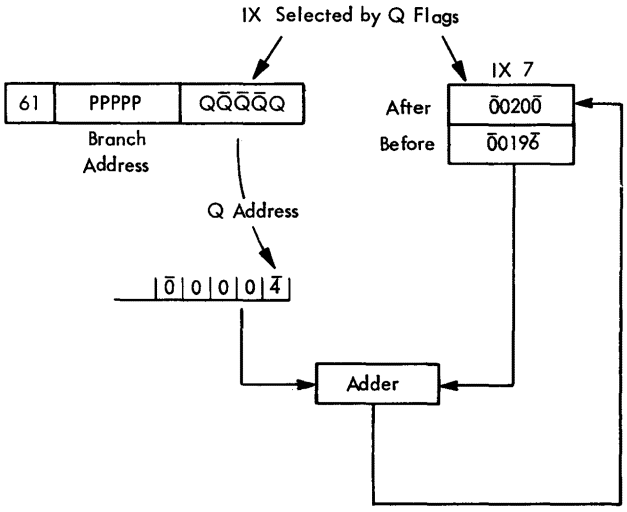


Figure 60. Branch and Modify Index Register

five digits in the Q portion of the instruction are used as the modifier (Figure 61). Flags in the Q field specify the index register to be modified. A flag in Q_{11} indicates a negative modifier.

Timing. T = 140.

Branch Conditionally and Modify Index Register (BCX-63)

Description. This instruction is similar to the Branch and Modify Index Register instruction except that a branch to the P address is conditional. The P address specifies the next instruction to be executed if (after modification) the sign of the index register has not changed or the result is not zero. (If more than one index register is modified, only the sign of the rightmost register is considered.) If the sign has changed or the result is zero, the next sequential instruction is executed. If an overflow would result from a carry out of the leftmost position of the index register field, the next sequential instruction is executed (Table 9).

Timing. T = 10 (6.5 + $5D_Q$ + L_X)
L_X = length of index register fields (s).

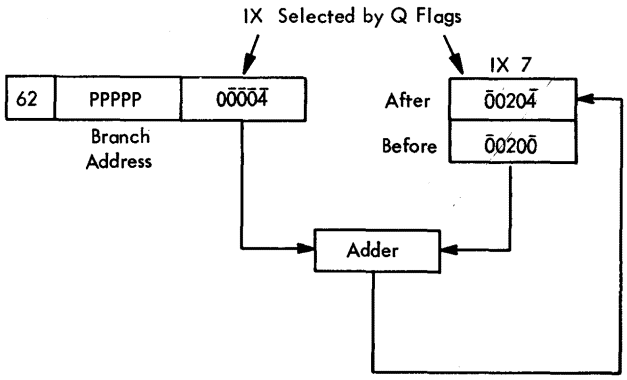


Figure 61. Branch and Modify Index Register Immediate

Table 9. Conditional Branch Examples

Modifier	IX (before)	IX (after)	Branch
00005	00015	00010	Yes
00010	00010	00000	No
00010	00008	00002	No
00005	00025	00020	Yes
00020	00020	00000	No
00010	00005	00005	No
00006	99995	00001	No
00040	99970	99930	Yes
00035	99985	00020	No

Branch Conditionally and Modify Index Register Immediate (BCXM-64)

Description. This instruction is similar in operation to the Branch Conditionally and Modify Index Register instruction except that *the five digits* in the Q portion of the instruction are used as the modifier (Figure 61). Flags in the Q field specify the index register to be modified. A flag in Q_{11} indicates a negative modifier.

Upon completion of the index register modification, a branch to the P address is conditional as described for the BCX instruction.

Timing. T = 140.

Branch and Load Index Register (BLX-65)

Description. This instruction loads the five digits of data specified by the Q address to the selected index register. The flags in Q_8 - Q_{10} positions select the appropriate register.

A flag bit in the units position of the Q field is transferred to the index register and a flag is automatically placed over the high-order position of the index register field. No other flags are transferred (Figure 62).

The P address specifies the next instruction to be executed.

Timing. T = 140.

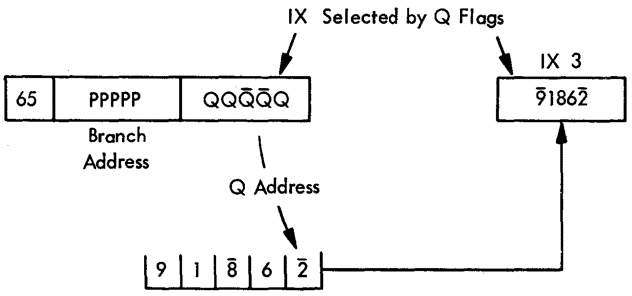


Figure 62. Branch and Load Index Register

Branch and Load Index Register Immediate (BLXM-66)

Description. This instruction is similar to the Branch and Load Index Register instruction except that the Q portion of the instruction is used as the data.

Timing. T = 140.

Branch and Store Index Register (BSX-67)

Description. This instruction stores the five digits of the selected index register at the Q address (Figure 63). If the contents of the index register are negative, a flag is stored with the units position. A flag is automatically placed in the leftmost position of the stored data. No other flags are transmitted.

Timing. T = 140.

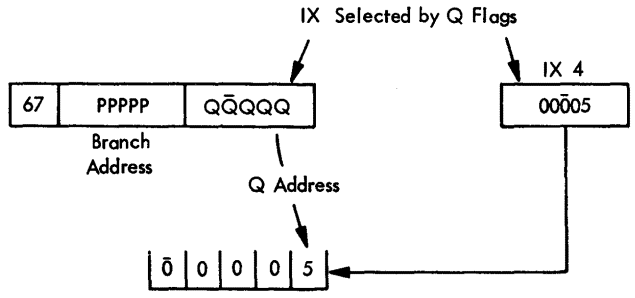


Figure 63. Branch and Store Index Register

Move Address (MA-70)

Description. The Move Address instruction causes the five digits specified by the Q address of the instruction to be moved to the P address (Figure 64). Flags in the P field remain unchanged.

Timing. T = 140.

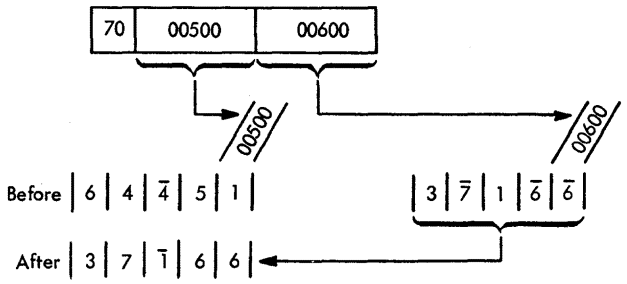


Figure 64. Move Address

Binary Capabilities Instructions

Ten instructions are provided with the Binary Capabilities feature. The H/P and E/Z indicators are set as in other arithmetic instructions. Octal fields have the same flexibility and restrictions that exist for decimal fields. Generally, fields must be at least two digits in length; a flag is used to specify the high-order digit of the field. A brief description of the characteristics of the decimal, binary, and octal number system is provided before the instruction descriptions.

Number Systems

DECIMAL

The decimal number 1375, for example, can be expressed as $1(10^3) + 3(10^2) + 7(10^1) + 5(10^0)$ which equals 1000 + 300 + 70 + 5. (Any number with a zero exponent equals one.)

BINARY

Binary data consists of two digits, zero and one. Thus, the decimal digits 0 through 9 are expressed in binary form, as follows:

DECIMAL	BINARY	DECIMAL	BINARY
0	0000	5	0101
1	0001	6	0110
2	0010	7	0111
3	0011	8	1000
4	0100	9	1001

Note that a one in the rightmost position of the binary number is equivalent to a decimal one; a one in the second position is equivalent to a decimal 2; a one in the third position, to a 4; a one in the fourth position, an 8; if there was a fifth position, a one in that position would be equivalent to 16; a one in the sixth, 32, and so on. The decimal equivalent of each position is twice that of the position to its right.

OCTAL

Octal data is expressed to the base 8 just as decimal and binary data are expressed to the base 10 and 2.

Thus, equivalent octal numbers progress as follows:

DECIMAL	OCTAL	DECIMAL	OCTAL
1	1	17	21
2	2	23	27
7	7	24	30
8	10	31	37
9	11	32	40
15	17	39	47
16	20	40	50

The octal number 2537, for example, is equal to the decimal number 1375; this relationship is explained in the following section.

Numbers, in systems other than decimal, are identified by subscripts to indicate the number system they represent. Thus, the preceding octal number 2537 is expressed as 2537_8.

Numbers without subscripts are assumed to be decimal.

Number Conversion

The Read Binary instruction of the Binary Capabilities feature reads binary data from the 1621. The binary data is stored in octal form as it enters core storage. Although this conversion is done automatically, a brief comparison of the two forms is given to show their relationship.

BINARY TO OCTAL CONVERSION

The first three positions of the binary number system have a maximum decimal equivalent of 7 ($111_2 = 7$). Because seven is the highest number in the octal system, a binary number can be separated in 3-position groups – each group representing the equivalent octal number, for example, 010101011111_2 can be separated as follows:

$$\underbrace{010}_{2}\ \underbrace{101}_{5}\ \underbrace{011}_{3}\ \underbrace{111}_{7}$$

Thus, $010101011111_2 = 2537_8$

Octal to binary conversion is, of course, simply the reverse of binary to octal.

OCTAL TO DECIMAL CONVERSION

Octal numbers can be converted to decimal numbers by expanding each position, as follows:

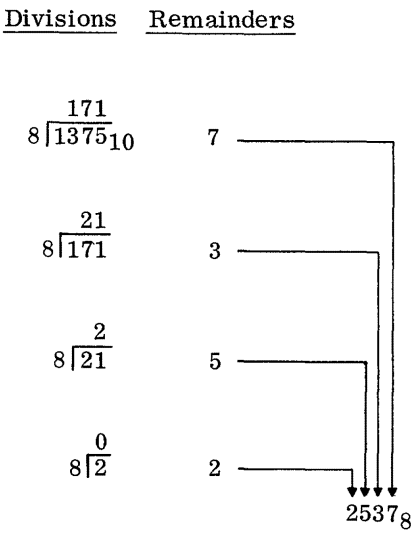
2537_8 equals $2(8^3) + 5(8^2) + 3(8^1) + 7(8^0)$

$2(512) + 5(64) + 3(8) + 7(1)$.

$2537_8 = 1375_{10}$

DECIMAL TO OCTAL CONVERSION

Decimal numbers can be converted to octal numbers by succesive divisions by eight until a zero quotient is obtained. The remainders, in the reverse order, form the octal number. For example, 1375_{10} is converted to 2537_8 as follows:



Branch On Bit (BBT-90)

Description. The Q_8 through Q_{11} digits of the instruction specify a four-digit core storage address (0000-9999). The bit configuration of the digit at this address is compared with the bit configuration of the Q_7 digit of the instruction. If any bit of the Q_7 digit corresponds to any bit in the digit specified by Q_8 - Q_{11} (the C bit is excluded from this comparison), the 1620 branches to the P address. If no successful comparison occurs, the next instruction in sequence is executed. Both the P address and the Q_8 - Q_{11} address may be indirect addresses. The Q_8 - Q_{11} address is restricted to the first 10,000 positions of core storage. An indirect Q_8 - Q_{11} address, however, makes possible the placement of the digit at any core storage address.

Figure 65 shows that the digit at 00500 is compared with the Q_7 digit. Since there is no 1-bit in the data at 00500, the branch does not occur.

Timing. T = 70.

Branch on Mask (BMK-91)

Description. The Q_8 through Q_{11} digits of the instruction specify a four-digit core storage address (0000-9999). The bit configuration of the digit at this address is compared with the bit configuration of the Q_7 digit of the instruction. A branch to the P address occurs when the 8, 4, 2, and 1 bits that make up the Q_7

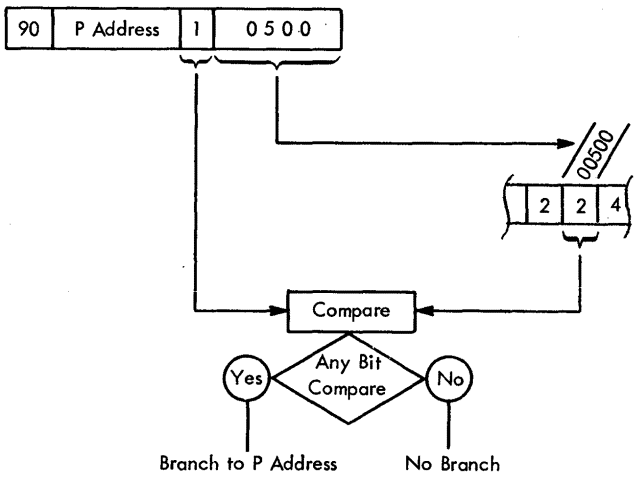


Figure 65. Branch on Bit

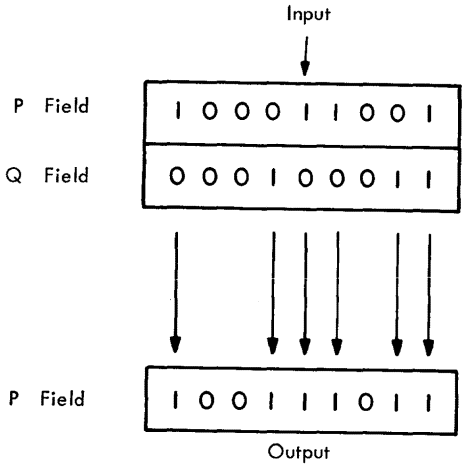


Figure 66. OR Logic

digit are also present in the digit specified by the Q_8 - Q_{11} address. The flag bit is considered in the comparison only if the digit at Q_7 contains a flag bit. If the Q_7 digit does not contain a flag bit, the presence of a flag bit in the data at Q_8 - Q_{11} is of no consequence. If the comparison is not successful, the next instruction in sequence is executed.

Although the Q_8 - Q_{11} address is restricted to the first 10,000 positions of core storage, an indirect Q_8 - Q_{11} address makes possible the placement of the digit at any core storage address. Both the P address and the Q_8 - Q_{11} address may be indirect addresses.

Timing. T = 70.

OR To Field (ORF-92)

Description. The OR logic consists of two inputs and a resulting output based upon the presence or absence of input bits as shown in Figure 66. The OR logic provides an output bit if either one of the inputs contains a bit.

The 4, 2, and 1 bits of each digit of the P field and the 4, 2, and 1 bits of the corresponding digits of the Q field become successive OR logic inputs. C bits and 8 bits are ignored. The resulting output replaces the P field data, as shown in Figure 67. C bits are added where required for correct parity. Flag bits in the P field are retained. Any 8 bits in the P field are destroyed. The first flag bit in the Q field terminates the operation. Q field data is not altered.

Figure 68 shows data flow for the instruction 92 00500 00600. The resulting data 3767 replaces the original P data 9127. The flag bit in the tens position of the P field is not disturbed. The Equal/Zero indicator is turned on if the resultant field has no numerical bits.

Timing. T = 10 (6 + 2 D_Q).

AND to Field (ANDF-93)

Description. The AND logic also consists of two inputs and a resulting output based on the presence or absence of input bits. The difference between AND and OR logic is that OR logic provides an output bit if either one of the inputs contains a bit; whereas, AND logic provides an output bit only when *both* inputs contain a bit, as illustrated in Figure 69.

The AND to Field instruction operates in the same manner as the OR to Field instruction except that AND logic is used in place of OR logic. The 4, 2, and 1 bits of

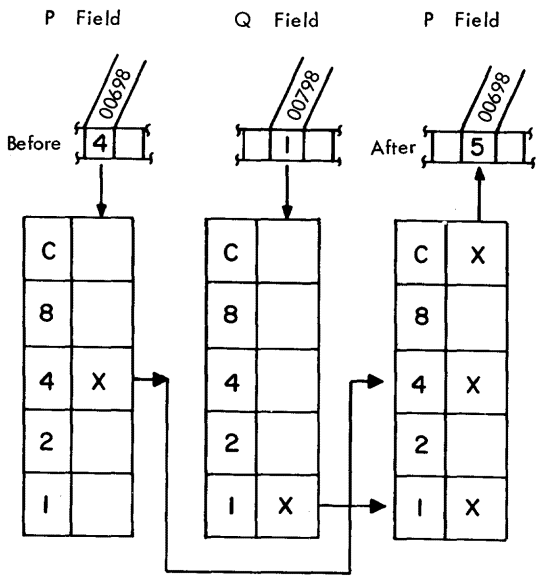


Figure 67. OR To Field, One Digit

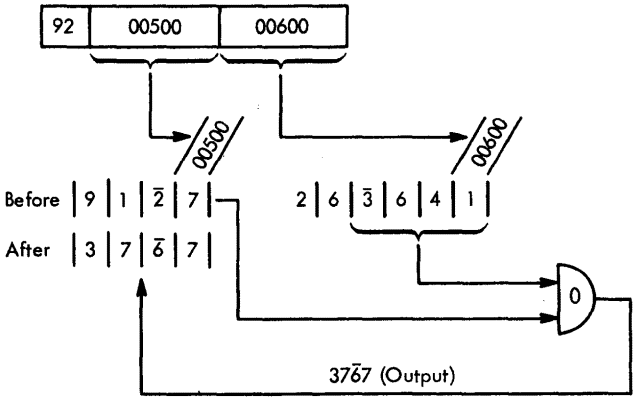


Figure 68. OR To Field – Data Flow

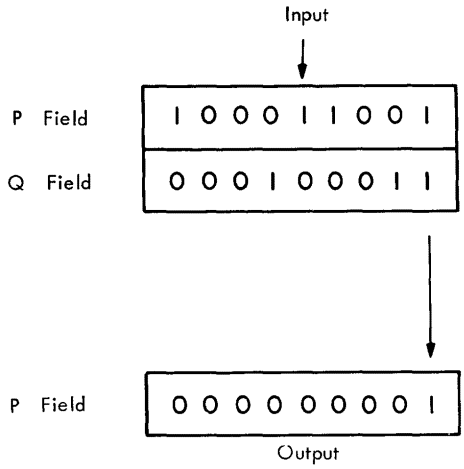


Figure 69. AND Logic

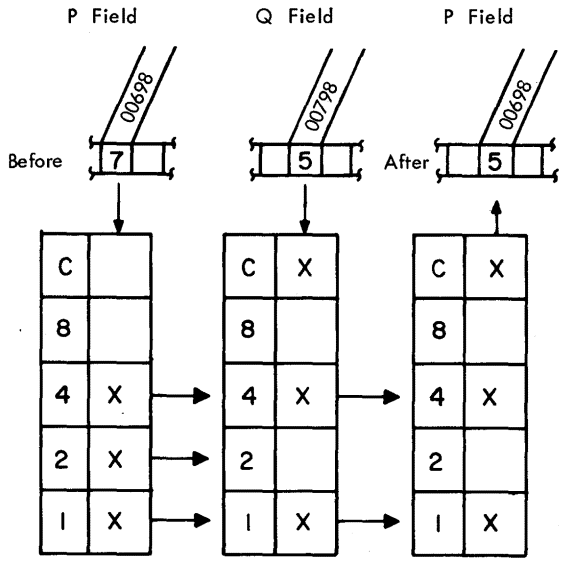


Figure 70. AND To Field, One Digit

each digit of the P field and the 4, 2, and 1 bits of the corresponding digits of the Q field become successive AND logic inputs as illustrated in Figure 70.

For example, using the same data as that used in Figure 68, the resulting output is 1001, as shown in Figure 71. The Equal/Zero indicator is turned on if the resultant field has no numerical bits.

Timing. $T = 10\ (6 + 2\ D_Q)$.

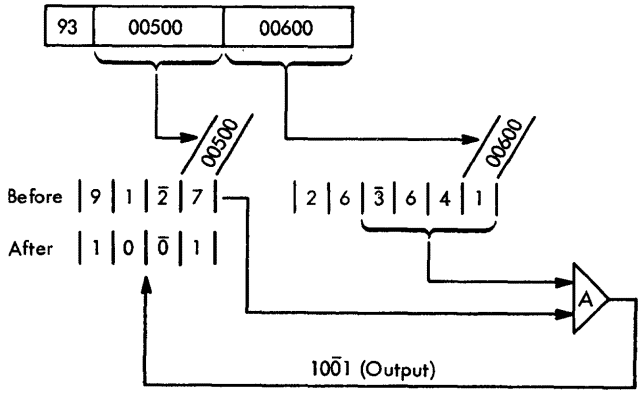


Figure 71. AND To Field – Data Flow

Exclusive OR to Field (EORF-95)

Description. The operation of the Exclusive OR to Field instruction is similar to that of the OR to Field instruction. The only difference is that the octal data in the P and Q fields is exclusive ORed – that is, the Exclusive OR logic provides an output bit if either one of the inputs contains a bit, but *NOT* if both inputs contain a bit, as illustrated in Figure 72.

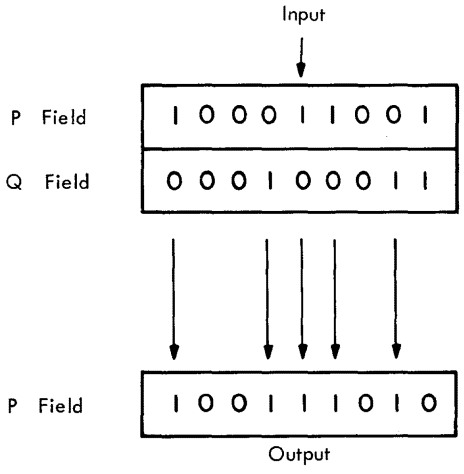


Figure 72. Exclusive OR Logic

The 4, 2, and 1 bits of each digit of the P field and the 4, 2, and 1 bits of the corresponding digits of the Q field become successive Exclusive OR logic inputs as illustrated in Figure 73.

The Equal/Zero indicator is turned on if the resultant field has no numerical bits.

Timing. $T = 10\ (6 + 2\ D_Q)$.

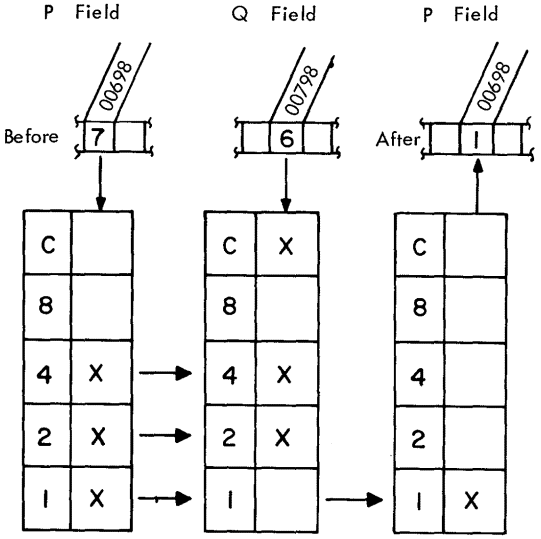


Figure 73. Exclusive OR to Field, One Digit

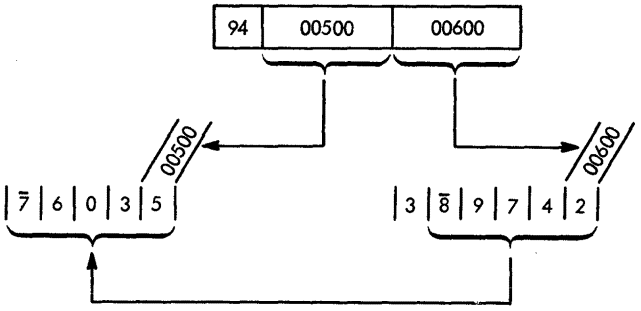


Figure 74. Complement Octal Field

Table 10. Complementing Octal

Q Data	8	9	7	4	2
Bit configuration of each Q digit	1000	1001	0111	0100	0010
Q Data (Octal)	0	1	7	4	2
Bit configuration after complementing (8 bits disregarded)	0111	0110	0000	0011	0101
Resulting P Data (Octal)	7	6	0	3	5

Complement Octal Field (CPLF-94)

Description. The Q data is complemented on an octal basis and transmitted to the P field. Conversion and transmission are terminated by the first flag bit detected in the Q data field even if a flag bit is present in the units position. The flag in the Q field is transmitted.

Since the octal system has no digit higher than 7, only the 4, 2, and 1 bits of each digit are complemented and transmitted. The 8 bits are neither complemented nor transmitted. The C bit is added to each complemented digit when required for parity. For example, a 4 digit is complemented to a 3 digit, and a C bit is added for parity. All replaced P data is lost. The Equal/Zero indicator is turned on if the resultant field has no numerical bits.

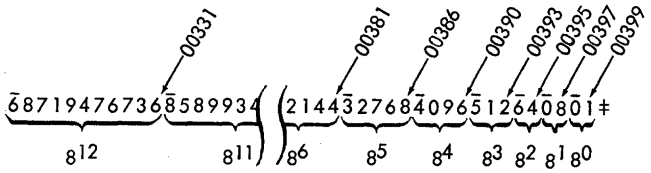
Figure 74 shows that Q data at 00600 is complemented to 76035 and stored at the P address. The original Q data remains at the Q address. Table 10 shows how the numerical bits of each digit are complemented. Although the 8 bit is shown with the 4, 2, and 1 bits of each digit, it, like the C and F bits, is not complemented. Note however, that the 8, 4, 2, and 1 bits actually form the binary representation of the decimal digit.

Timing. $T = 10\ (6 + 2\ D_Q)$.

Octal to Decimal Conversion (OTD-96)

Description. The principle of octal to decimal conversion has been previously explained. The 1620 accomplishes this conversion by using a table of numbers to the base eight. (A table of any number system less than or equal to nine can be used to convert numbers to base ten.)

If for example, the octal fields to be converted do not exceed 13 digits in length, the table need not contain any powers of eight higher than 12, as follows:



From the table it can be seen that 8^{12} equals 68, 719, $476{,}736_{10}$ which equals 1000000000000_8 (13 digits). The larger the size of the octal field, the higher the power of eight number that must be stored. A 14-digit octal number, for example, would require that the

table be expanded to include 8^{13}. The table may be stored anywhere in core storage.

The core storage address of the units digit of the table must be known to the programmer. The P address of the Octal to Decimal Conversion instruction specifies this address; in the preceding example the address is 00399.

The Q address specifies the address of the units digit of the octal field to be converted. A flag bit determines the length of the Q field.

The decimal equivalent of the octal number is developed from the power-of-eight table, as follows: Recall the principle of octal to decimal conversion, where $2537_8 = 2(8^3) + 5(8^2) + 3(8^1) + 7(8^0)$. Note that each digit of the octal number correlates with a power-of-eight number – the units digit with the 8^0, the tens digit with 8^1, the hundreds digit with 8^2, etc. The Octal to Decimal Conversion instruction is accomplished by multiplying each octal digit by its corresponding power of eight number, as illustrated in Figure 75. The multiply table is used and the products of these multiplications are developed and summed in the product area. The units digit and sign of the converted number is located at 00099; the leftmost digit is automatically flagged.

Before the product was formed, the Product Area (00080 through 00099) was cleared to zeros. If the results of the octal to decimal conversion will exceed these 20 positions, the positions in excess of 20 (00079, 00078, etc.) must be cleared to zeros before this instruction is executed.

Timing. $= 10\ (28 + D_Q\ (2D_Q - 1))$

Decimal to Octal Conversion (DTO-97)

Description. The principle of decimal to octal conversion has been previously explained. The 1620 accomplishes decimal to octal conversion in a similar manner, as follows:

The P address of this instruction specifies the address where the leftmost digit of the resultant octal number is to be stored.

$2 \times (8^3)$	=	2×512	=	1024
$5 \times (8^2)$	=	5×64	=	320
$3 \times (8^1)$	=	3×8	=	24
$7 \times (8^0)$	=	7×1	=	7
				1375

Figure 75. Octal to Decimal Conversion

The Q address specifies the address of the units digit of the power-of-eight number to be used in the first subtraction. Recall that decimal to octal conversion is a series of divisions. Division is accomplished in the 1620 by successive subtractions The power-of-eight number that must be addressed is one less than the number of digits that will be in the resultant octal field. If the octal number is to be five digits long, the fifth table entry (8^4 or $\bar{4}096$) is addressed. The programmer must know the size of the octal field that will be developed and the core storage address of the units digit of each power-of-eight number. If the 8^0 number, 01, is stored at 00399, the 8^1 units digit would be stored at 00397, the 8^2 units digit at 00395, etc.

The decimal field to be converted must be located at 00099 before this instruction is executed. The divisor, specified by the Q address, is successively subtracted from the decimal number. Because the Q address specifies a divisor whose value is in proximity to the value of the dividend, the remainder has no significance and no more than seven successful subtractions should occur. An eighth successful subtraction would turn on the Overflow indicator. The quotient digit is developed in the Multiplier/Quotient register and automatically transferred to the P address. The first quotient digit, which is the leftmost digit of the octal number, is automatically flagged. The next power-of-eight number is then subtracted from the remaining dividend and the second quotient digit is developed and transferred to the P address plus one. This series of divisions of power-of-eight numbers into remaining dividends continues until the last entry (8^0) has been used as a divisor. The last entry of the table is defined by a record mark to the right. Figure 76 shows how the decimal number $\bar{9}9999$ is converted to the octal number $\bar{3}03237$. The instruction 97 00500 00386 causes the 8^5 number to be subtracted from the decimal number. Three successful subtractions occur and the quotient digit 3 is stored at the P address 00500. The 8^4 number cannot be successfully subtracted from the remaining dividend (overdraws and corrections are not shown) and a zero is transferred to 00501. The 8^3 number is successfully subtracted three times, and a 3 is transferred to 00502. Two successful subtractions occur with the 8^2 number and a two is transferred to 00503. The 8^1 number and the 8^0 number are successfully subtracted 3 and 7 times, respectively. The detection of the record mark at 00400 stops the instruction execution. The developed quotient digits form the octal number $\bar{3}03237$ at core storage locations 00500-00505

Timing. $T = 10\ (31 + T_Q + 4.125\ T_Q\ (T_Q + 1))$

T_Q = Position number of table addressed

average octal digit = 3.5

Instruction: 97 00500 00386		
Decimal number at 00095-00099	$\bar{9}9999$	Stored octal digits
8^5 number at 00382-00386	$\bar{3}2768$	
First subtraction	67231	$\bar{1}$
	32768	
Second subtraction	34463	$\bar{2}$
	32768	
Third subtraction	01695	$\bar{3}$
8^4 number at 00387-00390	$\bar{4}096$	
No successful subtraction	01695	$\bar{3}0$
8^3 number at 00391-00393	512	
First subtraction	01183	$\bar{3}01$
	512	
Second subtraction	00671	$\bar{3}02$
	512	
Third subtraction	00159	$\bar{3}03$
8^2 number at 00394-00395	64	
First subtraction	00095	$\bar{3}031$
	64	
Second subtraction	00031	$\bar{3}032$
8^1 number at 00396-00397	08	
After three subtractions	00007	$\bar{3}0323$
8^0 number at 00398-00399	01	
After seven subtractions	00000	$\bar{3}03237$

Figure 76. Decimal to Octal Conversion

Read Binary Paper Tape (RBPT-37, $Q_{8,9}$-33)

Description. The Read Binary Paper Tape instruction operates in a manner similar to the Read Alphamerically (paper tape) instruction. Both instructions translate a single column to the 1620 two-digit form. The difference is that this instruction transfers the 8, O, and X tracks to even-numbered core storage locations as 1, 2, and 4 bits, respectively. The 1, 2, and 4 paper tape tracks are entered as 1, 2, and 4 bits into odd-numbered core storage positions. C bits are added to the core storage positions wherever necessary to maintain correct parity.

The P address of this instruction must refer to an odd-numbered storage location. Reading continues until an End-of-Line (EOL) character is interpreted. Note that the tape feed code is converted to 77 in core storage as shown in Figure 77. A tape character containing punches in all eight channels is treated as a tape feed code.

Timing. Time depends upon the size of the input record. Speed of the Paper Tape Reader is 150 cps.

Write Binary Paper Tape (WBPT-39, $Q_{8,9}$-32)

Description. The Write Binary Paper Tape instruction causes the 1, 2, and 4 bits of two adjacent core storage positions to be punched into one column of paper tape. The 1, 2, and 4 bits of odd-numbered core storage positions are punched into the 1, 2, and 4 tracks, respectively. The 1, 2, and 4 bits of even-numbered core storage positions are punched into the 8, O, and X tracks, respectively.

The P address of this instruction must refer to an odd-numbered position of core storage. Writing (punching) continues until an alphameric record mark (0=) character is encountered in core storage. The record mark is translated to an EOL punch in paper tape.

Timing. Time depends upon size of output record. Speed of the Tape Punch is 15 cps.

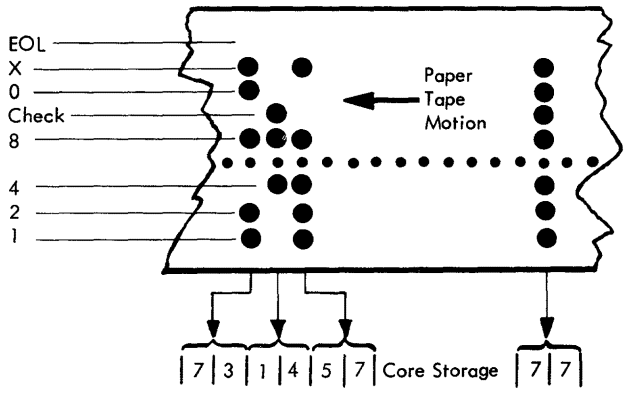


Figure 77. Read Binary Translation

The 1620 console (Figure 78) 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

Machine Status and Program Switches (Figure 79)

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 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.

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

Disk Storage Indicators

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

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 from the Disk Control Field.

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.

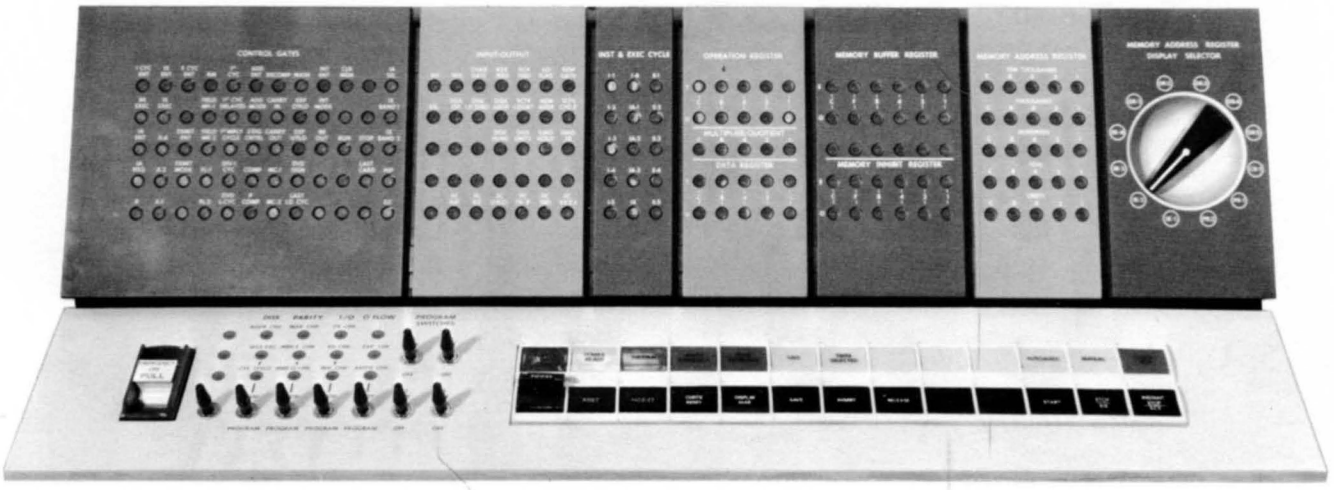


Figure 78. 1620 Model 2 Console

WLR/RBC (Wrong-Length Record/Read-Back Check) Indicator. This dual-purpose indicator and light are 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.

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 Indicator. This light and indicator are turned on when the digit in the even address portion of the MBR or MIR 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 Indicator. This light and indicator are turned on when the digit in the odd address portion of the MBR or MIR 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 Indicator. This light is turned on when a digit in MARS

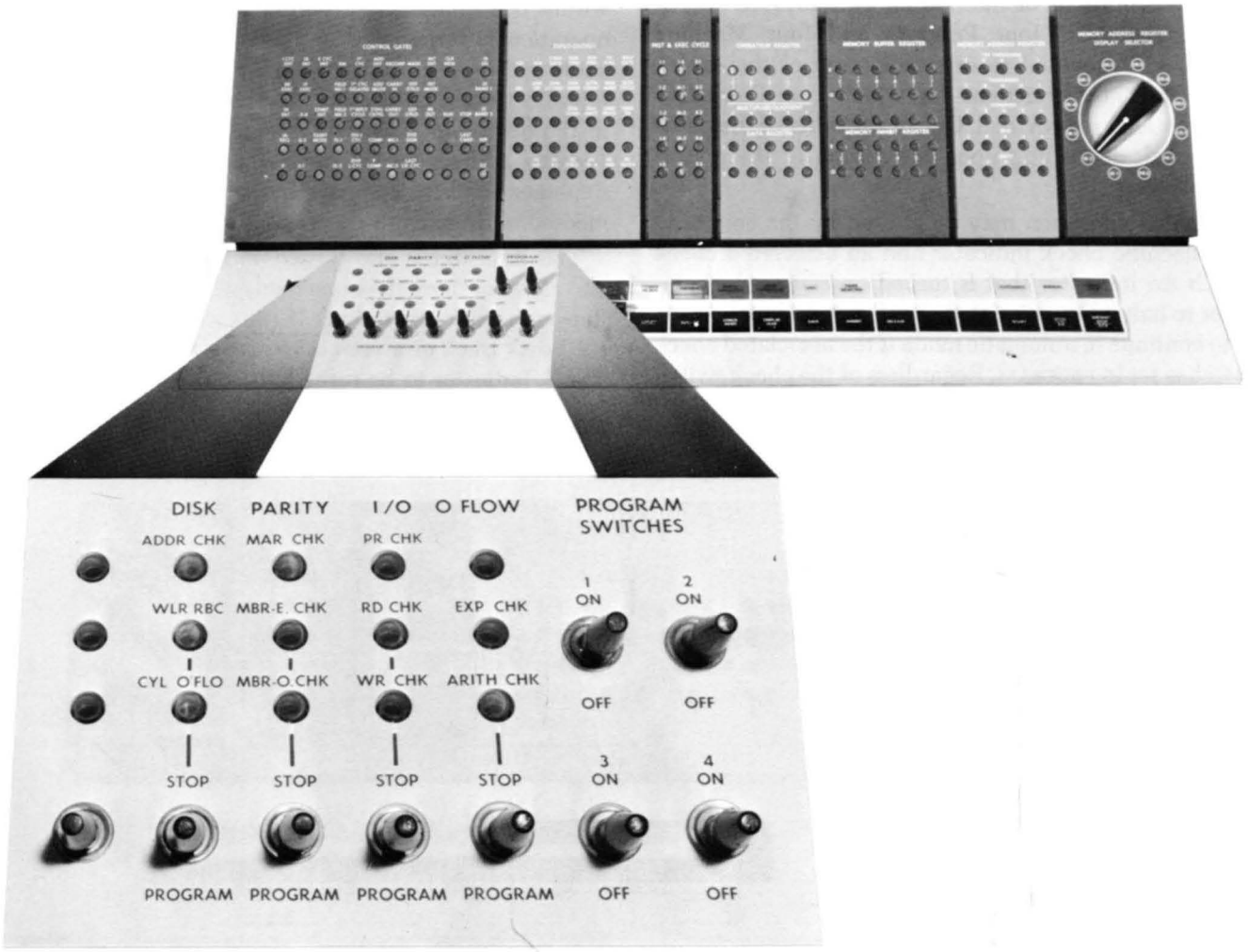


Figure 79. Indicator Lights and Switches

has a parity error or an invalid address. *These errors halt the machine immediately* The setting of the Parity Check switch has no effect upon this indicator.

Input/Output (I/O) Check Indicators

PR (Printer) Check. This indicator and light are turned on by the 1443 Printer when an output character with an even number of bits is detected while entering or leaving the print buffer. It is also turned on by a Sync Check. If the I/O Check switch is set to STOP the 1620 halts *after* the line is printed. If the I/O Check switch is set to PROGRAM and a Sync Check error is detected, the 1620 continues processing but the 1443 stops after the line is printed.

RD CHK (Read Check) Indicator. 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) Indicator. 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 11.

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 depressed, 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 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 depressed, the Arithmetic Check indicator remains on, and the computer continues to execute instructions in the automatic mode until another overflow occurs.

Table 11. Write Check Errors

Output Unit	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.
Printer	1620 halts after the line is printed	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.

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 in the program occurs when a switch specified by a Branch Indicator instruction has been set to ON by the console operator.

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.

Control Switches, Keys, and Signal Lights

Control keys (Figure 80) 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.

The manual and automatic lights are described first because of the effect of the manual and automatic modes of operation upon other keys.

Automatic and Manual Lights. When the Manual light is on and the Automatic light is off, 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 mode, that is, while executing a stored program or while transferring data to or from an input/output unit.

Manual mode is initiated and the Manual light is turned on by the execution of a Halt instruction or by pressing the Stop key or by pressing the Release key (on an I/O operation only). Pressing the Start key, Insert key, or Display MAR key initiates automatic mode and turns off the Manual light. The Save light, Read Interlock light, or Write Interlock 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; however, the computer is in *automatic mode* until the instruction has been single-cycled to completion. When the instruction is completed the Manual light remains on and the Automatic light is turned off.

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. Turning power on turns the Mask, IA (Indirect Addressing) and IX Band

Figure 80. 1620-2 Control Keys and Signal Lights

0 indicators on and turns the Interrupt, IX Band 1, and IX Band 2 indicators off.

Reset Key. The Reset key is used to restore machine status indicators, machine check indicators, and signal lights to their initial or reset condition and to mask all interrupts (in the 1710 Control System). This key also resets the *Parity and I/O indicators.* The Reset key is operative only in the manual mode When the computer is in the automatic mode, the Stop/SIE key should be pressed to put the computer in the manual mode and permit use of the Reset key.

Modify Key. This key is used in conjunction with the Check Reset key to reset core storage to zeros:

1. Press the Modify and Check Reset keys *simultaneously.* This sets up the circuitry required to clear core storage.
2. Press the Start key. Core storage is set to zeros.

If the Modify and Check Reset key are pressed simultaneously *in error,* pressing the Reset key nullifies the clear-storage circuitry that was set up. This key is operative only in the manual mode.

Check Reset Key. Pressing this key resets the following indicators: MBR-O, MBR-E, MAR, WR CHK, RD CHK, and PR CHK. It is operative when the computer is in either automatic or manual mode.

Display MAR. This key is used to display one of the twelve MARS registers in the following manner:

1. The MARS Display Selector switch is set to the register to be displayed.
2. The Display MAR key is pressed. The contents of the selected register are placed into the MAR register and displayed in MAR display lights

The selector switch should not be turned while the Display MAR key is pressed. This key is operative only when the computer is in the manual mode.

Save Key and Save Light. Pressing this key turns on the Save light and transfers the address in Instruction Register 1 (IR-1) to Product Address Register 1 (PR-1). The next Branch Back instruction executed in the program causes the Save light to be turned off and the address in PR-1 to be transferred back to IR-1.

If the Reset key is pressed before the Branch Back instruction is executed, the address in PR-1 is lost. The Check Reset key does not effect the Save key operation. If a multiply operation is performed before the Branch Back instruction is used, the saved address is lost because the contents of PR-1 are decremented for each new multiply digit during a multiply operation.

This key is operative only when the computer is in the manual mode.

Insert Key and TWPR SEL Light. Depression of the Insert key places the 1620 in automatic mode, turns on the TWPR (Typewriter) Select light and activates the typewriter keyboard so that direct entry of up to 100 characters may be made in numeric mode, starting at address 00000 and continuing into higher-numbered storage positions. After the 100th character has been entered the computer is returned to manual mode. The Insert key is operative only when the computer is in the manual mode. The Insert key on the typewriter has the same function.

The TWPR SEL light is turned on any time the console typewriter is selected or performing an input/output operation

Release Key. The Release key is used to terminate any input/output operation, including console keyboard entry of data into core storage. When this key is pressed, manual mode is initiated, the Manual light is turned on and, if it is on, the TWPR SEL light is turned off.

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

Start Key. The Start key is used to start program processing and to put the computer in automatic mode. Pressing the Start key turns on the Automatic light and turns off the Manual light. It is operative only when the computer is in manual mode.

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

The Stop/SIE key also serves as a Single Instruction Execute key. Successive depressions of the key cause one instruction to be executed for each depression. The Manual light remains on.

This key should be used to stop the computer when it is necessary to perform manual operations.

Instant Stop/SCE (Single Cycle Execute) Key. Depression of the Instant Stop/SCE key causes the machine to stop at the end of the 10-μsec machine cycle in progress when the key is depressed. Successive depressions of the key cause single machine cycles. Both the Manual and Automatic lights remain on.

When the key is pressed, the computer is stopped in the automatic mode (unless the key is pressed at the instant between instructions, which is very unlikely) and *remains in the automatic mode* until the instruction has been single-cycled to completion.

Because the Save, Insert, Display MAR, Modify, and Reset keys are not operative in the automatic mode, the Instant Stop/SCE key should *NOT* be used to stop

the computer when it is necessary to perform manual operations.

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

Thermal Light. The Thermal light is turned on if the internal temperatures of the 1620, 1622, 1625, or 1311 become too high. The Thermal light may be turned off by depressing the Reset key, after the internal machine temperatures return to normal.

Write Interlock. This light is turned on when the computer executes a write operation specifying an output unit and the unit is not in a ready condition.

This light is also turned on when a parity check error occurs while punching paper tape.

When either of these conditions occur, the computer is stopped in the automatic mode; manual corrective procedure is required. These conditions are readily detectable because the Automatic light remains on and the Write Interlock light is turned on. The Write Check indicator (and possibly one or more of the other I/O, Parity, or Disk indicators) may also be on. Depression of the Release key disconnects the output unit and puts the computer in manual mode. While in manual mode, depression of the Reset key turns off the Write Interlock light, and depression of the Check Reset key turns off the Parity and I/O indicators. The operator can then take the necessary corrective procedures.

In disk storage operations the Disk Check indicators are reset by pressing the Release and Reset keys simultaneously.

Read Interlock. This light is turned on when the computer executes a Read operation specifying an Input unit and the unit is not in a ready condition. A ready condition exists when the Power Ready light is on and the input unit is properly loaded and made ready.

Check Stop Light. The Check Stop light is turned on when the machine stops because of a Parity or I/O check, including disk drive and EXP CHK errors. 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 by the Check Reset key or by program interrogation or when the Parity or I/O switch is set to PROGRAM.

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.

Register Display Indicators

The console panel displays the contents of registers by small incandescent lights that represent the bits present in each digit of a register (Figure 81). Each light, representing a particular bit position, is on only when its corresponding bit is present in the digit displayed.

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

Multiplier/Quotient. This five-light register display shows each multiplier digit as it is used during a multiply operation. During divide, the M/Q register is used to develop quotient digits.

Data Register. Two lines, each with five lights, display the contents of the Data Register. This register performs three functions 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.
3. It holds one of the digits used in add or subtract operations.

Memory Buffer Register (MBR). All data read *from* core storage passes through this register. The two digits affected by a core storage address are displayed in the MBR. When the core storage location addressed is an even-numbered address, the digit at this location is placed in the MBR display in the E (even) line; the O (odd) line contains the digit in the next higher-numbered location. If the core storage location addressed is an odd-numbered address, the digit at this location is placed in the MBR display on the O line; the E line contains the digit in the next lower-numbered location. When the machine is in alphabetic mode, the complete two-digit representation of an alphameric character may be viewed at one time.

Memory Inhibit Register (MIR). All data *written into* core storage passes through this register. The two digits affected by a core storage address are displayed in the

MIR. When the core storage location addressed is an even-numbered address, the digit at this location is placed in the MIR display in the E line; the O line contains the digit in the next higher-numbered location. If the core storage location addressed is an odd-numbered address, the digit at this location is placed in the MIR display on the O line; the E line contains the digit in the next lower-numbered location. When the machine is in alphabetic mode, the complete two-digit representation of an alphameric character may be viewed at one time.

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

Memory Address Register Storage (MARS) Display Selector. The 12-position rotary switch permits any of the 12 MARS registers to be selected for display in MAR by depressing the Display MAR key. The position of the switch can be changed without altering the display. The rotary switch should not be turned, however, while the Display MAR key is depressed.

The MARS registers provide a visual indication of internal data flow for the console operator. Further information on the use of individual registers and the incrementing and decrementing of the MAR addresses may be found under CONSOLE OPERATING PROCEDURES.

Machine Status Lights

The remaining lights on the console panel, Figure 82, are used primarily by IBM Customer Engineers for diagnostic testing. However, some of these lights can

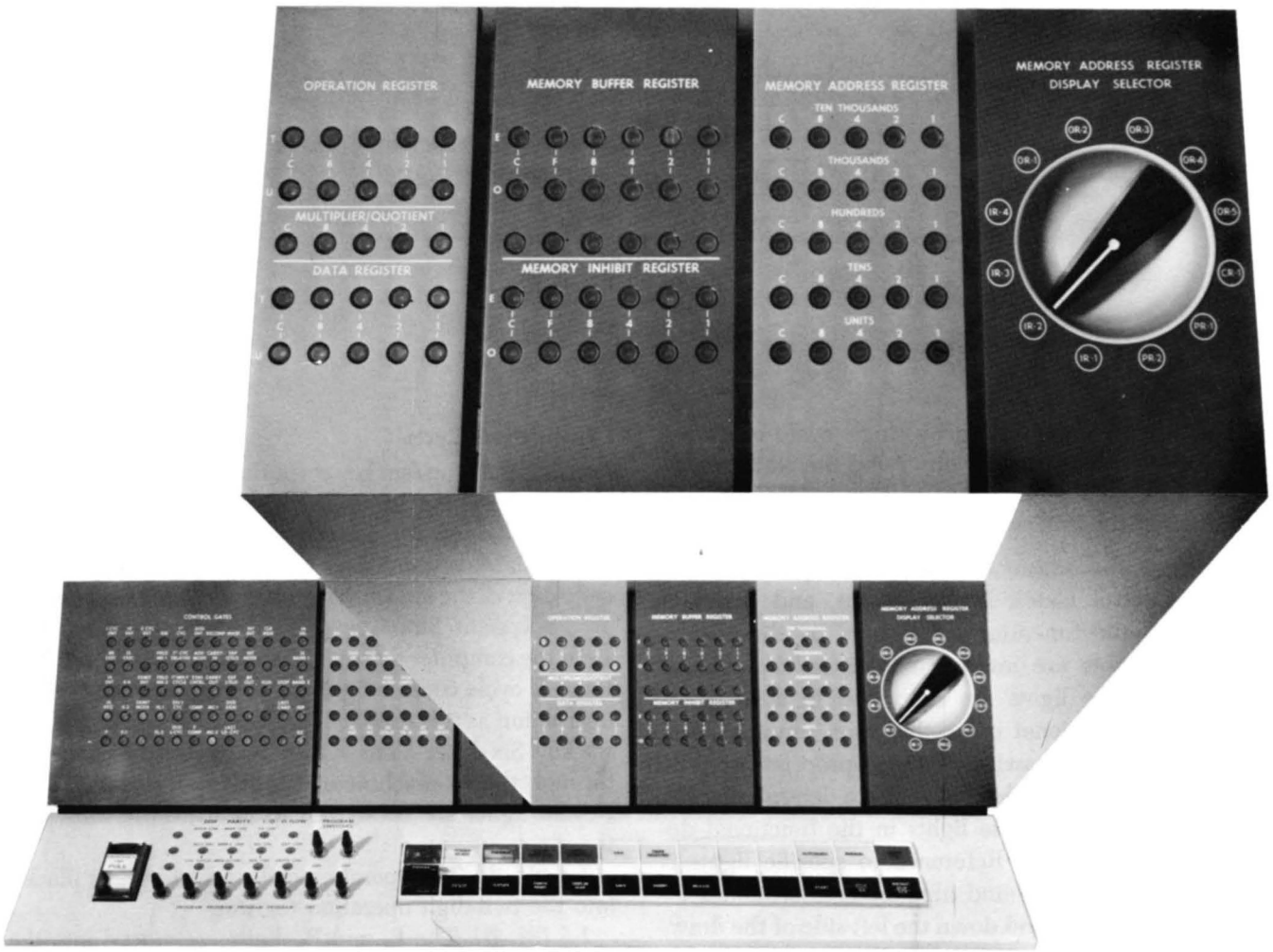


Figure 81. Register Display and Selector

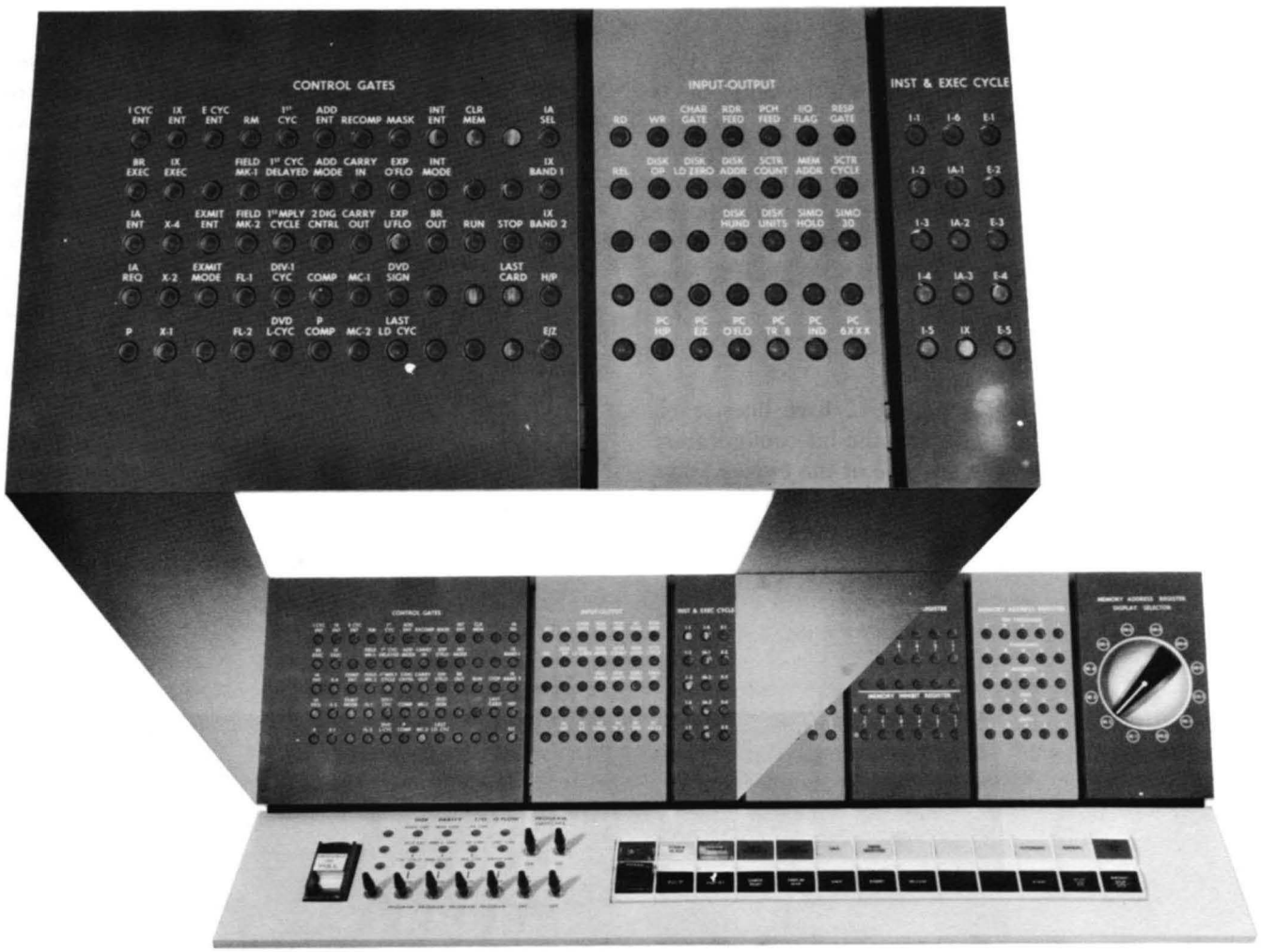


Figure 82. Machine Status Lights

be used in program testing by single-cycle executing an instruction and visually observing the status of internal indicators and observing the progress of the execution of the instruction.

Although the status lights are arranged in three groups—Control Gates, Input/Output, and Inst and Exec cycle—the functions represented by these three groups of lights are interrelated within the groups. Therefore, these lights are described in a sequence based upon functional use rather than physical location. Figure 83, a drawing of the appropriate console panel sections, is provided to facilitate determining the physical location of these lights in the functional descriptions that follow. Reference to specific lights is made by the numeric and alphabetic coordinates located across the top and down the left side of the drawing. Lights that are neither labeled nor explained have no programming or operating significance.

I (Instruction) Cycle

Pressing the Stop/SIE key stops the computer at the end of the instruction in operation at the time the key is pressed. Except for Branch instructions that will be executed, IR-1 contains the address of the first operation code digit of the next instruction. Successive depressions of the Instant Stop/SCE (Single Cycle Execute) key steps the computer through 10-μsec machine cycles. The I and E cycle console lights show the progression of an instruction as it is executed with each depression of the SCE key. Six depressions of the SCE key step the computer through the six machine cycles of any instruction. The I-cycle lights are turned on to indicate the following functions.

I-1 (21, A). The operation code, O_0 and O_1, is placed into the two-digit operation register.

I-2 (21, B). The P_2 and P_3 digits are placed into the ten-thousands and thousands positions of Operand Register 2 (OR-2).

I-3 (21, C). The P_4 and P_5 digits are placed into the hundredths and tens positions of OR-2.

I-4 (21, D). The P_6 digit is placed in the units position of OR-2.

I-5 (21, E). The Q_7, Q_8, and Q_9 digits are placed into the ten-thousands, thousands, and hundreds position of OR-1.

I-6 (22, A). The Q_{10} and Q_{11} digits are placed into the tens and units position of OR-1.

E CYC ENT (3, A). At the end of the I-6 cycle this light is turned on to indicate that the instruction just interpreted is about to be executed. At the end of the I cycle, IR-1 has stepped twelve addresses higher to the first operation code digit of the next instruction in the program. IR-1 remains at this address during the following E cycle.

If the instruction is a disk storage instruction, the data in the Disk Control Field must be loaded into appropriate storage registers.

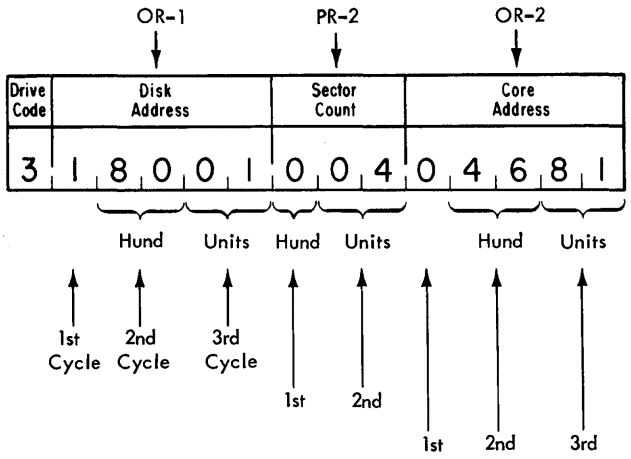
Loading Registers for Disk Storage Operations

The operation of loading these registers cannot be stepped through by successive depressions of the SCE key. All of the disk operations that are described here occur automatically following the I-6 cycle; however, certain types of errors, such as invalid addresses or invalid Q_{11} modifier digits, will stop the computer at the end of the loading operation and before the actual execution of the instruction using the disk storage drive, occurs. Therefore, at this point, the contents of the registers have not been altered and can be displayed in the Memory Address Register.

Disk OP (14, B). This light is turned on during I-6 time if the instruction addresses disk storage. It remains on during the load operation.

DISK LD ZERO (15, B). This light is turned on during the time the Drive code is examined to determine the correct Disk Storage module to be addressed. Also, at this time the OR-1, PR-2, and OR-2 registers are being set to zeros.

DISK ADDR (16, B). This light is on during the three machine cycles required to load the disk address into OR 1. The specific digits transferred in each cycle is indicated in the drawing which follows.



The function of the labels "HUND" and "UNITS," which have been arbitrarily assigned to aid in testing operations, are described later.

SCTR COUNT (17, B). This light is on during the two machine cycles required to load the sector count into PR-2.

MEM ADDR (18, B). This light is on during the three machine cycles required to load the core storage address into OR-2.

HUND and UNITS (16-17, B). These two lights are turned on to indicate which machine cycle is taking place during the loading operations. (Refer to preceding drawing.) The UNITS light is on when the units and tens positions are loaded; the HUND light is on when the hundreds and thousands positions are loaded. Both lights are *off* during the first cycle when OR 1 and OR 2 are loaded.

SCTR CYC (19, B). This light comes on after the third cycle of the OR-2 loading operation (MEM ADDR light). Certain types of errors such as invalid addresses or invalid Q_{11} modifier digits stop the computer at this point with this light on. Normally—that is, with no error conditions—the registers are loaded immediately following I-6 cycle. When the program is being "stepped through" the next light on, following the I-6 light, is the I-Cycle Entry light.

E (Execute) Cycle

During the E cycle the instruction is executed as specified by its operation code and P and Q addresses. The number of SCE key depressions necessary to step the computer through an E cycle is determined by the particular instruction being executed and the size of the P and Q fields.

The uses of the five E cycles are too extensive to describe in detail, therefore only their general usage is indicated. The specific use of the E cycle during an add instruction is described later.

E-1 (22, A). This light is on during operations involving a Q field.

E-2 (22, B). This light is on during operations involving the P field.

E-3 (22, C). This light is on during multiply and divide operations.

E-4 (22, D). This light is on during multiply and divide operations.

E-5 (22, E). This light is on during multiply and divide operations when the product area is cleared to zeros.

Add Operation

Nine status lights are related to an add operation. An explanation of the E cycles for an add operation are described first to facilitate understanding of the meaning and use of these lights.

E CYCLES OF ADD OPERATION—2 DIGITS

Data flow during the E cycles of an add instruction is illustrated in Figure 84. The Q field is in core storage positions 14000 and 14001; the P field is in positions 00801 and 00802.

1. During the E-1 cycle, the two digits in the Q field are placed into the tens and units positions of the Data Register.
2. During the first E-2 cycle the units position of the P field (an even address in this example) and the digit 6 from the odd address (which is not actually part of the specified Q field) are placed into MBR-E and MBR-O. The 3 from MBR-E and the 4 from the units position of the Data Register are sent to the Adder. The result of 7 is routed through the Memory Inhibit Register-Even (MIR-E) and returned back to the units position of the P field (00802). The 6 from MBR-O is routed through MIR-O and then returned to core storage position 00803.
3. During the second E-2 cycle the tens position of the Data Register (1) is transferred to the units position of the Data Register. A zero (C bit) is written into the tens position. The digit 5 from the tens position of the P field (an odd address)

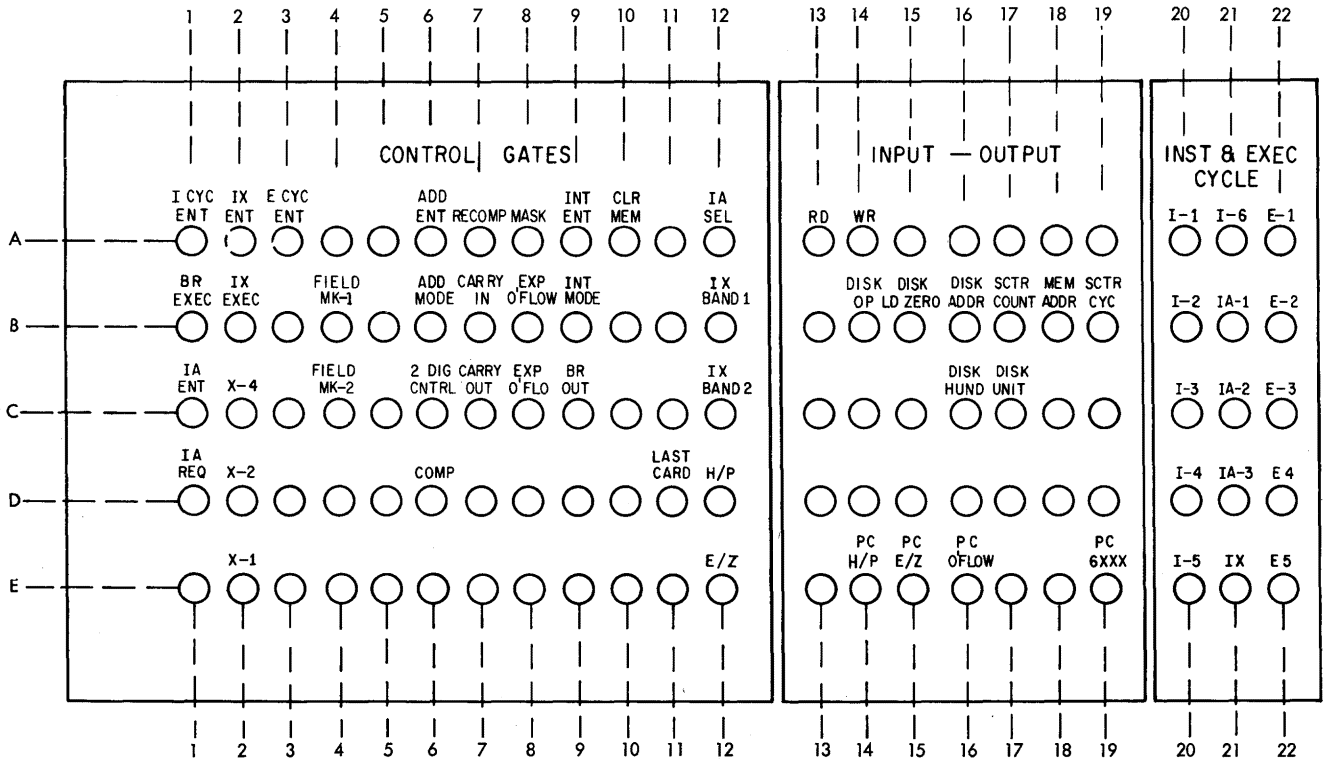


Figure 83. Drawing—Machine Status Lights

and the digit 4 from the even address (which is not actually part of the Q field) are placed into MBR-E and MBR-O. The 5 from MBR-O and the 1 from the units position of the Data Register are sent to the Adder. The result 6 is routed through MIR-O and returned back to the tens position of the P field. The 4 from MBR-E is routed through MIR-E and then returned to core storage position 00800.

Three cycles are required to add two digits, unless the units position of the Q field is an even address; in that case, only one E-2 cycle is taken and the first digit is added in two cycles. The remaining digits of the field are added as described above; three cycles for each two digits. Additional cycles may be required if the result must be recomplemented.

When an add instruction is single-cycle executed, the results of the addition can be verified at the end of each E-2 cycle; the original P digit is displayed in MBR, the Q digit is displayed in the units position of the Data Register, the new P digit (result) is displayed in MIR.

ADD ENT (6, A). When the Add Entry light comes on in conjunction with the E CYC ENT light, the operations following will be in the "add mode." It is turned off on the following cycle.

ADD MODE (6, B). This light is on during all of the E cycles of add, subtract, and compare instructions.

2 DIG CNTRL (6, C). The 2-Digit Control light, is turned on during the E-1 cycle to indicate that the following E-2 cycle is the "1st" E-2 cycle. (Refer to Figure 84.) This means that the Q address was an odd-numbered address and therefore both digits read into the Data Register will be used.

COMP (6, D). The Complement light is turned on to indicate the Q digits are complemented before they are sent to the Adder.

RECOMP (7, A). The Recomplement light indicates that an add or subtract result will be recomplemented at the end of the last E-2 cycle. Recomplementing is required in any add operation when the sign of the P and Q fields are initially unlike and the absolute value of the P field is less than the absolute value of the Q field.

CARRY OUT (7, C). This light is turned on during the cycle in which a carry occurs. A carry occurs when the addition of two digits results in a value greater than nine. On the following cycle the Carry In light will be on.

CARRY IN (7, B). When this light is on, the Q digit is increased by one as it is sent from the Data Register to the Adder.

H/P (12, D). The High/Positive light shows the condition of the internal High/Positive indicator as a result of the last arithmetic or compare operation.

E/Z (12, E). The Equal/Zero light shows the condition of the internal Equal/Zero indicator as a result of the last arithmetic or compare operation.

Indexing

Eight status lights are associated with the special feature Indexing.

IX ENT (2, A). The Indexing Entry light is turned on to indicate that the next cycle begins an indexing operation; that is, an operation in which a P or Q address is changed as the result of adding the value of an index register. If the P field is to be indexed, this light is turned on during I-4 time; if the Q field is to be indexed, this light is turned on during I-6 time. If, for example, the P field is to be indexed, the IX ENT light comes on during I-4 time and the next light to come on is the IX light. The I-5 light does not come on until after the fifth cycle of the five cycles required to index the P field.

IX (21, E). This light is on during the five cycles required to add the contents of an index register to a P or Q field. During single-cycle execution, the original P or Q field digit is displayed in MBR, the digit from the index register is displayed in the units position of the Data Register, and the result (the new P or Q digit) is displayed in MIR.

IX EXEC (2, B). The Index Execute light is on during any of the seven modify, store, or load index register instructions (Op codes 61-67).

X-4 (2, C). The Index 4-bit light is turned on when a flag bit is sensed in either the P_3 position or Q_8 position of an address.

X-2 (2, D). The index 2-bit light is turned on when a flag bit is sensed in either the P_4 position or Q_9 position of an address.

X-1 (2, E). The Index 1-bit light is turned on when a flag bit is sensed in either the P_5 position or Q_{10} position of an address.

IX BAND 1 (12, B). This light is on if Index Registers Band 1 has been selected by a Branch and Select instruction. It is off if Band 2 or Band 0 (no band) has been selected.

IX BAND 2 (12, C). This light is on if Index Registers Band 2 has been selected by a Branch and Select instruction. It is off if Band 1 or Band 0 (no band) has been selected.

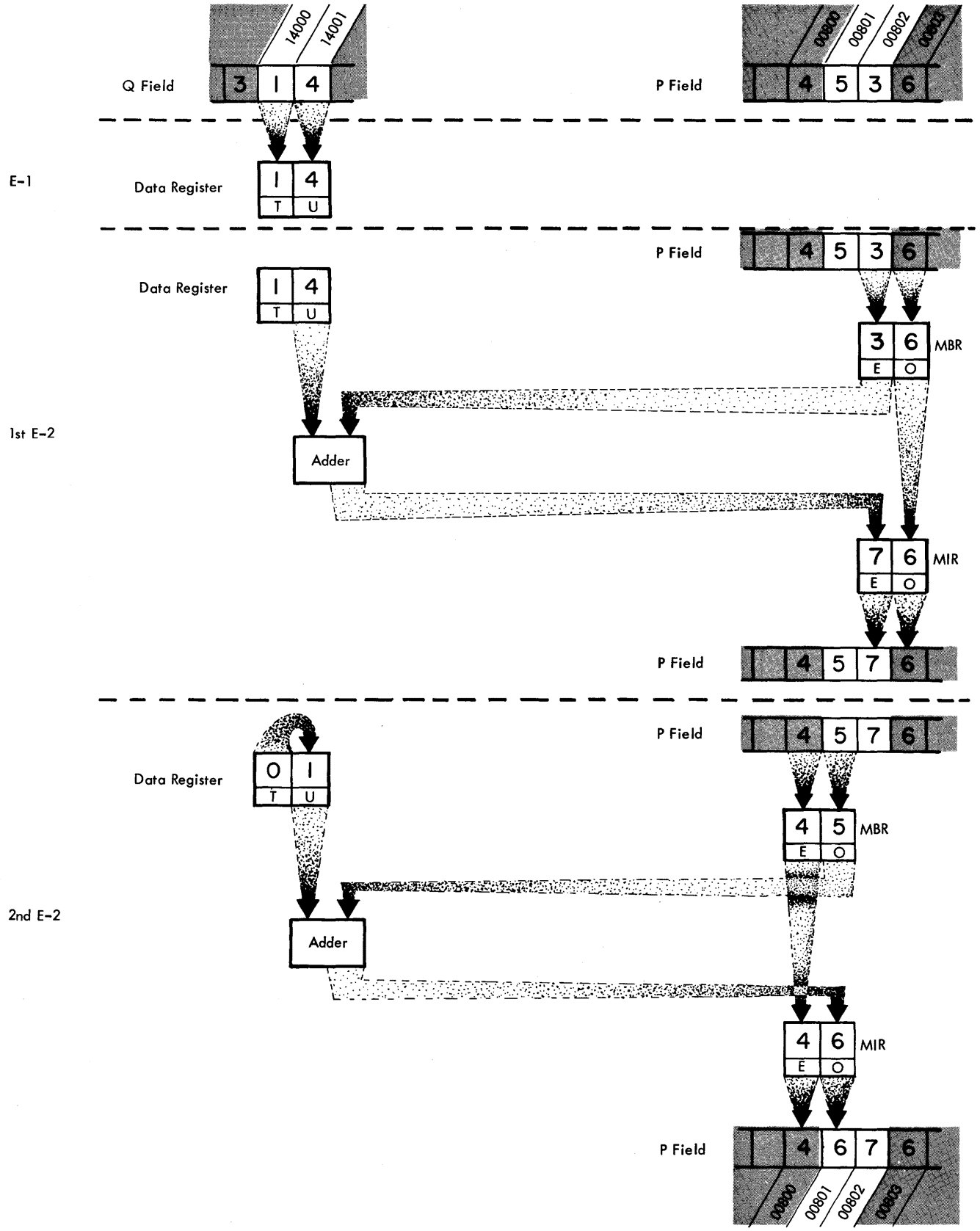


Figure 84. Data Flow – E Cycles

Indirect Addressing

Six status lights are associated with indirect addressing operations.

IA Select (12, A). The Indirect Addressing Select light is on when the indirect addressing feature has been turned on by a Branch and Select instruction with a Q_{11} digit of 9. The light and feature are also turned on when power is turned on. The feature and the light are turned off by a Branch and Select instruction with Q_{11} digit of 8.

IA REQ (1, D). The Indirect Address Required light is turned on when a flag bit is sensed in the P_6 or Q_{11} positions of an instruction (in which indirect addressing is permitted). If an indexing operation is *NOT* required (no flag bits in either P_3, P_4, P_5, Q_8, Q_9, and Q_{10}) the IA ENT light is also turned on.

IA ENT (1, C). The Indirect Address Entry light is turned on when an indirect address is required and an indexing operation is not required. It indicates that the next cycle will be the first cycle of the three cycles required to select an indirect address.

IA-1 (21, B). The Indirect Address-1 light is on during the first cycle of the three cycles required to load the indirect address into OR-2 or OR-1, depending upon whether the indirect address is a P field or Q field address respectively.

The specific digit positions loaded into the Operand Registers during each of the three indirect address cycles depends upon whether the P or Q address is an odd or even number as illustrated in Figure 85.

IA-2 (21, C). The Indirect Address-2 light is on during the second cycle of the indirect address loading operation.

IA-3 (21, D). The Indirect Address-3 light is on during the third cycle of the indirect address loading operation.

Miscellaneous Status Lights

Nine status lights are concerned with significant but unrelated functions.

BR EXC (1, B). The Branch Execute light comes on at the end of the branch operation, *if* the branch is to occur. It can be observed at I-Cycle Entry time.

Field Mk 1 (4, B). The Field Mark 1 light comes on when the flag bit in the high-order position of the Q field is detected in the Data Register.

Field Mk 2 (4, C). The Field Mark 2 light comes on when the flag bit in the high-order position of the P field is detected in the MBR Register.

EXP O'FLOW (8, B). The Exponent Overflow light is turned on and off during floating-point operations; however, if it is on at the end of the operation, an exponent overflow has occurred, that is, an exponent greater than +99 has been generated. The Exponent Check indicator and light will also be on.

EXP U'FLOW (8, C). The Exponent Underflow light is turned on and off as a normal function during floatingpoint operations; however, if it is on at the end of the operation, an exponent underflow has occurred, that is an exponent less than —99 has been generated. The Exponent Check indicator and light will also be on.

CLR MEM (10,A). The Clear Memory light is turned on as a result of pressing the Modify and Check Reset keys simultaneously. If the Start key is pressed (after pressing Modify and Check Reset) this light is on as core storage is being cleared.

LAST CARD (11, D). The Last Card light is turned on when the data from the last input card has been transferred from the 1622 Input Buffer to 1620 Core Storage without a parity error.

RD (13, A). The Read light comes on during operations involving an input unit.

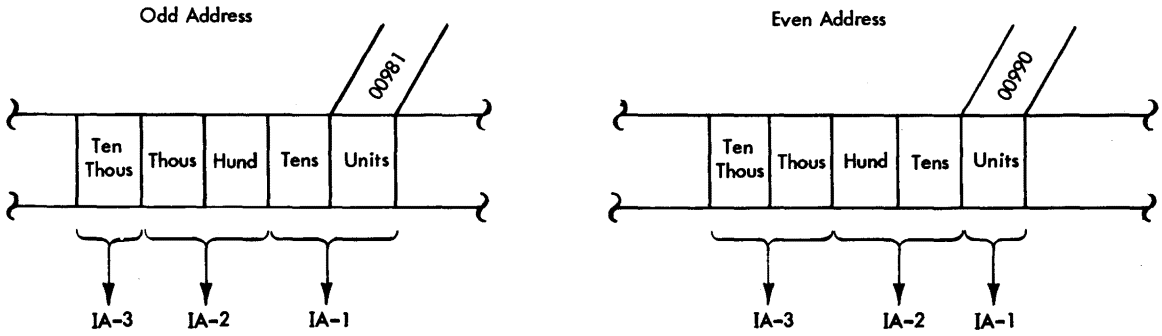


Figure 85. Positions Read During IA Cycles

WR (14, A). The Write light comes on during operations involving an output unit.

Status Lights—1710 Control System Only

Eight status lights are used only when the 1620 is used in a 1710 Control System.

MASK (8, A). 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.

INT ENT (9, A). The Interrupt Entry light is turned on to indicate that the next cycle is in the "interrupt mode."

INTERRUPT MODE (9, B). 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 or a Branch Out and Load instruction. In effect, the Interrupt Mode light is on when IR-3 and IR-4 are used in place of IR-1 and IR-2. If a Branch Out instruction is executed and this light is off, (system not in interrupt mode) this light is turned on and then off as the Branch Out instruction is executed.

BR OUT (9, C). The Branch Out light is turned on during a Branch Out and Load instruction. During a single-cycle execution of either of these two instructions, the Reset key must not be pressed until the instruction is completed. If the instruction is not completed – the BR OUT light will be on – and the Reset key is pressed, the computer remains in the interrupt mode.

PC H/P (14, E). During the time the computer is in the interrupt mode of operation, that status (ON or OFF) of the H/P indicator is transferred to and maintained by Process Control (PC) H/P indicator. The H/P indicator is then available for use during the interrupt mode.

This light indicates the status of the PC H/P indicator.

PC E/Z (15, E). During the time the computer is in the interrupt mode of operation, the status of the E/Z indicator is transferred to and maintained by the PC (Process Control) E/Z indicator. The E/Z indicator is then available for use during the interrupt mode.

This light indicates the status of the PC E/Z indicator.

PC O'FLOW (16, E). During the time the computer is in the interrupt mode of operation, the status of the Arithmetic Overflow indicator is transferred to and maintained by the PC overflow indicator. The Arithmetic Overflow indicator is then available for use during the interrupt mode.

This light indicates the status of the PC/O'FLOW indicator.

PC 6xxx (19, E). This indicator and light are turned on during either of the two branch indicator instructions – at I-5 time – if the indicator code is a 4-digit code instead of a 2-digit code. The last two digits of the code (Q_{10} and Q_{11}) are transferred to the data register at I-6 time.

The 1620 console typewriter (Figure 86) is used for both input and output.

Figure 86. IBM 1620-2 with I/O Typewriter

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 is locked except when entering data. Pressing the Insert key 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 on an insert operation. 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 keyboard 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 specified in the P address of the instruction and enters core storage at successively higher-numbered locations until the Release key is pressed.

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 86). Pressing either key again locks the keyboard and gives the computer an end-of-input/output indication.

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 turns on the Read Check indicator and light (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 and light (07). Also, a horizontal bar is overprinted across the center of the character. An invalid character with correct parity causes a special character (▌), identified as a "pillow," to be printed.

If a parity error occurs, 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.

Typewriter Features

The easily removed typing element traverses the stationary paper while printing a maximum of 15.5 cps.

A 9 and 3/8-inch pin feed platen is supplied with the system; a 10 and 7/8- inch solid platen is available. The moving type element provides an 8 and 1/2-inch printing line (85 characters) irrespective of platen size. Forty-seven alphabetic, numeric, and special characters can be printed. The numeric keys are colored differently for ease of keyboard selection. In addition, the characters listed in Table 12 print to indicate specific conditions.

Typewriter Controls

The controls and keys numbered one through eight (Figure 87) are used to facilitate typewriter operation. Instructions for their use follow.

1. Paper Release Lever – Move the lever forward to position or remove paper.
2. Paper Guide – Place this guide at the second mark from the left, and the left edge of the paper will correspond to 0 on the margin guide.
3. Line Spacer Lever – Position this lever for single or double vertical line spacing.
4. Multiple Copy Control Lever – Move this lever to compensate for additional copies. The platen is adjusted so that the typing element strikes squarely on the paper. The second marking is for one original copy with three carbon copies; the third marking is for one original with five copies.
5. Tab Control – To clear tab settings, tabulate to each tab stop that you wish to clear and press the top (CLR) of the tab control key.

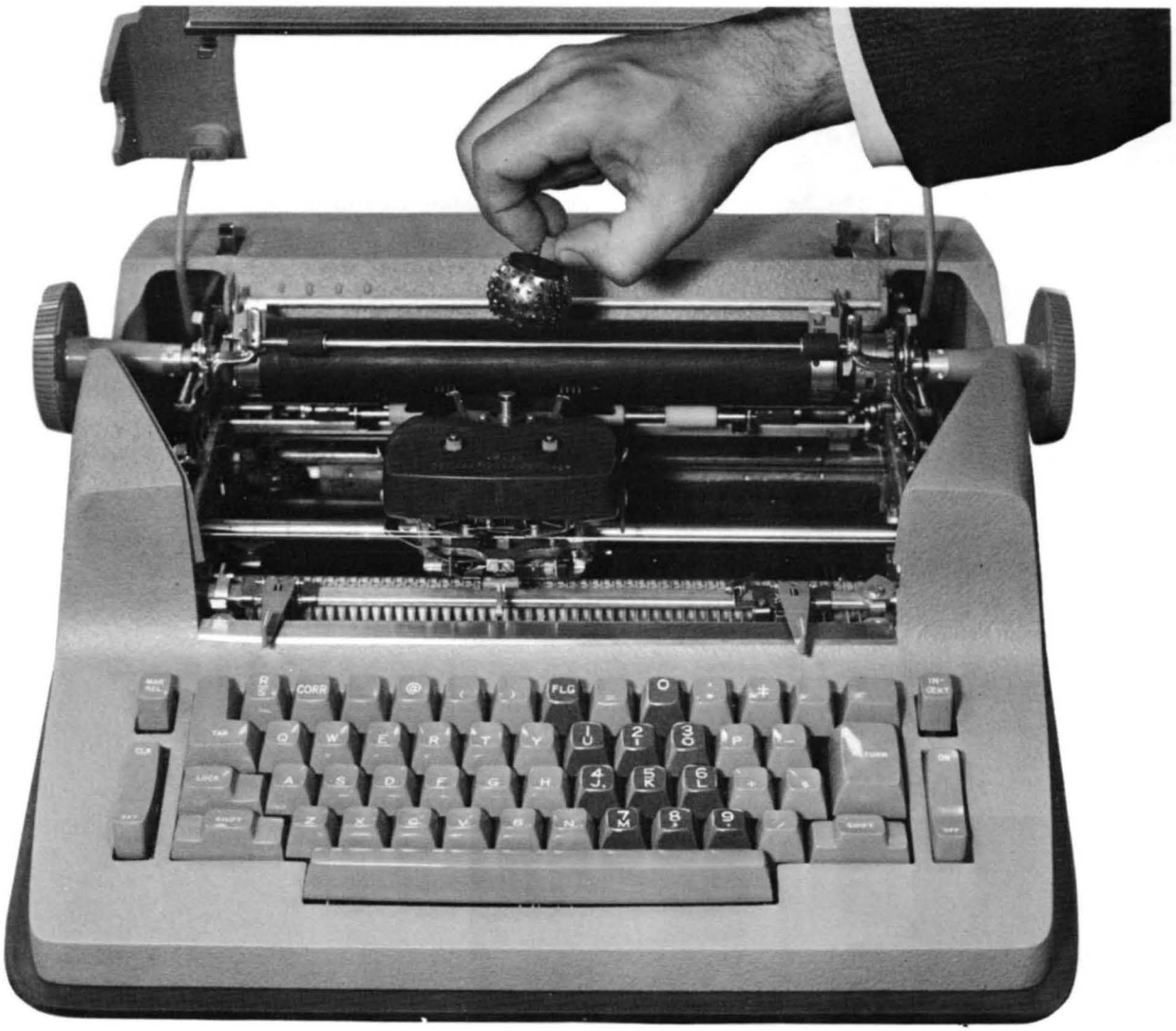
   To set tab stop, position the carrier at the desired point on the line and press the bottom (SET) of the tab control key.
6. Margin Release Key – Keeping this key pressed releases both the left and right margin stops.

   Margin Stops – To manually position the left and right margin stops push them toward the platen and guide them along the margin guide channel.
7. Insert Key – This key has the same function as the console Insert key. Either key may be pushed to ready the typewriter for input into location 00000.
8. CORR (Correction) Key – This key enables the operator to correct a manual entry error. For example, if 26 is erroneously entered into memory instead of 36, two depressions of the Correction key can be used to:
   (1) Effect the backspacing of memory (the address in OR-2 is decremented twice, once for each key depression).
   (2) Print two strike-through characters.

   The correct entry 36 can then be made, leaving 26 – – 36 as a printed record of operator action. Each time the Correction key is struck during Read Alphamerically operations (typewriter-selected), the address in OR-2 is decremented by two.
9. Paper Bail – On non pin-feed platens, the paper bail should be resting on the platen when the forms are being pressure-fed. If the forms are pin-fed, the paper bail should be raised up from the platen and moved forward toward the operator.

Table 12. Program Control Characters

Character	Symbol	Use
Record Mark	‡	Prints when ‡ key is struck during input, and when record mark is sensed during a Dump Numerically operation.
Overscore (labeled FLG on keyboard)	$\overline{2}$	Prints when FLG key is struck during input, and when a flag bit is sensed during output.
Strike-through	– or ~~A~~	Prints when CORR key is struck during input (see Correction key), and prints over or scores through parity errors during output.
Pillow	▮	Prints when an invalid character is sensed during output.
Release/Start	$\frac{R}{S}$	Prints when $\frac{R}{S}$ (Release and Start) key is struck.



Removing and Replacing the Typing Element

TO REMOVE TYPING ELEMENT

1. Place the typewriter in lower case, so that the arrow on the typing element points toward the platen (Figure 88).
2. Turn the motor control to OFF.
3. Raise the cover to its uppermost position.
4. Firmly press together the spring levers on top of the typing element.
5. Lift the typing element up and off, of the element post (Figure 89).

TO REPLACE TYPING ELEMENT

1. Hold the typing element by grasping the spring levers.
2. Point the arrow toward the platen; ease the typing element down into position until firmly seated (Figure 89).
3. Release the spring levers. When the typing element is seated, you will hear a click, and it cannot be removed without pressure on the spring levers.

Changing the Ribbon Cartridge

TO REMOVE RIBBON

1. Move carrier toward center of carrier rod.
2. Turn motor control to off.
3. Lift the cover to its uppermost position.
4. Raise the ribbon guide by shifting the ribbon change lever to the right (Figure 90).
5. Lift ribbon cartridge straight up off of the carrier ribbon posts (Figure 91).

Figure 88. Typing Element in Position

6. Ease ribbon through slots in ribbon guides.
7. To rewind excess ribbon, insert a wooden pencil in the cartridge hole. Turn in the direction of the arrow.

TO INSERT RIBBON

1. Position ribbon cartridge with exposed length of ribbon facing the platen.
2. Slide ribbon down through slots in the ribbon guides. (Figure 92).
3. Position the ribbon cartridge on the cartridge posts and press down firmly.
4. To rewind excess ribbon, turn either cartridge post in the direction of the arrow.
5. Move ribbon change lever to left, to lower ribbon guide into typing position.

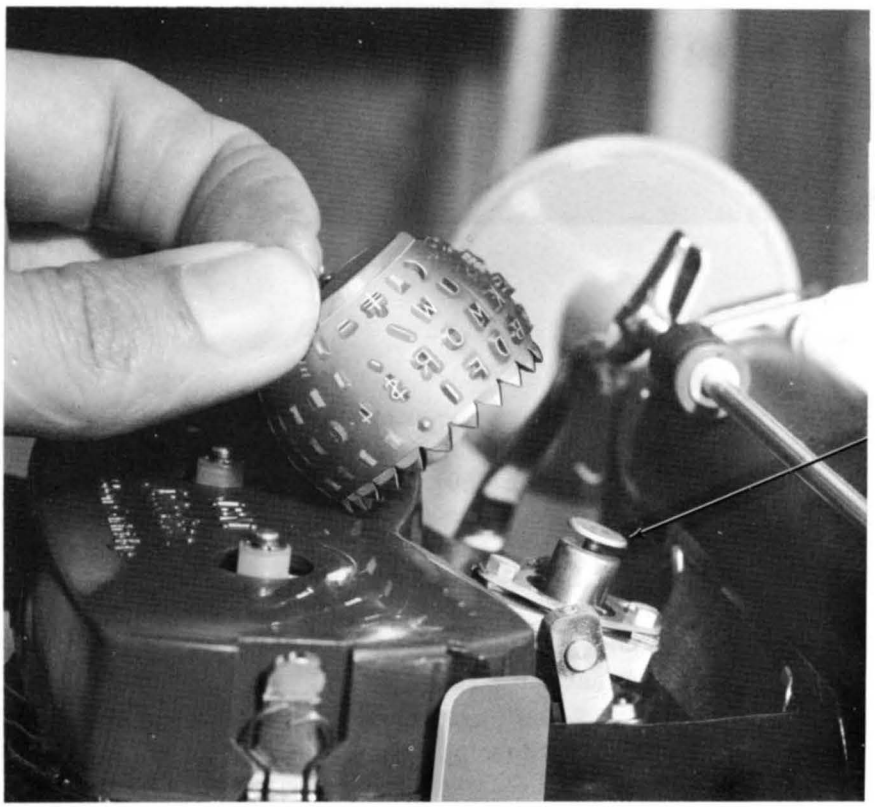


Figure 89. Removing and Replacing Typing Element

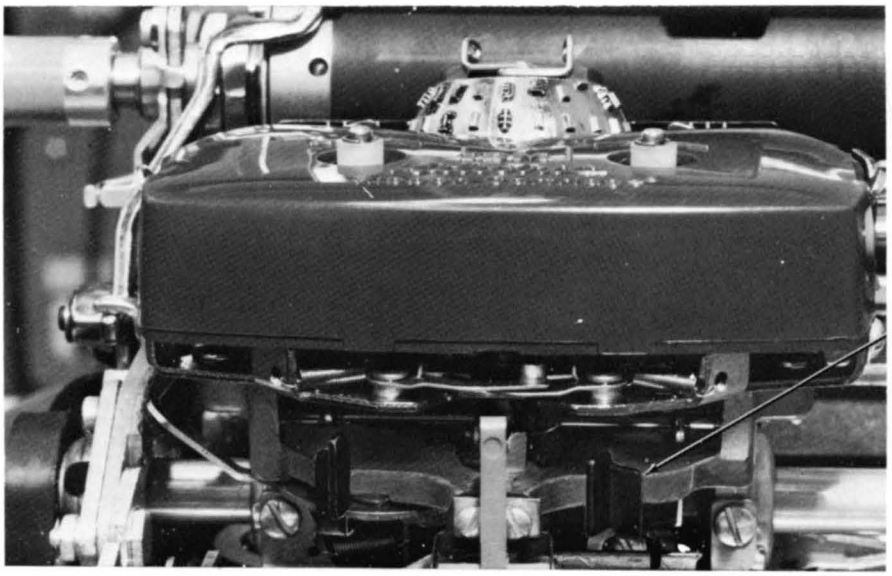


Figure 90. Release – Ribbon Change Lever

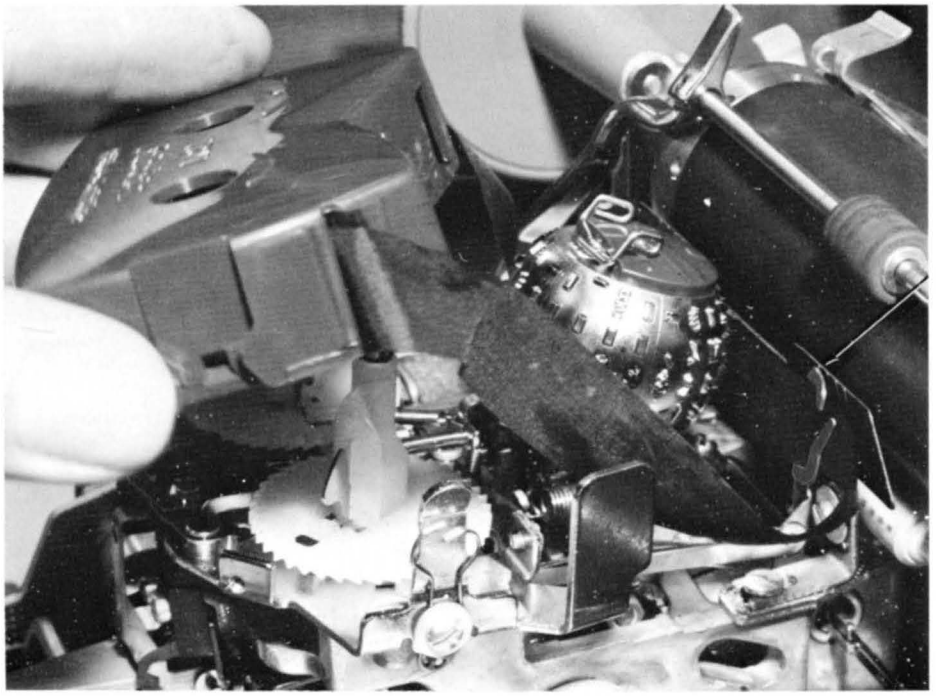
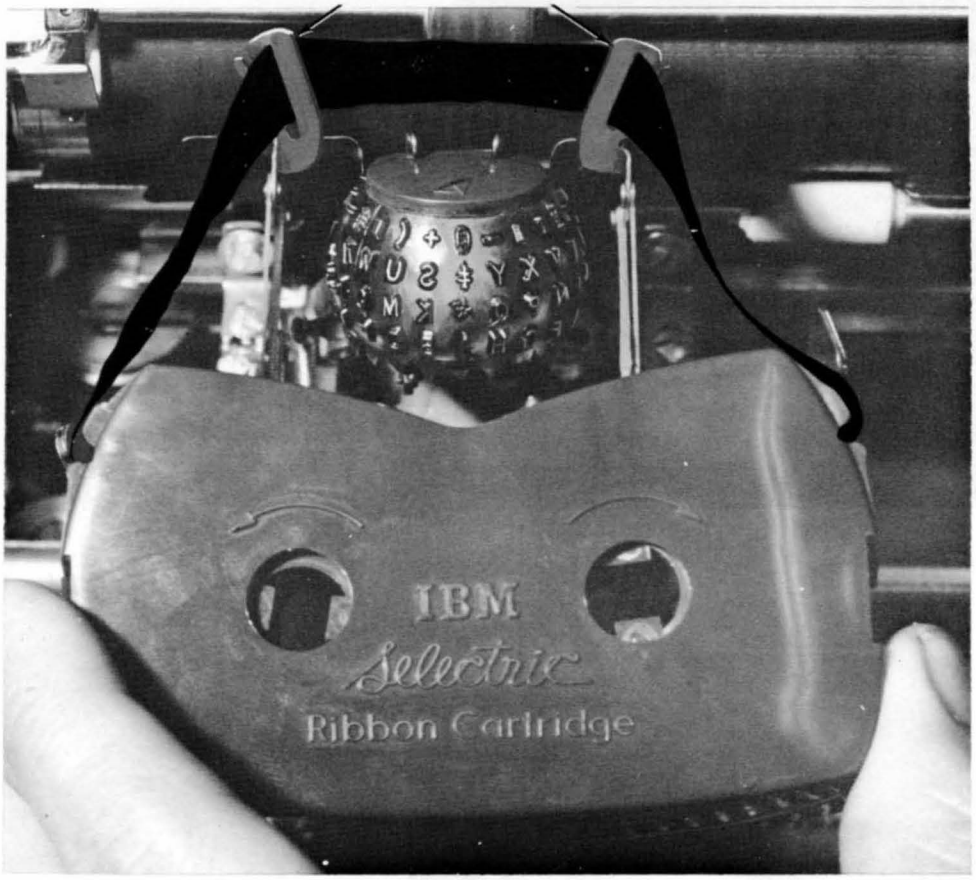


Figure 91. Removing Ribbon Cartridge


IBM
Selectric
Ribbon Cartridge

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

NOTE: The Insert key is operative only when the computer is in the manual mode. If the computer is running in the automatic mode, pressing the Stop/SIE key stops the computer at the end of the instruction in progress when the key was pressed and places the computer in the manual mode.

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 R/S key.	Releases typewriter. The Read Numerically instruction, entered in step 2, is executed.
4.	Type program steps and data	As each character is typed, it is stored at location xxxxx and succeedingly higher-numbered core storage positions.
5.	Press R/S key.	Terminates read instruction. 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 R/S key.	Releases typewriter. 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/SIE 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 R/S key.	Releases typewriter. The Read Numerically instruction, entered in Step 4, is executed.
6.	Type instructions and data.	As each character is typed, it is stored at location xxxxx and succeedingly higher-numbered core storage positions.
7.	Press R/S key.	Terminates read instruction. 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 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 pressed.
3.	Press R/S key.	Releases typewriter. Instruction entered in Step 2 is executed, and the system stops.

Check Program Step Sequence and Operation

	Operator Action	Explanation
1	Press Stop/SIE 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 Core Storage to Zeros

	Operator Action	Explanation
1.	Press Stop key.	Halts processing and initiates manual mode.
2.	Press Modify and Check Reset keys *simultaneously.*	Sets up circuitry for clearing core storage.
3.	Press Start key.	Core storage is cleared to zeros.

Display P and Q Addresses

	Operator Action	Explanation
1.	Press Stop/SIE 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 six times. *Note the on/off conditions of the machine status and check indicators and of the signal lights to be reset by Reset (step 5), so that proper restart can be initiated after the display has been completed.* At this time a branch to the original instruction also must be inserted if it is desired to excute this instruction and proceed with the normal program. (If reset and the subsequent necessity for branching and proper restart are undesirable, step 5 can be omitted and the P and Q addresess can be viewed, two digits at a time, during SCE.)	This steps through the six I cycles.
4.	Press Reset key.	Initiates manual mode.
5.	Turn MARS switch to OR-1 (Operand Address Register 1) and press 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 press the Display MAR key.	The P address, which is in OR-2, is displayed in MAR.

Timing

In some applications of the 1620 Data Processing System the input, output, and processing capabilities of the system can be used most effectively if careful timing consideration is given to the development of the program. In many systems, estimates of total job time are required in order to coordinate the processing of various programs in an installation.

This section includes methods of estimating timing requirements for input/output operations, for computing, and for total job time.

1622 Card Read-Punch

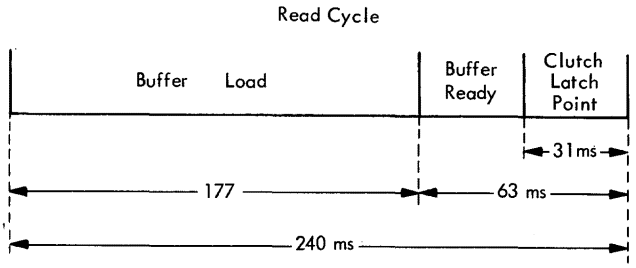
There are two models of the 1622 Card Read-Punch available. Both models are identical in programming and operation except for their different reading and punching speed as shown below:

	Model 1	Model 2
Reading	250 cpm	500 cpm
Punching	125 cpm	250 cpm

Both reading and punching operations are completely buffered. This means, for example, that a read instruction causes the data in the read buffer to be transferred to core storage and then causes the data in the following card to start reading into the read buffer at the beginning of the next card cycle. The processing unit is interlocked with the 1622 only during the 1.7ms required to read data out of the buffer unit and into core storage. Therefore, the card cycle time – the time required to read the data from the card into the buffer – can be used by the processing unit for computing or other input/output operations. The overlap time – that is, the time when the processing unit is performing other operations – varies, depending upon the length of the read or punch cycle. It must be remembered, however, that if another read instruction is given while the read buffer is accepting data from a card (buffer load), or if another punch instruction is given while the punch buffer is sending data to a card (buffer unload), then the computer is interlocked and *no other processing* can occur until the read or punch buffers are ready to accept or send data respectively.

1622 Model 1

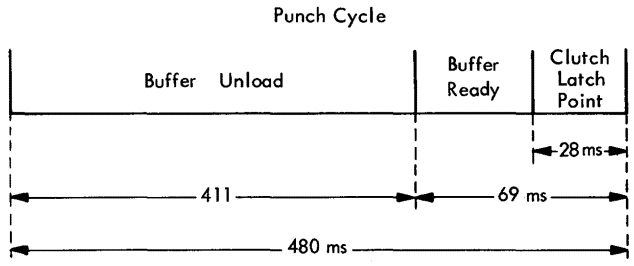
Read Cycle. The read cycle for the Model 1 is shown as follows:



A reading speed of 250 cpm provides a read cycle of 240 ms. In order to maintain continuous reading at 250 cpm, the next read instruction must be given *before* the clutch latch point which occurs 31 ms before the end of the cycle. For example, if the processing for Card 1 is completed and the next read instruction is given before the end of the buffer-load time of Card 2 (177 ms), the computer is interlocked until the buffer-ready time begins. At this point Card 2 data from buffer is transferred to core storage (in 1.7 ms), the computer is then released for other processing, and the buffer will automatically start reading Card 3 data *at the beginning of the next* card-read cycle.

Therefore, during continuous reading operations at 250 cpm, the overlap time is reduced from 240 ms to approximately 238 ms, if the next read instruction is given during the buffer-ready time.

Punch Cycle. The punch cycle for the Model 1 is shown as follows:



A punching speed of 125 cpm provides a punch cycle of 480 ms. In order to maintain continuous punching at

125 cpm, the next punch instruction must be given before the clutch latch point which occurs 28 ms before the end of the cycle.

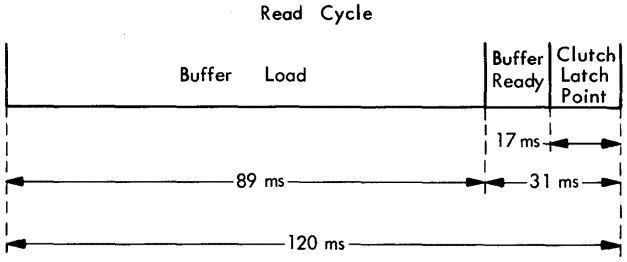
The punch is operated by a four-tooth clutch that completes one revolution every 480 ms. Because the clutch has four teeth, there are four clutch points occurring at 120-ms intervals in the punch cycle. If the processing time exceeds the 480 ms punch cycle, the next punch cycle can begin at any one of these points providing the next punch instruction is given 28 ms *before* the clutch point.

If the processing for the card is completed and the next punch instruction is given before the end of the buffer unload time (411 ms), the computer is interlocked until the buffer-ready time begins. At this point the data to be punched is transferred from core storage to the buffer (1.7 ms), the computer is then released for other processing, and the data will automatically start punching from the buffer *at the beginning of the next* card-punch cycle.

Therefore, during continuous punching operations at 125 cpm, the overlap time is reduced from 480 ms to approximately 478 ms, if the next punch instruction is given during the buffer-ready time.

1622 Model 2

Read Cycle. The read cycle of the Model 2 is shown as follows:

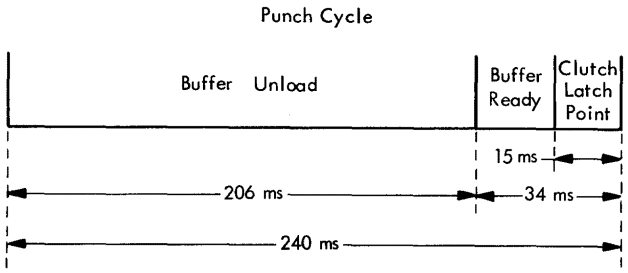


A reading speed of 500 cpm provides a reading cycle of 120 ms. The next read cycle must be given before the clutch latch point in order to maintain continuous reading at 500 cpm.

If the processing for the card is completed and the next read instruction is given before the end of buffer-load time (89 ms), the computer is interlocked until the buffer-ready time begins. At this point the data from the buffer is transferred to core storage (in 1.7 ms), the computer is then released for other processing, and the buffer will automatically start reading data from the next card *at the beginning of the next* card-read cycle.

Therefore, during continuous card reading at 500 cpm the overlap time is reduced from 120 ms to approximately 118 ms, if the next read instruction is given during the buffer-ready time.

Punch Cycle. The punch cycle of the Model 2 is shown as follows:



A punching speed of 250 cpm provides a punch cycle of 240 ms. In order to maintain continuous punching at 250 cpm, the next punch cycle must be given before the clutch latch point which occurs 15 ms before the end of the cycle.

The punch is operated by a four-tooth clutch that completes one revolution every 240 ms. Because the clutch has four teeth, there are four clutch points occurring at 60 ms intervals in the punch cycle. If the processing time exceeds the 240 ms punch cycle, the next punch cycle can begin at any one of those points, providing the next punch instruction is given 15 ms *before* the clutch point.

If the processing for the card is completed and the next punch instruction is given before the end of the buffer-unload time (206 ms), the computer is interlocked until the buffer-ready time begins. At this point the data to be punched is transferred from core storage to the buffer (in 1.7 ms), the computer is released for other processing, and the data will automatically start punching from the buffer *at the beginning of the next* card-punch cycle.

1443 Printer for the 1620

There are two models of the 1443 Printer available on the 1620. The only difference between the two models is their operating speed which is shown in the following table:

	Model 1			Model 2		
Character Set	13	39	52	13	39	52
Lines Per Minute	430	190	150	600	300	240
Print Cycle (ms)	140	316	400	100	200	250

Printing operations with the 1443 Printer are completely buffered. The processing unit is interlocked only during the 2.1 ms required to transfer the data from core storage to the printer buffer storage, the print cycle (as shown in the preceding table) is available for other processing operations.

If the printer receives another print instruction while it is printing from the buffer unit, the processing unit is interlocked until the print cycle has been completed. (Note: The Printer Busy indicator is provided with the 1443 so that interlocking in this manner can be avoided.)

The 1443 is asynchronous; that is, the printing time is not synchronized with the starting or stopping of a specific series of cycles. This means that there is no clutch latch point in the print cycle, and therefore a printing operation begins whenever, and as soon as, a print instruction is executed.

Each print instruction can be modified either to take one space or to suppress spacing. The time to space one line is included within the print-cycle time shown in the previous table. Additional spacing and line skipping are performed with a control instruction, and additional time is required for these operations. The computer is interlocked with the printer for 60 μsec; after that time other processing can occur during the space and skip operations. The time required for an immediate skip or space is 45 ms for the first line and 10 ms for each succeeding line. The time required for a delayed skip or space is 10 ms for each line *after* the second line; the time to skip or space for the first two lines is included within the printing cycle time.

TO ↓	FROM 00	09	19	29	39	49	59	69	79	89	99
00	75	88	101	114	127	140	153	167	179	192	204
09	175	188	201	214	227	240	253	267	279	292	304
19	143	156	169	182	195	208	221	235	247	260	272
29	153	166	179	192	205	218	231	245	257	270	282
39	168	181	194	207	220	233	246	260	272	285	297
49	184	197	210	223	236	249	262	276	288	301	313
59	200	213	226	239	252	265	278	292	304	317	329
69	215	228	241	254	267	280	293	307	319	332	344
79	232	245	258	271	284	297	310	324	336	349	361
89	248	261	274	287	300	313	326	340	352	365	377
99	263	276	289	302	315	328	345	355	367	380	392

Figure 93. Cylinder Seek Time

1311 Disk Storage Drive

Timing estimates for disk storage operations are dependent upon three factors: access time, rotational time, and the time actually required to transfer the data between core storage and disk storage.

Access Time

The access time is the time required to "seek" the record, that is, to position the read/write heads over the proper cylinder in the disk pack. When a seek instruction is executed, the read/write heads move from their present position to the "home" position and then move to the disk cylinder specified by the seek instruction. Figure 93 shows actual seek times for cylinder-to-cylinder movement in increments of ten cylinders. The minimum seek time—from cylinder 00 to cylinder 00—is 75 ms; the maximum seek time—from cylinder 99 to cylinder 99—is 392 ms. If the data on a disk pack is located completely in a random sequence throughout all 100 cylinders and the disk addresses specified by the seek instructions are in a random sequence, then the average seek time is 250 ms.

In some 1620 applications, specific kinds of data may be located *within* particular ranges of cylinders. For example, inventory records may be located within cylinders 00-19 and tooling records within cylinders 20-39. When estimating times for applications such as these, the average access time should be based upon the seek times within the specific range of cylinders actually used, as shown in Figure 94.

Cylinders Used	Average Seek Time
0 - 9	132 ms
0 - 19	144
0 - 29	156
0 - 39	169
0 - 49	182
0 - 59	198
0 - 69	210
0 - 79	224
0 - 89	238
0 - 99	250

Figure 94. Average Seek Times

OVERLAPPING SEEK TIME

All of the access time can be overlapped with other processing operations that do not involve disk storage. After a seek instruction is given, the computer is interlocked for just 160 μsec; after this time, the access motion is started and, simultaneously, the computer is released for other processing operations. However, average access times must still be estimated so that the read, write, or check disk operations that follow can be efficiently coordinated into the program.

Rotational Time

Rotational time is the time required for the sector address specified in the instruction to reach the read/write heads after the read instruction has been initiated. Since there is no way of predicting the relative positions of the disk sector addressed and the read/write head *at the time the instruction is executed,* rotational time is based upon an arithmetic average of the minimum and maximum rotational times.

There is a 2 ms delay, known as head select-delay, required for the selection of the appropriate read/write head. After this time, the sector addressed could be read if it were the next sector about to pass under the read/write head; therefore the minimum rotational time would be 2 ms. If, during the head select-delay time, the sector addressed is passing under the read/write head, then a complete rotation would be required before the sector address could be read or written. Therefore, the maximum rotational time would be 42 ms; 40 ms for one rotation and 2 ms head select-delay. The average of these two times, 22 ms, is used for estimating rotational time.

PROCESSING DURING ROTATIONAL TIME

After the appropriate sector has come under the read/write head, it can be assumed that the same sector will be available every 40 ms. Therefore the rotational time that elapses between a read and write instruction or between a write and check instruction could be utilized.

Assume, for example, that a two-sector record is to be read, updated, and then returned to disk storage. The timing chart and block diagram for this operation are shown in Figure 95.

The total time for this operation is 106 ms; the available processing time is 68 ms.

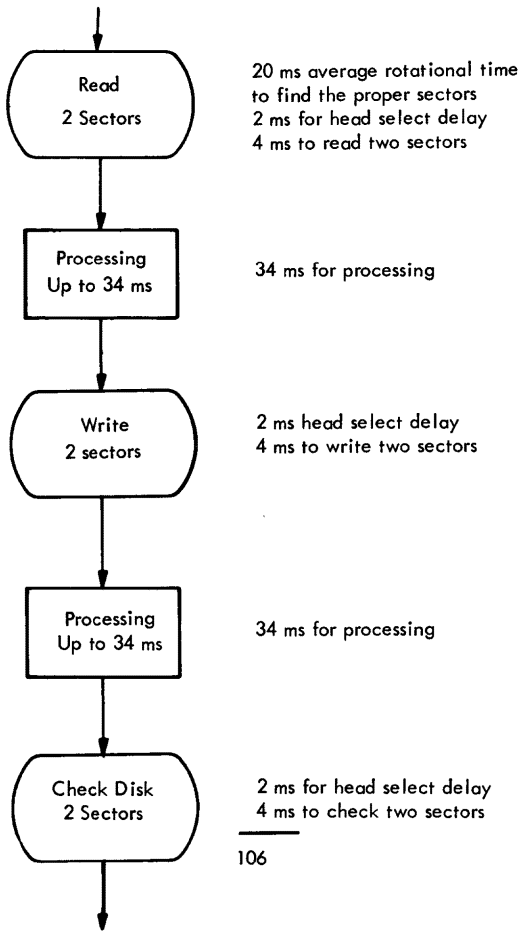


Figure 95. Disk Storage Timing — Two-Sector Record

Processing time is reduced as more sectors are read or written. The timing for a 4-sector operation illustrates this point:

Read	20 ms average rotational time 2 ms head select-delay time 8 ms to read 4 sectors
Process	30 ms processing
Write	2 ms head select-delay time 8 ms to write 4 sectors
Process	30 ms processing
Check Disk	2 ms head select-delay time 8 ms write check
	110 ms Total

The total time in the preceding example is 110 ms, only 4 milliseconds longer than the 2-sector operation. However, total processing time during the rotational time is 60 ms as opposed to 68 ms in the earlier example.

Processing operations performed during rotational time should be carefully timed in order to keep the processing within the allowed time; if not, the throughput time for the program will be increased by one 40-ms revolution for each extension of processing time.

Processing time between a write instruction and a check disk instruction can be used for updating control totals and/or arranging fields for printing.

Data Transfer Time

The data transfer time for a read, write, or check disk operation is 2 ms for each sector processed.

When estimating the time for one of these instructions, only the rotational time and the time to transfer the data are used. The 160 μsec required to execute the instruction in the processing unit is usually ignored because of the wide disparity in the units of measurements — milliseconds to microseconds — and also because the millisecond values are only average times

Console Typewriter

Many applications that require careful timing estimates make limited use of the console typewriter because of its relative slow speed as compared with other input/output units available. The following times are provided for the applications that use the console typewriter as an integral part of the system during processing.

Input

Input speed depends upon the typing speed of the console operator. The speed of the typewriter is 15.5 cps. Input operations are unbuffered.

Output

Output operations consist of either typing data (write instruction) or of the following typewriter control operations,

Return typing element
Tabulate
Space
Backspace
Index

Typing operations are unbuffered. Minimum execution times can be determined by dividing the number of characters to be printed by the speed of the typewriter; 15.5 cps. Actual execution times is a function of the number of characters printed, carriage returns, carriage shifts, etc.

Some of the typewriter control operations can be overlapped with other processing. Figure 96 shows the total time, interlock time, and overlap time for all typewriter control operations.

Operation	Total Time	Processing Unit Interlocked	Available Overlap Time
Return Element	800 ms	124 ms	676 ms
*Tabulate	250	56	194
Space	56	56	none
Backspace	56	56	none
Index	124	124	none

*Approximately 20 positions.

Figure 96. Timing for Typewriter Control Operations

Paper Tape

Paper tape can be used for both input and output to the system. Both operations are unbuffered and the time required to read or punch data can be determined by dividing the number of characters to be processed by the speed of the paper tape unit. The Paper Tape Reader operates at a speed of 150 cps, the Tape Punch operates at a rate of 15 cps.

Plotter

There are two models of the 1627 Plotter available. The difference between the two models is their operating speed. The Model 1 operates at a rate of 300 increments a second and the Model 2 operates at the rate of 200 increments a second. Throughput timing, however, is more directly related to the number and kinds of plotter commands given because some commands require more time than others, as shown in the following diagram:

Command	Model 1 Time	Model 2 Time
0 (Lower pen)	100 ms	100 ms
9 (Raise pen)	100	100
1, 2, 3, 4, 5, 6, 7, and 8	3.3	5

Throughput time is also dependent upon whether the system is operating at Single-Character (overlap) mode or Multiple-Character Record (nonoverlap) mode.

Single-Character (Overlap) Mode

The P-address of a write instruction can refer to any size record, but *by using single-character records,* each successive increment can be computed while the previous increment is being drawn. In this manner, 1620 computation instructions for the next increment or other processing are overlapped with plotting operations.

For example, a Write Numerically instruction and a data record consisting of the digit 5 followed by a record mark (5≡) requires 3.3 ms for the 1627 Model 1, or 5 ms for the 1627 Model 2. However, the 1620 CPU is interlocked for only 200 μsec.

If a second 1620 plotter instruction is attempted before completion of a previous plotter operation, the 1620 waits until the plotter can accept the next command.

Multiple Character Record (Nonoverlap) Mode

When a record in core storage consists of more than one character, only the plotting of the last character can be overlapped with processing. For example, a record consisting or 05555922077779 ≡ is translated as follows: lower pen, move pen four increments in a −y direction, raise pen, move two increments diagonally (+x, +y), lower pen, move drum four increments in the −x (paper-up motion) direction, raise pen.

The computer is interlocked during the plotting of all characters except the last character. The duration is approximately 330 ms with the 1627 Model 1 Plotter, or approximately 350 ms with the 1627 Model 2 Plotter. These timings are computed as follows:

Output Record Characters	1627-1 Plotting Time, ms	1627-2 Plotting Time, ms
0	100	100
5555	13	20
9	100	100
22	7	10
0	100	100
7777	13	20
9	0.3*	0.3*
	333.3	350.3

*Approximately 99.7 ms available for 1620 processing before the 1627 can act on another plotting instruction.

Estimating Process Time

The term "processing time" as it is used here refers to the execution times of instructions performed internally in the Processing Unit rather than instructions involving input/output units.

Appendix A Instruction Summary provides formulas for determining instruction times. They enable the programmer to figure the time required for the processing of data, and, in combination with the input/output operations, enable the programmer to effect efficient placement of instructions. To make realistic timing estimates for processing, it is necessary to consider the individual instructions used and the number of data characters involved in each operation. One approach that can be used is:

1. Develop a general block diagram for the problem to be solved.
2. Define the operation performed in each block.
3. Determine the number of and type of instructions required to accomplish the operations in the block. The length of the data fields should be known.
4. Using the formulas listed in the Instruction Summary, calculate the time required to perform the operations.

Throughput Timing

The Throughput Timing Chart for the 1620 Data Processing System is not only an aid in timing a program, but also is a valuable guide for program development.

Reading and Punching

Figure 97 is a block diagram for a continuous operation of reading a card, performing operations upon

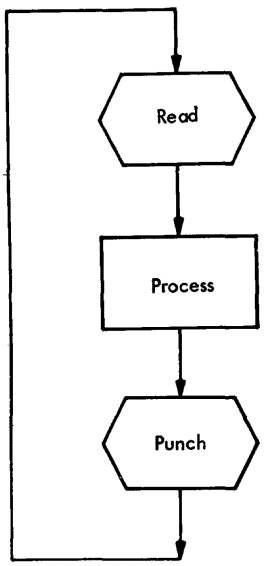


Figure 97. Block Diagram — Reading and Punching

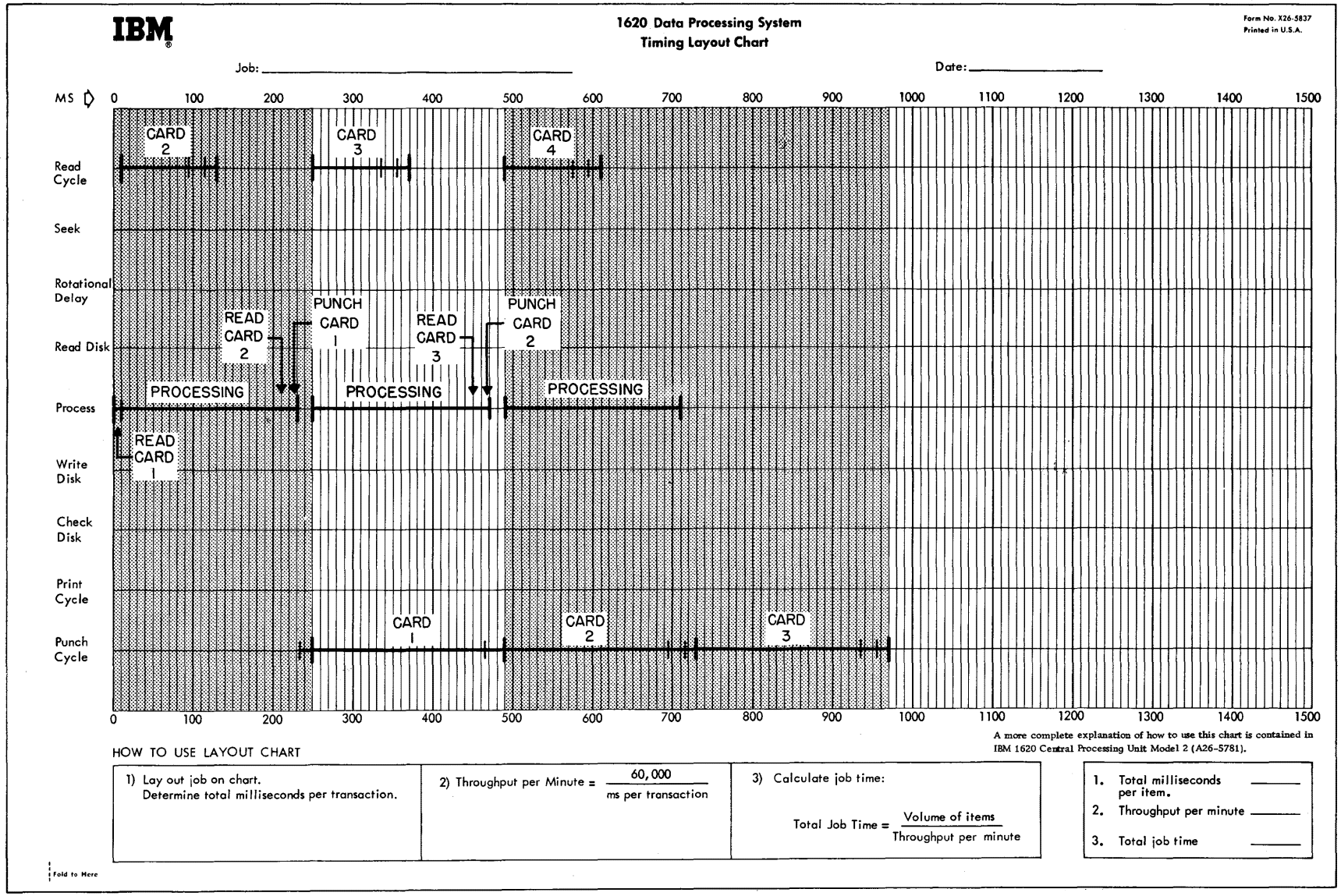


Figure 98. Layout Chart – Reading and Punching

data from the card, and then punching the results. A timing chart is shown in Figure 98. The times shown are for the 1622 Model 2 Card Read-Punch. In this example, a read instruction is given that reads the data from the read buffer (placed in there during loading operations) into core storage, which causes the data from Card 2 to start reading into the buffer at the beginning of the next cycle. The beginning of the next card-read cycle is depicted as starting shortly after the read instruction; however, either the first read cycle or first punch cycle could have started anywhere within the next 120 ms or 240 ms depending upon how much of the previous cycle was completed when the first read or punch instruction was given. For the purposes of illustration, the two cycles are shown as starting at approximately the same time. Approximately 220 ms of processing time is available if a continuous punching speed of 250 cpm is to be maintained. If the next read instruction is given before the clutch latch point of the next cycle (note that one cycle is skipped), then the buffer data is transferred to core storage and the buffer is loading the data from the next card *at the beginning of the next* read cycle, and the processing unit is released to start execution of the punch instruction. If the next punch instruction is executed before the clutch latch point of the next punch cycle, the data from core storage is transferred to the punch buffer, the processing unit is released for processing of the next card, and data will automatically start punching from the punch buffer at the beginning of the next cycle. In this example, the throughput is 250 items a minute with 220 ms of processing time available for each item.

Disk Storage and Printing Operations

Figure 99 is a block diagram of operations for the 1311 Disk Storage Drive and the 1443 Printer Model 1. In this example the 1622 Model 1 Card Reach-Punch is depicted. A timing chart for these operations is shown in Figure 100. This example graphically illustrates the overlapping of operations that is possible with the 1620 Data Processing System. For example, referring to Figure 100, the following operations are being overlapped:

1. Card 3 is being read into the read buffer.
2. The Disk Storage Drive is seeking the record from Card 2.
3. Processing unit is processing for Card 2.
4. Disk Record 2 is read, updated, written back into disk storage and checked.
5. Printer is printing record 1.
6. Punch is punching record 1.

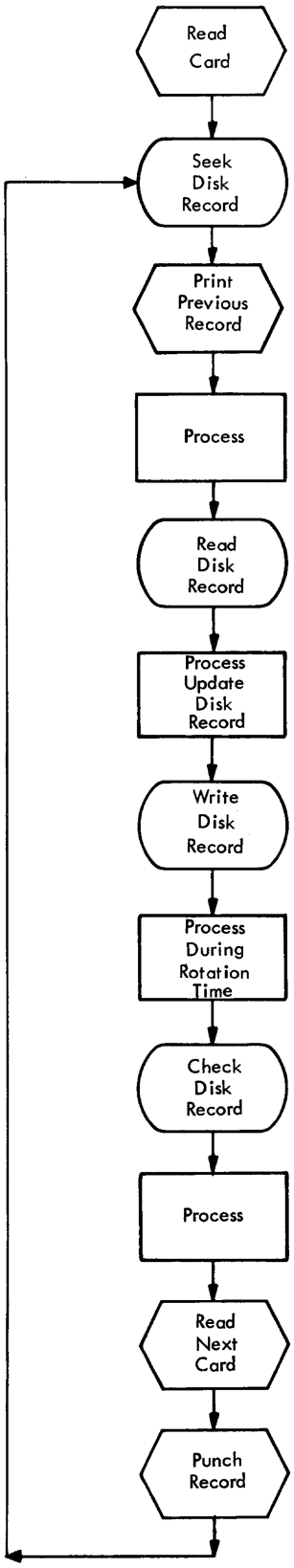


Figure 99. Block Diagram–Disk Storage and Printing Operation

This example also illustrates the point that estimating the throughput time may consist of merely determining the speed of the I/O unit attached to the system, determining what I/O operations can be overlapped, and then calculating the amount of process time available. If the estimated process time is less than the process time available, no other timing considerations may be necessary.

Printing Cycle Greater than Punch Cycle

Figure 101 shows the timing for a program using the 1622 Model 2 and the 1443 Model 2. The 1622-2 punch cycle is 240 ms, but the 1443-2 with the 52-character-set type bar has a print cycle of 250 ms. This example illustrates the significance of the four-tooth punch clutch in the 1622 Card Read-Punch. The print cycle and the processing time extend beyond the 240 ms punch cycle; however, it is possible in this example to give the next punch instruction before the second clutch point has been reached. Thus, the number of milliseconds required for each transaction is increased from 240 ms to 360 ms. The throughput speed is calculated by dividing the number of milliseconds required for each transaction into 60,000. The throughput in this example is 166 transactions per minute; the total milliseconds of processing time is approximately 297 ms.

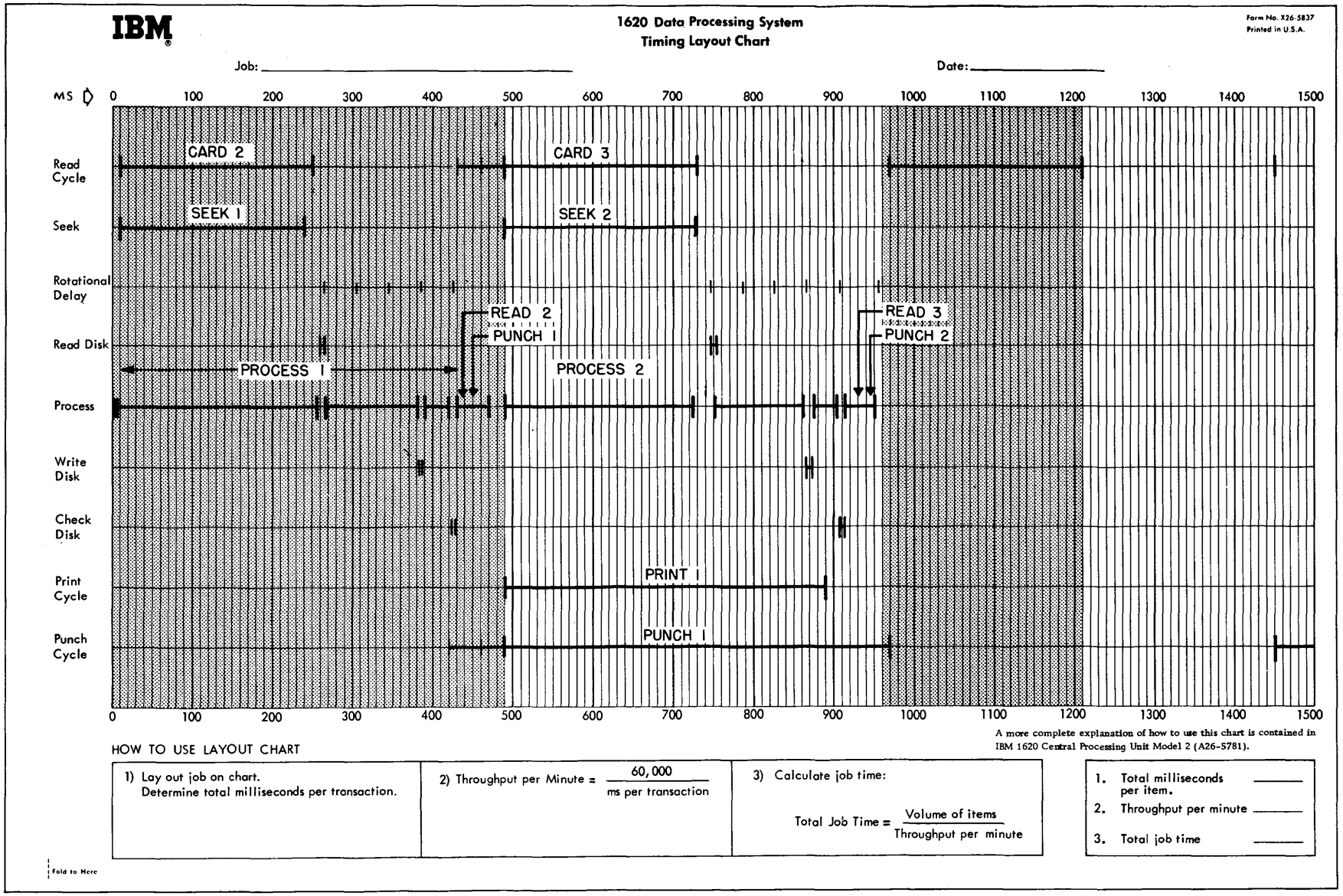


Figure 100. Layout Chart – Disk Storage and Printing Operation

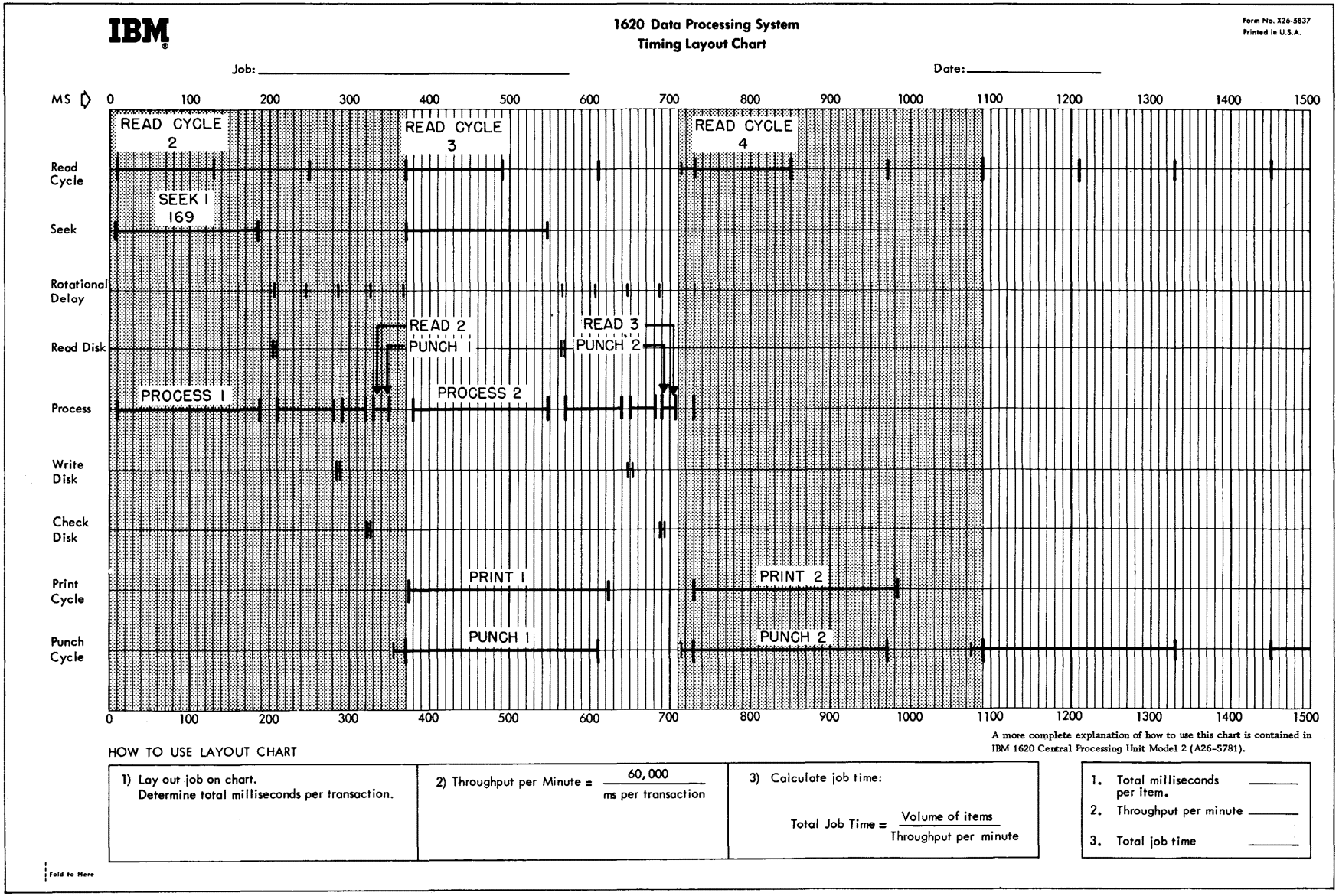


Figure 101. Printing Cycle Greater Than Punch Cycle

Appendix A

Instruction Summary

Instruction	Op Code	Op Code Modifier	SPS Mnemonic	① Timing	Notes	② P Address	② Q Address
Add	21		A	10 (6.5 +.5D_Q + D_P) – Basic Time 10 D_P Recomp time.	If signs initially unlike and numerical value of Q data is greater than P data.	Location of units position of Augend and Result.	Location of units position of Addend.
Add (I)	11		AM	Same as Add		Same as above.	X Q_{11} is units position of Addend.
*AND to Field	93		ANDF	10 (6 + 2D_Q)		Location of units position of P field.	Location of units position of Q field.
Branch	49		B	40		Location of next instruction executed.	X Not used.
*Branch and Load Index Register	65		BLX	140		Location of next instruction executed.	Flags in Q_8 - Q_{11} specify index register to be loaded. Data to be loaded is specified by Q address.
*Branch and Load Index Register (I)	66		BLXM	140		Same as code 65.	X Same as code 65 except the Q portion of the instruction contains the data to be loaded.
*Branch and Modify Index Register	61		BX	10 (6.5 + .5D_Q + L_X)	L_X = length of IX field.	Same as code 65.	Flags in Q_8 - Q_{10} specify index register to be modified. Data to be added to register is specified by Q address.
*Branch and Modify Index Register (I)	62		BXM	140		Same as code 65.	X Same as code 61 except the Q portion of instruction contains the data to be added.
Branch and Select	60		BS	60		Location of next instruction executed.	X Q_{11} = 0 Set Index Registers (XR) OFF Q_{11} = 1 XR Band 1 ON, only Q_{11} = 2 XR Band 2 ON, only Q_{11} = 8 indirect addressing OFF Q_{11} = 9 indirect addressing ON.
*Branch and Store Index Register	67		BSX	140		Location of next instruction executed.	Flags in Q_8 - Q_{10} specify index register to be stored. Contents of register is stored at location specified by Q address.
Branch and Transmit	27		BT	10 (7.5 + 1.5 D_Q)		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 field transmitted.
Branch and Transmit (I)	17		BTM	Same as above.		Same as above.	X Q_{11} is units position of field transmitted.
Branch and Transmit Floating	07		BTFL	10 (9.5 + 1.5L)		Same as above.	Location of units position of Exponent of field transmitted.
Branch Back	42		BB	20		X Not used.	X Not used.
*Branch Conditionally and Modify Index Registers	63		BCX	10 (6.5 + .5D_Q + L_X)	L_X = length of IX field.	Branch: location of next instruction executed. No Branch: not used.	Same as code 61.
*Branch Conditionally and Modify Index Registers (I)	64		BCXM	140		Same as above.	X Same as code 62.
Branch Indicator	46		BI	60		Branch: location of next instruction executed. No Branch: not used.	X Q_8 and Q_9 specify program switch or indicator tested.
Branch No Flag	44		BNF	70		Same as above.	Location tested for flag bit.
*Branch No Group Mark	55		BNG	70		Branch: location of next instruction executed. No Branch: not used.	Location tested for record mark.
Branch No Indicator	47		BNI	60		Same as above.	X Same as code 46.
Branch No Record Mark	45		BNR	70		Same as above.	Location tested for Record Mark character.
*Branch on Bit	90		BBT	70		Branch: location of next instruction executed. No Branch: not used.	Q_8 - Q_{11} specify address of digit to be compared against bit configuration of Q_7 position of instruction.
Branch on Digit	43		BD	70		Branch: location of next instruction executed. No Branch: not used.	Location tested for digit other than zero.
*Branch on Mask	91		BMK	70		Branch: Location of next instruction executed. No Branch: not used.	Q_8 - Q_{11} specify address of digit to be compared against bit configuration of Q_7 position of instruction.
*Check Disk	36	x07x1 x07x3 x07x5 x07x7	CDGN CDN CTGN CTN	10 (6 + 2200 + 200S) Average Time (S = number of sectors)	Check one sector – with WLRC Check one sector – no WLRC Check full track – with WLRC Check full track – no WLRC	Address of Disk Control Field	X Q_8 - Q_9 specify disk storage Q_{11} specifies function performed

Appendix A (Cont'd)

Instruction	Op Code	Op Code Modifier	SPS Mnemonic	① Timing	Notes	② P Address	② Q Address
Clear Flag	33		CF	70		Location of which flag is cleared.	X Not used.
Compare	24		C	10 (8 + 1.5 D_Z) - unlike signs 10 (6.5 + .5D_Q + Dp) - like signs	D_Z = Number of positions compared until a digit other than zero is detected in either field.	Location of units position of field compared with Q field.	Location of units position of field compared with P field.
Compare (I)	14		CM	Same as above.		Same as above.	X Q_{11} is units position of field compared with P field.
*Complement Octal Field	94		CPFL	10 (6 + 2D_Q)		Location of units position of P field.	Location of units position of Q field.
*Control - Printer Space Skip	34		K	 60 † 60 †	Total Time: Refer to instruction description	X Not used.	X Q_8 - Q_9 specifies printer code Q_{10} - Q_{11} specifies function performed.
Control - Typewriter Space Return Element Backspace Index Tabulate	34	 x01x1 x01x2 x01x3 x01x4 x01x8	K SPTY RCTY BKTY IXTY TBTY	 56 ms 124 ms † 56 ms 124 ms 56 ms †	 Total Time: 56 ms Total Time: 800 ms Total Time: 56 ms Total Time: 124 ms Total Time: 250 ms	X Not used	X Q_8 - Q_9 specifies typewriter code. Q_{11} specifies function performed.
*Decimal to Octal conversion	97		DTO	10 (31 + T_Q + 4.125T_Q (T_Q + 1))	T_Q = Position number of table addressed. Average octal digit = 3.5.	Location of leftmost position of field where the result is to be stored.	Address of units digit of the power-of-eight number to be used in first subtraction.
Divide	29		D	10 (6 + 13.5 Q_T + 9.75 $D_V Q_T$)	Average value of quotient digit = 4.5	Location in Product Area of units position of divisor for first subtraction.	Location of units position of Divisor.
Divide (I)	19		DM	Same as above.	Same as above.	Same as above.	X Q_{11} is units position of Divisor
Dump Numerically	35		DN			Core storage location of first character written.	X Q_8 - Q_9 specify output unit.
* Card * Paper Tape * Plotter * Printer Typewriter		x04xx x02xx x02xx x09xx x01xx	DNCD DNPT (None) PRD DNTY	1.7 ms † 15 characters a second 200 μsec † (single character). 2.1 ms † 15.5 characters a second			
*Exclusive OR to Field	95		EORF	10 (6 + 2D_Q)		Location of units position of P field.	Location of units position of Q field.
*Floating Add	01		FADD	10 (15 + 2.2L) average	If signs initially unlike and numerical value of Q data is greater than P data.	Location of units position of Exponent of Augend and Result.	Location of units position of Exponent of Addend.
*Floating Divide	09		FDIV	10 (34.5 + 27L + 9.75 L^2) average		Location of units position of Exponent of Dividend and Quotient.	Location of units position of Exponent of Divisor.
*Floating Multiply	03		FMUL	10 (28 + 3L + 4L_Z + 4L (L - L_Z))	L_Z = number of zeros in mantissa.	Location of units position of Exponent of Multiplicand and Product.	Location of units position of Exponent of Multiplier.
*Floating Shift Left	05		FSL	10 (7 + 2L + 2L')	L' = number of digits mantissa is increased by shift left.	Location of high-order position or resulting field.	Location of units position of field shifted.
*Floating Shift Right	08		FSR	10 (7 + 2L)		Location of units position of field shifted.	Location of units position of resulting field.
*Floating Subtract	02		FSUB	10 (15 + 2.2L) average 10L Recomp time.	If signs initially alike and numerical value of Q data is greater than P data.	Location of units position of Exponent of Minuend and Result.	Location of units position of Exponent of Subtrahend.
Halt	48		H	60		X Not used.	X Not used.
Load Dividend	28		LD	10 (17.5 + 1.5 D_N)	D_N = number of digits, including high-order zeros in the dividend.	Location in Product Area to which units position of Dividend is transmitted.	Location of units position of Dividend.
Load Dividend (I)	18		LDM	Same as above.		Same as above.	X Q_{11} is units position of Dividend.
*Move Address	70		MA	140		Location of units position of address of where data is moved to.	Location of units position of address of where data is moved from.
Move Flag	71		MF	80		Location to which flag is moved.	Location of flag to be moved.
Multiply	23		M	10 (16 + D_Q + 4L_Z + 4D_P (D_Q - L_Z))	L_Z is number of zeros in multiplier field.	Location of units position of multiplicand.	Location of units position of Multiplier.
Multiply (I)	13		MM	Same as above.	Same as above.	Same as above.	X Q_{11} is units position of Multiplier.
No Operation	41		NOP	60		X Not used.	X Not used.
*Octal to Decimal Conversion	96		OTD	10 (28 + D_Q (2D_Q - 1))		Location of units position of conversion table.	Location of units digit of the field to be converted.
*OR to Field	92		ORF	10 (6 + 2D_Q)		Location of units position of P field.	Location of units position of Q field.

Instruction	Op Code	Op Code Modifier	SPS Mnemonic	① Timing	Notes	② P Address	② Q Address
Read Alphamerically	37		RA			P-1: location where zone digit of first character is stored. P: location where numerical digit of first character is stored.	X Q_8 and Q_9 specify input unit.
* Card		x05xx	RACD	1.7 ms †			
* Paper Tape		x03xx	RAPT	150 characters per second			
* Typewriter		x01xx	RATY	Speed of operator			
*Read Binary Paper	37	x33xx	RBPT	150 characters per second.		Location where first character is stored.	X Q_8 - Q_9 specify Paper Tape Reader
*Read Disk	36			10 (6 + 2200 + 200S)		Address of Disk Control Field.	X Q_8 - Q_9 specify disk storage Q_{11} specifies function performed.
		x07x0	RDGN	Average Time	Read Sectors - with WLRC		
		x07x2	RDN		Read Sectors - no WLRC		
		x07x4	RTGN	S = number of sectors.	Read full track - with WLRC		
		x07x6	RTN		Read full track - no WLRC		
Read Numerically	36		RN			Location where first character is stored.	X Q_8 - Q_9 specify input unit.
* Card		x05xx	RNCD	1.7 ms †			
* Paper Tape		x03xx	RNPT	150 characters per second			
Typewriter		x01xx	RNTY	Speed of operator			
*Seek	34	x07x1	SK	160 μsec †	Average access time = 250 ms. Refer to timing section for additional information.	Address of Disk Control Field.	X Q_8 - Q_9 specifies disk storage code. Q_{11} specifies seek operation.
Set Flag	32		SF	70		Location at which flag is set.	X Not used.
Subtract	22		S	10 (6.5 + .5 D_Q + D_P) - Basic Time 10 D_P - Recomp Time	If signs initially alike and numerical value of Q data greater than P data.	Location of units position of minuend and result.	Location of units position of subtrahend.
Subtract (I)	12		SM	Same as above.	Same as above.	Same as above.	X Q_{11} is units position of Subtrahend.
Transmit Digit	25		TD	80		Location to which digit is transmitted.	Location of digit transmitted.
Transmit Digit (I)	15		TDM	80		Same as above.	X Q_{11} is digit transmitted.
Transmit Field	26		TF	10 (6.5 + 1.5 D_Q) Average		Location to which units position of field is transmitted.	Location of units position of field transmitted.
Transmit Field (I)	16		TFM	Same as above.		Same as above	X Q_{11} is units position of field transmitted.
Transmit Floating	06		TFL	10 (9.5 + 1.5 D_Q) Average		Location of units positions of Exponent of resulting field.	Location of units position of Exponent of field transmitted.
Transmit Record	31		TR	10 (6.5 + 1.5 D_Q)		Location to which high-order position of record is transmitted.	Location of high-order position of record transmitted.
Transmit Record No RM	30		TRNM	10 (6.5 + 1.5 D_Q)		Same as above.	Same as above.
Transfer Numeric Fill	73		TNF	10 (6 + D_P)		Location of units position of alphameric field.	Location of units position of numeric field.
Transfer Numeric Strip	72		TNS	10 (6 + Dp)		Same as above.	Same as above.
Write Alphamerically	39		WA			Core storage location of first character written.	X Q_8 - Q_9 specify output unit.
* Card		x04xx	WACD	1.7 ms †			
* Paper Tape		x02xx	WAPT	15 characters a second.			
* Plotter		x02xx	(None)	200 μsec ‡ (single character).			
* Printer		x09xx	PRA	2.1 ms †			
Typewriter		x01xx	WATY	15.5 characters a second.			
*Write Binary Paper Tape	37	x32xx	WBPT	15 characters a second.		Core storage location of first character written.	X Q_8 - Q_9 specify Paper Tape Punch.
*Write Disk	38			10 (6 + 2200 + 200S)		Address of Disk Control Field.	X Q_8 - Q_9 specify disk storage Q_{11} specifies function performed.
		x07x0	WDGN	Average Time	Write Sectors - with WLRC		
		x07x2	WDN		Write Sectors - no WLRC		
		x07x4	WTGN	(S= number of sectors)	Write full track - with WLRC		
		x07x6	WTN		Write full track - no WLRC		
Write Numerically	38		WN			Core storage location of first character written.	X Q_8 - Q_9 specify output unit.
* Card		x04xx	WNCD	1.7 ms †			
* Paper Tape		x02xx	WNPT	15 characters a second.			
* Plotter		x02xx	(None)	200 μsec ‡ (single character).			
* Printer		x09xx	PRN	2.1 ms †			
Typewriter		x01xx	WNTY	15.5 characters a second.			

NOTES:

1. Times consist of total time for instruction, or in some cases (as indicated by a " † ") only the time the CPU is interlocked. (Refer to "Notes"column. Times are provided in microseconds unless otherwise indicated.

2. "X" in the column indicates indirect addressing NOT possible.

D_P = Number of digits, including high-order zeros, inthe 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.
L = Number of digits in mantissa.
D_V = Number of digits, including high-order zeros, in the divisor.

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

Appendix B

Bit Configuration of Decimal Digits

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

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

Table Areas in Core Storage

Address	Area
00000-00099	Console Area
00080-00099	Product Area
00100-00299	Multiply 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. (1710 Control System)

IR-2 Saves return address when any branch and transmit instruction is executed.

IR-3 Contains interrupt address - used in place of IR-1 during interrupt program operation (1710 Control System only).

IR-4 Saves return address when any Branch and Transmit instruction is executed in the Interupt 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. In disk storage operations, used to store and control number of sectors in operation.

MAR Addresses core storage

MBR Receives digits leaving core storage.

MIR Receives digits entering 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 Ep.

Multiplier/Quotient Contains multiplier and quotient digits during multiply and automatic divide operations.

Data Register Decodes Q8 and Q9 digits of BI, BNI and I/O instructions.
Stores partial product digits during multiply instructions.
Stores digits affecting MARS during all I cycles.
Stores one of the digits used in any addition or subtraction.

Appendix C

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)
ǂ	ǂ	08421	08421		0-7-8		C	C8421	ǂ	03421	08421		0-7-8	(none)
ǂ̄	(none)	X8421	CX08421		12-7-8		C41	C8421	ǂ̄	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.

Appendix C (Cont'd)

NUMERIC MODE

Character	Input				Numeric Code	Core Storage		Output				
	Typewriter	Paper Tape	Disk Storage	Card		Zone	Numeric	Typewriter	Paper Tape	Disk Storage	Card	Printer
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 CXO	X82	11-0			F	$\bar{0}$	X	X82	11-0	-
-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	# ♯	# 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

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

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.

1443 Printer Output with Dump Numerically instructions

Character	Core Storage	Output	
		Write Numeric	Dump Numeric
numeric blank	C84	(blank)	@
‡	C82	(none)	‡
♯	C8421	(none)	G
numeric blank flagged	F84	(none)	*
$\bar{‡}$	F82	(none)	W
$\bar{♯}$	F8421	(none)	X

Appendix D

Core Storage Data Resulting from Reading Alphameric Card Data with RN Instruction

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				
	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.

Appendix E 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 a continuous line, it is sometimes necessary 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.. ≠). $\bar{d}$ddddd is the address of the character that will be replaced by the following DATA record mark, and aaaaa is the address into which the following DATA record will be read. 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 originally.

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. 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 following program).
2. Program cards (see their format which follows).
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 1991$\bar{0}$ 31 1990$\bar{5}$ 19920 25 1991$\bar{0}$ 00080 49 1991$\bar{5}$ ≠. 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 1991$\bar{0}$	Save digit where record mark will fall
00024	TR	31 1990$\bar{5}$ 19920	Store the instructions from the card
00036	TD	25 1991$\bar{0}$ 00080	Put the digit back again
00048	B	49 1991$\bar{5}$ ≠	Branch to 00000 or the program

Program Card Format

19901 ↓ xxxxx	19910 ↓ xxxxx	xxxxx	x - - - - - - - - - x ≠
1 - 5	6 - 10	11 - 15	20 - 79 80
Address for card col. 20	Address where ≠ will fall	Blank except last card	Program instructions or constants. Last digit is followed by a ≠

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 (19910) – 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 19905 19920, transmits the five instructions and the record mark from 19920-19980 to the address specified by the data at 19905 (19905 is an indirect address). This address was punched in columns 1-5 of the card.

The fourth instruction, 25 19910 00080, transfers the saved digit back to its original position, and wipes out the record mark.

The fifth instruction, 49 19915, branches the computer back to 00000, and the next card is read. The P address is indirect (19915) – the field at 19915 was loaded by card columns 11-15. These columns are blank for all but the last card. Blank card columns 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	≠

Cards 1 and 2 are standard 5-per-card load cards. Card 3 loads 60 digits of the multiply table.

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 00090 31 00085
00100 25 00090 00060 49 00095

In the last program load card, columns 20-79 must contain the multiply table digits for locations 00100-00159, 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. A26-5781-2

IBM

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]