

The IBM logo, consisting of the letters "IBM" in a bold, sans-serif font, is positioned inside a dark square.

Systems Reference Library

System Operation Reference Manual IBM 1401 Data Processing System IBM 1460 Data Processing System

This publication contains the instruction set for the IBM 1401 and IBM 1460 and the formula for calculating the execution time of each instruction. The operation code for every instruction is given in actual and mnemonic form, with examples of each.

The instructions and applicable timings for the required input-output units are discussed, as are the instructions for the IBM 1406 Storage Unit.

For general information on units attached to 1401 and 1460 systems, refer to the IBM 1401/1460 *Bibliography*, Form A24-1495.

For instructions and applicable timings for attached units, see:

- *Miscellaneous Input-Output Instructions* (1401, 1460), Form A24-3068.
- *Tape Input-Output Instructions* (1401, 1440, 1460), Form A24-3069.
- *Disk Storage Input-Output Instructions* (1401, 1440, 1460), Form A24-3070.

Preface

This publication is a reference text for the IBM 1401 and/or IBM 1460 Data Processing Systems. It provides a detailed explanation of the instructions used by the system to manipulate data. Detailed explanations of the instructions used with the required input-output units attached to the system are also included. The reader should be familiar with the IBM *1401 Systems Summary*, Form A24-1401, or the IBM *1460 Systems Summary*, Form A24-1496, and the various publications on applied programming material, such as Symbolic Programming System (SPS) and *Autocoder*.

The manual is divided into these sections:

- Introduction
- Arithmetic Operations
- Logic Operations
- Data Moving Operations
- Miscellaneous Operations
- Edit Operation
- IBM 1402 Operations
- IBM 1403 Operations
- IBM 1447 Operations
- IBM 1406 Operations

The sections are independent and do not have to be used in the order in which they appear.

The publication is intended for programmers and systems personnel who have a general knowledge of the IBM 1401 and/or IBM 1460 Data Processing Systems and who require a reference text for detailed information. It can also be used as a training aid in the instruction of programmers and operators.

It should be noted that other publications referenced here are, in most cases, prerequisites for a complete understanding of the material presented in this publication.

This publication, Form A24-3067-0, is a minor revision and consolidation of the applicable material from:

A24-1403-6, that includes the following Technical Newsletters:

N24-0028
N24-0030
N24-0037
N24-0043
N24-0052
N24-0054

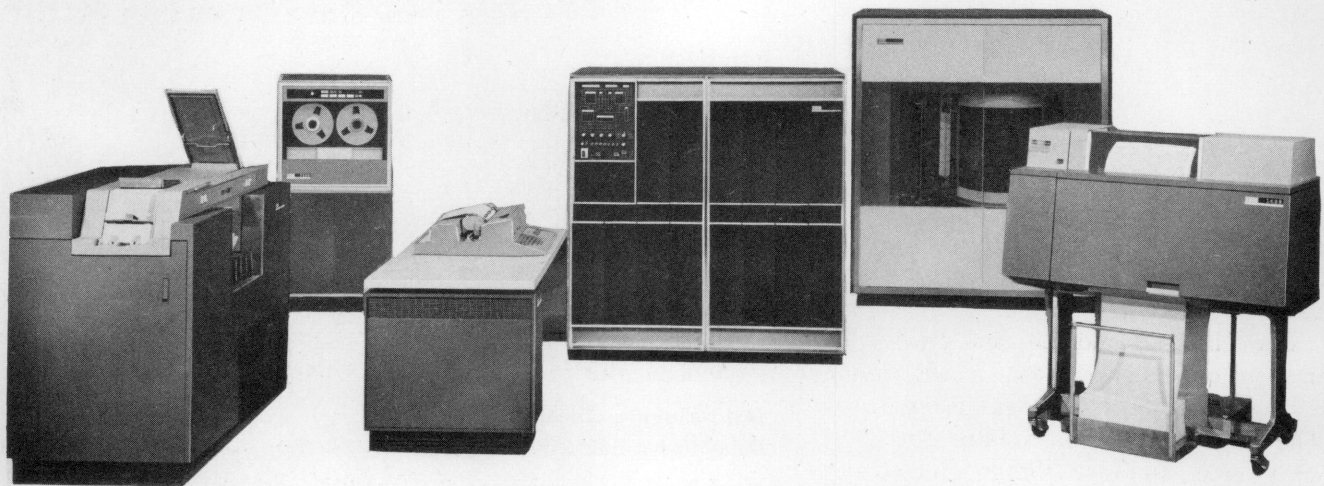
A24-3033-0

The format has been changed to conform to that of the Systems Reference Library; however, the original publications and applicable Technical Newsletters are not obsoleted, but are replaced by this publication. Refer to IBM *1401/1460 Bibliography*, Form A24-1495, for other publications.

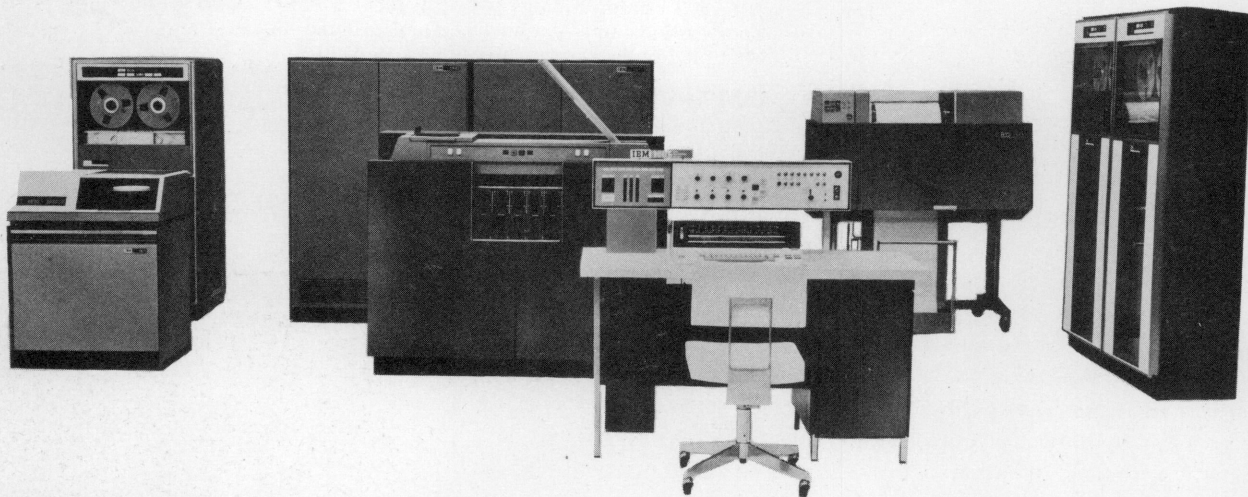
Copies of this and other IBM publications can be obtained through IBM Branch Offices. Address comments regarding the content of this publication to IBM Product Publications, Endicott, New York

Contents

Introduction	5
Processing	7
Stored Program Instructions	9
Addressing	10
Address Modification	12
System Register Operation	13
System Operations	17
Arithmetic Operations	17
Arithmetic Instructions	19
Logic Operations	22
Logic Instructions	23
Data-Moving Operations	26
Data-Moving Instructions	26
Miscellaneous Operations	29
Miscellaneous Instructions	29
Edit Operation	32
IBM 1402 Card Read-Punch Operations	34
IBM 1402 Card Read-Punch Instructions	35
Card Read-Punch Timing	39
IBM 1403 Printer Operations	40
IBM 1403 Printer Instructions	41
Combination Instructions	44
IBM 1403 Printer Timings	46
Combination Input-Output Operations Timing	47
Multiple Printer Output (IBM 1460)	48
IBM 1447 Console Operations	49
Console Instruction Format	50
IBM 1447 Console Instructions	50
Console I/O Printer Timing	51
IBM 1406 Storage Unit Operations	52
Addressing	52
IBM 1406 Storage Unit Instructions	52
Address Modification—Using Modify Address Instruction	53
Appendix	55
Index of 1401/1460 Instructions	56
Index	59



1401



1460

Figure 1. IBM 1401 Data Processing System, IBM 1460 Data Processing System

IBM 1401 Data Processing System IBM 1460 Data Processing System

The IBM 1401 Data Processing System and the compatible, more powerful, IBM 1460 Data Processing System are high-speed, solid-state processing systems having the program flexibility of larger systems.

Both the 1401 and the 1460 provide system configurations for processing unit records, magnetic tape and magnetic disk records, and character-sensed documents. The three basic types of 1401 and 1460 systems are card-, tape-, and disk storage-oriented systems.

A card-oriented 1401, 1460 system is especially useful when large volumes of card documents are used as source data and output data, with particular advantage in applications that involve the re-entry of data.

A tape-oriented 1401, 1460 system, designed to handle magnetic tape, has all the advantages of compact record handling and storage for high-speed data processing.

A disk storage-oriented 1401, 1460 system permits rapid access to large volumes of repetitive data without the necessity of processing large card volumes or sorting tape records.

The basic difference between the IBM 1401 and IBM 1460 is the internal processing speed. The internal processing speed of the 1401 system is 11.5 microseconds per cycle, and 6 microseconds for the 1460 system.

IBM 1401 systems and 1460 systems have the same basic instruction set. The 1401 programs can be run on 1460 systems, and 1460 programs can be run on 1401 systems if the features and I/O units required by the programs are present on the system. Compatibility may be lost in those cases where execution of a program depends on internal cycle speed or the relationship between internal speed and input-output speed. For example, programs employing timing loops (such as in IBM 1419 operations) and programs optimized to read-release and punch-release timing restrictions must be re-evaluated to determine possible limitations.

The Stored Program

The IBM 1401 and the IBM 1460 perform their functions by executing a series of instructions at high speed. A particular set of instructions, designed to solve a specific problem, is known as a *program*. Because these systems store their instructions internally, they are called *stored program* systems.

The 1401 and 1460 normally execute instructions sequentially. But sometimes it is necessary to skip over

a particular group of instructions, or otherwise change the sequence of the program. Branch instructions provided in the systems make it possible to alter the program and take the next instruction from another area of the stored program. This feature also makes it possible to repeat an instruction, or group of instructions, as often as desired.

A series of programmed tests determines the logical path of the program. These tests are made at various points in the program to control the course of program step execution for specific conditions that can arise during processing.

Processing Units

The processing units in each system are the controlling centers of the entire data processing system. The IBM 1401 Processing Unit (Figure 2) serves this function for the 1401 system and the IBM 1441 Processing Unit (Figure 3) serves this function for the 1460 system. Each processing unit can be divided into two parts:

- The arithmetic-logical unit
- The control section

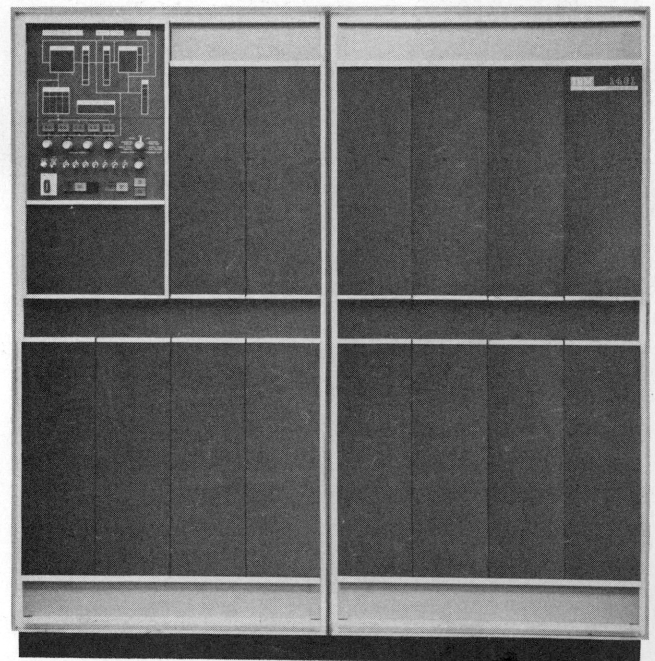


Figure 2. IBM 1401 Processing Unit

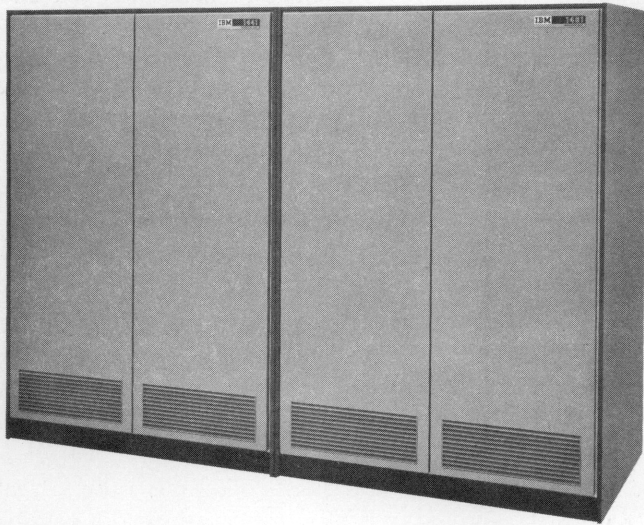


Figure 3. IBM 1441 Processing Unit and IBM 1461 Input-Output Control

The arithmetic-logical unit performs such operations as addition, subtraction, shifting, transferring, comparing, and storing. By adding the multiply-divide special feature the systems can perform direct multiplication and division. The unit also has logical ability—the ability to test various conditions encountered during processing and to take action called for by the result.

The control section directs and coordinates the entire system as a single multipurpose machine. These functions involve controlling the input-output units (1401 system only) and the arithmetic-logical operation of the processing unit, and transferring data to and from storage, within given design limits. This section directs the system according to the procedure originated by its human operators.

In the 1460 system, the controlling circuitry for the input-output units is contained in one of the models of the IBM 1461 Input-Output Control (Figure 3).

The capacity of the IBM 1401 Processing Unit is 1400; 2000; 4000; 8000; 12,000; or 16,000 alphameric characters of 8-bit core storage and the associated circuitry. The capacity of the IBM 1441 Processing Unit is 8,000; 12,000; or 16,000 alphameric characters of 8-bit core storage and the associated circuitry. The eight bits used in 8-bit core storage are six bits for binary coded decimal, a check bit, and an eighth bit for field definition.

MAGNETIC-CORE STORAGE

Both the IBM 1401 and 1460 Data Processing Systems use magnetic-core storage for storing instructions and data (Figure 4). All data in core storage is instantly available, and the special design of the core-storage unit makes each position individually *addressable*. This

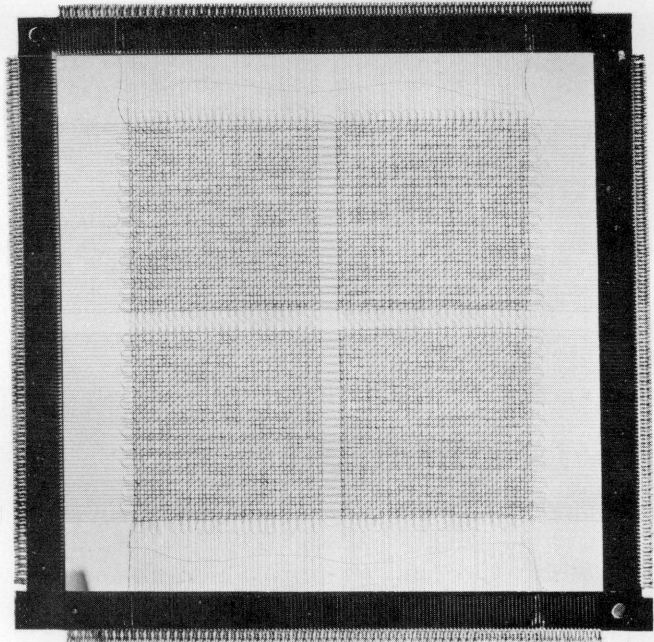


Figure 4. Magnetic Core Storage

means an instruction can designate the exact storage locations that contain the data needed for that step.

The physical make-up of each core-storage location makes it possible for the IBM 1401 and the IBM 1460 to perform arithmetic operations directly in the storage area. (This is called *add-to-storage* logic.)

Language

In the punched-card area of data processing, the language of the machine consists of holes punched in a card. As data processing needs increase, the basic card language remains the same. But in the transition from unit-record systems to either the IBM 1401 Data Processing System or the IBM 1460 Data Processing System, and from there to computer systems, another faster, more flexible machine language emerges.

Just as each digit, letter in the alphabet, or special character is coded into a card as a punched hole or a combination of punched holes, it is coded into magnetic storage as a pattern of magnetized spots.

Many different code patterns can be set up. The internal code used in both the IBM 1401 and IBM 1460 Data Processing Systems is called *binary-coded decimal* (Figure 5). All data and instructions are translated into this code as they are stored. No matter how information is introduced into the system (most commonly by means of punched cards or magnetic tape), the binary-coded-decimal code is used in all data flow and processing from that point on, until it is translated into printed output as reports and documents are written, or converted to punched-card code, for punched-

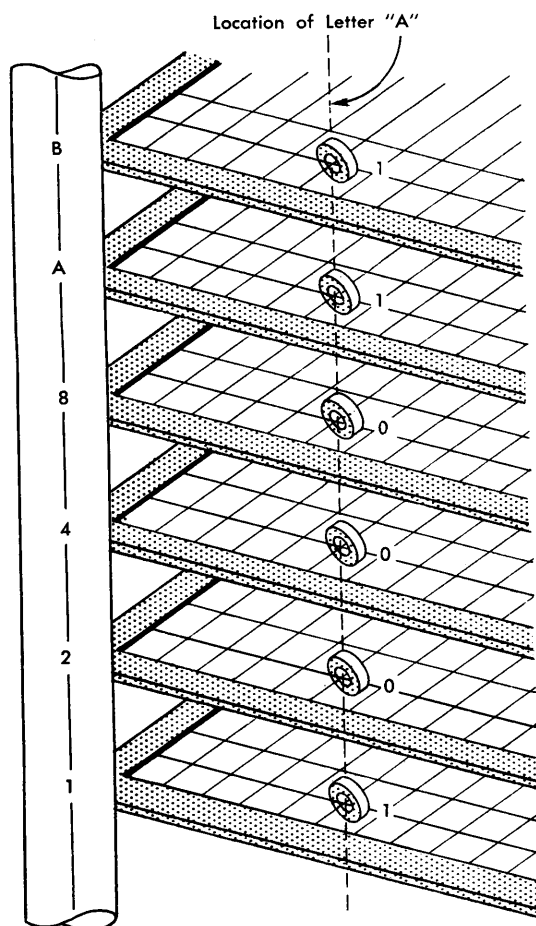


Figure 5. The Letter A Represented in Binary-Coded Form in Core Storage

card output. Converting input data to the 1401 or 1460 internal code, and subsequently reconvert, is completely automatic.

Processing

Processing is the manipulation of data from the time it is introduced to the system as input until the desired results are ready for output. The following functions are performed in the processing unit.

Logic

The logic function of any kind of data processing system is the ability to execute program steps; but even more, the ability to evaluate conditions and select alternative program steps on the basis of those conditions.

In unit-record equipment, an example of this logic is selector-controlled operations based on an X-punch or No X-punch, or based on a positive or negative value, or perhaps based on a comparison of control numbers in a given card field.

Similarly, the logic functions of both the 1401 and 1460 systems control comparisons, branching (alternative decisions similar in concept to selector-controlled procedures), move and load operations (transfer of data or instructions), and the general ability to perform a complicated set of program steps with necessary variations.

Arithmetic

The IBM 1401 and IBM 1441 Processing Units can add, subtract, multiply, and divide. Multiplication and division can be accomplished in any 1401 or 1460 system, by programmed sub-routines. When the extent of the calculations might otherwise limit the operation, a special multiply-divide feature is available.

Editing

As the term implies, editing adds significance to output data by punctuating and inserting special characters and symbols. Both the IBM 1401 and IBM 1460 have a unique ability to perform this function, automatically, with simple program instructions.

Internal Checking

Advanced circuit design is built into the 1401 and 1460 to assure accurate results. Self-checking within the machine consists of *parity*, *validity*, and *hole count*.

PARITY CHECKING

The IBM 1401 and IBM 1460 check characters at various locations in the system for odd-bit configurations. The 6-bit, binary-coded-decimal internal language used by both the 1401 and 1460 also has a check bit for odd-bit checking purposes, and a word mark bit. The check bit is added to all characters that would otherwise have an even number of bits.

Example: A character P has a binary-coded decimal equivalent of B 4 2 1. The check bit is added to give this character an odd number of bits (C B 4 2 1).

If the character has a word mark associated with it, the word mark is included in the test for odd-bit parity.

Example: If the character P has a word mark, the check bit is not added because the bit configuration is odd (WM B 4 2 1).

Whenever a parity error occurs, a console light turns on, indicating the place where the error occurred.

VALIDITY CHECKING

A bit configuration that does not comprise a valid character causes a validity error in the system. For example, an invalid character passes a parity check because it contains an odd number of bits but does not pass a validity check.

A validity check is performed on each character as it is read into the system by the card reader. An invalid character can get into core storage, but the validity-check circuits detect it and cause the system to stop. The validity light on the card reader turns on to indicate the error.

Four types of address validity checking are performed by the system. The operations, and when they are performed, follow:

1. *Checking for a core-storage address greater than the installed core-storage capacity.* The units position of an address on from 4 to 12 thousand positions of core storage are checked for the proper A-, B-bit configuration. This check is performed when the output of the B-register goes to the storage-address register.
2. *Effective address checking* is divided into three tests that occur whenever core storage is addressed. The three tests are:
 - a. Incorrect parity
 - b. Invalid address character
 - c. A check of the hundreds and units positions of an address for various core-storage sizes. For example, a 1400-position 1401 system is checked for addresses between 1400 and 4000. A 2000-position 1401 system is checked for addresses between 2000 and 4000. A 12,000-position 1401 or 1460 system is checked for addresses between 12000 and 16000. If any of these conditions are found, a validity check occurs and the system stops.
3. *Index checking* is performed during an indexing operation to check for modification to an address in excess of installed core-storage capacity.
4. *End-around check* is made at all times except for three special operations. The modification of the low-order position of core storage by -1 , except during a CLEAR operation, or the modification of the high-order position of core storage by $+1$, except during STORAGE SCAN and STORAGE PRINT OUT operations, causes invalid operation and a system stop.

HOLE-COUNT CHECK

Reliability is further assured in the 1401 and 1460 systems by the *hole-count* feature of the IBM 1402 Card Read-Punch. With this feature, the total number of holes read in each column of a card at the read-check station is compared with the total number of holes read from the same column of the same card as it passes the read station. Hole-count checking is also performed in the punch-feed side. A count of the total number of holes to be punched in each column of the card at the punch station is retained internally for one punch-feed cycle. Another column-by-column hole

count is taken as this same card passes the punch-check station, and the two counts are compared.

If a hole-count error (unequal comparison) occurs in either the read or punch side, the system stops and indicates the unit involved. The operator can determine where the error occurred by setting the mode switch to STORAGE SCAN and pressing the start key. The scan stops at the storage address of the column in error.

Variable Word Length

Stored programming involves the concept of *words*. A 1401 and 1460 word can be a single character, or a group of characters that represent a complete unit of information. Because the words are not limited to a specific number of storage positions, and because each position of core storage is addressable, each word occupies only that number of core-storage locations actually needed for an instruction or data field.

WORD MARKS

The use of the variable-length instruction and data format requires a method of determining the instruction and data-word length. This identification is provided by a word mark. Word marks are illustrated by underlining the characters with which they are associated.

The word mark serves several functions:

1. Indicates the beginning of an instruction.
2. Defines the size of a data word.
3. Signals the end of execution of an instruction.

The rules governing the use of word marks are:

1. Predetermined locations for word marks are assigned in planning the program. These predetermined word marks are normally expected to remain in these locations throughout the complete program. The word marks are set into storage locations by a loading routine.
2. Word marks are not moved with data during processing, except when a *load* instruction (see *Move and Load*) is used.
3. For an arithmetic operation, the *B-field* must have a defining word mark, and the *A-field* must have a word mark only when it is shorter than the B-field.
4. A load instruction moves the word mark and data from the A-field to the B-field, and clears any other word marks in the designated B-field, up to the length of the A-field.
5. When moving data from one location to another, only one of the fields need have a defining word mark, because the *move* instruction implies that both fields are the same length.
6. A word mark must be associated with the high-order character (operation code) of every instruction.

7. The 4-character **BRANCH UNCONDITIONAL** instruction, the 7-character **SET WORD MARK**, and **CLEAR STORAGE AND BRANCH** instructions are the only instructions that can be followed by a blank without a word mark. All other instructions must be followed by a word mark.
8. A word mark must be set in the storage position at the immediate right of the last character of the last instruction in the program.

Two operation codes are provided for setting and clearing word marks during program execution.

Stored Program Instructions

All machine functions are initiated by instructions from the stored program. Both systems use the variable-word-length concept and the length of an instruction can vary from one to eight characters, depending on the operation to be performed.

Instruction Format

<i>OP Code</i>	<i>A- or I-address</i>	<i>B-address</i>	<i>d-character</i>
<u>X</u>	XXX	XXX	X

Mnemonic. This is the mnemonic operation code used by the (SPS or) *Autocoder* processor program(s) to designate the actual machine operation code.

OP Code. This is always a single character that defines the basic operation to be performed. A word mark is always associated with the operation code position of an instruction.

A-Address. This always consists of three characters. It can identify the units position of the A-field, or it can be used to select a special unit or feature (tape

unit, disk storage unit, IBM 1419 Magnetic Character Reader, etc.).

I-Address. Instructions that can cause program branches use the I-address to specify the location of the next instruction to be executed if a branch occurs.

B-Address. This is a 3-character storage address that identifies the B-field. It usually addresses the units position of the B-field, but in some operations (such as tape read and write) it specifies the high-order position of a record-storage area.

d-Character. The d-character is used to modify an operation code. It is a single alphabetic, numerical, or special character, positioned as the last character of an instruction.

Examples of the six combinations possible in variable-length instructions are shown in Figure 6.

Instruction Descriptions

Specific instructions have been described in a standard format:

Title: This is the description of the instruction.

Instruction Format: This is the format of the particular instruction described. The mnemonic operation code used in the IBM 1401 (1460) Symbolic Programming System (SPS) is shown and the mnemonic for IBM 1401 (1460) *Autocoder* (A) is given, if it is different from the SPS mnemonic.

Function: This is the function of the instruction.

Word Marks: This is the effect of word marks with regard to data fields.

Timing: This is the formula to be used in calculating the timing of the instruction. Key to abbreviations used in formulas is shown in Figure 7.

NUMBER OF POSITIONS	OPERATION	INSTRUCTION FORMAT			
1	READ A CARD	OP code <u>1</u>			
2	SELECT STACKER	Op code <u>K</u>	d-character 2		
4	BRANCH	Op code <u>B</u>	I-address 400		
5	BRANCH IF INDICATOR ON	Op code <u>B</u>	I-address 625	d-character /	
7	ADD	Op code <u>A</u>	A-address 072	B-address 423	
8	BRANCH IF CHARACTER EQUAL	Op code <u>B</u>	I-address 650	B-address 080	d-character 4

Figure 6. IBM 1401 Instruction Formats

Key to abbreviations used in formula:	
L_A	= Length of the A-field
L_B	= Length of the B-field
L_C	= Length of multiplicand field
L	= Length of argument field
L_I	= Length of instruction
L_M	= Length of multiplier field
L_Q	= Length of quotient field
L_R	= Length of divisor field
L_S	= Length of sector
L_S	= Number of significant digits in divisor (Excludes high-order 0's and blanks)
L_W	= Length of A- or B-field, whichever is shorter
L_X	= Number of characters to be cleared
L_Y	= Number of characters back to right-most "0" in control field
L_Z	= Number of 0's inserted in a field
I/O	= Timing for input or output cycle
F_m	= Forms movement times. Allow 20 ms for first space, plus 5 ms for each additional space
N	= System processing cycle time (.0115 ms for 1401; .006 ms for 1460)
N_S	= Number of sectors
S_S	= Size of sectors
T_m	= Tape movement times
Σ	= Number of fields included in an operation

Figure 7. Timing Formula Coding

Notes: These are special notations or additional information that pertain to the operation.

Address Registers After Operation: The contents of the address registers are represented by the codes described in Figure 8.

Example: A practical application of the instruction is described and shown as labels for *Autocoder* and SPS

ABBREVIATION	MEANING
A	A-address of the instruction
B	B-address of the instruction
NSI	Address of the next sequential instruction
BI	Address of the next instruction if a branch occurs
L_A	The number of characters in the A-field
L_B	The number of characters in the B-field
L_D	The number of characters in a disk record
L_W	The number of characters in the A- or B-field, whichever is smaller
Ap	The previous setting of the A-address register
Bp	The previous setting of the B-address register
dpp	The d-character and the tens and units positions of the previous register setting
dbi	The d-character and the tens and units positions of the branch instruction
dbb	The d-character and blank in the units and tens position

Figure 8. Address Registers After Operation Coding

(Symbolic Programming System) languages. With the label is the actual machine address in parentheses. It is not necessary for the programmer to know the actual address of a label when writing the program. The processor program assigns the actual address during the program assembly.

Assembled Instruction: This is the actual machine language instruction that is assembled by the SPS or *Autocoder* processor programs from the symbolic entries shown in the example.

Addressing

Instructions and data used for processing in the 1401 and the 1460 are kept in core storage. Each core-storage position in the area has its own unique address.

The IBM 1401 Processing Unit is available in six different capacities of core storage: 1400; 2000; 4000; 8000; 12,000; and 16,000 positions.

The IBM 1441 Processing Unit for the 1460 system is available in three different core-storage capacities: 8,000; 12,000; and 16,000 positions. Each position of core storage is identified by a 3-digit address. To reflect addresses over 999, zone bits are placed over the hundreds and/or the units position of a 3-digit number in various combinations.

Storage addresses 0000-3999 have zone bits over the hundreds position, and no zone bits over the units position of the 3-character address.

Storage addresses 4000-7999 have zone bits over the hundreds position, and A-bits (0 zone) over the units position of the 3-character address.

Storage addresses 8000-11,999 have zone bits over the hundreds position, and B-bits (11 zone) over the units position of the 3-character address.

Storage addresses 12,000-15,999 have zone bits over the hundreds position, and A- and B-bits (12 zone) over the units position of the 3-character address. Figure 9 is a chart of the addressing system.

The system addresses core-storage locations by assigning a digit value to each bit that appears over the hundreds and units positions of the 3-character address.

Bit and Location	Digit Value
A-bit over hundreds position	1
B-bit over hundreds position	2
A-bit over units position	4
B-bit over units position	8

The machine adds the assigned digit values of the hundreds and units positions to determine the thousand block of storage addressed.

$$\begin{array}{rclclcl}
 \text{P99} & = & \begin{array}{c} A \\ B \\ \hline 099 \end{array} & = & \begin{array}{c} 1 \\ 2 \\ \hline 099 \end{array} & = & 3099 \\
 \text{I5R} & = & \begin{array}{cc} A & B \\ B & B \\ \hline 9 & 5 & 9 \end{array} & = & \begin{array}{cc} 1 & 8 \\ 2 & 8 \\ \hline 9 & 5 & 9 \end{array} & = & 11,959
 \end{array}$$

ACTUAL ADDRESSES	ZONE BITS OVER HUNDREDS POSITION	ZONE BITS OVER UNITS POSITION	3-CHARACTER ADDRESSES
0000 to 0999 1000 to 1999 2000 to 2999 3000 to 3999	No Zone Bits A-Bit (Zero-Zone) B-Bit (11-Zone) AB-Bits (12-Zone)	No Zone Bits No Zone Bits No Zone Bits No Zone Bits	000 to 999 ≠00 to Z99 !00 to R99 ?00 to I99
4000 to 4999 5000 to 5999 6000 to 6999 7000 to 7999	No Zone Bits A-Bit (Zero-Zone) B-Bit (11-Zone) AB-Bits (12-Zone)	A-Bit (Zero-Zone) A-Bit (Zero-Zone) A-Bit (Zero-Zone) A-Bit (Zero-Zone)	00≠ to 99Z ≠0≠ to Z9Z !0≠ to R9Z ?0≠ to I9Z
8000 to 8999 9000 to 9999 10000 to 10999 11000 to 11999	No Zone Bits A-Bit (Zero-Zone) B-Bit (11-Zone) AB-Bits (12-Zone)	B-Bit (11-Zone) B-Bit (11-Zone) B-Bit (11-Zone) B-Bit (11-Zone)	00! to 99R ≠0! to Z9R !0! to R9R ?0! to I9R
12000 to 12999 13000 to 13999 14000 to 14999 15000 to 15999	No Zone Bits A-Bit (Zero-Zone) B-Bit (11-Zone) AB-Bits (12-Zone)	AB-Bits (12-Zone) AB-Bits (12-Zone) AB-Bits (12-Zone) AB-Bits (12-Zone)	00? to 99I ≠0? to Z9I !0? to R9I ?0? to I9I

Figure 9. Addressing System

Data Field Addressing

A data field in core storage is addressed by specifying the low-order (units) position of the field in the A- or B-address of the instruction. The data field is read from right to left until a word mark in the high-order position is sensed.

Instruction Addressing

An instruction in core storage is addressed by giving the high-order (operation code) position of the instruction. All operation codes must have a word mark. (This word mark is normally set by the load routine when the instructions are loaded.) The machine reads an instruction from left to right until it senses the word mark associated with the next sequential instruction (see item 7 under *Word Marks* for exceptions). The final instruction in the program must have a word mark set at the right of its low-order position.

Example: Instruction address 400 (Figure 10) contains the operation code for the following instruction:

Op Code	A-address	B-address
<u>A</u>	542	560

When this instruction is executed, the data in the A-field is added to the data in the B-field:

```

0025347
04601231
04626578

```

The result is stored in the B-field.

Input-Output Storage Assignments

Three areas of storage are reserved for input and output data. Storage positions 001 through 080, are reserved

Instruction addressed by high-order position

STORAGE ADDRESS	400	401	402	403	404	405	406	407 (NSI)
INSTRUCTION	<u>A</u>	5	4	2	5	6	0	WM Op code

The word mark associated with the next sequential instruction (NSI) stops the reading of this instruction.

STORAGE ADDRESS	536	537	538	539	540	541	542	543
DATA	<u>0</u>	0	2	5	3	4	7	<u>8</u>

Word mark identifies high-order position of A-field.

STORAGE ADDRESS	553	554	555	556	557	558	559	560	561
DATA	<u>0</u>	4	6	0	1	2	3	1	<u>4</u>

Word mark identifies high-order position of B-field.

Figure 10. Data and Instruction Addressing

for the information from the 80 columns of the card. The second area of storage, positions 101 through 180, is reserved for assembling data to be punched. Positions 000 and 100 should not be used. Data stored in position 000 before a card read operation is replaced by CAB bits at the end of the read operation. Data stored

in position 100 before a punch operation is replaced by C82 bits at the end of the punch operation. The third area of storage, positions 201 through 300 or 332, is reserved for assembling characters to be printed. Positions 81 through 99, and 181 through 200, are available for normal storage use. When the reserved areas are not being used as specified, they can be used for other storage operations (Figure 11).

Address Modification

It becomes necessary in some 1401 and 1460 programs to perform the same operations repetitively, with a change only in the A- or B-address. Changing an address while retaining the rest of the instruction is called *address modification*. Address modification can decrease the number of program steps and the number of storage requirements. 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.

There are two basic methods of address modification. The first method does not require the indexing feature. The second method makes use of the indexing, which is a special feature for the 1401 and 1460 systems.

Address Modification Without Indexing

Address modification uses the A- and B-bit accumulation that can occur in the hundreds and units positions of a field. This accumulation is discussed in connection with overflow indication in the *Arithmetic Operations* section of this publication.

USING MODULUS 4 ARITHMETIC

For 1401 systems of 4,000 storage positions or less, A- and B-bit accumulation should occur only in the hundreds position, and is based on modulus 4 arithmetic. To understand how a modulus 4 arithmetic operation is accomplished, let us assign digital values to the A- and B-bit configurations:

- No A, No B = 0
- A = 1
- B = 2
- AB = 3

In a modulus 4 system, the highest digit is 3. Values in excess of three are equal to that value minus four. For example, 5 is a digit 1. In this system, only two factors can be accumulated at a time (Figure 12). Digit values in the high-order position of a field accumulate in the normal manner. In 1401 systems of 4,000 core-storage positions or less, it is assumed that there is a word mark in the high-order position of the address being modified.

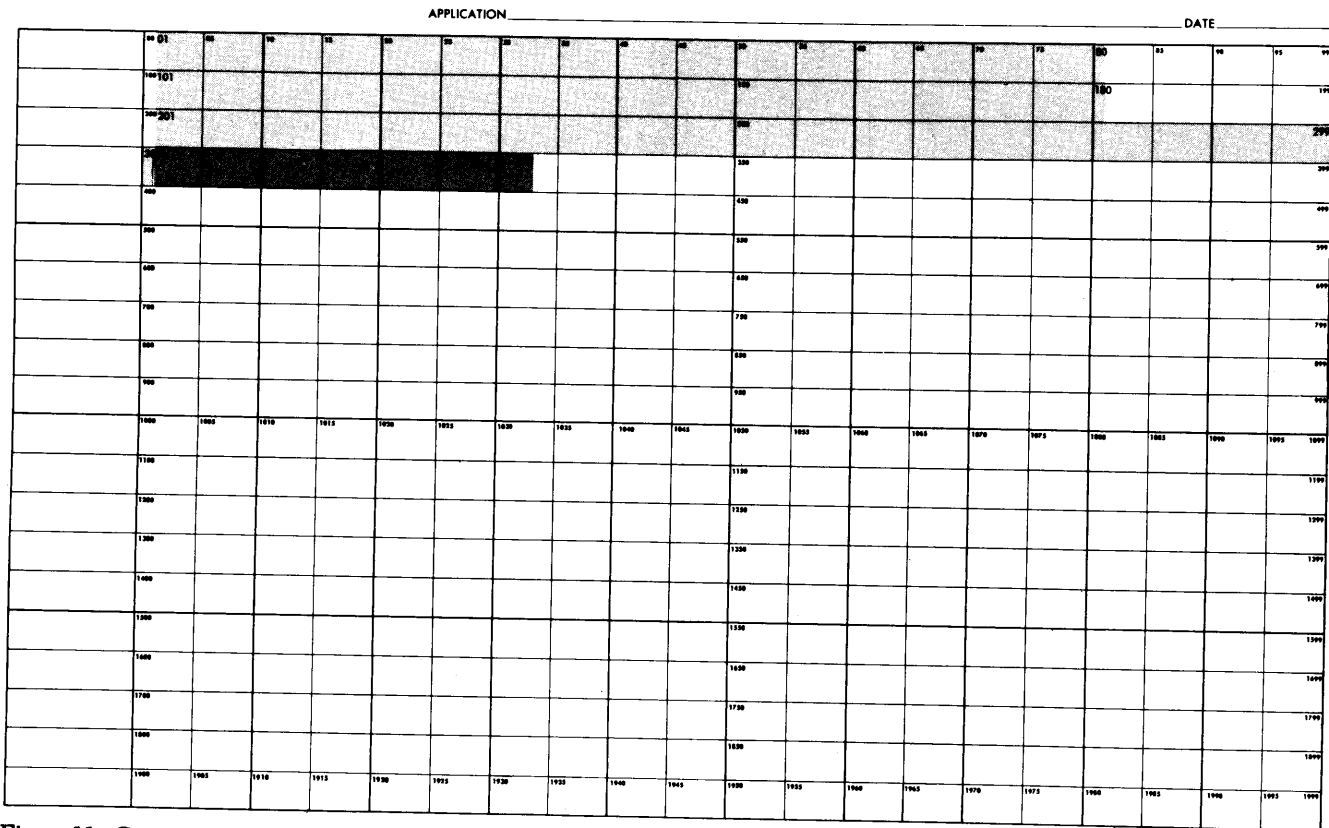


Figure 11. Core Storage Layout Chart

$A + A$	$= B$	or	$1 + 1 = 2$
$A + B$	$= AB$	or	$1 + 2 = 3$
$B + B$	$= \text{NoANoB}$	or	$2 + 2 = 0$
$A + AB$	$= \text{NoANoB}$	or	$1 + 3 = 0$
$A + \text{NoANoB}$	$= A$	or	$1 + 0 = 1$
$B + AB$	$= A$	or	$2 + 3 = 1$
$B + \text{NoANoB}$	$= B$	or	$2 + 0 = 2$
$AB + AB$	$= B$	or	$3 + 3 = 2$

Figure 12. A-Bit and B-Bit Values

Modification to a higher address in 000-999 address range is:

Increase address 472 by 345.

$$472 + 345 = 817$$

Modification to an address greater than 1000 is:

Increase address 912 by 314.

$$912 + 314 = 1226 \text{ or } S26$$

$S = A2$ (overflow in high-order position sets an A-bit using modulus 4 arithmetic and turns on the arithmetic overflow indicator).

Increase address 1754 (X54) by 1204 (S04).

$$X54 + S04 = R58$$

$$X = (A7)$$

$$S = (A2)$$

Using the rules of modulus 4 arithmetic, $A + A = B$ -bit, the new address is:

958 with a B-bit over the high-order position
($B9 = R$) or $R58$ (2958).

To decrease an address, a different means must be used. Modulus 4 arithmetic operates for addition only. Decreasing an address requires the addition of a complement, rather than doing a conventional subtract operation.

In 1401 systems of 4,000 core-storage positions or less, the 16,000's complement of the decrement is added to the address to be modified.

Decrease address 879 by 148.

$$16,000 - 148 = 15,852 \text{ (H5B)} = \text{complement}$$

$$879 + \text{H5B} = 731 \text{ (with arithmetic overflow)}$$

$$H = \text{BA8}$$

$$B = \text{BA2}$$

Using the modulus 16 rules, the arithmetic overflow adds an A-bit in the hundreds position (the hundreds position already contains A- and B-bits, and the units position contains A- and B-bits, the combination of which indicates a 15,000 to 15,999 block address). The addition of the A-bit increases the value of the zone bits to 16, which, according to modulus 16 rules has a new address value of 0 (000-999 block address). Therefore, the new address is 731, and the overflow indicator is ON.

System Register Operation

The IBM 1401 and 1460 Data Processing Systems can operate on and process data to produce a desired result by executing a series of instructions at high speed. A series of instructions designed to solve a problem is known as a *program*. Because these instructions are retained in core storage, it is more properly called a stored program.

The processing unit must interpret an instruction and perform the function prescribed by the instruction. To do this, various devices are used that are capable of receiving information, storing it, and transferring it as directed by control circuits. These devices are known as *registers*. The 1401 and 1460 have seven registers; four are address registers and three are character registers (Figure 13).

Address Registers

There are four address registers in the IBM 1401 and IBM 1441 Processing Units (without the multiply-divide and the processing overlap special features). One register controls the program sequence, and two other registers control the data transfer from one storage location to another. The fourth register specifies which storage location is active during a particular storage cycle.

I-ADDRESS REGISTER

The I- (Instruction) address register always contains the storage location of the next instruction character to be used by the stored program. The number in this register is increased by one as the instruction is read from left to right.

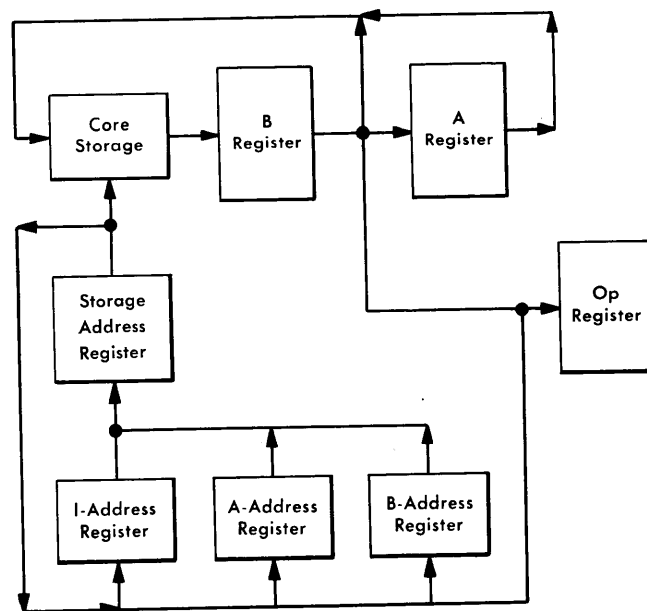


Figure 13. Processing Unit Registers

A-ADDRESS REGISTER

The A-address register normally contains the storage address of the data in the A-address portion of an instruction. Normally, as the instruction is executed, the number in this register is decreased by 1 after each storage cycle that involves the A-address.

NOTE: If the A-address portion of the instruction does not contain a core-storage address (for example %Ux), the contents of the A-address register are not modified as previously explained as the instruction is executed.

B-ADDRESS REGISTER

This register normally contains the storage location of the data in the B-address portion of an instruction. Normally, as a storage cycle involving the B-address is executed, the storage address in the B-address register is decreased by 1.

STORAGE-ADDRESS REGISTER

The storage-address register always contains the address of the core-storage position that was involved in the last particular machine cycle.

Character Registers

The A- and B-character registers and the Op-register are single-character registers used to store data during the execution of an instruction.

OP-REGISTER

The Op- (Operation) register stores the operation code of the instruction in process for the duration of the operation. The operation code is stored in BCD code including the check bit, but excluding the word mark.

B-REGISTER

Each character leaving core storage enters the B-register. The character is stored in 8-bit form (BCD code, check bit, and word mark). The B-register is reset and filled with a character from core storage on every storage cycle.

A-REGISTER

The A-register is reset and filled with the character from the B-register during each storage cycle that involves the A-address, and during all instruction cycles except the first and last I- (Instruction) cycle of each instruction. Data is stored in 8-bit form.

NOTE: Information can be written back into core storage directly from either the A- or B-register.

Figure 14 shows the I-phase of an operation and gives a detailed schematic for loading a 7-character instruction in the operation code register, in the A- and B-registers and in the I-, A-, and B-address registers. Eight storage cycles are required to load the complete instruction in the register. Each 1401 storage cycle

requires .0115 millisecond, and each 1460 storage cycle requires .006 millisecond.

NOTE: The A- and B-address registers contain 3-character addresses. Actual addresses are shown in this schematic because the storage display lights on the console show 4-digit addresses.

Chaining Instructions

In some programs, it becomes possible to perform a series of operations on several fields that are in consecutive storage locations. Some of the basic operations, such as ADD, SUBTRACT, MOVE, and LOAD, have the ability to be *chained* so that less time is required to perform the operations, and space is saved in storing instructions. Here is an example of the chaining technique: assume that four 5-position fields stored in sequence are to be added to four other sequential fields. This operation could be done using four 7-character instructions:

<u>A</u>	700	850
<u>A</u>	695	845
<u>A</u>	690	840
<u>A</u>	685	835

At the completion of the first instruction, the A-address register contains 695 and the B-address register contains 845. These are the same numbers that are in the A- and B-addresses in the second instruction. Eighty storage cycles would be required to execute these instructions, thus using up .920 millisecond (1401 time). Also, 28 storage positions are required to store these instructions.

By taking advantage of the fact that the A- and B-address registers contain the necessary information to perform the next instruction, this same sequence of operations can be executed as follows:

<u>A</u>	700	850
<u>A</u>		
<u>A</u>		
<u>A</u>		

Connecting instructions together in this manner is called *chaining*. The first add instruction contains both the A- and B-addresses. The following three instructions contain only the operation code for those instructions. The A- and B-addresses are the results left in the A- and B-address registers from the previous instruction. This type of operation requires 62 storage cycles, and takes .713 ms (1401 time) to execute. Storage of these chained instructions requires only ten storage positions.

The ability to chain a series of instructions does not depend on the use of the same operation code. Chained instructions may have various Op codes. To be oper-

CYCLE	OPERATION	Instruction Location	A	5	6	7	T	1	2	S	
			197	198	199	200	201	202	203	204	
I-Op	The operation code enters the B-register and the Op-register. Because this is the first I-cycle, the A-register is undisturbed.	I Register 0 1 9 7 OP Register A	B Register A A Address Register ? ? ? ?				A Register ? B Address Register ? ? ? ?				Cycle 1
I-1	The A-address register is reset to blanks during the first part of the cycle for all instructions. The B-address register is reset to blanks during the first part of the cycle for all operations except Move, Load, Store A- and Store B-address Register operation. During the I-1 cycle, the second instruction character (first character of the A-address) enters the thousands and hundreds positions of the A- and B-address registers and the A-register by the way of the B-register.	I Register 0 1 9 8 OP Register A	B Register 5 A Address Register 0 5 b b				A Register 5 B Address Register 0 5 b b				Cycle 2
I-2	The third character of the instruction enters the tens position of the A- and B-address registers, and the A-register through the B-register.	I Register 0 1 9 9 OP Register A	B Register 6 A Address Register 0 5 6 b				A Register 6 B Address Register 0 5 6 b				Cycle 3
I-3	The fourth instruction character enters the units position of the A- and B-address registers, and the A-register through the B-register.	I Register 0 2 0 0 OP Register A	B Register 7 A Address Register 0 5 6 7				A Register 7 B Address Register 0 5 6 7				Cycle 4
I-4	The B-address register is reset at the beginning of this cycle. The fifth instruction character (first character of the B-address) enters the hundreds position of the B-address register, and the A-register through the B-register.	I Register 0 2 0 1 OP Register A	B Register T A Address Register 0 5 6 7				A Register T B Address Register 1 3 b b				Cycle 5
I-5	The sixth instruction character goes to the tens position of the B-address register, and the A-register through the B-register.	I Register 0 2 0 2 OP Register A	B Register 1 A Address Register 0 5 6 7				A Register 1 B Address Register 1 3 1 b				Cycle 6
I-6	The seventh character of the instruction (last character of the B-address) enters the units position of the B-address register and the A-register through the B-register.	I Register 0 2 0 3 OP Register A	B Register 2 A Address Register 0 5 6 7				A Register 2 B Address Register 1 3 1 2				Cycle 7
I-7	The first character of the next instruction enters the B-register only. Because this is the last I-cycle for this instruction, the A-register and the Op-register, the A- and B-address registers are undisturbed. The detection of a word mark associated with this character signals the machine that this is the Op code for the next instruction. The loading operations stops, and the instruction that was just loaded is executed. Note that the I-address register contains the address of the high-order position of the next sequential instruction.	I Register 0 2 0 4 OP Register A	B Register S A Address Register 0 5 6 7				A Register 2 B Address Register 1 3 1 2				Cycle 8

Figure 14. Schematic of Instruction Loading

ated on, the A-fields must be in sequence, and the B-fields must be in sequence. Example:

A 900 850
M
A
M

For example, assume that the data fields are each ten characters long:

The ten characters at location 900 were added to 850.
The ten characters at location 890 were moved to 840.
The ten characters at location 880 were added to 830.
The ten characters at location 870 were moved to 820.

The description of each instruction includes the contents of the address registers after the operation has been performed. Figure 8 shows the abbreviations that indicate the contents of these registers.

By using this information, the programmer can determine the status of the registers and decide whether chaining is practical in specific cases.

NOTE: Instructions that do not contain core-storage addresses cannot be chained. For example, M %Ux xxx R is a tape read instruction. The tape unit is signaled as the machine reads the instruction. Although the A-address register contains %4x after the operation, chaining is impossible because the machine does not select the unit from the contents of the A-address register.

Most single-address instructions (Op code and an A-address) cause the A-address to be inserted in both the A- and B-address registers (for example A xxx). However, MOVE, LOAD, and STORE B-ADDRESS REGISTER (Op codes M, L, and H) do not disturb the B-address register, and permit the programmer to use the previous contents of that register as part of the instruction.

All no-address instructions (Op code only) use the previous contents of the A- and B-address registers.

The contents of the B-address register after a branch instruction (Op code and I-address) depend on whether or not the indexing feature is installed.

1. With the indexing feature installed, the B-address register contains the address of the next sequential instruction if a branch occurs.
2. Without the indexing feature installed, the B-address register is cleared to blanks whenever a branch occurs.

The operation performed by an IBM 1401 or 1460 Data Processing System can be arranged into these general classifications:

1. Arithmetic operations
2. Logic operations
3. Data moving operations
4. Miscellaneous operations
5. Edit operations
6. Input-output operations

Arithmetic Operations

The 1401 and 1460 perform the arithmetic operations of the system by executing the instructions associated with these operations. Adding, subtracting, and programmed multiply and divide operations make use of these arithmetic instructions.

Logic Operations

The 1401 and 1460 programs test for certain conditions that arise during processing, and transfer the program to a predetermined set of instructions or sub-routines, as a result of specific tests. The operations that perform the testing are called *conditional branch operations*.

Data Moving Operations

The 1401 and 1460 data moving operations are used to manipulate data within core storage during processing. Depending on the specific operation, one character, a group of characters or a part of one character can be involved in the operation.

Miscellaneous Operations

The miscellaneous operations in the 1401 and 1460 involve clearing core storage, inserting and removing of word marks from specific core-storage locations, programmed halt operations and other similar operations.

Edit Operation

The 1401 and 1460 editing operation can automatically insert all desired commas, decimals, dollar signs, asterisks, credit symbols, and minus signs in a numerical output field. In addition, unwanted zeros to the left of a significant digit can be suppressed.

Input-Output Operations

The 1401 and 1460 stored programs control the data transfer to and from the various attached input-output

units. Also, various unit operations are initiated by the stored program.

Arithmetic Operations

The IBM 1401 and 1460 Data Processing Systems add, subtract, multiply, and divide by applying the *add-to-storage* method of operation. The two factors to be combined are added within core storage without the use of special accumulators or counters. Because any storage area can be used as an accumulator field, the capacity for performing arithmetic functions is not limited by standard-size accumulators or by a predetermined number of accumulators within the system. Also, programming steps can be saved because some arithmetic operations require that only one field be transferred. In arithmetic operations, the 1401 and 1460 systems consider blanks and zeros the same. An unsigned field is considered positive by the system.

All arithmetic functions are performed under complete algebraic sign control. The sign of a factor is determined by the combination of zone bits in the units position of the fields specified by the instruction being executed.

Figure 15 shows the four possible combinations of zone bits and the values of the signs they represent.

The standard machine method of signing a field is to indicate a positive factor with A- and B-bits (12 zone) or No A- or B-bits (No zone), and to indicate a negative factor with a B-bit (11 zone).

The arithmetic operations in the IBM 1401 and 1460 Data Processing Systems are performed by using one of two types of add cycles incorporated in the system. The two types of add cycles are:

1. true add
2. complement add

SIGN	BCD CODE BIT CONFIGURATION	CARD CODE CONFIGURATION
Plus	No A- or B-Bit	No Zone
Plus	A- and B-Bits	12 Zone
Minus	B-Bit Only	11 Zone
Plus	A-Bit Only	0 Zone

Figure 15. Sign Bit Equivalents

TYPE OF OPER.	A-FLD. SIGN	B-FLD. SIGN	TYPE OF ADD CYCLE	SIGN OF RESULT
A D D	+	+	True Add	+
		-	Compl. Add	Sign of Greater Value
	-	+	Compl. Add	
		-	True Add	-
S U B T R A C T	+	-	True Add	-
		+	Compl. Add	Sign of Greater Value
	-	-	Compl. Add	
		+	True Add	+

Figure 16. Types of Add Cycles and Sign of Result for Add and Subtract Operations

The type of add cycle performed depends on the arithmetic operation and the signs and values of the two factors involved (Figure 16).

True Add

A true-add cycle is specified when the total number of minus signs is an even number (0 or 2). The signs considered are the signs of the factors and the sign of the operation.

The sign of the result after a true-add cycle carries the original sign of the B-field when either an add or a subtract operation is performed (Figure 17). The sign of the result on a zero and add or zero and subtract operation is covered in the individual description.

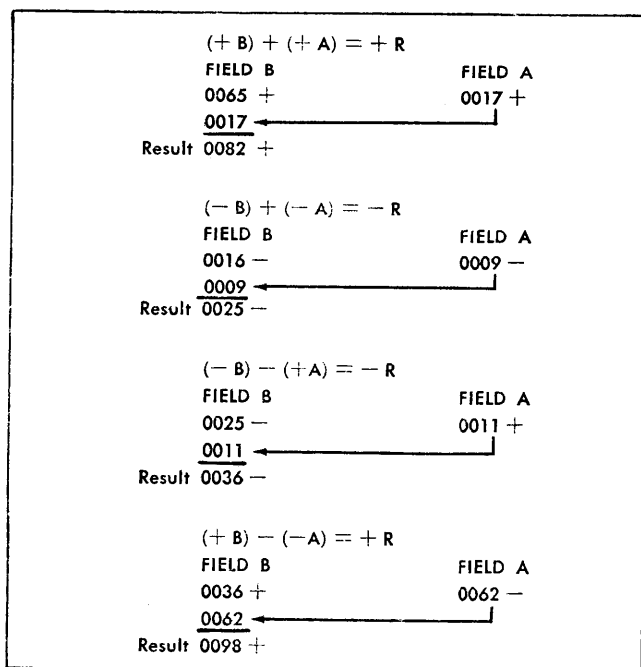


Figure 17. True-Add Cycle Examples

Complement Add

An uneven number of minus signs (1 or 3) specifies a complement-add cycle. The system converts the A-field factor to its nines complement figure and adds it to the B-field factor (plus one initial carry). The system then initiates a carry test to determine whether a carry occurred from the high-order position of the B-field. The presence of a carry indicates that the result in the B-field is a true figure (Figure 18). The original sign of the B-field is the sign of the result. The sign of the result on a zero and add or zero and subtract operation is covered in the individual description.

If there was no carry from the high-order position of the B-field, the result in the B-field is not a true figure. A recomplement cycle is performed to convert the result to a true figure. In an add operation that results in a negative figure, the sign of the result is always changed during a recomplement cycle (Figure 19). The system generates the new sign automatically. A positive factor is indicated by the presence of an A- and B-bit over the units position of the factor. After a complement-add cycle, the sign of the result carries the sign of the greater value factor.

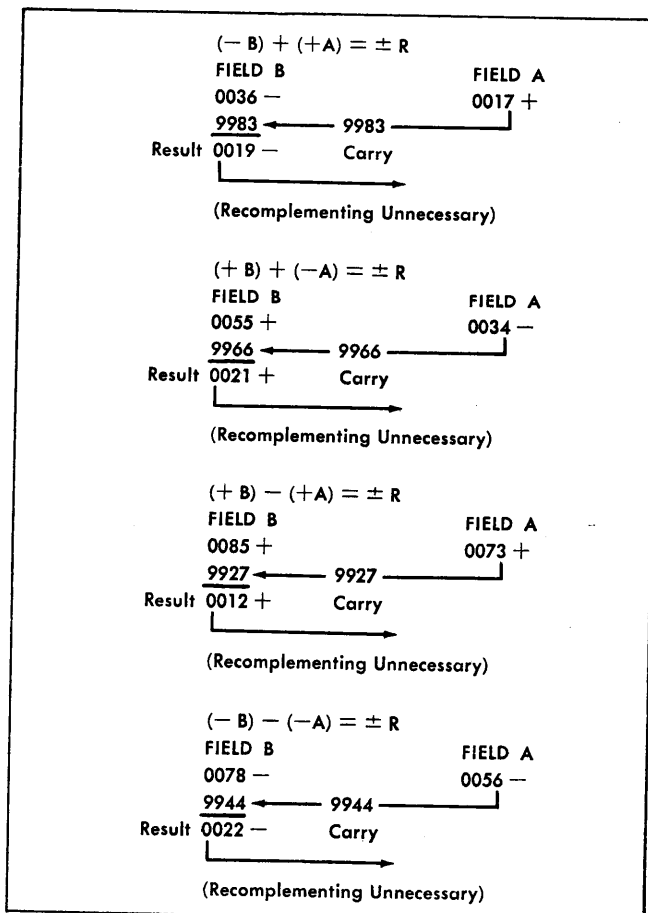


Figure 18. Complement-Add Cycle Examples

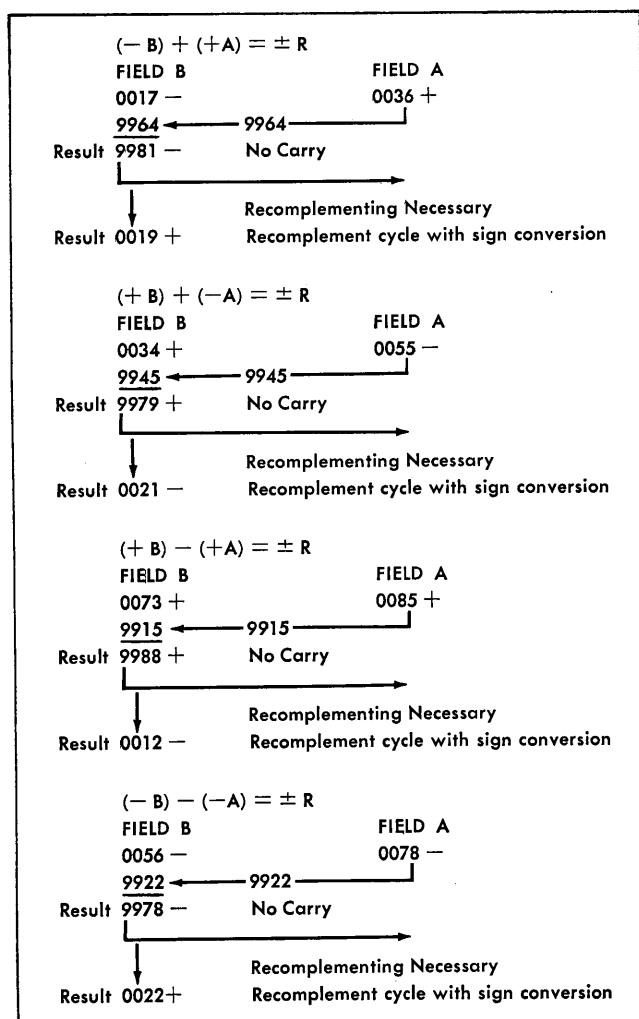


Figure 19. Complement-Add (with Recomplementing) Cycle Examples

Arithmetic Instructions

Add (Two Fields)

Instruction Format.

Mnemonic	Op Code	A-address	B-address
A	<u>A</u>	xxx	xxx

Function. The data in the A-field is added to the data in the B-field. The result is stored in the B-field.

Word Marks. The B-field must have a defining word mark, because it is this word mark that actually stops the add operation.

The A-field must have a word mark, only if it is shorter than the B-field. In this case, the transmission of data from the A-field stops after the A-field word mark is sensed. Zeros are then inserted in the A-register until the B-field word mark is sensed.

If the A-field is longer than the B-field, the high-order positions of the A-field that exceed the limits

imposed by the B-field word mark are not processed. For overflow conditions and considerations, assume that the A-field is the same length as the B-field. (See *Address Modification, Using Modulus 4 Arithmetic.*)

Timing.

1. If the operation does not require a recompute cycle:

1401:

$$T = .0115 (L_I + 3 + L_A + L_B) \text{ ms.}$$

1460:

$$T = .006 (L_I + 1 + L_A + L_B) \text{ ms.}$$

2. If a recompute cycle is taken:

1401:

$$T = .0115 (L_I + 3 + L_A + 4 L_B) \text{ ms.}$$

1460:

$T = .006 (L_I + 1 + L_A + 3 L_B) \text{ ms.}$ If the multiply-divide special feature is installed, the 1460 timing for a recompute cycle is:

$$T = .006 (L_I + 1 + L_A + 2 L_B) \text{ ms.}$$

Notes:

1. **Sign Control.** If a recompute cycle is automatically taken, the sign of the B (result) -field is changed and the result is always stored in true form. (See Figure 18.)
2. **Zone Bits.** If the fields to be added contain zone bits in other than the high-order position of the B-field and the sign positions of both fields, only the digits are used in a true-add operation. B-field zone bits are removed except for the units and high-order positions in a true-add operation. If a complement add takes place, zone bits are removed from all but the units positions of the B-field.
3. **Overflow.** If an overflow occurs during a true-add operation, the arithmetic overflow indicator is set, and the overflow indications are stored over the high-order digit of the B-field. When the A-field exceeds, or is equal to, the B-field length, and the A-field position that corresponds to the high-order B-field position contains a zone-bit, this zone-bit is added to any zone-bits present in the high-order B-field position.

Condition	Result
First overflow	A-bit
Second overflow	B-bit
Third overflow	A- and B-bits
Fourth overflow	No A- or B-bits

Conditions 1-4 are repeated for five and more overflow conditions. Overflow indication does not occur for a 1-position field.

The BRANCH IF ARITHMETIC OVERFLOW INDICATOR ON (B xxx Z) instruction tests and turns off the arithmetic overflow indicator and branches to a special instruction or group of instructions if this condition occurs. There is only one overflow indicator in the system. It is turned off by a BRANCH IF ARITHMETIC OVERFLOW INDICATOR ON instruction, or pressing the start reset key.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	A-L _w	B-L _B

Example. Add CURERN (0506) to YTDGRO (0708),
Figure 20.

SPS															
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				IND	d		
				ADDRESS	±	CHAR. ADJ.	IND	ADDRESS	±	CHAR. ADJ.	IND				
3	5	7	8	13	14	16	17	23	24	26	27	28	29		
0	1	0		A				CURERN					YTDGRO		

Autocoder															
Label				Operation				OPERAND							
5	15	20	21	25	30	35	40	45	50						
				A				CURERN, YTDGRO							

Assembled Instruction: A 506 708

Figure 20. Add (Two Fields)

Add (One Field)

Instruction Format.

Mnemonic	Op Code	A-address
A	<u>A</u>	xxx

Function. This format of the ADD instruction causes the data in the A-field to be added to itself.

Word Marks. The A-field must have a defining word mark. It is this word mark that stops the add operation. This instruction must be followed by a word mark in the position after the A-address.

Timing.

$$1401: T = .0115 (L_I + 3 + 2L_A) \text{ ms.}$$

$$1460: T = .006 (L_I + 1 + 2L_A) \text{ ms.}$$

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	A-L _A	A-L _A

Example. Add to itself the data at EXEMPT (0981),
Figure 21.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				IND	d		
				ADDRESS	±	CHAR. ADJ.	IND	ADDRESS	±	CHAR. ADJ.	IND				
3	5	7	8	13	14	16	17	23	24	26	27	28	29	30	31
0	1	0		A				EXEMPT							

Autocoder

Label				Operation				OPERAND							
5	15	20	21	25	30	35	40	45	50						
				A				EXEMPT							

Assembled Instruction: A 981

Figure 21. Add (One Field)

Zero and Add (Two Fields)

Instruction Format.

Mnemonic	Op Code	A-address	B-address
ZA	<u>Z</u>	xxx	xxx

Function. This instruction adds the A-field to a zeroed B-field. Technically, this is accomplished by moving the A-field to the B-field. The high-order positions of the B-field are set to zero if the B-field is larger than the A-field. The data from the A-field moves directly from the A-register to storage. Zone bits are stripped from all positions except the units position. Blanks in the A-field are stored as blanks in the B-field.

Word Marks. A word mark is required for definition of the B-field. It is required in the A-field, only if it is shorter than the B-field. If the A-field is shorter than the B-field, all extra high-order B-field positions contain zeros. But the transmission of data from the A-field stops when the A-field word mark is detected.

Timing. $T = N (L_I + 1 + L_A + L_B) \text{ ms.}$

Note: The sign of the result always has both A- and B-bits if it is positive. If the sign is negative, it has only a B-bit.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	A-L _w	B-L _B

Example. Zero WHTAX area (0796-0802) and add new TAX (0749-0754) to WHTAX (Figure 22).

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND								(B) OPERAND							
				ADDRESS				±	CHAR. ADJ.	IND	ADDRESS				±	CHAR. ADJ.	IND	d	
3	5	6	7	8	13	14	16	17	23	24	26	27	28	29	34	35	36	37	38
0	1	0			ZA				TAX					WHTAX					

Autocoder

Label				Operation				OPERAND							
5	15	20	21	25	30	35	40	45	50						
				ZA				TAX, WHTAX							

Assembled Instruction: ? 754 802

Figure 22. Zero and Add (Two Fields)

Zero and Add (One Field)

Instruction Format.

Mnemonic	Op Code	A-address
ZA	<u>Z</u>	xxx

Function. This format of the ZERO AND ADD instruction is used to strip the A-field of all zone bits, except in the units position (sign). The A-field sign is retained, and if it is plus, the bit configuration might change.

If the A-field plus-sign bit configuration is not an A- and B-bit, it is changed to the A- and B-bit configuration.

Word Marks. The A-field must have a word mark in its high-order position.

Timing. $T = N (L_I + 1 + 2L_A)$ ms.

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI A-L_A A-L_A

Example. Strip zone bits from TOTAL (0560) area (Figure 23).

SPS											
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND			
				ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±	CHAR. ADJ.	±
3	8	7	5	13	16	17	11	27	28	30	32
0	1	0		2A				TOTAL			

Autocoder											
Label	Operation	OPERAND									
		15	20	21	25	30	35	40	45	50	55
2A											

Figure 23. Zero and Add (One Field)

timing for a recomplement cycle is:

$$T = .006 (L_I + 1 + L_A + 2L_B) \text{ ms.}$$

Note. If a recomplement cycle is automatically taken, the sign of the B (result)-field is changed, and the result is always stored in true form.

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI A-L_W B-L_B

Example. Subtract CUFICA (0753) from CURGRO (0896), Figure 24.

SPS											
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND			
				ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±	CHAR. ADJ.	±
3	8	7	5	13	16	17	11	27	28	30	32
0	1	0		S				CUFICA			
								CURGRO			

Autocoder											
Label	Operation	OPERAND									
		15	20	21	25	30	35	40	45	50	55
S											

Assembled Instruction: S 753 896

Figure 24. Subtract (Two Fields)

Subtract (Two Fields)

Instruction Format.

Mnemonic Op Code A-address B-address
S S xxx xxx

Function. The numerical data in the A-field is subtracted from the numerical data in the B-field. The result is stored in the B-field. Refer to Figure 16 for the sign that results from a specific subtract operation.

Word Marks. A word mark is required to define the B-field. An A-field requires a word mark, only if it is shorter than the B-field. In this case, the A-field word mark stops transmission of data from the A-field.

Timing.

1. Subtract—no recomplement:

1401:

$$T = .0115 (L_I + 3 + L_A + L_B) \text{ ms.}$$

1460:

$$T = .006 (L_I + 1 + L_A + L_B) \text{ ms.}$$

2. Subtract—recomplement cycle necessary:

1401:

$$T = .0115 (L_I + 3 + L_A + 4L_B) \text{ ms.}$$

1460:

$$T = .006 (L_I + 1 + L_A + 3L_B) \text{ ms. If the multiply-divide special feature is installed, the 1460}$$

Subtract (One Field)

Instruction Format.

Mnemonic Op Code A-address
S S xxx

Function. The data at the A-address is subtracted from itself. If the A-field sign is minus, the result is a minus zero. If the A-field sign is plus, the result is a plus zero.

Word Marks. The A-field must have a defining word mark. This instruction must be followed by a word mark in the position after the A-address.

Timing.

1401:

$$T = .0115 (L_I + 3 + 2L_A) \text{ ms.}$$

1460:

$$T = .006 (L_I + 1 + 2L_A) \text{ ms.}$$

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI A-L_A A-L_A

Example. Subtract from itself the field labeled LIMIT (units position is 0395), Figure 25.

SPS											
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND			
3	6	7	8	13	14	15	17	23	24	25	26
0	1	0		S	L	I	M	I	T		

Autocoder											
Label	Operation	20	21	25	30	35	40	45	50	55	60
S	LIMIT										

Assembled Instruction: S 395

Figure 25. Subtract (One Field)

Zero and Subtract (Two Fields)

Instruction Format.

Mnemonic Op Code A-address B-address
 ZS 1 XXX XXX

Function. This instruction subtracts the A-field from a zeroed B-field. Technically, this is accomplished by moving the A-field to the B-field. The high-order positions of the B-field are set to zero if the B-field is larger than the A-field. The data from the A-field is moved directly from the A-register to the B-field. Zone bits are stripped from all but the sign (units) position.

Word Marks. A word mark is required to define the B-field. If the A-field is shorter than the B-field, the A-field must have a defining word mark to stop transmission of data to B. The extra high-order B-field positions contain zeros, if A is shorter than B.

Timing. $T = N (L_I + 1 + L_A + L_B)$ ms.

Note. If the A-field is positive, the B-field result is negative. If the A-field is negative, the B-field result is positive.

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
 NSI A-L_W B-L_B

Example. Zero ACCUM 1 (0755) and subtract TAX-EXP (0699) from ACCUM 1, Figure 26.

SPS											
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND			
3	6	7	8	13	14	15	17	23	24	25	26
0	1	0		S	T	A	X	E	X	P	

Autocoder											
Label	Operation	20	21	25	30	35	40	45	50	55	60
S	TAXEXP, ACCUM1										

Assembled Instruction: 1 699 755

Figure 26. Zero and Subtract (Two Fields)

Zero and Subtract (One Field)

Instruction Format.

Mnemonic Op Code A-address
 ZS 1 XXX

Function. This instruction causes the sign of the A-field to be changed.

Word Marks. The data in the A-field requires a word mark in its high-order position.

Timing. $T = N (L_I + 1 + 2L_A)$ ms.

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
 NSI A-L_A A-L_A

Example. Subtract LIMIT (0495) from zero, and change sign of LIMIT's value (Figure 27).

SPS											
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND			
3	6	7	8	13	14	15	17	23	24	25	26
0	1	0		Z	S	L	I	M	I	T	

Autocoder											
Label	Operation	20	21	25	30	35	40	45	50	55	60
ZS	LIMIT										

Assembled Instruction: 1 495

Figure 27. Zero and Subtract (One Field)

Logic Operations

The 1401 and 1460 programs can test for certain conditions that can arise during processing. When the tested conditions are present, the program transfers to a predetermined set of instructions or subroutines. The operations that perform these testing operations are called *conditional branch operations*.

For example, if an overflow occurs in an arithmetic operation, a second routine to handle this condition can be initiated by executing a BRANCH IF ARITHMETIC OVERFLOW INDICATOR ON instruction. The branching to this special routine is called a *conditional branch*. The sequential execution of program steps is bypassed, and the program branches to the address of the instruction specified by the I-address of the conditional branch instruction. If the condition is not present, then the system starts reading the instruction that appears at the immediate right of the conditional branch instruction (next sequential instruction).

All conditional branch instructions have a d-character that is used to specify the conditions necessary for a program transfer. A branch that occurs as a direct re-

sult of the execution of the instruction itself is called an *unconditional branch*. No special condition (other than the execution of the program step) is needed to transfer the program out of its normal sequential execution.

Any branch operation that ends with a successful branch to another portion of core storage for the next instruction address operates as follows:

The B-address register is reset to blanks during the next instruction operation (I-Op) cycle.

If the advanced programming special feature is installed on the 1401, or the indexing and store address register special feature is installed on the 1460, the indexing portion of the special feature causes the following:

1. the next sequential instruction (NSI) is placed in the B-address register
2. the B-address register is not set to blanks during the following instruction
3. the timing is increased one core-storage cycle.

Logic Instructions

Branch (Unconditional)

Instruction Format.

Mnemonic	Op Code	I-address
B	<u>B</u>	xxx

Function. This instruction always causes the program to branch to the address specified by the I-address portion of the instruction. This address contains the Op code of some instruction.

This unconditional branch operation is used to interrupt normal program sequence and continue the program at some other point, without testing for specific conditions.

Word Mark. A word mark must be associated with the core-storage position of the next I-address.

Timing.

Branch (without indexing):

$$T = N(L_I + 1) \text{ ms.}$$

Branch (with indexing):

$$T = N(L_I + 2) \text{ ms.}$$

Address Registers After Operation.

Branch (without indexing)

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	Blank

Branch (with indexing)

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	NSI

Example. Unconditionally branch to AGAIN (3498), Figure 28.

SPS																										
			(A) OPERAND												(B) OPERAND											
LINE	COUNT	LABEL	OPERATION	ADDRESS				±	CHAR.	ADJ.	±	CHAR.	ADJ.	ADDRESS				±	CHAR.	ADJ.	±	CHAR.	ADJ.	d		
3	5	6	7	8	13	14	15	16	17	23	27	28	29	30	31	32	33	34	35	36	37	38	39	40		
0	1	0			B																					

Autocoder		OPERAND									
Label	Operation	25	30	35	40	45	50	55	60	65	70
	B										

Assembled Instruction: B D98

Figure 28. Branch (Unconditional)

Branch if Indicator On

Instruction Format.

Mnemonic	Op Code	I-address	d-character
SPS B	<u>B</u>	xxx	x

A see Figure 29

Function. The d-character specifies the indicator tested. If the indicator is on, the next instruction is taken from the I-address. If the indicator is off, the next sequential instruction is taken. Figure 29 shows characters that are valid for the d-character and for

AUTOCODER MNEMONIC	d-CHARACTER	BRANCH ON
B	bl	Unconditional
BC9	9	Carriage Channel #9
BCV	@	Carriage Channel #12
BLC	A	"Last Card" switch (sense switch A)
BSS ±	B	Sense Switch B*
BSS ±	C	Sense Switch C*
BSS ±	D	Sense Switch D*
BSS ±	E	Sense Switch E*
BSS ±	F	Sense Switch F*
BSS ±	G	Sense Switch G*
BEF	K	End of Reel*†
BER	L	Tape Transmission Error*
BIN ±	N	Access Inoperable*
BIN ±	?	Reader Error if I/O Check Stop Switch is off†
BIN ±	!	Punch Error if I/O Check Stop Switch is off†
BPB	P	Printer Busy (print storage feature)*
BIN ±	±	Printer Error if I/O Check Stop Switch is off†
BU	/	Unequal Compare (B ≠ A)
BIN ±	*	Inquiry Clear*
BIN ±	Q	Inquiry Request*
BPCB	R	Printer Carriage Busy (print storage feature)*
BE	S	Equal Compare (B = A)*
BL	T	Low Compare (B < A)*
BH	U	High Compare (B > A)*
BIN ±	V	Read-Write Parity Check or Read-Back Check Error*
BIN ±	W	Wrong-Length Record*
BIN ±	X	Unequal-Address Compare*
BIN ±	Y	Any Disk-Unit Error Condition*
BAV	Z	Overflow†
BIN ±	%	Processing Check with Process Check Switch off†

*Special feature

†Conditions tested are reset by a BRANCH IF INDICATOR ON instruction.

‡The d-character must be used with this mnemonic.

Figure 29. Branch if Indicator On Mnemonic, d-character, and Conditions

the indicators they test. This figure also shows testing, for high, low, or equal, which is used when the high-low-equal compare special feature (1401; standard on 1460) is installed.

The indicators tested are not turned off by this instruction except as noted by a †. When carriage tape-channels 9 or 12 are sensed, the corresponding indicators are turned on. These carriage channel-indicators are turned off when any other carriage tape-channel is sensed. (Refer to the *Notes* section in *Control Carriage* for other information.) The next COMPARE instruction turns off the compare indicators.

Word Marks. Word marks are not affected.

Timing.

No Branch:

$$T = N (L_I + 1) \text{ ms.}$$

Branch (without indexing):

$$T = N (L_I + 1) \text{ ms.}$$

Branch (with indexing):

$$T = N (L_I + 2) \text{ ms.}$$

Address Registers After Operation. All d-characters (blank and other):

No Branch

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	dbb

Branch (without indexing):

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	Blank

Branch (with indexing):

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	NSI

Example. Test for the last card. If it is the last card, branch to END (0599), Figure 30.

SPS											
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND			
3	5	7	13	14	16	17	23	24	26	27	28
0	1	0	B	END							A

Autocoder											
Label	Operation	20	21	25	30	35	40	45	50	OPERAND	
BLC	END										

Assembled Instruction: **B 599 A**

Figure 30. Branch if Indicator On

Branch if Character Equal

Instruction Format.

Mnemonic	Op Code	I-address	B-address	d-character
SPS B	<u>B</u>	xxx	xxx	x
A BCE				

Function. This instruction causes the single character at the B-address to be compared to the d-character. If it has the same bit configuration as the d-character, the program branches to the I-address, otherwise the program continues sequentially. The d-character can be any combination of the six BCD code bits (BA 8421).

Word Marks. Word marks in the location tested have no effect on the operation.

Timing.

No Branch

$$T = N (L_I + 2) \text{ ms.}$$

Branch (without indexing)

$$T = N (L_I + 2) \text{ ms.}$$

Branch (with indexing)

$$T = N (L_I + 3) \text{ ms.}$$

Note. Care must be taken if the specified d-character is also the same d-character that can test and reset an indicator OFF, when it is used with a B xxx d instruction. If the indicator is on during the execution of the B xxx xxx d instruction, it is turned off.

Address Registers After Operation.

No Branch:

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	B-1

Branch (without indexing):

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	Blank

Branch (with indexing):

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	NSI

Example. This example shows how the chaining method can be used to test an entire field for blank characters. Each position in the area labeled AMOUNT (0350, 0349, 0348 and 0347) is individually tested for a blank character. If a blank is found, the program branches to BLANK (0601) for the next instruction. If the position tested contains a character, the program continues in sequence (Figure 31).

SPS											
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND			
3	5	7	13	14	16	17	23	24	26	27	28
0	1	0	B	BLANK				AMOUNT			
0	2	0	B								
0	3	0	B								
0	4	0	B								

Autocoder

Label	Operation	20	21	25	30	35	40	45	50	OPERAND	
BCE	BLANK, AMOUNT,										
BCE											
BCE											
BCE											

Assembled Instruction: **B 601 350 bl**

**B
B
B**

Figure 31. Branch if Character Equal

Branch if Word Mark and/or Zone

Instruction Format.

Mnemonic	Op Code	I-address	B-address	d-character
BWZ	<u>V</u>	xxx	xxx	x

Function. The single character at the B-address is examined for a particular bit configuration, as specified by the d-character. If the bit configuration is present as specified, the program branches to the I-address for the next instruction:

d-character	Condition
1	Word mark
2	No zone (No-A, No-B-bit)
B	12-zone (AB-bits)
K	11-zone (B, No-A-bit)
S	Zero-zone (A, No-B-bit)
3	Either a word mark, or no zone
C	Either a word mark, or 12-zone
L	Either a word mark, or 11-zone
T	Either a word mark, or zero-zone

Word Marks. These have been explained previously.

Timing.

No Branch:

$$T = N (L_I + 2) \text{ ms.}$$

Branch (without indexing):

$$T = N (L_I + 2) \text{ ms.}$$

Branch (with indexing):

$$T = N (L_I + 3) \text{ ms.}$$

Address Registers After Operation.

No Branch

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	B-1

Branch (without indexing)

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	Blank

Branch (with indexing)

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	NSI

Example. Test the units position of GROAMT (2498) for an 11-zone, and branch to NEGRTE (0598) for the next instruction. If there is no 11-zone, continue the program sequence (Figure 32).

SPS														
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d		
				ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±	CHAR. ADJ.	±			
3	5	6	7	8	13	14	15	17	23	24	25	28	30	31
					BWZ				NEGRTE					
					GROAMT									K

Autocoder											
Label		Operation		OPERAND							
5	10	15	20	25	30	35	40	45	50	55	60

Assembled Instruction: V 598 M98 K

Figure 32. Branch if Word Mark and/or Zone

Compare

Instruction Format.

Mnemonic	Op Code	A-address	B-address
C	<u>C</u>	xxx	xxx

Function. The data in the A-field is compared to an equal number of characters in the B-field. The bit configuration (BA 8421) of each character in the two fields is compared. The comparison turns on an indicator that can be tested by a subsequent BRANCH IF INDICATOR ON instruction. The indicator is reset by the next 7-character COMPARE instruction.

The same indicators set by the COMPARE instruction are also affected by a disk-unit operation (seek, read, write, and write check). The disk unit performs an address-compare operation automatically on the address in core storage, with the address on the disk record, by using the compare circuits and by setting the appropriate indicator (equal, high, or low). Therefore, careful consideration must be made in the use of a COMPARE instruction and the subsequent BRANCH IF INDICATOR ON instructions for testing the results of the COMPARE instruction when disk-unit operations are to be performed.

Word Marks. The first word mark encountered stops the operation. If the A-field is longer than the B-field, extra A-field positions at the left of the B-field word mark are not compared. If the B-field is longer than the A-field, an unequal-compare results.

Timing. $T = N (L_I + 1 + 2L_W) \text{ ms.}$

Note: Both fields must have exactly the same bit configurations, to be equal. For example, 00P(? = 0) compared to 00!(= 0) results in an unequal comparison.

All characters that can appear in storage can be compared. The ascending sequence of characters is as follows: blank · □ & \$ * - / , % # @ ? A through ! J through R ≠ S through Z 0 through 9.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	A-L _w	B-L _w

Example. Compare the department numbers punched in two cards. Department numbers are located in:

Card	Label	Actual Address
1	DEPTNO	1098
2	DEPTCD	0004

Then test the result of the compare operation. If the department numbers are equal, continue the program in sequence. If they are unequal, branch to TOTAL (0495) for the next instruction (Figure 33).

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d
				ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±	CHAR. ADJ.	±	
0	1	0		DEPTCO				DEPTNO				
0	2	0		TOTAL								

Autocoder

Label	Operation	Operand
C	DEPTCO, DEPTNO	
BU	TOTAL	

Assembled Instruction: C 004 ±98
B 495 /

Figure 33. Compare

Data-Moving Operations

The 1401 and 1460 data-moving operations are used to manipulate data within core storage during processing. Depending on the specific operation, one character, a group of characters, or a part of one character can be involved in the operation. A move operation does not affect word marks, but a load operation causes word marks, as well as data, to be transferred.

Data-Moving Instructions

Move Characters to A or B Word Mark (Two Fields)

Instruction Format.

Mnemonic	Op Code	A-address	B-address
SPS MCW	<u>M</u>	xxx	xxx
A MLC			

Function. The data in the A-field is moved to the B-field.

Word Marks. If both fields are the same length, only one of the fields must have a defining word mark. The first word mark encountered stops the operation. If the word mark is sensed in the A-field, the machine takes one more B-cycle to move the high-order character from A to B. At the end of the operation, the A-address register and the B-address register contain the addresses of the storage locations immediately to the left of the A- and B-fields processed by the instruction. The data at the A-address is unaffected by the move operation. Word marks in both fields are undisturbed.

Timing. $T = N (L_I + 1 + 2L_W)$ ms.

Note. If the fields are unequal in length, chaining can produce unwanted results, because one of the fields has not been completely processed. Thus, one of the registers will *not* contain the address of the units position of the left-adjacent field.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	A-L _w	B-L _w

Example. Move the 5-character field NAMIN (0750) to the 5-character field NAMOUT (0850), Figure 34.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d
				ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±	CHAR. ADJ.	±	
0	1	0		NAMIN				NAMOUT				

Autocoder

Label	Operation	Operand
MLC	NAMIN, NAMOUT	

Assembled Instruction: M 750 850

Figure 34. Move Characters to A or B Word Mark (Two Fields)

Move Characters to A or B Word Mark (One Field)

Instruction Format.

Mnemonic	Op Code	A-address
SPS MCW	<u>M</u>	xxx
A MLC		

Function. This format of the move operation can be used when it is desired to move fields from the A-area and store them sequentially in the B-area. It saves program storage space and time, because the B-address is automatically taken from the B-address register, and does not have to be written or interpreted as part of the instruction.

Word Marks. A word mark is required over the high-order position of the A- or B-field. The first word mark encountered stops the move operation.

Timing. $T = N (L_I + 1 + 2L_W)$ ms.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	A-L _w	Bp-L _w

Example. Move the following three fields (labeled EMPNO, DEPTNO and TAXCLS) and store them sequentially at RECOUNT (units position at 209), Figure 35.

Note: If the B-address register already contains the correct address, the B-label of the first instruction in the example can be eliminated.

	A-label	A-actual address	B-label	B-actual address
Employee number	EMPYNO	0101-0104		0201-0204
Department	DEPTNO	0108-0110		0205-0207
Tax Class	TAXCLS	0114-0115	RECOUT	0208-0209

SPS

5F3														
LINE	COUNT	LABEL	OPERATION	(A) OPERAND					(B) OPERAND					d
				ADDRESS	±	CHAR. ADJ.	IND	ADDRESS	±	CHAR. ADJ.	IND			
3	5	6	7	8	13	14	15	17	23	24	25	26	38	39
0	1	0			MC	W	T	A	X	C	L	S		
0	2	0			MC	W	D	E	P	T	N	O		
0	3	0			MC	W	E	M	P	T	N	O		
0	4	0												
						</								

Autocoder

Label	Operation	Operand
5	15	20
MLC	TAXCLS, RECOUT	25 30 35 40 45 50
MLC	DEPTNO	
MLC	EMPYNO	

Assembled Instruction: M 115 209
M 110
M 104

Figure 35. Move Characters to A or B Word Mark (One Field)

Move Characters and Suppress Zeros

Instruction Format.

Mnemonic	Op Code	A-address	B-address
MCS	<u>Z</u>	xxx	xxx

Function. The data in the A-field is moved to the B-field. After the move, high-order zeros and commas are replaced by blanks in the B-field. Any character that is not a comma, hyphen, blank, significant digit, or zero causes zero suppression to begin again. The sign is removed from the units position of the data field.

Word Marks. The A-field word mark stops transmission of data. B-field word marks, encountered during the move operation, are erased.

Timing. $T = N (L_I + 1 + 3L_A)$ ms.

Note: This description of the instruction assumes a 1401 or 1460 system without the expanded print edit special feature. If the feature is installed, a decimal does not restart zero suppression.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	A-L _A	B + 1

Example. Move and suppress the zeros in the 10-character field labeled INVBAL (0958) to the area labeled OUTPT4 (0448), Figure 36.

SPS

				(A) OPERAND					(B) OPERAND							
LINE	COUNT			LABEL	OPERATION	ADDRESS		±	CHAR. ADJ.	IND	ADDRESS		±	CHAR. ADJ.	IND	
3	5	6	7	8	13	14	15	16	17	23	24	25	26	34	35	36
0	1	0			MC	S	I	N	V	A	L					
												O	U	T	P	T

Autocoder

Label	Operation	Operand
5	15	20
MCS	INVBAL, OUTPT4	25 30 35 40 45 50

Assembled Instruction: Z 958 448

Figure 36. Move Characters and Suppress Zeros

Move Numeric

Instruction Format.

Mnemonic	Op Code	A-address	B-address
SPS MN	<u>D</u>	xxx	xxx
A MLNS			

Function. The numeric portion (8-4-2-1 bits) of the single character in the A-address is moved to the B-address. The zone portions (AB-bits) are undisturbed at both addresses.

Word Marks. Word marks are not required at either address, because the nature of the instruction always specifies that only one digit is to be transmitted.

Timing. $T = N (L_I + 3)$ ms.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	A-1	B-1

Example. Move the numeric portion of the units position of ONHAND (0986) to OUT5 (0789), Figure 37.

SPS

				(A) OPERAND				(B) OPERAND			
LINE	COUNT	LABEL	OPERATION	ADDRESS	±	CHAR. ADJ.	IND.	ADDRESS	±	CHAR. ADJ.	IND.
3	5	6	7	8	9	10	11	12	13	14	15
0	1	0		MM	ONHAND			OUT	5		

Autocoder

Label	Operation	Operand
5	15	20
MLNS	ONHAND, OUT5	25 30 35 40 45 50

Assembled Instruction: D 986 789

Figure 37. Move Numeric

Move Zone

Instruction Format.

Mnemonic	Op Code	A-address	B-address
SPS MLZS	<u>Y</u>	xxx	xxx

Function. Only the zone portion (AB-bits) is moved from the A-address to the B-address. The digit portions (8-4-2-1 bits) are undisturbed at both addresses.

Word Marks. Word marks are not required at either the A- or B-addresses, because this instruction involves a single character.

Timing. $T = N (L_I + 3)$ ms.

Address Registers After Operation.

<i>I-Add. Reg.</i>	<i>A-Add. Reg.</i>	<i>B-Add. Reg.</i>
NSI	A-1	B-1

Example. Move the zone bits from the units position of NEWBAL (3100) to the area labeled REC2 (3195), Figure 38.

SPS															
LINE	COUNT	LABEL	OPERATION	(A) OPERAND						(B) OPERAND					
				ADDRESS			CHAR. ADJ.			ADDRESS			CHAR. ADJ.		
3	5	6	7	8	13	14	15	16	17	23	24	25	26	27	28
0	1	0			MLZ					NEWBAL				REC2	

Figure 38. Move Zone

Load Characters to A Word Mark (Two Fields)

Instruction Format.

<i>Mnemonic</i>	<i>Op Code</i>	<i>A-address</i>	<i>B-address</i>
SPS LCA	<u>L</u>	xxx	xxx
A MLCWA			

Function. This instruction is commonly used to load data into designated printer or punch output areas of storage, and also to transfer data or instructions from a designated read-in area to another storage area. The data and word mark from the A-field are transferred to the B-field, and all other word marks in the B-field are cleared.

Word Marks. The A-field must have a defining word mark, because the A-field word mark stops the operation. *Note:* If the B-field is larger than the A-field, the B-field word mark is not cleared.

Timing. $T = N (L_I + 1 + 2L_A)$ ms.

Address Registers After Operation.

<i>I-Add. Reg.</i>	<i>A-Add. Reg.</i>	<i>B-Add. Reg.</i>
NSI	A-L _A	B-L _A

Example. Transfer the data and word marks from REC4 (0950) to OUT8 (0650), Figure 39.

SPS																
LINE	COUNT	LABEL	OPERATION	(A) OPERAND						(B) OPERAND						
				ADDRESS	±	CHAR. ADJ.	IND.	DATA	ADDRESS	±	CHAR. ADJ.	IND.	DATA	DATA	DATA	
3	8	6	7	8	13	14	15	17	23	27	28	34	35	36	38	
	0	1	0		LCAREC4						OUT8					

Autocoder	
Label	Operation
15	20 21 25 30 35 40 45
	MLCWAREC4.OUT8

Assembled Instruction: L 950 650

Figure 39. Load Characters to A Word Mark (Two Fields)

Load Characters to A Word Mark (One Field)

Instruction Format.

<i>Mnemonic</i>	<i>Op Code</i>	<i>A-address</i>
SPS LCA	<u>L</u>	xxx
A MLCWA		

Function. This format can be used when several A-fields (not necessarily in sequence) are to be loaded sequentially in the B-field. This instruction causes the A-field data and word mark to be moved to the B-field. B-field word marks are cleared, up to the A-field word mark.

Word Marks. The A-field word mark stops the operation. Therefore, B-field word marks, beyond the left limit of the A-field, are not cleared.

Timing. $T = N (L_I + 1 + 2L_A)$ ms.

Address Registers After Operation.

<i>I-Add. Reg.</i>	<i>A-Add. Reg.</i>	<i>B-Add. Reg.</i>
NSI	A-L _A	Bp-L _A
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	10	10
11	11	11
12	12	12
13	13	13
14	14	14
15	15	15
16	16	16
17	17	17
18	18	18
19	19	19
20	20	20
21	21	21
22	22	22
23	23	23
24	24	24
25	25	25
26	26	26
27	27	27
28	28	28
29	29	29
30	30	30
31	31	31
32	32	32
33	33	33
34	34	34
35	35	35
36	36	36
37	37	37
38	38	38
39	39	39
40	40	40
41	41	41
42	42	42
43	43	43
44	44	44
45	45	45
46	46	46
47	47	47
48	48	48
49	49	49
50	50	50
51	51	51
52	52	52
53	53	53
54	54	54
55	55	55
56	56	56
57	57	57
58	58	58
59	59	59
60	60	60
61	61	61
62	62	62
63	63	63
64	64	64
65	65	65
66	66	66
67	67	67
68	68	68
69	69	69
70	70	70
71	71	71
72	72	72
73	73	73
74	74	74
75	75	75
76	76	76
77	77	77
78	78	78
79	79	79
80	80	80
81	81	81
82	82	82
83	83	83
84	84	84
85	85	85
86	86	86
87	87	87
88	88	88
89	89	89
90	90	90
91	91	91
92	92	92
93	93	93
94	94	94
95	95	95
96	96	96
97	97	97
98	98	98
99	99	99
100	100	100

Example. Load the following three fields: EMPYNO, DEPTNO, and TAXCLS with their word marks to sequential locations, beginning at storage location 0201, Figure 40.

<i>A-label</i>	<i>A-actual address</i>	<i>B-label</i>	<i>B-actual address</i>
----------------	-----------------------------	----------------	-----------------------------

Employee				
number	EMPYNO	0101-0104		0201-0204
Department	DEPTNO	0108-0110		0205-0207
Tax Class	TAXCLS	0114-0115	PRINT1	0208-0209

[illegible][illegible]

```

Assembled Instruction: L 115
                     L 110
                     L 104

```

Figure 40. Load Characters to A Word Mark (One Field)

Miscellaneous Operations

The miscellaneous operations in IBM 1401 and 1460 Data Processing Systems involve the insertion of word marks into, and removal from, specific core-storage locations, clearing core-storage areas, programmed halt operations, and other similar operations.

Miscellaneous Instructions

Clear Storage

Instruction Format.

Mnemonic	Op Code	A-address
CS	$\angle$	xxx

Function. As many as 100 positions of core storage can be cleared of data and word marks when this instruction is executed. Clearing starts at the A-address and continues leftward to the nearest hundreds position. The cleared area is set to blanks.

Word Marks. Word marks are not required to stop the operation.

Timing. $T = N (L_I + 1 + L_X)$ ms.

Note: During the execution of this instruction, only the B-address register is used. Therefore, when chaining is being considered, the contents of the A-address register can be ignored.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	A	x 00-1

Example. Clear WAREA5 (0500-0563), Figure 41.

SPS											
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND			
				ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±	CHAR. ADJ.	±
3	8	7	8	13	14	15	16	17	18	19	20
0	1	0		CS		WAREA5					

Autocoder											
Label	Operation	OPERAND									
		15	16	20	21	25	30	35	40	45	50
CS				WAREA5							

Assembled Instruction: $\angle$ 563

Figure 41. Clear Storage

Clear Storage and Branch

Instruction Format.

Mnemonic	Op Code	I-address	B-address
CS	$\angle$	xxx	xxx

Function. This is the same as the CLEAR STORAGE instruction, except that the clearing starts at the B-

address. The I-address specifies the location of the next instruction.

Word Marks. Word marks are not required to stop the operation. It is not necessary to follow this instruction by a character with a word mark.

Timing.

Without indexing:

$$T = N (L_I + L_X) \text{ ms.}$$

With indexing:

$$T = N (L_I + 1 + L_X) \text{ ms.}$$

Address Registers After Operation.

Without indexing

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	Blank

With indexing

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	NSI

Example. Clear WAREA8 (0800-0898) and branch to START4 (0498) for the next instruction (Figure 42).

SPS											
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND			
				ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±	CHAR. ADJ.	±
3	8	7	8	13	14	15	16	17	18	19	20
0	1	0		CS		START4		WAREA8			

Autocoder											
Label	Operation	OPERAND									
		15	16	20	21	25	30	35	40	45	50
CS				START4		WAREA8					

Assembled Instruction: $\angle$ 498 898

Figure 42. Clear Storage and Branch

Set Word Mark (Two Addresses)

Instruction Format.

Mnemonic	Op Code	A-address	B-address
SW	$\angle$	xxx	xxx

Function. A word mark is set at each address specified in the instruction. The data at each address is undisturbed.

Word Marks. Word marks are set at both the A- and B-addresses specified. It is not necessary to follow this instruction by a character with a word mark.

Timing. $T = N (L_I + 3)$ ms.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	A-1	B-1

Example. Set word marks at locations BEGIN1 (3950) and BEGIN2 (3970), Figure 43.

SPS															
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d			
				ADDRESS	±	CHAR. ADJ.	10	ADDRESS	±	CHAR. ADJ.	10				
3	5	6	7	8	13	14	15	16	17	23	24	25	26	27	28
0	1	0													
			SW	BEGIN1						BEGIN2					

Autocoder										
Label		Operation		OPERAND						
6	15	16	20	21	25	30	35	40	45	50
		SW	BEGIN1, BEGIN2							

Assembled Instruction: 150 170

Figure 43. Set Word Mark (Two Addresses)

Set Word Mark (One Address)

Instruction Format.

Mnemonic Op Code A-address
SW XXX

Function. This format of the SET WORD MARK instruction causes a word mark to be set at the A-address. Data at this address is undisturbed.

Word Marks. A word mark is set at the A-address.

Timing. $T = N (L_I + 3)$ ms.

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI A-1 A-1

Example. Set a word mark at AREA2 (2901), Figure 44.

SPS															
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d			
				ADDRESS	±	CHAR. ADJ.	10	ADDRESS	±	CHAR. ADJ.	10				
3	5	6	7	8	13	14	15	16	17	23	24	25	26	27	28
0	1	0													
			SW	AREA2											

Autocoder									
Label	Operation	OPERAND							
5	15	20	25	30	35	40	45	50	55
	SW	AREA2							

Assembled Instruction: R01

Figure 44. Set Word Mark (One Address)

Clear Word Mark (Two Addresses)

Instruction Format.

Mnemonic Op Code A-address B-address
CW XXX XXX

Function. This instruction clears word marks at the locations specified by the A- and B-addresses, without disturbing the data there.

Word Marks. Word marks are cleared at the A- and B-addresses.

Timing. $T = N (L_I + 3)$ ms.

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI A-1 B-1

Example. Clear the word marks at NETPAY (1924) and ACCUM4 (3309), Figure 45.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				C			
				ADDRESS	±	CHAR. ADJ.	IND	ADDRESS	±	CHAR. ADJ.	IND				
3	5	6	7	8	13	14	15	16	17	23	24	25	26	27	28
0	1	0													
			CW	NETPAY											
										ACCUM4					

Autocoder										
Label	Operation	OPERAND								
6	15	16	20	21	25	30	35	40	45	
		CW	NETPAY, ACCUM4							

Assembled Instruction: Z24 C09

Figure 45. Clear Word Mark (Two Addresses)

Clear Word Mark (One Address)

Instruction Format.

Mnemonic Op Code A-address
CW XXX

Function. This format of the CLEAR WORD MARK instruction causes the word mark to be cleared at the A-address. Data at the A-address is not disturbed.

Word Marks. Word marks are cleared at the A-address only.

Timing. $T = N (L_I + 3)$ ms.

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI A-1 A-1

Example. Clear the word mark at RECNO1 (3608), Figure 46.

SPS															
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d			
				ADDRESS	±	CHAR. ADJ.	10	ADDRESS	±	CHAR. ADJ.	10				
3	5	6	7	8	13	14	15	16	17	23	24	25	26	27	28
0	1	0													
			CW	RECNO1											

Autocoder									
Label	Operation	OPERAND							
6	15	20	25	30	35	40	45	50	
	CW	RECNO1							

Assembled Instruction: F08

Figure 46. Clear Word Mark (One Address)

No Operation

Instruction Format.

Mnemonic Op Code
NOP N

Function. This code performs no operation. It can be substituted for the operation code of any instruction to make that instruction ineffective. It is commonly used in program modification to cause the machine to skip over specific instructions.

A-addresses of %xx and @xx should also be changed either to N's with associated word marks, or numeric characters, so that indexing and/or invalid core-storage addressing problems cannot occur.

Word Marks. The program operation resumes at the next operation code identified by a word mark.

Timing. $T = N (L_I + 1)$ ms.

Note. If characters without word marks follow an N operation code, these characters enter the A- and B-field registers. For example:

N 1234 A xxxx

In this instance, the address registers after operation would be:

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI 123 4bb

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI Ap Bp

Example. Leave one storage position open for an operation code such as READ CARD (1). Operation code 1 can be inserted if needed (Figure 47).

SPS													
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d	
3	5	6	7	8	13	14	15	16	17	23	24	25	26
0	1	0			N	O	P						

Autocoder													
Label	Operation	OPERAND											
6	15	16	20	21	25	30	35	40	45	50			
			N	O	P								

Assembled Instruction: N

Figure 47. No Operation

Halt

Instruction Format.

Mnemonic Op Code
H .

Function. This instruction causes the machine to stop and the stop-key light to turn on. Pressing the start key causes the program to start at the next instruction in sequence.

Word Marks. Word marks are not affected.

Timing. $T = N (L_I + 1)$ ms.

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI Ap Bp

Example. Figure 48 is a symbolic example of the HALT instruction.

SPS													
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d	
3	5	6	7	8	13	14	15	16	17	23	24	25	26
0	1	0			H								

Autocoder													
Label	Operation	OPERAND											
6	15	16	20	21	25	30	35	40	45	50			
			H										

Assembled Instruction: .

Figure 48. Halt

Halt and Branch

Instruction Format.

Mnemonic Op Code I-address
H . xxx

Function. This is the same as HALT, except that the next instruction is at the I-address.

Word Marks. Word marks are not affected.

Timing.

Without indexing:

$T = N (L_I + 1)$ ms.

With indexing:

$T = N (L_I + 2)$ ms.

Address Registers After Operation.

Without indexing

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI BI Blank

With indexing

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI BI NSI

Example. Stop the system, and branch to START2 (0895) for the next instruction when the start key is pressed (Figure 49).

LINE	COUNT	LABEL	OPERATION	(A) OPERAND						(B) OPERAND								
				ADDRESS		+	CHAR. ADJ.	IND.	ADDRESS		+	CHAR. ADJ.	IND.					
3	6	7	13	14	16	17	23	24	27	28	34	35	36	37	38	39		
0	1	0		H	START2													

Label		Operation	OPERAND							
6		1516	2021	25	30	35	40	45	5	
		H	START2							

Assembled Instruction: • 895

Figure 49. Halt and Branch

Edit Operation

The IBM 1401 and 1460 Data Processing Systems have a powerful edit instruction that can cause all desired commas, decimals, dollar signs, asterisks, credit symbols, and minus signs to be inserted automatically in a numerical output field. Unwanted zeros to the left of significant digits can be suppressed. Thus, editing in the IBM 1401 and 1460 is the automatic control of zero suppression, inserting identifying symbols, and punctuation of an output field (Figure 50).

	O P	A-address	B-address
Edit instruction	<u>E</u>	789	300
Storage		A-field (data) <u>00257426</u>	B-field (control word) <u>\$</u> bbb, bb0.bb & CR & **
Result of edit		<u>00257426</u>	B-field \$ 2,574.26 **

Figure 50. Editing

In editing, two fields are needed—the data field and a control field. The data field is the data edited for output. The control field specifies how the data field is edited. It specifies the location of punctuation and the condition of special characters, and indicates where zero suppression occurs. The two fields are compared, character by character, under the control of editing rules.

The control word has two parts: the body (which punctuates the A-field) and the status portion (which contains the dollar signs, sign-symbols, and class of total asterisks). The sign of the A-field determines whether or not sign symbols will print. The sign of the A-field is removed.

To edit a field, a LOAD CHARACTERS TO A WORD MARK instruction loads the control word into the specified printer output area. This puts the control word where the edited information will eventually go. Then, a MOVE CHARACTERS AND EDIT instruction (with the same B-address as the previous load instruction) performs

the editing function as it moves the data into the output area.

NOTE: A 1-position field cannot be edited. Figure 51 shows the use of these rules as applied to the data in Figure 50.

Move Characters and Edit

Instruction Format.

<i>Mnemonic</i>	<i>Op Code</i>	<i>A-address</i>	<i>B-address</i>
MCE	E	xxx	xxx

Function. The data field (A-field) is modified by the contents of the edit control field (B-field) and the result stored in the B-field. The data field and the control field are read from storage character-by-character, under control of the word marks and the editing rules. Any sign in the units position of the data field is removed during the operation.

Editing Rules.

Rule 1. All numerical, alphabetic, and special characters can be used in the control word. However, some of these characters have special meanings:

Control Character	Function
b (blank)	This is replaced with the character from the corresponding position of the A-field.
0 (zero)	This is used for zero suppression, and is replaced with a corresponding character from the A-field. Also the right-most "0" in the control word indicates the right-most limit of zero suppression.
• (decimal)	This remains in the edited field in the position where written. It is removed during a zero-suppress operation if it is to the left of the high-order significant digit. When used with the expanded print edit feature, it has an additional function (see <i>Expanded Print Edit in Special Features</i> , Form A24-3071).
, (comma)	This remains in the edited field in the position where written. It is removed during a zero-suppress operation if it is to the left of the high-order significant digit.
CR (credit)	This is undisturbed if the data sign is negative. It is blanked out if the data sign is positive. It can be used in body of control word without being subject to sign control.
- (minus)	This is the same as CR.
& (ampersand)	This causes a space in the edited field. It can be used in multiples.
* (asterisk)	This can be used in singular or in multiples, usually to indicate class of total. When it is used with the expanded print edit feature, it takes on an additional function (see <i>Expanded Print Edit in Special Features</i> , Form A24-3071).
\$ (dollar sign)	This is undisturbed in the position where it is written. When used with the expanded print edit feature, it has an additional function (see <i>Expanded Print Edit in Special Features</i> , Form A24-3071).

Cycle	TYPE OF CYCLE	ADDRESS REGISTERS			REG.		PUT BACK INTO STORAGE	"B" FIELD AT END OF CYCLE	REMARKS
		I	A	B	B	A			
1	I _{op}	002	?	?	<u>E</u>		<u>E</u>	\$ b b b , b b 0 . b b & C R & * *	Read Instr. OP Code
2	I ₁	003	07bb	07bb	7	7	7	same	Load A Address Register
3	I ₂	004	078b	078b	8	8	8	same	Load A Address Register
4	I ₃	005	0789	0789	9	9	9	same	Load A Address Register
5	I ₄	006	0789	03bb	3	3	3	same	Load B Address Register
6	I ₅	007	0789	030b	0	0	0	same	Load B Address Register
7	I ₆	008	0789	0300	0	0	0	same	Load B Address Register
8	I ₇	008	0789	0300	<u>OP</u>	0	<u>OP</u>	same	OP code of next instr.
9	A	008	0788	0300	6	6	6	same	Execute EDIT instr.
10	B	008	0788	0299	*	6	*	same	Rule 1
11	B	008	0788	0298	*	6	*	same	Rule 1
12	B	008	0788	0297	&	6	Blank	\$ b b b , b b 0 . b b & C R b * *	Rule 1
13	B	008	0788	0296	R	6	Blank	\$ b b b , b b 0 . b b & C b b * *	Rule 1 and 5
14	B	008	0788	0295	C	6	Blank	\$ b b b , b b 0 . b b & b b b * *	Rule 1 and 5
15	B	008	0788	0294	&	6	Blank	\$ b b b , b b 0 . b b b b b b * *	Rule 1
16	B	008	0788	0293	b	6	6	\$ b b b , b b 0 . b 6 b b b b * *	Rule 1
17	A	008	0787	0293	2	2	2	same	Rule 1
18	B	008	0787	0292	b	2	2	\$ b b b , b b 0 . 2 6 b b b b * *	Rule 1
19	A	008	0786	0292	4	4	4	same	Rule 1
20	B	008	0786	0291	.	4	.	same	Rule 1
21	B	008	0786	0290	0	4	4	\$ b b b , b b 4 . 2 6 b b b b * *	Zero Suppress—Rule 1 and 6
22	A	008	0785	0290	7	7	7	same	Rule 1
23	B	008	0785	0289	b	7	7	\$ b b b , b 7 4 . 2 6 b b b b * *	Rule 1
24	A	008	0784	0289	5	5	5	same	Rule 1
25	B	008	0784	0288	b	5	5	\$ b b b , 5 7 4 . 2 b b b b b * *	Rule 1
26	A	008	0783	0288	2	2	2	same	Rule 1
27	B	008	0783	0287	,	2	,	same	Rule 1
28	B	008	0783	0286	b	2	2	\$ b b 2 , 5 7 4 . 2 6 b b b b * *	Rule 1
29	A	008	0782	0286	0	0	0	same	Rule 1
30	B	008	0782	0285	b	0	0	\$ b 0 2 , 5 7 4 . 2 6 b b b b * *	Rule 1
31	A	008	0781	0285	<u>0</u>	<u>0</u>	<u>0</u>	same	Rule 1
32	B	008	0781	0284	b	<u>0</u>	0	\$ 0 0 2 , 5 7 4 . 2 6 b b b b * *	Rule 1
33	B	008	0781	0284	\$	<u>0</u>	\$	\$ 0 0 2 , 5 7 4 . 2 6 b b b b * *	Sense Word Mark—Rev. Scan—Rule 1 and 6
34	B	008	0781	0285	\$	<u>0</u>	\$	same	Rule 6
35	B	008	0781	0286	0	<u>0</u>	Blank	\$ b 0 2 , 5 7 4 . 2 6 b b b b * *	Rule 6
36	B	008	0781	0287	0	<u>0</u>	Blank	\$ b b 2 , 5 7 4 . 2 6 b b b b * *	Rule 6
37	B	008	0781	0288	2	<u>0</u>	2	same	Rule 6
38	B	008	0781	0289	,	<u>0</u>	,	same	Rule 6
39	B	008	0781	0290	5	<u>0</u>	5	same	Rule 6
40	B	008	0781	0291	7	<u>0</u>	7	same	Rule 6
41	B	008	0781	0292	<u>4</u>	<u>0</u>	4	\$ b b 2 , 5 7 4 . 2 6 b b b b * *	Rule 6

Figure 51. Step-by-Step Editing Operation

Rule 2. A word mark in the high-order position of the B-field controls the operation.

Rule 3. When the A-field word mark is sensed, the remaining commas in the control field are set to blanks.

Rule 4. The body of the control word is that portion beginning with the right-most blank or zero, and continuing to the left to the control character that governs the transfer of the last position of the data field. The remaining portion of the control field is the *status* portion.

Rule 7. Zero suppression is used if unwanted zeros to the left of significant digits in a data field are to be deleted (see Figure 52).

Zero suppression is the deletion of unwanted zeros at the left of significant digits in an output field (Figure 52).

A-field	0010900
Control word (B-field)	\$ bb, bb. bb
Forward scan	\$ 00,109. 00
Reverse scan	\$ bbb109. 00
Results of edit	\$ 109. 00

To perform zero-suppression operations properly, there must be at least one character to the left of the zero-suppression character in the control word.

1. The positions in the output field at the right of this special zero are replaced by the corresponding digits from the A-field.
2. The special zero is replaced by the corresponding digit from the A-field, when it is detected in the control field.
3. A word mark is automatically set in this position of the B- (output) field.
4. The scan continues until the B-field (high order) word mark is sensed and removed.

1. In the output field, blanks replace all zeros and punctuation, except hyphens at the left of the first significant character (up to and including the zero suppression code position).
2. When the automatically-set zero suppression word mark is sensed, it is erased and the operation ends.

	<i>I-Add. Reg.</i>	<i>A-Add. Reg.</i>	<i>B-Add. Reg.</i>
Without zero suppression	NSI	A-address minus the length of the A-field.	B-L _B
With zero suppression	NSI	A-address minus the length of the A-field.	Location of the special control zero plus 1.

Example. Edit the data labeled GROPAY (0985) by the edit-control word EDCONT (0325). Store the result in PRINT6 (0250), Figure 53.

[illegible]

Autocoder									
Label	Operation	OPERAND							
5	15	20	25	30	35	40	45	50	55
		MLCWAEDCONT,PRINTG							
		MCE GROPAV,PRINTG							
Assembled Instruction: L 325 250									
E 985 250									

This section describes the instructions the 1401 and 1460 use to control the IBM 1402 Card Read-Punch. The basic unit timings are also included with the other general information.

The data flow and feed schematic (Figure 54) shows the card feed transport areas for read and punch feeds, also the data flow for input and output of both feeds in the IBM 1402 Card Read-Punch (Model 1 for the 1401 system; Model 3 for the 1460 system).

The cards feed through the read side of the machine 9-edge first, face down. The feed path is from right to left, passing two sets of brushes. The read check brushes read 80 columns of the card to establish a hole count for checking purposes. The read brushes also read the 80 columns, prove hole count, and direct the data into core storage. At the end of the card transport path, three stackers are available to receive the cards. The normal read stacker is the stacker closest to the read hopper and is used unless the cards are program-directed to stackers 1 or 2.

The cards in the punch side of the IBM 1402 feed 12-edge first, face down. The feed path is from left to right, passing a blank station, a punching station, and a reading station. The punching station consists of 80

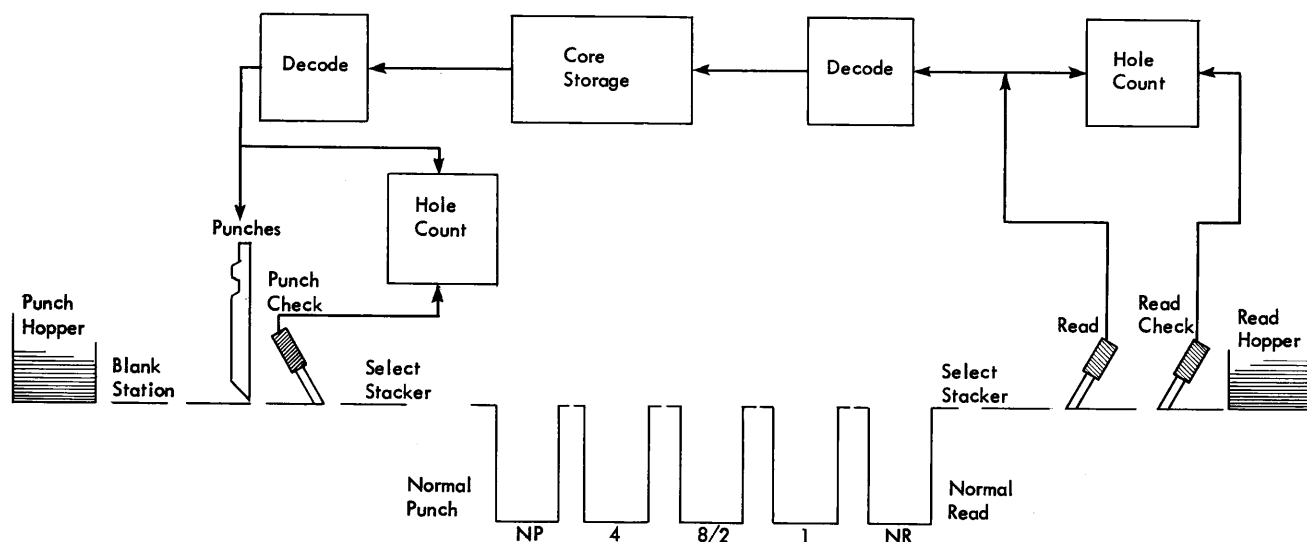


Figure 54. Data Flow—IBM 1402 Card Read-Punch

punches for recording information. The punch-reading brushes read all the holes punched in the 80 columns of the card for punch checking. This is a hole-count check between the information punched and the punches sensed at the punch brushes.

At the end of the card transport path on the punch side, three stackers are available to receive the cards. The normal punch stacker is used unless the cards are program-directed to stackers 4 or 8.

Card Reader

The card reader operates at a rated speed of 800 cycles per minute (one cycle every 75 milliseconds). The actual card speed realized is governed by the program routine for each particular run. (See *Card Read-Punch* timing.)

Card Punch

The card punch operates at a rated speed of 250 cycles per minute (240 milliseconds per cycle). Actual card punching, at optimum rate of 250 cards per minute, is controlled by punch instructions in the program.

IBM 1402 Card Read-Punch Instructions

Read Card

Instruction Format.

Mnemonic
R

Op Code
1

Function. This code causes a card to feed, and causes all 80 columns of information to be read into core-storage locations 001 through 080.

Word Marks. Word marks are undisturbed.

Timing. $T = N (L_I + 1) \text{ ms} + I/O.$

A card read cycle requires a total of 75 milliseconds. The cycle is divided into three separate sections (Figure 55).

1. Read start time is 21 ms. The read instruction must be given before card reading time in order to activate the card feed for that particular cycle. If the read instruction is given too late in the cycle, processing is delayed until the next card reading time occurs

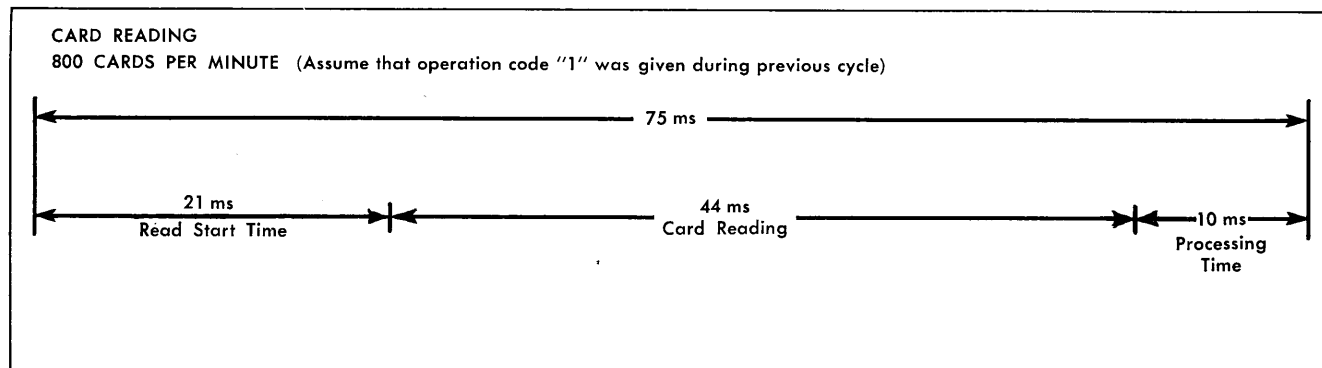


Figure 55. Card Read Cycle

3. Processing time is 10 ms. This part of the cycle is for processing. If *processing time* requires more than 10 milliseconds, the reader speed drops from 800 to 400 cards per minute.

2. Card punching time is 181 ms. The actual punching of the card takes place during this part of the cycle. The IBM 1401 or 1441 Processing Units are always interlocked during *card-punching time*.

3. Processing time is 22 ms. This is the remainder of the punch cycle that is allotted for *processing* by the system.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	Ap	181

Example. Feed a card, and punch (Figure 59).

SPS													
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d	
				ADDRESS	±	CHAR. ADJ.	182	ADDRESS	±	CHAR. ADJ.	182		
3	5	7	9	13	14	15	17	21	22	23	25	27	29
0	1	0		P									

Autocoder									
Label	Operation	20	21	25	30	35	40	45	50
6	P								

Assembled Instruction: 4

Figure 59. Punch Card

Punch Card and Branch

Instruction Format.

Mnemonic	Op Code	I-address
P	<u>4</u>	xxx

Function. This is the same as the PUNCH CARD instruction, except that the next instruction is taken from the I-address instead of from the next sequential instruction address. The branch occurs after the card has been punched.

Word Marks. Word marks are not affected.

Timing. $T = N (L_I + 1) \text{ ms} + I/O$.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	181

Example. Punch a card, and branch to START1 (1758), Figure 60.

SPS													
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d	
				ADDRESS	±	CHAR. ADJ.	IND-182	ADDRESS	±	CHAR. ADJ.	IND-182		
3	5	7	9	13	14	15	17	21	22	23	25	27	29
0	1	0		P									

Autocoder									
Label	Operation	20	21	25	30	35	40	45	50
6	P								

Assembled Instruction: 4 X58

Figure 60. Punch Card and Branch

Select Stacker

Instruction Format.

Mnemonic	Op Code	d-character
SS	<u>K</u>	x

Function. This instruction causes the card that was just read or punched to be selected into the stacker pocket specified by the d-character:

d-character	Feed	Stacker Pocket
1	READ	1
2	READ	8/2
4	PUNCH	4
8	PUNCH	8/2

Read Select. A SELECT STACKER instruction must be given during the first 10 ms after actual card reading is completed. Otherwise the command is ineffective. After a card is read, it continues to the stackers without stopping. Therefore, if no SELECT STACKER signal is received within the next 10 ms, the card stacks in the NORMAL stacker (NR). Read select instructions cannot be used following RP and WRP instructions because the select signal cannot be given within the prescribed 10 milliseconds.

Punch Select. The SELECT STACKER instruction is effective if given at any time between two PUNCH CARD instructions. However, if a punch check occurs, the error card is directed to the NORMAL (NP) stacker.

Word Marks. Word marks are not affected.

Timing. $T = N (L_I + 1) \text{ ms}$.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	dbb	dbb

Example. Enter the last card read into pocket 1 (Figure 61).

SPS													
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d	
				ADDRESS	±	CHAR. ADJ.	IND	ADDRESS	±	CHAR. ADJ.	IND		
3	5	7	9	13	14	15	17	21	22	23	25	27	29
0	1	0		SS									

Autocoder									
Label	Operation	20	21	25	30	35	40	45	50
6	SS								

Assembled Instruction: K 1

Figure 61. Select Stacker

Select Stacker and Branch

Instruction Format.

Mnemonic	Op Code	I-address	d-character
SPS SS	<u>K</u>	xxx	x
A SSB			

Function. This is the same as SELECT STACKER, except that the next instruction is taken from the I-address.

Word Marks. Word marks are not affected.

Timing. $T = N (L_I + 1)$ ms.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	dbb

Example. Select the last card punched, enter it in pocket 4, and branch to ROUT5 (0950) for the next instruction (Figure 62).

SPS												
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d
				ADDRESS	±	CHAR. ADJ.	IND	ADDRESS	±	CHAR. ADJ.	IND	
3	8	7	8	13	14	15	16	17	18	19	20	
0	1	0		SSB		ROUT5						

Autocoder											
Label	Operation	OPERAND									
		15	16	20	21	25	30	35	40	45	50
SSB	ROUT5	4									

Assembled Instruction: K 950 4

Figure 62. Select Stacker and Branch

Read and Punch

Instruction Format.

Mnemonic	Op Code
RP	<u>5</u>

Function. The functions of READ CARD (1) and PUNCH CARD (4) are combined in this operation code. The two operations overlap and, in effect, occur simultaneously. See Figure 69 for timing details.

Word Marks. Word marks are not affected.

Timing. $T = N (L_I + 1)$ ms + I/O.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	Ap	181 or 081

NOTE: The B-address register normally stands at 181 unless reading is completed after punching. In this case the B-address register contains 081.

Example. Read a card into the input area, and punch a card with information from the punch area (Figure 63).

SPS												
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d
				ADDRESS	±	CHAR. ADJ.	IND.	ADDRESS	±	CHAR. ADJ.	IND.	
3	8	7	8	13	14	15	16	17	18	19	20	21
0	1	0		RP								

Autocoder											
Label	Operation	OPERAND									
		15	16	20	21	25	30	35	40	45	50
RP											

Assembled Instruction: 5

Figure 63. Read and Punch

Read, Punch, and Branch

Instruction Format.

Mnemonic	Op Code	I-address
RP	<u>5</u>	xxx

Function. This is the same as the READ AND PUNCH instruction except that the next instruction is located at the I-address.

Word Marks. Word marks are not affected.

Timing. $T = N (L_I + 1)$ ms + I/O.

Address Registers After Operation.

I-Add. Reg.	A-Add. Reg.	B-Add. Reg.
NSI	BI	181 or 081

(See note under *Read and Punch*.)

Example. Read, punch, and branch to WORK5 (0596), Figure 64.

SPS																
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d				
				ADDRESS	±	CHAR. ADJ.	IND	ADDRESS	±	CHAR. ADJ.	IND					
3	8	6	7	8	13	14	15	16	17	23	27	28	34	35	36	37
RP WORK5																

Autocoder											
Label	Operation	OPERAND									
		15	16	20	21	25	30	35	40	45	50
RP											

Assembled Instruction: 5 596

Figure 64. Read, Punch, and Branch

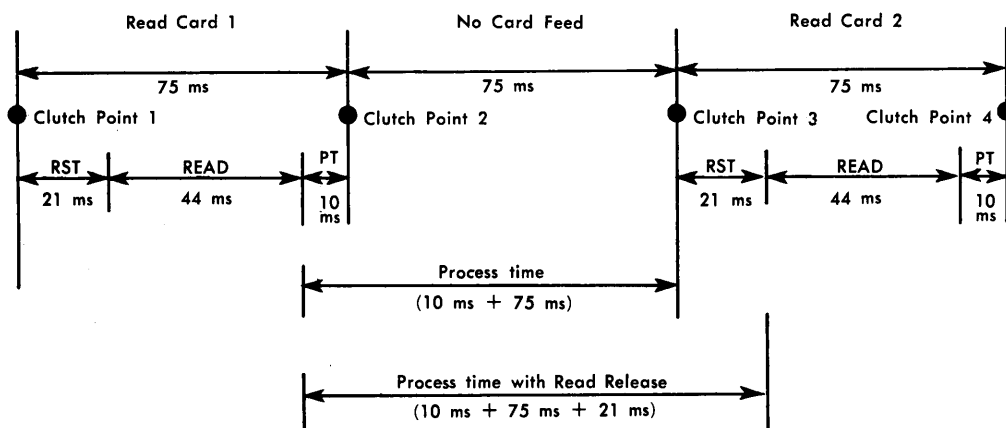


Figure 65. Read Operation Timing Chart

Card Read-Punch Timing

Card Reader

The feeding mechanism of the card reader is controlled by a single tooth clutch that completes one revolution every 75 ms. If the clutch receives a signal to read, the feed reads a card as soon as the clutch engages. Clutch engaging points are 75 ms apart.

A READ CARD instruction is normally given during the last 10 ms of a feed cycle, which is the normal process time. If the process time exceeds the 10 ms allowed, but is less than 85 ms, card feeding stops for a period of 75 ms (Figure 65). The timing chart shows that two card-read cycles (150 ms) are required to complete such an operation. When this condition is effective, 85 ms are available for processing.

Figure 66 is a table showing card reading speeds, and the processing time allotted for each.

Cards Minute Per	Length Of Cycle (ms)	Processing Time (ms)
800	75	10
400	150	85
266	225	160
200	300	235

Figure 66. Card Reading Speeds

Card Punch

The card punch is operated by a 4-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, during which the punch feed can be engaged to operate. As soon as the PUNCH CARD instruction is interpreted by the program, a signal is sent to the clutch. When the clutch reaches a clutch point, punch start time (PST) begins, followed by punch time and processing time.

If the punch clutch is impulsed during the 22-ms processing time each cycle, the punch operates continuously. If the punch clutch is not impulsed during the 22-ms processing time the punch cycle is delayed for 60 ms. This increases the processing time to 82 ms (Figure 67).

Figure 68 shows card punching speeds and the processing time available with each.

Combination Read and Punch

The 1401 or 1460 can read 250 cards per minute and punch 250 cards per minute if the READ AND PUNCH instruction (operation code 5) is used. The entire read operation is overlapped with card punching during the 240-ms punch cycle. The 22 ms normally available for processing during a punch cycle are also available during this combination operation.

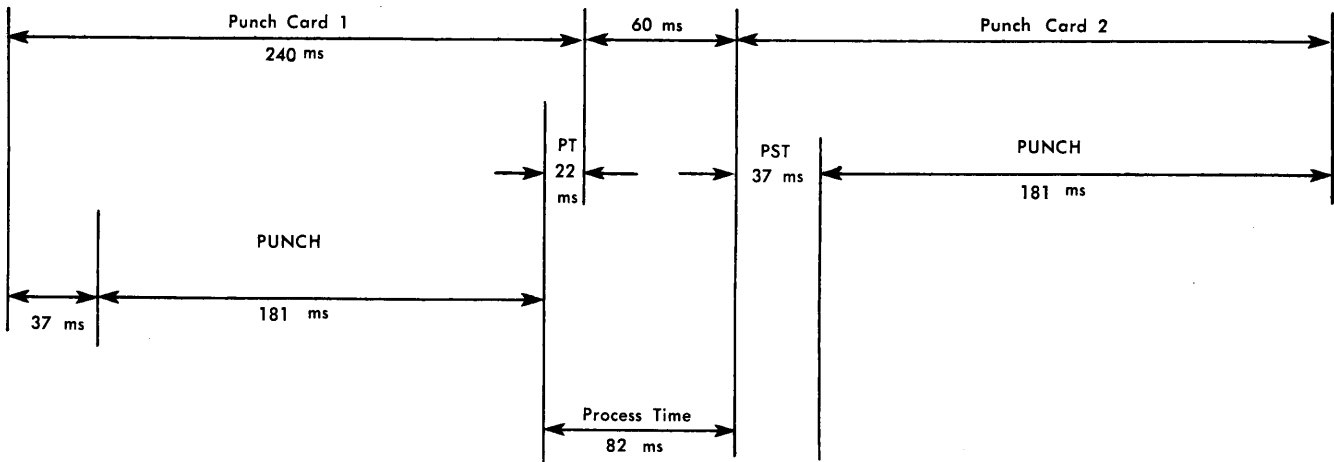


Figure 67. Punch Operation Timing Chart

Cards Punched Per Minute	Length of Cycle (ms)	Processing Time (ms)
250	240	22
200	300	82
166	360	142
143	420	202
125	480	262

Figure 68. Card Punching Speeds

Figure 69 is a chart showing the relationship between the read and punch operations.

NOTE: The position of the punch clutch can cause a maximum delay of 60 ms before the first read and punch operation is initiated. However, if processing time is kept within the prescribed time limits, continued use of the READ AND PUNCH instruction permits operation at 250 cards per minute.

IBM 1403 Printer Operations

This section describes the instructions the 1401 and 1460 use to control the IBM 1403 Printer. The basic unit

timings are also included with the other general information.

Data Flow

A printing operation requires moving and arranging of data into the core-storage print area (locations 201 through 300 for the 1401 or 201 through 332 for the 1401 and/or the 1460) before a printing operation is executed.

When a WRITE LINE instruction is given, the data to be printed is read out of core storage to the B-register, character by character. As each character is read into the B-register from a particular core-storage position, it is compared in the print compare area to the characters on the chain in the corresponding print position. When the comparison is equal, the hammer is fired, printing that character.

IBM 1403 Printer, Models 1 and 2

The IBM 1403 Printer, Models 1 and 2, operates at a maximum rate of 600 lines per minute, and prints only when the printer is impulsed to print. The print cycle is started when needed, and the printer immediately starts to print at the beginning of the cycle. The 100-millisecond print cycle is subdivided (Figure 70):

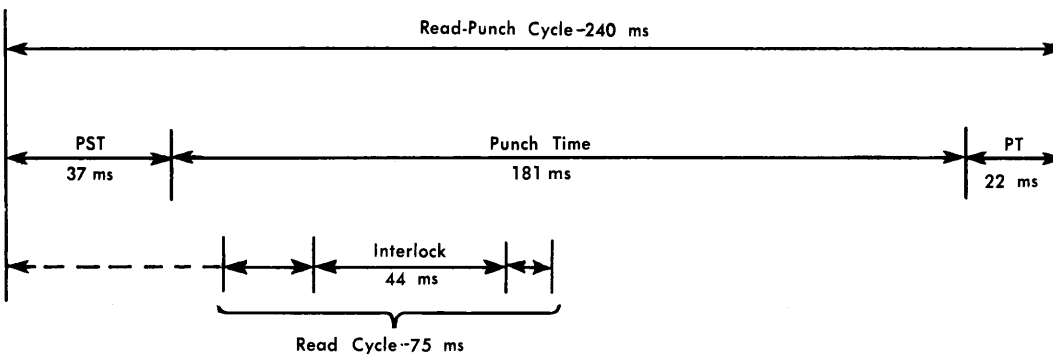


Figure 69. Read and Punch Operation Timing Chart

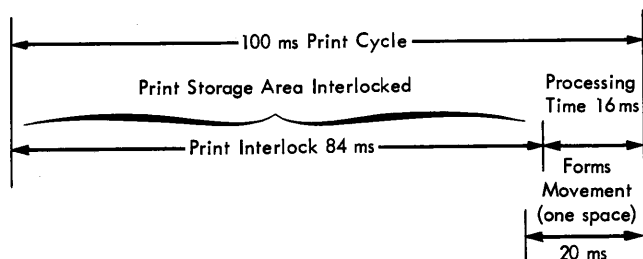


Figure 70. Print Operation Timing Chart

1. Print time is 84 ms. The line is printed during this part of the cycle. The IBM 1401 and 1441 Processing Units are interlocked during *print time* unless the print storage special feature is employed.
2. Process time is 16 ms. This is the normal *processing time* available during the print cycle (see *Print Storage in Special Features*, Form A24-3071).
3. Form movement time is 20 ms. The normal *form movement time* (one space) is always overlapped by processing time. Skipping time is not overlapped and must be included in the calculation of total program time.

An internal check is performed to ensure that the character set up at the print mechanism is the same as that sent from storage.

IBM 1403 Printer, Model 3

The IBM 1403 Printer, Model 3, operates at a rate of 1100 lines per minute. The 54.5-millisecond print cycle is divided into three parts (Figure 71):

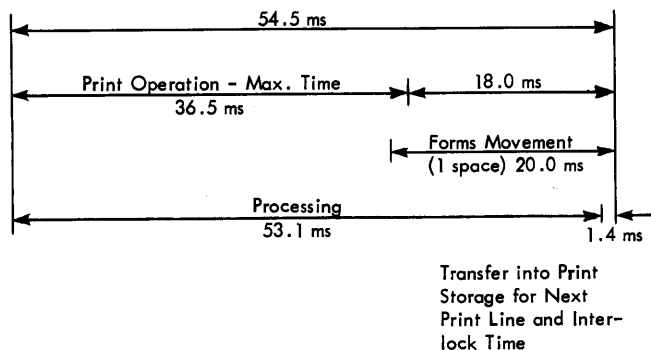


Figure 71. Print Operation Timing Chart—IBM 1401, Model 3

1. Print time is 36.5 ms. The line prints during this part of the cycle. The 1460 system is not interlocked during this time because the print storage feature is required on a 1460 system with a 1403 Model 3 attached.
2. Process time is 53.1 ms. This is the normal *processing time* available during the print cycle.

3. Form-movement time is 20 ms for a single space. The normal form-movement time is always overlapped by processing time.

IBM 1403 Printer Instructions

The instructions described here relate to the printing operations of the IBM 1403 Printer.

Write Line

Instruction Format.

Mnemonic Op Code
W 2

Function. This instruction causes the data in the print area to be transferred to the printer. The program continues after printing is complete. The printer takes one automatic space after printing a line.

Word Marks. Word marks are not affected.

Timing. $T = N (L_1 + 1) \text{ ms} + I/O.$

Note: The normal 84-ms interlock (1403, Models 1 and 2) during printing can be greatly reduced, if the print storage special feature is installed (see *Special Features*, Form A24-3071).

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI Ap 335
333 (print storage)

Example. Print the data in the print area (Figure 72).

SPS											
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND			
				ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±	CHAR. ADJ.	±
3	8	7	8	13	14	15	16	17	18	19	20
0	1	0									

Autocoder											
Label	Operation	OPERAND									
		15	16	17	18	19	20	21	22	23	24
8	16	17	18	19	20	21	22	23	24	25	26

Assembled Instruction: 2

Figure 72. Write Line

Write Line and Branch

Instruction Format.

Mnemonic Op Code I-address
W 2 xxx

Function. This is the same as the WRITE LINE instruction, except that the next instruction after printing is taken from the I-address.

Word Marks. Word marks are not affected.

Timing. $T = N (L_1 + 1) \text{ ms} + I/O.$

Address Registers After Operation.

<i>I-Add. Reg.</i>	<i>A-Add. Reg.</i>	<i>B-Add. Reg.</i>
NSI	BI	335
		333 (print storage)

Example. Print, and branch to STREC (0678) for the next instruction (Figure 73).

SPS																		
LINE		COUNT	LABEL		OPERATION	(A) OPERAND						(B) OPERAND						
						ADDRESS			+	CHAR.	IND.	ADDRESS			+	CHAR.	IND.	
3	8	6	7	9	13	14	16	17	23		27	30		34		38	39	d
O	I	O			W	S	T	R	E	C								

Autocoder									
Label	Operation	OPERAND							
5	15	20	25	30	35	40	45	50	55
	W	STREC							

Assembled Instruction: 2 678

Figure 73. Write Line and Branch

Write Word Marks

Instruction Format.

<i>Mnemonic</i>	<i>Op Code</i>	<i>d-character</i>
SPS W	<u>2</u>	□
A WM		

Function. The word marks associated with storage addresses in the print area print as the digit 1 in the corresponding print positions. After printing, the machine takes an automatic space, unless otherwise impulsed. The □ causes the word marks to be transferred to the printer.

Word Marks. Word marks remain in their original positions in the print area.

Timing. $T = N (L_I + 1) \text{ ms} + \text{I/O.}$

Note: On 1460 systems, the WRITE WORD MARKS instruction also causes:

1. group marks associated with storage addresses in the print area to print as a digit 2 in the corresponding print positions, and,
2. group-marks with word-marks associated with storage addresses in the print area to print as a digit 3 in the corresponding print positions.

Address Registers After Operation.

<i>I-Add. Reg.</i>	<i>A-Add. Reg.</i>	<i>B-Add. Reg.</i>
NSI	dbb	335
		333 (print storage)

Example. Print all word marks in the print area (Figure 74).

[illegible]

Autocoder									
Label	Operation	OPERAND							
5	15-19	20-21	25	30	35	40	45	50	55
	WM								

Assembled Instruction: 2 ☐

Figure 74. Write Word Marks

Write Word Marks and Branch

Instruction Format.

<i>Mnemonic</i>	<i>Op Code</i>	<i>I-address</i>	<i>d-character</i>
SPS W	<u>2</u>	xxx	□
A WM			

Function. This the same as WHITE WORD MARKS (2 □) instruction, except that the next instruction is taken from the I-address instead of from the next sequential instruction.

Word Marks. Word marks remain in their original positions in the print area.

Timing. $T = N (L_I + 1) \text{ ms} + \text{I/O.}$

Address Registers After Operation.

<i>I-Add. Reg.</i>	<i>A-Add. Reg.</i>	<i>B-Add. Reg.</i>
NSI	BI	335
		333 (print storage)

Example. Print word marks, and branch to RESTAR (0890), Figure 75.

SPS														
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d		
				ADDRESS	2	CHAR. ADJ.	IND.	ADDRESS	2	CHAR. ADJ.	IND.			
3	0	0	0	13	14	15	17	18	19	20	21	22	23	24
0	0	0	0	W. RESTAR										

Autocoder									
Label	Operation	OPERAND							
5	15-19	20-24	25	30	35	40	45	50	80
	WM	RESTAR							

Assembled Instruction: 2 890 ☐

Figure 75. Write Word Marks and Branch

Control Carriage

Instruction Format.

<i>Mnemonic</i>	<i>Op Code</i>	<i>d-character</i>
CC	F	x

Function. This instruction causes the carriage to move as specified by the d-character. A digit causes an immediate skip to a specified channel in the carriage tape. An alphabetic character with a 12-zone

d	Immediate skip to	d	Skip after print to
1	Channel 1	A	Channel 1
2	Channel 2	B	Channel 2
3	Channel 3	C	Channel 3
4	Channel 4	D	Channel 4
5	Channel 5	E	Channel 5
6	Channel 6	F	Channel 6
7	Channel 7	G	Channel 7
8	Channel 8	H	Channel 8
9	Channel 9	I	Channel 9
0	Channel 10	?	Channel 10
#	Channel 11	•	Channel 11
@	Channel 12	□	Channel 12
d	Immediate space	d	After print-space
J	1 space	/	1 space
K	2 spaces	S	2 spaces
L	3 spaces	T	3 spaces

Figure 76. d-Characters for Control Carriage

causes a skip to a specified channel after the next line is printed. An alphabetic character with an 11-zone causes an immediate space. A zero-zone character causes a space after the next line is printed. The table (Figure 76) shows the function of the d-character. If the carriage is in motion when a CONTROL CARRIAGE instruction is given, the program stops until the carriage comes to rest. At this point, the new carriage action is initiated, and then the program advances to the next instruction in storage.

Word Marks. Word marks are not affected.

Timing. $T = N (L_I + 1)$ ms plus remaining form-movement time, if carriage is moving when this instruction is given. The form-movement time is determined by the number of spaces the form moves. Allow 20 ms for the first space, plus 5 ms for each additional space.

Note: Care should be taken when punching channels 9 and 12 of the carriage control tape on a system with the print storage feature to ensure that these punches will not be ignored by line spacing (single, double, or triple). This is necessary because the next tape channel sensed resets the channel 9 and the channel 12 indicators. The next hole punched in the carriage tape following a 9- or 12-punch should be at least 1 + S spaces from the 9- or 12-punch (S equals number of lines being spaced).

The time required to perform a skip operation to channel 9 or 12 must be considered, so that the skip operation is completed when the test for the channel-9 or channel-12 hole is performed. If a BRANCH IF CARRIAGE CHANNEL #9 (or #12) instruction is executed before the skip operation to that channel hole is complete, the program does not branch to the specified subroutine, but goes to the next sequential instruction.

When an immediate skip or immediate space instruction is used, an additional space caused by the automatic carriage space taken after printing results. When a skip after print or space after print instruction is used, the automatic space is not taken.

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI dbb dbb

Example. Skip to channel 1 after print (Figure 77).

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d
				ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±	CHAR. ADJ.	±	
3	9	8	7	6	5	4	3	2	1	0	9	8
0	1	0										A

Autocoder

Label	Operation	25	30	35	40	45	50
	CC	A					

Assembled Instruction: F A

Figure 77. Control Carriage

Control Carriage and Branch

Instruction Format.

Mnemonic	Op Code	I-address	d-character
SPS CC	<u>F</u>	xxx	x
A CCB			

Function. This format of the CONTROL CARRIAGE instruction causes a program branch to the location specified by the I-address for the next instruction after interpretation of the d-character.

Word Marks. Word marks are not affected.

Timing. $T = N (L_I + 1)$ ms plus remaining form-movement time, if carriage is moving when this instruction is given. The form-movement time is determined by the number of spaces the form moves. Allow 20 ms for the first space, plus 5 ms for each additional space.

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI BI dbb

Example. Skip to channel 1 immediately, and branch to START3 (0498) for the next instruction (Figure 78).

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d
				ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±	CHAR. ADJ.	±	
3	9	8	7	6	5	4	3	2	1	0	9	8
0	1	0										A

Autocoder

Label	Operation	25	30	35	40	45	50
	CCB	START3	1				

Assembled Instruction: F 498 1

Figure 78. Control Carriage and Branch

Combination Instructions

It is often practical to combine two or three input-output functions in one instruction. Several special operation codes are provided to make it possible to perform some operations simultaneously. Each combination instruction has a corresponding instruction that permits automatic branching to a predetermined instruction address after the functions are complete.

Write and Read

Instruction Format.

Mnemonic Op Code
WR 3

Function. This instruction combines the functions of READ CARD (1) and WRITE LINE (2). The printer takes priority, and the print cycle is completed before the actual card reading operation takes place. However, the signal to start the reader can be accepted before the end of the print cycle. Thus, read start time overlaps the print cycle. See Figure 90 for timing details.

Word Marks. Word marks are not affected.

Timing. $T = N (L_I + 1) \text{ ms} + \text{I/O.}$

Note: If the system is equipped with the print storage special feature, the read operation can be performed as soon as the data is received in print storage.

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI Ap 081

Example. Print a line, and read a card Figure 79.

SPS													
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d	
3	9	8	7	13	14	15	16	17	18	19	20	21	22
0	1	0		WR									

Autocoder													
Label	Operation	25	30	35	40	45	50	OPERAND					
WR													

Assembled Instruction: 3

Figure 79. Write and Read

Write, Read, and Branch

Instruction Format.

Mnemonic Op Code I-address
WR 3 xxx

Function. This is the same as the WRITE AND READ instruction, except that the next instruction is taken from the I-address.

Word Marks. Word marks are not affected.

Timing. $T = N (L_I + 1) \text{ ms} + \text{I/O.}$

(See note under Write and Read.)

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI BI 081

Example. Print a line, read a card, and branch to CALC2 (0759), Figure 80.

SPS													
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d	
3	9	8	7	13	14	15	16	17	18	19	20	21	22
0	1	0		WR									

Autocoder													
Label	Operation	25	30	35	40	45	50	OPERAND					
WR													

Assembled Instruction: 3 759

Figure 80. Write, Read, and Branch

Write and Punch

Instruction Format.

Mnemonic Op Code -
WP 6

Function. This code combines the WRITE LINE (2) and PUNCH CARD (4) instructions. The printer always operates first, but the signal to start the punch is automatically given before the end of the print operation, so that actual card punching starts soon after the print cycle is complete. See Figure 91 for timing details.

Word Marks. Word marks are not affected.

Timing. $T = N (L_I + 1) \text{ ms} + \text{I/O.}$

Note: If the print storage special feature is installed, the signal to start the punch is received shortly after the transfer of data to the print storage area.

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI Ap 181

Example. Print a line, and punch a card (Figure 81).

SPS													
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND				d	
3	9	8	7	13	14	15	16	17	18	19	20	21	22
0	1	0		WP									

Autocoder													
Label	Operation	25	30	35	40	45	50	OPERAND					
WP													

Assembled Instruction: 6

Figure 81. Write and Punch

Write, Punch, and Branch

Instruction Format.

<i>Mnemonic</i>	<i>Op Code</i>	<i>I-address</i>
WP	6	xxx

Function. This is the same as WRITE AND PUNCH, except that the program branches automatically to the location in the I-address after punching is completed.

Word Marks. Word marks are not affected.

Timing. $T = N (L_I + 1) \text{ ms} + \text{I/O.}$

Note: If the print storage special feature is installed, the signal to start the punch is received shortly after the transfer of data to the print storage area.

Address Registers After Operation.

<i>I-Add. Reg.</i>	<i>A-Add. Reg.</i>	<i>B-Add. Reg.</i>
NSI	BI	181

Example. Print a line, punch a card, and branch to AREA8 (0895), Figure 82.

SPS																		
				(A) OPERAND					(B) OPERAND									
LINE	COUNT			LABEL	OPERATION			ADDRESS	÷	CHAR. ADJ.	IND.	ADDRESS	÷	CHAR. ADJ.	IND.			
3	5	6	7	8	13	14	15	16	17	23		27	28	34		38	39	
					WP	AREA 8												

Autocoder									
Label	Operation	OPERAND							
5	15	20	25	30	35	40	45	50	55
	WP	AREAR							

Assembled Instruction: 6 895

Figure 82. Write, Punch, and Branch

Write, Read, and Punch

Instruction Format.

<i>Mnemonic</i>	<i>Op Code</i>
WRP	7

Function. Printing, reading, and punching operations are performed when this command is given. The printer takes priority, and the reading and punching cycles start before the end of the actual print operation. See Figure 92 for timing details.

Word Marks. Word marks are not affected.

Timing. $T = N (L_I + 1) \text{ ms} + \text{I/O.}$

Address Registers After Operation.

<i>I-Add. Reg.</i>	<i>A-Add. Reg.</i>	<i>B-Add. Reg.</i>
NSI	Ap	181
		or
		081

(See note under *Read and Punch*).

Example. Print a line, read a card, and punch a card (Figure 83).

[illegible]

Autocoder									
Label	Operation	OPERAND							
5	15	16	20	21	25	30	35	40	45
	VRP								

Assembled Instruction: 7

Figure 83. Write, Read, and Punch

Write, Read, Punch, and Branch

Instruction Format.

<i>Mnemonic</i>	<i>Op Code</i>	<i>I-address</i>
WRP	7	xxx

Function. This is the same as WRITE, READ, AND PUNCH except that the next instruction, after punching is completed, is taken from the I-address.

Word Marks. Word marks are not affected.

Timing. $T = N (L_I + 1) \text{ ms} + \text{I/O.}$

Note: The B-address register normally stands at 181 unless reading is completed after punching. In this case the B-address register contains 081.

Address Registers After Operation.

<i>I-Add. Reg.</i>	<i>A-Add. Reg.</i>	<i>B-Add. Reg.</i>
NSI	BI	181
		or
		081

Example. Branch to ROUT4 (0980) after a line is printed, a card is read, and a card is punched (Figure 84).

SPS																													
LINE				COUNT				LABEL				OPERATION				(A) OPERAND						(B) OPERAND						C	
																ADDRESS		+	CHAR. ADJ.	IND.	ST	ADDRESS		+	CHAR. ADJ.	IND.	ST		
3	5	6	7	8	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32					
WRP ROUT 4																													

Autocoder									
Label		Operation				OPERAND			
5	15	16	20	21	25	30	35	40	45
			WRP ROUT4						

Assembled Instruction: 7 980

Figure 84. Write, Read, Punch, and Branch

IBM 1403 Printer Timings

Models 1 and 2

The IBM 1403 Printer, Models 1 and 2, operates at a maximum rated speed of 600 lines per minute. The 100-ms print cycle allows 16 ms of processing time.

Print

The 1401 and/or 1460 system is interlocked for 84 ms print time (Figure 85). The print storage area, however, is interlocked for the 84 ms during printing. Forms movement for single spacing is overlapped during the last 20 ms of the cycle. If additional forms-movement time or additional processing time is required by the application, this time must be added to the 100-ms cycle to determine printing speed.

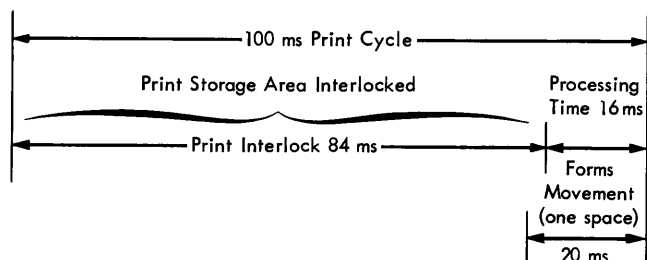


Figure 85. Print Operation Timing Chart

Figure 86 shows the effective printing speeds under various processing and forms-movement considerations.

Additional forms-skipping time beyond the first 8 lines is calculated by multiplying the number of lines skipped by 2.3 ms.

Some program instructions cause form movement to start immediately. If the printer is printing when an immediate forms control instruction is given, or if the carriage is already in motion, the system waits until the previous carriage operation is completed before the immediate skip is executed.

Immediate skips require 20 ms for the first space, 5 ms for each additional space up to 8, and then 2.3 ms

Lines Printed Per Minute	Length of Cycle (ms)	Available Process Time (ms)	Maximum Spaces Skipped Within Cycle
600	100	16	1
572	105	21	2
545	110	26	3
522	115	31	4
500	120	36	5
480	125	41	6
462	130	46	7
444	135	51	8

Figure 86. Effective Printing Speeds

LINES SKIPPED	TIME REQUIRED (ms)
1	20
2	25
3	30
4	35
5	40
6	45
7	50
8	55

Each space over 8 requires an additional 2.3 ms for models B, C, D. Model A requires 5 ms for each space after the first.

Figure 87. Form Movement Time

for each space thereafter, except Model A 1401 processing unit, which requires an additional 5 ms for each space after the first.

Figure 87 shows form-movement timing requirements for immediate-skip instructions.

Model 3

The IBM 1403 Printer, Model 3, operates at a maximum rated speed of 1100 lines per minute. The 54.5-ms print cycle (Figure 88) allows 53.1 ms of processing time because the print storage feature is required on a 1460 system with a 1403 Model 3 attached.

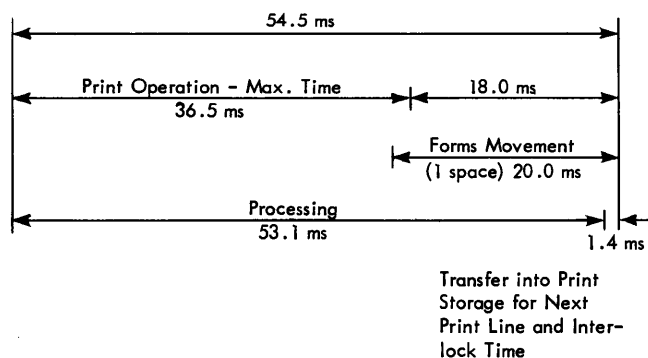


Figure 88. Print Operation Timing Chart—IBM 1403, Model 3

Forms movement for single spacing requires the last 20 ms of the print cycle. If additional forms movement time is required by the application, this time must be added to the 54.5-ms cycle to determine printing speed.

Figure 89 shows the effective printing speeds under various form-movement considerations.

Lines Printed Per Minute	Length of Cycle (ms)	Available Process Time (ms)	Maximum Spaces Skipped Within Cycle
1100	54.5	53.1	1
1007	59.5	58.1	2
930	64.5	63.1	3
863	69.5	68.1	4
805	74.5	73.1	5
755	79.5	78.1	6
710	84.5	83.1	7
670	89.5	88.1	8

Figure 89. Effective Printing Speeds—IBM 1403, Model 3

Additional form-skipping time beyond the first 8 lines is calculated by multiplying the number of lines skipped by 2.3 ms.

Some program instructions cause form movement to start immediately. If the printer is printing when an immediate forms-control instruction is given, or if the carriage is already in motion, the 1460 waits until the previous carriage operation is completed before the immediate skip is executed.

Immediate skips require 20 ms for the first space, 5 ms for each additional space up to 8, and then 2.3 ms for each space thereafter.

Refer to Figure 87 for the form-movement timing requirements for immediate-skip instructions.

Combination Input-Output Operations Timing

Total job-time improvements are often made through use of combination operation codes.

The first time a simultaneous operation is performed, the I/O cycle time may not exactly correspond to the cycle shown in the charts; but as the operation is continuously performed, the cycle time will be the same as the time specified in the timing charts.

Write and Read Operation

If reading and writing are to be done as a combined operation (operation code 3), the system can print 400

lines per minute and read 400 cards per minute. The combined cycle takes 150 ms; 18 ms are available for processing.

Figure 90 shows the division of the cycle, and the functions that are performed.

Note: Read start time extends for four milliseconds beyond print time.

The IBM 1403 Printer (Model 1 or 2) takes priority and operates first. The system is interlocked for the first 84 ms of the cycle. Forms movement can start 80 ms after printing begins and can continue until the end of the cycle. The read operation interlocks the processing unit for 44 ms. The remaining 18 ms are used for processing.

Installing the early card read and print storage special features increases the cards-and-lines-per-minute rate. The actual rate is dependent on the amount of process time required for the job.

Write and Punch Operation

The system can execute a combination WRITE AND PUNCH instruction (operation code 6) during a 300-ms cycle. Maximum output under these conditions is 200 lines and 200 cards per minute.

Figure 91 shows the relationship between writing and punching with 28 ms processing time available between successive operations. The IBM 1403 (Model 1 or 2) always takes priority and operates first. The signal to engage the clutch and initiate the punch operation is automatically given by the machine during the 84-ms print time when operation code 6 is used.

Write, Read, and Punch Operation

Cards can be read and punched at the rate of 200 cards per minute, and printing can occur at 200 lines per minute if the WRITE, READ, AND PUNCH instruction (oper-

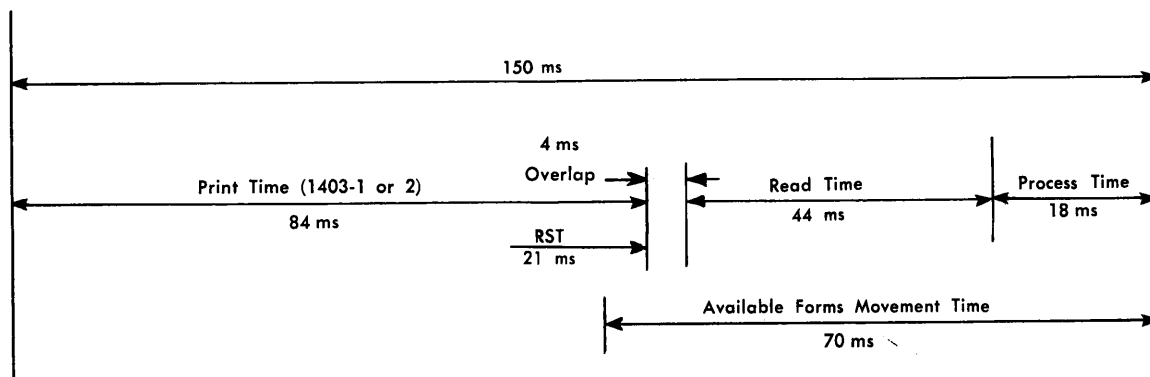
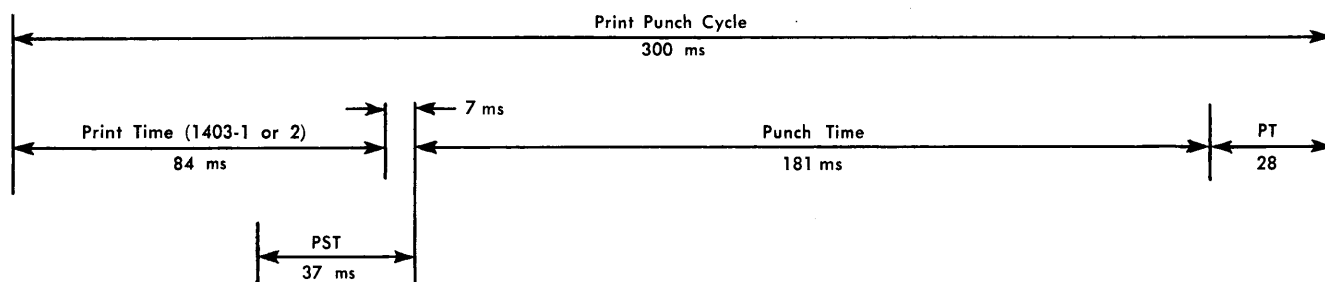


Figure 90. Write and Read Operation Timing Chart



Note: Punch start time extends for 7 ms beyond Print Time

Figure 91. Write and Punch Operation Timing Chart

ation code 7) is used. The IBM 1403 Printer (Model 1 or 2) takes priority and operates first. Reading time is completely overlapped with punching, and processing time available during the 300-ms cycle is 28 ms. The signals to start the reader and the punch are automatic in the combination instruction. Figure 92 shows the timing for the operation.

NOTE: The first WRITE, READ, AND PUNCH instruction could be extended to a maximum of 360 ms because of the clutch wait time. Subsequent successive WRITE, READ, AND PUNCH instructions require 300 ms each.

Multiple-Printer Output (IBM 1460)

It is now possible to attach two additional 1403 printers to an IBM 1460 Data Processing System. These additional printers greatly simplify applications that involve:

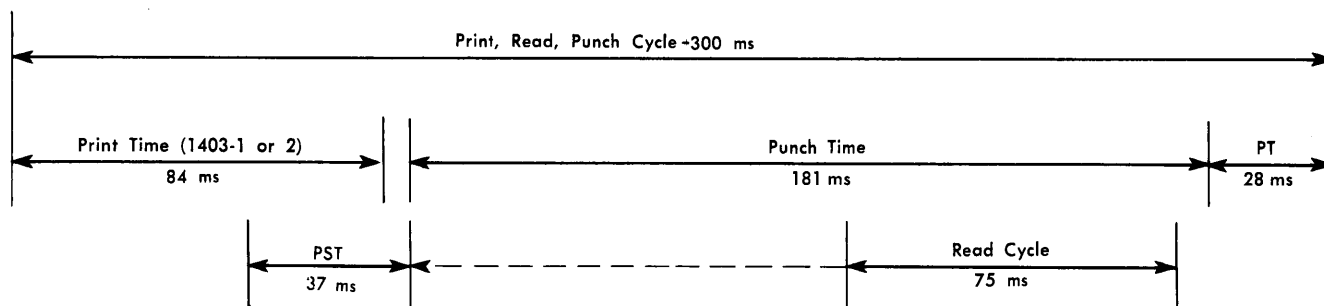
1. Off-line listing of engineering and scientific computation when the volume is too large for one printer.
2. Simultaneous running of several different reports from the same set of statistics.

A 1460 system equipped with multiple-printer output can have up to three separate printer-outputs attached. With the multiple-printer output capability, any combination of three IBM 1403 Printers, Models 2 and 3, can be attached to the system. The print storage special feature is a required feature on printer 1, and is a standard feature on the additional printers.

The first printer is attached to the system through the IBM 1461 Input-Output Control. Each additional printer has an IBM 1462 Printer Control Unit associated with it. The printer control unit contains the electronics and attachment circuitry necessary for operating the printer. (A printer control adapter—third printer special feature, must be installed on the first 1462 so that the second 1462 and its associated printer can be attached.)

Each printer attached to the system is assigned a unit number. The printer attached to the IBM 1461 Input-Output Control is always printer 1. The two additional printers are assigned unit numbers of 2 and 3. These numbers are assigned in the associated printer control unit by the customer engineer at installation time. If two 1403 Model 2's or Model 3's are switched, the assigned unit numbers stay with the printer control unit. After initial installation, the number of 1403 Model 2's and 1403 Model 3's installed on one system can be changed by a customer engineer only. The printer control units are internally wired to control either a 1403 Model 2 or a 1403 Model 3. (A printer adapter—1403 Model 3 special feature is required on the 1462 when it will control a 1403, Model 3.)

NOTE: Any 1403 features announced for the IBM 1461 Input-Output Control will also be available on the IBM 1462 Printer Control Unit.



Note: Punch start time extends for 7 ms beyond Print Time

Figure 92. Write, Read, and Punch Operation Timing Chart

Printer Selection

All the printers share the same output area, core-storage positions 201-332. A **PRINTER PRE-SELECT** instruction specifies which printer is to accept and print the data from core-storage positions 201-332. This instruction is a 2-character instruction and consists of an operation code and a d-modifier character. The operation code is the letter U with an associated word mark. The d-character is 1, 2, or 3 and specifies the associated printer unit number. This instruction is executed prior to executing any instruction involving a printer. The instruction activates the specified printer and deactivates the previously selected printer. The specified printer remains selected until the program selects another printer.

Either a system power-on, or a program-load operation involving an IBM 1402 Card Read-Punch, or a tape load operation involving a magnetic tape unit automatically selects and activates printer 1.

Programming Considerations

Any program written without printer pre-selection operates correctly with printer 1.

A single pre-selection card inserted in the program deck is all that is needed to use either printer 2 or printer 3. However, a **WRITE WORD MARKS** instruction is executed as a normal **WRITE A LINE** instruction on printers 2 and 3.

Operator Controls

The I/O check reset key on the console resets the check conditions for all the printers.

The check reset key on the printer resets the check conditions for that printer only.

Mode Switch Operation

A number of the operations initiated by the mode switch on the console will function on the additional printer(s), while other operations do not. The additional printer(s) functions the same as printer 1 when the mode switch is set to:

1. Single-cycle process
2. I/EX
3. Storage scan (when the auxiliary mode switch is also set to either print-storage scan 2 or print-storage scan 3).

Storage print-out and full-storage print are only operative on printer 1.

IBM 1447 Console Operations

The IBM 1447 Console (Model 1, 2, or 4) (Figure 93) is a required unit on an IBM 1460 Data Processing System. The console contains the system's operating



Figure 93. IBM 1447 Console, Model 2

keys, lights, and switches, which give the operator external control for setting up and checking system operation. For more detail on the keys, lights, switches, and operating procedures, refer to *IBM 1447 Console* (A24-3031).

The IBM 1447 Console, Model 3, (Figure 94) is available as a console inquiry station on a 1401 system. All the advantages of the IBM 1407 Console Inquiry Station, plus the increased print-out speed of 14.8 characters per second, provides the console operator with an efficient and time-saving method of inquiry and reply. For more detail on the keys, lights, and operating procedures, refer to *IBM 1447 Console*, Form A24-3031.



Figure 94. IBM 1447 Console, Model 3, with IBM 1409 Console Auxiliary

Console Instruction Format

A program-initiated data transmission between the IBM 1447 Console (Model 2, 3, or 4) and the attached system is started by executing the proper console instruction. The format for the IBM 1447 Console instruction is shown in Figure 95.

The various parts of a 1447 instruction and their uses are:

General Mode of Operation

This part of the instruction identifies the operation as either a move operation or a load operation. A move operation specifies that only the character coding is transmitted. A load operation specifies that both the character coding and any associated word marks are transmitted.

Operating Input-Output Unit

This part of the instruction specifies the console I/O printer as the active input-output unit for this operation.

B-Address

This part of the instruction specifies the first core-storage position involved in the operation.

d-Modifier Character

This part of the instruction specifies the data transmission direction. An R specifies a console printer-to-system data transmission; a W specifies a system-to-console printer data transmission.

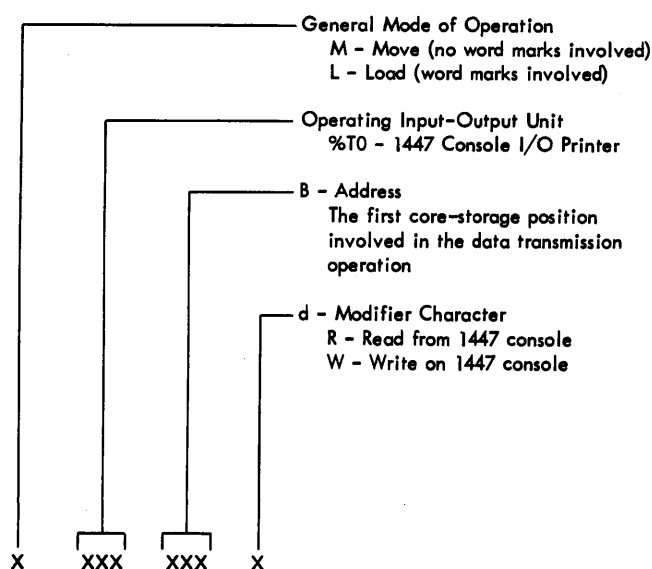


Figure 95. IBM 1447 Console I/O Printer Instruction Format

IBM 1447 Console Instructions

Read from 1447 Console

Instruction Format.

Mnemonic	Op Code	A-address	B-address	d-character
MU	<u>M</u>	%T0	BBB	R
LU	<u>L</u>			

Function. This instruction is used to enter data into core storage from the console I/O printer. The Op code specifies the mode of operation. If the operation takes place in the *move* mode (M Op code), word marks cannot be transmitted from the console printer into core storage. Any word marks already in the area that accepts the message will remain there.

If the operation takes place in the *load* mode (L Op code), word marks can be transmitted from the console printer into core storage when the word-mark key is pressed. Any word marks already in the area that accepts the message will be removed.

The A-address specifies the console I/O printer as the I/O unit involved in the operation. The B-address specifies the first core-storage position that will accept data from the console printer. The d-character specifies a console printer-to-system operation.

The console operator can start keying the data when the white type light on the console comes on. The console operator prints the data on the console printer and the characters enter core storage, beginning at the location specified by the B-address portion of the instruction.

The operator transmits a word mark by pressing the shift key and the word-mark key. The upper case (word-mark position) of the period key prints an inverted circumflex. The next character printed enters a core-storage position and has a word mark associated with it.

When the number of data positions to be entered into core storage exceeds the number of printing positions on one printer line, the print element automatically returns from the right-hand margin, executes a line-feed operation, and the keying operation continues on the next line.

The operation is normally ended when the operator presses the release key. This key operation inserts a group-mark with a word-mark in core storage, initiates a carrier-return and line-feed operation, and disconnects the printer from the system.

The operation can also be ended if a group-mark with a word-mark is sensed in core storage. This

5. The keyboard locks up.

51

The timing involved during an input operation is:
 $T = N (L_I + 1) + \text{console operator keying time.}$

The timing involved during an output operation is:
 $T = N (L_I + 1) + 68 (L_B) + 800 (\text{number of carrier return operations} - 1) \text{ ms.}^*$

*All system-console printer operations are unbuffered operations. Only one portion of either operation is overlapped by processing. This is the last carrier-return and line-feed operation that occurs at the end of an output operation.

IBM 1406 Storage Unit Operations

The IBM 1406 Storage Unit (Figure 98) provides a means of increasing a 4000-position 1401 system to 8000; 12,000; or 16,000 core-storage positions.

IBM 1406 Models

The 1406, Model 1, contains a block of 4000 core-storage positions. It increases the capacity of the system to 8000 positions.

Model 2 contains a block of 8000 core-storage positions. It increases the capacity of the system to 12,000 positions.

Model 3 contains a block of 12,000 core-storage positions. It increases the capacity of the system to 16,000 positions.

Addressing

The additional core-storage locations are addressed by the presence of zone bits located over the units position of each storage address over 3999. These zone bits are added to the basic 4000-character coding system.

Storage Addresses	Zone Bits over Units Position of Address
4,000- 7,999	A
8,000-11,999	B
12,000-15,999	AB

Address Validity

The IBM 1401 and 1460 check each address to ensure that it is valid for the storage capacity installed. The system stops on an address validity error, if an invalid address is encountered.

Core-Storage Capacity	Valid Addresses	Invalid Addresses
1401 { 1,400	0000-1399	1400-15999
Only { 2,000	0000-1999	2000-15999
{ 4,000	0000-3999	4000-15999
1401 { 8,000	0000-7999	8000-15999
and { 12,000	0000-11999	12000-15999
1460 { 16,000	0000-15999	NONE



Figure 98. IBM 1406 Storage Unit

Address Arithmetic

To facilitate address arithmetic, an additional operation code (MODIFY ADDRESS) is added to IBM 1401 systems equipped with more than 4000 characters of core storage and all 1460 systems.

IBM 1406 Storage Unit Instructions

Modify Address (Two Addresses)

Instruction Format.

Mnemonic	Op Code	A-address	B-address
MA	#	xxx	xxx

Function. This instruction adds the 3-character A-field (units position specified by the A-address) to the 3-character B-field (units position specified by the B-address). When applicable, the zone portions of the hundreds position and the units position of both fields are also added. The result of this numeric- and zone-bit addition is stored in the B-field. For example:

Location	Contents	3-Character Address	Actual Address
A-address	A-field	H9Y	7898
B-address	B-field	H09	3809
	Result	G0P	11707

Word Marks. Word marks are not affected, and are not required to define the A- or B-fields. If word marks are present, they are ignored and remain unchanged in both fields.

Timing.

If a carry from the hundreds position to the units position is required:

$$T = N (L_I + 9) \text{ ms.}$$

If no carry from the hundreds position to the units position is required:

$$T = N (L_I + 8) \text{ ms.}$$

Note: See rules for the addition of zone bits in the section, *Address Modification Without Indexing.*

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI A-3 B-1 or B-3

Example. Add the 3-character address labeled ADDA (0985) to the 3-character address labeled ADDB (1313), Figure 99.

SPS											
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND			
				ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±	CHAR. ADJ.	±
3	0	MA	ADDA	0985				1313			

Autocoder											
Label	Operation	ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±
MA	ADDA	0985				ADDB					

Assembled Instruction: # 985 T13

Figure 99. Modify Address (Two Addresses)

Modify Address (One Address)

Instruction Format.

Mnemonic Op Code A-address
MA # xxx

Function. This format of the MODIFY ADDRESS instruction causes the 3-character field, specified by the A-address, to be added to itself. The result is stored in the A-field.

Word Marks. Word marks are not required to define the A-field. If they are present, they are ignored and remain undisturbed in the A-field.

Timing.

If a carry from the hundreds position to the units position is required:

$$T = N (L_I + 9) \text{ ms.}$$

If no carry from the hundreds position to the units position is required:

$$T = N (L_I + 8) \text{ ms.}$$

Address Registers After Operation.

I-Add. Reg. A-Add. Reg. B-Add. Reg.
NSI A-3 A-1 or A-3

Example. Double the address labeled ADDC (2956), and store the result at ADDC (Figure 100).

SPS											
LINE	COUNT	LABEL	OPERATION	(A) OPERAND				(B) OPERAND			
				ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±	CHAR. ADJ.	±
3	0	MA	ADDC	2956							

Autocoder											
Label	Operation	ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±	CHAR. ADJ.	±	ADDRESS	±
MA	ADDC	2956									

Assembled Instruction: # R56

Figure 100. Modify Address (One Address)

Address Modification— Using the Modify Address Instruction

For 1401 systems with more than 4000 positions of core-storage (with the IBM 1406 Storage Unit), and all 1460 systems, A- and B-bit accumulation occurs in both the hundreds and units position. In these systems, the MODIFY ADDRESS instruction must be used. This instruction makes use of modulus 16 arithmetic to perform its operations.

In a modulus 16 system, the highest digit is 15. Values in excess of fifteen are equal to that value minus sixteen. For example, 16 is a digit 0, 17 is a digit 1, etc. In this system, only two factors can be accumulated at a time (Figure 101).

				Hundreds	Units
A + A	= B	or	2	8	
A + B	= AB	or	3	12	
B + B	= No A No B	or	0	0	
A + AB	= No A No B	or	0	0	
A + No A No B	= A	or	1	4	
B + AB	= A	or	1	4	
B + No A No B	= B	or	2	8	
AB + AB	= B	or	2	8	

Figure 101. Modulus 16 Arithmetic

Digit values in high- and low-order positions of a field accumulate in the normal manner.

The MODIFY ADDRESS instruction is standard in 1401 systems with more than 4000 core-storage positions, and all 1460 systems, and does not require word marks in the high-order positions of the addresses.

Increase address 7355 (C5V) by 1800 (Y00).

$C5V + Y00 = /5N$ (with arithmetic overflow)
C = (AB3)
V = (A5)
/ = (A1)
Y = (A8)
N = (B5)

To perform this address change, the MODIFY ADDRESS instruction must be used. Using the rules of modulus 16 arithmetic, the addition of the existing zone bits results in a No A, No B (0-zone value) in the hundreds position, and an A (zone value 4) in the units position. The addition of the AB and A-bits (= 0) in the hundreds position results in a zone carry-back (A-bit) to the units position. This A-bit added to the A-bit already there results in a B-bit (zone value 8) in the units posi-

tion. The arithmetic digit carry adds a new A-bit in the hundreds position. The result is address /5N (9155).

To decrease an address, only modulus 16 arithmetic is used. Because modulus 16 arithmetic operates for addition only, decreasing an address requires adding a complement, rather than doing a conventional subtract operation. The 16,000's complement of the decrement is added to the address to be modified. If the result is an address outside the storage limit of the system, an invalid address condition is indicated.

Decrease 1829 (Y29) by 161.

$16,000 - 161 = 15,839$ (H31) complement
 $Y29 + H31 = W68$ (with arithmetic overflow)
Y = A8
H = AB8
I = AB9

Using the modulus 16 rules and the MODIFY ADDRESS instruction, the addition of the existing zone bits results in zero (hundreds position, $A + AB = 4$; units position, $AB = 12$; $4 + 12 = 16$, which has an address value of 0). Then the arithmetic overflow adds a new A-bit in the hundreds position. The result is address W68 (1668).

PRINT ARRANGEMENT		DEFINED CHARACTER	CARD CODE	BCD CODE	PRINT ARRANGEMENT		DEFINED CHARACTER	CARD CODE	BCD CODE
A	H				A	H			
		BLANK		C	G	G	G	12-7	BA 421
.	.	.	12-3-8	BA 8 21	H	H	H	12-8	BA 8
□	)	□	12-4-8	CBA 84	I	I	I	12-9	CBA 8 1
		[Left Bracket (Special Character)	12-5-8	BA 84 1	—	—	! (Minus Zero)	11-0	B 8 2
		< Less Than (Special Character)	12-6-8	BA 84 2	J	J	J	11-1	CB 1
		≡ Group Mark (Note 1)	12-7-8	CBA 8421	K	K	K	11-2	CB 2
&	+	&	12	CBA	L	L	L	11-3	B 21
\$	\$	\$	11-3-8	CB 8 21	M	M	M	11-4	CB 4
*	*	*	11-4-8	B 84	N	N	N	11-5	B 4 1
		] Right Bracket (Special Char.)	11-5-8	CB 84 1	O	O	O	11-6	B 42
		; Semicolon (Special Character)	11-6-8	CB 84 2	P	P	P	11-7	CB 421
		Δ Delta (Mode Change)	11-7-8	B 8421	Q	Q	Q	11-8	CB 8
—	—	—	11	B	R	R	R	11-9	B 8 1
/	/	/	0-1	CA 1	≠	≠	≠ Record Mark	0-2-8	A 8 2
,	,	,	0-3-8	CA 8 21	S	S	S	0-2	CA 2
%	(	%	0-4-8	A 84	T	T	T	0-3	A 21
		✓ Word Separator	0-5-8	CA 84 1	U	U	U	0-4	CA 4
		\ (Special Character)	0-6-8	CA 84 2	V	V	V	0-5	A 4 1
		⏏ Tape Segment Mark	0-7-8	A 8421	W	W	W	0-6	A 42
≠	≠	⌘ (Special Character Note 2)		A	X	X	X	0-7	CA 421
#	=	#	3-8	8 21	Y	Y	Y	0-8	CA 8
@	'	@	4-8	C 84	Z	Z	Z	0-9	A 8 1
		: Colon (Special Character)	5-8	84 1	0	0	0	0	C 8 2
		> Greater Than (Special Character)	6-8	84 2	1	1	1	1	1
		√ Tape Mark	7-8	C 8421	2	2	2	2	2
&	&	? (Plus Zero)	12-0	CBA 8 2	3	3	3	3	C 21
A	A	A	12-1	BA 1	4	4	4	4	4
B	B	B	12-2	BA 2	5	5	5	5	C 4 1
C	C	C	12-3	CBA 21	6	6	6	6	C 42
D	D	D	12-4	BA 4	7	7	7	7	421
E	E	E	12-5	CBA 4 1	8	8	8	8	8
F	F	F	12-6	CBA 42	9	9	9	9	C 8 1

The system has the ability to read MLP card codes in the read feed only. The system ignores the 8-9 punches when they appear in the same column. The system does not punch out MLP card codes.

Note 1. If specified, this code can be made compatible with 705 Group Mark Code (12-5-8).

Note 2. The A-bit coding must be program-generated in the system (it cannot be read from a card; it can be punched as a zero). It is used in conjunction with the C-bit to indicate a blank position on tape that was written in even-bit parity.

Note 3. Other special character printing arrangements can be obtained.

Figure 102. IBM 1401/1460 Character Code Chart in Collating Sequence

Index of 1401/1460 Instructions

Instruction	SPS	Mnemonic	Autocoder	Form	Page
Add (One Field)		A		<u>A</u> (A)	20
Add (Two Fields)		A		<u>A</u> (A)(B)	19
Branch (Unconditional)		B		<u>B</u> (I)	23
Branch if Access Inoperable	B		BIN	<u>B</u> (I)N	23
Branch if Any Disk-Unit Error Condition	B		BIN	<u>B</u> (I)Y	23
Branch if Carriage Channel #9	B		BC9	<u>B</u> (I)9	23
Branch if Carriage Channel #12	B		BCV	<u>B</u> (I)@	23
Branch if Character Equal	B		BCE	<u>B</u> (I)(B)d	24
Branch if End of Reel	B		BEF	<u>B</u> (I)K	23
Branch if Equal Compare (B = A)	B		BE	<u>B</u> (I)S	23
Branch if High Compare (B > A)	B		BH	<u>B</u> (I)U	23
Branch if Inquiry Clear	B		BIN	<u>B</u> (I)*	23
Branch if Inquiry Request	B		BIN	<u>B</u> (I)Q	23
Branch if Last Card Switch (Sense Switch A)	B		BLC	<u>B</u> (I)A	23
Branch if Low Compare (B < A)	B		BL	<u>B</u> (I)T	23
Branch if Arithmetic Overflow	B		BAV	<u>B</u> (I)Z	23
Branch if Printer Busy (Print Storage Feature)	B		BPB	<u>B</u> (I)P	23
Branch if Printer Carriage Busy (Print Storage Feature)	B		BPCB	<u>B</u> (I)R	23
Branch if Printer Error (I/O Check Stop Switch Off)	B		BIN	<u>B</u> (I)≠	23
Branch if Processing Check (Check Stop Switch Off)	B		BIN	<u>B</u> (I)%	23
Branch if Punch Error (I/O Check Stop Switch Off)	B		BIN	<u>B</u> (I)!	23
Branch if Read-Write Parity Check or Read-Back Check Error	B		BIN	<u>B</u> (I)V	23
Branch if Reader Error (I/O Check Stop Switch Off)	B		BIN	<u>B</u> (I)?	23
Branch if Sense Switch B On	B		BSS	<u>B</u> (I)B	23
Branch if Sense Switch C On	B		BSS	<u>B</u> (I)C	23
Branch if Sense Switch D On	B		BSS	<u>B</u> (I)D	23
Branch if Sense Switch E On	B		BSS	<u>B</u> (I)E	23
Branch if Sense Switch F On	B		BSS	<u>B</u> (I)F	23
Branch if Sense Switch G On	B		BSS	<u>B</u> (I)G	23
Branch if Tape Transmission Error	B		BER	<u>B</u> (I)L	23
Branch if Unconditional		B		<u>B</u> (I)b	23
Branch if Unequal-Address Compare	B		BIN	<u>B</u> (I)X	23
Branch if Unequal Compare	B		BU	<u>B</u> (I)/	23
Branch if Word Mark		BWZ		<u>V</u> (I)(B)1	25
Branch if No Zone		BWZ		<u>V</u> (I)(B)2	25
Branch if 12-Zone		BWZ		<u>V</u> (I)(B)B	25
Branch if 11-Zone		BWZ		<u>V</u> (I)(B)K	25
Branch if Zero-Zone		BWZ		<u>V</u> (I)(B)S	25
Branch if Either a Word Mark, or No Zone		BWZ		<u>V</u> (I)(B)3	25
Branch if Either a Word Mark, or 12-Zone		BWZ		<u>V</u> (I)(B)C	25
Branch if Either a Word Mark, or 11-Zone		BWZ		<u>V</u> (I)(B)L	25
Branch if Either a Word Mark, or Zero Zone		BWZ		<u>V</u> (I)(B)T	25
Branch if Wrong-Length Record	B		BIN	<u>B</u> (I)W	23
Clear Storage		CS		<u>Z</u> (A)	29
Clear Storage and Branch		CS		<u>Z</u> (I)(B)	29
Clear Word Mark (One Address)		CW		<u>□</u> (A)	30
Clear Word Mark (Two Addresses)		CW		<u>□</u> (A)(B)	30
Compare		C		<u>C</u> (A)(B)	25
Control Carriage—Immediate Skip to Channel 1		CC		<u>F</u> 1	42
Control Carriage—Immediate Skip to Channel 2		CC		<u>F</u> 2	42
Control Carriage—Immediate Skip to Channel 3		CC		<u>F</u> 3	42
Control Carriage—Immediate Skip to Channel 4		CC		<u>F</u> 4	42
Control Carriage—Immediate Skip to Channel 5		CC		<u>F</u> 5	42
Control Carriage—Immediate Skip to Channel 6		CC		<u>F</u> 6	42
Control Carriage—Immediate Skip to Channel 7		CC		<u>F</u> 7	42
Control Carriage—Immediate Skip to Channel 8		CC		<u>F</u> 8	42
Control Carriage—Immediate Skip to Channel 9		CC		<u>F</u> 9	42
Control Carriage—Immediate Skip to Channel 10		CC		<u>F</u> 0	42
Control Carriage—Immediate Skip to Channel 11		CC		<u>F</u> #	42

Instruction	SPS	Mnemonic	Autocoder	Form	Page
Control Carriage—Immediate Skip to Channel 12		CC		F@	42
Control Carriage—Skip After Print to Channel 1		CC		FA	42
Control Carriage—Skip After Print to Channel 2		CC		FB	42
Control Carriage—Skip After Print to Channel 3		CC		FC	42
Control Carriage—Skip After Print to Channel 4		CC		FD	42
Control Carriage—Skip After Print to Channel 5		CC		FE	42
Control Carriage—Skip After Print to Channel 6		CC		FF	42
Control Carriage—Skip After Print to Channel 7		CC		FG	42
Control Carriage—Skip After Print to Channel 8		CC		FH	42
Control Carriage—Skip After Print to Channel 9		CC		FI	42
Control Carriage—Skip After Print to Channel 10		CC		F?	42
Control Carriage—Skip After Print to Channel 11		CC		F•	42
Control Carriage—Skip After Print to Channel 12		CC		F□	42
Control Carriage—Immediate Space, 1 Space		CC		FJ	42
Control Carriage—Immediate Space, 2 Spaces		CC		FK	42
Control Carriage—Immediate Space, 3 Spaces		CC		FL	42
Control Carriage—After-Print, 1 Space		CC		F/	42
Control Carriage—After-Print, 2 Spaces		CC		FS	42
Control Carriage—After-Print, 3 Spaces		CC		FT	42
Control Carriage and Branch		CC		Fd	43
Halt		H		•	31
Halt and Branch		H		•(I)	31
Load Characters to A Word Mark (One Field)	LCA		MLCWA	L(A)	28
Load Characters to A Word Mark (Two Fields)	LCA		MLCWA	L(A)(B)	28
Modify Address (One Address)		MA		#(A)	53
Modify Address (Two Addresses)		MA		#(A)(B)	52
Move Characters and Edit		MCE		E(A)(B)	32
Move Characters and Suppress Zeros		MCS		Z(A)(B)	27
Move Characters to A or B Word Mark (One Field)	MCW		MLC	M(A)	26
Move Characters to A or B Word Mark (Two Fields)	MCW		MLC	M(A)(B)	26
Move Numeric	MN		MLNS	D(A)(B)	27
Move Zone	MZ		MLZS	Y(A)(B)	27
No Operation		NOP		N	31
Punch Card		P		4	36
Punch Card and Branch		P		4(I)	37
Read and Punch		RP		5	38
Read Card		R		1	35
Read Card and Branch		R		1(I)	36
Read from 1447 Console		MU		M(%T0)(B)R	50
Read from 1447 Console with Word Marks		LU		L(%T0)(B)R	50
Read, Punch, and Branch		RP		5(I)	38
Select Stacker—Pocket 1		SS		K1	37
Select Stacker—Pocket 8/2 from Read Feed		SS		K2	37
Select Stacker—Pocket 4		SS		K4	37
Select Stacker—Pocket 8/2 from Punch Feed		SS		K8	37
Select Stacker and Branch	SS		SSB	K(I)d	38
Set Word Mark (One Address)		SW		•(A)	30
Set Word Mark (Two Addresses)		SW		•(A)(B)	29
Subtract (One Field)		S		S(A)	21
Subtract (Two Fields)		S		S(A)(B)	21
Write and Punch		WP		6	44
Write and Read		WR		3	44
Write Line		W		2	41
Write Line and Branch		W		2(I)	41
Write on 1447 Console		MU		M(%T0)(B)W	51
Write on 1447 Console with Word Marks		LU		L(%T0)(B)W	51
Write, Read, and Branch		WR		3(I)	44
Write, Read, and Punch		WRP		7	45
Write, Read, Punch, and Branch		WRP		7(I)	45

Instruction	SPS	Mnemonic	Autocoder	Form	Page
Write, Punch, and Branch		WP		<u>6</u> (I)	45
Write Word Marks	W		WM	<u>2</u> □	42
Write Word Marks and Branch	W		WM	<u>2</u> (I)□	42
Zero and Add (One Field)		ZA		<u>P</u> (A)	20
Zero and Add (Two Fields)		ZA		<u>P</u> (A)(B)	20
Zero and Subtract (One Field)		ZS		<u>I</u> (A)	22
Zero and Subtract (Two Fields)		ZS		<u>I</u> (A)(B)	22

Index

A-Address	9	I-Address	9
A-Address Register	14	I-Address Register	13
A-Register	14	IBM 1401 Processing Unit	5
Add-to-Storage Logic	6	IBM 1402 Card Read-Punch Instructions	35
Add (One Field)	20	IBM 1402 Card Read-Punch Operations	34
Add (Two Fields)	19	IBM 1403 Printer, Models 1 and 2	40
Address Modification	12	IBM 1403 Printer, Model 3	41
Address Modification—Using the Modify Address Instruction	53	IBM 1403 Printer Instructions	41
Address Modification Without Indexing	12	IBM 1403 Printer Operations	40
Address Registers	13	IBM 1403 Printer Timings	46
Address Registers After Operation	10	IBM 1406 Models	52
Address Validity Checking	8	IBM 1406 Storage Unit Instructions	52
Addressable Position	6	IBM 1406 Storage Unit Operations	52
Addressing	10	IBM 1441 Processing Unit	5
Arithmetic Instructions	19	IBM 1447 Console Instructions	50
Arithmetic Operations	17	IBM 1447 Console Operations	49
Assembled Instruction	10	IBM 1461 Input-Output Control	6
		IBM 1462 Printer Control Unit	48
B-Address	9	Index Checking	8
B-Address Register	14	Input-Output Storage Assignments	11
B-Register	14	Instruction Addressing	11
BCD (Binary-Coded-Decimal)	6	Instruction Descriptions	9
Binary-Coded-Decimal (BCD)	6	Instruction Format	9
Body	33	Internal Checking	7
Branch (Unconditional)	23	Internal Processing Speed, 1401 System	5
Branch if Character Equal	24	Internal Processing Speed, 1460 System	5
Branch if Indicator On	23	Language	6
Branch if Word Mark and/or Zone	25	Load Character to A Word Mark (One Field)	28
		Load Characters to A Word Mark (Two Fields)	28
Card-Oriented 1401, 1460 System	5	Logic	7
Card Punch Timing	39	Logic Instructions	23
Card Punching Time	37	Logic Operations	22
Card Read-Punch Timing	39		
Card Read Time	36	Magnetic-Core Storage	6
Card Reader Timing	39	Miscellaneous Instructions	29
Chaining Instructions	14	Miscellaneous Operations	29
Character Registers	14	Mnemonic	9
Checking Address Against Core-Storage Capacity	8	Mode Switch Operation, 1460 Multiple Printer Output	49
Clear Storage	29	Modify Address (One Address)	53
Clear Storage and Branch	29	Modify Address (Two Addresses)	52
Clear Word Mark (One Address)	30	Modulus 4 Arithmetic	12
Clear Word Mark (Two Address)	30	Modulus 16 Rules	13
Combination Input-Output Instructions	44	Move Characters and Edit	32
Combination Input-Output Operations Timing	47	Move Characters and Suppress Zeros	27
Combination Read and Punch Operation Timing	39	Move Characters to A or B Word Mark (One Field)	26
Compare	25	Move Characters to A or B Word Mark (Two Fields)	26
Complement Add	18	Move Numeric	27
Conditional Branch	22	Move Zone	27
Console I/O Printer Timing	51	Multiple Printer Output for 1460	48
Console Instruction Format	50		
Control Carriage	42	No Operation	31
Control Carriage and Branch	43	Notes	10
d-Character	9	Op Code	9
Data Field Addressing	11	Op-Register	14
Data-Moving Instructions	26	Operator Controls, 1460 Multiple Printer Output	49
Data-Moving Operations	26		
Disk-Oriented 1401, 1460 System	5	Parity Checking	7
		Printer Selection, 1460 Multiple Printer Output	49
Edit Operation	32	Processing	7
Editing	7	Processing Units	5
Effective Address Checking	8	Programming Considerations, 1460 Multiple Printer Output	49
End-Around Check	8	Punch Card	36
Example	10	Punch Card and Branch	37
		Punch Select	37
Function	9	Punch Start Time	37
Halt	31	Read and Punch	38
Halt and Branch	31	Read Card	35
Hole-Count Check	8	Read Card and Branch	36
		Read from 1447 Console	49

Read, Punch, and Branch	38	Unconditional Branch	23
Read Select	37	Validity Checking	7
Read Start Time	36	Variable Word Length	8
Select Stacker	37	Word Marks	8
Select Stacker and Branch	38	Words	8
Set Word Mark (One Address)	30	Write and Punch	44
Set Word Mark (Two Addresses)	29	Write and Read	44
Status	34	Write Line	41
Storage-Address Register	14	Write Line and Branch	41
Stored Program Instructions	9	Write on 1447 Console	51
Subtract (One Field)	21	Write, Read, and Branch	44
Subtract (Two Fields)	21	Write, Read, and Punch	45
System Operations	17	Write, Read, Punch, and Branch	45
System Register Operation	13	Write, Punch, and Branch	45
Tape-Oriented 1401, 1460 System	5	Write Word Marks	42
The Stored Program	5	Write Word Marks and Branch	42
Timing	9	Zero and Add (One Field)	20
Timing—1401, Models 1 and 2	46	Zero and Add (Two Fields)	20
Timing—1403, Model 3	46	Zero and Subtract (One Field)	22
Timing—Write and Read Operation	47	Zero and Subtract (Two Fields)	22
Timing—Write and Punch Operation	47		
Timing—Write, Read, and Punch Operation	47		
Title	9		
True-Add	18		



