File Number 1401/1460-01
Form A24-3067-2

IBM Systems Reference Library

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

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

In addition, this manual presents the instructions and applicable timings for the IBM 1402, 1403, 1406, and 1447. For general information on the units attached to the 1401 and 1460 systems, refer to the *IBM 1401 and 1460 Bibliography,* Form A24-1495.

This manual is the first of five reference manuals providing the complete instruction set for the IBM 1401 and 1460. The other four manuals are:

Miscellaneous Input/Ouput Instructions (A24-3068)
Tape Input/Output Instructions (A24-3069)
Disk Input/Output Instructions (A24-3070)
Special Feature Instructions (A24-3071)

To accommodate a particular system configuration, any combination of these five manuals can be placed in a single binder for the user's convenience.

Preface

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

The complete manual is divided functionally into these sections:

System Operation Reference Manual (A24-3067)

- Section A Introduction
- Section B System Operations
- Section C IBM 1406 Operations
- Section D IBM 1447 Operations
- Section E IBM 1402 and 1403 Operations
- Section N Code Chart and Index of Instructions

Tape Input/Output Instructions (A24-3069)

- Section F Tape Input/Output Operations

Disk Input/Output Instructions (A24-3070)

- Section G Disk Input/Output Operations

Miscellaneous Input/Output Instructions (A24-3068)

- Section H Miscellaneous Input/Output Operations

Special Feature Instructions (A24-3071)

- Section I Special Feature Operations

A System Reference Library can be compiled using those sections applicable to the user's machine configuration.

This publication is intended for programmers and systems personnel who have a general knowledge of the IBM 1401 or 1460 Data Processing Systems and who require a reference text for detailed information.

Other publications referenced here are, in most cases, prerequisites for a complete understanding of the material presented in this publication.

Third Edition (September 1966)

This edition, A24-3067-2, is a major revision of and obsoletes A24-3067-1.

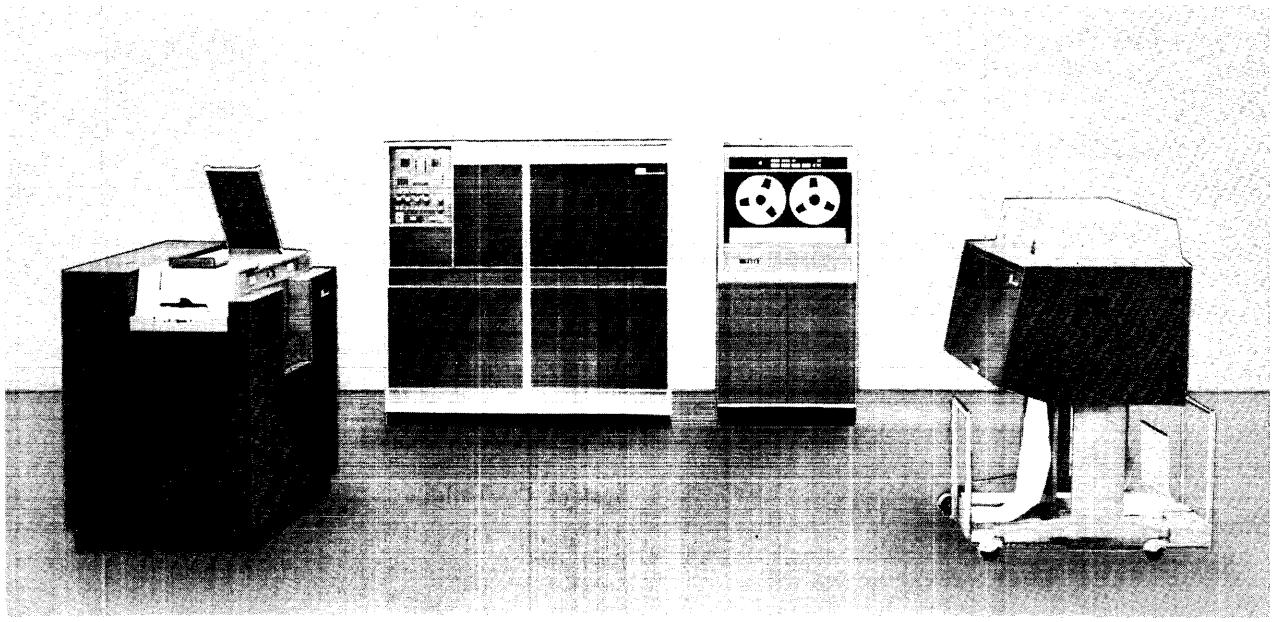
This revision does not obsolete the four companion reference manuals listed in the *Preface.*

Significant changes are indicated as follows:

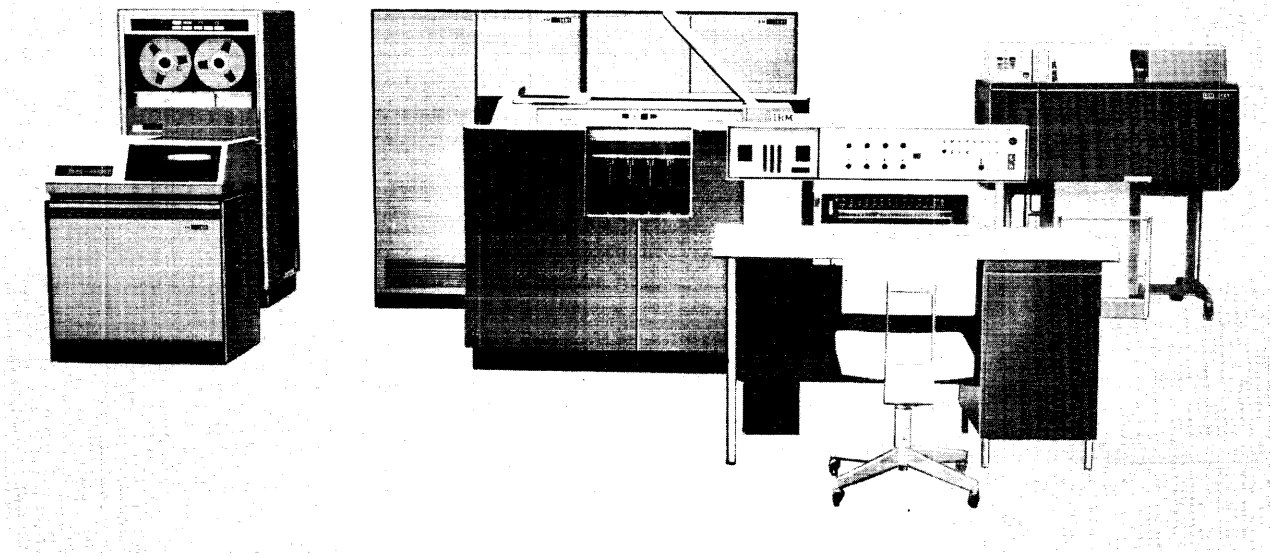
1. A dot (•) appears beside the page number when the entire page should be reviewed.
2. A vertical line (|) appears beside changed or added text.
3. A dot appears to the left of a figure title when the entire figure should be reviewed.

Copies of this and other IBM publications can be obtained through IBM Branch Offices.
This Manual has been prepared by the IBM Systems Development Division,
Product Publications, Dept. 171, P.O. Box 6, Endicott, N. Y. 13760.
A form has been provided at the back of this publication for readers' comments.

Contents



1401



1460

Figure A-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 A-2) serves this function for the 1401 system, and the IBM 1441 Processing Unit (Figure A-3) serves this function for the 1460 system. Each processing unit can be divided into two parts:

- The arithmetic-logical unit
- The control section

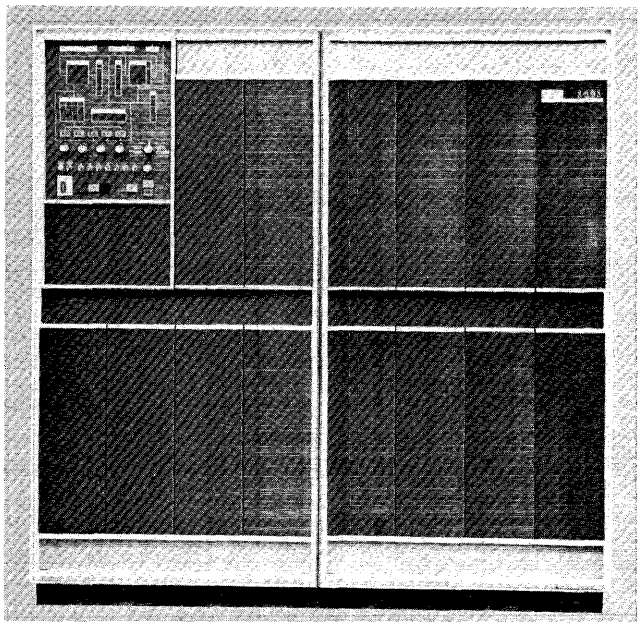


Figure A-2. IBM 1401 Processing Unit

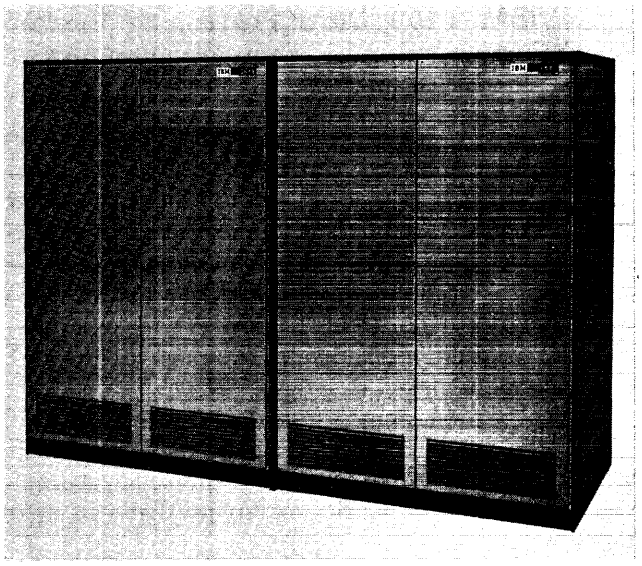


Figure A-3. IBM 1441 Processing Unit and IBM 1461 Input/Output Control

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 A-3).

The capacity of the IBM 1401 Processing Unit is 1,400; 2,000; 4,000; 8,000; 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 A-4). All data in core storage is instantly

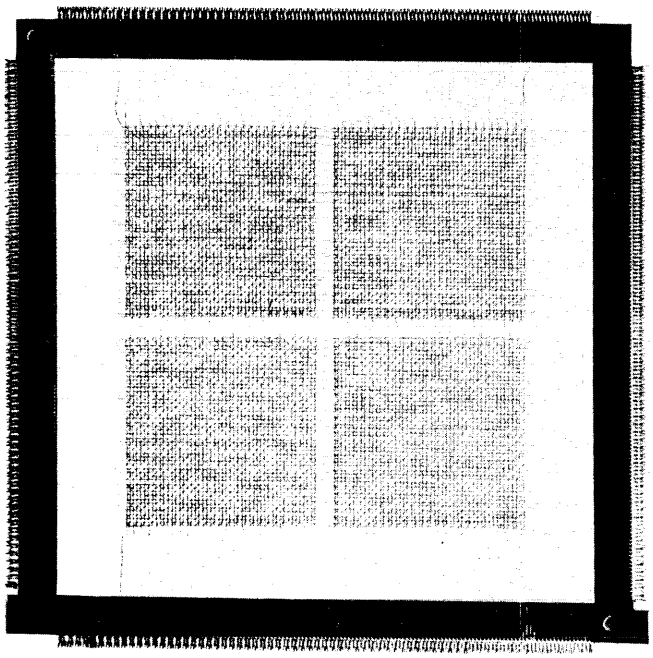


Figure A-4. Magnetic-Core Storage

available, and the special design of the core-storage unit makes each position individually *addressable.* This 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 A-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

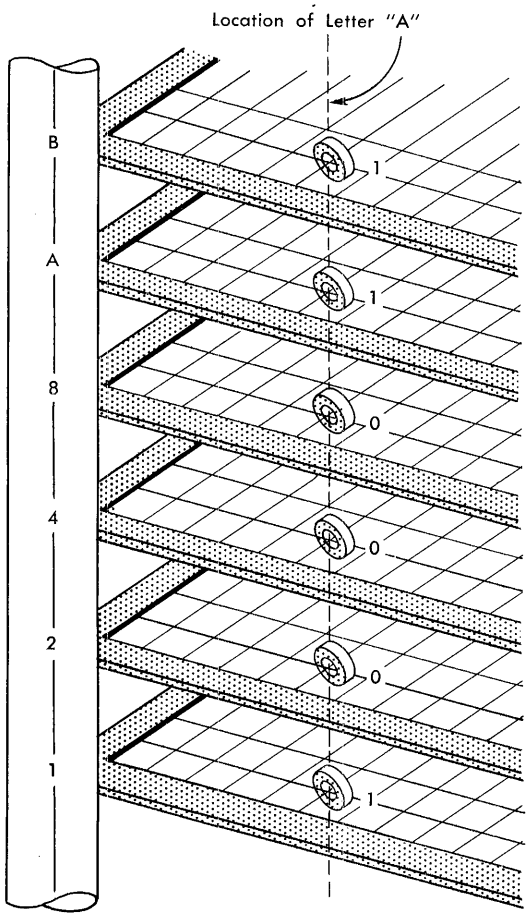


Figure A-5. The Letter A Represented in Binary-Coded Form in Core Storage

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-card output. Converting input data to the 1401 or 1460 internal code, and subsequently reconverting, 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 subroutines. 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 1,400-position 1401 system is checked for addresses between 1400 and 4000. A 2,000-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 card 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

Op Code	*A- or I-address*	*B-address*	*d-character*
X	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.

Instruction Descriptions

Specific instructions have been described in a standard format:

Title. This is the description of the instruction.

Instruction Length. The length of an instruction can be either 1, 2, 4, 5, 7, or 8 characters. It cannot be either 3 or 6 characters long. Characters beyond the usable limit of eight do not affect the operation. Addressing advances (mod + 1) until the next word mark is sensed before the instruction is executed.

Most instructions must have a word mark following the instruction in core storage. This word mark is normally associated with the core-storage location immediately following the instruction itself.

Figure A-6 shows examples of the combinations possible in variable-length instructions.

Instruction Format. This is the format of the particular instruction described. The mnemonics operation code used for Autocoder and SPS are given.

Function. This is the function of the instruction.

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

Timing. When the instruction-execution timing is always a constant, the actual time in milliseconds is given. When the instruction-execution time can vary because of field length or chaining, the formula is given. Figure A-7 is the key to the abbreviations used in the formulas.

Notes. These are special notations or additional information pertaining to the operation.

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

Chaining. This assists the programmer in determining whether instruction-chaining can be used effectively. In some cases, chaining proves useful even though it would not ordinarily be used. For example, another instruction can be chained to the MOVE CHARACTERS AND EDIT instruction if the programmer can use the contents of the address registers to advantage. When considering the use of chaining, be certain that the contents of the address registers are valid for all conditions relating to the instructions involved. (Refer to the specific instruction section.)

Example. A practical application of the instruction is described with the label of a typical actual machine address (in parentheses).

These examples for the instructions are representative, and are intended as exhibits of typical core-storage assignments, rather than specific, limited examples. Since the assembler program (autocoder or SPS) is usually allowed to establish core-storage addresses, the programmer need not ordinarily be concerned with specific machine-language storage locations, except when a program must be analyzed.

The few inflexible addresses of core-storage locations, such as index registers, are shown in the instructions as exact locations.

Assembled Instruction. This is the actual machine language instruction that is assembled by the processor program from the symbolic entries shown in the example.

When an explicit mnemonic is used, the op code, A-address, and d-modifier (when required) are automatically generated in most cases (refer to the specific operation in question).

Example: The coded instruction to cause information to print on the 1447 might be in the form: WCP (column 16 of the coding sheet), PRTOUT (written in column 21-26). Assume for the purpose of this example that the label PRTOUT actually represents core-storage location 0101. Autocoder would assemble this coded instruction into actual machine

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

Figure A-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 A-7. Timing Formula Coding

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_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
dbb	The d-character and blank in the units and tens position. The actual d-character is shown when possible.

Figure A-8. Address Registers after Operation Coding

language as follows: M %T0 101 W. In this instance, the machine-language op code (M), the A-address designating the 1447 (%T0), and the d-modifier character defining the operation as a write (W) were all explicitly defined in the Autocoder mnemonic: WCP.

Example: When an explicit Autocoder mnemonic is *not* provided, or if the programmer uses a "general" mnemonic, the A-address and the d-modifier *must* be written as part of the operand: MU (column 16-17), unit address %T0 (starting in column 21), B-address (starting next), and the specific d-character W (last). In this case, the results are the same: M %T0 101 W.

Example: The instruction can be coded entirely in machine language, if desired. In this case, the actual op code is written in column 19, and the d-modifier character is written in column 20.

Example: In other cases, the actual machine language op code is implied by the mnemonic, but the d-modifier character must still be coded in the operand, such as for the BRANCH IF CHARACTER EQUAL (BCE) instruction. Here, the machine language op code (B) is explicit, but the flexibility of the d-character requires that the programmer code the d-modifier.

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: 1,400; 2,000; 4,000; 8,000; 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 1000-3999 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 8000-11999 have zone bits over the hundreds position, and B-bits (11 zone) over the units position of the 3-character address.

Storage addresses 12000-15999 have zone bits over the hundreds position, A- and B-bits (12 zone) over the units position of the 3-character address. Figure A-9 is a chart of the addressing system.

ACTUAL ADDRESSES	ZONE BITS OVER HUNDREDS POSITION	ZONE BITS OVER UNITS POSITION	3-CHARACTER ADDRESSES
0000 to 0999	No Zone Bits	No Zone Bits	000 to 999
1000 to 1999	A-Bit (Zero-Zone)	No Zone Bits	≠00 to Z99
2000 to 2999	B-Bit (11-Zone)	No Zone Bits	100 to R99
3000 to 3999	AB-Bits (12-Zone)	No Zone Bits	?00 to 199
4000 to 4999	No Zone Bits	A-Bit (Zero-Zone)	00≠ to 99Z
5000 to 5999	A-Bit (Zero-Zone)	A-Bit (Zero-Zone)	≠0≠ to Z9Z
6000 to 6999	B-Bit (11-Zone)	A-Bit (Zero-Zone)	10≠ to R9Z
7000 to 7999	AB-Bits (12-Zone)	A-Bit (Zero-Zone)	?0≠ to 19Z
8000 to 8999	No Zone Bits	B-Bit (11-Zone)	001 to 99R
9000 to 9999	A-Bit (Zero-Zone)	B-Bit (11-Zone)	≠01 to Z9R
10000 to 10999	B-Bit (11-Zone)	B-Bit (11-Zone)	101 to R9R
11000 to 11999	AB-Bits (12-Zone)	B-Bit (11-Zone)	?01 to 19R
12000 to 12999	No Zone Bits	AB-Bits (12-Zone)	00? to 991
13000 to 13999	A-Bit (Zero-Zone)	AB-Bits (12-Zone)	≠0? to Z91
14000 to 14999	B-Bit (11-Zone)	AB-Bits (12-Zone)	10? to R91
15000 to 15999	AB-Bits (12-Zone)	AB-Bits (12-Zone)	?0? to 191

Figure A-9. 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.

?99	=	A B 099	=	1 2 099	=	3099
I5R	=	A B B B 9 5 9	=	1 8 2 8 9 5 9	=	11,959

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 A-10) contains the operation code for the following instruction:

Op Code	*A-address*	*B-address*
A	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 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

Instruction addressed by high-order position

STORAGE ADDRESS	400	401	402	403	404	405	406	407 (NSI)
INSTRUCTION	A	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.

A-address

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

A-field

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

B-address

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

B-field

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

Figure A-10. Data and Instruction Addressing

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 A-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 whether 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 feature, which is a special feature for the 1401 and 1460 systems.

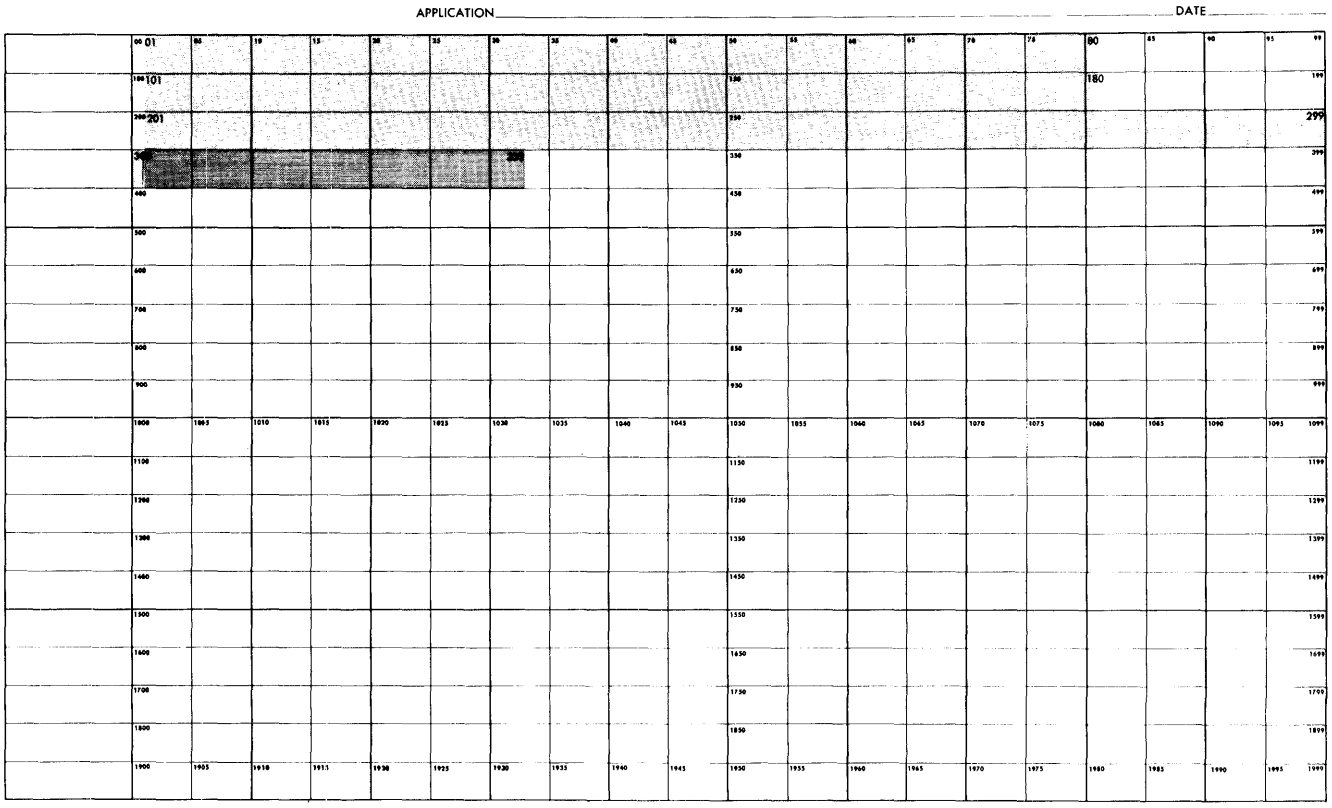


Figure A-11. Core Storage Layout Chart

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 A-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.

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 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)

A + A	= B	or	1 + 1 = 2
A + B	= AB	or	1 + 2 = 3
B + B	= NoANoB	or	2 + 2 = 0
A + AB	= NoANoB	or	1 + 3 = 0
A + NoANoB	= A	or	1 + 0 = 1
B + AB	= A	or	2 + 3 = 1
B + NoANoB	= B	or	2 + 0 = 2
AB + AB	= B	or	3 + 3 = 2

Figure A-12. A-Bit and B-Bit Values

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 (H5B) = complement
879 + H5B = 731 (with arithmetic overflow)
H = BA8
B = 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 15000 to 15999 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.

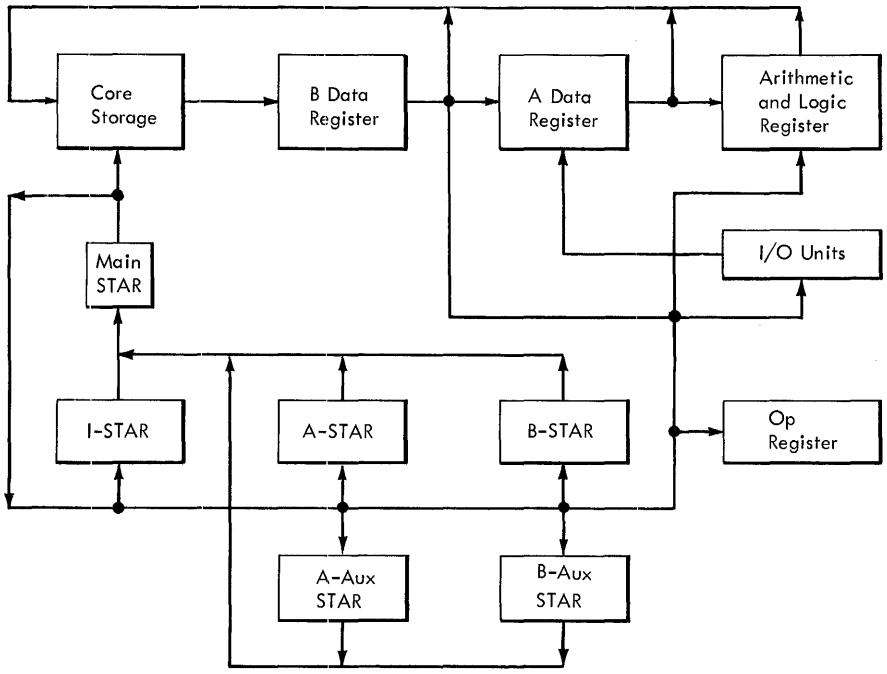
System Register Operation

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 A-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



Notes: The STAR's (Storage Address Registers) address the instruction or data location of the operation.

The Aux (Auxiliary) STAR's are used only with the Multiply-Divide Special Feature.

Figure A-13. System Data Flow Schematic

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.

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.

IBM 1401G A- and B-Address Registers

The IBM 1401G A- and B-address registers differ in use from the 1401, because in the 1401G, the reader-punch is controlled through the use of A-cycles rather than B-cycles. As a result, the final A-storage-address register and B-storage-address register differ from the 1401 as follows:

Operation	*1401 A-Storage Address Register*	*1401-G A-Storage Address Register*	*1401 B-Storage Address Register*	*1401-G B-Storage Address Register*
5 op	Ap	181 or 081	181 or 081	Bp
6 op	Ap	181	181	335
7 op	Ap	181 or 081	181 or 081	335
1 op	Ap	081	081	Bp
2 op	Ap	Ap	335	335
3 op	Ap	081	081	335
4 op	Ap	181	181	Bp
1XXX	BI	081	081	A
4XXX	BI	181	181	A
3XXX	BI	081	081	335
5XXX	BI	181 or 081	181 or 081	A
6XXX	BI	181	181	335
7XXX	BI	181 or 081	181 or 081	335
6R	dbb	081	081	335
4R	dbb	081	081	dbb
6XXXR	BI	081	081	335
4XXXR	BI	081	081	dbb

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 or from the Arith register.

Figure A-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:

```
A  700  850
A  695  845
A  690  840
A  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:

```
A  700  850
A
A
A
```

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

CYCLE	OPERATION	Instruction Location: A 197 | 5 198 | 6 199 | 7 200 | T 201 | 1 202 | 2 203 | S 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-Address Register: 0|1|9|7 · B-Register: A · A-Register: ? · Op Register: A · A-Address 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 hundreds positions of the A- and B-address registers and the A-register by the way of the B-register.	I-Address Register: 0|1|9|8 · B-Register: 5 · A-Register: 5 · Op Register: A · A-Address Register: 0|5|b|b · 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-Address Register: 0|1|9|9 · B-Register: 6 · A-Register: 6 · Op Register: A · A-Address Register: 0|5|6|b · 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-Address Register: 0|2|0|0 · B-Register: 7 · A-Register: 7 · Op Register: A · A-Address Register: 0|5|6|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. (The letter "T" is in the B-address register but "13" is displayed on the console.)	I-Address Register: 0|2|0|1 · B-Register: T · A-Register: T · Op Register: A · A-Address Register: 0|5|6|7 · 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-Address Register: 0|2|0|2 · B-Register: 1 · A-Register: 1 · Op Register: A · A-Address Register: 0|5|6|7 · 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-Address Register: 0|2|0|3 · B-Register: 2 · A-Register: 2 · Op Register: A · A-Address Register: 0|5|6|7 · 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-Address Register: 0|2|0|4 · B-Register: S · A-Register: 2 · Op Register: A · A-Address Register: 0|5|6|7 · B-Address Register: 1|3|1|2	Cycle 8

Figure A-14. Schematic of Instruction Loading

instructions may have various Op codes. To be operated 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 A-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 (if applicable).

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 operations 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 subroutines, 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 Operations

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.

Figure B-1 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

The type of add cycle performed depends on the arithmetic operation and the signs and values of the two factors involved (Figure B-2). All arithmetic operations are performed with algebraic sign control. The sign of a result depends on the operation, the magnitudes of the terms to be combined, and the sign-bits of these terms.

Because all arithmetic operations are performed with algebraic sign control, the *sign* of the result depends both on the operation, and on the magnitude and signs of the factors involved (Figure B-3).

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 B-1. Sign Bit Equivalents

TYPE OF OPER.	A-FLD. SIGN	B-FLD. SIGN	TYPE OF ADD CYCLE	SIGN OF RESULT
ADD	+	+	True Add	+
		−	Compl. Add	Sign of Field with Larger Magnitude
	−	+	Compl. Add	
		−	True Add	−
SUBTRACT	+	−	True Add	−
		+	Compl. Add	Sign of Field with Larger Magnitude
	−	−	Compl. Add	
		+	True Add	+

Figure B-2. Types of Add Cycles and Sign of Result for Add and Subtract Operations

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 B-4).

	Sign-Bits of A-Field						A	A	A	A	B	B	B	B	AB	AB	AB	AB
	Sign-Bits of B-Field			A	B	AB		A	B	AB		A	B	AB		A	B	AB
ADD	Resultant Sign When:	A > B		A	AB	AB		A	AB	AB	B	B	B	B		A	AB	AB
	Resultant Sign When:	A ≤ B		A	B	AB		A	B	AB	AB	AB	B	AB		A	B	AB
SUB	Resultant Sign When:	A > B	B	B	B	B	B	B	B	B		A	AB	AB	B	B	B	B
	Resultant Sign When:	A ≤ B	AB	AB	B	AB	AB	AB	B	AB		A	B	AB	AB	AB	B	AB

Figure B-3. Zone-Bit Table for Add and Subtract Operations

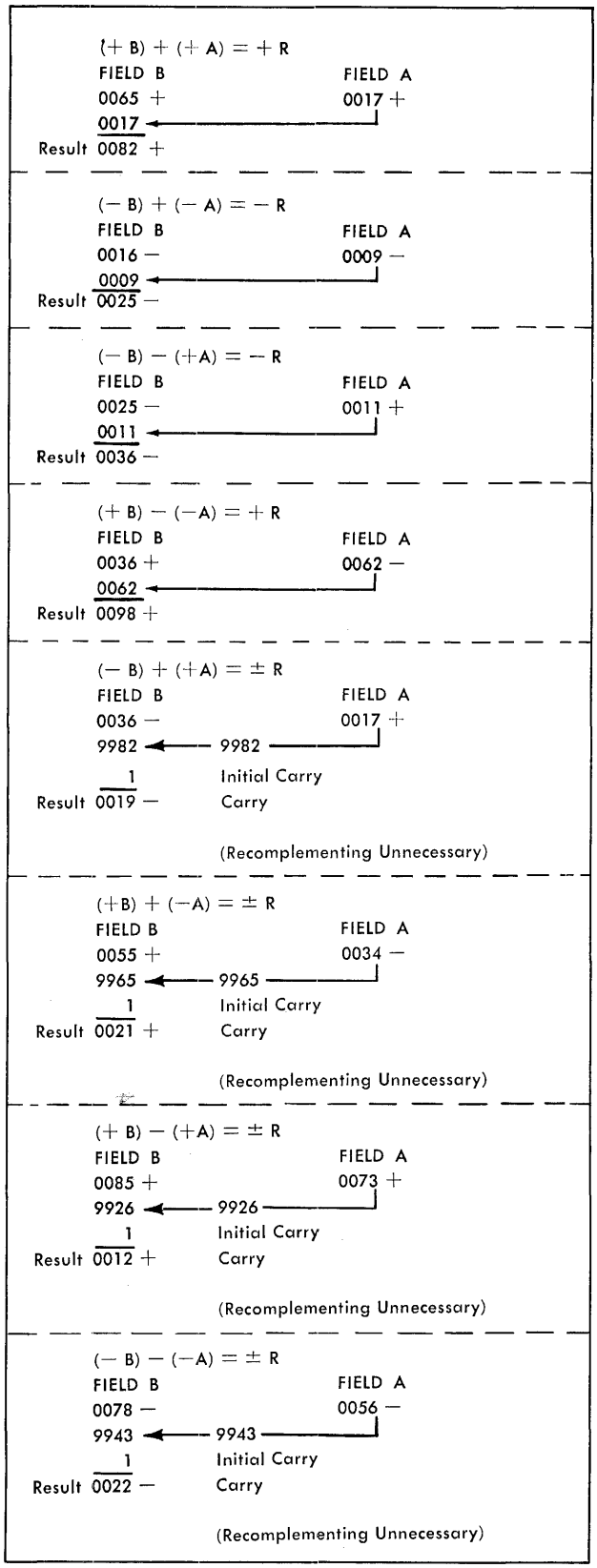


Figure B-4. True-Add and Complement-Add Cycle Examples

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 B-4). The original sign of the B-field is the sign of the result.

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 B-5). The system generates the new sign automatically.

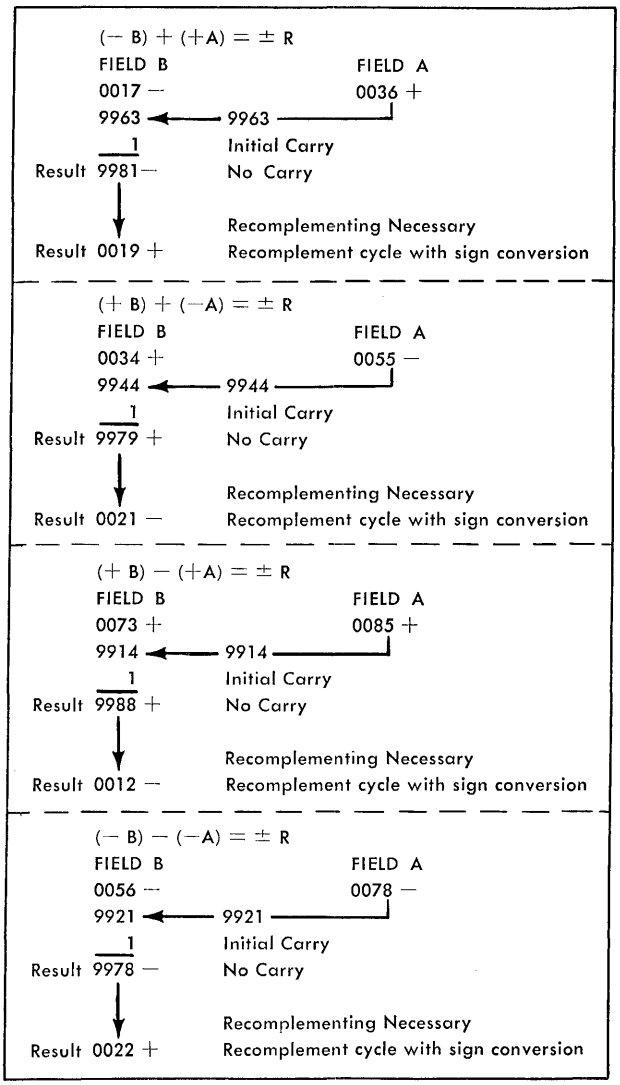


Figure B-5. Complement-Add (with Recomplement) Cycle Examples

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 field with the larger magnitude.

An accumulator field positioned in the last few available core-storage locations, such as units position of the field in 3999, will cause a wrap-around error, if an arithmetic function involving recomplementation is performed.

Arithmetic Instructions

Add (Two Fields)

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
A	A	AAA	BBB

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 recomplement cycle:

1401:

$T = .0115 (L_I + 3 + L_A + L_B)$ ms.

1460:

$T = .006 (L_I + 1 + L_A + L_B)$ ms.

2. If a recomplement cycle is taken:

1401:

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

1460:

$T = .006 (L_I + 1 + L_A + 3\ L_B)$ ms. If the multiply-divide special feature is installed, the 1460 timing for a recomplement cycle is:
$T = .006 (L_I + 1 + L_A + 2\ L_B)$ ms.

Notes:

1. *Sign Control.* 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. (See Figure B-4.)

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.

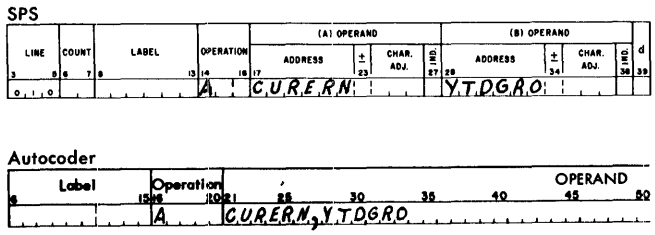
The BRANCH IF ARITHMETIC OVERFLOW INDICATOR ON (B xxx Z) instruction tests and turns off the arithmetic overflow indicator and branches to a special instruction or group of instructions if this condition occurs. There is only one overflow indicator in the system. It is turned off by a BRANCH IF ARITHMETIC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

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Add CURERN (0506) to YTDGRO (0708), Figure B-6.



Assembled Instruction: A 506 708

Figure B-6. Add (Two Fields)

Add (One Field)

Instruction Format.

Mnemonic	*Op Code*	*A-address*
A	A	AAA

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.

Timing.

1401:

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

1460:

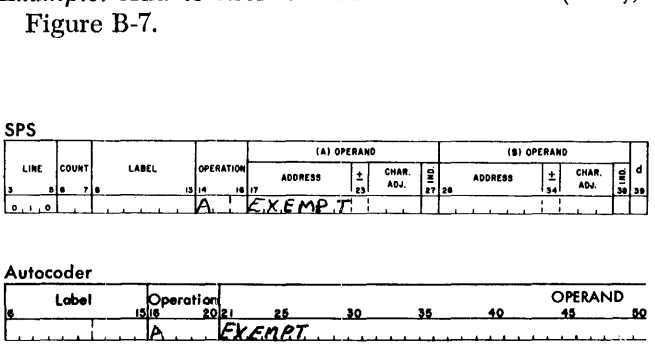
$T = .006\ (L_I + 1 + 2L_A)$ ms.

Address Registers After Operation.

I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
NSI	$A\text{-}L_A$	$A\text{-}L_A$

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Add to itself the data at EXEMPT (0981), Figure B-7.



Assembled Instruction: A 981

Figure B-7. Add (One Field)

Zero and Add (Two Fields)

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
ZA	?	AAA	BBB

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.

The Op code for this instruction is plus zero but prints as an ampersand.

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

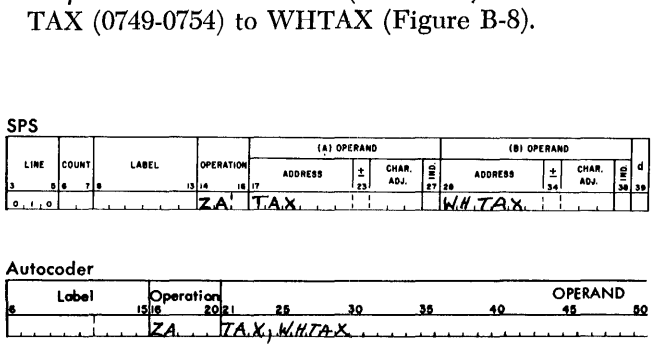
Note: The sign of the result always has both A- and B-bits if it is positive. If the sign is negative, it has only a B-bit.

Address Registers After Operation.

I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
NSI	$A\text{-}L_W$	$B\text{-}L_B$

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Zero WHTAX area (0796-0802) and add new TAX (0749-0754) to WHTAX (Figure B-8).



Assembled Instruction: ? 754 802

Figure B-8. Zero and Add (Two Fields)

Zero and Add (One Field)

Instruction Format.

Mnemonic	*Op Code*	*A-address*
ZA	?	AAA

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.

The Op code for this instruction is plus zero but prints as an ampersand.

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	$A\text{-}L_A$	$A\text{-}L_A$

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Strip zone bits from TOTAL (0560) area (Figure B-9).

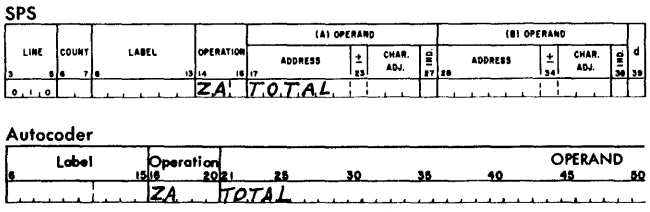


Figure B-9. Zero and Add (One Field)

Subtract (Two Fields)

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
S	S	AAA	BBB

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 B-2 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:

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

   1460:

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

2. Subtract – recomplement cycle necessary

   1401:

   $T = .0115\ L_I + 3 + L_A + 4L_B)$ ms.

   1460:

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

   $T = .006\ (L_I + 1 + L_A + 2L_B)$ 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	$A\text{-}L_W$	$B\text{-}L_B$

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Subtract CUFICA (0753) from CURGRO (0896), Figure B-10.

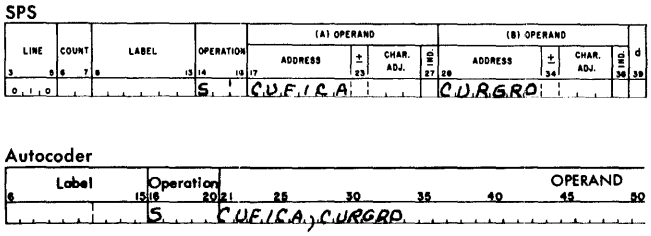


Figure B-10. Subtract (Two Fields)

Subtract (One Field)

Instruction Format.

Mnemonic	*Op Code*	*A-address*
S	S	AAA

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.

Timing.

1401:

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

1460:

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

Address Registers After Operation.

I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
NSI	$A\text{-}L_A$	$A\text{-}L_A$

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Subtract from itself the field labeled LIMIT (units position is 0395), Figure B-11.

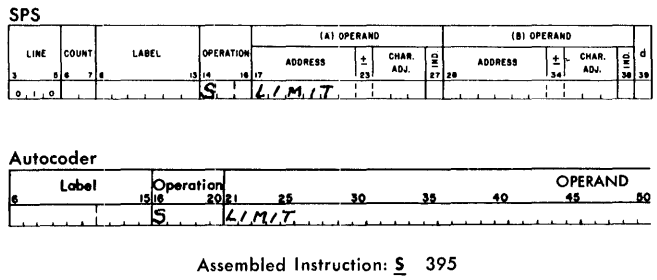


Figure B-11. Subtract (One Field)

Zero and Subtract (Two Fields)

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
ZS	!	AAA	BBB

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.

The Op code for this instruction is minus zero but prints as a dash.

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	$A\text{-}L_W$	$B\text{-}L_B$

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Zero ACCUM 1 (0755) and subtract TAXEXP (0699) from ACCUM 1, Figure B-12.

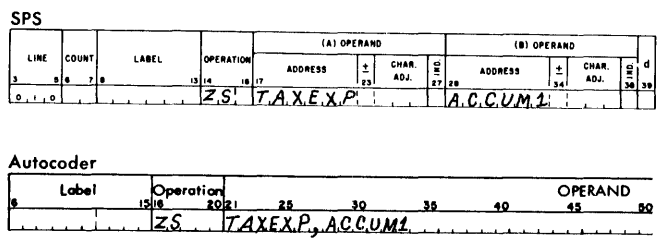


Figure B-12. Zero and Subtract (Two Fields)

Zero and Subtract (One Field)

Instruction Format.

Mnemonic	*Op Code*	*A-address*
ZS	!	AAA

Function. This instruction causes the sign of the A-field to be changed. The Op code for this instruction is minus zero but prints as a dash.

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	$A\text{-}L_A$	$A\text{-}L_A$

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Subtract LIMIT (0495) from zero, and change sign of LIMIT's value (Figure B-13).

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	(A) OPERAND ±	(A) OPERAND CHAR. ADJ.	(A) OPERAND IND	(B) OPERAND ADDRESS	(B) OPERAND ±	(B) OPERAND CHAR. ADJ.	(B) OPERAND IND	d
3 8	6 7	8 13	14 16	17		23		27	28	34		39
			ZS	LIMIT								

Autocoder

Label	Operation	OPERAND
6 15	16 20	21 25 30 35 40 45 50
	ZS	LIMIT

Assembled Instruction: ! 495

Figure B-13. 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 result 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.

Branch (Unconditional)

Instruction Format.

Mnemonic	*Op Code*	*I-address*
B	B	III

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

Timing.

Branch without indexing: T = 5N ms.
Branch with indexing: T = 6N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg*	*B-Add. Reg.*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Example. Unconditionally branch to AGAIN (3498), Figure B-14.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	(B) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	d
3 5	6 7	8 13	14 16	17 23			27	28 34			38	39
0 1 0			B	AGAIN								

Autocoder

Label	Operation	OPERAND
6 15	16 20	21 25 30 35 40 45 50
	B	AGAIN

Assembled Instruction: B D98

Figure B-14. Branch (Unconditional)

Branch if Indicator On

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
SPS B	B	III	d
A see Figure B-15			

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.
‡ d-modifier character must be coded in the operand portion of the instruction.

Figure B-15. Branch if Indicator On, Mnemonic d-Character, and Conditions

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 B-15 shows characters that are valid for the d-character and for 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, or Branch without indexing: T = 6N ms.
Branch with indexing: T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	dbb
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Example. Test for the last card. If it is the last card, branch to END (0599), Figure B-16.

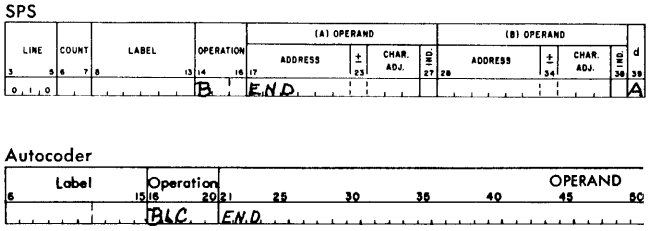


Figure B-16. Branch if Indicator On

Branch if Character Equal

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*B-address*	*d-character*
SPS B	B	III	BBB	d
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, or Branch without indexing:	$T = N (L_I + 2)$ ms.
Branch with indexing:	$T = N (L_I + 3)$ ms.

Note. The BRANCH IF CHARACTER EQUAL instruction does not reset any internal system indicators, as does the BRANCH IF READER ERROR instruction, for example.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	B-1
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

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 B-17).

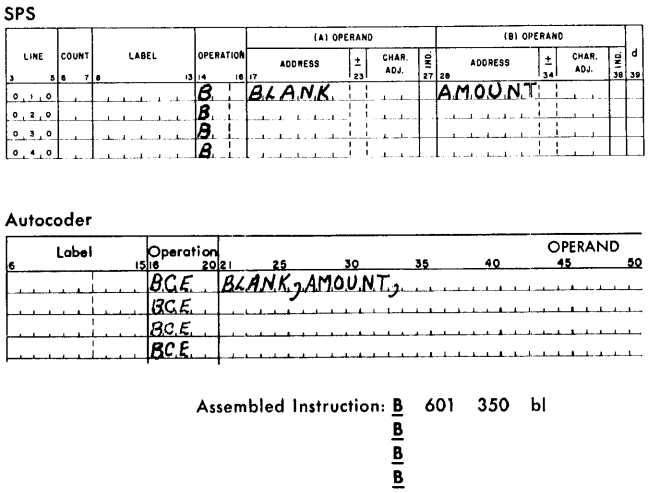


Figure B-17. Branch if Character Equal

Branch if Word Mark and/or Zone

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*B-address*	*d-character*
BWZ	V	III	BBB	d

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

Two explicit autocoder mnemonics are provided that do not require the third (d-modifier character) operand. These are BM for Branch if 11-Zone, and BW for Branch if Word Mark.

Word Marks. These have been explained previously.

Timing.

No Branch or Branch without indexing:	$T = N (L_I + 2)$ ms.
Branch with indexing:	$T = N (L_I + 3)$ ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	B-1
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

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 B-18).

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	(A) OPERAND ±	(A) OPERAND CHAR. ADJ.	(A) OPERAND IND	(B) OPERAND ADDRESS	(B) OPERAND ±	(B) OPERAND CHAR. ADJ.	(B) OPERAND IND	d
3 5	6 7	11 13	14 16	17 23			27	28 34			36	39
0 1 0			BWZ	NEGRTE				GROAMT				K

Autocoder

Label	Operation	OPERAND
6 15	16 20	21 25 30 35 40 45 80
	BM	NEGRTE,GROAMT

Assembled Instruction: V 598 M98 K

● Figure B-18. Branch if 11 Zone

Compare

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
C	C	AAA	BBB

The following explicit autocoder branch instructions are provided for use with the Compare instruction. When the specified mnemonic is used, the d-character is not written in the operand field.

Branch if Unequal (B ≠ A)	BU
Branch if Equal (B = A)	BE
Branch if High (B>A)	BH
Branch if Low (B<A)	BL

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. In this case, the high-compare indicator is set *on*.

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

Note: Both fields must have exactly the same bit configurations, to be equal. For example, 00? (? = 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 · □ [< ≡ & $ *] ; Δ − / , % ˇ \ ≠ b # @ : > √ ? A through I | 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

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code. When Compare instruction(s) are chained to a preceding Compare instruction, the compare-result indicators are set by the *first* unequal condition encountered in the composite field.

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 B-19).

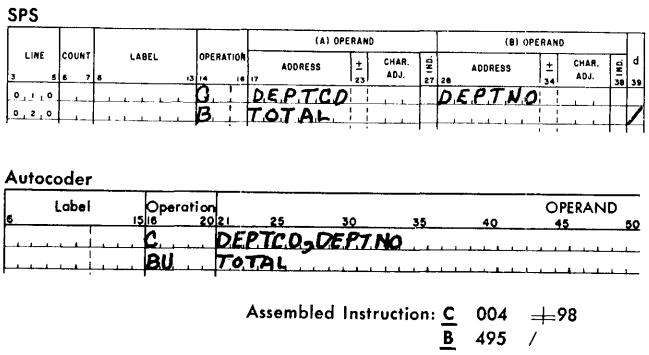


Figure B-19. Compare

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 A MLC or MCW	M	AAA	BBB

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

Chaining. See the *Move Characters to A or B Word Mark (One Field)* section.

Example. Move the 5-character field NAMIN (0750) to the 5-character field NAMOUT (0850), Figure B-20.

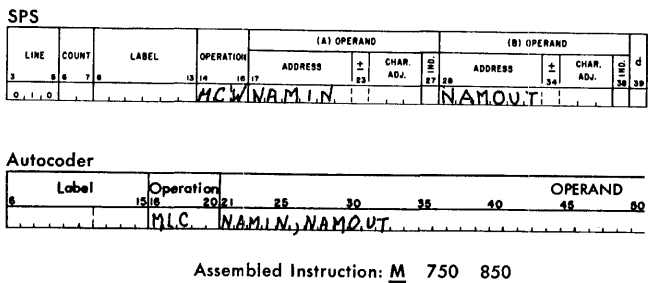


Figure B-20. Move Character to A or B Word Mark (Two Fields)

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

Instruction Format.

Mnemonic	*Op Code*	*A-address*
SPS MCW	M	AAA
A MLC or MCW		

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

Chaining. This instruction does not reset the B-storage address register and is, therefore, chained to the preceding operation, in that the previous B-STAR is used (see *Example*). If both the A- and B-STAR's contain usable addresses, only the operation code need be used.

Example. Move the following three fields (labeled EMPNO, DEPTNO and TAXCLS) and store them sequentially at RECOUT (units position at 209), Figure B-21.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	WS	(B) OPERAND ADDRESS	±	CHAR. ADJ.	WS	d
3 5	6 7	8 13	14 16	17	23		27	28	34		36	39
0 1 0			MCW	TAXCLS				RECOUT				
0 2 0			MCW	DEPTNO								
0 3 0			MCW	EMPYNO								
0 4 0												

Autocoder

Label	Operation	OPERAND
6 15	16 20	21 25 30 35 40 45 50
	MLC	TAXCLS,RECOUT
	MLC	DEPTNO
	MLC	EMPYNO

Assembled Instruction: M 115 209
M 110
M 104

Figure B-21. Move Characters to A or B Word Mark (One Field)

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

Move Characters and Suppress Zeros

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
MCS	Z	AAA	BBB

Function. The A-field data is moved to corresponding positions of the B-field as follows:

1. Numeric characters are transferred unconditionally.
2. Alphabetic and special characters are transferred unconditionally except that zone bits are removed from the units position.
3. Any zeros or commas *to the left* of the first significant numeric character are replaced by blanks in the B-field.
4. Any character other than comma, hyphen, blank, significant digit, or zero causes zero suppression to begin again.
5. The A-field sign is not transferred.
6. B-field word marks are cleared.
7. The A-field data is not affected.

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

Chaining. This instruction is not normally chained.

Example. Move and suppress the zeros in the 10-character field labeled INVBAL (0958) to the area labeled OUTPT4 (0448), Figure B-22.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	(B) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	d
3 5	6 7	8 13	14 16	17	23		27	28	34		38	39
0 1 0			MCS	INVBAL				OUTPT4				

Autocoder

Label	Operation	OPERAND
6 15	16 20 21	25 30 35 40 45 50
	MCS	INVBAL,OUTPT4

Assembled Instruction: Z 958 448

Figure B-22. Move Characters and Suppress Zeros

Move Numeric

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
SPS MN	D	AAA	BBB
A MLNS or MN			

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

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Move the numeric portion of the units position of ONHAND (0986) to OUT5 (0789), Figure B-23.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	(B) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	d
3 5	6 7	8 13	14 16	17	23		27	28	34		38	39
0 1 0			MN	ONHAND				OUT5				

Autocoder

Label	Operation	OPERAND
6 15	16 20 21	25 30 35 40 45 50
	MLNS	ONHAND,OUT5

Assembled Instruction: D 986 789

Figure B-23. Move Numeric

Move Zone

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
SPS MZ	Y	AAA	BBB
A MLZS or MZ			

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

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Move the zone bits from the units position of NEWBAL (3100) to the area labeled REC2 (3195), Figure B-24.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	(B) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	d
3 5	6 7	8 13	14 16	17	23		27	28	34		38	39
0 1 0			MZ	NEWBAL				REC2				

Autocoder

Label	Operation	OPERAND
6 15	16 20 21	25 30 35 40 45 50
	MLZS	NEWBAL,REC2

Assembled Instruction: Y A00 A95

Figure B-24. Move Zone

Load Characters to A Word Mark (Two Fields)

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
SPS LCA	L	AAA	BBB
A MLCWA or LCA			

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-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
NSI	$A\text{-}L_A$	$B\text{-}L_A$

Chaining. See the *Load Characters to A-Field Word* Mark (*One Field*) section.

Example. Transfer the data and word marks from REC4 (0950) to OUT8 (0650), Figure B-25.

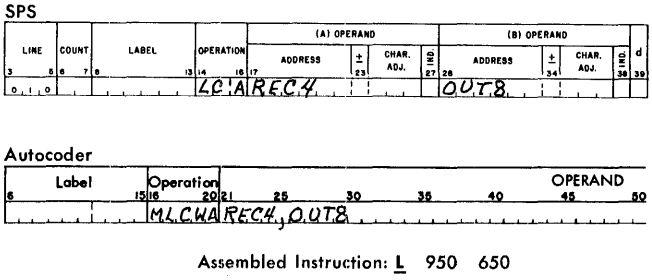


Figure B-25. Load Characters to A Word Mark (Two Fields)

Load Characters to A Word Mark (One Field)

Instruction Format.

Mnemonic	*Op Code*	*A-address*
SPS LCA	L	AAA
A MLCWA or LCA		

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-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
NSI	$A\text{-}L_A$	$Bp\text{-}L_A$

Chaining. This instruction does not reset the B-STAR and is, therefore, chained to the preceding operation, in that current contents of the B-STAR is used (see *Example*). If both the A- and B-STAR's contain usable addresses, only the operation code need be supplied.

Example. Load the following three fields: EMPYNO, DEPTNO, and TAXCLS with their word marks to sequential locations, beginning at storage location 0209, Figure B-26.

	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	PRINT1	0208-0209

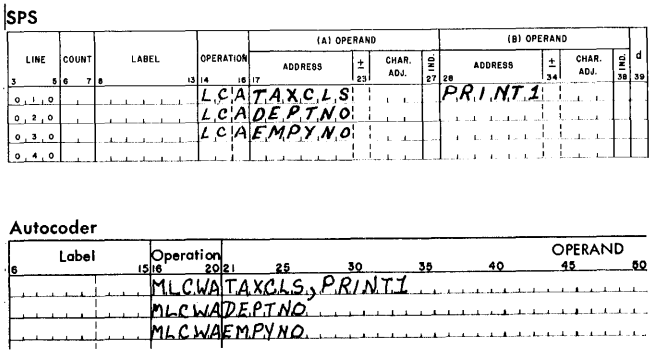


Figure B-26. Load Characters to A Word Mark (One Field)

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	/	AAA

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 (C-bit).

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	x00-1

Chaining. This instruction can be chained to the preceding operation by supplying only the operation code. For example, when the contents of storage location 333 are insignificant (or ORG 334 is used), the print area can be cleared by chaining two CLEAR STORAGE instructions to the preceding WRITE instruction.

Example. Clear WAREA5 (0500-0563), Figure B-27.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	IND	(B) OPERAND ADDRESS	±	CHAR. ADJ.	IND	d
			CS	WAREA5								

Autocoder

Label	Operation	OPERAND
	CS	WAREA5

Figure B-27. Clear Storage

Clear Storage and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*B-address*
CS	/	III	BBB

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:
$T = N (7 + L_X)$ ms.

With indexing:
$T = N (8 + L_X)$ ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
Without indexing:	NSI	BI	blank
With indexing:	NSI	BI	NSI

Example. Clear WAREA8 (0800-0898) and branch to START4 (0498) for the next instruction (Figure B-28).

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	IND	(B) OPERAND ADDRESS	±	CHAR. ADJ.	IND	d
			CS	START4				WAREA8				

Autocoder

Label	Operation	OPERAND
	CS	START4,WAREA8

Assembled Instruction: / 498 898

Figure B-28. Clear Storage and Branch

Set Word Mark (Two Addresses)

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
SW	,	AAA	BBB

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

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Set word marks at locations BEGIN1 (3950) and BEGIN2 (3970), Figure B-29.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	(A) OPERAND ±	(A) OPERAND CHAR. ADJ.	(A) OPERAND IND.	(B) OPERAND ADDRESS	(B) OPERAND ±	(B) OPERAND CHAR. ADJ.	(B) OPERAND IND.	d
0 1 0			SW	BEGIN1				BEGIN2				

Autocoder

Label	Operation	OPERAND
	SW	BEGIN1, BEGIN2

Assembled Instruction: , 150 170

Figure B-29. Set Word Mark (Two Addresses)

Set Word Mark (One Address)

Instruction Format.

Mnemonic	*Op Code*	*A-address*
SW	,	AAA

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

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Set a word mark at AREA2 (2901), Figure B-30.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	(A) OPERAND ±	(A) OPERAND CHAR. ADJ.	(A) OPERAND IND.	(B) OPERAND ADDRESS	(B) OPERAND ±	(B) OPERAND CHAR. ADJ.	(B) OPERAND IND.	d
0 1 0			SW	AREA2								

Autocoder

Label	Operation	OPERAND
	SW	AREA2

Assembled Instruction: , R01

Figure B-30. Set Word Mark (One Address)

Clear Word Mark (Two Addresses)

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
CW	□	AAA	BBB

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.

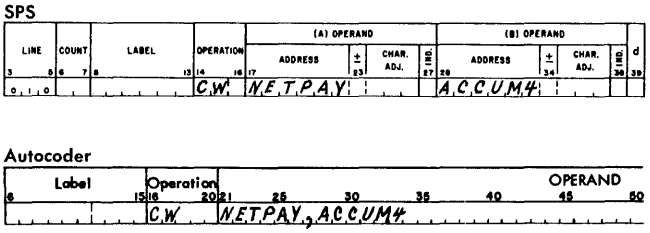
Timing. $T = N\ (L_I + 3)$ ms.

Address Registers After Operation.

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

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Clear the word marks at NETPAY (1924) and ACCUM4 (3309), Figure B-31.



Assembled Instruction: □ Z24 C09

Figure B-31. Clear Word Mark (Two Addresses)

Clear Word Mark (One Address)

Instruction Format.

Mnemonic	*Op Code*	*A-address*
CW	□	AAA

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.

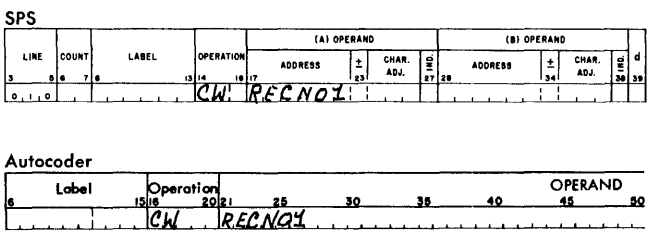
Timing. $T = N\ (L_I + 3)$ ms.

Address Registers After Operation.

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

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Clear the word mark at RECNO1 (3608), Figure B-32.



Assembled Instruction: □ F08

Figure B-32. 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:

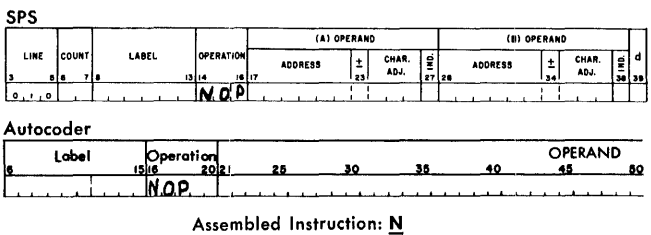
I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
NSI	123	4*bb**

*This factor (an incomplete address or constant) cannot be stored or chained.

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 B-33).



Assembled Instruction: N

Figure B-33. 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. The HALT instruction can have either one or seven characters. When the 7-character format is used, the A- and B-storage address register contents serve to identify the halt.

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 B-34 is a symbolic example of the HALT instruction. Any identifying information (constants, label-defined address constants, etc) can be coded in the A- and B-field operands. In this example, 888 and RETRY (0681) were chosen.

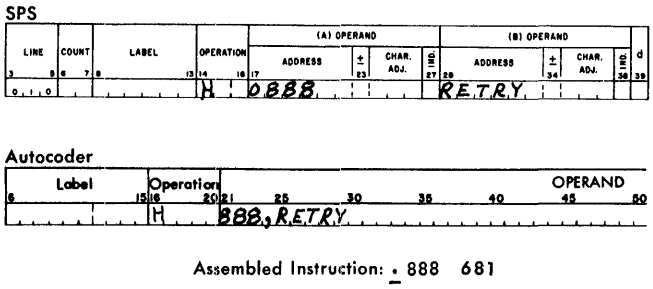


Figure B-34. Halt

Halt and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-Address*
H	.	III

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

This instruction can serve as a manual switch (without tying up a sense switch). If under certain conditions, the operator elects to proceed to the next sequential instruction rather than execute the branch portion of the HALT, he can press the Start Reset and Start keys.

Word Marks. Word marks are not affected.

Timing.

Without indexing:	T = 5N ms.
With indexing:	T = 6N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
Without indexing:	NSI	BI	blank
With indexing:	NSI	BI	NSI

Example. Stop the system, and branch to START2 (0895) for the next instruction when the start key is pressed (Figure B-35).

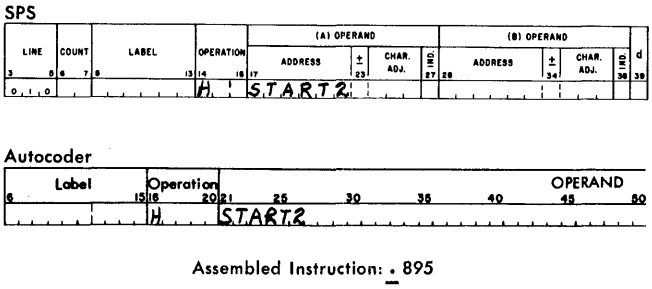


Figure B-35. Halt and Branch

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 B-36).

	OP	A-address	B-address
Edit instruction	E	789	300
		A-field (data)	B-field (control word)
Storage		00257426	$ bbb, bb0.bb & CR & **
			B-field
Result of edit		00257426	$ 2,574.26 **

Figure B-36. 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 unchanged and is not transferred to the B-field.

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 B-37 shows the use of these rules as applied to the data in Figure B-36.

Move Characters and Edit

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
MCE	E	AAA	BBB

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 remains unchanged during the operation. It is used to control portions of the operation, but is not transferred to the B-field.

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. If the edit control word contains a decimal, the edit-control-word zero must be to the left of the decimal. When a positive zero data field is edited (*without* the Expanded Print Edit special feature), the result is .00; when a negative zero field is edited the result is .00 CR (if CR was included in the edit control word). Note that *with* the Expanded Print Edit feature, nothing prints for a positive-zero field, and only the CR prints for a negative-zero field.
. (decimal)	This remains in the edited field where it was placed by loading the edit control word. It is removed during a zero-suppression 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 *Expanded Print Edit* in *Special Features* – Section I).
, (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.
– (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 *Expanded Print Edit* in *Special Features* – Section I).
$ (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 *Expanded Print Edit* in *Special Features* – Section I).

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	?	?	E		E	$bbb,bb0.bb&CR**	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	OP	0	OP	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	$bbb,bb0.bb&CRb**	Rule 1
13	B	008	0788	0296	R	6	Blank	$bbb,bb0.bb&Cbb**	Rule 1 and 5
14	B	008	0788	0295	C	6	Blank	$bbb,bb0.bb&bbb**	Rule 1 and 5
15	B	008	0788	0294	&	6	Blank	$bbb,bb0.bbbbbb**	Rule 1
16	B	008	0788	0293	b	6	6	$bbb,bb0.b6bbbb**	Rule 1
17	A	008	0787	0293	2	2	2	same	Rule 1
18	B	008	0787	0292	b	2	2	$bbb,bb0.26bbbb**	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	$bbb,bb4.26bbbb**	Zero Suppress—Rule1 and 7
22	A	008	0785	0290	7	7	7	same	Rule 1
23	B	008	0785	0289	b	7	7	$bbb,b74.26bbbb**	Rule 1
24	A	008	0784	0289	5	5	5	same	Rule 1
25	B	008	0784	0288	b	5	5	$bbb,574.2bbbbb**	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	$bb2,574.26bbbb**	Rule 1
29	A	008	0782	0286	0	0	0	same	Rule 1
30	B	008	0782	0285	b	0	0	$b02,574.26bbbb**	Rule 1
31	A	008	0781	0285	0	0	0	same	Rule 1
32	B	008	0781	0284	b	0	0	$002,574.26bbbb**	Rule 1
33	B	008	0781	0284	$	0	$	$002,574.26bbbb**	Sense Word Mark—Rev. Scan—Rule 1 and 6
34	B	008	0781	0285	$	0	$	same	Rule 6
35	B	008	0781	0286	0	0	Blank	$b02,574.26bbbb**	Rule 6
36	B	008	0781	0287	0	0	Blank	$bb2,574.26bbbb**	Rule 6
37	B	008	0781	0288	2	0	2	same	Rule 6
38	B	008	0781	0289	,	0	,	same	Rule 6
39	B	008	0781	0290	5	0	5	same	Rule 6
40	B	008	0781	0291	7	0	7	same	Rule 6
41	B	008	0781	0292	4	0	4	$bb2,574.26bbbb**	Rule 6

Figure B-37. 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. An A-field word mark is required for proper operation.

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 5. If the data field is positive, and if the CR or – symbols are located in the status portion of the control word, they are blanked out.

Rule 6. The data field can contain fewer, but must not contain more, positions than the number of blanks and zeros in the body of the control word. Dollar signs and asterisks are included in the body of the control word with the expanded print edit special feature.

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 B-38).

A-field	0010900
Control word (B-field)	$ bb, bb0. bb
Forward scan	$ 00,109. 00
Reverse scan	$ bbb109. 00
Results of edit	$ 109. 00

Figure B-38. Zero Suppression

Zero Suppression Operation

Zero suppression is the deletion of unwanted zeros to the left of significant digits in an output field (Figure B-38).

A special 0 is placed (in the body of the control word) in the right-most limit of zero suppression.

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.

Forward Scan.

1. The positions in the output field to 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.*

Reverse Scan.

1. In the output field, blanks replace all zeros and punctuation, except hyphens to the left of the first significant character (up to and including the zero-suppression code position).
2. If the edit-control-word zero is to the *right* of the decimal, zero suppression is re-activated: zeros to the right of the decimal are removed. (See the *Expanded Print Edit* section.)
3. When the automatically-set zero-suppression word mark is sensed, it is erased and the operation ends.

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

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
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.

Chaining. This instruction is not normally chained.

Example. Edit the data labeled GROPAY (0985) by the edit-control word EDCONT (0325). Store the result in PRINT6 (0250), Figure B-39.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	I/O	(B) OPERAND ADDRESS	±	CHAR. ADJ.	I/O	d
010			LCA	EDCONT				PRINT6				
020			MCE	GROPAY				PRINT6				

Autocoder

Label	Operation	OPERAND
	MLCWA	EDCONT,PRINT6
	MCE	GROPAY,PRINT6

Assembled Instruction: L 325 250
E 985 250

Figure B-39. Edit

The IBM 1406 Storage Unit (Figure C-1) provides a means of increasing a 4,000-position 1401 system to 8,000; 12,000; or 16,000 core-storage positions.

IBM 1406 Models

The 1406, Model 1, contains a block of 4,000 core-storage positions. It increases the capacity of the system to 8,000 positions.

Model 2 contains a block of 8,000 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.

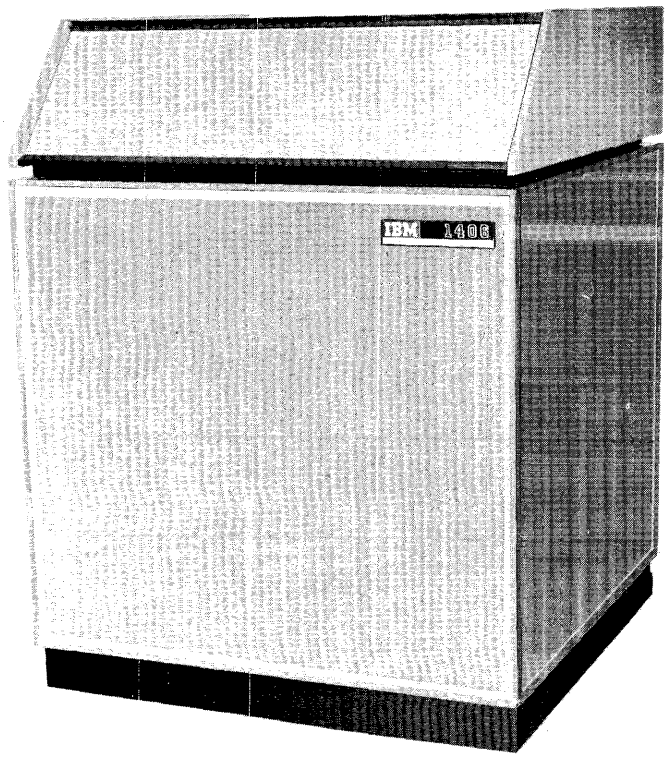


Figure C-1. IBM 1406 Storage Unit

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 4,000-character coding system.

Storage Addresses	*Zone Bits over Units Position of Address*
4000- 7999	A
8000-11999	B
12000-15999	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 Only	1,400	0000-1399	1400-15999
	2,000	0000-1999	2000-15999
	4,000	0000-3999	4000-15999
1401 and 1460	8,000	0000-7999	8000-15999
	12,000	0000-11999	12000-15999
	16,000	0000-15999	NONE

Address Arithmetic

To facilitate address arithmetic, an additional operation code (MODIFY ADDRESS) is added to IBM 1401 systems equipped with more than 4,000 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	#	AAA	BBB

Function. This instruction causes the 3-character A-field to be added to the 3-character B-field. Either or both fields can be indexed. The result, stored in the B-field, is comprised of the sum of the numeric

portions of the A-field and B-field, the address-zone bits of both fields, and the *original* B-field indexing bits (if any).

The MODIFY ADDRESS instruction operates as follows:

1. No *factor* or *address* indexing –
   001 modified by 567 = 568
   or
   D4B (15,442) modified by TO/ (5,301) = (20,743 – 16,000) = 74T (4,743).
2. A-field *factor* indexing –
   EFG (15,567 + X3) modified by 111 = F7H (15,678). Resultant B-field is not indexed because original B-field was not indexed.
3. B-field *factor* indexing –
   001 modified by JKL (10,123 + X2) = JKM (10,124 + X2). Index-register contents can be any valid address factor. Resultant B-field retains the original B-field indexing.
4. A- and B-field *factor* indexing – Same as B-field factor indexing.
5. A- and/or B-field *address* indexing (such as # 2C4 5W7) –
   Result depends on contents of index registers used as well as *factor* indexing of the addressed fields, using the basic rules outlined in the four above cases.

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:

$T = N (L_I + 9)$ ms.

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

$T = N (L_I + 8)$ ms.

NOTE 1: See rules for the addition of zone bits in the section, *Address Modification Without Indexing.*

NOTE 2: The SPS-1 assembler generates an ADD instruction for the Modify Address mnemonic. The SPS-2 assembler generates an ADD or a MODIFY ADDRESS instruction, depending on the SPS-2 control card.

Address Registers After Operation.

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

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Add the 3-character field labeled ADDA (0985) to the 3-character field labeled ADDB (1313), Figure C-2.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	(B) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	d
			MA	A.D.D.A.				A.D.D.B.				

Autocoder

Label	Operation	OPERAND
	MA	ADDA, ADDB

Assembled Instruction: # 985 T13

Figure C-2. Modify Address (Two Addresses)

Modify Address (One Address)

Instruction Format.

Mnemonic	*Op Code*	*A-address*
MA	#	AAA

Function. This format of the MODIFY ADDRESS instructions 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:

$T = N (L_I + 9)$ ms.

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

$T = N (L_I + 8)$ ms.

Address Registers After Operation.

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

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Double the address labeled ADDC (2956), and store the result at ADDC (Figure C-3).

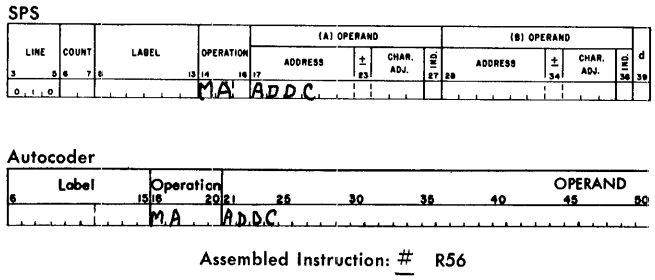


Figure C-3. Modify Address (One Address)

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 15 are equal to that value minus 16. 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 C-4).

			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 C-4. 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 4,000 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 position. 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 (H3I) complement
Y29 + H3I = 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).

The IBM 1447 Console (Model 1, 2, or 4) (Figure D-1) is a required unit on an IBM 1460 Data Processing System. The console contains the system's operating 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 D-2) is available as a console inquiry station with IBM 1409 Console Auxiliary (Figure D-3) 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, provide 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.

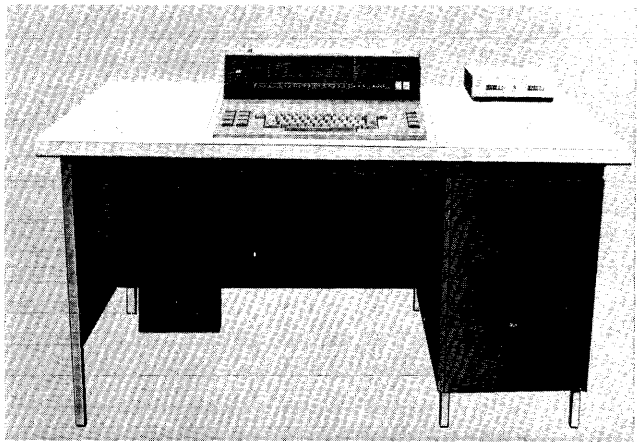


Figure D-2. IBM 1447 Console, Model 3

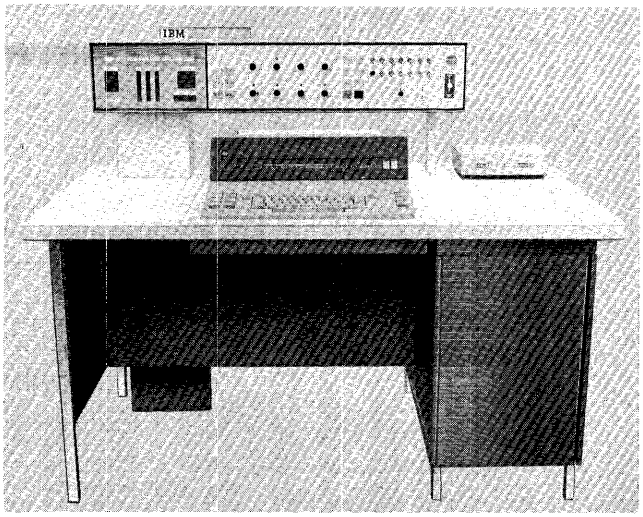


Figure D-1. IBM 1447 Console, Model 2

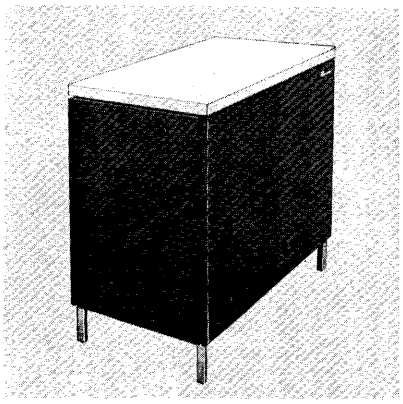


Figure D-3. IBM 1409 Console Auxiliary

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 D-4.

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

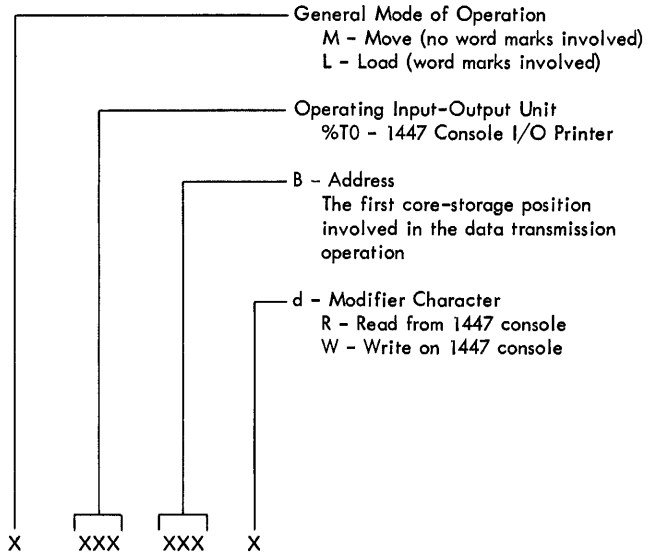


Figure D-4. IBM 1447 Console I/O Printer Instruction Format

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.

IBM 1447 Console Instructions

Instructions applying to the IBM 1447 cannot be successfully chained.

Read from 1447 Console

Instruction Format.

	Mnemonic	*Op Code*	*A-address*	*B-address*	*d-character*
SPS	MU	M	%T0	BBB	R
	LU	L			
A	RCP	M			
	RCPW	L			

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 signifies that the input message exceeded the core-storage area capacity and:

1. The operation ends and the printer is disconnected from the system.
2. The inquiry clear (*) indicator in the system comes ON.
3. The red TYPE light on the console comes ON.
4. A carrier-return and line-feed operation is initiated.
5. The keyboard locks up.

Word Marks. Depends on mode of operation. To end the operation correctly, a group-mark with a word-mark must be inserted into the core-storage position to the right of the position that contains the last character sent to the system from the console printer.

Timing. T = 9N ms + operator keying time.

Address Registers After Operation.

I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
NSI	%30	$B + L_B + 1$

Example. Transfer the data keyed on the console I/O printer to the area in core storage labeled INQIN (0785), Figure D-5.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	WM	(B) OPERAND ADDRESS	±	CHAR. ADJ.	WM	d
3 5	6 7	8 13	14 16	17 23			27	28 34			36	38
0 1 0			MU	%T0				INQIN				R

Autocoder

Label	Operation	OPERAND
6 15	16 20 21	25 30 35 40 45 50
	RCP	INQIN

Assembled Instruction: M %T0 785 R

● Figure D-5. Read from 1447 Console (Move Mode)

Write on 1447 Console

Instruction Format.

	Mnemonic	*Op Code*	*A-address*	*B-address*	*d-character*
SPS	MU	M	%T0	BBB	W
	LU	L			
A	WCP	M			
	WCPW	L			

Function. This instruction is used to transfer data from core storage to the console I/O printer. The Op code specifies the mode of operation. If the operation takes place in the move mode, word marks are ignored. The character with an associated word mark in core storage is printed as a character only. Functional control characters cause the specified carrier movement on the console printer, and the characters do not print. Refer to IBM *1447 Console* (Form A24-3031) for functional control characters and associated printer operation.

If the operation takes place in the load mode, the word marks are transmitted and printed. The word mark is printed before the associated character is printed. Functional control characters are also printed. The carrier movement normally specified by the character does not occur.

The A-address specifies the console I/O printer as the I/O unit involved in the operation, and turns on the white TYPE light if the printer is available for use. The B-address specifies the first core-storage position of the area that contains the data to be printed. The d-character W specifies a system-to-console printer operation.

The data reads out of core storage, beginning at the address specified in the instruction and continuing until a group-mark with a word-mark is encountered. The group-mark with a word-mark ends the operation, but does not print. A carrier-return operation, with an associated line-feed operation, occurs and the system advances to the next instruction.

If the end of a printed line is reached before the group-mark with a word-mark is sensed, printing is suspended and a carrier-return and line-feed operation is executed. When the carrier reaches the left-hand margin, the print-out operation continues.

Word Marks. Depends on mode of operation. A group-mark with a word-mark in core storage ends the operation.

Timing. $T = 9N + 68\ (L_B) + 800$ (number of carrier return operations -1) ms.

Address Registers After Operation.

I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
NSI	%30	B + L_B + 1

Example. Print out the data, beginning in the area labeled INQOUT (0785) end ending with a group-mark with a word-mark (Figure D-6).

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	IND	(B) OPERAND ADDRESS	±	CHAR. ADJ.	IND	d
3 5	6 7	8 13	14 16	17	23		27	28	34		36	39
0 1 0			MU	%TO				INQOUT				W

Autocoder

Label	Operation	OPERAND
6 15	16 20 21	25 30 35 40 45 50
	WCP	INQOUT

Assembled Instruction: M %T0 785 W

● Figure D-6. Write on 1447 Console (Move Mode)

Branch if Inquiry Request

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BIN	B	III	Q

Function. This branch indicator usually signifies that the console I/O printer has a message to send to the system. The indictor turns on during a console-inquiry operation when the operator presses the TYPE key.

NOTE: Refer to *IBM 1447 Console,* Form A24-3031, for special feature considerations and additional information concerning this instruction.

Word Marks. Word marks are not affected.

Timing.

No Branch, or Branch without indexing:	T = 6N ms.
Branch with indexing:	T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch	NSI	BI	*Qbb*
Branch (without indexing)	NSI	BI	blank
Branch (with indexing)	NSI	BI	NSI

Branch if Inquiry Clear

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BIN	B	III	*

Function. When special features are not involved, this branch indicator and the associated red type light are turned on when:

1. The console operator makes a keying mistake and instructs the system to disregard the message (by pressing the CANCEL key).
2. The input message exceeds the core-storage area capacity.
3. The 1447 circuitry detects a parity error during the 1447-to-core-storage transfer.
4. The 1447 circuitry detects a parity error during the core-storage-to-1447 transfer.

NOTE: Refer to *IBM 1447 Console,* Form A24-3031, for special feature considerations and additional information concerning this instruction.

Word Marks. Word marks are not affected.

Timing.

No Branch, or Branch without indexing:	T = 6N ms.
Branch with indexing:	T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch	NSI	BI	**bb*
Branch (without indexing)	NSI	BI	blank
Branch (with indexing)	NSI	BI	NSI

IBM 1402 Card Read-Punch Operations

This section describes the instructions the 1401 and 1460 use to control the IBM 1402 Card Read-Punch (Figure E-1). The basic unit timings are also included with the other general information.

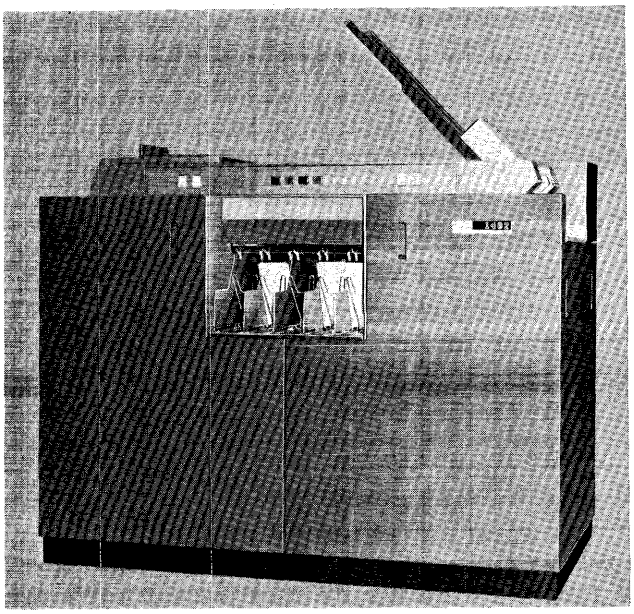


Figure E-1. IBM 1402 Card Read-Punch

Data Flow

The data flow and feed schematic (Figure E-2) 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; Model 4 or 5 for the 1401G 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 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.

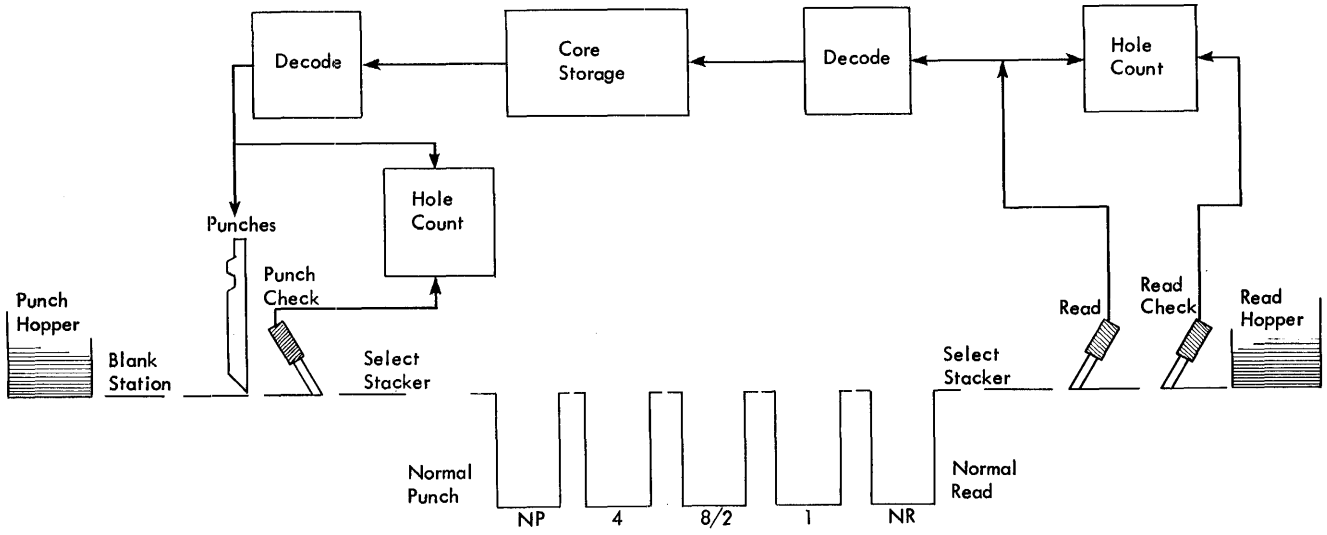


Figure E-2. Data Flow — IBM 1402 Card Read-Punch

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 readers (Models 2 and 3) operate 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. The card reader (Models 4 and 5) operates at a rated speed of 450 cycles per minute (one cycle every 133⅓ milliseconds). (See *Card Read-Punch* timing.)

Card Punches

The card punches (Models 1, 3, 4, and 5) operate 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

Instructions applying to the IBM 1402 cannot be chained.

Read Card

Instruction Format.

Mnemonic	*Op Code*
R	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 = 2N ms + I/O.

A card read cycle (Models 1 and 3) require a total of 75 milliseconds. The cycle is divided into three separate sections (Figure E-3).

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 in the following read cycle. The processing unit is interlocked during *read start time.*
2. Card read time is 44 ms. The actual reading of the card takes place during this part of the cycle and the data is read into core storage. The processing unit is interlocked during *card reading time.*
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.

The card-read portion of the 1402 Model 4 or 5 can operate at a speed of 450 cards per minute (CPM). The actual card speed realized is governed by the program routine for each particular run. A card-read speed of 450 CPM results in a read cycle of 133⅓ milliseconds (ms). Refer to Figure E-4 for a timing schematic of the read cycle. Early card read is a standard feature and minimizes the decrease in card-reading speed caused by lengthy processing routines. This is done by permitting the card-reading mechanism to engage sooner.

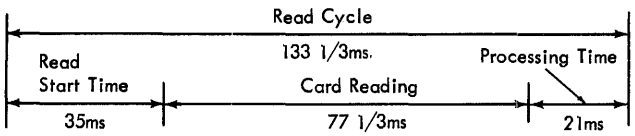


Figure E-4. IBM 1402, Model 4, Read Operation 450 CPM

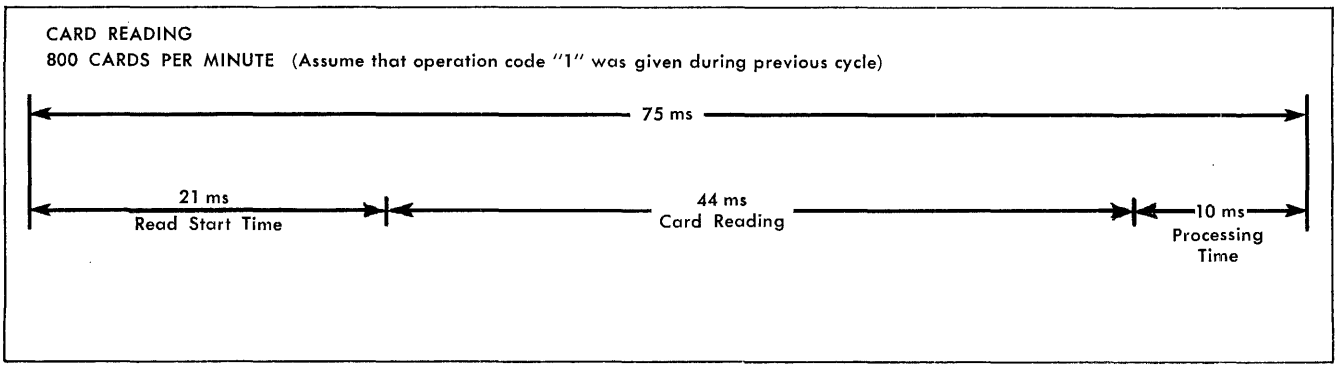
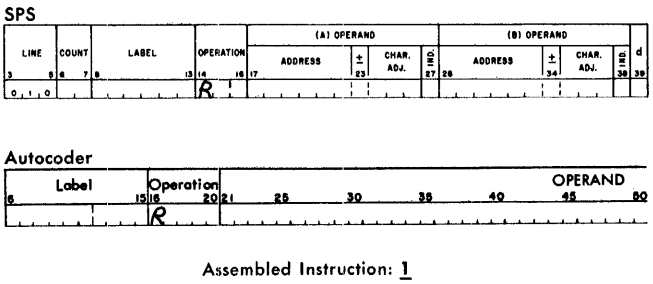


Figure E-3. Card Read Cycle (Models 1 and 3)

Address Registers After Operation.

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

Example. Read a card (Figure E-5).



Assembled Instruction: 1

Figure E-5. Read Card

Read Card and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-address*
R	1	III

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

Word Marks. Word marks are not affected.

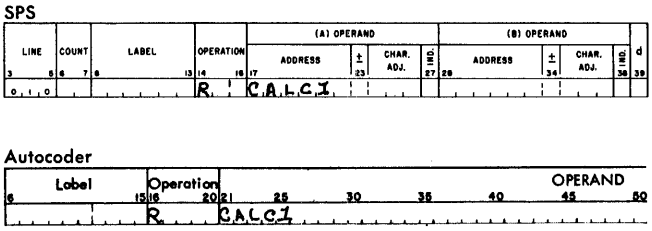
Timing.

Without indexing:	T = 5N ms + I/O.
With indexing:	T = 6N ms + I/O.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
Without indexing:	NSI	BI	081
With indexing:	NSI	BI	NSI

Example. Read a card, and branch to CALC1 (1500), Figure E-6.



Assembled Instruction: 1 V00

Figure E-6. Read Card and Branch

Punch Card

Instruction Format.

Mnemonic	*Op Code*
P	4

Function. The data in storage locations 101 through 180 is punched into a card.

Word Marks. Word marks are not affected.

Timing. T = 2N ms + I/O.

There are four points in the cycle (occurring at 60-millisecond intervals) when the punch feeding mechanism can receive an impulse to start the punch cycle.

The punch cycle is divided into three separate sections (Figure E-7).

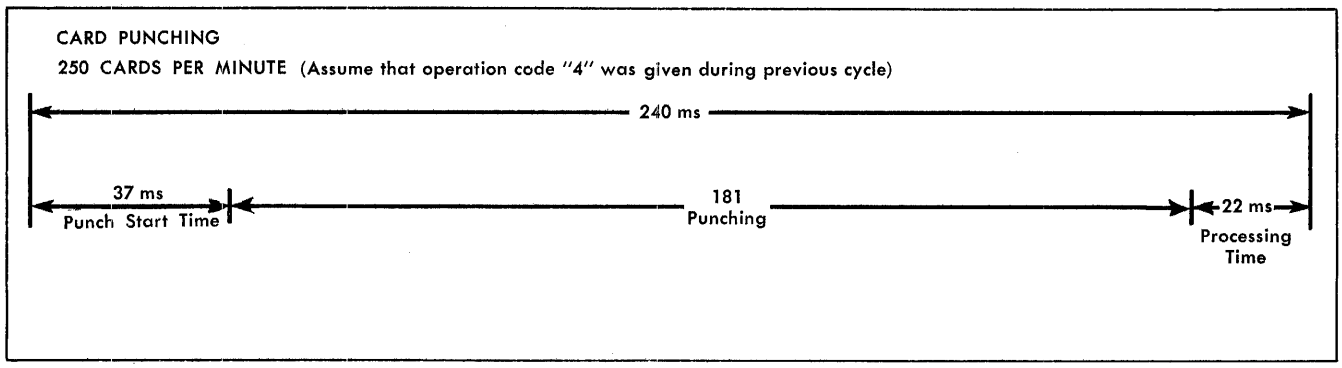


Figure E-7. Punch Cycle (Models 1, 3, and 4)

1. Punch start time is 37 ms. After the feed mechanism has been impulsed, the time required for the card to feed and be positioned for punching is called *punch start time.* The IBM 1401 or 1441 Processing Units are interlocked during punch start time.

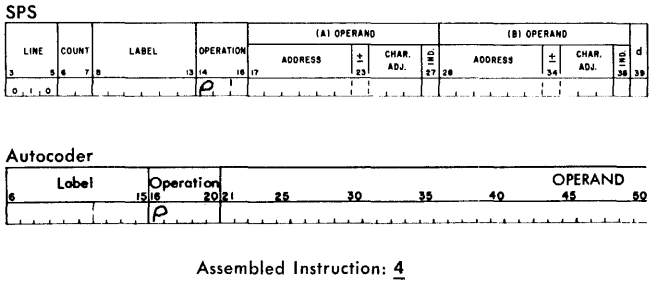
2. Card punching time is 181 ms. The actual punching of the card takes place during this part of the cycle. The IBM 1401 or 1441 Processing Units are always interlocked during *card-punching time.*

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 E-8).



Assembled Instruction: 4

Figure E-8. Punch Card

Punch Card and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-address*
P	4	III

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.

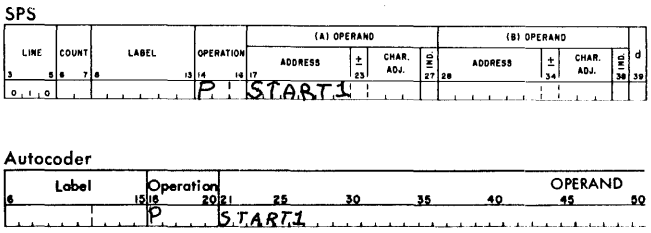
Timing.

Without indexing:	T = 5N ms + I/O
With indexing:	T = 6N ms + I/O

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
Without indexing:	NSI	BI	181
With indexing:	NSI	BI	NSI

Example. Punch a card, and branch to START1 (1758), Figure E-9.



Assembled Instruction: 4 X58

Figure E-9. Punch Card and Branch

Select Stacker

Instruction Format.

Mnemonic	*Op Code*	*d-character*
SS	K	d

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.

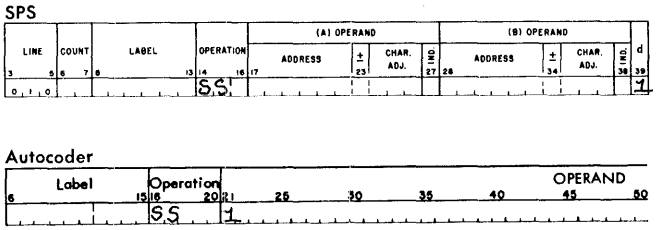
Word Marks. Word marks are not affected.

Timing. T = 3N ms.

Address Registers After Operation.

I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
NSI	d*bb*	d*bb*

Example. Enter the last card read into pocket 1 (Figure E-10).



Assembled Instruction: K 1

Figure E-10. Select Stacker

Select Stacker and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
SPS SS	K	III	d
A SSB			

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

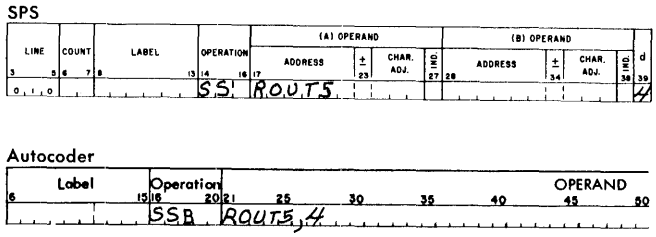
Timing.

Without indexing:	T = 6N ms.
With indexing:	T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
Without indexing:	NSI	BI	d*bb*
With indexing:	NSI	BI	NSI

Example. Select the last card punched, enter it in pocket 4, and branch to ROUT5 (0950) for the next instruction (Figure E-11).



Assembled Instruction: K 950 4

Figure E-11. Select Stacker and Branch

Read and Punch

Instruction Format.

Mnemonic	*Op Code*
RP	5

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 E-18 for timing details.

Word Marks. Word marks are not affected.

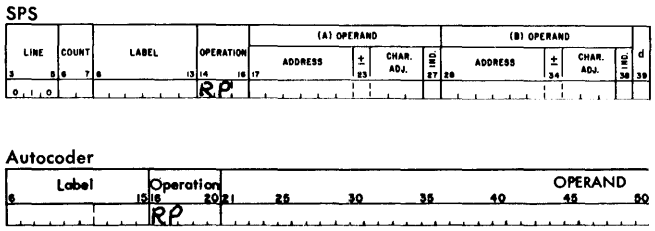
Timing. T = 2N ms + I/O.

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. In the 1401G, the A-register controls the data and the B-register can be at any previous address.

Example. Read a card into the input area, and punch a card with information from the punch area (Figure E-12).



Assembled Instruction: 5

Figure E-12. Read and Punch

Read, Punch and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-address*
RP	5	III

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.

Without indexing: T = 5N ms + I/O.
With indexing: T = 6N ms + I/O.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.**
Without indexing:	NSI	BI	181 or 081
With indexing:	NSI	BI	NSI

*(See note under *Read and Punch.*)

Example. Read, punch, and branch to WORK5 (0596), Figure E-13.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	(A) OPERAND ±	(A) OPERAND CHAR. ADJ.	(A) OPERAND MOD	(B) OPERAND ADDRESS	(B) OPERAND ±	(B) OPERAND CHAR. ADJ.	(B) OPERAND MOD	d
			RP	WORK5								

Autocoder

Label	Operation	OPERAND
	RP	WORK5

Assembled Instruction: 5 596

Figure E-13. Read, Punch, and Branch

Branch if Reader Error

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BIN	B	III	? (plus zero)

Function. If an error occurs during the card-read operation, this indicator is set, and the reader light glows on the console. If the I/O check-stop switch is OFF, this indicator can be tested and reset by the BRANCH-IF INDICATOR ON instruction. If the indicator is ON, the program is interrupted, and a branch to the I-address occurs. If the indicator is not ON, no branch occurs.

Word Marks. Word marks are not affected.

Timing.

No Branch, or Branch without indexing: T = 6N ms
Branch with indexing: T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	?*bb*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Branch if Punch Error

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BIN	B	III	! (minus zero)

Function. Same as for *Reader Error,* except that specifics apply to the punch operation.

Word Marks. Word marks are not affected.

Timing.

No Branch, or Branch without indexing: T = 6N ms
Branch with indexing: T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	!*bb*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Branch if Last Card

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BLC	B	III	A

Function. This instruction causes a branch to the address specified by the I-address, if the last card has *been* read and is *ready* for punching.

Word Marks. Word marks are not affected.

Timing.

No Branch, or Branch without indexing: T = 6N ms
Branch with indexing: T = 7N ms

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	A*bb*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

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 E-14). 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 E-15 is a table showing card reading speeds, and the processing time alloted for each.

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 E-16).

Figure E-17 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.

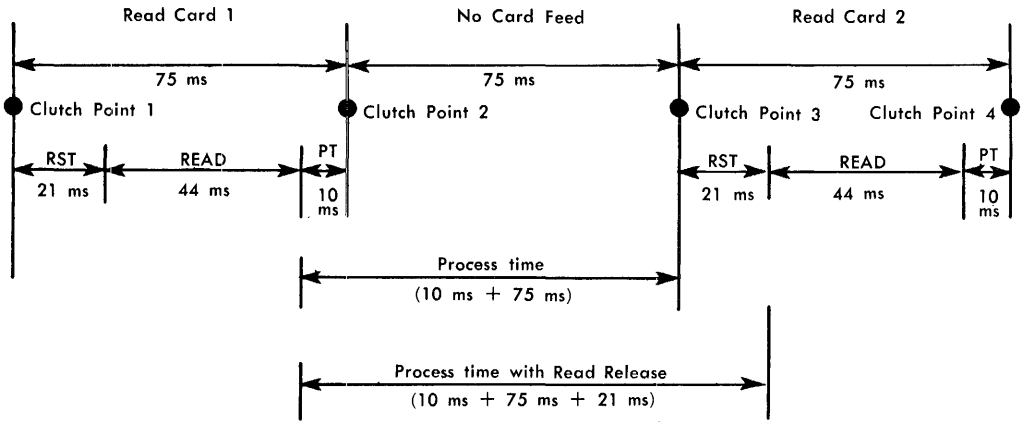


Figure E-14. Read Operation Timing Chart

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

Figure E-15. Card Reading Speeds

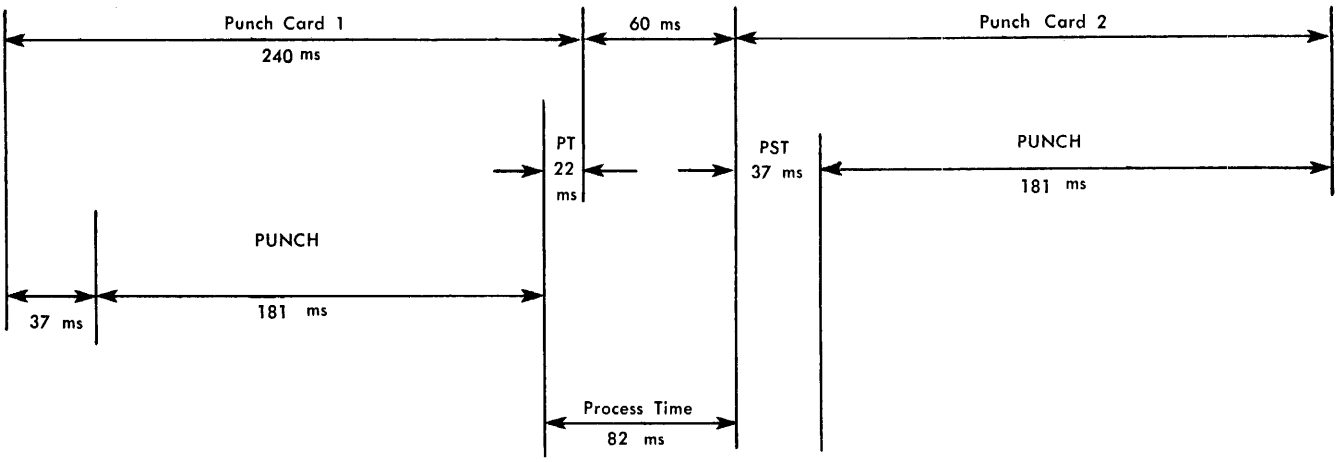


Figure E-16. Punch Operation Timing Chart

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 E-17. Card Punching Speeds

Figure E-18 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.

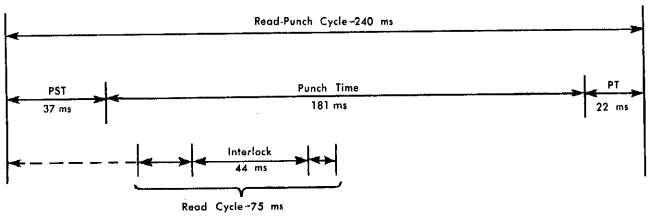


Figure E-18. Read and Punch Operation Timing Chart

Interleaving Input-Output Operations

It is possible to interleave either a read operation or a punch operation with a print operation on the 1401 Model G system.

Write and Read Operation

The write and read operation is performed by executing the print operation during the read start time and card reading time of a read cycle, as shown in Figure E-19. Depending on the program routine, this operation can result in a card-read speed and printing speed of 450 cards/lines per minute.

Write and Punch Operation

The write and punch operation is performed by executing the print operation during the punch start time and card punching time of the punch cycle as shown in Figure E-20. Depending on the program routine, this operation can result in a card-punch and printing speed of 250 cards/lines per minute.

Write, Read, And Punch Operation

The write, read, and punch operation is not completely interleaved. The write and punch portion of the operation is interleaved, and the read operation follows the interleaved operation. This operation results in a printing and card-reading/card-punching speed of 200 cards/lines per minute.

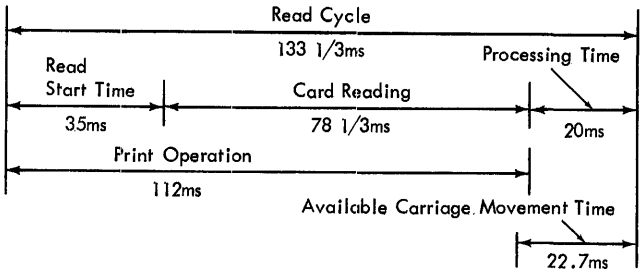


Figure E-19. Write and Read Operation

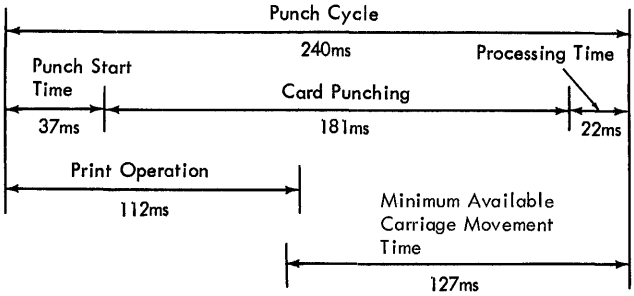


Figure E-20. Write and Punch Operation

IBM 1403 Printer Operations

This section describes the instructions the 1401 and 1460 use to control the IBM 1403 Printer (Figure E-21). 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.

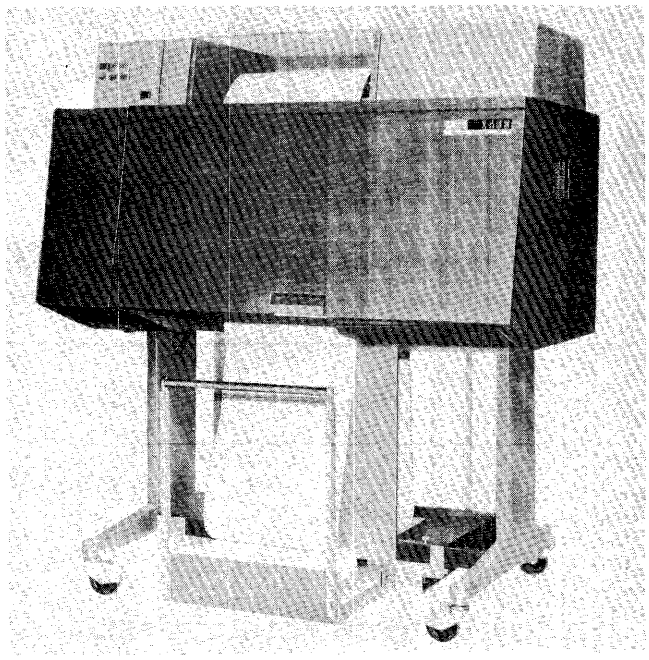


Figure E-21. IBM 1403 Printer

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 E-22).

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.

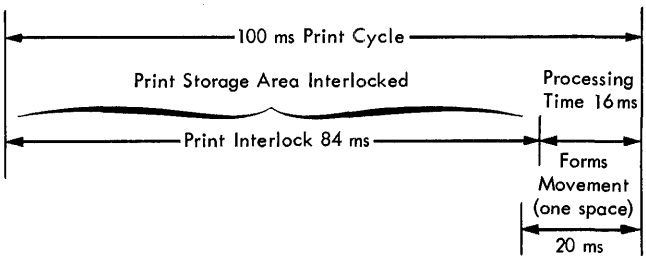


Figure E-22. Print Operation Timing Chart, IBM 1403 Models 1 and 2

2. Process time is 16 ms. This is the normal *processing time* available during the print cycle (see *Print Storage* in *Special Features,* Section I).
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 E-23).

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.

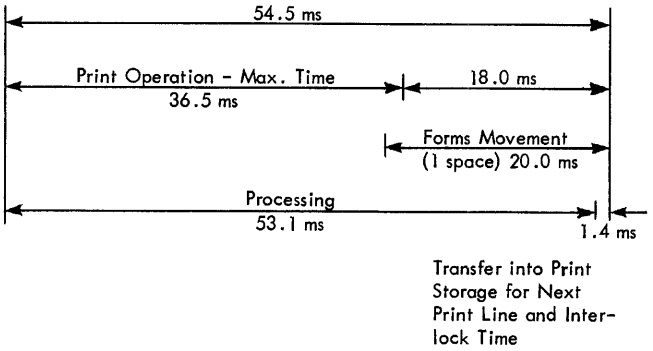


Figure E-23. Print Operation Timing Chart IBM 1403, Model 3

IBM 1403 Printer Models 4 and 5

Either Model 4 or Model 5 of the IBM 1403 Printer provides the 1401 Model G system with a printed output. Model 4 has a printing capacity of 100 positions; Model 5 has a printing capacity of 132 positions. Both printers are equipped with a dual-speed carriage. With the exception of speed, the operating characteristics of Models 4 and 5 are the same as the Models 1 and 2 used on other IBM 1401 Data Processing System configurations.

Models 4 and 5 of the printer have a rated printing speed of 465 lines per minute (LPM). The actual printing speed realized is governed by the program routine for each particular run. A printing speed of 465 LPM results in a print cycle of 129 ms. Refer to Figure E-24 for a timing schematic of a print cycle.

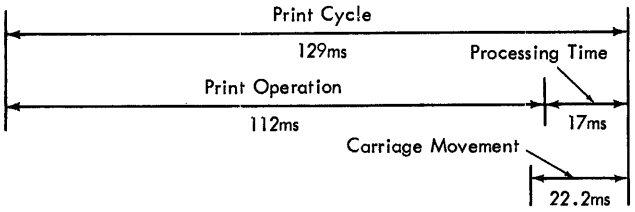


Figure E-24. IBM 1403, Model 4 or 5, Print Operation — 465 LPM

IBM 1403 Printer Instructions

Instructions applying to the IBM 1403 cannot be chained, but see the *Clear Storage* section.

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.

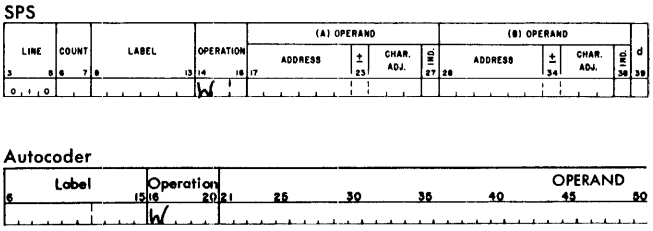
Timing. T = 2N ms + I/O.

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,* Section I).

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 E-25).



Assembled Instruction: 2

Figure E-25. Write Line

Write Line and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-address*
W	2	III

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.

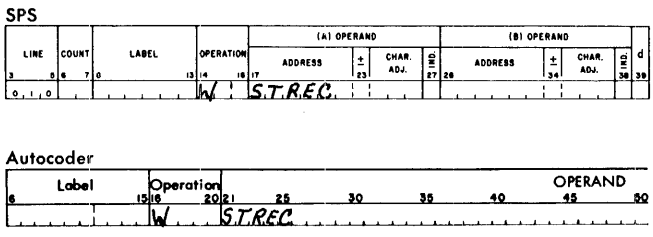
Timing.

Without indexing: T = 5N ms + I/O.
With indexing: T = 6N ms + I/O.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
Without indexing:	NSI	BI	333
With indexing:	NSI	BI	blank

Example. Print, and branch to STREC (0678) for the next instruction (Figure E-26).



Assembled Instruction: 2 678

Figure E-26. Write Line and Branch

Write Word Marks

Instruction Format.

Mnemonic	*Op Code*	*d-character*
SPS W	2	□
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 = 3N ms + I/O.

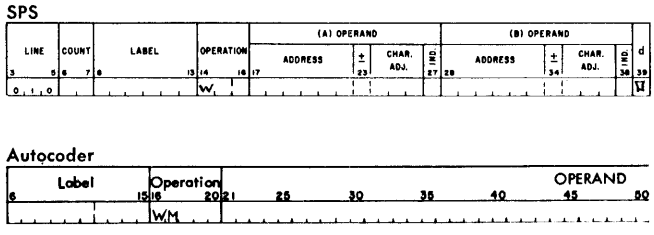
NOTE: On 1460 systems, the WRITE WORD MARKS instruction also causes:

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-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
NSI	d*bb*	335 333 (print storage)

Example. Print all word marks in the print area (Figure E-27).



Assembled Instruction: 2 □

Figure E-27. Write Word Marks

Write Word Marks and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
SPS W	2	III	□
A WM			

Function. This is the same as WRITE 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.

Without indexing:	T = 6N ms + I/O.
With indexing:	T = 7N ms + I/O.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
Without indexing:	NSI	BI	333
With indexing:	NSI	BI	blank

Example. Print word marks, and branch to RESTAR (0890), Figure E-28.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	WM	(B) OPERAND ADDRESS	±	CHAR. ADJ.	WM	d
3 8	6 7	8 13	14 16	17 23			27	28	34		38	39
0 1 0			W	RESTAR								

Autocoder

Label	Operation	OPERAND
6 15	20 21 25 30 35 40 45 50	
	WM	RESTAR

Assembled Instruction: 2 890 □

Figure E-28. Write Word Marks and Branch

Branch if Printer Error

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BIN	B	III	÷

Function. If an error occurs during a printer operation, this indicator is set ON, and the printer light glows on the console. This indicator can be tested to effect a branch. If the indicator is on, it is reset. The next instruction to be executed is at the location specified by the I-address of the BRANCH IF INDICATOR ON instruction.

NOTE: This error indicator applies to any 1403, 1443, or 1445 attached to the system, when the I/O check-stop switch is OFF.

Word Marks. Word marks are not affected.

Timing.

No Branch, or Branch without indexing:	T = 6N ms
Branch with indexing:	T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch	NSI	BI	‡*bb*
Branch (without indexing)	NSI	BI	blank
Branch (with indexing)	NSI	BI	NSI

Branch if Printer Busy

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BPB	B	III	P

Function. If the printer or printer carriage is currently occupied with another operation, this indicator can be tested to effect a branch to another series of instructions. The indicator is reset as soon as the printer is available for another operation. Using this instruction allows processing to continue while the printer is busy, thus, in effect, allowing temporary overlapping of processing and printer operation.

NOTE 1: This indicator is effective for any 1403, 1443, or 1445 attached to the system.

NOTE 2: The BRANCH IF PRINTER BUSY and BRANCH IF PRINTER CARRIAGE BUSY instructions should be included in any program where one print and space/skip operation has not been completed before the next print and space operation is initiated. If these instructions are not used, incorrect carriage spacing or skipping can result. The following sample program segment illustrates how these instructions might be coded:

```
PRBUSY   BPB    PRBUSY
PCBUSY   BPCB   PCBUSY
SPACE    CC     S
WRITE    W      NET PAY
         NSI
```

Frequently, there are other processing steps that can be performed while waiting for the carriage to complete the last print/space/skip function. The BRANCH IF PRINTER BUSY and BRANCH IF PRINTER CARRIAGE BUSY instructions could have branched to perform these other functions.

This precaution applies only to systems having buffered printers.

Word Marks. Word marks are not affected.

Timing.

No Branch, or Branch without indexing:	T = 6N ms
Branch with indexing:	T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	P*bb*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Branch if Printer Carriage Busy

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BPCB	B	III	R

Function. If the printer carriage is executing a forms-movement operation, this instruction can be used to branch to other instructions until the carriage is again available. The indicator is reset as soon as the printer carriage is no longer busy. Using this instruction allows processing to continue while the printer carriage is busy, thus, in effect, permitting temporary overlapping of processing and printer operations

NOTE: See Note 2 under *Branch if Printer Busy* section.

Word Marks. Word marks are not affected.

Timing.

No Branch, or Branch without indexing:	T = 6N ms
Branch with indexing:	T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	R*bb*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Branch if Channel 9

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BC9	B	III	9

Function. If the channel-9 position of the carriage control tape has been sensed, this instruction will cause a branch to the address specified by the I-address. This indicator is reset by the branch test, or by a channel-1 punch in the carriage-control tape.

Word Marks. Word marks are not affected.

Timing.

No branch, or branch without indexing:	T = 6N ms.
Branch with indexing:	T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch	NSI	BI	9*bb*
Branch (without indexing)	NSI	BI	blank
Branch (with indexing)	NSI	BI	NSI

Branch if Channel 12

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BCV	B	III	@

Function. If the channel 12 (forms overflow) position of the carriage control tape has been sensed, this instruction will cause a branch to the address specified by the I-address. This indicator is reset by the branch test, or by a channel-1 punch in the carriage-control tape.

Word Marks. Word marks are not affected.

Timing.

No branch, or branch without indexing:	T = 6N ms.
Branch with indexing:	T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	@*bb*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Carriage-Tape-Punch Considerations

A channel-1 carriage-tape punch is required for each form at the location of the first line of printing.

On unbuffered printers, a branch on channel 9 or 12 always interlocks the system. The last line of printing is one line *above* the channel-9 or -12 punch in the carriage tape.

On buffered printers, a branch on channel 9 or 12 immediately *before* printing will cause the last line to print one line *above* the channel-9 or -12 punch in the carriage tape. The same is true if the overflow-branch occurs while the printer is busy (currently engaged in the printing of a line). If the overflow (branch if channel 9 or 12) occurs immediately *after* printing, the last printed line of that form will be *on* the line corresponding to the channel-9 or -12 carriage-tape punch.

Control Carriage

Instruction Format.

Mnemonic	*Op Code*	*d-character*
CC	F	d

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

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 E-29) 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 = 3N 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.

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 E-29. d-Characters for Control Carriage

NOTE: Care should be taken when punching channels 9 and 12 of the carriage control tape 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.

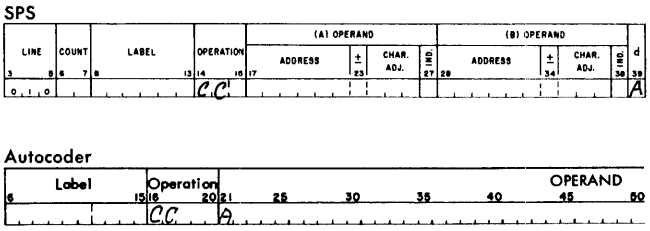
When an immediate skip or immediate space instruction is used, an additional space caused by the automatic carriage space is taken after printing. When a skip after print or space after print instruction is used, the automatic space is not taken.

When the carriage tape is positioned at a channel-1 punch, a skip to channel 1 immediate instruction (F1) does not move the paper.

Address Registers After Operation.

I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
NSI	d*bb*	d*bb*

Example. Skip to channel 1 after print (Figure E-30).



Assembled Instruction: F A

Figure E-30. Control Carriage

Control Carriage and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
SPS CC A CCB	F	III	d

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.

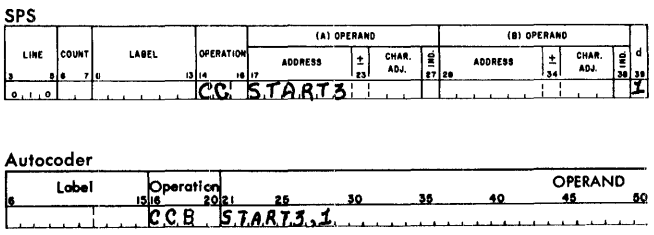
Without indexing: T = 6N ms*
With indexing: T = 7N ms*

* 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.

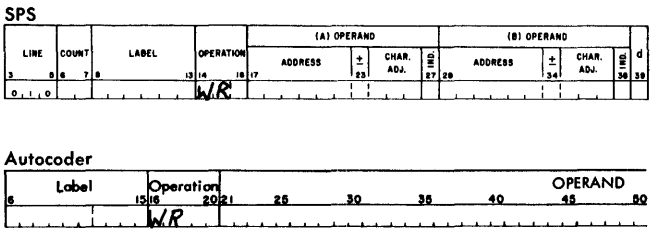
	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	d*bb*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Example. Skip to channel 1 immediately, and branch to START3 (0498) for the next instruction (Figure E-31).



Assembled Instruction: F 498 1

Figure E-31. Control Carriage and Branch



Assembled Instruction: 3

Figure E-32. Write and Read

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 E-43 for timing details.

Word Marks. Word marks are not affected.

Timing. T = 2N ms + 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 E-32.

Write, Read and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-address*
WR	3	III

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.

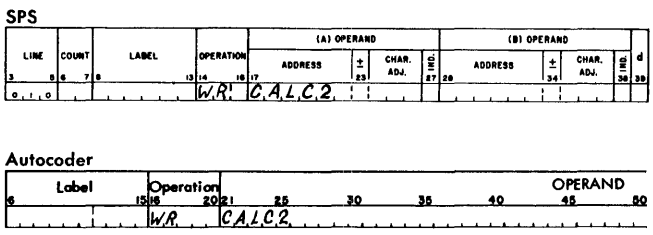
Timing.

Without indexing: T = 5N ms + I/O.
With indexing: T = 6N ms + I/O.
(See *Note* under *Read and Write.*)

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
Without indexing:	NSI	BI	081
With indexing	NSI	BI	NSI

Example. Print a line, read a card, and branch to CALC2 (0759), Figure E-33.



Assembled Instruction: 3 759

Figure E-33. 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 E-44 for timing details.

Word Marks. Word marks are not affected.

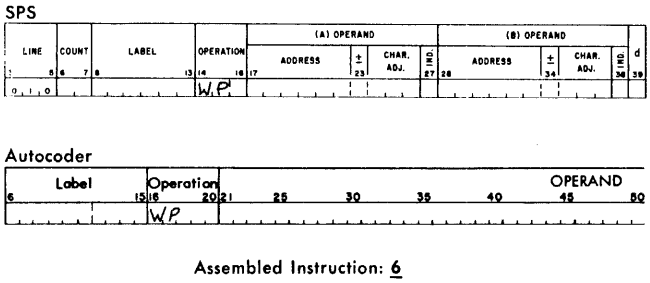
Timing. T = 2N ms + I/O.

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 E-34).



Assembled Instruction: 6

Figure E-34. Write and Punch

Write, Punch and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-address*
WP	6	III

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.

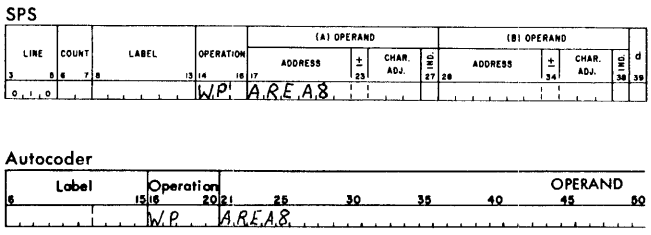
Without indexing: T = 5N ms + I/O.
With indexing: T = 6N ms + I/O.

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.*
Without indexing:	NSI	BI	181
With indexing:	NSI	BI	NSI

Example. Print a line, punch a card, and branch to AREA8 (0895), Figure E-35.



Assembled Instruction: 6 895

Figure E-35. Write, Punch, and Branch

Write, Read, and Punch

Instruction Format.

Mnemonic	*Op Code*
WRP	7

Function. Printing, reading, and punching operations are performed when this instruction is given. The printer takes priority, and the reading and punching cycles start before the end of the actual print operation. See Figure E-45 for timing details.

Word Marks. Word marks are not affected.

Timing. T = 2N ms + I/O.

Address Registers After Operation.

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

(See note under *Read and Punch.*)

Example. Print a line, read a card, and punch a card (Figure E-36).

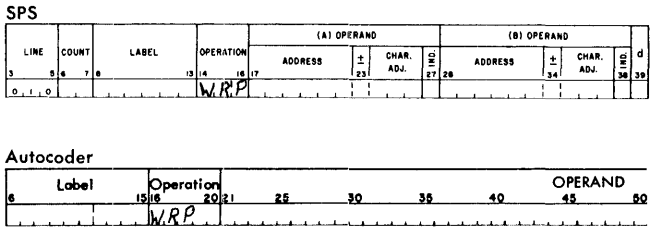


Figure E-36. Write, Read, and Punch

Write, Read, Punch and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-address*
WRP	7	III

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.

Without indexing:	T = 5N ms + I/O.
With indexing:	T = 6N ms + I/O.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
Without indexing:	NSI	BI	181 or 081
With indexing:	NSI	BI	NSI

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. Branch to ROUT4 (0980) after a line is printed, a card is read, and a card is punched (Figure E-37).

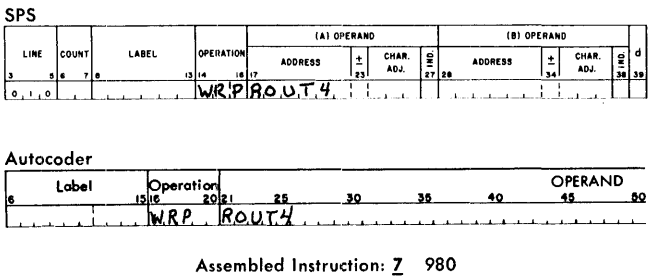


Figure E-37. Write, Read, Punch, and Branch

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 E-38). The print storage area, however, is interlocked for the 84 ms during printing. Form movement for single spacing is overlapped during the last 20 ms of the cycle. If additional form-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 E-39 shows the effective printing speeds under various processing and form-movement considerations.

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 form-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.

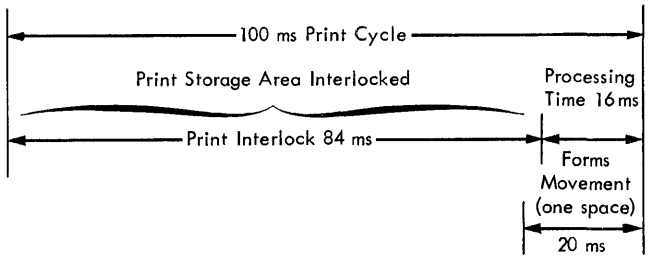


Figure E-38. Print Operation Timing Chart

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 E-39. Effective Printing Speeds (1403)

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

Figure E-40 shows form-movement timing requirements for immediate-skip instructions.

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 E-40. Form Movement Time

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 E-41) allows 53.1 ms of processing time because the print-storage feature is required on a 1460 system with a 1403 Model 3 attached.

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

Figure E-42 shows the effective printing speeds under various form-movement considerations.

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

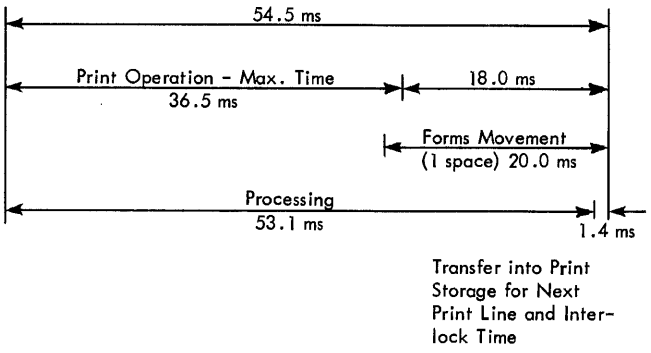


Figure E-41. IBM 1403, Model 3, Print Operation Timing Chart

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 E-42. IBM 1403, Model 3, Effective Printing Speeds

Some program instructions cause form movement to start immediately. If the printer is printing when an immediate form-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 E-40 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 E-43 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. Form 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.

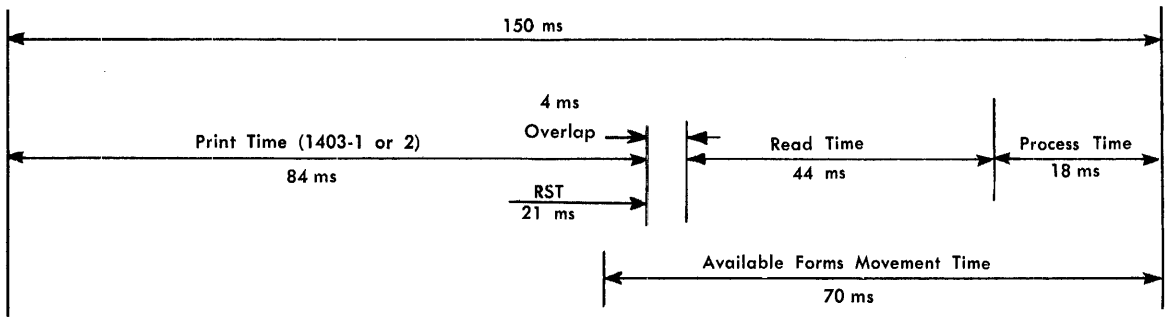


Figure E-43. Write and Read Operation Timing Chart

Installing the early-card-read and print-storage special features increases the cards-and-lines-per-minute rate. The actual rate depends 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 E-44 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 (operation 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 E-45 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.

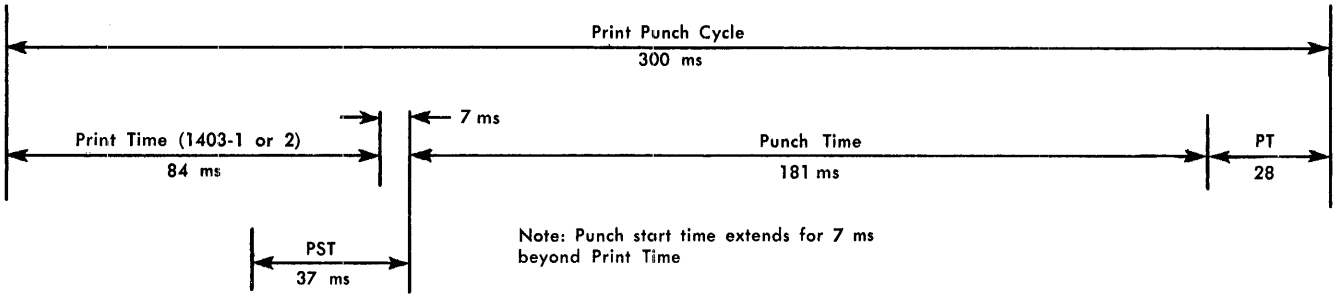


Figure E-44. Write and Punch Operation Timing Chart

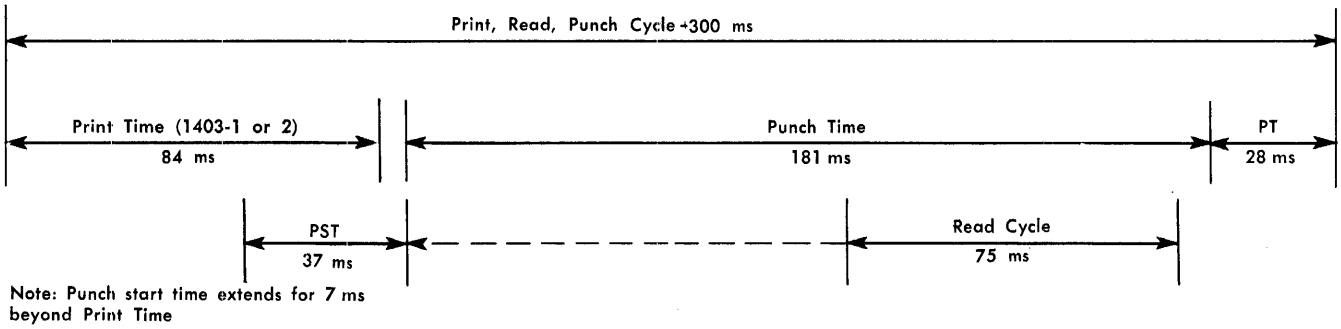


Figure E-45. Write, Read, and Punch Operation Timing Chart

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.)

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 LINE instruction on printer 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 operative only on printer 1.

Character Code Chart (in Collating Sequence)

CHARACTER	DEFINITION/ NOTE	PRINT † ARRANGEMENT A	PRINT † ARRANGEMENT H	** CARD CODE	BCD CODE	OPERATION WHEN USED AS OP CODE
BLANK					C	
.		.	.	12-3-8	B A 8 2 1	Halt
□		□	)	12-4-8	C B A 8 4	Clear Word Mark
[				12-5-8	B A 8 4 1	
<	Less Than			12-6-8	B A 8 4 2	
≑	Group Mark / 1			12-7-8	C B A 8 4 2 1	
&		&	+	12	C B A	
$		$	$	11-3-8	C B 8 2 1	
*		*	*	11-4-8	B 8 4	
]				11-5-8	C B 8 4 1	
;				11-6-8	C B 8 4 2	
Δ	Delta (Mode Change)			11-7-8	B 8 4 2 1	
—		—	—	11	B	
/		/	/	0-1	C A 1	Clear Storage
,		,	,	0-3-8	C A 8 2 1	Set Word Mark
%	2	%	(	0-4-8	A 8 4	Divide
Y	Word Separator			0-5-8	C A 8 4 1	
\	Left Oblique			0-6-8	C A 8 4 2	
⫴	Tape Segment Mark			0-7-8	A 8 4 2 1	
♭	3	≑	≑		A	
#	4	#	=	3-8	8 2 1	Modify Address
@	2	@	'	4-8	C 8 4	Multiply
:				5-8	8 4 1	
>	Greater Than			6-8	8 4 2	
√	Tape Mark			7-8	C 8 4 2 1	
?	(Plus Zero)	&	&	12-0	C B A 8 2	Zero and Add
A		A	A	12-1	B A 1	Add
B		B	B	12-2	B A 2	Branch
C		C	C	12-3	C B A 2 1	Compare
D		D	D	12-4	B A 4	Move Numerical (Bits)
E		E	E	12-5	C B A 4 1	Move Characters and Edit
F		F	F	12-6	C B A 4 2	Control Carriage (Printer)
G		G	G	12-7	B A 4 2 1	
H	2	H	H	12-8	B A 8	Store B-Address Register
I		I	I	12-9	C B A 8 1	
!	(Minus Zero)	—	—	11-0	B 8 2	Zero and Subtract
J		J	J	11-1	C B 1	
K		K	K	11-2	C B 2	Select Stacker (Card)
L		L	L	11-3	B 2 1	Load Characters to Word Mark
M		M	M	11-4	C B 4	Move Characters to Word Mark
N		N	N	11-5	B 4 1	No Operation
O		O	O	11-6	B 4 2	
P	2	P	P	11-7	C B 4 2 1	Move Characters to Record or Group Mark
Q	2	Q	Q	11-8	C B 8	Store A-Address Register
R		R	R	11-9	B 8 1	
≑	Record Mark	≑	≑	0-2-8	A 8 2	
S		S	S	0-2	C A 2	Subtract
T		T	T	0-3	A 2 1	
U		U	U	0-4	C A 4	Control Unit (Tape)
V		V	V	0-5	A 4 1	Branch if Word Mark and/or Zone
W	2	W	W	0-6	A 4 2	Branch if Bit Equal
X	2	X	X	0-7	C A 4 2 1	Move and Insert Zeros
Y		Y	Y	0-8	C A 8	Move Zone (Bits)
Z		Z	Z	0-9	A 8 1	Move Characters and Suppress Zeros
0		0	0	0	C 8 2	
1		1	1	1	1	Read a Card
2		2	2	2	2	Write a Line
3		3	3	3	C 2 1	Write and Read
4		4	4	4	4	Punch a Card
5		5	5	5	C 4 1	Read and Punch
6		6	6	6	C 4 2	Write and Punch
7		7	7	7	4 2 1	Write, Read, and Punch
8	2	8	8	8	8	Start Read Feed
9	2	9	9	9	C 8 1	Start Punch Feed

† Print arrangement H is required for both FORTRAN and COBOL Programming Systems. Otherwise, arrangement A is recommended. Other printing arrangements (B-K) are available.

** The 8-9 portion of 402/403 MLP codes does not cause a validity error, but is not stored and cannot be punched.

1 If specified, this code can be made compatible with 705 Group Mark Code (12-5-8).

2 This character can be used as an Op code if the associated special feature is installed.

3 The A-bit coding for this character (Substitute Blank) must be program-generated in the 1401 (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.

4 This character can be used as an Op code in systems with more than 4,000 positions of core storage.

Index of Instructions

Legend:

AAA – A-field operand (any valid address).

BBB – B-field operand (any valid address).

d – Required instruction modifier.

III – Branch address if test is successful.

n – Unit number.

Type – Applicable I/O device or system feature.

x00 – Any valid address ending in 00.

See associated text for other symbols.

If no mnemonic operation code is listed, code the actual machine-language op code in column 19 (with the d-modifier character in 20, if required) for autocoder, or in column 16 (with the d-modifier in column 39, if required) for SPS.

Instruction	Type	Mnemonics SPS	Mnemonics Autocoder	Format				Page
Add (One Field)		A	A	A	AAA			B-4
Add (Two Fields)		A	A	A	AAA	BBB		B-3
Backspace Tape	1012	SS	SS	K	A			F-17
Backspace Tape and Branch	1012	SS	SSB	K	III	A		F-17
Backspace Tape Record	Tape	CU	BSP	U	%Un	B		F-3
Branch (Unconditional)		B	B	B	III			B-9
Branch if Auto Select	1231	B	BIN	B	III	1		H-56
Branch if Bit Equal	Bit	BBE	BBE	W	III	BBB	d	I-7
Branch if Character Equal		B	BCE	B	III	BBB	d	B-10
Branch if Early Warning	1448	B	BIN	B	III	<		H-53
Branch if Empty Hopper	1231	B	BIN	B	III	4		H-56
Branch if End of Block	1448	B	BIN	B	III	>		H-52
Branch if End of Reel	Tape	B	BEF	B	III	K		F-6
Branch if Error	1285	B	BIN	B	III	1		H-63
Branch if Reader Transporting	1285	B	BIN	B	III	3		H-63
Branch if End of Line	1285	B	BIN	B	III	2		H-63
Branch if Marked Line	1285	B	BIN	B	III	4		H-63
Branch if Header Information	1285	B	BIN	B	III	5		H-63
Branch if Ready to Read a Line	1285	B	BIN	B	III	6		H-63
Branch if Reader Ready	1285	B	BIN	B	III	7		H-63
Branch if End of File	1285	B	BIN	B	III	8		H-63
Branch if Full Buffer	1231	B	BIN	B	III	2		H-56
Branch if Low	HLE	B	BL	B	III	T		I-23
Branch if High	HLE	B	BH	B	III	U		I-23
Branch if Equal	HLE	B	BE	B	III	S		I-23
Branch if I/O Channel-Busy Indicator On	1419	B	BIN	B	III	J		H-45
Branch if in Backspace Operation	1012	B	BIN	B	III	1		F-18
Branch if Indicator On		B	BIN	B	III	d		B-9
Branch if Indicator On	DDC	B	BIN	B	III	d		I-14
Branch if Indicator On	Ovlp	B	BIN	B	III	d		I-39
Branch if Indicator On	SS	B	BIN	B	III	d		I-48
Branch if Indicator On	1009	B	BIN	B	III	d		I-66
Branch if Indicator On	1009	B	BIN	B	III	d		H-5
Branch if Indicator On	1285	B	BIN	B	III	d		H-62
Branch if Indicator On	1301	B	BIN	B	III	d		G-37
Branch if Indicator On	1311	B	BIN	B	III	d		G-22
Branch if Indicator On	1404	B	BIN	B	III	0		H-9
Branch if Indicator On	1405	B	BIN	B	III	d		G-5
Branch if Indicator On	1407	B	BIN	B	III	d		H-12
Branch if Indicator On	1418	B	BIN	B	III	d		H-14
Branch if Indicator On	1428	B	BIN	B	III	d		H-18
Branch if Indicator On	7740	B	BIN	B	III	d		H-67
Branch if Input-Output Indicator On	1011	B	BIN	B	III	1		F-15
Branch if Magnetic Character Reader, Account-Number Field Indicator On	1412	B	BIN	B	III	6		H-31
Branch if Magnetic Character Reader, Document-Spacing Check Indicator On	1412	B	BIN	B	III	8		H-32
Branch if Magnetic Character Reader, Process-Control Field Indicator On	1412	B	BIN	B	III	5		H-30
Branch if Magnetic Character Reader, Reader-Not-Ready Indicator On	1412	B	BIN	B	III	2		H-29
Branch if Magnetic Character Reader, Read Check Indicator On	1412	B	BIN	B	III	3		H-29
Branch if Magnetic Character Reader, Transit-Routing Field Indicator On	1412	B	BIN	B	III	7		M-31
Branch if Magnetic Character Reader, Late Read Indicator On	1412	B	BIN	B	III	1		H-28
Branch if Magnetic Character Reader, Amount-Field Indicator On	1412	B	BIN	B	III	4		H-30
Branch if Paper Tape Reader Ready	1011	B	BIN	B	III	2		F-15
Branch if Printer Status	1445	B	BIN	B	III	D		H-64

Instruction	Type	Mnemonics SPS	Mnemonics Autocoder	Format				Page
Branch if Read Error and Overrun Detection	1231	B	BIN	B	III	5		H-57
Branch if Supply Reel Low or Chad Box Full	1012	B	BIN	B	III	6		F-20
Branch if Tape Error	Tape	B	BER	B	III	L		F-6
Branch if Tape Punch Is Not Ready to Read	1012	B	BIN	B	III	4		F-19
Branch if Tape Punch Not Ready to Accept Data	1012	B	BIN	B	III	3		F-19
Branch if Tape Punch Overextended	1012	B	BIN	B	III	5		F-20
Branch if Tape Punch Ready	1012	B	BIN	B	III	2		F-18
Branch if Timing Mark Check	1231	B	BIN	B	III	6		H-57
Branch if Word Mark or Zone		BWZ	BWZ	V	III	BBB		B-10
Branch if 1231 Ready to read	1231	B	BIN	B	III	3		H-56
Branch if 1419 Indicator On	1419	B	BIN	B	III	d		H-41
Clear Storage		CS	CS	/	AAA			B-16
Clear Storage and Branch		CS	CS	/	AAA	BBB		B-16
Clear Word Mark (One Address)		CW	CW	□	AAA			B-17
Clear Word Mark (Two Addresses)		CW	CW	□	AAA	BBB		B-17
Compare		C	C	C	AAA	BBB		B-11
Compare	1404	C	C	C	AAA	363		H-9
Control Carriage	1403	CC	CC	F	d			E-13
Control Carriage	1404	CC	CC	F	d			H-8
Carriage Control	1445	SS	SS	K	d			H-64, H-65
Control Carriage	1445	CC	CC	F	d			H-65
Control Carriage and Branch	1403	CC	CCB	F	III	d		E-14
Control Unit	1428	CU	CU	U	%S2	d		H-16
Diagnostic Read	Tape	CU	CU	U	%Un	A		F-5
Diagnostic Read	Tape	CU	CU	U	%Bn	A		F-5
Disable Interrupt	1448	SS	SS	K	<			H-51
Disable Interrupt and Branch	1448	SS	SSB	K	III	<		H-52
Disengage Magnetic Character Reader	1412	CU	CU	U	%S1	D		H-26
Divide	M-D	D	D	%	AAA	BBB		I-25
Enable Interrupt	1448	SS	SS	K	>			H-50
Enable Interrupt and Branch	1448	SS	SSB	K	III	>		H-52
Engage Magnetic Character Reader	1412	CU	CU	U	%S1	E		H-26
Go to Next Line	1285	SS	SS	K	G			H-62
Go to Next Line and Branch	1285	SS	SSB	K	III	G		H-62
Halt		H	H	.				B-16
Halt and Branch		H	H	.	III			B-16
Initialize a Message Transmission (XMIT)	1009	CU	CU	U	%D1	E		I-64
Initialize a Reply from Receiver (RCV)	1009	CU	CU	U	%D1	D		I-65
Line Space	1407	LU	WCPW	L	%T0	BBB	W	H-12
Line Space	1407	MU	WCP	M	%T0	BBB	W	H-12
Load Character from the Receiving 1009	1009	LU	LU	L	%D1	BBB	R	H-4, I-66
Load Character to the Transmitting 1009	1009	LU	LU	L	%D1	BBB	W	H-4, I-65
Load Characters to A-Field Word Mark (One Field)		LCA	LCA	L	AAA			B-15
Load Characters to A-Field Word Mark (Two Fields)		LCA	LCA	L	AAA	BBB		B-15
Load from Magnetic Character Reader	1412	LU	LU	L	%S1	BBB	R	H-27
Load Record	Tran	none	MRCWG	P	AAA	BBB	>	I-56
Mark a Line	1285	SS	SS	K	M			H-62
Mark a Line and Branch	1285	SS	SSB	K	III	M		H-62
Modify Address (One Address)	1406	MA	MA	#	AAA			C-2
Modify Address (Two Addresses)	1406	MA	MA	#	AAA	BBB		C-1

Instruction	Type	Mnemonics SPS	Mnemonics Autocoder	Format				Page
Move	1231	MU	MU	M	%S3	BBB	R	H-57
Move and Binary Code	Tape	MCW	MBC	M	AAA	BBB	B	I-9
Move and Binary Decode	Tape	MCW	MBD	M	AAA	BBB	A	I-8
Move and Insert Zeros	Tape	MIZ	MIZ	X	AAA	BBB		I-10
Move Character from the Receiving 1009	1009	MU	LU	M	%D1	BBB	R	H-4, I-65
Move Character to the Transmitting 1009	1009	MU	LU	M	%D1	BBB	W	H-4, I-65
Move Characters and Edit		MCE	MCE	E	AAA	BBB		B-20
Move Characters and Suppress Zeros		MCS	MCS	Z	AAA	BBB		B-13
Move Characters to A or B Word Mark (One Field)		MCW	MLC	M	AAA			B-13
Move Characters to A or B Word Mark (Two Fields)		MCW	MLC	M	AAA	BBB		B-12
Move Characters to Record Mark or Group Mark with Word Mark		MCM	MRCM	P	AAA	BBB		I-4
Move from Magnetic Character Reader	1412	MU	MU	M	%S1	BBB	R	H-27
Move Numeric		MN	MLNS	D	AAA	BBB		B-14
Move Zone		MZ	MLZS	Y	AAA	BBB		B-14
Multiply	M-D	M	M	@	AAA	BBB		I-24
No Operation		NOP	NOP	N				B-18
Operate in Load Mode	1009	SS	SS	K	L			I-67
Overlap Off	Ovlp	SS	SS	K	.			I-35
Overlap Off and Branch	Ovlp	SS	SSB	K	III	.		I-36
Overlap On	Ovlp	SS	SS	K	$			I-35
Overlap On and Branch	Ovlp	SS	SSB	K	III	$		I-35
Printer Selection	1445	CU	none	U	/			H-66
Printer Selection	1460	CU	none	U	1			E-20
Printer Selection	1460	CU	none	U	2			E-20
Printer Selection	1460	CU	none	U	3			E-20
Punch Card	R/P	P	P	4				E-3
Punch Card and Branch	R/P	P	P	4	III			E-4
Punch Card in Overlap Mode	Ovlp	P	P	4				I-39
Punch Column Binary		P	PCB	4	C			I-7
Punch Column Binary and Branch		P	PCB	4	III	C		I-7
Punch Card in Overlap Mode and Branch	Ovlp	P	P	4	III			I-39
Read from 1419 in Move Mode (Overlapped)	1419	MU	MU	M	@S1	BBB	R	H-40
Read-Punch Feed	PFR	P	RF	4	R			I-59
Read and Punch	R/P	RP	RP	5				E-5
Read and Write	7740	LU	LU	L	%A1	BBB	W	H-69
Read and Write	7740	LU	LU	L	%A1	BBB	R	H-69
Read and Write	7740	MU	MU	M	%A1	BBB	R	H-69
Read and Write	7740	MU	MU	M	%A1	BBB	W	H-69
Read Binary Tape	Tape	MU	RTB	M	%Bn	BBB	R	I-9
Read Card	R/P	R	R	1				E-2
Read Card and Branch	R/P	R	R	1	III			E-3
Read Card from 1404 Printer	1404	R	R	1	0			H-7
Read Card in Overlap Mode	Ovlp	R	R	1				I-38
Read Card in Overlap Mode and Branch	Ovlp	R	R	1	III			I-38
Read Column Binary		R	RCB	1	C			I-5
Read Column Binary and Branch		R	RCB	1	III	C		I-6
Read Compressed Tape	Tape	MU	MU	M	%Cn	BBB	R	I-10
Read Data	DDC	MU	MU	M	%H1	BBB	R	I-15
Read Data with Word Marks	DDC	LU	LU	L	%H1	BBB	R	I-15
Read Disk Full-Track	1405	MU	RDT	M	%F2	BBB	R	G-2
Read Disk Full-Track with Word Marks	1405	LU	RDTW	L	%F2	BBB	R	G-3
Read Disk Sector	7770	MU	RD	M	%F1	BBB	R	H-70
Read Disk Sector	7770	LU	RDW	L	%F1	BBB	R	H-70
Read Disk Sector-Count Overlay	1301	MU	RDCO	M	%F5	BBB	R	G-31

Instruction	Type	Mnemonics SPS	Mnemonics Autocoder	Format				Page
Read Disk Sector-Count Overlay with Word Marks	1301	LU	RDCOW	L	%F5	BBB	R	G-31
Read Disk Sector(s)	1311	MU	RD	M	%F1	BBB	R	G-14
Read Disk Sector(s)	1301	MU	RD	M	%F1	BBB	R	G-30
Read Disk Sector(s) with Word Marks	1311	LU	RDW	L	%F1	BBB	R	G-16
Read Disk Sector(s) with Word Marks	1301	LU	RDW	L	%F1	BBB	R	G-31
Read Disk Single-Record	1405	MU	RD	M	%F1	BBB	R	G-2
Read Disk Single-Record with Word Marks	1405	LU	RDW	L	%F1	BBB	R	G-3
Read Disk Track Sectors with Addresses	1301	MU	RDT	M	%F6	BBB	R	G-35
Read Disk Track Sectors with Addresses with Word Marks	1301	LU	RDTW	L	%F6	BBB	R	G-35
Read Disk Track Sectors with Addresses	1311	LU	RDTW	L	%F6	BBB	R	G-21
Read Disk Track Sectors with Addresses	1311	MU	RDT	M	%F6	BBB	R	G-21
Read Disk with Sector-Count Overlay with Word Marks	1311	LU	RDCOW	L	%F5	BBB	R	G-16
Read Disk with Sector-Count Overlay	1311	MU	RDCO	M	%F5	BBB	R	G-16
Read Disk-Track Record with Word Marks	TRRC	LU	RDTRW	L	%F2	BBB	R	I-49
Read Disk-Track Record	TRRC	MU	RDTR	L	%F2	BBB	R	I-49
Read Disk-Track Record with Address	TRRC	MU	RDTA	M	%F@	BBB	R	I-50
Read Disk-Track Record with Address with Word Marks	TRRC	LU	RDTAW	L	%F@	BBB	R	I-50
Read from Console Printer	1407	MU	RCP	M	%T0	BBB	R	H-10
Read from Console Printer with Word Marks	1407	LU	RCPW	L	%T0	BBB	R	H-10
Read from Paper Tape	1011	MU	MU	M	%P1	BBB	R	F-14
Read from Paper Tape with Word Marks	1011	LU	LU	L	%P1	BBB	R	F-15
Read from 1419 in Load Mode	1419	LU	LU	L	%S1	BBB	R	H-39
Read from 1419 in Load Mode (Overlapped)	1419	LU	LU	L	@S1	BBB	R	H-39
Read from 1419 in Move Mode	1419	MU	MU	M	%S1	BBB	R	H-40
Read from 1419 in Move Mode (Overlapped)	1419	MU	MU	M	@S1	BBB	R	H-40
Read from 1447 Console	1447	LU	RCPW	L	%T0	BBB	R	D-2
Read from 1447 Console	1447	MU	RCP	M	%T0	BBB	R	D-2
Read in Move Mode	1285	MU	MU	M	%V1	BBB	R	H-61
Read in Move Mode	1418	MU	MU	M	%S2	BBB	R	H-13
Read in Move Mode	1428	MU	MU	M	%S2	BBB	R	H-17
Read Request	DDC	SS	SS	K	C			I-13
Read Station Select	1418	SS	SS	K	D			H-14
Read Station Select	1428	SS	SS	K	D			H-18
Read Tape	Tape	MU	RT	M	%Un	BBB	R	F-2
Read Tape with Word Marks	Tape	LU	RTW	L	%Un	BBB	R	F-2
Read Tape without Word Marks in Overlap Mode	Ovlp	MU	MU	M	@Un	BBB	R	I-37
Read Tape with Word Marks in Overlap Mode	Ovlp	LU	LU	L	@Un	BBB	R	I-37
Read-Punch Feed and Branch	PFR	P	RF	4	III	R		I-60
Read-Punch Feed and Write	PFR	WP	WRF	6	R			I-60
Read-Punch Feed, Write and Branch	PFR	WP	WRF	6	III	R		I-61
Read, Punch and Branch	R/P	RP	RP	5	III			E-5
Reading Mode Determination	1428	SS	SS	K	D			H-16
Reset	DDC	SS	SS	K	E			I-13
Reset Overlap	Ovlp	SS	SS	K	□			I-36
Reset Overlap	1419	SS	SS	K	□			H-45
Reset Overlap and Branch	Ovlp	SS	SSB	K	III	□		I-36
Reset Overlap and Branch	1419	SS	SSB	K	III	□		H-46
Rewind Tape	Tape	CU	RWD	U	%Un	R		F-5
Rewind Tape and Unload	Tape	CU	RWU	U	%Un	U		F-5
Scan	1448	none	none	O	AAA			H-49
Scan Disk Low, Equal	Scan	MU	SDL	M	%F7	BBB	W	I-47
Scan Disk Equal	Scan	MU	SDE	M	%F8	BBB	W	I-47
Scan Disk High, Equal	Scan	MU	SDH	M	%F9	BBB	W	I-47
Scan Disk Low, Equal with Word Marks	Scan	LU	SDLW	L	%F7	BBB	W	I-47
Scan Disk Equal with Word Marks	Scan	LU	SDEW	L	%F8	BBB	W	I-47

Instruction	Type	Mnemonics SPS	Mnemonics Autocoder	Format				Page
Scan Disk High, Equal with Word Marks	Scan	LU	SDHW	L	%F9	BBB	W	I-47
Seek Disk	1301	MU	SD	M	%F0	BBB	R	G-30
Seek Disk	1301	LU	LU	L	%F0	BBB	R	G-30
Seek Disk	1311	LU	LU	L	%F0	BBB	R	G-12
Seek Disk	1311	MU	SD	M	%F0	BBB	R	G-12
Seek Disk	1405	LU	LU	L	%F0	BBB	R	G-2
Seek Disk	1405	MU	SD	M	%F0	BBB	R	G-2
Seek Disk	7770	LU	LU	L	%F0	BBB	R	H-70
Seek Disk	7770	MU	SD	M	%F0	BBB	R	H-70
Select Stacker	R/P	SS	SS	K	d			E-4
Select Stacker	1231	SS	SS	K	A			H-57
Select Stacker	1418	SS	SS	K	d			H-14
Select Stacker	1419	SS	SS	K	d			H-44
Select Stacker	1428	SS	SS	K	d			H-17
Select Stacker and Branch	R/P	SS	SSB	K	III	d		E-5
Select Stacker and Branch	1419	SS	SSB	K	III	d		H-44
Select Stacker – Magnetic Character Reader	1412	SS	SS	K	d			H-28
Send EOF (End-of-File)	1009	SS	SS	K	F			I-67
Set Direction to Receive	1009	SS	SS	K	D			I-67
Set Direction to Transmit	1009	SS	SS	K	E			I-67
Set Ready to Receive	1009	CU	CU	U	%D1	D		H-3
Set Word Mark (One Address)		SW	SW	,	AAA			B-17
Set Word Mark (Two Addresses)		SW	SW	,	AAA	BBB		B-17
Signal Control	7740	SS	SS	K	d			H-67
Skip and Blank Tape	Tape	CU	SKP	U	%Un	E		F-4
Space Suppress (1401)	1445	SS	SS	K	B			H-64
Start Punch Feed	R-PR	SPF	SPF	9				I-46
Start Read Feed	R-PR	SRF	SRF	8				I-45
Start Transmission	1009	CU	CU	%	%D1	E		H-3
Store A-Address Register	Store	SAR	SAR	Q	AAA			I-2
Store B-Address Register (One Field)	Store	SBR	SBR	H	AAA			I-3
Store B-Address Register (Two Fields)	Store	SBR	SBR	H	AAA	BBB		I-3
Subtract (One Field)		S	S	S	AAA			B-7
Subtract (Two Fields)		S	S	S	AAA	BBB		B-6
Suppress 3-Second Alarm	1009	SS	SS	K	A			H-6
Suppress 3-Second Alarm	1009	SS	SS	K	A			I-67
Tape Punch Readback Check	1012	MU	MU	M	%P1	BBB	R	F-17
Translate with Word Marks	Tran	none	TRW	T	AAA	x00	>	I-53
Translate without Word Marks	Tran	none	TR	T	AAA	x00		I-54
Unit Control	1419	CU	CU	U	%S1	d		H-38
Write and Punch	R/P	WP	WP	6				E-16
Write and Read	R/P	WR	WR	3				E-15
Write and Read on 1404 Printer	1404	WR	WR	3	0			H-8
Write Binary Tape	Tape	MU	WTB	M	%Bn	AAA	W	I-9
Write Data	DDC	MU	MU	M	%H1	BBB	W	I-16
Write Data with Word Marks	DDC	LU	LU	M	%H1	BBB	W	I-16
Write Disk Check	1301	MU	WDC	M	%F3	BBB	W	G-35
Write Disk Check with Word Marks	1301	LU	WDCW	M	%F3	BBB	W	G-35
Write Disk Check with Word Marks	1311	LU	WDCW	L	%F3	BBB	W	G-20
Write Disk Check	1311	MU	WDC	M	%F3	BBB	W	G-20
Write Disk Check with Word Marks	1405	LU	WDCW	L	%F3	BBB	W	G-4
Write Disk Check	1405	MU	WDC	M	%F3	BBB	W	G-4
Write Disk, Full-Track	1405	MU	WDT	M	%F2	BBB	W	G-3
Write Disk, Full-Track with Word Marks	1405	LU	WDTW	L	%F2	BBB	W	G-4
Write Disk Sector	7770	MU	WD	M	%F1	BBB	W	H-70
Write Disk Sector with Word Marks	7770	LU	WDW	L	%F1	BBB	W	H-70
Write Disk Sector-Count Overlay	1301	MU	MDCO	M	%F5	BBB	W	G-34

Instruction	Type	Mnemonics SPS	Mnemonics Autocoder	Format				Page
Write Disk Sector-Count Overlay with Word Marks	1301	LU	WDCOW	L	%F5	BBB	W	G-34
Write Disk Sector(s)	1301	MU	WD	M	%F1	BBB	W	G-33
Write Disk Sector(s)	1311	MU	WD	M	%F1	BBB	W	G-18
Write Disk Sector(s) with Word Marks	1311	LU	WDW	L	%F1	BBB	W	G-19
Write Disk Sector(s) with Word Marks	1301	LU	WDW	L	%F1	BBB	W	G-34
Write Disk Single-Record	1405	MU	WD	M	%F1	BBB	W	G-3
Write Disk Single-Record with Word Marks	1405	LU	WDW	L	%F1	BBB	W	G-4
Write Disk Track Sectors with Addresses	1301	MU	WDT	M	%F6	BBB	W	G-36
Write Disk Track Sectors with Addresses with Word Marks	1301	LU	WDTW	L	%F6	BBB	W	G-36
Write Disk with Sector-Count Overlay with Word Marks	1311	LU	WDCOW	L	%F5	BBB	W	G-19
Write Disk with Sector-Count Overlay	1311	MU	WDCO	M	%F5	BBB	W	G-19
Write Disk-Track Record with Word Marks	TRRC	LU	WDTRW	L	%F2	BBB	W	I-51
Write Disk-Track Record	TRRC	MU	WDTR	M	%F2	BBB	W	I-51
Write Disk-Track Record with Address with Word Marks	TRRC	LU	WDTAW	L	%F@	BBB	W	I-52
Write Disk-Track Record with Address	TRRC	MU	WDTA	M	%F@	BBB	W	I-52
Write Disk Track Sectors with Addresses	1311	MU	WDT	M	%F6	BBB	W	G-21
Write Disk Track Sectors with Addresses with Word Marks	1311	LU	WDTW	L	%F6	BBB	W	G-21
Write Line	1403	W	W	2				E-10
Write Line	1404	W	W	2				H-8
Write Line (1401)	1445	MU	W	M	%Y2	BBB	W	H-64
Write Line (1460)	1445	MU	W	M	%Y1	BBB	W	H-65
Write Line and Branch	1403	W	W	2	III			E-11
Write Line and Suppress Space (1460)	1445	MU	WS	M	%Y1	BBB	S	H-65
Write on Console Printer	1407	MU	WCP	M	%T0	BBB	W	H-11
Write on Console Printer with Word Marks	1407	LU	WCPW	L	%T0	BBB	W	H-11
Write on Tape Punch	1012	MU	MU	M	%P1	BBB	W	F-16
Write on 1447 Console	1447	MU	WCP	M	%T0	BBB	W	D-3
Write on 1447 Console in Load Mode	1447	LU	WCPW	L	%T0	BBB	W	D-3
Write, Punch and Branch	R/P	WP	WP	6	III			E-16
Write, Read, and Punch	R/P	WRP	WRP	7				E-16
Write, Read, Punch and Branch	R/P	WRP	WRP	7	III			E-17
Write, Read and Branch	R/P	WR	WR	3	III			E-15
Write Request	DDC	SS	SS	K	D			I-13
Write Tape	Tape	MU	WT	M	%Un	BBB	W	F-3
Write Tape Mark	Tape	CU	WTM	U	%Un	M		F-4
Write Tape with Word Marks	Tape	LU	WTW	L	%Un	BBB	W	F-3
Write Tape without Word Marks in Overlap Mode	Olvp	MU	MU	M	@Un	BBB	W	I-37
Write Tape with Word Marks in Overlap Mode	Olvp	LU	LU	L	@Un	BBB	W	I-37
Write Word Marks	1403	W	WM	2	□			E-11
Write Word Marks and Branch	1403	W	WM	2	III	□		E-11
Zero and Add (One Field)		ZA	ZA	?	AAA			B-6
Zero and Add (Two Fields)		ZA	ZA	?	AAA	BBB		B-5
Zero and Subtract (One Field)		ZS	ZS	!	AAA			B-8
Zero and Subtract (Two Fields)		ZS	ZS	!	AAA	BBB		B-7

Index

( ) = system [ ] = function, feature, or machine

IBM **Technical Newsletter**

File No. 1401-01

Re: Order No. A24-3067-2

This Newsletter No. GN24-0451

Date: March 26, 1970

Previous Newsletter Nos. None

SYSTEM OPERATION REFERENCE MANUAL
IBM 1401 DATA PROCESSING SYSTEM
IBM 1460 DATA PROCESSING SYSTEM

This Technical Newsletter provides replacement pages for the subject publication. Pages to be inserted and/or removed are:

A-5, A-6
A-9, A-10
B-5, B-6
B-9, B-10
B-11, B-12
B-19, B-20
E-5, E-6
E-13, E-14
N-3, N-4

A change to the text is indicated by a vertical line to the left of the change.

Summary of Amendments
Correction to Branch If Channel 9 and Branch If Channel 12.
Clarification of text and expansion of the Instruction Index.

Note: Please file this cover letter at the back of the manual to provide a record of changes.

IBM Corp., Product Publications Dept., Endicott, N. Y. 13760

PRINTED IN U.S.A.

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 do not need to be followed by a word mark. Note, however, that the 4-character BRANCH UNCONDITIONAL if not followed by a word mark must be followed by a blank to operate correctly. The others end automatically after the seventh character.

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 card 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

Op Code	*A- or I-address*	*B-address*	*d-character*
X	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.

Instruction Descriptions

Specific instructions have been described in a standard format:

Title. This is the description of the instruction.

Instruction Length. The length of an instruction can be either 1, 2, 4, 5, 7, or 8 characters. It cannot be either 3 or 6 characters long. Characters beyond the usable limit of eight do not affect the operation. Addressing advances (mod + 1) until the next word mark is sensed before the instruction is executed.

Most instructions must have a word mark following the instruction in core storage. This word mark is normally associated with the core-storage location immediately following the instruction itself.

Figure A-6 shows examples of the combinations possible in variable-length instructions.

Instruction Format. This is the format of the particular instruction described. The mnemonics operation code used for Autocoder and SPS are given.

Function. This is the function of the instruction.

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

Timing. When the instruction-execution timing is always a constant, the actual time in milliseconds is given. When the instruction-execution time can vary because of field length or chaining, the formula is given. Figure A-7 is the key to the abbreviations used in the formulas.

Notes. These are special notations or additional information pertaining to the operation.

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

Chaining. This assists the programmer in determining whether instruction-chaining can be used effectively. In some cases, chaining proves useful even though it would not ordinarily be used. For example, another instruction can be chained to the MOVE CHARACTERS AND EDIT instruction if the programmer can use the contents of the address registers to advantage. When considering the use of chaining, be certain that the contents of the address registers are valid for all conditions relating to the instructions involved. (Refer to the specific instruction section.)

Example. A practical application of the instruction is described with the label of a typical actual machine address (in parentheses).

These examples for the instructions are representative, and are intended as exhibits of typical core-storage assignments, rather than specific, limited examples. Since the assembler program (autocoder or SPS) is usually allowed to establish core-storage addresses, the programmer need not ordinarily be concerned with specific machine-language storage locations, except when a program must be analyzed.

The few inflexible addresses of core-storage locations, such as index registers, are shown in the instructions as exact locations.

Assembled Instruction. This is the actual machine language instruction that is assembled by the processor program from the symbolic entries shown in the example.

When an explicit mnemonic is used, the op code, A-address, and d-modifier (when required) are automatically generated in most cases (refer to the specific operation in question).

Example: The coded instruction to cause information to print on the 1447 might be in the form: WCP (column 16 of the coding sheet), PRTOUT (written in column 21-26). Assume for the purpose of this example that the label PRTOUT actually represents core-storage location 0101. Autocoder would assemble this coded instruction into actual machine

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

Figure A-6. IBM 1401 Instruction Formats

Instruction addressed by high-order position

STORAGE ADDRESS	400	401	402	403	404	405	406	407 (NSI)
INSTRUCTION	A	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.

A-address

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

A-field

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

B-address

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

B-field

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

Figure A-10. Data and Instruction Addressing

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 A-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 whether 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 feature, which is a special feature for the 1401 and 1460 systems.

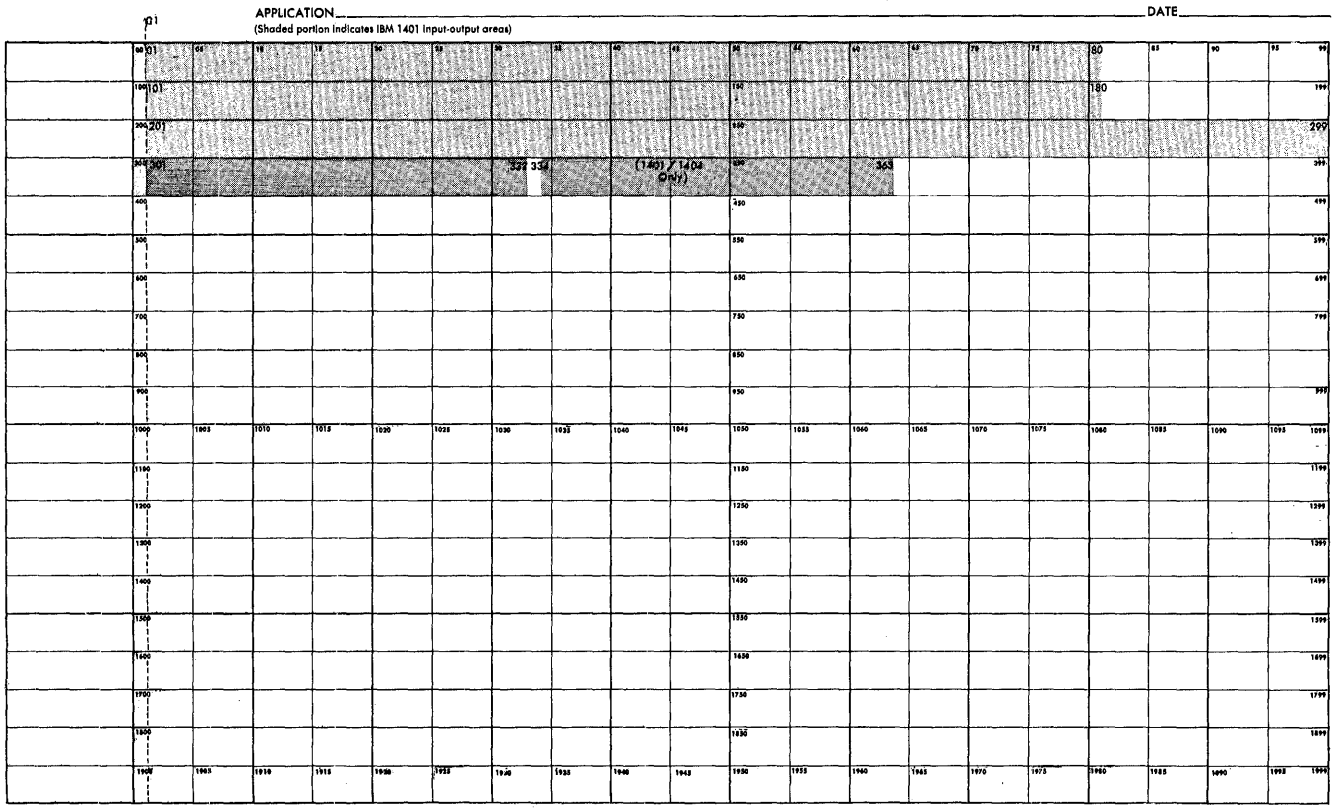


Figure A-11. Core Storage Layout Chart

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 A-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.

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 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)

A + A	= B	or	1 + 1 = 2
A + B	= AB	or	1 + 2 = 3
B + B	= NoANoB	or	2 + 2 = 0
A + AB	= NoANoB	or	1 + 3 = 0
A + NoANoB	= A	or	1 + 0 = 1
B + AB	= A	or	2 + 3 = 1
B + NoANoB	= B	or	2 + 0 = 2
AB + AB	= B	or	3 + 3 = 2

Figure A-12. A-Bit and B-Bit Values

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, 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 (H5B) = complement
879 + H5B = 731 (with arithmetic overflow)
H = BA8
B = 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 15000 to 15999 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 A-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

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	(A) OPERAND ±	(A) OPERAND CHAR. ADJ.	(A) OPERAND IND.	(B) OPERAND ADDRESS	(B) OPERAND ±	(B) OPERAND CHAR. ADJ.	(B) OPERAND IND.	d
			A	CURERN				YTDGRO				

Autocoder

Label	Operation	OPERAND
	A	CURERN,YTDGRO

Assembled Instruction: A 506 708

Figure B-6. Add (Two Fields)

Add (One Field)

Instruction Format.

Mnemonic	*Op Code*	*A-address*
A	A	AAA

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.

Timing.

1401:

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

1460:

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

Address Registers After Operation.

I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
NSI	$A\text{-}L_A$	$A\text{-}L_A$

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Add to itself the data at EXEMPT (0981), Figure B-7.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	(A) OPERAND ±	(A) OPERAND CHAR. ADJ.	(A) OPERAND IND.	(B) OPERAND ADDRESS	(B) OPERAND ±	(B) OPERAND CHAR. ADJ.	(B) OPERAND IND.	d
			A	EXEMPT								

Autocoder

Label	Operation	OPERAND
	A	EXEMPT

Assembled Instruction: A 981

Figure B-7. Add (One Field)

Zero and Add (Two Fields)

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
ZA	?	AAA	BBB

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.

The Op code for this instruction is plus zero but prints as an ampersand. See "Character Code Chart."

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. $T = N\ (L_I + 1 + L_A + L_B)$ 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	$A\text{-}L_W$	$B\text{-}L_B$

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Zero WHTAX area (0796-0802) and add new TAX (0749-0754) to WHTAX (Figure B-8).

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	(A) OPERAND ±	(A) OPERAND CHAR. ADJ.	(A) OPERAND IND.	(B) OPERAND ADDRESS	(B) OPERAND ±	(B) OPERAND CHAR. ADJ.	(B) OPERAND IND.	d
			ZA	TAX				WHTAX				

Autocoder

Label	Operation	OPERAND
	ZA	TAX,WHTAX

Assembled Instruction: ? 754 802

Figure B-8. Zero and Add (Two Fields)

Zero and Add (One Field)

Instruction Format.

Mnemonic	*Op Code*	*A-address*
ZA	?	AAA

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.

The Op code for this instruction is plus zero but prints as an ampersand. See "Character Code Chart."

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

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Strip zone bits from TOTAL (0560) area (Figure B-9).

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	WM	(B) OPERAND ADDRESS	±	CHAR. ADJ.	WM	d
3 5	6 7	8 13	14 16	17 23			27	28 34			38	39
0			ZA	TOTAL								

Autocoder

Label	Operation	OPERAND
6 15	16 20	21 25 30 35 40 45 50
	ZA	TOTAL

Assembled Instruction: ? 560

Figure B-9. Zero and Add (One Field)

Subtract (Two Fields)

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
S	S	AAA	BBB

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 B-2 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:

   $T = .0115 (L_I + 3 + L_A + L_B)$ ms.

   1460:

   $T = .006 (L_I + 1 + L_A + L_B)$ ms.

2. Subtract – recomplement cycle necessary

   1401:

   $T = .0115\ L_I + 3 + L_A + 4L_B)$ ms.

   1460:

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

   $T = .006 (L_I + 1 + L_A + 2L_B)$ 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

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

Example. Subtract CUFICA (0753) from CURGRO (0896), Figure B-10.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	WM	(B) OPERAND ADDRESS	±	CHAR. ADJ.	WM	d
3 5	6 7	8 13	14 16	17 23			27	28 34			38	39
0			S	CUFICA				CURGRO				

Autocoder

Label	Operation	OPERAND
6 15	16 20	21 25 30 35 40 45 50
	S	CUFICA,CURGRO

Assembled Instruction: S 753 896

Figure B-10. Subtract (Two Fields)

Logic Instructions

Branch (Unconditional)

Instruction Format.

Mnemonic	*Op Code*	*I-address*
B	B	III

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 Marks. A blank or a word mark must be associated with the core-storage position of the next address.

Timing.

Branch without indexing: T = 5N ms.
Branch with indexing: T = 6N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg*	*B-Add. Reg.*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Example. Unconditionally branch to AGAIN (3498), Figure B-14.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	(A) OPERAND ±	(A) OPERAND CHAR. ADJ.	(A) OPERAND I/O	(B) OPERAND ADDRESS	(B) OPERAND ±	(B) OPERAND CHAR. ADJ.	(B) OPERAND I/O	d
			B	AGAIN								

Autocoder

Label	Operation	OPERAND
	B	AGAIN

Assembled Instruction: B D98

Figure B-14. Branch (Unconditional)

Branch if Indicator On

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
SPS B	B	III	d
A see Figure B-15			

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.
‡ d-modifier character must be coded in the operand portion of the instruction.

Figure B-15. Branch if Indicator On, Mnemonic d-Character, and Conditions

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 B-15 shows characters that are valid for the d-character and for 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, or Branch without indexing: T = 6N ms.
Branch with indexing: T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	dbb
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Example. Test for the last card. If it is the last card, branch to END (0599), Figure B-16.

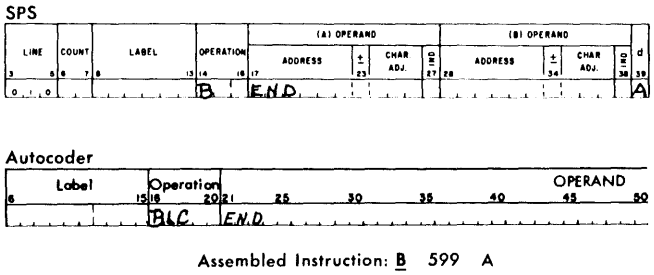


Figure B-16. Branch if Indicator On

Branch if Character Equal

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*B-address*	*d-character*
SPS B	B	III	BBB	d
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, or Branch without indexing: $T = N (L_I + 2)$ ms.
Branch with indexing: $T = N (L_I + 3)$ ms.

Note. The BRANCH IF CHARACTER EQUAL instruction does not reset any internal system indicators, as does the BRANCH IF READER ERROR instruction, for example.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	B-1
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

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 B-17).

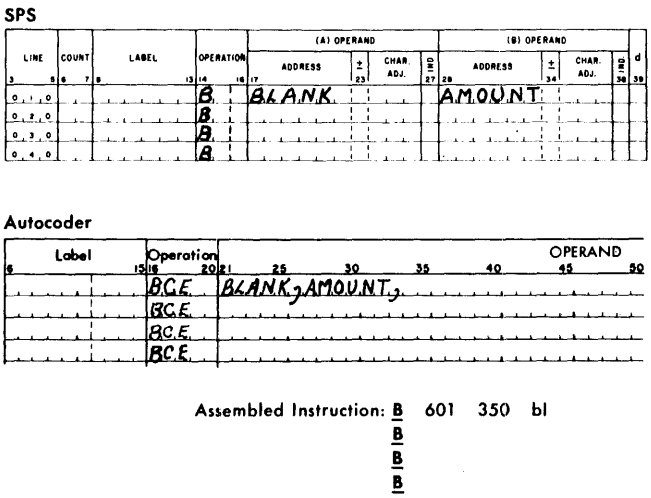


Figure B-17. Branch if Character Equal

Branch if Word Mark and/or Zone

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*B-address*	*d-character*
BWZ	V	III	BBB	d

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

Two explicit autocoder mnemonics are provided that do not require the third (d-modifier character) operand. These are BM for Branch if 11-Zone, and BW for Branch if Word Mark.

Word Marks. See table given in "Function."

Timing.

No Branch or Branch without indexing: $T = N (L_I + 2)$ ms.
Branch with indexing: $T = N (L_I + 3)$ ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	B-1
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code.

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 B-18).

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	(A) OPERAND ±	(A) OPERAND CHAR. ADJ.	(A) OPERAND ID	(B) OPERAND ADDRESS	(B) OPERAND ±	(B) OPERAND CHAR. ADJ.	(B) OPERAND ID	d
			BWZ	NEGRTE				GROAMT				K

Autocoder

Label	Operation	OPERAND
	BM	NEGRTE,GROAMT

Assembled Instruction: V 598 M98 K

Figure B-18. Branch if 11 Zone

Compare

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
C	C	AAA	BBB

The following explicit autocoder branch instructions are provided for use with the Compare instruction. When the specified mnemonic is used, the d-character is not written in the operand field.

Branch if Unequal ($B \neq A$)	BU
Branch if Equal ($B = A$)	BE
Branch if High ($B > A$)	BH
Branch if Low ($B < A$)	BL

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. In this case, the high-compare and the unequal-compare indicators are set on.

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

Note: Both fields must have exactly the same bit configurations, to be equal. For example, 00? (? = 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 · □ [< ≢ & $ *] ; Δ − / , % ⮟ \ ≠ b # @ : > √ ? A through I | 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

Chaining. This instruction can be chained to the preceding operation (if that instruction left usable address-register contents) by supplying only the operation code. When Compare instruction(s) are chained to a preceding Compare instruction, the compare-result indicators are set by the *first* unequal condition encountered in the composite field.

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 B-19).

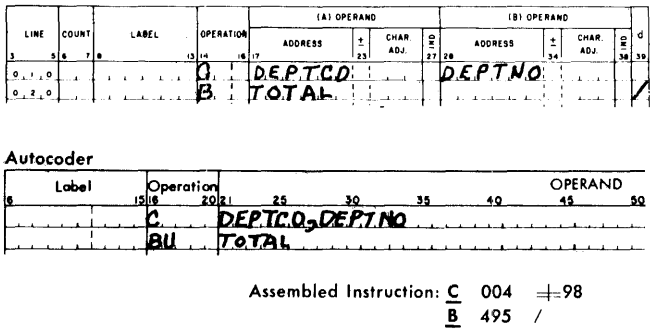


Figure B-19. Compare

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 *A* MLC or MCW	M	AAA	BBB

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

Chaining. See the *Move Characters to A or B Word Mark (One Field)* section.

Example. Move the 5-character field NAMIN (0750) to the 5-character field NAMOUT (0850), Figure B-20.

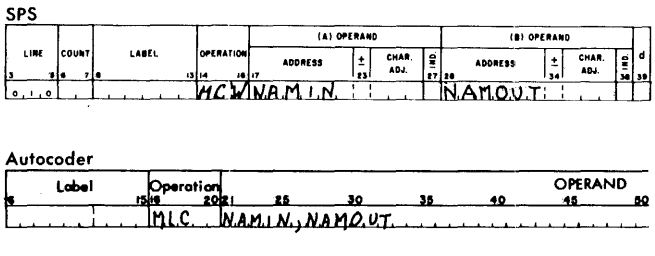


Figure B-20. Move Character to A or B Word Mark (Two Fields)

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. The HALT instruction normally has either one or seven characters. When the seven-character format is used, the A- and B-storage address register contents serve to identify the halt. (The two-, five-, and eight-character formats have similar purposes.)

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 B-34 is a symbolic example of the HALT instruction. Any identifying information (constants, label-defined address constants, etc) can be coded in the A- and B-field operands. In this example, 888 and RETRY (0681) were chosen.

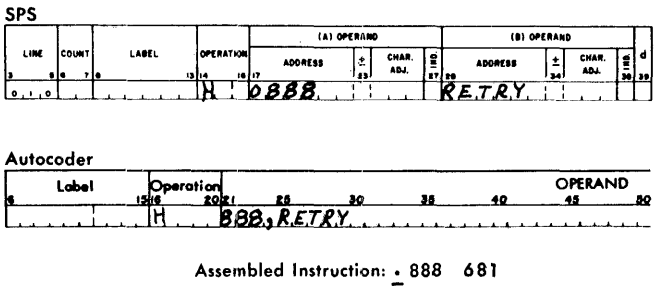


Figure B-34. Halt

Halt and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-Address*
H		III

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

This instruction can serve as a manual switch (without tying up a sense switch). If under certain conditions, the operator elects to proceed to the next sequential instruction rather than execute the branch portion of the HALT, he can press the Start Reset and Start keys.

Word Marks. Word marks are not affected.

Timing.

Without indexing:	T = 5N ms.
With indexing:	T = 6N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
Without indexing:	NSI	BI	blank
With indexing:	NSI	BI	NSI

Example. Stop the system, and branch to START2 (0895) for the next instruction when the start key is pressed (Figure B-35).

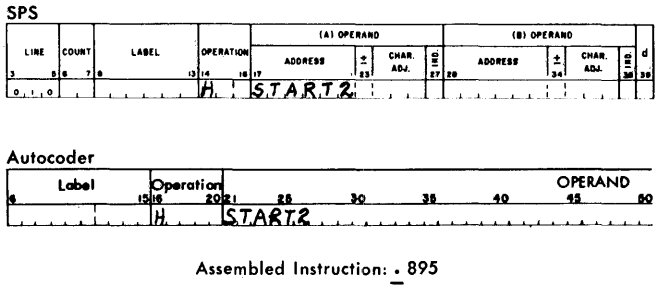


Figure B-35. Halt and Branch

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 B-36).

	OP	A-address	B-address
Edit instruction	E	789	300
		A-field (data)	B-field (control word)
Storage		00257426	$ bbb, bb0.bb & CR &**
			B-field
Result of edit		00257426	$ 2,574.26 **

Figure B-36. 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 unchanged and is not transferred to the B-field.

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 B-37 shows the use of these rules as applied to the data in Figure B-36.

Move Characters and Edit

Instruction Format.

Mnemonic	*Op Code*	*A-address*	*B-address*
MCE	E	AAA	BBB

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 remains unchanged during the operation. It is used to control portions of the operation, but is not transferred to the B-field.

Editing Rules.

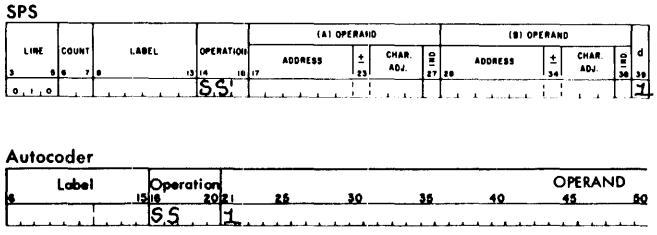
Rule 1. All numerical, alphabetic, and special characters can be used in the control word. However, some of these characters have special meanings:

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. If the edit control word contains a decimal, the edit-control-word zero must be to the left of the decimal. When a positive zero data field is edited (*without* the Expanded Print Edit special feature), the result is .00; when a negative zero field is edited the result is .00 CR (if CR was included in the edit control word). Note that *with* the Expanded Print Edit feature, nothing prints for a positive-zero field, and only the CR prints for a negative-zero field.
. (decimal)	This remains in the edited field where it was placed by loading the edit control word. It is removed during a zero-suppression 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 *Expanded Print Edit* in *Special Features* – Section I).
, (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.
– (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 *Expanded Print Edit* in *Special Features* – Section I).
$ (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 *Expanded Print Edit* in *Special Features* – Section I).

Address Registers After Operation.

I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
NSI	d*bb*	d*bb*

Example. Enter the last card read into pocket 1 (Figure E-10).



Assembled Instruction: K 1

Figure E-10. Select Stacker

Select Stacker and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
SPS SS	K	III	d
A SSB			

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

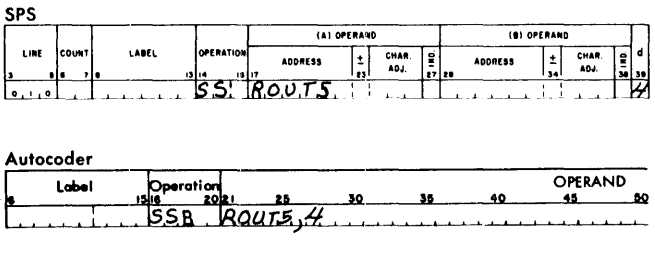
Timing.

Without indexing: T = 6N ms.
With indexing: T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
Without indexing:	NSI	BI	d*bb*
With indexing:	NSI	BI	NSI

Example. Select the last card punched, enter it in pocket 4, and branch to ROUT5 (0950) for the next instruction (Figure E-11).



Assembled Instruction: K 950 4

Figure E-11. Select Stacker and Branch

Read and Punch

Instruction Format.

Mnemonic	*Op Code*
RP	5

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 E-18 for timing details.

Word Marks. Word marks are not affected.

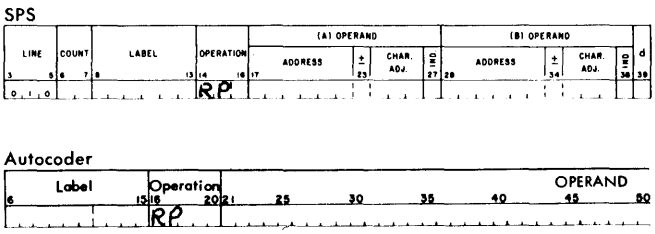
Timing. T = 2N ms + I/O.

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. In the 1401G, the A-register controls the data and the B-register can be at any previous address.

Example. Read a card into the input area, and punch a card with information from the punch area (Figure E-12).



Assembled Instruction: 5

Figure E-12. Read and Punch

Read, Punch and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-address*
RP	5	III

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.

Without indexing: T = 5N ms + I/O.
With indexing: T = 6N ms + I/O.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.**
Without indexing:	NSI	BI	181 or 081
With indexing:	NSI	BI	NSI

*(See note under *Read and Punch.*)

Example. Read, punch, and branch to WORK5 (0596), Figure E-13.

SPS

LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ	(B) OPERAND ADDRESS	±	CHAR ADJ	d
3 5	6 7	8 13	14 16	17 23			27 29	34		38 39
0 1 0			RP	WORK5						

Autocoder

Label	Operation	OPERAND
6 15	16 20	21 25 30 35 40 45 50
	RP	WORK5

Assembled Instruction: 5 596

Figure E-13. Read, Punch, and Branch

Branch if Reader Error

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BIN	B	III	? (plus zero)

Function. If an error occurs during the card-read operation, this indicator is set, and the reader light glows on the console. If the I/O check-stop switch is OFF, this indicator can be tested and reset by the BRANCH-IF INDICATOR ON instruction. If the indicator is ON, the program is interrupted, and a branch to the I-address occurs. If the indicator is not ON, no branch occurs.

Word Marks. Word marks are not affected.

Timing.

No Branch, or Branch without indexing: T = 6N ms
Branch with indexing: T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	?*bb*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Branch if Punch Error

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BIN	B	III	! (minus zero)

Function. Same as for *Reader Error,* except that specifics apply to the punch operation.

Word Marks. Word marks are not affected.

Timing.

No Branch, or Branch without indexing: T = 6N ms
Branch with indexing: T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	!*bb*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Branch if Last Card

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BLC	B	III	A

Function. This instruction causes a branch to the address specified by the I-address, if the last card in the read feed has been read or the last card in the punch feed is ready for punching.

Word Marks. Word marks are not affected.

Timing.

No Branch, or Branch without indexing: T = 6N ms
Branch with indexing: T = 7N ms

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	A*bb*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Branch if Printer Carriage Busy

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BPCB	B	III	R

Function. If the printer carriage is executing a forms-movement operation, this instruction can be used to branch to other instructions until the carriage is again available. The indicator is reset as soon as the printer carriage is no longer busy. Using this instruction allows processing to continue while the printer carriage is busy, thus, in effect, permitting temporary overlapping of processing and printer operations.

NOTE: See Note 2 under *Branch if Printer Busy* section.

Word Marks. Word marks are not affected.

Timing.

No Branch, or Branch without indexing: T = 6N ms
Branch with indexing: T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	R*bb*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Branch if Channel 9

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BC9	B	III	9

Function. If the channel-9 position of the carriage control tape has been sensed, this instruction will cause a branch to the address specified by the I-address. This indicator is reset by the next carriage tape channel sensed.

Word Marks. Word marks are not affected.

Timing.

No branch, or branch without indexing: T = 6N ms.
Branch with indexing: T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch	NSI	BI	9*bb*
Branch (without indexing)	NSI	BI	blank
Branch (with indexing)	NSI	BI	NSI

Branch if Channel 12

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
BCV	B	III	@

Function. If the channel 12 (forms overflow) position of the carriage control tape has been sensed, this instruction will cause a branch to the address specified by the I-address. This indicator is reset by the next carriage tape channel sensed.

Word Marks. Word marks are not affected.

Timing.

No branch, or branch without indexing: T = 6N ms.
Branch with indexing: T = 7N ms.

Address Registers After Operation.

	I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
No Branch:	NSI	BI	@*bb*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Carriage-Tape-Punch Considerations

A channel-1 carriage-tape punch is required for each form at the location of the first line of printing.

On unbuffered printers, a branch on channel 9 or 12 always interlocks the system. The last line of printing is one line *above* the channel-9 or -12 punch in the carriage tape.

On buffered printers, a branch on channel 9 or 12 immediately *before* printing will cause the last line to print one line *above* the channel-9 or -12 punch in the carriage tape. The same is true if the overflow-branch occurs while the printer is busy (currently engaged in the printing of a line). If the overflow (branch if channel 9 or 12) occurs immediately *after* printing, the last printed line of that form will be *on* the line corresponding to the channel-9 or -12 carriage-tape punch.

Control Carriage

Instruction Format.

Mnemonic	*Op Code*	*d-character*
CC	F	d

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

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 E-29) 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 = 3N 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.

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 E-29. d-Characters for Control Carriage

NOTE: Care should be taken when punching channels 9 and 12 of the carriage control tape 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 is taken after printing. When a skip after print or space after print instruction is used, the automatic space is not taken.

When the carriage tape is positioned at a channel-1 punch, a skip to channel 1 immediate instruction (F1) does not move the paper.

Address Registers After Operation.

I-Add. Reg.	*A-Add. Reg.*	*B-Add. Reg.*
NSI	d*bb*	d*bb*

Example. Skip to channel 1 after print (Figure E-30).

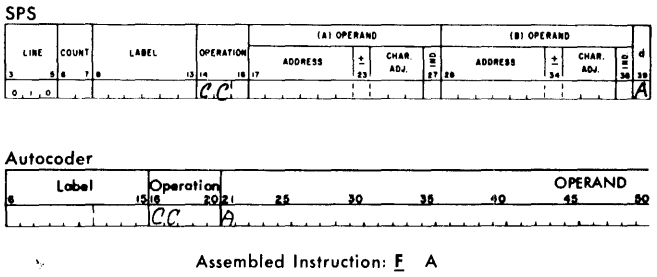


Figure E-30. Control Carriage

Control Carriage and Branch

Instruction Format.

Mnemonic	*Op Code*	*I-address*	*d-character*
SPS CC	F	III	d
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.

Without indexing: T = 6N ms*
With indexing: T = 7N 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.*
No Branch:	NSI	BI	d*bb*
Branch (without indexing):	NSI	BI	blank
Branch (with indexing):	NSI	BI	NSI

Example. Skip to channel 1 immediately, and branch to START3 (0498) for the next instruction (Figure E-31).

Instruction	Type	Mnemonics SPS	Autocoder		Format			Page
Add (One Field)		A	A	A	AAA			B - 4
Add (Two Fields)		A	A	A	AAA	BBB		B - 3
Backspace Tape	1012	SS	SS	K	A			F - 17
Backspace Tape and Branch	1012	SS	SSB	K	III	A		F - 17
Backspace Tape Record	Tape	CU	BSP	U	%Un	B		F - 3
Branch (Unconditional)		B	B	B	III			B - 9
Branch if Access Inoperable	Disk	B	BIN	B	III	N		G - 22, 37
Branch if Any Disk Condition	Disk	B	BIN	B	III	Y		G - 22, 38
Branch if Auto Select	1231	B	BIN	B	III	1		H - 56
Branch if Bit Equal	Bit	BBE	BBE	W	III	BBB	d	I - 7
Branch if Carriage Channel 9	1403	B	BC9	B	III	9		E - 13
Branch if Carriage Channel 12	1403	B	BCV	B	III	@		E - 13
Branch if Character Equal		B	BCE	B	III	BBB	d	B - 10
Branch if Early Warning	1448	B	BIN	B	III	<		H - 53
Branch if Empty Hopper	1231	B	BIN	B	III	4		H - 56
Branch if End of Block	1448	B	BIN	B	III	>		H - 52
Branch if End of File	1285	B	BIN	B	III	8		H - 63
Branch if End of Line	1285	B	BIN	B	III	2		H - 63
Branch if End of Reel	Tape	B	BEF	B	III	K		F - 6
Branch if Equal		B	BE	B	III	S		I - 23
Branch if Error	1285	B	BIN	B	III	1		H - 63
Branch if Full Buffer	1231	B	BIN	B	III	2		H - 56
Branch if Header Information	1285	B	BIN	B	III	5		H - 63
Branch if High		B	BH	B	III	U		I - 23
Branch if I/O Channel-Busy Indicator On	1419	B	BIN	B	III	J		H - 45
Branch if in Backspace Operation	1012	B	BIN	B	III	I		F - 18
Branch if Indicator On		B	BIN	B	III	d		B - 9
Branch if Indicator On	DDC	B	BIN	B	III	d		I - 14
Branch if Indicator On	Ovlp	B	BIN	B	III	d		I - 39
Branch if Indicator On	SS	B	BIN	B	III	d		I - 48
Branch if Indicator On	1009	B	BIN	B	III	d		I - 66
Branch if Indicator On	1009	B	BIN	B	III	d		H - 5
Branch if Indicator On	1285	B	BIN	B	III	d		H - 62
Branch if Indicator On	1301	B	BIN	B	III	d		G - 37
Branch if Indicator On	1311	B	BIN	B	III	d		G - 22
Branch if Indicator On	1404	B	BIN	B	III	0		H - 9
Branch if Indicator On	1405	B	BIN	B	III	d		G - 5
Branch if Indicator On	1407	B	BIN	B	III	d		H - 12
Branch if Indicator On	1418	B	BIN	B	III	d		H - 14
Branch if Indicator On	1428	B	BIN	B	III	d		H - 18
Branch if Indicator On	7740	B	BIN	B	III	d		H - 67
Branch if Input-Output Indicator On	1011	B	BIN	B	III	1		F - 15
Branch if Inquiry Clear		B	BIN	B	III	*		B - 9
Branch if Inquiry Request		B	BIN	B	III	Q		B - 9
Branch if Last Card (Switch)	1402	B	BLC	B	III	A		E - 6
Branch if Low		B	BL	B	III	T		I - 23
Branch if Magnetic Character Reader, Account-Number Field Indicator On	1412	B	BIN	B	III	6		H - 31
Branch if Magnetic Character Reader, Document-Spacing Check Indicator On	1412	B	BIN	B	III	8		H - 32
Branch if Magnetic Character Reader, Process-Control Field Indicator On	1412	B	BIN	B	III	5		H - 30
Branch if Magnetic Character Reader, Reader-Not-Ready Indicator On	1412	B	BIN	B	III	2		H - 29
Branch if Magnetic Character Reader, Read Check Indicator On	1412	B	BIN	B	III	3		H - 29
Branch if Magnetic Character Reader, Transit-Routing Field Indicator On	1412	B	BIN	B	III	7		M - 31
Branch if Magnetic Character Reader, Late Read Indicator On	1412	B	BIN	B	III	1		H - 28
Branch if Magnetic Character Reader, Amount-Field Indicator On	1412	B	BIN	B	III	4		H - 30
Branch if Marked Line	1285	B	BIN	B	III	4		H - 63
Branch if Overflow		B	BAV	B	III	Z		B - 9
Branch if Paper Tape Reader Ready	1011	B	BIN	B	III	2		F - 15

Instruction	Type	Mnemonics SPS	Autocoder		Format		Page
Branch if Printer Busy	1403	B	BPB	B	III	P	B - 9
Branch if Printer Carriage Busy	1403	B	BPCB	B	III	R	B - 9
Branch if Printer Error (I/O Check Switch Off)	1403	B	BIN	B	III	≠	B - 9
Branch if Printer Status	1445	B	BIN	B	III	D	H - 64
Branch if Processing Check (Process Check Switch Off)		B	BIN	B	III	%	B - 9
Branch if Punch Error (I/O Check Stop Switch Off)	1402	B	BIN	B	III	!	E - 6
Branch if Read-Back Check Error		B	BIN	B	III	V	B - 9
Branch if Read Error and Overrun Condition	1231	B	BIN	B	III	5	H - 5
Branch if Read-Write Parity Check		B	BIN	B	III	V	B - 9
Branch if Reader Error (I/O Check Stop Switch Off)		B	BIN	B	III	?	B - 9
Branch if Reader Ready	1285	B	BIN	B	III	7	H - 63
Branch if Reader Transporting	1285	B	BIN	B	III	3	H - 63
Branch if Reader Ready to Read a Line	1285	B	BIN	B	III	6	H - 63
Branch if Sense Switch A (Last Card)	1402	B	BLC	B	III	A	E - 8
Branch if Sense Switch B, C, D, E, F, G		B	BIS	B	III	B thru G	B - 9
Branch if Supply Reel Low or Chad Box Full	1012	B	BIN	B	III	6	F - 20
Branch if Tape Error	Tape	B	BER	B	III	L	F - 6
Branch if Tape Punch Is Not Ready to Read	1012	B	BIN	B	III	4	F - 19
Branch if Tape Punch Not Ready to Accept Data	1012	B	BIN	B	III	3	F - 19
Branch if Tape Punch Overextended	1012	B	BIN	B	III	5	F - 20
Branch if Tape Punch Ready	1012	B	BIN	B	III	2	F - 18
Branch if Tape Transmission Error		B	BER	B	III	L	B - 9
Branch if Timing Mark Check	1231	B	BIN	B	III	6	H - 57
Branch if Unconditonal		B	B	B	III		B - 9
Branch if Unequal Address Compare		B	BIN	B	III	X	B - 9
Branch if Unequal Compare		B	BU	B	III	/	I - 23
Branch if Word Mark or Zone		BWZ	BWZ	V	III	BBB d	B - 10
Branch if Wrong-Length Record		B	BIN	B	III	W	B - 9
Branch if 1231 Ready to Read	1231	B	BIN	B	III	3	H - 56
Branch if 1419 Indicator On	1419	B	BIN	B	III	d	H - 41
Clear Storage		CS	CS	/	AAA		B - 16
Clear Storage and Branch		CS	CS	/	AAA	BBB	B - 16
Clear Word Mark (One Address)		CW	CW	□	AAA		B - 17
Clear Word Mark (Two Addresses)		CW	CW	□	AAA	BBB	B - 17
Compare		C	C	C	AAA	BBB	B - 11
Compare	1404	C	C	C	AAA	363	H - 9
Control Carriage	1403	CC	CC	F	d		E - 13
Control Carriage	1404	CC	CC	F	d		H - 8
Carriage Control	1445	SS	SS	K	d		H-64, H - 65
Control Carriage	1445	CC	CC	F	d		H - 65
Control Carriage and Branch	1403	CC	CCB	F	III	d	E - 14
Control Unit	1428	CU	CU	U	%S2	d	H - 16
Diagnostic Read	Tape	CU	CU	U	%Un	A	F - 5
Diagnostic Read	Tape	CU	CU	U	%Bn	A	F - 5
Disable Interrupt	1448	SS	SS	K	<		H - 51
Disable Interrupt and Branch	1448	SS	SSB	K	III	<	H - 52
Disengage Magnetic Character Reader	1412	CU	CU	U	%S1	D	H - 26
Divide	M-D	D	D	%	AAA	BBB	I - 25
Enable Interrupt	1448	SS	SS	K	>		H - 50
Enable Interrupt and Branch	1448	SS	SSB	K	III	>	H - 52
Engage Magnetic Character Reader	1412	CU	CU	U	%S1	E	H - 26

READER'S COMMENT FORM

System Operation Reference Manual
IBM 1401 Data Processing System
IBM 1460 Data Processing System A24-3067-2

- Your comments, accompanied by answers to the following questions, help us produce better publications for your use. If your answer to a question is "No" or requires qualification, please explain in the space provided below. Comments and suggestions become the property of IBM.

	Yes	No
• Does this publication meet your needs?	☐	☐
• Did you find the material:		
Easy to read and understand?	☐	☐
Organized for convenient use?	☐	☐
Complete?	☐	☐
Well illustrated?	☐	☐
Written for your technical level?	☐	☐

- What is your occupation? _____
- How do you use this publication?

As an introduction to the subject?	☐	As an instructor in a class?	☐
For advanced knowledge of the subject?	☐	As a student in a class?	☐
For information about operating procedures?	☐	As a reference manual?	☐

Other _____

- Please give specific page and line references with your comments when appropriate. If you wish a reply, be sure to include your name and address.

COMMENTS:

- Thank you for your cooperation. No postage necessary if mailed in the U. S. A.

A24-3067-2

Staple

Fold

Fold

FIRST CLASS
PERMIT NO. 170
ENDICOTT, N. Y.

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY . . .

IBM Corporation
P. O. Box 6
Endicott, N. Y. 13760

Attention: Product Publications, Dept. 171

Fold

Fold

Cut Along Line

IBM 1401, 1460

Printed in U. S. A.

A24-3067-2

IBM®

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

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

Additional Comments:

IBM 1401, 1460 Printed in U. S. A. A24-3067-2

IBM®

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

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